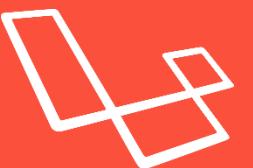


# Laravel Auth

# Laravel Notes

# Auth



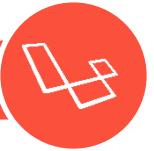
# What is Middleware ?

- **Middleware** acts as a bridge between a request and a response.
- **Middleware** provide a convenient mechanism for filtering HTTP requests entering your **application**

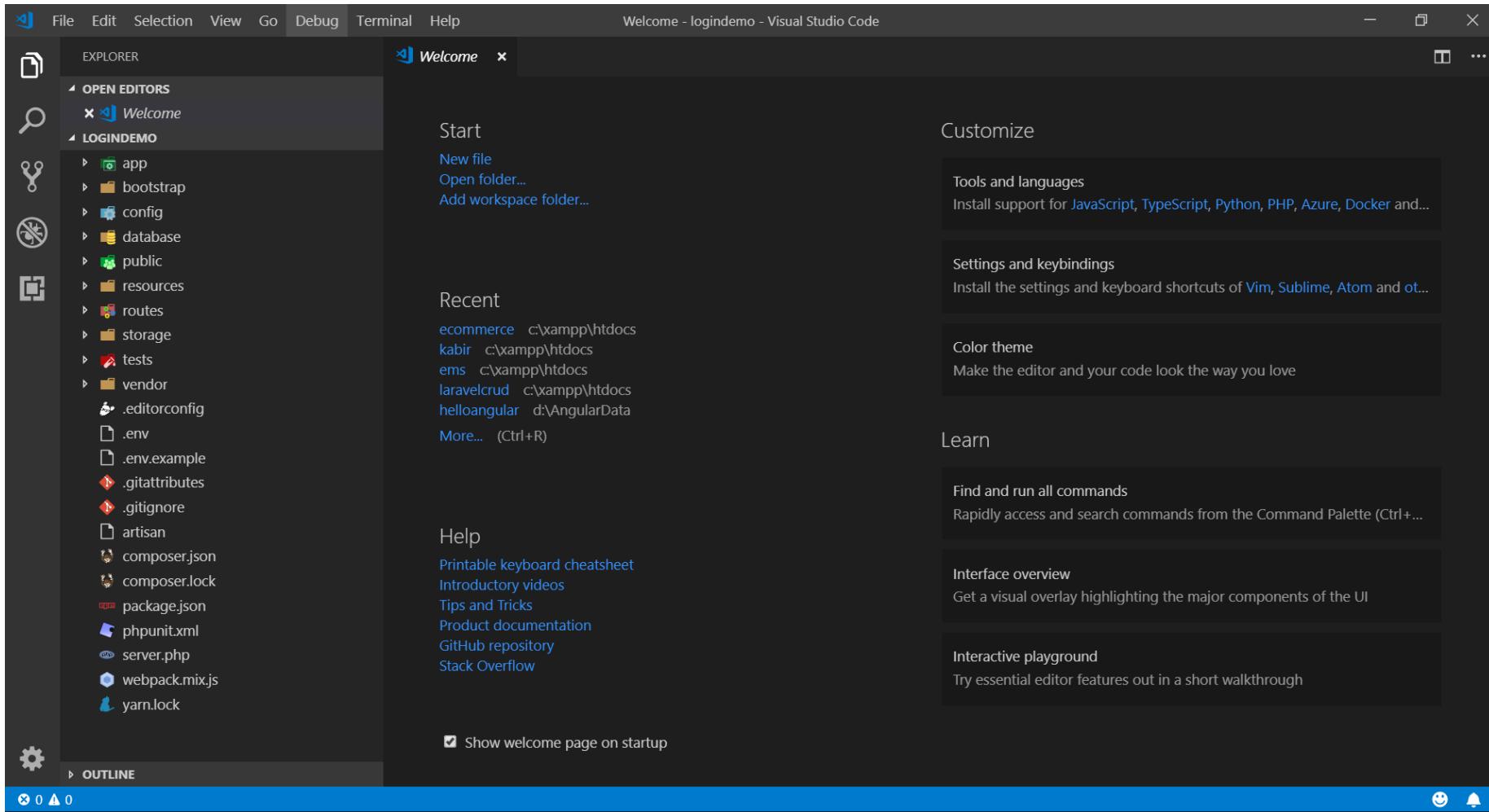


# Steps :

- Add New Field in DB
- Add new Field in View
- Modify Model
- Add Field in Validation and Form Process
- Create Middleware
- Use Middle Ware in Kernel
- Config Middleware
- Route Check
- Login Controller Check

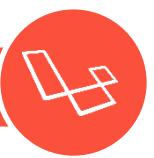


# Create New Project

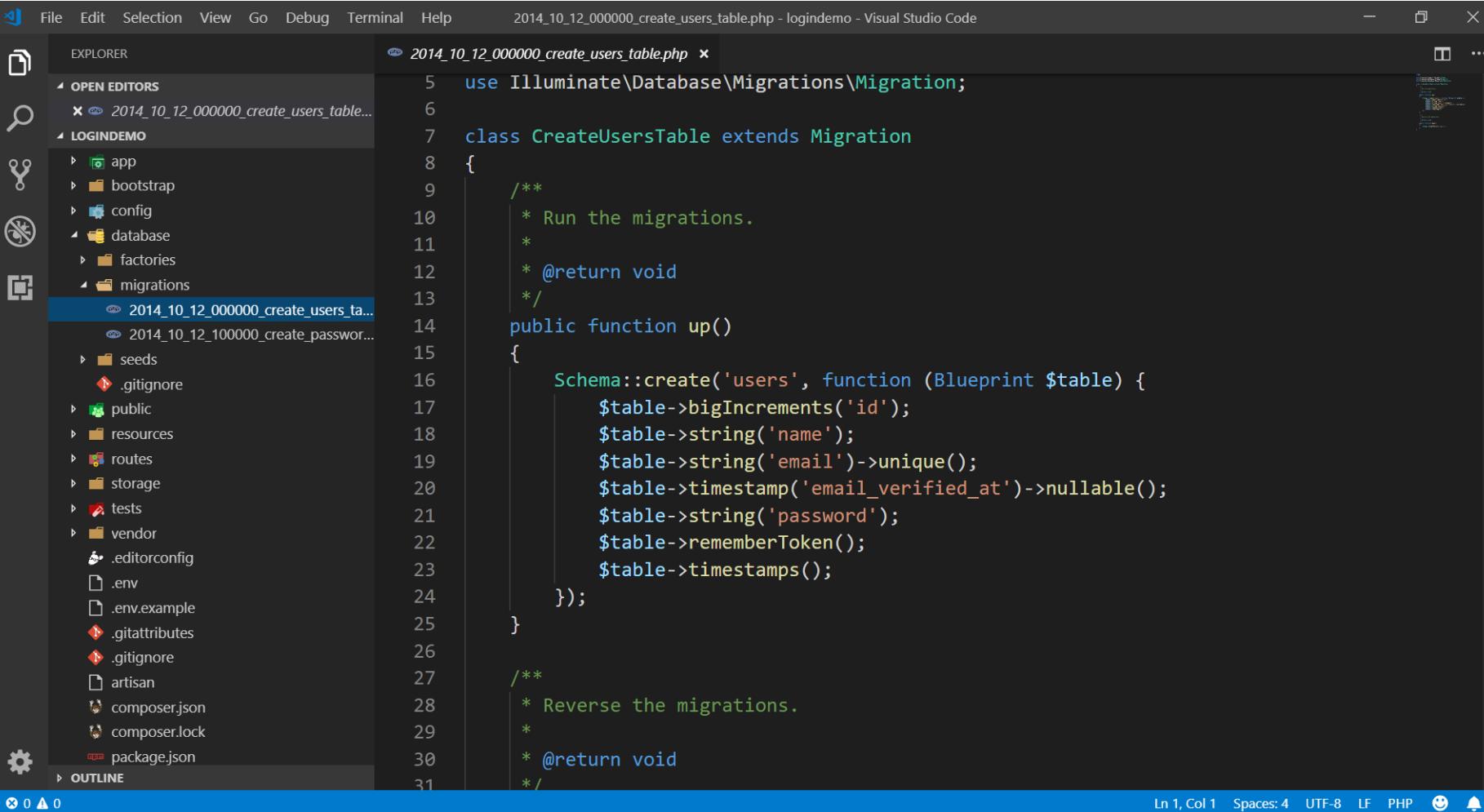


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Open User Migration File



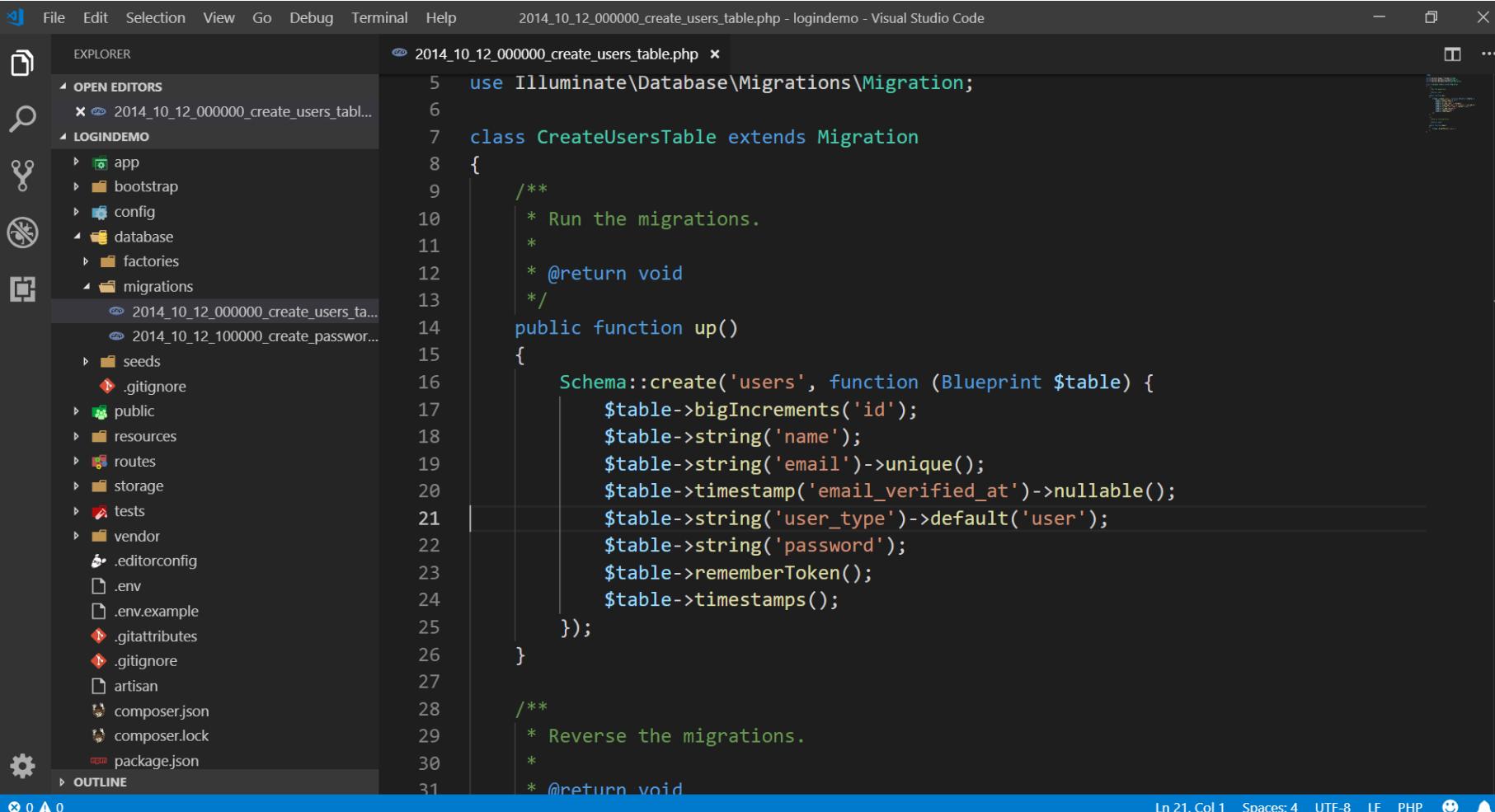
The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "LOGINDEMO". The "migrations" folder contains several files, with "2014\_10\_12\_000000\_create\_users\_table.php" selected.
- Editor:** The main editor pane displays the PHP code for the migration file "2014\_10\_12\_000000\_create\_users\_table.php".
- Code Content:**

```
5 use Illuminate\Database\Migrations\Migration;
6
7 class CreateUsersTable extends Migration
8 {
9     /**
10      * Run the migrations.
11      *
12      * @return void
13     */
14    public function up()
15    {
16        Schema::create('users', function (Blueprint $table) {
17            $table->bigIncrements('id');
18            $table->string('name');
19            $table->string('email')->unique();
20            $table->timestamp('email_verified_at')->nullable();
21            $table->string('password');
22            $table->rememberToken();
23        });
24    }
25
26    /**
27     * Reverse the migrations.
28     *
29     * @return void
30     */
31 }
```
- Status Bar:** Shows "Ln 1, Col 1" and other settings like "Spaces: 4", "UTF-8", "LF", "PHP", and icons for "git", "status", and "activity".



# Add User Type Field

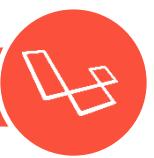


The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar contains an Explorer view with project files: app, bootstrap, config, database, factories, migrations (containing 2014\_10\_12\_000000\_create\_users\_table.php and 2014\_10\_12\_100000\_create\_passwords\_table.php), seeds, .gitignore, public, resources, routes, storage, tests, vendor, .editorconfig, .env, .env.example, .gitattributes, .gitignore, artisan, composer.json, composer.lock, and package.json. The bottom bar shows status icons for file changes, outline, and terminal.

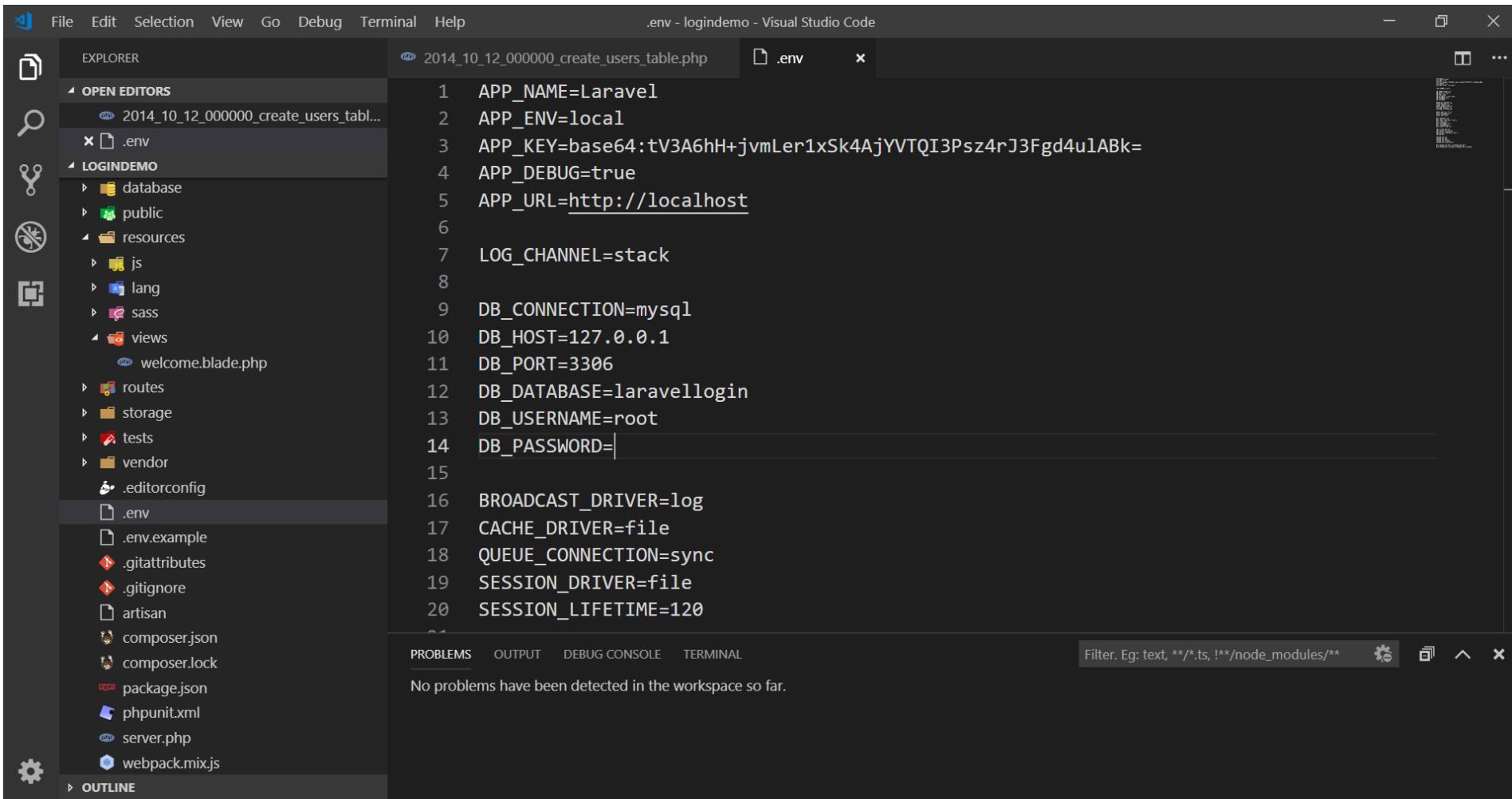
```
use Illuminate\Database\Migrations\Migration;
class CreateUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('user_type')->default('user');
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
}
```

Ln 21, Col 1 Spaces: 4 UTF-8 LF PHP ☺ 🔔



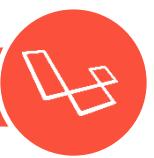
# Set DB Connection



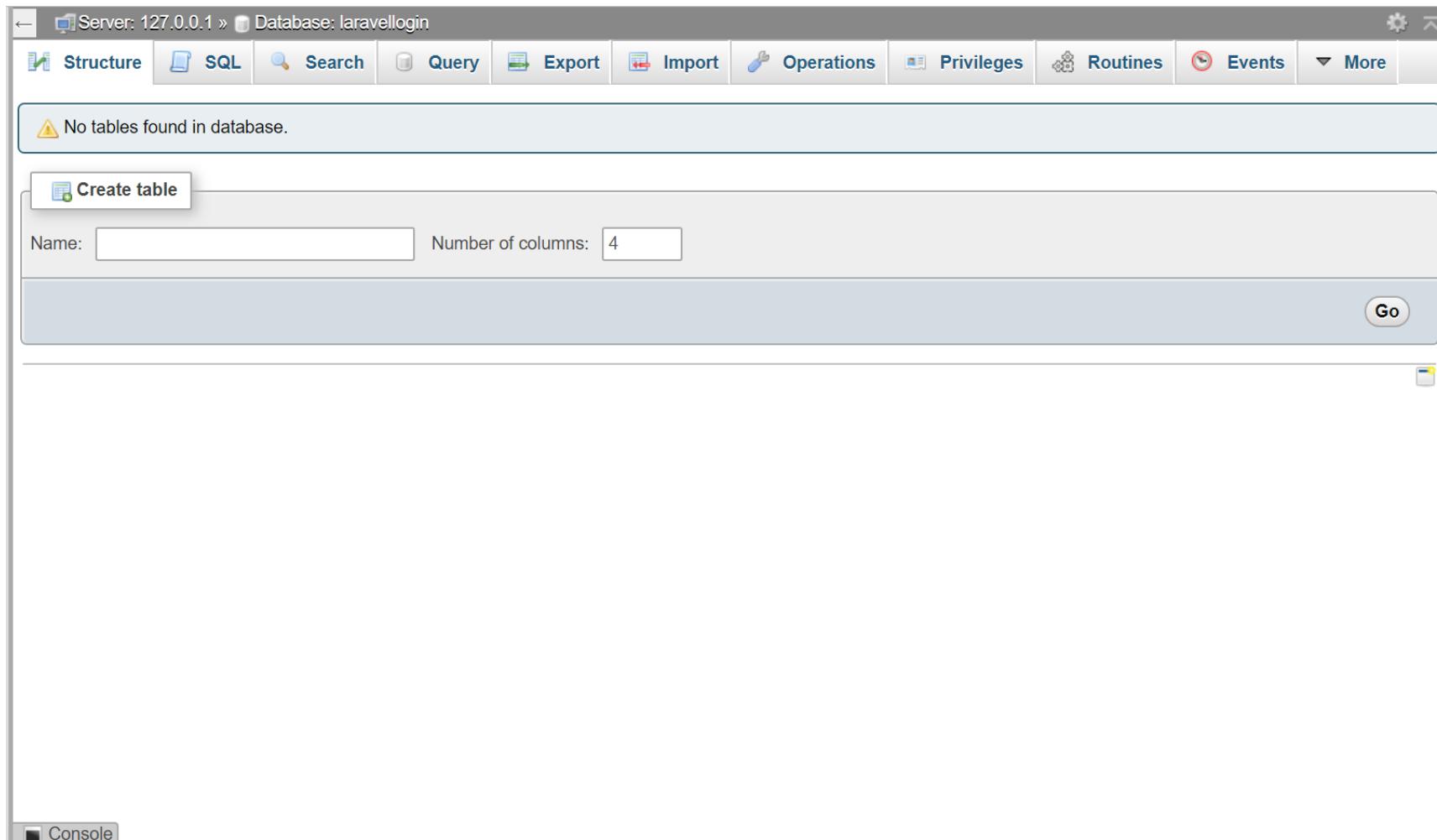
The screenshot shows the Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Editor Title:** .env - logindemo - Visual Studio Code
- Explorer:** Shows the project structure for "LOGINDEMO".
- Editor Content:** The .env file contains the following environment variables:

```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:tV3A6hH+jvmLer1xSk4AjYVTQI3Psz4rJ3Fgd4ulABk=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=laravellogin
13 DB_USERNAME=root
14 DB_PASSWORD=
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
```
- Bottom Bar:** PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL. A message states: "No problems have been detected in the workspace so far."
- Bottom Status Bar:** Filter: Eg: text, \*\*/\*.ts, !\*\*/node\_modules/\*\*

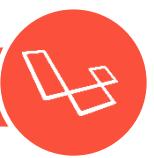


# Create DB in PhpMyAdmin



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Make Auth

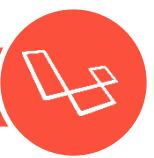
C:\Windows\system32\cmd.exe

```
c:\xampp\htdocs\logindemo>php artisan make:auth
```



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



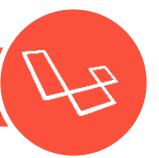
# Migrate DB

```
c:\Windows\system32\cmd.exe
c:\xampp\htdocs\logindemo>php artisan migrate
```



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Run Migrate Command

C:\Windows\system32\cmd.exe

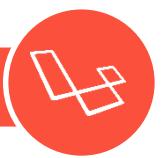
```
c:\xampp\htdocs\logindemo>php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table
```

```
c:\xampp\htdocs\logindemo>
```

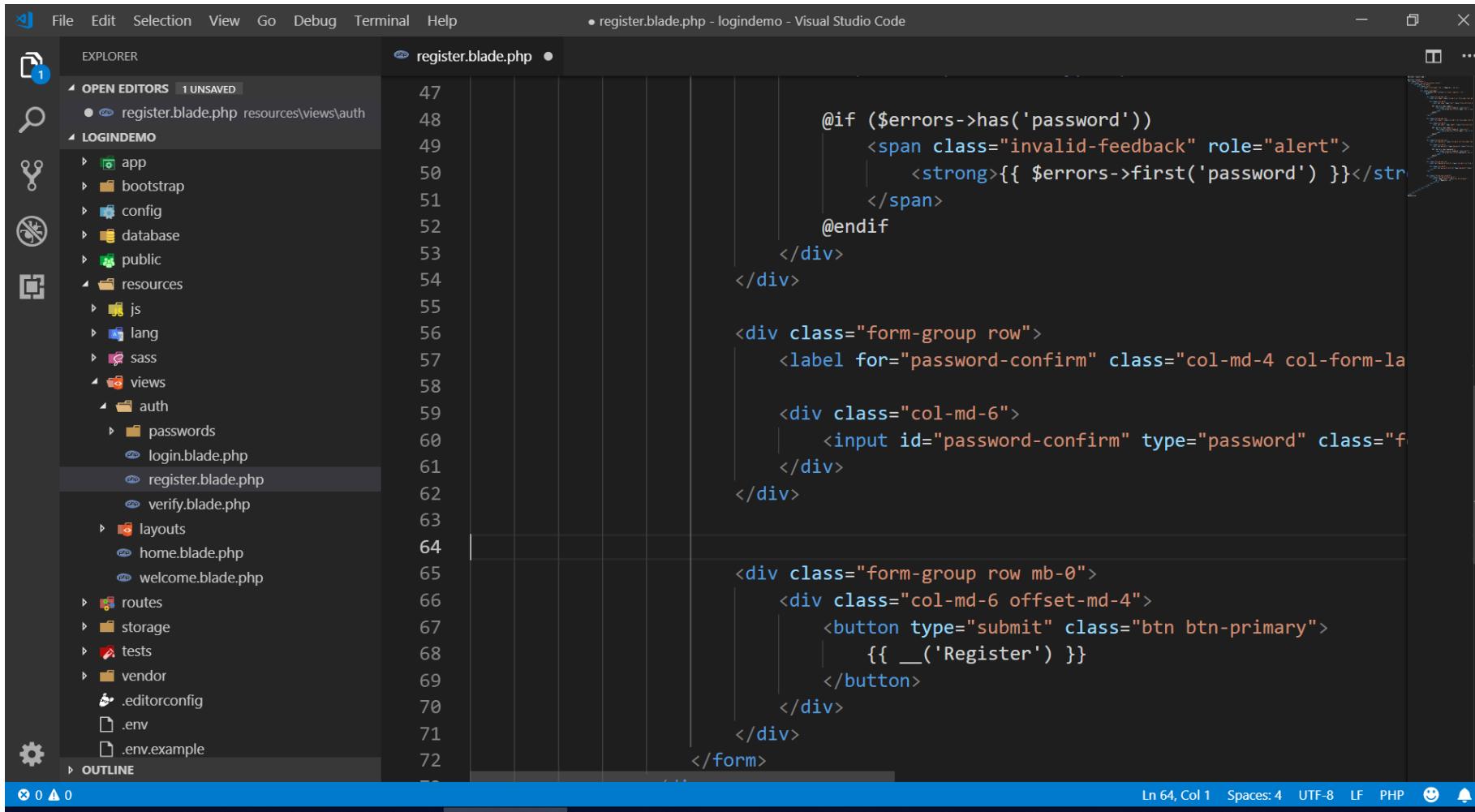


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Open Resource/Views/Auth/Register.blade.php



The screenshot shows the Visual Studio Code interface with the following details:

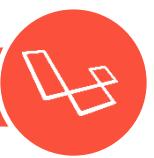
- File Path:** register.blade.php - logindemo - Visual Studio Code
- Explorer View:** Shows the project structure under "LOGINDEMO". The "auth" folder is expanded, showing "passwords", "login.blade.php", "register.blade.php", and "verify.blade.php".
- Code Editor:** Displays the content of "register.blade.php". The code includes Blade templating syntax like @if, @endif, and @foreach.
- Status Bar:** Shows "Ln 64, Col 1" and other settings like "Spaces: 4", "UTF-8", "LF", "PHP".

```
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
```

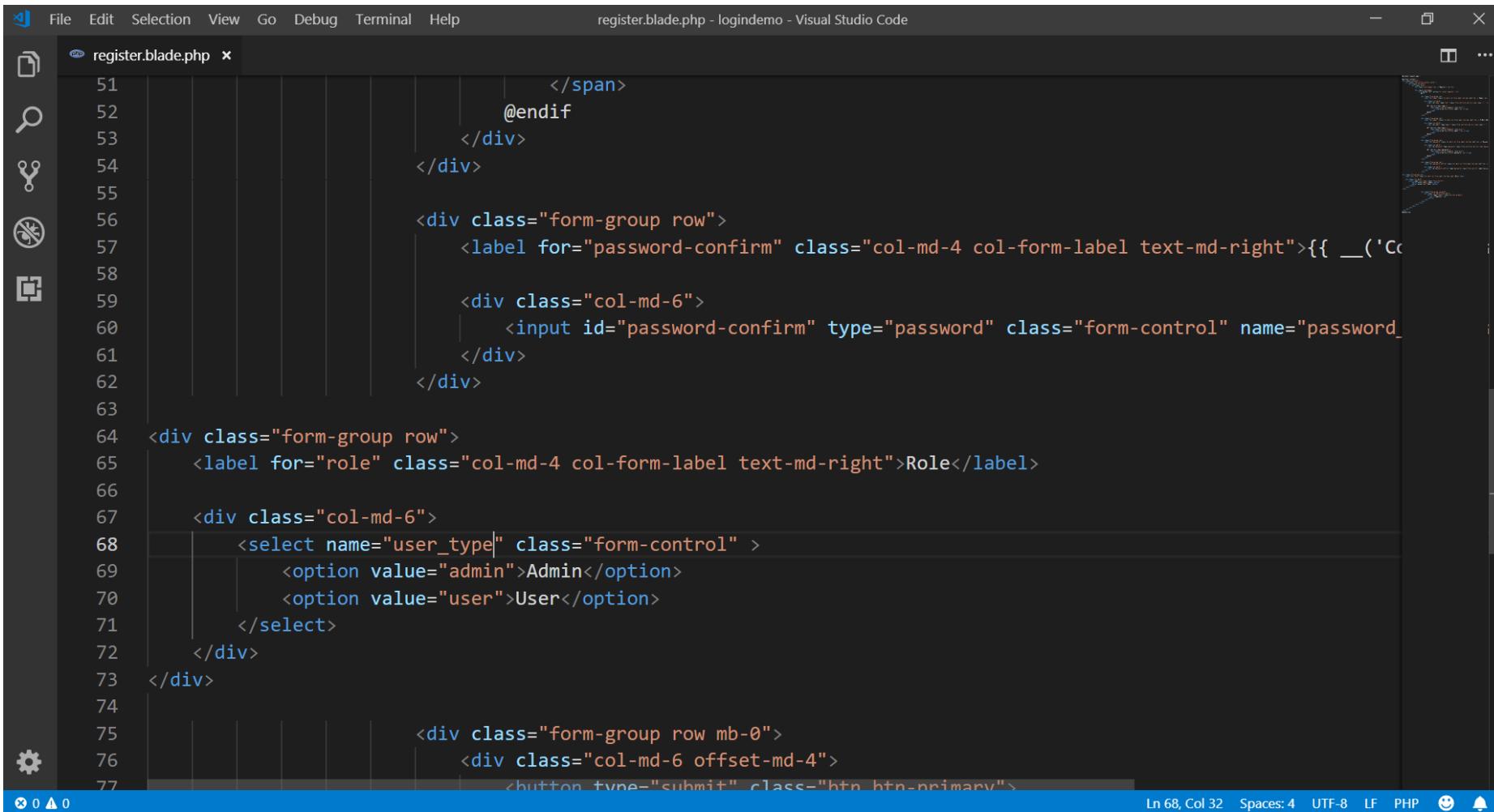
```
@if ($errors->has('password'))
    <span class="invalid-feedback" role="alert">
        <strong>{{ $errors->first('password') }}</strong>
    </span>
@endif
</div>
</div>

<div class="form-group row">
    <label for="password-confirm" class="col-md-4 col-form-label">
        Confirm Password
    </label>
    <div class="col-md-6">
        <input id="password-confirm" type="password" class="form-control" name="password_confirmation" required>
    </div>
</div>

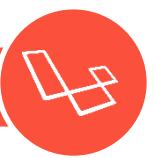
<div class="form-group row mb-0">
    <div class="col-md-6 offset-md-4">
        <button type="submit" class="btn btn-primary">
            {{ __('Register') }}
        </button>
    </div>
</div>
```



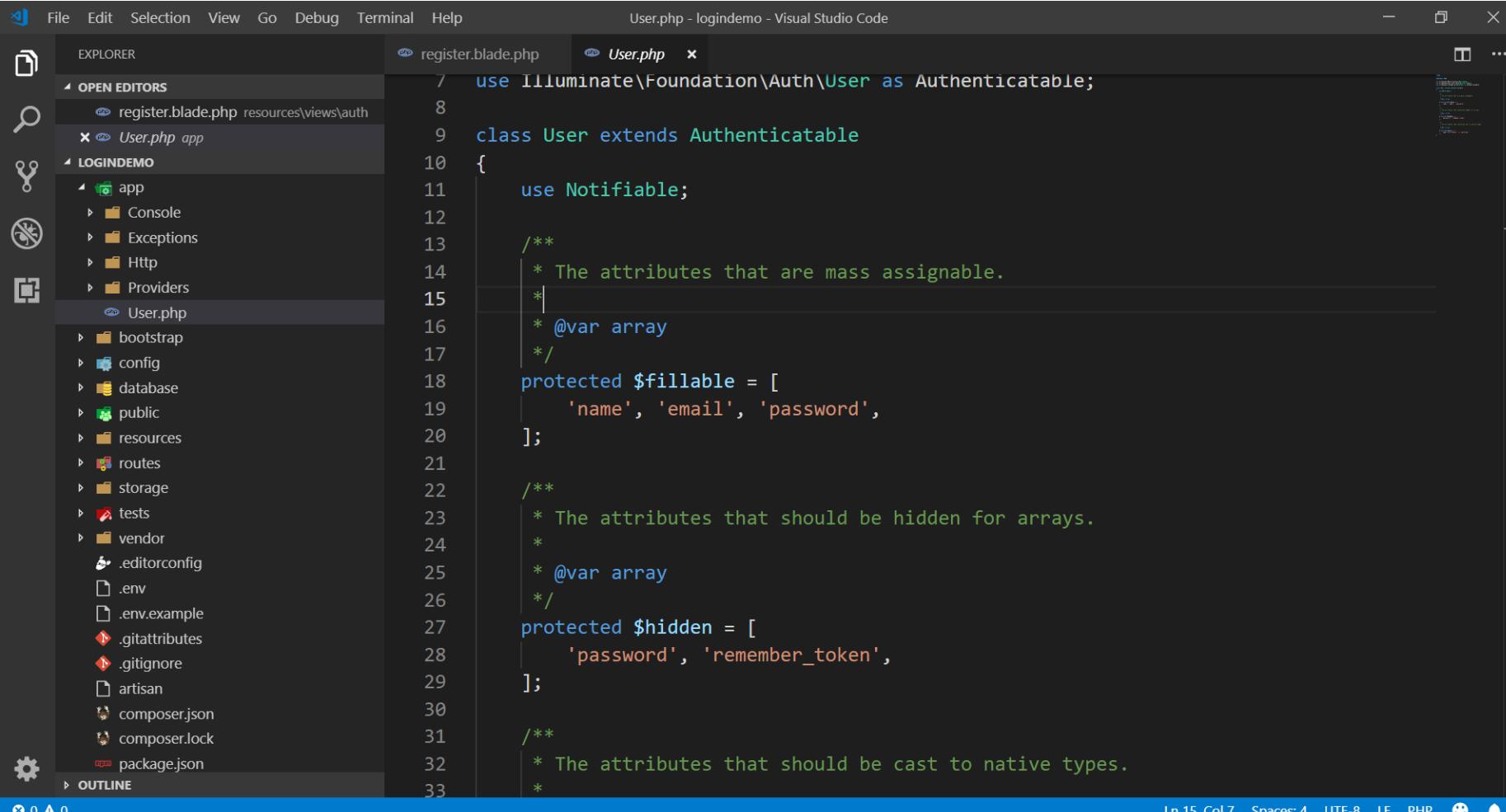
# Add Dropdown Code in Register.blade.php



```
File Edit Selection View Go Debug Terminal Help
register.blade.php - logindemo - Visual Studio Code
register.blade.php x
51 |           </span>
52 |       @endif
53 |     </div>
54 |   </div>
55 |
56   <div class="form-group row">
57     <label for="password-confirm" class="col-md-4 col-form-label text-md-right">{{ __('Co
58
59     <div class="col-md-6">
60       <input id="password-confirm" type="password" class="form-control" name="password_
61     </div>
62   </div>
63
64 <div class="form-group row">
65   <label for="role" class="col-md-4 col-form-label text-md-right">Role</label>
66
67   <div class="col-md-6">
68     <select name="user_type" class="form-control" >
69       <option value="admin">Admin</option>
70       <option value="user">User</option>
71     </select>
72   </div>
73 </div>
74
75   <div class="form-group row mb-0">
76     <div class="col-md-6 offset-md-4">
77       <button type="submit" class="btn btn-primary">
```



# Open User Model



The screenshot shows the Visual Studio Code interface with the following details:

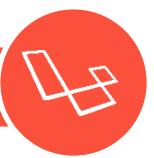
- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Editor Bar:** User.php - logindemo - Visual Studio Code.
- Explorer:** Shows the project structure for "logindemo".
  - OPEN EDITORS:** register.blade.php, User.php.
  - LOGINDEMO:** app (Console, Exceptions, Http, Providers, User.php), bootstrap, config, database, public, resources, routes, storage, tests, vendor, .editorconfig, .env, .env.example, .gitattributes, .gitignore, artisan, composer.json, composer.lock, package.json.
- User.php Content:**

```
use Illuminate\Foundation\Auth\User as Authenticatable;
class User extends Authenticatable
{
    use Notifiable;

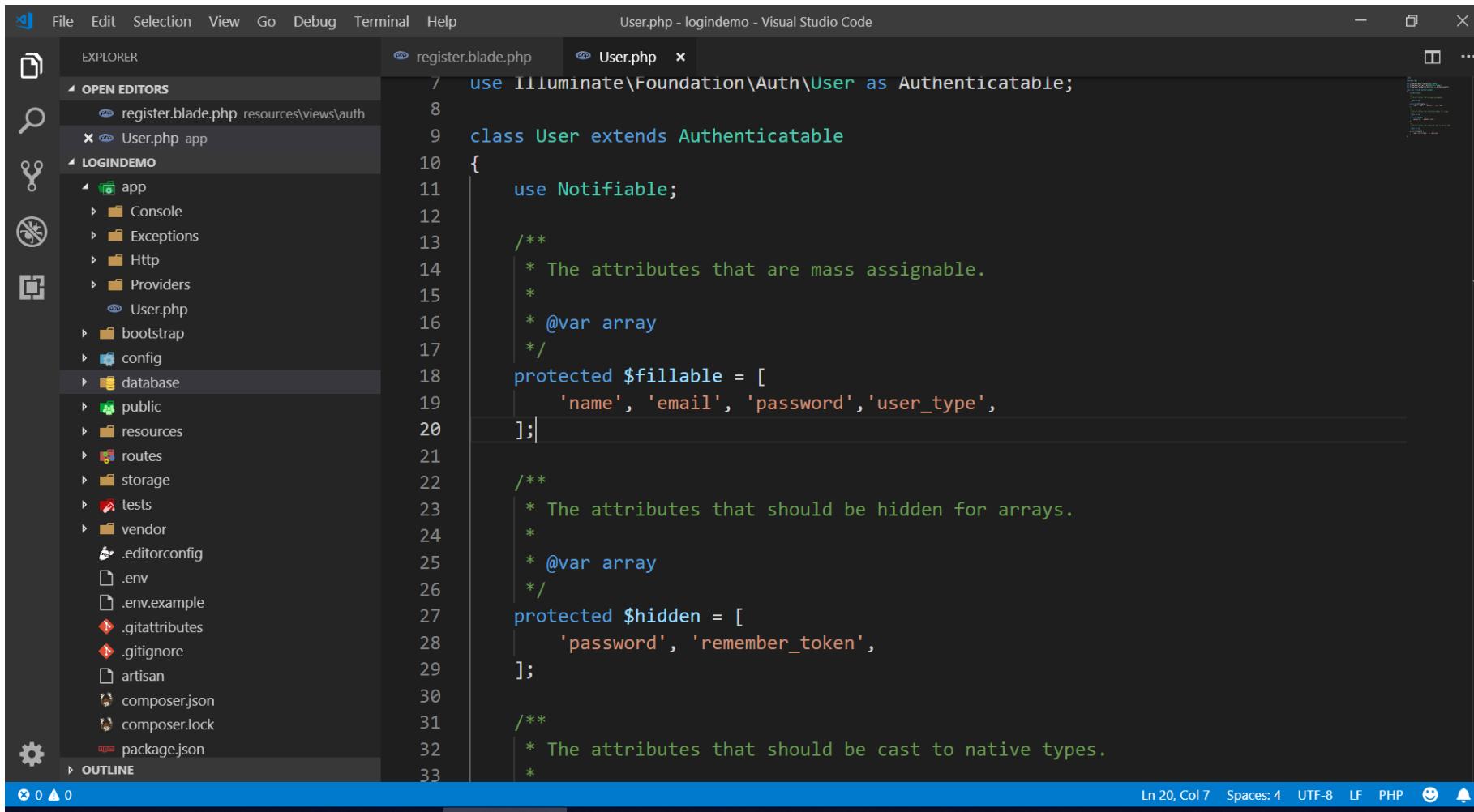
    /**
     * The attributes that are mass assignable.
     */
    protected $fillable = [
        'name', 'email', 'password',
    ];

    /**
     * The attributes that should be hidden for arrays.
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    /**
     * The attributes that should be cast to native types.
     */
}
```
- Status Bar:** Ln 15, Col 7, Spaces: 4, UTF-8, LF, PHP, 😊, 📲.



# Add user\_type filed.



The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Debug, Terminal, Help
- Title Bar:** User.php - logindemo - Visual Studio Code
- Explorer:** Shows the project structure of "logindemo". The "database" folder is currently selected.
- Editor:** The "User.php" file is open. The code is as follows:

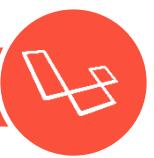
```
use Illuminate\Foundation\Auth\User as Authenticatable;
class User extends Authenticatable
{
    use Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password', 'user_type',
    ];

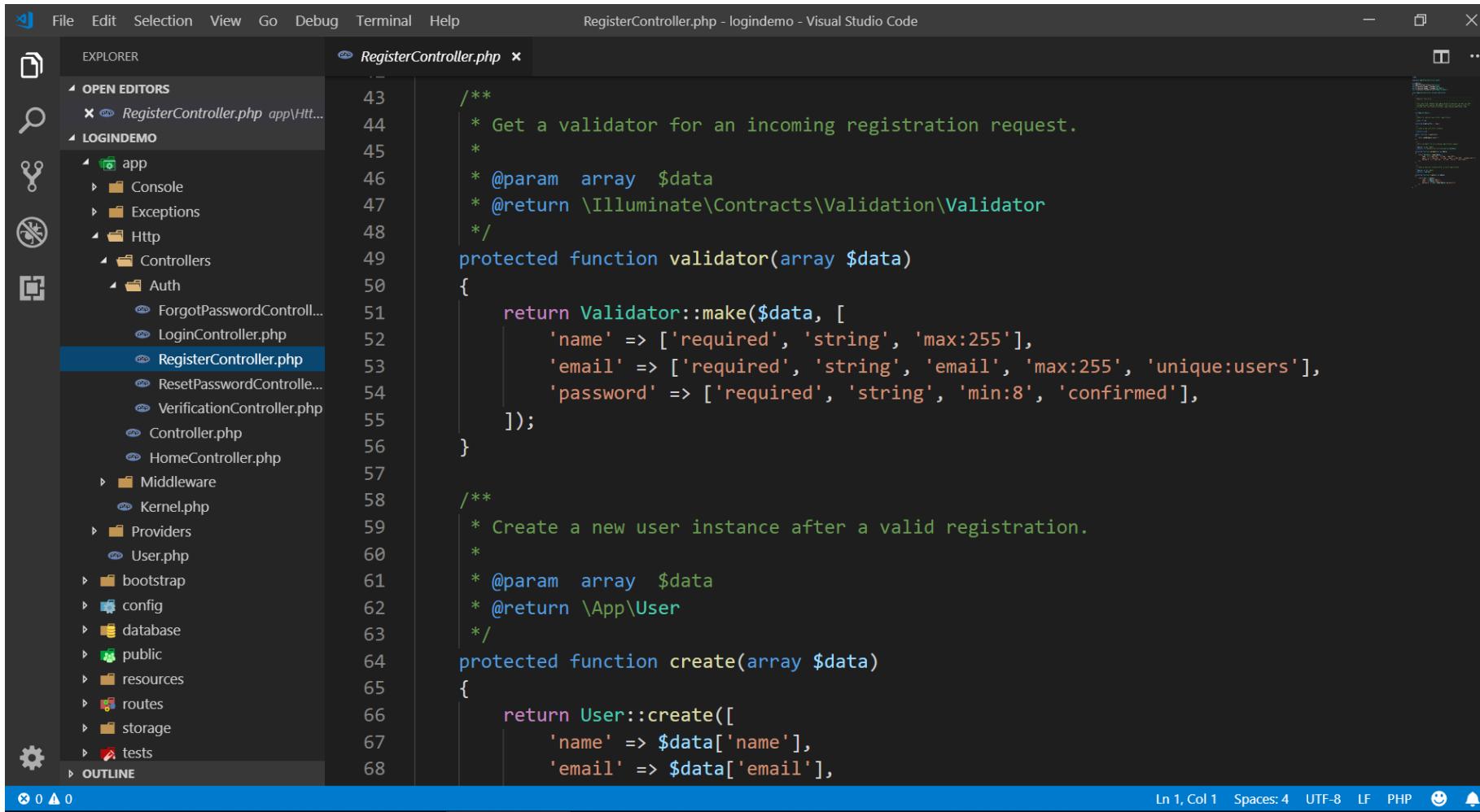
    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    /**
     * The attributes that should be cast to native types.
     *
     */
}
```

The status bar at the bottom indicates: Ln 20, Col 7 Spaces: 4 UTF-8 LF PHP



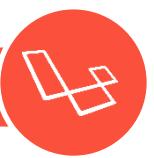
# Open RegisterController.php



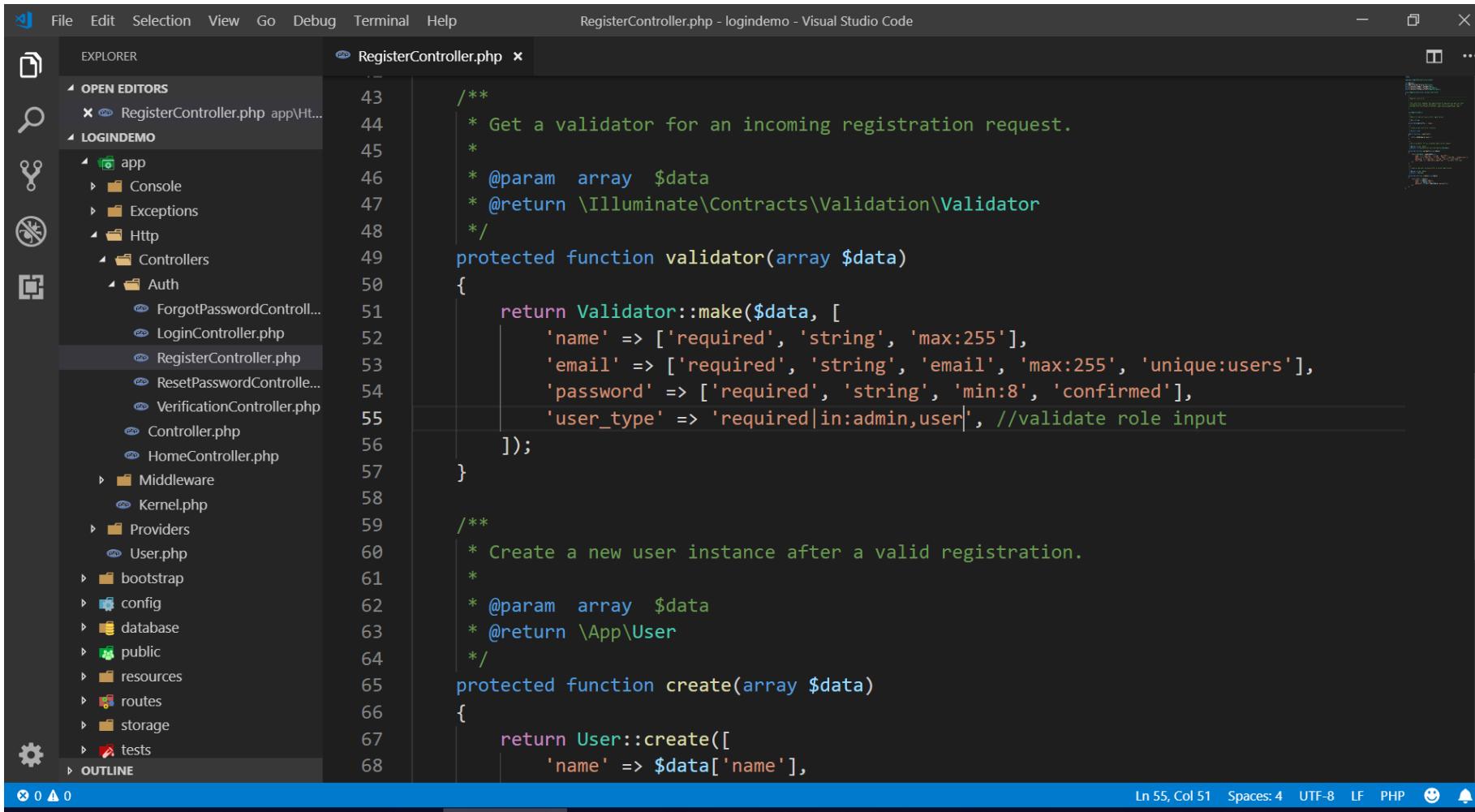
The screenshot shows the Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** RegisterController.php - logindemo - Visual Studio Code.
- Explorer View:** Shows the project structure under "OPEN EDITORS" and "LOGINDEMO". The "RegisterController.php" file is selected and highlighted in blue.
- Editor View:** Displays the PHP code for the RegisterController.php file. The code defines two protected functions: "validator" and "create".
- Bottom Status Bar:** ShowsLn 1, Col 1, Spaces: 4, UTF-8, LF, PHP, and a few icons.

```
43     /**
44      * Get a validator for an incoming registration request.
45      *
46      * @param array $data
47      * @return \Illuminate\Contracts\Validation\Validator
48      */
49     protected function validator(array $data)
50     {
51         return Validator::make($data, [
52             'name' => ['required', 'string', 'max:255'],
53             'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
54             'password' => ['required', 'string', 'min:8', 'confirmed'],
55         ]);
56     }
57
58     /**
59      * Create a new user instance after a valid registration.
60      *
61      * @param array $data
62      * @return \App\User
63      */
64     protected function create(array $data)
65     {
66         return User::create([
67             'name' => $data['name'],
68             'email' => $data['email'],
69         ]);
70     }
71 }
```



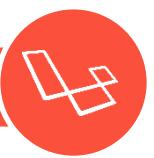
# Add Validator of User\_type



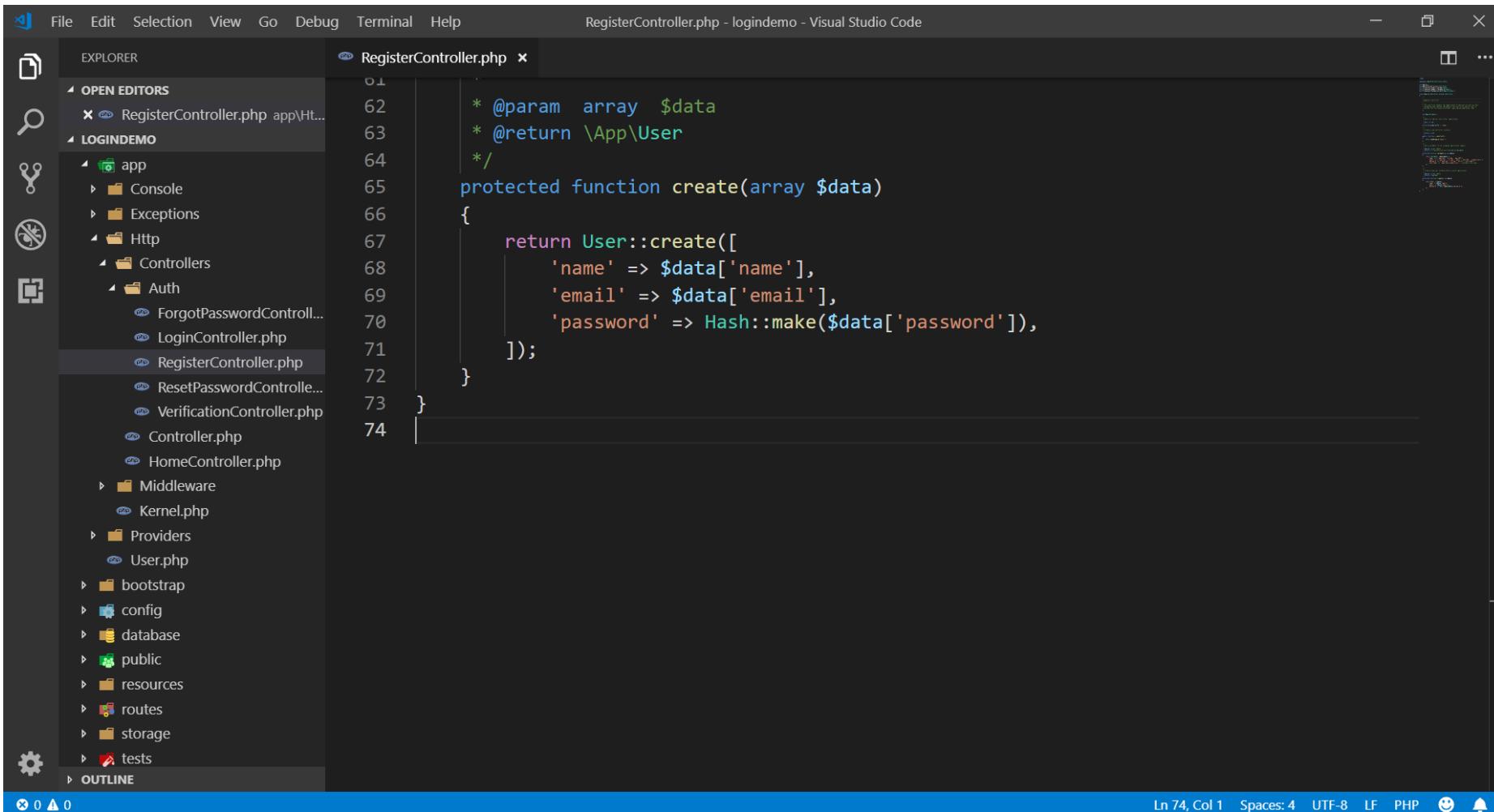
The screenshot shows a Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** RegisterController.php - logindemo - Visual Studio Code.
- Explorer:** Shows the project structure under "OPEN EDITORS" and "LOGINDEMO". The "RegisterController.php" file is selected in the list.
- Code Editor:** Displays the PHP code for the RegisterController.php file. The code includes a validator function that checks the "user\_type" field against the values "admin" or "user".
- Bottom Status Bar:** Shows the current line (Ln 55), column (Col 51), spaces (Spaces: 4), encoding (UTF-8), line separator (LF), PHP, and a few icons.

```
43     /**
44      * Get a validator for an incoming registration request.
45      *
46      * @param array $data
47      * @return \Illuminate\Contracts\Validation\Validator
48      */
49 protected function validator(array $data)
50 {
51     return Validator::make($data, [
52         'name' => ['required', 'string', 'max:255'],
53         'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
54         'password' => ['required', 'string', 'min:8', 'confirmed'],
55         'user_type' => 'required|in:admin,user', //validate role input
56     ]);
57 }
58 /**
59  * Create a new user instance after a valid registration.
60  *
61  * @param array $data
62  * @return \App\User
63  */
64 protected function create(array $data)
65 {
66     return User::create([
67         'name' => $data['name'],
```



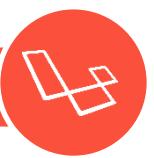
# Goto Create



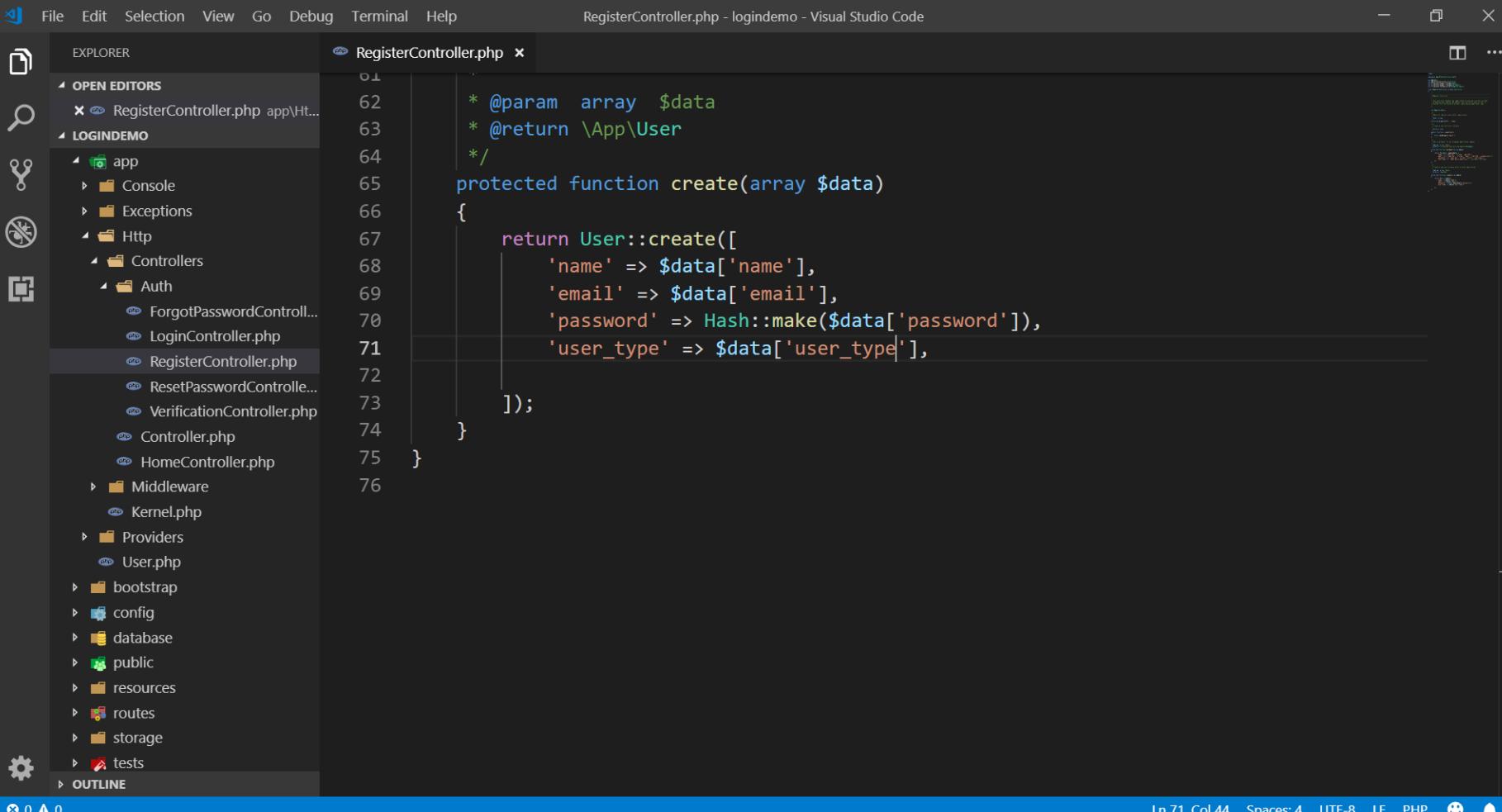
A screenshot of Visual Studio Code showing the file `RegisterController.php`. The code defines a protected function `create` that takes an array `$data` and returns a User object. The code uses the `User::create` method with three parameters: name, email, and password (hashed).

```
* @param array $data
* @return \App\User
*/
protected function create(array $data)
{
    return User::create([
        'name' => $data['name'],
        'email' => $data['email'],
        'password' => Hash::make($data['password']),
    ]);
}
```

The Explorer sidebar shows the project structure under `LOGINDEMO`, including `app`, `Http`, `Auth`, and other controller files like `LoginController.php` and `ResetPasswordController.php`. The status bar at the bottom indicates the file is `RegisterController.php`, line 74, column 1, with 4 spaces, using UTF-8 encoding.



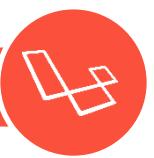
# Add UserType



The screenshot shows a Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** RegisterController.php - logindemo - Visual Studio Code.
- Explorer:** Shows the project structure under "OPEN EDITORS" and "LOGINDEMO". The "Auth" folder contains files like ForgotPasswordController.php, LoginController.php, RegisterController.php, ResetPasswordController.php, VerificationController.php, Controller.php, and HomeController.php. "Middleware" and "Providers" also contain User.php.
- Code Editor:** The "RegisterController.php" file is open, showing the following PHP code:

```
* @param array $data
* @return \App\User
*/
protected function create(array $data)
{
    return User::create([
        'name' => $data['name'],
        'email' => $data['email'],
        'password' => Hash::make($data['password']),
        'user_type' => $data['user_type'],
    ]);
}
```
- Status Bar:** Ln 71, Col 44, Spaces: 4, UTF-8, LF, PHP, smiley face, bell icon.



# Create MiddleWare of Admin

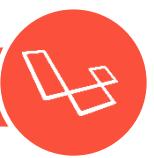
C:\Windows\system32\cmd.exe

```
c:\xampp\htdocs\logindemo>php artisan make:middleware Admin
```



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



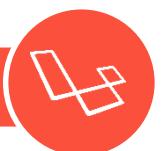
# Create User MiddleWare

C:\Windows\system32\cmd.exe

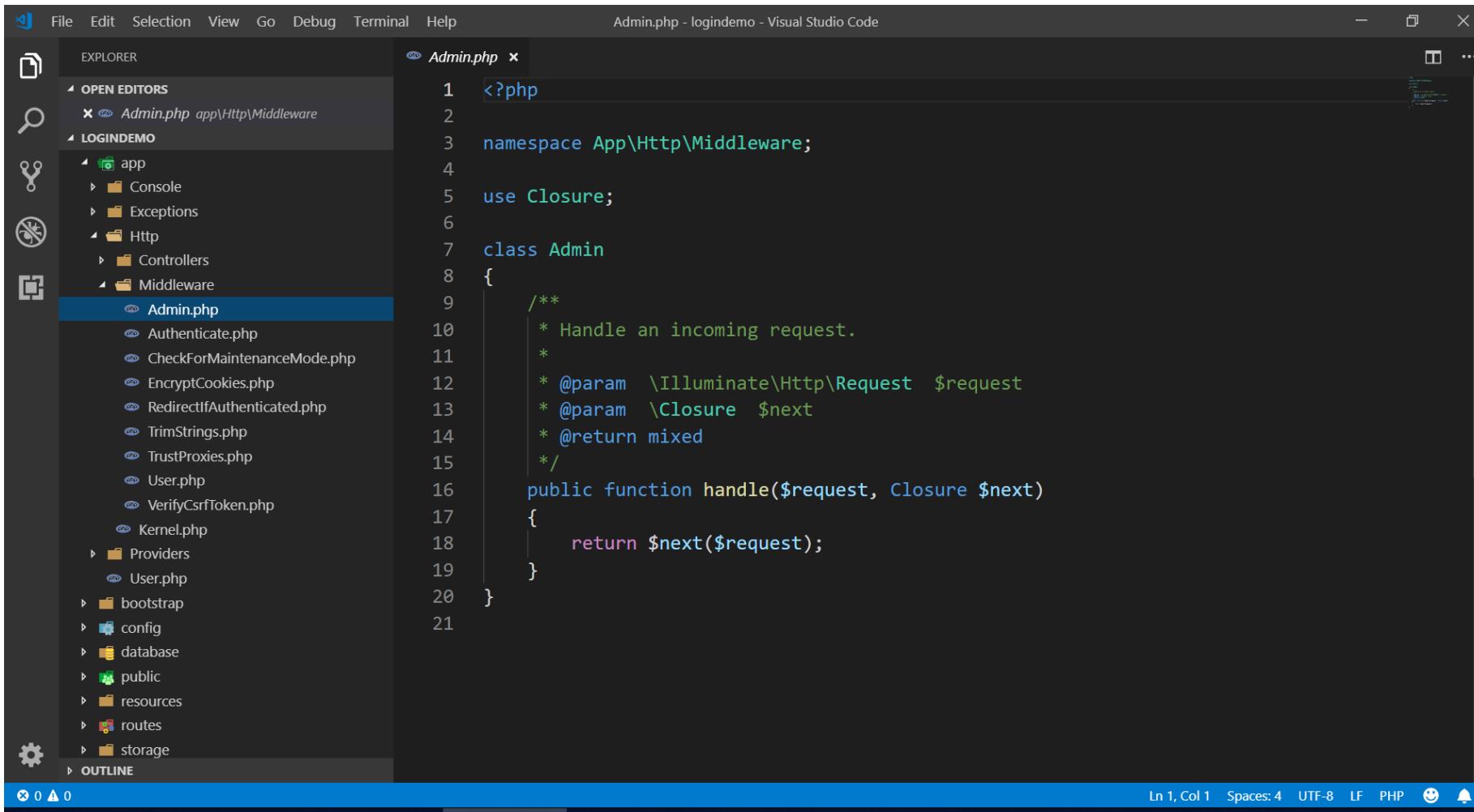
```
c:\xampp\htdocs\logindemo>php artisan make:middleware Admin  
Middleware created successfully.
```

```
c:\xampp\htdocs\logindemo>php artisan make:middleware User  
Middleware created successfully.
```

```
c:\xampp\htdocs\logindemo>
```



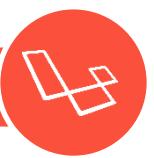
# Goto MiddleWare->Admin.php



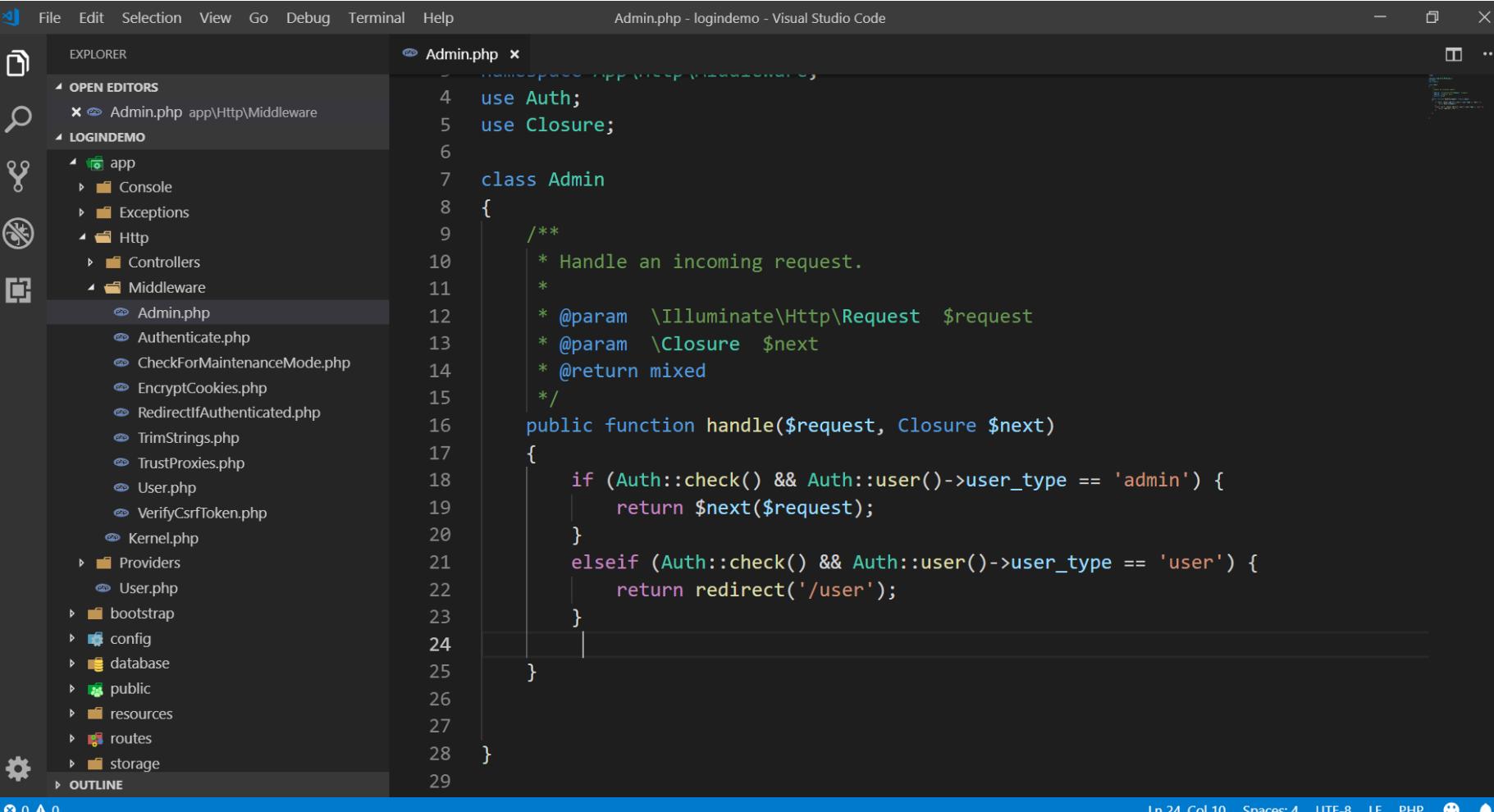
The screenshot shows the Visual Studio Code interface with the Admin.php file open in the editor. The file is located in the app\Http\Middleware directory of a project named LOGINDEMO. The code defines a class Admin that handles incoming requests using a Closure.

```
<?php  
namespace App\Http\Middleware;  
  
use Closure;  
  
class Admin  
{  
    /**  
     * Handle an incoming request.  
     *  
     * @param \Illuminate\Http\Request $request  
     * @param Closure $next  
     * @return mixed  
     */  
    public function handle($request, Closure $next)  
    {  
        return $next($request);  
    }  
}
```

The Explorer sidebar on the left shows the project structure, including the app, Http, and Middleware directories. The Admin.php file is selected in the list. The status bar at the bottom indicates the file is at Line 1, Column 1, with 4 spaces, in UTF-8 encoding, and is a PHP file.



# Add “Use Auth” and Condition

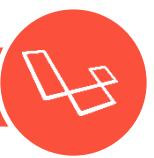


The screenshot shows the Visual Studio Code interface with the Admin.php file open in the editor. The code implements a middleware class named Admin that checks the user type before executing the next middleware or route handler.

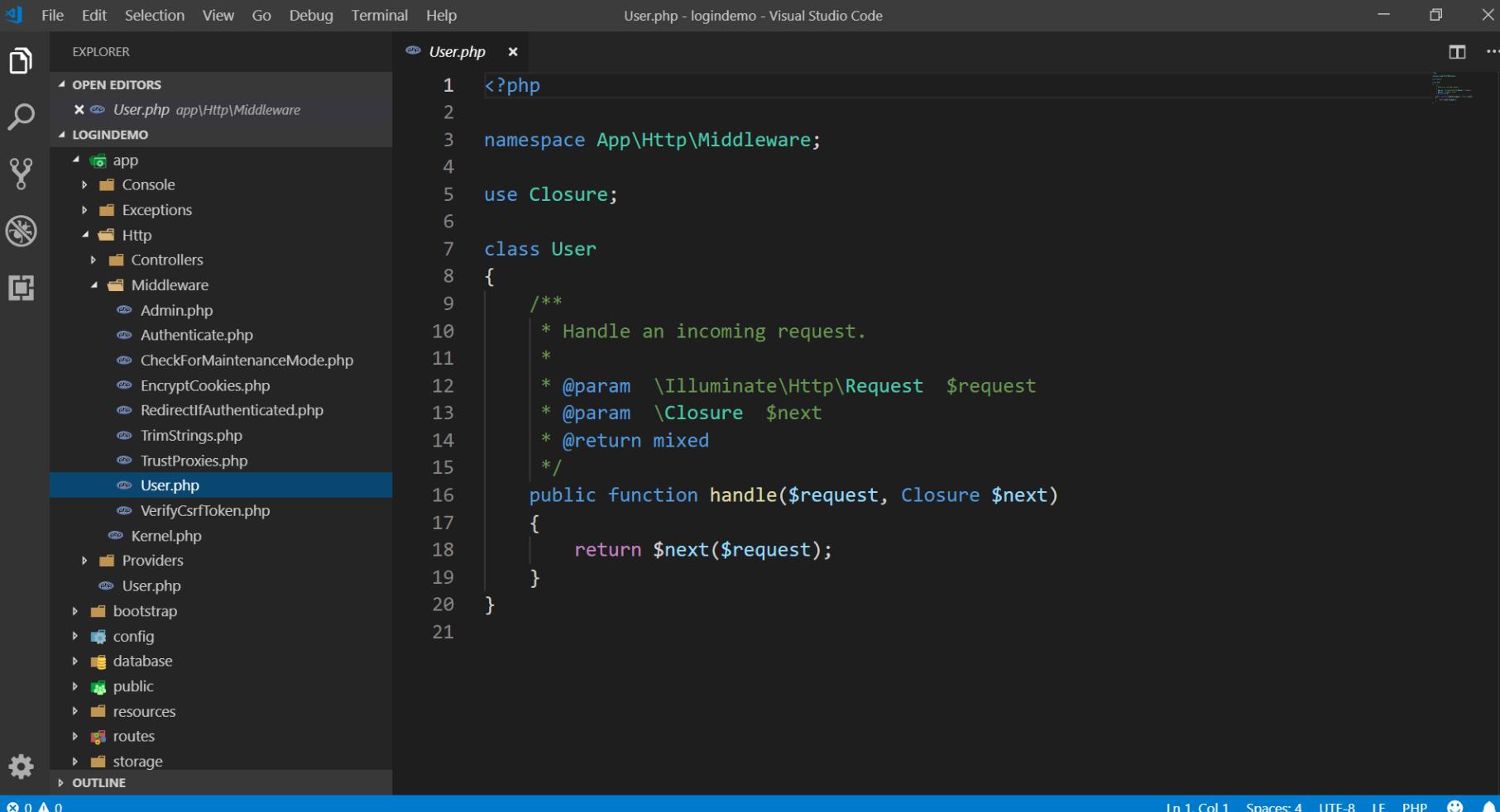
```
use Auth;
use Closure;

class Admin
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param Closure $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
        if (Auth::check() && Auth::user()->user_type == 'admin') {
            return $next($request);
        }
        elseif (Auth::check() && Auth::user()->user_type == 'user') {
            return redirect('/user');
        }
    }
}
```

The code editor shows syntax highlighting for PHP and annotations for parameters and return types. The status bar at the bottom indicates the file is at line 24, column 10, with 4 spaces, in UTF-8 encoding, and is a PHP file.



# Goto MiddleWare User.php

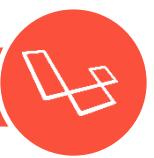


The screenshot shows the Visual Studio Code interface with the following details:

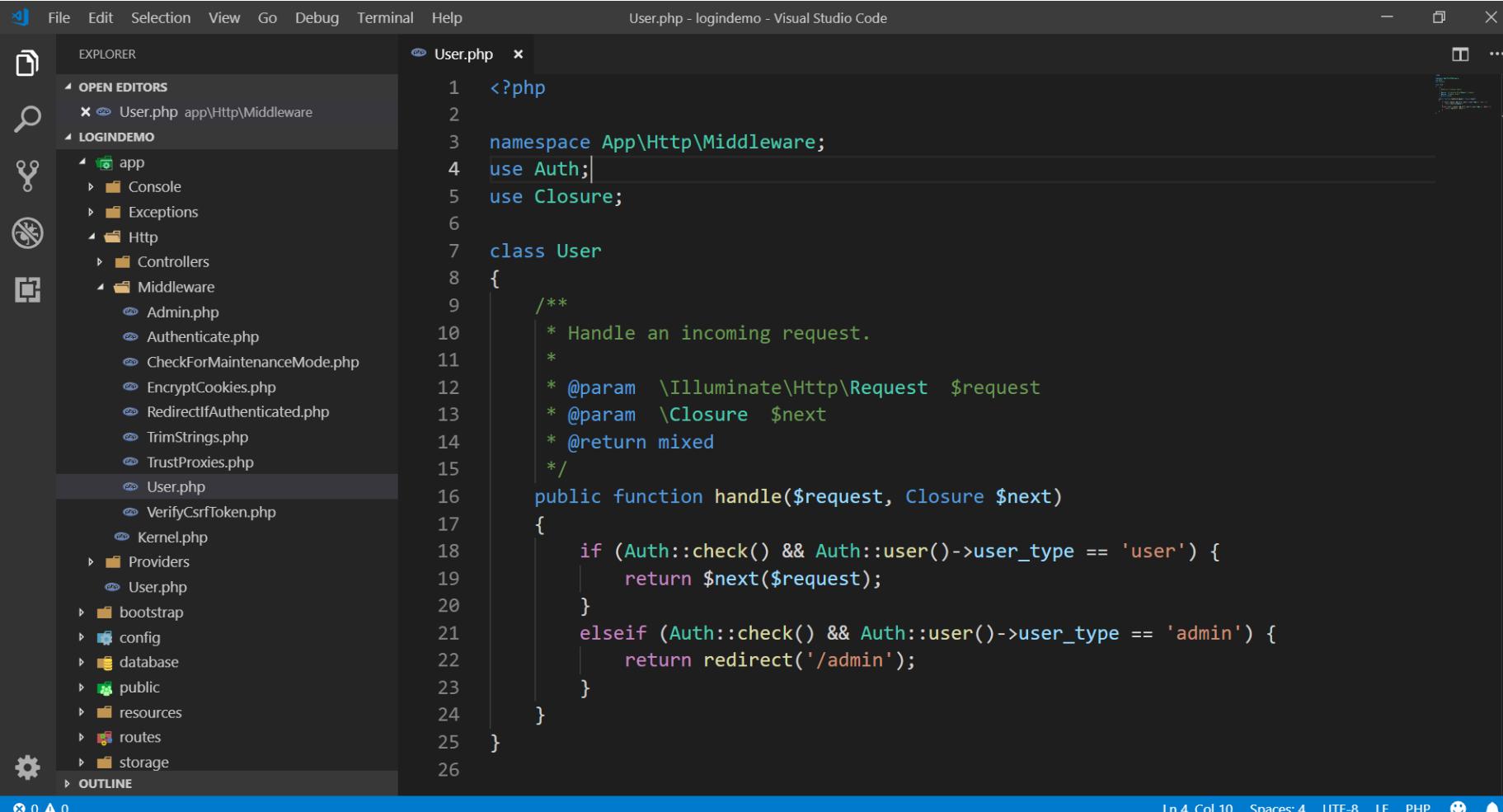
- File Path:** User.php - logindemo - Visual Studio Code
- Editor Content:**

```
<?php
namespace App\Http\Middleware;
use Closure;

class User
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param Closure $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
        return $next($request);
    }
}
```
- Explorer View:** Shows the project structure under "LOGINDEMO". The "Middleware" folder is expanded, and "User.php" is selected, highlighted with a blue background.
- Status Bar:** Shows "Ln 1, Col 1" and "Spaces: 4" and "UTF-8 LF PHP".



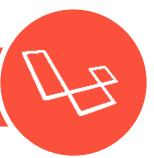
# Add “Use Auth” and Condition



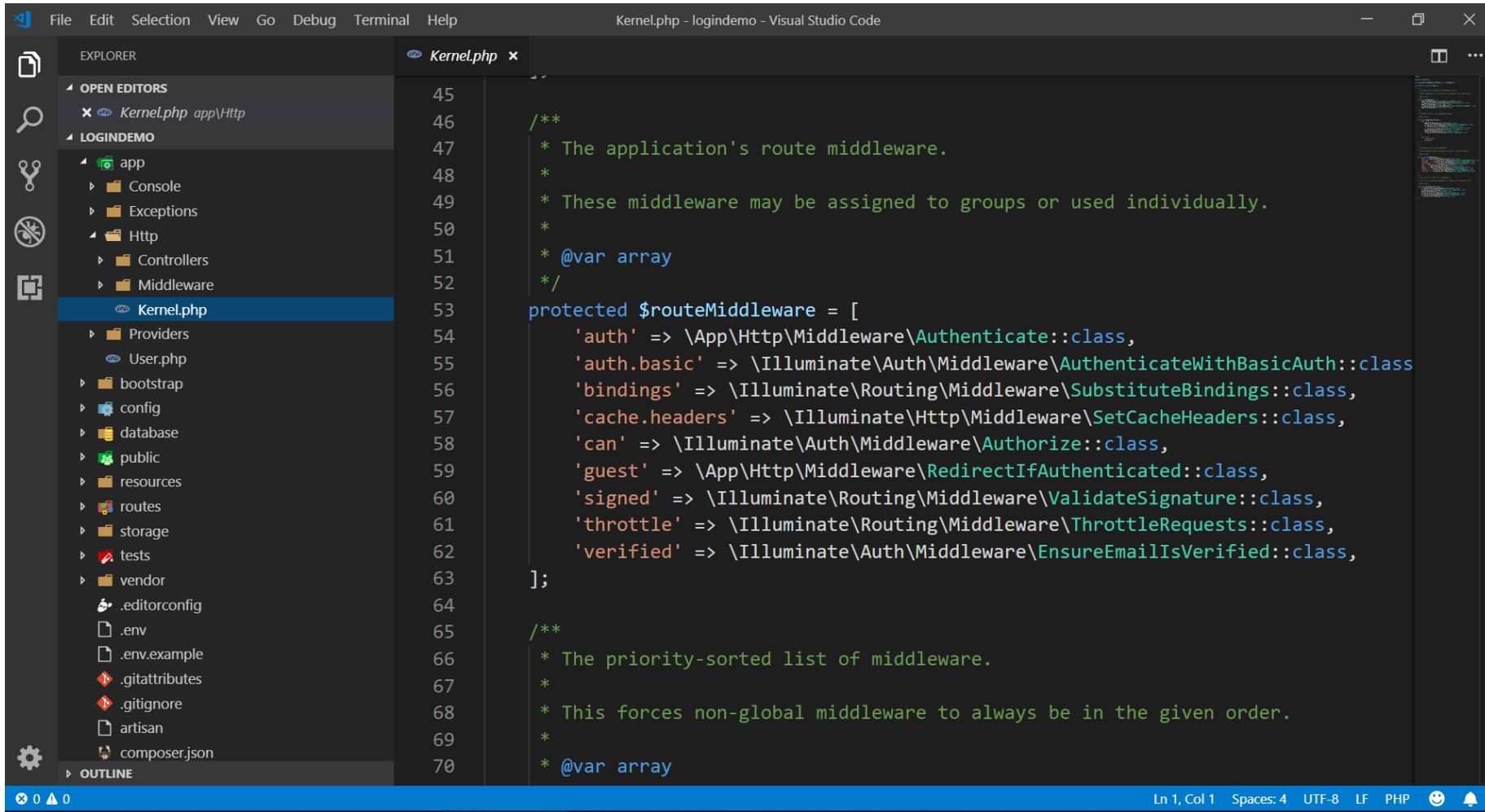
The screenshot shows the Visual Studio Code interface with the following details:

- File Path:** User.php - logindemo - Visual Studio Code
- Editor Content:**

```
1 <?php
2
3 namespace App\Http\Middleware;
4 use Auth;
5 use Closure;
6
7 class User
8 {
9     /**
10      * Handle an incoming request.
11      *
12      * @param \Illuminate\Http\Request $request
13      * @param \Closure $next
14      * @return mixed
15      */
16     public function handle($request, Closure $next)
17     {
18         if (Auth::check() && Auth::user()->user_type == 'user') {
19             return $next($request);
20         }
21         elseif (Auth::check() && Auth::user()->user_type == 'admin') {
22             return redirect('/admin');
23         }
24     }
25 }
```
- Explorer View:** Shows the project structure under "OPEN EDITORS" and "LOGINDEMO". The "Middleware" folder contains several files: Admin.php, Authenticate.php, CheckForMaintenanceMode.php, EncryptCookies.php, RedirectIfAuthenticated.php, TrimStrings.php, TrustProxies.php, User.php (which is selected), VerifyCsrfToken.php, and Kernel.php.
- Status Bar:** Shows "Ln 4, Col 10" and other settings like "Spaces: 4", "UTF-8", "LF", "PHP", and icons for file status and notifications.



# Open app/http/Kernel.php

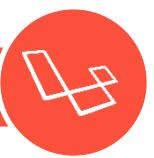


The screenshot shows a Visual Studio Code interface with the following details:

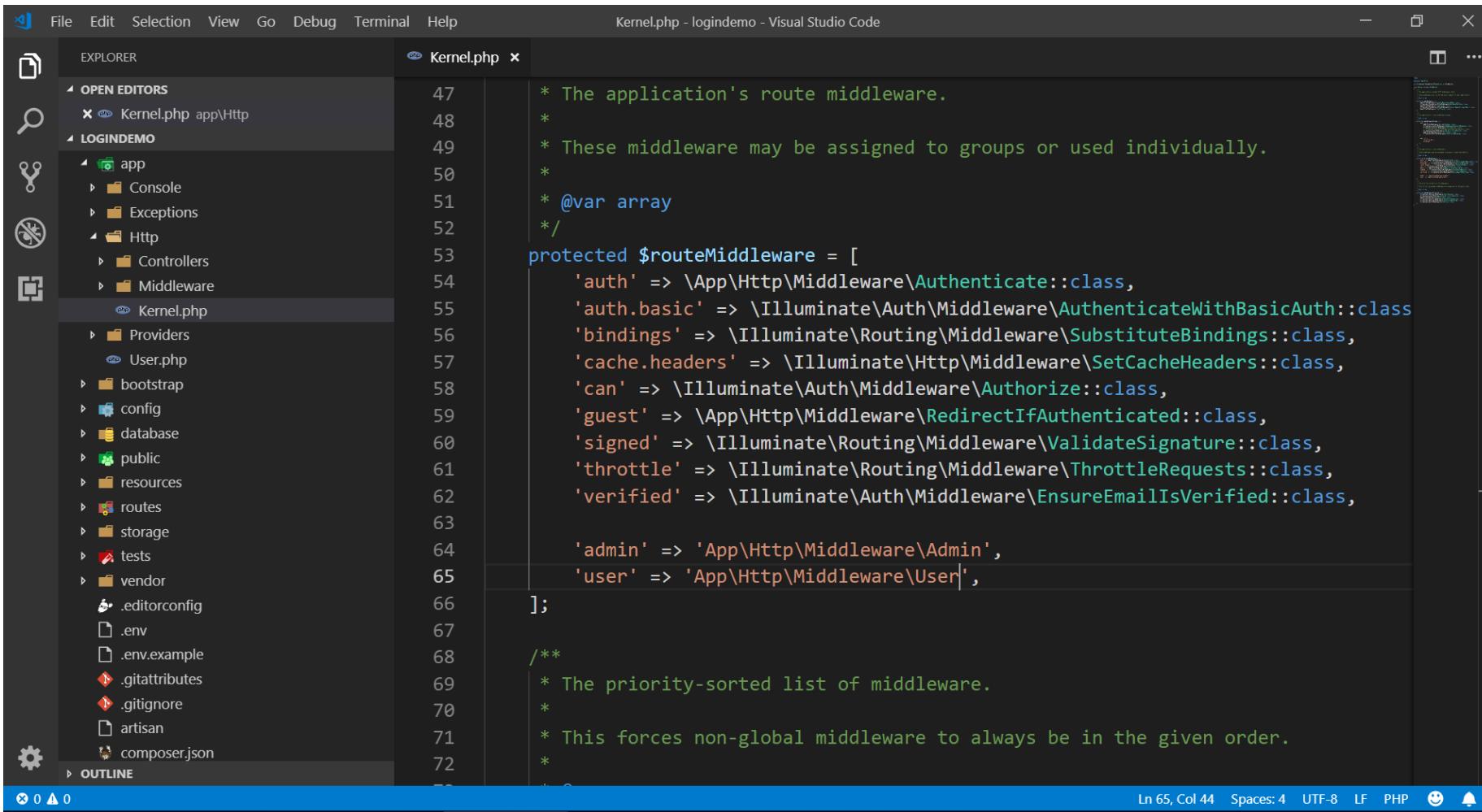
- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** Kernel.php - logindemo - Visual Studio Code.
- Explorer View:** Shows the project structure under "OPEN EDITORS" and "LOGINDEMO". The "Kernel.php" file is selected in the "Kernel.php" editor tab.
- Editor Tab:** Kernel.php
- Code Content:** The code is a PHP file defining the Kernel class. It includes comments explaining route middleware and priority sorted middleware lists, and uses the Illuminate\Support\Arr::var() method.

```
45
46     /**
47      * The application's route middleware.
48      *
49      * These middleware may be assigned to groups or used individually.
50      *
51      * @var array
52     */
53 protected $routeMiddleware = [
54     'auth' => \App\Http\Middleware\Authenticate::class,
55     'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
56     'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,
57     'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
58     'can' => \Illuminate\Auth\Middleware\Authorize::class,
59     'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
60     'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,
61     'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
62     'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
63 ];
64
65 /**
66  * The priority-sorted list of middleware.
67  *
68  * This forces non-global middleware to always be in the given order.
69  *
70  * @var array
```

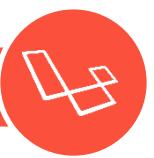
- Bottom Status Bar:** Ln 1, Col 1 | Spaces: 4 | UTF-8 | LF | PHP | 🎉 | 🔔



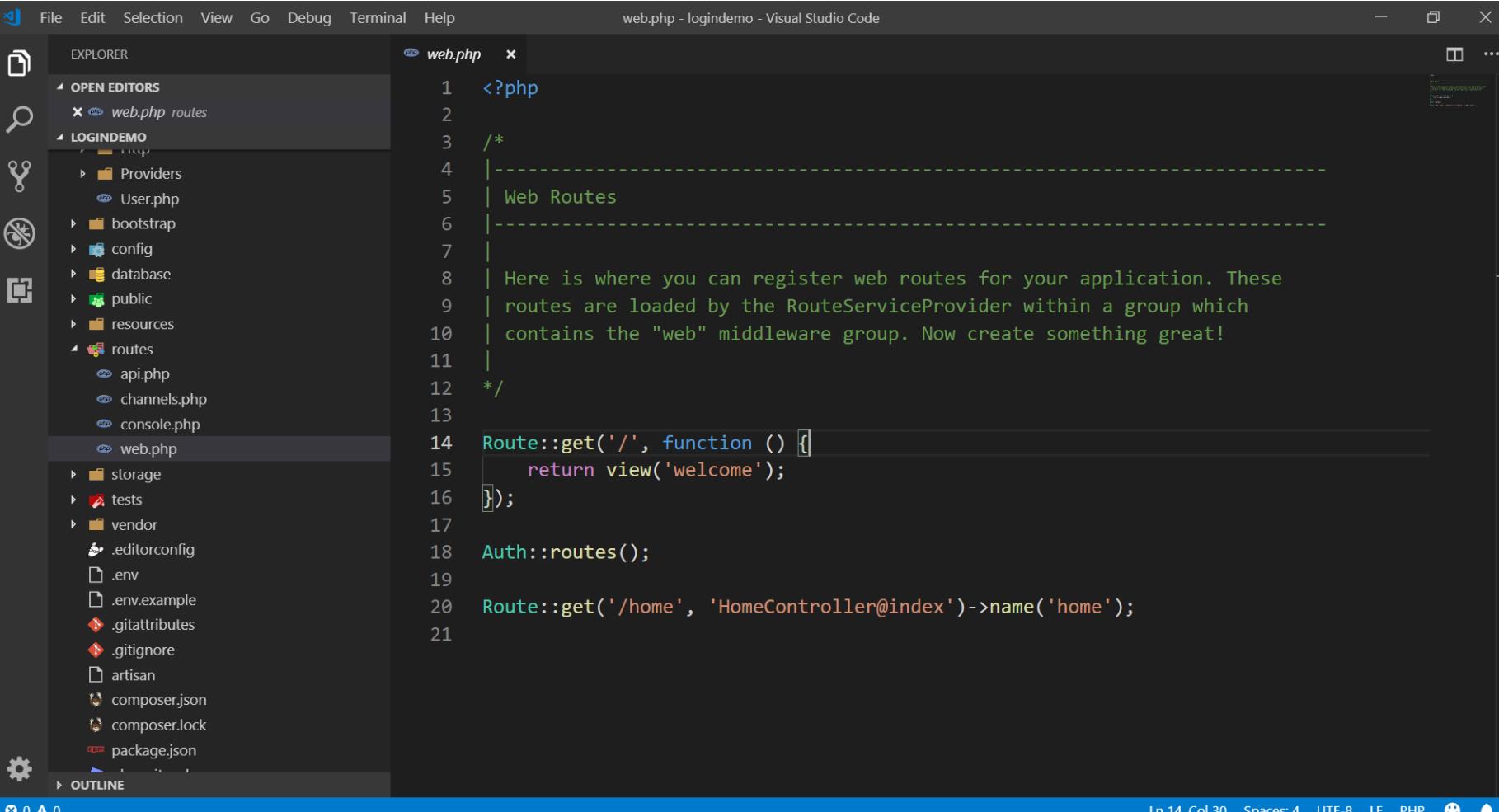
# Add Admin,User Middleware



```
* The application's route middleware.  
*  
* These middleware may be assigned to groups or used individually.  
*  
* @var array  
*/  
  
protected $routeMiddleware = [  
    'auth' => \App\Http\Middleware\Authenticate::class,  
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,  
    'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,  
    'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,  
    'can' => \Illuminate\Auth\Middleware\Authorize::class,  
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,  
    'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,  
    'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,  
    'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,  
  
    'admin' => 'App\Http\Middleware\Admin',  
    'user' => 'App\Http\Middleware\User',  
];  
  
/**  
 * The priority-sorted list of middleware.  
 *  
 * This forces non-global middleware to always be in the given order.  
 */
```



# Open Web.php



The screenshot shows the Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** web.php - logindemo - Visual Studio Code.
- Explorer View:** Shows the project structure for a Laravel application named "logindemo". The "routes" directory is expanded, showing files like api.php, channels.php, console.php, and web.php, which is currently selected.
- Code Editor:** Displays the content of the web.php file. The code includes comments explaining the purpose of the routes and provides examples of route definitions.
- Status Bar:** Shows the current line (Ln 14), column (Col 30), spaces (Spaces: 4), encoding (UTF-8), line feed (LF), PHP, and a few icons for status and notifications.

```
<?php
/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| contains the "web" middleware group. Now create something great!
|
*/
Route::get('/', function () {
    return view('welcome');
});

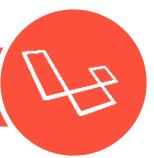
Auth::routes();

Route::get('/home', 'HomeController@index')->name('home');
```

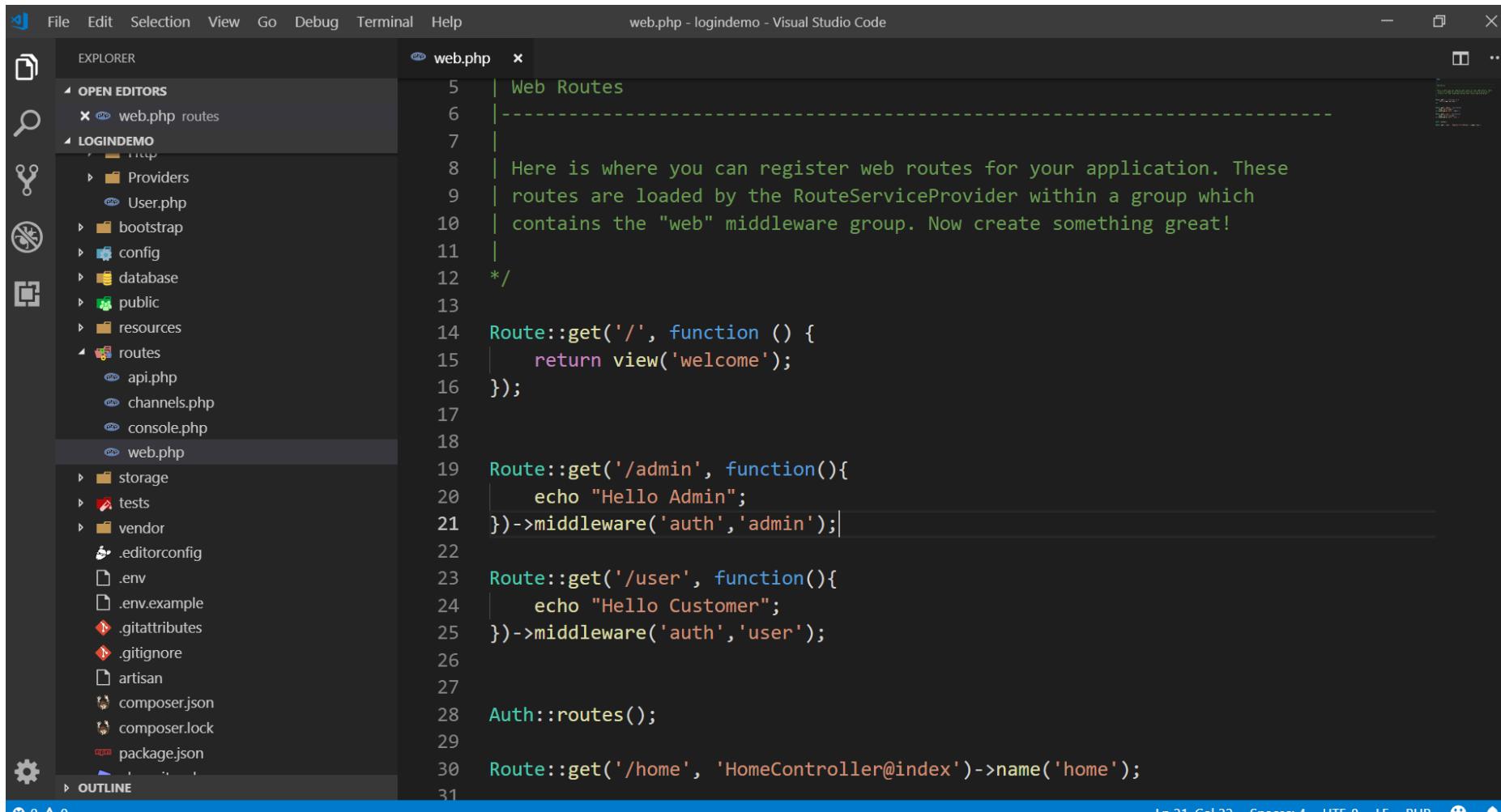


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)

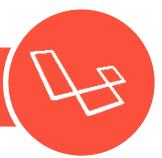


# Add Router with Middleware

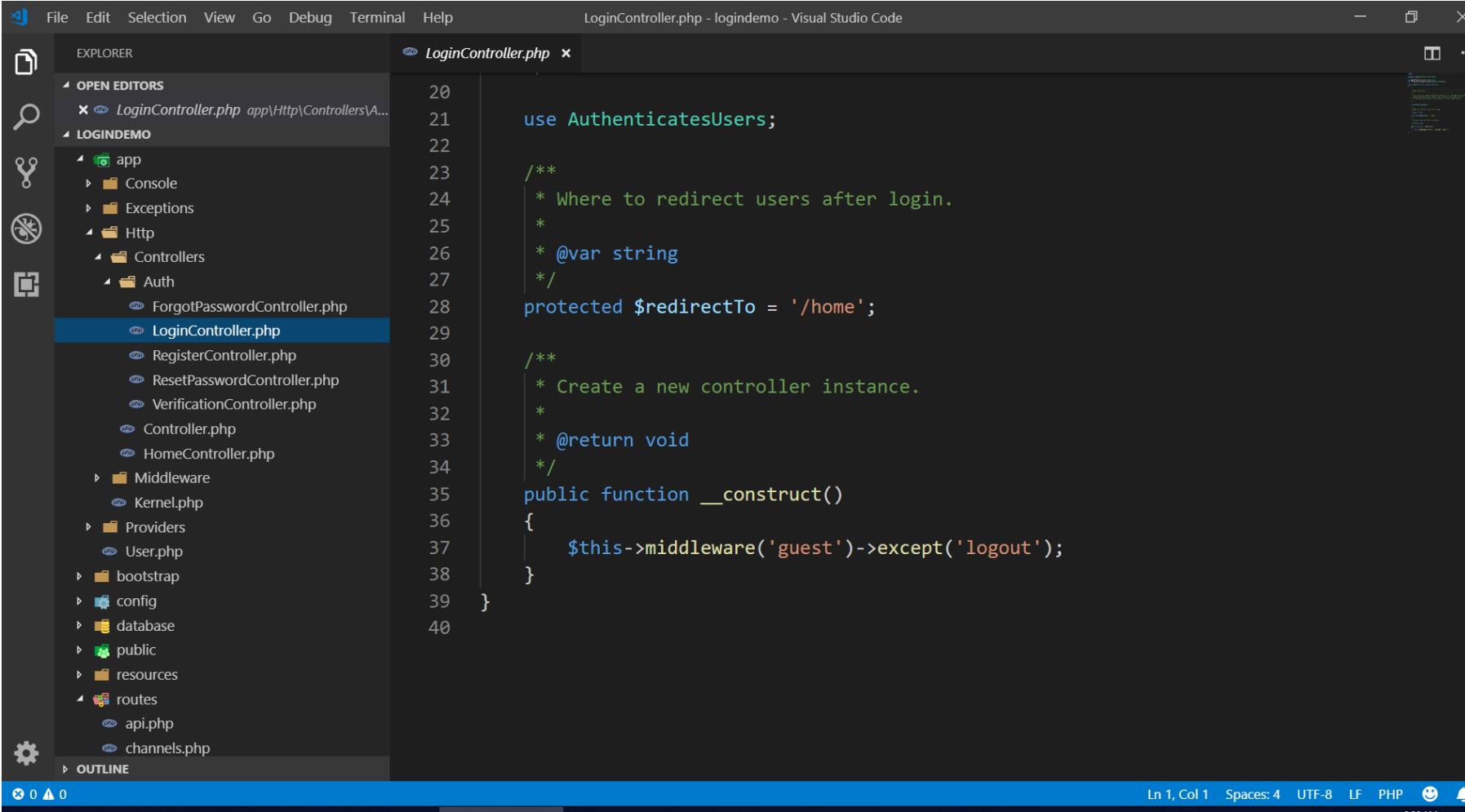


The screenshot shows the Visual Studio Code interface with the 'web.php' file open in the editor. The file is part of a Laravel application named 'LOGINDEMO'. The code in 'web.php' defines several routes:

```
5 | Web Routes
6 |
7 |
8 | Here is where you can register web routes for your application. These
9 | routes are loaded by the RouteServiceProvider within a group which
10 | contains the "web" middleware group. Now create something great!
11 |
12 */
13
14 Route::get('/', function () {
15     return view('welcome');
16 });
17
18
19 Route::get('/admin', function(){
20     echo "Hello Admin";
21 })->middleware('auth','admin');
22
23 Route::get('/user', function(){
24     echo "Hello Customer";
25 })->middleware('auth','user');
26
27
28 Auth::routes();
29
30 Route::get('/home', 'HomeController@index')->name('home');
31
```



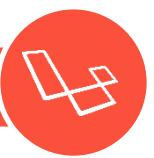
# Open LoginController.php & Remove Redirecto Line



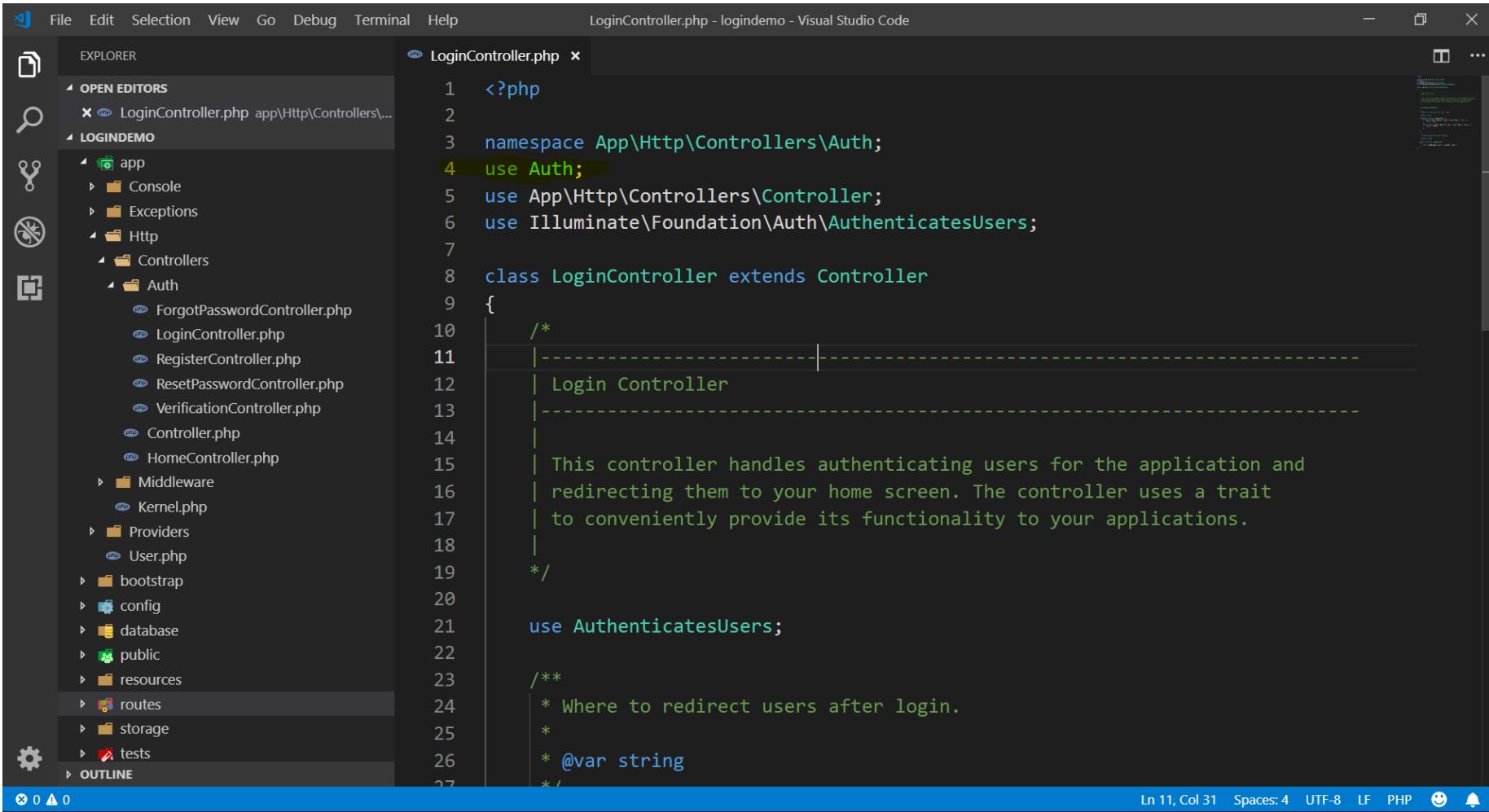
```
use AuthenticatesUsers;

/**
 * Where to redirect users after login.
 *
 * @var string
 */
protected $redirectTo = '/home';

/**
 * Create a new controller instance.
 *
 * @return void
 */
public function __construct()
{
    $this->middleware('guest')->except('logout');
}
```

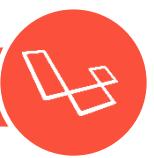


# Removed Line

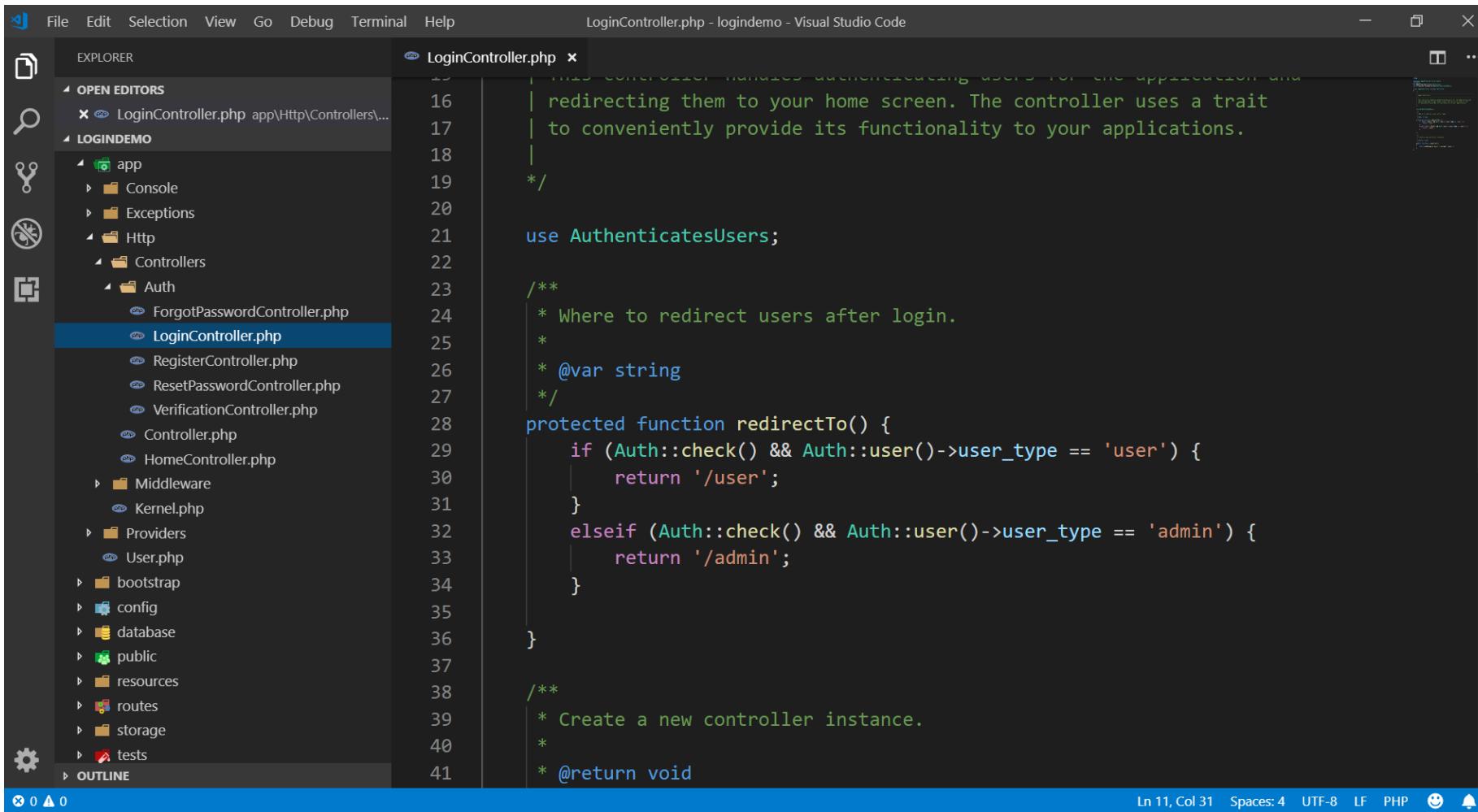


A screenshot of Visual Studio Code showing the `LoginController.php` file. The code is a PHP class definition for a login controller. A specific line of code, `use Auth;`, has been removed, indicated by a dashed red line and a red squiggle under the word `Auth`. The code also includes a detailed docblock describing the controller's purpose and functionality.

```
<?php  
namespace App\Http\Controllers\Auth;  
use Auth;  
use App\Http\Controllers\Controller;  
use Illuminate\Foundation\Auth\AuthenticatesUsers;  
  
class LoginController extends Controller  
{  
    /*  
     *-----|  
     | Login Controller  
     |-----  
     *-----|  
     | This controller handles authenticating users for the application and  
     | redirecting them to your home screen. The controller uses a trait  
     | to conveniently provide its functionality to your applications.  
     |-----  
     */  
  
    use AuthenticatesUsers;  
  
    /**  
     * Where to redirect users after login.  
     *  
     * @var string  
     */
```



# Add Function of Redirect With Condition



The screenshot shows a Visual Studio Code interface with the following details:

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** LoginController.php - logindemo - Visual Studio Code.
- Explorer:** Shows the project structure under "OPEN EDITORS" and "LOGINDEMO". The "LoginController.php" file is selected in the list.
- Editor:** Displays the PHP code for the LoginController.php file. The code includes a trait for authentication and a protected function "redirectTo()" which checks the user type (user or admin) to determine the redirect path.
- Bottom Status Bar:** Shows file statistics (Ln 11, Col 31), code style (Spaces: 4), encoding (UTF-8), line endings (LF), PHP, and a few icons.

```
use Illuminate\Foundation\Auth\Access\AuthorizesUsers;
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Foundation\Auth\ResetsPasswords;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Password;
use Illuminate\Support\Facades\Validator;
use Illuminate\Validation\Rule;
use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Foundation\Auth\ThrottlesLogins;
use Illuminate\Foundation\Auth\ValidatesRequests;
use Illuminate\Foundation\Auth\AuthenticatesUsers;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Redirect;
use Illuminate\Support\Facades\View;
```

```
class LoginController extends Controller
{
    /**
     * Where to redirect users after login.
     *
     * @var string
     */
    protected $redirectTo = '/';

    /**
     * Create a new controller instance.
     */
    public function __construct()
    {
        $this->middleware('guest');
        $this->middleware('throttle:login');
        $this->middleware('authenticatesusers');
    }

    /**
     * Handle a POST request to log the user in.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function login(Request $request)
    {
        $this->validate($request);

        $credentials = $request->only('email', 'password');

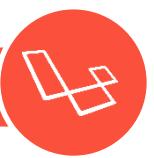
        if (Auth::attempt($credentials)) {
            $request->session()->regenerate();
            return $this->redirectTo();
        }

        return back()
            ->withInput($request->only('email'))
            ->withErrors(['email' => trans('auth.failed')]);
    }

    /**
     * Log the user out.
     *
     * @return \Illuminate\Http\Response
     */
    public function logout()
    {
        Auth::logout();

        $request->session()->regenerate();

        return Redirect::to('/login');
    }
}
```



# Run Project

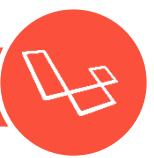
```
C:\Windows\system32\cmd.exe - php artisan serve
```

```
c:\xampp\htdocs\logindemo>php artisan serve
Laravel development server started: <http://127.0.0.1:8000>
```



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Create Admin User

The screenshot shows a web browser window with the title "Laravel" and the URL "127.0.0.1:8000/register". The page displays a "Register" form. The form fields are as follows:

- Name: Akash
- E-Mail Address: akash.padhiyar@gmail.com
- Password: (obscured)
- Confirm Password: (obscured)
- Role: Admin

A blue "Register" button is located at the bottom of the form.

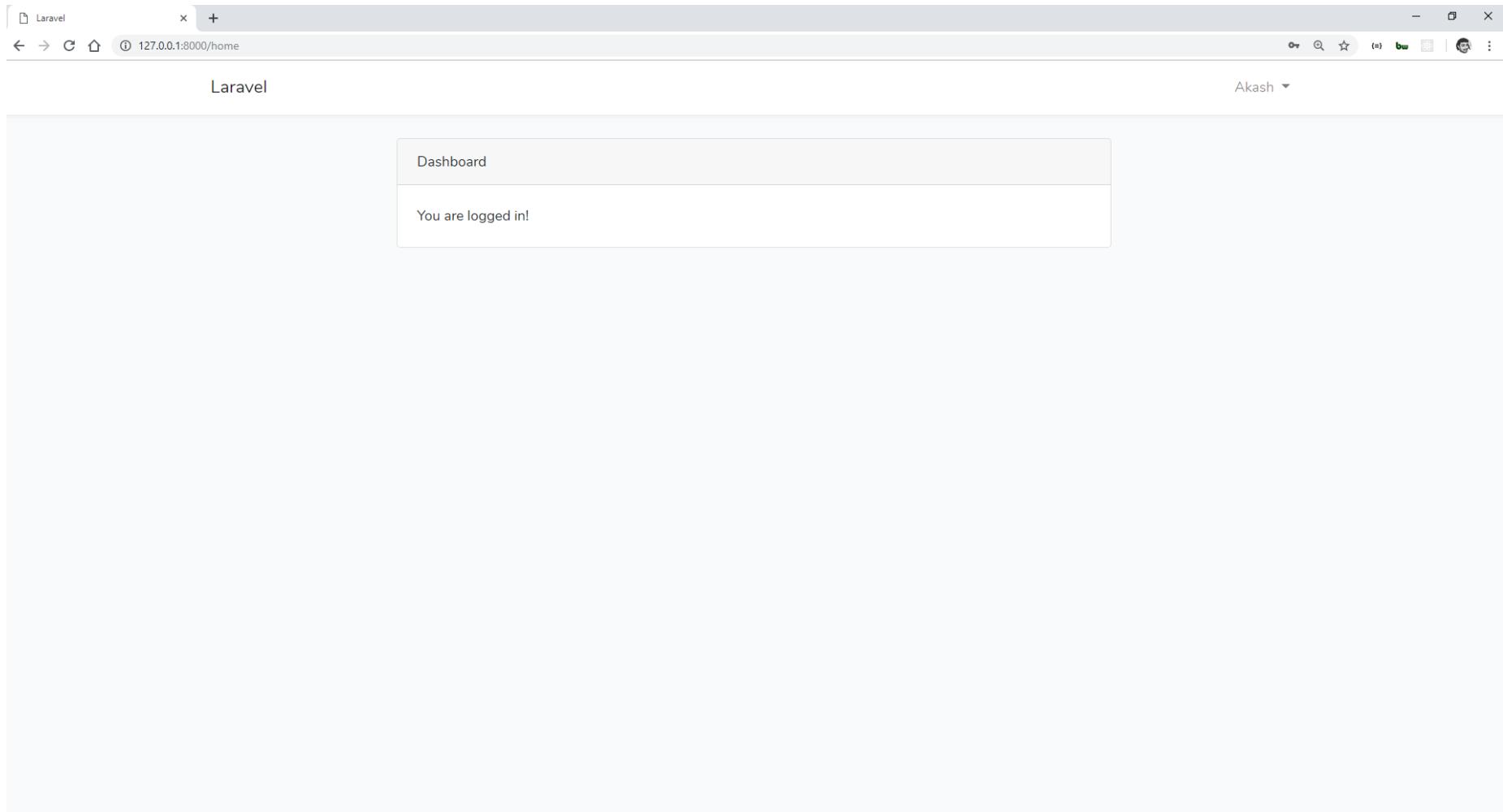


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)

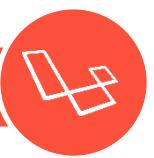


# Logout From Panel



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



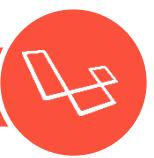
# Login Using Admin Role

The screenshot shows a web browser window titled "Laravel" with the URL "127.0.0.1:8000/login". The page displays a standard Laravel login form. The "E-Mail Address" field contains "akash.padhiyar@gmail.com" and the "Password" field contains a masked password. A "Remember Me" checkbox is unchecked. Below the form are "Login" and "Forgot Your Password?" buttons.

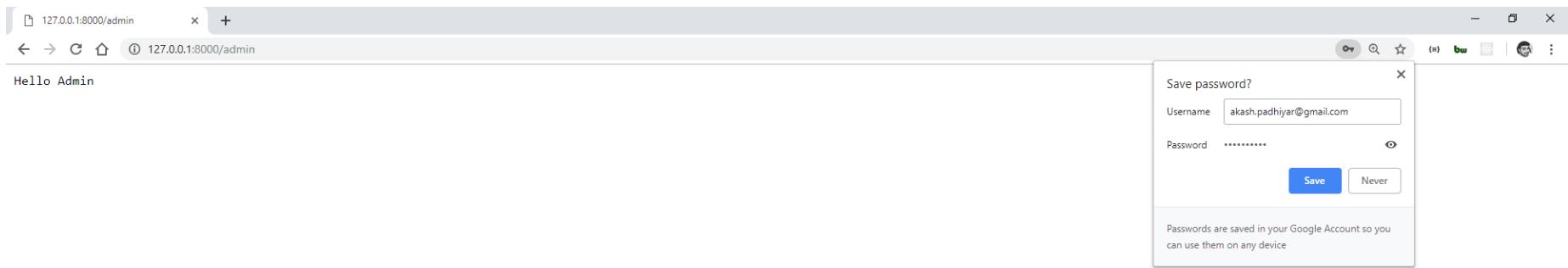


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)

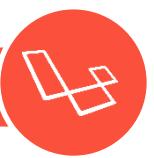


# Welcome to Admin Area



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Create User Role Account

A screenshot of a web browser window displaying a Laravel registration form. The title bar shows 'Laravel' and the URL '127.0.0.1:8000/register'. The page has a header with 'Laravel' on the left and 'Login Register' on the right. The main content is a 'Register' form with the following fields:

Field	Value
Name	User
E-Mail Address	user@gmail.com
Password	*****
Confirm Password	*****
Role	User

The 'Role' field has a dropdown arrow icon. A blue 'Register' button is at the bottom of the form.



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



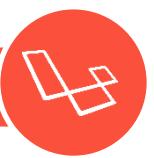
# Logout

A screenshot of a web browser window titled "Laravel". The address bar shows "127.0.0.1:8000/home". The page content includes a "Dashboard" header and a message "You are logged in!". In the top right corner, there is a "User" dropdown menu.

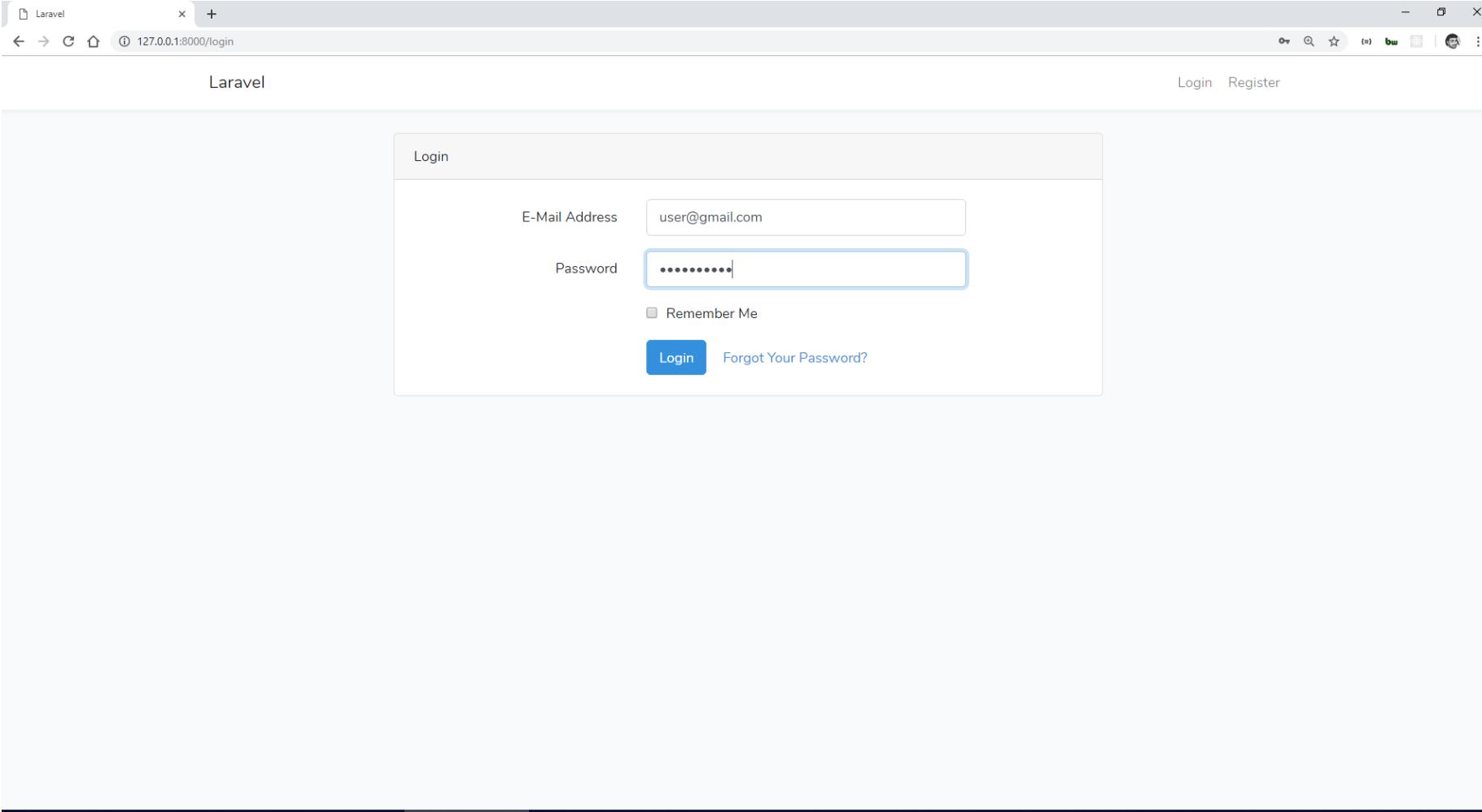


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Login Using User Account

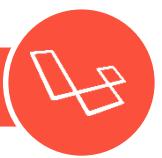


A screenshot of a web browser displaying a Laravel login form. The browser title bar shows "Laravel". The address bar displays "127.0.0.1:8000/login". The page header includes "Laravel" on the left and "Login Register" on the right. The main content is a "Login" form with fields for "E-Mail Address" (containing "user@gmail.com") and "Password" (containing masked text). There is a "Remember Me" checkbox, a "Login" button, and a "Forgot Your Password?" link.



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)

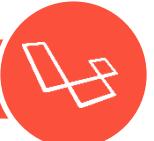


# Welcome to User



Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



# Check Record in PHPMyAdmin

← Server: 127.0.0.1 » Database: laravellogin » Table: users

☰

Browse Structure SQL Search Insert Export Import Privileges Operations Tracking More

Showing rows 0 - 1 (2 total, Query took 0.0011 seconds.)

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

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 ▾

+ Options

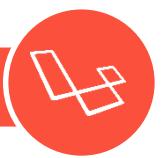
	id	name	email	email_verified_at	user_type	password
<input type="checkbox"/>	1	Akash	akash.padhiyar@gmail.com	NULL	admin	\$2y\$10\$KfNY29TUnxS.ng56u1lOOxaZ2XBZZy9ft6RD/iKjBU...
<input type="checkbox"/>	2	User	user@gmail.com	NULL	user	\$2y\$10\$1KjA3Bo7tvwrxsadU AwMuubyi4vqsg0ARkDLVyEtzHN...

Check all With selected: Edit Copy Delete Export

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

Query results operations

Print Copy to clipboard Export Display chart Create view



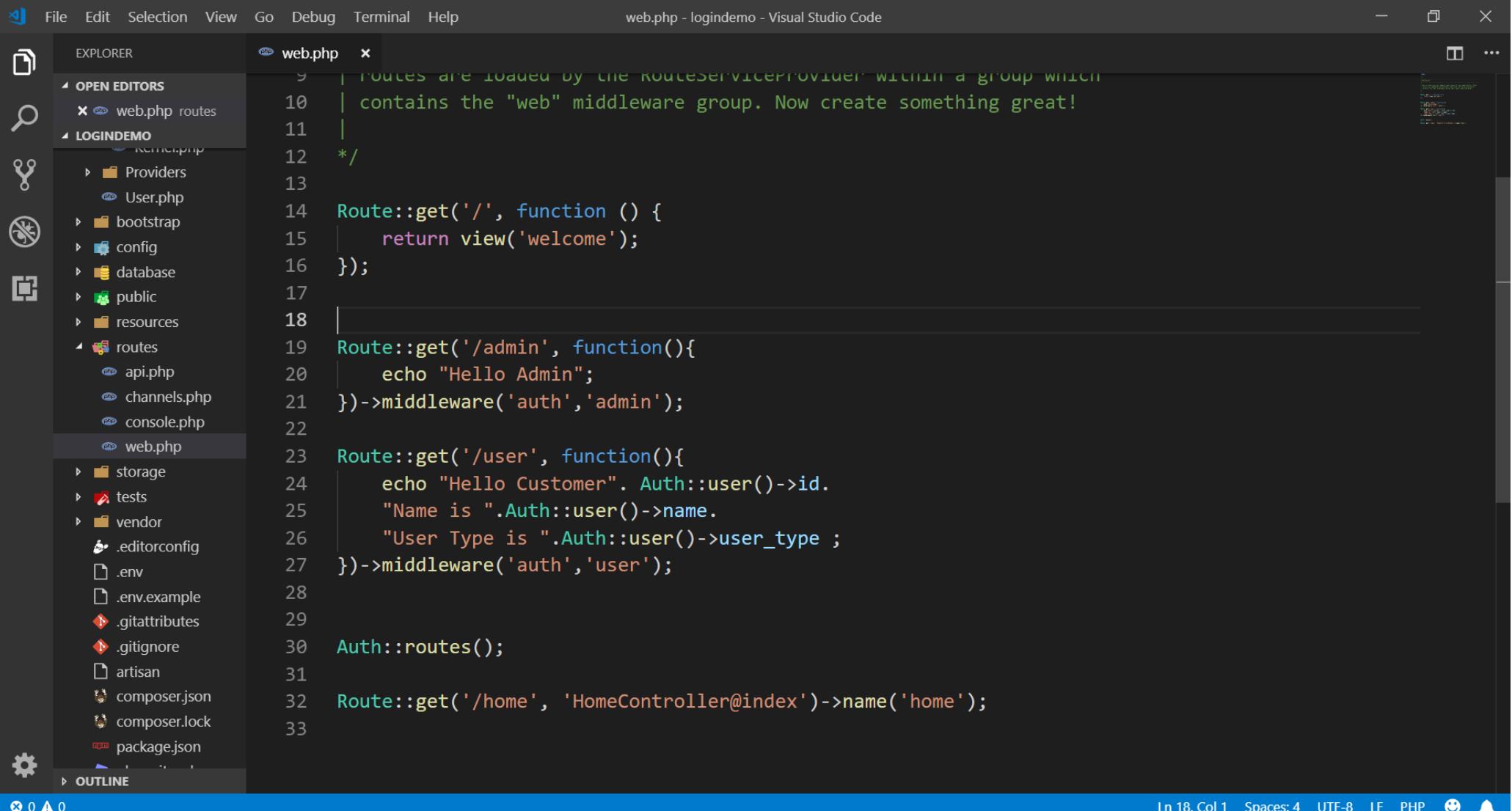


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)

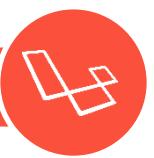


# Get Other Information



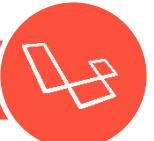
The screenshot shows a Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Title Bar:** web.php - logindemo - Visual Studio Code.
- Explorer View:** Shows the project structure:
  - OPEN EDITORS: web.php routes
  - LOGINDEMO:
    - Kernel.php
    - Providers
    - User.php
    - bootstrap
    - config
    - database
    - public
    - resources
    - routes
      - api.php
      - channels.php
      - console.php
      - web.php
    - storage
    - tests
    - vendor
    - .editorconfig
    - .env
    - .env.example
    - .gitattributes
    - .gitignore
    - artisan
    - composer.json
    - composer.lock
    - package.json
- Editor View:** The web.php file content is displayed, showing Laravel route definitions. The code includes comments explaining route loading and middleware usage.
- Bottom Status Bar:** Line 18, Col 1, Spaces: 4, UTF-8, LF, PHP, a smiley face icon.



# Snippet

- Auth::user()->id.
- Auth::user()->name.
- Auth::user()->user\_type ;



# Update for Laravel 6: *The New Way*

- composer require laravel/ui
- php artisan ui vue --auth
- <https://laravel.com/docs/6.x/authentication>



■ Enjoy ☺

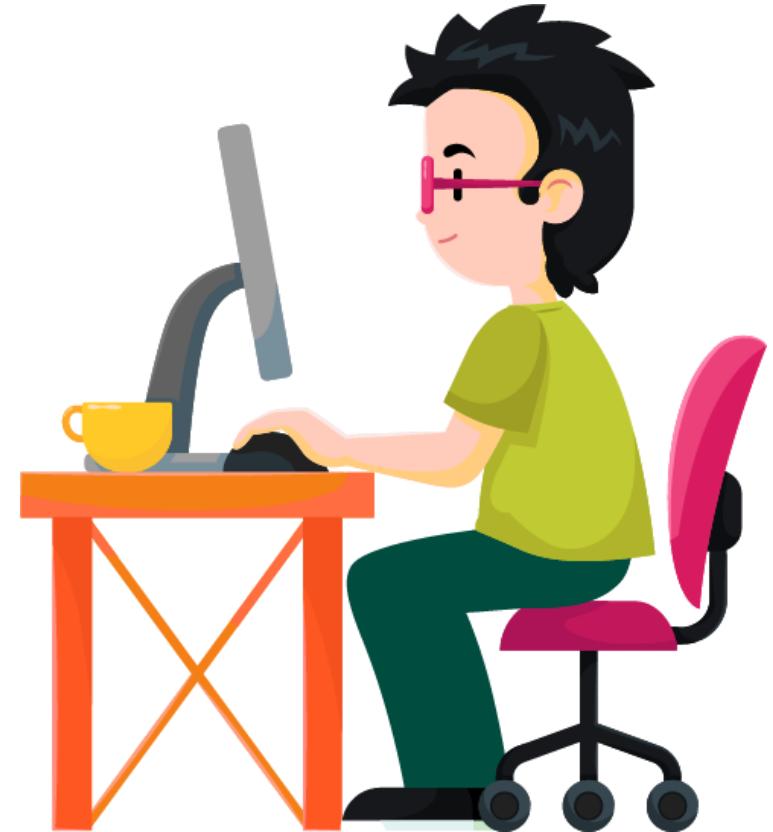


Akash Technolabs

[www.akashsir.com](http://www.akashsir.com)



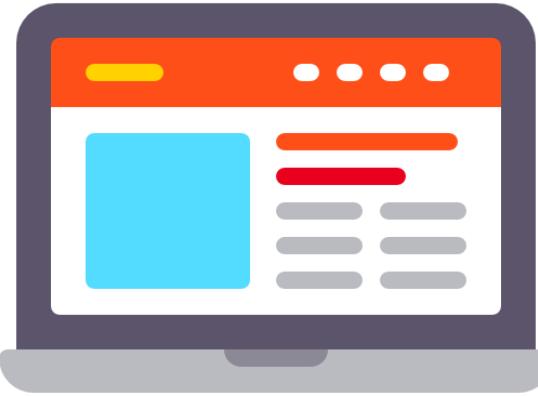
# Get Exclusive Video Tutorials



[www.aptutorials.com](http://www.aptutorials.com)

<https://www.youtube.com/user/Akashtips>





Get More Details

[www.akashsir.com](http://www.akashsir.com)



# If You Liked It !

## Rating Us Now



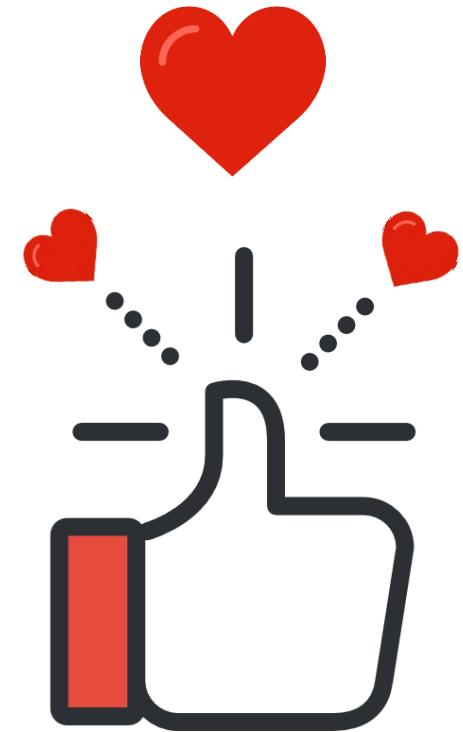
**Just Dial**

[https://www.justdial.com/Ahmedabad/Aakash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4\\_BZDET](https://www.justdial.com/Ahmedabad/Aakash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET)



**Sulekha**

<https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad>



# Connect With Me



Akash Padhiyar  
**#AkashSir**

[www.akashsir.com](http://www.akashsir.com)

[www.akashtechnolabs.com](http://www.akashtechnolabs.com)

[www.akashpadhiyar.com](http://www.akashpadhiyar.com)

[www.aptutorials.com](http://www.aptutorials.com)

## # Social Info



Akash.padhiyar



Akashpadhiyar



Akash\_padhiyar



+91 99786-21654



#Akashpadhiyar

#aptutorials