AIRPORT MANAGEMENT SYSTEM

A MINI-PROJECT REPORT

Submitted by

SRI NIDHI K S	241001264
SURIYA PRAKASH S	241001279
SUBASRI A S	241001271

in partial fulfillment of the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"AIRPORT MANAGEMENT SYSTEM"** is the bonafide work of **"SRINIDHI K S, SURIYA PRAKASAH S , SUBASRI A S"** who carried out the project work under my supervision.

SIGNATURE

R.SUBASHREE

ASSISTANT PROFESSOR SG

Dept.of Information Technology,
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on

____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Airports handle thousands of passengers, flights, and operations daily, making efficient management essential for smooth functioning. The **Airport Management System** is a database-driven application developed to automate and streamline airport operations such as flight scheduling, passenger management, ticket booking, and staff handling. The system is designed using **Java Swing** for the graphical user interface and **JDBC** for database connectivity with **MySQL**, ensuring reliable and secure data management.

This project provides separate modules for administrators and users. Administrators can manage flights, passenger details, and schedules, while users can search for flights, book tickets, and view booking information. By digitalizing and centralizing airport operations, this system minimizes manual effort, reduces errors, and enhances overall efficiency. The **Airport Management System** thus serves as a comprehensive solution for effective airport data management and improved service delivery to passengers.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson **DR. M.THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head Of The Department **Dr.P.VALARMATHIE** and our Deputy Head Of The Department **Dr.M.BABU PEOFE** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide

Assistant Professor.R. SUBASHREE , for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

1.SRINIDHI K S

2. SURIYA PRAKASH S

3. SUBASRI A S

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The Airport Management System is a software application developed to efficiently manage and automate airport operations. The project helps in maintaining flight details, passenger records, staff management, and ticket bookings in a simplified and systematic manner. The system provides an interactive Graphical User Interface (GUI) built using Java, which makes it easier for administrators and passengers to access and update information. The backend operations are handled through Java and JDBC, ensuring secure and reliable database connectivity

1.2 SCOPE OF THE WORK

The Airport Management System plays a crucial role in managing daily airport activities such as flight scheduling, passenger management, and ticket reservations. It helps airport authorities streamline their workflow and reduces manual paperwork. The system provides functionalities for both administrators and users—where administrators can manage flights, update schedules, and view reports, while users can check flight availability, make bookings, and view ticket details. This system can be implemented in both small and large airports to improve operational efficiency and provide better service to travelers

1.3 PROBLEM STATEMENT

Airports handle thousands of passengers, flights, and operations daily, making manual management inefficient and error-prone. Traditional methods of maintaining flight details, staff schedules, and passenger data often lead to duplication, data loss, and miscommunication. There is a growing need for a digital platform that can manage all airport operations systematically. The **Airport Management System** aims to address these issues by providing a centralized, database-driven system for easy access, storage, and management of all airport-related data.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The main aim of this project is to design and develop a computerized Airport Management System that simplifies the management of airport activities. The specific objective of the project include:

1.To maintain records of flights, passengers, and staff efficiently.

2.To provide a user-friendly GUI for easy interaction between users and the system.

3.To automate ticket booking and flight scheduling processes.

4.To ensure secure data handling using Java and JDBC connectivity with a relational database.

5.To reduce manual effort and improve overall accuracy and productivity of airport operations.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel i5
Memory Size	:	8GB (Minimum)
HDD	:	1 TB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 10
Front – End	:	Java Swing
Back - End	:	Java with JDBC
Database	:	MySql

CHAPTER 3

JAVA SWING PROGRAM(GUI)

```
import javax.swing.*;

import javax.swing.table.DefaultTableModel;

import java.awt.*;

import java.awt.event.*;

import java.sql.*;


public class AirportGUI extends JFrame {

   private Connection conn;

   private JTable airportTable, customerTable;

   private DefaultTableModel airportModel, customerModel;


   // Airport fields

   private JTextField codeField, nameField, cityField, countryField,
timingField;
```

```
    // Customer fields
    private JTextField custNameField, custAirportIDField;


    public AirportGUI() {
        setTitle("✈ Airport Management System");
        setSize(900, 600);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        getContentPane().setBackground(new Color(240, 248, 255));


        // === Database Connection ===
        try {
            conn = DriverManager.getConnection(

"jdbc:mysql://localhost:3306/airportdb?allowPublicKeyRetrieval=true&
useSSL=false",
                "root",
                "root@123"
            );
            System.out.println("Connected to database!");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, "DB Connection Failed:
" + e.getMessage());
            return;
        }

        // === Tabs ===
        JTabbedPane tabs = new JTabbedPane();

        // --- Airports Panel ---
        JPanel airportPanel = new JPanel(new BorderLayout());

        JPanel airportInput = new JPanel(new GridLayout(2, 5, 10, 5));
        airportInput.setBorder(BorderFactory.createTitledBorder("Airport
Details"));
```

```
    codeField = new JTextField(); nameField = new JTextField();
cityField = new JTextField();

    countryField = new JTextField(); timingField = new JTextField();

    airportInput.add(new JLabel("Code:"));
airportInput.add(codeField);

    airportInput.add(new JLabel("Name:"));
airportInput.add(nameField);

    airportInput.add(new JLabel("City:")); airportInput.add(cityField);

    airportInput.add(new JLabel("Country:"));
airportInput.add(countryField);

    airportInput.add(new JLabel("Timing:"));
airportInput.add(timingField);



    airportModel = new DefaultTableModel(new
String[]{"ID","Code","Name","City","Country","Timing"},0);

    airportTable = new JTable(airportModel);

    JScrollPane airportScroll = new JScrollPane(airportTable);

    airportScroll.setBorder(BorderFactory.createTitledBorder("Airport
Records"));
```

```
        JPanel airportButtons = new JPanel(new FlowLayout());

        JButton viewAirportBtn = new JButton("View All"); JButton
addAirportBtn = new JButton("Add");

        JButton updateAirportBtn = new JButton("Update City"); JButton
deleteAirportBtn = new JButton("Delete");

        airportButtons.add(viewAirportBtn);
airportButtons.add(addAirportBtn);

        airportButtons.add(updateAirportBtn);
airportButtons.add(deleteAirportBtn);



        airportPanel.add(airportInput, BorderLayout.NORTH);

        airportPanel.add(airportScroll, BorderLayout.CENTER);

        airportPanel.add(airportButtons, BorderLayout.SOUTH);



        // --- Customers Panel ---

        JPanel customerPanel = new JPanel(new BorderLayout());



        JPanel custInput = new JPanel(new GridLayout(2,2,10,5));
        custInput.setBorder(BorderFactory.createTitledBorder("Customer
Details"));

        custNameField = new JTextField(); custAirportIDField = new
JTextField();

        custInput.add(new JLabel("Customer Name:"));
custInput.add(custNameField);

        custInput.add(new JLabel("Airport ID:"));
custInput.add(custAirportIDField);



        customerModel = new DefaultTableModel(new
String[]{"ID","Customer Name","Airport ID"},0);

        customerTable = new JTable(customerModel);

        JScrollPane custScroll = new JScrollPane(customerTable);

        custScroll.setBorder(BorderFactory.createTitledBorder("Customer
Records"));



        JPanel custButtons = new JPanel(new FlowLayout());

        JButton viewCustBtn = new JButton("View All"); JButton
addCustBtn = new JButton("Add");

        JButton viewCustDetailsBtn = new JButton("View Details");
        custButtons.add(viewCustBtn); custButtons.add(addCustBtn);
    custButtons.add(viewCustDetailsBtn);


        customerPanel.add(custInput, BorderLayout.NORTH);

        customerPanel.add(custScroll, BorderLayout.CENTER);

        customerPanel.add(custButtons, BorderLayout.SOUTH);


        // --- Add Tabs ---

        tabs.addTab("Airports", airportPanel);

        tabs.addTab("Customers", customerPanel);

        add(tabs, BorderLayout.CENTER);


        // === Button Actions ===

        viewAirportBtn.addActionListener(e -> loadAirports());

        addAirportBtn.addActionListener(e -> addAirport());

        updateAirportBtn.addActionListener(e -> updateAirport());

        deleteAirportBtn.addActionListener(e -> deleteAirport());
        viewCustBtn.addActionListener(e -> loadCustomers());
        addCustBtn.addActionListener(e -> addCustomer());
        viewCustDetailsBtn.addActionListener(e ->
viewCustomerDetails());

        loadAirports();
        loadCustomers();

        setVisible(true);
    }

    // --- Airport Methods ---
    private void loadAirports() {
        try {
            airportModel.setRowCount(0);
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM airports");
            while(rs.next()){
                airportModel.addRow(new Object[]{
                    rs.getInt("airport_id"), rs.getString("code"),
rs.getString("name"),
                    rs.getString("city"), rs.getString("country"),
rs.getString("timing")
                });
            }
        } catch(Exception e){ JOptionPane.showMessageDialog(this,
e.getMessage()); }
    }

    private void addAirport() {
        try {
            PreparedStatement ps = conn.prepareStatement(
                "INSERT INTO airports (code,name,city,country,timing)
VALUES (?,?,?,?,?)");
```

```
            ps.setString(1, codeField.getText());
            ps.setString(2, nameField.getText());
            ps.setString(3, cityField.getText());
            ps.setString(4, countryField.getText());
            ps.setString(5, timingField.getText());
            ps.executeUpdate();
            JOptionPane.showMessageDialog(this, "Airport Added!");
            loadAirports();
        } catch(Exception e){ JOptionPane.showMessageDialog(this,
e.getMessage()); }
    }


    private void updateAirport() {
        try {
            PreparedStatement ps = conn.prepareStatement(
                "UPDATE airports SET city=? WHERE code=?");
            ps.setString(1, cityField.getText());
            ps.setString(2, codeField.getText());

            int rows = ps.executeUpdate();

            if(rows>0) JOptionPane.showMessageDialog(this,"City
Updated!");

            else JOptionPane.showMessageDialog(this,"Airport not found!");

            loadAirports();

        } catch(Exception e){ JOptionPane.showMessageDialog(this,
e.getMessage()); }

    }



    private void deleteAirport() {

        try {

            PreparedStatement ps = conn.prepareStatement("DELETE
FROM airports WHERE code=?");

            ps.setString(1, codeField.getText());

            int rows = ps.executeUpdate();

            if(rows>0) JOptionPane.showMessageDialog(this,"Airport
Deleted!");
```

```
            else JOptionPane.showMessageDialog(this,"No airport found!");
            loadAirports();
        } catch(Exception e){ JOptionPane.showMessageDialog(this,
e.getMessage()); }
    }


    // --- Customer Methods ---
    private void loadCustomers() {
        try {
            customerModel.setRowCount(0);
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM
customers");
            while(rs.next()){
                customerModel.addRow(new Object[]{
                    rs.getInt("customer_id"), rs.getString("customer_name"),
rs.getInt("airport_id")
                });
            }

        } catch(Exception e){ JOptionPane.showMessageDialog(this,
e.getMessage()); }

    }



    private void addCustomer() {

        try {

            PreparedStatement ps = conn.prepareStatement(

                "INSERT INTO customers(customer_name,airport_id)
VALUES (?,?)");

            ps.setString(1, custNameField.getText());

            ps.setInt(2, Integer.parseInt(custAirportIDField.getText()));

            ps.executeUpdate();

            JOptionPane.showMessageDialog(this,"Customer Added!");

            loadCustomers();

        } catch(Exception e){ JOptionPane.showMessageDialog(this,
e.getMessage()); }

    }
    private void viewCustomerDetails() {
        try {
            String name = custNameField.getText();
            PreparedStatement ps = conn.prepareStatement(
                "SELECT c.customer_name, a.name, a.city, a.timing " +
                "FROM customers c JOIN airports a ON c.airport_id =
a.airport_id " +
                "WHERE c.customer_name=?"
            );
            ps.setString(1, name);
            ResultSet rs = ps.executeQuery();
            if(rs.next()) {
                JOptionPane.showMessageDialog(this, "Customer: " +
rs.getString(1) +
                        "\nAirport: " + rs.getString(2) +
                        "\nCity: " + rs.getString(3) +
                        "\nTiming: " + rs.getString(4));
            } else {
                JOptionPane.showMessageDialog(this,"No customer found
with that name.");
            }
        } catch(Exception
e){ JOptionPane.showMessageDialog(this,e.getMessage()); }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(AirportGUI::new);
    }
}
```

PROGRAM CONNECTING WITH DATABASE

```
import java.sql.*;
import java.util.Scanner;

public class AirportDB {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        String url = "jdbc:mysql://localhost:3306/airportdb";
        String user = "root";
        String pass = "root@123";


        try (Connection con = DriverManager.getConnection(url, user,
pass)) {


            while (true) {
                System.out.println("\n=== Airport Database Menu ===");
                System.out.println("1. View all airports");
                System.out.println("2. Add a new airport");
                System.out.println("3. Update airport city");
                System.out.println("4. Delete an airport");
                System.out.println("5. Add new customer");
System.out.println("6. View customer flight details");

System.out.println("7. View all customers");

System.out.println("8. Exit");

System.out.print("Enter your choice: ");

int choice = sc.nextInt();

sc.nextLine();



switch (choice) {



    case 1:

        viewAirports(con);

        break;



    case 2:

        addAirport(con, sc);

        break;
```

```
case 3:

    updateAirportCity(con, sc);

    break;


case 4:

    deleteAirport(con, sc);

    break;


case 5:

    addCustomer(con, sc);

    break;


case 6:

    viewCustomerDetails(con, sc);

    break;


case 7:
```

```
                    viewAllCustomers(con);
                    break;

                case 8:
                    System.out.println("Goodbye!");
                    return;

                default:
                    System.out.println("Invalid choice!");
            }
        }

    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

```
static void viewAirports(Connection con) throws Exception {
    String q = "SELECT * FROM airports";
    PreparedStatement ps = con.prepareStatement(q);
    ResultSet rs = ps.executeQuery();


    System.out.println("\nID | Code | Name | City | Country | Timing");


    while (rs.next()) {
        System.out.println(
            rs.getInt("airport_id") + " | " +
            rs.getString("code") + " | " +
            rs.getString("name") + " | " +
            rs.getString("city") + " | " +
            rs.getString("country") + " | " +
            rs.getString("timing")
        );
    }
    }


    static void addAirport(Connection con, Scanner sc) throws Exception
{
        System.out.print("Code: ");
        String code = sc.nextLine();
        System.out.print("Name: ");
        String name = sc.nextLine();
        System.out.print("City: ");
        String city = sc.nextLine();
        System.out.print("Country: ");
        String country = sc.nextLine();
        System.out.print("Timing: ");
        String timing = sc.nextLine();


        String q = "INSERT INTO airports(code,name,city,country,timing)
VALUES(?,?,?,?,?)";
        PreparedStatement ps = con.prepareStatement(q);
        ps.setString(1, code);

        ps.setString(2, name);

        ps.setString(3, city);

        ps.setString(4, country);

        ps.setString(5, timing);



        ps.executeUpdate();

        System.out.println("Airport added!");

    }



    static void updateAirportCity(Connection con, Scanner sc) throws
Exception {

        System.out.print("Enter Airport ID to update: ");

        int id = sc.nextInt();

        sc.nextLine();



        System.out.print("Enter new city: ");
        String newCity = sc.nextLine();


        String q = "UPDATE airports SET city=? WHERE airport_id=?";

        PreparedStatement ps = con.prepareStatement(q);

        ps.setString(1, newCity);

        ps.setInt(2, id);


        int rows = ps.executeUpdate();

        if (rows > 0) {

            System.out.println("City updated!");

        } else {

            System.out.println("No airport found with that ID.");

        }

    }


    static void deleteAirport(Connection con, Scanner sc) throws
Exception {
        System.out.print("Enter Airport ID to delete: ");
        int id = sc.nextInt();


        String q = "DELETE FROM airports WHERE airport_id=?";
        PreparedStatement ps = con.prepareStatement(q);
        ps.setInt(1, id);


        int rows = ps.executeUpdate();
        if (rows > 0) {
            System.out.println("Airport deleted!");
        } else {
            System.out.println("No airport found with that ID.");
        }
    }


    static void addCustomer(Connection con, Scanner sc) throws
Exception {
        System.out.print("Enter customer name: ");

        String name = sc.nextLine();



        System.out.print("Enter airport ID: ");

        int aid = sc.nextInt();

        sc.nextLine();



        // Check if airport exists

        String check = "SELECT 1 FROM airports WHERE airport_id=?";

        PreparedStatement psCheck = con.prepareStatement(check);

        psCheck.setInt(1, aid);

        ResultSet rs = psCheck.executeQuery();

        if (!rs.next()) {

            System.out.println("Airport ID does not exist! Cannot add
customer.");

            return;

        }
        String q = "INSERT INTO customers(customer_name, airport_id)
VALUES (?, ?)";
        PreparedStatement ps = con.prepareStatement(q);
        ps.setString(1, name);
        ps.setInt(2, aid);


        ps.executeUpdate();
        System.out.println("Customer added!");
    }


    static void viewCustomerDetails(Connection con, Scanner sc) throws
Exception {
        System.out.print("Enter customer name to view details: ");
        String name = sc.nextLine();


        String q = "SELECT c.customer_name, a.name, a.city, a.timing " +
                "FROM customers c JOIN airports a ON c.airport_id =
a.airport_id " +
                "WHERE c.customer_name = ?";

        PreparedStatement ps = con.prepareStatement(q);
        ps.setString(1, name);

        ResultSet rs = ps.executeQuery();

        System.out.println("\nCustomer | Airport | City | Timing");

        if (rs.next()) {
            System.out.println(
                rs.getString(1) + " | " +
                rs.getString(2) + " | " +
                rs.getString(3) + " | " +
                rs.getString(4)
            );

        } else {

            System.out.println("No customer found with that name.");

        }

    }


    static void viewAllCustomers(Connection con) throws Exception {

        String q = "SELECT c.customer_name, a.name, a.city, a.timing " +

                   "FROM customers c JOIN airports a ON c.airport_id =
a.airport_id";


        PreparedStatement ps = con.prepareStatement(q);

        ResultSet rs = ps.executeQuery();


        System.out.println("\nCustomer | Airport | City | Timing");


        boolean found = false;
        while (rs.next()) {
            found = true;
            System.out.println(
                rs.getString(1) + " | " +
                rs.getString(2) + " | " +
                rs.getString(3) + " | " +
                rs.getString(4)
            );
        }

        if (!found) {
            System.out.println("No customers found.");
        }
    }
}
```

SQL QUERY

```
USE airportdb;

ALTER TABLE airport

ADD COLUMN timing VARCHAR(50);

CREATE TABLE IF NOT EXISTS customer (

  customer_id INT AUTO_INCREMENT PRIMARY KEY,

  name VARCHAR(100) NOT NULL,

  flight_code VARCHAR(10),

  FOREIGN KEY (flight_code) REFERENCES airport(code)

);

ALTER TABLE airport

ADD COLUMN timing VARCHAR(50);

DESC airport;

ALTER TABLE airport DROP COLUMN timings;

DESC airport;

CREATE TABLE customer (

    customer_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),

    flight_code VARCHAR(10),

    FOREIGN KEY (flight_code) REFERENCES airport(code)

);

DESC customer;

SELECT code FROM airport;

SELECT * FROM airport;

UPDATE airport SET timing='06:30 AM' WHERE airport_id=1;

UPDATE airport SET timing='09:45 AM' WHERE airport_id=2;

UPDATE airport SET timing='01:20 PM' WHERE airport_id=3;

UPDATE airport SET timing='07:15 PM' WHERE airport_id=4;

SELECT airport_id, code, timing FROM airport;

SELECT * FROM airport;

UPDATE airport SET timing='06:30 AM' WHERE airport_id=1;

UPDATE airport SET timing='09:45 AM' WHERE airport_id=2;

UPDATE airport SET timing='01:20 PM' WHERE airport_id=3;

UPDATE airport SET timing='07:15 PM' WHERE airport_id=4;
```

```
DESCRIBE customer;

SELECT * FROM airport;

UPDATE airport SET timing = '06:00' WHERE timing IS NULL;

DESCRIBE airport;

DESCRIBE airport;

SELECT * FROM airport;

SET SQL_SAFE_UPDATES = 0;

UPDATE airport SET timing = '06:30 AM' WHERE code = 'DEL';

UPDATE airport SET timing = '09:15 AM' WHERE code = 'Indigo';

UPDATE airport SET timing = '11:45 AM' WHERE code = 'AirArabia';

UPDATE airport SET timing = '03:00 PM' WHERE code = '345';

SELECT * FROM airport;

DROP TABLE customer;

CREATE TABLE customer (

    customer_id INT AUTO_INCREMENT PRIMARY KEY,

    name VARCHAR(100) NOT NULL,

    flight_code VARCHAR(10),
   FOREIGN KEY (flight_code) REFERENCES airport(code)

);

INSERT INTO customer (name, flight_code)

VALUES ('Srinidhi', 'DEL');

INSERT INTO customer (name, flight_code)

VALUES ('Rahul', '345');

SELECT * FROM customer;

SELECT * FROM customer;

UPDATE customer

SET flight_code = 'DEL'

WHERE name = 'Rahul';


INSERT INTO airport (code, name, city, country, timing)

VALUES ('MAA', 'Chennai Airport', 'Chennai', 'India', '10:00 AM');

DESCRIBE airport;

DESCRIBE customer;

SELECT * FROM customer;
```

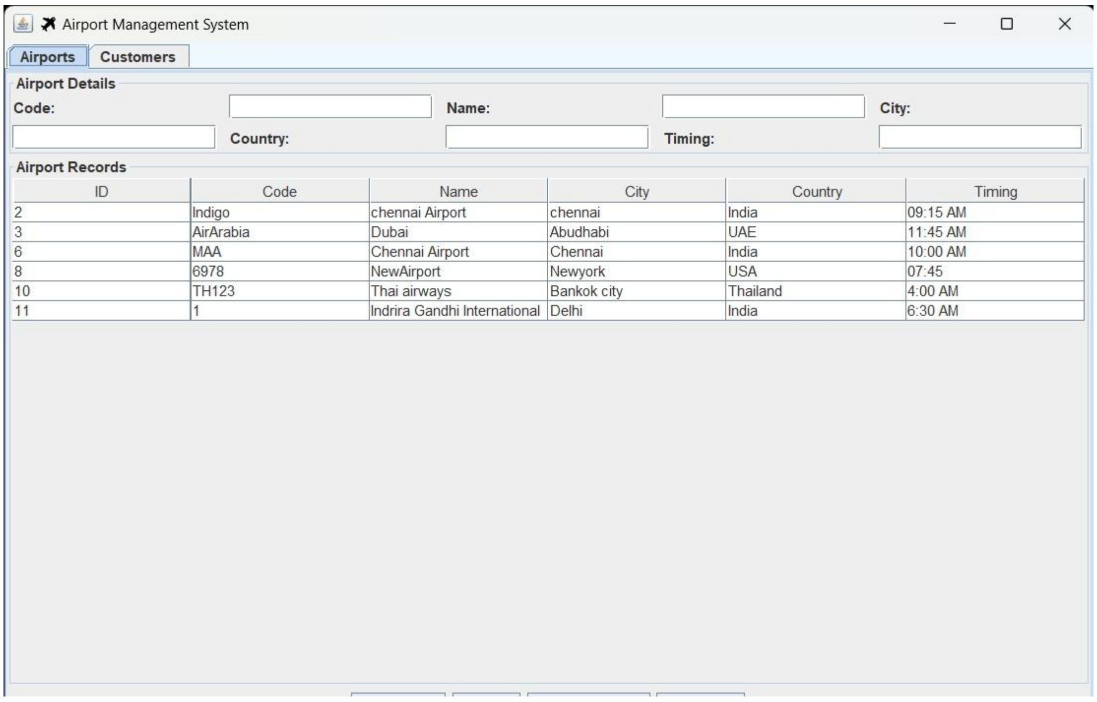
```
DROP TABLE IF EXISTS customers;

DROP TABLE IF EXISTS airports;

CREATE TABLE airports (

   airport_id INT PRIMARY KEY AUTO_INCREMENT,

   code VARCHAR(10),

   name VARCHAR(100),

   city VARCHAR(100),

   country VARCHAR(100),

   timing VARCHAR(20)

);

CREATE TABLE customers (

   customer_id INT PRIMARY KEY AUTO_INCREMENT,

   customer_name VARCHAR(100),

   airport_id INT,

   FOREIGN KEY (airport_id) REFERENCES airports(airport_id)

);

INSERT INTO airports(code, name, city, country, timing) VALUES
```

```
('DEL', 'Indira Gandhi International', 'New Delhi', 'India', '06:30 AM'),

('Indigo', 'chennai Airport', 'chennai', 'India', '09:15 AM'),

('AirArabia', 'Dubai', 'Abudhabi', 'UAE', '11:45 AM'),

('345', 'Nippon airlines', 'Tokyo', 'japan', '03:00 PM'),

('6978', 'hari', 'dolakpur', 'choota bheem', '05:00'),

('MAA', 'Chennai Airport', 'Chennai', 'India', '10:00 AM'),

('manipur', 'yashname', 'nicobar', 'india', '03:20');


SELECT airport_id, code, name FROM airports;

SELECT * FROM airports;

ALTER TABLE customers

DROP FOREIGN KEY customers_ibfk_1;


ALTER TABLE customers

ADD CONSTRAINT customers_ibfk_1

FOREIGN KEY (airport_id) REFERENCES airports(airport_id)

ON DELETE CASCADE;
```

```
SHOW CREATE TABLE customers;

SHOW CREATE TABLE customers;

ALTER TABLE customers

DROP FOREIGN KEY customers_ibfk_1;


ALTER TABLE customers

ADD CONSTRAINT customers_ibfk_1

FOREIGN KEY (airport_id) REFERENCES airports(airport_id)

ON DELETE CASCADE;


ALTER TABLE customers

DROP FOREIGN KEY customers_ibfk_1;

ALTER TABLE customers

ADD CONSTRAINT customers_ibfk_1

FOREIGN KEY (airport_id) REFERENCES airports(airport_id)

ON DELETE CASCADE;

SHOW DATABASES;
```

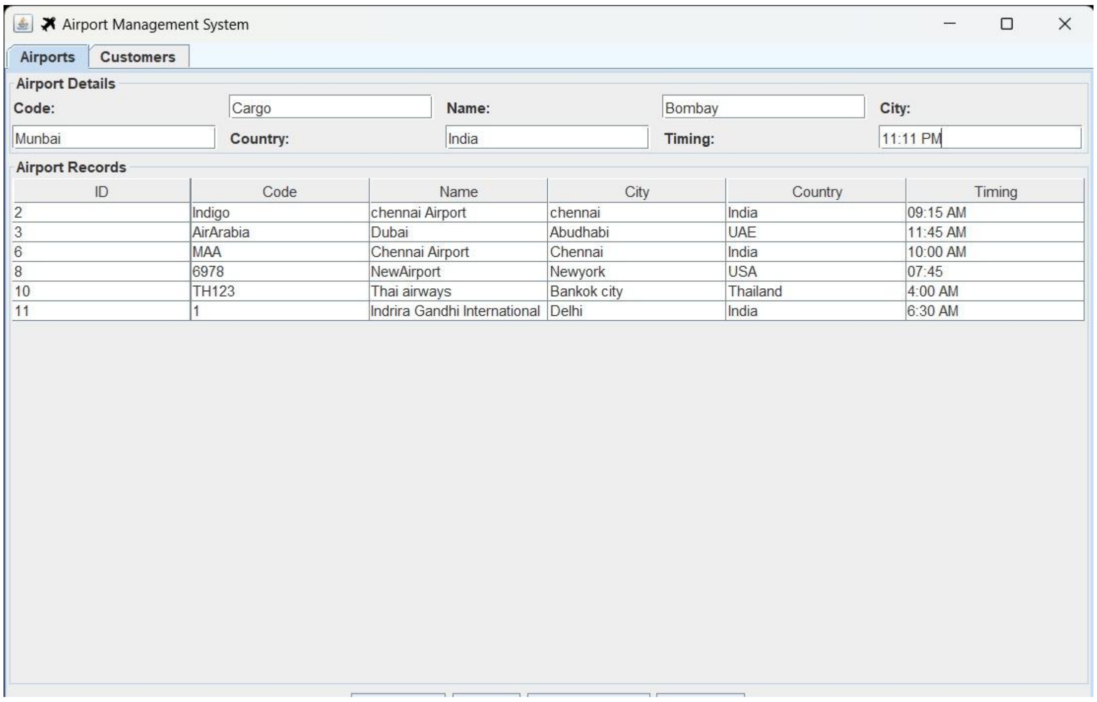
CHAPTER 4

SCREEN SHOTS



ID	Code	Name	City	Country	Timing
2	Indigo	chennai Airport	chennai	India	09:15 AM
3	AirArabia	Dubai	Abudhabi	UAE	11:45 AM
6	MAA	Chennai Airport	Chennai	India	10:00 AM
8	6978	NewAirport	Newyork	USA	07:45
10	TH123	Thai airways	Bankok city	Thailand	4:00 AM
11	1	Indrira Gandhi International	Delhi	India	6:30 AM

Fig 5.1 Introduction page


Airport Management System

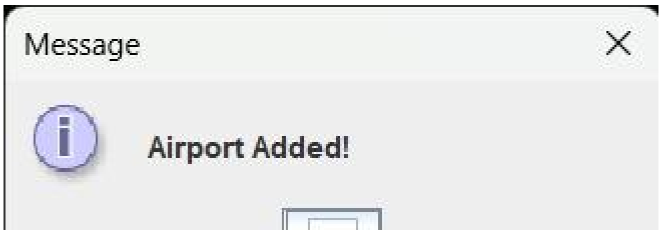
Airport Details

Code:	Cargo	Name:	Bombay	City:	
Munbai	Country:	India	Timing:	11:11 PM	

Airport Records

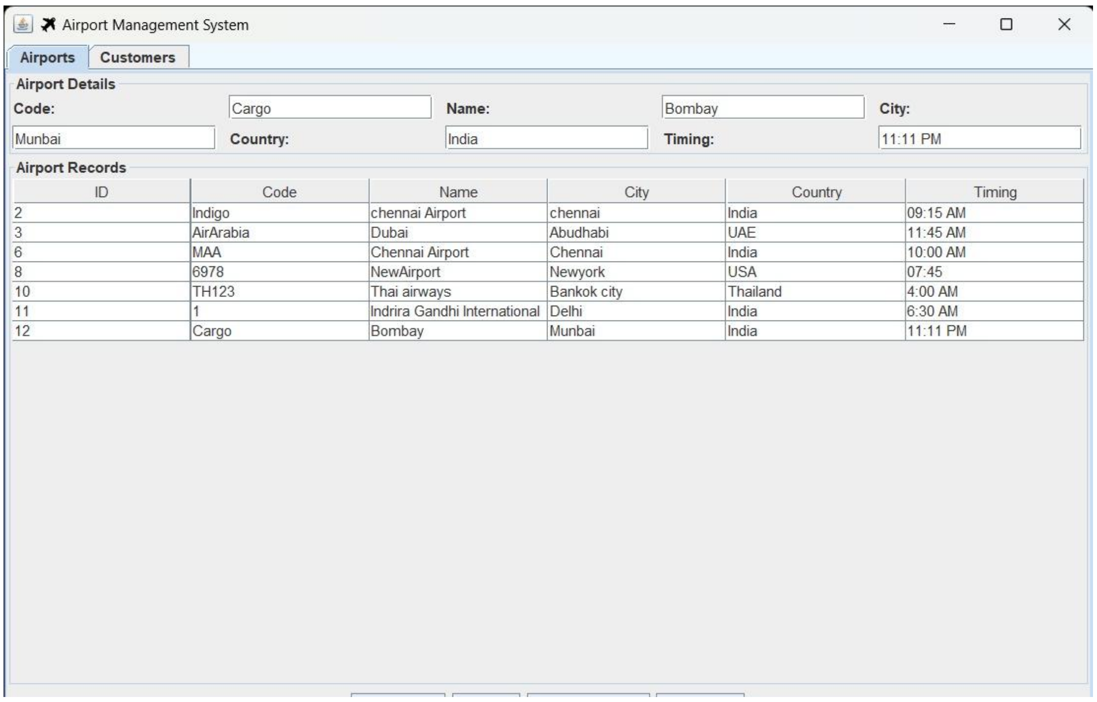
ID	Code	Name	City	Country	Timing
2	Indigo	chennai Airport	chennai	India	09:15 AM
3	AirArabia	Dubai	Abudhabi	UAE	11:45 AM
6	MAA	Chennai Airport	Chennai	India	10:00 AM
8	6978	NewAirport	Newyork	USA	07:45
10	TH123	Thai airways	Bankok city	Thailand	4:00 AM
11	1	Indrira Gandhi International	Delhi	India	6:30 AM

Fig 5.2 Passenger details


Message

Airport Added!

Fig 5.3 Passenger added successfully



Airport Management System

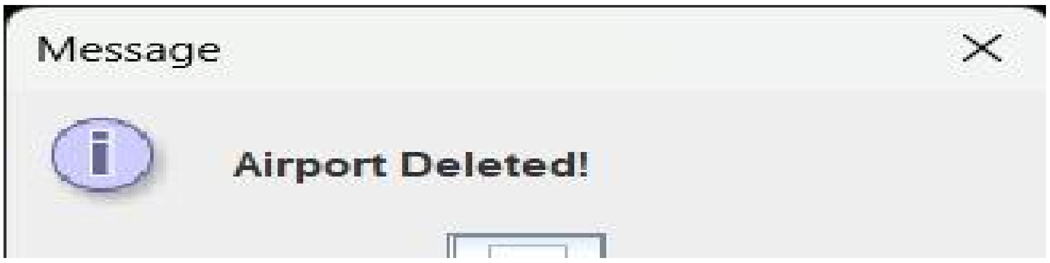
Airport Details

Code:	Cargo	Name:	Bombay	City:	Munbai
Country:	India	Timing:	11:11 PM		

Airport Records

ID	Code	Name	City	Country	Timing
2	Indigo	chennai Airport	chennai	India	09:15 AM
3	AirArabia	Dubai	Abudhabi	UAE	11:45 AM
6	MAA	Chennai Airport	Chennai	India	10:00 AM
8	6978	NewAirport	Newyork	USA	07:45
10	TH123	Thai airways	Bankok city	Thailand	4:00 AM
11	1	Indrira Gandhi International	Delhi	India	6:30 AM
12	Cargo	Bombay	Munbai	India	11:11 PM

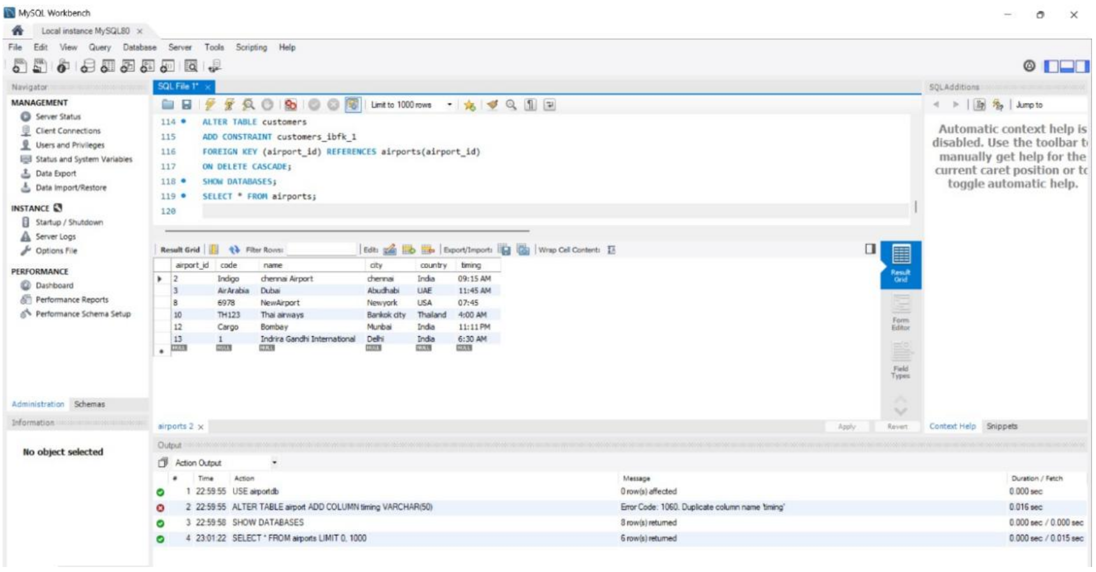
Fig 5.4 Booking creation



Message

Airport Deleted!

Fig 5.5 Deletion of Booking

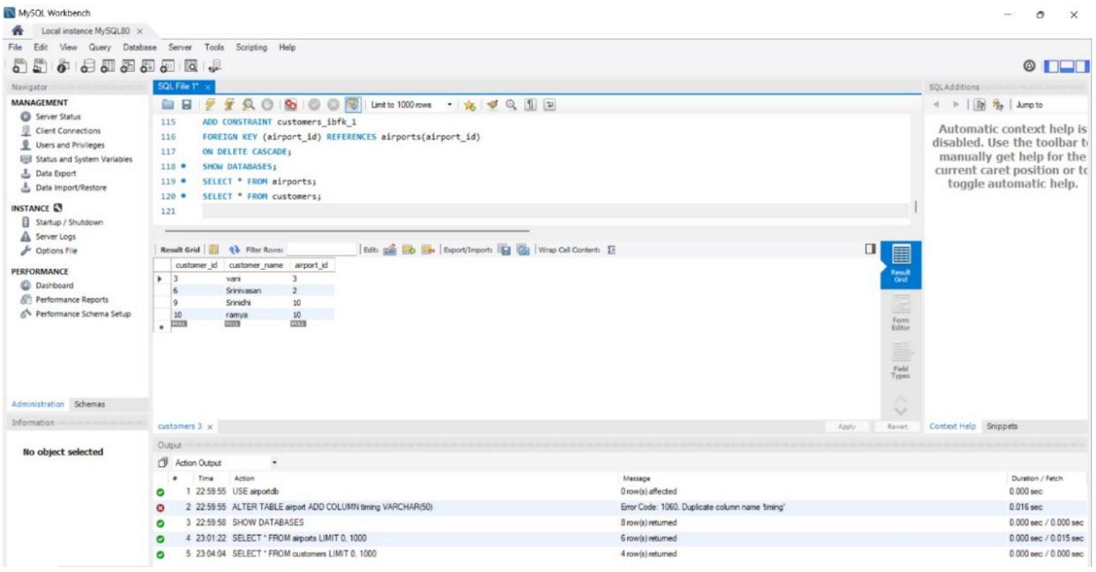


```sql
ALTER TABLE customers
ADD CONSTRAINT customers_ibfk_1
FOREIGN KEY (airport_id) REFERENCES airports(airport_id)
ON DELETE CASCADE;
SHOW DATABASES;
SELECT * FROM airports;
```

airport_id	code	name	city	country	timing
2	Indigo	chennai Airport	chennai	India	09:15 AM
3	AirArabia	Dubai	Abudhabi	UAE	11:45 AM
8	6978	NewAirport	Newyork	USA	07:45
10	TH123	Thai airways	Bankok city	Thailand	4:00 AM
12	Cargo	Bombay	Munbai	India	11:11 PM
13	1	Indira Gandhi International	Delhi	India	6:30 AM
NULL	NULL	NULL	NULL	NULL	NULL

```
#  Time      Action                                           Message                                     Duration / Fetch
1  22:59:55  USE airportdb                                    0 row(s) affected                           0.000 sec
2  22:59:55  ALTER TABLE airport ADD COLUMN timing VARCHAR(50) Error Code: 1060. Duplicate column name 'timing' 0.016 sec
3  22:59:58  SHOW DATABASES                                   8 row(s) returned                           0.000 sec / 0.000 sec
4  23:01:22  SELECT * FROM airports LIMIT 0, 1000             6 row(s) returned                           0.000 sec / 0.015 sec
```

Fig 5.6.1. Table 1-airports



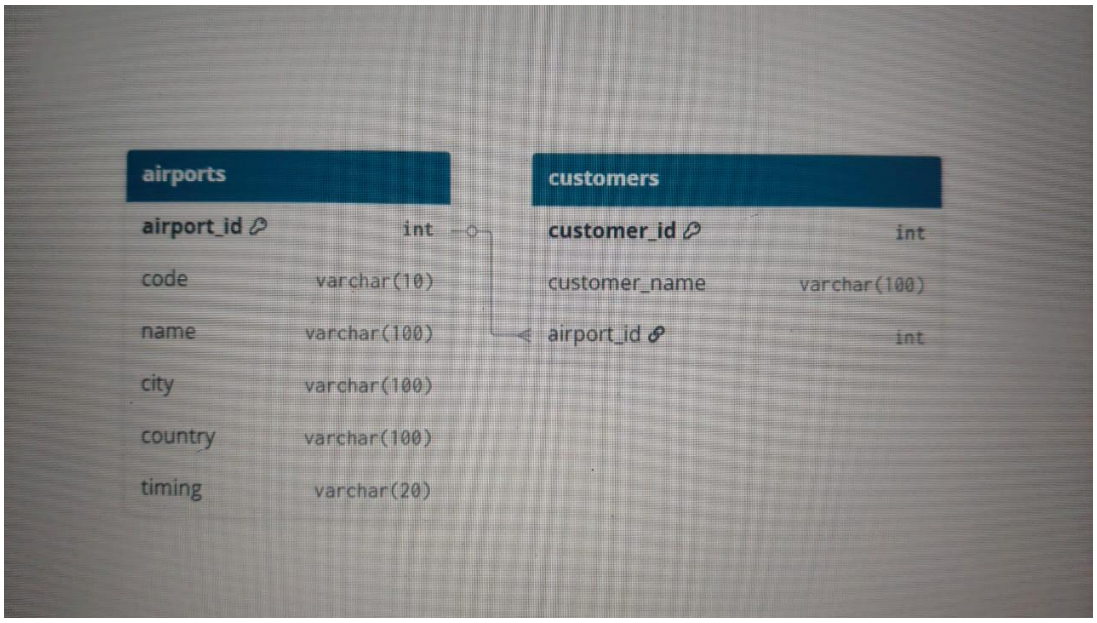
```sql
ADD CONSTRAINT customers_ibfk_1
FOREIGN KEY (airport_id) REFERENCES airports(airport_id)
ON DELETE CASCADE;
SHOW DATABASES;
SELECT * FROM airports;
SELECT * FROM customers;
```

customer_id	customer_name	airport_id
3	vani	3
6	Srinivasan	2
9	Srinidhi	10
10	ramya	10
NULL	NULL	NULL

```
#  Time      Action                                           Message                                     Duration / Fetch
1  22:59:55  USE airportdb                                    0 row(s) affected                           0.000 sec
2  22:59:55  ALTER TABLE airport ADD COLUMN timing VARCHAR(50) Error Code: 1060. Duplicate column name 'timing' 0.016 sec
3  22:59:58  SHOW DATABASES                                   8 row(s) returned                           0.000 sec / 0.000 sec
4  23:01:22  SELECT * FROM airports LIMIT 0, 1000             6 row(s) returned                           0.000 sec / 0.015 sec
5  23:04:04  SELECT * FROM customers LIMIT 0, 1000            4 row(s) returned                           0.000 sec / 0.000 sec
```

Fig 5.6.2.Table 2-customers

Fig 5.6 Database creation



airports	
airport_id	int
code	varchar(10)
name	varchar(100)
city	varchar(100)
country	varchar(100)
timing	varchar(20)

customers	
customer_id	int
customer_name	varchar(100)
airport_id	int

ER DIAGRAM

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENT

The Airport Management System developed in this project provides an efficient and user-friendly platform for managing various airport operations. With this system, users can easily access information related to flight schedules, passenger details, booking records, and other essential airport services. The organized management of data ensures smooth handling of day-to-day operations, reduces manual work, and minimizes errors. By offering clear visibility of available resources and service statuses, the system enhances coordination among airport staff and improves overall operational efficiency. Thus, the Airport Management System plays a vital role in improving airport service quality and ensuring a better experience for passengers and administrators.

In the future, the system can be enhanced with additional features to further improve its effectiveness. Advanced technologies such as automated check-in systems, real-time flight tracking, intelligent notifications, and enhanced security modules can be integrated. Incorporating data analytics and AI-based decision support systems can help predict flight delays, manage passenger flow, and optimize resource allocation. Integration with mobile applications can enable passengers to access flight information, receive alerts, and manage bookings more conveniently. With these improvements, the Airport Management System can provide a more seamless, reliable, and technologically advanced solution for airport operations.

REFERENCES

1. https://www.geeksforgeeks.org/java-swing/
2. https://docs.oracle.com/javase/tutorial/uiswing/
3. https://docs.oracle.com/javase/tutorial/jdbc/basics/
4. https://dev.mysql.com/doc/refman/8.0/en/
5. https://www.w3schools.com/sql/