Airline Reservation and Management System

1. Introduction

SkyHigh Airlines anticipates significant growth in air travel and aims to implement a **robust Airline Reservation and Management System (ARMS)**. This system is designed to manage flight schedules, handle passenger information, optimize seat allocations, and offer loyalty rewards to frequent travelers. Additionally, the system provides insights into operational efficiency by monitoring on-time performance and booking consistency.

The project aims to enhance **passenger experience**, ensure **real-time seat allocation**, and facilitate **remote access** for regional airports.

2. Objectives

1. **Flight Management:** Store flight details, including flight numbers, schedules, origin, destination, and available seats across Economy, Business, and First classes.
2. **Passenger Management:** Maintain passenger profiles, personal information, seat preferences, class preferences, and loyalty points.
3. **Booking Management:** Handle reservations with unique booking IDs, seat assignments, and booking status (Confirmed, Cancelled, Pending).
4. **Seat Allocation:** Dynamically update seat status and assignments, reflecting real-time availability.
5. **Loyalty Program:** Track miles, points, and reward tiers for frequent flyers.
6. **Operational Efficiency:** Monitor on-time performance and ensure consistent data handling during concurrent transactions.
7. **Remote Access:** Support centralized server management with remote client access via FEDERATED tables.

3. Technology Stack

Component	Technology/Tool Used
Database Management System	MySQL
Front-End UI	Streamlit (Python)

Component	Technology/Tool Used
Server-Client Connectivity	MySQL FEDERATED tables for remote access
Programming Language	Python (for Streamlit + triggers)
SQL Features Used	Stored Procedures, Triggers, User-Defined Functions, Recursive Queries, Transaction Isolation Levels

---

4. System Design

4.1 E-R Diagram

The E-R Diagram captures the core entities and relationships:

Entities:

- **Flight:** Stores flight schedule, seat counts, on-time rating.
- **Passenger:** Stores personal info, seat preference, loyalty account.
- **Booking:** Links passengers to flights and tracks seat assignment and booking status.
- **Seat:** Maintains seat status for each flight.
- **LoyaltyProgram:** Tracks points, miles, and tier information.

Relationships:

- Passenger → Booking: One-to-Many
- Flight → Booking: One-to-Many
- Flight → Seat: One-to-Many
- Passenger → LoyaltyProgram: One-to-One

Crow's foot notation was used to depict cardinalities.

4.2 Relational Schema

Table	Columns	Constraints
Flight	flight_id (PK), flight_number, departure_time, arrival_time, origin, destination, on_time_rating, economy_seats, business_seats, first_seats	PK, CHECK on seats and rating
Passenger	passenger_id (PK), first_name, last_name, email (UNIQUE), phone, seat_preference, class_preference, loyalty_id (FK)	FK → LoyaltyProgram.loyalty_id

Table	Columns	Constraints
Booking	booking_id (PK), passenger_id (FK), flight_id (FK), seat_number, booking_status, booking_date, class	FK → Passenger & Flight, ENUM status
Seat	seat_id (PK), flight_id (FK), seat_number, class, status	FK → Flight, ENUM status
LoyaltyProgram	loyalty_id (PK), passenger_id (FK UNIQUE), miles_accumulated, points_available, tier	FK → Passenger, points/miles >= 0

---

5. Implementation

5.1 Database Creation & Population

- **Tables Created:** Flight, Passenger, Booking, Seat, LoyaltyProgram.
- **Data Population:** Each table was populated with at least 5 records.
- **Constraints Implemented:** Primary keys, foreign keys, check constraints, unique constraints, and ENUM types.

5.2 Booking System (Python + Streamlit)

- Users can **make bookings**, selecting flight, seat, class, and booking status.
- Booking triggers update loyalty points automatically via a **MySQL trigger**.
- Real-time seat availability is displayed through queries on the Seat table.
- Existing bookings can be viewed using the Streamlit table interface.

5.3 Remote Access

- **FEDERATED tables** enabled remote client at regional airports to access central server data.
- Example: Booking deletion from the remote client updates the central database seamlessly.

5.4 Stored Procedure – CancelBooking

- Input: `booking_id`
- Actions:
    1. Remove seat assignment from the Booking table.
    2. Update the Seat table to mark the seat as `Available`.
- Ensures easy management of canceled bookings and real-time seat updates.

5.5 Trigger – Loyalty Points Update

- After each new booking, **10 loyalty points** are automatically added to the passenger's LoyaltyProgram record.
- Ensures points are always up-to-date without manual intervention.

5.6 User-Defined Function – Mileage-Based Upgrade

- Function: `CheckUpgrade(miles INT) RETURNS BOOLEAN`
- Returns `TRUE` if passenger has ≥ 10,000 miles; `FALSE` otherwise.
- Enables the system to quickly determine eligible passengers for seat upgrades.

5.7 Concurrency Control

- Implemented **SERIALIZABLE isolation level** to prevent booking conflicts when multiple users attempt to book the same seat concurrently.
- Ensures data consistency and prevents overbooking.

5.8 Recursive Query – Employee Hierarchy

- Demonstrates advanced SQL capabilities by finding all employees reporting (directly/indirectly) to a manager using **CTE with recursion**.

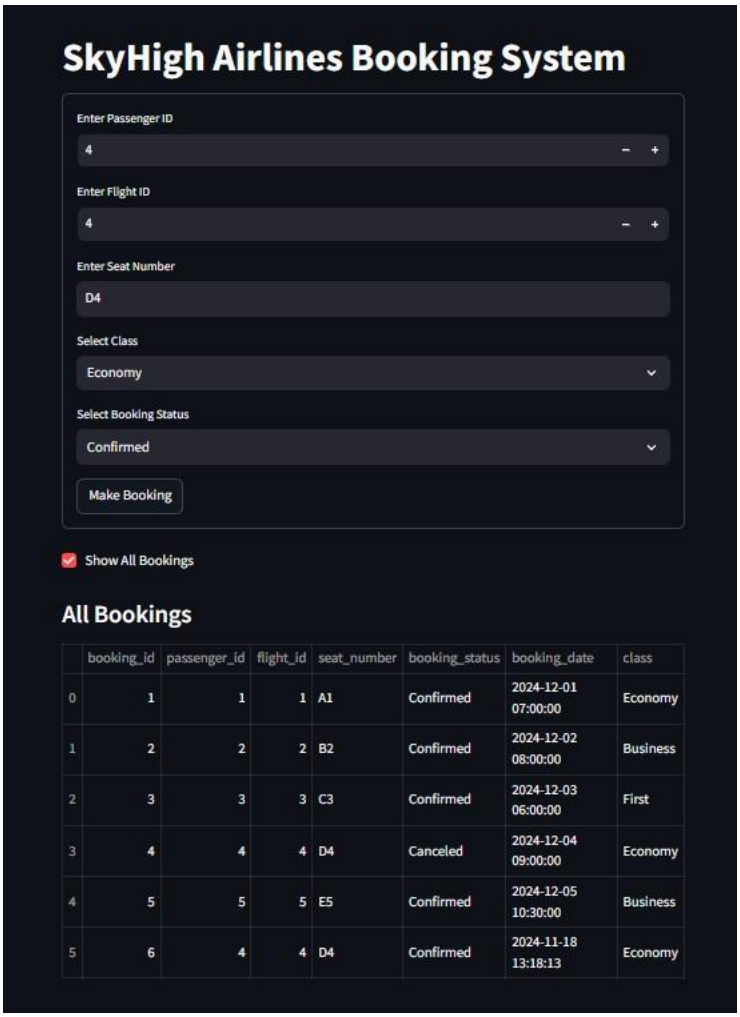
---

6. Functional Features

Feature	Description
Flight Scheduling	Add, modify, and view flight details
Passenger Management	Maintain profiles, preferences, and loyalty info
Seat Allocation	Dynamic seat assignment, preference handling, availability updates
Booking Management	Create, cancel, and view bookings
Loyalty Program Management	Automatic points update, mileage tracking, tier management
Remote Access	Federated database access for regional offices
Cancellation Handling	Stored procedure to mark canceled bookings and update seat availability
Upgrade Eligibility	Function to check if a passenger qualifies for mileage-based upgrades
Concurrency Control	Serializable transactions to avoid double-booking

---

7. Screenshot



---

8. Conclusion

The Airline Reservation and Management System provides a **comprehensive, real-time solution** for managing flights, passengers, bookings, seats, and loyalty programs. The system is designed for scalability, remote access, and operational efficiency, incorporating advanced SQL features, Python-based front-end, and robust database design.

The project successfully demonstrates:

- Real-time seat allocation
- Loyalty program automation
- Efficient cancellation handling
- Safe concurrent transaction management

This system can be **further enhanced** with features like payment gateway integration, advanced analytics for flight performance, and predictive seat assignment algorithms.

9. References

1. MySQL Documentation – https://dev.mysql.com/doc/
2. Streamlit Documentation – https://docs.streamlit.io/
3. “Database System Concepts” – Silberschatz, Korth, Sudarshan