Grupo 4 - Integrantes: Jose Riofrio, Francisco Jaramillo, Juan Alverca

1. **Tema**

Documentación del desarrollo de microservicios y su integración mediante herramientas colaborativas y APIs.

2. **Propósito**

El estudiante integra los conocimientos adquiridos sobre la arquitectura de microservicios, su conexión a través de API REST y el uso de herramientas colaborativas para la gestión y documentación del proyecto, como resultado, elaborará un informe técnico que describa las decisiones de diseño, las herramientas empleadas y el efecto de los microservicios en la escalabilidad y el mantenimiento del sistema.

3. **Desarrollo**

3.1 Resumen del microservicio creado.

Este microservicio tiene como función principal gestionar las categorías del sistema, proporciona endpoints para consultar todas las categorías, crear nuevas categorías y actualizar las existentes, garantizando que la información esté siempre organizada y actualizada.

Está construido con FastAPI y utiliza una base de datos SQL para almacenar los datos de las categorías, su diseño modular permite que se integre fácilmente con otros microservicios del sistema principal, como los relacionados con productos o inventario, facilitando la interoperabilidad, escalabilidad y mantenimiento del proyecto.

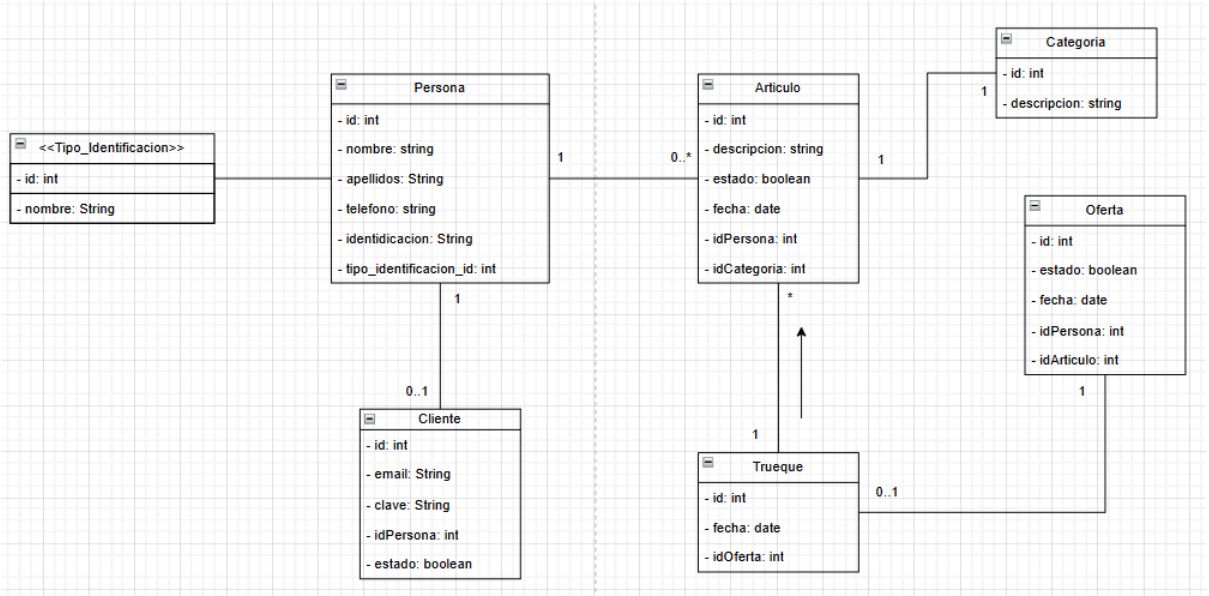
- **Nombre del microservicio, propósito y relación con el sistema principal.**

Nombre del microservicio: Microservicio de Categorías

Propósito: Este microservicio se encarga de gestionar las categorías dentro del sistema, permitiendo crear nuevas categorías, consultar las existentes y actualizar su información de manera sencilla y centralizada.

Relación con el sistema principal: Se integra con el sistema principal para proporcionar información consistente sobre las categorías a otros módulos, como productos o inventario, asegurando que todos los componentes que dependen de las categorías puedan acceder a datos actualizados y confiables.

- **Diagrama de clases**

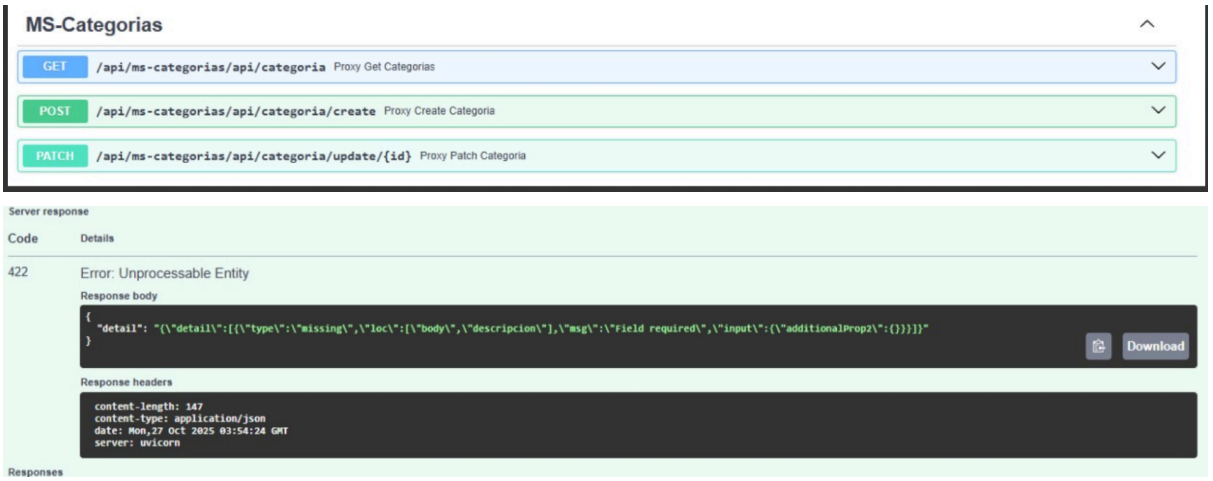


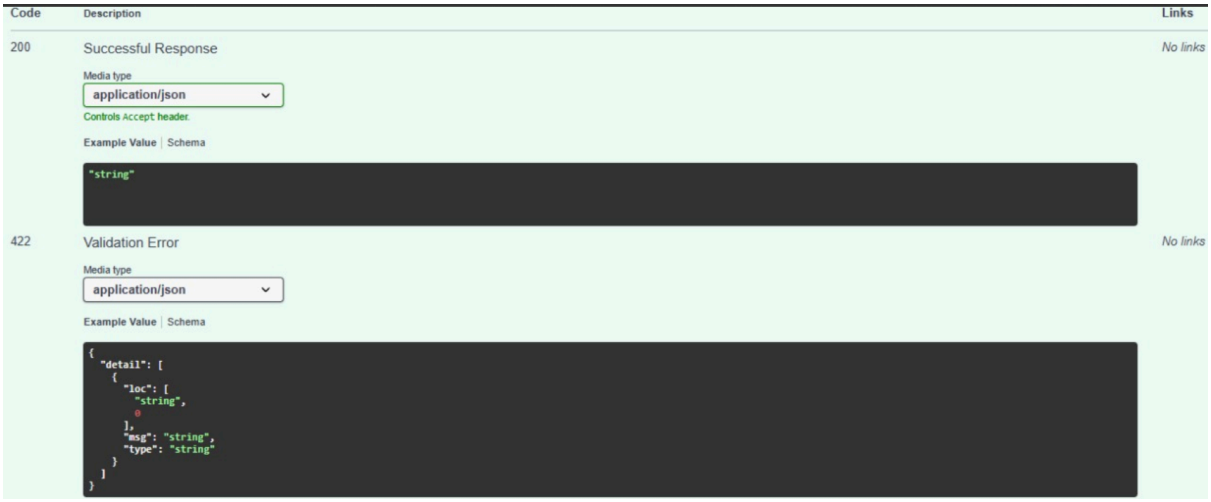
3.2 Integración mediante APIs

- **Descripción del método de comunicación; REST, API Gateway o endpoints compartidos.**

El microservicio de Categorías utiliza comunicación REST a través de endpoints HTTP, lo que permite que otros servicios o aplicaciones puedan interactuar con él de manera sencilla y estándar, cada operación (consultar, crear o actualizar categorías) se expone mediante endpoints específicos que reciben solicitudes y devuelven respuestas en formato JSON. Este enfoque facilita la integración con el sistema principal y otros microservicios, asegurando una interoperabilidad clara y escalable sin necesidad de exponer directamente la lógica interna del servicio.

- **Ejemplos de llamadas y respuestas; fragmentos Postman o Swagger.**



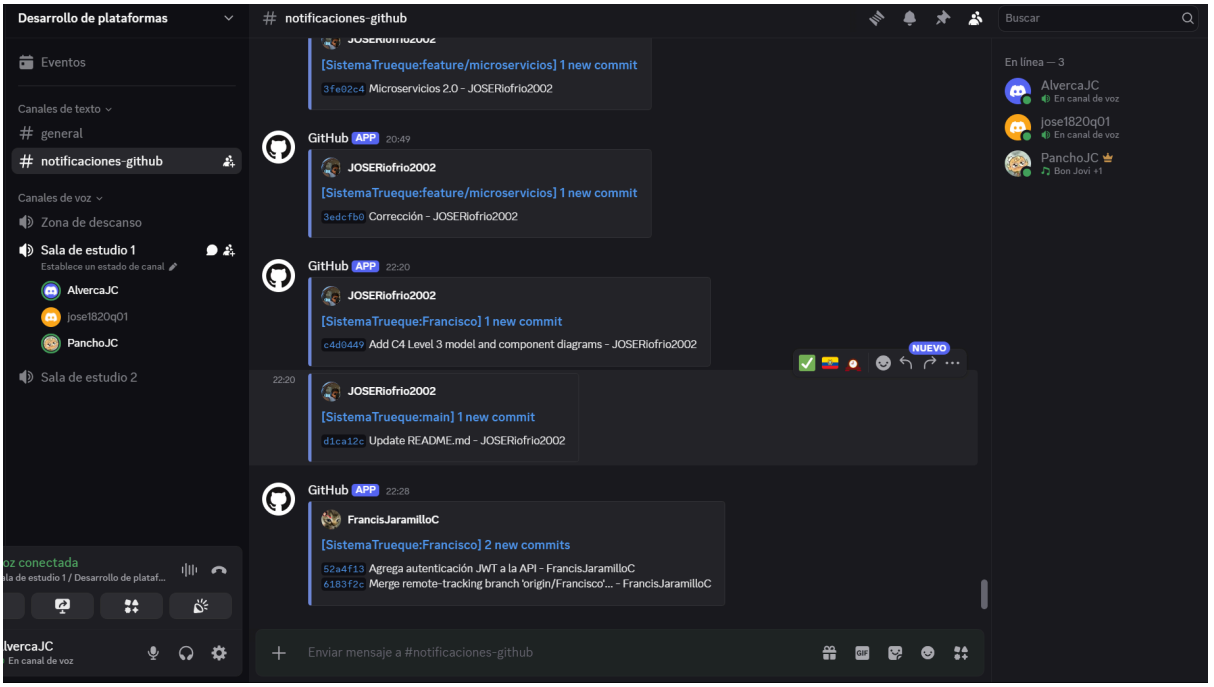


3.3 Herramientas colaborativas utilizadas

- **Describir el uso de Trello, Taiga, GitHub Projects, Discord u otras herramientas para la coordinación del equipo.**

Para la coordinación del equipo, utilizamos GitHub como la principal herramienta de trabajo colaborativo y gestión del código fuente, mediante este repositorio compartimos los avances del proyecto y llevamos un control de las distintas versiones de cada módulo, además, integramos el repositorio con Discord, lo que nos permite recibir alertas automáticas sobre los cambios realizados y mantener una comunicación fluida entre los miembros del equipo, como a su vez realizamos reuniones periódicas en Discord para organizar las tareas, evaluar el progreso y coordinar las responsabilidades de cada participante del proyecto.

- **Evidenciar cómo estas herramientas contribuyeron a la planificación y seguimiento de tareas.**



3.4 Buenas prácticas y aprendizajes

- **Identificar dos buenas prácticas aplicadas durante el desarrollo.**

Validación de datos mediante esquemas (Schema Validation): Se utilizó `CategoriaSchema` para asegurar que los datos recibidos en las solicitudes cumplan con la estructura y tipos esperados, evitando errores y garantizando la integridad de la información en la base de datos.

Separación de responsabilidades y modularidad: El microservicio mantiene una estructura clara y modular, separando la definición de rutas (`APIRouter`), los modelos de datos (`CategoriaSchema`) y la conexión a la base de datos (`conn`). Esto facilita el mantenimiento, la escalabilidad y la comprensión del código.

- **Reflexión personal sobre cómo la modularización mejora la escalabilidad y mantenimiento del proyecto.**

La modularización permite que un proyecto crezca y evolucione de manera ordenada, ya que divide el sistema en partes más pequeñas y manejables, cada módulo puede desarrollarse, probarse y actualizarse de forma independiente, sin afectar al resto del sistema, esto no solo facilita el mantenimiento, sino que también aumenta la escalabilidad, porque es posible mejorar o ampliar una parte específica sin tener que modificar toda la aplicación, además, la modularización promueve una mejor organización del código, una colaboración más eficiente entre los desarrolladores y una mayor capacidad para adaptarse a nuevos requerimientos o tecnologías, en conjunto, esta práctica convierte al sistema en una estructura más flexible, sostenible y preparada para el crecimiento.

4. **Anexos**

Repositorio:
FrancisJaramilloC/SistemaTrueque at Francisco