# EXCHANGE HUB

Arquitectura

GERONIMO TRUJILLO – JULIAN POSSO – JUAN RUIZ
SENA 2025

## Contenido

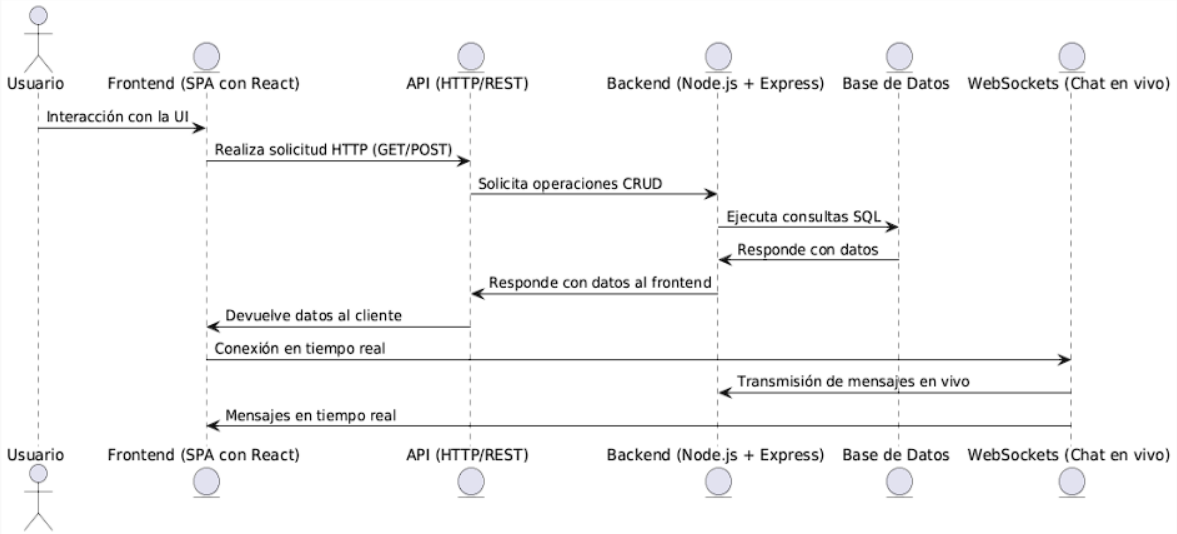
# 1. INTRODUCCIÓN

**Objetivo del documento:** Describir la arquitectura técnica en la cual se encuentra desarrollado el proyecto, a su vez el contexto y explicación de esta misma.

## 2. DESCRIPCIÓN GENERAL DE LA ARQUITECTURA

El proyecto de **EXCHANGE HUB** se encuentra desarrollado bajo una arquitectura de **Cliente-Servidor** la cual se enfoca en mantener separados el **BACKEND** y **FRONTEND**, facilitando su escalabilidad y comodidad de cara a futuras actualizaciones o correcciones del software.

El flujo funcional de esta arquitectura radica en la comunicación del **FRONTEND** con el **BACKEND** mediante una API que se encarga de enviar las peticiones realizadas por el **FRONTEND** hacia el **BACKEND,** para luego enviar al **FRONTEND** una respuesta con el fin de que la información sea visualizada por el usuario.




## 3. COMPOSICIÓN DE CLIENTE Y SERVIDOR

- **Cliente:** Desarrollado como **SPA(Single Page Application)** utilizando REACT como framework o librería principal con una clara separación de sus componentes, paginas, estilos y control de flujo con el uso de contextos.

- **Servidor:** Desarrollado en el entorno de Node.js utilizando Express como framework para la creación del servidor. Cuenta con una clara separación de controladores, rutas y archivos de configuración.

## 4. COMUNICACÓN ENTRE COMPONENTES (BACKEND, FRONTEND, BASE DE DATOS)

La comunicación entre **FRONTEND** y **BACKEND** se realiza de dos maneras, en la primera encontramos el protocolo **HTTP** utilizado por la API que lo hace mediante AXIOS para realizar la operaciones CRUD que llegan desde el **FRONTEND** al **BACKEND** para ser registradas en la base de datos. Por otra parte contamos con WebSockets los cuales son utilizados para el tema del sistema de comunicación interna del software en pocas palabras un chat en tiempo real, por eso mismo utilizamos Sokects para poder tener una comunicación en vivo.

**Diagrama de Flujo de Datos**

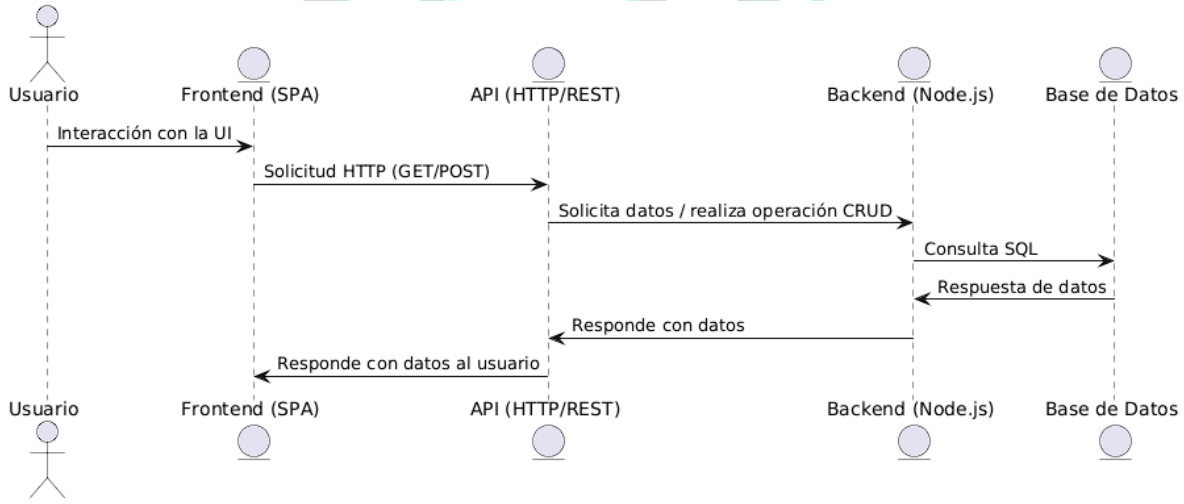
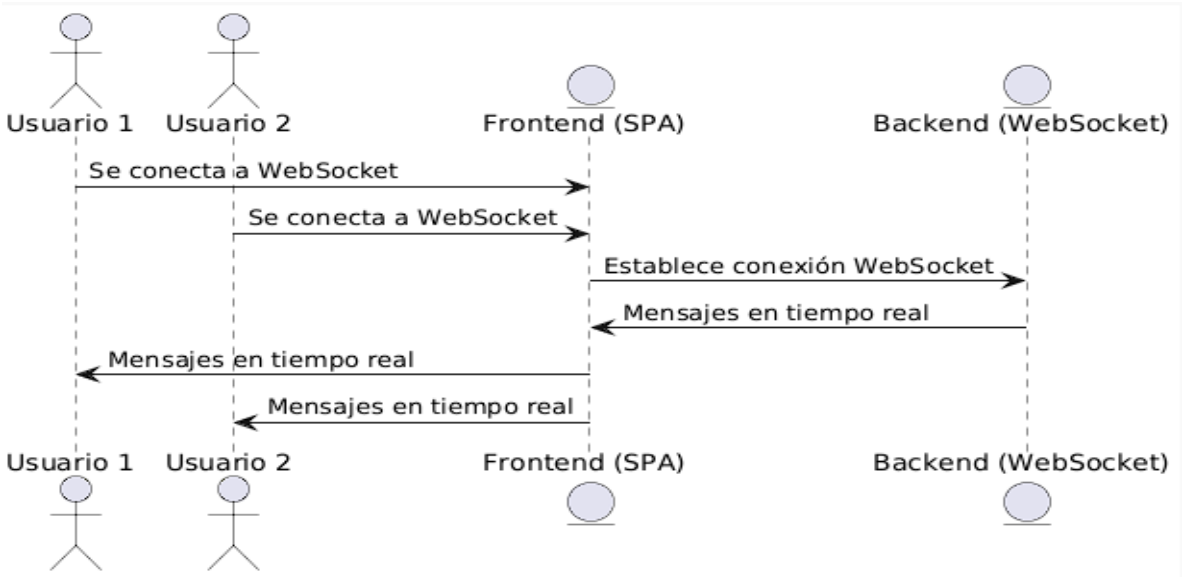


**Diagrama de Comunicación en Tiempo Real (WebSockets)**




**Interacción entre Backend y Base de datos**

La aplicación necesita almacenar datos y por eso el **BACKEND** mantiene comunicación con la base de datos mediante sus controladores los cuales se encargan de realizar las operaciones CRUD ejecutando las consultas SQL, recibiendo la respuesta de la base de datos, manipulándola para posteriormente enviarla por la API hacia el **FRONTEND** si es necesario.

**Manejo de errores y excepciones**

Todas las comunicaciones entre **FRONTEND** y **BACKEND - BACKEND** y **Base de Datos,** se realizan de manera asíncrona para maximizar el ahorro de recursos aprovechando las multitareas. También cabe recalcar que todas estas

comunicaciones asincrónicas están controladas por **TRY CATCH** permitiendo la fácil captura e identificación de errores, asegurando que el flujo de ejecución no se rompa y el sistema reaccione de manera adecuada.