Laravel

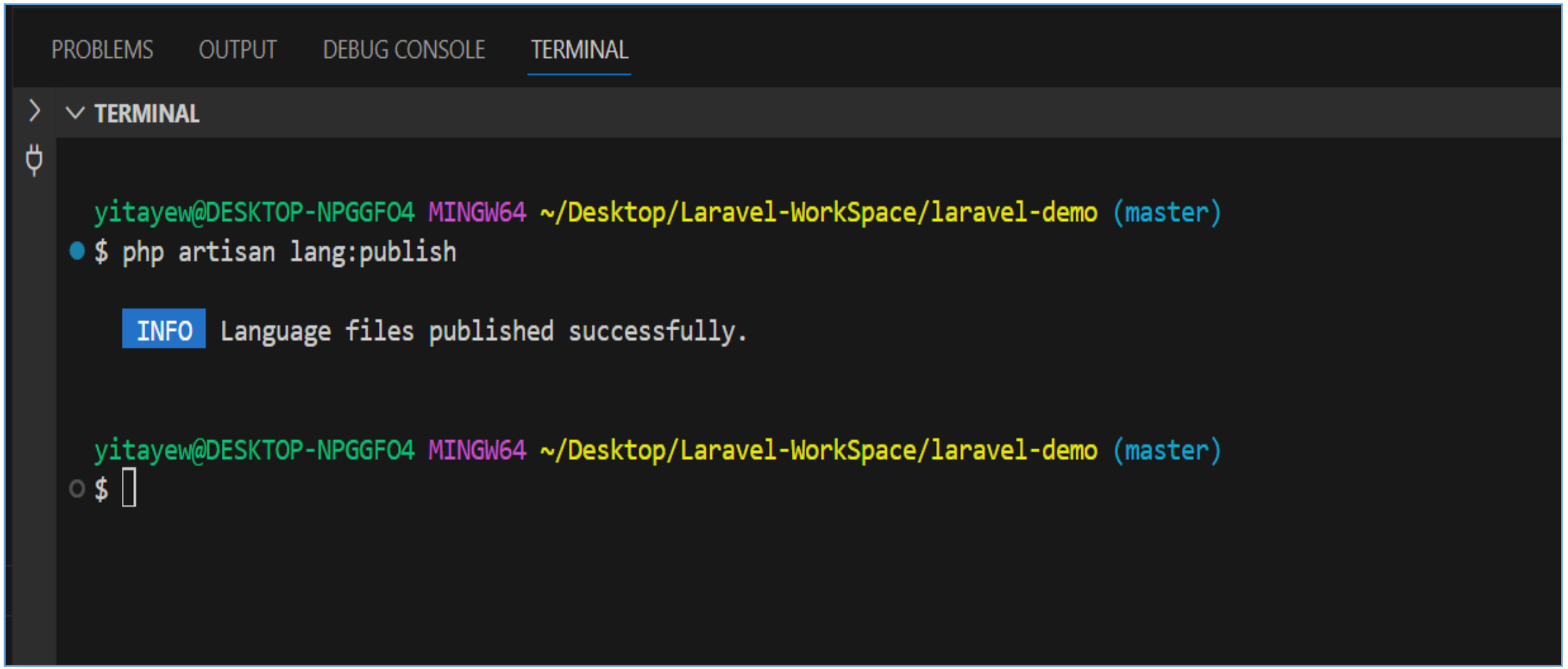
Outlines of Discussion

- Localization In Laravel
- CRUD Operation using HTML Form
- Pagination in Laravel
- Stub in Laravel
- Migration In Laravel
- Seeding in laravel

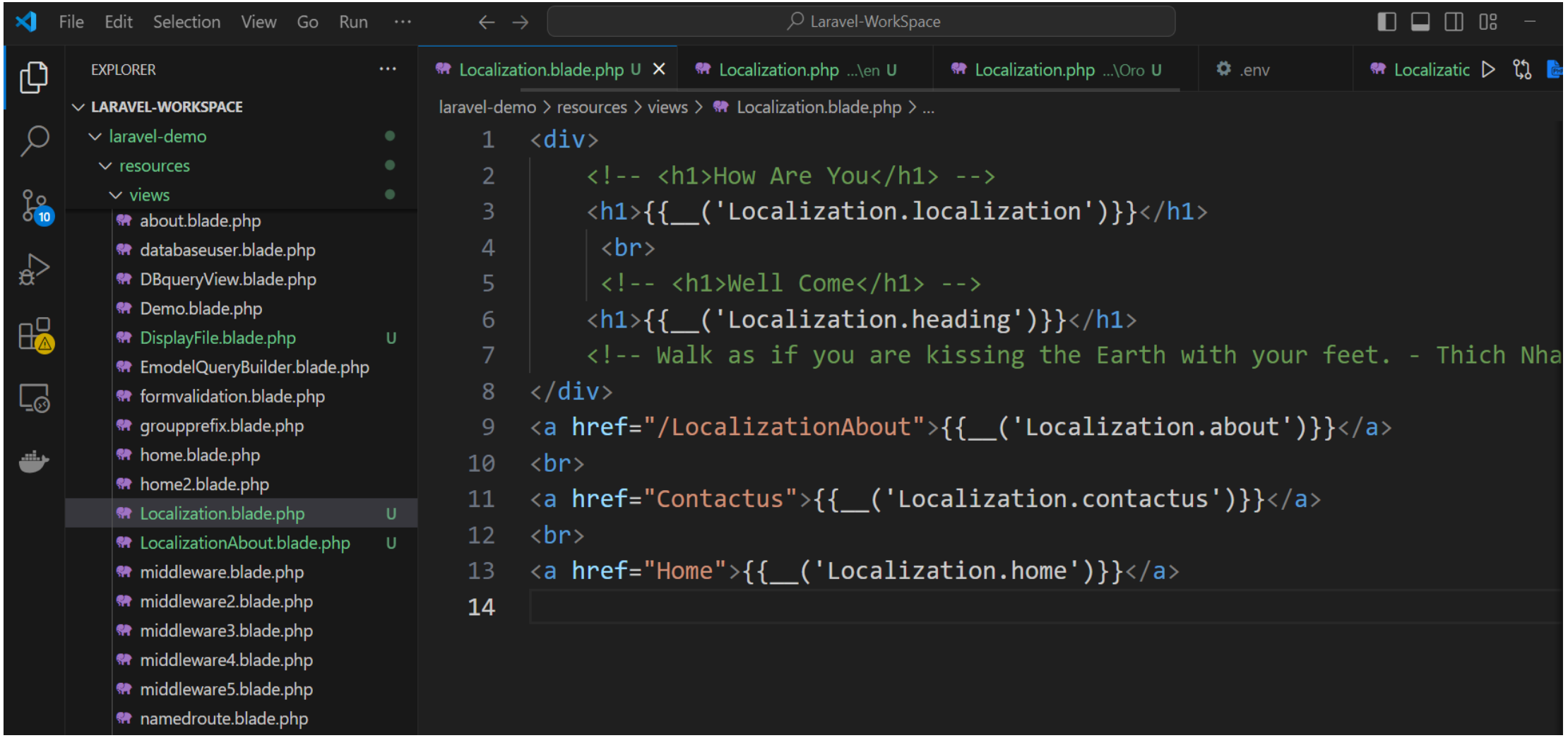
Localization In Laravel

- Localization in Laravel makes it easy to support multiple languages in your application. This is useful for applications that need to serve users in different regions or for those who prefer to interact in their native languages.
- Laravel provides a convenient way to manage language files and handle translations throughout your application.

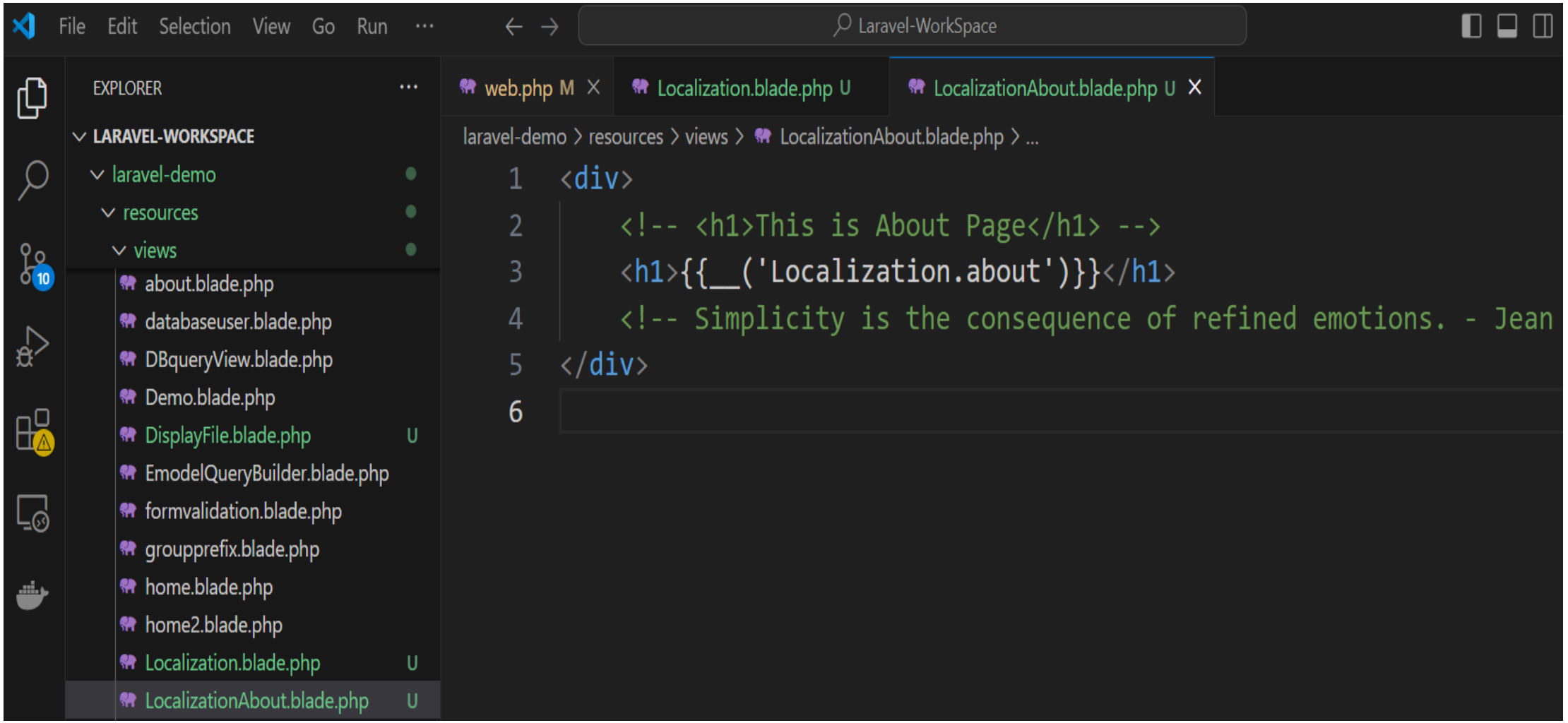
Publishing Lang folder



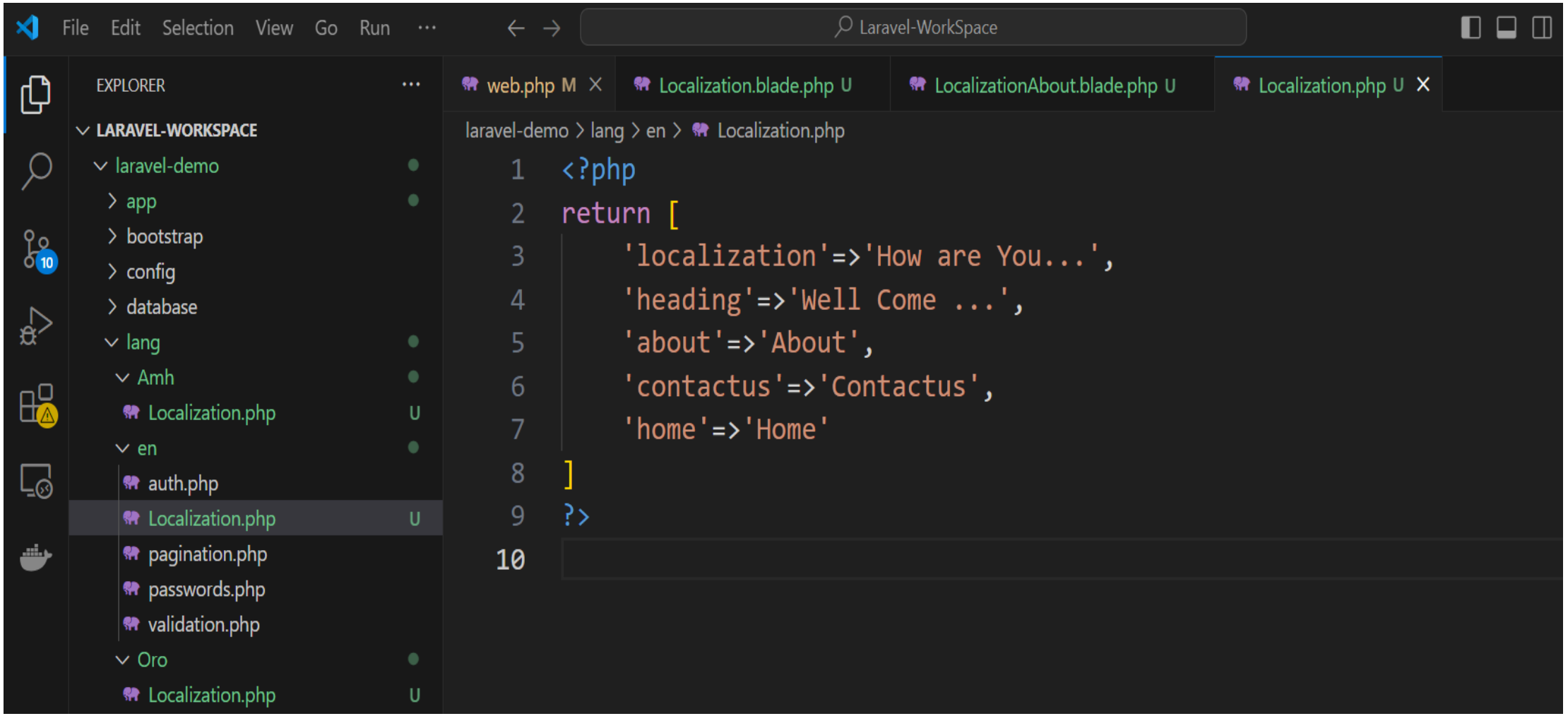
Example (View-Localization)



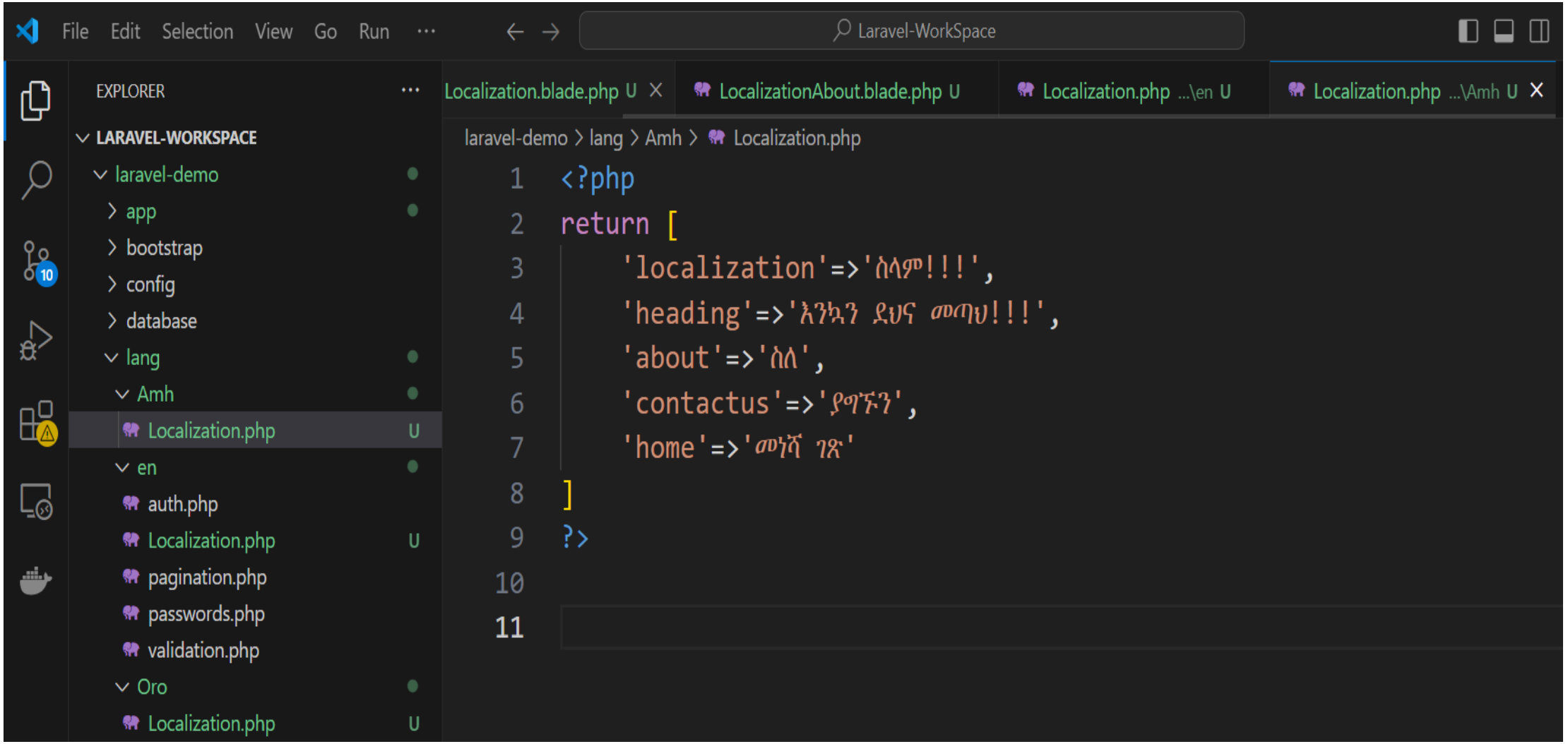
View (LocalizationAbout)



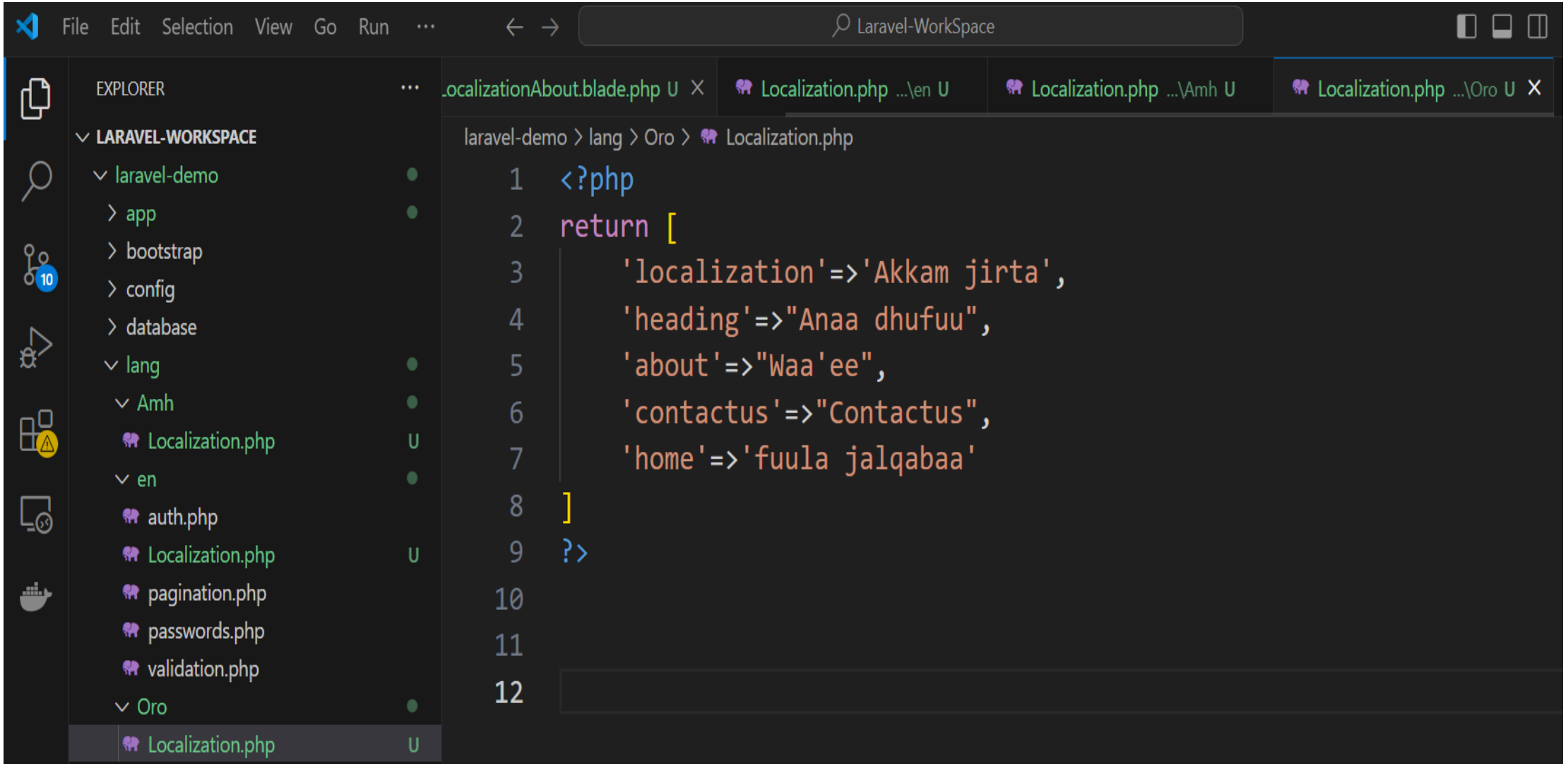
Localization(Eng)



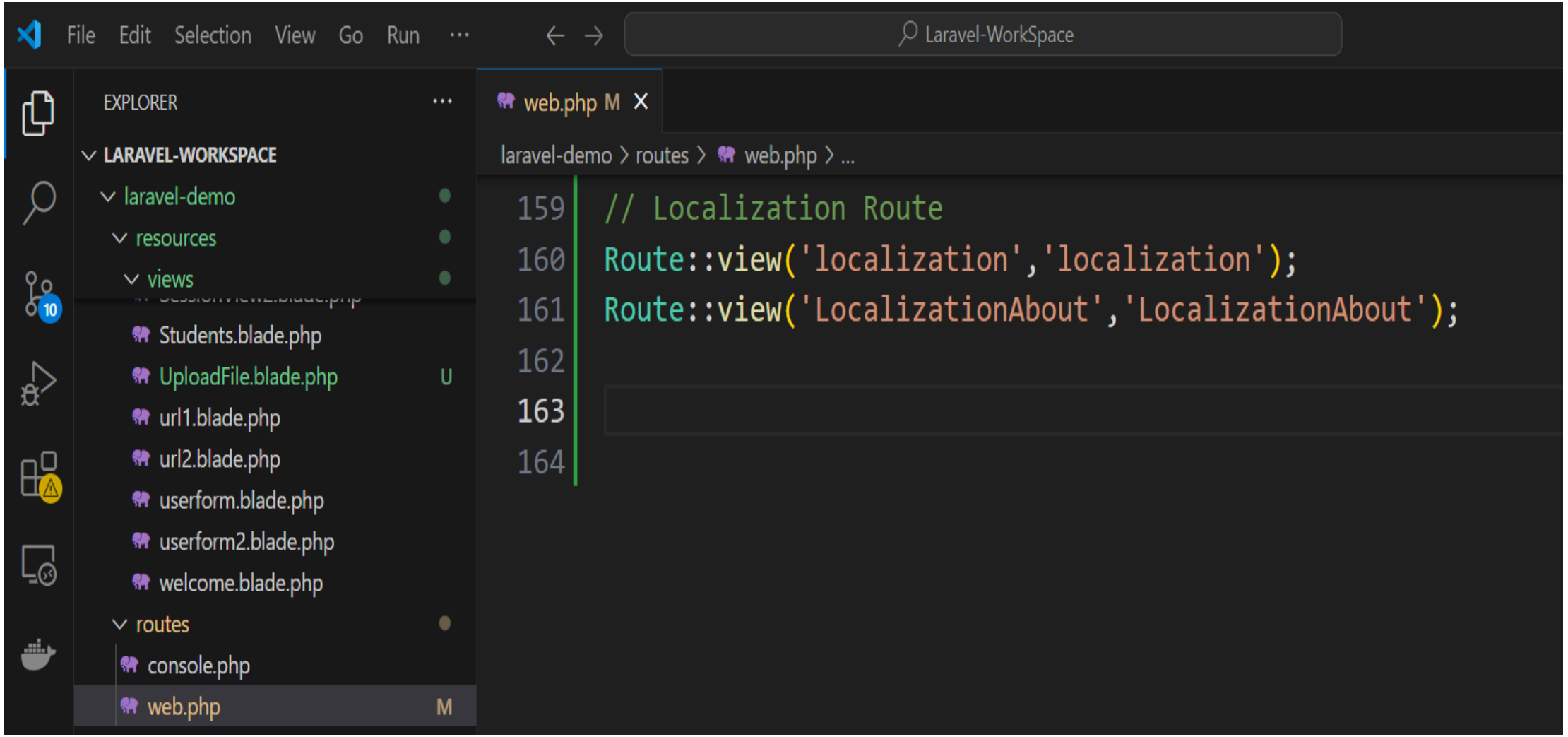
Localization (Ahm)



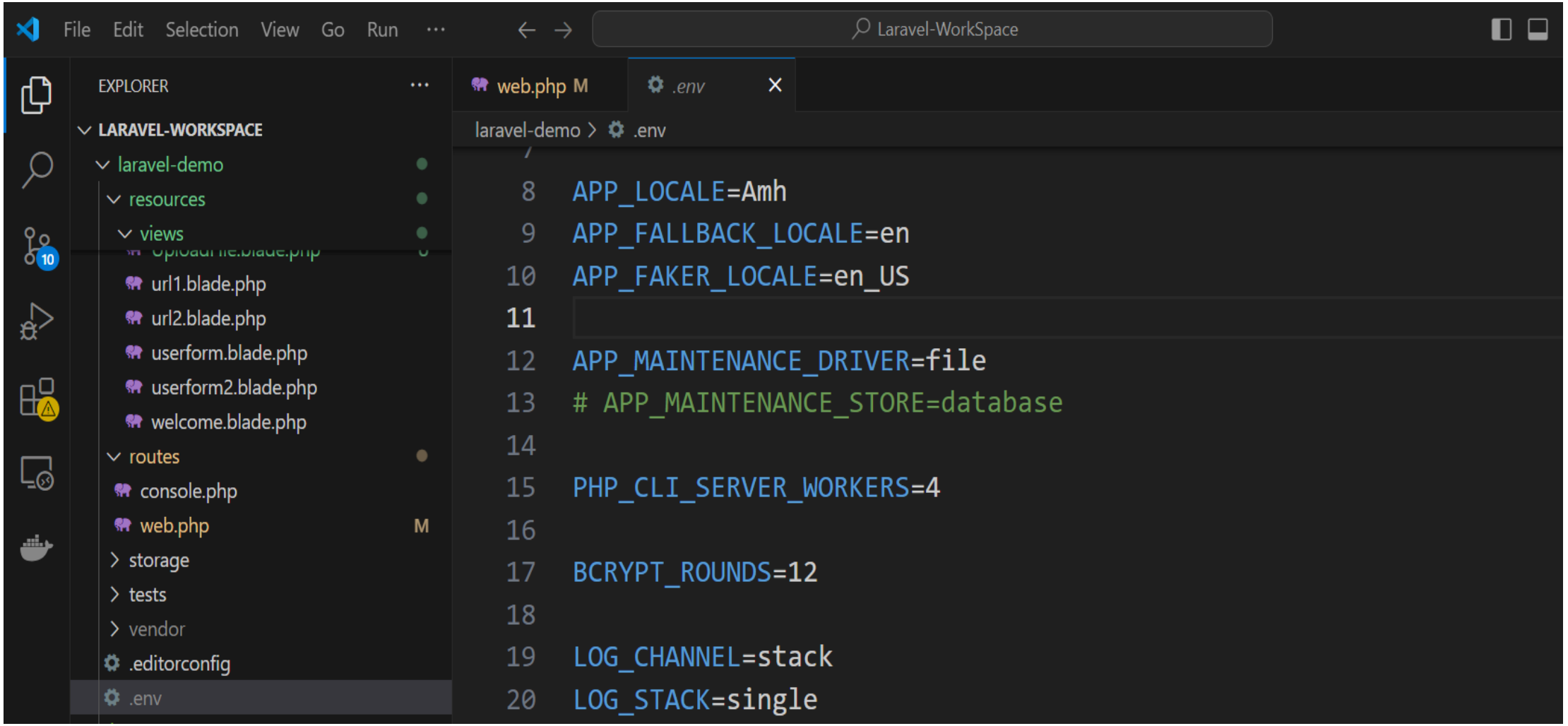
Localization (Oro)



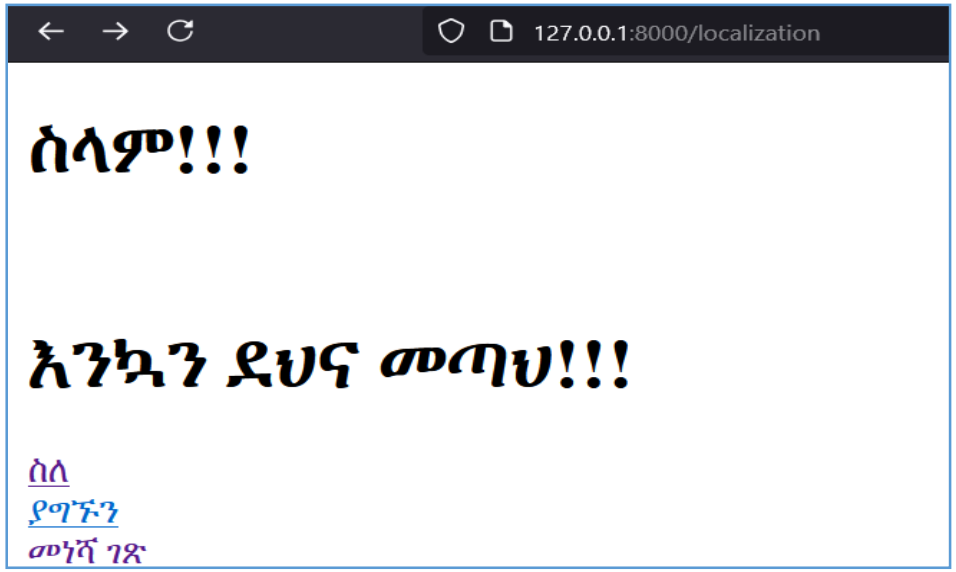
Route

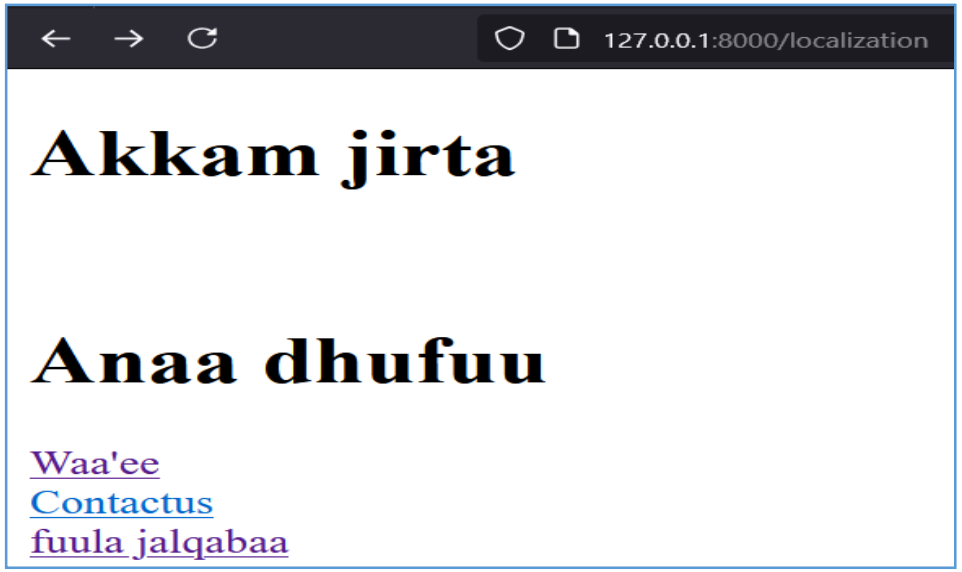

File Edit Selection View Go Run ···
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
Students.blade.php
UploadFile.blade.php U
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
web.php M
laravel-demo > routes > web.php > ...
159 // Localization Route
160 Route::view('localization','localization');
161 Route::view('LocalizationAbout','LocalizationAbout');
162
163
164

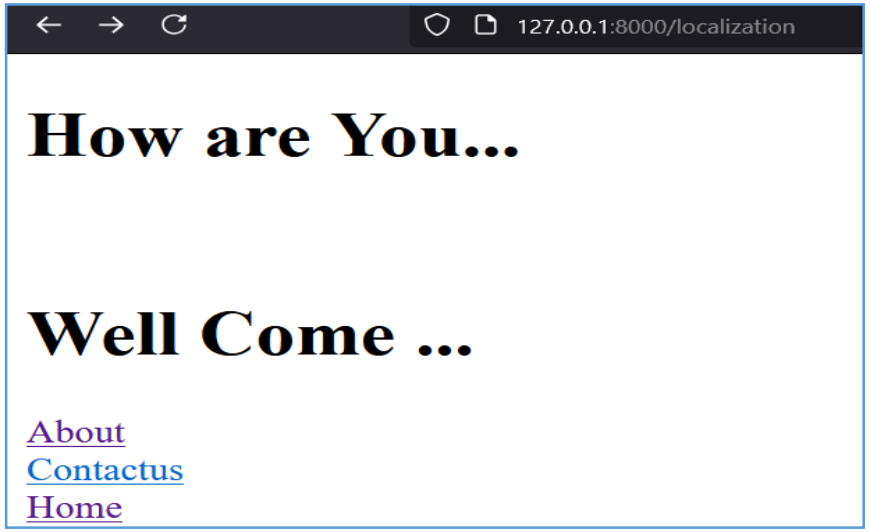
Config on (env folder)



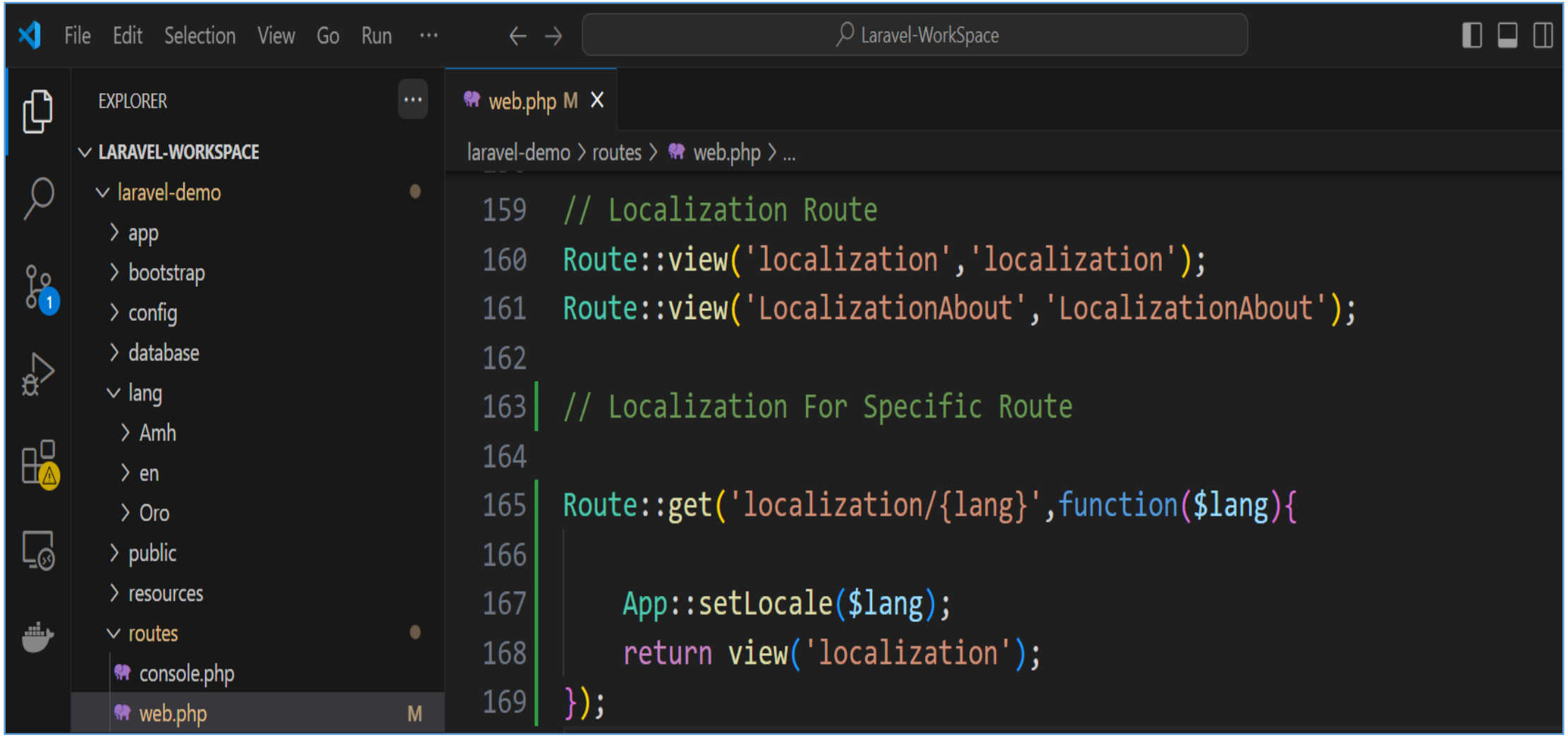
Output


127.0.0.1:8000/localization
ሰላም!!!
እንኳን ደህና መጣህ!!!
ስለ
ያግኙን
መነሻ ገጽ

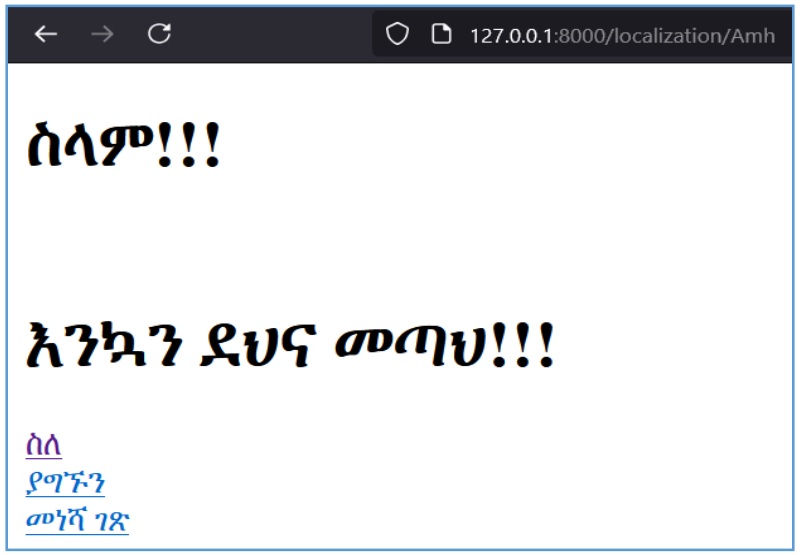

127.0.0.1:8000/localization
Akkam jirta
Anaa dhufuu
Waa'ee
Contactus
fuula jalqabaa

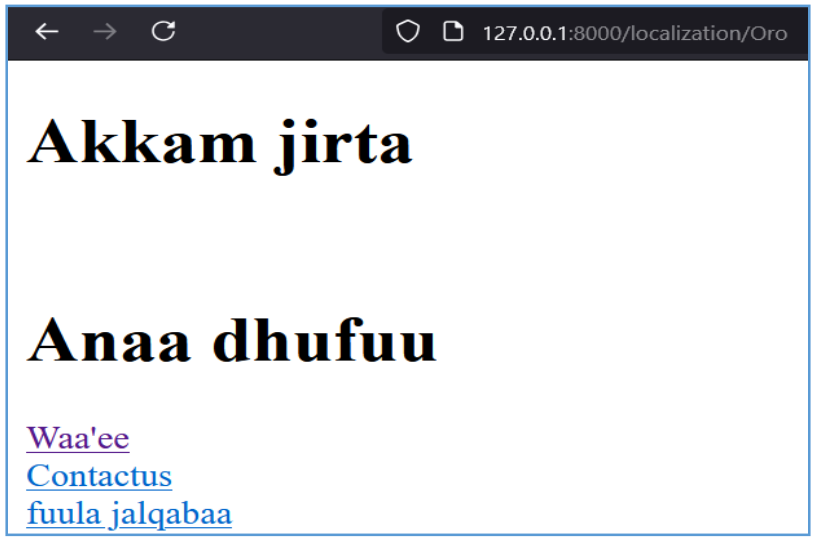

127.0.0.1:8000/localization
How are You...
Well Come ...
About
Contactus
Home

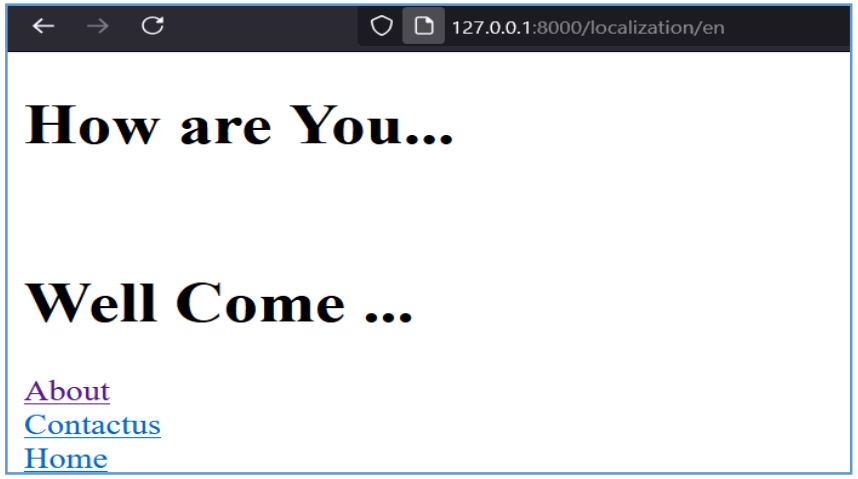
Localization For Specific Route



Cont. ...


127.0.0.1:8000/localization/Amh
ሰላም!!!
እንኳን ደህና መጣህ!!!
ስለ
ያግኙን
መነሻ ገጽ

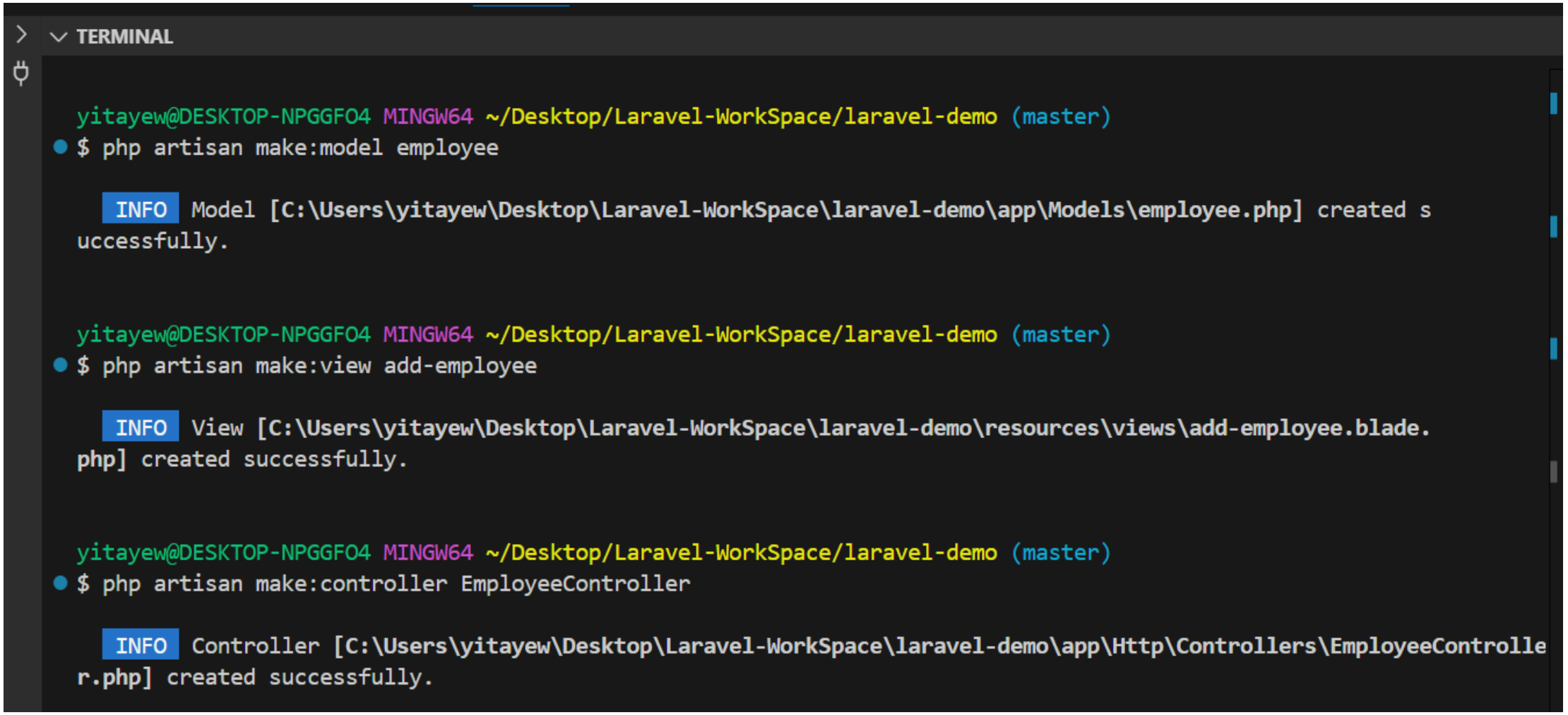

127.0.0.1:8000/localization/Oro
Akkam jirta
Anaa dhufuu
Waa'ee
Contactus
fuula jalqabaa


127.0.0.1:8000/localization/en
How are You...
Well Come ...
About
Contactus
Home

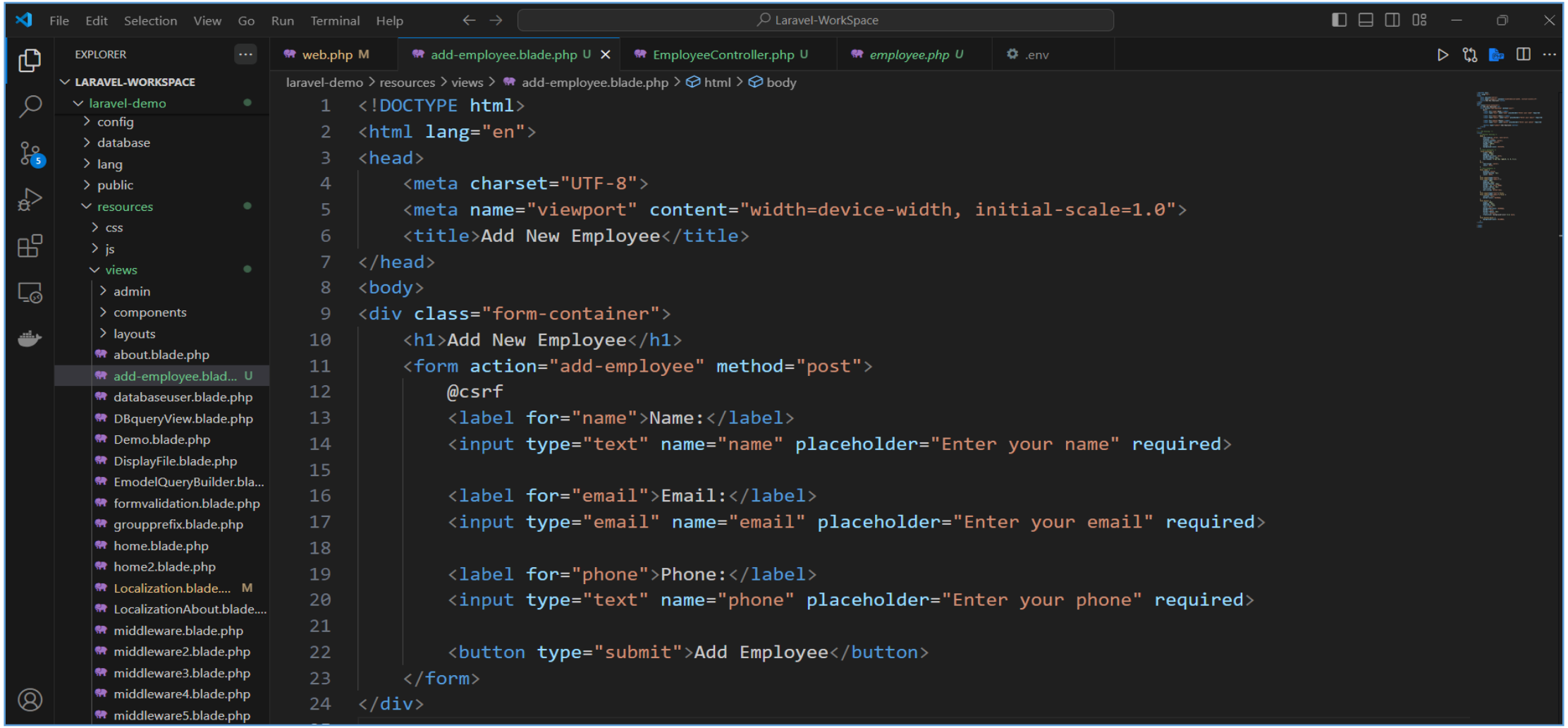
CRUD Operation (Insert Data in MySQL Table With HTML form)

- Inserting data into a MySQL table through an HTML form in Laravel 11 involves several steps, including setting up the route, creating a controller method, defining the form view, and creating the database model. Here’s a guide on how to accomplish this.

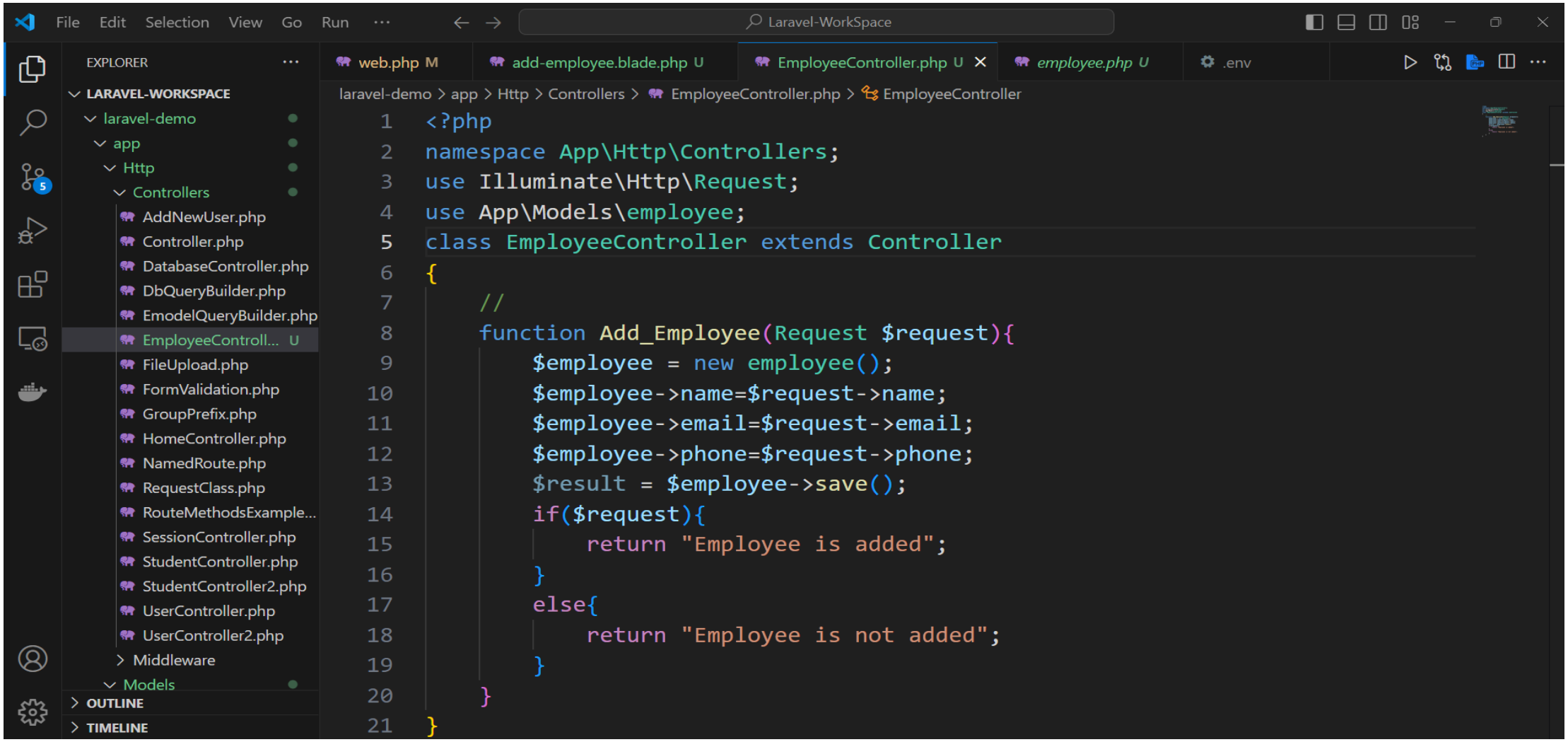
Creating (Model, View, Controller)



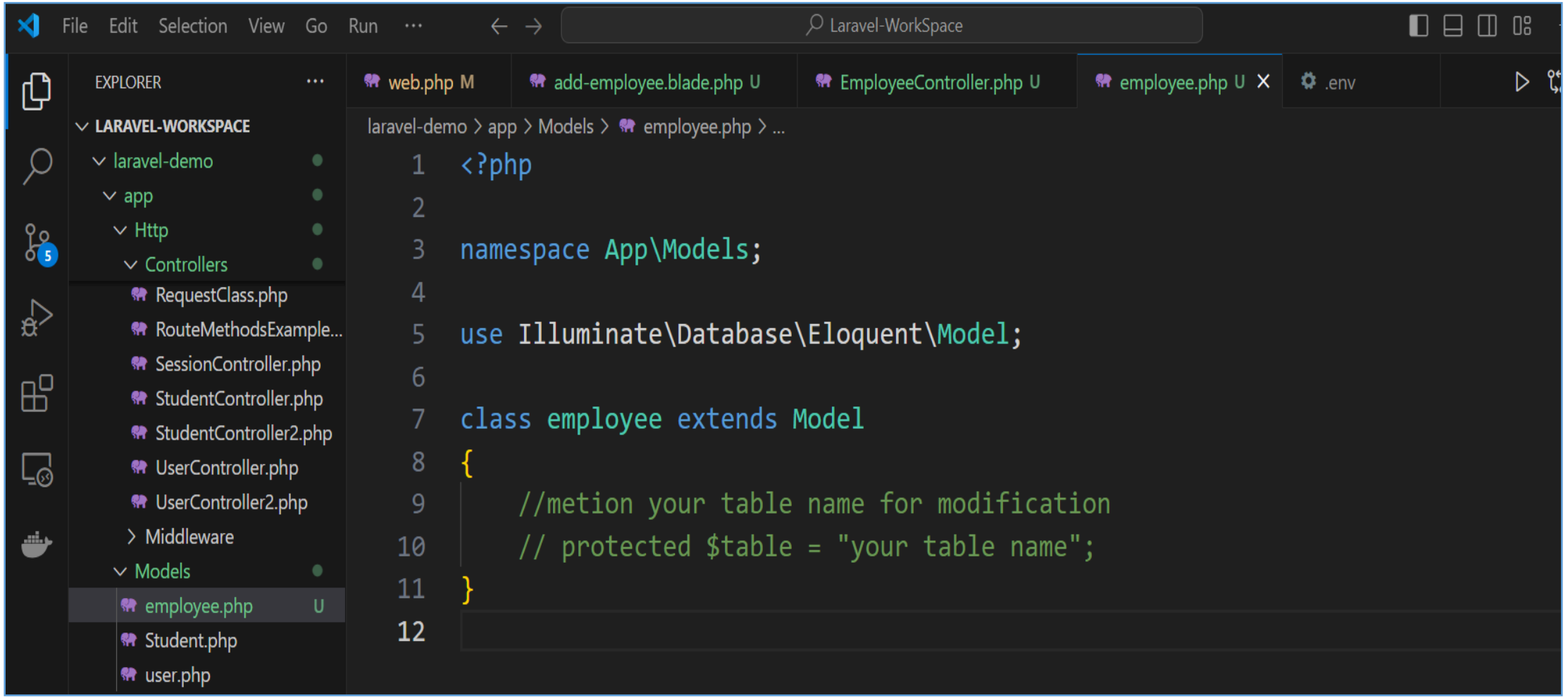
View (add-employee)



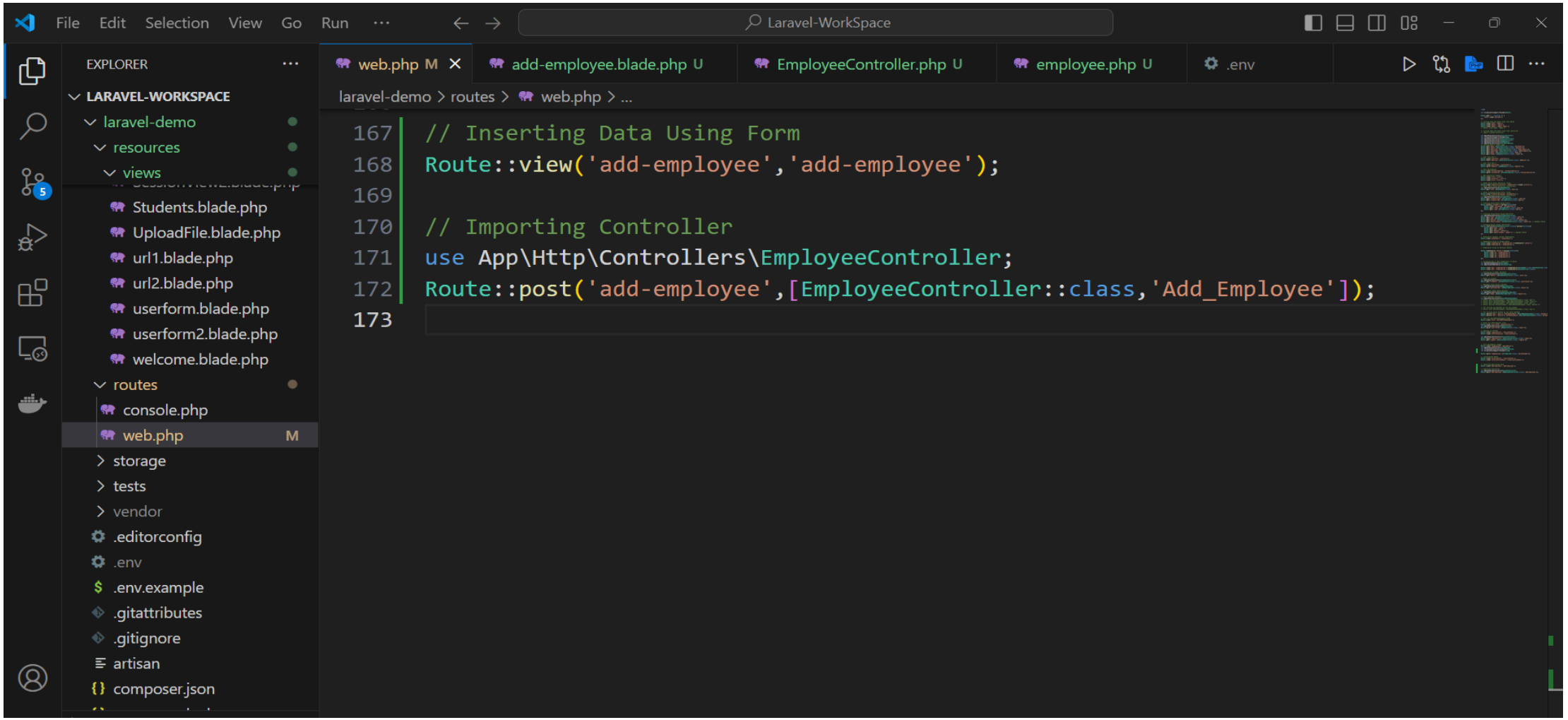
Controller (EmployeeController)



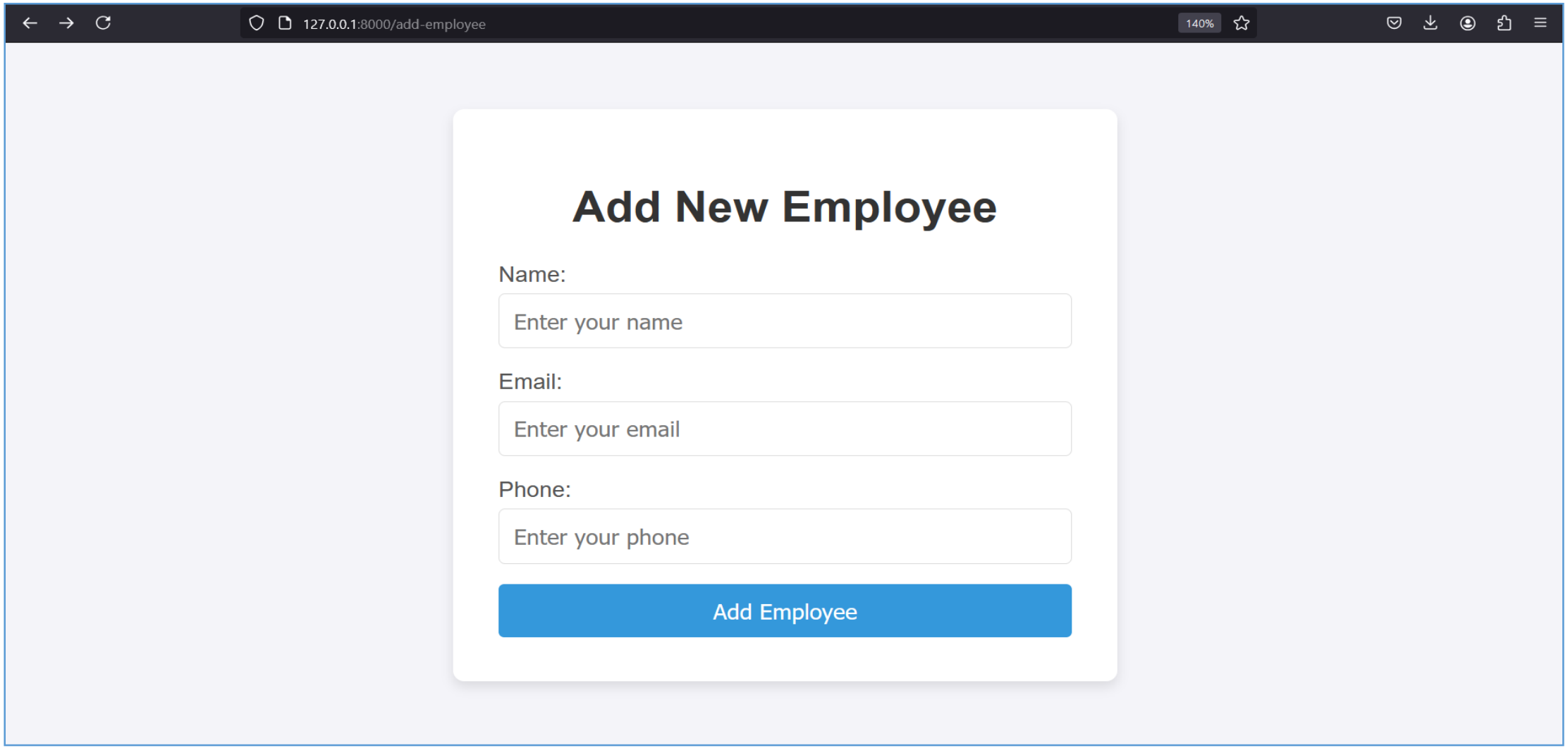
Model (employee)



Route


File Edit Selection View Go Run
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
Students.blade.php
UploadFile.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
.env.example
.gitattributes
.gitignore
artisan
composer.json
web.php M
add-employee.blade.php U
EmployeeController.php U
employee.php U
.env
laravel-demo > routes > web.php > ...
167 // Inserting Data Using Form
168 Route::view('add-employee','add-employee');
169
170 // Importing Controller
171 use App\Http\Controllers\EmployeeController;
172 Route::post('add-employee',[EmployeeController::class,'Add_Employee']);
173

Output


127.0.0.1:8000/add-employee
140%
Add New Employee
Name:
Enter your name
Email:
Enter your email
Phone:
Enter your phone
Add Employee

Cont. ...

Showing rows 0 - 3 (4 total, Query took 0.0002 seconds.)

```
SELECT * FROM `employees`
```

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

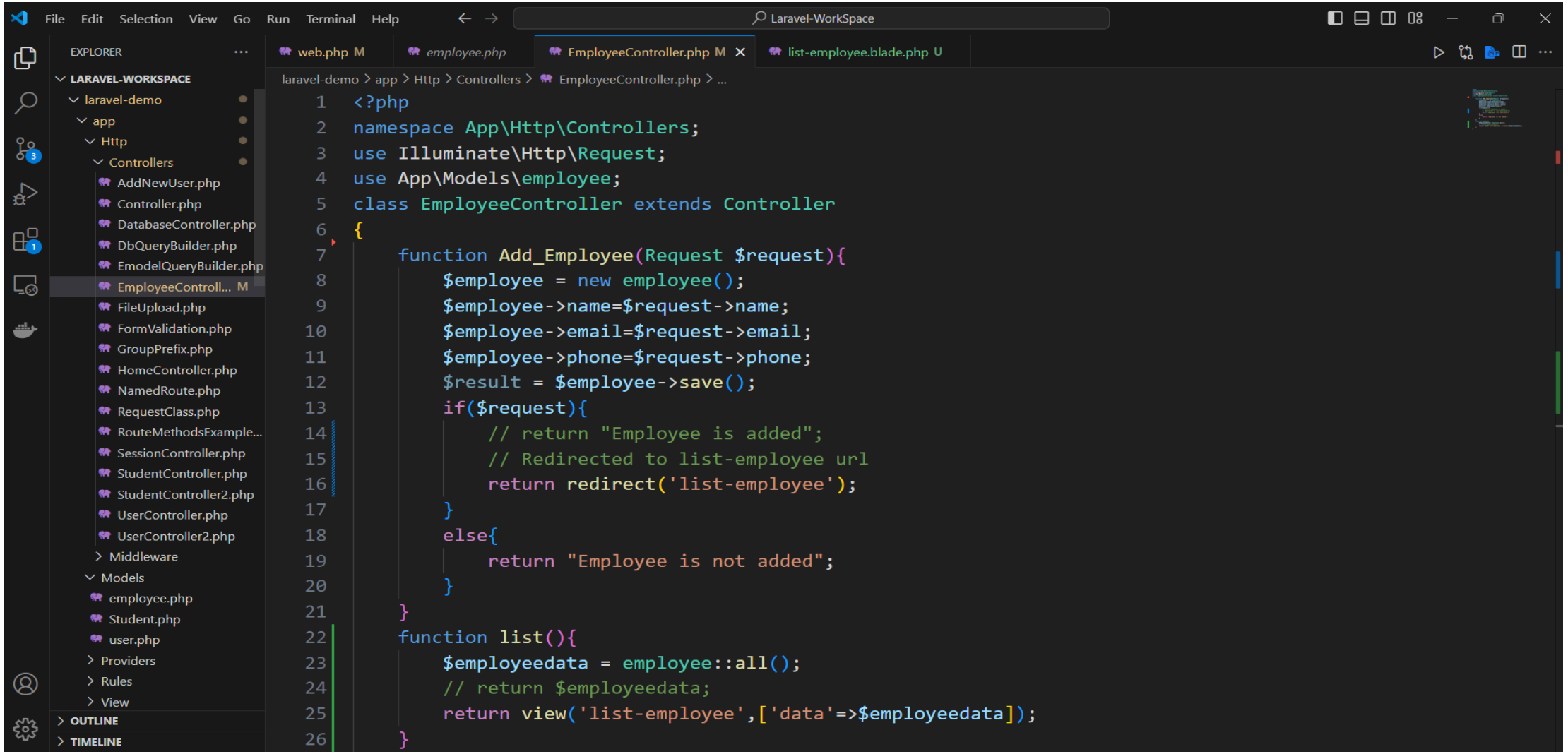
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

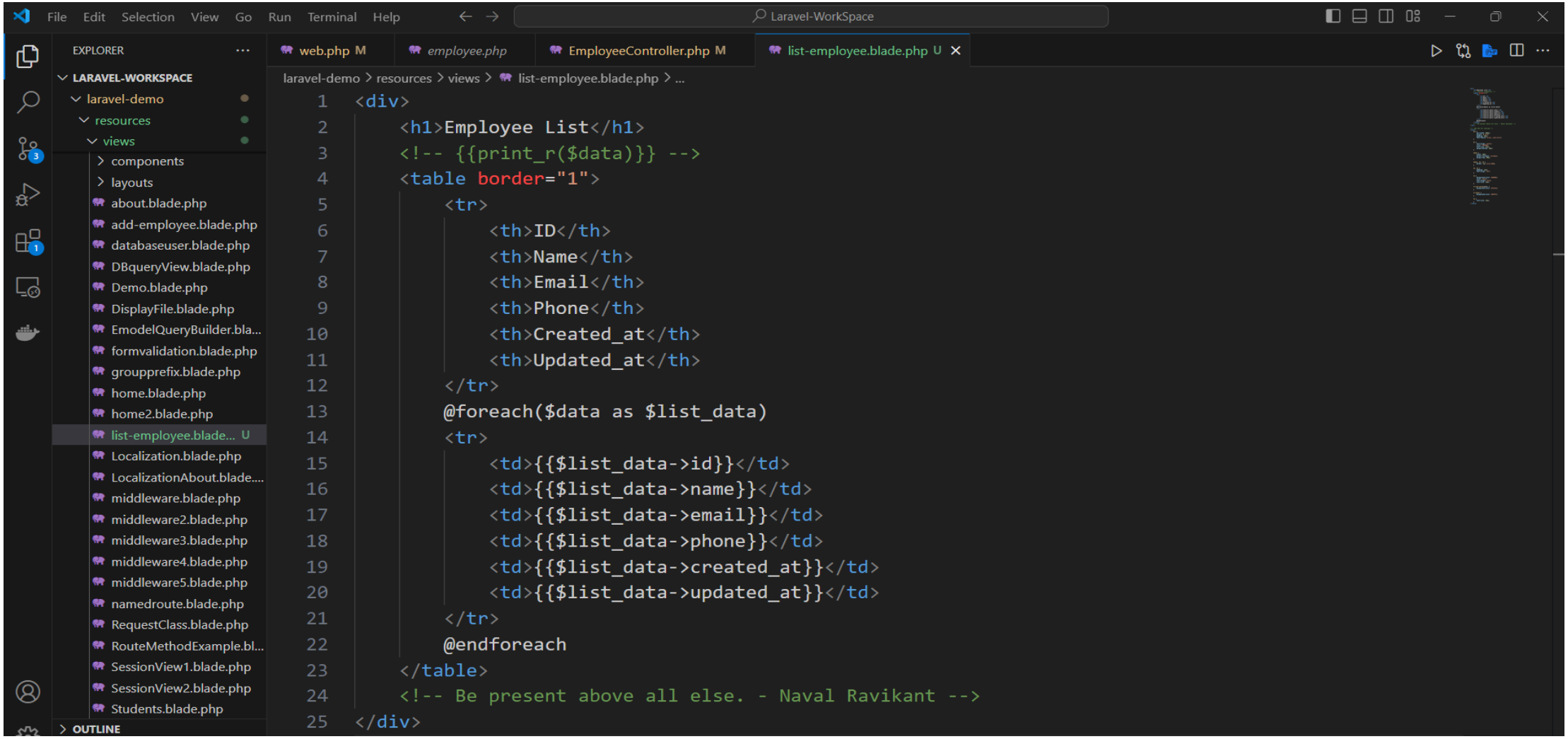
	id	name	email	phone	created_at	updated_at
Edit Copy Delete	1	Yitayew Solomon	yitayewsolomon3@gmail.com	0937559718	2024-11-12	2024-11-12
Edit Copy Delete	2	Haileab Solomon	haileabsolomon@gmail.com	0937559718	2024-11-12	2024-11-12
Edit Copy Delete	3	Yared Solomon	yaredsolomon@gmail.com	0937559718	2024-11-12	2024-11-12
Edit Copy Delete	6	Natanim Yitayew	natan@gmail.com	0937559720	2024-11-12	2024-11-12

Check all *With selected:* Edit Copy Delete Export

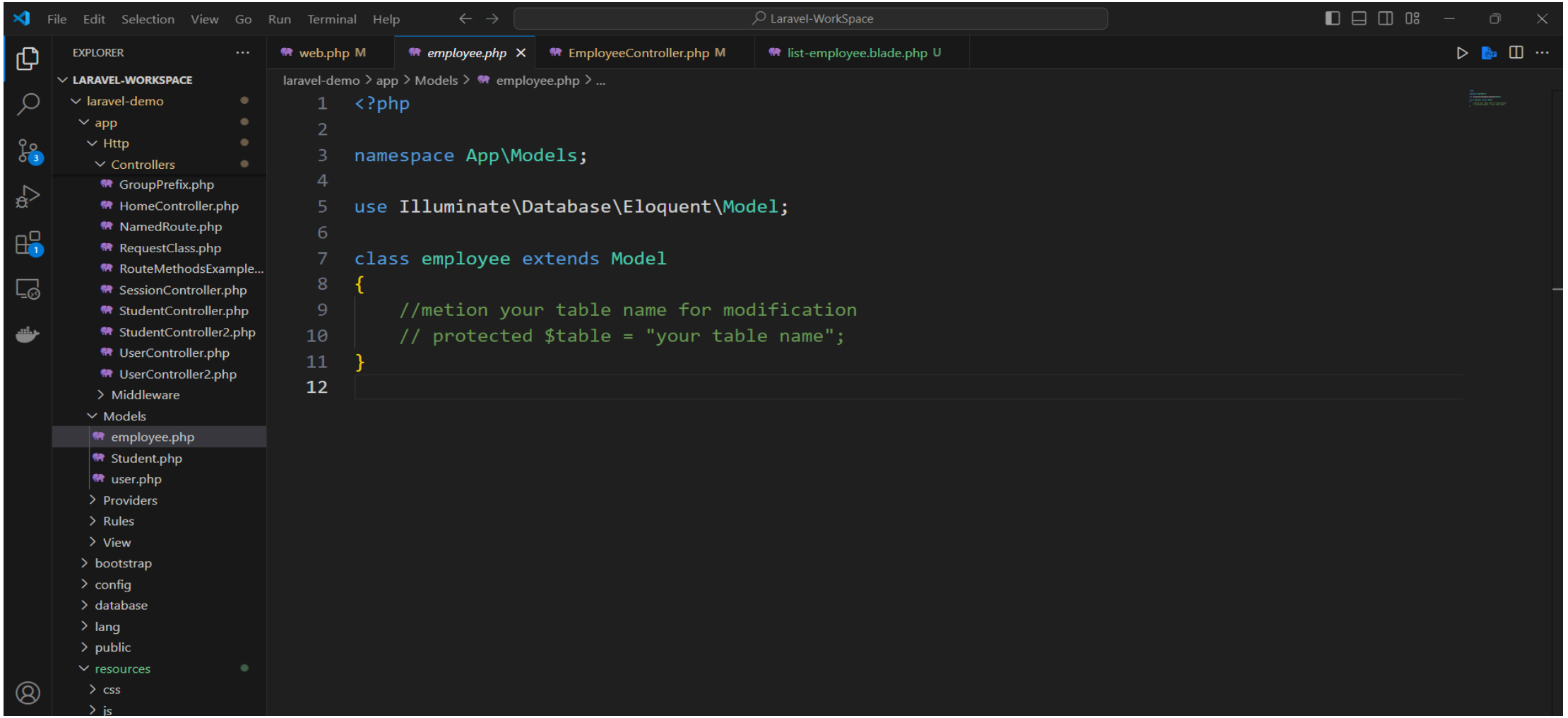
Get and Display Data From MySql Table



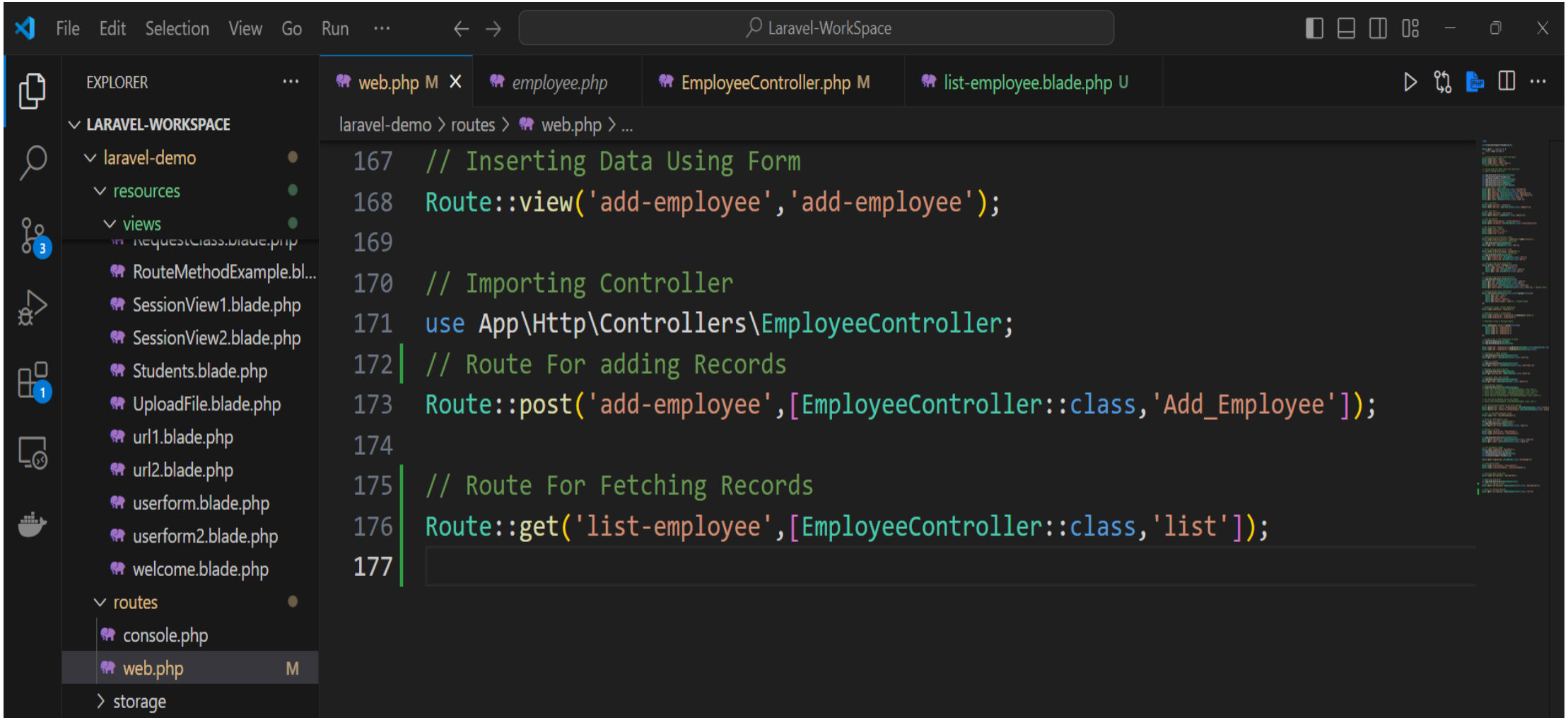
View (List-employee)



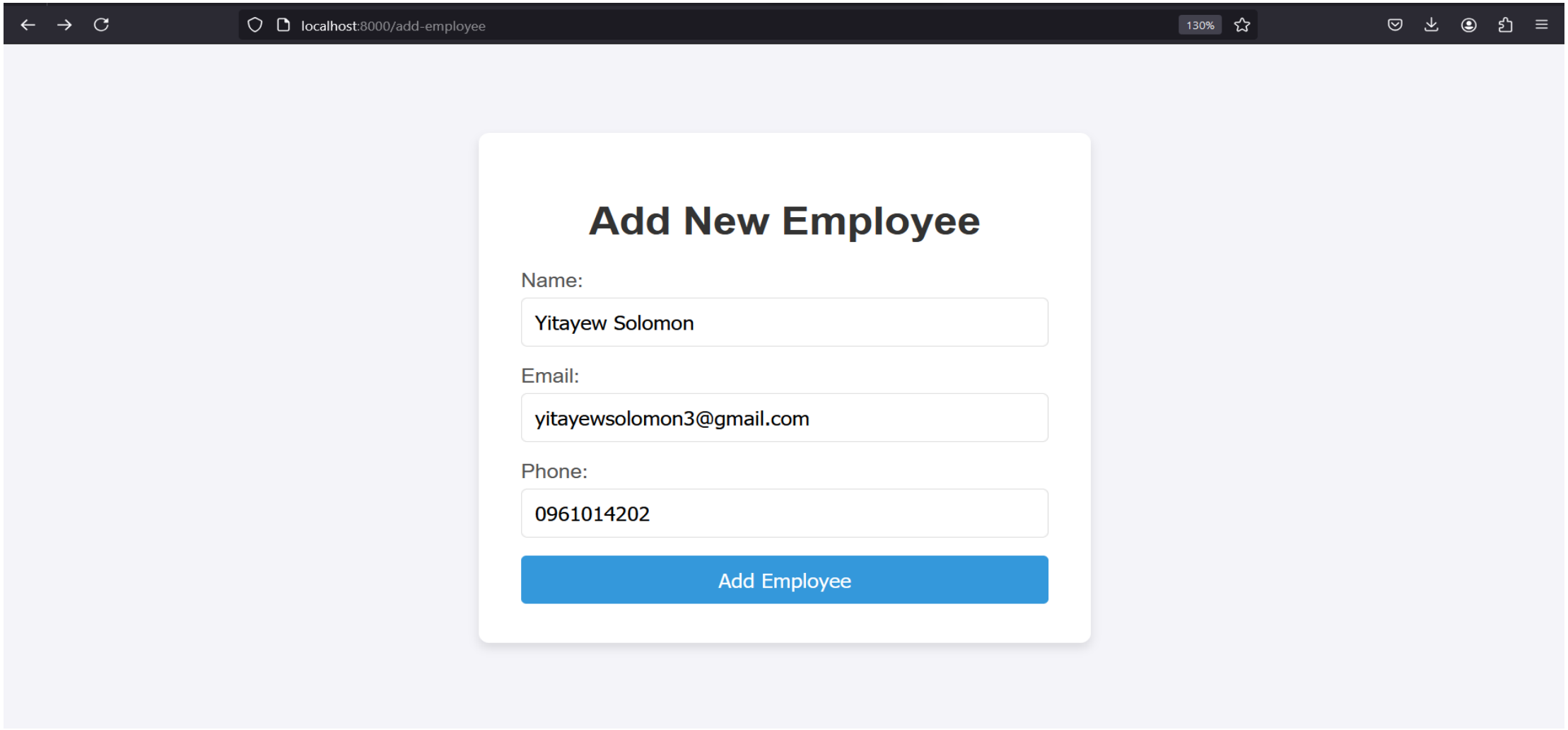
Model (Employee)



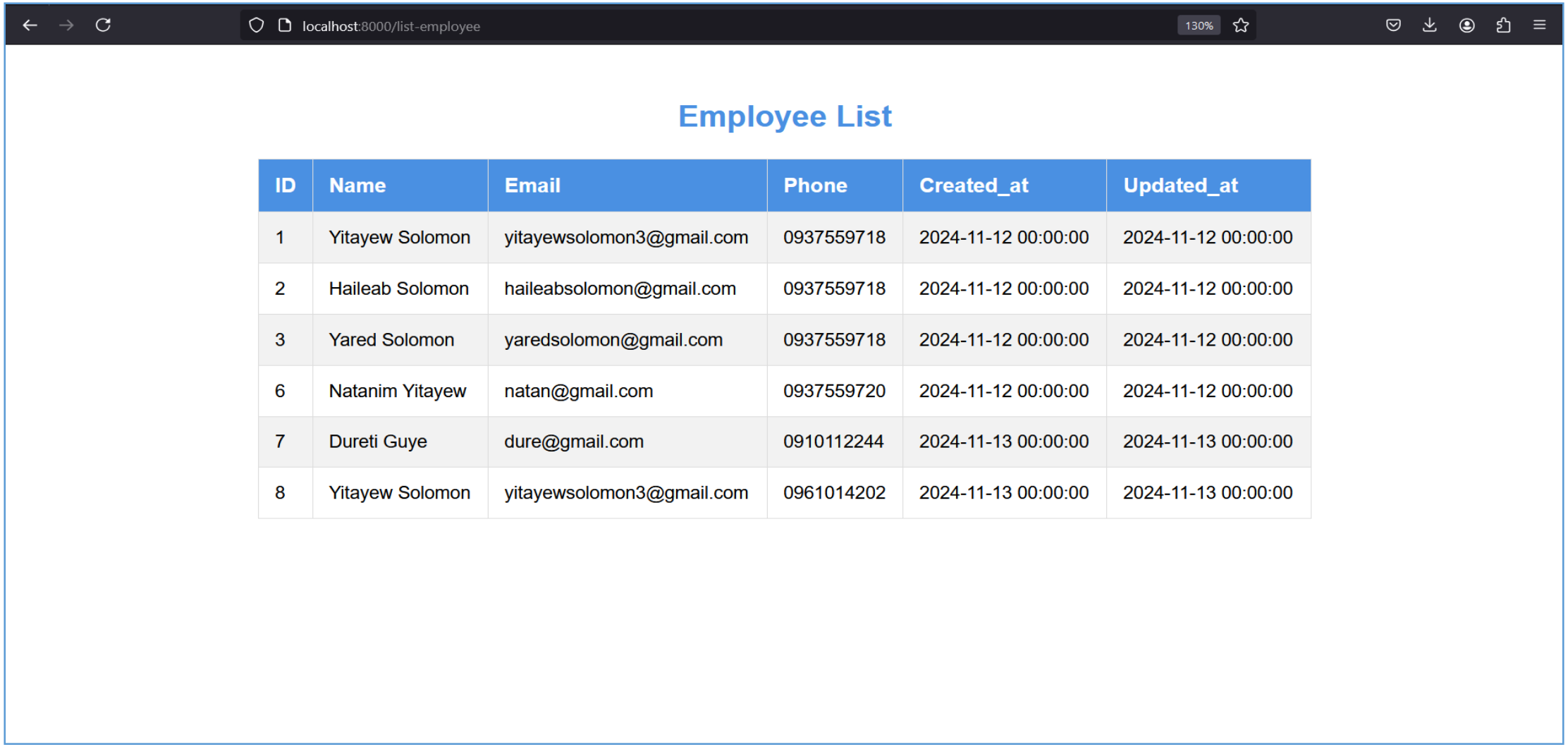
Route


File Edit Selection View Go Run
Laravel-WorkSpace
EXPLORER
web.php M
employee.php
EmployeeController.php M
list-employee.blade.php U
laravel-demo > routes > web.php > ...
LARAVEL-WORKSPACE
laravel-demo
resources
views
RouteMethodExample.bl...
SessionView1.blade.php
SessionView2.blade.php
Students.blade.php
UploadFile.blade.php
url1.blade.php
url2.blade.php
userform.blade.php
userform2.blade.php
welcome.blade.php
routes
console.php
web.php M
storage
167 // Inserting Data Using Form
168 Route::view('add-employee','add-employee');
169
170 // Importing Controller
171 use App\Http\Controllers\EmployeeController;
172 // Route For adding Records
173 Route::post('add-employee',[EmployeeController::class,'Add_Employee']);
174
175 // Route For Fetching Records
176 Route::get('list-employee',[EmployeeController::class,'list']);
177

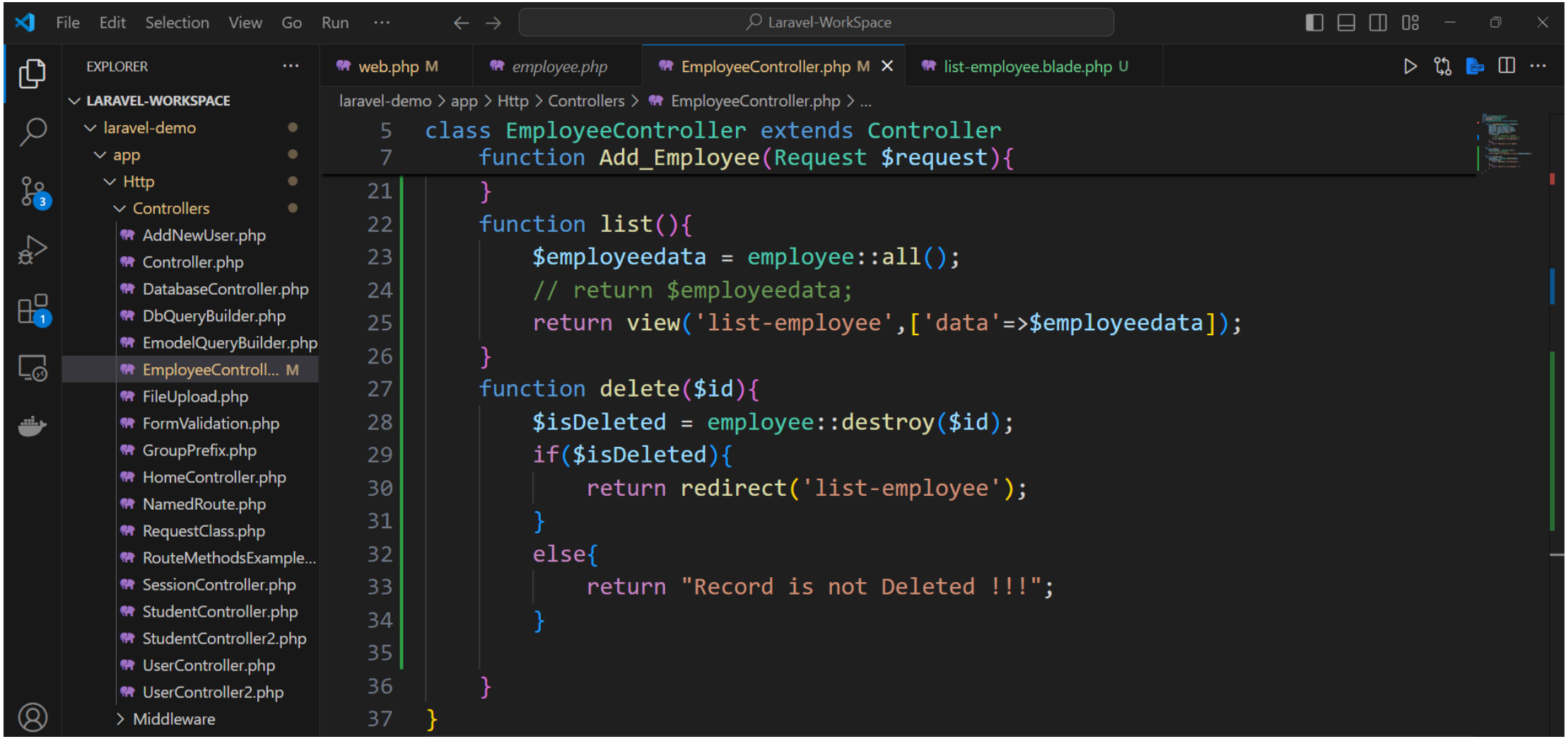
Output (http://localhost:8000/add-employee)



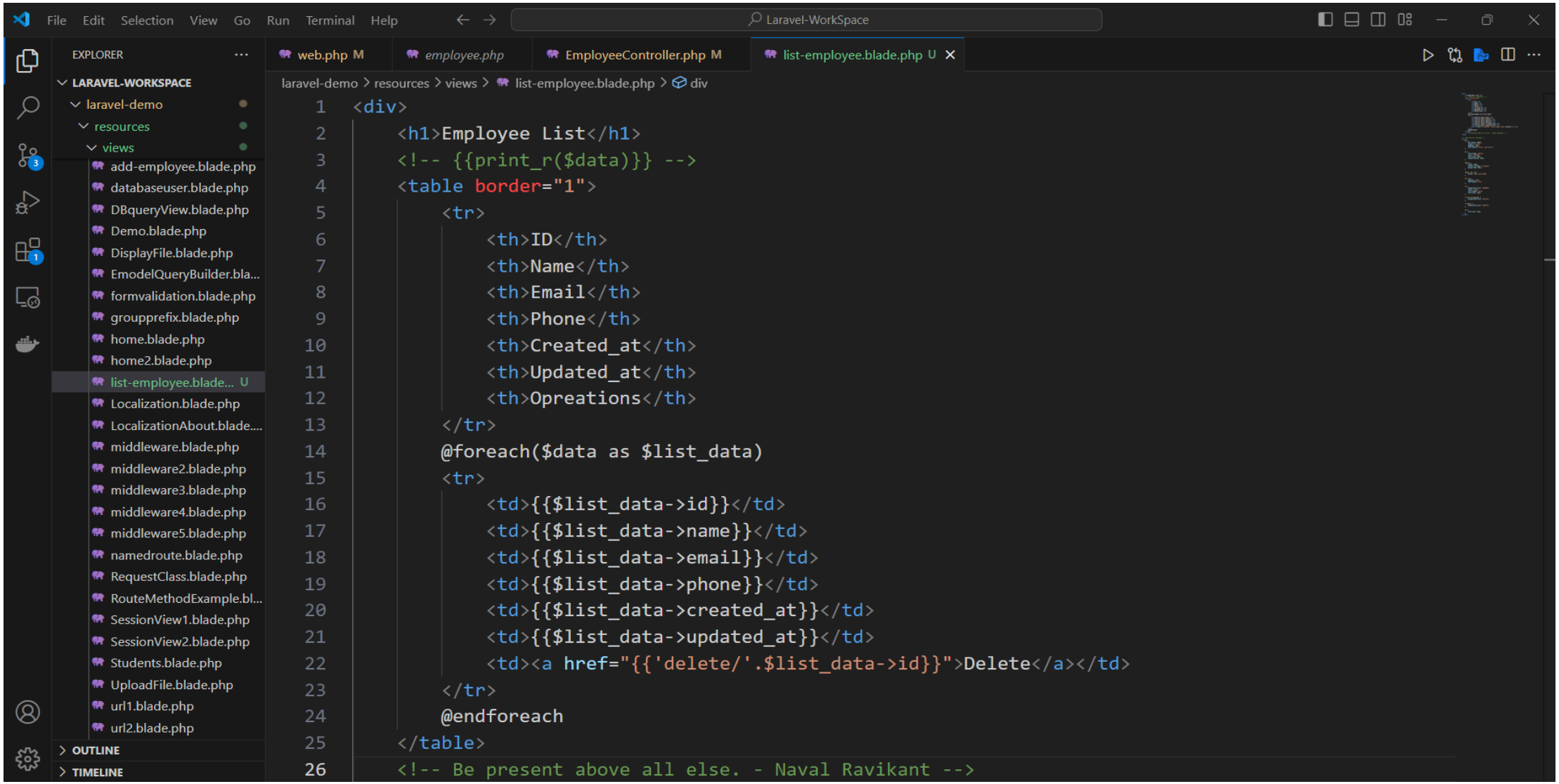
Cont. ... (http://localhost:8000/list-employee)



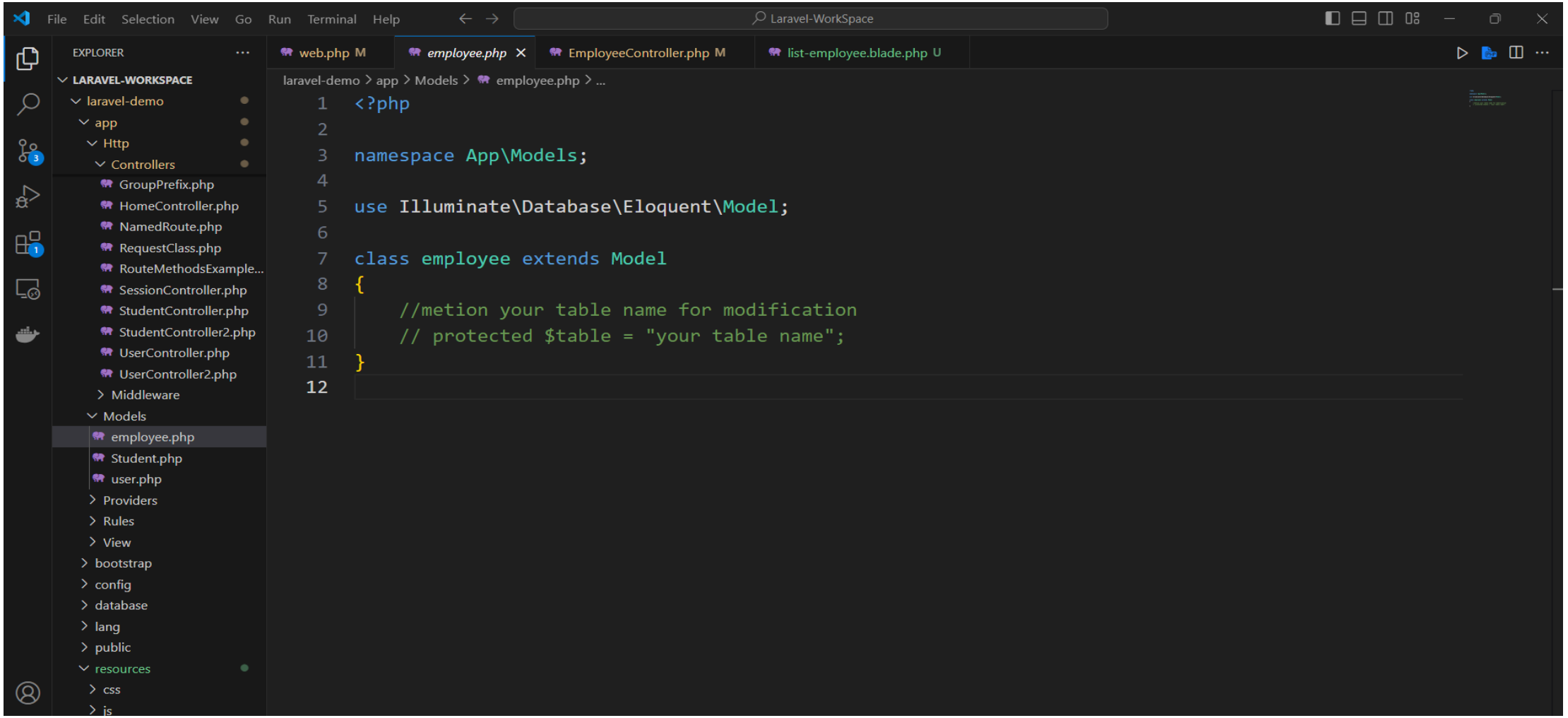
Delete Data From MySql Table



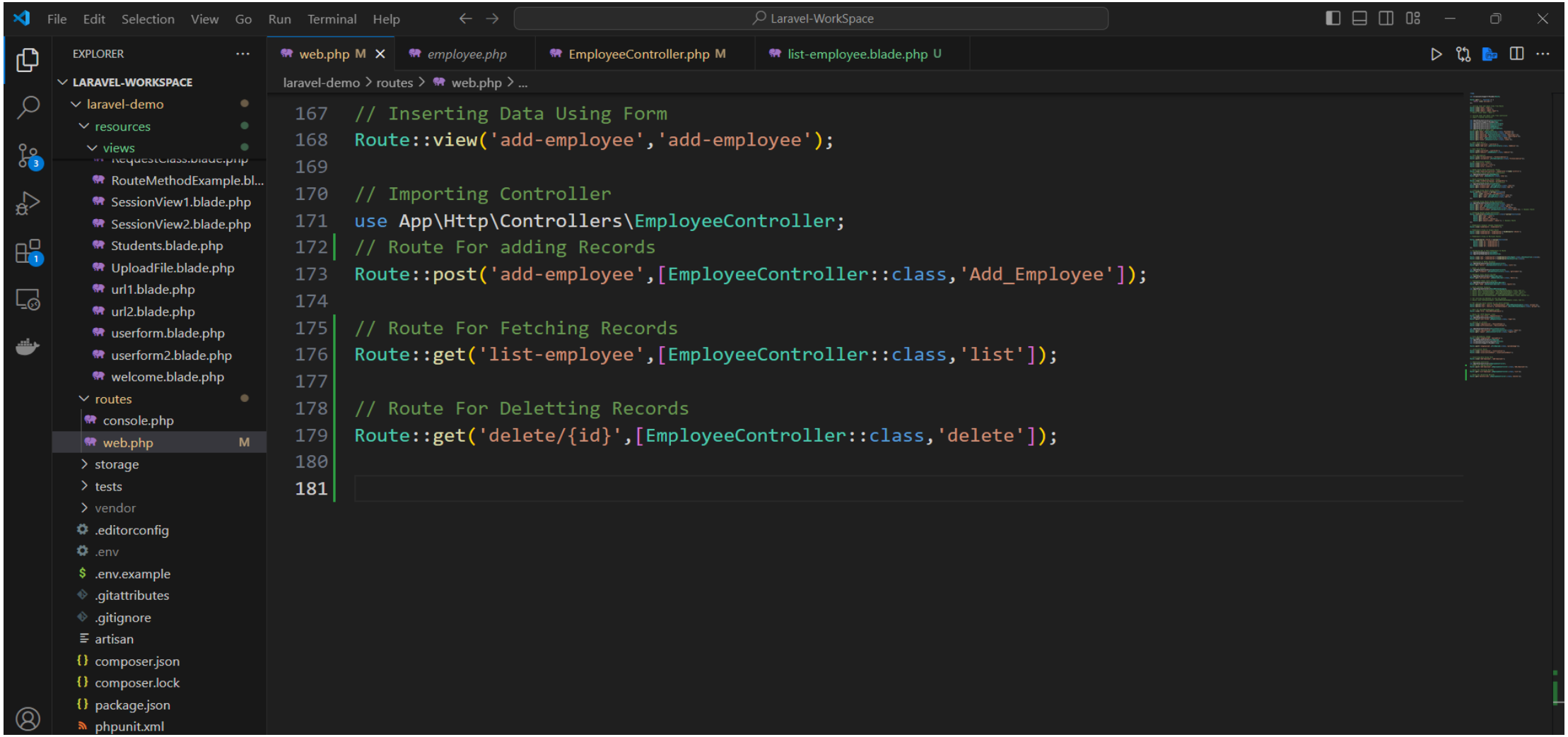
View (List-employee)



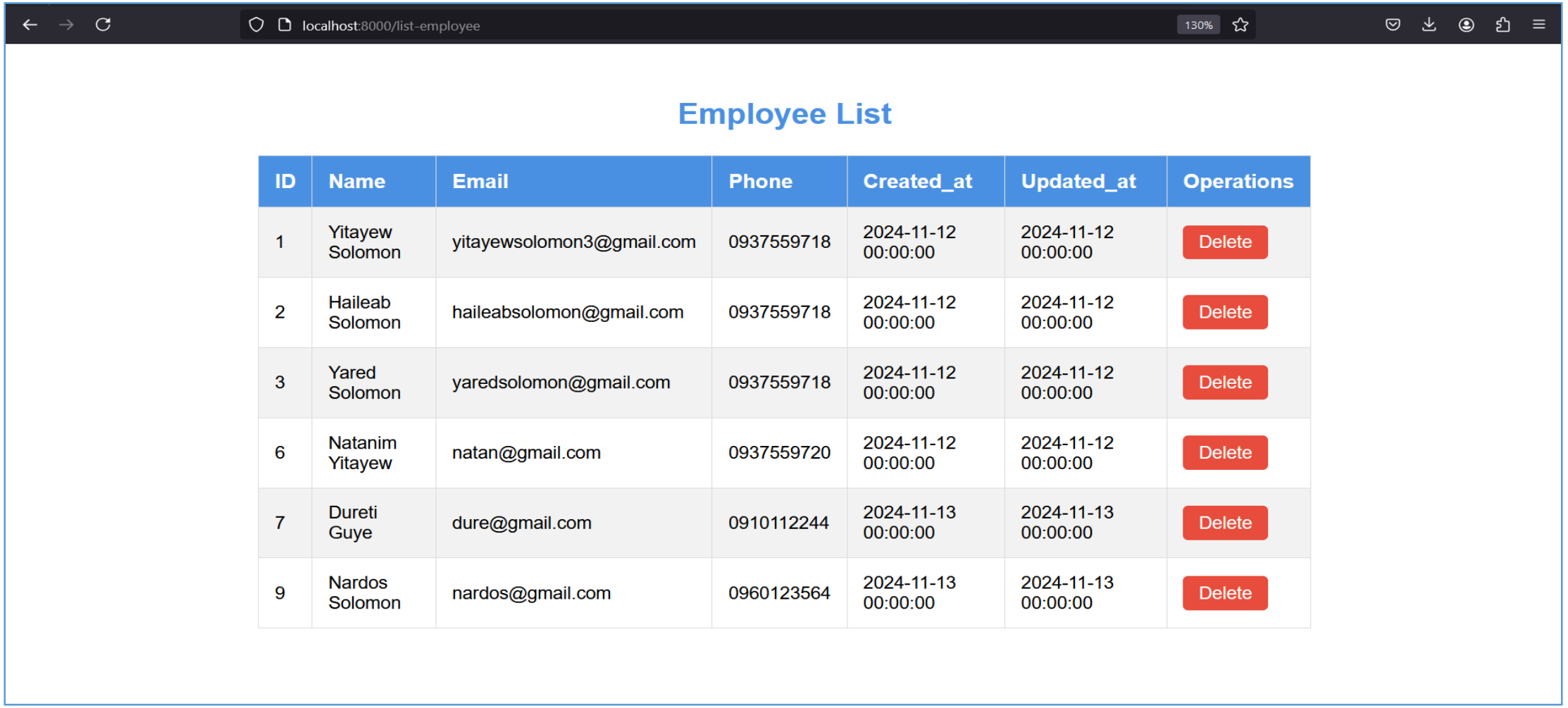
Model (Employee)



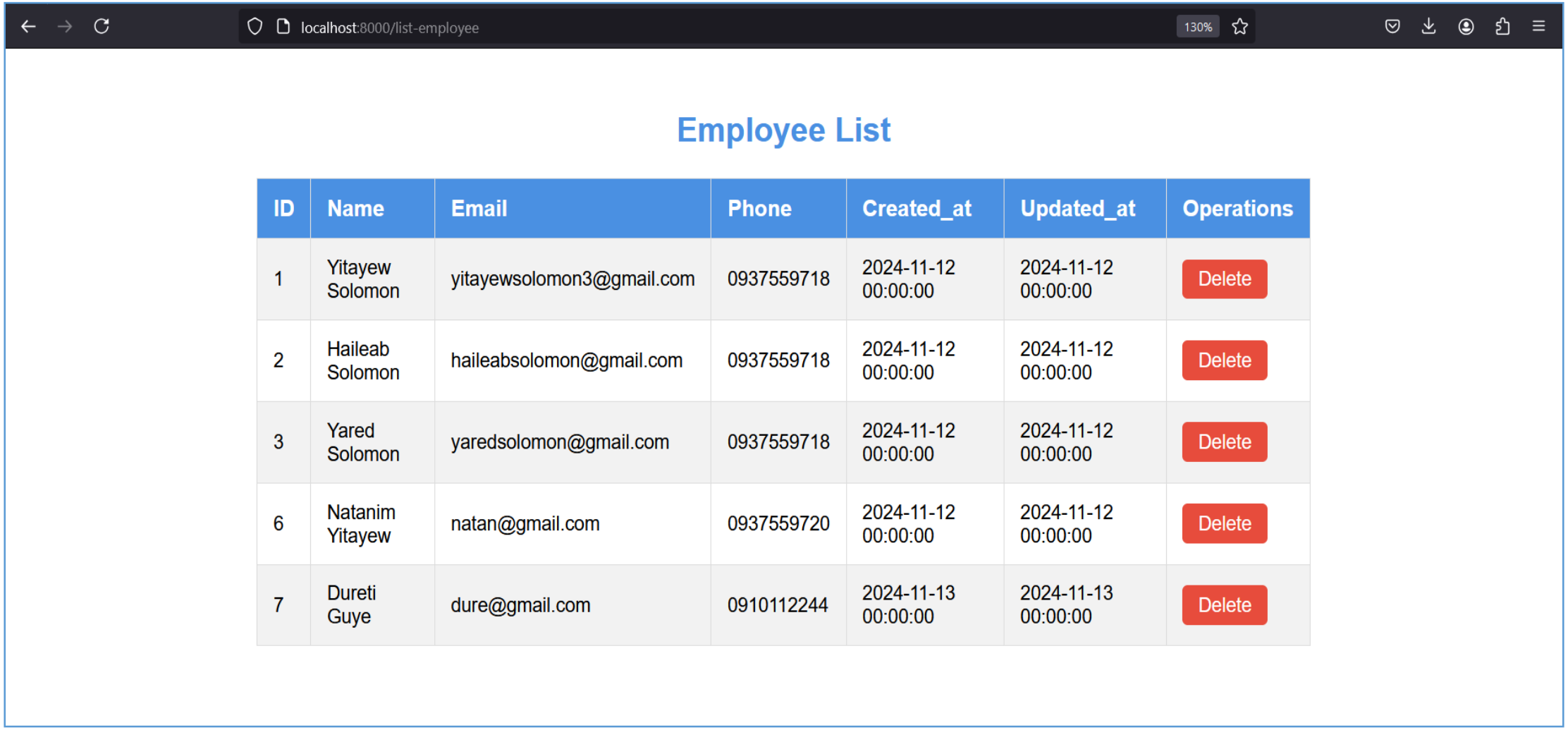
Route


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
RouteMethodExample.bl...
SessionView1.blade.php
SessionView2.blade.php
Students.blade.php
UploadFile.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
.env.example
.gitattributes
.gitignore
artisan
composer.json
composer.lock
package.json
phpunit.xml
web.php M
employee.php
EmployeeController.php M
list-employee.blade.php U
laravel-demo > routes > web.php > ...
// Inserting Data Using Form
Route::view('add-employee','add-employee');
// Importing Controller
use App\Http\Controllers\EmployeeController;
// Route For adding Records
Route::post('add-employee',[EmployeeController::class,'Add_Employee']);
// Route For Fetching Records
Route::get('list-employee',[EmployeeController::class,'list']);
// Route For Deletting Records
Route::get('delete/{id}',[EmployeeController::class,'delete']);

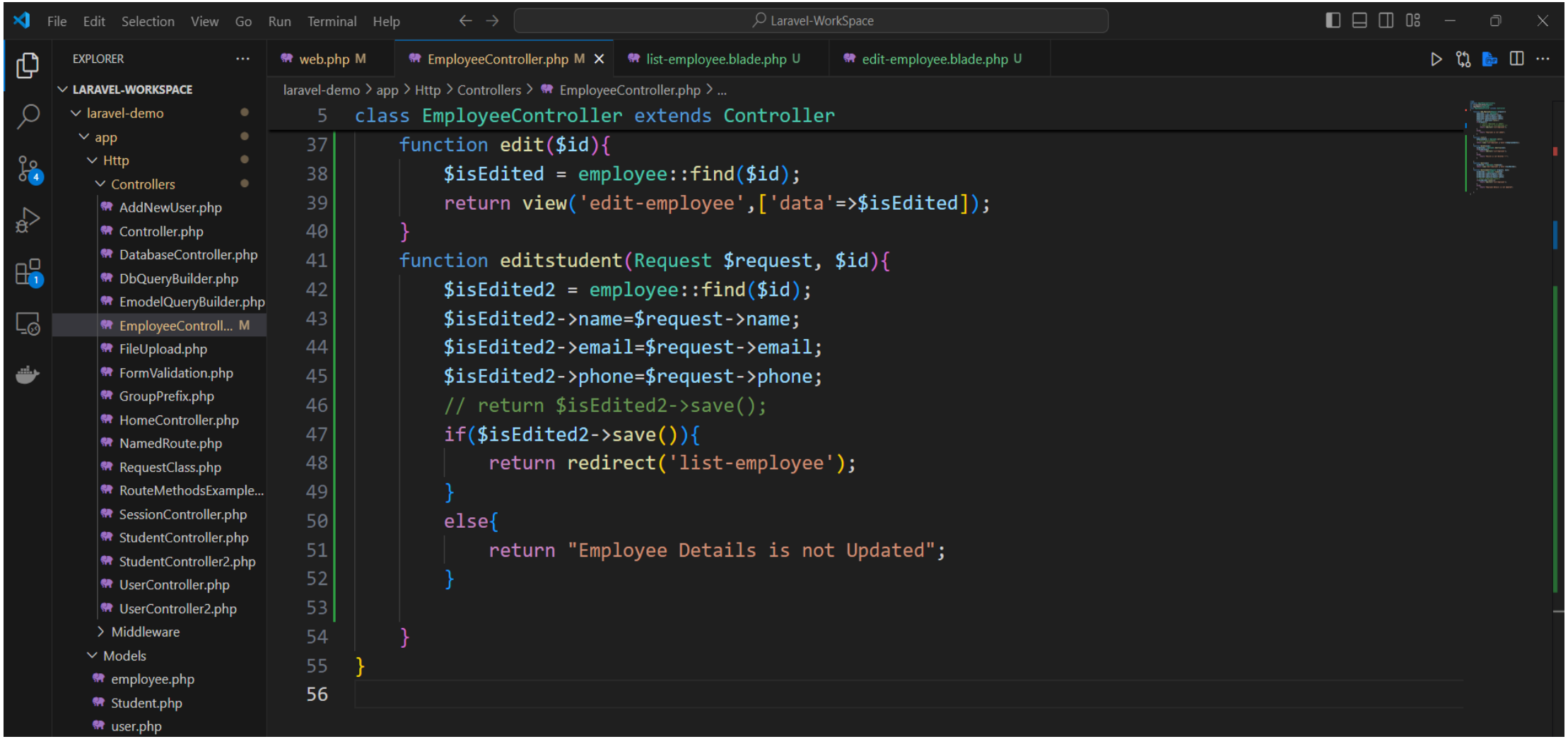
Output (http://localhost:8000/list-employee)



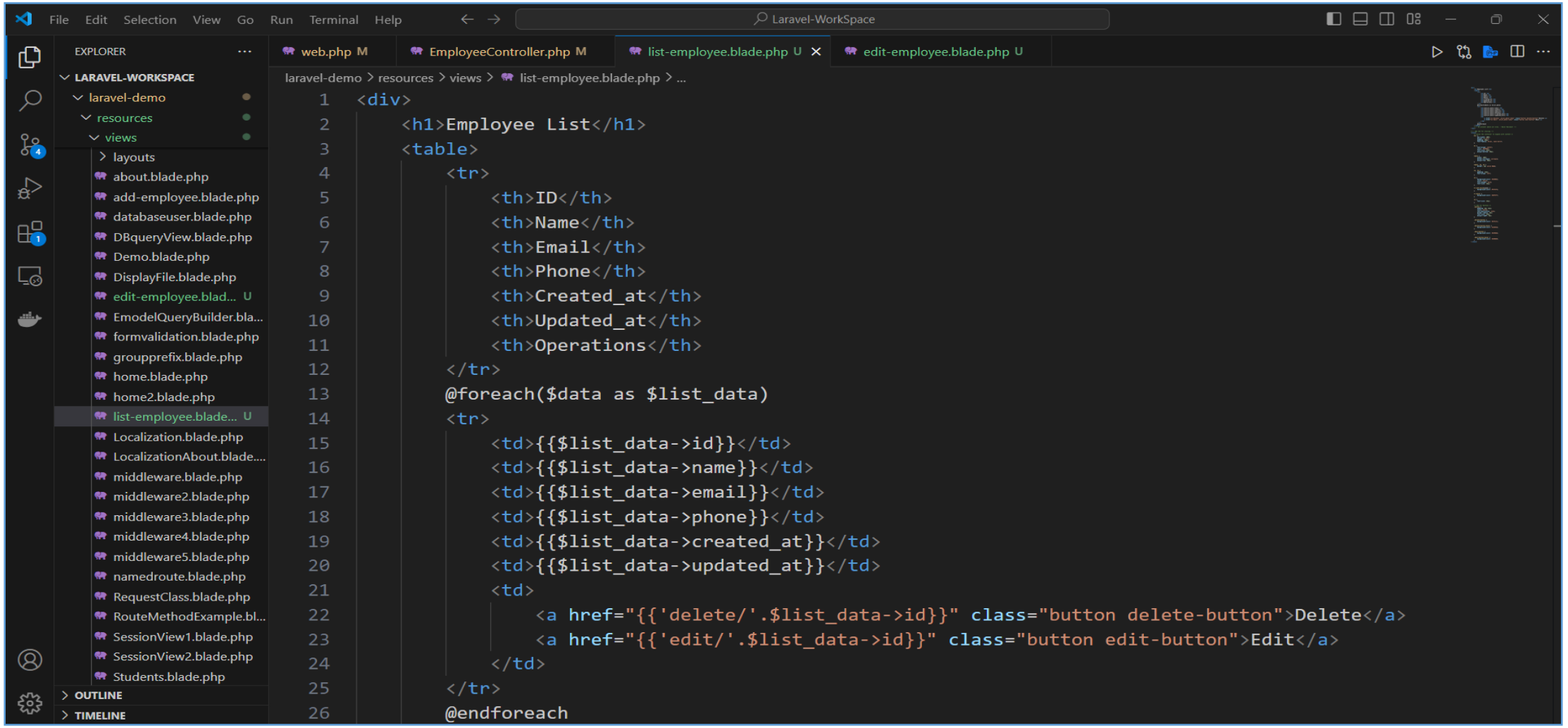
Output (After Deleting)



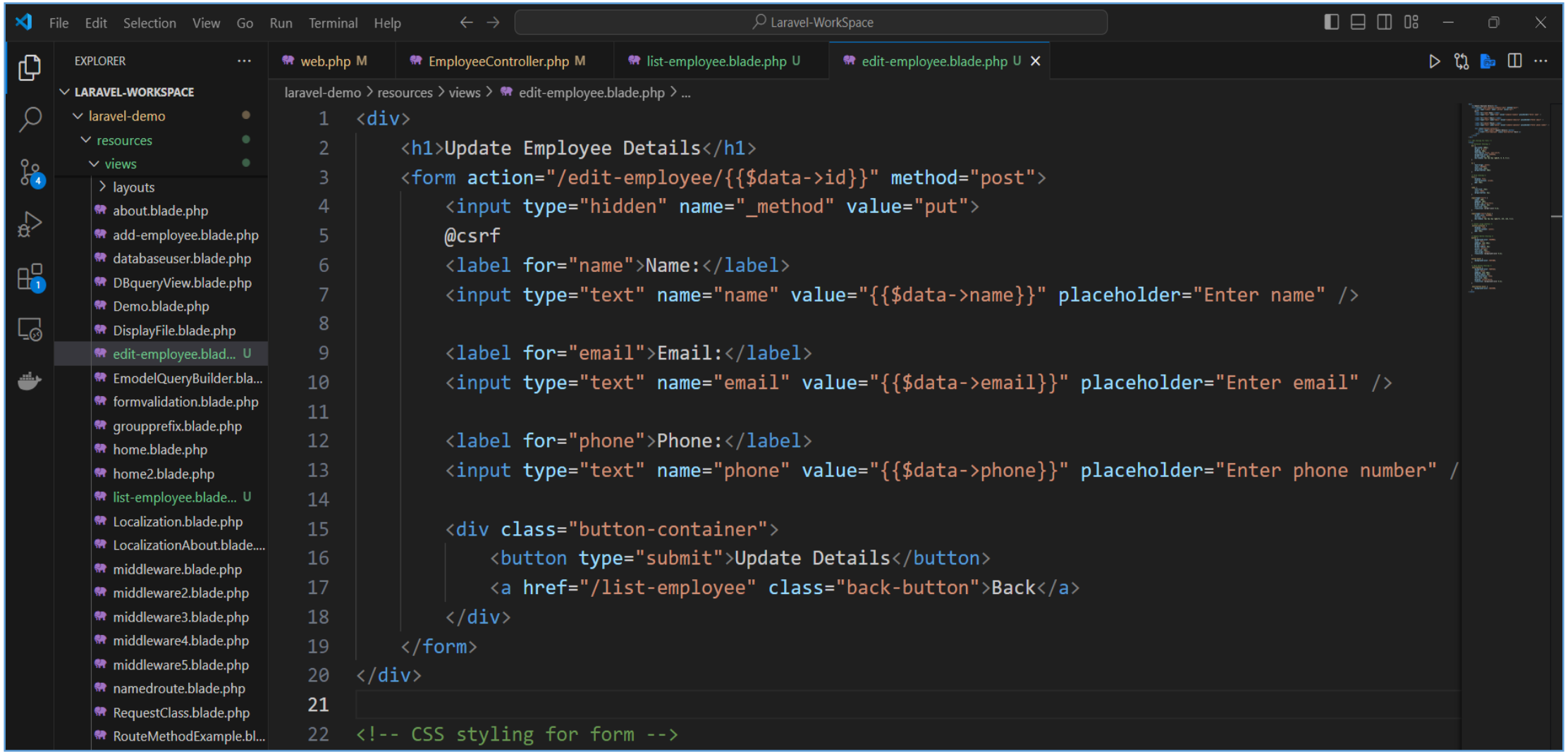
Populate Data in HTML form from MySql Table



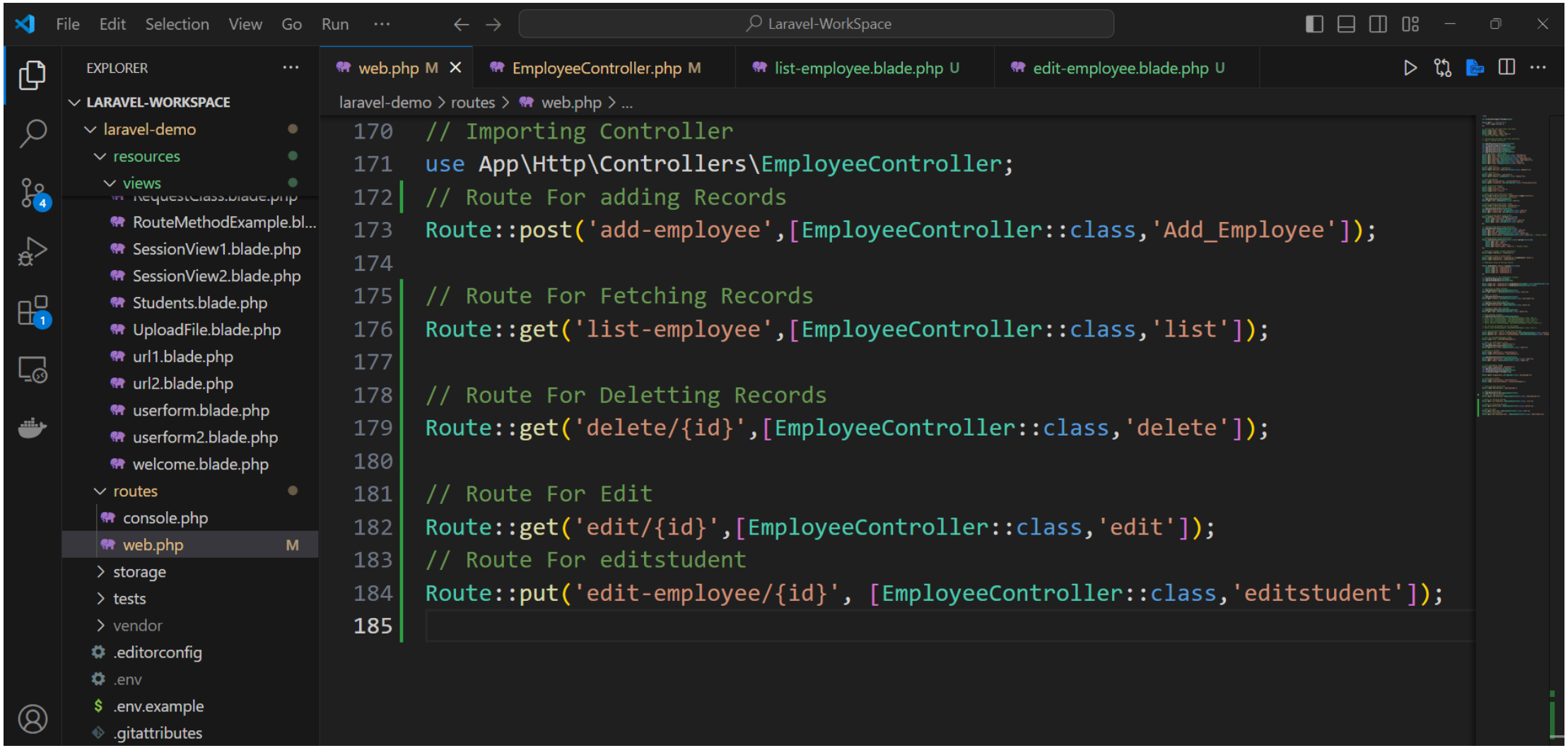
View (List-Employee)



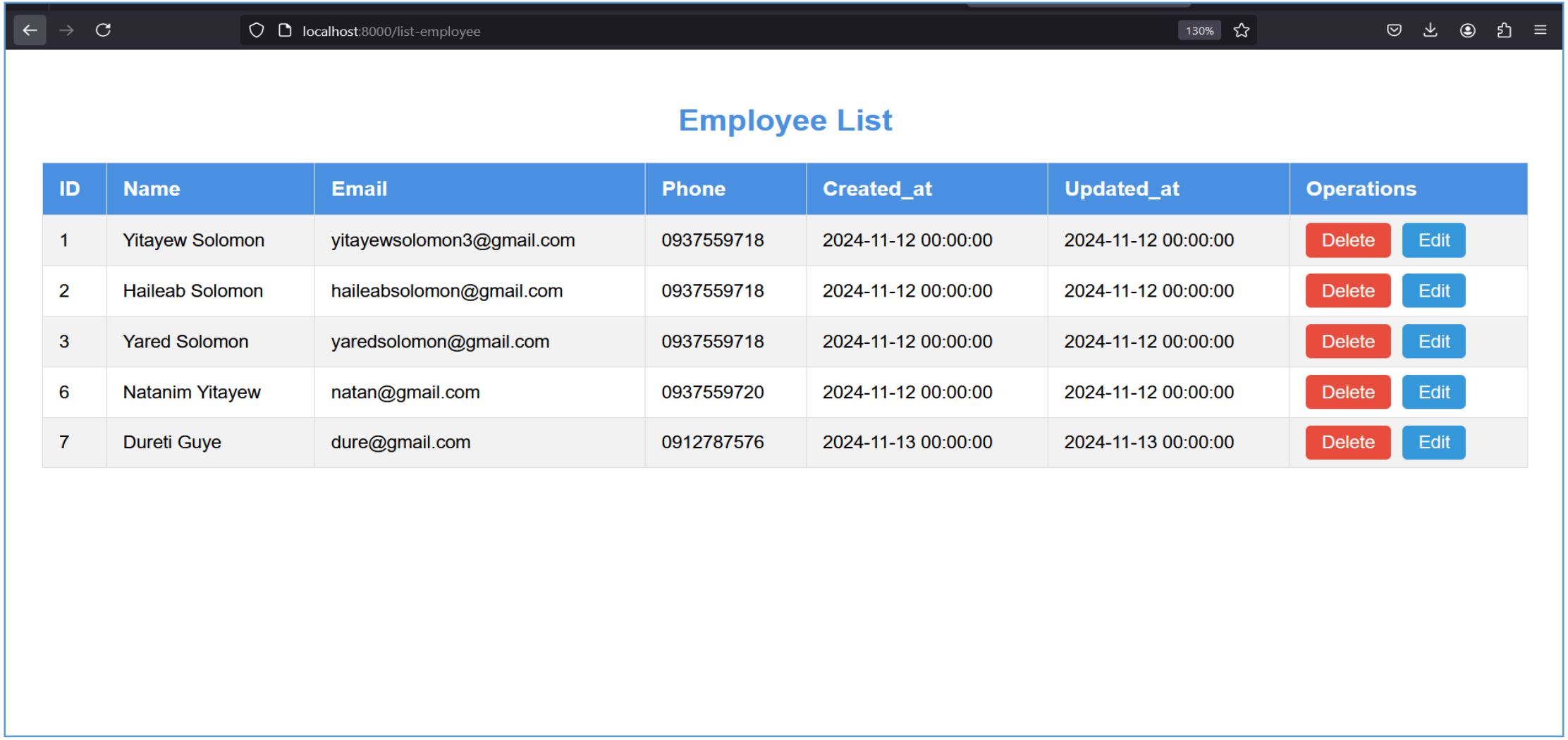
View (edit-employee)



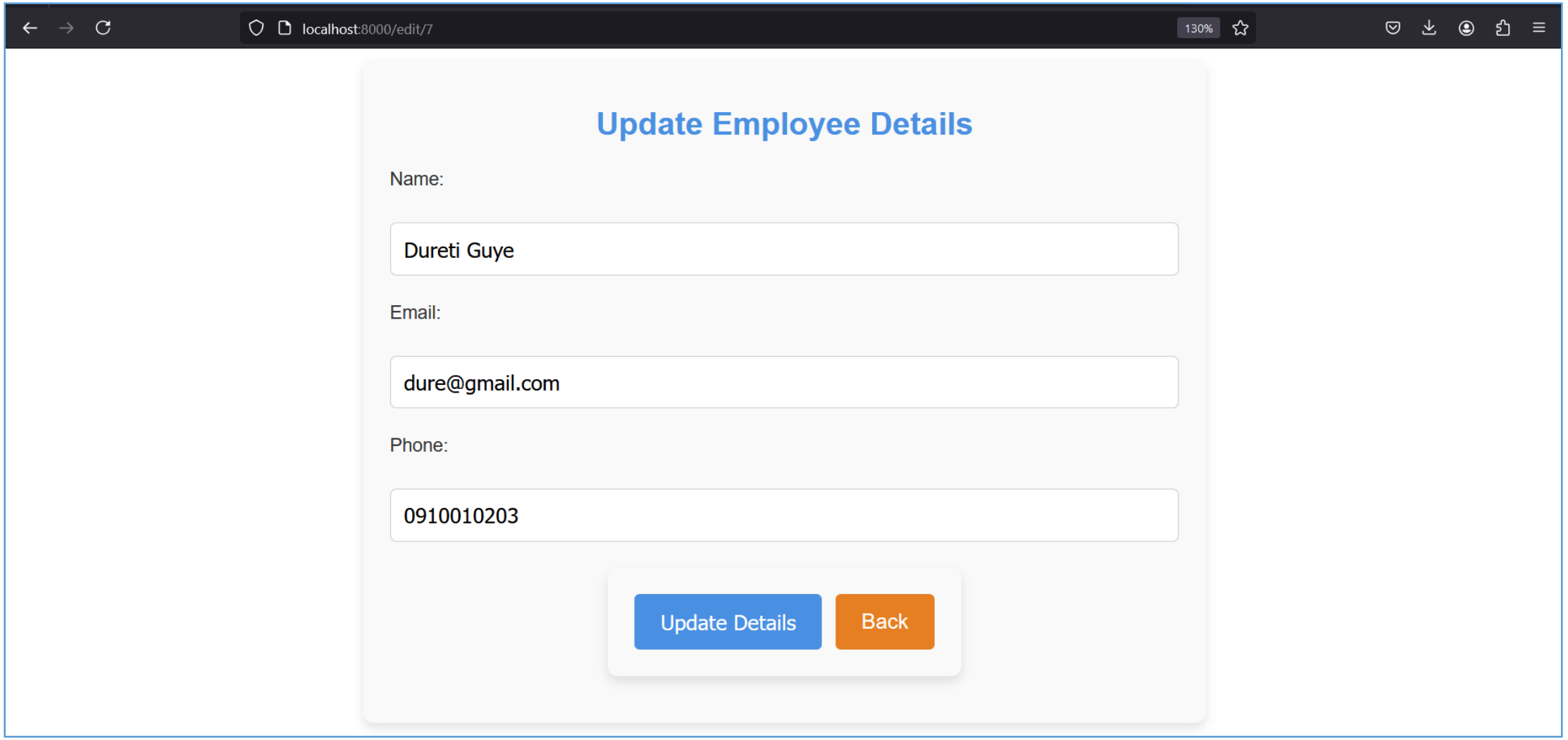
Route


File Edit Selection View Go Run
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
RouteMethodExample.bl...
SessionView1.blade.php
SessionView2.blade.php
Students.blade.php
UploadFile.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
.env.example
.gitattributes
web.php M
EmployeeController.php M
list-employee.blade.php U
edit-employee.blade.php U
laravel-demo > routes > web.php > ...
// Importing Controller
use App\Http\Controllers\EmployeeController;
// Route For adding Records
Route::post('add-employee',[EmployeeController::class,'Add_Employee']);
// Route For Fetching Records
Route::get('list-employee',[EmployeeController::class,'list']);
// Route For Deletting Records
Route::get('delete/{id}',[EmployeeController::class,'delete']);
// Route For Edit
Route::get('edit/{id}',[EmployeeController::class,'edit']);
// Route For editstudent
Route::put('edit-employee/{id}', [EmployeeController::class,'editstudent']);

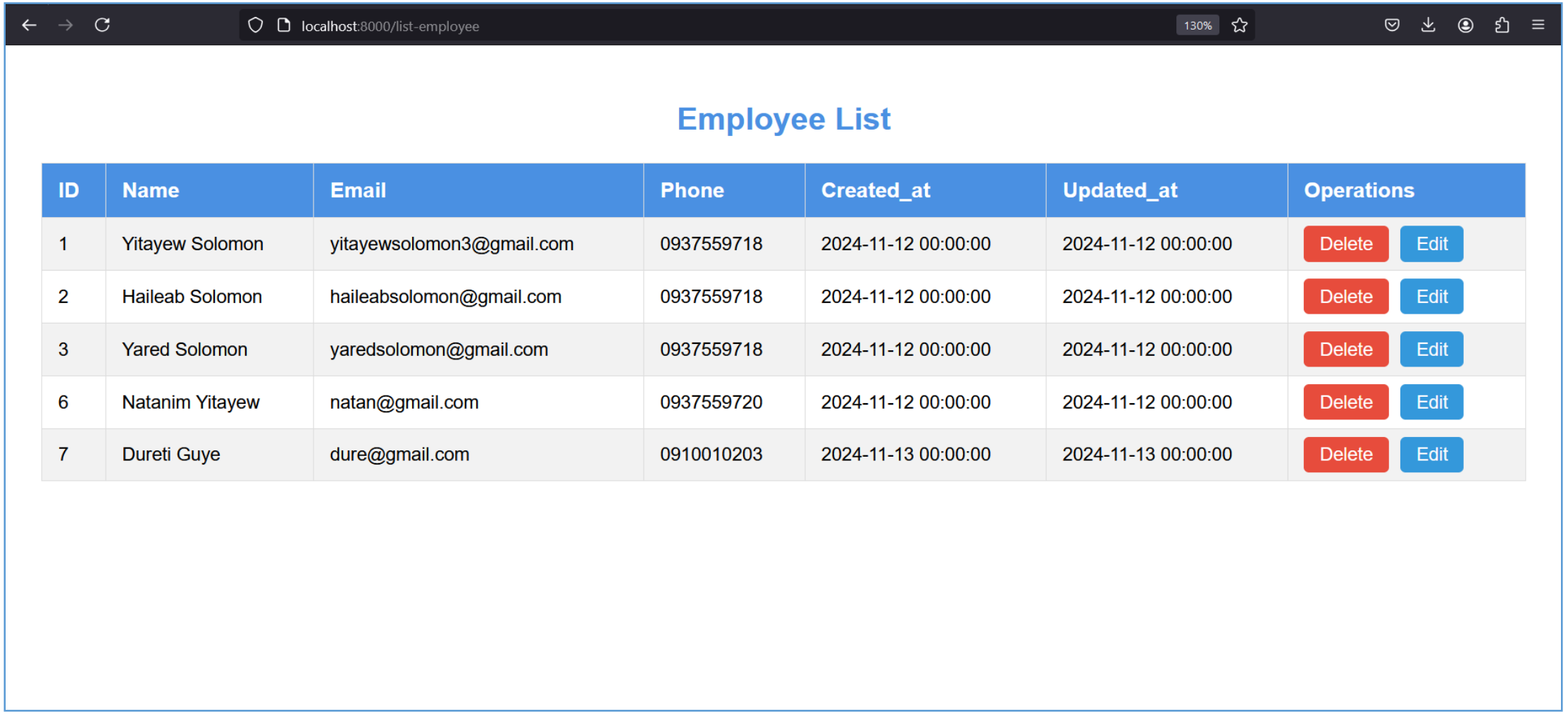
Output (http://localhost:8000/list-employee)



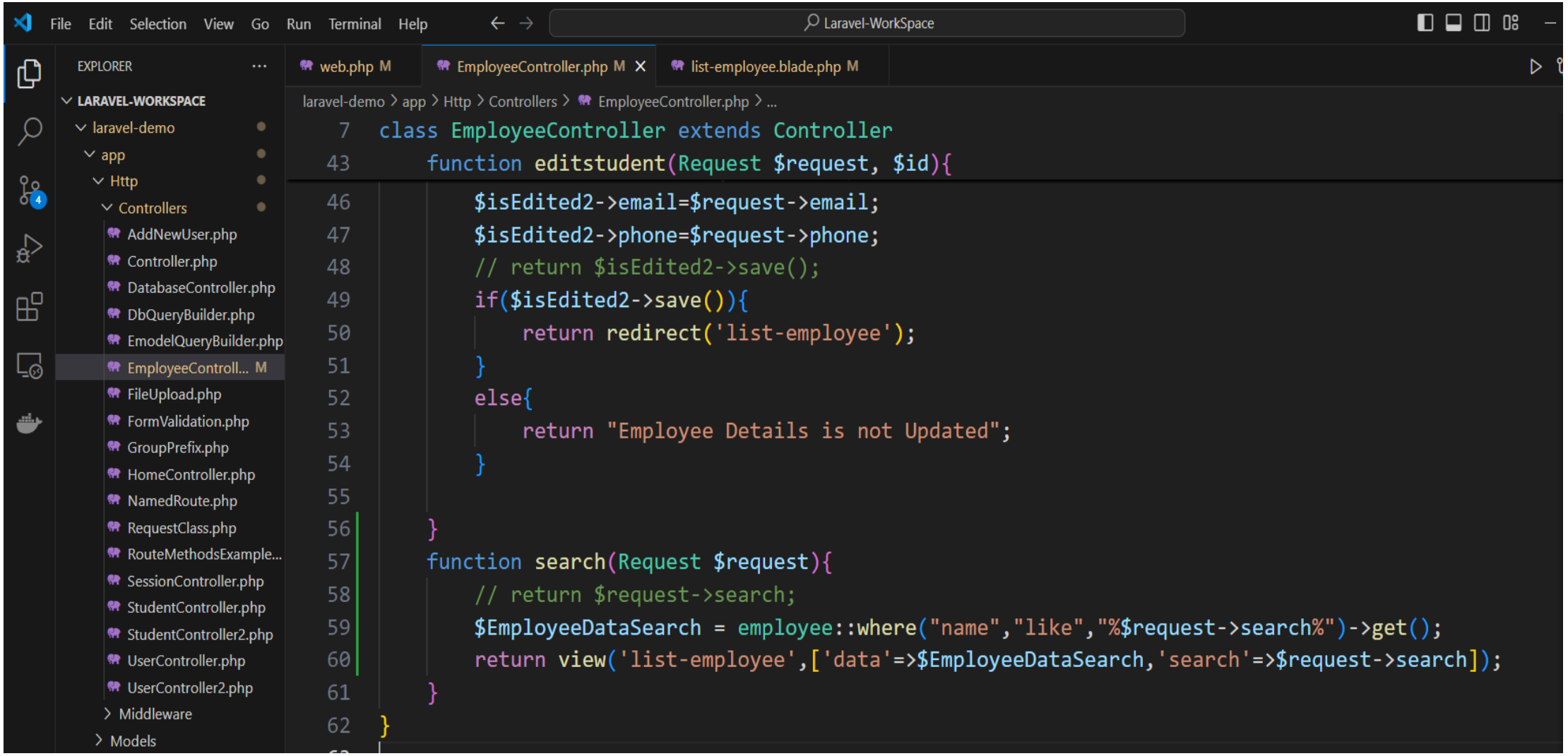
Output(http://localhost:8000/edit/7)



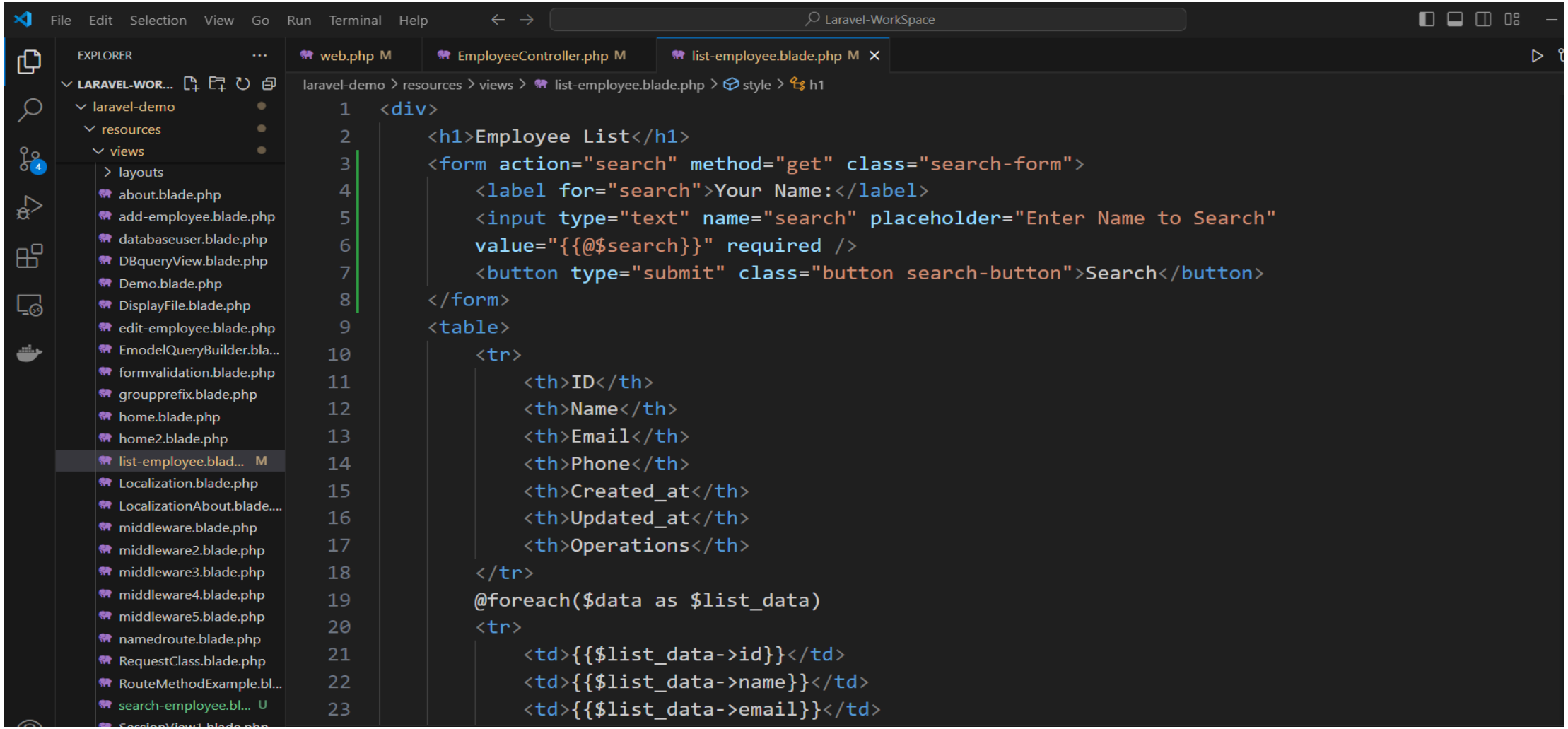
Output (http://localhost:8000/list-employee)



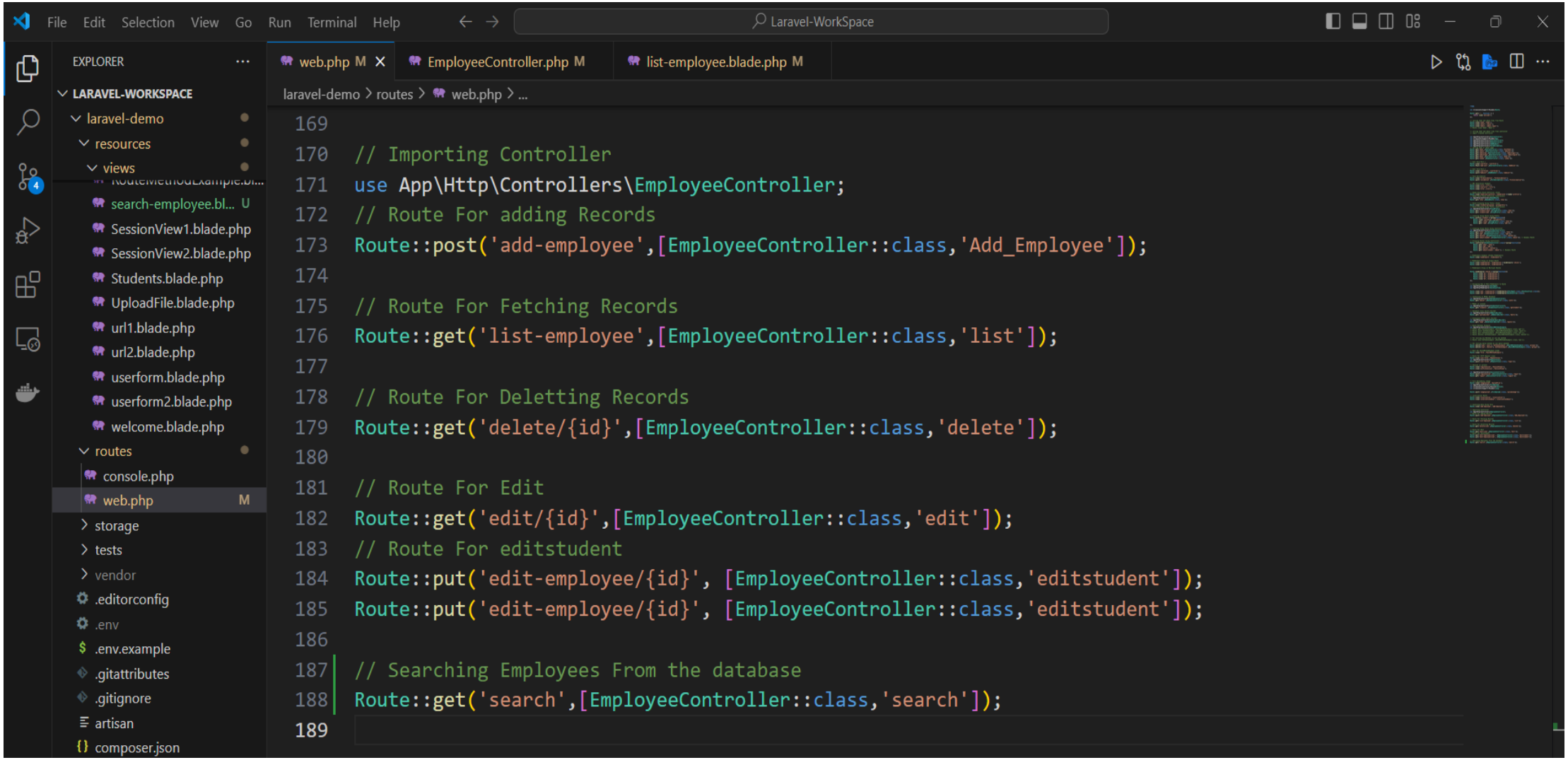
Search Data From MySql Database Table



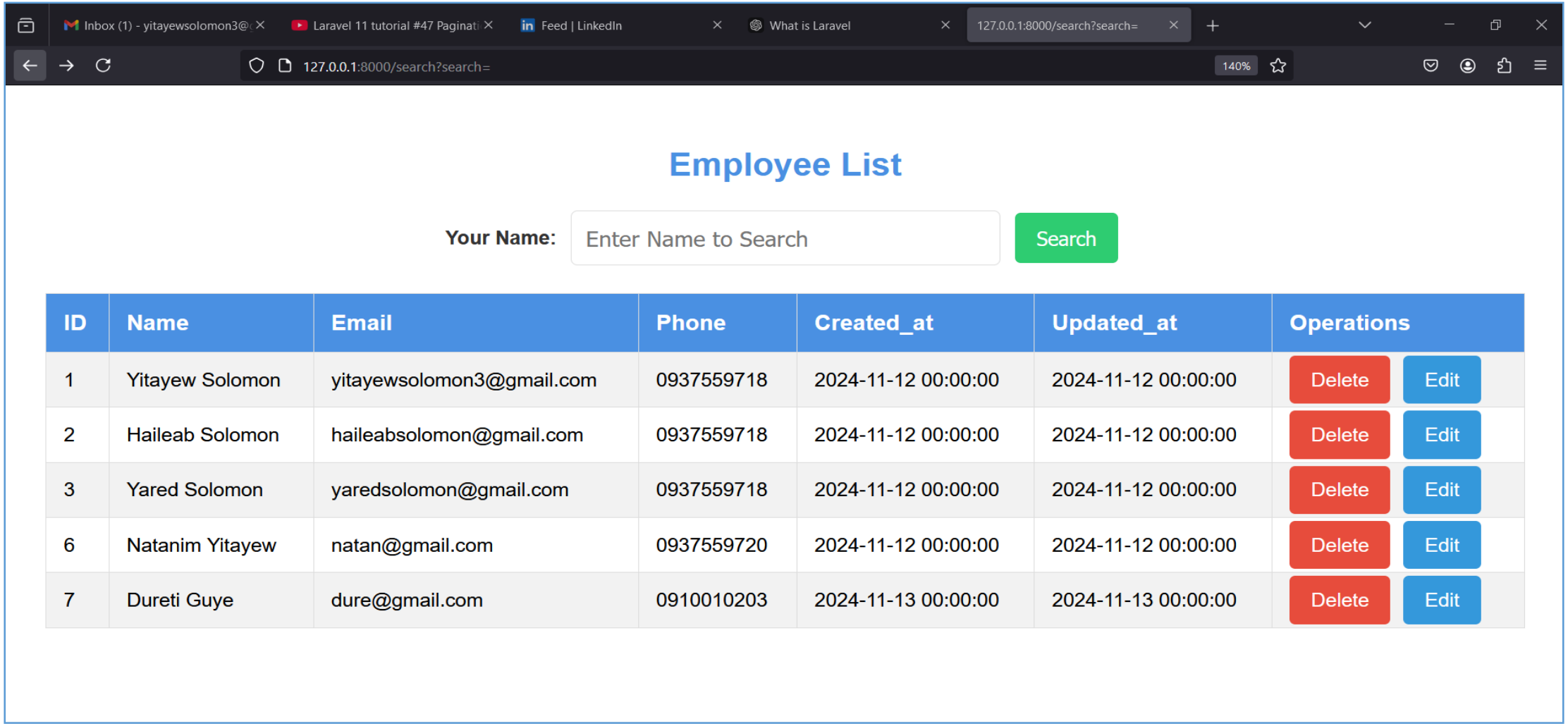
View (list-employee)



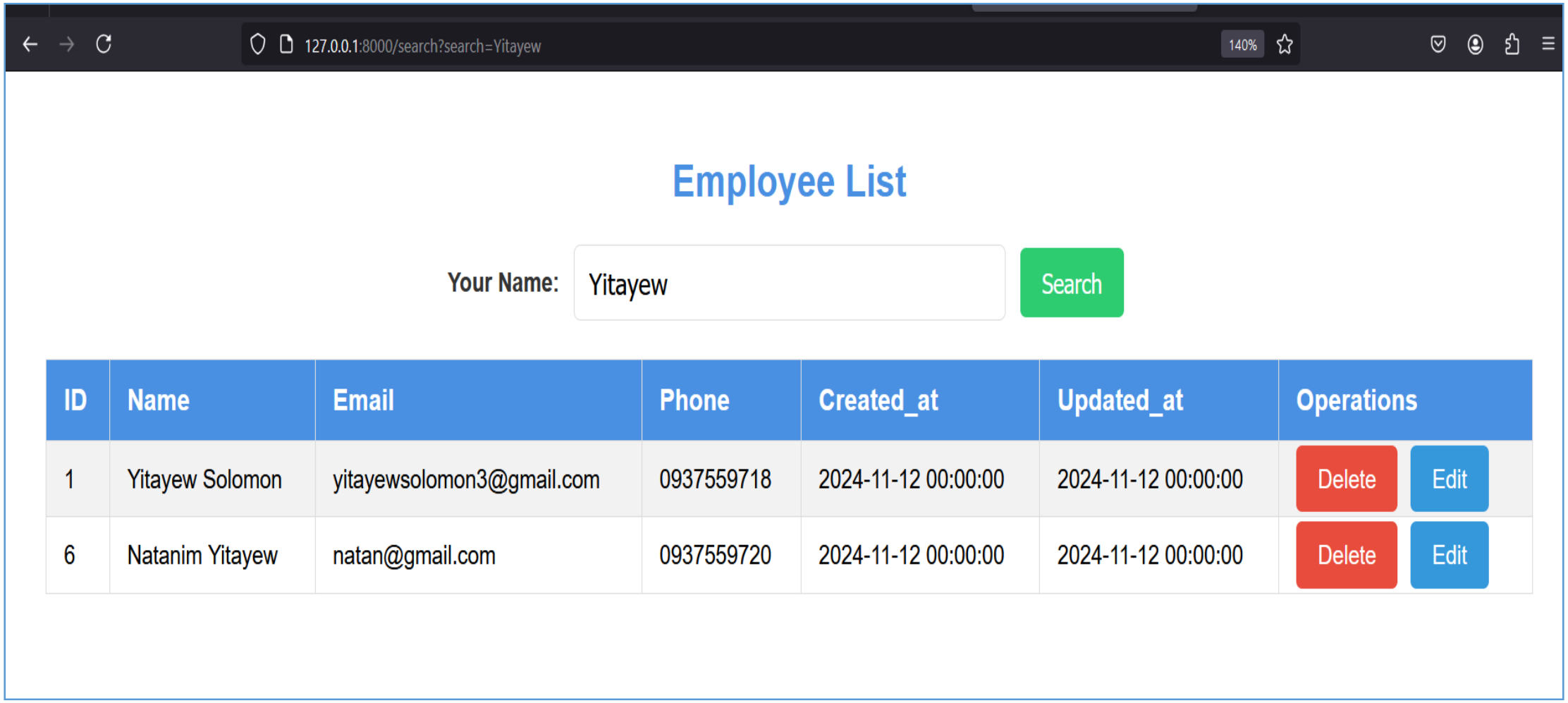
Route


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WORKSPACE
laravel-demo
resources
views
search-employee.bl... U
SessionView1.blade.php
SessionView2.blade.php
Students.blade.php
UploadFile.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
.env.example
.gitattributes
.gitignore
artisan
composer.json
web.php M
EmployeeController.php M
list-employee.blade.php M
laravel-demo > routes > web.php > ...
169
170 // Importing Controller
171 use App\Http\Controllers\EmployeeController;
172 // Route For adding Records
173 Route::post('add-employee',[EmployeeController::class,'Add_Employee']);
174
175 // Route For Fetching Records
176 Route::get('list-employee',[EmployeeController::class,'list']);
177
178 // Route For Deletting Records
179 Route::get('delete/{id}',[EmployeeController::class,'delete']);
180
181 // Route For Edit
182 Route::get('edit/{id}',[EmployeeController::class,'edit']);
183 // Route For editstudent
184 Route::put('edit-employee/{id}', [EmployeeController::class,'editstudent']);
185 Route::put('edit-employee/{id}', [EmployeeController::class,'editstudent']);
186
187 // Searching Employees From the database
188 Route::get('search',[EmployeeController::class,'search']);
189

Output (http://127.0.0.1:8000/search?search=)



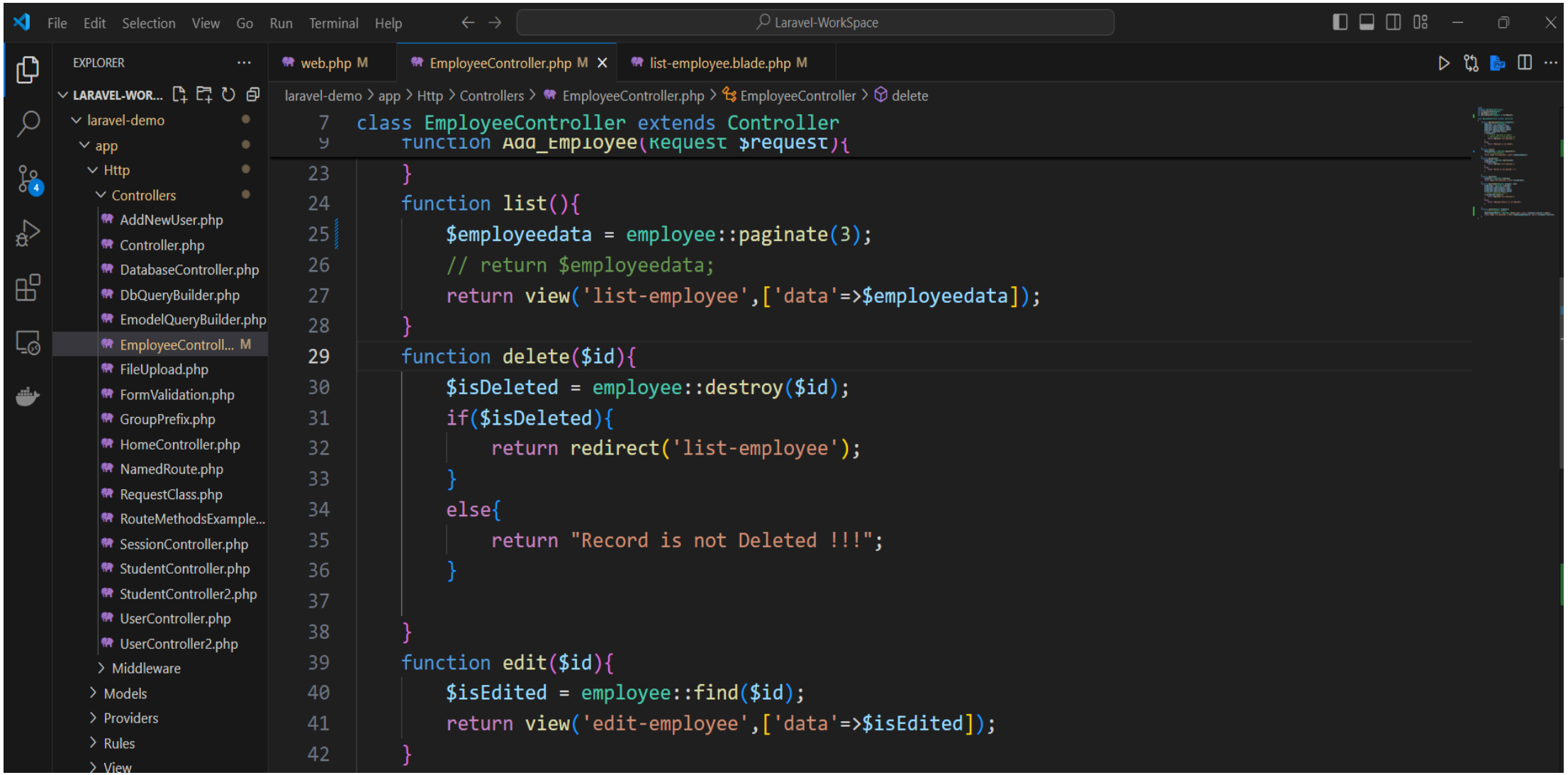
Output (http://127.0.0.1:8000/search?search=Yitayew)



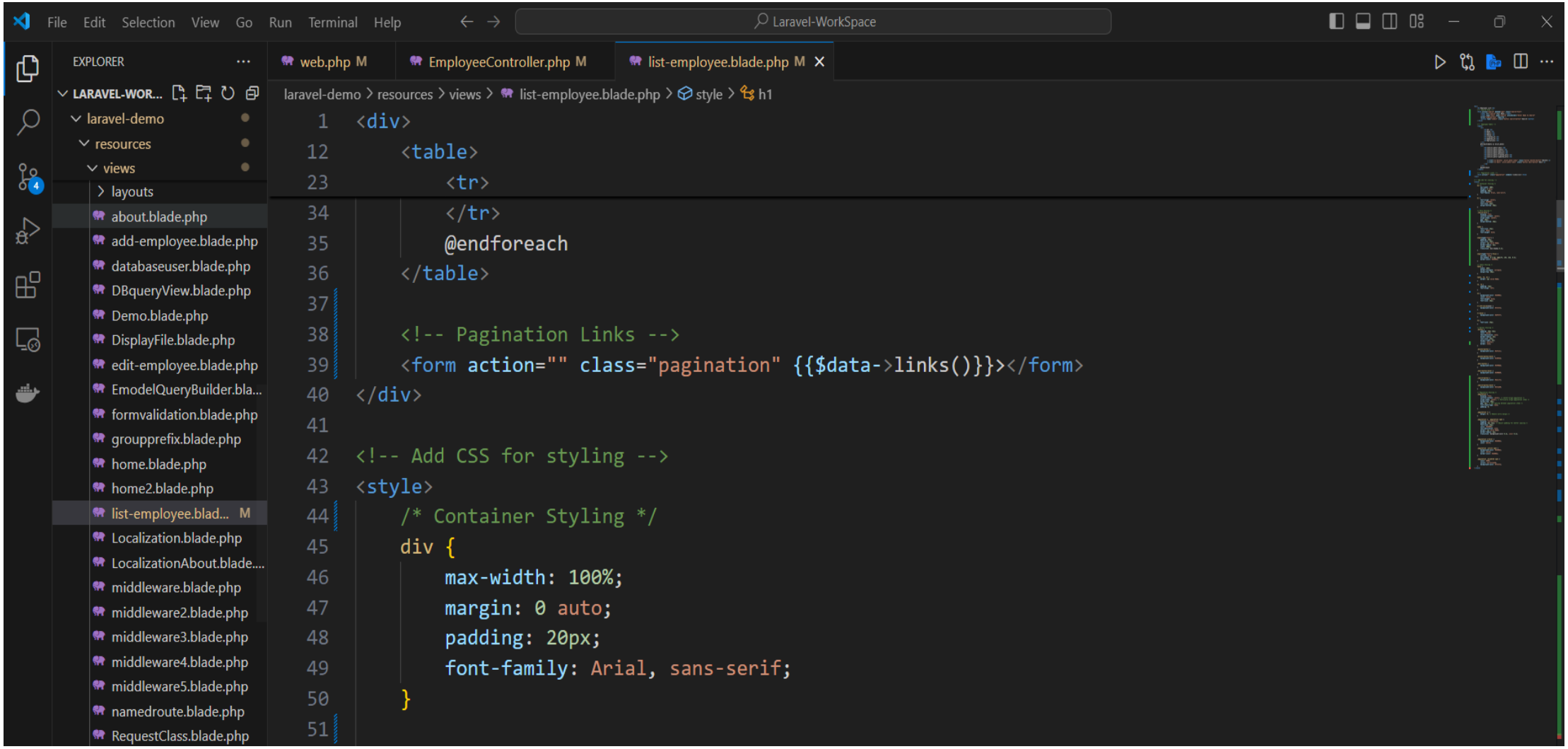
Pagination in Laravel

- Pagination in Laravel makes it easy to divide large datasets into manageable chunks, enhancing the performance and usability of your application. Laravel provides built-in methods for paginating database queries, making the process simple and efficient.

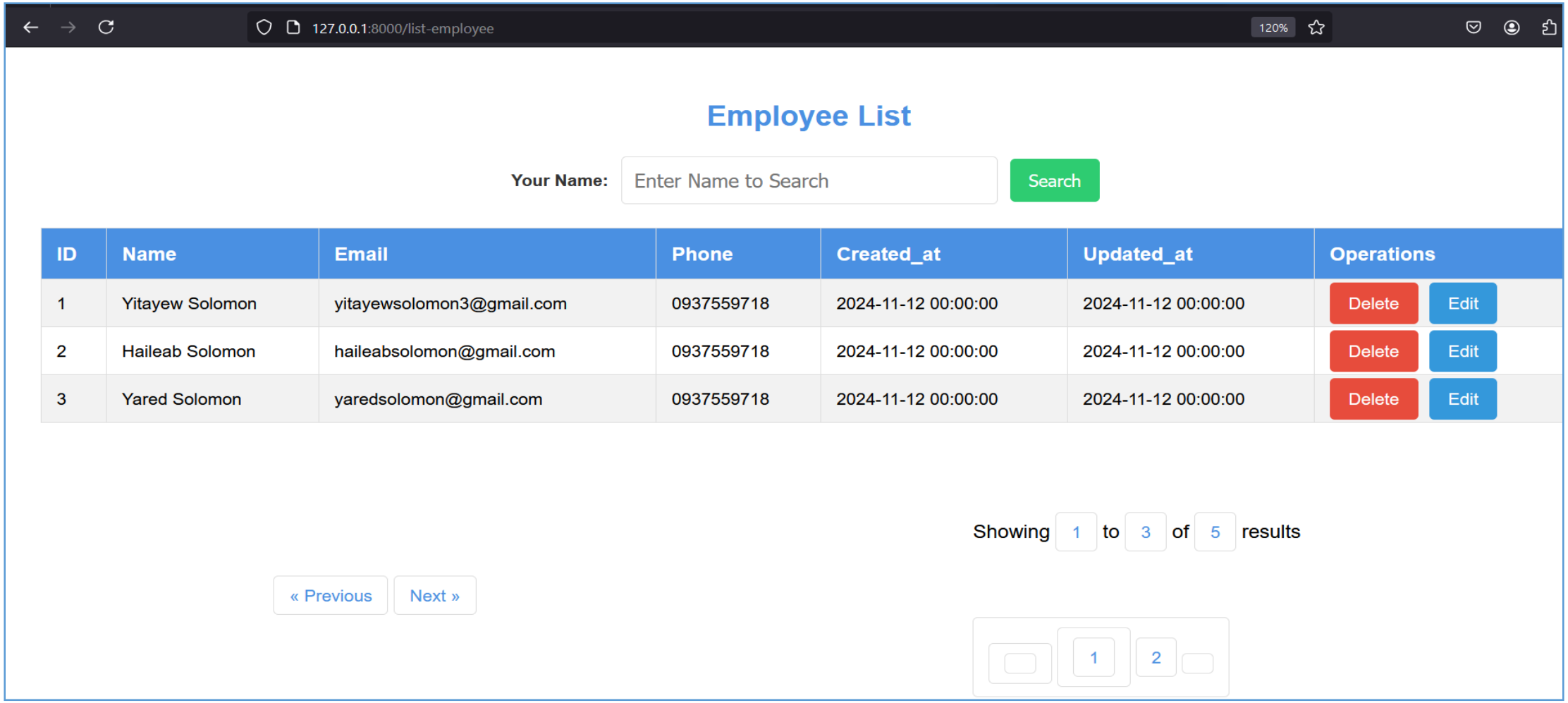
Example


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
web.php M
EmployeeController.php M
list-employee.blade.php M
laravel-demo > app > Http > Controllers > EmployeeController.php > EmployeeController > delete
LARAVEL-WOR...
laravel-demo
app
Http
Controllers
AddNewUser.php
Controller.php
DatabaseController.php
DbQueryBuilder.php
EmodelQueryBuilder.php
EmployeeControll... M
FileUpload.php
FormValidation.php
GroupPrefix.php
HomeController.php
NamedRoute.php
RequestClass.php
RouteMethodsExample...
SessionController.php
StudentController.php
StudentController2.php
UserController.php
UserController2.php
Middleware
Models
Providers
Rules
View
class EmployeeController extends Controller
function add_employee(Request $request){
}
function list(){
$employeedata = employee::paginate(3);
// return $employeedata;
return view('list-employee',['data'=>$employeedata]);
}
function delete($id){
$isDeleted = employee::destroy($id);
if($isDeleted){
return redirect('list-employee');
}
else{
return "Record is not Deleted !!!";
}
}
function edit($id){
$isEdited = employee::find($id);
return view('edit-employee',['data'=>$isEdited]);
}

Cont. ...


File Edit Selection View Go Run Terminal Help
Laravel-WorkSpace
EXPLORER
LARAVEL-WOR...
laravel-demo
resources
views
layouts
about.blade.php
add-employee.blade.php
databaseuser.blade.php
DBqueryView.blade.php
Demo.blade.php
DisplayFile.blade.php
edit-employee.blade.php
EmodelQueryBuilder.bla...
formvalidation.blade.php
grouppprefix.blade.php
home.blade.php
home2.blade.php
list-employee.blad... M
Localization.blade.php
LocalizationAbout.blade....
middleware.blade.php
middleware2.blade.php
middleware3.blade.php
middleware4.blade.php
middleware5.blade.php
namedroute.blade.php
RequestClass.blade.php
web.php M
EmployeeController.php M
list-employee.blade.php M
laravel-demo > resources > views > list-employee.blade.php > style > h1
1
12 <table>
23 <tr>
34 </tr>
35 @endforeach
36 </table>
37
38 <!-- Pagination Links -->
39 <form action="" class="pagination" {{$data->links()}}></form>
40
41
42 <!-- Add CSS for styling -->
43 <style>
44 /* Container Styling */
45 div {
46 max-width: 100%;
47 margin: 0 auto;
48 padding: 20px;
49 font-family: Arial, sans-serif;
50 }
51

Output


127.0.0.1:8000/list-employee
120%
Employee List
Your Name:
Enter Name to Search
Search
ID
Name
Email
Phone
Created_at
Updated_at
Operations
1
Yitayew Solomon
yitayewsolomon3@gmail.com
0937559718
2024-11-12 00:00:00
2024-11-12 00:00:00
Delete
Edit
2
Haileab Solomon
haileabsolomon@gmail.com
0937559718
2024-11-12 00:00:00
2024-11-12 00:00:00
Delete
Edit
3
Yared Solomon
yaredsolomon@gmail.com
0937559718
2024-11-12 00:00:00
2024-11-12 00:00:00
Delete
Edit
Showing 1 to 3 of 5 results
« Previous
Next »
1
2

Exercises

- **Perform CRUD Operation for Other Multi Medea Documents**

1. **Image**
2. **Audio**
3. **Video**
4. **Others**

Stub in Laravel

- A stub in Laravel is a pre-defined template or boilerplate code used to generate specific file types or components of the framework, such as controllers, models, or other classes.
- Stubs help speed up the development process by providing a base structure for these components, which can then be customized according to the application’s needs.

Customizing Stubs

Laravel allows you to customize stubs if you need specific changes to the generated files.

1. **Publish Stubs**: To customize stubs, you need to publish them first. Use the following command:

   bash Copy code

   ```
   php artisan stub:publish
   ```

   This will copy the default stubs to the `stubs` directory at the root of your project.

2. **Edit the Stubs**: Navigate to the `stubs` directory in your project and edit the required stub file.

   For example:

   - `controller.stub`: For controllers
   - `model.stub`: For models
   - `migration.stub`: For migrations

Migration In Laravel

- In Laravel, migration is a way to manage and version-control your database schema. Migrations allow developers to define and modify database structures in a programmatic and version-controlled manner, making it easy to synchronize the database schema across different environments.

Benefits of Migrations

- **Version Control**: Track changes to the database schema over time.
- **Team Collaboration**: Makes it easy for teams to work on the same database structure.
- **Rollback Support**: Easily undo changes to the database.
- **Automated Deployment**: Define schema changes in code and execute them during deployment.

Creating a Migration

Creating a Migration

To create a migration file, use the Artisan command:

bash Copy code

```
php artisan make:migration create_table_name_table
```

This generates a migration file in the `database/migrations` directory with a timestamp in the filename, ensuring order.

For example:

bash Copy code

```
php artisan make:migration create_employees_table
```

Cont. ...

Copy code

```php
class CreateEmployeesTable extends Migration
{
    public function up()
    {
        Schema::create('employees', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->string('phone');
            $table->timestamps(); // Adds `created_at` and `updated_at` columns
        });
    }

    public function down()
    {
        Schema::dropIfExists('employees');
    }
}
```

Cont. ...

Running Migrations

To execute all pending migrations and apply changes to the database:

bash Copy code

```
php artisan migrate
```

This command reads the migration files and updates the database schema.

Rolling Back Migrations

To undo the last batch of migrations:

bash Copy code

```
php artisan migrate:rollback
```

Cont. ...

To reset all migrations (undo all changes):

bash Copy code

```
php artisan migrate:reset
```

To rollback and reapply all migrations in a single command:

bash Copy code

```
php artisan migrate:refresh
```

Migration Commands Summary

Migration Commands Summary

Command	Description
`php artisan make:migration`	Create a new migration file.
`php artisan migrate`	Run all pending migrations.
`php artisan migrate:rollback`	Rollback the last batch of migrations.
`php artisan migrate:reset`	Rollback all migrations.
`php artisan migrate:refresh`	Rollback and reapply all migrations.
`php artisan migrate:fresh`	Drop all tables and rerun all migrations.

Seeding in laravel

- In Laravel, seeding is the process of **populating your database** with sample or default data. Seeders are classes that allow you to insert records into your database tables.
- This is particularly useful when you need to populate the database with data during **development** or **testing**, or if you need to pre-populate certain fields in production.

Benefits of Seeding

- **Quick Data Population**: Allows you to quickly insert sample data into your database for testing and development.
- **Reusable:** Seeders can be re-run without affecting your production data (with proper configuration).
- **Testing:** Ensures that your application works with real-like data during the testing phase.

Cont. ...

Creating a Seeder

To create a new seeder, use the Artisan command:

bash Copy code

```
php artisan make:seeder SeederName
```

For example, to create a seeder for employees:

bash Copy code

```
php artisan make:seeder EmployeesTableSeeder
```

Cont. ...

This creates a seeder file inside the `database/seeders` directory:

php Copy code

```php
<?php

namespace Database\Seeders;

use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;

class EmployeesTableSeeder extends Seeder
{
    public function run()
    {
        DB::table('employees')->insert([
            'name' => 'John Doe',
            'email' => 'john.doe@example.com',
            'phone' => '1234567890',
        ]);
    }
}
```

Cont. ...

Summary of Seeder Commands

Command	Description
`php artisan make:seeder SeederName`	Create a new seeder class.
`php artisan db:seed`	Run all seeders defined in `DatabaseSeeder`.
`php artisan db:seed --class=SeederName`	Run a specific seeder.
`php artisan migrate --seed`	Run migrations and seed the database.
`php artisan make:factory FactoryName --model=ModelName`	Create a new factory.

Thank you!

Appreciate your action.