

# DESARROLLO FULL STACK

## PROYECTO FINAL



- Miguel Armando Rascon Dominguez

## DOCUMENTACIÓN TÉCNICA: PROYECTO CRIMSON BLADES

### 1. ENLACES DEL PROYECTO

- **Repositorio GitHub (Monorepo):** <https://github.com/armandiucs114200-ui/barberia-fullstack>
  - **Despliegue Backend (Render):** <https://barber-cgfw.onrender.com>
  - **Despliegue Frontend (Vercel):** <https://barberia-fullstack.vercel.app>
- 

### 2. DESCRIPCIÓN GENERAL

**Crimson Blades** es una plataforma integral de gestión para una barbería premium. Permite a los clientes visualizar un catálogo dinámico de servicios, consultar el clima en tiempo real mediante una API externa y agendar citas. Para la administración, cuenta con un **Admin Command Center** que ofrece métricas financieras y control de flujo de clientes.

---

### 3. ARQUITECTURA Y TECNOLOGÍAS

El sistema utiliza una arquitectura desacoplada de alto rendimiento:

- **Frontend:** Next.js 15 con Tailwind CSS para una interfaz reactiva y optimizada.
  - **Backend:** Node.js con Express.js, organizado bajo el patrón controlador-servicio.
  - **Base de Datos:** Supabase (PostgreSQL) con manejo de **Service Role Key** para operaciones administrativas seguras en el servidor.
  - **Integración:** WeatherAPI para datos meteorológicos dinámicos.
- 

### 4. SEGURIDAD Y AUTENTICACIÓN (JWT)

Se ha implementado un sistema de seguridad basado en **JSON Web Tokens (JWT)**:

1. **Generación:** Tras un login exitoso, el servidor firma un token único.
2. **Persistencia:** El frontend almacena este token de forma segura para mantener la sesión.

3. **Middleware:** Todas las rutas sensibles del backend (como la actualización de estados de citas) requieren que el token sea validado antes de procesar la petición.
4. **Aislamiento de Llaves:** Las llaves maestras de Supabase se mantienen exclusivamente en las variables de entorno del servidor (Render), nunca en el cliente.

## 5. ENDPOINTS PRINCIPALES

Método	Ruta	Acceso	Función
POST	/api/auth/login	Público	Autenticación de usuario.
GET	/api/equipo	Público	Obtiene los barberos para el catálogo.
POST	/api/reservas	Privado	Crea una reserva vinculada a un usuario.
PATCH	/api/reservas/:id	Admin	Cambia el estado de la cita (Completado/Cancelado).
GET	/api/weather	Público	Retorna el clima actual (vía WeatherAPI).

## 8. GUÍA DE INSTALACIÓN LOCAL

Para ejecutar este proyecto localmente, requiere configurar las siguientes variables de entorno:

### En backend/.env:

Plaintext

PORT=4000

SUPABASE\_URL=tu\_url\_de\_supabase

SUPABASE\_KEY=tu\_service\_role\_key

WEATHER\_API\_KEY=tu\_weather\_key

JWT\_SECRET=tu\_clave\_secreta

### En frontend/.env.local:

Plaintext

NEXT\_PUBLIC\_API\_URL=http://localhost:4000/api

CREDENCIALES DE PRUEBA

\* \*\*Admin:\*\* `admin@BARBERIA.com` / `12345678`

GET http://localh

POST Login

Barber Shop

No environment

Barber Shop API Docs / Auth / Login

POST

http://localhost:4000/api/auth/login

Send

Docs

Params

Auth

Headers (9)

Body

Scripts

Settings

Cookies

raw

JSON

Schema

Beautify

```
1 {
2   "email": "admin@barberia.com",
3   "password": "12345678"
4 }
```

Body

200 OK

1.63 s

1.24 KB

Preview

Visualize

JSON

Preview

Visualize

```
1 {
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6ImYwMTQxNjZhLWUwOWh0NDU1YS1hYzI2LTdhZjEyNDVhNWZhNyIsImVtYWlsIjoieWRtaW5AYmFyYmVyaWEuY29tIiwicm9sZSI6InVzdWYyaW8iLCJpYXQiOiE3NzIyODY1MzksImV4cCI6MTc3MjMxNTMzOX0.ZuqTXMnK3SSeq-R_JVp3BPHYz8TTzFX5RrXWYjFA4l4",
3   "user": {
4     "id": "f014166a-a09b-455a-ac26-7af1245a5aa7",
5     "email": "admin@barberia.com",
6     "role": "usuario"
7   }
8 }
```





