Module 4) Laravel Framework

1.Introduction to Laravel

Laravel is a popular open-source PHP web application framework created by Taylor Otwell in 2011. It was developed to provide a more elegant and developer-friendly alternative to frameworks like CodeIgniter.

Laravel follows the MVC architecture and emphasizes simplicity, readability, and modern PHP practices. Laravel versions started from Laravel 1 (2011) to the latest Laravel 11. Major versions introduced features such as Eloquent ORM, Blade templating, routing improvements, middleware, task scheduling, and API authentication.

Laravel differs from other PHP frameworks due to its expressive syntax, strong ecosystem, built-in authentication, ORM support, queue handling, and extensive community support.

2.Laravel MVC Architecture

MVC stands for Model-View-Controller. It separates application logic into three interconnected components.

Model handles database interactions using Eloquent ORM. View manages the UI using Blade templates. Controller processes user requests and returns responses.

Example: A UserController fetches data from the User model and passes it to a Blade view for display.

3.Routing in Laravel

Routing in Laravel defines how URLs respond to requests. Routes are defined in routes/web.php or api.php. Named routes allow assigning names to routes for easy reference.

Route parameters accept dynamic values in URLs. Example: Route::get('/user/{id}', [UserController::class, 'show'])->name('user.show');

4.Blade Templating Engine

The Blade is a powerful templating engine in a Laravel framework.
The blade allows to use the templating engine easily, and it makes the syntax writing very simple. The blade templating engine provides its own structure such as conditional statements and loops.
To create a blade template, you just need to create a view file and save it with a .blade.php extension instead of .php extension.

The blade templates are stored in the /resources/view directory. The main advantage of using the blade template is that we can create the master template, which can be extended by other files.

Also You can make common header and footer also

5.Database Migrations and Eloquent ORM

Migrations are like version control for your database, allowing your team to define and share the application's database schema definition. If you have ever had to tell a teammate to manually add a column to their local database schema after pulling in your changes from source control, you've faced the problem that database migrations solve.

The Laravel Schema facade provides database agnostic support for creating and manipulating tables across all of Laravel's supported database systems. Typically, migrations will use this facade to create and modify database tables and columns.

Migrations allow version control for database schema using PHP code. Eloquent ORM simplifies database operations by using models. CRUD Example: Create: User::create([]) Read: User::all() Update: User::find(1)->update([]) Delete: User::destroy(1)

6.Laravel Middleware

Middleware acts as a bridge between a request and a response. It is a type of filtering mechanism.

Most of use in Login as for authentication Laravel includes a middleware that verifies whether the user of the application is authenticated or not. If the user is authenticated, it redirects to the home page otherwise, if not, it redirects to the login page.

- Middleware type

We need to register each and every middleware before using it. There are Three types of Middleware in Laravel.

Global Middleware: use and applied in complete website in single time

Route Middleware: use and apply in specific single route at a time

Group Middleware: use and applied in specific Route pages Exa if 20 page and want apply on 2 page // bunch of the routes

The Global Middleware will run on every HTTP request of the application, whereas the Route Middleware will be assigned to a specific route. The middleware can be registered at app/Http/Kernel.php. This file contains two properties: $middleware and $routeMiddleware. $middleware property is used to register Global Middleware and $routeMiddleware property is used to register route specific middleware.

Middleware filters HTTP requests before they reach controllers. Used for authentication, logging, and CORS. Example: auth middleware restricts access to authenticated users only.

7.Laravel Authentication

Laravel provides a powerful, secure, and flexible authentication system that allows developers to implement user login, registration, password reset, and authorization with minimal configuration.

Laravel's authentication is designed to handle common security concerns while remaining easy to customize.

Laravel provides built-in authentication using Breeze, Jetstream, or UI. Guards define authentication methods, providers define user data source. Authentication setup includes migrations, routes, controllers, and views.

- Features of Laravel Authentication
  - User registration and login
  - Password hashing using bcrypt
  - Password reset and email verification
  - Session-based and token-based authentication
  - Role-based access control
  - Protection against CSRF and SQL injection

8.Testing in Laravel

Testing is a crucial part of modern web application development. It helps ensure that an application works as expected, reduces bugs, and improves overall code quality. Laravel provides a robust and developer-friendly testing environment that makes writing and running tests simple and efficient.

- Importance of Testing in Web Applications
  - Ensures application reliability and stability
  - Detects bugs early in the development cycle
  - Improves code maintainability
  - Reduces cost of fixing errors after deployment
  - Increases confidence while updating or refactoring code

- Testing Tools Available in Laravel

  Laravel comes with several built-in testing tools:

  - PHPUnit: Default testing framework used by Laravel Supports unit and feature testing
  - Feature Tests: Test application behavior including routes, controllers, and viewsSimulate HTTP requests
  - Unit Tests: Test individual classes or functions independently
  - Database Testing: Uses database migrations and transactions Supports in-memory SQLite databases for faster tests