

## Desarrollo del Backend, Arquitectura Base y Modelo C4 del Sistema Multiplataforma.

Producto esperado:

Un repositorio remoto (GitHub o GitLab) correctamente estructurado y documentado que contenga:

1. **Backend funcional, con Endpoints REST implementados (usuarios, autenticación y entidades base).**
- **Autenticación**

POST	/api/auth/registro	Registrar un nuevo usuario
POST	/api/auth/login	Iniciar sesión
GET	/api/auth/me	Obtener información del usuario actual

- **Clientes**

GET	/api/clientes/	Get Clientes
POST	/api/clientes/	Create Cliente
GET	/api/clientes/{id}	Get Cliente
PUT	/api/clientes/{id}	Update Cliente
DELETE	/api/clientes/{cliente_id}	Delete Cliente

- **Trueques**

GET	/api/trueques/	Get Trueques
POST	/api/trueques/	Create Trueque
GET	/api/trueques/{id}	Get Trueque
DELETE	/api/trueques/{trueque_id}	Delete Trueque

- **Tipo identificación**

GET	/api/trueques/	Get Trueques
POST	/api/trueques/	Create Trueque
GET	/api/trueques/{id}	Get Trueque
DELETE	/api/trueques/{trueque_id}	Delete Trueque

- Personas

GET	/api/personas/	Get Personas
POST	/api/personas/	Create Persona
GET	/api/personas/{id}	Get Persona
PUT	/api/personas/{id}	Update Persona
DELETE	/api/personas/{persona_id}	Delete Persona

- Categorías

GET	/api/categorias/	Get Categorías
POST	/api/categorias/	Create Categoría
PATCH	/api/categorias/{id}	Patch Categoría
DELETE	/api/categorias/{categoria_id}	Delete Categoría

- Artículos

GET	/api/articulos/	Get Articulos
POST	/api/articulos/	Create Articulo
GET	/api/articulos/{id}	Get Articulo
DELETE	/api/articulos/{articulo_id}	Delete Articulo
DELETE	/api/articulos/secure-delete/{articulo_id}	Secure Delete Articulo

- Ofertas

GET	/api/ofertas/	Get Ofertas
POST	/api/ofertas/	Create Oferta
GET	/api/ofertas/{id}	Get Oferta
PUT	/api/ofertas/{id}	Update Oferta
DELETE	/api/ofertas/{oferta_id}	Delete Oferta

- Transacciones

POST	/api/transactions/api/transactions/{tx_id}/confirm	Confirm Delivery
GET	/api/transactions/api/transactions/{tx_id}/history	Get History
GET	/api/transactions/api/transactions	List Transactions

- Configuración del entorno, dependencias y estructura modular del proyecto.
- Dependencias

```
1 annotated-types==0.7.0
2 anyio==4.11.0
3 bcrypt==5.0.0
4 certifi==2025.10.5
5 cffi==2.0.0
6 click==8.3.0
7 colorama==0.4.6
8 cryptography==46.0.3
9 dnspython==2.8.0
10 email-validator==2.3.0
11 fastapi==0.119.0
12 fastapi-cli==0.0.13
13 fastapi-cloud-cli==0.3.1
14 greenlet==3.2.4
15 h11==0.16.0
16 httpcore==1.0.9
17 httptools==0.7.1
18 httpx==0.28.1
19 idna==3.11
20 Jinja2==3.1.6
21 mariadb==1.1.14
22 markdown-it-py==4.0.0
23 MarkupSafe==3.0.3
24 mdurl==0.1.2
25 packaging==25.0
26 passlib==1.7.4
27 pycparser==2.23
28 pydantic==2.12.2
29 PyJWT==2.8.0
30 python-jose[cryptography]==3.3.0
31 pydantic_core==2.41.4
32 Pygments==2.19.2
33 PyMySQL==1.1.2
34 python-dotenv==1.1.1
35 python-multipart==0.0.20
36 PyYAML==6.0.3
37 rich==14.2.0
38 rich-toolkit==0.15.1
39 rignore==0.7.1
40 sentry-sdk==2.42.0
```

- Integración inicial de un microservicio o módulo independiente.

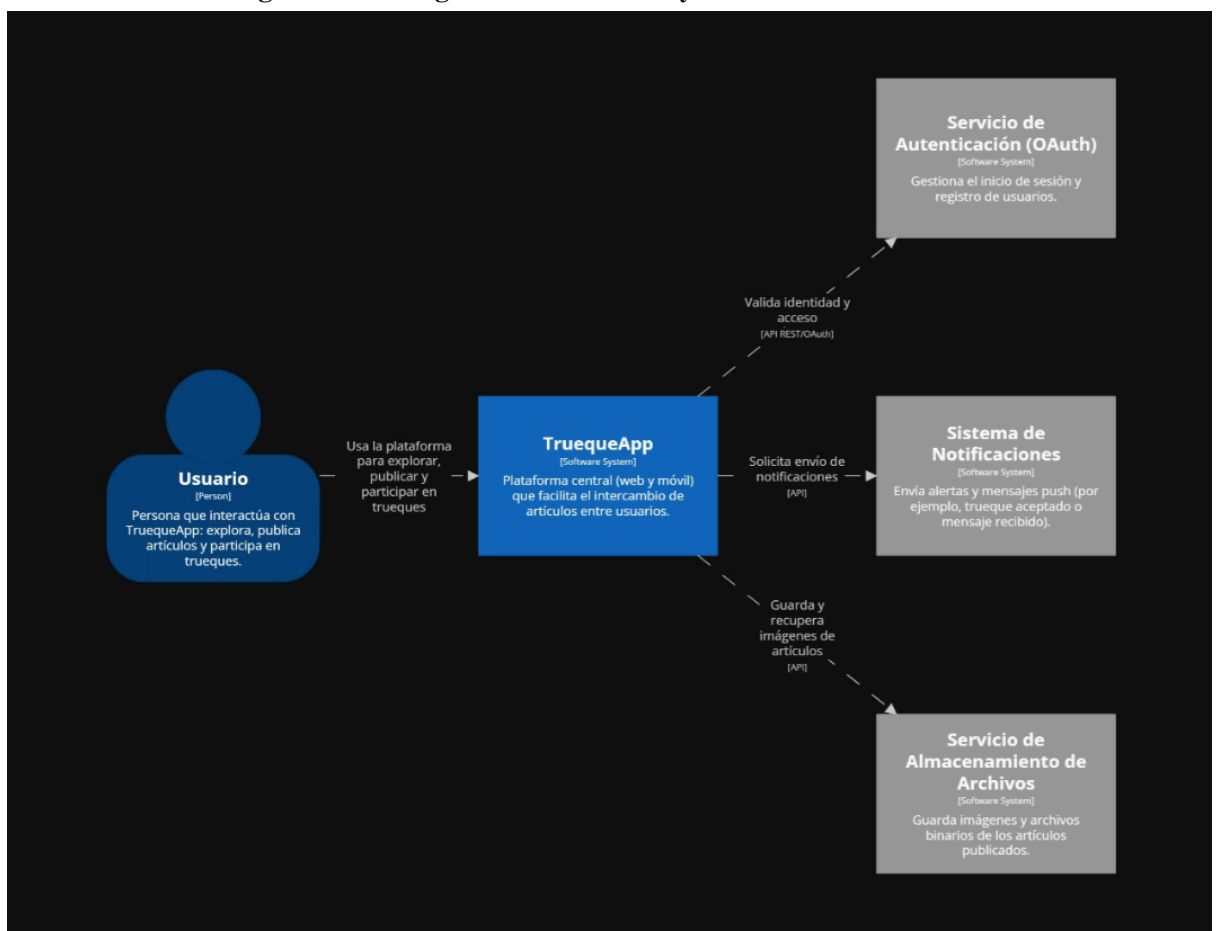
```

1 from fastapi import APIRouter, HTTPException
2 from categoria_schema import CategoriaSchema
3 from db import conn
4 from categoria import categorias
5
6 categoria_router = APIRouter()
7
8 @categoria_router.get("/api/categoria")
9 def get_categorias():
10     result = conn.execute(categorias.select()).fetchall()
11     return [dict(row._mapping) for row in result]
12
13 @categoria_router.post("/api/categoria/create")
14 def create_categoria(data_categoria: CategoriaSchema):
15     new_categoria = data_categoria.dict()
16     if new_categoria.get("id") is None:
17         new_categoria.pop("id", None)
18     conn.execute(categorias.insert().values(new_categoria))
19     return {"message": "categoria creada"}
20
21 @categoria_router.patch("/api/categoria/update/{id}")
22 def patch_categoria(id: int, data_categoria: CategoriaSchema):
23     data = data_categoria.dict(exclude_unset=True)
24     if "id" in data:
25         data.pop("id")
26     result = conn.execute(categorias.update().where(categorias.c.id == id).values(data))
27     if result.rowcount == 0:
28         return {"message": f"No se encontró categoria con id {id}"}
29     return {"message": "categoria actualizada correctamente"}
30

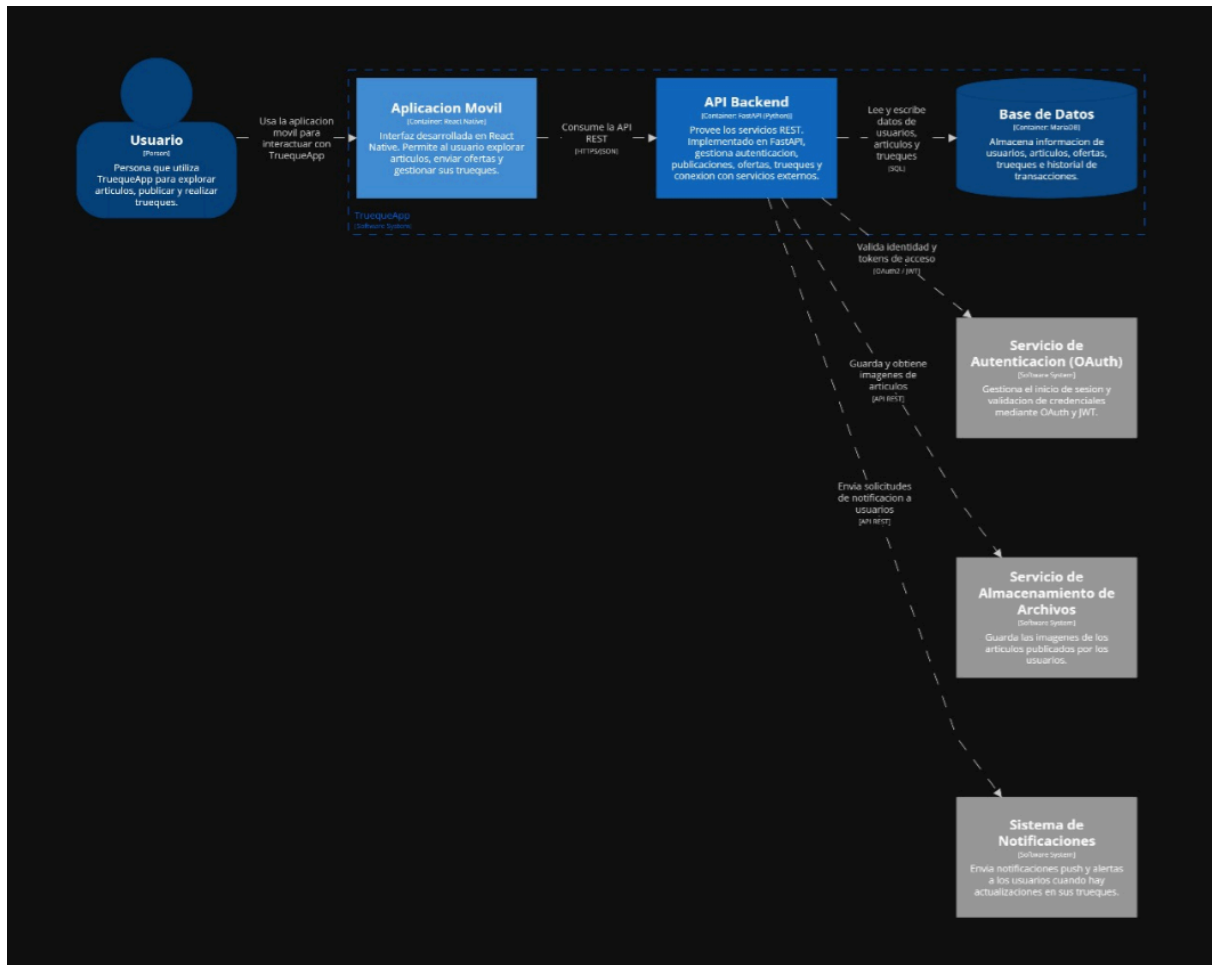
```

## 2. Modelo C4 – Arquitectura del sistema:

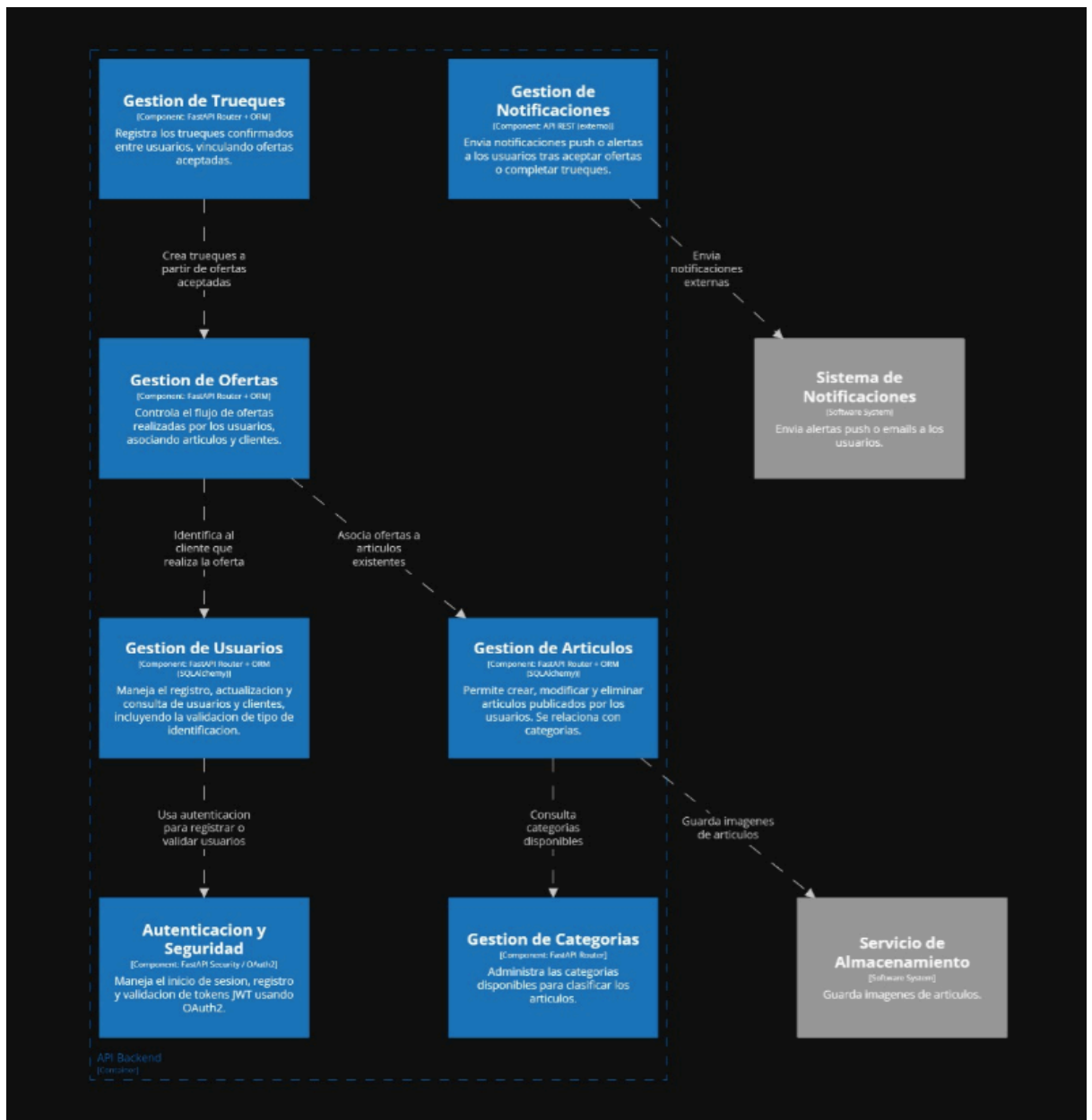
- Nivel 1: Context Diagram – visión general del sistema y actores externos.



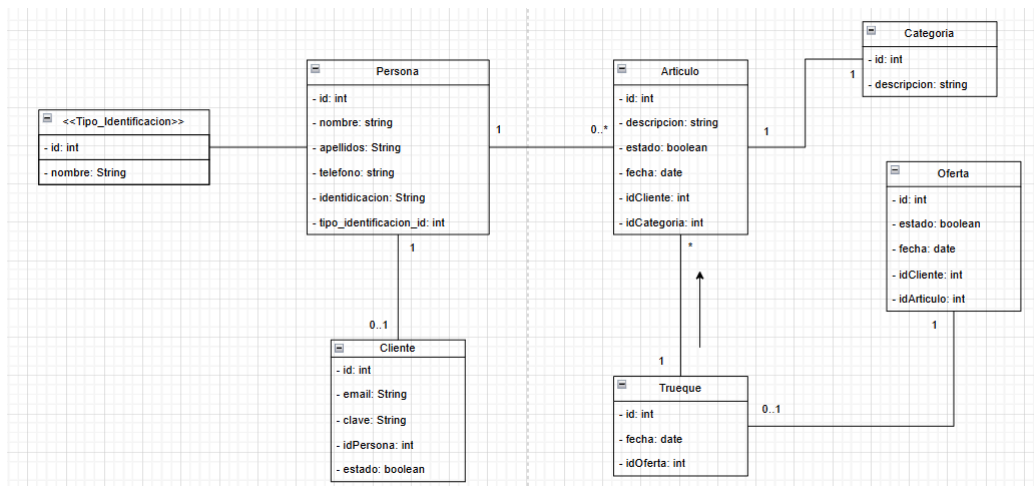
- Nivel 2: Container Diagram – relación entre backend, frontend y base de datos.



- **Nivel 3: Component Diagram – estructura interna del backend (controladores, servicios, repositorios).**



- Nivel 4: Code Diagram – diagramas de clase



### **3. Documentación técnica:**

- **Descripción de arquitectura y dependencias.**

TruequeApp es una plataforma web y móvil diseñada para permitir el intercambio (trueque) de artículos entre usuarios registrados.

Su arquitectura está basada en el modelo multicapa y modular, siguiendo el estilo arquitectónico de microservicios ligeros con separación de responsabilidades clara entre frontend, backend, base de datos y servicios auxiliares.

El sistema se compone de varios contenedores (módulos) independientes que interactúan entre sí mediante API REST y eventos asíncronos (notificaciones y mensajería).

Esta estructura favorece la escalabilidad, mantenibilidad y seguridad, permitiendo desplegar o actualizar cada módulo sin afectar al resto del sistema.

#### **Componentes Principales**

- **Frontend Web (WebApp)**

Tecnología: HTML5, CSS3, JavaScript (Vanilla JS)

Función: Es la capa de presentación del sistema. Proporciona una interfaz gráfica intuitiva, amigable y responsiva, accesible desde distintos dispositivos (computadoras, tabletas y teléfonos móviles).

Permite a los usuarios realizar las siguientes acciones:

Crear cuentas y autenticarse.

Publicar, editar o eliminar artículos.

Buscar artículos mediante filtros.

Gestionar propuestas de trueque.

Acceder a mensajería y notificaciones.

#### **Dependencias:**

Se comunica con el Backend (API REST) mediante peticiones HTTP/HTTPS (fetch o XMLHttpRequest).

Utiliza Bootstrap o CSS Grid/Flexbox para el diseño adaptable (responsivo).

Puede integrar JavaScript modular para separar lógicas (por ejemplo: [auth.js](#), [articles.js](#), [trades.js](#)).

- **Aplicación Móvil**

Tecnología: React Native.



Función: Permite a los usuarios interactuar desde dispositivos móviles con las mismas funcionalidades que la versión web.

#### **Dependencias:**

Consume el Backend/API Central.

Se comunica con el Servicio de Notificaciones para recibir alertas y mensajes en tiempo real.

- **Backend / API Central**

Tecnología: FastAPI.

Función: Es el núcleo lógico del sistema, encargado de procesar las solicitudes del frontend, aplicar las reglas de negocio y coordinar la comunicación entre los módulos internos.

#### **Responsabilidades:**

Autenticación (JWT / OAuth2).

Gestión de usuarios, artículos y trueques.

Envío de correos electrónicos de verificación.

Integración con servicios externos.

#### **Dependencias internas:**

Base de Datos: para persistir toda la información.

Servicio de Almacenamiento: para gestionar imágenes de los artículos.

Servicio de Búsqueda: para indexar artículos y ejecutar búsquedas rápidas.

Servicio de Notificaciones: para comunicación en tiempo real.

Sistema de Correo: para enviar correos de confirmación y notificación.

- **Base de Datos**

Tecnología: MariaDB

Función: Almacenar de forma estructurada toda la información del sistema, como usuarios, artículos, mensajes y trueques.

#### **Dependencias:**

Accedida únicamente por el Backend.

#### **4. Informe de seguridad:**

- **Evidencia de aplicación de principios OWASP Top 10, autenticación JWT u OAuth2 y cifrado de contraseñas.**

```

# CONFIGURACIÓN DE HASHEO DE CONTRASEÑAS
# bcrypt es un algoritmo de hasheo muy seguro y lento
pwd_context = CryptContext(schemes=["bcrypt"], deprecated="auto")

def hash_password(password: str) -> str:
    return pwd_context.hash(password)

def verify_password(plain_password: str, hashed_password: str) -> bool:
    return pwd_context.verify(plain_password, hashed_password)

# FUNCIONES PARA MANEJO DE JWT

def create_access_token(data: dict, expires_delta: Optional[timedelta] = None) -> str:
    # Copiamos la data para no modificar el original
    to_encode = data.copy()

    # Calculamos cuándo expira el token
    if expires_delta:
        expire = datetime.utcnow() + expires_delta
    else:
        expire = datetime.utcnow() + timedelta(minutes=ACCESS_TOKEN_EXPIRE_MINUTES)

    # Agregamos la fecha de expiración al token
    to_encode.update({"exp": expire})

    # Creamos el token firmado con nuestra clave secreta
    encoded_jwt = jwt.encode(to_encode, SECRET_KEY, algorithm=ALGORITHM)

    return encoded_jwt

def verify_token(token: str) -> Optional[dict]:
    try:
        # Intentamos decodificar el token
        payload = jwt.decode(token, SECRET_KEY, algorithms=[ALGORITHM])
        return payload
    except JWTError:
        # Si falla (token inválido, expirado, o modificado), retornamos None
        return None

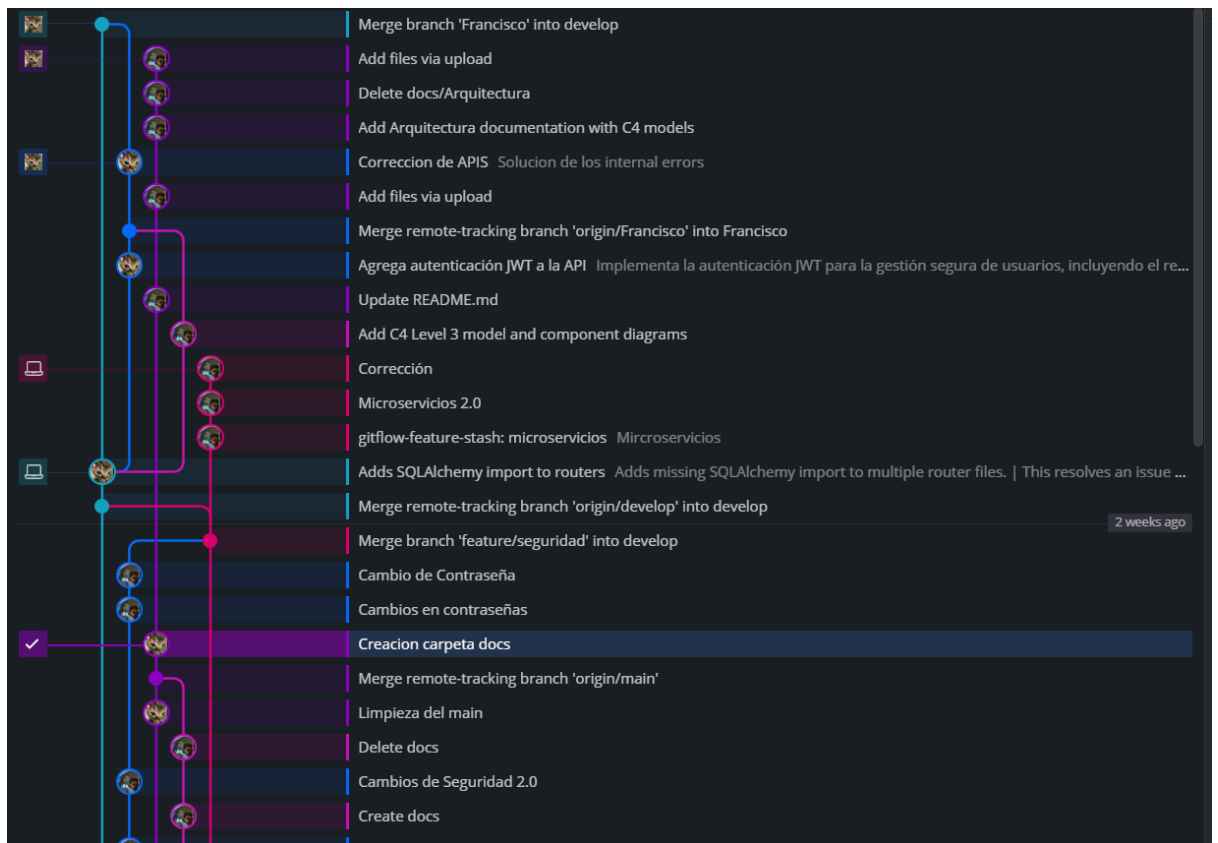
```

```

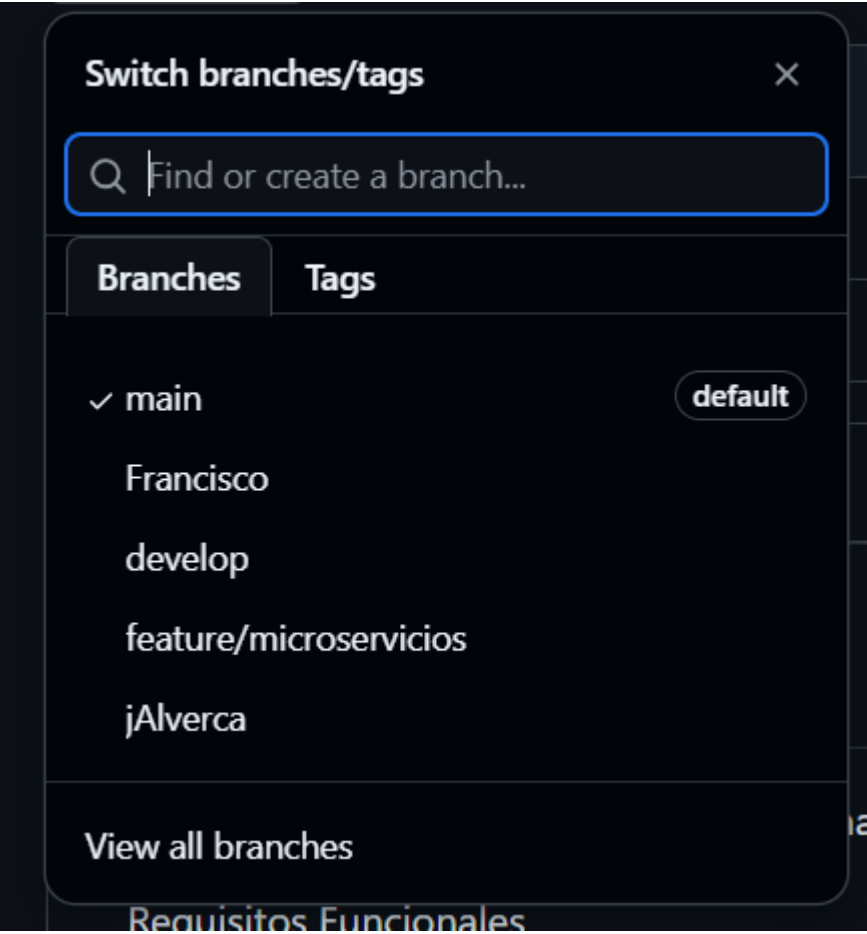
6 # CONFIGURACIÓN DEL ESQUEMA DE SEGURIDAD
7 # HTTPBearer es el esquema que espera un token en el header:
8
9 security = HTTPBearer()
10
11 def get_current_user(credentials: HTTPAuthorizationCredentials = Depends(security)) -> dict:
12     # Extraer el token del objeto credentials
13     token = credentials.credentials
14
15     # Verificar el token usando nuestra función de auth_utils
16     payload = verify_token(token)
17
18     # Si el token es inválido, verify_token retorna None
19     if payload is None:
20         raise HTTPException(
21             status_code=status.HTTP_401_UNAUTHORIZED,
22             detail="Token inválido o expirado. Por favor, inicia sesión nuevamente.",
23             headers={"WWW-Authenticate": "Bearer"}, # Estándar HTTP para autenticación
24         )
25
26     # Extraer el username del payload del token
27     # "sub" es el campo estándar para el identificador del usuario
28     username: Optional[str] = payload.get("sub")
29     if username is None:
30         raise HTTPException(
31             status_code=status.HTTP_401_UNAUTHORIZED,
32             detail="Token no contiene información válida del usuario.",
33             headers={"WWW-Authenticate": "Bearer"},
34         )
35     return payload
36
37
38 def get_optional_user(credentials: Optional[HTTPAuthorizationCredentials] = Depends(security)) -> Optional[dict]:
39     if credentials is None:
40         return None
41     token = credentials.credentials
42     payload = verify_token(token)
43     return payload if payload else None

```

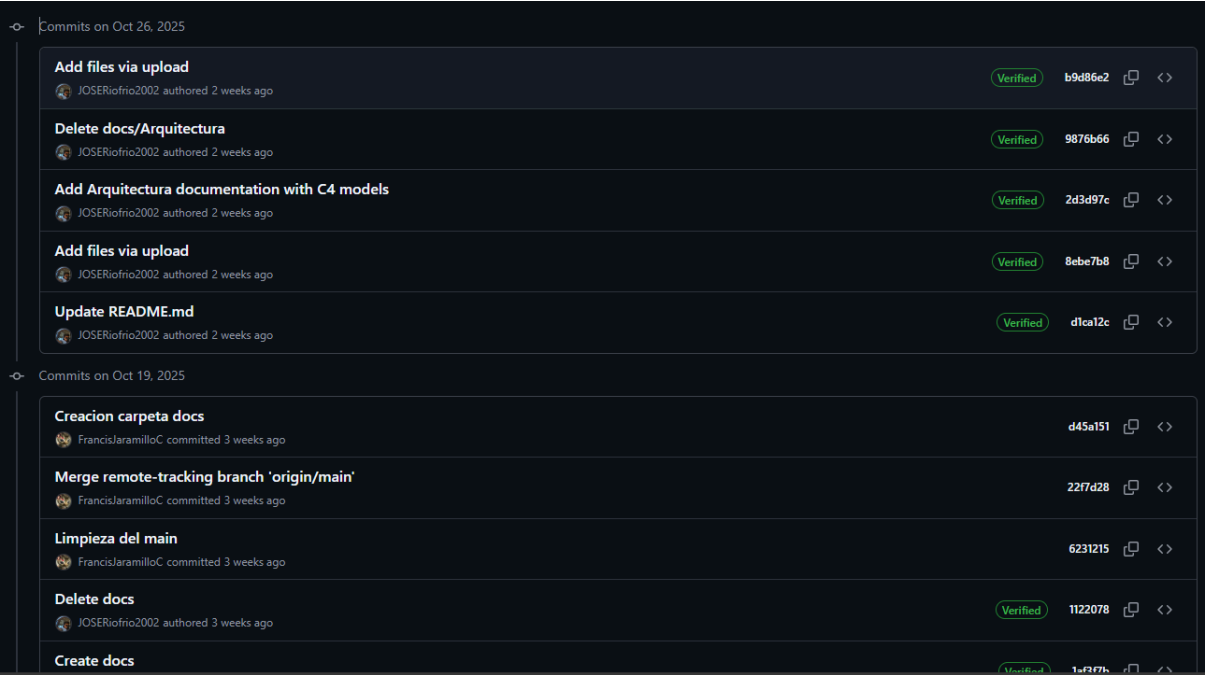
## 5. Control de versiones con GitKraken y flujo GitFlow:



































- **Ramas principales**



- Commits descriptivos y merges ordenados.



Commits on Oct 23, 2025		
Merge branch 'feature/seguridad' into develop	 JOSE Rios committed 2 weeks ago	 a6382bf 
Merge remote-tracking branch 'origin/develop' into develop	 Francis Jaramillo committed 2 weeks ago	 7e09ea7 
Adds SQLAlchemy import to routers ...	 Francis Jaramillo committed 2 weeks ago	 0b67828 
Commits on Oct 26, 2025		
Add C4 Level 3 model and component diagrams	 JOSE Rios authored 2 weeks ago	<span>Verified</span>  c4d0449 
Agrega autenticación JWT a la API ...	 Francis Jaramillo committed 2 weeks ago	 52a4f13 
Merge remote-tracking branch 'origin/Francisco' into Francisco	 Francis Jaramillo committed 2 weeks ago	 6183f2c 
Correccion de APIS ...	 Francis Jaramillo committed 2 weeks ago	 d9aa072 
Commits on Oct 30, 2025		
Merge branch 'Francisco' into develop	 Francis Jaramillo committed last week	 ec64079 
Commits on Nov 6, 2025		
Transaccioon ...	 jAverca committed 17 minutes ago	 3ed9f7d 
Transaction schema ...	 jAverca committed 16 minutes ago	 5b066bd 
Router de transaccion ...	 jAverca committed 15 minutes ago	 c39826c 