

Programación III - Proyecto Final



Integrantes:

- ❖ Álvarez, Verónica Belén
- ❖ Cabrera, Yanina Estela
- ❖ Figueredo, Julia Valeria

Profesores:

- ❖ Aguirre, Fernando
- ❖ Crozy, German

Fecha de Presentación:

07/07/2025

Descripción:

El presente proyecto tiene como finalidad el desarrollo de una plataforma web de gestión de turnos para una clínica dermatológica ficticia denominada "**Beyava**", la cual ofrece servicios estéticos y de cuidado de la piel. Esta plataforma está orientada principalmente a los pacientes, quienes podrán registrarse, iniciar sesión y acceder a un panel personalizado.

Desde su cuenta, los pacientes tendrán la posibilidad de:

- ❖ Consultar un listado actualizado de médicos especialistas.
- ❖ Ver información detallada sobre tratamientos ofrecidos.
- ❖ Reservar turnos disponibles de manera simple e intuitiva.
- ❖ Visualizar imágenes relacionadas con los tratamientos, como parte del material promocional e informativo.

Previo al inicio de sesión, el sitio mostrará una página principal de bienvenida, con el nombre y logo de la clínica, junto a imágenes representativas del rubro estético y dermatológico, para transmitir profesionalismo y estética visual. Este sistema busca facilitar la organización de turnos, brindar acceso ágil a la información y mejorar la experiencia del paciente en su interacción con la clínica.

Objetivo General:

Desarrollar una plataforma web para la clínica dermatológica "**Beyava**" que permita gestionar turnos en línea, ofreciendo a los pacientes un entorno interactivo donde puedan visualizar información médica y estética, reservar turnos, y conocer más sobre los tratamientos y productos disponibles.

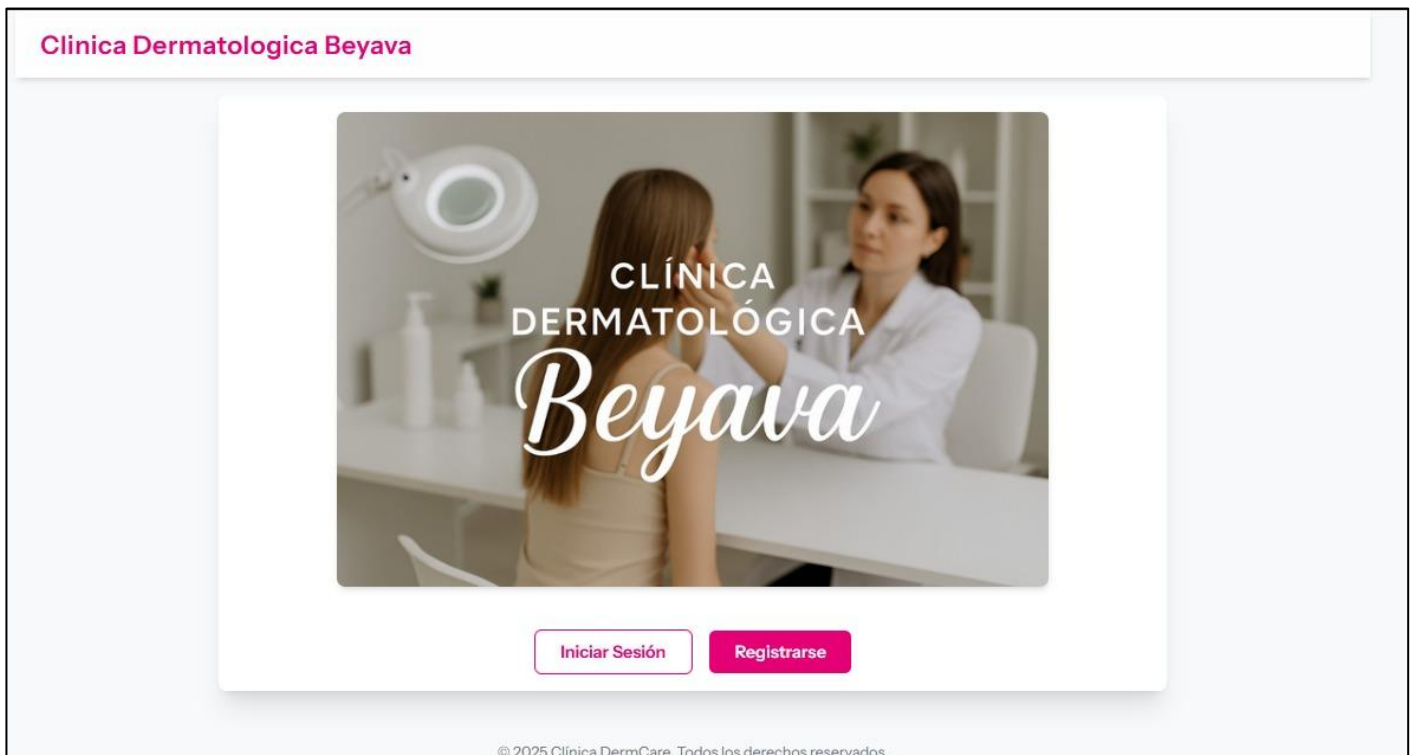
Objetivos Específicos:

- ❖ Diseñar una interfaz amigable y estética que represente la identidad visual de la clínica Beyava.
- ❖ Implementar un sistema de registro e inicio de sesión para pacientes, con almacenamiento seguro en base de datos.
- ❖ Desarrollar una sección interna para pacientes donde puedan: Consultar médicos disponibles, Ver tratamientos y descripciones, Reservar turnos online.

- ❖ Acceder a imágenes informativas sobre productos utilizados.
- ❖ Crear una página de inicio pública con el logo, nombre e imágenes que reflejen los servicios estéticos de la clínica.
- ❖ Facilitar la navegación y la experiencia de usuario mediante un diseño claro, moderno y adaptado a distintos dispositivos.

Se fueron cambiando los nombres del consultorio, por último, de Optó por este nombre **“Clínica Dermatológica Beyava”**.

Este es Nuestro Inicio de Pagina Web



En primer lugar, se creó el proyecto base con Laravel y se configuró el archivo .env para conectar correctamente la aplicación con la base de datos. Luego se diseñó la estructura de la base de datos, creando tablas para entidades como pacientes, médicos, turnos y tratamientos.

Posteriormente, se generaron los modelos, migraciones y controladores correspondientes a cada entidad utilizando los comandos de Artisan. Esto permitió definir la lógica de negocio, la estructura de datos y las rutas necesarias para la interacción entre el usuario y la base de datos.

1. Creación del Proyecto:

```
MINGW64/c:/laragon/www/centro-estetico

INTEL W10PRO@Leo-jm23 MINGW64 /c:/laragon/www/centro-estetico
$ composer create-project laravel/laravel centro-estetico
Creating a "laravel/laravel" project at "./centro-estetico"
Installing laravel/laravel (v12.0.11)
- Downloading laravel/laravel (v12.0.11)
- Installing laravel/laravel (v12.0.11): Extracting archive
Created project in C:\laragon\www\centro-estetico\centro-estetico
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 110 installs, 0 updates, 0 removals
- Locking brick/math (0.13.1)
- Locking carbonphp/carbon-doctrine-types (3.2.0)
- Locking dflydev/dot-access-data (v3.0.3)
- Locking doctrine/inflector (2.0.10)
- Locking doctrine/lexer (3.0.1)
- Locking dragonmantank/cron-expression (v3.4.0)
- Locking egulias/email-validator (4.0.4)
- Locking fakerphp/faker (v1.24.1)
- Locking filp/whoops (2.18.3)
- Locking fruitcake/php-cors (v1.3.0)
- Locking graham-campbell/result-type (v1.1.3)
- Locking guzzlehttp/guzzle (7.9.3)
- Locking guzzlehttp/promises (2.2.0)
- Locking guzzlehttp/psr7 (2.7.1)
- Locking guzzlehttp/uri-template (v1.0.4)
- Locking hamcrest/hamcrest-php (v2.1.1)
- Locking laravel/framework (v12.19.3)
- Locking laravel/pail (v1.2.3)
- Locking laravel/pint (v1.22.1)
- Locking laravel/prompts (v0.3.5)
- Locking laravel/sail (v1.43.1)
- Locking laravel/serializable-closure (v2.0.4)
- Locking laravel/tinker (v2.10.1)
- Locking league/commonmark (2.7.0)
- Locking league/config (v1.2.0)
- Locking league/flysystem (3.30.0)
- Locking league/flysystem-local (3.30.0)
- Locking league/mime-type-detection (1.16.0)
- Locking league/uri (7.5.1)
- Locking league/uri-interfaces (7.5.0)
- Locking mockery/mockery (1.6.12)
- Locking monolog/monolog (3.9.0)
- Locking myclabs/deep-copy (1.13.1)
- Locking nesbot/carbon (3.10.1)
- Locking nette/schema (v1.3.2)
- Locking nette/utils (v4.0.7)
- Locking nikic/php-parser (v5.5.0)
```

2. Cambio en el env.

```

1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:MtWzt0SwjnWoMtuGyhkDTfqaJL4g4iZbcweWSPpUYy=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 APP_LOCALE=en
8 APP_FALLBACK_LOCALE=en
9 APP_FAKER_LOCALE=en_US
10
11 APP_MAINTENANCE_DRIVER=file
12 # APP_MAINTENANCE_STORE=database
13
14 PHP_CLI_SERVER_WORKERS=4
15
16

```

Would you like to create it? (yes/no) [yes]

3. Muestra de la Base de Datos:

HeidiSQL 12.8.0.6908 - Administrador de sesiones: Laragon.MySQL

Nombre de la sesión: Laragon.MySQL * Host: 127.0...

Tipo de red: MariaDB or MySQL (TCP/IP)

Librería: libmariadb.dll

Nombre del host / IP: 127.0.0.1

Usuario: root

Contraseña:

Puerto: 3306

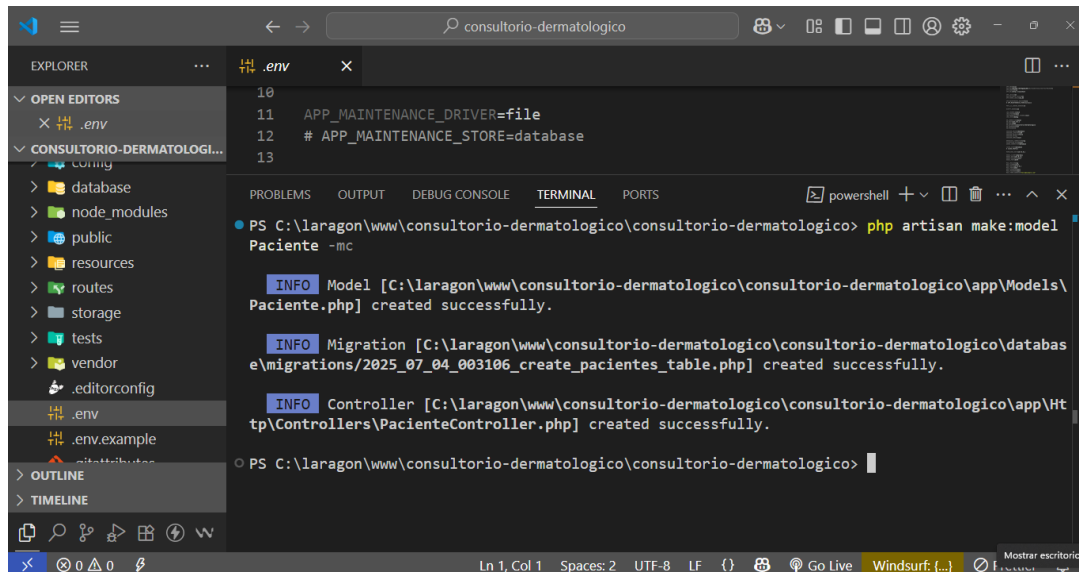
Bases de datos: centro-estetico

Comentario:

Laragon.MySQL\centro-estetico\ - HeidiSQL Portable 12.8.0.6908

Nombre	Filas	Tamaño	Creado	Actualizado	Motor	Comen
cache	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
cache_locks	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
failed_jobs	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
jobs	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
job_batches	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
medicos	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
migrations	9	16,0 KiB	2025-07-04 23:19:56		InnoDB	
pacientes	2	32,0 KiB	2025-07-04 23:19:58		InnoDB	
password_reset...	0	16,0 KiB	2025-07-04 23:19:57		InnoDB	
productos	0	16,0 KiB	2025-07-04 23:19:59		InnoDB	
regalos	0	48,0 KiB	2025-07-04 23:20:01		InnoDB	
sessions	1	48,0 KiB	2025-07-04 23:19:58		InnoDB	
tratamientos	0	16,0 KiB	2025-07-04 23:19:58		InnoDB	
turnos	0	80,0 KiB	2025-07-04 23:20:00		InnoDB	
users	0	32,0 KiB	2025-07-05 15:43:52		InnoDB	

4. Creación de los Modelos-Migraciones-Controladores:



The screenshot shows the Visual Studio Code interface with the Explorer view on the left displaying the project structure of 'consultorio-dermatologico'. The main editor shows the '.env' file with configuration for APP_MAINTENANCE_DRIVER=file and APP_MAINTENANCE_STORE=database. The TERMINAL view at the bottom shows the execution of 'php artisan make:model Paciente -mc', which successfully creates the model, migration, and controller.

```
PS C:\laragon\www\consultorio-dermatologico\consultorio-dermatologico> php artisan make:model Paciente -mc

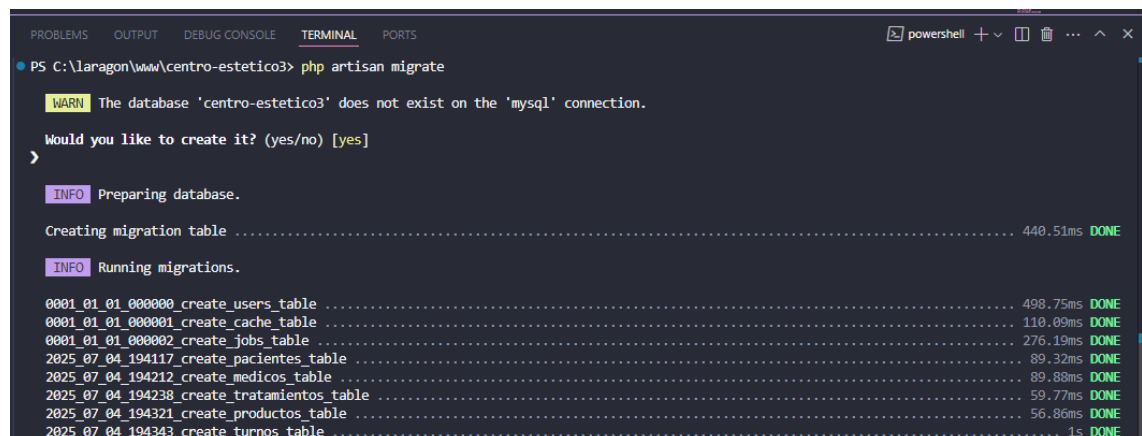
INFO Model [C:\laragon\www\consultorio-dermatologico\consultorio-dermatologico\app\Models\Paciente.php] created successfully.

INFO Migration [C:\laragon\www\consultorio-dermatologico\consultorio-dermatologico\database\Migrations\2025_07_04_003106_create_pacientes_table.php] created successfully.

INFO Controller [C:\laragon\www\consultorio-dermatologico\consultorio-dermatologico\app\Http\Controllers\PacienteController.php] created successfully.

PS C:\laragon\www\consultorio-dermatologico\consultorio-dermatologico>
```

5. Migraciones y Base de Datos Corridas:



The screenshot shows the Visual Studio Code terminal with the command 'php artisan migrate' executed. It displays a warning that the database 'centro-estetico3' does not exist, followed by a confirmation to create it. The terminal then shows the successful execution of the migration, creating the database and running the migrations.

```
PS C:\laragon\www\centro-estetico3> php artisan migrate

WARN The database 'centro-estetico3' does not exist on the 'mysql' connection.

Would you like to create it? (yes/no) [yes]
>

INFO Preparing database.

Creating migration table ..... 448.51ms DONE

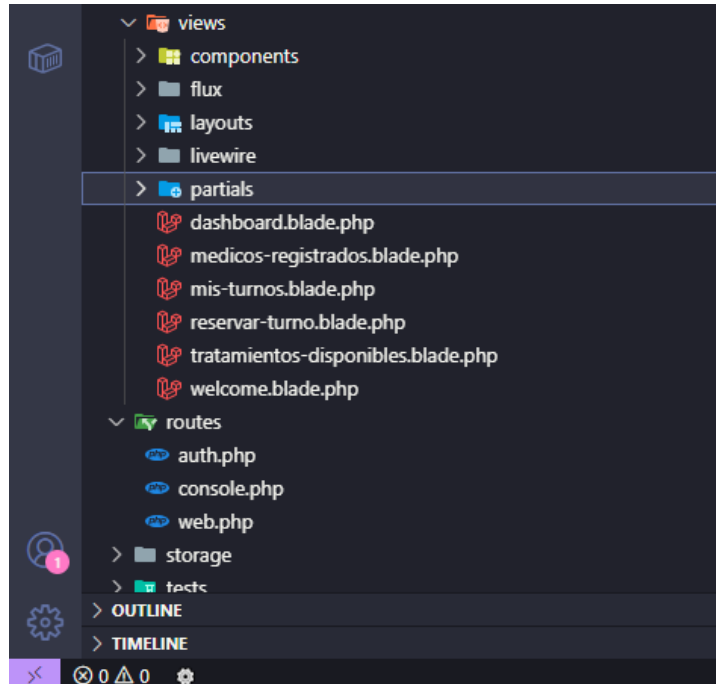
INFO Running migrations.

0001_01_01_000000_create_users_table ..... 498.75ms DONE
0001_01_01_000001_create_cache_table ..... 110.09ms DONE
0001_01_01_000002_create_jobs_table ..... 276.19ms DONE
2025_07_04_194117_create_pacientes_table ..... 89.32ms DONE
2025_07_04_194212_create_medicos_table ..... 89.88ms DONE
2025_07_04_194239_create_tratamientos_table ..... 59.77ms DONE
2025_07_04_194321_create_productos_table ..... 56.86ms DONE
2025_07_04_194343_create_turnos_table ..... 1s DONE
```

Una vez definidas las migraciones, se comenzaron a desarrollar las vistas con Blade y Livewire juntos con las Rutas que se utilizaron, creando interfaces dinámicas y amigables para el usuario, incluyendo:

- **Una página de inicio pública**, donde se muestra el logo de la clínica, imágenes representativas del rubro estético y accesos a registro e inicio de sesión.
- **Un panel interno para pacientes**, donde pueden ver médicos disponibles, tratamientos ofrecidos, productos destacados y turnos reservables.

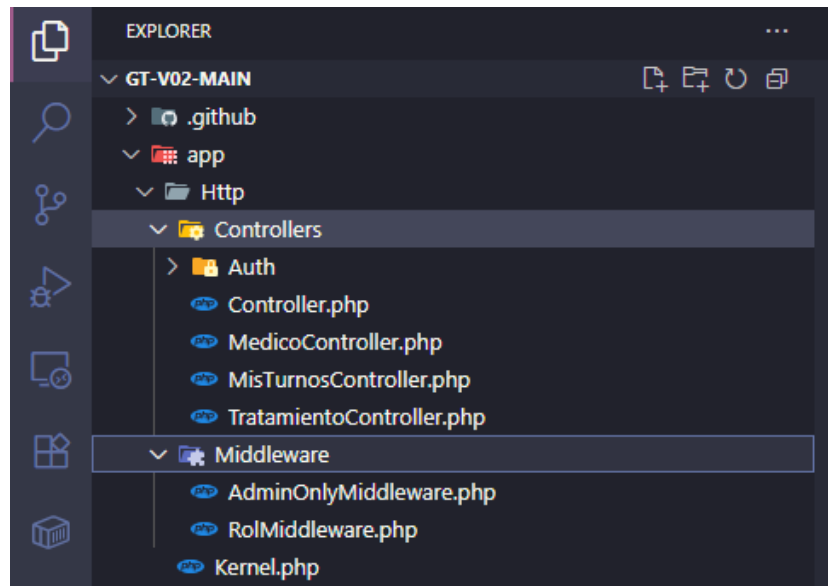
6. Vistas que Utilizamos con Blade:



Ejemplo de una de las Vistas

```
mis-turnos.blade.php
resources > views > mis-turnos.blade.php > ...
1 <@x-layouts.app :title="__('Mis Turnos')">
2 <div class="p-6">
3 <h1 class="text-2xl font-bold mb-4">Mis Turnos</h1>
4 @if($turnos->isEmpty())
5 <div class="mb-2 p-2 bg-blue-100 text-blue-800 rounded">No estás inscrito en ningún turno.</div>
6 @else
7 <div class="space-y-4">
8 @foreach($turnos as $turno)
9 <div class="p-4 border rounded-xl bg-white flex justify-between items-center">
10 <div>
11 <div class="font-semibold text-lg text-gray-800">
12 {{ \Carbon\Carbon::parse($turno->fecha)->format('d/m/Y') }}
13 de {{ \Carbon\Carbon::parse($turno->hora_inicio)->format('H:i') }}
14 a {{ \Carbon\Carbon::parse($turno->hora_fin)->format('H:i') }}
15 </div>
16 <div class="text-sm text-gray-600">
17 Estado: <span class="font-medium @if($turno->estado == 'reservado') text-red-600 @else text-green-600 @endif">{{ ucfirst($turno->estado) }}</span>
18 </div>
19 @if($turno->tratamiento)
20 <div class="text-sm text-gray-600">
21 Tratamiento: {{ $turno->tratamiento->nombre }}
22 </div>
23 @endif
24 @if($turno->medico)
25 <div class="text-sm text-gray-600">
26 Médico: {{ $turno->medico->name }} {{ $turno->medico->last_name }}
27 </div>
28 @endif
29 <div class="text-sm text-gray-500 mt-1">
30 Comentario: {{ $turno->comentario ?? 'N/A' }}
31 </div>
32 </div>
33 <form method="POST" action="{{ route('turno-eliminar', $turno->id) }}" onsubmit="return confirm('¿Seguro que deseas eliminar tu inscripción a este turno?')">
34 @csrf
35 @method('DELETE')
36 <x-flux-button type="submit" color="danger">Eliminar</x-flux-button>
37 </form>
38 </div>
39 @endforeach
40 </div>
41 @endif
42 </div>
43 </x-layouts.app>
44
```

7. Controladores que se utilizaron y el Middleware:



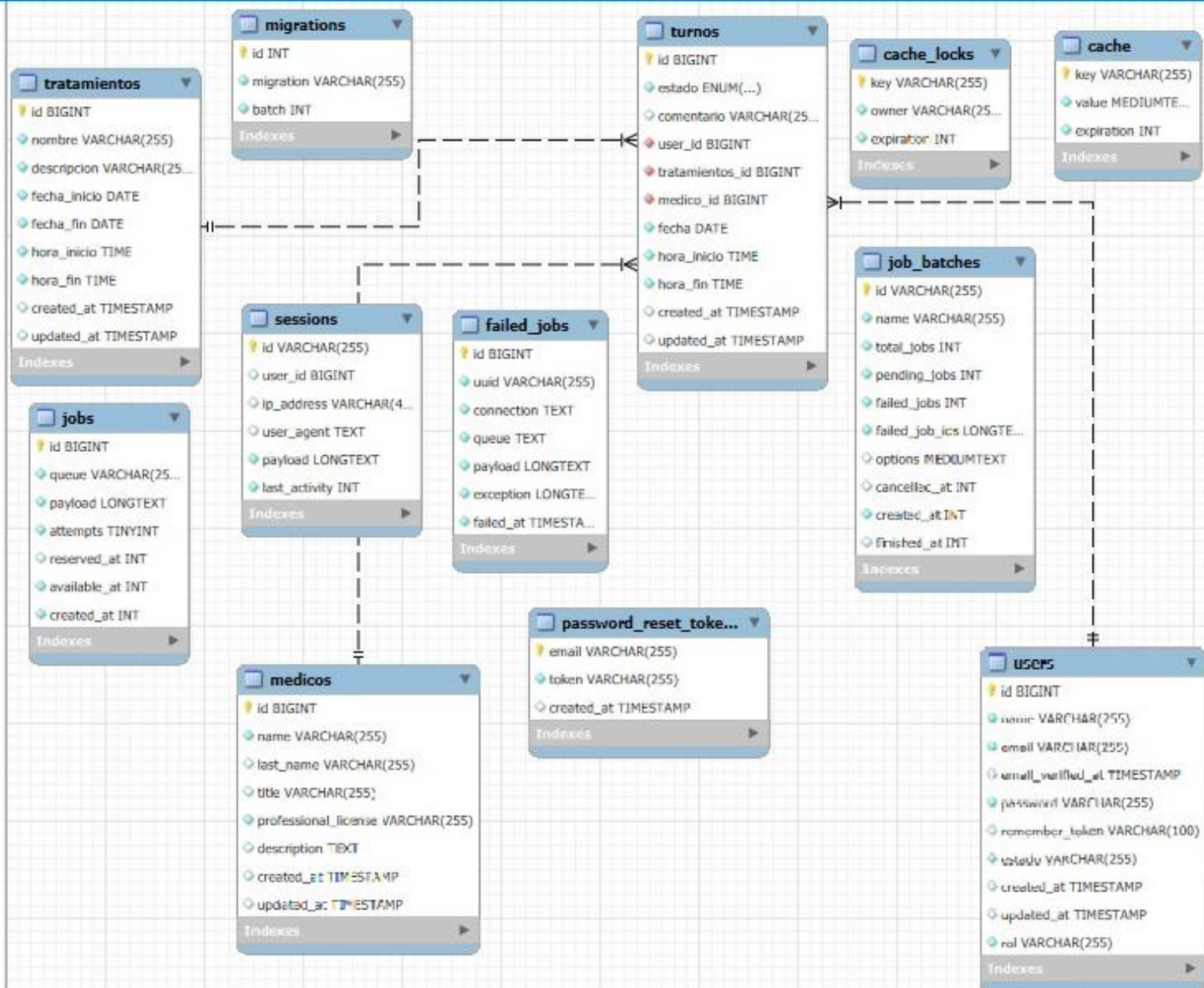
Ejemplo de uno de los controladores

```
MisTurnosController.php X
app > Http > Controllers > MisTurnosController.php > MisTurnosController > ReservarTurno
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Support\Facades\Auth;
7 use App\Models\Turnos; // Cambiamos de Taller a Turno
8 use App\Models\User;
9 use App\Models\Medico;
10 use App\Models\Tratamiento;
11 use Illuminate\Validation\ValidationException;
12
13 class MisTurnosController extends Controller
14 {
15
16     public function index()
17     {
18         // Usar la relación correcta 'turnos' en el modelo User
19         // Asegúrate de que el modelo User tiene un método 'turnos()' que define la relación
20         $turnos = Auth::user()->turnos ?? collect(); // Asume que un usuario puede tener muchos turnos
21         //var_dump($turnos);die();
22         return view('mis-turnos', compact('turnos')); // Cambiamos a 'mis-turnos' para la vista
23     }
24
25     public function ReservarTurno ()
26     {
27         $medicos = Medico::all();
28         $tratamientos = Tratamiento::all();
29
30         return view('reservar-turno', compact('medicos', 'tratamientos'));
31     }
32
33     public function store(Request $request)
34     {
35         try {
36             // Reglas de validación para los campos del formulario
37             $validatedData = $request->validate([
38                 'medico_id' => 'required|exists:medicos,id',
39                 'tratamientos_id' => 'required|exists:tratamientos,id',
40                 'fecha' => 'required|date|after_or_equal:today', // La fecha no puede ser anterior a hoy
41                 'hora_inicio' => 'required|date_format:H:i',
42                 'hora_fin' => 'required|date_format:H:i|after:hora_inicio', // La hora de fin debe ser posterior a la de inicio
43                 'comentario' => 'nullable|string|max:500',
44             ]);
45
46             $user = Auth::user(); // Obtener el usuario autenticado
47
48         }
```


8. Rutas que se Utilizaran:

```
routes > web.php > ...
1  <?php
2
3  use App\Livewire\Settings\Appearance;
4  use App\Livewire\Settings>Password;
5  use App\Livewire\Settings\Profile;
6  use Illuminate\Support\Facades\Route;
7  use App\Livewire\Admin\TurnosAdmin;
8  use App\Livewire\Admin\UsuariosAdmin;
9  use App\Http\Controllers\MedicoController;
10 use App\Http\Controllers\TratamientoController;
11
12 Route::resource('medicos', MedicoController::class);
13 Route::get('/', function () {
14     return view('welcome');
15 }->name('home');
16
17 Route::view('dashboard', 'dashboard')
18     ->middleware(['auth', 'verified'])
19     ->name('dashboard');
20
21 Route::middleware(['auth'])->group(function () {
22     Route::redirect('settings', 'settings/profile');
23
24     Route::get('settings/profile', Profile::class)->name('settings.profile');
25     Route::get('settings/password', Password::class)->name('settings.password');
26     Route::get('settings/appearance', Appearance::class)->name('settings.appearance');
27     Route::get('mis-turnos', [App\Http\Controllers\MisTurnosController::class, 'index']->name('mis-turnos');
28
29     Route::get('/reservar-turno', [App\Http\Controllers\MisTurnosController::class, 'ReservarTurno']->name('reservar-turno');
30     Route::post('/reservar-turno', [App\Http\Controllers\MisTurnosController::class, 'store']->name('turnos.store');
31
32     Route::delete('/mis-turnos/{turno}', [App\Http\Controllers\MisTurnosController::class, 'destroy']->name('turno-eliminar');
33
34     // Rutas para la gestión de médicos
35     Route::middleware(['auth'])->group(function () { // Opcional: proteger estas rutas con autenticación
36         Route::get('/medicos-registrados', [App\Http\Controllers\MedicoController::class, 'index']->name('medicos-registrados');
37         //Route::post('/medicos-registrados', [App\Http\Controllers\MedicoController::class, 'store']->name('medicos.store');
38     });
39
40     //ruta medicos
41     Route::get('/medicos-registrados', [App\Http\Controllers\MedicoController::class, 'index']->name('medicos-registrados');
42
43     //ruta tratamientos
44     Route::get('/tratamientos-disponibles', [App\Http\Controllers\TratamientoController::class, 'index']->name('tratamientos-disponibles');
45
46 });
47
48
49 Route::middleware(['role:admin'])->group(function () {
50     // Rutas solo para admin
51 });
52
53 Route::middleware(['auth', 'can:admin'])->group(function () {
54 });
55
56 require __DIR__.'/auth.php';
57 Route::middleware(['auth', 'can:admin, App\Models\User'])->group(function () {
58     Route::get('/admin/turnos', TurnosAdmin::class)->name('admin.turnos');
59     Route::get('/admin/usuarios', UsuariosAdmin::class)->name('admin.usuarios');
60 });
```

Diagrama Entidad-Relación



Proyecto subido al GitHub

The screenshot shows the GitHub interface for a repository named 'ClinicaDermatologica' by user 'veronicabelen'. The repository is public and has 2 commits. The file list includes: .github/workflows, app, bootstrap, config, database, public, resources, routes, storage, tests, and .editorconfig. The right sidebar shows the 'About' section with no description, website, or topics provided. It also lists 'Releases' (none published), 'Packages' (none published), and 'Languages' (Blade 91.0%, PHP 8.9%).

veronicabelen / ClinicaDermatologica Public

<> Code Issues Pull requests Actions Projects Security Insights

main 1 Branch 0 Tags

Go to file

<> Code

About

No description, website, or topics provided.

Readme

Activity

0 stars

0 watching

0 forks

Report repository

Releases

No releases published

Packages

No packages published

Languages

Blade 91.0% PHP 8.9%

Conclusión

El desarrollo de este proyecto permitió aplicar conocimientos prácticos sobre Laravel, Livewire y la estructura MVC, creando un sistema funcional y visualmente atractivo para la gestión de turnos en una clínica dermatológica. A través de una interfaz intuitiva, los pacientes pueden registrarse, acceder a información médica relevante y gestionar sus propios turnos, lo que mejora la organización y eficiencia del servicio. La plataforma "Beyava" representa una solución tecnológica adaptada a las necesidades del sector estético, demostrando cómo el uso adecuado de herramientas modernas puede optimizar la experiencia tanto del usuario como del administrador del sistema.

Bibliografía:

- Link del GitHub:
<https://github.com/veronicabelen/ClinicaDermatologica>