Laravel

Outlines of Discussion

- Database Query Builder
- Eloquent ORM
- Route Methods in Laravel
- HTTP Request Class
- Session in laravel

Database Query Builder

- Laravel's Query Builder provides a convenient, fluent **interface** for creating and running database queries. It is a **layer between** raw **SQL queries** and **Eloquent ORM**, allowing you to work with databases without writing raw SQL.
- The Query Builder supports all **major database** systems, including MySQL, PostgreSQL, SQLite, and SQL Server.

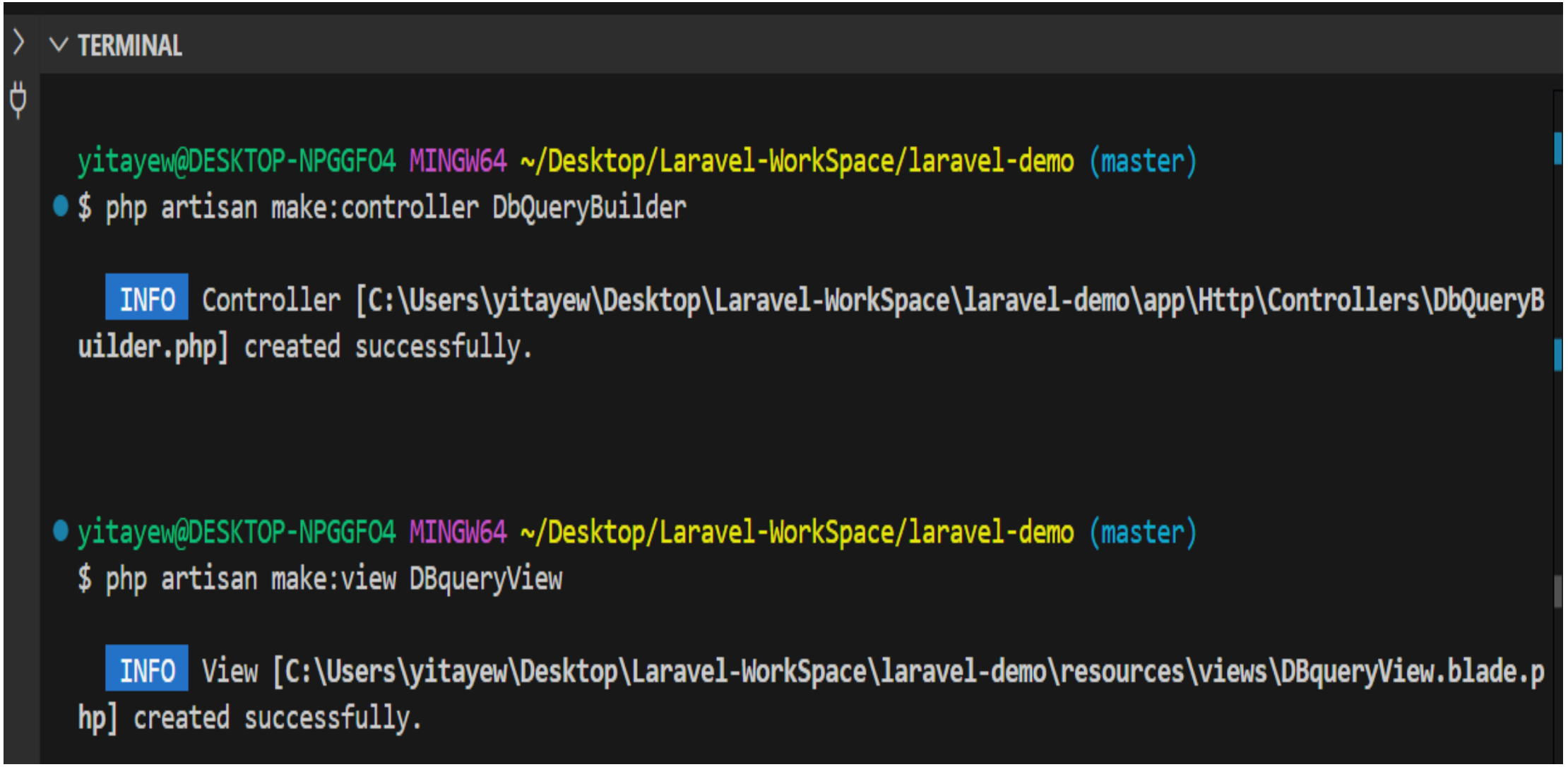
Cont. …

- The **Database Query Builder** in Laravel is a powerful and flexible interface for building and executing database queries in a structured, fluent, and programmatic way.
- It allows developers to interact with the database without writing **raw SQL queries**, making the code cleaner, readable, and secure.

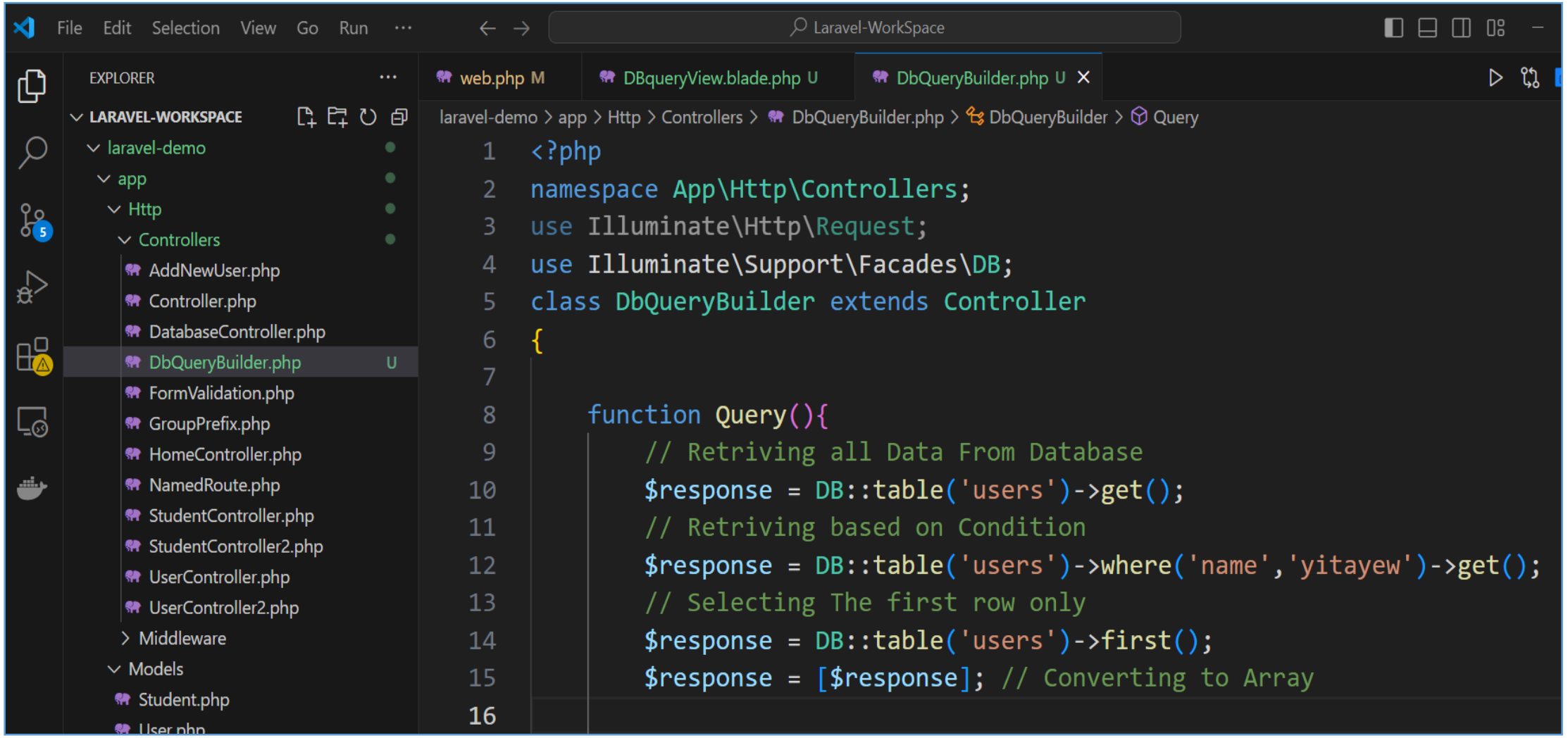
Key Features

- **Chainable Methods**: Query Builder methods are chainable, making queries more readable and easier to build.
- **Protection Against SQL Injection**: Automatically binds parameters, making queries safer.
- **Supports CRUD Operations**: Includes methods for all basic database operations.

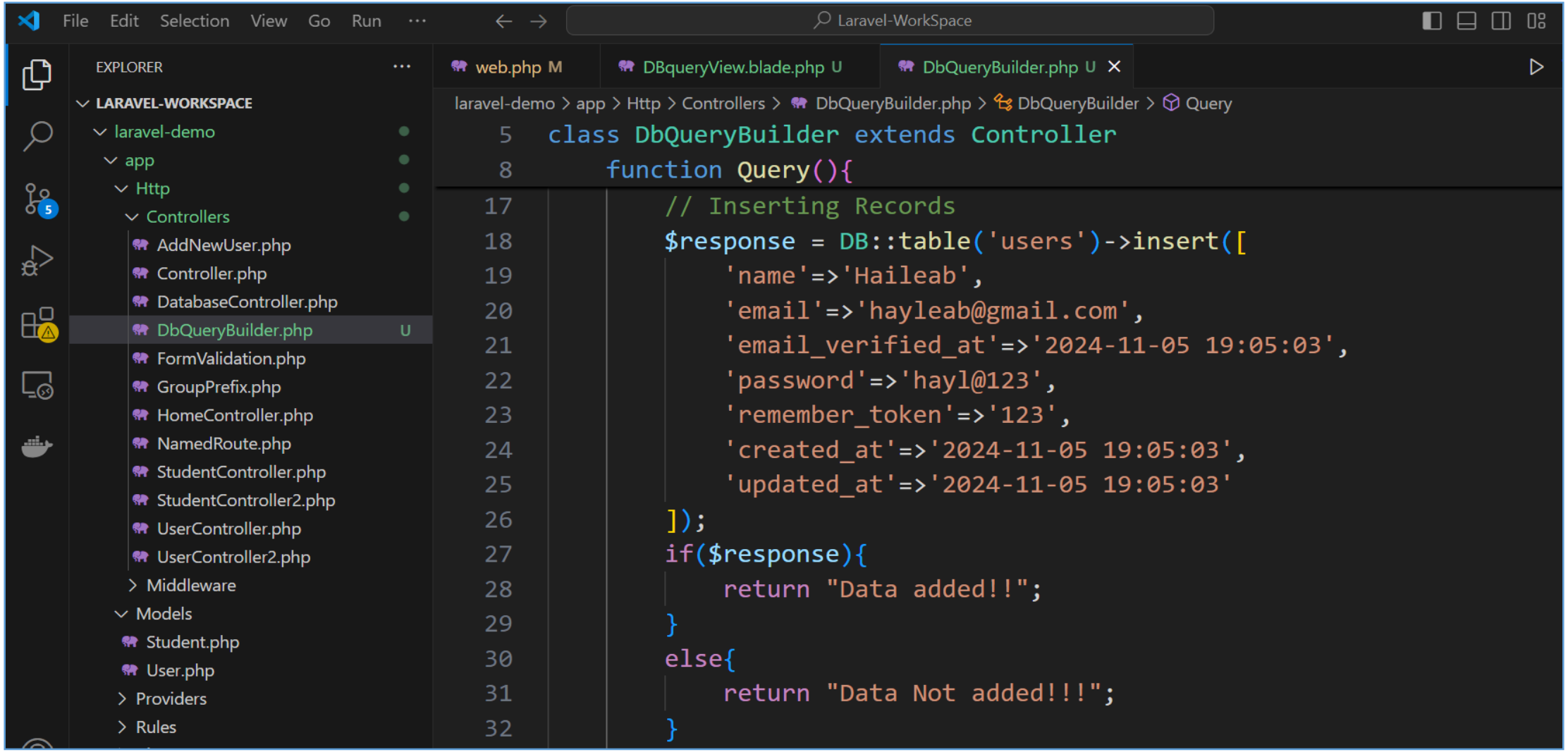
Example


TERMINAL
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:controller DbQueryBuilder
INFO Controller [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\app\Http\Controllers\DbQueryBuilder.php] created successfully.
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view DBqueryView
INFO View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\DBqueryView.blade.php] created successfully.

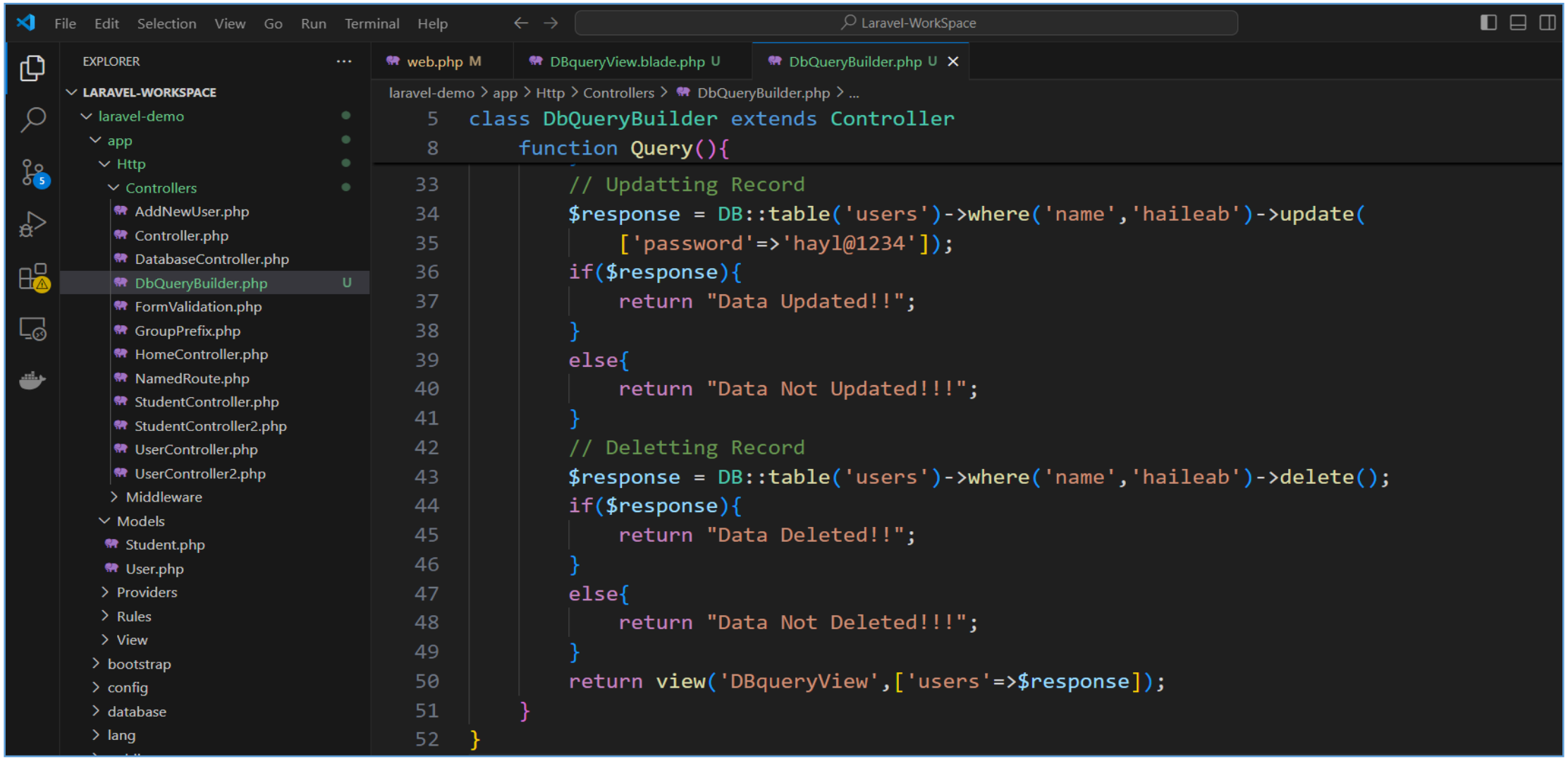
Data Manipulation Using Laravel



Inserting Records



Update and Delete Records



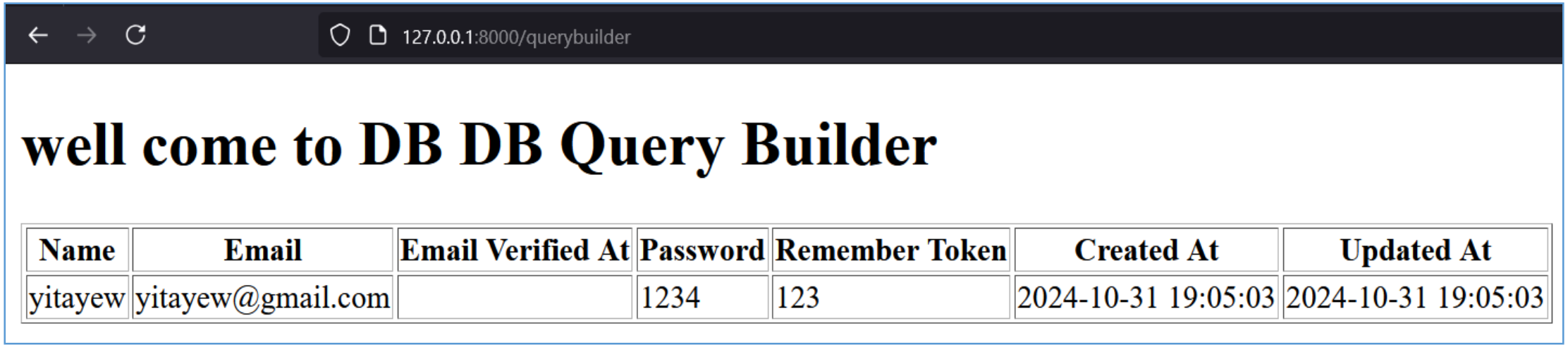
Route

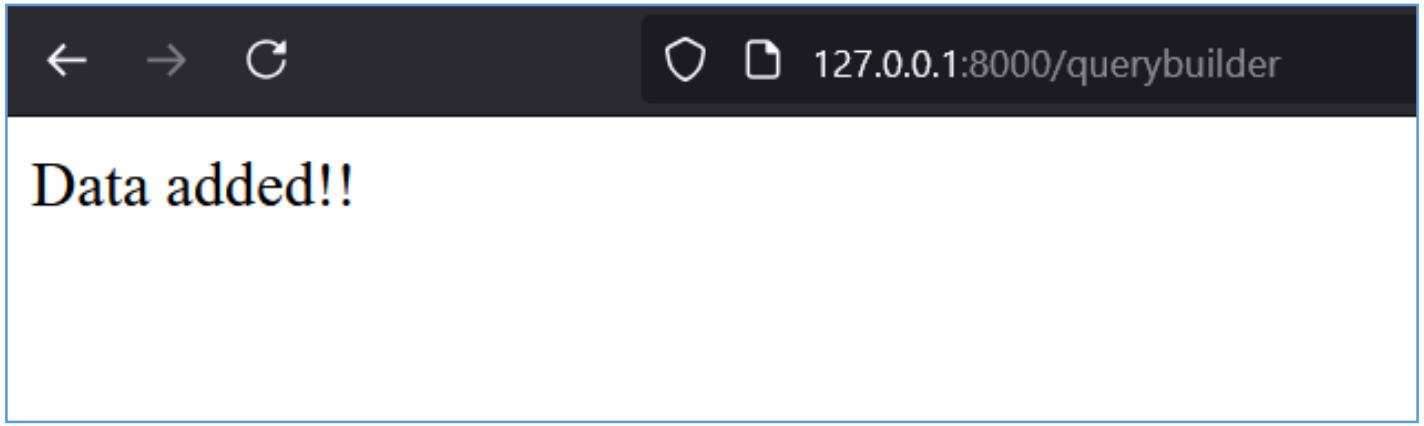
```php
// Model in Laravel
use App\Http\Controllers\StudentController2;
Route::get('students',[StudentController2::class,'getstudent']);

// Database Class Quiery Builder
use App\Http\Controllers\DbQueryBuilder;
Route::get('querybuilder',[DbQueryBuilder::class,'Query']);

// Eleouquent Model Query Builder
use App\Http\Controllers\EmodelQueryBuilder;
Route::get('emqb',[EmodelQueryBuilder::class,'equery']);
```

Cont. …


127.0.0.1:8000/querybuilder
well come to DB DB Query Builder
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


127.0.0.1:8000/querybuilder
Data added!!

Summary

Here's a summary of CRUD operations using Laravel's Database Query Builder in table format:

Operation	Description	Example
Create	Insert a single or multiple records into a database table.	`DB::table('users')->insert(['name' => 'John Doe', 'email' => 'johndoe@example.com']);`
	Insert multiple records.	`DB::table('users')->insert([['name' => 'Jane'], ['name' => 'Mark']]);`
	Insert and get the ID of the new record.	`$id = DB::table('users')->insertGetId(['name' => 'Alice']);`
Read	Retrieve all records from a table.	`$users = DB::table('users')->get();`
	Retrieve a single record.	`$user = DB::table('users')->where('id', 1)->first();`
	Retrieve specific columns.	`$names = DB::table('users')->pluck('name');`
	Filter records using `where`.	`$users = DB::table('users')->where('status', 'active')->get();`

Cont. …

Update	Update a record.	`DB::table('users')->where('id', 1)->update(['name' => 'Updated Name']);`
	Increment a numeric column.	`DB::table('posts')->where('id', 1)->increment('views');`
	Decrement a numeric column.	`DB::table('posts')->where('id', 1)->decrement('views', 2);`
Delete	Delete a specific record.	`DB::table('users')->where('id', 1)->delete();`
	Delete all records in a table.	`DB::table('users')->delete();`
	Truncate the table (delete all records and reset IDs).	`DB::table('users')->truncate();`

This table highlights CRUD functions commonly used with Laravel's Query Builder, with sample syntax for each operation.

Eloquent Model Query Builder

- Laravel's Eloquent ORM (Object-Relational Mapping) provides a powerful, expressive way to interact with databases by allowing developers to represent database tables as classes and rows as objects.
- With Eloquent, each model represents a table, where class methods translate into SQL commands. For instance, an User model corresponds to a users table, and interacting with this model feels intuitive, using simple syntax like ***$users = User::all()*** to retrieve all records, rather than needing to write complex SQL.

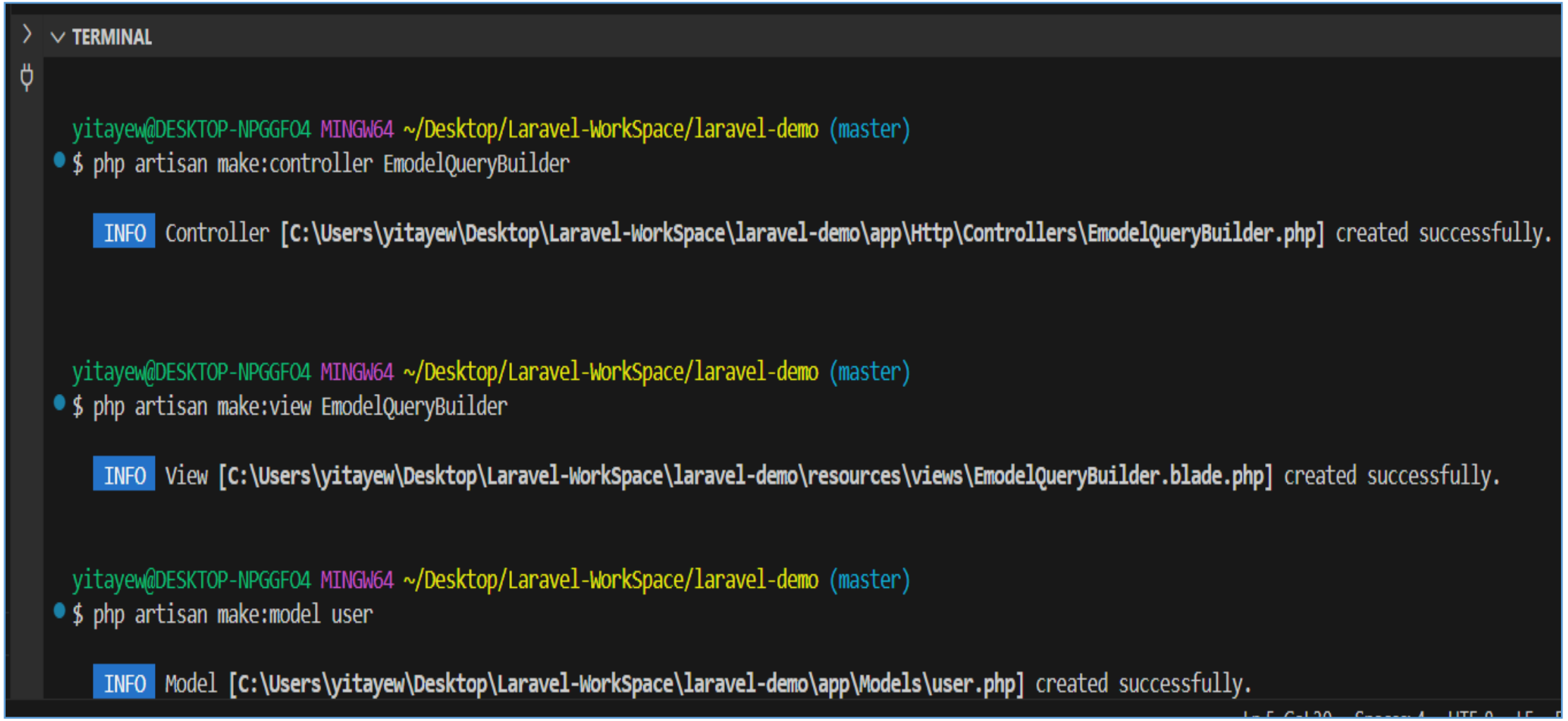
Cont. ...

- The Eloquent Query Builder takes Eloquent further by allowing more complex queries with chainable methods, making it easy to build and customize database requests without writing SQL.
- Eloquent supports a range of methods like **where**(), **orderBy**(), and **select**(), which can be combined in logical chains to create precise queries.

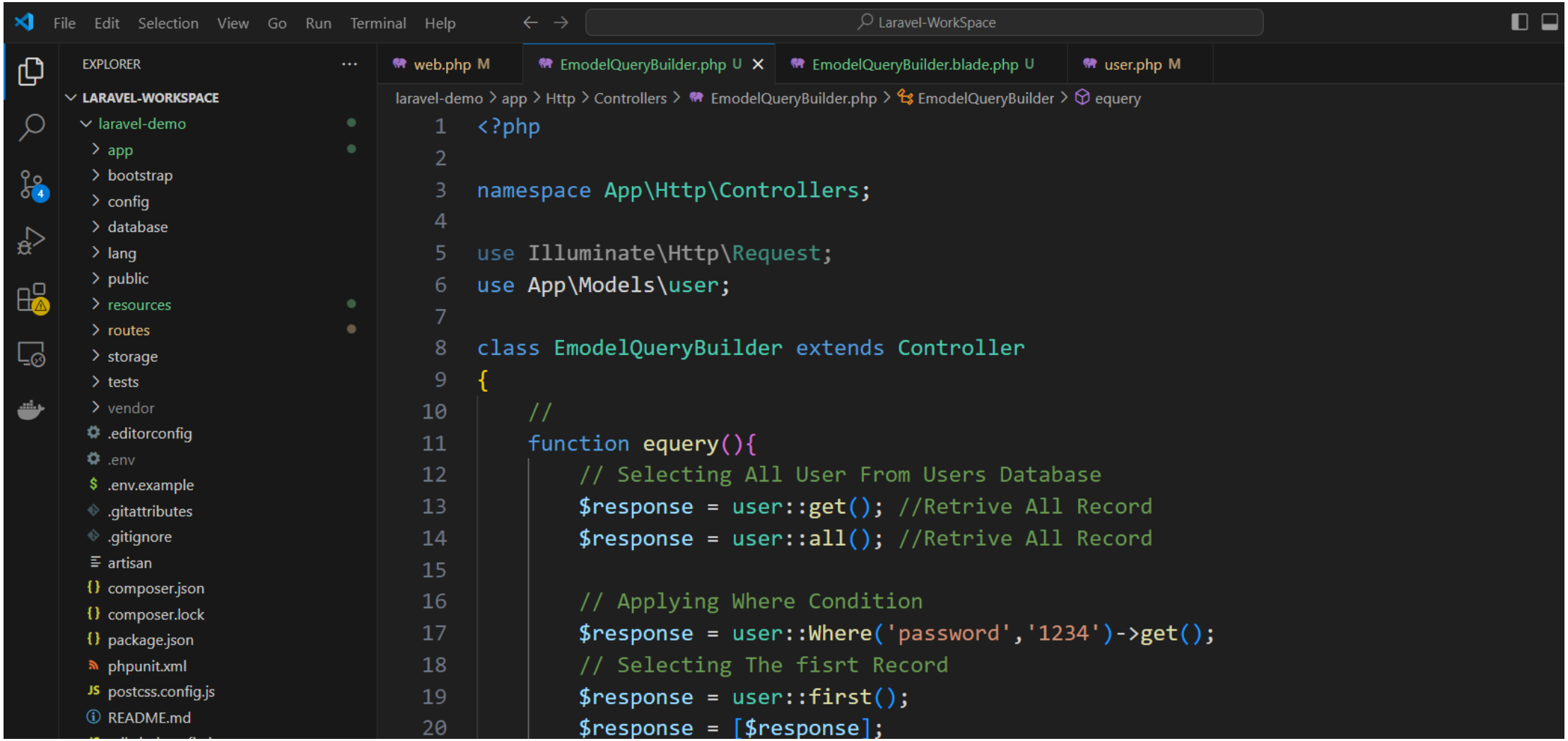
Cont. ...

- Relationships are another key feature, where methods like hasMany, belongsTo, and belongsToMany allow developers to define and query relationships between tables, reflecting the structure of relational databases directly in code.
- Eloquent also includes built-in support for CRUD operations, with methods like create(), update(), and delete() for simple database interactions.

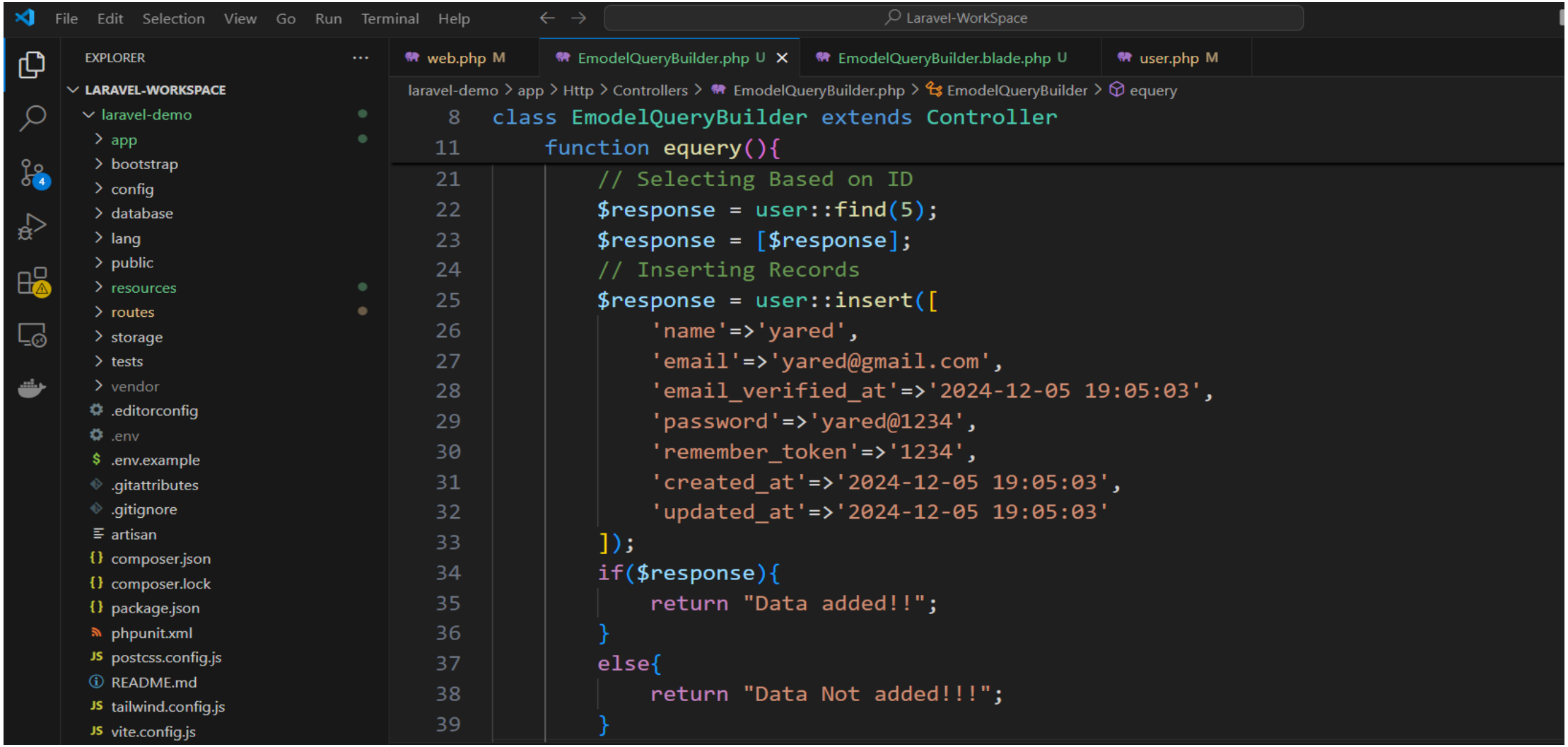
Creating(Controller, View and Model)



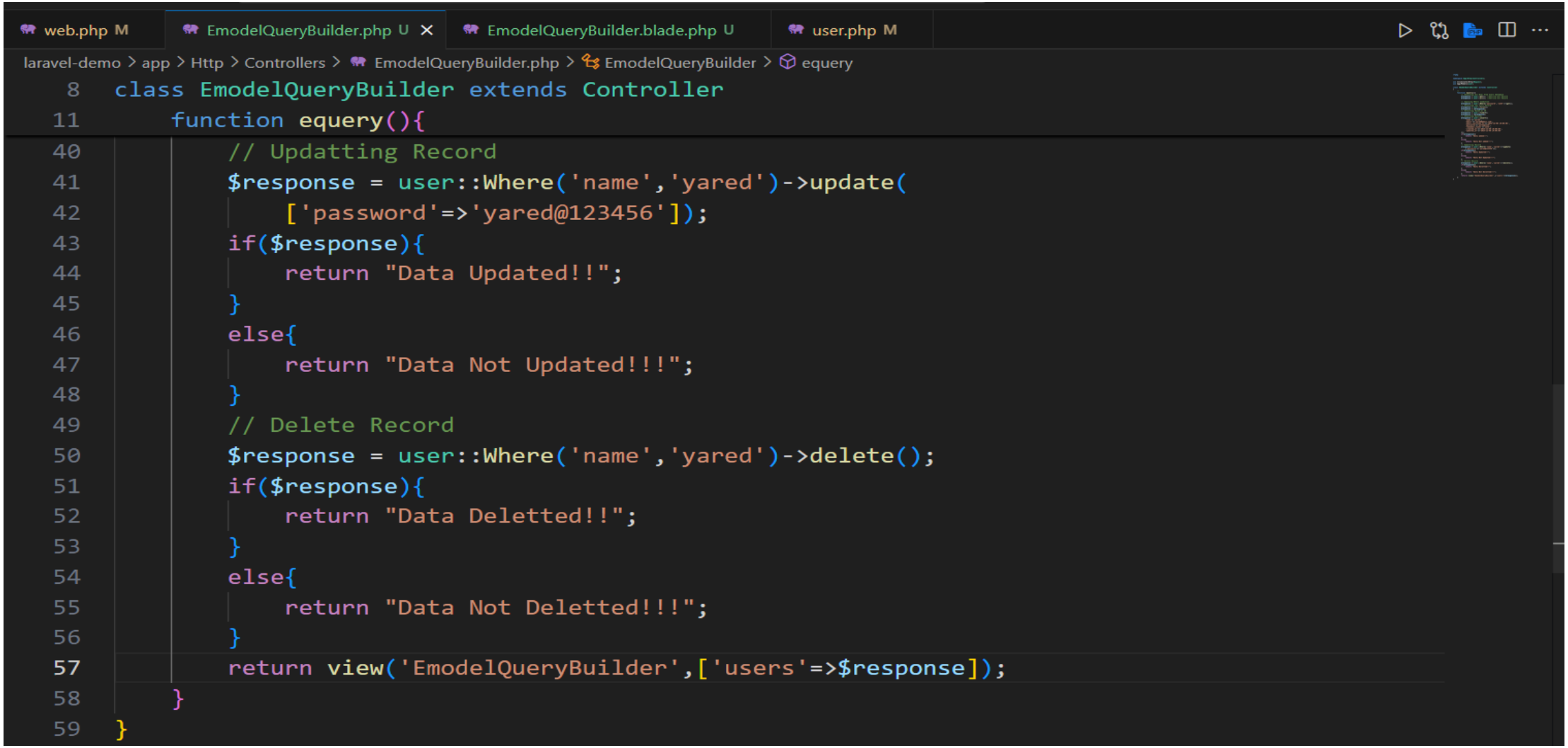
Controller


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
app
bootstrap
config
database
lang
public
resources
routes
storage
tests
vendor
.editorconfig
.env
.env.example
.gitattributes
.gitignore
artisan
composer.json
composer.lock
package.json
phpunit.xml
postcss.config.js
README.md
web.php M
EmodelQueryBuilder.php U
EmodelQueryBuilder.blade.php U
user.php M
laravel-demo > app > Http > Controllers > EmodelQueryBuilder.php > EmodelQueryBuilder > equery
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\user;
7
8 class EmodelQueryBuilder extends Controller
9 {
10 //
11 function equery(){
12 // Selecting All User From Users Database
13 $response = user::get(); //Retrive All Record
14 $response = user::all(); //Retrive All Record
15
16 // Applying Where Condition
17 $response = user::Where('password','1234')->get();
18 // Selecting The fisrt Record
19 $response = user::first();
20 $response = [$response];

Cont. ...


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
app
bootstrap
config
database
lang
public
resources
routes
storage
tests
vendor
.editorconfig
.env
.env.example
.gitattributes
.gitignore
artisan
composer.json
composer.lock
package.json
phpunit.xml
postcss.config.js
README.md
tailwind.config.js
vite.config.js
web.php M
EmodelQueryBuilder.php U
EmodelQueryBuilder.blade.php U
user.php M
laravel-demo > app > Http > Controllers > EmodelQueryBuilder.php > EmodelQueryBuilder > equery
8 class EmodelQueryBuilder extends Controller
11 function equery(){
21 // Selecting Based on ID
22 $response = user::find(5);
23 $response = [$response];
24 // Inserting Records
25 $response = user::insert([
26 'name'=>'yared',
27 'email'=>'yared@gmail.com',
28 'email_verified_at'=>'2024-12-05 19:05:03',
29 'password'=>'yared@1234',
30 'remember_token'=>'1234',
31 'created_at'=>'2024-12-05 19:05:03',
32 'updated_at'=>'2024-12-05 19:05:03'
33]);
34 if($response){
35 return "Data added!!";
36 }
37 else{
38 return "Data Not added!!!";
39 }

Cont. ...


web.php M
EmodelQueryBuilder.php U
EmodelQueryBuilder.blade.php U
user.php M
laravel-demo > app > Http > Controllers > EmodelQueryBuilder.php > EmodelQueryBuilder > equery
8 class EmodelQueryBuilder extends Controller
11 function equery(){
40 // Updatting Record
41 $response = user::Where('name','yared')->update(
42 ['password'=>'yared@123456']);
43 if($response){
44 return "Data Updated!!";
45 }
46 else{
47 return "Data Not Updated!!!";
48 }
49 // Delete Record
50 $response = user::Where('name','yared')->delete();
51 if($response){
52 return "Data Deletted!!";
53 }
54 else{
55 return "Data Not Deletted!!!";
56 }
57 return view('EmodelQueryBuilder',['users'=>$response]);
58 }
59 }

View

web.php M | EmodelQueryBuilder.php U | EmodelQueryBuilder.blade.php U | user.php M

laravel-demo > resources > views > EmodelQueryBuilder.blade.php > div > table > tr > th

```
<div>
    <!-- People find pleasure in different ways. I find it in keeping my mind clear. - Marcus
    <h1>Well Come To Query Builder Using Emodel</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>
```

Model

web.php M | EmodelQueryBuilder.php U | EmodelQueryBuilder.blade.php U | user.php M

laravel-demo > app > Models > user.php > ...

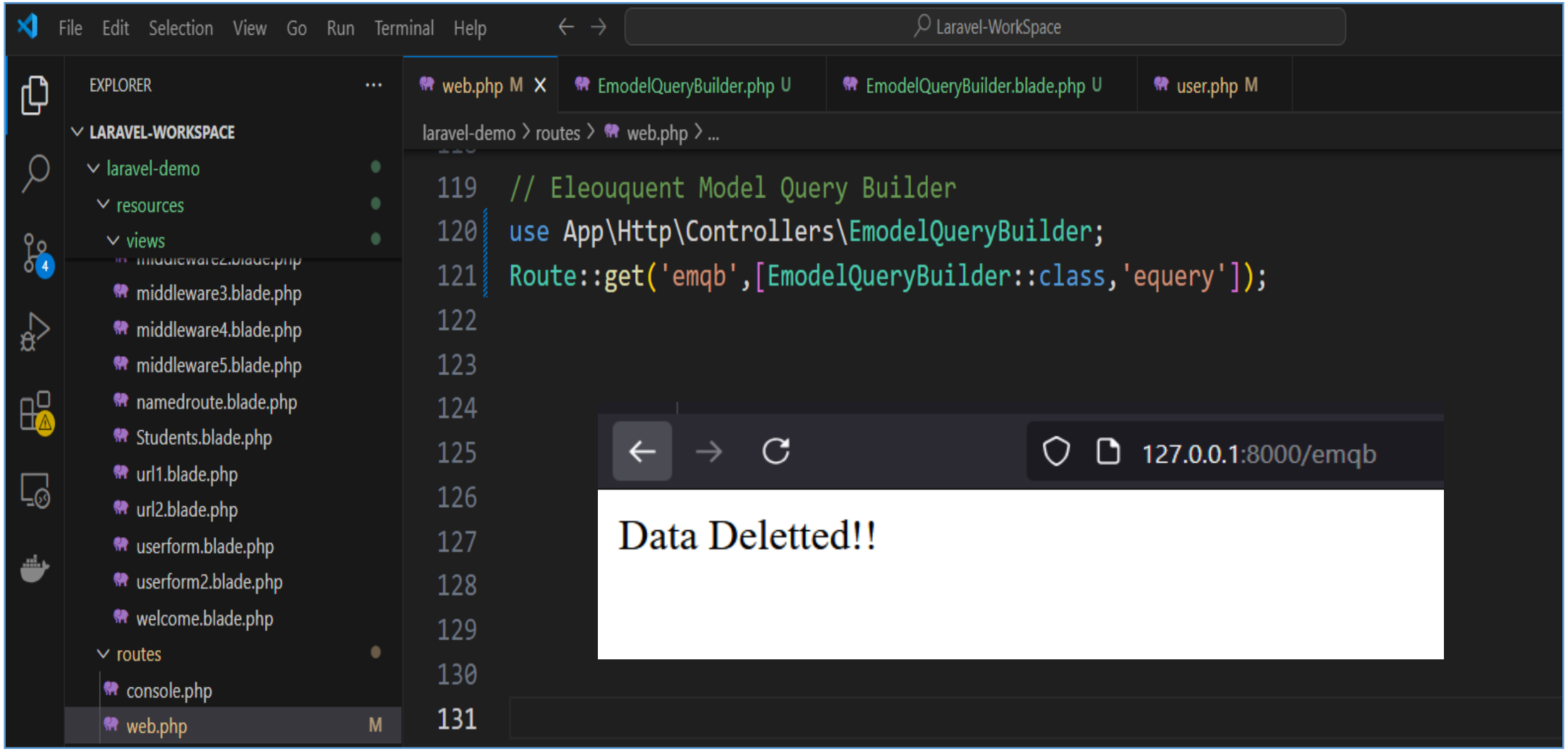
```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class user extends Model
{
    //
    public $timestamps=false;
}
```

Route


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
EmodelQueryBuilder.php U
EmodelQueryBuilder.blade.php U
user.php M
laravel-demo > routes > web.php > ...
119 // Eleouquent Model Query Builder
120 use App\Http\Controllers\EmodelQueryBuilder;
121 Route::get('emqb',[EmodelQueryBuilder::class,'equery']);
122
123
124
125
126
127
128
129
130
131
127.0.0.1:8000/emqb
Data Deletted!!

Summary

Here's a table summarizing CRUD operations using Laravel's Eloquent Model and Query Builder methods:

Operation	Eloquent Model	Query Builder	Description
Create	`$user = new User();` `$user->name = 'John';` `$user->save();` or `User::create(['name' => 'John']);`	`DB::table('users')->insert(['name' => 'John']);`	Inserts new records into the database.
Read	`User::all();` `User::find(1);` `User::where('status', 'active')->get();`	`DB::table('users')->get();` `DB::table('users')->where('status', 'active')->first();`	Retrieves records from the database.

Cont. ...

Update	`$user = User::find(1);` `$user->name = 'Jane';` `$user->save();` or `User::where('id', 1)->update(['name' => 'Jane']);`	`DB::table('users')->where('id', 1)->update(['name' => 'Jane']);`	Updates existing records in the database.
Delete	`$user = User::find(1);` `$user->delete();` or `User::where('id', 1)->delete();`	`DB::table('users')->where('id', 1)->delete();`	Deletes records from the database.

This table shows how to perform basic CRUD operations with both Eloquent Models and Query Builder methods in Laravel. Eloquent provides an object-oriented approach, while Query Builder is a more flexible option for more complex queries.

Eloquent ORM vs Query Builder in Laravel

- Laravel provides **two primary** ways to interact with the database: **Eloquent ORM** and **Query Builder**. While both serve the purpose of querying and manipulating data, they differ in their **approach**, **features**, and **use cases**.

Eloquent ORM

Eloquent ORM

1. **Definition:**

   Eloquent is Laravel's **Object-Relational Mapping (ORM)**, which provides an object-oriented and model-centric approach to interact with the database. It maps database tables to PHP classes and rows to objects, enabling developers to use models for querying and manipulating data.

2. **Features:**

   - **Model-Based**: Each database table corresponds to a model.
   - **Active Record Implementation**: Data is manipulated through model instances.
   - **Relationships**: Supports defining and working with database relationships like one-to-one, one-to-many, and many-to-many.
   - **Eloquent Methods**: Provides methods like `find()`, `save()`, and `delete()`.

Example

3. **Example:**

   Fetching all users:

```php
$users = User::all();
```

   Fetching a user by ID:

```php
$user = User::find(1);
```

4. **Best For:**
   - Applications with complex relationships.
   - When leveraging Laravel's built-in ORM features like relationships, scopes, and mutators.

Query Builder

Query Builder

1. **Definition**:

   Query Builder is a more **direct and procedural approach** to interact with the database. It allows developers to build database queries programmatically without relying on models or object-oriented design.

2. **Features**:

   - **Fluent API**: Methods are chainable and readable.
   - **No Models Required**: Operates directly on database tables.
   - **Highly Flexible**: Can be used for custom and complex SQL queries.
   - **Parameter Binding**: Helps prevent SQL injection.

Example

3. **Example:**

   Fetching all users:

```php
$users = DB::table('users')->get();
```

   Fetching a user by ID:

```php
$user = DB::table('users')->where('id', 1)->first();
```

4. **Best For:**

   - Lightweight operations or raw SQL-like queries.
   - When no model is required, e.g., for custom tables or views.

Difference

Key Differences

Feature	Eloquent ORM	Query Builder
Approach	Object-Oriented (Model-centric)	Procedural (Query-centric)
Ease of Use	Easier for complex relationships	Flexible for custom SQL queries
Performance	May have more overhead due to models	Generally faster for simple queries
Relationships	Built-in support (e.g., `hasOne`, `belongsTo`)	Manual implementation required
Use Case	CRUD operations with defined models	Custom queries or lightweight tasks
Code Readability	Cleaner and more descriptive	Explicit but less structured

When to use

When to Use

- **Eloquent ORM:**
  Use when working with structured data models and their relationships. Ideal for most Laravel applications.
- **Query Builder:**
  Use for raw queries, dynamic SQL, or when working with tables that don't have an associated model.

Both Eloquent ORM and Query Builder are powerful, and the choice depends on the specific needs of your application. Often, developers combine both for optimal results.

Route Methods in Laravel

In Laravel, route methods are used to define how your application responds to various HTTP requests. Here's a breakdown of common route methods, with descriptions and examples:

Route Method	Description	Example
GET	Retrieves data without making changes, typically used for displaying views or data.	`Route::get('/users', [UserController::class, 'index']);`
POST	Used to submit data to the server, often for creating new resources.	`Route::post('/users', [UserController::class, 'store']);`
PUT	Replaces an entire resource, typically used for updating existing records.	`Route::put('/users/{id}', [UserController::class, 'update']);`
PATCH	Partially updates a resource, often for modifying specific fields of an existing record.	`Route::patch('/users/{id}', [UserController::class, 'updatePartial']);`

Cont. ...

OPTIONS	Describes communication options for the target resource, commonly used for CORS preflight requests in APIs.	`Route::options('/users', function () { return response()->json(['options' => 'GET, POST']); });`
ANY	Responds to any HTTP method for the specified route.	`Route::any('/users', function () { /* logic */ });`
MATCH	Allows you to specify an array of HTTP methods to respond to on a single route.	`Route::match(['get', 'post'], '/users', [UserController::class, 'handleRequest']);`

Examples of Route Methods in a Controller Context

1. **GET** - Display all users:

```php
Route::get('/users', [UserController::class, 'index']);
```

2. **POST** - Add a new user:

```php
Route::post('/users', [UserController::class, 'store']);
```

3. **PUT** - Update an existing user:

```php
Route::put('/users/{id}', [UserController::class, 'update']);
```

Cont. ...

4. **PATCH** - Partially update a user's info:

```php
Route::patch('/users/{id}', [UserController::class, 'updatePartial']);
```

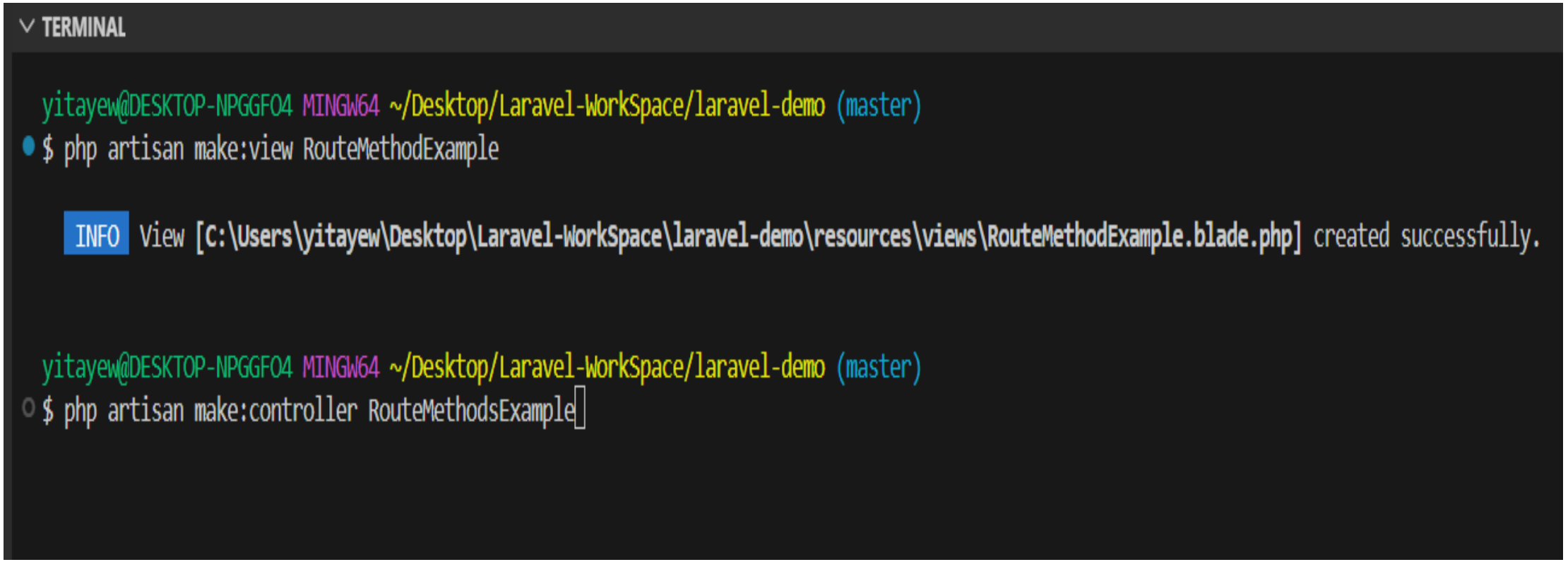
5. **DELETE** - Delete a user:

```php
Route::delete('/users/{id}', [UserController::class, 'destroy']);
```

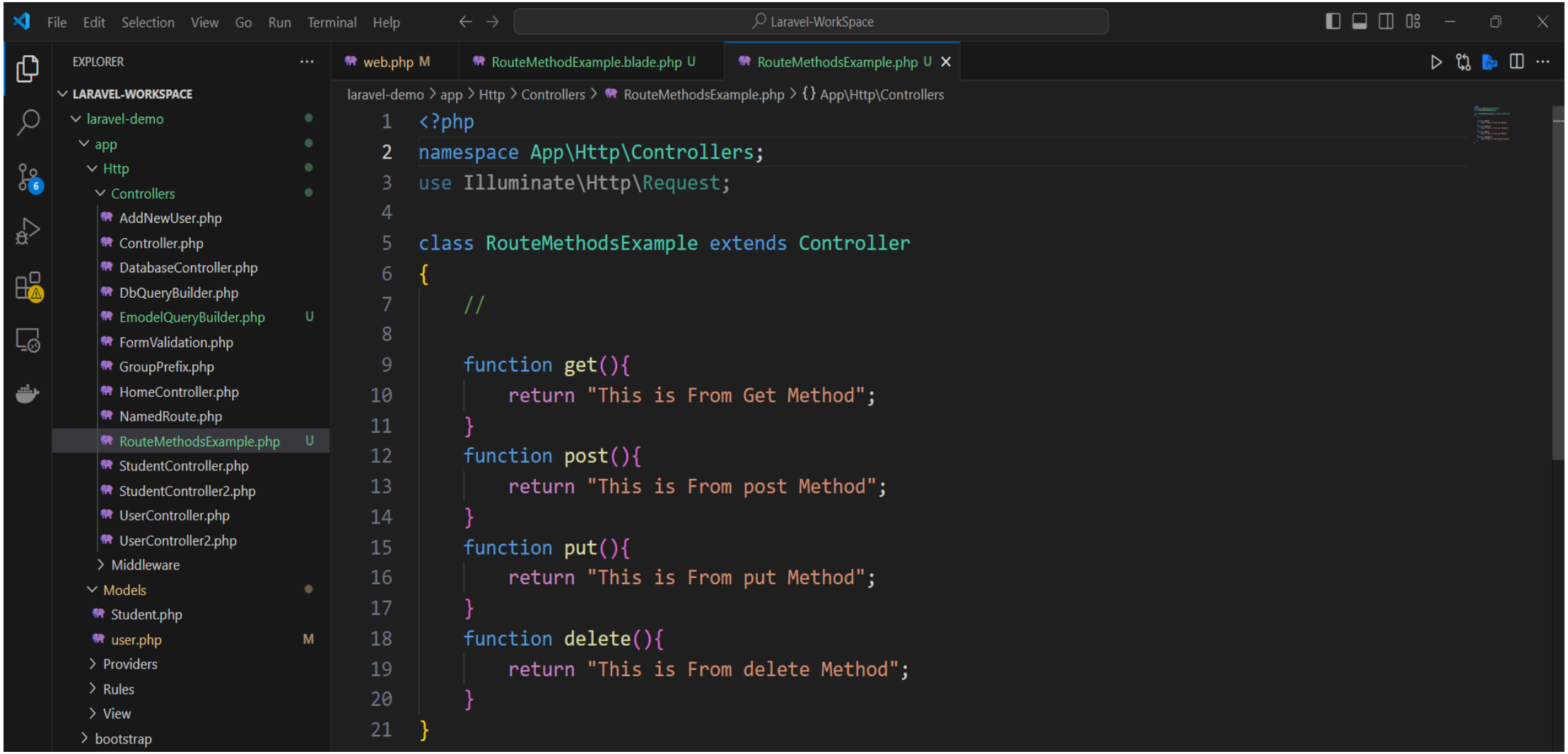
6. **ANY** - Accept any HTTP request type:

```php
Route::any('/users', [UserController::class, 'handleAny']);
```

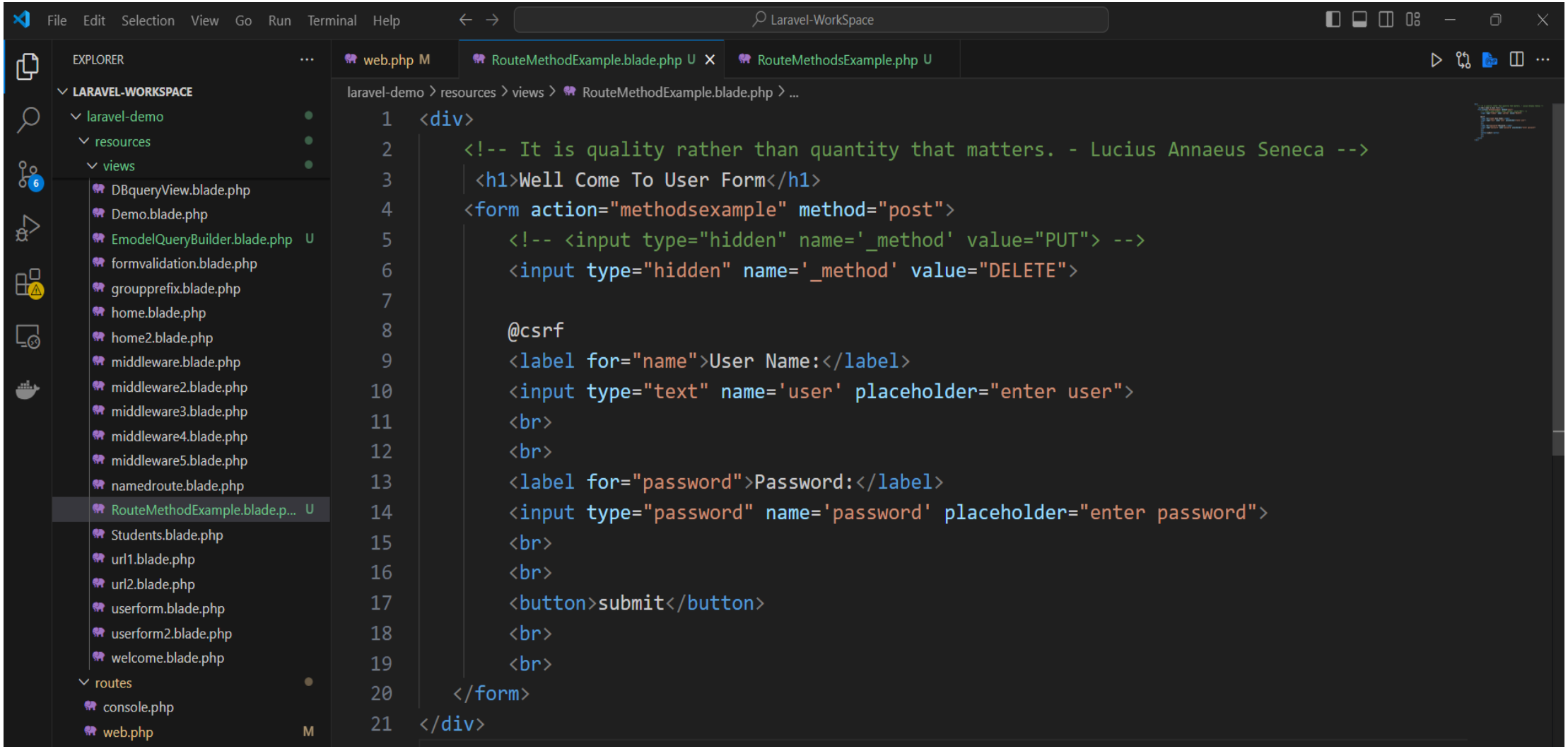
Examples


TERMINAL
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view RouteMethodExample
INFO View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\RouteMethodExample.blade.php] created successfully.
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:controller RouteMethodsExample

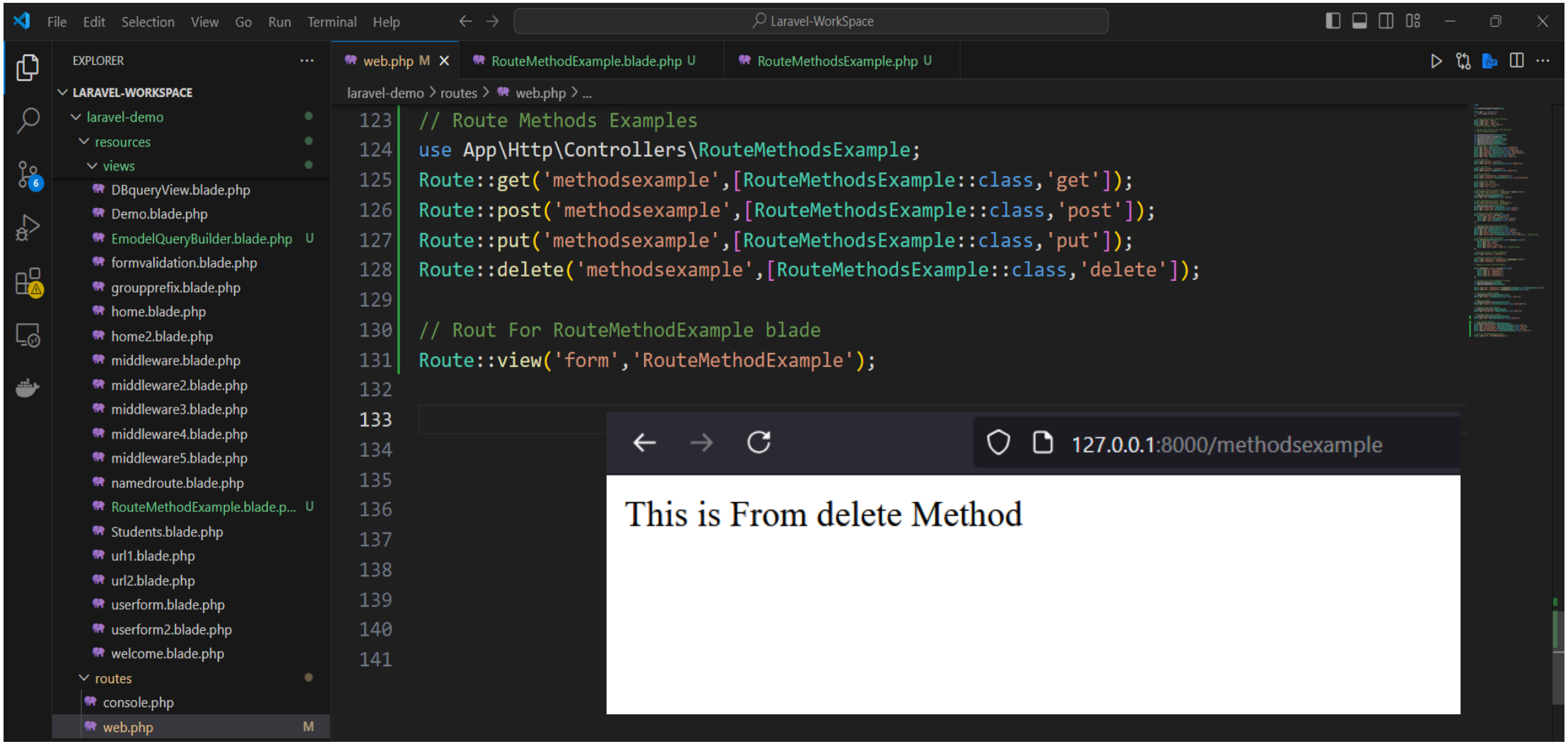
Controller (RouteMethodExample)



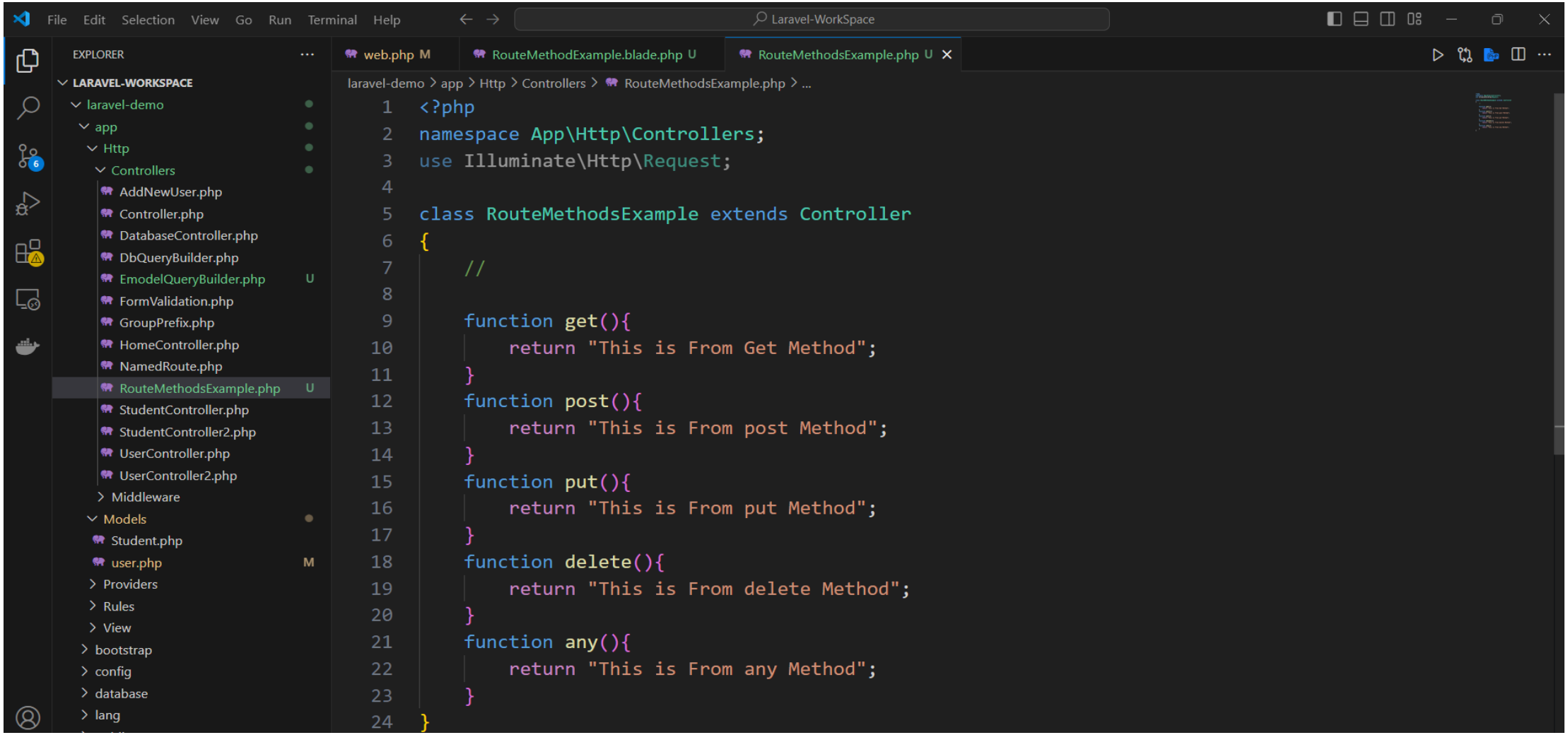
View (RouteMethodExample)



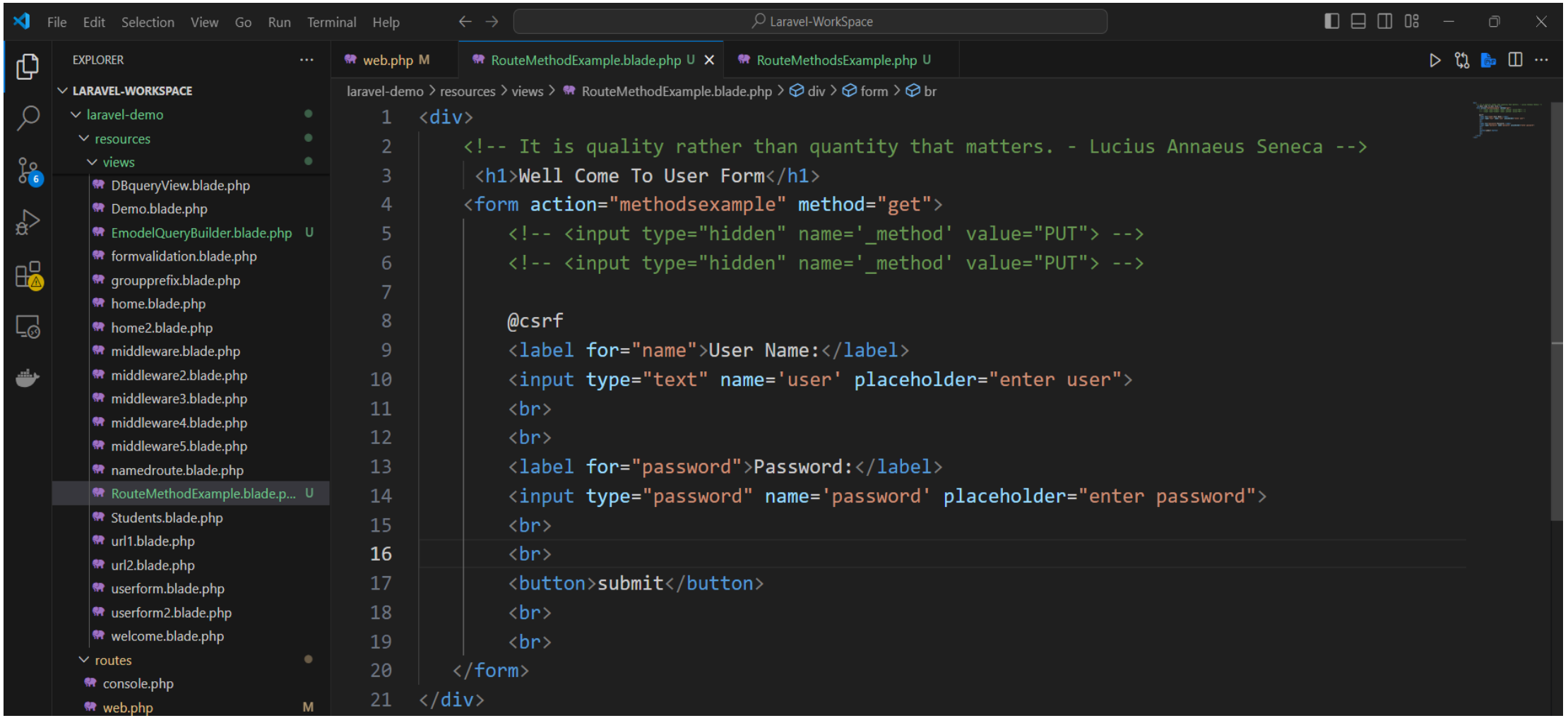
Route


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
DBqueryView.blade.php
Demo.blade.php
EmodelQueryBuilder.blade.php U
formvalidation.blade.php
groupprefix.blade.php
home.blade.php
home2.blade.php
middleware.blade.php
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RouteMethodExample.blade.p... U
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
RouteMethodExample.blade.php U
RouteMethodsExample.php U
laravel-demo > routes > web.php > ...
123 // Route Methods Examples
124 use App\Http\Controllers\RouteMethodsExample;
125 Route::get('methodsexample',[RouteMethodsExample::class,'get']);
126 Route::post('methodsexample',[RouteMethodsExample::class,'post']);
127 Route::put('methodsexample',[RouteMethodsExample::class,'put']);
128 Route::delete('methodsexample',[RouteMethodsExample::class,'delete']);
129
130 // Rout For RouteMethodExample blade
131 Route::view('form','RouteMethodExample');
132
133
134
135
136
137
138
139
140
141
127.0.0.1:8000/methodsexample
This is From delete Method

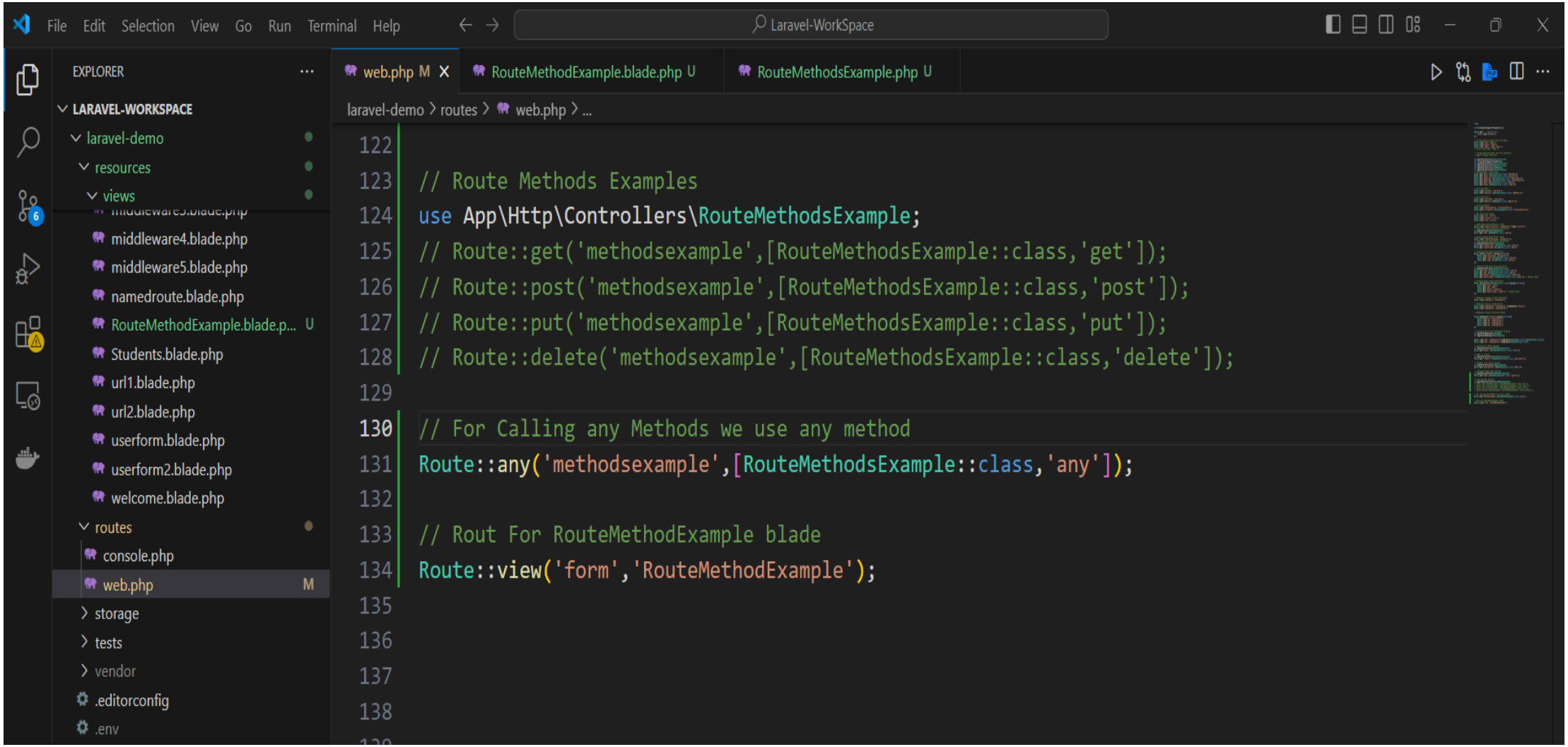
Cont. … any Method



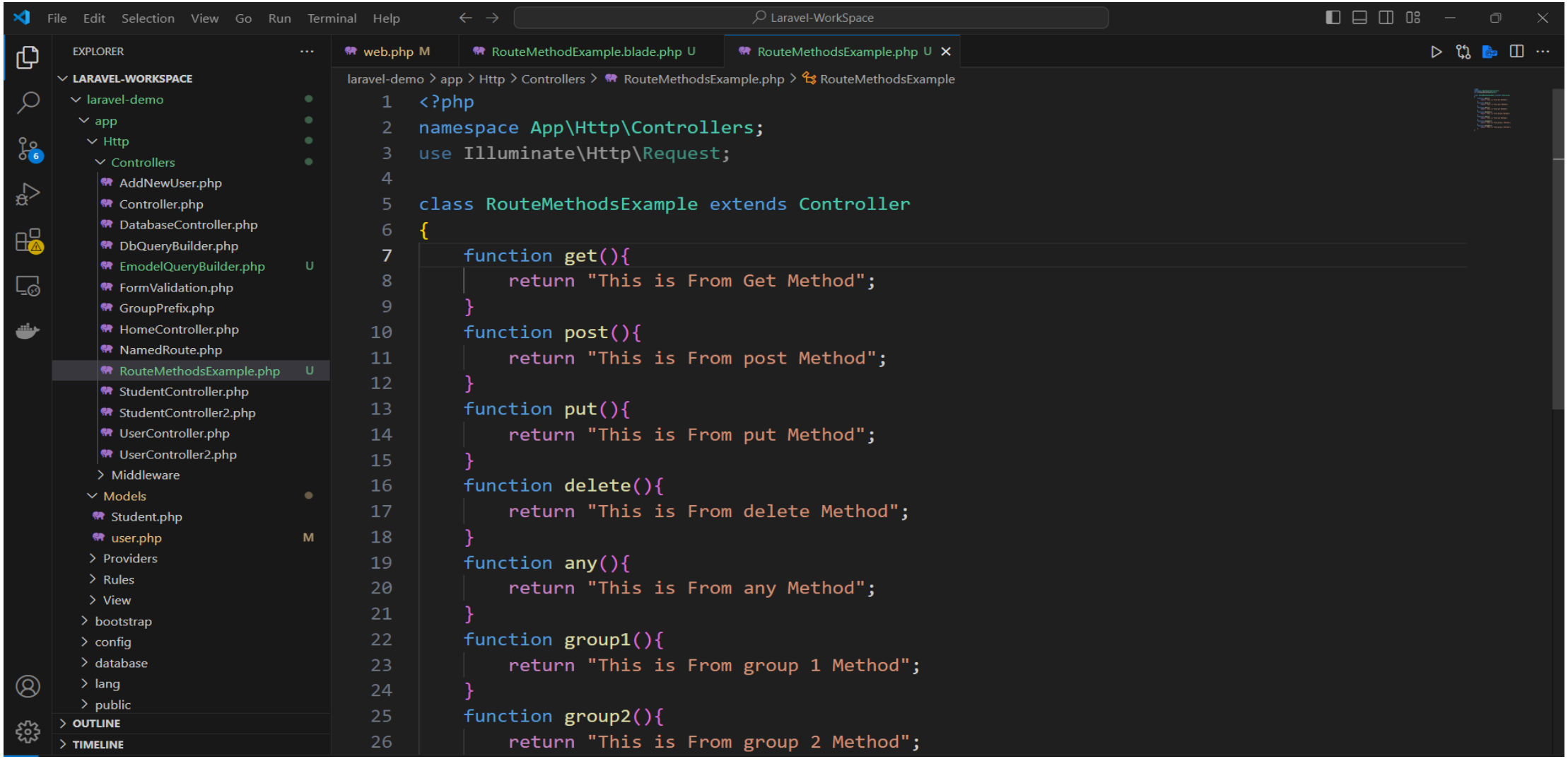
Cont. ...


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
DBqueryView.blade.php
Demo.blade.php
EmodelQueryBuilder.blade.php U
formvalidation.blade.php
groupprefix.blade.php
home.blade.php
home2.blade.php
middleware.blade.php
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RouteMethodExample.blade.p... U
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
RouteMethodExample.blade.php U
RouteMethodsExample.php U
laravel-demo > resources > views > RouteMethodExample.blade.php > div > form > br
1
2 <!-- It is quality rather than quantity that matters. - Lucius Annaeus Seneca -->
3 <h1>Well Come To User Form</h1>
4 <form action="methodsexample" method="get">
5 <!-- <input type="hidden" name='_method' value="PUT"> -->
6 <!-- <input type="hidden" name='_method' value="PUT"> -->
7
8 @csrf
9 <label for="name">User Name:</label>
10 <input type="text" name='user' placeholder="enter user">
11
12
13 <label for="password">Password:</label>
14 <input type="password" name='password' placeholder="enter password">
15
16
17 <button>submit</button>
18
19
20 </form>
21
OUTLINE
TIMELINE

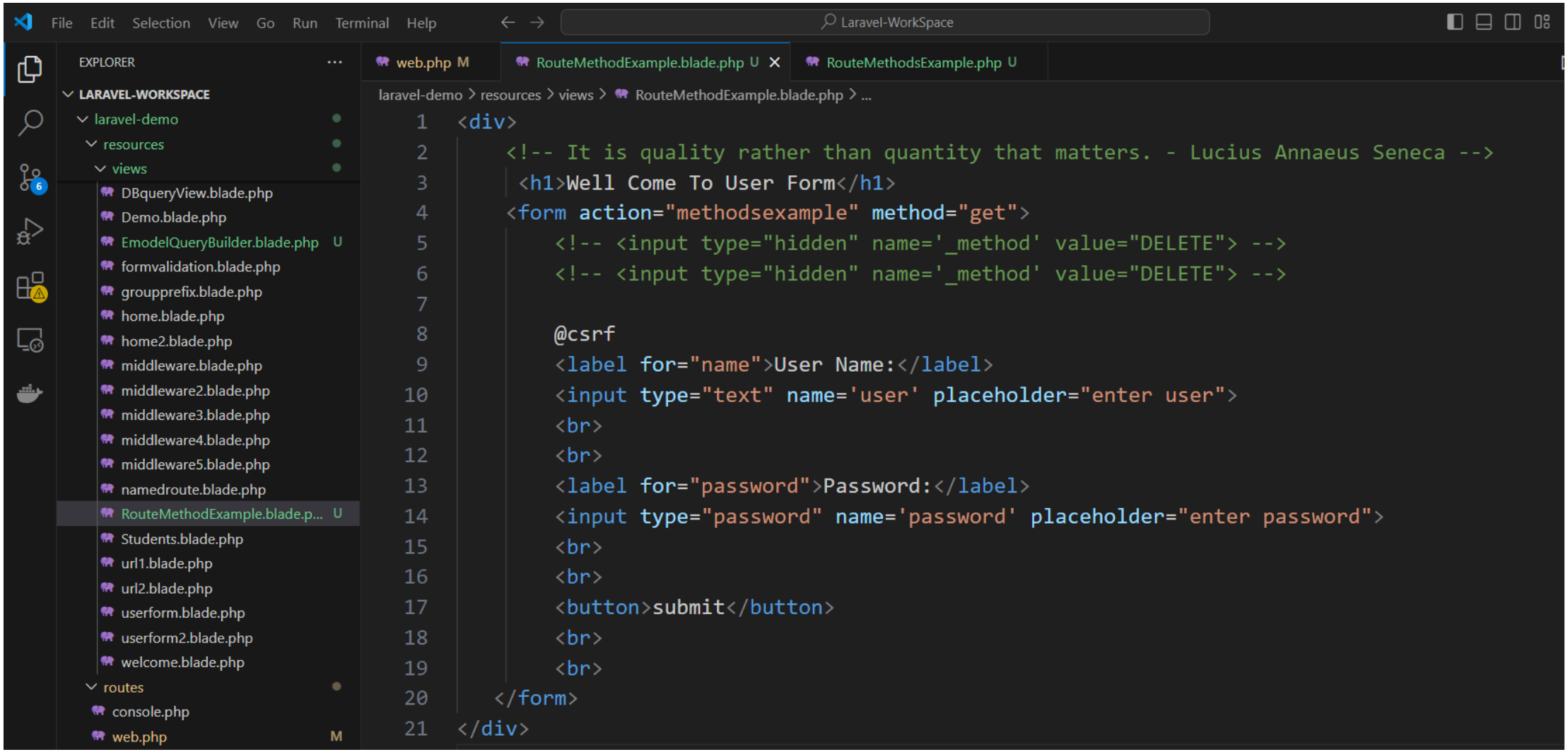
Cont. ...


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RouteMethodExample.blade.p... U
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
storage
tests
vendor
.editorconfig
.env
web.php M
RouteMethodExample.blade.php U
RouteMethodsExample.php U
laravel-demo > routes > web.php > ...
122
123 // Route Methods Examples
124 use App\Http\Controllers\RouteMethodsExample;
125 // Route::get('methodsexample',[RouteMethodsExample::class,'get']);
126 // Route::post('methodsexample',[RouteMethodsExample::class,'post']);
127 // Route::put('methodsexample',[RouteMethodsExample::class,'put']);
128 // Route::delete('methodsexample',[RouteMethodsExample::class,'delete']);
129
130 // For Calling any Methods we use any method
131 Route::any('methodsexample',[RouteMethodsExample::class,'any']);
132
133 // Rout For RouteMethodExample blade
134 Route::view('form','RouteMethodExample');
135
136
137
138

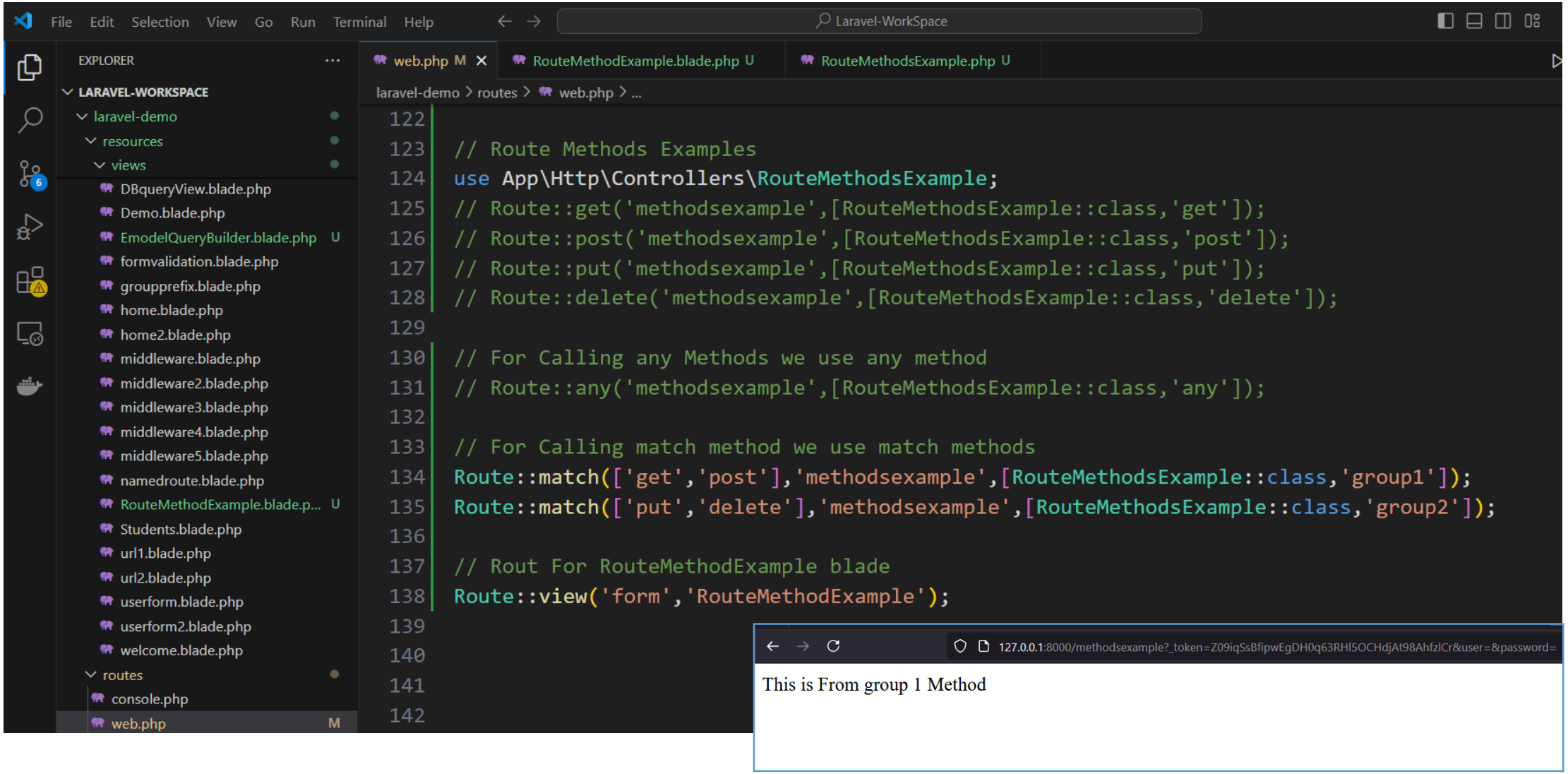
Cont. ... match method



Cont. ...

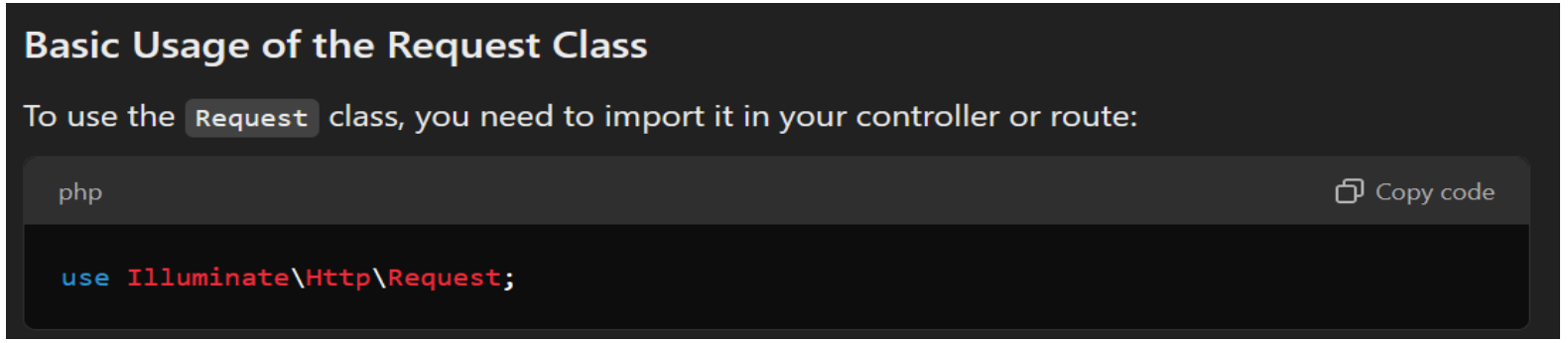

File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
DBqueryView.blade.php
Demo.blade.php
EmodelQueryBuilder.blade.php U
formvalidation.blade.php
groupprefix.blade.php
home.blade.php
home2.blade.php
middleware.blade.php
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RouteMethodExample.blade.p... U
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
RouteMethodExample.blade.php U
RouteMethodsExample.php U
laravel-demo > resources > views > RouteMethodExample.blade.php > ...
1
2 <!-- It is quality rather than quantity that matters. - Lucius Annaeus Seneca -->
3 <h1>Well Come To User Form</h1>
4 <form action="methodsexample" method="get">
5 <!-- <input type="hidden" name='_method' value="DELETE"> -->
6 <!-- <input type="hidden" name='_method' value="DELETE"> -->
7
8 @csrf
9 <label for="name">User Name:</label>
10 <input type="text" name='user' placeholder="enter user">
11
12
13 <label for="password">Password:</label>
14 <input type="password" name='password' placeholder="enter password">
15
16
17 <button>submit</button>
18
19
20 </form>
21
12/17/2024
Laravel Framework
45

Cont. ...

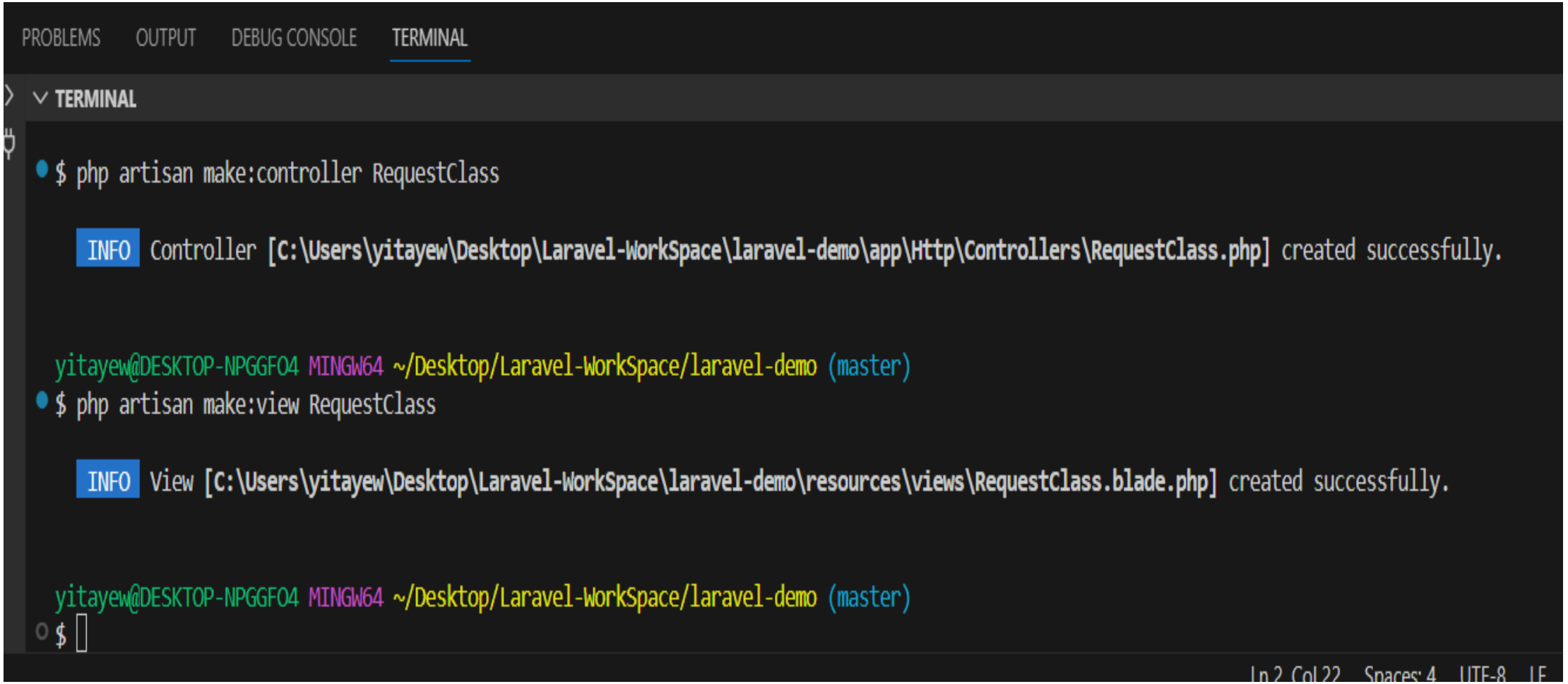

File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
DBqueryView.blade.php
Demo.blade.php
EmodelQueryBuilder.blade.php U
formvalidation.blade.php
groupprefix.blade.php
home.blade.php
home2.blade.php
middleware.blade.php
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RouteMethodExample.blade.p... U
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
RouteMethodExample.blade.php U
RouteMethodsExample.php U
laravel-demo > routes > web.php > ...
122
123 // Route Methods Examples
124 use App\Http\Controllers\RouteMethodsExample;
125 // Route::get('methodsexample',[RouteMethodsExample::class,'get']);
126 // Route::post('methodsexample',[RouteMethodsExample::class,'post']);
127 // Route::put('methodsexample',[RouteMethodsExample::class,'put']);
128 // Route::delete('methodsexample',[RouteMethodsExample::class,'delete']);
129
130 // For Calling any Methods we use any method
131 // Route::any('methodsexample',[RouteMethodsExample::class,'any']);
132
133 // For Calling match method we use match methods
134 Route::match(['get','post'],'methodsexample',[RouteMethodsExample::class,'group1']);
135 Route::match(['put','delete'],'methodsexample',[RouteMethodsExample::class,'group2']);
136
137 // Rout For RouteMethodExample blade
138 Route::view('form','RouteMethodExample');
139
140
141
142
127.0.0.1:8000/methodsexample?_token=Z09iqSsBfipwEgDH0q63RHI5OCHdjAt98AhfzlCr&user=&password=
This is From group 1 Method

HTTP Request Class

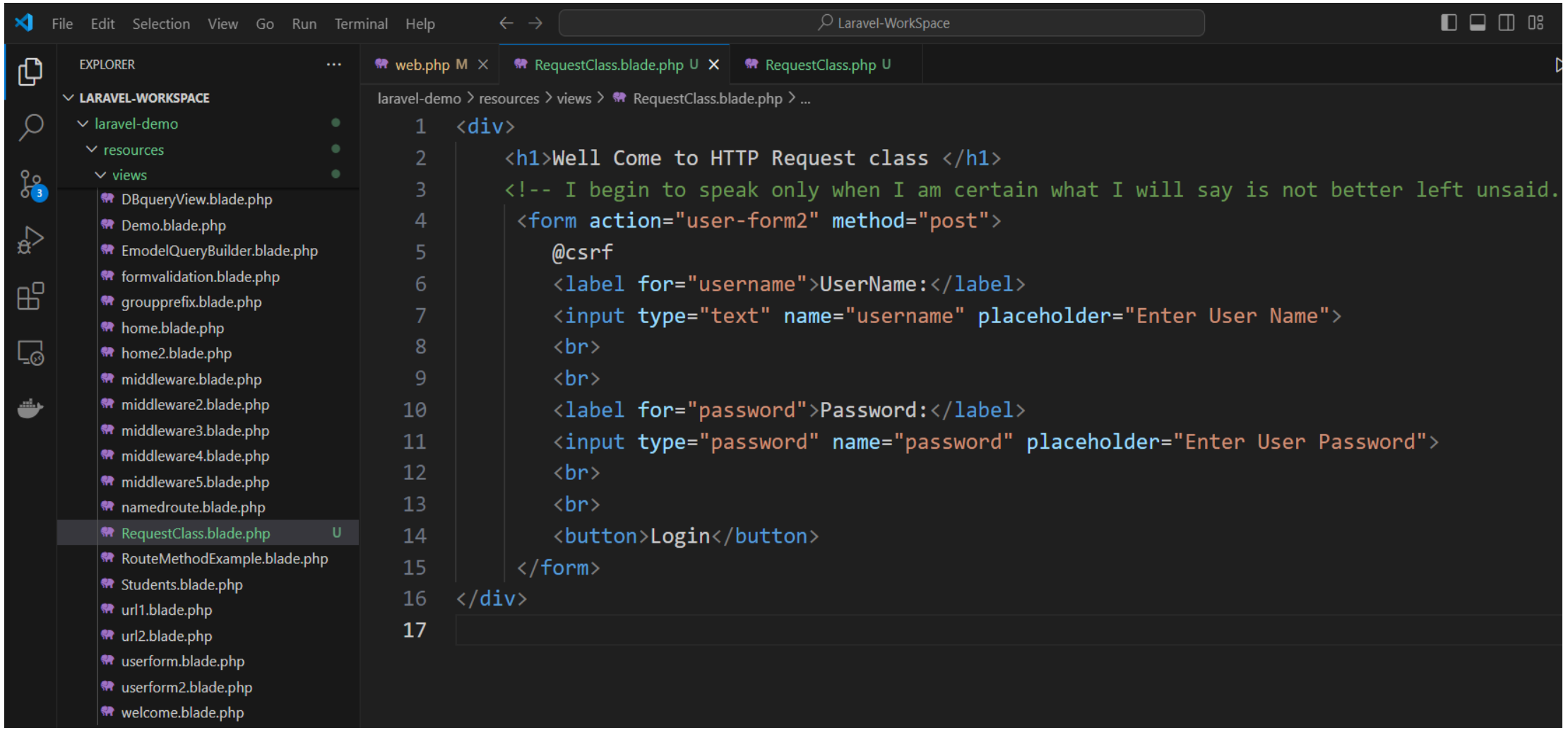
- The HTTP Request class in Laravel is a powerful component used to interact with **incoming HTTP requests**. It provides methods to retrieve input data, query parameters, cookies, files, and more.
- This class is automatically injected into controller methods and can be used to access and validate request data conveniently.



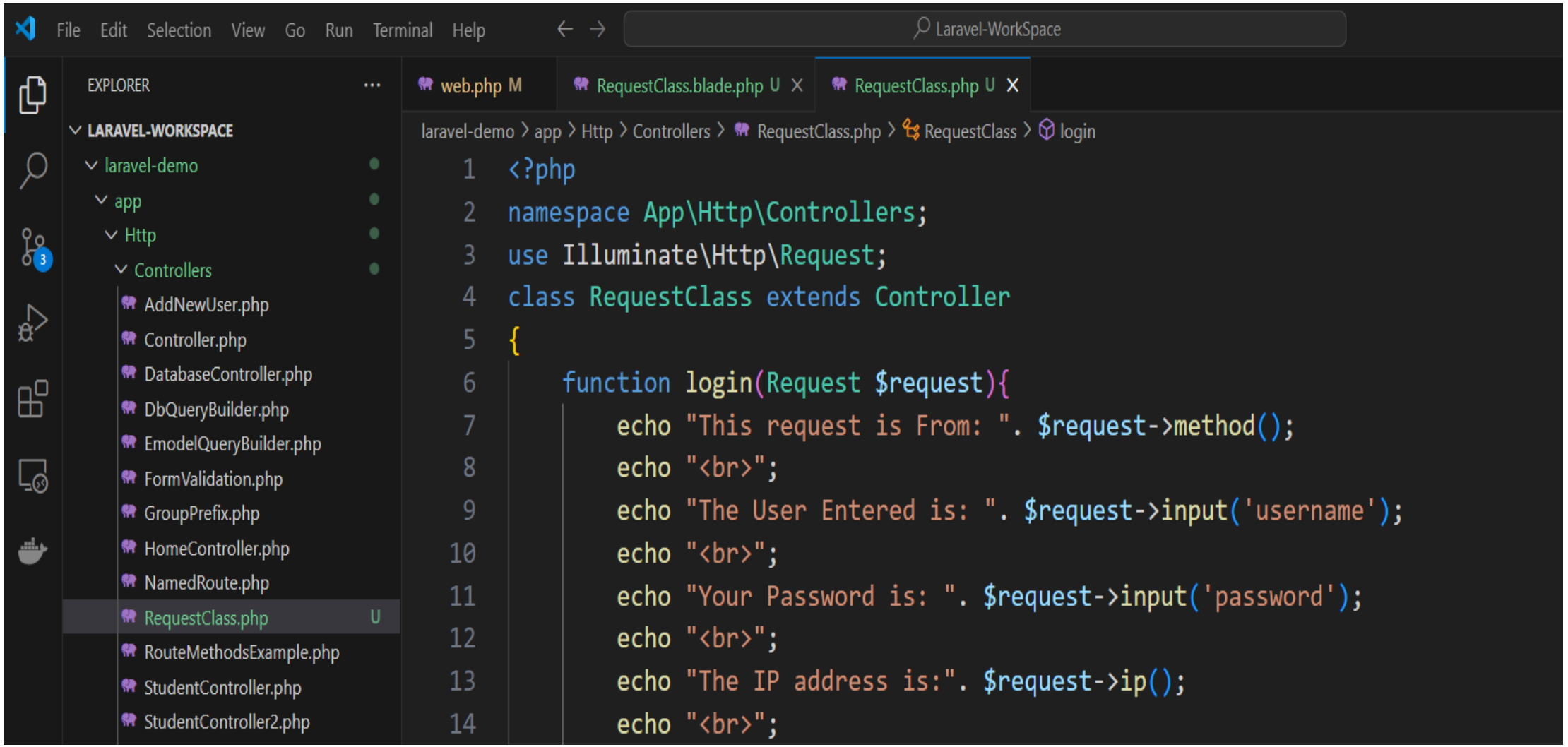
Example


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
TERMINAL
$ php artisan make:controller RequestClass
INFO Controller [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\app\Http\Controllers\RequestClass.php] created successfully.
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$ php artisan make:view RequestClass
INFO View [C:\Users\yitayew\Desktop\Laravel-WorkSpace\laravel-demo\resources\views\RequestClass.blade.php] created successfully.
yitayew@DESKTOP-NPGGFO4 MINGW64 ~/Desktop/Laravel-WorkSpace/laravel-demo (master)
$

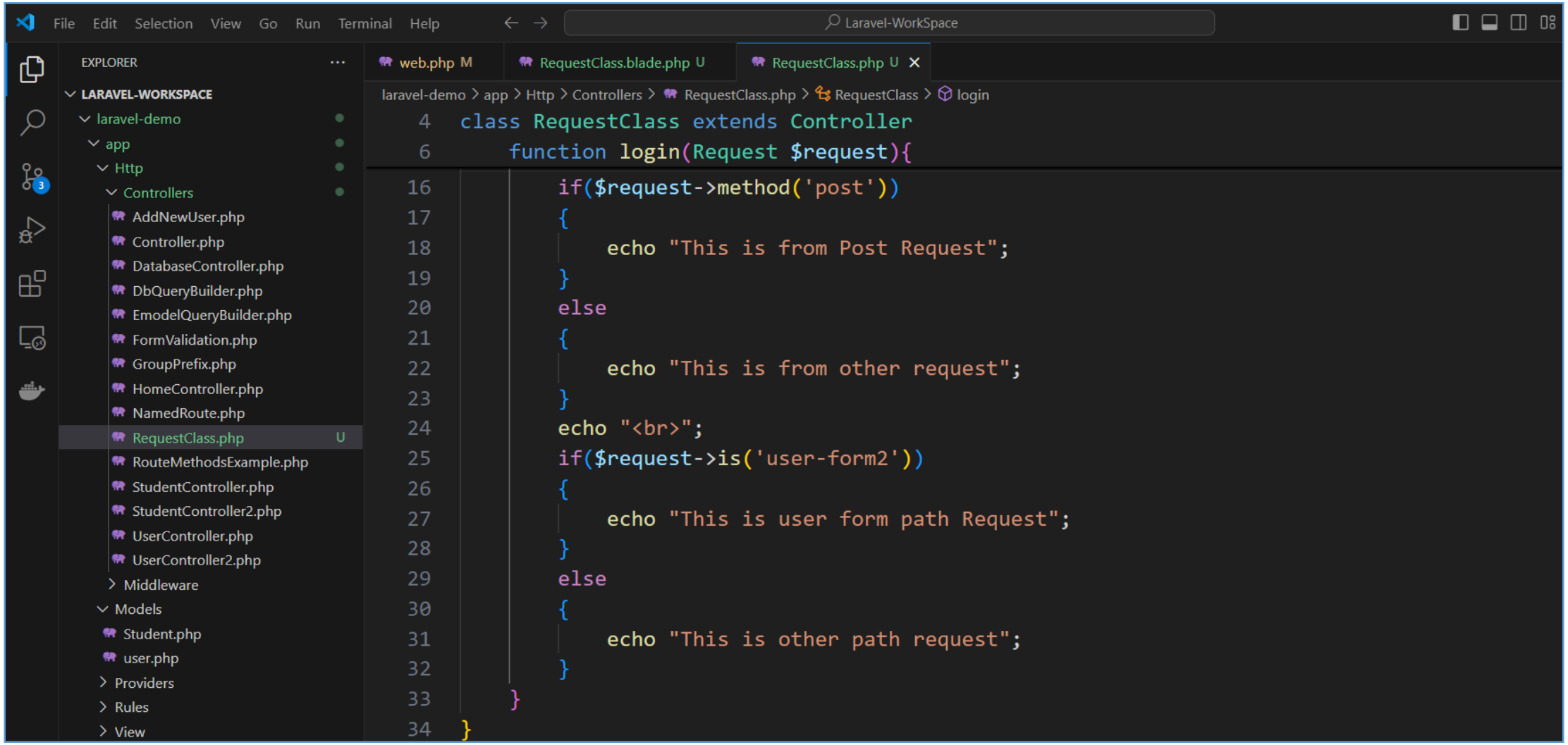
View (RequestClass)



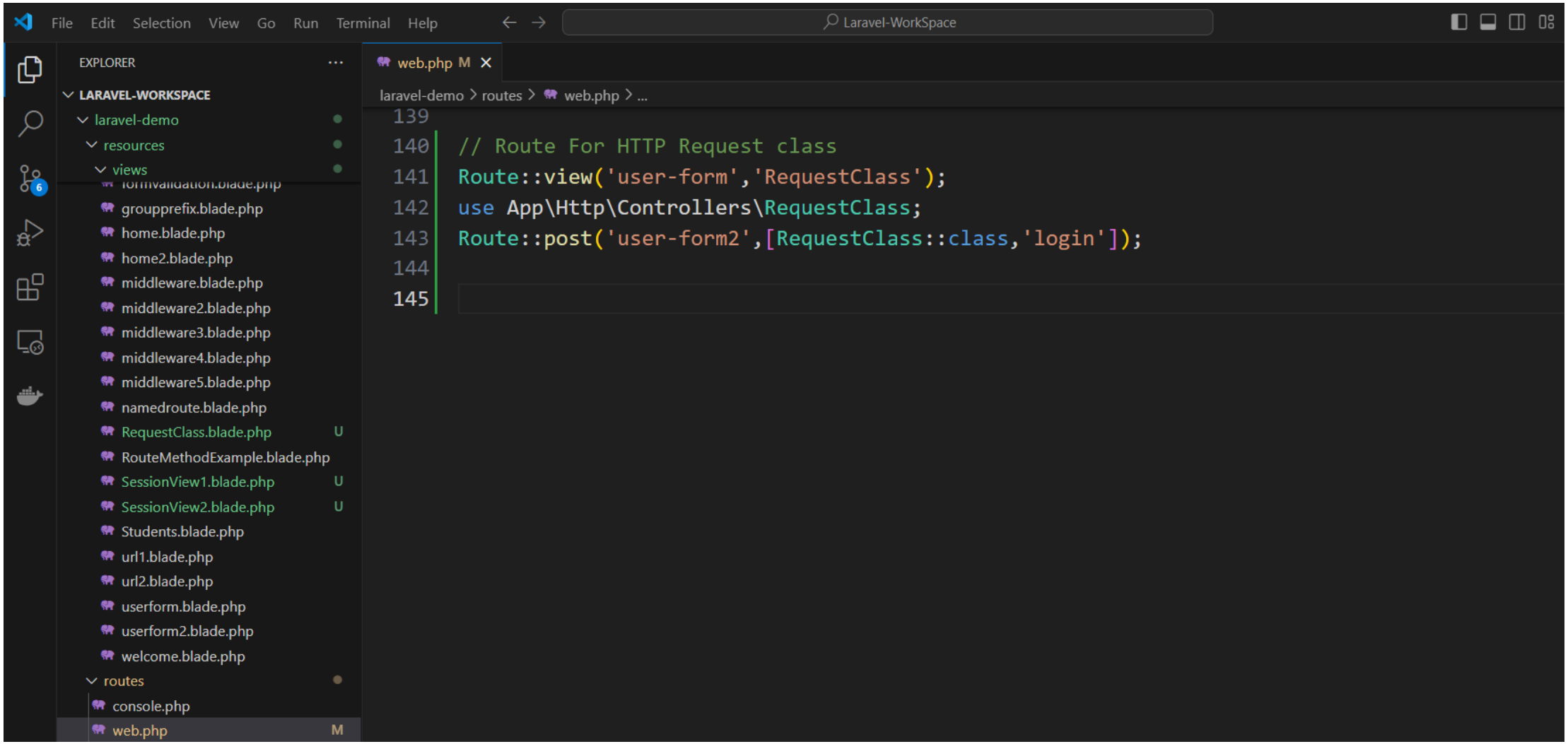
Controller (RequestClass)



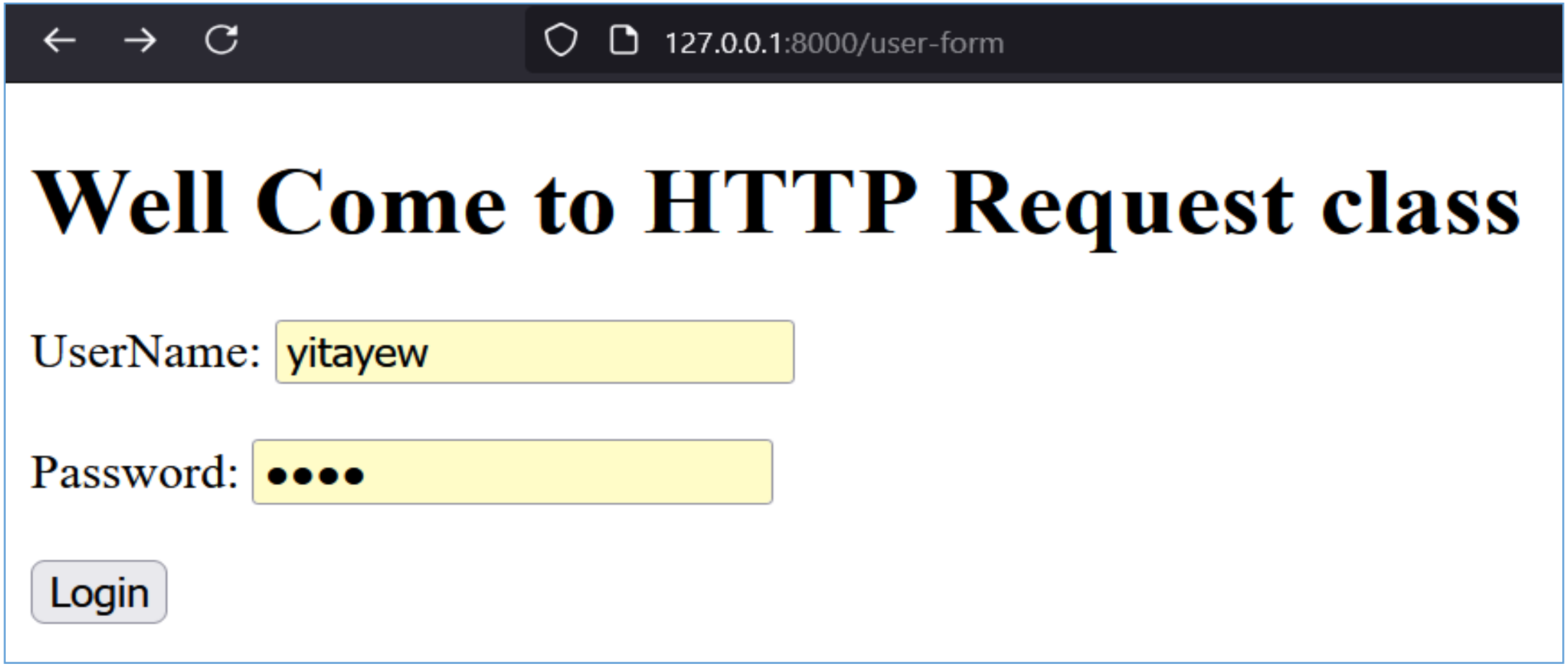
Cont. ...


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
app
Http
Controllers
AddNewUser.php
Controller.php
DatabaseController.php
DbQueryBuilder.php
EmodelQueryBuilder.php
FormValidation.php
GroupPrefix.php
HomeController.php
NamedRoute.php
RequestClass.php U
RouteMethodsExample.php
StudentController.php
StudentController2.php
UserController.php
UserController2.php
Middleware
Models
Student.php
user.php
Providers
Rules
View
web.php M
RequestClass.blade.php U
RequestClass.php U
laravel-demo > app > Http > Controllers > RequestClass.php > RequestClass > login
4 class RequestClass extends Controller
6 function login(Request $request){
16 if($request->method('post'))
17 {
18 echo "This is from Post Request";
19 }
20 else
21 {
22 echo "This is from other request";
23 }
24 echo "
";
25 if($request->is('user-form2'))
26 {
27 echo "This is user form path Request";
28 }
29 else
30 {
31 echo "This is other path request";
32 }
33 }
34 }

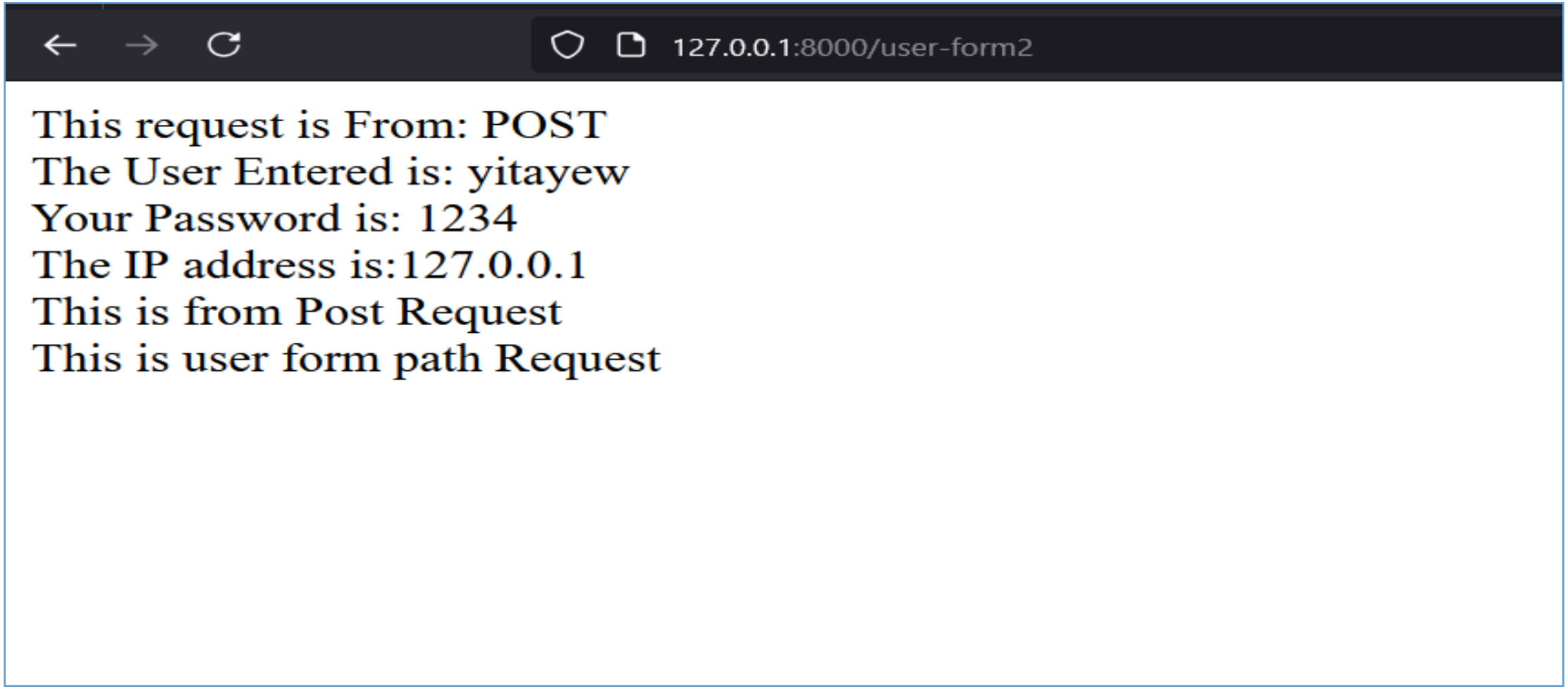
Route


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
grouprefix.blade.php
home.blade.php
home2.blade.php
middleware.blade.php
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RequestClass.blade.php U
RouteMethodExample.blade.php
SessionView1.blade.php U
SessionView2.blade.php U
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
laravel-demo > routes > web.php > ...
139
140 // Route For HTTP Request class
141 Route::view('user-form','RequestClass');
142 use App\Http\Controllers\RequestClass;
143 Route::post('user-form2',[RequestClass::class,'login']);
144
145

Form(user-form)



Output


127.0.0.1:8000/user-form2
This request is From: POST
The User Entered is: yitayew
Your Password is: 1234
The IP address is:127.0.0.1
This is from Post Request
This is user form path Request

Common Methods in the Request Class

Common Methods in the Request Class

Method	Description	Example
`all()`	Retrieves all input data as an array.	`$data = $request->all();`
`input('key')`	Retrieves a specific input value by key.	`$name = $request->input('name');`
`only(['key1', 'key2'])`	Retrieves only specific inputs from the request.	`$data = $request->only(['name', 'email']);`
`except(['key1', 'key2'])`	Retrieves all input data except specified keys.	`$data = $request->except(['password']);`
`has('key')`	Checks if a particular key is present in the request.	`if ($request->has('name')) { /* logic */ }`
`filled('key')`	Checks if an input is present and not empty.	`if ($request->filled('name')) { /* logic */ }`
`query('key')`	Retrieves a query parameter from the URL.	`$page = $request->query('page');`

Cont. ...

`cookie('cookieName')`	Retrieves a cookie value.	`$cookie = $request->cookie('cookieName');`
`file('fileKey')`	Retrieves an uploaded file.	`$file = $request->file('avatar');`
`method()`	Retrieves the HTTP method of the request (e.g., GET, POST).	`$method = $request->method();`
`isMethod('post')`	Checks if the request method matches the specified method.	`if ($request->isMethod('post')) { /* logic */ }`
`header('headerName')`	Retrieves a value from the request headers.	`$userAgent = $request->header('User-Agent');`
`ip()`	Retrieves the client's IP address.	`$ipAddress = $request->ip();`
`ajax()`	Checks if the request was made via AJAX.	`if ($request->ajax()) { /* logic */ }`
`json()`	Retrieves JSON data from the request, if available.	`$data = $request->json()->all();`

Advantage of Request Class

Key Benefits

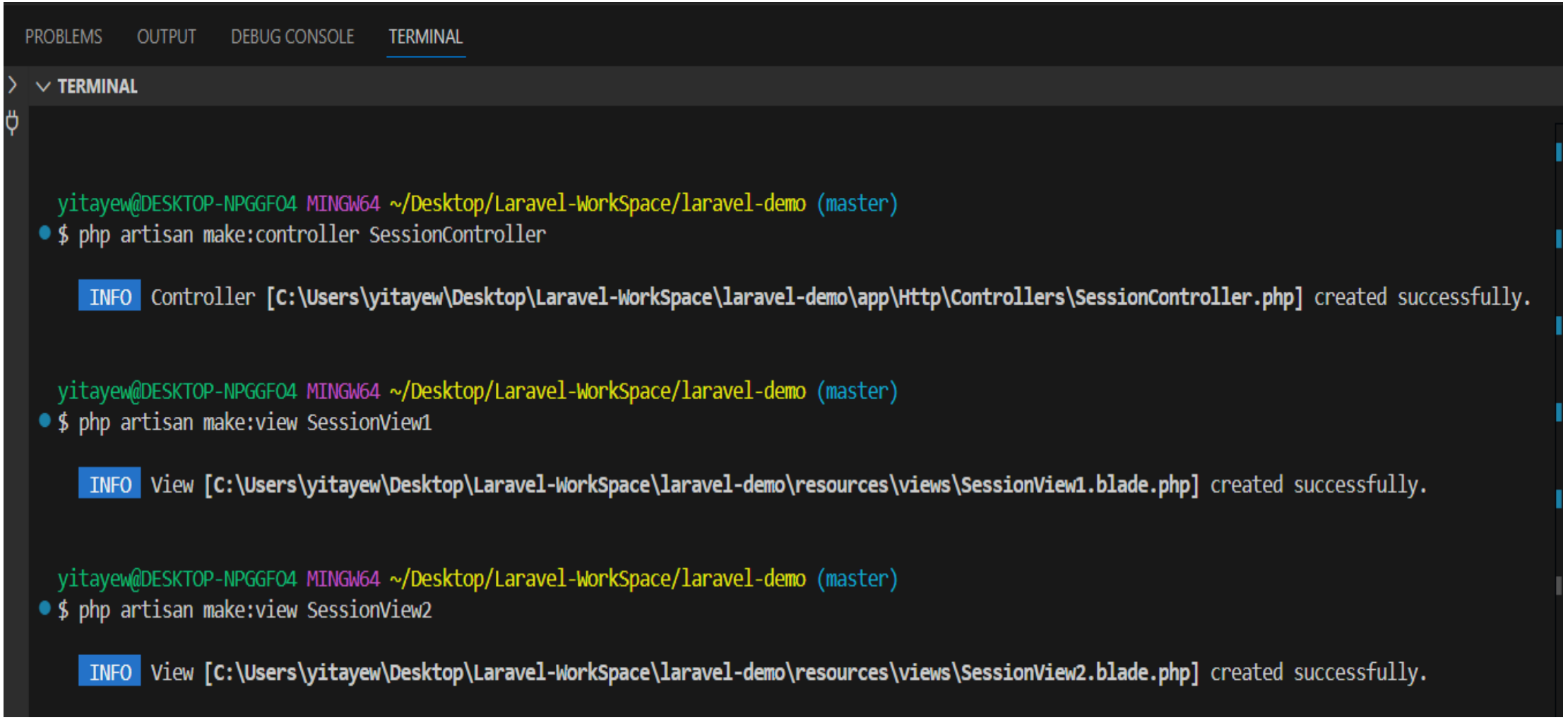
1. **Convenient Access:** The `Request` class provides easy access to all aspects of the HTTP request.
2. **Validation Integration:** Request validation can be performed directly on the incoming data.
3. **Flexible Input Handling:** Request inputs, files, headers, and more can be accessed, making it flexible for handling user data.

The `Request` class is foundational for processing HTTP requests in Laravel, making it straightforward to work with user input, files, and headers in a unified way.

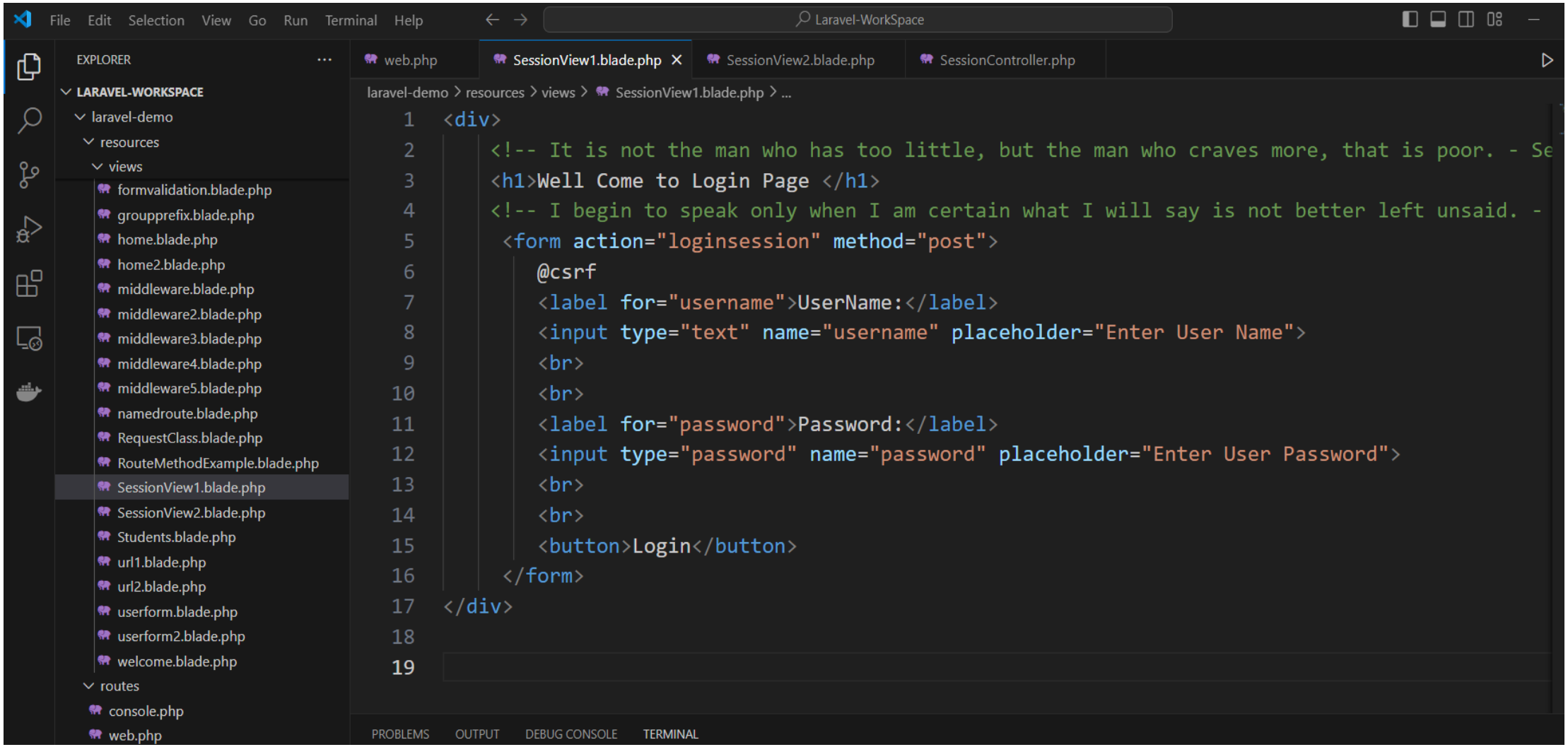
Session in laravel

- In Laravel, a session is a way to **store information** about a user across multiple requests, allowing data to persist between pages. Sessions are commonly used for tasks such as storing **user authentication** data, **flash messages**, form data, and other **temporary** user information.
- Laravel provides a robust session management system, which is configured in the config/session.php file, and supports several backends for session storage, such as file storage, cookies, database, Redis, and memory-based stores like Memcached.

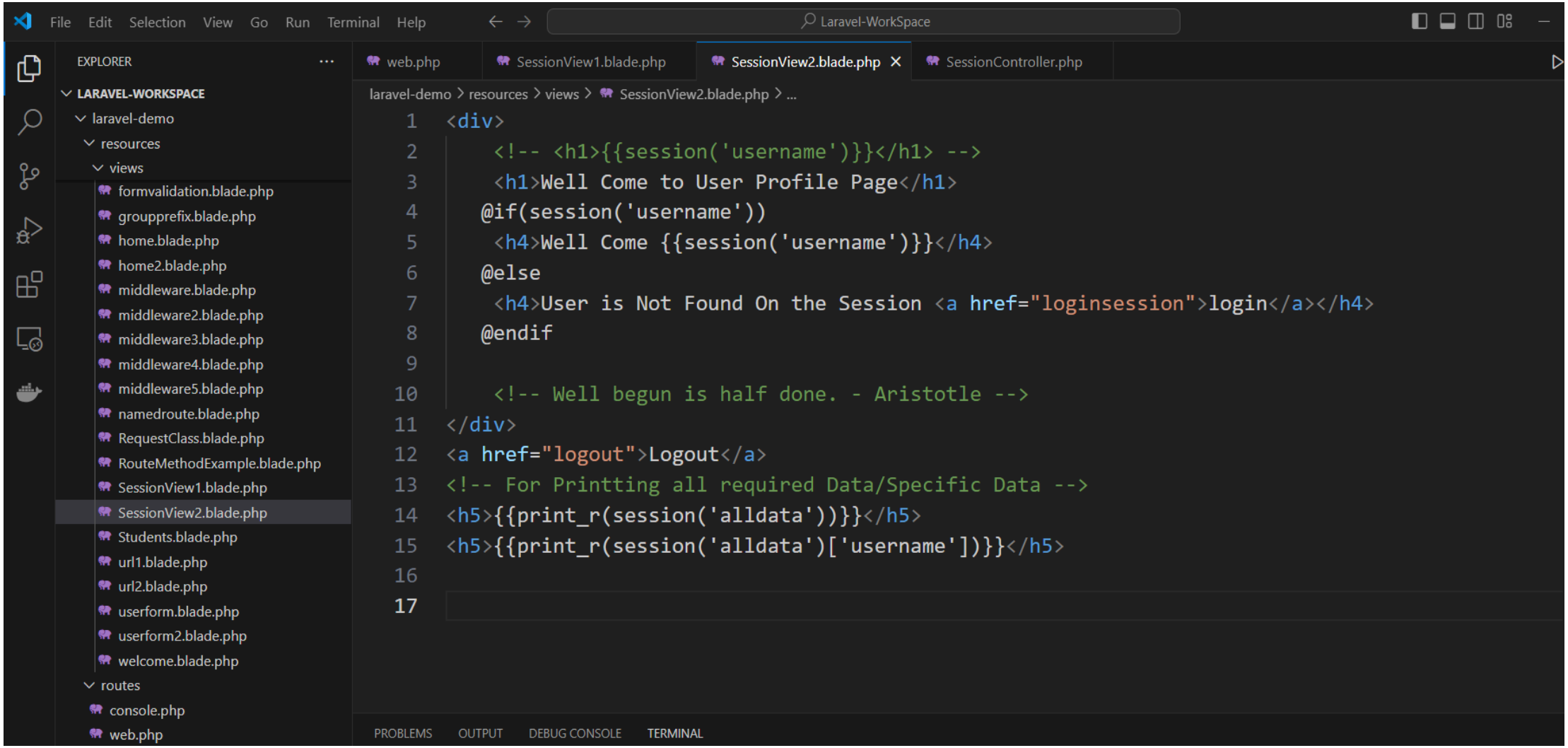
Example (Creating Controller and View)



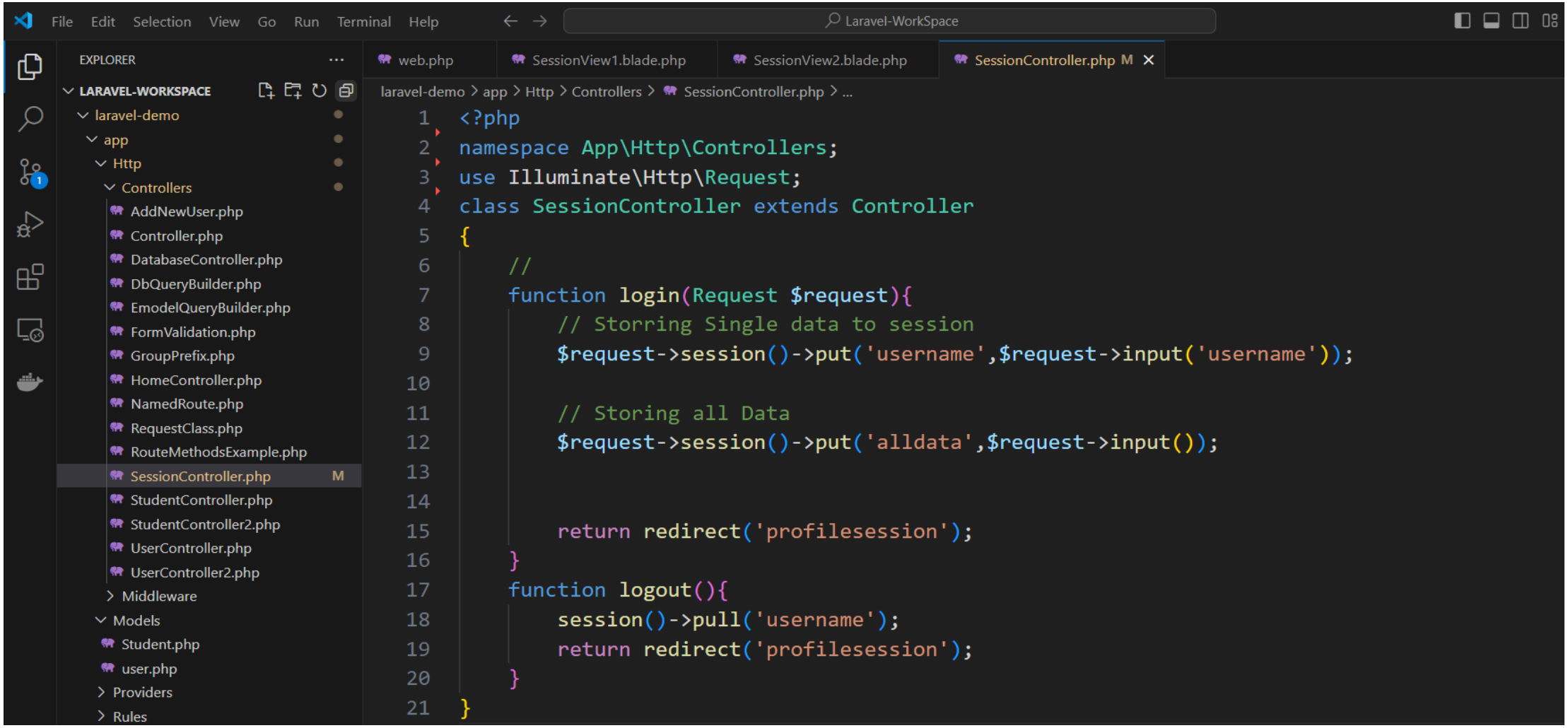
Example: View(SessionView1)



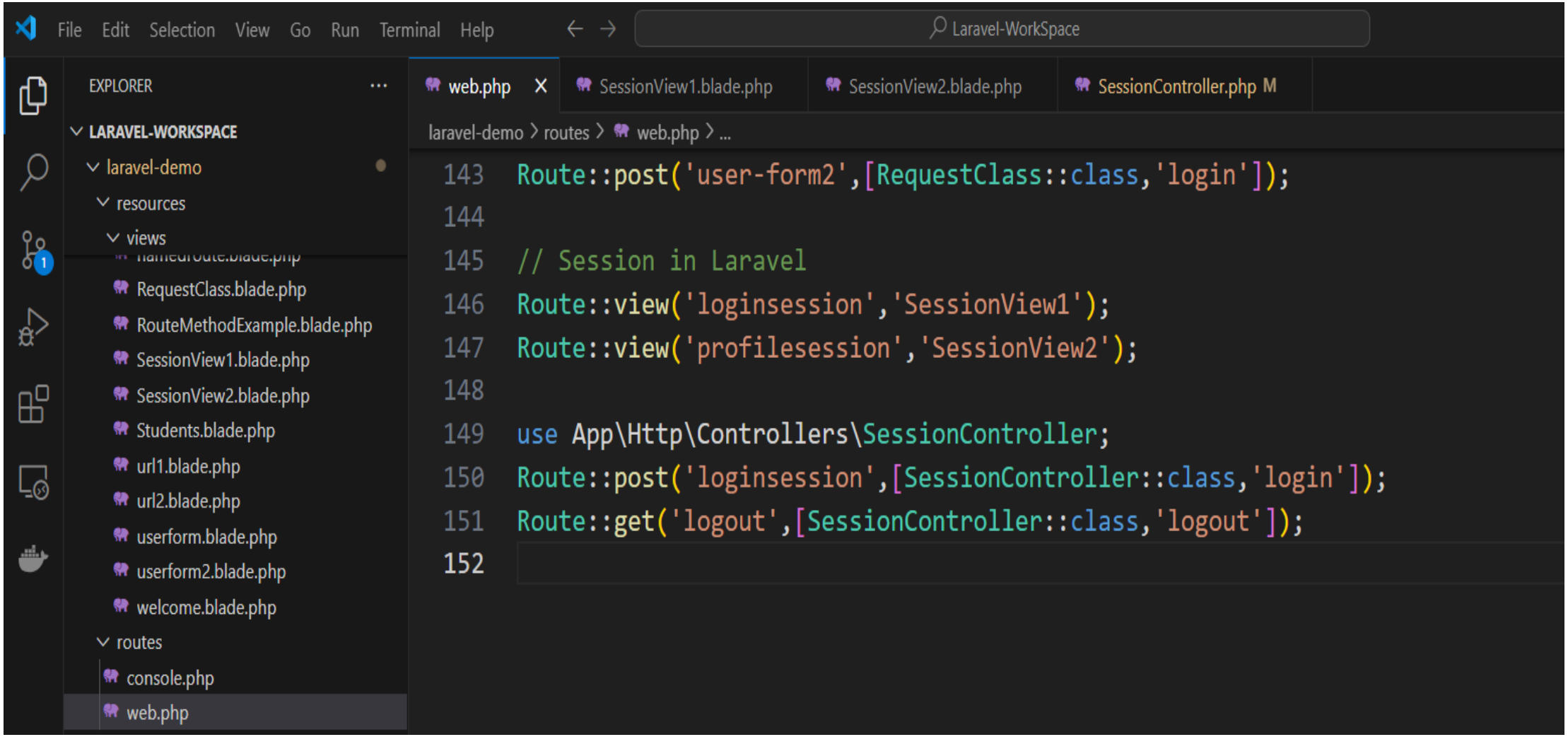
View (SessionView2)



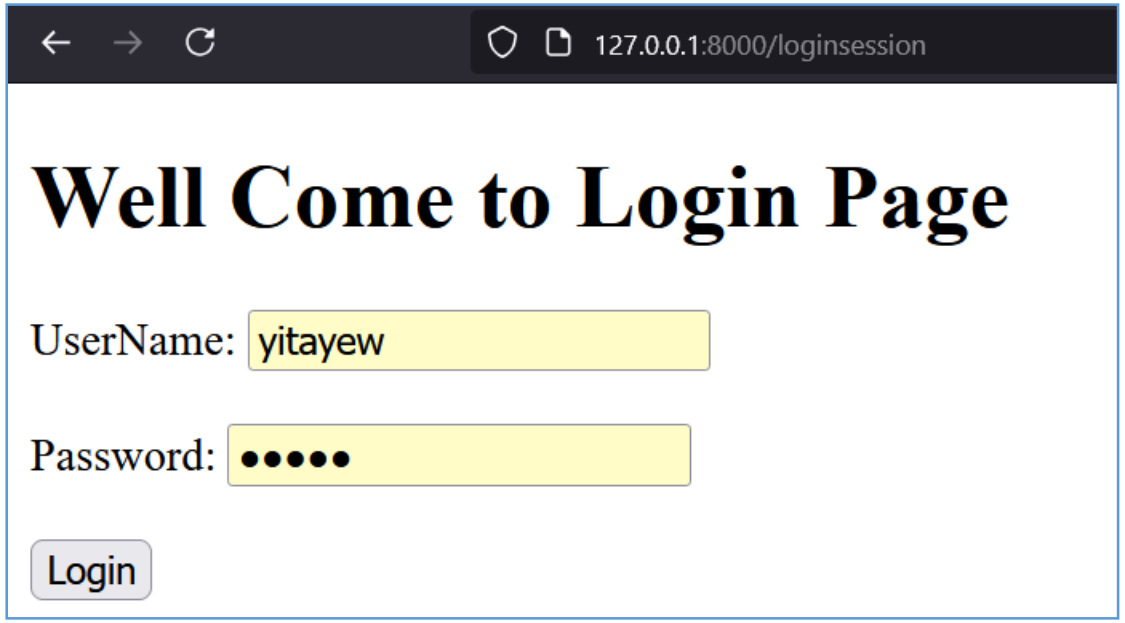
Controller (SessionController)

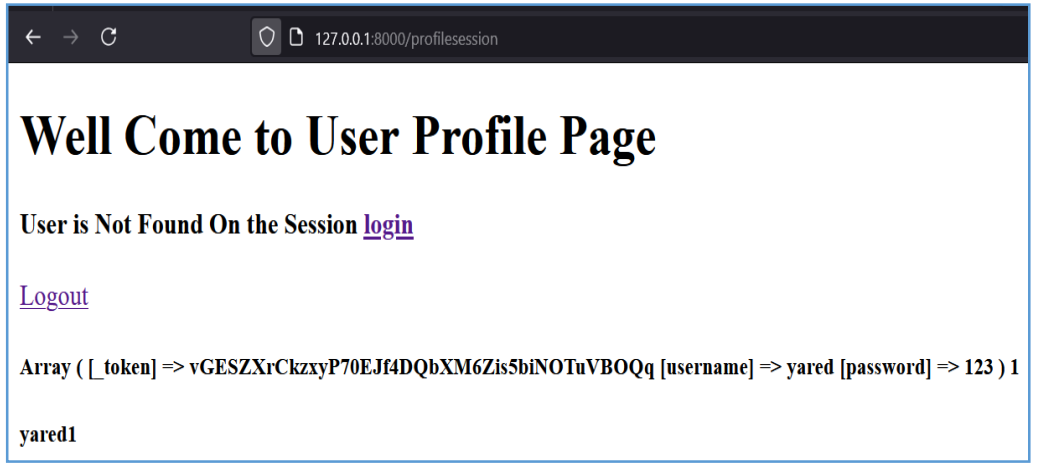


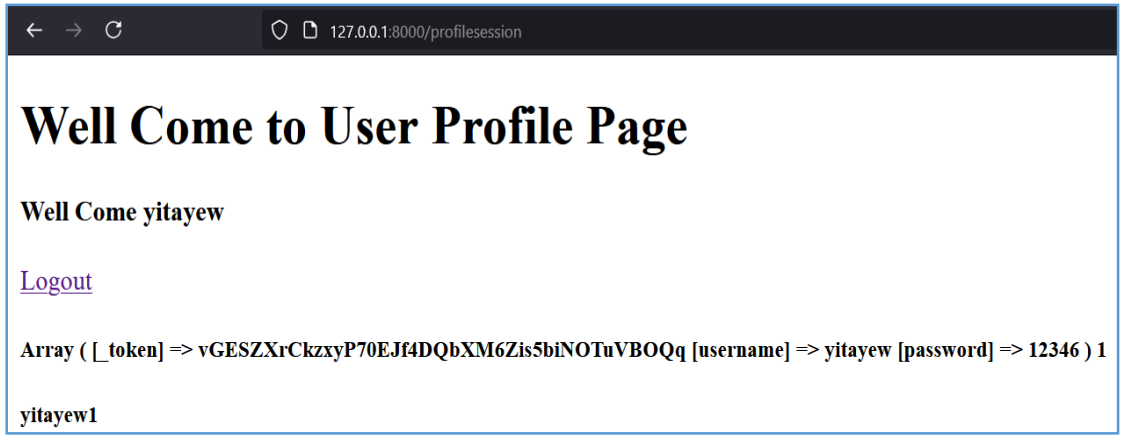
Route

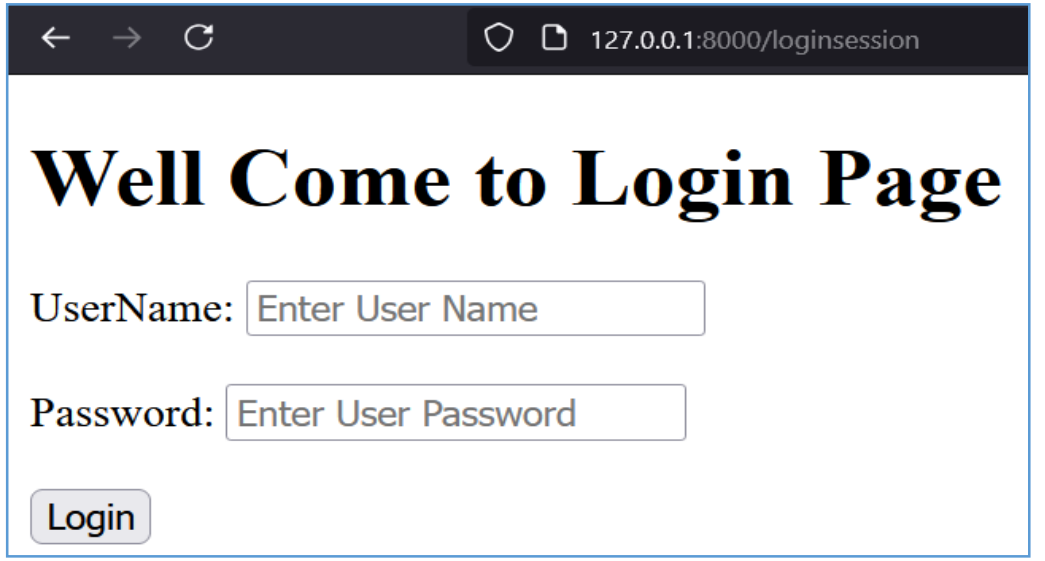

File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
RequestClass.blade.php
RouteMethodExample.blade.php
SessionView1.blade.php
SessionView2.blade.php
Students.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php
web.php
SessionView1.blade.php
SessionView2.blade.php
SessionController.php M
laravel-demo > routes > web.php > ...
143 Route::post('user-form2',[RequestClass::class,'login']);
144
145 // Session in Laravel
146 Route::view('loginsession','SessionView1');
147 Route::view('profilesession','SessionView2');
148
149 use App\Http\Controllers\SessionController;
150 Route::post('loginsession',[SessionController::class,'login']);
151 Route::get('logout',[SessionController::class,'logout']);
152

Output

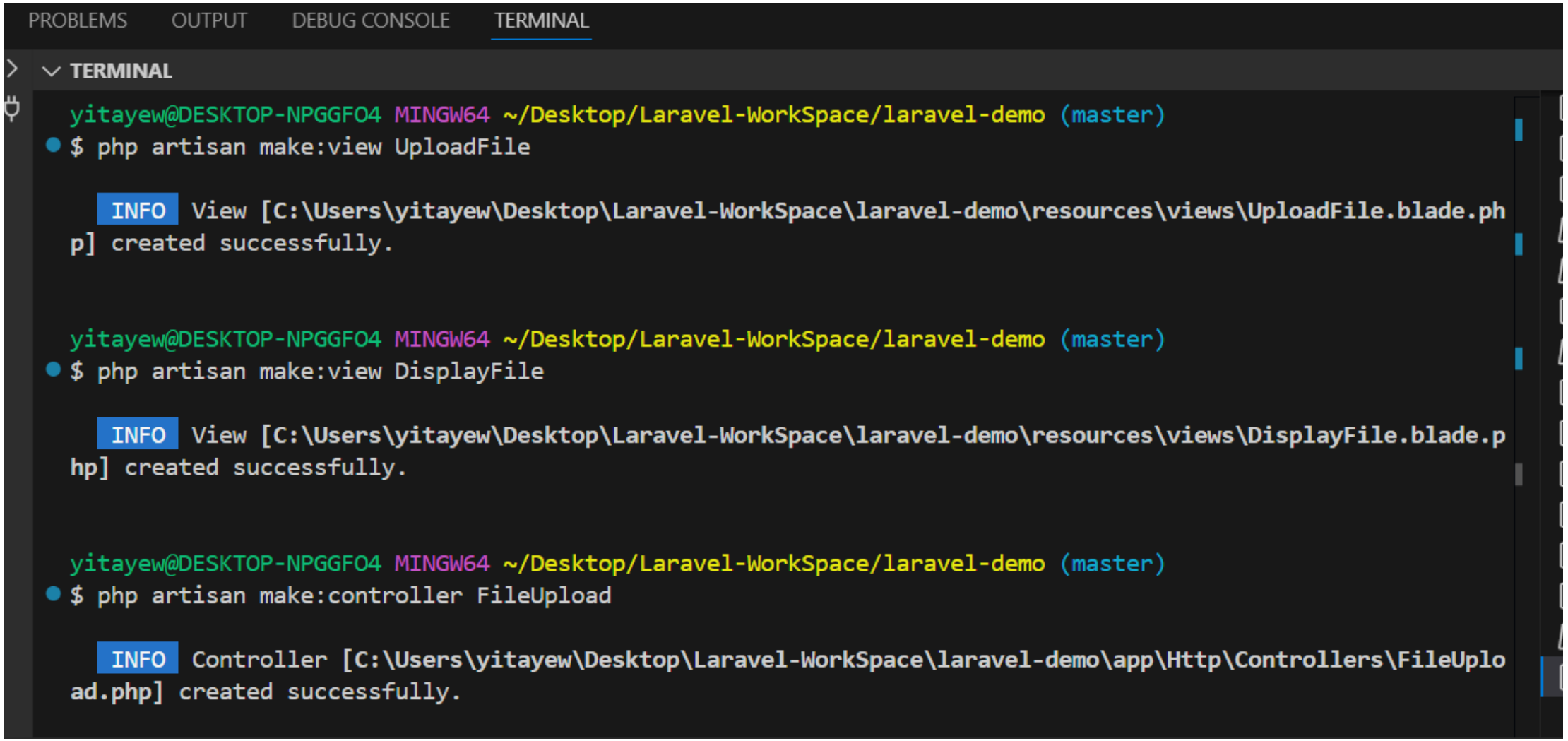

127.0.0.1:8000/loginsession
Well Come to Login Page
UserName: yitayew
Password: •••••
Login


127.0.0.1:8000/profilesession
Well Come to User Profile Page
User is Not Found On the Session login
Logout
Array ([_token] => vGESZXrCkzxyP70EJf4DQbXM6Zis5biNOTuVBOQq [username] => yared [password] => 123) 1
yared1

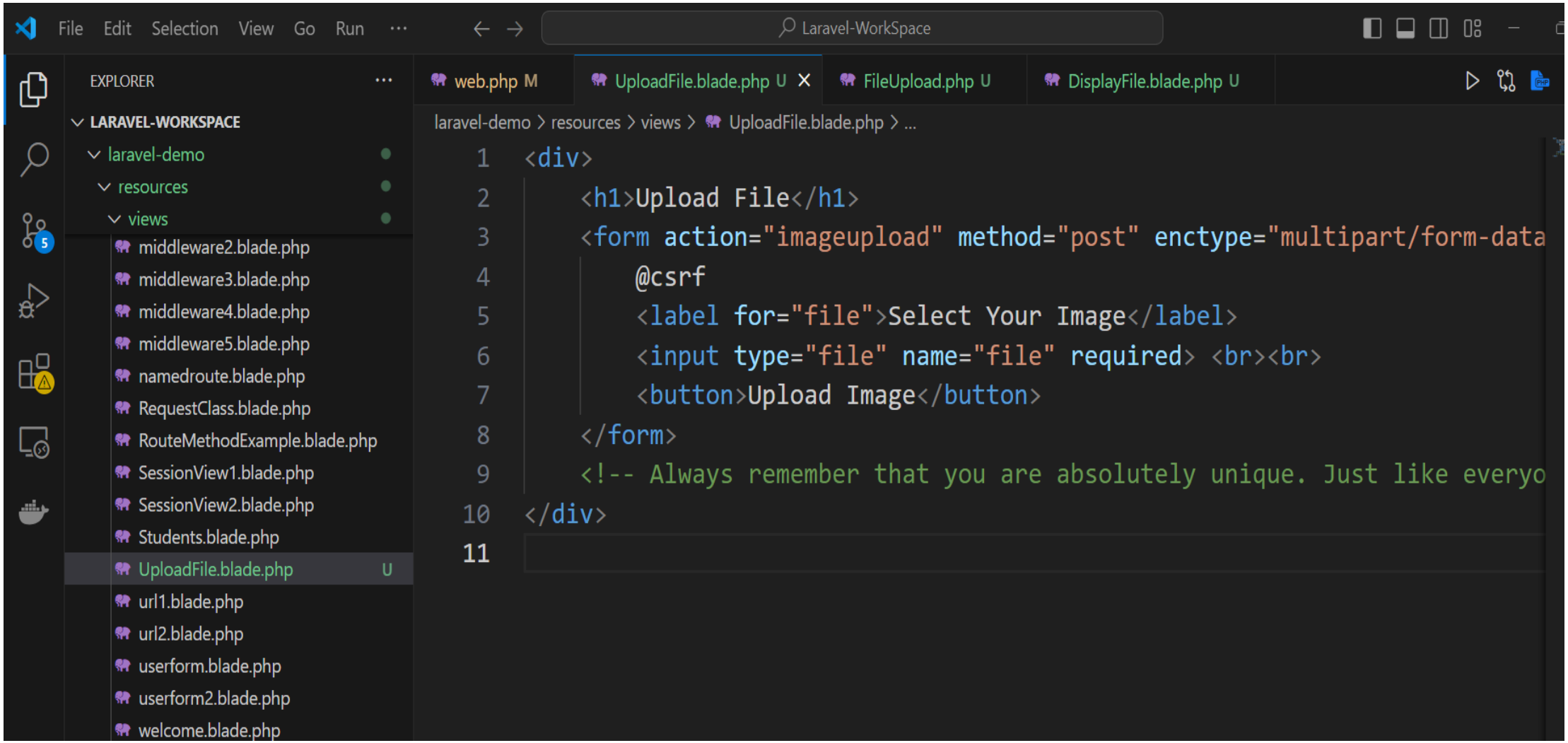

127.0.0.1:8000/profilesession
Well Come to User Profile Page
Well Come yitayew
Logout
Array ([_token] => vGESZXrCkzxyP70EJf4DQbXM6Zis5biNOTuVBOQq [username] => yitayew [password] => 12346) 1
yitayew1


127.0.0.1:8000/loginsession
Well Come to Login Page
UserName: Enter User Name
Password: Enter User Password
Login

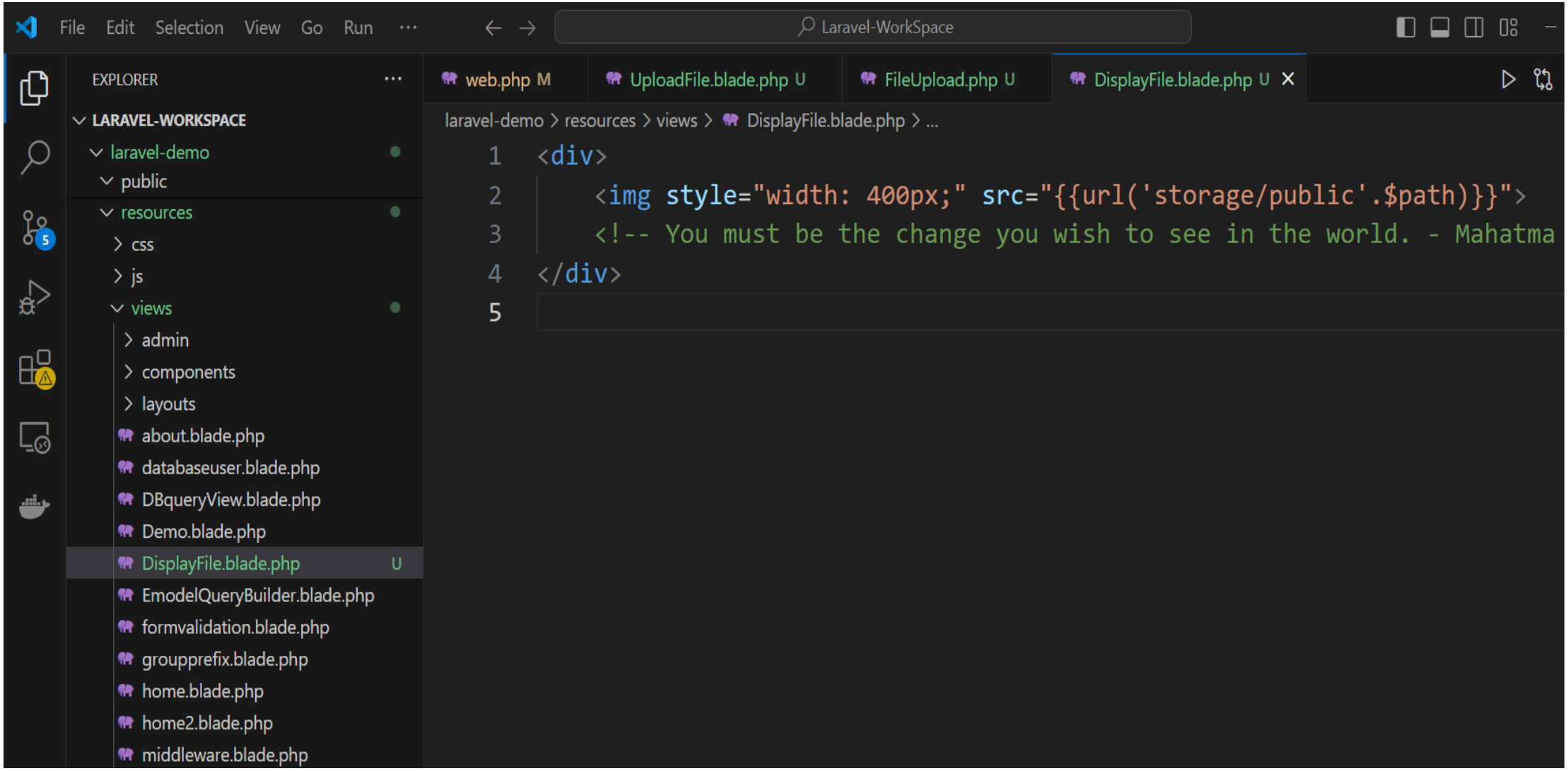
Upload file | Upload and Display Image



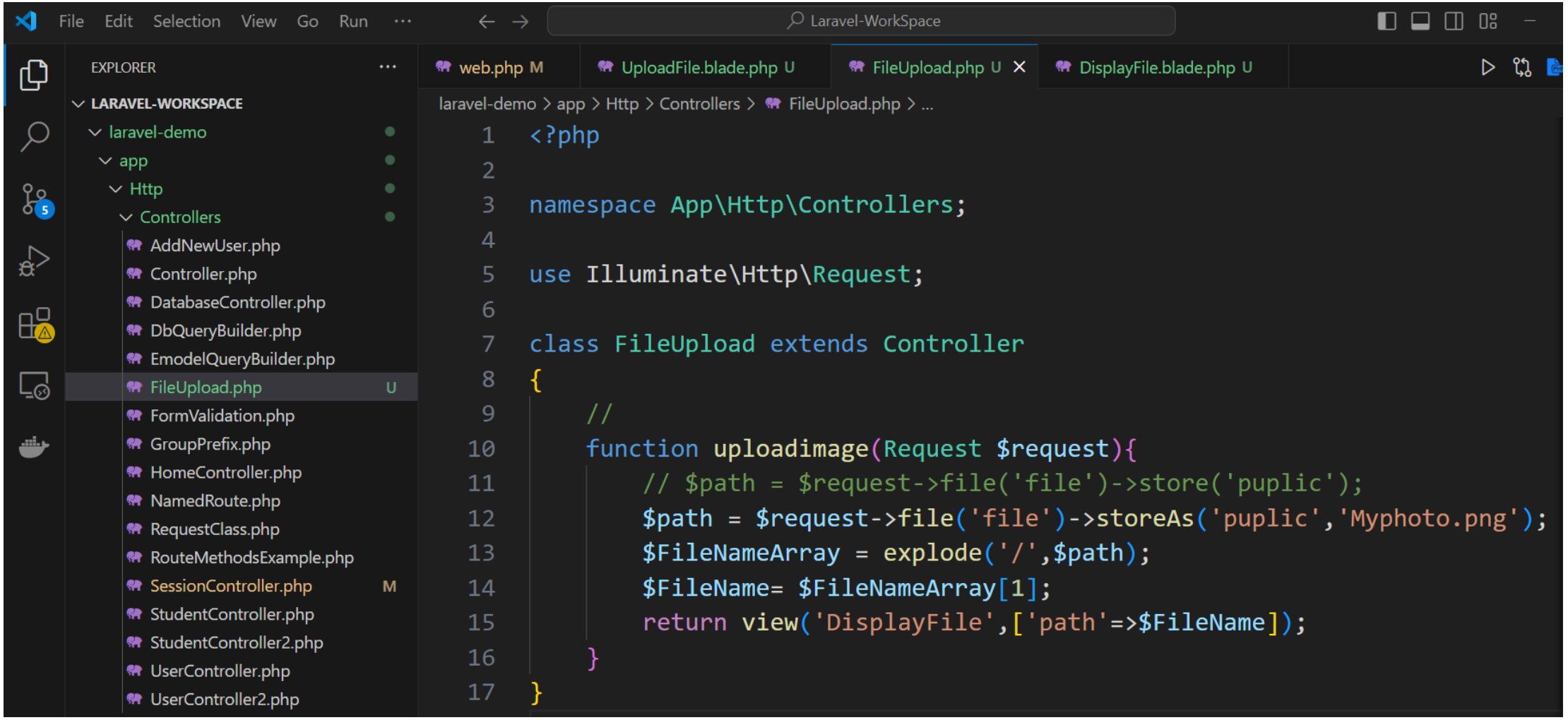
View (FileUpload)



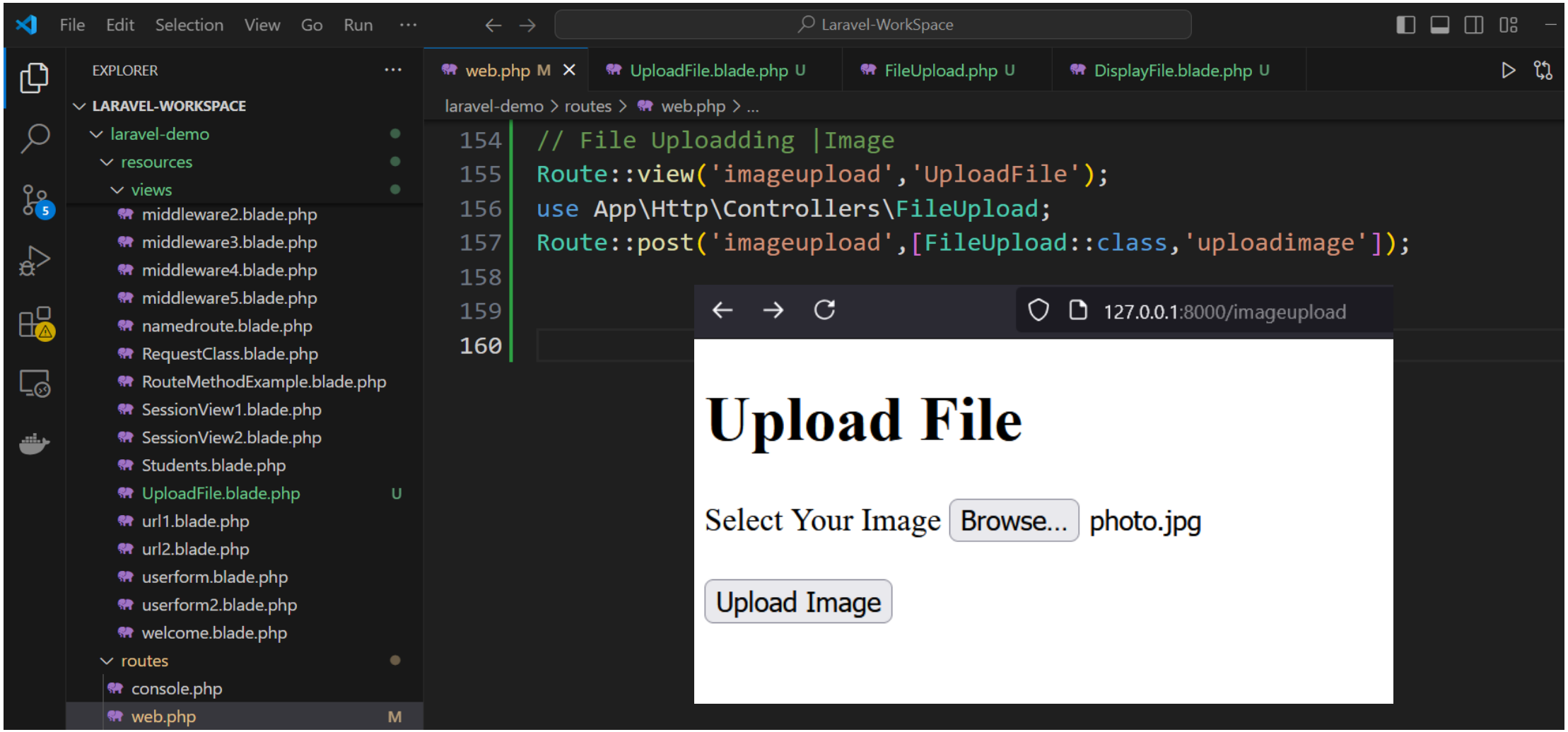
View(DisplayFile)



Controller (FileUpload)



Route


File Edit Selection View Go Run
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RequestClass.blade.php
RouteMethodExample.blade.php
SessionView1.blade.php
SessionView2.blade.php
Students.blade.php
UploadFile.blade.php U
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
UploadFile.blade.php U
FileUpload.php U
DisplayFile.blade.php U
laravel-demo > routes > web.php > ...
154 // File Uploadding |Image
155 Route::view('imageupload','UploadFile');
156 use App\Http\Controllers\FileUpload;
157 Route::post('imageupload',[FileUpload::class,'uploadimage']);
158
159
160
127.0.0.1:8000/imageupload
Upload File
Select Your Image Browse... photo.jpg
Upload Image

Thank you!

Appreciate your action.