

Documentación Técnica: API Central de Gestión

Proyecto: Kingdom Barber

Versión: 1.0

Fecha: 29 de Septiembre, 2025

Autores: Juan Rivera, Andrés Vallejo, Alejandro Urrego



DESCRIPCIÓN GENERAL

Este documento detalla la arquitectura, características y uso de la **API central** desarrollada en **Node.js y Express** para el proyecto Kingdom Barber.

La API sirve como el backend principal, permitiendo gestionar todos los datos de la barbería, incluyendo citas, clientes, barberos, servicios y una galería de imágenes. Está diseñada para ser consumida por múltiples aplicaciones front-end, como el panel de administración (dashboard) y la página web pública de agendamiento de citas.



CARACTERÍSTICAS PRINCIPALES

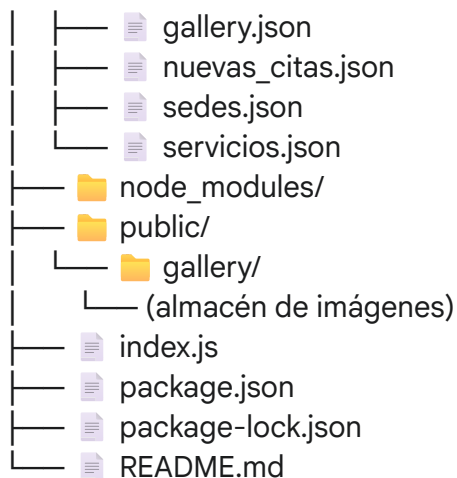
- **Gestión de Citas (CRUD):** Endpoints completos para crear, leer, actualizar y eliminar citas.
- **Consulta de Datos Maestros:** Rutas para obtener información de sedes, barberos, clientes y servicios.
- **Gestión de Galería de Imágenes (CRUD):** Sube, actualiza, consulta y elimina imágenes, incluyendo el manejo de los archivos físicos en el servidor.
- **Recepción de Mensajes:** Un endpoint dedicado para recibir y guardar mensajes enviados desde el formulario de contacto.
- **Persistencia de Datos:** Utiliza archivos **JSON** como sistema de base de datos, lo que facilita la configuración, portabilidad y mantenimiento.
- **Servidor de Archivos Estáticos:** Sirve las imágenes de la galería para que puedan ser consumidas directamente desde el front-end.



ESTRUCTURA DEL PROYECTO

El proyecto está organizado para separar la lógica, los datos y los recursos públicos.

```
.
├── database/
│   ├── barberos.json
│   ├── citas.json
│   ├── clientes.json
│   └── contactanos.json
```



- **index.js:** Archivo principal que contiene toda la lógica del servidor Express, la configuración de middlewares y la definición de todos los endpoints.
- **database/:** Directorio que funciona como la base de datos del sistema. Cada archivo .json representa una "tabla".
- **public/gallery/:** Carpeta donde se almacenan físicamente todas las imágenes subidas.



GUÍA DE INICIO RÁPIDO


Sigue estos pasos para levantar el servidor de la API en un entorno local.

Prerrequisitos

- Tener instalado **Node.js** (se recomienda una versión LTS).

Instalación y Ejecución

1. **Clonar el repositorio:**
git clone
https://github.com/JuanRivera24/nombre-del-repositorio.git
2. **Navegar al directorio del proyecto:**
cd nombre-del-repositorio
3. **Instalar las dependencias:**
npm install
4. **Iniciar el servidor:**
node index.js

 Una vez iniciado, la API estará corriendo y escuchando peticiones en <http://localhost:3001>.



ENDPOINTS DE LA API

5.1. Endpoints Compartidos (Datos Maestros)

Método	Ruta	Descripción
GET	/sedes	Devuelve una lista de todas las sedes.
GET	/barberos	Devuelve una lista de todos los barberos.
GET	/servicios	Devuelve una lista de todos los servicios.
GET	/clientes	Devuelve una lista de todos los clientes.
GET	/citas	Devuelve el historial completo de citas.

5.2. Gestión de Citas Nuevas

Método	Ruta	Descripción
GET	/nuevas_citas	Obtiene todas las citas agendadas, ordenadas por fecha.
POST	/nuevas_citas	Crea una nueva cita. Requiere un cuerpo JSON con detalles.
PUT	/nuevas_citas/:id	Actualiza la información de una cita existente por su ID.
DELETE	/nuevas_citas/:id	Elimina una cita específica por su ID.

5.3. Galería y Contacto

Método	Ruta	Descripción
POST	/contactanos	Guarda un nuevo mensaje del formulario de contacto.
GET	/gallery	Devuelve la lista de todas las imágenes de la galería.
POST	/gallery/upload	Sube una nueva imagen (multipart/form-data).
PUT	/gallery/:id	Actualiza la descripción o categoría de una imagen.
DELETE	/gallery/:id	Elimina una imagen de la base de datos y del servidor.

STACK TECNOLÓGICO

- **Node.js:** Entorno de ejecución para JavaScript del lado del servidor.
- **Express.js:** Framework minimalista para construir la API REST.
- **CORS:** Middleware para habilitar el Intercambio de Recursos de Origen Cruzado.
- **Multer:** Middleware para gestionar la subida de archivos (multipart/form-data).