UNIVERSIDAD POLITÉCNICA DEL VALLE DEL ÉVORA

INGENIERÍA EN SISTEMAS COMPUTACIONALES

Materia: Tecnologías y aplicaciones en internet

EP1. Elaborar un reporte digital a partir de un caso práctico de la creación de un desarrollo web

Alumno y Matrícula:
Kevin Francisco Ochoa Luna – 220120060
Ever Manuel Lopez Ibarra -- 220120060

Grupo: SC8-1

Asesor(a): **Lucina Yacira Cuevas Leyva**

Leopoldo Sánchez Celis, Angostura, Sinaloa a 10 de Abril del 2025

Resumen

RateFlix es un proyecto web que se centra en el permiso de los usuarios para crear, leer y calificar películas de manera simple y segura. El público objetivo incluye adultos para adultos interesados en el cine. Sus características incluyen: conexión, registro de usuarios, el sistema de clasificación con las estrellas, los elementos favoritos, las categorías de películas y las 10 principales.

Bootstrap se utiliza para crear una interfaz atractiva, adaptativa y funcional, compatible con CSS y Google Fonts para una mejor presentación visual. En la parte auxiliar, Laravel ha sido seleccionada, gracias a su estructura basada en MVC, administrando fácilmente la ruta, la autenticidad y la conexión a la base de datos, además de sus objetivos de seguridad al encriptar y proteger las contraseñas contra ataques XSS e inyecciones SQL.

Se desarrollaron rutas específicas, controladores personalizados y vistas dinámicas utilizando Blade. La gestión de usuarios y comentarios fue una prioridad, implementando autenticación mediante middleware. La base de datos está configurada desde el archivo .Vev y los formularios que se han autenticado para una experiencia segura y robusta.

Introducción

En la era digital, las opiniones y evaluaciones son herramientas importantes para la gestión del consumidor, especialmente en el campo del entretenimiento. Rateflix nace como una plataforma web interactiva dedicada a los entusiastas de las películas, lo que permite a los usuarios registrarse, comentar y calificar películas mediante un sistema de estrellas. Esta aplicación intenta crear una comunidad digital confiable con opiniones claras, útiles y constructivas. El desarrollo de Rateflix integra las últimas tecnologías tanto en el perímetro como en el backend, junto con herramientas como Laravel, HTML, CSS, Bootstrap, Google Fonts y una base de datos relacional para garantizar una gestión segura y eficiente de los usuarios y contenido.

Objetivos del proyecto

Desarrollar un sitio móvil interactivo que permita a los usuarios crear, consultar y puntuar reseñas de películas mediante un sistema de clasificación por estrellas. El objetivo es ofrecer un espacio digital accesible y funcional que promueva opiniones más claras, útiles y constructivas sobre una amplia variedad de películas. El sitio integrará tecnologías de frontend y backend, así como APIs y bases de datos para asegurar una correcta gestión de usuarios, contenido y calificaciones.

Audiencia

El proyecto está dirigido a un público adolescente en adelante, incluyendo jóvenes y adultos interesados en el cine, que buscan una plataforma confiable donde puedan expresar sus opiniones sobre películas y descubrir nuevas recomendaciones basadas en valoraciones de otros usuarios. La interfaz está pensada para ser intuitiva y atractiva, adaptándose a los hábitos y preferencias de navegación de esta audiencia.

Funcionalidades principales

- Inicio de sesión
- Registro y gestión de usuarios mediante bases de datos.
- Creación y visualización de reseñas de películas.
- Sistema de puntuación mediante estrellas.
- Clasificación de contenido por categorías.
- Sección de “Favoritos” para guardar películas preferidas.
- Top 10 de películas mejor valoradas por los usuarios.

Selección de las tecnologías

- **Laravel**
- **HTML**
- **CSS**

Frontend

- **Aplicación del Lado del Cliente**

Bootstrap es principalmente un framework de desarrollo web que ayuda a la creación de interfaces, esto debido a que ofrece una gran variedad de componentes y estilos prediseñados.

- **Librerías Utilizadas**

  Google Fonts es un servicio gratuito que ofrece una amplia colección de fuentes tipográficas de código abierto, fácilmente integrables en proyectos web. La adición de este al proyecto fue debido principalmente a que se los recursos de espacio y almacenamiento se ahorran, así como mejora la estética, legibilidad y el rendimiento. Asegurando así compatibilidad con diferentes navegadores y dispositivos.

- **Justificación del framework utilizado**

  Por el lado del cliente, se eligió Bootstrap porque le ayuda a crear rápidamente una interfaz atractiva y funcional. Su sistema de cuadrícula promueve la organización de elementos visuales. Esto es ideal para mostrar una película, categoría y una visión general superior de una manera ordenada y adaptable.

  Además, Bootstrap tiene una variedad de componentes predefinidos, incluidos botones, mapas, advertencias y más que aceleran el desarrollo y aseguran un aspecto moderno sin escribir mucho código CSS. De esta manera, puede centrarse en la experiencia y el contenido del usuario, y lograr un diseño limpio y receptivo que se ajuste a todas las pantallas.

Backend

- **Aplicación del Lado del Servidor**

  Laravel es un framework de desarrollo de backend php el cual ofrece una sintaxis clara y fácil de entender, lo que proporciona productividad y reduce errores. Esta herramienta simplifica tareas y emplea plantillas para vistas dinámicas. Así como también incluye seguridad, escalabilidad y optimización para un buen rendimiento.

- **Librerías Utilizadas**

La implementación de una API RESTful utilizando Laravel y JSON como formato de datos tiene varias ventajas importantes que hacen que esta opción sea ideal para desarrollar aplicaciones en la actualidad. Al combinar el uso de estas herramientas, Laravel hace que crear APIs sea más fácil, confiable y con un buen rendimiento, además de ser sencillo de mantener a largo plazo. Esto tuvo implementación en el guardado de la información del catálogo de películas que se mostrará en la aplicación.

- **Justificación del framework utilizado**

  Para el desarrollo del prototipo de reseñas de películas, se eligió Laravel como el framework para el lado del servidor, esto debido a su facilidad de uso y estructura organizada. Gracias a su arquitectura basada en MVC (modelo, vista, controlador), Laravel permite administrar películas, comentarios e información de los usuarios de una manera clara y, por supuesto, eficiente.

  Además, el uso de este framework proporciona distintas herramientas integradas de Laravel para la autenticación de datos y verificación de los mismos. Lo que significa que solo los usuarios registrados pueden dejar revisiones y comentarios. También hace que sea más fácil la conexión entre el código y la base de datos, haciendo que la consulta de información y la utilización de esta sea más rápidas y seguras.

Desarrollo del backend

a. Implementar rutas y controladores

Rutas

En el archivo *routes/web.php* se definen las rutas principales de la aplicación, organizando el acceso a distintas secciones como películas, reseñas, comentarios, categorías, favoritos y perfil de usuario. También se incluyen las rutas necesarias para la autenticación. Este archivo organiza la navegación y las funcionalidades principales de la aplicación.

```
<?php
```

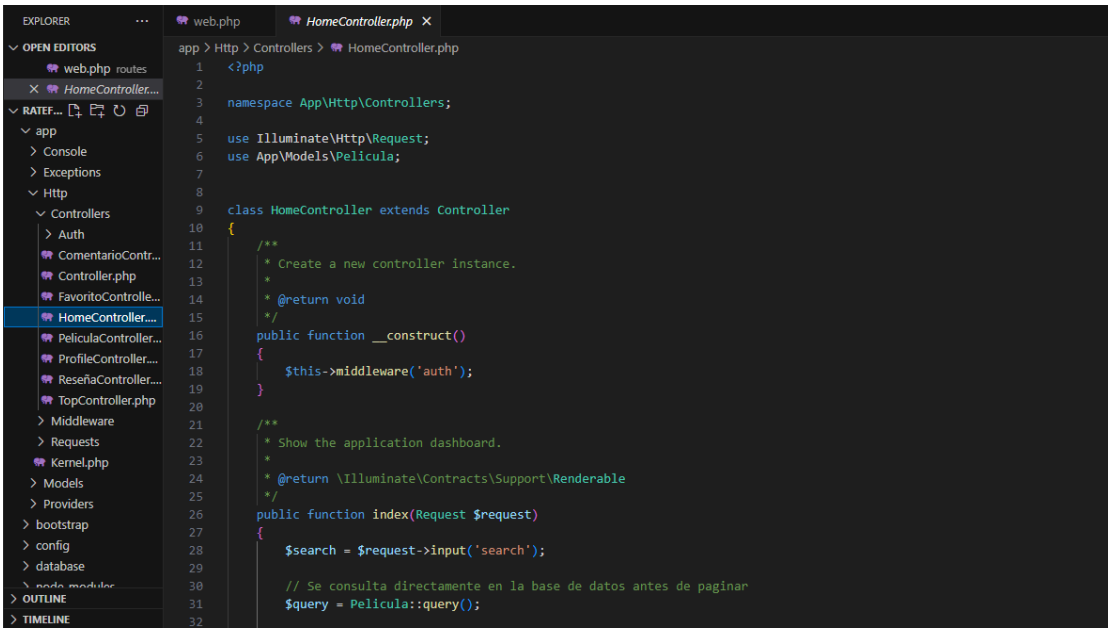
```php
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\PeliculaController;
use App\Http\Controllers\ReseñaController;
use App\Models\Pelicula;
use App\Http\Controllers\FavoritoController;
use App\Http\Controllers\ComentarioController;
use App\Http\Controllers\CategoryController;
use App\Http\Controllers\HomeController;
use App\Http\Controllers\ProfileController;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------*/

// Página de bienvenida
Route::get('/', function () {
    return view('welcome');
});

// Rutas de autenticación
Auth::routes();

// Vistas estáticas
Route::get('/categories',[CategoryController::class,
'index'])->name('categories');
Route::view('/top', 'top')->name('top');
Route::view('/favorites', 'favorites')->name('favorites');

// CRUD de películas y reseñas
Route::resource('peliculas', PeliculaController::class);
Route::resource('reseñas',
ReseñaController::class)->except(['create']);

// Crear reseña desde una película específica
Route::get('/peliculas/{id}/reseñas/create',[ReseñaController:
:class,
'createDesdePelicula'])->name('reseñas.createDesdePelicula');

// Almacenar reseña
Route::resource('reseñas', ReseñaController::class);

// Crear reseña
Route::get('/reseñas/create',[ReseñaController::class,
'create'])->name('reseñas.create');
Route::get('peliculas/{id}/reseñas/create',[ReseñaController::
class,
'createDesdePelicula'])->name('reseñas.createDesdePelicula');

//Comentarios
Route::get('/peliculas/{id}/comentarios/create',
[ComentarioController::class,
'createDesdePelicula'])->name('comentarios.createDesdePelicula
');
```

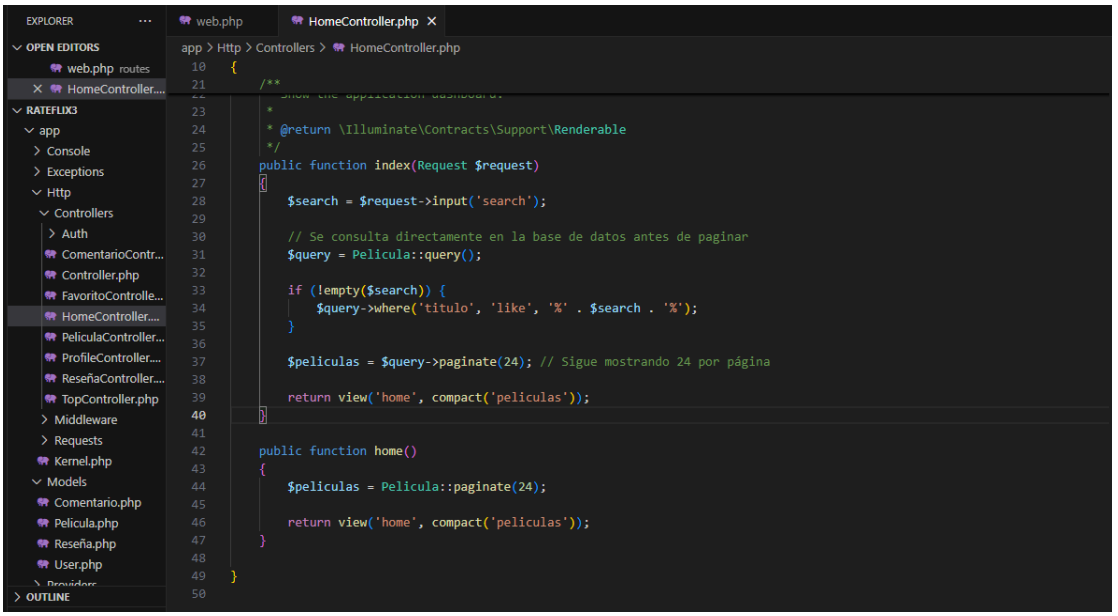
```
Route::post('/peliculas/{pelicula}/comentarios',
[ComentarioController::class,
'store'])->name('comentarios.store');

// Categorias
Route::get('/categories/{genero?}',
[PeliculaController::class,
'porGenero'])->name('peliculas.porGenero');
Route::get('/categories', [PeliculaController::class,
'categorias'])->name('categories');

// Top
Route::get('/top', [TopController::class,
'index'])->name('top.index');
Route::get('/top',
[App\Http\Controllers\PeliculaController::class,
'top'])->name('peliculas.top');
Route::get('/top', [PeliculaController::class,
'top'])->name('peliculas.top');

// Home
Route::get('/home', [HomeController::class,
'home'])->name('home');
Route::get('/home',
[App\Http\Controllers\HomeController::class,
'home'])->name('home');

// Buscador
Route::get('/home', [HomeController::class,
'index'])->name('home');

// Perfil
Route::middleware(['auth'])->group(function () {
Route::get('/perfil',[ProfileController::class,
'index'])->name('perfil');
    Route::post('/perfil/update', [ProfileController::class,
'update'])->name('perfil.update');
});

// Favoritos
Route::post('/pelicula/{id}/favorito',[PeliculaController::cla
ss,
'toggleFavorito'])->middleware('auth')->name('pelicula.favorit
o');
Route::get('/favorites',[PeliculaController::class,
'mostrarFavoritos'])
    ->middleware('auth')
    ->name('favorites');
```

Controladores

Los controladores son los archivos que ayudan a manejar las clases que tiene interacción, es decir, usan la lógica que tienen para ejecutar las peticiones del usuario. También procesan los datos, interactúan con la base de datos y dan respuesta a ello.

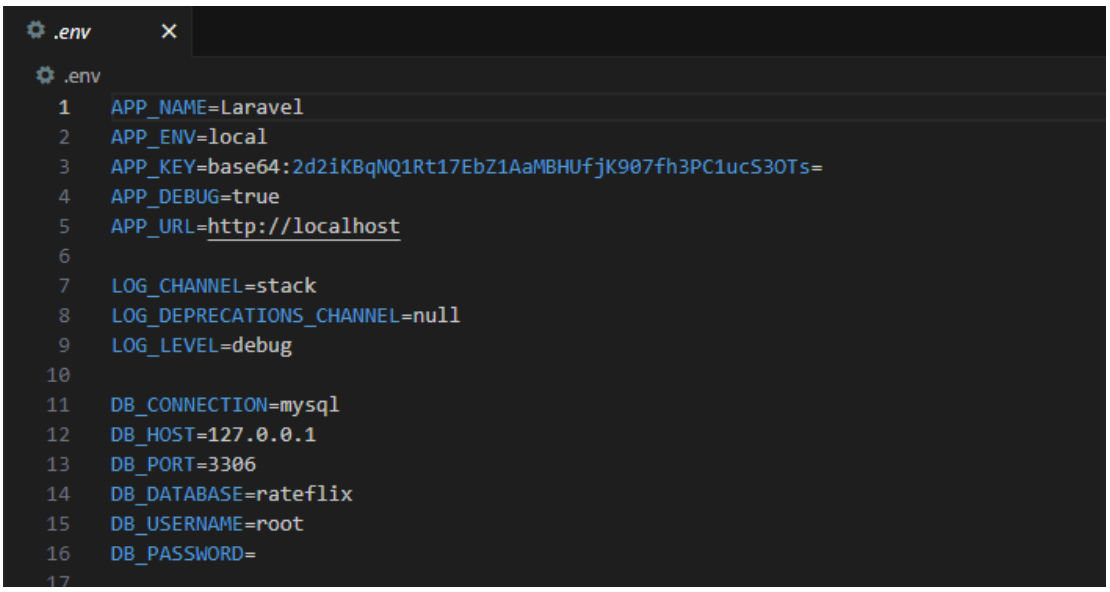
Para el desarrollo de este proyecto se necesitaron de diferentes controladores según la necesidad de cada vista, teniendo como principales *ComentarioController*, *FavoritoController*, *PeliculaController*, *TopController* y el de ejemplo de la imagen, *HomeController*:





b. Conectar con la base de datos

Existe un archivo de nombre *.env*, el cual tiene las credenciales de conexión a la base de datos, conteniendo lo siguiente:



Del mismo modo, en el archivo *config/database.php* se usan las variables antes mostradas para establecer la conexión, del siguiente modo:

c. Conexión de API para comunicación con el frontend

En el caso de este proyecto no se está usando una API para tener comunicación entre las estructuras, debido a que se implementan vistas tipo ".blade.php" y no se tienen formularios tradicionales. Por lo que no se requiere de dicho recurso para la comunicación de frontend y backend, todo se maneja por Laravel directamente de las rutas web.

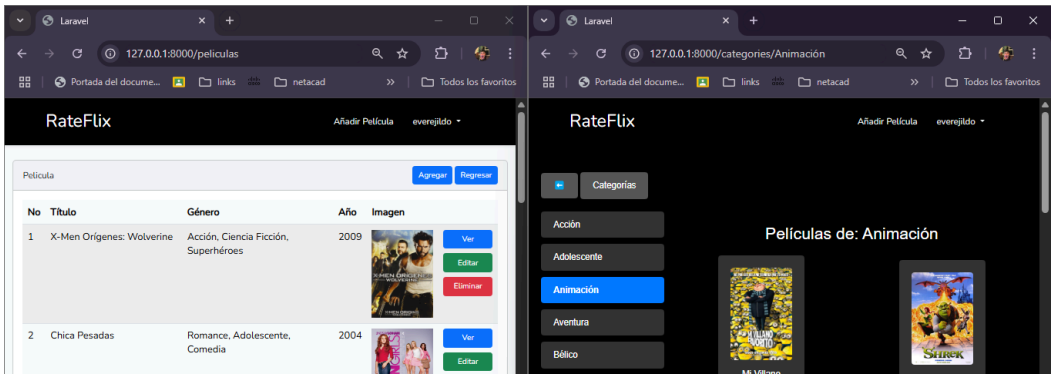
Implementar seguridad

a. Autenticación

- **Middleware**: en Laravel se usa auth para proteger las rutas, lo cual asegura que los usuarios tengan que autenticarse para acceder a ciertas vistas y funcionalidades. Como ejemplo se tiene el siguiente middleware, el cual se encarga de verificar la autenticación del usuario, siendo esto falso lo redirige al inicio de sesión.

```php
// Perfil
Route::middleware(['auth'])->group(function () {
    Route::get('/perfil', [ProfileController::class, 'index'])->name('perfil');
    Route::post('/perfil/update', [ProfileController::class, 'update'])->name('perfil.update');
});
```

- **Uso de auth**: dentro de este proyecto es necesario estar autenticado, o bien, tener una cuenta e iniciar sesión para poder realizar cualquier cosa que no sea iniciar sesión o crear cuenta. Por lo que en la mayoría de las vistas se implementa el uso de auth, principalmente por que en todas aparece la barra de navegación.

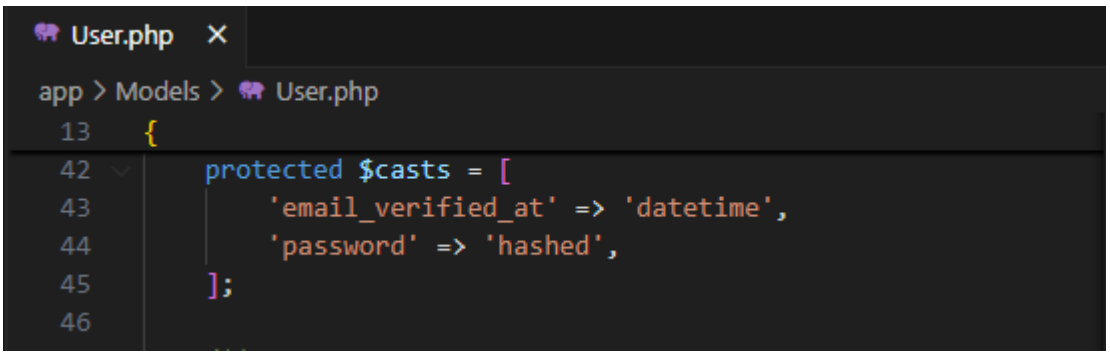


b. Cifrado de contraseñas

Laravel nunca guarda contraseñas en texto plano. Utiliza el algoritmo bcrypt para almacenarlas de forma segura. Este cifrado es unidireccional y resistente a ataques de fuerza bruta.

La línea *'password' => 'hashed'* en el *$casts* array del modelo *User* se usa para que Laravel automatice el proceso de cifrado seguro de las contraseñas antes de almacenarlas en la base de datos, lo cual es esencial para la seguridad de las aplicaciones, ya que sin esta

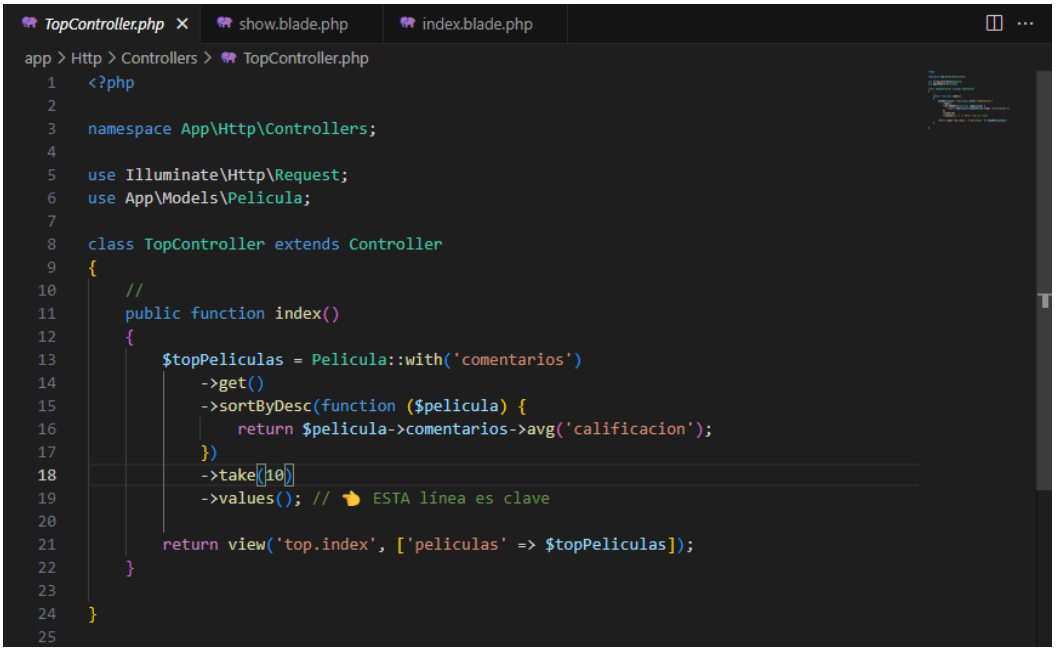
configuración, las contraseñas se almacenarían en texto plano, lo que sería una vulnerabilidad crítica.



c. Protección contra inyecciones SQL y XSS

- **Protección contra XSS:** por escapado automático de blade en Laravel automáticamente escapa el contenido, así que si un usuario intenta inyectar algo como *<script>alert('xss')</script>*, simplemente se mostrará en texto plano.

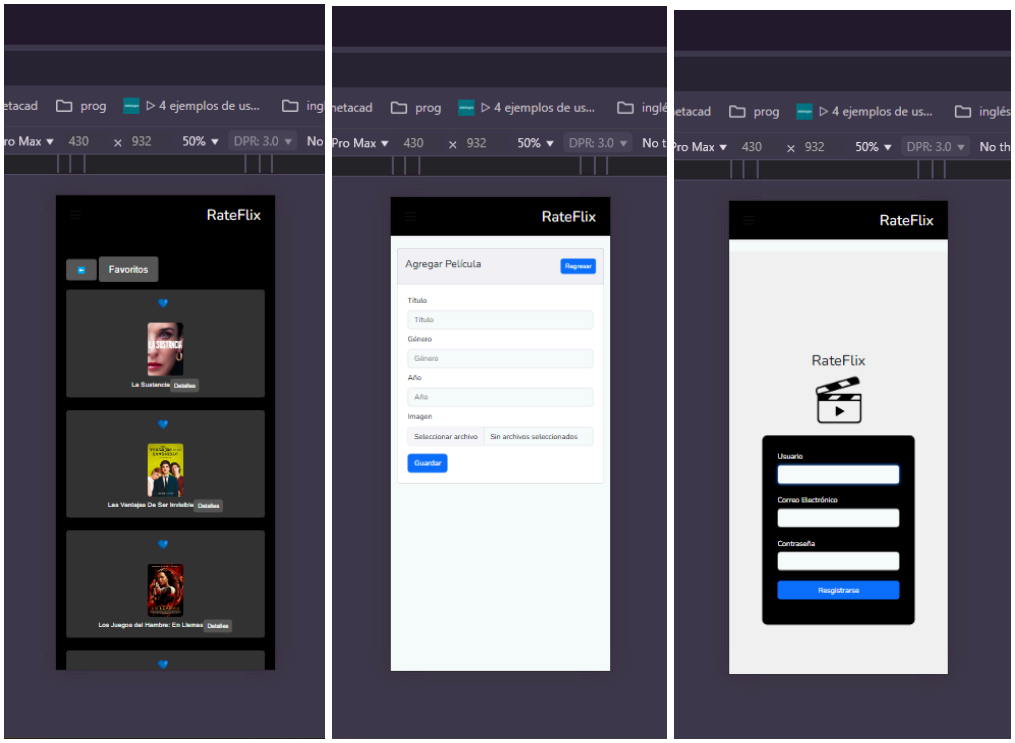
- **Protección contra Inyecciones SQL con Eloquent:** Esto se encuentra en métodos tipo `index()` o `show()` del controlador de comentarios. El uso de Eloquent y Query Builder en tus controladores protege automáticamente contra inyecciones SQL.



Desarrollo del frontend

- **Implementar el diseño responsive con CSS y librerías**

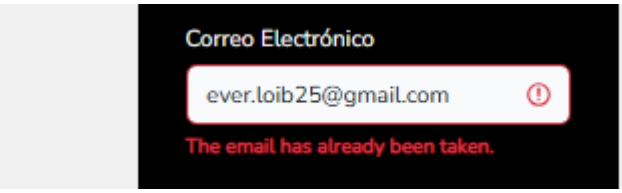
  En estas vistas se muestran algunas secciones del sitio web RateFlix, donde se puede ver cómo se adapta a dispositivos móviles mediante el diseño responsive. Para llevarlo a cabo se utilizó Bootstrap junto con estilos personalizados en CSS, además de la integración de la librería de Google Fonts para mejorar la tipografía y presentación visual.



- **Validar formularios**

  La validación de registro asegura la unicidad de las cuentas, verificando que la dirección de correo electrónico proporcionada no esté ya asociada a un usuario existente en nuestra base de datos. Este proceso, implementado en el controlador mediante las reglas de validación de Laravel, garantiza que cada usuario se registre con una dirección de correo electrónico distinta, manteniendo la integridad y seguridad de la plataforma. Como ejemplo se muestra el método *validator()* del *RegisterController*, las líneas esas son las responsables de verificar que el correo electrónico sea único en la tabla de usuarios:

```
RegisterController.php
app > Http > Controllers > Auth > RegisterController.php
12    {
43        /**
48        | */
49        protected function validator(array $data)
50        {
51            return Validator::make($data, [
52                'name' => ['required', 'string', 'max:255'],
53                'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
54                'password' => ['required', 'string', 'min:8'], // Se eliminó 'confirmed'
55            ]);
56        }
57
```

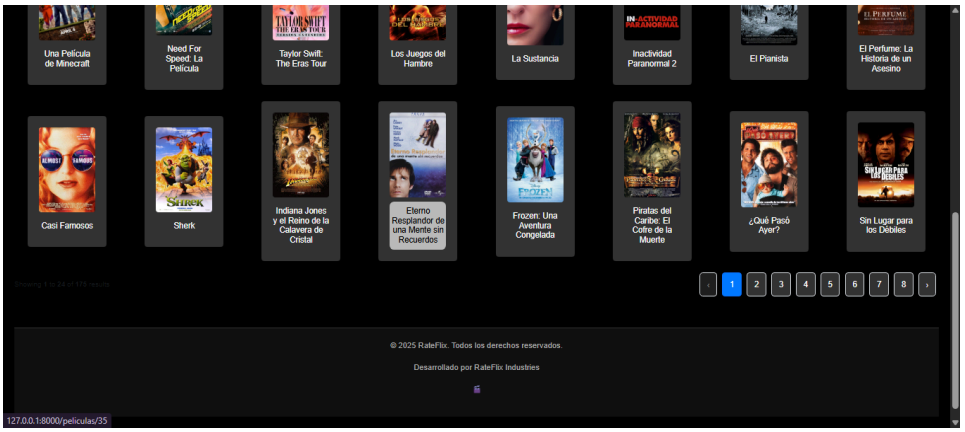


Pruebas y optimización

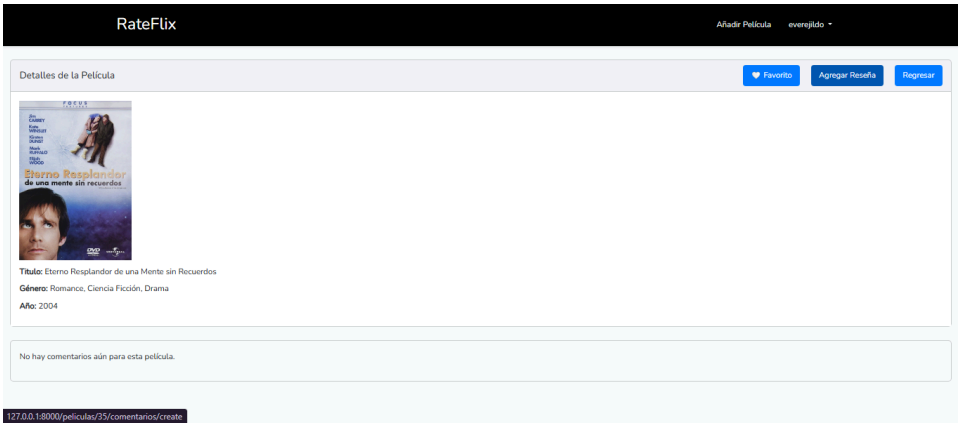
En la siguiente secuencia de imágenes se muestra el proceso de prueba para agregar reseñas a alguna película dentro del sitio:

1. Iniciar sesión

2. Seleccionar la película a reseñar

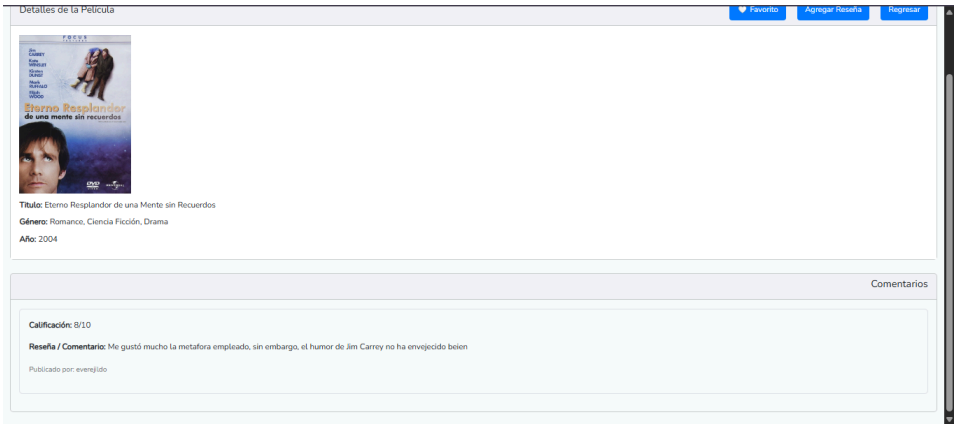


3. Dar clic en “Agregar Reseña”



4. Ingresar el comentario y la calificación, y dar clic en “Guardar Reseña”

5. Finalmente se muestra el comentario dentro de dicha sección de donde se muestra la película



Después de esto, se puede notar que cada función hace lo que debe, teniendo una estructura lineal, desde el iniciar sesión hasta el ver la reseña realizada y mostrándose según se especifica.

Conclusión

El desarrollo del proyecto ha demostrado la capacidad de existir una plataforma moderna, intuitiva y de seguridad para compartir comentarios sobre películas. El uso de Laravel en la parte auxiliar ha permitido una estructura sólida, clara y efectiva, mientras que Bootstrap y herramientas adicionales facilitan una reacción y un diseño agradable. Se implementa el sistema auténtico, así como las medidas de seguridad, garantiza la protección de datos y la integridad del contenido.

Rate Flix logra su objetivo principal: desarrollar una comunidad en línea donde los usuarios pueden expresar sus opiniones y explorar nuevas películas basadas en evaluaciones reales.

Este proyecto no solo fortalece las habilidades técnicas del grupo de desarrolladores, sino que también proporciona una solución digital con impactos sociales y culturales.