Done by Sych Denys

1. High-Level Overview of Functions and Use Cases

1.1 System functionality:

- **R1. Flight Search**

  Find available flights by date, route, destination.

- **R2. Booking**

  Choose flights, enter passenger data, and receive an e-ticket.

- **R3. Payments**

  Pay for tickets and additional services.

- **R4. Passenger Accounts**

  Store personal info, view reservations, manage bookings.

- **R5. Check-In**

  Online check-in and boarding passes.

- **R6. Administration**

  Airplane staff manages flights, prices, and system operations.

1.2 Use cases:

UC1.

Title	Search flights
Primary Actor	Passenger
Secondary Actors	Database
Success Scenario	1. Passenger enters search criteria (departure city, arrival city, date, route). 2. System validates search criteria. 3. System queries flight database for matching flights. 4. System displays list of available flights with details (times, prices, airlines, available seats). 5. Passenger can filter/sort results by price, time, or airline.
Extentions	3a. No flights found: System displays "No flights available for selected criteria" message and suggests alternative dates or nearby airports.

UC2.

Title	Book tickets
Primary Actor	Passenger
Secondary Actors	Database
Success Scenario	1. Passenger selects a flight from search results. 2. System displays flight details and booking form. 3. Passenger enters passenger information. 4. Passenger selects seat (if available). 5. System calculates total price including taxes/fees. 6. Passenger proceeds to payment. 7. Payment is processed successfully. 8. System generates booking confirmation with reference number. 9. System sends e-ticket to passenger's email. 10. System updates flight available seats count.
Extentions	7a. Payment failed: System displays error and returns to payment step. 8a. Seat selection unavailable: System returns to choosing seat

UC3.

Title	Make payment
Primary Actor	Passenger
Secondary Actors	Bank system
Success Scenario	1. Passenger selects payment method. 2. Passenger enters payment details securely. 3. System validates payment information format. 4. System sends payment request to bank system. 5. Payment is approved. 7. System records successful payment with transaction ID. 8. System updates booking status to "CONFIRMED". 9. System generates payment receipt.
Extentions	5a. Payment declined: System informs passenger and suggests alternative payment method.

UC4.

Title	Manage passenger account
Primary Actor	Passenger
Secondary Actors	Database
Success Scenario	1. Passenger logs into account. 2. System displays dashboard with options: View Profile, Booking History, Preferences. 3. Passenger selects "View/Edit Profile". 4. System displays current profile information. 5. Passenger updates information. 6. System validates new information. 7. System saves changes and confirms update. 8. Passenger views booking history with filter options. 9. Passenger can download past e-tickets or invoices.

UC5.

Title	Check in
Primary Actor	Passenger
Secondary Actors	Boarding pass
Success Scenario	1. Passenger accesses check-in page (24-48 hours before flight). 2. System verifies passenger eligibility for check-in. 3. Passenger enters booking reference and last name. 4. System retrieves booking details. 5. Passenger confirms passenger details. 6. Passenger confirms seat. 7. System assigns boarding pass number. 8. System generates digital boarding pass. 9. Passenger downloads/prints boarding pass or adds to mobile wallet. 10. System updates passenger status to "CHECKED_IN".
Preconditions	1. Booking exists and is confirmed. 2. Flight departure is within check-in window.

UC6.

Title	Manage system
Primary Actor	Airplane staff
Secondary Actors	Database
Success Scenario	1. Staff logs into admin portal. 2. System displays admin dashboard with modules: Flight Management, User Management, Reports, System Settings. 3. Staff selects "Flight Management". 4. System displays list of flights. 5. Staff adds new flight with details (schedule, prices, capacity). 6. System validates and saves flight data. 7. Staff updates existing flight (change schedule, prices, or status). 8. Staff views booking reports and analytics. 9. Staff manages user accounts and permissions.
Preconditions	1. Staff has admin privileges.

UC7.

Title	Cancel booking
Primary Actor	Passenger
Secondary Actors	Database
Success Scenario	1. Passenger accesses booking management. 2. Passenger selects booking to cancel. 3. System displays cancellation policy and refund amount. 4. Passenger confirms cancellation. 5. System updates booking status to "CANCELLED". 6. System releases seat back to available inventory. 7. System initiates refund process (if applicable). 8. System sends cancellation confirmation email.
Extensions	3a. Non-refundable ticket: System informs passenger no refund will be issued. 7a. Partial refund: System calculates and processes partial refund based on policy.
Preconditions	1. Booking exists and is not already cancelled. 2. Passenger is authorized to cancel the booking.

2. Entities and Their Attributes

Entity 1: Passenger

Description: Represents a person who books or travels on flights. Stores personal and contact information.

Attributes:

- id (INT, PK, Auto-increment) - Unique identifier for each passenger
- firstName (VARCHAR(50), NN) - Legal first name
- lastName (VARCHAR(50), NN) - Legal last name
- email (VARCHAR(100), UQ, NN) - Primary contact email
- phoneNumber (VARCHAR(20)) - Contact phone number
- passportNumber (VARCHAR(20), UQ) - Passport identification number
- dateOfBirth (DATE) - Birth date for age verification

Entity 2: Flight

Description: Represents a scheduled airplane journey between two locations. Contains flight details, schedule, capacity, and pricing information.

Attributes:

- id (INT, PK, Auto-increment) - Unique flight identifier
- airline (VARCHAR(50), NN) - Operating airline company
- departureAirportCode (VARCHAR(3), NN) - IATA code of departure airport
- arrivalAirportCode (VARCHAR(3), NN) - IATA code of arrival airport
- departureCity (VARCHAR(50), NN) - Departure city name
- arrivalCity (VARCHAR(50), NN) - Arrival city name
- scheduledDeparture (DATETIME, NN) - Planned departure date and time
- scheduledArrival (DATETIME, NN) - Planned arrival date and time
- totalSeats (INT, NN) - Total seating capacity
- availableSeats (INT, NN) - Currently available seats
- price (DECIMAL(10,2), NN) - Standard ticket price before taxes

- status (VARCHAR(20), DEFAULT 'SCHEDULED') - Current status (SCHEDULED, BOARDING, DEPARTED, ARRIVED, CANCELLED, DELAYED)
- gateNumber (VARCHAR(5)) - Boarding gate assignment

Entity 3: Booking

Description: Represents a reservation transaction for one or more passengers on a specific flight. Tracks the booking process from creation to completion.

Attributes:

- id (INT, PK, Auto-increment) - Unique booking identifier
- passengerId (INT, FK) - Passenger making the booking
- flightId (INT, FK) - Selected flight for booking
- date (DATETIME) - Date and time of booking creation
- totalPassengers (INT, NN) - Number of passengers in this booking
- totalAmount (DECIMAL(10,2), NN) - Total cost including all passengers and fees
- status (VARCHAR(20), DEFAULT 'PENDING') - Current status (PENDING, CONFIRMED, CANCELLED, CHECKED_IN, COMPLETED)

Entity 4: Ticket

Description: Represents an individual travel document for one passenger. Each passenger in a booking gets their own ticket with specific seat assignment.

Attributes:

- id (INT, PK Auto-increment) - Unique ticket identifier
- bookingId (INT, FK) - Associated booking reference
- passengerId (INT, FK) - Specific passenger for this ticket
- flightId (INT, FK) - Specific flight for travel
- seatNumber (VARCHAR(4)) - Assigned seat
- class (VARCHAR(20), NN) - Service class (ECONOMY, BUSINESS, FIRST)
- fare (DECIMAL(10,2), NN) - Individual ticket price

- status (VARCHAR(20), DEFAULT 'ISSUED') - Ticket state (ISSUED, CHECKED_IN, BOARDED, USED, VOID)
- boardingPassNumber (VARCHAR(10)) - Boarding pass identifier for check-in
- boardingTime (DATETIME) - Actual boarding time
- baggage (VARCHAR(20)) - Included baggage information

Entity 5: Payment

Description: Records financial transactions associated with bookings. Handles payment processing, tracking, and refund management.

Attributes:

- id (INT, PK, Auto-increment) - Unique payment identifier
- bookingId (INT, FK, UQ) - Associated booking (one payment per booking)
- method (VARCHAR(20), NN) - Payment type (CREDIT_CARD, DEBIT_CARD, PAYPAL, BANK_TRANSFER)
- amount (DECIMAL(10,2), NN) - Transaction amount
- date (DATETIME) - Date and time of payment
- transactionId (VARCHAR(50), UQ) - External payment gateway reference
- status (VARCHAR(20), DEFAULT 'COMPLETED') - Payment state (PENDING, COMPLETED, FAILED, REFUNDED, PARTIALLY_REFUNDED)
- refundAmount (DECIMAL(10,2)) - Amount refunded if applicable
- refundDate (DATETIME) - Date of refund if applicable

3. Relationships Between Entities

Relationship 1: Passenger ↔ Booking (1:N)

- **Type**: One-to-Many
- **Description**: One passenger can make multiple bookings, but each booking has exactly one primary passenger.
- **Foreign Key**: Booking.passengerId references Passenger.id

Relationship 2: Flight ↔ Booking (1:N)

- **Type**: One-to-Many
- **Description**: One flight can have many bookings, but each booking is for exactly one flight.
- **Foreign Key**: Booking.flightId references Flight.id

Relationship 3: Booking ↔ Ticket (1:N)

- **Type**: One-to-Many
- **Description**: One booking can contain multiple tickets (for multiple passengers), but each ticket belongs to exactly one booking.
- **Foreign Key:** Ticket.bookingId references Booking.id

Relationship 4: Passenger ↔ Ticket (1:N)

- **Type**: One-to-Many
- **Description**: One passenger can have multiple tickets (for different flights/ dates), but each ticket is assigned to exactly one passenger.
- **Foreign Key**: Ticket.passengerId references Passenger.id

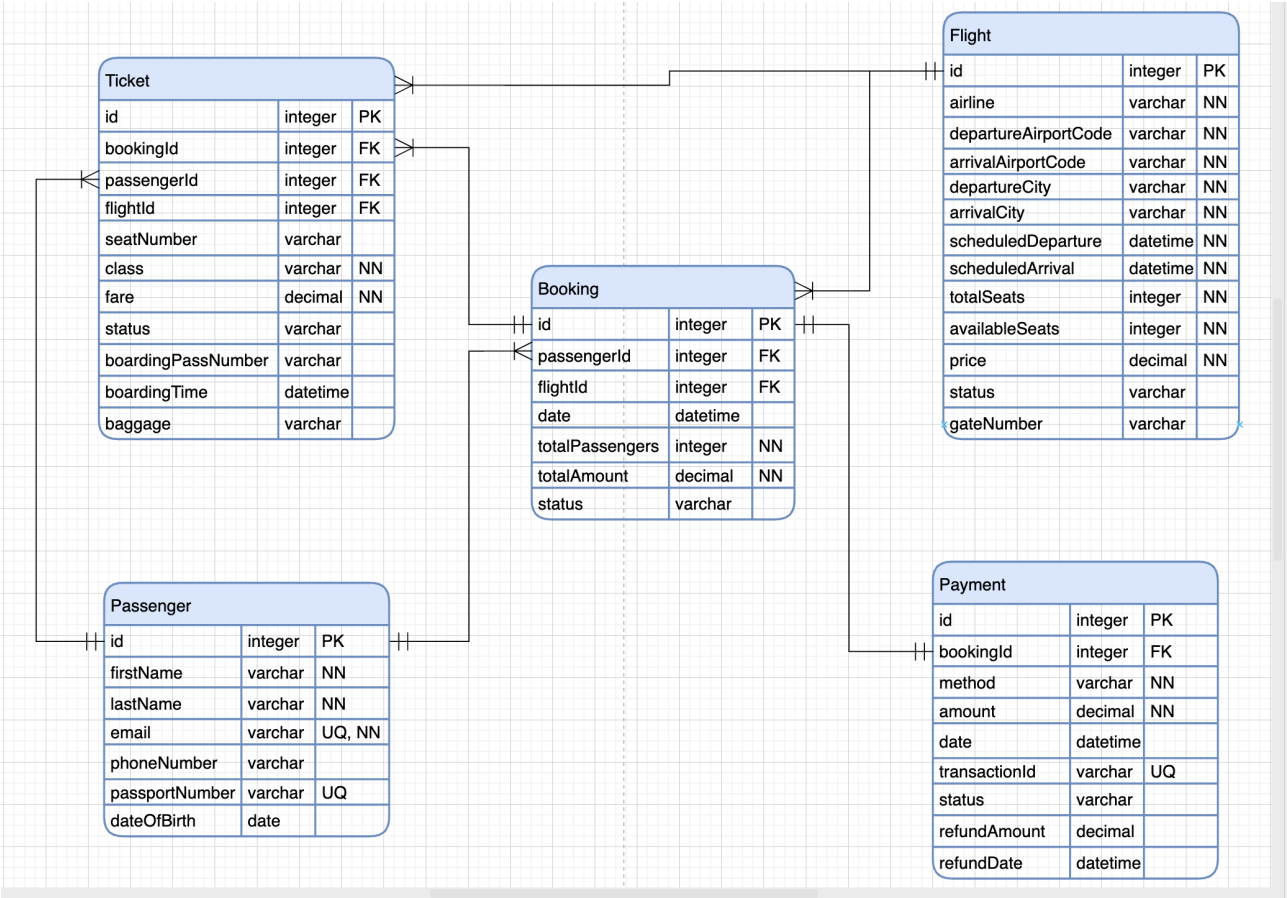
Relationship 5: Flight ↔ Ticket (1:N)

- **Type**: One-to-Many
- **Description**: One flight can have many tickets, but each ticket is for exactly one flight.
- **Foreign Key**: Ticket.flightId references Flight.id

Relationship 6: Booking ↔ Payment (1:1)

- **Type**: One-to-One
- **Description**: Each booking has exactly one payment transaction, and each payment is for exactly one booking.
- **Foreign Key**: Payment.bookingId references Booking.id

4. ER diagram


Ticket
id integer PK
bookingId integer FK
passengerId integer FK
flightId integer FK
seatNumber varchar
class varchar NN
fare decimal NN
status varchar
boardingPassNumber varchar
boardingTime datetime
baggage varchar
Flight
id integer PK
airline varchar NN
departureAirportCode varchar NN
arrivalAirportCode varchar NN
departureCity varchar NN
arrivalCity varchar NN
scheduledDeparture datetime NN
scheduledArrival datetime NN
totalSeats integer NN
availableSeats integer NN
price decimal NN
status varchar
gateNumber varchar
Booking
id integer PK
passengerId integer FK
flightId integer FK
date datetime
totalPassengers integer NN
totalAmount decimal NN
status varchar
Passenger
id integer PK
firstName varchar NN
lastName varchar NN
email varchar UQ, NN
phoneNumber varchar
passportNumber varchar UQ
dateOfBirth date
Payment
id integer PK
bookingId integer FK
method varchar NN
amount decimal NN
date datetime
transactionId varchar UQ
status varchar
refundAmount decimal
refundDate datetime