

Computer Security Project Report

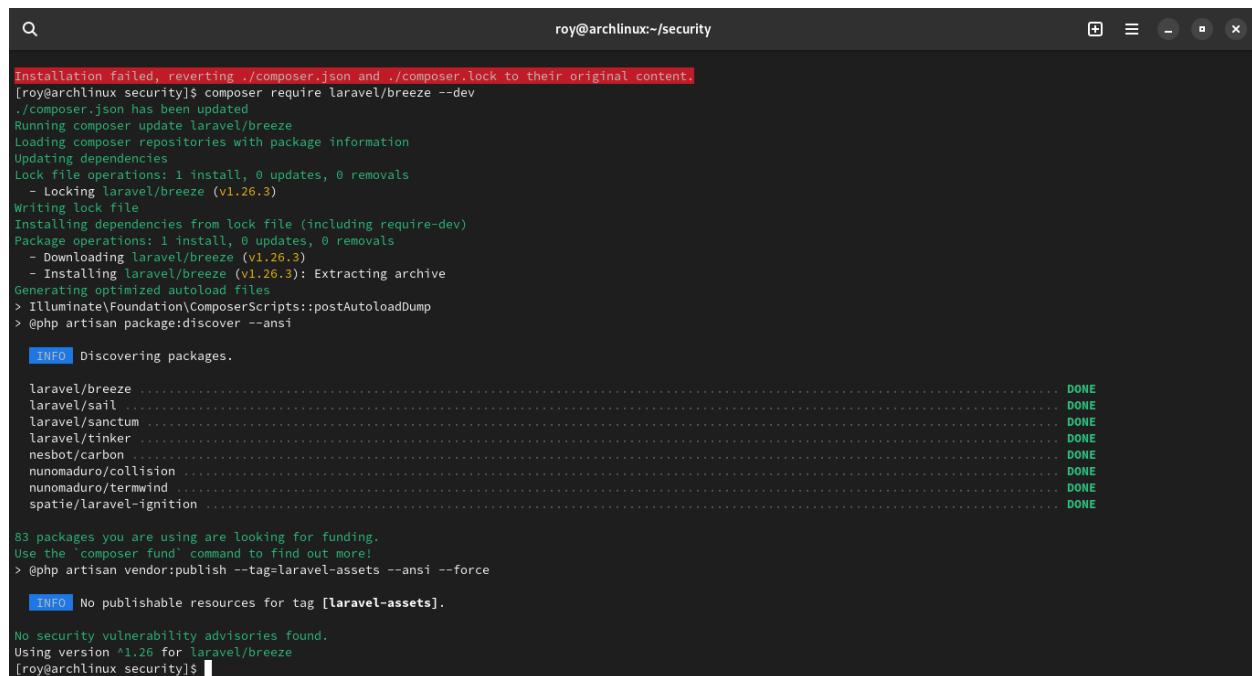
Done by

Roy Gebrayel 202111125
, Abbas Hamyeh 202111049

Overview :

- This is a project made in laravel in goal of learning the security and backend implementations and features aimed in this course
- We tried to work on laravel 10 having to deal with most of the security implementations and flows in the web app
- This is a hospital management system where patients , doctors , users and admins are found

1. User Authentication :



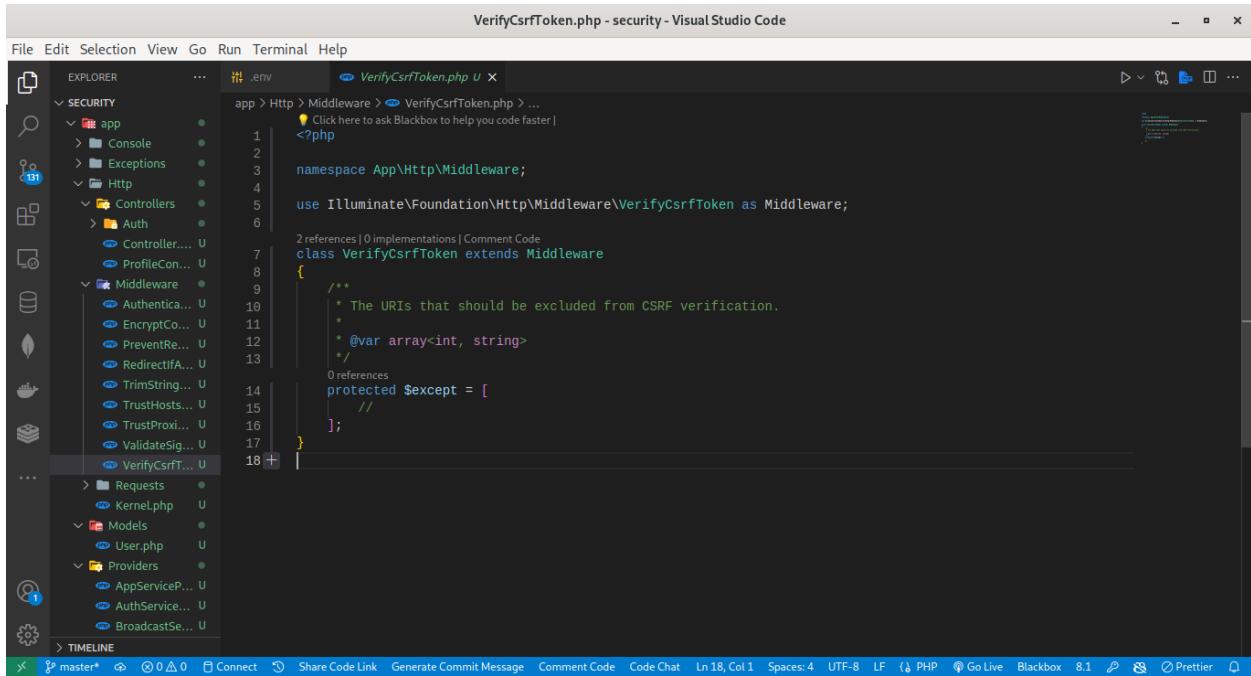
The screenshot shows a terminal window with the following text:

```
[roy@archlinux security]$ composer require laravel/breeze --dev
./composer.json has been updated
Running composer update laravel/breeze
Loading composer repositories with package information
Updating dependencies
Lock file operations: 1 install, 0 updates, 0 removals
  - Locking laravel/breeze (v1.26.3)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 1 install, 0 updates, 0 removals
  - Downloading laravel/breeze (v1.26.3)
  - Installing laravel/breeze (v1.26.3): Extracting archive
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
INFO  Discovering packages.

laravel/breeze ..... DONE
laravel/sail ..... DONE
laravel/sanctum ..... DONE
laravel/tinker ..... DONE
nesbot/carbon ..... DONE
nunomaduro/collision ..... DONE
nunomaduro/termwind ..... DONE
spatie/laravel-ignition ..... DONE

83 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force
INFO  No publishable resources for tag [laravel-assets].
No security vulnerability advisories found.
Using version ^1.26 for laravel/breeze
[roy@archlinux security]$
```

Installing laravel breeze to help us in the authentication flow

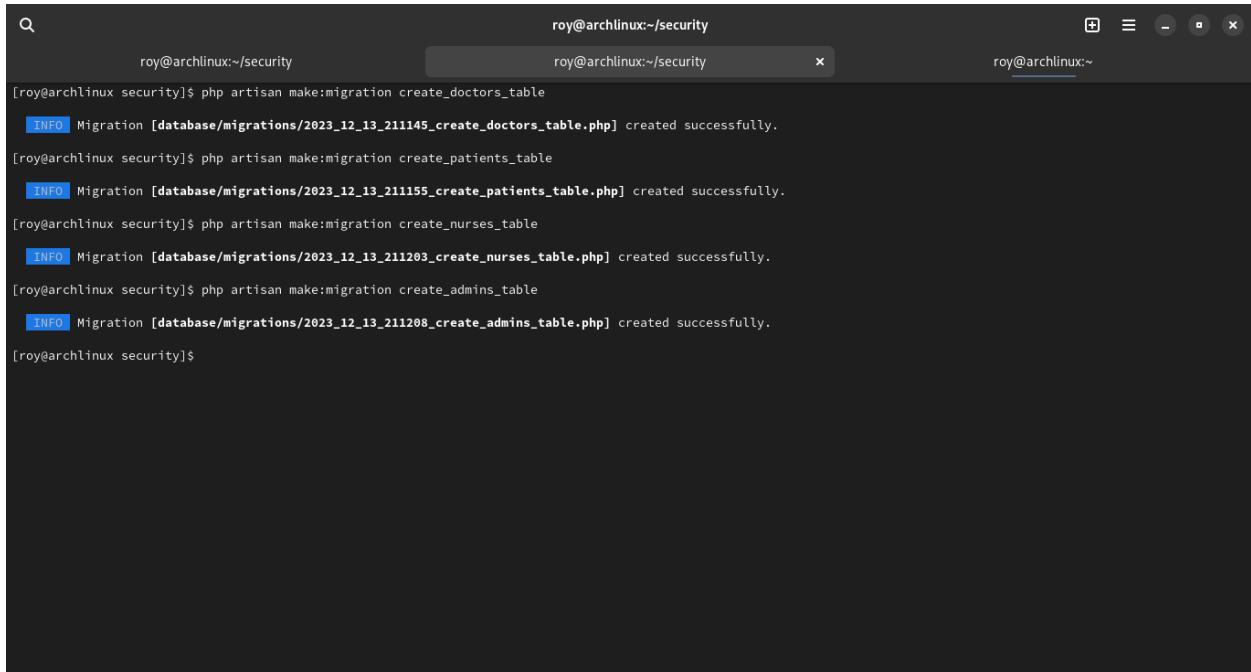


The screenshot shows the Visual Studio Code interface with the title bar "VerifyCsrfToken.php - security - Visual Studio Code". The Explorer sidebar on the left shows a project structure under "SECURITY" with folders like "app", "Console", "Exceptions", "Http", "Controllers", "Auth", "Middleware", "Models", and "Providers". The "VerifyCsrfToken.php" file is open in the main editor area. The code is as follows:

```
<?php  
namespace App\Http\Middleware;  
  
use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;  
  
class VerifyCsrfToken extends Middleware  
{  
    /**  
     * The URIs that should be excluded from CSRF verification.  
     */  
    protected $except = [  
        //  
    ];  
}
```

The status bar at the bottom shows "master" and other terminal-related information.

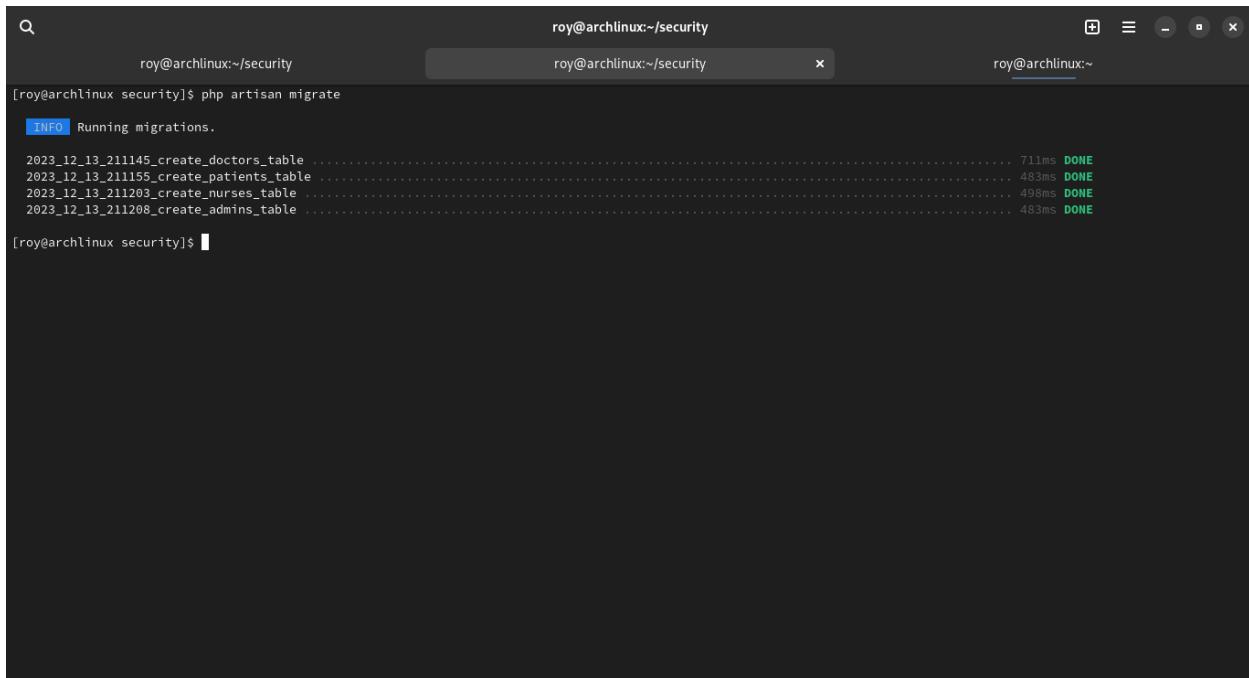
This is the folder where our auth folder is located



The screenshot shows a terminal window with the prompt "roy@archlinux:~/security". The user runs several artisan commands to create database tables:

```
[roy@archlinux security]$ php artisan make:migration create_doctors_table  
[INFO] Migration [database/migrations/2023_12_13_211145_create_doctors_table.php] created successfully.  
  
[roy@archlinux security]$ php artisan make:migration create_patients_table  
[INFO] Migration [database/migrations/2023_12_13_211155_create_patients_table.php] created successfully.  
  
[roy@archlinux security]$ php artisan make:migration create_nurses_table  
[INFO] Migration [database/migrations/2023_12_13_211203_create_nurses_table.php] created successfully.  
  
[roy@archlinux security]$ php artisan make:migration create_admins_table  
[INFO] Migration [database/migrations/2023_12_13_211208_create_admins_table.php] created successfully.
```

Creating the databases :

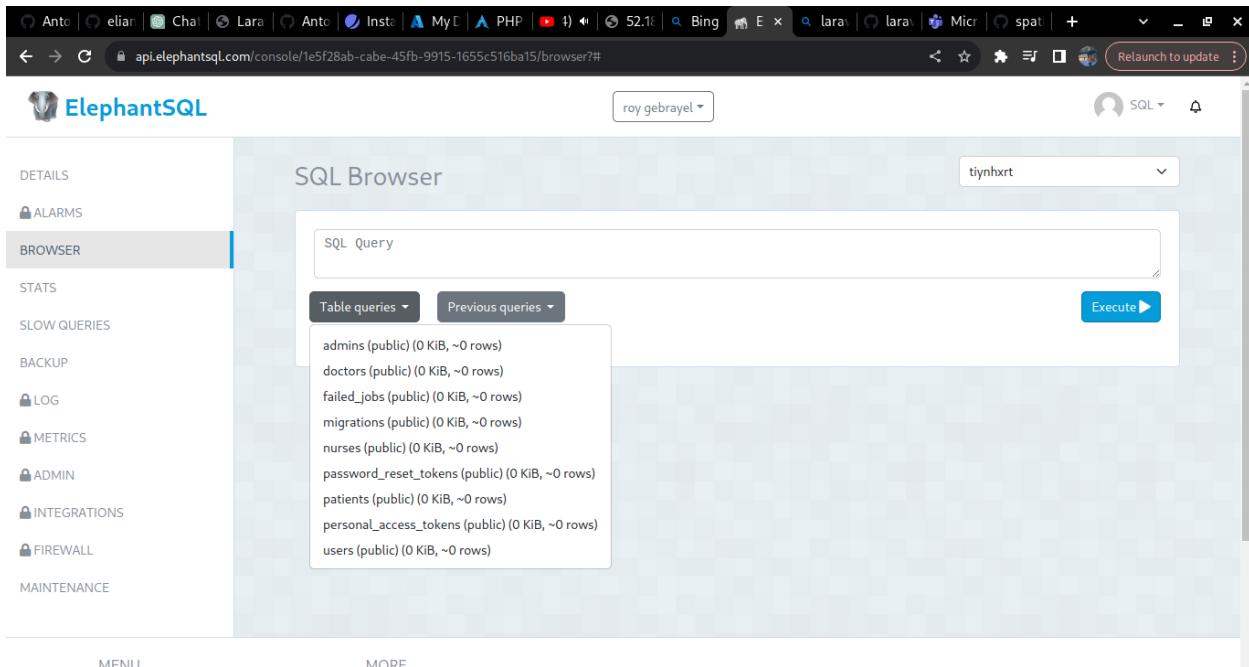


```
[roy@archlinux security]$ php artisan migrate
INFO: Running migrations.

2023_12_13_211145_create_doctors_table ..... 711ms DONE
2023_12_13_211155_create_patients_table ..... 483ms DONE
2023_12_13_211203_create_nurses_table ..... 498ms DONE
2023_12_13_211208_create_admins_table ..... 483ms DONE

[roy@archlinux security]$
```

Deploying them into the hosted BaaS postgres database with `php artisan migrate`



The screenshot shows the ElephantSQL SQL Browser interface. On the left, there's a sidebar with various monitoring and management tabs: DETAILS, ALARMS, BROWSER (which is selected), STATS, SLOW QUERIES, BACKUP, LOG, METRICS, ADMIN, INTEGRATIONS, FIREWALL, and MAINTENANCE. The main area is titled "SQL Browser" and contains a "SQL Query" input field and a dropdown menu for "Table queries" and "Previous queries". Below this, a list of tables is displayed with their sizes and row counts:

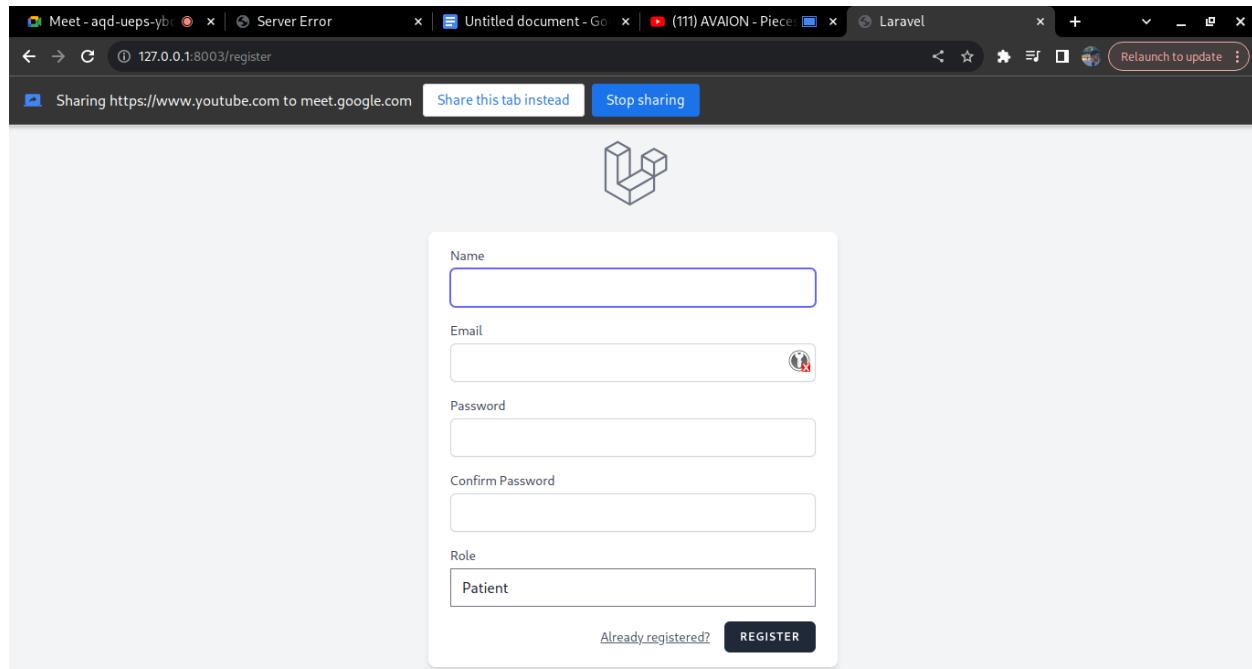
Table	Type	Size	Rows
admins	(public)	(0 KiB, ~0 rows)	
doctors	(public)	(0 KiB, ~0 rows)	
failed_jobs	(public)	(0 KiB, ~0 rows)	
migrations	(public)	(0 KiB, ~0 rows)	
nurses	(public)	(0 KiB, ~0 rows)	
password_reset_tokens	(public)	(0 KiB, ~0 rows)	
patients	(public)	(0 KiB, ~0 rows)	
personal_access_tokens	(public)	(0 KiB, ~0 rows)	
users	(public)	(0 KiB, ~0 rows)	

Those are the created databases in elephantSQL

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

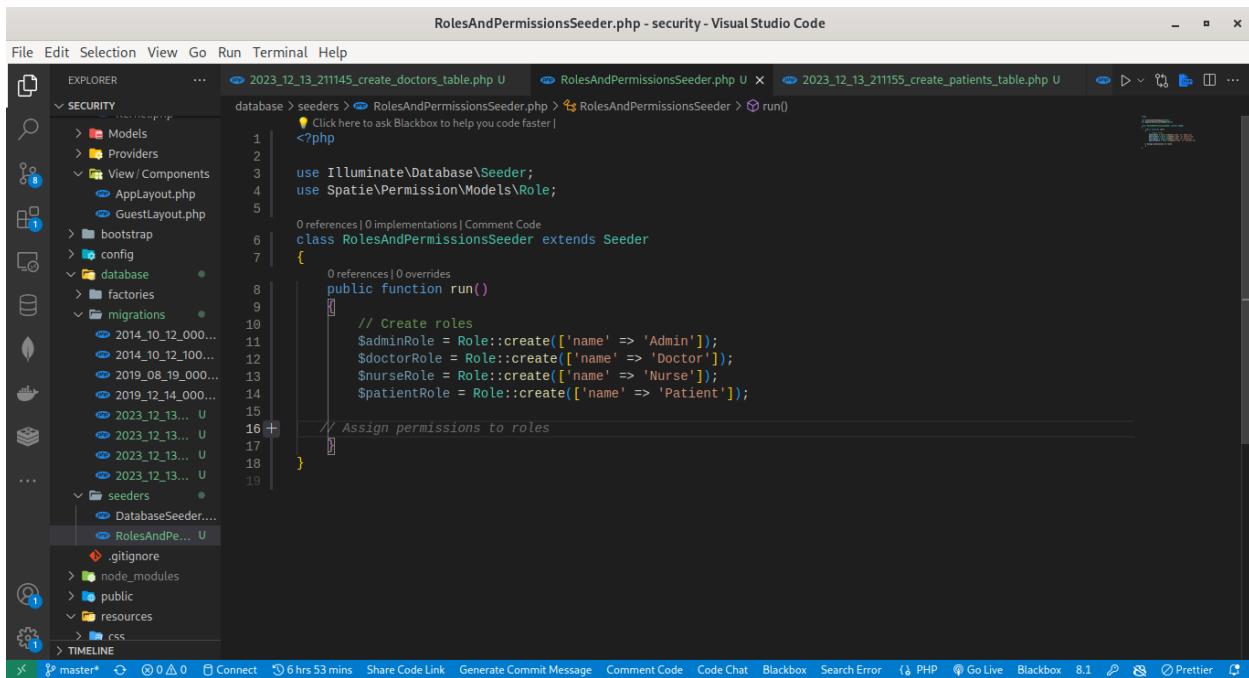
- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** web.php - security - Visual Studio Code
- Left Sidebar (Explorer):**
 - SECURITY**: dashboard.blade.php, patient, dashboard.blade.php, auth, confirm-passw..., forgot-passw..., login.blade.php, register.blade.php, reset-password..., verify-email.blade.php, components, layouts, profile, dashboard.blade.php, upload.blade.php, welcome.blade.php.
 - routes**: api.php, auth.php, channels.php, console.php, web.php (marked with a green M).
 - storage**, tests, vendor, .editorconfig.
- Code Editor:** The main editor area contains PHP code for the `web.php` file, specifically defining routes for a Laravel application. It includes middleware like `auth` and `verified`, and controllers like `AdminController` and `DoctorControllerRoute`.
- Code Suggestions:** A sidebar on the right shows suggestions for the current code being edited.
- Bottom Status Bar:** master*, 24 mins, Share Code Link, Generate Commit Message, Comment Code, Code Chat, Blackbox Searching..., Search Error, Go Live, Blackbox, 8.1, Prettier.

This is our web.php route file and this is where we inserted that all the written endpoints would need authentication in order to access it



THIS IS the registration page where all authentication and validation are done on it and when the user clicks on register all data are sent to the database “users”

2 - ACCESS CONTROL



The screenshot shows the Visual Studio Code interface with the title "RolesAndPermissionsSeeder.php - security - Visual Studio Code". The code editor displays the following PHP code:

```
<?php
use Illuminate\Database\Seeder;
use Spatie\Permission\Models\Role;

class RolesAndPermissionsSeeder extends Seeder
{
    public function run()
    {
        // Create roles
        $adminRole = Role::create(['name' => 'Admin']);
        $doctorRole = Role::create(['name' => 'Doctor']);
        $nurseRole = Role::create(['name' => 'Nurse']);
        $patientRole = Role::create(['name' => 'Patient']);

        // Assign permissions to roles
    }
}
```

The Explorer sidebar on the left shows the project structure, including files like 2023_12_13_211145_create_doctors_table.php, RolesAndPermissionsSeeder.php, and 2023_12_13_211155_create_patients_table.php.

Create a database seed to defines roles

```
<?php

use App\Models\Doctor;
use App\Models\Patient;
use App\Models\Admin;
use Illuminate\Support\Facades\Auth;
use Spatie\Permission\Models\Role;

// For Doctor
$doctor = Doctor::create($request->all());
$doctor->assignRole('doctor');

// For Admin
$admin = new Admin($request->all());
$admin->save();
$admin->assignRole('admin');

// For Patient
$patient = Patient::create($request->all());
$patient->assignRole('patient');
```

This file assigns roles for each user

```
protected function validator(array $data)
{
    return Validator::make($data, [
        // Validation rules
    ]);
}

protected function create(array $data)
{
    $user = User::create([
        // Create user
    ]);

    // Additional logic based on user role
    if ($data['role'] === 'doctor') {
        // Custom logic for doctors
        // For example, redirect to doctor-specific route
        return redirect()->route('admin.doctor.create');
    }

    // Default behavior
    return $user;
}

protected function registered(\Illuminate\Http\Request $request, $user)
{
    // Handle post-registration tasks, if needed
}
```

This file secures the route for a page if the user is a doctor (role = doctor)

3 - Data Encryption:

hashing.php - security - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER config > hashing.php

```
14 |     Supported: "bcrypt", "argon", "argon2id"
15 |
16 |
17 |
18 |     'driver' => 'bcrypt',
19 |
20 |
21 |     /**
22 |     |-----+
23 |     | Bcrypt Options
24 |     |-----+
25 |
26 |     | Here you may specify the configuration options that should be used when
27 |     | passwords are hashed using the Bcrypt algorithm. This will allow you
28 |     | to control the amount of time it takes to hash the given password.
29 |
30 |
31 |     'bcrypt' => [
32 |         'rounds' => env('BCRYPT_ROUNDS', 12),
33 |         'verify' => true,
34 |     ],
35 |
36 |
37 |     /**
38 |     |-----+
39 |     | Argon Options
40 |     |-----+
41 |
42 |     | Here you may specify the configuration options that should be used when
43 |     | passwords are hashed using the Argon algorithm. These will allow you
44 |     | to control the amount of time it takes to hash the given password.
45 |
46 |
47 |     'argon' => [
48 |         'memory' => 65536,
49 |         'threads' => 1,
50 |         'time' => 4,
51 |         'verify' => true,
52 |     ],
53 |
54 ];
55 |
```

master 0 24 mins Share Code Link Generate Commit Message Comment Code Code Chat Blackbox Search Error LF PHP Go Live Blackbox 8.1 Prettier

hashing.php - security - Visual Studio Code

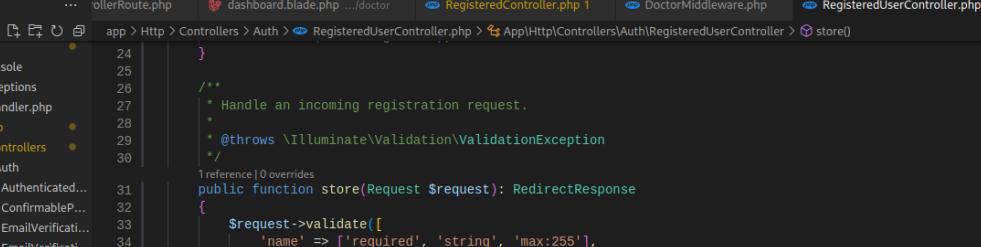
File Edit Selection View Go Run Terminal Help

EXPLORER config > hashing.php

```
36 |
37 |
38 |     /**
39 |     |-----+
40 |     | Argon Options
41 |     |-----+
42 |
43 |     | Here you may specify the configuration options that should be used when
44 |     | passwords are hashed using the Argon algorithm. These will allow you
45 |     | to control the amount of time it takes to hash the given password.
46 |
47 |
48 |     'argon' => [
49 |         'memory' => 65536,
50 |         'threads' => 1,
51 |         'time' => 4,
52 |         'verify' => true,
53 |     ],
54 |
55 |;
```

master 0 24 mins Share Code Link Generate Commit Message Comment Code Code Chat Blackbox Search Error LF PHP Go Live Blackbox 8.1 Prettier

In the config file in the hash file , we found two drivers .. argon and bcrypt (both of these are hashing algorithms to hash passwords)



```
RegisteredUserController.php - security - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER ... controllerRoute.php dashboard.blade.php .../doctor RegisteredController.php 1 DoctorMiddleware.php RegisteredUserController.php x DoctorMiddleware.php RegisteredUserController.php x store)

SECURITY app Http Controllers Auth RegisteredUserController.php x App\Http\Controllers\Auth\RegisteredUserController.php x store)

app
  - Console
  - Exceptions
    - Handler.php
  - Http
    - Controllers
      - Auth
        - Authenticated...
        - ConfirmableP...
        - EmailVerificati...
        - EmailVerificati...
        - NewPassword...
        - PasswordCont...
        - PasswordRes...
        - RegisteredUser...
        - VerifyEmailCo...
        - AdminController...
        - Controller.php
        - DoctorController...
        - DoctorsController...
        - FileController.php
        - PatientsController...
        - ProfileController...
        - RBAC.php
        - Registered... 1
      - Middleware
  - Routes
  - Services
  - Stubs
  - Tests
  - Views

  24 }
  25 }
  26 /**
  27 * Handle an incoming registration request.
  28 *
  29 * @throws \Illuminate\Validation\ValidationException
  30 */
  31 Reference|0 overrides
  32 public function store(Request $request): RedirectResponse
  33 {
  34     $request->validate([
  35         'name' => ['required', 'string', 'max:255'],
  36         'email' => ['required', 'string', 'lowercase', 'email', 'max:255', 'unique:' . User::class],
  37         'password' => ['required', 'confirmed', Rules\Password::defaults()],
  38     ]);
  39
  40     $user = User::create([
  41         'name' => $request->name,
  42         'email' => $request->email,
  43         'password' => Hash::make($request->password),
  44     ]);
  45
  46     event(new Registered($user));
  47
  48     Auth::login($user);
  49
  50     return redirect(RouteServiceProvider::HOME);
  51 }
  52 }
```

This is how we hashed the password in the user registration controller

4 - Session Management:

session.php - security - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER ... web.php session.php x .gitignore .env filesystems.php upload.blade.php FileController.php

SECURITY

app

bootstrap

config

- app.php
- auth.php
- broadcasting.php
- cache.php
- cors.php
- database.php
- filesystems.php
- hashing.php
- logging.php
- mail.php
- queue.php
- sanctum.php
- services.php
- session.php
- view.php

database

factories

migrations

- 2014_10_12_0000...
- 2014_10_12_100...
- 2019_08_19_000...
- 2019_12_14_000...
- 2023_12_13_2111...

SESSION

config > session.php

```
25 | Session Lifetime
26 |
27 | -----
28 | Here you may specify the number of minutes that you wish the session
29 | to be allowed to remain idle before it expires. If you want them
30 | to immediately expire on the browser closing, set that option.
31 |
32 /**
33 | -----
34 | Session Encryption
35 |
36 | This option allows you to easily specify that all of your session data
37 | should be encrypted before it is stored. All encryption will be run
38 | automatically by Laravel and you can use the Session like normal.
39 |
40 /**
41 | -----
42 | Session File Location
43 |
44 /**
45 | -----
46 | -----
47 | -----
48 | -----
49 | -----
50 | -----
51 /**
52 | -----
53 | -----
54 | -----
55 | -----
```

TIMELINE

The screenshot shows the Visual Studio Code interface with the title bar "session.php - security - Visual Studio Code". The left sidebar displays the project structure under "EXPLORER", including the "config" folder which contains "session.php". The main editor area shows the code for "session.php". The code handles session configuration, including setting the domain to the environment variable SESSION_DOMAIN, enabling HTTPS-only cookies, and setting session security to true. It also includes logic for HTTP access only and preventing JavaScript from accessing session values.

```
151 | Here you may change the domain of the cookie used to identify a session
152 | in your application. This will determine which domains the cookie is
153 | available to in your application. A sensible default has been set.
154 |
155 |
156 */
157
158 'domain' => env('SESSION_DOMAIN'),
159 /*
160 | -----
161 | HTTPS Only Cookies
162 | -----
163 |
164 By setting this option to true, session cookies will only be sent back
165 to the server if the browser has a HTTPS connection. This will keep
166 the cookie from being sent to you when it can't be done securely.
167
168 */
169
170 'secure' => env('SESSION_SECURE_COOKIE', true),
171 /*
172 | -----
173 | HTTP Access Only
174 | -----
175 |
176 Setting this value to true will prevent JavaScript from accessing the
177 value of the cookie and the cookie will only be accessible through
178 the HTTP protocol. You are free to modify this option if needed.
179
180
181
```

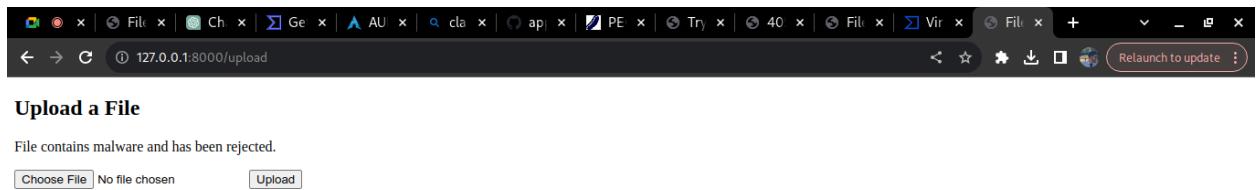
Here is the session config file where we handle all session timeout and session configurations and session token security

The screenshot shows the Visual Studio Code interface with the title bar "DoctorsController.php - security - Visual Studio Code". The left sidebar displays the project structure under "EXPLORER", including the "Http" folder which contains "Controllers" and "Auth". The main editor area shows the code for the "store" method of the "DoctorsController". The code performs validation on the request, stores data in the session, creates a new doctor record, regenerates the session ID for security, and redirects with a success message.

```
14 |
15 |
16 |
17 public function store(Request $request)
18 {
19     // Validation logic
20     $request->validate([
21         'name' => 'required|string|max:255',
22         'specialization' => 'required|string|max:255',
23         'email' => 'required|email|unique:doctors|max:255',
24         // Add more validation rules as needed
25     ]);
26     Log::info('created a doctor');
27
28     // Store data in the session
29     $request->session()->put('doctor_name', $request->input('name'));
30
31     // Create a new Doctor record
32     Doctor::create($request->all());
33     Log::info('created a doctor record.');
34
35     // Regenerate the session ID for security
36     $request->session()->regenerate();
37
38     // Redirect with success message
39     return redirect()->route('admin.dashboard')->with('success', 'Doctor added successfully.');
40 }
```

Here in the doctor controller i am storing my data in a session and then i am regenerating the session id for security purposes (this is one of the main use cases of sessions)

6. File Upload Security:



This is the upload page where you can add files into the pgsql db

FileController.php - security - Visual Studio Code

```

File Edit Selection View Go Run Terminal Help
EXPLORER upload.blade.php create.blade.php CspMiddleware.php Kernel.php dashboard.blade.php .../admin FileController.php 2023_12_14_135129_create_files_table.php php.ini File.php
app > Http > Controllers > FileController.php > App\Http\Controllers\FileController > upload()
    3 references to implementations | Comment code
    class FileController extends Controller
    {
        1 reference | overrides
        public function showUploadForm()
        {
            return view('upload');
        }

        1 reference | overrides
        public function upload(Request $request)
        {
            1 reference | overrides
            Request::validate([
                'file' => 'required|mimes:jpeg,png/pdf|max:2048',
            ]);

            $file = $request->file('file');
            $fileDetails = [
                'filename' => $file->getClientOriginalName(),
                'uploaded_at' => now(),
                // Add any other fields as needed
            ];

            // Use the File model to insert the file details into the database
            File::create($fileDetails);

            // Scan the file for malware using VirusTotal API
            $apiKey = 'b47d363baca3c7e7bc45c841d15b8ba9bca75bd6bcaace1b7f42d63078297';

            $response = http::asMultipart()
                ->withHeaders([
                    'x-apikey' => $apiKey,
                ])
                ->attach('file', file_get_contents($file->getRealPath()), $file->getClientOriginalName())
                ->post('https://www.virustotal.com/api/v3/files');

            // Ensure the response is successful before trying to decode it
            if ($response->successful()) {
                $result = $response->json();

                // Check if the response has the expected structure
                if (isset($result['data']['attributes']['last_analysis_stats']['malicious']) && $result['data']['attributes']['last_analysis_stats']['malicious'] === 0) {
                    // File is clean, proceed as usual
                    return redirect()->back()->with('message', 'File uploaded successfully.');
                } else {
                    // File is flagged by antivirus engines, handle accordingly
                    // Also, you might want to notify the user that the file contains malware.
                }
            }
        }
    }
}

```

File Controller.php

This is the fileupload controller that takes a file as input and sends it into the db
And what files could be passed to it



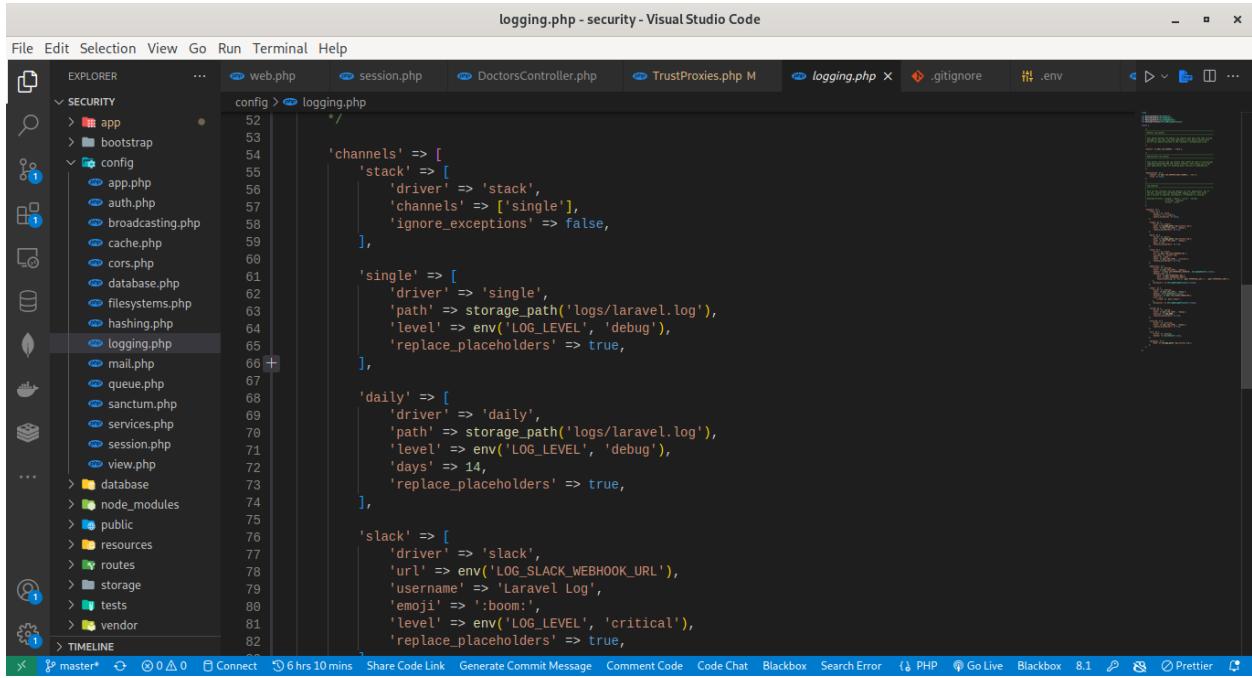
Upload a File

File contains malware and has been rejected.

No file chosen

We also inserted a third party library that tests with all the malware file testing the file uploaded and response if there is malware or not

6 - Error handling and logging



The screenshot shows the Visual Studio Code interface with the file 'logging.php' open. The code defines a configuration for logging channels:

```
/*
 * Configuration for the logging system.
 */

'channels' => [
    'stack' => [
        'driver' => 'stack',
        'channels' => ['single'],
        'ignore_exceptions' => false,
    ],
    'single' => [
        'driver' => 'single',
        'path' => storage_path('logs/laravel.log'),
        'level' => env('LOG_LEVEL', 'debug'),
        'replace_placeholders' => true,
    ],
    'daily' => [
        'driver' => 'daily',
        'path' => storage_path('logs/laravel.log'),
        'level' => env('LOG_LEVEL', 'debug'),
        'days' => 14,
        'replace_placeholders' => true,
    ],
    'slack' => [
        'driver' => 'slack',
        'url' => env('LOG_SLACK_WEBHOOK_URL'),
        'username' => 'Laravel Log',
        'emoji' => ':boom:',
        'level' => env('LOG_LEVEL', 'critical'),
        'replace_placeholders' => true,
    ],
],
```

This is the config of the logging file where it is divided into channels

DoctorsController.php - security - Visual Studio Code

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... dController.php 1 DoctorMiddleware.php RegisteredUserController.php User.php session.php DoctorsController.php > ...
SECURITY app Http Controllers > DoctorsController.php > App\Http\Controllers\DoctorsController > create()
app > Http > Controllers > DoctorsController.php > App\Http\Controllers\DoctorsController > create()
14
1 reference | 0 overrides
public function store(Request $request)
{
    // Validation logic
    $request->validate([
        'name' => 'required|string|max:255',
        'specialization' => 'required|string|max:255',
        'email' => 'required|email|unique:doctors|max:255',
        // Add more validation rules as needed
    ]);
    Log::info('created a doctor');

    // Store data in the session
    $request->session()->put('doctor_name', $request->input('name'));

    // Create a new Doctor record
    Doctor::create($request->all());
    Log::info('created a doctor record.');

    // Regenerate the session ID for security
    $request->session()->regenerate();

    // Redirect with success message
    return redirect()->route('admin.dashboard')->with('success', 'Doctor added successfully.');
}

```

master* 0 48 mins Share Code Link Generate Commit Message Comment Code Code Chat Blackbox Search Error LF PHP Go Live Blackbox 8.1 Prettier

This is the doctor controller and here we can see where we used `log::info` to send it into the logs of the app

laravel.log - security - Visual Studio Code

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... DoctorsController.php M laravel.log TrustProxies.php M .gitignore .env fileSystems.php upload.blade.php > ...
SECURITY storage > logs > laravel.log
storage > logs > laravel.log
resources
routes
api.php
auth.php
channels.php
console.php
web.php
storage
app
framework
logs
.gitignore
laravel.log
tests
vendor
.editorconfig
.env
.gitattributes
.gitignore
.phpunit.result.cache
artisan
composer.json
composer.lock
package-lock.json
package.json
phpunit.xml

```

7341 #14 /home/roy/security/vendor/laravel/framework/src/Illuminate/View/Middleware/ShareErrorsFromSession.php(49): Illuminate\View\Middleware\ShareErrorsFromSession::handle()
7342 #15 /home/roy/security/vendor/laravel/framework/src/Illuminate/Pipeline\Pipeline.php(180): Illuminate\View\Middleware\ShareErrorsFromSession::handle()
7343 #16 /home/roy/security/vendor/laravel/framework/src/Illuminate\Session\Middleware\StartSession.php(121): Illuminate\Pipeline\Pipeline::next()
7344 #17 /home/roy/security/vendor/laravel/framework/src/Illuminate/Pipeline\Pipeline.php(64): Illuminate\Session\Middleware\StartSession::handle()
7345 #18 /home/roy/security/vendor/laravel/framework/src/Illuminate/Pipeline\Pipeline.php(180): Illuminate\Session\Middleware\StartSession::handle()
7346 #19 /home/roy/security/vendor/laravel/framework/src/Illuminate/Cookie\Middleware/AddQueuedCookiesToResponse.php(71): Illuminate\Pipeline\Pipeline::next()
7347 #20 /home/roy/security/vendor/laravel/framework/src/Illuminate/Pipeline\Pipeline.php(180): Illuminate\Cookie\Middleware\EncryptCookies::handle()
7348 #21 /home/roy/security/vendor/laravel/framework/src/Illuminate/Cookie\Middleware\EncryptCookies.php(67): Illuminate\Pipeline\Pipeline::next()
7349 #22 /home/roy/security/vendor/laravel/framework/src/Illuminate/Pipeline\Pipeline.php(180): Illuminate\Cookie\Middleware\EncryptCookies::handle()
7350 #23 /home/roy/security/vendor/laravel/framework/src/Illuminate/Pipeline\Pipeline.php(116): Illuminate\Pipeline\Pipeline::next()
7351 #24 /home/roy/security/vendor/laravel/framework/src/Illuminate\Routing\Router.php(805): Illuminate\Pipeline\Pipeline::next()
7352 #25 /home/roy/security/vendor/laravel/framework/src/Illuminate\Routing\Router.php(784): Illuminate\Routing\Router::match()
7353 #26 /home/roy/security/vendor/laravel/framework/src/Illuminate\Routing\Router.php(748): Illuminate\Routing\Router::findRoute()
7354 #27 /home/roy/security/vendor/laravel/framework/src/Illuminate\Routing\Router.php(737): Illuminate\Routing\Router::findRoute()
7355 #28 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Kernel.php(200): Illuminate\Routing\Router::findRoute()
7356 #29 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(141): Illuminate\Foundation\Http\Kernel::handle()
7357 #30 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Middleware\TransformsRequest.php(27): Illuminate\Pipeline\Pipeline::next()
7358 #31 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Middleware\ConvertEmptyStringToNull.php(27): Illuminate\Foundation\Http\Middleware\TransformsRequest::handle()
7359 #32 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(180): Illuminate\Foundation\Http\Middleware\ConvertEmptyStringToNull::handle()
7360 #33 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Middleware\ValidatePostSize.php(46): Illuminate\Pipeline\Pipeline::next()
7361 #34 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Middleware\TrimStrings.php(46): Illuminate\Foundation\Http\Middleware\ValidatePostSize::handle()
7362 #35 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(180): Illuminate\Foundation\Http\Middleware\TrimStrings::handle()
7363 #36 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Middleware\ValidatePostSize.php(46): Illuminate\Pipeline\Pipeline::next()
7364 #37 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(180): Illuminate\Foundation\Http\Middleware\ValidatePostSize::handle()
7365 #38 /home/roy/security/vendor/laravel/framework/src/Illuminate\Foundation\Http\Middleware\PreventRequestsDuringMaintenance.php(27): Illuminate\Pipeline\Pipeline::next()
7366 #39 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(180): Illuminate\Foundation\Http\Middleware\PreventRequestsDuringMaintenance::handle()
7367 #40 /home/roy/security/vendor/laravel/framework/src/Illuminate\Http\Middleware\HandleCors.php(49): Illuminate\Pipeline\Pipeline::next()
7368 #41 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(180): Illuminate\Http\Middleware\HandleCors::handle()
7369 #42 /home/roy/security/vendor/laravel/framework/src/Illuminate\Http\Middleware\TrustProxies.php(39): Illuminate\Pipeline\Pipeline::next()
7370 #43 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(180): Illuminate\Http\Middleware\TrustProxies::handle()
7371 #44 /home/roy/security/vendor/laravel/framework/src/Illuminate\Pipeline\Pipeline.php(116): Illuminate\Pipeline::next()

master* 0 6 hrs 10 mins Share Code Link Generate Commit Message Comment Code Code Chat Blackbox Search Error LF Log PHP Go Live Blackbox 8.1 Prettier

Here are the logs of the app

```
auth.php - security - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER ... .iredUserController.php User.php session.php DoctorsController.php TrustProxies.php .gitignore auth.php > routes ... auth.php
SECURITY
routes > auth.php
admin
doctor
patient
auth
components
layouts
profile
routes
api.php
auth.php
channels.php
console.php
web.php
storage
TIMELINE
auth.php - security - Visual Studio Code
auth.php
42 Route::post('email/verification-notification', [EmailVerificationNotificationController::class, 'store'])
| ->middleware('throttle:6,1')
| ->name('verification.send');
43
44
45
46 Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])
| ->name('password.confirm');
47
48 Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);
49
50 Route::put('password', [PasswordController::class, 'update'])
| ->name('password.update');
51
52
53 Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])
| ->name('logout');
54
55
56 // Handling 404 for undefined routes
57 Route::fallback(function () {
58     abort(404, 'Page Not Found');
59 });
60
61 // Handling 500 for server errors
62 Route::fallback(function () {
63     abort(500, 'Server Error');
64 });
65
66
42 Route::post('email/verification-notification', [EmailVerificationNotificationController::class, 'store'])
| ->middleware('throttle:6,1')
| ->name('verification.send');
43
44
45
46 Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])
| ->name('password.confirm');
47
48 Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);
49
50 Route::put('password', [PasswordController::class, 'update'])
| ->name('password.update');
51
52
53 Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])
| ->name('logout');
54
55
56 // Handling 404 for undefined routes
57 Route::fallback(function () {
58     abort(404, 'Page Not Found');
59 });
60
61 // Handling 500 for server errors
62 Route::fallback(function () {
63     abort(500, 'Server Error');
64 });
65
66
```

Here is the auth.php , and this is where we handles the errors that we might get come into .. code status ; 404 , 500

7 - sql injection prevention and penetration testing

```

    /**
     * Handle an incoming registration request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'lowercase', 'email', 'max:255', 'unique:' . User::class],
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
        ]);

        $user = User::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password),
        ]);

        event(new Registered($user));

        Auth::login($user);

        return redirect(RouteServiceProvider::HOME);
    }
}

```

Here we inserted eloquent orm and validation to prevent all sql injection hacks

```

roy@archlinux:~/security
[21:48:57] [INFO] testing 'MySQL < 5.0.12 time-based blind - ORDER BY, GROUP BY clause (BENCHMARK)'
[21:48:57] [INFO] testing 'PostgreSQL > 8.1 time-based blind - ORDER BY, GROUP BY clause'
[21:48:58] [INFO] testing 'PostgreSQL time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[21:48:58] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind - ORDER BY clause (heavy query)'
[21:48:58] [INFO] testing 'Oracle time-based blind - ORDER BY, GROUP BY clause (DBMS_LOCK.SLEEP)'
[21:48:58] [INFO] testing 'Oracle time-based blind - ORDER BY, GROUP BY clause (DBMS_PIPE.RECEIVE_MESSAGE)'
[21:48:58] [INFO] testing 'Oracle time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[21:48:58] [INFO] testing 'HSQLDB > 1.7.2 time-based blind - ORDER BY, GROUP BY clause (heavy query)'
[21:48:58] [INFO] testing 'HSQLDB > 2.0 time-based blind - ORDER BY, GROUP BY clause (heavy query)'

it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n]

[21:49:06] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[21:49:10] [INFO] testing 'Generic UNION query (random number) - 1 to 10 columns'
[21:49:14] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[21:49:19] [INFO] testing 'MySQL UNION query (random number) - 1 to 10 columns'
[21:49:24] [WARNING] POST parameter 'username' does not seem to be injectable
[21:49:24] [INFO] testing if POST parameter 'password' is dynamic
[21:49:24] [WARNING] POST parameter 'password' does not appear to be dynamic
[21:49:24] [WARNING] heuristic (basic) test shows that POST parameter 'password' might not be injectable
[21:49:24] [INFO] testing for SQL injection on POST parameter 'password'
[21:49:24] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[21:49:28] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
[21:49:33] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT)'
[21:49:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[21:49:39] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[21:49:42] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (comment)'
[21:49:43] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (comment)'
[21:49:44] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - comment)'
[21:49:44] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[21:49:46] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[21:49:48] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[21:49:49] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (Microsoft Access comment)'
[21:49:51] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (Microsoft Access comment)'
[21:49:53] [INFO] testing 'MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause'
[21:49:55] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'

```

Testing our app using sqlmap , a tool made for testing sql injection

```
roy@archlinux:~
```

```
roy@archlinux:~/security
```

```
[roy@archlinux ~]$ sqlmap -u "http://127.0.0.1:8000/login" --data="username=test&password=test" --risk=3 --level=5
```

```
---  
--- [H] --- {1.7.10#stable}  
--- . [D] | . | . |  
--- | [I] . [I] . , . |  
--- | . V... | . | https://sqlmap.org
```

```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
```

```
[*] starting @ 21:52:05 /2023-12-14/
```

```
[21:52:05] [INFO] testing connection to the target URL  
[21:52:05] [WARNING] the web server responded with an HTTP error code (419) which could interfere with the results of the tests  
you have not declared cookie(s), while server wants to set its own ('laravel_session=eyJpdiI6In...IjoiIn0%3D'). Do you want to use those [Y/n]
```

```
[21:52:07] [INFO] testing if the target URL content is stable  
[21:52:07] [INFO] target URL content is stable  
[21:52:07] [INFO] testing if POST parameter 'username' is dynamic  
[21:52:07] [WARNING] POST parameter 'username' does not appear to be dynamic  
[21:52:07] [INFO] heuristic (basic) test shows that POST parameter 'username' might not be injectable  
[21:52:07] [INFO] testing for SQL injection on POST parameter 'username'  
[21:52:07] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'  
[21:52:10] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'  
[21:52:14] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT)'  
[21:52:16] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'  
[21:52:18] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (subquery - comment)'
```

Here we gave him username = test & password = test

```
roy@archlinux:~
```

```
roy@archlinux:~/security
```

```
[21:55:46] [INFO] testing connection to the target URL  
you have not declared cookie(s), while server wants to set its own ('XSRF-TOKEN=eyJpdiI6ImN...IjoiIn0%3D;laravel_session=eyJpdiI6Ik...IjoiIn0%3D'). Do you want to use those [Y/n] Y  
[21:55:47] [INFO] testing if the target URL content is stable  
[21:55:47] [INFO] target URL content is stable  
other non-custom parameters found. Do you want to process them too? [Y/n/q] Y  
[21:55:47] [INFO] testing if URI parameter '#1#' is dynamic  
[21:55:48] [WARNING] URI parameter '#1#' does not appear to be dynamic  
[21:55:48] [INFO] heuristic (basic) test shows that URI parameter '#1#' might not be injectable  
[21:55:48] [INFO] testing for SQL injection on URI parameter '#1#'  
[21:55:48] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'  
[21:55:49] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'  
[21:55:49] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'  
[21:55:49] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'  
[21:55:49] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'  
[21:55:49] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'  
[21:55:50] [INFO] testing 'Generic inline queries'  
[21:55:50] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'  
[21:55:50] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'  
[21:55:50] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'  
[21:55:50] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'  
[21:55:51] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'  
[21:55:51] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'  
[21:55:51] [INFO] testing 'Oracle AND time-based blind'  
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] Y  
[21:55:51] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'  
[21:55:51] [WARNING] URI parameter '#1#' does not seem to be injectable  
[21:55:51] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'  
[21:55:51] [WARNING] HTTP error codes detected during run:  
500 (Internal Server Error) - 73 times
```

```
[*] ending @ 21:55:51 /2023-12-14/
```

```
[roy@archlinux ~]$
```

After finishing the testing it gives us that no parameters could be injectable

```
[roy@archlinux ~]$ sqlmap -u http://127.0.0.1:8000/login --batch -D tiyhxrt --tables
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 21:59:49 /2023-12-14

[21:59:49] [WARNING] you've provided target URL without any GET parameters (e.g. 'http://www.site.com/article.php?id=1') and without providing any POST parameters through option '--data'
do you want to try URI injections in the target URL itself? [Y/n/q] Y
[21:59:49] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('XSRF-TOKEN=eyJpdiI6Im5...IjoIn0%3D;laravel_session=eyJpdiI6Ijd...IjoIn0%3D'). Do you want to use t
hose [Y/n] Y
[21:59:50] [INFO] testing if the target URL content is stable
[21:59:50] [INFO] target URL content is stable
other non-custom parameters found. Do you want to process them too? [Y/n/q] Y
[21:59:50] [INFO] testing if URI parameter '#1#' is dynamic
[21:59:50] [WARNING] URI parameter '#1#' does not appear to be dynamic
[21:59:50] [INFO] heuristic (basic) test shows that URI parameter '#1#' might not be injectable
[21:59:50] [INFO] testing for SQL injection on URI parameter '#1#'
[21:59:50] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[21:59:51] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[21:59:51] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[21:59:52] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[21:59:52] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[21:59:52] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[21:59:52] [INFO] testing 'Generic inline queries'
[21:59:52] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[21:59:53] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[21:59:53] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
```

Here trying to get tables from the database using this query

```
[roy@archlinux ~]$ sqlmap -u http://127.0.0.1:8000/login --batch -T users --columns
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 22:02:26 /2023-12-14

[22:02:26] [WARNING] you've provided target URL without any GET parameters (e.g. 'http://www.site.com/article.php?id=1') and without providing any POST parameters through option '--data'
do you want to try URI injections in the target URL itself? [Y/n/q] Y
[22:02:26] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('XSRF-TOKEN=eyJpdiI6InE...IjoIn0%3D;laravel_session=eyJpdiI6Ilp...IjoIn0%3D'). Do you want to use t
hose [Y/n] Y
[22:02:26] [INFO] testing if the target URL content is stable
[22:02:27] [INFO] target URL content is stable
other non-custom parameters found. Do you want to process them too? [Y/n/q] Y
[22:02:27] [INFO] testing if URI parameter '#1#' is dynamic
[22:02:27] [WARNING] URI parameter '#1#' does not appear to be dynamic
[22:02:27] [WARNING] heuristic (basic) test shows that URI parameter '#1#' might not be injectable
[22:02:27] [INFO] testing for SQL injection on URI parameter '#1#'
[22:02:27] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[22:02:28] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[22:02:28] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[22:02:28] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[22:02:29] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[22:02:29] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[22:02:29] [INFO] testing 'Generic inline queries'
[22:02:29] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[22:02:29] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[22:02:30] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[22:02:30] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
```

Here trying to get users from the database users table

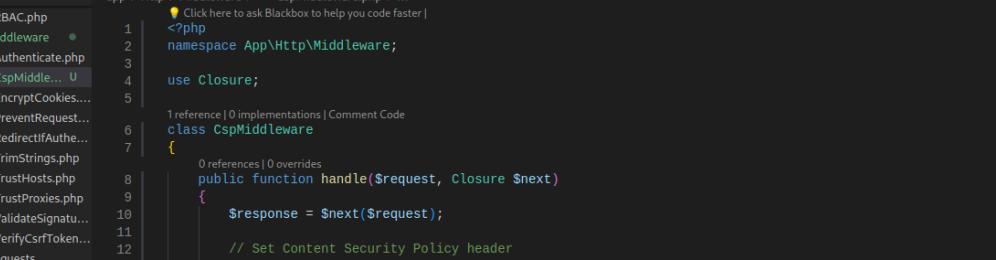
(none of these test was successfully and this means that our sql injection prevention was totally accepted)

```
[roy@archlinux ~]$ XSpear -u http://127.0.0.1:8000/ -v 1
) (
( /(\ )\ )
)\)(O)(O)/ ( ( ) ) (
((_)\\ /(_))` ) )\ \ / ( )(
((_)_) /(( _ /(_))(_))((_)\
\ \ // __|((_)\\(_)) ((_)_ ((_
> < _\_\| _- \)/ _- / _- | | _- |
/_\_-|---/_|-/_/ \_\_\|\_--|_|_-
/|_|
\|_|
[ v1.4.1 ]
[+] analysis request..
[+] used test-reflected-params mode(default)
[+] creating a test query [for reflected @ param]
[+] test query generation is complete. [32 query]
[+] starting XSS Scanning. [10 threads]
[########################################] [32/32] [100.00%] [00:01] [00:00] [23.06/s]
[+] finish scan. the report is being generated..
+-----+
| [ XSpear report ] |
| http://127.0.0.1:8000/ |
| 2023-12-14 22:39:34 +0200 ~ 2023-12-14 22:39:35 +0200 Found 4 issues. |
+-----+
| NO | TYPE | ISSUE | METHOD | PARAM | PAYLOAD | DESCRIPTION |
+-----+
| 0 | INFO | STATIC ANALYSIS | GET | - | <original query> | Not set HSTS |
| 1 | INFO | STATIC ANALYSIS | GET | - | <original query> | Content-Type: text/html; charset=UTF-8 |
| 2 | LOW | STATIC ANALYSIS | GET | - | <original query> | Not Set X-Frame-Options |
| 3 | MEDIUM | STATIC ANALYSIS | GET | - | <original query> | Not Set CSP |
+-----+
< Available Objects >
Not found

< Raw Query >
Not found
[roy@archlinux ~]$
```

Installing another tool called XSpear that scans you the whole project and gives you a whole report about all warnings and security vulnerabilities

.. in this report we found out that we didn't setups CSP and that is a medium warning we have



```
app > Http > Middleware > CspMiddleware.php > ...
Click here to ask Blackbox to help you code faster!
<?php
namespace App\Http\Middleware;

use Closure;

class CspMiddleware
{
    public function handle($request, Closure $next)
    {
        $response = $next($request);

        // Set Content Security Policy header
        $response->header(
            'Content-Security-Policy',
            'default-src \'self\'; script-src \'self\' https://cdn.example.com'
        );

        return $response;
    }
}
```

Here we created a middleware to implement the CSP

```

Kernel.php - security - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER app > Http > Kernel.php > App\Http\Kernel > middleware
  SECURITY
    +-- Middleware
      +-- RBAC.php
      +-- Authenticate.php
      +-- CspMiddle... U
      +-- EncryptCookies...
      +-- PreventRequest...
      +-- RedirectIfAuth...
      +-- TrimStrings.php
      +-- TrustHosts.php
      +-- TrustProxies.php
      +-- ValidateSignatu...
      +-- VerifyCsrfToken...
    +-- Requests
      +-- Kernel.php M
        +-- Models
          +-- Admin.php
          +-- Doctor.php
          +-- File.php
          +-- Patient.php
          +-- User.php
        +-- Providers
        +-- View
      +-- bootstrap
        +-- config
          +-- app.php
          +-- auth.php
    +-- TIMELINE
  .env filesystems.php upload.blade.php create.blade.php M CspMiddleware.php U Kernel.php M
  D v 🌐 📁 ...
```

The application's global HTTP middleware stack.

These middleware are run during every request to your application.

`protected $middleware = [`

- `// \App\Http\Middleware\TrustHosts::class,`
- `\App\Http\Middleware\TrustProxies::class,`
- `\Illuminate\Http\Middleware\HandleCors::class,`
- `\App\Http\Middleware\PreventRequestsDuringMaintenance::class,`
- `\Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,`
- `\App\Http\Middleware\TrimStrings::class,`
- `\Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,`
- `\App\Http\Middleware\CspMiddleware::class,`

The application's route middleware groups.

`protected $middlewareGroups = [`

- `'web' => [`
- `\App\Http\Middleware\EncryptCookies::class,`
- `\Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,`
- `\Illuminate\Session\Middleware\StartSession::class,`

We added the middleware into the kernel.php

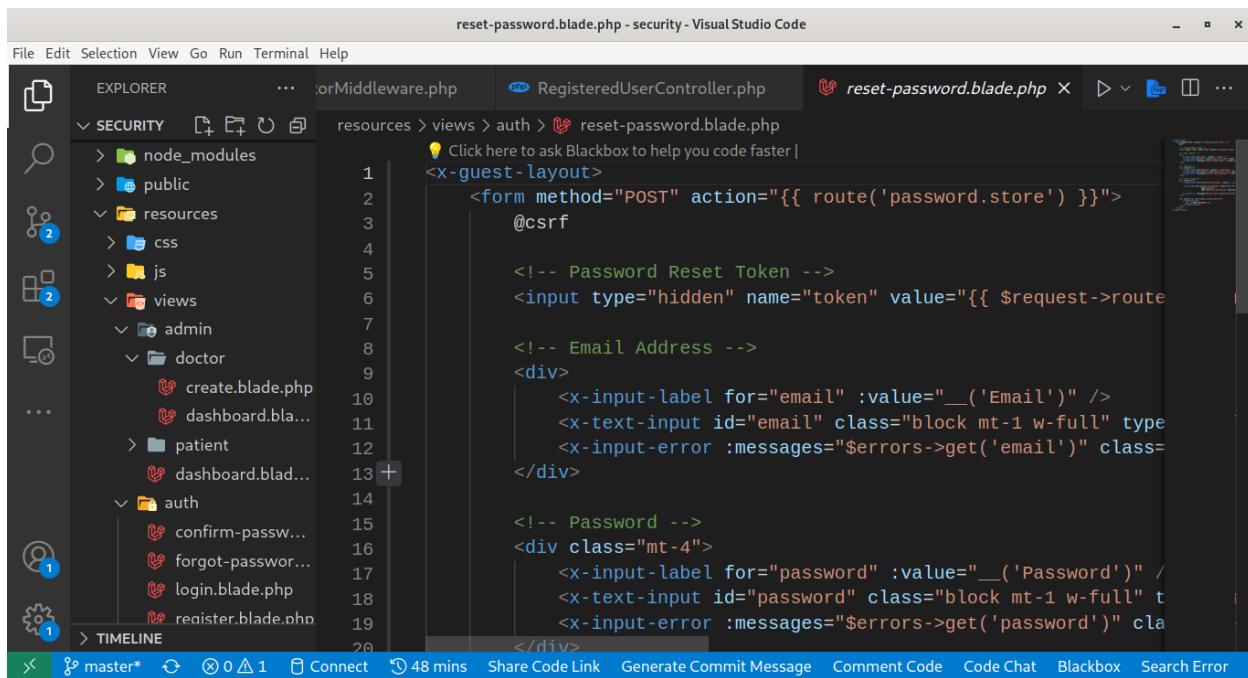
```

roy@archlinux:~/security
[roy@archlinux security]$ XSpear -u http://127.0.0.1:8000/ -v 1
[+] analysis request..
[+] used test-reflected-params mode(default)
[+] creating a test query [for reflected 0 param ]
[+] test query generation is complete. [32 query]
[+] starting XSS Scanning. [10 threads]
[########################################] [32/32] [100.00%] [00:01] [00:00] [22.39/s]
[+] finish scan. the report is being generated.
+-----+
| [ XSpear report ] |
| http://127.0.0.1:8000/ |
| 2023-12-14 22:50:48 +0200 ~ 2023-12-14 22:50:49 +0200 Found 4 issues. |
+-----+
| NO | TYPE | ISSUE | METHOD | PARAM | PAYLOAD | DESCRIPTION |
+-----+
| 0 | INFO | STATIC ANALYSIS | GET | - | <original query> | Not set HSTS |
| 1 | INFO | STATIC ANALYSIS | GET | - | <original query> | Content-Type: text/html; charset=UTF-8 |
| 2 | LOW | STATIC ANALYSIS | GET | - | <original query> | Not Set X-Frame-Options |
| 3 | INFO | STATIC ANALYSIS | GET | - | <original query> | Enabled CSP |
+-----+
< Available Objects >
Not found

< Raw Query >
Not found
[roy@archlinux security]$
```

Rerunning the XSpear testing tool and found out that no critical warning are found and that we fixed the CSP vulnerability

Additional setups :



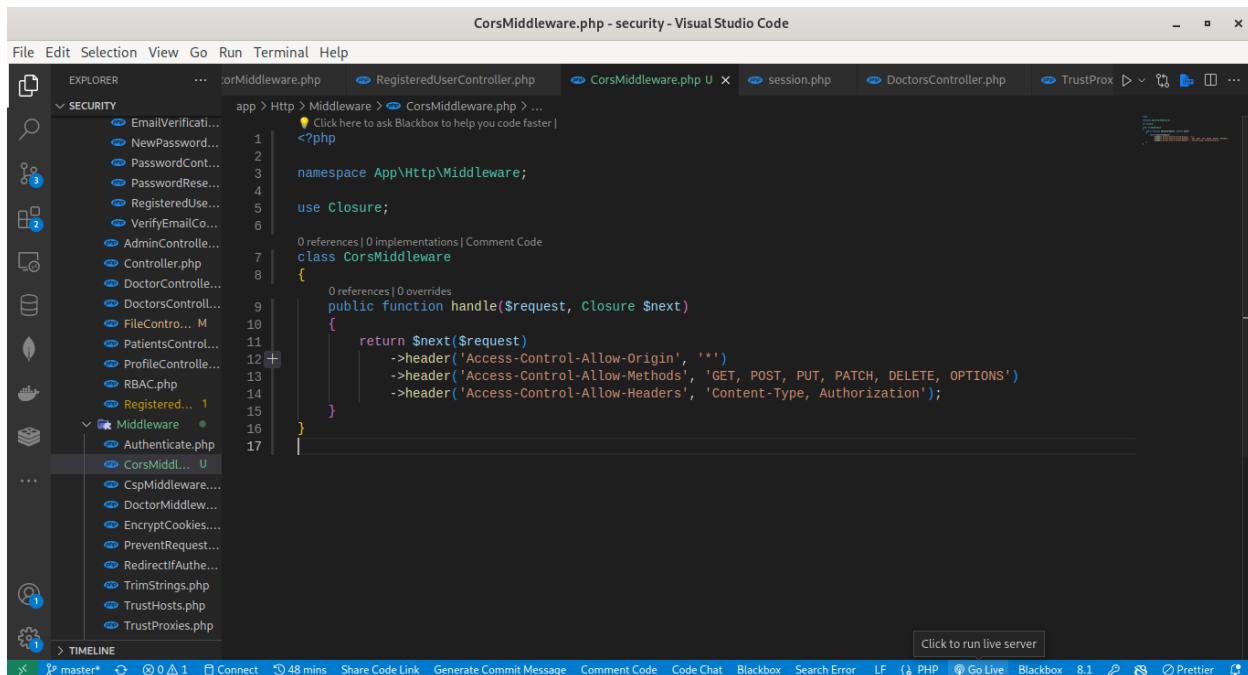
```
<x-guest-layout>
    <form method="POST" action="{{ route('password.store') }}>
        @csrf

        <!-- Password Reset Token -->
        <input type="hidden" name="token" value="{{ $request->route

        <!-- Email Address -->
        <div>
            <x-input-label for="email" :value="__('Email')"/>
            <x-text-input id="email" class="block mt-1 w-full" type="text" />
            <x-input-error :messages="$errors->get('email')" class="mt-2" />
        </div>

        <!-- Password -->
        <div class="mt-4">
            <x-input-label for="password" :value="__('Password')"/>
            <x-text-input id="password" class="block mt-1 w-full" type="password" />
            <x-input-error :messages="$errors->get('password')" class="mt-2" />
        </div>
```

In every form it is necessary to implement the CSRF to prevent all csrf attacks



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

use Closure;

class CorsMiddleware
{
    public function handle($request, Closure $next)
    {
        return $next($request)
            ->header('Access-Control-Allow-Origin', '*')
            ->header('Access-Control-Allow-Methods', 'GET, POST, PUT, PATCH, DELETE, OPTIONS')
            ->header('Access-Control-Allow-Headers', 'Content-Type, Authorization');
    }
}
```

Creating a new middleware to enable CORS on laravel

Kernel.php - security - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORERh.php .env filesystems.php upload.blade.php create.blade.php CspMiddleware.php Kernel.php M ▶ 🔍 📂

SECURITY app > Http > Kernel.php > AppHttpKernel > middleware

```
* @var array<int, class-string|string>
*/
0 references
protected $middleware = [
    // \App\Http\Middleware\TrustHosts::class,
    \App\Http\Middleware\TrustProxies::class,
    \Illuminate\Http\Middleware\HandleCors::class,
    \App\Http\Middleware\PreventRequestsDuringMaintenance::class,
    \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,
    \App\Http\Middleware\TrimStrings::class,
    \Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,
    \App\Http\Middleware\CspMiddleware::class,
    \App\Http\Middleware\CorrelationId::class,
];
/**
 * The application's route middleware groups.
 *
 * @var array<string, array<int, class-string|string>>
*/
0 references
protected $middlewareGroups = [
    'web' => [
        \App\Http\Middleware\EncryptCookies::class,
        \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
        \Illuminate\Session\Middleware\StartSession::class,
        \Illuminate\View\Middleware\ShareErrorsFromSession::class,
        \App\Http\Middleware\VerifyCsrfToken::class,
        \Illuminate\Routing\Middleware\SubstituteBindings::class,
    ],
],
lan... => [
```

Added the cors middleware into the kernel provider

And this how we implemented cors in laravel

File Edit Selection View Go Run Terminal Help

EXPLORER dashboard.blade.php .../doctor RegisteredController.php 1 DoctorMiddleware.php RegisteredUserController.php CorsMiddleware

```
<h2>Welcome, Dr. {{ Auth::user()->name }}</h2>
<!-- Display relevant information for the doctor -->
<div class="card mt-4">
    <div class="card-header">
        Upcoming Appointments
    </div>
    <div class="card-body">
        <!-- Display a list of upcoming appointments -->
        <ul>
            <li>Appointment 1 - Date and Time</li>
            <li>Appointment 2 - Date and Time</li>
        <!-- Add more appointments as needed -->
        </ul>
    </div>
</div>

<div class="card mt-4">
    <div class="card-header">
        Patient Information
    </div>
    <div class="card-body">
        <!-- Display relevant patient information or link to patient records -->
        <p>You have X patients under your care.</p>
        <a href="{{ route('patients.index') }}">View Patient Records</a>
    </div>
</div>

<!-- Add more sections as needed -->
```

Writing {{ }} is automatic escaping and that's to prevent XSS attacks also we can use {!! !!} for output that should not be escaped.

Final Review

- This project helped us a lot with understanding the security implementations in web apps
- This project helped us with knowing how to test our app and prevent it from all vulnerabilities using many tools and logics
- This projects helped us learn laravel as a new backend framework
- This is our github project : [SecurityProject](#) , we would be happy for any contribution

This project was created in 13/12/2023 and submitted in 15/12/2023