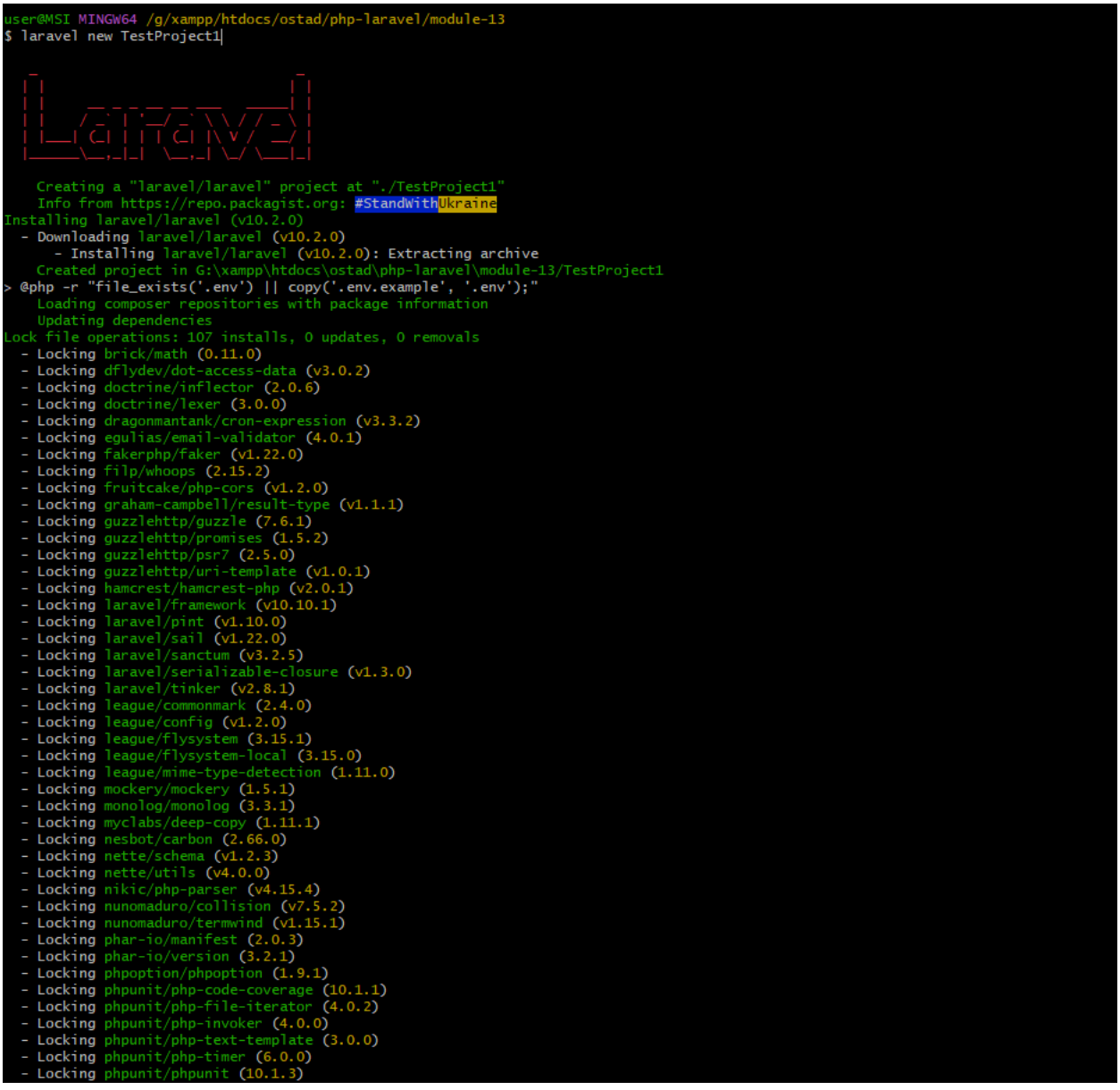
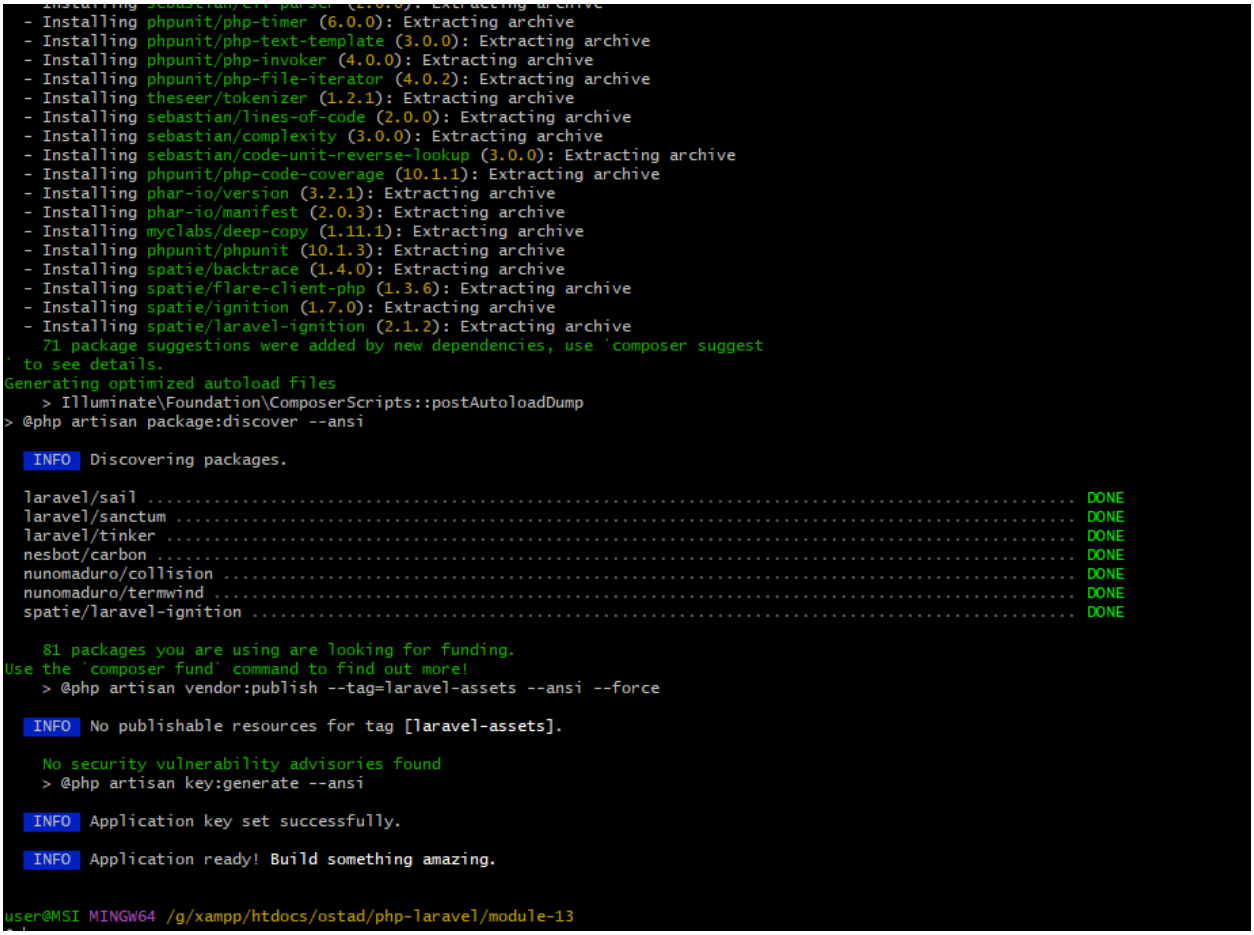
Part-1 : Laravel installation Process

I already had Laravel installed globally. So I used the following command in the project folder

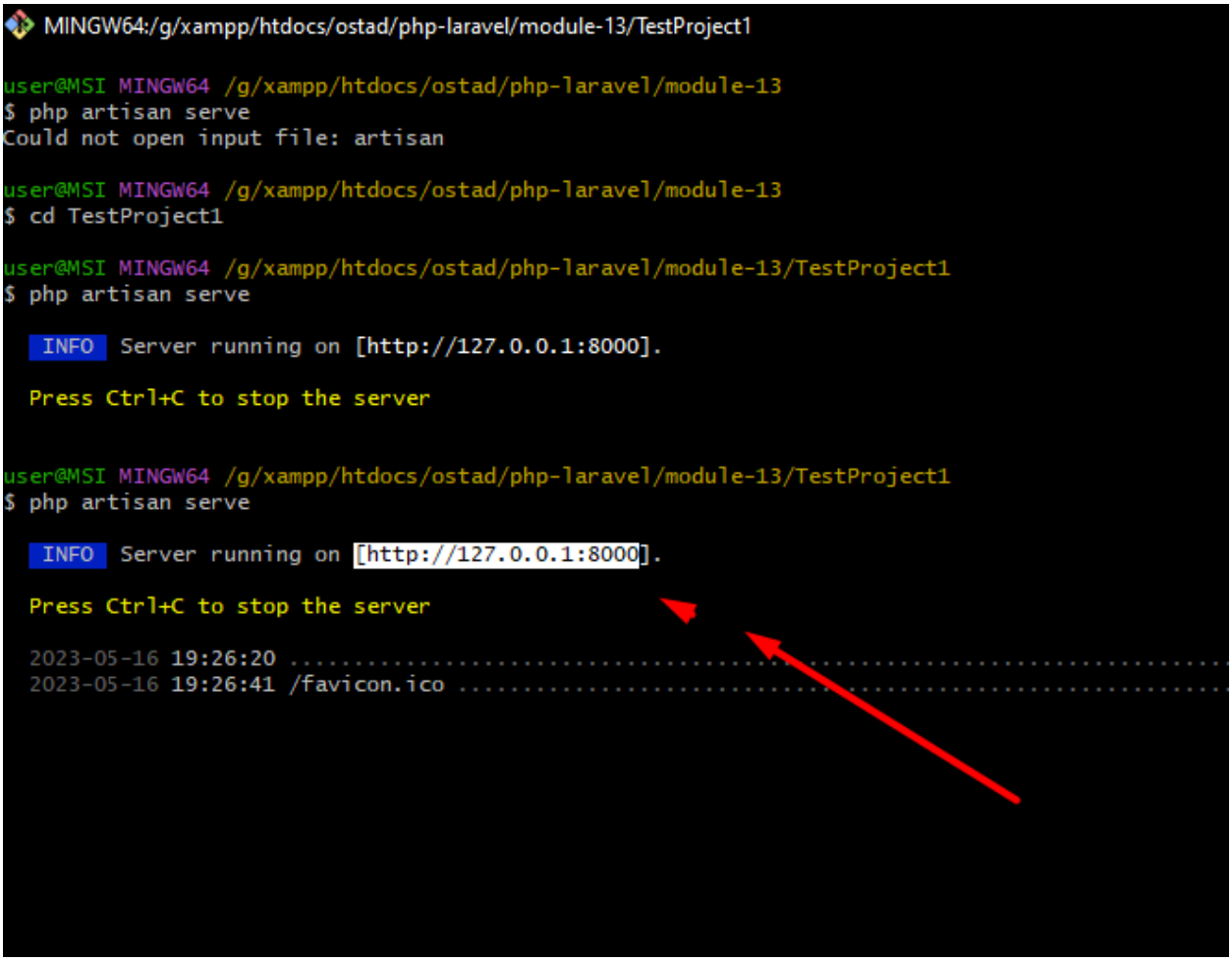
```
laravel new TestProject1
```

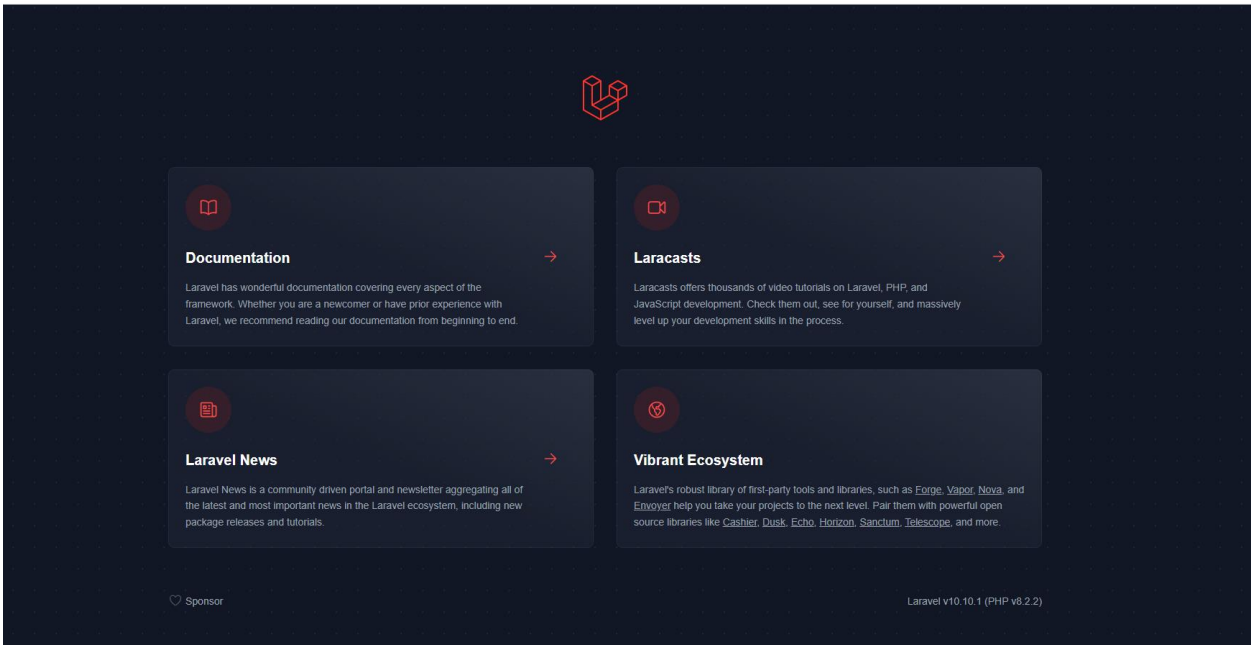




Installation is done.

Validating the Installation:


MINGW64:/g/xampp/htdocs/ostad/php-laravel/module-13/TestProject1
user@MSI MINGW64 /g/xampp/htdocs/ostad/php-laravel/module-13
$ php artisan serve
Could not open input file: artisan
user@MSI MINGW64 /g/xampp/htdocs/ostad/php-laravel/module-13
$ cd TestProject1
user@MSI MINGW64 /g/xampp/htdocs/ostad/php-laravel/module-13/TestProject1
$ php artisan serve
INFO Server running on [http://127.0.0.1:8000].
Press Ctrl+C to stop the server
user@MSI MINGW64 /g/xampp/htdocs/ostad/php-laravel/module-13/TestProject1
$ php artisan serve
INFO Server running on [http://127.0.0.1:8000].
Press Ctrl+C to stop the server
2023-05-16 19:26:20
2023-05-16 19:26:41 /favicon.ico


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.
Sponsor
Laravel v10.10.1 (PHP v8.2.2)

Part-2: Laravel folder Structure

app: This directory holds the essential code of the application, encompassing models, controllers, and other classes that define the business logic and functionality.

bootstrap: The bootstrap directory contains the initialization files for the framework and its components. It consists of files responsible for autoloading, error handling, and configuration loading.

config: The config directory stores configuration files for various aspects of the Laravel application, such as database settings, mail configurations, service providers, and more. These files can be customized to modify the behavior of your application.

database: This directory manages the database-related files of the application. It includes migrations, which handle the creation and modification of database tables, and seeders, which allow you to populate the database with initial data.

public: The public directory serves as the web root of your Laravel application. It contains the entry point for incoming HTTP requests, the index.php file, as well as publicly accessible assets like CSS, JavaScript, and image files.

resources: The resources directory holds various resource files used by the application, such as views, language files, and assets that require compilation,

such as SCSS or JavaScript files. Views are stored in the views subdirectory and are responsible for rendering HTML templates.

routes: The routes directory contains route definition files that determine how the application responds to different HTTP requests. It includes web.php for handling web requests and api.php for handling API requests. In these files, you can define routes, middleware, and route groups.

storage: The storage directory is used to store temporary and long-term files generated by the application. This includes logs, session files, cache files, and uploaded files. Additionally, it serves as the default location for Laravel's file-based caching and session storage.

tests: The tests directory comprises automated tests for the application. It contains test cases that cover different parts of the application's functionality. Laravel provides a testing framework that facilitates the writing and execution of tests.

vendor: The vendor directory is managed by Composer, a dependency management tool used by Laravel. It contains all the third-party libraries and packages that your application depends on. Composer handles the installation and updates of these dependencies based on the requirements defined in the composer.json file.

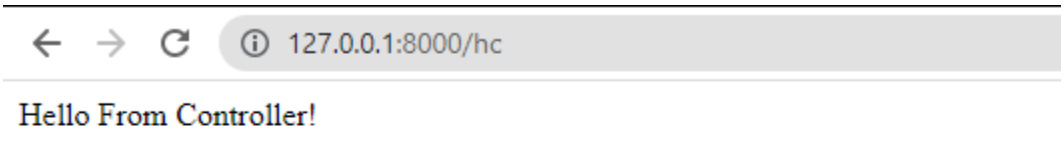
Creating New Route:

```php
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\HelloController;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/

Route::get('/', function () {
    return view('welcome');
});

Route::get('/hello' , function (){
    return 'Hello World';

});

Route::get('/hc', [HelloController::class,'greet'] );
```

Hello World

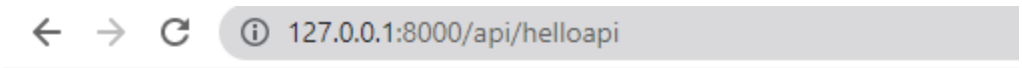
Hello from controller

```php
namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HelloController extends Controller
{
    function greet(){
        return 'Hello From Controller!';
    }
    function greetAPI(){
        return 'Hello API';
    }
}
```



Hello From Controller!

API:



Hello API