Hellow!Muddsser Hasnain!Thanks For Get Admission In Webeducatorz.org

Full Stack Web Development

06 Months Duration

3 Classes In A Week

LMS Link: https://lms.webeducatorz.org

Your Portal Details:

Email:Malik.eng5@gmail.com

password: 12344321

Regards

Accounts Webeducatorz

Accounts Department: 03184544494

Admission Department: 03264300993

thank you for your coperate.

Full stack Development

Full Stack JavaScript

Laravel with React

...........................

Road map

HTML .... CSS.......JavaScript.......Bootstrap/Tailwind

PHP.......MySQL ........Laravel.....Codeignter.....Mern Stack

Mean Stack........ React......... Angular...... VueJS.... Blog

AdSense.......SEO....SEM.....Social Media Marketing Campaign.

Full Stack Web Development

Section 1: Basic Web Developer

HTML

CSS

Boot Starp

Basic JS

PHP

MySQL

Section 2: Advance Development

Laravel.... Advantages of FrameWorks = Job Title (Laravel Developer)

Section 3: MERN Stack

React.... MongoDB.... NodeJS.... ExpressJS

Suggestion

Angular.....Vue....

**1 : Sir Abdullah Waseem : GitHub link**

https://github.com/websaazwebsolution/FSWD-Batch-31-LWR-Batch-18.git

<https://github.com/TheWebUstaad/FSWD-Batch-31-LWR-Batch-18.git>

2: Dash board creating link.

<https://chatgpt.com/share/681e0590-6368-8003-8c82-3a13983474d7>

**2 : Sir Asad Mukhtar : GitHub link**

<https://github.com/asadmukhtarr/evssystem>

Laravel

Lecture # 20

So far we have created a dashboard and a blog project using php. Now we will created blog project using Laravel frame work.

In Laravel we will create a website having index , courses, services, and about page , and with that web site will use multi user , multi role dashboard system means each user will have its different dashboard according to its role ...... and single user may have multi roles in a company . and For a single role there may be multi users ......

In our case we will have two users .... Admin and User.

Laravel Day One :

- Open Source MVC Fram Work

- What is frame work

- Model

- Views

- Controller

- .php x .blade.php

**A PHP framework with a robust ecosystem**

Laravel is a web application framework with expressive, elegant syntax.

A web framework provides a structure and starting point for creating your application, allowing you to focus on creating something amazing while we sweat the details.

Laravel strives to provide an amazing developer experience while providing powerful features such as thorough dependency injection, an expressive database abstraction layer, queues and scheduled jobs, unit and integration testing, and more. <https://laravel.com/> ,

Laravel has many open-source packages, starter kits and products we can use to import and ease of use environment to create wonderful application more efficiently.

[**Why Laravel?**](https://laravel.com/docs/12.x#why-laravel)

There are a variety of tools and frameworks available to you when building a web application. However, we believe Laravel is the best choice for building modern, full-stack web applications.

**A Progressive Framework**

We like to call Laravel a "progressive" framework. By that, we mean that Laravel grows with you. If you're just taking your first steps into web development, Laravel's vast library of documentation, guides, and [video tutorials](https://laracasts.com/) will help you learn the ropes without becoming overwhelmed.

If you're a senior developer, Laravel gives you robust tools for [dependency injection](https://laravel.com/docs/12.x/container), [unit testing](https://laravel.com/docs/12.x/testing), [queues](https://laravel.com/docs/12.x/queues), [real-time events](https://laravel.com/docs/12.x/broadcasting), and more. Laravel is fine-tuned for building professional web applications and ready to handle enterprise work loads.

**A Scalable Framework**

Laravel is incredibly scalable. Thanks to the scaling-friendly nature of PHP and Laravel's built-in support for fast, distributed cache systems like Redis, horizontal scaling with Laravel is a breeze. In fact, Laravel applications have been easily scaled to handle hundreds of millions of requests per month.

Need extreme scaling? Platforms like [Laravel Cloud](https://cloud.laravel.com/) allow you to run your Laravel application at nearly limitless scale.

**A Community Framework**

Laravel combines the best packages in the PHP ecosystem to offer the most robust and developer friendly framework available. In addition, thousands of talented developers from around the world have [contributed to the framework](https://github.com/laravel/framework). Who knows, maybe you'll even become a Laravel contributor.

[Creating a Laravel Application](https://laravel.com/docs/12.x#creating-a-laravel-project)

[Installing PHP and the Laravel Installer](https://laravel.com/docs/12.x#installing-php)

Before creating your first Laravel application, make sure that your local machine has [PHP](https://php.net/), [Composer](https://getcomposer.org/), and [the Laravel installer](https://github.com/laravel/installer) installed. In addition, you should install either [Node and NPM](https://nodejs.org/) or [Bun](https://bun.sh/) so that you can compile your application's frontend assets.

If you don't have PHP and Composer installed on your local machine, the following commands will install PHP, Composer, and the Laravel installer on macOS, Windows, or Linux:

Laravel is an open source MVC (Model view controller) base framework software, which is helpful In running PHP organized files.

Laravel is an open-source framework,

What is a framework: Is a software that help a programmer to write and modify syntax of any language either it is front end language so that code can be more robust, secure, fast and efficient. It helps to write organized code and help to code in team work. For every task to perform a specific folder is created.

**Before lecture 20 we created a blogproject using php, we created login pages, action page, created tables in mysql using xxamp. Creat .php files to view. This was a project with out a laravel framework , and we use database connections and queries in same files.**

Now in **Lecture 20** and onward . we will create a blogproject using Laravel framework.

Laravel is a MVC php Framework . Laravel MVC (Model view controller) controls view by creating routs for each view/ i.e page.

Model : Database logics

View : Front end

Controller: Logics

To install Laravel frame work into our system , first we need a php-composer to download from google. Composer is a software that run our php-commands, in command prompts (cmd) .

Google -> php composer -> Download composer from -> <https://getcomposer.org/>. - > after downloading go to downloads and click to install composer -> An installation wizard appear , from installation wizard make sure to select -> install for all user option -> after from next window -> select developer mode -> from next window in installation wizard make sure php.exe file is selected in the composer installation path. If php.exe is not showing in path, go to xamp folder, + php folder + select php.exe file -> click next.

After installation is completed, go to cmd from start menu, in cmd type command

Composer -v +Enter (To check the version of composer).

Now after installing php composer, we will install Laravel with the help of this composer,

We take command from Laravel.com “composer global require Laravel/installer”.

Go to start menu, and right click command prompt **run as administrator** to command prompt.

In command prompt type command “composer global require Laravel/installer”.

After Laravel is installed, we can check in command prompt, to check if Laravel installed or not by command,

Now we will create Laravel folder inside our xamp folder, go to xamp folder + htdocs folder here create a folder named Laravel. we can create Laravel in any folder but inside xamp folder where myphp admin / MySQL is installed, it will help us to configure database easily.

Now go into Laravel folder and into that folder, right click and open Terminal here, in that terminal, run command

Laravel new Tech-app + Enter.

While creating project compiler will ask few questions …

1. - Which starter kit would you like to install ,

Select none.

1. - Which testing framework do you prefer ?

[0] Pest

[1] PHPunit

Select 0 , Pest

Now after this command new project with your name Tech-project will be created.

Laravel Day one

Open new Tech-project in VS-Code ,

MVC Controls in following ways.

Model View Controller // login.php action/login.php

Model (Database logics)

* App -> models - > table names

View (Front end work)

* Resources -> View

Controller(logics)

* App -> http -> Controller - > files

Database

* Migration blue print

Localhost / folder\_name

Laravel Server

* **Php artisan serve** // Run this command in vs-code terminal. To start Laravel server.

**Database connectivity To Laravel Project Migrate command**

To create database through Laravel into mysql/ xamp , we write following command into terminal.

* **Php artisan migrate**
* // **This above command will migrate all the new Tech\_project database and tables into MySQL.**

**After running artisan migrate command ……**

**We can use Tabe Plus application to add different databases at a single app and view tables of any database through this Table Plus app.**

**Or we can see our new database and table of our Tech\_project created in Laravel server are migrated to our MySQL database, with a database name Tech\_project.**

**File Extensions:**

* In Laravel for users view files , i.e in view section where we make files using html, css, bootstrap an php ,we don’t use extension .php. we use extension i.e. **file name.blade.php.**
* i.e index.blade.php , about.blade.php , contact\_us . blade.php

1. Got To: Vs\_Code – Tech\_project – resources – view
2. Create a new folder in view named “layouts” in layouts create a new file named : app.blade.php.
3. Open html template index.html file in vs\_code. now copy its all code, and past into “Tech\_project/resources/views/layouts/app.blade.php “.
4. Now from this “app.blade.php” file working of our Tech\_project actually starts.
5. Close index.html we don’t need it any more.
6. Now from app.blade.php
7. We know that header, footer and navigation bar (Top or side) are mostly same on all pages.
8. From app.blade.php file we copy only hero section, i.e leaving / excluding header, footer and navbar, we will **Cut** all inside code and go to Tech\_project/resources/view/welcome.blade.php file.
9. Select all the by default code created while creating Tech\_project through Laravel, select all and delete all by default code and past that code which we cut from app.blade.php.
10. The place where we cut the hero code, we will create a section using @yield command. i.e use code @ yield “contents ” ……… This @yield actually will create a section with a name content. We will call this section in any other page.
11. Now at the top of the file welcome.blade.php , where we pasted our code, we write code.

@extends(‘layouts.app’)

@section(‘contents’)

Pasted code from app.blade.php

@endsection

1. Now in app.blade.php file will only have header, footer and nav bar.

We will check our welcome and app.blade.php file by calling through our Laravel server.

For this go to terminal, run php artsan server command. Laravel server will start and show message i.e INFO Server running on [http://127.0.0.1:8000].

We press Cntrl key and double click <http://127.0.0.1:8000>. And our server home route will show us welcome.blade.php.

Laravel

Lecture # 21

Laravel Day 2:

- php aritsan serve

- Routes

(index.php about.php)

about.blade.php url link

- With functional Routes

- with Controllers

- route('name')

-How Tyo set Image in Laravle

<img src={{ asset('images/logo.pnh') }}

- migartion : Blueprint

How we create Any table in laravel

php artisan make:migration create\_courses\_table

php artisan make:migartion create\_services\_table

-Prefix /login /register /dahsboard mywbeist.com/admin/login

Day one we studied what is Laravel frame work , it is a MVC (Model View Controller) frame work , it work through MVC scheme.

**Yesterday we worked on welcome.blade.php and app.blade.php** ……..

**Today** we will work on about.blade.php ………..

1. Open vs.code ….. Got to tech\_project……. Go to resources ….. go to views and create a file about.blade.php.
2. Go to the website template we downloaded yesterday …….. open about.html into vs.code. and here we will remove header and footer , because we will import/extends header and footer from app.blade.php. we will keep only hero section / main section. Copy all the code of main / hero section.
3. Open file about.blade.php and at the top of the file write

@extends(‘layouts.app’) // This extends actually app.blade.php file

where we have only header and footer code.

@section(‘contents’) // This line will import section named content, that we created in app.blade.php.

1. Now we paste the about hero / main section code here after @section (‘content’), and after pasting code we will write @endsection.
2. Now our about.blade.php page is ready .

We can title a page with concatenate of website title……….

<title> @yield('title') || WebSaaz Solution - Professional Software Development</title>

Now we can go to any page of our website and call this yield in the title of that page…….. we write following two lines………..

@extends('layouts.app')

@section ('title', 'About us')

**How to set image in Laravel**

In Laravel we add image in following way …. Because when we are using blade template we have to use asset-tag in image source src- attribute. {{ assets ( ‘ image/ image name ‘ )}}

In the image tag we have src (source) attribute, we add following code in src-......

**In our welcome.blade.php class we will add. Herp image ............at line number 37........**

 <img style="width: 400px !important;" src="{{ asset('images/hero.png') }}" alt="WebSaaz Solution - Digital Transformation" class="ml-auto  w-full h-auto rounded-lg shadow-2xl max-w-md">

**Similarly, we use same asset. Tag for all images in laravel ......**

**How to make tables in database through Laravel project:**

As we see in Lecture 20 , that to create database of our Laravel project into MYSQL ......

We use

**Php artisan migrate ..........command to migrate new database of Laravel server into MySql** ............

**We will see in MYSQL that new database named “Tech\_Project” is created into MYSQL.**

**Now we will create new tables into laravel project** and **migrate them to our MYSQL database named “Tech\_Project”**

We use migration command to create tables into our database .... MYSQL ........

**Command for Terminal:**

Php artisan make :migration create\_course table

Php artisan make : migration create services\_table

We will see two tables added in migration tab in Tech\_project in vs.code.

Go to vs.code --- Tech\_project ------- migration ......... go to table\_courses file.........

Open table\_courses file ........ and Look for Schema : : create ( ‘courses ’ , function(Blueprint $table) : void) {

$table-> id ();

$table->string(‘name’);

Similarly add column name with datatype...........

}

Now we will create an Admin panel .......... For this we will download a template from google or create from Chat Gpt.

We downloaded a template ( D:\Full Stack Web Developer\FSWD Lecture 2 Admin Layout Templete )

1. Go to vs.code ....... Tehc\_project-------resources-------views
2. Create a new folder “ admin” ............. inside admin folder ........... create a folder “layouts” ............. Inside folder layouts create a file .......... “ layouts.blade.php” ...........
3. Now go to ........ D:\Full Stack Web Developer\FSWD Lecture 2 Admin Layout Templete\AdminLTE-4.0.0-rc1\AdminLTE-4.0.0-rc1\dist\
4. Copy three folders i.e assets , css, and js ......... and paste them into D:\Full Stack Web Developer\New folder\htdocs\laravel\tech-app\public\adminassets
5. Tech\_project \ public\adminassets ............
6. Now will will have three folders in adminassets folders i.e assets , css, js..........

NOW Crete admin dashboard ...........

1. Now create a new file at go to vs.code ...... resources.............view ........... admin ...... in admin folder create a new file named ......... admindashboard.blade.php.
2. Now open index.html file from following in vs.code .....
3. D:\Full Stack Web Developer\FSWD Lecture 2 Admin Layout Templete\AdminLTE-4.0.0-rc1\AdminLTE-4.0.0-rc1\dist\pages\index.html.
4. From index.html keep only header, footer and Nav bar ........ and cut all the main/hero section and paste it into dashboard.blade.php
5. Before pasting into dashboard.blade.php write on front
6. @extents (‘layouts.app’)
7. @section(‘content’)
8. Paste code from index.html
9. @endsection
10. Now we have created a dashboard page for admin......... of our website (welcome.blade.php).

**Creating Rout for Admin dashboard**

1. Go to Tech\_project ------ routes ............web.php
2. In web.php, we will see routes of welcome (home) page, about.php, ....... similarly, we will create route for dashboard.blade.php.
3. We can create route by two methods ........ By command Route :: ........ or By Controller ......
4. Create all the admin routes through prefix- group command , so that all the routes of admin i.e login, register will be created and manage by admin. i.e
5. Dashboard mywebsite.com/admin/login ........

Now to include our css-cdn file into dashboard for styles ......... our dashboard style file (i.e header , footer and Navbar are in tech\_proect/resources/view/admin/layouts/ layout.blade.php)

We will href – css file into tech\_project/resources/view/admin/layouts/layout.blade.php ---

For this we open the file and add code as at line 45 .... as fillowing.

<link rel="stylesheet" href="{{ asset('adminassets/css/adminlte.css') }}" />

Now we can check our Laravel server ( <http://127.0.0.1:8000> ), when we open laravel server it will show us page of our laravel application website ........

Laravel

Lecture # 22

Laravel Multi Role Admin panel

- How to update and insert column in migration

php artisan make:migration add\_role\_id\_to\_users\_table --table=users

- php artisan make:model Role -m

Day 3 Laravel

Laravel Login System

-1st Step

- Table - Users user -> role\_id

- Table - Role role\_id

- Multipule Dashboard

- User / Admin

- migration -> blue Print (ok) (users/role)

-2nd Step

- Seeder Dummy Content (admin/users with password)

-3rd Step

-MiddleWare (Aunty -> Camera Man)

-4th Step

-Model

-5th step

-login page View

-6th step

-Controller (logics)

-7th step

-routes with (controller + Middleware )

So far as we have created our website through laravel and created its about page, and created layouts in different files ........... we also created web routs of welcome page and about pages.

Then we created an admin panel dashboard and its layouts and also created its web routes.

Theory : What ever we create in database of Tech\_project of Laravel server, we must migrate.

What ever table we made in database , we create its model, and controller, seeder and middleware.

Today we will see.

Laravel Multi Role Admin panel in which we create users in user table, and role table and role id so that multiple users can have their different dashboard with their role id .......

* First of all we have created table Users in database tech\_project. And migrated it to mqsl.
* Today we will see to update and insert column role\_id , in user table and migrate it to mysql.
* We use following command.....

**Command for Terminal Laravel server**

**STEP 1:** Create table – Users ... with primary key role\_id

Create table – Role ....with primary key ....role\_id (when wer create table

We must have to create its model in Laravel, we can do it in separate command or in same

Command ).

Migrate blue print of multi user / multi dashboard Tech\_app

Database to Mysql database.

**1 Create Role Id in Users Table:**

1. *Php artisan make-migration add\_role\_id\_to\_users\_table<space>--table = users*
2. This above command will create a file in tech\_project ...... migration

........2025\_05\_23\_131931\_add\_role\_id\_to\_users\_table.php,

1. Open this file and in the public function up() {
2. public function up(): void
3. {
4. Schema::table('users', function (Blueprint $table) {
6. });
7. Add code

$table->bigInteger('role\_id')->after('id');

This code will add a column ‘role\_id’ into the table ‘users ‘, after column named ‘id’

  public function up(): void {

        Schema::table('users', function (Blueprint $table) {

$table->bigInteger('role\_id')->after('id'); // after (‘id’) means to create

// role\_id column after first column ‘id’

    });

}

1. Now go to terminal (By pressing Cntrl+ j) and Run following command -----

***php artisan migrate. +Enter***

1. This will update the Laravel Tech\_Project users table with a new column role\_id in MYSQL database.

**Now we will create role table and role Model with a single command:**

For every database entry table blueprint, we create through Laravel server database in Tech project, we always create its model as well. Then we migrate it to MySQL through migrate command.

**Command**

We can use short command to make and migrate together when making model.

***Php artisan make: model Role -m*** +Enter

This command will create two files

1. First fill will be created in Tech\_project........ **database...........migration.......date\_Any\_number\_create\_role\_table.php**
2. Second file will be created in Tech project ------ **models-----Role.php**
3. The second file is a blue print of the database table.
4. This command will automatically do migration from Laravel server to MySQL. Because of -m . We don’t need to write migration command properly.
5. Open First file i.e.

**database...........migration.......date\_Any\_number\_create\_role\_table.php**

**add code at**

 public function up(): void

    {

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

            $table->bigincrements('role\_id'); // ad role\_id for each user

            $table->string('role\_name');

            $table->string('role\_slug'); // slug is url..... localhost/admin.xyz

            $table->timestamps();

        });

    }

1. Open Second file **Role.php** and **add following code**........

class Role extends Model

{

    use HasFactory; // right click on hasfactory and select import namespace,

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

  public function users(): HasMany {

        return $this->hasmany("App\User"); // This means that one role may have many // users .

    }

}

1. **Now , Go.vs.code tech\_project --- Model – Users.php ...... this file is model of user table.....**

We have model for every table created in database---migration.

Add following code ......... before last line closed braces.

public function roles(): BelongsTo {

            return $this->belongsTo("App/Role");

    } // This means that each use must belongs to a certain role and in role.php we said in hasmany function that a user may have many role.

**STEP 2: Create Seeder For Users Table and Role Table:**

Now we will create seeder for both table i.e role and user ..........

Open vs.code ......Terminal ......... Command

***php artisan make:seed UsersTableSeeder +Enter***

***php artisan make:seed RoleTableSeeder +Enter***

*Now we have two files created in tech\_project......databases......seeders.*

1. *Open file RoleTableSeeder.php*

Add code ....... in

 public function run(): void

    {

         DB::table  ("roles")->insert([

                "role\_name" =>   "Admin",

                "role\_slug" =>   "admin",

         ]);

         DB::table  ("roles")->insert([

                "role\_name" =>   "User",

                "role\_slug" =>   "user",

         ]);

    }

1. Open file UserTableSeeder.php

Add code ..........

 public function run(): void

    {

        DB::table("users")->insert([

            "role\_id"   => 1,

            "name"      =>  "Admin",

            "email"     =>  "admin@gail.com",

            "password"  =>  bcrypt("pass@admin "),

        ]);

         DB::table("users")->insert([

            "role\_id"   => 1,

            "name"      =>  "User",

            "email"     =>  "user@gail.com",

            "password"  =>  bcrypt("pass@user"),

        ]);

    }

1. Open file DatabaseSeeder.php

Add code .......

 public function run(): void

    {

        // User::factory(10)->create();

       $this->call(UsersTableSeeder::class);

        $this->call(RoleTableSeeder::class);

    }

To shift all the record entries we done in user seeder , role seeder and database seeder from Tech\_app to database we run following commands in Laravel terminal.

Php artisan db:seeder

// This command will create new dummy records into our database users table.

**STEP 3:**  **Create Middlewares**

1. **Create Middleware for Users, Admin dashboard and Redirect Authenticated**

Command to make middleware

1. Php artisan make:middleware AdminMiddleware

Open this file from Tech\_project...... app......http.........Middleware

Add code.

*public function handle(Request $request, Closure $next): Response*

*{*

*if(auth::check() && Auth::user()->role\_id == 1){*

*return $next($request);*

*} else{*

*return redirect()->route("login");*

*}*

*}*

1. Php artisan make:middleware UserMiddleware

Open this file from Tech\_project...... app......http.........Middleware

Add code.

*public function handle(Request $request, Closure $next): Response*

*{*

*if (auth::check() && Auth::user()->role\_id == 2) {*

*return $next($request);*

*} else {*

*return redirect()->route("login");*

*}*

*}*

=================

1. Php artisan make:middleware RoleMiddleware

Open this file from Tech\_project...... app......http.........Middleware

Add code.

 public function handle(Request $request, Closure $next, ...$roles): Response

    {

        if(!Auth::check()){

            return redirect()->route("login");

        }

        $user = Auth::user();

        if(in\_array($user->role->role\_slug,$roles)){

            return $next($request);

        }

    }

Laravel

Lecture # 23

-5th step

-login page View

-6th step

-Controller (logics)

**We will create a login panel for admin** ......

1. Go to vs.code tech\_project ........ resources ....... views.........admin..........
2. In admin folder ...... create a file login.blade.php..

**In Step 6 we will create controller logics ......**

Command for creating controller

Php artisan make:controller AuthController // This controller decide which dashboard to open for

// which user.

Open AuthController file from Vs.code –Tech project........ app......http......controller .....AuthController.php

Also Open

Open AdminController file from Vs.code –Tech project........ app......http......controller ............. Admin........

Controller.php

Copy public login function from admin controller and paste in Auth controller

 public function login(Request $request)

    {

        // dd($request->all());

        // return "login";

        $credebtials = $request->validate([

            'email' => 'required|email',

            'password' => 'required',

        ]);

// After pasting change following if condition

        if (Auth::attempt($credebtials)) {

            $user = Auth::user();

            if ($user->role\_id === 1) {

                return redirect()->route("admin.dashboard");

            } elseif ($user->role\_id === 2) {

                return redirect()->route("user.Dashboard");

            }

        }

        return back()->withErrors([

            'loginEmail' => 'The provided credentials do not match our records.',

        ])->onlyInput('loginEmail');

    }

After Creating Auth Controller , we will set route for the auth controller, Go to routes -> web.php and set route for login page through the Authcontroller,

For this add following code into routes / web.php

Route::get('/login',[AuthController::class, 'ShowloginForm'])->name('login');

Route::post('/login', [AuthController::class, 'login'])->name('do.login');

Route::middleware(['auth', 'role:admin'])->group(function () {

    Route::get('admin/dashboard', [AdminController::class, 'dashboard'])->name('admin.dashboard');

}

);

Also set this route into Views / login.blade.php , at line 63.

  <form action="{{route('login')}} “method="post">

Go to Vs-code Terminal and write command

Php artisan serve // Enter

And this above command starts Laravel server , and show us the url of the server i.e

INFO Server running on [http://127.0.0.1:8000]

We can control+click on http server name and this server web view will be opend in the browser i.e chrome and it will show our projects main page (website) .........

Now in the header section we will write / login and press Enter.

<http://127.0.0.1:8000/login>

This will open the .......Views / login.blade.php ( main login page of our website) .......

Now create Admin controller by following command in Vs-code Terminal.

Php artisan make:controller AdminController

Open UserController file from Vs.code –Tech project........ app......http......controller .....AdminController.php

Add following code ...

class AdminController extends Controller

{

    public function dashboard() {

        return view('admin.dashboard');

    }

}

Now set route to the admin controller, got to View/routes ...... and add following code ......to set route for admin controller.

Route::middleware(['auth', 'role:admin'])->group(function () {

Route::get('admin/dashboard', [AdminController::class, 'dashboard'])->name('admin.dashboard');

}

);

Now create User controller by following command in Vs-code Terminal.

Php artisan make:controller UserController

Open UserController file from Vs.code –Tech project........ app......http......controller .....UserController.php

Add following code .......

class UserController extends Controller

{

    public function dashboard(): View{

      return view('users.dashboard');

    }

}

Now Set route for User Controller ......... For this open web.php file , go to Vs.code ........ Tech\_project.........routes / web.php

Add following code in web.php to set user route........

Route::middleware(['auth', 'role:user'])->group(function () {

Route::get('admin/dashboard', [AdminController::class, 'dashboard'])->name('admin.dashboard');

}

);

Laravel

Lecture # 24

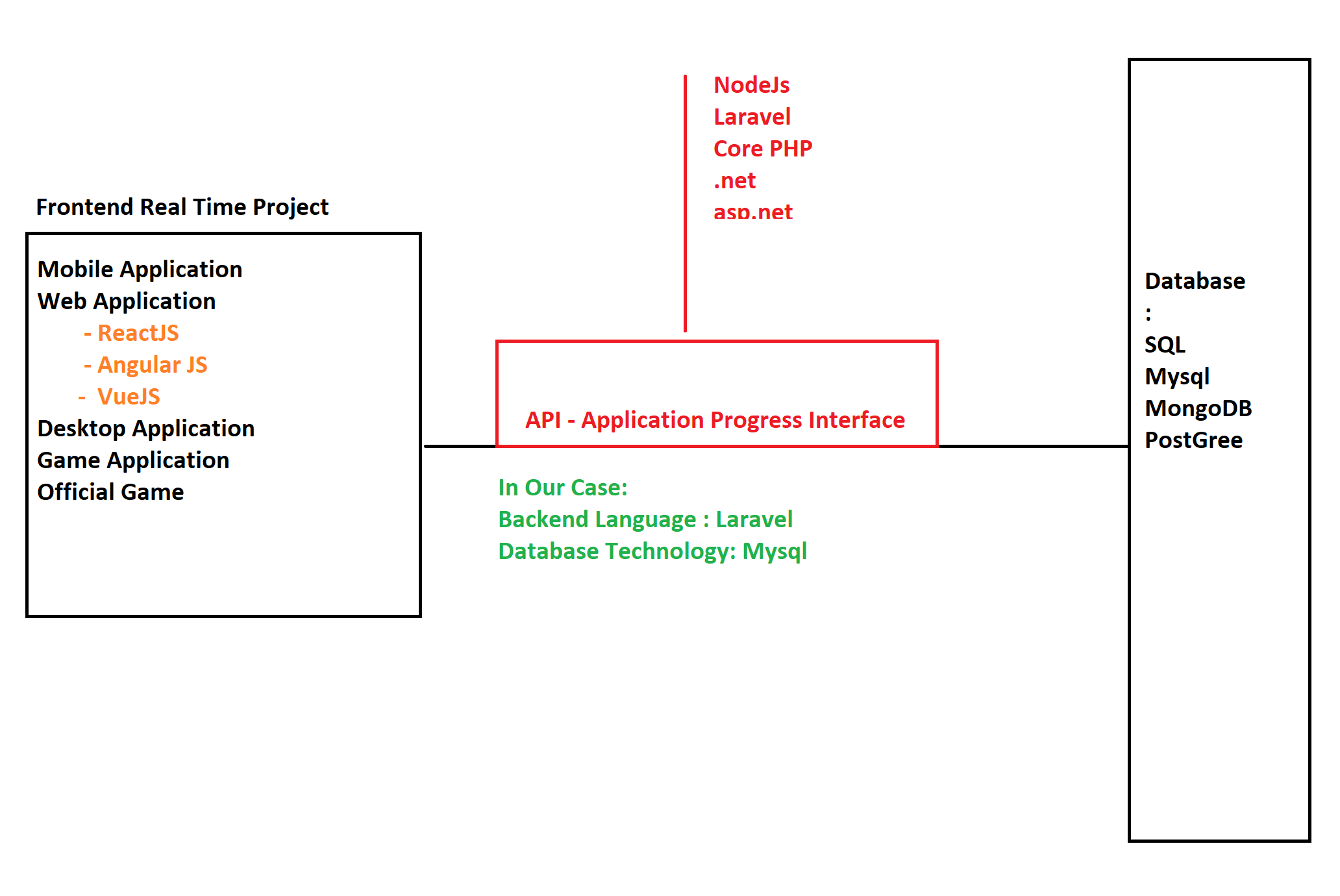
Laravel

Lecture # 29

This is the last lecture on Laravel ......... Today we will create an API (Application Programming Interface).

And our Api.php file will be saved in our Laravel project in Routs.

Following diagram show how Api works ...



These Api will be connect \ configure with any framework of Js .........Angular-js, React-js, Vue-js , Express-js.

**2 : Sir Asad Mukhtar : GitHub link**

<https://github.com/asadmukhtarr/evssystem>

How to import a project from git hub into our Laravel vs.code project.......

1. Download project form git hub and unzip it.
2. Copy unzip project file into the wamp server / htdocs/Laravel / New-Project.
3. After copying project into Laravel folder. Now open command prompt and prompt to the project folder.

c:\Users\digit> cd .. Enter

c:\> d: .......Enter

d:\> dir

d:\> cd full stack web developer

d:\Full Stack Web Developer> dir ..Enter

d:\Full Stack Web Developer>cd new folder ...Enter

d:\full stack web developer\new folder> cd htdocs... Enter

d:\full stack web developer\new folder\ htdocs> cd Laravel...Enter

d:\full stack web developer\new folder\htdocs\Laravel>cd project\_name.. Enter

d:\full stack web developer\new folder\htdocs\Laravel\project\_name> composer install .....Enter

1. Now go to vs.code open project that you want to import .
2. Open .env file , if .env file not available....... change name of .env.example file to .env and use it as .env file.

In .env file check

*DB\_CONNECTION=mysql // Database type must be mysql and Not sqllite or other.*

*DB\_HOST=127.0.0.1 // Host Laravel server ip.*

*DB\_PORT=3306*

*DB\_DATABASE=evssystem // This is database name , we must create a new daabase in MYSQL with this same name.*

*DB\_USERNAME=root // User should be root , can be other but root user is easy to process.*

*DB\_PASSWORD= // Password is empty.*

1. Cope name of database a create a new database in MYSQL , with this same name as in .env file.
2. Now got to command prompt again, and type command,

*Php artisan migrate.*

1. This command will migrate all the Laravel new project database file/tables into our actual MYSQL database we just created in above step.
2. We can check by going into MYSQL database and check if tables are created into our new database or not, these tables will definitely / must be created.
3. Now if we go to vs.code .env file of new project and look for APP\_KEY= , we have to create that key in .env file by following command in command prompt.

*Php artisan key:generate*

1. *We can check into .env file that APP\_KEY = (some vale) ........ has been generated.*
2. Now we run another command into command prompt to link our public folder with our storage folder vs.code/ app project.

*Php artisan storage:link*

1. Now we will run our command to start Laravel server.

Php artisan server

1. Now our project will be start working.

FSWD

Lecture # 30

Full Stack Development

**MERN STACK**

Mongo DB ............. Mongo DB ........... Database

EXPRESS JS .................. Baby Frame work

React JS ............... Library

Node JSS ................ Frame work.

Combining these Technically These are not four things ............. These are two things.

1 = React Js ........... Will be used independently

2 = Node Js + Express JS + MongoDB...... These three things will be used in React JS.

For MERN Sack ......... We start Learning JavaScript ....

Java Script:

**What is Java Script:**

JavaScript is a programming language used to make website interactive and dynamic. It runs directly in the browser allowing you to create things like.

JavaScript is based on actions......... i.e.

Mouse button click.

Mouse Over,

Key Typing

Sliders and animations

Form validations,

Dynamic content (e.g. live search, charts)

Real-Time chats, games, etc.

java scrip is written in side Php code, just like CSS which can be written in three ways i.e. Inline CSS, Internal CSS, or external CSS.

Similarly, Js (Java script) can be written in two ways i.e. External Js and Internal Js. Js can’t be written inline.

In Java script a web page is called a document.

JavaScript very important feature is that is runs directly into web browser...........

Java Script was only for interactive web pages but it becomes dynamic after coming MERN Stack...... Mern stack made JS as a full stack (Back end + Front end) language.

...........

1. To Start Javascript, go to project folder and create a new folder named javascript.
2. Go to folder ............. wamp/htdocs/Laravel/evssystem/javascript.

Inside javascript folder create one more fielder, named as Basic.

1. Got to project folder .......wamp\htdocs\Laravel\javascript\Basic.
2. Go into folder Basic, right click and select open with command prompt, and open with vs. code.
3. Now basic folder is opened in both command prompt, and vs.code .
4. Now in vs.code \Basic ....... create a new file .........named index.html.
5. In index.html ............. write ! and press Enter ..........
6. Change the title of the html file as .......... Javascript.
7. In the body create a heading tag ....... <h1> JavaScript </h1>

java scrip is written in side Php code, just like CSS which can be written in three ways i.e. Inline CSS, Internal CSS, or external CSS.

Similarly, Js (Java script) can be written in two ways i.e. External Js and Internal Js. Js can’t be written inline.

<body>

    <h1> JavaScript </h1>

    <script>

        console.log('Muddaser Hasnain , Malik , Web Developer');

    </script>

</body>

27 – July – 2025, Today I wrote my first JavaScript program. And learned to embed JavaScript code as internal and external JavaScript.

We worked in index.html , use internal JavaScript in index.html.

..............

We worked in index2.html, and also created index2.js which is an external js file.

...............

FSWD

Lecture # 31

Full Stack Development

We continue our work in index2.html, and created a Calculator.

Calculator.html , and Calculator.js files were created.

.......................

FSWD

Lecture # 32

Full Stack Development

**What is React:**

React is a free and open-source front-end JavaScript Library that aims to make building user-interfaces base on components more “seamless”. It is maintained by Meta and a community of individual developers and companies.

..............

React makes our websites faster real-time and big application, i.e. Frond end of Netflix, Facebook, web educators built in React-Js.

.............

We studied back end PHP and Laravel ............

But we can use PHP with Node.js or Laravel..........

For big project good to use Laravel ...... For smaller project we can use Node.js.

And at front end we can use ......React ......which is compatible with Laravel as well as Node.js.......

React always connect its front end with back end using Api’s .........

React is a front end frame work ......... back end we can use any thing php + Laravel, or Php + Node.js or Python.

[www.gtmetrix.com](http://www.gtmetrix.com). We can check speed of a website using this link.

-------------

**Requirement to install React:**

1. First of all, we need to install **Node.js.**

Java Script framework and libraries don’t need any heavy system requirement, we can even install and work on Pentium-3.

1. Install React.Js.......
2. Go to command prompt, navigate to your project folder.
3. Go to xamp\htdocs\project\
4. Run command :

npx \ create-react-app react-evs-proje

We can find upto date React frameworks from this link.

https://react.dev/link/cra

1. When React installation complete following messages will appear on command prompt.

269 packages are looking for funding

run `npm fund` for details

9 vulnerabilities (3 moderate, 6 high)

To address all issues (including breaking changes), run:

npm audit fix --force

Run `npm audit` for details.

Success! Created project at D:\Full Stack Web Developer\New folder\htdocs\laravel\evssystem\project

Inside that directory, you can run several commands:

npm start

Starts the development server.

npm run build

Bundles the app into static files for production.

npm test

Starts the test runner.

npm run eject

Removes this tool and copies build dependencies, configuration files

and scripts into the app directory. If you do this, you can’t go back!

We suggest that you begin by typing:

cd project

npm start

Happy hacking!

PS D:\Full Stack Web Developer\New folder\htdocs\laravel\evssystem>

FSWD

Lecture # 33

Full Stack Development

React – Class 2

Open your project i.e react-evs-project , in vs.code

We will see tree folders, and four files. As following

Node-modules

(System specific backbone folder for react project, node.js and react project cannot run with out this folder) When project is given to someone, node-modules folder is not given. We can delete

Node module folder but our project will stop working, we can include again by writing

Command , "npm Install" , in terminal . by running this command our node-modules folder will

be added again.

Public

Public folder contains all the files and settings of web services, main file in public folder is index.html.

And logo192,png , file , When we run our project by nmp start command , app.js is executed and this react logo is called on our main page in the browser .

Src

In source folder , we have app.css , app.js, Our project development / coding totally based on src folder.

Gitignore

Gitignore link our project with github.

Package-lock-json

Includes all the details of packages when package created, when release, which version.

Package-json

Include all packages used in react, on basis of which noted-modules run.

ReadMe.md

Details of project for users / user manual.

After Creating project using react library

Now we will create our source code as following.

1 = Create a folder named component in src folder

2 = In component folder, create file home.jsx.

Write following code:

*Import React from 'react';*

*Const Home – () => {*

*Return {*

*<div>*

*<h1> This is a Home component <h1>*

*</div>*

*};*

*};*

*Export default Home;*

3 = In file App.js , write following code .

Import React from 'react';

Import Home from './components/Home' // We import home.js page in App.js

Function App () {

Return {

<div>

<h1> Hellow, React <h1>

<Home / > // We called Home.js in App.js here.

</div>

};

}

Export default App;

4 = In file index.js , we will call App.js

Write following code:

*import React from 'react';*

*import ReactDOM from 'react-dom/client';*

*import App from './App'; // We import App.js in index.js*

*import { BrowserRouter } from 'react-router-dom';*

*const root = ReactDOM.createRoot(document.getElementById('root'));*

*root.render (*

*<BrowserRouter>*

*<App / > // We call App.js in index.js here.*

*</BrowserRouter>*

*);*

5 = **Install an extension rfc or type react extensions in extension menu and import extension ES7+ React/Redux/React-Native Snippets.**

**This extension we help us in coding........suggestions.**

6 = Now Create a new file in components About.jsx

**Make sure first letter of components file name is capital as About.jsx.**

Now with the help of above extension, we just type rfc+Enter,

And following code will automatically written.

Import React from 'React'

Export default function About () {

Return (

<div> About </div>

)

}

7 = Now import above About.jsx file in App.jsx

Import React from 'react';

Import Home from './components/Home' // We import home.js page in App.js

Import About from '/components / About' // We import About.js page in App.js

Function App () {

Return {

<div>

<h1> Hellow, React <h1>

<Home / > // We called Home.jsx in App.js here.

< About / > // We called About.jsx in App.js

</div>

};

}

Export default App;

8 = Now Similarly, create Contact.jsx and Products.jsx in components, and import & call them in App.js.

**We import and call, all new components(pages) created in src/components folder, and all pages /components will be imported in App.js and App.js is already imported and called in index.js.**

*9 = Now Create Contact.jsx file in components ,*

*In this file type rfc+Enter*

*Folowing code will automatically written.*

*import React from 'react'*

*export default function Contact() {*

*return (*

*);*

*}*

*Import and call this contact.jsx in App.js*

*10 = Now create file Products.jsx in src/components , and import and call it in App.js*

*In Products.jsx, type rfc + Enter*

*Following code will appear*

*import React from 'react';*

*export default function Products() {*

*return (*

*);*

*}*

*Now import and call all above self created src/components , i.e Abou.jsxt, Contact.jsx, and Producs.jsx in App.js.*

*Following will be the code.*

*import React from 'react';*

*import Home from "./components/Home";*

*import About from "./components/About";*

*import Contact from "./components/Contact";*

*import Products from "./components/Products";*

*import Header from "./components/includes/Header";*

*import Test from "./components/Test";*

*function App() { // function App() is creteing a function, we can*

*return ( write it as constructor instead of function i.e*

*<div> const App = () => { }*

*<Home />*

*<About />*

*<Contact />*

*< Products / >*

*</div>*

**);**

**}**

*export default App;*