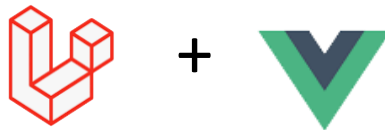


LARAVEL API

&

VUE JS

MANUAL BOOK



By

Tin Mai Zaw

Northern City

Published in 2024

COURSE OUTLINE

► Week-1

- Installations and Setup
- Routing
- CRUD Part-1
- CRUD Part-2
- Assignment-1

Week-2

- Upload File
- Validations
- Pagination & Search
- Assignment-2

Week-3

- Middleware
- Authentication
- Laravel API and Postman
- Assignment-3

Week-4

- Vue JS Installation & setup
- Template Syntax
- Components
- Assignment-3

Week-5

- Vue Router
- State Management with Vuex
- Assignment-5

Week-6

- Composition API
- Axios
- Assignment-6

Week-7

- Restful API CRUD requests
- Forms
- Assignment-7

Week-8

- Unit Testing
- Git & Github
- Docker
- Project Deployment

Week-9

- Coffeeshop project part-1
- Coffeeshop project part-2
- Coffeeshop project part-3
- Coffeeshop project part-4

Week-10

- Recipes project part-1
- Recipes project part-2
- Recipes project part-3
- Recipes project part-4

Week-11

- E-commerce project part-1
- E-commerce project part-2
- E-commerce project part-3
- E-commerce project part-4

Week-12

- Course Reviews
- Questions & Answers

WEEK-1

- Installations and Setup

- 🎥 Video Lesson-1

- Routing

- 🎥 Video Lesson-2

- CRUD Part-1

- 🎥 Video Lesson-3

- CRUD Part-2

- 🎥 Video Lesson-4

- Assignment-1

Installations and Setup

🎬 Laravel Video Lesson 1



Installations and Setup

- Xamp Server installation
- Composer installation
- VSCode Installation
- Running the first app

XAMPP Server Installation

////////////////////////////////////

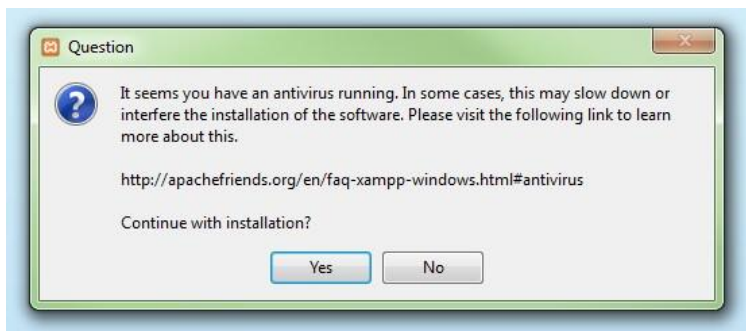
Download

////////////////////////////////////

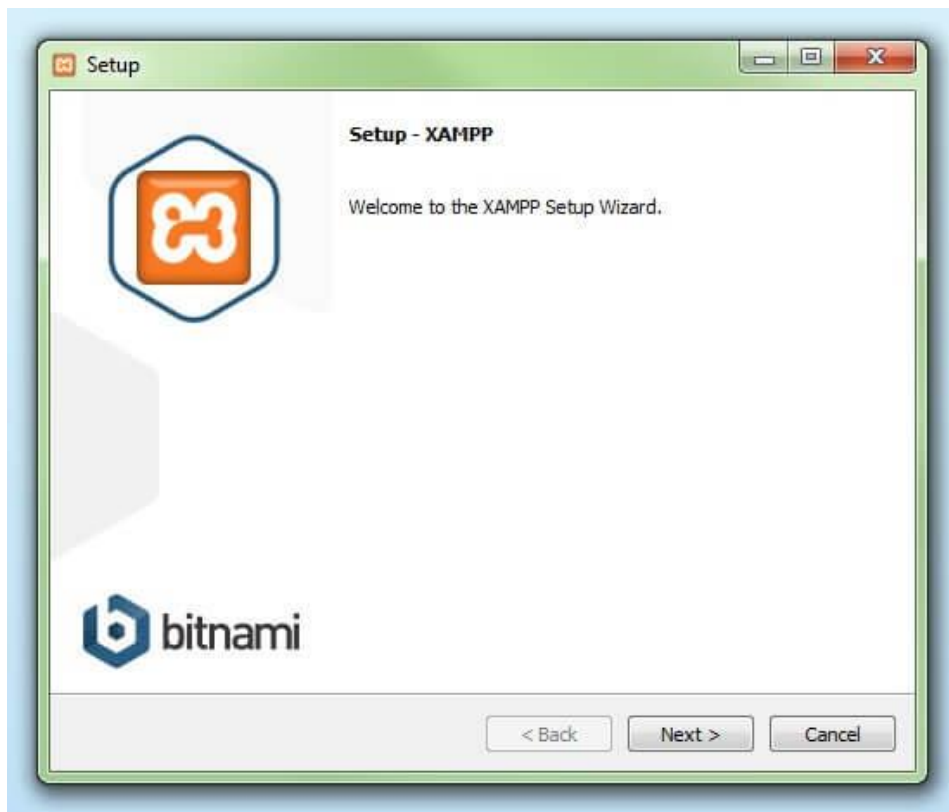
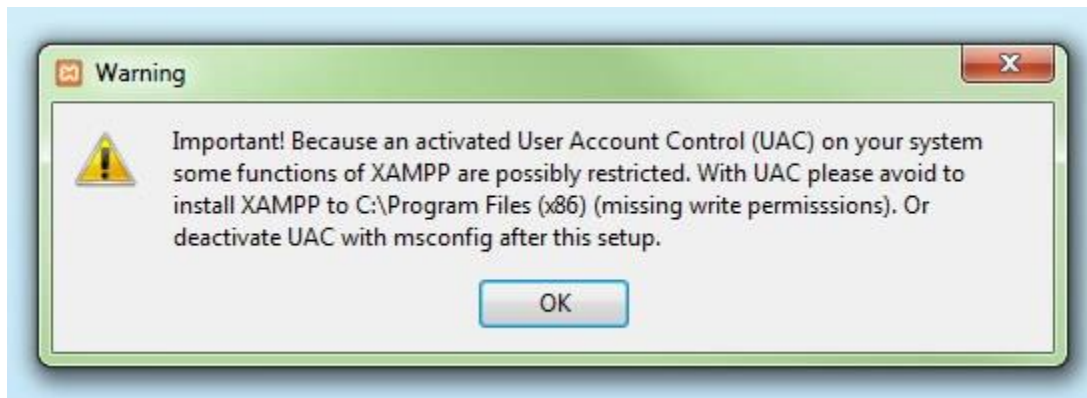
အောက်က download link ကိုခေါက်ပြီး xampp server ကို ဒေါင်းလိုက်ပါ။

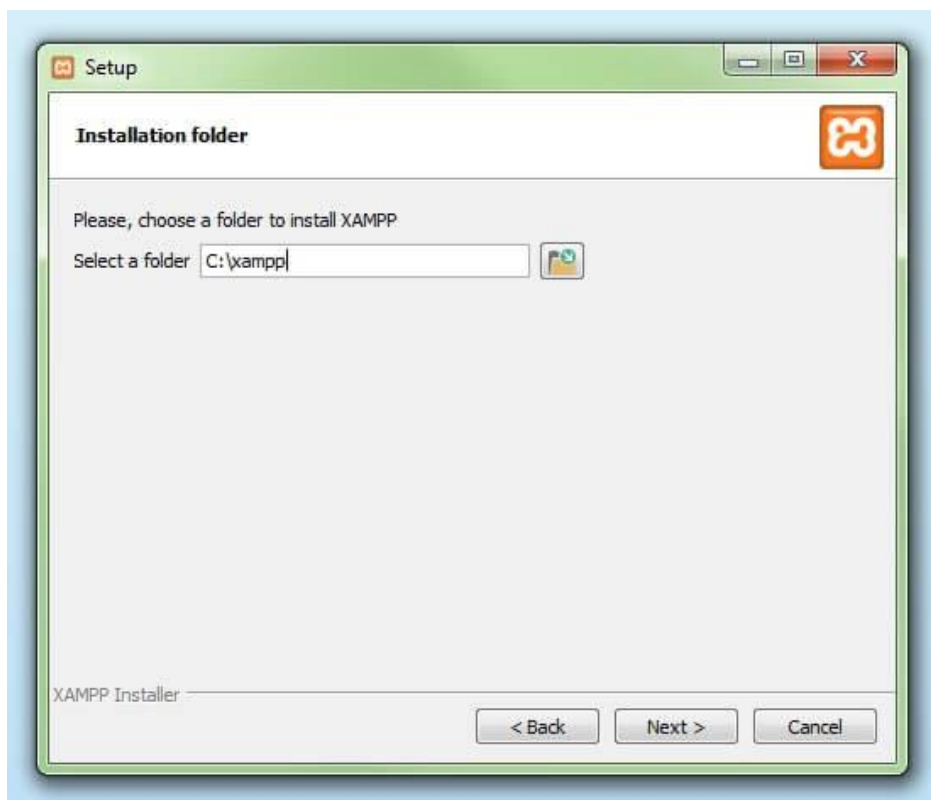
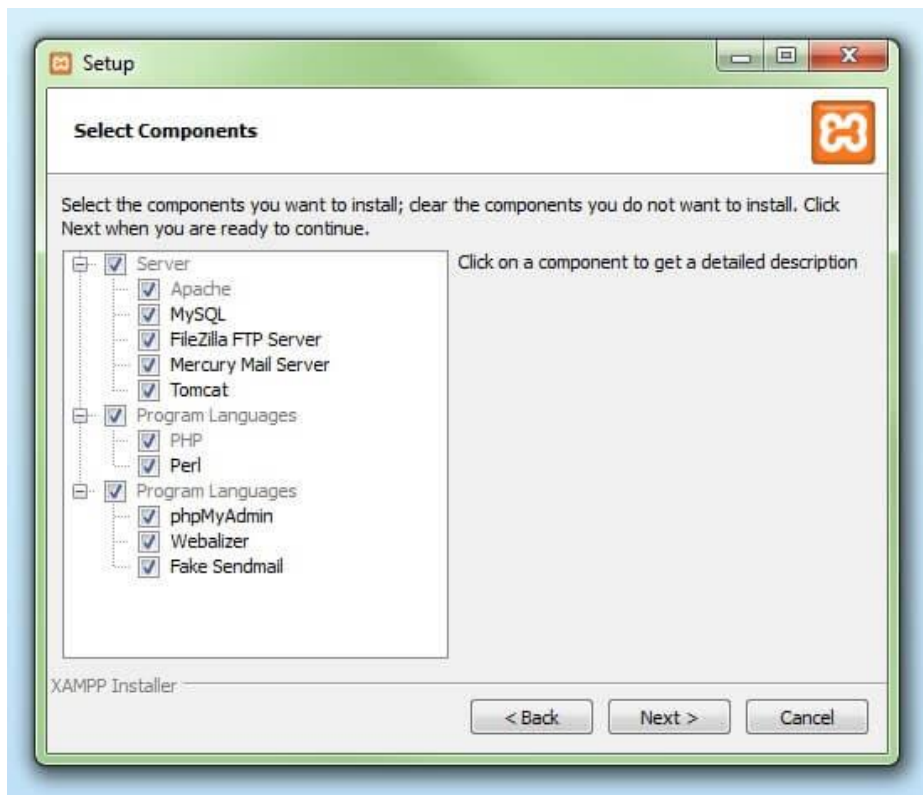
<https://www.apachefriends.org/download.html>

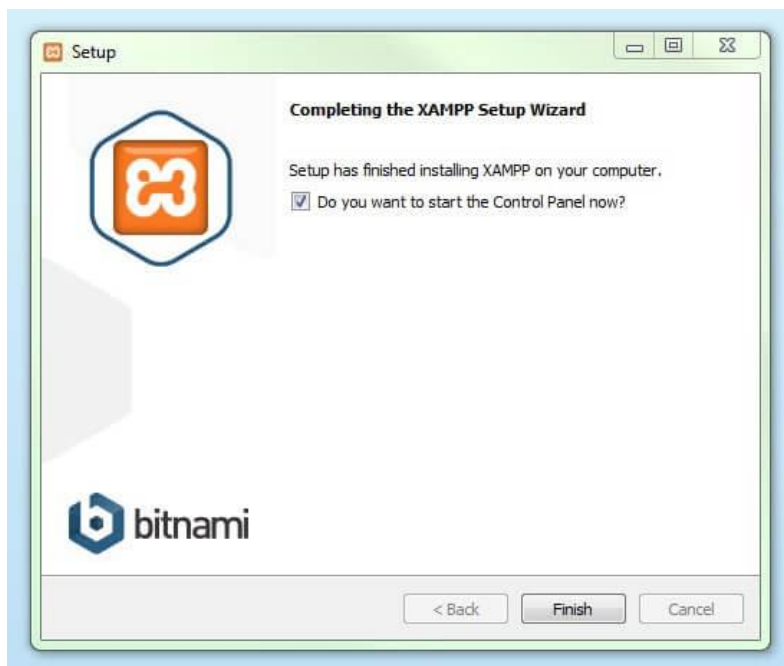
ဒေါင်းပြီး သွားရင် အောက်ပါအတိုင်း installation ကို အဆင့်ဆင့်ပြုလုပ်ပါ။ တစ်ခါတစ်လေမှာ xampp server installation မလုပ်ခင် antivirus တွေကို deactivate ခဏလုပ်ထားဖို့လိုပါတယ်။

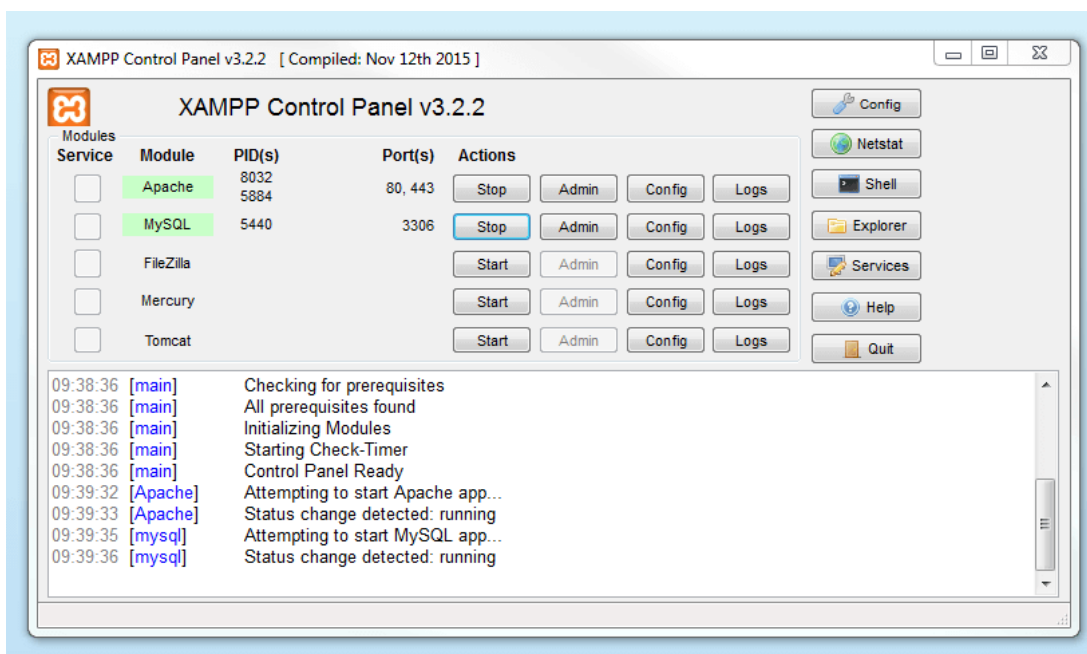
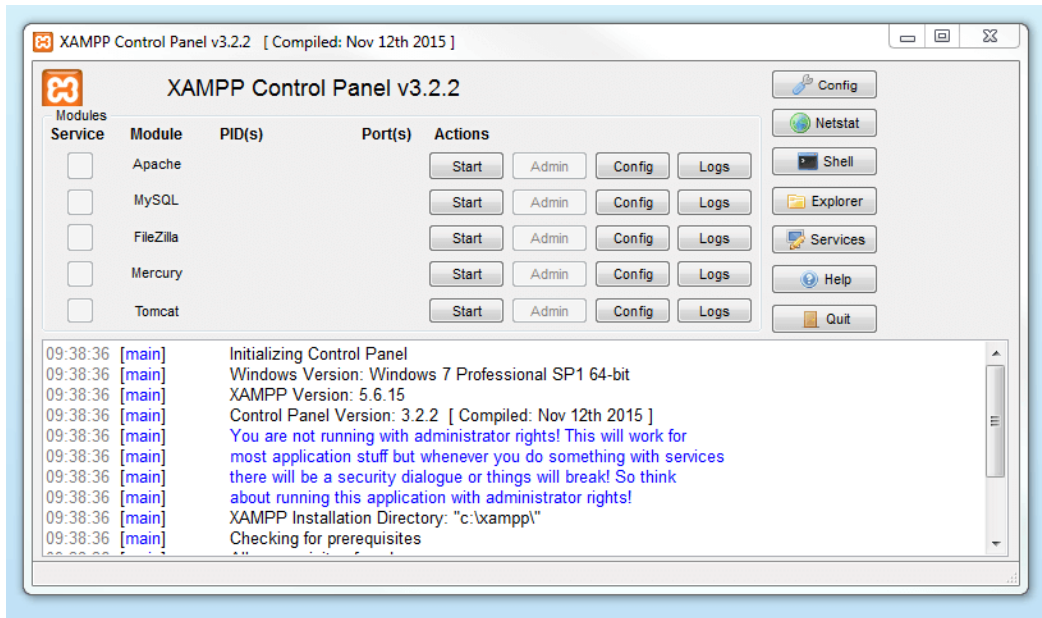


Before installing XAMPP, it is advisable to disable the anti-virus program temporarily







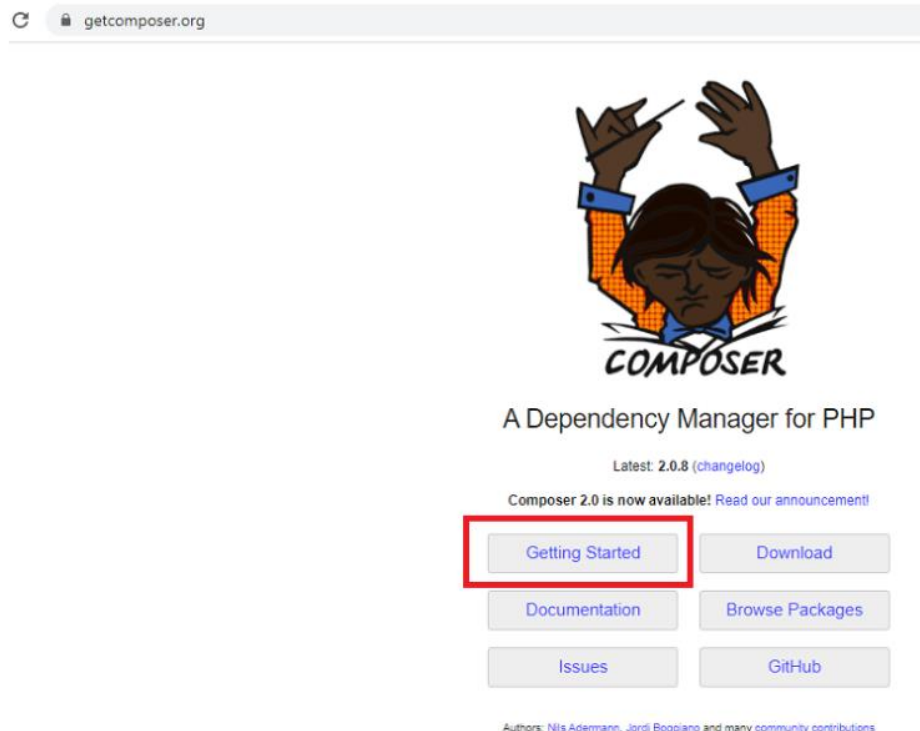


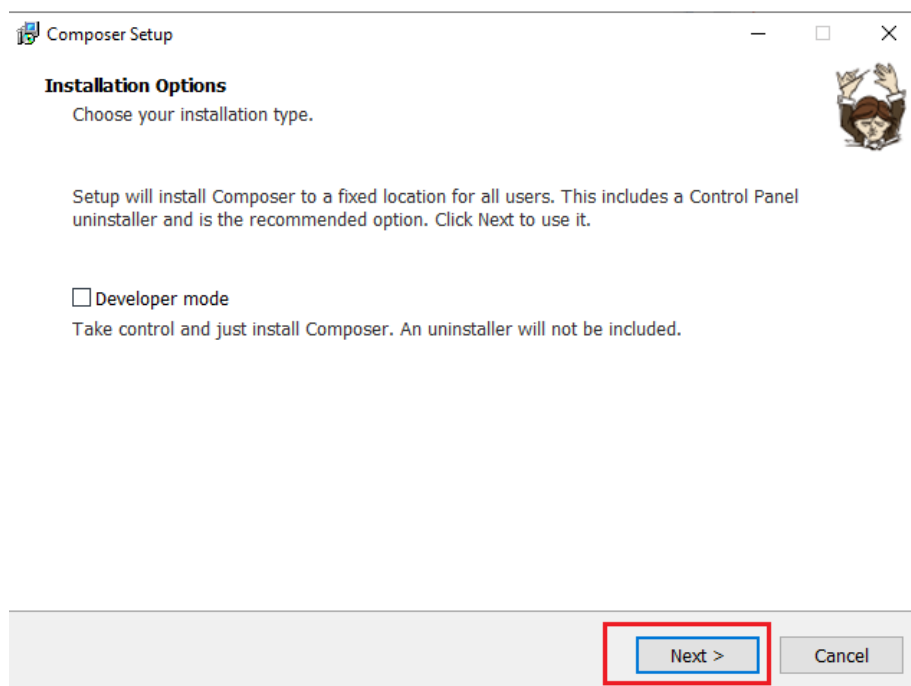
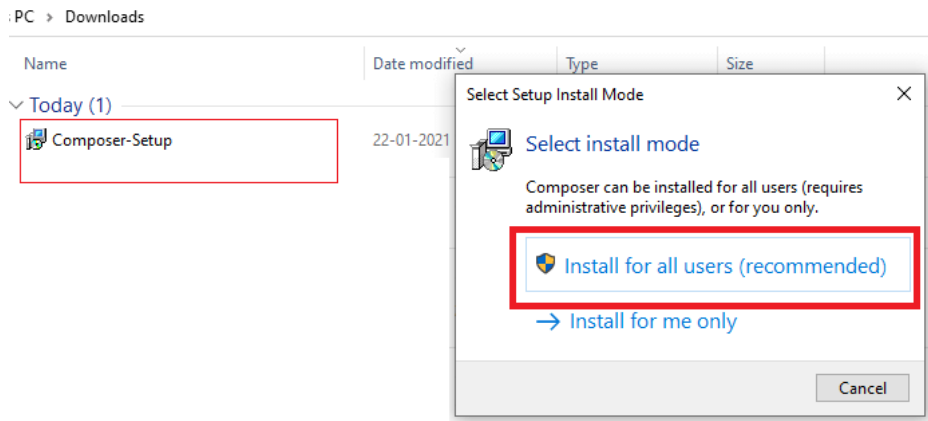
Composer Installation

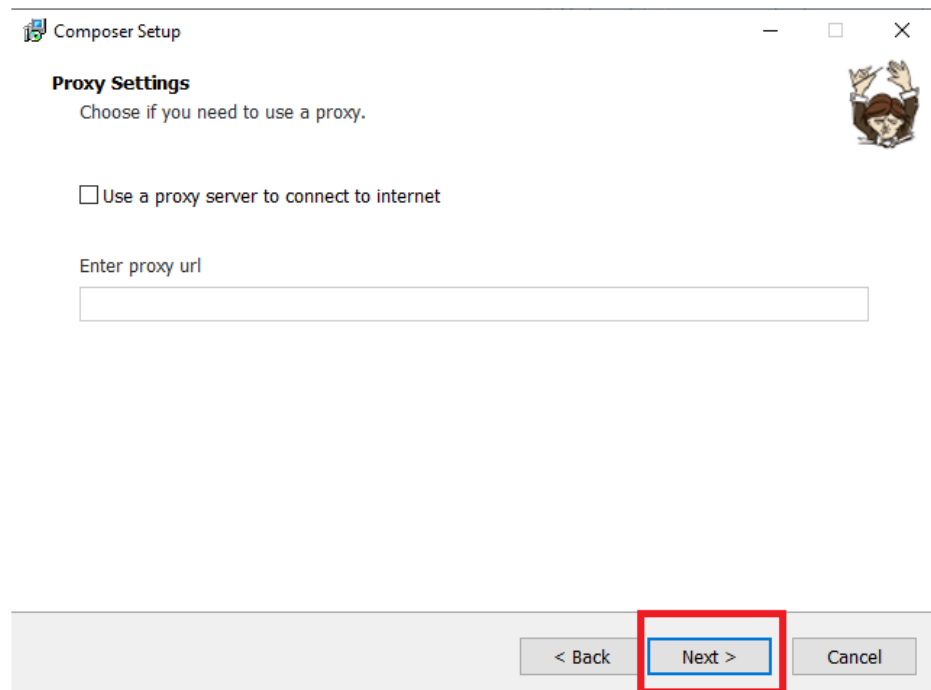
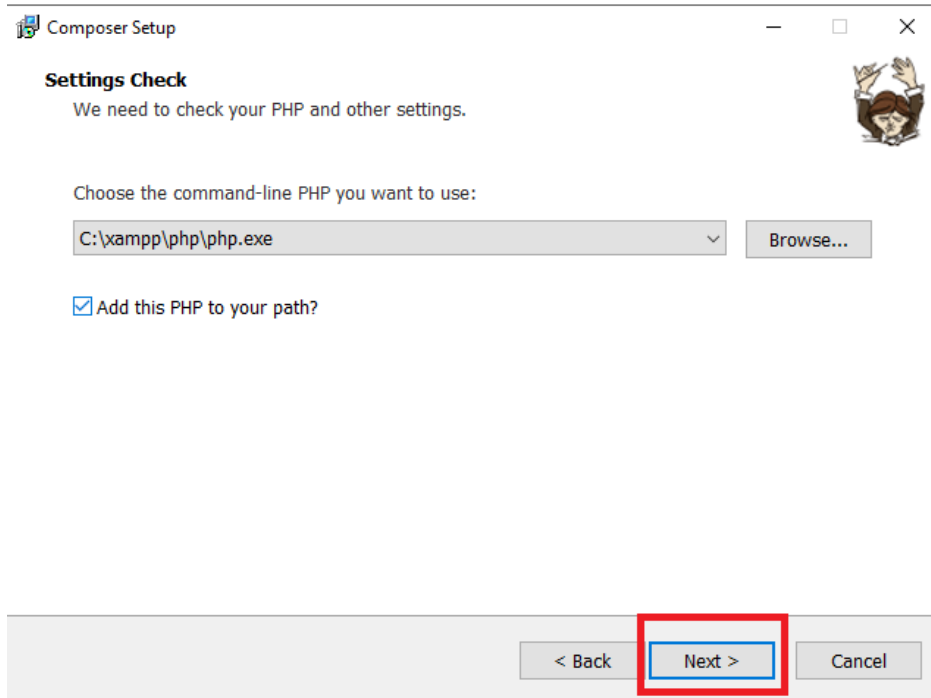
Laravel framework ကို install မလုပ်ခင် composer application ကို ကြို ပြီး install လုပ်ထားရပါမယ်။ composer application ရှိမှသာလျှင် Laravel framework ကို ဒေါင်းလို့ရမှာ ဖြစ်ပါတယ်။

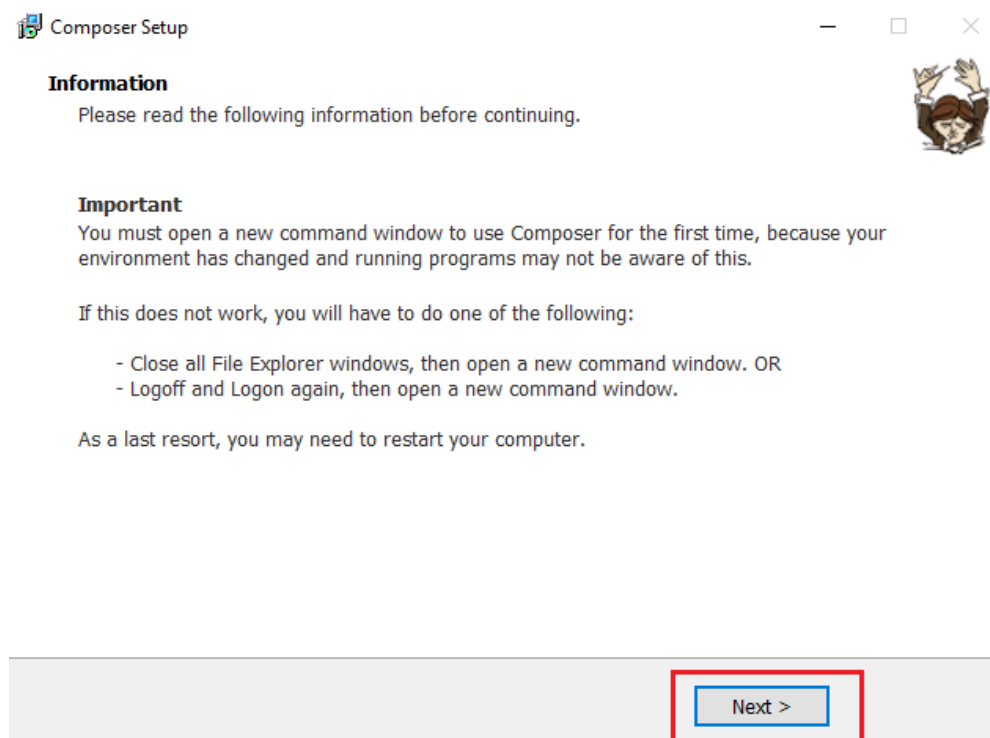
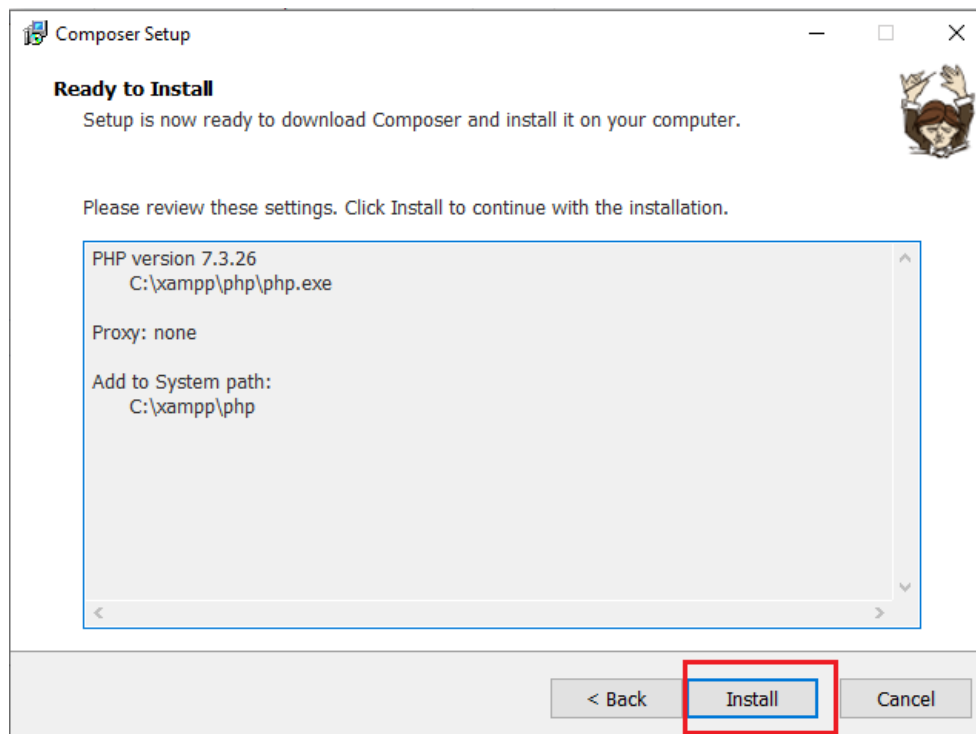
အောက်က download link ကိုသွားပြီး composer ကို ဒေါင်းပြီး အောက်ပါအတိုင်း installation ကို အဆင့်ဆင့် ပြုလုပ်ရပါမယ်။ ပြည်တွင်းကလူတွေကတော့ VPN ခံပြီးမှပဲ ဒေါင်းလို့ရမှာ ဖြစ်ပါတယ်။

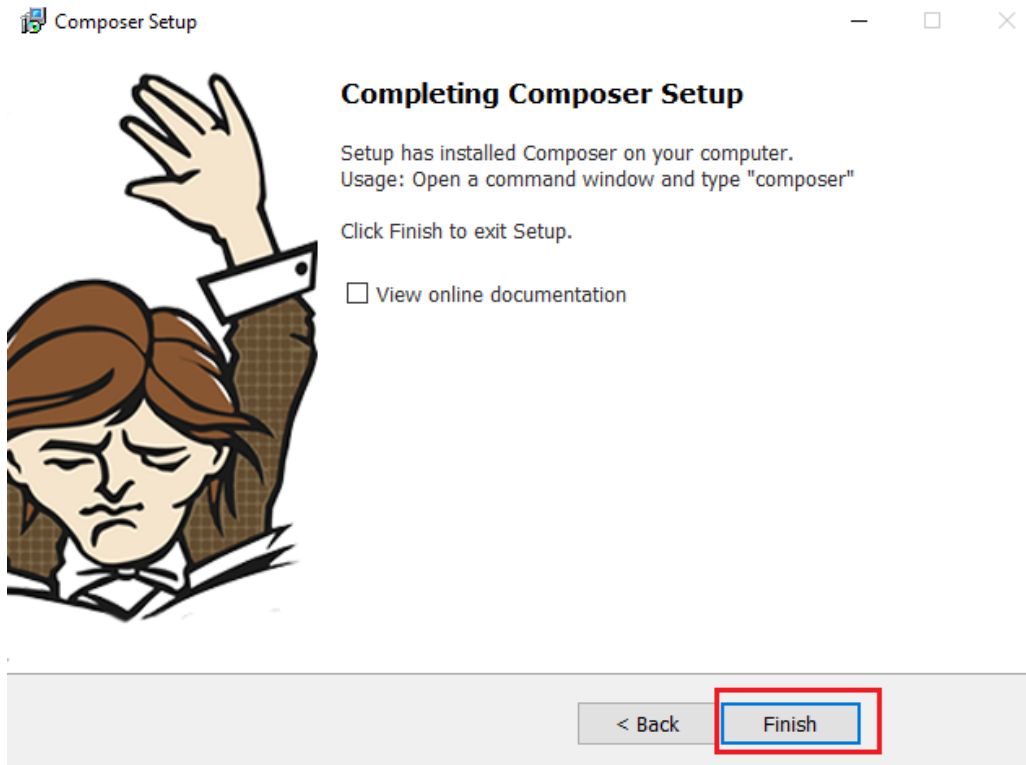
<https://getcomposer.org>



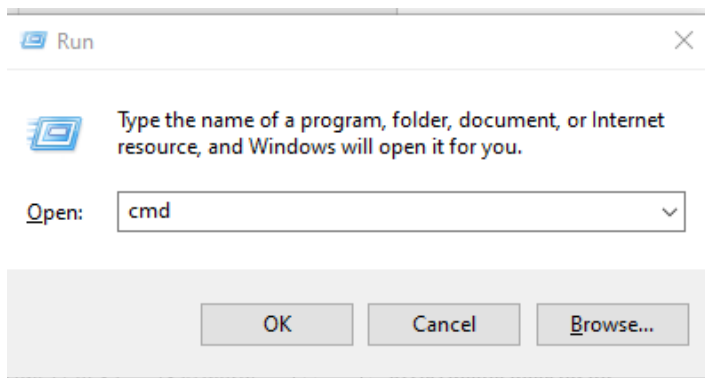








Composer ဒေါင်းပြီးသွားရင် အောက်ပါအတိုင်း command window ကို ဖွင့်ပြီး composer ကိုသုံးလိုရမရ စစ်ဆေးရမှာ ဖြစ်ပါတယ်။



C:\WINDOWS\system32\cmd.exe

Microsoft Windows [Version 10.0.19045.4412]
(c) Microsoft Corporation. All rights reserved.

C:\Users\soe wai moe>cd\

C:\>cd xampp

C:\xampp>cd htdocs

C:\xampp\htdocs>composer

Composer

Composer version 2.7.6 2024-05-04 23:03:15

Usage:

command [options] [arguments]

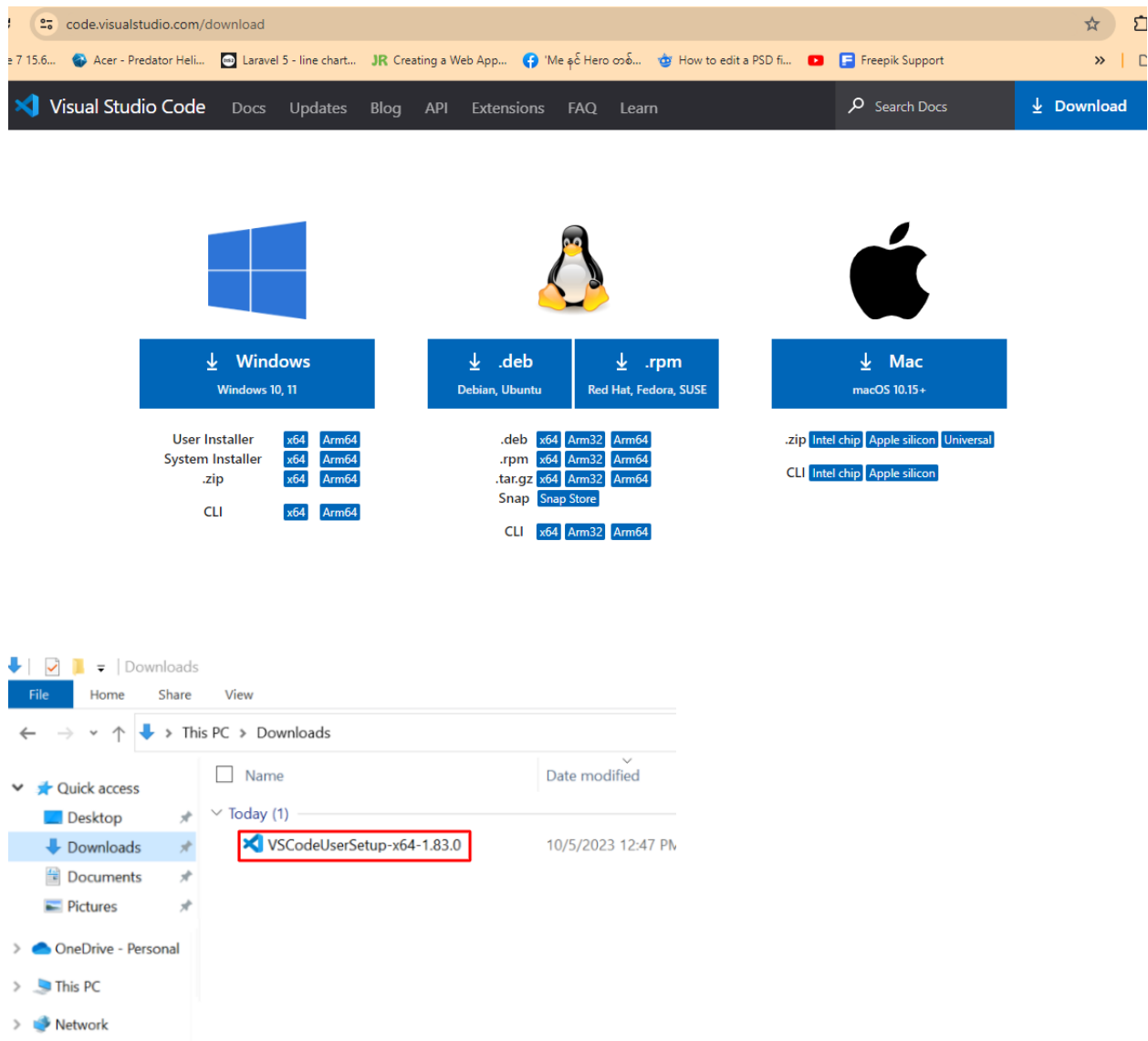
Options:

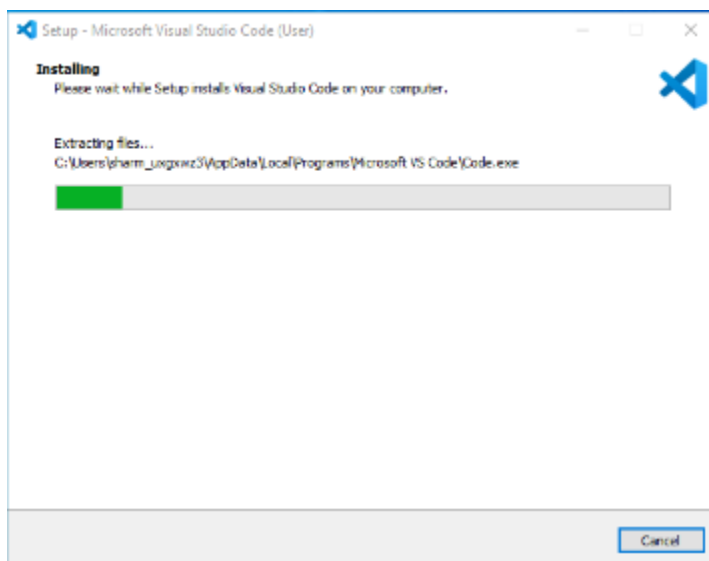
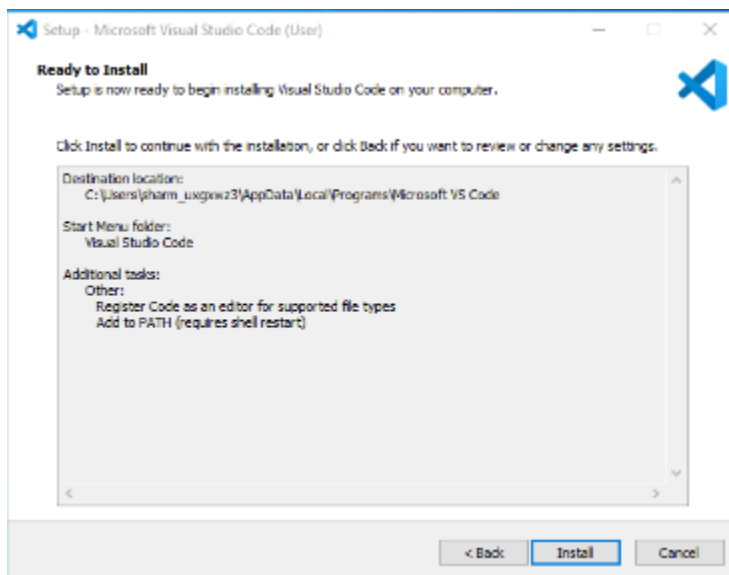
-h, --help Display help for the given command. When no command is specified, lists all available commands.
-q, --quiet Do not output any message

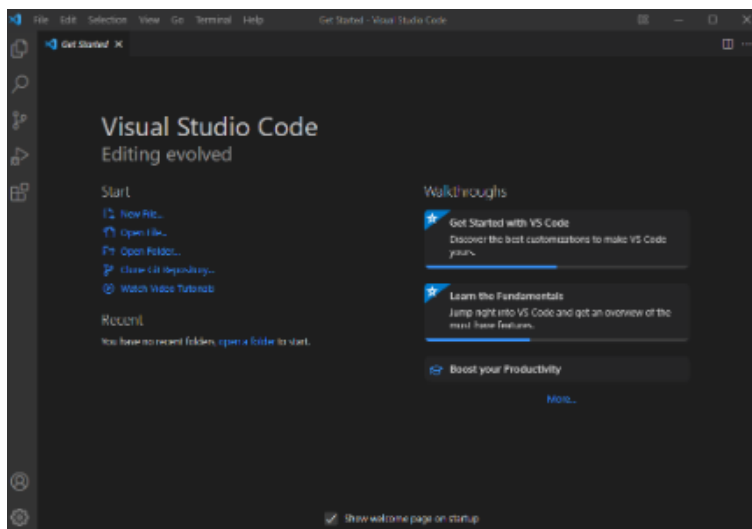
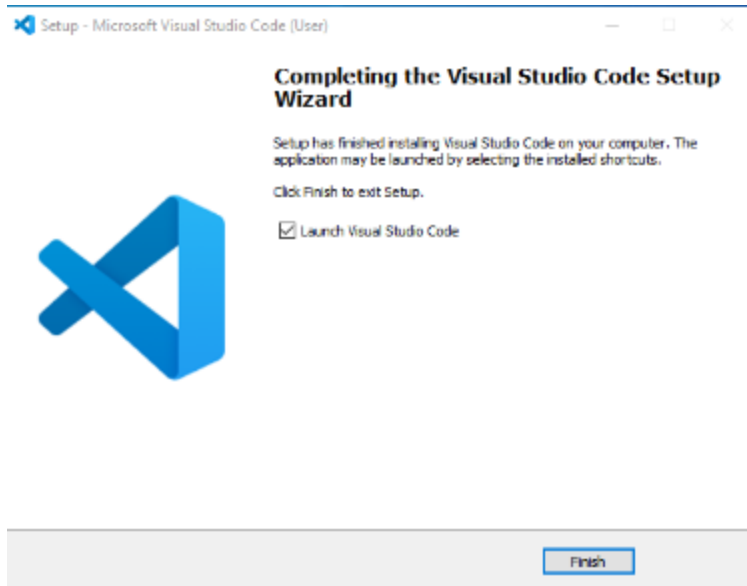
VSCode Installation

အောက်က download link ကိုသွားပြီး VSCode ကိုဒေါင်းရပါမယ်။ VSCode application Installation ကို အောက်ပါပုံတွေအတိုင်း အဆင့်ဆင့်လုပ်ဆောင်ရပါမယ်။

<https://code.visualstudio.com/download>







Running the first Laravel Application

အောက်မှာဖော်ပြထားတဲ့အတိုင်း Laravel project အသစ်တစ်ခုကို step by step တည်ဆောက်ရပါမယ်။

////////////////////////////////////

Step 1: Create a Laravel Project

////////////////////////////////////

VSCode မှာ terminal အသစ်တစ်ခုဖွင့်ပြီး အောက်ပါ command ကို ရိုက်ရပါမယ်။ run box ကနေ MS DOS command ဖွင့်ပြီး dos command ကနေလည်း ဆောက်လို့ရပါတယ်။

[command]

composer create-project laravel/laravel project-name

Replace project-name with the desired name for your project.

Step 2: Navigate to Your Project Directory

[command]

cd project-name

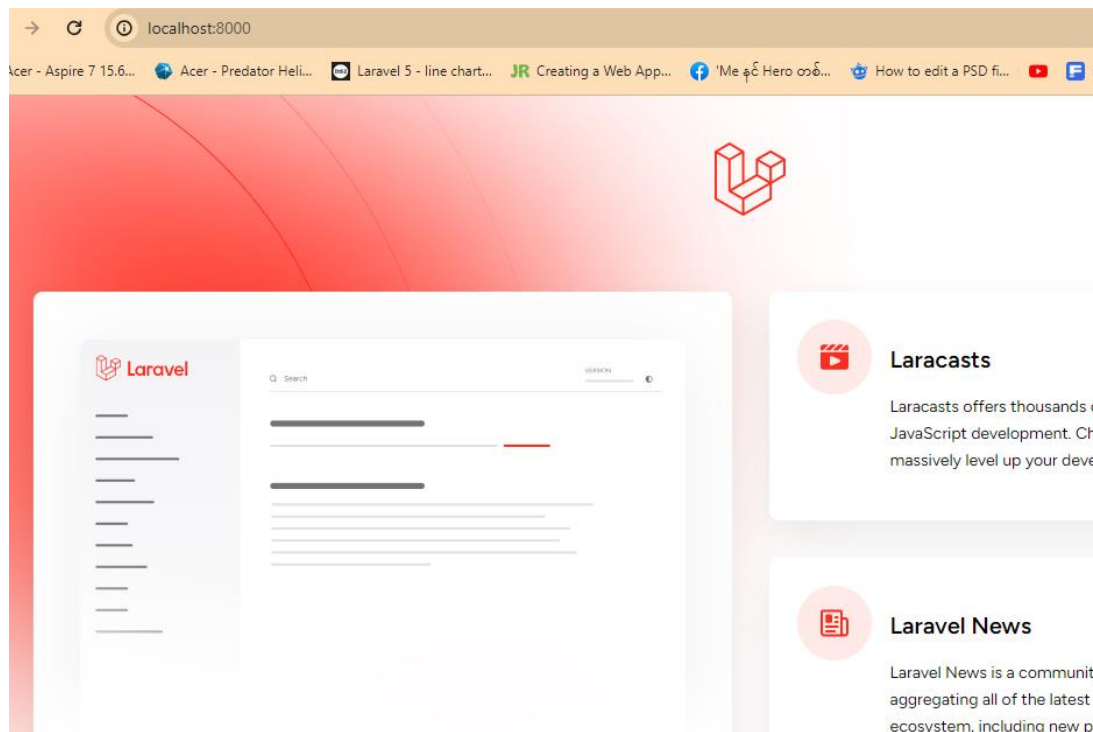
Step 3: Serve the Application

[command]

php artisan serve

This will start the server at <http://127.0.0.1:8000>.

⇒ Here is the output:



Congratulations! ပါ။ Laravel project အသစ်အောင်မြင်စွာ တည်ဆောက်ပြီးသွားပါပြီ။ Application တစ်ခုကို ရေးသားဖန်တီးဖို့ အဆင်သင့်ဖြစ်နေပါပြီ။

Laravel Components

Model-View-Controller (MVC)

1. Model:

- Model ဆိုတာက Data Unit တစ်ခု ဒါမှမဟုတ် app တစ်ခုလုံးရဲ့ basic business logic ကို ကိုယ်စားပြုထားပါတယ်။
- အခြေခံအားဖြင့် database table တွေကို ကိုယ်စားပြုထားပါတယ်။
- Table model တွေကို project ရဲ့ app/Models directory မှာ သိမ်းထားပါတယ်။

[code]:

```
namespace App\Models;
```

```
use Illuminate\Database\Eloquent\Factories\HasFactory;
```

```
use Illuminate\Database\Eloquent\Model;
```

```
class User extends Model
```

```
{
```

```
    use HasFactory;
```

```
    // Define fillable attributes
```

```
    protected $fillable = ['name', 'email', 'password'];
```

```
}
```

2. View:

- View က application ရဲ့ frontend အပိုင်းကို ကိုယ်စားပြုထားပါတယ်။
- Project ရဲ့ HTML, CSS, JavaScript file တွေကို ဒီ folder ထဲမှာ ထည့်ထားပါတယ်။
- View file တွေကို Project ရဲ့ resources/views directory မှာ သိမ်းထားပါတယ်။
- Laravel framework မှာ Blade templates တွေနဲ့ သိမ်းရပါတယ်။ file တွေကို blade extension တွေနဲ့ရေးပြီးသိမ်းရပါမယ်။

[code]:

```
<!-- resources/views/home.blade.php -->
<!DOCTYPE html>
<html>
<head>
  <title>Laravel</title>
</head>
<body>
  <h1>Welcome to Laravel 11</h1>
</body>
</html>
```

3. Controller:

- Frontend ကနေ လာတဲ့ User request နဲ့ response တွေကို handle လုပ်ပေးပါတယ်။

- Controller file တွေကို project ရဲ့ app/Http/Controllers directory မှာ သိမ်းထားပါတယ်။
- Application logic ၊ data processing အားလုံးကို controller ကတာဝန်ယူရပါတယ်။

[code]:

```
namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller
{
    public function index()
    {
        return view('home');
    }
}
```

Routes

- ကျောင်းသားတွေတော်တော်များများ MVC မှာပဲ အာရုံစိုက်လေ့လာပြီး Routing မှာ တိုင်ပါတ်နေတဲ့သူတွေအများကြီးရှိပါတယ်။ Routing မှာ route security ပိုင်းနဲ့ middleware ပိုင်းတွေက ကျနော်တို့ကို ဒုက္ခပေးနေတတ်ပါတယ်။ routes/web.php file မှာ သွားပြီး app url တွေကို control လုပ်ရတာ ဖြစ်ပါတယ်။

Example:

[code]:

```
use App\Http\Controllers\HomeController;

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

This route maps the root URL (/) to the index method of the HomeController.

.env File

The .env file contains environment-specific configuration, such as database credentials, application URL, and more. This file is located in the root directory of your Laravel project.

Example of a .env file:

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:YOUR_APP_KEY_HERE
APP_DEBUG=true
APP_URL=http://localhost
```

```
LOG_CHANNEL=stack
```

```
DB_CONNECTION=sqlite
#DB_HOST=127.0.0.1
#DB_PORT=3306
#DB_DATABASE=your_database_name
#DB_USERNAME=your_username
#DB_PASSWORD=your_password
```

```
BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
```

SESSION_LIFETIME=120

Database Setup

Step 1: Configure the Database

Edit the .env file to include your database configuration:

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=laravel_db

DB_USERNAME=root

DB_PASSWORD=

Replace laravel_db, root, and password with your actual database name, username, and password.

Step 2: Create a Migration

Run the following command to create a new migration file:

[command]:

php artisan make:migration create_users_table

This will create a migration file in the database/migrations directory.

Step 3: Define the Migration

Edit the generated migration file to define the schema for the users table:

[code]:

```
use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateUsersTable extends Migration
{
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }

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

Step 4: Run the Migration

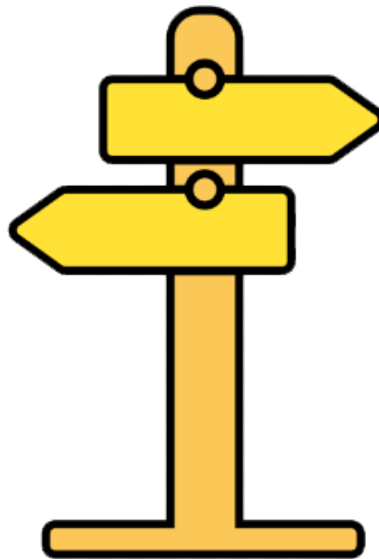
Run the following command to execute the migration and create the table in the database:

[command]:

php artisan migrate

Routing

📺 Laravel Video Lesson 2

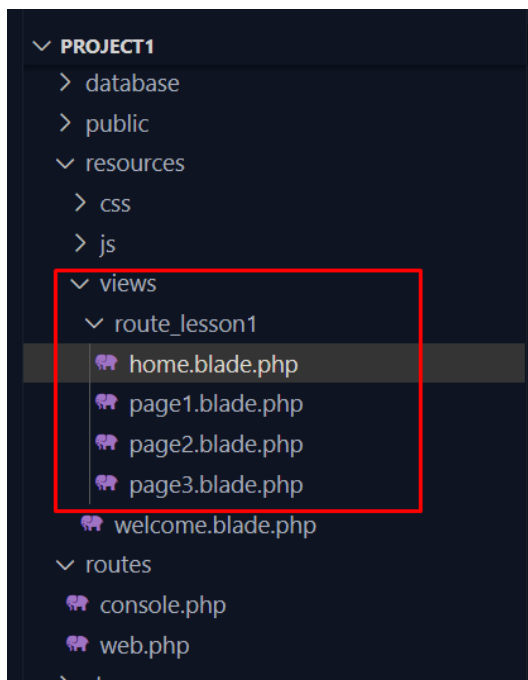


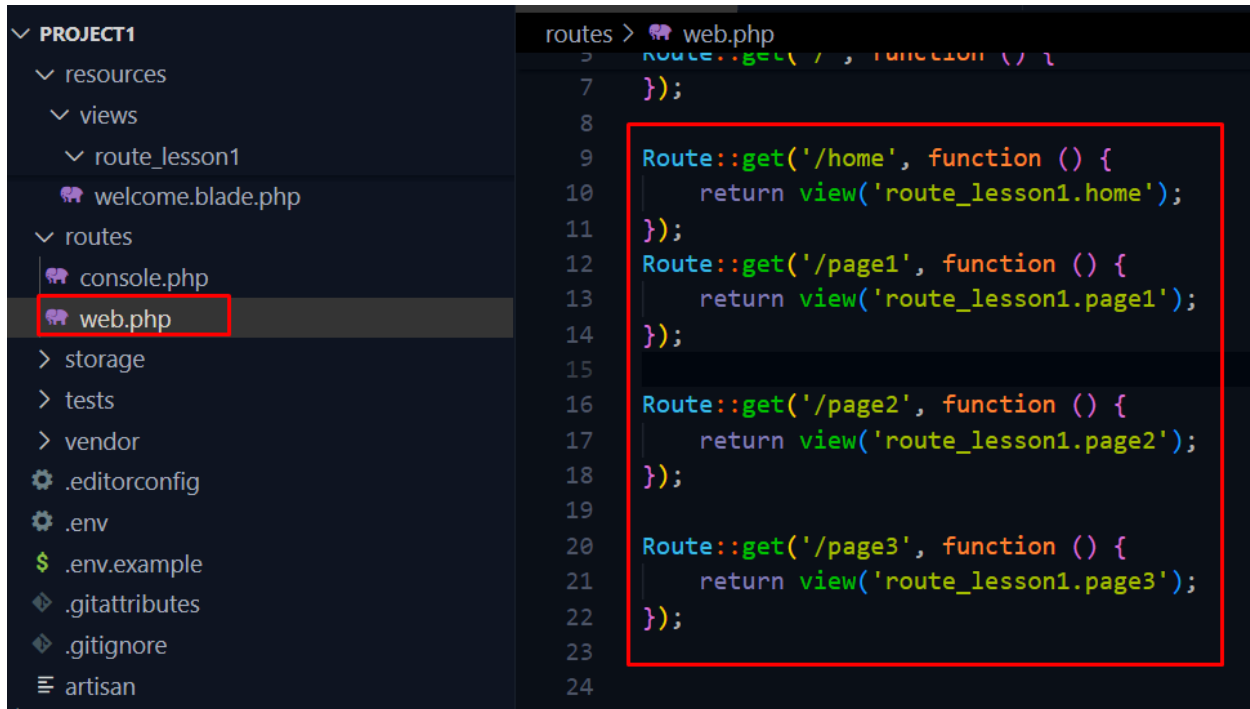
Routing in Laravel –

- Basic Routing
- Route parameters
- Named Routes
- Checking route list

Basic Routing

app/routes.php file ထဲမှာ basic routing ကို function နဲ့ ရေးကြပါမယ်။ basic routing ကို နည်းနည်း သဘောပေါက်သွားရင်တော့ Controller သုံးပြီး ရေးကြပါမယ်။ Controller နဲ့ သုံးတတ်သွားပြီဆို route name တွေပေးပြီး routing တွေကို implement လုပ်ရပါမယ်။ ရေးရင်ရေးရင်း project scope ကြီးလာပြီ ဆိုရင် route list ကို မကြာခဏထုတ် ကြည့်ပြီး ဆက်ရေးသွားရမှာ ဖြစ်ပါတယ်။





```
routes > web.php
5  Route::get('/', function () {
6      return view('welcome');
7  });
8
9  Route::get('/home', function () {
10     return view('route_lesson1.home');
11 });
12 Route::get('/page1', function () {
13     return view('route_lesson1.page1');
14 });
15
16 Route::get('/page2', function () {
17     return view('route_lesson1.page2');
18 });
19
20 Route::get('/page3', function () {
21     return view('route_lesson1.page3');
22 });
23
24
```

=>routes/web.php

```
Route::get('/home', function () {
    return view('route_lesson1.home');
});
```

```
Route::get('/page1', function () {
    return view('route_lesson1.page1');
});
```

```
Route::get('/page2', function () {
    return view('route_lesson1.page2');
});
```

```
Route::get('/page3', function () {
    return view('route_lesson1.page3');
});
```


Run the project as the following:

[command]:

php artisan serv



Home Page

Laravel Basic Route

[Home](#) | [Page 1](#) | [Page 2](#) | [Page 3](#)

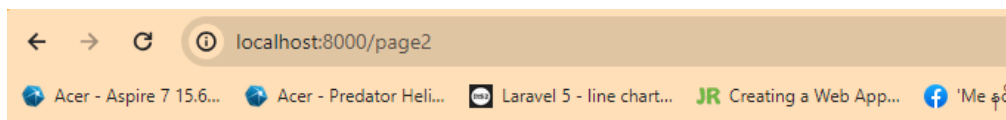
Lorem ipsum dolor sit amet consectetur adipisicing elit. Ab veniam sit, alias iste id recusandae culpa eius blanditiis aliquam accusamus dolorum placeat magni minus! Laborum, corporis adipisci pariatur aliquam maiores autem ut i atque illum sequi ab ut provident facilis. Nisi doloremque recusandae distinctio minus culpa? Possimus nostrum qu fugit veniam consequatur laborum, illum nesciunt officiis ducimus doloribus! Placeat sapiente deleniti dolorum iste voluptates ut modi provident. Natus, error unde nostrum accusantium deleniti, facilis blanditiis excepturi consequa nobis quasi possimus magni voluptatem at. Ipsum debitis repudiandae, doloremque perferendis quo eos! Ex dolore accusamus perferendis fugiat. Totam voluptas quam deserunt libero dolorem nobis, aliquam quae corrupti ipsum n necessitatibus atque unde corporis laborum minus libero, delectus ut dolores amet eius cumque alias reiciendis nisi totam vero minus, optio asperiores. Tempora vel cupiditate nesciunt, quae voluptates id pariatur dignissimos volupt possimus incidunt? Consequatur a quam et accusantium dignissimos, fugiat nobis nulla, velit voluptatem, atque ver ex velit laboriosam magnam iure, cumque aliquam voluptas dolorum maiores harum? Sapiente numquam doloribu



This is Page 1

[Home](#) | [Page 1](#) | [Page 2](#) | [Page 3](#)

Lorem ipsum dolor sit amet consectetur adipisicing elit. Ab veniam sit, alias iste id recusandae culpa aliquam accusamus dolorum placeat magni minus! Laborum, corporis adipisci pariatur aliquam mai atque illum sequi ab ut provident facilis. Nisi doloremque recusandae distinctio minus culpa? Possit fugit veniam consequatur laborum, illum nesciunt officiis ducimus doloribus! Placeat sapiente deler voluptates ut modi provident. Natus, error unde nostrum accusantium deleniti, facilis blanditiis exce nobis quasi possimus magni voluptatem at. Ipsum debitis repudiandae, doloremque perferendis quo accusamus perferendis fugiat. Totam voluptas quam deserunt libero dolorem nobis, aliquam quae cc necessitatibus atque unde corporis laborum minus libero, delectus ut dolores amet eius cumque alias totam vero minus, optio asperiores. Tempora vel cupiditate nesciunt, quae voluptates id pariatur digi possimus incident? Consequatur a quam et accusantium dignissimos, fugiat nobis nulla, velit volupt ex velit laboriosam magnam iure, cumque aliquam voluptas dolorum maiores harum? Sapiente num



This is Page 2

[Home](#) | [Page 1](#) | [Page 2](#) | [Page 3](#)

Lorem ipsum dolor sit amet consectetur adipisicing elit. Ab veniam sit, alias iste id recusandae culpa eius aliquam accusamus dolorum placeat magni minus! Laborum, corporis adipisci pariatur aliquam maiores at atque illum sequi ab ut provident facilis. Nisi doloremque recusandae distinctio minus culpa? Possimus no fugit veniam consequatur laborum, illum nesciunt officiis ducimus doloribus! Placeat sapiente deleniti dol voluptates ut modi provident. Natus, error unde nostrum accusantium deleniti, facilis blanditiis excepturi c nobis quasi possimus magni voluptatem at. Ipsum debitis repudiandae, doloremque perferendis quo eos! E accusamus perferendis fugiat. Totam voluptas quam deserunt libero dolorem nobis, aliquam quae corrupti necessitatibus atque unde corporis laborum minus libero, delectus ut dolores amet eius cumque alias reicie totam vero minus, optio asperiores. Tempora vel cupiditate nesciunt, quae voluptates id pariatur dignissim possimus incident? Consequatur a quam et accusantium dignissimos, fugiat nobis nulla, velit voluptatem, ex velit laboriosam magnam iure, cumque aliquam voluptas dolorum maiores harum? Sapiente numquam

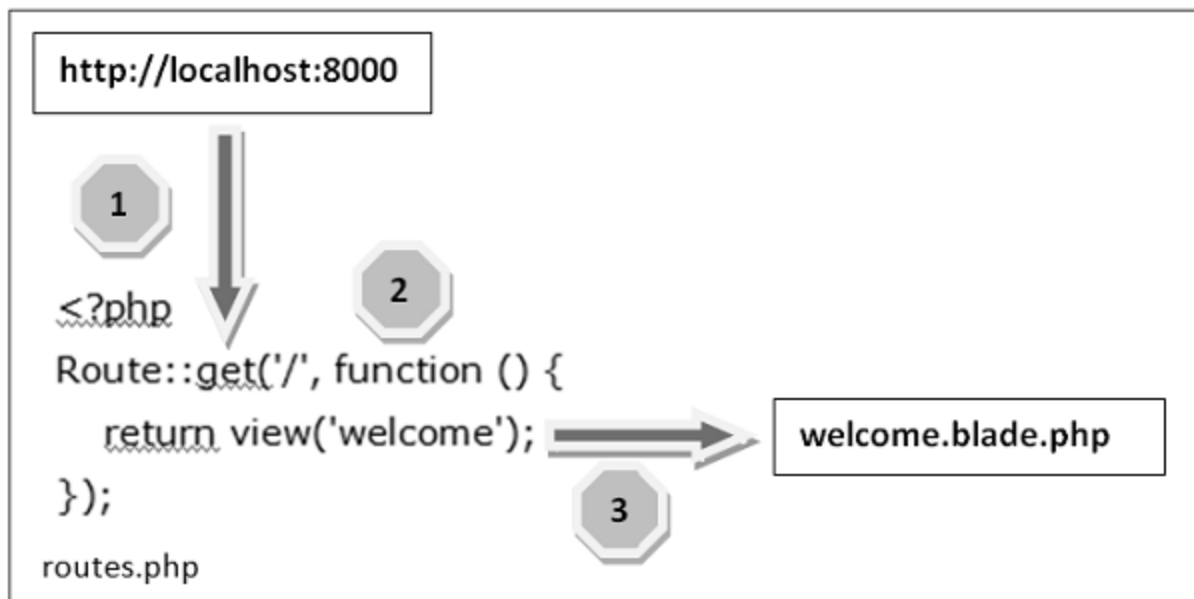


This is Page 3

[Home](#) | [Page 1](#) | [Page 2](#) | [Page 3](#)

Lorem ipsum dolor sit amet consectetur adipisicing elit. Ab veniam sit, alias iste id recusandae culpa eius blanditi aliquam accusamus dolorum placeat magni minus! Laborum, corporis adipisci pariatur aliquam maiores autem ut atque illum sequi ab ut provident facilis. Nisi doloremque recusandae distinctio minus culpa? Possimus nostrum < fugit veniam consequatur laborum, illum nesciunt officiis ducimus doloribus! Placeat sapiente deleniti dolorum is voluptates ut modi provident. Natus, error unde nostrum accusantium deleniti, facilis blanditiis excepturi consequ nobis quasi possimus magni voluptatem at. Ipsum debitis repudiandae, doloremque perferendis quo eos! Ex dolor accusamus perferendis fugiat. Totam voluptas quam deserunt libero dolorem nobis, aliquam quae corrupti ipsum : necessitatibus atque unde corporis laborum minus libero, delectus ut dolores amet eius cumque alias reiciendis ni totam vero minus, optio asperiores. Tempora vel cupiditate nesciunt, quae voluptates id pariatur dignissimos volu possimus incidunt? Consequatur a quam et accusantium dignissimos, fugiat nobis nulla, velit voluptatem, atque v ex velit laboriosam magnam iure, cumque aliquam voluptas dolorum maiores harum? Sapiente numquam dolorib

The routing mechanism is shown in the image given below –



```

13
14 Route::get('/page/{id}', function ($id) {
15
16     if($id==1){
17         return view('route_lesson1.page1');
18     }
19     else if($id==2){
20         return view('route_lesson1.page2');
21     }
22     else if($id==3){
23         return view('route_lesson1.page3');
24     }
25
26
27 });
28
29
30

```

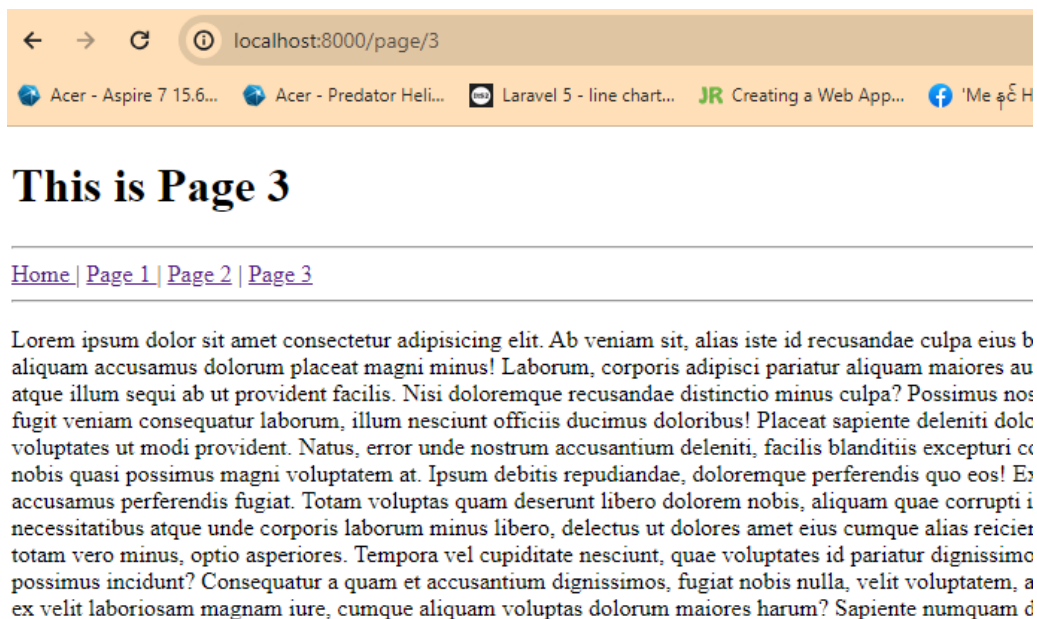
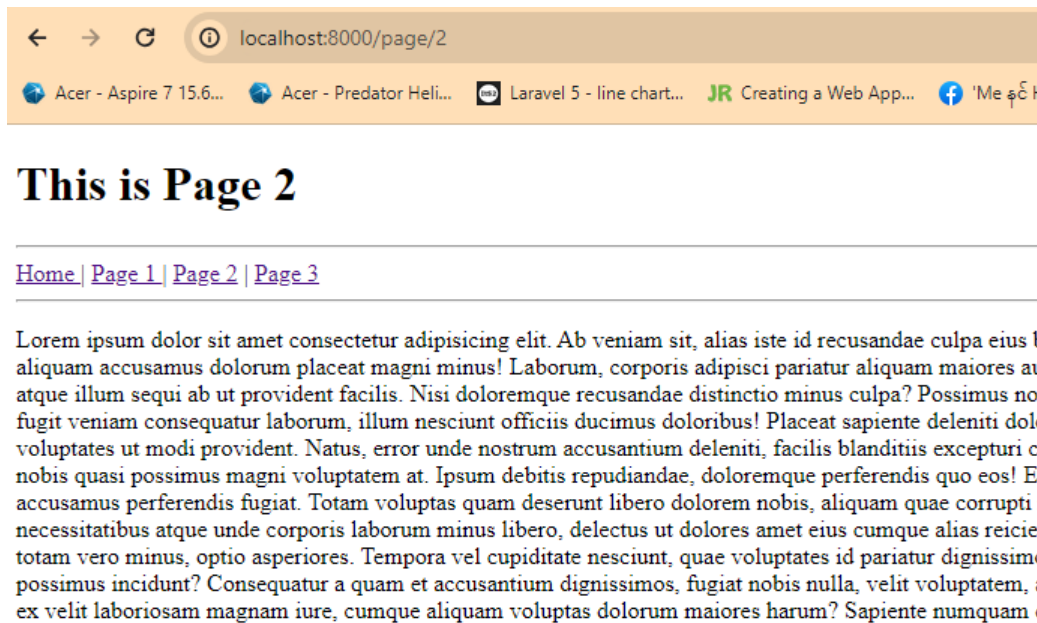
Output:



This is Page 1

[Home](#) | [Page 1](#) | [Page 2](#) | [Page 3](#)

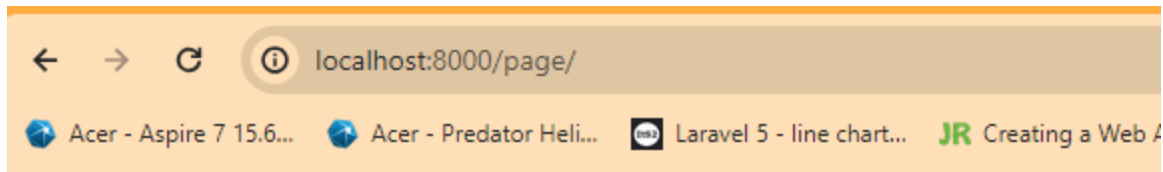
Lorem ipsum dolor sit amet consectetur adipisicing elit. Ab veniam sit, alias iste id recusandae cū aliquam accusamus dolorum placeat magni minus! Laborum, corporis adipisci pariatur aliquam r atque illum sequi ab ut provident facilis. Nisi doloremque recusandae distinctio minus culpa? Po fugit veniam consequatur laborum, illum nesciunt officiis ducimus doloribus! Placeat sapiente de voluptates ut modi provident. Natus, error unde nostrum accusantium deleniti, facilis blanditiis e nobis quasi possimus magni voluptatem at. Ipsum debitis repudiandae, doloremque perferendis q accusamus perferendis fugiat. Totam voluptas quam deserunt libero dolorem nobis, aliquam quae necessitatibus atque unde corporis laborum minus libero, delectus ut dolores amet eius cumque a totam vero minus, optio asperiores. Tempora vel cupiditate nesciunt, quae voluptates id pariatur c possimus incidunt? Consequatur a quam et accusantium dignissimos, fugiat nobis nulla, velit vol ex velit laboriosam magnam iure, cumque aliquam voluptas dolorum maiores harum? Sapiente n



Optional Parameters

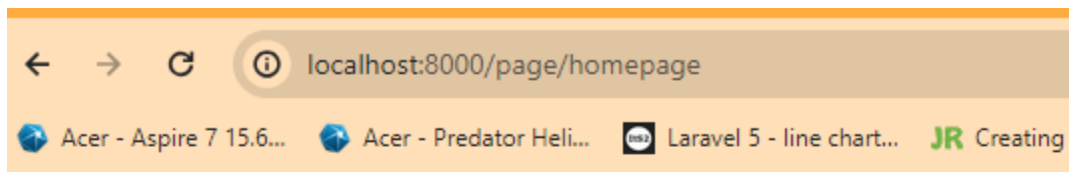
```
Route::get('user/{name?}', function ($name = 'NorthernCity') { return $name;});
```

Without parameter:



Name =Northern City

With parameter:



Name =homepage

Named Routes

[command]:

```
php artisan make:controller PageController
```

routes>web.php

```
Route::get('home',[PageController::class,'home'])->name('page.home');
```

```
Route::get('page1',[PageController::class,'page1'])->name('page.p1');
```

```
Route::get('page2',[PageController::class,'page2'])->name('page.p2');
```

```
Route::get('page2',[PageController::class,'page3'])->name('page.p3');
```

Show Route List

[command]

```
php artisan route:list
```

```
PS C:\xampp\htdocs\project1> php artisan route:list

GET|HEAD      / ..... page.home > PageController@home
GET|HEAD      home ..... page.p1 > PageController@page1
GET|HEAD      page1 ..... page.p2 > PageController@page2
GET|HEAD      page2 ..... page.p3 > PageController@page3
GET|HEAD      page3 .....
GET|HEAD      up .....

Showing [6] routes
```

Apply route names to your project. Update your current codes as the following:

```
<a href="{{route('page.home')}}"> Home </a> |
```

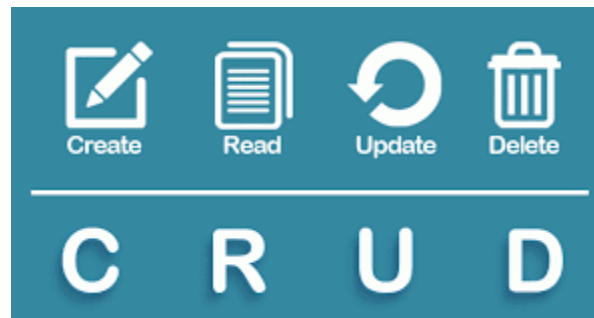
```
<a href="{{route('page.p1')}}"> Page 1 </a> |
```

```
<a href="{{route('page.p2')}}"> Page 2</a> |
```

```
<a href="{{route('page.p3')}}"> Page 3</a>
```

CRUD Part-1

📺 Video Lesson 3



CRUD Part-1

ဒီအပိုင်းမှာ database ထဲကို data အသွင်းအထုတ် ဆောင်ရွက်တာကို လေ့ကျင့်သွားမှာ ဖြစ်ပါတယ်။ Laravel project installation လုပ်ပြီးသွားပြီဆိုရင် အောက်ပါအတိုင်းအဆင့်ဆင့် လိုက်လုပ်ကြည့်ပါ။

////////////////////////////////////

Step-1: Project installation

////////////////////////////////////

[command]

php artisan create-project crud-app

////////////////////////////////////

Step-2: Database Config

////////////////////////////////////

.env

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

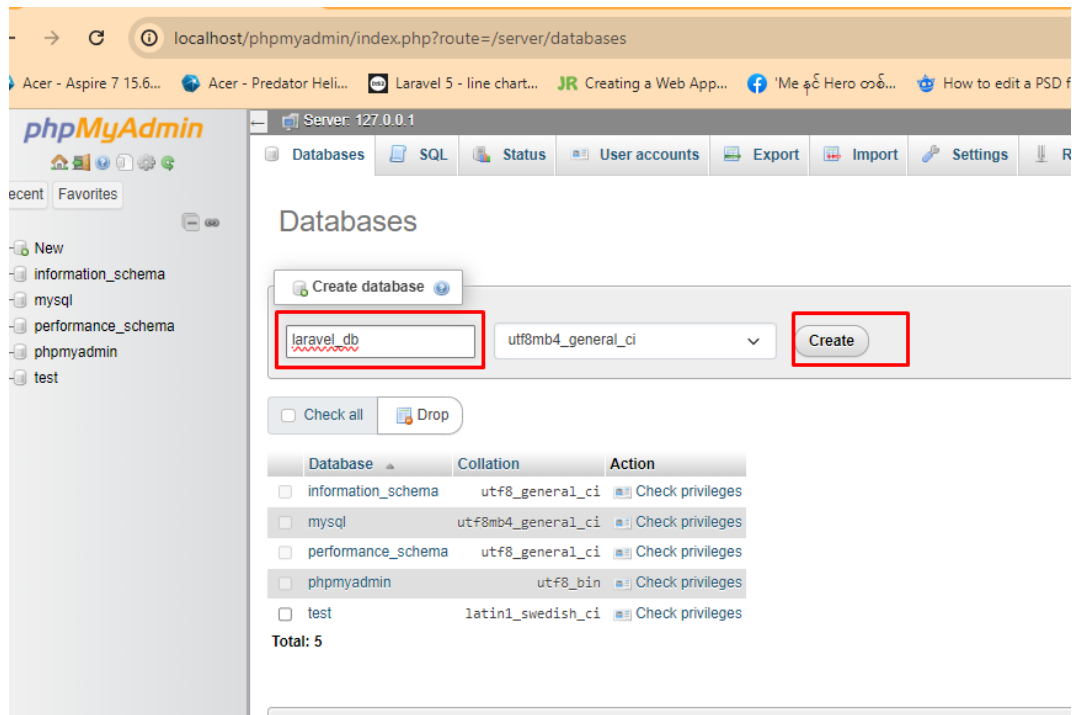
DB_DATABASE=laravel_db

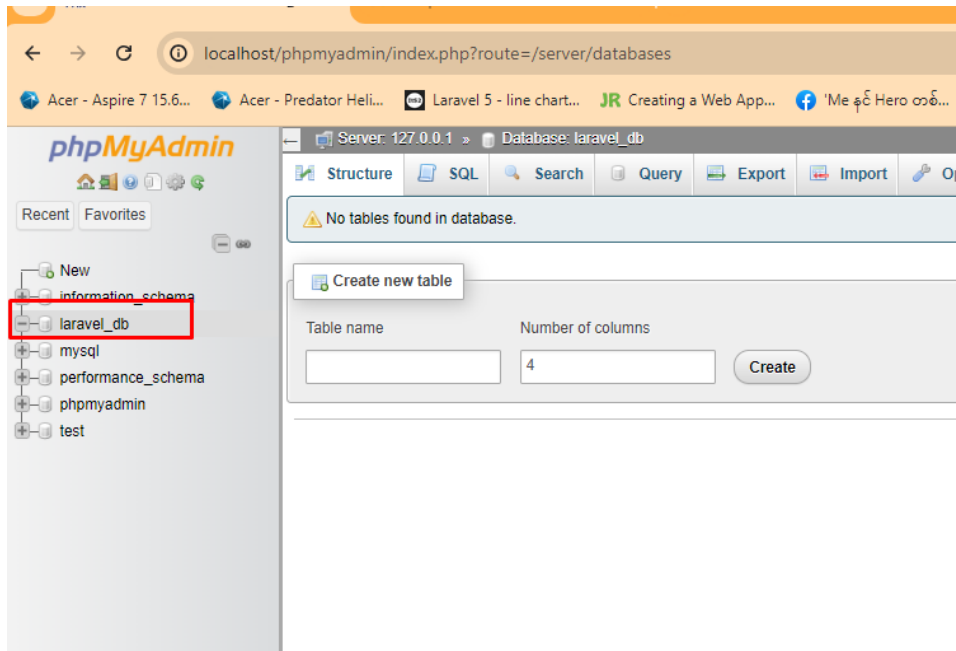
DB_USERNAME=root

DB_PASSWORD=

.env file ထဲမှာ database setup ကို config လုပ်ပြီးသွားရင်တော့ local database server ကို သွားပြီး Laravel_db ဆိုတဲ့ database တစ်ခု သွားဆောက်ပေးရပါမယ်။ database အခွန်ပဲဆောက်ပေးရုံသက်သက်ပါ။ ကျန်တဲ့ database ထဲက table file တွေအားလုံးကို Laravel migration table file ကနေ control လုပ်ပြီး

auto generate လုပ်သွားမှာ ဖြစ်ပါတယ်။ ဒီလို Laravel php code ကနေ control လုပ်ပြီး auto generate လုပ်ပေးတာကို migrate လုပ်တယ်လို့ခေါ်ပါတယ်။ အခြားသော java တို့ C# တွေမှာ တော့ persistence လုပ်တယ်လို့ခေါ်ပါတယ်။ ဒါဆို Laravel_db ကို local database server ရှိတဲ့ xampp server ကို သွားပြီး အောက်ပါအတိုင်း manual သွားဆောက်လိုက်ရအောင်နော်။





Database configuration ပြီးသွားပြီးဆိုတော့ Model, Controller နဲ့ Migration table ဆောက်လို့ရပါပြီ။

////////////////////////////////////

Step-3: Create Model, Controller and Table Migration

////////////////////////////////////

[command]

php artisan make:model Product -m -r

ဒီ php artisan command က တစ်ပြိုင်တည်း file သုံးခုဆောက်လိုက်တာ ဖြစ်ပါတယ်။

Product Model file, ProductController File နဲ့ database migration folder ထဲမှာ products_table file

တွေကို generate လုပ်လိုက်တာဖြစ်ပါတယ်။

-m => migration လို့အဓိပ္ပါယ်ရပါတယ်။

-r => resource လို့အဓိပ္ပါယ်ရပါတယ်။

```
////////////////////////////////////
```

Step-4: Database Migration table configuration

```
////////////////////////////////////
```

=> database/migrations/

```
public function up(): void
```

```
{
```

```
    Schema::create('products', function (Blueprint $table) {
```

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

```
        $table->string('name');
```

```
        $table->string('price');
```

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

```
    });
```

```
}
```

```
////////////////////////////////////
```

Step-5: Model Config

```
////////////////////////////////////
```

=> App\Models

```
class Product extends Model
```

```
{
```

```
    use HasFactory;
```

```
    protected $fillable = [
```

```
        'name',
```

```
        'price',
```

```
    ];
```

```
}
```

Model Configuration ပြီးသွားရင် Controller ကိုသွားပြီး data control function တွေမှာ data control code နဲ့ redirect code တွေသွားရေးကြပါတယ်။

```
////////////////////////////////////
```

Step-6: ProductController configuration

```
////////////////////////////////////
```

```
=> App\Http\Controllers\ProductController
```

```
use App\Models\Product;
```

```
use Illuminate\Http\Request;
```

```
class ProductController extends Controller
```

```
{
```

```
    /**
```

```
     * Display a listing of the resource.
```

```
     */
```

```
    public function index()
```

```
    {
```

```
        $products=Product::all();
```

```
        return view('products.index',compact('products'));
```

```
    }
```

```
    /**
```

```
     * Show the form for creating a new resource.
```

```
     */
```

```
    public function create()
```

```
    {
```

```

        return view('products.create');
    }

    /**
     * Store a newly created resource in storage.
     */
    public function store(Request $request)
    {

        Product::create([
            'name' => $request->name,
            'price' => $request->price,
        ]);

        return redirect()->route('products.index');

    }

[ ... more codes ]
}

```

🔑 model ကို import လုပ်ဖို့မမေ့ပါနဲ့

```
use App\Models\Product;
```

🔑 Model ကိုသုံးပြီး data တွေကို အောက်ပါအတိုင်း ဆွဲထုတ်ပါ။

```
$products=Product::all();
```

all function နဲ့ ဆွဲထုတ်ထားတဲ့ data array တွေကို အောက်ပါအတိုင်း frontend view ကို compact function သုံးပြီး ပို့ပါတယ်။ compact function ထဲမှာ \$ ကို မထည့်ရပါဘူး။ ဖြုတ်ထားရပါမယ်။ frontend template ရောက်မှသာ dollar sign ပြန်ထည့်ပြီး \$products ဆိုပြီး ပြန်လည် ဆွဲထုတ်ရမှာဖြစ်ပါတယ်။

Return view("products.index", compact('products'));

Function create ကတော့ create blade file ကိုပဲ ဖွင့်ပေးမှာ ဖြစ်ပါတယ်။ database ထဲက data တွေကို index function ထဲကလို template file ကို သယ်ယူသွားဖို့ data တွေ လောလောဆယ် မရှိပါဘူး။

Function create ရဲ့ လုပ်ဆောင်ချက်က create.blade.php file ကို ဖွင့်ပေးရုံပါပဲ။

Function store ကတော့ create.blade.php file ကနေ request နဲ့ရောက်လာတဲ့ data တွေကို database ထဲကို ထည့်ပေးရမှာ ဖြစ်ပါတယ်။ database ထဲကို data ထည့်ပေးတဲ့ model create function ရှိပါတယ်။ Request data အားလုံးကို တစ်ခါတည်းအောက်ပါအတိုင်းထည့်နိုင်ပါတယ်။

Product::create(\$request->all());

ဒါမှမဟုတ်တစ်ခုပြီးတစ်ခုအောက်ပါအတိုင်း array နဲ့ ထည့်နိုင်ပါတယ်။

Product::create([

'name'=>\$request->name,

'price'=>\$request->price,

]);

Controller configuration ပြီးသွားရင် frontend view template တွေကိုရေးပါမယ်။

////////////////////////////////////

Step-7: create frontend view templates

////////////////////////////////////

=>resources/views

```

|----- products
|
|----- index.blade.php
|----- create.blade.php

```


index.blade.php

```
<!DOCTYPE html>
<html>
  <head>
    <title>CRUD Part1</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH"
crossorigin="anonymous">
      <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
YvpcrYf0tY3IHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcldsIK1eN7N6jleHz"
crossorigin="anonymous"></script>

    <link
      rel="stylesheet"
      type="text/css"
      href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"
    />
  </head>
  <body>
    <div class="container bg-secondary text-white text-center p-5 mt-4">

      <h1>Product CRUD Part 1</h1>

    </div>

    <div class="container my-4">
      <a href="{{ route('products.create') }}" class="btn btn-primary">
        Add New Product
      </a>
    </div>

    <div class="container my-4">
      <table class="table table-striped">
        <thead>
          <tr>
            <td>ID</td>
            <td>Product Name</td>
            <td>Price</td>
```

```

        <td>Action</td>
      </tr>
    </thead>

    <tbody>
      @foreach($products as $product)

        <tr>
          <td>{{ $product->id }}</td>
          <td>{{ $product->name }}</td>
          <td>{{ $product->price }}</td>

          <td>
            <a href="#" class="btn btn-primary"><i class="fa fa-pencil"></i> Edit </a>
            <a href="#" class="btn btn-danger"><i class="fa fa-trash"></i> Delete </a>
          </td>
        </tr>

      @endforeach
    </tbody>
  </table>
</div>
</body>
</html>

```

create.blade.php

```

<!DOCTYPE html>
<html>
  <head>
    <title>CRUD Part 1</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
    rel="stylesheet" integrity="sha384-
    QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH"
    crossorigin="anonymous">
    <script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-

```

```
YvpcrYf0tY3IHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcldslK1eN7N6jleHz"  
crossorigin="anonymous"></script>
```

```
<link  
  rel="stylesheet"  
  type="text/css"  
  href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"  
>  
</head>  
<body>  
  <div class="container bg-secondary text-white text-center p-5 my-4">  
  
    <h1> Product CRUD Part 1</h1>  
  
  </div>  
  <div class="container my-4">  
  
    <a href="{{route('products.index')}}"> Home </a> / Create  
  
  </div>  
  
  <div class="container my-4">  
    <div class="row">  
      <div class="col-lg-6">  
        <form  
          action="{{ route('products.store') }}"  
          method="POST"  
          enctype="multipart/form-data"  
        >  
          @csrf  
  
          <div class="form-group">  
            <label name="name"> Name </label>  
            <input type="text" name="name" class="form-control" placeholder="Product Name" />  
          </div>  
          <div class="form-group">  
            <label name="name"> Price </label>  
            <input type="text" name="price" class="form-control" placeholder="Product Price" />  
          </div>  
  
          <div class="form-group">  
            <label name="name"> </label>
```

```

        <input
            type="submit"
            name="submit"
            class="form-control btn btn-danger"
            value="Save Product"
        />
    </div>
</form>
</div>
<div class="col-lg-6 text-center p-5">
    

</div>
</div>

</div>
</body>
</html>

?>

```

```

////////////////////////////////////

```

Step-8 : Create Routes

```

////////////////////////////////////

```

```

use App\Http\Controllers\ProductController;

Route::resource('products',ProductController::class);

```

```

////////////////////////////////////

```

Step-9 : Check Route list

```
////////////////////////////////////
```

[command]:

PS C:\xampp\htdocs\project1> php artisan route:list

GET HEAD	products	products.index › ProductController@ind...
POST	products	products.store › ProductController@sto...
GET HEAD	products/create	products.create › ProductContro...
GET HEAD	products/{product}	products.show › ProductContr...
PUT PATCH	products/{product}	products.update › ProductCon...
DELETE	products/{product}	products.destroy › ProductCo...
GET HEAD	products/{product}/edit	products.edit › Product...

```
////////////////////////////////////
```

Step-10 : Migrate table

```
////////////////////////////////////
```

[command]:

php artisan make:migrate

Migration success ဖြစ်ပြီဆို server run ကို data entry နဲ့ data display တွေကို testing စလုပ်လို့ရပါပြီ။

```
PS C:\xampp\htdocs\project1> php artisan migrate
```

```
INFO Preparing database
```

```
Open folder in new window (ctrl + click)
```

```
Creating migration table ..... 311.65ms DONE
```

```
INFO Running migrations.
```

```
0001_01_01_000000_create_users_table ..... 1s DONE
```

```
0001_01_01_000001_create_cache_table ..... 306.18ms DONE
```

```
0001_01_01_000002_create_jobs_table ..... 973.29ms DONE
```

```
2024_06_12_082216_create_products_table ..... 139.29ms DONE
```

```
PS C:\xampp\htdocs\project1>
```

Server: 127.0.0.1 » Database: laravel_db

Structure SQL Search Query Export Import Operations Privileges Routines Events Triggers

Filters

Containing the word:

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> cache	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> cache_locks	★ 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"/> jobs	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
<input type="checkbox"/> job_batches	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> migrations	★ Browse Structure Search Insert Empty Drop	4	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"/> products	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> sessions	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
10 tables	Sum	4	InnoDB	utf8mb4_general_ci	240.0 KiB	0 B

⬆ ☐ Check all With selected:

ကျနော်တို့ Laravel project application ကိုအောက်ပါအတိုင်း run ကြည့်ပါ။

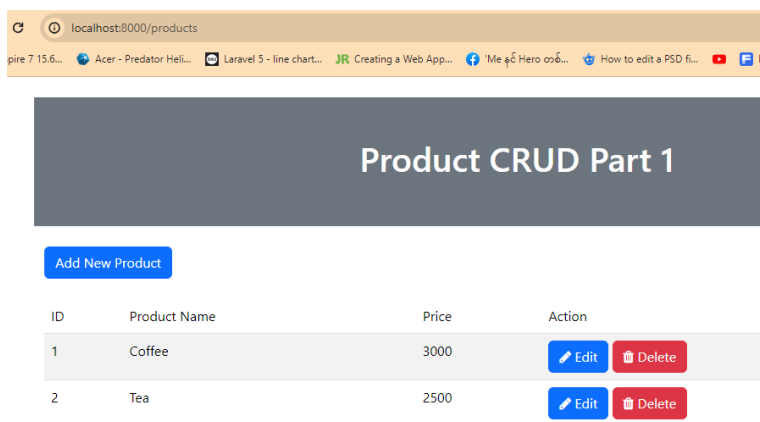
[command]:

php artisan serv

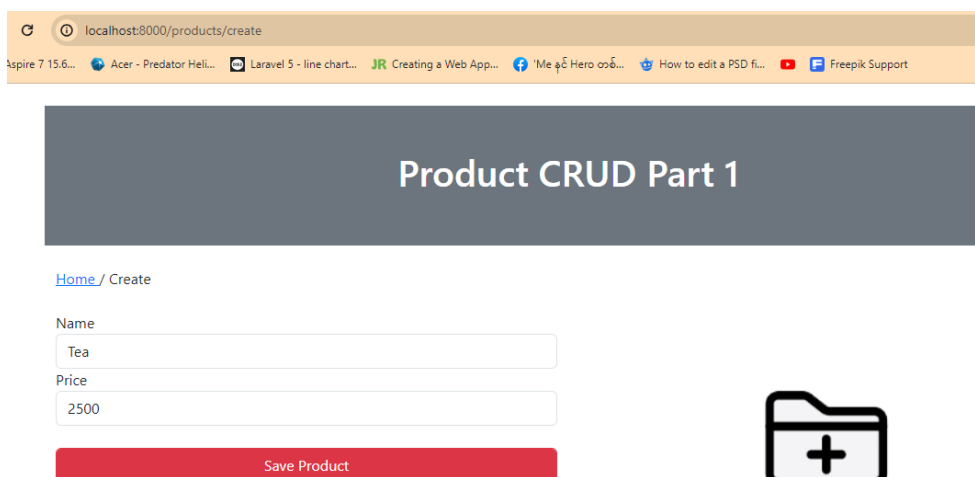
Or

php artisan serve

<http://localhost:8000/products>

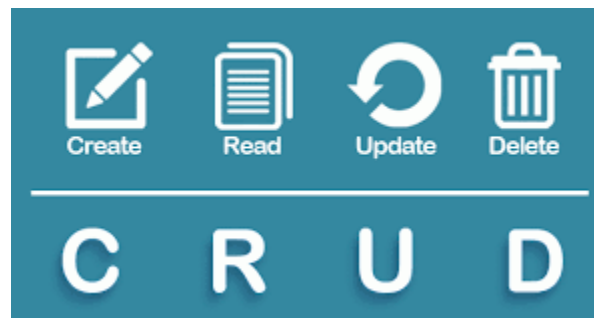


<http://localhost:8000/products/create>



CRUD Part-2

📺 Video Lesson 4



CRUD Part 2 (Edit, Delete)

Update နဲ့ delete operation တွေပြီးတိုင်း route redirection တွေရှိပါမယ်။ ရှိပြီးသား blade file တွေဖြစ်တဲ့ index blade file အောက်ပါအတိုင်း ပြင်ဆင်ရေးသားရပါမယ်။ Data delete operation အတွက် delete method ကို အသုံးပြုရသလို update လုပ်မယ်ဆိုရင် path method ဒါမှ မဟုတ် put method ကို အသုံးပြုရပါမယ်။ အဓိက laravel မှာ form တွေနဲ့ data ပို့တဲ့အခါတိုင်း csrf (cross site resource forgery ကို security token အတွက် အမြဲတမ်းအသုံးပြုရပါတယ်။ @csrf မေ့ခဲ့ရင် data sending အလုပ်လုပ်မှာ မဟုတ်ပါဘူး။

```
////////////////////////////////////
```

Step-1: update codes in Index.blade.php as the following:

```
////////////////////////////////////
```

```
<a href="{{route('products.edit',$product->id)}}" class="btn btn-primary"><i class="fa fa-pencil"></i>
Edit </a>
```

```
<a href="#" class="btn btn-danger" onclick="event.preventDefault();
document.getElementById('delete-form-{{ $product->id }}').submit();" >
```

```
<i class="fa fa-trash"></i> Delete
```

```
</a>
```

```
<form id="delete-form-{{ $product->id }}" action="{{ route('products.destroy',$product->id) }}"
method="POST" class="d-none">
```

```
@csrf
```

```
@method('DELETE')
```

```
</form>
```

```
////////////////////////////////////
```

Step-2: add the following codes into the ProductController

```
////////////////////////////////////
```

```
public function edit($id)
{
    $product=Product::findOrFail($id);
    return view('products.edit',compact('product'));
}
```

```
/**
```

```
* Update the specified resource in storage.
```

```
*/
```

```
public function update(Request $request, $id)
```

```

{
    $product=Product::findOrFail($id);
    $product->name=$request->name;
    $product->price=$request->price;
    $product->save();

    return redirect()->route('products.index');
}

public function destroy($id)
{
    $product=Product::findOrFail($id);
    $product->delete();

    return redirect()->route('products.index');
}

```

////////////////////////////////////

Step-3: Create new view edit.blade.php file

////////////////////////////////////

```

<!DOCTYPE html>
<html>
    <head>
        <title>CRUD Part 2</title>

        <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH"
crossorigin="anonymous">
        <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
YvpcrYf0tY3IHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcldslK1eN7N6jleHz"
crossorigin="anonymous"></script>

        <link
            rel="stylesheet"
            type="text/css"
            href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"
        />
    </head>
    <body>

```

```
<div class="container bg-secondary text-white text-center p-5 my-4">

    <h1> Product CRUD Part 1</h1>

</div>
<div class="container my-4">

    <a href="{{route('products.index')}}"> Home </a> / Create

</div>

<div class="container my-4">
    <div class="row">
        <div class="col-lg-6">
            <form
                action="{{ route('products.store') }}"
                method="POST"

            >

                @csrf

                <div class="form-group">
                    <label name="name"> Name </label>
                    <input type="text" name="name" class="form-control" placeholder="Product Name" />
                </div>
                <div class="form-group">
                    <label name="name"> Price </label>
                    <input type="text" name="price" class="form-control" placeholder="Product Price" />
                </div>

                <div class="form-group">
                    <label name="name"> </label>
                    <input
                        type="submit"
                        name="submit"
                        class="form-control btn btn-danger"
                        value="Save Product"
                    />
                </div>
            </form>
        </div>
        <div class="col-lg-6 text-center p-5">
            

        </div>
    </div>
</div>
```

```
</div>
</body>
</html>
```

```
?>
```

```
////////////////////////////////////
Step-4: Run the project
////////////////////////////////////
```

[command]:
Php artisan serv

<http://localhost:8000/products>



Product CRUD Part 2

Add New Product

ID	Product Name	Price	Action
1	Coffee Update	3000	Edit Delete
2	Tea	2500	Edit Delete



Product CRUD Part 2

Add New Product

ID	Product Name	Price	Action
1	Coffee	3000	Edit Delete
2	Tea	2000	Edit Delete



Product CRUD Part 2

[Home](#) / Edit

Name

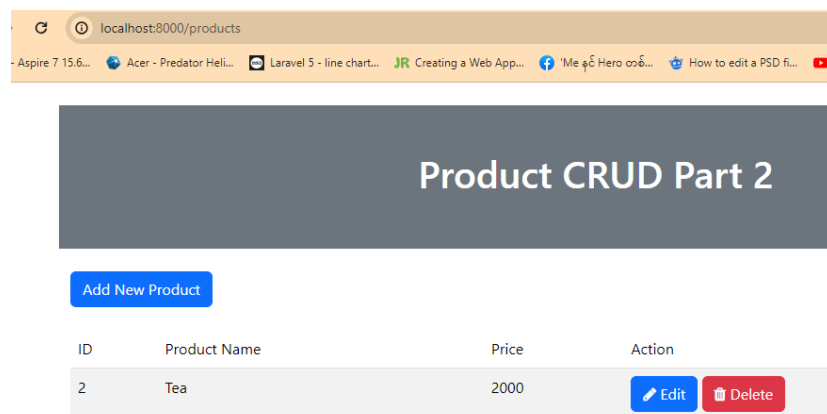
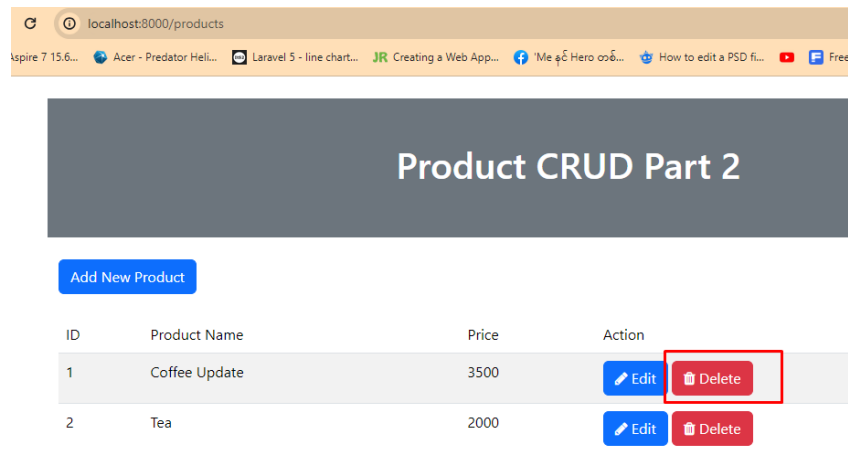
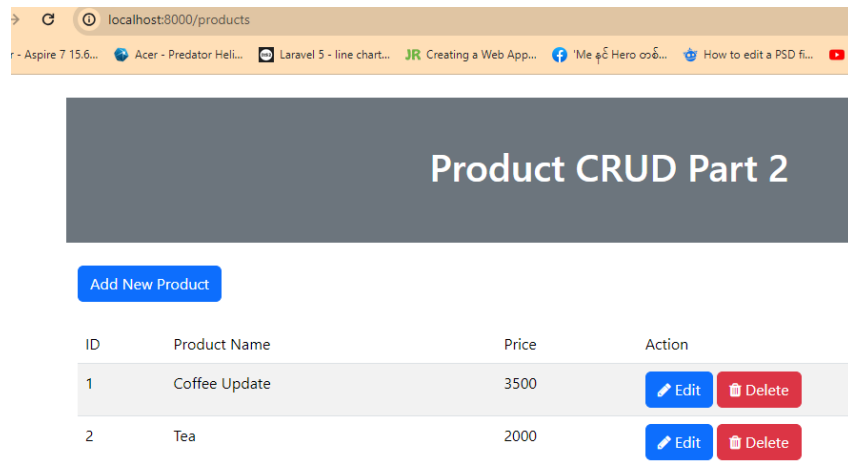
Coffee Update

Price

3500

Update Product

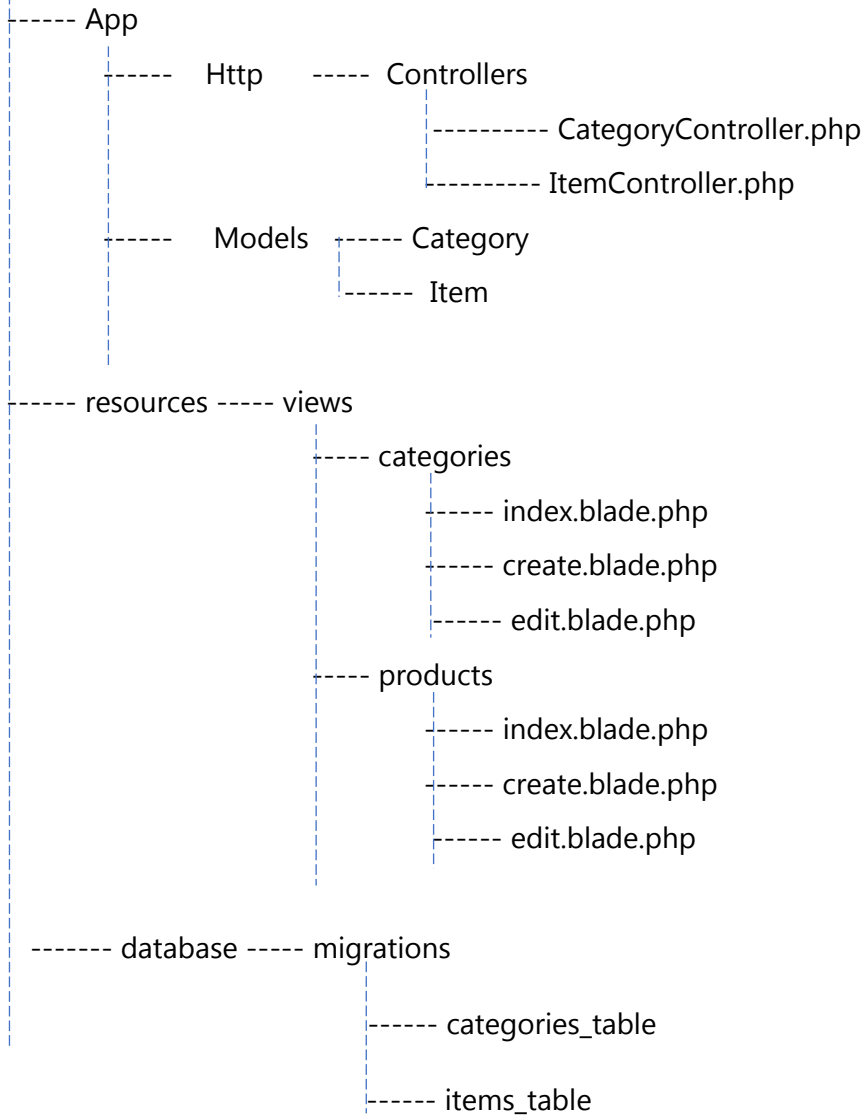




Assignment-1

CRUD Operations အပြည့်အစုံရေးရန်-

Assignment-1



Hints:

Database: assignment_db1

Table 1: categories_table

- id
- name

Table 2: items_table

- Id
- category
- name
- price

Model 1: Category

```
protected $fillable = [  
    'name',  
];
```

Model 2:Item

```
protected $fillable = [  
    'category',  
    'name',  
    'price',  
];  
  
];
```