**🛫 SISTEMA DE GESTIÓN DE VUELOS – LISTA DOBLEMENTE ENLAZADA + ORM + API REST**

Este proyecto implementa un sistema completo para la gestión de vuelos en un aeropuerto, combinando una estructura de datos personalizada (Lista Doblemente Enlazada), persistencia con SQLAlchemy y una API REST desarrollada con FastAPI.

### 🔧 CARACTERÍSTICAS PRINCIPALES

#### 📦 Estructura TDA (Lista Doblemente Enlazada)

Definida en tda\_lista.py, esta estructura permite manejar dinámicamente la lista de vuelos con operaciones eficientes de inserción, eliminación y navegación:

* **Agregar vuelos al inicio o final de la lista.**
* **Eliminar vuelos según su posición.**
* **Obtener vuelos por índice.**
* **Vaciar la lista completamente.**
* **Iterar en ambos sentidos.**

#### 🧠 Lógica de Gestión de Vuelos

El archivo lista\_vuelos.py integra la estructura TDA con funcionalidades avanzadas:

* **Insertar vuelos con historial de cambios (undo/redo).**
* **Eliminar vuelos de manera reversible.**
* **Consultar vuelos según su estado (programado, emergencia, retrasado).**
* **Modificar vuelos sin perder versiones anteriores.**

#### 🗃️ ORM con SQLAlchemy

El archivo models.py define los modelos y la estructura de la base de datos:

* **Vuelo:** Representa cada vuelo (ID, código, estado, hora, origen, destino).
* **EstadoVuelo:** Enumeración con tres estados posibles:
  + programado
  + emergencia
  + retrasado
* **Nodo:** Clase auxiliar para la estructura de lista (no persistente en base de datos).

#### 🌐 API REST con FastAPI

La API definida en app.py permite interactuar con el sistema:

* **Registrar vuelos (POST).**
* **Eliminar vuelos por ID (DELETE).**
* **Listar todos los vuelos o filtrar por estado (GET).**
* **Realizar operaciones con historial (ej: deshacer, rehacer).**

### ⚙️ REQUISITOS

* Python 3.x
* Dependencias listadas en requirements.txt:
  + fastapi
  + uvicorn
  + sqlalchemy
  + pydantic