**Module 13 Assignment**

**Part 1: Laravel Installation**

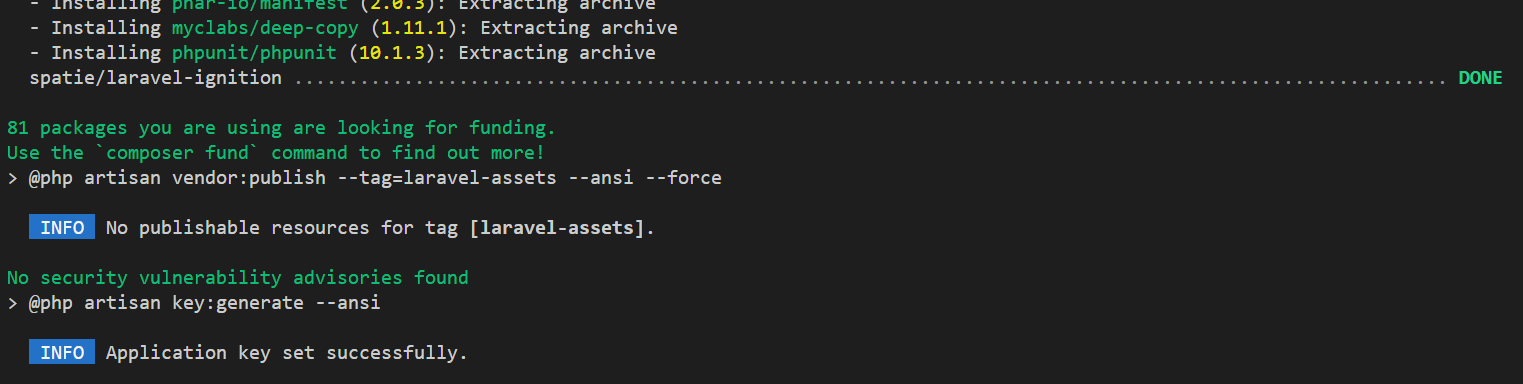
1. Install the latest version of Laravel using either Composer or the Laravel installer.
2. Document the steps you took to install Laravel.
3. Verify that Laravel is installed and working correctly by creating a new Laravel project and running the development server. Take a screenshot of the running server.

**Solution:**

**Part1**:

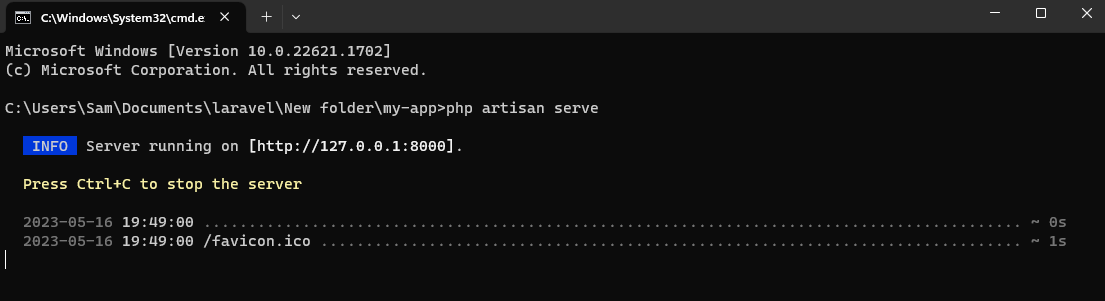
1. **Installing laravel:** To install laravel I have used the following steps given bellow:

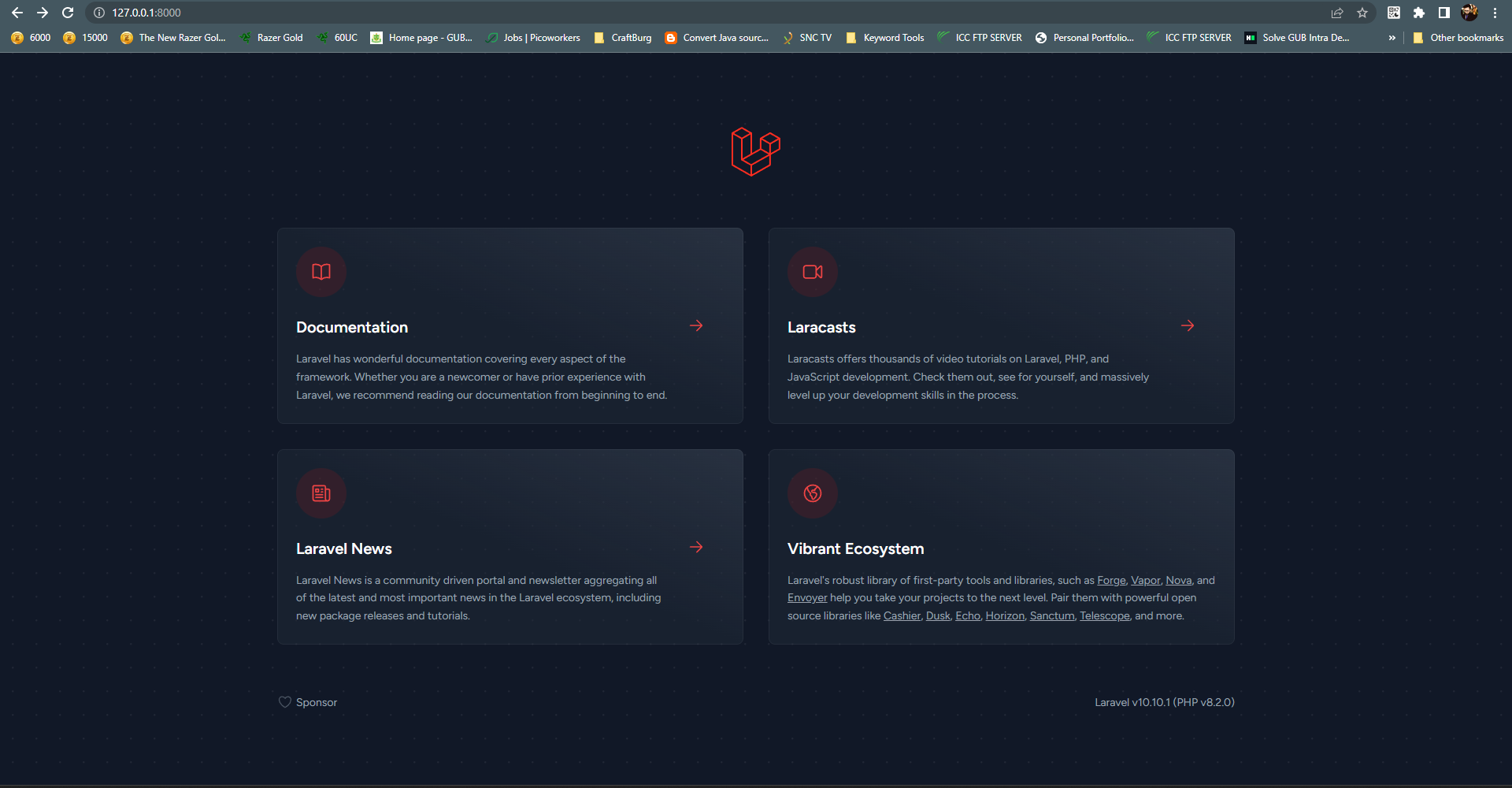
* **Step1:** At first I have downloaded composer from the following link: <https://getcomposer.org/> and install it.
* **Step2:** Then I have created a folder and open it in VS code.
* **Step3:** Then I have opened terminal and enter the following code: composer create-project laravel/laravel my-app **to create a new project** named **my-app**.



1. **Running Development Server:** To run the development server I have used following steps:

* **Step1**: Open the folder where I have created the project.
* **Step2**: Open cmd on that folder and type the following line: php artisan server
* **Step3:** Screenshot of the running server:





**Part2:** **Laravel Folder Structure**

♦ Describe the purpose of each of the following folders in a Laravel project:

* app
* bootstrap
* config
* database
* public
* resources
* routes
* storage
* tests
* Vendor

**Solution**: Here is a description of the purpose of each folder in a typical Laravel project:

* **app**: This folder contains the core application files, including the models, controllers, and other classes specific to your application's business logic. The app folder is where you will primarily write and organize your application's code.
* **bootstrap**: The bootstrap folder contains files that are responsible for bootstrapping and initializing the Laravel framework. It includes files like app.php, which loads the necessary dependencies and sets up the environment for your application.
* **config**: The config folder contains configuration files for various aspects of your Laravel application, such as database connections, application settings, caching configurations, and more. You can modify these files to customize the behavior of your application.
* **database**: The database folder is where you define your application's database migrations, seeders, and factories. Migrations are used to create and modify database tables, seeders are used to populate the database with initial data, and factories help in generating dummy data for testing purposes.
* **public**: The public folder is the web root of your application. It contains the entry point (index.php) for your application and serves as the public-facing directory. Files placed in this folder can be accessed directly by the users via the web browser, such as CSS, JavaScript, and image files.
* **resources**: The resources folder contains the views, language files, and assets used by your application. It includes subfolders like views for storing Blade templates (PHP-based templates), lang for language files, and assets for assets like CSS, JavaScript, and images. These resources are typically used to render the front-end of your application.
* **routes**: The routes folder contains route definitions for your application. Routes define the URLs and corresponding actions or controller methods that should be executed when a specific URL is accessed. The web.php file contains routes for web application routes, while api.php defines routes for API endpoints.
* **storage**: The storage folder is used to store various files generated or used by your application. It includes subfolders like app (for application-specific files), framework (for files generated by Laravel), logs (for log files), and more. It is also where uploaded files and cached files are stored by default.
* **tests**: The tests folder is where you write automated tests for your application. It includes subfolders for different types of tests, such as unit tests, feature tests, and browser tests. You can use testing frameworks like PHPUnit and Laravel Dusk to write and execute tests to ensure the functionality and integrity of your application.
* **vendor**: The vendor folder contains the dependencies installed via Composer, which is the package manager used by Laravel. It includes all the external libraries and packages required by your application, such as Laravel itself and other third-party packages. You typically don't modify the files in this folder directly.

♦ **Create a new route in your Laravel project that displays a simple "Hello, World!" message. Take a screenshot of the running route.**

**Solution:** To create a new route in my Laravel project that displays a simple "Hello, World!" message I have used the following steps:

* **Step1**: Open the routes/web.php file.
* **Step2**: Inside the web.php file, create a new route and add the following route definition:

Route::get('/', function () {

    return view('welcome');

});

Route::get('/', function () {

    return view('HelloWorld');

});

* **Step3**: Create a new HelloWorld.blade.php file in resources/views
* **Step4**: Inside the HelloWorld.blade.php file, add the following codes:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Hello World!</title>

</head>

<body>

    <h1 style="text-align: center;">Hello World!</h1>

</body>

</html>

* **Step5**: Save the file and run the server <http://127.0.0.1:8000>
* **Output:**

