



DIRECCIÓN ACADÉMICA
VICERRECTORADO ACADÉMICO

Facultad de Ingeniería

Carrera de Ingeniería en Tecnologías de la
Información

Informe de Actividad de Investigación Formativa

Periodo Académico
2024-1S



Contenido

1.	Autores	3
2.	Personal Académico	3
3.	Resultados de Aprendizaje de la asignatura:	3
4.	Tema de la Actividad de la Investigación Formativa:	3
5.	Objetivos de las actividades:	3
6.	Fecha de la ejecución:	3
7.	Desarrollo del Informe	4
7.1	Introducción.	4
7.2	Descripción de la metodología	5
7.3	Descripción de las acciones realizadas:	7
7.4	Resultados	39
7.5	Bibliografía	41
8.	ANEXO	42



1. Autores

- Luis Sebastian Shagñay Ruiz

2. PERSONAL ACADÉMICO

- Director de Carrera: Jorge Delgado
- Profesor de Asignatura: Ing. Milton Paul Lopez

3. RESULTADOS DE APRENDIZAJE DE LA ASIGNATURA:

En la asignatura "Interoperabilidad de plataformas", el estudiante ha aprendido a integrar sistemas de software utilizando estándares y protocolos de comunicación. Ha diseñado arquitecturas interoperables, desarrollado y consumido APIs y servicios web, y asegurado la coherencia y seguridad de los datos entre sistemas. Además, ha mejorado sus habilidades de comunicación y trabajo en equipo para colaborar en proyectos multidisciplinarios.

4. TEMA DE LA ACTIVIDAD DE LA INVESTIGACIÓN FORMATIVA:

Desarrollo de una Aplicación Web Interoperable utilizando el Framework Laravel

5. OBJETIVOS DE LAS ACTIVIDADES:

- Instalar correctamente las herramientas Visual Studio Code, Xampp y Composer.
- Descargar e instalar el framework Laravel para su correcta utilización en el desarrollo de la interfaz.

6. FECHA DE LA EJECUCIÓN:

- 12 de junio de 2024



7. DESARROLLO DEL INFORME

7.1 Introducción.

En la era actual del desarrollo web, la creación de aplicaciones interactivas y eficientes es fundamental para satisfacer las demandas del mercado. Para lograrlo, se requiere el uso efectivo de tecnologías modernas que faciliten tanto la implementación como la experiencia del usuario.

Este proyecto se centra en el desarrollo de un formulario web utilizando una combinación de tecnologías ampliamente adoptadas en la industria. La aplicación del conocimiento en Laravel, Node.js, Bootstrap, XAMPP y Visual Studio Code nos permitirá construir un formulario web dinámico y atractivo, mientras aprovechamos las fortalezas de cada tecnología.

Utilizaremos Laravel, un potente framework de PHP, para manejar la lógica del servidor y facilitar la creación y manipulación de datos. Node.js será utilizado para gestionar las dependencias del proyecto y automatizar tareas relacionadas con el desarrollo web, garantizando así una experiencia de desarrollo fluida y eficiente.

Bootstrap, un framework de diseño front-end, nos brindará una amplia gama de componentes y estilos predefinidos para mejorar la apariencia y la usabilidad del formulario, asegurando una experiencia de usuario intuitiva y agradable. Además, aprovecharemos XAMPP para crear un entorno de desarrollo local que simule un servidor en vivo, lo que nos permitirá probar y depurar la aplicación de manera efectiva antes de su implementación.

Todo el proceso de desarrollo se llevará a cabo en Visual Studio Code, un editor de código fuente altamente funcional que ofrece características avanzadas para la escritura de código, la depuración y la colaboración en proyectos de desarrollo de software.



Al finalizar este proyecto, no solo habremos desarrollado un formulario web completamente funcional, sino que también habremos adquirido un profundo conocimiento sobre el uso y la integración efectiva de estas tecnologías modernas en el desarrollo de aplicaciones web.

7.2 Descripción de la metodología

Para el desarrollo de la aplicación web interoperable utilizando el framework Laravel, se seguirá una metodología ágil, que permitirá una gestión eficiente del proyecto y una adaptación flexible a los cambios. A continuación, se detallan las fases principales de la metodología:

Planificación y Requisitos:

Recolección de Requisitos: Identificación y documentación de los requisitos funcionales y no funcionales de la aplicación web. Reuniones con los stakeholders para entender sus necesidades y expectativas.

Definición de Objetivos: Establecimiento de objetivos claros y medibles para el proyecto, asegurando que todos los miembros del equipo entiendan el propósito y alcance del trabajo.

Diseño de la Arquitectura:

Especificación de la Arquitectura: Diseño de la arquitectura de la aplicación, incluyendo la estructura de la base de datos, la organización del código y la integración de servicios externos.

Elección de Tecnologías: Selección de las tecnologías y herramientas necesarias, incluyendo Laravel para el desarrollo backend, Blade para las vistas y otros componentes esenciales.



Desarrollo Iterativo:

Sprints de Desarrollo: Implementación de un enfoque basado en sprints cortos y manejables, donde cada sprint abarca un conjunto de funcionalidades específicas. Se realizarán reuniones diarias de seguimiento (daily stand-ups) para asegurar el progreso continuo y la identificación temprana de obstáculos.

Desarrollo de Funcionalidades: Codificación de las funcionalidades de la aplicación según los requisitos definidos, utilizando las mejores prácticas de programación y asegurando la interoperabilidad con otros sistemas.

Integración y Pruebas:

Pruebas Unitarias y Funcionales: Desarrollo y ejecución de pruebas unitarias y funcionales para asegurar que cada componente de la aplicación funcione correctamente y cumpla con los requisitos establecidos.

Pruebas de Integración: Verificación de que todos los componentes de la aplicación interactúen correctamente entre sí y con sistemas externos. Uso de herramientas de automatización de pruebas para mejorar la eficiencia y cobertura.

Despliegue y Documentación:

Despliegue en Entornos de Prueba: Implementación de la aplicación en entornos de prueba para realizar pruebas de aceptación con los usuarios finales y obtener feedback.

Despliegue en Producción: Una vez validadas las pruebas, despliegue de la aplicación en el entorno de producción siguiendo un plan de despliegue cuidadosamente planificado para minimizar riesgos.

Documentación: Elaboración de la documentación técnica y de usuario, incluyendo guías de instalación, configuración y uso de la aplicación.

Mantenimiento y Mejora Continua:

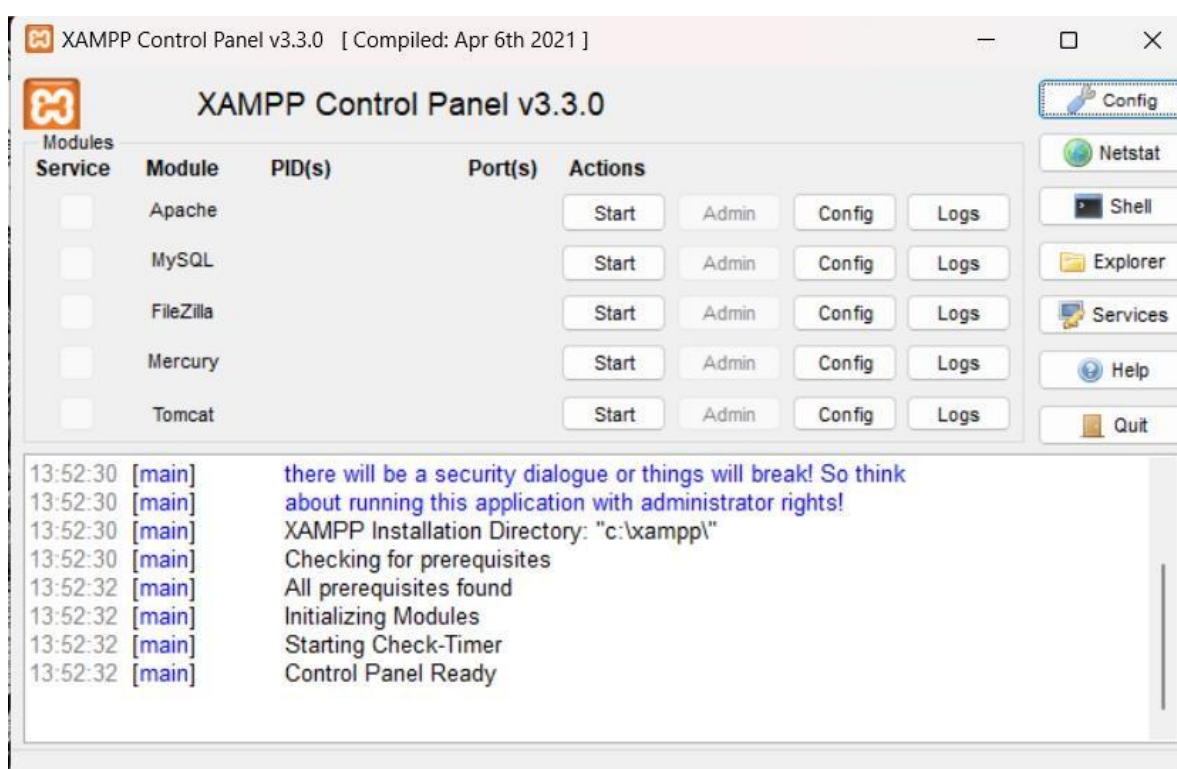
Monitoreo y Soporte: Monitoreo continuo de la aplicación en producción para detectar y resolver problemas rápidamente. Provisión de soporte técnico a los usuarios.

Mejora Continua: Recopilación de feedback de los usuarios y análisis de métricas de rendimiento para identificar áreas de mejora y planificar futuras actualizaciones y mejoras de la aplicación.

Esta metodología ágil asegura que el desarrollo de la aplicación sea eficiente, flexible y orientado a satisfacer las necesidades de los usuarios, garantizando una solución robusta y de alta calidad

7.3 Descripción de las acciones realizadas:

Para la correcta instalación de Laravel necesitamos de los siguientes programas.
Xampp





Node.js

```
Create an HTTP Server Write Tests Read and Hash a File Streams Pipeline Work with T
```

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain' });
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with `node server.mjs`
```

JavaScript [Copy to clipboard](#)

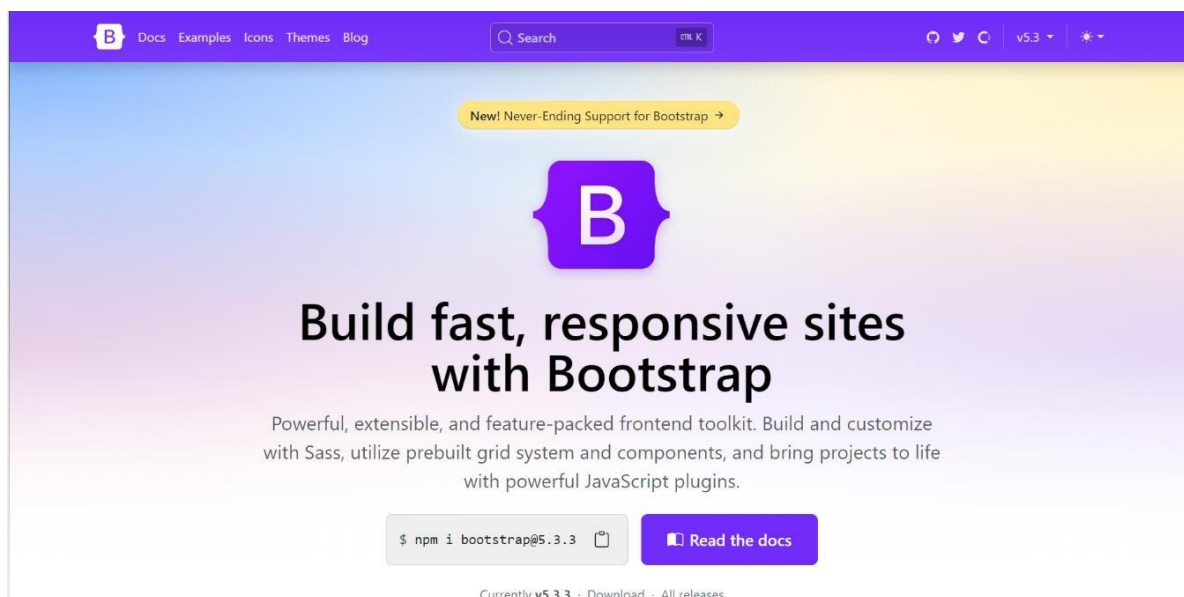
Visual Studio Code

```
Create an HTTP Server Write Tests Read and Hash a File Streams Pipeline Work with T
```

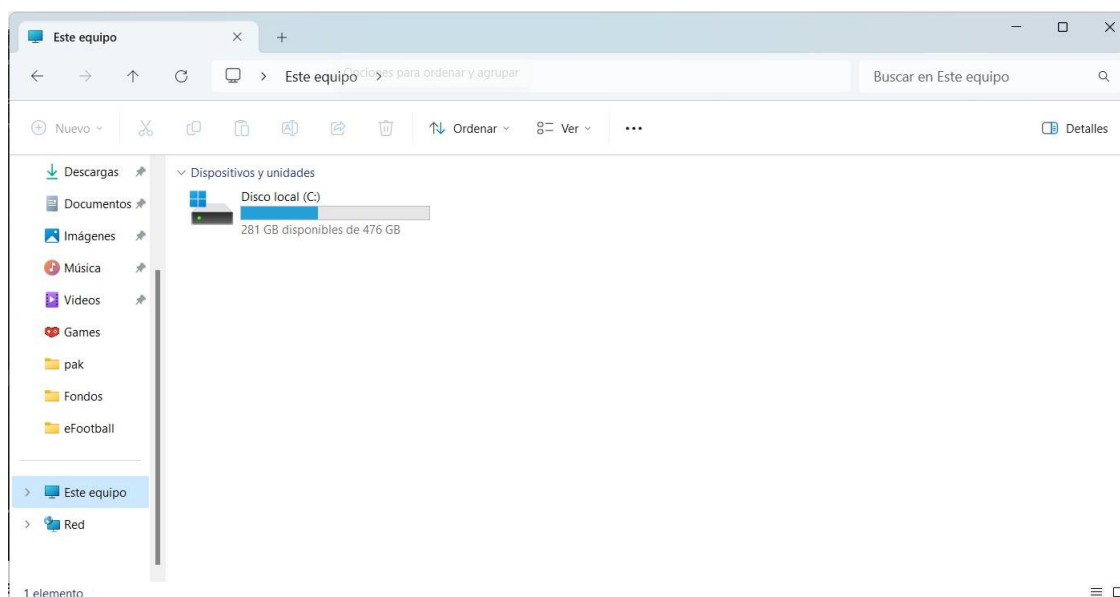
```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain' });
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with `node server.mjs`
```

JavaScript [Copy to clipboard](#)

Bootstrap

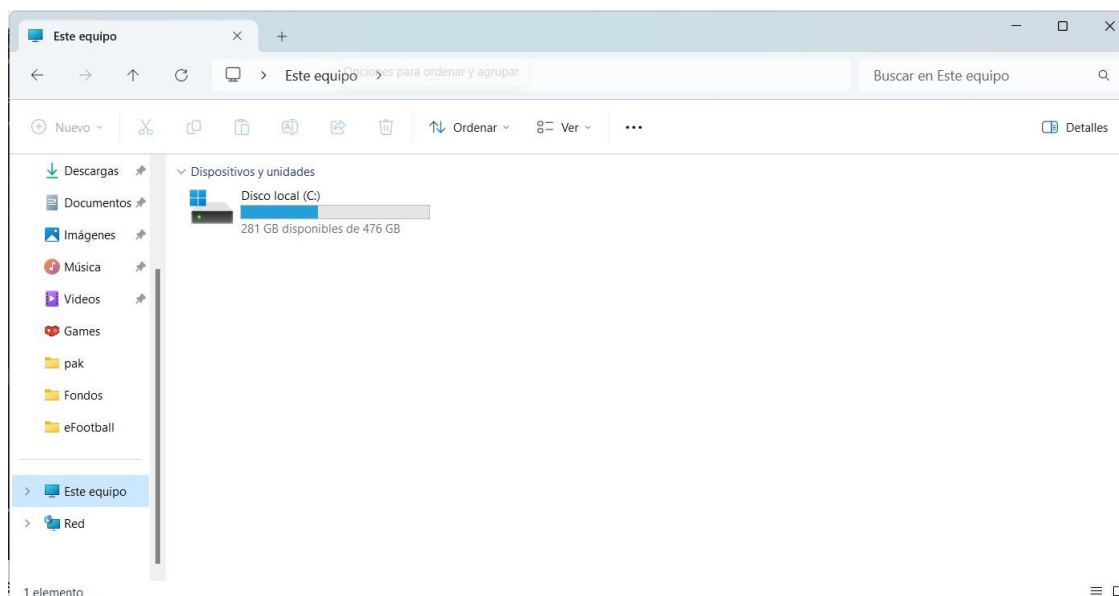


Para empezar con la instalación de Laravel primero nos dirigimos al disco local C para entrar la carpeta nombra como xamp

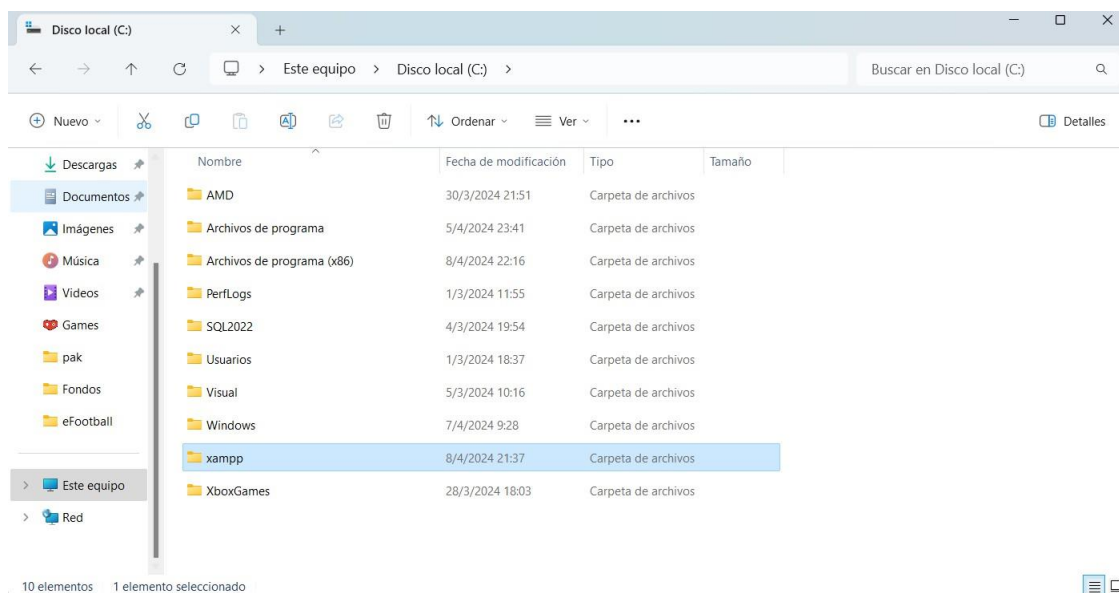




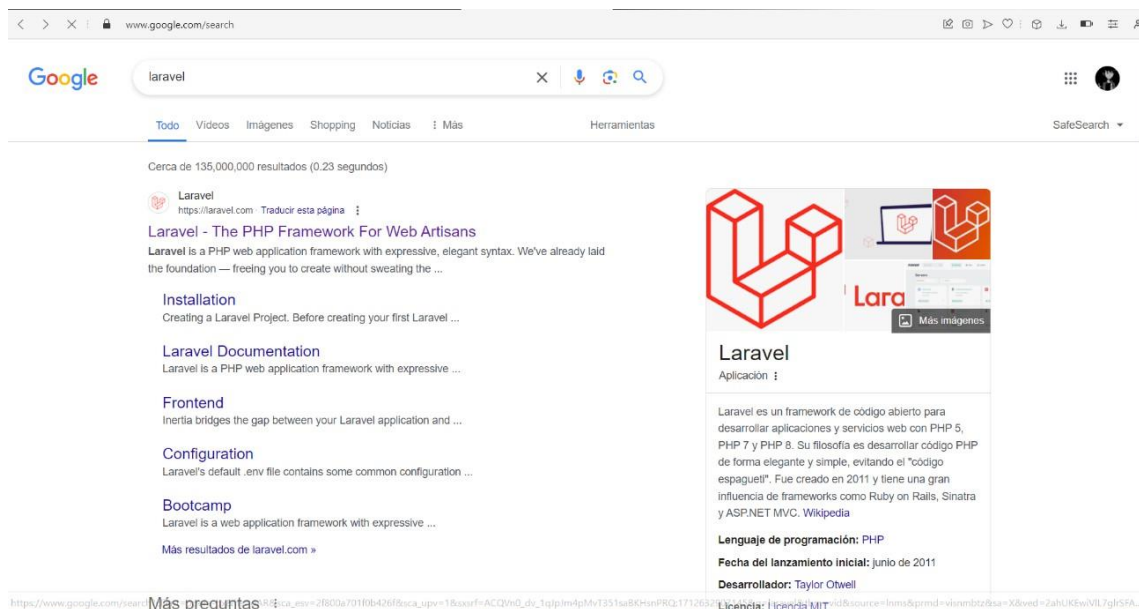
Ingresamos a la carpeta de xampp



Una vez ubicada la ruta de htdocs, nos dirigimos a Google para empezar con la instalación de Laravel

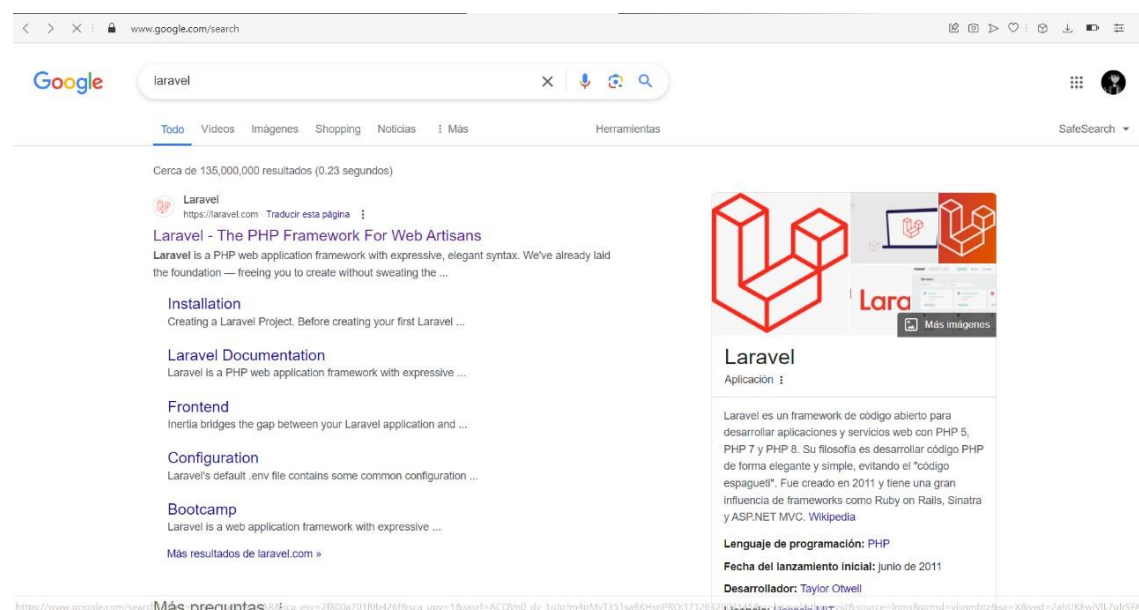


Buscamos en Google Laravel



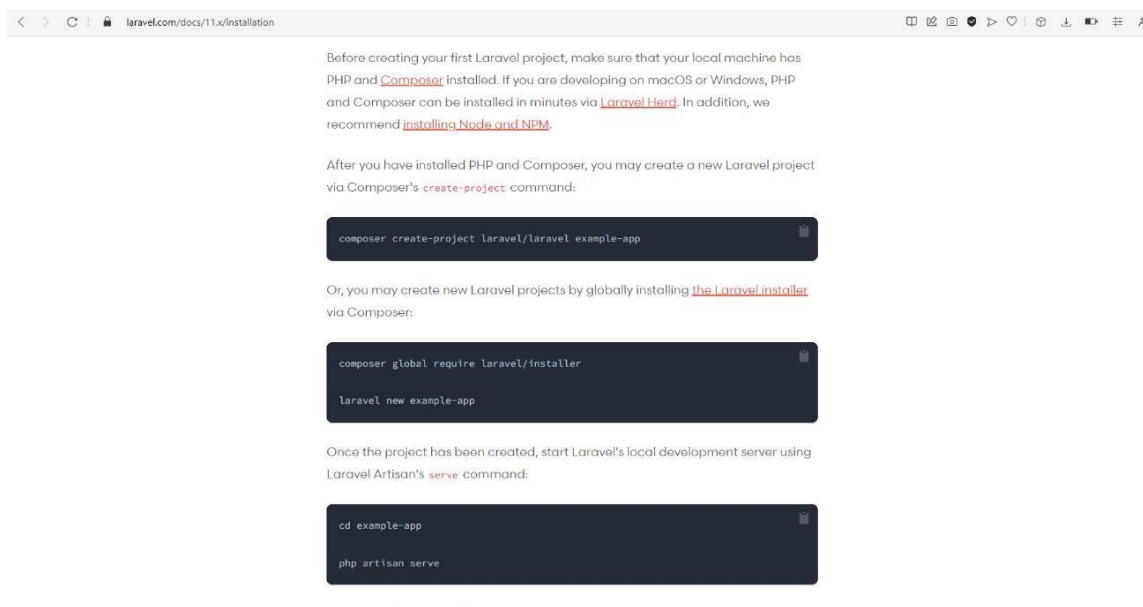
The screenshot shows a Google search for 'laravel'. The search bar contains 'laravel' and the results show 'Cerca de 135,000,000 resultados (0.23 segundos)'. The first result is 'Laravel - The PHP Framework For Web Artisans' with a link to 'https://laravel.com'. Below the title, it says 'Laravel is a PHP web application framework with expressive, elegant syntax. We've already laid the foundation — freeing you to create without sweating the ...'. There are links for 'Installation', 'Laravel Documentation', 'Frontend', 'Configuration', and 'Bootcamp'. On the right, there is a large image of the Laravel logo and a snippet of the framework's description.

Entramos en la página principal del framework para instalarlo por medio del cmd

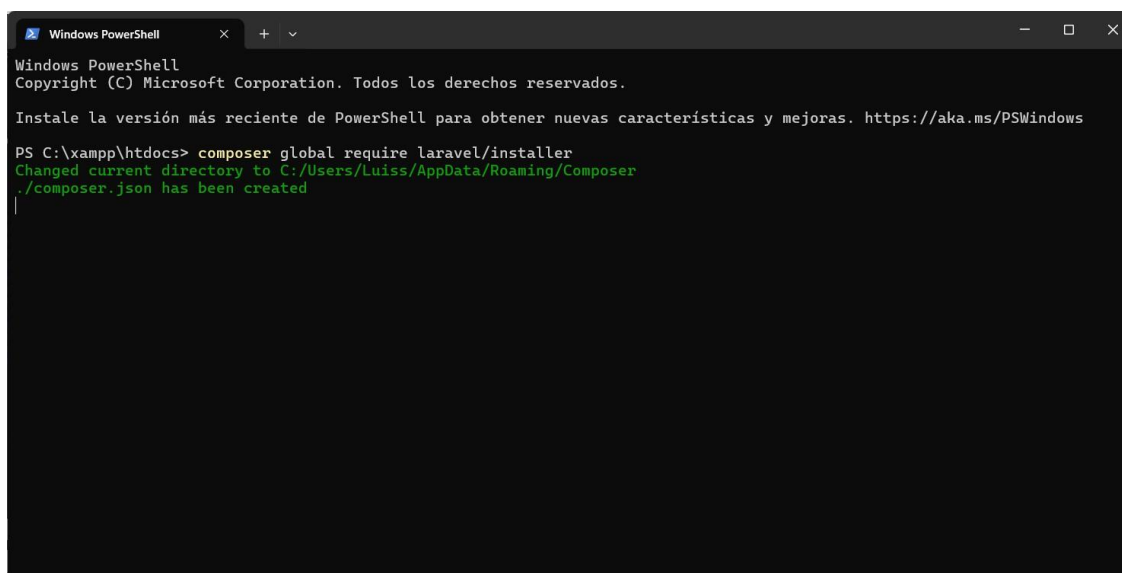


The screenshot shows the Laravel website. The header has the Laravel logo and the text 'Laravel - The PHP Framework For Web Artisans'. Below the header, there is a large image of the Laravel logo and a snippet of the framework's description. The main content area has links for 'Installation', 'Laravel Documentation', 'Frontend', 'Configuration', and 'Bootcamp'. The footer has the text 'Laravel is a PHP web application framework with expressive, elegant syntax. We've already laid the foundation — freeing you to create without sweating the ...'.

Le damos a get started para que nos dirija a al documento donde están los pasos para la instalación



Una vez tenemos los comandos para la correcta instalación abrimos nuestro cmd en la ruta donde vamos a instalar y crear nuestro proyecto Laravel



La instalación comenzara de forma automática

```
C:\WINDOWS\system32\cmd. x + v
- Installing psr/log (3.0.0): Extracting archive
- Installing egulias/email-validator (4.0.2): Extracting archive
  - Installing symfony/mailer (v7.0.6): Extracting archive
  - Installing symfony/error-handler (v7.0.6): Extracting archive
- Installing symfony/http-kernel (v7.0.6): Extracting archive
  - Installing symfony/finder (v7.0.0): Extracting archive
- Installing ramsey/collection (2.0.0): Extracting archive
- Installing brick/math (0.11.0): Extracting archive
  - Installing ramsey/uuid (4.7.5): Extracting archive
  - Installing psr/simple-cache (3.0.0): Extracting archive
- Installing nunomaduro/termwind (v2.0.1): Extracting archive
- Installing symfony/translation-contracts (v3.4.2): Extracting archive
- Installing symfony/translation (v7.0.4): Extracting archive
  - Installing psr/clock (1.0.0): Extracting archive
  - Installing symfony/clock (v7.0.5): Extracting archive
- Installing carbonphp/carbon-doctrine-types (3.2.0): Extracting archive
- Installing nesbot/carbon (3.2.4): Extracting archive
  - Installing monolog/monolog (3.5.0): Extracting archive
  - Installing league/mime-type-detection (1.15.0): Extracting archive
  - Installing league/flysystem (3.27.0): Extracting archive
- Installing league/flysystem-local (3.25.1): Extracting archive
- Installing nette/utils (v4.0.4): Extracting archive
  - Installing nette/schema (v1.3.0): Extracting archive
- Installing dflydev/dot-access-data (v3.0.2): Extracting archive
- Installing league/config (v1.2.0): Extracting archive
- Installing league/commonmark (2.4.2): Extracting archive
  - Installing laravel/serializable-closure (v1.3.3): Extracting archive
- Installing laravel/prompts (v0.1.17): Extracting archive
- Installing laravel/framework (v11.2.0): Extracting archive
```

Si el proceso se realizó de forma correcta nos aparecerá la siguiente ventana de instalación.

```
C:\WINDOWS\system32\cmd. x + v
- Installing symfony/process (v7.0.4): Extracting archive
- Installing symfony/polyfill-mbstring (v1.29.0): Extracting archive
- Installing symfony/polyfill-intl-normalizer (v1.29.0): Extracting archive
- Installing symfony/polyfill-intl-grapheme (v1.29.0): Extracting archive
- Installing symfony/polyfill-ctype (v1.29.0): Extracting archive
- Installing symfony/string (v7.0.4): Extracting archive
- Installing symfony/service-contracts (v3.4.2): Extracting archive
- Installing symfony/console (v7.0.6): Extracting archive
- Installing illuminate/collections (v11.2.0): Extracting archive
- Installing laravel/prompts (v0.1.17): Extracting archive
- Installing voku/portable-ascii (2.0.1): Extracting archive
- Installing symfony/translation-contracts (v3.4.2): Extracting archive
- Installing symfony/translation (v7.0.4): Extracting archive
- Installing symfony/polyfill-php80 (v1.29.0): Extracting archive
- Installing symfony/polyfill-php83 (v1.29.0): Extracting archive
- Installing psr/clock (1.0.0): Extracting archive
- Installing symfony/clock (v7.0.5): Extracting archive
- Installing carbonphp/carbon-doctrine-types (3.2.0): Extracting archive
- Installing nesbot/carbon (3.2.4): Extracting archive
- Installing illuminate/support (v11.2.0): Extracting archive
- Installing symfony/finder (v7.0.0): Extracting archive
- Installing illuminate/filesystem (v11.2.0): Extracting archive
- Installing laravel/installer (v5.7.1): Extracting archive
Generating autoload files
18 packages you are using are looking for funding.
Use the 'composer fund' command to find out more!
No security vulnerability advisories found.
Using version ^5.7 for laravel/installer

C:\xampp\htdocs>
```


Una vez tenemos instalado Laravel podemos crear nuestro proyecto para ello la versión 11 nos permitirá la configuración previa del mismo. (Si damos solo enters en la instalación se nos configura de forma automática lo recomendado)

```
C:\WINDOWS\system32\cmd. x + v
- Installing symfony/polyfill-php83 (v1.29.0): Extracting archive
- Installing psr/clock (1.0.0): Extracting archive
- Installing symfony/clock (v7.0.5): Extracting archive
- Installing carbonphp/carbon-doctrine-types (3.2.0): Extracting archive
- Installing nesbot/carbon (3.2.4): Extracting archive
- Installing illuminate/support (v11.2.0): Extracting archive
- Installing symfony/finder (v7.0.0): Extracting archive
- Installing illuminate/filesystem (v11.2.0): Extracting archive
- Installing laravel/installer (v5.7.1): Extracting archive
Generating autoload files
18 packages you are using are looking for funding.
Use the 'composer fund' command to find out more!
No security vulnerability advisories found.
Using version ^5.7 for laravel/installer

C:\xampp\htdocs>laravel new Messi

Laravel

Would you like to install a starter kit? [No starter kit]:
[none] No starter kit
[breeze] Laravel Breeze
[jetstream] Laravel Jetstream
> |
```

Una vez realizada la configuración nos saldrá un mensaje si se instaló de forma correcta junto a la base de datos de igual manera

```
C:\WINDOWS\system32\cmd. x + v
nunomaduro/termwind ..... DONE
pestphp/pest-plugin-laravel ..... DONE
spatie/laravel-ignition ..... DONE

91 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].

[INFO] Preparing tests directory.

No security vulnerability advisories found.
phpunit.xml ..... File already exist
s.
tests/Pest.php ..... File created.
tests/TestCase.php ..... File already exists.
tests/Unit/ExampleTest.php ..... File already exists.
tests/Feature/ExampleTest.php ..... File already exists.

[INFO] Application ready in [Messi]. You can start your local development using:

→ cd Messi
→ php artisan serve

New to Laravel? Check out our bootcamp and documentation. Build something amazing!

C:\xampp\htdocs>|
```

Ya con el proyecto instalado e iniciado debemos empezar con la creación del formulario que nos permite por defecto Laravel con su Autenticación + Bootstrap para después personalizar con lo solicitado en la actividad.

```
No security vulnerability advisories found.  
Using version ^4.5 for laravel/ui  
PS C:\xampp\htdocs\Messi> php artisan ui bootstrap --auth
```

Si, se aplicó los códigos de manera correcta nos saldrá el siguiente mensaje de confirmación.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
pestphp/pest-plugin-laravel ..... DONE  
spatie/laravel-ignition ..... DONE  
  
91 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 ^4.5 for laravel/ui
```

Ahora vamos a crear el formulario por defecto que nos brinda Bootstrap para ello usamos el siguiente comando

```
PS C:\xampp\htdocs\Messi> php artisan ui bootstrap --auth  
  
The [Controller.php] file already exists. Do you want to replace it? (yes/no) [yes]  
> yes  
  
[INFO] Authentication scaffolding generated successfully.  
[INFO] Bootstrap scaffolding installed successfully.  
[WARN] Please run [npm install && npm run dev] to compile your fresh scaffolding.  
PS C:\xampp\htdocs\Messi>
```

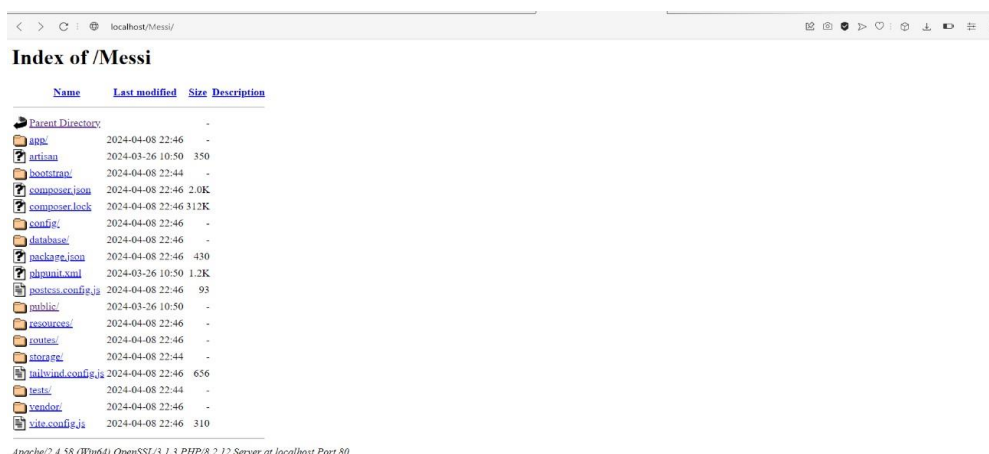
Una vez contamos con el formulario por defecto creado usamos los comandos recomendados para el npm y poder levantar el servidor

```
PS C:\xampp\htdocs\Messi> npm --version  
10.5.0  
PS C:\xampp\htdocs\Messi> npm install  
[.....] - idealTree\Messi: sill idealTree buildDeps
```

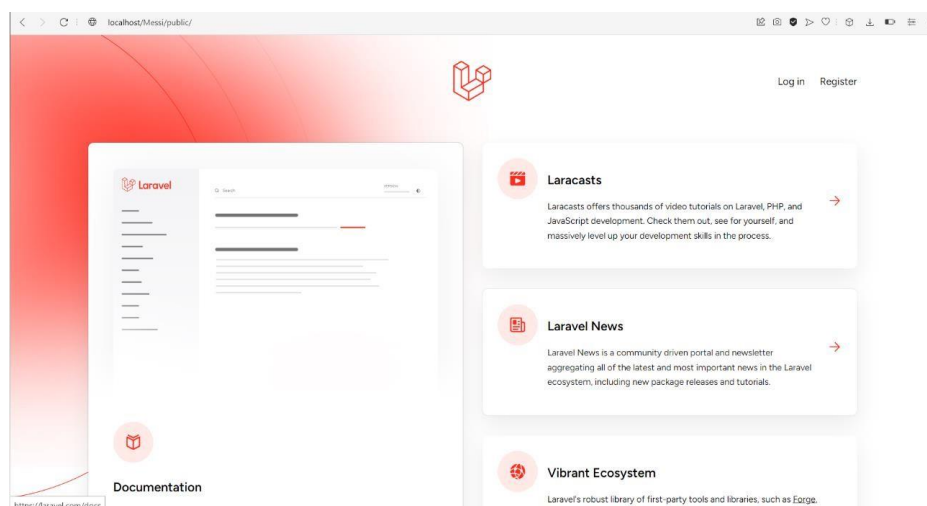

Ahora para poder iniciar el servidor necesitamos que todos los programas mencionados al inicio

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
VITE v5.2.8 ready in 200 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
LARAVEL v11.2.0 plugin v1.0.2
→ APP_URL: http://localhost
```

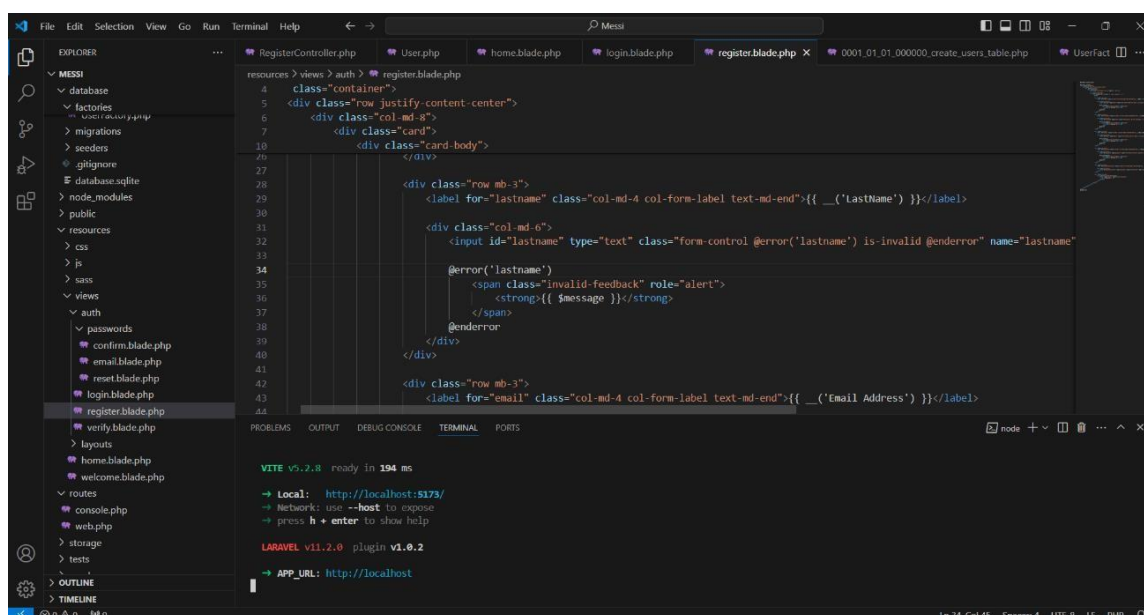
Ahora podemos acceder por el localhost y seleccionar la única carpeta que nos permitirá acceder al index que es la que tienes como nombre de Public.



Una vez ingresamos nos aparecerá la pagina index por defecto de Laravel. Ya en el index en la url debemos añadir /home el cual nos redireccionara al Login ya que en el código genero directo una vista con ese nombre que guardara el formulario de registro y de login.



Además de esto debemos tener en cuenta que es una plantilla otorgada la cual se puede ajustar a las necesidades del programador.



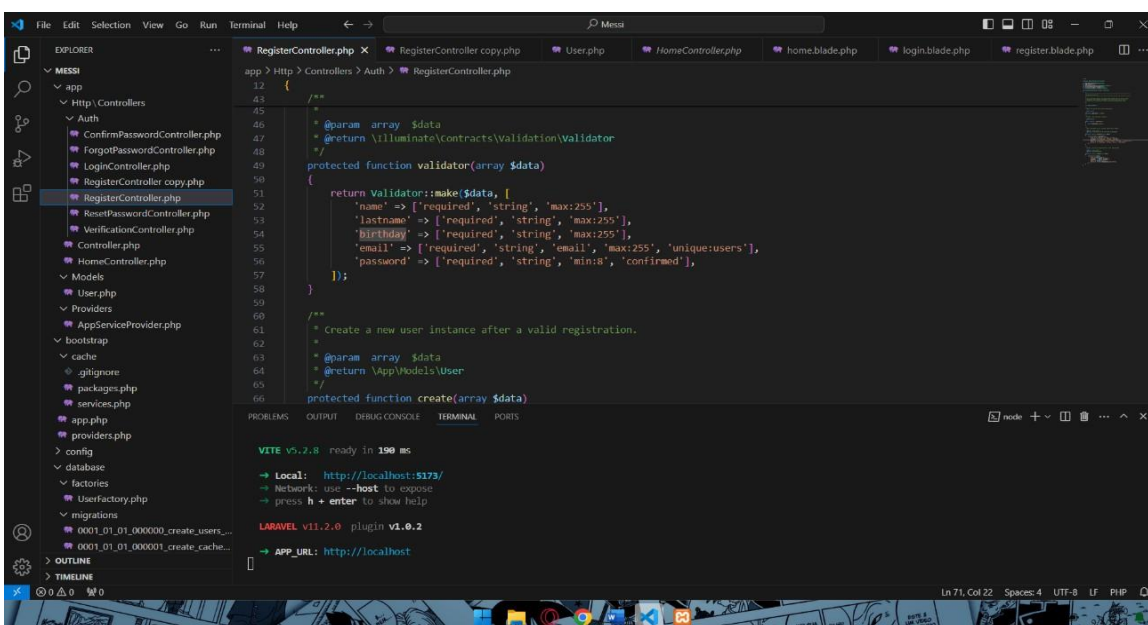
```

resources > views > auth > register.blade.php
4  class="container">
5  <div class="row justify-content-center">
6  <div class="col-md-8">
7  <div class="card">
10 <div class="card-body">
16
27
28
29 <div class="row mb-3">
30 <label for="lastname" class="col-md-4 col-form-label text-md-end">{{ __('LastName') }}</label>
31
32 <div class="col-md-6">
33 <input id="lastname" type="text" class="form-control @error('lastname') is-invalid @enderror" name="lastname"
34
35 @error('lastname')
36 <span class="invalid-feedback" role="alert">
37 <strong>{{ $message }}</strong>
38 </span>
39 @enderror
40 </div>
41
42 <div class="row mb-3">
43 <label for="email" class="col-md-4 col-form-label text-md-end">{{ __('Email Address') }}</label>
44

```

VITE v5.2.8 ready in 194 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
LARAVEL v11.2.0 plugin v1.0.2
→ APP_URL: http://localhost

Para que nuestro formulario funcione de forma correcta no debemos olvidarnos que debemos ejecutar o dicho de forma más exacta migrar los modelos de tablas que se generaron para tener la conexión a nuestra base de Datos.



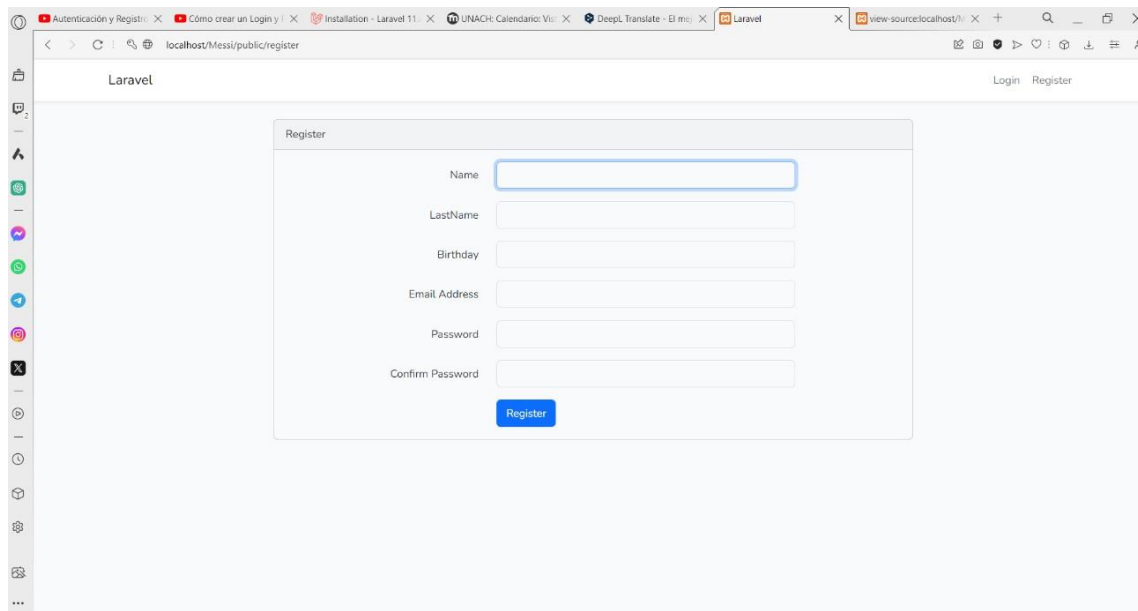
```

app > Http > Controllers > Auth > RegisterController.php
12 {
13     /**
14      *
15      * @param array $data
16      * @return \Illuminate\Contracts\Validation\Validator
17      */
18     protected function validator(array $data)
19     {
20         return Validator::make($data, [
21             'name' => ['required', 'string', 'max:255'],
22             'lastname' => ['required', 'string', 'max:255'],
23             'birthday' => ['required', 'string', 'max:255'],
24             'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
25             'password' => ['required', 'string', 'min:8', 'confirmed'],
26         ]);
27     }
28
29     /**
30      * Create a new user instance after a valid registration.
31      *
32      * @param array $data
33      * @return \App\Models\User
34      */
35     protected function create(array $data)
36     {
37
38     }
39 }

```

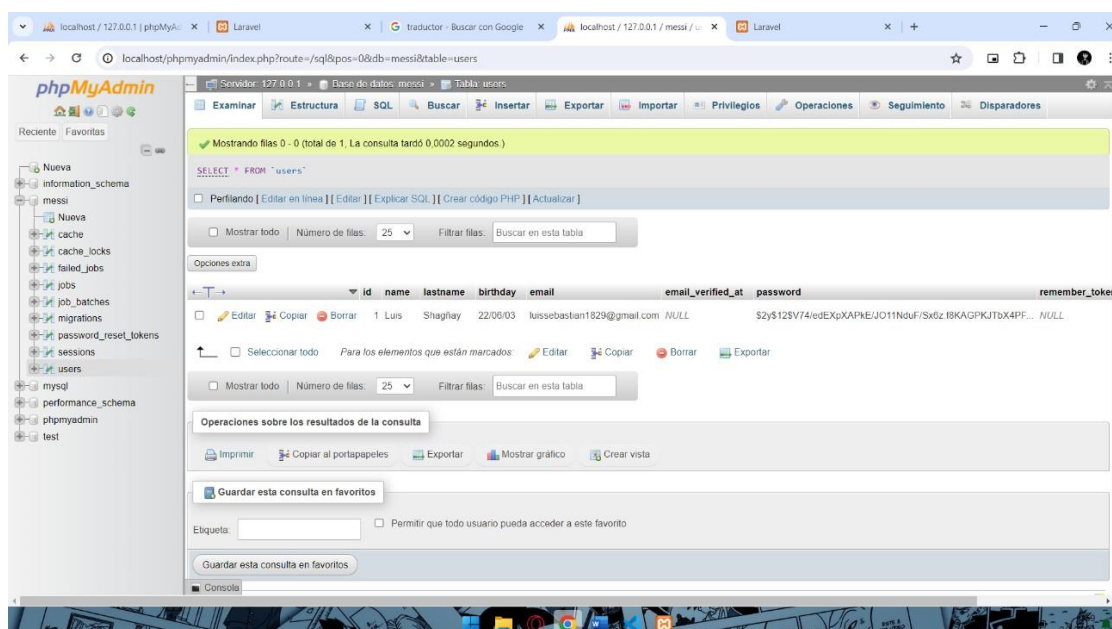
VITE v5.2.8 ready in 190 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
LARAVEL v11.2.0 plugin v1.0.2
→ APP_URL: http://localhost

Para poder personalizar a gusto no debemos olvidar que debemos cambiar en aquel php que hace la conexión, que define las tablas y migra las tablas para que al momento de correr el servidor no nos genere ningún error y guarde de forma correcta los datos.



The screenshot shows a web browser window with the URL `localhost/Messi/public/register`. The page displays a registration form titled "Register". The form includes input fields for "Name", "LastName", "Birthday", "Email Address", "Password", and "Confirm Password". A blue "Register" button is located at the bottom right of the form. The browser's address bar shows the URL, and the page title is "Laravel".

Una vez Llenamos el formulario podemos verificar que la conexión con la base de datos se realice de forma correcta y guarda nuestros datos.

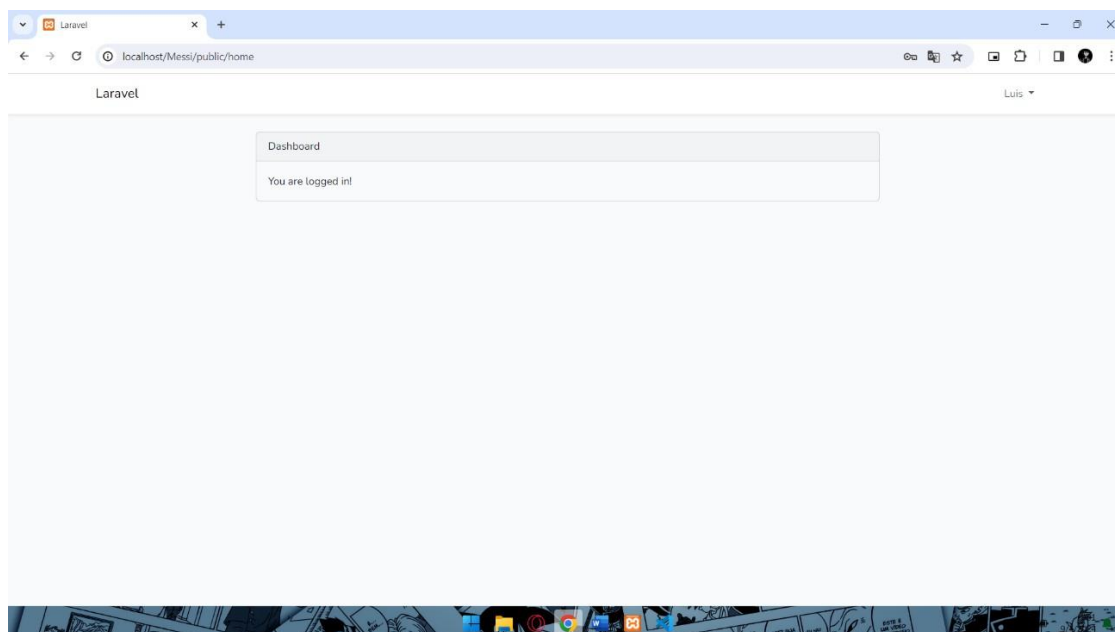


The screenshot shows the phpMyAdmin interface. The left sidebar displays the database structure, including a database named "messi" and a table named "users". The main area shows the "users" table with the following columns: `id`, `name`, `lastname`, `birthday`, `email`, `email_verified_at`, `password`, and `remember_token`. The table contains one row of data for a user named "Luis Shaghtay". The interface includes various tools for examining, editing, and exporting data.

id	name	lastname	birthday	email	email_verified_at	password	remember_token
1	Luis	Shaghtay	22/06/03	luissebastian1829@gmail.com	NULL	\$2y\$12\$V74/edExpXAPKE/JO11NqUf/Sx6z18KAGPKJTBx4PF...	NULL



Como resultado final en el proyecto podemos acceder con las credenciales ingresadas a un Dashboard básico de igual manera cerrar sesión.



Una vez tenemos la interface de inicio vamos a aumentar la seguridad en el inicio de sesión solicitando para verificación del email para confirmar el login

Para ello vamos a agregar las siguientes líneas de código a nuestro proyecto

```
55 MAIL_MAILER=smtp
56 MAIL_HOST=sandbox.smtp.mailtrap.io
57 MAIL_PORT=2525
58 MAIL_USERNAME=41e40091833dcb
59 MAIL_PASSWORD=67505f3dadeddb
60 MAIL_FROM_ADDRESS="Luissebastian1829@gmail.com"
61 MAIL_FROM_NAME="${APP_NAME}"
62
```

Agregamos al perfil del usuario la verificación del Email.

```
<?php

namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Laravel\Sanctum\HasApiTokens;

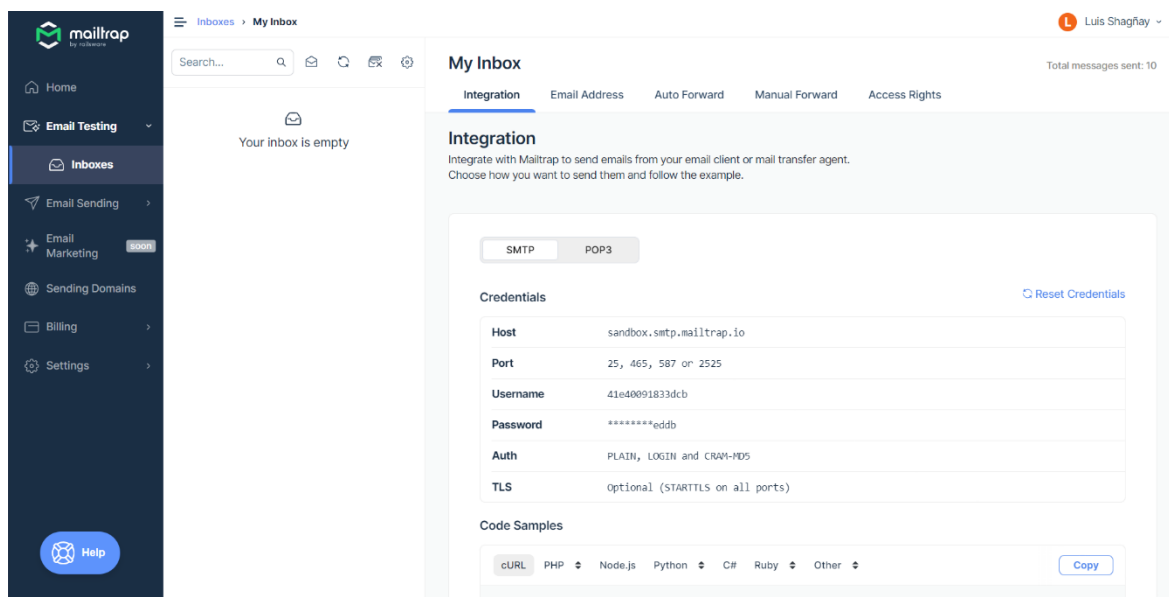
class User extends Authenticatable implements MustVerifyEmail
{
```

Además, desmarcamos la siguiente línea de código.

```
'features' => [
    Features::registration(),
    Features::resetPasswords(),
    Features::emailVerification(),
    Features::updateProfileInformation(),
    Features::updatePasswords(),
    Features::twoFactorAuthentication([
        'confirm' => true,
        'confirmPassword' => true,
        // 'window' => 0,
    ]),
],
```

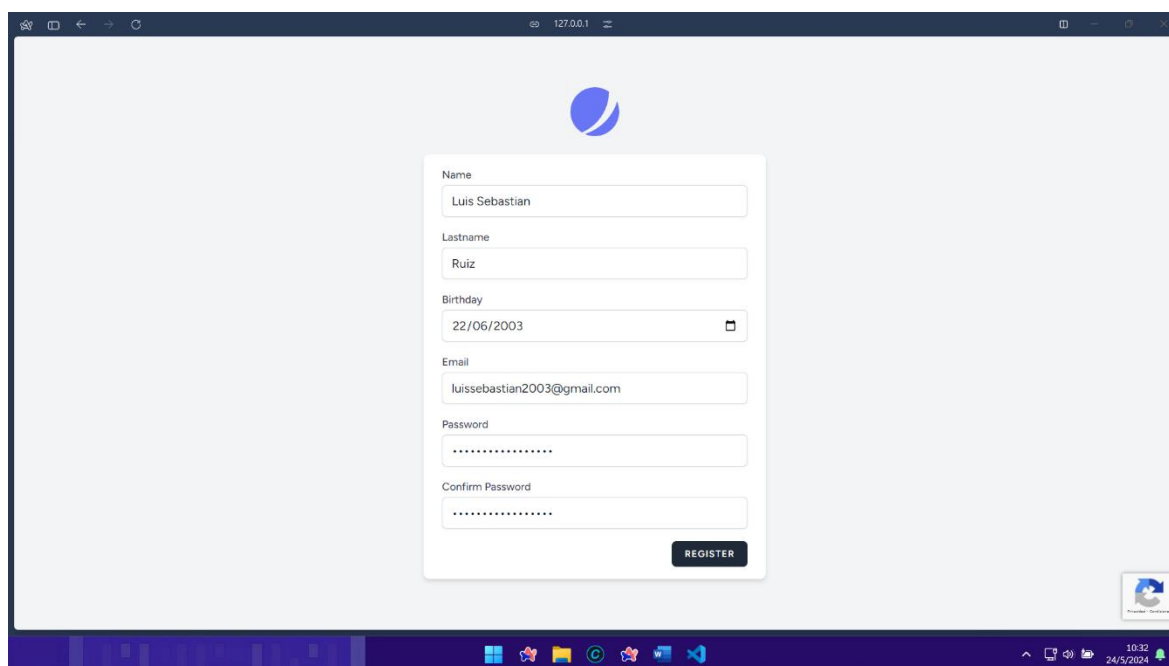


Ahora configuremos nuestro servidor de emails



The screenshot shows the Mailtrap web interface. On the left is a dark sidebar with navigation links: Home, Email Testing, Inboxes (selected), Email Sending, Email Marketing, Sending Domains, Billing, and Settings. The main area is titled 'My Inbox' and shows 'Your inbox is empty'. To the right, the 'Integration' tab is active, displaying SMTP configuration fields. The 'Credentials' section includes: Host (sandbox.smtp.mailtrap.io), Port (25, 465, 587 or 2525), Username (41e40091833dcb), Password (*****eddb), Auth (PLAIN, LOGIN and CRAM-MD5), and TLS (optional (STARTTLS on all ports)). Below this is a 'Code Samples' section with tabs for cURL, PHP, Node.js, Python, C#, Ruby, and Other. A 'Copy' button is visible next to the cURL tab.

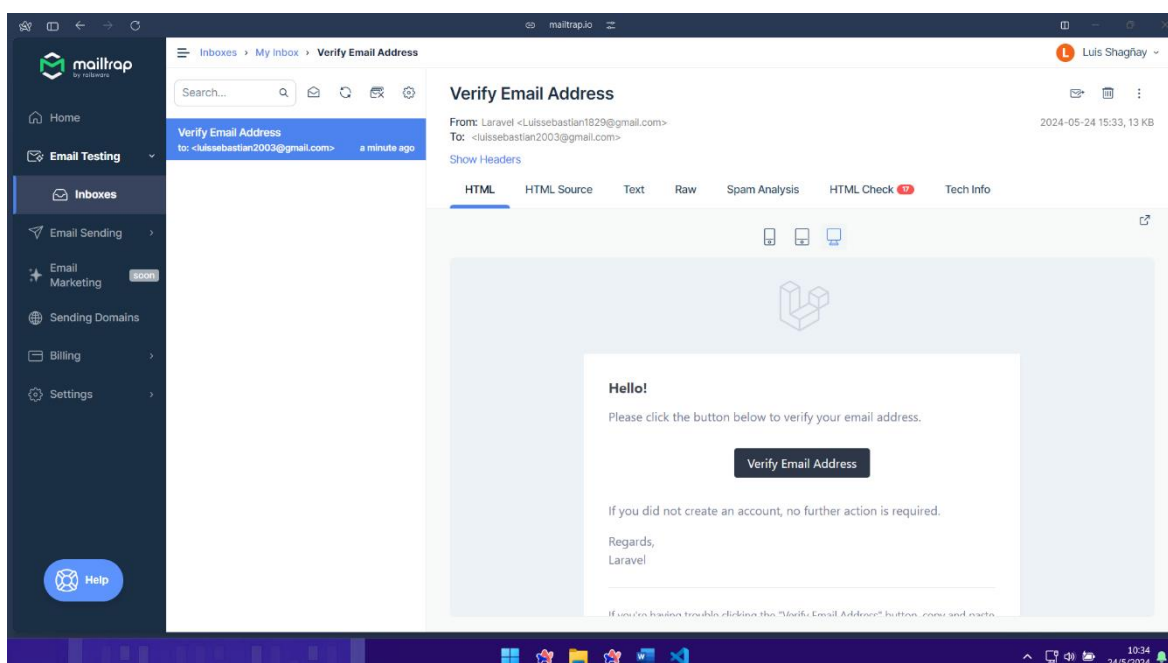
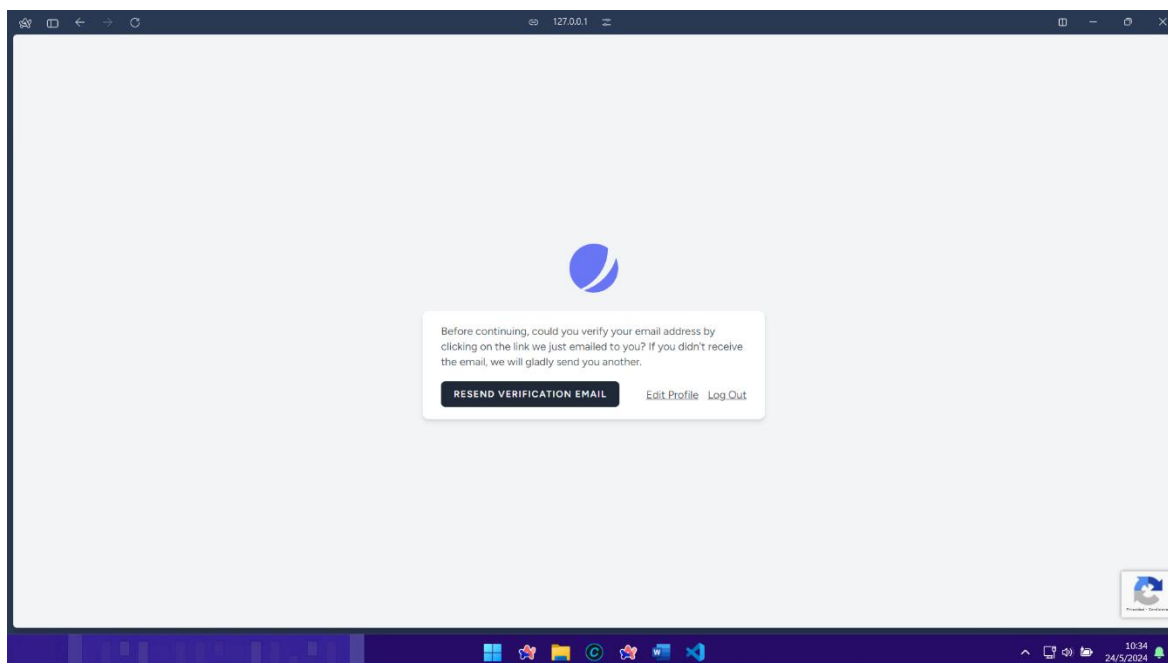
Y procedemos a registrarlos



The screenshot shows a web registration form on a light gray background. The form has a blue circular logo at the top. The fields are: Name (Luis Sebastian), Lastname (Ruiz), Birthday (22/06/2003), Email (luissebastian2003@gmail.com), Password (*****), and Confirm Password (*****). A 'REGISTER' button is at the bottom right of the form. The browser's address bar shows '127.0.0.1'. The Windows taskbar at the bottom shows the time as 10:32 on 24/5/2024.

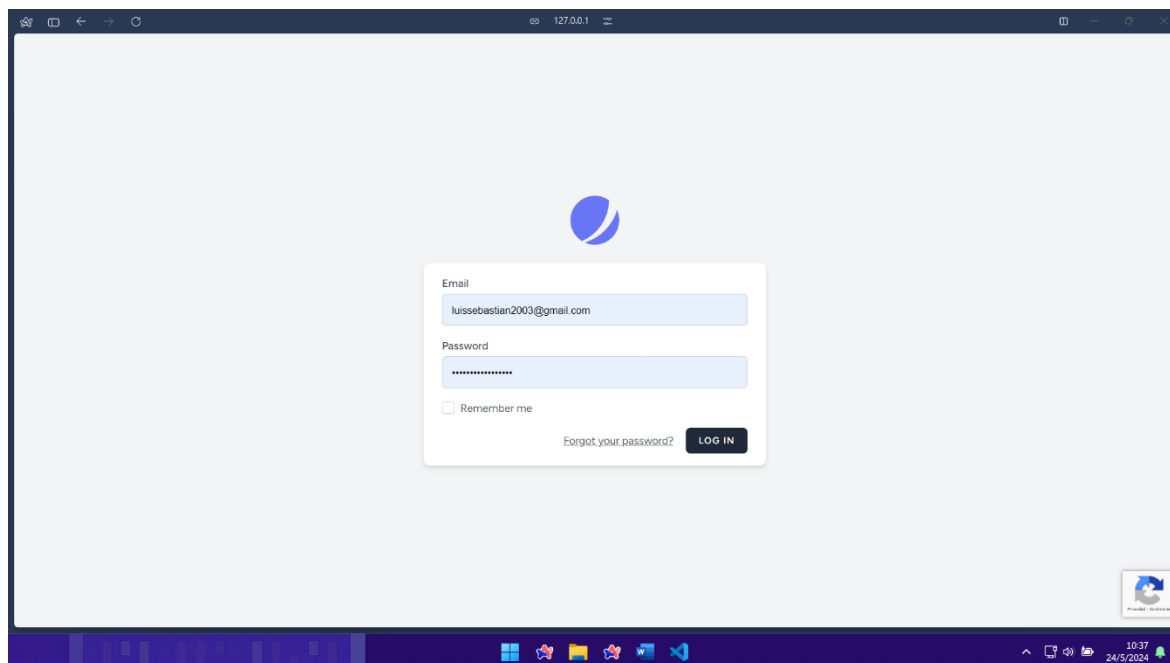


Y confirmar la verificación del email.





Para aceptar y loguearnos



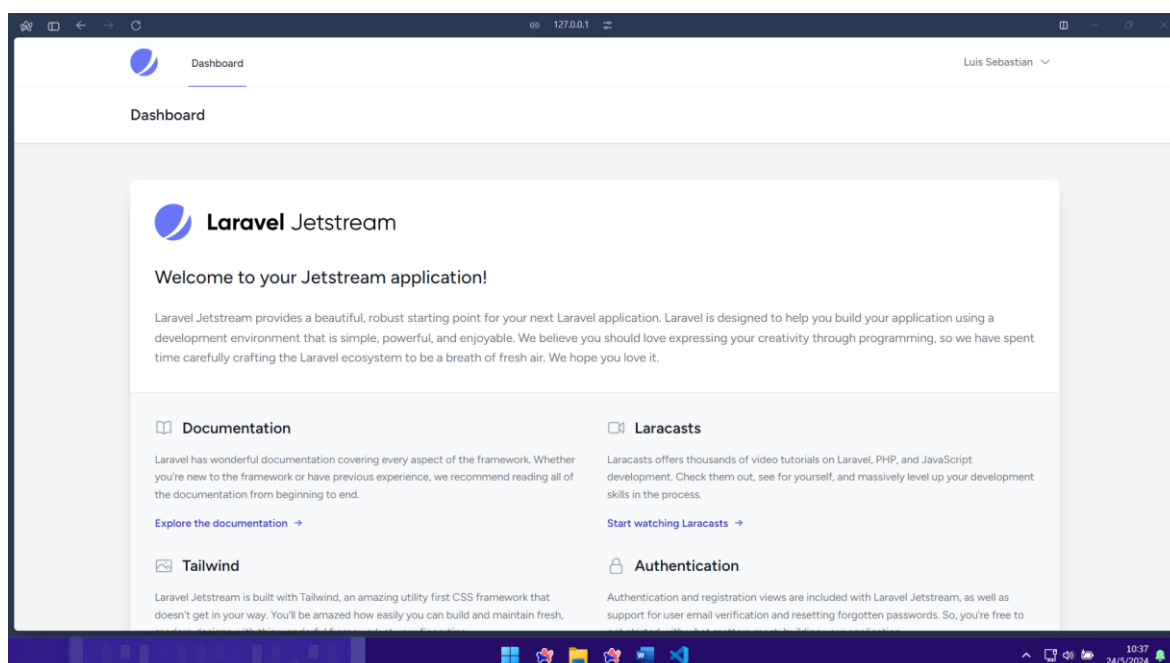
Email

luissebastian2003@gmail.com

Password

☐ Remember me

[Forgot your password?](#) **LOG IN**



Laravel Jetstream

Welcome to your Jetstream application!

Laravel Jetstream provides a beautiful, robust starting point for your next Laravel application. Laravel is designed to help you build your application using a development environment that is simple, powerful, and enjoyable. We believe you should love expressing your creativity through programming, so we have spent time carefully crafting the Laravel ecosystem to be a breath of fresh air. We hope you love it.

Documentation

Laravel has wonderful documentation covering every aspect of the framework. Whether you're new to the framework or have previous experience, we recommend reading all of the documentation from beginning to end.

[Explore the documentation →](#)

Laracasts

Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.

[Start watching Laracasts →](#)

Tailwind

Laravel Jetstream is built with Tailwind, an amazing utility first CSS framework that doesn't get in your way. You'll be amazed how easily you can build and maintain fresh, modern designs with Tailwind. For more information, see the [Tailwind documentation](#).

Authentication

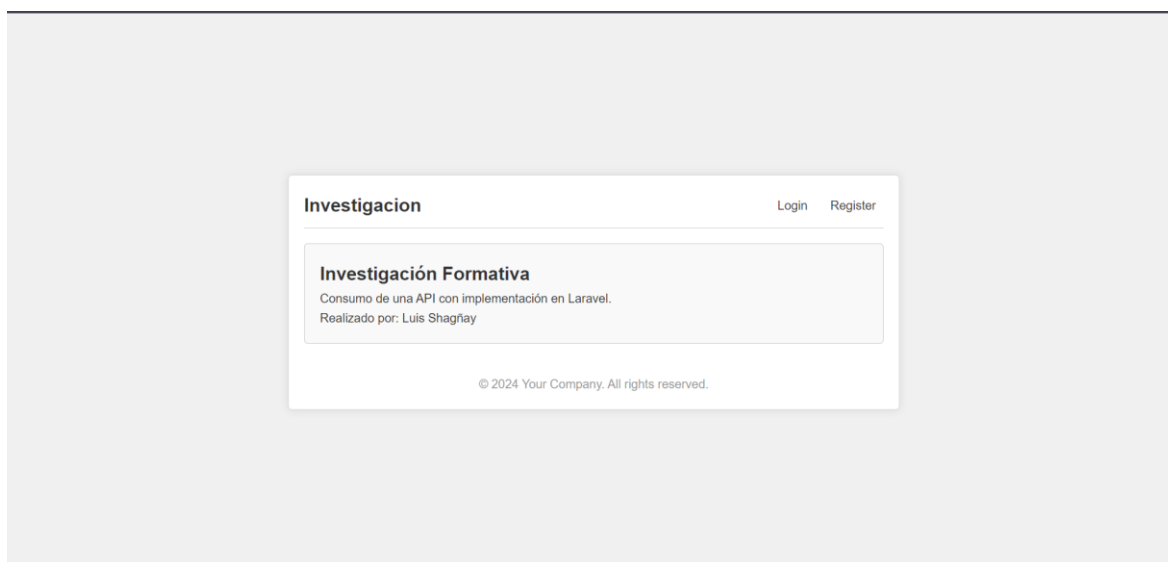
Authentication and registration views are included with Laravel Jetstream, as well as support for user email verification and resetting forgotten passwords. So, you're free to get started with your application right away.



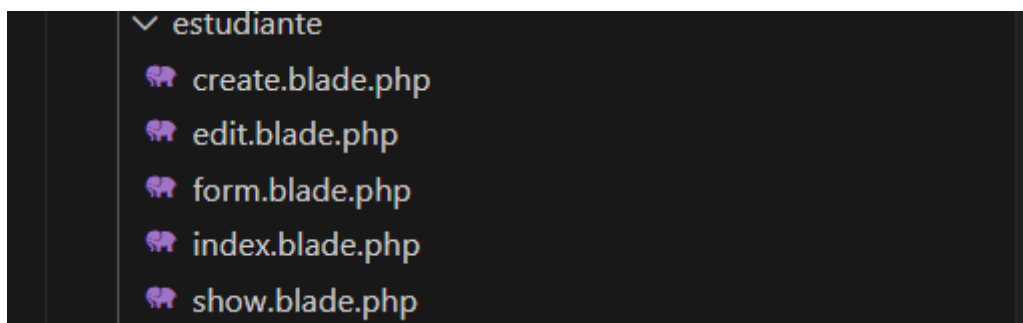
Con la página funcionando de manera correcta comenzaremos con las distintas implementaciones para el debido consumo de nuestra API

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta name="viewport" content="width=device-width, initial-scale=1.0">
5 <title>Dashboard</title>
6 <link rel="stylesheet" href="{{ asset('css/styles.css') }}">
7 </head>
8 <body>
9 <div class="container">
10 <header>
11 <h1>Investigacion</h1>
12 <nav>
13 <ul>
14 <li><a href="#"></a></li>
15 <li><a href="#"></a></li>
16 <li><a href="#"></a></li>
17 <li><a href="#"></a></li>
18 <li><a href="{{ route('login') }}" class="text-sm text-gray-700 dark:text-gray-100"></a></li>
19 <li><a href="{{ route('register') }}" class="ml-4 text-sm text-gray-700 dark:text-gray-100"></a></li>
20 </ul>
21 </nav>
22 </header>
23 <main>
24 <section class="content">
25 <h2>Investigación Formativa</h2>
26 </section>
27 </div>
28 </body>
29 </html>
```

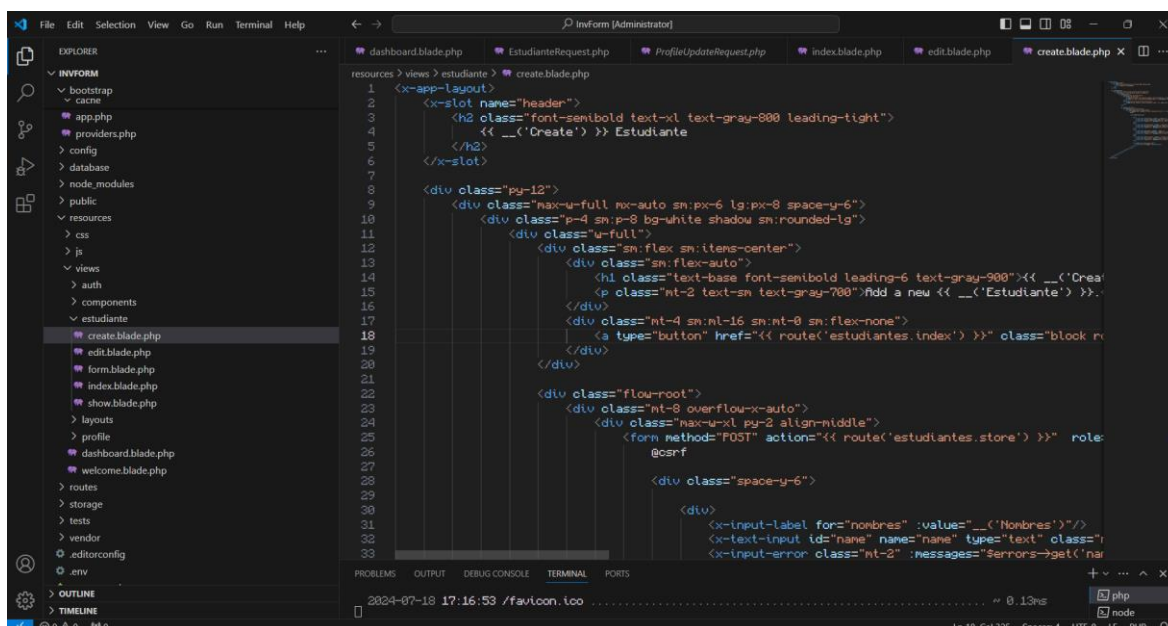
Comenzamos por personalizar nuestra página editando el Welcome.blade



Ahora creamos los archivos necesarios para la correcta implementación de nuestra API



Cada uno de los archivos indicados anteriormente se encargaran de manejar cada una de las opciones que nos ofrece el crud



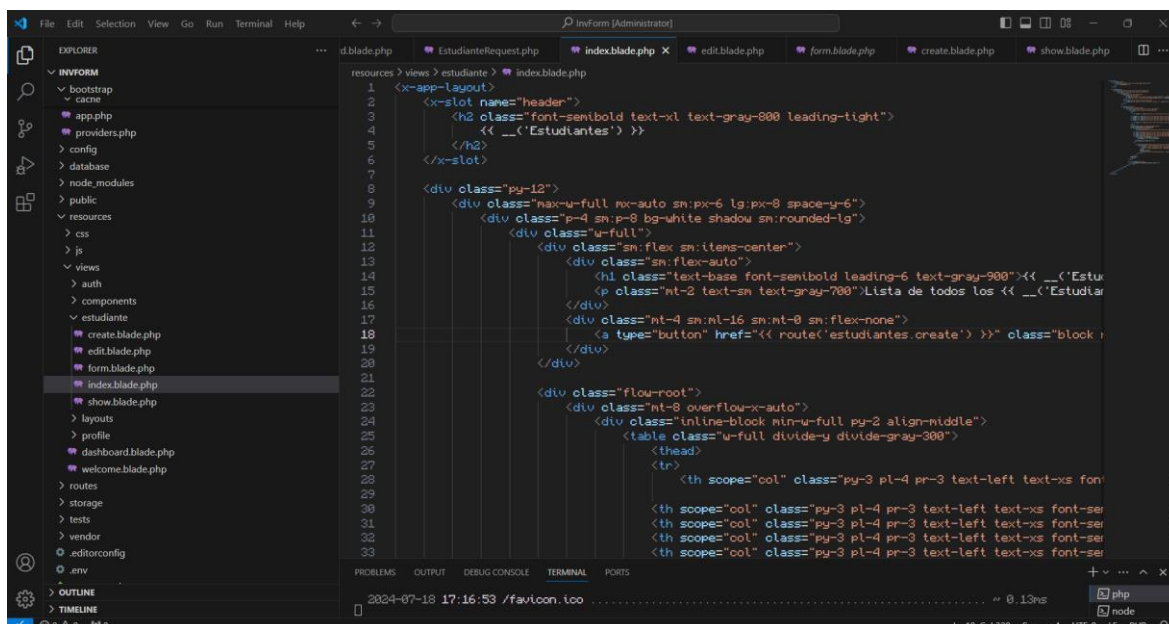


```
File Edit Selection View Go Run Terminal Help
Inform [Administrator]

resources > views > estudiante > edit.blade.php
1 <x-app-layout>
2   <x-slot name="header">
3     <h2 class="font-sans font-size-2xl text-gray-800 leading-tight">
4       <@__('Update') __> Estudiante
5     </h2>
6   </x-slot>
7
8   <div class="py-12">
9     <div class="max-w-full mx-auto sm:px-6 lg:px-8 space-y-6">
10      <div class="p-4 sm:p-8 bg-white shadow sm:rounded-lg">
11        <div class="w-full">
12          <div class="sm:flex sm:items-center">
13            <div class="sm:flex-auto">
14              <h1 class="text-base font-sans font-weight-bold leading-6 text-gray-900">{{ __('Update') }}
15              <p class="mt-2 text-sm text-gray-700">Update existing {{ __('Estudiante') }}
16            </div>
17            <div class="mt-4 sm:mt-0 sm:flex-shrink-0">
18              <a type="button" href="{{ route('estudiantes.index') }}" class="block px-4 py-2 text-sm font-weight-medium text-white bg-blue-600 rounded-md focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-blue-500">Actualizar
19            </div>
20          </div>
21        </div>
22      </div>
23    </div>
24
25    <div class="flow-root">
26      <div class="mt-8 overflow-x-auto">
27        <div class="max-w-xl py-2 align-middle">
28          <form method="POST" action="{{ route('estudiantes.update', $estudiante) }}">
29            <@csrf>
30            @include('estudiante.form')
31          </form>
32        </div>
33      </div>
34    </div>
35  </x-app-layout>
```

```
File Edit Selection View Go Run Terminal Help
Inform [Administrator]

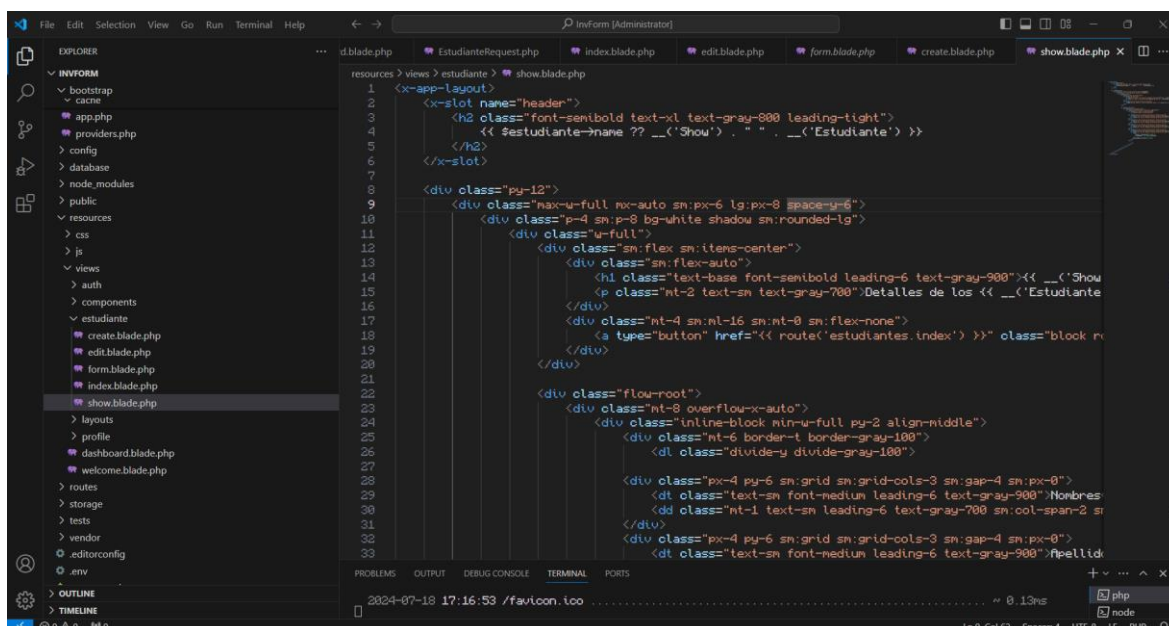
resources > views > estudiante > form.blade.php
1 <div class="space-y-6">
2
3   <div>
4     <x-input-label for="nombres" :value="__('Nombres')"/>
5     <x-text-input id="name" name="name" type="text" class="mt-1 block w-full" :value="old('name')"/>
6     <x-input-error class="mt-2" :messages="$errors->get('name')"/>
7   </div>
8
9   <div>
10    <x-input-label for="apellidos" :value="__('Apellidos')"/>
11    <x-text-input id="apellido" name="apellido" type="text" class="mt-1 block w-full" :value="old('apellido')"/>
12    <x-input-error class="mt-2" :messages="$errors->get('apellido')"/>
13  </div>
14
15  <div>
16    <x-input-label for="email" :value="__('Email')"/>
17    <x-text-input id="email" name="email" type="text" class="mt-1 block w-full" :value="old('email')"/>
18    <x-input-error class="mt-2" :messages="$errors->get('email')"/>
19  </div>
20
21  <div>
22    <x-input-label for="create_at" :value="__('Fecha de Creación')"/>
23    <?php $fecha = $estudiante['create_at'];
24    $fechaNueva = new DateTime($fecha);
25    <?php $fechaNueva->modify('+1 day');
26    <x-text-input id="create_at" name="create_at" type="date" class="mt-1 block w-full" :value="$fechaNueva->format('Y-m-d')"/>
27    <x-input-error class="mt-2" :messages="$errors->get('create_at')"/>
28  </div>
29
30  <div class="flex items-center gap-4">
31    <x-primary-button>Submit</x-primary-button>
32  </div>
33</div>
```



```

1 <x-app-layout>
2   <x-slot name="header">
3     <h2 class="font-sans font-size-2xl text-gray-800 leading-tight">
4       <@__('Estudiantes')>
5     </h2>
6   </x-slot>
7
8   <div class="py-12">
9     <div class="max-w-7xl mx-auto sm:px-6 lg:px-8 space-y-6">
10      <div class="p-4 sm:p-8 bg-white shadow sm:rounded-lg">
11        <div class="u-full">
12          <div class="sm:flex sm:items-center">
13            <div class="sm:flex-auto">
14              <h1 class="text-base font-sans font-size-2xl text-gray-900">@__('Estu
15              <p class="mt-2 text-sm text-gray-700">Lista de todos los @__('Estudiar
16            </div>
17            <div class="mt-4 sm:mt-0 sm:flex-shrink-0">
18              <a type="button" href="@__('route('estudiantes.create'))" class="block
19            </div>
20          </div>
21
22          <div class="flow-root">
23            <div class="mt-8 overflow-x-auto">
24              <div class="inline-block min-w-full py-2 align-middle">
25                <table class="u-full divide-y divide-gray-300">
26                  <thead>
27                    <tr>
28                      <th scope="col" class="py-3 px-4 pr-3 text-left text-sm font
29                    </th>
30                    <th scope="col" class="py-3 px-4 pr-3 text-left text-sm font-se
31                    <th scope="col" class="py-3 px-4 pr-3 text-left text-sm font-se
32                    <th scope="col" class="py-3 px-4 pr-3 text-left text-sm font-se
33                    <th scope="col" class="py-3 px-4 pr-3 text-left text-sm font-se

```



```

1 <x-app-layout>
2   <x-slot name="header">
3     <h2 class="font-sans font-size-2xl text-gray-800 leading-tight">
4       <@__($estudiante->name ?? @__('Show') . " " . @__('Estudiante'))>
5     </h2>
6   </x-slot>
7
8   <div class="py-12">
9     <div class="max-w-7xl mx-auto sm:px-6 lg:px-8 space-y-6">
10      <div class="p-4 sm:p-8 bg-white shadow sm:rounded-lg">
11        <div class="u-full">
12          <div class="sm:flex sm:items-center">
13            <div class="sm:flex-auto">
14              <h1 class="text-base font-sans font-size-2xl text-gray-900">@__('Show
15              <p class="mt-2 text-sm text-gray-700">Detalles de los @__('Estudiante
16            </div>
17            <div class="mt-4 sm:mt-0 sm:flex-shrink-0">
18              <a type="button" href="@__('route('estudiantes.index'))" class="block
19            </div>
20          </div>
21
22          <div class="flow-root">
23            <div class="mt-8 overflow-x-auto">
24              <div class="inline-block min-w-full py-2 align-middle">
25                <div class="mt-6 border-t border-gray-100">
26                  <div class="divide-y divide-gray-100">
27                    <div class="px-4 py-6 sm:grid sm:grid-cols-3 sm:gap-4 sm:px-0">
28                      <dt class="text-sm font-medium leading-6 text-gray-900">Nombre:
29                      <dd class="mt-1 text-sm leading-6 text-gray-700 sm:col-span-2 sm
30                    </div>
31                    <div class="px-4 py-6 sm:grid sm:grid-cols-3 sm:gap-4 sm:px-0">
32                      <dt class="text-sm font-medium leading-6 text-gray-900">Apellido:
33                      <dd class="mt-1 text-sm leading-6 text-gray-700 sm:col-span-2 sm

```

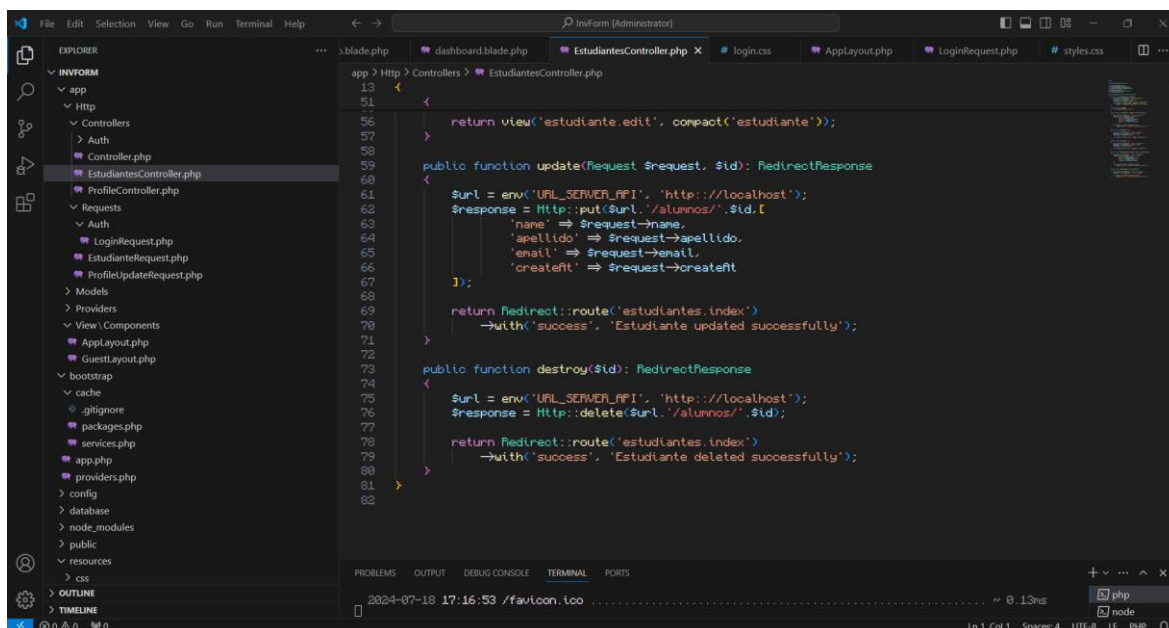
Una vez terminamos lo que es la creación de nuestros .blade que manejaran cada una de las acciones del CRUD realizaremos nuestro controlador que se encargara el consumo de nuestra API realizada con Spring



Controlador

```
1 1?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\Http;
7 use Illuminate\Support\Facades\Redirect;
8 //Agregadas por plantilla
9 use Illuminate\Http\RedirectResponse;
10 use Illuminate\View\View;
11
12 class EstudiantesController extends Controller
13 {
14     public function index(Request $request){
15         $url = env('URL_SERVER_API', 'http://localhost');
16         $response = Http::get($url.'/alumnos');
17         $estudiantes = $response->json();
18         //return view('estudiante.index', compact('estudiantes'));
19         return view('estudiante.index', compact('estudiantes'))
20             ->with('l', ($request->input('page') - 1) * 20);
21     }
22
23     public function create(){
24         return view('estudiante.create');
25     }
26
27     public function store(Request $request): RedirectResponse
28     {
29         $url = env('URL_SERVER_API', 'http://localhost');
30         $response = Http::post($url.'/alumnos',[
31             'name' => $request->name,
32             'apellido' => $request->apellido,
33             'email' => $request->email,
34         ]);
35     }
36 }
```

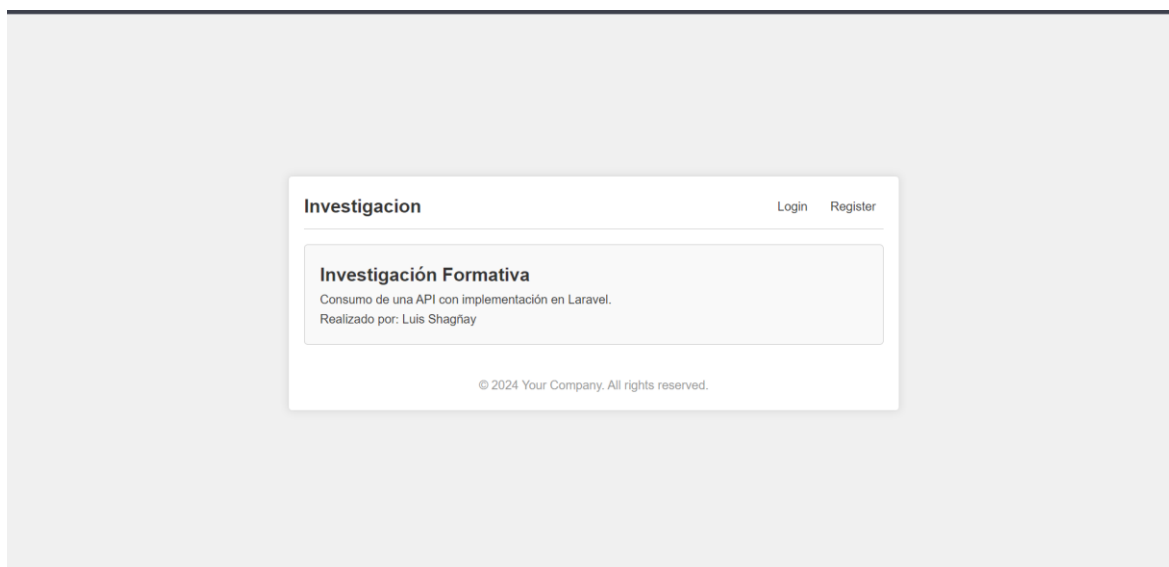
```
37
38     public function show($id): View
39     {
40         $url = env('URL_SERVER_API', 'http://localhost');
41         $response = Http::get($url.'/alumnos/'.$id);
42         $estudiante = $response->json();
43         return view('estudiante.show', compact('estudiante'));
44     }
45
46     public function edit($id): View
47     {
48         $url = env('URL_SERVER_API', 'http://localhost');
49         $response = Http::get($url.'/alumnos/'.$id);
50         $estudiante = $response->json();
51         return view('estudiante.edit', compact('estudiante'));
52     }
53
54     public function update(Request $request, $id): RedirectResponse
55     {
56         $url = env('URL_SERVER_API', 'http://localhost');
57         $response = Http::put($url.'/alumnos/'.$id,[
58             'name' => $request->name,
59             'apellido' => $request->apellido,
60             'email' => $request->email,
61         ]);
62     }
63 }
```



```
13 <
51 <
56 <
57 <
58 <
59 <
60 <
61 <
62 <
63 <
64 <
65 <
66 <
67 <
68 <
69 <
70 <
71 <
72 <
73 <
74 <
75 <
76 <
77 <
78 <
79 <
80 <
81 <
82 <
```

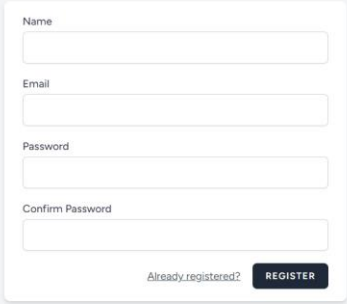
Para concluir realizamos las debidas validaciones desde nuestra interface grafica del proyecto

Pagina Principal



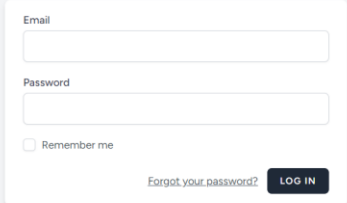


Registro



Registration form interface. The form is centered on a light gray background with a logo above it. It contains four input fields: Name, Email, Password, and Confirm Password. Below the fields is a link "Already registered?" and a "REGISTER" button.

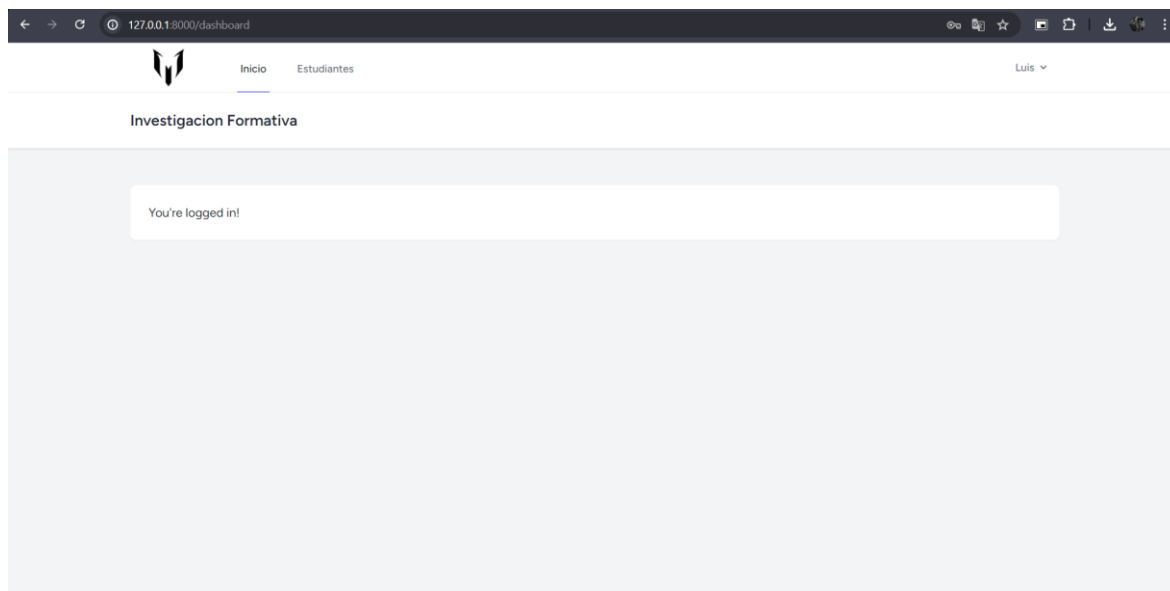
Login



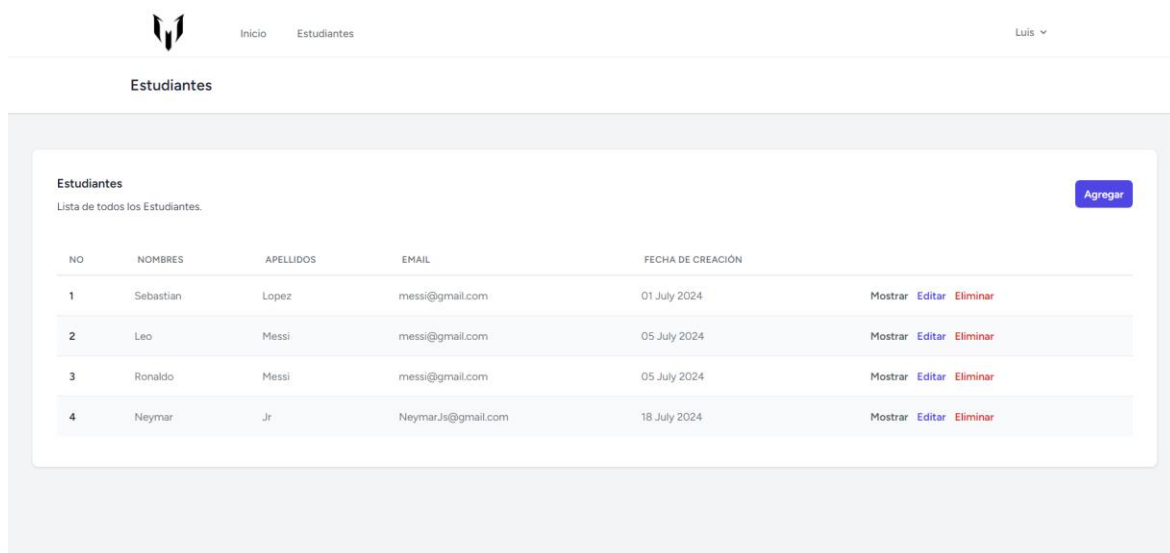
Login form interface. The form is centered on a light gray background with a logo above it. It contains two input fields: Email and Password. Below the fields is a checkbox labeled "Remember me", a link "Forgot your password?", and a "LOG IN" button.



DashBoard




API - Estudiantes





Mostrar - Datos

 Inicio Estudiantes Luis ▾

Show Estudiante


Show Estudiante

Detalles de los Estudiante.

Atras

Nombres	Sebastian
Apellidos	Lopez
Email	messi@gmail.com
Fecha de Creación	01 July 2024

Actualizar - Datos

 Inicio Estudiantes

inglés español Google Translate

 Luis ▾

Update Estudiante

Update Estudiante

Update existing Estudiante.

Atras

Nombres

Sebastian

Apellidos

Lopez

Email

messi@gmail.com


Fecha de Creación

01/07/2024

SUBMIT



Eliminar - Datos

 Inicio Estudiantes

127.0.0.1:8000 dice
¿Es seguro que desea eliminar el registro?

Aceptar Cancelar

Estudiantes

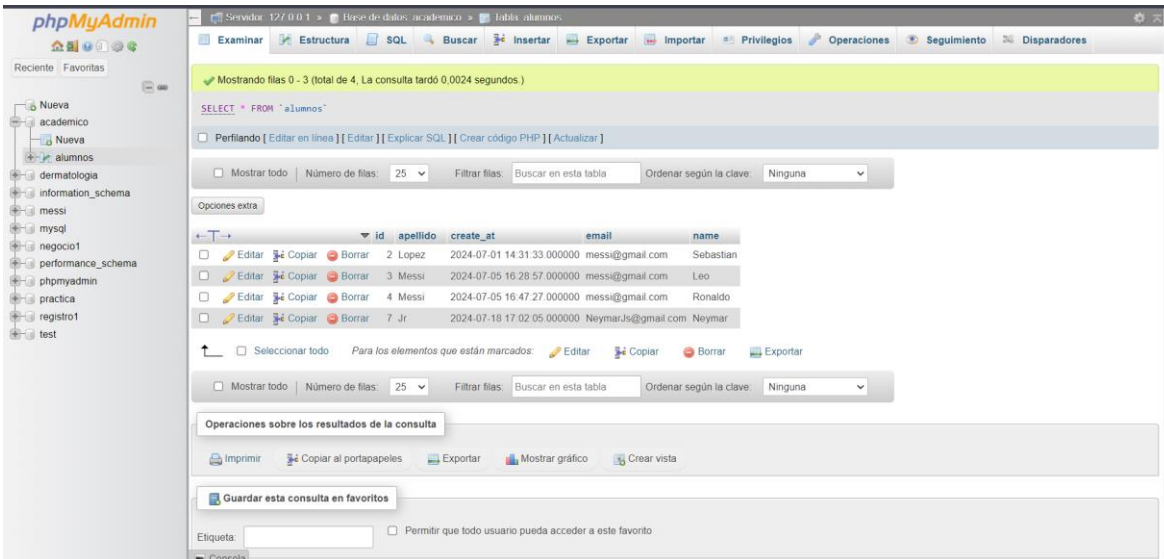
Estudiantes

Lista de todos los Estudiantes.

Agregar

NO	NOMBRES	APELLIDOS	EMAIL	FECHA DE CREACIÓN	
1	Sebastian	Lopez	messi@gmail.com	01 July 2024	Mostrar Editar Eliminar
2	Leo	Messi	messi@gmail.com	05 July 2024	Mostrar Editar Eliminar
3	Ronaldo	Messi	messi@gmail.com	05 July 2024	Mostrar Editar Eliminar
4	Neymar	Jr	NeymarJr@gmail.com	18 July 2024	Mostrar Editar Eliminar

Base de Datos



Mostrando filas 0 - 3 (total de 4, La consulta tardó 0.0024 segundos)

SELECT * FROM `alumnos`

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

	id	apellido	create_at	email	name
<input type="checkbox"/>	2	Lopez	2024-07-01 14:31:33 000000	messi@gmail.com	Sebastian
<input type="checkbox"/>	3	Messi	2024-07-05 16:28:57 000000	messi@gmail.com	Leo
<input type="checkbox"/>	4	Messi	2024-07-05 16:47:27 000000	messi@gmail.com	Ronaldo
<input type="checkbox"/>	7	Jr	2024-07-18 17:02:05 000000	NeymarJr@gmail.com	Neymar

Seleccionar todo | Para los elementos que están marcados: Editar Copiar Borrar Exportar

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

Operaciones sobre los resultados de la consulta

Imprimir Copiar al portapapeles Exportar Mostrar gráfico Crear vista

Guardar esta consulta en favoritos

Etiqueta: Permitir que todo usuario pueda acceder a este favorito

Consola

7.3.1 MARCO TEÓRICO

Visual Studio Code (VS Code)

Propósito: Visual Studio Code es un entorno de desarrollo integrado (IDE) desarrollado por Microsoft que está diseñado para facilitar la escritura y la depuración de código. Está optimizado para trabajar con una variedad de lenguajes de programación, incluyendo HTML, CSS, JavaScript y varios otros [1].

Implementación: VS Code proporciona una interfaz de usuario intuitiva con características como resaltado de sintaxis, finalización de código, depuración integrada, control de versiones y una amplia gama de extensiones que pueden personalizarse según las necesidades del desarrollador [1].

Características:

- Interfaz de usuario altamente personalizable.
- Soporte para una amplia gama de lenguajes de programación y tecnologías web.
- Integración con herramientas de control de versiones como Git.
- Amplia gama de extensiones disponibles para mejorar la funcionalidad del IDE.
- Depuración integrada para identificar y corregir errores en el código.

Bootstrap

Propósito: Bootstrap es un marco de trabajo front-end de código abierto desarrollado por Twitter que facilita el diseño y la estilización de sitios web y aplicaciones web. Está diseñado para ser fácil de usar y adaptable a dispositivos de diferentes tamaños y resoluciones [2].

Implementación: Bootstrap se implementa mediante la inclusión de su archivo de hoja de estilos (CSS) y archivos de JavaScript en el proyecto. Los desarrolladores pueden utilizar las clases predefinidas de Bootstrap en el marcado HTML para aplicar estilos y comportamientos específicos a los elementos de la página [2].

Características:

- Conjunto de componentes y estilos predefinidos que facilitan la creación de interfaces de usuario atractivas.
- Diseño responsivo que se adapta automáticamente a diferentes tamaños de pantalla.
- Grid system que permite la creación de diseños de página flexibles y organizados.
- Componentes personalizables y fácilmente modificables utilizando clases de CSS predefinidas.
- Soporte para navegadores web modernos y dispositivos móviles.

Node.js

Propósito: Node.js es un entorno de ejecución de JavaScript del lado del servidor que permite a los desarrolladores construir aplicaciones web escalables y de alto rendimiento utilizando JavaScript tanto en el lado del cliente como en el servidor. Está diseñado para manejar un gran número de conexiones simultáneas con una eficiencia excepcional [3].

Implementación: Node.js se implementa mediante la instalación del entorno de ejecución en el servidor y la escritura de código JavaScript para definir la lógica de la aplicación. Los desarrolladores pueden utilizar el gestor de paquetes npm (Node Package Manager) para instalar módulos y bibliotecas de terceros que extienden la funcionalidad de Node.js [3].

Características:

- Modelo de E/S sin bloqueo y orientado a eventos que permite manejar múltiples solicitudes simultáneamente de manera eficiente.
- Amplia gama de módulos y bibliotecas disponibles a través de npm para facilitar el desarrollo de aplicaciones.
- Soporte para la creación de servidores web, API RESTful, aplicaciones en tiempo real y otros tipos de aplicaciones web.
- Escalabilidad horizontal que permite escalar fácilmente la aplicación agregando más instancias de Node.js.

Composer

Propósito: Composer es una herramienta de administración de dependencias para PHP que se utiliza para instalar y administrar bibliotecas y paquetes de PHP en un proyecto. Facilita la gestión de las dependencias y la incorporación de bibliotecas de terceros en una aplicación PHP [4].

Implementación: Composer se implementa mediante la instalación del binario de Composer en el sistema y la creación de un archivo composer.json en el directorio raíz del proyecto. En este archivo, se especifican las dependencias del proyecto y Composer se encarga de descargar e instalar las bibliotecas necesarias [4].

Características:

- Gestión de dependencias simple y eficiente para proyectos PHP.
- Instalación automática de bibliotecas y sus dependencias a través de la línea de comandos.
- Soporte para la declaración de dependencias específicas de versión y restricciones de versión.
- Repositorio central de paquetes llamado Packagist que contiene una amplia gama de bibliotecas y paquetes PHP disponibles para su instalación.
- Facilidad para compartir y reutilizar código a través de paquetes y bibliotecas de terceros.

XAMPP

Propósito: XAMPP es un paquete de software libre y de código abierto que proporciona un entorno de desarrollo local para crear y probar aplicaciones web antes de desplegarlas en un servidor en vivo. Incluye componentes como el servidor web Apache, el sistema de gestión de bases de datos MySQL, el intérprete PHP y otros [5].

Implementación: XAMPP se implementa mediante la descarga e instalación del paquete de software en el sistema operativo local. Una vez instalado, se inicia el servidor Apache y se configuran los archivos de configuración según las necesidades del proyecto [5].

Características:

- Entorno de desarrollo local todo en uno que incluye un servidor web, un sistema de gestión de bases de datos y otros componentes necesarios para desarrollar aplicaciones web.
- Fácil de instalar y configurar en sistemas operativos Windows, Linux y macOS.
- Interfaz de usuario gráfica que permite iniciar, detener y configurar los servicios incluidos en XAMPP.
- Soporte para la ejecución de aplicaciones web PHP y MySQL en el entorno local sin necesidad de una conexión a Internet.
- Amplia comunidad de usuarios y desarrolladores que proporciona soporte y recursos adicionales para trabajar con XAMPP.

SQL (Structured Query Language)

Propósito: SQL es un lenguaje de programación utilizado para gestionar bases de datos relacionales. Permite realizar operaciones como la creación, modificación y eliminación de datos en una base de datos, así como la realización de consultas para recuperar información específica [6].

Implementación: SQL se implementa mediante la ejecución de comandos SQL a través de un cliente de base de datos o mediante la integración de consultas SQL dentro del código de la aplicación. Los desarrolladores pueden utilizar lenguajes de programación como PHP para conectarse a la base de datos y ejecutar consultas SQL [6].

Características:

- Lenguaje de programación estándar para la gestión de bases de datos relacionales.
- Sintaxis declarativa que permite especificar qué datos se desean recuperar o manipular sin necesidad de definir cómo realizar la operación.
- Conjunto de comandos básicos como SELECT, INSERT, UPDATE y DELETE para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una base de datos.

- Soporte para la creación y manipulación de tablas, índices, vistas, procedimientos almacenados y otros objetos de base de datos.
- Compatible con una amplia gama de sistemas de gestión de bases de datos relacionales, incluyendo MySQL, PostgreSQL, SQLite, SQL Server y Oracle.

Spring Framework

Propósito: Spring Framework es un marco de trabajo para aplicaciones Java que proporciona una infraestructura integral para el desarrollo de aplicaciones empresariales. Facilita la creación de aplicaciones mediante la gestión de dependencias, la programación orientada a aspectos y la integración con diversos servicios y tecnologías [7].

Implementación: Spring se implementa mediante la inclusión de las bibliotecas del framework en el proyecto y la configuración de los componentes de la aplicación mediante archivos XML o anotaciones Java. Ofrece soporte para crear aplicaciones basadas en capas, manejar transacciones y gestionar beans [7].

Características:

- Inversión de Control (IoC) y Gestión de Dependencias.
- Programación Orientada a Aspectos (AOP).
- Soporte para la creación de aplicaciones web con Spring MVC.
- Integración con diversos servicios y tecnologías como bases de datos, mensajería y seguridad.

Eureka

Propósito: Eureka es un servicio de descubrimiento de servicios desarrollado por Netflix que permite a las aplicaciones localizar y comunicarse entre sí en un entorno distribuido. Es parte del ecosistema de microservicios y facilita la gestión de servicios en aplicaciones escalables [8].

Implementación: Eureka se implementa mediante la configuración de servidores Eureka y clientes en la aplicación. Los servicios se registran en el servidor Eureka y los clientes consultan el servidor para encontrar otros servicios [8].

Características:

- Registro y descubrimiento de servicios.
- Balanceo de carga y recuperación de fallos.
- Integración con otros componentes del ecosistema de microservicios como Spring Cloud.

API Gateway

Propósito: Un API Gateway es un componente que actúa como una puerta de enlace para las solicitudes de API, facilitando la gestión y enrutamiento de las solicitudes hacia los servicios backend. Ayuda a consolidar la funcionalidad de enrutamiento, autenticación y monitoreo en un solo punto [9].

Implementación: El API Gateway se implementa configurando reglas de enrutamiento y políticas de seguridad que dirigen las solicitudes a los servicios adecuados. Puede integrarse con servicios de autenticación y autorización para asegurar el acceso [9].

Características:

- Enrutamiento de solicitudes a servicios backend.
- Gestión de autenticación y autorización.
- Monitoreo y análisis de tráfico.
- Balanceo de carga y gestión de fallos.

7.4 Resultados

1. Gestión Eficiente de Datos de Estudiantes:

- **Crear:** Permite agregar nuevos registros de estudiantes a la base de datos de manera eficiente a través de formularios en la interfaz de usuario.
- **Leer:** Facilita la visualización y búsqueda de registros de estudiantes existentes, permitiendo consultar y mostrar datos en la interfaz de usuario.
- **Actualizar:** Ofrece la capacidad de modificar información de estudiantes, asegurando que los datos permanezcan actualizados y correctos.
- **Eliminar:** Permite eliminar registros de estudiantes cuando sea necesario, gestionando la integridad de los datos en la base de datos.



2. Integración Sólida entre Frontend y Backend:

- **Consumo de API:** Utiliza peticiones HTTP (GET, POST, PUT, DELETE) para interactuar con la API desde el frontend de Laravel, asegurando una comunicación fluida entre el cliente y el servidor.
- **Respuesta en Tiempo Real:** Recibe respuestas rápidas de la API y actualiza la interfaz de usuario sin necesidad de recargar la página, mejorando la experiencia del usuario.

3. Mejoras en el Desarrollo y Mantenimiento:

- **Escalabilidad:** Facilita la escalabilidad del proyecto al separar la lógica del frontend y del backend, lo que permite actualizaciones y mantenimiento más manejables.
- **Modularidad:** La arquitectura de Laravel y la separación de preocupaciones entre cliente y servidor permiten un desarrollo modular y mantenible.

4. Seguridad y Autenticación:

- **Protección de Datos:** Implementa mecanismos de autenticación y autorización para garantizar que solo usuarios autorizados puedan acceder y modificar los datos de estudiantes.
- **Validación:** Utiliza validación de datos en el backend para prevenir la entrada de datos incorrectos o maliciosos a través de la API.

5. Experiencia de Usuario Mejorada:

- **Interfaz Amigable:** Ofrece una interfaz de usuario intuitiva para interactuar con la API, proporcionando formularios y tablas bien diseñadas para gestionar los datos de estudiantes.
- **Notificaciones:** Implementa notificaciones y mensajes de éxito o error para mantener al usuario informado sobre el estado de las operaciones CRUD.

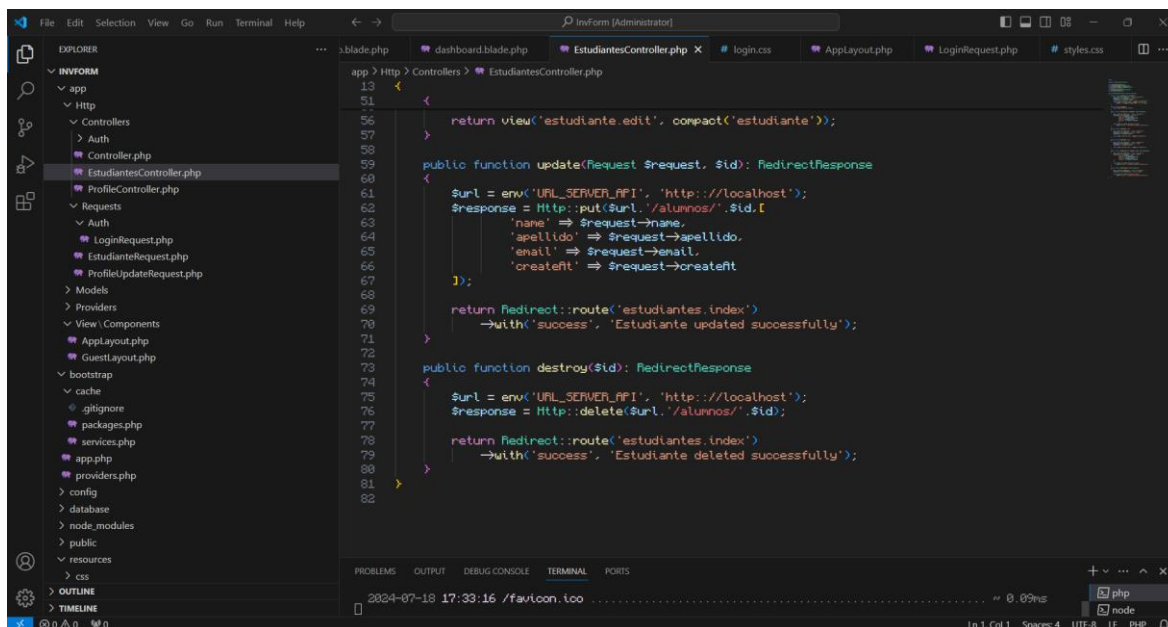
6. Ejecución Eficiente del Proyecto:

- **Desarrollo Ágil:** Utiliza las herramientas y recursos proporcionados por Laravel para desarrollar el proyecto de manera ágil y eficiente.
- **Documentación:** Facilita la documentación de la API y del proyecto en general, proporcionando una referencia clara para desarrolladores futuros.

7.5 Bibliografía

- [1] F. Cristancho, "Visual Studio Code," *Journal of Software Engineering*, vol. 2022, no. 7, pp. 1-8, Jul. 2022.
- [2] F. Cristancho, "Bootstrap Framework," *Web Development Today*, vol. 2022, no. 6, pp. 15-22, Jun. 2022.
- [3] F. Cristancho, "Node.js," *Programming Journal*, vol. 2022, no. 8, pp. 45-53, Aug. 2022.
- [4] F. Cristancho, "Composer for PHP," *PHP Development Magazine*, vol. 2022, no. 5, pp. 30-37, May 2022.
- [5] F. Cristancho, "XAMPP: Local Development Environment," *Open Source Technology Review*, vol. 2022, no. 3, pp. 12-20, Mar. 2022.
- [6] F. Cristancho, "Structured Query Language (SQL)," *Database Systems Journal*, vol. 2022, no. 4, pp. 25-32, Apr. 2022.
- [7] F. Cristancho, "Spring Framework Overview," *Java Development Insights*, vol. 2022, no. 9, pp. 10-18, Sep. 2022.
- [8] F. Cristancho, "Eureka Service Discovery," *Microservices Journal*, vol. 2022, no. 2, pp. 22-29, Feb. 2022.
- [9] F. Cristancho, "API Gateway Concepts," *API Management Review*, vol. 2022, no. 6, pp. 5-13, Jun. 2022.

8. ANEXO



```

app > Http > Controllers > EstudiantesController.php
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82

return view('estudiante.edit', compact('estudiante'));


public function update(Request $request, $id): RedirectResponse
{
    $url = env('URL_SERVER_API', 'http://localhost');
    $response = Http::put($url.'/alumnos/'.$id.[
        'name' => $request->name,
        'apellido' => $request->apellido,
        'email' => $request->email,
        'createit' => $request->createit
    ]);

    return Redirect::route('estudiantes.index')
        ->with('success', 'Estudiante updated successfully');
}

public function destroy($id): RedirectResponse
{
    $url = env('URL_SERVER_API', 'http://localhost');
    $response = Http::delete($url.'/alumnos/'.$id);

    return Redirect::route('estudiantes.index')
        ->with('success', 'Estudiante deleted successfully');
}

```


Inicio Estudiantes
Luis ▾

Estudiantes

Lista de todos los Estudiantes.

Estudiantes

Lista de todos los Estudiantes.

Agregar

NO	NOMBRES	APELLIDOS	EMAIL	FECHA DE CREACIÓN	
1	Sebastian	Lopez	messi@gmail.com	01 July 2024	Mostrar Editar Eliminar
2	Leo	Messi	messi@gmail.com	05 July 2024	Mostrar Editar Eliminar
3	Ronaldo	Messi	messi@gmail.com	05 July 2024	Mostrar Editar Eliminar
4	Neymar	Jr	Neymar.Jr@gmail.com	18 July 2024	Mostrar Editar Eliminar