Facultad de Ingeniería
UNER

LIBRO VIRTUAL INTERACTICO DE BIOLOGÍA

INGENIERÍA DE SOFTWARE II

CORDOBA - ZABALA

INDICE

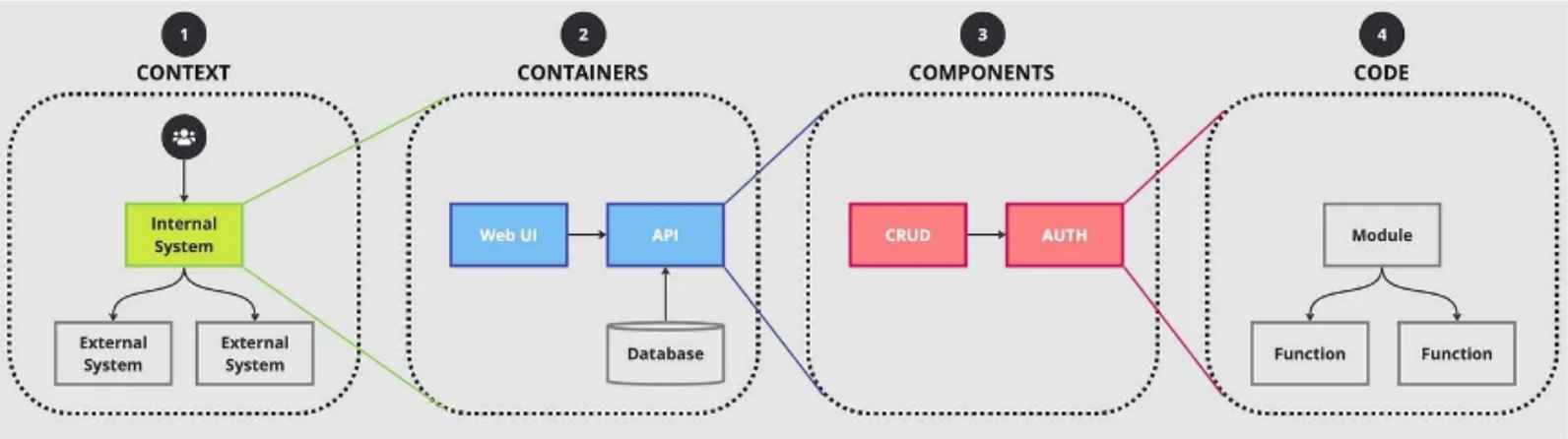
CONTEXTO DEL PROYECTO

El proyecto surge ante la necesidad de mejorar la enseñanza de biología.
El sistema ofrece una plataforma web educativa que integra textos, simulaciones, modelos 3D y cuestionarios interactivos.

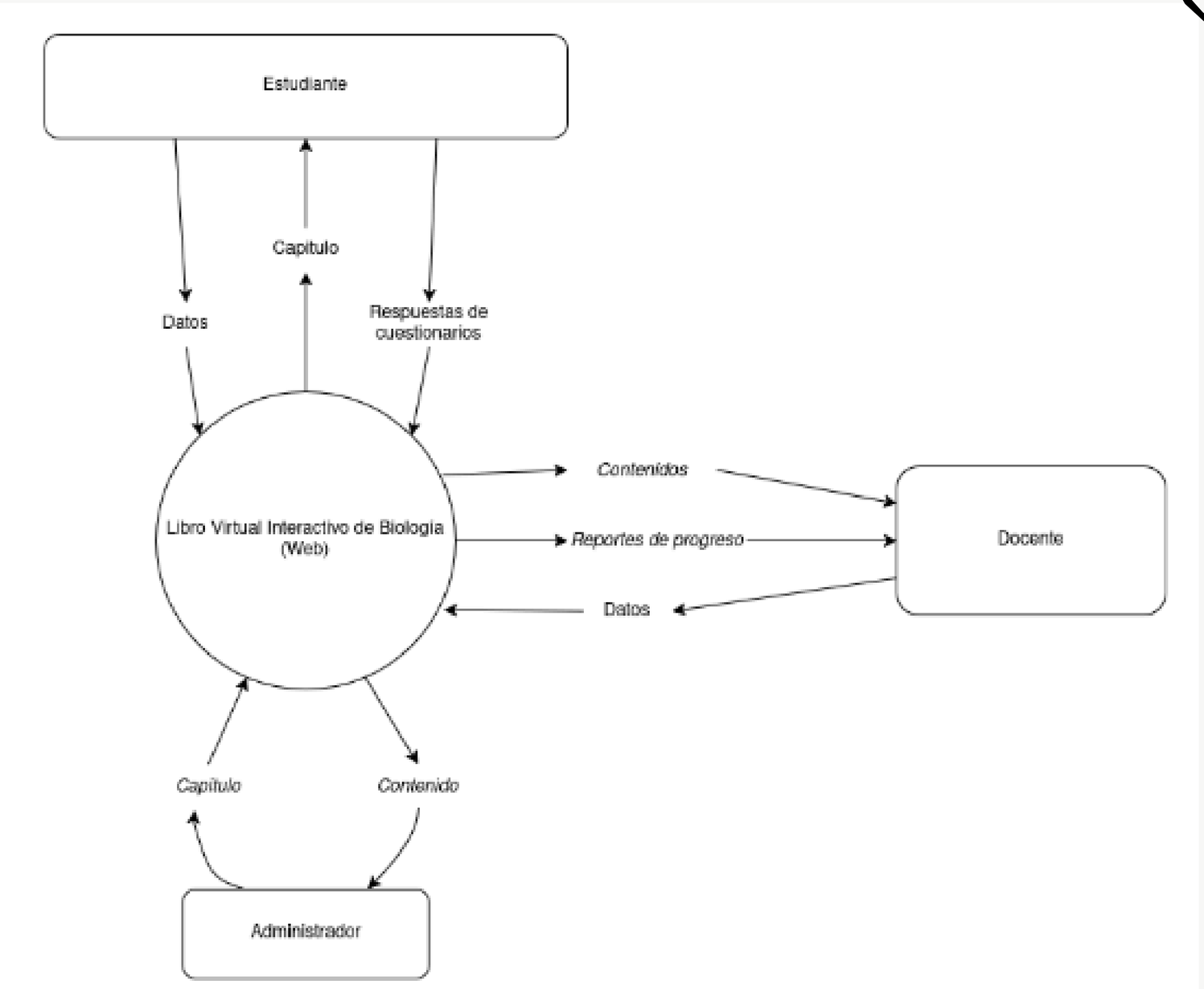
Usuarios principales del sistema:

- Estudiantes
- Docentes
- Administradores

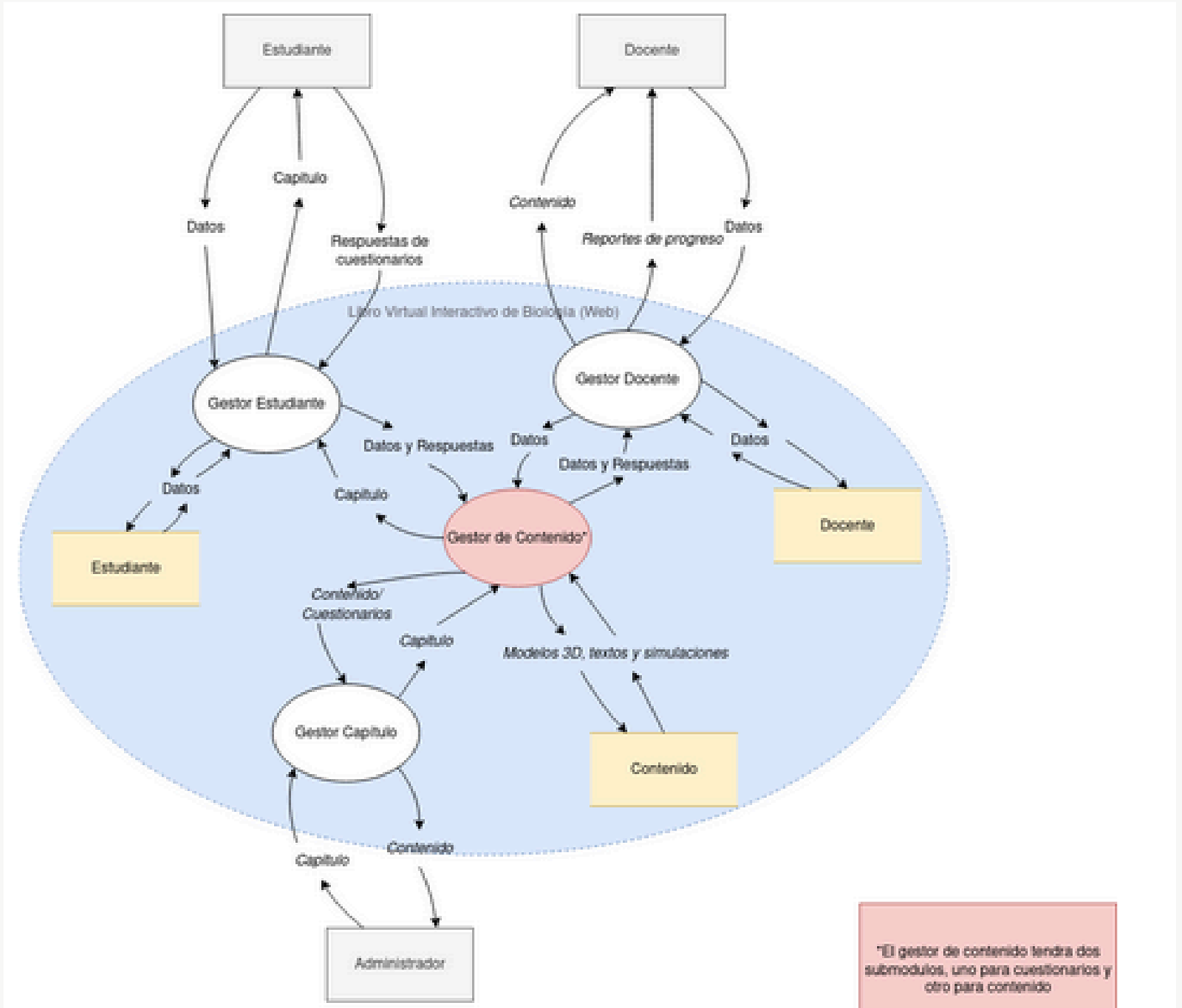
DIAGRAMA C4


1
CONTEXT
Internal System
External System
External System
2
CONTAINERS
Web UI
API
Database
3
COMPONENTS
CRUD
AUTH
4
CODE
Module
Function
Function

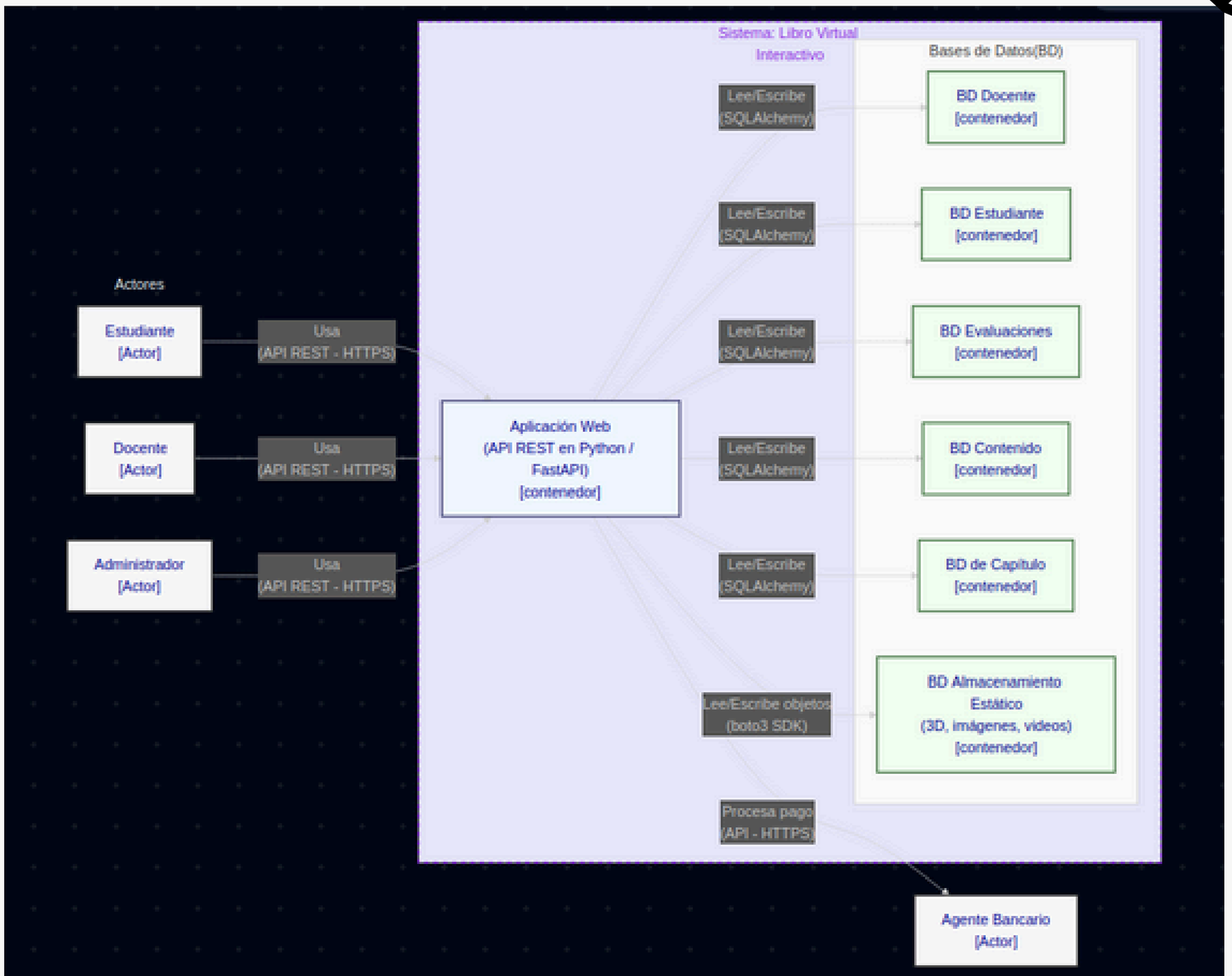
C4 – CONTEXTO NIVEL 0



C4 – CONTEXTO NIVEL 1



C4 – CONTENEDORES



C4 – COMPONENTES

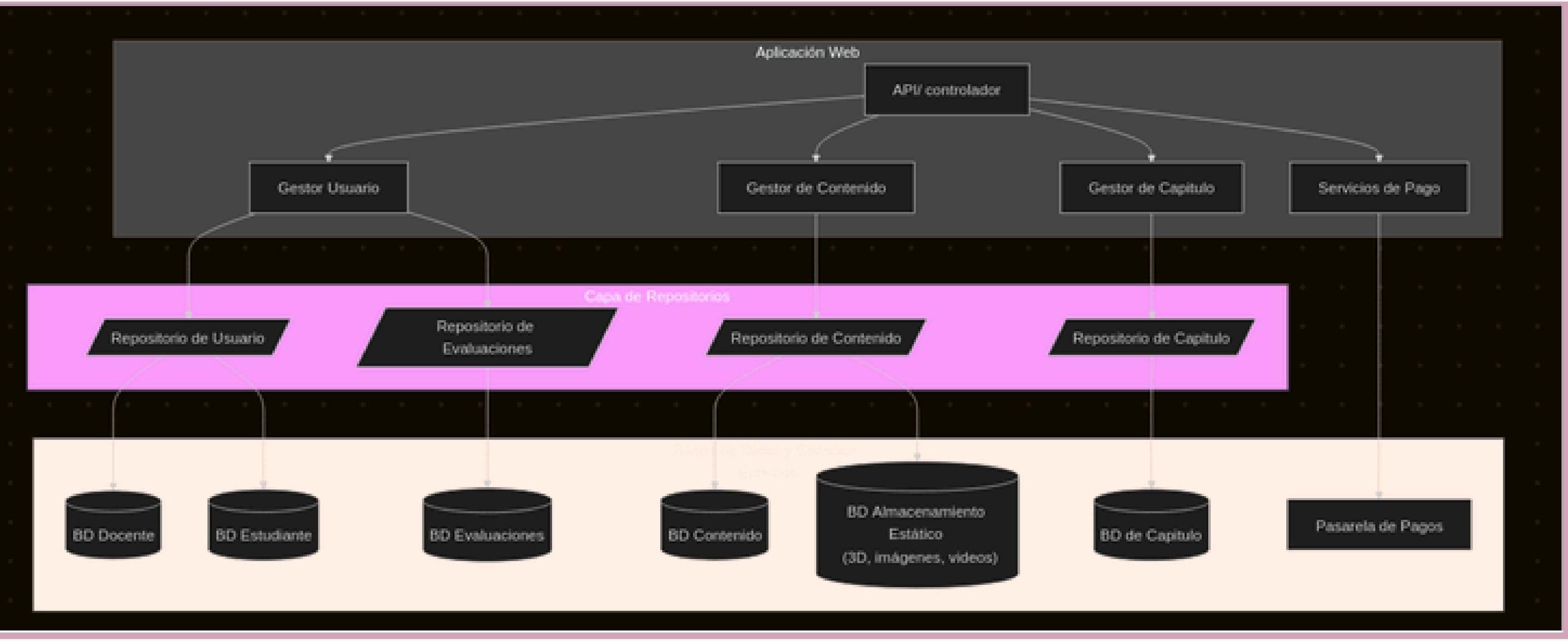


DIAGRAMA DE CLASES UML

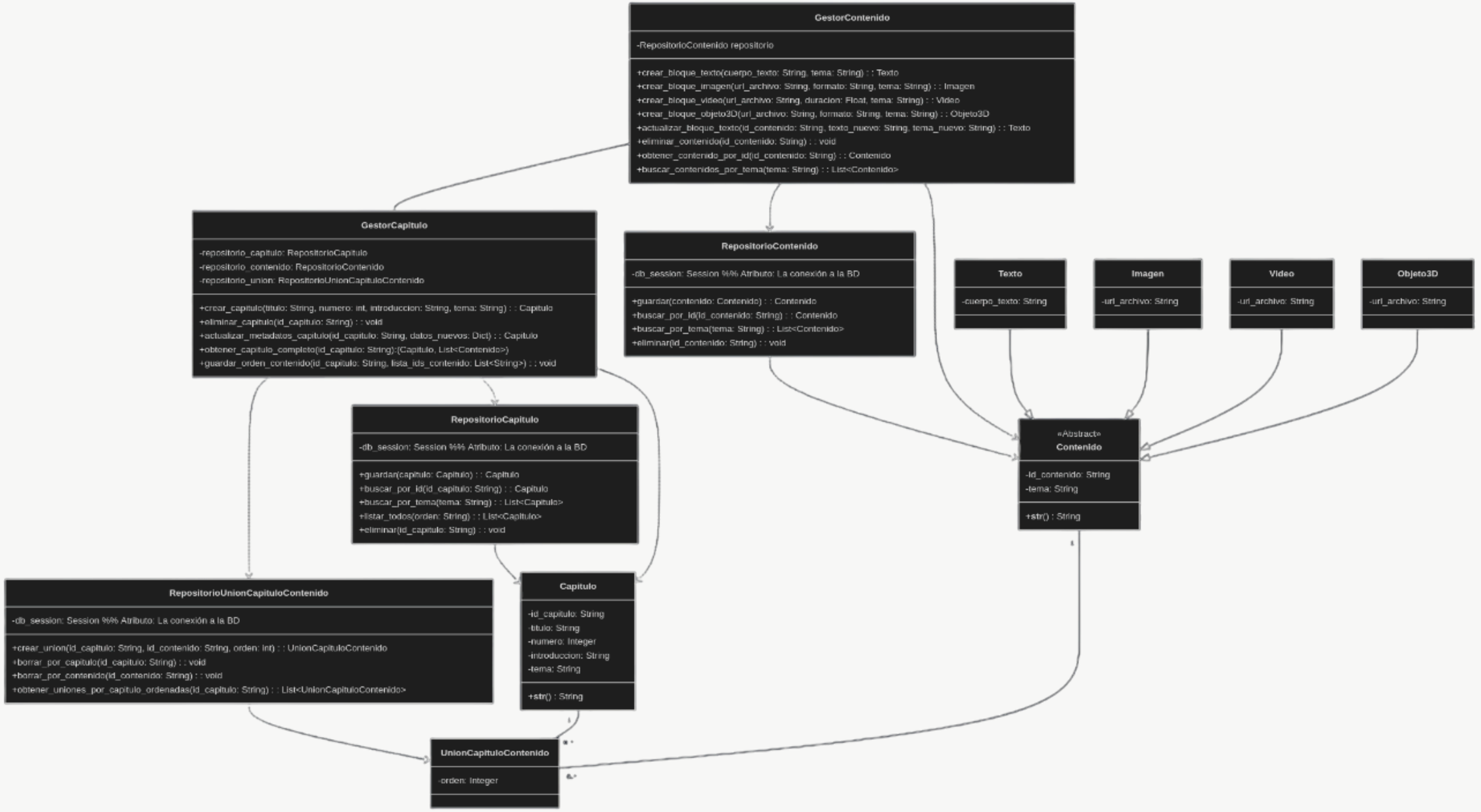
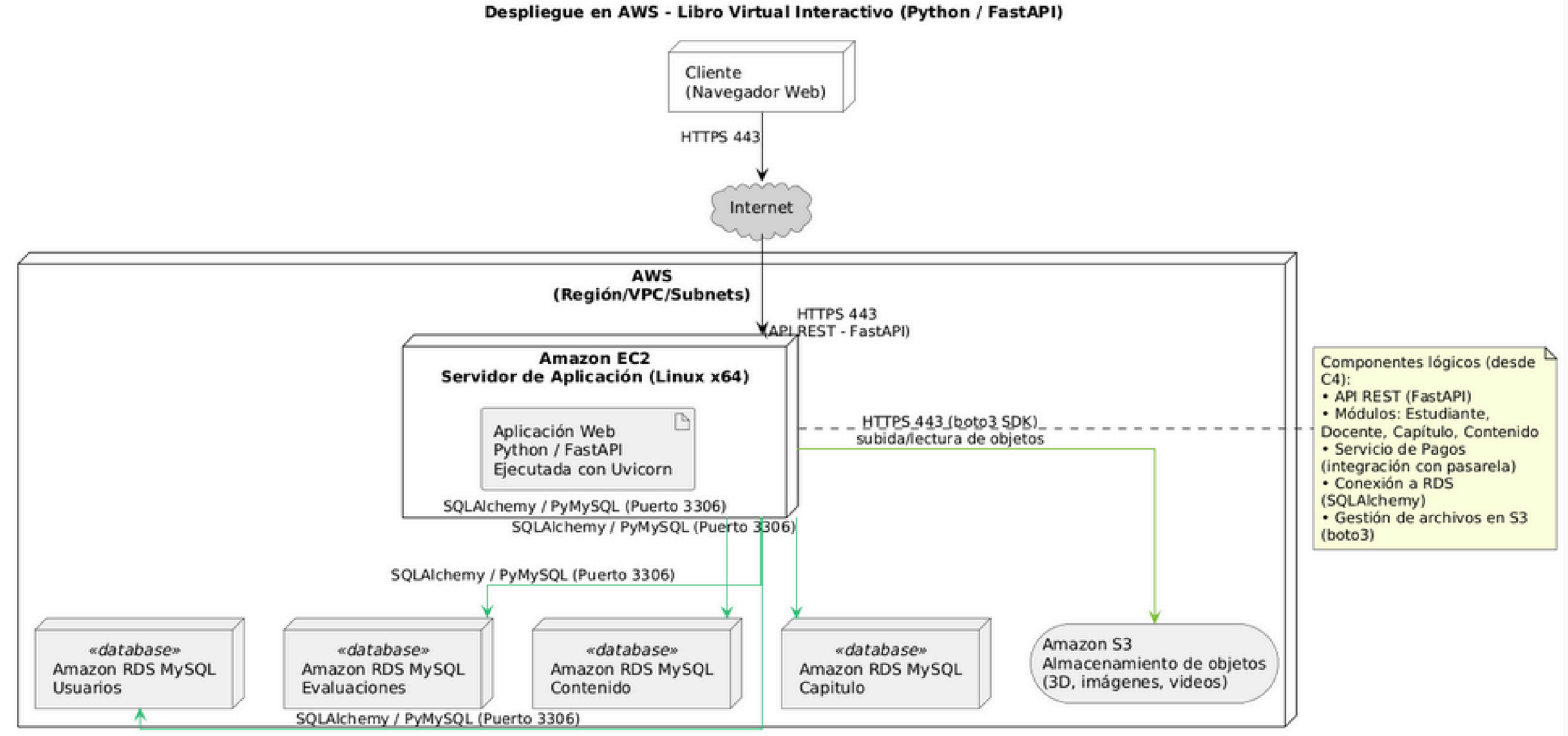
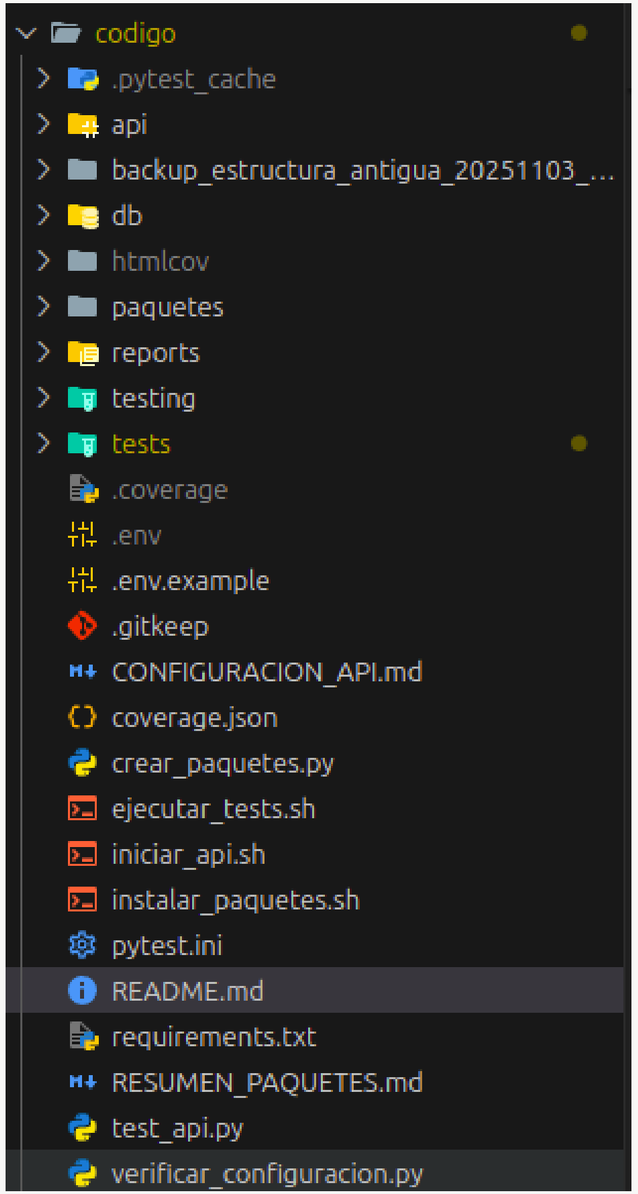


DIAGRAMA DE DESPLIEGUE



C4 – CÓDIGO

C4 – CÓDIGO



📋 Requisitos

- Python >= 3.8
- SQLAlchemy >= 2.0.0
- PyMySQL >= 1.1.0
- MySQL 8.0 (AWS RDS)

📚 Documentación

- **paquetes/README.md**: Arquitectura de paquetes completa
- **db/README.md**: Estructura de bases de datos
- **db/ESQUEMAS.md**: Esquemas detallados de tablas
- **db/DIAGRAMAS.md**: Diagramas de relaciones
- **ARQUITECTURA.md**: Visión general del sistema

🔧 Scripts Útiles

Script	Descripción
`instalar_paquetes.sh`	Instala todos los paquetes
`verificar_paquetes.py`	Verifica instalación correcta
`ejemplo_paquetes.py`	Ejemplos de uso
`crear_paquetes.py`	Regenera estructura de paquetes
`db/crear_tablas.py`	Crea tablas en MySQL
`db/test_conexiones.py`	Verifica conexión a BD

👥 Autores

Anibal Cordoba & Zabala

C4 – CÓDIGO


Libro Interactivo
Sistema de Gestión de Capítulos de Biología
Crear Capítulo
Ver Capítulos
Desarrollado por Anibal Cordoba & Matias Zabala
v2.0 - Arquitectura de Paquetes


Lista de Capítulos
Inicio
Nuevo Capítulo
Capítulo 1
Traducción
Mecanismo de traducción
Vamos a ver que es la traducción e imágenes que nos ayudarán a entender mejor
Eliminar


TESTING

TESTING

🧪 Testing

Estado Actual: ✅ 115/115 tests pasando (100%)

Métrica	Valor
Tests Totales	115
Tests Pasando	115 (100%) ✅
Coverage Routers	100% ⭐
Coverage Models	89% ✅
Coverage Total	31%
Tiempo Ejecución	~3.7s

Suites de Tests Implementadas

- ✅ **CP01_01** (13 tests) - Visualizar capítulo publicado
- ✅ **CP01_02** (19 tests) - Manejo de errores y seguridad
- ✅ **CP02_01** (26 tests) - Crear capítulo
- ✅ **CP02_02** (15 tests) - Actualizar capítulo
- ✅ **CP02_03** (10 tests) - Eliminar capítulo
- ✅ **CP02_04** (12 tests) - Listar y filtrar capítulos
- ✅ **CP02_05** (8 tests) - Validaciones de estado
- ✅ **test_models** (12 tests) - Tests unitarios ORM

Ejecutar Tests

```
# Todos los tests
./ejecutar_tests.sh all

# Por caso de prueba
./ejecutar_tests.sh cp02_01    # Crear capítulo
./ejecutar_tests.sh cp02_02    # Actualizar capítulo
./ejecutar_tests.sh cp02_03    # Eliminar capítulo

# Ver reportes
xdg-open htmlcov/index.html
```

Documentación de Testing

- testing/GUIA_RAPIDA_TESTING.md - Guía rápida de ejecución
- testing/RESUMEN_COMPLETO_TESTING.md - Documento consolidado (115 tests)
- testing/ - Todos los reportes detallados

🎯 Caso de Prueba CP01_01

Información General

Campo	Valor
ID	CP01_01
Caso de Uso Relacionado	CU_01 Visualizar contenido
Descripción	Accede a un capítulo existente en estado publicado
Área Funcional	Contenidos
Funcionalidad	Lectura de capítulo

📊 Resumen Ejecutivo

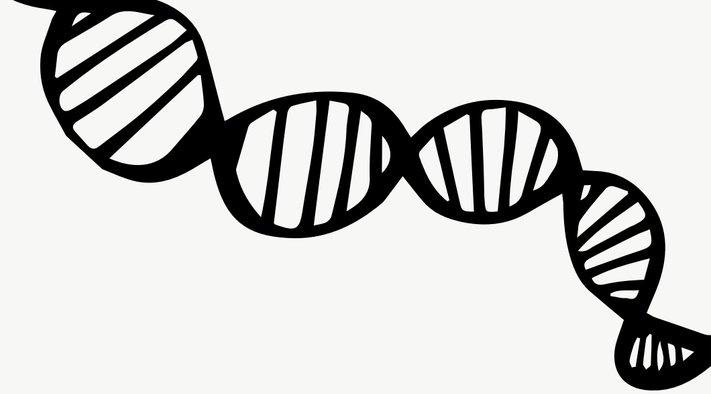
Métrica	Valor
Caso de Prueba	CP02_01 - Crear capítulo con campos mínimos válidos
Total Tests	26
Tests Pasados	✅ 26 (100%)
Tests Fallidos	❌ 0 (0%)
Tiempo Ejecución	1.07s
Coverage Total	30%

TESTING

```python
import pytest
from fastapi import status


class TestCP01_01_VisualizarCapituloPublicado:
    """
    Suite de tests para el caso de prueba CP01_01.
    Verifica la visualización exitosa de un capitulo publicado.
    """

    def test_visualizar_capitulo_publicado_exitoso(self, client, capitulo_publicado):
        """
        Test Principal CP01_01: Visualizar capitulo publicado con éxito.

        GIVEN: Existe un capitulo en estado PUBLICADO en la base de datos
        WHEN: Se realiza una petición GET a /capitulos/{id}
        THEN:
            - Se retorna status 200 OK
            - Se obtienen todos los campos del capitulo
            - El titulo se muestra correctamente
            - El número se muestra correctamente
            - La introducción se muestra correctamente
            - El estado es PUBLICADO
        """
        # Arrange
        capitulo_id = capitulo_publicado.id_capitulo

        # Act
        response = client.get(f"/api/capitulos/{capitulo_id}")

        # Assert
        assert response.status_code == status.HTTP_200_OK, \
            "Debe retornar 200 OK al visualizar un capitulo publicado"

        data = response.json()

        # Verificar que se retornan todos los campos esperados
        assert "id_capitulo" in data, "Debe incluir el ID del capitulo"
        assert "titulo" in data, "Debe incluir el titulo"
        assert "numero" in data, "Debe incluir el número"
        assert "introduccion" in data, "Debe incluir la introducción"
        assert "tema" in data, "Debe incluir el tema"
        assert "estado" in data, "Debe incluir el estado"
        assert "fecha_creacion" in data, "Debe incluir fecha de creación"

```

```python
import pytest
from fastapi import status
import uuid


class TestCP02_01_CrearCapituloExitoso:
    """
    Suite de tests para creación exitosa de capitulos.
    Valida el flujo completo de alta de capitulo.
    """

    def test_crear_capitulo_campos_minimos(self, client, sample_capitulo_data):
        """
        Test Principal CP02_01: Crear capítulo con campos mínimos válidos.

        GIVEN: Datos mínimos válidos (titulo, numero, tema)
        WHEN: Se realiza POST a /api/capitulos/
        THEN:
            - Status 201 Created
            - Capitulo guardado en BD
            - ID generado automáticamente
            - Fechas de creación/modificación generadas
        """
        # Arrange
        capitulo_data = {
            "titulo": "Nuevo Capitulo de Prueba",
            "numero": 999,
            "tema": "Testing Creación",
            "introduccion": "Introducción del nuevo capítulo"
        }

        # Act
        response = client.post("/api/capitulos/", json=capitulo_data)

        # Assert
        assert response.status_code == status.HTTP_201_CREATED, \
            "Debe retornar 201 Created al crear un capítulo"

```

CONCLUSIONES



- DECISIONES DE DISEÑO
- DIFICULTADES
- APRENDIZAJES OBTENIDOS
- VISTA A FUTURO

FIN

¡GRACIAS POR ESCUCHAR!