AVIATION MANAGEMENT SYSTEM

A PROJECT REPORT
for
Mini Project (KCA353)
Session (2024-25)

Submitted by

Nikhil Nigam
2300290140108
Prashant Kumar
2300290140122
Neha Sharma
2300290140106

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Ms. Divya Singhal
Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

DECEMBER 2024

CERTIFICATE

Certified that Nikhil Nigam 2300290140108, Prashant Kumar 2300290140122 and Neha Sharma 2300290140106 and have carried out the project work titled "Aviation Management System" (Mini- Project-KCA353) for Master of Computer Application from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the students themselves. The contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Ms. Divya Singhal	**Dr. Arun Tripathi**
Assistant Professor	**Head**
Department of Computer Applications	**Department of Computer Applications**
KIET Group of Institutions, Ghaziabad	**KIET Group of Institutions, Ghaziabad**

AVIATION MANAGEMENT SYSTEM

Nikhil Nigam, Neha Sharma, Prashant Kumar

ABSTRACT

The Aviation Management System (AMS) is a comprehensive and innovative software solution designed to streamline and optimize the diverse operations of the aviation industry. AMS integrates key functionalities such as flight scheduling, passenger services, crew management, and maintenance tracking. By automating these critical processes, the system aims to enhance operational efficiency, ensure regulatory compliance, and improve the overall passenger experience.

The Airline Management System (AMS) offers several key features designed to streamline operations and improve efficiency. It provides centralized data management for real-time updates, ensuring seamless coordination across all departments. Advanced automation capabilities enable efficient flight scheduling, crew assignments, and maintenance tracking. The system also enhances safety and ensures strict regulatory compliance monitoring. Additionally, its scalable architecture allows for the seamless integration of future technologies, making it adaptable to evolving industry needs.

AMS redefines the standards of efficiency and reliability in the aviation sector by addressing operational challenges through cutting-edge technology.

ACKNOWLEDGEMENTS

We extend our heartfelt gratitude to our project supervisor, **Ms. Divya Singhal**, for her invaluable guidance and support throughout the project. Her constructive insights and constant encouragement have been pivotal to the successful completion of this work.

We are deeply thankful to **Dr. Arun Kumar Tripathi, Head of the Department of Computer Applications**, for his administrative help and insightful comments.

We are also grateful to our families, peers, and friends, whose unwavering support and understanding played a crucial role in the completion of this project.

Nikhil Nigam
Prashant Kumar
Neha Sharma

TABLE OF CONTENTS

Page.no

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Overview

The **Aviation Management System (AMS)** is a multi-functional software solution developed to manage the diverse and interconnected operations of the aviation industry. From **flight scheduling** and **crew management** to **passenger services** and **aircraft maintenance**, AMS offers a centralized platform to automate and optimize operations.

The aviation industry operates in a dynamic environment where the efficient management of resources and operations is critical. Traditional manual systems are no longer sufficient to handle the increasing volume of flights, passengers, and operational data. The **Aviation Management System** addresses these challenges by integrating multiple functionalities into a unified platform, enabling airlines, airports, and ground services to collaborate seamlessly.

By leveraging **modern technologies** such as Java and MySQL, the AMS provides real-time updates, automates repetitive tasks, and ensures compliance with safety and regulatory standards. The system is designed with scalability in mind, allowing it to adapt to the growing needs of airlines and the evolving requirements of the aviation industry.

1.2 Objectives

The primary objectives of the **Aviation Management System** are:

1. **Enhance Operational Efficiency**
   a. Streamline processes such as flight scheduling, passenger check-ins, crew assignments, and maintenance planning.
   b. Reduce delays and improve the overall punctuality of operations.
2. **Improve Passenger and Staff Experience**
   a. Provide a user-friendly platform for passengers to book tickets, check flight statuses, and manage their itineraries.

    b. Enable staff to access critical information in real-time, facilitating faster decision-making.

3. **Ensure Safety and Regulatory Compliance**
    a. Maintain detailed records of aircraft maintenance schedules, crew certifications, and flight logs.
    b. Ensure adherence to aviation safety standards and government regulations.
4. **Optimize Resource Utilization**
    a. Effectively allocate crew, aircraft, and maintenance resources to minimize costs and maximize efficiency.
5. **Enable Data-Driven Decision Making**
    a. Use real-time analytics to gain insights into customer preferences, operational efficiency, and resource utilization.

1.3 Problem Statement

The aviation industry is a highly complex ecosystem where numerous operations must be managed simultaneously. However, many airlines and airports still rely on outdated manual processes and fragmented systems, leading to inefficiencies and higher operational costs. Key challenges include:

1. **Inefficient Flight Scheduling**
    a. Existing systems often fail to account for real-time changes such as weather delays or maintenance issues, resulting in disrupted schedules.
2. **Crew Management Challenges**
    a. Assigning crew members to flights manually is time-consuming and prone to errors. Ensuring that crew certifications are up-to-date is also challenging without proper tracking.
3. **Lack of Integration Between Modules**
    a. Passenger booking systems, maintenance logs, and flight operations often operate as separate entities, leading to data duplication and inconsistencies.
4. **Safety Risks Due to Poor Maintenance Tracking**
    a. Inadequate tracking of aircraft maintenance schedules can lead to safety issues, regulatory violations, and increased risks of accidents.
5. **Customer Dissatisfaction**

   a. Delayed flights, poor communication, and limited access to real-time updates result in a negative passenger experience.

1.4 Proposed Solution

The **Aviation Management System** offers an innovative solution to these challenges by providing a **centralized platform** that integrates all major operations of the aviation industry. Key features of the AMS include:

1. **Centralized Data Management**
   a. A single database stores all operational data, eliminating redundancy and ensuring consistency.
2. **Automation of Processes**
   a. Automates repetitive tasks such as flight scheduling, crew assignments, and maintenance tracking, reducing human errors and saving time.
3. **Real-Time Data Access**
   a. Provides real-time updates on flight statuses, passenger bookings, and crew availability, enabling quick and informed decision-making.
4. **User-Friendly Interface**
   a. Offers intuitive dashboards for passengers, staff, and administrators, ensuring ease of use and accessibility.
5. **Scalability and Flexibility**
   a. Designed to scale as the needs of airlines grow, with the ability to integrate future technologies such as artificial intelligence for predictive analytics.
6. **Enhanced Safety Compliance**
   a. Tracks and logs all maintenance activities, ensuring that aircraft are inspected and repaired on time, in line with aviation safety standards.

1.5 Conceptual Overview of AMS

To illustrate the high-level design of the system, a **conceptual overview diagram** is included in figure 1.1, showing the interaction between the core modules and the central database.

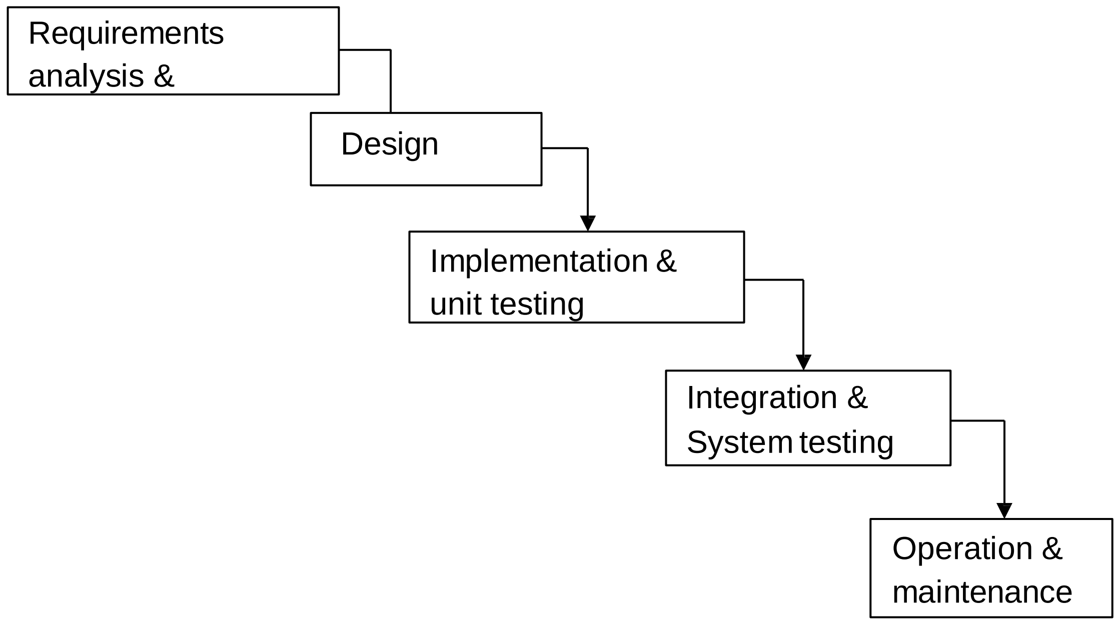


Fig 1.1: Conceptual Overview of Aviation Management System

- **Modules**:
  - **Flight Management**: Schedules flights, tracks delays, and updates statuses in real-time.
  - **Passenger Management**: Handles ticket bookings, check-ins, and baggage information.
  - **Crew Management**: Assigns crew, monitors their availability, and ensures compliance with training and certifications.
  - **Maintenance Management**: Tracks maintenance schedules, logs repair activities, and ensures regulatory compliance.
- **Central Database**: Acts as the backbone of the system, storing and synchronizing data from all modules

Chapter 2

Feasibility Study

2.1 Technical Feasibility

The **Aviation Management System (AMS)** leverages robust, scalable, and widely accepted technologies that ensure its successful implementation and long-term viability.

1. **Frontend Technology**
    a. **JavaFX**: Provides a rich user interface for the system, enabling intuitive navigation for passengers, staff, and administrators.
    b. **Benefits**: Cross-platform compatibility, modern GUI capabilities, and integration with backend services.
2. **Backend Technology**
    a. **Java Servlets**: Handles business logic, including flight scheduling, passenger management, and data processing.
    b. **JDBC**: Manages communication between the backend and the database, ensuring secure and efficient data handling.
3. **Database**
    a. **MySQL**: A relational database system that supports complex queries and large datasets. It stores critical data like passenger details, flight schedules, crew assignments, and maintenance logs.
    b. **Key Benefits**: Scalability, security, and high performance.
4. **APIs and External Integration**
    a. APIs allow for integration with external systems like payment gateways, weather monitoring, and global flight tracking services.

These technologies are proven to handle the demands of enterprise-level applications, ensuring the AMS is technically feasible and future-proof.

2.2 Financial Feasibility

The development and implementation of AMS are cost-effective due to the use of **open-source technologies** and efficient resource allocation.

1. **Development Costs**
    a. Salaries for developers, testers, and project managers.
    b. Hardware procurement: Servers and workstations.
    c. Software licenses for development tools.
2. **Operational Costs**
    a. **Server Hosting**: For deploying the AMS and managing the database.
    b. **Maintenance and Updates**: Routine maintenance to keep the system updated and secure.
3. **Cost-Benefit Analysis**
    a. **Cost Savings**: Automating flight scheduling and crew management reduces human errors and associated costs.
    b. **Revenue Growth**: Improved passenger experience leads to higher customer retention and revenue.

The estimated costs are justified by the long-term operational savings and revenue growth enabled by AMS.

2.3 Operational Feasibility

The system is designed to be user-friendly and seamlessly integrate with existing operations.

1. **Ease of Adoption**
    a. A modular architecture allows phased implementation, enabling airlines to adopt the system gradually without disrupting operations.
    b. User manuals and training sessions reduce the learning curve for staff.
2. **Scalability**
    a. AMS can accommodate growing passenger volumes and expanding operations as airlines add more routes and flights.
3. **Risk Management**
    a. Regular backups and data encryption ensure business continuity and protect sensitive data from breaches or losses.

AMS is operationally feasible, offering a smooth transition for all stakeholders, from staff to passengers.

Chapter 3

System Requirements

3.1 Hardware Requirements

The hardware requirements are designed to support the efficient functioning of the AMS across all modules.

1. **Server Specifications**
    a. **Processor**: Intel Core i5 or higher.
    b. **RAM**: Minimum 16GB for handling concurrent users and processes.
    c. **Storage**: 512GB SSD for quick data access and retrieval.
    d. **Network**: High-speed internet connection for real-time data synchronization.
2. **Workstation Requirements**
    a. **Processor**: Intel Core i3 or equivalent.
    b. **RAM**: 8GB to support the user interface and processing needs.
    c. **Display**: 1080p monitor for clear visualization of dashboards and data.
3. **Peripheral Devices**
    a. Printers for boarding passes and reports.
    b. Barcode scanners for passenger check-in and baggage handling.

3.2 Software Requirements

The software stack is selected for its reliability, scalability, and ease of integration with other systems.

1. **Operating System**
    a. Compatible with Windows, Linux, or macOS, providing flexibility for deployment.
2. **Frontend Development**
    a. **Technology**: JavaFX
    b. **Purpose**: To create an intuitive and visually appealing user interface for administrators, staff, and passengers.
3. **Backend Development**

    a. **Technology**: Java Servlets
    b. **Purpose**: Manages business logic, processes user requests, and interacts with the database.
4. **Database**
    a. **Technology**: MySQL 8.0
    b. **Purpose**: To store and manage data, ensuring quick retrieval and secure storage of sensitive information.
5. **Tools and Frameworks**
    a. **Eclipse IDE**: For coding and debugging.
    b. **MySQL Workbench**: For database schema design and management.

3.3 Security and Backup

Security and data integrity are critical for the AMS, given the sensitivity of passenger information and operational data.

1. **Security Measures**
    a. **Encryption**: All data transactions are encrypted using SSL/TLS protocols to prevent unauthorized access.
    b. **Authentication**: Role-based access control (RBAC) ensures that only authorized personnel can access specific modules.
2. **Backup and Recovery**
    a. **Backup Frequency**: Daily backups of the database and application data are stored in both local and cloud servers.
    b. **Recovery Plan**: Regular testing of backup data ensures that the system can be restored quickly in the event of hardware failure or cyberattacks.

3.4 Performance Requirements

AMS is designed to handle high-performance demands to ensure a smooth experience for users.

1. **Concurrent Users**

   a. Supports 100+ concurrent users, including passengers, staff, and administrators.

2. **Response Time**
   a. Average response time for booking a flight or updating flight status should be under 2 seconds.
3. **Downtime**
   a. Scheduled downtime for maintenance should not exceed 2 hours per month.

3.5 Integration Requirements

AMS must integrate with existing and external systems to enhance its functionality.

1. **Payment Gateways**
   a. Integration with gateways like PayPal, Stripe, or Razorpay for secure online transactions.
2. **Flight Tracking Systems**
   a. Integration with real-time flight tracking APIs to update passengers and staff on flight statuses.
3. **Government Portals**
   a. Systems for regulatory compliance reporting and passenger data sharing with government authorities (e.g., customs, immigration).

Chapter 4

System Design

4.1 System Architecture

The **Aviation Management System (AMS)** employs a **three-tier architecture** to ensure modularity, scalability, and maintainability.

1. **Presentation Layer**
    a. Includes the **Graphical User Interface (GUI)** developed using JavaFX.
    b. Accessible by three types of users:
        i. **Administrators**: Manage flights, crew, and maintenance.
        ii. **Staff**: Assist passengers and update flight statuses.
        iii. **Passengers**: Book tickets, check-in, and view itineraries.
2. **Business Logic Layer**
    a. Handles core functionalities such as:
        i. **Flight Scheduling**: Automatically assigns aircraft and crew.
        ii. **Maintenance Management**: Tracks maintenance schedules and logs.
    b. Developed using **Java Servlets**, ensuring smooth communication between the UI and the database.
3. **Database Layer**
    a. Centralized storage using **MySQL**, housing data related to passengers, flights, crew, and maintenance logs.
    b. Ensures data consistency and integrity across all modules.

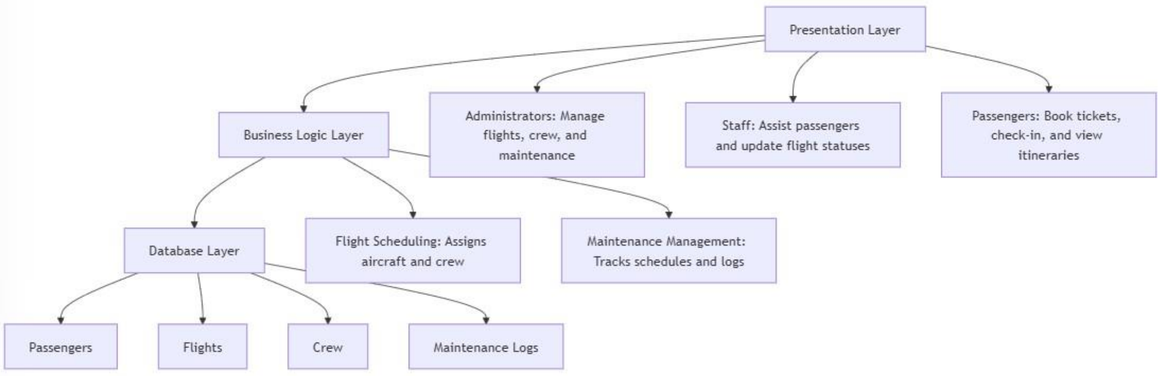

Presentation Layer
Business Logic Layer
Administrators: Manage flights, crew, and maintenance
Staff: Assist passengers and update flight statuses
Passengers: Book tickets, check-in, and view itineraries
Database Layer
Flight Scheduling: Assigns aircraft and crew
Maintenance Management: Tracks schedules and logs
Passengers
Flights
Crew
Maintenance Logs

Figure 4.1: System Architecture Diagram

4.2 Entity-Relationship Diagram (ERD)

The **ERD** highlights the relationships between various entities in the system:

1. **Entities and Attributes**:
    a. **Flights**: Flight number, departure and arrival times, status.
    b. **Passengers**: Name, contact information, booking reference.
    c. **Crew**: ID, certifications, assigned flights.
    d. **Maintenance Logs**: Aircraft ID, issue details, repair status.
2. **Relationships**:
    a. A **Flight** is linked to multiple **Passengers** through bookings.
    b. Each **Crew Member** is assigned to multiple flights.
    c. **Maintenance Logs** are associated with specific aircraft

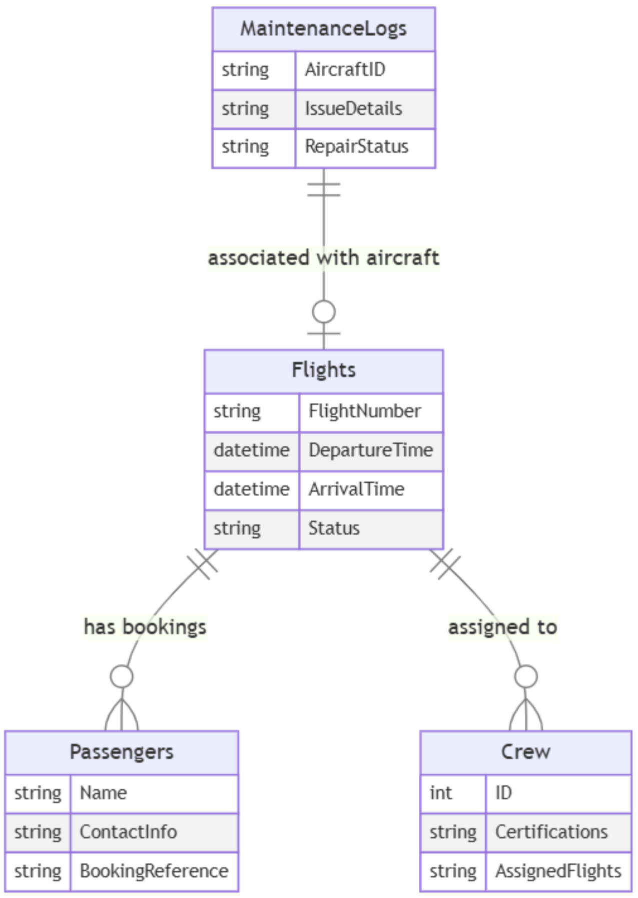


Figure 4.2: ER Diagram

4.3 Use Case Diagram

The **Use Case Diagram** illustrates how different users interact with the AMS.

1. **Actors**:

    a. **Administrator**: Manages flight schedules, assigns crew, and oversees maintenance.
    b. **Staff**: Manages passenger check-ins and flight updates.
    c. **Passenger**: Books tickets, checks flight status, and manages bookings.

2. **Use Cases**:
    a. **Flight Scheduling**: Admin schedules flights and assigns resources.
    b. **Passenger Check-In**: Staff updates passenger details and prints boarding passes.
    c. **Ticket Booking**: Passengers search for and book available flights.

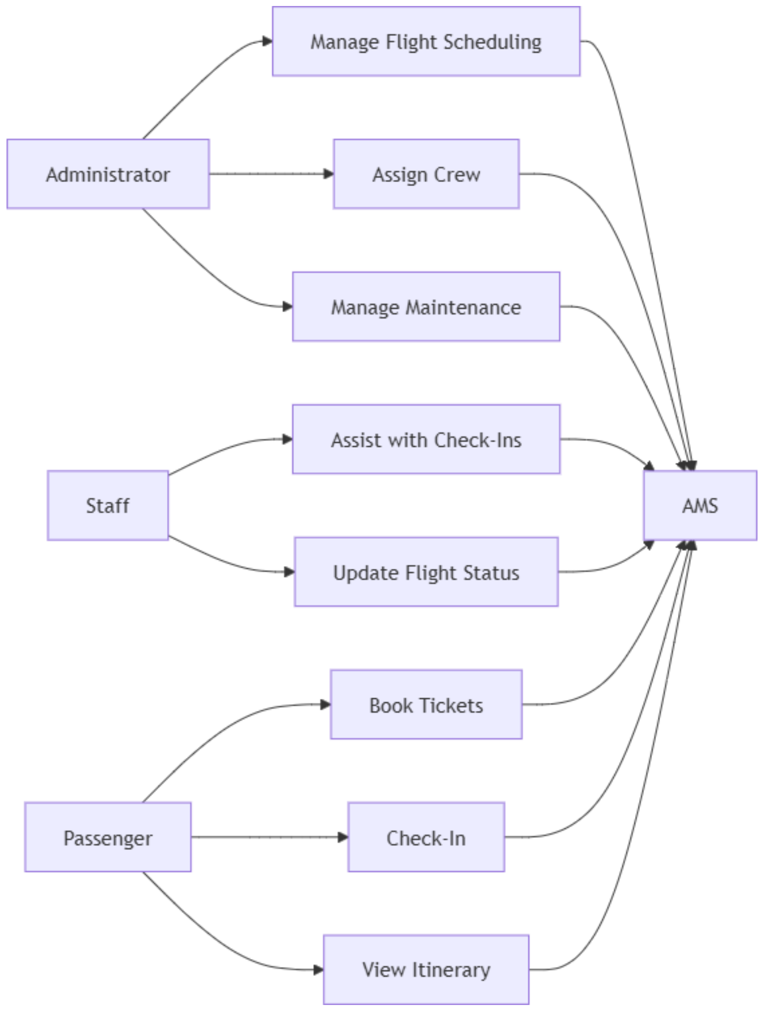


Figure 4.3: Use Case Diagram

4.4 Data Flow Diagram (DFD)

The Data Flow Diagram (DFD) provides a visual representation of how data flows within the AMS.

1. **Level 0 DFD**:
    a. Represents the entire system as a single process.
    b. Shows data inputs (e.g., passenger information) and outputs (e.g., flight schedules).

Level 0 DFD

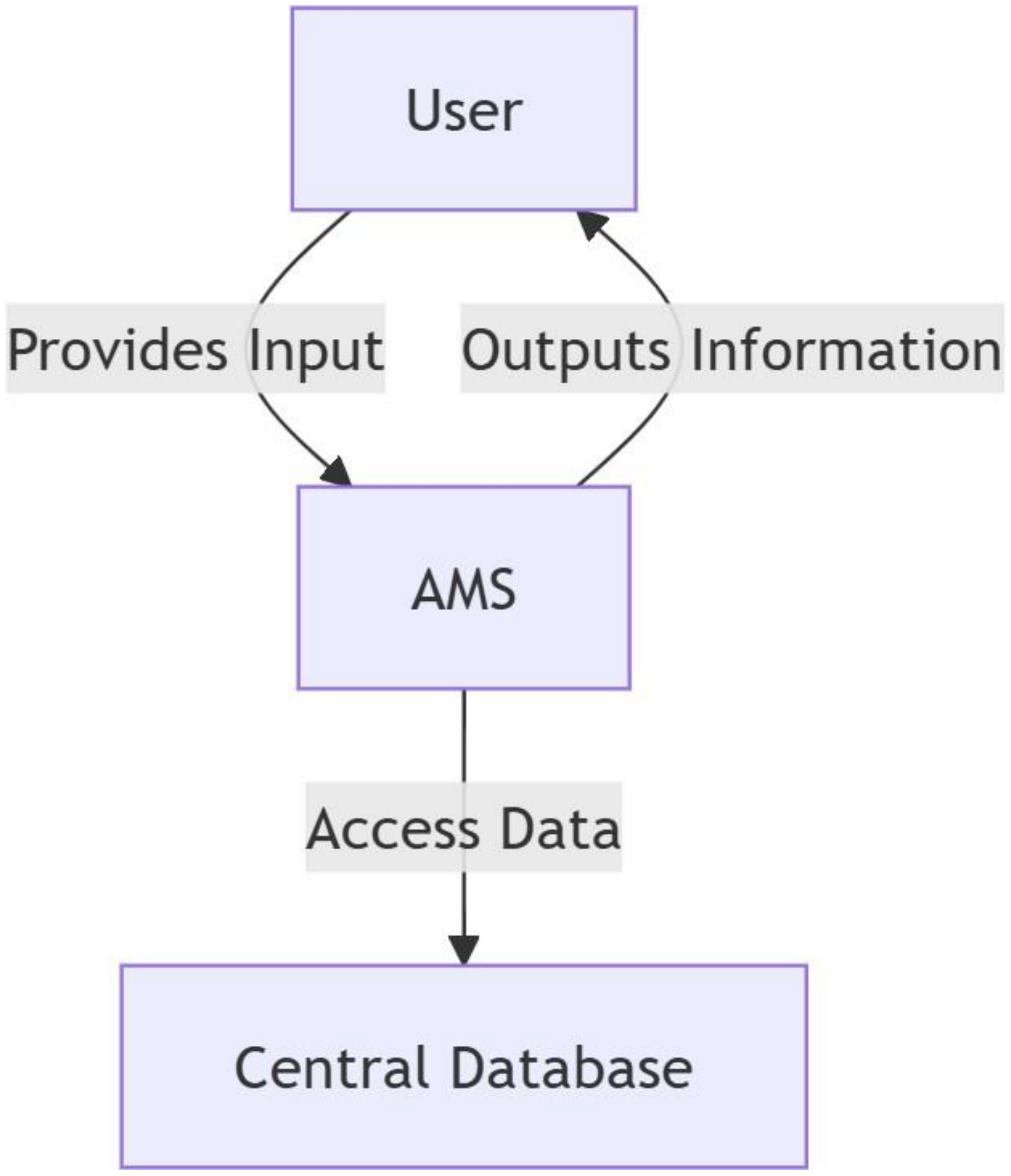


Figure 4.4: Level 0 DFD

2. **Level 1 DFD**:
    a. Breaks down the main process into sub-processes:
        i. **Booking Management**: Passenger inputs are processed for ticket generation.
        ii. **Flight Updates**: Admin updates flight statuses, which are displayed to passengers.

1st Level DFD

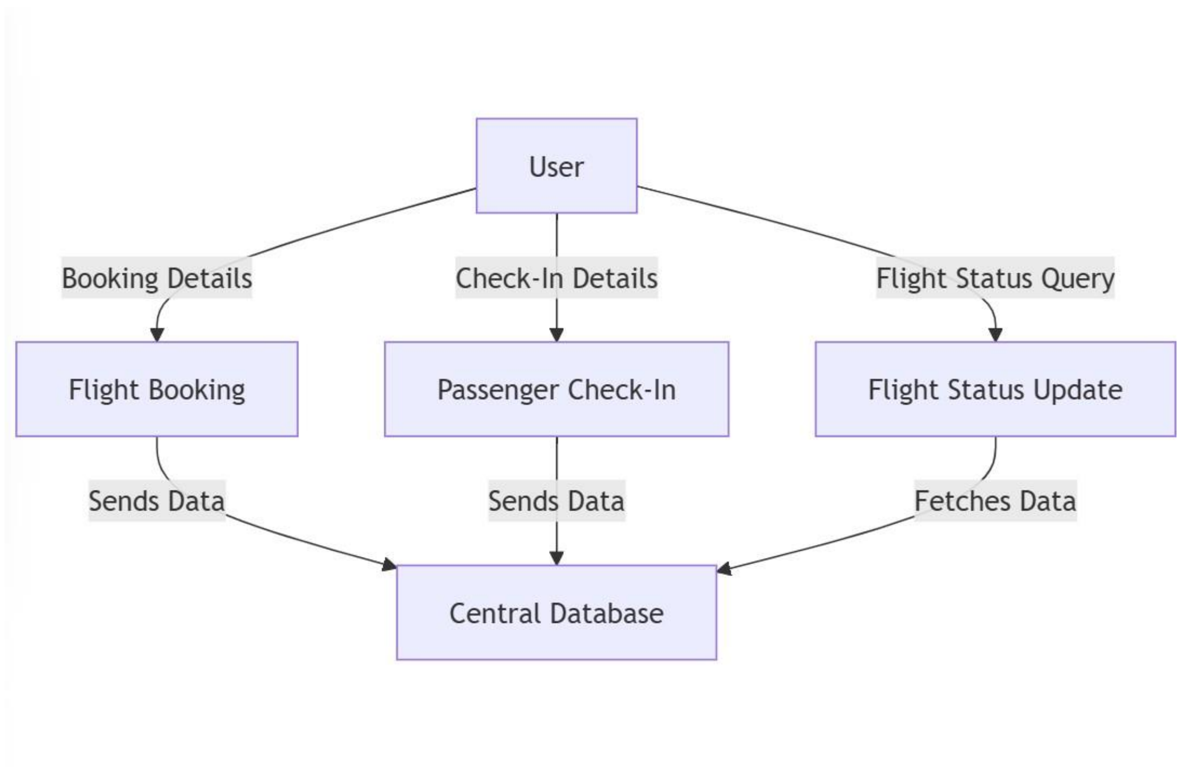


Fig. 4.5 1st Level DFD

3. **Level 2 DFD**:**Booking Management**

a. **Search Flights:**

   i. Passenger provides input (source, destination, date, and class preferences).

   ii. The system retrieves available flights from the database.

b. **Select Flight:**

   i. Passenger selects a desired flight from the search results.

   ii. The system prompts for passenger details (e.g., name, contact information).

c. **Process Payment:**

   i. Payment details are collected and processed through a secure payment gateway.

   ii. Upon successful payment, the booking details are updated in the database.

d. **Ticket Generation:**

   i. A ticket is generated and emailed to the passenger.

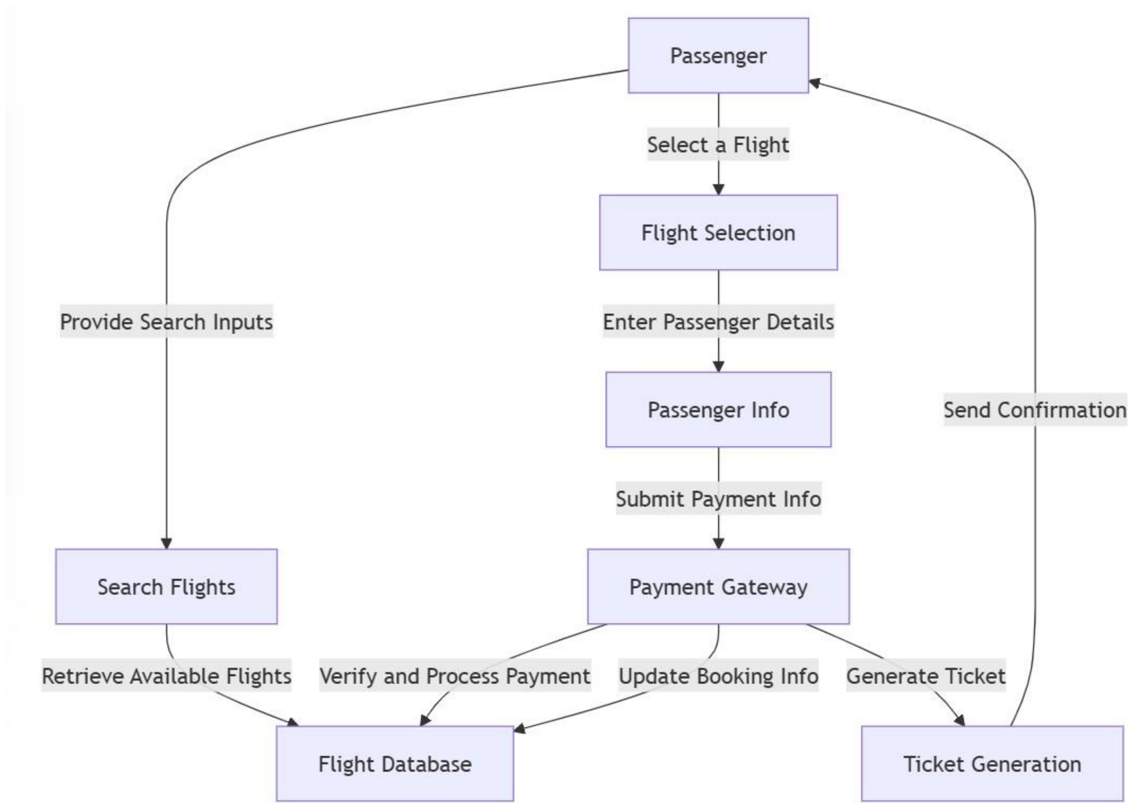


Fig. 4.6 Level 2 DFD:Booking Management

4. Level 2 DFD: Flight Updates

a. **Admin Login:**

   i. Admin logs into the system and accesses the flight status update module.

b. **Select Flight:**

   i. Admin selects a specific flight from the database.

c. **Update Status:**

   i. Admin updates the flight status (e.g., delayed, on-time, canceled).

d. **Notify Passengers:**

i. Updated flight status is saved in the database.

ii. Notifications are sent to passengers via email or SMS**.**

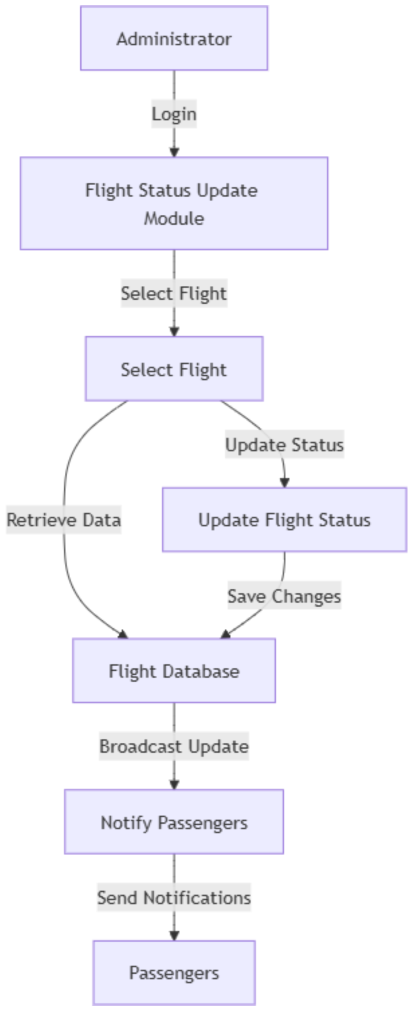


Fig. 4.7 Level 2 DFD: Flight Updates

Chapter 5

Module Descriptions

The AMS is designed with a **modular architecture**, allowing each module to function independently while integrating seamlessly with others. Below are descriptions of the core modules:

5.1 Flight Management Module

This module is the backbone of AMS, handling the scheduling and management of flights.

Features:

1. **Flight Scheduling**: Automates the process of assigning flights to available aircraft.
2. **Status Updates**: Tracks and updates flight statuses (e.g., delayed, on-time).
3. **Route Management**: Stores and manages flight routes, including alternate routes for emergencies.

User Interactions:

- **Administrator**: Creates schedules and updates flight details.
- **Passengers**: View updated flight statuses.

5.2 Passenger Management Module

This module manages passenger interactions with the system, from ticket booking to check-in.

Features:

1. **Booking System**:

    a. Allows passengers to search for flights, select seats, and pay securely using integrated payment gateways.
2. **Check-In System**:
    a. Provides self-check-in functionality, allowing passengers to print boarding passes.
3. **Passenger Records**:
    a. Stores passenger details for generating reports and sending notifications.

User Interactions:

- **Passengers**: Use the module to book tickets and manage their itineraries.
- **Staff**: Assist passengers with bookings and boarding passes.

5.3 Crew Management Module

The crew management module ensures that qualified personnel are assigned to flights and tracks their compliance with aviation regulations.

Features:

1. **Crew Scheduling**: Assigns crew members to flights based on availability and qualifications.
2. **Compliance Monitoring**:
    a. Tracks crew certifications and ensures they are up-to-date.
    b. Prevents crew from being assigned to flights if they exceed their duty hours.

User Interactions:

- **Administrator**: Manages crew assignments and monitors compliance.

5.4 Maintenance Management Module

This module ensures aircraft are safe for operation by tracking maintenance schedules and logs.

Features:

1. **Maintenance Scheduling**: Automatically flags aircraft for maintenance based on usage or time intervals.
2. **Maintenance Logs**:
    a. Tracks completed inspections and repairs.
    b. Generates alerts for overdue maintenance.

User Interactions:

- **Administrator**: Manages maintenance tasks and logs.
- **Staff**: Updates logs after inspections or repairs.

5.5 Tools and Utilities Module

This module provides supporting tools and utilities to enhance system functionality.

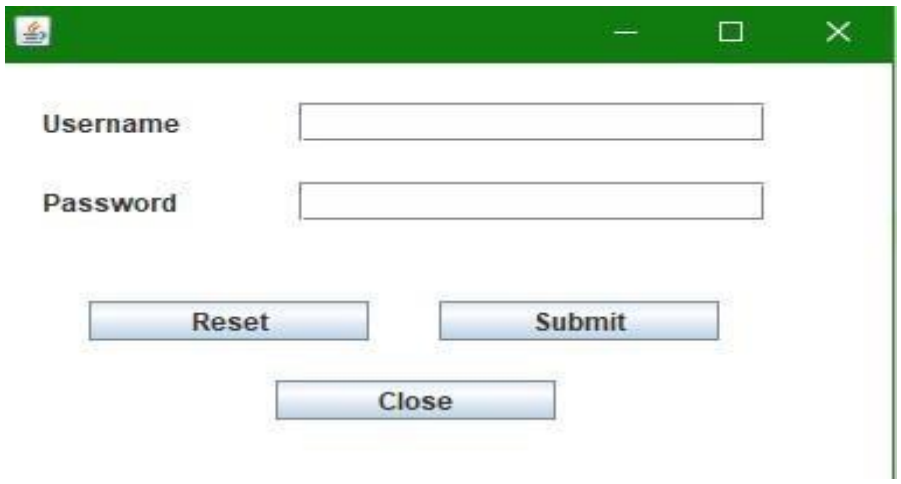
Features:

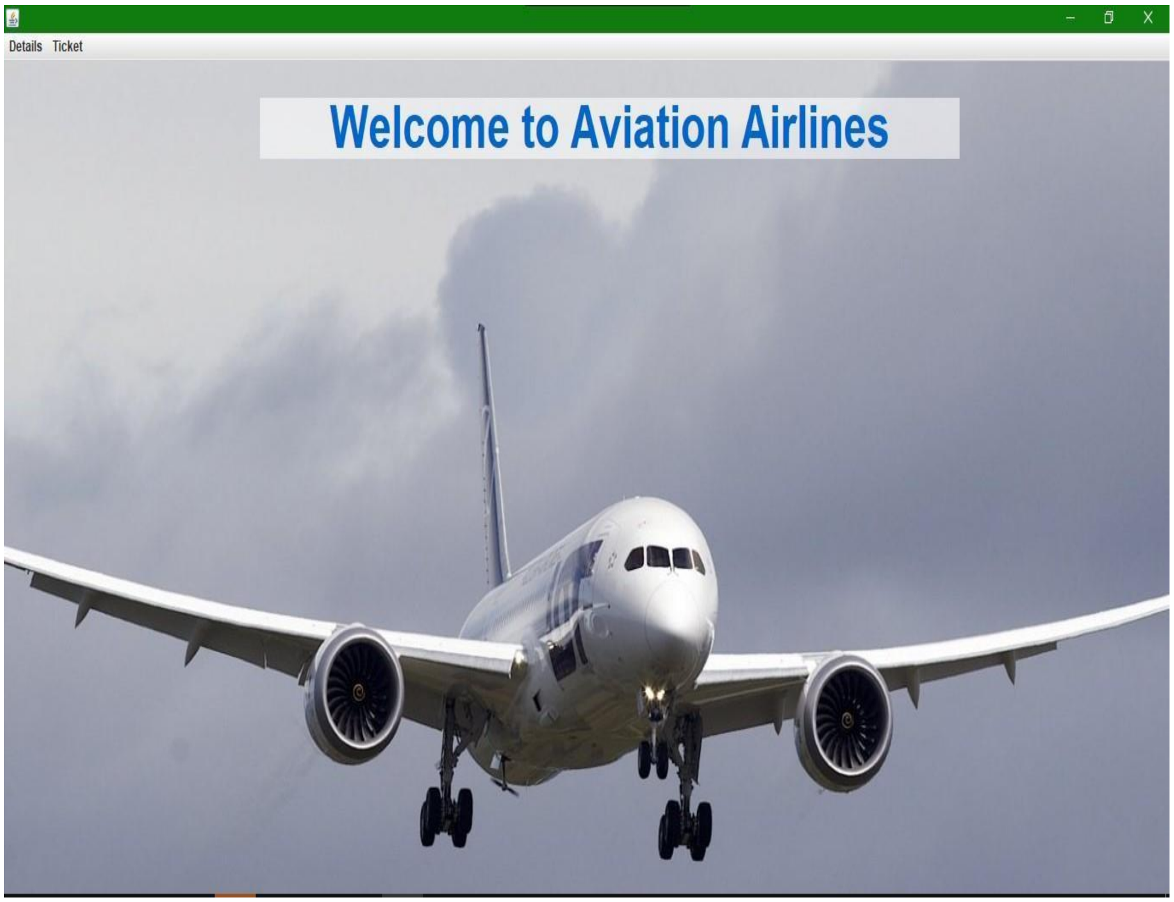
1. **Report Generation**:
    a. Generates detailed reports on flight schedules, crew assignments, and passenger data.
2. **Notifications**:
    a. Sends real-time updates to passengers (e.g., flight delays) and staff (e.g., maintenance alerts).

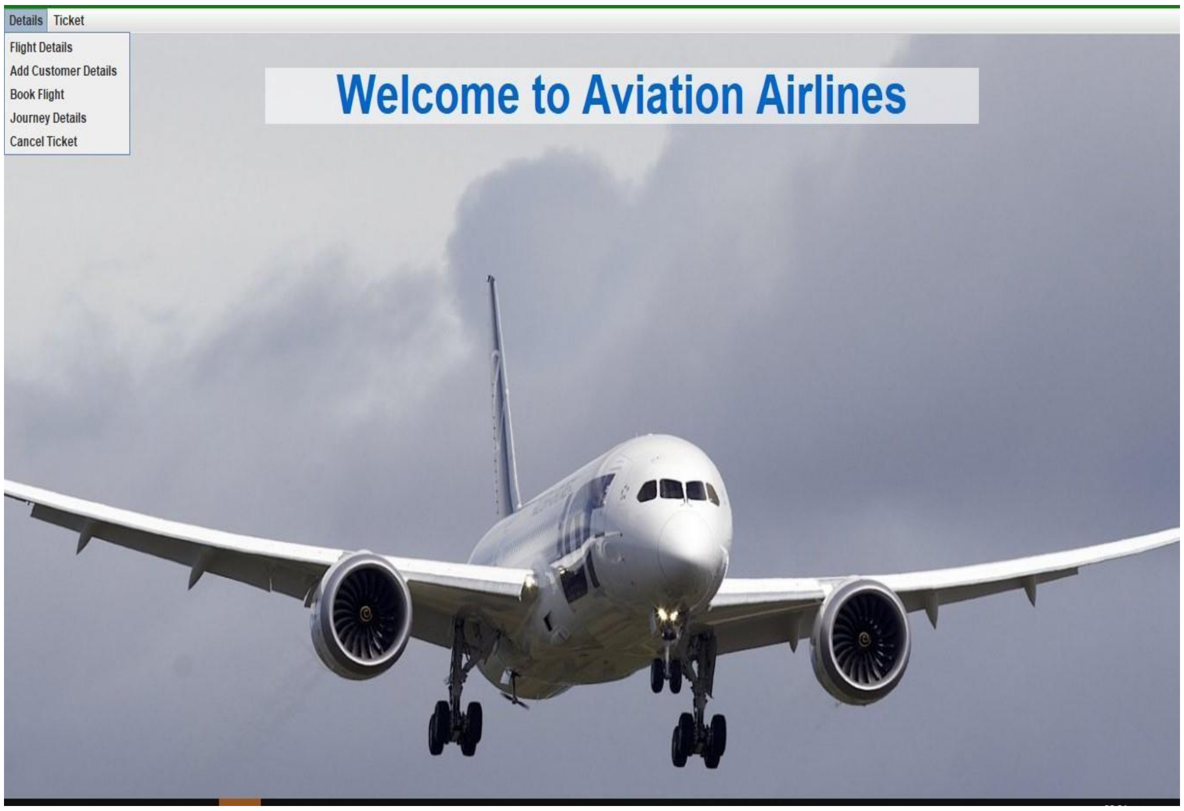
User Interactions:

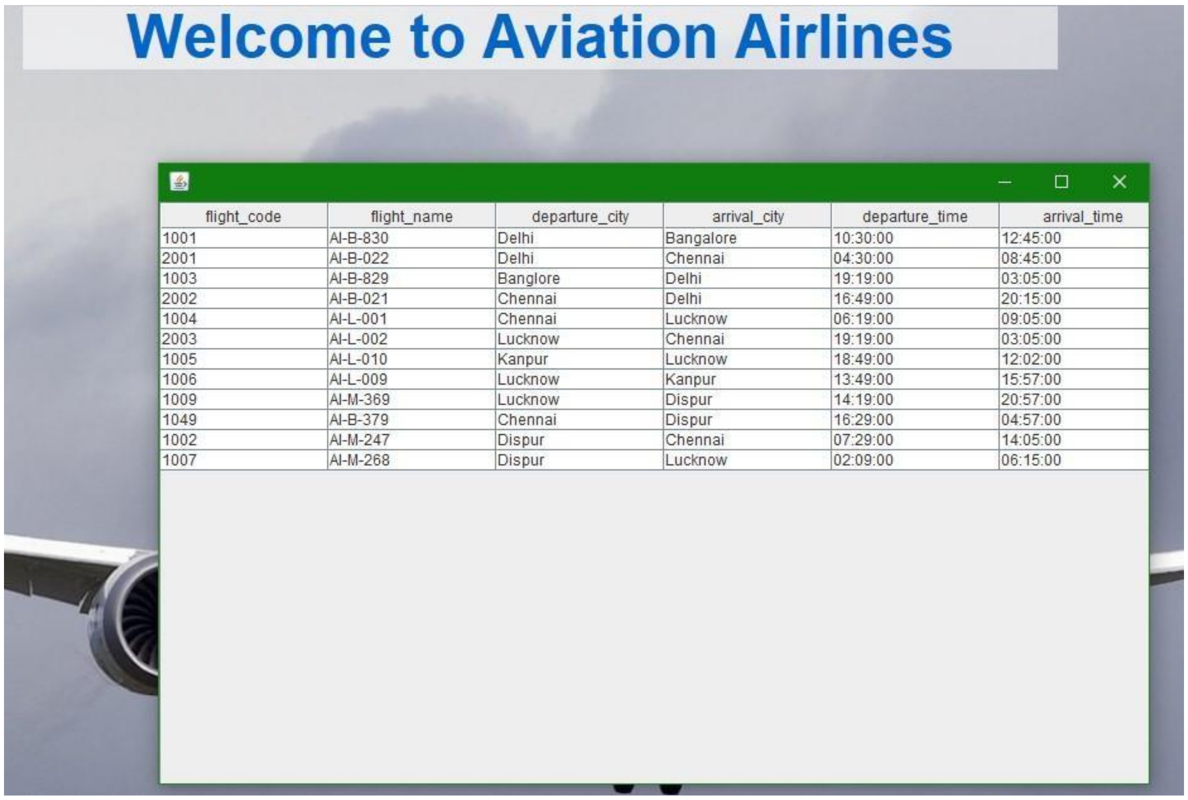
- **Administrator**: Configures system notifications and generates reports.
- **Passengers**: Receive flight-related alerts.

5.1.1 Modules Screenshot.



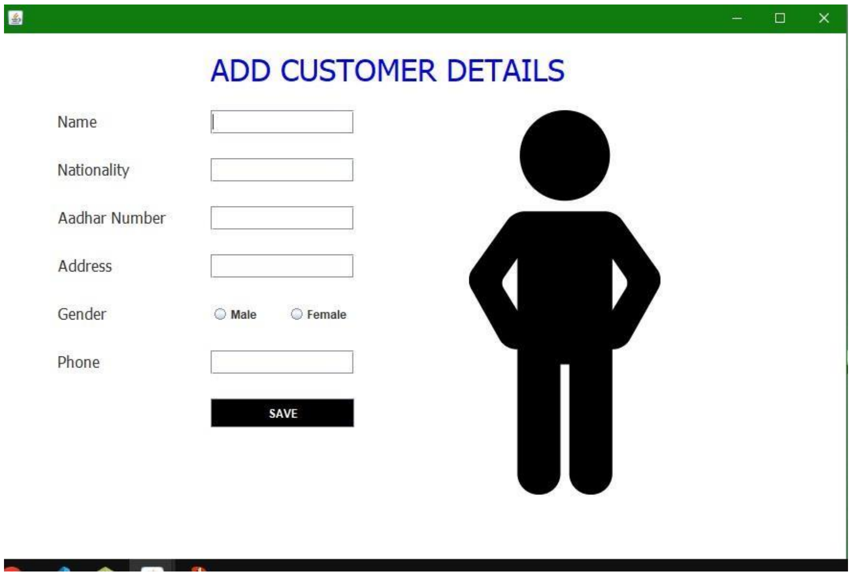

Details
Ticket
Welcome to Aviation Airlines


Details
Ticket
Flight Details
Add Customer Details
Book Flight
Journey Details
Cancel Ticket
Welcome to Aviation Airlines



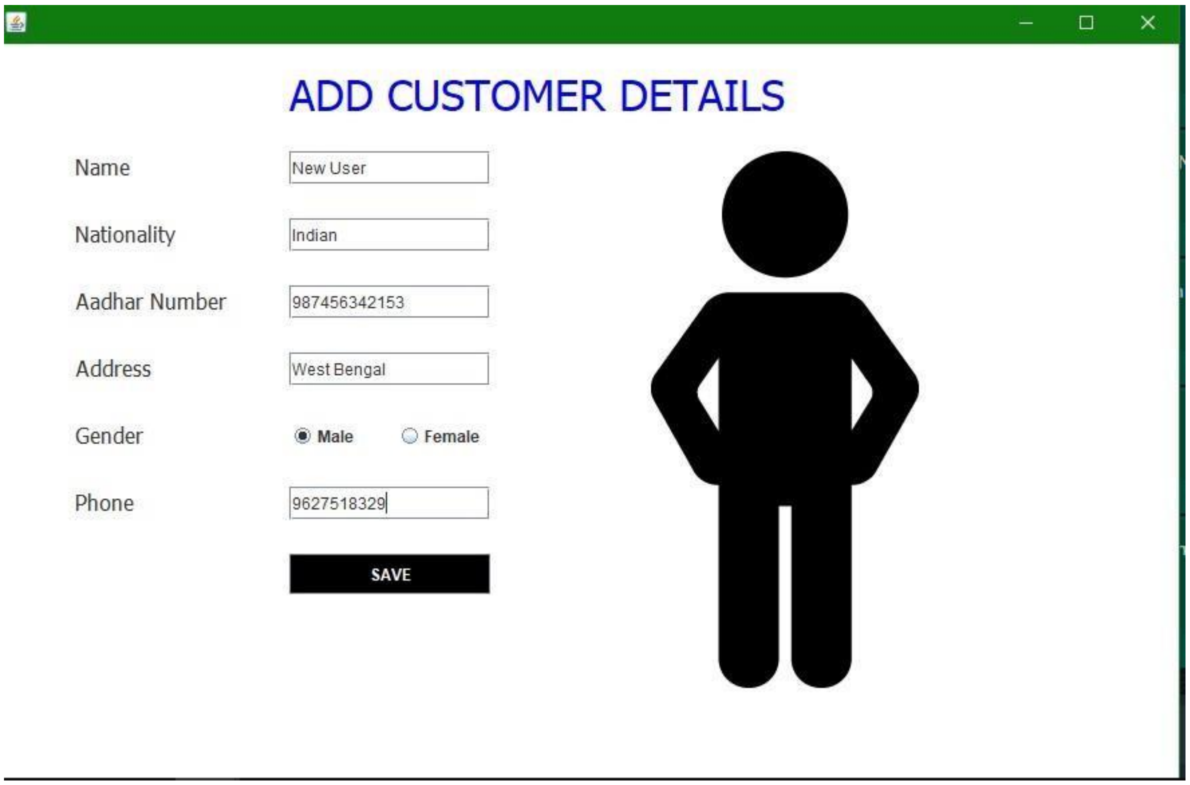
Welcome to Aviation Airlines

flight_code	flight_name	departure_city	arrival_city	departure_time	arrival_time
1001	AI-B-830	Delhi	Bangalore	10:30:00	12:45:00
2001	AI-B-022	Delhi	Chennai	04:30:00	08:45:00
1003	AI-B-829	Banglore	Delhi	19:19:00	03:05:00
2002	AI-B-021	Chennai	Delhi	16:49:00	20:15:00
1004	AI-L-001	Chennai	Lucknow	06:19:00	09:05:00
2003	AI-L-002	Lucknow	Chennai	19:19:00	03:05:00
1005	AI-L-010	Kanpur	Lucknow	18:49:00	12:02:00
1006	AI-L-009	Lucknow	Kanpur	13:49:00	15:57:00
1009	AI-M-369	Lucknow	Dispur	14:19:00	20:57:00
1049	AI-B-379	Chennai	Dispur	16:29:00	04:57:00
1002	AI-M-247	Dispur	Chennai	07:29:00	14:05:00
1007	AI-M-268	Dispur	Lucknow	02:09:00	06:15:00



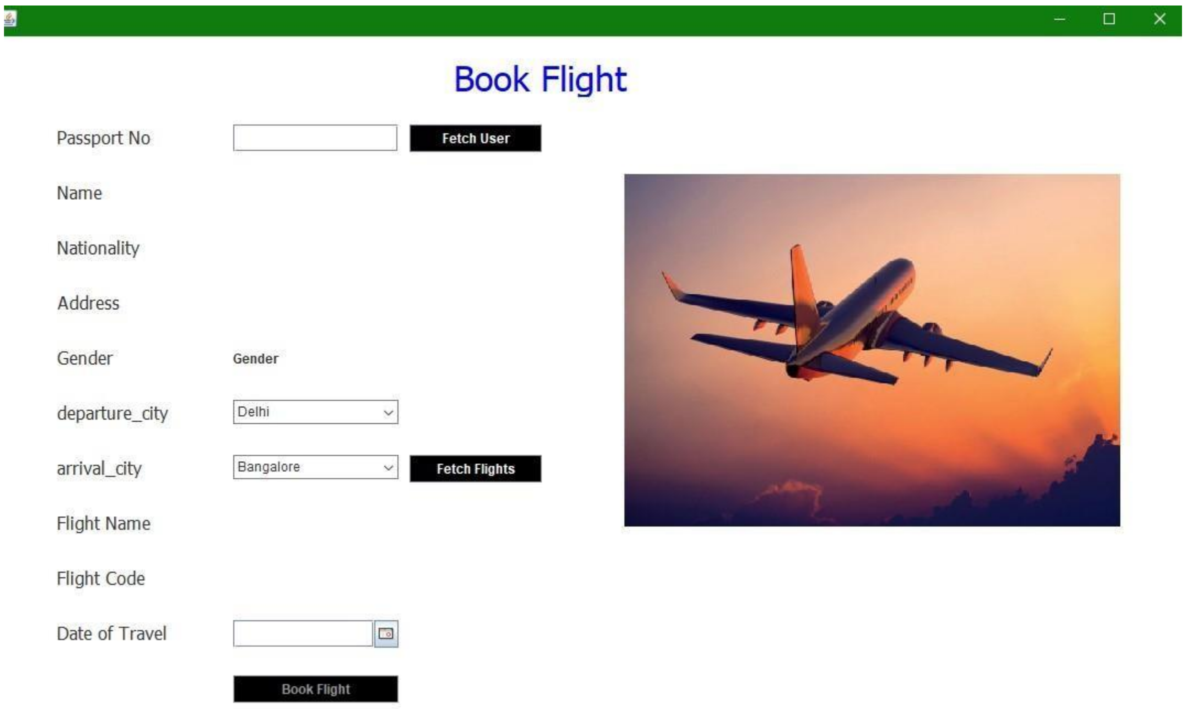
ADD CUSTOMER DETAILS

Name	
Nationality	
Aadhar Number	
Address	
Gender	○ Male ○ Female
Phone	



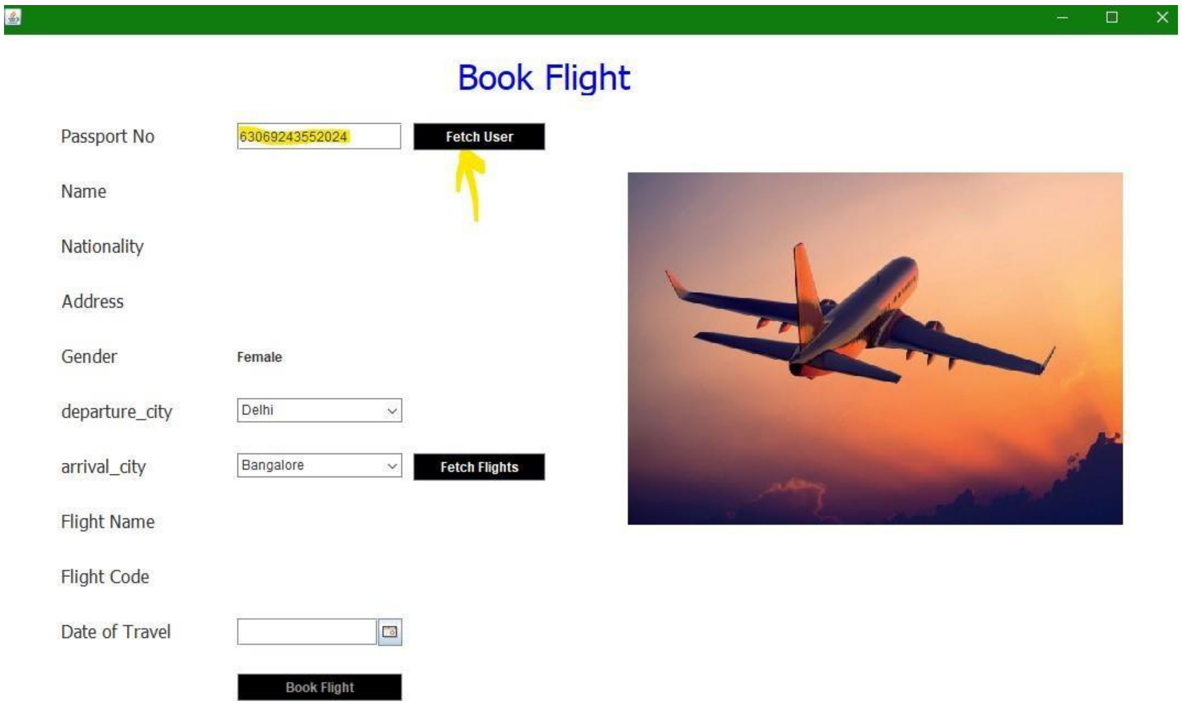
ADD CUSTOMER DETAILS

Name	New User
Nationality	Indian
Aadhar Number	987456342153
Address	West Bengal
Gender	[✓ Male] Female
Phone	9627518329



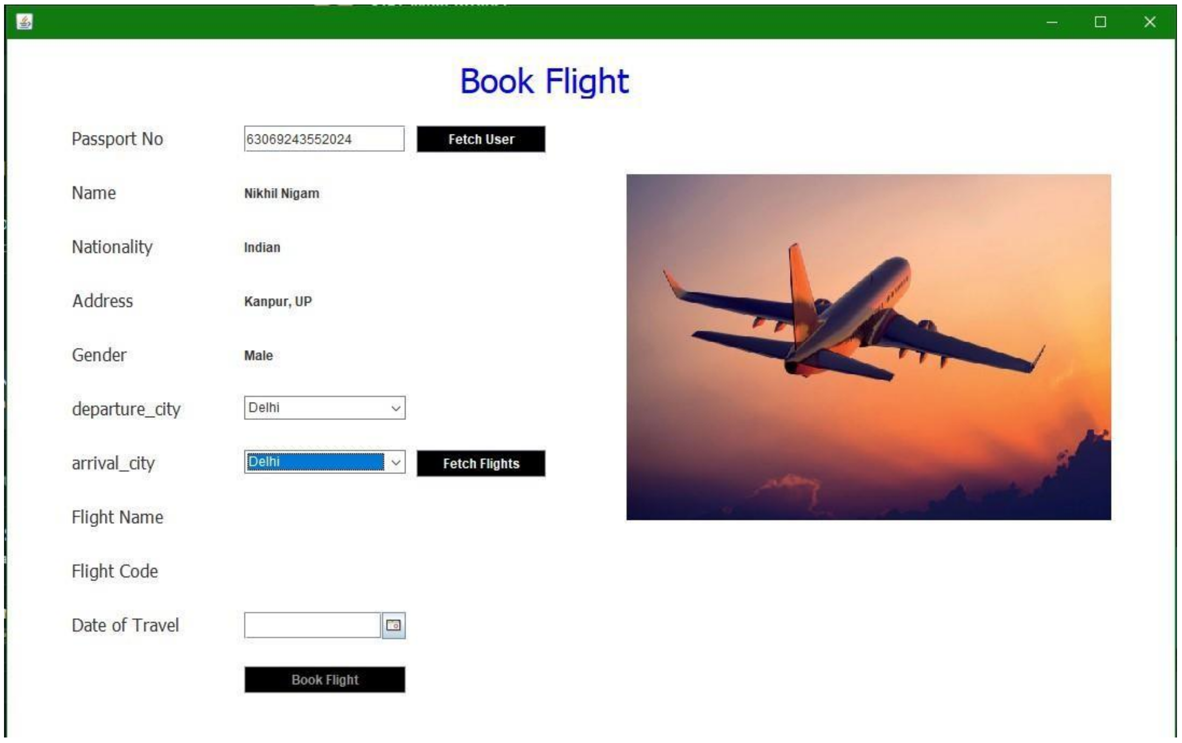
Book Flight

Passport No	
Name	
Nationality	
Address	
Gender	Gender
departure_city	Delhi
arrival_city	Bangalore
Flight Name	
Flight Code	
Date of Travel	



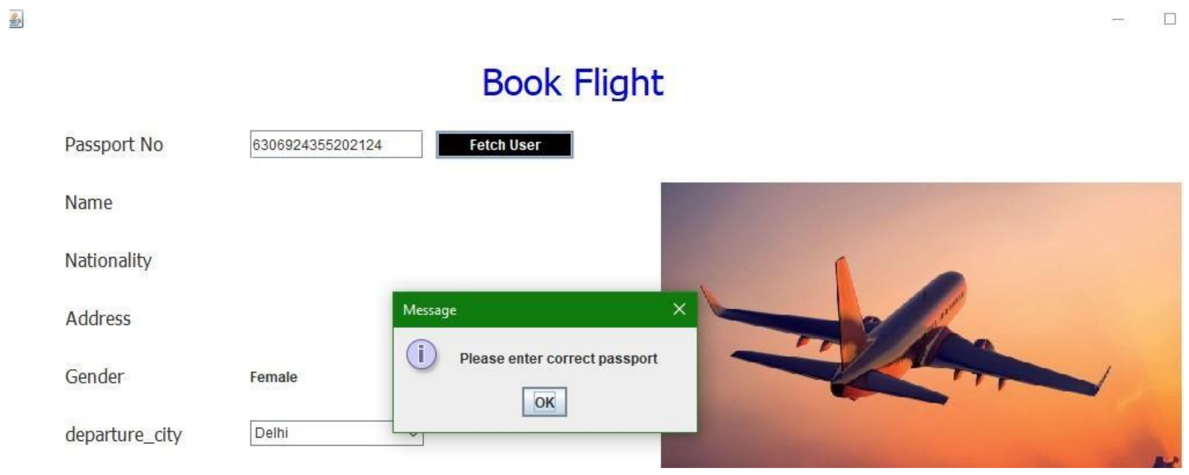
Book Flight

Passport No	63069243552024	Fetch User
Name		
Nationality		
Address		
Gender	Female	
departure_city	Delhi	
arrival_city	Bangalore	Fetch Flights
Flight Name		
Flight Code		
Date of Travel		
	Book Flight	



Book Flight

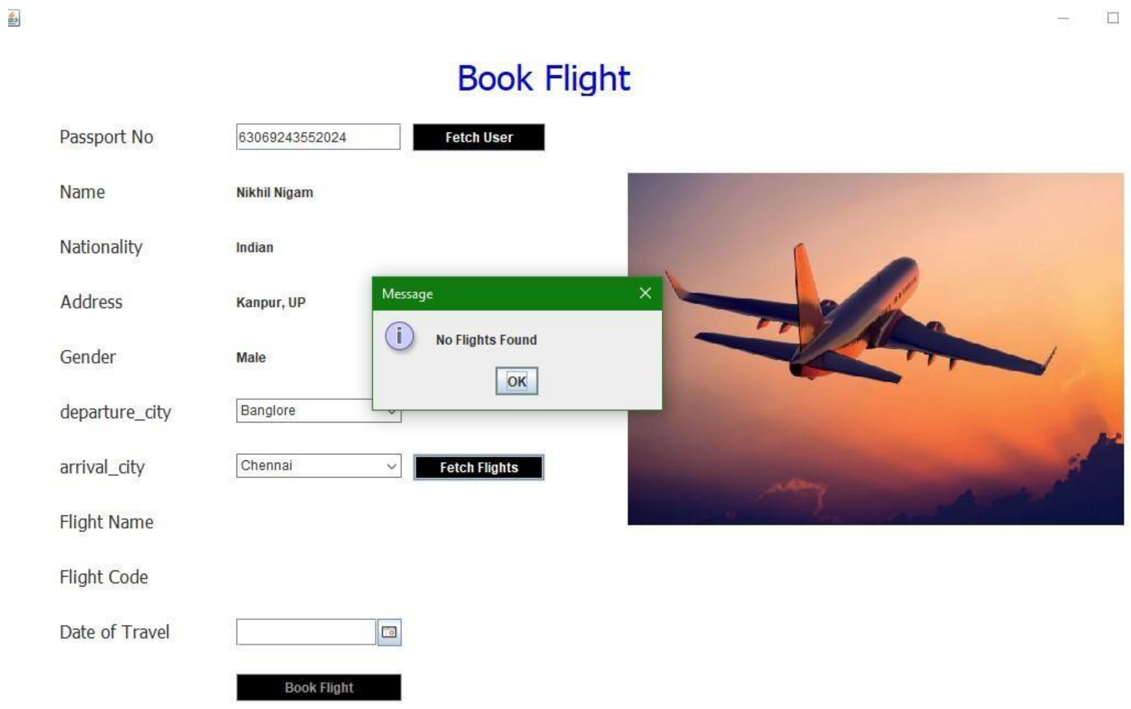
Passport No	63069243552024	Fetch User
Name	Nikhil Nigam	
Nationality	Indian	
Address	Kanpur, UP	
Gender	Male	
departure_city	Delhi	
arrival_city	Delhi	Fetch Flights
Flight Name		
Flight Code		
Date of Travel		
	Book Flight	


Book Flight

Passport No	6306924355202124
Name	
Nationality	
Address	
Gender	Female
departure_city	Delhi

Message

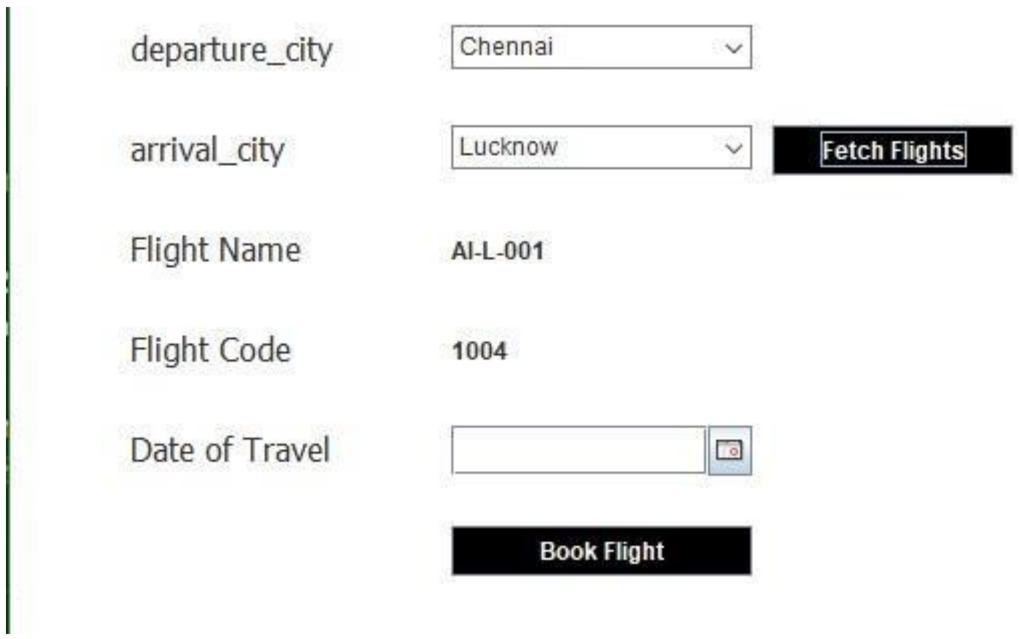
Please enter correct passport


Book Flight

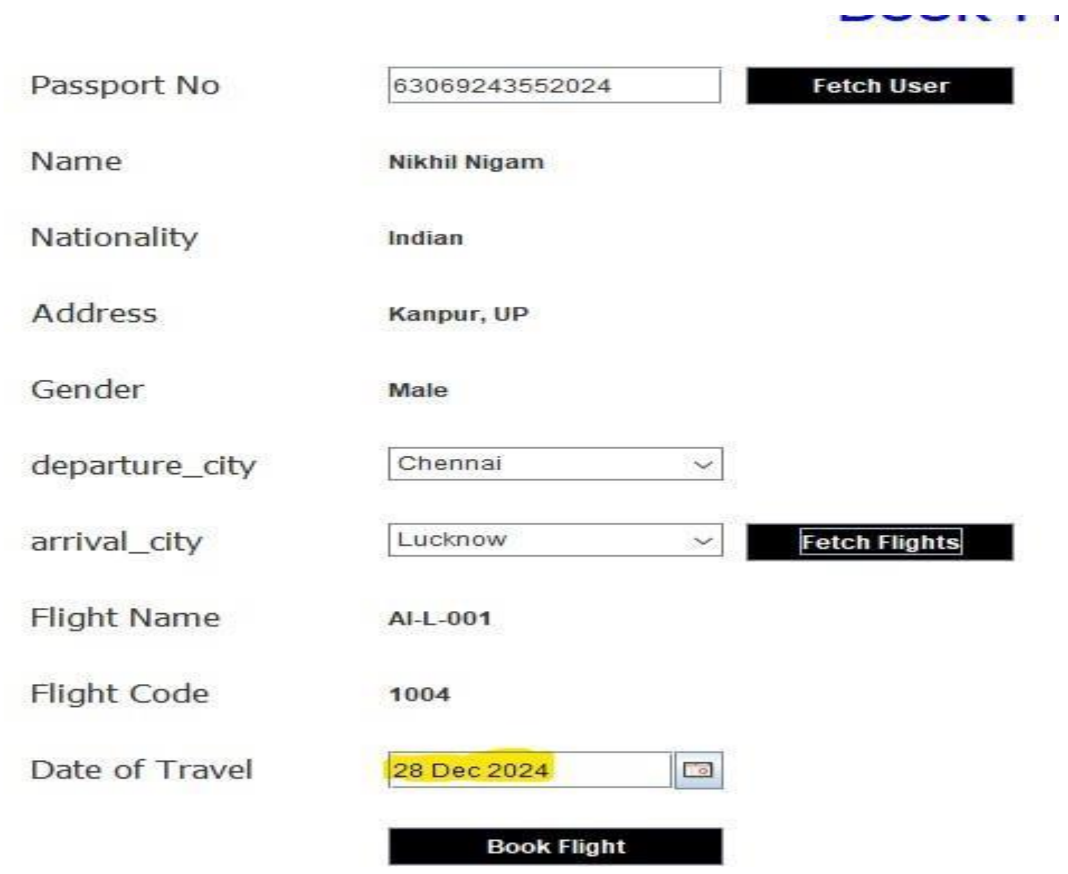
Passport No	63069243552024
Name	Nikhil Nigam
Nationality	Indian
Address	Kanpur, UP
Gender	Male
departure_city	Banglore
arrival_city	Chennai
Flight Name	
Flight Code	
Date of Travel	

Message

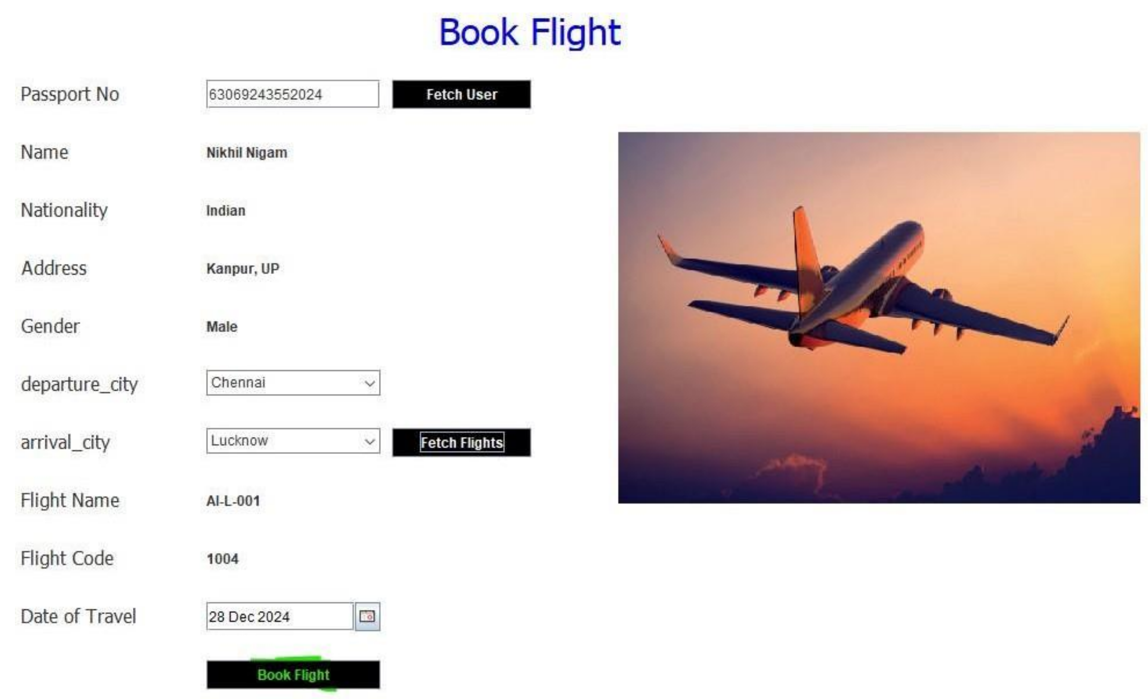
No Flights Found



departure_city	Chennai
arrival_city	Lucknow
Flight Name	AI-L-001
Flight Code	1004
Date of Travel	

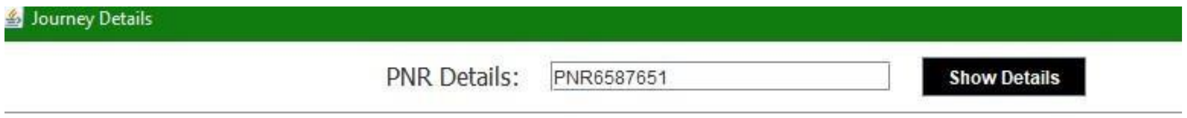


Passport No	63069243552024
Name	Nikhil Nigam
Nationality	Indian
Address	Kanpur, UP
Gender	Male
departure_city	Chennai
arrival_city	Lucknow
Flight Name	AI-L-001
Flight Code	1004
Date of Travel	28 Dec 2024



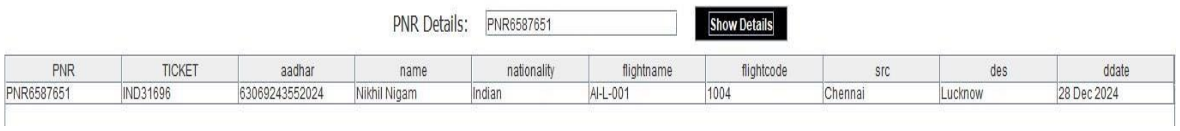
Book Flight

Passport No	63069243552024
Name	Nikhil Nigam
Nationality	Indian
Address	Kanpur, UP
Gender	Male
departure_city	Chennai
arrival_city	Lucknow
Flight Name	AI-L-001
Flight Code	1004
Date of Travel	28 Dec 2024



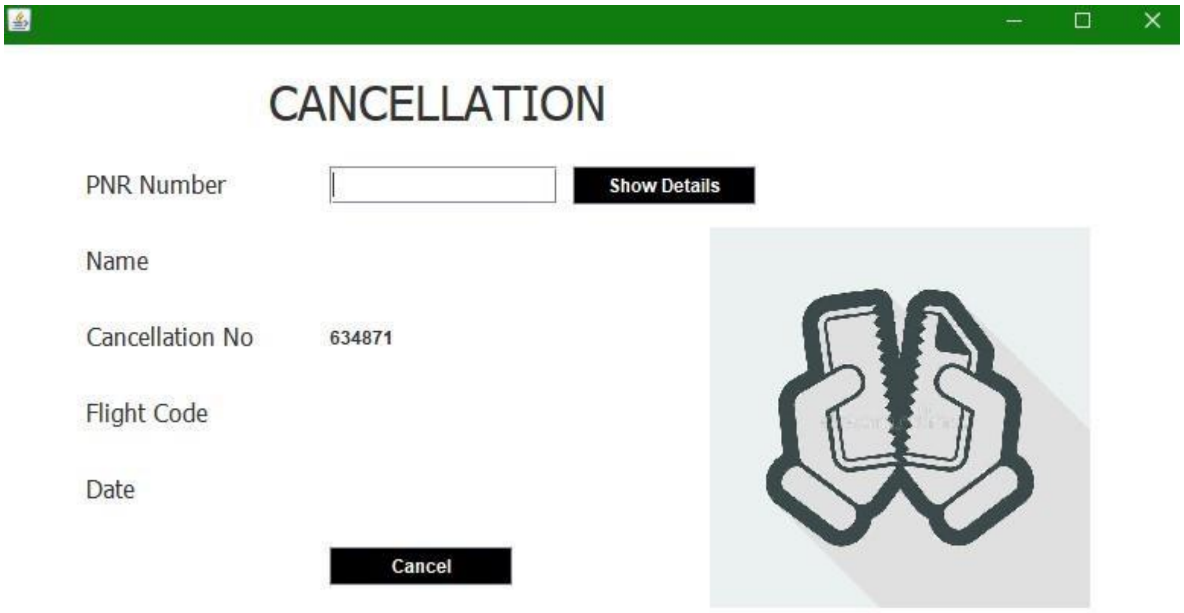
Journey Details

PNR Details: PNR6587651



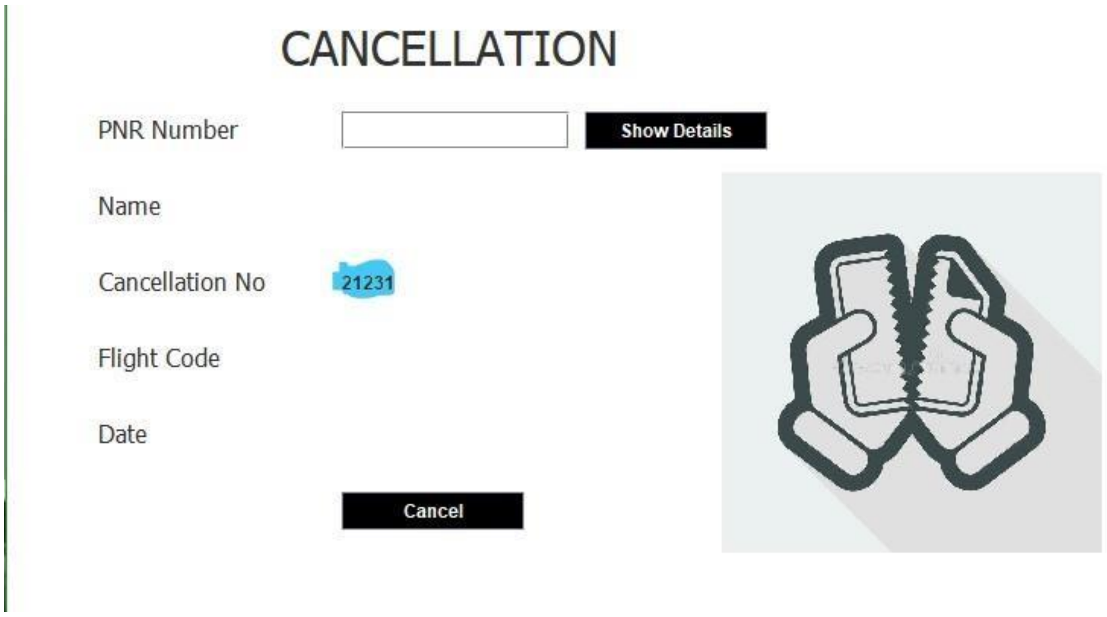
PNR Details: PNR6587651

PNR	TICKET	aadhar	name	nationality	flightname	flightcode	src	des	ddate
PNR6587651	IND31696	63069243552024	Nikhil Nigam	Indian	AI-L-001	1004	Chennai	Lucknow	28 Dec 2024



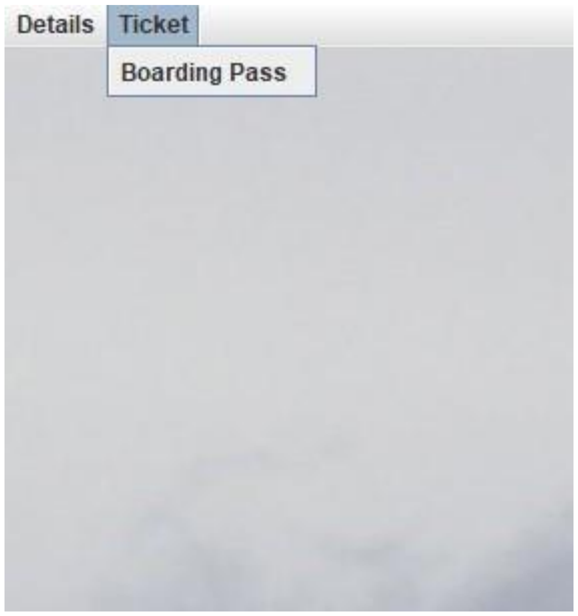
CANCELLATION

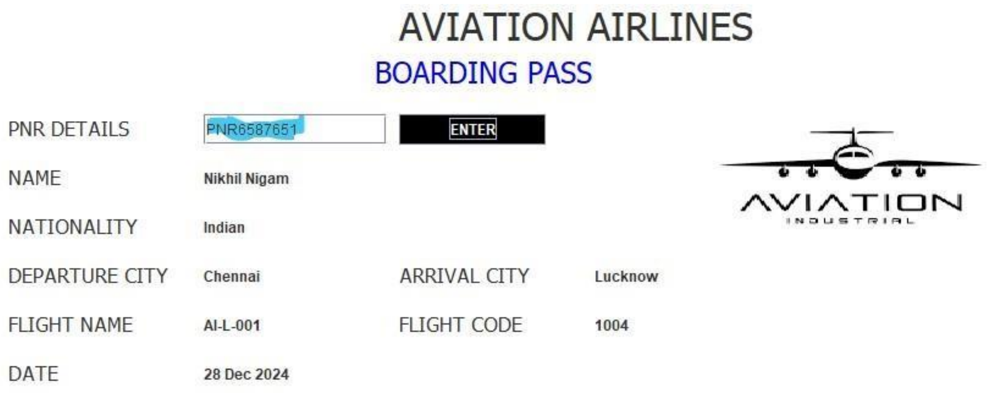
PNR Number	
Name	
Cancellation No	634871
Flight Code	
Date	



CANCELLATION

PNR Number	
Name	
Cancellation No	21231
Flight Code	
Date	




AVIATION AIRLINES

BOARDING PASS

PNR DETAILS	PNR6587651	ENTER	
NAME	Nikhil Nigam		
NATIONALITY	Indian		
DEPARTURE CITY	Chennai	ARRIVAL CITY	Lucknow
FLIGHT NAME	AI-L-001	FLIGHT CODE	1004
DATE	28 Dec 2024		

Chapter 6

Implementation

6.1 Database Design

The **database** is the core of the **Aviation Management System (AMS)**. It stores all the critical information related to flights, passengers, crew, and maintenance. The design focuses on ensuring data integrity, scalability, and efficiency.

Entities and Relationships

The database design includes several key tables, each representing an entity in the system:

1. **Flights**:
   a. Fields: Flight ID, Flight Number, Departure Time, Arrival Time, Aircraft ID, Crew ID.
   b. Relationships: Linked to the **Crew** and **Maintenance** tables.
2. **Passengers**:
   a. Fields: Passenger ID, Name, Contact Information, Ticket Number, Flight ID.
   b. Relationships: Linked to **Flights** via bookings.
3. **Crew**:
   a. Fields: Crew ID, Name, Position, Certifications, Availability.
   b. Relationships: Linked to **Flights** for crew assignments.
4. **Maintenance**:
   a. Fields: Maintenance ID, Aircraft ID, Maintenance Type, Scheduled Date, Completed Date.
   b. Relationships: Linked to **Aircraft** to track maintenance events.
5. **Aircraft**:
   a. Fields: Aircraft ID, Model, Year of Manufacture, Maintenance Log, Status.
   b. Relationships: Linked to **Maintenance** and **Flights**.

Normalization

- The database follows **third normal form (3NF)** to reduce redundancy and ensure data integrity. This helps avoid issues like duplicate passenger records and inconsistent flight schedules.

Indexes

- **Primary Keys**: Each table has a unique primary key (e.g., Flight ID, Passenger ID).
- **Foreign Keys**: Tables are linked with foreign keys (e.g., Passenger ID in the Flights table).

6.2 Backend Development

The **backend** of the AMS handles all business logic and data processing. It is developed using **Java** and **Java Servlets** to ensure high performance, reliability, and scalability.

Java Servlets

- Java Servlets are used to handle user requests from the frontend (GUI) and process them on the server-side.
- Servlets are responsible for:
    a. **Flight Scheduling**: Assigning flights to available aircraft and crew.
    b. **Passenger Management**: Adding and retrieving passenger data from the database.
    c. **Maintenance Tracking**: Updating maintenance schedules and status.

JDBC (Java Database Connectivity)

- **JDBC** is used to connect Java Servlets to the **MySQL** database.
- It allows data to be retrieved, inserted, and updated efficiently.

Business Logic Layer

- The core business logic is housed in **Java classes**, which handle operations such as:
    a. **Flight Booking**: Checking flight availability, calculating ticket prices, and confirming bookings.
    b. **Crew Assignment**: Ensuring that available crew members meet qualifications for each flight.
    c. **Maintenance Scheduling**: Flagging aircraft for maintenance and tracking maintenance history.

6.3 Frontend Development

The **frontend** of AMS provides an intuitive user interface for administrators, staff, and passengers. It is developed using **JavaFX** for its rich graphical capabilities and ease of use.

JavaFX

- **JavaFX** provides a modern GUI for the system, ensuring that users can interact with the system seamlessly.
    a. **Passenger Interface**: Allows users to search for flights, book tickets, check-in, and view flight details.
    b. **Staff Interface**: Allows airport staff to manage passenger check-ins, update flight statuses, and handle boarding.
    c. **Administrator Interface**: Provides access to more advanced functionalities like flight scheduling, crew management, and reporting.

Responsive Design

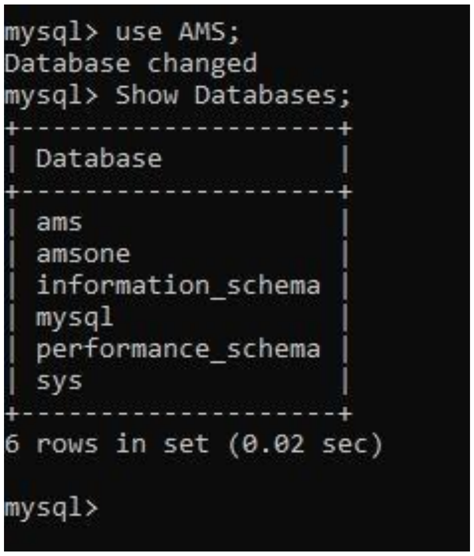
- The frontend is designed to be responsive, ensuring that the interface is accessible from both desktop and mobile devices.

6.4 Integration of Modules

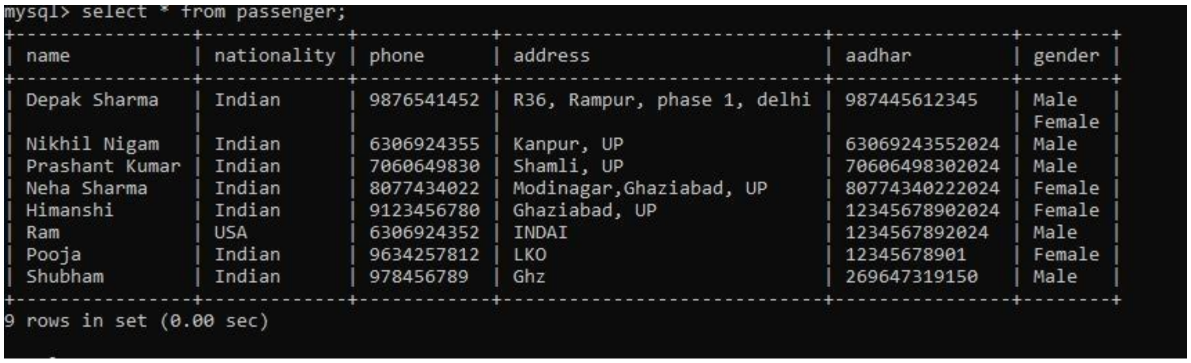
The integration of AMS modules ensures seamless data flow between different parts of the system:

1. **Flight Management and Passenger Management**: Passengers are linked to flights through bookings. When a passenger books a ticket, the flight status is updated in real-time.
2. **Crew Management and Flight Scheduling**: Crew members are assigned to flights based on their availability and qualifications.
3. **Maintenance Management and Flight Scheduling**: Flights are automatically flagged for rescheduling if the aircraft is due for maintenance.

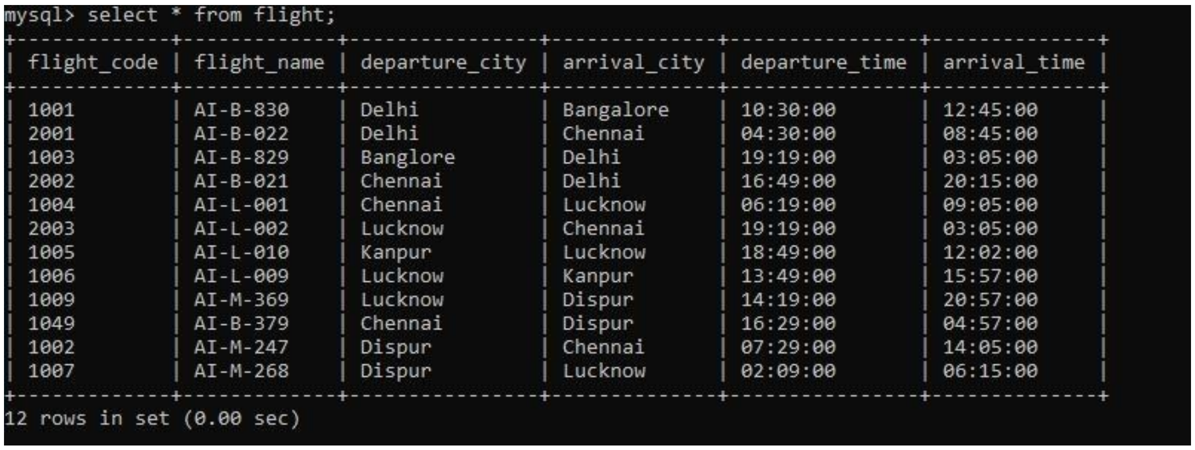
6.1.1 Database Screenshot.



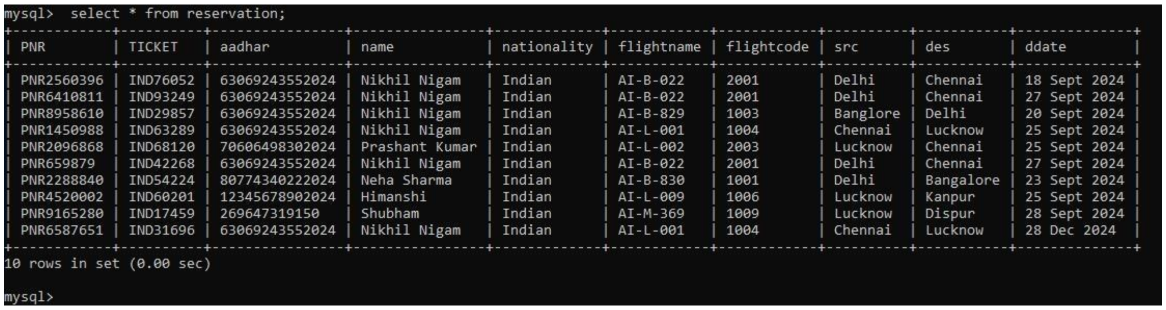
```
mysql> use AMS;
Database changed
mysql> Show Databases;
+--------------------+
| Database           |
+--------------------+
| ams                |
| amsone             |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.02 sec)

mysql>
```



```
mysql> select * from passenger;
+----------------+-------------+------------+-----------------------------+----------------+--------+
| name           | nationality | phone      | address                     | aadhar         | gender |
+----------------+-------------+------------+-----------------------------+----------------+--------+
| Depak Sharma   | Indian      | 9876541452 | R36, Rampur, phase 1, delhi | 987445612345   | Male   |
|                |             |            |                             |                | Female |
| Nikhil Nigam   | Indian      | 6306924355 | Kanpur, UP                  | 63069243552024 | Male   |
| Prashant Kumar | Indian      | 7060649830 | Shamli, UP                  | 70606498302024 | Male   |
| Neha Sharma    | Indian      | 8077434022 | Modinagar,Ghaziabad, UP     | 80774340222024 | Female |
| Himanshi       | Indian      | 9123456780 | Ghaziabad, UP               | 12345678902024 | Female |
| Ram            | USA         | 6306924352 | INDAI                       | 1234567892024  | Male   |
| Pooja          | Indian      | 9634257812 | LKO                         | 12345678901    | Female |
| Shubham        | Indian      | 978456789  | Ghz                         | 269647319150   | Male   |
+----------------+-------------+------------+-----------------------------+----------------+--------+
9 rows in set (0.00 sec)
```



```
mysql> select * from flight;
+-------------+-------------+----------------+--------------+----------------+--------------+
| flight_code | flight_name | departure_city | arrival_city | departure_time | arrival_time |
+-------------+-------------+----------------+--------------+----------------+--------------+
| 1001        | AI-B-830    | Delhi          | Bangalore    | 10:30:00       | 12:45:00     |
| 2001        | AI-B-022    | Delhi          | Chennai      | 04:30:00       | 08:45:00     |
| 1003        | AI-B-829    | Banglore       | Delhi        | 19:19:00       | 03:05:00     |
| 2002        | AI-B-021    | Chennai        | Delhi        | 16:49:00       | 20:15:00     |
| 1004        | AI-L-001    | Chennai        | Lucknow      | 06:19:00       | 09:05:00     |
| 2003        | AI-L-002    | Lucknow        | Chennai      | 19:19:00       | 03:05:00     |
| 1005        | AI-L-010    | Kanpur         | Lucknow      | 18:49:00       | 12:02:00     |
| 1006        | AI-L-009    | Lucknow        | Kanpur       | 13:49:00       | 15:57:00     |
| 1009        | AI-M-369    | Lucknow        | Dispur       | 14:19:00       | 20:57:00     |
| 1049        | AI-B-379    | Chennai        | Dispur       | 16:29:00       | 04:57:00     |
| 1002        | AI-M-247    | Dispur         | Chennai      | 07:29:00       | 14:05:00     |
| 1007        | AI-M-268    | Dispur         | Lucknow      | 02:09:00       | 06:15:00     |
+-------------+-------------+----------------+--------------+----------------+--------------+
12 rows in set (0.00 sec)
```



```
mysql>  select * from reservation;
+------------+----------+----------------+----------------+-------------+------------+------------+----------+-----------+--------------+
| PNR        | TICKET   | aadhar         | name           | nationality | flightname | flightcode | src      | des       | ddate        |
+------------+----------+----------------+----------------+-------------+------------+------------+----------+-----------+--------------+
| PNR2560396 | IND76052 | 63069243552024 | Nikhil Nigam   | Indian      | AI-B-022   | 2001       | Delhi    | Chennai   | 18 Sept 2024 |
| PNR6410811 | IND93249 | 63069243552024 | Nikhil Nigam   | Indian      | AI-B-022   | 2001       | Delhi    | Chennai   | 27 Sept 2024 |
| PNR8958610 | IND29857 | 63069243552024 | Nikhil Nigam   | Indian      | AI-B-829   | 1003       | Banglore | Delhi     | 20 Sept 2024 |
| PNR1450988 | IND63289 | 63069243552024 | Nikhil Nigam   | Indian      | AI-L-001   | 1004       | Chennai  | Lucknow   | 25 Sept 2024 |
| PNR2096868 | IND68120 | 70606498302024 | Prashant Kumar | Indian      | AI-L-002   | 2003       | Lucknow  | Chennai   | 25 Sept 2024 |
| PNR659879  | IND42268 | 63069243552024 | Nikhil Nigam   | Indian      | AI-B-022   | 2001       | Delhi    | Chennai   | 27 Sept 2024 |
| PNR2288840 | IND54224 | 80774340222024 | Neha Sharma    | Indian      | AI-B-830   | 1001       | Delhi    | Bangalore | 23 Sept 2024 |
| PNR4520002 | IND60201 | 12345678902024 | Himanshi       | Indian      | AI-L-009   | 1006       | Lucknow  | Kanpur    | 25 Sept 2024 |
| PNR9165280 | IND17459 | 269647319150   | Shubham        | Indian      | AI-M-369   | 1009       | Lucknow  | Dispur    | 28 Sept 2024 |
| PNR6587651 | IND31696 | 63069243552024 | Nikhil Nigam   | Indian      | AI-L-001   | 1004       | Chennai  | Lucknow   | 28 Dec 2024  |
+------------+----------+----------------+----------------+-------------+------------+------------+----------+-----------+--------------+
10 rows in set (0.00 sec)

mysql>
```

Chapter 7

Testing and Quality Assurance

7.1 Testing Overview

Testing ensures that the **Aviation Management System (AMS)** meets all functional and non-functional requirements. The testing phases include unit testing, integration testing, system testing, and acceptance testing.

Testing Phases

1. **Unit Testing**

2. Unit tests focus on individual components of the AMS, ensuring that each function works as expected in isolation.

   a. **Flight Booking**: Tests check if the booking process correctly handles seat availability, payment processing, and ticket issuance.
   b. **Crew Assignment**: Ensures crew members are assigned only to flights for which they are qualified.
   c. **Maintenance Scheduling**: Ensures that aircraft maintenance is scheduled appropriately and that flights are not assigned to maintenance-requiring aircraft.

3. **Integration Testing**

Integration tests focus on how different modules interact with each other. For instance:

   a. **Flight Management and Passenger Management**: Ensures that passenger bookings and flight details sync correctly across modules.
   b. **Crew Management and Flight Scheduling**: Verifies that crew members are correctly assigned to flights based on availability.

4. **System Testing**

This phase tests the entire AMS as a complete system. It verifies that all components (database, backend, frontend) work together seamlessly to perform the required operations.

System tests ensure that the AMS can handle multiple users simultaneously, process bookings efficiently, and update flight statuses in real time.

5. **Acceptance Testing**

Acceptance testing is conducted by end-users to ensure the system meets their needs. It includes:

a. **User Interface Testing**: Ensures that passengers, staff, and administrators can interact with the system easily.
b. **Real-world Scenarios**: Tests typical workflows such as booking a flight, checking in, and scheduling maintenance.

7.2 Test Cases and Scenarios

1. **Flight Booking Test Case**

Objective: Ensure that passengers can book tickets successfully.

a. **Test Steps**:
    i. Search for available flights.
    ii. Select a flight and enter passenger details.
    iii. Complete the payment.
b. **Expected Result**: Booking confirmation is displayed, and passenger data is stored in the database.

2. **Crew Assignment Test Case**

Objective: Ensure crew members are assigned to the correct flights.

a. **Test Steps**:
    i. View available crew members.
    ii. Assign crew to a specific flight based on their qualifications and availability.
b. **Expected Result**: Crew assignments are updated, and notifications are sent to the crew.

3. **Maintenance Tracking Test Case**

Objective: Ensure that aircraft requiring maintenance are flagged and rescheduled.

a. **Test Steps**:
   i. Schedule a maintenance check for an aircraft.
   ii. Attempt to schedule a flight for the same aircraft.
b. **Expected Result**: The system should prevent scheduling the flight until maintenance is completed.

4. **Passenger Check-In Test Case**

Objective: Ensure passengers can check in and receive boarding passes.

a. **Test Steps**:
   i. Enter passenger details.
   ii. Verify flight booking.
   iii. Issue a boarding pass.
b. **Expected Result**: Boarding pass is generated and available for the passenger.

7.3 Bug Tracking and Resolution

Bugs and issues identified during testing were tracked using a bug tracking tool (e.g., **Jira** or **Bugzilla**). Each issue was assigned a priority, and developers worked to fix the bugs in the following manner:

1. **Bug Identification**: Developers and testers log bugs during the testing phase.
2. **Bug Resolution**: Each bug is assigned to the appropriate developer for resolution.
3. **Retesting**: Once the bug is fixed, the affected tests are rerun to ensure that the system works as expected.

7.4 Performance Testing

Performance testing ensures that the AMS can handle the expected load. Key performance indicators (KPIs) include:

1. **Response Time**: The time it takes for the system to respond to user inputs (e.g., booking tickets or checking in).

2. **Throughput**: The number of transactions the system can handle per second, especially during peak hours.
3. **System Scalability**: The ability of the system to scale as the number of users or data increases.

7.5 Security Testing

Security testing focuses on ensuring that AMS is resistant to unauthorized access, data breaches, and cyberattacks. Tests include:

1. **Penetration Testing**: Attempts to exploit vulnerabilities in the system.
2. **Access Control Testing**: Ensures that only authorized users can access sensitive data.

Chapter 8

Deployment

8.1 Deployment Process

Deployment is the final stage where the **Aviation Management System (AMS)** is moved from the development environment to a live production environment. The deployment process consists of several key stages:

1. Development Environment

The **development environment** is where the initial coding and testing take place. Here, developers work on implementing the core functionalities of the AMS, including backend logic, frontend interface, and database design. The development environment typically includes:

- **Local Servers**: Hosting the application and database for local testing.
- **Development Tools**: IDEs such as Eclipse for coding, MySQL Workbench for managing databases, and Git for version control.

Steps:

- Code is written and tested in isolated components.
- Integration testing is conducted to ensure that modules work together properly.
- Frequent debugging and unit testing ensure the system is stable.

2. Staging Environment

The **staging environment** simulates real-world conditions and allows the final round of testing before deployment to production. This environment mirrors the production environment as closely as possible.

- **Staging Servers**: Used to host the AMS in a near-production state.
- **Mock Data**: The system is tested using data similar to what would be encountered in real-world scenarios.

- **User Acceptance Testing (UAT)**: End-users (such as airline staff) test the system's functionalities to ensure that it meets their needs.

Steps:

- Final bug fixes are implemented based on feedback from users.
- Performance and security testing is conducted to verify scalability and resistance to attacks.
- User feedback is incorporated into the system, ensuring smooth deployment.

3. Production Environment

The **production environment** is the live environment where the AMS is made available to users. It is hosted on cloud or on-premises servers and directly interacts with real customer data.

- **Cloud Hosting**: AMS is deployed on cloud servers (e.g., AWS, Azure) to ensure scalability and easy management.
- **Database Hosting**: The MySQL database is hosted on dedicated database servers or cloud services for reliability.

Steps:

- Deploy the AMS on the production servers.
- Ensure database connections are correctly established.
- Perform a **live data migration** to transfer any pre-existing data from legacy systems.

4. Continuous Monitoring

Once AMS is deployed, continuous monitoring is essential to track system performance, server health, and any anomalies. Monitoring tools such as **Prometheus** or **Datadog** are used to track:

- **Server Uptime**: Ensuring the system is always available.
- **Performance Metrics**: Monitoring response times and transaction volumes.
- **Security Logs**: Checking for unauthorized access attempts and other potential vulnerabilities.

8.2 Installation and Configuration

Once the system is deployed, **installation and configuration** steps are necessary to make the system operational and accessible to users.

1. System Installation

- **Backend Installation**: Install Java, Servlets, and MySQL on the server.
- **Frontend Installation**: Deploy JavaFX interfaces on users' workstations.
- **Database Setup**: Initialize the MySQL database with all required tables and relationships.

2. Configuration Settings

- **System Configuration**: Configure server settings, including network ports, firewall rules, and security certificates.
- **User Configuration**: Set up different roles (e.g., admin, staff, passenger) and configure permissions for each role.
- **Backup Configuration**: Set up automated backup schedules for both data and the entire system.

8.3 User Manual

A **User Manual** is provided to ensure smooth adoption of the AMS. It includes detailed instructions on how to use each module and perform common tasks.

1. Administrator Guide

- **Flight Scheduling**: How to create, modify, and delete flight schedules.
- **Crew Assignment**: Steps for assigning crew members to flights based on qualifications and availability.
- **Maintenance Management**: How to schedule and log maintenance activities.

2. Staff Guide

- **Booking Management**: How to assist passengers with booking and managing their tickets.
- **Check-In Process**: Instructions for managing passenger check-ins and printing boarding passes.

3. Passenger Guide

- **Flight Booking**: How to search for flights, book tickets, and manage bookings.
- **Check-In and Boarding**: How to check in and get a boarding pass.

8.4 Backup and Recovery Plan

A comprehensive **Backup and Recovery Plan** ensures business continuity and data integrity in case of system failure or other disruptions.

1. Backup Strategy

- **Full Backups**: Scheduled daily to back up the entire system, including databases and application files.
- **Incremental Backups**: Hourly backups of data to reduce the amount of storage required.
- **Cloud Storage**: Backups are stored both on-site and in the cloud to ensure redundancy.

2. Disaster Recovery Plan

- **Failover Mechanism**: In case of server failure, traffic is rerouted to backup servers.
- **Recovery Time Objective (RTO)**: The system should be up and running within 1 hour of a failure.
- **Recovery Point Objective (RPO)**: The data loss threshold is set to no more than 1 hour.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

The **Aviation Management System (AMS)** successfully integrates the critical operations of an airline into a single, cohesive platform. The AMS addresses the key challenges in the aviation industry, such as inefficiencies in flight scheduling, crew management, maintenance tracking, and passenger services. By automating these processes, AMS improves operational efficiency, reduces costs, and enhances the passenger experience.

Key accomplishments of the AMS include:

- **Automation** of flight scheduling, crew assignments, and maintenance management.
- **Improved passenger experience** through an intuitive interface for booking, check-in, and flight status tracking.
- **Enhanced safety and compliance**, ensuring that maintenance schedules and crew certifications are tracked in real-time.

The system is scalable and modular, ensuring it can grow with the needs of airlines, airports, and aviation authorities.

9.2 Achievements

1. **Streamlined Operations**: AMS reduces the time required for flight scheduling, crew assignments, and maintenance tracking by automating these tasks.
2. **Real-Time Data Access**: The system provides real-time updates on flight status, ensuring passengers and staff have access to up-to-date information.
3. **Regulatory Compliance**: AMS ensures that aircraft maintenance schedules and crew qualifications are automatically tracked, minimizing the risk of safety violations.
4. **Customer Satisfaction**: Passengers benefit from an intuitive and seamless booking and check-in process, reducing waiting times and improving overall satisfaction.

9.3 Challenges Faced

Despite the successes, several challenges were encountered during the development and implementation of AMS:

1. **Integration with Legacy Systems**: Migrating data from legacy systems to the AMS database required significant effort, as older systems were not designed to integrate with modern software solutions.
2. **User Adoption**: Some airline staff were initially resistant to change, requiring comprehensive training sessions to ease the transition to the new system.
3. **Scalability**: Ensuring the system could handle a high volume of data and simultaneous users required careful performance testing and optimization.

9.4 Future Work

The AMS is designed to be a **scalable and flexible solution** that can be enhanced over time. Future work includes the following improvements:

1. **Artificial Intelligence (AI) for Predictive Maintenance**: Using machine learning algorithms, AMS can predict aircraft maintenance needs based on historical data and sensor data from aircraft, reducing unscheduled maintenance.
2. **Enhanced Data Analytics**: Implementing **data analytics** tools to provide insights into flight efficiency, customer preferences, and operational bottlenecks, allowing airlines to optimize operations further.
3. **Mobile Application**: Developing a **mobile app** to allow passengers to book flights, check-in, and receive notifications on their smartphones.
4. **Blockchain Technology**: Implementing **blockchain** for secure ticketing and transparent passenger data handling, reducing fraud and enhancing security.

9.5 Closing Remarks

The **Aviation Management System (AMS)** has proven to be an essential tool for managing modern airline operations. By addressing key challenges in the industry, AMS helps airlines optimize their processes, reduce costs, and enhance passenger satisfaction. As the aviation industry continues to evolve, the AMS will evolve alongside it, incorporating new technologies and providing even greater benefits to airlines and passengers.

References

1. **Books**:
    a. Schildt, Herbert. *Java: A Beginner's Guide*. McGraw-Hill Education, 209.
    b. Jones, Tim. *Database Systems: A Practical Approach to Design, Implementation, and Management*. Pearson Education, 2009.
2. **Articles**:
    a. Barkan, Samuel. "Aviation Management Software Solutions." *Journal of Aviation Technology and Engineering*, vol. 6, no. 2, 2023.
    b. Smith, Alan. "The Role of Software in Modern Airline Operations." *Aviation Weekly*, 2022.
3. **Websites**:
    a. MySQL Documentation: https://dev.mysql.com/doc/
    b. Oracle Java Documentation: https://docs.oracle.com/javase/
4. **API Documentation**:
    a. Payment Gateway API (e.g., Stripe or PayPal) documentation for integrating secure payment processing into the AMS.