Mis Viajes

Informe de viajes para el mes: 3

ID: 2, Cliente: prueba1

Destino: Italia, Salida: 2025-03-03, Regreso: 2025-03-05, Precio: 275.0

ID: 3, Cliente: prueba1

Destino: Italia, Salida: 2025-03-03, Regreso: 2025-03-03, Precio: 275.0

ID: 5, Cliente: prueba1

Destino: Italia, Salida: 2025-03-03, Regreso: 2025-03-03, Precio: 275.0

ID: 6, Cliente: prueba1

Destino: Italia, Salida: 2025-03-03, Regreso: 2025-03-03, Precio: 275.0

ID: 7, Cliente: prueba1

Destino: Francia, Salida: 2025-03-03, Regreso: 2025-03-03, Precio: 270.0

ID: 8, Cliente: prueba1

Destino: Italia, Salida: 2025-03-03, Regreso: 2025-03-03, Precio: 375.0

ID: 9, Cliente: prueba1

Destino: Italia, Salida: 2025-03-03, Regreso: 2025-03-03, Precio: 300.0

ID: 10, Cliente: prueba1

Destino: España, Salida: 2025-03-08, Regreso: 2025-03-08, Precio: 240.0

ID: 11, Cliente: bbbb

Destino: Francia, Salida: 2025-03-08, Regreso: 2025-03-15, Precio: 198.0

Explicación del código

El código maneja operaciones CRUD (Crear, Leer, Actualizar y Eliminar) en una base de datos SQLite para gestionar información de viajes. Se usan consultas SQL con la biblioteca sqlite3 en Python.

---

1. getMisViajes(email)

¿Para qué sirve?

Esta función obtiene todos los viajes asociados a un cliente según su correo electrónico.

¿Cómo lo hace?

* Conexión a la base de datos con sqlite3.connect("viajes.db").
* Se crea un cursor para ejecutar la consulta SQL.
* Consulta SQL con JOINs para obtener información de varias tablas relacionadas:
* viaje (tabla principal)
* cliente (para obtener el nombre del cliente)
* vuelo (para relacionar el viaje con un vuelo)
* destino (para saber a qué destino pertenece el vuelo)
* Se ejecuta la consulta con el email como parámetro.
* Se obtiene el resultado con fetchall().
* Se cierra la conexión y se devuelve la lista de viajes.

Resultado de la consulta

Viajes encontrados:

[(1, 'prueba1', 'Francia', '2025-02-28', '2025-02-28', 198.0),
(2, 'prueba1', 'Italia', '2025-03-03', '2025-03-05', 275.0),
(3, 'prueba1', 'Italia', '2025-03-03', '2025-03-03', 275.0),
(4, 'prueba1', 'Italia', '2025-02-27', '2025-03-03', 275.0),
(5, 'prueba1', 'Italia', '2025-03-03', '2025-03-03', 275.0),
(6, 'prueba1', 'Italia', '2025-03-03', '2025-03-03', 275.0),
(7, 'prueba1', 'Francia', '2025-03-03', '2025-03-03', 270.0),
(8, 'prueba1', 'Italia', '2025-03-03', '2025-03-03', 375.0),
(9, 'prueba1', 'Italia', '2025-03-03', '2025-03-03', 300.0)]

---

2. putMisViajes(nueva_fecha_salida, nueva_fecha_regreso, viaje_id)

¿Para qué sirve?

Actualiza las fechas de salida y regreso de un viaje existente.

¿Cómo lo hace?

* Conexión a la base de datos.
* Se ejecuta una consulta SQL UPDATE con los nuevos valores.
* Se confirman los cambios con commit().
* Se cierra la conexión.

---

3. delMisViajes(viaje_id)

¿Para qué sirve?

Elimina un viaje de la base de datos según su ID.

¿Cómo lo hace?

* Conexión a la base de datos.
* Se ejecuta una consulta SQL DELETE con el ID del viaje.
* Se confirma la eliminación con commit().
* Se cierra la conexión.

---

4. insertar_viaje(cliente_email, vuelo_id, fecha_salida, fecha_regreso, precio)

¿Para qué sirve?

Inserta un nuevo viaje en la base de datos.

¿Cómo lo hace?

Obtiene el último ID de viaje con obtener_ultimo_id_viaje() y lo incrementa en 1.

* Ejecuta un INSERT INTO en la tabla viaje con los valores recibidos.
* Guarda los cambios con commit().
* Cierra la conexión.

Tecnologías utilizadas

* Python: Lenguaje de programación utilizado para manejar la lógica del sistema.
* SQLite (sqlite3): Base de datos ligera y embebida en el proyecto.
* SQL: Lenguaje de consulta para manejar la base de datos.