Module 15 Assignment

Syed Amir Hamza

Task 1: Request Validation

Implement request validation for a registration form that contains the following fields: name, email, and password. Validate the following rules:

name: required, string, minimum length 2.
email: required, valid email format.
password: required, string, minimum length 8.

Answer

```
public function ValidateAndRegister(Request $request)
{
 //   dd($request->all());

    $request->validate(
        [
            'name'=>'required|string|min:2',
            'email'=>'required|email',
            'password'=>'required|string|min:8'
        ]
        );
        return back()->withSuccess('Register Success !!');

}
```

Task 2: Request Redirect

Create a route /home that redirects to /dashboard using a 302 redirect.

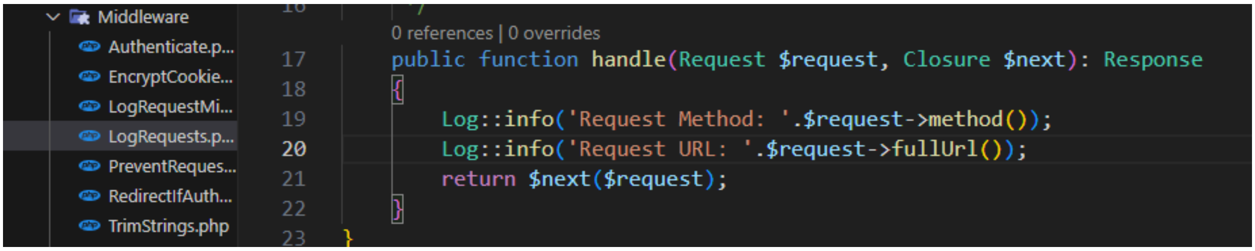
Answer

```
Route::get('/home', function () {
    return redirect('/dashboard', 302);
});
```

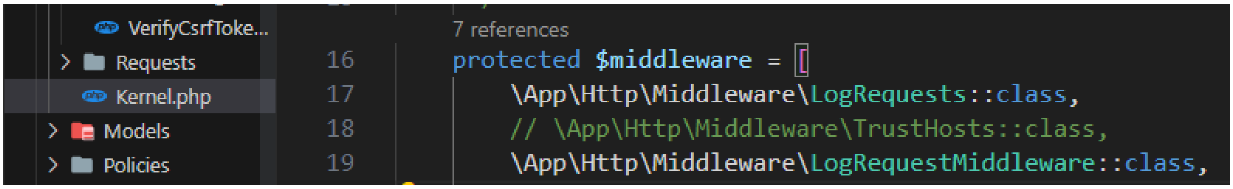
Task 3: Global Middleware

Create a global middleware that logs the request method and URL for every incoming request. Log the information to the Laravel log file.

Answer



```
    0 references | 0 overrides
    public function handle(Request $request, Closure $next): Response
    {
        Log::info('Request Method: '.$request->method());
        Log::info('Request URL: '.$request->fullUrl());
        return $next($request);
    }
}
```

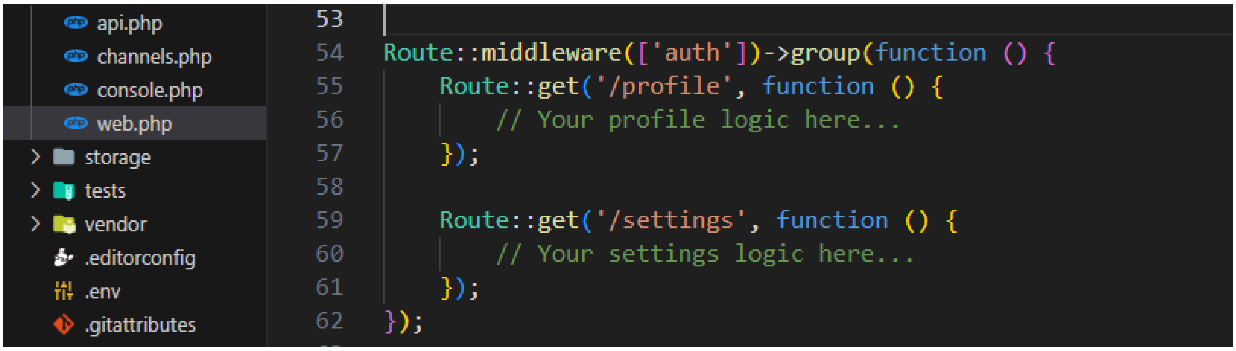


```
    7 references
    protected $middleware = [
        \App\Http\Middleware\LogRequests::class,
        // \App\Http\Middleware\TrustHosts::class,
        \App\Http\Middleware\LogRequestMiddleware::class,
```

Task 4: Route Middleware

Create a route group for authenticated users only. This group should include routes for /profile and /settings. Apply a middleware called AuthMiddleware to the route group to ensure only authenticated users can access these routes.

Answer



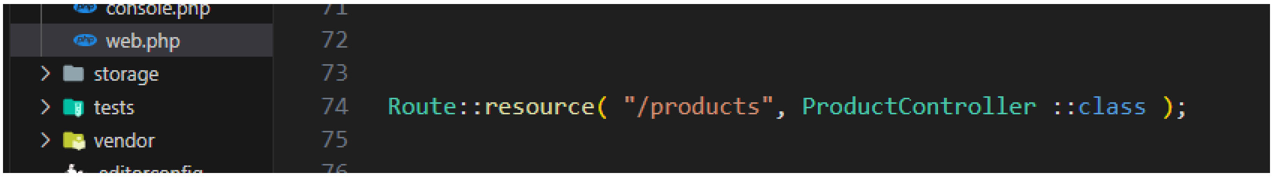
```

Route::middleware(['auth'])->group(function () {
    Route::get('/profile', function () {
        // Your profile logic here...
    });

    Route::get('/settings', function () {
        // Your settings logic here...
    });
});
```

Task 5: Controller

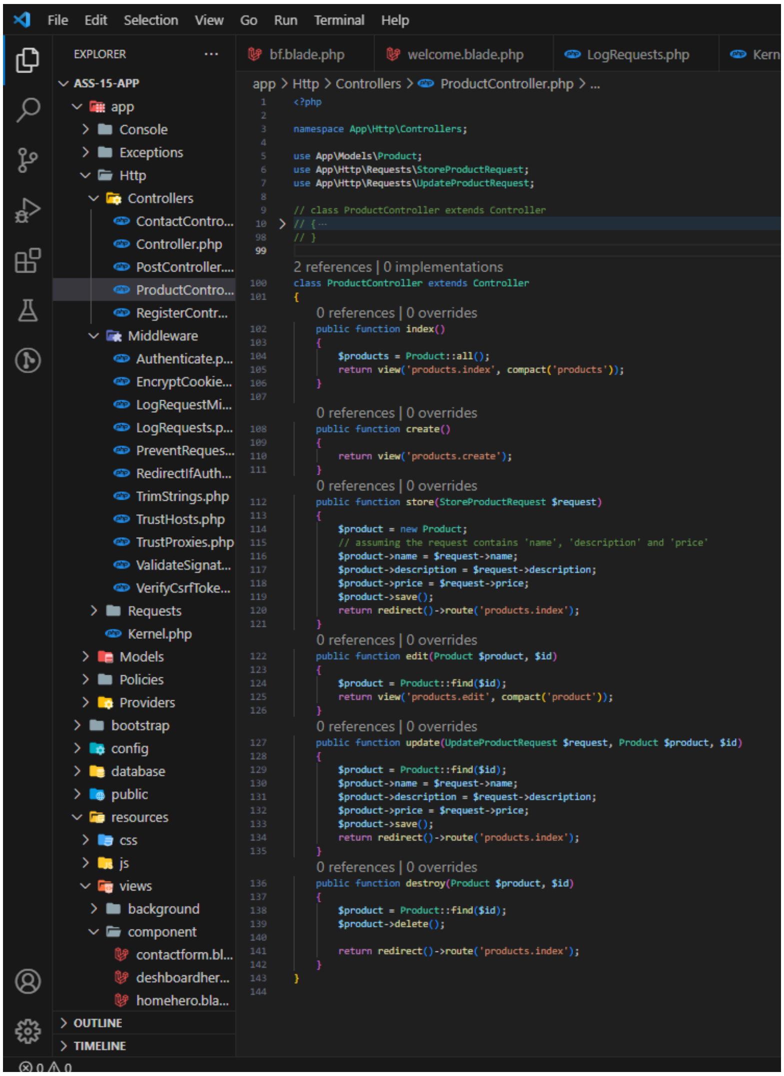
Create a controller called ProductController that handles CRUD operations for a resource called Product. Implement the following methods:

index(): Display a list of all products.
create(): Display the form to create a new product.
store(): Store a newly created product.
edit($id): Display the form to edit an existing product.
update($id): Update the specified product.
destroy($id): Delete the specified product.

Answer



```php
Route::resource( "/products", ProductController ::class );
```

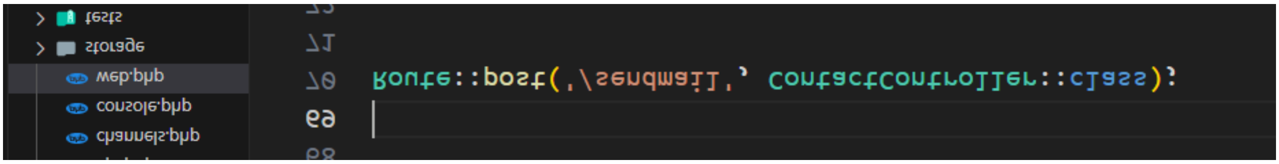


```php
<?php

namespace App\Http\Controllers;

use App\Models\Product;
use App\Http\Requests\StoreProductRequest;
use App\Http\Requests\UpdateProductRequest;

// class ProductController extends Controller
// {···
// }

2 references | 0 implementations
class ProductController extends Controller
{
    0 references | 0 overrides
    public function index()
    {
        $products = Product::all();
        return view('products.index', compact('products'));
    }

    0 references | 0 overrides
    public function create()
    {
        return view('products.create');
    }
    0 references | 0 overrides
    public function store(StoreProductRequest $request)
    {
        $product = new Product;
        // assuming the request contains 'name', 'description' and 'price'
        $product->name = $request->name;
        $product->description = $request->description;
        $product->price = $request->price;
        $product->save();
        return redirect()->route('products.index');
    }
    0 references | 0 overrides
    public function edit(Product $product, $id)
    {
        $product = Product::find($id);
        return view('products.edit', compact('product'));
    }
    0 references | 0 overrides
    public function update(UpdateProductRequest $request, Product $product, $id)
    {
        $product = Product::find($id);
        $product->name = $request->name;
        $product->description = $request->description;
        $product->price = $request->price;
        $product->save();
        return redirect()->route('products.index');
    }
    0 references | 0 overrides
    public function destroy(Product $product, $id)
    {
        $product = Product::find($id);
        $product->delete();

        return redirect()->route('products.index');
    }
}
```

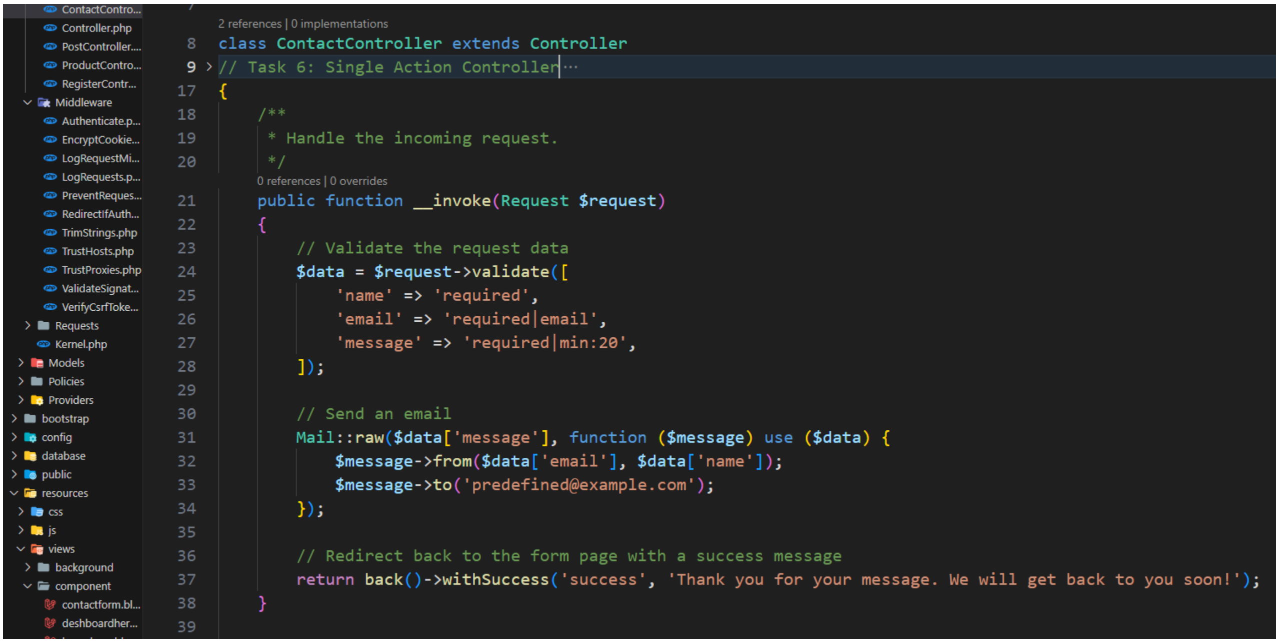
Task 6: Single Action Controller

Create a single action controller called ContactController that handles a contact form submission. Implement the __invoke() method to process the form submission and send an email to a predefined address with the submitted data.

Answer



```php
Route::post('/sendmail', ContactController::class);
```

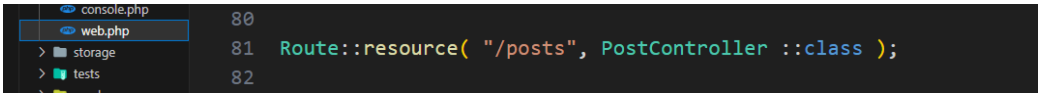


```php
class ContactController extends Controller
// Task 6: Single Action Controller···
{
    /**
     * Handle the incoming request.
     */
    public function __invoke(Request $request)
    {
        // Validate the request data
        $data = $request->validate([
            'name' => 'required',
            'email' => 'required|email',
            'message' => 'required|min:20',
        ]);

        // Send an email
        Mail::raw($data['message'], function ($message) use ($data) {
            $message->from($data['email'], $data['name']);
            $message->to('predefined@example.com');
        });

        // Redirect back to the form page with a success message
        return back()->withSuccess('success', 'Thank you for your message. We will get back to you soon!');
    }
```

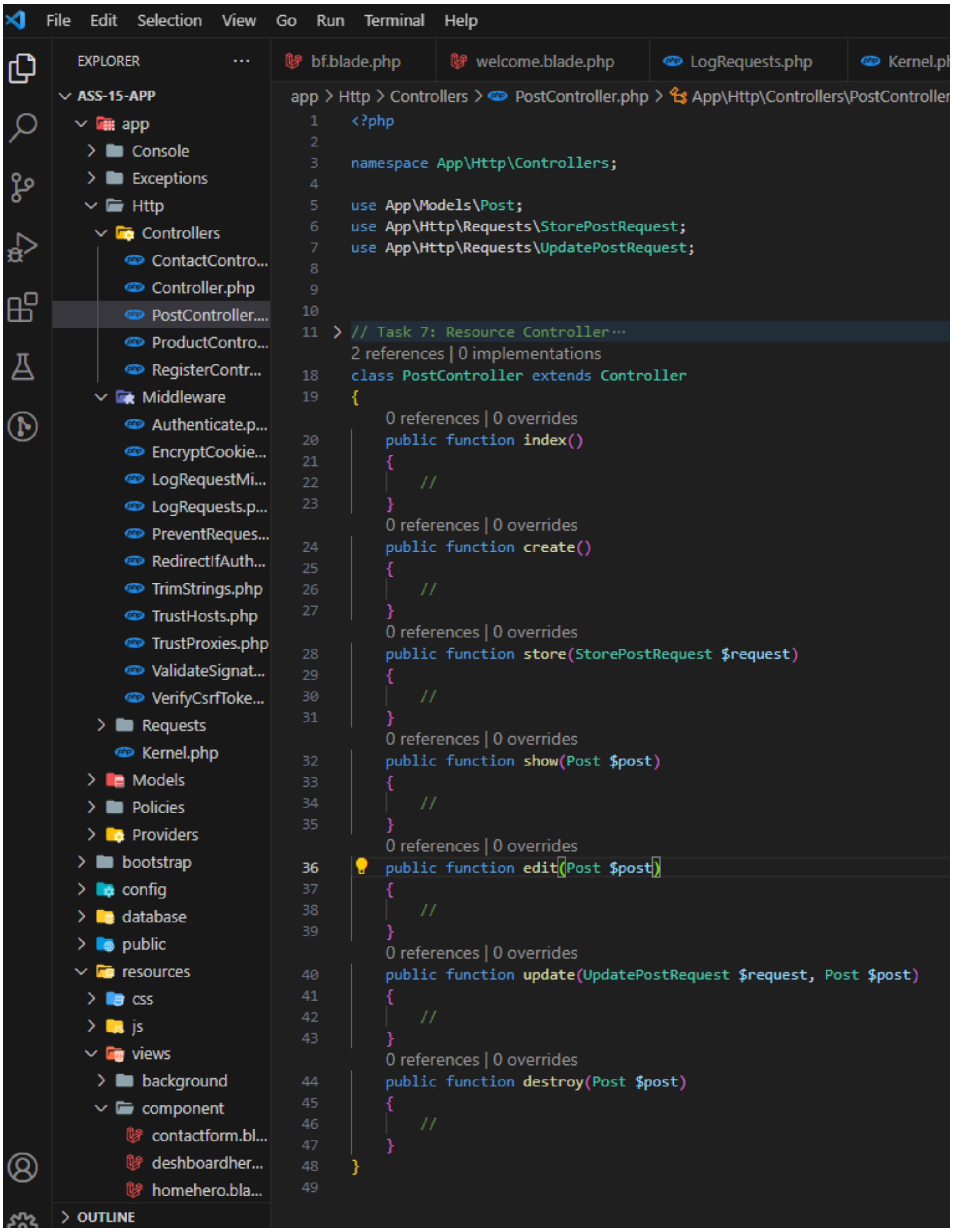
Task 7: Resource Controller

Create a resource controller called PostController that handles CRUD operations for a resource called Post. Ensure that the controller provides the necessary methods for the resourceful routing conventions in Laravel.

Answer



```php
Route::resource( "/posts", PostController ::class );
```

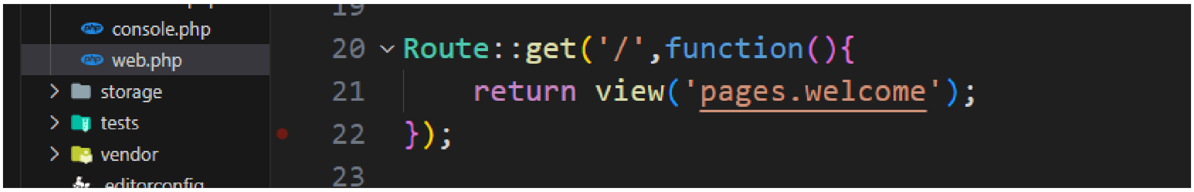


```php
<?php

namespace App\Http\Controllers;

use App\Models\Post;
use App\Http\Requests\StorePostRequest;
use App\Http\Requests\UpdatePostRequest;


// Task 7: Resource Controller···
2 references | 0 implementations
class PostController extends Controller
{
    0 references | 0 overrides
    public function index()
    {
        //
    }
    0 references | 0 overrides
    public function create()
    {
        //
    }
    0 references | 0 overrides
    public function store(StorePostRequest $request)
    {
        //
    }
    0 references | 0 overrides
    public function show(Post $post)
    {
        //
    }
    0 references | 0 overrides
    public function edit(Post $post)
    {
        //
    }
    0 references | 0 overrides
    public function update(UpdatePostRequest $request, Post $post)
    {
        //
    }
    0 references | 0 overrides
    public function destroy(Post $post)
    {
        //
    }
}
```

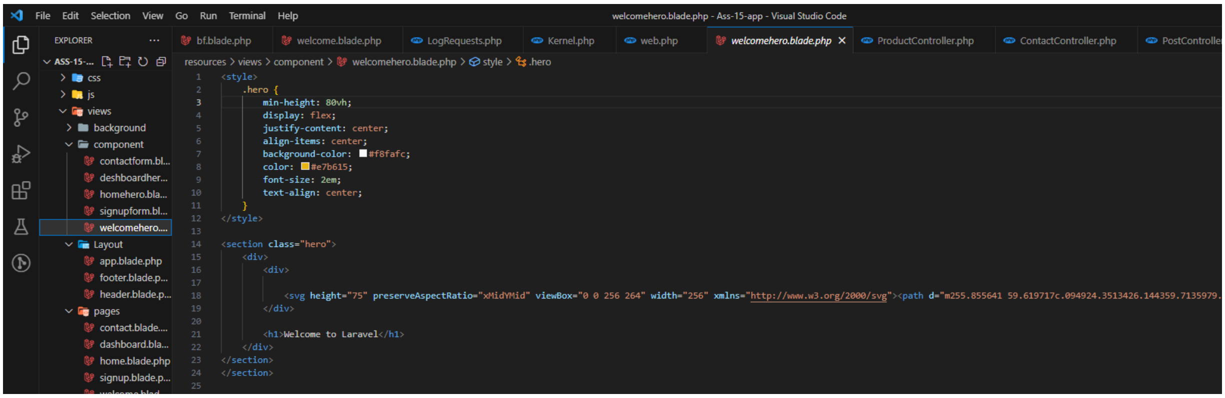
Task 8: Blade Template Engine

Create a Blade view called welcome.blade.php that includes a navigation bar and a section displaying the text "Welcome to Laravel!".

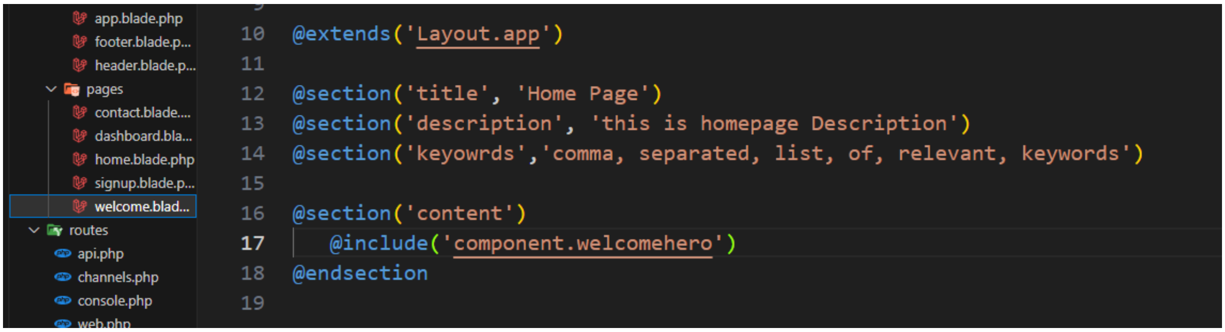
Answer



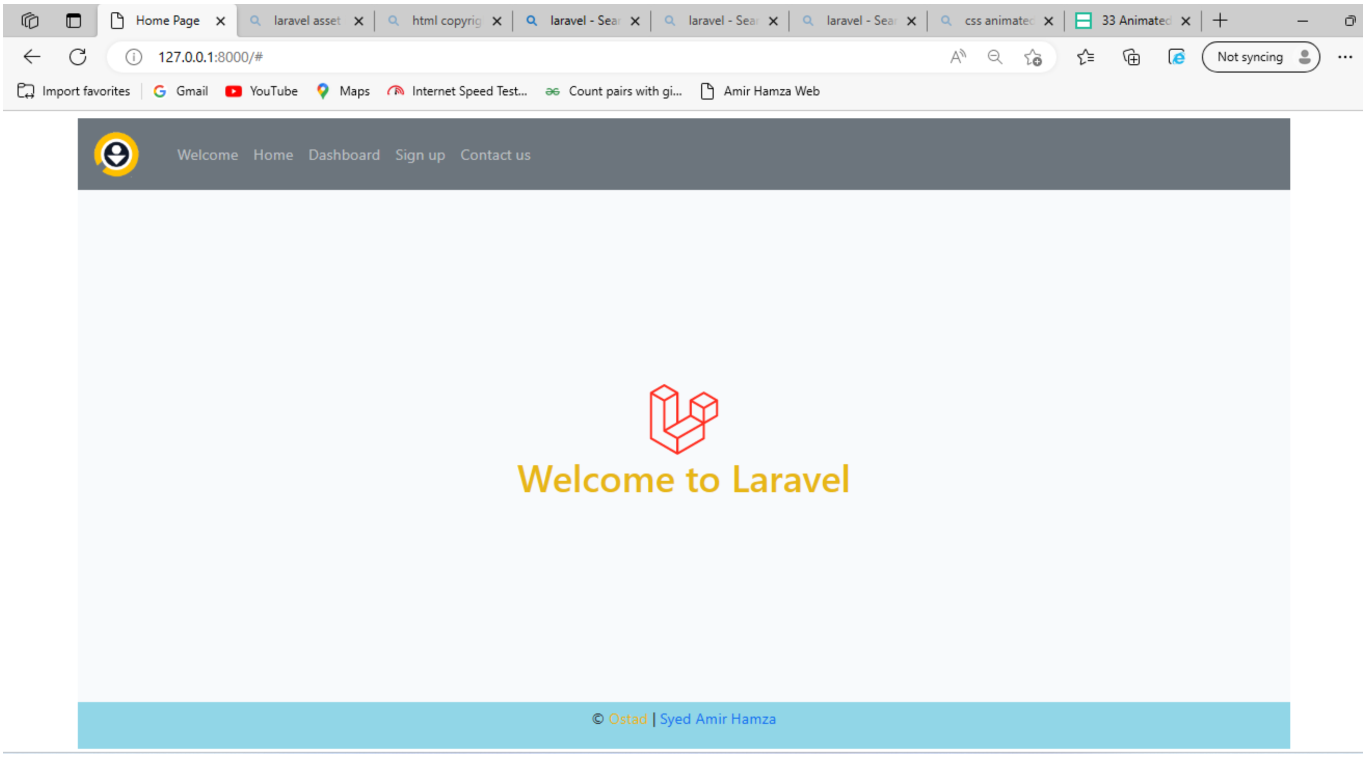
```
Route::get('/',function(){
    return view('pages.welcome');
});
```



```
<style>
    .hero {
        min-height: 80vh;
        display: flex;
        justify-content: center;
        align-items: center;
        background-color: #f8fafc;
        color: #e7b615;
        font-size: 2em;
        text-align: center;
    }
</style>

<section class="hero">
    <div>
        <div>

            <svg height="75" preserveAspectRatio="xMidYMid" viewBox="0 0 256 264" width="256" xmlns="http://www.w3.org/2000/svg"><path d="m255.855641 59.619717c.094924.3513426.144359.7135979.
        </div>

        <h1>Welcome to Laravel</h1>
    </div>
</section>
</section>
```



```
@extends('Layout.app')

@section('title', 'Home Page')
@section('description', 'this is homepage Description')
@section('keyowrds','comma, separated, list, of, relevant, keywords')

@section('content')
    @include('component.welcomehero')
@endsection
```



Welcome to Laravel

© Ostad | Syed Amir Hamza

THE END