Dr. Budi Raharjo, S.Kom, M.Kom, MM.

PENGEMBANGAN WEB PHP dan MySQL

YAYASAN PRIMA AGUS TEKNIK

PENGEMBANGAN WEB PHP dan MySQL

Dr. Budi Raharjo, S.Kom, M.Kom, MM.

BIODATA PENULIS

Dr. Budi Raharjo, S.Kom, M.Kom, MM lahir di Semarang, tanggal 22 Februari 1985. Beliau adalah Alumni dari Universitas Bina Nusantara (BINUS University) Jakarta dan juga alumni Universitas Kristen Satya wacana (UKSW) Salatiga. Dr. Budi Raharjo telah menjadi Dosen pada Universitas STEKOM pada mata kuliah Kepemimpinan (Leadership), mata kuliah Pengantar Akuntansi, Manajemen Proses, Manajemen Akuntansi dan Manajemen Resiko Bisnis. Selain sebagai dosen Universitas STEKOM, Dr. Budi Raharjo, M.Kom, MM juga mempunyai bisnis sendiri dalam bidang perhotelan dan juga sebagai wirausaha dalam bidang pemasok unggas (ayam) beku, ke berbagai kota besar, khususnya Jakarta dan sekitarnya.

Pengalaman beliau berwirausaha menjadi bekal utama dalam penulisan buku ajar yang diterbitkan oleh Yayasan Prima Agus Teknik (YPAT) Semarang. Oleh sebab itu bukunya berisi langkah langkah praktis yang mudah diikuti oleh para mahasiswa, saat mahasiswa mengikuti proses perkuliahan pada Universitas Sains dan Teknologi Komputer (Universitas STEKOM). Memiliki Jabatan Akademik Lektor 300 dan Menjabat sebagai Wakil Rektor 1 bidang (Akademik) di kampus Universitas STEKOM Semarang.

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-634-7227-25-6 (PDF)
9 786347 227256

PENGEMBANGAN WEB PHP DAN MySQL

Penulis :
Dr. Budi Raharjo, S.Kom, M.Kom, MM.

ISBN : 978-634-7227-25-6

Editor :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Penyunting :
Dr. Joseph Teguh Santoso, M.Kom.

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom.

Penebit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

Anggota IKAPI No: 279 / ALB / JTE / 2023

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa, karena atas rahmat dan karunia-Nya, buku berjudul **"Pengembangan Web PHP dan MySQL"** ini dapat diselesaikan. Buku ini hadir sebagai panduan komprehensif bagi para pengembang web, baik pemula maupun yang telah berpengalaman, yang ingin memahami dan menerapkan teknologi PHP dan MySQL dalam proyek pengembangan web berskala kecil hingga besar.

Seiring berkembangnya teknologi informasi, kebutuhan akan aplikasi web yang andal, efisien, dan aman semakin meningkat. Melalui buku ini, penulis berupaya menjembatani kebutuhan tersebut dengan menyajikan materi mulai dari konsep dasar hingga penerapan lanjutan dalam pengembangan web. Setiap bab disusun secara sistematis dan praktis agar pembaca dapat dengan mudah memahami tahapan pengembangan aplikasi web berbasis PHP dan MySQL.

Bab pertama menjelaskan fundamental penggunaan PHP dan MySQL untuk membangun proyek besar, termasuk praktik rekayasa perangkat lunak, penulisan kode yang terstruktur, serta pentingnya dokumentasi dan optimasi. Dalam bab kedua, pembaca akan dibimbing untuk memahami teknik debugging secara mendalam, yang sangat penting dalam proses pengembangan agar aplikasi berjalan tanpa kesalahan.

Pada bab-bab berikutnya, buku ini membahas implementasi fitur penting seperti autentikasi dan personalisasi pengguna, pembuatan keranjang belanja untuk toko daring, pengelolaan konten melalui sistem manajemen konten (CMS), hingga pembangunan layanan email dan forum web. Semua topik disertai contoh-contoh penerapan praktis yang dapat menjadi referensi langsung ketika membangun aplikasi nyata.

Tidak hanya itu, buku ini juga mengupas metode pembuatan dokumen dalam format PDF maupun RTF secara dinamis, fitur yang semakin diperlukan dalam berbagai kebutuhan digital saat ini. Dengan uraian materi yang lengkap dan dilengkapi daftar pustaka, penulis berharap buku ini dapat menjadi sumber literasi yang bermanfaat untuk mengembangkan kemampuan teknis serta memperluas wawasan Anda dalam bidang pengembangan web.

Akhir kata, penulis mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan, saran, dan kritik yang membangun selama proses penulisan buku ini. Semoga buku ini dapat memberikan kontribusi nyata bagi pembaca dalam mengembangkan aplikasi web modern yang inovatif dan profesional.

Selamat belajar dan berkarya!

Semarang, Juli 2025
Penulis

Dr. Budi Raharjo, S.Kom, M.Kom, MM

DAFTAR ISI

BAB 1
MENGGUNAKAN PHP DAN MYSQL UNTUK PROYEK BESAR

Pada bagian awal buku ini, kami telah membahas berbagai komponen dan penggunaan PHP dan MySQL. Meskipun kami telah mencoba membuat semua contoh kami menarik dan relevan, contoh-contoh tersebut cukup sederhana, yang terdiri dari satu atau dua skrip hingga sekitar 100 baris kode. Saat Anda membangun aplikasi Web di dunia nyata, hal-hal jarang sesederhana ini. Beberapa tahun yang lalu, ada saat ketika situs Web "interaktif" memiliki formulir surat dan hanya itu saja.

Namun, saat ini, situs Web telah menjadi aplikasi Web yaitu, perangkat lunak biasa yang dikirimkan melalui Web. Perubahan fokus ini berarti perubahan skala. Situs Web berkembang dari beberapa skrip menjadi ribuan dan ribuan baris kode. Proyek sebesar ini memerlukan perencanaan dan pengelolaan seperti pengembangan perangkat lunak lainnya. Sebelum kita beralih untuk melihat proyek-proyek di bagian buku ini, kita akan melihat beberapa teknik yang dapat Anda gunakan untuk mengelola proyek Web yang cukup besar. Ini adalah seni yang sedang berkembang dan jelas sulit untuk melakukannya dengan benar: Anda dapat melihatnya melalui pengamatan di pasar.

1.1 MENERAPKAN REKAYASA PERANGKAT LUNAK PADA PENGEMBANGAN WEB

Seperti yang mungkin sudah Anda ketahui, rekayasa perangkat lunak adalah penerapan pendekatan yang sistematis dan terukur pada pengembangan perangkat lunak. Yaitu, penerapan prinsip-prinsip rekayasa pada pengembangan perangkat lunak. Ini juga merupakan pendekatan yang sangat kurang dalam banyak proyek Web. Ini karena dua alasan utama:

- Alasan pertama adalah bahwa pengembangan Web sering kali mirip dengan pengembangan laporan tertulis. Ini adalah latihan dalam struktur dokumen, desain grafis, dan produksi. Ini adalah paradigma berorientasi dokumen. Ini semua baik dan bagus untuk situs statis berukuran kecil hingga sedang, tetapi seiring kita meningkatkan jumlah konten dinamis di situs Web ke tingkat di mana situs Web menawarkan layanan daripada dokumen, paradigma ini tidak lagi cocok. Banyak orang tidak berpikir untuk menggunakan praktik rekayasa perangkat lunak untuk sebuah proyek sama sekali.
- Alasan kedua mengapa praktik rekayasa perangkat lunak tidak digunakan adalah bahwa pengembangan aplikasi Web berbeda dari pengembangan aplikasi normal dalam banyak hal. Kita berhadapan dengan waktu tunggu yang jauh lebih singkat, tekanan konstan untuk membangun situs sekarang. Rekayasa perangkat lunak adalah tentang melakukan segala sesuatunya dengan tertib, terencana, dan meluangkan waktu untuk perencanaan.

Dengan proyek Web, sering kali persepsi yang ada adalah bahwa kita tidak punya waktu untuk merencanakan. Ketika kita gagal merencanakan proyek Web, kita berakhir dengan masalah yang sama seperti jika kita gagal merencanakan proyek perangkat lunak apa pun: aplikasi yang bermasalah, tenggat waktu yang terlewat, dan kode yang tidak dapat dibaca. Maka, triknya

adalah menemukan bagian-bagian rekayasa perangkat lunak yang berfungsi dalam disiplin baru pengembangan aplikasi web ini, dan membuang bagian-bagian yang tidak berfungsi.

Merencanakan dan Menjalankan Proyek Aplikasi Web

Tidak ada metodologi atau siklus hidup proyek terbaik untuk proyek Web. Akan tetapi, ada sejumlah hal yang harus Anda pertimbangkan untuk dilakukan untuk proyek Anda. Kami akan mencantumkannya di sini, dan membahas beberapa di antaranya secara lebih rinci di bagian berikut. Pertimbangan ini disusun dalam urutan tertentu, tetapi Anda tidak harus mengikuti urutan ini jika tidak sesuai dengan proyek Anda. Penekanannya di sini adalah pada kesadaran akan masalah dan pemilihan teknik yang sesuai untuk Anda.

- ☑ Sebelum memulai, pikirkan tentang apa yang ingin Anda bangun. Pikirkan tentang tujuan akhirnya. Pikirkan tentang siapa yang akan menggunakan aplikasi Web Anda; yaitu, audiens target Anda. Banyak proyek Web yang secara teknis sempurna gagal karena tidak ada yang memeriksa apakah ada pengguna yang tertarik untuk aplikasi tersebut.
- ☑ Cobalah untuk memecah aplikasi Anda menjadi beberapa komponen. Bagian atau langkah proses apa yang dimiliki aplikasi Anda? Bagaimana masing-masing komponen tersebut bekerja? Bagaimana mereka akan saling melengkapi? Menyusun skenario, storyboard, atau bahkan kasus penggunaan dapat berguna untuk mencari tahu hal ini.
- ☑ Setelah Anda memiliki daftar komponen, lihat komponen mana yang sudah ada. Jika modul yang sudah ditulis sebelumnya memiliki fungsi tersebut, pertimbangkan untuk menggunakannya. Jangan lupa untuk mencari kode yang sudah ada di dalam dan di luar organisasi Anda. Khususnya di komunitas Open Source, banyak komponen kode yang sudah ada sebelumnya tersedia secara gratis untuk digunakan. Putuskan kode apa yang harus Anda tulis dari awal dan seberapa besar pekerjaan itu.
- ☑ Buat keputusan tentang masalah proses. Hal ini terlalu sering diabaikan dalam proyek Web. Yang saya maksud dengan masalah proses adalah hal-hal seperti standar pengodean, struktur direktori, manajemen kontrol versi, lingkungan pengembangan, tingkat dan standar dokumentasi, serta alokasi tugas kepada anggota tim.
- ☑ Bangun prototipe berdasarkan semua informasi sebelumnya. Tunjukkan kepada pengguna. Ulangi.
- ☑ Ingatlah bahwa, dalam semua ini, penting dan bermanfaat untuk memisahkan konten dan logika dalam aplikasi Anda. Kami akan menjelaskan ide ini secara lebih rinci sebentar lagi.
- ☑ Lakukan pengoptimalan yang menurut Anda perlu.
- ☑ Saat Anda melakukannya, ujilah secara menyeluruh seperti yang Anda lakukan pada proyek pengembangan perangkat lunak lainnya.

Menggunakan Kembali Kode

Programmer sering membuat kesalahan dengan menulis ulang kode yang sudah ada. Ketika Anda mengetahui komponen aplikasi apa yang Anda butuhkan, atau dalam skala yang lebih kecil, fungsi apa yang Anda butuhkan, periksa apa yang tersedia sebelum memulai

pengembangan. Terkadang programmer menulis ulang fungsi secara tidak sengaja karena mereka belum melihat manual untuk melihat apakah fungsi yang ada menyediakan fungsionalitas yang mereka butuhkan. Selalu simpan manual sebagai bookmark jika Anda sedang online, atau unduh versi saat ini dan telusuri secara lokal. Namun, perlu dicatat bahwa manual online diperbarui cukup sering, dan Anda juga memiliki keuntungan karena dapat menelusuri manual yang diberi anotasi.

Manual beranotasi merupakan sumber yang fantastis karena berisi komentar, saran, dan contoh kode dari pengguna lain yang sering menjawab pertanyaan yang sama yang mungkin Anda miliki setelah membaca halaman manual dasar. Anda dapat mengaksesnya di http://www.php.net/manual/ Beberapa programmer yang berasal dari latar belakang bahasa yang berbeda mungkin tergoda untuk menulis fungsi wrapper untuk mengganti nama fungsi PHP agar sesuai dengan bahasa yang mereka kenal.

Praktik ini terkadang disebut "*syntactic sugar*". Ini adalah ide yang buruk ini akan membuat kode Anda lebih sulit dibaca dan dikelola orang lain. Jika Anda mempelajari bahasa baru, Anda harus mempelajari cara menggunakannya dengan benar. Selain itu, menambahkan level pemanggilan fungsi dengan cara ini akan memperlambat kode Anda. Jika mempertimbangkan semua hal, ini adalah pendekatan yang harus dihindari. Jika Anda menemukan bahwa fungsionalitas yang Anda perlukan tidak ada di pustaka PHP utama, Anda memiliki dua pilihan. Jika Anda membutuhkan sesuatu yang cukup sederhana, Anda dapat memilih untuk menulis fungsi atau objek Anda sendiri.

Namun, jika Anda ingin membangun fungsionalitas yang cukup rumit seperti keranjang belanja, sistem email Web, atau forum Web Anda tidak akan terkejut mengetahui bahwa fungsionalitas tersebut mungkin sudah dibuat oleh orang lain. Salah satu kelebihan bekerja di komunitas Open Source adalah kode untuk komponen aplikasi seperti ini sering kali tersedia secara gratis. Jika Anda menemukan komponen yang mirip dengan yang ingin Anda bangun, meskipun tidak sepenuhnya benar, Anda dapat melihat kode sumbernya sebagai titik awal untuk modifikasi atau membangun komponen Anda sendiri. Jika Anda akhirnya mengembangkan fungsi atau komponen Anda sendiri, Anda harus mempertimbangkan untuk menyediakannya bagi komunitas PHP setelah selesai. Prinsip inilah yang membuat komunitas pengembang PHP menjadi kelompok yang membantu, aktif, dan berpengetahuan luas.

1.2 MENULIS KODE YANG DAPAT DIPELIHARA

Masalah pemeliharaan sering kali diabaikan dalam aplikasi Web, terutama karena kita sering menulisnya dengan tergesa-gesa. Memulai kode, dan menyelesaikannya dengan cepat terkadang tampak lebih penting daripada merencanakannya terlebih dahulu. Namun, sedikit waktu yang diinvestasikan di awal dapat menghemat banyak waktu di kemudian hari saat Anda siap membangun iterasi aplikasi berikutnya.

Standar Pengodean

Sebagian besar organisasi TI besar memiliki standar pengodean pedoman untuk gaya penulisan internal dalam memilih nama file dan variabel, pedoman untuk memberi komentar pada kode, pedoman untuk membuat indentasi kode, dan sebagainya. Karena paradigma

dokumen yang sebelumnya sering diterapkan pada pengembangan Web, standar pengodean terkadang diabaikan di area ini. Jika Anda membuat kode sendiri atau dalam tim kecil, mudah untuk meremehkan pentingnya standar pengodean. Jangan lakukan itu karena tim dan proyek Anda mungkin akan berkembang. Maka Anda tidak hanya akan berakhir dengan kekacauan di tangan Anda, tetapi juga sekelompok programmer yang tidak dapat memahami kode yang ada.

Konvensi Penamaan

Tujuan mendefinisikan konvensi penamaan adalah

- Agar kode mudah dibaca. Jika Anda mendefinisikan variabel dan nama fungsi dengan bijaksana, Anda seharusnya dapat membaca kode secara virtual seperti Anda membaca kalimat bahasa Inggris, atau setidaknya pseudocode.
- Agar nama pengenal mudah diingat. Jika pengenal Anda diformat secara konsisten, akan lebih mudah untuk mengingat apa yang Anda sebut sebagai variabel atau fungsi tertentu.

Nama variabel harus menggambarkan data yang dikandungnya. Jika Anda menyimpan nama belakang seseorang, sebut saja `$surname`. Anda perlu menemukan keseimbangan antara panjang dan keterbacaan. Misalnya, menyimpan nama dalam `$n` akan memudahkan pengetikan, tetapi kodenya akan sulit dipahami.

Menyimpan nama dalam `$surname_of_the_current_user` lebih informatif, tetapi banyak yang harus diketik (lebih mudah membuat kesalahan pengetikan) dan tidak benar-benar menambah banyak hal. Anda perlu membuat keputusan tentang kapitalisasi. Nama variabel peka huruf besar/kecil dalam PHP, seperti yang telah kami sebutkan sebelumnya. Anda perlu memutuskan apakah nama variabel Anda akan ditulis dengan huruf kecil semua, huruf besar semua, atau campuran, misalnya, huruf pertama kata ditulis dengan huruf kapital. Kami cenderung menggunakan semua huruf kecil, karena itu yang paling mudah diingat.

Membedakan variabel dan konstanta dengan huruf besar juga merupakan ide yang bagus skema yang umum adalah menggunakan semua huruf kecil untuk variabel (misalnya, `$result`) dan semua huruf besar untuk konstanta (misalnya, `PI`). Salah satu praktik buruk yang digunakan oleh beberapa programmer adalah memiliki dua variabel dengan nama yang sama tetapi kapitalisasi yang berbeda, hanya karena bisa, seperti `$name` dan `$Name` misalnya. Saya harap jelas mengapa ini merupakan ide yang buruk. Sebaiknya juga hindari skema kapitalisasi yang lucu seperti `$WaReZ` karena tidak seorang pun akan dapat mengingat cara kerjanya. Anda juga harus memikirkan skema apa yang akan digunakan untuk nama variabel multikata. Misalnya:

```
$username
$user_name
$UserName
```

adalah semua skema yang pernah saya lihat digunakan. Tidak masalah yang mana yang Anda pilih, tetapi Anda harus mencoba untuk konsisten tentang hal ini. Anda mungkin juga ingin menetapkan batas maksimum yang masuk akal yaitu dua hingga tiga kata dalam nama

variabel. Nama fungsi memiliki banyak pertimbangan yang sama, dengan beberapa tambahan. Nama fungsi pada umumnya harus berorientasi pada kata kerja.

Pertimbangkan fungsi bawaan PHP seperti `addslashes()` atau `mysql_connect()`, yang menjelaskan apa yang akan dilakukannya terhadap atau dengan parameter yang diteruskannya. Ini sangat meningkatkan keterbacaan kode. Perhatikan bahwa kedua fungsi ini memiliki skema penamaan yang berbeda untuk menangani nama fungsi yang terdiri dari banyak kata. Fungsi PHP tidak konsisten dalam hal ini, mungkin sebagian karena ditulis oleh sekelompok besar orang. Ingat juga bahwa nama fungsi tidak peka huruf besar-kecil dalam PHP. Anda mungkin harus tetap menggunakan format tertentu, hanya untuk menghindari kebingungan.

Anda mungkin ingin mempertimbangkan untuk menggunakan skema penamaan modul yang digunakan dalam banyak modul PHP, yaitu, mengawali nama fungsi dengan nama modul. Misalnya, semua fungsi MySQL dimulai dengan `mysql_`, dan semua fungsi IMAP dimulai dengan `imap_`. Jika, misalnya, Anda memiliki modul keranjang belanja dalam kode Anda, Anda dapat mengawali fungsi dalam modul itu dengan cart. Pada akhirnya, tidak masalah konvensi dan standar apa yang Anda gunakan saat menulis kode, selama beberapa pedoman yang konsisten diterapkan.

Mengomentari Kode Anda

Semua program harus dikomentari pada tingkat yang wajar. Anda mungkin bertanya tingkat komentar seperti apa yang wajar. Umumnya Anda harus mempertimbangkan untuk menambahkan komentar pada setiap item berikut:

- File, baik skrip lengkap atau file yang disertakan. Setiap file harus memiliki komentar yang menyatakan jenis file ini, kegunaannya, siapa yang menulisnya, dan kapan diperbarui.
- Fungsi. Komentar fungsi harus menentukan apa yang dilakukan fungsi, input apa yang diharapkan, dan apa yang dikembalikannya.
- Kelas. Komentar harus menjelaskan tujuan kelas. Metode kelas harus memiliki jenis dan tingkat komentar yang sama seperti fungsi lainnya.
- Potongan kode dalam skrip atau fungsi. Saya sering merasa berguna untuk menulis skrip dengan memulai dengan serangkaian komentar gaya pseudocode lalu mengisi kode untuk setiap bagian. Jadi, skrip awal mungkin menyerupai ini:

```
<?
 // validate input data
 // send to database
 // report results
 ?>
```

Ini cukup berguna karena setelah Anda mengisi semua bagian dengan fungsi pemanggilan atau apa pun, kode Anda sudah diberi komentar.

- Kode atau trik yang rumit. Jika Anda membutuhkan waktu seharian untuk melakukan sesuatu, atau Anda harus melakukannya dengan cara yang aneh, tulis komentar yang menjelaskan alasan Anda melakukannya dengan cara itu. Dengan cara ini, saat Anda melihat kode tersebut lagi, Anda tidak akan bingung dan berpikir, "Apa sih yang seharusnya dilakukan?"

Pedoman umum lain yang harus diikuti adalah Anda harus memberi komentar saat mengerjakannya. Anda mungkin berpikir Anda akan kembali dan memberi komentar pada kode Anda saat Anda selesai mengerjakan proyek. Saya jamin ini tidak akan terjadi, kecuali jika Anda memiliki jadwal pengembangan yang jauh lebih sedikit dan disiplin diri yang lebih tinggi daripada kami.

Indentasi

Seperti dalam bahasa pemrograman apa pun, Anda harus membuat indentasi kode dengan cara yang masuk akal dan konsisten. Ini seperti membuat resume atau surat bisnis. Indentasi membuat kode Anda lebih mudah dibaca dan lebih cepat dipahami. Secara umum, setiap blok program yang termasuk dalam struktur kontrol harus diberi indentasi dari kode di sekitarnya. Tingkat indentasi harus terlihat (dengan kata lain, lebih dari satu spasi) tetapi tidak berlebihan. Saya biasanya berpendapat bahwa penggunaan tab harus dihindari. Meskipun mudah diketik, tab menghabiskan banyak ruang layar pada monitor banyak orang. Kami menggunakan tingkat indentasi dua hingga tiga spasi untuk semua proyek.

Cara Anda menata kurung kurawal juga menjadi masalah. Dua skema yang paling umum adalah sebagai berikut:

Skema 1:

```
if (condition) {
   // do something
 }
```

Skema 2:

```
if (condition)
 {
   // do something else
 }
```

Terserah Anda untuk menggunakan yang mana. Skema yang Anda pilih harus (sekali lagi) digunakan secara konsisten di seluruh proyek untuk menghindari kebingungan.

Membagi Kode

Kode monolitik yang sangat besar itu buruk. Beberapa orang akan memiliki satu skrip besar yang melakukan semuanya dalam satu baris kode utama. Jauh lebih baik untuk membagi kode menjadi fungsi dan/atau kelas dan memasukkan item terkait ke dalam file include. Misalnya, Anda dapat memasukkan semua fungsi terkait basis data ke dalam file bernama `dbfunctions.php`.

Alasan untuk membagi kode menjadi potongan-potongan yang masuk akal meliputi hal-hal berikut:

- ☑ Membuat kode lebih mudah dibaca dan dipahami.
- ☑ Membuat kode lebih dapat digunakan kembali dan meminimalkan redundansi. Misalnya, dengan file `dbfunctions.php` sebelumnya, Anda dapat menggunakannya kembali di setiap skrip yang perlu Anda hubungkan ke basis data. Jika Anda perlu mengubah cara kerjanya, Anda harus mengubahnya hanya di satu tempat.
- ☑ Memfasilitasi kerja tim. Jika kode dipecah menjadi beberapa komponen, Anda dapat menetapkan tanggung jawab atas komponen tersebut kepada anggota tim. Ini juga berarti Anda dapat menghindari situasi di mana seorang programmer menunggu programmer lain menyelesaikan `GiantScript.php` sehingga ia dapat melanjutkan pekerjaannya sendiri.

Di awal proyek, Anda harus meluangkan waktu untuk memikirkan cara memecah proyek menjadi beberapa komponen yang direncanakan. Ini mengharuskan Anda untuk membuat batasan antara area fungsionalitas, tetapi jangan sampai Anda terjebak di dalamnya karena hal ini dapat berubah setelah Anda memulai proyek. Anda juga perlu memutuskan komponen mana yang perlu dibangun terlebih dahulu, komponen mana yang bergantung pada komponen lain, dan jangka waktu untuk mengembangkan semuanya.

Bahkan jika semua anggota tim akan mengerjakan semua bagian kode, sebaiknya tetapkan tanggung jawab utama untuk setiap komponen kepada orang tertentu. Pada akhirnya, orang inilah yang akan bertanggung jawab jika terjadi kesalahan pada komponennya. Seseorang juga harus mengambil alih tugas sebagai manajer pembuatan yaitu, orang yang memastikan bahwa semua komponen berjalan sesuai rencana dan bekerja dengan komponen lainnya. Orang ini biasanya juga mengelola kontrol versi kita akan membahasnya lebih lanjut nanti di bab ini. Orang ini dapat menjadi manajer proyek, atau dapat ditugaskan sebagai tanggung jawab terpisah.

1.3 MENGGUNAKAN STRUKTUR DIREKTORI STANDAR

Saat memulai proyek, Anda perlu memikirkan bagaimana struktur komponen Anda akan tercermin dalam struktur direktori situs Web Anda. Sama seperti ide yang buruk untuk memiliki satu skrip raksasa yang memuat semua fungsi, biasanya juga merupakan ide yang buruk untuk memiliki satu direktori raksasa yang memuat semuanya. Putuskan bagaimana Anda akan membaginya antara komponen, logika, konten, dan pustaka kode bersama. Dokumentasikan struktur Anda dan pastikan bahwa setiap orang yang mengerjakan proyek

memiliki salinannya sehingga mereka dapat menemukan berbagai hal. Ini mengarah ke poin berikutnya.

Mendokumentasikan dan Membagikan Fungsi Internal

Saat Anda mengembangkan pustaka fungsi, sediakan pustaka tersebut untuk programmer lain di tim Anda. Umumnya, setiap programmer dalam tim menulis kumpulan fungsi basis data, tanggal, atau debugging miliknya sendiri. Ini membuang-buang waktu. Sediakan fungsi dan kelas untuk orang lain. Ingatlah bahwa meskipun kode disimpan di area atau direktori yang umum tersedia untuk tim Anda, mereka tidak akan tahu keberadaannya kecuali Anda memberi tahu mereka. Kembangkan sistem untuk mendokumentasikan pustaka fungsi internal, dan sediakan pustaka tersebut untuk programmer di tim Anda.

Menerapkan Kontrol Versi

Kontrol versi adalah seni manajemen perubahan bersamaan sebagaimana diterapkan pada pengembangan perangkat lunak. Sistem kontrol versi umumnya bertindak sebagai repositori atau arsip pusat dan menyediakan antarmuka terkendali untuk mengakses dan berbagi kode Anda (dan mungkin dokumentasi). Bayangkan situasi di mana Anda mencoba memperbaiki beberapa kode, tetapi malah merusaknya secara tidak sengaja dan tidak dapat mengembalikannya ke keadaan semula, tidak peduli seberapa keras Anda mencoba. Atau, Anda atau klien memutuskan bahwa versi situs sebelumnya lebih baik. Atau, Anda perlu kembali ke versi sebelumnya karena alasan hukum.

Bayangkan situasi lain di mana dua anggota tim pemrograman Anda ingin mengerjakan berkas yang sama. Mereka berdua mungkin membuka dan mengedit berkas tersebut pada saat yang sama, saling menimpa perubahan satu sama lain. Mereka berdua mungkin memiliki salinan yang mereka kerjakan secara lokal dan mengubahnya dengan cara yang berbeda. Jika Anda pernah membayangkan hal-hal ini terjadi, seorang programmer mungkin akan duduk-duduk tanpa melakukan apa pun sambil menunggu programmer lain selesai mengedit berkas. Anda dapat memecahkan semua masalah ini dengan sistem kontrol versi.

Sistem ini dapat melacak perubahan pada setiap berkas dalam repositori sehingga Anda tidak hanya dapat melihat status terkini berkas, tetapi juga bagaimana tampilannya pada waktu tertentu di masa lalu. Fitur ini memungkinkan Anda mengembalikan kode yang rusak ke versi yang berfungsi. Anda dapat menandai sekumpulan contoh berkas tertentu sebagai versi rilis, yang berarti Anda dapat melanjutkan pengembangan kode tetapi mendapatkan akses ke salinan versi yang saat ini dirilis kapan saja. Sistem kontrol versi juga membantu beberapa programmer dalam mengerjakan kode bersama-sama. Setiap programmer dapat memperoleh salinan kode dalam repositori (disebut memeriksanya) dan ketika mereka membuat perubahan, perubahan ini dapat digabungkan kembali ke repositori (diperiksa atau dikomit). Oleh karena itu, sistem kontrol versi dapat melacak siapa yang membuat setiap perubahan pada suatu sistem.

Sistem ini biasanya memiliki fasilitas untuk mengelola pembaruan bersamaan. Artinya, dua programmer sebenarnya dapat memodifikasi berkas yang sama pada waktu yang sama. Misalnya, bayangkan John dan Mary sama-sama telah memeriksa salinan rilis terbaru proyek mereka. John menyelesaikan perubahannya pada berkas tertentu dan memeriksanya. Mary

juga mengubah berkas itu, dan mencoba memeriksanya juga. Jika perubahan yang mereka buat tidak berada di bagian berkas yang sama, sistem kontrol versi akan menggabungkan dua versi berkas tersebut. Jika perubahan saling bertentangan, Mary akan diberi tahu dan diperlihatkan dua versi yang berbeda. Ia kemudian dapat menyesuaikan versi kodenya untuk menghindari konflik.

Sistem kontrol versi yang digunakan oleh mayoritas pengembang UNIX dan/atau Open Source adalah CVS, yang merupakan singkatan dari *Concurrent Versions System*. CVS adalah Open Source, disertakan dengan hampir setiap UNIX, dan Anda juga bisa mendapatkannya untuk PC yang menjalankan DOS atau Windows dan Mac. CVS mendukung model klien-server sehingga Anda dapat memeriksa kode masuk atau keluar dari mesin mana pun dengan koneksi Internet, dengan asumsi bahwa server CVS terlihat di internet.

CVS digunakan untuk pengembangan PHP, Apache, dan Mozilla, di antara proyek-proyek terkenal lainnya, setidaknya sebagian karena alasan ini. Anda dapat mengunduh CVS untuk sistem Anda dari beranda CVS di *http://www.cvshome.org/* Meskipun sistem dasar CVS adalah alat baris perintah, berbagai add-on memberikan tampilan depan yang lebih cantik, termasuk tampilan depan berbasis Java dan Windows. Ini juga dapat diakses dari beranda CVS. Ada alternatif komersial untuk CVS. Salah satunya adalah *Visual Source Safe* dari Microsoft, yang terintegrasi erat dengan produk Visual Studio mereka.dih

Memilih Lingkungan Pengembangan

Berbicara tentang kontrol versi memunculkan topik yang lebih umum tentang lingkungan pengembangan. Yang Anda perlukan hanyalah editor teks dan browser untuk pengujian, tetapi banyak pengguna PHP, terutama mereka yang terbiasa dengan lingkungan terintegrasi, telah menyatakan minat pada lingkungan pengembangan terintegrasi, atau IDE. Saat ini tidak ada IDE yang matang untuk programmer PHP.

Anda tentu saja dapat mengunduh atau menulis ekstensi penyorotan sintaksis Anda sendiri untuk editor yang ada atau rangkaian pengembangan Web yang ada. Banyak di antaranya tersedia dari berbagai tempat daring (terlalu banyak untuk dicantumkan di sini—cari menurut editor pilihan Anda). Ada sejumlah proyek yang muncul untuk membangun IDE PHP khusus. Pada saat penulisan, ini termasuk yang berikut:

- Zend IDE, untuk Windows atau UNIX, tersedia dari: http://www.zend.com/
  Sistem klien-server ini memiliki semua fitur editor normal, ditambah fasilitas debugging bawaan seperti pengawasan dan breakpoint. Memerlukan lisensi untuk server dan untuk setiap klien. Lisensi individual atau situs dapat dibeli daring.
- KPHPDevelop, untuk Lingkungan Desktop K di Linux, tersedia dari: http://kphpdev.sourceforge.net/
  IDE ini dirancang khusus untuk tim yang mengerjakan proyek bersama melalui mekanisme penguncian berkas. (Mereka berencana untuk mendukung CVS di versi mendatang.) IDE ini juga mendukung penyorotan sintaks dan akses basis data untuk MySQL, PostgreSQL, dan Sybase.
- PHPCoder, untuk platform Win32, tersedia dari: http://phpcoder.stsoft.cjb.net

Ini cukup bagus dan baru. Ini adalah lingkungan terintegrasi yang mendukung mode pratinjau skrip melalui penyambungan ke eksekusi PHP serta mengintegrasikan dokumentasi untuk kemudahan penggunaan. PHPCoder adalah perangkat lunak gratis.

- PHPEdit, untuk platform Win32, tersedia dari: http://www.phpedit.com
  Proyek ini belum dirilis ke publik pada saat penulisan, jadi saya tidak dapat mengomentarinya. Mereka berencana untuk mendukung semua fitur yang biasa dan memiliki manual bawaan.
- PHPGem, tersedia dari: http://phpgem.ru.net:8100/
  Dengan parameter basis data Anda, PHPGem akan menghasilkan kode antarmuka dasar, yang berfungsi dengan MySQL dan PostgreSQL, antara lain. PHPGem sendiri adalah skrip PHP, jadi ia akan berfungsi dengan lingkungan apa pun yang Anda miliki.

Tidak diragukan lagi, ada proyek lain yang sedang dikerjakan. Akan menarik untuk melihat apakah salah satu IDE ini muncul sebagai yang terdepan.

1.4 MENDOKUMENTASIKAN PROYEK ANDA

Anda dapat membuat berbagai jenis dokumentasi untuk proyek pemrograman Anda, termasuk, tetapi tidak terbatas pada yang berikut ini:

- ✓ Dokumentasi desain
- ✓ Dokumentasi teknis/panduan pengembang
- ✓ Kamus data (termasuk dokumentasi kelas)
- ✓ Panduan pengguna (meskipun sebagian besar aplikasi Web harus menjelaskan sendiri)

Tujuan kami di sini bukanlah untuk mengajarkan Anda cara menulis dokumentasi teknis, tetapi untuk menyarankan agar Anda mempermudah hidup Anda dengan mengotomatiskan sebagian dari proses tersebut.

Dalam beberapa bahasa, ada cara untuk membuat beberapa dokumen ini secara otomatis terutama dokumentasi teknis dan kamus data. Misalnya, `javadoc membuat pohon file HTML yang berisi prototipe dan deskripsi anggota kelas untuk program Java. Beberapa utilitas jenis ini tersedia untuk PHP. Beberapa di antaranya adalah:

- ☑ phpDoc, tersedia dari; http://sourceforge.net/projects/phpdoc Ini menyimpan dokumentasi dalam basis data MySQL. Perhatikan bahwa istilah phpDoc digunakan untuk menggambarkan beberapa proyek jenis ini, dan ini adalah salah satunya.
- ☑ PHPDocumentor, tersedia dari; http://phpdocu.sourceforge.net PHPDocumentor memberikan keluaran yang sangat mirip dengan javadoc dan tampaknya berfungsi dengan cukup baik.
- ☑ phpautodoc, tersedia dari; http://sourceforge.net/projects/phpautodoc/ Sekali lagi, phpautodoc menghasilkan keluaran yang mirip dengan javadoc.

Tempat yang bagus untuk mencari lebih banyak aplikasi jenis ini (dan komponen PHP secara umum) adalah di SourceForge: http://sourceforge.net SourceForge terutama digunakan oleh komunitas UNIX/Linux, tetapi ada juga banyak proyek untuk platform lain.

Pembuatan Prototipe

Pembuatan prototipe adalah siklus pengembangan yang umum digunakan untuk mengembangkan aplikasi Web. Prototipe adalah alat yang berguna untuk menentukan persyaratan pelanggan. Biasanya, prototipe adalah versi aplikasi yang disederhanakan dan berfungsi sebagian yang dapat digunakan dalam diskusi dengan klien dan sebagai dasar sistem akhir. Sering kali, beberapa iterasi atas prototipe menghasilkan aplikasi akhir. Keuntungan dari pendekatan ini adalah memungkinkan Anda bekerja sama dengan klien atau pengguna akhir untuk menghasilkan sistem yang akan mereka sukai dan memiliki rasa kepemilikan.

Agar dapat "merakit" prototipe dengan cepat, Anda memerlukan beberapa keterampilan dan alat khusus. Di sinilah pendekatan berbasis komponen bekerja dengan baik. Jika Anda memiliki akses ke serangkaian komponen yang sudah ada sebelumnya, baik internal maupun yang tersedia untuk umum, Anda akan dapat melakukannya dengan lebih cepat. Alat lain yang berguna untuk pengembangan prototipe yang cepat adalah templat. Kita akan membahasnya di bagian berikutnya. Ada dua masalah utama dengan menggunakan pendekatan pembuatan prototipe. Anda perlu menyadari masalah-masalah ini agar dapat menghindarinya dan menggunakan pendekatan ini secara maksimal. Masalah pertama adalah bahwa programmer sering kali merasa sulit untuk membuang kode yang telah mereka tulis karena satu dan lain alasan. Prototipe sering kali ditulis dengan cepat, dan dengan melihat ke belakang, Anda dapat melihat bahwa Anda belum membangun prototipe dengan cara yang optimal, atau bahkan mendekati optimal.

Bagian kode yang rumit dapat diperbaiki, tetapi jika struktur keseluruhannya salah, Anda akan mengalami masalah. Masalahnya adalah bahwa aplikasi Web sering kali dibangun di bawah tekanan waktu yang sangat besar dan mungkin tidak ada waktu untuk memperbaikinya. Anda kemudian terjebak dengan sistem yang dirancang dengan buruk dan sulit dipelihara. Anda dapat menghindarinya dengan melakukan sedikit perencanaan seperti yang telah kita bahas sebelumnya dalam bab ini. Ingat juga bahwa terkadang lebih mudah untuk membuang sesuatu dan memulai lagi daripada memperbaikinya. Meskipun ini mungkin tampak seperti sesuatu yang tidak sempat Anda lakukan, ini sering kali akan menyelamatkan Anda dari banyak kesulitan di kemudian hari.

Masalah kedua dengan pembuatan prototipe adalah bahwa suatu sistem dapat berakhir menjadi prototipe abadi. Setiap kali Anda merasa telah selesai, klien Anda akan menyarankan beberapa perbaikan atau fungsi tambahan atau pembaruan ke situs. Penambahan fitur ini dapat menghentikan Anda untuk menyetujui suatu proyek. Untuk menghindari masalah ini, buatlah rencana proyek dengan jumlah iterasi yang tetap, dan tanggal setelah itu tidak ada fungsi baru yang dapat ditambahkan tanpa perencanaan ulang, penganggaran, dan penjadwalan.

Memisahkan Logika dan Konten

Anda mungkin familier dengan ide menggunakan HTML untuk menggambarkan struktur dokumen Web, dan *cascading style sheet* (CSS) untuk menggambarkan tampilannya. Ide pemisahan presentasi dari konten ini dapat diperluas ke skrip. Secara umum, situs akan lebih mudah digunakan dan dirawat dalam jangka panjang jika Anda dapat memisahkan logika

dari konten dari presentasi. Ini bermuara pada pemisahan PHP dan HTML Anda. Untuk proyek sederhana dengan sejumlah kecil baris kode atau skrip, ini dapat menjadi lebih merepotkan daripada bermanfaat. Seiring dengan semakin besarnya proyek Anda, penting untuk menemukan cara memisahkan logika dan konten. Jika Anda tidak melakukannya, kode Anda akan semakin sulit dipelihara. Jika Anda atau pihak yang berwenang memutuskan untuk menerapkan desain baru ke situs Web Anda dan banyak HTML yang tertanam dalam kode Anda, mengubah desain akan menjadi mimpi buruk.

Tiga pendekatan dasar untuk memisahkan logika dan konten adalah sebagai berikut:

- Gunakan file include untuk menyimpan berbagai bagian konten. Ini adalah pendekatan yang sederhana, tetapi jika situs Anda sebagian besar statis, pendekatan ini dapat berfungsi dengan cukup baik.
- Gunakan fungsi atau kelas API dengan serangkaian fungsi anggota untuk memasukkan konten dinamis ke dalam templat halaman statis.
- Gunakan sistem templat. Sistem ini mengurai templat statis dan menggunakan ekspresi reguler untuk mengganti tag placeholder dengan data dinamis. Keuntungan utama dari hal ini adalah jika orang lain mendesain template Anda, seperti desainer grafis, ia tidak perlu tahu apa pun tentang kode PHP. Anda seharusnya dapat menggunakan template yang disediakan dengan modifikasi minimum.

Sejumlah sistem template tersedia. Mungkin yang paling populer adalah FastTemplate, tersedia dari; http://thewebmasters.net

1.5 MENGOPTIMALKAN KODE

Jika Anda tidak memiliki latar belakang pemrograman Web, pengoptimalan mungkin tampak sangat penting. Saat menggunakan PHP, sebagian besar waktu yang dihabiskan pengguna untuk menunggu aplikasi Web berasal dari waktu koneksi dan pengunduhan. Pengoptimalan kode Anda tidak akan banyak berpengaruh pada waktu-waktu tersebut.

Menggunakan Pengoptimalan Sederhana

Namun, ada beberapa pengoptimalan sederhana yang dapat Anda lakukan yang akan membuat perbedaan. Banyak di antaranya yang berkaitan dengan aplikasi yang mengintegrasikan basis data seperti MySQL dengan kode PHP Anda, dan beberapa di antaranya tercantum sebagai berikut:

(1) Mengurangi koneksi basis data. Menghubungkan ke basis data sering kali merupakan bagian paling lambat dari skrip apa pun. Anda dapat mengatasinya dengan menggunakan koneksi persisten.

(2) Mempercepat kueri basis data. Kurangi jumlah kueri yang Anda buat, dan pastikan kueri tersebut dioptimalkan. Dengan kueri yang rumit (dan karenanya lambat), biasanya ada lebih dari satu cara untuk menguliti kucing. Jalankan kueri Anda dari antarmuka baris perintah basis data dan bereksperimenlah dengan berbagai pendekatan untuk mempercepat prosesnya. Di MySQL, Anda dapat menggunakan

pernyataan EXPLAIN untuk melihat di mana kueri mungkin salah arah. Secara umum, prinsipnya adalah meminimalkan gabungan dan memaksimalkan penggunaan indeks.

(3) Minimalkan pembuatan konten statis dari PHP. Jika setiap bagian HTML yang Anda hasilkan berasal dari echo atau `print()`, prosesnya akan memakan waktu lebih lama. (Ini adalah salah satu argumen untuk beralih ke logika dan konten terpisah seperti yang dijelaskan sebelumnya.) Ini juga berlaku untuk membuat tombol gambar secara dinamis Anda mungkin ingin menggunakan PHP untuk membuat tombol sekali lalu menggunakannya kembali sesuai kebutuhan. Jika Anda membuat halaman statis murni dari fungsi atau templat setiap kali halaman dimuat, pertimbangkan untuk menjalankan fungsi atau menggunakan templat sekali dan menyimpan hasilnya.

(4) Gunakan fungsi string alih-alih ekspresi reguler jika memungkinkan. Fungsi string lebih cepat.

(5) Gunakan string dengan tanda kutip tunggal alih-alih string dengan tanda kutip ganda jika memungkinkan. PHP mengevaluasi string yang diberi tanda kutip ganda, mencari variabel yang akan diganti. String yang diberi tanda kutip tunggal tidak dievaluasi. Di sisi lain, jika string tersebut diberi tanda kutip tunggal, kemungkinan besar itu adalah konten statis. Tinjau apa yang Anda lakukan dan lihat apakah Anda dapat menyingkirkan string tersebut sama sekali dengan mengubahnya menjadi HTML statis.

Menggunakan Produk Zend

Zend Technologies memiliki mesin skrip PHP (*Open Source*) untuk PHP 4 dan seterusnya. Mesin ini jauh lebih cepat (dikutip dua hingga sepuluh kali) daripada mesin versi 3, jadi jika Anda belum memperbaruinya, bantulah diri Anda sendiri dan instal versi 4. Selain mesin dasar, Anda juga dapat mengunduh Zend Optimizer. Ini adalah pengoptimal multi-pass yang akan mengoptimalkan kode untuk Anda, dan dapat meningkatkan kecepatan skrip Anda dari 40% menjadi 100%. Anda memerlukan PHP 4.0.2 atau lebih tinggi untuk menjalankan pengoptimal. Meskipun sumbernya tertutup, aplikasi ini dapat diunduh gratis dari situs Zend: http://www.zend.com

Add-on ini bekerja dengan mengoptimalkan kode yang dihasilkan oleh kompilasi skrip Anda saat dijalankan. Produk Zend yang akan datang meliputi Zend Cache, yang akan mengkompilasi skrip PHP Anda dan menyimpannya dalam cache sehingga skrip tersebut hanya dikompilasi ulang saat ditulis ulang. Fitur ini akan memberikan peningkatan kinerja lainnya.

Pengujian

Meninjau dan menguji kode merupakan poin dasar lain dari rekayasa perangkat lunak yang sering diabaikan dalam pengembangan Web. Cukup mudah untuk mencoba menjalankan sistem dengan dua atau tiga kasus pengujian, lalu berkata, "yup, ini berfungsi dengan baik." Ini adalah kesalahan yang umum dilakukan. Pastikan Anda telah menguji dan meninjau beberapa skenario secara menyeluruh sebelum membuat proyek siap diproduksi. Kami menyarankan dua pendekatan yang dapat Anda gunakan untuk mengurangi tingkat bug pada kode Anda. (Anda tidak akan pernah dapat menghilangkan bug sama sekali; tetapi Anda tentu dapat menghilangkan atau meminimalkan sebagian besar bug.)

Pertama, terapkan praktik peninjauan kode. Ini adalah proses di mana programmer atau tim programmer lain melihat kode Anda dan menyarankan perbaikan. Jenis analisis ini sering kali akan menyarankan:

- Kesalahan yang Anda lewatkan
- Kasus uji yang belum Anda pikirkan
- Optimasi
- Peningkatan keamanan
- Komponen yang sudah ada yang dapat Anda gunakan untuk meningkatkan sepotong kode
- Fungsionalitas tambahan

Bahkan jika Anda bekerja sendiri, akan menjadi hal yang baik untuk menemukan "teman kode" yang berada dalam situasi yang sama dan meninjau kode untuk satu sama lain.

Saran kedua yang kami miliki adalah Anda menemukan penguji untuk aplikasi Web Anda yang mewakili pengguna akhir produk. Perbedaan utama antara aplikasi Web dan aplikasi desktop adalah bahwa siapa pun dan semua orang akan menggunakan aplikasi Web. Anda tidak boleh membuat asumsi bahwa pengguna akan terbiasa dengan komputer. Anda tidak dapat memberi mereka manual tebal atau kartu referensi cepat. Sebaliknya, Anda harus membuat aplikasi Web terdokumentasi sendiri dan jelas. Anda harus memikirkan cara-cara di mana pengguna ingin menggunakan aplikasi Anda.

Kegunaan benar-benar yang terpenting. Sangat sulit untuk memahami masalah yang akan dihadapi pengguna akhir yang naif jika Anda adalah seorang programmer atau peselancar web yang berpengalaman. Salah satu cara untuk mengatasinya adalah dengan mendapatkan penguji yang mewakili pengguna pada umumnya. Salah satu cara yang telah kami lakukan di masa lalu adalah merilis aplikasi web hanya dalam versi beta. Ketika Anda merasa telah mengatasi sebagian besar bug, publikasikan aplikasi tersebut kepada sekelompok kecil pengguna uji dan dapatkan lalu lintas yang sedikit melalui situs tersebut. Tawarkan layanan gratis kepada 100 pengguna pertama sebagai imbalan atas umpan balik tentang situs tersebut.

Saya jamin mereka akan memberikan beberapa kombinasi data atau penggunaan yang belum pernah Anda pikirkan. Jika Anda membangun situs web untuk perusahaan klien, mereka sering kali dapat menyediakan sekumpulan pengguna naif yang baik dengan meminta staf di perusahaan mereka untuk bekerja melalui situs tersebut. (Hal ini memiliki manfaat intrinsik untuk meningkatkan rasa kepemilikan klien terhadap suatu situs.)

BAB 2
DEBUGGING

Bab ini akan membahas debugging skrip PHP. Jika Anda telah membaca beberapa contoh dalam buku ini atau pernah menggunakan PHP sebelumnya, Anda mungkin telah mengembangkan beberapa keterampilan dan teknik debugging sendiri. Seiring dengan semakin kompleksnya proyek Anda, debugging bisa menjadi lebih sulit. Meskipun keterampilan Anda meningkat, kesalahan lebih mungkin melibatkan beberapa file, atau interaksi antara kode beberapa orang.

Dalam Bab ini penulis akan menjelaskan tentang

- ✓ Jenis kesalahan pemrograman
- ✓ Kesalahan sintaksis
- ✓ Kesalahan runtime
- ✓ Kesalahan logika
- ✓ Pesan kesalahan
- ✓ Tingkat kesalahan
- ✓ Memicu kesalahan Anda sendiri
- ✓ Menangani kesalahan dengan baik
- ✓ Debugging jarak jauh

2.1 KESALAHAN PEMROGRAMAN

Terlepas dari bahasa yang Anda gunakan, ada tiga jenis umum kesalahan program:

- Kesalahan Sintaksis
- Kesalahan Runtime
- Kesalahan Logika

Kita akan membahas masing-masing secara singkat sebelum membahas beberapa taktik untuk mendeteksi, menangani, menghindari, dan mengatasi kesalahan.

Kesalahan Sintaksis

Bahasa memiliki seperangkat aturan yang disebut sintaksis suatu bahasa, yang harus diikuti oleh pernyataan-pernyataan agar dianggap valid. Hal ini berlaku untuk bahasa alami, seperti bahasa Inggris, dan bahasa pemrograman, seperti PHP. Jika suatu pernyataan tidak mengikuti aturan suatu bahasa, maka pernyataan tersebut dikatakan memiliki kesalahan sintaksis. Kesalahan sintaksis sering juga disebut kesalahan parser ketika membahas bahasa yang ditafsirkan, seperti PHP, atau kesalahan kompiler ketika membahas bahasa yang dikompilasi, seperti C atau Java.

Jika kita melanggar aturan sintaksis bahasa Inggris, ada kemungkinan besar orang-orang akan tetap mengetahui apa yang ingin kita katakan. Hal ini biasanya tidak terjadi pada bahasa pemrograman. Jika skrip tidak mengikuti aturan sintaksis PHP jika skrip tersebut mengandung kesalahan sintaksis parser PHP tidak akan dapat memproses sebagian atau seluruhnya. Orang-orang pandai menyimpulkan informasi dari data parsial atau yang saling

bertentangan. Komputer tidak. Di antara banyak aturan lainnya, sintaksis PHP mengharuskan pernyataan diakhiri dengan titik koma, string diapit tanda kutip, dan parameter yang diteruskan ke fungsi dipisahkan dengan koma dan diapit tanda kurung. Jika kita melanggar aturan ini, skrip PHP kita kemungkinan besar tidak akan berfungsi, dan kemungkinan akan menghasilkan pesan kesalahan saat pertama kali kita mencoba menjalankannya.

Salah satu kekuatan PHP yang hebat adalah pesan kesalahan yang berguna yang diberikannya saat terjadi kesalahan. Pesan kesalahan PHP biasanya akan memberi tahu Anda apa yang salah, di file mana kesalahan terjadi, dan di baris mana kesalahan ditemukan.
Pesan kesalahan menyerupai berikut ini:

```
Parse error: parse error in
 /home/book/public_html/chapter23/error.php on line 2
```

Kesalahan ini disebabkan oleh skrip berikut:

```
<?
     $date = date(m.d.y’);
?>
```

Anda dapat melihat bahwa kami mencoba meneruskan string ke fungsi date() tetapi secara tidak sengaja melewatkan tanda kutip pembuka yang akan menandai awal string. Kesalahan sintaksis sederhana seperti ini biasanya paling mudah ditemukan. Kita dapat membuat kesalahan serupa, tetapi lebih sulit ditemukan dengan lupa mengakhiri string, seperti yang ditunjukkan dalam contoh ini:

```
<?
     $date = date(‘m.d.y);
?>
```

Skrip ini akan menghasilkan pesan kesalahan berikut:

```
Parse error: parse error in
 /home/book/public_html/chapter23/error.php on line 4
```

Jelas, karena skrip kita hanya memiliki tiga baris, kesalahan kita sebenarnya tidak ada pada baris keempat. Kesalahan saat Anda membuka sesuatu, tetapi gagal menutupnya akan sering muncul seperti ini. Anda dapat mengalami masalah ini dengan tanda kutip tunggal dan ganda serta dengan berbagai bentuk tanda kurung. Skrip berikut akan menghasilkan kesalahan sintaksis yang serupa:

```
<?
   if (true)
   {
```

```
    echo “error here”;
?>
```

Kesalahan ini sulit ditemukan jika terjadi akibat gabungan beberapa file. Kesalahan ini juga sulit ditemukan jika terjadi dalam file besar. Melihat "kesalahan parse pada baris 1001" dari file 1000 baris sudah cukup untuk merusak hari Anda. Namun, secara umum, kesalahan sintaksis adalah jenis kesalahan yang paling mudah ditemukan. Jika Anda membuat kesalahan sintaksis, PHP akan memberi Anda pesan yang memberi tahu Anda di mana menemukan kesalahan Anda.

Kesalahan Runtime

Kesalahan runtime lebih sulit dideteksi dan diperbaiki. Sebuah skrip mengandung kesalahan sintaksis atau tidak. Jika skrip mengandung kesalahan sintaksis, parser akan mendeteksinya. Kesalahan runtime tidak hanya disebabkan oleh konten skrip Anda. Kesalahan ini dapat bergantung pada interaksi antara skrip Anda dan peristiwa atau kondisi lainnya.
Pernyataan berikut;

```
include (“filename.php”);
```

adalah pernyataan PHP yang benar-benar valid. Pernyataan ini tidak mengandung kesalahan sintaksis. Namun, pernyataan ini mungkin menghasilkan kesalahan runtime. Jika Anda menjalankan pernyataan ini dan filename.php tidak ada atau pengguna yang menjalankan skrip tersebut ditolak izin bacanya, Anda akan mendapatkan kesalahan seperti ini:

```
Fatal error: Failed opening required ‘filename.php’
 (include_path=’.:/usr/local/lib/php’) in
 /home/book/public_html/chapter23/error.php on line 1
```

Meskipun tidak ada yang salah dengan kode kita, karena bergantung pada berkas yang mungkin ada atau tidak pada waktu yang berbeda saat kode dijalankan, kode dapat menghasilkan galat waktu proses. Ketiga pernyataan berikut semuanya adalah PHP yang valid. Sayangnya, jika digabungkan, ketiganya mencoba melakukan hal yang mustahil membagi dengan nol.

```
$i = 10;
$j = 0;
$k = $i/$k
```

Potongan kode ini akan menghasilkan peringatan berikut:

```
Warning: Division by zero in
 /home/book/public_html/chapter23/div0.php on line 3
```

Ini akan membuatnya sangat mudah untuk diperbaiki. Hanya sedikit orang yang akan mencoba menulis kode yang mencoba membagi dengan nol dengan sengaja, tetapi mengabaikan untuk memeriksa masukan pengguna sering kali mengakibatkan jenis kesalahan ini. Ini adalah salah satu dari banyak kesalahan runtime yang mungkin Anda lihat saat menguji kode Anda.,Penyebab umum kesalahan runtime meliputi hal-hal berikut:

- panggilan ke fungsi yang tidak ada
- membaca atau menulis file
- interaksi dengan MySQL atau basis data lain
- koneksi ke layanan jaringan
- kegagalan untuk memeriksa data input Kami akan membahas masing-masing secara singkat.

Panggilan ke Fungsi yang Tidak Ada

Sangat mudah untuk secara tidak sengaja memanggil fungsi yang tidak ada. Fungsi bawaan sering kali diberi nama yang tidak konsisten. Mengapa `strip_tags()` memiliki garis bawah, sedangkan `stripslashes()` tidak? Juga mudah untuk memanggil salah satu fungsi Anda sendiri yang tidak ada dalam skrip saat ini, tetapi mungkin ada di tempat lain. Jika kode Anda berisi panggilan ke fungsi yang tidak ada, seperti

```
nonexistent_function();
```

Anda akan melihat pesan kesalahan yang mirip dengan ini:

```
Fatal error: Call to undefined function: nonexistent_function()
 in /home/book/public_html/chapter23/error.php on line 1
```

Demikian pula, jika Anda memanggil fungsi yang sudah ada, tetapi memanggilnya dengan jumlah parameter yang salah, Anda akan menerima peringatan. Fungsi `strstr()` memerlukan dua string: tumpukan jerami untuk dicari dan jarum untuk ditemukan. Jika sebaliknya kita memanggilnya seperti ini:

```
strstr();
```

Kita akan mendapatkan peringatan berikut:

```
Fatal error: Call to undefined function: nonexistent_function()
 in /home/book/public_html/chapter23/error.php on line 1
```

Karena PHP tidak mengizinkan fungsi yang kelebihan beban, baris ini akan selalu salah, tetapi kita mungkin tidak selalu melihat peringatan ini. Pernyataan yang sama dalam skrip berikut juga salah:

```
<?
```

```
  if($var == 4)
  {
    strstr();
  }
?>
```

tetapi kecuali dalam kasus yang mungkin jarang terjadi, di mana variabel `$var` memiliki nilai 4, panggilan ke `strstr()` tidak akan terjadi, dan tidak akan ada peringatan yang dikeluarkan. Memanggil fungsi secara tidak benar mudah dilakukan, tetapi karena pesan kesalahan yang dihasilkan mengidentifikasi baris dan panggilan fungsi yang menyebabkan masalah, keduanya juga mudah diperbaiki. Pesan kesalahan hanya sulit ditemukan jika proses pengujian Anda buruk dan tidak menguji semua kode yang dieksekusi secara kondisional. Saat Anda menguji, salah satu tujuannya adalah mengeksekusi setiap baris kode tepat satu kali. Tujuan lainnya adalah menguji semua kondisi batas dan kelas input.

2.2 MEMBACA ATAU MENULIS FILE

Meskipun segala sesuatu bisa salah pada suatu saat selama masa pakai program Anda, beberapa hal lebih mungkin terjadi daripada yang lain. Kesalahan saat mengakses file cukup mungkin terjadi sehingga Anda perlu menanganinya dengan baik. Hard drive gagal atau penuh, dan kesalahan manusia mengakibatkan perubahan izin direktori. Fungsi seperti `fopen()` yang mungkin gagal sesekali umumnya memiliki nilai balik untuk memberi sinyal bahwa kesalahan telah terjadi. Untuk `fopen()`, nilai balik false menunjukkan kegagalan. Untuk fungsi yang memberikan pemberitahuan kegagalan, Anda perlu memeriksa dengan cermat nilai pengembalian setiap panggilan dan menindaklanjuti kegagalan.

Interaksi dengan MySQL atau Basis Data Lain

Menghubungkan dan menggunakan MySQL dapat menghasilkan banyak kesalahan. Fungsi `mysql_connect()` sendiri dapat menghasilkan setidaknya kesalahan berikut:

- **MySQL Connection Failed:** `Can't connect to MySQL server on 'hostname' (111)`
- **MySQL Connection Failed:** `Can't connect to local MySQL server through socket '/tmp/mysql.sock' (111)`
- **MySQL Connection Failed:** `Unknown MySQL Server Host 'hostname' (2)`
- **MySQL Connection Failed:** `Access denied for user: 'username@localhost' (Using password: YES)`

Seperti yang mungkin Anda harapkan, `mysql_connect()` memberikan nilai balik false saat terjadi kesalahan. Ini berarti Anda dapat dengan mudah menangkap dan menangani jenis kesalahan umum ini.

Jika Anda tidak menghentikan eksekusi skrip Anda secara teratur dan menangani kesalahan ini, skrip Anda akan mencoba terus berinteraksi dengan basis data. Mencoba menjalankan kueri dan mendapatkan hasil tanpa koneksi MySQL yang valid akan mengakibatkan pengunjung Anda melihat layar yang tampak tidak profesional yang penuh dengan pesan kesalahan. Banyak fungsi PHP terkait MySQL yang umum digunakan seperti

mysql_pconnect(), mysql_select_db(), dan mysql_query() juga mengembalikan false untuk menunjukkan bahwa terjadi kesalahan.

Jika terjadi kesalahan, Anda dapat mengakses teks pesan kesalahan menggunakan fungsi mysql_error(), atau kode kesalahan menggunakan fungsi mysql_errno(). Jika fungsi MySQL terakhir tidak menghasilkan kesalahan, mysql_error() mengembalikan string kosong dan mysql_errno() mengembalikan 0. Misalnya, dengan asumsi bahwa kita telah terhubung ke server dan memilih database untuk digunakan, cuplikan kode berikut

```
$result = mysql_query( 'select * from does_not_exist' );
echo mysql_errno();
echo '<BR>';
echo mysql_error();
```

mungkin mengeluarkan

```
1146
Table 'dbname.does_not_exist' doesn't exist
```

Perhatikan bahwa output dari fungsi-fungsi ini merujuk pada fungsi MySQL terakhir yang dijalankan (selain mysql_error() atau mysql_errno()). Jika Anda ingin mengetahui hasil dari suatu perintah, pastikan untuk memeriksanya sebelum menjalankan perintah lainnya.

Seperti kegagalan interaksi file, kegagalan interaksi basis data akan terjadi. Bahkan setelah menyelesaikan pengembangan dan pengujian layanan, Anda terkadang akan menemukan bahwa daemon MySQL (mysqld) telah mogok atau kehabisan koneksi yang tersedia. Jika basis data Anda berjalan pada mesin fisik lain, Anda mengandalkan perangkat keras dan komponen perangkat lunak lain yang dapat gagal koneksi jaringan lain, kartu jaringan, router, dan sebagainya antara server Web Anda dan mesin basis data. Anda perlu ingat untuk memeriksa apakah permintaan basis data Anda berhasil sebelum mencoba menggunakan hasilnya. Tidak ada gunanya mencoba menjalankan kueri setelah gagal terhubung ke basis data dan tidak ada gunanya mencoba mengekstrak dan memproses hasilnya setelah menjalankan kueri yang gagal.

Penting untuk dicatat pada titik ini bahwa ada perbedaan antara kueri yang gagal dan kueri yang hanya gagal mengembalikan data apa pun atau memengaruhi baris apa pun. Kueri SQL yang berisi kesalahan sintaksis SQL atau merujuk ke basis data, tabel, atau kolom yang memang ada mungkin gagal. Kueri berikut, misalnya

```
select * from does_not_exist;
```

akan gagal karena nama tabel tidak ada, dan menghasilkan nomor kesalahan dan pesan yang dapat diambil dengan mysql_errno() dan mysql_error().

Kueri SQL yang valid secara sintaksis, dan hanya merujuk ke basis data, tabel, dan kolom yang ada pada umumnya tidak akan gagal. Namun, kueri tersebut mungkin tidak

mengembalikan hasil apa pun jika kueri tersebut menanyakan tabel kosong atau mencari data yang tidak ada. Dengan asumsi bahwa Anda telah berhasil terhubung ke basis data, dan memiliki tabel bernama exists dan kolom bernama column name, kueri berikut, misalnya

```
select * from exists where column_name = 'not in database';
```

akan berhasil tetapi tidak mengembalikan hasil apa pun. Sebelum Anda menggunakan hasil kueri, Anda perlu memeriksa kegagalan dan tidak adanya hasil.

Koneksi Ke Layanan Jaringan

Meskipun perangkat dan program lain pada sistem Anda terkadang akan gagal, kegagalan tersebut seharusnya jarang terjadi kecuali kualitasnya buruk. Saat menggunakan jaringan untuk terhubung ke mesin lain dan perangkat lunak pada mesin tersebut, Anda perlu menerima bahwa beberapa bagian sistem akan sering gagal. Untuk terhubung dari satu mesin ke mesin lain, Anda mengandalkan banyak perangkat dan layanan yang tidak berada di bawah kendali Anda. Dengan risiko menjadi repetitif, Anda benar-benar perlu memeriksa dengan saksama nilai pengembalian fungsi yang mencoba berinteraksi dengan layanan jaringan. Pemanggilan fungsi seperti

```
$sp = fsockopen("localhost", 5000 );
```

tidak akan memberikan pesan kesalahan jika gagal dalam upayanya untuk terhubung ke port 5000 pada mesin localhost. Menulis ulang panggilan sebagai

```
$sp = fsockopen ( "localhost", 5000, &$errorno, &$errorstr );
if(!$sp)
     echo "ERROR: $errorno: $errorstr";
```

akan memeriksa nilai yang dikembalikan untuk melihat apakah terjadi kesalahan, dan menampilkan pesan kesalahan yang mungkin membantu Anda memecahkan masalah. Dalam kasus ini, akan menghasilkan output:

```
ERROR: 111: Connection refused
```

Kesalahan runtime lebih sulit dihilangkan daripada kesalahan sintaksis karena parser tidak dapat memberi sinyal kesalahan saat kode pertama kali dijalankan. Karena kesalahan runtime terjadi sebagai respons terhadap kombinasi kejadian, kesalahan tersebut sulit dideteksi dan dipecahkan. Parser tidak dapat secara otomatis memberi tahu Anda bahwa baris tertentu akan menghasilkan kesalahan. Pengujian Anda perlu memberikan salah satu situasi yang menyebabkan kesalahan.

Penanganan kesalahan runtime memerlukan sejumlah pemikiran ke depan; untuk memeriksa berbagai jenis kegagalan yang mungkin terjadi, dan kemudian mengambil tindakan yang tepat. Pengujian yang cermat juga diperlukan untuk mensimulasikan setiap kelas

kesalahan runtime yang mungkin terjadi. Ini tidak berarti Anda perlu mencoba mensimulasikan setiap kesalahan yang mungkin terjadi. Misalnya, MySQL dapat menyediakan satu dari sekitar 200 nomor kesalahan dan pesan yang berbeda. Anda perlu mensimulasikan kesalahan pada setiap pemanggilan fungsi yang mungkin menghasilkan kesalahan, dan kesalahan dari setiap jenis yang ditangani oleh blok kode yang berbeda.

Kegagalan Memeriksa Data Input

Sering kali kita membuat asumsi tentang data input yang akan dimasukkan oleh pengguna. Jika data ini tidak sesuai dengan harapan kita, hal itu dapat menyebabkan kesalahan, baik kesalahan runtime maupun kesalahan logika (dijelaskan secara terperinci di bagian berikut). Contoh klasik kesalahan runtime terjadi saat kita menangani data input pengguna dan kita lupa untuk `AddSlashes()` padanya. Ini berarti jika kita memiliki pengguna dengan nama seperti O'Grady yang berisi apostrof, kita akan mendapatkan kesalahan dari fungsi basis data. Kita akan membahas lebih lanjut tentang kesalahan karena asumsi tentang data input di bagian berikutnya.

Kesalahan Logika

Kesalahan logika dapat menjadi jenis kesalahan yang paling sulit ditemukan dan dihilangkan. Jenis kesalahan ini terjadi ketika kode yang benar-benar valid melakukan persis apa yang diperintahkan, tetapi bukan itu yang dimaksudkan oleh penulisnya.
Kesalahan logika dapat disebabkan oleh kesalahan pengetikan sederhana, seperti:

```
for ( $i = 0; $i < 10; $i++ );
 {
 echo “doing something<BR>”;
 }
```

Potongan kode ini sepenuhnya valid. Kode ini mengikuti sintaksis PHP yang valid. Kode ini tidak bergantung pada layanan eksternal apa pun, jadi kecil kemungkinannya gagal saat dijalankan. Kecuali jika Anda memeriksanya dengan sangat saksama, kode ini mungkin tidak akan melakukan apa yang Anda pikirkan atau yang diinginkan oleh pemrogram. Sekilas, kode ini tampak akan mengulang for loop sepuluh kali, menggemakan "melakukan sesuatu" setiap kali.

Penambahan titik koma yang tidak relevan di akhir baris pertama berarti bahwa loop tidak memiliki efek pada baris berikutnya. For loop akan mengulang sepuluh kali tanpa hasil, dan kemudian pernyataan gema akan dieksekusi sekali. Karena kode ini adalah cara yang sepenuhnya valid, tetapi tidak efisien, untuk menulis kode guna mencapai hasil ini, parser tidak akan mengeluh. Komputer sangat ahli dalam beberapa hal, tetapi tidak memiliki akal sehat atau kecerdasan. Komputer akan melakukan persis seperti yang diperintahkan. Anda perlu memastikan bahwa apa yang Anda katakan kepadanya adalah persis apa yang Anda inginkan.

Kesalahan logika tidak disebabkan oleh kegagalan kode apa pun, tetapi hanya kegagalan programmer untuk menulis kode yang memerintahkan komputer untuk melakukan apa yang diinginkannya. Akibatnya, kesalahan tidak dapat dideteksi secara otomatis. Anda tidak akan diberi tahu bahwa kesalahan telah terjadi, dan Anda tidak akan diberi nomor baris untuk mencari masalahnya. Kesalahan logika hanya akan dideteksi melalui pengujian yang

tepat. Kesalahan logika seperti contoh sepele sebelumnya cukup mudah terjadi, tetapi juga mudah diperbaiki karena saat pertama kali kode Anda dijalankan, Anda akan melihat keluaran yang berbeda dari yang Anda harapkan. Sebagian besar kesalahan logika sedikit lebih berbahaya.

Kesalahan logika yang merepotkan biasanya terjadi karena asumsi pengembang yang salah. Bab 1, "Menggunakan PHP dan MySQL untuk Proyek Besar," merekomendasikan penggunaan pengembang lain untuk meninjau kode guna menyarankan kasus pengujian tambahan, dan menggunakan orang-orang dari audiens target daripada pengembang untuk pengujian. Mengasumsikan bahwa orang hanya akan memasukkan jenis data tertentu sangat mudah dilakukan, dan sangat mudah tidak terdeteksi jika Anda melakukan pengujian sendiri. Misalkan Anda memiliki kotak teks Jumlah Pesanan di situs perdagangan. Pernahkah Anda berasumsi bahwa orang hanya akan memasukkan angka positif? Jika pengunjung memasukkan angka negatif sepuluh, apakah perangkat lunak Anda akan mengembalikan uang ke kartu kreditnya dengan harga barang sepuluh kali lipat?

Misalkan Anda memiliki kotak untuk memasukkan jumlah dolar. Apakah Anda mengizinkan orang memasukkan jumlah dengan atau tanpa tanda dolar? Apakah Anda mengizinkan orang memasukkan angka dengan ribuan yang dipisahkan oleh koma? Beberapa hal ini dapat diperiksa di sisi klien (menggunakan, misalnya, JavaScript) untuk mengurangi beban server Anda. Jika Anda meneruskan informasi ke halaman lain, pernahkah terpikir oleh Anda bahwa mungkin ada karakter yang memiliki arti khusus dalam URL seperti spasi dalam string yang Anda teruskan? Kesalahan logika yang jumlahnya tak terbatas mungkin terjadi. Tidak ada cara otomatis untuk memeriksanya. Satu-satunya solusi adalah pertama-tama, mencoba menghilangkan asumsi yang telah Anda kodekan secara implisit ke dalam skrip dan kedua, menguji secara menyeluruh dengan setiap jenis masukan yang valid dan tidak valid yang memungkinkan, memastikan bahwa Anda mendapatkan hasil yang diantisipasi untuk semuanya.

2.3 BANTUAN DEBUGGING VARIABEL

Seiring dengan semakin kompleksnya proyek, akan berguna untuk memiliki beberapa kode utilitas guna membantu Anda mengidentifikasi penyebab kesalahan. Sepotong kode yang mungkin berguna bagi Anda terdapat dalam Listing 2.1. Kode ini akan menggemakan konten variabel yang diteruskan ke halaman Anda.

Listing 2.1 `dump_variables.php` Kode Ini Dapat Dimasukkan ke Halaman untuk Membuang Isi Variabel untuk Debugging

```
<?
   // these lines format the output as HTML comments
   // and call dump_array repeatedly

   echo "\n<!-- BEGIN VARIABLE DUMP -->\n\n";
```

```
    echo "<!-- BEGIN GET VARS -->\n";
    echo "<!-- ".dump_array($HTTP_GET_VARS)." -->\n";

    echo "<!-- BEGIN POST VARS -->\n";
    echo "<!-- ".dump_array($HTTP_POST_VARS)." -->\n";

    echo "<!-- BEGIN SESSION VARS -->\n";
    echo "<!-- ".dump_array($HTTP_SESSION_VARS)." -->\n";

    echo "<!-- BEGIN COOKIE VARS -->\n";
    echo "<!-- ".dump_array($HTTP_COOKIE_VARS)." -->\n";

    echo "\n<!-- END VARIABLE DUMP -->\n";

// dump_array() takes one array as a parameter
// It iterates through that array, creating a string
// to represent the array as a set
function dump_array($array)
{
 if(is_array($array))
 {
    $size = count($array);
    $string = "";
    if($size)
    {
      $count = 0;
      $string .= "{ ";
      // add each element's key and value to the string
      foreach($array as $var => $value)
      {
       $string .= "$var = $value";
       if($count++ < ($size-1))
       {
       $string .= ", ";
       }
}
         $string .= " }";
       }
       return $string;
 }
 else
 {
       // if it is not an array, just return it
       return $array;
      }
 }
 ?>
```

Kode ini akan mengulangi empat larik variabel yang diterima halaman. Jika halaman dipanggil dengan variabel GET, variabel POST, cookie, atau memiliki variabel sesi, ini akan menjadi output. Sebaris HTML akan dibuat untuk setiap jenis variabel. Baris-baris tersebut akan menyerupai ini:

```
<!-- { var1 = 'value1', var2 = 'value2', var3 = 'value3' } -->
```

Kami telah meletakkan output dalam komentar HTML sehingga dapat dilihat, tetapi tidak akan mengganggu cara browser menampilkan elemen halaman yang terlihat. Ini adalah cara yang baik untuk menghasilkan informasi debugging. Output yang tepat akan bergantung pada variabel yang diteruskan ke halaman, tetapi ketika ditambahkan ke Listing sebelumnya, salah satu contoh autentikasi, menambahkan baris berikut ke HTML yang dihasilkan oleh skrip:

```
<!-- BEGIN VARIABLE DUMP -->

<!-- BEGIN GET VARS -->
<!--  -->
<!-- BEGIN POST VARS -->
<!-- { userid = testuser, password = test123 } -->
<!-- BEGIN SESSION VARS -->
<!-- { valid_user = testuser } -->
<!-- BEGIN COOKIE VARS -->
<!-- { PHPSESSID = 2552dc82bb465af56d65e9300f75fd68 } -->

<!-- END VARIABLE DUMP -->
```

Anda dapat melihat bahwa ia menampilkan variabel POST yang dikirim dari formulir login pada halaman sebelumnya userid dan password. Ia juga menampilkan variabel sesi yang kita gunakan untuk menyimpan nama pengguna valid_user. Skrip kita menggemakan angka pseudo-acak, PHPSESSID, yang disimpan dalam cookie tersebut untuk mengidentifikasi pengguna tertentu.

Tingkat Pelaporan Kesalahan

PHP memungkinkan Anda untuk mengatur seberapa cerewetnya ia harus menangani kesalahan. Anda dapat mengubah jenis kejadian apa yang akan menghasilkan pesan. Secara default, PHP akan melaporkan semua kesalahan selain pemberitahuan. Tingkat pelaporan kesalahan diatur menggunakan serangkaian konstanta yang telah ditetapkan sebelumnya, ditunjukkan dalam Tabel 2.1.

Tabel 2.1 Konstanta Pelaporan Kesalahan

Nilai	Nama	Arti
1	`E_ERROR`	Laporan kesalahan fatal saat runtime
2	`E_WARNING`	Melaporkan kesalahan non-fatal saat runtime

4	**E_PARSE**	Laporan kesalahan penguraian
8	**E_NOTICE**	Pemberitahuan laporan pemberitahuan bahwa sesuatu yang telah Anda lakukan mungkin merupakan kesalahan
16	**E_CORE_ERROR**	Melaporkan kegagalan saat memulai mesin PHP
32	**E_CORE_WARNING**	Melaporkan kegagalan yang tidak fatal selama memulai mesin PHP
64	**E_COMPILE_ERROR**	Laporkan kesalahan dalam kompilasi
128	**E_COMPILE_WARNING**	Melaporkan kesalahan yang tidak fatal dalam kompilasi
256	**E_USER_ERROR**	Laporkan kesalahan yang dipicu pengguna
512	**E_USER_WARNING**	Laporkan peringatan yang dipicu pengguna
1024	**E_USER_NOTICE**	Laporkan pemberitahuan yang dipicu pengguna
2048	**E_ALL**	Laporkan semua kesalahan dan peringatan

Setiap konstanta mewakili jenis kesalahan yang dapat dilaporkan atau diabaikan. Misalnya, jika Anda menetapkan tingkat kesalahan sebagai E_ERROR, hanya kesalahan fatal yang akan dilaporkan. Konstanta ini dapat digabungkan menggunakan aritmatika biner, untuk menghasilkan tingkat kesalahan yang berbeda. Tingkat kesalahan default, melaporkan semua kesalahan selain pemberitahuan, ditetapkan sebagai

```
E_ALL & ~E_NOTICE
```

Ekspresi ini terdiri dari dua konstanta yang telah ditetapkan sebelumnya yang digabungkan menggunakan operator aritmatika bitwise. Ampersand (&) adalah operator AND bitwise dan tilde (~) adalah operator NOT bitwise. Ekspresi ini dapat dibaca sebagai

```
E_ALL AND NOT
E_NOTICE.
```

E_ALL sendiri pada dasarnya merupakan gabungan dari semua jenis kesalahan lainnya. Kesalahan ini dapat digantikan oleh level-level lain yang di-OR bersama-sama menggunakan operator bitwise atau (|).

```
E_ERROR | E_WARNING | E_PARSE | E_NOTICE | E_CORE_ERROR | E_CORE_WARNING |
E_COMPILE_ERROR |E_COMPILE_WARNING | E_USER_ERROR | E_USER_WARNING |
E_USER_NOTICE
```

Demikian pula, tingkat pelaporan kesalahan default dapat ditetapkan oleh semua tingkat kesalahan kecuali pemberitahuan OR bersama-sama.

```
E_ERROR | E_WARNING | E_PARSE | E_NOTICE | E_CORE_ERROR | E_CORE_WARNING |
E_COMPILE_ERROR |E_COMPILE_WARNING | E_USER_ERROR | E_USER_WARNING |
E_USER_NOTICE
```

2.4 MENGUBAH PENGATURAN PELAPORAN KESALAHAN

Anda dapat mengatur pengaturan pelaporan kesalahan secara global, dalam berkas php.ini atau per skrip. Untuk mengubah pelaporan kesalahan untuk semua skrip, Anda dapat mengubah keempat baris ini dalam berkas php.ini default:

```
error_reporting =       E_ALL & ~E_NOTICE
display_errors  =       On
log_errors      =       Off
track_errors    =       Off
```

Pengaturan global default adalah:

- melaporkan semua kesalahan kecuali pemberitahuan
- mengeluarkan pesan kesalahan sebagai HTML ke keluaran standar
- tidak mencatat pesan kesalahan ke disk
- tidak melacak kesalahan, menyimpan kesalahan dalam variabel `$php_errormsg`

Perubahan yang paling mungkin Anda lakukan adalah menaikkan level pelaporan kesalahan ke E_ALL. Ini akan mengakibatkan banyak pemberitahuan yang dilaporkan, untuk insiden yang mungkin menunjukkan kesalahan, atau mungkin hanya terjadi karena programmer memanfaatkan sifat PHP yang diketik lemah dan fakta bahwa ia secara otomatis menginisialisasi variabel ke 0.

Saat melakukan debugging, Anda mungkin merasa berguna untuk menyetel level `error_reporting` lebih tinggi. Dalam kode produksi, jika Anda memberikan pesan kesalahan yang berguna, mungkin lebih profesional untuk menonaktifkan `display_errors` dan mengaktifkan `log_errors`, sambil membiarkan level `error_reporting` tinggi. Anda kemudian akan dapat merujuk ke kesalahan terperinci dalam log jika masalah dilaporkan. Mengaktifkan `track_errors` dapat membantu Anda mengatasi kesalahan dalam kode Anda sendiri, daripada membiarkan PHP menyediakan fungsionalitas default-nya. Meskipun PHP menyediakan pesan kesalahan yang berguna, perilaku default-nya terlihat buruk saat terjadi kesalahan. Secara default, saat kesalahan fatal terjadi, PHP akan menampilkan yang berikut:

```
<br>
<b>Error Type</b>:  error message in <b>path/file.php</b>
on line <b>lineNumber</b><br>
```

dan berhenti mengeksekusi skrip. Untuk kesalahan yang tidak fatal, teks yang sama akan ditampilkan, tetapi eksekusi tetap diperbolehkan.

Output HTML ini membuat kesalahan terlihat jelas, tetapi terlihat buruk. Gaya pesan kesalahan tidak mungkin sesuai dengan tampilan situs lainnya. Hal ini juga dapat mengakibatkan pengguna Netscape tidak melihat output sama sekali jika konten halaman ditampilkan dalam tabel. Hal ini karena HTML yang membuka tetapi tidak menutup elemen tabel, seperti

```
<table>
<tr><td>
<br>
<b>Error Type</b>:  error message in <b>path/file.php</b>
on line <b>lineNumber</b><br>
```

akan ditampilkan sebagai layar kosong oleh Netscape.

Kita tidak harus mempertahankan perilaku penanganan kesalahan default PHP, atau bahkan menggunakan pengaturan yang sama untuk semua berkas. Untuk mengubah tingkat pelaporan kesalahan untuk skrip saat ini, Anda dapat memanggil fungsi `error_reporting()`. Melewati konstanta laporan kesalahan, atau kombinasi keduanya, menyetel tingkat dengan cara yang sama seperti yang dilakukan direktif serupa di php.ini. Fungsi tersebut mengembalikan tingkat pelaporan kesalahan sebelumnya. Cara umum untuk menggunakan fungsi tersebut adalah seperti ini:

```
// turn off error reporting
$old_level = error_reporting(0);
// here, put code that will generate warnings
// turn error reporting back on
error_reporting($old_level);
```

Potongan kode ini akan menonaktifkan pelaporan kesalahan, sehingga kita dapat menjalankan beberapa kode yang mungkin akan menghasilkan peringatan yang tidak ingin kita lihat. Menonaktifkan pelaporan kesalahan secara permanen adalah ide yang buruk karena akan mempersulit Anda menemukan kesalahan pengodean dan memperbaikinya.

Memicu Kesalahan Anda Sendiri

Fungsi `trigger_error()` dapat digunakan untuk memicu kesalahan Anda sendiri. Kesalahan yang dibuat dengan cara ini akan ditangani dengan cara yang sama seperti kesalahan PHP biasa. Fungsi ini memerlukan pesan kesalahan, dan secara opsional dapat diberikan jenis kesalahan. Jenis kesalahan harus berupa salah satu dari `E_USER_ERROR`, `E_USER_WARNING`, atau `E_USER_NOTICE`. Jika Anda tidak menentukan jenis, defaultnya adalah `E_USER_NOTICE`. Anda menggunakan `trigger_error()` seperti yang ditunjukkan pada berikut ini:

```
trigger_error("This computer will self destruct in 15 seconds",
  E_USER_WARNING);
```

(Perhatikan bahwa fungsi ini ditambahkan pada PHP versi 4.0.1.)

Penanganan Kesalahan dengan Baik

Jika Anda berasal dari latar belakang C++ atau Java, Anda mungkin kehilangan penanganan pengecualian saat menggunakan PHP. Pengecualian memungkinkan fungsi memberi sinyal bahwa kesalahan telah terjadi dan menyerahkan penanganan kesalahan kepada penangan pengecualian. Meskipun PHP tidak memiliki pengecualian, PHP 4.0.1

memperkenalkan mekanisme yang dapat digunakan dengan cara yang sama. Anda telah melihat bahwa Anda dapat memicu kesalahan Anda sendiri. Anda juga dapat menyediakan penangan kesalahan Anda sendiri untuk menangkap kesalahan. Fungsi `set_error_handler()` memungkinkan Anda menyediakan fungsi yang akan dipanggil saat kesalahan, peringatan, dan pemberitahuan tingkat pengguna terjadi. Anda memanggil `set_error_handler()` dengan nama fungsi yang ingin Anda gunakan sebagai penangan kesalahan.

Fungsi penanganan kesalahan Anda harus mengambil dua parameter; jenis kesalahan dan pesan kesalahan. Berdasarkan kedua variabel ini, fungsi Anda dapat memutuskan cara menangani kesalahan. Jenis kesalahan harus menjadi salah satu konstanta jenis kesalahan yang ditentukan. Pesan kesalahan adalah string deskriptif.
Panggilan ke `set_error_handler()` akan terlihat seperti ini:

```
set_error_handler("myErrorHandler");
```

Setelah memberi tahu PHP untuk menggunakan fungsi yang disebut `myErrorHandler()`, kita kemudian harus menyediakan fungsi dengan nama itu. Fungsi ini harus memiliki prototipe berikut:

```
myErrorHandler(int error_type, string error_msg)
```

tetapi apa yang sebenarnya dilakukannya terserah Anda.
Tindakan logis mungkin mencakup

- Menampilkan pesan kesalahan yang diberikan
- Menyimpan informasi dalam berkas log
- Mengirim kesalahan melalui email ke suatu alamat
- Mengakhiri skrip dengan panggilan untuk keluar

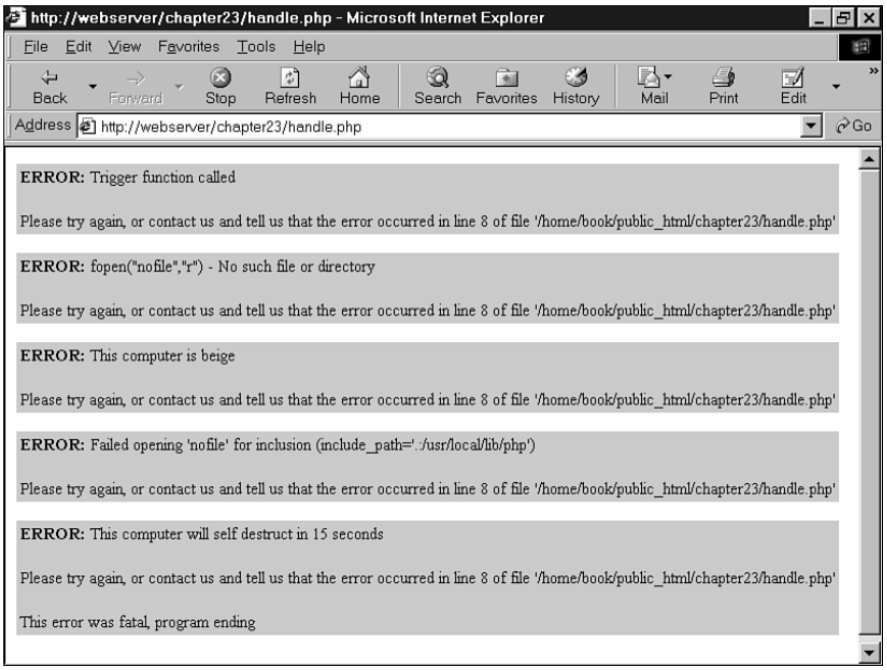
Listing 2.2 berisi skrip yang mendeklarasikan penangan kesalahan, menyetel penangan kesalahan menggunakan `set_error_handler()`, lalu menghasilkan beberapa kesalahan.

Listing 2.2 Handle.Php; Skrip Ini Mendeklarasikan Penangan Kesalahan Kustom Dan Menghasilkan Kesalahan Yang Berbeda

```
<?
  // The error handler function
function myErrorHandler ($errno, $errstr)
  {
  echo " <br><table bgcolor = '#cccccc'><tr><td>
        <P><B>ERROR:</B> $errstr
        <P>Please try again, or contact us and tell us that
        the error occurred in line ".__LINE__." of file '".__FILE__."'";
  if ($errno == E_USER_ERROR||$errno == E_ERROR)
  {
    echo "<P>This error was fatal, program ending";
```

```
    echo “</td></tr></table>”;
    //close open resources, include page footer, etc
    exit;
  }
  echo “</td></tr></table>”;
}
// Set the error handler
set_error_handler(“myErrorHandler”);

//trigger different levels of error
trigger_error(“Trigger function called”, E_USER_NOTICE);
fopen(“nofile”, “r”);
trigger_error(“This computer is beige”, E_USER_WARNING);
include (“nofile”);
trigger_error(“This computer will self destruct in 15 seconds”,
  E_USER_ERROR);
?>
```

Output dari skrip ini ditunjukkan pada Gambar 23.1.



Gambar 23.1 Anda Dapat Memberikan Pesan Kesalahan Yang Lebih Ramah Daripada Php Jika Anda Menggunakan Penangan Kesalahan Anda Sendiri.

Penangan kesalahan khusus ini tidak melakukan apa pun selain perilaku default. Karena kode ini ditulis oleh Anda, Anda dapat membuatnya melakukan apa saja. Penangan ini memberi Anda pilihan tentang apa yang akan diberitahukan kepada pengunjung Anda ketika terjadi kesalahan dan bagaimana menyajikan informasi tersebut agar sesuai dengan situs lainnya. Yang lebih penting, penangan ini memberi Anda fleksibilitas untuk memutuskan apa

yang terjadi. Haruskah skrip dilanjutkan? Haruskah pesan dicatat atau ditampilkan? Haruskah dukungan teknis diberitahukan secara otomatis?

Penting untuk dicatat bahwa penangan kesalahan Anda tidak akan bertanggung jawab untuk menangani semua jenis kesalahan. Beberapa kesalahan, seperti kesalahan penguraian dan kesalahan fatal runtime akan tetap memicu perilaku default. Jika ini menjadi masalah bagi Anda, pastikan Anda memeriksa parameter dengan saksama sebelum meneruskannya ke fungsi yang dapat menghasilkan kesalahan fatal dan memicu kesalahan tingkat E_USER_ERROR Anda sendiri jika parameter Anda akan menyebabkan kegagalan.

Debugging Jarak Jauh

Versi 3 dari PHP menyertakan debugger jarak jauh. Tujuannya adalah untuk mengirim informasi terperinci tentang kesalahan dan kejadian saat kode Anda dijalankan ke port yang dapat dipantau. Fungsi ini tidak tersedia di PHP versi 4 tanpa membeli Zend IDE. Jika Anda masih menggunakan PHP 3 dan memiliki bug yang sangat mengganggu yang tidak dapat dilacak dengan cara lain, Anda mungkin ingin mencari petunjuk konfigurasi PHP enable-debugger dan pengaturan php.ini debugger.host, debugger.port, dan debugger.enabled. Saat diaktifkan, debugger jarak jauh akan membuang ratusan baris informasi untuk memberi tahu Anda dengan tepat apa yang terjadi dalam skrip yang sedang berjalan.

BAB 3
AUTENTIKASI DAN PERSONALISASI PENGGUNA

Dalam proyek ini, kami akan meminta pengguna untuk mendaftar di situs web kami. Setelah mereka melakukannya, kami akan dapat melacak apa yang mereka minati dan menunjukkan konten yang sesuai. Ini disebut personalisasi pengguna. Proyek khusus ini akan memungkinkan pengguna untuk membuat serangkaian penanda di web, dan menyarankan tautan lain yang mungkin mereka anggap menarik berdasarkan perilaku mereka sebelumnya. Secara umum, personalisasi pengguna dapat digunakan di hampir semua aplikasi berbasis web untuk menunjukkan kepada pengguna konten yang mereka inginkan dalam format yang mereka inginkan.

Dalam proyek ini, dan proyek-proyek lainnya yang akan menyusul, kami akan mulai dengan melihat serangkaian persyaratan yang mirip dengan yang mungkin Anda dapatkan dari klien. Kami akan mengembangkan persyaratan tersebut menjadi serangkaian komponen solusi, membangun desain untuk menghubungkan komponen-komponen tersebut bersama-sama, dan kemudian mengimplementasikan masing-masing komponen. Dalam proyek ini, kami akan menerapkan fungsi berikut:

- Login dan mengautentikasi pengguna
- Mengelola kata sandi
- Merekam preferensi pengguna
- Personalisasi konten
- Merekomendasikan konten berdasarkan pengetahuan yang ada tentang pengguna

3.1 IDENTIFIKASI DAN PERSONALISASI PENGGUNA

Ada beberapa alternatif untuk autentikasi pengguna, seperti yang telah kita lihat di bagian lain buku ini. Karena kita ingin menghubungkan pengguna dengan beberapa informasi personalisasi, kita akan menyimpan login dan kata sandi pengguna dalam basis data MySQL dan mengautentikasi berdasarkan informasi tersebut. Jika kita akan mengizinkan pengguna untuk login dengan nama pengguna dan kata sandi, kita akan memerlukan komponen berikut:

- Pengguna harus dapat mendaftarkan nama pengguna dan kata sandi. Kita akan memerlukan beberapa batasan pada panjang dan format nama pengguna dan kata sandi. Kita harus menyimpan kata sandi dalam format terenkripsi demi alasan keamanan.
- Pengguna harus dapat login dengan detail yang mereka berikan dalam proses pendaftaran.
- Pengguna harus dapat keluar setelah selesai menggunakan situs. Hal ini tidak terlalu penting jika orang menggunakan situs dari PC rumah, tetapi sangat penting untuk keamanan jika mereka menggunakan situs dari PC bersama.
- Situs harus dapat memeriksa apakah pengguna masuk atau tidak, dan mengakses data untuk pengguna yang masuk.

- Pengguna harus dapat mengubah kata sandi mereka sebagai bantuan untuk keamanan.
- Pengguna terkadang lupa kata sandi mereka. Mereka seharusnya dapat mengatur ulang kata sandi mereka tanpa memerlukan bantuan pribadi dari kami. Cara umum untuk melakukannya adalah dengan mengirimkan kata sandi kepada pengguna melalui alamat email yang telah diberikannya saat registrasi. Ini berarti kami perlu menyimpan alamat emailnya saat registrasi. Karena kami menyimpan kata sandi dalam bentuk terenkripsi dan tidak dapat mendekripsi kata sandi asli, kami sebenarnya perlu membuat kata sandi baru, mengaturnya, dan mengirimkannya melalui pos kepada pengguna.

Kami akan menulis fungsi untuk semua bagian fungsionalitas ini. Sebagian besar dari fungsi tersebut dapat digunakan kembali, atau dapat digunakan kembali dengan sedikit modifikasi, dalam proyek lain.

Menyimpan Bookmark

Untuk menyimpan bookmark pengguna, kita perlu menyiapkan beberapa ruang di basis data MySQL kita. Kita memerlukan fungsi berikut:

- Pengguna harus dapat mengambil dan melihat bookmark mereka.
- Pengguna harus dapat menambahkan bookmark baru. Kita harus memeriksa apakah URL ini valid.
- Pengguna harus dapat menghapus bookmark.

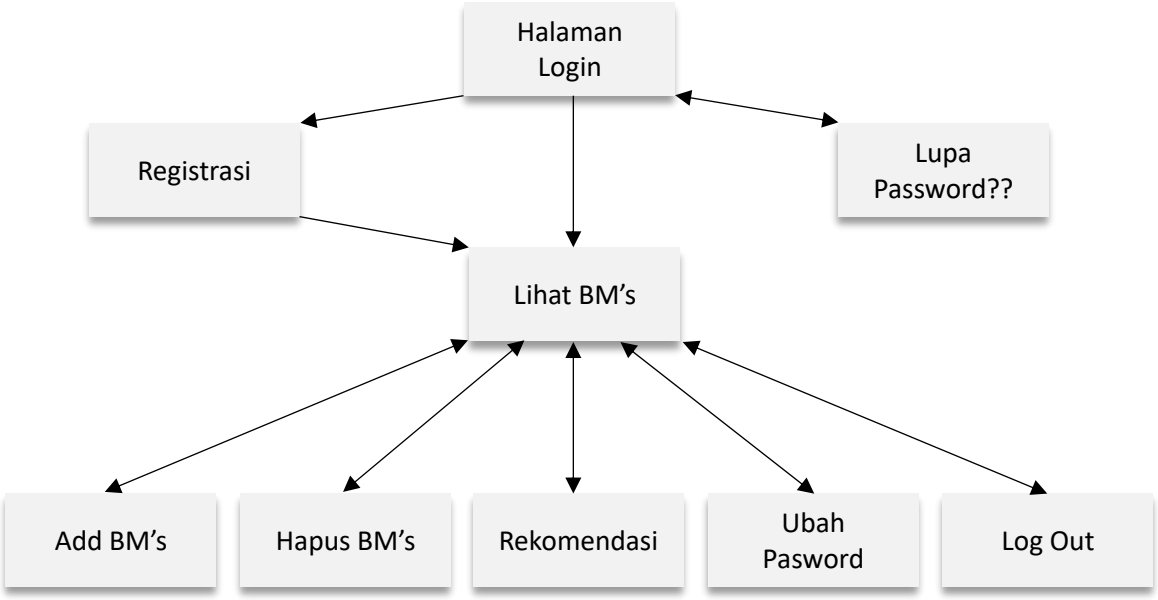
Sekali lagi, kita dapat menulis fungsi untuk setiap bagian fungsi ini.

Merekomendasikan Bookmark

Kita dapat menggunakan sejumlah pendekatan berbeda untuk merekomendasikan bookmark kepada pengguna. Kita dapat merekomendasikan yang paling populer atau yang paling populer dalam suatu topik. Untuk proyek ini, kita akan menerapkan sistem saran "yang sepemikiran" yang mencari pengguna yang memiliki bookmark yang sama dengan pengguna yang login, dan menyarankan bookmark mereka yang lain kepada pengguna. Untuk menghindari rekomendasi bookmark pribadi, kita hanya akan merekomendasikan bookmark yang disimpan oleh lebih dari satu pengguna lain. Kita dapat menulis fungsi untuk mengimplementasikan fungsi ini.

Gambaran Umum Solusi

Setelah mencoret-coret serbet, kami membuat diagram alur sistem yang ditunjukkan pada Gambar 3.1.



Gambar 3.1 Diagram Ini Menunjukkan Kemungkinan Jalur Melalui Sistem Phpbookmark.

Kita akan membangun modul untuk setiap kotak pada diagram ini—ada yang memerlukan satu skrip dan ada yang memerlukan dua skrip. Kita juga akan menyiapkan pustaka fungsi untuk

- ✓ Autentikasi pengguna.
- ✓ Penyimpanan dan pengambilan bookmark.
- ✓ Validasi data.
- ✓ Koneksi basis data.
- ✓ Output ke browser. Kita akan membatasi semua produksi HTML ke pustaka fungsi ini, memastikan bahwa presentasi visual konsisten di seluruh situs. (Ini adalah pendekatan API fungsi untuk memisahkan logika dan konten.).

Kita juga perlu membangun basis data back-end untuk sistem. Ringkasan file yang disertakan ditunjukkan pada Tabel 3.1.

Tabel 3.1 File Dalam Aplikasi Phpbookmark

Nama file	Keterangan
`bookmarks.sql`	Pernyataan SQL untuk membuat database PHPBookmark
`login.php`	Halaman depan dengan formulir login untuk sistem
`register_form.php`	Formulir bagian pengguna untuk mendaftar di sistem
`register_new.php`	Skrip untuk memproses pendaftaran baru
`forgot_form.php`	Formulir bagi pengguna untuk diisi jika mereka lupa kata sandinya
`forgot_passwd.php`	Skrip untuk mengatur ulang kata sandi yang terlupakan
`member.php`	Halaman untuk pengguna, dengan tampilan semua bookmark-nya saat ini
`add_bm_form.php`	Formulir untuk menambahkan bookmark baru
`add_bms.php`	Skrip untuk benar-benar menambahkan bookmark baru ke database
`delete_bms.php`	Skrip untuk menghapus bookmark yang dipilih dari daftar pengguna

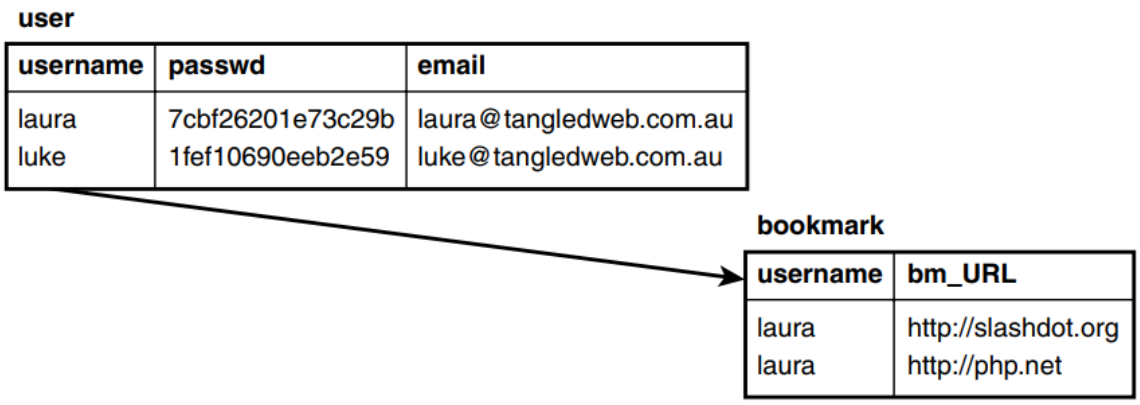
recommend.php	Skrip untuk menyarankan rekomendasi kepada pengguna, berdasarkan pengguna dengan minat yang sama
change_passwd_form.php	Formulir yang harus diisi oleh anggota jika ingin mengubah kata sandinya
change_passwd.php	Script untuk mengubah password pengguna di database
change_passwd.php	Script untuk mengeluarkan pengguna dari aplikasi
bookmark_fns.php	Koleksi termasuk untuk aplikasi
data_valid_fns.php	Fungsi untuk memvalidasi data input pengguna
db_fns.php	Fungsi untuk terhubung ke database
user_auth_fns.php	Fungsi untuk otentikasi pengguna
url_fns.php	Fungsi untuk menambah dan menghapus bookmark dan untuk merekomendasi
output_fns.php	Fungsi yang memformat output sebagai HTML
Bookmark.gif	Logo untuk PHPBookmark

Kita akan mulai dengan mengimplementasikan basis data MySQL untuk aplikasi ini karena basis data ini akan dibutuhkan untuk menjalankan hampir semua fungsi lainnya. Kemudian kita akan mengerjakan kode sesuai urutan penulisannya, mulai dari halaman depan, melewati autentikasi pengguna, hingga penyimpanan dan pengambilan bookmark, dan akhirnya ke rekomendasi. Urutan ini cukup logis yang perlu dilakukan hanyalah menyelesaikan dependensi dan membangun terlebih dahulu hal-hal yang akan dibutuhkan untuk modul selanjutnya.

3.2 MENERAPKAN BASIS DATA

Kita hanya memerlukan skema yang cukup sederhana untuk basis data PHPBookmark. Kita perlu menyimpan pengguna beserta alamat email dan kata sandi mereka. Kita juga perlu menyimpan URL bookmark. Satu pengguna dapat memiliki banyak bookmark, dan banyak pengguna dapat mendaftarkan bookmark yang sama. Oleh karena itu, kita memiliki dua tabel, pengguna dan bookmark, seperti yang ditunjukkan pada Gambar 3.2. Tabel pengguna akan menyimpan nama pengguna pengguna (yang merupakan kunci utama), kata sandi, dan alamat email. Tabel bookmark akan menyimpan pasangan nama pengguna dan bookmark (bm_URL). Nama pengguna dalam tabel ini akan merujuk kembali ke nama pengguna dari tabel pengguna.

SQL untuk membuat basis data ini, dan untuk membuat pengguna guna menghubungkan ke basis data dari Web, ditunjukkan pada Listing 3.1. Anda harus mengeditnya jika Anda berencana untuk menggunakannya pada sistem Anda—ubah kata sandi pengguna menjadi sesuatu yang lebih aman!



Gambar 3.2 skema Basis Data Untuk Sistem Phpbookmark.

Listing 24.1 Bookmarks.Sql; File Sql Untuk Menyiapkan Basis Data Bookmark

```
create database bookmarks;
use bookmarks;

create table user  (
 username varchar(16) primary key,
 passwd char(16) not null,
 email varchar(100) not null
);

create table bookmark (
   username varchar(16) not null,
   bm_URL varchar(255) not null,
   index (username),
   index (bm_URL)
);

grant select, insert, update, delete
on bookmarks.*
to bm_user@localhost identified by ‘password’;
```

Anda dapat menyiapkan basis data ini di sistem Anda dengan menjalankan serangkaian perintah ini sebagai pengguna MySQL root. Anda dapat melakukannya dengan perintah berikut di baris perintah sistem Anda:

```
mysql -u root -p < bookmarks.sql
```

Anda kemudian akan diminta untuk mengetikkan kata sandi Anda. Setelah basis data disiapkan, mari kita lanjutkan dan terapkan situs dasar.

Menerapkan Situs Dasar

Halaman pertama yang akan kita buat akan disebut login.php karena menyediakan kesempatan bagi pengguna untuk masuk ke sistem. Kode untuk halaman pertama ini ditunjukkan pada listing 3.2.

Listing 24.2 login.php; Halaman Depan Sistem PHPBookmark

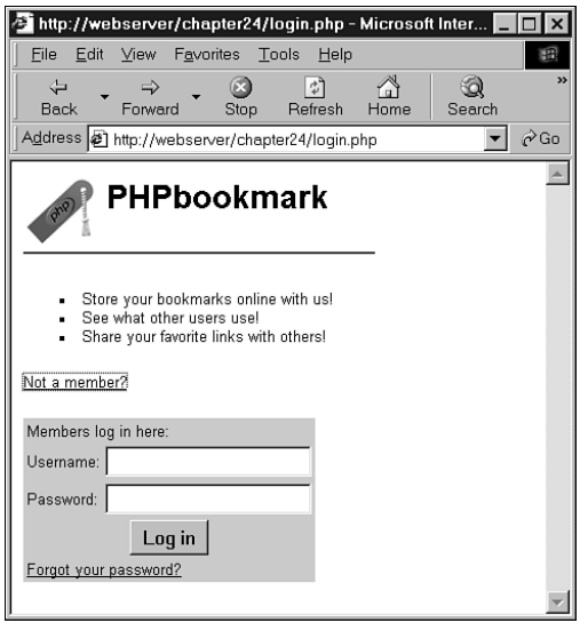
```
<?
  require_once("bookmark_fns.php");
  do_html_header("");
  display_site_info();
  display_login_form();
  do_html_footer();
?>
```

Kode ini terlihat sangat sederhana, karena sebagian besar memanggil fungsi dari API fungsi yang akan kita buat untuk aplikasi ini. Kita akan melihat detail fungsi-fungsi ini sebentar lagi. Hanya dengan melihat berkas ini, kita dapat melihat bahwa kita menyertakan berkas (yang berisi fungsi-fungsi) lalu memanggil beberapa fungsi untuk merender header HTML, menampilkan beberapa konten, dan merender footer HTML. Keluaran dari skrip ini ditunjukkan pada Gambar 3.3.

Semua fungsi untuk sistem disertakan dalam berkas bookmark_fns.php, yang ditunjukkan pada Listing 3.3.

Listing 3.3 Bookmark_Fns.Php; Menyertakan Berkas Fungsi Untuk Aplikasi Bookmark

```
<?
    // We can include this file in all our files
    // this way, every file will contain all our functions
    require_once("data_valid_fns.php");
    require_once("db_fns.php");
    require_once("user_auth_fns.php");
    require_once("output_fns.php");
    require_once("url_fns.php");
 ?>
```



Gambar 3.3 Halaman Depan Sistem Phpbookmark Dihasilkan Oleh Fungsi Rendering Html Di Login.Php.

Seperti yang Anda lihat, berkas ini hanyalah wadah untuk lima berkas penyertaan lainnya yang akan kita gunakan dalam aplikasi ini. Kita telah menyusunnya seperti ini karena fungsi-fungsi tersebut terbagi dalam kelompok-kelompok logis. Beberapa dari kelompok-kelompok ini mungkin berguna untuk proyek-proyek lain, jadi kita menempatkan setiap kelompok fungsi ke dalam berkas yang berbeda, yang akan memberi tahu kita di mana menemukannya saat kita membutuhkannya lagi. Kita menyusun berkas `bookmark_fns.php` karena kita akan menggunakan sebagian besar dari lima berkas fungsi di sebagian besar skrip kita. Lebih mudah untuk menyertakan satu berkas ini di setiap skrip daripada memiliki lima pernyataan penyertaan.

Perhatikan bahwa konstruksi `require_once()` hanya ada di PHP dari versi 4.0.1pl2. Jika Anda menggunakan versi sebelumnya, Anda perlu menggunakan `require()` atau `include()` dan memastikan bahwa file tidak dimuat beberapa kali. Dalam kasus khusus ini, kami menggunakan fungsi dari file output_fns.php. Ini semua adalah fungsi langsung yang menghasilkan HTML yang cukup sederhana. File ini menyertakan empat fungsi yang telah kami gunakan di `login.php`, yaitu, `do_html_header()`, `display_site_info()`, `display_login_form()`, dan `do_html_footer()`, di antara yang lainnya. Kami tidak akan membahas semua fungsi ini secara terperinci, tetapi kami akan melihat satu sebagai contoh. Kode untuk `do_html_header()` ditampilkan dalam Listing 3.4.

Listing 3.4 Fungsi `Do_Html_Header()` Dari `Output_Fns.Php`; Fungsi Ini Menghasilkan Header Standar Yang Akan Muncul Di Setiap Halaman Dalam Aplikasi

```
function do_html_header($title)
{
 // print an HTML header
?>
```

```
<html>
<head>
  <title><?=$title?></title>
  <style>
    body { font-family: Arial, Helvetica, sans-serif; font-size: 13px }
    li, td { font-family: Arial, Helvetica, sans-serif; font-size: 13px }
    hr { color: #3333cc; width=300; text-align=left}
    a { color: #000000 }
  </style>
   </head>
   <body>
   <img src="bookmark.gif" alt="PHPbookmark logo" border=0
    align=left valign=bottom height = 55 width = 57>
  <h1> PHPbookmark</h1>
  <hr>
<?
  if($title)
      do_html_heading($title);
}
```

Seperti yang dapat Anda lihat, satu-satunya logika dalam fungsi ini adalah menambahkan judul dan tajuk yang sesuai ke halaman. Fungsi lain yang telah kami gunakan di `login.php` serupa. Fungsi `display_site_info()` menambahkan beberapa teks umum tentang situs; `display_login_form()` menampilkan formulir abu-abu yang ditunjukkan pada Gambar 3.3; dan `do_html_footer()` menambahkan footer HTML standar ke halaman. (Keuntungan mengisolasi atau menghapus HTML dari aliran logika utama dibahas dalam Bab 1, "Menggunakan PHP dan MySQL untuk Proyek Besar." Kami akan menggunakan pendekatan fungsi API di sini, dan pendekatan berbasis templat di bab berikutnya sebagai kontras.) Melihat Gambar 3.3, Anda dapat melihat bahwa ada tiga opsi di halaman ini pengguna dapat mendaftar, masuk jika mereka telah mendaftar, atau mengatur ulang kata sandi mereka jika mereka lupa. Untuk mengimplementasikan modul-modul ini, kita akan beralih ke bagian berikutnya, autentikasi pengguna.

3.3 MENERAPKAN AUTENTIKASI PENGGUNA

Ada empat elemen utama pada modul autentikasi pengguna: registrasi pengguna, login dan logout, mengubah kata sandi, dan menyetel ulang kata sandi. Kita akan membahas masing-masing elemen ini secara bergantian.

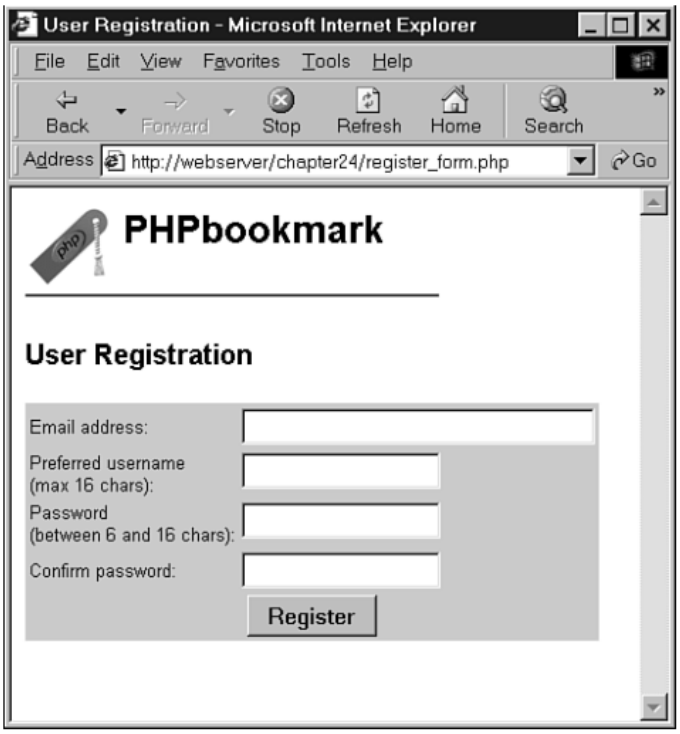
Registrasi

Untuk mendaftarkan pengguna, kita perlu mendapatkan detailnya melalui formulir dan memasukkannya ke dalam basis data. Saat pengguna mengklik tautan "Bukan anggota?" pada halaman login.php, mereka akan dibawa ke formulir registrasi yang dibuat oleh `register_form.php`. Skrip ini ditunjukkan pada Listing 3.5.

Listing 3.5 `register_form.php`; Formulir Ini Memberikan Pengguna Kesempatan untuk Mendaftar dengan PHPBookmarks

```
<?
  require_once("bookmark_fns.php");
  do_html_header("User Registration");

  display_registration_form();

  do_html_footer();
?>
```

Sekali lagi, Anda dapat melihat bahwa halaman ini cukup sederhana dan hanya memanggil fungsi dari pustaka output di `output_fns.php`. Output skrip ini ditunjukkan pada Gambar 3.4.



Gambar 3.4 Formulir Pendaftaran Mengambil Detail Yang Kita Butuhkan Untuk Basis Data. Kita Meminta Pengguna Untuk Mengetikkan Kata Sandi Mereka Dua Kali, Jika Mereka Melakukan Kesalahan.

Form abu-abu pada halaman ini dikeluarkan oleh fungsi `display_registration_form()`, yang terdapat dalam `output_fns.php`. Ketika pengguna mengklik tombol Daftar, ia akan dibawa ke skrip `register_new.php`. Skrip ini ditampilkan dalam Listing 3.6.

Listing 3.6 `register_new.php`; Skrip Ini Memvalidasi Data Pengguna Baru dan Menempatkannya dalam Basis Data

```
<?
  // include function files for this application
```

```
require_once("bookmark_fns.php");

// start session which may be needed later
// start it now because it must go before headers
session_start();

// check forms filled in
if (!filled_out($HTTP_POST_VARS))
{
  do_html_header("Problem:");
  echo "You have not filled the form out correctly - please go back"
       ." and try again.";
  do_html_footer();
  exit;
}

// email address not valid
if (!valid_email($email))
{
  do_html_header("Problem:");
  echo "That is not a valid email address.  Please go back "
       ." and try again.";
  do_html_footer();
  exit;
}

// passwords not the same
if ($passwd != $passwd2)
{
  do_html_heading("Problem:");
  echo "The passwords you entered do not match - please go back"
       ." and try again.";
  do_html_footer();
  exit;
}

// check password length is ok
// ok if username truncates, but passwords will get
// munged if they are too long.
if (strlen($passwd)<6 || strlen($passwd) >16)
{
  do_html_header("Problem:");
  echo "Your password must be between 6 and 16 characters."
       ."Please go back and try again.";
  do_html_footer();
  exit;
}
```

```
  // attempt to register
  $reg_result = register($username, $email, $passwd);
  if ($reg_result == "true")
  {
    // register session variable
    $valid_user = $username;
    session_register("valid_user");

    // provide link to members page
    do_html_header("Registration successful");
    echo "Your registration was successful.  Go to the members page "
         ."to start setting up your bookmarks!";
    do_HTML_URL("member.php", "Go to members page");
  }
  else
  {
    // otherwise provide link back, tell them to try again
    do_html_header("Problem:");
    echo $reg_result;
    do_html_footer();
    exit;
  }

  // end page
  do_html_footer();
?>
```

Ini adalah skrip pertama yang memiliki kompleksitas yang telah kita lihat dalam aplikasi ini. Skrip dimulai dengan menyertakan file fungsi aplikasi dan memulai sesi. Selanjutnya, kita memvalidasi data masukan dari pengguna. Ada sejumlah kondisi yang harus kita uji. Kondisi-kondisi tersebut adalah

- Periksa apakah formulir telah diisi. Kita mengujinya dengan memanggil fungsi `filling_out()` sebagai berikut: `if (!filled_out($HTTP_POST_VARS))` Fungsi ini adalah fungsi yang kita tulis sendiri. Fungsi ini ada di pustaka fungsi dalam file `data_valid_fns.php`. Kita akan melihat fungsi ini sebentar lagi.
- Periksa apakah alamat email yang diberikan valid. Kami mengujinya sebagai berikut: `if (valid_email($email))` Sekali lagi, ini adalah fungsi yang telah kami tulis, yang ada di pustaka `data_valid_fns.php`.
- Periksa apakah kedua kata sandi yang disarankan pengguna sama, sebagai berikut: `if ($passwd != $passwd2)`
- Periksa apakah kata sandi memiliki panjang yang tepat, sebagai berikut: `if (strlen($passwd)<6 || strlen($passwd) >16)`

Dalam contoh kami, kata sandi harus memiliki panjang minimal 6 karakter agar lebih sulit ditebak, dan kurang dari 16 karakter, sehingga akan muat di dalam basis data. Fungsi validasi

data yang telah kami gunakan di sini, `filling_out()` dan `valid_email()`, masing-masing ditampilkan dalam Listing 3.7 dan listing 3.8.

Listing 3.7 Fungsi `stuffed_out()` dari `data_valid_fns.php`; Fungsi ini memeriksa apakah formulir telah diisi

```
function filled_out($form_vars)
{
  // test that each variable has a value
  foreach ($form_vars as $key => $value)
  {
    if (!isset($key) || ($value == “”))
      return false;
  }
  return true;
}
```

Listing 3.8 Fungsi `valid_email()` dari `data_valid_fns.php`—Fungsi Ini Memeriksa Apakah Alamat Email Valid

```
function valid_email($address)
{
  // check an email address is possibly valid
  if (ereg(“^[a-zA-Z0-9_]+@[a-zA-Z0-9\-]+\.[a-zA-Z0-9\-\.]+$”, $address))
    return true;
  else
    return false;
}
```

Fungsi `filling_out()` mengharapkan untuk diberikan array variabel secara umum, ini akan berupa array `$HTTP_POST_VARS` atau `$HTTP_GET_VARS`. Fungsi ini akan memeriksa apakah semuanya sudah diisi, dan mengembalikan true jika sudah diisi dan false jika belum diisi. Fungsi `valid_email()` menggunakan ekspresi reguler yang telah dikembangkan untuk memvalidasi alamat email. Fungsi ini mengembalikan true jika alamat tampak valid, dan false jika tidak. Setelah kami memvalidasi data input, kami benar-benar dapat mencoba dan mendaftarkan pengguna. Jika Anda melihat kembali Listing 3.6, Anda akan melihat bahwa kami melakukannya sebagai berikut:

```
$reg_result = register($username, $email, $passwd);
if ($reg_result == “true”)
{
   // register session variable
   $valid_user = $username;
   session_register(“valid_user”);

   // provide link to members page
```

```
    do_html_header("Registration successful");
    echo "Your registration was successful.  Go to the members page "
          ."to start setting up your bookmarks!";
    do_HTML_URL("member.php", "Go to members page");
 }
```

Seperti yang dapat Anda lihat, kami memanggil fungsi `register()` dengan nama pengguna, alamat email, dan kata sandi yang telah dimasukkan. Jika berhasil, kami mendaftarkan nama pengguna sebagai variabel sesi dan menyediakan tautan ke halaman anggota utama kepada pengguna. Ini adalah output yang ditunjukkan pada Gambar 3.5.

Gambar 3.5 Pendaftaran Berhasil Pengguna Sekarang Dapat Membuka Halaman Anggota.

Fungsi `register()` ada di pustaka yang disertakan yang disebut `user_auth_fns.php`. Fungsi ini ditampilkan dalam Listing 3.9.

Listing 3.9 Fungsi `register()` dari `user_auth_fns.php`; Fungsi Ini Mencoba Menempatkan Informasi Pengguna Baru di Basis Data

```
function register($username, $email, $password)
// register new person with db
// return true or error message
{
  // connect to db
  $conn = db_connect();
  if (!$conn)
```

```
    return "Could not connect to database server - please try later.";

  // check if username is unique
  $result = mysql_query("select * from user where username='$username'");
  if (!$result)
    return "Could not execute query";
  if (mysql_num_rows($result)>0)
    return "That username is taken - go back and choose another one.";

  // if ok, put in db
  $result = mysql_query("insert into user values
                        ('$username', password('$password'), '$email')");

  if (!$result)
    return "Could not register you  in database - please try again later.";

  return true;
}
```

Tidak ada yang baru dalam fungsi ini fungsi ini terhubung ke basis data yang telah kita siapkan sebelumnya. Jika nama pengguna yang dipilih sudah diambil, atau basis data tidak dapat diperbarui, fungsi ini akan mengembalikan false. Jika tidak, fungsi ini akan memperbarui basis data dan mengembalikan true.

Satu hal yang perlu diperhatikan adalah bahwa kita melakukan koneksi basis data yang sebenarnya dengan fungsi yang telah kita tulis, yang disebut `db_connect()`. Fungsi ini hanya menyediakan satu lokasi yang berisi nama pengguna dan kata sandi untuk terhubung ke basis data. Dengan demikian, jika kita mengubah kata sandi basis data, kita hanya perlu mengubah satu file dalam aplikasi kita. Fungsi ini ditunjukkan dalam Listing 3.10.

Listing 3.10 Fungsi `db_connect()` dari `db_fns.php`; Fungsi Ini Terhubung ke Basis Data MySQL

```
$result = mysql_query("insert into user values
                  ('$username', password('$password'), '$email')");
if (!$result)
      return "Could not register you  in database - please try again later.";
      return true;
}
```

Saat pengguna terdaftar, mereka dapat masuk dan keluar menggunakan halaman masuk dan keluar biasa. Kita akan membangunnya selanjutnya.

Login

Jika pengguna mengetikkan detail mereka ke dalam formulir di `login.php` (lihat Gambar 3.3) dan mengirimkannya, mereka akan dibawa ke skrip yang disebut `member.php`. Skrip ini akan memasukkan mereka ke dalam formulir jika mereka masuk. Skrip ini juga akan

menampilkan bookmark yang relevan kepada pengguna yang masuk. Ini adalah pusat dari seluruh aplikasi. Skrip ini ditampilkan dalam Listing 3.11.

Listing 3.11 member.php; Script ini adalah Hub Utama Aplikasi

```
<?

 // include function files for this application
 require_once("bookmark_fns.php");
 session_start();

if ($username && $passwd)
// they have just tried logging in
{

    if (login($username, $passwd))
    {
      // if they are in the database register the user id
      $valid_user = $username;
      session_register("valid_user");
    }
    else
    {
      // unsuccessful login
      do_html_header("Problem:");
      echo "You could not be logged in.
           You must be logged in to view this page.";
      do_html_url("login.php", "Login");
      do_html_footer();
      exit;
    }
}

do_html_header("Home");
check_valid_user();
// get the bookmarks this user has saved
if ($url_array = get_user_urls($valid_user));
  display_user_urls($url_array);

// give menu of options
display_user_menu();

do_html_footer();

?>
```

Pertama, kita periksa apakah pengguna datang dari halaman depan—dengan kata lain, apakah pengguna baru saja mengisi formulir login dan coba login pengguna dengan cara berikut:

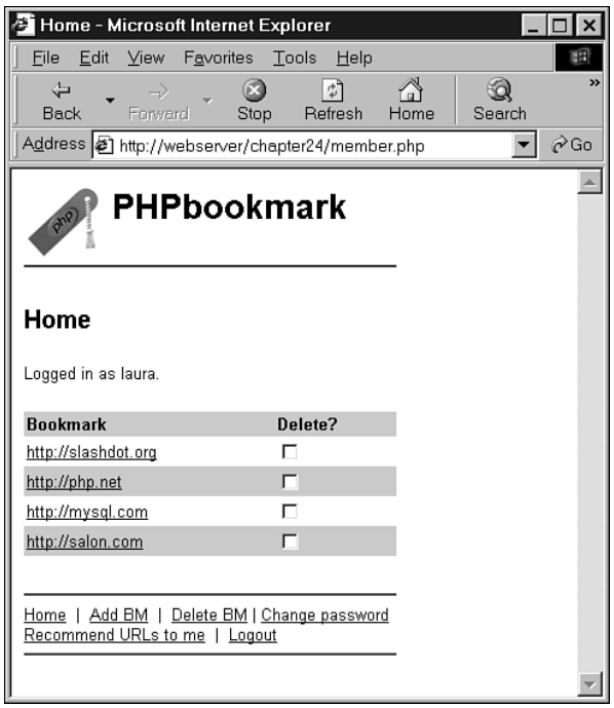
```
if ($username && $passwd)
// they have just tried logging in
{
    if (login($username, $passwd))
    {
      // if they are in the database register the user id
      $valid_user = $username;
      session_register("valid_user");
}
```

Anda dapat melihat bahwa kami mencoba untuk login menggunakan fungsi yang disebut `login()`. Kami telah mendefinisikannya di pustaka `user_auth_fns.php`, dan kami akan melihat kodenya sebentar lagi. Jika dia berhasil login, kami mendaftarkan sesinya seperti yang kami lakukan sebelumnya, menyimpan nama pengguna dalam variabel sesi `$valid_user`. Jika semuanya berjalan lancar, kami kemudian menunjukkan halaman anggota kepada pengguna:

```
do_html_header("Home");
check_valid_user();
// get the bookmarks this user has saved
if ($url_array = get_user_urls($valid_user));
  display_user_urls($url_array);

// give menu of options
display_user_menu();

do_html_footer();
```

Halaman ini dibentuk lagi menggunakan fungsi output. Anda akan melihat bahwa kami menggunakan beberapa fungsi baru lainnya. Fungsi-fungsi tersebut adalah `check_valid_user()`, dari `user_auth_fns.php`; `get_user_urls()`, dari `url_fns.php`; dan `display_user_urls()` dari `output_fns.php`. Fungsi `check_valid_user()` memeriksa apakah pengguna saat ini memiliki sesi terdaftar. Fungsi ini ditujukan untuk pengguna yang belum masuk, tetapi sedang berada di tengah sesi. Fungsi `get_user_urls()` mengambil bookmark pengguna dari basis data, dan `display_user_urls()` mengeluarkan bookmark ke browser dalam bentuk tabel. Kita akan membahas `check_valid_user()` sebentar lagi dan dua fungsi lainnya di bagian penyimpanan dan pengambilan bookmark.

Skrip member.php mengakhiri halaman dengan menampilkan menu dengan fungsi display_user_menu(). Beberapa contoh keluaran yang ditampilkan oleh member.php ditunjukkan pada Gambar 3.6.



Gambar 3.6 Skrip Member.Php Memeriksa Apakah Pengguna Telah Login; Mengambil Dan Menampilkan Bookmark-Nya; Dan Memberinya Menu Pilihan.

Sekarang kita akan melihat fungsi login() dan check_valid_user() lebih dekat. Fungsi login() ditunjukkan pada listing 3.12.

Listing 3.12 Fungsi login() dari user_auth_fns.php; Fungsi Ini Memeriksa Rincian Pengguna terhadap Basis Data

```
function login($username, $password)
// check username and password with db
// if yes, return true
// else return false
{
    // connect to db
    $conn = db_connect();
    if (!$conn)
        return 0;

// check if username is unique
$result = mysql_query("select * from user
                where username='$username'
                and passwd = password('$password')");
if (!$result)
```

```
      return 0;
if (mysql_num_rows($result)>0)
      return 1;
else
      return 0;
}
```

Seperti yang dapat Anda lihat, fungsi ini terhubung ke basis data dan memeriksa apakah ada pengguna dengan kombinasi nama pengguna dan kata sandi yang diberikan. Fungsi ini akan mengembalikan true jika ada, atau false jika tidak ada atau jika kredensial pengguna tidak dapat diperiksa. Fungsi `check_valid_user()` tidak terhubung ke basis data lagi, tetapi hanya memeriksa apakah pengguna memiliki sesi terdaftar, yaitu, bahwa ia telah masuk. Fungsi ini ditunjukkan dalam Listing 3.13.

Listing 3.13 Fungsi `check_valid_user()` dari `user_auth_fns.php`—Fungsi Ini Memeriksa Apakah Pengguna Memiliki Sesi yang Valid

```
function check_valid_user()
// see if somebody is logged in and notify them if not
{
  global $valid_user;
  if (session_is_registered("valid_user"))
  {
    echo "Logged in as $valid_user.";
    echo "<br>";
  }
  else
  {
    // they are not logged in
    do_html_heading("Problem:");
    echo "You are not logged in.<br>";
    do_html_url("login.php", "Login");
    do_html_footer();
    exit;
  }
}
```

Jika pengguna tidak masuk, fungsi tersebut akan memberitahu pengguna bahwa ia harus masuk untuk melihat halaman ini, dan memberinya tautan ke halaman masuk.

LogOut

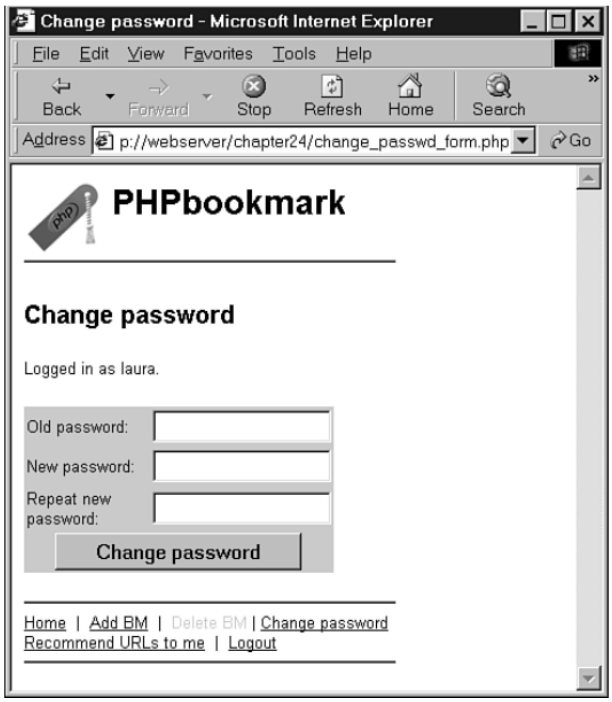
Anda mungkin telah memperhatikan bahwa ada tautan bertanda "Keluar" pada menu di Gambar 3.6. Ini adalah tautan ke skrip logout.php. Kode untuk skrip ini ditunjukkan pada Lsiting 3.14.

Listing 3.14 logout.php; Skrip Ini Mengakhiri Sesi Pengguna

```
<?
// include function files for this application
require_once("bookmark_fns.php");
session_start();
$old_user = $valid_user;  // store  to test if they *were* logged in
$result_unreg = session_unregister("valid_user");
$result_dest = session_destroy();

// start output html
do_html_header("Logging Out");

if (!empty($old_user))
{
  if ($result_unreg && $result_dest)
  {
    // if they were logged in and are now logged out
    echo "Logged out.<br>";
    do_html_url("login.php", "Login");
  }
  else
  {
    // they were logged in and could not be logged out
    echo "Could not log you out.<br>";
  }
}
else
{
  // if they weren't logged in but came to this page somehow
  echo "You were not logged in, and so have not been logged out.<br>";
  do_html_url("login.php", "Login");
}
do_html_footer();
?>
```

3.4 MENGUBAH KATA SANDI

Jika pengguna mengikuti opsi menu "Ubah Kata Sandi", ia akan disajikan dengan formulir yang ditunjukkan pada Gambar 3.7. Formulir ini dibuat oleh skrip change_passwd_form.php. Ini adalah skrip sederhana yang hanya menggunakan fungsi dari pustaka keluaran, jadi kami tidak menyertakan sumbernya di sini. Ketika formulir ini dikirimkan, skrip change_passwd.php akan dipicu, yang ditunjukkan pada listing 3.15.



Gambar 3.7 Skrip Change_Passwd_Form.Php Menyediakan Formulir Tempat Pengguna Dapat Mengubah Kata Sandi Mereka.

Listing 3.15 change_passwd.php; Skrip Ini Mencoba Mengubah Kata Sandi Pengguna

```
<?
  require_once("bookmark_fns.php");
  session_start();
  do_html_header("Changing password");
  check_valid_user();
  if (!filled_out($HTTP_POST_VARS))
  {
    echo "You have not filled out the form completely.
         Please try again.";

    display_user_menu();
    do_html_footer();
    exit;
  }
  else
  {
    if ($new_passwd!=$new_passwd2)
        echo "Passwords entered were not the same.  Not changed.";
    else if (strlen($new_passwd)>16 || strlen($new_passwd)<6)
        echo "New password must be between 6 and 16 characters.  Try again.";
    else
    {
      // attempt update
      if (change_password($valid_user, $old_passwd, $new_passwd))
      echo "Password changed.";
```

```
        else
          echo “Password could not be changed.”;
        }

  }
    display_user_menu();
    do_html_footer();
?>
```

Skrip ini memeriksa apakah pengguna sudah masuk (menggunakan `check_valid_user()`), bahwa mereka telah mengisi formulir kata sandi (menggunakan `filling_out()`), dan bahwa kata sandi baru tersebut sama dan memiliki panjang yang tepat. Semua ini bukanlah hal baru. Jika semuanya berjalan lancar, skrip ini akan memanggil fungsi `change_password()` sebagai berikut:

```
if (change_password($valid_user, $old_passwd, $new_passwd))
  echo “Password changed.”;
else
  echo “Password could not be changed.”;
```

Fungsi ini berasal dari pustaka `user_auth_fns.php` kami, dan kode untuk fungsi ini ditampilkan dalam Listing 3.16.

Listing 3.16 Fungsi `change_password()` dari `user_auth_fns.php`; Fungsi Ini Mencoba Memperbarui Kata Sandi Pengguna dalam Basis Data

```
function change_password($username, $old_password, $new_password)
// change password for username/old_password to new_password
// return true or false
  {
    // if the old password is right
    // change their password to new_password and return true
    // else return false
    if (login($username, $old_password))
    {
      if (!($conn = db_connect()))
        return false;
      $result = mysql_query( “update user
                              set passwd = password(‘$new_password’)
                              where username = ‘$username’”);
      if (!$result)
        return false;  // not changed
      else
        return true;  // changed successfully
    }
    else
```

```
    return false; // old password was wrong
}
```

Fungsi ini memeriksa apakah kata sandi lama yang diberikan sudah benar, menggunakan fungsi `login()` yang telah kita bahas sebelumnya. Jika benar, maka fungsi tersebut terhubung ke basis data dan memperbarui kata sandi ke nilai yang baru.

3.5 MENGATUR ULANG KATA SANDI YANG TERLUPAKAN

Selain mengubah kata sandi, kita perlu menangani situasi umum saat pengguna lupa kata sandinya. Perhatikan bahwa di halaman depan, `login.php`, kita menyediakan tautan bagi pengguna dalam situasi ini, yang bertanda, "*Lupa kata sandi Anda?*" Tautan ini akan membawa pengguna ke skrip yang disebut `forget_form.php`, yang menggunakan fungsi output untuk menampilkan formulir seperti yang ditunjukkan pada Gambar 3.8. Skrip ini sangat sederhana, hanya menggunakan fungsi output, jadi kita tidak akan membahasnya di sini. Saat formulir dikirimkan, skrip tersebut memanggil skrip `forget_passwd.php`, yang lebih menarik. Skrip ini ditunjukkan pada Daftar 3.17.

Gambar 3.8 Skrip Forget_Form.Php Menyediakan Formulir Yang Dapat Digunakan Pengguna Untuk Meminta Agar Kata Sandi Mereka Direset Dan Dikirimkan Kepada Mereka.

Listing 3.17 `forget_passwd.php`; Skrip Ini Mereset Kata Sandi Pengguna ke Nilai Acak dan Mengirimkan Kata Sandi Baru Melalui Email

```
<?
  require_once("bookmark_fns.php");
  do_html_header("Resetting password");

  if ($password=reset_password($username))
  {
```

```
    if (notify_password($username, $password))
      echo "Your new password has been sent to your email address.";
    else
      echo "Your password could not be mailed to you."
           ." Try pressing refresh.";
    }
    else
     echo "Your password could not be reset - please try again later.";

    do_html_url("login.php", "Login");

  do_html_footer();
?>
```

Seperti yang dapat Anda lihat, skrip ini menggunakan dua fungsi utama untuk menjalankan tugasnya: `reset_password()` dan `notify_password()`. Mari kita bahas masing-masing fungsi ini secara bergantian. Fungsi `reset_password()` menghasilkan kata sandi acak untuk pengguna dan memasukkannya ke dalam basis data. Kode untuk fungsi ini ditunjukkan pada Listing 24.18.

Listing 3.18 Fungsi `reset_password()` dari `user_auth_fns.php`; Skrip Ini Mengatur Ulang Kata Sandi Pengguna ke Nilai Acak dan Mengirimkan Kata Sandi Baru Melalui Email

```
function reset_password($username)
// set password for username to a random value
// return the new password or false on failure
{
  // get a random dictionary word b/w 6 and 13 chars in length
  $new_password = get_random_word(6, 13);
  // add a number  between 0 and 999 to it
  // to make it a slightly better password
  $rand ((double) microtime() * 1000000);
  $rand_number = rand(0, 999);
  $new_password .= $rand_number;

  // set user's password to this in database or return false
  if (!($conn = db_connect()))
       return false;
  $result = mysql_query("update user
                         set passwd = password('$new_password')
                         where username = '$username'");
  if (!$result)
    return false;  // not changed
  else
    return $new_password;  // changed successfully
 }
```

Fungsi ini menghasilkan kata sandi acak dengan mengambil kata acak dari kamus, menggunakan fungsi `get_random_word()` dan menambahkan angka acak antara 0 dan 999. Fungsi `get_random_word()` juga ada di pustaka `user_auth_fns.php`. Fungsi ini ditampilkan dalam Listing 3.19.

Listing 3.19 Fungsi `get_random_word()` dari `user_auth_fns.php`; Fungsi Ini Mendapatkan Kata Acak dari Kamus untuk Digunakan dalam Menghasilkan Kata Sandi

```
function get_random_word($min_length, $max_length)
// grab a random word from dictionary between the two lengths
// and return it
{
  // generate a random word
  $word = "";
  $dictionary = "/usr/dict/words"; // the ispell dictionary
  $fp = fopen($dictionary, "r");
  $size = filesize($dictionary);

  // go to a random location in dictionary
  $rand ((double) microtime() * 1000000);
  $rand_location = rand(0, $size);
  fseek($fp, $rand_location);

  // get the next whole word of the right length in the file
  while (strlen($word)< $min_length || strlen($word)>$max_length)
  {
    if (feof($fp))
      fseek($fp, 0); // if at end, go to start
      $word = fgets($fp, 80); // skip first word as it could be partial
      $word = fgets($fp, 80); // the potential password
  };
  $word=trim($word); // trim the trailing \n from fgets
  return $word;
}
```

Agar berfungsi, fungsi ini memerlukan kamus. Jika Anda menggunakan sistem UNIX, pemeriksa ejaan bawaan ispell dilengkapi dengan kamus kata, yang biasanya terletak di /usr/dict/words, seperti di sini. Jika Anda menggunakan sistem lain atau tidak ingin menginstal ispell, jangan khawatir!

File-file ini diformat dengan setiap kata pada baris terpisah, dipisahkan oleh baris baru. Untuk mendapatkan kata acak dari file ini, kami memilih lokasi acak antara 0 dan ukuran file, dan membaca dari file di sana. Jika kita membaca dari lokasi acak ke baris baru berikutnya, kemungkinan besar kita hanya akan memperoleh sebagian kata, jadi kita lewati baris tempat kita membuka berkas, dan mengambil kata berikutnya sebagai kata kita dengan memanggil

fgets() dua kali. Fungsi ini memiliki dua hal yang cerdas. Yang pertama adalah, jika kita mencapai akhir berkas saat mencari sebuah kata, kita kembali ke awal:

```
if (feof($fp))
       fseek($fp, 0);        // if at end, go to start
```

Yang kedua adalah kita dapat mencari kata dengan panjang tertentu kita memeriksa setiap kata yang kita tarik dari kamus, dan, jika tidak berada di antara $min_length dan $max_length, kita terus mencari. Kembali ke reset_password(), setelah kita membuat kata sandi baru, kita memperbarui basis data untuk mencerminkan hal ini, dan mengembalikan kata sandi baru tersebut ke skrip utama. Ini kemudian akan diteruskan ke notify_password(), yang akan mengirimkannya melalui email kepada pengguna. Mari kita lihat fungsi notify_password(), yang ditunjukkan pada Listing 3.20.

Listing 3.20 Fungsi Notify_Password() Dari User_Auth_Fns.Php; Fungsi Ini Mengirimkan Kata Sandi Yang Direset Melalui Email Kepada Pengguna

```
function notify_password($username, $password)
// notify the user that their password has been changed
{
  if (!($conn = db_connect()))
    return false;
  $result = mysql_query("select email from user
                        where username='$username'");
  if (!$result)
    return false;  // not changed
  else if (mysql_num_rows($result)==0)
    return false; // username not in db
  else
  {
    $email = mysql_result($result, 0, "email");
    $from = "From: support@phpbookmark \r\n";
    $mesg = "Your PHPBookmark password has been changed to $password \r\n"
          ."Please change it next time you log in. \r\n";
  if (mail($email, "PHPBookmark login information", $mesg, $from))
    return true;
  else
    return false;
  }
}
```

Dalam fungsi ini, jika diberikan nama pengguna dan kata sandi baru, kita tinggal mencari alamat email pengguna tersebut di database, lalu menggunakan fungsi mail() PHP untuk mengirimkannya kepada pengguna tersebut. Akan lebih aman untuk memberikan pengguna kata sandi yang benar-benar acak terbuat dari kombinasi huruf besar dan kecil,

angka, dan tanda baca daripada kata dan angka acak kami. Namun, kata sandi seperti 'zigzag487' akan lebih mudah dibaca dan diketik oleh pengguna kami daripada kata sandi yang benar-benar acak. Sering kali membingungkan bagi pengguna untuk menentukan apakah karakter dalam string acak adalah 0 atau O (nol atau O kapital), atau 1 atau l (satu atau L huruf kecil).

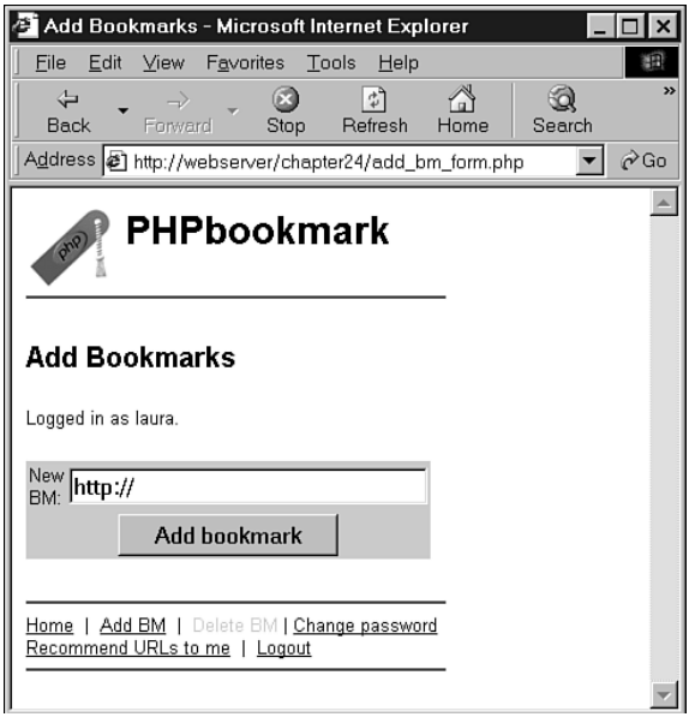
Pada sistem kami, berkas kamus berisi sekitar 45.000 kata. Jika seorang peretas tahu bagaimana kami membuat kata sandi, dan tahu nama pengguna, ia masih harus mencoba 22.500.000 kata sandi rata-rata untuk menebak satu kata sandi. Tingkat keamanan ini tampaknya memadai untuk jenis aplikasi ini bahkan jika pengguna kami mengabaikan saran yang dikirim melalui email untuk mengubahnya.

3.6 MENERAPKAN PENYIMPANAN DAN PENGAMBILAN BOOKMARK

Sekarang kita akan melanjutkan dan melihat bagaimana bookmark pengguna disimpan, diambil, dan dihapus.

Menambahkan Bookmark

Pengguna dapat menambahkan bookmark dengan mengeklik tautan Tambahkan Bookmark di menu pengguna. Ini akan membawa mereka ke formulir yang ditunjukkan pada Gambar 3.9.



Gambar 3.9 Skrip Add_Bm_Form.Php Menyediakan Formulir Tempat Pengguna Dapat Menambahkan Bookmark Ke Halaman Bookmark Mereka.

Sekali lagi, skrip ini sederhana dan hanya menggunakan fungsi output, jadi kita tidak akan membahasnya di sini. Saat formulir dikirimkan, skrip tersebut memanggil skrip add_bms.php, yang ditunjukkan pada listing 3.21.

Listing 3.21 add_bms.php; Skrip Ini Menambahkan Bookmark Baru ke Halaman Pribadi Pengguna

```
<?
  require_once("bookmark_fns.php");
  session_start();
  do_html_header("Adding bookmarks");
  check_valid_user();
  if (!filled_out($HTTP_POST_VARS))
    {
  echo "You have not filled out the form completely.
  Please try again.";
  display_user_menu();
  do_html_footer();
      exit;
 }
 else
 {
       // check URL format
       if (strstr($new_url, "http://")===false)
       $new_url = "http://".$new_url;

 // check URL is valid
 if (@fopen($new_url, "r"))
 {
  // try to add bm
  if (add_bm($new_url))
  echo "Bookmark added.";
  else
  echo "Could not add bookmark.";
 }
 else
  echo "Not a valid URL.";
}

// get the bookmarks this user has saved
if ($url_array = get_user_urls($valid_user));
  display_user_urls($url_array);
  display_user_menu();
  do_html_footer();
?>
```

Sekali lagi skrip ini mengikuti pola validasi, entri basis data, dan keluaran. Untuk memvalidasi, pertama-tama kita periksa apakah pengguna telah mengisi formulir menggunakan `filling_out()`.

Kemudian kita lakukan dua pemeriksaan URL. Pertama, menggunakan `strstr()`, kita lihat apakah URL dimulai dengan `http://`. Jika tidak, kita tambahkan ini ke awal URL. Setelah

kita melakukan ini, kita dapat benar-benar memeriksa apakah URL benar-benar ada. Seperti yang mungkin Anda ingat dari Bab 17, "Menggunakan Fungsi Jaringan dan Protokol," kita dapat menggunakan `fopen()` untuk membuka URL yang dimulai dengan `http://`. Jika kita dapat membuka berkas ini, kita asumsikan URL tersebut valid dan memanggil fungsi `add_bm()` untuk menambahkannya ke basis data.

Perlu dicatat bahwa `fopen()` hanya akan dapat membuka berkas jika server Anda memiliki akses langsung ke Internet. Jika perlu mengakses server HTTP lain melalui server proxy, `fopen()` tidak akan berfungsi. Fungsi ini dan fungsi lainnya yang terkait dengan bookmark semuanya ada di pustaka fungsi `url_fns.php`. Anda dapat melihat kode untuk fungsi `add_bm()` di Listing 3.22.

Listing 3.22 Fungsi `Add_Bm()` Dari `Url_Fns.Php`—Fungsi Ini Menambahkan Bookmark Baru Ke Basis Data

```
function add_bm($new_url)
{
  // Add new bookmark to the database

    echo "Attempting to add ".htmlspecialchars($new_url)."<BR>";
    global $valid_user;
    if (!($conn = db_connect()))
      return false;

    // check not a repeat bookmark
    $result = mysql_query("select * from bookmark
                           where username='$valid_user'
                           and bm_URL='$new_url'");
if ($result && (mysql_num_rows($result)>0))
    return false;

// insert the new bookmark
if (!mysql_query( "insert into bookmark values
                          ('$valid_user', '$new_url')"))
     return false;
return true;
}
```

Fungsi ini cukup sederhana. Fungsi ini memeriksa apakah pengguna belum mencantumkan bookmark ini di dalam basis data. (Meskipun kecil kemungkinan mereka akan memasukkan bookmark dua kali, ada kemungkinan dan bahkan besar kemungkinan mereka akan menyegarkan halaman.) Jika bookmark tersebut baru, maka bookmark tersebut dimasukkan ke dalam basis data. Jika melihat kembali `add_bm.php`, Anda dapat melihat bahwa hal terakhir yang dilakukannya adalah memanggil `get_user_urls()` dan `display_user_urls()`, sama seperti `member.php`. Kita akan melanjutkan dan melihat fungsi-fungsi ini selanjutnya.

3.7 MENAMPILKAN BOOKMARK

Dalam skrip member.php dan fungsi add_bm(), kita menggunakan fungsi get_user_urls() dan display_user_urls(). Fungsi-fungsi ini mengambil bookmark pengguna dari basis data dan menampilkannya, masing-masing. Fungsi get_user_urls() ada di pustaka url_fns.php, dan fungsi display_user_urls() ada di pustaka output_fns.php. Fungsi get_user_urls() ditunjukkan pada Listing 3.23.

Listing 3.23 Fungsi Get_User_Urls() Dari Url_Fns.Php; Fungsi Ini Mengambil Bookmark Milik Pengguna Dari Database

```
function get_user_urls($username)
{
  // extract from the database all the URLs this user has stored
  if (!($conn = db_connect()))
    return false;
  $result = mysql_query( "select bm_URL
                          from bookmark
                          where username = '$username'");
if (!$result)
return false;

//create an array of the URLs
$url_array = array();
for ($count = 1; $row = mysql_fetch_row ($result); ++$count)
{
    $url_array[$count] = $row[0];
  }
  return $url_array;
};
```

Mari kita bahas fungsi ini secara singkat. Fungsi ini mengambil nama pengguna sebagai parameter, dan mengambil bookmark untuk pengguna tersebut dari basis data. Fungsi ini akan mengembalikan array URL ini atau false jika bookmark tidak dapat diambil. Array dari get_user_urls() dapat diteruskan ke display_user_urls(). Ini lagi-lagi merupakan fungsi keluaran HTML sederhana untuk mencetak URL pengguna dalam format tabel yang bagus, jadi kita tidak akan membahasnya di sini. Lihat kembali Gambar 3.6 untuk melihat seperti apa keluarannya. Fungsi ini sebenarnya memasukkan URL ke dalam formulir. Di samping setiap URL terdapat kotak centang yang memungkinkan penanda ditandai untuk dihapus. Kita akan membahasnya selanjutnya.

Menghapus Penanda

Saat pengguna menandai beberapa penanda untuk dihapus dan mengklik opsi Hapus Penanda di menu, formulir yang berisi URL akan dikirimkan. Setiap kotak centang dihasilkan oleh kode berikut dalam fungsi display_user_urls():

```
echo “<td><input type=checkbox name=\”del_me[]\”
            value=\”$url\”></td>”;
```

Nama setiap input adalah `del_me[]`. Ini berarti bahwa, dalam skrip PHP yang diaktifkan oleh formulir ini, kita akan memiliki akses ke array yang disebut `$del_me` yang akan berisi semua bookmark yang akan dihapus. Mengklik opsi Delete BM akan mengaktifkan skrip `delete_bms.php`. Skrip ini ditampilkan dalam Listing 3.12.

Listing 3.24 `Delete_Bms.Php`; Skrip Ini Menghapus Bookmark Dari Basis Data

```
<?
  require_once(“bookmark_fns.php”);
  session_start();
  do_html_header(“Deleting bookmarks”);
  check_valid_user();
  if (!filled_out($HTTP_POST_VARS))
  {
    echo “ You have not chosen any bookmarks to delete.
          Please try again.”;
    display_user_menu();
    do_html_footer();
    exit;
  }
  else
  {
    if (count($del_me) >0)
    {
      foreach($del_me as $url)
      {
      if (delete_bm($valid_user, $url))
        echo “Deleted “.htmlspecialchars($url).”.<br>”;
      else
        echo “Could not delete “.htmlspecialchars($url).”.<br>”;
      }
    }
  else
    echo “No bookmarks selected for deletion”;
  }

  // get the bookmarks this user has saved
  if ($url_array = get_user_urls($valid_user));
  display_user_urls($url_array);

  display_user_menu();
  do_html_footer();
?>
```

Kami memulai skrip ini dengan melakukan validasi biasa. Ketika kami mengetahui bahwa pengguna telah memilih beberapa bookmark untuk dihapus, kami menghapusnya dalam loop berikut:

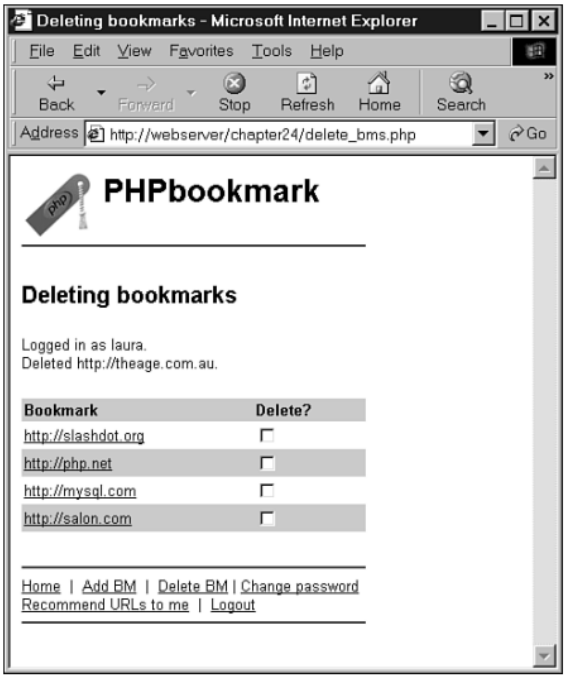
```
foreach($del_me as $url)
{
  if (delete_bm($valid_user, $url))
    echo "Deleted ".htmlspecialchars($url)."."<br>";
  else
    echo "Could not delete ".htmlspecialchars($url)."."<br>";
}
```

Seperti yang dapat Anda lihat, fungsi `delete_bm()` melakukan pekerjaan sebenarnya untuk menghapus bookmark dari database. Fungsi ini ditunjukkan pada Listing 3.25.

Listing 3.25 Fungsi `delete_bm()` di `url_fns.php`—Fungsi Ini Menghapus Satu Bookmark dari Daftar Pengguna

```
function delete_bm($user, $url)
{
  // delete one URL from the database
  if (!($conn = db_connect()))
    return false;

  // delete the bookmark
  if (!mysql_query( "delete from bookmark
                  where username='$user' and bm_url='$url'"))
    return false;
  return true;
}
```

Seperti yang Anda lihat, ini lagi-lagi merupakan fungsi yang cukup sederhana. Fungsi ini mencoba menghapus bookmark untuk pengguna tertentu dari basis data. Satu hal yang perlu diperhatikan adalah kita ingin menghapus pasangan nama pengguna-bookmark tertentu. Pengguna lain mungkin masih memiliki URL ini yang di-bookmark. Beberapa contoh keluaran dari menjalankan skrip hapus pada sistem kami ditunjukkan pada Gambar 3.10.



Gambar 3.10 Skrip Penghapusan Memberitahukan Pengguna Tentang Bookmark Yang Dihapus Dan Kemudian Menampilkan Bookmark Yang Tersisa.

Seperti pada skrip `add_bms.php`, ketika perubahan pada basis data telah dilakukan, kami menampilkan daftar bookmark baru menggunakan `get_user_urls()` dan `display_user_urls()`.

Mengimplementasikan Rekomendasi

Akhirnya, kita sampai pada skrip rekomendasi tautan, `recommend.php`. Ada banyak cara berbeda yang dapat kita lakukan untuk membuat rekomendasi. Kami telah memutuskan untuk melakukan apa yang kami sebut rekomendasi "sepemikiran". Yaitu, kami akan mencari pengguna lain yang memiliki setidaknya satu bookmark yang sama dengan pengguna yang kami berikan. Bookmark lain dari pengguna lain tersebut mungkin juga menarik bagi pengguna yang kami berikan. Cara termudah untuk mengimplementasikan ini sebagai kueri SQL adalah dengan menggunakan subkueri. Pertama, kami mendapatkan daftar pengguna yang serupa dalam subkueri, lalu kami melihat bookmark mereka dalam kueri luar.

Namun, seperti yang mungkin Anda ingat, MySQL tidak mendukung subkueri. Kita harus menjalankan dua kueri berbeda dan memasukkan output dari kueri pertama ke kueri berikutnya. Kita dapat melakukannya dengan menyiapkan tabel sementara dengan hasil dari kueri pertama, atau dengan memproses hasil kueri pertama melalui PHP. Kita telah memilih pendekatan kedua. Kedua pendekatan tersebut memiliki kelebihan. Menggunakan tabel sementara mungkin sedikit lebih cepat, tetapi pemrosesan dalam PHP memudahkan pengujian dan modifikasi kode. Kita mulai dengan menjalankan kueri berikut:

```
select distinct(b2.username)
from bookmark b1, bookmark b2
where b1.username=’$valid_user’
```

```
and b1.username != b2.username
and b1.bm_URL = b2.bm_URL
```

Kueri ini menggunakan alias untuk menggabungkan bookmark tabel basis data dengan dirinya sendiri konsep yang aneh tetapi terkadang berguna. Bayangkan bahwa sebenarnya ada dua tabel bookmark, satu disebut b1 dan satu disebut b2. Di b1, kita melihat pengguna saat ini dan bookmark-nya. Di tabel lainnya, kita melihat bookmark semua pengguna lainnya. Kita mencari pengguna lain (`b2.username`) yang memiliki URL yang sama dengan pengguna saat ini (`b1.bm_URL = b2.bm_URL`) dan bukan pengguna saat ini (`b1.username != b2.username`). Kueri ini akan memberi kita daftar orang-orang yang berpikiran sama dengan pengguna kita saat ini. Berbekal daftar ini, kita dapat mencari bookmark mereka yang lain dengan kueri berikut:

```
select bm_URL
from bookmark
where username in $sim_users
and bm_URL not in $user_urls
group by bm_URL
having count(bm_URL)>$popularity
```

Variabel `$sim_users` berisi daftar pengguna yang memiliki pemikiran yang sama. Variabel `$user_urls` berisi daftar bookmark pengguna saat ini jika b1 sudah memiliki bookmark, tidak ada gunanya merekomendasikannya kepadanya. Terakhir, kami menambahkan beberapa penyaringan dengan variabel `$popularity` kami tidak ingin merekomendasikan URL yang terlalu pribadi, jadi kami hanya menyarankan URL yang telah di-bookmark oleh sejumlah pengguna lain.

Jika kami mengantisipasi banyak pengguna yang menggunakan sistem kami, kami dapat menaikkan $popularity untuk hanya menyarankan URL yang telah di-bookmark oleh sejumlah besar pengguna. URL yang di-bookmark oleh banyak orang mungkin memiliki kualitas yang lebih tinggi dan tentu saja memiliki daya tarik yang lebih umum daripada halaman Web pada umumnya. Skrip lengkap untuk membuat rekomendasi ditunjukkan pada listing 3.26 dan 3.27. Skrip utama untuk membuat rekomendasi disebut `recommend.php` (lihat Daftar 24.26). Skrip ini memanggil fungsi pemberi rekomendasi `recommend_urls()` dari `url_fns.php` (lihat listing 3.27).

Daftar 3.26 `Recommend.Php`; Skrip Ini Menyarankan Beberapa Bookmark Yang Mungkin Disukai Pengguna

```
<?
  require_once("bookmark_fns.php");
  session_start();
  do_html_header("Recommending URLs");
  check_valid_user();
  $urls = recommend_urls($valid_user);
```

```
  display_recommended_urls($urls);
  display_user_menu();
  do_html_footer();
?>
```

Listing 3.27 Fungsi recommend_urls() dari url_fns.php—Skrip Ini Mengerjakan Rekomendasi Aktual

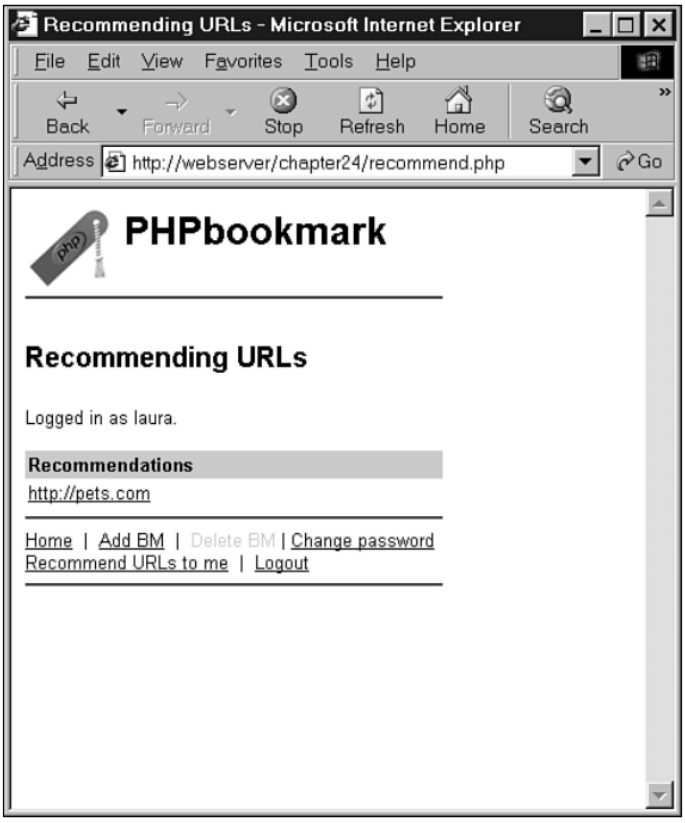
```
{
  // We will provide semi intelligent recommendations to people
  // If they have an URL in common with other users, they may like
  // other URLs that these people like
  if (!($conn = db_connect()))
     return false;

 // find other matching users
 // with an url the same as you

 if (!($result = mysql_query("
                    select distinct(b2.username)
                    from bookmark b1, bookmark b2
                    where b1.username='$valid_user'
                    and b1.username != b2.username
                    and b1.bm_URL = b2.bm_URL
                    ")))
    return false;
if (mysql_num_rows($result)==0)
    return false;
```

Listing 3.27 Lanjutan

```
// create set of users with urls in common
// for use in IN clause
$row = mysql_fetch_object($result);
$sim_users = "('".($row->username)."'";
while ($row = mysql_fetch_object($result))
{
          $sim_users .= ", '".($row->username)."'";
}
$sim_users .= ")";

// create list of user urls
// to avoid replicating ones we already know about
if (!($result = mysql_query("
          select bm_URL
          from bookmark
          where username='$valid_user'")))
```

```
 return false;

// create set of user urls for use in IN clause
$row = mysql_fetch_object($result);
$user_urls = "('".($row->bm_URL)."'";
while ($row = mysql_fetch_object($result))
{
      $user_urls .= ", '".($row->bm_URL)."'";
}
$user_urls .= ")";

// as a simple way of excluding people's private pages, and
// increasing the chance of recommending appealing URLs, we
// specify a minimum popularity level
// if $popularity = 1, then more than one person must have
// an URL before we will recommend it

// find out max number of possible URLs
if (!($result = mysql_query("
                    select bm_URL
                    from bookmark
                    where username in $sim_users
                    and bm_URL not in $user_urls
                    group by bm_URL
                    having count(bm_URL)>$popularity
                    ")))
return false;

if (!($num_urls=mysql_num_rows($result)))
 return false;

$urls = array();
// build an array of the relevant urls
for ($count=0; $row = mysql_fetch_object($result); $count++)
{
           $urls[$count] = $row->bm_URL;
    }
    return $urls;
}
```

Beberapa contoh keluaran dari `recommend.php` ditunjukkan pada Gambar 3.11.



Gambar 3.11 Skrip Tersebut Merekomendasikan Bahwa Pengguna Ini Mungkin Menyukai `Pets.Com`. Dua Pengguna Lain Dalam Basis Data Yang Menyukai `Slashdot.Org` Telah Menandai Situs Ini.

Penutup dan Kemungkinan Ekstensi

Itulah fungsi dasar aplikasi PHPBookmark. Ada banyak kemungkinan ekstensi. Anda dapat mempertimbangkan untuk menambahkan

- Pengelompokan bookmark berdasarkan topik
- Tautan *"Tambahkan ini ke bookmark saya"* untuk rekomendasi
- Rekomendasi berdasarkan URL terpopuler dalam basis data, atau pada topik tertentu
- Antarmuka administratif untuk menyiapkan dan mengelola pengguna dan topik
- Cara untuk membuat bookmark yang direkomendasikan lebih cerdas atau lebih cepat
- Keluar otomatis pengguna setelah waktu tertentu
- Pemeriksaan kesalahan tambahan pada masukan pengguna Bereksperimenlah! Ini cara terbaik untuk belajar.

BAB 4
MEMBUAT KERANJANG BELANJA

4.1 MEMBANGUN KATALOG DARING

Kita sudah memiliki basis data untuk katalog Book-O-Rama. Namun, mungkin diperlukan beberapa perubahan dan penambahan untuk aplikasi ini. Salah satunya adalah menambahkan kategori buku, seperti yang dinyatakan dalam persyaratan. Kita juga perlu menambahkan beberapa informasi ke basis data kita yang sudah ada tentang alamat pengiriman, detail pembayaran, dan sebagainya. Kita sudah tahu cara membuat antarmuka ke basis data MySQL menggunakan PHP, jadi bagian solusi ini seharusnya cukup mudah.

Melacak Pembelian Pengguna Saat Berbelanja

Ada dua cara dasar untuk melacak pembelian pengguna saat berbelanja. Salah satunya adalah dengan memasukkan pilihannya ke dalam basis data kita, dan yang lainnya adalah dengan menggunakan variabel sesi. Menggunakan variabel sesi untuk melacak pilihan dari satu halaman ke halaman lainnya akan lebih mudah ditulis karena kita tidak perlu terus-menerus meminta informasi ini ke basis data. Ini juga akan menghindari situasi di mana kita berakhir dengan banyak data sampah di basis data dari pengguna yang hanya menjelajah dan berubah pikiran. Oleh karena itu, kita perlu merancang variabel sesi atau serangkaian variabel untuk menyimpan pilihan pengguna. Saat pengguna selesai berbelanja dan membayar pembeliannya, kita akan memasukkan informasi ini ke dalam basis data kita sebagai catatan transaksi. Kita juga dapat menggunakan data ini untuk memberikan ringkasan status keranjang belanja saat ini di salah satu sudut halaman, sehingga pengguna tahu kapan saja berapa banyak yang akan ia belanjakan.

Pembayaran

Dalam proyek ini, kita akan menjumlahkan pesanan pengguna dan mengambil rincian pengiriman. Kita tidak akan benar-benar memproses pembayaran. Ada banyak sekali sistem pembayaran yang tersedia, dan implementasi untuk masing-masing sistem berbeda. Kita akan menulis fungsi tiruan yang dapat diganti dengan antarmuka ke sistem pilihan Anda. Sistem pembayaran umumnya dijual di wilayah geografis yang lebih spesifik daripada buku ini. Cara kerja berbagai antarmuka pemrosesan waktu nyata umumnya serupa. Anda perlu mengatur akun pedagang dengan bank untuk kartu yang ingin Anda terima. Penyedia sistem pembayaran Anda akan menentukan parameter apa yang perlu Anda berikan ke sistem mereka.

Sistem pembayaran akan mengirimkan data Anda ke bank, dan mengembalikan kode keberhasilan dari salah satu dari berbagai jenis kode kesalahan. Sebagai imbalan atas penyampaian data Anda, gerbang pembayaran akan mengenakan biaya pengaturan atau biaya tahunan, dan biaya berdasarkan jumlah atau nilai transaksi Anda. Beberapa penyedia bahkan mengenakan biaya untuk transaksi yang ditolak. Sistem pembayaran yang Anda pilih akan memerlukan informasi dari pelanggan (seperti nomor kartu kredit), informasi pengenal dari Anda (untuk menentukan akun pedagang mana yang akan dikreditkan), dan jumlah total transaksi. Kita dapat menghitung total pesanan dari variabel sesi keranjang belanja pengguna.

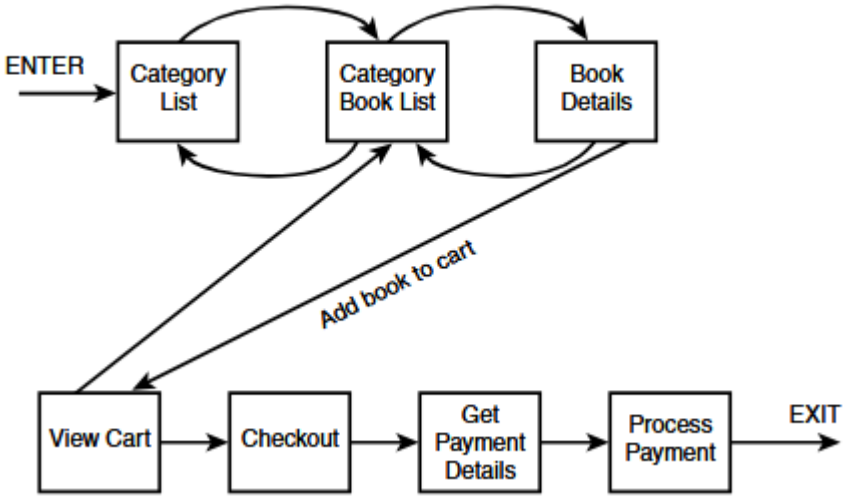
Kita akan mencatat detail pesanan akhir dalam basis data, dan menghapus variabel sesi pada saat itu.

4.2 ANTARMUKA ADMINISTRASI

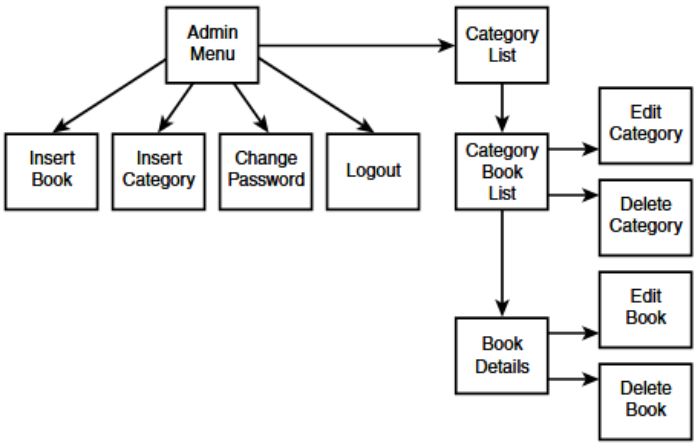
Selain semua ini, kita akan membangun antarmuka administrator yang akan memungkinkan kita menambahkan, menghapus, dan mengedit buku dan kategori dari basis data. Satu suntingan umum yang mungkin kita buat adalah mengubah harga suatu barang (misalnya, untuk penawaran atau penjualan khusus). Ini berarti bahwa ketika kita menyimpan pesanan pelanggan, kita juga harus menyimpan harga yang dibayarkannya untuk suatu barang. Akan menjadi mimpi buruk akuntansi jika satu-satunya catatan yang kita miliki adalah barang apa yang dipesan setiap pelanggan, dan berapa harga masing-masing barang saat ini. Ini juga berarti bahwa jika pelanggan harus mengembalikan atau menukar barang, kita akan memberinya jumlah kredit yang tepat. Kita tidak akan membangun antarmuka pemenuhan dan pelacakan pesanan untuk contoh ini. Anda dapat menambahkannya ke sistem dasar ini sesuai dengan kebutuhan Anda.

Ikhtisar Solusi

Mari kita gabungkan semua bagiannya. Ada dua tampilan dasar sistem: tampilan pengguna dan tampilan administrator. Setelah mempertimbangkan fungsionalitas yang dibutuhkan, kami membuat dua desain alur sistem, satu untuk setiap tampilan. Desain ini masing-masing ditunjukkan pada Gambar 4.1 dan Gambar 4.2.



Gambar 4.1 Tampilan Pengguna Sistem Book-O-Rama Memungkinkan Pengguna Menelusuri Buku Berdasarkan Kategori, Melihat Detail Buku, Menambahkan Buku Ke Keranjang Belanja, Dan Membelinya.



Gambar 4.2 Tampilan Administrator Sistem Book-O-Rama Memungkinkan Penyisipan, Penyuntingan, Dan Penghapusan Buku Dan Kategori.

Pada Gambar 4.1, kami menunjukkan tautan utama antara skrip di bagian pengguna situs. Pelanggan akan masuk terlebih dahulu ke halaman utama, yang mencantumkan semua kategori buku di situs. Dari sana, ia dapat membuka kategori buku tertentu, dan dari sana ke detail buku individual. Kami akan memberikan tautan kepada pengguna untuk menambahkan buku tertentu ke keranjangnya. Dari keranjang, ia akan dapat keluar dari toko daring. Gambar 4.2 menunjukkan antarmuka administrator—ini memiliki lebih banyak skrip, tetapi tidak banyak kode baru di sini. Skrip ini memungkinkan administrator masuk dan menyisipkan buku dan kategori.

Cara termudah untuk menerapkan penyuntingan dan penghapusan buku dan kategori adalah dengan menunjukkan kepada administrator versi antarmuka pengguna yang sedikit berbeda dengan situs tersebut. Administrator masih dapat menelusuri kategori dan buku, tetapi alih-alih memiliki akses ke keranjang belanja, administrator akan dapat membuka buku atau kategori tertentu dan mengedit atau menghapus buku atau kategori tersebut. Dengan membuat skrip yang sama sesuai untuk pengguna biasa dan administrator, kita dapat menghemat waktu dan tenaga.

Tiga modul kode utama untuk aplikasi ini adalah sebagai berikut:

- ✓ Katalog
- ✓ Keranjang belanja dan pemrosesan pesanan (Kami telah menggabungkan keduanya karena keduanya sangat terkait.)
- ✓ Administrasi.

Seperti pada proyek terakhir, kita juga akan membangun dan menggunakan sekumpulan pustaka fungsi. Untuk proyek ini, kita akan menggunakan API fungsi yang mirip dengan yang ada pada proyek terakhir. Kita akan mencoba membatasi bagian kode kita yang menghasilkan HTML ke dalam satu pustaka tunggal untuk mendukung prinsip pemisahan logika dan konten dan, yang lebih penting, untuk membuat kode kita lebih mudah dibaca dan dikelola. Kita juga perlu membuat beberapa perubahan kecil pada basis data Book-O-Rama untuk proyek ini. Ringkasan berkas dalam aplikasi ditunjukkan pada Tabel 4.1.

Tabel 4.1 Berkas dalam Aplikasi Keranjang Belanja

Nama	Modul	Keterangan
index.php	Katalog	Halaman depan situs utama untuk pengguna. Menunjukkan kepada pengguna daftar kategori dalam sistem
show_cat.php	Kataglog	Menunjukkan kepada pengguna semua buku dalam kategori tertentu.
show_book.php	Katalog	Menampilkan rincian pengguna buku tertentu
show_cart.php	Keranjang Belanja	Menunjukkan isi keranjang belanja kepada pengguna. Juga digunakan untuk menambahkan barang ke keranjang.
checkout.php	Keranjang Belanja	Memberikan detail pesanan lengkap kepada pengguna. Mendapatkan detail pengiriman.
purchase.php	Keranjang belanja	Mendapatkan rincian pembayaran dari pengguna.
process.php	Keranjang belanja	Memproses rincian pembayaran dan menambahkan pesanan ke basis data.
login.php	Administrasi	Memungkinkan adminstrator untuk masuk dan membuat perubahan
logout.php	Administrasi	Logout pengguna admin
admin.php	Administrasi	Menu administrasi utama
change_password_form.php	Administrasi	Formulir untuk mengizinkan administrator mengubah kata sandi lognya
change_password.php	Administrasi	Mengubah kata sandi adminstrator
insert_category_form.php	Administrasi	Formulir untuk mengizinkan administrator menambahkan kategori baru ke basis data.
insert_category.php	Administrasi	Memasukkan kategori baru ke dalam basis data
insert_category.php	Administrasi	Formulir untuk mengizinkan administrator menambah buku baru ke sistem
insert_book.php	Administrasi	Memasukkan buku baru ke dalam basis data
edit_category_form.php	Administrasi	Formulir untuk mengizinkan administrator mengedit kategori
edit_category.php	Administrasi	Kategori pembaruan dalam database
edit_book_form.php	Administrasi	Formulir untuk mengizinkan administrator mengedit rincian buku.
edit_book.php	Administrasi	Memperbarui buku di database
delete_category.php	Administrasi	Menghapus kategori dari basis data
delete_book.php	Administasi	Menghapus buku dari basis data
book_sc_fns.php	Fungsi	Kumpulan berkas penyertaan untuk aplikasi
admin_fns.php	Fungsi	Kumpulan fungsi yang digunakan oleh skrip adminstratif
book_fns.php	Fungsi	Kumpulan fungsi untuk menyimpan dan mengambil data buku

book_fns.php	Fungsi	Kumpulan fungsi untuk menyimpan dan mengambil data pesanan
output_fns.php	Fungsi	Kumpulan Fungsi untuk mengeluarkan HTML
data_valid_fns.php	Fungsi	Kumpulan fungsi untuk memvalidasi data masukan
db_fns.php	Fungsi	Kumpulan fungsi untuk menghubungkan ke basis data book_sc
user_auth_fns.php	Fungsi	Kumpulan fungsi untuk otentikasi pengguna administratif
book_sc.sql	SQL	SQL untuk menyiapkan basis data book_sc
populate.sql	SQL	SQL untuk memasukkan beberapa data sampel ke dalam basis data book_sc

4.3 MENERAPKAN BASIS DATA

Seperti yang telah kami sebutkan sebelumnya, SQL untuk membuat basis data book_sc ditunjukkan pada Listing 4.1.

Listing 4.1 Book_Sc.Sql—Sql Untuk Membuat Basis Data Book_Sc

```
create database book_sc;

use book_sc;

create table customers
(
  customerid int unsigned not null auto_increment primary key,
  name char(40) not null,
  address char(40) not null,
  city char(20) not null,
  state char(20),
  zip char(10),
  country char(20) not null
);

create table orders
(
  orderid int unsigned not null auto_increment primary key,
  customerid int unsigned not null,
  amount float(6,2),
  date date not null,
  order_status char(10),
  ship_name char(40) not null,
  ship_address char(40) not null,
  ship_city char(20) not null,
  ship_state char(20),
  ship_zip char(10),
```

```
  ship_country char(20) not null
);

create table books
(
  isbn char(13) not null primary key,
  author char(30),
  title char(60),
  catid int unsigned,
  price float(4,2) not null,
  description varchar(255)
);

create table categories
(
  catid int unsigned not null auto_increment primary key,
  catname char(40) not null
);

create table order_items
(
  orderid int unsigned not null,
  isbn char(13) not null,
  item_price float(4,2) not null,
  quantity tinyint unsigned not null,
  primary key (orderid, isbn)
);

create table admin
(
  username char(16) not null primary key,
  password char(16) not null
);

grant select, insert, update, delete
on book_sc.*
to book_sc@localhost identified by 'password';
```

Meskipun tidak ada yang salah dengan antarmuka Book-O-Rama yang asli, kami memiliki beberapa persyaratan lain sekarang karena kami akan membuatnya tersedia secara daring. Perubahan yang telah kami buat pada basis data asli adalah sebagai berikut:

- Penambahan lebih banyak kolom alamat untuk pelanggan ini lebih penting sekarang karena kami sedang membangun aplikasi yang lebih realistis.
- Penambahan alamat pengiriman ke pesanan. Alamat kontak pelanggan mungkin tidak sama dengan alamat pengiriman, terutama jika ia menggunakan situs untuk membeli hadiah.

- Penambahan tabel kategori dan catid ke tabel buku. Menyortir buku ke dalam kategori akan membuat situs lebih mudah dijelajahi.
- Penambahan `item_price` ke tabel `order_items` untuk mengenali fakta bahwa harga suatu barang mungkin berubah. Kami ingin mengetahui berapa biayanya saat pelanggan memesannya.
- Penambahan tabel admin untuk menyimpan detail login dan kata sandi administrator.
- Penghapusan tabel ulasan; Anda dapat menambahkan ulasan sebagai ekstensi untuk proyek ini. Sebagai gantinya, setiap buku memiliki kolom deskripsi yang akan berisi uraian singkat tentang buku tersebut.

Untuk menyiapkan basis data ini di sistem Anda, jalankan skrip `book_sc.sql` melalui MySQL sebagai pengguna root, sebagai berikut:

```
mysql -u root -p < book_sc.sql
```

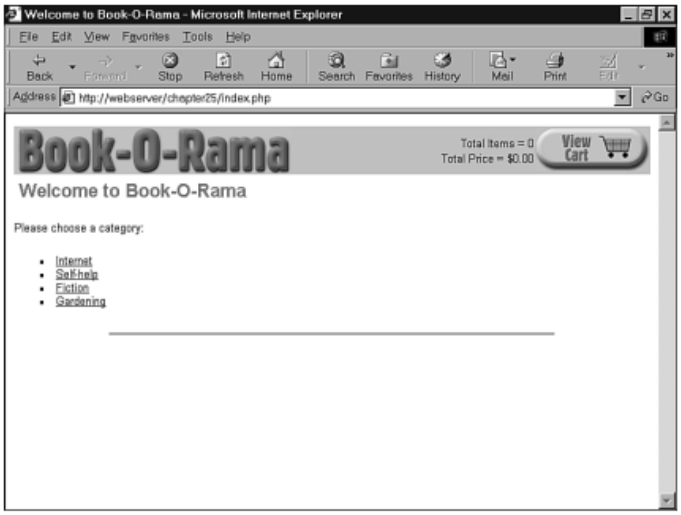
(Anda perlu memberikan kata sandi root Anda.)
Sebelumnya, Anda harus mengubah kata sandi untuk pengguna `book_sc` menjadi sesuatu yang lebih baik daripada

```
'password'.
```

Kami juga telah menyertakan file data sampel. Ini disebut `populate.sql`. Anda dapat memasukkan data sampel ke dalam basis data dengan menjalankannya melalui MySQL dengan cara yang sama.

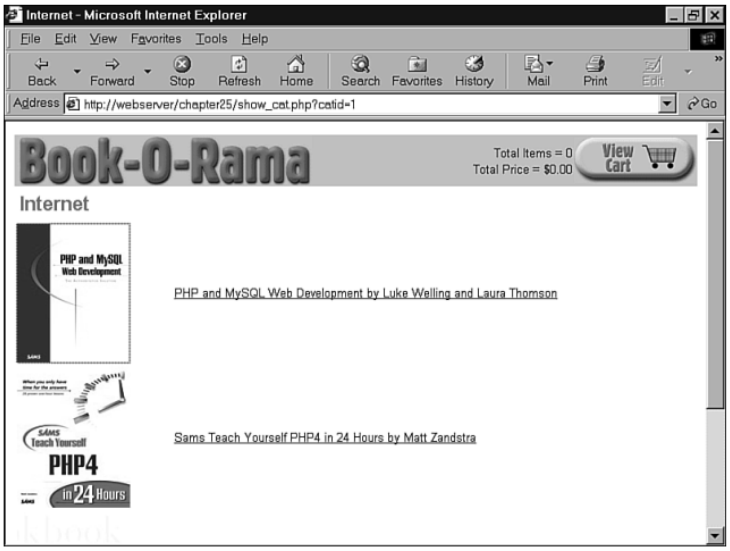
4.4 MENGIMPLEMENTASIKAN KATALOG ONLINE

Tiga skrip katalog ada dalam aplikasi ini: halaman utama, halaman kategori, dan halaman detail buku. Halaman depan situs ini dibuat oleh skrip yang disebut `index.php`. Output skrip ini ditunjukkan pada Gambar 4.3.

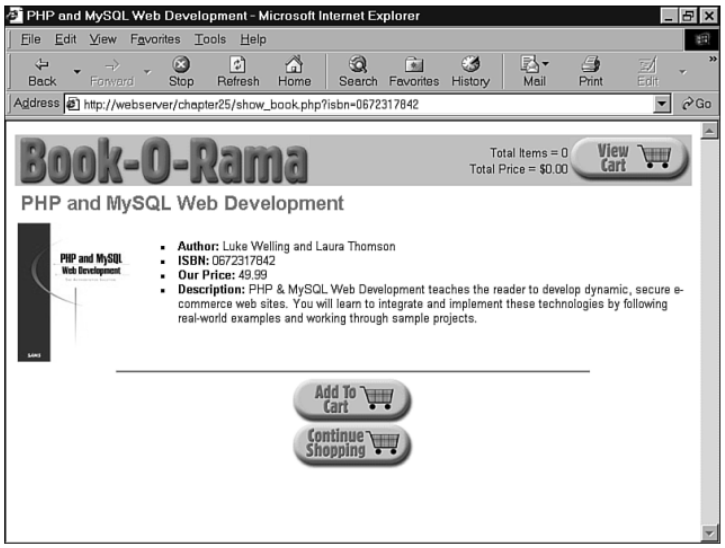


Gambar 4.3 Halaman Depan Situs Mencantumkan Kategori Buku Yang Tersedia Untuk Dibeli.

Anda akan melihat bahwa, selain daftar kategori di situs, terdapat tautan ke keranjang belanja di sudut kanan atas layar dan beberapa informasi ringkasan tentang apa yang ada di keranjang. Ini akan muncul di setiap halaman saat pengguna menjelajah dan berbelanja.



Gambar 4.4 Setiap Buku Dalam Kategori Tersebut Dicantumkan Dengan Foto.



Gambar 4.5 Setiap Buku Memiliki Halaman Detail Yang Menampilkan Informasi Lebih Lanjut Termasuk Deskripsi Yang Panjang.

Jika pengguna mengeklik salah satu kategori, ia akan dibawa ke halaman kategori, yang dibuat oleh skrip `show_cat.php`. Halaman kategori untuk bagian buku Internet ditunjukkan pada Gambar 4.4. Semua buku dalam kategori Internet tercantum sebagai tautan. Jika pengguna mengeklik salah satu tautan ini, ia akan dibawa ke halaman detail buku. Halaman detail buku untuk satu buku ditunjukkan pada Gambar 4.5. Pada halaman ini, selain tautan Lihat Keranjang, kami memiliki tautan Tambahkan ke Keranjang tempat pengguna dapat memilih item. Kami akan membahasnya lagi saat membahas cara membuat keranjang belanja nanti.

Mari kita lihat masing-masing dari ketiga skrip ini.

Mencantumkan Kategori

Skrip pertama, index.php, mencantumkan semua kategori dalam basis data. Skrip ini ditampilkan dalam Listing 4.2.

Listing 4.2 Index.Php; Skrip Untuk Membuat Halaman Depan Situs

```
<?
  include ('book_sc_fns.php');
  // The shopping cart needs sessions, so start one
  session_start();
  do_html_header("Welcome to Book-O-Rama");

  echo "<p>Please choose a category:</p>";

  // get categories out of database
  $cat_array = get_categories();

  // display as links to cat pages
  display_categories($cat_array);

  // if logged in as admin, show add, delete, edit cat links
  if(session_is_registered("admin_user"))
  {
      display_button("admin.php", "admin-menu", "Admin Menu");
  }
  do_html_footer();
?>
```

Skrip dimulai dengan menyertakan book_sc_fns.php, file yang menyertakan semua pustaka fungsi untuk aplikasi ini. Setelah itu, kita harus memulai sesi. Ini diperlukan agar fungsi keranjang belanja berfungsi. Setiap halaman di situs akan menggunakan sesi tersebut. Ada beberapa panggilan ke fungsi keluaran HTML seperti do_html_header() dan do_html_footer() (keduanya terdapat dalam output_fns.php). Kita juga memiliki beberapa kode yang memeriksa apakah pengguna masuk sebagai administrator dan memberinya beberapa opsi navigasi yang berbeda jika dia masuk—kita akan membahasnya lagi di bagian fungsi administrasi. Bagian terpenting dari skrip ini adalah

```
// get categories out of database
$cat_array = get_categories();

// display as links to cat pages
display_categories($cat_array);
```

Fungsi get_categories() dan display_categories() masing-masing ada di pustaka fungsi book_fns.php dan output_fns.php. Fungsi get_categories() mengembalikan array kategori dalam sistem, yang kemudian kita teruskan ke display_categories(). Mari kita lihat kode untuk get_categories(). Kode tersebut ditunjukkan pada Daftar 4.3.

Listing 4.3 Fungsi get_categories() dari output_fns.php—Fungsi yang Mengambil Daftar Kategori dari Basis Data

```
function get_categories()
{
  // query database for a list of categories
  $conn = db_connect();
  $query = "select catid, catname
            from categories";
  $result = @mysql_query($query);
  if (!$result)
    return false;
  $num_cats = @mysql_num_rows($result);
  if ($num_cats ==0)
     return false;
  $result = db_result_to_array($result);
  return $result;
}
```

Seperti yang dapat Anda lihat, fungsi ini terhubung ke basis data dan mengambil daftar semua ID dan nama kategori. Kami telah menulis dan menggunakan fungsi yang disebut db_result_to_array(), yang terdapat di db_fns.php. Fungsi ini ditampilkan dalam Listing 4.4. Fungsi ini mengambil pengidentifikasi hasil MySQL dan mengembalikan larik baris yang diindeks secara numerik, di mana setiap baris adalah larik asosiatif.

Listing 4.4 Fungsi db_result_to_array() dari db_fns.php; Fungsi yang Mengubah Pengidentifikasi Hasil MySQL menjadi Larik Hasil

```
function db_result_to_array($result)
{
  $res_array = array();

  for ($count=0; $row = @mysql_fetch_array($result); $count++)
    $res_array[$count] = $row;

  return $res_array;
}
```

Dalam kasus kami, kami akan mengembalikan array ini kembali ke index.php, tempat kami meneruskannya ke fungsi display_categories() dari output_fns.php. Fungsi ini menampilkan setiap kategori sebagai tautan ke halaman yang berisi buku-buku dalam kategori tersebut. Kode untuk fungsi ini ditunjukkan dalam listing 4.5.

Listing 4.5 Fungsi display_categories() dari output_fns.php—Fungsi yang Menampilkan Array Kategori sebagai Daftar Tautan ke Kategori Tersebut

```
function display_categories($cat_array)
{
  if (!is_array($cat_array))
  {
    echo "No categories currently available<br>";
    return;
  }
  echo "<ul>";
  foreach ($cat_array as $row)
  {
    $url = "show_cat.php?catid=".($row["catid"]);
    $title = $row["catname"];
    echo "<li>";
    do_html_url($url, $title);
  }
  echo "</ul>";
  echo "<hr>";
}
```

Fungsi ini mengonversi setiap kategori dari basis data menjadi tautan. Setiap tautan menuju ke skrip berikutnya show_cat.php tetapi masing-masing memiliki parameter yang berbeda, ID kategori atau catid. (Ini adalah nomor unik, yang dibuat oleh MySQL, dan digunakan untuk mengidentifikasi kategori.) Parameter ke skrip berikutnya ini akan menentukan kategori mana yang akan kita lihat.

4.5 MENCANTUMKAN BUKU DALAM KATEGORI

Proses untuk mencantumkan buku dalam kategori serupa. Skrip yang melakukan ini disebut show_cat.php. Skrip ini ditampilkan dalam listing 4.6.

Listing 4.6 show_cat.php—Skrip Ini Menampilkan Buku dalam Kategori Tertentu

```
<?
  include ('book_sc_fns.php');
  // The shopping cart needs sessions, so start one
  session_start();
  $name = get_category_name($catid);

  do_html_header($name);
```

```
  // get the book info out from db
  $book_array = get_books($catid);

  display_books($book_array);

  // if logged in as admin, show add, delete book links
  if(session_is_registered("admin_user"))
  {
    display_button("index.php", "continue", "Continue Shopping");
    display_button("admin.php", "admin-menu", "Admin Menu");
    display_button("edit_category_form.php?catid=$catid","edit-category",
                "Edit Category");
  }
  else
    display_button("index.php", "continue-shopping", "Continue Shopping");

  do_html_footer();
?>
```

Listing 4.7 Fungsi `get_category_name()` dari `book_fns.php`; Fungsi Ini Mengonversi ID Kategori ke Nama Kategori

```
function get_category_name($catid)
  {
    // query database for the name for a category id
    $conn = db_connect();
    $query = "select catname from categories where catid = $catid";

    $result = @mysql_query($query);
    if (!$result)
      return false;
    $num_cats = @mysql_num_rows($result);
    if ($num_cats ==0)
      return false;
    $result = mysql_result($result, 0, "catname");
  return $result;
}
```

Skrip ini sangat mirip strukturnya dengan halaman indeks, dengan perbedaan bahwa kita mengambil buku, bukan kategori. Kita mulai dengan `session_start()` seperti biasa, lalu mengonversi ID kategori yang telah diberikan ke nama kategori menggunakan fungsi `get_category_name()` sebagai berikut:

```
$name = get_category_name($catid);
```

Fungsi ini mencari nama kategori di database. Hal ini ditunjukkan pada Listing. 4.7.

Setelah kita mengambil nama kategori, kita dapat membuat header HTML dan melanjutkan untuk mengambil dan mencantumkan buku-buku dari basis data yang termasuk dalam kategori yang kita pilih, sebagai berikut:

```
$book_array = get_books($catid);
display_books($book_array);
```

Fungsi `get_books()` dan `display_books()` sangat mirip dengan fungsi `get_categories()` dan `display_categories()`, jadi kita tidak akan membahasnya di sini. Satu-satunya perbedaan adalah kita mengambil informasi dari tabel buku, bukan tabel kategori. Fungsi `display_books()` menyediakan tautan ke setiap buku dalam kategori melalui skrip `show_book.php`. Sekali lagi, setiap tautan diakhiri dengan parameter. Kali ini, itu adalah ISBN untuk buku yang dimaksud. Di bagian bawah skrip `show_cat.php`, Anda akan melihat bahwa ada beberapa kode untuk menampilkan beberapa fungsi tambahan jika seorang administrator masuk. Kita akan membahasnya di bagian fungsi administratif.

Menampilkan Detail Buku

Skrip `show_book.php` mengambil ISBN sebagai parameter dan mengambil serta menampilkan detail buku tersebut. Kode untuk skrip ini ditampilkan dalam listing 4.8.

Listing 4.8 show_book.php—Skrip Ini Menampilkan Detail Buku Tertentu

```
<?
  include ('book_sc_fns.php');
  // The shopping cart needs sessions, so start one
  session_start();

  // get this book out of database
  $book = get_book_details($isbn);
  do_html_header($book["title"]);
  display_book_details($book);

  // set url for "continue button"
  $target = "index.php";
  if($book["catid"])
  {
    $target = "show_cat.php?catid=".$book["catid"];
  }
  // if logged in as admin, show edit book links
  if( check_admin_user() )
  {
    display_button("edit_book_form.php?isbn=$isbn", "edit-item", "Edit Item");
    display_button("admin.php", "admin-menu", "Admin Menu");
    display_button($target, "continue", "Continue");
  }
```

```
  else
  {
    display_button("show_cart.php?new=$isbn", "add-to-cart",
                   "Add ".$book["title"]." To My Shopping Cart");
    display_button($target, "continue-shopping", "Continue Shopping");
  }

  do_html_footer();
?>
```

Sekali lagi dengan skrip ini, kita melakukan hal yang sangat mirip seperti pada dua halaman sebelumnya. Kita mulai dengan memulai sesi, lalu menggunakan

```
$book = get_book_details($isbn);
```

untuk mengeluarkan informasi buku dari basis data, dan

```
display_book_details($book);
```

untuk menampilkan data dalam HTML.

Satu hal yang perlu diperhatikan di sini adalah `display_book_details()` mencari berkas gambar untuk buku sebagai `images/$isbn.jpg`. Jika berkas ini tidak ada, tidak ada gambar yang akan ditampilkan. Sisa skrip mengatur navigasi. Pengguna biasa akan memiliki pilihan Lanjutkan Belanja, yang akan membawanya kembali ke halaman kategori, dan Tambahkan ke Keranjang, yang akan menambahkan buku ke keranjang belanjanya. Jika pengguna masuk sebagai administrator, ia akan mendapatkan beberapa opsi berbeda, yang akan kita bahas di bagian administrasi. Itu melengkapi dasar-dasar sistem katalog. Mari kita lanjutkan dan lihat kode untuk fungsionalitas keranjang belanja.

4.6 MENERAPKAN KERANJANG BELANJA

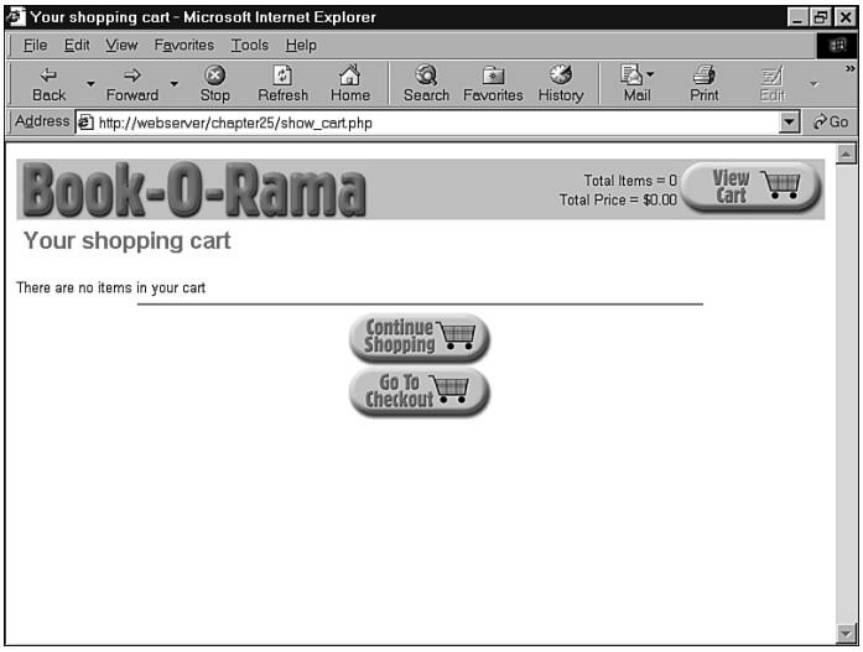
Fungsionalitas keranjang belanja semuanya berputar di sekitar variabel sesi yang disebut $cart. Ini adalah array asosiatif yang memiliki ISBN sebagai kunci dan jumlah sebagai nilai. Misalnya, jika saya menambahkan satu salinan buku ini ke keranjang belanja saya, array akan berisi

0672317842 => 1

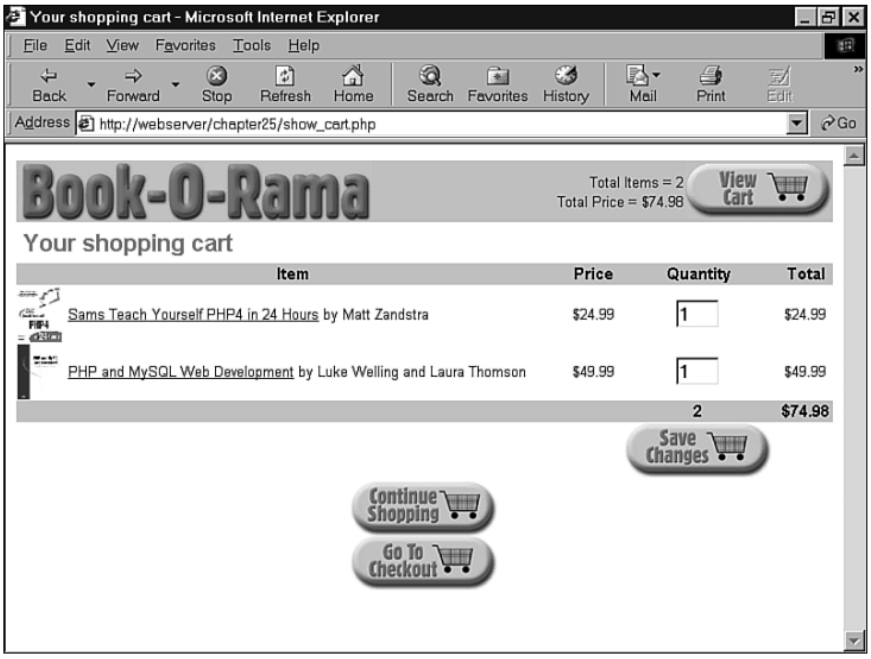
Yaitu, satu salinan buku dengan ISBN 0672317842. Saat kita menambahkan item ke keranjang, item tersebut akan ditambahkan ke array. Saat kita melihat keranjang, kita akan menggunakan array `$cart` untuk mencari detail lengkap item dalam basis data. Kita juga menggunakan dua variabel sesi lainnya untuk mengontrol tampilan di header yang menunjukkan Total Item dan Total Harga. Variabel ini masing-masing disebut `$items` dan `$total_price`.

Menggunakan Skrip `show_cart.php`

Mari kita mulai melihat bagaimana kode keranjang belanja diimplementasikan dengan melihat skrip `show_cart.php`. Ini adalah skrip yang menampilkan halaman yang akan kita kunjungi jika kita mengklik tautan Lihat Keranjang atau Tambahkan ke Keranjang. Jika kita memanggil `show_cart.php` tanpa parameter apa pun, kita akan melihat isinya. Jika kita memanggilnya dengan ISBN sebagai parameter, item dengan ISBN tersebut akan ditambahkan ke keranjang. Untuk memahami ini sepenuhnya, lihat Gambar 4.6 terlebih dahulu.



Gambar 4.6 Skrip Show_`Cart.`Php Tanpa Parameter Hanya Menunjukkan Isi Keranjang Kita.



Gambar 4.7 Skrip Show_Cart.Php Dengan Parameter Baru Menambahkan Item Baru Ke Keranjang.

Dalam kasus ini, kita telah mengeklik tautan Lihat Keranjang saat keranjang kita kosong; artinya, kita belum memilih barang apa pun untuk dibeli. Gambar 4.7 memperlihatkan keranjang kita sedikit lebih jauh di jalur saat kita telah memilih dua buku untuk dibeli. Dalam kasus ini, kita telah sampai ke halaman ini dengan mengeklik tautan Tambahkan ke Keranjang pada halaman show_book.php untuk buku ini, Pengembangan Web PHP dan MySQL. Jika Anda perhatikan dengan saksama bilah URL, Anda akan melihat bahwa kita telah memanggil skrip dengan parameter kali ini. Parameter tersebut disebut baru dan memiliki nilai 0672317842—artinya, ISBN untuk buku yang baru saja kita tambahkan ke keranjang.

Dari halaman ini, Anda dapat melihat bahwa kita memiliki dua opsi lainnya. Ada tombol Simpan Perubahan yang dapat kita gunakan untuk mengubah jumlah item di keranjang. Untuk melakukannya, Anda dapat mengubah jumlah secara langsung dan mengeklik Simpan Perubahan. Ini sebenarnya adalah tombol kirim yang membawa kita kembali ke skrip show_cart.php untuk memperbarui keranjang. Selain itu, ada tombol Buka Kasir yang dapat diklik pengguna saat ia siap untuk pergi. Kita akan membahasnya sebentar lagi. Untuk saat ini, mari kita lihat kode untuk skrip show_cart.php. Kode ini ditampilkan dalam listing 4.9.

Listing 4.9 show_cart.php; Skrip Ini Mengontrol Keranjang Belanja

```
<?
  include ('book_sc_fns.php');
  // The shopping cart needs sessions, so start one
  session_start();

  if($new)
  {
    //new item selected
    if(!session_is_registered("cart"))
    {
      $cart = array();
      session_register("cart");
      $items = 0;
      session_register("items");
      $total_price = "0.00";
      session_register("total_price");
    }
    if($cart[$new])
      $cart[$new]++;
    else
      $cart[$new] = 1;
    $total_price = calculate_price($cart);
    $items = calculate_items($cart);

  }
  if($save)
  {
```

```
    foreach ($cart as $isbn => $qty)
    {
      if($$isbn==”0”)
        unset($cart[$isbn]);
      else
        $cart[$isbn] = $$isbn;
    }
    $total_price = calculate_price($cart);
    $items = calculate_items($cart);
  }

  do_html_header(“Your shopping cart”);

  if($cart&&array_count_values($cart))
    display_cart($cart);
  else
  {
    echo “<p>There are no items in your cart”;
    echo “<hr>”;
  }
  $target = “index.php”;

  // if we have just added an item to the cart,
  // continue shopping in that category
  if($new)
  {
    $details =  get_book_details($new);
    if($details[“catid”])
      $target = “show_cat.php?catid=”.$details[“catid”];
  }
  display_button($target, “continue-shopping”, “Continue Shopping”);
  $path = $PHP_SELF;
  $path = str_replace(“show_cart.php”, “”, $path);
  display_button(“https://”.$SERVER_NAME.$path.”checkout.php”,
                 “go-to-checkout”, “Go To Checkout”);
  do_html_footer();
?>
```

Ada tiga bagian utama pada skrip ini: menampilkan keranjang, menambahkan item ke keranjang, dan menyimpan perubahan pada keranjang. Kami akan membahasnya di tiga bagian berikutnya.

Melihat Keranjang

Dari halaman mana pun kita berasal, kita akan menampilkan isi keranjang. Dalam kasus dasar, saat pengguna baru saja mengeklik Lihat Keranjang, ini adalah satu-satunya bagian kode yang akan dieksekusi, seperti berikut:

```
if($cart&&array_count_values($cart))
    display_cart($cart);
else
{
    echo “<p>There are no items in your cart”;
    echo “<hr>”;
}
```

Seperti yang dapat Anda lihat dari kode ini, jika kita memiliki keranjang belanja dengan beberapa konten, kita akan memanggil fungsi `display_cart()`. Jika keranjang belanja kosong, kita akan memberikan pesan kepada pengguna tentang hal itu.

Fungsi `display_cart()` hanya mencetak konten keranjang belanja sebagai format HTML yang dapat dibaca, seperti yang dapat Anda lihat pada Gambar 4.6 dan 4.7. Kode untuk fungsi ini dapat ditemukan di `output_fns.php`, yang disertakan di sini sebagai listing 4.10. Meskipun ini adalah fungsi tampilan, fungsinya cukup rumit, jadi kami menyertakannya di sini.

Listing 4.10 Fungsi `display_cart()` dari `output_fns.php` Fungsi Ini Memformat dan Mencetak Konten Keranjang Belanja

```
function display_cart($cart, $change = true, $images = 1)
{

  // display items in shopping cart
  // optionally allow changes (true or false)
  // optionally include images (1 - yes, 0 - no)

  global $items;
  global $total_price;

echo “<table border = 0 width = 100% cellspacing = 0>
       <form action = show_cart.php method = post>
       <tr><th colspan = “. (1+$images) .” bgcolor=\”#cccccc\”>Item</th>
       <th                                  bgcolor=\”#cccccc\”>Price</th><th
       bgcolor=\”#cccccc\”>Quantity</th>
       <th bgcolor=\”#cccccc\”>Total</th></tr>”;

// display each item as a table row
foreach ($cart as $isbn => $qty)
{
  $book = get_book_details($isbn);
  echo “<tr>”;
  if($images ==true)
  {
   echo “<td align = left>”;
   if (file_exists(“images/$isbn.jpg”))
   {
```

```
      $size = GetImageSize("images/".$isbn."jpg");
      if($size[0]>0 && $size[1]>0)
      {
       echo "<img src=\"images/".$isbn."jpg\" border=0 ";
       echo "width = ". $size[0]/3 ." height = " .$size[1]/3 . ">";
       }
}
else
      echo " ";
echo "</td>";
}
echo "<td align = left>";
echo "<a href = \"show_book.php?isbn=".$isbn."\">"
             .$book["title"]."</a> by ".$book["author"];

      echo "</td><td align = center>$".number_format($book["price"], 2);
      echo "</td><td align = center>";

      // if we allow changes, quantities are in text boxes
      if ($change == true)
        echo "<input type = text name = \"$isbn\" value = $qty size = 3>";
      else
        echo $qty;
      echo "</td><td align =center>$".number_format($book["price"]*$qty,2)
      ."</td></tr>\n";
   }

// display total row
echo "<tr>
       <th colspan = ". (2+$images) ." bgcolor=\"#cccccc\"> </td>
       <th align = center bgcolor=\"#cccccc\">
          $items
       </th>
       <th align = center bgcolor=\"#cccccc\">
          \$".number_format($total_price, 2).
        "</th>
       </tr>";

// display save change button
if($change == true)
{
 echo "<tr>
          <td colspan = ". (2+$images) ."> </td>
          <td align = center>
           <input type = hidden name = save value = true>
           <input type = image src = \"images/save-changes.gif\"
                 border = 0 alt = \"Save Changes\">
```

```
        </td>
        <td> </td>
    </tr>";
 }
 echo "</form></table>";
}
```

Alur dasar fungsi ini adalah sebagai berikut:

1. Ulangi setiap item dalam keranjang, dan berikan ISBN setiap item ke `get_book_details()` sehingga kita dapat meringkas detail setiap buku.
2. Berikan gambar untuk setiap buku, jika ada. Kita menggunakan tag tinggi dan lebar gambar HTML untuk mengubah ukuran gambar sedikit lebih kecil di sini. Ini berarti gambar akan sedikit terdistorsi, tetapi cukup kecil sehingga ini tidak menjadi masalah besar.
3. Jadikan setiap entri keranjang tautan ke buku yang sesuai, yaitu, ke `show_book.php` dengan ISBN sebagai parameter.
4. Jika kita memanggil fungsi dengan parameter `$change` yang disetel ke true (atau tidak disetel default-nya adalah true), tampilkan kotak dengan jumlah di dalamnya sebagai formulir dengan tombol Simpan Perubahan di bagian akhir. (Saat kita menggunakan kembali fungsi ini setelah menyelesaikan pembayaran, kita tidak ingin pengguna dapat mengubah pesanannya.)

Tidak ada yang terlalu rumit dalam fungsi ini, tetapi fungsinya cukup banyak, jadi sebaiknya Anda membacanya dengan saksama.

Menambahkan Barang ke Keranjang

Jika pengguna telah membuka halaman `show_cart.php` dengan mengklik tombol Tambahkan ke Keranjang, kita harus melakukan beberapa hal sebelum dapat menunjukkan isi keranjangnya. Secara khusus, kita perlu menambahkan barang yang sesuai ke keranjang, sebagai berikut. Pertama, jika pengguna belum memasukkan barang apa pun ke keranjangnya sebelumnya, ia tidak akan memiliki keranjang, jadi kita perlu membuatnya:

```
if(!session_is_registered("cart"))
  {
  $cart = array();
  session_register("cart");
  $items = 0;
  session_register("items");
  $total_price = "0.00";
  session_register("total_price");
  }
```

Pertama-tama, keranjang belanja kosong.

Kedua, setelah kita tahu bahwa keranjang belanja sudah disiapkan, kita dapat menambahkan barang ke dalamnya:

```
if($cart[$new])
  $cart[$new]++;
else
  $cart[$new] = 1
```

Di sini, kami memeriksa apakah barang tersebut sudah ada di keranjang. Jika sudah, kami menambah jumlah barang tersebut di keranjang sebanyak satu. Jika belum, kami menambahkan barang baru ke keranjang.

Ketiga, kita perlu menghitung total harga dan jumlah barang dalam keranjang. Untuk ini, kita menggunakan fungsi `calculate_price()` dan `calculate_items()`, sebagai berikut:

```
$total_price = calculate_price($cart);
$items = calculate_items($cart);
```

Fungsi-fungsi ini terletak di pustaka fungsi book_fns.php. Kode untuk fungsi-fungsi ini ditampilkan dalam Listing 4.11 dan 4.12, masing-masing.

Listing 4.11 Fungsi `calculate_price()` dari `book_fns.php`—Fungsi ini Menghitung dan Mengembalikan Total Harga Isi Keranjang Belanja

```
function calculate_price($cart)
{
  // sum total price for all items in shopping cart
  $price = 0.0;
  if(is_array($cart))
  {
    $conn = db_connect();
    foreach($cart as $isbn => $qty)
    {
      $query = "select price from books where isbn='$isbn'";
      $result = mysql_query($query);
      if ($result)
      {
        $item_price = mysql_result($result, 0, "price");
        $price +=$item_price*$qty;
      }
    }
  }
  return $price;
}
```

Seperti yang dapat Anda lihat, fungsi `calculate_price()` bekerja dengan mencari harga setiap item dalam keranjang belanja di basis data. Ini agak lambat, jadi untuk menghindari melakukan ini lebih sering dari yang seharusnya, kami akan menyimpan harga

(dan jumlah total item juga) sebagai variabel sesi dan hanya menghitung ulang saat keranjang belanja berubah.

Listing 4.12 Fungsi `Calculate_Items()` Dari `Book_Fns.Php`; Fungsi Ini Menghitung Dan Mengembalikan Jumlah Total Item Dalam Keranjang Belanja

```
function calculate_items($cart)
{
  // sum total items in shopping cart
  $items = 0;
  if(is_array($cart))
  {
    foreach($cart as $isbn => $qty)
    {
      $items += $qty;
    }
  }
  return $items;
}
```

Fungsi `calculate_items()` lebih sederhana—fungsi ini hanya menelusuri keranjang dan menjumlahkan jumlah setiap item untuk mendapatkan jumlah total item.

Menyimpan Keranjang yang Diperbarui

Jika kita membuka skrip `show_cart.php` setelah mengklik tombol Save Changes, prosesnya sedikit berbeda. Dalam kasus ini, kita membukanya melalui pengiriman formulir. Jika Anda mencermati kode tersebut, Anda akan melihat bahwa tombol Save Changes adalah tombol kirim untuk formulir. Formulir ini berisi variabel tersembunyi save. Jika variabel ini ditetapkan, kita tahu bahwa kita membuka skrip ini dari tombol Save Changes. Ini berarti bahwa pengguna mungkin telah mengedit nilai jumlah di keranjang, dan kita perlu memperbaruinya.

Jika Anda melihat kembali kotak teks di bagian formulir Save Changes dari skrip tersebut, Anda akan melihat bahwa kotak tersebut diberi nama sesuai dengan ISBN item yang diwakilinya, seperti berikut:

```
echo "<input type = text name = \"$isbn\" value = $qty size = 3>";
```

Sekarang lihat bagian skrip yang menyimpan perubahan:

```
if($save)
  {
   foreach ($cart as $isbn => $qty)
   {
    if($$isbn=="0")
      unset($cart[$isbn]);
    else
```

```
        $cart[$isbn] = $$isbn;
    }
    $total_price = calculate_price($cart);
    $items = calculate_items($cart);
}
```

Anda dapat melihat bahwa kami sedang memeriksa keranjang belanja, dan untuk setiap `$isbn` di keranjang, kami memeriksa variabel dengan nama itu, menggunakan notasi variabel `$$isbn`. Sebagai pengingat, saat kami merujuk ke `$$isbn`, kami sebenarnya merujuk ke variabel yang namanya disimpan di `$isbn`. Ini adalah kolom formulir dari formulir Simpan Perubahan. Jika salah satu kolom ditetapkan ke nol, kami menghapus item tersebut dari keranjang belanja secara keseluruhan, menggunakan `unset()`. Jika tidak, kami memperbarui keranjang agar sesuai dengan kolom formulir, sebagai berikut:

```
if($$isbn==”0”)
unset($cart[$isbn]);
else
$cart[$isbn] = $$isbn
```

Setelah pembaruan ini, kami kembali menggunakan `calculate_price()` dan `calculate_items()` untuk menghitung nilai baru dari variabel sesi `$total_price` dan `$items`.

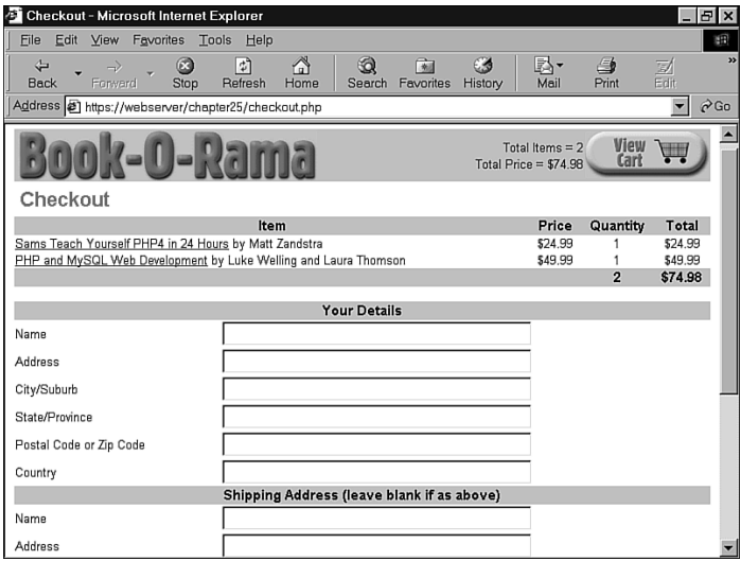
Mencetak Ringkasan Bilah Header

Anda akan melihat bahwa di bilah header setiap halaman di situs, ringkasan tentang apa yang ada di keranjang belanja disajikan. Ini diperoleh dengan mencetak nilai variabel sesi `$total_price` dan `$items`. Ini dilakukan dalam fungsi `do_html_header()`. Variabel-variabel ini didaftarkan saat pengguna pertama kali mengunjungi halaman `show_cart.php`. Kami juga memerlukan beberapa logika untuk menangani kasus-kasus ketika pengguna belum mengunjungi halaman tersebut. Logika ini juga ada dalam fungsi `do_html_header()`:

```
if(!$items) $items = “0”;
if(!$total_price) $total_price = “0.00”;
```

Melakukan Pembayaran

Saat pengguna mengklik tombol Go to Checkout dari keranjang belanja, skrip checkout.php akan diaktifkan.



Gambar 4.8 Skrip checkout.php mendapatkan detail pelanggan.

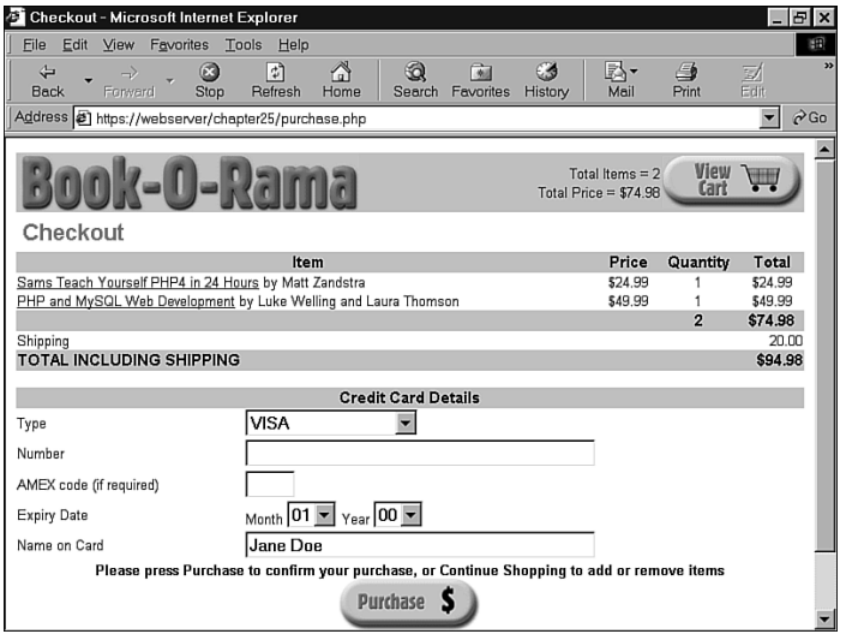
Halaman checkout dan halaman-halaman di belakangnya harus diakses melalui SSL, tetapi contoh aplikasi tidak memaksa Anda untuk melakukan ini. Halaman checkout ditunjukkan pada Gambar 4.8. Skrip ini mengharuskan pelanggan untuk memasukkan alamatnya (dan alamat pengiriman jika berbeda). Ini adalah skrip yang cukup sederhana, yang dapat Anda lihat dengan melihat kode pada Listing 4.13.

Listing 4.13 Checkout.Php; Skrip Ini Mendapatkan Detail Pelanggan

```
<?
  //include our function set
  include ('book_sc_fns.php');

  // The shopping cart needs sessions, so start one session_start();

  do_html_header("Checkout");

  if($cart&&array_count_values($cart))
  {
    display_cart($cart, false, 0);
    display_checkout_form($HTTP_POST_VARS);
  }
  else
    echo "<p>There are no items in your cart";
    display_button("show_cart.php", "continue-shopping", "Continue Shopping");

  do_html_footer();
?>
```

Tidak ada kejutan besar dalam skrip ini. Jika keranjang kosong, skrip akan memberi tahu pelanggan; jika tidak, skrip akan menampilkan formulir yang ditunjukkan pada Gambar 4.8.

Jika pengguna melanjutkan dengan mengklik tombol Beli di bagian bawah formulir, ia akan dibawa ke skrip purchase.php. Anda dapat melihat output skrip ini pada Gambar 4.9.



Gambar 4.9 Skrip Purchase.Php Menghitung Pengiriman Dan Total Pesanan Akhir, Dan Mendapatkan Detail Pembayaran Pelanggan.

Kode untuk skrip ini sedikit lebih rumit daripada kode untuk checkout.php. Kode ini ditunjukkan pada listing 4.14.

Listing 4.14 Purchase.Php Skrip Ini Menyimpan Detail Pesanan Dalam Basis Data Dan Mendapatkan Detail Pembayaran

```
<?
  include ('book_sc_fns.php');
  // The shopping cart needs sessions, so start one

  session_start();

  do_html_header("Checkout");

  // if filled out
  if($from=='process'||$cart&&$name&&$address&&$city&&$zip&&$country)
  {
    // able to insert into database
    if($from!='process')
    {
      if(!insert_order($HTTP_POST_VARS))
      {
        echo "Could not store data, please try again.";
        display_button("checkout.php", "back", "Back");
```

```
        do_html_footer();
        exit;
      }
    }

    //display cart, not allowing changes and without pictures
    display_cart($cart, false, 0);
    display_shipping(calculate_shipping_cost());

    //get credit card details
    display_card_form($HTTP_POST_VARS);

    display_button("show_cart.php", "continue-shopping", "Continue Shopping");
  }
  else
  {
    echo "You did not fill in all the fields, please try again.<hr>";
    echo "<form action = 'checkout.php' method = post>";

    // pass the data this page received back so the user won't have to re-enter
    foreach($HTTP_POST_VARS as $name => $value)
    echo "<input type = hidden name = $name value = '$value'>\n";

    display_form_button("back", "Back");
    echo "</form>";
  }
do_html_footer();
?>
```

Logikanya di sini sederhana: Kami memeriksa apakah pengguna telah mengisi formulir dan memasukkan detail ke dalam basis data menggunakan fungsi yang disebut `insert_order()`. Ini adalah fungsi sederhana yang memasukkan detail pelanggan ke dalam basis data. Kode untuk fungsi ini ditunjukkan pada Daftar 25.15.

Listing 4.15 Fungsi `insert_order()` dari `order_fns.php`—Fungsi ini memasukkan semua detail pesanan pelanggan ke dalam basis data

```
function insert_order($order_details)
{
   global $total_price;
   global $cart;
   //extract order_details out as variables
   extract($order_details);

   //set shipping address same as address

   if(!$ship_name&&!$ship_address&&!$ship_city&&
```

```
!$ship_state&&!$ship_  zip&&!$ship_
country)
 {
  $ship_name = $name;
  $ship_address = $address;
  $ship_city = $city;
  $ship_state = $state;
  $ship_zip = $zip;
  $ship_country = $country;
 }

$conn = db_connect();

//insert customer address
 $query = "select customerid from customers where
           name = '$name' and address = '$address'
           and city = '$city' and state = '$state'
           and zip = '$zip' and country = '$country'";
 $result = mysql_query($query);
 if(mysql_numrows($result)>0)
 {
    $customer_id = mysql_result($result, 0, "customerid");
 }
 else
 {
   $query = "insert into customers values
        ('', '$name','$address','$city','$state','$zip','$country')";
   $result = mysql_query($query);

   if (!$result)
     return false;
 }
 $query = "select customerid from customers where
           name = '$name' and address = '$address'
           and city = '$city' and state = '$state'
           and zip = '$zip' and country = '$country'";
 $result = mysql_query($query);
 if(mysql_numrows($result)>0)
    $customerid = mysql_result($result, 0, "customerid");
 else
    return false;
 $date = date("Y-m-d");
 $query = "insert into orders values
    ('', $customerid, $total_price, '$date', 'PARTIAL', '$ship_name',
    '$ship_address','$ship_city','$ship_state','$ship_zip',
    '$ship_country')";
 $result = mysql_query($query);
```

```
 if (!$result)
      return false;

 $query = "select orderid from orders where
            customerid = $customerid and
            amount > $total_price-.001 and
            amount < $total_price+.001 and
            date = '$date' and
            order_status = 'PARTIAL' and
            ship_name = '$ship_name' and
            ship_address = '$ship_address' and
            ship_city = '$ship_city' and
            ship_state = '$ship_state' and
            ship_zip = '$ship_zip' and
            ship_country = '$ship_country'";
 $result = mysql_query($query);
 if(mysql_numrows($result)>0)
    $orderid = mysql_result($result, 0, "orderid");
 else
    return false;

 // insert each book
 foreach($cart as $isbn => $quantity)
 {
     $detail = get_book_details($isbn);
     $query = "delete from order_items where
           orderid = $orderid and isbn =  '$isbn'";
     $result = mysql_query($query);
     $query = "insert into order_items values
           ('$orderid', '$isbn', ".$detail["price"].", $quantity)";
     $result = mysql_query($query);
     if(!$result)
        return false;
     }
     return $orderid;
}
?>
```

Fungsi ini agak panjang karena kita perlu memasukkan rincian pelanggan, rincian pesanan, dan rincian setiap buku yang ingin mereka beli. Kemudian kita hitung biaya pengiriman ke alamat pelanggan dan beri tahu mereka berapa biayanya dengan baris kode berikut:

```
display_shipping(calculate_shipping_cost());
```

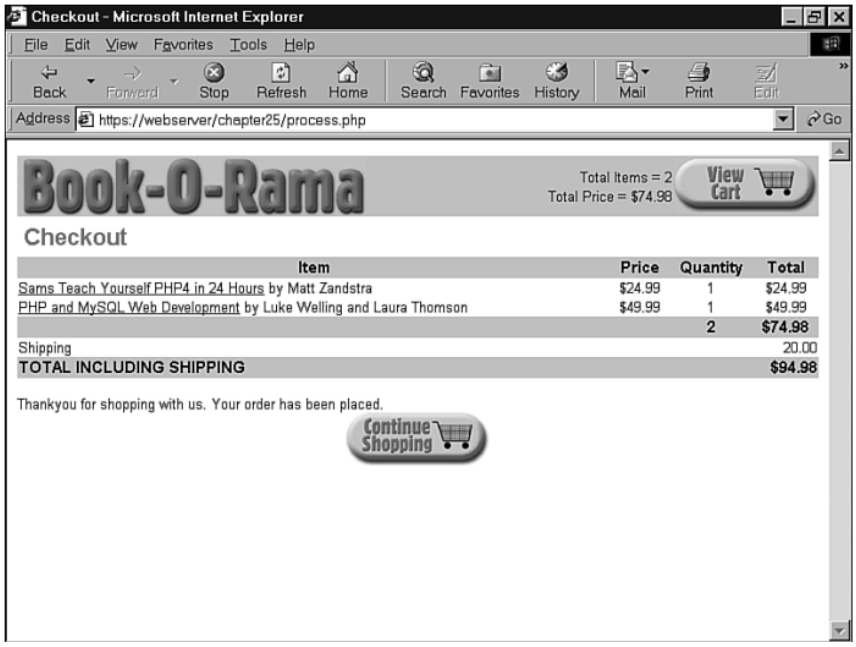
Kode yang kita gunakan di sini untuk `calculate_shipping_cost()` selalu mengembalikan Rp. 325.600. Saat Anda benar-benar menyiapkan situs belanja, Anda harus memilih metode

pengiriman, mencari tahu berapa biayanya untuk berbagai tujuan, dan menghitung biaya yang sesuai. Kemudian kita tampilkan formulir bagi pengguna untuk mengisi rincian kartu kreditnya menggunakan fungsi

display_card_form() dari pustaka output_fns.php.

Mengimplementasikan Pembayaran

Saat pengguna mengklik tombol Beli, kita akan memproses rincian pembayarannya menggunakan skrip process.php. Anda dapat melihat hasil pembayaran yang berhasil pada Gambar 4.10.



Gambar 4.10 Transaksi Ini Berhasil, Dan Barang Akan Segera Dikirim.

Kode untuk process.php dapat ditemukan di Listing 4.16.

Listing 4.16 Process.Php; Skrip Process.Php Memproses Pembayaran Pelanggan Dan Memberitahu Hasilnya

```
<?
  include ('book_sc_fns.php');
  // The shopping cart needs sessions, so start one
  session_start();
  do_html_header("Checkout");

  if($cart&&$card_type&&$card_number&&$card_month&&$card_year&&$card_name)
 {
  //display cart, not allowing changes and without pictures
  display_cart($cart, false, 0);
  display_shipping(calculate_shipping_cost());
```

```
  if(process_card($HTTP_POST_VARS))
  {

//empty shopping cart
 session_destroy();
 echo "Thankyou for shopping with us.  Your order has been placed.";
 display_button("index.php", "continue-shopping", "Continue Shopping");
}
else
{
     echo "Could not process your card, please contact the card issuer or
     try ➥again.";

     display_button("purchase.php", "back", "Back");
    }
 }
 else
 {
  echo "You did not fill in all the fields, please try again.<hr>";
  echo "<form action = 'purchase.php' method = post>";
  echo "<input type = hidden name = from value = process>\n";

// pass the data this page received back so the user won't have to re-enter
foreach($HTTP_POST_VARS as $name => $value)
  echo "<input type = hidden name = $name value = '$value'>\n";

  display_form_button("back", "Back");
  echo "</form>";
 }

 do_html_footer();
?>
The crux of this script is these lines:
if(process_card($HTTP_POST_VARS))
{
  //empty shopping cart
  session_destroy();
  echo "Thankyou for shopping with us.  Your order has been placed.";
  display_button("index.php", "continue-shopping", "Continue Shopping");
}
```

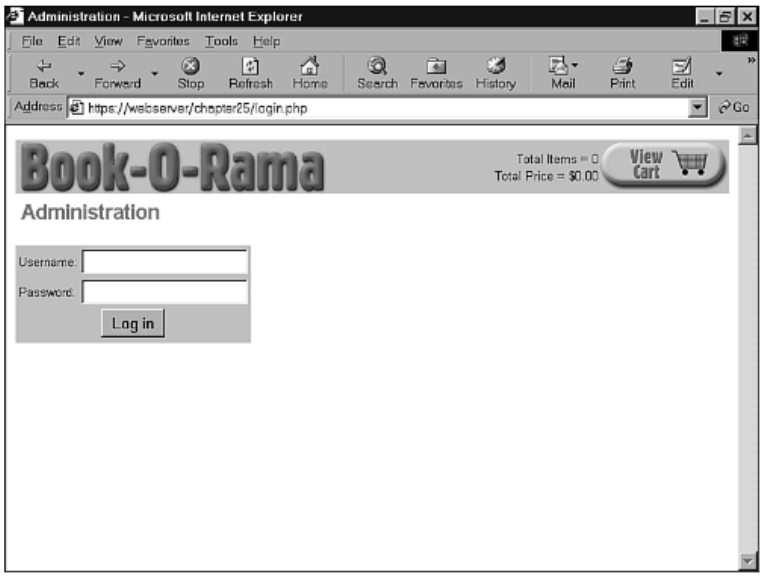
Seperti di tempat lain tempat kami merujuk langsung ke $HTTP_POST_VARS, Anda perlu mengaktifkan `track_vars` agar ini berfungsi. Kami memproses kartu pengguna, dan, jika semuanya berhasil, menghapus sesinya. Fungsi pemrosesan kartu seperti yang telah kami tulis hanya mengembalikan true. Saat Anda menyiapkan situs langsung, Anda perlu membuat keputusan tentang mekanisme kliring transaksi yang ingin Anda gunakan. Anda dapat

- Mendaftar dengan penyedia kliring transaksi. Ada banyak sekali alternatif di sini, tergantung pada area tempat tinggal Anda. Beberapa di antaranya akan menawarkan kliring waktu nyata, dan yang lainnya tidak. Apakah Anda memerlukan kliring langsung tergantung pada layanan yang Anda tawarkan. Jika Anda menyediakan layanan daring, kemungkinan besar Anda menginginkannya; jika Anda mengirimkan barang, itu tidak terlalu penting. Apa pun itu, penyedia ini membebaskan Anda dari tanggung jawab menyimpan nomor kartu kredit.
- Kirim nomor kartu kredit ke diri Anda sendiri melalui email terenkripsi, misalnya, dengan menggunakan PGP atau GPG seperti yang dibahas dalam Bab 15. Saat Anda menerima dan mendekripsi email, Anda dapat memproses transaksi ini secara manual.
- Simpan nomor kartu kredit di basis data Anda. Kami tidak menyarankan opsi ini kecuali Anda benar-benar tahu apa yang Anda lakukan dengan keamanan sistem. Anda dapat membaca Bab 15 untuk detail lebih lanjut tentang mengapa
- ini merupakan ide yang buruk.

Itu saja untuk modul keranjang belanja dan pembayaran.

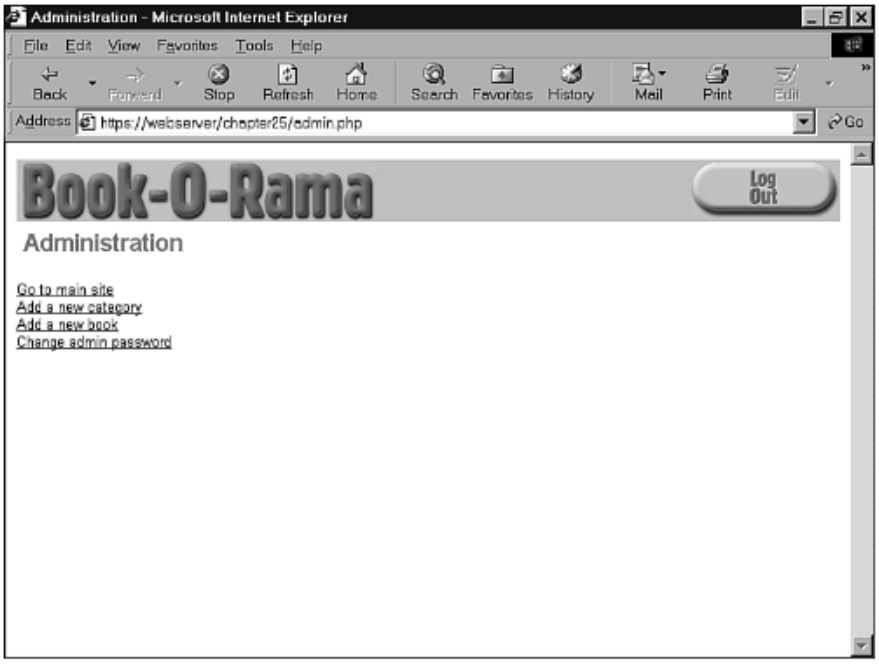
4.7 MENERAPKAN ANTARMUKA ADMINISTRASI

Antarmuka administrasi yang telah kami terapkan sangat sederhana. Yang telah kami lakukan hanyalah membangun antarmuka Web ke basis data dengan beberapa autentikasi front-end. Ini sebagian besar merupakan kode yang sama seperti yang digunakan dalam Bab 3. Kami telah menyertakannya di sini untuk kelengkapan, tetapi dengan sedikit diskusi.



Gambar 4.11 Pengguna Harus Melewati Halaman Login Untuk Mengakses Fungsi Admin.

Antarmuka administrasi mengharuskan pengguna untuk masuk melalui file `login.php`, yang kemudian membawanya Ke menu administrasi, `admin.php`. Halaman login ditunjukkan pada Gambar 4.11. (Kami tidak menyertakan file `login.php` di sini demi singkatnya file ini hampir sama persis dengan file yang ada di Bab 3. Menu administrasi ditunjukkan pada Gambar 4.12.



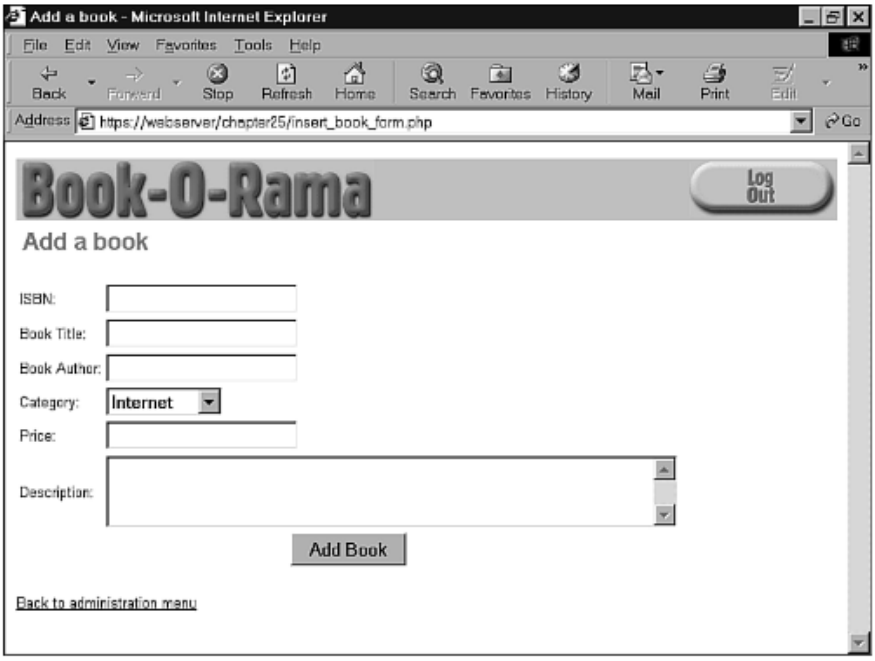
Gambar 4.12 Menu Administrasi Memungkinkan Akses Ke Fungsi Admin.

Listing 4.17 admin.php Skrip ini mengotentikasi Administrator dan mengizinkannya mengakses fungsi admin

```
<?

// include function files for this application
require_once("book_sc_fns.php");
session_start();

if ($username && $passwd)
// they have just tried logging in
{
  if (login($username, $passwd))
    {
    // if they are in the database register the user id
    $admin_user = $username;
    session_register("admin_user");
  }
  else
  {
    // unsuccessful login
    do_html_header("Problem:");
    echo "You could not be logged in.
         You must be logged in to view this page.<br>";
    do_html_url("login.php", "Login");
    do_html_footer();
    exit;
  }
```

```
}

do_html_header("Administration");
if (check_admin_user())
  display_admin_menu();
else
  echo "You are not authorized to enter the administration area.";
do_html_footer();
?>
```

Kode ini mungkin terlihat familier; mirip dengan skrip dari Bab 3. Setelah administrator mencapai titik ini, ia dapat mengubah kata sandinya atau keluar kode ini identik dengan kode di Bab 3, jadi kami tidak akan membahasnya di sini.



Gambar 4.13 Formulir Ini Memungkinkan Administrator Memasukkan Buku Baru Ke Dalam Katalog Daring.

Kami mengidentifikasi pengguna administrasi setelah login melalui variabel sesi `$admin_user` dan fungsi `check_admin_user()`. Fungsi ini dan fungsi lain yang digunakan oleh skrip administratif dapat ditemukan di pustaka fungsi `admin_fns.php`.

Jika administrator memilih untuk menambahkan kategori atau buku baru, ia akan membuka `insert_category_form.php` atau `insert_book_form.php`, sebagaimana mestinya. Setiap skrip ini menyajikan formulir untuk diisi oleh administrator. Setiap skrip diproses oleh skrip terkait (`insert_category.php` dan `insert_book.php`), yang memverifikasi bahwa formulir telah diisi dan memasukkan data baru ke dalam basis data. Kami akan melihat versi buku dari skrip tersebut saja, karena keduanya sangat mirip satu sama lain. Output dari `insert_book_form.php` ditunjukkan pada Gambar 4.13. Anda akan melihat

bahwa kolom Kategori untuk buku adalah elemen HTML SELECT. Opsi untuk SELECT ini berasal dari panggilan ke fungsi `get_categories()` yang telah kita bahas sebelumnya. Saat tombol Tambah Buku diklik, skrip `insert_book.php` akan diaktifkan. Kode untuk skrip ini ditampilkan dalam listing 4.18.

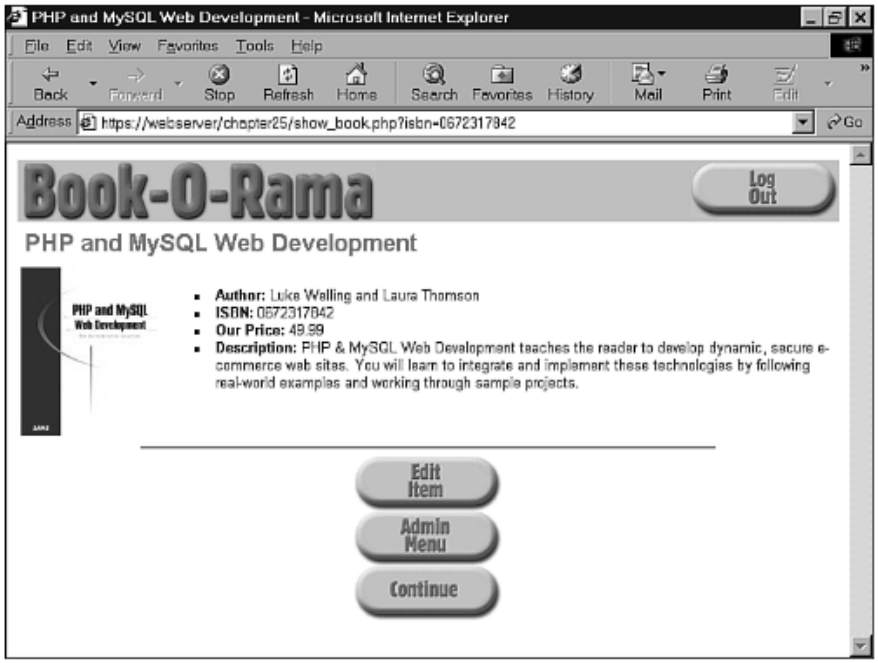
Listing 4.18 `insert_book.php`; Skrip ini Memvalidasi Data Buku Baru dan Memasukkannya ke dalam Database

```
<?

// include function files for this application
require_once("book_sc_fns.php");
session_start();

do_html_header("Adding a book");
if (check_admin_user())
{
  if (filled_out($HTTP_POST_VARS))
  {
    if(insert_book($isbn, $title, $author, $catid, $price, $description))
      echo "Book '$title' was added to the database.<br>";
    else
      echo "Book '$title' could not be added to the database.<br>";
  }
  else
    echo "You have not filled out the form.  Please try again.";
  do_html_url("admin.php", "Back to administration menu");
}
else
  echo "You are not authorised to view this page.";

do_html_footer();

?>
```

Anda dapat melihat bahwa skrip ini memanggil fungsi `insert_book()`. Fungsi ini dan fungsi lain yang digunakan oleh skrip administratif dapat ditemukan di pustaka fungsi `admin_fns.php`.

Selain menambahkan kategori dan buku baru, pengguna administratif dapat mengedit dan menghapus item ini. Kami telah menerapkan ini dengan menggunakan kembali kode sebanyak mungkin. Ketika administrator mengklik tautan Buka situs utama di menu administrasi, ia akan masuk ke indeks kategori di `index.php` dan dapat menavigasi situs dengan cara yang sama seperti pengguna biasa, menggunakan skrip yang sama. Namun, ada perbedaan dalam navigasi administratif: Administrator akan melihat opsi yang berbeda berdasarkan fakta bahwa mereka memiliki variabel sesi terdaftar `$admin_user`. Misalnya, jika

kita melihat halaman show_book.php yang telah kita lihat sebelumnya di bab ini, kita akan melihat beberapa opsi menu yang berbeda. Lihat Gambar 4.14.

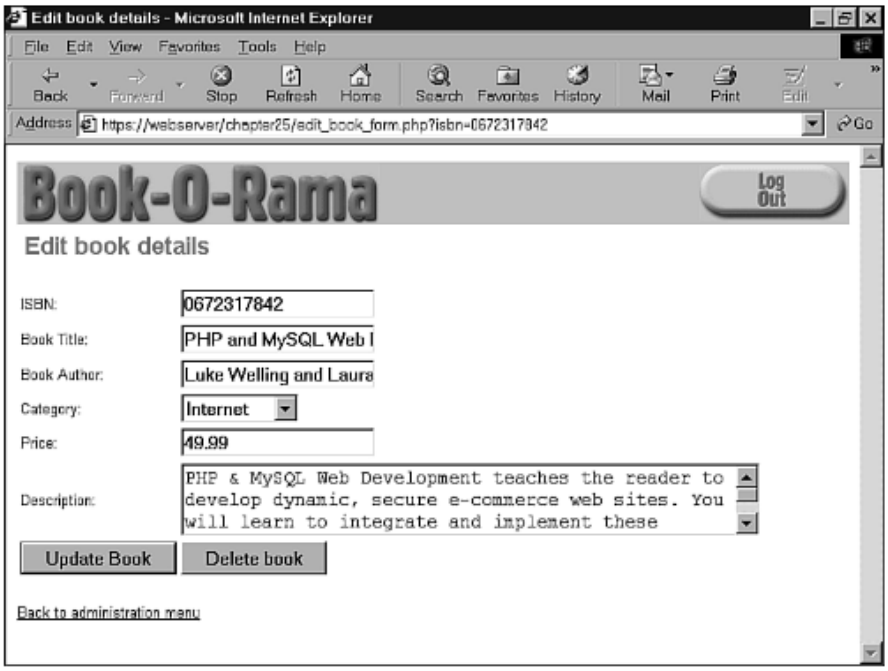


Gambar 25.14 Skrip Show_Book.Php Menghasilkan Keluaran Yang Berbeda Untuk Pengguna Administratif.

Administrator memiliki akses ke dua opsi baru di halaman ini: Edit Item dan Menu Admin. Anda juga akan melihat bahwa kita tidak melihat keranjang belanja di sudut kanan atas sebagai gantinya, kita memiliki tombol Log Out. Kode untuk ini semuanya ada di sana, kembali ke listing 4.8, sebagai berikut:

```
if( check_admin_user() )
{
  display_button("edit_book_form.php?isbn=$isbn", "edit-item", "Edit Item");
  display_button("admin.php", "admin-menu", "Admin Menu");
  display_button($target, "continue", "Continue");
}
```

Jika Anda melihat kembali skrip show_cat.php, Anda akan melihat bahwa skrip tersebut juga memiliki opsi-opsi ini. Jika administrator mengklik tombol Edit Item, ia akan masuk ke skrip edit_book_form.php. Output dari skrip ini ditunjukkan pada Gambar 4.15.



Gambar 4.15 Skrip Edit_Book_Form.Php Memberi Akses Kepada Administrator Untuk Mengedit Detail Buku Atau Menghapus Buku.

Sebenarnya, ini adalah formulir yang sama yang kita gunakan untuk mendapatkan detail buku pada awalnya. Kita membuat opsi ke dalam formulir tersebut untuk memasukkan dan menampilkan data buku yang ada. Kita melakukan hal yang sama dengan formulir kategori. Untuk memahami maksud kita, lihat listing 4.19.

Listing 4.19 Fungsi `display_book_form()` dari `admin_fns.php`; Formulir Ini Berfungsi Ganda sebagai Formulir Penyisipan dan Penyuntingan

```
function display_book_form($book = “”)
 // This displays the book form.
 // It is very similar to the category form.
 // This form can be used for inserting or editing books.
 // To insert, don’t pass any parameters.  This will set $edit
 // to false, and the form will go to insert_book.php.
 // To update, pass an array containing a book.  The
 // form will be displayed with the old data and point to update_book.php.
 // It will also add a “Delete book” button.
 {

   // if passed an existing book, proceed in “edit mode”
   $edit = is_array($book);

     // most of the form is in plain HTML with some
     // optional PHP bits throughout
  ?>
       <form method=post
```

```
            action="<?=$edit?"edit_book.php":"insert_book.php";?>">
        <table border=0>
        <tr>
          <td>ISBN:</td>
          <td><input type=text name=isbn
           value="<?=$edit?$book["isbn"]:""; ?>"></td>
  </tr>
  <tr>
       <td>Book Title:</td>
       <td><input type=text name=title
          value="<?=$edit?$book["title"]:""; ?>"></td>
   </tr>
   <tr>
     <td>Book Author:</td>
     <td><input type=text name=author
          value="<?=$edit?$book["author"]:""; ?>"></td>
   </tr>
   <tr>
     <td>Category:</td>
     <td><select name=catid>
     <?
       // list of possible categories comes from database
       $cat_array=get_categories();
       foreach ($cat_array as $thiscat)
       {
            echo "<option value=\"";
            echo $thiscat["catid"];
            echo "\"";
            // if existing book, put in current catgory
            if ($edit && $thiscat["catid"] == $book["catid"])
                echo " selected";
            echo ">";
            echo $thiscat["catname"];
            echo "\n";
       }
       ?>
       </select>
     </td>
   </tr>
   <tr>

   <td>Price:</td>
   <td><input type=text name=price
              value="<?=$edit?$book["price"]:""; ?>"></td>
</tr>
<tr>
   <td>Description:</td>
```

```
   <td><textarea rows=3 cols=50
           name=description>
           <?=$edit?$book[“description”]:””; ?>
           </textarea></td>
 </tr>
 <tr>
      <td <? if (!$edit) echo ”colspan=2”; ?> align=center>
       <?
         if ($edit)
         // we need the old isbn to find book in database
         // if the isbn is being updated
         echo “<input type=hidden name=oldisbn
         value=\””.$book[“isbn”].”\”>”;
   ?>
 <input type=submit
      value=”<?=$edit?”Update”:”Add”; ?> Book”>
 </form></td>
 <?
    if ($edit)
    {
    echo “<td>”;
    echo “<form method=post action=\”delete_book.php\”>”;
    echo “<input type=hidden name=isbn
          value=\””.$book[“isbn”].”\”>”;
    echo “<input type=submit
          value=\”Delete book\”>”;
    echo “</form></td>”;
    }
   ?>
  </td>
 </tr>
</table>
</form>
<?
}
```

Jika kita memasukkan array yang berisi data buku, formulir akan ditampilkan dalam mode edit dan akan mengisi kolom dengan data yang ada:

```
<input type=text name=price
        value=”<?=$edit?$book[“price”]:””; ?>”>
```

Kita bahkan mendapatkan tombol kirim yang berbeda. Bahkan, untuk formulir edit, kita mendapatkan dua - satu untuk memperbarui buku, dan satu untuk menghapusnya. Skrip ini memanggil skrip `edit_book.php` dan `delete_book.php`, yang memperbarui basis data sesuai dengan itu.

Versi kategori dari skrip ini bekerja dengan cara yang hampir sama kecuali untuk satu hal. Ketika seorang administrator mencoba menghapus kategori, kategori itu tidak akan dihapus jika masih ada buku di dalamnya. (Ini diperiksa dengan kueri basis data.) Ini menghindari masalah apa pun yang mungkin kita dapatkan dengan anomali penghapusan. Dalam kasus ini, jika sebuah kategori dihapus yang masih memiliki buku di dalamnya, buku-buku ini akan menjadi yatim piatu. Kita tidak akan tahu di kategori mana buku-buku itu berada, dan kita tidak akan memiliki cara untuk menavigasi ke sana! Itulah ikhtisar antarmuka administrasi.

Memperluas Proyek

Kita telah membangun sistem keranjang belanja yang cukup sederhana. Ada banyak tambahan dan penyempurnaan yang dapat kami lakukan:

- Di toko daring sungguhan, Anda perlu membangun semacam sistem pelacakan dan pemenuhan pesanan saat ini, tidak ada cara untuk melihat pesanan yang telah dilakukan.
- Pelanggan ingin dapat memeriksa kemajuan pesanan mereka tanpa harus menghubungi Anda. Kami merasa penting bahwa pelanggan tidak harus masuk untuk menelusuri. Namun, menyediakan cara bagi pelanggan yang sudah ada untuk mengautentikasi diri mereka sendiri memberi mereka kemampuan untuk melihat pesanan sebelumnya, dan memberi Anda kemampuan untuk menggabungkan perilaku menjadi profil.
- Saat ini, gambar untuk buku harus di-FTP ke direktori gambar dan diberi nama yang benar. Anda dapat menambahkan unggahan file ke halaman penyisipan buku untuk mempermudah hal ini.
- Anda dapat menambahkan login pengguna, personalisasi, dan rekomendasi buku; ulasan daring; program afiliasi; pengecekan tingkat stok; dan sebagainya. Kemungkinannya tidak terbatas. Menggunakan Sistem yang Ada

Jika Anda ingin membuat keranjang belanja yang sangat lengkap dan berjalan dengan cepat, Anda mungkin ingin mencoba menggunakan sistem keranjang belanja yang sudah ada. Salah satu keranjang belanja Open Source yang terkenal yang diimplementasikan dalam PHP adalah FishCartSQL, tersedia dari http://www.fishcart.org/ . Ini memiliki banyak fitur canggih seperti pelacakan pelanggan, penjualan berjangka waktu, berbagai bahasa, pemrosesan kartu kredit, dan dukungan untuk beberapa toko daring di satu server. Versi terkini (pada saat penulisan) ditulis dalam PHP 3 dan menggunakan PHPLib untuk kontrol sesi. Tentu saja, ketika Anda menggunakan sistem yang sudah ada, Anda selalu menemukan ada hal-hal yang tidak dimilikinya yang Anda inginkan, dan sebaliknya. Keuntungan dari produk Open Source adalah Anda dapat masuk dan mengubah hal-hal yang tidak Anda sukai.

BAB 5
MEMBANGUN SISTEM MANAJEMEN KONTEN

5.1 MENGEDIT KONTEN

Pertama, kita perlu memikirkan cara memasukkan konten ke dalam sistem, dan cara menyimpan serta mengedit konten tersebut.

Memasukkan Konten ke dalam Sistem

Kita perlu memutuskan cara pengiriman cerita dan komponen desain. Tiga metode yang mungkin dapat digunakan.

FTP

Penulis dan desainer dapat diberikan akses FTP ke area di server Web, dan mereka kemudian dapat mengunggah file dari komputer lokal mereka ke server. Perlu ada standar penamaan yang kaku untuk file yang diunggah (untuk mengidentifikasi gambar mana yang termasuk dalam cerita mana) atau sistem berbasis Web untuk menangani hal ini secara terpisah dari unggahan FTP. Penggunaan FTP juga menimbulkan masalah dengan izin dalam situasi ini. Karena fleksibilitas yang dibutuhkan oleh contoh ini, kita tidak akan menggunakan FTP untuk memungkinkan pengguna mengunggah file.

Metode Pengunggahan File

PHP dapat menangani hal ini dengan sangat mudah. Metode unggah berkas juga memberi kita kesempatan untuk menyimpan teks dalam basis data, bukan sebagai berkas. Untuk melakukannya, kita akan membaca berkas sementara dan menyimpan isinya dalam basis data, daripada menyalinnya ke tempat lain dalam sistem berkas. Kita tidak akan menggunakan unggah berkas untuk cerita dalam proyek ini. Kita akan membahas keunggulan basis data dibandingkan sistem berkas nanti.

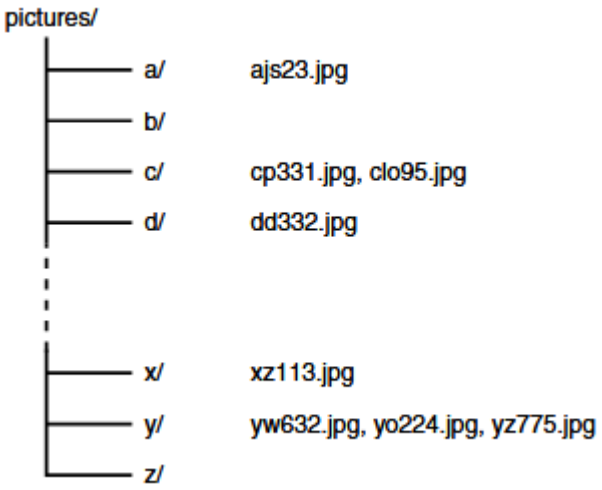
Penyuntingan Daring

Kita dapat mengizinkan pengguna membuat dan menyunting dokumen tanpa menggunakan FTP atau unggah berkas. Sebagai gantinya, Anda dapat memberi kontributor kotak masukan area teks besar di layar tempat konten cerita mereka dapat disunting. Metode ini sederhana, tetapi sering kali efektif. Peramban Web tidak menyediakan fasilitas penyuntingan teks apa pun di luar fungsi potong-tempel dari sistem operasi. Namun, ketika Anda hanya perlu membuat perubahan kecil misalnya, untuk mengoreksi kesalahan ejaan sangat cepat untuk memunculkan konten dan mengubahnya. Serupa dengan unggah berkas, data formulir dapat ditulis ke berkas atau disimpan dalam basis data.

Basis Data Versus Penyimpanan File

Keputusan penting yang harus dibuat pada tahap awal adalah bagaimana konten akan disimpan setelah diunggah ke dalam sistem. Karena kami akan menyimpan metadata di samping teks cerita, kami telah memilih untuk meletakkan bagian teks dari konten ke dalam basis data. Meskipun MySQL mampu menyimpan data multimedia, kami memilih untuk menyimpan gambar yang diunggah pada sistem file. Kami hanya akan menyimpan nama file gambar di basis data. Dengan menggunakan sistem file, tag `<IMG SRC>` dapat merujuk ke file

gambar secara langsung seperti biasa. Ketika sejumlah besar data disimpan, penting untuk mengoptimalkan penyimpanan data Anda. Sama seperti basis data yang memerlukan indeks yang tepat agar efisien, sistem file akan sangat diuntungkan dari struktur direktori yang dipikirkan dengan matang. Contohnya dapat dilihat pada Gambar 5.1.



Gambar 5.1 Struktur Direktori Ini Telah Disusun Untuk Pengunggahan Berkas.

Sistem berkas dalam kasus ini dibagi menjadi direktori yang mewakili karakter pertama dari setiap nama berkas. Oleh karena itu, 10.000 berkas akan tersebar di 26 direktori dengan sekitar 400 berkas di setiap direktori. Ini akan jauh lebih cepat diakses daripada 10.000 berkas sekaligus dalam satu direktori. Perlu diperhatikan bahwa nama berkas yang digunakan harus dibuat oleh skrip pemrosesan pengunggahan untuk memastikan bahwa nama tersebut unik. Contoh pada Gambar 5.1 mengasumsikan bahwa semua nama berkas akan dimulai dengan huruf kecil.

Struktur Dokumen

Contoh berita yang akan kita gunakan adalah berita pendek satu atau dua paragraf dengan satu gambar, yang dirancang untuk orang-orang yang terburu-buru. Dokumen tersebut terstruktur karena berisi judul berita dan satu atau dua paragraf teks dengan gambar. Semakin terstruktur suatu dokumen, semakin mudah dokumen tersebut dipecah untuk disimpan dalam basis data. Keuntungannya adalah semua dokumen dapat disajikan dengan cara yang sangat konsisten dan terstruktur. Ambil contoh berita kita. Kita akan menyimpan judul berita di kolom yang terpisah dari teks berita, dan menurut sifatnya gambar tersebut merupakan komponen terpisah dari dokumen. Dengan judul berita sebagai item terpisah, kita dapat menentukan jenis huruf dan gaya standar untuk menampilkannya, dan dapat dengan mudah memisahkannya dari sisa berita untuk membentuk halaman judul berita utama kita.

Pendekatan lain untuk dokumen besar adalah dengan memiliki hubungan satu-ke-banyak dengan paragraf individual; yaitu, menyimpan setiap paragraf sebagai baris terpisah dalam basis data, masing-masing ditautkan ke ID dokumen induk. Struktur dokumen dinamis semacam itu akan memungkinkan Anda untuk menyajikan halaman konten untuk setiap

dokumen dan menampilkan setiap bagian secara independen, atau menampilkan seluruh dokumen sekaligus.

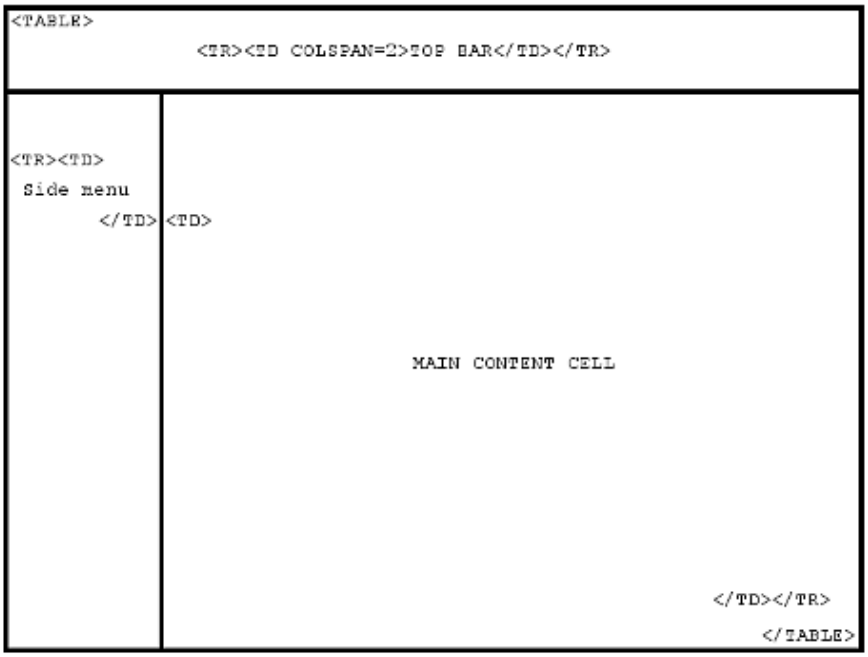
5.2 MENGGUNAKAN METADATA

Kami telah memutuskan bahwa setiap rekaman cerita terdiri dari judul, teks cerita, dan gambar. Namun, tidak ada alasan kami tidak dapat menyimpan data lain dalam rekaman yang sama. Sistem kami akan secara otomatis memasukkan nilai untuk siapa yang membuat cerita dan kapan terakhir dimodifikasi. Ini dapat secara otomatis ditampilkan di bagian bawah cerita untuk menandatangani dan memberi cap waktu tanpa penulis perlu khawatir menambahkan informasi. Mungkin juga berguna untuk menambahkan data yang tidak ditampilkan, yang dikenal sebagai metadata. Contoh bagusnya adalah menyimpan kata kunci yang akan digunakan untuk indeks mesin pencari. Daripada memindai seluruh teks setiap cerita, mesin pencari akan melihat metadata kata kunci untuk setiap cerita dan menentukan relevansi hanya dari itu. Ini memungkinkan administrator situs untuk memiliki kendali penuh atas kata dan frasa pencarian mana yang cocok dengan dokumen mana.

Dalam contoh ini, kita akan mengizinkan sejumlah kata kunci untuk dikaitkan dengan sebuah cerita, dan menetapkan nilai bobot pada setiap kata kunci untuk menunjukkan seberapa relevan kata kunci tersebut pada skala 1 hingga 10. Kita kemudian dapat mengembangkan algoritma mesin pencari yang memberi peringkat kecocokan menurut relevansi yang ditentukan manusia untuk cerita, daripada algoritma rumit yang harus menafsirkan prosa bahasa Inggris dan membuat keputusan berdasarkan pemahamannya yang terbatas dan diatur oleh aturan yang tetap. Ini bukan berarti Anda harus menyimpan metadata dalam basis data. Tidak ada yang menghentikan Anda untuk menggunakan tag <META> dalam HTML, atau bahkan menggunakan XML untuk membuat dokumen Anda. Namun, ada baiknya memanfaatkan basis data yang sudah mengurus dokumen Anda kapan pun Anda bisa.

Memformat Output

Contoh situs berita kita mengikuti format yang sederhana tetapi terstruktur saat menampilkan halaman. Setiap halaman berisi sejumlah cerita yang semuanya diformat dengan cara yang sama. Pertama-tama, judul ditampilkan dalam huruf besar, diikuti oleh foto di bawahnya di sebelah kiri dan kemudian teks cerita di sebelah kanan. Seluruh halaman terdapat dalam templat halaman standar untuk mempertahankan pencitraan merek dan konsistensi situs secara menyeluruh. Gambar 5.2 menunjukkan struktur halaman logis yang akan kita gunakan.



Gambar 5.2 Struktur Halaman

Tata letak seperti ini sangat populer berapa banyak situs yang Anda kunjungi secara rutin yang memiliki bilah menu di sisi kiri atau tautan cepat di bagian atas, dengan navigasi luar tetap ada di tempatnya, tidak peduli halaman mana yang Anda lihat?

Menerapkan struktur templat seperti ini dari desain halaman sangatlah mudah. Dalam sumber HTML, Anda akan menemukan <TD> tempat sel konten utama dimulai. Pada titik ini, bagi HTML menjadi dua berkas; separuh pertama menjadi berkas header dan separuh kedua menjadi footer. Setiap kali Anda menampilkan halaman, tampilkan header terlebih dahulu, halaman, dan terakhir footer. Mengimplementasikan situs dengan templat header dan footer memungkinkan berkas templat ini mudah diubah jika desain situs diperbarui.

PHP menyediakan dua opsi konfigurasi yang tampaknya berguna dalam situasi ini. Direktif `auto_prepend_file` dan `auto_append_file` dapat ditetapkan berdasarkan per direktori untuk menentukan berkas yang disertakan sebelum atau setelah skrip diproses. Namun, ini memiliki beberapa keterbatasan. Jika Anda memiliki skrip yang tidak menghasilkan keluaran dan mengirim header pengalihan seperti

```
<? header("Location: destination.php"); ?>
```

berkas header dan footer akan tetap ditampilkan dan pengalihan akan gagal karena keluaran telah dikirim ke peramban Web. Hal ini juga menyebabkan masalah dengan cookie dan fungsi manajemen sesi PHP 4 bawaan karena header cookie tidak akan berfungsi dengan benar setelah output apa pun dikirim ke peramban Web.

Manipulasi Gambar

Penulis yang menyumbangkan cerita mungkin akan menyediakan foto mereka sendiri untuk melengkapi cerita. Kami menginginkan konsistensi, tetapi apa yang terjadi ketika

seorang penulis mengunggah gambar besar dan berkualitas tinggi dan penulis lain mengunggah gambar mini kecil? Dengan asumsi bahwa gambar yang dimaksud terutama adalah foto, kami dapat menggunakan gambar JPEG saja, dan memanfaatkan fungsi dalam PHP untuk memanipulasi gambar. Kami telah membuat skrip sederhana yang disebut `resize_image.php` yang akan mengubah ukuran gambar secara langsung sehingga dapat ditampilkan dengan tag `<IMG SRC>`. Skrip ini ditampilkan dalam listing 5.1.

Listing 5.1 `Resize_Image.Php` Mengubah Ukuran Gambar Jpeg Secara Langsung

```
<?
if (!$max_width)
  $max_width = 150;
if (!$max_height)
  $max_height = 100;

$size = GetImageSize($image);
$width = $size[0];

$height = $size[1];

$x_ratio = $max_width / $width;
$y_ratio = $max_height / $height;

if ( ($width <= $max_width) && ($height <= $max_height) ) {
  $tn_width = $width;
  $tn_height = $height;
}
else if (($x_ratio * $height) < $max_height) {
  $tn_height = ceil($x_ratio * $height);
  $tn_width = $max_width;
}
else {
  $tn_width = ceil($y_ratio * $width);
  $tn_height = $max_height;
}

$src = ImageCreateFromJpeg($image);
$dst = ImageCreate($tn_width,$tn_height);
ImageCopyResized($dst, $src, 0, 0, 0, 0,
  $tn_width,$tn_height,$width,$height);
header("Content-type: image/jpeg");
ImageJpeg($dst, null, -1);
ImageDestroy($src);
ImageDestroy($dst);
?>
```

Skrip tersebut mengambil tiga parameter nama berkas gambar yang akan ditampilkan, lebar maksimum, dan tinggi maksimum dalam piksel. Ini bukan berarti bahwa jika ukuran maksimum yang ditetapkan adalah 200x200, gambar akan diskalakan menjadi 200x200. Melainkan, gambar akan diperkecil secara proporsional sehingga lebar atau tinggi yang lebih besar adalah 200 piksel dan dimensi lainnya adalah 200 piksel atau lebih kecil. Misalnya, gambar berukuran 400x300 akan diperkecil menjadi 200x150. Dengan cara ini, proporsi gambar akan dipertahankan.

Mengubah ukuran secara cepat di server merupakan pilihan yang lebih baik daripada hanya menetapkan atribut HEIGHT dan WIDTH ke tag <IMG SRC>. Gambar besar beresolusi tinggi yang dikirimkan penulis mungkin berukuran beberapa megabita; tetapi jika diskalakan ke ukuran yang wajar, ukurannya bisa di bawah 100k. Dengan demikian, Anda tidak perlu mengunduh berkas yang besar dan meminta peramban untuk mengubah ukurannya. Di sini, kami menggunakan fungsi ImageCopyResized() untuk mengubah skala gambar secara cepat ke ukuran yang dibutuhkan. Kunci untuk operasi pengubahan ukuran adalah kalkulasi parameter lebar dan tinggi yang baru. Kami menemukan rasio antara dimensi aktual dan maksimum. $max_width dan $max_height dapat dimasukkan ke dalam string kueri; jika tidak, nilai default yang ditetapkan di bagian atas daftar akan digunakan.

```
$x_ratio = $max_width / $width;
$y_ratio = $max_height / $height;
```

Jika gambar sudah lebih kecil dari dimensi maksimum yang ditentukan, lebar dan tinggi tidak akan diubah. Jika tidak, rasio X atau Y digunakan untuk menskalakan kedua dimensi secara merata sehingga gambar yang diperkecil tidak melebar atau terjepit, seperti berikut:

```
if ( ($width <= $max_width) && ($height <= $max_height) ) {
      $tn_width = $width;
      $tn_height = $height;
}
else if (($x_ratio * $height) < $max_height) {
      $tn_height = ceil($x_ratio * $height);
      $tn_width = $max_width;
}
else {
      $tn_width = ceil($y_ratio * $width);
      $tn_height = $max_height;
}
```

Desain/Gambaran Umum Solusi

Ringkasan berkas dalam aplikasi ini ditunjukkan pada Tabel 5.1.

Tabel 5.1 Berkas Dalam Aplikasi Manajemen Konten

Nama	Tipe	Deskripsi
create_database.sql	SQL	SQL untuk menyiapkan database konten dan beberapa contoh data
include_fns.php	Fungsi	Koleksi file penyertaan untuk aplikasi
db_fns.php	Fungsi	Kumpulan fungsi untuk menghubungkan ke database konten
select_fns.php	Fungsi	Kumpulan fungsi untuk membantu pembuatan daftar SELECT
user_auth_fns.php	Fungsi	Kumpulan fungsi untuk mengautentikasi pengguna
header.php	Template	Ditampilkan di bagian atas setiap halaman konten
footer.php	Template	Ditampilkan di bagian bawah setiap halaman konten
logo.gif	Gambar	File logo ditampilkan di header.php
headlines.php	Aplikasi	Menampilkan judul terbaru dari setiap halaman situs
page.php	Aplikasi	Mencantumkan judul berita dan teks cerita untuk halaman tertentu
resize_image.php	Aplikasi	Mengubah ukuran gambar secara cepat untuk headlines.php
search_form.php	Aplikasi	Formulir untuk memasukkan kata kunci untuk mencari konten situs
search.php	Aplikasi	Menampilkan judul konten yang cocok dengan kriteria kata kunci
login.php	Aplikasi	Mengotentikasi kata sandi pengguna dan memasukkannya ke dalam sistem
logout.php	Aplikasi	Mengeluarkan pengguna dari sistem
stories.php	Aplikasi	Mencantumkan cerita yang telah ditulis oleh pengguna yang masuk dengan opsi untuk menambahkan, mengubah, atau menghapus cerita
story.php	Aplikasi	Layar detail cerita untuk mengedit atau menambahkan cerita baru
story_submit.php	Aplikasi	Menambahkan cerita baru atau melakukan perubahan dari data yang dimasukkan di story.php
delete_story.php	Aplikasi	Memproses permintaan penghapusan cerita dari stories php
keywords.php	Aplikasi	Mencantumkan kata kunci untuk sebuah cerita dengan opsi untuk menambahkan atau menghapus kata kunci
keyword_add.php	Aplikasi	Proses permintaan penambahan kata kunci dari keywords.php
keyword_delete.php	Aplikasi	Memproses permintaan penghapusan kata kunci dari keywords.php

publish.php	Aplikasi	Daftar cerita editor yang menunjukkan cerita mana yang diterbitkan dengan opsi untuk mengubah status masing-masing
publish_story.php	Aplikasi	Memproses permintaan publikasi dari publish.php
unpublish_story.php	Aplikasi	Memproses permintaan batal terbit dari publish.php

Mendesain Basis Data

Listing 5.2 menunjukkan kueri SQL yang digunakan untuk membuat basis data bagi sistem konten. Daftar ini merupakan bagian dari berkas create_database.sql. Berkas pada CD juga berisi kueri untuk mengisi basis data dengan beberapa contoh pengguna dan cerita.

Listing 5.2 Kutipan dari create_database.sql—Berkas SQL untuk Menyiapkan Basis Data Konten

```
drop database if exists content;

create database content;

use content;

drop table if exists writers;

create table writers (
  username    varchar(16) primary key,
  password    varchar(16) not null,
  full_name   text
);

drop table if exists stories;

create table stories (
   id      int primary key auto_increment,
   writer        varchar(16) not null,   # foreign key writers.username
   page          varchar(16) not null,   # foreign key pages.code
   headline      text,
   story_text    text,
   picture       text,
   created       int,
   modified      int,
   published     int
);

drop table if exists pages;
```

```
create table pages (
    code        varchar(16) primary key,
    description text
);
drop table if exists writer_permissions;

create table writer_permissions (
    writer    varchar(16) not null,       # foreign key writers.username
    page     varchar(16) not null         # foreign key pages.code
);

drop table if exists keywords;

create table keywords (
  story      int not null,                # foreign key stories.id
  keyword    varchar(32) not null,
  weight     int not null
);

grant select, insert, update, delete
on content.*
to content@localhost identified by 'password';
```

Kita perlu menyimpan sedikit informasi tentang setiap penulis, termasuk nama login dan kata sandi, di tabel writers. Kita akan menyimpan nama lengkap mereka untuk ditampilkan setelah setiap artikel dan untuk menyapa mereka saat mereka login.

Tabel pages berisi judul halaman untuk setiap halaman tempat cerita dapat ditampilkan. Tabel writer_permissions menerapkan hubungan many-to-many yang menunjukkan halaman mana yang dapat digunakan penulis untuk mengirimkan cerita.
Tabel stories berisi kolom terpisah untuk headline, story_text, dan picture seperti yang dibahas sebelumnya. Kolom created, modified, dan published adalah kolom integer dan akan menyimpan nilai cap waktu Unix dari waktu yang relevan. Untuk membuat database, jalankan perintah berikut:

```
mysql –u root < create_database.sql
```

Implementasi

Sekarang setelah kita memiliki database dan fungsi pengubahan ukuran gambar untuk digambar, mari kita mulai membangun bagian utama sistem.

Front End

Mari kita mulai dengan melihat `headlines.php`, yang ditampilkan dalam Listing 5.3, yang merupakan halaman pertama yang akan dilihat pengunjung situs. Kita ingin menunjukkan kepadanya berita utama dari setiap halaman.

Listing 5.3 headlines.php Menampilkan Judul Terbaru dari Setiap Halaman

```
<?
include (“include_fns.php”);
include (“header.php”);

$conn = db_connect();

$pages_sql = “select * from pages order by code”;
$pages_result = mysql_query($pages_sql, $conn);

while ($pages = mysql_fetch_array($pages_result)) {

$story_sql = “select * from stories
               where page = ‘$pages[code]’
               and published is not null
               order by published desc”;
$story_result = mysql_query($story_sql, $conn);
if (mysql_num_rows($story_result)) {
  $story = mysql_fetch_array($story_result);
  echo “<TABLE BORDER=0 WIDTH=400>”;
  echo “<TR>”;
  echo “<TD ROWSPAN=2 WIDTH=100>”;
  if ($story[picture])
  echo “<IMG SRC=\”resize_image.php?image=$story[picture]\”>”;
  echo “</TD>”;
  echo “<TD>”;
  echo “<H3>$pages[description]</H3>”;
  echo $story[headline];
  echo “</TD>”;
  echo “</TR>”;
  echo “<TR><TD ALIGN=RIGHT>”;
  echo “<A HREF=\”page.php?page=$pages[code]\”>”;
  echo “<FONT SIZE=1>Read more $pages[code] ...</FONT>”;
  echo “</A>”;
  echo “</TABLE>”;
  }
 }

include (“footer.php”);
?>
```

Skrip ini, seperti halnya semua skrip publik, menyertakan header.php di awal dan footer.php di akhir. Oleh karena itu, setiap keluaran yang dihasilkan oleh skrip ditampilkan dalam sel konten utama di halaman. Pekerjaan berat dilakukan oleh dua kueri basis data. Pertama,

```
select * from pages order by code
```

akan mengambil daftar halaman yang ada di basis data. Selanjutnya, isi loop

```
select * from stories
where page = '$pages[code]'
and published is not null
order by published desc
```

dieksekusi untuk menemukan cerita pada halaman tersebut dalam urutan terbalik dari tanggal penerbitan. Gambar 5.3 menunjukkan output dari `headline.php` menggunakan data aplikasi contoh.

Gambar 5.3 Menampilkan Berita Utama Dari Setiap Halaman Dalam Situs.

Di samping setiap berita utama, tautan dibuat dalam bentuk berikut:

```
<A HREF="page.php?page=1"><FONT SIZE=1>Read more news...</FONT></A>
```

Hal ini dilakukan dalam loop sebelumnya sehingga nilai string kueri `$page` dan nama halaman dicetak di samping berita utama yang relevan. Mengklik tautan ini akan membawa pengunjung ke `page.php` daftar lengkap berita untuk halaman tertentu. Sumber untuk `page.php` dapat ditemukan di listing 5.4.

Listing 5.4 page.php Menampilkan Semua Cerita yang Diterbitkan pada Halaman

```
<?
include (“include_fns.php”);
include (“header.php”);

$conn = db_connect();

if (!$page) {
 header(“Location: headlines.php”);
 exit;
}

$sql = “select * from stories
        where page = ‘$page’
        and published is not null
        order by published desc”;
$result = mysql_query($sql, $conn);

while ($story = mysql_fetch_array($result))
{
 echo “<H2>”.$story[headline].”</H2>”;
 if ($story[picture]) {
    $size = getImageSize($story[picture]);
    $width  = $size[0];
    $height = $size[1];
    echo “<IMG SRC=\”$story[picture]\” HEIGHT=$height WIDTH=$width
ALIGN=LEFT>”;
     }
     echo $story[story_text];
     $w = get_writer_record($story[writer]);
     echo “<br><FONT SIZE=1>”;
     echo $w[full_name].”, “;
     echo date(“M d, H:i”, $story[modified]);
     echo “</FONT>”;
}

include (“footer.php”);
?>
```

akan mengarahkan pengunjung kembali ke halaman judul sehingga penghilangan $page tidak akan menyebabkan kesalahan. *Permintaan pertama:*

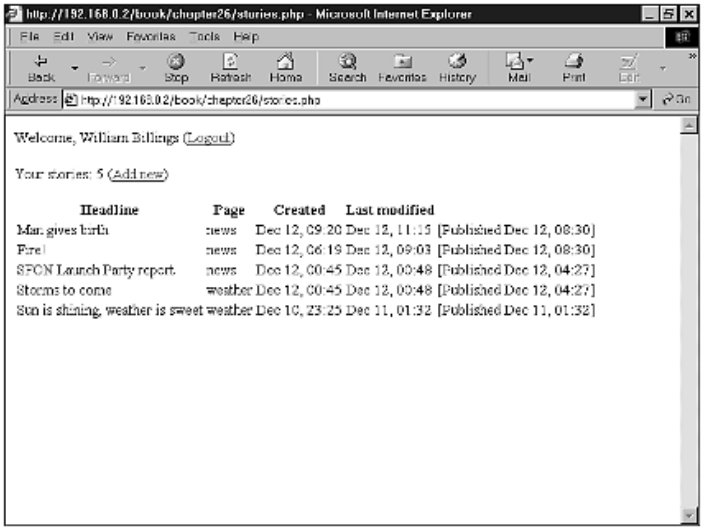
```
if (!$page) {
    header(“Location: headlines.php”);
    exit;
}
```

menemukan semua berita yang diterbitkan pada halaman yang ditentukan, dengan berita yang paling baru diterbitkan terlebih dahulu. Dalam setiap putaran, gambar yang diunggah dan teks berita dicetak ke layar, diikuti oleh nama penulis dan tanggal perubahan terakhir. Gambar 5.4 menunjukkan page.php yang sedang beraksi, menampilkan semua item halaman berita untuk contoh aplikasi kita.

Gambar 5.4 Menampilkan Semua Berita Yang Diterbitkan Pada Halaman Berita.

Back End

Mari kita lihat selanjutnya bagaimana berita dapat ditambahkan ke sistem. Penulis memulai di `stories.php`. Skrip ini, setelah penulis diautentikasi, menampilkan daftar berita yang telah ditulis penulis dan menampilkan tanggal penerbitan, atau menawarkan opsi untuk menambahkan berita baru, mengedit atau menghapus berita yang sudah ada, atau mengatur kata kunci pencarian. Contoh ditunjukkan pada Gambar 5.5.



Gambar 5.5 Halaman Manajemen Cerita Untuk Penulis.

Layar ini tidak diformat di dalam berkas header dan footer, meskipun bisa saja diformat jika diinginkan. Karena hanya penulis dan editor yang akan menggunakan skrip ini, dalam contoh kami, kami hanya memilih untuk memformat sebanyak yang diperlukan untuk membuat sistem yang dapat digunakan. Kode untuk `stories.php` ditampilkan dalam Daftar 5.5.

Listing 5.5 `stories.php` Adalah Antarmuka bagi Penulis untuk Mengelola Cerita Mereka

```
<?
include (“include_fns.php”);
session_register(“auth_user”);

if (!check_auth_user()) {
?>
<FORM ACTION=”login.php” METHOD=POST>
<TABLE BORDER=0>
<TR>
  <TD>Username</TD>
  <TD><INPUT SIZE=16 NAME=”username”></TD>
</TR>
<TR>
  <TD>Password</TD>
  <TD><INPUT SIZE=16 TYPE=”PASSWORD” NAME=”password”></TD>
</TR>
</TABLE>
<INPUT TYPE=SUBMIT VALUE=”Log in”>
</FORM>
<?
}
else {
 $conn = db_connect();

 $w = get_writer_record($auth_user);
 echo “Welcome, “.$w[full_name];
 echo “ (<A HREF=\”logout.php\”>Logout</A>)”;
 echo “<p>”;

 $sql = “select * from stories where writer = ‘$auth_user’ “.
       “order by created desc”;
 $result = mysql_query($sql, $conn);

 echo “Your stories: “;
 echo mysql_num_rows($result);
 echo “ (<A HREF=\”story.php\”>Add new</A>)”;
 echo “<br><br>”;

if (mysql_num_rows($result)) {
```

```
 echo “<TABLE>”;
 echo “<TR><TH>Headline</TH><TH>Page</TH>”;
 echo “<TH>Created</TH><TH>Last modified</TH></TR>”;
 while ($qry = mysql_fetch_array($result)) {
    echo “<TR>”;
    echo “<TD>”;
    echo $qry[headline];
    echo “</TD>”;
    echo “<TD>”;
    echo $qry[page];
    echo “</TD>”;
    echo “<TD>”;
    echo date(“M d, H:i”, $qry[created]);
    echo “</TD>”;
    echo “<TD>”;
    echo date(“M d, H:i”, $qry[modified]);
    echo “</TD>”;
    echo “<TD>”;
    if ($qry[published])
     echo “[Published “.date(“M d, H:i”, $qry[published]).”]”;
    else {

    echo “[<A HREF=\”story.php?story=”.$qry[id].”\”>edit</A>] “;
    echo “[<A HREF=\”delete_story.php?story=”.$qry[id].”\”>delete</A>] “;
    echo “[<A HREF=\”keywords.php?story=”.$qry[id].”\”>keywords</A>]”;
   }
   echo “</TD>”;
   echo “</TR>”;
 }
 echo “</TABLE>”;
}

}
?>
```

Langkah pertama adalah memeriksa apakah pengguna telah diautentikasi, dan jika tidak, hanya menampilkan formulir login. Variabel sesi `$auth_user` akan ditetapkan setelah penulis login. Autentikasi di sini tidak terlalu aman, dan pada kenyataannya Anda akan lebih berhati-hati untuk memastikan bahwa penulis diautentikasi dengan benar. Formulir login dikirimkan ke `login.php`, yang memeriksa nama pengguna dan kata sandi terhadap nilai basis data. Jika login berhasil, pengguna dikembalikan ke halaman asalnya, menggunakan nilai $HTTP_REFERER. Ini berarti bahwa skrip login dapat dipanggil dari halaman pemanggil mana pun dalam sistem. Selanjutnya, kami menyambut penulis dengan namanya dan memberinya kesempatan untuk keluar. Tautan ini akan selalu muncul di bagian atas `stories.php` sehingga ia dapat dengan mudah keluar setelah selesai.

```
$w = get_writer_record($auth_user);

echo “Welcome, “.$w[full_name];
echo “ (<A HREF=\”logout.php\”>Logout</A>)”;
```

Fungsi get_writer_record() didefinisikan dalam db_fns.php dan mengembalikan array dari semua kolom dalam tabel penulis untuk nama pengguna yang dimasukkan. Skrip logout.php cukup menghapus nilai $auth_user. SQL berikut menemukan semua cerita penulis, dimulai dengan yang paling baru ditambahkan:

```
select * from stories where writer = ‘$auth_user’
order by created desc
```

Kami menyimpan stempel waktu yang dibuat, dimodifikasi, dan dipublikasikan pada setiap rekaman cerita. Saat cerita baru ditambahkan, stempel waktu yang dibuat dan dimodifikasi akan disetel ke waktu sistem. Setiap perubahan berikutnya hanya memperbarui bidang yang dimodifikasi. Semua informasi ini ditampilkan di layar cerita, pertama dengan:

```
echo date(“M d, H:i”, $qry[created]);
```

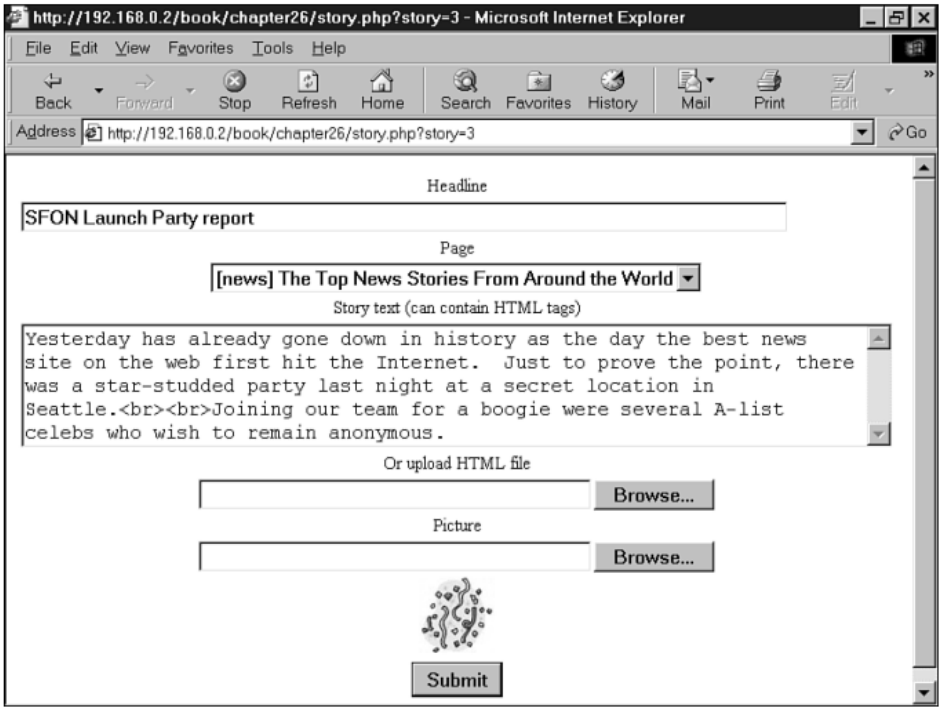
dan kemudian

```
echo date(“M d, H:i”, $qry[modified]);
```

dan terakhir

```
if ($qry[published])
  echo “[Published “.date(“M d, H:i”, $qry[published]).”]”;
else {
  echo “[<A HREF=\”story.php?story=”.$qry[id].”\”>edit</A>] “;
  echo “[<A HREF=\”delete_story.php?story=”.$qry[id].”\”>delete</A>] “;
  echo “[<A HREF=\”keywords.php?story=”.$qry[id].”\”>keywords</A>]”;
}
```

Ini akan menampilkan tanggal terbit jika sesuai; jika tidak, akan ditampilkan tautan untuk mengedit atau menghapus cerita tersebut dan untuk menetapkan kata kunci pencarian. Skrip untuk memasukkan cerita baru atau mengedit cerita yang sudah ada adalah story.php, dan ditunjukkan pada Gambar 5.6 saat mengedit salah satu cerita dalam contoh basis data aplikasi.



Gambar 26.6 Mengedit Cerita.

Listing 5.6 `story.php` Digunakan untuk Membuat atau Mengedit Cerita

```
<?
include (“include_fns.php”);
if (isset($story))
 $s = get_story_record($story);
?>

<FORM ACTION=”story_submit.php” METHOD=POST ENCTYPE=”multipart/form-data”>
<INPUT TYPE=HIDDEN NAME=”story” VALUE=”<? echo $story;?>”>
<INPUT TYPE=HIDDEN NAME=”destination” VALUE=”<? echo $HTTP_REFERER;?>”>
<TABLE>

<TR>
  <TD ALIGN=CENTER>Headline<TD>
</TR>
<TR>
 <TD><INPUT SIZE=80 NAME=”headline”
             VALUE=”<? echo $s[headline];?>”></TD>
</TR>

<TR>
  <TD ALIGN=CENTER>Page<TD>
</TR>
<TR>
  <TD ALIGN=CENTER><? echo query_select(“page”,
            “select p.code, p.description
```

```
            from pages p, writer_permissions w
            where p.code = w.page
            and w.writer = '$auth_user'", $s[page]);?></TD>
</TR>

<TR>
  <TD ALIGN=CENTER>Story text (can contain HTML tags)</TD>
</TR>
<TR>
  <TD><TEXTAREA COLS=80 ROWS=7 NAME="story_text"
           WRAP=VIRTUAL><? echo $s[story_text];?></TEXTAREA>
  </TD>
</TR>
<TR>
 <TD ALIGN=CENTER>Or upload HTML file</TD>
</TR>
<TR>
 <TD ALIGN=CENTER><INPUT TYPE=FILE NAME="html" SIZE=40></TD>
</TR>

<TR>
  <TD ALIGN=CENTER>Picture</TD>
</TR>
<TR>
  <TD ALIGN=CENTER><INPUT TYPE=FILE NAME="picture" SIZE=40></TD>
</TR>

<?
if ($s[picture]) {
 $size   = getImageSize($s[picture]);
 $width  = $size[0];
 $height = $size[1];
?>
<TR>
  <TD ALIGN=CENTER>
    <IMG SRC="<? echo $s[picture];?>"
      WIDTH=<? echo $width;?> HEIGHT=<? echo $height;?>>
  </TD>
</TR>
<? } ?>

<TR>
    <TD ALIGN=CENTER><INPUT TYPE=SUBMIT VALUE="Submit"></TD>
</TR>

</TABLE>
</FORM>
```

Skrip yang sama dapat digunakan untuk menambah atau mengedit, dan tindakannya bergantung pada apakah $story ditetapkan saat skrip dipanggil.

```
if (isset($story))
$s = get_story_record($story);
```

Fungsi get_story_record() didefinisikan dalam db_fns.php dan mengembalikan array dari semua kolom dalam tabel stories untuk ID story yang ditentukan. Jika tidak ada ID story yang dimasukkan, $story akan bernilai null dan $s tidak akan berisi elemen array.

```
<INPUT SIZE=80 NAME="headline" VALUE="<? echo $s[headline];?>">
```

Jika $story tidak ditetapkan, kode sebelumnya tidak akan menghasilkan nilai apa pun dari pernyataan PHP, sehingga kotak masukan judul akan kosong. Jika $story ditetapkan, kotak tersebut akan berisi teks judul untuk berita yang sedang diedit.

```
echo query_select("page",
"select p.code, p.description
from pages p, writer_permissions w
where p.code = w.page
and w.writer = '$auth_user'", $s[page]);
```

Fungsi query_select() didefinisikan dalam select_fns.php dan mengembalikan kode HTML untuk menghasilkan daftar SELECT dari kueri SQL yang diberikan. Parameter pertama adalah atribut NAME untuk SELECT. Kueri SQL dalam parameter kedua memilih dua kolom, yang pertama adalah bagian VALUE dari setiap opsi, dan yang kedua muncul setelah tag OPTION dan merupakan teks yang sebenarnya ditampilkan dalam daftar. Parameter ketiga bersifat opsional. Ia menambahkan atribut SELECTED ke opsi yang nilainya cocok dengan nilai yang ditentukan.

```
<INPUT TYPE=HIDDEN NAME="story" VALUE="<?echo $story;?>">
```

Ini menyiapkan variabel placeholder, yang menetapkan nilai baru untuk story dari $story yang dimasukkan. Saat formulir dikirimkan, story_submit.php memeriksa apakah ada nilai untuk $story dan menghasilkan pernyataan SQL UPDATE atau INSERT yang sesuai. Kode untuk story_submit.php ditampilkan dalam Listing 5.7.

Listing 5.7 story_submit.php Digunakan untuk Memasukkan atau Memperbarui Cerita dalam Database

```
<?

// story_action.php
// add / modify story record
```

```
include ("include_fns.php");

$conn = db_connect();

$time = time();

if ( ($html) && (dirname($html_type) == "text") ) {
 $fp = fopen($html, "r");
 $story_text = addslashes(fread($fp, filesize($html)));
 fclose($fp);
}

if ($story) {   // It's an update
    $sql = "update stories

        set headline = '$headline',
            story_text = '$story_text',
            page = '$page',
            modified = $time
        where id = $story";
}
else {          // It's a new story
 $sql = "insert into stories
          (headline, story_text, page, writer, created, modified)
        values
         ('$headline', '$story_text', '$page', '$auth_user', $time, $time)";
}

$result = mysql_query($sql, $conn);

if (!$result) {
 echo "There was a database error when executing <PRE>$sql</PRE>";
 echo mysql_error();
 exit;
}

if ( ($picture) && ($picture != "none") ) {

  if (!$story)
   $story = mysql_insert_id();

 $type = basename($picture_type);

 switch ($type) {
  case "jpeg":
  case "pjpeg":   $filename = "pictures/$story.jpg";
```

```
                    copy ($picture, $filename);
                    $sql = "update stories
                            set picture = '$filename'
                            where id = $story";
                    $result = mysql_query($sql, $conn);
                    break;
   default:        echo "Invalid picture format: $picture_type";
  }
}

header("Location: $destination");
?>
```

Tautan hapus cerita memanggil `delete_story.php`, yang menjalankan pernyataan DELETE sederhana dan mengembalikan penulis ke halaman pemanggil. Kode untuk `delete_story.php` ditunjukkan pada Listing 5.8.

Listing 5.8 `delete_story.php` Digunakan untuk Menghapus Cerita dari Basis Data

```
<?
// delete_story.php

include ("include_fns.php");

$conn = db_connect();

$now = time();

$sql = "delete from stories where id = $story";
$result = mysql_query($sql, $conn);

header("Location: $HTTP_REFERER");
?>
```

Pencarian

Mengklik tautan kata kunci pada daftar cerita akan memunculkan formulir baru untuk memasukkan kata kunci pada cerita tersebut. Tidak ada batasan jumlah kata kunci yang dapat dimasukkan, dan setiap kata kunci diberi nilai bobot, dengan nilai yang lebih tinggi menunjukkan bahwa kata kunci tersebut lebih relevan. Gambar 5.7 menunjukkan layar yang digunakan untuk menetapkan kata kunci pada cerita tertentu.

- Skrip `keywords.php` cukup mudah, jadi kita tidak akan membahasnya secara mendetail. Skrip tersebut disertakan pada CD-ROM.
- Skrip ini memicu skrip `keyword_add.php` dan `keyword_delete.php`. Skrip ini juga mudah, dan karenanya tidak disertakan di sini.

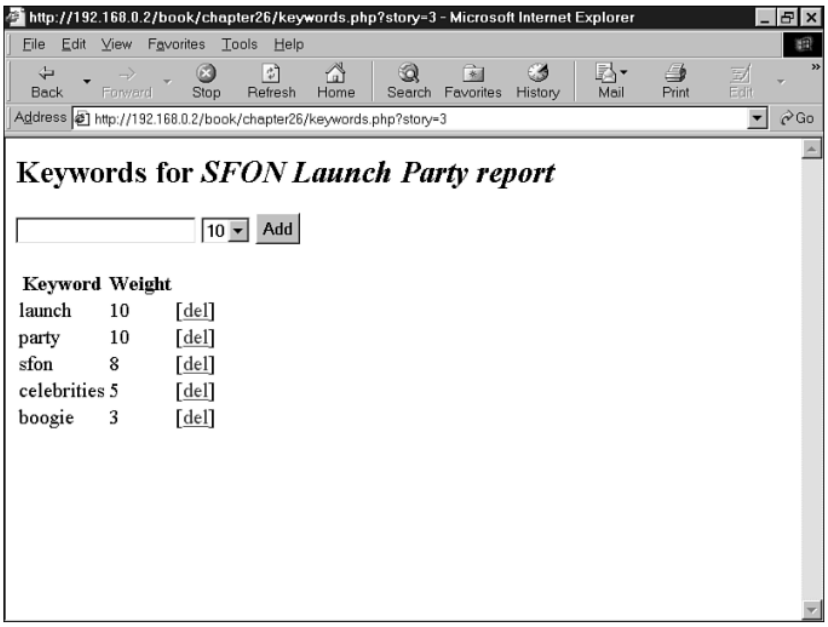
➲ Skrip keyword_add.php menggunakan kueri berikut untuk menambahkan kata kunci baru ke dalam basis data:

```
insert into keywords (story, keyword, weight)
values ($story, '$keyword', $weight)
```

Dalam konteks yang sama, keyword_delete.php menggunakan kueri berikut untuk menghapus kata kunci:

```
delete from keywords where story = $story and keyword = '$keyword'
```

Yang menarik adalah cara nilai bobot digunakan untuk menghitung angka persentase relevansi saat melakukan penelusuran.



Gambar 5.7 Menetapkan Kata Kunci Untuk Sebuah Cerita.

Formulir pencarian di search_form.php berisi satu bidang untuk kata kunci, dan dikirimkan ke search.php, yang meminta basis data cerita langsung untuk menemukan konten yang cocok. Sumber untuk search.php ditunjukkan pada listing 5.9.

Listing 5.9 search.php Menemukan Cerita yang Cocok dan Menghitung Skor Kecocokan Persentase

```
<?
include ("include_fns.php");
include ("header.php");

$conn = db_connect();
```

```
if ($keyword) {
  $k = split(" ", $keyword);
  $num_keywords = count($k);
  for ($i=0; $i<$num_keywords; $i++) {
    if ($i)
      $k_string .= "or k.keyword = '".$k[$i]."' ";
    else
      $k_string .= "k.keyword = '".$k[$i]."' ";
   }
   $and .= "and ($k_string) ";
}

$sql = "select s.id,
                s.headline,
                10 * sum(k.weight) / $num_keywords as score
          from stories s, keywords k
          where s.id = k.story
          $and
          group by s.id, s.headline
          order by score desc, s.id desc";

$result = mysql_query($sql, $conn);

echo "<H2>Search results</H2>";

if (mysql_num_rows($result)) {
  echo "<TABLE>";
  while ($qry = mysql_fetch_array($result)) {
    echo "<TR><TD>";
    echo $qry[headline];
    echo "</TD><TD>";
    echo floor($qry[score])."%";
    echo "</TD></TR>";
  }
  echo "</TABLE>";
}
else {
  echo "No matching stories found";
}");
include ("footer.php");
?>
```

Pertama, rangkaian kata kunci yang dimasukkan ke dalam skrip dibagi menjadi kata-kata pencarian individual. Kami tidak menggunakan teknik pencarian tingkat lanjut dalam contoh ini, seperti mengizinkan pencari untuk menggunakan kata kunci AND atau OR atau mengelompokkan kata-kata menjadi frasa.

```
if ($keyword) {
  $k = split(“ “, $keyword);
  $num_keywords = count($k);
  for ($i=0; $i<$num_keywords; $i++) {
   if ($i)
    $k_string .= “or k.keyword = ‘“.$k[$i].”’ “;
    else
      $k_string .= “k.keyword = ‘“.$k[$i].”’ “;
    }
    $and .= “and ($k_string) “;
}
```

Kode ini menggunakan fungsi PHP `split()` untuk membuat array yang berisi setiap kata dalam string `$keyword` yang dipisahkan oleh karakter spasi. Jika hanya satu kata yang ditentukan, ia tetap mengembalikan array elemen tunggal dan loop berikutnya dieksekusi sekali. Pada akhirnya kondisi yang disimpan dalam `$and` akan terlihat seperti ini

```
and (k.keyword = ‘keyword1’ or k.keyword = ‘keyword2’ or k.keyword =
‘keyword3’)
```

Kueri pencarian yang dibangun berdasarkan kode sebelumnya akan menjadi

```
select s.id,
        s.headline,
        10 * sum(k.weight) / $num_keywords as score
from stories s, keywords k
where s.id = k.story
and (k.keyword = ‘keyword1’
 or k.keyword = ‘keyword2’
 or k.keyword = ‘keyword3’)
group by s.id, s.headline
order by score desc, s.id desc
```

Perhitungan skor adalah jumlah bobot dari semua kata kunci yang cocok dibagi dengan jumlah kata kunci yang dicari lalu dikalikan sepuluh. Ini lebih baik untuk pencarian yang semua kata kunci yang dimasukkan cocok dengan kata kunci dalam basis data. Karena bobot berkisar dari 1 hingga 10, nilai maksimum untuk skor adalah 100. Pencarian untuk tiga kata kunci hanya akan menjadi kecocokan 100% dengan sebuah cerita jika ketiganya ditemukan untuk cerita itu dan masing-masing memiliki bobot 10.

5.3 LAYAR EDITOR

Satu-satunya bagian dari sistem yang belum kita bahas adalah bagaimana sebuah cerita benar-benar diterbitkan setelah ditulis. Skrip publisher.php membuat cerita menjadi aktif. Skrip ini ditampilkan dalam listing 5.10.

Listing 5.10 publisher.php Mencantumkan Semua Dokumen sehingga Editor Dapat Memilih Mana yang Akan Ditampilkan di Situs Aktif

```
<?
include (“include_fns.php”);
$conn = db_connect();

$sql = “select * from stories order by modified desc”;
$result = mysql_query($sql, $conn);

echo “<H2>Editor admin</H2>”;
echo “<TABLE>”;
echo “<TR><TH>Headline</TH><TH>Last modified</TH></TR>”;
while ($story = mysql_fetch_array($result)) {
   echo “<TR><TD>”;
   echo $story[headline];
   echo “</TD><TD>”;
   echo date(“M d, H:i”, $story[modified]);
   echo “</TD><TD>”;
   if ($story[published]) {
   echo“[<A HREF=\”unpublish_story.php?story=$story[id]\”>unpublish</A>] “;
}
else {
 echo “[<A HREF=\”publish_story.php?story=$story[id]\”>publish</A>] “;
 echo “[<A HREF=\”delete_story.php?story=$story[id]\”>delete</A>] “;
}
 echo “[<A HREF=\”story.php?story=$story[id]\”>edit</A>] “;

 echo “</TD></TR>”;
}
echo “</TABLE>”;
?>
```

Skrip ini hanya boleh disediakan bagi orang-orang yang berwenang untuk menerbitkan cerita ke situs langsung. Dalam contoh aplikasi kami, ini akan menjadi editor situs, tetapi tidak ada kontrol akses pada skrip ini demi kesederhanaan. Namun, skrip ini harus dilindungi dalam situasi langsung. Ini sangat mirip dengan stories.php kecuali bahwa editor diberikan layar yang menunjukkan cerita untuk setiap penulis, bukan hanya miliknya sendiri. Pernyataan if memastikan bahwa opsi yang sesuai disajikan untuk setiap cerita. Cerita yang diterbitkan dapat dibatalkan penerbitannya, dan cerita yang tidak diterbitkan dapat diterbitkan atau dihapus. Ketiga tautan ini masing-masing dikirimkan ke unpublish_story.php, publisher_story.php, dan delete_story.php. Skrip publisher_story.php menggunakan kueri SQL berikut:

```
update stories set published = $now
where id = $story
```

Ini akan menandai cerita sebagai telah diterbitkan dan mengizinkannya untuk dilihat publik. Demikian pula, `unpublish_story.php` menggunakan kueri berikut untuk menandai cerita sebagai tidak diterbitkan dan menghentikannya agar tidak ditampilkan ke publik:

```
update stories set published = null
            where id = $story
```

Tautan edit muncul terlepas dari apakah cerita tersebut diterbitkan atau tidak, sehingga editor selalu dapat membuat perubahan. Ini berbeda dengan tingkat akses penulis, di mana mereka hanya dapat mengubah cerita sebelum diterbitkan.

Memperluas Proyek

Ada beberapa cara agar proyek ini dapat diperluas untuk membuat sistem manajemen konten yang lebih komprehensif:

- Anda dapat mengizinkan sekelompok pengguna untuk mengerjakan cerita bersama-sama (kolaborasi).
- Anda dapat menerapkan tata letak halaman yang lebih fleksibel sehingga editor dapat memposisikan teks dan gambar pada halaman.
- Pustaka gambar dapat dibuat sehingga gambar yang sering digunakan tidak terduplikasi, dan kata kunci pencarian ditetapkan ke gambar serta teks cerita.
- Anda juga dapat menambahkan fungsi pemeriksaan ejaan ke editor konten. Pemeriksaan dapat diterapkan menggunakan, misalnya, pustaka aspell.

BAB 6
MEMBANGUN LAYANAN EMAIL BERBASIS WEB

6.1 PENDAHULUAN

PHP memiliki dukungan IMAP dan POP3 yang sangat baik, tetapi disediakan melalui pustaka fungsi IMAP. Untuk menggunakan kode yang disajikan dalam bab ini, Anda harus menginstal pustaka IMAP. Anda dapat mengetahui apakah Anda sudah menginstalnya dengan melihat output fungsi phpinfo(). Jika belum, Anda harus mengunduh ekstensinya. Anda bisa mendapatkan versi terbaru melalui

FTP dari ftp://ftp.cac.washington.edu/imap/c-client.tar.Z

Di bawah UNIX, unduh sumbernya dan kompilasi untuk sistem operasi Anda. Setelah Anda melakukannya, salin `rfc822.h`, `mail.h`, dan `linkage.h` ke `/usr/local/include` atau direktori lain di jalur include Anda, jalankan skrip konfigurasi PHP, tambahkan direktif `--with-imap` ke parameter lain yang Anda gunakan, dan kompilasi ulang PHP. Ada dokumentasi tentang mengkompilasi versi Windows sendiri, tetapi jauh lebih rumit daripada mengkompilasi untuk UNIX. Jika Anda menggunakan platform Windows, ada alternatif yang lebih mudah. Anda dapat mengunduh versi PHP yang telah dikompilasi sebelumnya yang dikompilasi dengan berbagai ekstensi, termasuk ekstensi IMAP, dari

http://www.php4win.de

Satu hal menarik yang perlu diperhatikan adalah meskipun ini disebut fungsi IMAP, fungsi ini juga berfungsi sama baiknya dengan POP3 dan NNTP (*Network News Transfer Protocol*). Kita akan menggunakannya untuk IMAP dan POP3, tetapi aplikasi Warm Mail dapat dengan mudah diperluas untuk menggunakan NNTP.

Pustaka ini memiliki sejumlah besar fungsi, tetapi untuk mengimplementasikan fungsi tersebut dalam aplikasi ini, kami hanya akan menggunakan beberapa fungsi saja. Kami akan menjelaskan fungsi-fungsi ini saat kami menggunakannya, tetapi perlu diketahui bahwa masih banyak lagi fungsi lainnya. Lihat dokumentasi jika kebutuhan Anda berbeda dengan kebutuhan kami, atau jika Anda ingin menambahkan fitur tambahan ke aplikasi. Anda dapat membuat aplikasi email yang cukup berguna hanya dengan sebagian kecil fungsi bawaan. Ini berarti Anda hanya perlu membaca sebagian kecil dokumentasi. Fungsi IMAP yang kami gunakan dalam bab ini adalah

- **`imap_open()`**
- **`imap_close()`**
- **`imap_headers()`**
- **`imap_header()`**
- **`imap_fetchheader()`**

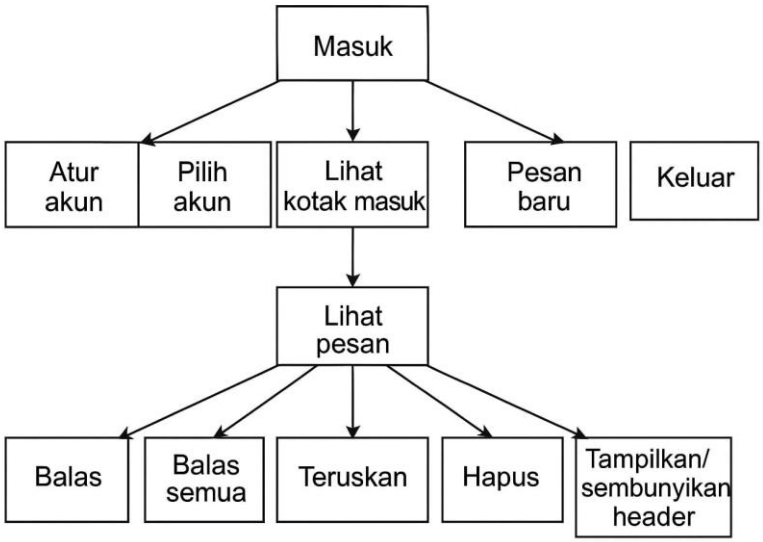
- `imap_body()`
- `imap_delete()`
- `imap_expunge()`

Agar pengguna dapat membaca emailnya, kita perlu mendapatkan detail server dan akunnya. Daripada mendapatkan detail ini dari pengguna setiap saat, kita akan menyiapkan basis data nama pengguna dan kata sandi untuk pengguna sehingga kita dapat menyimpan detailnya.

Sering kali orang memiliki lebih dari satu akun email (satu untuk rumah dan satu lagi untuk kantor, misalnya), dan kita harus mengizinkan mereka untuk terhubung ke salah satu akun mereka. Oleh karena itu, kita harus mengizinkan mereka memiliki beberapa set informasi akun dalam basis data. Kita harus mengizinkan pengguna untuk membaca, membalas, meneruskan, dan menghapus email yang ada, serta mengirim yang baru. Kita dapat melakukan semua bagian pembacaan menggunakan IMAP atau POP3, dan semua bagian pengiriman menggunakan SMTP dengan `mail()`.

Ikhtisar Solusi

Alur umum melalui sistem berbasis Web ini tidak akan jauh berbeda dari klien email lainnya. Diagram yang menunjukkan alur sistem dan modul ditunjukkan pada Gambar 6.1.



Gambar 6.1 Antarmuka untuk Warm Mail memberikan fungsionalitas tingkat kotak surat dan fungsionalitas tingkat pesan kepada pengguna.

Seperti yang dapat Anda lihat, pertama-tama kami akan meminta pengguna untuk masuk, lalu memberinya pilihan. Pengguna akan dapat menyiapkan akun email baru atau memilih salah satu akun yang sudah ada untuk digunakan. Pengguna juga akan dapat melihat email masuknya menanggapi, meneruskan, atau menghapusnya dan mengirim email baru. Kami juga akan memberinya opsi untuk melihat tajuk terperinci untuk pesan tertentu. Melihat tajuk lengkap dapat memberi tahu Anda banyak hal tentang pesan. Anda dapat melihat dari mesin mana email itu berasal—alat yang berguna untuk melacak spam. Anda dapat melihat mesin mana yang meneruskannya dan pada jam berapa email itu sampai ke setiap host berguna untuk menetapkan kesalahan atas pesan yang tertunda. Anda mungkin juga dapat

melihat klien email mana yang digunakan pengirim jika aplikasi menambahkan informasi opsional ke tajuk. Kami telah menggunakan arsitektur aplikasi yang sedikit berbeda untuk proyek ini. Alih-alih memiliki serangkaian skrip, satu untuk setiap modul, kami memiliki skrip yang sedikit lebih panjang, `index.php`, yang bekerja seperti perulangan peristiwa dari program yang digerakkan oleh GUI. Setiap tindakan yang kami lakukan di situs dengan menekan tombol akan membawa kami kembali ke `index.php`, tetapi dengan parameter yang berbeda. Bergantung pada parameternya, fungsi yang berbeda akan dipanggil untuk menunjukkan keluaran yang sesuai kepada pengguna. Fungsi-fungsi tersebut ada di pustaka fungsi, seperti biasa.

Arsitektur ini cocok untuk aplikasi kecil seperti ini. Arsitektur ini cocok untuk aplikasi yang sangat digerakkan oleh peristiwa, di mana tindakan pengguna memicu fungsionalitas. Menggunakan event handler tunggal tidak cocok untuk arsitektur yang lebih besar atau proyek yang dikerjakan oleh sebuah tim. Ringkasan file dalam proyek Warm Mail ditunjukkan pada Tabel 6.1.

Tabel 6.1 File Dalam Aplikasi Warm Mail

Nama	Tipe	Keterangan
`index.php`	Aplikasi	Skrip utama yang menjalankan seluruh aplikasi
`include_fns.php`	Fungsi	Koleksi file penyertaan untuk aplikasi
`data_valid_fns.php`	Fungsi	Kumpulan fungsi untuk memvalidasi data input
`db_fns.php`	Fungsi	Kumpulan fungsi untuk menghubungkan ke database email
`db_fns.php`	Fungsi	Koleksi fungsi terkait email untuk membuka kotak surat, membaca surat, dan sebagainya
`db_fns.php`	Fungsi	Kumpulan fungsi untuk mengeluarkan HTML
`user_auth_fns.php`	Fungsi	Kumpulan fungsi untuk mengautentikasi pengguna
`create_database.sql`	SQL	SQL untuk mengatur database `book_sc` dan mengatur pengguna

Mari kita lanjutkan dan lihat aplikasinya.

Menyiapkan Basis Data

Basis data untuk Warm Mail cukup sederhana karena kita tidak akan menyimpan email apa pun di dalamnya. Kita perlu menyimpan pengguna sistem. Untuk setiap pengguna, kita perlu menyimpan kolom berikut:

- *username* — (username) Nama pengguna pilihan mereka untuk Warm Mail
- *password* — (password) Kata sandi pilihan mereka untuk Warm Mail
- *address* —(address) Alamat email pilihan mereka, yang akan muncul di kolom Dari pada email yang mereka kirim dari sistem
- *displayname* —Nama yang “dapat dibaca manusia” yang ingin mereka tampilkan dalam email dari mereka kepada orang lain

Kita juga perlu menyimpan setiap akun yang ingin diperiksa pengguna dengan sistem. Untuk setiap akun, kita perlu menyimpan informasi berikut:

- *username* —Pengguna Warm Mail yang memiliki akun ini.
- *server*—Mesin tempat akun berada, misalnya, localhost atau mail.tangledweb.com.au.
- *port*—Port untuk terhubung saat menggunakan akun ini. Biasanya nilainya adalah 110 untuk server POP3 dan 143 untuk server IMAP.
- *type*—Protokol yang digunakan untuk terhubung ke server ini, baik 'POP3' atau 'IMAP'.
- *remoteuser*—Nama pengguna untuk terhubung ke server email.
- *remotepassword*—Kata sandi untuk terhubung ke server email.
- *accountid*—Kunci unik untuk mengidentifikasi akun.

Anda dapat menyiapkan basis data untuk aplikasi ini dengan menjalankan SQL yang ditunjukkan pada listing 6.1.

Listing 6.1 `create_database.sql`—SQL untuk Membuat Basis Data Email

```
create database mail;

use mail;

create table users
(
  username char(16) not null primary key,
  password char(16) not null,
  address char(100) not null,
  displayname char(100) not null
);

create table accounts
(
  username char(16) not null,
  server char(100) not null,
  port int not null,
  type char(4) not null,
  remoteuser char(50) not null,
  remotepassword char(50) not null,
  accountid int unsigned not null auto_increment primary key
);

grant select, insert, update, delete
on mail.*
to mail@localhost identified by 'password';
```

Ingatlah bahwa Anda dapat menjalankan SQL ini dengan mengetik

```
mysql -u root -p < create_database.sql
```

Anda perlu memberikan kata sandi root Anda. Anda harus mengubah kata sandi untuk pengguna email di `create_database.sql` dan di `db_fns.php` sebelum menjalankannya. Pada CD-ROM, kami juga telah menyediakan file SQL yang disebut `populate.sql`. Dalam aplikasi ini, kami tidak akan membuat proses pendaftaran atau administrasi pengguna. Anda dapat menambahkannya sendiri jika Anda ingin menggunakan perangkat lunak ini dalam skala yang lebih besar, tetapi jika Anda menginginkannya untuk penggunaan pribadi, Anda hanya perlu memasukkan diri Anda ke dalam basis data. Skrip populate.sql menyediakan proforma untuk melakukan ini, jadi masukkan detail Anda ke dalamnya dan jalankan untuk menyiapkan diri Anda sebagai pengguna.

6.2 ARSITEKTUR SKRIP

Seperti yang telah kami sebutkan sebelumnya, aplikasi ini menggunakan satu skrip untuk mengendalikan semuanya. Skrip ini disebut index.php. Skrip ini ditunjukkan pada listing 6.2. Skrip ini cukup panjang, tetapi kami akan membahasnya bagian demi bagian.

listing 6.2 `index.php`; Tulang Punggung Sistem Surat Hangat

```
<?
// This file is the main body of the Warm Mail application.
// It works basically as a state machine and shows users the
// output for the action they have chosen

//*****************************************************************
// Stage 1: pre-processing
// Do any required processing before page header is sent
// and decide what details to show on page headers
//*****************************************************************

    include ('include_fns.php');
    session_start();

    $buttons = array();

    //append to this string if anything processed before header has output
    $status = '';

// need to process log in or out requests before anything else
 if($username||$password)
 {
    if(login($username, $passwd))
    {
     $status .= "<p>Logged in successfully.<br><br><br><br><br><br>";
     $auth_user = $username;
     session_register("auth_user");
   }
   else
```

```
  {
    $status .= "<p>Sorry, we could not log you in with that
                username and password.<br><br><br><br><br><br>";
  }
 }
 if($action == 'log-out')
 {
  session_destroy();
  unset($action);
  unset($auth_user);
 }

//need to process choose, delete or store account before drawing header
switch ( $action )
{
 case 'delete-account' :
 {
  delete_account($auth_user, $account);
  break;
 }
 case 'store-settings' :
 {

      store_account_settings($auth_user, $HTTP_POST_VARS);
      break;
 }
 case 'select-account' :
 {
      // if have chosen a valid account, store it as a session variable
      if($account&&account_exists($auth_user, $account))
      {
        $selected_account = $account;
        session_register('selected_account');
       }
     }
 }
 // set the buttons that will be on the tool bar
 $buttons[0] = 'view-mailbox';
 $buttons[1] = 'new-message';
 $buttons[2] = 'account-setup';
 //only offer a log out button if logged in
 if(check_auth_user())
 {
   $buttons[4] = 'log-out';
 }

//*****************************************************************
```

```
// Stage 2: headers
// Send the HTML headers and menu bar appropriate to current action
//*********************************************************************

 if($action)
 {
  // display header with application name and description of page or action
  do_html_header($auth_user, "Warm Mail - ".
                  format_action($action), $selected_account);
 }
 else
 {
   // display header with just application name
   do_html_header($auth_user, "Warm Mail", $selected_account);
 }

 display_toolbar($buttons);
 //*********************************************************************
// Stage 3: body
// Depending on action, show appropriate main body content
//*********************************************************************

//display any text generated by functions called before header
 echo $status;

 if(!check_auth_user())
 {
    echo "<P>You need to log in";
    if($action&&$action!='log-out')
      echo " to go to ".format_action($action);
    echo ".<br><br>";
    display_login_form($action);
 }
 else
 {
  switch ( $action )
  {
   // if we have chosen to setup a new account, or have just added or
   // deleted an account, show account setup page
   case 'store-settings' :
   case 'account-setup' :
   case 'delete-account' :
   {
    display_account_setup($auth_user);
    break;
  }
  case 'send-message' :
```

```
  {
    if(send_message($to, $cc, $subject, $message))
      echo "<p>Message sent.<br><br><br><br><br><br>";
    else
      echo "<p>Could not send message.<br><br><br><br><br><br>";
    break;
  }
  case 'delete' :
  {
    delete_message($auth_user, $selected_account, $messageid);
    //note deliberately no 'break' - we will continue to the next case
  }
  case 'select-account' :
  case 'view-mailbox' :
  {
   // if mailbox just chosen, or view mailbox chosen, show mailbox
   display_list($auth_user, $selected_account);
   break;
  }
  case 'show-headers' :
  case 'hide-headers' :
case 'view-message' :
  {
    // if we have just picked a message from the list, or were looking at
    // a message and chose to hide or view headers, load a message
    $fullheaders = ($action=='show-headers');
    display_message($auth_user, $selected_account, $messageid,
          $fullheaders)
;
     break;
    }
    case 'reply-all' :
    {
      //set cc as old cc line
      if(!$imap)
          $imap = open_mailbox($auth_user, $selected_account);
      if($imap)
      {
        $header = imap_header($imap, $messageid);
        if($header->reply_toaddress)
          $to = $header->reply_toaddress;
        else
          $to = $header->fromaddress;
        $cc = $header->ccaddress;
        $subject = 'Re: '.$header->subject;
        $body = add_quoting(stripslashes(imap_body($imap, $messageid)));
       imap_close($imap);
```

```
      display_new_message_form($auth_user, $to, $cc, $subject, $body);
   }
   break;
}
case 'reply' :
{
    //set to address as reply-to or from of the current message
    if(!$imap)
        $imap = open_mailbox($auth_user, $selected_account);
    if($imap)
    {
        $header = imap_header($imap, $messageid);
        if($header->reply_toaddress)
           $to = $header->reply_toaddress;
        else
           $to = $header->fromaddress;
     $subject = 'Re: '.$header->subject;
     $body = add_quoting(stripslashes(imap_body($imap, $messageid)));
     imap_close($imap);
     display_new_message_form($auth_user, $to, $cc, $subject, $body);
   }
   break;
}
case 'forward' :
{
 //set message as quoted body of current message
 if(!$imap)
     $imap = open_mailbox($auth_user, $selected_account);
 if($imap)
 {
     $header = imap_header($imap, $messageid);
     $body = add_quoting(stripslashes(imap_body($imap, $messageid)));
     $subject = 'Fwd: '.$header->subject;
     imap_close($imap);

    display_new_message_form($auth_user, $to, $cc, $subject, $body);
  }
  break;
}
case 'new-message' :
{
       display_new_message_form($auth_user, $to, $cc, $subject, $body);
       break;
     }
    }
  }
```

```
//*****************************************************************
// Stage 4: footer
  do_html_footer();
//***************************************************************** ?>
```

Skrip ini menggunakan pendekatan penanganan peristiwa. Skrip ini berisi pengetahuan atau logika tentang fungsi mana yang perlu dipanggil untuk setiap peristiwa. Peristiwa dalam kasus ini dipicu oleh pengguna yang mengeklik berbagai tombol di situs, yang masing-masing memilih tindakan. Sebagian besar tombol diproduksi oleh fungsi `display_button()`, tetapi fungsi `display_form_button()` digunakan jika tombol tersebut merupakan tombol kirim. Kedua fungsi ini ada di `output_fns.php`. Semua ini melompat ke URL formulir

`index.php?action=log-out`

Nilai variabel `$action` saat `index.php` dipanggil menentukan pengendali peristiwa mana yang akan diaktifkan. Empat bagian utama skrip tersebut adalah sebagai berikut:

1. Kami melakukan beberapa pemrosesan yang harus dilakukan sebelum kami mengirim tajuk halaman ke peramban, seperti memulai sesi, menjalankan praproses apa pun untuk tindakan yang telah dipilih pengguna, dan memutuskan seperti apa tampilan tajuk tersebut.
2. Kami memproses dan mengirim tajuk dan bilah menu yang sesuai untuk tindakan yang telah dipilih pengguna.
3. Kami memilih badan skrip mana yang akan dijalankan, tergantung pada tindakan yang dipilih. Tindakan yang berbeda memicu panggilan fungsi yang berbeda.
4. Kami mengirim catatan kaki halaman.

Jika Anda melihat sekilas kode skrip tersebut, Anda akan melihat bahwa keempat bagian ini ditandai dengan komentar. Untuk memahami skrip ini sepenuhnya, mari kita bahas penggunaan situs ini secara aktual tindakan demi tindakan.

Log Masuk dan Keluar

Saat pengguna memuat halaman index.php, ia akan melihat output yang ditunjukkan pada Gambar 6.2.

Gambar 6.2 Layar Masuk Untuk Warm Mail Meminta Nama Pengguna Dan Kata Sandi Anda.

Ini adalah perilaku default untuk aplikasi. Tanpa `$action` yang dipilih, dan tanpa detail login yang diberikan, kami akan menjalankan bagian kode berikut. Pada tahap praproses, kami menjalankan kode berikut:

```
include ('include_fns.php');
session_start();
```

Baris ini memulai sesi yang akan digunakan untuk melacak variabel sesi `$auth_user` dan `$selected_account`, yang akan kita bahas nanti.

Untuk menghemat pekerjaan saat menyesuaikan antarmuka pengguna, tombol yang muncul pada bilah alat dikontrol oleh array. Kami mendeklarasikan array kosong,

```
$buttons = array();
```

dan mengatur tombol yang kami inginkan pada halaman:

```
$buttons[0] = 'view-mailbox';
$buttons[1] = 'new-message';
$buttons[2] = 'account-setup';
```

Untuk tahap header, kami mencetak header biasa:

```
do_html_header($auth_user, "Warm Mail", $selected_account);
 ...
display_toolbar($buttons);
```

Kode ini mencetak bilah judul dan tajuk, lalu bilah alat tombol yang dapat Anda lihat pada Gambar 6.2. Fungsi-fungsi ini dapat ditemukan di pustaka fungsi `output_fns.php`, tetapi karena Anda dapat dengan mudah melihat efeknya pada gambar, kami tidak akan membahasnya di sini. Sekarang kita masuk ke isi kode:

```
if(!check_auth_user())
{
 echo “<P>You need to log in”;
 if($action&&$action!=’log-out’)
       echo “ to go to “.format_action($action);
 echo “.<br><br>”;
 display_login_form($action);
}
```

Fungsi `check_auth_user()` berasal dari pustaka `user_auth_fns.php`. Kami telah menggunakan kode yang sangat mirip di beberapa proyek sebelumnya fungsi ini memeriksa apakah pengguna telah masuk. Jika tidak, seperti yang terjadi di sini, kami akan menunjukkan formulir masuk, yang dapat Anda lihat pada Gambar 5.2. Kami menggambar formulir ini dalam fungsi `display_login_form()` dari output_fns.php. Jika pengguna mengisi formulir dengan benar dan menekan tombol Masuk, ia akan melihat output yang ditunjukkan pada Gambar 5.3.

Gambar 6.3 Setelah Berhasil Login, Pengguna Dapat Mulai Menggunakan Aplikasi.

Pada eksekusi skrip ini, kita akan mengaktifkan beberapa bagian kode. Formulir login memiliki dua kolom, `$username` dan `$password`. Jika kolom ini telah diisi, segmen kode praproses berikut akan diaktifkan:

```
if($username||$password)
{
  if(login($username, $passwd))
  {
   $status .= "<p>Logged in successfully.<br><br><br><br><br><br>";
   $auth_user = $username;
   session_register("auth_user");
}
else
{
    $status .= "<p>Sorry, we could not log you in with that
                username and password.<br><br><br><br><br><br>";
  }
}
```

Seperti yang Anda lihat, kode tersebut memanggil fungsi `login()`, yang mirip dengan fungsi yang digunakan dalam Bab 1 dan 2. Jika semuanya berjalan lancar, kami mendaftarkan nama pengguna dalam variabel sesi `$auth_user`.

Selain mengatur tombol-tombol yang kita lihat ketika tidak login, kita tambahkan tombol lain untuk memungkinkan pengguna keluar lagi, sebagai berikut:

```
if(check_auth_user())
{
    $buttons[4] = 'log-out';
}
```

Anda dapat melihat tombol Log Out ini pada Gambar 6.3.

Pada tahap header, kita kembali menampilkan header dan tombol. Pada bagian body, kita menampilkan pesan status yang telah kita atur sebelumnya:

```
echo $status;
```

Setelah itu, tinggal mencetak footer dan menunggu untuk melihat apa yang akan dilakukan pengguna selanjutnya.

6.3 MENYIAPKAN AKUN

Saat pengguna pertama kali mulai menggunakan sistem Warm Mail, ia perlu menyiapkan beberapa akun email. Jika pengguna mengklik tombol Pengaturan Akun, ini akan menyetel variabel `$action` ke account-setup dan memanggil kembali skrip `index.php`. Pengguna kemudian akan melihat output yang ditunjukkan pada Gambar 6.4.

Gambar 6.4 Pengguna Perlu Menyiapkan Detail Akun Emailnya Sebelum Ia Dapat Membaca Emailnya.

Lihat kembali skrip pada Listing 6.2. Kali ini karena nilai `$action`, kita mendapatkan perilaku yang berbeda. Kita mendapatkan header yang sedikit berbeda, seperti berikut:

```
do_html_header($auth_user, "Warm Mail - ".
format_action($action), $selected_account);
```

Yang lebih penting lagi, kita mendapatkan tubuh yang berbeda, sebagai berikut:

```
case 'store-settings' :
case 'account-setup' :
case 'delete-account' :
{
 display_account_setup($auth_user);
 break;
}
```

Ini adalah pola yang umum: Setiap perintah memanggil suatu fungsi. Dalam kasus ini, kita memanggil fungsi `display_account_setup()`. Kode untuk fungsi ini ditunjukkan dalam Listing 6.3.

Listing 6.3 Fungsi `Display_Account_Setup()` Dari `Output_Fns.Php`—Fungsi Untuk Mendapatkan Dan Menampilkan Rincian Akun

```
function display_account_setup($auth_user)
 {
 //display empty 'new account' form
```

```
display_account_form($auth_user);
$list = get_accounts($auth_user);

// display each stored account
foreach($list as $key => $account)
{

// display form for each accounts details.
// note that we are going to send the password for all accounts in the
HTML
// this is not really a very good idea
display_account_form($auth_user, $account['accountid'],
        $account['server'],
         $account['remoteuser'],
         $account['remotepassword'], $account['type'],
         $account['port']);
  }
}
```

Saat kita memanggil fungsi ini, fungsi ini akan menampilkan formulir kosong untuk menambahkan akun baru, diikuti oleh formulir yang dapat diedit yang berisi setiap akun email pengguna saat ini.

Fungsi `display_account_form()` akan menampilkan formulir yang dapat kita lihat pada Gambar 6.4. Anda dapat melihat bahwa kita menggunakannya dalam dua cara berbeda di sini: Kita menggunakannya tanpa parameter untuk menampilkan formulir kosong, dan kita menggunakannya dengan serangkaian parameter lengkap untuk menampilkan rekaman yang sudah ada. Fungsi ini ada di pustaka `output_fns.php`; fungsi ini hanya menampilkan HTML sehingga kita tidak akan membahasnya di sini. Fungsi yang mengambil semua akun yang sudah ada adalah `get_accounts()`, dari pustaka `mail_fns.php`. Fungsi ini ditampilkan dalam listing 6.4.

Listing 6.4 Fungsi `get_accounts()` dari `mail_fns.php`; Fungsi untuk Mengambil Semua Detail Akun untuk Pengguna Tertentu

```
function get_accounts($auth_user)
{
 $list = array();
 if(db_connect())
 {
      $query = "select * from accounts where username = '$auth_user'";
      $result = mysql_query($query);
            if($result)
  {
   while($settings = mysql_fetch_array($result))
        array_push( $list, $settings);
  }
```

```
  else
   return false;
 }
 return $list;
}
```

Seperti yang dapat Anda lihat, fungsi ini terhubung ke basis data, mengambil semua akun untuk pengguna tertentu, dan mengembalikannya sebagai array.

6.4 MEMBUAT AKUN BARU

Jika pengguna mengisi formulir akun dan mengklik tombol Simpan Perubahan, tindakan pengaturan penyimpanan akan diaktifkan. Mari kita lihat kode penanganan peristiwa untuk ini dari `index.php`. Pada tahap praproses, kami menjalankan kode berikut:

```
case 'store-settings' :
{
 store_account_settings($auth_user, $HTTP_POST_VARS);
 break;
}
```

Fungsi `store_account_settings()` menulis detail akun baru ke dalam basis data. Kode untuk fungsi ini ditunjukkan pada listing 6.5.

Listing 6.5 Fungsi `store_account_settings()` dari `mail_fns.php`—Fungsi untuk Menyimpan Detail Akun Baru untuk Pengguna

```
function store_account_settings($auth_user, $settings)
{
  if(!filled_out($settings))
  {
    echo "All fields must be filled in.  Try again.<br><br>";
    return false;
  }
  else
  {
    if($settings['account']>0)
      $query = "update accounts  set server = '$settings[server]',
                port = $settings[port], type = '$settings[type]',
                remoteuser = '$settings[remoteuser]',
                remotepassword = '$settings[remotepassword]'
               where accountid = $settings[account]
                and username = '$auth_user'";
 else
    $query = "insert into accounts values ('$auth_user',
                '$settings[server]', $settings[port],
                '$settings[type]', '$settings[remoteuser]',
```

```
                    '$settings[remotepassword]', NULL)";
 if(db_connect() && mysql_query($query))
 {
      return true;
 }
 else
 {
   echo "could not store changes.<br><br><br><br><br><br>";
   return false;
  }
 }
}
```

Seperti yang Anda lihat, dua pilihan dalam fungsi ini sesuai dengan memasukkan akun baru atau memperbarui akun yang sudah ada. Fungsi ini menjalankan kueri yang sesuai untuk menyimpan detail akun. Setelah menyimpan detail akun, kita kembali ke `index.php`, ke tahap isi utama:

```
case 'store-settings' :
case 'account-setup' :
case 'delete-account' :
{
  display_account_setup($auth_user);
  break;
}
```

Seperti yang dapat Anda lihat, kami kemudian menjalankan fungsi `display_account_setup()` seperti sebelumnya untuk mencantumkan detail akun pengguna. Akun yang baru ditambahkan sekarang akan disertakan.

Memodifikasi Akun Yang Ada

Proses untuk memodifikasi akun yang ada sangat mirip. Pengguna dapat mengubah detail akun dan mengeklik tombol Simpan Perubahan. Sekali lagi ini akan memicu tindakan store-settings, tetapi kali ini akan memperbarui detail akun alih-alih memasukkannya.

Menghapus Akun

Untuk menghapus akun, pengguna dapat mengeklik tombol Hapus Akun yang ditampilkan di bawah setiap daftar akun. Ini mengaktifkan tindakan hapus-akun. Di bagian praproses skrip index.php, kami akan menjalankan kode berikut:

```
case 'delete-account' :
{
 delete_account($auth_user, $account);
 break;
}
```

Kode ini memanggil fungsi `delete_account()`. Kode untuk fungsi ini ditampilkan dalam Listing 6.6. Penghapusan akun perlu ditangani sebelum header karena pilihan akun mana yang akan digunakan ada di dalam header. Daftar akun perlu diperbarui sebelum ini dapat dibuat dengan benar.

Listing 6.6 Fungsi `delete_account()` dari `mail_fns.php`—Fungsi untuk Menghapus Rincian Akun Tunggal

```
function delete_account($auth_user, $accountid)
{
 //delete one of this user's accounts from the DB

 $query = "delete from accounts where
             accountid='$accountid' and
             username = '$auth_user'";
 if(db_connect())
 {
     $result = mysql_query($query);
 }
 return $result
}
```

Setelah eksekusi kembali ke `index.php`, tahap body akan menjalankan kode berikut:

```
case 'store-settings' :
case 'account-setup' :
case 'delete-account' :
{
  display_account_setup($auth_user);
  break;
}
```

Anda akan mengenali ini sebagai kode yang sama yang kita jalankan sebelumnya ini hanya menampilkan daftar akun pengguna.

6.5 MEMBACA EMAIL

Setelah pengguna menyiapkan beberapa akun, kita dapat beralih ke permainan utama: menghubungkan ke akun-akun ini dan membaca email.

Memilih Akun

Kita perlu memilih salah satu akun pengguna untuk membaca email. Akun yang saat ini dipilih disimpan dalam variabel sesi `$selected_account`. Jika pengguna memiliki satu akun yang terdaftar dalam sistem, akun tersebut akan dipilih secara otomatis saat ia masuk, seperti berikut:

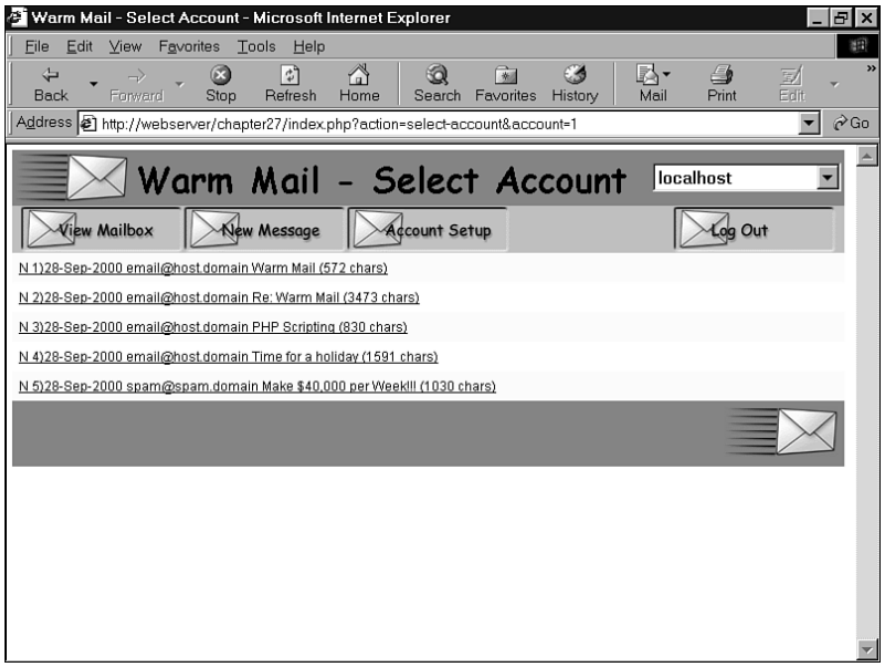
```
if(number_of_accounts($auth_user)==1)
{
  $accounts = get_account_list($auth_user);
  $selected_account = $accounts[0];
  session_register("selected_account");
}
```

Fungsi number_of_accounts(), dari mail_fns.php, digunakan untuk mengetahui apakah pengguna memiliki lebih dari satu akun. Fungsi get_account_list() mengambil array nama akun pengguna. Dalam kasus ini, hanya ada satu, sehingga kita dapat mengaksesnya sebagai nilai 0 dari array tersebut. Fungsi number_of_accounts() ditampilkan dalam Listing 6.7.

Listing 6.7 Fungsi number_of_accounts() dari mail_fns.php—Fungsi untuk Mengetahui Berapa Banyak Akun yang Telah Didaftarkan Pengguna

```
function number_of_accounts($auth_user)
{
    // get the number of accounts that belong to this user
$query = "select count(*) from accounts where username = '$auth_user'";

    if(db_connect())
    {
     $result = mysql_query($query);
       if($result)
            return mysql_result($result, 0, 0);
    }
    return 0;
 }
```

Fungsi get_account_list() mirip dengan fungsi get_accounts() yang telah kita bahas sebelumnya, kecuali fungsi ini hanya mengambil nama akun. Jika pengguna memiliki beberapa akun yang terdaftar, ia harus memilih satu untuk digunakan. Dalam kasus ini, header akan berisi SELECT yang mencantumkan kotak surat yang tersedia. Memilih yang sesuai akan secara otomatis menampilkan kotak surat untuk akun tersebut. Anda dapat melihatnya pada Gambar 6.5.



Gambar 6.5 Setelah Akun Localhost Dipilih Dari Kotak Select, Email Dari Akun Tersebut Diunduh Dan Ditampilkan.

Opsi SELECT ini dibuat dalam fungsi `do_html_header()` dari `output_fns.php`, seperti yang ditunjukkan dalam fragmen kode berikut:

```
<?
      // include the account select box only if the user has more than one
account
      if(number_of_accounts($auth_user)>1)
   {
       echo "<form target='index.php?action=open-mailbox' method=post>";
       echo '<td bgcolor = "#ff6600" align = right valign = middle>';
       display_account_select($auth_user, $selected_account);
       echo '</td>';
       echo "</form>";
   }
 ?>
```

Kami biasanya menghindari pembahasan HTML yang digunakan dalam contoh-contoh dalam buku ini, tetapi HTML yang dihasilkan oleh fungsi `display_account_select()` patut dicoba. Bergantung pada akun yang dimiliki pengguna saat ini, `display_account_select()` akan menghasilkan HTML seperti ini:

```
<select onchange=window.location=this.options[selectedIndex].value name=account>
  <option value = 0 selected>
    Choose Account</a>
  <option value = 'index.php?action=select-account&account=10'>
```

```
    mail.domain.com
 </option>
 <option value = 'index.php?action=select-account&account=11'>
    mail.server.com
 </option>
 <option value = 'index.php?action=select-account&account=9'>
      localhost
   </option>
 </select>
```

Sebagian besar kode ini hanyalah elemen select HTML, tetapi juga menyertakan sedikit JavaScript. Dengan cara yang sama seperti PHP dapat menghasilkan HTML, kode ini juga dapat digunakan untuk menghasilkan skrip sisi klien. Setiap kali terjadi peristiwa perubahan pada elemen ini, JavaScript akan menyetel `window.location` ke nilai opsi. Jika pengguna Anda memilih opsi pertama dalam select, `window.location` akan disetel ke `'index.php?action=select-account&account=10'`. Ini akan mengakibatkan URL ini dimuat. Jelas, jika pengguna memiliki browser yang tidak mendukung JavaScript atau menonaktifkan JavaScript, kode ini tidak akan berpengaruh.

Fungsi `display_account_select()`, dari `output_fns.php`, digunakan untuk mendapatkan daftar akun yang tersedia dan menampilkan SELECT. Fungsi ini juga menggunakan fungsi `get_account_list()` yang telah kita bahas sebelumnya. Memilih salah satu opsi dalam SELECT akan mengaktifkan acara `select_account`. Jika Anda melihat URL pada Gambar 6.5, Anda dapat melihat ini ditambahkan di akhir URL, bersama dengan ID akun yang dipilih. Ini memiliki dua efek. Pertama, dalam tahap praproses index.php, akun yang dipilih akan disimpan dalam variabel sesi `$selected_account`, seperti berikut:

```
case 'select-account' :
{
// if have chosen a valid account, store it as a session variable
if($account&&account_exists($auth_user, $account))
{
  $selected_account = $account;
  session_register('selected_account');
}
}
```

Kedua, ketika tahap body skrip dieksekusi, kode berikut akan dieksekusi:

```
case 'select-account' :
case 'view-mailbox' :
{
    // if mailbox just chosen, or view mailbox chosen, show mailbox
    display_list($auth_user, $selected_account);
    break;
```

```
}
```

Seperti yang dapat Anda lihat, kami melakukan tindakan yang sama di sini seolah-olah pengguna telah memilih opsi Lihat Kotak Surat. Kami akan membahasnya selanjutnya.

Melihat Isi Kotak Surat

Isi kotak surat dapat dilihat dengan fungsi `display_list()`. Fungsi ini akan menampilkan daftar semua pesan di kotak surat. Kode untuk fungsi ini ditampilkan dalam listing 6.8.

Listing 6.8 Fungsi `display_list()` dari `output_fns.php`—Fungsi untuk Menampilkan Semua Pesan Kotak Surat

```
function display_list($auth_user, $accountid)
{
    // show the list of messages in this mailbox

    global $table_width;

    if(!$accountid)
    {

    echo “No mailbox selected<br><br><br><br><br><br>.”;
 }
 else
 {

    $imap = open_mailbox($auth_user, $accountid);

    if($imap)
    {
      echo “<table width = $table_width cellspacing = 0
                    cellpadding = 6  border = 0>”;

     $headers = imap_headers($imap);
     // we could reformat this data, or get other details using
     // imap_fetchheaders, but this is not a bad summary so we just echo
    each

    $messages = sizeof($headers);
    for($i = 0; $i<$messages; $i++)
    {
      echo “<tr><td bgcolor = ‘“;
      if($i%2)
            echo “#ffffff”;
      else
            echo “#ffffcc”;
```

```
      echo "'><a href ='index.php?action=view
                  message&messageid=".($i+1)."'>";
      echo $headers[$i];
      echo "</a></td></tr>\n";
     }
     echo "</table>";
  }
  else
  {
    $account = get_account_settings($auth_user, $accountid);
    echo "could not open mail box
   ".$account['server'].".<br><br><br><br>";
  }
 }
}
```

Dalam fungsi ini, kita sebenarnya mulai menggunakan fungsi IMAP PHP. Dua bagian utama dari fungsi ini adalah membuka kotak surat dan membaca tajuk pesan. Kita membuka kotak surat untuk akun pengguna dengan memanggil fungsi `open_mailbox()` yang telah kita tulis di `mail_fns.php`. Fungsi ini ditunjukkan pada listing 6.9.

Listing 6.9 Fungsi `open_mailbox()` dari `mail_fns.php`—Fungsi Ini Menghubungkan ke Kotak Surat Pengguna

```
function open_mailbox($auth_user, $accountid)
{
 // select mailbox if there is only one
 if(number_of_accounts($auth_user)==1)
 {
   $accounts = get_account_list($auth_user);
   $selected_account = $accounts[0];
   session_register("selected_account");
   $accountid = $selected_account;
 }

 // connect to the POP3 or IMAP server the user has selected

   $settings = get_account_settings($auth_user, $accountid);
   if(!sizeof($settings)) return 0;
   $mailbox = "{".$settings[server];
   if($settings[type]=='POP3')
         $mailbox .= '/pop3';
   $mailbox .= ":".$settings[port]."}INBOX";

// suppress warning, remember to check return value
@  $imap = imap_open($mailbox, $settings[remoteuser],
           $settings[remotepassword]);
```

```
        return  $imap;
}
```

Kami sebenarnya membuka kotak surat dengan fungsi imap_open(). Fungsi ini memiliki prototipe berikut:

```
int imap_open (string mailbox, string username, string password [,int flags])
```

Parameter yang perlu Anda berikan kepadanya adalah sebagai berikut:

- kotak surat—String ini harus berisi nama server dan nama kotak surat, dan secara opsional nomor port dan protokol. Format string ini adalah

  ```
  {namahost/protokol:port}namakotak
  ```

  Jika protokol tidak ditentukan, protokol tersebut akan menggunakan IMAP secara default. Dalam kode yang telah kami tulis, Anda dapat melihat bahwa kami menentukan POP3 jika pengguna telah menentukan protokol tersebut untuk akun tertentu.
  Misalnya, untuk membaca email dari komputer lokal menggunakan port default, kami akan menggunakan nama kotak surat berikut untuk IMAP

  ```
  {localhost:143}INBOX
  dan yang ini untuk POP3
  {localhost/pop3:110}INBOX
  ```

- *Username*—Nama pengguna untuk akun tersebut
- *password*—Kata sandi untuk akun tersebut

Anda juga dapat meneruskannya dengan tanda opsional untuk menentukan opsi seperti “buka kotak surat dalam mode baca-saja”.

Satu hal yang perlu diperhatikan adalah bahwa kami telah menyusun string kotak surat sepotong demi sepotong dengan operator penggabungan sebelum meneruskannya ke `imap_open()`. Anda harus berhati-hati dalam menyusun string ini karena string yang berisi {$ dapat menyebabkan masalah di PHP 4. Pemanggilan fungsi ini mengembalikan aliran IMAP jika kotak surat dapat dibuka, dan false jika tidak dapat dibuka. Setelah selesai dengan aliran IMAP, Anda dapat menutupnya menggunakan `imap_close(imap_stream)`. Dalam fungsi kami, aliran IMAP diteruskan kembali ke program utama. Kami kemudian menggunakan fungsi `imap_headers()` untuk mendapatkan tajuk email untuk ditampilkan:

```
$headers = imap_headers($imap);
```

Fungsi ini mengembalikan informasi tajuk untuk semua pesan email di kotak surat yang telah kami hubungkan. Informasi dikembalikan sebagai array, satu baris per pesan. Kami belum

memformat ini. Ia hanya menampilkan satu baris per pesan, jadi Anda dapat melihat seperti apa tampilan outputnya dari Gambar 6.5. Anda dapat memperoleh informasi lebih lanjut tentang tajuk email menggunakan `imap_header()` yang membingungkan dan memiliki nama yang mirip. Dalam kasus ini, fungsi `imap_headers()` memberi kita cukup detail untuk tujuan kita.

Membaca Pesan Email

Kita telah menyiapkan setiap pesan dalam fungsi `display_list()` sebelumnya untuk ditautkan ke pesan email tertentu. Setiap tautan berbentuk

```
index.php?action=view-message&messageid=6
```

MessageID adalah nomor urut yang digunakan dalam header yang kita peroleh sebelumnya. Perhatikan bahwa pesan IMAP diberi nomor dari 1, bukan 0. Jika pengguna mengklik salah satu tautan ini, ia akan melihat output seperti yang ditunjukkan pada Gambar 6.6.

Gambar 6.6 Dengan Menggunakan Tindakan View-Message, Kita Akan Melihat Pesan Tertentu; Dalam Kasus Ini, Pesan Tersebut Adalah Spam.

Saat kita memasukkan parameter ini ke dalam skrip index.php, kita akan menjalankan kode berikut:

```
case ‘show-headers’ :
case ‘hide-headers’ :
case ‘view-message’ :
{
  // if we have just picked a message from the list, or were looking at
  // a message and chose to hide or view headers, load a message
```

```
  $fullheaders = ($action==’show-headers’);
  display_message($auth_user, $selected_account, $messageid,
     $fullheaders)
  break;
}
```

Anda akan melihat bahwa kami sedang memeriksa nilai $action agar sama dengan ‘show-headers’. Dalam kasus ini, nilainya akan salah, dan $fullheaders akan ditetapkan sama dengan salah. Kami akan melihat tindakan ‘show-headers’ sebentar lagi.
Garis `$fullheaders = ($action==’show-headers’);` bisa saja ditulis lebih rinci—dan mungkin lebih jelas—sebagai

```
if($action==’show-headers’)
      $fullheaders = true;
else
      $fullheaders = false;
```

Selanjutnya, kita panggil fungsi `display_message()`. Sebagian besar fungsi ini menghasilkan HTML biasa, jadi kita tidak akan membahasnya di sini. Fungsi ini memanggil fungsi `retrieve_message()` untuk mendapatkan pesan yang sesuai dari kotak surat:

```
$message = retrieve_message($auth_user, $accountid, $messageid, $fullheaders);
```

Fungsi `retrieve_message()` ada di pustaka `mail_fns.php`. Anda dapat melihat kodenya di Listing 6.10.

Listing 6.10 Fungsi `retrieve_message()` dari `mail_fns.php`—Fungsi ini mengambil satu pesan tertentu dari kotak surat

```
function retrieve_message($auth_user, $accountid, $messageid, $fullheaders)
{
  $message = array();
if(!($auth_user && $messageid && $accountid))
return false;

$imap = open_mailbox($auth_user, $accountid);

if(!$imap)
  return false;

$header = imap_header($imap, $messageid);

if(!$header)
  return false;
```

```
$message['body'] = imap_body($imap, $messageid);
if(!$message['body'])
  $message['body'] = '[This message has no body]\n\n\n\n\n\n';

if($fullheaders)
  $message['fullheaders'] = imap_fetchheader($imap, $messageid);
else
  $message['fullheaders'] = '';
```

Sekali lagi, kami menggunakan open_mailbox() untuk membuka kotak surat pengguna. Namun, kali ini, kami mencari pesan tertentu. Dengan menggunakan pustaka fungsi ini, kami mengunduh tajuk pesan dan isi pesan secara terpisah.

Tiga fungsi IMAP yang kami gunakan di sini adalah imap_header(), imap_fetchheader(), dan imap_body(). Perhatikan bahwa kedua fungsi tajuk tersebut berbeda dari imap_headers(), yang kami gunakan sebelumnya. Nama keduanya agak membingungkan. Singkatnya,

- imap_headers()—Mengembalikan ringkasan tajuk untuk semua pesan di kotak surat. Mengembalikannya sebagai array dengan satu elemen per pesan.
- imap_header()—Mengembalikan tajuk untuk satu pesan tertentu dalam bentuk objek.
- imap_fetchheader()—Mengembalikan tajuk untuk satu pesan tertentu dalam bentuk string.

Dalam kasus ini, kami menggunakan imap_header() untuk mengisi kolom header tertentu dan imap_fetchheader() untuk menunjukkan header lengkap kepada pengguna jika diminta. (Kita akan membahasnya nanti.) Kami menggunakan imap_header() dan imap_body() untuk membuat array yang berisi semua elemen pesan yang kami minati. Kami memanggil imap_header() sebagai berikut:

```
$header = imap_header($imap, $messageid);
```

Kemudian, kami dapat mengekstrak setiap kolom yang kami perlukan dari objek:

```
$message['subject'] = $header->subject;
```

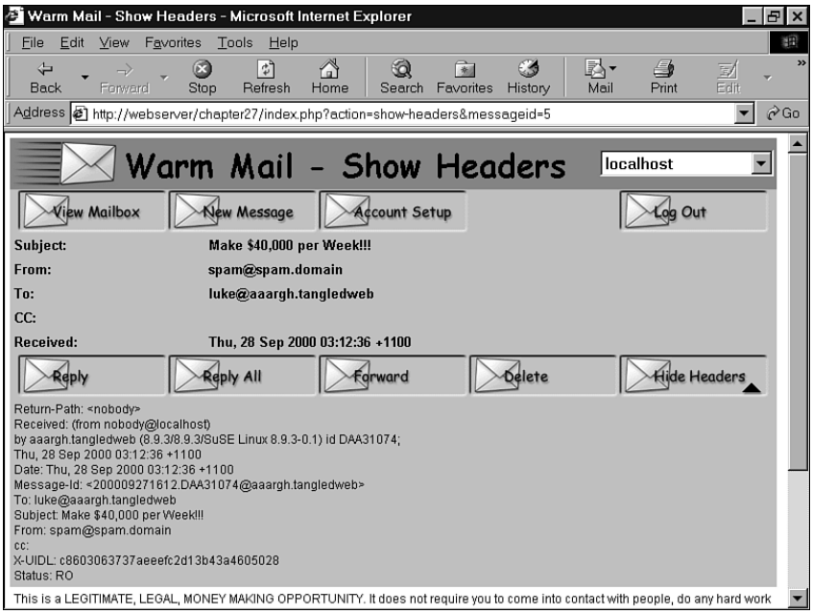
Kami memanggil imap_body() untuk menambahkan isi pesan ke array kami sebagai berikut:

```
$message['body'] = imap_body($imap, $messageid);
```

Akhirnya kita menutup kotak surat dengan imap_close() dan mengembalikan array yang telah kita buat. Fungsi display_message() kemudian dapat menampilkan kolom pesan dalam bentuk yang dapat Anda lihat pada Gambar 6.6.

Melihat Header Pesan

Seperti yang dapat Anda lihat pada Gambar 6.6, terdapat tombol Show Headers. Ini mengaktifkan opsi show-headers, yang menambahkan header pesan lengkap ke tampilan pesan. Jika pengguna mengklik tombol ini, ia akan melihat output yang mirip dengan yang ditunjukkan pada Gambar 6.7.



Gambar 6.7 Penggunaan Show-Headers Untuk Melihat Header Lengkap Pesan Ini Akan Membantu Pengguna Melacak Sumber Spam.

Seperti yang mungkin Anda perhatikan, penanganan peristiwa untuk view-message juga mencakup show-headers (dan padanannya hide-headers). Jika opsi ini dipilih, kita melakukan hal yang sama seperti sebelumnya. Namun dalam retrieve_message(), kita juga mengambil teks lengkap dari header, sebagai berikut:

```
if($fullheaders)
$message['fullheaders'] = imap_fetchheader($imap, $messageid);
```

Kita kemudian dapat menampilkan header ini kepada pengguna.

Menghapus Email

Jika pengguna mengklik tombol Hapus pada email tertentu, ia akan mengaktifkan tindakan "hapus". Ini akan menjalankan kode berikut dari index.php:

```
case 'delete' :
{
  delete_message($auth_user, $selected_account, $messageid);
  //note deliberately no 'break' - we will continue to the next case
}
case 'select-account' :
```

```
case ‘view-mailbox’ :
{
  // if mailbox just chosen, or view mailbox chosen, show mailbox
  display_list($auth_user, $selected_account);
  break;
}
```

Seperti yang dapat Anda lihat, pesan dihapus menggunakan fungsi `delete_message()`, lalu kotak surat yang dihasilkan ditampilkan seperti yang telah dibahas sebelumnya. Kode untuk fungsi `delete_message()` ditampilkan dalam listing 6.11.

Listing 6.11 Fungsi `delete_message()` dari `mail_fns.php`—Fungsi Ini Menghapus Satu Pesan Tertentu dari Kotak Surat

```
function delete_message($auth_user, $accountid, $message_id)
{
  // delete a single message from the server

  $imap = open_mailbox($auth_user, $accountid);
  if($imap)
  {
     imap_delete($imap, $message_id);
     imap_expunge($imap);
     imap_close($imap);
     return true;
   }
   return false;
 }
```

Seperti yang Anda lihat, fungsi ini menggunakan sejumlah fungsi IMAP. Fungsi yang baru adalah `imap_delete()` dan `imap_expunge()`. Perhatikan bahwa `imap_delete()` hanya menandai pesan untuk dihapus. Anda dapat menandai pesan sebanyak yang Anda suka. Panggilan ke `imap_expunge()` sebenarnya menghapus pesan.

6.6 MENGIRIM EMAIL

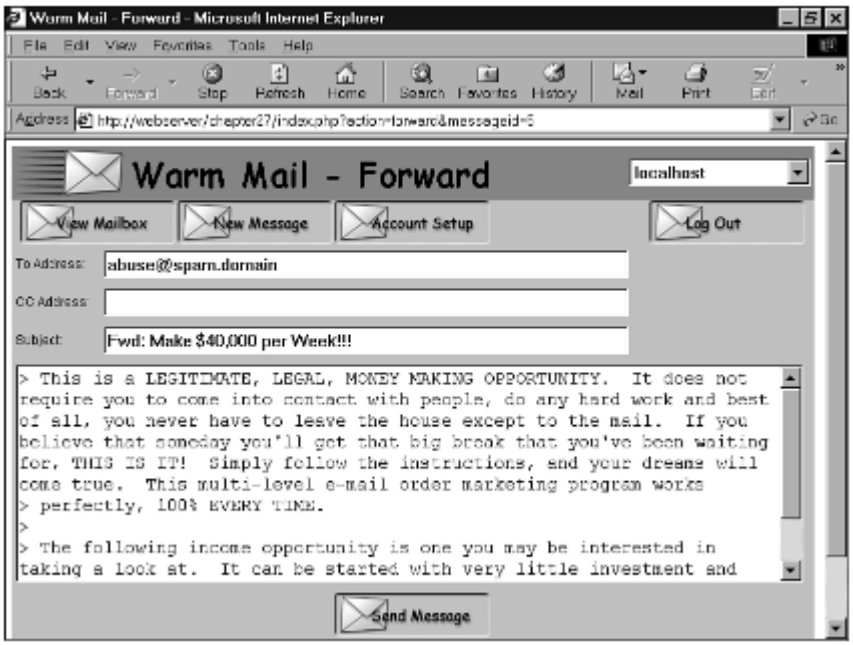
Akhirnya kita sampai pada tahap mengirim email. Ada beberapa cara untuk melakukannya dari skrip ini: Pengguna dapat mengirim pesan baru, membalas, atau meneruskan email. Mari kita lihat cara kerjanya.

Mengirim Pesan Baru

Pengguna dapat memilih opsi ini dengan mengklik tombol Pesan Baru. Ini mengaktifkan tindakan “`new-message`”, yang mengeksekusi kode berikut di index.php:

```
case ‘new-message’ :
{
      display_new_message_form($auth_user, $to, $cc, $subject, $body);
```

Formulir pesan baru hanyalah formulir untuk mengirim email. Anda dapat melihat seperti apa bentuknya pada Gambar 6.8. Gambar ini sebenarnya menunjukkan penerusan email, bukan email baru, tetapi formulirnya sama. Kita akan melihat penerusan dan balasan selanjutnya.



Gambar 6.8 Dengan menggunakan penerusan email, kita dapat melaporkan pengirim spam.

Mengklik tombol Kirim Pesan akan memanggil tindakan "`send-message`", yang akan mengeksekusi kode berikut:

```
case 'send-message' :
{
  if(send_message($to, $cc, $subject, $message))
      echo "<p>Message sent.<br><br><br><br><br><br>";
  else
      echo "<p>Could not send message.<br><br><br><br><br><br>";
  break;
}
```

Kode ini memanggil fungsi `send_message()`, yang sebenarnya mengirimkan email. Fungsi ini ditunjukkan pada Listing 6.12.

Listing 6.12 Fungsi `send_message()` dari `mail_fns.php`—Fungsi ini Mengirim Pesan yang Telah Diketik Pengguna

```
function send_message($to, $cc, $subject, $message)
{
```

```
      //send one email via PHP

      global $auth_user;
      if (!db_connect())
      {
            return false;
      }
      $query = "select address from users where username='$auth_user'";

      $result = mysql_query($query);
      if (!$result)
      {
            return false;
      }
      else if (mysql_num_rows($result)==0)
      {
            return false;
      }
      else
      {
            $other = "From: ".mysql_result($result, 0, "address")."\r\ncc:
      $cc";
            if (mail($to, $subject, $message, $other))
                  return true;
      else
      {
            return false;
   }
  }
 }
```

Seperti yang dapat Anda lihat, fungsi ini menggunakan `mail()` untuk mengirim email. Namun, pertama-tama, fungsi ini memuat alamat email pengguna dari basis data untuk digunakan di kolom From pada email.

Membalas atau Meneruskan Email

Fungsi Reply, Reply All, dan Forward semuanya mengirim email dengan cara yang sama seperti New Message. Perbedaan cara kerjanya adalah fungsi ini mengisi bagian-bagian formulir pesan baru sebelum menunjukkannya kepada pengguna. Lihat kembali Gambar 6.8. Pesan yang kita teruskan telah diberi indentasi dengan simbol >, dan baris Subject diawali dengan To. Demikian pula, opsi Reply dan Reply All akan mengisi penerima, baris subject, dan pesan yang diberi indentasi.

Kode untuk melakukan ini diaktifkan di bagian body `index.php`, sebagai berikut:

```
case 'reply-all' :
```

```
{
  //set cc as old cc line
  if(!$imap)
      $imap = open_mailbox($auth_user, $selected_account);
  if($imap)
  {
    $header = imap_header($imap, $messageid);
    if($header->reply_toaddress)
     $to = $header->reply_toaddress;
    else
     $to = $header->fromaddress;
    $cc = $header->ccaddress;
    $subject = 'Re: '.$header->subject;
    $body = add_quoting(stripslashes(imap_body($imap, $messageid)));
    imap_close($imap);

    display_new_message_form($auth_user, $to, $cc, $subject, $body);
   }
   break;
 }
 case 'reply' :
 {
    //set to address as reply-to or from of the current message
    if(!$imap)
    $imap = open_mailbox($auth_user, $selected_account);
    if($imap)
    {
    $header = imap_header($imap, $messageid);
     if($header->reply_toaddress)
      $to = $header->reply_toaddress;
     else
      $to = $header->fromaddress;
     $subject = 'Re: '.$header->subject;
     $body = add_quoting(stripslashes(imap_body($imap, $messageid)));
     imap_close($imap);

     display_new_message_form($auth_user, $to, $cc, $subject, $body);
 }

     break;        //note deliberately no 'break'
 }
 case 'forward' :
 {
   //set message as quoted body of current message
   if(!$imap)
     $imap = open_mailbox($auth_user, $selected_account);
   if($imap)
```

```
    {
      $header = imap_header($imap, $messageid);
      $body = add_quoting(stripslashes(imap_body($imap, $messageid)));
      $subject = 'Fwd: '.$header->subject;
      imap_close($imap);

      display_new_message_form($auth_user, $to, $cc, $subject, $body);
    }
    break;
 }
```

Anda dapat melihat bahwa masing-masing opsi ini menyiapkan tajuk yang sesuai, menerapkan pemformatan sebagaimana diperlukan, dan memanggil fungsi `display_new_message_form()` untuk menyiapkan formulir. Itulah rangkaian lengkap fungsi untuk pembaca email Web kita.

BAB 7
MEMBANGUN PENGELOLA MILIS

Setelah Anda membangun basis pelanggan situs Web Anda, alangkah baiknya jika Anda dapat tetap berhubungan dengan mereka dengan mengirimkan buletin. Dalam bab ini, kami akan menerapkan antarmuka untuk pengelola milis (atau MLM). Beberapa MLM memungkinkan setiap pelanggan untuk mengirim pesan ke pelanggan lain. Program kami akan menjadi sistem buletin, di mana hanya administrator milis yang dapat mengirim pesan. Kami akan menyebut sistem kami Pyramid-MLM.

Aplikasi kami akan memungkinkan administrator membuat beberapa milis dan mengirim buletin ke masing-masing milis tersebut secara terpisah. Aplikasi ini akan menggunakan unggahan file untuk memungkinkan administrator mengunggah versi teks dan HTML dari buletin yang telah mereka buat secara offline. Ini berarti administrator dapat menggunakan perangkat lunak apa pun yang mereka sukai untuk membuat buletin. Pengguna akan dapat berlangganan ke salah satu milis di situs kami dan memilih apakah akan menerima buletin dalam bentuk teks atau HTML.

7.1 PENDAHULUAN

Kami ingin membangun sistem penulisan dan pengiriman buletin daring. Sistem ini harus memungkinkan berbagai buletin dibuat dan dikirim ke pengguna, dan memungkinkan pengguna untuk berlangganan satu atau banyak buletin.

Secara khusus, persyaratan untuk sistem ini adalah

- Administrator harus dapat menyiapkan dan mengubah milis.
- Administrator harus dapat mengirim buletin teks dan HTML ke semua pelanggan milis tunggal.
- Pengguna harus dapat mendaftar untuk menggunakan situs, dan memasukkan serta mengubah detail mereka.
- Pengguna harus dapat berlangganan ke salah satu milis di situs.
- Pengguna harus dapat berhenti berlangganan dari milis yang mereka langgani.
- Pengguna harus dapat menyimpan preferensi mereka untuk buletin berformat HTML atau teks biasa.
- Demi alasan keamanan, pengguna tidak boleh dapat mengirim email ke milis atau melihat alamat email satu sama lain.
- Pengguna dan administrator harus dapat melihat informasi tentang milis.
- Pengguna dan administrator harus dapat melihat buletin terdahulu yang telah dikirim ke milis (arsip).

Komponen Solusi

Ada sejumlah komponen yang kita perlukan untuk memenuhi persyaratan. Komponen utama adalah menyiapkan basis data daftar, pelanggan, dan buletin yang diarsipkan; mengunggah buletin yang telah dibuat secara offline; dan mengirim email dengan lampiran.

7.2 MENYIAPKAN BASIS DATA DAFTAR DAN PELANGGAN

Kita akan melacak nama pengguna dan kata sandi setiap pengguna sistem, serta daftar daftar yang telah mereka langgani. Kita juga akan menyimpan preferensi setiap pengguna untuk menerima email teks atau HTML, sehingga kita dapat mengirim versi buletin yang sesuai kepada pengguna.

Administrator akan menjadi pengguna khusus dengan kemampuan untuk membuat milis baru dan mengirim buletin ke milis tersebut. Fungsi yang bagus untuk sistem seperti ini adalah arsip buletin sebelumnya. Pelanggan mungkin tidak menyimpan kiriman sebelumnya, tetapi mungkin ingin mencari sesuatu. Arsip juga dapat berfungsi sebagai alat pemasaran untuk buletin karena calon pelanggan dapat melihat seperti apa buletin tersebut.

Menyiapkan basis data ini di MySQL dan antarmukanya di PHP tidak akan memiliki hal baru atau sulit di dalamnya.

Unggah Berkas

Kita memerlukan antarmuka untuk memungkinkan administrator mengirim buletin, seperti yang disebutkan sebelumnya. Yang belum kita bahas adalah bagaimana administrator akan membuat buletin tersebut. Kita dapat menyediakan formulir tempat mereka dapat mengetik atau menempelkan konten buletin. Namun, sistem kita akan lebih mudah digunakan jika administrator dapat membuat buletin di editor favorit mereka, lalu mengunggah berkas ke server Web. Ini juga akan memudahkan administrator untuk menambahkan gambar ke buletin HTML.

Kita perlu menggunakan formulir yang sedikit lebih rumit daripada yang pernah kita gunakan sebelumnya. Kita akan meminta administrator untuk mengunggah versi teks dan HTML buletin, beserta gambar sebaris yang masuk ke HTML.

Setelah buletin diunggah, kita perlu membuat antarmuka agar administrator dapat melihat pratinjau buletin sebelum mengirimnya. Dengan cara ini, administrator dapat memastikan bahwa semua berkas telah diunggah dengan benar.

Mengirim Email dengan Lampiran

Untuk proyek ini, kami ingin dapat mengirim buletin teks biasa atau versi HTML "canggih" kepada pengguna, sesuai dengan preferensi mereka. Untuk mengirim file HTML dengan gambar tertanam, kami perlu menemukan cara untuk mengirim lampiran. Fungsi mail() sederhana PHP tidak mendukung pengiriman lampiran dengan mudah. Sebagai gantinya, kami akan menggunakan kelas Mail HTML MIME yang luar biasa yang dibuat oleh Richard Heyes. Kelas ini dapat menangani lampiran HTML, dan akan secara otomatis melampirkan gambar apa pun yang terdapat dalam file HTML.

Anda bebas menggunakan skrip ini dalam pekerjaan Anda sendiri. Skrip ini dirilis sebagai Postcard-Ware. Jika Anda menggunakannya, kirimkan kartu pos kepada penulis. Alamatnya ada di situs webnya.

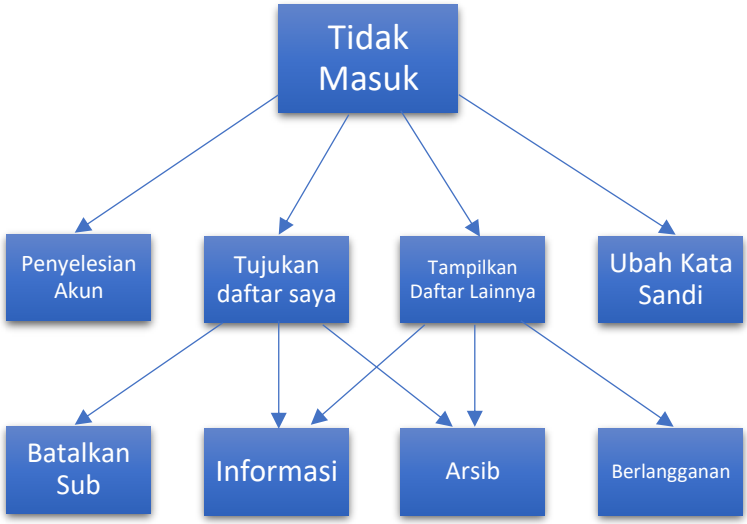
Ringkasan Solusi

Kami telah memulai lagi dengan menggambar serangkaian diagram alur sistem untuk menunjukkan jalur yang mungkin diambil pengguna melalui sistem. Dalam kasus ini, kami telah menggambar tiga diagram untuk mewakili tiga rangkaian interaksi berbeda yang dapat

dilakukan pengguna dengan sistem. Pengguna memiliki tindakan yang diizinkan berbeda saat mereka tidak masuk, saat mereka masuk sebagai pengguna biasa, dan saat mereka masuk sebagai administrator. Tindakan ini ditunjukkan pada Gambar 7.1, 28.2, dan 28.3, secara berurutan.

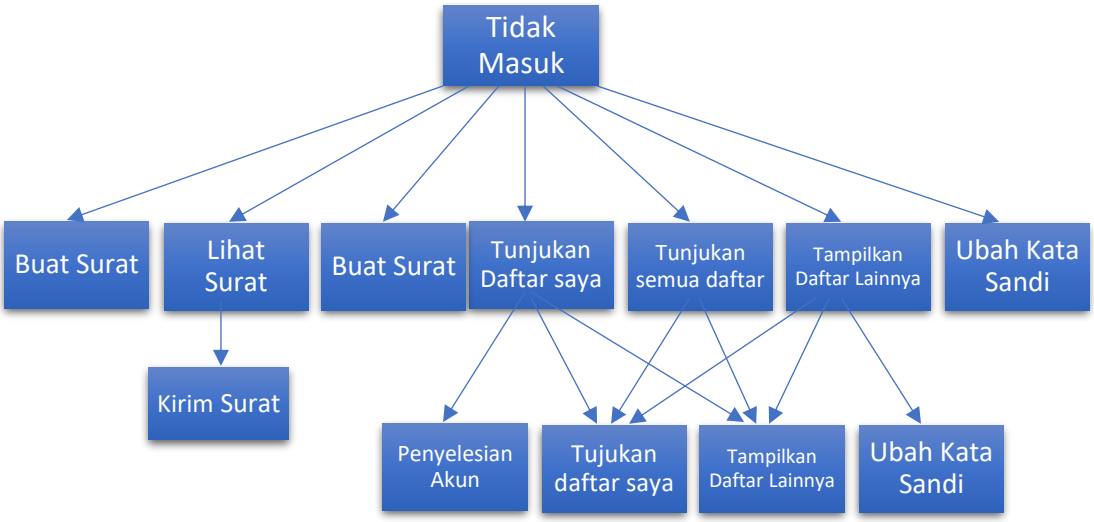
Gambar 7.1 Pengguna hanya dapat memilih sejumlah tindakan terbatas saat ia tidak masuk.

Pada Gambar 7.1 Anda dapat melihat tindakan yang dapat dilakukan oleh pengguna yang tidak login. Seperti yang Anda lihat, pengguna dapat login (jika sudah memiliki akun), membuat akun (jika belum memilikinya), atau melihat milis yang tersedia untuk pendaftaran (sebagai taktik pemasaran).



Gambar 7.2 Setelah masuk, pengguna dapat mengubah preferensi mereka melalui berbagai pilihan.

Gambar 7.2 menunjukkan tindakan yang dapat dilakukan pengguna setelah masuk. Pengguna dapat mengubah pengaturan akunnya (alamat email dan preferensi), mengubah kata sandinya, dan mengubah daftar langganannya.



Gambar 7.3 Administrator memiliki tindakan tambahan yang tersedia bagi mereka.

Gambar 7.3 menunjukkan tindakan yang tersedia jika administrator telah masuk. Seperti yang dapat Anda lihat, administrator memiliki sebagian besar fungsi yang tersedia bagi pengguna, dan beberapa opsi tambahan. Ia juga dapat membuat milis baru, membuat pesan baru untuk milis dengan mengunggah file, dan melihat pratinjau pesan sebelum mengirimnya.

Karena kami telah menggunakan pendekatan berbasis peristiwa lagi, tulang punggung aplikasi tersebut terdapat dalam satu berkas, index.php, yang memanggil sekumpulan pustaka fungsi. Gambaran umum berkas dalam aplikasi ini ditunjukkan pada Tabel 7.1.

Tabel 7.1 Berkas dalam Aplikasi Mailing List Manager

Nama file	Jenis	Keterangan
`indeks.php`	Aplikasi	Skrip utama yang menjalankan seluruh aplikasi.
`include_fns.php`	Fungsi	Kumpulan berkas penyertaan untuk aplikasi ini.
`data_valid_fns.php`	Fungsi	Kumpulan fungsi untuk memvalidasi data masukan.
`db_fns.php`	Fungsi	Kumpulan fungsi untuk menghubungkan ke basis data mlm.
`mlm_fns.php`	Fungsi	Kumpulan fungsi khusus untuk aplikasi ini.
`output_fns.php`	Fungsi	Kumpulan fungsi untuk mengeluarkan HTML.
`oupload.php`	Komponen	Skrip yang mengelola komponen unggah berkas dari peran administrator. Dipisahkan untuk memudahkan keamanan.
`user_auth_fns.php`	Fungsi	Kumpulan fungsi untuk mengautentikasi pengguna.
`create_database.sql`	SQL	SQL untuk menyiapkan basis data mlm dan menyiapkan pengguna Web dan pengguna administratif.

Kami akan mengerjakan implementasi proyek ini, dimulai dengan basis data tempat kami akan menyimpan informasi pelanggan dan daftar.

Menyiapkan Basis Data

Untuk aplikasi ini, kami perlu menyimpan detail tentang

- *Daftar:* Milis yang tersedia untuk berlangganan.
- *Pelanggan:* Pengguna sistem dan preferensi mereka.
- *Sub_lists:* Catatan pengguna mana yang berlangganan daftar mana (hubungan banyak ke banyak)
- *Mail:* Catatan pesan email yang telah dikirim.
- *Images:* Karena kita ingin dapat mengirim pesan email yang terdiri dari beberapa file (yaitu, teks dan HTML ditambah sejumlah gambar), kita juga perlu melacak gambar mana yang disertakan dengan setiap email.

SQL yang telah kita tulis untuk membuat basis data ini ditunjukkan pada Listing 7.1.

Listing 7.1 `create_database.sql`—SQL untuk Membuat Basis Data MLM

```
create database mlm;

use mlm;

create table lists
(
  listid int auto_increment not null primary key,
  listname char(20) not null,
  blurb varchar(255)
);

create table subscribers
(
  email char(100) not null primary key,
  realname char(100) not null,
  mimetype char(1) not null,
  password char(16) not null,
  admin tinyint not null
);

# stores a relationship between a subscriber and a list
create table sub_lists
(
  email char(100) not null,
  listid int not null
);

create table mail
(
  mailid int auto_increment not null primary key,
  email char(100) not null,
  subject char(100) not null,
```

```
  listid int not null,
  status char(10) not null,
  sent datetime,
  modified timestamp
);

#stores the images that go with a particular mail
create table images
(
  mailid int not null,
  path char(100) not null,
  mimetype char(100) not null
);

grant select, insert, update, delete
on mlm.*
to mlm@localhost identified by 'password';
insert into subscribers values
('admin@localhost', 'Administrative User', 'H', password('admin'), 1);
```

Ingatlah bahwa Anda dapat menjalankan SQL ini dengan mengetik

```
mysql -u root -p < create_database.sql
```

Anda perlu memberikan kata sandi root Anda. (Tentu saja, Anda dapat menjalankan skrip ini melalui pengguna MySQL mana pun dengan hak istimewa yang sesuai; kami hanya menggunakan root di sini demi kesederhanaan.)

Anda harus mengubah kata sandi untuk pengguna mlm dan administrator dalam skrip Anda sebelum menjalankannya. Beberapa kolom dalam basis data ini memerlukan sedikit penjelasan lebih lanjut, jadi mari kita bahas secara singkat. Tabel daftar berisi `listid` dan `listname`. Tabel ini juga berisi uraian singkat, yang merupakan deskripsi tentang isi daftar tersebut.

Tabel `subscribers` berisi alamat email (`email`) dan nama (`realname`) pelanggan. Tabel ini juga menyimpan kata sandi mereka dan tanda (`admin`) untuk menunjukkan apakah pengguna ini adalah administrator atau bukan. Kami juga akan menyimpan jenis email yang ingin mereka terima dalam `mimetype`. Jenis ini dapat berupa `H` untuk HTML atau `T` untuk teks.

Tabel `sublists` berisi alamat email (`email`) dari tabel `subscribers` dan `listid` dari tabel `lists`. Tabel surat berisi informasi tentang setiap pesan email yang dikirim melalui sistem. Tabel ini menyimpan `id` unik (`mailid`), alamat asal surat (`email`), baris subjek email (`subject`), dan `listid` dari daftar yang telah atau akan dikirimi surat. Teks atau HTML sebenarnya dari pesan tersebut bisa berupa file besar, jadi kami akan menyimpan arsip pesan sebenarnya di luar basis data. Kami juga akan melacak beberapa informasi status umum:

apakah pesan telah dikirim (`status`), kapan dikirim (`sent`), dan stempel waktu untuk menunjukkan kapan catatan ini terakhir diubah (`modified`).

Terakhir, kami menggunakan tabel gambar untuk melacak gambar apa pun yang terkait dengan pesan HTML. Sekali lagi, gambar-gambar ini bisa berukuran besar, jadi kami akan menyimpannya di luar basis data demi efisiensi. Sebagai gantinya, kami akan melacak `mailid` yang terkait dengannya, jalur ke lokasi tempat gambar sebenarnya disimpan, dan tipe MIME gambar (`mimetype`), misalnya, `image/gif`. SQL yang ditunjukkan sebelumnya juga menyiapkan pengguna untuk PHP agar dapat terhubung, dan pengguna administratif untuk sistem.

7.3 ARSITEKTUR SKRIP

Seperti pada proyek terakhir, kami telah menggunakan pendekatan berbasis peristiwa untuk proyek ini. Tulang punggung aplikasi ada di file index.php. Skrip ini memiliki empat segmen utama, yaitu

1. Prapemrosesan: Lakukan pemrosesan apa pun yang harus dilakukan sebelum header dapat dikirim.
2. Siapkan dan kirim header: Buat dan kirim awal halaman HTML.
3. Lakukan tindakan: Tanggapi peristiwa yang telah diteruskan. Seperti pada contoh terakhir kita, peristiwa tersebut terdapat dalam variabel $action.
4. Kirim footer.

Hampir semua pemrosesan aplikasi dilakukan dalam berkas ini. Aplikasi ini juga menggunakan pustaka fungsi yang tercantum dalam Tabel 7.1, seperti yang disebutkan sebelumnya. Daftar lengkap skrip index.php ditampilkan dalam Listing 7.2.

Listing 7.2 `index.php`—Berkas Aplikasi Utama untuk Pyramid-MLM

```
<?

/*********************************************************************
* Section 1 : pre-processing
*********************************************************************/

   include ('include_fns.php');
   session_start();

   $buttons = array();

   //append to this string if anything processed before header has output
   $status = '';

   // need to process log in or out requests before anything else
   if($email&&$password)
```

```
    {
        $login = login($email, $password);

        if($login == 'admin')
    {
        $status .= "<p><b>".get_real_name($email)."</b> logged in"
                    ." successfully as <b>Administrator</b><br><br><br><br><br>";
        $admin_user = $email;
        session_register("admin_user");
    }
    else if($login == 'normal')
    {
        $status .= "<p><b>".get_real_name($email)."</b> logged in"
                ." successfully.<br><br>";
        $normal_user = $email;
        session_register("normal_user");
    }
    else
    {
        $status .= "<p>Sorry, we could not log you in with that
        email address and password.<br>";
    }
}
if($action == 'log-out')
{
    session_destroy();
    unset($action);
    unset($normal_user);
    unset($admin_user);
}

/*************************************************************************
* Section 2: set up and display headers
*************************************************************************/

// set the buttons that will be on the tool bar
if(check_normal_user())
{
    // if a normal user

    $buttons[0] = 'change-password';
```

```
    $buttons[1] = 'account-settings';
    $buttons[2] = 'show-my-lists';
    $buttons[3] = 'show-other-lists';
    $buttons[4] = 'log-out';
}
else if(check_admin_user())
{

    // if an administrator
    $buttons[0] = 'change-password';
    $buttons[1] = 'create-list';
    $buttons[2] = 'create-mail';
    $buttons[3] = 'view-mail';
    $buttons[4] = 'log-out';
    $buttons[5] = 'show-all-lists';
    $buttons[6] = 'show-my-lists';
    $buttons[7] = 'show-other-lists';
}
else
{
    // if not logged in at all
    $buttons[0] = 'new-account';
    $buttons[1] = 'show-all-lists';
    $buttons[4] = 'log-in';
}

if($action)
{
    // display header with application name and description of page or action
    do_html_header("Pyramid-MLM - ".format_action($action));
}
else
{
    // display header with just application name
    do_html_header("Pyramid-MLM");
}

display_toolbar($buttons);

//display any text generated by functions called before header echo $status;
```

```
/*********************************************************************
* Section 3: perform action
*********************************************************************/

// only these actions can be done if not logged in
switch ( $action )
{
    case 'new-account' :
    {
        unset($normal_user);
        unset($admin_user);
        display_account_form($normal_user, $admin_user);
        break;
    }
    case 'store-account' :
    {
        if (store_account($normal_user, $admin_user, $HTTP_POST_VARS))
            $action = '';
        if(!check_logged_in())
            display_login_form($action);
        break;
    }
    case 'log-in' :
    case '':
    {
        if(!check_logged_in())
            display_login_form($action);
        break;
    }
    case 'show-all-lists' :
    {
        display_items("All Lists", get_all_lists(), 'information',
            'show-archive','');
        break;
    }
    case 'show-archive' :
    {
        display_items("Archive For ".get_list_name($id),
            get_archive($id), 'view-html', 'view-text', '');
        break;
    }
```

```
    case 'information' :
    {
        display_information($id);
        break;
    }
}

    //all other actions require user to be logged in

    if(check_logged_in())
    {
        switch ( $action )
        {
            case 'account-settings' :
            {
                display_account_form($normal_user, $admin_user, get_email(),
                get_real_name(get_email()), get_mimetype(get_email()));
                break;
            }
            case 'show-other-lists' :
            {
                display_items("Unsubscribed Lists",
                get_unsubscribed_lists(get_email()), 'information',
                'show-archive', 'subscribe');
                break;
            }
            case 'subscribe' :
            {
                subscribe(get_email(), $id);
                display_items("Subscribed Lists", get_subscribed_lists(get_email()),
                'information', 'show-archive', 'unsubscribe');
                break;
            }
            case 'unsubscribe' :
            {
                unsubscribe(get_email(), $id);
                display_items("Subscribed Lists", get_subscribed_lists(get_email()),
            'information', 'show-archive', 'unsubscribe');
            break;
        }
        case '':
```

```
        case 'show-my-lists' :
        {
            display_items("Subscribed Lists", get_subscribed_lists(get_email()),
                'information', 'show-archive', 'unsubscribe');
            break;
        }
        case 'change-password' :
        {
            display_password_form();
            break;
        }
        case 'store-change-password' :
        {
            if(change_password(get_email(), $old_passwd,
            $new_passwd, $new_passwd2))
            {
                echo "<p>OK: Password changed.<br><br><br><br><br><br>";
            }
            else
            {
                echo "<p>Sorry, your password could not be changed.";
                display_password_form();
            }
            break;
        }
    }
}
// The following actions may only be performed by an admin user
if(check_admin_user())
{
    switch ( $action )
    {
        case 'create-mail' :
        {
            display_mail_form(get_email());
            break;
        }
        case 'create-list' :
        {
            display_list_form(get_email());
            break;
```

```
        }
        case 'store-list' :
        {
            if(store_list($admin_user, $HTTP_POST_VARS))
            {
                echo "<p>New list added<br>";
                display_items("All Lists", get_all_lists(), 'information',
                        'show-archive','');
            }
            else
                echo "<p>List could not be stored, please try "
                        ."again.<br><br><br><br><br>";
            break;
        }
        case 'send' :
        {
            send($id, $admin_user);
            break;
        }
        case 'view-mail' :
        {
            display_items("Unsent Mail", get_unsent_mail(get_email()),
                'preview-html', 'preview-text', 'send');
            break;
        }
    }
}

/*********************************************************************
* Section 4: display footer
*********************************************************************/
    do_html_footer();
?>
```

Anda dapat melihat empat segmen kode yang ditandai dengan jelas dalam daftar ini. Pada tahap praproses, kami menyiapkan sesi dan memproses tindakan apa pun yang perlu dilakukan sebelum header dapat dikirim. Dalam hal ini, ini termasuk masuk dan keluar.

Pada tahap header, kami menyiapkan tombol menu yang akan dilihat pengguna, dan menampilkan header yang sesuai menggunakan fungsi `do_html_header()` dari `output_fns.php`. Fungsi ini hanya menampilkan bilah header dan menu, jadi kami tidak akan membahasnya di sini. Di bagian utama skrip, kami menanggapi tindakan yang telah dipilih

pengguna. Tindakan ini dibagi menjadi tiga subset: tindakan yang dapat diambil jika tidak masuk, tindakan yang dapat diambil oleh pengguna normal, dan tindakan yang dapat diambil oleh pengguna administratif. Kami memeriksa untuk melihat apakah akses ke dua set tindakan terakhir diizinkan menggunakan fungsi `check_logged_in()` dan `check_admin_user()`. Fungsi-fungsi ini terletak di pustaka fungsi `user_auth_fns.php`. Kode untuk fungsi-fungsi tersebut, dan untuk fungsi `check_normal_user()` ditunjukkan pada Listing 7.3.

Listing 7.3 Fungsi-fungsi dari `user_auth_fns.php`—Fungsi-fungsi ini memeriksa apakah pengguna telah masuk atau belum, dan pada level berapa

```
function check_normal_user()
// see if somebody is logged in and notify them if not
{
  global $normal_user;
  if (session_is_registered("normal_user"))
    return true;
  else
    return false;
  }

function check_admin_user()
// see if somebody is logged in and notify them if not
{
`   global $admin_user;

  if (session_is_registered("admin_user"))
    return true;
  else
    return false;
}

function check_logged_in()
{
  return ( check_normal_user() || check_admin_user() );
}
```

Seperti yang dapat Anda lihat, fungsi-fungsi ini menggunakan variabel sesi `$normal_user` dan `$admin_user` untuk memeriksa apakah pengguna telah masuk. Kita akan membahas tentang pengaturan variabel sesi ini sebentar lagi. Di bagian akhir skrip, kita mengirim footer HTML menggunakan fungsi `do_html_footer()` dari `output_fns.php`.

Mari kita lihat sekilas ikhtisar tindakan yang mungkin dilakukan dalam sistem. Tindakan-tindakan ini ditunjukkan dalam Tabel 7.2

Tabel 7.2 Tindakan yang Mungkin Dilakukan dalam Aplikasi Mailing List Manager

Tindakan	Dapat Digunakan Oleh	Keterangan
`log-in`	Siapa pun	Memberikan pengguna formulir login
`log-out`	Siapa pun	Mengakhiri sesi
`new-account`	Siapa pun	Membuat akun baru untuk pengguna
`store-account`	Siapa pun	Menyimpan detail akun
`show-all-lists`	Siapa pun	Menampilkan daftar milis yang tersedia
`show-archive`	Siapa pun	Menampilkan surat berita yang diarsipkan untuk daftar tertentu
`Information`	Siapa pun	Menampilkan informasi dasar tentang daftar tertentu
`account-settings`	Pengguna yang masuk	Menampilkan pengaturan akun pengguna
`show-other-lists`	Pengguna yang masuk	Menampilkan milis yang tidak berlangganan oleh pengguna
`show-my-lists`	Pengguna yang masuk	Menampilkan milis tempat pengguna berlangganan
`Subscribe`	Pengguna yang masuk	Melanggankan pengguna ke daftar tertentu
`unsubscribe`	Pengguna yang masuk	Membatalkan langganan pengguna dari daftar tertentu
`change-password`	Pengguna yang masuk	Menampilkan formulir perubahan kata sandi
`store-change-password`	Pengguna yang masuk	Memperbarui kata sandi pengguna di database kata sandi
`create-mail`	Administrator	Menampilkan formulir untuk memungkinkan pengunggahan buletin
`create-list`	Administrator	Menampilkan formulir untuk memungkinkan milis baru dibuat
`store-list`	Administrator	Menyimpan detail milis dalam database
`view-mail`	Administrator	Menampilkan buletin yang telah diunggah tetapi belum dikirim
`send`	Administrator	Mengirim buletin ke pelanggan

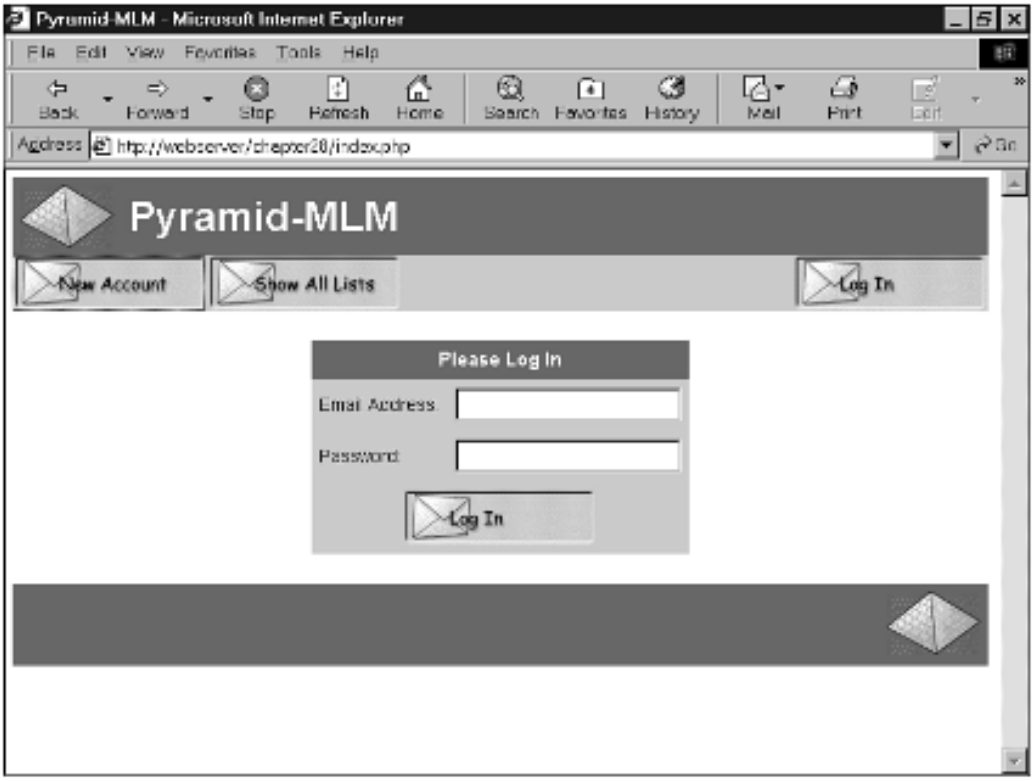
Satu kelalaian yang mencolok dari tabel ini adalah opsi seperti `store-mail`, yaitu, tindakan yang benar-benar mengunggah buletin yang dimasukkan melalui `create-mail` oleh administrator. Fungsionalitas tunggal ini sebenarnya ada di file yang berbeda, `upload.php`. Kami menaruhnya di file terpisah karena ini memudahkan kami, para programmer, untuk melacak masalah keamanan.

Kami akan membahas penerapan tindakan ini dalam tiga kelompok yang tercantum dalam Tabel 7.2, yaitu, tindakan untuk orang yang tidak `login`; tindakan untuk pengguna yang `login`; dan tindakan untuk administrator.

Mengimplementasikan Login

Ketika pengguna baru datang ke situs kami, ada tiga hal yang kami ingin mereka lakukan. Pertama, lihat apa yang kami tawarkan; kedua, daftar dengan kami; dan ketiga, login. Kami akan melihat masing-masing hal ini secara bergantian.

Pada Gambar 7.4, Anda dapat melihat layar yang kami tampilkan kepada pengguna saat mereka pertama kali datang ke situs kami.



Gambar 7.4 Saat masuk, pengguna dapat membuat akun baru, melihat daftar yang tersedia, atau sekadar masuk.

Kita akan membahas pembuatan akun baru dan masuk sekarang, dan kembali melihat detail daftar di bagian "Menerapkan Fungsi Pengguna" dan "Menerapkan Fungsi Administratif" nanti.

Membuat Akun Baru

Jika pengguna memilih opsi menu Akun Baru, tindakan akun baru akan diaktifkan. Kode berikut akan diaktifkan di index.php:

```
case 'new-account' :
{
  unset($normal_user);
  unset($admin_user);
  display_account_form($normal_user, $admin_user);
  break;
}
```

Kode ini secara efektif mengeluarkan pengguna jika mereka saat ini sedang masuk, dan menampilkan formulir detail akun seperti yang ditunjukkan pada Gambar 7.5.

Gambar 7.5 Formulir pembuatan akun baru memungkinkan pengguna untuk memasukkan detail mereka.

Formulir ini dibuat oleh fungsi `display_account_form()` dari pustaka `output_fns.php`. Fungsi ini digunakan di sini dan dalam tindakan pengaturan akun untuk menampilkan formulir guna memungkinkan pengguna menyiapkan akun. Jika fungsi ini dipanggil dari tindakan pengaturan akun, formulir akan diisi dengan data akun pengguna yang ada. Di sini formulir kosong, siap untuk detail akun baru. Karena fungsi ini hanya menghasilkan HTML, kita tidak akan membahasnya di sini.

Tombol kirim pada formulir ini memanggil tindakan `store-account`. Kode untuk tindakan ini adalah sebagai berikut:

```
case 'store-account' :
{
  if (store_account($normal_user, $admin_user, $HTTP_POST_VARS))
    $action = '';
  if(!check_logged_in())
    display_login_form($action);
  break;
}
```

Fungsi `store_account()` menulis detail akun ke dalam basis data. Kode untuk fungsi ini ditunjukkan dalam Listing 7.4.

Listing 7.3 Fungsi `store_account()` dari mlm_fns.php—Fungsi-fungsi ini memeriksa apakah pengguna sudah login atau belum, dan pada level berapa

```
// add a new subscriber to the database, or let a user modify their data
function store_account($normal_user, $admin_user, $details)
{
```

```
if(!filled_out($details))
{
  echo "All fields must be filled in. Try again.<br><br>";
  return false;
}
else
{
  if(subscriber_exists($details['email']))
  {
    //check logged in as the user they are trying to change
    if(get_email()==$details['email'])
    {
      $query = "update subscribers set realname = '$details[realname]',
                mimetype = '$details[mimetype]'
                where email = '" . $details[email] . "'";
      if(db_connect() && mysql_query($query))
      {
        return true;
      }
      else
      {
        echo "could not store changes.<br><br><br><br><br><br>";
        return false;
      }
    }
  else
  {
    echo "<p>Sorry, that email address is already registered here.";
    echo "<p>You will need to log in with that address to change "
         ." Web settings.";
    return false;
  }
}
else // new account
{
  $query = "insert into subscribers
  values ('$details[email]','$details[realname]',
       '$details[mimetype]', password('$details[new_password]'), 0)";
  if(db_connect() && mysql_query($query))
  {
    return true;
  }
  else
{
  echo "Could not store new account.<br><br><br><br><br><br>";
  return false;
}
```

```
    }
  }
}

// need to process log in or out requests before anything else
if($email&&$password)
  {
    $login = login($email, $password);
    if($login == 'admin')
      {
        $status .= "<p><b>".get_real_name($email)."</b> logged in"
             ." successfully as <b>Administrator</b><br><br><br><br><br>";
        $admin_user = $email;
        session_register("admin_user");
      }
    else if($login == 'normal')
    {
      $status .= "<p><b>".get_real_name($email)."</b> logged in"
             ." successfully.<br><br>";
      $normal_user = $email;
      session_register("normal_user");
    }
    else
    {
      $status .= "<p>Sorry, we could not log you in with that
                  email address and password.<br>";
    }
  }
```

Fungsi ini pertama-tama memeriksa apakah pengguna telah mengisi detail yang diperlukan. Jika ini sudah benar, fungsi tersebut akan membuat pengguna baru, atau memperbarui detail akun jika pengguna tersebut sudah ada. Pengguna hanya dapat memperbarui detail akun pengguna yang login sebagai pengguna tersebut.

Ini diperiksa menggunakan fungsi `get_email()`, yang mengambil alamat email pengguna yang saat ini login. Kita akan membahasnya nanti, karena fungsi ini menggunakan variabel sesi yang ditetapkan saat pengguna login.

Login

Jika pengguna mengisi formulir login yang kita lihat pada Gambar 7.4 dan mengklik tombol Log In, ia akan memasukkan skrip `index.php` dengan variabel `$email` dan `$password` yang ditetapkan. Ini akan mengaktifkan kode login, yang berada dalam tahap pra-pemrosesan skrip, sebagai berikut:

Listing 7.4 Fungsi `login()` dari `user_auth_fns.php`—Fungsi Ini Memeriksa Detail Login Pengguna

```
function login($email, $password)
```

```
// check username and password with db
// if yes, return login type
// else return false
{
  // connect to db
  $conn = db_connect();
  if (!$conn)
    return 0;

  $query = "select admin from subscribers
                where email='$email'
                and password = password('$password')";

  //echo $query;
  $result = mysql_query($query);
  if (!$result)
    return false;
  if (mysql_num_rows($result)<1)
    return false;
  if(mysql_result($result, 0, 0) == 1)
    return 'admin';
  else
    return 'normal';
}
```

Seperti yang dapat Anda lihat, pertama-tama kami mencoba untuk login menggunakan fungsi login() dari pustaka `user_auth_fns.php`. Fungsi ini sedikit berbeda dari fungsi login yang telah kami gunakan di tempat lain, jadi kami akan membahasnya. Kode untuk fungsi ini ditunjukkan pada Listing 7.4.

Sebelumnya dengan fungsi login, kami telah mengembalikan true jika login berhasil dan false jika tidak. Dalam kasus ini, kami tetap mengembalikan false jika login gagal, tetapi jika berhasil kami mengembalikan tipe pengguna, baik 'admin' atau 'normal'. Kami memeriksa tipe pengguna dengan mengambil nilai yang disimpan di kolom admin di tabel pelanggan, untuk kombinasi alamat email dan kata sandi tertentu. Jika tidak ada hasil yang dikembalikan, kami mengembalikan false. Jika pengguna adalah administrator, nilai ini akan menjadi 1 (true), dan kami mengembalikan 'admin'. Jika tidak, kami mengembalikan 'normal'.

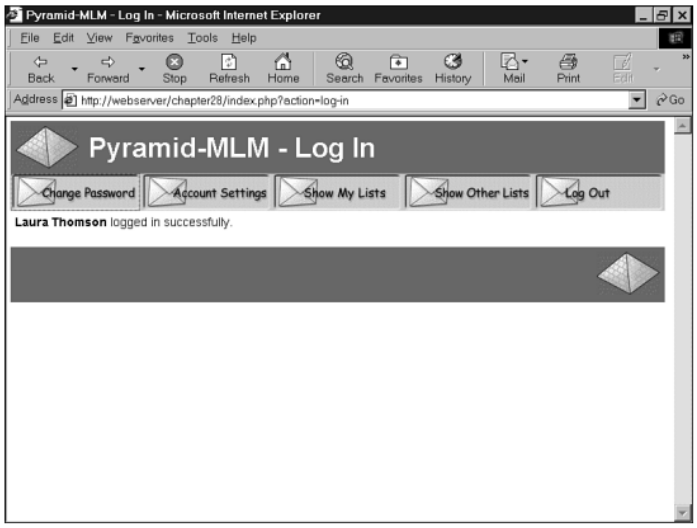
Kembali ke baris eksekusi utama, kami mendaftarkan variabel sesi untuk melacak siapa pengguna kami. Ini akan menjadi `$admin_user` jika dia adalah administrator, atau `$normal_user` jika dia adalah pengguna biasa. Mana pun dari variabel ini yang kami tetapkan akan berisi alamat email pengguna. Untuk menyederhanakan pemeriksaan alamat email pengguna, kami menggunakan `get_email()` `get email()` yang disebutkan sebelumnya.

Fungsi ini ditunjukkan dalam Listing 7.5.

Listing 28.5 fungsi `get_email()` dari `mlm_fns.php`—Mengembalikan Alamat Email Pengguna yang Masuk

```
function get_email()
{
  global $normal_user;
  global $admin_user;

  if (session_is_registered("normal_user"))
    return $normal_user;
  if (session_is_registered("admin_user"))
    return $admin_user;

  return false;
}
```

Kembali ke program utama kita, kita laporkan kepada pengguna apakah ia sudah masuk atau belum, dan pada level berapa. Output dari satu kali percobaan masuk ditunjukkan pada Gambar 7.6.



Gambar 7.6 Sistem melaporkan kepada pengguna bahwa proses masuk berhasil.

Sekarang setelah kita memasukkan pengguna, kita dapat melanjutkan ke fungsi pengguna.
Menerapkan Fungsi Pengguna

Ada lima hal yang kita inginkan agar dapat dilakukan pengguna setelah mereka masuk:

- Melihat daftar yang tersedia untuk berlangganan
- Berlangganan dan berhenti berlangganan dari daftar
- Mengubah cara akun mereka disiapkan
- Mengubah kata sandi mereka
- Keluar

Anda dapat melihat sebagian besar opsi ini pada Gambar 7.6. Sekarang kita akan melihat penerapan masing-masing opsi ini.

7.4 MELIHAT DAFTAR

Kami akan menerapkan sejumlah opsi untuk melihat daftar yang tersedia dan detail daftar. Pada Gambar 7.6, Anda dapat melihat dua opsi berikut: Tampilkan Daftar Saya, yang akan mengambil daftar yang dilanggani pengguna ini, dan Tampilkan Daftar Lainnya, yang akan mengambil daftar yang tidak dilanggani pengguna.

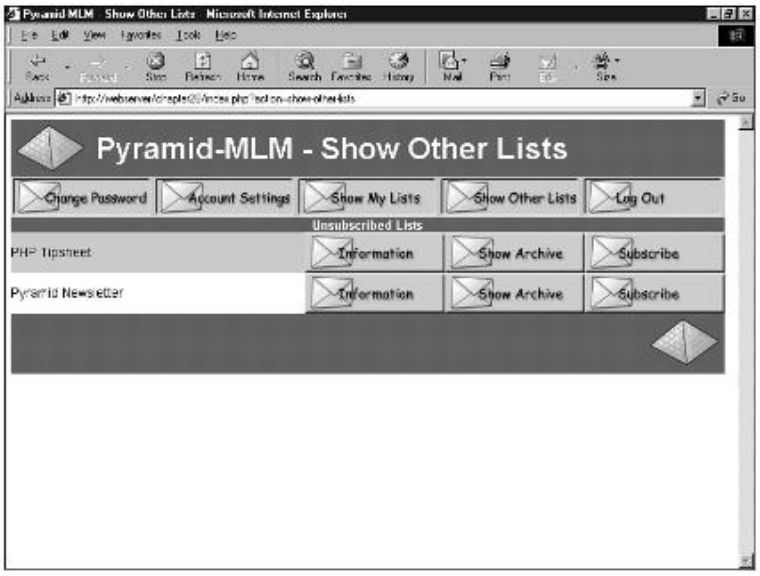
Jika Anda melihat kembali Gambar 7.4, Anda akan melihat bahwa kami memiliki opsi lain, Tampilkan Semua Daftar, yang akan mengambil semua milis yang tersedia di sistem. Agar sistem benar-benar dapat diskalakan, kami harus menambahkan fungsi paging (untuk menampilkan, katakanlah, 10 hasil per halaman). Kami tidak melakukannya di sini demi singkatnya.

Ketiga opsi menu ini mengaktifkan tindakan `show-my-lists`, `show-other-lists` dan `show-all-lists` Seperti yang mungkin Anda sadari, semua tindakan ini bekerja dengan sangat mirip. Kode untuk ketiga tindakan ini adalah sebagai berikut:

```
case ‘show-all-lists’ :
{
  display_items(“All Lists”, get_all_lists(), ‘information’,
                ‘show-archive’,’’);
  break;
}
case ‘show-other-lists’ :
{
    display_items(“Unsubscribed Lists”,
                get_unsubscribed_lists(get_email()), ‘information’,
                ‘show-archive’, ‘subscribe’);
  break;
}
case ‘’:
case ‘show-my-lists’ :
{
    display_items(“Subscribed Lists”, get_subscribed_lists(get_email()),
                ‘information’, ‘show-archive’, ‘unsubscribe’);
    break;
}
```

Seperti yang Anda lihat, semua tindakan ini memanggil fungsi `display_items()` dari pustaka `output_fns.php`, tetapi masing-masing memanggilnya dengan parameter yang berbeda. Semuanya juga menggunakan fungsi `get_email()` yang telah disebutkan sebelumnya untuk mendapatkan alamat email yang sesuai bagi pengguna ini.

Untuk melihat apa yang dilakukan fungsi ini, lihat Gambar 7.7.



Gambar 7.7 Fungsi `display_items()` telah digunakan untuk menyusun daftar daftar yang tidak dilanggani oleh pengguna.

Ini adalah halaman Tampilkan Daftar Lainnya. Mari kita lihat kode untuk fungsi `display_items()`. Kode ini ditampilkan dalam Listing 7.6.

Listing 7.6 Fungsi `display_items()` dari `output_fns.php`—Fungsi Ini Digunakan untuk Menampilkan Daftar Item dengan Tindakan Terkait

```
function display_items($title, $list, $action1='', $action2='', $action3='')
{
  global $table_width;
  echo "<table width = $table_width cellspacing = 0 cellpadding = 0
        border = 0>";

  // count number of actions
  $actions = (($action1!='') + ($action2!='') + ($action3!=''));

  echo   "<tr>
         <th colspan = ". (1+$actions) ." bgcolor='#5B69A6'> $title </th>
         </tr>";

  // count number of items
  $items = sizeof($list);
  if($items == 0)
    echo "<tr>
      <td colspan = ".(1+$actions)."align = center>No Items to Display</td>
      </tr>";
  else
  {
    // print each row
    for($i = 0; $items; $i++)
```

```
    {
      if($i%2) // background colors alternate
        $bgcolor = "'#ffffff'";
      else
        $bgcolor = "'#ccccff'";
      echo "<tr
           <td bgcolor = $bgcolor
           width = ". ($table_width - ($actions*149)) .">";
      echo $list[$i][1];
      if($list[$i][2])
        echo " - ".$list[$i][2];
      echo "</td>";

  // create buttons for up to three actions per line
  for($j = 1; $j<=3; $j++)
  {
    $var = 'action'.$j;
    if($$var)
      {
        echo "<td bgcolor = $bgcolor width = 149>";
        // view/preview buttons are a special case as they link to a file
      if($$var == 'preview-html'||$$var == 'view-html'||
        $$var == 'preview-text'||$$var == 'view-text')
      display_preview_button($list[$i][3], $list[$i][0], $$var);
    else
    display_button( $$var, "&id=" . $list[$i][0] );
      echo "</td>";
      }
    }
    echo "</tr>\n";
    }
  echo "</table>";
  }
}
```

Fungsi ini akan menampilkan tabel item, dengan setiap item memiliki hingga tiga tombol tindakan terkait. Fungsi ini mengharapkan lima parameter, yaitu:

- `$title` adalah judul yang muncul di bagian atas tabel—dalam kasus yang ditunjukkan pada Gambar 7.7, kita meneruskan judul Unsubscribed Lists, seperti yang ditunjukkan dalam cuplikan kode yang dibahas sebelumnya untuk tindakan "Show Other Lists".
- `$list` adalah larik item yang akan ditampilkan di setiap baris tabel. Dalam kasus ini, ini adalah larik daftar yang saat ini tidak dilanggani pengguna. Kita membangun larik ini (dalam kasus ini) dalam fungsi `get_unsubscribed_lists()`, yang akan kita bahas sebentar lagi. Ini adalah larik multidimensi, dengan setiap baris dalam larik berisi hingga empat bagian data tentang setiap baris. Secara berurutan, sekali lagi:

- `$list[n][0]` harus berisi id item, yang biasanya berupa nomor baris. Ini memberi tombol tindakan id baris yang akan dioperasikannya. Dalam kasus kita, kita akan menggunakan id dari basis data—lebih lanjut tentang ini sebentar lagi.
- `$list[n][1]` harus berisi nama item. Ini akan menjadi teks yang ditampilkan untuk item tertentu. Misalnya, dalam kasus yang ditunjukkan pada Gambar 7.7, nama item di baris pertama tabel adalah PHP Tipsheet.
- `$list[n][2]` dan `$list[n][3]` bersifat opsional. Kita menggunakan ini untuk menyampaikan bahwa ada lebih banyak informasi. Keduanya masing-masing sesuai dengan teks informasi lebih banyak dan id informasi lebih banyak.

Sebagai alternatif, kita dapat menggunakan kata kunci sebagai indeks array.

Parameter ketiga, keempat, dan kelima pada fungsi digunakan untuk meneruskan tiga tindakan yang akan ditampilkan pada tombol yang sesuai dengan setiap item. Pada Gambar 7.7, ini adalah tiga tombol tindakan yang ditampilkan sebagai Informasi, Tampilkan Arsip, dan Berlangganan. Kami memperoleh tiga tombol ini untuk halaman Tampilkan Semua Daftar dengan meneruskan nama tindakan, informasi, `show-archive`, dan `subscribe`. Dengan menggunakan fungsi `display_button()`, tindakan ini telah diubah menjadi tombol dengan kata-kata tersebut di dalamnya, dan tindakan yang sesuai ditetapkan untuknya.

Setiap tindakan Tampilkan memanggil fungsi `display_items()` dengan cara yang berbeda, seperti yang dapat Anda lihat dengan melihat kembali tindakannya. Selain memiliki judul dan tombol tindakan yang berbeda, masing-masing dari ketiganya menggunakan fungsi yang berbeda untuk membangun array item yang akan ditampilkan. Tampilkan Semua Daftar menggunakan fungsi `get_all_lists()`, Tampilkan Daftar Lainnya menggunakan fungsi `get_unsubscribed_lists()`, dan Tampilkan Daftar Saya menggunakan fungsi `get_subscribed_lists()`. Semua fungsi ini bekerja dengan cara yang sama. Semuanya berasal dari pustaka fungsi `mlm_fns.php`.

Kita akan melihat `get_unsubscribed_lists()` karena itulah contoh yang telah kita ikuti sejauh ini. Kode untuk fungsi `get_unsubscribed_lists()` ditunjukkan pada Listing 7.7.

Listing 7.7 Fungsi `get_unsubscribed_lists()` dari `mlm_fns.php`—Fungsi ini digunakan untuk membangun serangkaian milis yang tidak diikuti oleh pengguna

```
function get_unsubscribed_lists($email)
{
  $list = array();

$query = "select lists.listid, listname, email from lists left join sub_lists
       on lists.listid = sub_lists.listid
       and email='$email' where email is NULL order by listname";

if(db_connect())
{
  $result = mysql_query($query);
  if(!$result)
    echo "<p>Unable to get list from database.";
```

```
  $num = mysql_numrows($result);
  for($i = 0; $i<$num; $i++)
  {
    array_push($list, array(mysql_result($result, $i, 0),
          mysql_result($result, $i, 1)));
  }
}
return $list;
}
```

Seperti yang Anda lihat, fungsi ini memerlukan alamat email yang dimasukkan ke dalamnya. Ini harus menjadi alamat email pelanggan yang sedang kita tangani. Fungsi `get_subscribed_lists()` juga memerlukan alamat email sebagai parameter, tetapi fungsi `get_all_lists()` tidak memerlukannya karena alasan yang jelas.

Dengan alamat email pelanggan, kita terhubung ke basis data dan mengambil semua daftar yang tidak dilanggani pelanggan tersebut. Seperti biasa untuk kueri semacam ini di MySQL, kita menggunakan LEFT JOIN untuk menemukan item yang tidak cocok.

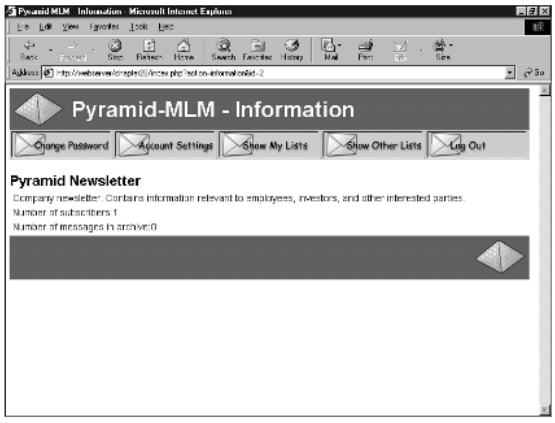
Kita mengulang hasilnya dan membuat array baris demi baris menggunakan fungsi bawaan `array_push()`. Sekarang setelah kita mengetahui bagaimana daftar ini dibuat, mari kita lihat tombol tindakan yang terkait dengan tampilan ini.

7.5 MELIHAT INFORMASI DAFTAR

Tombol Informasi yang ditunjukkan pada Gambar 7.7 memicu tindakan 'informasi', yaitu sebagai berikut:

```
case ‘information’ :
  {
    display_information($id);
    break;
  }
```

Untuk melihat apa yang dilakukan fungsi `display_information()`, lihat Gambar 7.8.



Gambar 7.8 Fungsi `display_information()` menampilkan uraian singkat tentang milis.

Fungsi ini menampilkan beberapa informasi umum tentang milis tertentu, serta mencantumkan jumlah pelanggan dan jumlah buletin yang telah dikirim ke milis tersebut dan tersedia di arsip. Kode untuk fungsi ini ditampilkan dalam Listing 7.8.

Listing 7.8 Fungsi `display_information()` dari `output_fns.php`—Fungsi Ini Menampilkan Informasi Daftar

```
function display_information($listid)
{
  if(!$listid)
    return false;
    $info = load_list_info($listid);
  if($info)
  {
    echo “<h2>”.pretty($info[listname]).”</h2>”;
    echo ‘<p>’.pretty($info[blurb]);
    echo ‘<p>Number of subscribers:’ . $info[subscribers];
    echo ‘<p>Number of messages in archive:’ . $info[archive];
  }
}
```

Fungsi `display_information()` menggunakan dua fungsi lain untuk membantunya mencapai tugas Web: fungsi `load_list_info()` dan fungsi `pretty()`. Fungsi `load_list_info()` sebenarnya mengambil data dari basis data. Fungsi `pretty()` hanya memformat data dari basis data dengan menghapus garis miring, mengubah baris baru menjadi pemisah baris HTML, dan seterusnya.

Mari kita lihat sekilas fungsi `load_list_info()`. Fungsi ini ada di pustaka fungsi `mlm_fns.php`. Kode untuk fungsi ini ditampilkan dalam Listing 7.9.

Listing 7.9 Fungsi `load_list_info()` dari `mlm_fns.php`—Fungsi Ini Membangun Array Informasi Daftar

```
function load_list_info($listid)
{
  if(!$listid)
    return false;

  if(!db_connect())
    return false;

  $query = “select listname, blurb from lists where listid = $listid”;
  $result = mysql_query($query);
  if(!$result)
  {
    echo “Cannot retrieve this list”;
    return false;
```

```
}
$info = mysql_fetch_array($result);

$query = "select count(*) from sub_lists where listid = $listid";
$result = mysql_query($query);
if($result)
{
  $info['subscribers'] = mysql_result($result, 0, 0);
}
  $query = "select count(*) from mail where listid = $listid
           and status = 'SENT'";
  $result = mysql_query($query);
  if($result)
  {
    $info['archive'] = mysql_result($result, 0, 0);
  }
  return $info;
}
```

Fungsi ini menjalankan tiga kueri basis data untuk mengumpulkan nama dan uraian untuk suatu milis dari tabel daftar; jumlah pelanggan dari tabel `sub_lists`; dan jumlah buletin yang dikirim dari tabel surat.

Melihat Arsip Milis

Selain melihat uraian milis, pengguna dapat melihat semua surat yang telah dikirim ke milis dengan mengeklik tombol Tampilkan Arsip. Ini mengaktifkan tindakan `show-archive`, yang memicu kode berikut:

```
case 'show-archive' :
{
  display_items("Archive For ".get_list_name($id),
                get_archive($id), 'view-html', 'view-text', '');
  break;
}
```

Sekali lagi, fungsi ini menggunakan fungsi `display_items()` untuk mencantumkan berbagai item surat yang telah dikirim ke daftar tersebut. Item-item ini diambil menggunakan fungsi `get_archive()` dari `mlm_fns.php`. Fungsi ini ditampilkan dalam Listing 7.10.

Listing 7.10 Fungsi `get_archive()` dari `mlm_fns.php`—Fungsi Ini Membangun Rangkaian Buletin yang Diarsipkan untuk Daftar Tertentu

```
function get_archive($listid)
{
  //returns an array of the archived mail for this list
  //array has rows like (mailid, subject)
```

```
  $list = array();
  $listname = get_list_name($listid);

  $query = "select mailid, subject, listid from mail
           where listid = $listid and status = 'SENT' order by sent";

  if(db_connect())
  {
    $result = mysql_query($query);
    if(!$result)
    {
      echo "<p>Unable to get list from database - $query.";
      return false;
    }
    $num = mysql_numrows($result);
    for($i = 0; $i<$num; $i++)
    {
      $row = array(mysql_result($result, $i, 0),
                mysql_result($result, $i, 1), $listname, $listid);
      array_push($list, $row);
    }
  }
  return $list;
}
```

Sekali lagi, fungsi ini mendapatkan informasi yang diperlukan—dalam hal ini, detail email yang telah dikirim—dari basis data dan membuat larik yang sesuai untuk diteruskan ke fungsi `display_items()`.

Berlangganan dan Berhenti Berlangganan

Pada daftar milis yang ditunjukkan pada Gambar 7.7, setiap milis memiliki tombol yang memungkinkan pengguna untuk berlangganan. Demikian pula, jika pengguna menggunakan opsi Tampilkan Daftar Saya untuk melihat milis yang telah mereka langgani, mereka akan melihat tombol Berhenti Berlangganan di samping setiap milis.

Tombol-tombol ini mengaktifkan tindakan berlangganan dan berhenti berlangganan, yang masing-masing memicu dua bagian kode berikut:

```
  case 'subscribe' :
  {
    subscribe(get_email(), $id);
    display_items("Subscribed Lists", get_subscribed_lists(get_email()),
                'information', 'show-archive', 'unsubscribe');
    break;
  }
  case 'unsubscribe' :
  {
```

```
        unsubscribe(get_email(), $id);
        display_items("Subscribed Lists", get_subscribed_lists(get_email()),
                  'information', 'show-archive', 'unsubscribe');
        break;
  }
```

Dalam setiap kasus, kami memanggil fungsi (`subscribe()` atau `unsubscribe()`) lalu menampilkan kembali daftar milis yang sekarang menjadi langganan pengguna menggunakan fungsi `display_items()` lagi. Fungsi `subscribe()` dan `unsubscribe()` ditampilkan dalam Listing 7.11.

Listing 7.11 Fungsi `subscribe()` dan `unsubscribe()` dari `mlm_fns.php`—Fungsi-Fungsi Ini Menambah dan Menghapus Langganan untuk Pengguna

```
function subscribe($email, $listid)
{
  if(!$email||!$listid||!list_exists($listid)||!subscriber_exists($email))
    return false;

  //if already subscribed exit
  if(subscribed($email, $listid))
    return false;

  if(!db_connect())
    return false;

  $query = "insert into sub_lists values ('$email', $listid)";

  $result = mysql_query($query);
            return $result;
}

function unsubscribe($email, $listid)
{
  if(!$email||!$listid)
    return false;

  if(!db_connect())
    return false;

$query = "delete from sub_lists where email = '$email' and listid = $listid";
$result = mysql_query($query);
return $result;
}
```

Fungsi `subscribe()` menambahkan baris ke tabel `sub_lists` yang sesuai dengan langganan; fungsi `unsubscribe()` menghapus baris ini.

Mengubah Pengaturan Akun

Tombol Pengaturan Akun, saat diklik, mengaktifkan tindakan pengaturan akun. Kode untuk tindakan ini adalah sebagai berikut:

```
case 'account-settings' :
{
  display_account_form($normal_user, $admin_user, get_email(),
  get_real_name(get_email()), get_mimetype(get_email()));
  break;
}
```

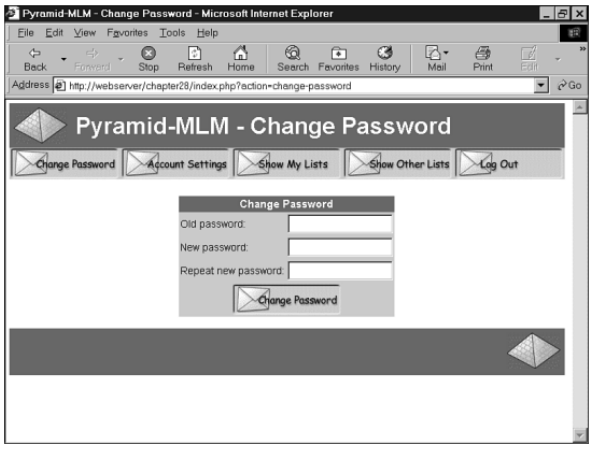
Seperti yang Anda lihat, kami menggunakan kembali fungsi `display_account_form()` yang kami gunakan untuk membuat akun di awal. Namun, kali ini kami memasukkan detail pengguna saat ini, yang akan ditampilkan dalam formulir untuk memudahkan pengeditan. Saat pengguna mengklik tombol kirim dalam formulir ini, tindakan store-account diaktifkan seperti yang dibahas sebelumnya.

Mengubah Kata Sandi

Mengklik tombol Ubah Kata Sandi mengaktifkan tindakan ubah kata sandi, yang memicu kode berikut:

```
case 'change-password' :
{
  display_password_form();
  break;
}
```

Fungsi `display_password_form()` (dari pustaka `output_fns.php`) hanya menampilkan formulir bagi pengguna untuk mengubah kata sandinya. Formulir ini ditunjukkan pada Gambar 7.9.



Gambar 7.9 Fungsi `display_password_form()` memungkinkan pengguna untuk mengubah kata sandi mereka.

Saat pengguna mengklik tombol Ubah Kata Sandi di bagian bawah formulir ini, tindakan `store-change-password` akan diaktifkan. Kode untuk ini adalah sebagai berikut:

```
case 'store-change-password' :
{
  if(change_password(get_email(), $old_passwd,
    $new_passwd, $new_passwd2))
  {
    echo "<p>OK: Password changed.<br><br><br><br><br><br>";
  }
  else
  {
    echo "<p>Sorry, your password could not be changed.";
    display_password_form();
  }
  break;
}
```

Seperti yang dapat Anda lihat, kode ini mencoba mengubah kata sandi menggunakan fungsi `change_password()` dan melaporkan keberhasilan atau kegagalan kepada pengguna. Fungsi `change_password()` dapat ditemukan di pustaka fungsi `user_auth_fns.php`. Kode untuk fungsi ini ditampilkan dalam Listing 7.12.

Listing 7.12 Fungsi `change_password()` dari `user_auth_fns.php`—Fungsi ini Memvalidasi dan Memperbarui Kata Sandi Pengguna.

```
function change_password($email, $old_password, $new_password,
                         $new_password_conf)
// change password for email/old_password to new_password
// return true or false
{
  // if the old password is right
  // change their password to new_password and return true
  // else return false
  if (login($email, $old_password))
  {
    if($new_password==$new_password_conf)
    {
      if (!($conn = db_connect()))
        return false;
      $query = "update subscribers
             set password = password('$new_password')
             where email = '$email'";
      $result = mysql_query($query);
      return $result;
    }
```

```
    else
      echo “<p> Your passwords do not match.”;
    }
    else
      echo “<p> Your old password is incorrect.”;

  return false; // old password was wrong
}
```

Fungsi ini mirip dengan fungsi pengaturan dan pengubahan kata sandi lain yang telah kita bahas. Fungsi ini membandingkan dua kata sandi baru yang dimasukkan oleh pengguna untuk memastikan keduanya sama, dan jika sama, fungsi ini mencoba memperbarui kata sandi pengguna dalam basis data.

Keluar dari Akun

Saat pengguna mengklik tombol Keluar, tindakan keluar akan dipicu. Kode yang dijalankan oleh tindakan ini dalam skrip utama sebenarnya ada di bagian praproses skrip, seperti berikut:

```
if($action == ‘log-out’)
{
  session_destroy();
  unset($action);
  unset($normal_user);
  unset($admin_user);
}
```

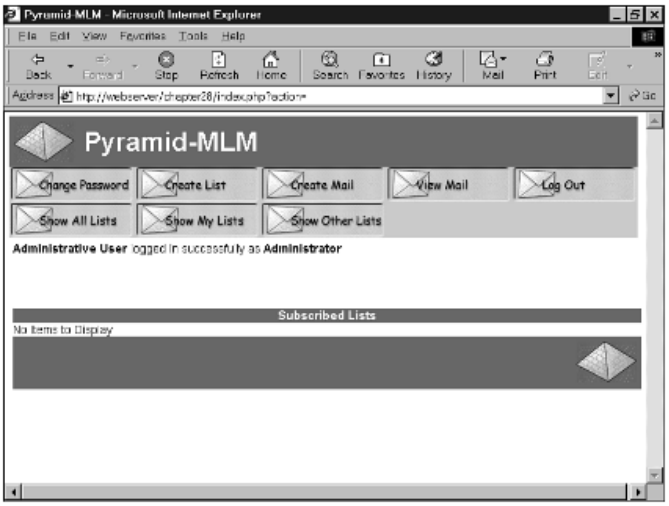
Potongan kode ini membuang variabel sesi dan menghancurkan sesi. Perhatikan bahwa potongan kode ini juga menghapus variabel `$action`—ini berarti kita memasukkan pernyataan kasus utama tanpa tindakan, yang memicu kode berikut:

```
case ‘’:
{
  if(!check_logged_in())
    display_login_form($action);
  break;
}
```

Ini akan memungkinkan pengguna lain untuk masuk, atau memungkinkan pengguna untuk masuk sebagai orang lain.

7.6 MENERAPKAN FUNGSI ADMINISTRATIF

Jika seseorang masuk sebagai administrator, ia akan mendapatkan beberapa opsi menu tambahan, yang dapat dilihat pada Gambar 7.10.



Gambar 7.10 Menu administrator memungkinkan pembuatan dan pemeliharaan milis.

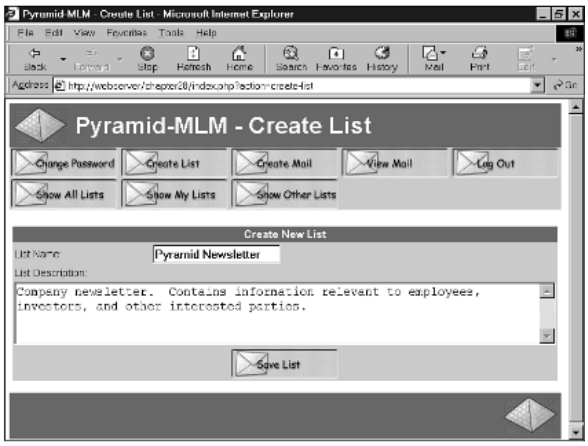
Opsi tambahan yang mereka miliki adalah Buat Daftar (buat milis baru), Buat Surat (buat buletin baru), dan Lihat Surat (lihat dan kirim buletin yang dibuat yang belum dikirim). Kita akan membahas masing-masing secara bergantian.

Membuat Daftar Baru

Jika administrator memilih untuk membuat daftar baru dengan mengklik tombol Buat Daftar, ia akan mengaktifkan tindakan buat-daftar, yang dikaitkan dengan kode berikut:

```
case 'create-list' :
{
  display_list_form(get_email());
break;
}
```

Fungsi `display_list_form()` menampilkan formulir yang memungkinkan administrator memasukkan detail daftar baru. Fungsi ini dapat ditemukan di pustaka `output_fns.php`. Fungsi ini hanya menampilkan HTML, jadi kita tidak akan membahasnya di sini. Output fungsi ini ditunjukkan pada Gambar 7.11.



Gambar 7.11 Pilihan Buat Daftar mengharuskan administrator memasukkan nama dan deskripsi (atau uraian) untuk daftar baru tersebut.

Saat administrator mengklik tombol Simpan Daftar, tindakan penyimpanan daftar akan diaktifkan, yang memicu kode berikut di index.php:

```
case ‘store-list’ :
{
  if(store_list($admin_user, $HTTP_POST_VARS))
  {
    echo “<p>New list added<br>”;
    display_items(“All Lists”, get_all_lists(), ‘information’,
                  ‘show-archive’,’’);
  }
  else
    echo “<p>List could not be stored, please try “
                  .”again.<br><br><br><br><br>”;
  break;
}
```

Seperti yang dapat Anda lihat, kode tersebut mencoba menyimpan detail daftar baru dan kemudian menampilkan daftar daftar yang baru. Detail daftar disimpan dengan fungsi `store_list()`. Kode untuk fungsi ini ditampilkan dalam Listing 7.13.

Listing 7.13 Fungsi store_list() dari mlm_fns.php—Fungsi Ini Menyisipkan Mailing List Baru ke dalam Basis Data

```
function store_list($admin_user, $details)
{
  if(!filled_out($details))
  {
    echo “All fields must be filled in. Try again.<br><br>”;
      return false;
  }
  else
  {
    if(!check_admin_user($admin_user))
      return false;
    // how did this function get called by somebody not logged in as admin?
    if(!db_connect())
    {
      return false;
    }
    $query = “select count(*) from lists where listname = ‘$details[name]’”;
    $result = mysql_query($query);
    if(mysql_result($result, 0, 0) > 0)
  {
    echo “Sorry, there is already a list with this name.”;
    return false;
```

```
    }
  $query = "insert into lists values (NULL,
           '$details[name]',
           '$details[blurb]')";
  $result = mysql_query($query);
    return $result;
  }
}
```

Fungsi ini melakukan beberapa pemeriksaan validasi sebelum menulis ke basis data: Fungsi ini memeriksa apakah semua detail telah diberikan, bahwa pengguna saat ini adalah administrator, dan bahwa nama daftar tersebut unik. Jika semuanya berjalan lancar, daftar tersebut ditambahkan ke tabel daftar dalam basis data.

Mengunggah Newsletter Baru

Akhirnya, kita sampai pada inti utama aplikasi ini: mengunggah dan mengirim newsletter ke milis. Saat administrator mengklik tombol Buat Email, tindakan buat-email akan diaktifkan, seperti berikut:

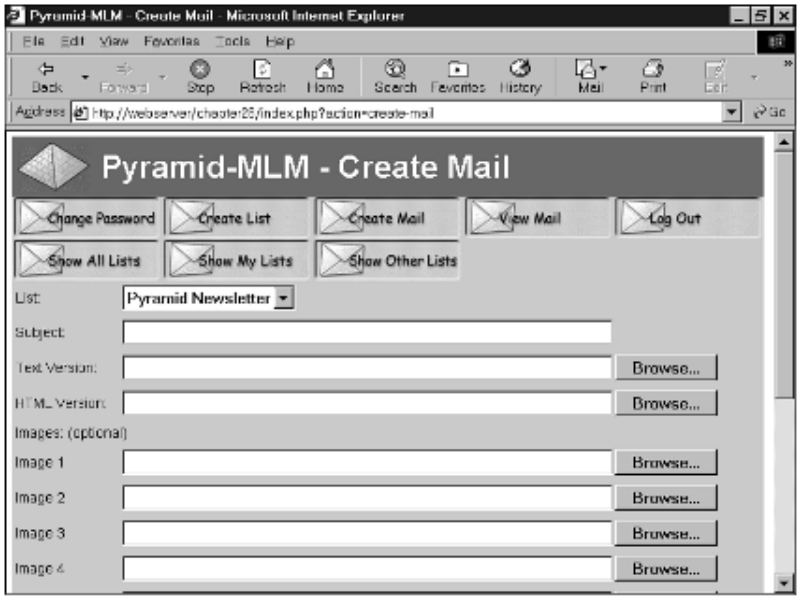
```
case 'create-mail' :
{
  display_mail_form(get_email());
    break;
}
```

Administrator akan melihat formulir yang ditunjukkan pada Gambar 7.12. Ingatlah bahwa untuk aplikasi ini, kami berasumsi bahwa administrator telah membuat buletin secara luring dalam format HTML dan teks dan akan mengunggah kedua versi tersebut sebelum mengirimnya. Kami memilih untuk menerapkannya dengan cara ini agar administrator dapat menggunakan perangkat lunak favorit mereka untuk membuat buletin. Hal ini membuat aplikasi lebih mudah diakses.

Anda dapat melihat bahwa formulir ini memiliki sejumlah bidang yang harus diisi oleh administrator. Di bagian atas terdapat kotak drop-down berisi milis yang dapat dipilih. Administrator juga harus mengisi subjek untuk buletin—ini adalah baris Subjek untuk email yang akan dikirim.

Semua bidang formulir lainnya adalah bidang unggah berkas, yang dapat Anda lihat dari tombol Telusuri di sebelahnya. Untuk mengirim buletin, administrator harus mencantumkan versi teks dan HTML buletin ini (meskipun jelas Anda dapat mengubahnya sesuai kebutuhan).

Ada juga sejumlah bidang gambar opsional tempat administrator dapat mengunggah gambar apa pun yang telah disematkannya dalam HTML-nya. Setiap berkas ini harus ditentukan dan diunggah secara terpisah.



Gambar 7.12 Opsi Buat Email memberi administrator antarmuka untuk mengunggah berkas buletin.

Formulir yang Anda lihat mirip dengan formulir unggah berkas biasa, kecuali dalam kasus ini, kita menggunakannya untuk mengunggah beberapa berkas. Hal ini memerlukan beberapa perbedaan kecil dalam sintaksis formulir, dan dalam cara kita menangani berkas yang diunggah di ujung lainnya. Kode untuk fungsi `display_mail_form()` ditampilkan dalam Listing 7.14.

Listing 7.14 Fungsi `display_mail_form()` dari `output_fns.php`—Fungsi Ini Menampilkan Formulir Unggah Berkas

```
function display_mail_form($email, $listid=0)
{
  // display html form for uploading a new message
  global $table_width;
  $list = get_all_lists();
  $lists = sizeof($list);
  ?>
  <table cellpadding=4 cellspacing = 0 border = 0 width = <
  ?=$table_width?>>
  <form enctype='multipart/form-data' action='upload.php' method='post'>
  <tr>
  <td bgcolor = "#cccccc">
  List:
  </td>
    <td bgcolor = "#cccccc">
    <select name = list>
  <?
  for($i = 0; $i<$lists; $i++)
    {
```

```
        echo "<option value = ".$list[$i][0];
          if ($listid== $list[$i][0]) echo " selected";
        echo ">".$list[$i][1]."</option>\n";
        }
        ?>
        </select>
      </td>
    </tr>
<tr>
    <td bgcolor = "#cccccc">
      Subject:
    </td>
    <td bgcolor = "#cccccc">
      <input type = text name = subject value = "<?=$subject?>"
        size = 60 ></td>
    </tr>
    <tr><td bgcolor = "#cccccc">
      Text Version:
    </td><td bgcolor = "#cccccc">
      <input type=file name='userfile[0]' size = 60>
    </td></tr>
    <tr><td bgcolor = "#cccccc">
      HTML Version:
    </td><td bgcolor = "#cccccc">
      <input type=file name='userfile[1]' size = 60>
    </td></tr>
    <tr><td bgcolor = "#cccccc" colspan =2>Images: (optional)
<?
    $max_images = 10;
    for($i = 0; $i<10; $i++)
    {
      echo "<tr><td bgcolor = '#cccccc'>Image ". ($i+1) ." </td>";
      echo "<td bgcolor = '#cccccc'>";
      echo "<input type=file name='userfile[".($i+2)."]' size =
      60></td></tr>";
    }
?>
    <tr><td colspan = 2 bgcolor = '#cccccc' align = center>
    <input type = hidden name = max_images value = <?=$max_images?>>
    <input type = hidden name = listid value = <?=$listid?>>
    <? display_form_button('upload-files'); ?>
</td>
    </form>
      </tr>
    </table>
<?
}
```

Hal yang perlu diperhatikan di sini adalah bahwa file yang ingin kita unggah akan memiliki nama yang dimasukkan dalam serangkaian input, masing-masing bertipe file, dan dengan nama yang berkisar dari `userfile[0]` hingga `userfile[n]`. Intinya, kita memperlakukan kolom formulir ini dengan cara yang sama seperti kita memperlakukan kotak centang, dan menamainya menggunakan konvensi array.

Jika Anda ingin mengunggah beberapa file melalui skrip PHP, Anda harus mengikuti konvensi ini. Dalam skrip yang memproses formulir ini, kita sebenarnya akan mendapatkan tiga array. Mari kita lihat skrip tersebut.

7.7 MENANGANI PENGUNGGAHAN BEBERAPA FILE

Anda mungkin ingat bahwa kita meletakkan kode pengunggahan file dalam file terpisah. Daftar lengkap file tersebut, `upload.php`, ditampilkan dalam Listing 7.15.

Listing 7.15 `upload.php`—Skrip Ini Mengunggah Semua File yang Diperlukan untuk Buletin

```
<?
  // this functionality is in a separate file to allow us to be
  // more paranoid with it

  // if anything goes wrong, we will exit

  $max_size = 50000;

  include ('include_fns.php');
  session_start();

  // only admin users can upload files
  if(!check_admin_user())
  {
    echo "You do not seem to be authorized to use this page.";
    exit;
  }

  // set up the admin toolbar buttons
  $buttons = array();
  $buttons[0] = 'change-password';
  $buttons[1] = 'create-list';
  $buttons[2] = 'create-mail';
  $buttons[3] = 'view-mail';
  $buttons[4] = 'log-out';
  $buttons[5] = 'show-all-lists';
  $buttons[6] = 'show-my-lists';
  $buttons[7] = 'show-other-lists';

  do_html_header("Pyramid-MLM - Upload Files");
```

```
display_toolbar($buttons);

// check that the page is being called with the required data
if(!$userfile_name[0]||!$userfile_name[1]||!$subject||!$list)
{
   echo "Problem: You did not fill out the form fully. The images are the
        only optional fields. Each message needs a subject, text version
        and an HTML version.";
   do_html_footer();
   exit;
}
if(!db_connect())
{
   echo "<p>Could not connect to db";
   do_html_footer();
   exit;
}

// add mail details to the DB
$query = "insert into mail values (NULL,  '$admin_user',
                        '$subject', '$list', 'STORED', NULL, NULL)";

$result = mysql_query($query);
if(!$result)
{
   do_html_footer();
   exit;
}

//get the id MySQL assigned to this mail
$mailid = mysql_insert_id();

if(!$mailid)
{
   do_html_footer();
   exit;
}

// creating directory will fail if this is not the first message archived
// that's ok
@ mkdir("archive/$list", 0700);

// it is a problem if creating the specific directory for this mail fails
if(!mkdir("archive/$list/$mailid", 0700))
{
   do_html_footer();
   exit;
```

```
}

// iterate through the array of uploaded files
$i = 0;
while ($userfile[$i]&&$userfile[$i]!='none')
{
  echo "<p>Uploading ".$userfile_name[$i]." - ";
  echo $userfile_size[$i]." bytes.<br>";
  if ($userfile_size[$i]==0)
  {
    echo "Problem: $userfile_name[$i] is zero length";
    $i++;
    continue;
  }

  if ($userfile_size[$i]>$max_size)
  {
    echo "Problem: $userfile_name[$i] is over 10000 bytes";
    $i++;
    continue;
  }

  // we would like to check that the uploaded image is an image
  // if getimagesize() can work out Web size, it probably is.
  if($i>1&&!getimagesize($userfile[$i]))
  {
    echo "Problem: $userfile_name[$i] is corrupt, or not a gif, jpeg or png";
    $i++;
    continue;
  }

  // file 0 (the text message) and file 1 (the html message) are special
  //cases

  if($i==0)
    $destination = "archive/$list/$mailid/text.txt";
  else if($i == 1)
    $destination = "archive/$list/$mailid/index.html";
  else
  {
    $destination = "archive/$list/$mailid/".$userfile_name[$i];
    $query = "insert into images values ($mailid,
              '".$userfile_name[$i]."',
              '".$userfile_type[$i]."')";
    $result = mysql_query($query);
  }
```

```
    //if we are using PHP version >= 4.03
/*
    if (!is_uploaded_file($userfile[$i]))
    {
      // possible file upload attack detected
        echo "Something funny happening with '$userfile', not uploading.";
        do_html_footer();
        exit;
    }

    move_uploaded_file($userfile[$i], $destination);
*/
    // if version <= 4.02
    copy ($userfile[$i], $destination);

    unlink($userfile[$i]);

    $i++;
  }

  display_preview_button($list, $mailid, 'preview-html');
  display_preview_button($list, $mailid, 'preview-text');
  display_button('send', "&id=$mailid");

  echo "<br><br><br><br><br>";
  do_html_footer();
?>
```

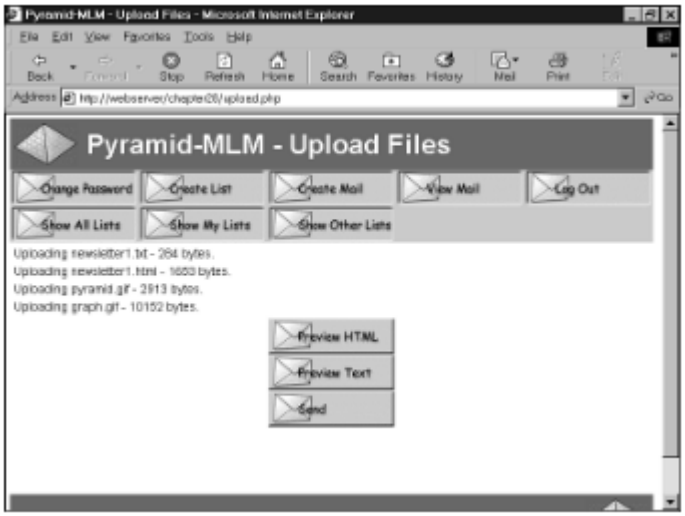
Mari kita telusuri langkah-langkah pada Listing 7.15.

Pertama, kita memulai sesi dan memeriksa apakah pengguna masuk sebagai administrator—kita tidak ingin membiarkan orang lain mengunggah file. Secara tegas, kita mungkin juga harus memeriksa variabel `$list` dan `$mailid` untuk karakter yang tidak diinginkan, tetapi kita mengabaikannya demi singkatnya.

Selanjutnya, kita menyiapkan dan mengirim header untuk halaman tersebut, dan memvalidasi bahwa formulir telah diisi dengan benar. Ini penting di sini karena ini adalah formulir yang cukup rumit untuk diisi oleh pengguna.

Kemudian kita membuat entri untuk email ini di basis data, dan menyiapkan direktori di arsip tempat email akan disimpan. Selanjutnya adalah bagian utama skrip, yang memeriksa dan memindahkan setiap file yang diunggah. Ini adalah bagian yang berbeda saat mengunggah beberapa file. Sekarang kita memiliki tiga array untuk ditangani. Array ini disebut `$userfile`, `$userfile_name`, dan `$userfile_size`. Array tersebut sesuai dengan padanannya yang memiliki nama yang sama dalam unggahan file tunggal, kecuali bahwa masing-masing array tersebut adalah array. File pertama dalam formulir akan dirinci dalam `$userfile[0]`, `$userfile_name[0]`, dan `$userfile_size[0]`.

Dengan tiga array ini, kami melakukan pemeriksaan keamanan biasa dan memindahkan file ke arsip. Terakhir, kami memberikan administrator beberapa tombol yang dapat mereka gunakan untuk melihat pratinjau buletin yang telah mereka unggah sebelum mereka mengirimkannya, dan tombol untuk mengirimkannya. Anda dapat melihat output dari upload.php pada Gambar 7.13.



Gambar 7.13 Skrip unggah melaporkan file yang diunggah dan ukurannya.

Melihat Pratinjau Buletin

Ada dua cara administrator dapat melihat pratinjau buletin sebelum mengirimnya. Ia dapat mengakses fungsi pratinjau dari layar unggah jika ia ingin melihat pratinjau segera setelah mengunggah.

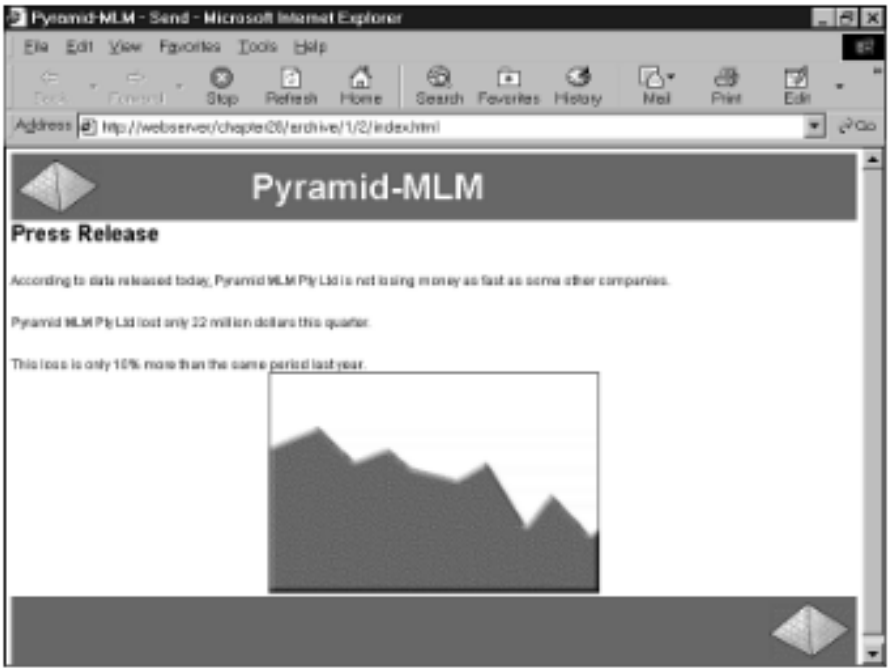
Dia juga dapat mengeklik tombol Lihat Email, yang akan menunjukkan semua buletin yang belum terkirim dalam sistem, jika dia ingin melihat pratinjau dan mengirim email nanti. Tombol Lihat Email mengaktifkan tindakan `view-mail`, yang memicu kode berikut:

```
case ‘view-mail’ :
{
  display_items(“Unsent Mail”, get_unsent_mail(get_email()),
                ‘preview-html’, ‘preview-text’, ‘send’);
  break;
}
```

Seperti yang dapat Anda lihat, ini kembali menggunakan fungsi `display_items()` dengan tombol untuk tindakan `preview-html`, `preview-text`, dan `send`.

Satu hal menarik yang perlu diperhatikan adalah bahwa tombol "Preview" tidak benar-benar memicu suatu tindakan, tetapi menautkan langsung ke buletin di arsip. Jika Anda melihat kembali Listing 7.6 dan 28.15, Anda akan melihat bahwa kami menggunakan fungsi `display_preview_button()` untuk membuat tombol-tombol ini, alih-alih fungsi `display_button()` yang biasa.

Fungsi `display_button()` membuat tautan gambar ke skrip dengan parameter GET jika diperlukan; fungsi `display_preview_button()` memberikan tautan biasa ke arsip. Tautan ini akan muncul di jendela baru, dicapai dengan menggunakan atribut `target=new` dari tag jangkar HTML. Anda dapat melihat hasil pratinjau versi HTML buletin pada Gambar 7.14.



Gambar 7.14 Pratinjau buletin HTML, lengkap dengan gambar.

Mengirim Pesan

Mengklik tombol Kirim untuk menerima buletin akan mengaktifkan tindakan pengiriman, yang memicu kode berikut:

```
case 'send' :
{
  send($id, $admin_user);
  break;
}
```

Ini memanggil fungsi `send()`, yang dapat Anda temukan di pustaka `mlm_fns.php`. Ini adalah fungsi yang cukup panjang. Ini juga merupakan titik di mana kita menggunakan kelas HTML MIME Mail. Kode untuk fungsi kita ditunjukkan dalam Listing 7.16.

Listing 7.16 Fungsi `send()` dari `mlm_fns.php`–Fungsi Ini Akhirnya Mengirimkan Newsletter

```
// create the message from the stored DB entries and files
// send test messages to the administrator, or real messages to the whole list
function send($mailid, $admin_user)
{
  if(!check_admin_user($admin_user))
```

```
    return false;

if(!($info = load_mail_info($mailid)))
{
    echo "Cannot load list information for message $mailid";
    return false;
}
$subject = $info[0];
$listid = $info[1];
$status = $info[2];
$sent = $info[3];

$from_name = 'Pyramid MLM';

$from_address = 'return@address';

$query = "select email from sub_lists where listid = $listid";

$result = mysql_query($query);
if (!$result)
{
    echo $query;
    return false;
}
else if (mysql_num_rows($result)==0)
{
    echo "There is nobody subscribed to list number $listid";
    return false;
}
else
{
    include('class.html.mime.mail.inc');

    $mail = new html_mime_mail();

    // read in the text version
    $filename = "archive/$listid/$mailid/text.txt";
    $fp = fopen ($filename, "r");
    $text = fread($fp, filesize($filename));
    fclose ($fp);

    // read in the HTML version
    $filename = "archive/$listid/$mailid/index.html";
    $fp = fopen ($filename, "r");
    $html = fread($fp, filesize($filename));
    fclose ($fp);
```

```
  // get the list of images that relate to this message
  $query = "select path, mimetype from images where mailid = $mailid";
  if(db_connect())
  {
    $result = mysql_query($query);
    if(!$result)
    {
      echo "<p>Unable to get image list from database.";
      return false;
    }
    $num = mysql_numrows($result);
    for($i = 0; $i<$num; $i++)
    {
    //load each image from disk
    $filename = "archive/$listid/$mailid/".mysql_result($result, $i, 0);
    $fp = fopen($filename, 'r');
    $image = fread($fp, filesize($filename));
    fclose($fp);

    // add images to the mimemail object
    $mail->add_html_image($image, mysql_result($result, $i, 0),
              mysql_result($result, $i, 1));
  }
}

// add HTML and text to the mimemail object
$mail->add_html($html, $text);

// note that we build and encode the message here outside the loop,
// not repeatedly inside the loop
$mail->build_message();
if($status == 'STORED')
{

//send the HTML version of the message to administrator
$mail->send(get_real_name($admin_user), $admin_user,
$from_name, $from_address, $subject);

//send the text version of the message to administrator
mail(get_real_name($admin_user)." <".$admin_user.">", $subject,
$text, "From: $from_name <$from_address>");
echo "Mail sent to $admin_user";
  $query = "update mail set status = 'TESTED' where mailid = $mailid";
if(db_connect())
{
  $result = mysql_query($query);
}
```

```
  echo "<p>Press send again to send mail to whole list.<center>";
display_button('send', "&id=$mailid");
echo "</center>";
}
  else if($status == 'TESTED')
{

//send to whole list
$query = "select subscribers.realname, sub_lists.email,
subscribers.mimetype
from sub_lists, subscribers
where listid = $listid and
sub_lists.email = subscribers.email";
if(!db_connect())
  return false;
  $result = mysql_query($query);
if(!$result)
  echo "<p>Error getting subscriber list";
  $count = 0;

// for each subscriber
while( $subscriber = mysql_fetch_row($result) )
{
  if($subscriber[2]=='H')

  //send HTML version to people who want it
  $mail->send($subscriber[0], $subscriber[1], $from_name,
  $from_address, $subject);
  else
  //send text version to people who don't want HTML mail
  mail($subscriber[0]." <".$subscriber[1].">", $subject,
  $text, "From: $from_name <$from_address>");
  $count++;
}

$query = "update mail set status = 'SENT', sent = now()
where mailid = $mailid";
if(db_connect())
{
  $result = mysql_query($query);
}
  echo "<p>A total of $count messages were sent.";
}
  else if($status == 'SENT')
{
  echo "<p>This mail has already been sent.";
}
```

```
  }
}
```

Fungsi ini melakukan beberapa hal yang berbeda. Fungsi ini menguji pengiriman buletin ke administrator sebelum mengirimkannya. Fungsi ini melacaknya dengan melacak status surat dalam basis data. Saat skrip unggah mengunggah surat, skrip ini menetapkan status awal surat tersebut menjadi "STORED".

Jika fungsi send() menemukan bahwa surat berstatus "STORED", fungsi ini akan memperbaruinya menjadi "TESTED" dan mengirimkannya ke administrator. Status "TESTED" berarti buletin telah diuji kirim ke administrator. Jika statusnya adalah "TESTED", statusnya akan diubah menjadi "SENT" dan dikirim ke seluruh daftar.

Ini berarti setiap bagian surat pada dasarnya harus dikirim dua kali: sekali dalam mode uji dan sekali dalam mode nyata. Fungsi ini juga mengirim dua jenis email yang berbeda: versi teks, yang dikirim menggunakan fungsi `mail()` PHP; dan jenis HTML, yang dikirim menggunakan kelas HTML MIME Mail. Kita telah menggunakan `mail()` berkali-kali dalam buku ini, jadi mari kita lihat bagaimana kita menggunakan kelas HTML MIME Mail. Kita tidak akan membahas kelas ini secara menyeluruh, tetapi akan menjelaskan bagaimana kita menggunakannya dalam aplikasi yang cukup umum ini.

Kita mulai dengan menyertakan berkas kelas dan membuat contoh kelas:

```
include('class.html.mime.mail.inc');
$mail = new html_mime_mail();
```

Kami kemudian memuat detail gambar dari basis data dan mengulanginya, menambahkan setiap gambar ke bagian surat yang ingin kami kirim:

```
$mail->add_html_image($image,
                  mysql_result($result, $i, 0),
                  mysql_result($result, $i, 1));
```

Tiga parameter yang kami berikan ke `add_html_image()` adalah konten gambar aktual sebagaimana yang dibaca dari berkas, nama berkas, dan tipe MIME berkas.

Kami juga perlu menambahkan teks isi, baik dalam format HTML maupun teks:

```
$mail->add_html($html, $text);
```

Kemudian kami membuat isi email yang sebenarnya:

```
$mail->build_message();
```

Terakhir, setelah membuat isi pesan, kami dapat mengirimkannya. Kami melakukan ini dengan mengambil dan mengulang setiap pengguna yang berlangganan ke daftar ini, dan

menggunakan HTML MIMEMIME Mail `send()` atau `mail()` biasa tergantung pada preferensi tipe MIME pengguna:

```
if($subscriber[2]=='H')
  //send HTML version to people who want it
  $mail->send($subscriber[0], $subscriber[1], $from_name,
                  $from_address, $subject);
else
  //send text version to people who don't want HTML mail
  mail($subscriber[0]." <".$subscriber[1].">", $subject,
                  $text, "From: $from_name <$from_address>");
```

Parameter pertama `$mail->send()` harus berupa nama asli pengguna, dan parameter kedua harus berupa alamat emailnya. Selesai! Kita sekarang telah menyelesaikan pembuatan aplikasi milis.

Memperluas Proyek

Seperti biasa dengan proyek-proyek ini, ada banyak cara untuk memperluas fungsionalitas. Anda mungkin ingin

- Mengonfirmasi keanggotaan dengan pelanggan sehingga orang tidak dapat berlangganan tanpa izin mereka. Ini biasanya dilakukan dengan mengirimkan email ke akun mereka dan menghapus mereka yang tidak membalas. Pendekatan ini juga akan membersihkan alamat email yang salah dari basis data.
- Memberikan wewenang kepada administrator untuk menyetujui atau menolak pengguna yang ingin berlangganan milis mereka.
- Menambahkan fungsionalitas milis terbuka yang memungkinkan setiap anggota untuk mengirim email ke milis.
- Hanya mengizinkan anggota terdaftar untuk melihat arsip milis tertentu.
- Mengizinkan pengguna untuk mencari milis yang sesuai dengan kriteria tertentu. Misalnya, pengguna mungkin tertarik dengan buletin golf. Setelah jumlah buletin bertambah melewati ukuran tertentu, pencarian akan berguna untuk menemukan buletin tertentu.

BAB 8
MEMBANGUN FORUM WEB

Salah satu cara yang baik untuk membuat pengguna kembali ke situs Anda adalah dengan menawarkan forum Web. Forum ini dapat digunakan untuk berbagai keperluan, seperti grup diskusi filosofis dan dukungan teknis produk. Dalam bab ini, kami mengimplementasikan forum Web dalam PHP. Alternatifnya adalah menggunakan paket yang sudah ada, seperti Phorum, untuk menyiapkan forum Anda.

Forum Web terkadang juga disebut papan diskusi atau grup diskusi berulir. Ide forum adalah orang-orang dapat mengeposkan artikel atau pertanyaan kepada mereka, dan orang lain dapat membaca dan menjawab pertanyaan mereka. Setiap topik diskusi dalam forum disebut utas. Kami akan mengimplementasikan forum Web yang disebut blah-blah dengan fungsi berikut.

- Memulai utas diskusi baru dengan mengeposkan artikel
- Mengeposkan artikel sebagai balasan terhadap artikel yang sudah ada
- Melihat artikel yang telah diposkan
- Melihat utas percakapan dalam forum
- Melihat hubungan antarartikel, yaitu, melihat artikel mana yang merupakan balasan terhadap artikel lain

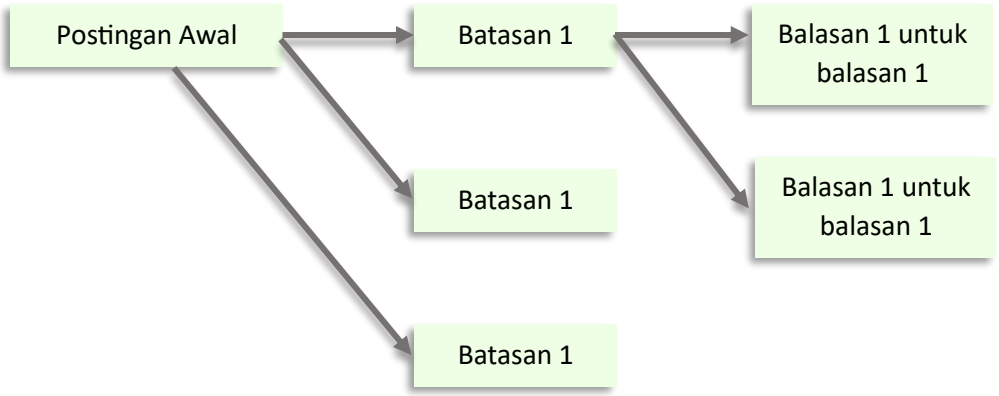
8.1 PENDAHULUAN

Menyiapkan forum sebenarnya merupakan masalah yang cukup menarik. Kita akan memerlukan beberapa cara untuk menyimpan artikel dalam basis data dengan informasi penulis, judul, tanggal, dan konten. Sekilas, ini mungkin tampak tidak jauh berbeda dari basis data Book-O-Rama.

Namun, cara kerja sebagian besar perangkat lunak diskusi berulir adalah, selain menunjukkan artikel yang tersedia, perangkat lunak ini juga akan menunjukkan hubungan antarartikel. Artinya, Anda dapat melihat artikel mana yang merupakan balasan untuk artikel lain (dan artikel mana yang ditindaklanjuti) dan artikel mana yang merupakan topik diskusi baru.

Memutuskan cara menampilkan hubungan ini memerlukan pemikiran yang cermat. Untuk sistem ini, pengguna harus dapat melihat pesan individual, utas percakapan dengan hubungan yang ditampilkan, atau semua utas pada sistem.

Pengguna juga harus dapat mengeposkan topik atau balasan baru. Ini adalah bagian yang mudah. Seperti yang telah kami katakan sebelumnya, menyimpan dan mengambil penulis dan teks pesan itu mudah. Bagian tersulit dari aplikasi ini adalah menemukan struktur basis data yang akan menyimpan informasi yang kita inginkan, dan cara menavigasi struktur tersebut secara efisien. Struktur artikel dalam diskusi mungkin terlihat seperti yang ditunjukkan pada Gambar 8.1.



Gambar 8.1 Sebuah artikel dalam diskusi berulir mungkin merupakan artikel pertama dalam topik baru, tetapi lebih umum merupakan respons terhadap artikel lain.

Dalam diagram ini, Anda dapat melihat bahwa kami memiliki posting awal yang dimulai dari suatu topik, dengan tiga balasan. Beberapa balasan memiliki balasan. Balasan ini dapat memiliki balasan, dan seterusnya.

Melihat diagram tersebut memberi kita petunjuk tentang cara menyimpan dan mengambil data artikel dan tautan antarartikel. Diagram ini menunjukkan struktur pohon. Jika Anda telah melakukan banyak pemrograman, Anda akan tahu bahwa ini adalah salah satu struktur data pokok yang digunakan. Dalam diagram tersebut terdapat simpul—atau artikel—dan tautan—atau hubungan antarartikel—seperti dalam struktur pohon apa pun.

(Jika Anda tidak familier dengan pohon sebagai struktur data, jangan khawatir—kami akan membahas dasar-dasarnya seiring berjalannya waktu.)

Trik agar semua ini berfungsi adalah

1. Menemukan cara untuk memetakan struktur pohon ini ke dalam penyimpanan—dalam kasus kami, ke dalam basis data MySQL.
2. Menemukan cara untuk merekonstruksi data sesuai kebutuhan.

Kita akan mulai dengan menerapkan basis data MySQL yang akan memungkinkan kita untuk menyimpan artikel di antara penggunaan. Kita akan membangun antarmuka sederhana untuk memungkinkan penyimpanan artikel.

Saat kita memuat daftar artikel untuk dilihat, kita akan memuat tajuk setiap artikel ke dalam kelas PHP `tree_node`. Setiap `tree_node` akan berisi tajuk artikel dan serangkaian balasan untuk artikel tersebut.

Balasan akan disimpan dalam array. Setiap balasan akan menjadi `tree_node`, yang dapat berisi array balasan untuk artikel tersebut, yang merupakan `tree_node` itu sendiri, dan seterusnya. Ini berlanjut hingga kita mencapai apa yang disebut node daun dari pohon, node yang tidak memiliki balasan apa pun. Kita kemudian akan memiliki struktur pohon yang tampak seperti yang ada pada Gambar 8.1.

Beberapa terminologi: Pesan yang kita balas dapat disebut node induk dari node saat ini. Balasan apa pun terhadap pesan tersebut dapat disebut anak dari node saat ini. Jika Anda membayangkan bahwa struktur pohon ini seperti pohon keluarga, ini akan mudah diingat.

Artikel pertama dalam struktur pohon ini—yang tidak memiliki induk—kadang-kadang disebut node akar.

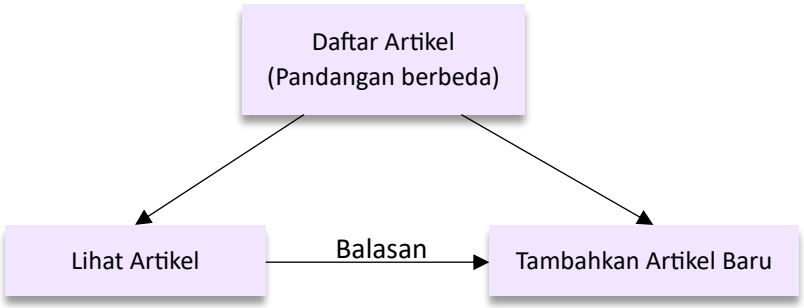
Kita memutuskan untuk menggunakan kelas untuk struktur ini karena ini adalah cara termudah untuk membangun struktur data yang kompleks dan berkembang secara dinamis untuk aplikasi ini. Ini juga berarti kita memiliki kode yang cukup sederhana dan elegan untuk melakukan sesuatu yang cukup rumit.

Ikhtisar Solusi

Untuk benar-benar memahami apa yang telah kita lakukan dengan proyek ini, mungkin ada baiknya untuk mengerjakan kode tersebut, yang akan kita lakukan sebentar lagi. Aplikasi ini tidak terlalu rumit dibandingkan dengan beberapa aplikasi lainnya, tetapi kodenya sedikit lebih rumit.

Hanya ada tiga halaman nyata dalam aplikasi ini.

Kita akan memiliki halaman indeks utama yang menampilkan semua artikel di forum sebagai tautan ke artikel tersebut. Dari sini, Anda akan dapat menambahkan artikel baru, melihat artikel yang tercantum, atau mengubah cara artikel dilihat dengan memperluas dan menciutkan cabang pohon. (Lebih lanjut tentang ini sebentar lagi.) Diagram aliran sistem ditunjukkan pada Gambar 8.2.



Gambar 8.2 Ada tiga bagian utama dari sistem forum blah-blah.

Ringkasan berkas dalam aplikasi ini ditunjukkan pada Tabel 8.1.

Tabel 8.1 Berkas dalam Aplikasi Forum Web

Nama	Type	Keterangan
`index.php`	Aplikasi	Halaman utama yang akan dilihat pengguna saat memasuki situs. Berisi daftar semua artikel di situs yang dapat diperluas dan diciutkan.
`new_post.php`	Aplikasi	Formulir yang digunakan untuk memposting artikel baru.
`store_new_post.php`	Aplikasi	Menyimpan artikel yang dimasukkan dalam formulir new_post.php.
`view_post.php`	Aplikasi	Menampilkan postingan individual dan daftar balasan terhadap postingan tersebut.
`treenode_class.php`	Library	Berisi kelas treenode, yang akan kita gunakan untuk menampilkan hierarki postingan.

include_fns.php	Library	Mengumpulkan semua pustaka fungsi lain untuk aplikasi ini (file jenis Pustaka lain yang tercantum di sini).
data_valid_fns.php	Library	Fungsi validasi data.
db_fns.php	Library	Fungsi konektivitas basis data.
discussion_fns.php	Library	Fungsi untuk menangani penyimpanan dan pengambilan postingan.
output_fns.php	Library	Fungsi untuk mengeluarkan HTML.
create_database.sql	SQL	SQL untuk menyiapkan basis data yang diperlukan untuk aplikasi ini.

Mari kita lanjutkan dan lihat implementasinya.

8.2 BASIS DATA FORUM WEB

Ada beberapa atribut yang perlu kita simpan tentang setiap artikel yang diposting ke forum: orang yang menulisnya, yang disebut poster; judul artikel; kapan artikel tersebut diposting; dan isi artikel. Oleh karena itu, kita memerlukan tabel artikel. Kita akan membuat ID unik untuk setiap artikel, yang disebut `postid`.

Setiap artikel perlu memiliki beberapa informasi tentang tempatnya dalam hierarki. Kita dapat menyimpan informasi tentang anak-anak artikel dengan artikel tersebut. Namun, setiap artikel dapat memiliki banyak balasan, sehingga hal ini dapat menyebabkan beberapa masalah dalam konstruksi basis data. Karena setiap artikel hanya dapat menjadi balasan terhadap satu artikel lainnya, lebih mudah untuk menyimpan referensi ke artikel induk, yaitu, artikel yang dibalas oleh artikel ini. Hal ini memberi kita data berikut untuk disimpan untuk setiap artikel:

- `postid`: ID unik untuk setiap artikel
- `parent`: Postid dari artikel induk
- `poster`: Penulis artikel ini
- `title`: Judul artikel ini
- `posted`: Tanggal dan waktu artikel tersebut diposting
- `message`: Isi artikel

Kita akan menambahkan beberapa pengoptimalan untuk ini.

Saat kita mencoba menentukan apakah sebuah artikel memiliki balasan, kita harus menjalankan kueri untuk melihat apakah ada artikel lain yang memiliki artikel ini sebagai induk. Kita akan memerlukan informasi ini untuk setiap posting yang kita daftarkan. Semakin sedikit kueri yang harus kita jalankan, semakin cepat kode kita akan berjalan. Kita dapat menghilangkan kebutuhan untuk kueri ini dengan menambahkan kolom untuk menunjukkan apakah ada balasan. Kita akan menyebut kolom ini children dan menjadikannya Boolean secara efektif—nilainya akan menjadi 1 jika node memiliki children, dan 0 jika tidak.

Selalu ada harga yang harus dibayar untuk pengoptimalan. Di sini kita memilih untuk menyimpan data yang berlebihan. Karena kita menyimpan data dalam dua cara, kita harus berhati-hati untuk memastikan bahwa kedua representasi tersebut saling cocok. Ketika kita

menambahkan anak, kita harus memperbarui induknya. Jika kita mengizinkan penghapusan anak, kita perlu memperbarui simpul induk untuk memastikan bahwa basis data tersebut konsisten. Dalam proyek ini, kita tidak akan membangun fasilitas untuk menghapus artikel, jadi kita akan terhindar dari separuh masalah ini. Jika Anda memutuskan untuk memperluas kode ini, ingatlah masalah ini.

Perlu dicatat bahwa beberapa basis data akan sedikit lebih membantu kita di sini. Jika kita menggunakan Oracle, ia dapat mempertahankan integritas relasional untuk kita. Dengan menggunakan MySQL, yang tidak mendukung pemicu atau batasan kunci asing, kita perlu menulis pemeriksaan dan penyeimbangan kita sendiri untuk memastikan bahwa data masih masuk akal setiap kali kita menambahkan atau menghapus rekaman.

Kita akan melakukan satu pengoptimalan lain: Kita akan memisahkan isi pesan dari data lain dan menyimpannya dalam tabel terpisah. Alasannya adalah karena atribut ini akan memiliki tipe MySQL text. Memiliki tipe ini dalam tabel dapat memperlambat kueri pada tabel tersebut. Karena kita akan melakukan banyak kueri kecil untuk membangun struktur pohon, ini akan memperlambatnya cukup banyak. Dengan isi pesan dalam tabel terpisah, kita dapat mengambilnya saat pengguna ingin melihat pesan tertentu.

MySQL dapat mencari catatan berukuran tetap lebih cepat daripada catatan berukuran variabel. Jika kita perlu menggunakan data berukuran variabel, kita dapat membantu dengan membuat indeks pada bidang yang akan digunakan untuk mencari basis data. Untuk beberapa proyek, sebaiknya kita membiarkan bidang teks dalam catatan yang sama dengan yang lainnya dan menentukan indeks pada semua kolom yang akan kita cari.

Namun, indeks memerlukan waktu untuk dibuat, dan data dalam forum kita kemungkinan akan berubah sepanjang waktu, jadi kita perlu membuat ulang indeks kita secara berkala. Kami juga akan menambahkan atribut area jika nanti kami memutuskan untuk mengimplementasikan beberapa obrolan dengan satu aplikasi. Kami tidak akan mengimplementasikannya di sini, tetapi dengan cara ini area tersebut akan dicadangkan untuk penggunaan di masa mendatang.

Dengan mempertimbangkan semua hal ini, SQL untuk membuat basis data bagi basis data forum ditunjukkan pada Listing 8.1.

Listing 8.1 `create_database.sql`–SQL untuk Membuat Basis Data Diskusi

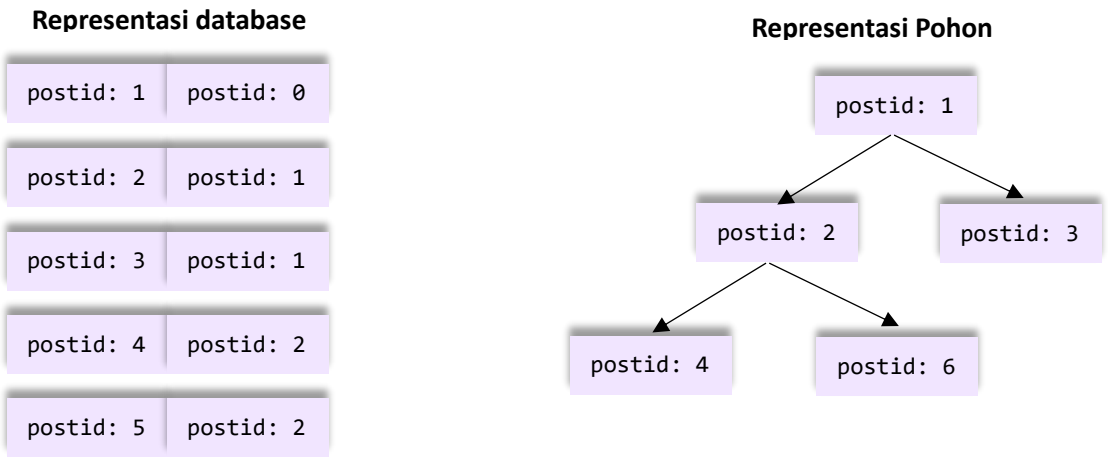
```
create database discussion;

use discussion;

create table header
(
  parent int not null,
  poster char(20) not null,
  title char(20) not null,
  children int default 0 not null,
  area int default 1 not null,
  posted datetime not null,
```

```
  postid int unsigned not null auto_increment primary key
);

create table body
(
  postid int unsigned not null primary key,
  message text
);

grant select, insert, update, delete
on discussion.*
to discussion@localhost identified by 'password';
```

Anda dapat membuat struktur basis data ini dengan menjalankan skrip ini melalui MySQL sebagai berikut:

```
mysql -u root -p < create_database.sql
```

Anda perlu memberikan kata sandi root. Anda mungkin juga harus mengubah kata sandi yang telah kita buat untuk pengguna diskusi menjadi sesuatu yang lebih baik. Untuk memahami bagaimana struktur ini akan menampung artikel dan hubungan antarartikel tersebut, lihat Gambar 8.3.



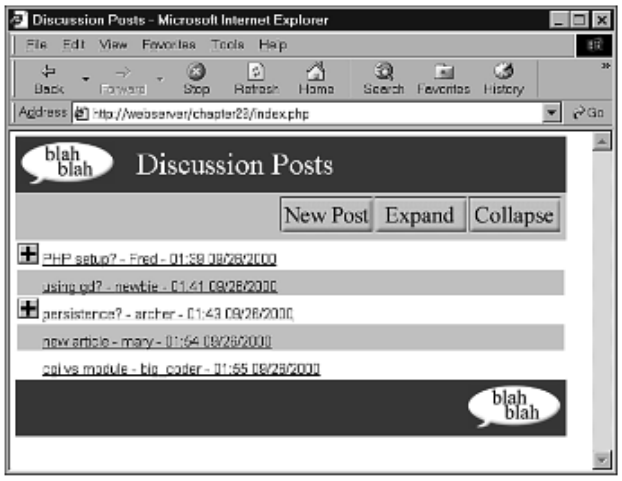
Gambar 8.3 Basis data menyimpan struktur pohon dalam bentuk relasional yang diratakan.

Seperti yang dapat Anda lihat, kolom induk untuk setiap artikel dalam basis data menyimpan postid artikel di atasnya dalam pohon. Artikel induk adalah artikel yang dibalas. Anda juga dapat melihat bahwa simpul akar, `postid 1`, tidak memiliki induk. Semua topik diskusi baru akan berada di posisi ini. Untuk artikel jenis ini, kami menyimpan induknya sebagai 0 (nol) dalam basis data.

Melihat Pohon Artikel

Selanjutnya, kami memerlukan cara untuk mendapatkan informasi dari basis data dan menyajikannya kembali dalam struktur pohon. Kami akan melakukannya dengan halaman

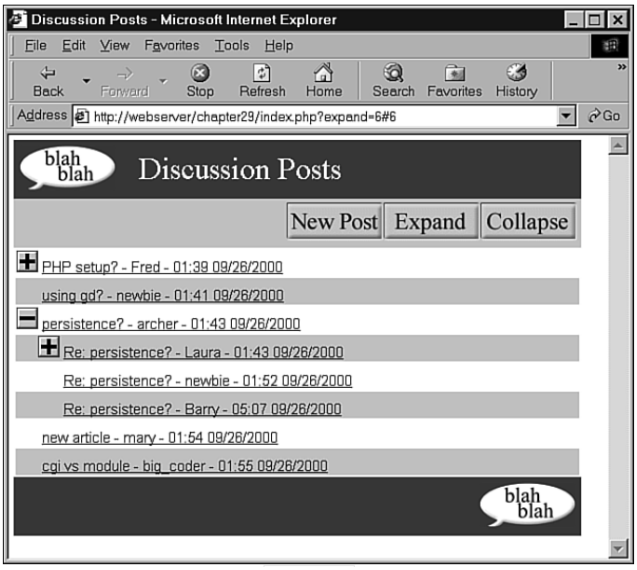
utama, `index.php`. Untuk tujuan penjelasan ini, kami telah memasukkan beberapa contoh posting melalui skrip posting artikel `new_post.php` dan `store_new_post.php`. Kami akan membahasnya di bagian berikutnya. Kami akan membahas daftar artikel terlebih dahulu karena merupakan tulang punggung situs. Setelah ini, semua hal lainnya akan mudah.
Gambar 8.4 menunjukkan tampilan awal artikel di situs yang akan dilihat pengguna.



Gambar 8.4 Tampilan awal daftar artikel menunjukkan artikel dalam bentuk "diciutkan".

Yang kita lihat di sini adalah semua artikel awal. Tidak ada satu pun yang merupakan balasan; semuanya adalah artikel pertama tentang topik tertentu. Anda akan melihat bahwa kita memiliki sejumlah opsi. Ada bilah menu yang memungkinkan kita menambahkan posting baru dan memperluas atau menciutkan tampilan artikel yang kita miliki.

Untuk memahami apa artinya ini, lihat posting. Beberapa di antaranya memiliki simbol plus di sebelahnya. Ini berarti bahwa artikel ini telah dibalas. Untuk melihat balasan ke artikel tertentu, Anda dapat mengeklik simbol plus. Hasil dari mengeklik salah satu simbol ini ditunjukkan pada Gambar 8.5.

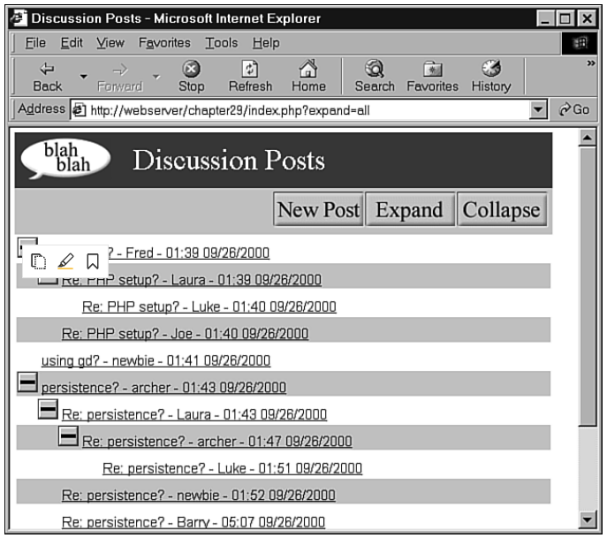


Gambar 8.5 Rangkaian diskusi tentang persistensi telah diperluas.

Seperti yang Anda lihat, mengeklik simbol plus telah menampilkan balasan ke artikel pertama tersebut. Simbol plus kini telah berubah menjadi simbol minus. Jika kita mengeklik ini, semua artikel di utas ini akan diciutkan, mengembalikan kita ke tampilan awal.

Anda mungkin juga memperhatikan bahwa salah satu balasan memiliki simbol plus di sebelahnya. Ini berarti ada balasan untuk balasan ini. Ini dapat berlanjut hingga kedalaman yang sewenang-wenang, dan Anda dapat melihat setiap set balasan dengan mengeklik simbol plus yang sesuai.

Dua opsi bilah menu, Perluas dan Ciutkan, masing-masing akan memperluas semua utas yang mungkin dan menciutkan semua utas yang mungkin. Hasil dari mengklik tombol Perluas ditunjukkan pada Gambar 8.6.



Gambar 8.6 Semua rangkaian diskusi kini telah diperluas.

Jika Anda perhatikan Gambar 8.5 dan 8.6 dengan saksama, Anda dapat melihat bahwa kami meneruskan beberapa parameter kembali ke index.php di baris perintah

Memperluas dan Menciutkan

Mari kita lihat bagaimana hal itu dilakukan dengan melihat skrip index.php, yang ditunjukkan pada Listing 8.2.

Listing 8.2 `index.php`–Skrip untuk Membuat Tampilan Artikel di Halaman Utama Aplikasi

```
<?
  include ('include_fns.php');
  session_start();

  // check if we have created our session variable
  if(!session_is_registered('expanded'))
  {
    $expanded = array();
    session_register('expanded');
  }
```

```
  // check if an expand button was pressed
  // expand might equal 'all' or a postid or not be set
  if($expand)
  {
    if($expand == 'all')
      expand_all($expanded);
    else
      $expanded[$expand] = true;
  }

  // check if a collapse button was pressed
  // collapse might equal all or a postid or not be set
  if($collapse)
  {
    if($collapse=="all")
      unset($expanded);
    else
      unset($expanded[$collapse]);
  }

  do_html_header("Discussion Posts");

  display_index_toolbar();

  // display the tree view of conversations
  display_tree($expanded);

  do_html_footer();
?>
```

Skrip ini menggunakan tiga variabel untuk menjalankan tugasnya. Variabel tersebut adalah:

- Variabel sesi $expanded, yang melacak utas mana yang diekspansi. Variabel ini dapat dipertahankan dari tampilan ke tampilan, sehingga kita dapat memiliki beberapa utas yang diekspansi. Variabel ini adalah array asosiatif yang berisi `postid` artikel yang balasannya akan ditampilkan dalam bentuk ekspansi.
- Parameter `$expand`, yang memberi tahu skrip utas baru mana yang akan diekspansi.
- Parameter `$collapse`, yang memberi tahu skrip utas mana yang akan diciutkan.

Saat kita mengklik simbol plus atau minus atau tombol Expand atau `Collapse`, keduanya akan memanggil ulang skrip index.php dengan parameter baru untuk $expand atau `$collapse`. Kita menggunakan $expanded dari halaman ke halaman untuk melacak utas mana yang harus diekspansi dalam tampilan apa pun.

Skrip dimulai dengan memulai sesi dan menambahkan variabel $expanded sebagai variabel sesi jika hal ini belum dilakukan. Setelah itu, skrip memeriksa apakah parameter $expand atau `$collapse` telah diberikan dan memodifikasi array $expanded sebagaimana mestinya. Lihat kode untuk parameter `$expand`:

```
if($expand)
{
  if($expand == 'all')
    expand_all($expanded);
  else
    $expanded[$expand] = true;
}
```

Jika kita mengklik tombol Expand, fungsi expand_all() dipanggil untuk menambahkan semua thread yang memiliki balasan ke dalam array $expanded. (Kita akan membahasnya sebentar lagi.) Jika kita mencoba untuk memperluas thread tertentu, kita akan diberikan postid melalui $expand. Oleh karena itu, kita menambahkan entri baru ke array $expanded untuk mencerminkan hal ini. Fungsi expand_all() ditunjukkan pada Listing 8.3.

Listing 8.3 Fungsi expand_all() dari discussion_fns.php— Memproses Array $expanded untuk Memperluas Semua Thread di Forum

```
function expand_all(&$expanded)
{
  // mark all threads with children as to be shown expanded
  $conn = db_connect();
  $query = "select postid from header where children = 1";
  $result = mysql_query($query);
  $num = mysql_numrows($result);
  for($i = 0; $i<$num; $i++)
  {
    $expanded[mysql_result($result, $i, 0)]=true;
  }
}
```

Fungsi ini menjalankan kueri basis data untuk mencari tahu utas mana di forum yang memiliki balasan, sebagai berikut:

```
select postid from header where children = 1
```

Masing-masing artikel yang dikembalikan kemudian ditambahkan ke array $expanded. Kita menjalankan kueri ini untuk menghemat waktu nanti. Kita dapat menambahkan semua artikel ke daftar yang diperluas, tetapi akan sia-sia jika mencoba memproses balasan yang tidak ada. Menciutkan artikel berfungsi dengan cara yang serupa tetapi berlawanan, sebagai berikut:

```
if($collapse)
{
  if($collapse=="all")
    unset($expanded);
```

```
  else
  unset($expanded[$collapse]);
}
```

Anda dapat menghapus item dari array $expanded dengan membatalkan pengaturannya. Kami menghapus utas yang akan diciutkan, atau membatalkan pengaturan seluruh array jika seluruh halaman akan diciutkan.

Semua ini adalah praproses, jadi kami tahu artikel mana yang harus ditampilkan dan mana yang tidak. Bagian utama skrip adalah panggilan ke `display_tree($expanded);` yang sebenarnya akan menghasilkan pohon artikel yang ditampilkan.

8.3 MENAMPILKAN ARTIKEL

Mari kita lihat fungsi `display_tree()`, yang ditunjukkan pada Listing 8.4.

Listing 8.4 Fungsi `display_tree()` dari `output_fns.php`—Membuat Node Root dari Struktur Pohon

```
function display_tree($expanded, $row = 0, $start = 0)
{
  // display the tree view of conversations

  global $table_width;
  echo “<table width = $table_width>”;

  // see if we are displaying the whole list or a sublist
  if($start>0)
    $sublist = true;
  else
    $sublist = false;

  // construct tree structure to represent conversation summary
  $tree = new treenode($start, ‘’, ‘’, ‘’, 1, true, -1, $expanded, $sublist);

  // tell tree to display itself
  $tree->display($row, $sublist);

  echo “</table>”;
}
```

Peran utama fungsi ini adalah membuat simpul akar dari struktur pohon. Kita menggunakannya untuk menampilkan seluruh indeks dan membuat subpohon balasan pada halaman `view_post.php`. Seperti yang Anda lihat, fungsi ini memerlukan tiga parameter. Yang pertama, `$expanded`, adalah daftar `postid` artikel yang akan ditampilkan dalam bentuk yang diperluas. Yang kedua, `$row`, adalah indikator nomor baris yang akan digunakan untuk menentukan warna bergantian dari baris dalam daftar.

Parameter ketiga, `$start`, memberi tahu fungsi di mana harus mulai menampilkan artikel. Ini adalah `postid` dari simpul akar agar pohon tersebut dibuat dan ditampilkan. Jika kita menampilkan semuanya, seperti saat kita berada di halaman utama, ini akan menjadi 0 (nol), yang berarti menampilkan semua artikel tanpa induk. Jika parameter ini adalah 0, kita tetapkan `$sublist` ke false dan tampilkan seluruh pohon.

Jika parameter lebih besar dari 0, kita gunakan sebagai simpul akar pohon yang akan ditampilkan, tetapkan `$sublist` ke true dan buat serta tampilkan hanya sebagian dari pohon tersebut. (Kita akan menggunakan ini dalam skrip `view_post.php`.)

Hal terpenting yang dilakukan fungsi ini adalah membuat instance dari kelas treenode yang mewakili akar pohon. Ini sebenarnya bukan artikel tetapi berfungsi sebagai induk dari semua artikel tingkat pertama, yang tidak memiliki induk. Setelah pohon dibangun, kita cukup memanggil fungsi display-nya untuk benar-benar menampilkan daftar artikel.

Menggunakan Kelas treenode

Kode untuk kelas treenode ditampilkan dalam Listing 8.5.

Listing 8.5 Kelas treenode dari treenode_class.php—Tulang Punggung Aplikasi

```
<?
// functions for loading, contructing and
// displaying the tree are in this file
class treenode
{
  // each node in the tree has member variables containing
  // all the data for a post except the body of the message
  var $m_postid;
  var $m_title;
  var $m_poster;
  var $m_posted;
  var $m_children;
  var $m_childlist;
  var $m_depth;

  function treenode($postid, $title, $poster, $posted, $children,
                $expand, $depth, $expanded, $sublist)
  {
    // the constructor sets up the member variables, but more
    // importantly recursively creates lower parts of the tree
    $this->m_postid = $postid;
    $this->m_title = $title;
    $this->m_poster = $poster;
    $this->m_posted = $posted;
    $this->m_children =$children;
    $this->m_childlist = array();
    $this->m_depth = $depth;

    // we only care what is below this node if it
```

```
    // has children and is marked to be expanded
    // sublists are always expanded
    if(($sublist||$expand) && $children)
    {
      $conn = db_connect();
      $query = "select * from header where parent = $postid order by posted";
      $result = mysql_query($query);

      for ($count=0; $row = @mysql_fetch_array($result); $count++)
      {
        if($sublist||$expanded[ $row['postid'] ] == true)
          $expand = true;
        else
          $expand = false;
          $this->m_childlist[$count]= new treenode($row['postid'],
          $row['title'], $row['poster'], $row['posted'], $row['children'],
          $expand,
          $depth+1, $expanded, $sublist);
        }
      }
    }

  function display($row, $sublist = false)
  {
    // as this is an object, it is responsible for displaying itself

    // $row tells us what row of the display we are up to
    // so we know what color it should be

    // $sublist tells us whether we are on the main page
    // or the message page. Message pages should have
    // $sublist = true.
    // On a sublist, all messages are expanded and there are
    // no "+" or "-" symbols.

    // if this is the empty root node skip displaying
    if($this->m_depth>-1)
    {
      //color alternate rows
      echo "<tr><td bgcolor = ";
      if ($row%2)
        echo "'#cccccc'>";
      else
        echo "'#ffffff'>";

      // indent replies to the depth of nesting
      for($i = 0; $i<$this->m_depth; $i++)
```

```
        {
          echo “<img src = ‘images/spacer.gif’ height = 22
                        width = 22 alt = ‘’ valign = bottom>”;
        }

      // display + or - or a spacer
      if ( !$sublist && $this->m_children && sizeof($this->m_childlist))
      // we’re on the main page, have some children, and they’re expanded
      {
        // we are expanded - offer button to collapse
        echo “<a href = ‘index.php?collapse=”.
                        $this->m_postid.”#$this->m_postid’
             ><img src = ‘images/minus.gif’ valign = bottom
             height = 22 width = 22 alt = ‘Collapse Thread’ border = 0></a>”;
      }
      else if(!$sublist && $this->m_children)
      {
        // we are collapsed - offer button to expand
        echo “<a href = ‘index.php?expand=”.
          $this->m_postid.”#$this->m_postid’><img src = ‘images/plus.gif’
          height = 22 width = 22 alt = ‘Expand Thread’ border = 0></a>”;
    }
    else
    {
      // we have no children, or are in a sublist, do not give button
      echo “<img src = ‘images/spacer.gif’ height = 22 width = 22
      alt = ‘’valign = bottom>”;
    }
    echo “ <a name = $this->m_postid ><a href =
           ‘view_post.php?postid=$this->m_postid’>$this->m_title -
           $this->m_poster - “.reformat_date($this->m_posted).”</a>”;
           echo “</td></tr>”;

      // increment row counter to alternate colors
      $row++;
    }
    // call display on each of this node’s children
    // note a node will only have children in its list if expanded
    $num_children = sizeof($this->m_childlist);
    for($i = 0; $i<$num_children; $i++)
    {
      $row = $this->m_childlist[$i]->display($row, $sublist);
    }
    return $row;
    }
};
?>
```

Kelas ini berisi fungsionalitas yang menggerakkan tampilan pohon dalam aplikasi ini. Satu contoh kelas treenode berisi detail tentang satu posting dan tautan ke semua posting balasan dari kelas tersebut. Ini memberi kita variabel anggota berikut:

```
var $m_postid;
var $m_title;
var $m_poster;
var $m_posted;
var $m_children;
var $m_childlist;
var $m_depth;
```

Perhatikan bahwa treenode tidak memuat isi artikel. Tidak perlu memuat ini hingga pengguna membuka skrip view_post.php. Kita perlu mencoba membuatnya relatif cepat, karena kita melakukan banyak manipulasi data untuk menampilkan daftar pohon, dan perlu menghitung ulang saat halaman di-refresh, atau tombol ditekan.

Skema penamaan untuk variabel-variabel ini mengikuti skema penamaan yang umum digunakan dalam aplikasi OO—variabel dimulai dengan m_ untuk mengingatkan kita bahwa variabel tersebut adalah variabel anggota kelas.
Sebagian besar variabel ini berkorespondensi langsung dengan baris dari tabel header di basis data kita.

Pengecualian adalah $m_childlist dan $m_depth. Kita akan menggunakan variabel $m_childlist untuk menyimpan balasan untuk artikel ini. Variabel $m_depth akan menyimpan jumlah tingkat pohon yang kita turunkan—ini akan digunakan untuk membuat tampilan. Fungsi konstruktor menyiapkan nilai semua variabel, sebagai berikut:

```
function treenode ($postid, $title, $poster, $posted, $children,
                   $expand, $depth, $expanded, $sublist)
{
  // the constructor sets up the member variables, but more
  // importantly recursively creates lower parts of the tree
  $this->m_postid = $postid;
  $this->m_title = $title;
  $this->m_poster = $poster;
  $this->m_posted = $posted;
  $this->m_children =$children;
  $this->m_childlist = array();
  $this->m_depth = $depth;
```

Ketika kita membangun root treenode dari display_tree() dari halaman utama, kita sebenarnya membuat node -dummy tanpa artikel yang terkait dengannya. Kita memasukkan beberapa nilai awal:

```
$tree = new treenode($start, '', '', '', 1, true, -1, $expanded, $sublist);
```

Ini menciptakan simpul akar dengan `$postid` nol. Ini dapat digunakan untuk menemukan semua posting tingkat pertama karena mereka memiliki induk nol. Kami menetapkan kedalaman ke -1 karena simpul ini sebenarnya bukan bagian dari tampilan. Semua posting tingkat pertama akan memiliki kedalaman nol, dan berada di ujung kiri layar. Kedalaman berikutnya melangkah ke kanan.

Hal terpenting yang terjadi dalam konstruktor ini adalah bahwa simpul anak dari simpul ini diwujudkan. Kami memulai proses ini dengan memeriksa apakah kami perlu memperluas simpul anak. Kami hanya melakukan proses ini jika sebuah simpul memiliki beberapa anak, dan kami telah memilih untuk menampilkannya:

```
if(($sublist||$expand) && $children)
{
  $conn = db_connect();
```

Kemudian kita terhubung ke database, dan mengambil semua postingan anak, sebagai berikut:

```
$query = "select * from header where parent = $postid order by posted";
$result = mysql_query($query);
```

Kita kemudian mengisi array `$m_childlist` dengan instance dari kelas treenode, yang memuat balasan terhadap posting yang disimpan dalam treenode ini, sebagai berikut:

```
for ($count=0; $row = @mysql_fetch_array($result); $count++)
{
  if($sublist||$expanded[ $row['postid'] ] == true)
    $expand = true;
  else
    $expand = false;
  $this->m_childlist[$count]= new treenode($row['postid'],$row['title'],
                              $row['poster'],$row['posted'],
                              $row['children'], $expand,
                              $depth+1, $expanded, $sublist);
}
```

Baris terakhir ini akan membuat simpul pohon baru, mengikuti proses yang sama persis dengan yang baru saja kita lalui, tetapi untuk level berikutnya di bawah pohon. Ini adalah bagian rekursif. Simpul pohon induk memanggil konstruktor simpul pohon, meneruskan postid-nya sendiri sebagai induk, dan menambahkan satu ke kedalamannya sendiri sebelum meneruskannya.

Setiap simpul pohon pada gilirannya akan dibuat dan membuat anak-anaknya sendiri hingga kita kehabisan balasan atau level yang ingin kita perluas. Setelah semua itu selesai, kita panggil fungsi display dari `root treenode` (ini kembali ke `display_tree()`), sebagai berikut:

```
$tree->display($row, $sublist);
```

Fungsi `display()` dimulai dengan memeriksa apakah ini adalah root node tiruan:

```
if($this->m_depth>-1)
```

Dengan cara ini, tiruan tersebut dapat ditinggalkan dari display. Kita tidak ingin melewatkan root node sepenuhnya. Kita tidak ingin simpul tersebut muncul, tetapi simpul tersebut perlu memberi tahu anak-anaknya bahwa simpul tersebut perlu menampilkan diri mereka sendiri.

Fungsi tersebut kemudian mulai menggambar tabel yang berisi artikel. Fungsi tersebut menggunakan operator modulus (%) untuk memutuskan warna latar belakang apa yang harus dimiliki baris ini (sehingga baris tersebut bergantian):

```
//color alternate rows
echo “<tr><td bgcolor = “;if ($row%2)
  echo “‘#cccccc’>”;
else
  echo “‘#ffffff’>”;
```

Kemudian variabel anggota `$m_depth` digunakan untuk menghitung seberapa banyak indentasi item saat ini. Anda akan melihat dengan melihat kembali angka-angka bahwa semakin dalam balasan berada, semakin jauh indentasinya. Ini dilakukan sebagai berikut:

```
// indent replies to the depth of nesting
for($i = 0; $i<$this->m_depth; $i++)
{
  echo “<img src = ‘images/spacer.gif’ height = 22
                width = 22 alt = ‘’ valign = bottom>”;
}
```

Bagian fungsi selanjutnya menentukan apakah akan menyediakan tombol plus atau minus atau tidak menyediakan sama sekali:

```
// display + or - or a spacer
if ( !$sublist && $this->m_children && sizeof($this->m_childlist))
// we’re on the main page, have some children, and they’re expanded
{
  // we are expanded - offer button to collapse
  echo “<a href = ‘index.php?collapse=”.
                $this->m_postid.”#$this->m_postid’
       ><img src = ‘images/minus.gif’ valign = bottom
       height = 22 width = 22 alt = ‘Collapse Thread’ border = 0></a>”;
  }
  else if(!$sublist && $this->m_children)
```

```
  {

  // we are collapsed - offer button to expand
  echo "<a href = 'index.php?expand=".
       $this->m_postid."#$this->m_postid'><img src = 'images/plus.gif'
       height = 22 width = 22 alt = 'Expand Thread' border = 0></a>";
}
else
{
  // we have no children, or are in a sublist, do not give button
  echo "<img src = 'images/spacer.gif' height = 22 width = 22
            alt = ''valign = bottom>";
}
```

Berikutnya, kami menampilkan detail sebenarnya dari node ini:

```
echo " <a name = $this->m_postid ><a href =
       'view_post.php?postid=$this->m_postid'>$this->m_title -
       $this->m_poster - ".reformat_date($this->m_posted)."</a>";
echo "</td></tr>";
```

Kita mengubah warna untuk baris berikutnya:

```
// increment row counter to alternate colors
$row++;
```

Setelah itu, ada beberapa kode yang akan dieksekusi oleh semua treenode, termasuk yang root, sebagai berikut:

```
// call display on each of this node's children
// note a node will only have children in its list if expanded
$num_children = sizeof($this->m_childlist);
for($i = 0; $i<$num_children; $i++)
{
  $row = $this->m_childlist[$i]->display($row, $sublist);
}
return $row;
```

Sekali lagi ini adalah panggilan fungsi rekursif, yang memanggil setiap anak simpul ini untuk menampilkan diri mereka sendiri. Kami meneruskan warna baris saat ini kepada mereka dan meminta mereka untuk meneruskannya kembali setelah mereka selesai menggunakannya, sehingga kami dapat melacak warna bergantian. Itu saja untuk kelas ini. Kodenya cukup rumit. Anda mungkin ingin bereksperimen dengan menjalankan aplikasi dan kemudian kembali untuk melihatnya lagi saat Anda merasa nyaman dengan apa yang dilakukannya.

Melihat Artikel Perorangan

Panggilan `display_tree()` akhirnya memberi kita tautan ke sekumpulan artikel. Jika kita mengklik salah satu artikel ini, kita akan masuk ke skrip `view_post.php`, dengan parameter `postid` artikel yang akan dilihat. Contoh keluaran dari skrip ini ditunjukkan pada Gambar 8.7.

Skrip ini menunjukkan isi pesan, serta balasan untuk pesan ini. Anda akan melihat bahwa balasan ditampilkan lagi sebagai pohon, tetapi kali ini diperluas sepenuhnya, dan tanpa tombol plus atau minus. Ini adalah efek dari sakelar `$sublist` yang mulai beraksi. Mari kita lihat kode untuk view_post.php, yang ditunjukkan pada Listing 8.6.

Gambar 8.7 Sekarang kita dapat melihat isi pesan untuk posting ini.

Listing 8.6 view_post.php—Menampilkan Satu Isi Pesan

```
<?
  // include function libraries
  include ('include_fns.php');

  // get post details
  $post = get_post($postid);

  do_html_header($post["title"]);

  // display post
  display_post($post);

  // if post has any replies, show the tree view of them
  if($post['children'])
```

```
  {
    echo “<br><br>”;
    display_replies_line();
    display_tree($expanded, 0, $postid);
  }

  do_html_footer();
?>
```

Skrip ini menggunakan tiga fungsi utama untuk melakukan tugasnya: `get_post()`, `display_post()`, dan `display_tree()`. Fungsi `get_post()` menarik detail fungsi dari database. Kode untuk fungsi ini ditunjukkan pada Listing 8.7.

Listing 8.7 Fungsi `get_post()` dari `discussion_fns.php`—Mengambil Pesan dari Database

```
function get_post($postid)
{
  // extract one post from the database and return as an array

  if(!$postid) return false;

  $conn = db_connect();

  //get all header information from ‘header’
  $query = “select * from header where postid = $postid”;
  $result = mysql_query($query);
  if(mysql_numrows($result)!=1)
    return false;
  $post = mysql_fetch_array($result);

  // get message from body and add it to the previous result
  $query = “select * from body where postid = $postid”;
  $result2 = mysql_query($query);
  if(mysql_numrows($result2)>0)
  {
    $body = mysql_fetch_array($result2);
    if($body)
    {
      $post[‘message’] = $body[‘message’];
    }
  }
  return $post;
}
```

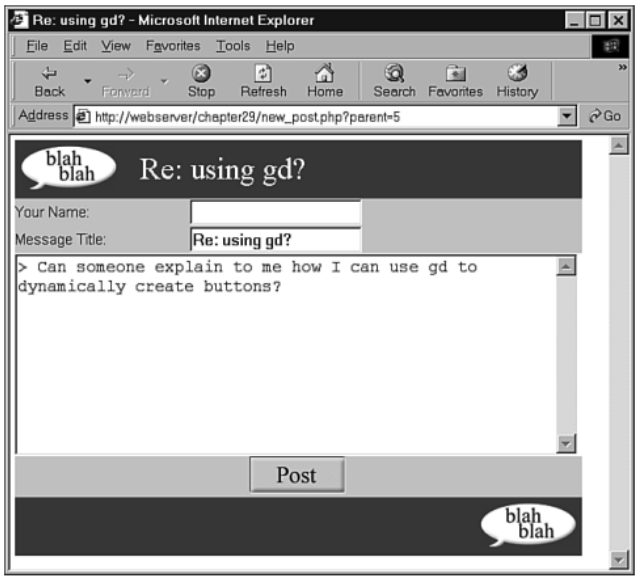
Fungsi ini, dengan postid, akan menjalankan dua kueri yang diperlukan untuk mengambil tajuk dan isi pesan untuk kiriman tersebut, dan menyatukannya ke dalam satu larik asosiatif yang kemudian dikembalikan.

Hasil fungsi ini kemudian diteruskan ke fungsi `display_post()` dari `output_fns.php`. Ini hanya mencetak larik dengan beberapa format HTML, jadi kami tidak menyertakannya di sini. Terakhir, skrip `view_post.php` memeriksa apakah ada balasan untuk artikel ini dan memanggil `display_tree()` untuk menampilkannya dalam format sublist—yaitu, diperluas sepenuhnya tanpa plus atau minus.

8.4 MENAMBAHKAN ARTIKEL BARU

Setelah semua itu, sekarang kita dapat melihat bagaimana kiriman baru ditambahkan ke forum. Pengguna dapat melakukannya dengan dua cara: pertama, dengan mengklik tombol Kiriman Baru di halaman indeks, dan kedua, dengan mengklik tombol Balas di halaman `view_post.php`.

Kedua tindakan ini mengaktifkan skrip yang sama, `new_post.php`, hanya dengan parameter yang berbeda. Gambar 8.8 menunjukkan output dari `new_post.php` ketika kita telah mencapainya dengan menekan tombol Balas.



Gambar 8.8 Balasan memiliki teks asli yang secara otomatis dimasukkan dan ditandai.

Parameter yang dimasukkan sebagai induk akan menjadi `postid` induk dari posting baru. Jika Anda mengklik Posting Baru alih-alih Balas, Anda akan mendapatkan `parent=0` di URL. Kedua, Anda akan melihat bahwa untuk balasan, teks pesan asli disisipkan dan ditandai dengan karakter "`>`" seperti yang terjadi di sebagian besar program pembaca email dan berita. Ketiga, Anda dapat melihat bahwa judul pesan ini secara default adalah judul pesan asli yang diawali dengan "`Re:`". Mari kita lihat kode yang menghasilkan output ini. Kode ini ditampilkan dalam Listing 8.8.

Listing 8.8 new_post.php—Memungkinkan Pengguna Mengetik Posting Baru atau Membalas Posting yang Ada

```
<?
  include ('include_fns.php');

  if(!$area)
    $area = 1;

  if(!$error)
  {
    if(!$parent)
    {
      $parent = 0;
        if(!$title)
          $title = "New Post";
    }
    else
    {
      // get post name
      $title = get_post_title($parent);

      // append Re:
      if(strstr($title, "Re: ") == false )
        $title = "Re: ".$title;

      //make sure title will still fit in db
      $title = substr($title, 0, 20);

      //prepend a quoting pattern to the post you are replying to
      $message = add_quoting(get_post_message($parent));
    }
  }
  do_html_header("$title");

  display_new_post_form($parent, $area, $title, $message, $name);

  if($error)
    echo "Your message was not stored. Make sure you have filled in all
        fields and try again.";

  do_html_footer();
?>
```

Setelah beberapa pengaturan awal, skrip ini memeriksa apakah induknya nol atau sebaliknya. Jika nol, ini adalah topik baru, dan tidak banyak pekerjaan lebih lanjut yang diperlukan.

Jika ini adalah balasan ($parent adalah postid dari artikel yang sudah ada), maka skrip akan melanjutkan dan menyiapkan judul dan teks pesan asli, sebagai berikut:

```
// get post name
$title = get_post_title($parent);

// append Re:
if(strstr($title, "Re: ") == false )
  $title = "Re: ".$title;
//make sure title will still fit in db
$title = substr($title, 0, 20);
//prepend a quoting pattern to the post you are replying to
$message = add_quoting(get_post_message($parent));
```

Fungsi yang digunakan di sini adalah get_post_title(), get_post_message(), dan add_quoting(). Semua fungsi ini berasal dari pustaka discussion_fns.php. Fungsi-fungsi tersebut ditampilkan dalam Listing 8.9, 29.10, dan 29.11.

Listing 8.9 Fungsi get_post_title() dari discussion_fns.php—Mengambil Judul Pesan dari Basis Data

```
function get_post_title($postid)
{

  // extract one post's name from the database
  if(!$postid) return "";
    $conn = db_connect();

  //get all header information from 'header'
  $query = "select title from header where postid = $postid";
  $result = mysql_query($query);

  if(mysql_numrows($result)!=1)
    return "";
    return mysql_result($result, 0, 0);
}
```

Listing 8.10 Fungsi get_post_message() dari discussion_fns.php—Mengambil Isi Pesan dari Database

```
function get_post_message($postid)
{
  // extract one post's message from the database
  if(!$postid) return "";
    $conn = db_connect();
    $query = "select message from body where postid = $postid";
    $result = mysql_query($query);
```

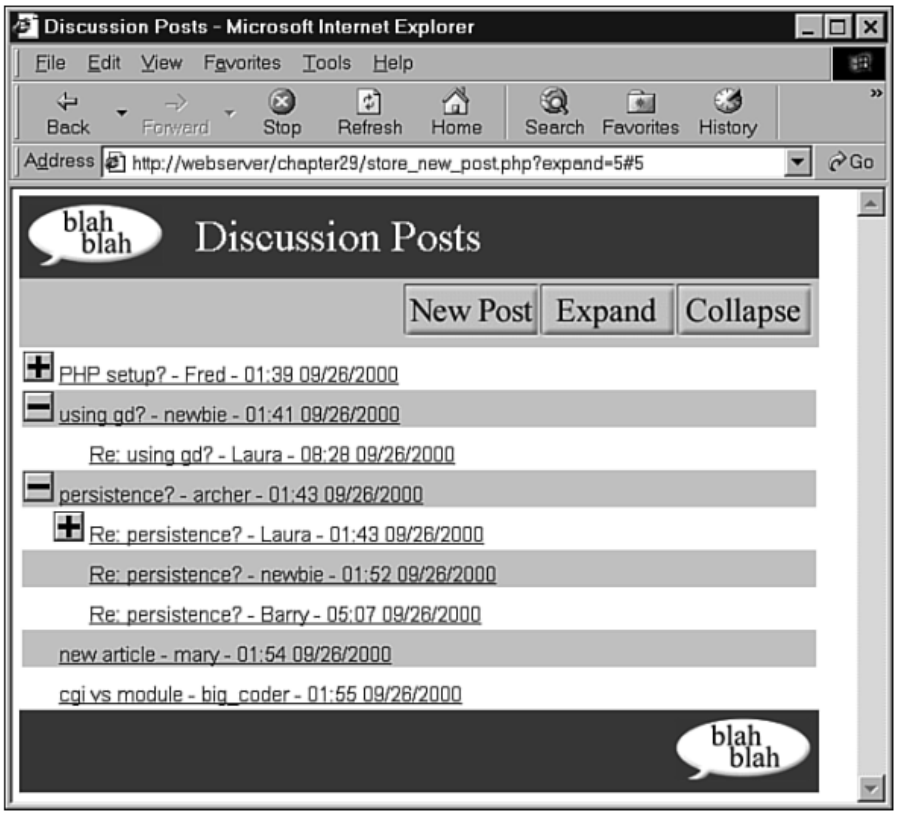
```
  if(mysql_numrows($result)>0)
  {
    return mysql_result($result,0,0);
  }
}
```

Dua fungsi pertama ini mengambil tajuk dan isi artikel (masing-masing) dari basis data. Fungsi `add_quoting()` memformat ulang string untuk memulai setiap baris teks asli dengan simbol, yang defaultnya adalah >.

Listring 8.11 Fungsi `add_quoting()` dari `discussion_fns.php`—Menandai Teks Pesan dengan Simbol ">"

```
function add_quoting($string, $pattern = "> ")
{
  // add a quoting pattern to mark text quoted in your reply
  return $pattern.str_replace("\n", "\n$pattern", $string);
}
```

Setelah pengguna mengetik balasannya dan mengklik tombol Post, ia akan dibawa ke skrip `store_new_post.php`. Contoh keluaran dari skrip ini ditunjukkan pada Gambar 8.9.



Gambar 8.9 Posting baru sekarang terlihat di pohon.

Listing 8.12 store_new_post.php—Menempatkan Posting Baru di Basis Data

```
<?
  include (“include_fns.php”);
  if($id = store_new_post($HTTP_POST_VARS))
  {
    include (“index.php”);
  }
  else
  {
    $error = true;
    include (“new_post.php”);
  }
?>
```

Seperti yang Anda lihat, ini adalah skrip pendek. Tugas utamanya adalah memanggil fungsi store_new_post(). Halaman ini tidak memiliki konten visualnya sendiri. Jika penyimpanan berhasil, kita akan melihat halaman indeks. Jika tidak, kita akan kembali ke halaman new_post.php, sehingga pengguna dapat mencoba lagi. Fungsi store_new_post() ditampilkan dalam Listing 8.13.

Listing 8.13 Fungsi store_new_post() dari discussion_fns.php—Memvalidasi dan Menyimpan Posting Baru dalam Basis Data

```
function store_new_post($post)
{
  // validate clean and store a new post

  $conn = db_connect();
  // check no fields are blank
  if(!filled_out($post))
    return false;

  $post = clean_all($post);

  //check parent exists
  if($post[“parent”]!=0)
  {
    $query=“select postid from header where postid = ‘“.$post[‘parent’].”’”;
    $result = mysql_query($query);
    if(mysql_numrows($result)!=1)
    {
      return false;
    }
  }

  // check not a duplicate
```

```
$query = "select header.postid from header, body where
         header.postid = body.postid and
         header.parent = ".$post['parent']." and
         header.poster = '".$post['poster']."' and
         header.title = '".$post['title']."' and
         header.area = ".$post['area']." And
         body.message = '".$post['message']."'";
$result = mysql_query($query);
if (!$result)
{
  return false;
}
if(mysql_numrows($result)>0)
  return mysql_result($result, 0, 0);
  $query = "insert into header values
           ('".$post['parent']."', '".$post['poster']."',
           '".$post['title']."', 0, '".$post['area']."', now(),
           NULL
           )";
$result = mysql_query($query);
if (!$result)
{
  return false;
}

// note that our parent now has a child
$query = "update header set children = 1 where postid = ".$post['parent'];
$result = mysql_query($query);
if (!$result)
{
  return false;
}

// find our post id, note that there could be multiple headers
// that are the same except for id and probably posted time
$query = "select header.postid from header left
         join body on header.postid = body.postid
         where parent = '".$post["parent"]."'
         and poster = '".$post["poster"]."'
         and title = '".$post["title"]."'
         and body.postid is NULL";
$result = mysql_query($query);
if (!$result)
{
  return false;
}
if(mysql_numrows($result)>0)
```

```
    $id = mysql_result($result, 0, 0);

  if($id)
  {
    $query = "insert into body values ($id, '".$post["message"]."')";
    $result = mysql_query($query);
    if (!$result)
    {
      return false;
    }
  return $id;
  }
}
```

Ini adalah fungsi yang panjang, tetapi tidak terlalu rumit. Fungsi ini panjang karena memasukkan posting berarti memasukkan entri dalam tabel header dan body, dan memperbarui baris artikel induk dalam tabel header untuk menunjukkan bahwa artikel tersebut sekarang memiliki anak.

Itulah akhir kode untuk aplikasi forum Web.

Ekstensi

Ada banyak ekstensi yang dapat Anda tambahkan ke proyek ini:

- Anda dapat menambahkan navigasi ke opsi tampilan, sehingga dari sebuah posting Anda dapat menavigasi ke pesan berikutnya, pesan sebelumnya, pesan berikutnya dalam thread, atau pesan sebelumnya dalam thread.
- Anda dapat menambahkan antarmuka administrasi untuk menyiapkan forum baru dan menghapus posting lama.
- Anda dapat menambahkan autentikasi pengguna sehingga hanya pengguna terdaftar yang dapat memposting.
- Anda dapat menambahkan semacam mekanisme moderasi atau penyensoran.

Lihat sistem yang ada untuk mendapatkan ide.

BAB 9
MEMBUAT DOKUMEN PRIBADI DALAM PORTABLE DOCUMENT FORMAT

Di situs yang digerakkan oleh layanan, terkadang kami perlu memberikan dokumen pribadi, yang dibuat sebagai respons terhadap masukan dari pengunjung kami. Ini dapat digunakan untuk menyediakan formulir yang diisi secara otomatis atau untuk membuat dokumen pribadi, seperti dokumen hukum, surat, atau sertifikat.

Contoh kami dalam bab ini akan menyajikan halaman penilaian keterampilan daring kepada pengguna dan membuat sertifikat.

Dalam bab ini penulis akan menerangkan tentang:

- Cara menggunakan pemrosesan string PHP untuk mengintegrasikan templat dengan data pengguna guna membuat dokumen Rich Text Format (RTF)
- Cara menggunakan pendekatan serupa untuk membuat dokumen Portable Document Format (PDF)
- Cara menggunakan fungsi PDFlib PHP untuk membuat dokumen PDF serupa

9.1 PENDAHULUAN

Kami ingin dapat memberikan ujian yang terdiri dari sejumlah pertanyaan kepada pengunjung kami. Jika mereka menjawab cukup banyak pertanyaan dengan benar, kami akan membuat sertifikat bagi mereka untuk menunjukkan bahwa mereka telah lulus ujian. Agar komputer dapat menandainya dengan mudah, pertanyaan kita akan berupa pilihan ganda, yang terdiri dari satu pertanyaan dan sejumlah kemungkinan jawaban. Hanya satu dari kemungkinan jawaban untuk setiap pertanyaan yang akan benar.

Jika pengguna memperoleh nilai kelulusan pada pertanyaan tersebut, ia akan diberikan sertifikat. Idealnya, format berkas untuk sertifikat kita harus

1. Mudah dirancang
2. Dapat memuat berbagai elemen yang berbeda seperti gambar bitmap dan vektor
3. Menghasilkan cetakan berkualitas tinggi
4. Hanya memerlukan berkas kecil untuk diunduh
5. Dibuat hampir seketika
6. Biaya produksinya rendah
7. Dapat digunakan pada banyak sistem operasi
8. Sulit untuk digandakan atau dimodifikasi secara curang
9. Tidak memerlukan perangkat lunak khusus untuk melihat atau mencetak
10. Ditampilkan dan dicetak secara konsisten untuk semua penerima

Seperti banyak keputusan yang perlu kita buat dari waktu ke waktu, kita mungkin perlu berkompromi saat memilih format pengiriman untuk memenuhi sebanyak mungkin dari sepuluh atribut ini.

9.2 MENGEVALUASI FORMAT DOKUMEN

Keputusan terpenting yang perlu kita buat adalah format apa yang akan digunakan untuk mengirimkan sertifikat. Pilihannya meliputi kertas, teks ASCII, HTML, Microsoft Word, atau format pengolah kata lain, Rich Text Format, PostScript, dan Portable Document Format. Dengan mempertimbangkan sepuluh atribut yang disebutkan sebelumnya, kita dapat mempertimbangkan dan membandingkan beberapa pilihan kita.

Kertas

Mengirimkan sertifikat di atas kertas memiliki beberapa keuntungan yang jelas. Kita memegang kendali penuh atas prosesnya. Kita dapat melihat dengan tepat seperti apa tampilan setiap keluaran sertifikat sebelum mengirimkannya kepada penerima. Kita tidak perlu khawatir tentang perangkat lunak atau bandwidth, dan sertifikat dapat dicetak dengan langkah-langkah anti-pemalsuan.

Sertifikat tersebut akan memenuhi semua kebutuhan kita kecuali untuk atribut 5 dan 6. Sertifikat tidak dapat dibuat dan dikirimkan dengan cepat. Pengiriman melalui pos dapat memakan waktu berhari-hari atau berminggu-minggu, tergantung pada lokasi kita dan penerima.

Setiap sertifikat juga akan menghabiskan biaya beberapa sen hingga beberapa dolar untuk biaya pencetakan dan ongkos kirim dan mungkin lebih banyak lagi untuk penanganan. Pengiriman elektronik otomatis akan lebih murah.

ASCII

Mengirim dokumen sebagai ASCII atau teks biasa memiliki beberapa keuntungan. Kompatibilitas tidak akan menjadi masalah. Bandwidth yang dibutuhkan akan kecil, jadi biayanya akan sangat rendah. Kesederhanaan hasil akhirnya akan membuatnya sangat mudah dirancang dan sangat cepat untuk membuat skrip.

Namun, jika kami memberikan pengunjung kami file ASCII, kami memiliki sedikit kendali atas tampilan sertifikat mereka. Kami tidak dapat mengendalikan font atau pemisah halaman. Kami hanya dapat menyertakan teks dan memiliki sedikit kendali atas pemformatan. Kami tidak memiliki kendali atas duplikasi atau modifikasi dokumen oleh penerima. Ini adalah metode yang memudahkan penerima untuk mengubah sertifikatnya secara curang.

HTML

Pilihan yang jelas untuk mengirimkan dokumen di Web adalah HTML. Hypertext Markup Language secara khusus dirancang untuk tujuan ini. Seperti yang mungkin sudah Anda ketahui, HTML mencakup kontrol pemformatan, sintaksis untuk menyertakan objek seperti gambar, dan kompatibel (dengan beberapa variasi) dengan berbagai sistem operasi dan perangkat lunak. HTML cukup sederhana, jadi akan mudah dirancang dan cepat untuk membuat dan mengirimkan skrip.

Kelemahan penggunaan HTML untuk aplikasi ini meliputi: dukungan terbatas untuk format terkait pencetakan seperti pemisah halaman; sedikit konsistensi dalam keluaran pada berbagai platform dan program; dan kualitas pencetakan yang bervariasi. Selain itu, meskipun HTML dapat menyertakan semua jenis elemen eksternal, kemampuan browser untuk

menampilkan atau menggunakan elemen-elemen ini tidak dapat dijamin untuk jenis yang tidak biasa.

9.3 FORMAT PENGOLAH KATA

Khususnya untuk proyek intranet, menyediakan dokumen sebagai dokumen pengolah kata masuk akal. Namun, untuk proyek Internet, menggunakan format pengolah kata milik sendiri akan mengecualikan beberapa pengunjung, tetapi mengingat dominasi pasarnya, Microsoft Word masuk akal. Sebagian besar pengguna akan memiliki akses ke Word atau pengolah kata yang akan mencoba membaca berkas Word.

Membuat dokumen sebagai dokumen Microsoft Word memiliki beberapa keuntungan. Selama Anda memiliki salinan Word, mendesain dokumen menjadi mudah. Kami memiliki kontrol yang sangat baik atas tampilan cetak dokumen kami dan banyak fleksibilitas dengan isinya. Anda juga dapat membuatnya relatif sulit bagi penerima untuk mengubahnya dengan memberi tahu Word untuk meminta kata sandi.

Sayangnya, berkas Word bisa berukuran besar, terutama jika berisi gambar atau elemen kompleks lainnya. Tidak ada cara mudah untuk membuatnya secara dinamis dengan PHP. Formatnya didokumentasikan, tetapi merupakan format biner dan dokumentasi format disertai dengan ketentuan lisensi.

Rich Text Format

Rich Text Format atau RTF memberi kita sebagian besar kekuatan Word, tetapi file-file tersebut lebih mudah dibuat. Kita masih memiliki fleksibilitas atas tata letak dan pemformatan halaman yang dicetak. Kita masih dapat menyertakan elemen-elemen seperti gambar vektor atau bitmap. Kita masih dapat cukup yakin bahwa pengguna akan melihat hasil yang serupa dengan kita saat mereka melihat atau mencetak dokumen.

RTF adalah format teks Microsoft Word. Format ini dimaksudkan sebagai format pertukaran untuk mentransfer dokumen antar program yang berbeda. Dalam beberapa hal, format ini mirip dengan HTML. Format ini menggunakan sintaksis dan kata kunci daripada data biner untuk menyampaikan informasi pemformatan. Oleh karena itu, format ini relatif dapat dibaca manusia.

Cara termudah untuk membuat dokumen RTF adalah dengan memilih opsi Simpan Sebagai RTF di pengolah kata Anda. Karena file RTF hanya berisi teks, file-file tersebut dapat dibuat secara langsung dan file-file yang sudah ada dapat dengan mudah dimodifikasi.

Karena formatnya terdokumentasi dan tersedia secara bebas, RTF dapat dibaca oleh lebih banyak perangkat lunak daripada format biner Word. Namun perlu diingat bahwa pengguna yang membuka file RTF yang rumit di versi Word yang lama atau pengolah kata yang berbeda akan sering melihat hasil yang agak berbeda. Setiap versi Word yang baru memperkenalkan kata kunci baru ke RTF, jadi implementasi yang lama biasanya akan mengabaikan kontrol yang tidak mereka pahami atau tidak mereka pilih untuk diterapkan.

Dari daftar awal kami, sertifikat RTF akan mudah dirancang menggunakan Word atau pengolah kata lain; dapat memuat berbagai elemen yang berbeda seperti gambar vektor dan

bitmap; memberikan cetakan berkualitas tinggi; dapat dibuat dengan mudah dan cepat; dan dapat dikirimkan secara elektronik dengan biaya rendah.

Sertifikat ini akan berfungsi dengan berbagai aplikasi dan sistem operasi, meskipun dengan hasil yang agak bervariasi. Di sisi negatifnya, dokumen RTF dapat dengan mudah dan bebas dimodifikasi oleh siapa pun, yang merupakan masalah untuk sertifikat dan beberapa jenis dokumen lainnya. Ukuran file mungkin menjadi cukup besar untuk dokumen yang rumit. RTF merupakan pilihan yang baik untuk banyak aplikasi pengiriman dokumen, jadi kami akan menggunakannya sebagai salah satu pilihan di sini.

PostScript

PostScript, dari Adobe, adalah bahasa deskripsi halaman. Bahasa ini merupakan bahasa pemrograman yang kuat dan kompleks yang dimaksudkan untuk merepresentasikan dokumen dengan cara yang tidak bergantung pada perangkat—yaitu, deskripsi yang akan menghasilkan hasil yang konsisten di berbagai perangkat seperti printer dan layar. Bahasa ini didokumentasikan dengan sangat baik. Setidaknya tersedia tiga buku lengkap, serta situs web yang tak terhitung jumlahnya.

Dokumen PostScript dapat berisi format, teks, gambar, font tertanam, dan elemen lain yang sangat tepat. Anda dapat dengan mudah membuat dokumen PostScript dari aplikasi dengan mencetaknya ke driver printer PostScript. Jika Anda tertarik, Anda bahkan dapat belajar memprogramnya secara langsung.

Dokumen PostScript cukup portabel. Dokumen ini akan memberikan hasil cetak berkualitas tinggi yang konsisten dari berbagai perangkat dan sistem operasi yang berbeda. Ada beberapa kelemahan signifikan dalam penggunaan PostScript untuk mendistribusikan dokumen:

- File-file tersebut dapat berukuran besar.
- Banyak orang perlu mengunduh perangkat lunak tambahan untuk menggunakannya.

Sebagian besar pengguna UNIX dapat menangani berkas PostScript, tetapi pengguna Windows biasanya perlu mengunduh penampil seperti GSview, yang menggunakan penerjemah PostScript Ghostscript. Perangkat lunak ini tersedia untuk berbagai platform. Meskipun tersedia gratis, kami tidak ingin memaksa orang untuk mengunduh lebih banyak perangkat lunak.

Portable Document Format (juga dari Adobe) dirancang sebagai cara untuk mendistribusikan dokumen yang akan berperilaku konsisten pada platform yang berbeda, dan memberikan keluaran berkualitas tinggi yang dapat diprediksi di layar atau di atas kertas. Adobe menggambarkan PDF sebagai "standar de facto terbuka untuk distribusi dokumen elektronik di seluruh dunia. Adobe PDF adalah format file universal yang mempertahankan semua font, format, warna, dan grafik dari dokumen sumber apa pun, terlepas dari aplikasi dan platform yang digunakan untuk membuatnya.

Dokumen PDF memberikan keluaran yang konsisten dan berkualitas tinggi, mampu memuat elemen seperti gambar bitmap dan vektor, dapat menggunakan kompresi untuk membuat file kecil, dapat dikirimkan secara elektronik dan murah, dapat digunakan pada sistem operasi utama, dan dapat menyertakan kontrol keamanan.

Yang menjadi kendala PDF adalah kenyataan bahwa sebagian besar perangkat lunak yang digunakan untuk membuat dokumen PDF bersifat komersial. Pembaca diperlukan untuk melihat file PDF, tetapi Acrobat Reader tersedia gratis untuk Windows, UNIX, dan Macintosh dari Adobe. Banyak pengunjung situs Anda yang sudah familier dengan ekstensi .pdf dan kemungkinan besar sudah menginstal pembaca tersebut.

File PDF merupakan cara yang baik untuk mendistribusikan dokumen yang menarik dan dapat dicetak, khususnya dokumen yang tidak ingin penerimanya dapat dengan mudah mengubahnya. Kita akan melihat dua cara berbeda untuk membuat sertifikat PDF.

Agar sistem berfungsi, kita perlu memeriksa pengetahuan pengguna dan (dengan asumsi mereka lulus ujian) membuat sertifikat yang melaporkan kinerja mereka. Kita akan bereksperimen dengan membuat sertifikat ini dengan tiga cara berbeda: dua menggunakan PDF dan satu menggunakan RTF. Mari kita lihat persyaratan masing-masing komponen ini secara mendetail.

Sistem Tanya Jawab

Menyediakan sistem yang fleksibel untuk penilaian daring yang memungkinkan berbagai jenis pertanyaan, berbagai jenis media untuk informasi pendukung, umpan balik yang berguna untuk jawaban yang salah, dan pengumpulan dan pelaporan statistik yang cerdas, akan menjadi tugas yang rumit.

Dalam bab ini, kita terutama tertarik pada tantangan membuat dokumen khusus untuk pengiriman melalui Web, jadi kita hanya akan membangun sistem kuis yang sangat sederhana. Kuis tidak bergantung pada perangkat lunak khusus apa pun. Kuis menggunakan formulir HTML untuk mengajukan pertanyaan dan skrip PHP untuk memproses jawaban.

Perangkat Lunak Pembuatan Dokumen

Tidak diperlukan perangkat lunak tambahan di server Web untuk membuat dokumen RTF atau PDF dari templat, tetapi Anda memerlukan perangkat lunak untuk membuat templat tersebut. Untuk menggunakan fungsi pembuatan PDF PHP, Anda perlu mengompilasi dukungan PDF ke dalam PHP. (Kita akan membahas lebih lanjut tentang ini sebentar lagi.)

Perangkat Lunak untuk Membuat Templat RTF

Anda dapat menggunakan pengolah kata pilihan Anda untuk membuat berkas RTF. Kami menggunakan Microsoft Word untuk membuat templat sertifikat.

Jika Anda lebih suka pengolah kata lain, sebaiknya uji keluaran di Word karena ini adalah perangkat lunak yang akan digunakan sebagian besar pengunjung Anda.

Perangkat Lunak untuk Membuat Templat PDF

Dokumen PDF sedikit lebih sulit dibuat. Cara termudah adalah dengan membeli Adobe Acrobat. Perangkat lunak ini akan memungkinkan Anda membuat PDF berkualitas tinggi dari berbagai aplikasi. Kami menggunakan Acrobat untuk membuat berkas templat untuk proyek ini.

Untuk membuat berkas, kami menggunakan Microsoft Word untuk mendesain dokumen. Salah satu alat dalam paket Acrobat adalah Adobe Distiller. Di dalam Distiller, kami perlu memilih beberapa opsi non-default. Berkas harus disimpan dalam format ASCII, dan

kompresi harus dinonaktifkan. Setelah opsi ini ditetapkan, membuat berkas PDF semudah mencetak.

Opsi lain untuk membuat PDF adalah program konversi ps2pdf, yang seperti namanya mengonversi berkas PostScript menjadi berkas PDF. Program ini memiliki keuntungan karena gratis, tetapi tidak selalu menghasilkan keluaran yang baik untuk dokumen dengan gambar atau font nonstandar. Konverter ps2pdf disertakan dengan paket Ghostscript yang disebutkan sebelumnya.

Tentunya, jika Anda akan membuat berkas PDF dengan cara ini, Anda perlu membuat berkas PostScript terlebih dahulu. Pengguna UNIX biasanya akan menggunakan utilitas a2ps atau dvips untuk tujuan ini.

Jika Anda bekerja di lingkungan Windows, Anda juga dapat membuat file PostScript tanpa Adobe Distiller, meskipun melalui proses yang sedikit lebih rumit. Anda perlu memasang driver printer PostScript. Misalnya, Anda dapat menggunakan driver Apple LaserWriter IINT.

Untuk membuat file PostScript, Anda perlu memilih printer ini dan opsi Print to File, yang biasanya terdapat pada kotak dialog Print. Sebagian besar aplikasi Windows kemudian akan menghasilkan file dengan ekstensi .prn. Ini seharusnya berupa file PostScript. Anda mungkin harus mengganti nama ini menjadi file .ps. Anda seharusnya dapat melihatnya menggunakan GSview atau penampil PostScript lain, atau membuat file PDF menggunakan utilitas ps2pdf.

Perlu diketahui bahwa driver printer yang berbeda menghasilkan keluaran PostScript dengan kualitas yang berbeda-beda. Anda mungkin menemukan bahwa beberapa file PostScript yang Anda hasilkan memberikan kesalahan saat dijalankan melalui utilitas ps2pdf. Kami sarankan untuk menggunakan driver printer yang berbeda.

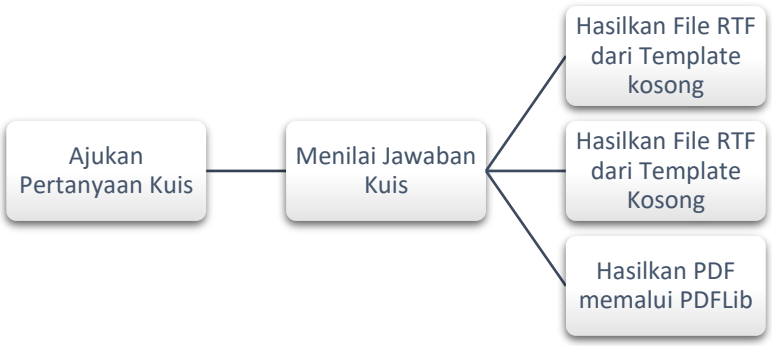
Jika Anda hanya ingin membuat sejumlah kecil file PDF, layanan daring Adobe mungkin cocok untuk Anda. Dengan biaya $9,99 per bulan, Anda dapat mengunggah file dalam sejumlah format dan mengunduh file PDF. Layanan ini berfungsi dengan baik untuk sertifikat kami, tetapi tidak memungkinkan Anda memilih opsi yang penting untuk proyek ini. PDF yang dibuat akan disimpan sebagai file biner dan dikompresi. Hal ini membuatnya sangat sulit untuk dimodifikasi.

Perangkat Lunak untuk Membuat PDF Secara Terprogram

Dukungan untuk membuat dokumen PDF tersedia dari dalam PHP. Tersedia dua pustaka fungsi yang berbeda, dengan tujuan yang sama. Karena keduanya bergantung pada pustaka eksternal, keduanya tidak dikompilasi ke dalam PHP secara default.

Kami akan membuat sistem dengan tiga kemungkinan hasil. Seperti yang ditunjukkan pada Gambar 9.1, kami akan mengajukan pertanyaan kuis, menilai jawaban, lalu membuat sertifikat dengan salah satu dari tiga cara:

- Kami akan membuat dokumen RTF dari templat kosong.
- Kami akan membuat dokumen PDF dari templat kosong.
- Kami akan membuat dokumen PDF secara terprogram melalui PDFlib.



Gambar 9.1 Sistem sertifikasi kami akan menghasilkan satu dari tiga sertifikat yang berbeda.

Ringkasan berkas dalam proyek sertifikasi ditunjukkan pada Tabel 9.1.

Tabel 9.1 Berkas dalam Aplikasi Sertifikasi

Nama	Tipe	Keterangan
index.html	HTML Page	Formulir HTML yang berisi pertanyaan kuis
score.php	Aplikasi	Skrip untuk menilai jawaban pengguna
rtf.php	Aplikasi	Skrip untuk menghasilkan sertifikat RTF dari template
pdf.php	Aplikasi	Skrip untuk menghasilkan sertifikat PDF dari template
pdflib.php	Aplikasi	Skrip untuk menghasilkan sertifikat PDF menggunakan PDFlib
signature.tif	Gambar	Gambar bitmap tanda tangan yang akan disertakan pada sertifikat PDFlib
PHPCertification.rtf	RTF	Templat sertifikat RTF
PHPCertification.pdf	PDF	Templat sertifikat PDF

Mari kita lanjutkan dan lihat aplikasinya.

Mengajukan Pertanyaan

File index.html mudah dipahami. File tersebut harus berisi formulir HTML yang menanyakan nama pengguna dan jawaban atas sejumlah pertanyaan. Dalam aplikasi penilaian yang sebenarnya, kemungkinan besar kita akan mengambil pertanyaan-pertanyaan ini dari basis data. Di sini kita berfokus pada pembuatan sertifikat, jadi kita hanya akan memasukkan beberapa pertanyaan ke dalam HTML.

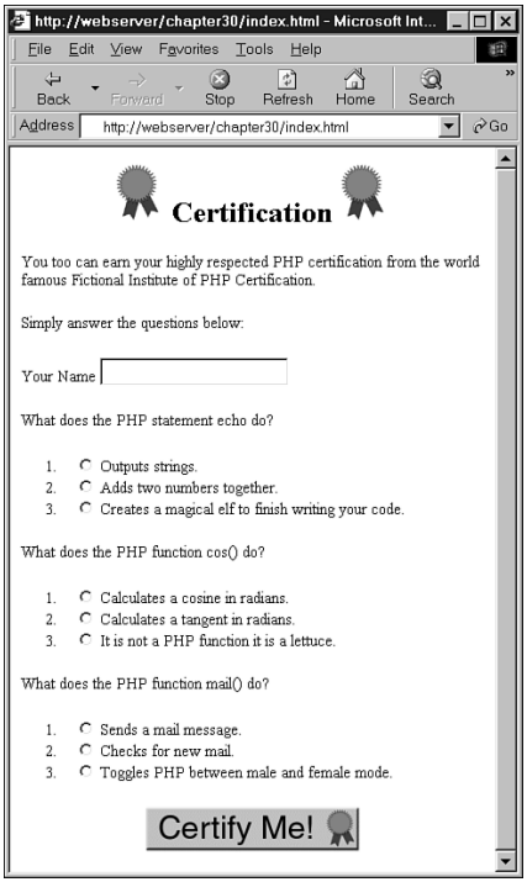
Kolom nama adalah input teks. Setiap pertanyaan memiliki tiga tombol radio untuk memungkinkan pengguna menunjukkan jawaban yang diinginkannya. Formulir tersebut memiliki tombol gambar sebagai tombol kirim. Kode untuk halaman ini ditunjukkan dalam Listing 9.1.

Listing 9.1 index.html—Halaman HTML yang Berisi Pertanyaan Kuis

```
<html>
  <body>
    <h1><p align = center>
```

```
        <img src = “rosette.gif” alt = “”>
        Certification
        <img src = “rosette.gif” alt = “”></h1>
    <p>You too can earn your highly respected PHP certification
      from the world famous Fictional Institute of PHP Certification.
    <p>Simply answer the questions below:

    <form action = score.php method = post>

        <p>Your Name <input type = text name = name>

        <p>What does the PHP statement echo do?
        <ol>
        <li><input type = radio name = q1 value = 1>
          Outputs strings.
        <li><input type = radio name = q1 value = 2>
          Adds two numbers together.
        <li><input type = radio name = q1 value = 3>
          Creates a magical elf to finish writing your code.

      </ol>

      <p>What does the PHP function cos() do?
      <ol>
        <li><input type = radio name = q2 value = 1>
          Calculates a cosine in radians.
        <li><input type = radio name = q2 value = 2>
          Calculates a tangent in radians.
        <li><input type = radio name = q2 value = 3>
          It is not a PHP function it is a lettuce.
      </ol>

      <p>What does the PHP function mail() do?
      <ol>
        <li><input type = radio name = q3 value = 1>
          Sends a mail message.
        <li><input type = radio name = q3 value = 2>
          Checks for new mail.
        <li><input type = radio name = q3 value = 3>
          Toggles PHP between male and female mode.
      </ol>
      <p align = center><input type = image src = “certify-me.gif” border = 0>

    </form>
  </body>
</html>
```

Hasil pemuatan index.html di browser Web ditunjukkan pada Gambar 9.2.



Gambar 9.2 index.html meminta pengguna untuk menjawab pertanyaan kuis.

Memberi Nilai Jawaban

Saat pengguna mengirimkan jawabannya pada pertanyaan di `index.html`, kita perlu memberinya nilai dan menghitung skor. Ini dilakukan oleh skrip yang disebut `score.php`. Kode untuk skrip ini ditunjukkan pada Listing 9.2.

Listing 9.2 `score.php`—Skrip untuk Memberi Nilai Ujian

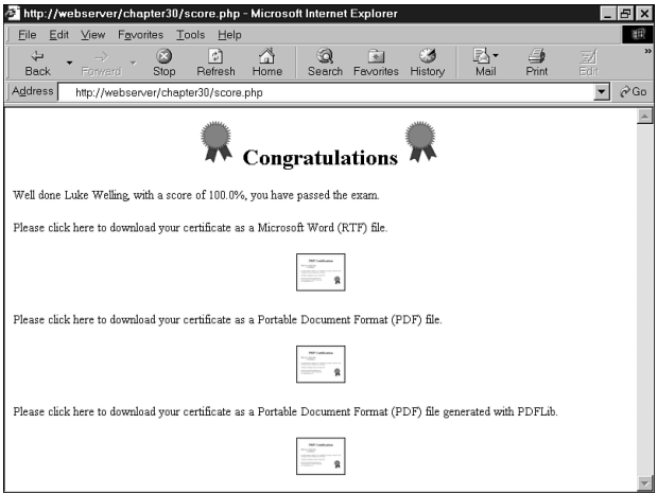
```
<?
  // check that all the data was received
  if($q1==''||$q2==''||$q3==''||$name='')
  {
    echo "<h1><p align = center><img src = 'rosette.gif' alt = ''>
                              Sorry:
                              <img src = 'rosette.gif' alt = ''></h1>";
    echo "<p>You need to fill in your name and answer all questions";
  }
  else
  {
    //add up the scores
    $score = 0;
```

```
if($q1 == 1) // the correct answer for q1 is 1
  $score++;
if($q2 == 1) // the correct answer for q2 is 1
  $score++;
if($q3 == 1) // the correct answer for q3 is 1
  $score++;

//convert score to a percentage
$score = $score / 3 * 100;

if($score < 50)
{
  // this person failed
  echo "<h1><p align = center><img src = 'rosette.gif' alt = ''>
                        Sorry:
                        <img src = 'rosette.gif' alt = ''></h1>";

  echo "<p>You need to score at least 50% to pass the exam";
}
else
{
  // create a string containing the score to one decimal place
  $score = number_format($score, 1);

  echo "<h1><p align = center><img src = 'rosette.gif' alt = ''>
                        Congratulations
                        <img src = 'rosette.gif' alt = ''></h1>";
  echo "<p>Well done $name, with a score of $score%,
            you have passed the exam. ";

  // provide links to scripts that generate the certificates
  echo "<p>Please click here to download your certificate as
            a Microsoft Word (RTF) file. ";
  echo "<form action = 'rtf.php' method = get>";
  echo "<center>
          <input type = image src = 'certificate.gif' border = 0>
        </center>";
  echo "<input type = hidden name = score value = '$score'>";
  echo "<input type = hidden name = name value = '$name'>";
  echo "</form>";

  echo "<p>Please click here to download your certificate as
          a Portable Document Format (PDF) file. ";
  echo "<form action = 'pdf.php' method = get>";
  echo "<center>
          <input type = image src = 'certificate.gif' border = 0>
        </center>";
```

```
        echo "<input type = hidden name = score value = '$score'>";
        echo "<input type = hidden name = name value = '$name'>";
        echo "</form>";

        echo "<p>Please click here to download your certificate as
             a Portable Document Format (PDF) file generated with PDFLib. ";
        echo "<form action = 'pdflib.php' method = get>";
        echo "<center>
                    <input type = image src = 'certificate.gif' border = 0>
              </center>";

        echo "<input type = hidden name = score value = '$score'>";
        echo "<input type = hidden name = name value = '$name'>";
        echo "</form>";
    }
  }
?>
```

Skrip ini akan menampilkan pesan jika pengguna tidak menjawab semua pertanyaan atau mendapat skor kurang dari nilai kelulusan yang kami pilih. Jika pengguna berhasil menjawab pertanyaan, ia akan diizinkan untuk membuat sertifikat. Hasil kunjungan yang berhasil ditunjukkan pada Gambar 9.3.

Dari sini, pengguna memiliki tiga pilihan. Ia dapat memiliki sertifikat RTF, atau salah satu dari dua sertifikat PDF. Kita akan melihat skrip yang bertanggung jawab untuk masing-masing pilihan.



Gambar 9.3 score.php menyajikan pilihan kepada pengunjung yang berhasil untuk membuat sertifikat dengan salah satu dari tiga cara.

9.4 MEMBUAT SERTIFIKAT RTF

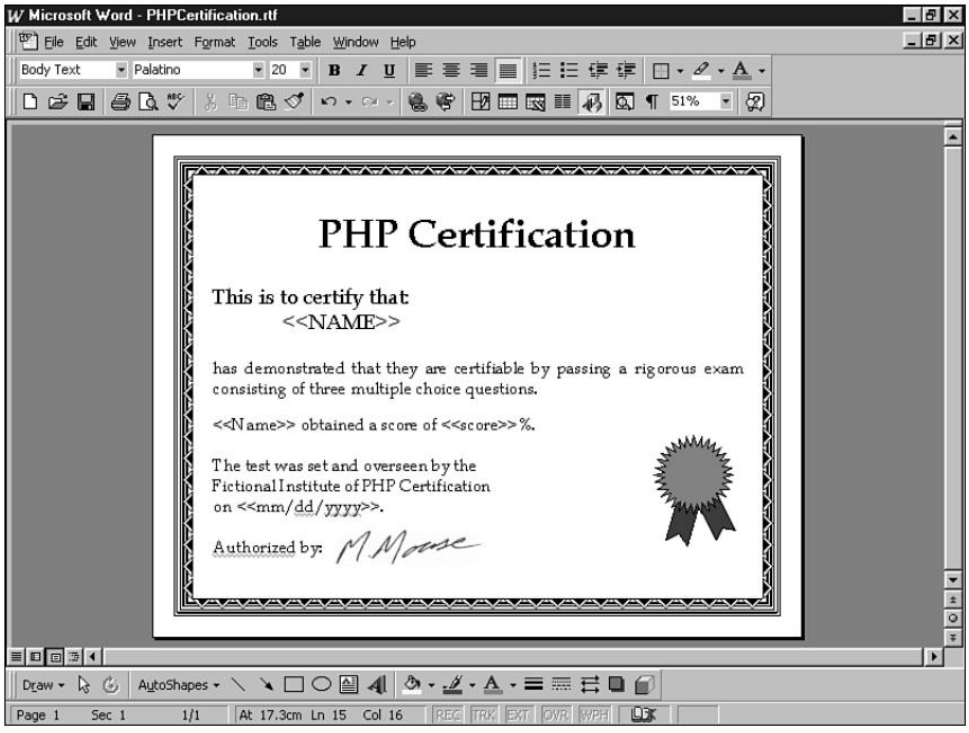
Tidak ada yang menghalangi kita untuk membuat dokumen RTF dengan menulis teks ASCII ke file atau variabel string, tetapi itu berarti mempelajari serangkaian sintaksis lainnya. Berikut adalah dokumen RTF yang sangat sederhana:

```
{\rtf1
{\fonttbl {\f0 Arial;}{\f1 Times New Roman;}}
\f0\fs28 Heading\par
\f1\fs20 This is an rtf document.\par
}
```

Dokumen ini menyiapkan tabel fon dengan dua fon: Arial, yang disebut sebagai `f0`, dan Times New Roman, yang disebut sebagai `f1`. Kemudian, ia menulis Heading menggunakan `f0` (Arial) dalam ukuran 28 (14 poin). Kontrol \par menunjukkan pemisah paragraf. Kemudian, kita menulis `This is an rtf document using f1` (Times New Roman) pada ukuran 20 (10 poin).

Kita dapat membuat dokumen seperti ini secara manual, tetapi tidak ada fungsi penghemat tenaga kerja bawaan PHP untuk mempermudah bagian-bagian yang sulit, seperti memasukkan grafik. Untungnya, dalam banyak dokumen, struktur, gaya, dan sebagian besar teks bersifat statis, dan hanya bagian-bagian kecil yang berubah dari orang ke orang. Cara yang lebih efisien untuk membuat dokumen adalah menggunakan templat.

Kita dapat membuat dokumen yang kompleks, seperti yang ditunjukkan pada Gambar 9.4, dengan mudah menggunakan pengolah kata.



Gambar 9.4 Dengan menggunakan pengolah kata, kita dapat membuat templat yang kompleks dan menarik dengan mudah.

Templat kita menyertakan placeholder seperti `<<NAMA>>` untuk menandai tempat di mana data dinamis akan disisipkan. Tidak penting seperti apa tampilan placeholder ini. Kita menggunakan deskripsi yang bermakna di antara dua set kurung siku. Penting bagi kita untuk

memilih placeholder yang sangat tidak mungkin muncul secara tidak sengaja di bagian dokumen lainnya. Ini akan membantu Anda menata templat jika placeholder kira-kira sama panjangnya dengan data yang akan diganti.

Placeholder dalam dokumen ini adalah `<<NAMA>>`, `<<Nama>>`, `<<skor>>`, dan `<<mm/dd/yyyy>>`. Perhatikan bahwa kita menggunakan NAMA dan Nama, karena kita bermaksud menggunakan metode peka huruf besar/kecil untuk menggantinya. Sekarang setelah kita memiliki templat, kita memerlukan skrip untuk mempersonalisasikannya. Skrip ini disebut `rtf.php`, dan kodenya ditunjukkan pada Listing 9.3.

Listing 9.3 rtf.php—Skrip untuk Menghasilkan Sertifikat RTF yang Dipersonalisasi

```
<?
// check we have the parameters we need
  if( !$name || !$score )
  {
    echo “<h1>Error:</h1>This page was called incorrectly”;
  }
  else
  {
    //generate the headers to help a browser choose the correct application
    header( “Content-type: application/msword” );
    header( “Content-Disposition: inline, filename=cert.rtf”);

    $date = date( “F d, Y” );

    // open our template file
    $filename = “PHPCertification.rtf”;
    $fp = fopen ( $filename, “r” );

    //read our template into a variable
    $output = fread( $fp, filesize( $filename ) );

    fclose ( $fp );

    // replace the place holders in the template with our data
    $output = str_replace( “<<NAME>>”, strtoupper( $name ), $output );
    $output = str_replace( “<<Name>>”, $name, $output );
    $output = str_replace( “<<score>>”, $score, $output );
    $output = str_replace( “<<mm/dd/yyyy>>”, $date, $output );

    // send the generated document to the browser
    echo $output;
  }
?>
```

Skrip ini melakukan beberapa pemeriksaan kesalahan dasar untuk memastikan bahwa semua detail pengguna telah dimasukkan, lalu beralih ke pembuatan sertifikat. Keluaran skrip ini akan berupa file RTF, bukan file HTML, jadi kita perlu memberi tahu browser pengguna tentang fakta ini. Ini penting agar browser dapat mencoba membuka file dengan aplikasi yang benar, atau memberikan kotak dialog Save As... type jika tidak mengenali ekstensi RTF.

Kita tentukan tipe MIME file yang kita hasilkan menggunakan fungsi header() PHP untuk mengirim header HTTP yang sesuai sebagai berikut:

```
header( "Content-type: application/msword" );
header( "Content-Disposition: inline, filename=cert.rtf");
```

Header pertama memberi tahu browser bahwa kita sedang mengirim file Microsoft Word (tidak sepenuhnya benar, tetapi kemungkinan besar merupakan aplikasi pembantu untuk membuka file RTF).

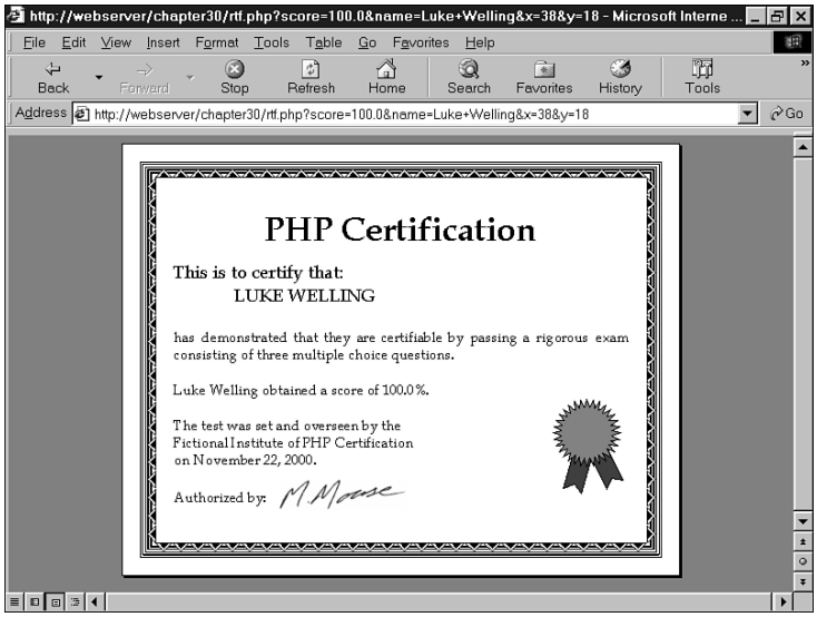
Header kedua memberi tahu browser untuk secara otomatis menampilkan konten file, dan nama file yang disarankan adalah `cert.rtf`. Ini adalah nama file default yang akan dilihat pengguna jika ia mencoba menyimpan file dari dalam browser-nya.

Setelah header dikirim, kita membuka dan membaca file RTF templat kita ke dalam variabel $output, dan menggunakan fungsi `str_replace()` untuk mengganti placeholder kita dengan data aktual yang ingin kita tampilkan dalam file. Baris

```
$output = str_replace( "<<Name>>", $name, $output );
```

akan mengganti setiap kemunculan placeholder <<Name>> dengan konten variabel **$name.**

Setelah melakukan substitusi, yang perlu dilakukan hanyalah menggemakan output ke browser. Contoh hasil dari skrip ini ditunjukkan pada Gambar 9.5.



Gambar 9.5 rtf.php menghasilkan sertifikat dari templat RTF.

Pendekatan ini bekerja dengan sangat baik. Panggilan ke `str_replace()` berjalan sangat cepat, meskipun templat dan konten `$output` kita cukup panjang. Masalah utama dari sudut pandang aplikasi ini adalah pengguna akan memuat sertifikat di pengolah kata untuk mencetaknya. Ini mungkin undangan bagi orang untuk mengubah output. RTF tidak memungkinkan kita membuat dokumen hanya-baca.

9.5 MEMBUAT SERTIFIKAT PDF DARI TEMPLAT

Proses pembuatan sertifikat PDF dari templat sangat mirip. Perbedaan utamanya adalah saat kita membuat file PDF, beberapa placeholder kita mungkin diselingi dengan kode pemformatan. Misalnya, jika kita melihat file templat sertifikat yang telah kita buat, kita dapat melihat bahwa placeholder sekarang terlihat seperti ini:

```
<<N)-13(AME)-10(>)-6(>
<<Na)-9(m)0(e)-18(>>
<)-11(<)1(sc)-17(or)-6(e)-6(>)-11(>
<)-11(<)1(m)-12(m)0(/d)-6(d)-19(/)1(yy)-13(yy)-13(>>
```

Jika Anda melihat berkas tersebut, Anda akan melihat bahwa, tidak seperti RTF, ini bukanlah format yang dapat dibaca manusia dengan mudah.
Ada beberapa cara berbeda untuk mengatasi hal ini.

Kita dapat memeriksa setiap placeholder ini dan menghapus kode pemformatan. Hal ini sebenarnya tidak terlalu memengaruhi tampilan dokumen pada akhirnya karena kode yang disematkan pada templat sebelumnya menunjukkan seberapa banyak ruang yang harus disisakan di antara huruf-huruf placeholder yang akan kita ganti. Namun, jika kita menggunakan pendekatan ini, kita harus memeriksa dan mengedit berkas PDF secara manual dan mengulanginya setiap kali kita mengubah atau memperbarui berkas. Ini bukan masalah besar jika hanya menggunakan empat placeholder, tetapi akan menjadi mimpi buruk jika, misalnya, Anda memiliki banyak dokumen dengan banyak placeholder, dan Anda memutuskan untuk mengubah kop surat pada semua dokumen.

Kita dapat menghindari masalah ini dengan menggunakan teknik yang berbeda. Anda dapat menggunakan Adobe Acrobat untuk membuat formulir PDF—mirip dengan formulir HTML dengan kolom kosong bernama. Anda kemudian dapat menggunakan skrip PHP untuk membuat apa yang disebut berkas FDF (Forms Data Format), yang pada dasarnya merupakan sekumpulan data yang akan digabungkan dengan templat.

Anda dapat membuat FDF menggunakan pustaka fungsi FDF PHP: khususnya, fungsi `fdf_create()` untuk membuat berkas; fungsi `fdf_set_value()` untuk mengatur nilai kolom; dan fungsi `fdf_set_file()` untuk mengatur berkas formulir templat terkait. Anda kemudian dapat meneruskan berkas ini kembali ke peramban dengan tipe MIME yang sesuai, dalam hal ini `vnd.fdf`, dan plug-in Acrobat Reader peramban akan mengganti data ke dalam formulir.

Ini adalah cara yang bagus untuk melakukan berbagai hal, tetapi memiliki dua keterbatasan. Pertama, cara ini mengasumsikan bahwa Anda memiliki salinan Acrobat. Kedua, sulit untuk mengganti teks yang sebaris daripada teks yang tampak seperti bidang formulir. Ini

mungkin menjadi masalah atau mungkin juga tidak, tergantung pada apa yang ingin Anda lakukan. Kami sebagian besar telah menggunakan pembuatan PDF untuk membuat surat-surat yang banyak hal harus diganti sebaris. FDF tidak berfungsi dengan baik untuk tujuan ini. Jika Anda mengisi otomatis formulir pajak daring, misalnya, ini tidak akan menjadi masalah.

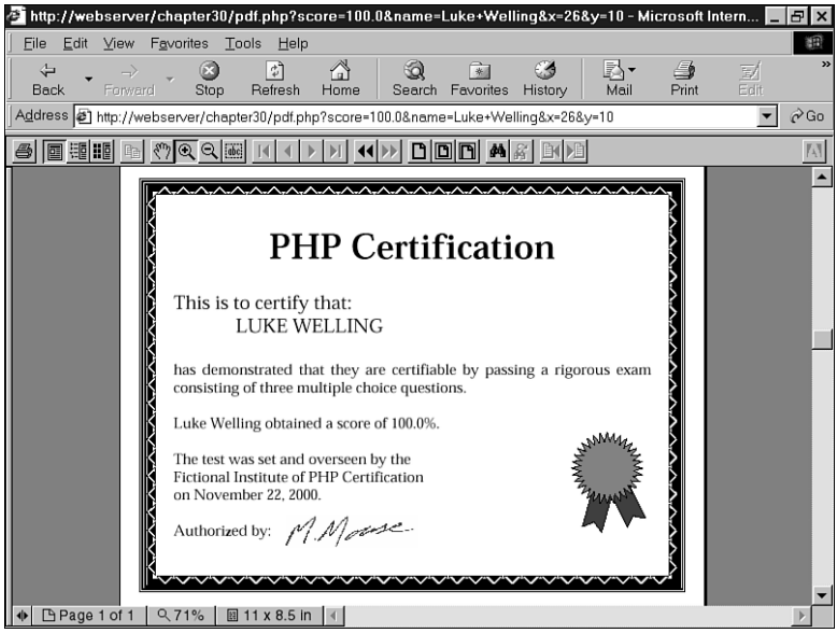
Sekarang kita beralih ke solusi PDF untuk masalah sebelumnya. Kita masih dapat menemukan dan mengganti placeholder dalam berkas PDF kita jika kita menyadari bahwa kode format tambahan hanya terdiri dari tanda hubung, angka, dan tanda kurung dan karenanya dapat dicocokkan melalui ekspresi reguler. Kita telah menulis sebuah fungsi, `pdf_replace()`, untuk secara otomatis menghasilkan ekspresi reguler yang cocok untuk placeholder dan mengganti placeholder tersebut dengan teks yang sesuai. Selain penambahan ini, kode untuk menghasilkan sertifikat melalui templat PDF sangat mirip dengan versi RTF. Skrip ini ditampilkan dalam Listing 9.4.

Listing 9.4 pdf.php—Skrip untuk Menghasilkan Sertifikat PDF yang Dipersonalisasi Melalui Templat

```
<?
  set_time_limit( 180 ); // this script can be very slow

  function pdf_replace( $pattern, $replacement, $string )
  {
    $len = strlen( $pattern );
    $regexp = '';
    for ( $i = 0; $i<$len; $i++ )
    {
      $regexp .= $pattern[$i];
      if ($i<$len-1)
        $regexp .= "(\)\-{0,1}[0-9]*\(){0,1}";
    }
    return ereg_replace ( $regexp, $replacement, $string );
    if(!$name||!$score)
    {
      echo "<h1>Error:</h1>This page was called incorrectly";
    }
    else
    {
      //generate the headers to help a browser choose the correct application
      header( "Content-Disposition: filename=cert.pdf");
      header( "Content-type: application/pdf" );

      $date = date( "F d, Y" );

      // open our template file
      $filename = "PHPCertification.pdf";
      $fp = fopen ( $filename, "r" );
```

```
        //read our template into a variable
        $output = fread( $fp, filesize( $filename ) );

        fclose ( $fp );

        // replace the place holders in the template with our data
        $output = pdf_replace( "<<NAME>>", strtoupper( $name ), $output );
        $output = pdf_replace( "<<Name>>", $name, $output );
        $output = pdf_replace( "<<score>>", $score, $output );
        $output = pdf_replace( "<<mm/dd/yyyy>>", $date, $output );

        // send the generated document to the browser
        echo $output;
    }
?>
```

Skrip ini menghasilkan versi khusus dari dokumen PDF kami. Dokumen yang ditunjukkan pada Gambar 9.6 akan dapat dicetak dengan baik di berbagai sistem, dan lebih sulit bagi penerima untuk mengubah atau mengeditnya. Anda dapat melihat bahwa dokumen PDF pada Gambar 9.6 tampak hampir sama persis dengan dokumen RTF pada Gambar 9.5.



Gambar 9.6 pdf.php menghasilkan sertifikat dari templat PDF.

Satu masalah dengan pendekatan ini adalah kode berjalan cukup lambat karena pencocokan ekspresi reguler diperlukan. Ekspresi reguler berjalan jauh lebih lambat daripada `str_replace()` yang dapat kita gunakan untuk versi RTF.

Jika Anda akan mencocokkan sejumlah besar placeholder atau mencoba menghasilkan banyak dokumen ini di server yang sama, Anda mungkin ingin melihat pendekatan lain. Ini

akan menjadi masalah yang lebih kecil untuk templat yang lebih sederhana. Sebagian besar data dalam berkas ini adalah data yang mewakili gambar.

9.6 MENGHASILKAN DOKUMEN PDF MENGGUNAKAN PDFLIB

PDFlib ditujukan untuk menghasilkan dokumen PDF dinamis melalui Web. Ini bukan bagian dari PHP, melainkan pustaka terpisah, dengan sejumlah besar fungsi yang dimaksudkan untuk dipanggil dari berbagai bahasa pemrograman. Pengikatan bahasa tersedia untuk C, C++, Java, Perl, Python, Tcl, dan ActiveX/COM.

Yang menarik, belum ada pengikatan PHP PDFlib resmi yang tersedia. Pengikatan saat ini tidak didokumentasikan atau didukung oleh PDFlib. Situs Web PDFlib menyatakan, "Pengikatan bahasa lain, seperti PHP, akan didukung di masa mendatang," tetapi telah menyatakan hal ini setidaknya selama satu tahun. Meskipun ada kemungkinan bahwa pengikatan PHP resmi mungkin lebih baik daripada yang sekarang ketika (atau jika) tiba, yang sekarang sangat bagus.

Skrip Hello World untuk PDFlib

Setelah Anda memiliki PHP dan menginstalnya dengan PDFlib yang diaktifkan, Anda dapat mengujinya dengan program sederhana seperti contoh Hello World dalam Listing 9.5.

Listing 9.5 `testpdf.php`—Contoh Hello World Klasik Menggunakan PDFlib Melalui PHP

```
<?
//create file
  $fp = fopen("hello.pdf", "w");
  if(!$fp)
  {
    echo "Error: could not create the PDF file";
    exit;
  }

  // start the pdf document
  $pdf = pdf_open($fp);
  pdf_set_info($pdf, "Creator", "pdftest.php");
  pdf_set_info($pdf, "Author", "Luke Welling and Laura Thomson");
  pdf_set_info($pdf, "Title", "Hello World (PHP)");

  // US letter is 11" x 8.5" and there are approximately 72 points per inch
  pdf_begin_page($pdf, 8.5*72, 11*72);
  pdf_add_outline($pdf, "Page 1");
  pdf_set_font($pdf, "Helvetica-Bold", 24, "host");
  pdf_set_text_pos($pdf, 50, 700);

  // write text
  pdf_show($pdf,"Hello,world!");
  pdf_continue_text($pdf,"(says PHP)");
```

```
   // end the document
   pdf_end_page($pdf);
   pdf_close($pdf);
   fclose($fp);

   // display a link to download
   echo “download the pdf <a href = ‘hello.pdf’>here</a>”;
?>
```

Kesalahan paling mungkin yang akan Anda lihat jika skrip ini gagal adalah sebagai berikut:

```
Fatal error: Call to undefined function: pdf_open() in
/home/book/public_html/chapter30/pdftest.php on line 6
```

Ini berarti Anda tidak memiliki ekstensi PDFlib yang dikompilasi ke dalam PHP.

Instalasinya cukup mudah, tetapi beberapa detail berubah tergantung pada versi PHP dan PDFlib yang Anda gunakan. Tempat yang bagus untuk memeriksa saran terperinci adalah catatan yang disumbangkan pengguna pada halaman PDFlib dalam manual PHP yang diberi anotasi.

Saat skrip ini aktif dan berjalan di sistem Anda, sekarang saatnya untuk melihat cara kerjanya. Bagian pertama kode, termasuk baris

```
$fp = fopen(“hello.pdf”, “w”);
```

membuat file yang dapat ditulis. Perlu dicatat bahwa kode di sini menulis langsung ke direktori saat ini meskipun kita telah membahas sejumlah alasan mengapa merupakan ide yang buruk untuk menyiapkan izin Anda guna mengizinkan PHP menulis di dalam pohon Web.

Baris `$pdf = pdf_open($fp);` menginisialisasi dokumen PDF menggunakan file yang telah kita buka. Anda juga dapat memanggil `pdf_open()` tanpa parameter untuk membuat dokumen di memori yang akan langsung ditampilkan ke browser. Dalam hal apa pun, Anda perlu menangkap nilai pengembalian `pdf_open()`, karena setiap panggilan berikutnya ke fungsi PDF akan membutuhkannya.

Fungsi `pdf_set_info()` memungkinkan Anda memberi tag dokumen dengan subjek, judul, pembuat, penulis, daftar kata kunci, dan satu bidang khusus yang ditentukan pengguna. Di sini kita menetapkan pembuat, penulis, dan judul. Perhatikan bahwa keenam bidang info bersifat opsional.

```
pdf_set_info($pdf, “Creator”, “pdftest.php”);
pdf_set_info($pdf, “Author”, “Luke Welling and Laura Thomson”);
pdf_set_info($pdf, “Title”, “Hello World (PHP)”);
```

Dokumen PDF terdiri dari sejumlah halaman. Untuk memulai halaman baru, kita perlu memanggil `pdf_begin_page()`. Selain pengenal yang dikembalikan oleh `pdf_open()`, `pdf_begin_page()` memerlukan dimensi halaman. Setiap halaman dalam dokumen dapat memiliki ukuran yang berbeda, tetapi kecuali Anda memiliki alasan kuat untuk tidak melakukannya, Anda harus menggunakan ukuran kertas yang umum.

PDFlib berfungsi dalam poin, baik untuk ukuran halaman, maupun untuk menemukan lokasi koordinat pada setiap halaman. Sebagai referensi, A4 berukuran sekitar 595 x 842 poin dan ukuran letter AS berukuran 612 x 792 poin. Ini berarti bahwa baris kita

```
pdf_begin_page($pdf, 8.5*72, 11*72);
```

membuat halaman dalam dokumen kita, berukuran untuk kertas letter AS.

Dokumen PDF tidak perlu hanya berupa dokumen yang dapat dicetak. Banyak fitur PDF yang dapat disertakan dalam dokumen seperti hyperlink dan bookmark. Fungsi `pdf_add_outline()` akan menambahkan bookmark ke kerangka dokumen. Bookmark dalam dokumen akan muncul di panel terpisah di Acrobat Reader, yang memungkinkan kita untuk langsung melompat ke bagian penting.

Baris ini `pdf_add_outline($pdf, "Page 1");` menambahkan entri kerangka berlabel Halaman 1, yang akan merujuk ke halaman saat ini. Font yang tersedia pada sistem bervariasi dari sistem operasi ke sistem operasi dan bahkan dari mesin ke mesin. Untuk menjamin hasil yang konsisten, satu set font inti akan berfungsi dengan setiap pembaca PDF. Ke-14 font inti adalah

- Courier
- Courier-Bold
- Courier-Oblique
- Courier-BoldOblique
- Helvetica
- Helvetica-Bold
- Helvetica-Oblique
- Helvetica-BoldOblique
- Times-Roman
- Times-Bold
- Times-Italic
- Times-BoldItalic
- Symbol
- ZapfDingbats

Font di luar set ini dapat disematkan dalam dokumen, tetapi ini akan meningkatkan ukuran file dan mungkin tidak dapat diterima di bawah lisensi apa pun yang Anda miliki untuk font tertentu tersebut. Kita dapat memilih font, ukurannya, dan pengodean karakter sebagai berikut:

```
pdf_set_font($pdf, "Helvetica-Bold", 24, "host");
```

Ukuran font ditentukan dalam poin. Kami telah memilih pengodean karakter host. Nilai yang diizinkan adalah winansi, builtin, macroman, ebcdic, atau host. Arti dari berbagai nilai tersebut adalah sebagai berikut:

- winansi—ISO 8859-1 ditambah karakter khusus yang ditambahkan oleh Microsoft seperti simbol Euro.
- macroman—Pengodean Mac Roman. Set karakter Macintosh default.
- ebcdic—EBCDIC seperti yang digunakan pada sistem IBM AS/400.
- builtin—Gunakan pengodean yang ada di dalam font. Biasanya digunakan dengan font dan simbol non-Latin.
- host—Secara otomatis memilih macroman pada Mac, ebcdic pada sistem berbasis EBCDIC, dan winansi pada semua sistem lainnya.

Jika Anda tidak perlu menyertakan karakter khusus, pilihan pengodean tidaklah penting.

Dokumen PDF tidak seperti dokumen HTML atau dokumen pengolah kata. Teks tidak hanya dimulai di kiri atas dan mengalir ke baris lain sesuai kebutuhan. Kita perlu memilih tempat untuk meletakkan setiap baris teks. Seperti yang telah disebutkan, PDF menggunakan titik untuk menentukan lokasi. Titik asal (koordinat x, y [0, 0]) berada di sudut kiri bawah halaman.

Mengingat halaman kita berukuran 612 x 792 poin, titik (50, 700) berada sekitar dua pertiga inci dari kiri halaman dan sekitar sepertiga inci dari atas. Untuk mengatur posisi teks kita di titik ini, kita menggunakan `pdf_set_text_pos($pdf, 50, 700);`

Terakhir, setelah menyiapkan halaman, kita dapat menulis beberapa teks di atasnya. Untuk menambahkan teks di posisi saat ini menggunakan font saat ini, kita menggunakan `pdf_show()`.

Baris `pdf_show($pdf,"Hello,world!");` menambahkan tes "Hello World!" ke dokumen kita. Untuk pindah ke baris berikutnya dan menulis lebih banyak teks, kita menggunakan `pdf_continue_text()`. Untuk menambahkan string "(Says PHP)", kita menggunakan `pdf_continue_text($pdf,"(says PHP)");`

Lokasi pasti tempat elemen ini akan muncul bergantung pada jenis huruf dan ukuran yang dipilih. Setelah selesai menambahkan elemen ke halaman, kita perlu memanggil `pdf_end_page()` sebagai berikut:

```
pdf_end_page($pdf);
```

Setelah kita menyelesaikan seluruh dokumen PDF, kita perlu menutupnya menggunakan `pdf_close()`. Saat kita membuat file, kita juga perlu menutup file tersebut.
Baris `pdf_close($pdf); fclose($fp);` menyelesaikan pembuatan dokumen Hello World kita. Yang perlu kita lakukan adalah menyediakan cara untuk mengunduhnya.

```
echo "download pdf <a href = 'hello.pdf'>here</a>";
```

Contoh ini diambil dari contoh bahasa C dalam dokumentasi PDFlib dan seharusnya menyediakan titik awal. Dokumen yang ingin kita buat untuk sertifikat lebih rumit, termasuk bingkai, gambar vektor, dan gambar bitmap. Dengan dua teknik lainnya, kita menambahkan fitur-fitur ini menggunakan pengolah kata kita. Dengan PDFlib, kita harus menambahkannya secara manual.

9.6 MEMBUAT SERTIFIKAT KITA DENGAN PDFLIB

Untuk menggunakan PDFlib, kita telah memilih untuk membuat beberapa kompromi. Meskipun hampir pasti memungkinkan untuk menduplikasi sertifikat yang kita gunakan sebelumnya, diperlukan upaya lebih besar untuk membuat dan memposisikan setiap elemen secara manual daripada menggunakan alat seperti Microsoft Word untuk membantu menata dokumen.

Kita menggunakan teks yang sama seperti sebelumnya, termasuk roset merah dan tanda tangan bitmap, tetapi kita tidak akan menduplikasi batas yang rumit. Kode lengkap untuk skrip ini ditunjukkan pada Listing 9.6.

Listing 9.6 `pdflib.php`—Membuat Sertifikat Kita Menggunakan PDFlib

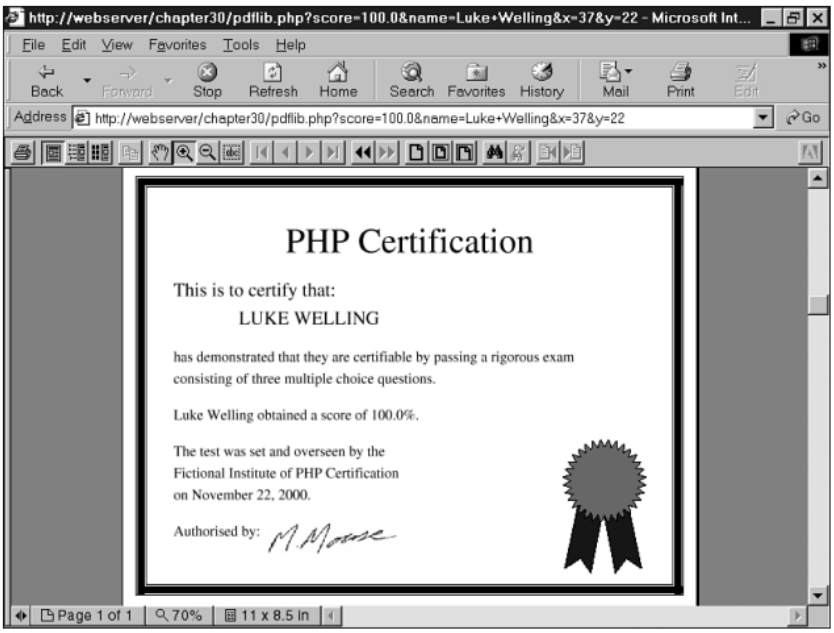
```
<?
  if(!$name||!$score)
  {
    echo “<h1>Error:</h1>This page was called incorrectly”;
  }
  else
  {
    //generate the headers to help a browser choose the correct application
    header( “Content-type: application/pdf” );
    header( “Content-Disposition: filename=cert.pdf”);

    $date = date( “F d, Y” );

    // create a pdf document in memory
    $pdf = pdf_open();

    // set up the page size in points
    // US letter is 11” x 8.5”
    // there are approximately 72 points per inch
    $width = 11*72;
    $height = 8.5*72;

    pdf_begin_page($pdf, $width, $height);

    // draw a borders

    $inset = 20; // space between border and page edge
    $border = 10; // width of main border line
    $inner = 2; // gap within the border

    //draw outer border
    pdf_rect($pdf,$inset-$inner,
                  $inset-$inner,
                  $width-2*($inset-$inner),
```

```
                $height-2*($inset-$inner));
pdf_stroke($pdf);

//draw main border $border points wide
pdf_setlinewidth($pdf, $border);
pdf_rect($pdf,$inset+$border/2,
                $inset+$border/2,
                $width-2*($inset+$border/2),
                $height-2*($inset+$border/2));
pdf_stroke($pdf);
pdf_setlinewidth($pdf, 1.0);

//draw inner border
pdf_rect($pdf,$inset+$border+$inner,
                $inset+$border+$inner,
                $width-2*($inset+$border+$inner),
                $height-2*($inset+$border+$inner));
pdf_stroke($pdf);

//add text
pdf_set_font($pdf, "Times-Roman", 48, "host");

$startx = ($width - pdf_stringwidth($pdf, "PHP Certification"))/2;
pdf_show_xy($pdf, "PHP Certification", $startx, 490);

pdf_set_font($pdf, "Times-Roman", 26, "host");
                    $startx = 70;

pdf_show_xy($pdf, "This is to certify that:", $startx, 430);
pdf_show_xy($pdf, strtoupper($name), $startx+90, 391);

pdf_set_font($pdf, "Times-Roman", 20, "host");

pdf_show_xy($pdf,  "has demonstrated that they are certifiable ".
                    "by passing a rigorous exam", $startx, 340);
pdf_show_xy($pdf,  "consisting of three multiple choice questions.",
                    $startx, 310);

pdf_show_xy($pdf,  "$name obtained a score of $score"."%.", $startx, 260);
pdf_show_xy($pdf, "The test was set and overseen by the ", $startx, 210);
pdf_show_xy($pdf, "Fictional Institute of PHP Certification",
                    $startx, 180);

pdf_show_xy($pdf, "on $date.", $startx, 150);
pdf_show_xy($pdf, "Authorised by:", $startx, 100);
```

```
// add bitmap signature image
$signature = pdf_open_image_file($pdf, "tiff",
                    "/htdocs/book/chapter30/signature.tif");
pdf_place_image($pdf, $signature, 200, 75, 1);
pdf_close_image($pdf, $signature);

pdf_setrgbcolor_fill($pdf, 0, 0, .4); //dark blue

pdf_setrgbcolor_stroke($pdf, 0, 0, 0); // black

// draw ribbon 1
pdf_moveto($pdf, 630, 150);
pdf_lineto($pdf, 610, 55);
pdf_lineto($pdf, 632, 69);
pdf_lineto($pdf, 646, 49);
pdf_lineto($pdf, 666, 150);
pdf_closepath($pdf);
pdf_fill($pdf);

// outline ribbon 1
pdf_moveto($pdf, 630, 150);
pdf_lineto($pdf, 610, 55);
pdf_lineto($pdf, 632, 69);
pdf_lineto($pdf, 646, 49);
pdf_lineto($pdf, 666, 150);
pdf_closepath($pdf);
pdf_stroke($pdf);

// draw ribbon 2
pdf_moveto($pdf, 660, 150);
pdf_lineto($pdf, 680, 49);
pdf_lineto($pdf, 695, 69);
pdf_lineto($pdf, 716, 55);
pdf_lineto($pdf, 696, 150);
pdf_closepath($pdf);
pdf_fill($pdf);

// outline ribbon 2
pdf_moveto($pdf, 660, 150);
pdf_lineto($pdf, 680, 49);
pdf_lineto($pdf, 695, 69);
pdf_lineto($pdf, 716, 55);
pdf_lineto($pdf, 696, 150);
pdf_closepath($pdf);
pdf_stroke($pdf);

pdf_setrgbcolor_fill($pdf, .8, 0, 0); //red
```

```
    //draw rosette
    draw_star(665, 175, 32, 57, 10, $pdf, true);

    //outline rosette
    draw_star(665, 175, 32, 57, 10, $pdf, false);

    pdf_end_page($pdf);
    pdf_close($pdf);
  }

  function draw_star($centerx, $centery, $points, $radius,
                     $point_size, $pdf, $filled)
  {
    $inner_radius = $radius-$point_size;

    for ($i = 0; $i<=$points*2; $i++ )
    {
      $angle= ($i*2*pi())/($points*2);

      if($i%2)
      {
        $x = $radius*cos($angle) + $centerx;
        $y = $radius*sin($angle) + $centery;
      }
      else
      {
        $x = $inner_radius*cos($angle) + $centerx;
        $y = $inner_radius*sin($angle) + $centery;
      }
      if($i==0)
        pdf_moveto($pdf, $x, $y);
      else if($i==$points*2)
        pdf_closepath($pdf);
      else
        pdf_lineto($pdf, $x, $y);
    }
    if($filled)
      pdf_fill($pdf);
    pdf_stroke($pdf);
  }
?>
```

Sertifikat yang dihasilkan menggunakan skrip ini ditunjukkan pada Gambar 9.7. Seperti yang dapat Anda lihat, sertifikat ini cukup mirip dengan yang lain, kecuali bahwa batasnya lebih sederhana dan bintangnya terlihat sedikit berbeda. Ini karena kami telah menggambarnya ke dalam dokumen daripada menggunakan berkas clip art yang sudah ada.



Gambar 9.7 `pdflib.php` menggambar sertifikat ke dalam dokumen PDF.

Kita akan melihat beberapa bagian skrip ini yang berbeda dari contoh sebelumnya. Pengunjung perlu mendapatkan detail mereka sendiri pada sertifikat, jadi kita akan membuat dokumen di memori, bukan di berkas. Jika kita menulisnya ke berkas, kita perlu memikirkan mekanisme untuk membuat nama berkas yang unik, mencegah orang mengintip sertifikat orang lain, dan menentukan cara untuk menghapus berkas sertifikat lama guna mengosongkan ruang hard drive di server kita. Untuk membuat dokumen di memori, kita memanggil `pdf_open()` tanpa parameter sebagai berikut:

```
$pdf = pdf_open();
```

Batas yang disederhanakan akan terdiri dari tiga garis: batas tebal dan dua batas tipis, satu di dalam batas utama dan satu di luar. Kita akan menggambar semuanya sebagai persegi panjang. Untuk memposisikan batas sedemikian rupa sehingga kita dapat dengan mudah mengubah ukuran halaman atau tampilan batas, kita akan mendasarkan semua posisi batas pada variabel yang sudah kita miliki, `$width` dan `$height` dan beberapa yang baru: `$inset`, `$border`, dan `$inner`. Kita akan menggunakan $inset untuk menentukan berapa banyak titik lebar border di tepi halaman, `$border` untuk menentukan ketebalan border utama, dan `$inner` untuk menentukan seberapa lebar celah antara border utama dan border tipis.

Jika Anda telah menggambar dengan API grafik lain, menggambar dengan PDFlib akan menghadirkan beberapa kejutan. Border tipis itu mudah. Untuk membuat persegi panjang, kita menggunakan `pdf_rect()`, yang memerlukan parameter pengenal dokumen PDF, koordinat `x` dan `y` sudut kiri bawah persegi panjang, dan lebar dan tinggi persegi panjang. Karena kita ingin tata letak kita fleksibel, kita menghitungnya dari variabel yang telah kita tetapkan.

```
pdf_rect($pdf,$inset-$inner,
              $inset-$inner,
              $width-2*($inset-$inner),
              $height-2*($inset-$inner));
```

Panggilan ke pdf_rect() menyiapkan jalur berbentuk persegi panjang. Untuk menggambar bentuk tersebut, kita perlu memanggil fungsi pdf_stroke() sebagai berikut:

```
pdf_stroke($pdf);
```

Untuk menggambar batas utama, kita perlu menentukan lebar garis. Lebar garis default adalah 1 poin. Panggilan ke pdf_setlinewidth() berikut akan menyetelnya ke $border (dalam kasus ini 10) poin:

```
pdf_setlinewidth($pdf, $border);
```

Setelah lebar ditetapkan, kita kembali membuat persegi panjang dengan pdf_rect() dan memanggil pdf_stroke() untuk menggambarnya.

```
pdf_rect( $pdf, $inset+$border/2,
          $inset+$border/2,
          $width-2*($inset+$border/2),
          $height-2*($inset+$border/2));
          pdf_stroke($pdf);
```

Setelah kita menggambar satu garis lebar, kita perlu mengingat untuk mengatur kembali lebar garis ke 1 dengan kode ini:

```
pdf_setlinewidth($pdf, 1.0);
```

Kita akan menggunakan pdf_show_xy() untuk memposisikan setiap baris teks pada sertifikat. Untuk sebagian besar baris teks, kita menggunakan margin kiri yang dapat dikonfigurasi ($startx) sebagai koordinat x dan nilai yang dipilih oleh mata sebagai koordinat y.

Karena kita ingin judul berada di tengah halaman, kita perlu mengetahui lebarnya untuk memposisikan sisi kirinya. Kita bisa mendapatkan lebarnya menggunakan pdf_stringwidth(). Panggilan pdf_stringwidth($pdf, "PHP Certification"); akan mengembalikan lebar string "PHP Certification" dalam font dan ukuran font saat ini.

Seperti versi sertifikat lainnya, kita akan menyertakan tanda tangan sebagai bitmap yang dipindai. Tiga pernyataan berikut

```
$signature = pdf_open_image_file($pdf, "tiff",
                                 "/htdocs/book/chapter30/signature.tif");
pdf_place_image($pdf, $signature, 200, 75, 1);
```

```
pdf_close_image($pdf, $signature);
```

akan membuka berkas TIFF yang berisi tanda tangan, menambahkan gambar ke halaman di lokasi yang ditentukan, dan menutup berkas TIFF. Jenis berkas lain juga dapat digunakan. Satu-satunya parameter yang mungkin tidak dapat dijelaskan sendiri adalah parameter kelima untuk `pdf_place_image()`. Fungsi ini tidak terbatas untuk memasukkan gambar pada ukuran aslinya. Parameter kelima adalah faktor skala. Kami telah memilih untuk menampilkan gambar dalam ukuran penuh dan menggunakan 1 sebagai faktor skala, tetapi dapat menggunakan angka yang lebih besar untuk memperbesar gambar, atau pecahan untuk mengecilkannya.

Item tersulit untuk ditambahkan ke sertifikat kami menggunakan PDFlib adalah roset. Kami tidak dapat secara otomatis membuka dan menyertakan Berkas Meta Windows yang berisi roset yang telah kami miliki, tetapi kami bebas menggambar bentuk apa pun yang kami suka.

Untuk menggambar bentuk yang diisi seperti salah satu pita, kami dapat menulis kode berikut. Di sini kami mengatur warna guratan atau garis menjadi hitam dan warna isian atau bagian dalam menjadi biru tua:

```
pdf_setrgbcolor_fill($pdf, 0, 0, .4); //dark blue
pdf_setrgbcolor_stroke($pdf, 0, 0, 0); // black
```

Di sini kita menyiapkan poligon bersisi lima untuk menjadi salah satu pita kita dan kemudian mengisinya:

```
pdf_moveto($pdf, 630, 150);
pdf_lineto($pdf, 610, 55);
pdf_lineto($pdf, 632, 69);
pdf_lineto($pdf, 646, 49);
pdf_lineto($pdf, 666, 150);
pdf_closepath($pdf);
pdf_fill($pdf);
```

Karena kita ingin poligon tersebut juga digariskan, kita perlu menyiapkan lintasan yang sama untuk kedua kalinya, tetapi panggil `pdf_stroke()` alih-alih `pdf_fill()`.

Karena bintang multititik merupakan bentuk berulang yang kompleks, kita telah menulis sebuah fungsi untuk menghitung lokasi di lintasan tersebut bagi kita. Fungsi kita disebut `draw_star()` dan memerlukan koordinat x dan y untuk pusat, jumlah titik yang diperlukan, radius, panjang titik, pengenal dokumen PDF, dan nilai Boolean untuk menunjukkan apakah bentuk bintang tersebut harus diisi atau hanya garis luar.

Fungsi `draw_star()` menggunakan beberapa trigonometri dasar untuk menghitung lokasi serangkaian titik guna menyusun bintang. Untuk setiap titik yang kita minta agar ada di

bintang kita, kita temukan sebuah titik pada radius bintang dan sebuah titik pada lingkaran yang lebih kecil `$point_size` di dalam lingkaran luar dan buat garis di antara keduanya.

Satu hal yang perlu diperhatikan adalah bahwa fungsi trigonometri PHP seperti `cos()` dan `sin()` bekerja dalam radian, bukan derajat. Dengan menggunakan fungsi dan matematika, kita dapat secara akurat menghasilkan bentuk berulang yang kompleks. Jika kita menginginkan pola yang rumit untuk batas halaman kita, kita dapat menggunakan pendekatan yang sama. Ketika semua elemen halaman kita dihasilkan, kita perlu mengakhiri halaman dan dokumen.

Masalah dengan Header

Satu hal kecil yang perlu diperhatikan dalam semua skrip ini adalah kita perlu memberi tahu browser jenis data apa yang akan kita kirimkan kepadanya. Kita telah melakukan ini dengan mengirimkan header HTTP tipe konten, misalnya

```
header( “Content-type: application/msword” );
```

atau

```
header( “Content-type: application/pdf” );
```

Satu hal yang perlu diperhatikan adalah bahwa browser menangani header ini secara tidak konsisten. Secara khusus, Internet Explorer sering memilih untuk mengabaikan tipe MIME dan mencoba mendeteksi tipe file secara otomatis.

Beberapa header kami tampaknya menyebabkan masalah dengan header kontrol sesi. Ada beberapa cara untuk mengatasinya. Kami menemukan bahwa menggunakan parameter GET daripada parameter POST atau variabel sesi dapat menghindari masalah tersebut.

Solusi lain adalah tidak menggunakan PDF sebaris, tetapi meminta pengguna untuk mengunduhnya seperti yang ditunjukkan dalam contoh Hello World PDFlib. Anda juga dapat menghindari masalah jika Anda bersedia menulis dua versi kode yang sedikit berbeda, satu untuk Netscape dan satu untuk Internet Explorer.

Masalah ini merupakan masalah yang diketahui dengan kombinasi file PDF yang dibuat secara dinamis di Internet Explorer dan plug-in Acrobat Reader.

DAFTAR PUSTAKA

Adam, S. I., & Andolo, S. (2019, August). A new PHP web application development framework based on MVC architectural pattern and ajax technology. In 2019 1st International Conference on Cybernetics and Intelligent System (ICORIS) (Vol. 1, pp. 45-50). IEEE.

Beighley, L., & Morrison, M. (2008). Head First PHP & MySQL: A Brain-Friendly Guide. O'Reilly Media.

Bhardwaj, H. (2021). PHP Mysql For Advanced Learning. Booksclinic Publishing.

Blum, R. (2025). PHP, MYSQL, & JavaScript All-in-one for Dummies. John Wiley & Sons.

Chen, X., & Zhang, J. (2021, December). The Applications PHP, HTML and MYSQL in Development of Website–Query Function. In ICMLCA 2021; 2nd International Conference on Machine Learning and Computer Application (pp. 1-4). VDE.

Cobo, Á. (2005). PHP y MySQL: Tecnología para el desarrollo de aplicaciones web. Ediciones Díaz de Santos.

Connolly, R., Hoar, R., Mukherjee, S., & Bhattacharjee, A. K. (2015). Fundamentals of web development. Pearson.

Curioso, A., Bradford, R., & Galbraith, P. (2010). Expert PHP and MySQL. John Wiley & Sons.

Dutonde, P. D., Mamidwar, S. S., Korvate, M. S., Bafna, S., & Shirbhate, D. D. (2022). Website developmemt technologies: A review. *Int. J. Res. Appl. Sci. Eng. Technol*, *10*(1), 359-366.

Gerner, J., & Naramore, E. (2005). Professional Lamp Linux, Apache, Mysql, & Php 5 Web Development. John Wiley & Sons.

Gilmore, W. J. (2006). Beginning PHP and MySQL 5: From novice to professional. Apress.

Gilmore, W. J. (2008). Beginning PHP and MySQL: from novice to professional. Berkeley, CA: Apress.

Glass, M. K., Le Scouarnec, Y., Naramore, E., Mailer, G., Stolz, J., & Gerner, J. (2004). *Beginning PHP, Apache, MySQL Web Development*. John Wiley & Sons.

Greenspan, J., & Bulger, B. (2001). MySQL/PHP database applications. John Wiley & Sons, Inc..

Guion, A., Burton, K., Hodgkins, H., Morris Jr, D., & Wood, M. M. J. (2015, December). Implementation of an interactive database interface utilizing HTML, PHP, JavaScript, and MySQL in support of water quality assessments in the Northeastern North Carolina Pasquotank Watershed. In AGU Fall Meeting Abstracts (Vol. 13).

Korhonen, K., Donadini, F., Riisager, P., & Pesonen, L. J. (2008). GEOMAGIA50: an archeointensity database with PHP and MySQL. Geochemistry, Geophysics, Geosystems, 9(4).

Kromann, F. M. (2018). Beginning PHP and MySQL: from novice to professional. Apress.

Kurien, A. T. J., Mathew, S. A., & Mana, S. C. (2022, April). Development of PHP and MySQL based Digital Asset Management System for Secure Organizations. In 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 1859-1863). IEEE.

Lee, J., & Ware, B. (2003). Open Source Web Development with LAMP: Using Linux, Apache, MySQL, Perl, and PHP. Addison-Wesley Professional.

Lerdorf, R., Tatroe, K., & MacIntyre, P. (2006). Programming Php. " O'Reilly Media, Inc.".

Matthews, M. (2014). PHP and MySQL web development: A beginner's guide. McGraw-Hill Education Group.

Mishra, A. (2014). Critical comparison of PHP and ASP .NET for web development. International Journal of Scientific & Technology Research, 3(7), 331-333.

Mishra, D. P., Rout, K. K., & Salkuti, S. R. (2021). Modern tools and current trends in web-development. Indonesian Journal of Electrical Engineering and Computer Science, 24(2), 978-985.

Naramore, E., Gerner, J., Scouarnec, Y. L., Stolz, J., & Glass, M. K. (2015). Beginning PHP5, Apache, and MySQL Web Development.

Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. " O'Reilly Media, Inc.".

Nixon, R. (2021). Learning PHP, MySQL & JavaScript: a step-by-step guide to creating dynamic websites. " O'Reilly Media, Inc.".

Noviana, R. (2022). Pembuatan aplikasi penjualan berbasis web monja store menggunakan php dan mysql. Jurnal Teknik Dan Science, 1(2), 112-124.

Prokofyeva, N., & Boltunova, V. (2017). Analysis and practical application of PHP frameworks in development of web information systems. Procedia Computer Science, 104, 51-56.

Rochkind, M. (2013). Expert PHP and MySQL: application design and development. Apress.

Sotnik, S., Manakov, V., & Lyashenko, V. (2023). Overview: PHP and MySQL features for creating modern web projects.

Stobart, S., & Vassileiou, M. (2012). PHP and MySQL Manual: simple, yet powerful Web programming. Springer Science & Business Media.

Stoeva, M. (2014). Interactive Multimedia tool for dynamic generation of web interfaces with HTML5/PHP/MySQL and JavaScript. International Journal of Emerging Technology & Advanced Engineering, 4(9), 412-418.

Suehring, S., & Valade, J. (2013). PHP, MySQL, Javascript & HTML5 all-in-one for Dummies. John Wiley & Sons.

Surjandari, I., Rachman, A., Panjaitan, Y. A. B., & Rosyidah, A. (2017, October). Development of theses categorization system search engine using PHP and MySQL. In 2017

International Conference on Information Technology Systems and Innovation (ICITSI) (pp. 194-199). IEEE.

Trichkova, E. (2005, February). Application of PHP and MySQL for search and retrieval Web services in Web information systems. In Proceedings of First International Conference on Information Systems & Datagrids, Sofia, Bulgaria.

Ullman, L. (2011). *PHP and MySQL for dynamic Web sites: visual quickpro guide*. Peachpit Press.

Valade, J. (2004). PHP and MySQL for Dummies. John Wiley & Sons.

Vidal-Silva, C., Jiménez, C., Madariaga, E., & Urzúa, L. (2020). Applying PHP codeigniter for easy web development.

Wandschneider, M. (2006). Core Web application development with PHP and MySQL. Pearson Education India.

Welling, L., & Thomson, L. (2003). PHP and MySQL Web development. Sams publishing.

West, A. W. (2014). Practical PHP and MySQL Website Databases: A Simplified Approach. Apress.

West, A. W., & Prettyman, S. (2018). Practical PHP 7, MySQL 8, and MariaDB Website Databases. Apress.

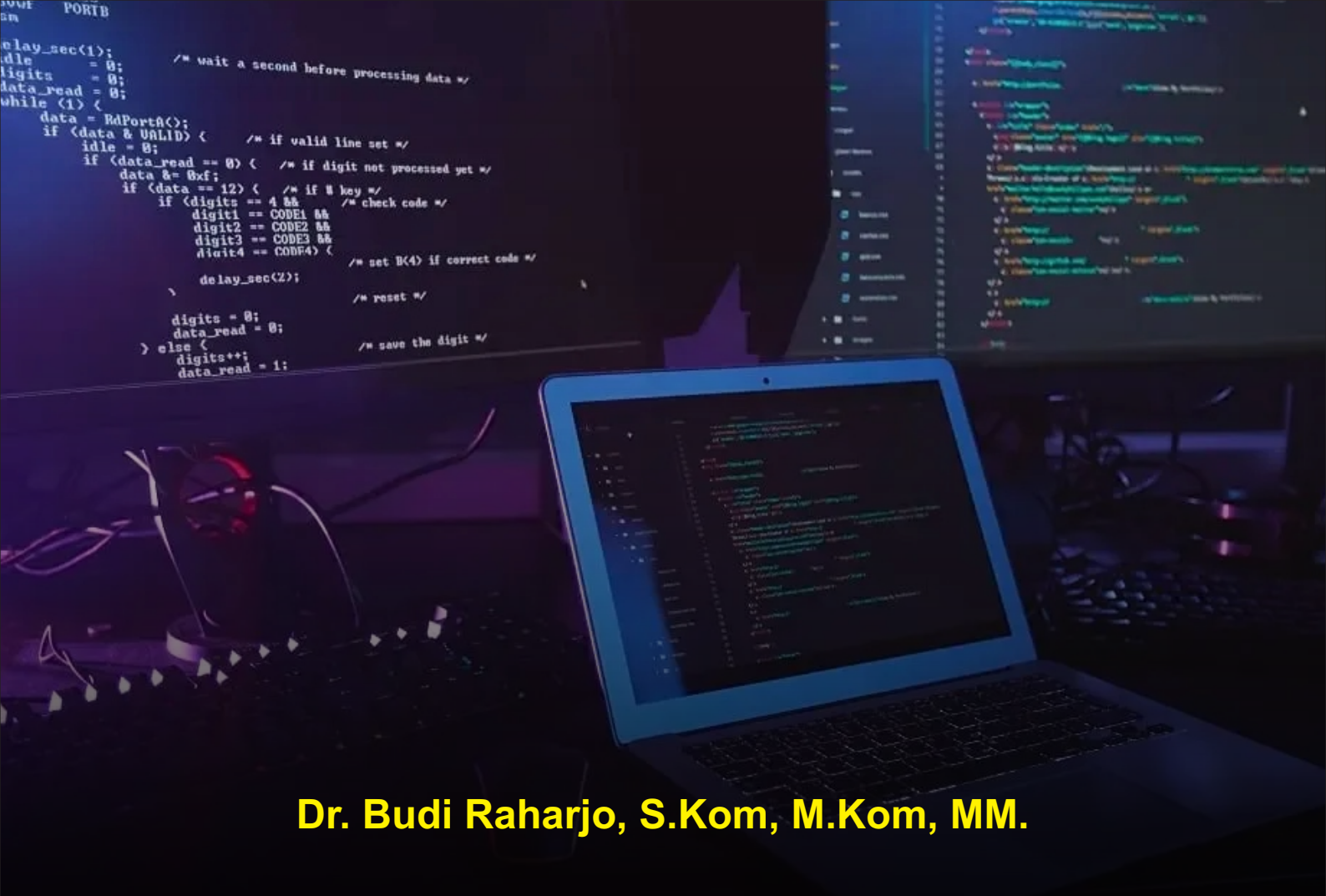
Williams, H. E., & Lane, D. (2004). Web Database Applications with PHP and MySQL: Building Effective Database-Driven Web Sites. " O'Reilly Media, Inc.".

Wu, D. (2022). The application and management system of scientific research projects based on PHP and MySQL. *Journal of Interconnection Networks*, *22*(Supp02), 2143043.

Yadav, N., Rajpoot, D. S., & Dhakad, S. K. (2019, November). LARAVEL: a PHP framework for e-commerce website. In 2019 Fifth International Conference on Image Information Processing (ICIIP) (pp. 503-508). IEEE.

Yank, K. (2004). Build your own database driven website using PHP & MySQL. SitePoint Pty Ltd.

Yu, X., & Yi, C. (2010, November). Design and Implementation of the Website Based on PHP & MYSQL. In *2010 International Conference on E-Product E-Service and E-Entertainment* (pp. 1-4). IEEE.



Dr. Budi Raharjo, S.Kom, M.Kom, MM.

PENGEMBANGAN WEB PHP dan MySQL

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-634-7227-25-6 (PDF)
9 786347 227256