YAYASAN PRIMA AGUS TEKNIK

PEMROGRAMAN WEB DENGAN CODEIGNITER 4

Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD

Dendy Kurniawan, S.Kom., M.Kom

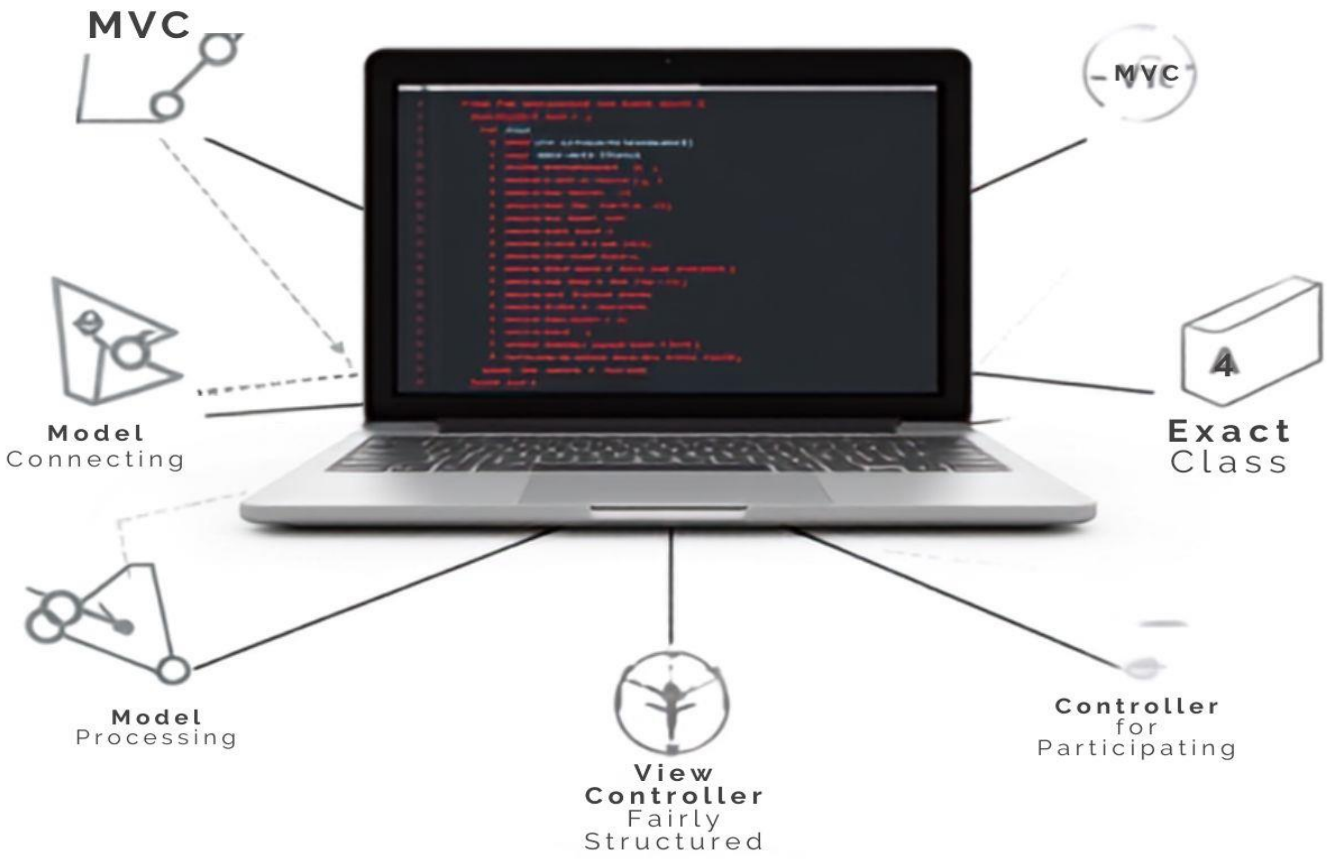
PEMROGRAMAN WEB

DENGAN

CODEIGNITER 4

Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD

Dendy Kurniawan, S.Kom., M.Kom



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 978-634-7695-49-9 (PDF)
9 786347 695499

PEMROGRAMAN WEB DENGAN CODEIGNITER 4
Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD

Penulis:
Dendy Kurniawan, S.Kom., M.Kom

ISBN : 978-634-7695-49-9 (PDF)

Editor:
Edy Siswanto, M.M., M.Kom

Penyunting :
Nur Rokhman S.ST, M.Kom

Desain Sampul dan Tata Letak :
Indra Ava Diantra, S. Kom, M.T

Penerbit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas Stekom)
Tahun 2026

Anggota IKAPI No : 279 / ALB / JTE / 2023

Redaksi :
Jl. Majapahit No 605 Semarang
Telp. 08122925000
Fax . 024-6710144
Email: penerbit_ypat@stekom.ac.id

Distributor Tunggal:
UNIVERSITAS STEKOM
Jl. Majapahit No 605 Semarang
Tlpn. 08122925000
Fax . 024-6710144
Email: info@stekom.ac.id

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, sehingga buku ajar "PEMROGRAMAN WEB DENGAN CODEIGNITER 4: Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD" ini dapat diselesaikan dengan baik.

Buku ini disusun sebagai bahan pembelajaran bagi pembaca dalam memahami dan mempraktikkan pemrograman web menggunakan CodeIgniter 4. Materi disajikan secara bertahap, mulai dari persiapan lingkungan pengembangan, konfigurasi CodeIgniter 4, konsep *Model–View–Controller* (MVC), hingga pengembangan aplikasi CRUD (*Create, Read, Update, Delete*) dengan dukungan MySQL, Bootstrap 5, AJAX, dan jQuery.

Penyajian materi dalam buku ini mengutamakan langkah-langkah praktikum agar pembaca dapat memahami konsep sekaligus menerapkannya secara langsung dalam pengembangan aplikasi web.

Penulis menyadari bahwa buku ini masih memiliki keterbatasan. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan untuk penyempurnaan buku ini di masa mendatang.

Semoga buku ini dapat memberikan manfaat bagi pembaca, dosen, dan pembaca dalam mempelajari pemrograman web dengan CodeIgniter 4.

Semarang, Agustus 2026
Penulis

Dendy Kurniawan, S.Kom., M.Kom

DAFTAR ISI

BAB 1
PERSIAPAN LINGKUNGAN PENGEMBANGAN

Pengembangan aplikasi web tidak hanya membutuhkan kemampuan dalam menulis kode program, tetapi juga memerlukan lingkungan pengembangan (*development environment*) yang sesuai agar seluruh proses pembuatan, pengujian, dan pemeliharaan aplikasi dapat dilakukan secara optimal. Lingkungan pengembangan merupakan kumpulan perangkat lunak yang saling terintegrasi untuk mendukung proses pembangunan aplikasi, mulai dari penyusunan kode program, pengelolaan basis data, hingga proses menjalankan aplikasi pada komputer lokal. Dengan lingkungan pengembangan yang telah dipersiapkan dengan baik, proses pembelajaran menjadi lebih terstruktur dan berbagai kesalahan teknis dapat diminimalkan.

Dalam pengembangan aplikasi menggunakan framework CodeIgniter 4, terdapat beberapa perangkat lunak yang perlu dipersiapkan sebelum proses pembuatan aplikasi dilakukan. Perangkat lunak tersebut meliputi XAMPP sebagai web server lokal, Composer sebagai *dependency manager* untuk PHP, Visual Studio Code sebagai editor kode program, serta CodeIgniter 4 sebagai framework yang digunakan untuk membangun aplikasi berbasis arsitektur *Model–View–Controller* (MVC). Keempat komponen tersebut memiliki peran yang saling melengkapi sehingga membentuk lingkungan pengembangan yang siap digunakan.

XAMPP berfungsi sebagai server lokal yang menyediakan layanan Apache dan MySQL/MariaDB sehingga aplikasi dapat dijalankan tanpa harus menggunakan layanan hosting. Melalui XAMPP, pembaca dapat melakukan proses pengembangan dan pengujian aplikasi secara mandiri pada komputer masing-masing. Selain itu, keberadaan phpMyAdmin yang terdapat di dalam XAMPP juga memudahkan proses pembuatan dan pengelolaan basis data yang akan digunakan pada aplikasi.

Selanjutnya, Composer digunakan sebagai alat untuk mengelola berbagai pustaka (*library*) dan dependensi yang dibutuhkan oleh aplikasi PHP modern. Pada CodeIgniter 4, Composer menjadi salah satu cara yang paling umum digunakan untuk melakukan instalasi framework, memperbarui versi aplikasi, maupun menambahkan paket pendukung lainnya. Penggunaan Composer juga memberikan kemudahan dalam menjaga konsistensi struktur proyek sesuai dengan standar pengembangan aplikasi PHP.

Proses penulisan kode program dilakukan menggunakan Visual Studio Code. Editor kode ini dipilih karena memiliki antarmuka yang sederhana, ringan, serta mendukung berbagai ekstensi yang membantu proses pengembangan aplikasi berbasis PHP. Dukungan fitur seperti *syntax highlighting*, *code completion*, terminal terintegrasi, dan pengelolaan berkas menjadikan Visual Studio Code sebagai salah satu editor yang banyak digunakan oleh pengembang aplikasi web.

Framework CodeIgniter 4 menjadi inti dari seluruh proses praktikum dalam buku ini. Framework ini menerapkan konsep *Model–View–Controller* (MVC) yang bertujuan memisahkan logika program, pengelolaan data, dan tampilan aplikasi sehingga kode program

menjadi lebih rapi, mudah dipelihara, dan mudah dikembangkan. Dengan memahami cara menyiapkan lingkungan pengembangan sejak awal, pembaca akan lebih mudah mengikuti setiap tahapan praktikum pada bab-bab berikutnya.

Melalui bab ini, pembaca akan mempelajari langkah-langkah instalasi seluruh perangkat lunak yang dibutuhkan hingga lingkungan pengembangan siap digunakan. Setiap tahapan disusun secara sistematis agar pembaca dapat mengikuti proses instalasi dengan mudah sebelum melanjutkan ke proses konfigurasi dan pengembangan aplikasi menggunakan CodeIgniter 4 pada bab selanjutnya.

1.1 INSTAL WEB SERVER XAMPP

Gambar 1.1 Logo XAMPP

Sebelum mulai menulis kode aplikasi web dengan menggunakan framework CodeIgniter 4, ada beberapa langkah persiapan yang perlu dilakukan terlebih dahulu. Salah satunya adalah menyiapkan web server yang akan menjadi tempat meng-host aplikasi web yang akan dikembangkan. Salah satu pilihan terbaik untuk pengembangan aplikasi web secara lokal adalah menggunakan XAMPP, sebuah paket perangkat lunak yang menyediakan Apache, MySQL (MariaDB), dan PHP dalam satu instalasi.

XAMPP adalah stack perangkat lunak yang mudah diinstal dan digunakan untuk pengembangan aplikasi web berbasis PHP. XAMPP memiliki keunggulan utama, yaitu kemudahan instalasi dan konfigurasi, menjadikannya pilihan yang ideal untuk pengembangan aplikasi web di lingkungan lokal (localhost). Selain itu, XAMPP kompatibel dengan berbagai sistem operasi, seperti Windows, macOS, dan Linux.

Pada bagian ini, kita akan memandu Anda untuk menginstal XAMPP di komputer Anda. Proses instalasi ini akan mencakup langkah-langkah untuk mengunduh dan mengonfigurasi XAMPP, serta menjalankan layanan yang dibutuhkan seperti Apache dan MySQL. Setelah XAMPP terinstal dan berjalan dengan baik, Anda akan siap untuk mengembangkan aplikasi web menggunakan CodeIgniter 4.

Mengapa XAMPP?

XAMPP menjadi pilihan populer di kalangan pengembang web karena beberapa alasan:

1) Instalasi Mudah: Anda tidak perlu mengonfigurasi setiap komponen secara manual. XAMPP mengatur Apache, PHP, dan MySQL agar siap digunakan langsung.
2) Gratis dan Open Source: XAMPP adalah perangkat lunak sumber terbuka yang dapat diunduh dan digunakan tanpa biaya.

3) Kompatibilitas: XAMPP bekerja di berbagai sistem operasi (Windows, macOS, dan Linux).
4) Kebutuhan Minimal: XAMPP sudah menyediakan hampir semua komponen yang diperlukan untuk mengembangkan aplikasi PHP, termasuk phpMyAdmin untuk mengelola database.

Dengan XAMPP yang sudah terpasang di sistem Anda, kita akan melanjutkan ke tahap berikutnya, yaitu menginstal CodeIgniter 4 dan mengonfigurasi proyek pengembangan web Anda. Sebelum itu, mari kita mulai dengan mempersiapkan lingkungan pengembangan lokal yang stabil dan siap pakai.

Mengunduh XAMPP

Langkah pertama adalah mengunduh file instalasi XAMPP dari situs resmi.

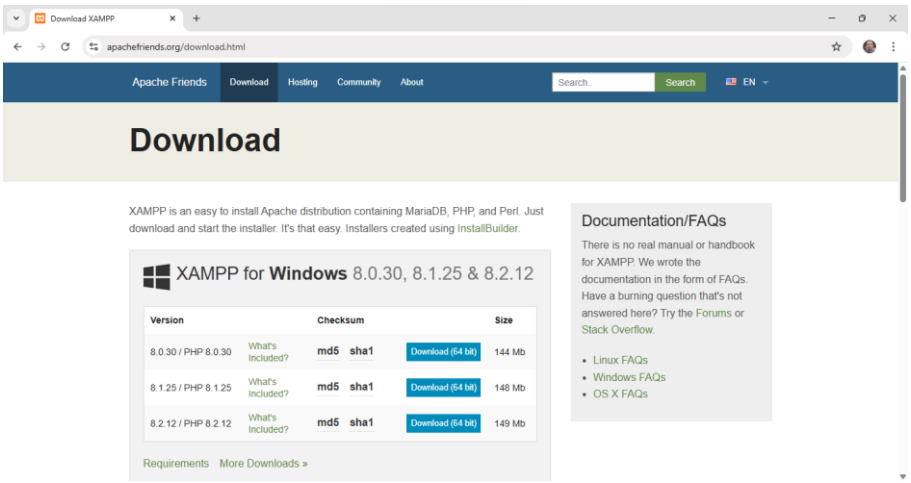
1) Kunjungi situs resmi XAMPP
   Buka browser Anda dan akses situs resmi XAMPP di *https://www.apachefriends.org/download.html*
2) Pilih versi XAMPP yang sesuai dengan sistem operasi Anda

Di halaman utama, Anda akan melihat pilihan untuk mengunduh XAMPP untuk berbagai sistem operasi (Windows, Linux, macOS). Klik pada tombol yang sesuai dengan sistem operasi yang Anda gunakan.

a) Untuk Windows, pilih "XAMPP for Windows".
b) Untuk macOS, pilih "XAMPP for OS X".
c) Untuk Linux, pilih "XAMPP for Linux".

Untuk menjalankan Code Igniter 4 direkomendasikan menggunakan versi PHP 8.x

1) Klik tombol “Download”
   Setelah memilih sistem operasi yang tepat, klik tombol "Download" untuk memulai pengunduhan file instalasi.



Gambar 1.2 Download XAMPP

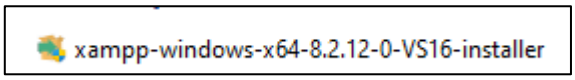
2) Tunggu hingga proses pengunduhan selesai

File instalasi akan diunduh dalam bentuk file executable (.exe untuk Windows, .dmg untuk macOS, dan tar.gz untuk Linux). Pastikan Anda menyimpan file ini di lokasi yang mudah diakses, seperti di desktop atau folder unduhan.

Instalasi XAMPP untuk Windows

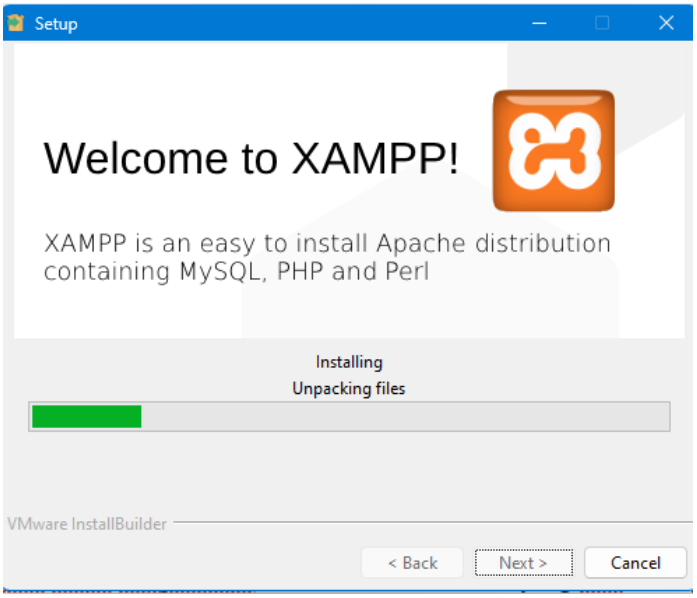
Setelah file instalasi berhasil diunduh, ikuti langkah-langkah berikut untuk menginstal XAMPP.

1) Jalankan file installer
2) Klik dua kali pada file unduhan XAMPP (.exe) untuk memulai proses instalasi.



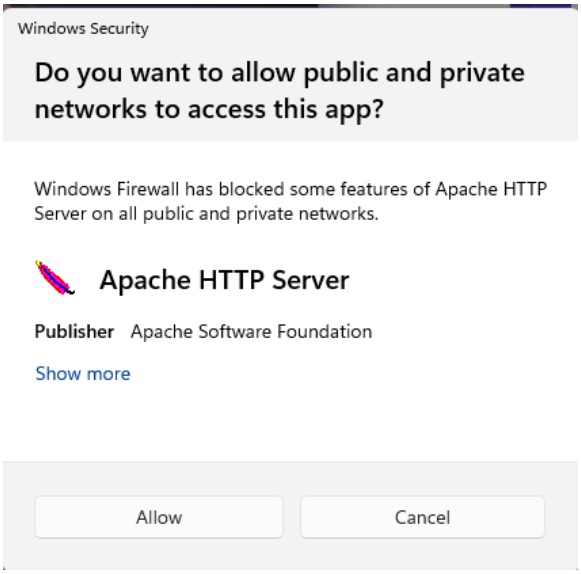
Gambar 1.3 Hasil Download XAMPP

3) Pilih bahasa instalasi (biasanya Bahasa Inggris), lalu klik OK.
4) Pilih komponen yang ingin diinstal Pada langkah ini, Anda akan diminta untuk memilih komponen yang ingin diinstal. Secara default, semua komponen penting (Apache, MySQL, PHP, dan phpMyAdmin) sudah terpilih. Anda bisa membiarkannya seperti itu dan langsung klik Next.
5) Pilih direktori instalasi atau folder tempat Anda ingin menginstal XAMPP. Biasanya, lokasi default adalah C:\xampp. Klik Next untuk melanjutkan.
6) Mulai instalasi dengan Klik Next dan kemudian Install untuk memulai proses instalasi. Tunggu hingga proses instalasi selesai. Ini mungkin memakan waktu beberapa menit tergantung pada kecepatan komputer Anda.



Gambar 1.4 Proses Instalasi XAMPP

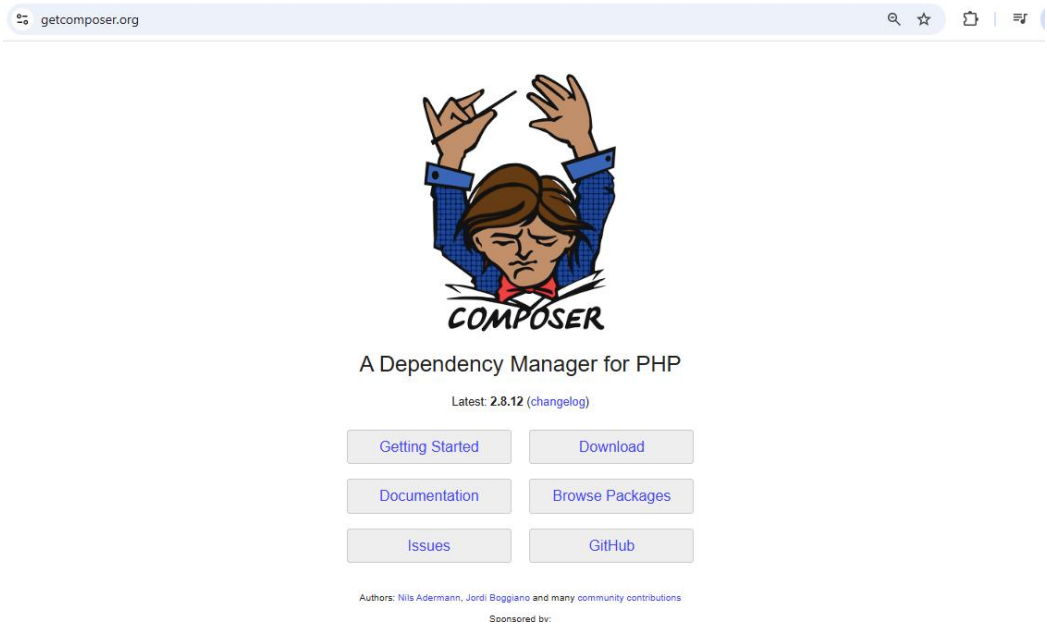
7) Jika muncul notifikasi keamanan windows seperti ini silahkan klik allow



Gambar 1.5 Notifikasi Kemanan Windows

8) Selesai. Setelah instalasi selesai, Anda akan melihat jendela yang memberi tahu bahwa instalasi telah berhasil. Klik Finish untuk menyelesaikan proses instalasi. Anda juga akan diminta untuk menjalankan XAMPP Control Panel, yang bisa Anda pilih atau lewati.

1.2 INSTAL COMPOSER



Gambar 1.6 Icon Composer

Apa itu Composer?

Composer adalah manajer dependensi untuk PHP yang digunakan untuk mengelola pustaka (*library*) dan paket-paket PHP yang digunakan dalam proyek pengembangan web.

Dengan Composer, Anda dapat dengan mudah mengelola dan mengunduh pustaka yang diperlukan oleh aplikasi PHP Anda, termasuk framework seperti CodeIgniter 4. Composer juga menangani autoloading kelas-kelas PHP dan mengelola versi paket, memastikan bahwa aplikasi Anda menggunakan versi yang kompatibel dengan dependensinya.

Secara umum, Composer memiliki dua fungsi utama:

1) Mengelola dependensi: Composer memungkinkan Anda untuk menginstal dan memperbarui pustaka pihak ketiga yang dibutuhkan aplikasi Anda secara otomatis. Anda cukup mendefinisikan pustaka yang ingin digunakan di dalam file *composer.json*, dan Composer akan menangani proses pengunduhan dan pembaruan versi pustaka tersebut.
2) Autoloading: Composer menghasilkan file autoload yang memungkinkan aplikasi Anda untuk memuat kelas PHP secara otomatis tanpa perlu menulis kode require atau include secara manual.

Dengan Composer, Anda bisa mengelola berbagai pustaka dan framework, termasuk CodeIgniter 4, dengan lebih mudah, efisien, dan terstruktur.

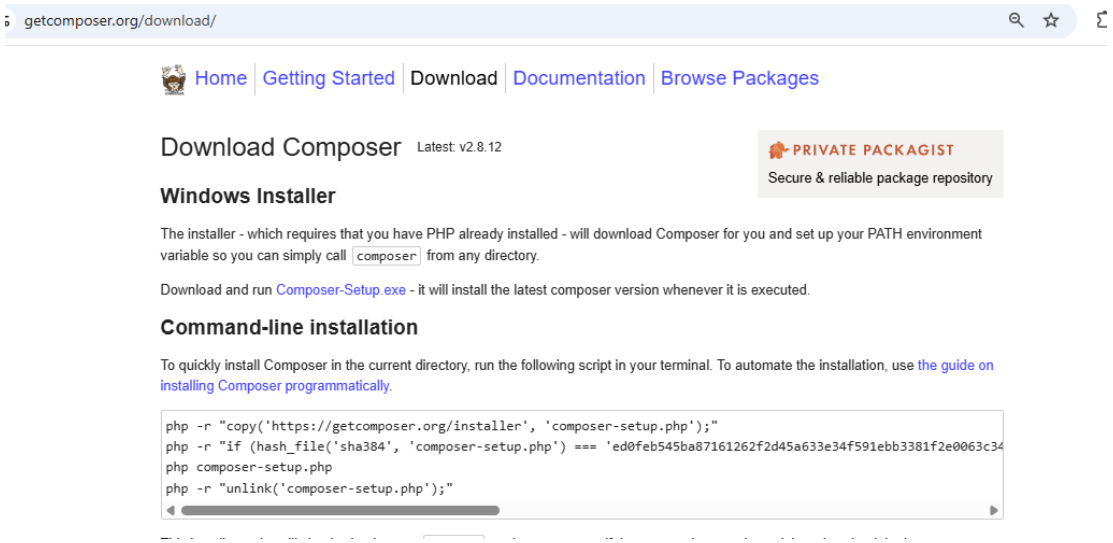
Mengapa Menggunakan Composer?

1) Manajemen Dependensi yang Mudah: Composer memungkinkan Anda untuk mengelola berbagai pustaka PHP dengan cara yang lebih terstruktur dan mudah dipahami. Anda dapat menentukan versi pustaka yang tepat dan Composer akan memastikan bahwa pustaka-pustaka yang terpasang kompatibel satu sama lain.
2) Autoloading Kelas: Composer dapat secara otomatis memuat kelas yang Anda butuhkan dalam proyek tanpa perlu menulis autoloading secara manual.
3) Mempermudah Kolaborasi: Dengan menggunakan Composer, Anda dan tim pengembang lain dapat bekerja pada proyek yang sama tanpa khawatir tentang masalah ketidakcocokan versi pustaka atau dependensi yang hilang.

Langkah-langkah Menginstal Composer

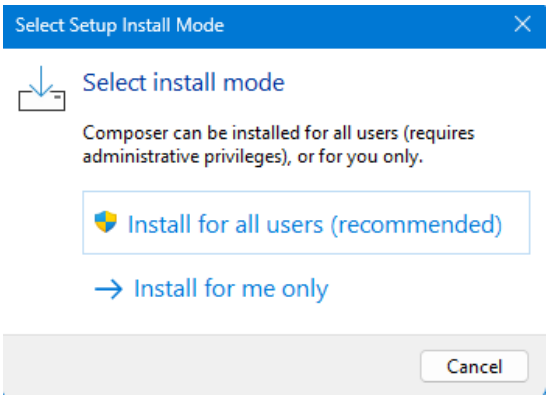
Berikut adalah langkah-langkah untuk menginstal Composer pada sistem operasi Windows:

1) Unduh Installer Composer untuk Windows
2) Kunjungi situs resmi Composer di *https://getcomposer.org*
3) Klik pada tombol "Download" untuk Windows. Pilih dan unduh Composer-Setup.exe, yaitu installer yang sudah terkonfigurasi untuk Windows.



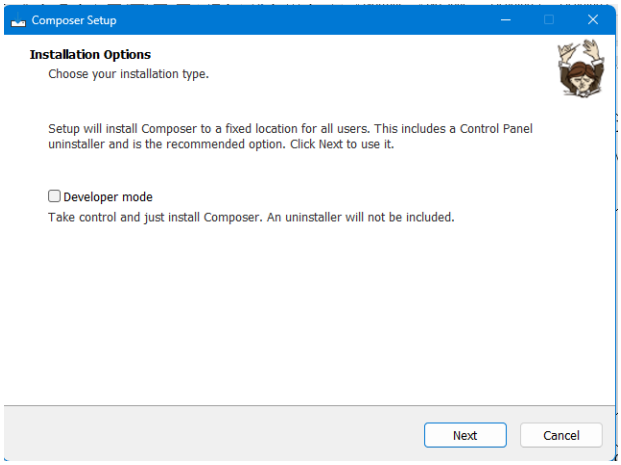
Gambar 1.7 Download Composer

4) Jalankan Installer. Setelah unduhan selesai, klik dua kali file Composer-Setup.exe untuk memulai proses instalasi.
5) Ikuti petunjuk di layar: Pilih PHP yang digunakan. Jika Anda telah menginstal XAMPP, maka installer akan otomatis mendeteksi PHP yang terpasang di C:\xampp\php\php.exe. Pastikan PHP terdeteksi dengan benar.



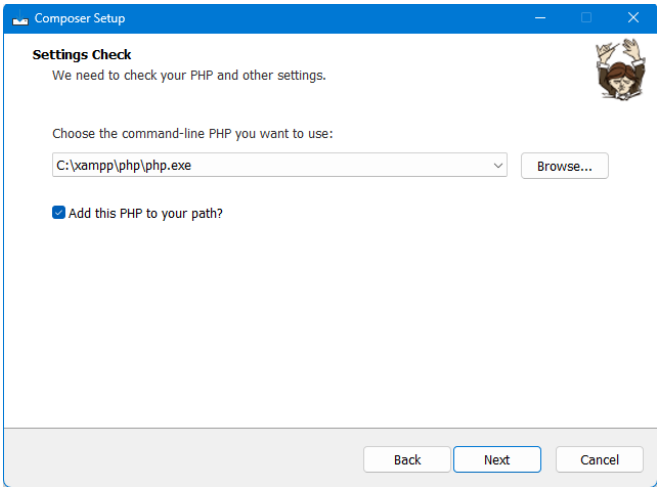
Gambar 1.8 Setup Instalasi

6) Akan tampil halaman seperti dibawah dan Pilih next



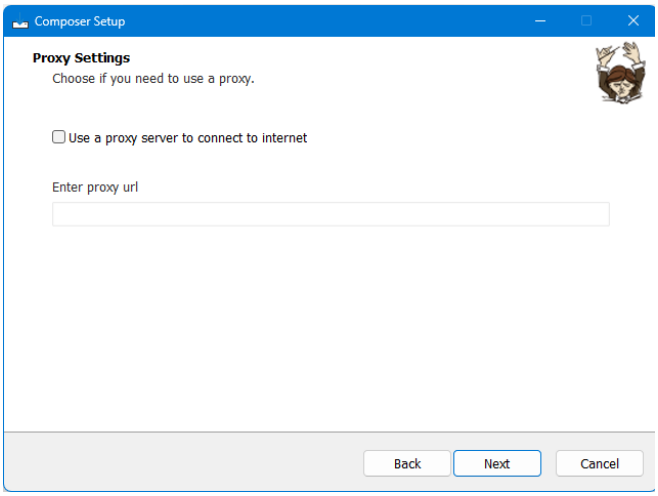
Gambar 1.9

7) Pastikan anda telah selesai menginstal XAMPP dan dapat dijalankan.



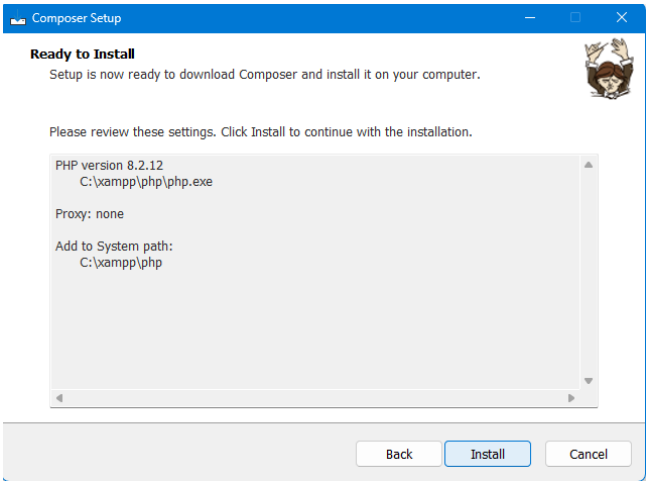
Gambar 1.10 Composer Setup

8) Pada bagian ini di next saja

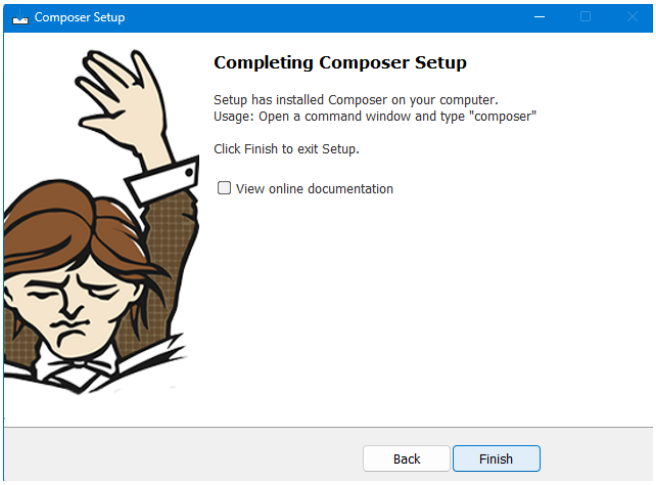


Gambar 1.11 Proxi Setting Setup Composer

9) Pada tahapan akhir klik instal – next dan finish



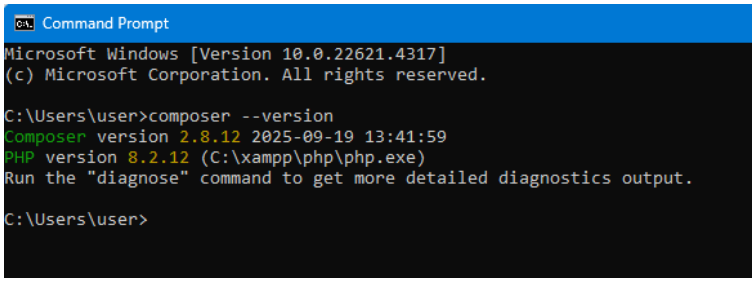
Gambar 1.12 Composer Setup Ready to Instal



Gambar 1.13 Selesai Penginstalan

10) Verifikasi Instalasi
11) Buka Command Prompt (CMD) atau PowerShell dan ketik perintah berikut:
12) composer --version

Jika Composer terinstal dengan benar, Anda akan melihat versi Composer yang terpasang, seperti:



Gambar 1.14 CMD untuk Memeriksa Composer telah Berhasil di Instal

1.3 INSTAL TEKS EDITOR (VS CODE)

Gambar 1.15 Icon VS Code

Apa itu Visual Studio Code?

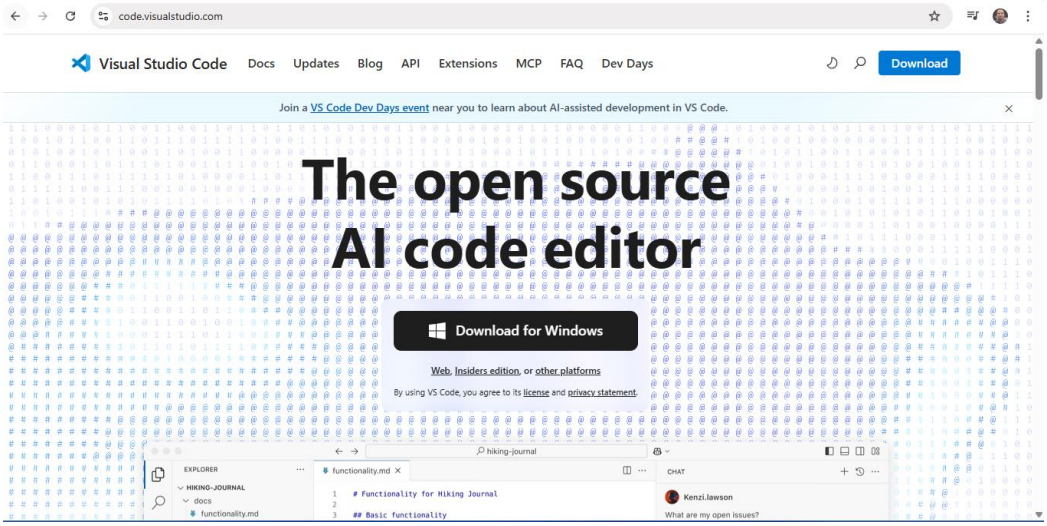
Visual Studio Code (sering disingkat VS Code) adalah teks editor gratis dan ringan yang dikembangkan oleh Microsoft. VS Code sangat populer di kalangan pengembang web dan aplikasi karena memiliki banyak fitur yang mendukung pemrograman modern, seperti:

1) Highlighting sintaksis untuk berbagai bahasa pemrograman.
2) Auto-completion dan IntelliSense untuk membantu menulis kode lebih cepat dan benar.
3) Terminal bawaan untuk menjalankan perintah langsung dari dalam editor.
4) Ekstensi (extensions) yang sangat banyak, misalnya untuk PHP, CodeIgniter, Git, Laravel, dsb.
5) Integrasi Git, debugger, dan manajemen proyek yang sangat baik.

VS Code adalah alat yang sangat direkomendasikan untuk menulis kode PHP dan mengembangkan proyek dengan CodeIgniter 4.

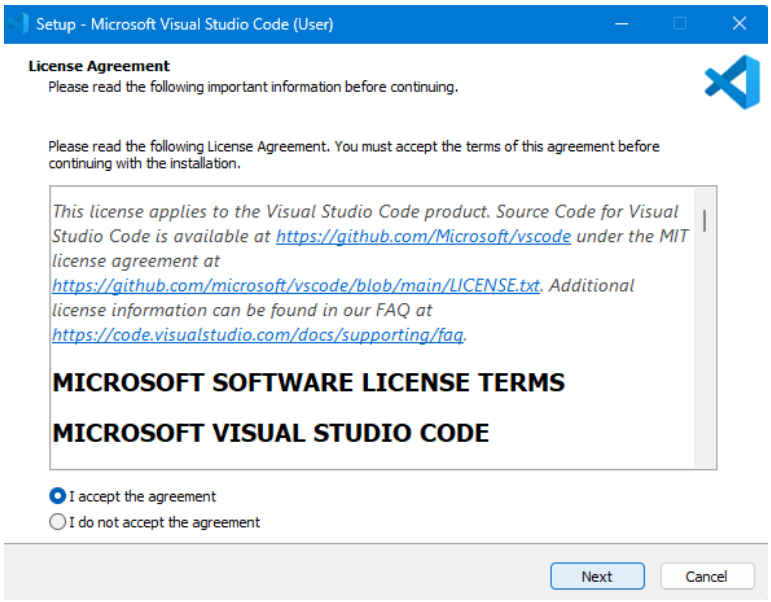
Langkah-langkah Menginstal VS Code

1) Unduh Visual Studio Code, Buka browser dan akses situs resmi VS Code: *https://code.visualstudio.com/*
2) Di halaman utama, klik tombol "*Download for Windows*" (atau sistem operasi Anda).



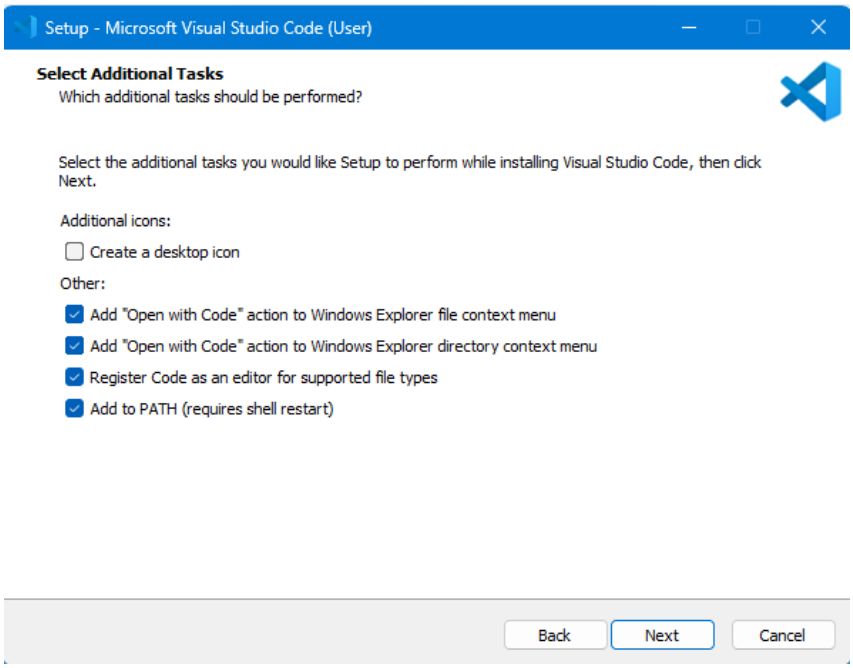
Gambar 1.16 Download VS Code

3) VS Code akan otomatis mengunduh installer yang sesuai untuk sistem operasi Anda
4) Jalankan file installer (VSCodeUserSetup-x64-*.exe).
5) Klik Next di layar pembuka.
6) Baca dan setujui perjanjian lisensi, lalu klik Next.



Gambar 1.17 Perjanjian Instalasi VS Code

7) Pilih folder instalasi (biarkan default: C:\Program Files\Microsoft VS Code), klik Next.
8) Pada bagian “Select Additional Tasks”, centang opsi berikut:
   - Add to PATH (agar bisa dipanggil lewat Command Prompt)
   - Register Code as an editor for supported file types
   - Add “Open with Code” action to Windows Explorer



Gambar 1.18 Setup Instalasi VS Code

9) Klik Next, lalu klik Install.
10) Tunggu hingga proses instalasi selesai, kemudian klik Finish. Setelah instalasi selesai, VS Code akan terbuka secara otomatis (jika opsi "Launch" dicentang).

Gambar 1.19 Selesai Penginstalan

11) Langkah Setelah Instalasi: Konfigurasi Awal VS Code
    a. Buka VS Code
    b. Install Ekstensi Penting untuk Pengembangan Web & PHP. Klik pada ikon **Extensions** (ikon kotak di sidebar kiri), lalu cari dan instal ekstensi berikut:

Tabel 1.1 Fungsi Ekstensi dalam VS Code

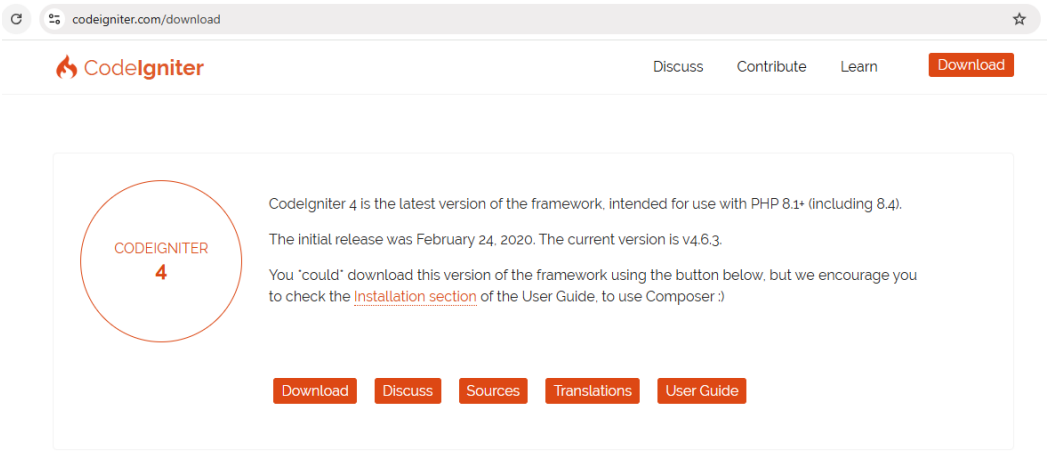
Ekstensi	Fungsi
PHP Intelephense	Autocomplete dan intelligence untuk PHP
CodeIgniter 4 Snippets	Snippet dan bantuan sintaks untuk CodeIgniter 4
PHP Debug	Debugging aplikasi PHP
Live Server	Menyediakan server 13ocal untuk preview HTML
Prettier	Untuk memformat kode secara otomatis
GitLens	Integrasi Git yang canggih

c. Setelah instalasi, restart VS Code agar semua ekstensi aktif dengan baik.

1.4 INSTAL CODE IGNITER 4

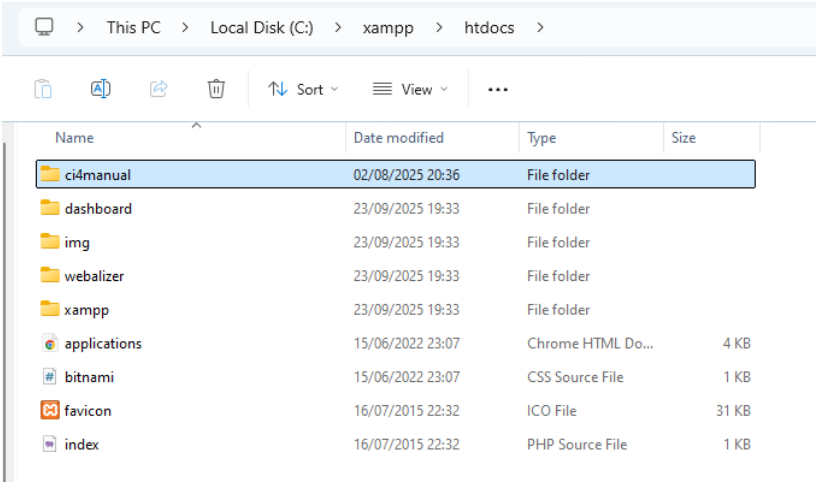
Instal Manual Download

1) silahkan buka *https://codeigniter.com/download* dan download codeigniter 4 nya



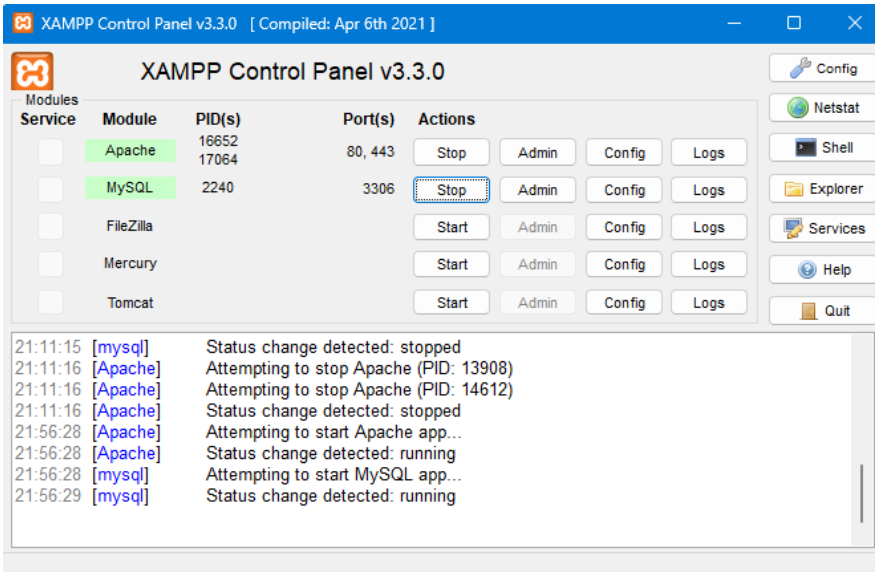
Gambar 1.20 Download Code Igniter 4

2) Extrack dan Ganti nama folder sesuai keinginan kita dan simpan di Lokasi penginstalan web server XAMPP nya (htdocs). Disini nama folder diubah menjadi **ci4manual**



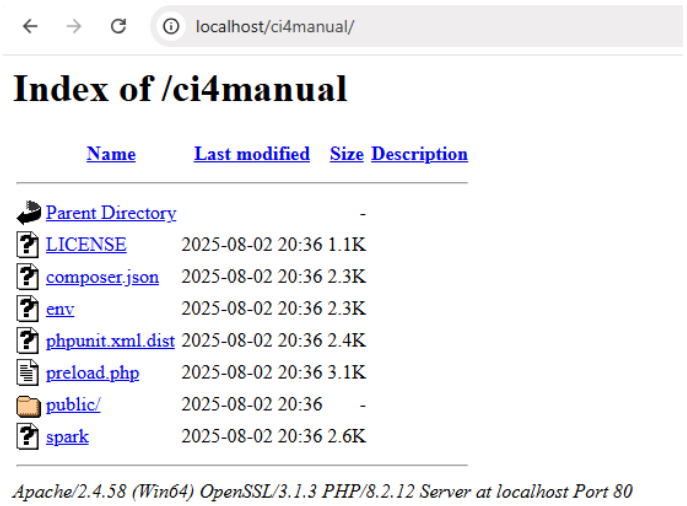
Gambar 1.21 Ubah Nama Folder Hasil Download

3) Jalankan webserver XAMPP untuk apache



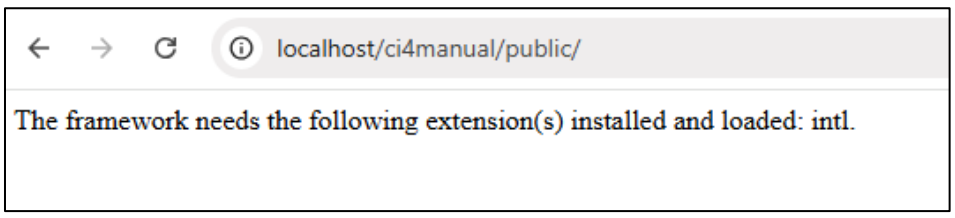
Gambar 1.22 Jalankan Xampp

4) Buka aplikasi browser dan tulis localhost/ci4manual pilih folder public



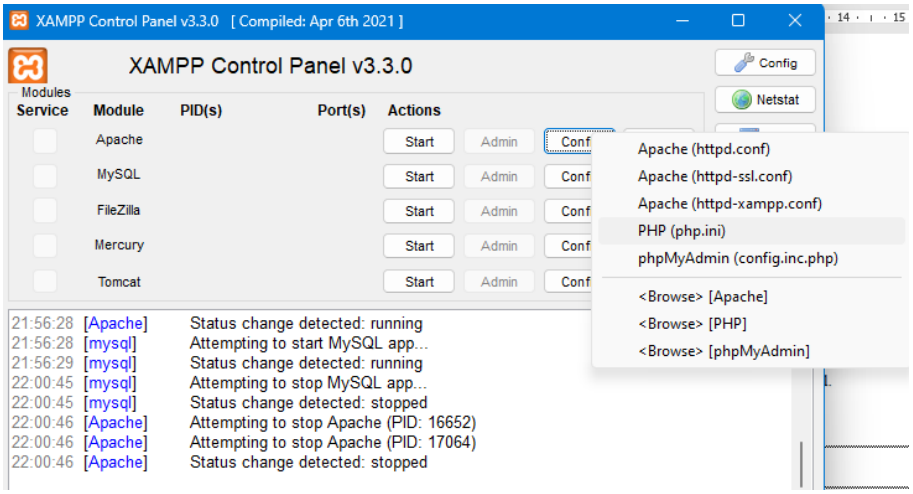
Gambar 1.23 Tampilan Awal Codeigniter di Web Browser

5) Akan tampil eror seperti berikut



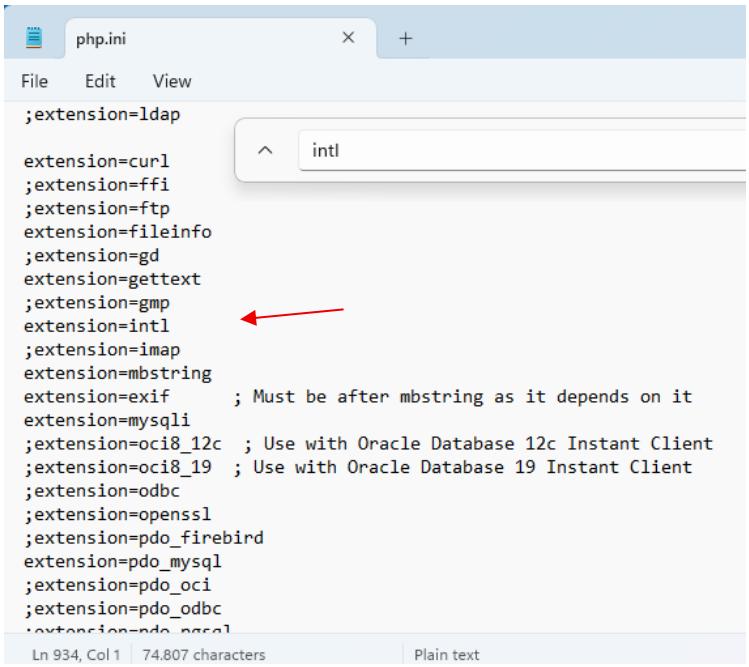
Gambar 1.24 Notifikasi Eror pada Tampilan

6) Stop apache pada XAMPP dan klik pada config apache



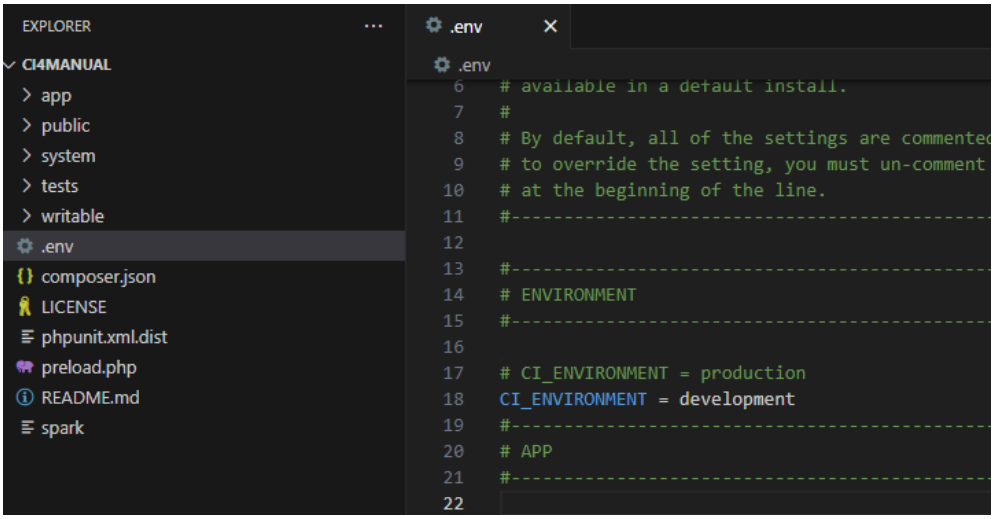
Gambar 1.25 Tampilan Konfigurasi XAMPP

7) Pilih PHP (php.ini) lalu cari extensi intl, hapus tanda ";" pada bagial awal extension=intl lalu klik simpan dan close php.ini



Gambar 1.26 Lokasi Extention Int

8) Buka projek ci4manual kita dari teks editor vs code, cari file Bernama **env** kemudian rename menjadi**.env** (diawalai dengan titik). selanjutnya tambahkan perintah **CI_ENVIRONMENT = development**



Gambar 1.27 Isi File.env

Keterangan:

Apa itu file env?

a) Itu adalah **template file environment** bawaan CI4.
b) Isinya konfigurasi dasar untuk environment (pengaturan server, mode development/production, database, dll).
c) CI4 tidak langsung memakainya, tapi menunggu Anda mengubahnya jadi.env (dengan titik di depan).

Mengapa harus diganti jadi.env?

a) File dengan nama.env adalah standar untuk Environment Configuration di banyak framework (termasuk Laravel, CI4, dsb).
b) Setelah diganti jadi.env, CI4 akan membaca isi file ini dan menyesuaikan konfigurasi project.
c) Kalau tidak diubah, CI4 akan pakai default setting yang ada di file /app/Config/Boot/production.php atau /app/Config/Boot/development.php.

Bagian CI_ENVIRONMENT

Di dalam.env ada baris seperti ini:

```
# CI_ENVIRONMENT = production
```

Tanda **#** artinya komentar → tidak dijalankan.

Kalau diubah menjadi:

```
CI_ENVIRONMENT = development
```

Maka CodeIgniter akan berjalan di **mode development**, artinya:

a) Semua error akan ditampilkan di browser → memudahkan debugging.
b) Cocok saat proses ngoding.

Kalau pakai:

CI_ENVIRONMENT = production

Maka CodeIgniter jalan di **mode production**, artinya:

a) Error tidak ditampilkan ke user (lebih aman).
b) Cocok untuk server/website yang sudah online.

Jadi singkatnya:

a) Ubah nama file env →.env.
b) Aktifkan pengaturan environment dengan set:

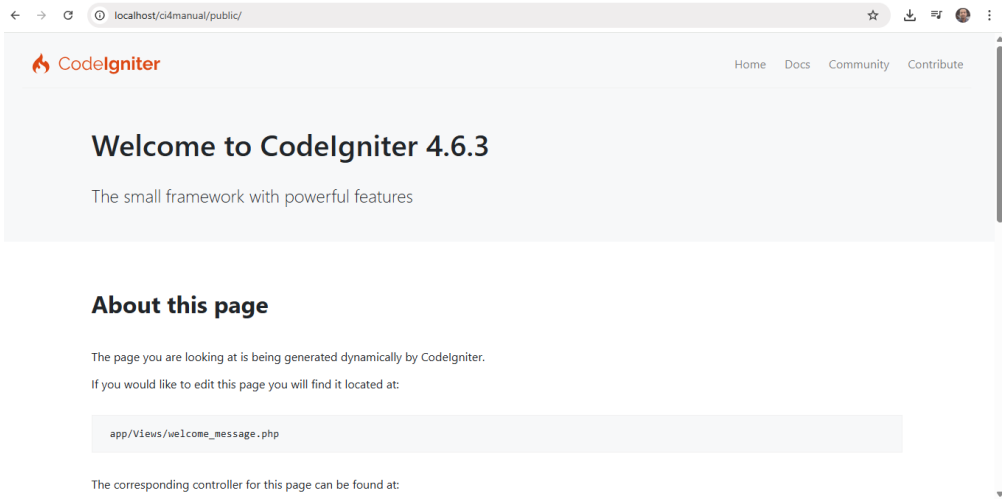
CI_ENVIRONMENT = development

(supaya error dan debugging muncul saat belajar/ngoding).

a) Kalau nanti sudah upload ke hosting/production server → ubah jadi:

CI_ENVIRONMENT = production

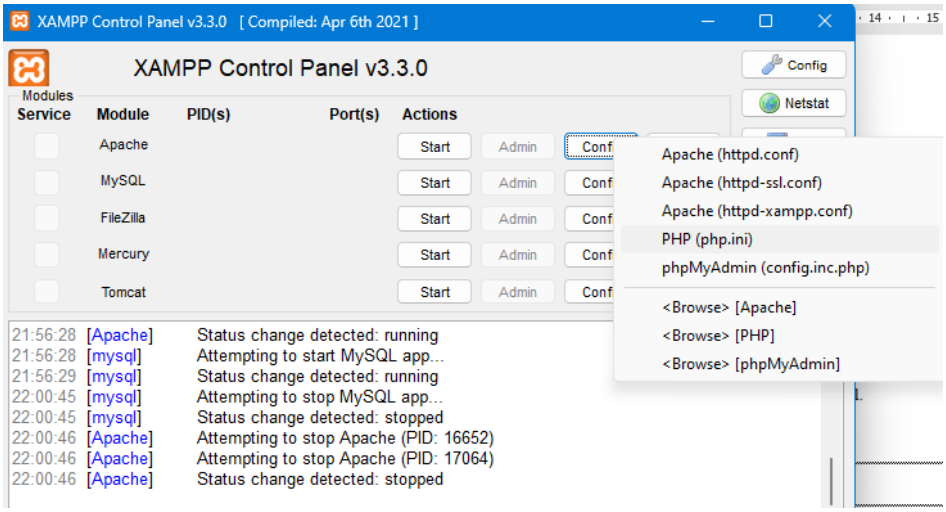
9) Start ulang apache pada xampp dan refresh halaman di browser



Gambar 1.28 Tampilan Awal CodeIgniter

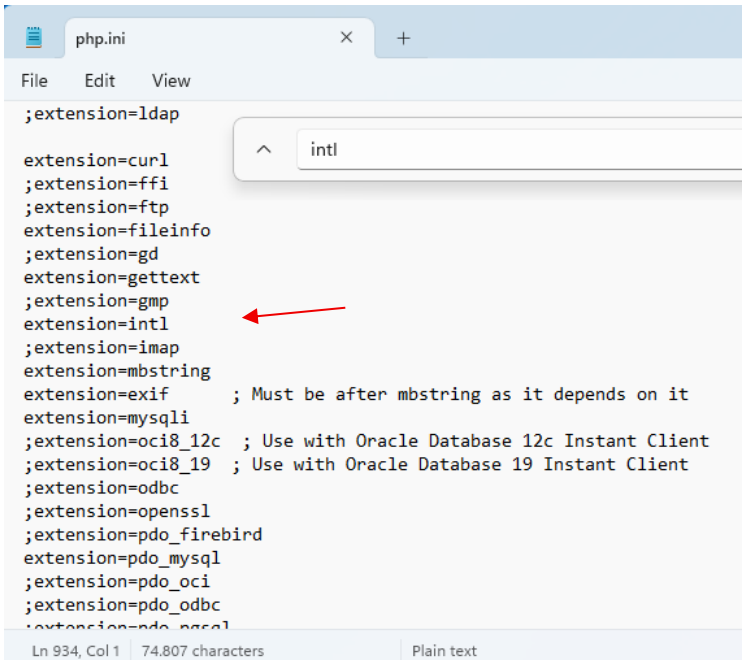
Instal dari Composer

1) Pastikan extension itl sudah di aktifkan pada config php.ini di apcahe XAMPP. Pada bagian apache di XAMPP dan klik pada config apachenya



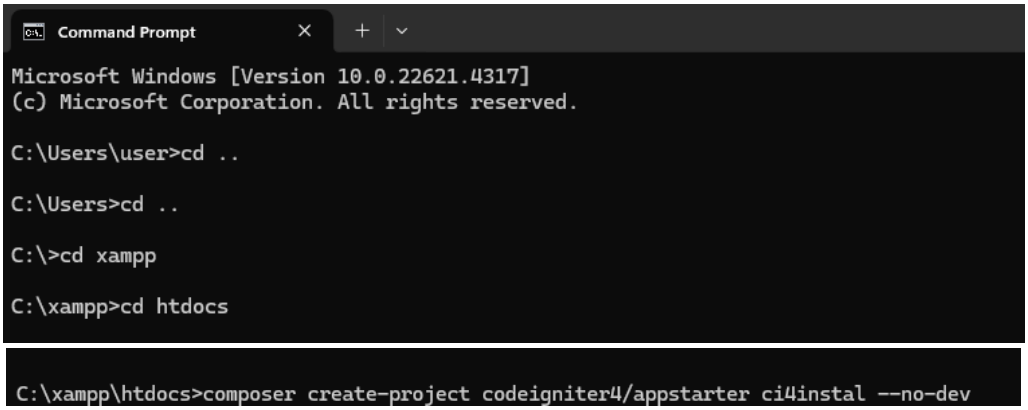
Gambar 1.29 Konfigurasi Extension Pada XAMPP

2) Pilih PHP (php.ini) lalu cari extensi intl, hapus tanda ";" pada bagial awal extension=intl lalu klik simpan dan close php.ini



Gambar 1.30 Lokasi Ekstension intl

3) Simpan configurasi ini dan close config php.ini
4) Buka command prompt (cmd). Arahkan menuju directori tempat penginstalan XAMPP dan tuliskan perintah seperti berikut

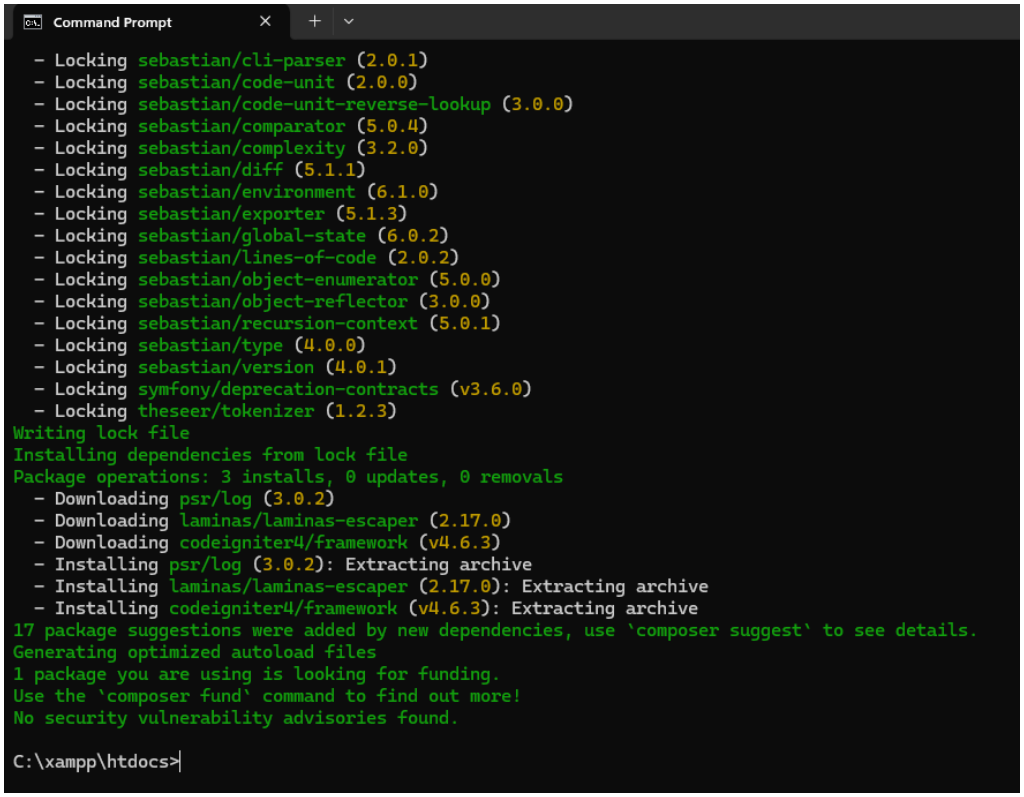


Gambar 1.31 Tampilan Command Promt

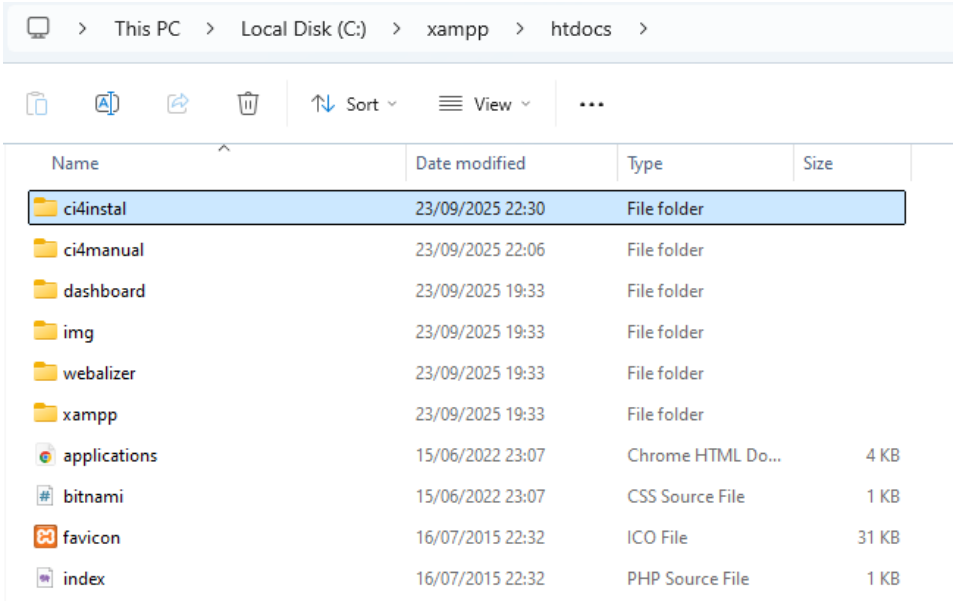
Keterangan:

a) cd .. (untuk Kembali ke directori sebelumnya)
b) cd xampp (untuk masuk ke directori XAMPP)
c) composer create-project codeigniter4/appstarter ci4instal --no-dev (perintah instal CI 4)

5) akan melakuakn instalasi dan tampil seperti berikut

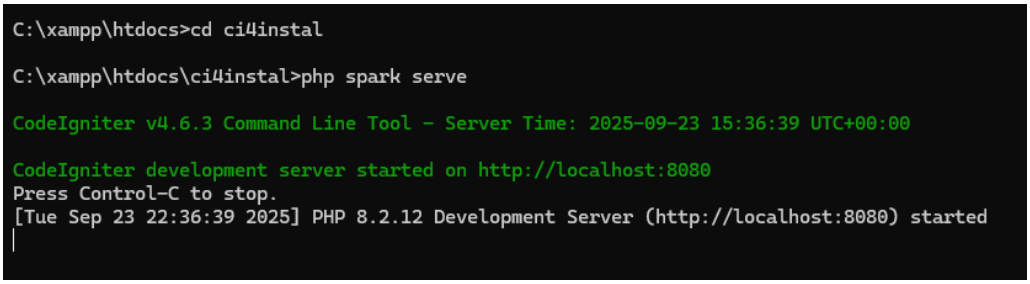


Gambar 1.32 Proses Instalasi Composer di CMD



Gambar 1.33 Hasil Instalasi pada Explore

6) masuk kedalam directori instalasi codeigniter nya dengan perintah cd ci4instal dan lakukan perintah php spark serve



Gambar 1.34 Perintah PHP Spark Serve pada CMD

Note: php spark serve adalah perintah bawaan CodeIgniter 4 untuk menjalankan development server (server bawaan PHP) langsung dari terminal/command prompt.

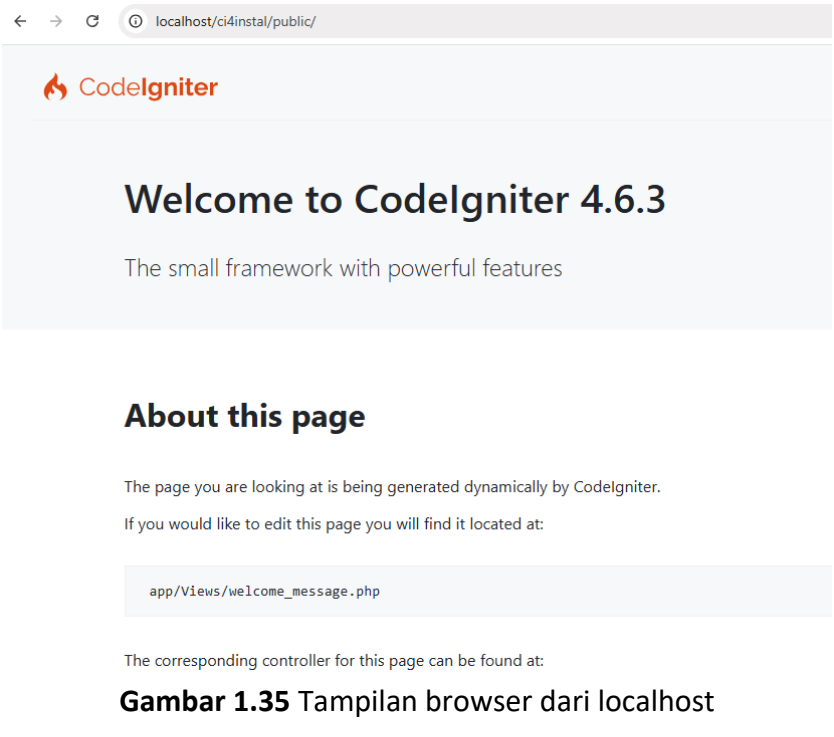
Fungsinya

a) Menjalankan aplikasi CI4 tanpa harus lewat Apache/Nginx.
b) Membuka server lokal di alamat default: *http://localhost:8080*

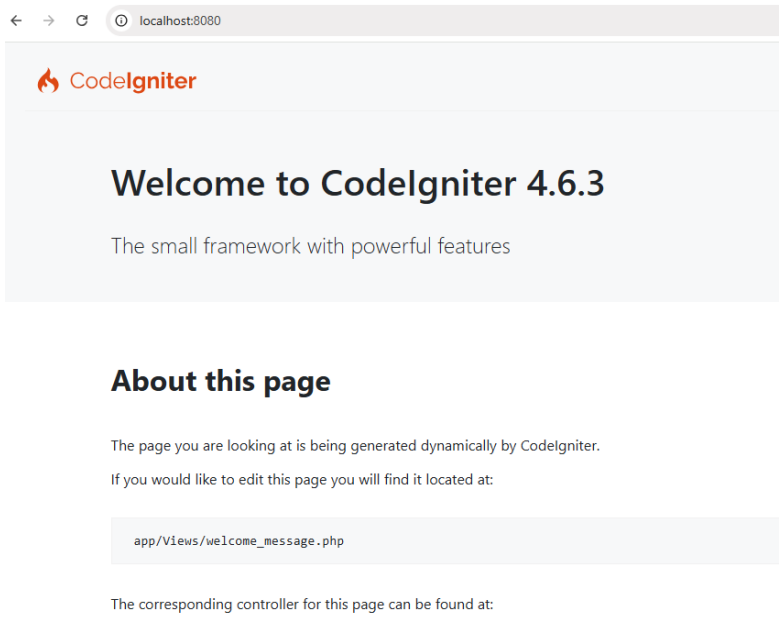
Apa itu Nginx?

a) Nginx (dibaca: *engine-x*) adalah web server modern, mirip seperti Apache (yang ada di XAMPP).
b) Digunakan untuk melayani website/aplikasi agar bisa diakses lewat browser.
c) Selain web server, Nginx juga bisa jadi reverse proxy, load balancer, dan caching server.

7) Silahkan buka hasil instalan ini menggunakan web browser dengan menuliskan *http://localhost/ci4instal* atau *http://localhost:8080*



Gambar 1.35 Tampilan browser dari localhost



Gambar 1.36 Tampilan Browser dari Port 8080

TUGAS BAB 1

1. Install PHP dan Composer di komputer Anda.
2. Buat project baru CodeIgniter 4 dengan nama bebas
3. Jalankan server dengan perintah: php spark serve
4. Akses di browser *http://localhost:8080* dan pastikan halaman default CodeIgniter tampil.
5. Buat screenshot hasil instalasi CodeIgniter 4 di komputer Anda (halaman welcome page CI4).

6. Pahami dan pelajari struktur folder yang ada di CodeIgniter 4
7. Tuliskan fungsi dari masing – masing folder yang paling sering digunakan!

Rangkuman Bab

Pada bab ini telah dibahas tahapan awal yang harus dipersiapkan sebelum membangun aplikasi web menggunakan CodeIgniter 4. Persiapan lingkungan pengembangan merupakan langkah yang sangat penting karena seluruh proses implementasi aplikasi pada bab berikutnya bergantung pada keberhasilan tahap ini. Lingkungan pengembangan yang telah dikonfigurasi dengan baik akan membantu proses penulisan, pengujian, dan pemeliharaan aplikasi menjadi lebih mudah dan terstruktur.

Pembaca telah mempelajari proses instalasi XAMPP sebagai web server lokal, Composer sebagai pengelola dependensi PHP, Visual Studio Code sebagai editor kode program, serta CodeIgniter 4 sebagai framework utama yang digunakan dalam praktikum. Selain itu, pembaca juga telah memahami struktur awal proyek CodeIgniter 4 sehingga dapat mengenali lokasi penyimpanan controller, model, view, konfigurasi aplikasi, maupun direktori publik yang akan digunakan selama proses pengembangan aplikasi.

Keberhasilan menyelesaikan seluruh tahapan pada bab ini menjadi dasar yang sangat penting untuk melanjutkan pembelajaran pada bab berikutnya. Seluruh perangkat lunak yang telah diinstal dan dikonfigurasi akan digunakan kembali pada setiap kegiatan praktikum sehingga pembaca tidak perlu melakukan proses instalasi ulang selama struktur proyek masih tersedia.

Sebelum melanjutkan ke Bab 2, pastikan seluruh perangkat lunak telah berhasil dijalankan tanpa kendala. Apache dan MySQL pada XAMPP harus berada dalam kondisi aktif, Composer dapat diakses melalui terminal, Visual Studio Code dapat digunakan untuk membuka proyek CodeIgniter 4, serta aplikasi berhasil dijalankan melalui browser menggunakan alamat *localhost*. Dengan memastikan seluruh komponen tersebut telah berfungsi dengan baik, proses pembelajaran pada bab berikutnya akan berlangsung lebih lancar.

Bab berikutnya akan membahas konfigurasi awal CodeIgniter 4 serta pengenalan konsep *Model–View–Controller* (MVC). Pada tahap tersebut pembaca akan mulai memahami bagaimana sebuah aplikasi dibangun menggunakan struktur framework sehingga setiap komponen memiliki tugas dan tanggung jawab yang berbeda. Pemahaman terhadap konsep MVC menjadi fondasi utama sebelum melakukan implementasi fitur-fitur CRUD pada bab-bab selanjutnya.

Refleksi Pembelajaran

Persiapan lingkungan pengembangan merupakan langkah awal yang sering dianggap sederhana, namun memiliki pengaruh yang sangat besar terhadap keberhasilan proses pengembangan aplikasi. Seluruh tahapan instalasi dan konfigurasi yang telah dipelajari pada bab ini bertujuan untuk membangun lingkungan kerja yang stabil sehingga setiap proses praktikum pada bab-bab berikutnya dapat berjalan dengan baik. Oleh karena itu, pembaca tidak hanya diharapkan mampu mengikuti setiap langkah instalasi, tetapi juga memahami fungsi dari setiap perangkat lunak yang digunakan dalam proses pengembangan aplikasi.

Setelah menyelesaikan bab ini, pembaca sebaiknya mencoba melakukan instalasi dan konfigurasi kembali secara mandiri tanpa melihat panduan secara langsung. Kegiatan tersebut akan membantu meningkatkan pemahaman mengenai fungsi masing-masing perangkat serta membiasakan pembaca dalam menyelesaikan permasalahan teknis yang mungkin muncul ketika melakukan instalasi pada komputer yang berbeda. Kemampuan tersebut menjadi bekal yang penting karena dalam praktik pengembangan perangkat lunak, seorang pengembang sering dihadapkan pada proses instalasi, konfigurasi, maupun penyesuaian lingkungan kerja sesuai kebutuhan proyek.

Penerapan dalam Dunia Nyata

Lingkungan pengembangan yang dipelajari pada bab ini merupakan standar yang banyak digunakan dalam pengembangan aplikasi web di lingkungan akademik maupun industri. Berbagai perusahaan pengembang perangkat lunak memanfaatkan web server lokal, editor kode, serta framework pengembangan untuk mempercepat proses pembangunan aplikasi sebelum dipublikasikan ke server produksi. Dengan demikian, kemampuan melakukan instalasi dan konfigurasi lingkungan pengembangan bukan hanya dibutuhkan pada saat perkuliahan, tetapi juga menjadi salah satu kompetensi dasar yang diperlukan ketika bekerja sebagai web developer atau software engineer.

Pemahaman mengenai proses instalasi yang benar juga akan membantu pembaca ketika bekerja secara kolaboratif dalam sebuah tim pengembang. Setiap anggota tim harus menggunakan lingkungan kerja yang seragam agar aplikasi dapat dijalankan dengan konfigurasi yang sama pada setiap komputer. Hal tersebut akan meminimalkan terjadinya kesalahan akibat perbedaan konfigurasi sistem dan mempermudah proses pengembangan maupun pengujian aplikasi.

Tips Pengembangan Aplikasi

Biasakan untuk menyimpan seluruh proyek pada direktori yang mudah diakses dan menggunakan struktur folder yang rapi sejak awal pengembangan. Selain itu, lakukan pencadangan (*backup*) proyek secara berkala agar data tetap aman apabila terjadi kerusakan sistem atau kesalahan saat melakukan pengembangan. Pastikan pula seluruh perangkat lunak seperti XAMPP, Composer, Visual Studio Code, dan CodeIgniter 4 menggunakan versi yang saling kompatibel sehingga proses pengembangan dapat berjalan dengan lebih stabil dan mengurangi potensi munculnya kesalahan yang tidak diinginkan.

Persiapan Menuju Bab Berikutnya

Setelah seluruh proses instalasi dan konfigurasi berhasil diselesaikan, pastikan lingkungan pengembangan telah siap digunakan dengan menjalankan aplikasi CodeIgniter 4 melalui browser tanpa mengalami kendala. Keberhasilan pada tahap ini menunjukkan bahwa seluruh komponen telah terpasang dengan benar dan siap digunakan pada proses praktikum selanjutnya. Pada bab berikutnya, pembahasan akan difokuskan pada konfigurasi awal aplikasi serta pengenalan konsep *Model–View–Controller* (MVC) sebagai dasar dalam membangun aplikasi web yang terstruktur menggunakan framework CodeIgniter 4. Oleh karena itu, pastikan seluruh tahapan pada bab ini telah dipahami dengan baik sebelum melanjutkan ke materi berikutnya.

BAB 2
CONFIGURASI AWAL DAN KONSEP MVC (MODEL VIEW CONTROLLER)

Setelah lingkungan pengembangan berhasil dipersiapkan pada bab sebelumnya, langkah berikutnya adalah melakukan konfigurasi awal aplikasi CodeIgniter 4 serta memahami konsep dasar yang menjadi fondasi dalam pengembangan aplikasi menggunakan framework tersebut. Tahap ini merupakan bagian yang sangat penting karena menentukan bagaimana aplikasi akan disusun, dijalankan, dan dikembangkan pada proses selanjutnya.

CodeIgniter 4 dirancang sebagai framework yang menerapkan arsitektur *Model–View–Controller* (MVC). Arsitektur ini bertujuan untuk memisahkan logika aplikasi, pengelolaan data, dan tampilan antarmuka ke dalam komponen yang berbeda. Dengan adanya pemisahan tersebut, proses pengembangan menjadi lebih terstruktur, kode program lebih mudah dipahami, serta pemeliharaan aplikasi dapat dilakukan dengan lebih efektif. Pendekatan ini juga memudahkan pengembang dalam melakukan pengembangan lanjutan tanpa harus mengubah keseluruhan struktur aplikasi.

Selain memahami konsep MVC, konfigurasi awal pada CodeIgniter 4 juga menjadi tahapan yang tidak dapat diabaikan. Beberapa pengaturan dasar, seperti konfigurasi lingkungan aplikasi, pengaturan alamat dasar (*base URL*), serta struktur direktori proyek perlu dipahami agar aplikasi dapat dijalankan dengan baik. Kesalahan pada tahap konfigurasi sering kali menyebabkan aplikasi tidak dapat diakses atau menampilkan kesalahan ketika proses pengembangan dilakukan. Oleh karena itu, pemahaman terhadap konfigurasi dasar akan membantu pembaca menghindari berbagai kendala teknis pada saat praktikum.

Pada bab ini pembaca juga mulai mengenal struktur folder yang terdapat pada CodeIgniter 4. Setiap direktori memiliki fungsi yang berbeda, seperti penyimpanan Controller, Model, View, file konfigurasi, aset aplikasi, maupun direktori publik yang dapat diakses melalui browser. Pemahaman terhadap struktur proyek akan memudahkan pembaca dalam menentukan lokasi penyimpanan setiap file yang dibuat selama proses pengembangan aplikasi.

Selanjutnya, pembaca akan mempelajari fungsi masing-masing komponen dalam pola MVC. Model bertugas mengelola data dan berinteraksi dengan basis data, View bertanggung jawab menampilkan informasi kepada pengguna, sedangkan Controller menjadi penghubung yang mengatur alur komunikasi antara Model dan View. Ketiga komponen tersebut bekerja secara terpadu sehingga menghasilkan aplikasi yang lebih terorganisasi dibandingkan dengan penulisan kode PHP secara prosedural.

Penerapan konsep MVC juga memberikan berbagai keuntungan dalam pengembangan perangkat lunak, antara lain meningkatkan keterbacaan kode program, mempermudah proses pengujian, mengurangi duplikasi kode, serta memudahkan pengembangan fitur baru. Oleh

karena itu, hampir seluruh framework PHP modern menerapkan konsep serupa sebagai standar dalam pengembangan aplikasi berbasis web.

Melalui bab ini pembaca diharapkan tidak hanya mampu menjalankan aplikasi CodeIgniter 4, tetapi juga memahami alasan di balik penggunaan setiap komponen dalam framework. Pemahaman konseptual tersebut akan menjadi bekal penting ketika pembaca mulai membangun aplikasi CRUD pada bab-bab berikutnya. Dengan memahami hubungan antara Model, View, dan Controller, pembaca akan lebih mudah mengikuti setiap langkah praktikum serta mampu mengembangkan aplikasi secara mandiri sesuai kebutuhan.

2.1 PENDAHULUAN

Tujuan Pembelajaran

Setelah mengikuti menyelesaikan pembahasan pada bab ini, pembaca diharapkan mampu:

a) Memahami konsep dasar MVC (*Model – View – Controller*) dalam framework CodeIgniter 4.
b) Membuat Controller sederhana untuk menangani request dari user.
c) Membuat View untuk menampilkan data ke user.
d) Menghubungkan View**,** dan Controller ke dalam sebuah alur kerja sederhana.

Konsep Dasar MVC

a) **Model:** Bagian yang bertugas mengatur data (CRUD) dan berinteraksi dengan database.
b) **View:** Bagian yang mengatur tampilan (UI) kepada pengguna.
c) **Controller:** Bagian penghubung antara Model dan View, yang berisi logika aplikasi.

Analogi:

a) **Model** = Dapur & Bahan Masakan
b) **Controller** = Koki
c) **View** = Makanan yang disajikan di meja

2.2 ATURAN PENAMAAN FILE DAN CLASS PADA MVC DI CODEIGNITER 4

Dalam CodeIgniter 4, struktur file dan penamaan class harus mengikuti aturan tertentu agar framework bisa mengenali file dengan benar. Berikut adalah pedoman yang wajib diperhatikan:

Controller

a) Lokasi: app/Controllers/
b) Aturan Penamaan:
   1) Nama file dan nama class harus sama. Contoh: Pembaca.php berisi class Pembaca.
   2) Nama class huruf pertama kapital (PascalCase).
      Benar: Pembaca.php → class Pembaca
      Salah: pembaca.php → class pembaca
   3) Class Controller harus extends BaseController.
   4) Method di dalam controller biasanya ditulis dengan camelCase. Contoh: public function getDataPembaca().

Contoh pembuatan: nama file **Pembaca.php**

```php
<?php

namespace App\Controllers;

class Mahasiswa extends BaseController
{
    public function index()
    {
        return view('mahasiswa_view');
    }
}
```

Gambar 2.1 Struktur Kontroller

Model

a) Lokasi: app/Models/

b) Aturan Penamaan:

1) Nama file dan nama class harus sama (PascalCase).
2) PembacaModel.php → class PembacaModel
   Nama model biasanya diakhiri dengan kata Model.
   PembacaModel (dianjurkan)
   BarangModel (dianjurkan)
   Pembaca (tidak dianjurkan karena bisa rancu dengan Controller).
3) Model harus extends CodeIgniter\Model.
4) Properti $table, $primaryKey, dan $allowedFields harus sesuai dengan struktur database.
   Contoh: file dengan nama **PembacaModel.php**

```php
<?php

namespace App\Models;

use CodeIgniter\Model;

class MahasiswaModel extends Model
{
    protected $table = 'mahasiswa';
    protected $primaryKey = 'id';
    protected $allowedFields = ['nama', 'nim', 'jurusan'];
}
```

Gambar 2.2 Struktur Model

View

a) Lokasi: app/Views/

b) Aturan Penamaan:

1) Nama file view ditulis dengan huruf kecil dan **snake_case** (dianjurkan).
   pembaca_view.php (dianjurkan)
   tambah_pembaca.php (dianjurkan)
   pembacaView.php (tidak di anjurkan)
2) View berupa file PHP biasa (HTML + PHP).
3) Tidak perlu deklarasi class, cukup kode HTML dengan PHP.
4) Nama file view dipanggil tanpa ekstensi .php. dari class controller
   Contoh: return view('pembaca_view');
   Contoh View: file dengan nama **pembaca_view.php**

```
<!DOCTYPE html>
<html>
<head>
    <title>Data Mahasiswa</title>
</head>
<body>
    <h1>Daftar Mahasiswa</h1>
    <p>Ini adalah contoh view di CodeIgniter 4</p>
</body>
</html>
```

Gambar 2.3 Contoh View

Rangkuman Aturan

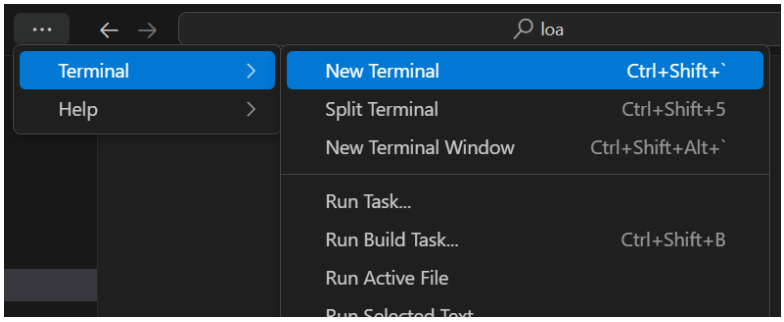
Tabel 2.1 Rangkuman aturan pembuatan MVC

Komponen	Lokasi	Aturan Penamaan	Contoh File / Class
Controller	app/Controllers/	PascalCase, sama dengan nama file	Pembaca.php → class Pembaca
Model	app/Models/	PascalCase + akhiran Model	PembacaModel.php → class PembacaModel
View	app/Views/	snake_case (huruf kecil, underscore)	pembaca_view.php

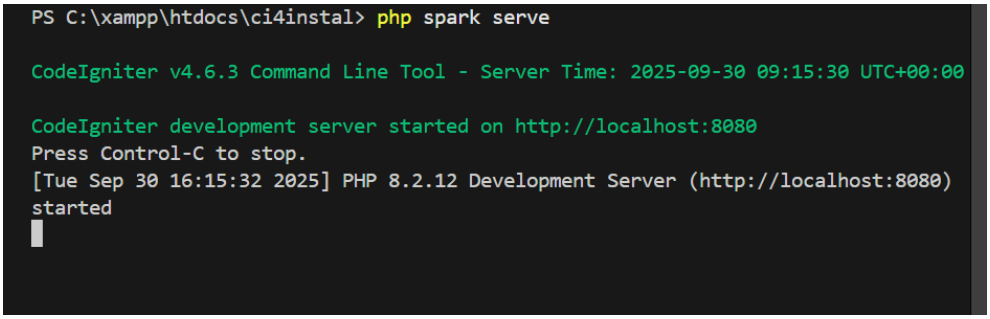
2.3 PRAKTIKUM MVC

Membuat Controller

a) Buka Projek sebelumnya dengan teks editor vs code dan jalankan php spark serve pada terminal (menu terminal – pilih new terminal)



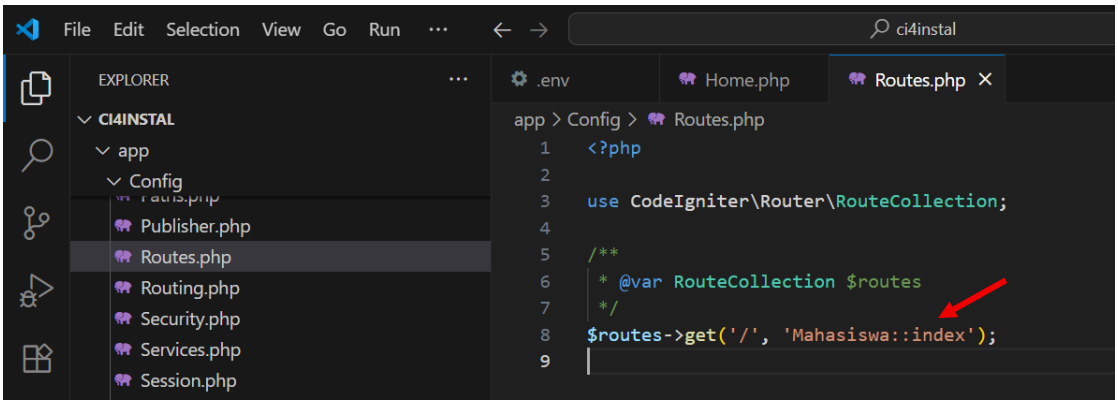
Gambar 2.4 Halaman Membuka Terminal



Gambar 2.5 Perintah PHP Spark Serve pada Terminal

b) Buka localhost: 8080 pada browser

c) Controller pertama kali yang di panggil: app/config/routes.php sekarang arahkan ke controller pembaca (yang akan kita buat pada tahapan berikutnya)

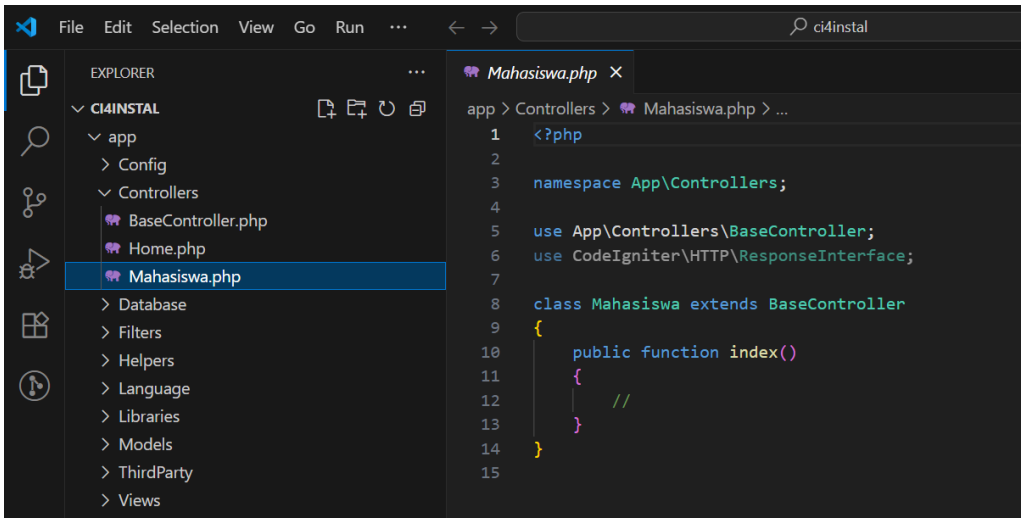


Gambar 2.6 Halaman Routes.PHP

d) Jalankan perintah untuk membuat controller baru:
php spark make: controller Pembaca

```
PS C:\xampp\htdocs\ci4instal> php spark make:controller Mahasiswa

CodeIgniter v4.6.3 Command Line Tool - Server Time: 2025-09-30 09
:19:41 UTC+00:00

File created: APPPATH\Controllers\Mahasiswa.php

PS C:\xampp\htdocs\ci4instal>
```

Gambar 2.7 Membuat Controller Baru dari Terminal



Gambar 2.8 Hasil Pembuatan Controller dari Terminal

e) Buka file app/Controllers/Pembaca.php lalu isi kode sederhana:

Gambar 2.9 Perintah pada Controller Pembaca

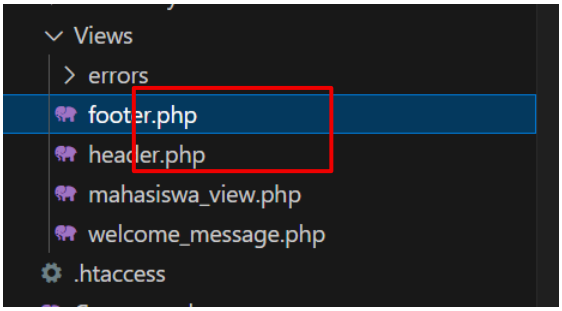
Membuat view

a) Buat file baru di folder app/Views dengan nama pembaca_view.php
b) Tambahkan kode berikut:

Gambar 2.10 Perintah pada Pembaca_View.PHP

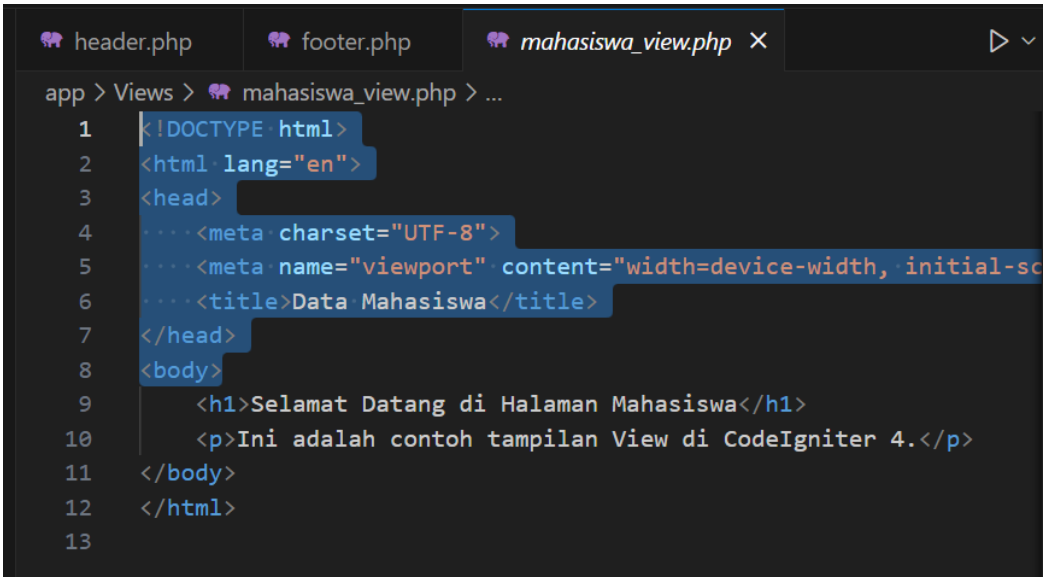
Memecah halaman view

a) Buat 2 file baru di view dengan nama header.php dan footer.php

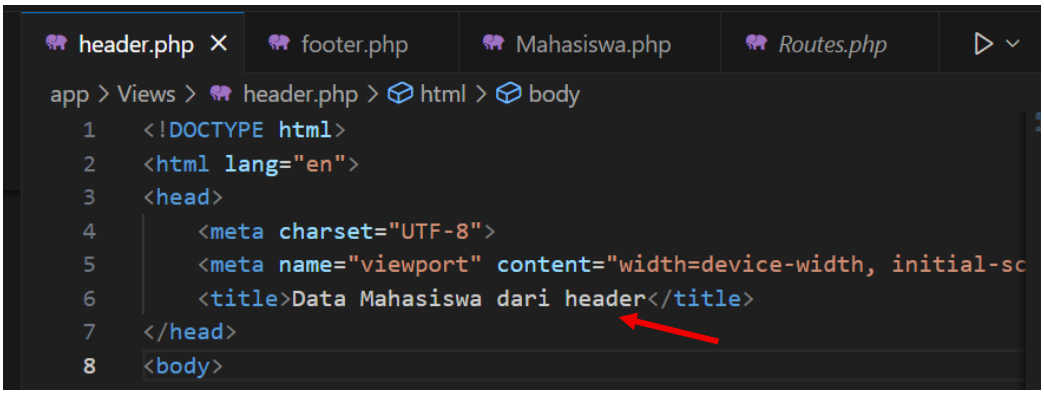


Gambar 2.11 File Footer dan Header yang Dibuat

b) Pisahkan sintaks atau koding line 1 – 8 yang ada di pembaca_view.php kedalam header.php kemudian ubahkan ubah atau tambahkan kata sebagai saat dijalankan nantinya (seperti yang di tunjuk arah panah atau gambar 2.12)

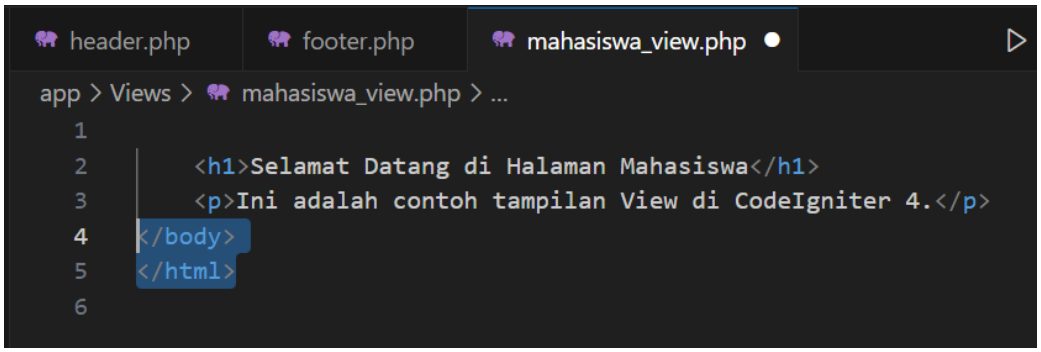


Gambar 2.12 Perintah yang Dimasukan ke Header.PHP

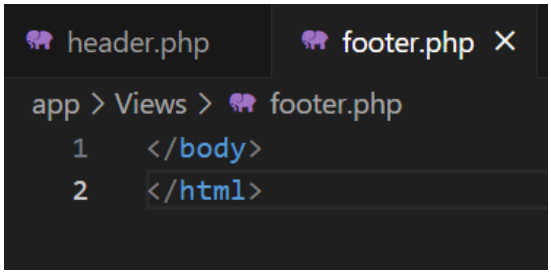


Gambar 2.13 Perintah pada Header.PHP

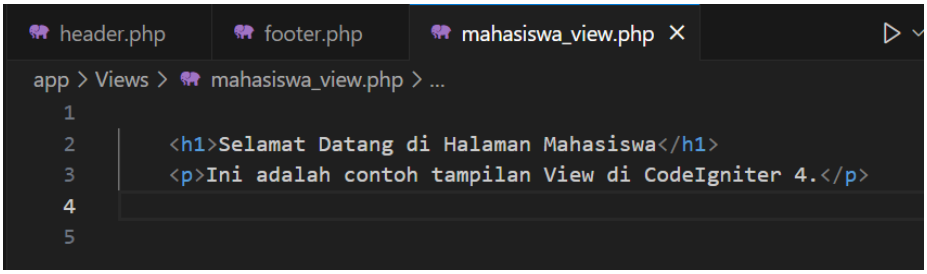
c) Pisahkan sintaks atau koding line 4-5 yang ada di pembaca_view.php terbaru kedalam footer.php



Gambar 2.14 Pemisahan pada Pembaca_View.PHP



Gambar 2.15 Pemisahan pada Fotter.PHP

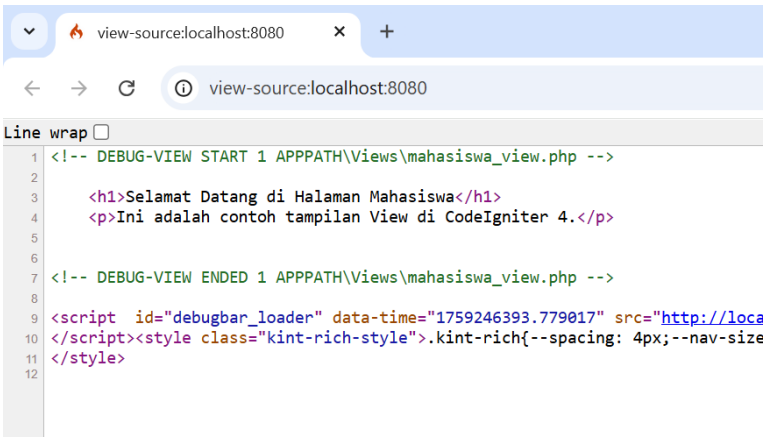


Gambar 2.16 Pemisahan pada Pembaca_View.PHP

d) Buka didalam web browser, pada bagian title masih localhost:8080 padahal tadi sudah di tulis <title>Data Pembaca dari header</title>. Artinya file header belum di muat

Gambar 2.17 Halaman Browser 8080

e) Klik kanan pada browser dan hasilnya seperti dibawah



Gambar 2.18 Hasil dari Inspek Halaman

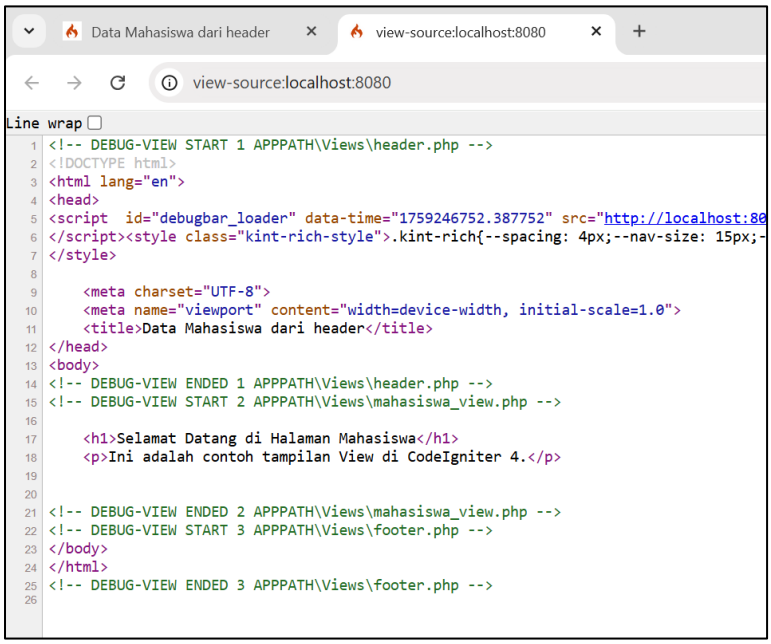
- Belum ada tag pembuka html, head, title dan body (dapat di identifikasi jika file header.php belum di load)
- Belum ada tag penutup body dan html pada bagian akhir (dapat di identifikasi jika file footer.php belum di load)

Tugas

Bagaimana caranya agar ke 3 file header.php; pembaca_vie.php; footer.php dapat di buka secara bersamaan dan menghasilkan seperti gambar di bawah ini

Gambar 2.19 Hasil Header

Hasil running setelah di perbaiki



Gambar 2.20 Hasil View Page Source Setelah di Perbaiki

Rangkuman Bab

Pada bab ini telah dibahas konfigurasi awal CodeIgniter 4 beserta konsep dasar arsitektur *Model–View–Controller* (MVC) yang menjadi landasan utama dalam pengembangan aplikasi menggunakan framework tersebut. Pemahaman terhadap konfigurasi awal dan struktur proyek merupakan langkah penting sebelum melakukan implementasi fitur-fitur aplikasi pada bab selanjutnya.

Pembaca telah mempelajari fungsi setiap direktori yang terdapat pada CodeIgniter 4, mengenal proses konfigurasi dasar aplikasi, serta memahami hubungan antara komponen Model, View, dan Controller dalam menangani permintaan dari pengguna. Setiap komponen memiliki tanggung jawab yang berbeda namun saling terhubung sehingga menghasilkan alur kerja aplikasi yang sistematis dan mudah dipelihara.

Selain itu, pembaca juga telah mempraktikkan pembuatan Controller, View, dan struktur halaman sederhana sebagai implementasi awal konsep MVC. Melalui latihan tersebut dapat dipahami bahwa Controller berperan sebagai pengatur alur proses aplikasi, Model digunakan untuk mengelola data, sedangkan View bertugas menampilkan informasi kepada pengguna. Pemisahan fungsi tersebut menjadikan struktur aplikasi lebih rapi dan memudahkan proses pengembangan pada tahap berikutnya.

Penguasaan konsep MVC menjadi salah satu kompetensi dasar yang harus dimiliki sebelum membangun aplikasi berbasis CodeIgniter 4. Seluruh fitur yang akan dikembangkan pada bab-bab berikutnya, seperti koneksi database, pembuatan form input, pengolahan data, validasi, hingga implementasi CRUD, akan memanfaatkan pola kerja yang sama. Oleh karena itu, pemahaman terhadap konsep MVC akan sangat membantu pembaca dalam mengikuti seluruh rangkaian praktikum.

Sebelum melanjutkan ke Bab 3, pastikan aplikasi CodeIgniter 4 telah berhasil dijalankan dengan baik, struktur direktori proyek telah dipahami, serta Controller dan View yang telah dibuat dapat ditampilkan melalui browser tanpa mengalami kesalahan. Keberhasilan menyelesaikan tahapan ini menunjukkan bahwa lingkungan pengembangan telah siap digunakan untuk membangun aplikasi yang lebih kompleks.

Pada bab berikutnya, pembaca akan mulai menghubungkan aplikasi dengan basis data MySQL, membuat Model untuk mengelola data, menyusun antarmuka menggunakan Bootstrap 5, serta membangun form input sebagai awal implementasi aplikasi CRUD. Dengan bekal pemahaman mengenai konsep MVC pada bab ini, proses pengembangan aplikasi pada bab berikutnya akan menjadi lebih mudah dipahami dan diterapkan.

Refleksi Pembelajaran

Setelah menyelesaikan bab ini, pembaca diharapkan tidak hanya mampu melakukan konfigurasi awal aplikasi CodeIgniter 4, tetapi juga memahami struktur dan alur kerja aplikasi yang dibangun menggunakan pola ***Model–View–Controller*** **(MVC)**. Pemahaman terhadap konsep tersebut penting karena proses pengembangan pada bab-bab berikutnya akan terus menggunakan hubungan antara Model, View, dan Controller. Dengan memahami fungsi setiap komponen sejak awal, pembaca akan lebih mudah membaca struktur proyek, menentukan lokasi file, serta memahami alur proses ketika sebuah aplikasi dijalankan.

Pembaca juga perlu mencoba kembali proses pembuatan Controller dan View secara mandiri untuk memastikan bahwa konsep yang telah dipelajari benar-benar dipahami. Latihan secara berulang akan membantu pembaca membedakan tanggung jawab setiap komponen dalam MVC dan mengurangi ketergantungan terhadap contoh program yang terdapat dalam buku. Kemampuan memahami struktur aplikasi secara mandiri akan menjadi modal penting ketika pembaca mulai membangun fitur yang lebih kompleks pada tahap berikutnya.

Penerapan dalam Dunia Nyata

Konsep *Model–View–Controller* yang dipelajari pada bab ini memiliki penerapan yang luas dalam pengembangan aplikasi berbasis web. Pemisahan antara pengelolaan data, proses aplikasi, dan tampilan membantu tim pengembang bekerja secara lebih terorganisasi. Dalam proyek pengembangan perangkat lunak, struktur yang jelas akan mempermudah proses pemeliharaan aplikasi ketika terjadi perubahan kebutuhan maupun ketika fitur baru perlu ditambahkan.

Penerapan struktur MVC juga memberikan keuntungan ketika sebuah aplikasi dikembangkan oleh beberapa orang dalam satu tim. Setiap bagian aplikasi dapat dikerjakan dan dipelihara secara lebih terarah sesuai dengan tanggung jawab masing-masing komponen. Dengan demikian, pemahaman mengenai MVC yang diperoleh melalui praktikum tidak hanya berguna untuk menyelesaikan tugas perkuliahan, tetapi juga menjadi dasar keterampilan yang dapat diterapkan ketika pembaca mengembangkan proyek aplikasi web secara mandiri maupun dalam lingkungan kerja.

Tips Pengembangan Aplikasi

Dalam mengembangkan aplikasi menggunakan CodeIgniter 4, biasakan untuk memahami fungsi setiap direktori dan menempatkan file sesuai dengan struktur yang telah disediakan oleh framework. Hindari menempatkan seluruh kode program pada satu file karena kebiasaan tersebut dapat membuat aplikasi sulit dipahami dan dipelihara. Gunakan Controller, Model, dan View sesuai dengan tanggung jawabnya agar struktur aplikasi tetap terorganisasi.

Selain itu, biasakan untuk menggunakan penamaan file, class, method, dan variabel yang konsisten. Penamaan yang jelas akan membantu pembaca maupun pengembang lain memahami tujuan dari setiap bagian kode. Setiap kali melakukan perubahan konfigurasi, jalankan kembali aplikasi untuk memastikan perubahan tersebut tidak menimbulkan kesalahan pada bagian lain. Kebiasaan melakukan pemeriksaan secara bertahap akan membantu menemukan kesalahan lebih awal dan membuat proses pengembangan menjadi lebih terkontrol.

Persiapan Menuju Bab Berikutnya

Setelah menyelesaikan bab ini, pastikan konfigurasi awal CodeIgniter 4 telah berjalan dengan baik dan aplikasi dapat diakses melalui browser. Pastikan pula pembaca telah memahami struktur dasar proyek serta dapat menjelaskan perbedaan fungsi Model, View, dan Controller. Pemahaman tersebut akan menjadi dasar ketika aplikasi mulai dihubungkan dengan basis data dan digunakan untuk mengelola informasi secara dinamis.

Pada bab berikutnya, pembahasan akan berlanjut pada proses membangun database dan form input menggunakan CodeIgniter 4 serta Bootstrap 5. Konsep MVC yang telah dipelajari pada bab ini akan digunakan kembali dalam proses tersebut. Oleh karena itu, pembaca disarankan memastikan seluruh latihan pada bab ini telah berhasil dijalankan sebelum melanjutkan ke tahap berikutnya sehingga proses pengembangan aplikasi dapat dilakukan secara bertahap dan terstruktur.

BAB 3
DATABASE DAN FORM INPUT DENGAN BOOTSTRAP 5

Setelah memahami konfigurasi awal aplikasi dan konsep *Model–View–Controller* (MVC), tahapan berikutnya dalam pengembangan aplikasi web adalah menghubungkan aplikasi dengan basis data serta membangun antarmuka yang dapat digunakan untuk mengelola data. Basis data merupakan salah satu komponen utama dalam sebuah sistem informasi karena berfungsi sebagai tempat penyimpanan seluruh informasi yang akan diproses oleh aplikasi. Tanpa adanya basis data, aplikasi hanya mampu menampilkan informasi secara statis dan tidak dapat melakukan pengelolaan data secara dinamis.

Pada framework CodeIgniter 4, proses pengolahan data dilakukan melalui komponen Model yang berfungsi sebagai penghubung antara aplikasi dengan basis data. Seluruh proses seperti membaca, menambahkan, mengubah, maupun menghapus data dilakukan melalui Model sehingga logika akses data tidak ditempatkan secara langsung pada Controller. Pendekatan ini menjadikan struktur aplikasi lebih rapi, mudah dipahami, dan sesuai dengan prinsip pengembangan perangkat lunak modern.

Sebelum aplikasi dapat berinteraksi dengan basis data, diperlukan proses konfigurasi koneksi agar CodeIgniter 4 mengetahui informasi server database yang digunakan. Konfigurasi tersebut meliputi nama host, nama basis data, nama pengguna, kata sandi, serta jenis *database driver* yang digunakan. Setelah konfigurasi berhasil dilakukan, aplikasi dapat mulai melakukan komunikasi dengan database untuk menyimpan maupun mengambil data sesuai kebutuhan.

Selain aspek pengolahan data, sebuah aplikasi juga memerlukan antarmuka yang baik agar mudah digunakan oleh pengguna. Oleh karena itu, pada bab ini digunakan Bootstrap 5 sebagai framework CSS yang membantu mempercepat proses pembuatan tampilan aplikasi. Dengan memanfaatkan komponen-komponen Bootstrap, tampilan form menjadi lebih rapi, responsif, dan konsisten tanpa harus menulis kode CSS dari awal. Penggunaan Bootstrap juga mempermudah pembaca dalam memahami bagaimana membangun antarmuka aplikasi yang sederhana namun tetap profesional.

Melalui praktikum pada bab ini, pembaca akan mulai membangun aplikasi pengelolaan data dengan membuat struktur basis data, melakukan konfigurasi koneksi database, menyusun Model, Controller, dan View, serta membuat halaman input data menggunakan Bootstrap 5. Seluruh tahapan tersebut disusun secara sistematis sehingga pembaca dapat memahami hubungan antara data yang tersimpan di database dengan tampilan yang muncul pada aplikasi.

Pemahaman terhadap materi pada bab ini menjadi dasar yang sangat penting sebelum memasuki proses implementasi fitur CRUD secara lengkap pada bab berikutnya. Oleh karena itu, setiap langkah praktikum perlu dilakukan secara berurutan agar seluruh komponen aplikasi dapat saling terhubung dengan baik. Dengan menyelesaikan seluruh kegiatan pada

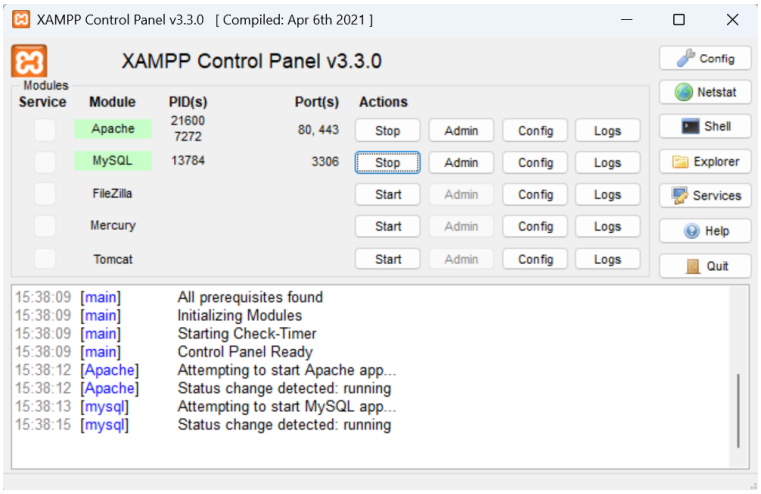
bab ini, pembaca diharapkan telah memiliki fondasi yang kuat dalam membangun aplikasi web berbasis database menggunakan CodeIgniter 4.

3.1 PEMBAHASAN PADA BAB INI

a) Membuat Database
b) Membuat Tabel
c) Setting konfigurasi awal.env
d) Control, View dan Model
e) Bootstrap 5, CSS dan Javascript
f) Membuat form input data

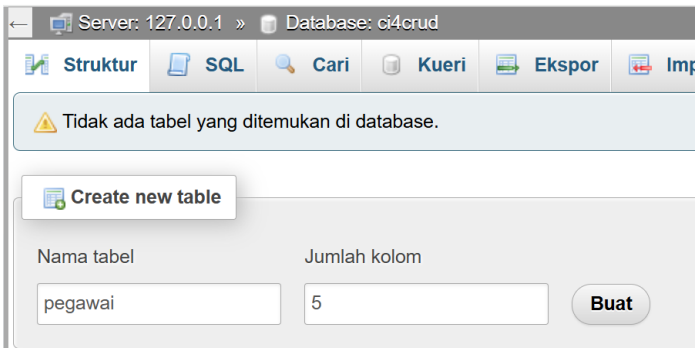
3.2 IMPLEMENTASI

1) Pastikan web server sudah di aktifkan (apache dan mysql)



Gambar 3.1 Server XAMPP

2) Buat database di localhost/phpMyAdmin dengan nama bebas misalkan: **ci4crud** kemudian buat table bernama **pegawai** dengan isian **5 (lima) kolom**



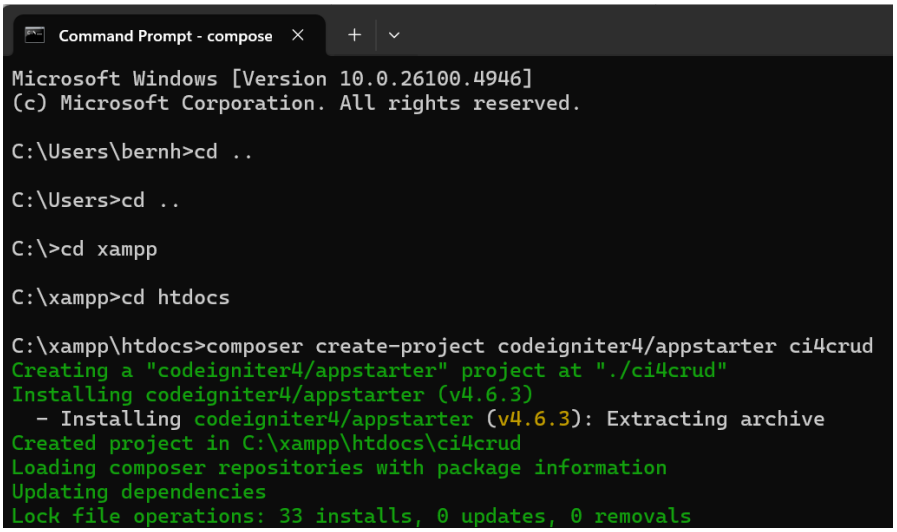
Gambar 3.2 Membuat Datasbase

3) Atur strukturnya sebagai berikut ini

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra
1	id	int(11)			Tidak	*Tidak ada*		AUTO_INCREMENT
2	nama	varchar(50)	utf8mb4_general_ci		Tidak	*Tidak ada*		
3	email	varchar(50)	utf8mb4_general_ci		Tidak	*Tidak ada*		
4	bidang	enum('finance', 'marketing', 'hr')	utf8mb4_general_ci		Tidak	*Tidak ada*		
5	alamat	varchar(100)	utf8mb4_general_ci		Tidak	*Tidak ada*		

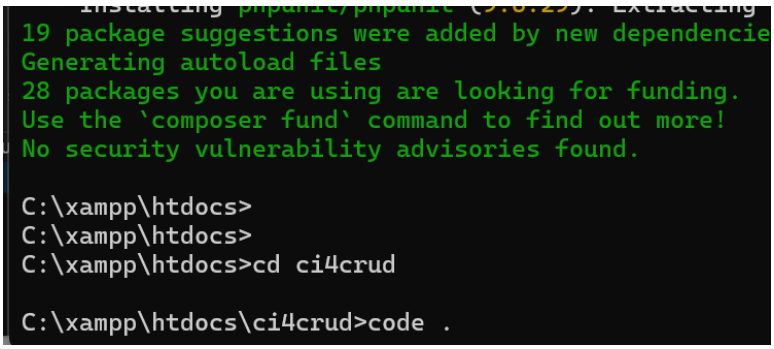
Gambar 3.3 Stuktur Database

4) Instal CI4 menggunakan terminal Command Prompt. Kali ini instal code igniter 4 dengan perintah: composer create-project codeigniter4/appstarter ci4crud
Tinggu prosesnya sampai selesai



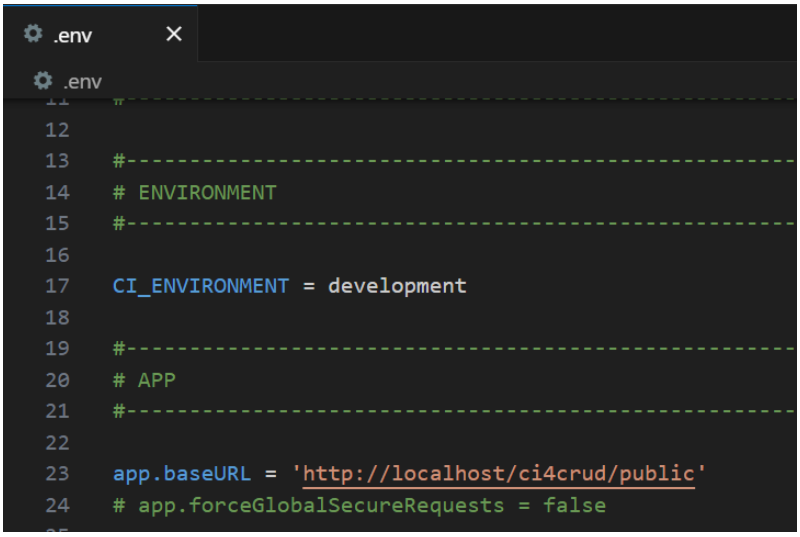
Gambar 3.4 Instal Codeigniter 4 dengan CMD

5) Arahkan ke direktori penginstalan di ci4crud, buka dengan vs code debgan perintah code. (titik) lalu tekan enter

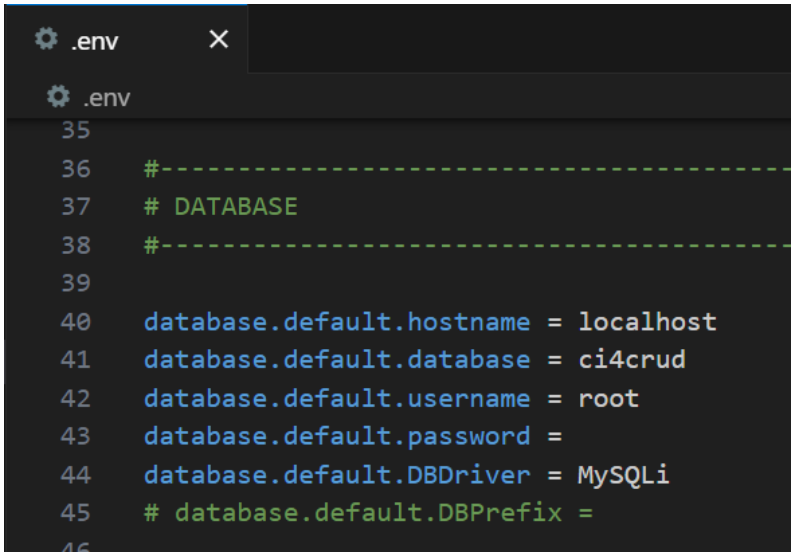


Gambar 3.5 Perintah Membuka Vs Code dengan CMD

6) Duplicate file env dan hasil duplicate di ganti nama menjadi .env lalu edit sintacnya menjadi seperti dibawah ini. Simpan hasil perubahan dan close file .env tersebut

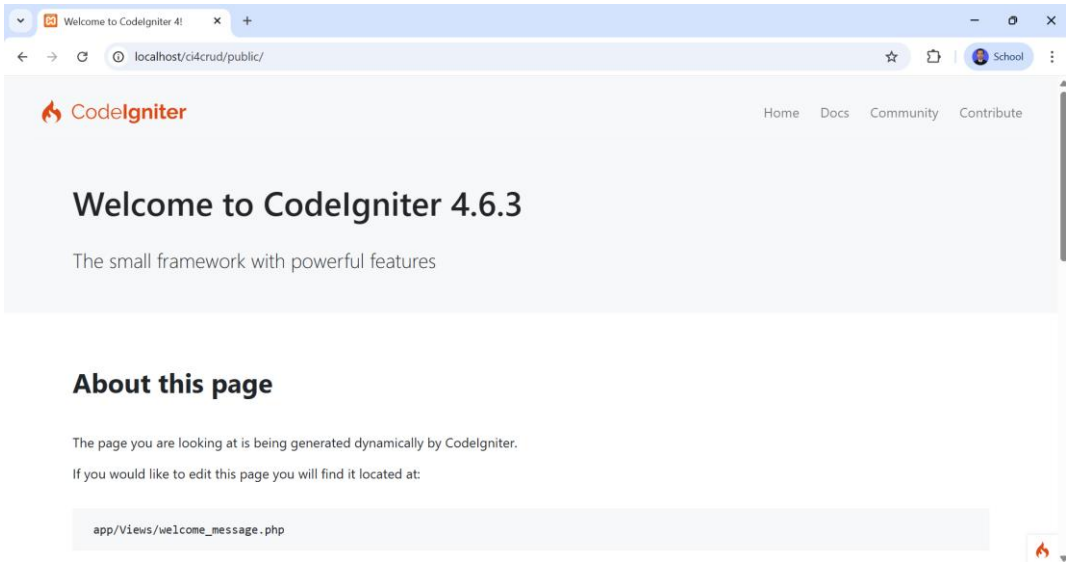


Gambar 3.6 Sesuaikan BaseURL



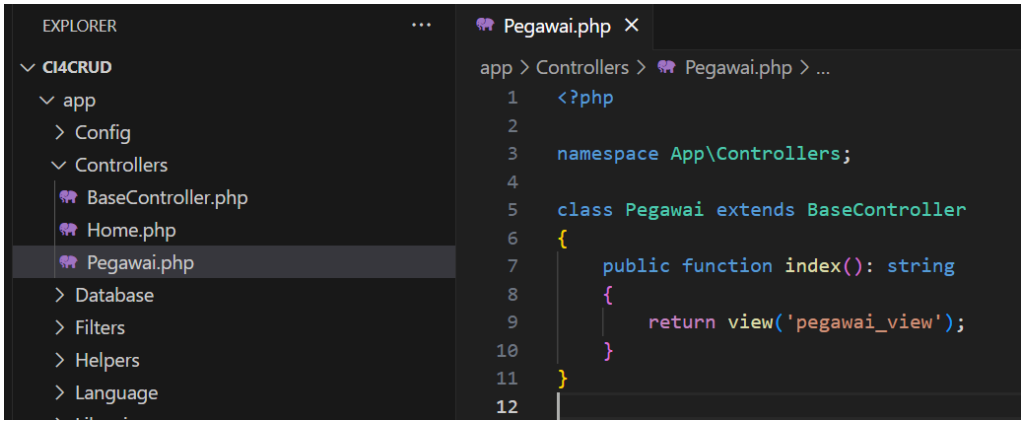
Gambar 3.7 Sesuaikan Database

7) Silahkan pastikan dan coba di buka hasil penginstalan tersebut dengan web browser dengan mengunjungi Alamat *http://localhost/ci4crud/public*



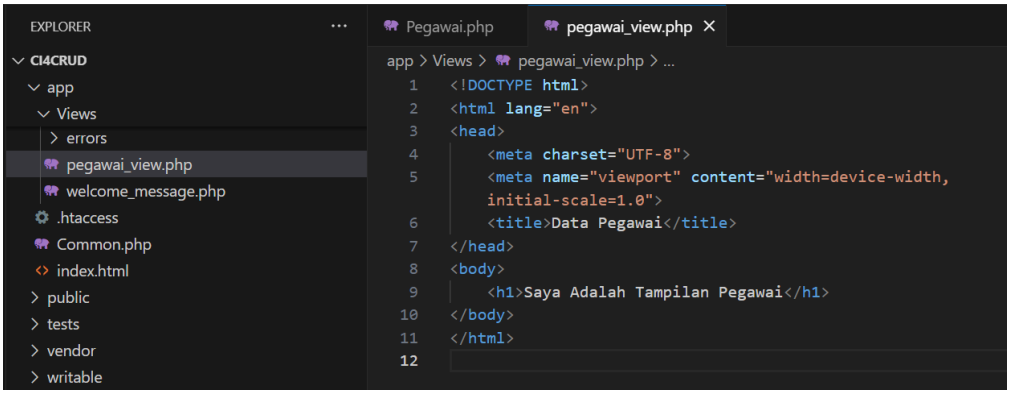
Gambar 3.8 Halaman Awal CodeIgniter

8) Buat controller baru bernama **Pegawai.php** pada CI4CRUD/app/controllers. Bisa duplicate dari controller Home. Namun nama file dan nama kelas harus di sesuaikan Kembali serta arahkan ke halaman "pegawai_view"



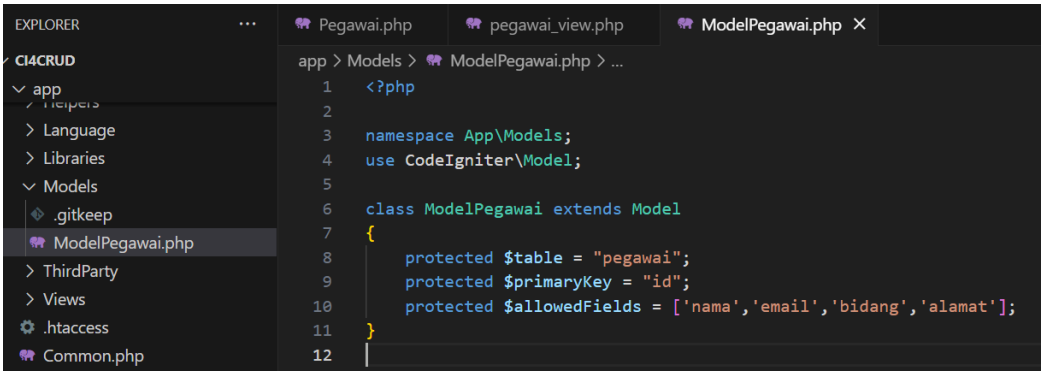
Gambar 3.9 Perintah Pegawai.PHP

9) Buat file **pagawai_view.php** pada direktori CI4CRUD/app/Views



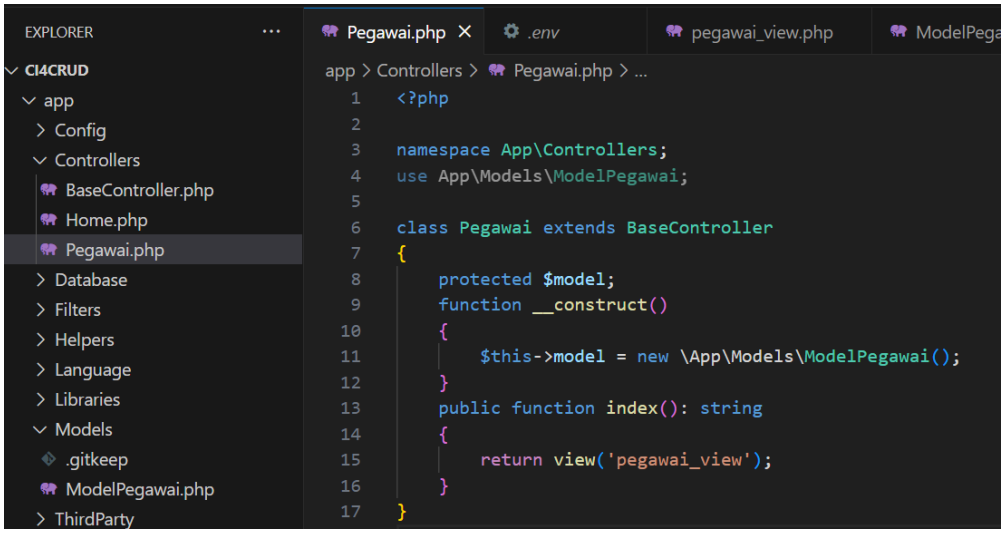
Gambar 3.10 Perintah View Pegawai

10) Buat **ModelPegawai.php** pada direktori CI4CRUD/app/Models dan tulis sintak seperti dibawah ini



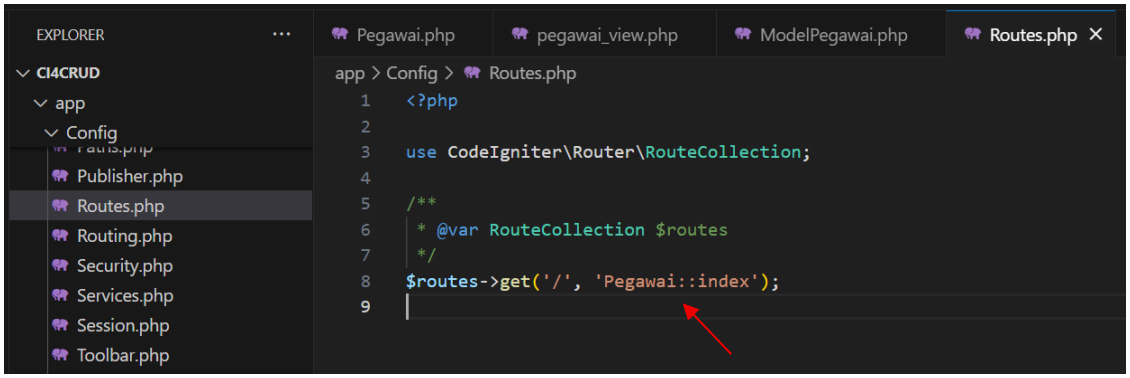
Gambar 3.11 Perintah Model Pegawai

11) Lakukan load model yang baru di buat ini dari controller **Pegawai.php** atur menjadi seperti dibawah ini



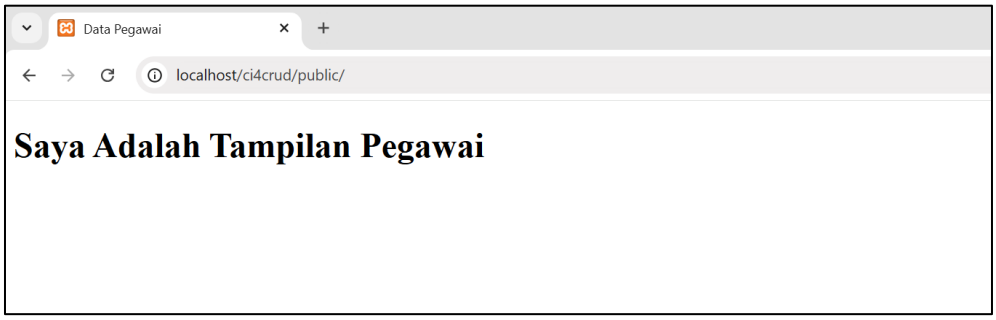
Gambar 3.11 Perintah Load Model pada Pegawai.PHP

12) Ubah default controller dan arahkan ke controller yang sudah di buat sebelumnya dengan nama pegawai. Buka file Routes.php pada direktori CI4CRUD/app/config dan arahkan ke controller Pegawai seperti dibawah ini lalu simpan.



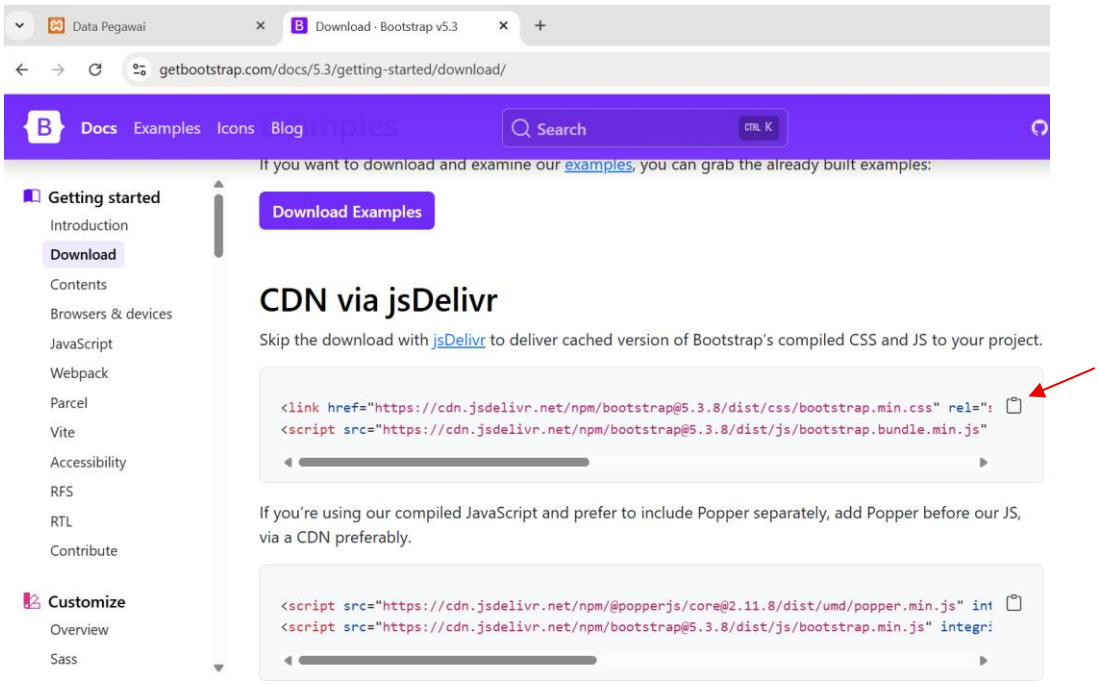
Gambar 3.12 Ubah Default Controller pada Routes

13) Refresh halaman browser projek kita dan pastikan terbuka seperti ini



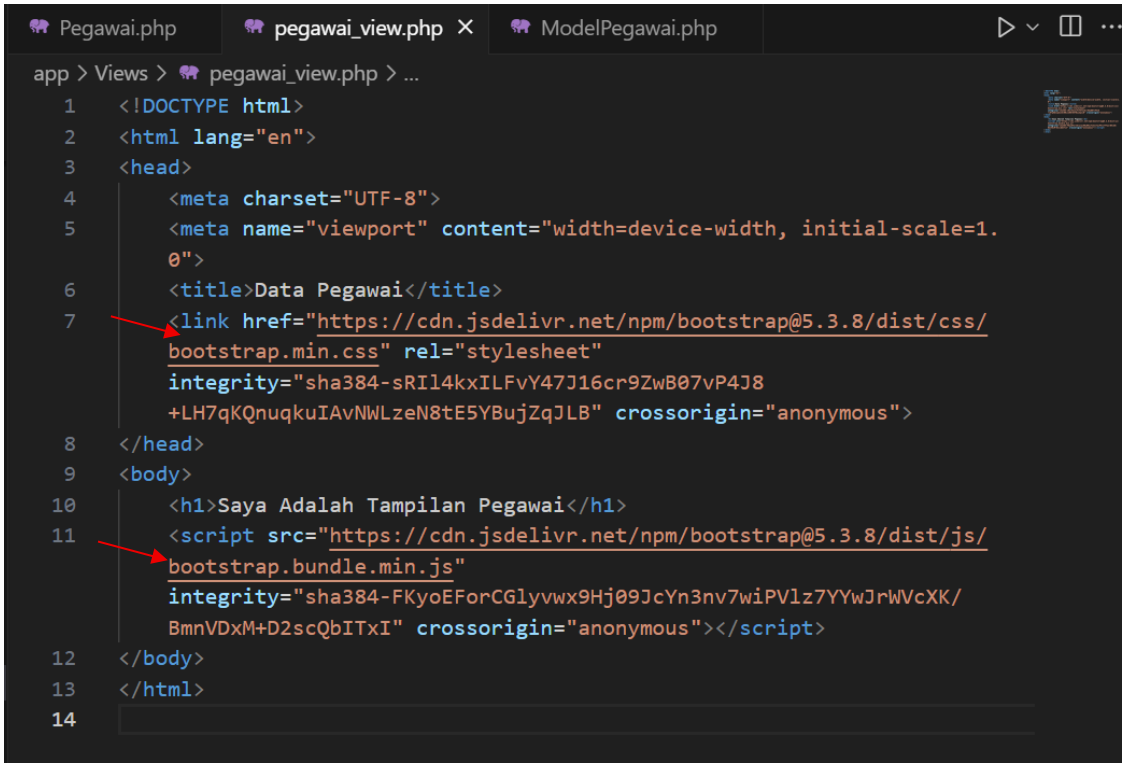
Gambar 3.13 Halaman Default Setelah Diubah

14) Atur tampilan pegawai tersebut dengan memanfaatkan bootsratp 5. Silahkan buka halaman bootstap *https://getbootstrap.com/docs/5.3/getting-started/download/* copy link css dan js bootstarp.



Gambar 3.14 Halaman Bootstrap 5

15) Paste pada file pegawai_view.php. untuk CSS letakkan pada baris sebelum penutup </head> dan java script pada baris sebelum penutup </body>.



Gambar 3.15 Masukan Bootstrap ke Projek

16) Supaya tampilan coding tidak terlalu kekanan, bisa menekan tombol Alt + Z pada keyboard.
17) Supaya lebih rapi dan mudah diingat berikan komentar sepian link tersebut. Lalu buat tag div class container untuk membungkus konten yang akan kita buat.

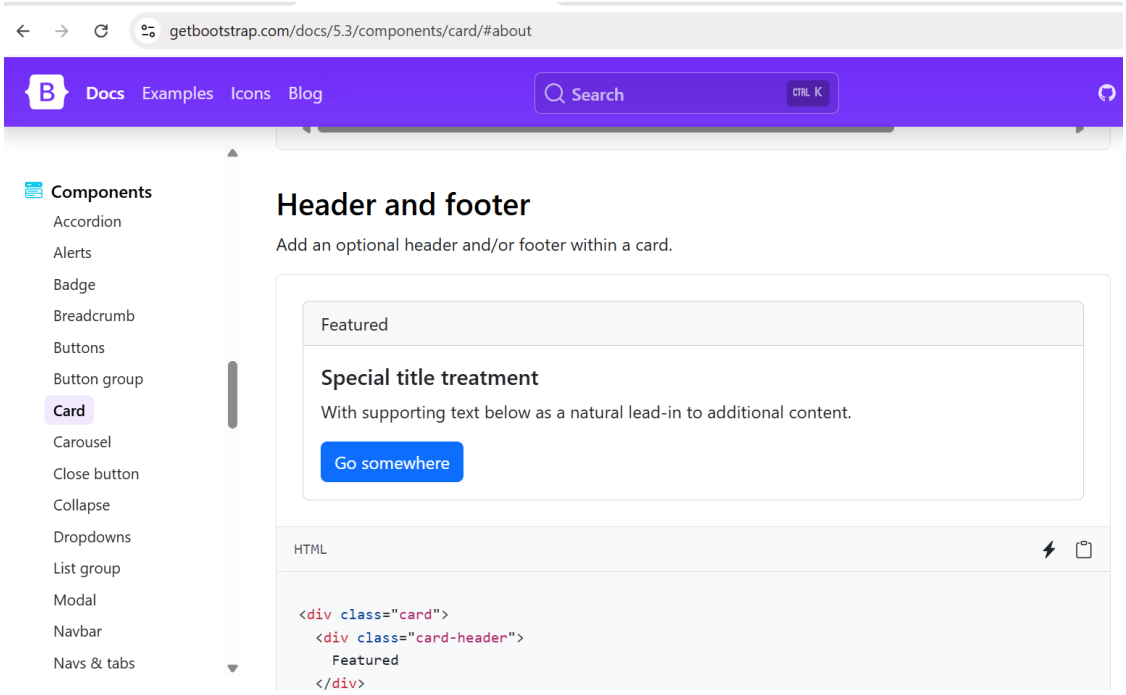
Pegawai.php | pegawai_view.php | ModelPegawai.php

app > Views > pegawai_view.php > html > body > div.container

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.
    0">
    <title>Data Pegawai</title>
    <!-- CSS BOOTSTRAP -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/css/
    bootstrap.min.css" rel="stylesheet"
    integrity="sha384-sRIl4kxILFvY47J16cr9ZwB07vP4J8
    +LH7qKQnuqkuIAvNWLzeN8tE5YBujZqJLB" crossorigin="anonymous">
</head>
<body>
    <!-- CONTAINER -->
    <div class="container">

    </div>
    <!-- JAVA SCRIPT BOOTSTRAP-->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/js/
    bootstrap.bundle.min.js"
    integrity="sha384-FKyoEForCGlyvwx9Hj09JcYn3nv7wiPVlz7YYwJrWVcXK/
    BmnVDxM+D2scQbITxI" crossorigin="anonymous"></script>
</body>
</html>
```

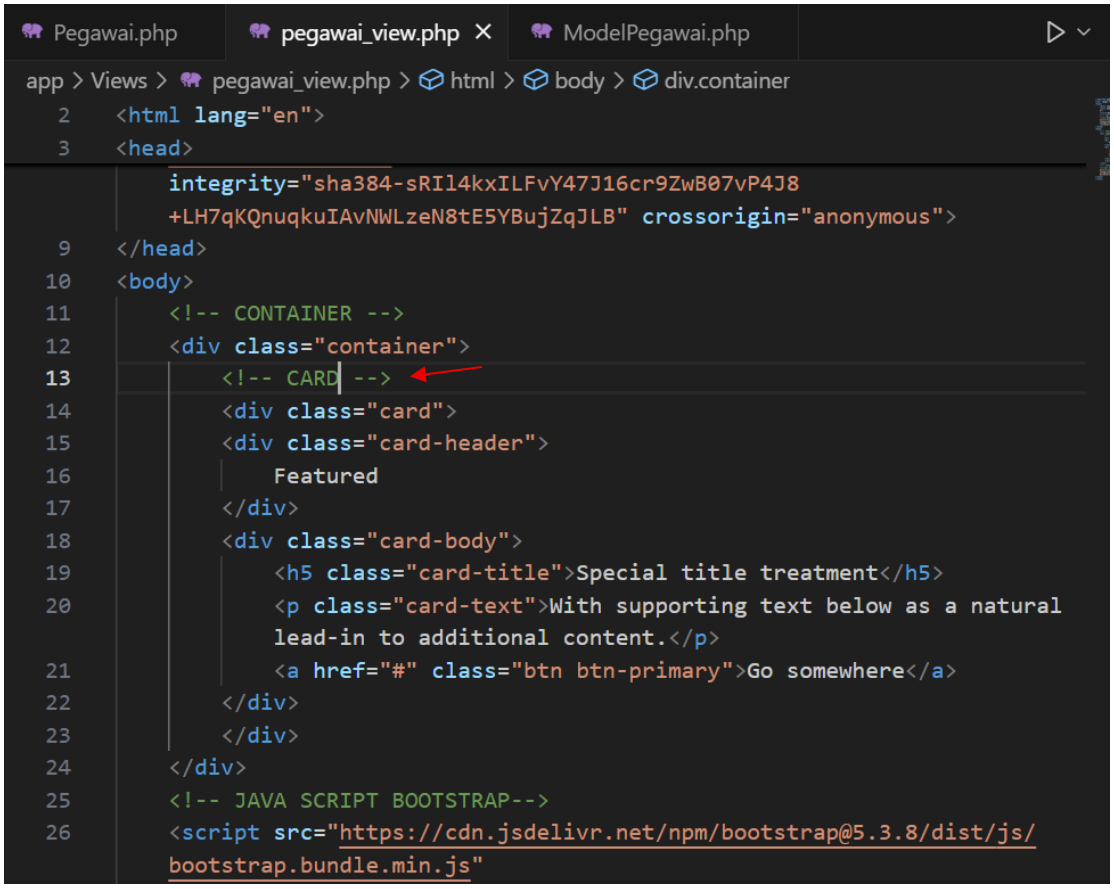
Gambar 3.16 Bungkus Tag Container

18) Cari dokumentasi card pada bootstrap lalu pilih sesuai keinginan. Misalkan disini pilih yang header and footer lalu klik copy pada coding nya



Gambar 3.17 Copy Dokumen Card dari Bootstrap

19) Paste pada projek tadi didalam container. Jangan lupa berikan komentar



Gambar 3.18 Paste Perintah Card dari Bootstrap ke Projek

```
    integrity="sha384-FKyoEForCGlyvwx9Hj09JcYn3nv7wiPVlz7YYwJrWVcXK/
    BmnVDxM+D2scQbITxI" crossorigin="anonymous"></script>
</body>
</html>
```

Gambar 3.19 Lanjutan Perintah Gambar 3.18

20) Ganti judul dari card tersebut menjadi data pegawai dan untuk sementara isi dari body card tersebut di hapus terlebih dahulu.

```
<body>
    <!-- CONTAINER -->
    <div class="container">
        <!-- CARD -->
        <div class="card">
        <div class="card-header">
            Data Pegawai
        </div>
        <div class="card-body">

        </div>
        </div>
    </div>
    <!-- JAVA SCRIPT BOOTSTRAP-->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/js/
    bootstrap.bundle.min.js"
    integrity="sha384-FKyoEForCGlyvwx9Hj09JcYn3nv7wiPVlz7YYwJrWVcXK/
    BmnVDxM+D2scQbITxI" crossorigin="anonymous"></script>
</body>
</html>
```

Gambar 3.20 Sesuaikan Judul Card

21) Lihat hasilnya pada browser

Gambar 3.21 Hasil Card pada Browser

22) Anda dapat memodifikasi tampilan tersebut sesuka anda dengan menggunakan style yang anda bisa. Misalkan disini kita tambahkan style sendiri dan letakkan pada bawah css dari bootstarp. Dan tambahkan atribut pada class card-header.

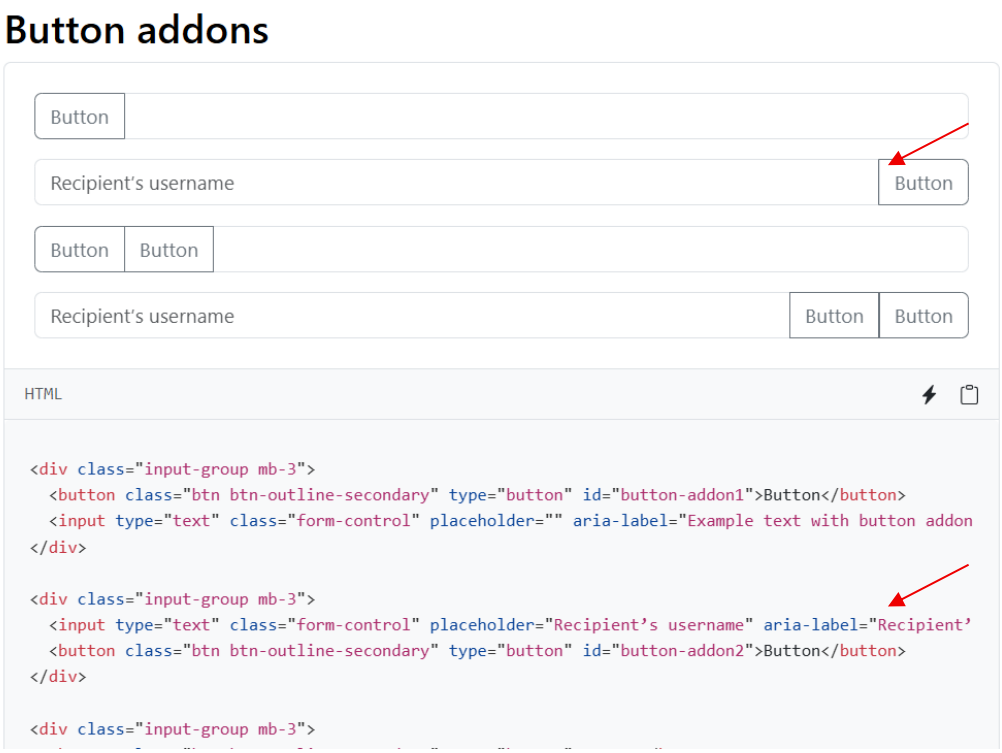
```
Pegawai.php | pegawai_view.php | ModelPegawai.php
app > Views > pegawai_view.php > html > head
2   <html lang="en">
3   <head>
        0">
6       <title>Data Pegawai</title>
7       <!-- CSS BOOTSTRAP -->
8       <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/css/
        bootstrap.min.css" rel="stylesheet"
        integrity="sha384-sRIl4kxILFvY47J16cr9ZwB07vP4J8
        +LH7qKQnuqkuIAvNWLzeN8tE5YBujZqJLB" crossorigin="anonymous">
9       <!-- CSS MANUAL -->
10      <style>
11          .container{
12              max-width: 800px;
13              margin-top: 20px;
14          }
15      </style>
16  </head>
17  <body>
18      <!-- CONTAINER -->
19      <div class="container">
20          <!-- CARD -->
21          <div class="card">
22          <div class="card-header bg-secondary text-white">
23              Data Pegawai
24          </div>
25          <div class="card-body">
26
```

Gambar 3.22 Modifikasi Tampilan dengan CSS Manual

23) Lihat hasilnya

Gambar 3.23 Hasil Modifikasi

24) Tambahkan tombol untuk pencarian. Cari dokumen input-group pada bootsrap. Misalakan disini pilih button addons yang bagian ke 2. Copy sintac untuk link yang ke 2 saja

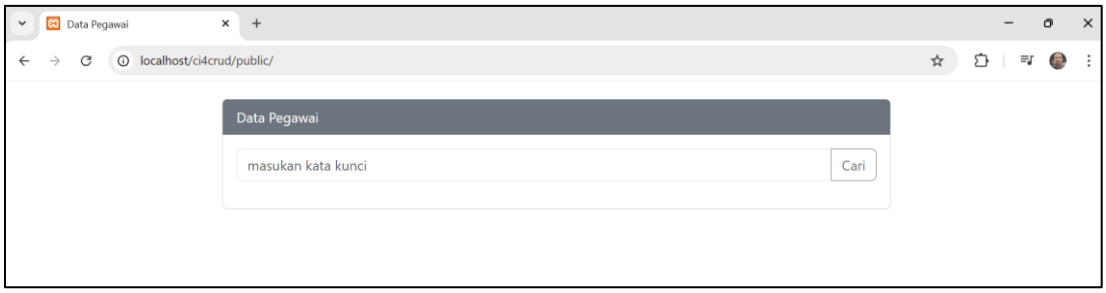


Gambar 3.24 Cari Dokumen Tombol pada Bootstrap

25) Paste didalam card body dan atur text sesuai kebutuhan seperti dibawah ini. Dan tambahkan nama = ”katakunci”

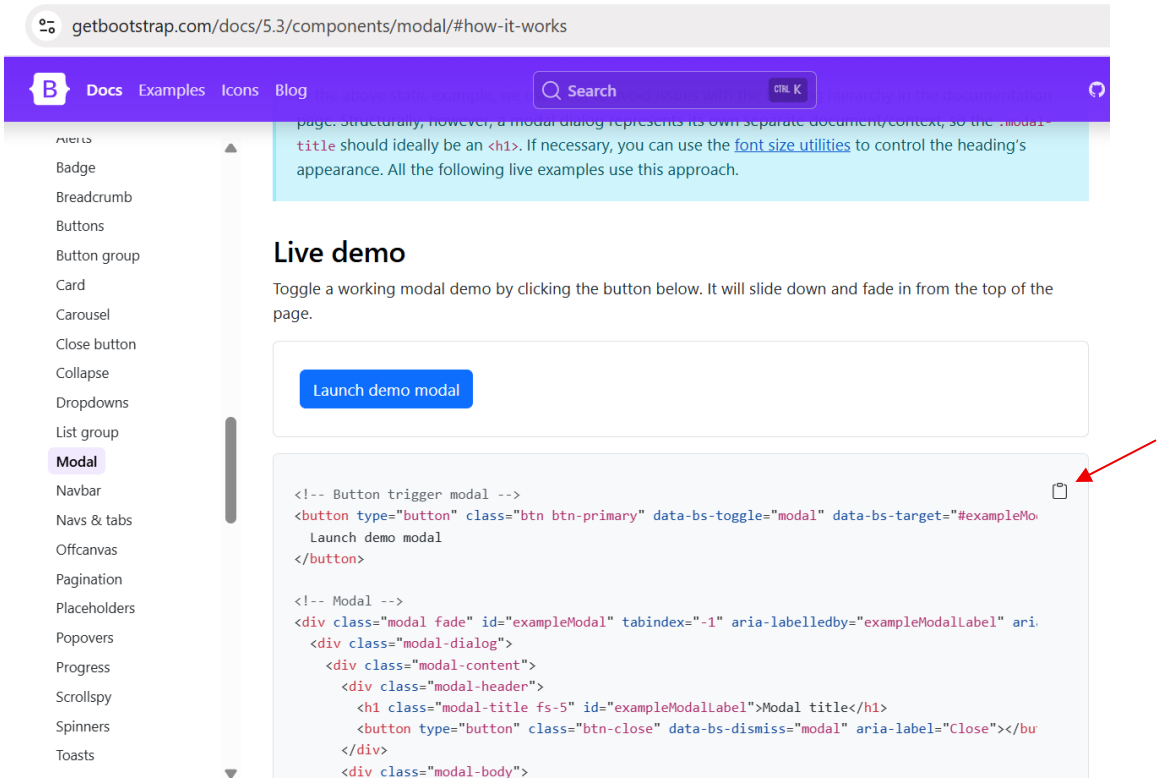
Gambar 3.25 Paste Perintah Tombol dari Bootstrap ke Projek

26) Hasilnya akan seperti ini



Gambar 3.26 Hasil Tombol pada Browser

27) Sekarang buat form untuk menginputkan data baru dibagian bawah pencarian. Kita gunakan fitur modal yang ada di bootstrap. Silahkan cari dokumen modal dari bootstrap. Disini gunakan modal yang Launc demo modal dan copykan codingnya.



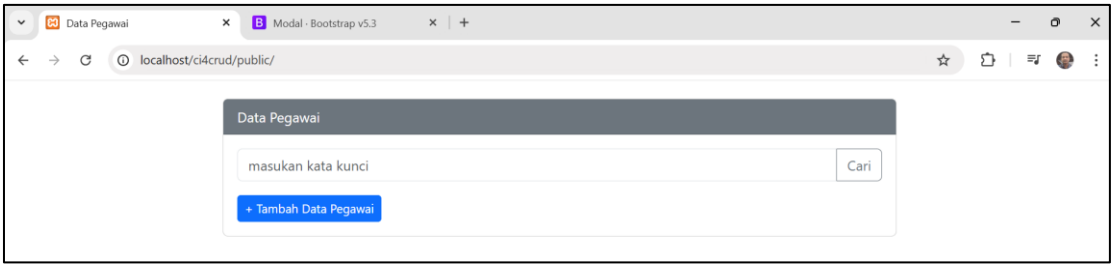
Gambar 3.27 Copy Dokumen Form pada Bootstrap

28) Paste coding tersebut tepat di bawah tag penutup form pencarian. Jangan lupa berikan komentar sebagai penanda misalkan “modal tambah data” dan sesuaikan codingngnya seperti dibawah ini

```
30                <button class="btn btn-outline-secondary" type="button" id="button-addon2">Cari</button>
31            </div>
32        </form>
33        <!-- Button trigger modal : TAMBAH DATA -->
34            <button type="button" class="btn btn-primary btn-sm" data-bs-toggle="modal"
              data-bs-target="#exampleModal">
35            + Tambah Data Pegawai
36            </button>
37
38            <!-- Modal -->
39            <div class="modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel"
              aria-hidden="true">
40            <div class="modal-dialog">
41                <div class="modal-content">
42                <div class="modal-header">
43                    <h1 class="modal-title fs-5" id="exampleModalLabel">Form Tambah Data Pegawai</h1>
44                    <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></
                      button>
45                </div>
46                <div class="modal-body">
47                    ...
48                </div>
49                <div class="modal-footer">
50                    <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Batal</
                      button>
51                    <button type="button" class="btn btn-primary">Simpan</button>
52                </div>
53                </div>
54            </div>
55            </div>
```

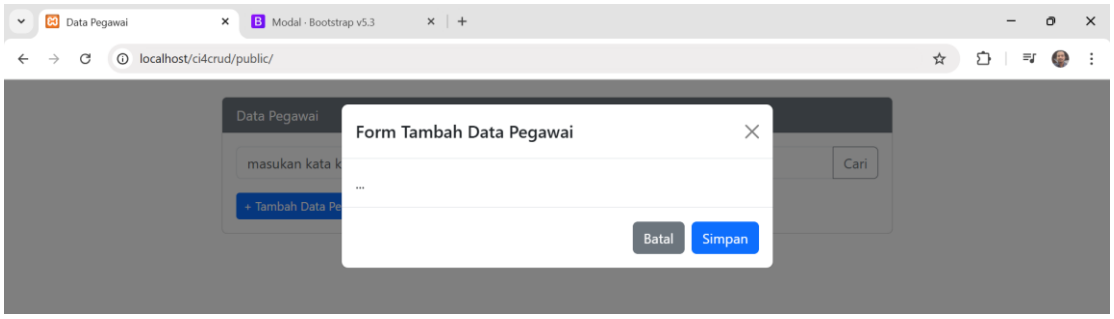
Gambar 3.28 Perintah Form Paste pada Projek

29) Simpan hasil pengerjaan dan coba lihat browser sambil mencoba klik tombolnya



Gambar 3.29 Hasil Form

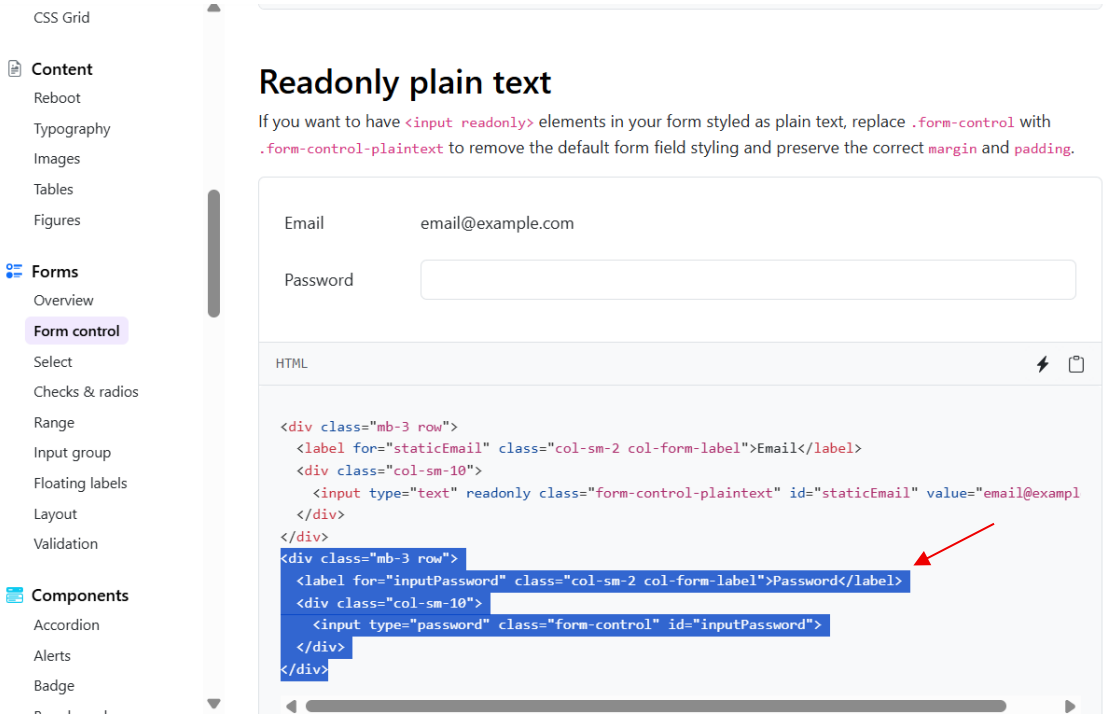
Saat di klik tambah data pegawai



Gambar 3.30 Hasil Saat di Klik

30) Sekarang buatkan inputan pada body modal tersebut. Cari form control pada bootstrap, dan kali ini pilih sebagai contoh pilih readonly plain text. Namun yang kita ambil hanya inputan password saja seperti yang di blok pada gambar di bawah ini.

Copy coding tersbut secara manual, block koding yang diinginkan kemudian ctrl+c / klik kanan copy



Gambar 3.31 Salin Input Body pada Bootstrap

31) Paste kebagian body modal dan jangan lupa berikan catatan komentar serta rubah menjadi seperti di bawah ini

Gambar 3.32 Paste pada Projek

32) Copy input text nama tersebut dan paste dibawahnya untuk input an email. Ubah nama tesebut menjadi email

```
47                  <!-- FORM INPUT NAMA -->
48                  <div class="mb-3 row">
49                      <label for="inputNama" class="col-sm-2 col-form-label">Nama</label>
50                      <div class="col-sm-10">
51                          <input type="text" class="form-control" id="inputNama">
52                      </div>
53                  </div>
54                  <div class="mb-3 row">
55                      <label for="inputEmail" class="col-sm-2 col-form-label">Email</label>
56                      <div class="col-sm-10">
57                          <input type="text" class="form-control" id="inputEmail">
58                      </div>
59                  </div>
```

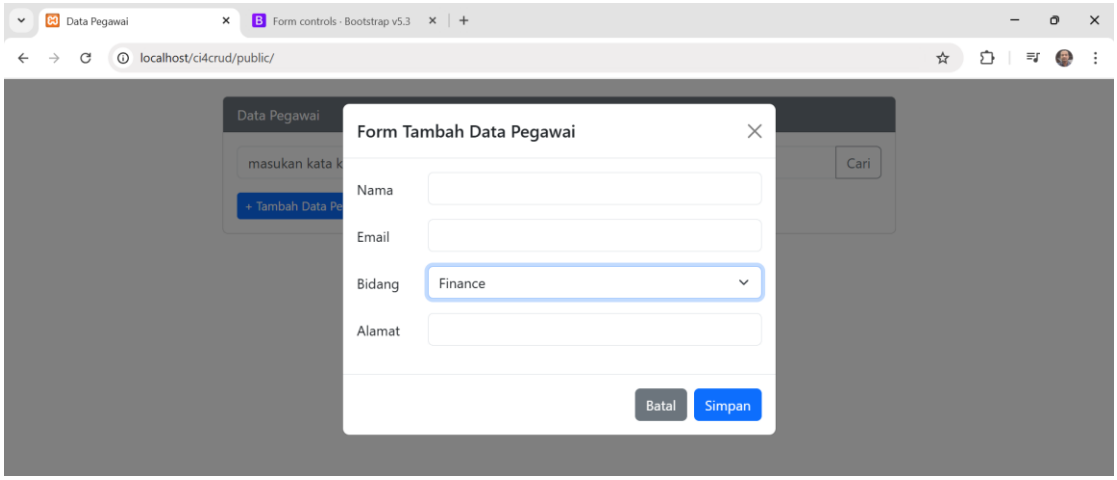
Gambar 3.33 Inputan Email

33) Copy lagi 2 input text tersebut dan paste dibagian bawahnya untuk inputan bidang dan Alamat. Namun pada inputang bidang kita rumah menggunakan select. Ubah seperti dibawah ini

```
54                  <div class="mb-3 row">
55                      <label for="inputEmail" class="col-sm-2 col-form-label">Email</label>
56                      <div class="col-sm-10">
57                          <input type="text" class="form-control" id="inputEmail">
58                      </div>
59                  </div>
60                  <div class="mb-3 row">
61                      <label for="inputBidang" class="col-sm-2 col-form-label">Bidang</label>
62                      <div class="col-sm-10">
63                          <select id="InputBidang" class="form-select">
64                              <option value="Finance">Finance</option>
65                              <option value="Marketing">Marketing</option>
66                              <option value="HR">HR</option>
67                          </select>
68                      </div>
69                  </div>
70                  <div class="mb-3 row">
71                      <label for="inputAlamat" class="col-sm-2 col-form-label">Alamat</label>
72                      <div class="col-sm-10">
73                          <input type="text" class="form-control" id="inputAlamat">
74                      </div>
75                  </div>
```

Gambar 3.34 Inputan Bidang dan Alamat

34) Silahkan lihat hasilnya di browser



Gambar 3.35 Hasil Form Input

Rangkuman Bab

Pada bab ini telah dibahas proses membangun koneksi antara aplikasi CodeIgniter 4 dengan basis data MySQL serta pembuatan antarmuka input data menggunakan Bootstrap 5. Tahapan tersebut merupakan awal dari proses pengembangan aplikasi yang mampu mengelola data secara dinamis. Dengan adanya koneksi database, aplikasi tidak lagi hanya menampilkan halaman statis, tetapi telah mampu melakukan proses penyimpanan dan pengelolaan informasi secara terstruktur.

Pembaca telah mempelajari proses pembuatan basis data, konfigurasi koneksi database pada CodeIgniter 4, penyusunan Model sebagai pengelola data, serta implementasi Controller dan View yang saling terhubung dalam pola *Model–View–Controller* (MVC). Selain itu, pembaca juga telah memahami bagaimana Bootstrap 5 digunakan untuk membangun tampilan form yang lebih rapi, responsif, dan mudah digunakan oleh pengguna.

Melalui kegiatan praktikum yang telah dilakukan, pembaca dapat melihat bahwa setiap data yang dimasukkan melalui form akan diproses oleh Controller, diteruskan ke Model, kemudian disimpan ke dalam basis data. Proses tersebut menunjukkan bagaimana setiap komponen dalam arsitektur MVC bekerja secara terpadu sehingga menghasilkan aplikasi yang terstruktur dan mudah dikembangkan. Pemahaman terhadap alur kerja ini akan sangat membantu ketika aplikasi mulai dikembangkan menjadi sistem yang lebih kompleks.

Penerapan Bootstrap 5 pada halaman input juga memberikan pengalaman kepada pembaca mengenai pentingnya antarmuka pengguna dalam sebuah aplikasi. Selain berfungsi untuk memperindah tampilan, penggunaan komponen Bootstrap membantu menciptakan halaman yang lebih konsisten, responsif, dan nyaman digunakan pada berbagai ukuran perangkat. Dengan demikian, aspek fungsional dan aspek visual aplikasi dapat berjalan secara seimbang.

Sebelum melanjutkan ke bab berikutnya, pastikan seluruh proses koneksi database telah berjalan dengan baik, data dapat dimasukkan melalui form tanpa mengalami kesalahan, serta struktur Model, Controller, dan View telah tersusun sesuai dengan konsep MVC yang telah dipelajari. Keberhasilan pada tahap ini menjadi indikator bahwa aplikasi telah siap untuk dikembangkan lebih lanjut.

Pada Bab 4, pembaca akan mulai mengimplementasikan proses validasi data serta memanfaatkan teknologi AJAX dan jQuery untuk meningkatkan interaktivitas aplikasi. Dengan memanfaatkan teknik tersebut, proses pengolahan data dapat dilakukan secara lebih cepat dan efisien tanpa harus memuat ulang seluruh halaman, sehingga aplikasi yang dibangun menjadi lebih responsif dan memberikan pengalaman penggunaan yang lebih baik.

Selain menguasai aspek teknis dalam membangun aplikasi, pembaca juga diharapkan mulai membiasakan diri untuk menerapkan prinsip pengembangan perangkat lunak yang sistematis. Setiap komponen yang dibuat, mulai dari struktur basis data, Model, Controller, hingga View, sebaiknya disusun secara terorganisir agar aplikasi mudah dipahami, dipelihara, dan dikembangkan pada masa mendatang. Kebiasaan mengikuti alur pengembangan yang terstruktur sejak tahap awal akan memberikan manfaat yang besar ketika pembaca

mengembangkan aplikasi dengan skala yang lebih kompleks, baik dalam kegiatan akademik maupun pada implementasi di dunia industri. Oleh karena itu, seluruh tahapan pada bab ini hendaknya dipahami tidak hanya sebagai latihan praktikum, tetapi juga sebagai penerapan konsep dasar pengembangan aplikasi web yang baik menggunakan framework CodeIgniter 4.

Refleksi Pembelajaran

Setelah menyelesaikan bab ini, pembaca diharapkan mulai memahami bahwa aplikasi web yang dinamis membutuhkan hubungan yang baik antara antarmuka aplikasi, proses pengolahan data, dan basis data. Proses membuat database, melakukan konfigurasi koneksi, menyusun Model, Controller, dan View, serta membuat form input merupakan tahapan yang saling berkaitan. Kesalahan pada salah satu bagian dapat memengaruhi proses pada bagian lainnya. Oleh karena itu, pembaca perlu memahami alur data secara keseluruhan dan tidak hanya berfokus pada penulisan kode pada satu komponen saja.

Pembaca juga disarankan untuk mencoba kembali proses praktikum secara mandiri dengan memperhatikan setiap perubahan yang terjadi pada aplikasi. Perhatikan bagaimana data yang dimasukkan melalui form dapat diteruskan ke proses aplikasi dan kemudian disimpan ke dalam database. Dengan melakukan latihan secara berulang, pembaca akan semakin memahami hubungan antara form, Controller, Model, dan database serta mampu menemukan kesalahan ketika aplikasi tidak memberikan hasil sesuai yang diharapkan.

Penerapan dalam Dunia Nyata

Konsep pengelolaan database dan form input yang dipelajari pada bab ini merupakan bagian yang sangat umum dalam pembangunan sistem informasi. Berbagai aplikasi seperti sistem akademik, inventaris, penjualan, kepegawaian, perpustakaan, maupun administrasi membutuhkan mekanisme untuk menerima masukan dari pengguna dan menyimpannya secara terstruktur ke dalam database. Dengan demikian, keterampilan yang diperoleh melalui praktikum pada bab ini dapat diterapkan pada berbagai jenis studi kasus pengembangan aplikasi.

Penggunaan Bootstrap 5 juga menunjukkan bahwa pengembangan aplikasi tidak hanya memperhatikan bagaimana data diproses, tetapi juga bagaimana pengguna berinteraksi dengan aplikasi. Form yang tersusun dengan baik dapat membantu pengguna memahami informasi apa yang harus dimasukkan dan mengurangi kesalahan dalam proses pengisian data. Dalam pengembangan aplikasi di lingkungan nyata, aspek fungsional dan tampilan perlu diperhatikan secara bersama-sama agar aplikasi dapat digunakan dengan baik oleh pengguna.

Tips Pengembangan Aplikasi

Dalam membangun aplikasi berbasis database, biasakan untuk memeriksa konfigurasi koneksi database sebelum mencari kesalahan pada bagian kode lainnya. Pastikan nama database, tabel, dan kolom yang digunakan dalam program sesuai dengan struktur yang telah dibuat. Perbedaan penamaan yang sederhana dapat menyebabkan proses pengolahan data tidak berjalan sebagaimana mestinya.

Selain itu, gunakan struktur kode yang konsisten dan hindari menempatkan seluruh proses aplikasi pada satu bagian. Pisahkan proses pengolahan data melalui Model, pengaturan

alur aplikasi melalui Controller, dan tampilan melalui View. Dengan menerapkan pemisahan tersebut sejak awal, aplikasi akan lebih mudah diperiksa ketika terjadi kesalahan dan lebih mudah dikembangkan pada tahap berikutnya.

Dalam pembuatan form menggunakan Bootstrap 5, perhatikan pula konsistensi penamaan *input*, *label*, dan elemen form lainnya. Struktur form yang rapi akan membantu proses pengolahan data serta memudahkan pembaca ketika melakukan pengembangan dan perubahan pada aplikasi. Biasakan menguji form setelah selesai dibuat dengan memasukkan data yang sesuai dengan kebutuhan aplikasi.

Persiapan Menuju Bab Berikutnya

Setelah menyelesaikan bab ini, pastikan database telah berhasil dibuat dan aplikasi CodeIgniter 4 dapat terhubung dengan database tanpa mengalami kendala. Pastikan pula form input dapat ditampilkan melalui browser dan proses pengiriman data telah berjalan sesuai dengan langkah praktikum. Jika seluruh tahapan tersebut telah berhasil, berarti aplikasi telah memiliki dasar yang diperlukan untuk melanjutkan proses pengembangan berikutnya.

Pada bab selanjutnya, pembahasan akan berfokus pada validasi data serta implementasi AJAX menggunakan jQuery. Kemampuan yang telah diperoleh pada bab ini, khususnya mengenai form, Controller, Model, dan database, akan digunakan kembali dalam proses tersebut. Oleh karena itu, pembaca diharapkan memahami dengan baik alur pengolahan data yang telah dipraktikkan agar dapat mengikuti tahapan berikutnya dengan lebih mudah.

BAB 4
VALIDASI DAN IMPLEMENTASI AJAX JQUERY

Setelah aplikasi berhasil terhubung dengan basis data dan memiliki halaman input yang berfungsi dengan baik, tahapan berikutnya adalah meningkatkan kualitas proses pengolahan data melalui penerapan validasi serta penggunaan teknologi AJAX dan jQuery. Kedua komponen tersebut memiliki peranan penting dalam membangun aplikasi web yang lebih interaktif, responsif, dan mampu memberikan pengalaman penggunaan yang lebih baik. Pada aplikasi modern, proses pengolahan data tidak hanya berfokus pada keberhasilan penyimpanan informasi, tetapi juga memperhatikan bagaimana aplikasi memberikan umpan balik kepada pengguna secara cepat dan akurat.

Validasi data merupakan mekanisme yang digunakan untuk memastikan bahwa setiap data yang dimasukkan oleh pengguna telah memenuhi ketentuan yang ditetapkan sebelum diproses lebih lanjut. Dengan adanya validasi, kesalahan seperti data kosong, format yang tidak sesuai, maupun data yang tidak lengkap dapat dideteksi sejak awal sehingga kualitas data yang disimpan dalam basis data tetap terjaga. Proses validasi juga membantu mengurangi potensi kesalahan yang dapat memengaruhi kinerja aplikasi pada tahap selanjutnya.

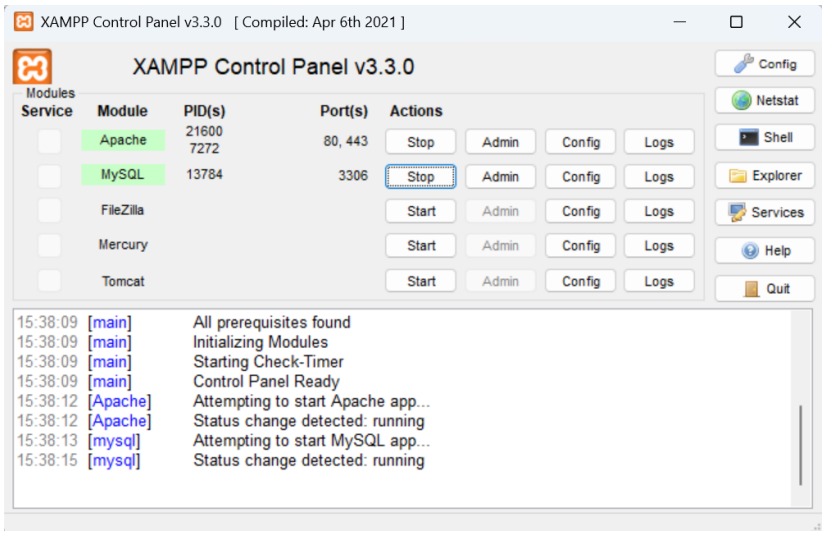
Selain melakukan validasi, aplikasi web juga perlu mampu memberikan respons yang cepat ketika pengguna melakukan suatu tindakan, seperti menyimpan data atau memperbarui informasi. Salah satu teknologi yang banyak digunakan untuk mendukung kebutuhan tersebut adalah Asynchronous JavaScript and XML (AJAX). Melalui AJAX, komunikasi antara browser dan server dapat dilakukan tanpa harus memuat ulang seluruh halaman (*reload*), sehingga proses pengolahan data menjadi lebih efisien dan nyaman bagi pengguna. Walaupun istilah AJAX mengandung kata XML, dalam implementasi saat ini pertukaran data lebih banyak menggunakan format JSON karena lebih sederhana dan mudah diproses.

Untuk mempermudah implementasi AJAX, pada bab ini digunakan jQuery sebagai pustaka JavaScript yang menyediakan berbagai fungsi siap pakai dalam menangani manipulasi halaman web, pengolahan *event*, serta komunikasi dengan server. Penggunaan jQuery memungkinkan penulisan kode menjadi lebih ringkas dibandingkan JavaScript murni, sehingga pembaca dapat lebih fokus memahami alur pengolahan data daripada mempelajari sintaks JavaScript yang lebih kompleks. Meskipun saat ini banyak teknologi baru yang berkembang, jQuery masih banyak digunakan pada berbagai aplikasi web karena kemudahan implementasi dan kompatibilitasnya.

Pada bab ini pembaca akan mempelajari bagaimana menghubungkan form input dengan proses validasi, mengirimkan data menggunakan AJAX, menerima respons dari server, serta menampilkan informasi hasil proses kepada pengguna tanpa harus melakukan pemuatan ulang halaman. Setiap tahapan praktikum disusun secara bertahap agar pembaca dapat memahami hubungan antara antarmuka aplikasi, Controller, Model, dan basis data dalam menghasilkan proses pengolahan data yang berjalan secara dinamis.

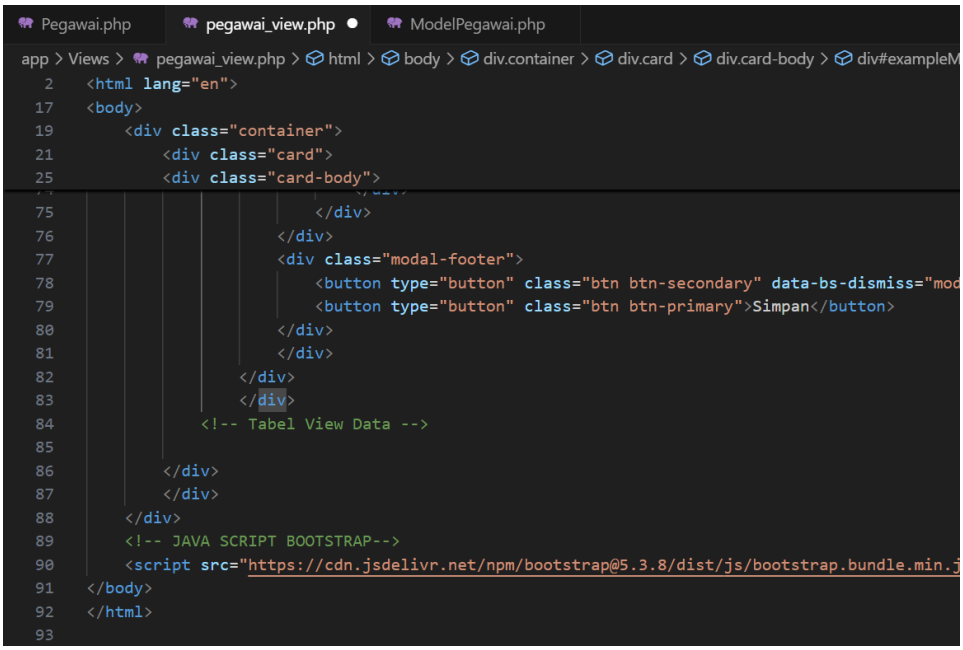
Pemahaman terhadap materi pada bab ini menjadi langkah penting sebelum aplikasi dikembangkan lebih lanjut pada proses pengelolaan data, pencarian, dan implementasi fitur CRUD secara lengkap. Dengan menguasai validasi dan AJAX, pembaca diharapkan mampu membangun aplikasi yang tidak hanya berfungsi dengan baik, tetapi juga memberikan pengalaman penggunaan yang lebih profesional serta sesuai dengan kebutuhan pengembangan aplikasi web modern.

1) Pastikan web server sudah di aktifkan (apache dan mysql)



Gambar 4.1 XAMPP

2) Buka projek CRUD bagian 1 sebelumnya dengan VS Code. Edit coding pegawai_view.php yang ada pada direktori CI4CRUD/app/views. Tambahkan table untuk menampilkan data pada bawah tombol modal



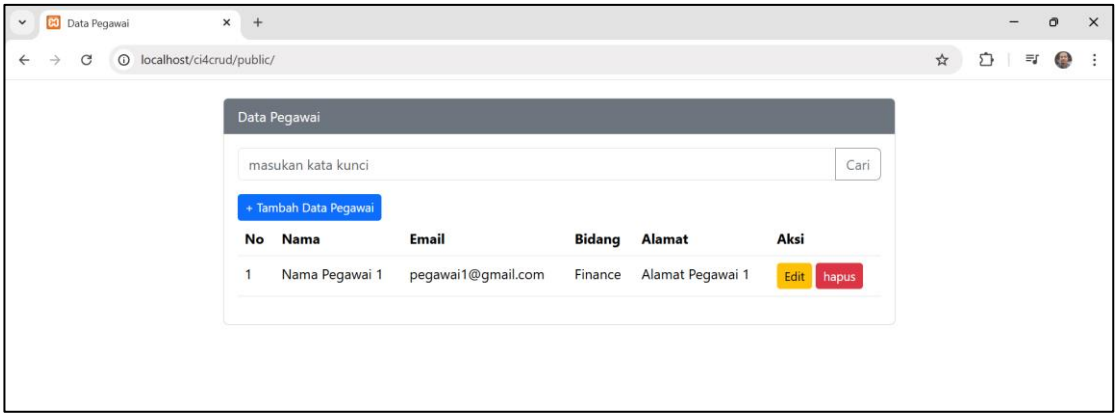
Gambar 4.2 Projek Pegawai_View.PHP

3) Ubah sintaknya menjadi seperti ini

```
Pegawai.php | pegawai_view.php | ModelPegawai.php
app > Views > pegawai_view.php > html > body > div.container > div.card > div.card-body > table.table > tbody
  2   <html lang="en">
 17   <body>
 19       <div class="container">
 21           <div class="card">
 25           <div class="card-body">
 83               </div>
 84               <!-- Tabel View Data -->
 85               <table class="table">
 86                   <thead>
 87                       <tr>
 88                           <th>No</th>
 89                           <th>Nama</th>
 90                           <th>Email</th>
 91                           <th>Bidang</th>
 92                           <th>Alamat</th>
 93                           <th>Aksi</th>
 94                       </tr>
 95                   </thead>
 96                   <tbody>
 97                       <tr>
 98                           <td>1</td>
 99                           <td>Nama Pegawai 1</td>
100                           <td>pegawai1@gmail.com</td>
101                           <td>Finance</td>
102                           <td>Alamat Pegawai 1</td>
103                           <td>
104                               <button type="button" class="btn btn-warning btn-sm">Edit</button>
105                               <button type="button" class="btn btn-danger btn-sm">hapus</button>
106                           </td>
107                       </tr>
108                   </tbody>
109               </table>
```

Gambar 4.3 Perbarui Sintak Pegawai_View

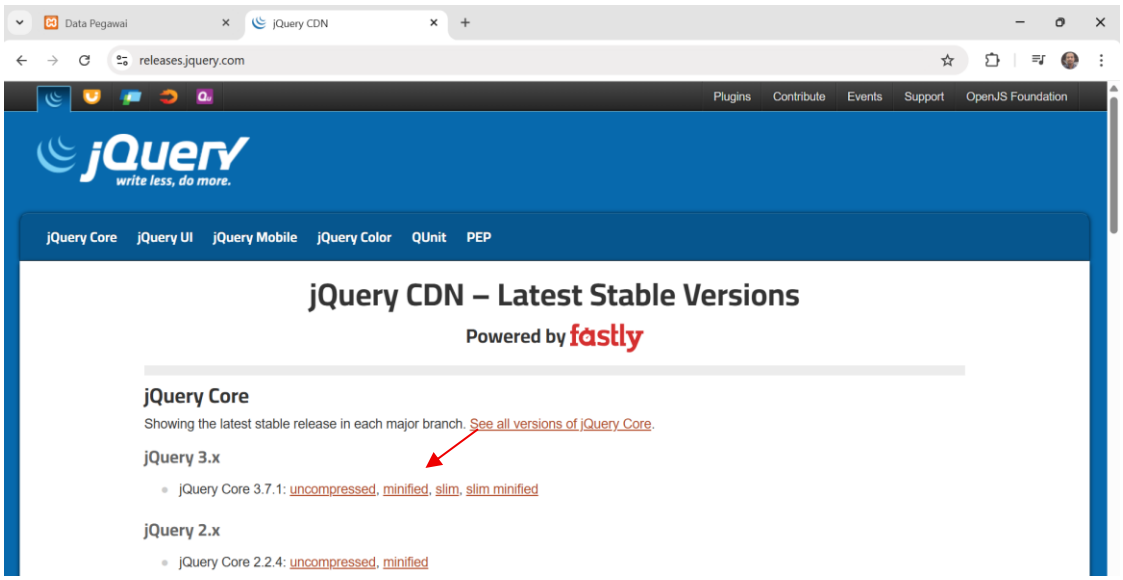
4) Lihat hasilnya pada browser



Gambar 4.4 Hasil Pegawai_View

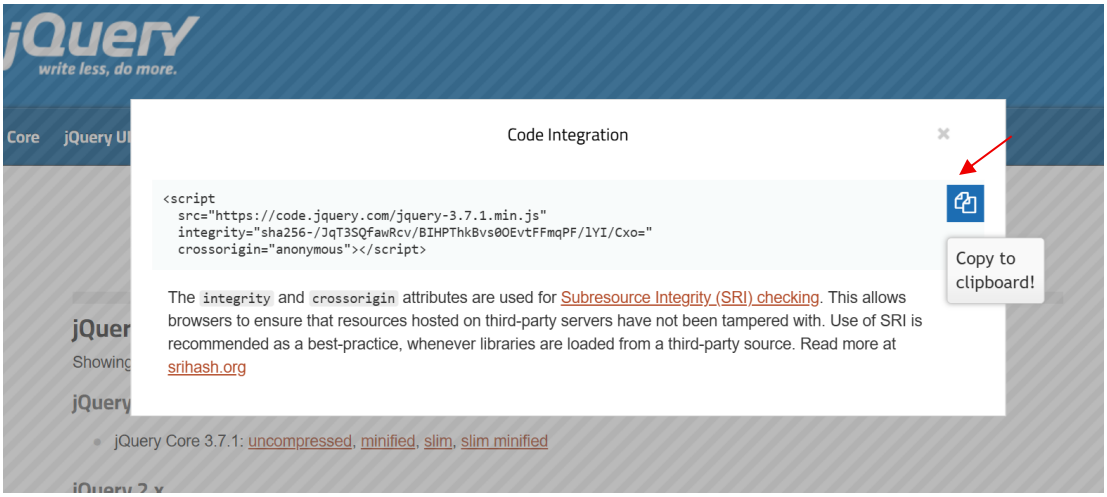
5) Fungsikan tombol simpan pada tambah data pegawai menggunakan ajax jquery. **AJAX** (*Asynchronous JavaScript and XML*) adalah teknik untuk bertukar data dengan server secara asinkron tanpa memuat ulang seluruh halaman web, sementara **jQuery** adalah

pustaka JavaScript yang menyederhanakan berbagai tugas pengembangan web, termasuk penggunaan AJAX. Singkatnya, AJAX adalah metodenya, dan jQuery adalah alat yang memudahkan implementasinya. Untuk menggunakan fungsi jQuery ini silahkan cari jQuery CDN pada browser atau kunjungi link https://releases.jquery.com/ lalu pilih jQuery core => jQuery 3.x pilih yang minified



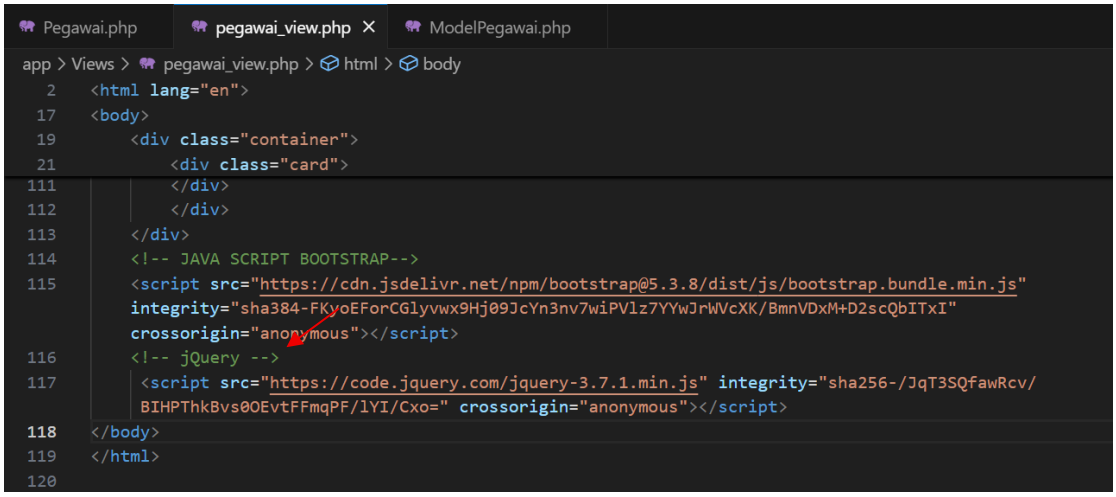
Gambar 4.5 Download jQuery

6) Copy coding tersebut



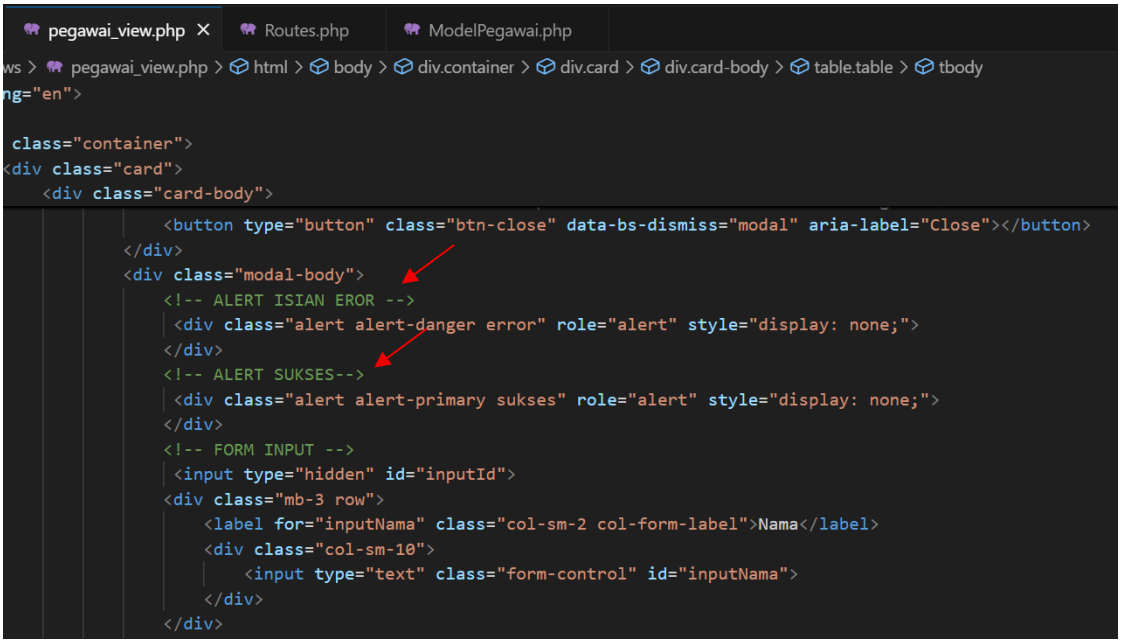
Gambar 4.6 Copy Script jQuery

7) Paste pada projek pegawai_view.php pada bawah javascript bootstrap



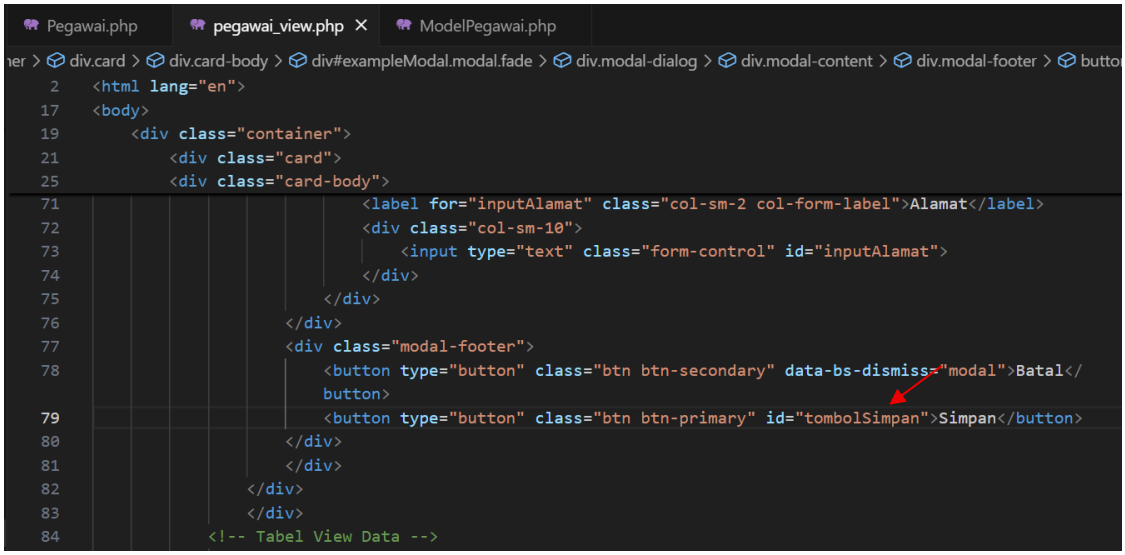
Gambar 4.7 Lokasi Paste Scrip jQuery

8) Tambahkan pesan eror dan sukses pada atas form sebagai notifikasi jika data berhasil / gagal disimpan



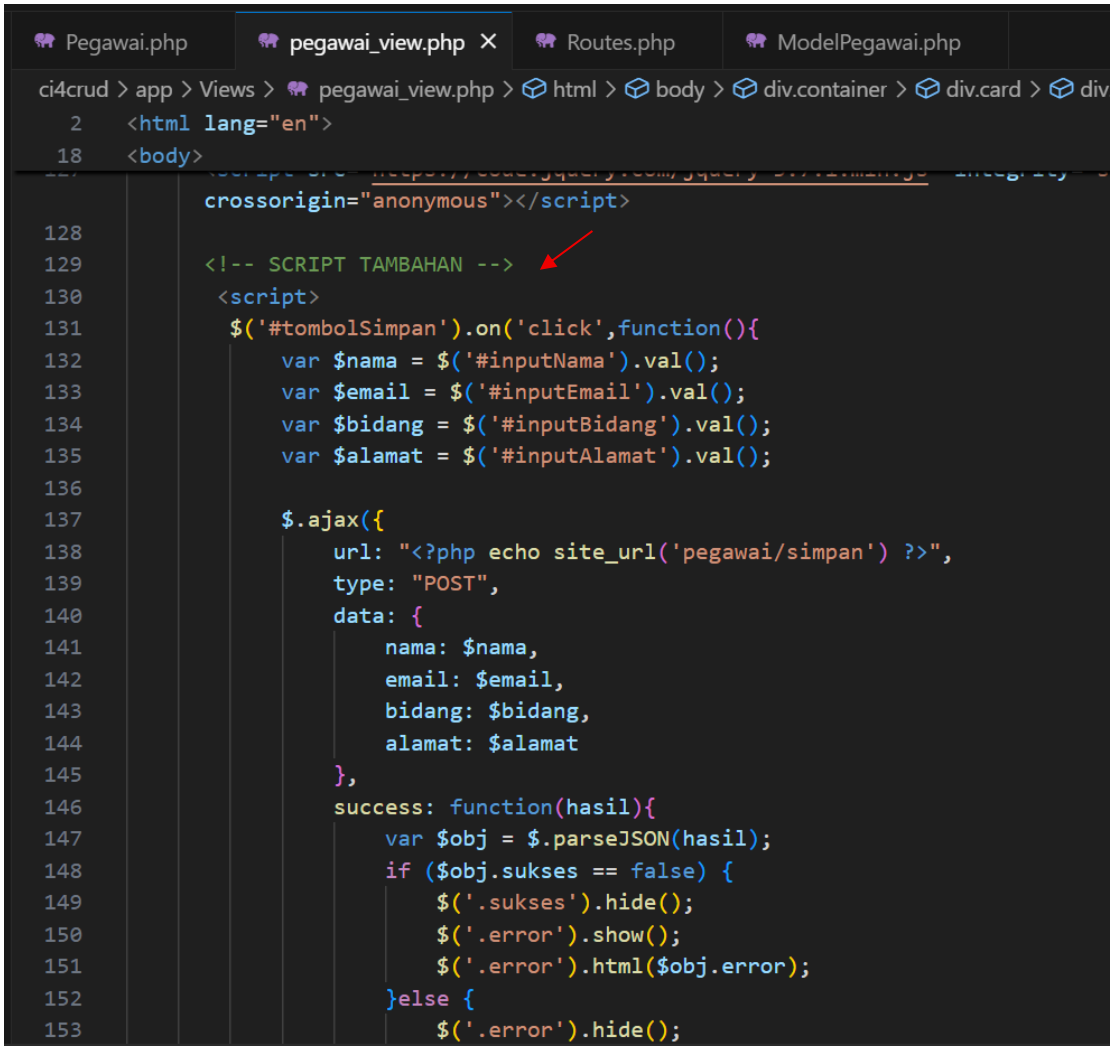
Gambar 4.8 Tambah Pesan Eror dan Sukses

9) Tambahkan id=tombolSimpan pada tombol simpan modal sebagai syarat untuk mengolah dengan java script



Gambar 4.9 Tambah ID Tombol Simpan

10) Buat perintah java script seperti dibawah ini pada pegawai_view.php pada bawahnya jQuery



Gambar 5.10 Script Tambahan jQuery

```
                    $('.sukses').show();
                    $('.sukses').html($obj.sukses);
                }
            }
        });
    });
</script>
</body>
</html>
```

Gambar 4.11 Lanjutan Cript Tambahan jQuery

11) Buat tambahan perintah pada routes.php pada direktori CI4CRUD/app/config/routers.php

Pegawai.php | pegawai_view.php | Routes.php | ModelPegawai.php

app > Config > Routes.php

```php
<?php

use CodeIgniter\Router\RouteCollection;

/**
 * @var RouteCollection $routes
 */
$routes->get('/', 'Pegawai::index');
$routes->post('/pegawai/simpan', 'Pegawai::simpan');
$routes->setAutoRoute(true);

```

Gambar 4.12 Update Perintah Routes

12) Pada controller Pegawai.php buat function simpan

Pegawai.php | pegawai_view.php | Routes.php | ModelPegawai.php

ci4crud > app > Controllers > Pegawai.php > Pegawai > simpan

```php
<?php

namespace App\Controllers;
use App\Models\ModelPegawai;

class Pegawai extends BaseController
{
    protected $model;
    function __construct()
    {
        $this->model = new \App\Models\ModelPegawai();
    }

    public function simpan()
    {
        $validasi = \Config\Services::validation();
        $aturan = [
            'nama' => [
                'label' => 'Nama',
                'rules' => 'required|min_length[5]',
                'errors' => [
                    'required' => '{field} harus diisi',
                    'min_length' => 'Minimun karakter untuk field {field} adalah 5 karakter'
```

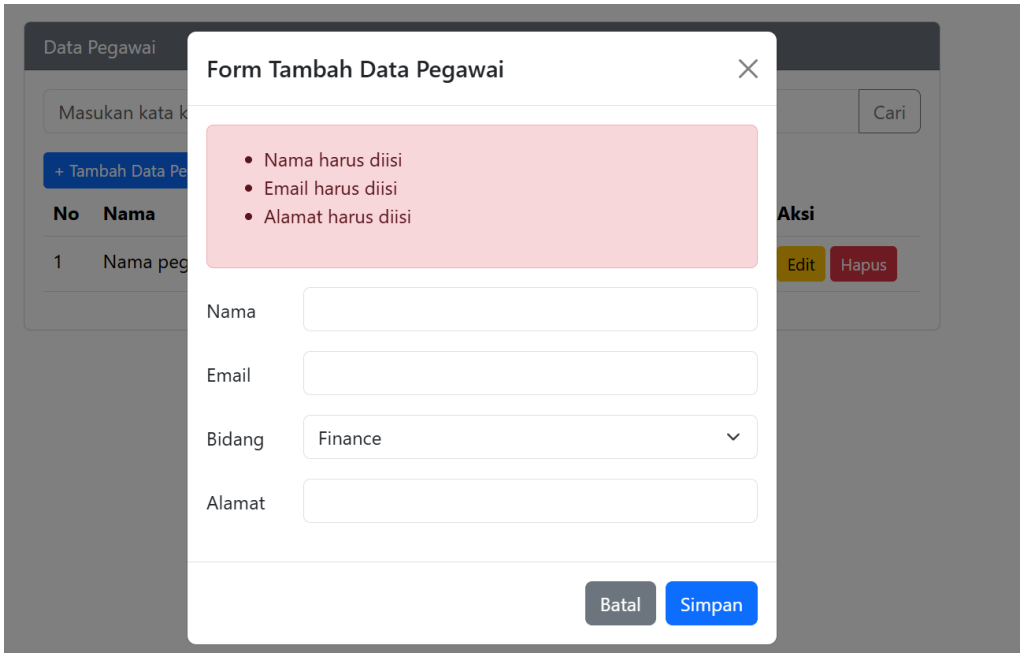
```
            ]
        ],
        'email' => [
            'label' => 'Email',
            'rules' => 'required|min_length[5]|valid_email',
            'errors' => [
                'required' => '{field} harus diisi',
                'min_length' => 'Minimun karakter untuk field {field} adalah 5 karakter',
                'valid_email' => 'Email yang Anda masukan tidak valid'
            ]
        ],
        'alamat' => [
            'label' => 'Alamat',
            'rules' => 'required|min_length[5]',
            'errors' => [
                'required' => '{field} harus diisi',
                'min_length' => 'Minimun karakter untuk field {field} adalah 5 karakter'
            ]
        ],
```

Gambar 4.13 Tambah Function Simpan pada Controller Pegawai

```
        ];
        $validasi->setRules($aturan);
        if($validasi->withRequest($this->request)->run()){
            $nama = $this->request->getPost('nama');
            $email = $this->request->getPost('email');
            $bidang = $this->request->getPost('bidang');
            $alamat = $this->request->getPost('alamat');

            $hasil['sukses'] = "Berhasil memasukan data";
            $hasil['error'] = true;
        }else{
            $hasil['sukses'] = false;
            $hasil['error'] = $validasi->listErrors();
        }

        return json_encode($hasil);
    }
    public function index(): string
    {
        return view('pegawai_view');
    }
}
```

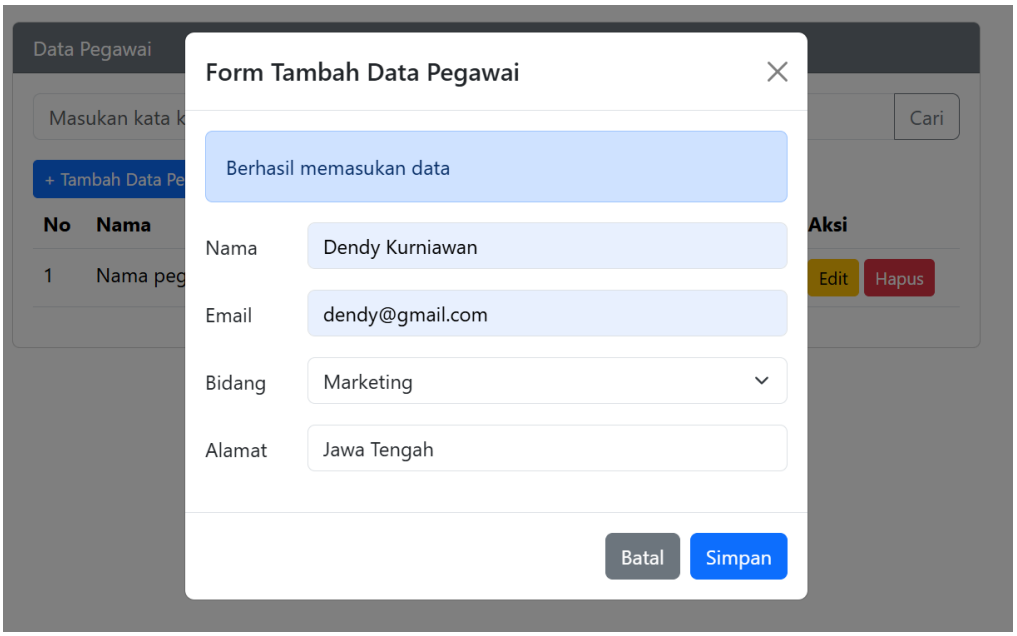
Gambar 4.14 Lanjutan Perintah Function pada Controller Pegawai

13) Coba menginputkan data yang tidak lengkap



Gambar 4.15 Hasil Form Pegawai

14) Coba menginputkan data yang lengkap dan sesuai dan klik simpan



Gambar 4.16 Form Pegawai Saat Diisi Data

Note: Sampai tahapan ini baru muncul notifikasi "Berhasil memasukan data" namun data sesungguhnya belum tersimpan di database (cek dan refresh halaman phpmyadmin => table pegawai), belum ada data yang ditambahkan.

TUGAS:

Modifikasi function simpan pada controller Pegawai.php agar mengirimkan data ke modelPegawai.php supaya data bisa tersimpan di database.

Rangkuman Bab

Pada bab ini telah dibahas proses validasi data serta implementasi teknologi AJAX menggunakan jQuery dalam aplikasi berbasis CodeIgniter 4. Kedua komponen tersebut merupakan bagian penting dalam meningkatkan kualitas aplikasi karena mampu memastikan data yang diproses sesuai dengan ketentuan yang telah ditetapkan serta memberikan respons yang lebih cepat kepada pengguna tanpa harus melakukan pemuatan ulang halaman.

Pembaca telah mempelajari bagaimana menerapkan validasi pada form input, menghubungkan proses pengiriman data menggunakan AJAX, menerima respons dari Controller, serta menampilkan hasil proses kepada pengguna secara dinamis. Seluruh tahapan tersebut menunjukkan bahwa proses pengolahan data tidak hanya bergantung pada penyimpanan informasi ke dalam basis data, tetapi juga memerlukan mekanisme komunikasi yang baik antara browser dan server agar aplikasi dapat berjalan secara optimal.

Penerapan AJAX dan jQuery memberikan pengalaman baru dalam membangun aplikasi web yang lebih interaktif. Pengguna tidak perlu lagi menunggu halaman dimuat kembali setiap kali melakukan penyimpanan atau pengolahan data, sehingga proses penggunaan aplikasi menjadi lebih efisien dan nyaman. Selain meningkatkan kenyamanan pengguna, pendekatan ini juga membantu pengembang dalam membangun aplikasi yang memiliki tampilan lebih responsif sesuai dengan kebutuhan aplikasi web modern.

Melalui kegiatan praktikum pada bab ini, pembaca juga memperoleh pemahaman mengenai pentingnya pengujian terhadap setiap proses pengolahan data. Validasi yang baik akan mengurangi kemungkinan terjadinya kesalahan data, sedangkan penggunaan AJAX membantu memastikan bahwa komunikasi antara aplikasi dan server berlangsung secara efektif. Kombinasi kedua teknologi tersebut menjadi salah satu fondasi penting dalam membangun aplikasi yang stabil, mudah digunakan, dan memiliki kualitas yang baik.

Selain aspek teknis, pembaca diharapkan mulai memahami bahwa kualitas sebuah aplikasi tidak hanya ditentukan oleh banyaknya fitur yang dimiliki, tetapi juga oleh bagaimana aplikasi mampu memberikan pengalaman penggunaan yang nyaman, cepat, dan mudah dipahami. Oleh karena itu, penerapan validasi serta komunikasi data secara asinkron menggunakan AJAX merupakan salah satu praktik yang perlu dibiasakan dalam setiap proses pengembangan aplikasi berbasis web.

Pada bab berikutnya, pembaca akan mempelajari proses menampilkan data dari basis data, menerapkan pagination, serta membangun fitur pencarian data. Seluruh kemampuan yang telah diperoleh pada bab ini akan menjadi dasar dalam mengembangkan aplikasi CRUD yang lebih lengkap sehingga aplikasi mampu mengelola data secara lebih efektif, efisien, dan mudah digunakan.

Refleksi Pembelajaran

Setelah menyelesaikan bab ini, pembaca diharapkan memahami bahwa sebuah aplikasi web tidak cukup hanya mampu menerima dan menyimpan data, tetapi juga harus mampu memastikan bahwa data yang diberikan pengguna sesuai dengan kebutuhan aplikasi. Penerapan validasi menjadi bagian penting dalam proses tersebut karena dapat membantu mendeteksi kesalahan input sebelum data diproses lebih lanjut. Dengan memahami fungsi

validasi, pembaca dapat melihat bahwa kualitas aplikasi juga dipengaruhi oleh kualitas data yang diterima dan diproses oleh sistem.

Pembaca juga perlu memahami perbedaan antara proses pengolahan data secara biasa dengan proses yang menggunakan AJAX. Melalui praktikum pada bab ini, pembaca dapat mengamati bahwa AJAX memungkinkan komunikasi antara halaman aplikasi dengan server dilakukan tanpa harus memuat ulang keseluruhan halaman. Penggunaan jQuery membantu menyederhanakan proses tersebut sehingga pembaca dapat memahami alur komunikasi antara antarmuka pengguna dan server secara lebih mudah. Untuk memperkuat pemahaman, pembaca disarankan mencoba kembali setiap tahapan praktikum secara mandiri dan memperhatikan perubahan yang terjadi ketika proses validasi maupun pengiriman data dijalankan.

Penerapan dalam Dunia Nyata

Validasi data merupakan bagian yang hampir selalu ditemukan dalam berbagai aplikasi berbasis web. Form pendaftaran, sistem akademik, aplikasi penjualan, sistem administrasi, maupun aplikasi pengelolaan data lainnya membutuhkan mekanisme untuk memastikan bahwa informasi yang dimasukkan pengguna sesuai dengan ketentuan yang diperlukan. Dengan demikian, kemampuan menerapkan validasi yang dipelajari pada bab ini dapat digunakan pada berbagai jenis aplikasi dan studi kasus yang akan dikembangkan pembaca pada tahap berikutnya.

Penggunaan AJAX juga memberikan gambaran mengenai bagaimana aplikasi web dapat memberikan respons yang lebih interaktif kepada pengguna. Pada aplikasi yang memiliki banyak proses pengolahan data, pengguna akan lebih nyaman apabila setiap tindakan dapat memperoleh respons tanpa harus selalu melakukan pemuatan ulang halaman. Konsep tersebut banyak digunakan dalam aplikasi web yang membutuhkan interaksi secara dinamis antara pengguna dan server. Oleh karena itu, pemahaman mengenai AJAX dan jQuery dapat menjadi salah satu dasar bagi pembaca untuk memahami pola interaksi aplikasi web yang lebih dinamis.

Tips Pengembangan Aplikasi

Dalam menerapkan validasi, biasakan menentukan aturan data yang diperlukan sebelum membuat kode program. Periksa jenis informasi yang harus diisi, bagian yang wajib diisi, serta kondisi yang dapat menyebabkan data dianggap tidak sesuai. Dengan menentukan aturan tersebut sejak awal, proses implementasi validasi akan menjadi lebih terarah dan hasil aplikasi dapat lebih mudah diuji.

Pada penggunaan AJAX dan jQuery, lakukan pengujian secara bertahap setiap kali menambahkan proses baru. Periksa apakah data berhasil dikirim, apakah server dapat menerima permintaan, dan apakah hasil proses dapat diterima kembali oleh halaman aplikasi. Jika terjadi kesalahan, periksa setiap bagian dari alur tersebut secara berurutan sehingga sumber masalah dapat ditemukan dengan lebih mudah.

Biasakan pula memberikan informasi yang jelas kepada pengguna ketika terjadi kesalahan dalam pengisian data maupun ketika proses pengolahan data berhasil dilakukan. Pesan yang mudah dipahami akan membantu pengguna mengetahui tindakan yang harus

dilakukan berikutnya. Dengan demikian, fungsi validasi tidak hanya berperan dalam menjaga kualitas data, tetapi juga mendukung kemudahan penggunaan aplikasi.

Persiapan Menuju Bab Berikutnya

Setelah menyelesaikan bab ini, pastikan proses validasi telah berjalan sesuai dengan kebutuhan form dan implementasi AJAX dapat digunakan untuk melakukan komunikasi antara halaman aplikasi dengan server. Pastikan pula kode jQuery yang digunakan dapat dijalankan dengan baik dan tidak menimbulkan kesalahan pada halaman aplikasi. Pengujian terhadap beberapa kondisi input perlu dilakukan agar pembaca dapat memastikan bahwa aplikasi memberikan respons sesuai dengan kondisi yang dihadapi.

Pada bab berikutnya, pembahasan akan berlanjut pada proses menampilkan data dari database, menerapkan pagination, dan menyediakan fitur pencarian. Kemampuan yang telah diperoleh pada bab ini, khususnya mengenai pengolahan form, komunikasi dengan server, dan pemberian respons kepada pengguna, akan menjadi dasar untuk membangun proses pengelolaan data yang lebih lengkap. Oleh karena itu, pembaca diharapkan memastikan seluruh tahapan pada bab ini telah dipahami dan dapat dijalankan sebelum melanjutkan ke pembahasan berikutnya.

BAB 5
MENAMPILKAN DATA, PAGINATION DAN PENCARIAN

Setelah aplikasi berhasil melakukan proses penyimpanan data ke dalam basis data, tahapan berikutnya adalah menampilkan kembali data tersebut dalam bentuk informasi yang mudah dibaca dan dikelola oleh pengguna. Kemampuan menampilkan data merupakan salah satu fungsi utama dalam sebuah aplikasi berbasis web karena hampir seluruh sistem informasi memerlukan fasilitas untuk melihat, mencari, dan mengelola data yang telah tersimpan. Oleh karena itu, proses penyajian data tidak hanya berfokus pada aspek teknis pengambilan data dari basis data, tetapi juga memperhatikan kemudahan pengguna dalam mengakses informasi yang dibutuhkan.

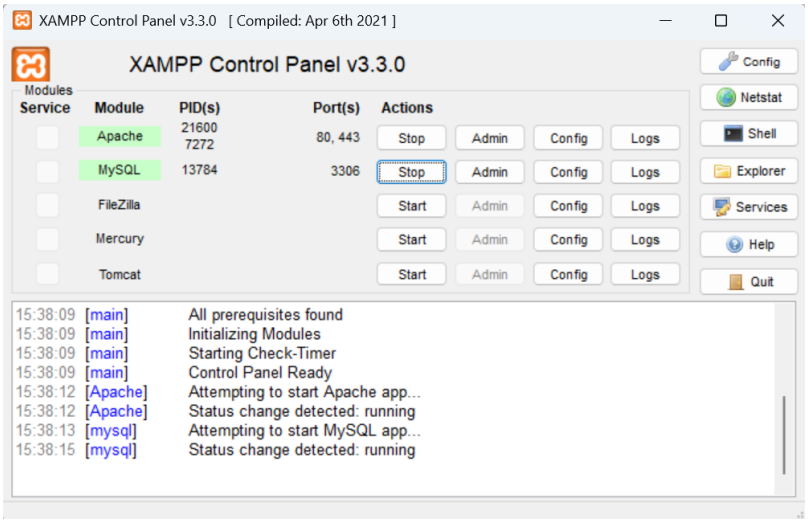
Pada framework CodeIgniter 4, proses menampilkan data dilakukan melalui kerja sama antara Model, Controller, dan View sesuai dengan konsep *Model–View–Controller* (MVC). Model bertugas mengambil data dari basis data, Controller mengolah data yang diterima dari Model, sedangkan View menampilkan informasi tersebut dalam bentuk tabel atau komponen antarmuka lainnya. Pembagian tugas tersebut membuat proses pengembangan aplikasi menjadi lebih terstruktur dan memudahkan pengembang ketika melakukan perubahan maupun pengembangan fitur baru.

Seiring bertambahnya jumlah data yang tersimpan, proses penyajian data juga perlu memperhatikan aspek efisiensi. Menampilkan seluruh data dalam satu halaman dapat menyebabkan halaman menjadi terlalu panjang, memperlambat proses pemuatan, serta menyulitkan pengguna dalam menemukan informasi yang diperlukan. Oleh karena itu, pada bab ini diterapkan teknik **pagination**, yaitu membagi data ke dalam beberapa halaman sehingga informasi dapat ditampilkan secara bertahap. Selain meningkatkan kinerja aplikasi, pagination juga memberikan tampilan yang lebih rapi dan mudah digunakan.

Selain pagination, aplikasi juga memerlukan fasilitas pencarian (*search*) untuk membantu pengguna menemukan data tertentu dengan lebih cepat. Fitur pencarian menjadi salah satu kebutuhan dasar dalam berbagai sistem informasi karena memungkinkan pengguna memperoleh informasi yang diinginkan tanpa harus menelusuri seluruh data secara manual. Implementasi fitur pencarian pada CodeIgniter 4 memanfaatkan kemampuan Model dalam menyusun kondisi pencarian berdasarkan kata kunci yang dimasukkan oleh pengguna, sehingga proses pencarian dapat dilakukan secara efektif dan efisien.

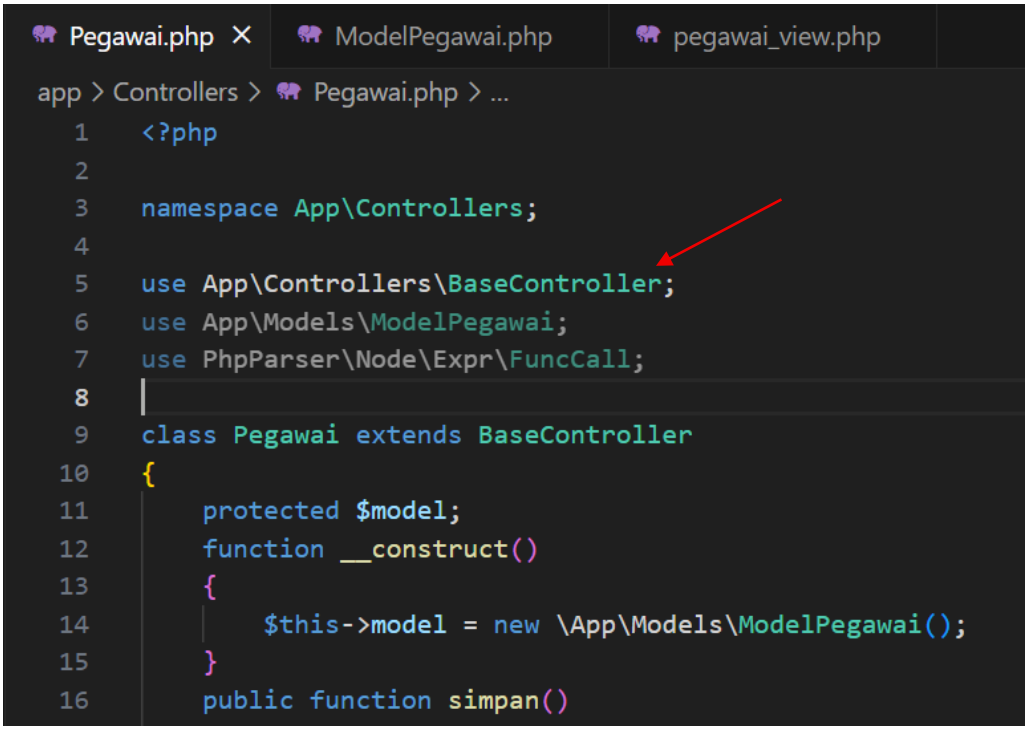
Pada bab ini pembaca akan mempelajari proses pengambilan data dari basis data, menampilkan data dalam bentuk tabel, menerapkan pagination, serta mengembangkan fitur pencarian data. Seluruh tahapan praktikum disusun secara bertahap agar pembaca dapat memahami bagaimana data yang tersimpan pada basis data diproses dan disajikan kembali kepada pengguna melalui antarmuka aplikasi.

1) Pastikan web server sudah di aktifkan (apache dan mysql)



Gambar 5.1 XAMPP

2) Pada bagian 2 sebelumnya terdapat tugas untuk **Modifikasi function simpan pada controller Pegawai.php agar mengirimkan data ke modelPegawai.php supaya data bisa tersimpan di database.** dan berikut Adalah jawabannya: Buka projek CRUD bagian 2 sebelumnya dengan VS Code. Ubah sintak controller Pegawai.php menjadi seperti berikut ini



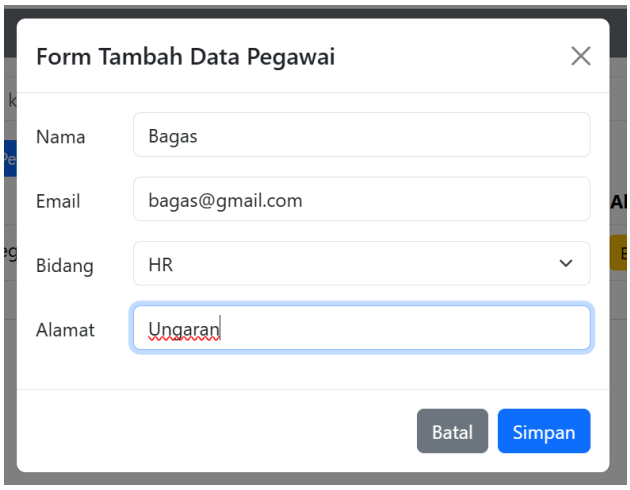
Gambar 5.2 Tambahkan Perintah pada Controller Pegawai

Dan juga pada bagian berikut ini

```php
class Pegawai extends BaseController
    public function simpan()
                ]
                ],
        ];
        $validasi->setRules($aturan);
        if ($validasi->withRequest($this->request)->run()) {
            $data = [
            'nama' => $this->request->getPost('nama'),
            'email' => $this->request->getPost('email'),
            'bidang' => $this->request->getPost('bidang'),
            'alamat' => $this->request->getPost('alamat'),
            ];

            $this->model->save($data);

            $hasil['sukses'] = "Berhasil memasukan data";
            $hasil['error'] = true;
        }else{
            $hasil['sukses'] = false;
            $hasil['error'] = $validasi->listErrors();
        }

        return json_encode($hasil);
    }
    public function index(): string
    {
        return view('pegawai_view');
```

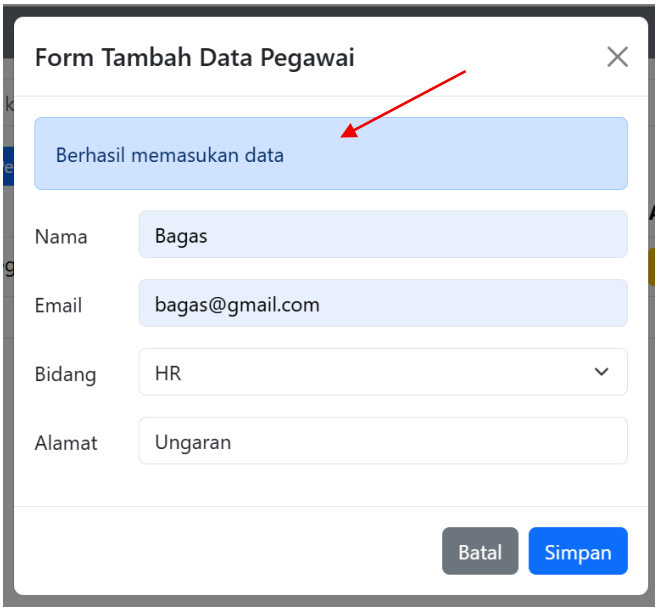
Gambar 5.3 Tambahkan Perintah Lainnya pada Controller Pegawai

3) Silahkan coba untuk menambahkan data di form input



Gambar 5.4 Hasil Format Input Tambah Data Pegawai

4) Klik simpan



Gambar 5.5 Notifikasi Tombol Simpan Data Pegawai

5) Data sudah berhasil disimpan ke daabase.

	id	nama	email	bidang	alamat
Ubah Salin Hapus	1	Dendy Kurniawan	dendy@gmail.com	marketing	Kendal Jawa Tengah
Ubah Salin Hapus	2	Antoni	antoni@gmail.com	marketing	Semarang
Ubah Salin Hapus	3	Bagas	bagas@gmail.com	hr	Ungaran

Gambar 5.6 Database Pegawai

6) Pada sampai tahapan ini data belum ditampilkan pada table data pegawai. Dan pada form input data pegawai meskipun sudah berhasil menginputkan data, namun pada form isian data pegawai masih terdapat tulisan data yang diinput sebelumnya. Silahkan atur supaya data tersebut menjadi hilang atau kosong seperti belum terisi. Buka file pegawai_view.php yang ada pada direktori view data tambahkan class tombol-tutup di tombol close dan batal pada modal

```
39          <!-- Modal -->
40          <div class="modal fade" id="exampleModal" tabindex="-1"
            aria-labelledby="exampleModalLabel" aria-hidden="true">
41          <div class="modal-dialog">
42              <div class="modal-content">
43              <div class="modal-header">
44                  <h1 class="modal-title fs-5" id="exampleModalLabel">Form Tambah Data
                    Pegawai</h1>
45                  <button type="button" class="btn-close tombol-tutup"
                    data-bs-dismiss="modal" aria-label="Close"></button>
46              </div>
47              <div class="modal-body">
```

Gambar 5.7 Update Pegawai View

```
83                  </div>
84                  <div class="modal-footer">
85                      <button type="button" class="btn btn-secondary tombol-tutup"
                        data-bs-dismiss="modal">Batal</button>
86                      <button type="button" class="btn btn-primary" id="tombolSimpan">Simpan</
                        button>
87                  </div>
88                  </div>
```

Gambar 5.8 Update Pegawai View Baris 85

7) Masih didalam file pegawai_view.php, buat function baru pada script untuk membuat perintah menghapus isian form tersebut

```
128     <!-- SCRIPT TAMBAHAN -->
129      <script>
130      function bersihkan(){
131       $('#inputNama').val('');
132       $('#inputEmail').val('');
133       $('#inputAlamat').val('');
134      }
135      $('.tombol-tutup').on('click',function(){
136       if($('.sukses').is(":visible")){
137           window.location.href = "<?php echo current_url(). "?" .$_SERVER['QUERY_STRING']
138           ?>";
139       }
140       $('.alert').hide();
141       bersihkan();
142      });
143       $('#tombolSimpan').on('click',function(){
```

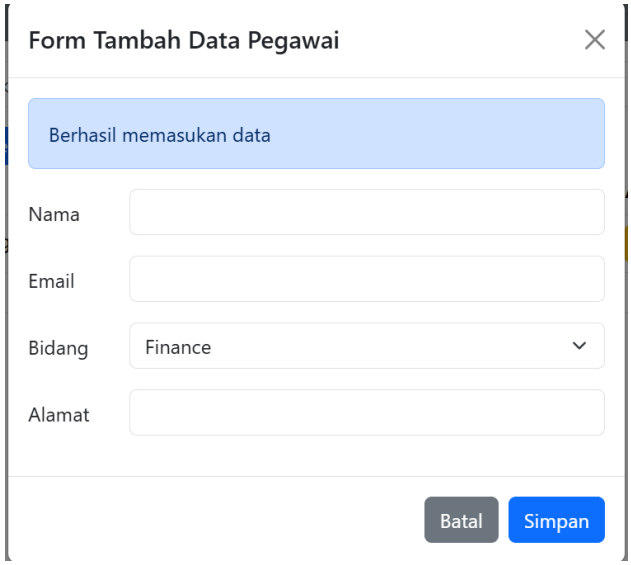
Gambar 5.9 Tambahkan Function dalam Scipt Pegawai View

8) Jalankan function bersihkan pada saat data berhasil disimpan juga

```
162                         $('.error').show();
163                         $('.error').html($obj.error);
164                     }else {
165                         $('.error').hide();
166                         $('.sukses').show();
167                         $('.sukses').html($obj.sukses);
168                         bersihkan();
169                     }
170                 }
171             });
```

Gambar 5.10 Jalankan Function

9) Silahkan coba masukan data baru dan klik simpan. Hanya aka nada notifikasi berhasil disimpan dan data yang telah dimasukan telah hilang pada fomr input data



Gambar 5.11 Hasil Form Data Pegawai

10) Silahkan tambahkan data baru menggunakan form input data minimal 10 data dan cek pada database nya

	id	nama	email	bidang	alamat
Ubah Salin Hapus	1	Dendy Kurniawan	dendy@gmail.com	marketing	Kendal Jawa Tengah
Ubah Salin Hapus	2	Antoni	antoni@gmail.com	marketing	Semarang
Ubah Salin Hapus	3	Bagas	bagas@gmail.com	hr	Ungaran
Ubah Salin Hapus	4	Carla	carla@gmail.com	finance	Siliwangi
Ubah Salin Hapus	5	Deviana	devi@gmail.com	marketing	Surabaya
Ubah Salin Hapus	6	Eko Patrio	eko@gmail.com	marketing	Semarang
Ubah Salin Hapus	7	Farida	farida@gmail.com	hr	Surakarta
Ubah Salin Hapus	8	Gabriel	gabriel@gmail.com	hr	Kendal
Ubah Salin Hapus	9	Haryanto	haryanto@gmail.com	finance	Ungaran
Ubah Salin Hapus	10	Ihsan	ihsan@gmail.com	hr	Weleri
Ubah Salin Hapus	11	Jesika	jesika@gmail.com	hr	Batang

Gambar 5.12 Penambahan Data Baru dalam Database

11) Tampilkan data tersebut di table tampil data pegawai. Buka controller Pegawai.php berikan perintah pada function index seperti berikut

```
Pegawai.php X | ModelPegawai.php | pegawai_view.php
app > Controllers > Pegawai.php > Pegawai
 9   class Pegawai extends BaseController
17       public function simpan()
61               $hasil['sukses'] = false;
62               $hasil['error'] = $validasi->listErrors();
63           }
64           return json_encode($hasil);
65       }
66       public function index(): string
67       {
68           $data['dataPegawai']= $this->model->orderBy('id','desc')->findAll();
69           return view('pegawai_view', $data);
70       }
71   }
```

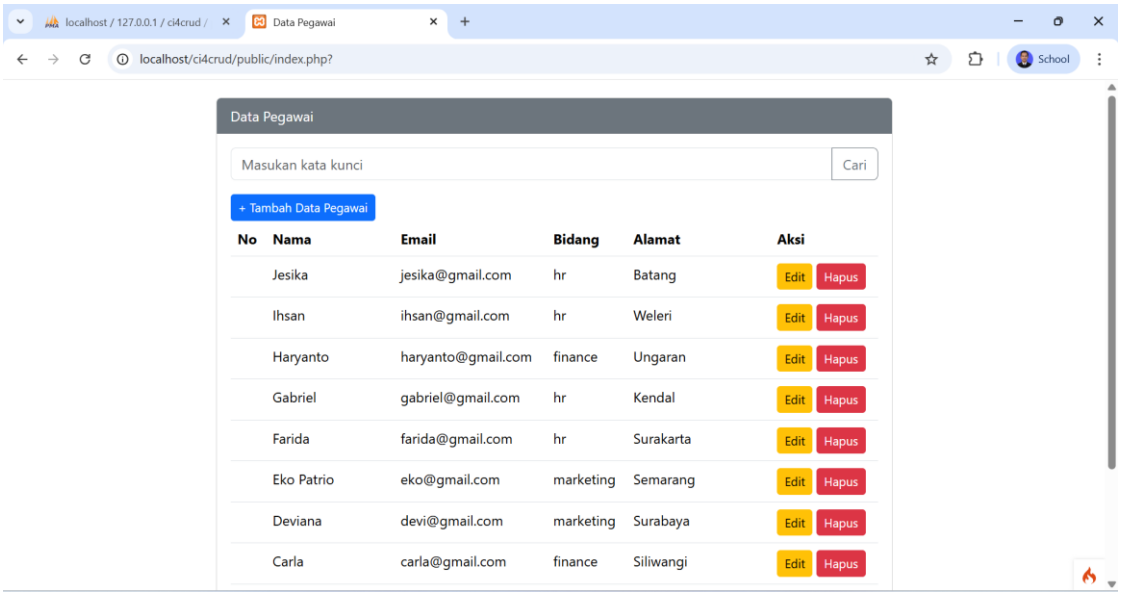
Gambar 5.13 Update Kontroler Pegawai

12) Pada pegawai_view.php rubah perintahnya menjadi seperti berikut

```
Pegawai.php | ModelPegawai.php | pegawai_view.php ●
app > Views > pegawai_view.php > html > body > div.container > div.card > div.card-body
 2   <html lang="en">
18   <body>
20       <div class="container">
22           <div class="card">
26               <div class="card-body">
102                  </thead>
103                  <tbody>
104                      <?php
105                      foreach($dataPegawai as $k => $v){
106                      ?>
107                      <tr>
108                          <td></td>
109                          <td><?=$v['nama'] ?></td>
110                          <td><?=$v['email'] ?></td>
111                          <td><?=$v['bidang'] ?></td>
112                          <td><?=$v['alamat'] ?></td>
113                          <td>
114                              <button type="button" class="btn btn-warning btn-sm">Edit</button>
115                              <button type="button" class="btn btn-danger btn-sm">Hapus</button>
116                          </td>
117                      </tr>
118                      <?php } ?>
119                  </tbody>
120
```

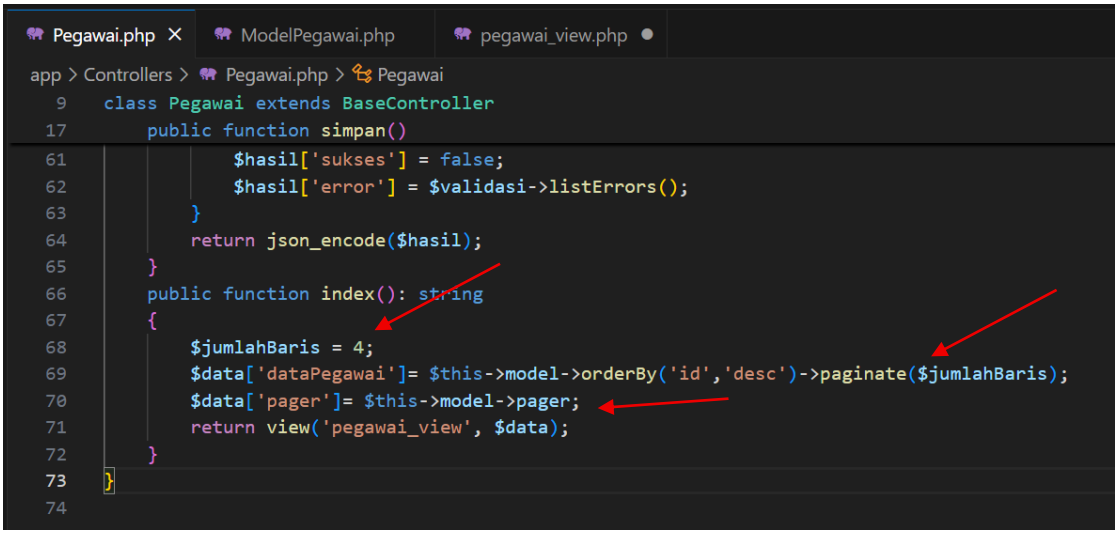
Gambar 5.14 Update pada View Pegawai

13) Silahkan cek dulu halaman browsernya

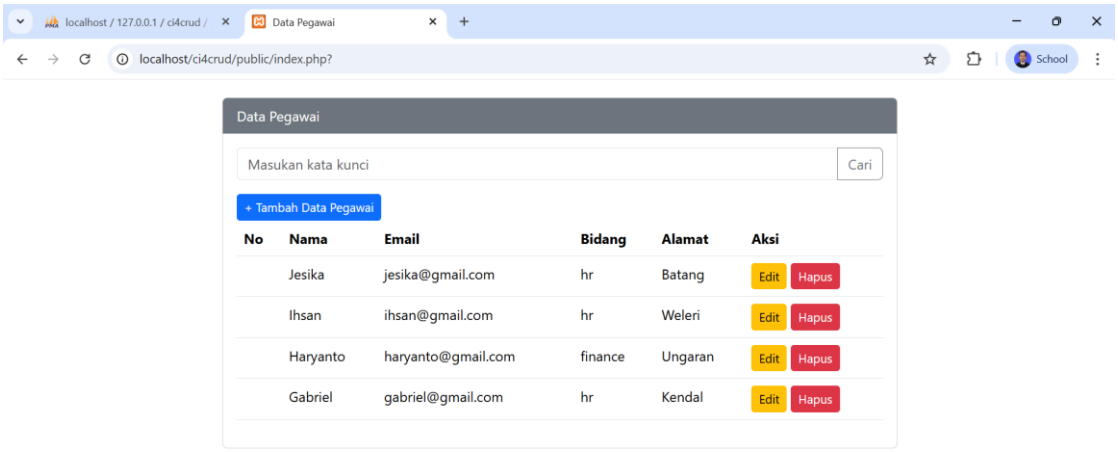


Gambar 5.15 Hasil Tampilan Data Pegawai

14) Semua data akan ditampilan disini. Namun jika data terdapat didatabase berjumlah ratusan data, maka akan ditampilkan semuanya, hal ini akan membnuat kurang efektif dan terlalu berat saat dibuka. Maka dari itu buatkan pagination. Pagination adalah teknik untuk membagi konten digital yang besar menjadi halaman-halaman terpisah yang lebih kecil dan mudah dinavigasi.
15) Rubah koding pada controller Pegawai.php yang awalnya findAll menjadi paginate. Lalu lihat hasil pada browser, seharusnya data yang tampil sudah sesuai dengan jumlah nilai yang di inputkan pada jumlah Baris.

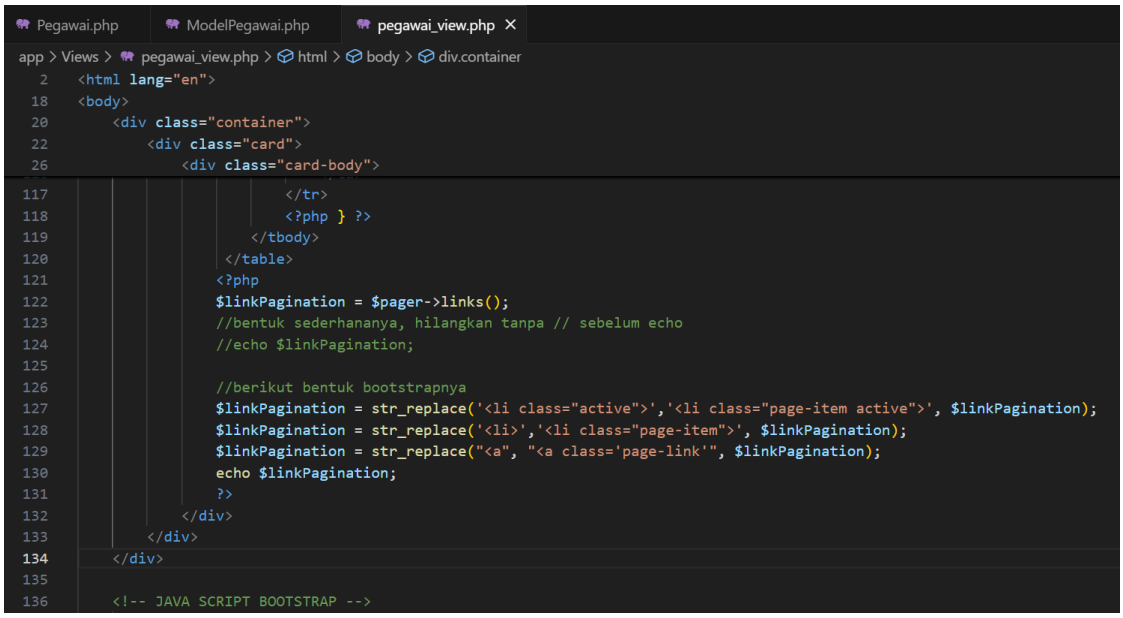


Gambar 5.16 Buat Paginasi pada Kontroler Pegawai



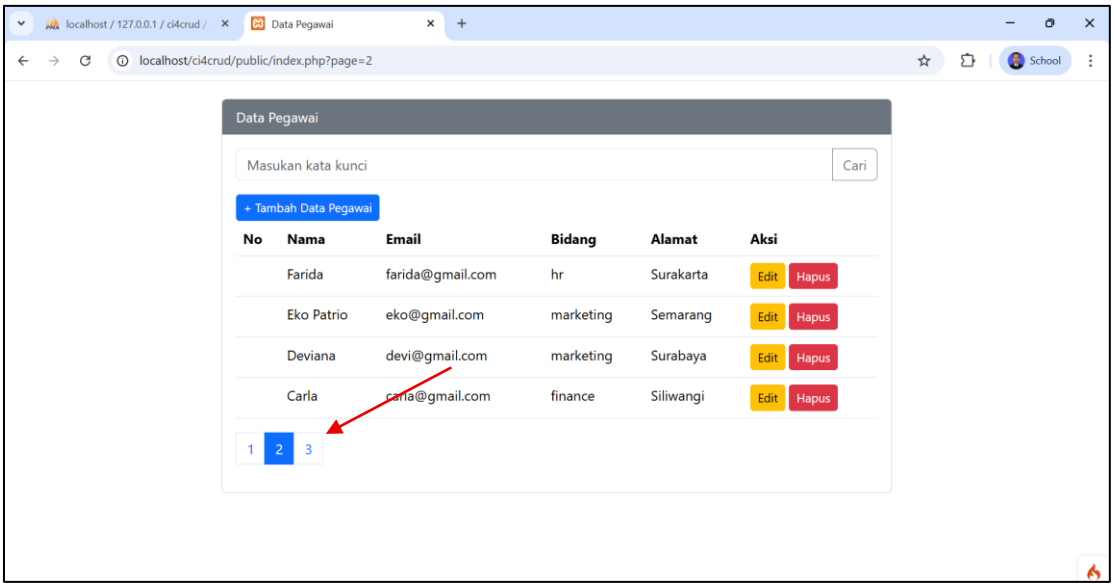
Gambar 5.17 Hasil Tampilan Setelah di Ubah Pagination

16) Tambhakan link unutk melihat halaman beikutnya. Buka pegawai_view.php dan buat perintah sepetti berikut di bagian bawah table



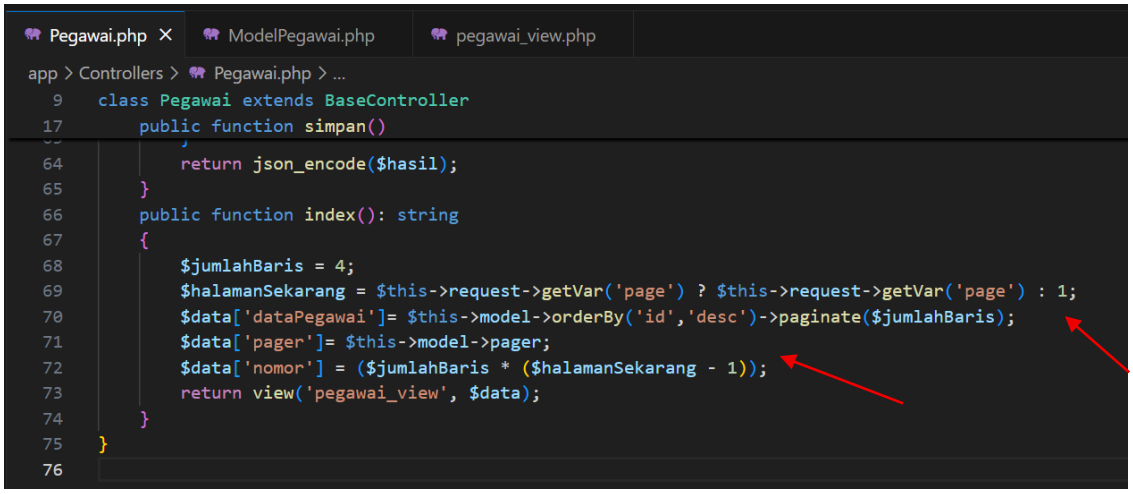
Gambar 5.18 Tambahkan Link Halaman Berikutnya

17) Maka akan tampil seperti berikut ini



Gambar 5.19 Hasil Link Halaman Berikutnya

18) Berikan nomor urut pada setiap data. Nomor ini akan ditampilkan urut sesuai jumlah data, bukan berdasarkan data yang ditampilkan berdasarkan pagination. Buka controller Pegawai.php dan berikan data nomor seperti berikut



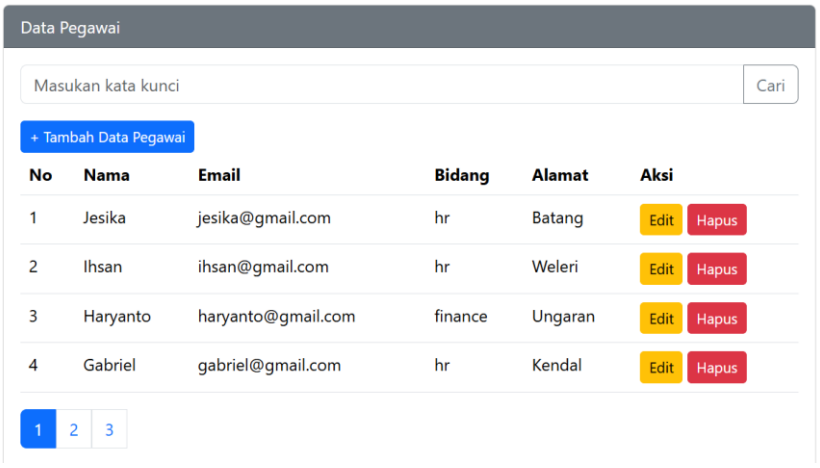
Gambar 5.20 Pemberian Nomor Urut Data

19) Pada bagian pegawai_view.php buat perintah untuk menampilkan nomor dengan increment

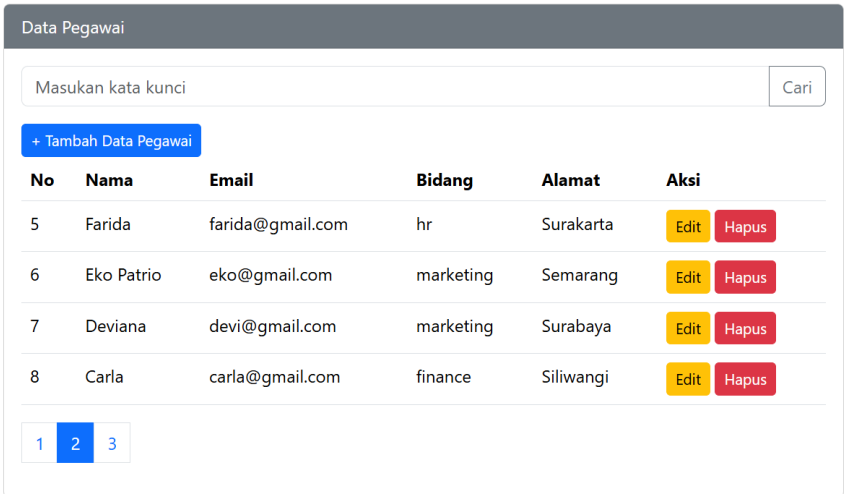
```
103                 <tbody>
104                     <?php
105                     foreach($dataPegawai as $k => $v){
106                     ?>
107                     <tr>
108                         <td><?= ++$nomor; ?></td>
109                         <td><?=$v['nama'] ?></td>
110                         <td><?=$v['email'] ?></td>
111                         <td><?=$v['bidang'] ?></td>
112                         <td><?=$v['alamat'] ?></td>
113                         <td>
```

Gambar 5.21 Perintah Nomor Urut

20) Hasilnya akan seperti berikut ini



Gambar 5.22 Hasil Halaman 1

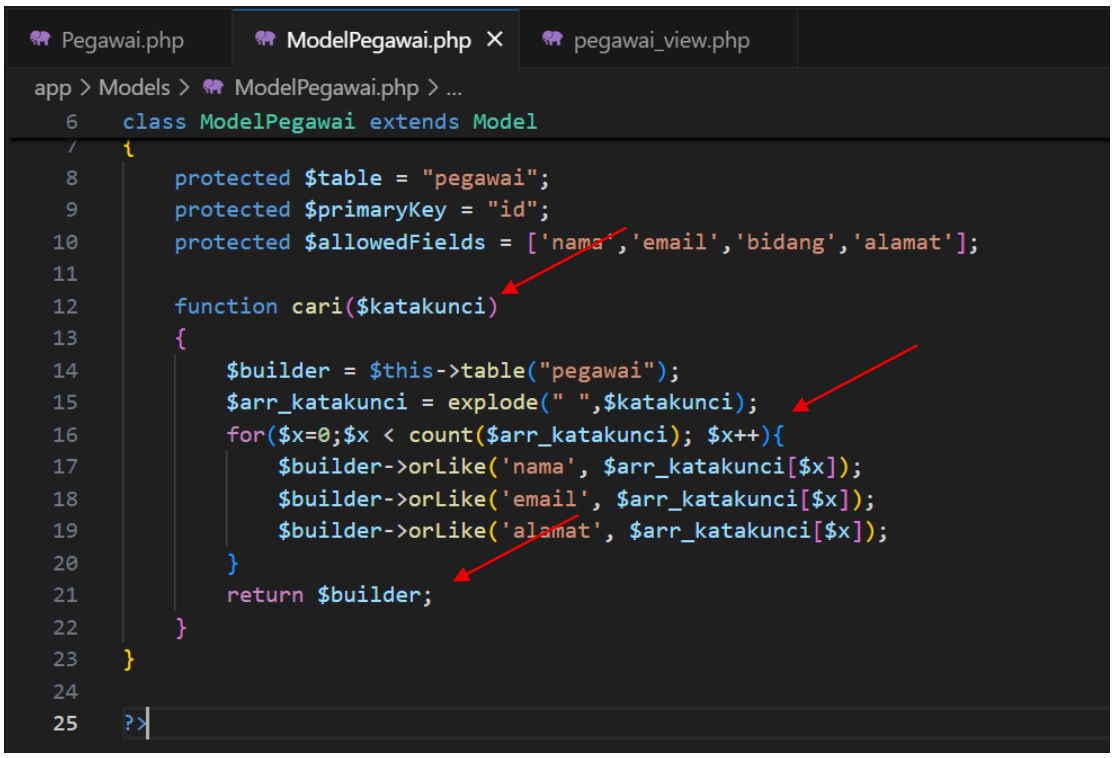


Gambar 5.23 Hasil Halaman 2

21) Sekarang saatnya mengaktifkan fitur pencarian. Pada pegawai_view.php tambahkan perintah seperti berikut

Gambar 5.24 Fitur Pencarian pada View Pegawai

22) Buat function cari pada PegawaiModel.php dengan perintah seperti berikut ini



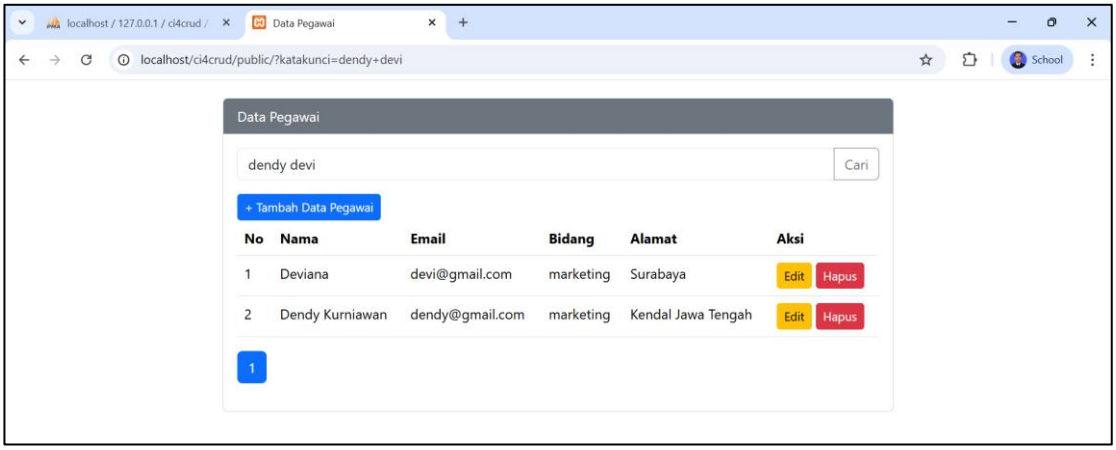
Gambar 5.25 Function Cari pada Model Pegawai

23) Kemudian pada controller Pegawai.php sesuaikan sintaknya seperti berikut ini

```php
class Pegawai extends BaseController
    public function simpan()
        } else {
            $hasil['sukses'] = false;
            $hasil['error'] = $validasi->listErrors();
        }
        return json_encode($hasil);
    }
    public function index(): string
    {
        $jumlahBaris = 4;
        $katakunci = $this->request->getGet('katakunci');
        if($katakunci){
            $pencarian = $this->model->cari($katakunci);
        }else{
            $pencarian = $this->model;
        }

        $halamanSekarang = $this->request->getVar('page') ? $this->request->getVar('page') : 1;
        $data['dataPegawai']= $pencarian->orderBy('id','desc')->paginate($jumlahBaris);
        $data['pager']= $this->model->pager;
        $data['nomor'] = ($jumlahBaris * ($halamanSekarang - 1));
        $data['katakunci'] = $katakunci ?? '';

        return view('pegawai_view', $data);
    }
}
```

Gambar 5.26 Perintah Pencarian pada Kontroler Pegawai

24) Jalankan fitur pencarian tersebut



Gambar 5.27 Hasil Fitur Pencarian

Meskipun menggunakan 2 kata kunci yang dipisahkan dengan spasi, data akan di tampilkan berdasrkan kata masing – masing kunci yang di tuliskan (ada 2 kata kunci yang dimasukan). Ini karena perintah **$arr_katakunci = explode(" ",$katakunci);** yang ada pada pegawaiModel.php baris ke 15

Rangkuman Bab

Pada bab ini telah dibahas proses menampilkan data dari basis data ke dalam aplikasi menggunakan framework CodeIgniter 4. Melalui penerapan konsep *Model–View–Controller* (MVC), data yang tersimpan pada basis data dapat diambil oleh Model, diproses oleh

Controller, kemudian ditampilkan kepada pengguna melalui View dalam bentuk tabel yang terstruktur dan mudah dipahami. Tahapan ini melengkapi proses pengembangan aplikasi sehingga data yang sebelumnya hanya dapat disimpan kini juga dapat ditampilkan dan dikelola dengan lebih baik.

Pembaca telah mempelajari cara mengambil data dari basis data, menampilkan data pada halaman aplikasi, menerapkan pagination untuk membatasi jumlah data yang ditampilkan dalam setiap halaman, serta mengembangkan fitur pencarian berdasarkan kata kunci. Seluruh proses tersebut menunjukkan bahwa pengelolaan data tidak hanya berfokus pada proses penyimpanan, tetapi juga pada bagaimana data dapat diakses kembali secara cepat, tepat, dan efisien oleh pengguna.

Penerapan pagination memberikan manfaat yang cukup besar ketika jumlah data dalam basis data terus bertambah. Dengan membagi data ke dalam beberapa halaman, tampilan aplikasi menjadi lebih rapi, proses pemuatan halaman menjadi lebih ringan, serta pengguna dapat menavigasi data dengan lebih mudah. Demikian pula dengan fitur pencarian yang memungkinkan pengguna menemukan informasi tertentu tanpa harus menelusuri seluruh isi tabel secara manual. Kedua fitur tersebut merupakan komponen penting yang hampir selalu digunakan dalam berbagai aplikasi berbasis web.

Melalui kegiatan praktikum pada bab ini, pembaca juga memperoleh pengalaman dalam menghubungkan proses pengambilan data dengan penyajian informasi pada antarmuka aplikasi. Setiap perubahan yang terjadi pada basis data dapat langsung ditampilkan melalui View sehingga pengguna selalu memperoleh informasi yang terbaru. Hal ini menunjukkan pentingnya integrasi antara Model, Controller, dan View dalam menghasilkan aplikasi yang dinamis serta mampu merespons kebutuhan pengguna secara efektif.

Selain memahami implementasi teknis, pembaca diharapkan mulai menyadari pentingnya merancang tampilan data yang informatif dan mudah digunakan. Penyajian data yang baik tidak hanya memperhatikan aspek estetika, tetapi juga kemudahan pengguna dalam membaca informasi, melakukan navigasi halaman, serta menemukan data yang dibutuhkan. Dengan demikian, kualitas sebuah aplikasi tidak hanya ditentukan oleh fungsi yang dimilikinya, tetapi juga oleh kemampuan aplikasi dalam menyajikan informasi secara jelas dan efisien.

Sebagai penutup bab ini, seluruh materi yang telah dipelajari menjadi landasan penting dalam menyelesaikan implementasi aplikasi CRUD secara utuh. Kemampuan menampilkan data, mengelola pagination, dan menerapkan fitur pencarian akan dipadukan dengan proses pembaruan serta penghapusan data pada bab berikutnya. Dengan bekal tersebut, pembaca diharapkan mampu membangun aplikasi pengelolaan data yang lebih lengkap, terstruktur, dan sesuai dengan kebutuhan pengembangan sistem informasi berbasis web menggunakan CodeIgniter 4.

Refleksi Pembelajaran

Setelah menyelesaikan bab ini, pembaca diharapkan memahami bahwa proses pengelolaan data dalam aplikasi tidak berhenti pada tahap menyimpan informasi ke dalam database. Data yang telah tersimpan perlu dapat ditampilkan kembali secara terstruktur agar pengguna dapat membaca dan memanfaatkannya sesuai kebutuhan. Melalui praktikum pada

bab ini, pembaca dapat memahami bagaimana data yang terdapat pada database diambil melalui Model, diproses oleh Controller, kemudian disajikan melalui View sehingga menjadi informasi yang dapat digunakan oleh pengguna.

Penerapan pagination dan fitur pencarian juga memberikan pemahaman mengenai pentingnya efisiensi dalam penyajian data. Pembaca perlu menyadari bahwa semakin banyak data yang tersimpan, semakin penting bagi aplikasi untuk memiliki mekanisme penyajian dan pencarian informasi yang teratur. Oleh karena itu, pembaca disarankan mencoba kembali proses praktikum dengan jumlah data yang lebih banyak sehingga dapat mengamati manfaat pagination dan pencarian secara lebih nyata. Latihan tersebut juga dapat membantu pembaca memahami hubungan antara jumlah data, tampilan aplikasi, dan kemudahan pengguna dalam memperoleh informasi.

Penerapan dalam Dunia Nyata

Fitur menampilkan data, pagination, dan pencarian merupakan bagian yang umum ditemukan pada berbagai aplikasi sistem informasi. Sistem akademik dapat menggunakannya untuk menampilkan daftar pembaca, aplikasi penjualan dapat menggunakannya untuk menampilkan daftar produk, sedangkan sistem inventaris dapat memanfaatkannya untuk membantu pengguna menemukan barang tertentu. Dengan demikian, kemampuan yang dipelajari pada bab ini dapat diterapkan pada berbagai jenis aplikasi yang membutuhkan pengelolaan data dalam jumlah yang relatif banyak.

Dalam lingkungan aplikasi yang digunakan secara nyata, penyajian data yang baik akan membantu pengguna memperoleh informasi secara lebih cepat dan terarah. Pengguna tidak harus melihat seluruh data dalam satu halaman atau menelusuri informasi secara manual. Pagination membantu mengatur tampilan data menjadi beberapa bagian, sedangkan fitur pencarian membantu mempersempit informasi berdasarkan kebutuhan pengguna. Kedua fitur tersebut menjadi bagian penting dalam membangun aplikasi yang mudah digunakan dan mampu mendukung aktivitas pengelolaan informasi.

Tips Pengembangan Aplikasi

Dalam menampilkan data, biasakan menyusun informasi dalam bentuk yang mudah dibaca dan memiliki struktur yang konsisten. Gunakan judul kolom yang jelas serta pastikan informasi yang ditampilkan sesuai dengan kebutuhan pengguna. Hindari menampilkan informasi yang tidak diperlukan karena dapat membuat tampilan menjadi terlalu padat dan mengurangi kemudahan pengguna dalam memahami data.

Pada penerapan pagination, perhatikan jumlah data yang ditampilkan pada setiap halaman agar tampilan tetap nyaman digunakan. Pastikan pula navigasi halaman dapat digunakan dengan baik sehingga pengguna dapat berpindah dari satu halaman ke halaman lainnya dengan mudah. Untuk fitur pencarian, gunakan kata kunci dan kondisi pencarian yang sesuai dengan data yang tersedia sehingga hasil yang ditampilkan tetap relevan dengan kebutuhan pengguna.

Selain itu, lakukan pengujian dengan berbagai kondisi data. Coba aplikasi ketika data masih sedikit, ketika jumlah data bertambah, maupun ketika kata kunci pencarian tidak menghasilkan data. Pengujian dengan kondisi yang berbeda akan membantu pembaca

mengetahui bagaimana aplikasi merespons setiap keadaan dan menemukan bagian yang masih perlu diperbaiki.

Persiapan Menuju Bab Berikutnya

Setelah menyelesaikan bab ini, pastikan aplikasi telah mampu mengambil dan menampilkan data dari database dengan baik. Pastikan pagination dapat digunakan untuk berpindah halaman serta fitur pencarian dapat memberikan hasil sesuai dengan kata kunci yang dimasukkan. Pengujian sebaiknya dilakukan menggunakan beberapa data agar pembaca dapat memastikan bahwa setiap fungsi berjalan sesuai dengan tujuan yang diharapkan.

Pada bab berikutnya, aplikasi akan dilengkapi dengan fitur edit dan hapus data sehingga proses pengelolaan data menjadi lebih lengkap. Kemampuan menampilkan dan mencari data yang telah dipelajari pada bab ini akan digunakan sebagai bagian dari proses tersebut. Dengan demikian, pembaca akan semakin memahami bagaimana seluruh komponen yang telah dipelajari dari bab-bab sebelumnya saling terhubung untuk membentuk aplikasi CRUD menggunakan CodeIgniter 4 secara utuh.

BAB 6
EDIT DAN HAPUS DATA

Bab ini merupakan tahapan akhir dalam proses pembangunan aplikasi pengelolaan data menggunakan framework CodeIgniter 4. Setelah pada bab sebelumnya aplikasi mampu menampilkan data, melakukan pagination, serta menyediakan fasilitas pencarian, langkah berikutnya adalah melengkapi aplikasi dengan kemampuan untuk memperbarui (*update*) dan menghapus (*delete*) data. Kedua fitur tersebut merupakan bagian penting dari konsep CRUD (*Create, Read, Update, Delete*) yang menjadi dasar dalam pengembangan berbagai aplikasi berbasis web maupun sistem informasi.

Kemampuan melakukan perubahan terhadap data memberikan fleksibilitas kepada pengguna ketika terdapat informasi yang perlu diperbaiki atau diperbarui. Dalam praktiknya, data yang telah tersimpan di dalam basis data tidak selalu bersifat tetap. Perubahan identitas pengguna, pembaruan informasi, maupun koreksi terhadap kesalahan input merupakan kondisi yang sering dijumpai dalam penggunaan aplikasi sehari-hari. Oleh karena itu, fitur edit menjadi salah satu komponen yang wajib tersedia agar data yang tersimpan selalu sesuai dengan kondisi sebenarnya.

Selain proses pembaruan data, aplikasi juga harus mampu menghapus informasi yang sudah tidak diperlukan. Fitur hapus berfungsi untuk menjaga kualitas dan konsistensi data sehingga basis data tidak dipenuhi oleh informasi yang sudah tidak relevan. Namun demikian, proses penghapusan data perlu dilakukan secara hati-hati karena data yang telah dihapus umumnya tidak dapat dikembalikan. Oleh sebab itu, pada pengembangan aplikasi sering kali ditambahkan mekanisme konfirmasi sebelum proses penghapusan dilakukan sebagai bentuk perlindungan terhadap kesalahan pengguna.

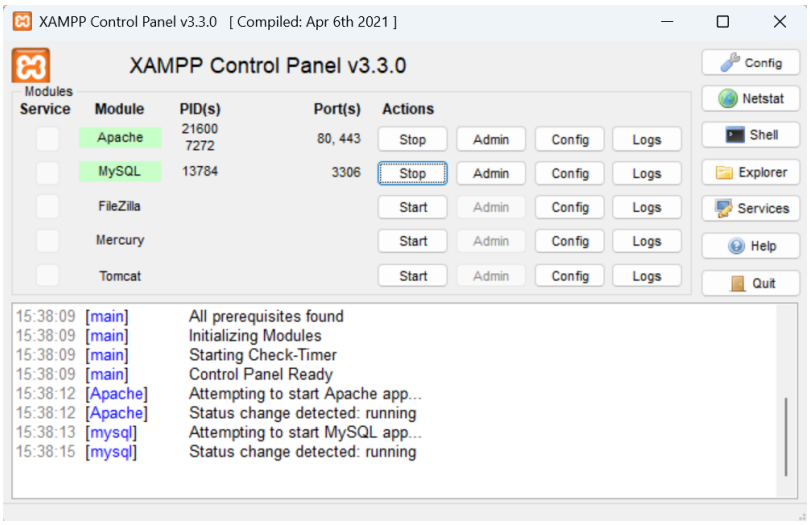
Dalam implementasinya, proses edit dan hapus tetap mengikuti pola kerja *Model–View–Controller* (MVC) yang telah dipelajari pada bab-bab sebelumnya. Controller menerima permintaan dari pengguna, kemudian meneruskannya kepada Model untuk melakukan proses pembaruan atau penghapusan data pada basis data. Setelah proses selesai, Controller mengirimkan respons kepada View agar informasi terbaru dapat ditampilkan kembali kepada pengguna. Alur kerja tersebut menunjukkan bahwa seluruh komponen MVC saling bekerja sama dalam menghasilkan aplikasi yang terstruktur dan mudah dipelihara.

Melalui kegiatan praktikum pada bab ini, pembaca akan mempelajari tahapan pengambilan data berdasarkan identitas tertentu, menampilkan data pada form edit, melakukan proses pembaruan informasi, serta menghapus data yang telah tersimpan pada basis data. Setiap langkah disusun secara sistematis agar pembaca dapat memahami hubungan antara proses pengambilan data, perubahan informasi, dan penyimpanan hasil perubahan ke dalam basis data.

Dengan menyelesaikan seluruh materi pada bab ini, pembaca diharapkan telah mampu membangun sebuah aplikasi CRUD sederhana yang memiliki fungsi lengkap, mulai dari menambahkan data, menampilkan data, mencari data, memperbarui data, hingga

menghapus data. Penguasaan seluruh tahapan tersebut menjadi bekal yang penting dalam mengembangkan aplikasi web yang lebih kompleks pada mata kuliah maupun proyek pengembangan perangkat lunak di masa mendatang.

1) Pastikan web server sudah di aktifkan (apache dan mysql)

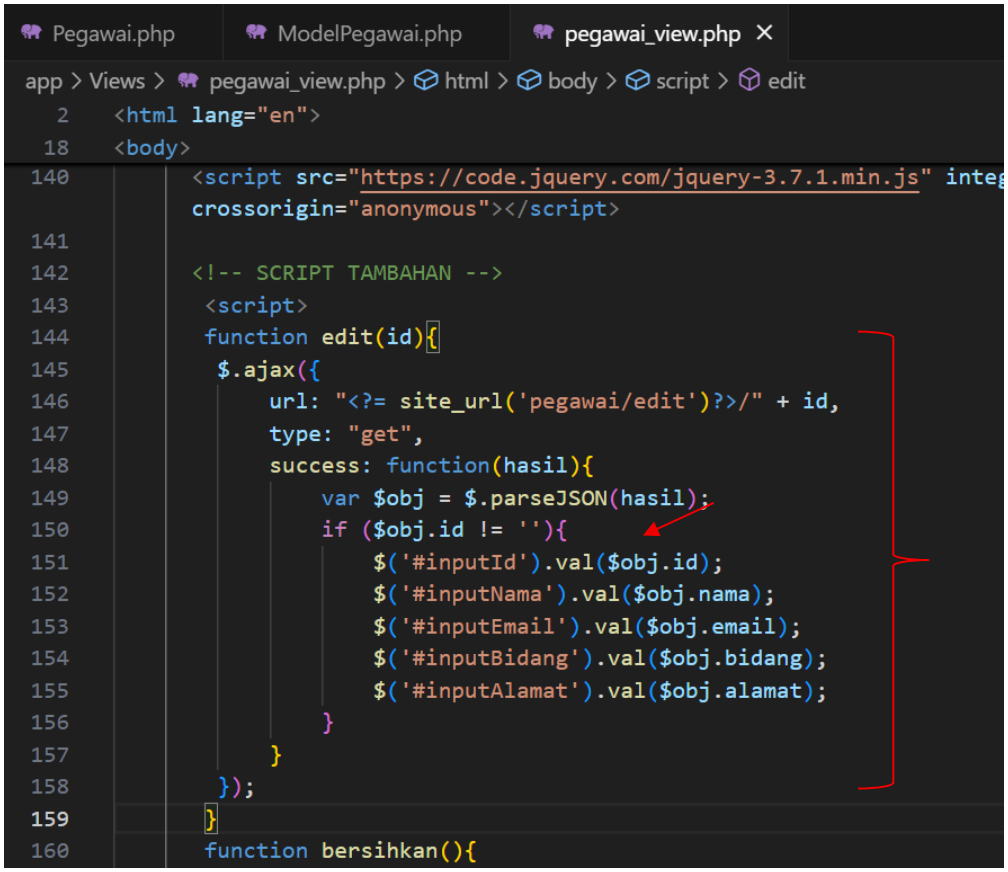


Gambar 6.1 XAMPP

2) Buka file pegawai_view.php dan tambahkan script seperti berikut ini pada bagian tombol edit

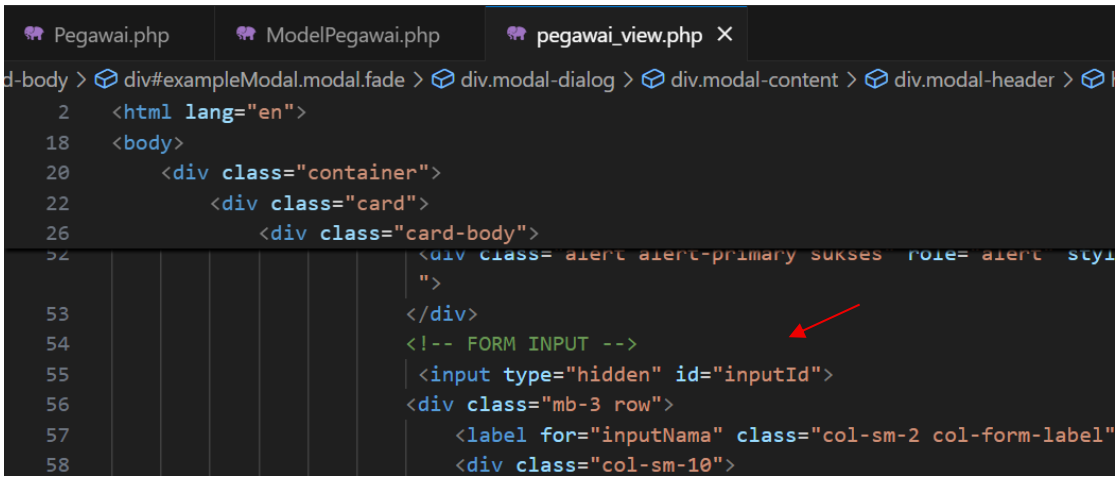
Gambar 6.2 Menambah Tombol Edit dan Hapus pada View Pegawai

3) Lalu tambahkan function edit pada script sperti berikut



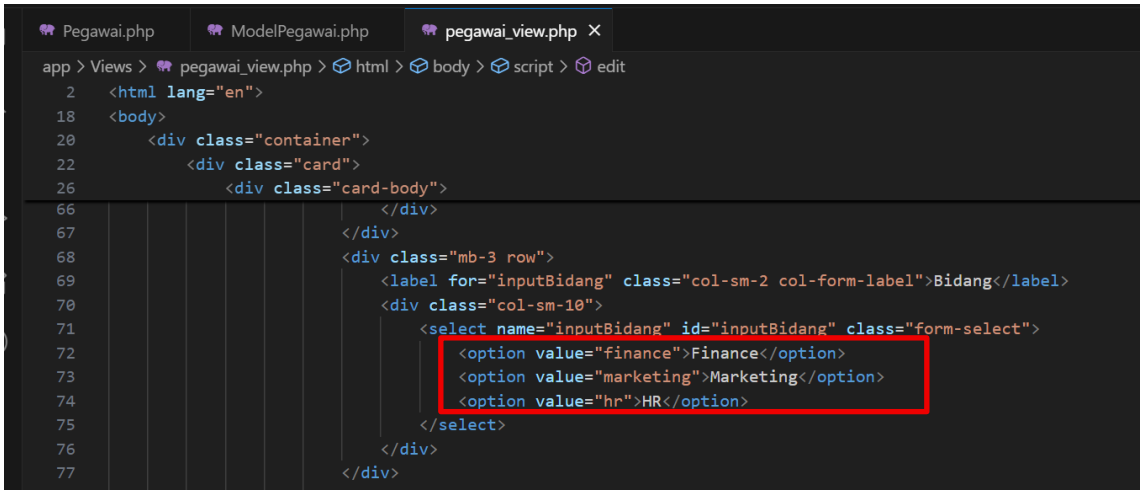
Gambar 6.3 Function Edit

4) Berikan input bantuan pada form modal meskipun typenya hidden



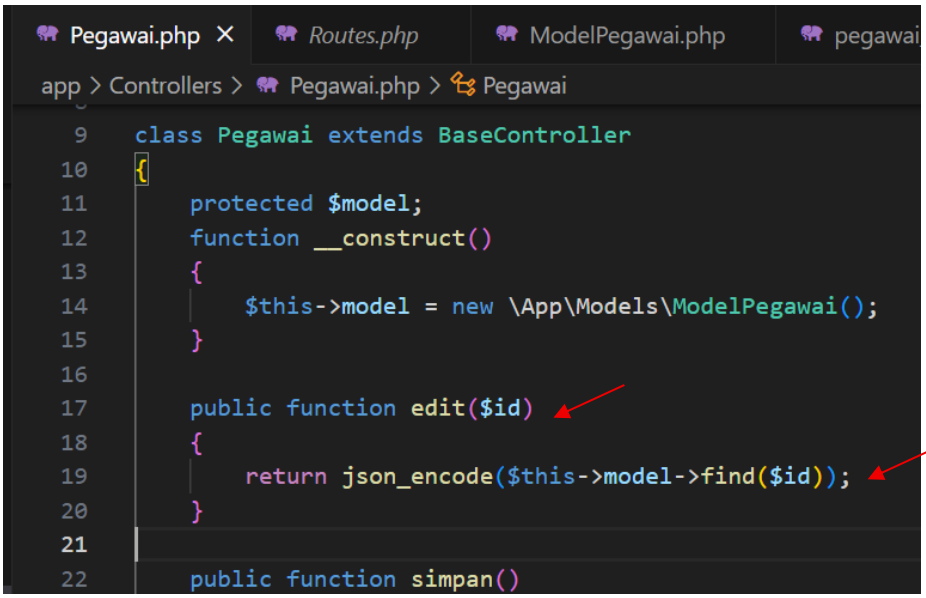
Gambar 6.4 Form Modal Edit

5) Pastikan value kode HTML pada pilihan bidang sudah sesuai dengan enum database bidang. Besar kecilnya huruf sangat berpengarus (harus sama persis antara di kode html dengan di enum database bagian bidang)



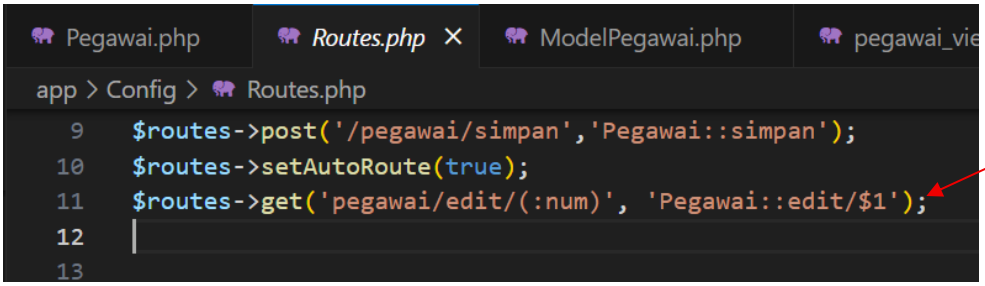
Gambar 6.5 Cek Value Select dengan Enum Database

6) Buatkan controller edit pada controller Pegawai.php seperti berikut ini



Gambar 6.6 Tambah Perintah Controller Edit

7) Tambahkan perintah pada routes

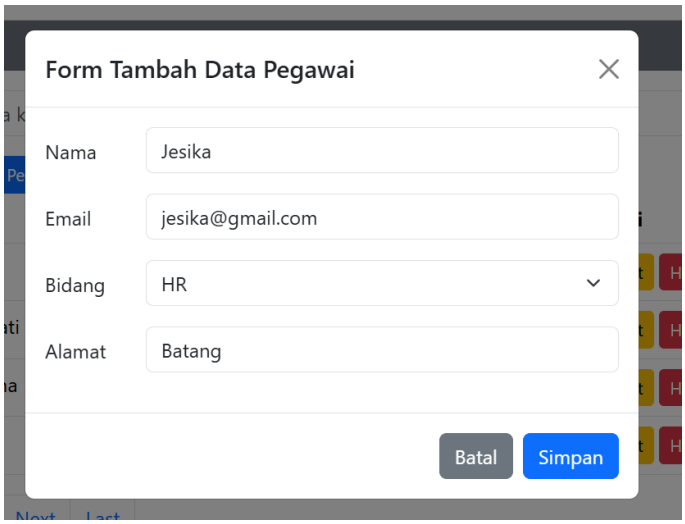


Gambar 6.7 Tambah Perintah pada Routes

Table 6.1 Penjelasan Sintak Gambar 6.7

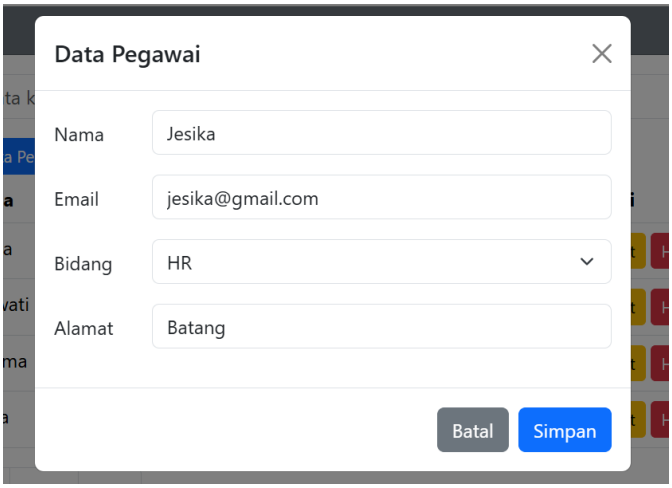
Bagian	Arti
$routes->get()	Jenis HTTP request yang diterima (GET). Bisa juga $routes->post(), $routes->put(), dll.
'pegawai/edit/(:num)'	Pola URL yang akan ditangkap. (:num) artinya *placeholder angka* (regex untuk numeric).
'Pegawai::edit/$1'	Controller dan method yang akan dipanggil. $1 berarti nilai dari (:num) tadi.

8) Silahkan jalankan tombol edit nya, maka seharusnya data yang akan di edit sudah masuk ke dalam form modal seperti berikut



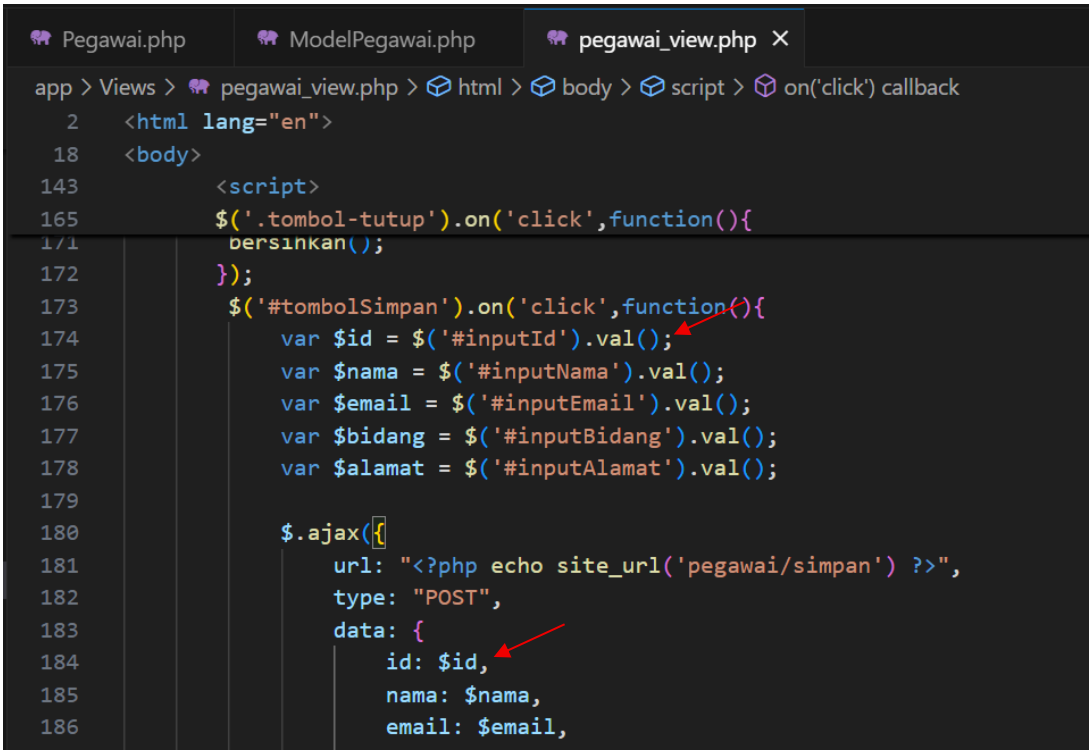
Gambar 6.8 Hasil Jalankan Tombol Edit

9) Karena kita menggunakan 1 form untuk isian dan edit data, maka sebaiknya judul form nya diberikan nama yang bisa di gunakan ke dua fungsi tersebut, seperti contoh Data Pegawai. Edit pada bagian pegawai_view.php dan cari judul form nya



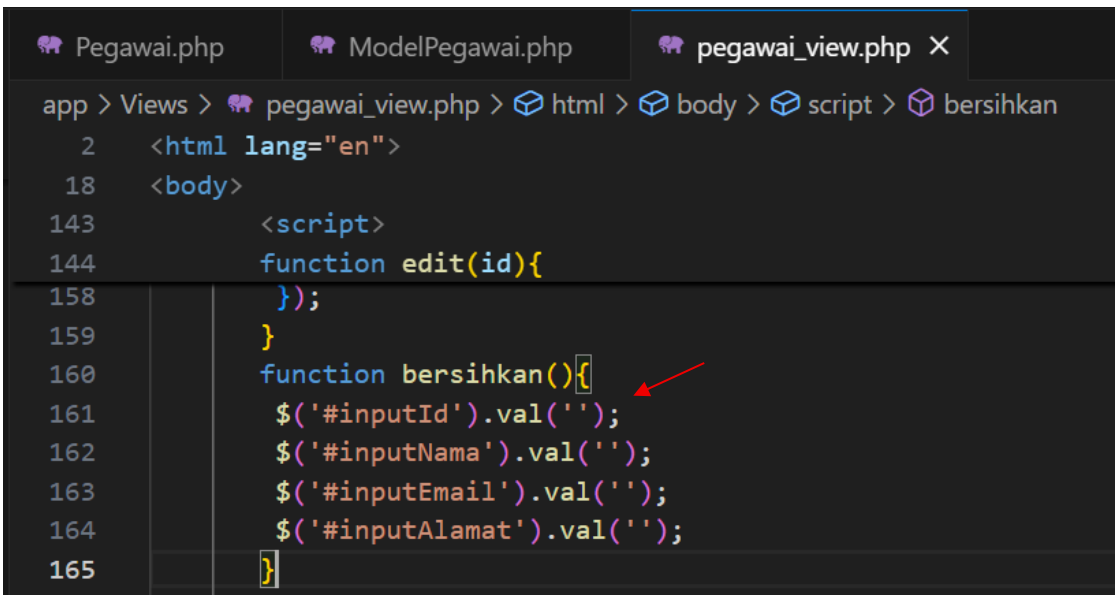
Gambar 6.9 Ubah Judul Form

10) Proses ini belum selesai. Selanjutnya tambahkan id juga pada tombol simpan



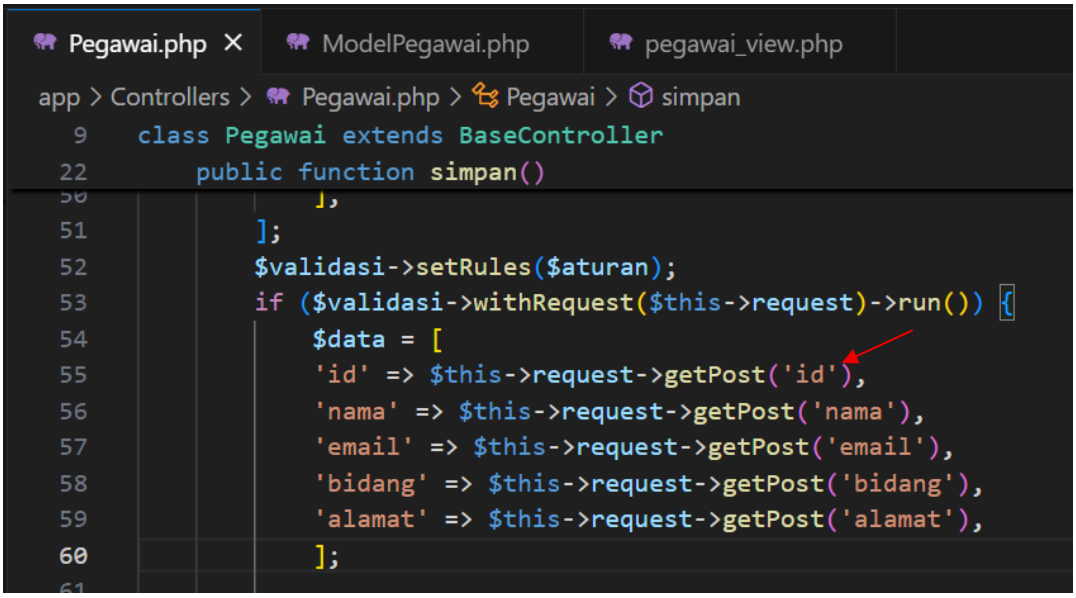
Gambar 6.10 Tambahkan ID pada Tombol Simpan

11) Begitu juga pada function bersihkan pada pegawai_view.php



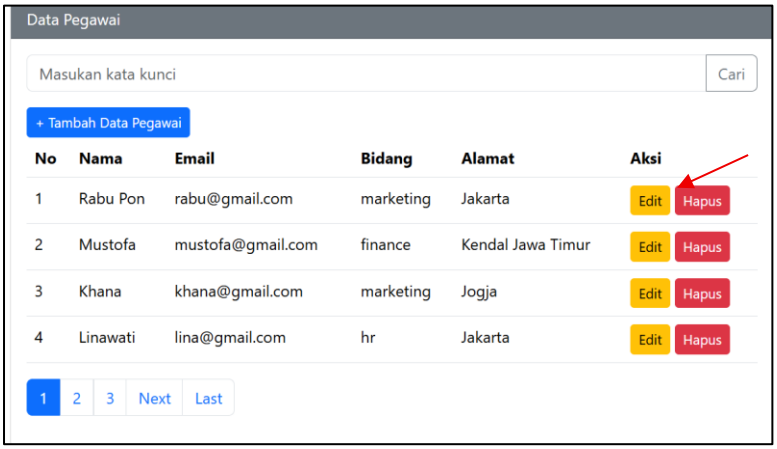
Gambar 6.11 Tambahkan Inputid pada View Pegawai

12) Berikan juga pada controller pegawai.php

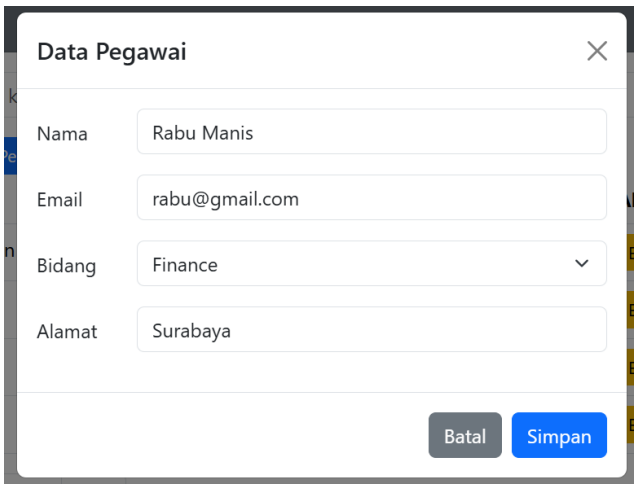


Gambar 6.12 Tambah ID pada Controller Pegawai

13) Silahkan coba edit salah 1 data yang sudah ada

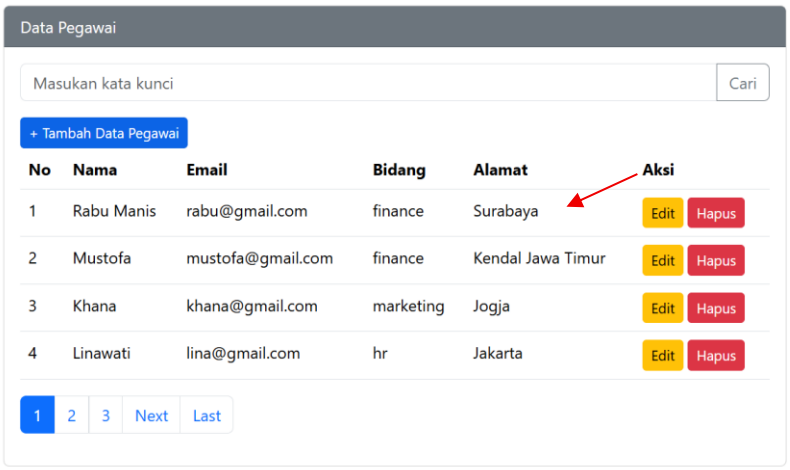


Gambar 6.13 Klok Tombol Edit



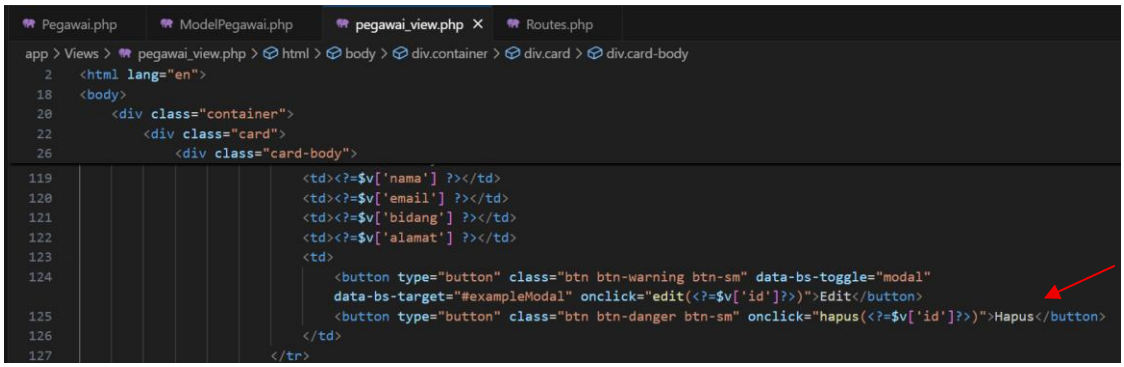
Gambar 6.14 Edit Data Sebagai Uji Coba

Setelah di edit



Gambar 6.15 Setelah di Edit

14) Silahkan juga cek Kembali fitur pencarian dan tambah data baru, seharusnya tetap berjalan sebagaimana mestinya.
15) Sekarang jalankan fungsi hapusnya. Aktifkan tombol hapus pada pegawai_view.php

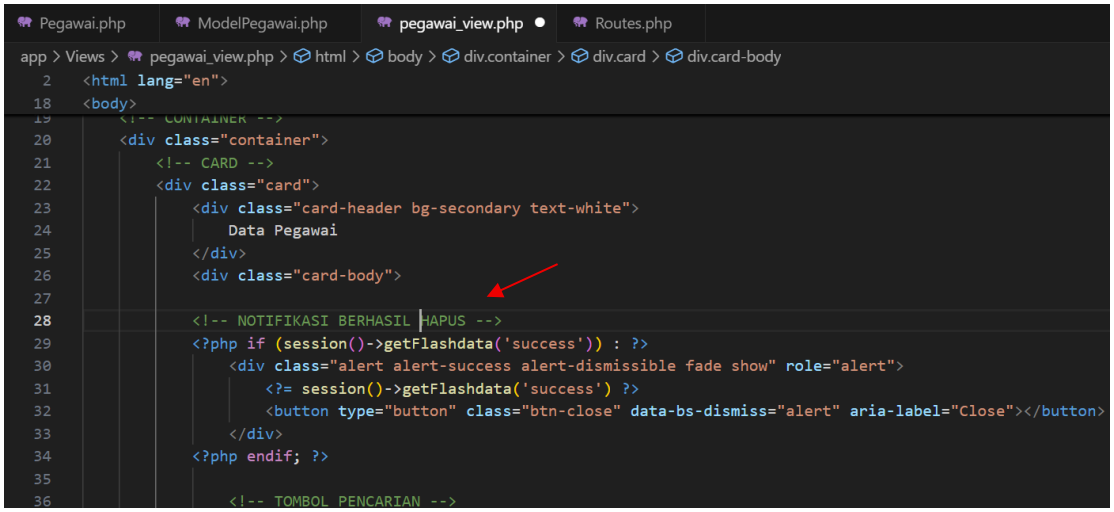


Gambar 6.16 Jalankan Fungsi Hapus

16) Berikan perintah script function hapus seperti dibawah ini

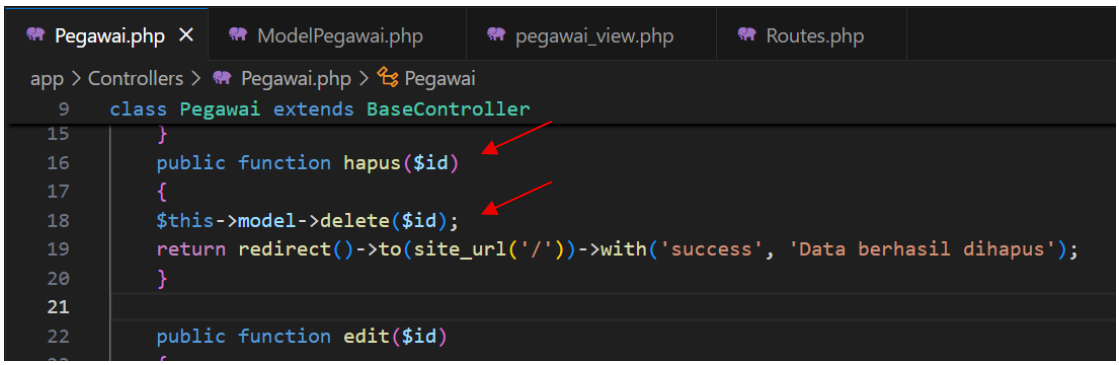
Gambar 6.17 Tambakan Function pada View Pegawai

17) Berikan notifikasi jika data telah berhasil dihapus



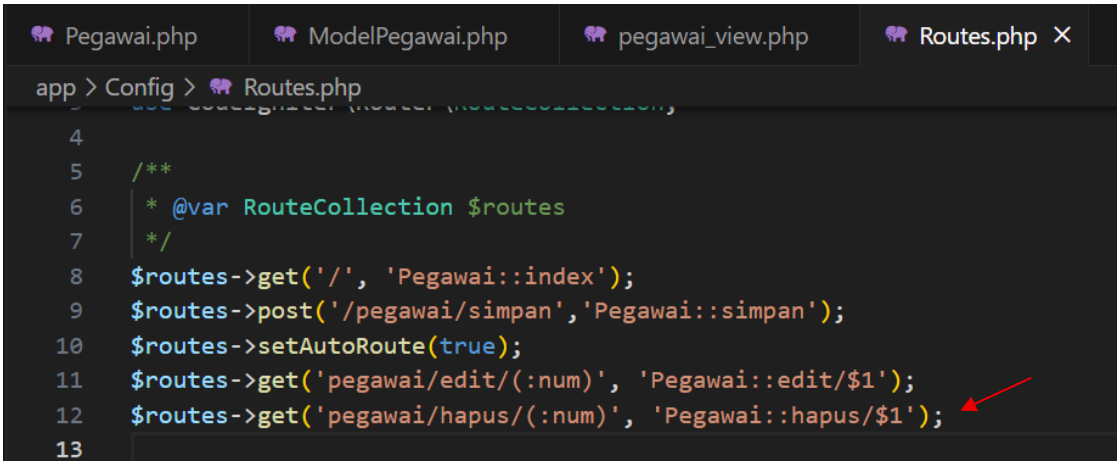
Gambar 6.18 Notifikasi Hapus pada View Pegawai

18) Buat function hapus pada controller pegawai.php



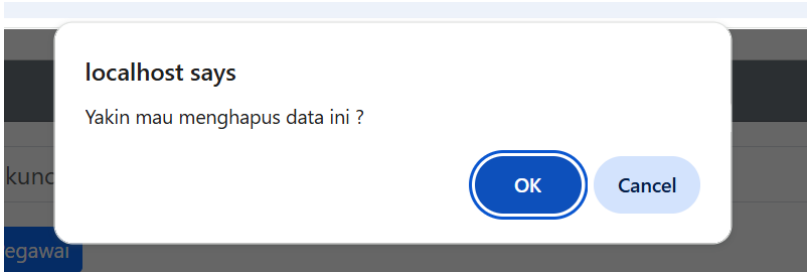
Gambar 6.19 Function pada Kontroller Pegawai

19) Perkenalkan function hapus tersebut di routes nya

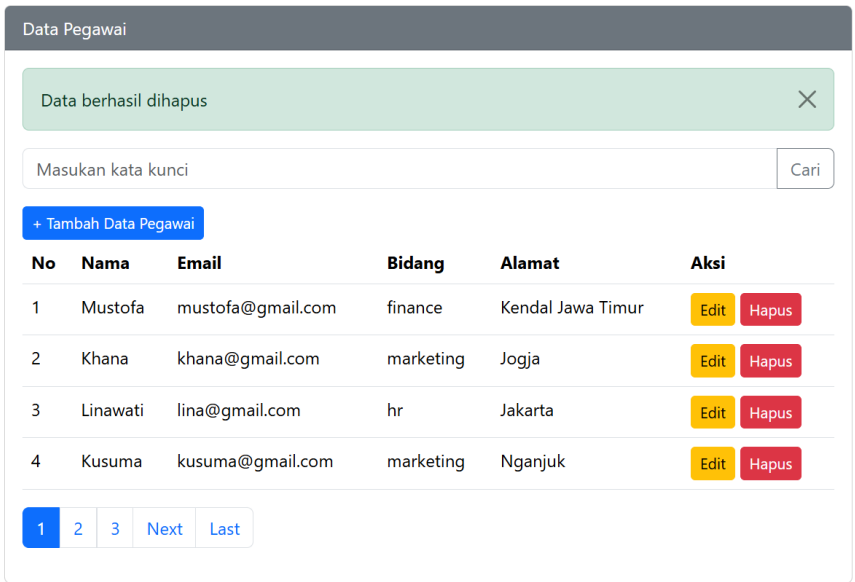


Gambar 6.20 Update Routes

20) Silahkan coba salah 1 data di hapus



Gambar 6.21 Tombol Hapus di Tekan



Gambar 6.22 Setelah Salah 1 Data Dihapus

Melalui tutorial ini, diharapkan pembaca dapat memahami dasar pembuatan aplikasi CRUD menggunakan CodeIgniter 4 dengan baik. Langkah-langkah yang dijelaskan mulai dari pengaturan awal hingga proses simpan, pencarian, pagination, edit, dan hapus data dapat menjadi bekal awal dalam membuat aplikasi web yang dinamis.

Semoga jonsheet ini bisa menjadi panduan praktis bagi pembaca untuk terus belajar dan mengembangkan kemampuan di bidang pemrograman web. Jangan berhenti di sini masih banyak hal menarik yang bisa dieksplorasi dari CodeIgniter dan dunia pengembangan web di internet.

Rangkuman Bab

Pada bab ini telah dibahas implementasi fitur Edit dan Hapus Data sebagai penyempurna aplikasi CRUD menggunakan framework CodeIgniter 4. Kedua fitur tersebut melengkapi kemampuan aplikasi dalam mengelola data sehingga seluruh proses utama, mulai dari menambahkan, menampilkan, memperbarui, hingga menghapus data, dapat dilakukan melalui satu sistem yang terintegrasi. Dengan demikian, aplikasi yang dibangun telah memenuhi konsep dasar pengelolaan data yang umum diterapkan pada berbagai sistem informasi berbasis web.

Pembaca telah mempelajari bagaimana mengambil data berdasarkan identitas tertentu, menampilkan informasi pada form edit, memperbarui data melalui Model dan Controller, serta menyimpan hasil perubahan ke dalam basis data. Selain itu, pembaca juga telah memahami mekanisme penghapusan data beserta alur komunikasi antara antarmuka aplikasi, Controller, Model, dan basis data sehingga setiap perubahan yang dilakukan pengguna dapat diproses secara tepat dan konsisten.

Implementasi fitur edit dan hapus menunjukkan bahwa setiap komponen dalam arsitektur *Model–View–Controller* (MVC) memiliki peran yang saling melengkapi. Model bertanggung jawab melakukan pengolahan data pada basis data, Controller mengatur jalannya proses sesuai permintaan pengguna, sedangkan View menyajikan informasi hasil pengolahan dalam bentuk antarmuka yang mudah dipahami. Pola kerja tersebut menjadikan aplikasi lebih terstruktur, mudah dipelihara, serta lebih mudah dikembangkan ketika diperlukan penambahan fitur pada masa mendatang.

Melalui seluruh rangkaian praktikum pada bab ini, pembaca tidak hanya memahami cara menuliskan kode program, tetapi juga memperoleh pengalaman mengenai bagaimana sebuah aplikasi web dibangun secara bertahap mulai dari perencanaan struktur data hingga implementasi fitur pengelolaan informasi. Pengalaman tersebut diharapkan dapat menjadi dasar dalam mengembangkan aplikasi yang lebih kompleks dengan tetap menerapkan prinsip-prinsip pengembangan perangkat lunak yang baik.

Selain aspek implementasi teknis, pembaca juga diharapkan memahami pentingnya menjaga integritas dan konsistensi data ketika melakukan proses pembaruan maupun penghapusan informasi. Setiap perubahan terhadap data harus dilakukan secara cermat agar informasi yang tersimpan tetap akurat, dapat dipertanggungjawabkan, serta mendukung kebutuhan pengguna. Oleh karena itu, proses pengujian terhadap setiap fungsi aplikasi perlu dilakukan secara menyeluruh sebelum aplikasi digunakan pada lingkungan yang sebenarnya.

Dengan selesainya bab ini, seluruh rangkaian praktikum dalam buku telah berhasil membentuk sebuah aplikasi CRUD berbasis CodeIgniter 4 yang utuh. Kompetensi yang diperoleh pada setiap bab saling melengkapi dan membentuk pemahaman yang menyeluruh mengenai proses pengembangan aplikasi web menggunakan framework CodeIgniter 4. Diharapkan pembaca tidak hanya mampu mengikuti setiap langkah praktikum, tetapi juga mampu menerapkan konsep yang telah dipelajari untuk membangun aplikasi sesuai kebutuhan, mengembangkan fitur baru, serta menyelesaikan berbagai permasalahan yang ditemui dalam proses pengembangan sistem informasi berbasis web.

Refleksi Pembelajaran

Setelah menyelesaikan bab ini, pembaca diharapkan telah memahami tahapan akhir dalam membangun aplikasi pengelolaan data menggunakan CodeIgniter 4, yaitu melakukan perubahan dan penghapusan data. Fitur edit dan hapus merupakan bagian penting dalam konsep CRUD karena memungkinkan pengguna tidak hanya memasukkan dan melihat data, tetapi juga melakukan penyesuaian terhadap informasi yang telah tersimpan. Melalui praktikum pada bab ini, pembaca dapat memahami bagaimana sebuah perubahan yang dilakukan melalui antarmuka aplikasi dapat diproses hingga menghasilkan perubahan pada database.

Pembaca juga perlu memperhatikan alur proses ketika melakukan edit data. Data yang akan diubah terlebih dahulu perlu ditampilkan kembali pada form sehingga pengguna dapat melihat informasi sebelumnya dan melakukan perubahan sesuai kebutuhan. Setelah perubahan dilakukan, data tersebut diproses kembali melalui komponen aplikasi hingga tersimpan sebagai informasi terbaru. Dengan memahami alur tersebut, pembaca dapat melihat hubungan antara proses pengambilan data, pengolahan data, dan penyimpanan kembali ke database secara lebih menyeluruh.

Pada proses hapus data, pembaca perlu memahami bahwa tindakan tersebut harus dilakukan secara cermat karena dapat menyebabkan informasi yang tersimpan tidak lagi tersedia. Oleh karena itu, sebelum melakukan penghapusan, pembaca perlu memastikan bahwa data yang dipilih memang merupakan data yang dimaksud. Kebiasaan melakukan pemeriksaan sebelum menjalankan suatu proses akan membantu mengurangi kesalahan ketika aplikasi digunakan dalam kondisi sebenarnya.

Penerapan dalam Dunia Nyata

Fitur edit dan hapus data merupakan fungsi yang hampir selalu terdapat dalam aplikasi pengelolaan informasi. Pada sistem akademik, misalnya, administrator dapat memperbarui data pembaca yang mengalami perubahan. Pada sistem penjualan, informasi produk dapat diperbarui ketika terjadi perubahan harga atau informasi lainnya. Demikian pula pada sistem inventaris, data barang dapat diperbarui atau dihapus ketika terjadi perubahan kondisi maupun kebutuhan pengelolaan data.

Penerapan fitur tersebut menunjukkan bahwa aplikasi berbasis web harus mampu menyesuaikan data dengan kondisi yang terus berubah. Informasi yang tersimpan pada suatu sistem tidak selalu bersifat tetap sehingga pengguna memerlukan fasilitas untuk melakukan perubahan maupun penghapusan data sesuai dengan kebutuhan. Kemampuan yang dipelajari

pada bab ini dapat menjadi dasar bagi pembaca untuk mengembangkan berbagai aplikasi pengelolaan data dengan studi kasus yang lebih beragam.

Dalam pengembangan aplikasi secara nyata, proses edit dan hapus juga perlu diperhatikan dari sisi kemudahan penggunaan. Pengguna harus dapat mengetahui data yang sedang diubah dan memahami tindakan yang akan dilakukan. Dengan demikian, pembaca tidak hanya perlu memperhatikan keberhasilan kode program, tetapi juga perlu mempertimbangkan bagaimana fungsi tersebut dapat digunakan secara jelas dan mudah oleh pengguna aplikasi.

Tips Pengembangan Aplikasi

Dalam membuat fitur edit, pastikan data yang akan ditampilkan pada form sesuai dengan identitas data yang dipilih. Periksa kembali setiap nilai sebelum proses pembaruan dilakukan agar informasi yang tersimpan tidak mengalami perubahan yang tidak diinginkan. Setelah proses edit selesai, lakukan pemeriksaan terhadap data yang telah diperbarui untuk memastikan bahwa hasilnya sesuai dengan input pengguna.

Untuk fitur hapus, biasakan melakukan pemeriksaan terlebih dahulu terhadap data yang akan dihapus. Jika diperlukan dalam alur aplikasi yang digunakan, berikan kesempatan kepada pengguna untuk memastikan kembali tindakan tersebut sebelum proses dilakukan. Pengujian juga perlu dilakukan terhadap beberapa kondisi agar dapat diketahui apakah data yang dipilih benar-benar terhapus dan apakah daftar data pada halaman aplikasi telah menampilkan kondisi terbaru.

Selain itu, lakukan pengujian terhadap keseluruhan fungsi CRUD setelah fitur edit dan hapus selesai dibuat. Pastikan proses tambah, tampil, edit, dan hapus dapat berjalan secara berurutan tanpa mengganggu fungsi lainnya. Pengujian secara menyeluruh akan membantu pembaca menemukan hubungan antarfitur dan memastikan aplikasi yang telah dibangun dapat digunakan sesuai dengan tujuan praktikum.

Persiapan Setelah Menyelesaikan Seluruh Praktikum

Dengan selesainya bab ini, seluruh tahapan utama dalam pembangunan aplikasi CRUD menggunakan CodeIgniter 4 telah dipelajari. Pembaca telah melalui proses mulai dari mempersiapkan lingkungan pengembangan, memahami struktur MVC, menghubungkan aplikasi dengan database, membuat form input, menerapkan validasi, menggunakan AJAX dan jQuery, menampilkan data, menerapkan pagination dan pencarian, hingga melakukan edit dan hapus data.

Sebelum mengakhiri seluruh rangkaian praktikum, pembaca disarankan untuk menjalankan kembali aplikasi secara menyeluruh dan memastikan setiap fungsi dapat digunakan dengan baik. Cobalah melakukan proses penambahan data, menampilkan data, mencari informasi tertentu, melakukan perubahan, kemudian menghapus data yang sudah tidak diperlukan. Kegiatan tersebut dapat membantu memastikan bahwa setiap materi yang telah dipelajari pada bab-bab sebelumnya telah terintegrasi dalam satu aplikasi.

Keseluruhan proses tersebut diharapkan dapat memberikan pengalaman belajar yang lebih utuh mengenai pengembangan aplikasi web menggunakan CodeIgniter 4. Pembaca tidak hanya memperoleh pengalaman dalam mengikuti langkah-langkah praktikum, tetapi juga

memahami bagaimana berbagai komponen dan fitur aplikasi saling terhubung untuk menghasilkan sistem pengelolaan data yang dapat digunakan secara nyata.

PENUTUP

Puji syukur ke hadirat Tuhan Yang Maha Esa, seluruh rangkaian materi dalam buku ajar PEMROGRAMAN WEB DENGAN CODEIGNITER 4: Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD telah disusun secara sistematis untuk memberikan pemahaman sekaligus pengalaman praktis kepada pembaca dalam membangun aplikasi web menggunakan framework CodeIgniter 4. Buku ini dirancang sebagai buku ajar praktikum sehingga setiap materi disusun secara bertahap, dimulai dari konsep dasar hingga implementasi aplikasi CRUD yang utuh. Pendekatan tersebut diharapkan dapat membantu pembaca mempelajari setiap tahapan secara lebih mudah, terarah, dan sesuai dengan alur pengembangan perangkat lunak.

Pembahasan dalam buku ini diawali dengan persiapan lingkungan pengembangan yang meliputi instalasi perangkat lunak pendukung, konfigurasi CodeIgniter 4, serta pengenalan struktur proyek yang digunakan selama proses praktikum. Tahapan awal ini menjadi fondasi penting karena keberhasilan dalam menyiapkan lingkungan pengembangan akan sangat menentukan kelancaran proses implementasi pada bab-bab berikutnya. Melalui proses tersebut, pembaca diperkenalkan dengan berbagai perangkat yang umum digunakan oleh pengembang aplikasi web profesional, seperti XAMPP, Composer, Visual Studio Code, dan CodeIgniter 4.

Selanjutnya, buku ini mengajak pembaca untuk memahami konsep *Model–View–Controller* (MVC) sebagai arsitektur utama dalam pengembangan aplikasi menggunakan CodeIgniter 4. Pemahaman terhadap konsep ini menjadi dasar yang sangat penting karena seluruh implementasi pada bab-bab berikutnya dibangun berdasarkan pola kerja MVC. Dengan memahami peran masing-masing komponen, yaitu Model sebagai pengelola data, View sebagai penyaji antarmuka, dan Controller sebagai pengatur alur aplikasi, pembaca diharapkan mampu menyusun kode program yang lebih terstruktur, mudah dipelihara, serta mudah dikembangkan sesuai kebutuhan.

Pada tahap implementasi, pembaca mempelajari proses membangun koneksi dengan basis data, menyusun Model, membuat antarmuka menggunakan Bootstrap 5, serta mengembangkan form input sebagai awal dari proses pengelolaan data. Materi kemudian dilanjutkan dengan penerapan validasi data, penggunaan AJAX dan jQuery untuk meningkatkan interaktivitas aplikasi, serta implementasi berbagai fitur yang mendukung pengelolaan data secara dinamis. Melalui tahapan tersebut, pembaca tidak hanya mempelajari cara menulis kode program, tetapi juga memahami bagaimana setiap komponen saling berinteraksi dalam menghasilkan sebuah aplikasi yang berjalan secara utuh.

Pembahasan selanjutnya difokuskan pada implementasi konsep CRUD *(Create, Read, Update, Delete)* yang merupakan salah satu kemampuan dasar dalam pengembangan sistem informasi berbasis web. Pembaca mempelajari cara menyimpan data, menampilkan data dalam bentuk tabel, menerapkan pagination untuk mengelola jumlah data yang ditampilkan, menyediakan fitur pencarian, melakukan perubahan data, hingga menghapus data yang sudah tidak diperlukan. Keseluruhan proses tersebut disusun secara bertahap sehingga pembaca

dapat mengikuti setiap langkah praktikum dengan mudah dan memahami hubungan antara teori dengan implementasi secara langsung.

Melalui seluruh rangkaian praktikum yang disajikan, pembaca diharapkan mampu memahami bagaimana sebuah aplikasi web dibangun mulai dari tahap persiapan lingkungan kerja hingga menjadi aplikasi CRUD yang lengkap. Lebih dari sekadar mengikuti langkah-langkah praktikum, pembaca diharapkan mampu memahami alasan di balik setiap proses yang dilakukan, sehingga dapat menerapkan konsep yang sama ketika menghadapi studi kasus atau proyek pengembangan aplikasi yang berbeda. Kemampuan untuk memahami konsep dan menerapkannya secara mandiri merupakan tujuan utama dari penyusunan buku ajar ini.

Di sisi lain, penulis menyadari bahwa perkembangan teknologi web berlangsung sangat cepat. Framework, bahasa pemrograman, maupun perangkat pendukung akan terus mengalami pembaruan seiring dengan berkembangnya kebutuhan industri. Oleh karena itu, buku ini tidak dimaksudkan sebagai satu-satunya sumber belajar, melainkan sebagai fondasi yang dapat membantu pembaca membangun pemahaman awal mengenai pengembangan aplikasi web menggunakan CodeIgniter 4. Pembaca tetap dianjurkan untuk mempelajari dokumentasi resmi, mengikuti perkembangan teknologi, serta terus berlatih mengembangkan berbagai jenis aplikasi agar kemampuan yang dimiliki semakin meningkat.

Penulis juga berharap buku ini dapat menjadi salah satu referensi yang bermanfaat bagi pembaca, dosen, maupun praktisi yang ingin mempelajari pengembangan aplikasi web berbasis framework CodeIgniter 4. Penyusunan materi secara bertahap diharapkan mampu memberikan pengalaman belajar yang lebih terstruktur, sehingga pembaca tidak hanya mengetahui cara menggunakan framework, tetapi juga memahami prinsip-prinsip dasar dalam membangun aplikasi web yang baik, terorganisasi, dan mudah dikembangkan.

Akhirnya, penulis menyadari bahwa buku ini masih memiliki berbagai keterbatasan, baik dari sisi penyajian maupun kedalaman materi. Oleh karena itu, kritik, saran, dan masukan yang bersifat membangun sangat diharapkan sebagai bahan evaluasi untuk penyempurnaan pada edisi berikutnya. Semoga buku ajar PEMROGRAMAN WEB DENGAN CODEIGNITER 4: Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD dapat memberikan manfaat yang nyata, menjadi pendamping dalam proses pembelajaran, serta mampu meningkatkan kompetensi pembaca dalam mengembangkan aplikasi web yang berkualitas, terstruktur, dan sesuai dengan kebutuhan dunia akademik maupun industri.

Selamat belajar, terus berlatih, dan jadikan setiap proses pengembangan aplikasi sebagai sarana untuk meningkatkan kemampuan, kreativitas, serta profesionalisme dalam bidang pemrograman web. Jangan pernah berhenti untuk mengeksplorasi teknologi, mencoba berbagai studi kasus, dan mengembangkan solusi yang inovatif, karena pengalaman yang diperoleh melalui praktik secara langsung merupakan bekal yang sangat berharga dalam menghadapi perkembangan dunia teknologi informasi yang terus berubah. Semoga ilmu dan keterampilan yang diperoleh melalui buku ini dapat menjadi fondasi yang kuat untuk menghasilkan aplikasi web yang berkualitas, memberikan manfaat bagi masyarakat, serta mendukung perjalanan akademik maupun karier profesional sebagai pengembang perangkat lunak di masa depan.

DAFTAR PUSTAKA

Akbar, N., & Kurniawan, D. (2024). Perancangan sistem informasi penjualan gadget di PT. Kafilah Teknologi Indonesia. *Jurnal Riset Teknik Komputer (JURTIKOM), 1*(1), 1–7. https://doi.org/10.69714/ftk6tr19

Aqham, A. A., Siswanto, E., & Kurniawan, D. (2023). Metode enterprise architecture planning dalam sistem informasi pengelolaan data inventaris. *Jurnal Teknologi Informasi dan Komunikasi, 14*(1), 201–208.

CodeIgniter Foundation. (2026). *CodeIgniter 4 user guide*. CodeIgniter Foundation.

CodeIgniter Foundation. (2026). *CodeIgniter 4 user guide: Pagination*. CodeIgniter Foundation.

CodeIgniter Foundation. (2026). *CodeIgniter 4 user guide: Using CodeIgniter's model*. CodeIgniter Foundation.

GetComposer.org. (2026). *Composer: Dependency manager for PHP*. Composer.

Hakikiyah, S. A., Sudirman, B., Aqham, A. A., Kurniawan, D., & Muthohir, M. (2026). Implementasi sistem informasi penjualan berbasis web menggunakan framework Laravel 12: Studi kasus pada Toko Ulkids Pegandon. *Jurnal Informatika dan Teknologi Komputer, 6*(1), 115–127.

Kurniawan, D. (2023). *Belajar pemrograman web dasar HTML, CSS, & Javascript untuk pemula*. Yayasan Prima Agus Teknik.

Pamungkas, S. P., Kurniawan, D., & Kurnialensya, T. (2025). Perancangan sistem informasi penjualan berbasis web menggunakan metode object oriented. *Jurnal Ilmiah Sistem Informasi (JUISI), 4*(1), 25–36.

PHP Documentation Group. (2026). *PHP manual*. The PHP Group.

The Bootstrap Authors. (2026). *Bootstrap 5 documentation*. Bootstrap.

The jQuery Foundation. (2026). *jQuery API documentation*. jQuery.

PEMROGRAMAN WEB
DENGAN
CODEIGNITER 4
Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD

Dendy Kurniawan, S.Kom., M.Kom

TENTANG PENULIS

Dendy Kurniawan, S.Kom., M.Kom. merupakan dosen pada bidang Sistem Informasi dan Komputer di Universitas Sains dan Teknologi Komputer (Universitas STEKOM), Indonesia. Penulis memiliki pengalaman dalam bidang pendidikan tinggi, pengajaran, serta penelitian di bidang teknologi sistem informasi, rek ayasa perangkat lunak, Internet of Things (IoT), kecerdasan buatan, dan pengembangan aplikasi berbasis teknologi informasi. Dalam kegiatan akademik, penulis aktif mengembangkan bahan ajar dan melakukan penelitian yang berkaitan dengan pengembangan sistem dan penerapan teknologi informasi. Buku “Pemrograman Web dengan CodeIgniter 4: Konsep, Praktikum MVC, dan Pengembangan Aplikasi CRUD” disusun sebagai bagian dari upaya penulis dalam menyediakan bahan pembelajaran praktis bagi pembaca untuk memahami konsep dan penerapan pemrograman web menggunakan framework CodeIgniter 4. Penulis dapat dihubungi melalui email: dendy@stekom.ac.id

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 978-634-7695-49-9 (PDF)