Gisma University of Applied Sciences

Gisma University of Applied Sciences

Department of Computer and Data Sciences

Name - Aniket Sonu

Student Id - GH1048274

Project Name - Airline Reservation Database Management System

Course Name - M605 Advanced Databases

Professor - Dr. Mazhar Hameed

Project URLs:

GitHub Link - AniketSonu/AirlineReservation_Database_Project
Video Link – DataBase Project Video.mp4

Submission Date – 16-Dec-2025

Table Of Content

Introduction:

The Airline Reservation Database Management System deals with the three collections like Passengers Details, Airlines Details and Booking Details. In these collections, we are managing the relation between

passengers, airlines and booking and performing business operation like Insertion Data, Deleting Data and many more. I have Choose the **airlines and travel industry.** In this we store many data such as passenger details, flight schedules, seat availability, and booking. This system demonstrates how modern database technologies can be used together to build scalable and reliable reservation platforms.

System Design:

I have designed this database in NoSQL using MongoDB Compass because of its flexible nature to store data. In this database, three collections are created Named:

1. Passenger – In this collection, I am storing passenger data like name, address, phone no with Email_ID as Primary Key.

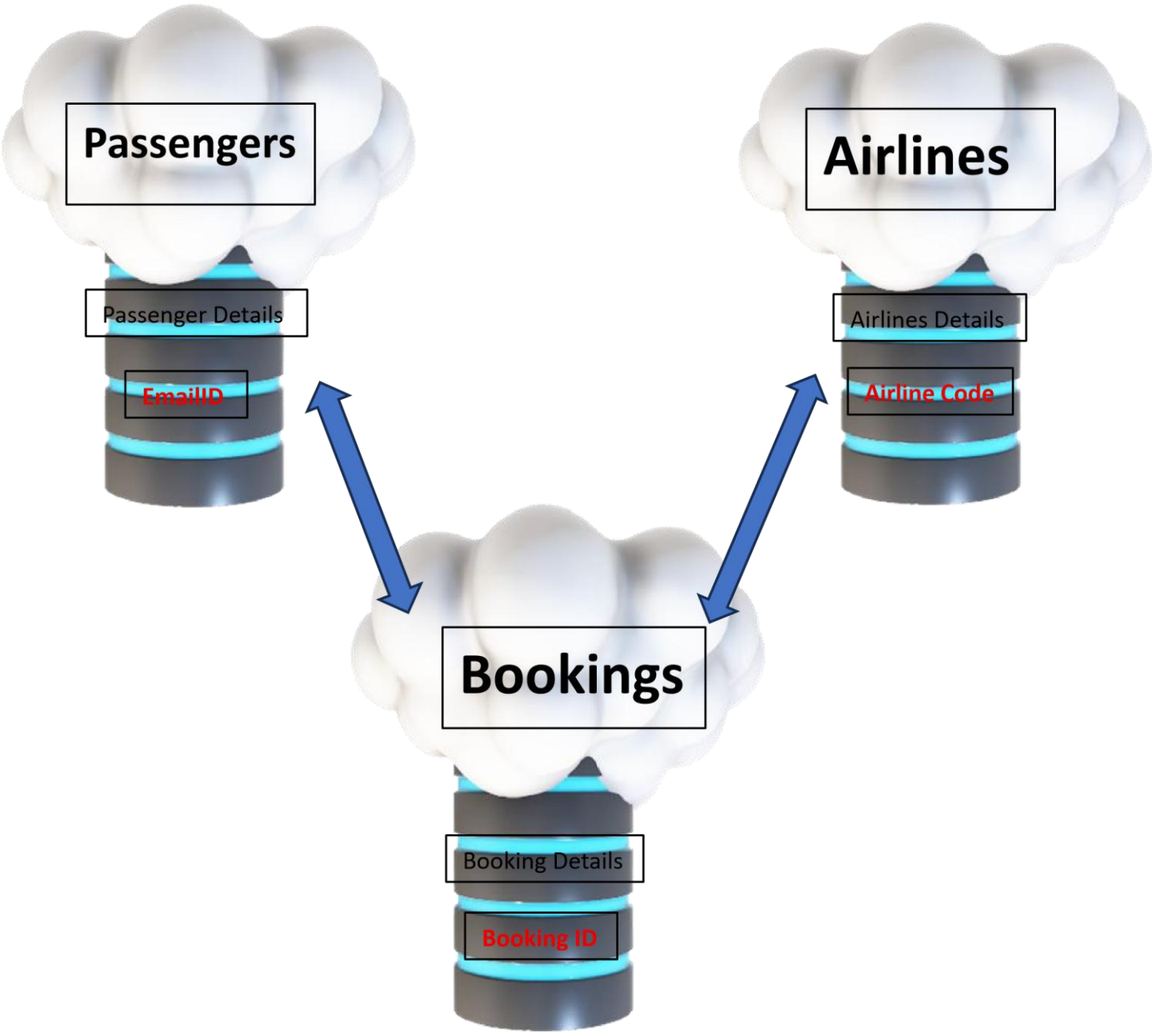
2. Airlines – In this Collection, I am adding Airlines name, Airline code, Airline number, Source and destination city, and Timings. With Airline code as Primary Key.

3. Bookings – In this collection, I am storing Booking details with the passengers and airlines Ids as referencing collection to perform all CRUD operations through booking Collection.

Database Schema Design:

As shown in below diagram, Passengers and Airlines are two independent collections but both relate to Bookings Collections with passengers Email_id and Airline_code as foreign Keys.

Collections Schema Diagram



Implementation:

Database creation – In MongoDB we can easily create a database on the homepage + button, but in this project, I have created this database using Java Spring Boot Program which is also Integrated with this database to perform operation like I am doing through VS code Playgrounds.

Steps: -

- Insertion – I am adding data in Database collection through both modes.

  insertOne()

  insertMany()

- Read – The command to read data is finding (). This gives us all the data in database. We can also fetch data by the help of parameters passes in the function.

- Update: Update is using to update all data fields in the documents using updateOne() and updateMany(). This will change one entry and many entries respectively.

- Delete: Like the Update, we are using deleteOne() and DeleteMany data from Document.

- I have created a playground for Group, Match, Project queries on Database. For Example - bookings.aggregate([{$match: {BookingStatus:"Confirmed”} }]) is showing results which have booking status “Confirmed” only.

- Also, I have created Lookup queries to perform Joins in NoSQL. In Database, booking Collection are stored with both Airlines and passenger IDs as Foreign Key.

  By using lookup, it is showing me data of booking Data, Passenger data and airline data combined.

  I am also performing Indexing search in playground to check how many document are getting scanned in process through Indexing and CollScan.

Please refer to:

There are 4 separate folders on GitHub repository consisting all playground to perform queries on Database. Please watch the video for Execution results.

Results –

The Airline Reservation Database Management system giving results as expected as I am performing queries like retrieving passengers and airlines whole details with Booking details with the help of manual transaction logic. All operation is working fine giving result as expected.

Please refer to the video for the working output and examples.

Challenges and Solutions:

1. First Challenge was occurred when we are using transaction entries on MongoDB compass as it is not supporting.

Solution: Mongosh should be installed or MongoDb Atlas should be downloaded to perform transaction

2. MongoDB does not have strong relational constraints.

Solution: Application-level validations were implemented to ensure data accuracy.

Conclusion and Future Work:

The Project successfully proves the use of **NoSQL (MongoDB)** in building Airline Reservation Database Management System. This project is designed to handling large and dynamic datasets commonly found in airline applications.

Future work will add more feature like front end code, real world application, and transaction with real time database so we can build a full working mobile Application or Web application to make it more user friendly.

Thanks For Your Attention!