**Database Development – CW2**

**Module Code: 7CS082**

**Submitted By: 100614702**

Table of Contents

[1. **Introduction** 2](#_Toc125624682)

[2. **Improvement done on database based on feedbacks on Assessment1** 2](#_Toc125624683)

[3. **Development and Operation Environment** 2](#_Toc125624684)

[4. **System Design & Implementation** 2](#_Toc125624685)

[4.1. Database Connection 2](#_Toc125624686)

[4.2. Entity Relationship Diagram 3](#_Toc125624687)

[4.3. Navigation Map of the System 4](#_Toc125624688)

[4.4. User Interface Design 4](#_Toc125624689)

[5. **Description of the system** 19](#_Toc125624690)

[6. **Report Generation** 23](#_Toc125624691)

[7. **Conclusion** 30](#_Toc125624692)

[8. **Reference** 30](#_Toc125624693)

# **Introduction**

The Student Flight Management system is a tool created for the airline industry to improve efficiency and increase revenue, assist managers and employees in decision making, and provide organization and easy access to information. Project’s well-designed database is crucial for any company, and it is user-friendly for all customers. The report's initial section details the development and implementation of the system using MySQL as the backend database.

The reason for selecting a Windows Forms Application (.NET Framework) is that it comes with pre-constructed, automatically created code for design. This makes it easier to design the Window pages and provides more time to focus on the appearance rather than the structure of the page. Microsoft offers all the necessary tools for design, like checkboxes, textboxes, drop-down menus, and radio buttons, which makes the programming process efficient and quick.

This report will detail how to connect a Windows Forms Application with MySQL, improvement done on database based on feedback given on Assessment-1, Development and Operation Environment, Entity Relationship Diagram, Navigation map of the system, user interface design, the important features of the application, and the stored procedures that have been implemented in the system's database.

# **Improvement done on database based on feedbacks on Assessment1**

* The 'Employee\_position' table is introduced to the database to differentiate pilots from other employees working on airplanes. This allows for the calculation of working hours for pilots and the assignment of pilots to their airplanes based on type rating. The table contains the columns 'Id', 'Position', and 'Salary' for employees.
* A 'Stop' table has been added to the database to store the number of stops for a scheduled flight for passengers. The table contains the columns 'Id', 'Arrival', 'Departure', 'Airport\_Id', and 'Flight\_Id'. The 'Arrival' and 'Departure' columns store the datetime of the airplane, and the 'Airport\_Id' and 'Flight\_Id' columns are used to link the 'Stop' table with the 'Flight' and 'Airport' tables.

# **Development and Operation Environment**

**Operating System**: Windows 10 (64 bits)

**CPU :** 2.4 GHz Intel Core i5

**RAM :** 8GB

**Visual Studio:** This project uses visual studio for implementation of applications.

**MYSQL:** The airline system uses MySQL as its database to manage and store data. NoSQL, on the other hand, is a non-relational database that is better suited for handling large amounts of unstructured data. Despite this, the MySQL database is sufficient for the system described in the article.

# **System Design & Implementation**

## 4.1. Database Connection

To establish a link to a current MySQL database:

1. To open the MySQL Connections Manager window, click on the icon in the Server Explorer menu bar.
2. To create a new connection, select "Add New Connection" option.
3. Enter a unique name for the connection in the designated "Connection Name" field.
4. Ensure that "TCP/IP (standard)" is selected as the connection method.
5. In the Parameters tab, input or adjust the following details:

* Hostname and Port, which should be the hostname or IP address and port number of the computer hosting the MySQL server, for instance, localhost if the server is installed on the same machine. The standard port value is 3306.
* The Username should be a valid MySQL user account name and the Password should be the corresponding password for the account.
* A default schema name is needed to open the connection, pick a name from the list.

1. To check the connection information, press the Test Connection button.
2. To create and save the new connection, press OK. The new connection will now be visible in the MySQL Connections Manager. (MYSQL Documentation)

## Entity Relationship Diagram

Revised database following the implementation of recommended modifications.

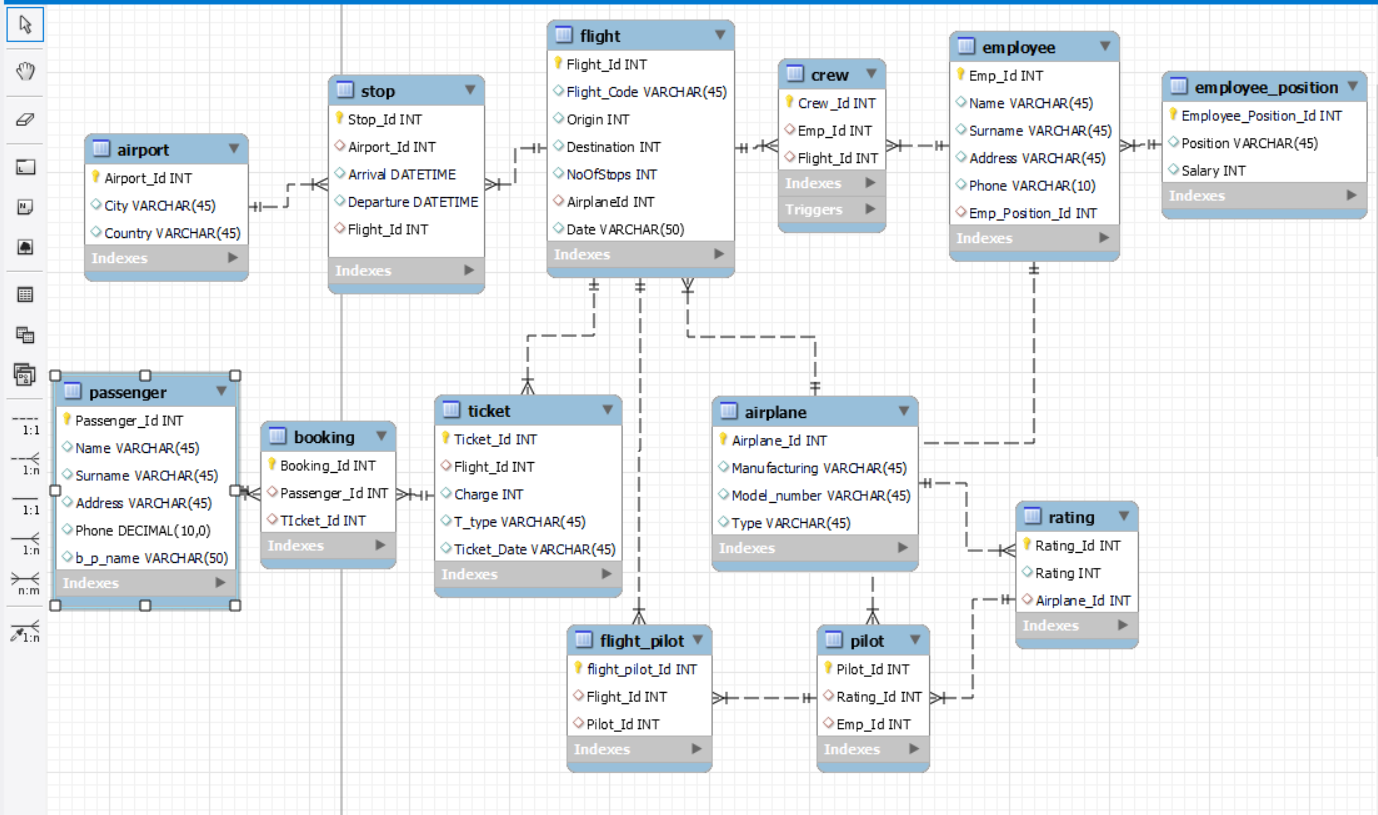


Fig 1: ER Diagram

## Navigation Map of the System

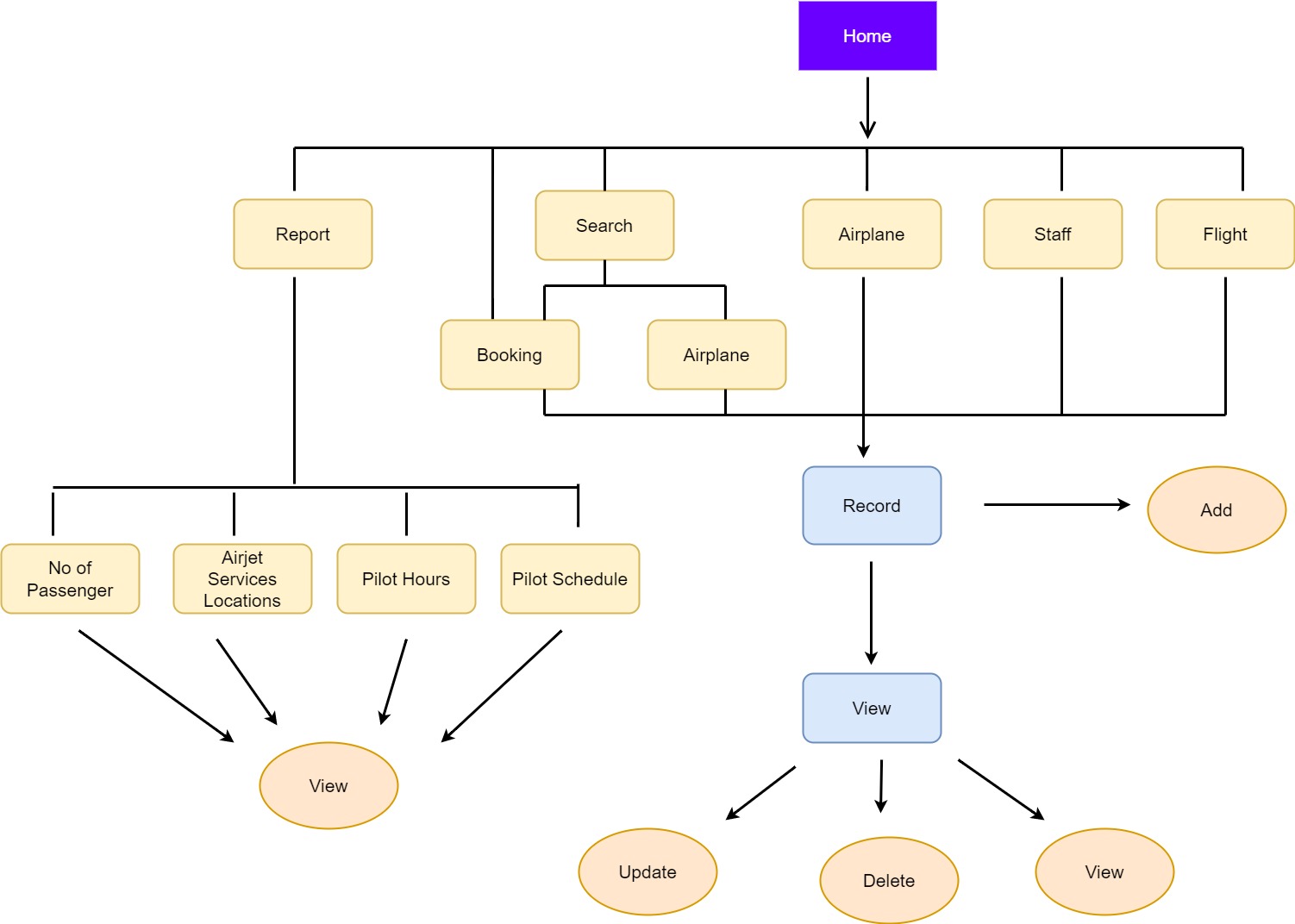


Fig 2: Navigation Map

## User Interface Design

In Windows Forms, a form that is connected to a database can directly perform various operations on the data through a query. The forms have been created to be user-friendly and offer customizable options through a variety of features to create a form.

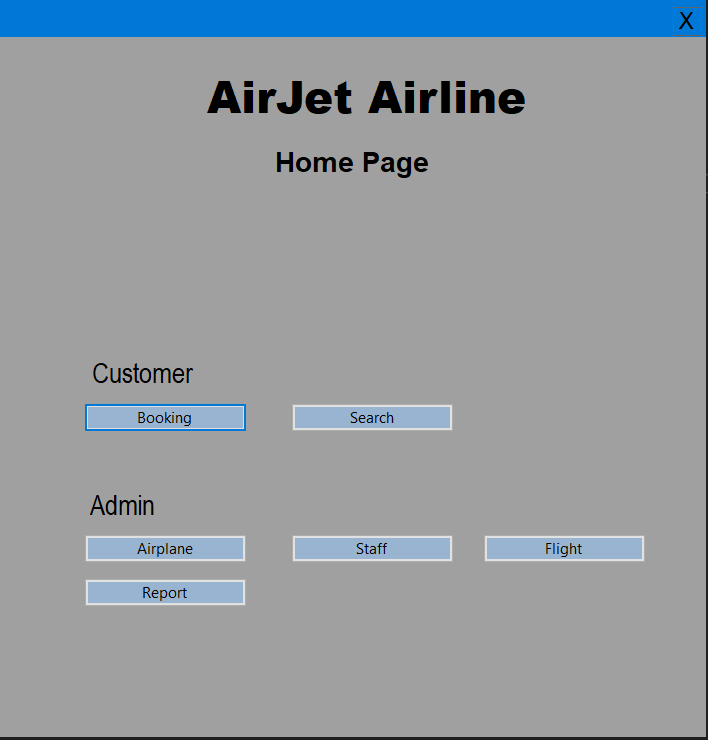
This application consists of 15 different forms that can be navigated between. The figures below depict the user interface design of the Airline Management System.

Fig 3: Home Page

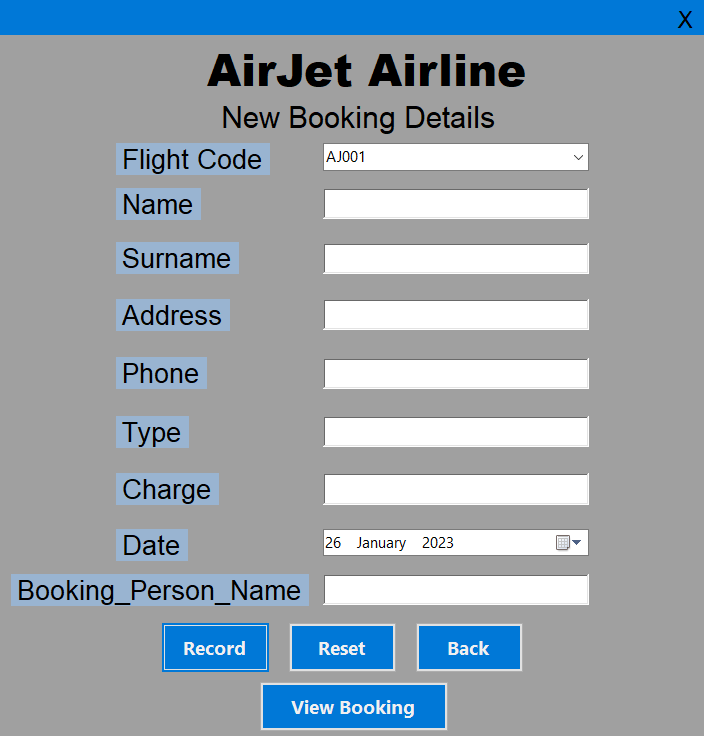


Fig 4: New Booking Page

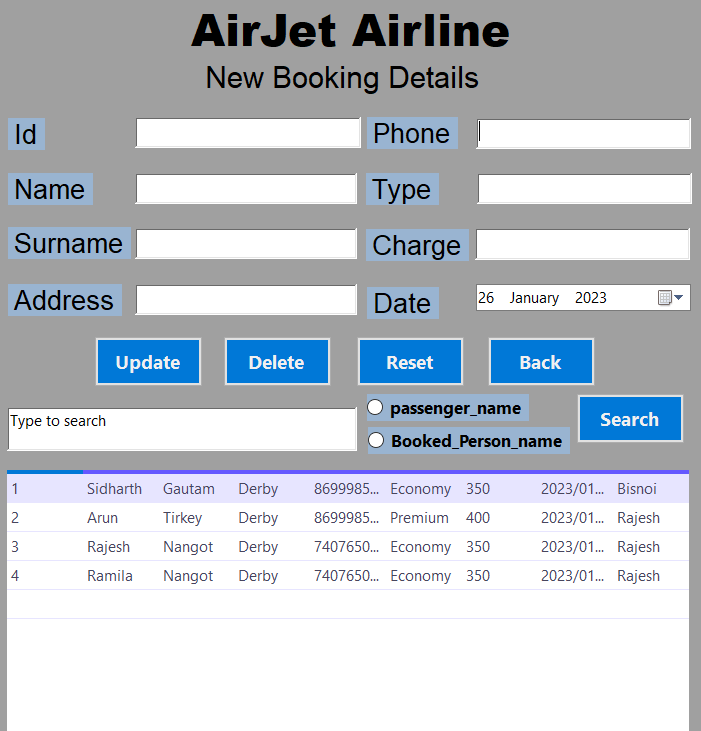


Fig 5: Booking Details Page

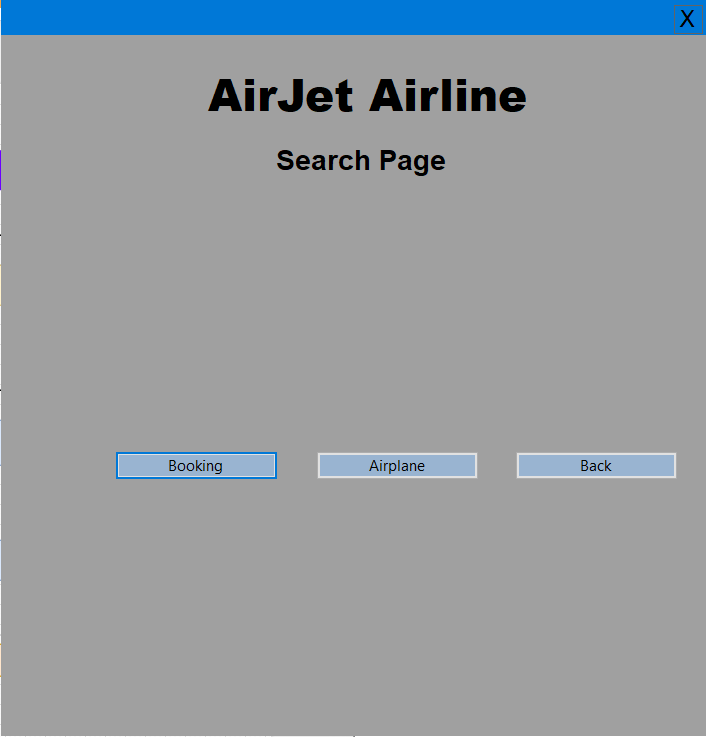


Fig 6: Search Page

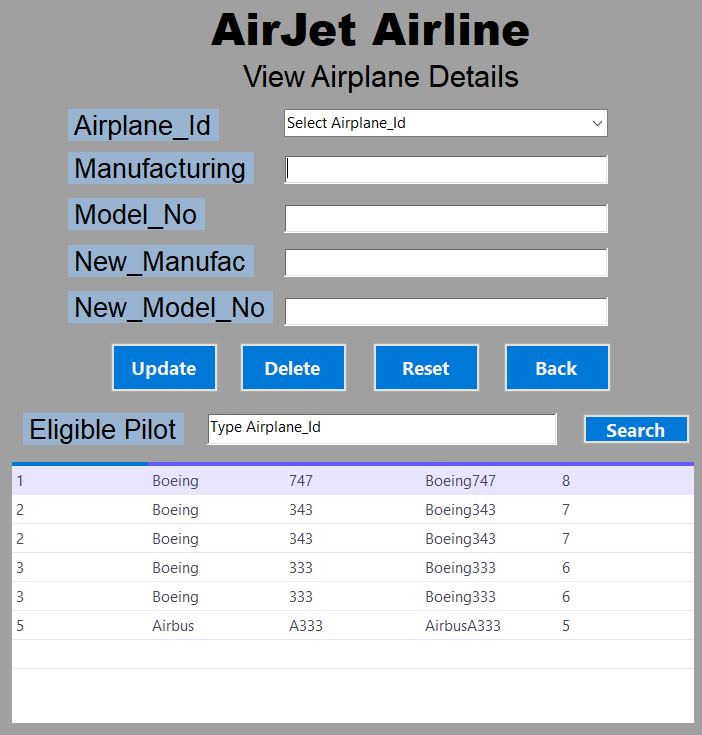


Fig 7: Airplane Details Page

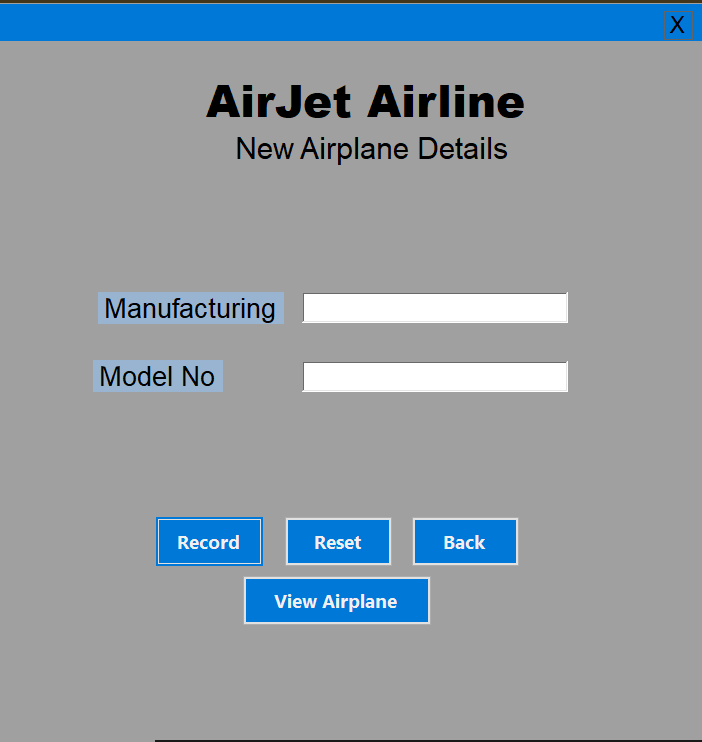


Fig 8: New Airplane Detail Page

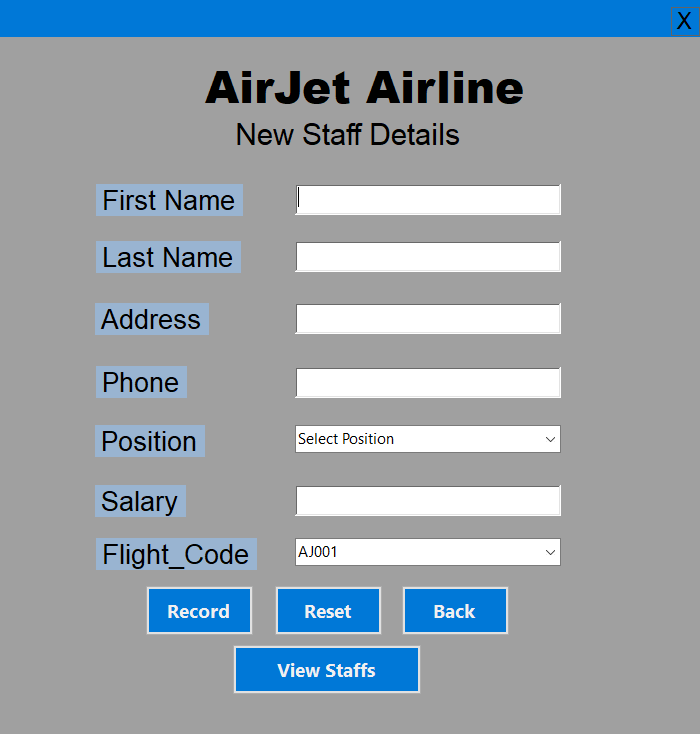


Fig 9: New Staff Detail Page



Fig 10: Staff Details Page

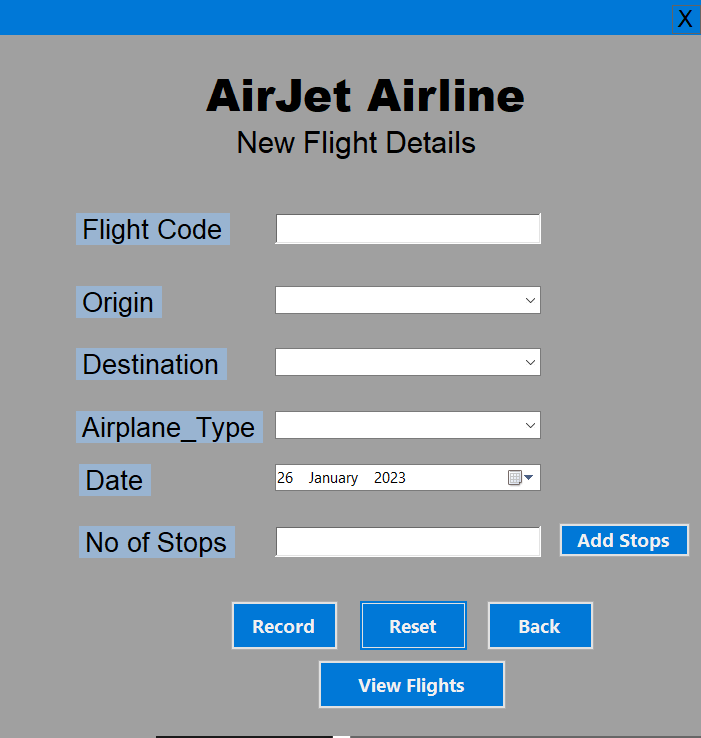


Fig 11: New Flight Detail Page

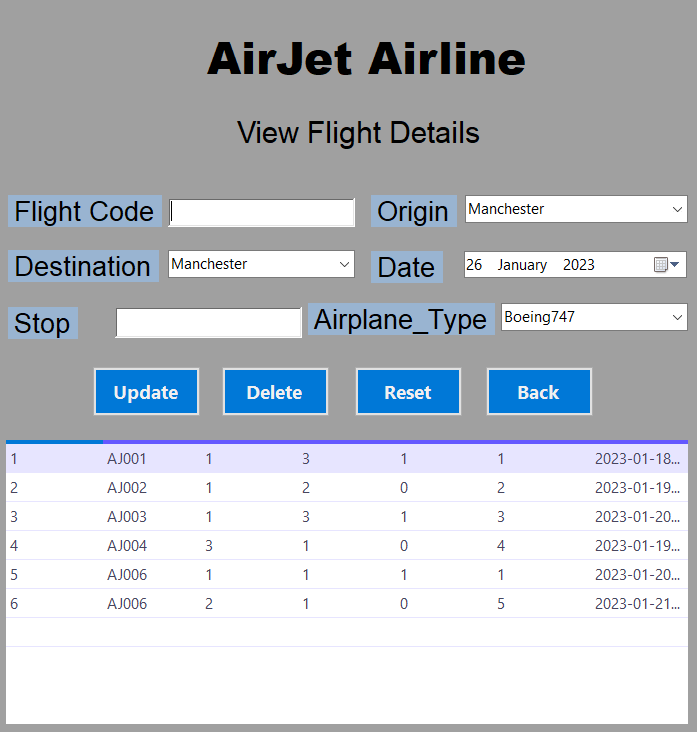


Fig 12: Flight Details Page

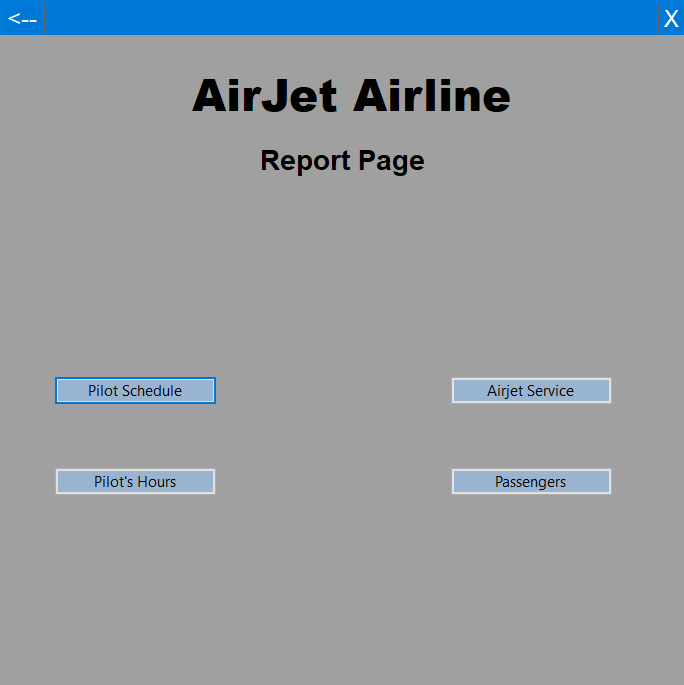


Fig 13: Report Page

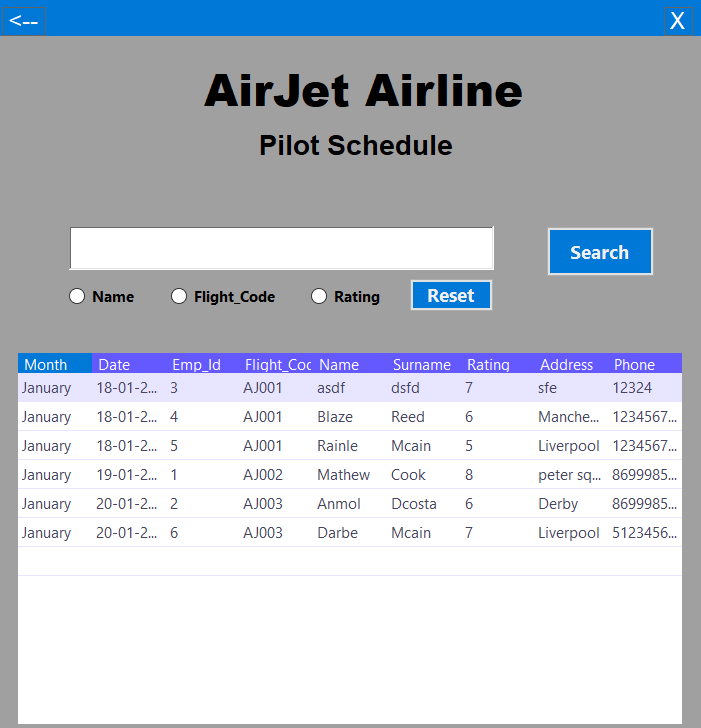


Fig 14: Pilot Schedule Page

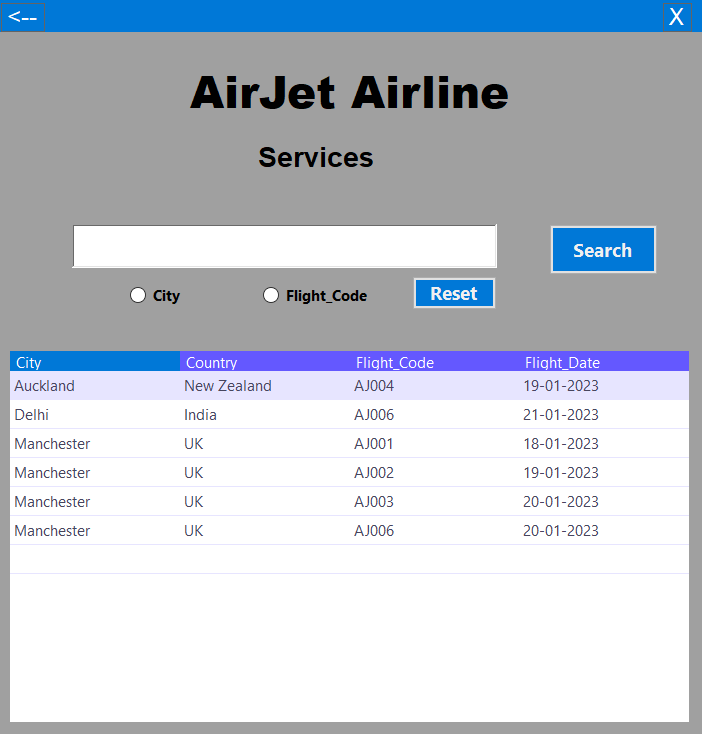


Fig 15: Services Page

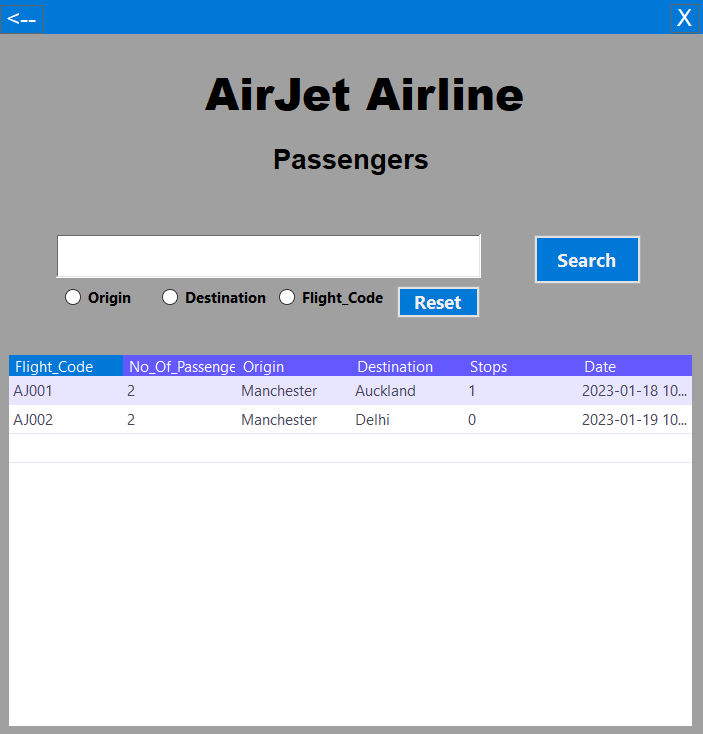


Fig 16: Passengers Page

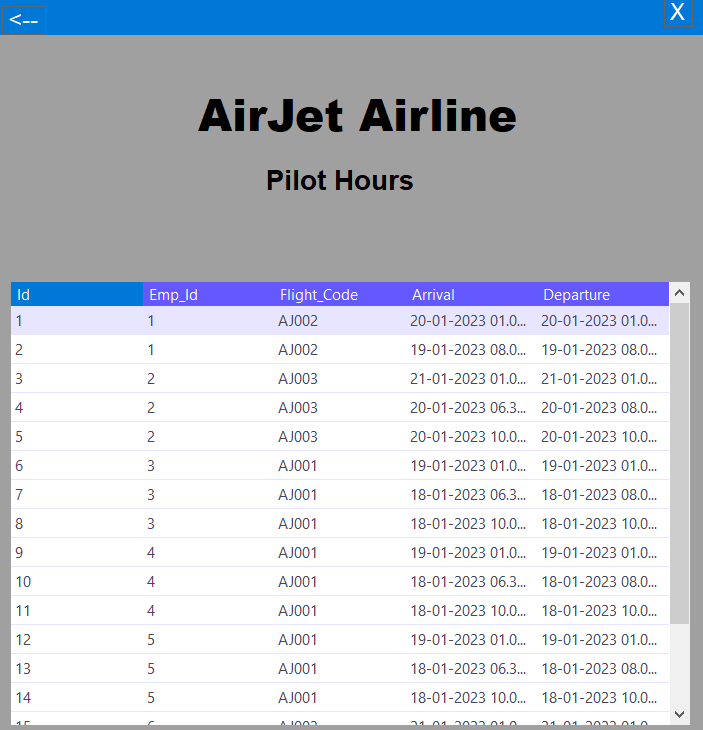
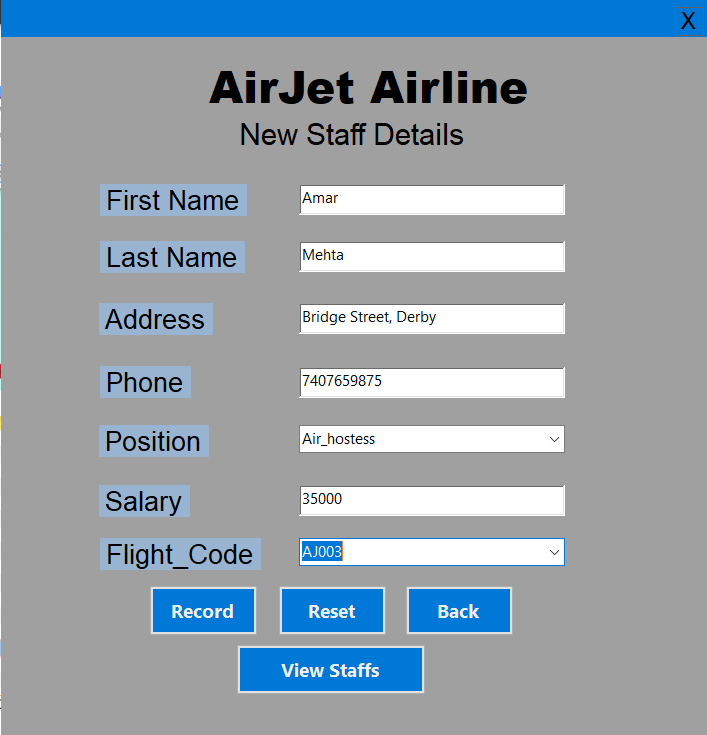


Fig 17: Pilot Hours Page

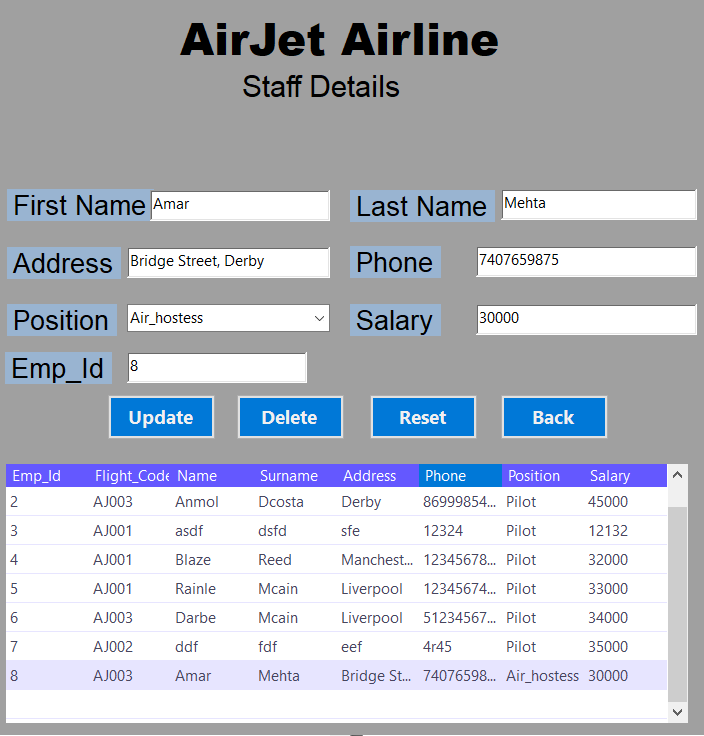
# **Description of the system**

1. **Employee -> add new employee**

The admin must input the necessary information for a new employee member on the employee page, and upon clicking the "record" button, the employee member's information will be added to the database.

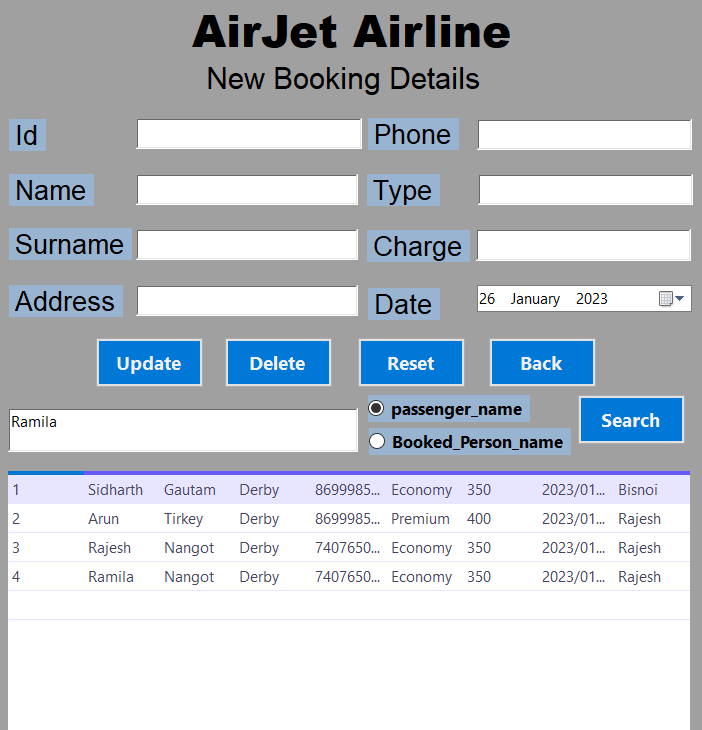
 Graphical user interface

Description automatically generated



1. **Booking -> Find Passengers**

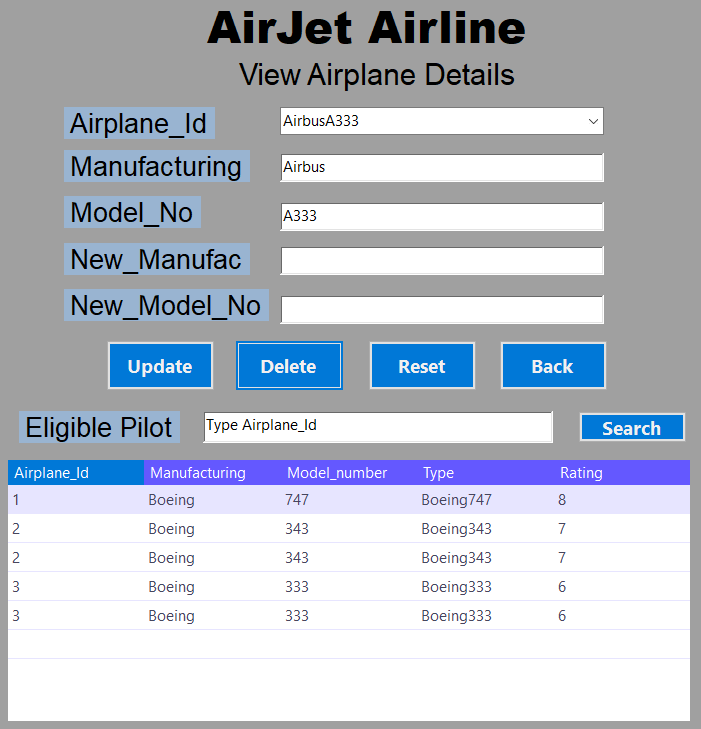
An administrator can retrieve a specific record from the database by clicking the search button and selecting the desired radio button. For example, if the "Passenger name" radio button is selected and the name of the passenger is entered, the corresponding passenger record will be retrieved.

 Graphical user interface, application

Description automatically generated

1. **Airplane -> Delete Airplane**

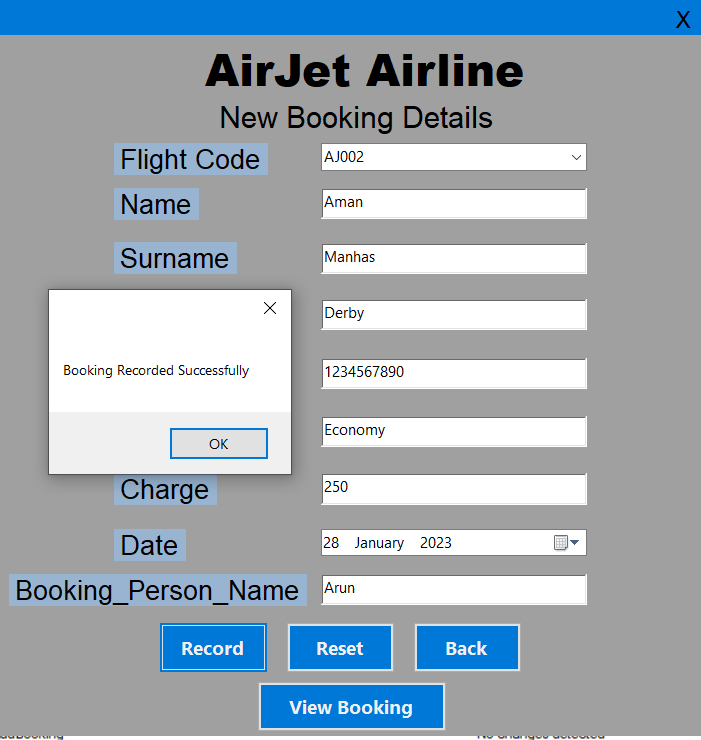
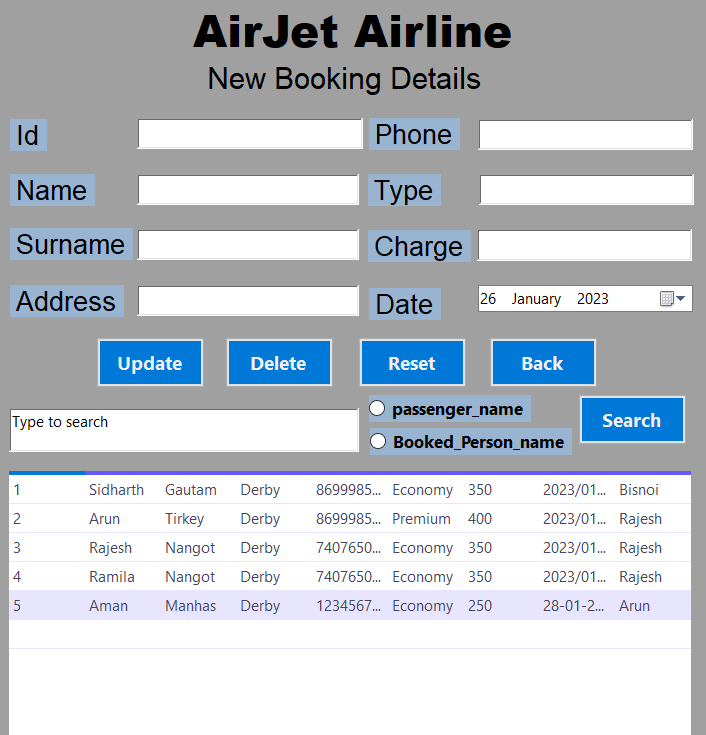
The process of deleting a specific record from the "airplane" table in the database involves clicking the delete button, which will then remove the intended record from the database. Table

Description automatically generated 

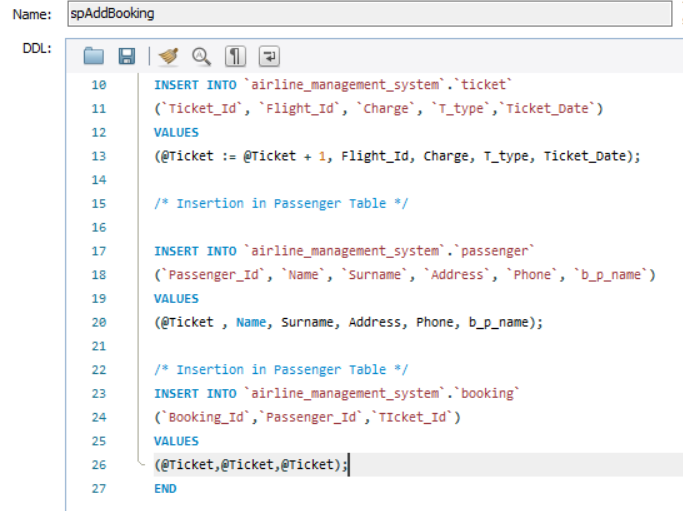
Airplane\_Id =5, Type = AirbusA333 is successfully deleted.

1. **Book / Cancel Flight -> Book a new ticket**

When an administrator wants to book a new flight, they must enter the travel details and then press the "Record" button in Booking page. This will book the flight for the passengers and it can be seen in View Booking page.

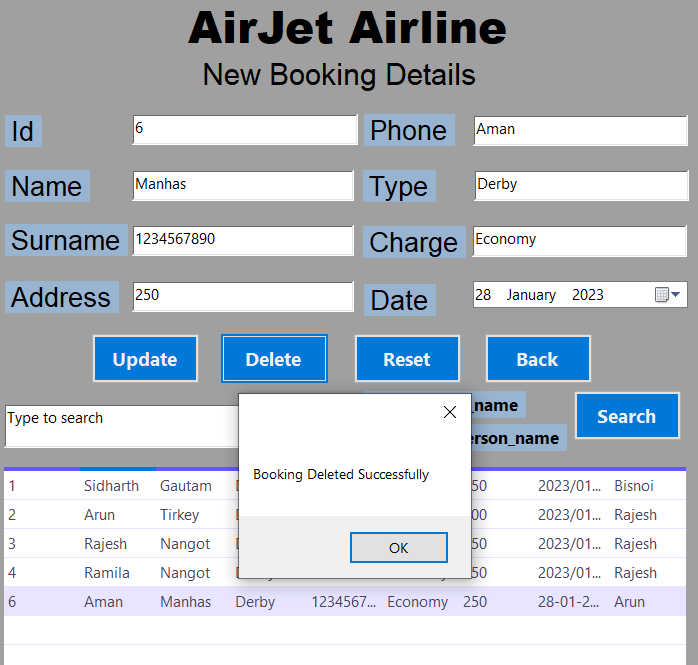
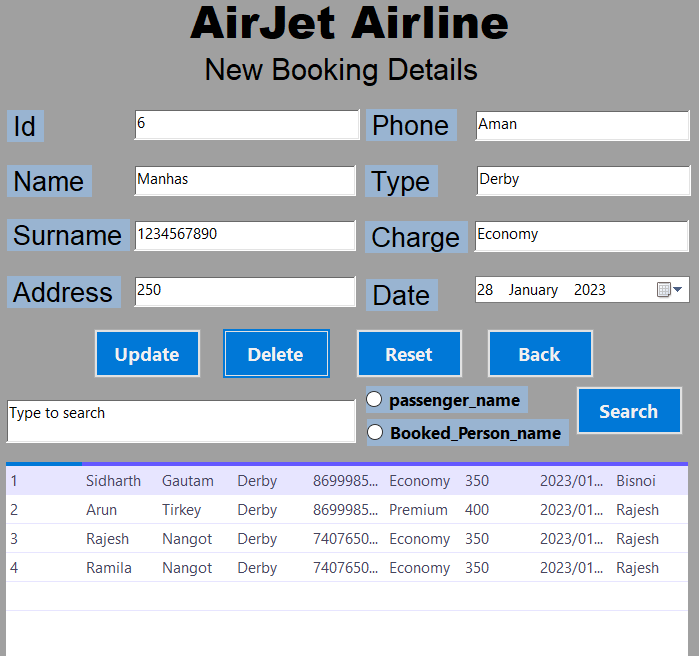
 

Record button stored procedure MySQL code



1. **Cancel Booking**

When an administrator wishes to cancel a reservation that has been made, they should press the delete button on the flight details page. This will remove the booking from the database.

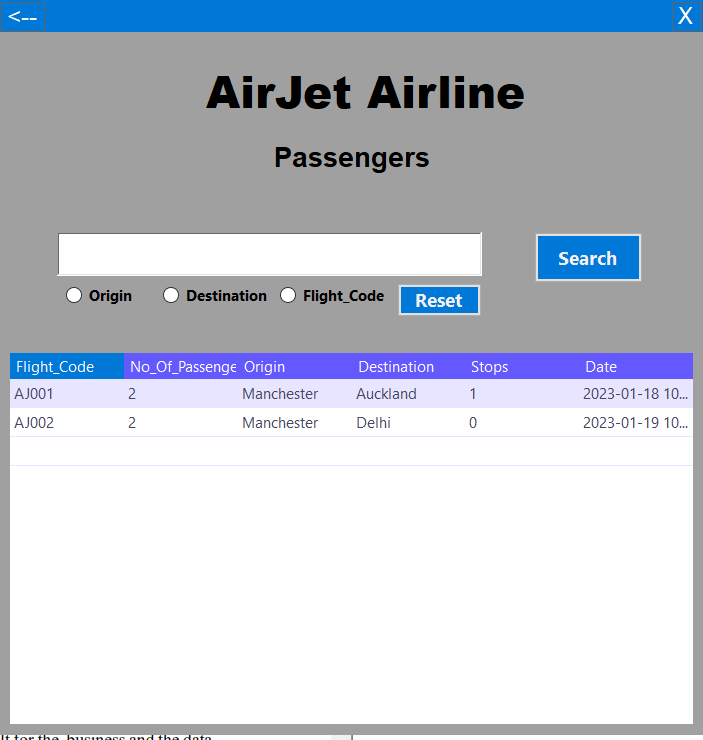
 

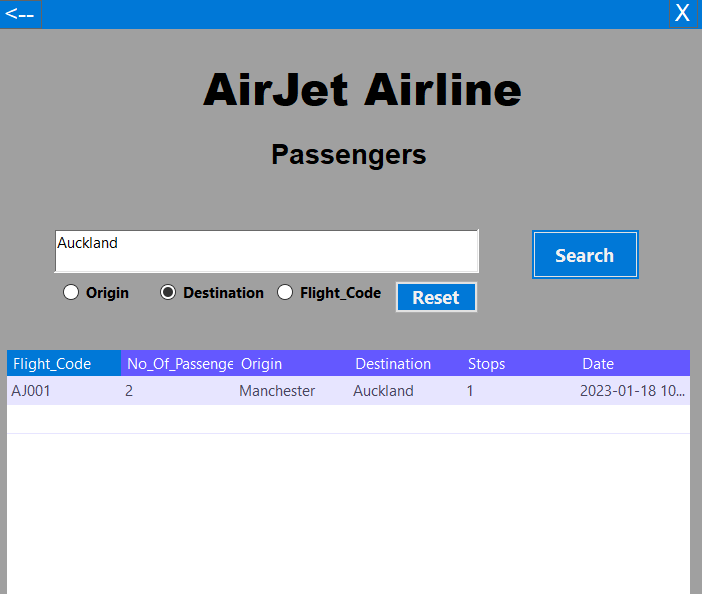
# **Report Generation**

Creating a well-designed database system is essential for any business and generating reports from the database allows the business to easily view a summary of the information, making it simple to print and providing a clear, attractive representation of the data. Window Forms offers advanced, easy to make reports that can be created by pulling data from one or more tables and queries. The purpose of creating these reports is to enable business administrators to make informed decisions, achieve the goals of the business by analyzing the current status, and provide business information during meetings. Reports in Windows form can be created by either selecting a table and using the report tool to customize the desired fields from various tables, or by using queries.

The reports are generated for the following conditions.

1. **No of passengers by flight**





* MySQL query 1 :- **spReportPassengers**

CREATE DEFINER=`root`@`localhost` PROCEDURE `spReportPassengers`()

BEGIN

/\*Display Number of passengers by Flight\_Code\*/

select flight.Flight\_Code, count(flight.Flight\_Code) as No\_Of\_Passengers, (select City from airport where Airport\_Id = flight.Origin) as Origin, (select City from airport where Airport\_Id = flight.Destination) as Destination, flight.NoOfStops as Stops, flight.Date

From passenger

Inner join

ticket on passenger.Passenger\_Id = ticket.Ticket\_Id

inner join

flight on flight.Flight\_Id = ticket.Flight\_Id

group by flight.Flight\_Code, flight.Origin, flight.Destination, flight.NoOfStops, flight.Date;

END

* MYSQL query 2 :- **spReportSelectedPassenger**

CREATE DEFINER=`root`@`localhost` PROCEDURE `spReportSelectedPassenger`(in Inputdata varchar(30), in SelectedField varchar(30))

BEGIN

select flight.Flight\_Code, count(flight.Flight\_Code) as No\_Of\_Passengers, (select City from airport where Airport\_Id = flight.Origin) as Origin, (select City from airport where Airport\_Id = flight.Destination) as Destination, flight.NoOfStops as Stops, flight.Date

From passenger

Inner join

ticket on passenger.Passenger\_Id = ticket.Ticket\_Id

inner join

flight on flight.Flight\_Id = ticket.Flight\_Id

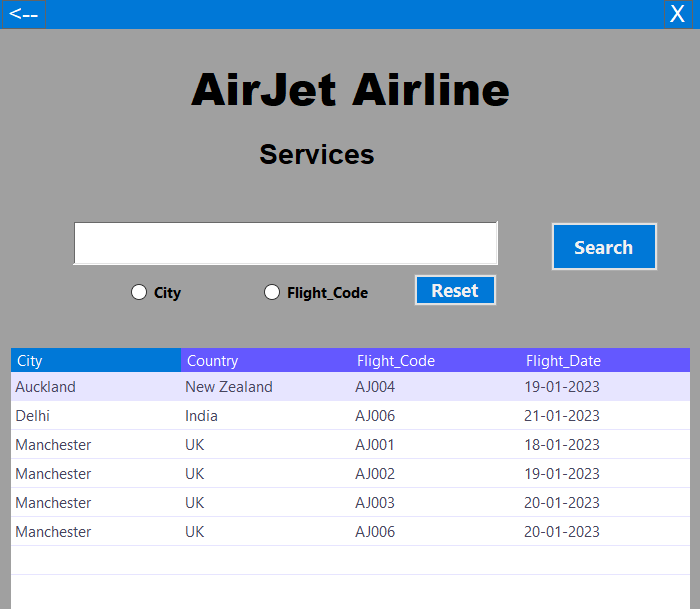
where SelectedField = (select Airport\_Id From airport where airport.City = "'" +Inputdata +"'" limit 1)

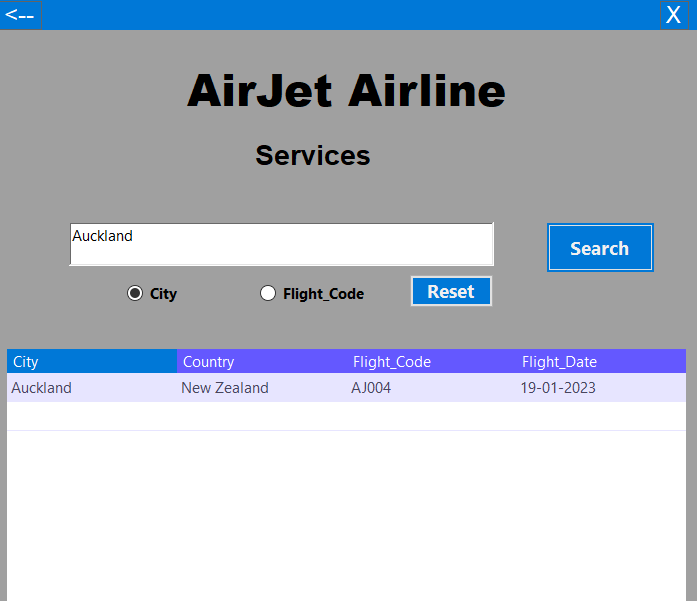
group by flight.Flight\_Code, flight.Origin, flight.Destination, flight.NoOfStops, flight.Date

order by count(flight.Flight\_Code) desc;

END

1. **List of company destinations and their schedule**

****

****

* MySQL query 1 :- **spReportServices**

CREATE DEFINER=`root`@`localhost` PROCEDURE `spReportServices`()

BEGIN

select airport.City, airport.Country, flight.Flight\_Code, CAST(flight.Date AS DATE) AS Flight\_Date

From airport

Inner join

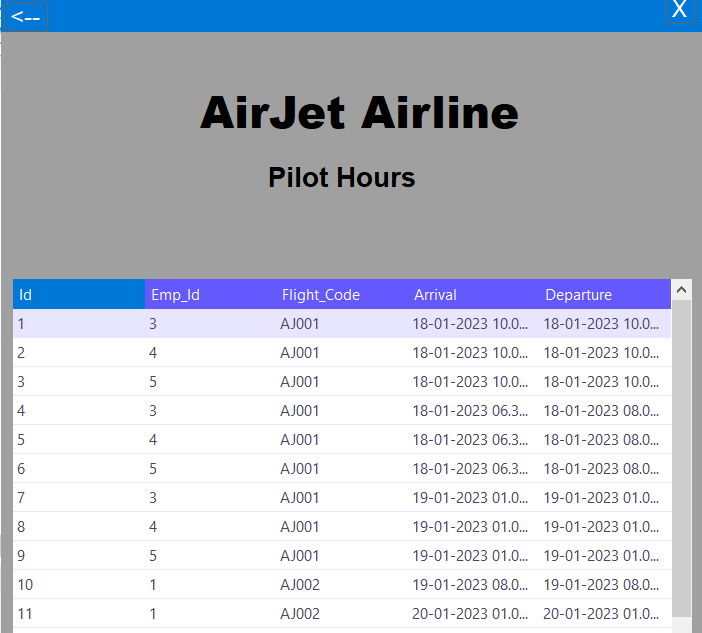
flight on flight.Origin = airport.Airport\_Id

group by airport.City, flight.Flight\_Code, airport.Country, flight.Date

order by airport.City;

END

1. **Number of working hours of all pilots**

****

* MySQL query 1 :- **spReportPilotHours**

CREATE DEFINER=`root`@`localhost` PROCEDURE `spReportPilotHours`()

BEGIN

select ROW\_NUMBER() OVER (ORDER BY (SELECT NULL)) AS Id, pilot.Emp\_Id, Flight\_Code, Arrival, Departure

From pilot

Inner join

employee on employee.Emp\_Id = pilot.Emp\_Id

Inner Join

crew on crew.Emp\_Id = employee.Emp\_Id

Inner Join

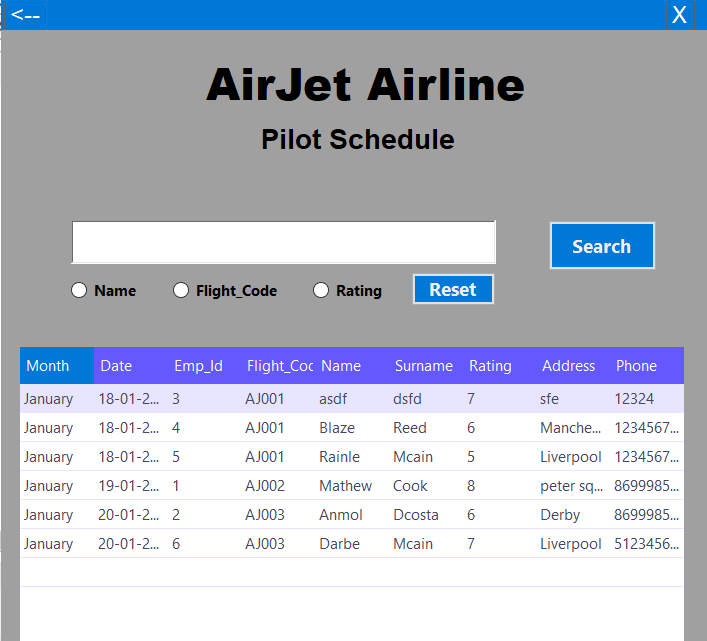
flight on flight.Flight\_Id = crew.Flight\_Id

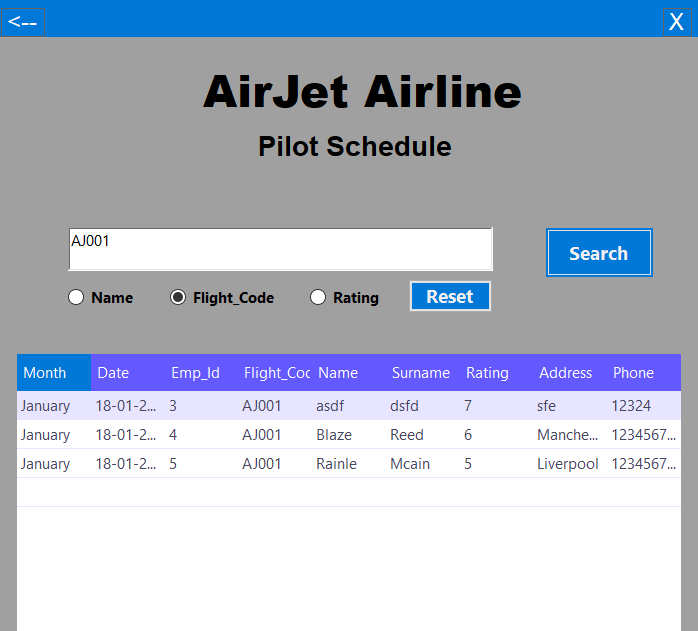
inner join

stop on stop.Flight\_Id = flight.Flight\_Id;

END

1. **Pilot schedule by month**

****

****

**MYSQL query 1 :-**

CREATE DEFINER=`root`@`localhost` PROCEDURE `spReportPilotScheduleNoInput`()

BEGIN

select monthname(flight.Date) as Month , date(flight.Date) as Date, employee.Emp\_Id, flight.Flight\_Code, employee.Name, employee.Surname, rating.Rating, employee.Address, employee.Phone

From rating

Inner join

pilot on pilot.Rating\_Id = rating.Rating\_Id

Inner join

employee on employee.Emp\_Id = pilot.Emp\_Id

Inner Join

crew on crew.Emp\_Id = employee.Emp\_Id

Inner Join

flight on flight.Flight\_Id = crew.Flight\_Id

group by flight.Date,employee.Emp\_Id,flight.Flight\_Code, employee.Name, employee.Surname, employee.Address, employee.Phone, rating.Rating

order by flight.Date;

END

# **Conclusion**

To conclude, a well-designed, efficient system needs well designed database connectivity with front end. MYSQL and Windows Form are loosely coupled with each other using Open database Connectivity and SQL query is written on front end which cause concern to easily get information about content in database. Stored procedure are used to display output from various tables combined into one using JOIN function and WHERE clause to bind outcome with input parameters. The application should have most of the processing on backend but now heavily on database servers rather than on backend program like PHP, dotNet that holds the load and distribute it according to load acquired. This project could be more security tight, Cross-platform which could run on different platforms.

# **Reference**

1. (MYSQL Documentation ) Available at: https://dev.mysql.com/doc/visual-studio/en/visual-studio-connection-manager.html#visual-studio-connection-manager-basic