

Laravel Installation and Folder Structure Assignment

Part 1: Laravel Installation

Step 1: Install Composer

Laravel utilizes Composer to manage its dependencies. First, download Composer:

1. Open Terminal.
2. Run the following command:

```
curl -sS https://getcomposer.org/installer | php
```

3. Move the `composer.phar` file to make it globally accessible:

```
mv composer.phar /usr/local/bin/composer
```

4. Verify the installation by running:

composer

You should see the Composer's version and a list of commands.

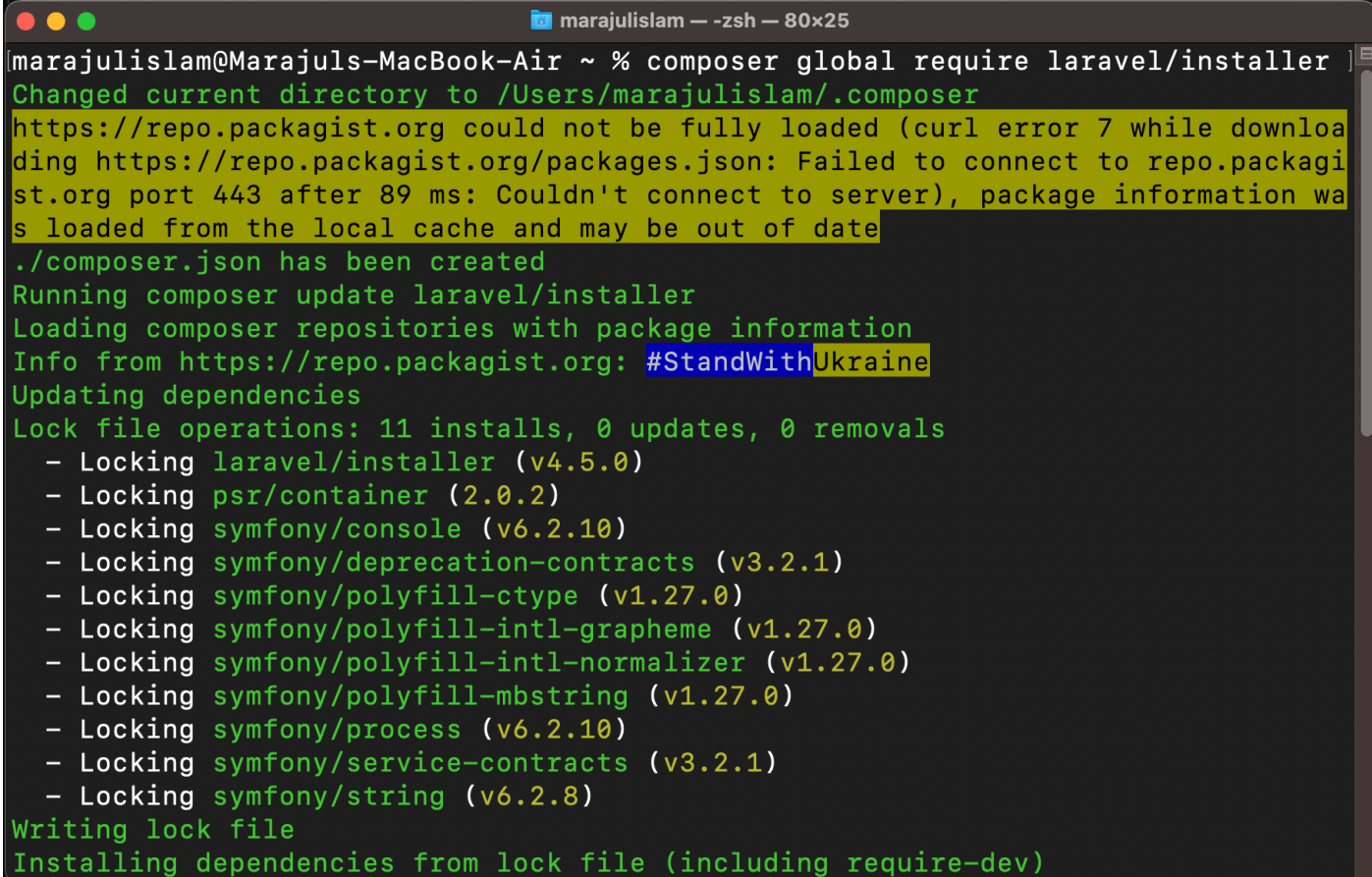
[illegible]

Step 2: Install Laravel

1. Install Laravel by running:

```
composer global require laravel/installer
```

2. Make sure that the ~/.composer/vendor/bin directory is in your system's PATH.

A terminal window titled 'marajulislam - zsh - 80x25' showing the execution of 'composer global require laravel/installer'. The output shows the current directory changed to the user's global composer directory, a warning about a network error (curl error 7) when fetching package information from packagist.org, and the successful creation of the composer.json file. It then lists the locked dependencies for the installation, including laravel/installer and various Symfony components, and finally shows the lock file being written and dependencies being installed.

```
marajulislam@Marajuls-MacBook-Air ~ % composer global require laravel/installer ]
Changed current directory to /Users/marajulislam/.composer
https://repo.packagist.org could not be fully loaded (curl error 7 while downloading https://repo.packagist.org/packages.json: Failed to connect to repo.packagist.org port 443 after 89 ms: Couldn't connect to server), package information was loaded from the local cache and may be out of date
./composer.json has been created
Running composer update laravel/installer
Loading composer repositories with package information
Info from https://repo.packagist.org: #StandWithUkraine
Updating dependencies
Lock file operations: 11 installs, 0 updates, 0 removals
- Locking laravel/installer (v4.5.0)
- Locking psr/container (2.0.2)
- Locking symfony/console (v6.2.10)
- Locking symfony/deprecation-contracts (v3.2.1)
- Locking symfony/polyfill-ctype (v1.27.0)
- Locking symfony/polyfill-intl-grapheme (v1.27.0)
- Locking symfony/polyfill-intl-normalizer (v1.27.0)
- Locking symfony/polyfill-mbstring (v1.27.0)
- Locking symfony/process (v6.2.10)
- Locking symfony/service-contracts (v3.2.1)
- Locking symfony/string (v6.2.8)
Writing lock file
Installing dependencies from lock file (including require-dev)
```

Step 3: Create a New Laravel Project

1. To create a new Laravel project, use the new command:

```
laravel new project-name
```

2. Replace project-name with your desired project name.

You now have a new Laravel project in a directory with your specified name.

```
Assignment-13 — -zsh — 81x24
Last login: Mon May 15 20:30:11 on ttys000
marajulislam@Marajuls-MacBook-Air Assignment-13 % composer create-project laravel/laravel /laravel/Laravel-Test
Creating a "laravel/laravel" project at "./Laravel-Test"
Installing laravel/laravel (v10.2.0)
- Downloading laravel/laravel (v10.2.0)
- Installing laravel/laravel (v10.2.0): Extracting archive
Created project in /Applications/XAMPP/xamppfiles/htdocs/Practice Laravel/Module-13 Laravel Foundation Step-12/Assignment-13/Laravel-Test
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 107 installs, 0 updates, 0 removals
- Locking brick/math (0.11.0)
- Locking dflydev/dot-access-data (v3.0.2)
- Locking doctrine/inflector (2.0.6)
- Locking doctrine/lexer (3.0.0)
- Locking dragonmantank/cron-expression (v3.3.2)
- Locking egulias/email-validator (4.0.1)
- Locking fakerphp/faker (v1.21.0)
- Locking filp/whoops (2.15.2)
- Locking fruitcake/php-cors (v1.2.0)
- Locking graham-campbell/result-type (v1.1.1)
- Locking guzzlehttp/guzzle (7.6.0)
```

Step 4: Run the New Laravel Project

1. To Run a new Laravel project, use the new command:

`php artisan serve`

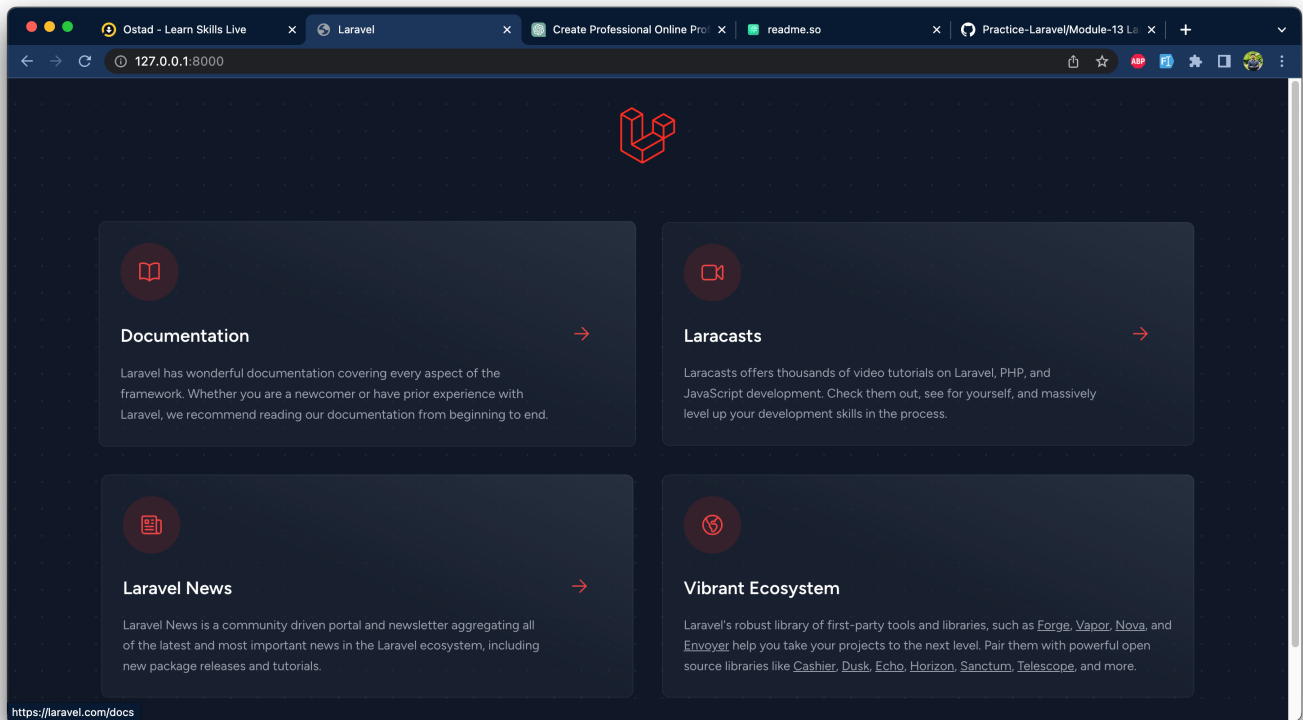
```
Laravel-Test — php • php artisan serve — 86x23
Last login: Mon May 15 20:32:37 on ttys000
marajulislam@Marajuls-MacBook-Air Laravel-Test % php artisan serve

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

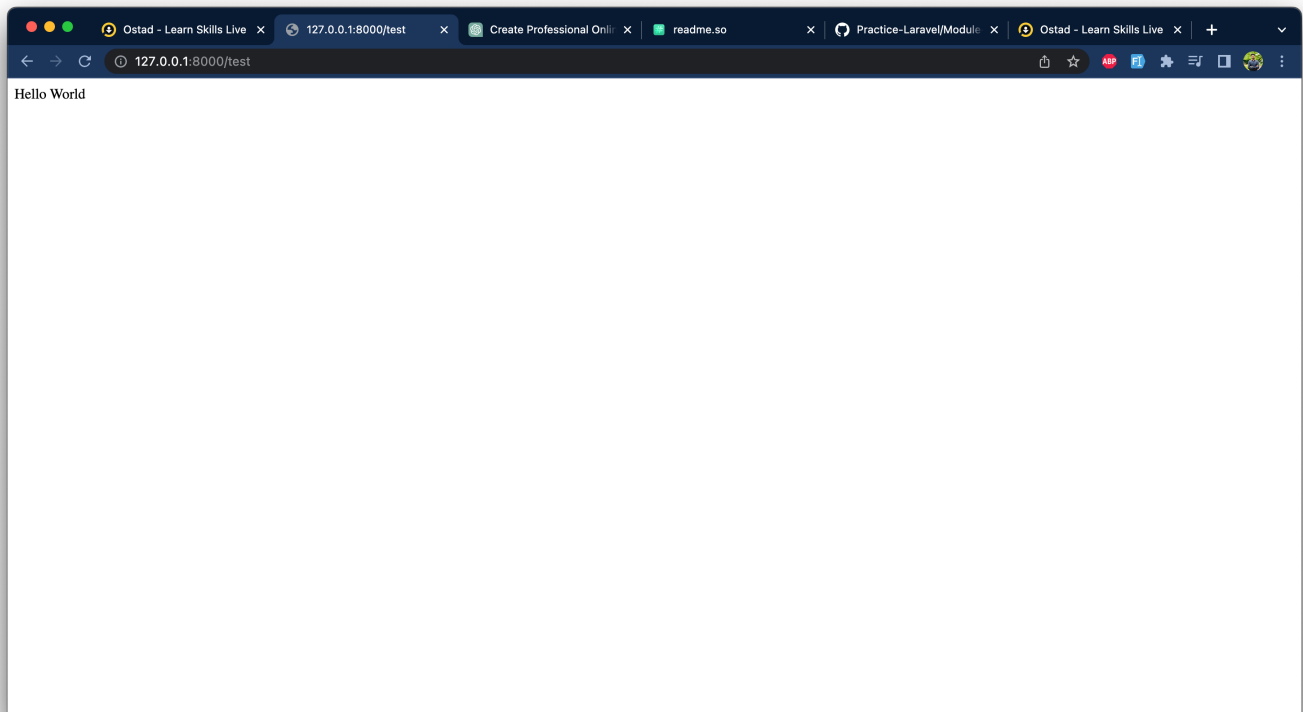
Press Ctrl+C to stop the server

2023-05-15 20:35:47 ..... ~ 0s
2023-05-15 20:35:47 /favicon.ico ..... ~ 1s
```

Step 4: Show the New Laravel Project in Browser



Step 4: Create a new route in the Laravel project that displays a simple "Hello, World!" message



Part 2: Laravel Folder Structure

app

This directory contains the core code of your application. This is where all your custom classes, exceptions, controllers, middleware, providers, and traits live.

bootstrap

The bootstrap directory contains the app.php file which bootstraps the framework. It also houses the cache files for speed optimization.

config

In the config directory, you'll find all of the configuration files for your application. This is a great place to look for various settings defined by Laravel and your own custom settings.

database

This directory contains your database migration and seeds. It's also where you would store all your SQLite databases, should you choose to use SQLite.

public

This is the directory that your web server points at. It contains the index.php file, which is the entry point for all requests entering your application.

resources

The resources directory houses your views, raw assets (LESS, SASS, CoffeeScript), and localization files.

routes

All of the route definitions for your application are housed in the routes directory. By default, several route files are included with Laravel: web.php, api.php, console.php, and channels.php.

storage

The storage directory contains your compiled Blade templates, file-based sessions, file caches, and other files generated by the framework. This folder is segregated into app, framework, and logs directories.

tests

This directory contains your automated tests. An example PHPUnit is provided out of the box.

vendor

The vendor directory contains your Composer dependencies. It is also the default location for package development.

By understanding the purpose of each folder in the Laravel framework, you can better navigate and manage your projects. Always keep in mind that the Laravel framework is very flexible, and these directories can be configured to your liking.