

Apa itu Laravel

Laravel adalah sebuah framework aplikasi web berbasis PHP yang dirancang untuk memudahkan proses pengembangan aplikasi web dengan cara yang lebih elegan dan ekspresif. Framework ini diciptakan oleh Taylor Otwell dan pertama kali dirilis pada tahun 2011. Laravel hadir sebagai platform web development yang bersifat *open source*. Yang menarik dari Laravel adalah sintaksnya yang ekspresif dan elegan yang dirancang khusus untuk memudahkan dan mempercepat proses web development.

Dengan Laravel, tugas-tugas umum developer dapat dikurangi pada sebagian besar proyek-proyek web seperti *routing*, *session* dan *caching*. Disamping itu, laravel berusaha menggabungkan pengalaman-pengalaman development dalam bahasa lain, seperti Ruby on Rails, ASP.NET, MVC dan Sinatra.

Kenapa Memakai Laravel

Sesuai dengan motto laravel itu sendiri **“PHP doesn’t hurt, code happy & enjoy the fresh air”**. Tujuan utama dari laravel adalah mempermudah *coding* dalam membuat sebuah produk web. Bahkan laravel termasuk dalam best php framework 2014 versi webdesignmoo dan yang paling banyak digunakan oleh developer. Ini membuktikan bahwa menggunakan Laravel memang dapat mempercepat dan mempermudah *development* website.

Memulai Laravel (Instalasi dan Konfigurasi)

Requirement

Laravel sangatlah mudah untuk dikonfigurasi untuk mengembangkan sebuah projek. Pada bagian ini, saya akan menjelaskan *software/tools* apa saja yang diperlukan, proses instalasi dan proses konfigurasinya.

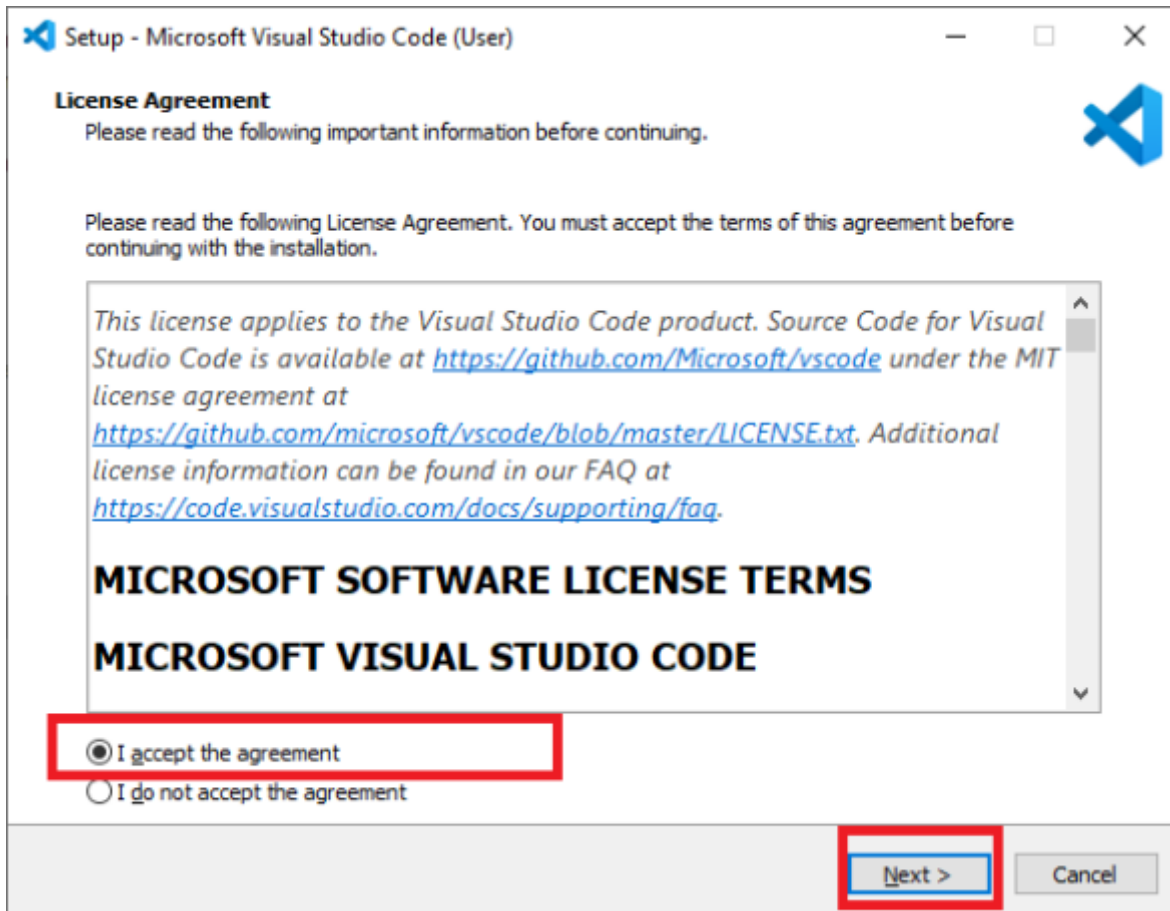
Adapun kebutuhan yang harus disediakan diantaranya :

Text Editor

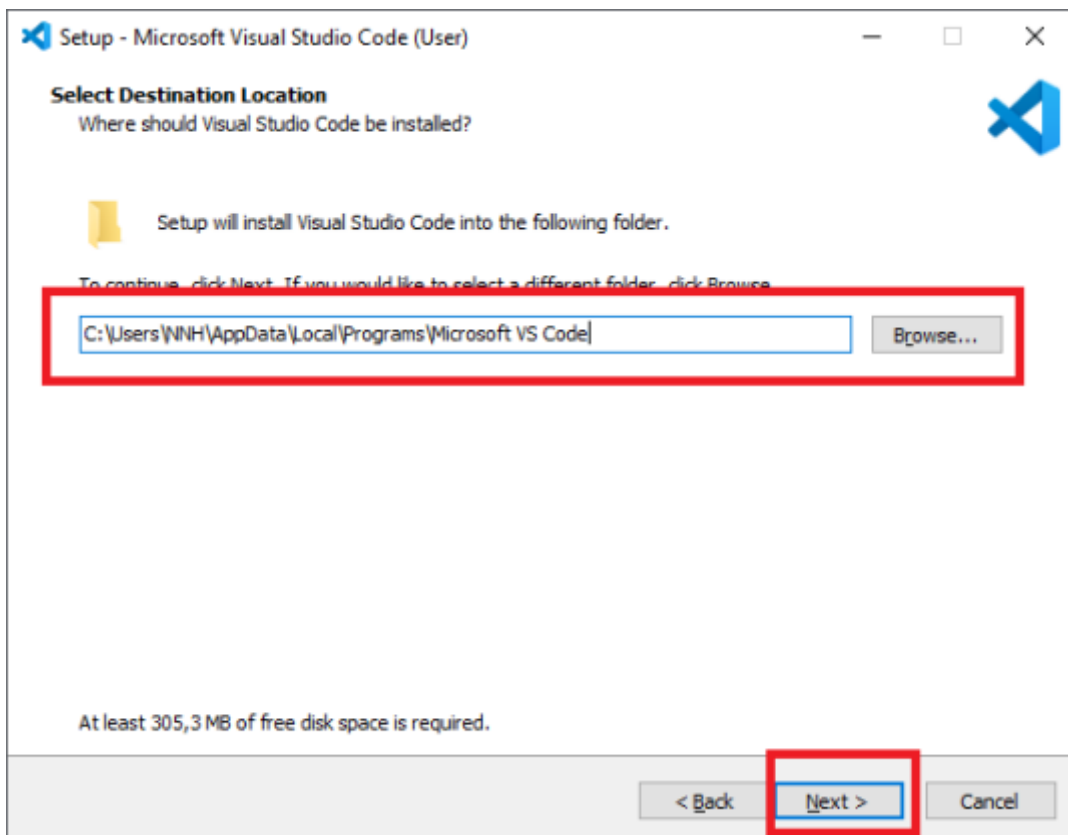
Pilih text editor yang sesuai dengan kebutuhan atau selera Anda. Disini saya menggunakan text editor Visual Studio Code. Anda bisa juga menggunakan PHPStorm, Aptana, Netbeans, Notepad++ dan lain-lain. Berikut adalah langkah-langkah penginstalan text editor yang saya gunakan (Visual studio code) :

- Kunjungi situs resmi visual studio code : <https://code.visualstudio.com/download>

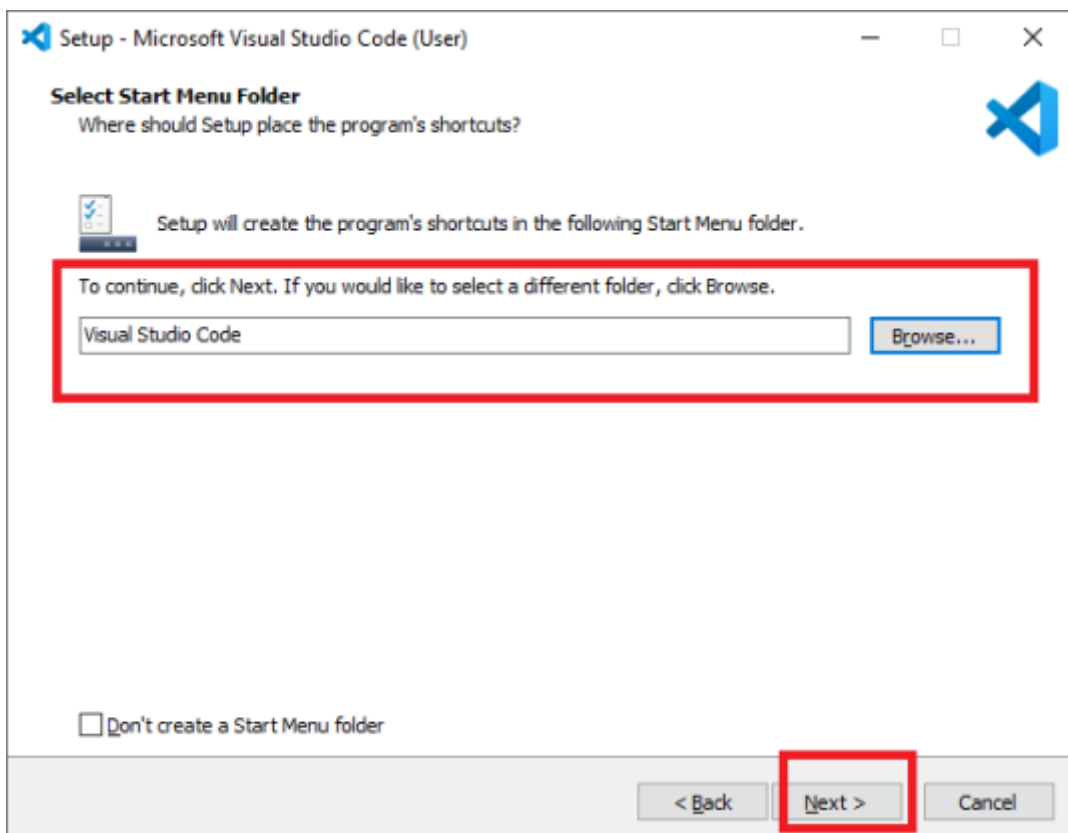
- Klik versi yang kalian butuhkan, terdapat tiga yang bisa dijalankan di desktop yaitu Windows, Linux, dan Macbook
- Buka folder VSCode yang sudah di download
- Klik dua kali folder tersebut, Setelah itu akan ada noted Setup - Microsoft Visual Studio Code (User) License Agreement atau perjanjian lisensi yang bahwasanya setuju menginstall VSCode dengan persyaratan yang telah ditentukan
- I Klik "I accept the agreement" lalu klik next



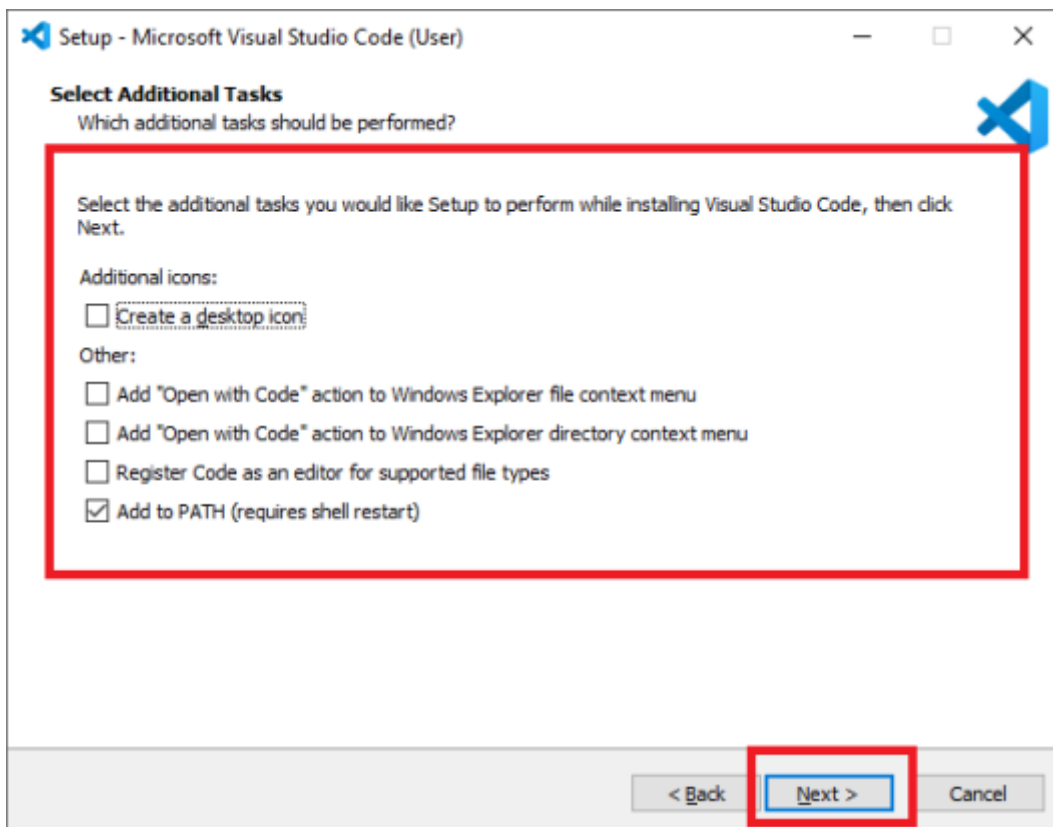
- Selanjutnya akan ada Select Destination Location atau memilih lokasi tujuan. Jadi folder tersebut akan diletakkan di mana



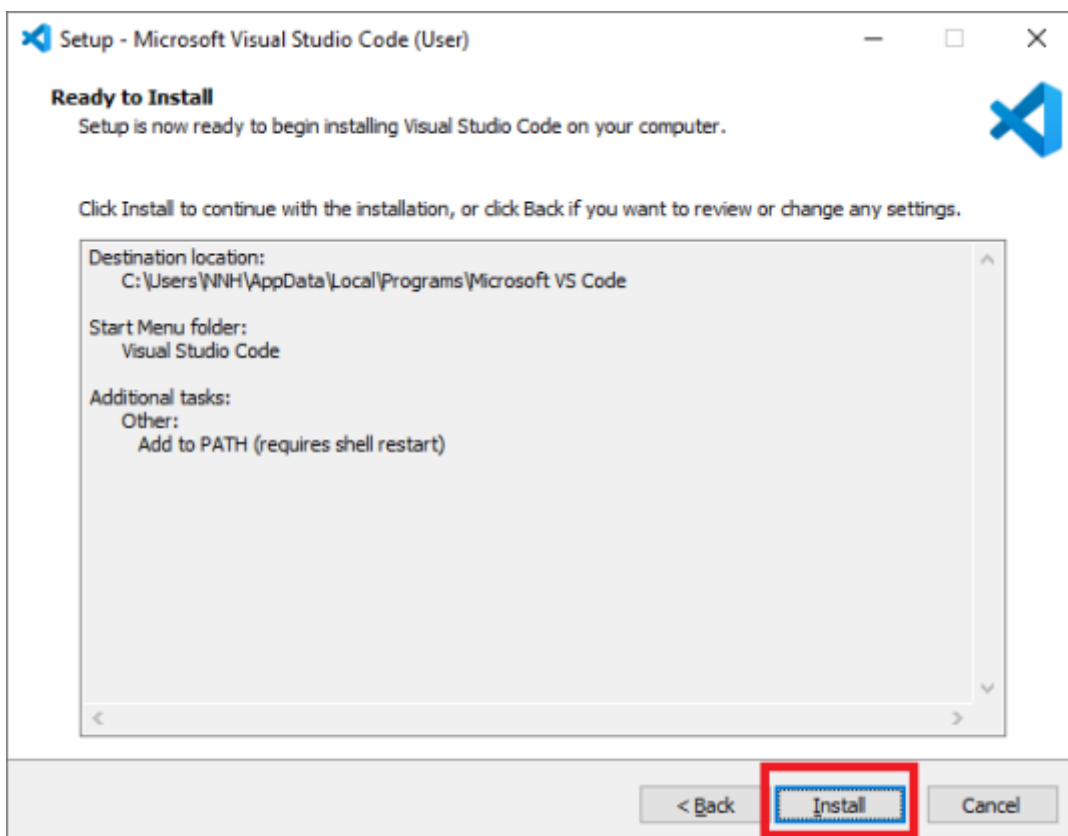
- Klik 'Next' pada tahap ini



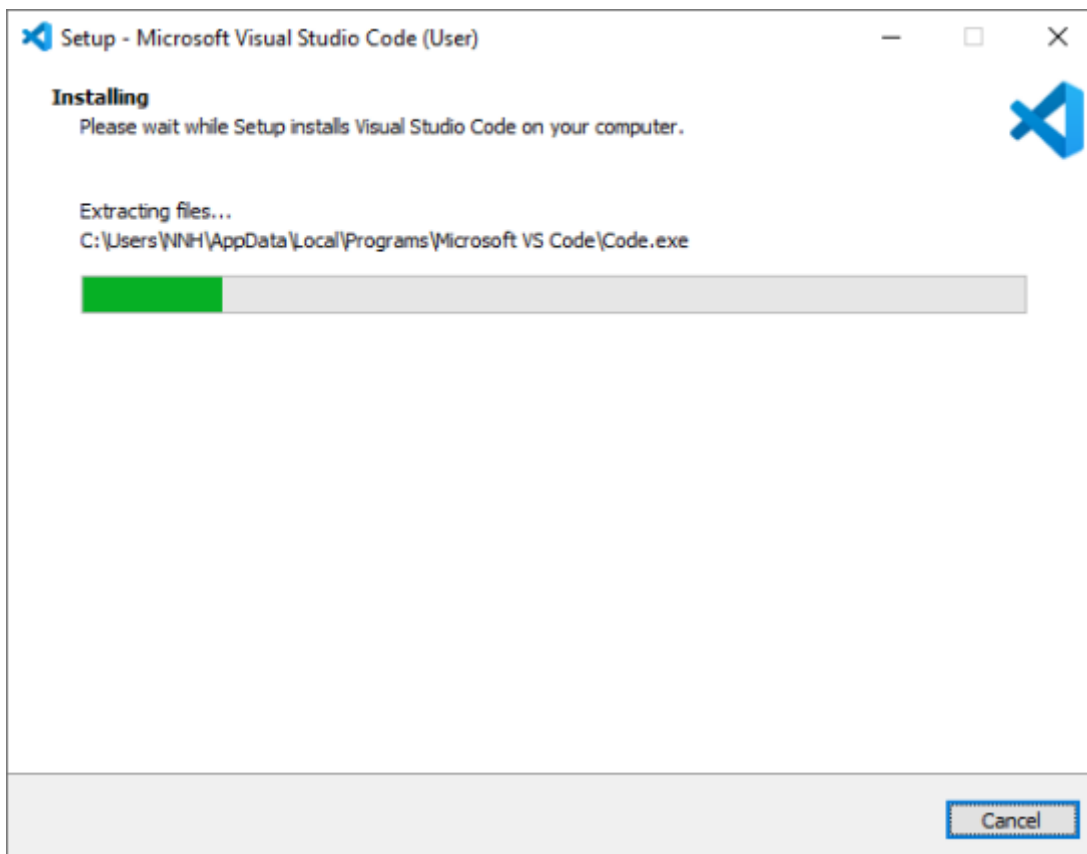
- Pilih optional secara default, kemudian klik 'Next'



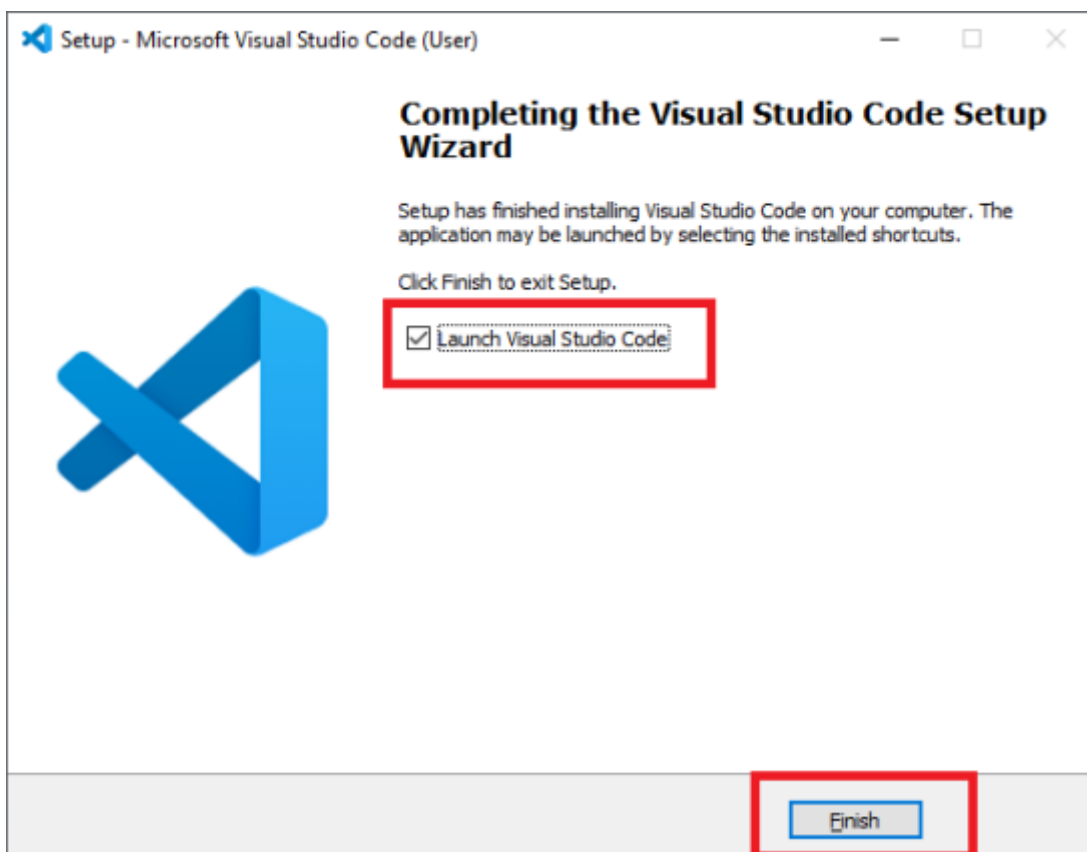
- Klik 'Install'



- Tunggu proses instalasi hingga selesai.



-
- Centang 'Launch Visual Studio Code' jika anda ingin langsung membuka aplikasinya, kemudian klik 'Finish'.



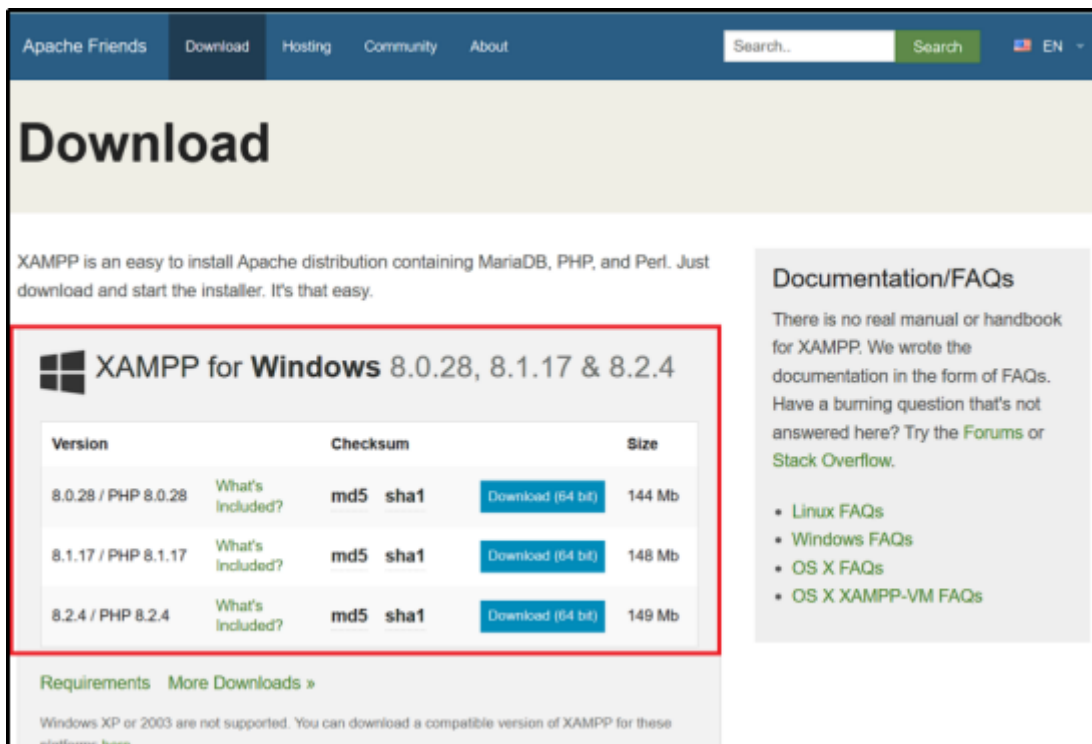
Webserver dan Database

Yang terpenting dalam instalasi Laravel yaitu bahwa versi PHP minimal versi ≥ 5.4 dan Sekarang Laravel telah mencapai versi 11. Pastikan Anda memiliki PHP versi 8.1 atau yang lebih baru untuk dapat menggunakan Laravel versi terbaru.

Sebagai contoh, jika Anda menggunakan XAMPP, pastikan versi XAMPP yang Anda gunakan mendukung PHP 8.1 atau Anda dapat menginstal PHP versi 8.1 secara terpisah.

Berikut Langkah-langkah Instalasi XAMPP :

1. Buka situs resmi dari *software* XAMPP : <https://www.apachefriends.org/download.html>
2. Pilih sesuai dengan sistem operasi yang anda gunakan. Di situs tersebut tersedia 3 sistem operasi yang mendukung XAMPP diantaranya windows, linux dan OS X
3. Dalam hal ini karena komputer saya menggunakan sistem operasi Windows, maka saya akan pilih versi XAMPP for windows



Apache Friends Download Hosting Community About Search.. Search EN

Download

XAMPP is an easy to install Apache distribution containing MariaDB, PHP, and Perl. Just download and start the installer. It's that easy.

XAMPP for Windows 8.0.28, 8.1.17 & 8.2.4

Version	Checksum	Size
8.0.28 / PHP 8.0.28	What's Included? md5 sha1	Download (64 bit) 144 Mb
8.1.17 / PHP 8.1.17	What's Included? md5 sha1	Download (64 bit) 148 Mb
8.2.4 / PHP 8.2.4	What's Included? md5 sha1	Download (64 bit) 149 Mb

Requirements More Downloads »

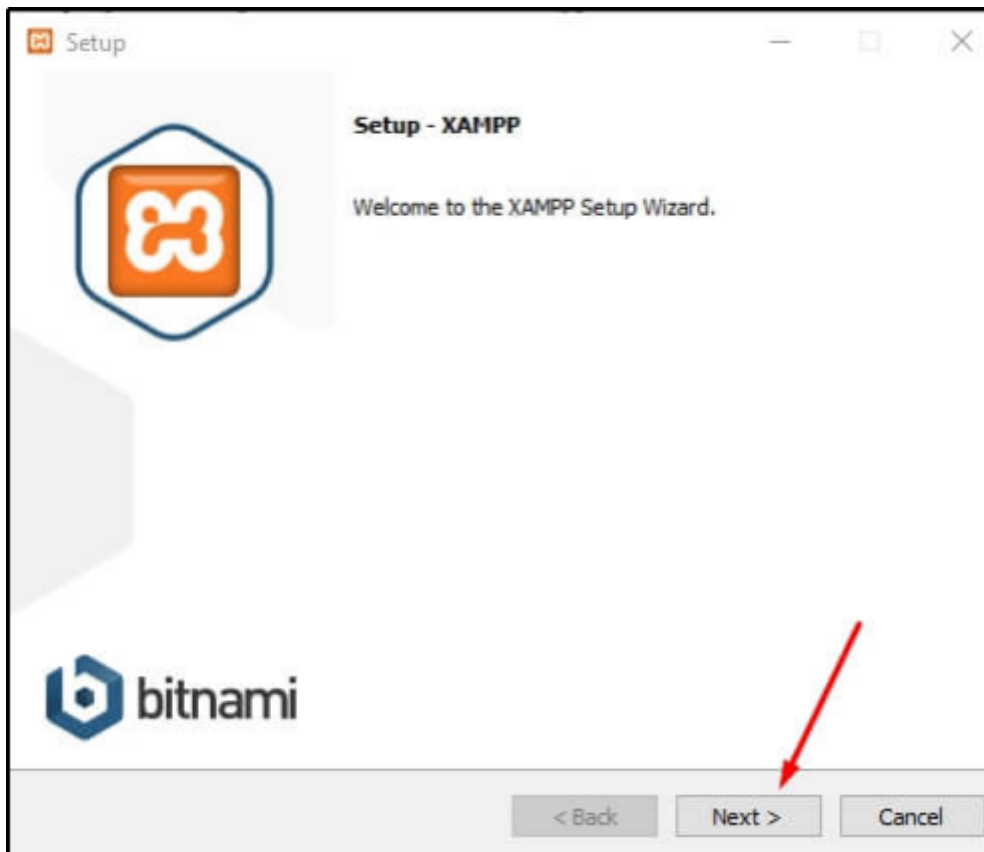
Windows XP or 2003 are not supported. You can download a compatible version of XAMPP for these platforms [here](#).

Documentation/FAQs

There is no real manual or handbook for XAMPP. We wrote the documentation in the form of FAQs. Have a burning question that's not answered here? Try the [Forums](#) or [Stack Overflow](#).

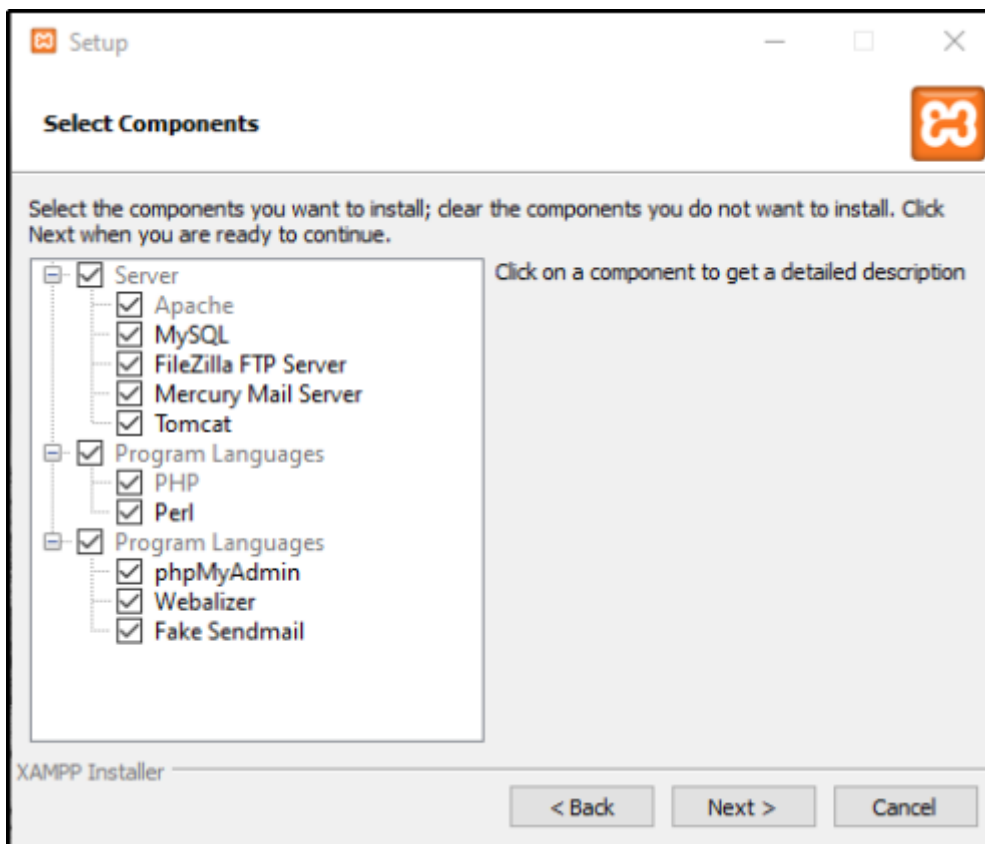
- [Linux FAQs](#)
- [Windows FAQs](#)
- [OS X FAQs](#)
- [OS X XAMPP-VM FAQs](#)

- 4.
5. Tekan tombol download pada salah satu versi yang dapat kalian pilih. disini saya memilih versi 8.1.17. Karena komputer saya menggunakan sistem operasi 64 bit maka muncul tombol download (64-bit) sementara jika sistem operasi windows kalian versi 32 bit silahkan disesuaikan.
6. Buka folder Xampp yang sudah di download, Kemudian klik kanan dan run as administrator
7. Selanjutnya klik 'Next'



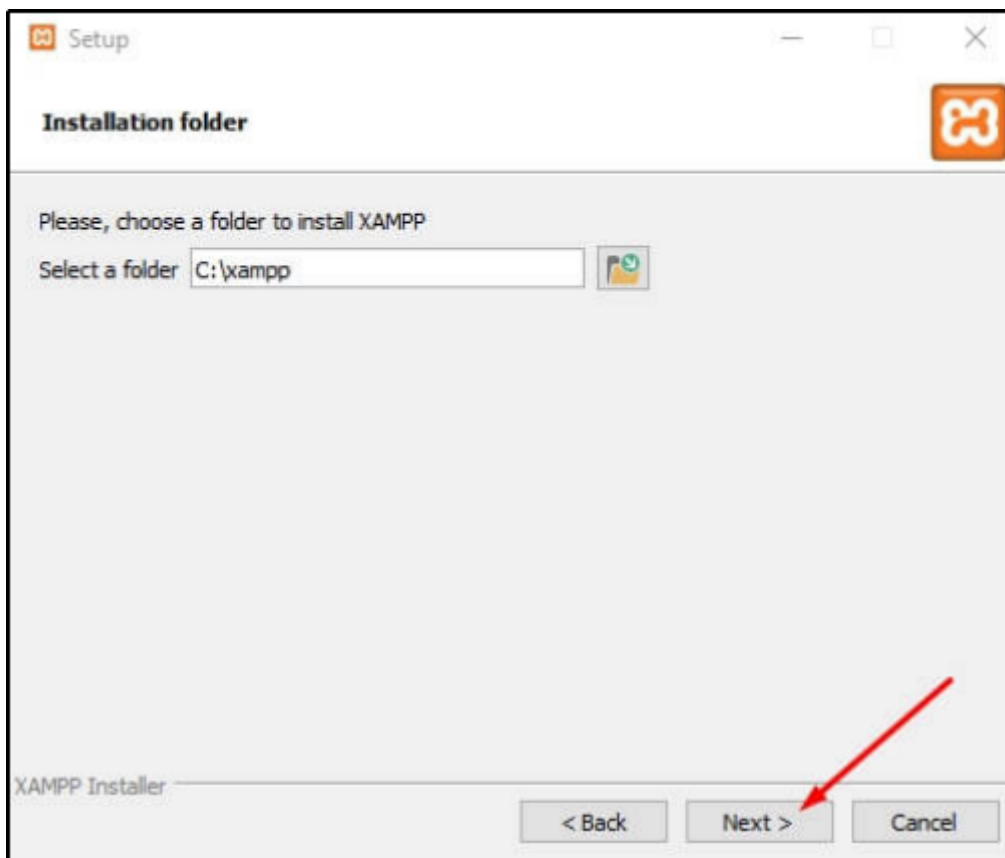
8.

9. Pada tahapan ini, Anda akan diminta untuk memilih aplikasi yang mau diinstal. Centang saja semua pilihan dan klik tombol Next.



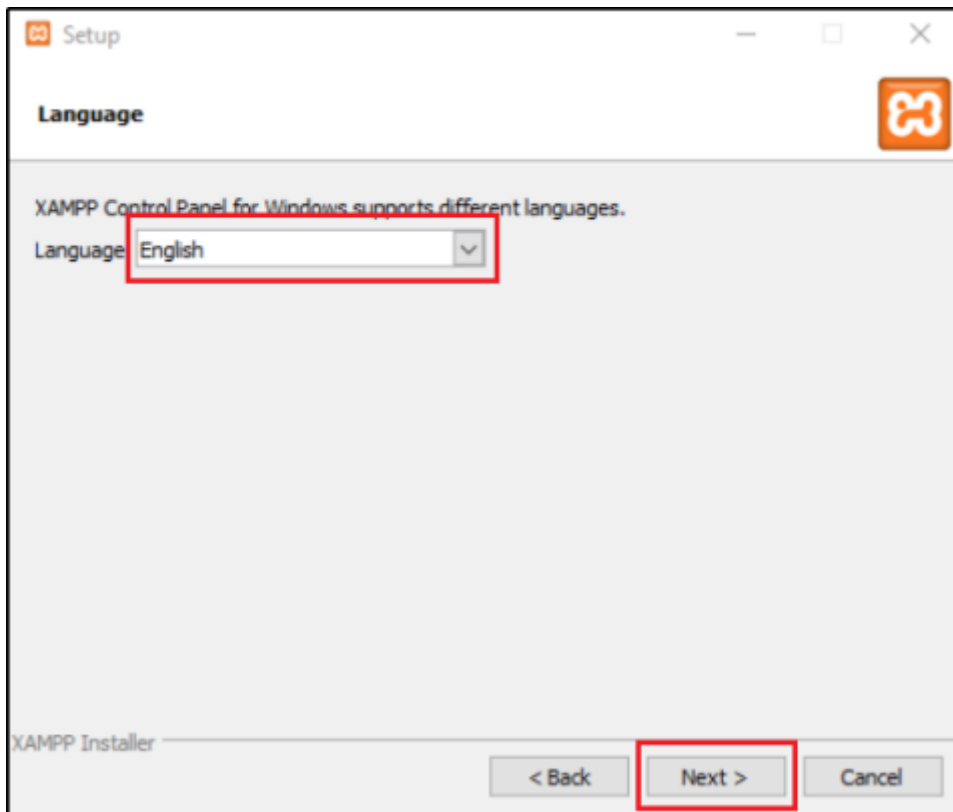
10.

11. Pilih folder instalasi XAMPP. Saya sarankan pilih default saja yaitu di C:/xampp, namun jika ingin menyimpannya di partisi lain juga tidak masalah. Lalu Klik Next



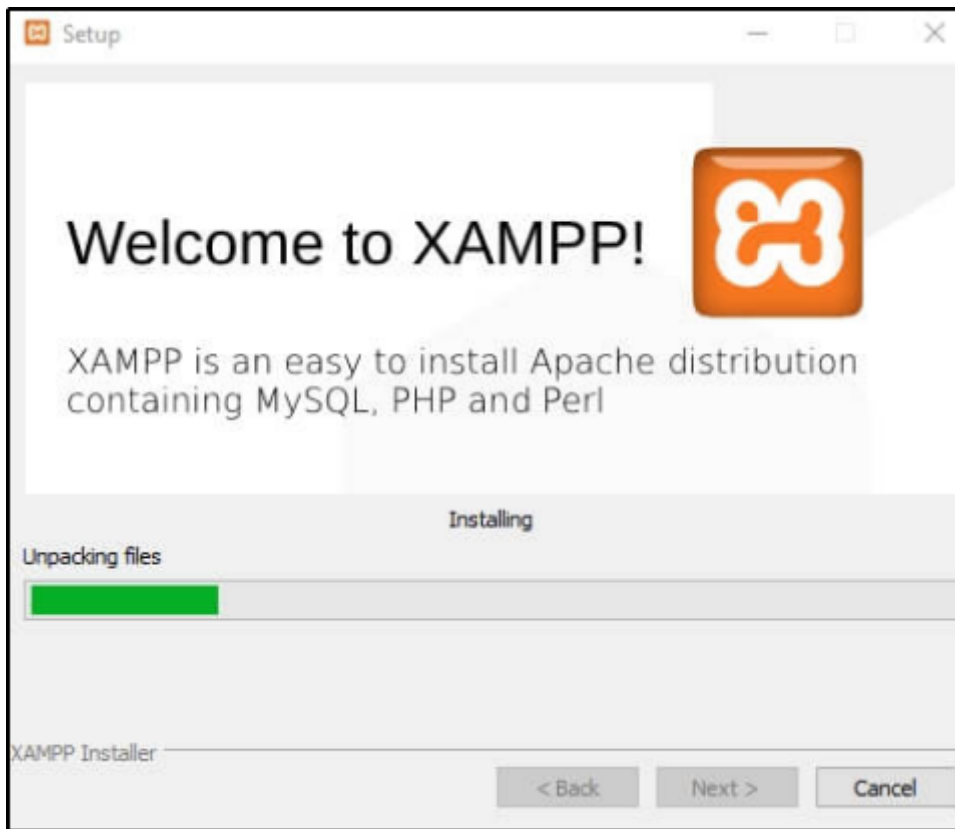
12.

13. Pilih bahasa English saja, lalu Klik Next



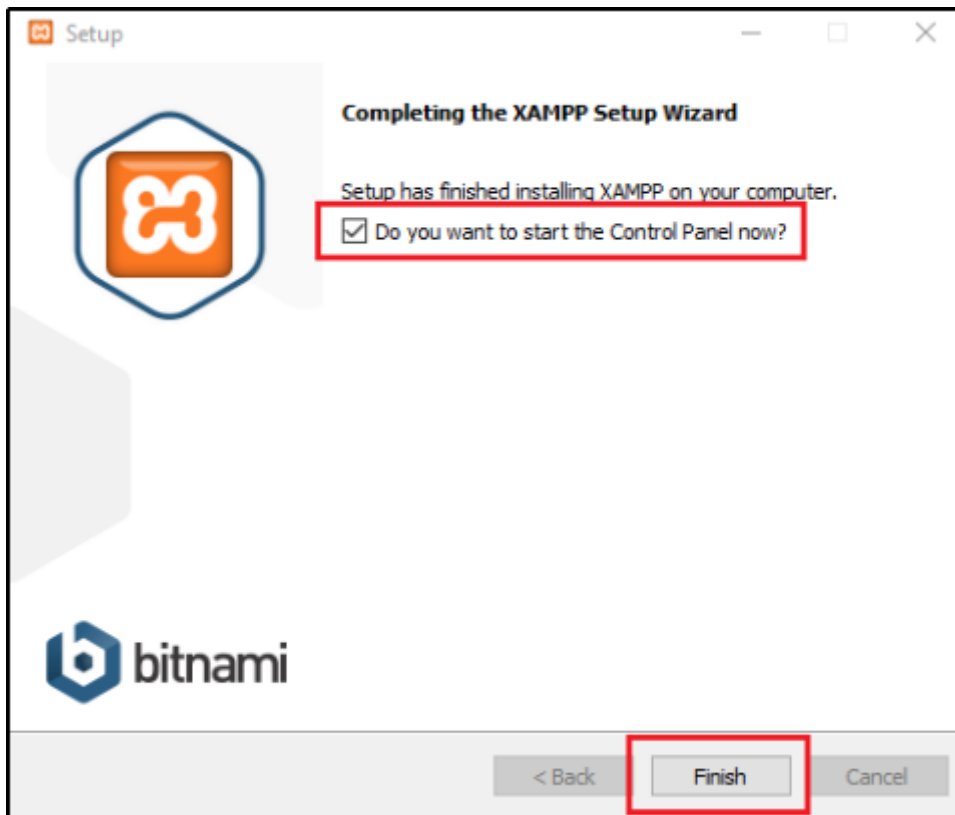
14.

15. Tunggu hingga proses install XAMPP selesai



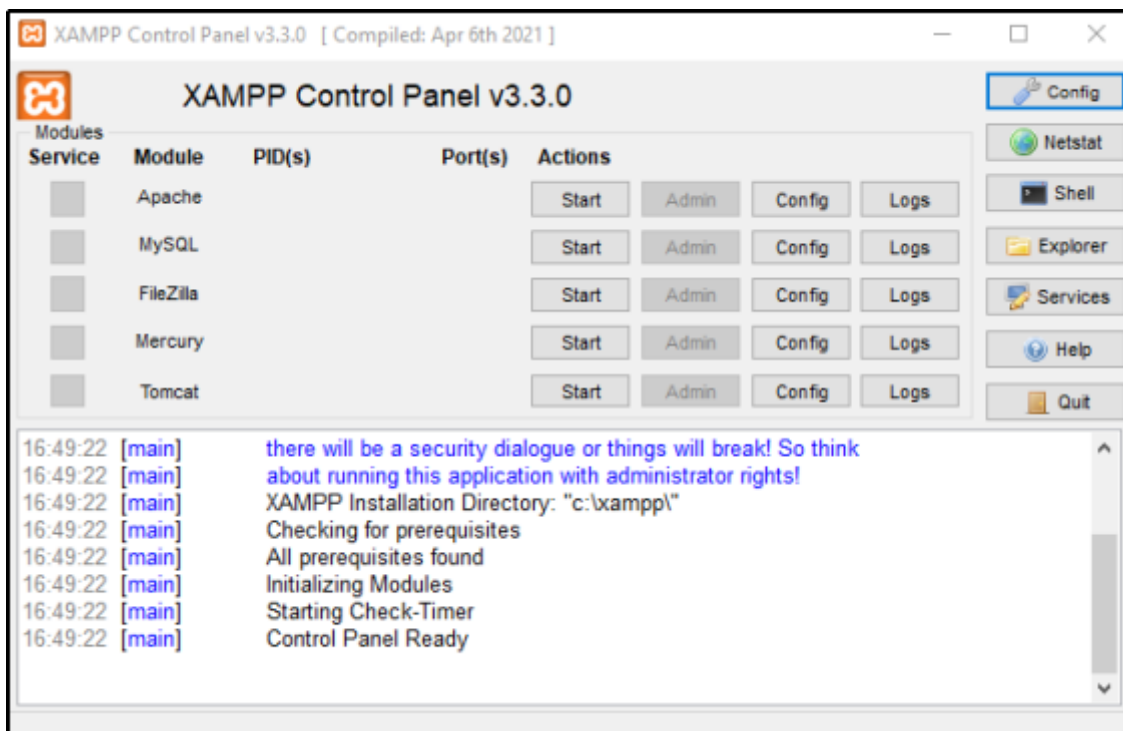
16.

17. Jika sudah muncul jendela seperti di bawah ini, klik tombol Finish untuk menyelesaikannya. Selain itu, akan muncul opsi apakah Anda mau langsung menjalankan aplikasi XAMPP atau tidak. Jika ya, maka centang opsi tersebut.



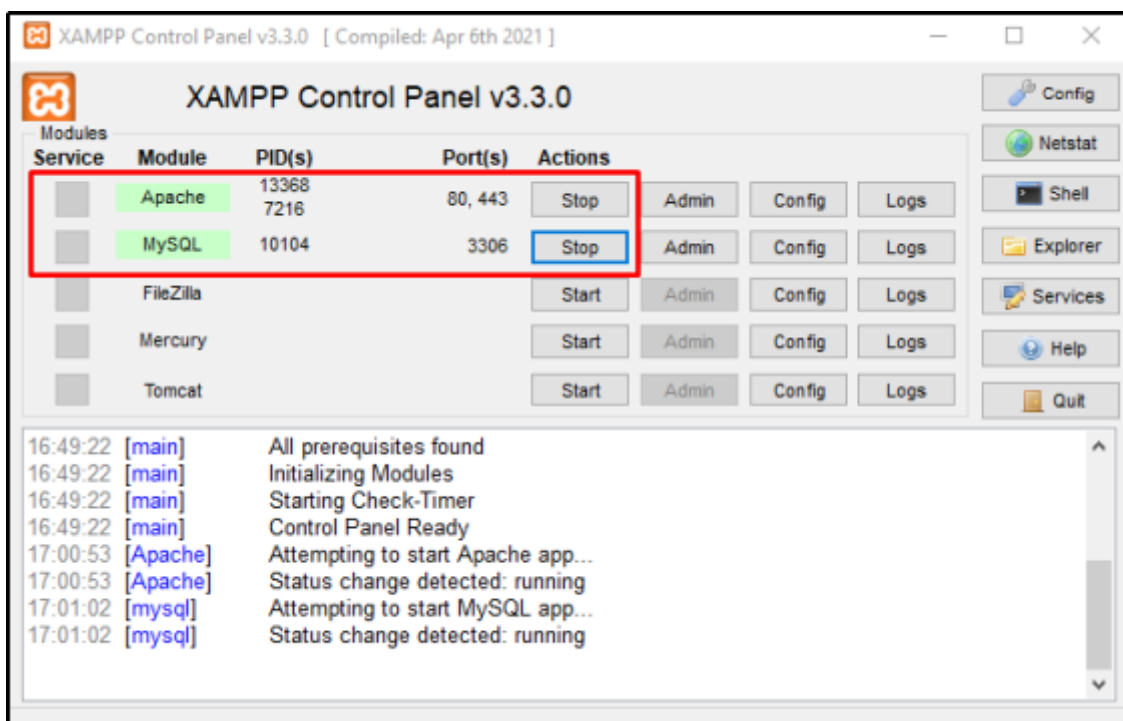
18.

19. Bukalah aplikasi XAMPP, bisa melalui Start Menu atau Desktop, dan klik icon XAMPP.



20.

21. Setelah terbuka, silahkan klik tombol Start pada kolom Action sehingga tombol tersebut berubah menjadi Stop. Dengan mengklik tombol tersebut, artinya itulah aplikasi yang dijalankan. Biasanya jika saya menggunakan XAMPP, yang saya start hanyalah aplikasi Apache dan MySQL, karena saya tidak memerlukan aplikasi seperti Filezilla, dan lain-lain.



22.

23. Sekarang bukalah browser kesukaan Anda, dan coba ketikkan "**localhost**" di address bar. Jika muncul tampilan seperti gambar di bawah ini, instalasi telah berhasil.

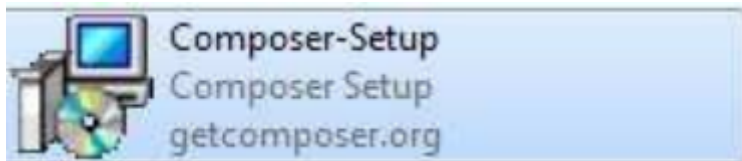


24.

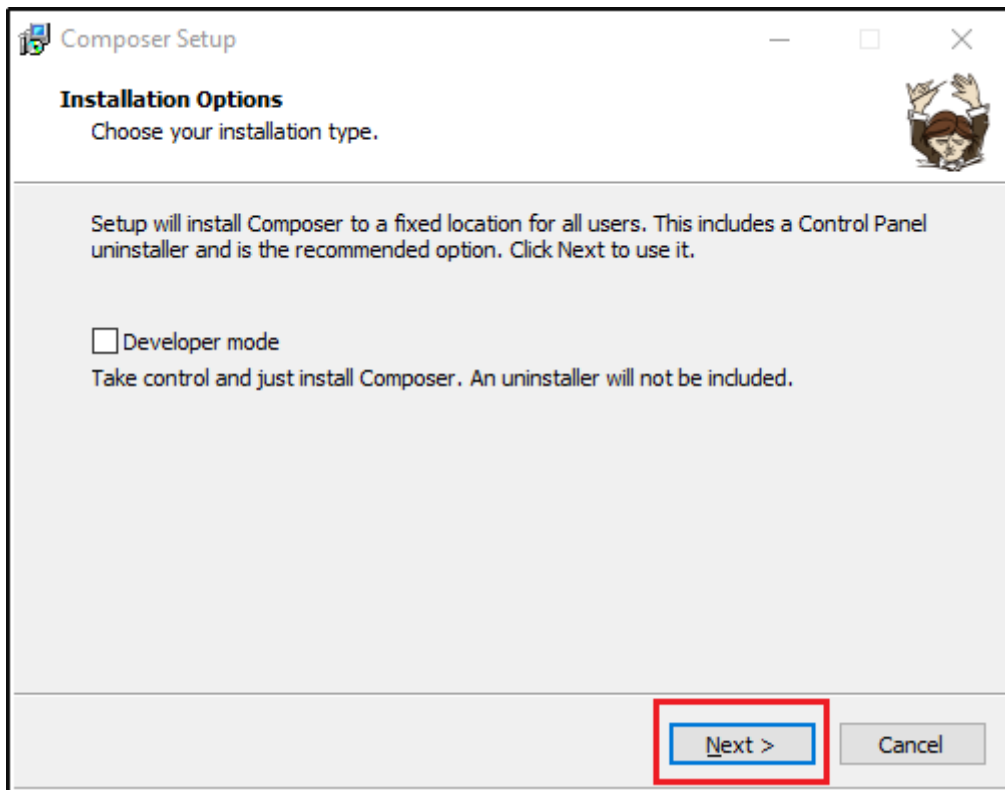
Composer

Untuk dapat menginstal laravel kita akan menggunakan composer. Composer adalah package manager untuk PHP yang mengelola dependensi dalam proyek secara efisien, termasuk dalam proyek Laravel.. Adapun proses instalasi composer adalah sebagai berikut :

1. Unduh composer di : <https://getcomposer.org/Composer-Setup.exe>

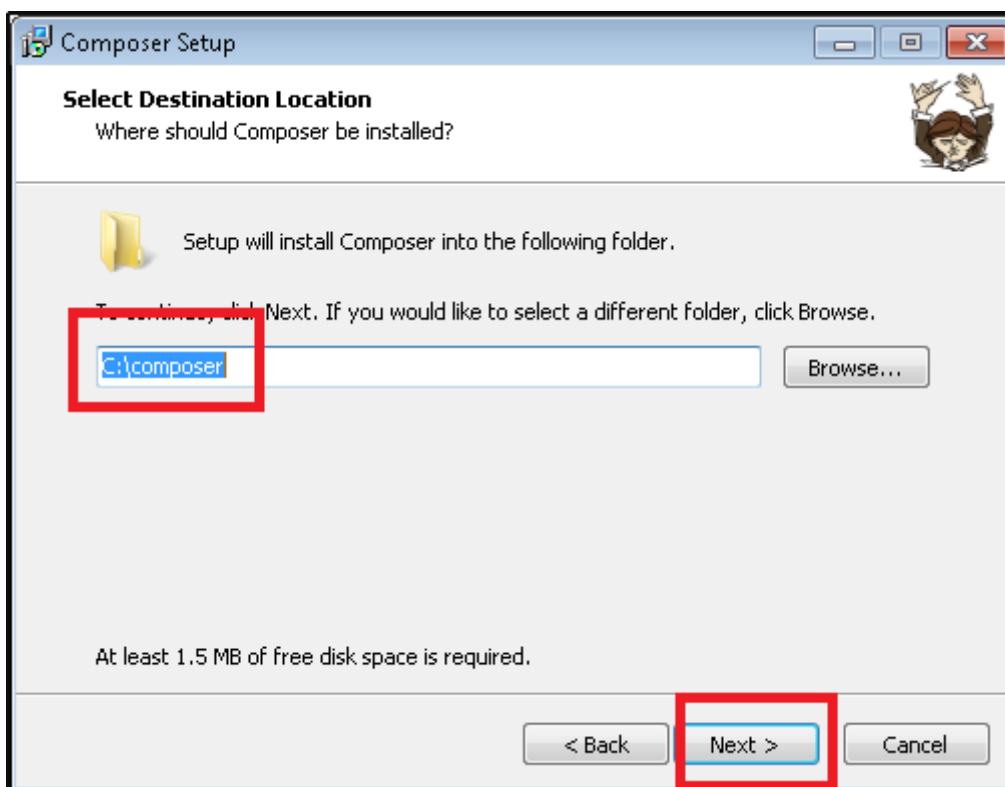


- 2.
3. Setelah file installer berhasil didownload, lakukan penginstallan dengan cara double klik file installer composer.



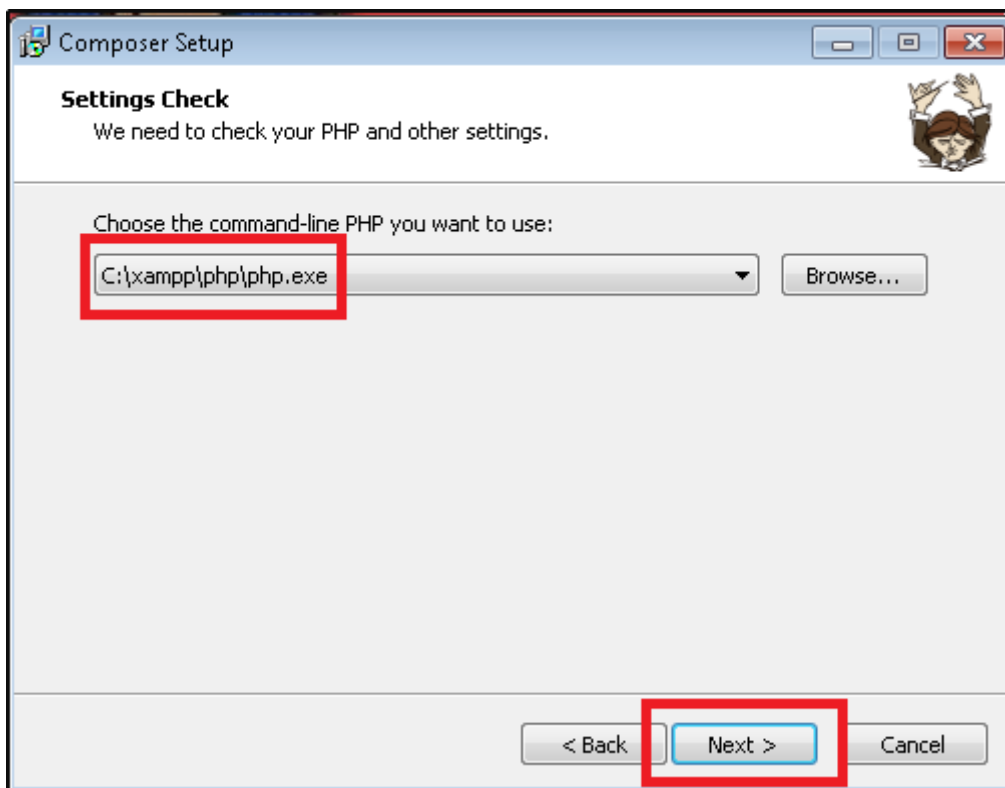
4.

5. Tentukan folder installer untuk composer di komputer, biasanya di C:\Composer.



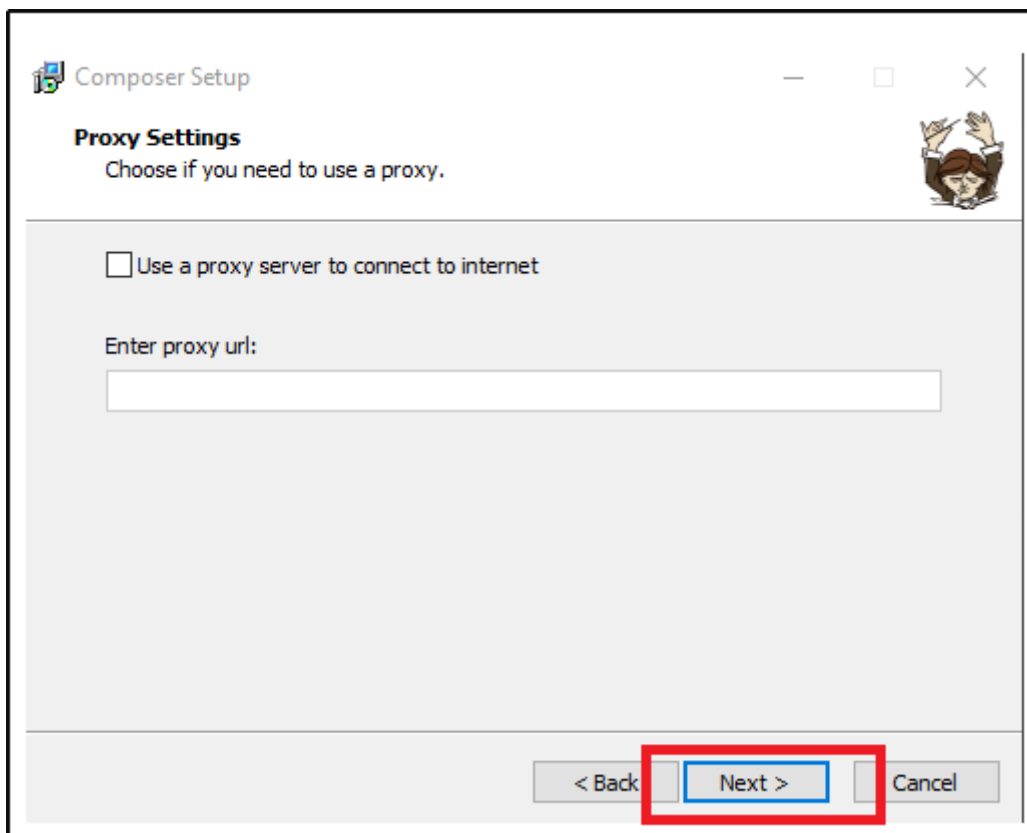
6.

7. Kemudian pilih lokasi PHP yang sudah terinstall. Jika Anda menggunakan XAMPP, lokasi file ini akan berada di C:\xampp\php\php.exe. Klik Next.



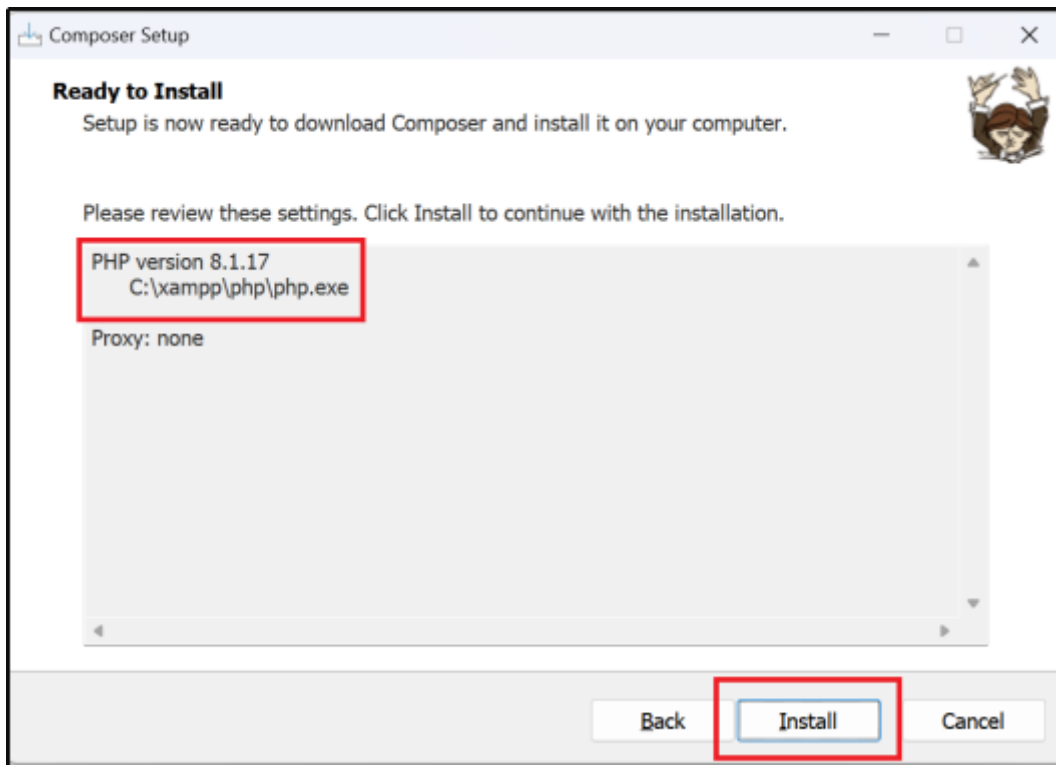
8.

9. Kemudian Anda akan diminta untuk memilih, apakah Anda akan menggunakan proxy atau tidak. Jika Anda ingin menggunakan proxy, klik centang dan masukkan URL proxy Anda. Apabila Anda tidak ingin menggunakan proxy, langsung klik Next untuk melanjutkan instalasi.



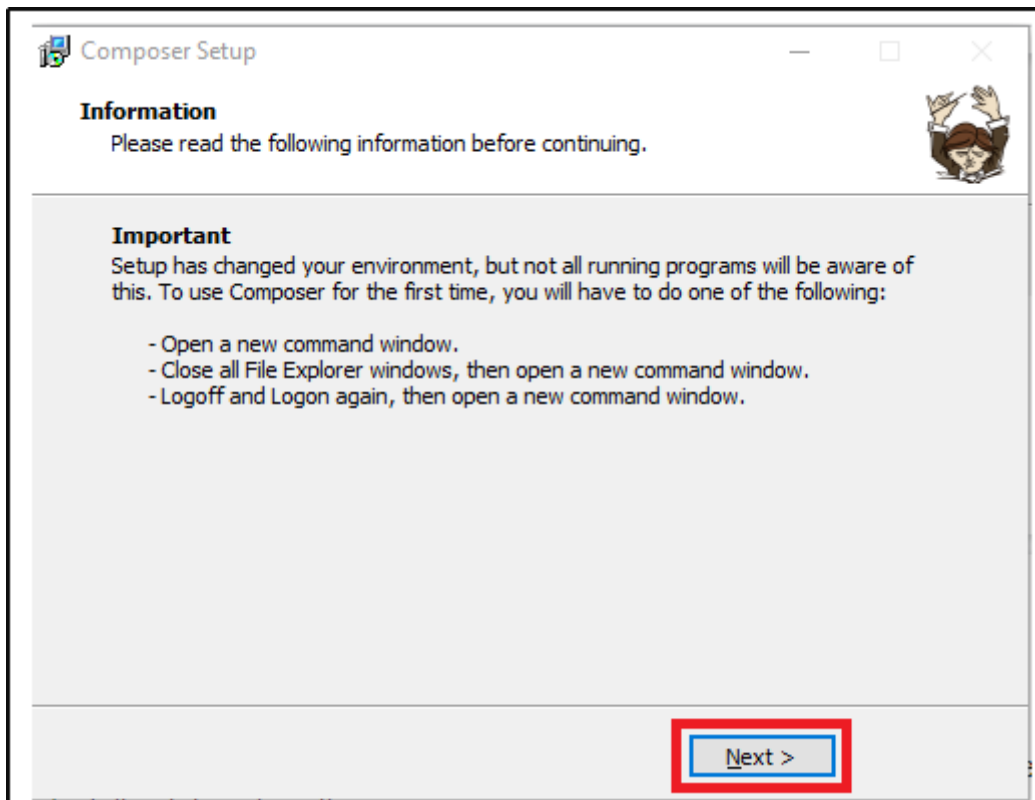
10.

11. Selanjutnya pastikan kalau proses instalasi berjalan di lokasi file yang seharusnya, yaitu C:\xampp\php\php.exe. Jika sudah benar, klik Install.



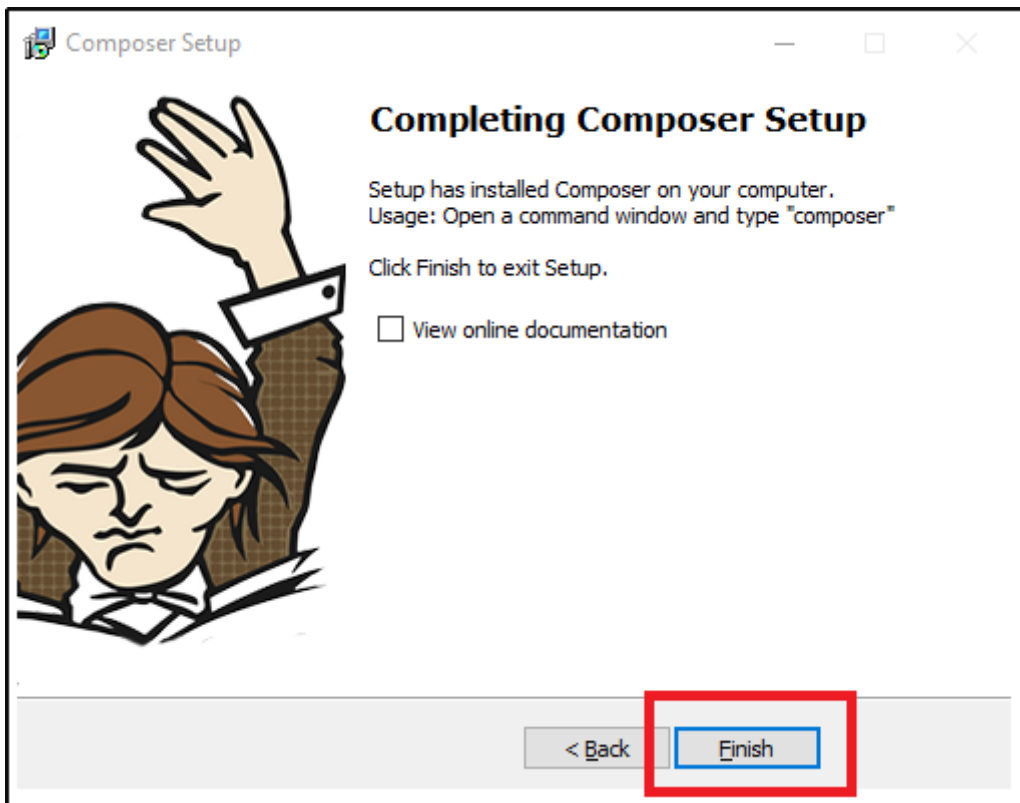
12.

13. Berikutnya adalah tampilan pemberitahuan bahwa telah terjadi perubahan pada Windows environment. Perubahan ini dimaksudkan agar Composer dapat dijalankan pada Command Prompt. Klik Next.



14.

15. Proses instalasi selesai, klik Finish untuk menutup jendela instalasi Composer.



16.

17. Untuk mengecek apakah composer berhasil di install, lakukan langkah-langkah berikut. Silahkan masuk ke terminal atau klik windows+R, kemudian ketikkan cmd kemudian Enter. Kemudian masukkan perintah composer. Jika composer berhasil diinstall akan muncul tampilan seperti gambar berikut.

```
Microsoft Windows [Version 10.0.22621.1928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LENOVO>composer

Composer version 2.5.8 2023-06-09 17:13:21

Usage:
  command [options] [arguments]

Options:
  -h, --help                Display help for the given command
  -q, --quiet                Do not output any message
  -V, --version              Display this application version
      --ansi|--no-ansi       Force (or disable) ANSI output
  -n, --no-interaction       Do not ask any interactive questions
      --profile              Display timing and memory usage
      --no-plugins            Whether to disable plugins.
      --no-scripts           Skips the execution of all scripts
  -d, --working-dir=WORKING-DIR If specified, use the given directory
      --no-cache             Prevent use of the cache
  -v|vv|vvv, --verbose      Increase the verbosity of messages
```

18.

Install Package Laravel

1. Untuk mengunduh paket laravel dapat menggunakan yaitu :mendownload langsung menggunakan composer langsung.

2. Buka Command Prompt kemudian arahkan direktori ke htdoc XAMPP yang sudah di instal. kemudian ketikan sintak berikut :

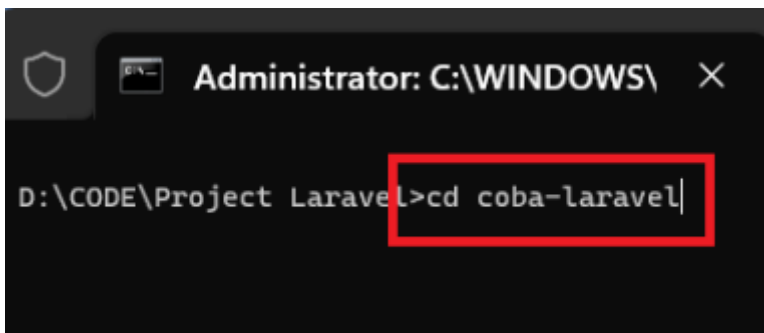
```
composer create-project laravel/laravel coba-laravel
```

3. Tunggu sampai proses unduh berhasil.



```
- Installing sebastian/object-enumerator (5.0.0): Extracting archive
- Installing sebastian/global-state (6.0.1): Extracting archive
- Installing sebastian/exporter (5.0.0): Extracting archive
- Installing sebastian/environment (6.0.1): Extracting archive
- Installing sebastian/diff (5.0.3): Extracting archive
- Installing sebastian/comparator (5.0.0): Extracting archive
- Installing sebastian/code-unit (2.0.0): Extracting archive
- Installing sebastian/cli-parser (2.0.0): Extracting archive
- Installing phpunit/php-timer (6.0.0): Extracting archive
- Installing phpunit/php-text-template (3.0.0): Extracting archive
- Installing phpunit/php-invoker (4.0.0): Extracting archive
- Installing phpunit/php-file-iterator (4.0.2): Extracting archive
- Installing theseer/tokenizer (1.2.1): Extracting archive
- Installing sebastian/lines-of-code (2.0.0): Extracting archive
- Installing sebastian/complexity (3.0.0): Extracting archive
- Installing sebastian/code-unit-reverse-lookup (3.0.0): Extracting archive
- Installing phpunit/php-code-coverage (10.1.2): Extracting archive
- Installing phar-io/version (3.2.1): Extracting archive
- Installing phar-io/manifest (2.0.3): Extracting archive
- Installing myclabs/deep-copy (1.11.1): Extracting archive
- Installing phpunit/phpunit (10.2.6): Extracting archive
- Installing spatie/backtrace (1.5.3): Extracting archive
- Installing spatie/flare-client-php (1.4.1): Extracting archive
- Installing spatie/ignition (1.9.0): Extracting archive
- Installing spatie/laravel-ignition (2.2.0): Extracting archive
60/108 [=====] 55%
```

- 4.
5. Jika proses unduh telah selesai, kemudian buka folder yang tadi sudah di download dengan mengetikkan cd {nama folder} contoh yang sudah dibuat “cd coba-laravel”



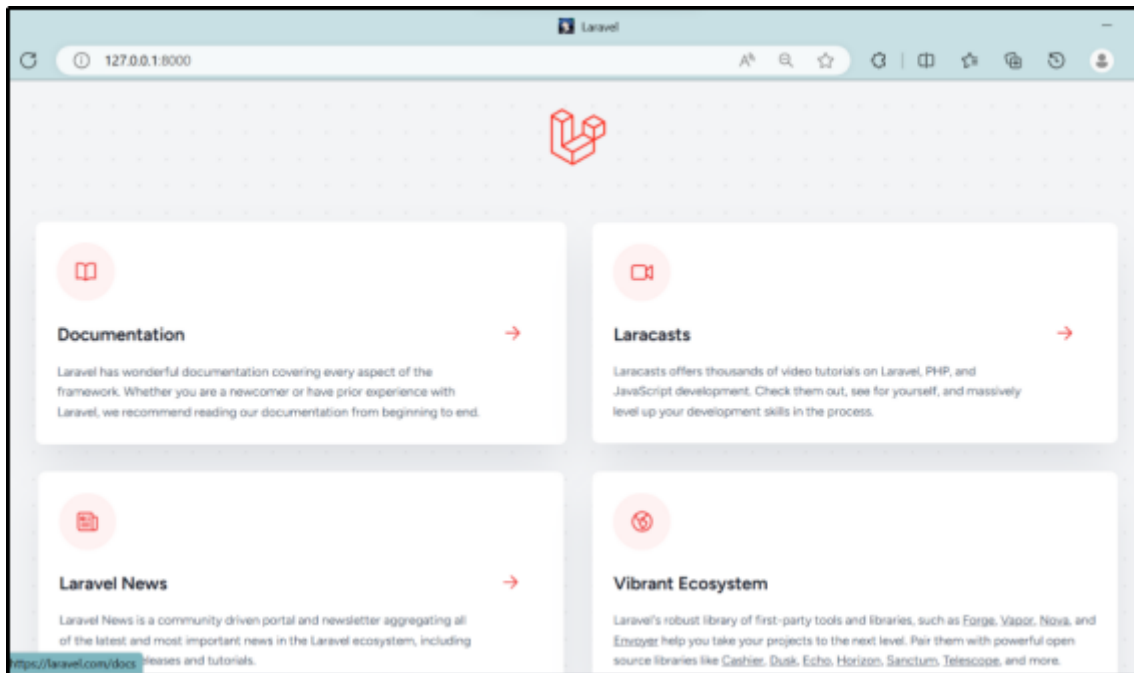
```
Administrator: C:\WINDOWS\
D:\CODE\Project Laravel>cd coba-laravel|
```

- 6.
7. Untuk mengecek apakah laravel sudah terinstal yaitu dengan cara jalankan perintah php artisan serve untuk menjalankan sebagai server. Secara default url nya adalah http://localhost:8000. Untuk mengubah port tinggal menjalankan perintah php artisan serve --port={port}. Kalian bisa ubah {port} sesuai keinginan kalian.


```
: \CODE\Project Laravel\coba-laravel >php artisan serve  
INFO Server running on [http://127.0.0.1:8000].  
Press Ctrl+C to stop the server
```

8.

9. Buka browser kemudian ketik localhost:8000. Jika tampilan sebagai berikut, berarti laravel yang diinstal sudah berjalan.



10.

Struktur Proyek Laravel

Jika kita buka folder laravel tersebut maka kita akan menemukan folder-folder dan file sebagai berikut :

- app/
- bootstrap/
- config/
- database/
- public/
- resources/
- routes/
- storage/
- test/
- vendor/
- .editorconfig

- `.env`
- `.env.example`
- `.gitattributes`
- `.gitignore`
- `artisan`
- `composer.json`
- `composer.lock`
- `package.json`
- `phpunit.xml`
- `README.md`
- `vite.config.js`

berikut adalah penjelasan struktur-struktur proyek laravel diatas .

App/

Folder app berisi kode-kode inti dari aplikasi seperti Model, Controller, Commands, Listener, Events, dll. Poinnya, hampir semua class dari aplikasi berada di folder ini.

Bootstrap/

Folder ini berisi berbagai file yang bertanggung jawab untuk menginisialisasi lingkungan aplikasi dan mengatur konfigurasi awal sebelum aplikasi Laravel benar-benar dijalankan. yang digunakan untuk meningkatkan kinerja aplikasi.

Config/

Folder config seperti namanya, berisi semua file konfigurasi aplikasi Anda.

Database/

Folder database berisi database migrations, model factories, dan seeds. Folder ini akan bertanggung jawab dengan pembuatan dan pengisian tabel-tabel database.

Public/

Folder public memiliki file `index.php` yaitu entry point dari semua requests yang masuk/diterima ke aplikasi. Folder ini juga tempat menampung gambar, Javascript, dan CSS.

Resources/

Folder resources berisi semua route yang disediakan aplikasi. Sebagai default, beberapa file routing akan tersedia seperti: web.php, api.php, console.php, dan channels.php. Folder ini adalah tempat dimana kita memberikan koleksi definisi route aplikasi.

Routes/

folder "routes" berfungsi sebagai tempat untuk mendefinisikan berbagai rute (routes) atau aturan yang menentukan bagaimana aplikasi akan merespons permintaan HTTP yang masuk. Rute menentukan tindakan apa yang harus diambil oleh aplikasi ketika permintaan diterima dari URL tertentu.

Storage/

Folder storage adalah tempat dimana cache, logs, dan file sistem yang ter-compile hidup.

Test/

Folder tests adalah tempat dimana unit dan integration tests tinggal.

Vendor/

Folder "vendor" berisi dependensi pihak ketiga yang diinstal melalui komposer (Composer).

.editorconfig

Berguna untuk memberi IDE/text editor instruksi tentang standar coding Laravel seperti whitespace, besar identasi, dll.

.env & .env.example

Tempat dimana variable environment aplikasi ditempatkan (variabel yang diekspektasikan akan berbeda di setiap sistem) seperti nama database, username database, password database.

.gitattributes & .gitignore

File konfigurasi git.

Artisan

Memungkinkan anda untuk menjalankan perintah artisan dari command line.

Composer.json & Composer.lock

File konfigurasi untuk composer. File ini adalah informasi dasar tentang proyek dan juga mendefinisikan dependencies yang digunakan.

Package.json

Mirip-mirip dengan composer.json tapi untuk aset-aset dan dependencies front-end.

Phpunit.xml

Sebuah file konfigurasi untuk PHPUnit, tools yang digunakan Laravel untuk testing.

README.md

Sebuah markdown file yang memberikan pengenalan dasar tentang Laravel.

Mengenal Dasar Routing

Routing Dasar

Hal pertama yang akan kita pelajari yaitu tentang laravel pada materi ini adalah tentang routing, apa itu Routing ?

Sebelum beranjak ke materi yang detail tentang routing akan saya coba beri analogi tentang routing ini. Bayangkan jika anda sedang akan login ke facebook, hal pertama yang harus anda lakukan adalah menuliskan url facebook di web browser kemudian enter dan hasilnya akan muncul homepage login facebook. Jika anda mencari sesuatu di google, Anda menulis kata yang dicari di beranda Google kemudian google akan menampilkan hasil dari yang anda cari.

Nah dari analogi diatas, routing digunakan untuk meng-handle request yang kita berikan ke aplikasi web. Bisa jadi routing adalah jembatan yang menghubungkan kita dengan respon yang akan diberikan oleh web aplikasi. Jadi setiap ada permintaan (request) terhadap alamat tertentu, maka akan alamat akan dieksekusi terlebih dahulu dalam routing sebelum akhirnya akan menampilkan hasil (response).

Jika masih bingung tentang routing, akan kita coba langsung di dalam laravel ini.

Buka folder laravel yang telah di instal dengan text editor Anda, kemudian buka file `web.php` di folder `routes/web.php`. Berikut adalah isian dari `routes/web.php`.

`routes/web.php`

```
Route::get('/', function(){  
    return view("welcome");  
});
```

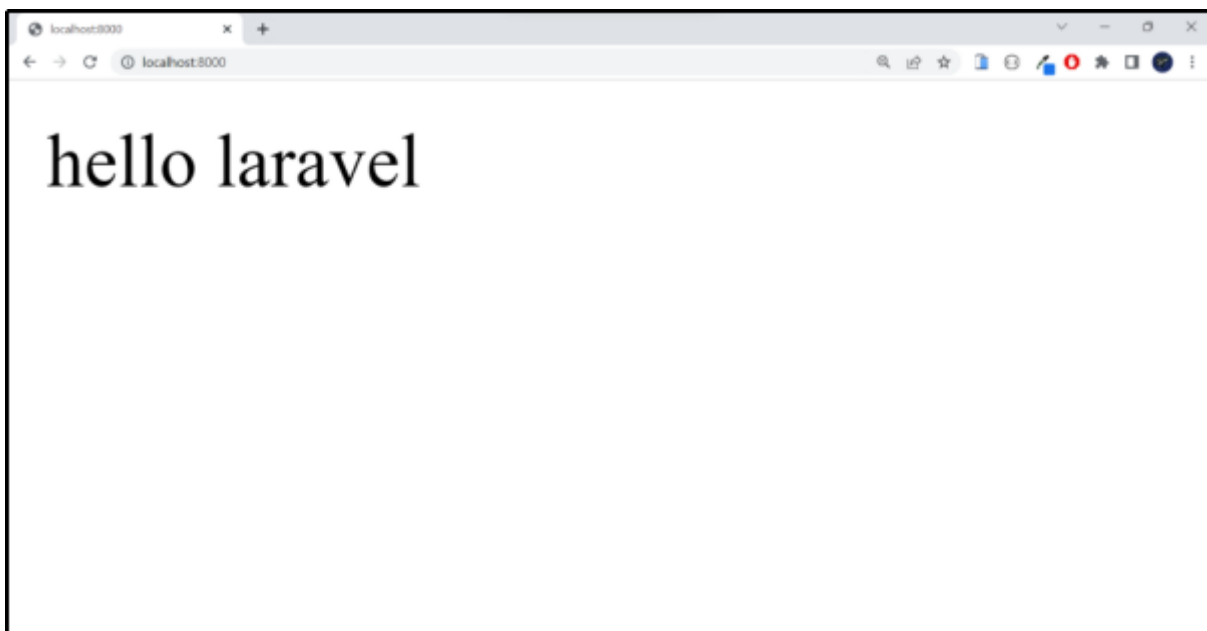
```
} )
```

Ganti respon route diatas menjadi

routes/web.php

```
Route::get('/', function(){  
    return 'Halo bang Laravel';  
})
```

Kemudian buka web browser dan ketikan alamat localhost:8000, kemudian hasilnya seperti gambar dibawah ini.

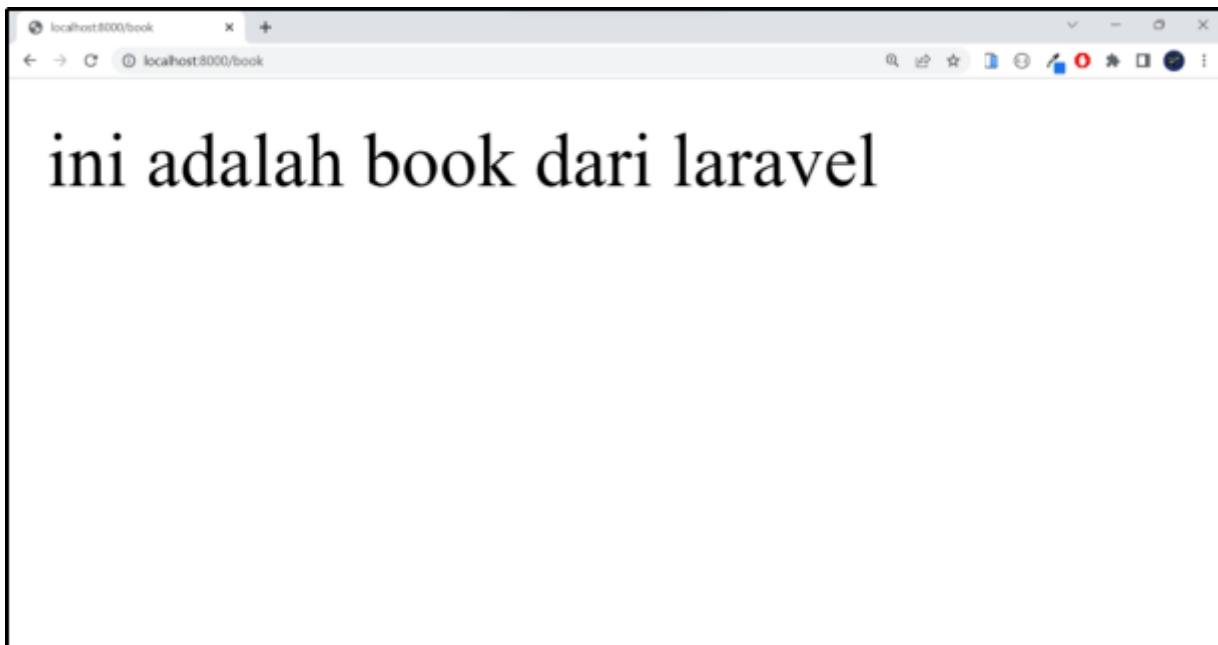


Gambar diatas adalah hasil (response) dari route yang telah kita manipulasi tadi. Kemudian kita akan coba membuat route baru dengan mengetikan sintak sebagai berikut.

routes/web.php

```
Route::get('/book', function(){  
    return 'ini adalah book dari laravel';  
});
```

Route diatas akan menghasilkan hasil sebagai berikut :



Routes selalu dideklarasikan menggunakan kelas Routes dan salah satu method yang dipakai untuk request sebuah halaman webpage yaitu GET menggunakan HTTP. GET request ini dikirim setiap waktu ketika kita mengetikkan sebuah alamat web di web browser.

Disamping method GET, ada juga method POST yang digunakan untuk membuat sebuah permintaan (request) dan menyediakan sebuah data yang relatif kecil. Normalnya method ini digunakan sebagai sebuah hasil submit dari form dimana data akan dikirimkan ke database tanpa ditampilkan ke URL.

Ada banyak method yang disediakan oleh kelas routes khususnya untuk restful, diantaranya.

- `Route::get();`
- `Route::post();`
- `Route::put();`
- `Route::delete();`
- `Route::any();`

Kita akan mempelajari method route tersebut di depan khususnya dengan yang berkaitan dengan RESTful routing pada saat proses CRUD (Create, Read, Update dan Delete)

Routing Berparameter

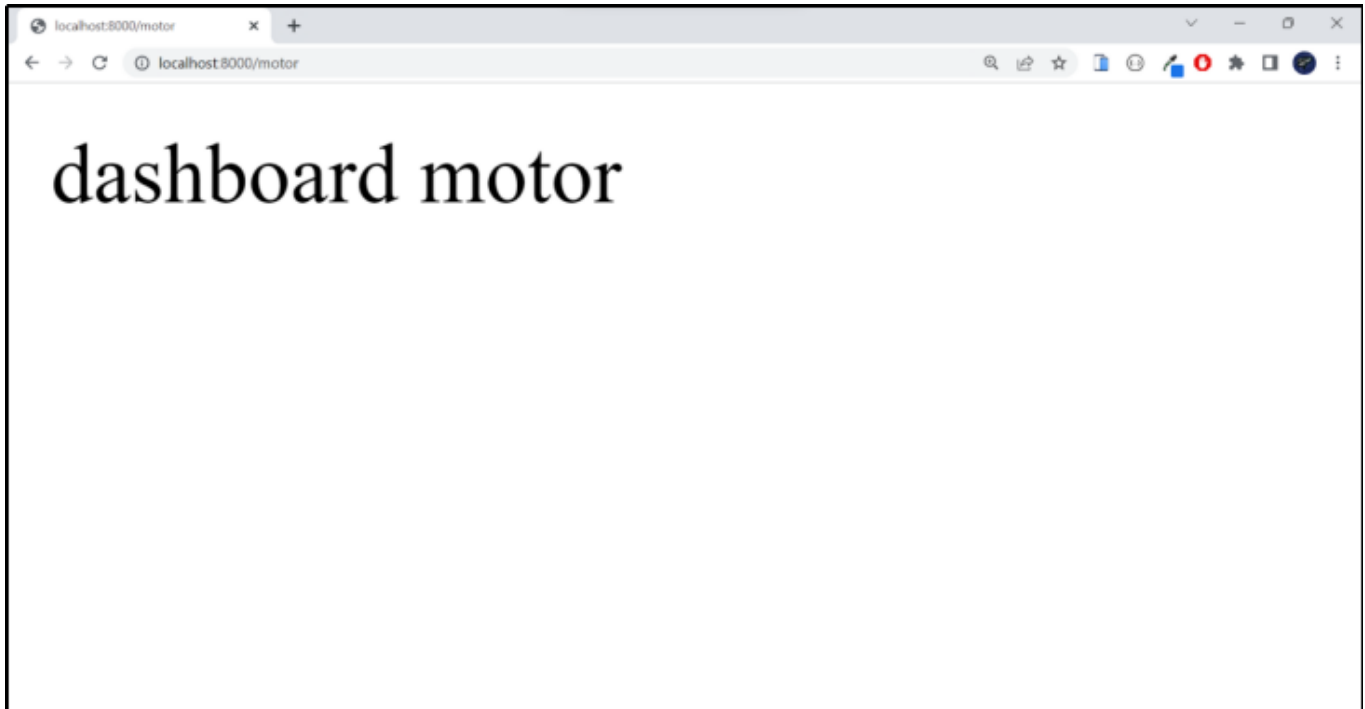
Route berparameter ini dapat digunakan untuk menempatkan sebuah nilai ke route atau URL yang digunakan untuk berbagai keperluan yang dibutuhkan nantinya.

Langsung kita coba, buka file `web.php` kemudian ketikkan Route baru berikut ini.

```
routes/web.php
```

```
Route::get('/motor', function(){  
    return 'dashboard motor';  
});
```

Route diatas akan menghasilkan hasil sebagai berikut :

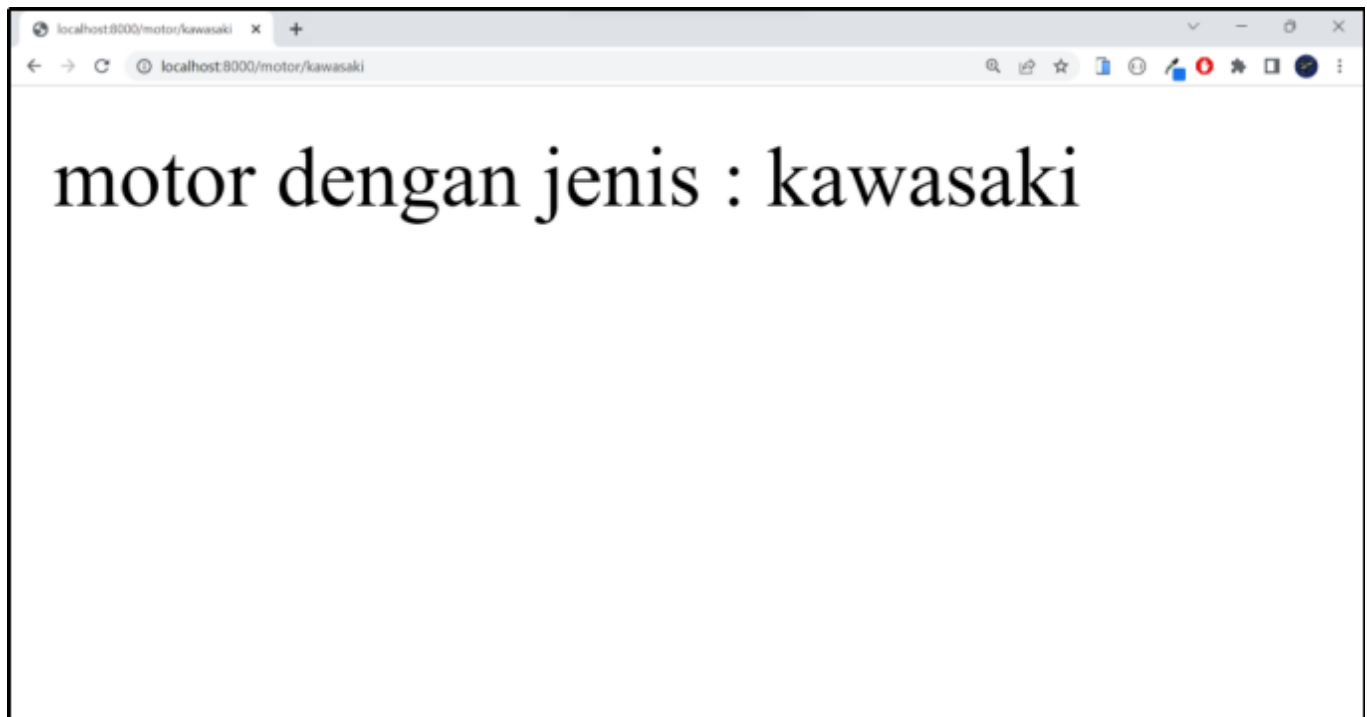


Masih di routes yang sama, kemudian kita akan membuat satu route yang mempunyai parameter yang berfungsi untuk mengirimkan sebuah nilai untuk ditampilkan ke web browser.

routes/web.php

```
Route::get('/motor/{jenis}', function($jenis){  
    return 'motor dengan jenis : ' . $jenis;  
});
```

Kemudian kita ketik URL di browser dan berikan nilai “Kawasaki” untuk route yang berparameter , localhost/motor/kawasaki dan hasilnya adalah sebagai berikut.



Kamu dapat mencoba dengan berbagai nilai untuk diberikan ke route diatas seperti:

- localhost:8000/motor/ninja
- localhost:8000/motor/nmax
- localhost:8000/motor/moge

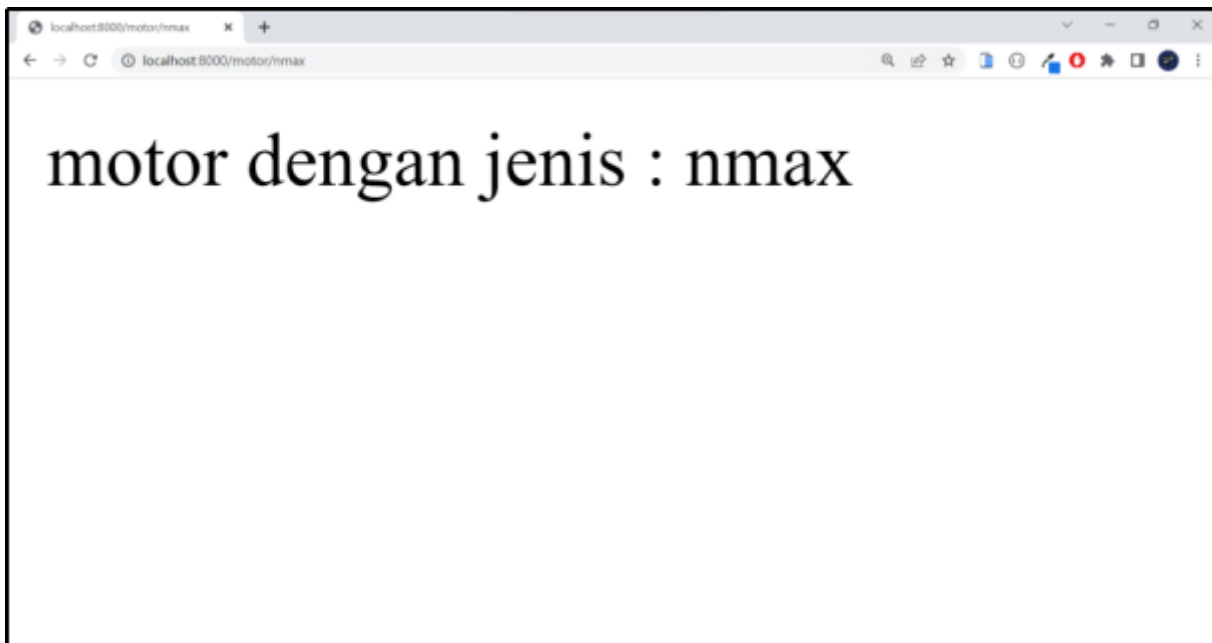
Dalam contoh diatas, kita mengirimkan sebuah nilai yaitu “Kawasaki” pada route berparameter `motor/{jenis}` dan dalam function route tersebut kita deklarasikan variabel `$jenis` untuk ditampilkan pada saat response.

Sebuah parameter juga dapat dijadikan sebuah pilihan jika parameter tersebut tidak diisi (null) atau diberi nilai default maka dapat ditambahkan sebuah tanya tanya (?)

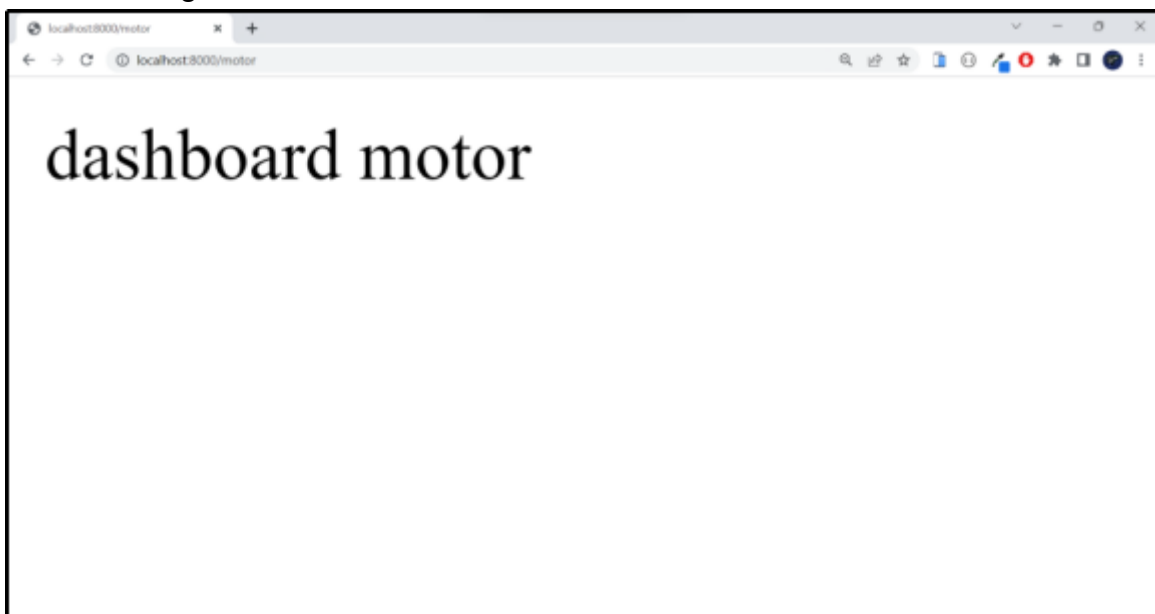
`routes/web.php`

```
Route::get('/motor/{jenis?}', function($jenis = null){  
    if($jenis == null) return "dashboard motor";  
    return 'motor dengan jenis : ' . $jenis;  
});
```

Jika route diatas ada fungsi logika nya yaitu jika nilai `$jenis = null` (kita tidak memberikan nilai di URL) maka akan mengembalikan "Motor Dashboard Page";.Tapi jika variabel `$jenis` kita beri nilai maka akan mengembalikan "Motor dengan jenis "<nilai_variabel>". Kita dapat lihat pada contoh sebagai berikut.



Tapi kalau kita biarkan atau kita tidak mengisi nilai “nmax” untuk URL diatas, maka hasilnya adalah sebagai berikut :



Disamping nilai default (null), kita juga dapat memberikan nilai sesuai dengan kebutuhan. Misalkan kita beri contoh default untuk variabel `$jenis = "Sport"`, maka route nya seperti ini

`routes/web.php`

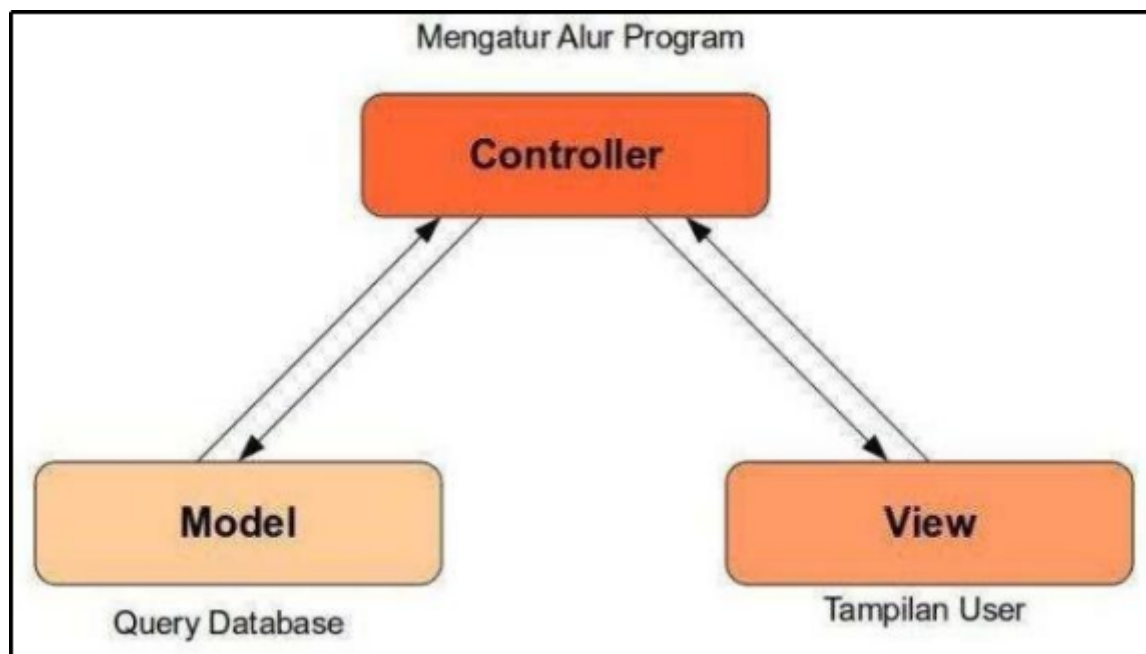
```
Route::get('/motor/{jenis?}', function($jenis = "Sport"){  
    return 'motor dengan jenis : ' . $jenis;  
});
```

Route diatas apabila kita eksekusi dengan tidak memberikan nilai di URL nya maka akan mengembalikan response dengan nilai "**Motor dengan jenis Sport**". Tapi kalau kita memberikan nilai di URLnya misalkan "**Bebek**", maka hasil responnya adalah sebagai berikut "**Motor dengan jenis Bebek**".

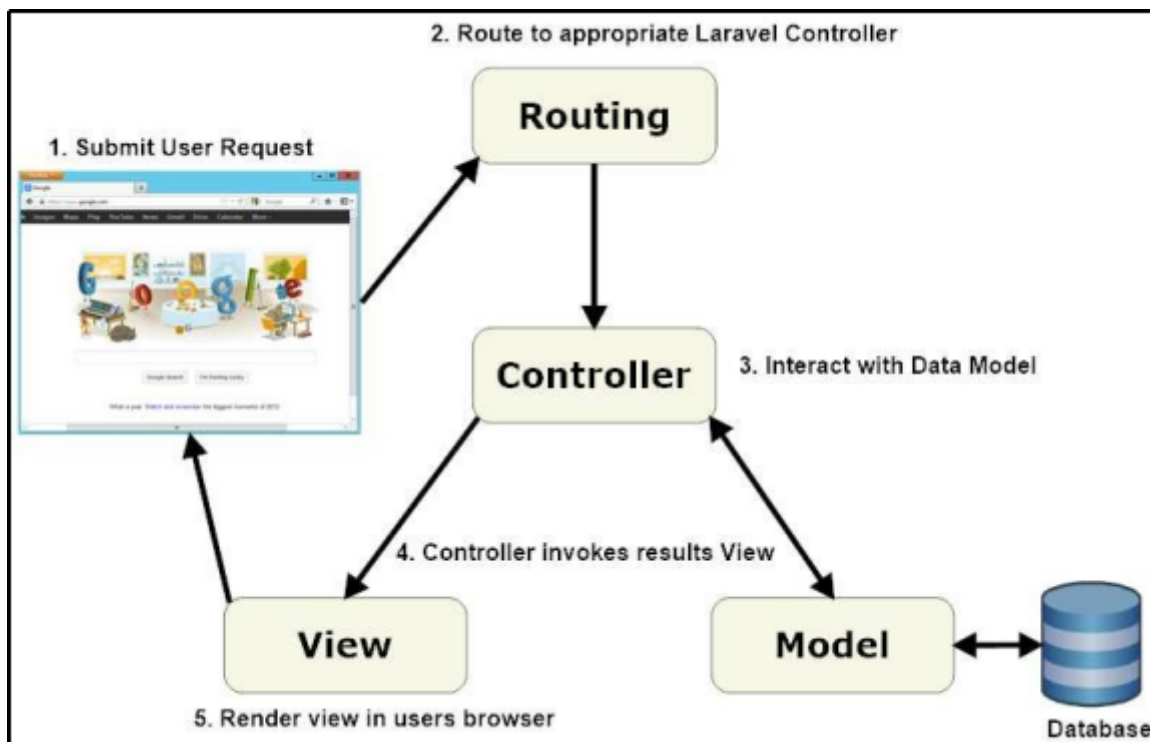
Mengenal MVC (Model-View-Controller)

MVC atau kepanjangan dari Model-View-Controller adalah sebuah metode yang digunakan dalam pengembangan suatu aplikasi yang memisahkan data (model) dari tampilan / frontend (View) dan logic dari aplikasi itu sendiri (Controller). MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, antarmuka pengguna dan kontrol dalam sebuah aplikasi.

Model digunakan untuk proses query atau manipulasi data ke/dari database. Sedangkan View kaitannya erat dengan antarmuka / frontend tampilan sebuah web seperti HTML, CSS dan JS dan data yang bersifat client. Controller adalah logika dari sebuah web. Menjembatani komunikasi antara Model dan View. Kalau digambarkan alur proses MVC adalah sebagai berikut :



Adapun gambar simulasi proses MVC pada Laravel diperlihatkan pada gambar dibawah ini.



Untuk lebih jelasnya kita langsung praktikan proses MVC pada laravel. Pertama kita membuat sebuah controller dengan nama BookController. Disini saya menggunakan composer untuk membuat kontroller. Dengan sintak sebagai berikut :

```
php artisan make:controller BookController
```

```
PS D:\CODE\ict\coba-laravel> php artisan make:controller BookController
```

Setelah itu **BookController** isi sebagai berikut :

App/Http/Controllers/BookController.php

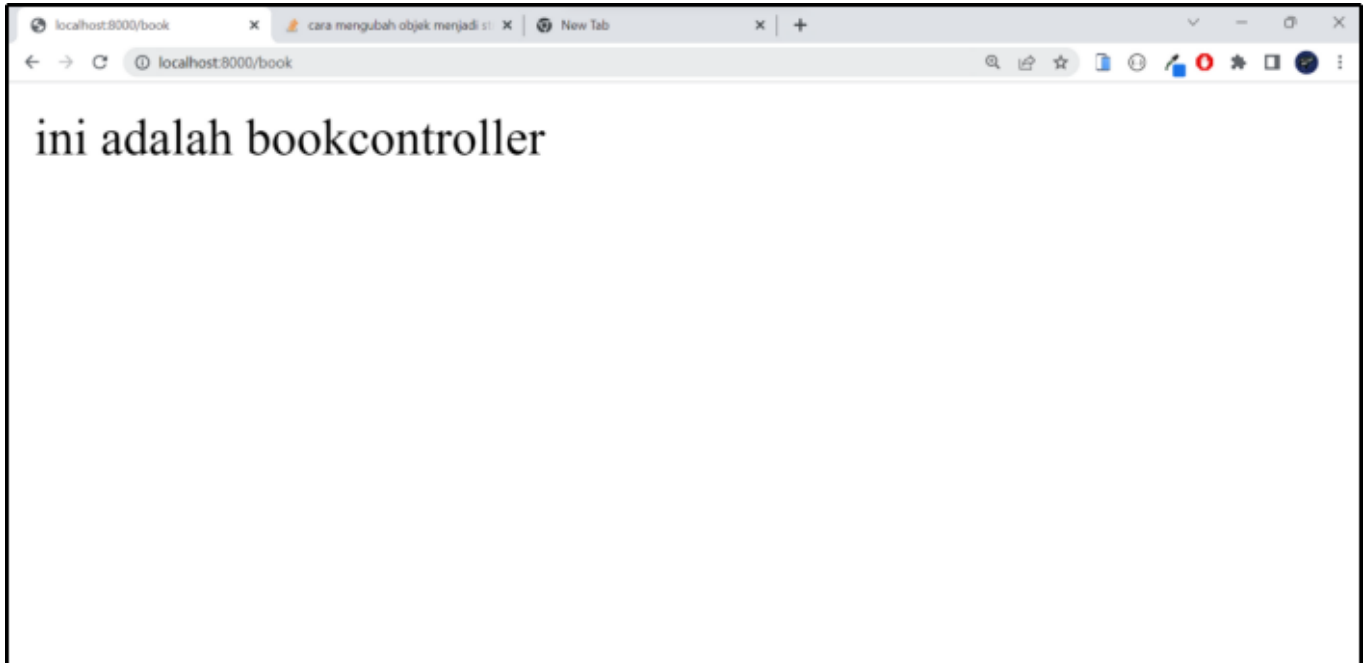
```
<?php
class BookController extends Controller {
    public function index(){
        return 'ini adalah bookcontroller';
    }
}
```

Setelah membuat controller, kemudian kita buka file web.php. Buat sebuah route yang merujuk pada kontroller yang telah dibuat diatas.

routes/web.php

```
Route::get('/book', [BookController::class, 'index']);
```

Penjelasan route diatas adalah Route menggunakan method **get** untuk menampilkan *response* dari method **index** dari controller **BookController**. Untuk melihat output dari proses diatas ketikkan URL sebagai berikut localhost:8000/book . Adapun hasilnya seperti gambar dibawah ini.



Kita akan coba bagaimana mengirimkan sebuah parameter / nilai dari route ke controller. Ganti route user kemudian tambahkan parameter “judul”.

routes/web.php

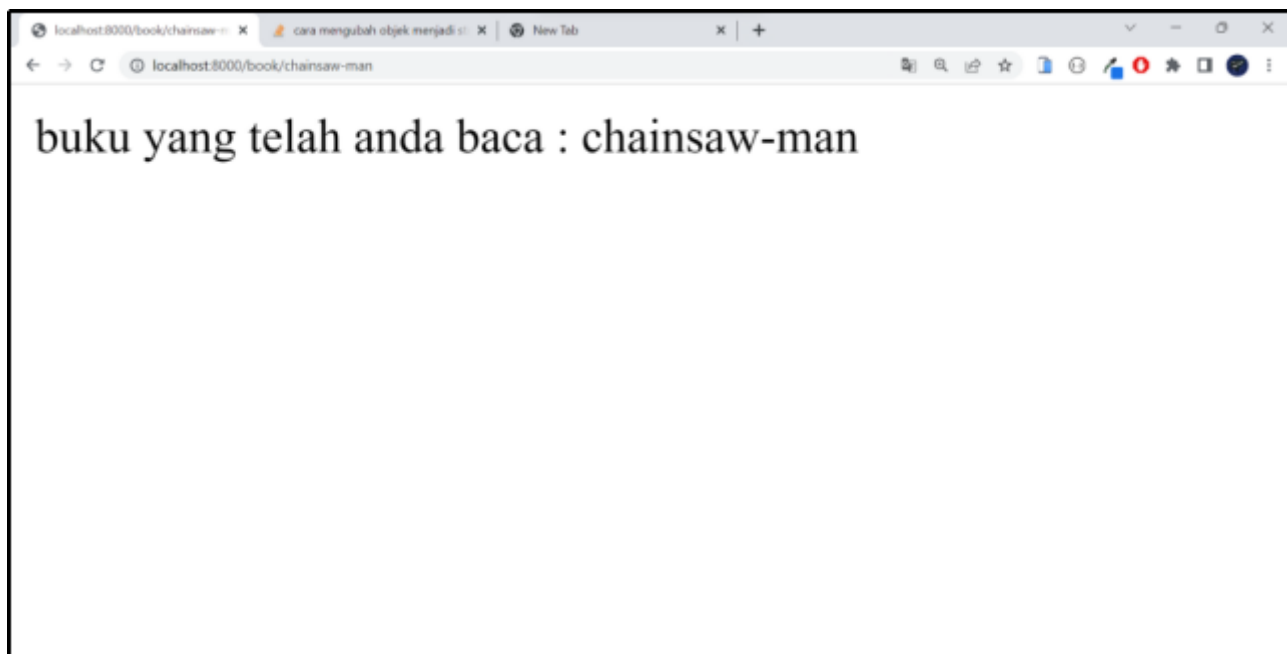
```
Route::get('/book/{judul}', [BookController::class, 'viewJudul']);
```

Lalu kita buat satu buah method viewJudul di BookController

App/Http/Controllers/BookController.php

```
<?php
class BookController extends Controller {
    public function viewJudul($judul){
        return 'buku yang telah anda baca : ' . $judul;
    }
}
```

Kemudian cek kode diatas dengan mengetikkan URL sebagai berikut pada web browser localhost:8000/book/chainsaw-man



Setelah itu, kita akan coba mengintegrasikan Controller dengan View. Pada folder **resources/views/** kemudian buat sebuah file PHP dengan nama **v_book.blade.php** dan isikan kode sebagai berikut.

`resources/views/v_book.blade.php`

```
<h1>Page view</h1>
<p>Buku yang telah anda baca berjudul <h1>{{ $judul }}</h1></p>
```

Buka controller **BookController** kemudian edit method **viewJudul** dan isikan kode sebagai berikut.

`app/Http/Controllers/BookController.php`

```
<?php
class BookController extends Controller {
    public function viewJudul($judul){
        return view('v_book', compact('judul'));
    }
}
```

Method `viewJudul` diatas akan mengeksekusi view `v_book.blade.php` pada folder `resources/view`. Dan kalau kita jalankan di web browser dengan mengetikan URL sebagai berikut `localhost:8000/book/chainsaw-man` hasilnya adalah sebagai berikut.



Mengenal Blade Template Engine Laravel

Template engine adalah sebuah method untuk mempersingkat penulisan kode yang lebih panjang contoh nya yaitu smartt, twigg dan doo. Sedangkan blade itu sendiri adalah template engine bawaan laravel. Blade menawarkan penulisan kode/sintax yang mudah dan singkat untuk dipakai dalam menghasilkan kode HTML.

Pada bagian View inilah fungsi Blade sangat dibutuhkan. View seperti yang sudah kita tahu berfungsi menampilkan sebuah halaman web, namun bukan berarti dalam view tersebut tidak bisa melakukan proses logic. Disinilah peran blade yang dibutuhkan untuk membantu menuliskan logic agar menjadi lebih simple. Disamping itu, blade juga berfungsi untuk memisahkan layout suatu web dengan layout tertentu dan blade sendiri mendukung inheritance (OOP). Semua file blade harus menggunakan ekstensi **.blade**. Contoh jika kita membuat sebuah file **Book.php** maka untuk bisa menggunakan fitur blade, maka harus diberi nama menjadi **Book.blade.php**. Berikut adalah perbedaan mendasar antara sintak PHP dan blade.

PHP Syntax	Blade Syntax
<code><?php echo \$var; ?></code>	<code>{{ \$var }}</code>
<code><?php echo htmlentities(\$var) ; ?></code>	<code>{{{ \$var }}}}</code>
<code><?php if(\$cond) : ?> .. <?php endif; ?></code>	<code>@if(\$cond) ... @endif</code>

Blade juga mendukung penuh proses looping dan kondisi-logika PHP seperti **@for**,

@foreach, **@while** , **@if** dan **@elseif**. Supaya tidak bingung dalam menuliskan sintak berikut akan saya contohkan menulis sintak php biasa dengan sintak blade.

echo variabel

```
/* script php */
<?php echo "Halo ini cara lama"; ?>

/* script blade template */
{{ "Halo ini cara lama" }}
```

echo variabel dengan nilai default

```
/* script php */
<?php echo isset($name) ? name : 'guest'; ?>

/* script blade template */
{{ $name or 'guest' }}
```

kondisi percabangan

```
/* script php */
<?php
if($status == 0){
    echo "Proses Gagal";
}elseif($status == 1){
    echo "Proses Berhasil";
}else{
    echo "Tidak diketahui";
}

/* script blade template */
@if($status == 0)
    {{ "Proses Gagal" }}
}elseif($status == 1)
    {{ "Proses Berhasil" }}
@else
    {{ "Tidak diketahui" }}
@endif

/* kebalikan dari if, kondisi tidak memenuhi syarat yang ada */
@unless
```

```
    {{ "Anda tidak berhak mengakses halaman ini" }}  
@endunless
```

Looping atau Iterasi

```
/* script php */  
  
// for statement  
<?php  
    for($i = 1; $i < 10; $i++){  
        echo $i;  
    }  
?>  
  
// while statement  
<?php  
    while($i < 10){  
        echo $i++;  
    }  
?>  
  
// foreach statement  
<?php  
    foreach($arr as $value){  
        $value = $value + 1;  
        echo $value;  
    }  
?>  
  
/* script blade template */  
  
// for statement  
@for($i = 1; $i < 10; $i++)  
    {{ $i }}  
@endfor  
  
// while statement  
@while($i < 10)  
    {{ $i++ }}  
@endwhile  
  
// foreach statement  
@foreach($arr as $value)  
    {{ $value = $value + 1 }}
```



```
    {{ $value }}  
@endforeach
```

include sub-view

```
/* script php */  
include 'folder/subview';  
  
/* script blade template */  
@include('folder.subview')
```

Penggunaan yield

Penggunaan "yield" dalam Blade Template Engine Laravel merupakan cara yang efektif untuk mengoptimalkan struktur layout dalam pengembangan web dengan kerangka kerja Laravel. "Yield" digunakan untuk mengelola konten yang dinamis dalam berbagai bagian template tanpa harus mengulangi kode yang sama di setiap halaman. Mari kita lihat mengapa penggunaan "yield" ini sangat berguna.

Contoh penggunaan yield

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title>@yield('title')</title>  
  </head>  
  <body>  
    <main>@yield('content')</main>  
  </body>  
</html>
```

buat halaman spesifik yaitu home dan about pada view resources laravel

resources/views/home.blade.php

```
@extends('layout')  
@section('title', 'Halaman Utama')  
@section('content')  
  <h1>Selamat datang di halaman utama!</h1>  
@endsection
```

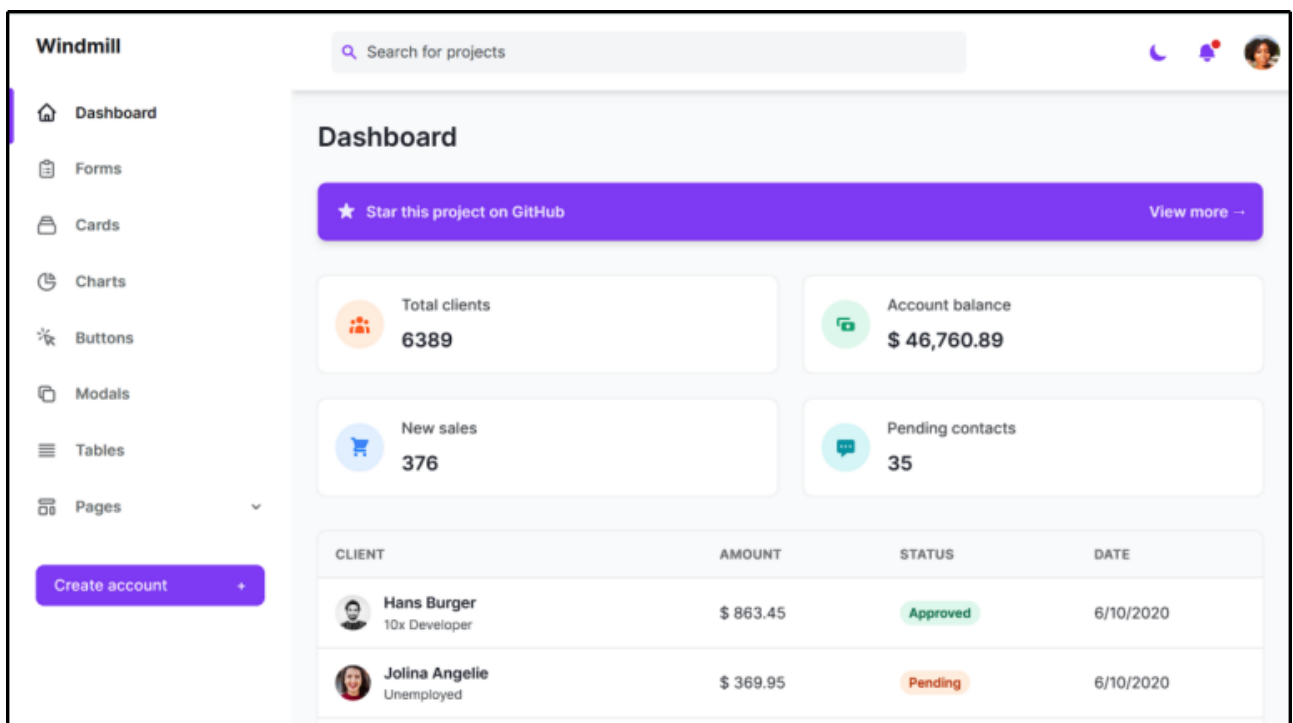
resources/views/about.blade.php

```
@extends('layout')
@section('title', 'Halaman Tentang Kami')
@section('content')
    <h1>Tentang Kami</h1>
@endsection
```

Latihan Memakai Template Admin Dashboard

1. Pilih Template Admin Dashboard

Cari dan pilih template admin dashboard yang sesuai dengan kebutuhan Anda. Ada banyak template yang tersedia secara gratis atau berbayar di internet, seperti AdminLTE, Bootstrap Admin Themes, dan lainnya. disini saya memakai template admin windmill



2. Integrasi Template ke Laravel

3. Membuat Bagian Blade untuk Navbar, Sidebar, Header, Body, dan Footer

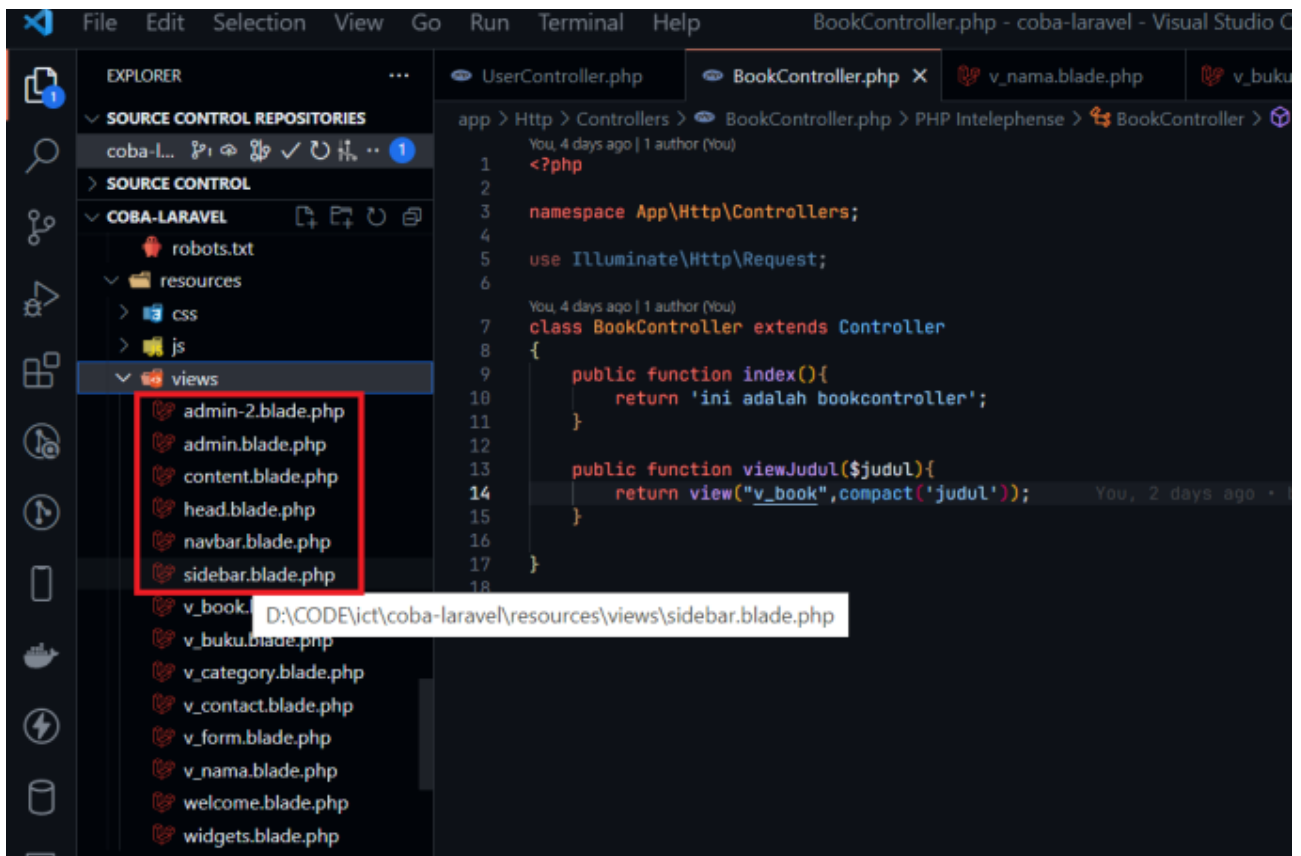
Masuk ke folder **resources/views** kemudian buat terlebih dahulu file blade untuk adminnya dan isi file adminnya dari template sudah kita pilih

Contoh : **admin.blade.php**

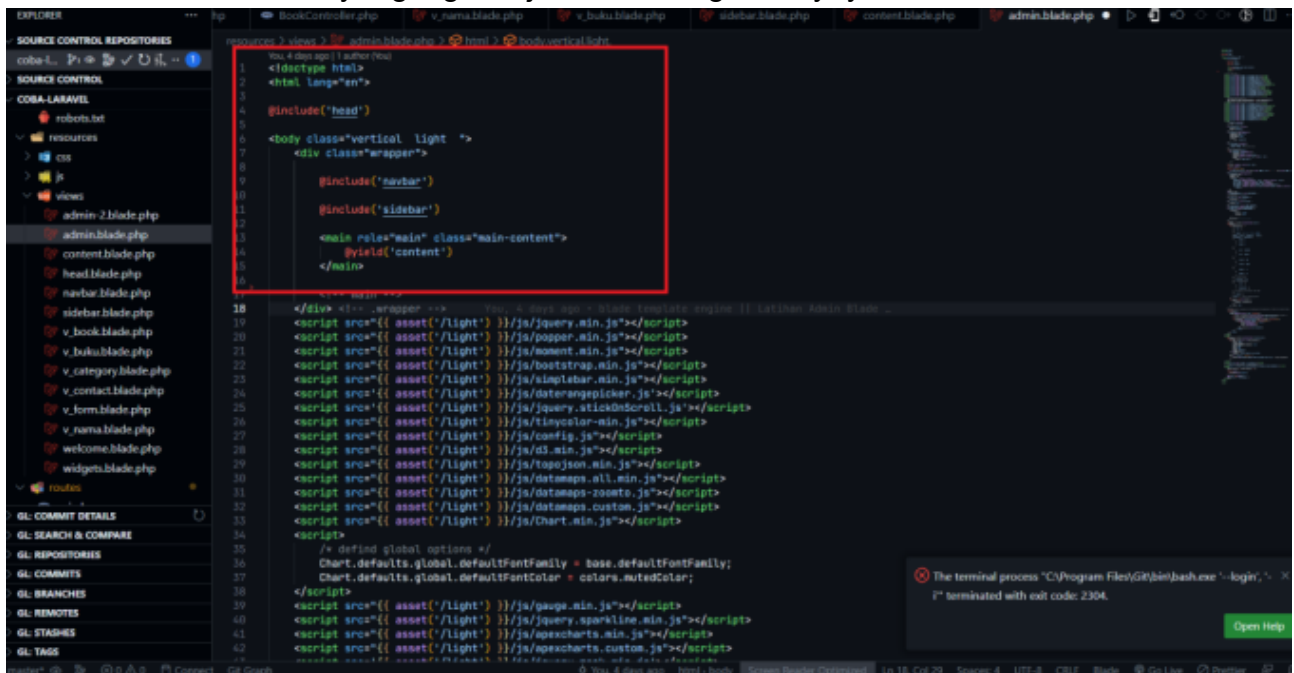
Lalu Sisipkan file html yang sudah kita pilih ke **admin.blade.php**

Kemudian cari komponen-komponen yang ingin dipisah

Dari Navbar, Sidebar, Header Dan Footer Buat Masing Masing File Untuk Komponen-Komponen Tertentu



Setelah itu, buatlah file yang ingin di jadikan sebagai bodynya.



Kemudian Jika Template Admin Anda Masih Belum Sepenuhnya Berfungsi Mungkin Itu Dikarenakan File File Yang Dibutuhkan Belum Terpanggil Seperti Javascript,Css,Gambar Dll.

Disini Kita Akan Menggunakan Fungsi **Asset**.Penggunaan umum dari asset adalah untuk membangun URL yang benar ke aset yang berada di dalam direktori "public" di dalam

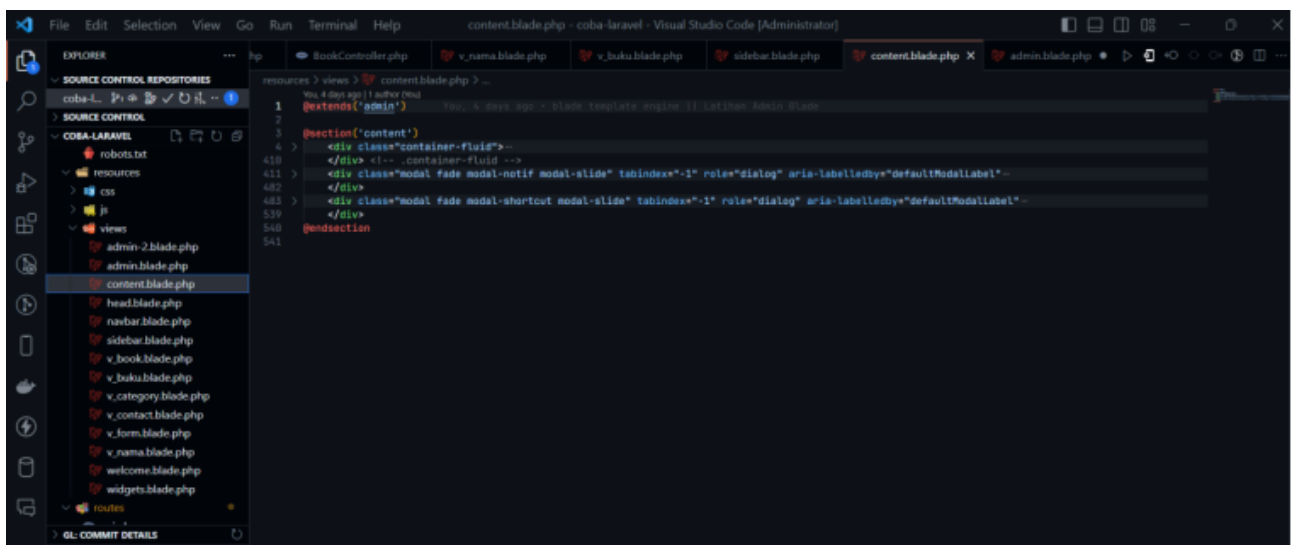
proyek Laravel Anda. Alasan penggunaan direktori "public" adalah karena konten di dalam direktori ini dapat diakses langsung melalui URL, menjadikannya tempat yang tepat untuk menyimpan aset-aset publik.

```
19 <script src="{{ asset('/light') }}" />js/jquery.min.js"></script>
20 <script src="{{ asset('/light') }}" />js/popper.min.js"></script>
21 <script src="{{ asset('/light') }}" />js/moment.min.js"></script>
22 <script src="{{ asset('/light') }}" />js/bootstrap.min.js"></script>
23 <script src="{{ asset('/light') }}" />js/simplebar.min.js"></script>
24 <script src="{{ asset('/light') }}" />js/daterangepicker.js"></script>
25 <script src="{{ asset('/light') }}" />js/jquery.stickOnScroll.js"></script>
26 <script src="{{ asset('/light') }}" />js/tinycolor-min.js"></script>
27 <script src="{{ asset('/light') }}" />js/config.js"></script>
28 <script src="{{ asset('/light') }}" />js/d3.min.js"></script>
29 <script src="{{ asset('/light') }}" />js/topojson.min.js"></script>
30 <script src="{{ asset('/light') }}" />js/datamaps.all.min.js"></script>
31 <script src="{{ asset('/light') }}" />js/datamaps-zoomto.js"></script>
32 <script src="{{ asset('/light') }}" />js/datamaps.custom.js"></script>
33 <script src="{{ asset('/light') }}" />js/Chart.min.js"></script>
34 <script>
35     /* definid global options */
36     Chart.defaults.global.defaultFontFamily = base.defaultFontFamily;
37     Chart.defaults.global.defaultFontColor = colors.mutedColor;
38 </script>
39 <script src="{{ asset('/light') }}" />js/gauge.min.js"></script>
40 <script src="{{ asset('/light') }}" />js/jquery.sparkline.min.js"></script>
41 <script src="{{ asset('/light') }}" />js/apexcharts.min.js"></script>
42 <script src="{{ asset('/light') }}" />js/apexcharts.custom.js"></script>
43 <script src="{{ asset('/light') }}" />js/jquery.mask.min.js"></script>
44 <script src="{{ asset('/light') }}" />js/select2.min.js"></script>
45 <script src="{{ asset('/light') }}" />js/jquery.steps.min.js"></script>
46 <script src="{{ asset('/light') }}" />js/jquery.validate.min.js"></script>
47 <script src="{{ asset('/light') }}" />js/jquery.timepicker.js"></script>
48 <script src="{{ asset('/light') }}" />js/dropzone.min.js"></script>
49 <script src="{{ asset('/light') }}" />js/uppy.min.js"></script>
50 <script src="{{ asset('/light') }}" />js/quill.min.js"></script>
51 <script>
```

4. Membuat Halaman Konten Spesifik

Dalam folder "resources/view", buat halaman-halaman blade yang akan mengisi bagian konten dari template.

Misalnya dalam "dashboard.blade.php".



```
resources > views > content.blade.php > ...
1 @extends('admin')
2
3 @section('content')
4 <div class="container-fluid">
5 <div class="modal fade modal-notif modal-slide" tabindex="-1" role="dialog" aria-labelledby="defaultModalLabel"
6 </div>
7 <div class="modal fade modal-shortcut modal-slide" tabindex="-1" role="dialog" aria-labelledby="defaultModalLabel"
8 </div>
9 @endsection
```

5. Route dan Tampilan Konten

Tentukan rute di dalam file "web.php" yang akan menampilkan halaman konten.

routes/web.php

```
Route::get('/dashboard', function(){  
    return view('dashboard');  
});
```

6. Uji Tampilan

Jalankan server Laravel Anda dan buka URL yang sesuai dengan rute yang Anda tetapkan (misalnya, '/dashboard').

Schema Builder Dasar

Konfigurasi Koneksi Database

Salah satu bagian penting dari konfigurasi aplikasi Laravel adalah pengaturan koneksi ke database. Laravel menyediakan dukungan untuk berbagai database seperti MySQL, PostgreSQL, SQLite, dan SQL Server. Proses konfigurasi ini melibatkan pengaturan informasi koneksi database di file environment (`.env`) dan file konfigurasi (`config/database.php`).

File `.env` digunakan untuk menyimpan konfigurasi lingkungan aplikasi, termasuk detail koneksi database.

1. Buka file `.env` di root direktori proyek Laravel Anda.
2. Cari bagian konfigurasi database, yang biasanya terlihat seperti ini:

```
DB_CONNECTION=mysql  
DB_HOST=127.0.0.1  
DB_PORT=3306  
DB_DATABASE=nama_database  
DB_USERNAME=username_database  
DB_PASSWORD=password_database
```

3. Ganti `nama_database` , `username_database` , dan `password_database` dengan informasi database Anda.

Setelah konfigurasi dilakukan, Anda dapat menguji koneksi ke database dengan menjalankan migrasi. dengan menjalankan perintah berikut.

```
php artisan migrate
```

Membuat Tabel

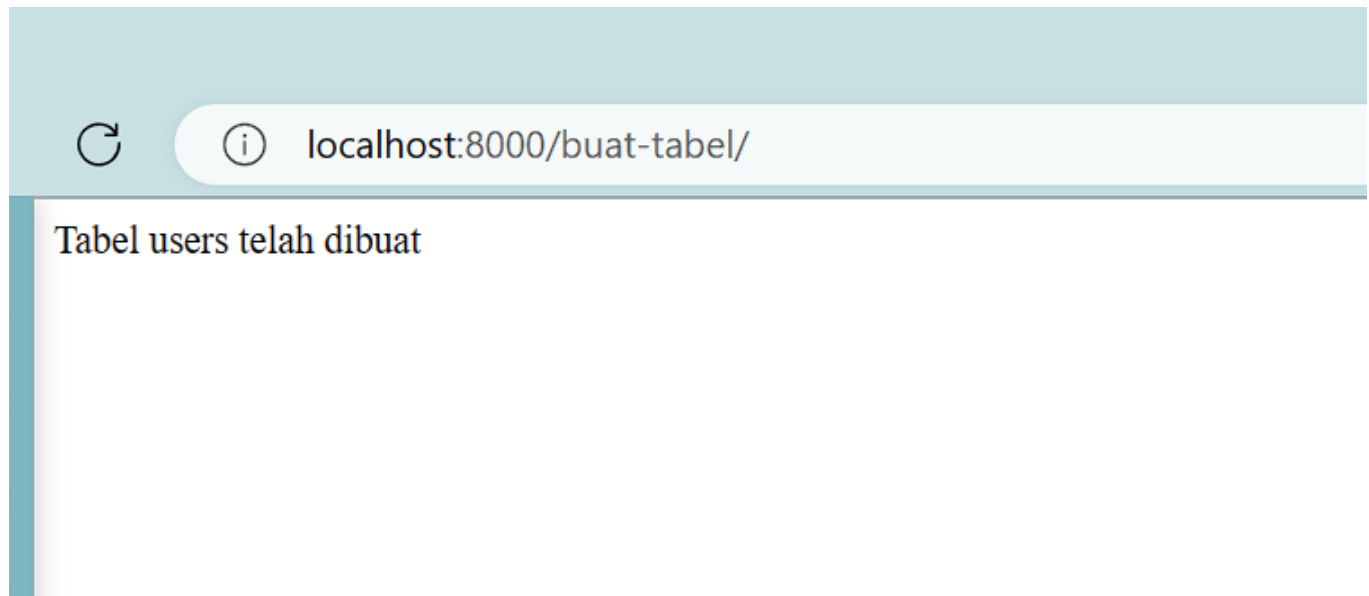
Buka file `routes.php` dan buat route baru untuk membuat sebuah tabel.

routes/web.php

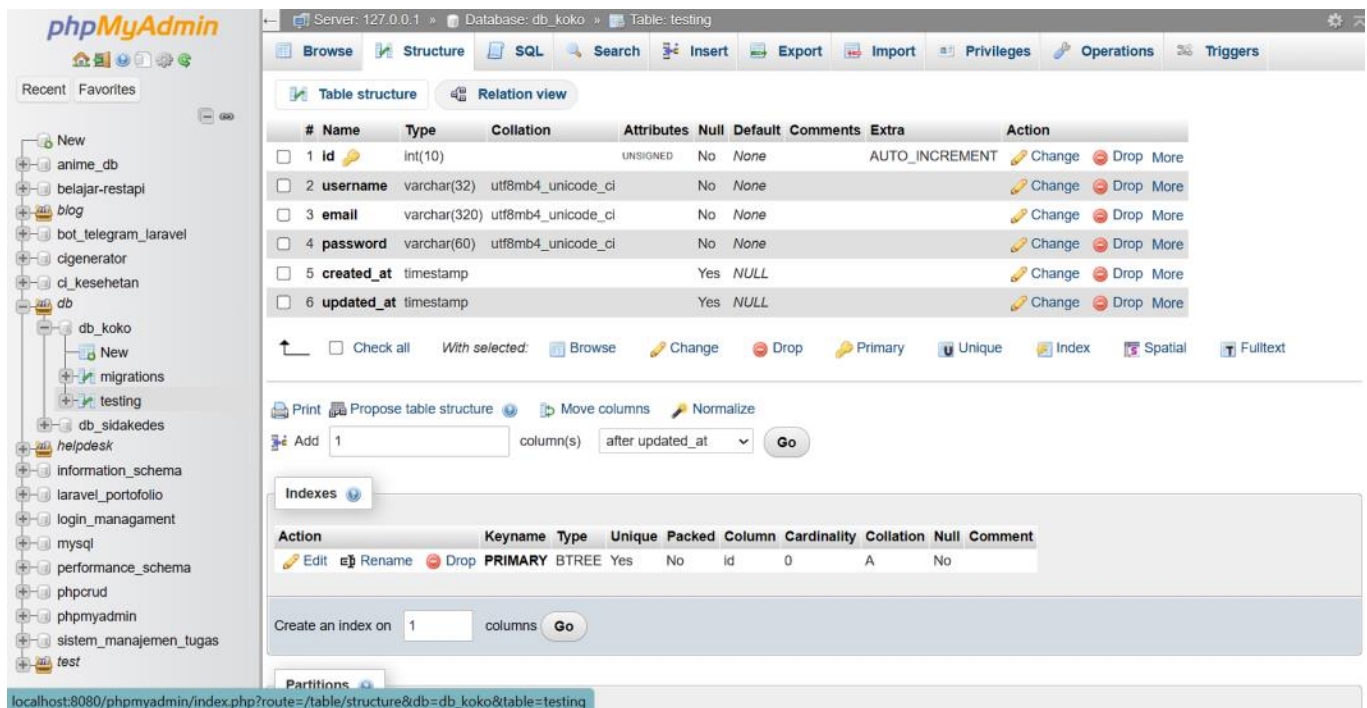
```
Route::get('users', function(){
    Schema::create('users', function($table){
        $table->increments('id');
        $table->string('username', 100);
        $table->string('email', 100);
        $table->string('password', 60);
        $table->timestamps;
    });
    return "table users telah dibuat";
});
```

Schema::create mempunyai dua parameter. Parameter pertama, yaitu untuk mendefinisikan nama tabel yang akan dibuat sedangkan parameter kedua yaitu untuk mendefinisikan struktur tabel dari tabel user.

Untuk mengeksekusi kode diatas kemudian buka browser dan ketikan URL dibawah ini
<http://localhost:8000/users>



masuk ke database mysql, pilih database yang telah di buat, untuk mengecek apakah tabel telah dibuat.



Tabel users telah dibuat di mysql

Menambah, Rename dan Menghapus Kolom

Kadang-kadang di tengah proses development suatu aplikasi, adakalanya kita ingin menambahkan, menghapus atau bahkan mengganti nama dari suatu kolom. Berikut adalah sintak untuk manipulasi kolom.

Tambah Kolom

```
Schema::table('users', function($table){
    $table->text('alamat');
});
```

Rename Kolom

```
Schema::table('users', function($table){
    $table->renameColumn('alamat', 'alamatKirim');
});
```

Menghapus Kolom

```
Schema::table('users', function($table){
    $table->dropColumn(['alamat', 'no_telp']);
});
```

Menambahkan Index dan Foreign Key

Hal ini sangat berguna jika ada tabel yang berelasi dengan tabel lain dengan cara menambahkan foreign key atau menambahkan hal yang unik untuk kolom dari suatu tabel.

Menambahkan index pada kolom

Menambahkan Foreign Key

Tipe Kolom

Dari tabel user yang telah kita buat diatas, ada tipe data increments , string dan timestamps merupakan tipe data yang nantinya akan dikonversikan ke tipe data yang ada dalam database mysql.

- **Increments**

tipe ini akan memberikan nilai integer dengan nilai yang bertambah secara otomatis (increments)

```
$table->increments('id');
```

- **bigIncrements**

jika dirasa tipe increments tidak cukup untuk kamu. Bisa menggunakan method bigIncrements() yang akan membuat tipe data big integer.

```
$table->bigIncrements('id');
```

- **string**

method ini digunakan untuk menghasilkan tipe data varchar di mysql. Parameter pertama untuk memberi nama kolom / field sedangkan kolom kedua untuk memberi rentang nilai yang diberikan untuk kolom tersebut.

```
$table->string('username', 32);
```

- **text**

method text ini digunakan untuk menyimpan data teks yang berukuran besar seperti menyimpan artikel, berita ataupun postingan dari blog.

```
$table->text('post');
```


- **integer**

biasanya digunakan untuk tipe data yang berupa bilangan.

```
$table->integer('age', 3);
```

- **float**

float digunakan untuk menyimpan bilangan pecahan atau desimal.

```
$table->float('size');
```

- **boolean**

tipe ini hanya mempunyai nilai true atau false.

```
$table->boolean('isSmart');
```

- **date**

sesuai dengan namanya, tipe ini digunakan untuk menyimpan tanggal / date

```
$table->date('departure');
```

- **timestamps**

method ini digunakan untuk menyimpan data tanggal dan waktu dalam format TIMESTAMP.

Pada tabel user diatas, kolom timestamps akan menghasilkan dua kolom di tabel users mysql yaitu created_at dan updated_at.

```
$table->timestamps();
```

Migration

Kita membuat struktur tabel, relasi dan lain-lainnya biasanya langsung dari mysql langsung atau menulis sintak SQL dan mendeskripsikan tabel beserta kolom apa saja yang dibutuhkan, tapi apa yang akan terjadi jika kita secara tidak sengaja menghapus database tersebut ? apa yang akan terjadi jika kamu belajar sebagai team ? mungkin kamu akan memberikan SQL dump ke masing-masing anggota untuk menjaga database agar selaras. Dari permasalahan dasar seperti itulah fungsi migrations sangat dibutuhkan. Dengan menggunakan fitur migrations ini, kita dapat membuat, memodifikasi dan menghapus suatu tabel atau relasi antar tabel dengan menggunakan kode program dari laravel itu sendiri yaitu migrations. Dengan menggunakan migrations, kamu dan tim dan menjaga konsistensi struktur database, tabel-

tabel beserta kolomnya. Atau jika masih bingung dari penjelasan diatas, kita akan langsung coba mempraktekannya.

Membuat Migrations

Pertama, kita akan membuat tabel post dengan menggunakan migrations. Masuk ke Command Prompt windows, kemudian arahkan ke direktori laravel proyek. Untuk membuat file migrations gunakan perintah berikut :

```
php artisan make:migration <nama_file_migrations>
```

```
$ php artisan make:migration users
INFO Migration [D:\CODE\ict\course-laravel-irfan\database\migrations\2023_10_02_070205_users.php] created successfully.
```

Jika hasilnya sama dengan gambar diatas, kemudian kita cek pada direktori app/database/migrations disini ada file migrations php dengan nama 2023_10_02_070205_create_table_users.php. Nama awalnya bisa berbeda-beda tapi nama akhirnya pasti sama. Dan apabila kita buka, akan menghasilkan kode php sebagai berikut :

app/database/migrations/2023_10_02_070205_users.php

```
<?php
```

```
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration {
    public function up(): void {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
        });
    }

    public function down(): void {
        Schema::dropIfExists('users');
    }
};
```

Disana terdapat method up() dan down(). Dimana method up() digunakan untuk membuat table dan memanipulasi kolom dari tabel sedangkan method down() biasanya digunakan untuk

menghapus tabel atau kolom. Pada method up() dan down()kemudian kita definisikan tabel users beserta kolom nya .

```
app/database/migration/2023_10_02_070205_users.php
```

```
Schema::create('users', function (Blueprint $table) {
    $table->increments('id');
    $table->string('nama', 128);
    $table->string('email');
    $table->string('password', 60);
    $table->timestamps();
});
```

Menjalankan Migrations

Setelah itu masuk ke command prompt yang tadi, kemudian kita akan menjalankan file migration diatas dengan sintak sebagai berikut, jika ada konfirmasi install, ketik (y) kemudian Enter :

```
php artisan migrate
```

```
Linux@kali:~$ php artisan migrate
INFO Running migrations.

2014_10_12_000000_create_users_table ..... 169ms DONE
2014_10_12_100000_create_password_reset_tokens_table ..... 76ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 112ms DONE
2019_12_14_000001_create_personal_access_tokens_table ..... 130ms DONE
2023_08_24_064354_book ..... 80ms DONE
2023_10_02_070205_users ..... 0ms DONE
```

File tersebut sudah berhasil, kemudian masuk ke database mysql untuk mengeceknya

Filters

Containing the word:

	Table	Action	Rows	Type	Collation	Size	Overhead	
<input type="checkbox"/>	books	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-	
<input type="checkbox"/>	failed_jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-	
<input type="checkbox"/>	migrations	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-	
<input type="checkbox"/>	password_reset_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-	
<input type="checkbox"/>	personal_access_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-	
<input type="checkbox"/>	testing	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-	
<input type="checkbox"/>	users	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-	
7 tables			Sum	6	InnoDB	utf8mb4_general_ci	176.0 KiB	0 B

☐ Check all

With selected:

Diatas, pada database `db_laravel-11` terdapat dua tabel yaitu tabel `migrations` dan `users`. Karena kita menggunakan fitur migrations. Maka otomatis laravel akan membuat tabel migrations yang isinya tentang tabel-tabel yang sudah dimigrasi ke MySQL.

Rolling Migrations

Seperti yang telah kita ketahui, migrations digunakan untuk kepentingan struktur tabel atau relasi. Tapi kita asumsikan, kita sedang dalam suatu kondisi yang mengharuskan kita mengatur ulang tabel-tabel yang telah kita buat. Dari situ, kita membutuhkan untuk me-rollback perubahan dari tim yang sudah dibuat. Maka kita dapat menggunakan perintah `rollback`

```
php artisan migrate:rollback
```

```
$ php artisan migrate:rollback
INFO Rolling back migrations.

2023_10_02_070205_users ..... 4ms DONE
2023_08_24_064354_book ..... 33ms DONE
2019_12_14_000001_create_personal_access_tokens_table ..... 25ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 97ms DONE
2014_10_12_100000_create_password_reset_tokens_table ..... 72ms DONE
2014_10_12_000000_create_users_table ..... 15ms DONE
```

Jika kita mengecek didatabase `db_laravel`, maka tabel yang suda dibuat menggunakan migrasi akan dibalikan kembali pada saat kita terakhir kali menggunakan perintah `migrate`. Atau jika kita ingin me-rollback semua tabel migrasi, maka gunakan perintah `reset`.

```
php artisan migrate:reset
```

Filters

Containing the word:

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> migrations	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> testing	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
2 tables	Sum	0	InnoDB	utf8mb4_general_ci	32.0 KiB	0 B

☐ Check all

With selected:

[Print](#)
[Data dictionary](#)

Seeding

Selain fitur migrations yang dapat membuat struktur tabel dan relasi menggunakan kode program, kita juga dapat menggunakan fitur seeding untuk memasukan data ke tabel tersebut. Seed adalah segala sesuatu yang harus dimuat dalam sebuah aplikasi untuk memastikan aplikasi dapat berjalan dengan baik. Hal ini biasanya dianggap sebagai tahap untuk pengujian sistem dan demo. Sebagian besar aplikasi membutuhkan data referensi untuk dimuat guna untuk memastikan kesuksesan proses pengembangan, testing maupun produksi. Untuk mempermudah pemahaman tentang seeding maka akan kita praktikan langsung. Sebelumnya file migrasi yang telah dibuat di bab sebelumnya, kita migrasikan kembali ke database untuk menghasilkan tabel users.

Membuat Seeding

Untuk membuat seeder di laravel, laravel mempunyai command artisan untuk membuat seeder yaitu:

```
php artisan make:seeder UserSeeder
```

Kemudian buatlah seedingnya pada direktori seeders

Database/seeders/UserSeeder.php

```
<?php
class UserSeeder extends Seeder {
    public function run(): void
    {
        User::create([
            'nama' => 'Admin',
            'email' => 'admin@gmail.com',
            'password' => Hash::make('rahasia123')
        ]);
    }
}
```

```
}  
}
```

Pada kode diatas ada kode `Hash::make('admin123')`, kode tersebut untuk mengenkripsi password (rahasia123). Hasil enkripsi nya bisa kita lihat didepan pada saat file seeder sudah dieksekusi. Sesudah kita membuat kelas `UsersSeeder.php`, supaya laravel mengenali file seeder yang sudah dibuat kemudian kita daftarkan ke kelas `DatabaseSeeder.php` yang berada dalam satu direktori dengan kelas tersebut yaitu `app/database/seeder`s.

`database/seeder/DatabaseSeeder.php`

```
<?php  
  
class DatabaseSeeder extends Seeder {  
    public function run(): void  
    {  
        $this->call([  
            UserSeeder::class  
        ]);  
    }  
}
```

Menjalankan Seeding

Untuk menjalankan file seeder, masih di command prompt gunakan perintah berikut, jika ada konfirmasi untuk melakukan seeding ketik (y) lalu tekan Enter.

```
php artisan db:seed
```

```
$ php artisan db:seed  
INFO Seeding database.
```