Manual Técnico
Luis Regalado 202131920
Salón De Belleza

Software:

- **Lenguajes y Frameworks:**
  - **Java:**
    Lenguaje de programación orientado a objetos, usado en el desarrollo del backend para implementar la lógica de negocio del sistema. Ofrece alto rendimiento, seguridad y una arquitectura robusta..

  - **Angular:**
    Framework de desarrollo web basado en TypeScript, utilizado para construir la interfaz de usuario del sistema. Facilita la creación de aplicaciones dinámicas, reactivas y de una sola página (SPA), con una arquitectura basada en componentes.

  - **TypeScript:**
    Aunque comúnmente asociado al frontend, también fue utilizado en el backend para tareas específicas del proyecto. Su tipado estático y orientación a objetos facilita el mantenimiento del código.

  - **SCSS (Sassy CSS):**
    Extensión de CSS que permite escribir hojas de estilo más organizadas y reutilizables. En este proyecto se usó para definir estilos del backend que interactúan o complementan componentes del frontend.

- **Base de Datos: MySQL Workbench**
  Herramienta visual para diseño, modelado y administración de bases de datos MySQL. Se utilizó para gestionar la base de datos del sistema, incluyendo creación de esquemas, tablas y consultas.

Sistema Operativo:

- **Windows 10:**
  Sistema operativo de Microsoft ampliamente usado, compatible con una gran variedad de software de desarrollo. Proporciona una interfaz amigable y soporte para múltiples entornos de programación.

- **Fedora 40:**
  Distribución de Linux moderna y enfocada en desarrolladores. Ofrece acceso a software actualizado, herramientas de línea de comandos y gran estabilidad para el desarrollo de software**.**

- **IDE:**
  - **Visual Studio Code:**
    Editor de código fuente ligero y extensible, ideal para desarrollo frontend. Fue utilizado como entorno de desarrollo para programar en Angular, SCSS y TypeScript, gracias a sus múltiples extensiones y soporte para tecnologías web modernas.

  - **NetBeans:**
    Entorno de desarrollo integrado (IDE) empleado para la creación de la lógica del backend en Java. Proporciona herramientas integradas para la depuración, diseño y gestión de proyectos Java.

- **GITHUB**

Es un sistema de control de versiones de código y gestión de proyectos, a su vez también funciona como una plataforma de estilo de red social

Estructura del Proyecto

General:

Project2_IPC2/
├── salon-app/ → Frontend (Angular y VisualStudio Code)
├── salonDeBelleza/ → Backend (Java, con NetBeans)

Frontend:

salon-app/
├── .idea/ → Configuraciones del IDE (IntelliJ/WebStorm, opcional).
├── .vscode/ → Configuraciones de VS Code (depuración, extensiones).
├── angular.json → Configuración del proyecto Angular.
├── package.json → Dependencias y scripts del proyecto.
├── tsconfig.json → Configuración de compilación TypeScript.
├── src/
│ ├── index.html → HTML principal de la app.
│ ├── main.ts → Punto de entrada de la aplicación Angular.
│ └── app/
│ ├── admin/ → Gestión de usuarios, panel de control para administradores.
│ ├── ads/ → Módulo de anuncios o contenido promocional.
│ ├── auth/ → Login, logout, validaciones y autenticación de usuarios.
│ ├── app-bar/ → Barra de navegación y elementos visuales globales.
│ ├── bienvenida/ → Página de inicio/bienvenida.
│ ├── cliente/ → Funcionalidades específicas para el rol cliente.
│ ├── core/ → Servicios compartidos, guardias, interceptores.
│ ├── empleado/ → Funcionalidades para el rol empleado (acciones específicas).
│ ├── reserva/ → Módulo de reservas (creación, consulta, modificación).
│ └── servicio/ → Gestión de servicios ofrecidos (peluquería, spa, etc.).

Backend:

salonDeBelleza/
├── nbproject/ → Configuraciones del proyecto para NetBeans.

```
├── src/
│   └── main/
│       ├── java/
│       │   └── com/salon/
│       │       ├── controller/   → Controladores REST que manejan las solicitudes HTTP.
│       │       ├── model/        → Entidades del sistema (Usuario, Servicio, Reserva, etc.).
│       │       └── service/      → Servicios que implementan la lógica de negocio.
│       └── resources/
│           ├── application.properties  → Configuraciones de la app (puerto, DB, etc.).
│           └── ...                → Otros recursos necesarios (mensajes, plantillas).
```

Mapeo Fisico de la Base de Datos

```
-- Tabla de Roles
CREATE TABLE roles (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL UNIQUE -- admin, marketing, servicio, empleado, cliente
);

-- Tabla de Usuarios (clientes, empleados, administradores, etc.)
CREATE TABLE users (
    id INT PRIMARY KEY AUTO_INCREMENT,
    email VARCHAR(100) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL, -- Almacenar hash
    dpi VARCHAR(20),
    phone VARCHAR(15),
    address VARCHAR(255),
    role_id INT NOT NULL,
    FOREIGN KEY (role_id) REFERENCES roles(id)
);

-- Tabla de Servicios
```

```
CREATE TABLE services (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    image_url VARCHAR(255),
    duration_min INT NOT NULL, -- Duración en minutos
    price DECIMAL(10, 2) NOT NULL
);

-- Tabla de Relación Empleados-Servicios (qué empleados ofrecen qué servicios)
CREATE TABLE employee_services (
    employee_id INT,
    service_id INT,
    PRIMARY KEY (employee_id, service_id),
    FOREIGN KEY (employee_id) REFERENCES users(id),
    FOREIGN KEY (service_id) REFERENCES services(id)
);

-- Tabla de Horarios de Empleados
CREATE TABLE employee_schedules (
    id INT PRIMARY KEY AUTO_INCREMENT,
    employee_id INT NOT NULL,
    day_of_week INT NOT NULL, -- 1 (Lunes) a 7 (Domingo)
    start_time TIME NOT NULL,
    end_time TIME NOT NULL,
    FOREIGN KEY (employee_id) REFERENCES users(id)
);

-- Tabla de Citas
CREATE TABLE appointments (
    id INT PRIMARY KEY AUTO_INCREMENT,
    client_id INT NOT NULL,
    employee_id INT NOT NULL,
    service_id INT NOT NULL,
    start_time DATETIME NOT NULL,
    end_time DATETIME NOT NULL,
```

```
    status ENUM('pendiente', 'confirmado', 'cancelado') DEFAULT
'pendiente',
    FOREIGN KEY (client_id) REFERENCES users(id),
    FOREIGN KEY (employee_id) REFERENCES users(id),
    FOREIGN KEY (service_id) REFERENCES services(id)
);

-- Tabla de Anuncios
CREATE TABLE advertisements (
    id INT PRIMARY KEY AUTO_INCREMENT,
    type ENUM('video', 'imagen', 'texto') NOT NULL,
    content_url VARCHAR(255) NOT NULL, -- Enlace al recurso
    start_date DATE NOT NULL,
    end_date DATE NOT NULL
);

-- Tabla de Catálogos de Servicios (PDF)
CREATE TABLE service_catalogs (
    id INT PRIMARY KEY AUTO_INCREMENT,
    service_id INT NOT NULL UNIQUE,
    pdf_url VARCHAR(255) NOT NULL,
    uploaded_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (service_id) REFERENCES services(id)
);

SELECT id,email,password,dpi,phone,address,role_id FROM users;

-- Tabla de perfiles de clientes
CREATE TABLE client_profiles (
    user_id INT PRIMARY KEY,
    photo_url VARCHAR(255),
    description TEXT,
    hobbies TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
```

```
)

-- Eliminar tabla antigua de horarios de empleados (si existe)
DROP TABLE IF EXISTS employee_schedules;

-- Crear tabla de horario general
CREATE TABLE salon_hours (
    id INT PRIMARY KEY AUTO_INCREMENT,
    day_of_week INT NOT NULL UNIQUE, -- 1 (Lunes) a 7 (Domingo)
    opening_time TIME NOT NULL,
    closing_time TIME NOT NULL,
    is_active BOOLEAN DEFAULT TRUE
);

-- Insertar días base
INSERT INTO salon_hours (day_of_week, opening_time, closing_time)
VALUES
(1, '08:00:00', '18:00:00'),
(2, '08:00:00', '18:00:00'),
(3, '08:00:00', '18:00:00'),
(4, '08:00:00', '18:00:00'),
(5, '08:00:00', '18:00:00'),
(6, '09:00:00', '14:00:00'),
(7, '09:00:00', '12:00:00');

-- Tabla de Facturas
CREATE TABLE invoices (
    id INT PRIMARY KEY AUTO_INCREMENT,
    appointment_id INT NOT NULL UNIQUE,
    total DECIMAL(10,2) NOT NULL,
    generated_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    payment_status ENUM('pendiente', 'pagado') DEFAULT 'pendiente',
    FOREIGN KEY (appointment_id) REFERENCES appointments(id)
);

-- Actualizar tabla de citas
ALTER TABLE appointments
```

```
ADD COLUMN no_show BOOLEAN DEFAULT FALSE,
ADD COLUMN invoice_id INT,
ADD CONSTRAINT fk_invoice FOREIGN KEY (invoice_id)
REFERENCES invoices(id);

-- Tabla de Lista Negra
CREATE TABLE blacklist (
    client_id INT PRIMARY KEY,
    reason TEXT,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (client_id) REFERENCES users(id)
);

insert into services (name, description, image_url, duration_min, price)
values
('Corte de Cabello', 'Corte de cabello para hombres y mujeres.',
'https://example.com/corte.jpg', 30, 150.00),
('Manicura', 'Servicio de manicura con esmalte.',
'https://example.com/manicura.jpg', 45, 200.00),
('Pedicura', 'Servicio de pedicura con esmalte.',
'https://example.com/pedicura.jpg', 45, 200.00),
('Masaje Relajante', 'Masaje relajante de cuerpo completo.',
'https://example.com//masaje.jpg', 60, 300.00);

– trabajadores
insert into users (email, password, dpi, phone, address, role_id) values
('t@test.com',
'$2a$10$dVXf63kVZPO0N7urHY99Z.qTb2wNVcW6GI5yrb6OXnNT0pJi
39xY2', '1234567890123', '1234567890', 'Calle 1, Ciudad', 4),
('t2@test.com',
'$2a$10$dVXf63kVZPO0N7urHY99Z.qTb2wNVcW6GI5yrb6OXnNT0pJi
39xY2', '1234567890123', '1234567890', 'Calle 2, Ciudad', 4),
('t3@test.com',
'$2a$10$dVXf63kVZPO0N7urHY99Z.qTb2wNVcW6GI5yrb6OXnNT0pJi
39xY2', '1234567890123', '1234567890', 'Calle 3, Ciudad', 4),
```

('t4@test.com',
'$2a$10$dVXf63kVZPO0N7urHY99Z.qTb2wNVcW6GI5yrb6OXnNT0pJi
39xY2', '1234567890123', '1234567890', 'Calle 4, Ciudad', 4);

-- Servicios de los trabajadores
insert into employee_services (employee_id, service_id) values
(4, 1),
(4, 2),
(5, 1),
(6, 1),
(7, 1),
(7, 2);

-- Usuario administrador
insert into users (email, password, dpi, phone, address, role_id) values
('admin@gmail.com',
'$2a$10$dVXf63kVZPO0N7urHY99Z.qTb2wNVcW6GI5yrb6OXnNT0pJi
39xY2', '1234567890123', '1234567890', 'Calle 1, Ciudad', 1);

alter table users add active boolean default true;

Diagrama de Componentes

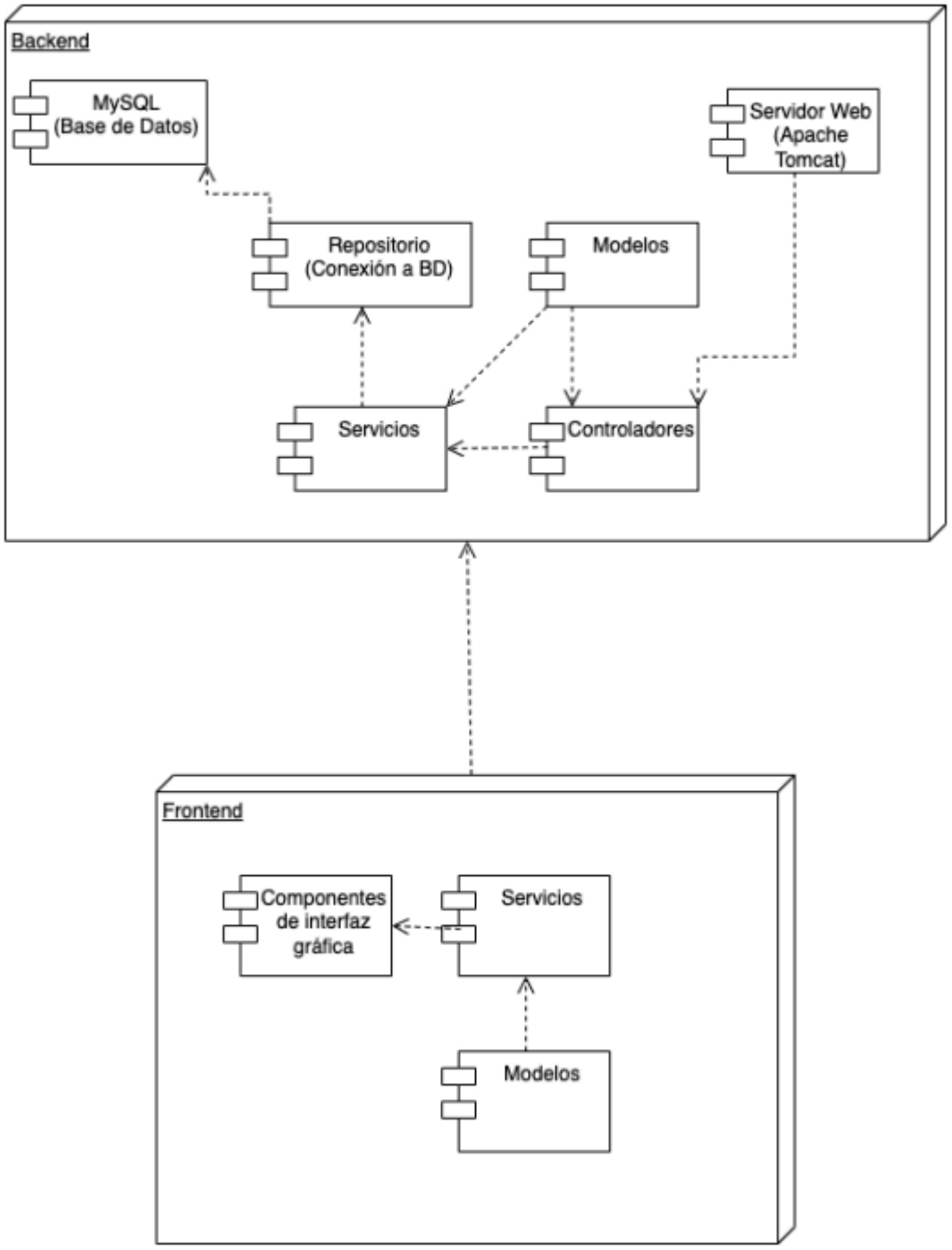
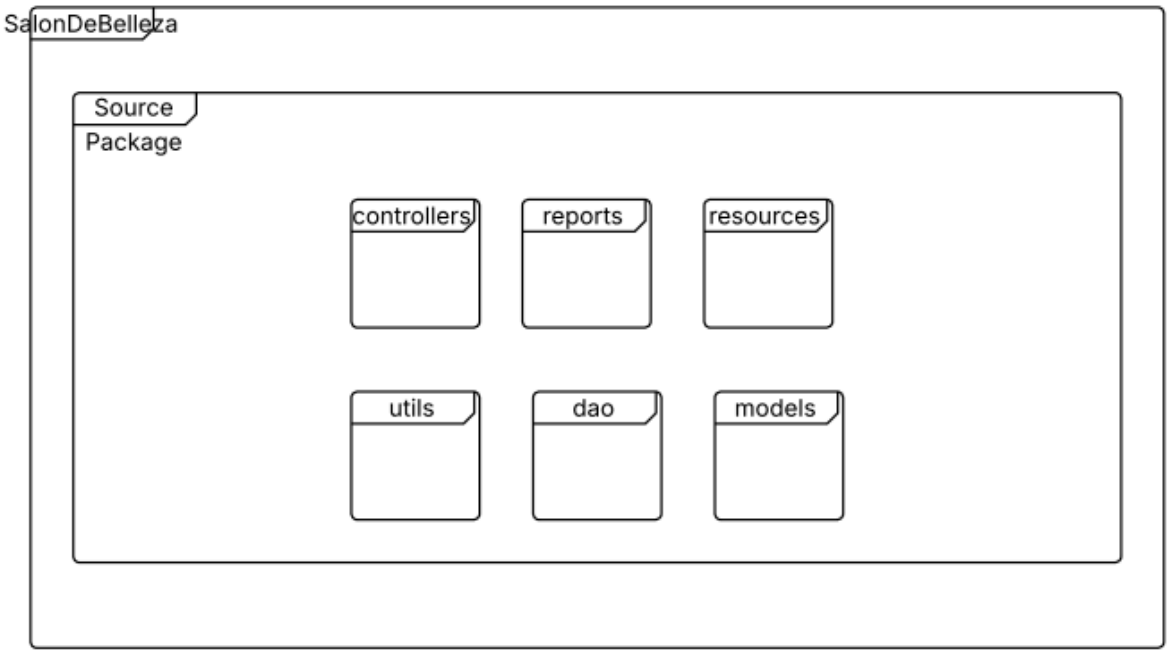


Diagrama de Paquetes



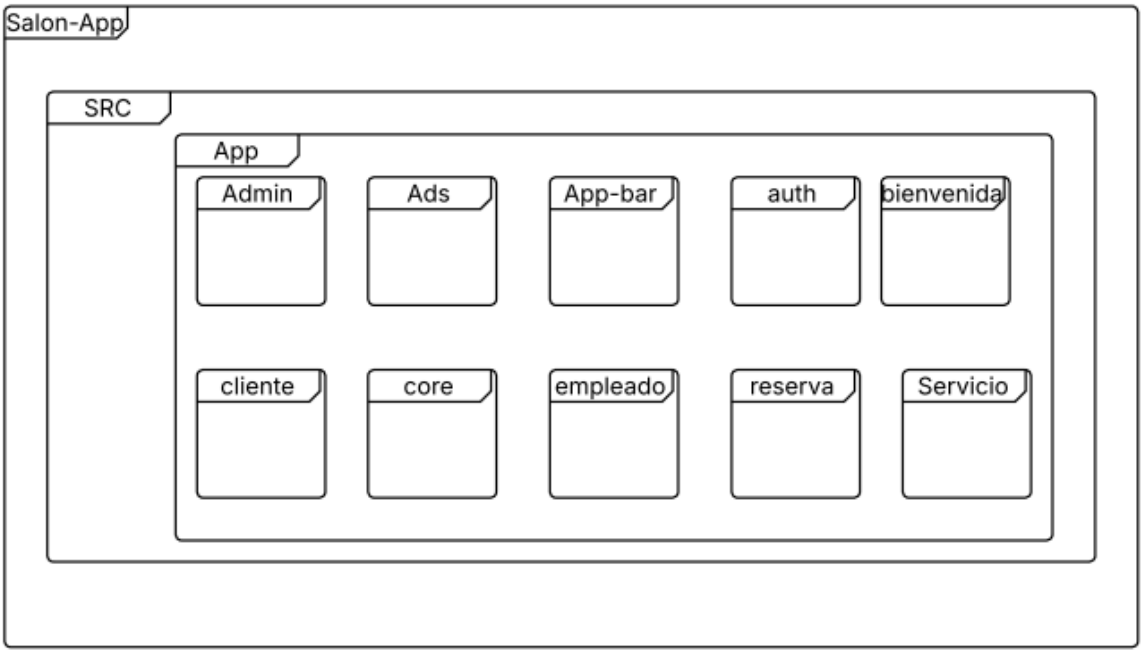
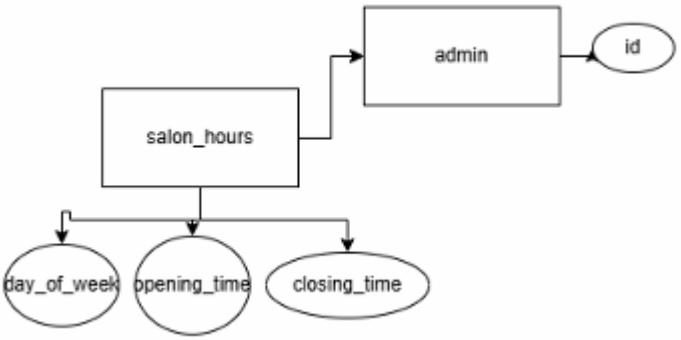
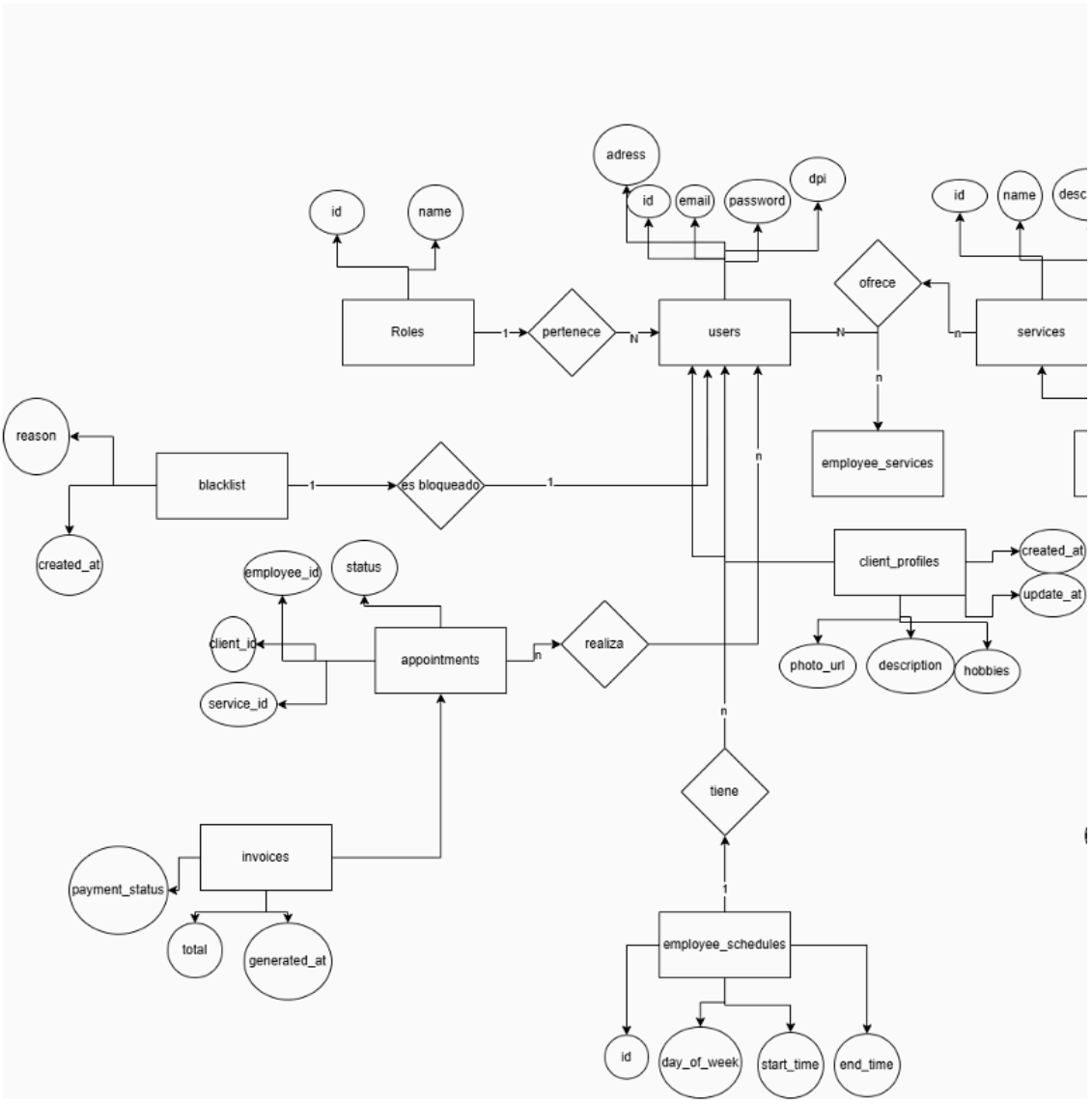


Diagrama Entidad Relación


admin
id
salon_hours
day_of_week
opening_time
closing_time


adress
id
email
password
dpi
id
name
Roles
1
pertenece
N
users
N
ofrece
n
services
id
name
reason
blacklist
1
es bloqueado
1
employee_services
created_at
employee_id
status
client_profiles
created_at
update_at
client_id
appointments
n
realiza
photo_url
description
hobbies
service_id
n
tiene
invoices
payment_status
total
generated_at
1
employee_schedules
id
day_of_week
start_time
end_time

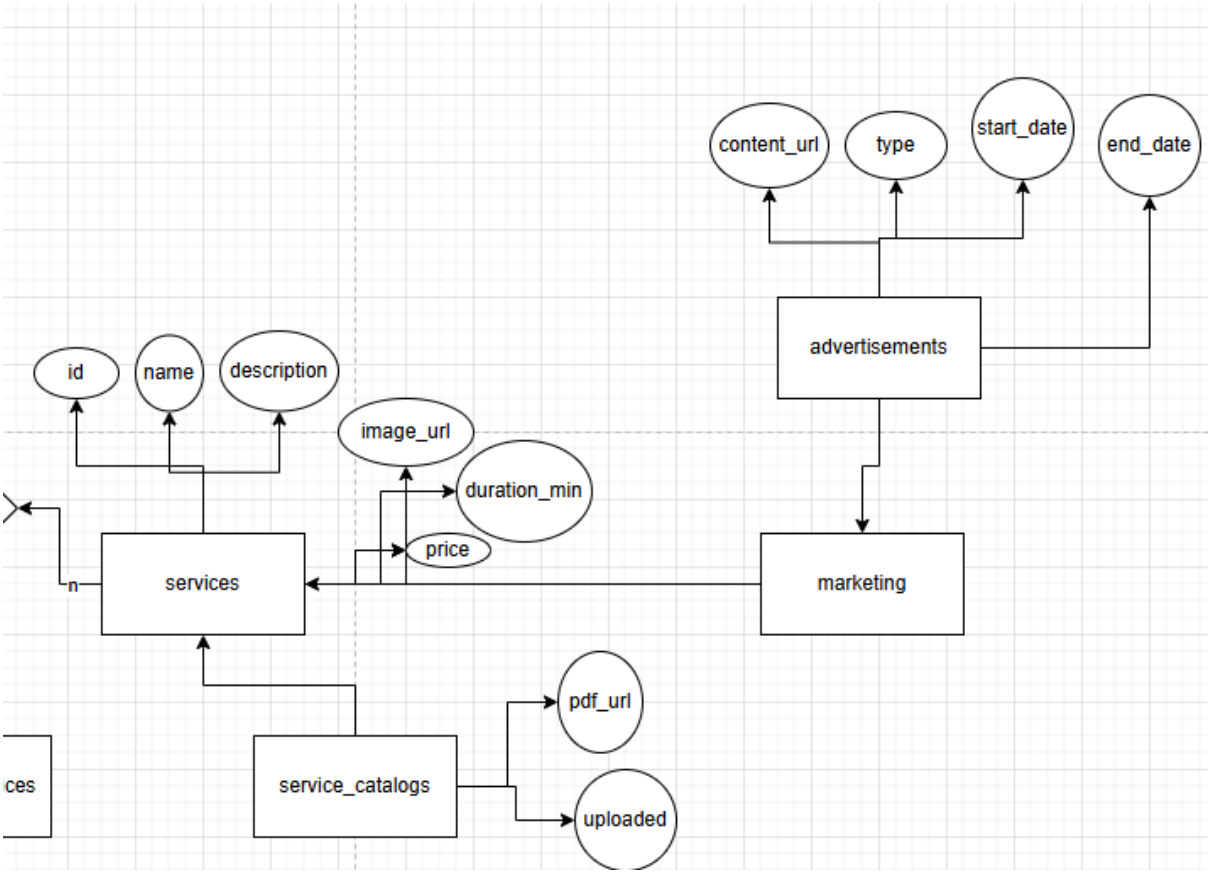
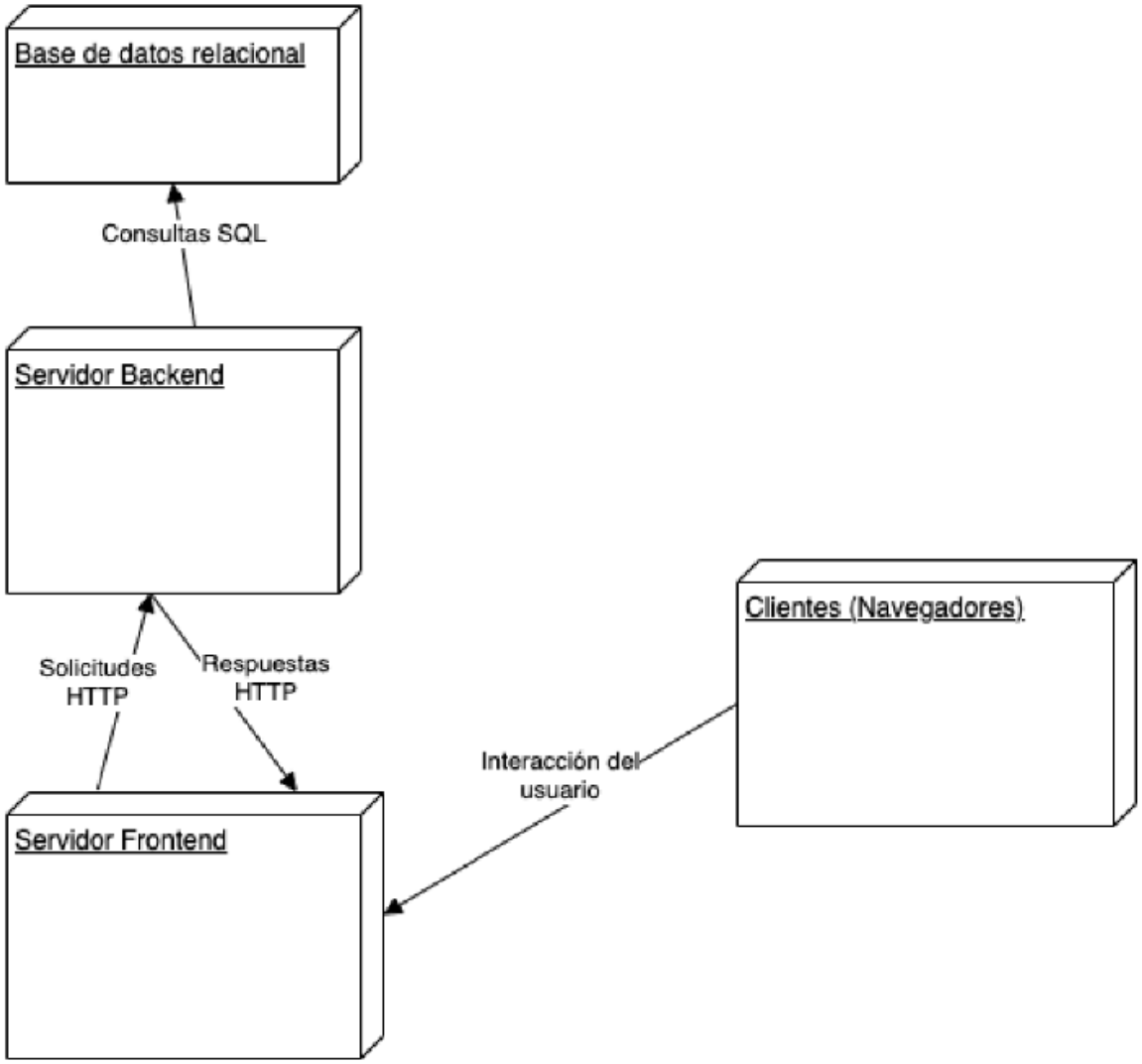


Diagrama de Despliegue



INSTRUCCIONES PARA EJECUCIÓN

Para ejecutar este proyecto es necesario tener instalado Java , Angular y Node en la computadora que se desee utilizar.

Java puede ser instalado desde la página oracle https://www.oracle.com/java/technologies/downloads/

Instalar Node puede ser instalado desde la herramienta en linea de comando llamada nvm la cual puede ser descargada del repositorio en GitHub.

Angular debe ser instalado desde la CMD o Terminal de tu computadora con los siguientes comandos: `npm install @angular/cli`

`ng version`

Primero se debe ejecutar el Backend el cual es SalonDeBelleza ya que este es el que conectará con la base de datos y hará que funcione correctamente el frontend. Se recomienda iniciarlo con NetBeans. Puedes usar el IDE de tu preferencia.

Luego de que se haya ejecutado correctamente el BackEnd se debe iniciar el FrontEnd mediante Visual Studio Code (de preferencia) y en la terminal ejecutar el comando npm i. Luego de la instalación que realiza npm i se debe de ejecutar el comando ng serve. Esto para que se pueda abrir la ruta y se puede usar el programa. La base de datos ya viene en el archivo para no tener complicaciones.