¡Excelente! Detallar y probar los endpoints es un paso crucial para asegurar que nuestro backend es sólido antes de construir la interfaz.

Aquí tienes una guía de prueba exhaustiva usando **Postman**. Vamos a simular el flujo de trabajo completo.

### Preparación en Postman

1. **Crea dos usuarios:** Usa POST /api/auth/register para crear:
   * employee@test.com (password: password123)
   * admin@test.com (password: password123)
2. **Haz que uno sea admin:** Usa DBeaver para actualizar el rol del segundo usuario.

Generated sql

UPDATE "Users" SET role = 'admin' WHERE email = 'admin@test.com';

1. **Obtén sus Tokens:** Usa POST /api/auth/login para ambos usuarios y guarda sus accessToken. Los usaremos en los siguientes pasos. Llamémoslos TOKEN\_EMPLOYEE y TOKEN\_ADMIN.

### Guía de Pruebas de Endpoints

#### ****Flujo 1: Un empleado crea un ticket y lo revisa.****

**1.1. Crear un Ticket (Como Employee)**

* **Método:** POST
* **URL:** http://localhost:8081/api/tickets
* **Authorization:** Bearer Token -> TOKEN\_EMPLOYEE
* **Body (JSON):** (Asumimos que el ID de la categoría 'Problema de Software' es 2, según nuestro seeder).

Generated json

{

"title": "No puedo abrir la aplicación de reportes",

"description": "Desde esta mañana, cuando intento abrir la aplicación de reportes me sale un error 503. He probado reiniciando el equipo. Mi compañero tampoco puede entrar.",

"categoryId": 2

}

IGNORE\_WHEN\_COPYING\_START

content\_copy download

Use code [with caution](https://support.google.com/legal/answer/13505487). Json

IGNORE\_WHEN\_COPYING\_END

* **Respuesta Esperada:** Status 201 Created. Un objeto JSON con los datos del ticket creado, incluyendo su nuevo id, y statusId apuntando al estado "Abierto". Guarda el id de este ticket para los siguientes pasos (ej. TICKET\_ID = 1).

**1.2. Ver "Mis Tickets" (Como Employee)**

* **Método:** GET
* **URL:** http://localhost:8081/api/tickets
* **Authorization:** Bearer Token -> TOKEN\_EMPLOYEE
* **Respuesta Esperada:** Status 200 OK. Un array JSON que contiene el ticket que acabas de crear. Los campos creator, category, etc., deben venir poblados con sus datos.

**1.3. Ver el Detalle de su Propio Ticket (Como Employee)**

* **Método:** GET
* **URL:** http://localhost:8081/api/tickets/{TICKET\_ID} (reemplaza {TICKET\_ID} con el ID real)
* **Authorization:** Bearer Token -> TOKEN\_EMPLOYEE
* **Respuesta Esperada:** Status 200 OK. Un objeto JSON con todos los detalles del ticket, incluyendo un array comments (que estará vacío por ahora).

#### ****Flujo 2: Un admin gestiona el ticket.****

**2.1. Ver Todos los Tickets (Como Admin)**

* **Método:** GET
* **URL:** http://localhost:8081/api/tickets
* **Authorization:** Bearer Token -> TOKEN\_ADMIN
* **Respuesta Esperada:** Status 200 OK. Un array JSON con todos los tickets del sistema, incluyendo el que creó el empleado.

**2.2. Actualizar el Ticket: Asignarlo y Cambiar Estado (Como Admin)**

* **Método:** PUT
* **URL:** http://localhost:8081/api/tickets/{TICKET\_ID}
* **Authorization:** Bearer Token -> TOKEN\_ADMIN
* **Body (JSON):** (Asumimos que el ID del admin@test.com es 2 y el ID del estado 'En Progreso' es 2).

Generated json

{

"assigneeId": 2,

"statusId": 2

}

IGNORE\_WHEN\_COPYING\_START

content\_copy download

Use code [with caution](https://support.google.com/legal/answer/13505487). Json

IGNORE\_WHEN\_COPYING\_END

* **Respuesta Esperada:** Status 200 OK. Un objeto JSON con los datos del ticket actualizado. Verifica que assignee.id sea 2 y status.id sea 2.

**2.3. Probar Acceso Denegado (Como Employee)**

* **Intenta** ejecutar la misma petición PUT anterior, pero usando el TOKEN\_EMPLOYEE.
* **Respuesta Esperada:** Status 403 Forbidden. La ruta está protegida por authJwt.isAdmin.

#### ****Flujo 3: Comunicación a través de Comentarios****

**3.1. Admin añade un comentario (Como Admin)**

* **Método:** POST
* **URL:** http://localhost:8081/api/tickets/{TICKET\_ID}/comments
* **Authorization:** Bearer Token -> TOKEN\_ADMIN
* **Body (JSON):**

Generated json

{

"content": "Hola, gracias por el reporte. Estamos investigando el problema con el servidor de reportes."

}

IGNORE\_WHEN\_COPYING\_START

content\_copy download

Use code [with caution](https://support.google.com/legal/answer/13505487). Json

IGNORE\_WHEN\_COPYING\_END

* **Respuesta Esperada:** Status 201 Created. Un objeto JSON con los datos del nuevo comentario, incluyendo la información de su author.

**3.2. Employee añade un comentario (Como Employee)**

* **Método:** POST
* **URL:** http://localhost:8081/api/tickets/{TICKET\_ID}/comments
* **Authorization:** Bearer Token -> TOKEN\_EMPLOYEE
* **Body (JSON):**

Generated json

{

"content": "¡Muchas gracias! Estaré atento a las novedades."

}

IGNORE\_WHEN\_COPYING\_START

content\_copy download

Use code [with caution](https://support.google.com/legal/answer/13505487). Json

IGNORE\_WHEN\_COPYING\_END

* **Respuesta Esperada:** Status 201 Created.

**3.3. Ver el Ticket con Comentarios (Como Admin o Employee)**

* **Método:** GET
* **URL:** http://localhost:8081/api/tickets/{TICKET\_ID}
* **Authorization:** TOKEN\_ADMIN o TOKEN\_EMPLOYEE
* **Respuesta Esperada:** Status 200 OK. Un objeto JSON del ticket. Ahora, el array comments debería contener dos objetos, en orden de creación.

#### ****Flujo 4: Obtener datos para los formularios del frontend****

**4.1. Obtener Estados de Ticket (Como Admin o Employee)**

* **Método:** GET
* **URL:** http://localhost:8081/api/data/ticket-statuses
* **Authorization:** TOKEN\_ADMIN o TOKEN\_EMPLOYEE
* **Respuesta Esperada:** Status 200 OK. Un array con todos los estados que insertamos con el seeder.

**4.2. Obtener Categorías de Ticket (Como Admin o Employee)**

* **Método:** GET
* **URL:** http://localhost:8081/api/data/ticket-categories
* **Authorization:** TOKEN\_ADMIN o TOKEN\_EMPLOYEE
* **Respuesta Esperada:** Status 200 OK. Un array con todas las categorías.

**4.3. Obtener Lista de Admins (Como Admin)**

* **Método:** GET
* **URL:** http://localhost:8081/api/data/admins
* **Authorization:** TOKEN\_ADMIN
* **Respuesta Esperada:** Status 200 OK. Un array que contiene al usuario admin@test.com (con solo id y name).
* **Prueba de seguridad:** Intenta hacer la misma petición con TOKEN\_EMPLOYEE. Deberías obtener un 403 Forbidden.

Una vez que hayas completado estas pruebas y todo funcione como se espera, tu backend estará listo y validado.

**Guarda tu progreso:**

Generated bash

git add .

git commit -m "feat(backend): implement full API for tickets and comments"

IGNORE\_WHEN\_COPYING\_START

content\_copy download

Use code [with caution](https://support.google.com/legal/answer/13505487). Bash

IGNORE\_WHEN\_COPYING\_END

Ahora sí, con total confianza en nuestra API, podemos pasar al frontend. El **Paso 6** será construir las vistas para listar tickets y la vista de detalle.

**¿Listo para volver a Vue?**