Part 1:

To install Laravel, I followed these steps:

1. After I have installed PHP and Composer, I create a new Laravel Project via the Composer **create-project** command:

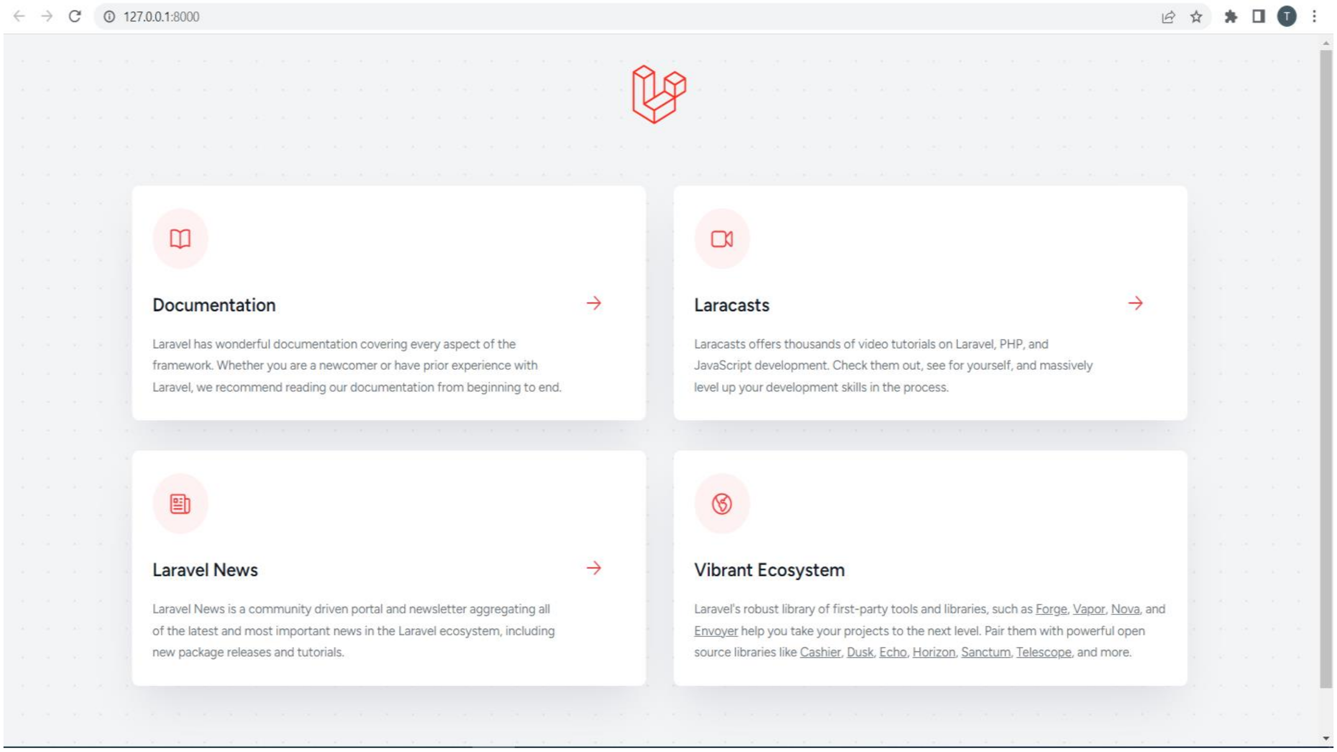
   composer create-project laravel/laravel assignment13

2. After the project has been created, start Laravel's local development server using the Laravel’s Artisan CLI command:

   cd assignment13

   php artisan serve

Below Screen sort shows the Running of my development Server



Documentation

Laravel has wonderful documentation covering every aspect of the framework. Whether you are a newcomer or have prior experience with Laravel, we recommend reading our documentation from beginning to end.

Laracasts

Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.

Laravel News

Laravel News is a community driven portal and newsletter aggregating all of the latest and most important news in the Laravel ecosystem, including new package releases and tutorials.

Vibrant Ecosystem

Laravel's robust library of first-party tools and libraries, such as Forge, Vapor, Nova, and Envoyer help you take your projects to the next level. Pair them with powerful open source libraries like Cashier, Dusk, Echo, Horizon, Sanctum, Telescope, and more.

Part 2:

Describing the purpose of each folder in a Laravel Project:

app: This folder contains the core application code, including the models, controllers, and other classes that define the business logic and functionality of the application.

bootstrap: This folder contains the files responsible for bootstrapping the Laravel framework and initializing the application. It includes the application's service providers, the application configuration, and the initial files loaded when the framework starts.

config: The config folder contains various configuration files for the Laravel application. It includes files for database connections, application settings, caching configuration, and other settings that determine how the application behaves.

database: This folder is used for database-related files. It includes migration files, which are used to create and modify database tables, as well as seed files, which are used to populate the database with initial data.

public: The public folder is the document root of your application. It contains the front controller (index.php) and the assets (such as CSS, JavaScript, and images) that are accessible to the public. This folder is typically where you put your publicly accessible files.

resources: The resources folder contains views, language files, and other assets that are used by your application. It includes subfolders for views, language localization files, and assets such as CSS and JavaScript files.

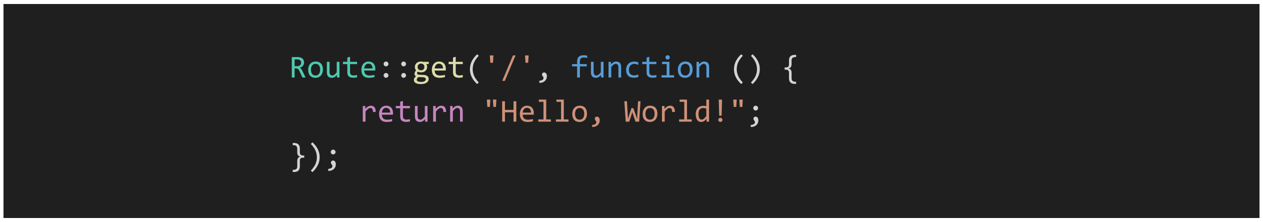
routes: The routes folder contains the route definition files for your application. It includes web.php for defining web routes and api.php for defining API routes. Routes determine how the application responds to incoming requests and which controllers or closures should be invoked.

storage: This folder is used for storing various files generated by the application. It includes subfolders for app-specific storage, such as logs, framework-generated files, and cached views. It also contains a symbolic link to the "public/storage" directory for serving public files stored in the storage/app/public directory.

tests: The tests folder is used for storing automated tests for your application. It includes subfolders for unit tests, feature tests, and other types of tests. Laravel provides a testing framework that allows you to write and run tests to verify the functionality of your application.

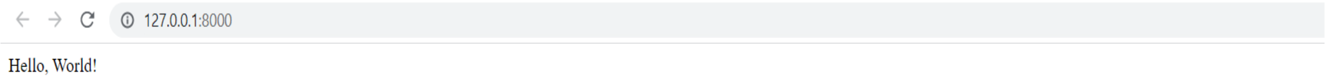
vendor: The vendor folder contains the dependencies installed via Composer, which is a dependency management tool used by Laravel. It includes the third-party libraries and packages that your application relies on. This folder is typically generated and managed by Composer, and you usually don't need to modify its contents directly.

I have Created a new route in my Laravel project that displays a simple "Hello, World!" message.The route is:



```
Route::get('/', function () {
    return "Hello, World!";
});
```

Secreensort of this route in server:



127.0.0.1:8000

Hello, World!