



## HealthAPI - Sistema de Citas Médicas

### Descripción General

**HealthAPI** es un sistema backend para la gestión de citas médicas desarrollado con arquitectura hexagonal (puertos y adaptadores). Permite:

- Gestionar pacientes, médicos y citas médicas
- Agendar citas médicas con validación de traslapes
- Consultar historial de citas por paciente
- Asignar médicos a consultorios con franjas horarias
- Registrar observaciones y seguimiento de consultas

**Objetivo:** Proporcionar una API funcional, escalable y mantenible para sistemas de salud.

### Colaboradores del proyecto





- Jerónimo Roldán Cardona – Servicio 1: Gestión de agendamiento de citas
- Santiago Herrera Martínez – Servicio 2: Consulta de citas por paciente
- Samuel Restrepo Del Valle – Servicio 3: Asignación de médico a consultorio

---

### Arquitectura

El proyecto sigue **Arquitectura Hexagonal (Clean Architecture)** con separación clara de responsabilidades:

### Ventajas de esta arquitectura:

-  Independencia de frameworks y librerías
-  Testable: cada capa se puede probar aisladamente
-  Escalable: fácil agregar nuevos casos de uso
-  Mantenible: cambios en infraestructura no afectan la lógica de negocio

---

## ✂ Tecnologías

### Backend

- **Runtime:** Node.js
- **Lenguaje:** TypeScript
- **Framework Web:** Fastify
- **Base de Datos:** PostgreSQL

### Herramientas de Desarrollo

- **API Testing:** Bruno
- **Administración DB:** pgAdmin 4
- **Formato:** Prettier
- **Gestor de Paquetes:** pnpm/npm

---

## Migraciones de base de datos

El Proyecto utiliza un sistema de migraciones versionado ubicado en /migraciones.

### Migraciones Incluidas:

**000** tabla de control de migraciones

**001** tablas base (paciente, medico, disponibilidad, cita\_medica)

**002** tabla consultorio y relaciones

**003** índices compuestos

---

## API Endpoints

### Base URL

http://localhost:3000/api

### Citas Médicas

#### Crear Cita

POST /api/citas: Este endpoint espera mediante el body los siguientes parámetros:

- idPaciente
- idDisponibilidad
- fecha
- estado ("programada", "cancelada", "realizada")
- motivo
- observaciones

Seguido de pasarle el parámetro con la información correspondiente a cada uno, se debe presentar la siguiente respuesta: **Respuesta (200) OK.**

### Listar Todas las Citas

GET /api/citas: Este endpoint se ejecuta mediante BRUNO directamente, lo cual nos pasa en forma de una lista todas las citas que estén registradas en la base de datos respectivamente.

### Obtener Cita por ID

GET /api/citas/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos buscar, seguido a esto, ejecutar la request en bruno y recibir la información de si fue encontrado o no la cita respectivamente.

### Actualizar Cita

PUT /api/citas/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos actualizar, y seguido a esto, en el body pasarle el o los parámetros que deseamos actualizar en la cita, sean varios cambios o solo uno.

### Eliminar Cita

DELETE /api/citas/:id: En este endpoint es necesario pasar al final de la URL el ID que deseamos eliminar, si el sistema lo encuentra, procede a eliminarlo por completo de la base de datos.

---

## Médicos

### Crear Médico

POST /api/medicos: Este endpoint espera mediante el body los siguientes parámetros:

- nombreMedico
- emailMedico
- especialidadMedico

Seguido de pasarle el parámetro con la información correspondiente a cada uno, se debe presentar la siguiente respuesta: **Respuesta (200) OK.**

## Listar Médicos

GET /api/medicos: Este endpoint se ejecuta mediante BRUNO directamente, lo cual nos pasa en forma de una lista todos los médicos que estén registrados en la base de datos respectivamente.

### Buscar médico por ID

GET /api/medico/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos buscar, seguido a esto, ejecutar la request en bruno y recibir la información de si fue encontrado o no el médico respectivamente.

### Actualizar médico

PUT /api/medico/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos actualizar, y seguido a esto, en el body pasarle el o los parámetros que deseamos actualizar en la cita, sean varios cambios o solo uno.

### Eliminar médico

DELETE/api/medico/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos eliminar, si el sistema lo encuentra, procede a eliminarlo por completo de la base de datos.

---

## Pacientes

### Crear Paciente

POST /api/pacientes: Este endpoint espera mediante el body los siguientes parámetros:

- nombre
- email
- telefono

Seguido de pasarle el parámetro con la información correspondiente a cada uno, se debe presentar la siguiente respuesta: **Respuesta (200) OK.**

### Listar pacientes

GET /api/pacientes: Este endpoint se ejecuta mediante BRUNO directamente, lo cual nos pasa en forma de una lista todos los pacientes que estén registrados en la base de datos respectivamente.

### Buscar pacientes por ID

GET /api/paciente/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos buscar, seguido a esto, ejecutar la request en bruno y recibir la información de si fue encontrado o no el paciente respectivamente.

### Actualizar paciente

PUT /api/paciente/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos actualizar, y seguido a esto, en el body pasarle el o los parámetros que deseamos actualizar en el paciente, sean varios cambios o solo uno.

### Eliminar paciente

DELETE/api/paciente/id: En este endpoint es necesario pasar al final de la URL el ID que deseamos eliminar, si el sistema lo encuentra, procede a eliminarlo por completo de la base de datos.

---

## ENDPOINTS DE SERVICIO 1

### CITAS MÉDICAS

Método	Endpoint	Descripción	Código Éxito
POST	/api/citas	Crear cita médica	201
GET	/api/citas	Listar todas las citas	200
GET	/api/citas/:id	Obtener cita por ID	200
PUT	/api/citas/:id	Actualizar cita	200
DELETE	/api/citas/:id	Eliminar cita	200
POST	/api/citas/agendar	Agendar cita (con validaciones)	201

## ENDPOINTS DE SERVICIO 2

### Resumen

Caso	Endpoint	Código HTTP	Descripción
Éxito	GET /api/pacientes/1/citas	200	Paciente con citas
Éxito	GET /api/pacientes/2/citas	200	Paciente sin citas (lista vacía)
Error	GET /api/pacientes/999/citas	404	Paciente no existe
Error	GET /api/pacientes/abc/citas	400	ID no es número
Error	GET /api/pacientes/-1/citas	400	ID negativo
Error	GET /api/pacientes/0/citas	400	ID cero

### ENDPOINTS DE SERIVIO 3

Método	Endpoint	Descripción	Código Éxito
POST	/api/disponibilidades	Crear disponibilidad	201
GET	/api/disponibilidades	Listar todas las disponibilidades	200
GET	/api/disponibilidades/:id	Obtener disponibilidad por ID	200
GET	/api/medicos/:idMedico/disponibilidades	Listar disponibilidades de un médico	200
GET	/api/consultorios/:idConsultorio/disponibilidades	Listar disponibilidades de un consultorio	200
PUT	/api/disponibilidades/:id	Actualizar disponibilidad	200
DELETE	/api/disponibilidades/:id	Eliminar disponibilidad	200

---

### Validaciones implementadas

#### Validaciones de unicidad:

- No duplicar ID paciente
- No duplicar ID médico
- No duplicar nombre consultorio

#### Validaciones de NO duplicidad:

- Paciente no puede tener 2 citas superpuestas
- Medico no puede tener doble reserva en horario traslapado
- Consultorio no puede tener doble uso en horario traslapado
- No permitir asignación duplicada de médico-consultorio-horario

#### Validaciones de traslapes:

- Detectar traslapes de horario para médicos
- Detectar traslapes de horario para consultorios
- Detectar traslapes de horario para pacientes

- Validar que la cita esté dentro de la disponibilidad del médico

### **Testing con Bruno**

La carpeta /bruno contiene todas las peticiones pre-configuradas:

1. Abrir Bruno
2. File → Open Collection
3. Seleccionar la carpeta /bruno dentro del proyecto HealthAPI
4. Navegar por las request ya preestablecidas para mayor facilidad de test