php

Laravel

## Discussion Outline

- Middleware in Laravel
- Types of Middleware
- Connect to MYSQL Database
- Model in Laravel

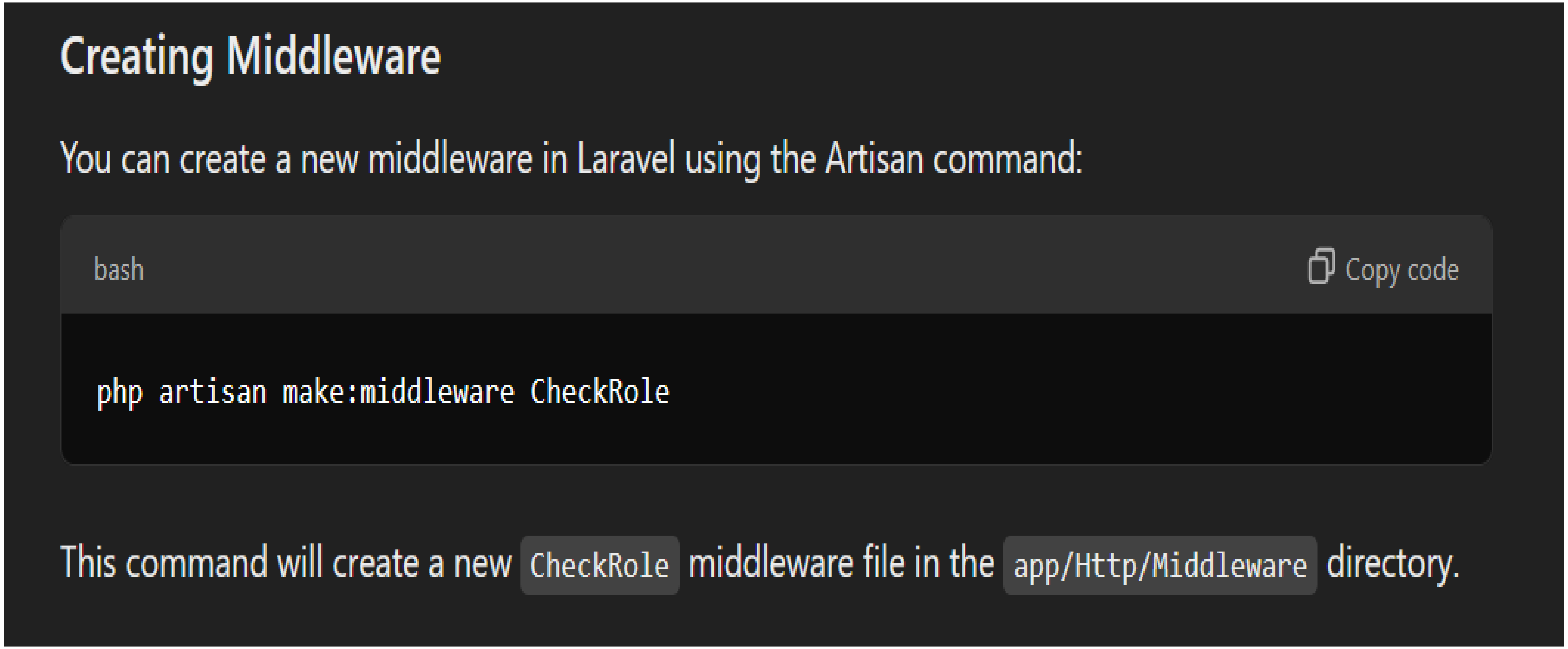
## Middleware in Laravel

- In Laravel, middleware is a layer between the **request** and the **application** that filters HTTP requests entering the application.
- Middleware can be used for tasks like **authentication**, **logging**, and **modifying** request data. It’s a powerful tool for **controlling access** to specific parts of your application and for pre- or post-processing requests.

## Common Uses for Middleware

- ❖**Authentication**: Ensuring only logged-in users can access certain routes.
- ❖**Logging**: Logging each request or tracking user activity.
- ❖**CORS**: Adding Cross-Origin Resource Sharing headers to the response.
- ❖**Data Sanitization**: Modifying request data before passing it to the controller.
- ❖**Maintenance Mode**: Redirecting all users to a maintenance page when the site is down.

## Creating Middleware



### Creating Middleware

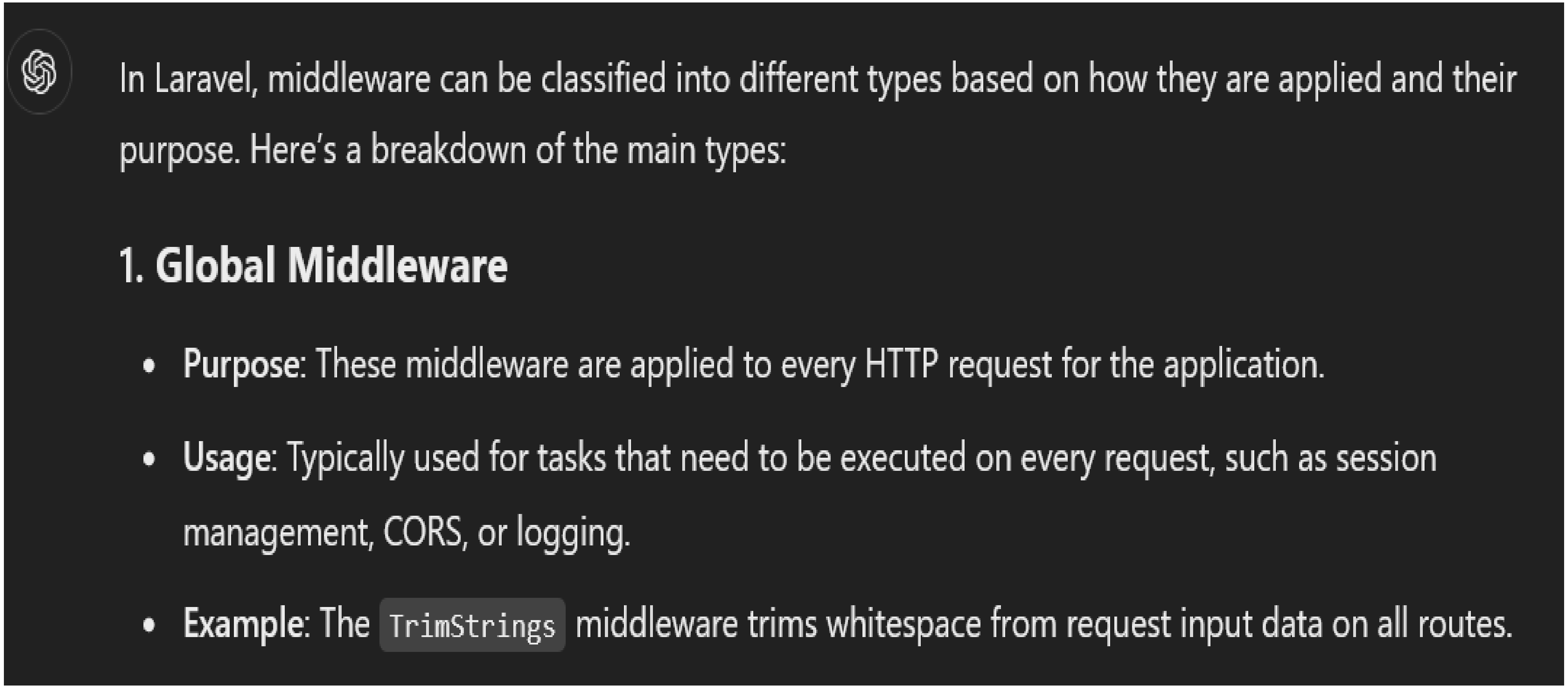
You can create a new middleware in Laravel using the Artisan command:

bash Copy code

```bash
php artisan make:middleware CheckRole
```

This command will create a new `CheckRole` middleware file in the `app/Http/Middleware` directory.

## Types of Middleware

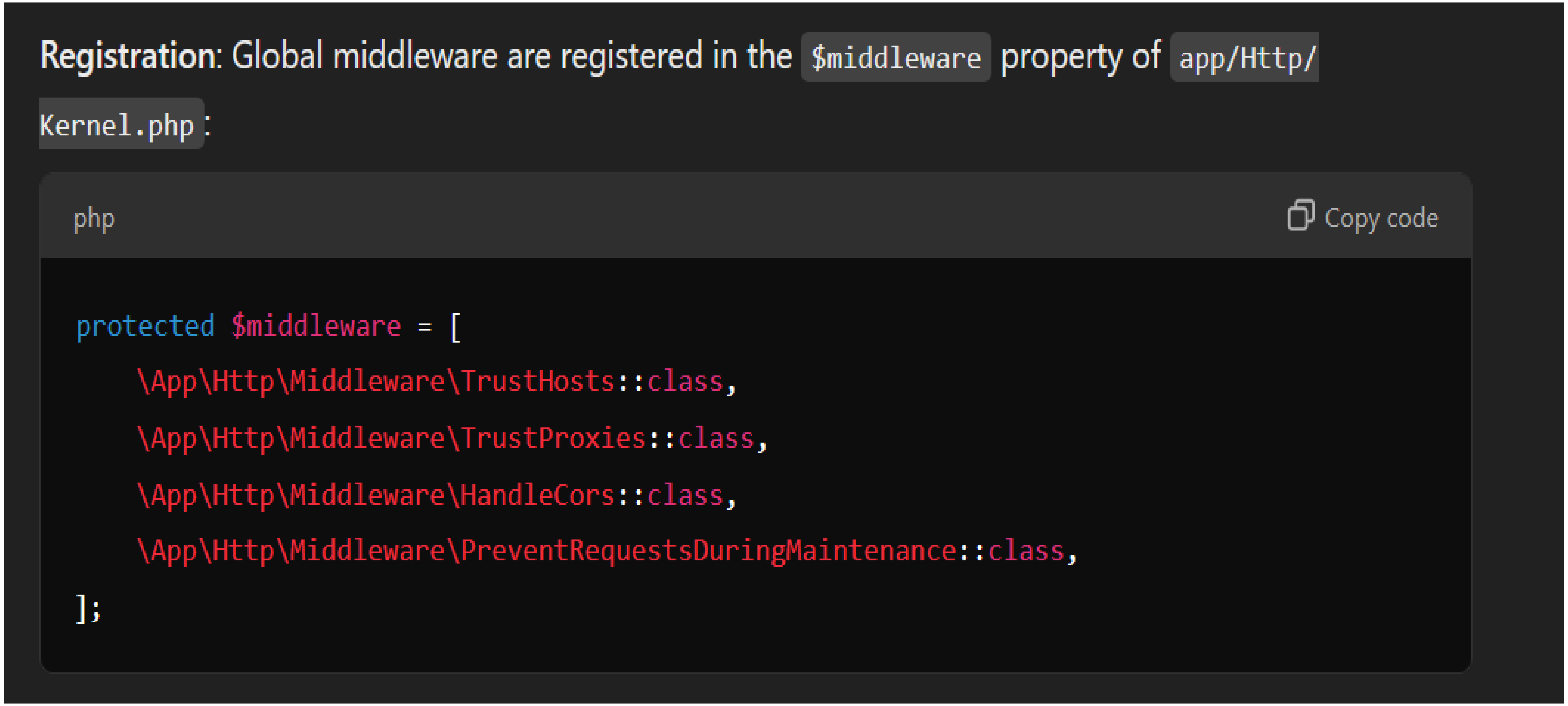


In Laravel, middleware can be classified into different types based on how they are applied and their purpose. Here's a breakdown of the main types:

### 1. Global Middleware

- **Purpose**: These middleware are applied to every HTTP request for the application.
- **Usage**: Typically used for tasks that need to be executed on every request, such as session management, CORS, or logging.
- **Example**: The `TrimStrings` middleware trims whitespace from request input data on all routes.

## Cont. ...

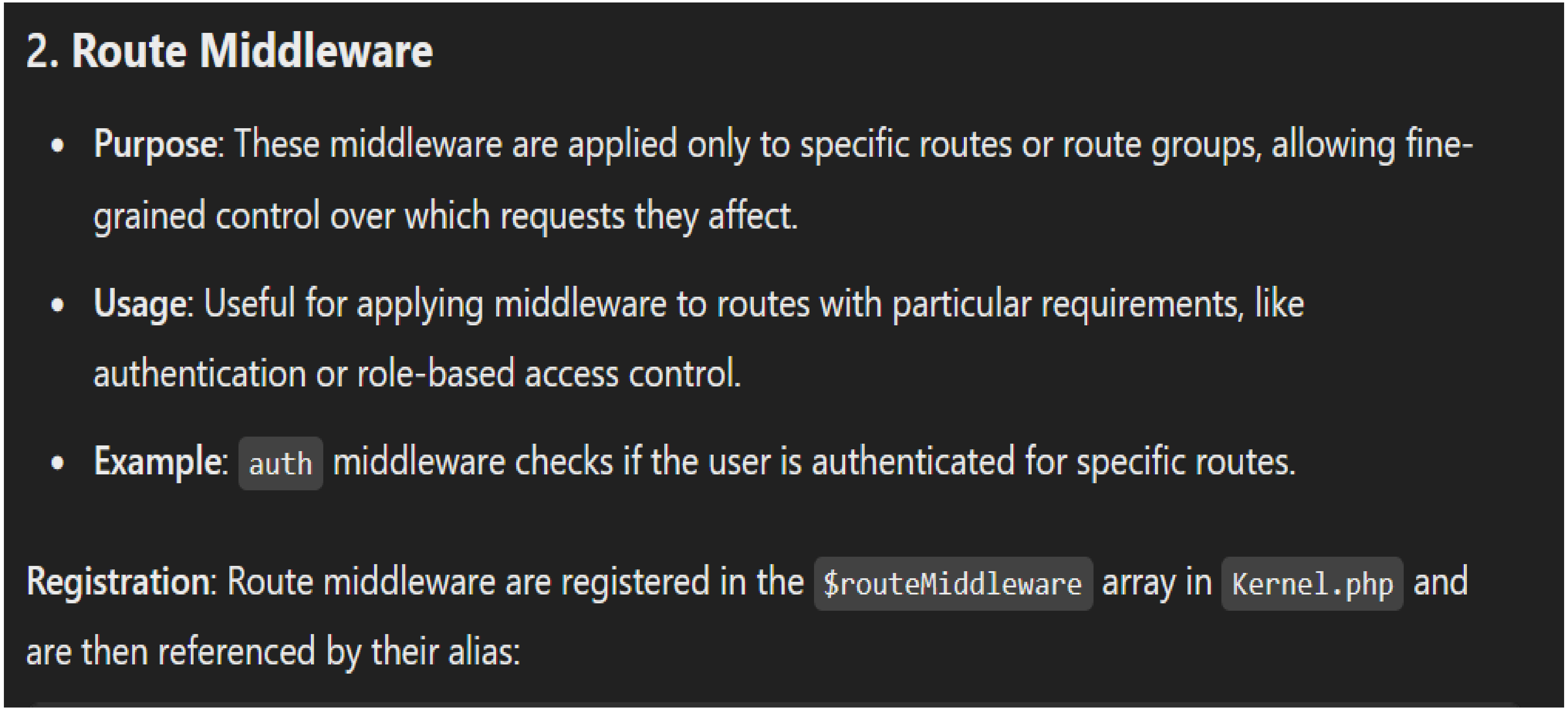


**Registration**: Global middleware are registered in the `$middleware` property of `app/Http/Kernel.php`:

php Copy code

```php
protected $middleware = [
    \App\Http\Middleware\TrustHosts::class,
    \App\Http\Middleware\TrustProxies::class,
    \App\Http\Middleware\HandleCors::class,
    \App\Http\Middleware\PreventRequestsDuringMaintenance::class,
];
```

## Cont. …

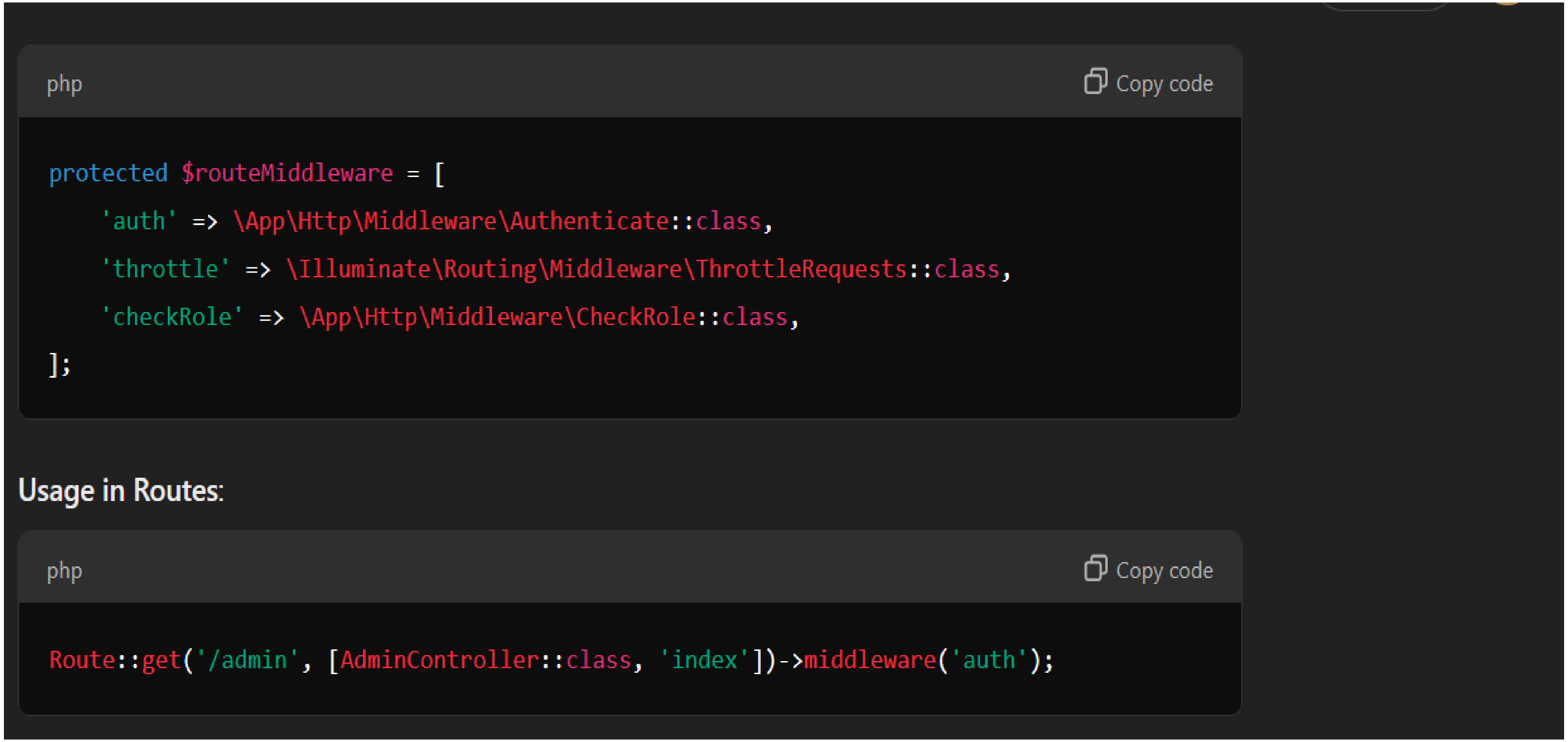


### 2. Route Middleware

- **Purpose**: These middleware are applied only to specific routes or route groups, allowing fine-grained control over which requests they affect.
- **Usage**: Useful for applying middleware to routes with particular requirements, like authentication or role-based access control.
- **Example**: `auth` middleware checks if the user is authenticated for specific routes.

**Registration**: Route middleware are registered in the `$routeMiddleware` array in `Kernel.php` and are then referenced by their alias:

## Cont. …


php

Copy code

```php
protected $routeMiddleware = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
    'checkRole' => \App\Http\Middleware\CheckRole::class,
];
```

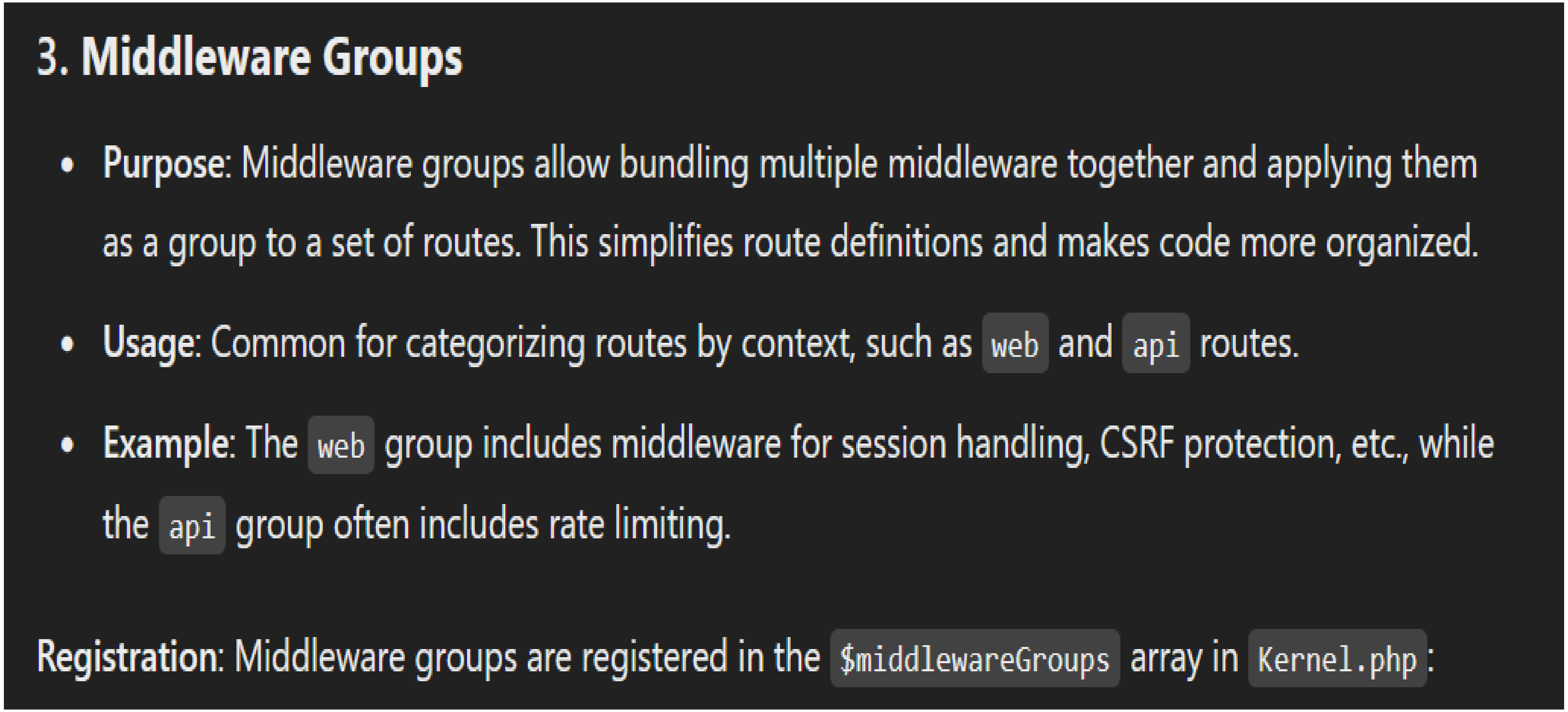
Usage in Routes:

php

Copy code

```php
Route::get('/admin', [AdminController::class, 'index'])->middleware('auth');
```

# Cont. …

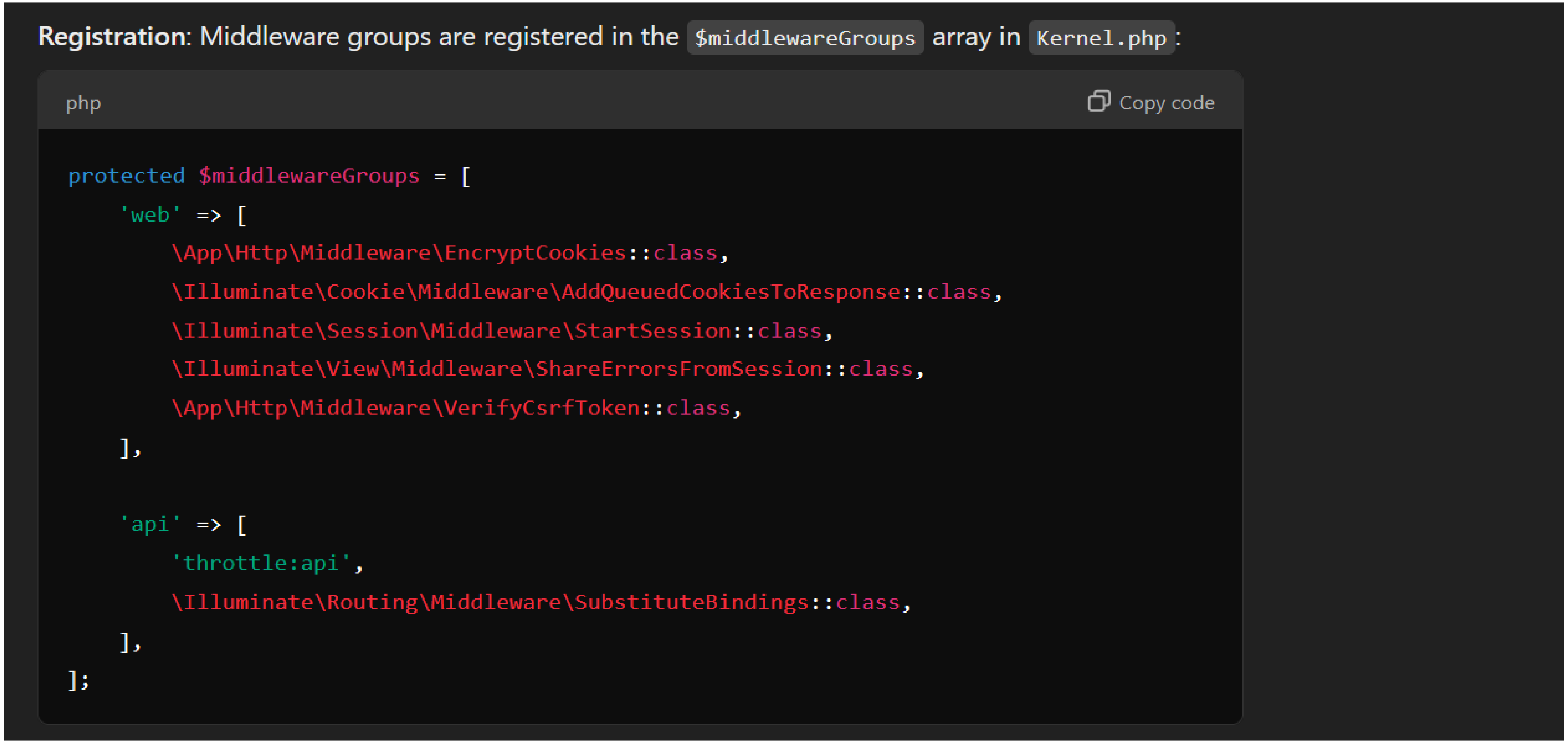


### 3. Middleware Groups

- **Purpose**: Middleware groups allow bundling multiple middleware together and applying them as a group to a set of routes. This simplifies route definitions and makes code more organized.
- **Usage**: Common for categorizing routes by context, such as `web` and `api` routes.
- **Example**: The `web` group includes middleware for session handling, CSRF protection, etc., while the `api` group often includes rate limiting.

**Registration**: Middleware groups are registered in the `$middlewareGroups` array in `Kernel.php`:

## Cont. ...



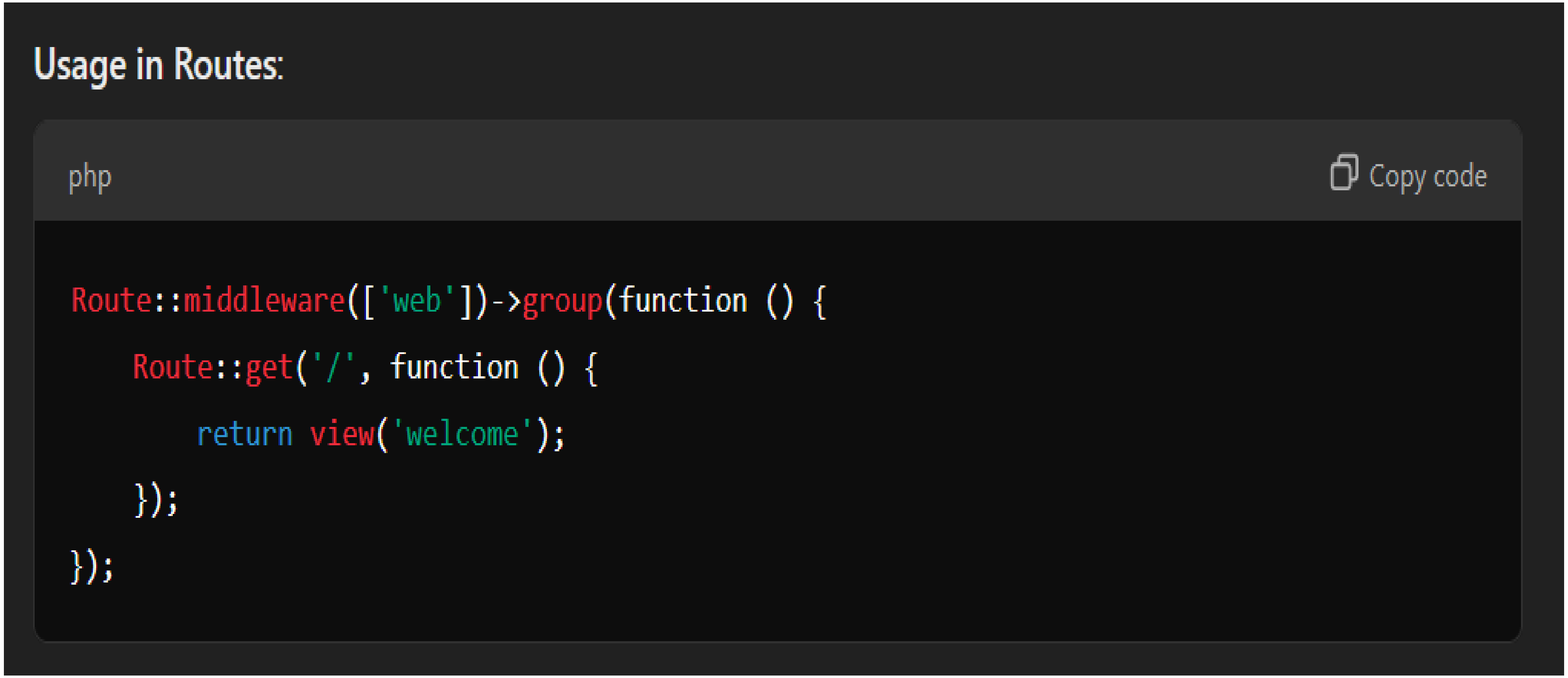
**Registration**: Middleware groups are registered in the `$middlewareGroups` array in `Kernel.php`:

php

Copy code

```php
protected $middlewareGroups = [
    'web' => [
        \App\Http\Middleware\EncryptCookies::class,
        \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
        \Illuminate\Session\Middleware\StartSession::class,
        \Illuminate\View\Middleware\ShareErrorsFromSession::class,
        \App\Http\Middleware\VerifyCsrfToken::class,
    ],

    'api' => [
        'throttle:api',
        \Illuminate\Routing\Middleware\SubstituteBindings::class,
    ],
];
```

## Cont. ...

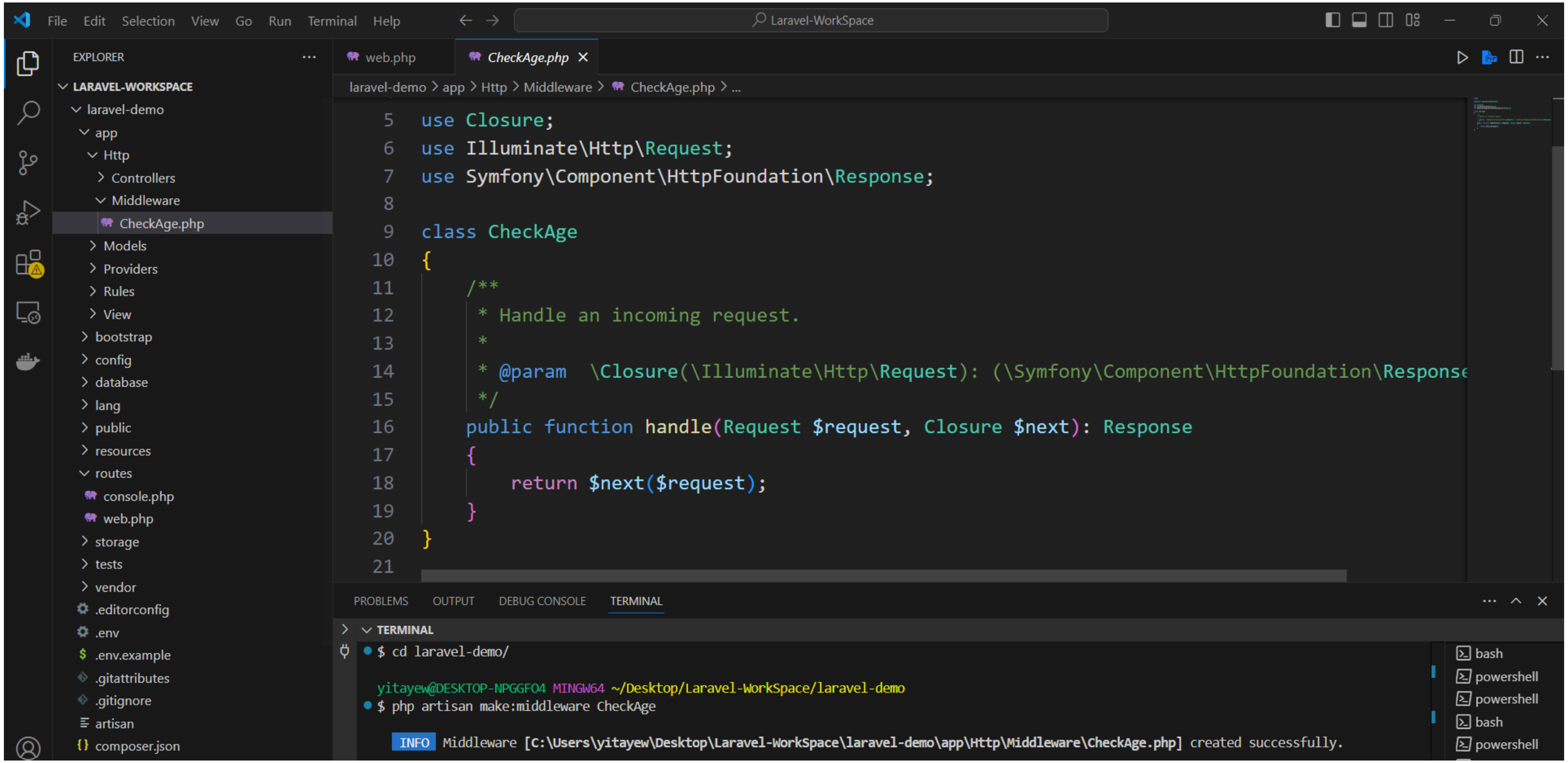

Usage in Routes:

php

Copy code

```php
Route::middleware(['web'])->group(function () {
    Route::get('/', function () {
        return view('welcome');
    });
});
```

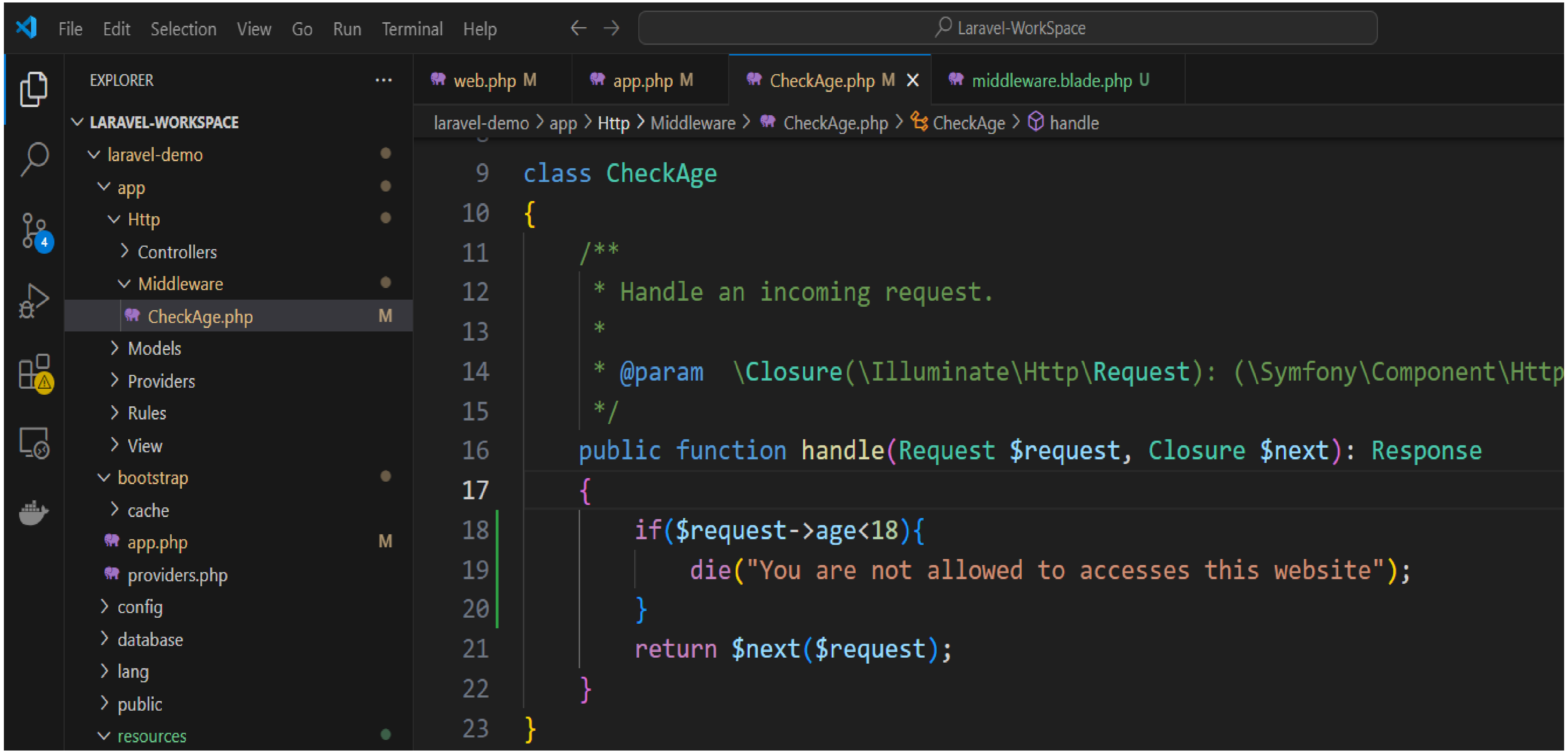
# Example (Global Route)



```php
use Closure;
use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;

class CheckAge
{
    /**
     * Handle an incoming request.
     *
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response
     */
    public function handle(Request $request, Closure $next): Response
    {
        return $next($request);
    }
}

```

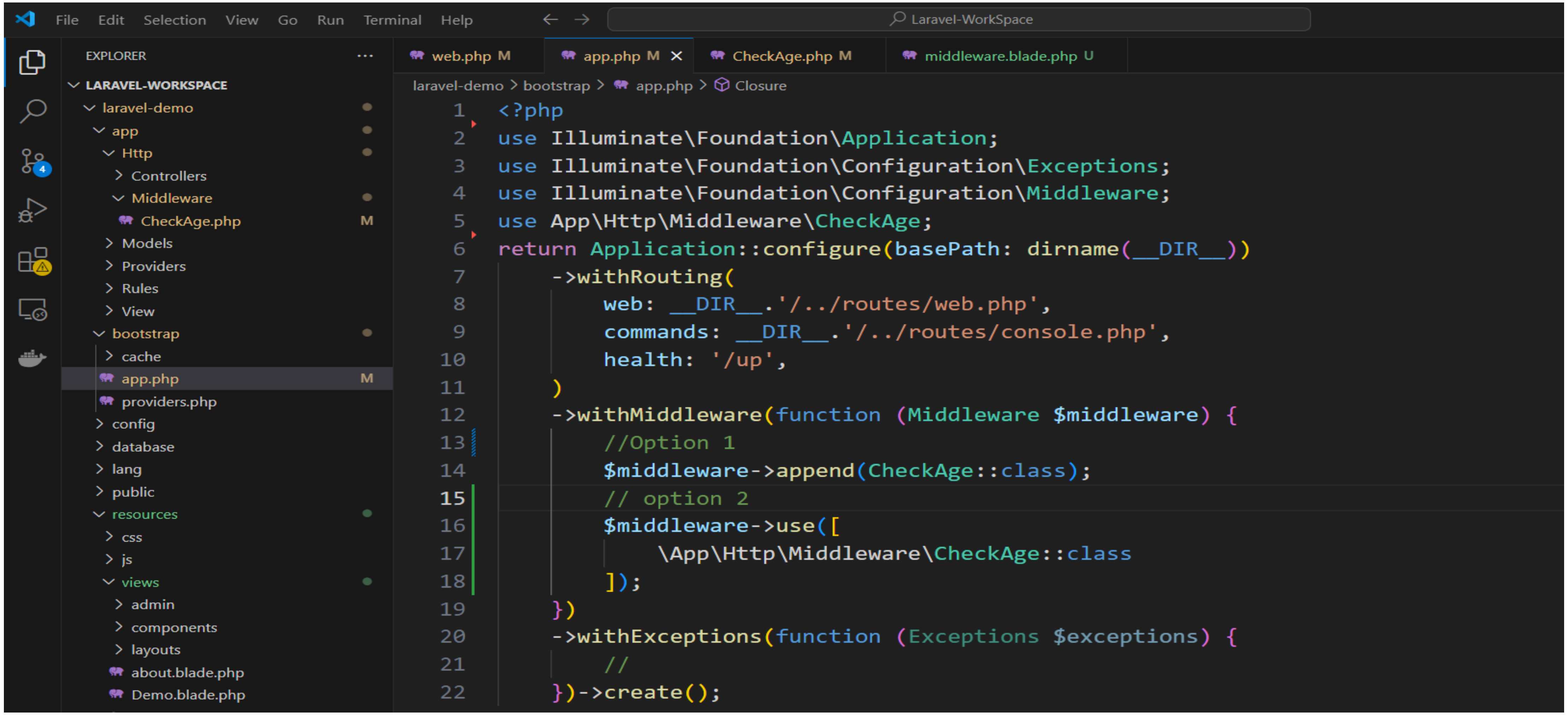
```
$ cd laravel-demo/

yitayew@DESKTOP-NPGGF04 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo
$ php artisan make:middleware CheckAge

  INFO  Middleware [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\app\Http\Middleware\CheckAge.php] created successfully.
```

# Cont. ...



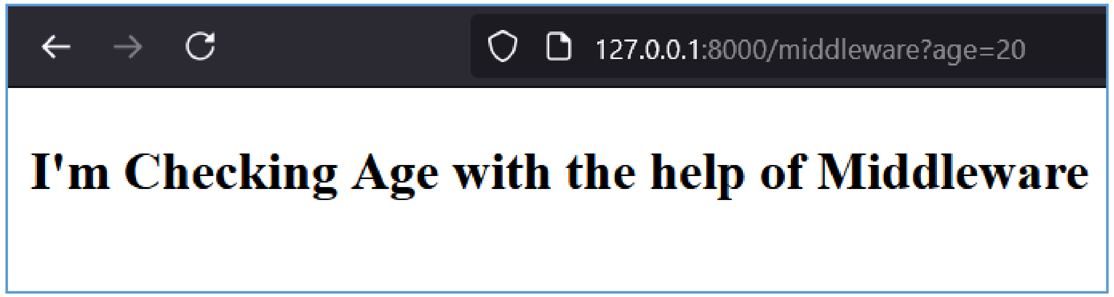
```php
class CheckAge
{
    /**
     * Handle an incoming request.
     *
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\Http
     */
    public function handle(Request $request, Closure $next): Response
    {
        if($request->age<18){
            die("You are not allowed to accesses this website");
        }
        return $next($request);
    }
}
```

# Cont. …

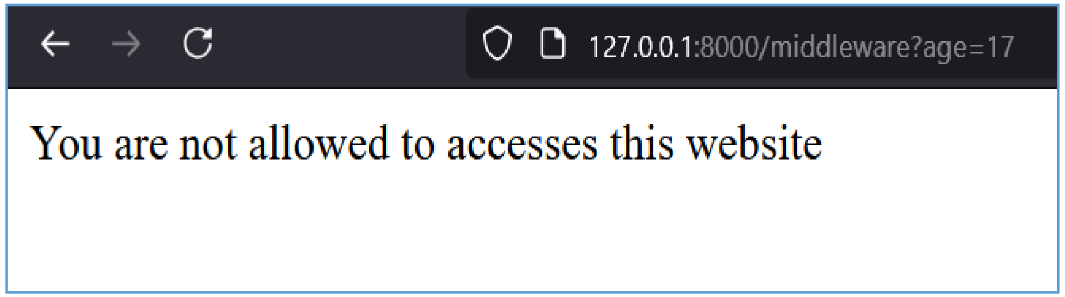


```php
<?php
use Illuminate\Foundation\Application;
use Illuminate\Foundation\Configuration\Exceptions;
use Illuminate\Foundation\Configuration\Middleware;
use App\Http\Middleware\CheckAge;
return Application::configure(basePath: dirname(__DIR__))
    ->withRouting(
        web: __DIR__.'/../routes/web.php',
        commands: __DIR__.'/../routes/console.php',
        health: '/up',
    )
    ->withMiddleware(function (Middleware $middleware) {
        //Option 1
        $middleware->append(CheckAge::class);
        // option 2
        $middleware->use([
            \App\Http\Middleware\CheckAge::class
        ]);
    })
    ->withExceptions(function (Exceptions $exceptions) {
        //
    })->create();
```

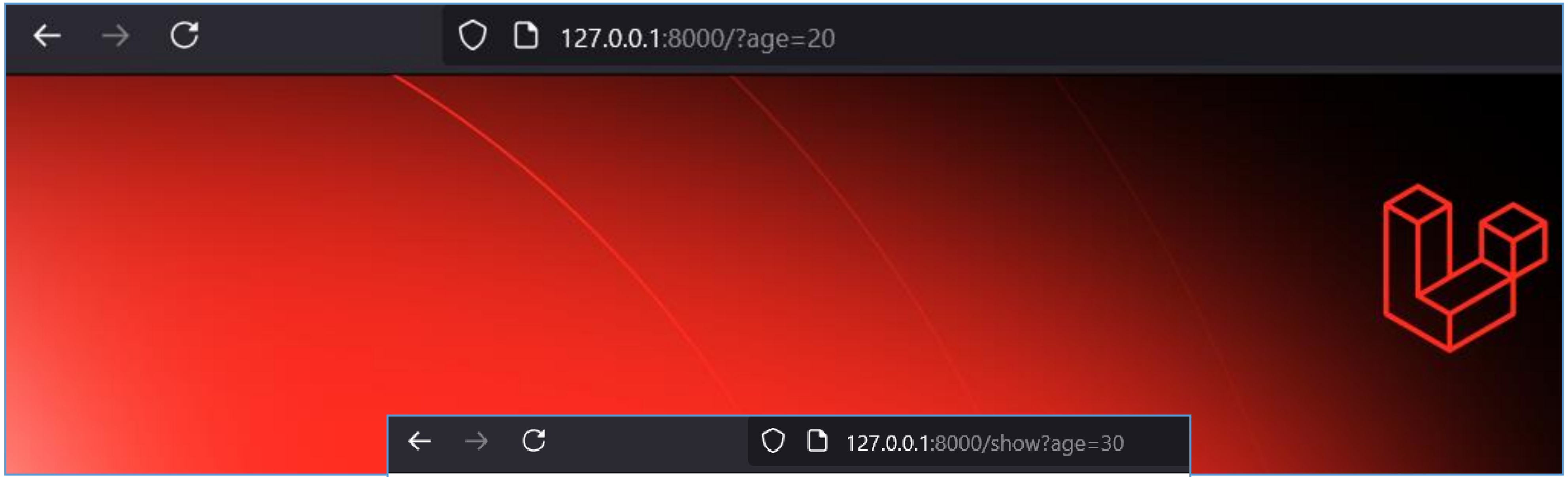
## Cont. ...

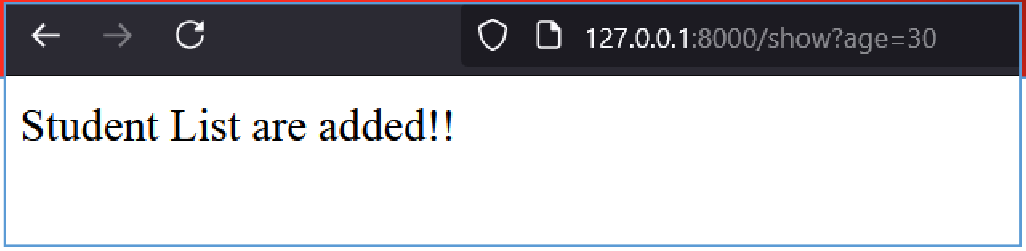

127.0.0.1:8000/middleware?age=20

I'm Checking Age with the help of Middleware


127.0.0.1:8000/middleware?age=17

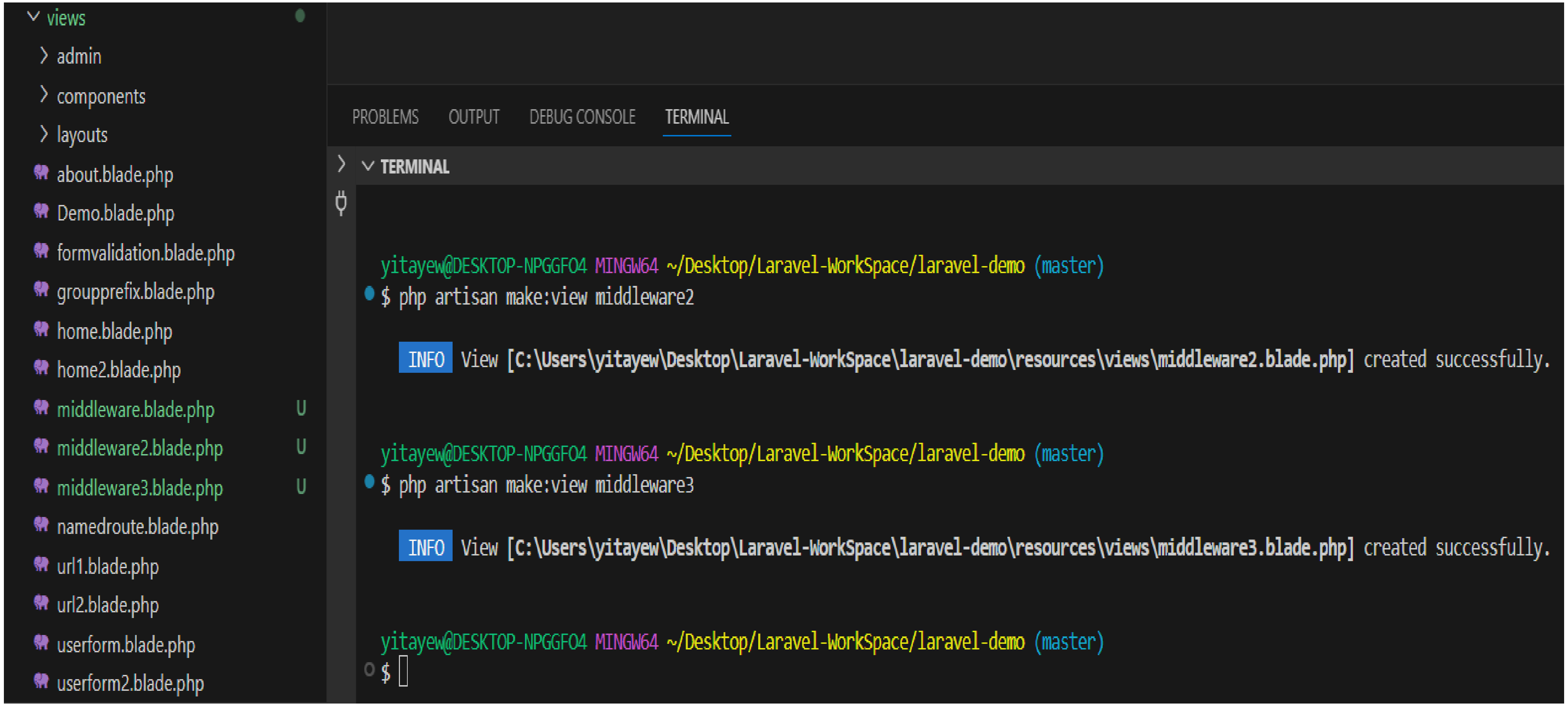
You are not allowed to accesses this website


127.0.0.1:8000/?age=20


127.0.0.1:8000/show?age=30

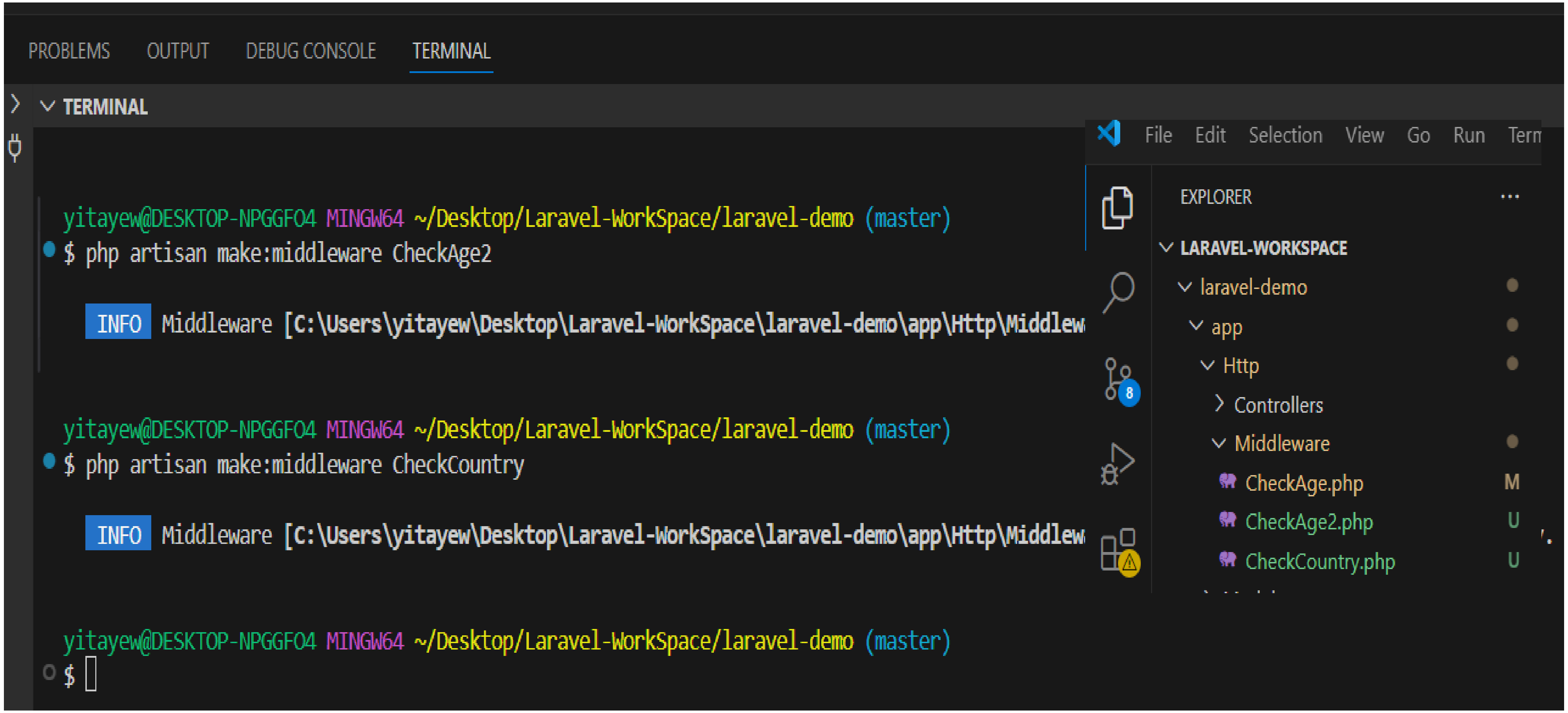
Student List are added!!

# Example (Middleware Group) Apply on Single Route



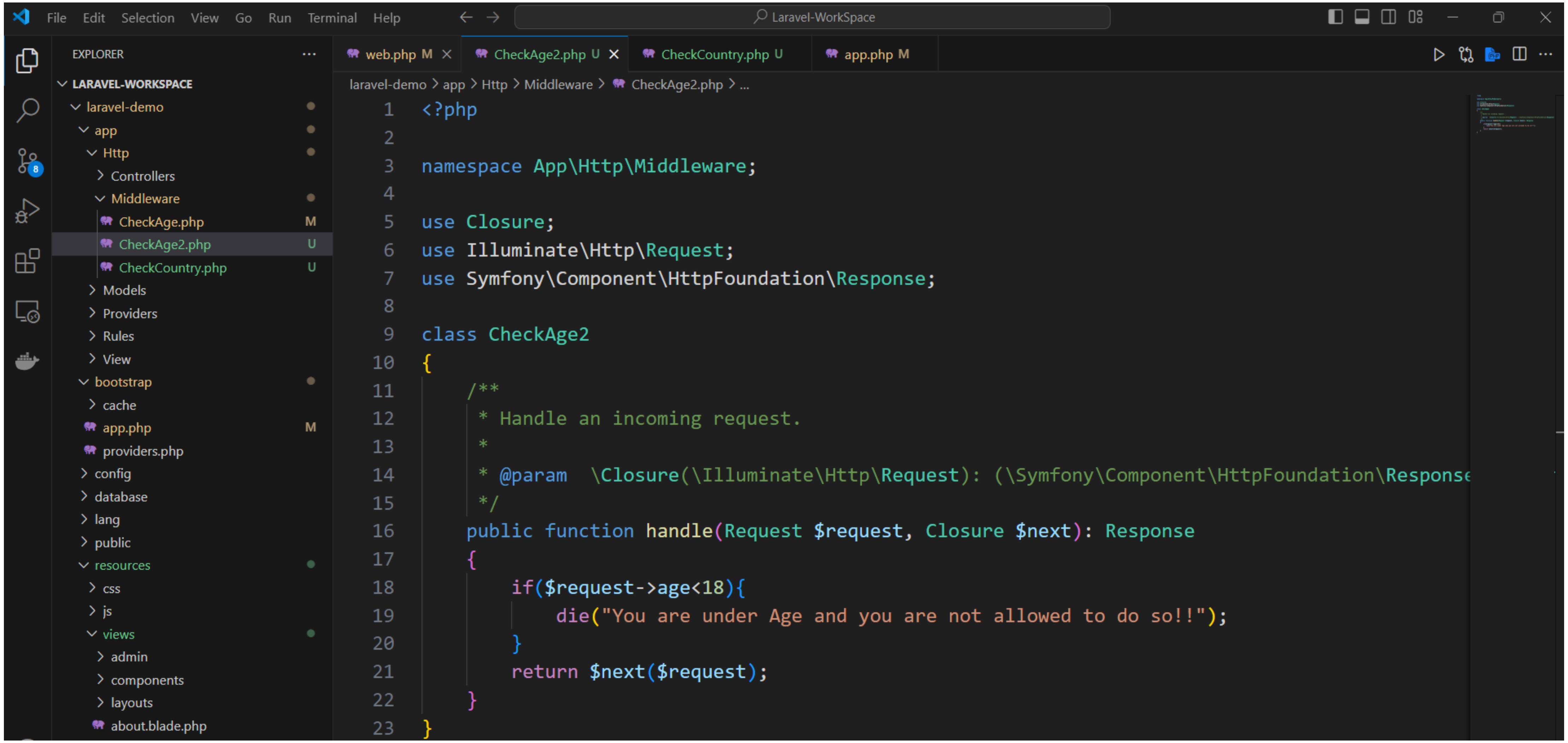
```
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view middleware2

  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\middleware2.blade.php] created successfully.

yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view middleware3

  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\middleware3.blade.php] created successfully.

yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$
```

## Cont. ...



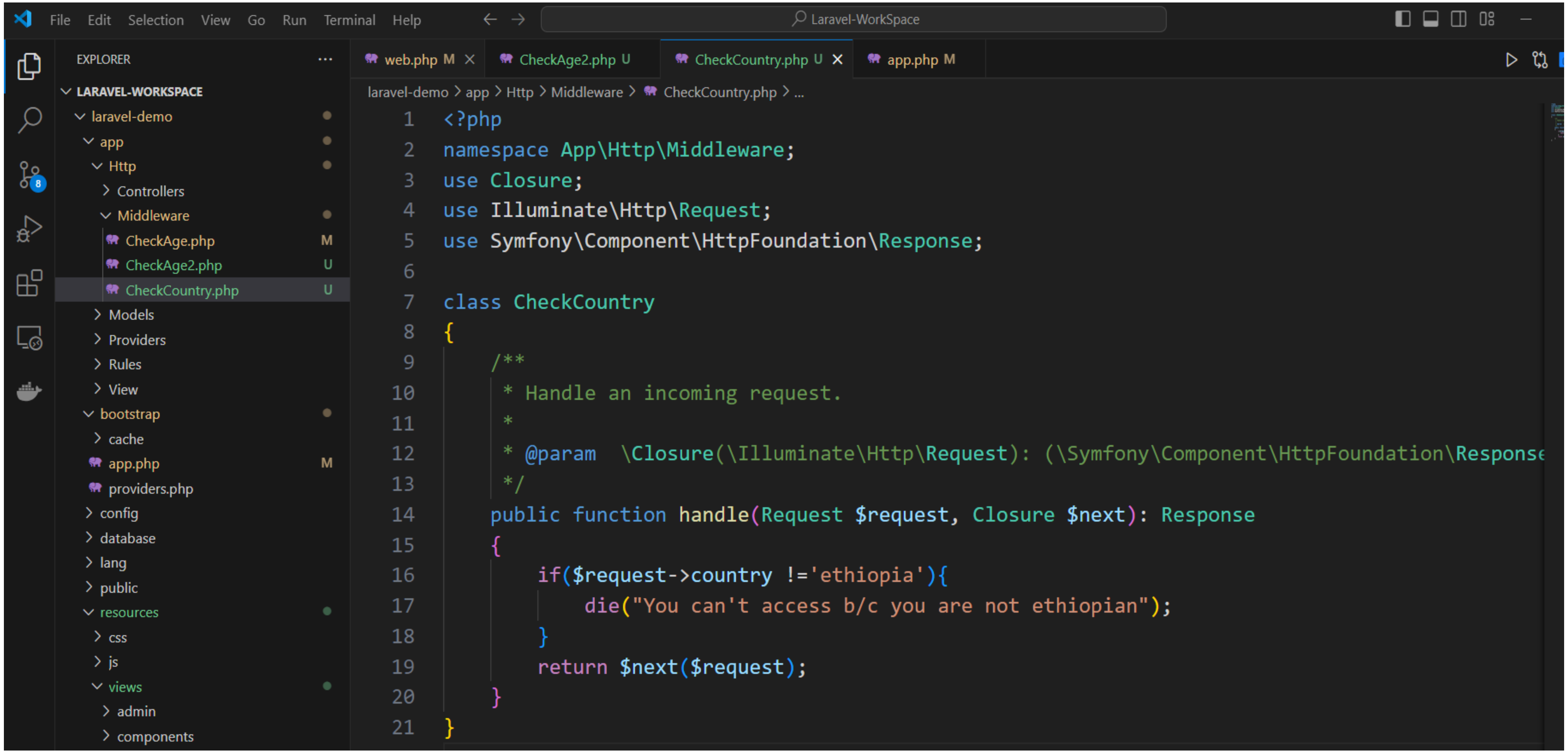
```
yitayew@DESKTOP-NPGGF04 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:middleware CheckAge2

   INFO  Middleware [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\app\Http\Middlew

yitayew@DESKTOP-NPGGF04 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:middleware CheckCountry

   INFO  Middleware [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\app\Http\Middlew

yitayew@DESKTOP-NPGGF04 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ 
```

# Cont. …



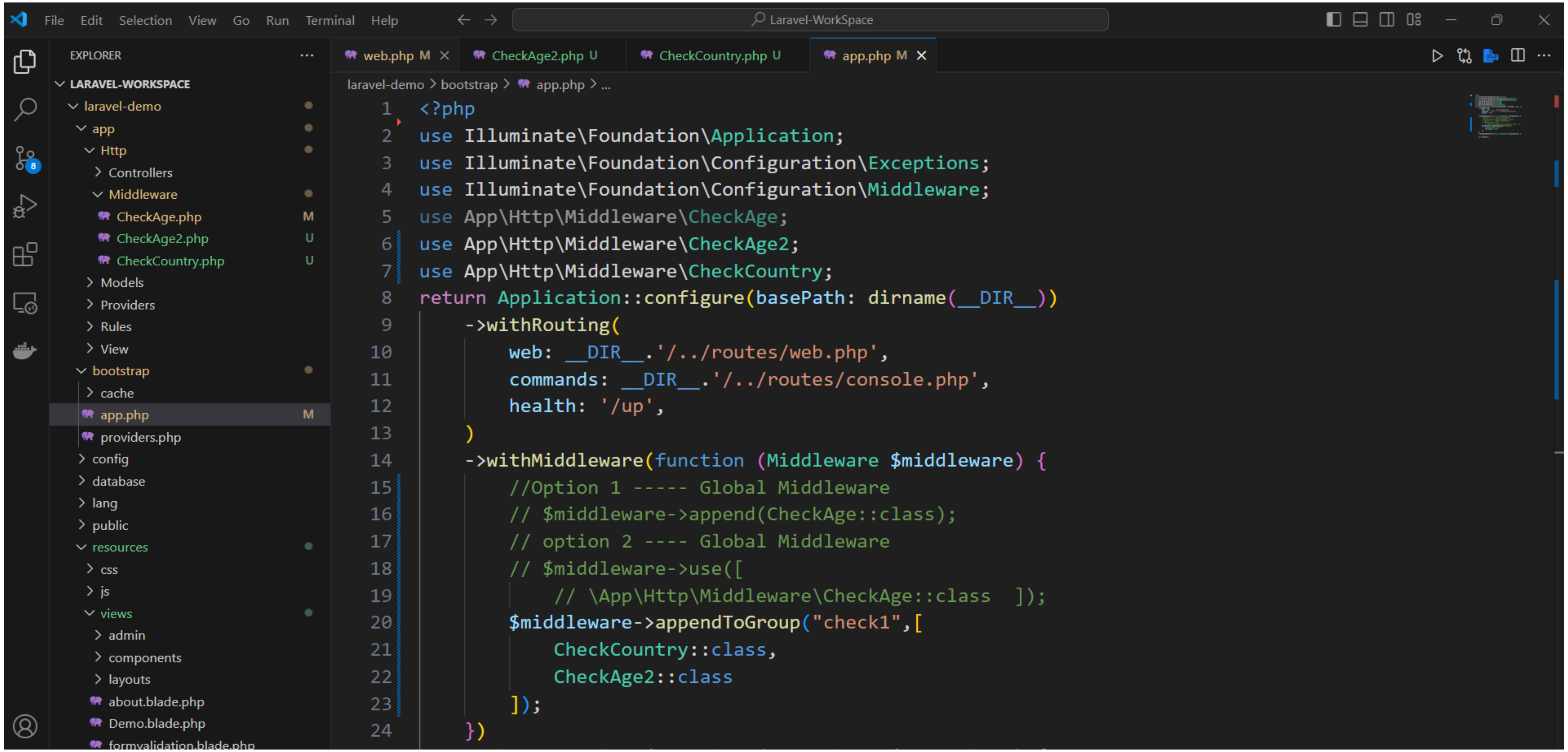
```php
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;

class CheckAge2
{
    /**
     * Handle an incoming request.
     *
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response
     */
    public function handle(Request $request, Closure $next): Response
    {
        if($request->age<18){
            die("You are under Age and you are not allowed to do so!!");
        }
        return $next($request);
    }
}
```

# Cont. ...



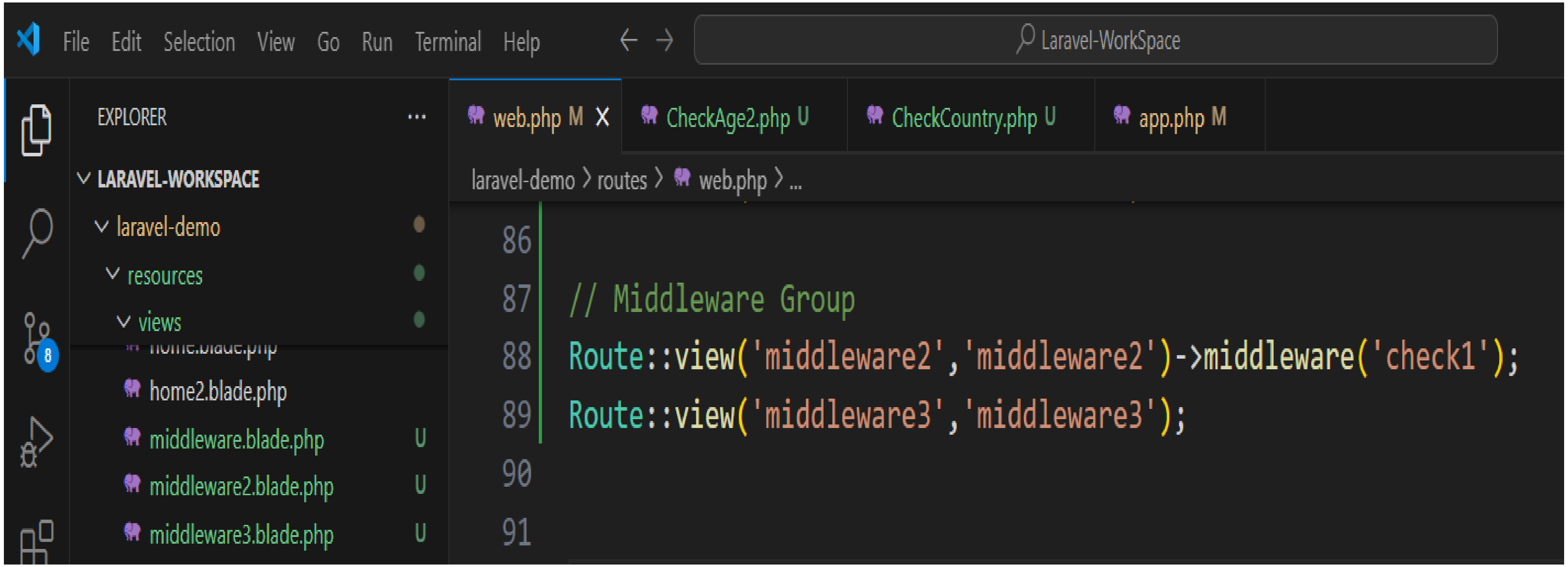
```php
<?php
namespace App\Http\Middleware;
use Closure;
use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;

class CheckCountry
{
    /**
     * Handle an incoming request.
     *
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response
     */
    public function handle(Request $request, Closure $next): Response
    {
        if($request->country !='ethiopia'){
            die("You can't access b/c you are not ethiopian");
        }
        return $next($request);
    }
}
```

# Count. …



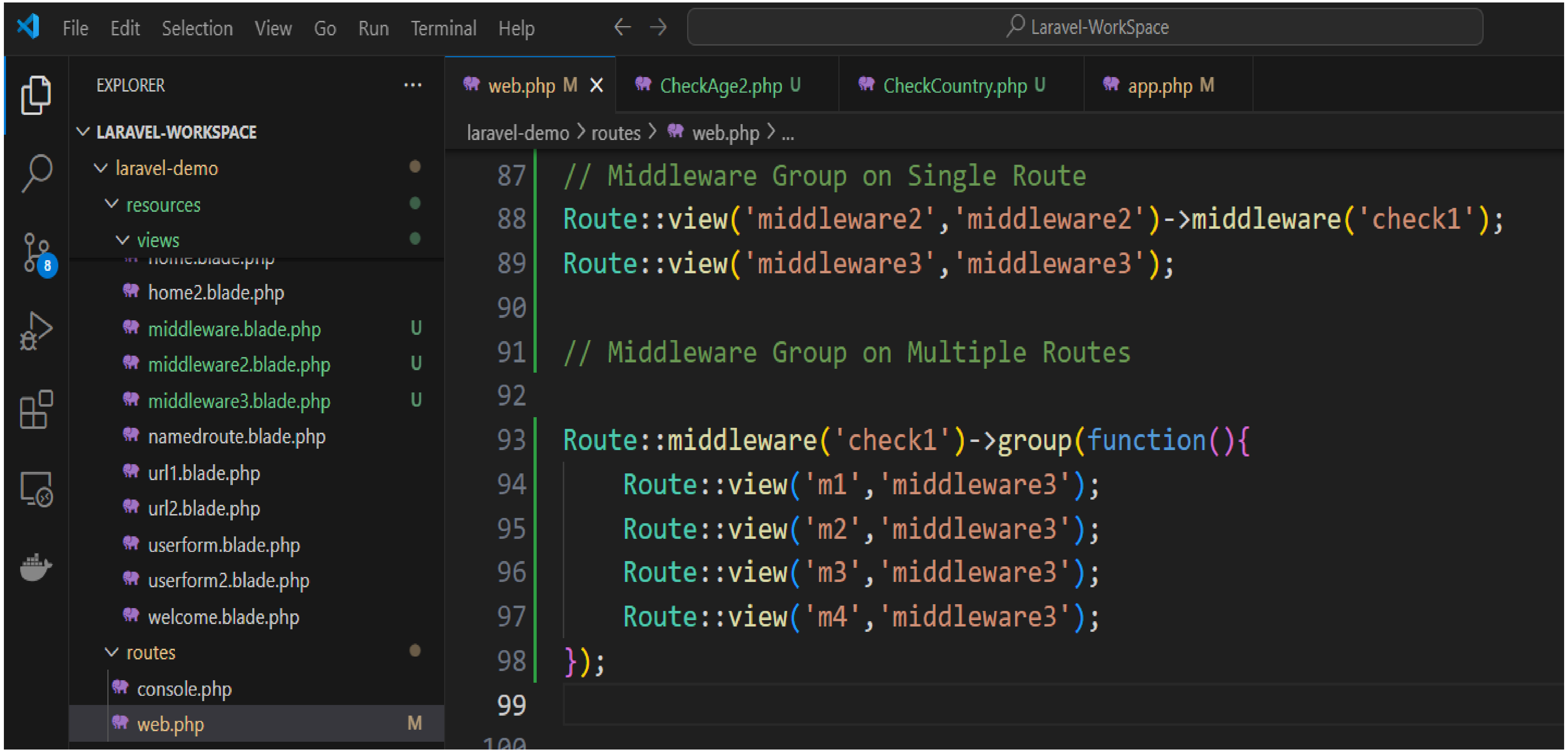
```php
<?php
use Illuminate\Foundation\Application;
use Illuminate\Foundation\Configuration\Exceptions;
use Illuminate\Foundation\Configuration\Middleware;
use App\Http\Middleware\CheckAge;
use App\Http\Middleware\CheckAge2;
use App\Http\Middleware\CheckCountry;
return Application::configure(basePath: dirname(__DIR__))
    ->withRouting(
        web: __DIR__.'/../routes/web.php',
        commands: __DIR__.'/../routes/console.php',
        health: '/up',
    )
    ->withMiddleware(function (Middleware $middleware) {
        //Option 1 ----- Global Middleware
        // $middleware->append(CheckAge::class);
        // option 2 ---- Global Middleware
        // $middleware->use([
            // \App\Http\Middleware\CheckAge::class  ]);
        $middleware->appendToGroup("check1",[
            CheckCountry::class,
            CheckAge2::class
        ]);
    })
```

## Cont. …



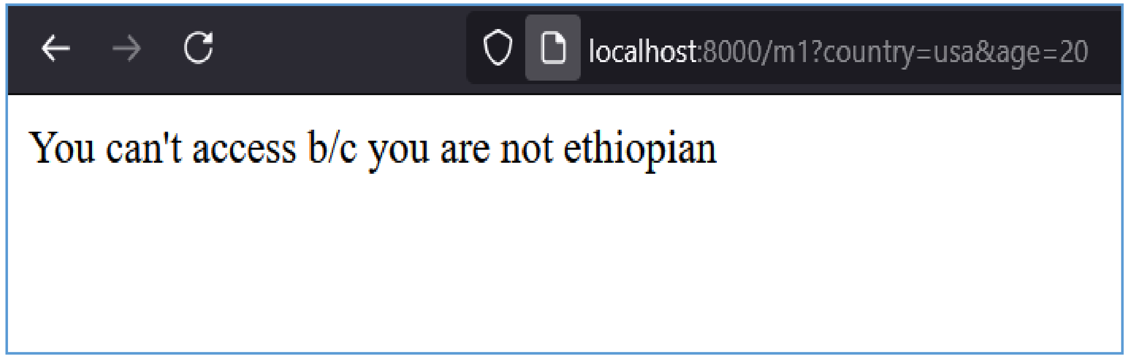
```php

// Middleware Group
Route::view('middleware2','middleware2')->middleware('check1');
Route::view('middleware3','middleware3');


```

# For Multiple Route



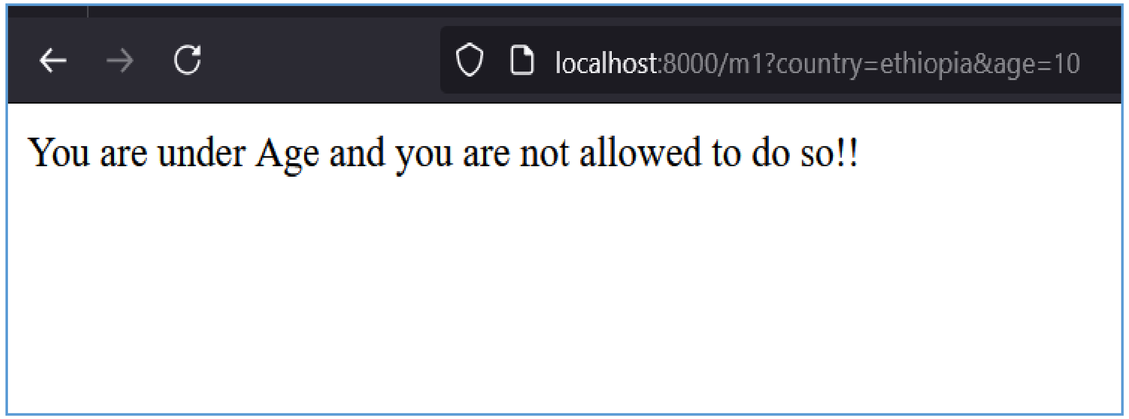
```php
// Middleware Group on Single Route
Route::view('middleware2','middleware2')->middleware('check1');
Route::view('middleware3','middleware3');

// Middleware Group on Multiple Routes

Route::middleware('check1')->group(function(){
    Route::view('m1','middleware3');
    Route::view('m2','middleware3');
    Route::view('m3','middleware3');
    Route::view('m4','middleware3');
});
```

## Cont. ...



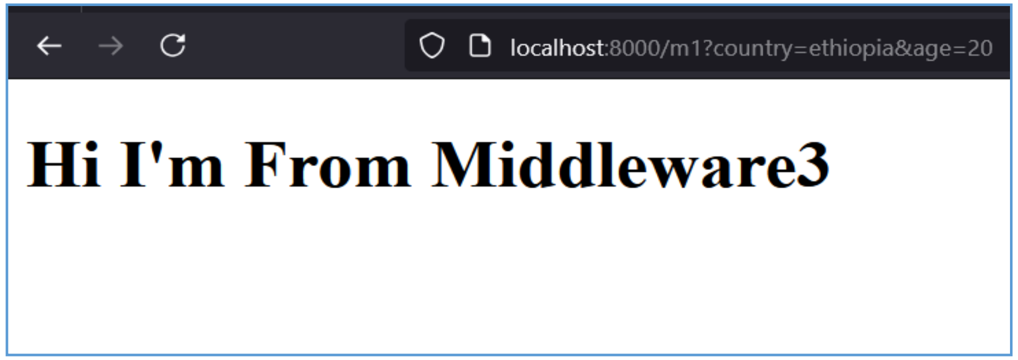
```
localhost:8000/m1?country=usa&age=20
```

You can't access b/c you are not ethiopian



```
localhost:8000/m1?country=ethiopia&age=10
```

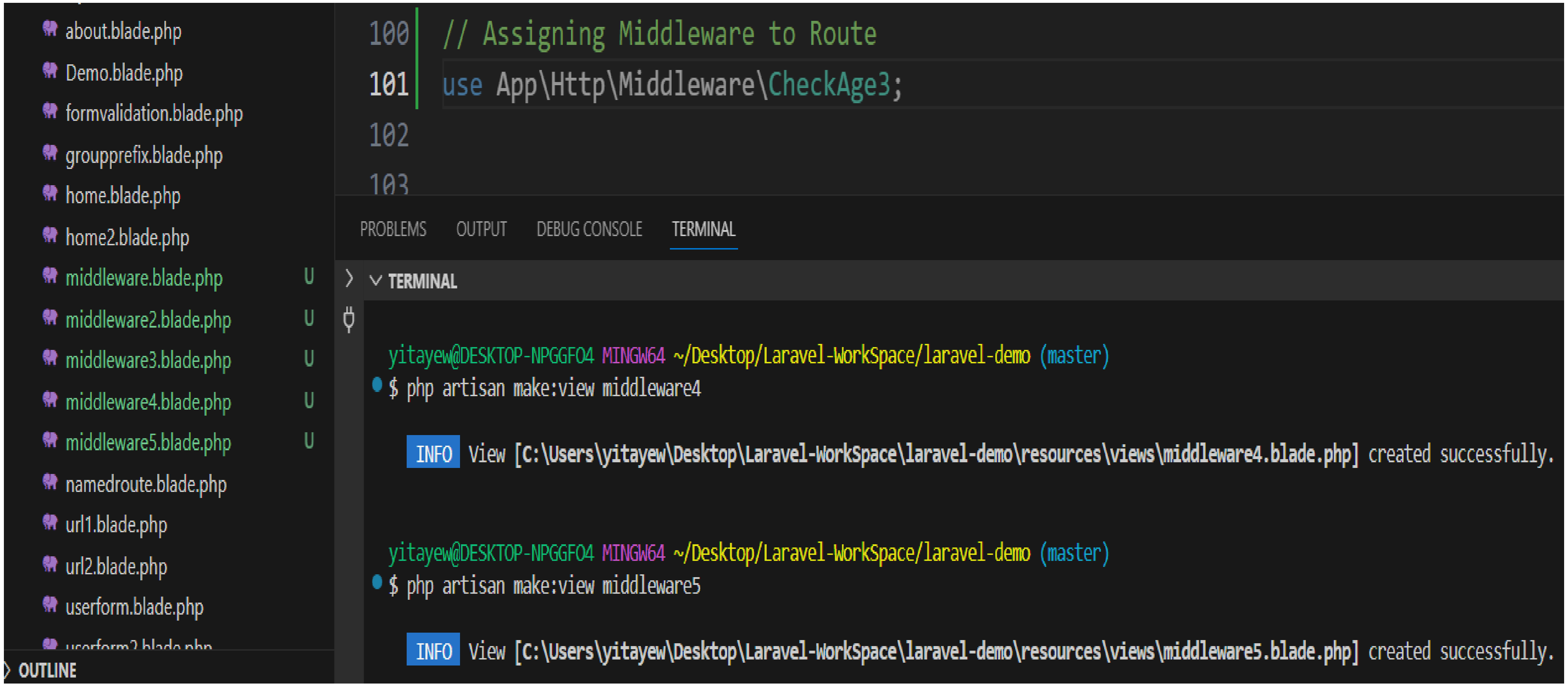
You are under Age and you are not allowed to do so!!



```
localhost:8000/m1?country=ethiopia&age=20
```

**Hi I'm From Middleware3**

# Assigning Middleware to Route

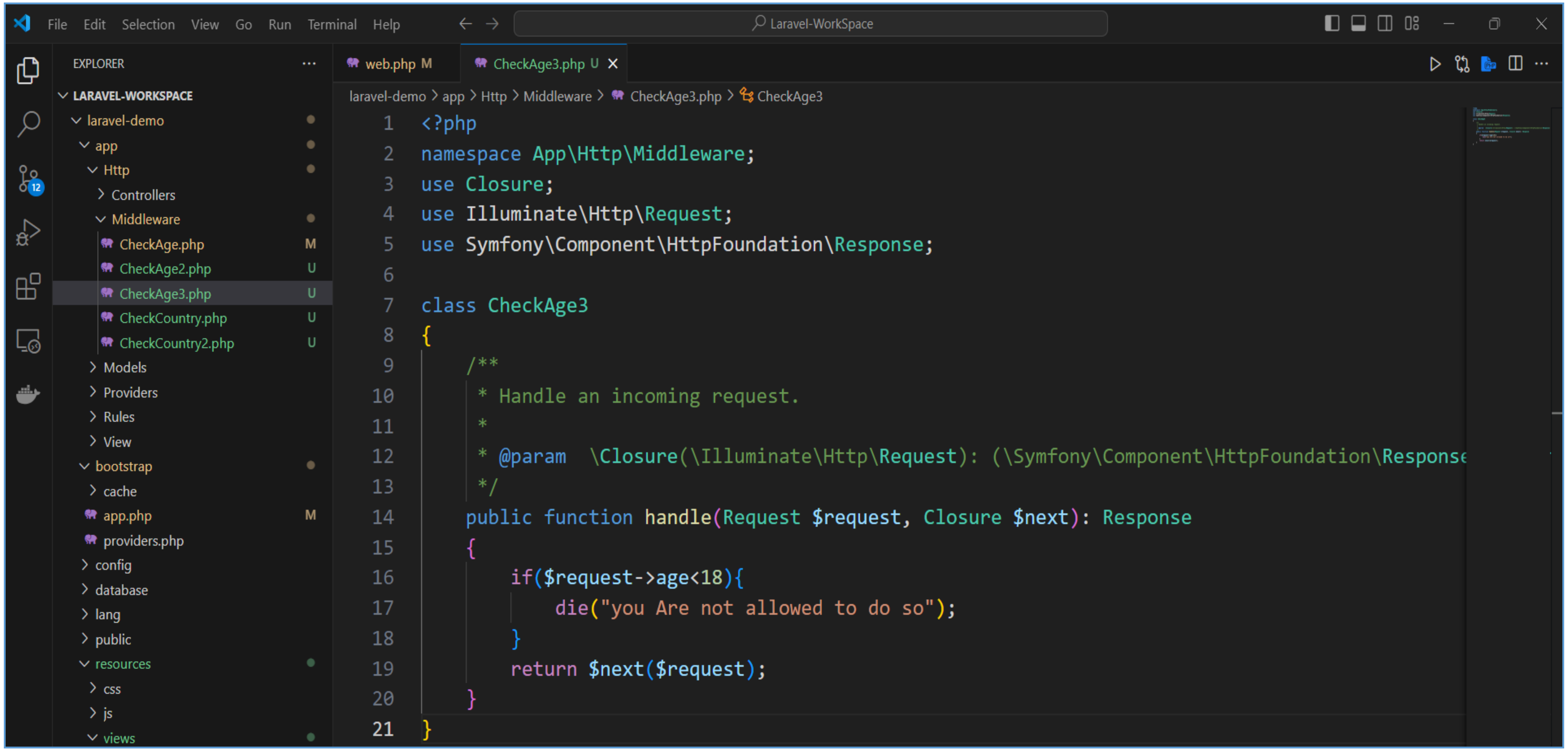


about.blade.php
Demo.blade.php
formvalidation.blade.php
groupprefix.blade.php
home.blade.php
home2.blade.php
middleware.blade.php U
middleware2.blade.php U
middleware3.blade.php U
middleware4.blade.php U
middleware5.blade.php U
namedroute.blade.php
url1.blade.php
url2.blade.php
userform.blade.php

```php
// Assigning Middleware to Route
use App\Http\Middleware\CheckAge3;
```

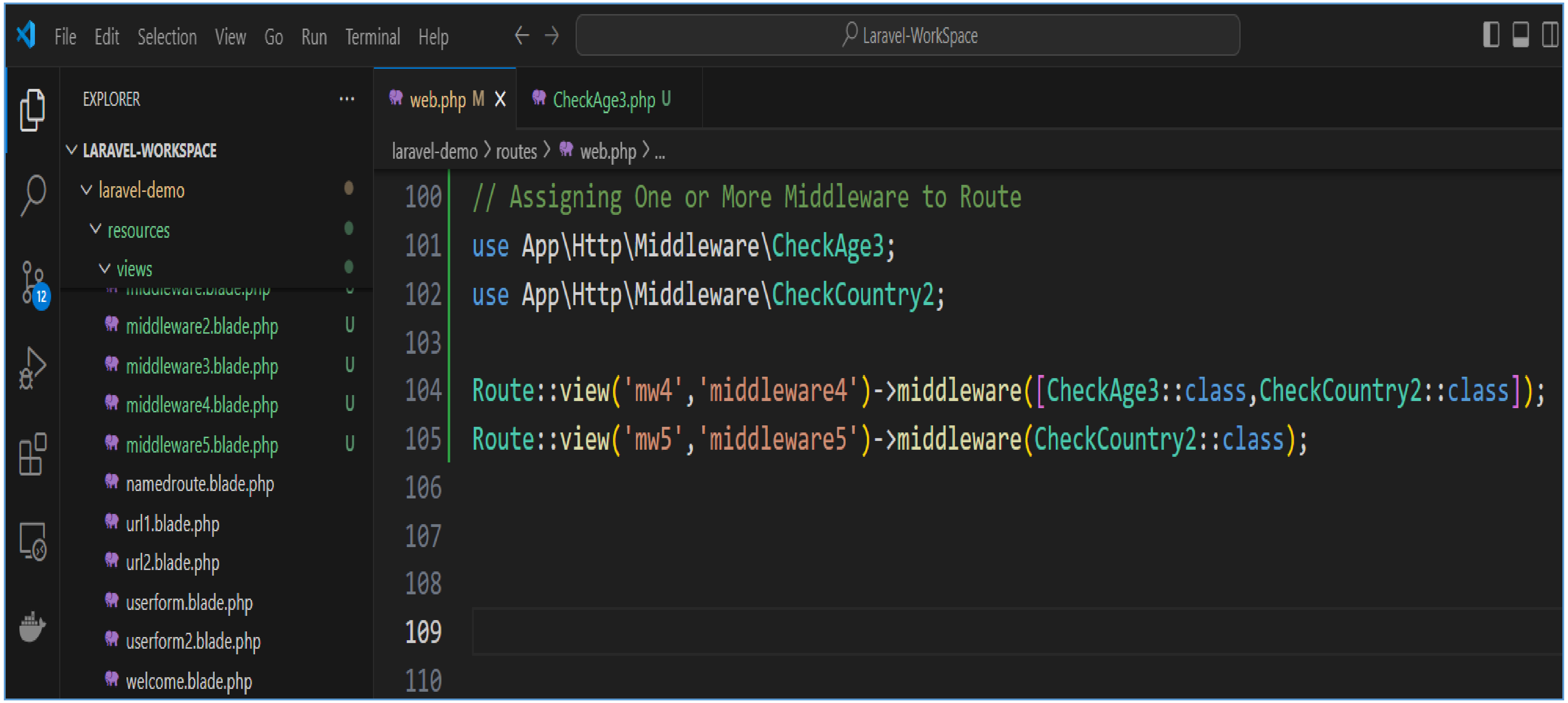
```
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view middleware4

  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\middleware4.blade.php] created successfully.

yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view middleware5

  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\middleware5.blade.php] created successfully.
```

# Cont. ...



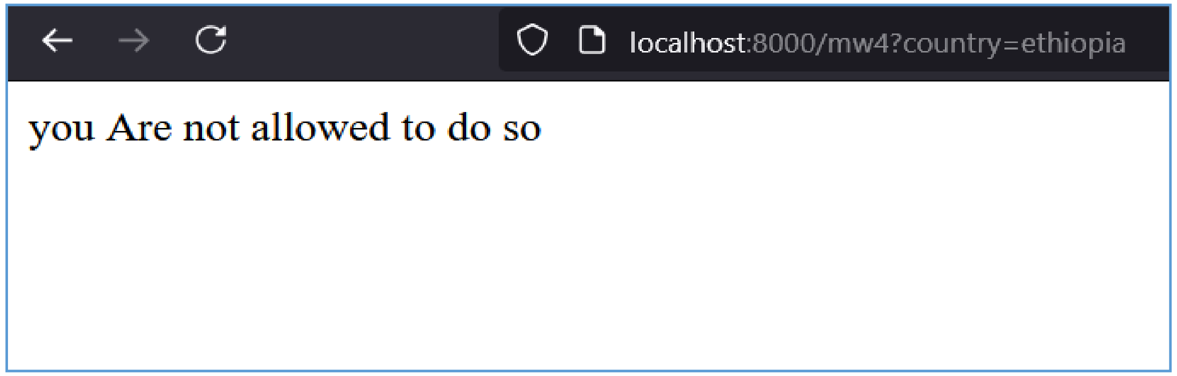
```php
<?php
namespace App\Http\Middleware;
use Closure;
use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;

class CheckAge3
{
    /**
     * Handle an incoming request.
     *
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response
     */
    public function handle(Request $request, Closure $next): Response
    {
        if($request->age<18){
            die("you Are not allowed to do so");
        }
        return $next($request);
    }
}
```

# Cont. …

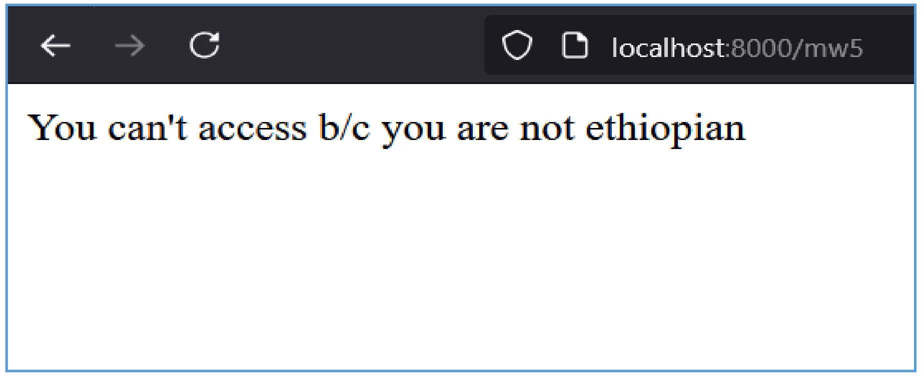


```php
// Assigning One or More Middleware to Route
use App\Http\Middleware\CheckAge3;
use App\Http\Middleware\CheckCountry2;

Route::view('mw4','middleware4')->middleware([CheckAge3::class,CheckCountry2::class]);
Route::view('mw5','middleware5')->middleware(CheckCountry2::class);
```

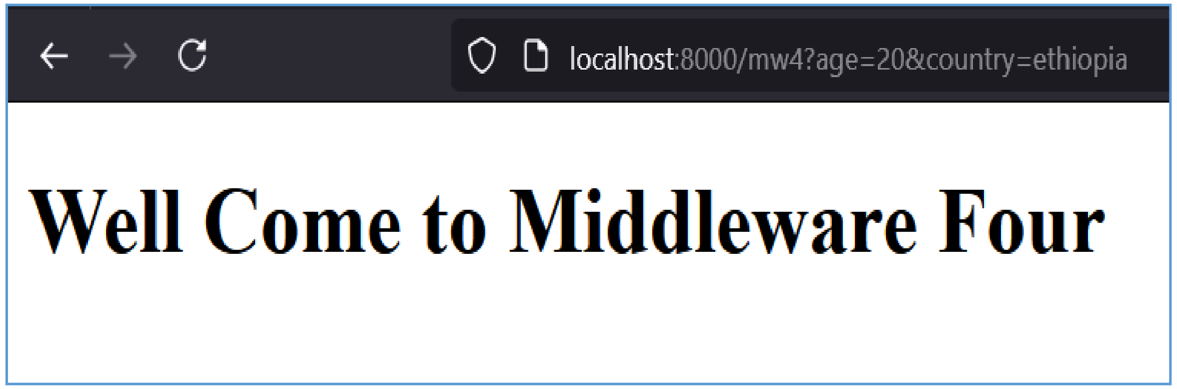
## Cont. ...


localhost:8000/mw4?country=ethiopia

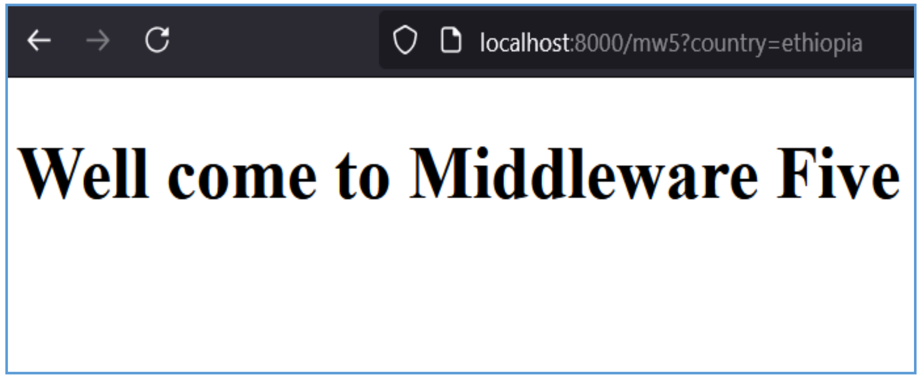
you Are not allowed to do so


localhost:8000/mw5

You can't access b/c you are not ethiopian

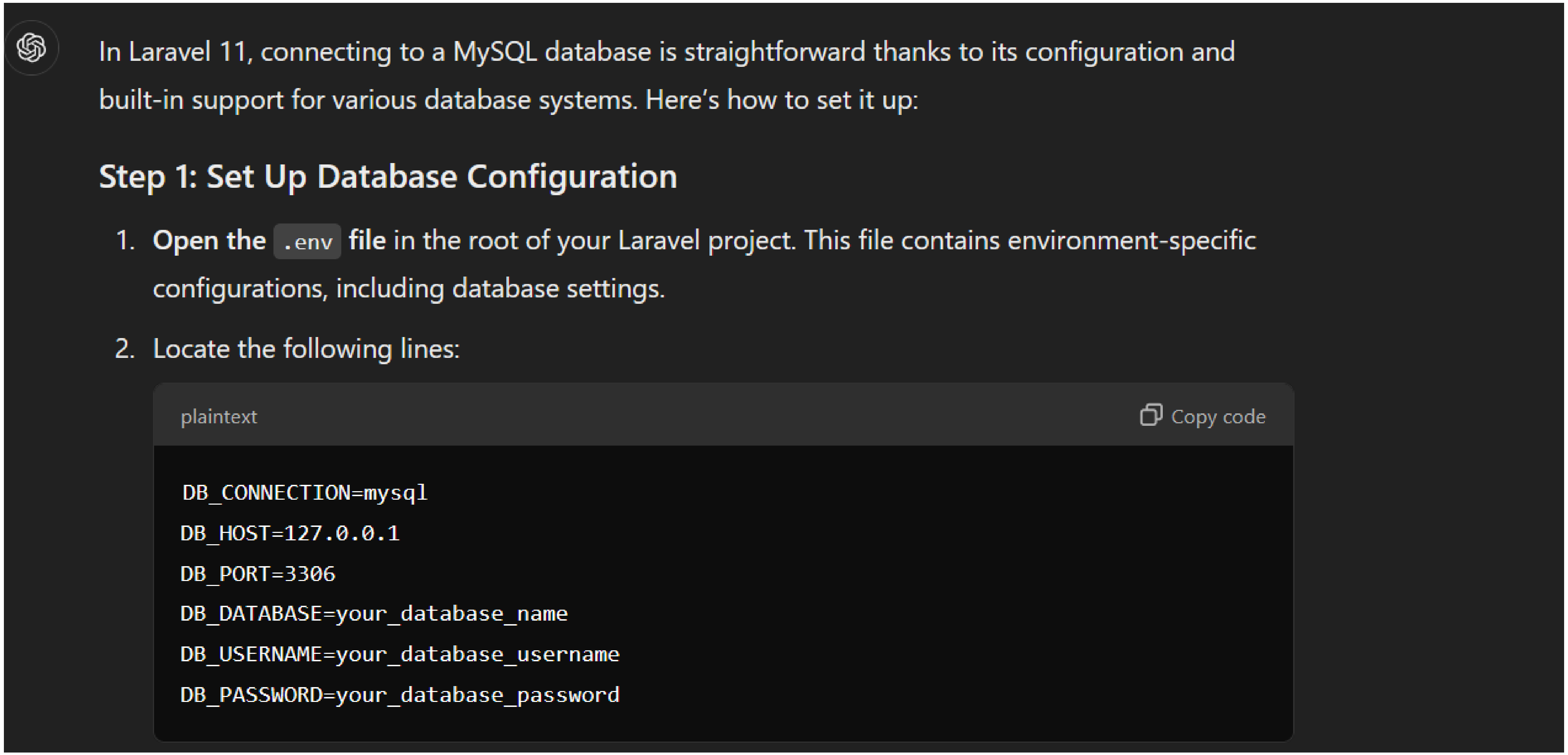

localhost:8000/mw4?age=20&country=ethiopia

Well Come to Middleware Four


localhost:8000/mw5?country=ethiopia

Well come to Middleware Five

## Connect to MYSQL Database



In Laravel 11, connecting to a MySQL database is straightforward thanks to its configuration and built-in support for various database systems. Here's how to set it up:

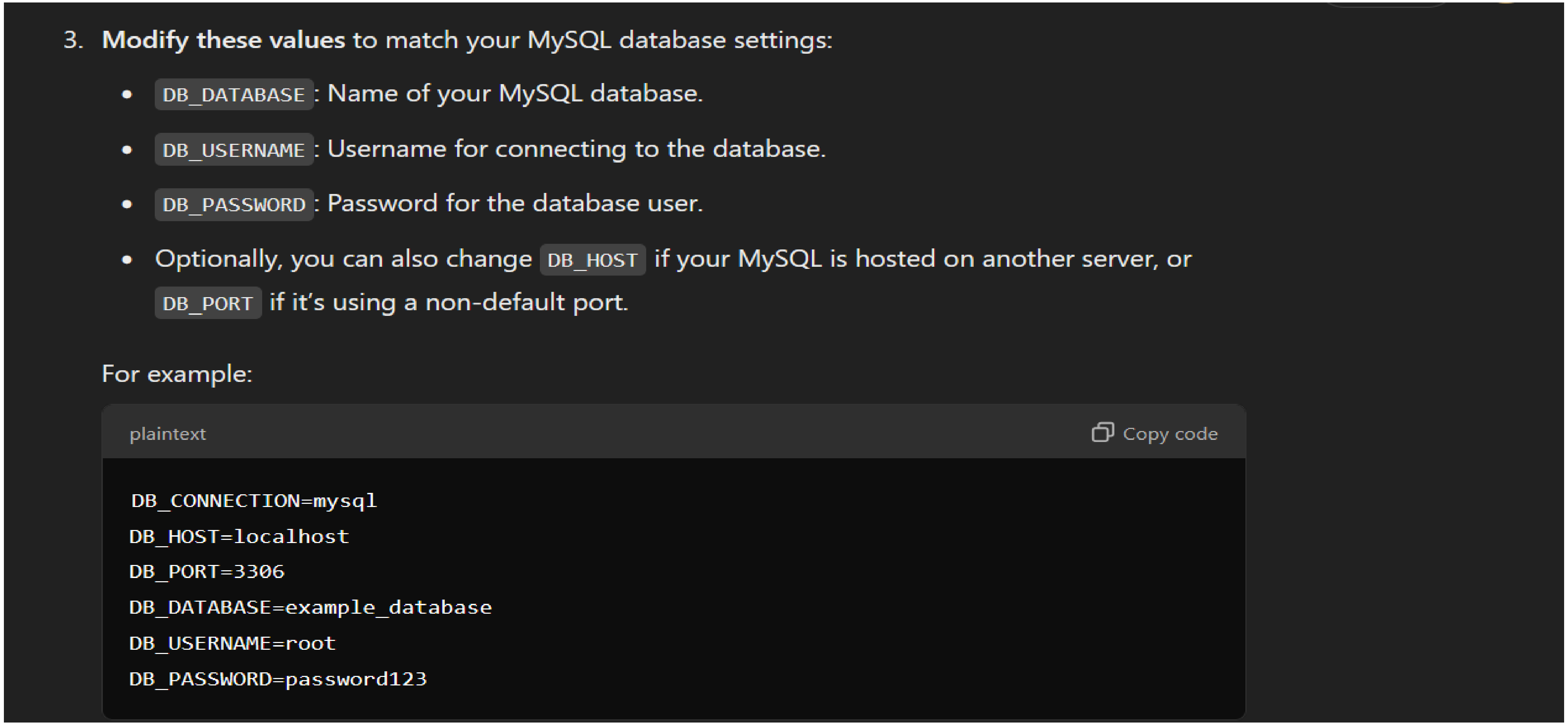
### Step 1: Set Up Database Configuration

1. **Open the `.env` file** in the root of your Laravel project. This file contains environment-specific configurations, including database settings.
2. Locate the following lines:

plaintext

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=your_database_name
DB_USERNAME=your_database_username
DB_PASSWORD=your_database_password
```

# Cont. …

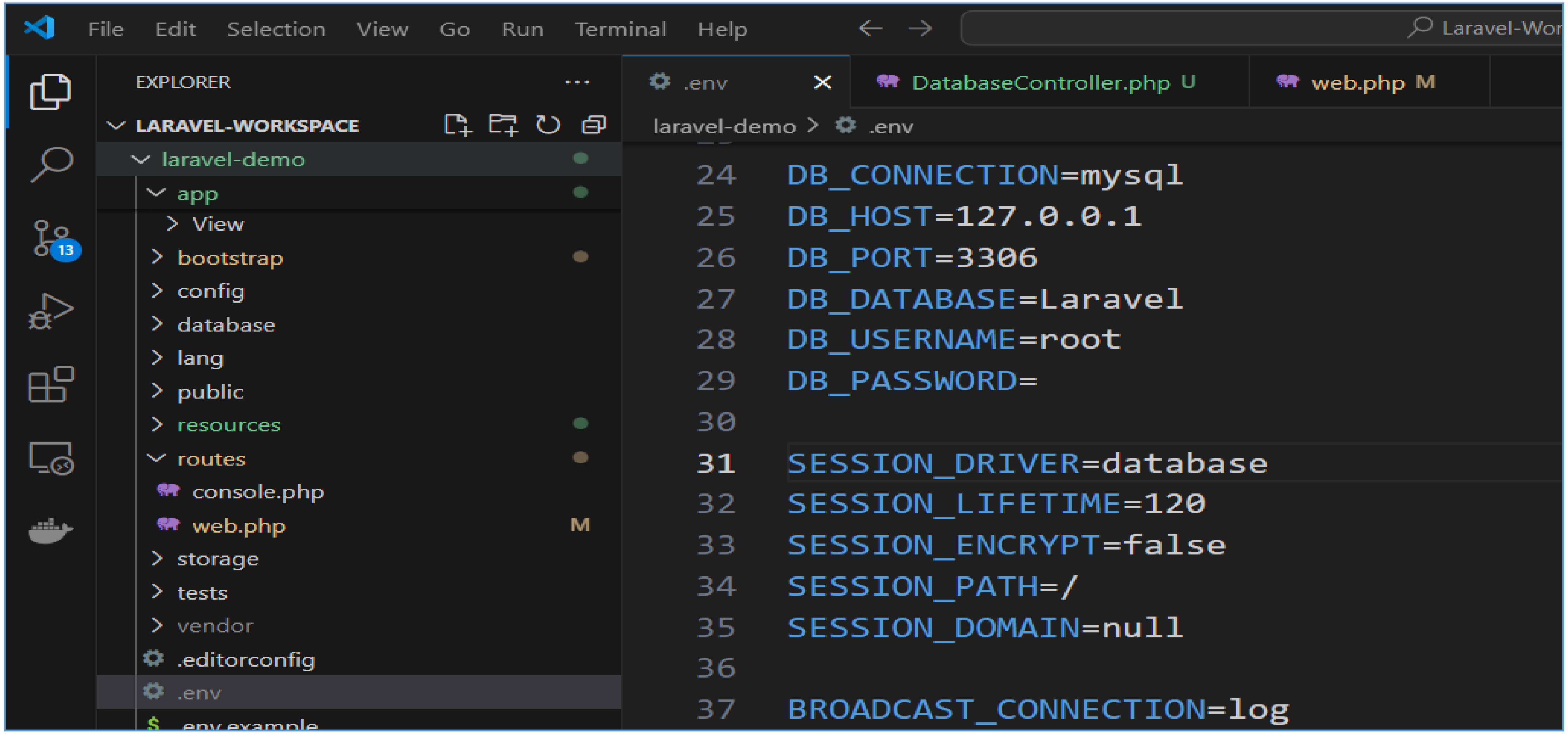


3. **Modify these values** to match your MySQL database settings:
    - `DB_DATABASE`: Name of your MySQL database.
    - `DB_USERNAME`: Username for connecting to the database.
    - `DB_PASSWORD`: Password for the database user.
    - Optionally, you can also change `DB_HOST` if your MySQL is hosted on another server, or `DB_PORT` if it's using a non-default port.

    For example:

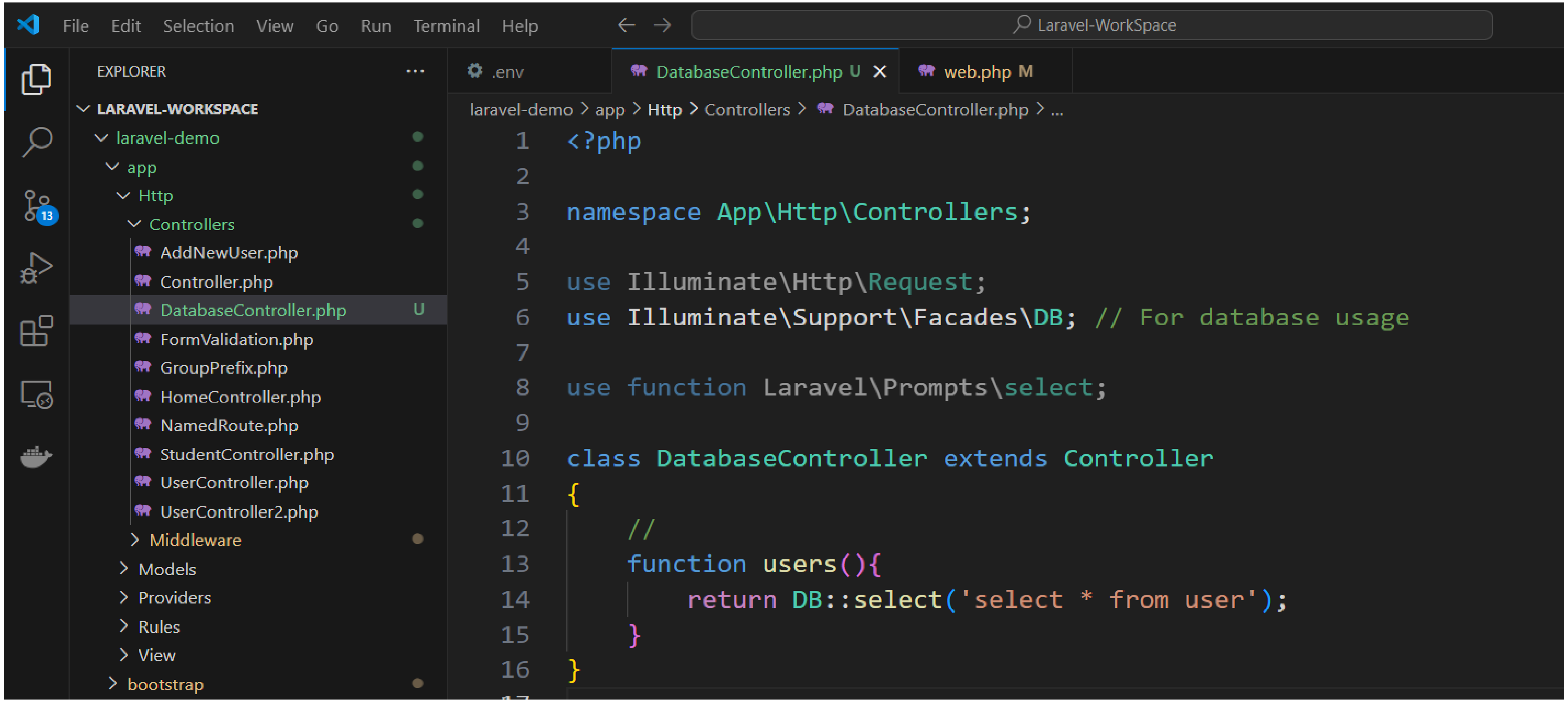
```
DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=example_database
DB_USERNAME=root
DB_PASSWORD=password123
```

## Cont. ...



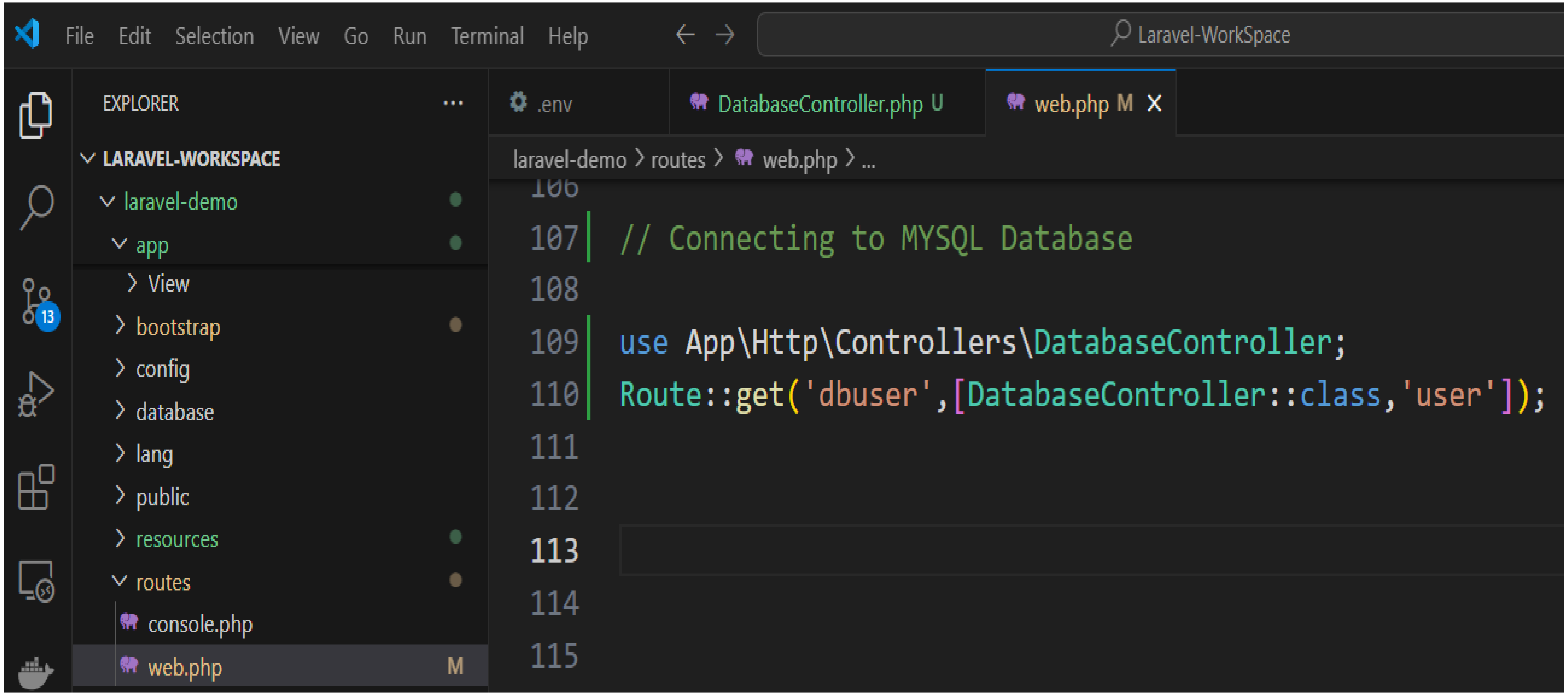
```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=Laravel
DB_USERNAME=root
DB_PASSWORD=

SESSION_DRIVER=database
SESSION_LIFETIME=120
SESSION_ENCRYPT=false
SESSION_PATH=/
SESSION_DOMAIN=null

BROADCAST_CONNECTION=log
```

## Cont. …



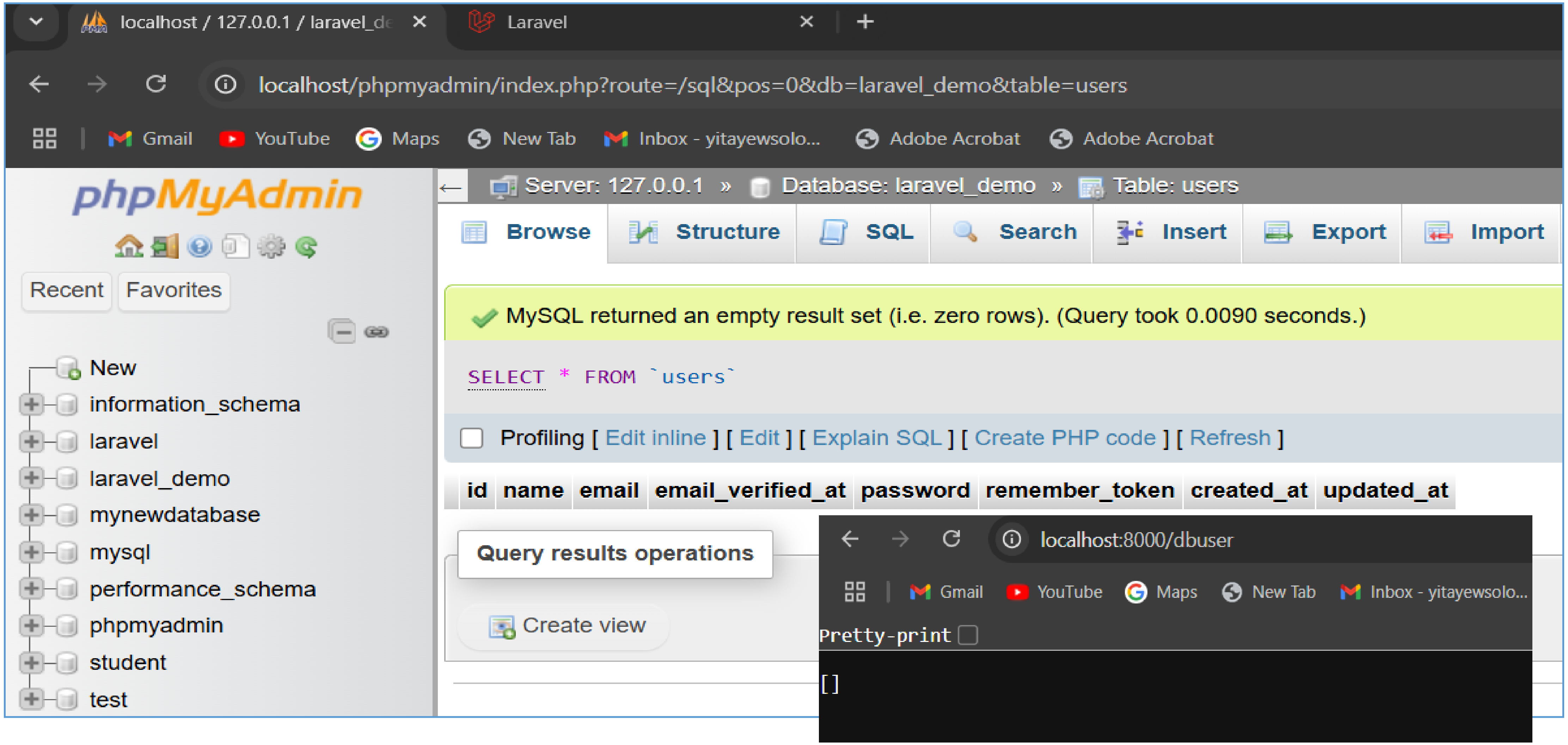
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB; // For database usage

use function Laravel\Prompts\select;

class DatabaseController extends Controller
{
    //
    function users(){
        return DB::select('select * from user');
    }
}
```

## Cont. …



```
// Connecting to MYSQL Database

use App\Http\Controllers\DatabaseController;
Route::get('dbuser',[DatabaseController::class,'user']);
```

# Cont. …



localhost/phpmyadmin/index.php?route=/sql&pos=0&db=laravel_demo&table=users

Server: 127.0.0.1 » Database: laravel_demo » Table: users

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0090 seconds.)

```
SELECT * FROM `users`
```

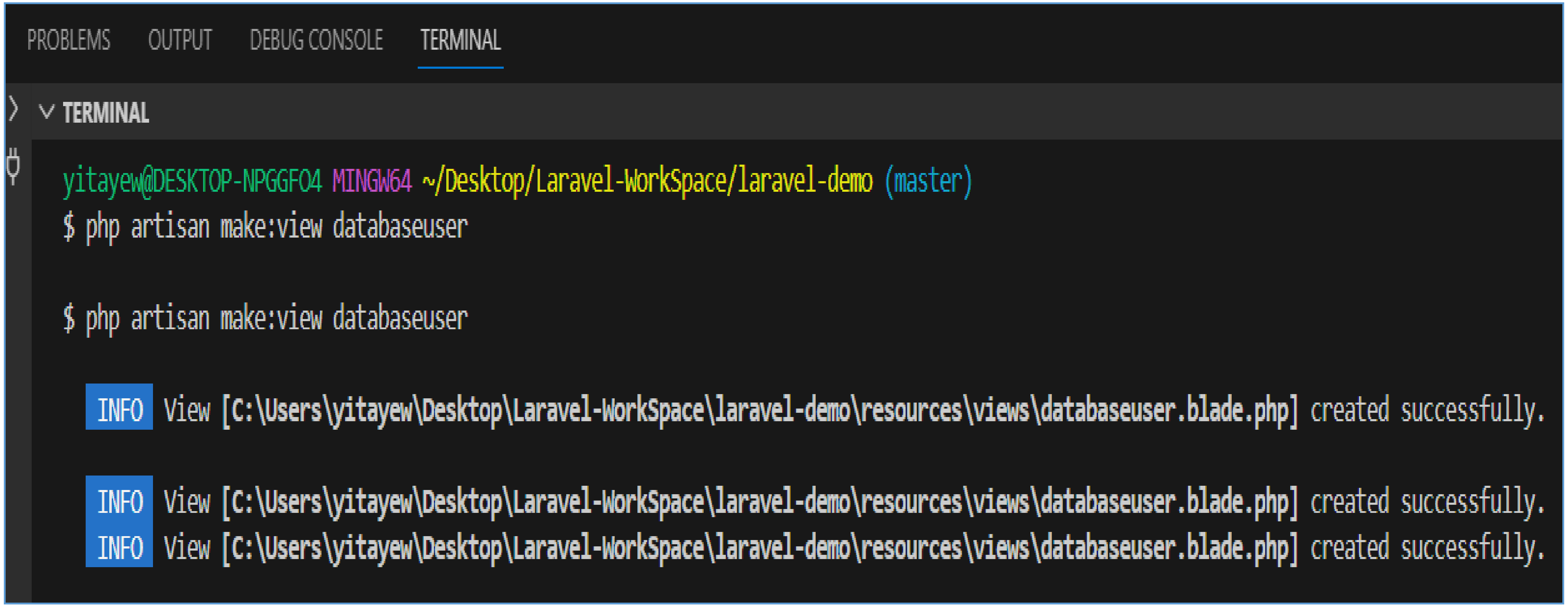
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

| id | name | email | email_verified_at | password | remember_token | created_at | updated_at |
|---|---|---|---|---|---|---|---|

localhost:8000/dbuser

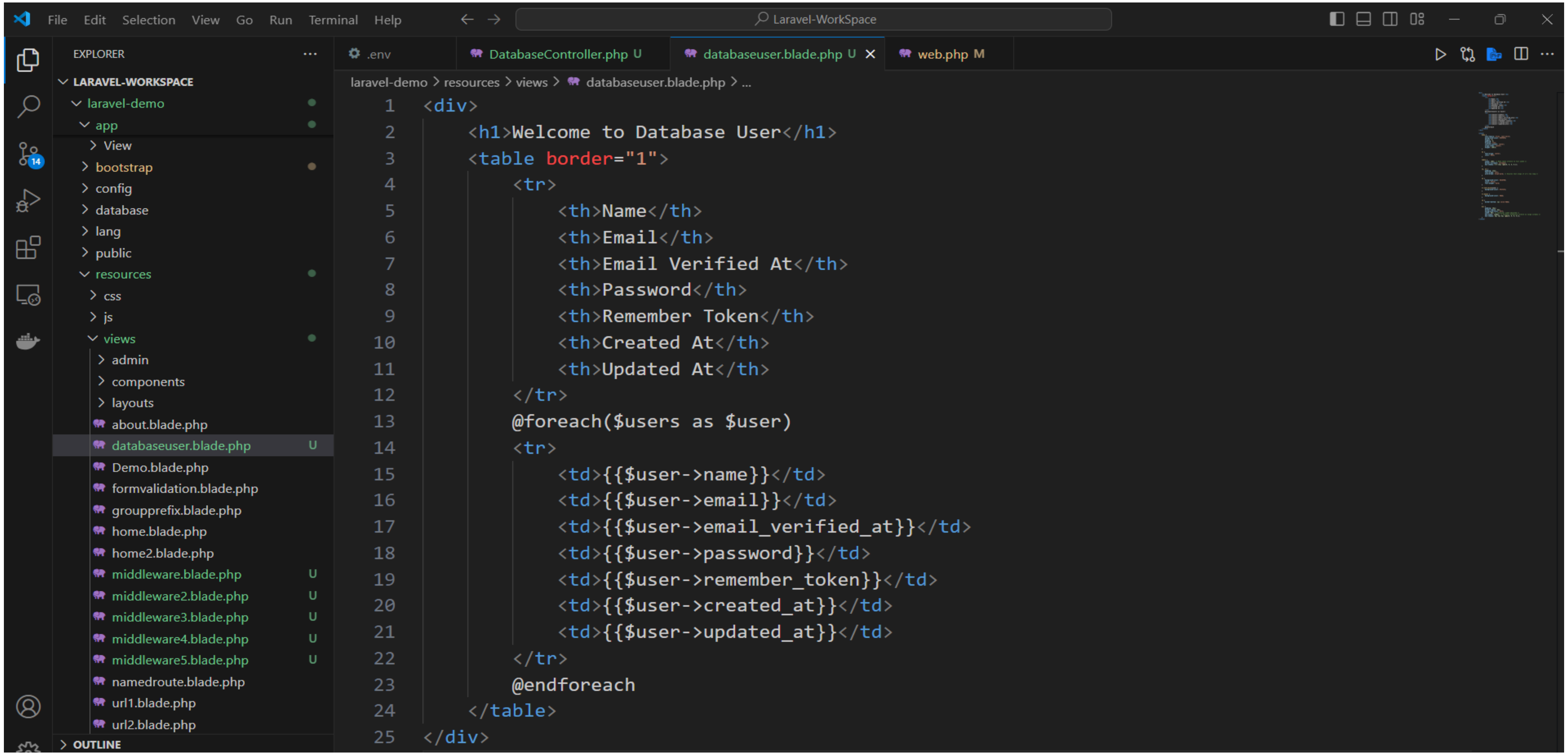
```
[]
```

## Display Database Data on UI



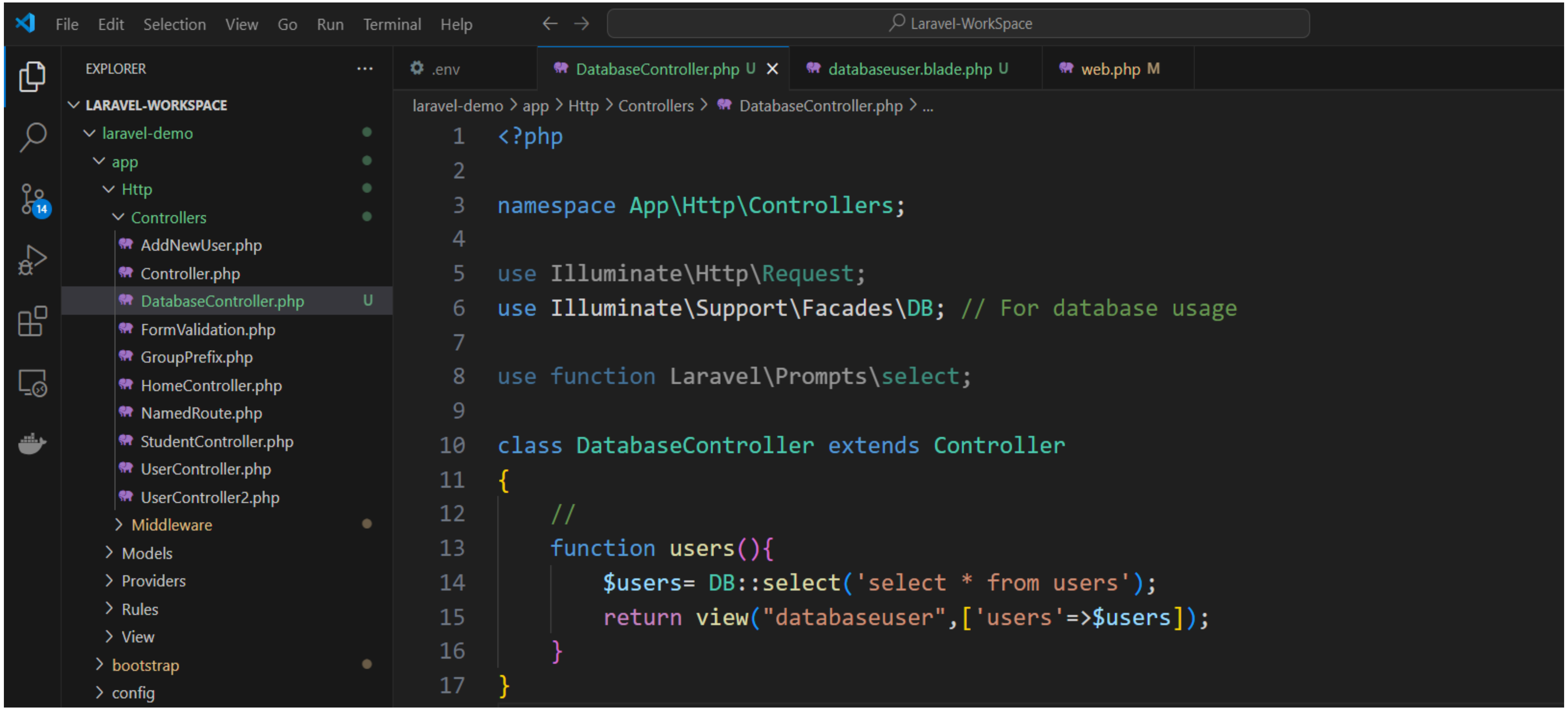
```
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view databaseuser

$ php artisan make:view databaseuser

  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\databaseuser.blade.php] created successfully.

  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\databaseuser.blade.php] created successfully.
  INFO  View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\databaseuser.blade.php] created successfully.
```

# Cont. …



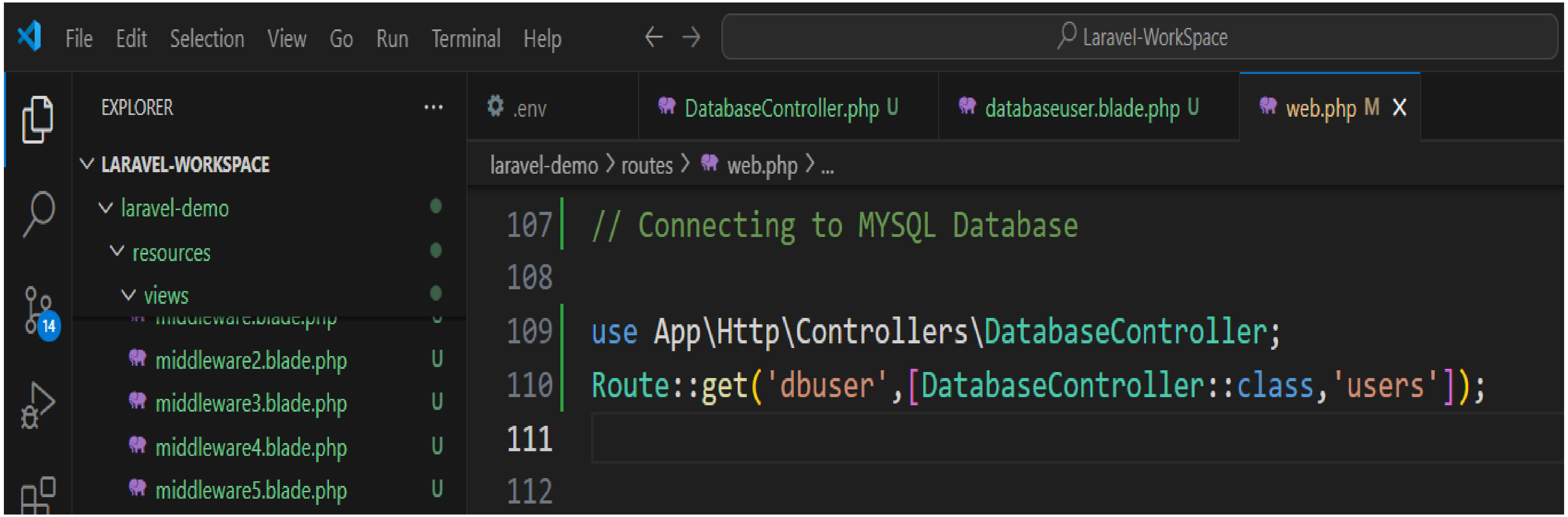
```
<div>
    <h1>Welcome to Database User</h1>
    <table border="1">
        <tr>
            <th>Name</th>
            <th>Email</th>
            <th>Email Verified At</th>
            <th>Password</th>
            <th>Remember Token</th>
            <th>Created At</th>
            <th>Updated At</th>
        </tr>
        @foreach($users as $user)
        <tr>
            <td>{{$user->name}}</td>
            <td>{{$user->email}}</td>
            <td>{{$user->email_verified_at}}</td>
            <td>{{$user->password}}</td>
            <td>{{$user->remember_token}}</td>
            <td>{{$user->created_at}}</td>
            <td>{{$user->updated_at}}</td>
        </tr>
        @endforeach
    </table>
</div>
```

## Cont. ...



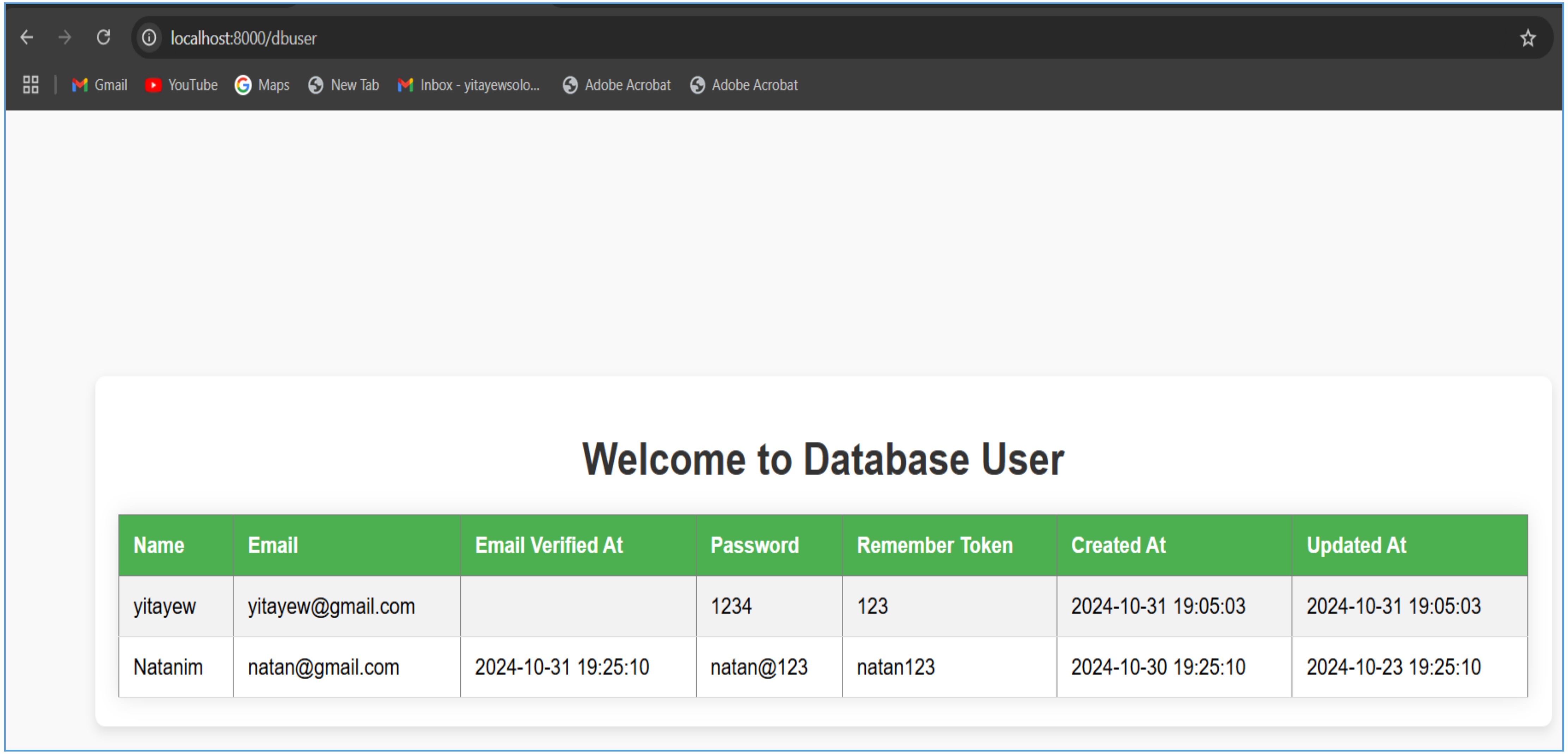
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB; // For database usage

use function Laravel\Prompts\select;

class DatabaseController extends Controller
{
    //
    function users(){
        $users= DB::select('select * from users');
        return view("databaseuser",['users'=>$users]);
    }
}
```

## Cont. …



```
// Connecting to MYSQL Database

use App\Http\Controllers\DatabaseController;
Route::get('dbuser',[DatabaseController::class,'users']);

```

## Cont. ...


localhost:8000/dbuser

Gmail YouTube Maps New Tab Inbox - yitayewsolo... Adobe Acrobat Adobe Acrobat

### Welcome to Database User

| Name | Email | Email Verified At | Password | Remember Token | Created At | Updated At |
|---|---|---|---|---|---|---|
| yitayew | yitayew@gmail.com | | 1234 | 123 | 2024-10-31 19:05:03 | 2024-10-31 19:05:03 |
| Natanim | natan@gmail.com | 2024-10-31 19:25:10 | natan@123 | natan123 | 2024-10-30 19:25:10 | 2024-10-23 19:25:10 |

## Model in Laravel

- In Laravel (and other MVC frameworks), a model represents the **data** and the **business logic** of an application. It is one of the core components of the Model-View-Controller (MVC) architecture, which Laravel follows.

## Key Points about Models in Laravel

- ❖**Data Representation**: Models are typically used to represent a **single table** in the database, and each **instance** of a model represents a **single row** in that table.

- For example, a **User** model would correspond to a **users** table in the database, and each User instance would represent a specific user record.

## Cont. …

- ❖**Eloquent ORM**: Laravel uses Eloquent ORM (Object-Relational Mapping) for database interaction. Eloquent makes it easy to work with database records using an expressive, simple syntax, without writing raw SQL queries.
- ❖Each model is linked to a database table, and **Eloquent** provides methods to query, **insert**, **update**, and **delete** records with ease.

## Cont. …

- ❖**Database Interaction**: Models act as an intermediary between the database and the application logic. They contain methods and properties for querying and manipulating the data.

- For instance, with Eloquent, you can fetch a record like User::find(1) to retrieve a user with ID 1, or
- User::where('email', 'example@example.com')->get() to fetch users based on their email.

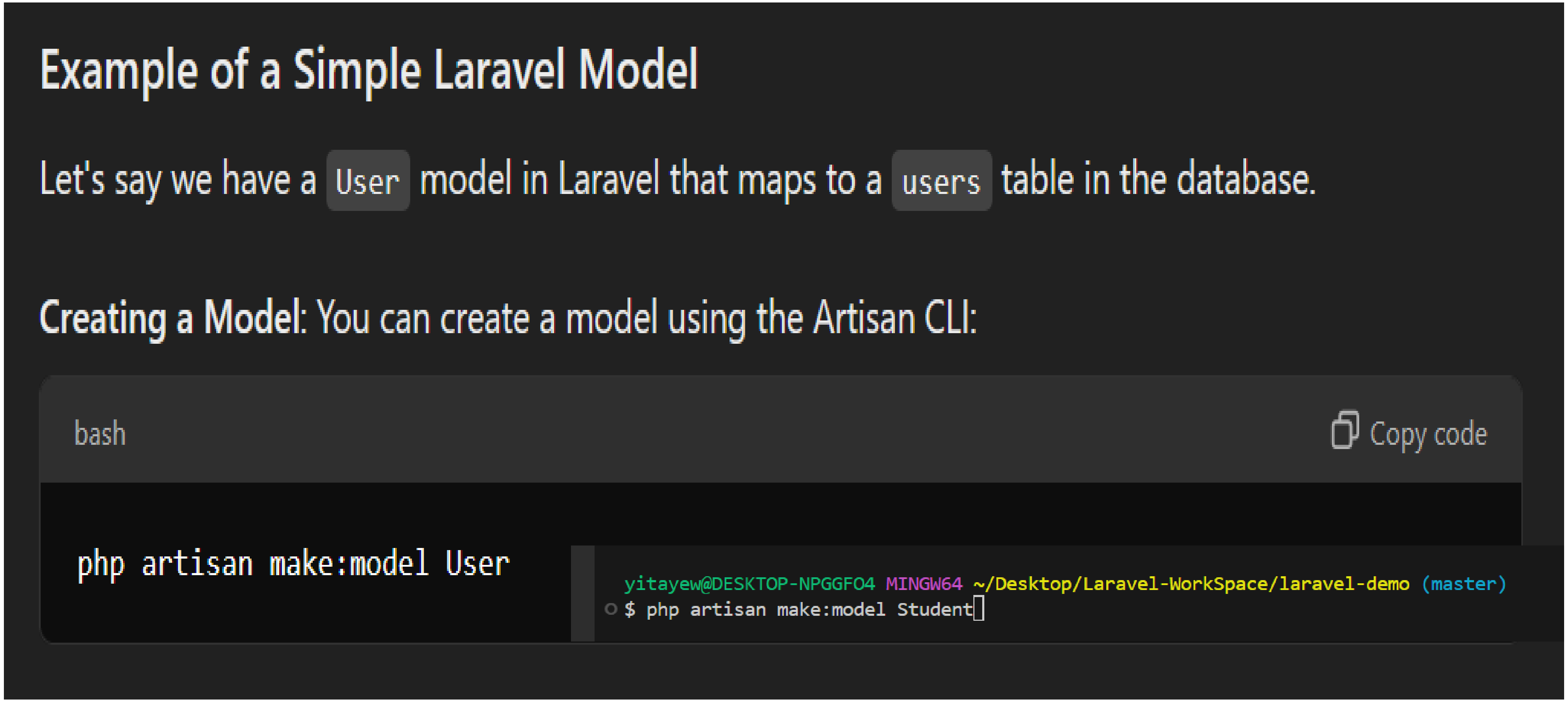
## Cont. …

- Relationships: Eloquent allows defining relationships between models, making it easy to work with related data. Examples of relationships include:
  - ✓ One-to-One (e.g., a User has one Profile)
  - ✓ One-to-Many (e.g., a Post has many Comments)
  - ✓ Many-to-Many (e.g., a User belongs to many Roles)

## Cont. …

- ❖ **Business Logic**: Models can also contain business logic or data manipulation methods specific to that model.
- This helps keep the application logic organized and separated from the controller or view.

## Example



### Example of a Simple Laravel Model

Let's say we have a `User` model in Laravel that maps to a `users` table in the database.

**Creating a Model:** You can create a model using the Artisan CLI:

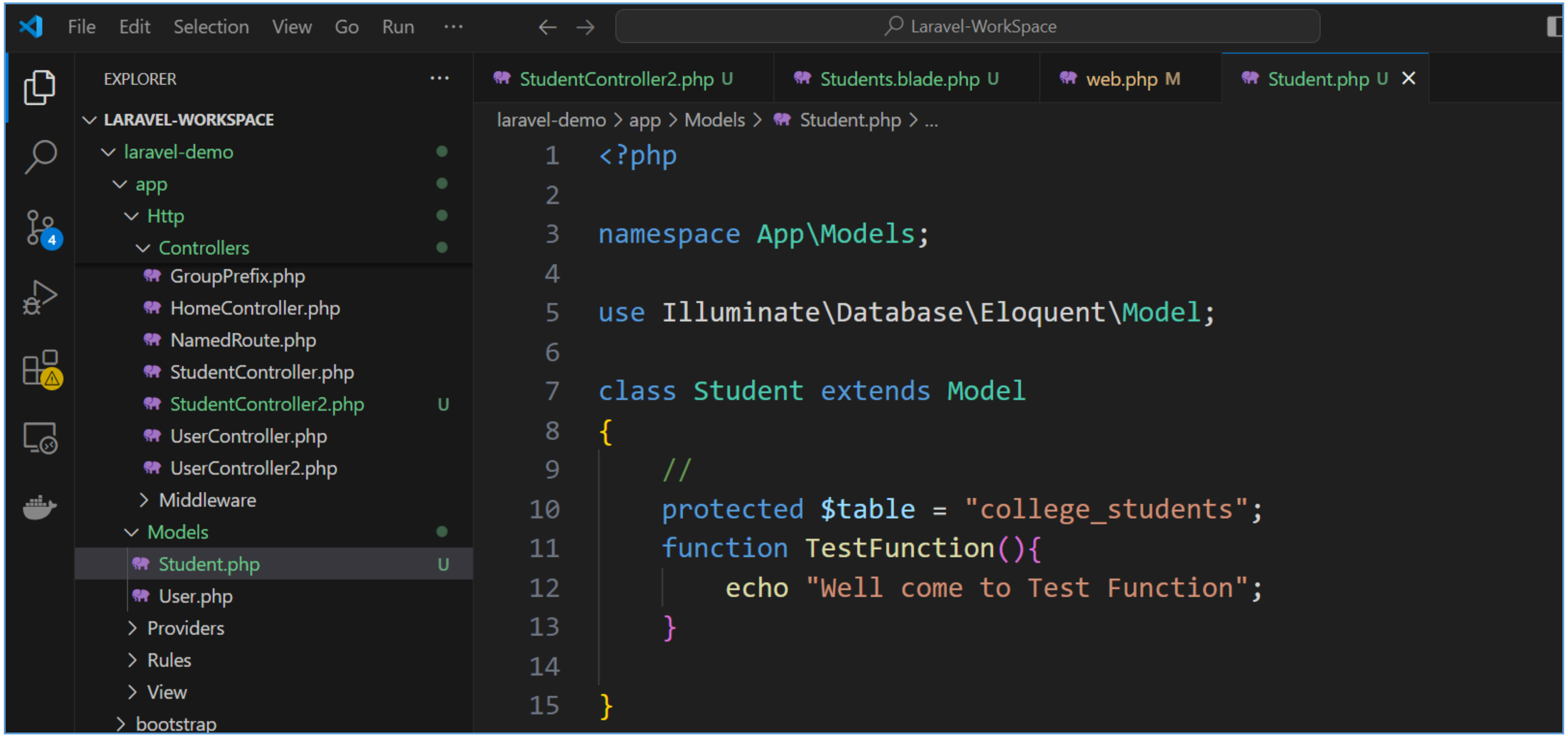
bash

Copy code

```bash
php artisan make:model User
```

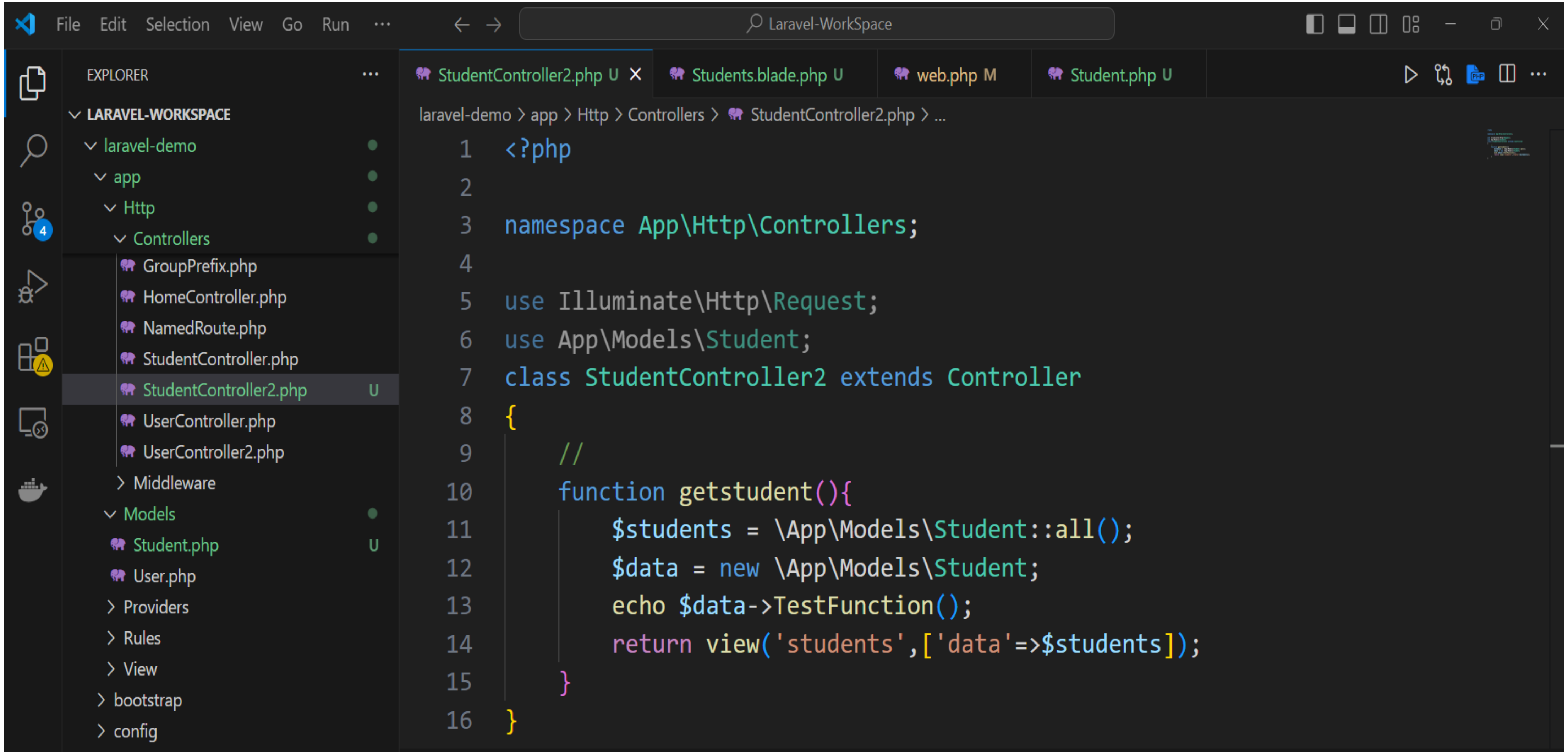
```
yitayew@DESKTOP-NPGGF04 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:model Student
```

# Cont. ...



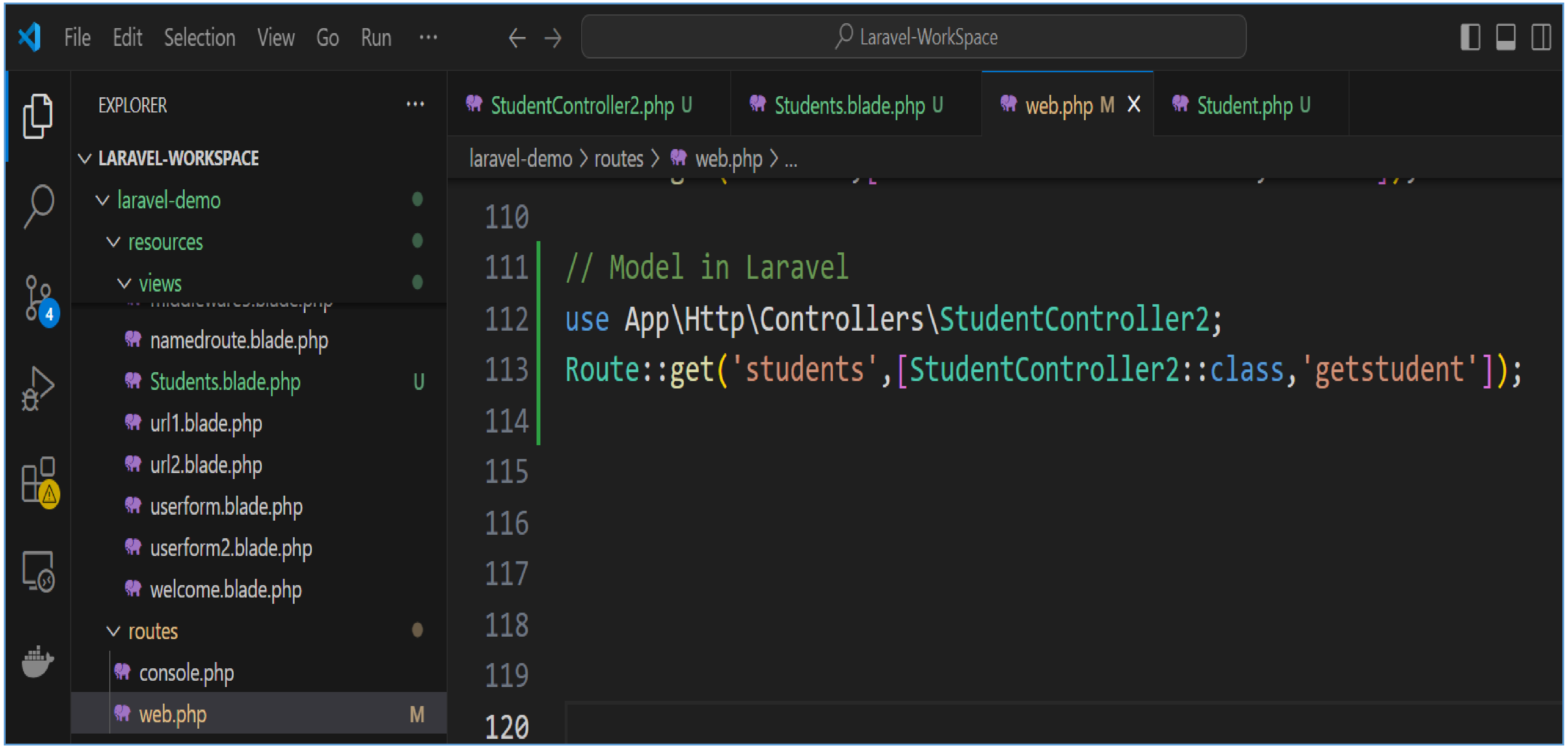
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Student extends Model
{
    //
    protected $table = "college_students";
    function TestFunction(){
        echo "Well come to Test Function";
    }

}
```

# Cont. ...



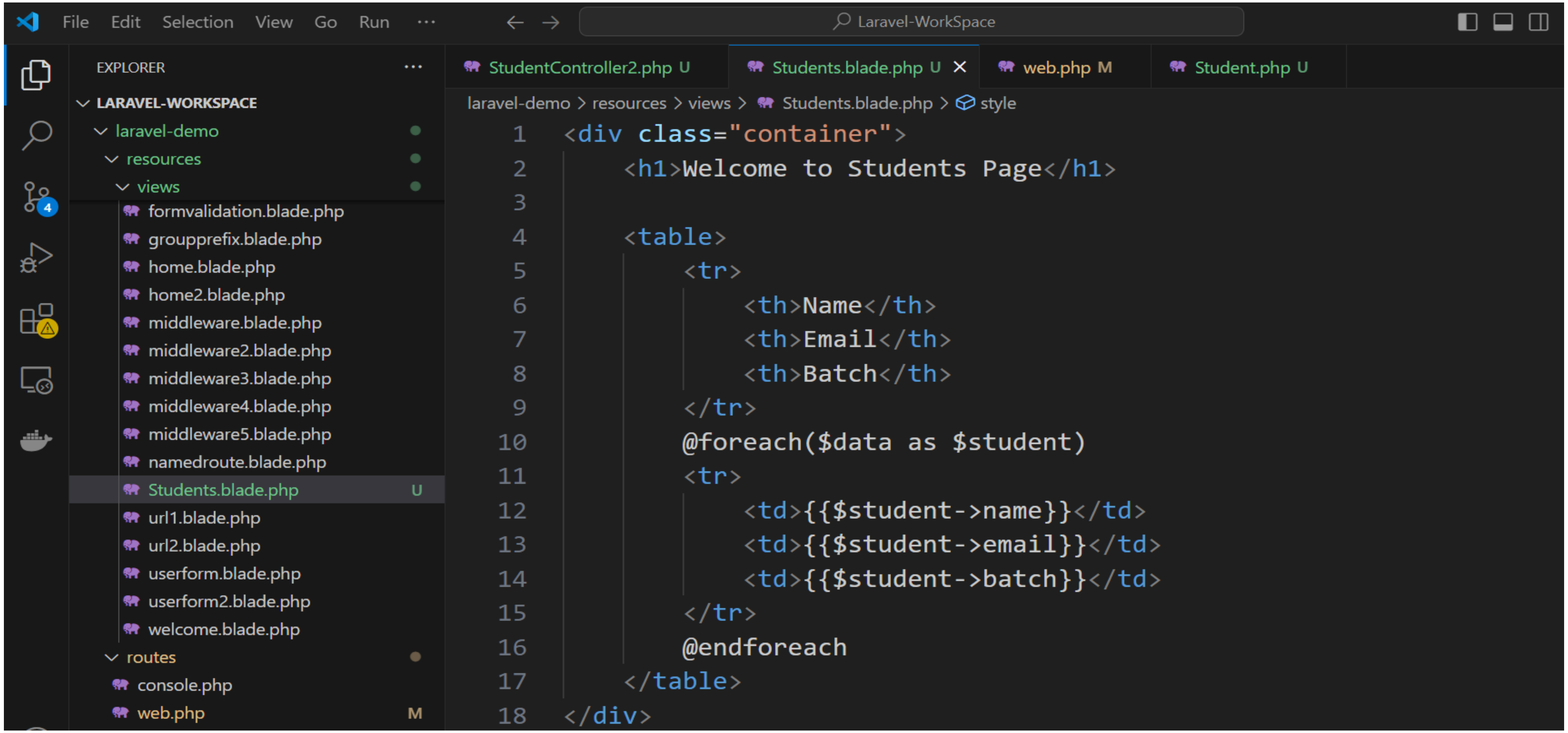
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Student;
class StudentController2 extends Controller
{
    //
    function getstudent(){
        $students = \App\Models\Student::all();
        $data = new \App\Models\Student;
        echo $data->TestFunction();
        return view('students',['data'=>$students]);
    }
}
```

# Cont. ...



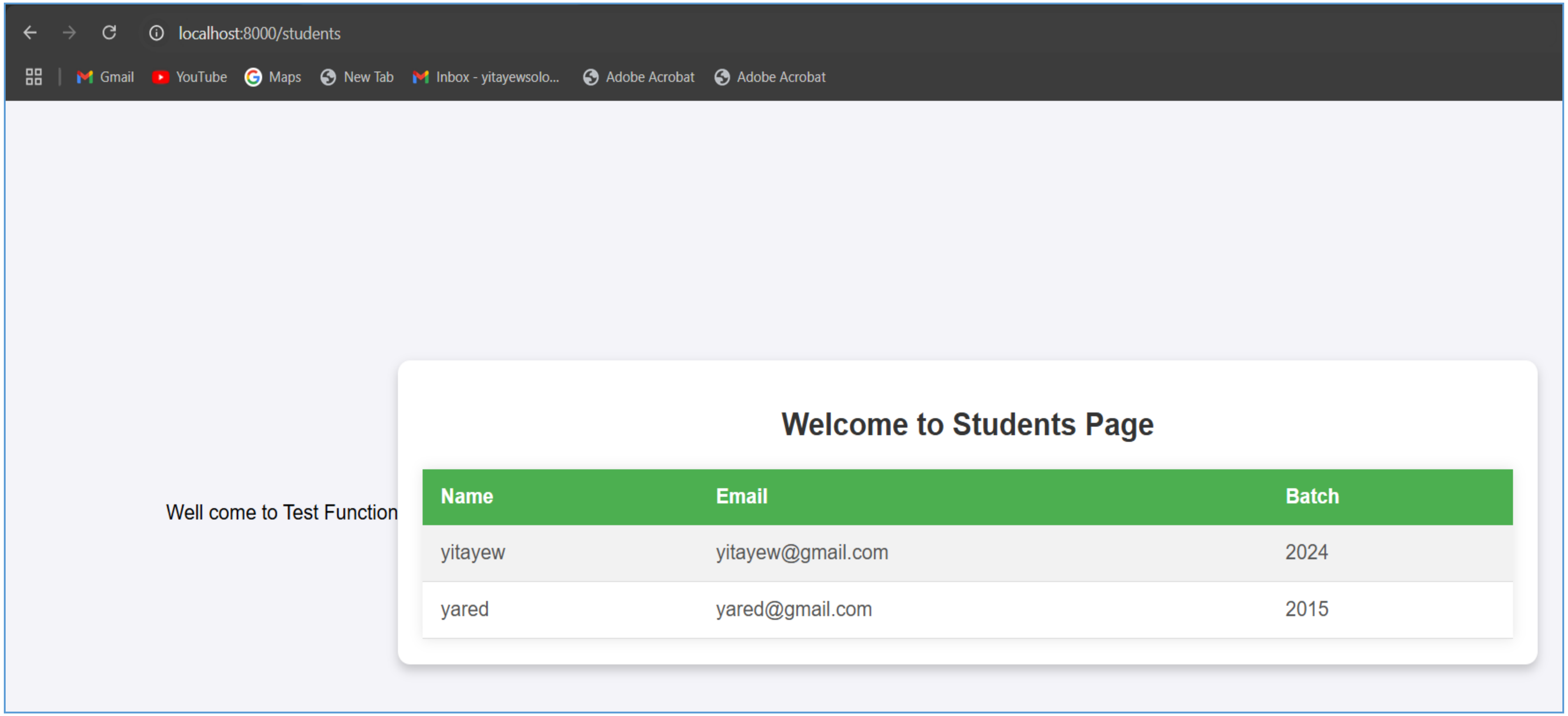
```php
// Model in Laravel
use App\Http\Controllers\StudentController2;
Route::get('students',[StudentController2::class,'getstudent']);
```

## Cont. …



```
<div class="container">
    <h1>Welcome to Students Page</h1>

    <table>
        <tr>
            <th>Name</th>
            <th>Email</th>
            <th>Batch</th>
        </tr>
        @foreach($data as $student)
        <tr>
            <td>{{$student->name}}</td>
            <td>{{$student->email}}</td>
            <td>{{$student->batch}}</td>
        </tr>
        @endforeach
    </table>
</div>
```

## Cont. ...



Well come to Test Function

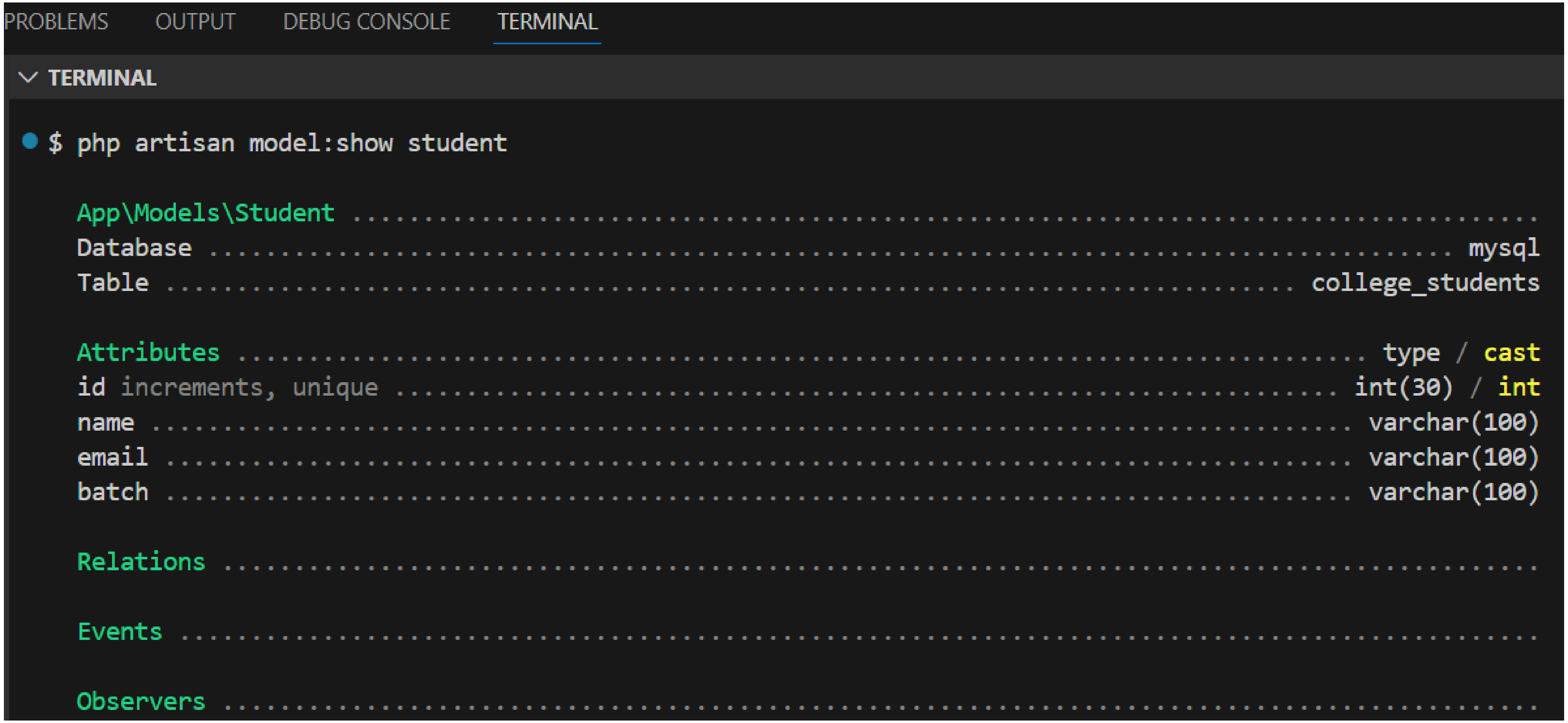
Welcome to Students Page

| Name | Email | Batch |
|---|---|---|
| yitayew | yitayew@gmail.com | 2024 |
| yared | yared@gmail.com | 2015 |

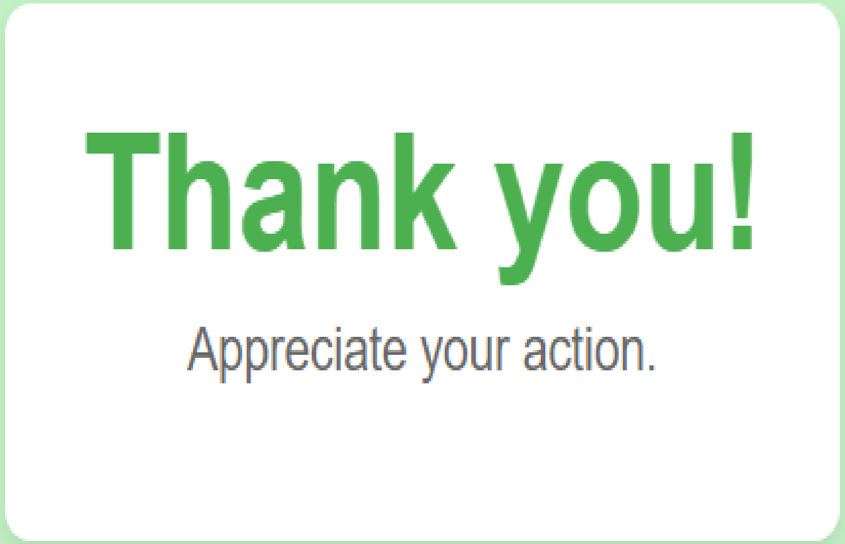
## Model Inspect in Laravel

- In Laravel, Model Inspection allows you to gain insights into the structure and properties of a model, including its table name, attributes, relationships, scopes, and more.
- Laravel provides several methods and tools that you can use to inspect models, which is particularly useful when debugging or developing complex applications.

## Cont. ...



```
● $ php artisan model:show student

    App\Models\Student .........................................................
    Database ........................................................... mysql
    Table ...................................................... college_students

    Attributes ................................................... type / cast
    id increments, unique .................................... int(30) / int
    name ..................................................... varchar(100)
    email .................................................... varchar(100)
    batch .................................................... varchar(100)

    Relations ...................................................................

    Events ......................................................................

    Observers ...................................................................
```



Thank you!

Appreciate your action.