Itinerary Management System API

By Abdulmalik Olumoh

Project Outline

1. Problem Identification

Managing travel itineraries often requires juggling various platforms to keep track of flights, accommodations, and activities. This leads to inefficiency and disorganization, especially when plans change. The **Itinerary Management System API** aims to solve this by providing a centralized platform where users can easily manage and update their complete travel itineraries in one place.

2. Scope Definition

- **Core Features**:
  - **CRUD Operations**: Users will be able to Create, Read, Update, and Delete (CRUD) flights, accommodations (stays), and activities.
  - **Itinerary Management**: Users can manage an itinerary by grouping multiple flights, stays, and activities.
- **Additional Features for Future Versions**:
  - **User Authentication**: Add user accounts to store itineraries for future trips.
  - **Collaborative Itineraries**: Enable multiple users to contribute to a shared itinerary.
  - **API Integrations**: Integrate with external services for live flight updates and activity recommendations.

3. Target Audience

- **Primary Users**: Individual travelers and small groups (families or friends) who want a simple tool to manage their travel plans.
- **Secondary Users**: Small travel agencies that could use the API to organize and store itineraries for their clients.

4. Use Cases

1. **Creating a New Itinerary**: A traveler creates a new itinerary for an upcoming trip, adding flights, stays, and activities as they book them.
2. **Updating a Flight**: A traveler modifies their itinerary when their flight is rescheduled.
3. **Deleting an Activity**: A traveler removes a previously scheduled activity that they no longer wish to attend.
4. **Viewing Itinerary Details**: A user accesses a full view of their itinerary, including all flights, accommodations, and activities.

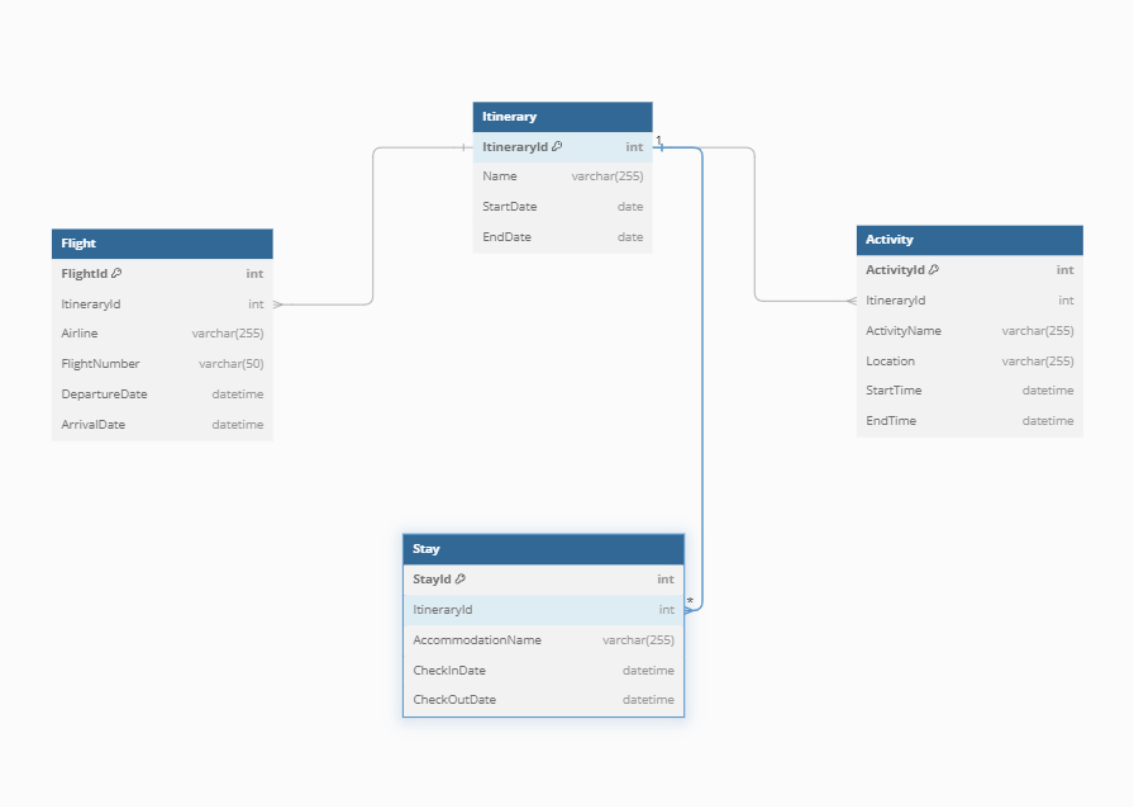
5. ER Diagram

The data model will have at least three core entities: **Itinerary**, **Flight**, **Stay**, and **Activity**. The relationships between them are as follows:

- An **Itinerary** can contain multiple **Flights**, **Stays**, and **Activities**.
- Each **Flight**, **Stay**, or **Activity** belongs to a specific **Itinerary**.

ER Diagram Structure:

- **Itinerary**: ItineraryId, Name, StartDate, EndDate
- **Flight**: FlightId, ItineraryId, Airline, FlightNumber, DepartureDate, ArrivalDate
- **Stay**: StayId, ItineraryId, AccommodationName, CheckInDate, CheckOutDate
- **Activity**: ActivityId, ItineraryId, ActivityName, Location, StartTime, EndTime


Itinerary
ItineraryId int
Name varchar(255)
StartDate date
EndDate date
Flight
FlightId int
ItineraryId int
Airline varchar(255)
FlightNumber varchar(50)
DepartureDate datetime
ArrivalDate datetime
Activity
ActivityId int
ItineraryId int
ActivityName varchar(255)
Location varchar(255)
StartTime datetime
EndTime datetime
Stay
StayId int
ItineraryId int
AccommodationName varchar(255)
CheckInDate datetime
CheckOutDate datetime
1
*