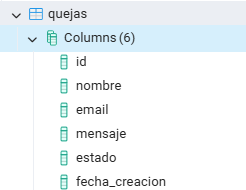
ACTIVIDADES DESARROLLADAS EN PRACTICAS PRE PROFESIONALES   
Sprint 2

Lunes 28 de abril de 2025

**Actividades realizadas:**

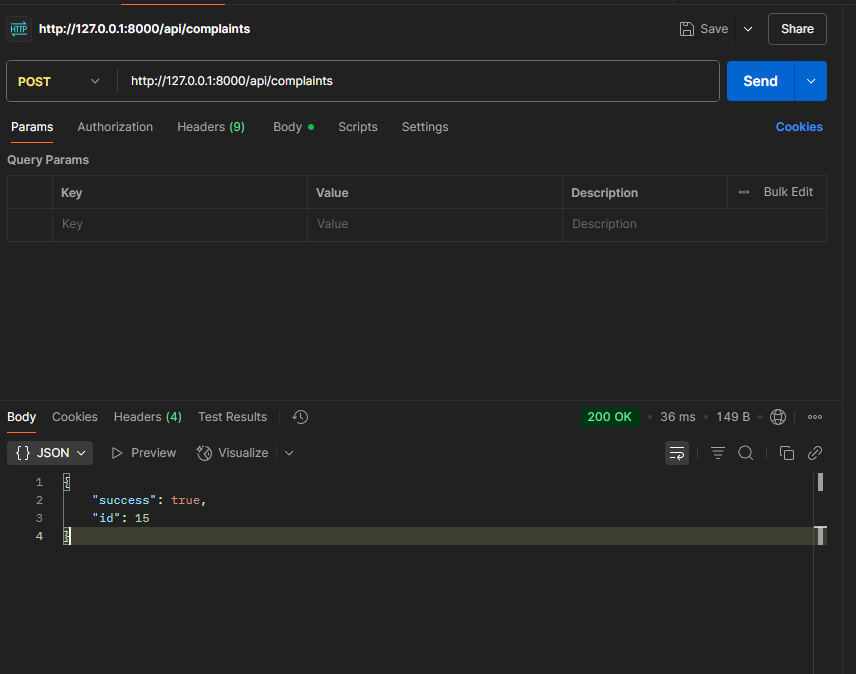
Instalación y configuración de PostgreSQL:  
El sprint comenzó con la instalación de PostgreSQL, un sistema de gestión de bases de datos relacional conocido por su confiabilidad y capacidad para manejar datos complejos. Se eligió PostgreSQL por su soporte para consultas avanzadas y su comunidad activa, ideal para un sistema que necesita almacenar quejas de manera estructurada. Se creó una base de datos llamada quejas utilizando herramientas gráficas como pgAdmin o comandos directos en la terminal de PostgreSQL. Dentro de esta base de datos, se diseñó una tabla también llamada quejas, con campos cuidadosamente definidos: un identificador único que se genera automáticamente, un campo de texto para el nombre (limitado a 50 caracteres), otro para el correo electrónico (hasta 100 caracteres), un campo de texto largo para el mensaje de la queja, un campo para el estado (con valor predeterminado “Abierta”), y un campo de fecha que registra automáticamente el momento en que se crea la queja. Para verificar que la tabla estuviera bien configurada, se ejecutaron comandos en la terminal para inspeccionar su estructura y asegurarse de que los tipos de datos y restricciones fueran correctos. Este paso fue crucial para evitar problemas futuros al insertar o consultar datos. Los desafíos incluyeron familiarizarse con los comandos de PostgreSQL y garantizar que la base de datos estuviera accesible desde el servidor Node.js, lo que requirió configurar correctamente las credenciales de acceso.



Martes 29 de abril de 2025

**Actividades realizadas:**

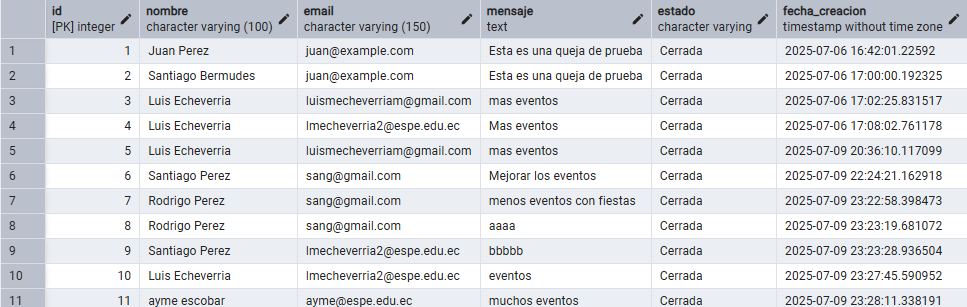
Desarrollo del endpoint POST /api/quejas:  
El enfoque de este día fue conectar el backend con la base de datos para que las quejas enviadas desde el formulario se almacenaran permanentemente. Se mejoró el endpoint POST /api/quejas para que interactuara con PostgreSQL, utilizando una biblioteca que facilita las conexiones entre Node.js y la base de datos. Se diseñó un proceso para insertar los datos del formulario (nombre, correo, mensaje) en la tabla quejas, asegurando que la operación fuera segura y no vulnerable a ataques maliciosos, como la inyección de datos. También se añadieron validaciones en el servidor para verificar que todos los campos estuvieran completos antes de almacenarlos, devolviendo un mensaje de error si faltaba información. Para probar esta funcionalidad, se utilizó Postman, una herramienta que permite simular solicitudes al servidor. Se enviaron datos de ejemplo, como una queja con un nombre, correo y mensaje, y se verificó que se guardaran correctamente en la base de datos al consultar la tabla. Los retos incluyeron depurar errores de conexión entre el servidor y la base de datos, como problemas con contraseñas o puertos mal configurados, que se resolvieron ajustando los parámetros de conexión.



Miercoles 30 de abril de 2025

**Actividades realizadas:**

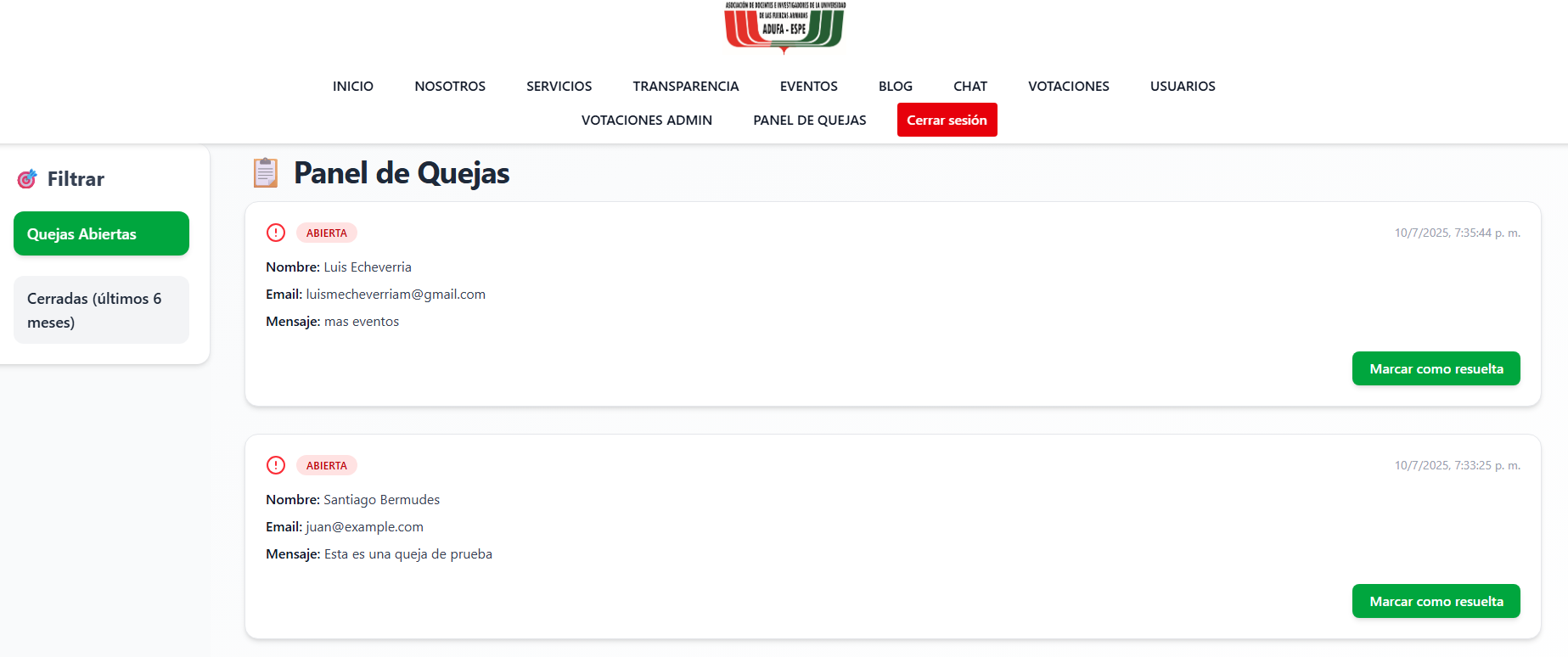
Pruebas de integración completas:  
Este día se dedicó a probar todo el flujo del sistema: desde que el usuario llena el formulario en el frontend, los datos se envían al backend, y finalmente se almacenan en la base de datos. Se realizaron múltiples pruebas enviando quejas con diferentes datos para asegurarse de que todo funcionara correctamente, incluyendo casos con errores intencionales, como correos inválidos o mensajes vacíos, para confirmar que el sistema los detectaba y respondía adecuadamente. También se verificó que los datos llegaran correctamente a la base de datos consultando la tabla quejas después de cada envío. Esto aseguró que no hubiera pérdida de información ni errores en el almacenamiento. Se prestó especial atención a la comunicación entre el frontend y el backend, asegurando que las políticas de acceso (CORS) permitieran solicitudes sin problemas. Los desafíos incluyeron manejar casos extremos, como envíos simultáneos o fallos temporales del servidor, que se resolvieron ajustando la lógica del backend y probando con diferentes navegadores. Al final del día, el sistema demostró ser estable, con un flujo completo desde la interfaz hasta la base de datos.

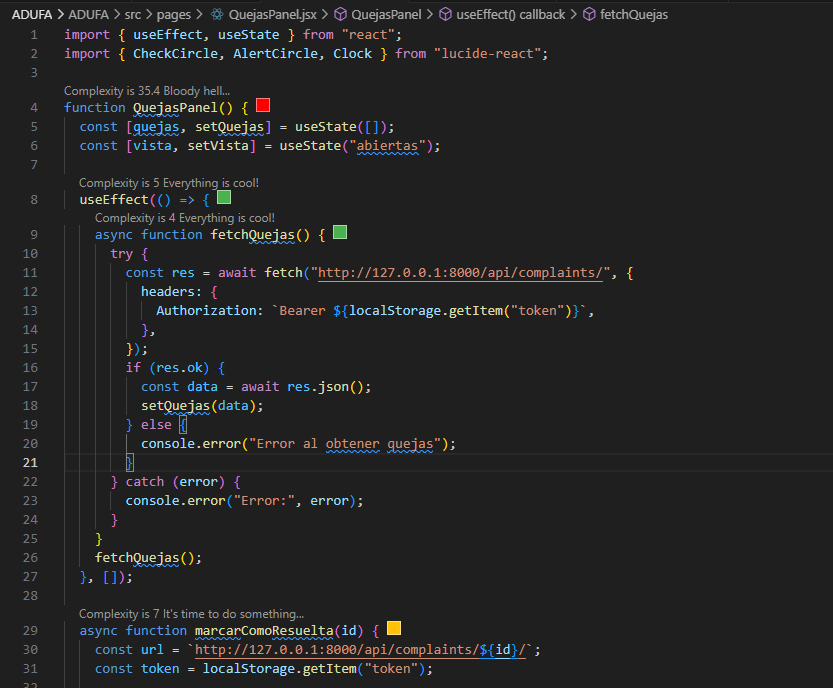


Jueves 1 de mayo de 2025

**Actividades realizadas:**

Creación de componente PanelAdmin.jsx:  
Se desarrolló un nuevo componente en React, llamado PanelAdmin.jsx, para que los administradores pudieran visualizar todas las quejas registradas. Este componente mostraba los datos en una tabla con columnas claras: nombre, correo, mensaje, fecha de creación y estado. Para alimentar esta tabla, se creó un nuevo endpoint en el backend (GET /api/quejas) que recuperaba todas las quejas de la base de datos. Este endpoint se diseñó para ser eficiente, devolviendo los datos en un formato que el frontend pudiera procesar fácilmente. En el frontend, se usó una herramienta para hacer solicitudes al servidor y mostrar los resultados en la tabla, estilizada con Tailwind CSS para que fuera visualmente atractiva y fácil de leer, con bordes claros y colores que diferenciaban cada fila. Las pruebas iniciales confirmaron que el panel mostraba correctamente todas las quejas, y se ajustaron detalles como el tamaño de las columnas para mejorar la legibilidad. Un desafío fue manejar casos en los que la base de datos estuviera vacía, lo que se resolvió mostrando un mensaje como “No hay quejas registradas” en la interfaz.





Viernes 2 de mayo de 2025

**Actividades realizadas:**

Mejoras visuales y funcionales en el panel:  
Se trabajó en refinar el panel administrativo para hacerlo más funcional y atractivo. Se ajustaron las columnas de la tabla para que tuvieran un espaciado uniforme y se añadieron opciones para filtrar las quejas por estado, como “Abierta” o “Cerrada”, usando un menú desplegable o botones interactivos. Esto permitió a los administradores enfocarse en quejas específicas sin revisar toda la lista. También se mejoraron los estilos visuales, aplicando colores más distintivos para los estados (por ejemplo, verde para “Abierta” y rojo para “Cerrada”) y añadiendo efectos visuales, como bordes resaltados al pasar el cursor sobre una fila. Se manejaron casos especiales, como cuando no había datos, mostrando mensajes claros para evitar confusión. El equipo revisó todo el código del sprint para asegurar que fuera claro y organizado, eliminando cualquier elemento innecesario. Finalmente, se documentaron los avances en un archivo detallado y se subieron al repositorio de GitHub, completando el sprint con un sistema que ahora podía almacenar y mostrar quejas de manera efectiva.



