UNIVERSIDAD
CRISTÓBAL COLÓN

Entregables Finales

Reporte de actividad

Presenta:
NICOLAS CABALLERO ARZATE
ÁNGEL GONZÁLEZ DÍAZ

ING. SISTEMAS COMPUTACIONALES 7MO A

Asesor:
MTRO. JESUS LEONARDO LOPEZ HERNANDEZ

Veracruz, Veracruz
Noviembre, 2025

Arquitectura y Documentación

1. Arquitectura real de la REST API

Tecnologías principales: - FastAPI, SQLAlchemy, Pydantic, Uvicorn. Estructura y capas: - main.py: instancia FastAPI y registra rutas (/api). - models.py: define Usuario y Encuesta. - routes.py: define endpoints CRUD y dependencia get_db. - crud.py y schemas.py: separación lógica de acceso a datos y validación. Flujo: HTTP -> Router -> get_db(Session) -> crud -> DB -> JSON response.

EndPoints REST

Método	Ruta	Descripción
POST	/api/usuarios	Crear Usuario
GET	/api/usuarios	Listar Usuarios
GET	/api/usuarios/{usuario_id}	Obtener Usuario por ID
PUT	/api/usuarios/{usuario_id}	Actualizar Usuario
DELETE	/api/usuarios/{usuario_id}	Eliminar Usuario
POST	/api/encuestas	Crear Encuesta
GET	/api/encuestas	Listar Encuestas
GET	/api/encuestas/{encuesta_id}	Obtener Encuesta por ID
PUT	/api/encuestas/{encuesta_id}	Actualizar Encuesta
DELETE	/api/encuestas/{encuesta_id}	Eliminar Encuesta

2. Arquitectura real de la SOAP API

Tecnologías principales: Spyne y wsgiref. Estructura y flujo: - HelloService(ServiceBase) con método say_hello(name). - Application con Soap11 in/out protocol. - WsgiApplication envuelve la app. make_server('0.0.0.0',8001,wsgi_app) levanta el servicio en el puerto 8001.

3. Comparativa completa: REST vs SOAP

Formato y contrato: REST usa JSON y OpenAPI; SOAP usa XML y WSDL. Transporte: ambos via HTTP. SOAP usa envelopes y puede usar WS-Security. Acoplamiento: SOAP es más acoplado por contrato (WSDL); REST es más flexible. Errores: REST devuelve códigos HTTP; SOAP usa Faults. Usos: REST para apps modernas; SOAP para interoperabilidad y sistemas legados.

4. Pasos para ejecutar REST API

1) Crear y activar entorno virtual.
2) pip install -r requirements.txt.
3) export INIT_DB=true para crear tablas la primera vez.
4) uvicorn rest_api.app.main:app --reload --host 0.0.0.0 --port 800
5) Acceder a Swagger UI:http://localhost:8000/docs
6) Comandos curl ejemplo:

Crear: curl -X POSThttp://localhost:8000/api/usuarios

- -H "Content-Type: application/json"
- -d'{"nombre":"Juan","apellidos":"Perez", "email":"juan@test.com", "telefono":"123", "genero":"H"}'
- -Listar: curl "http://localhost:8000/api/usuarios"

5. Pasos para ejecutar tu SOAP API

1) Crear y activar entorno virtual.
2) pip install spyne
3) Ejecutar el script: python soap_api/app/main.py
4) Acceder al servicio en http://localhost:8001
5) Ejemplo de request SOAP (say_hello):

POST http://localhost:8001/ with Content-Type: text/xml Body: Angel

6. Ejemplos reales de consumo

REST (curl): - Crear usuario: curl -X POST http://localhost:8000/api/usuarios
-H "Content-Type: application/json"
-d'{"nombre":"Ana","apellidos":"Lopez", "email":"ana@test.com", "telefono":"111", "genero":"M"}'

SOAP (curl with raw XML): - Llamada say_hello: curl -X POST http://localhost:8001/
-H "Content-Type: text/xml; charset=utf-8" -d 'Angel'.

Retos y conclusiones

Esta arquitectura es una estrategia comercialmente potente pero técnicamente peligrosa.

Al ofrecer REST y SOAP simultáneamente, la plataforma garantiza una versatilidad total: puede conectarse tanto con aplicaciones móviles modernas (vía REST) como con sistemas empresariales rígidos (vía SOAP). Básicamente, no pierde ningún cliente por problemas de compatibilidad.

Sin embargo, conectar dos servicios independientes a la misma base de datos crea una trampa de mantenimiento. Es casi seguro que existe lógica duplicada, lo que significa que cualquier cambio en las reglas de negocio (como una nueva validación para las encuestas) obliga a modificar código en dos lugares distintos, duplicando el esfuerzo y aumentando el riesgo de errores o datos corruptos.