Fuzzy Systems and Soft Computing
ISSN : 1819-4362

AIR-LINE BOOKING SYSTEM

Krushna Chandra Sahu, Prof. Sasmita Tripathy

Computer Science And Engineering
Gandhi Institute For Technology, INDIA
krushna.sahu2020@gift.edu.in

ABSTRACT— The Airline Booking System is a full-stack web application developed using HTML, CSS, and JavaScript for the front-end, .NET for the back-end, and MySQL as the database management system. This project aims to provide a seamless and efficient online booking experience for airline passengers.

The application allows users to search for flights based on their desired origin, destination, and travel dates. It then displays a list of available flights, along with essential details such as flight duration, layovers, and pricing. Users can select their preferred flight and proceed to the booking page, where they can enter their personal information and payment details.

Keywords: travel booking, reservation management, inventory management,
customer relationship management (CRM)

Introduction

This project is aim to design Airline Booking system that can be used over the phone. However, understanding the technology behind booking system is undoubtedly involve the use of latest technology and software and all of these point to detail research which in this project, titled literature review.

After the research is done, then the knowledge gained will be analysed to give detail idea of specification and requirements for the system.

1.1 Scope

The main scope of the project will be focused on designing a system that will pull up customer's booking record, converting or linking the database records (customer, flight and booking) to the object oriented and event driven-based interface to simplify data input. The project platform is undecided yet as the investigation and analysis stage will pave the way for system requirement or platform options that are available. Currently Unified Modelling language (UML) or Entity Relation Diagram will be one of the tools to start with in showing the relationships between the records. Also software development life cycle strategy is going to be one of my approaches to this project in gathering design and application requirements (table design, table relation, programming language etc). Base on the preliminary investigation, the project graphical user interface for data input and retrieval will be designed in .NET and JS .

1.2 Background

The aim of this project is to design and implement an application that will be used for taking payments for booked flight. This will necessitate connection of database to GUI to improve the data input and output methods graphically.

This application will be however design to compatible with Ms Access for an easy running on local computer but the record can be converted presumably into high level database program like SQL or oracle at the cooperative level for server based.

1.3 Roadmap of report

Chapter 1 briefly describe the project objectives, scope and background.

Chapter 2 is focused on detailed research on technologies behind booking system and tools available for designing any booking system.

Chapter 3 provides brief review of the chapter one (literature review) and specification requirements.

Chapter 4 entails how the requirements were derived, tools to be used and graphical prototype of the interfaces and database schema.

Chapter 5 describes the practical implementation of the design stage, the interfaces, and software choices to accomplish the system functional requirements.

Chapter 6 is focused on checking if the main design at the implementation stage is doing what is designed for.

Appendix shows the Screenshot of testing that can not be inserted into the Test Result table for the avoidance of disarray in the table.

Reference: List of the books and website consulted during the research.

2 Literature Review

This chapter will be centred on the investigation to pave the way for the different approaches available and technology behind the development of any booking system. The research will help in understanding how the booking system works and implements the idea in designing different booking system application (e.g. flight booking/reservation system).

Before the existence of object oriented programming, applications like booking system, sales-related applications are less user-friendly and also laborious to design as many programming languages then are monolithically developed, which are less powerful in accomplishing useful and powerful tasks. Bringing object oriented languages into lime light with GUI supportability makes the commercial applications worthwhile and also rapid interest of companies to switch to the GUI-based applications for their businesses.

2.1 Booking system

What is booking system? In a simple and brief definition, booking system can be described as an application that allows user to input, process and output the booking-related data to provide useful and readable booking report or information.

Booking system can be designed in different ways based on the user requirements and business needs. However, designing high-end booking system with user-friendly interface is more complex but the technology is relatively the same comparing it to the Microsoft Access form. The following shows difference between basic (figure 1) booking system and more powerful one (figure 2). The powerful has a connection (e.g. net etc.) between the interface and the SQL server or Database. There are many database connections available to suite programmers and flexible approaches to create connections to the database or SQL servers.

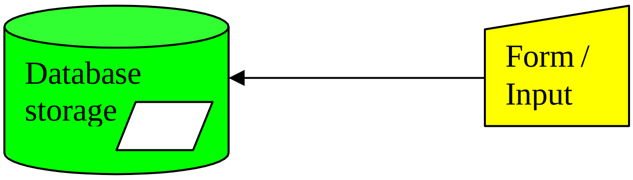


Figure 1 Database connection

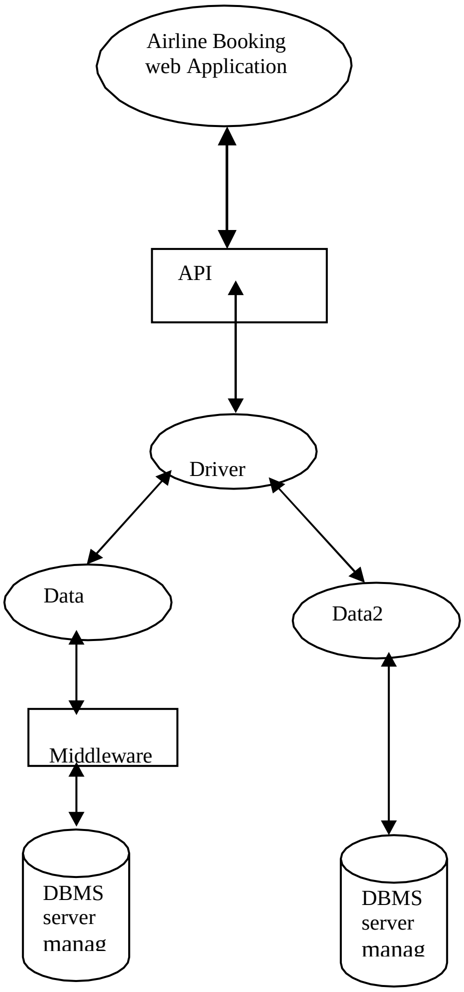


Figure 2 the Pure DataBase Connection Platform

2.2 Different types of database connectivity

The research has shed light on available connections investigate for suitability and compatibility fro the project and tools available in designing useful application.

"Database connectivity provides best-in-class data access solutions that leverage the full power of SQL Server, and enables development of scalable, high- performance applications in widely adopted frameworks"

(http://msdn.microsoft.com/en-us/sqlserver/connectivity.aspx#SNAC).

Database connectivity can be described as a heart of the application that implements connection between the database record storage and GUI. There are different types for connections available for different programming languages for easier interoperation. The following are the brief examples of connections available for use by the programmers

- **ADO.Net**: is the primary data access connection for Net languages. It is used to connect to data sources and retrieve, handle and update data that is stored in SQL server. (http://msdn.microsoft.com/en-us/library/h43ks021.aspx).
- **OLE DB**: it provides data access as ODBC but more codes are written as opposed to ODBC.
- **ODBC**: is the data accesses objection or connection for application written in C and C++ languages.
- **JDBC**: this is designed by the Microsoft as a free ware for easy connections for Java based application to SQL server and also to facilitate the writing and reading of SQL statements. It supports SQL server from 2000 to 2008 R5.
- **PHP5**: is the connectivity design for PHP application to allow reading and writing of SQL data from within PHP scripts (http://msdn.microsoft.com/en-us/sqlserver/connectivity.aspx#SNAC).

2.3 Overview of DBC

In an airline booking system project made using HTML, CSS, JS, .NET, and MySQL Database, the database connection is established using .NET technology. The .NET framework provides the ADO.NET library, which is used to connect, retrieve, and manipulate data in a database. Specifically, for MySQL Database, MySQL Connector/NET is used, which is an implementation of the ADO.NET interface that allows you to connect to MySQL database server from .NET applications.

The connection string is a crucial part of the database connection, which contains the necessary information to connect to the database server, such as the server address, database name, and authentication details. The connection string is then used to create a connection object, which is used to open and close the connection to the database.

Once the connection is established, SQL queries can be executed using the connection object to retrieve or manipulate data in the database. The data retrieved from the database can then be displayed on the front-end using HTML, CSS, and JS.

It is important to note that the connection to the database should be closed after use to free up resources and prevent any potential security risks. This can be done using the "using" statement in C#, which automatically closes the connection when the code exits the block.

2.4 Role of DBC in API

In an airline booking system project made in HTML, CSS, JS, .NET, and MySQL Database, the role of a database connection in the API is crucial. The API (Application Programming Interface) is responsible for handling requests and responses between the client-side and the server-side. When it comes to the database, the API needs to interact with the MySQL Database to perform CRUD (Create, Read, Update, Delete) operations.

To establish a connection between the API and the MySQL Database, you will need to use a database connector, such as MySQL Connector/NET, which is an ADO.NET provider for MySQL. This connector allows .NET applications to connect to MySQL databases and perform database operations.

2.5 SQL IN .NET

First, let's design the database schema for the airline booking system. We'll create the following tables:

Flights: stores information about each flight, including flight number, departure and arrival airports, departure and arrival dates, and flight duration.

Bookings: stores information about each booking, including booking ID, flight number, passenger name, and booking status.

Passengers: stores information about each passenger, including passenger ID, name, and contact information.

2.6 Conclusion

1) **The airline booking system project is a full-stack web application that allows users to search for flights, book tickets, and manage their bookings. The system is built using HTML, CSS, and JavaScript for the front-end, and .NET with MySQL Database for the back-end.**

2)

3) **The front-end of the application provides an intuitive and user-friendly interface that allows users to search for flights based on their origin, destination, and travel dates. The search results are displayed in a clear and concise manner, allowing users to easily compare flights and select the one that best suits their needs. Once a flight is selected, users can proceed to book their tickets by providing their personal and payment details.**

4)

5) **The back-end of the application is powered by .NET and MySQL Database. The .NET framework provides a robust and scalable platform for building the application's business logic and data access layer. The MySQL Database is used to store all the application's data, including flight information, user details, and booking records.**

6)

7) **The system also includes an admin panel that allows administrators to manage flights, users, and bookings. Administrators can add, edit, or delete flights, as well as view and manage user bookings.**

3 Analysis

3.1 Introduction

8) This chapter will summarize the result from the research stage to descriptively analyse the system requirements and specifications and the project objectives.

3.2 Application description

The application should be able to allow customer details input, booking flight, accepting manual payment and cancelling any booking without refund and all of these details are processed and stored in a database system.

3.2.1 Application objectives

This application will be designed mainly for individual or small business use. It can be improved to suite large scale enterprise which will be discussed later in this report. Recapping the purpose of the project and to guide at the design stage, the following are the application objectives. The user will be able to:

- Enter customer details into database via GUI
- Book flight tickets
- Edit customer details
- Delete customer records
- Take payments

3.3 Application input

The application will have interactive interface where the user can enter data. The input capture will be different depending on the task the user is to perform. The following are the inputs

Customer detail GUI: the user will enter the customer interface from here and save into the database storage.

Flight ticket interface: user will book ticket for registered customers.

Payment: the basic transaction is carried out via this interface.

Cancel: Flight ticket or transaction cancellation is input through this form.

Search: the user will search for customer details using unique data field.

3.4 Application process (use case)

The process occurs in the main database. This is diagrammatically described below.

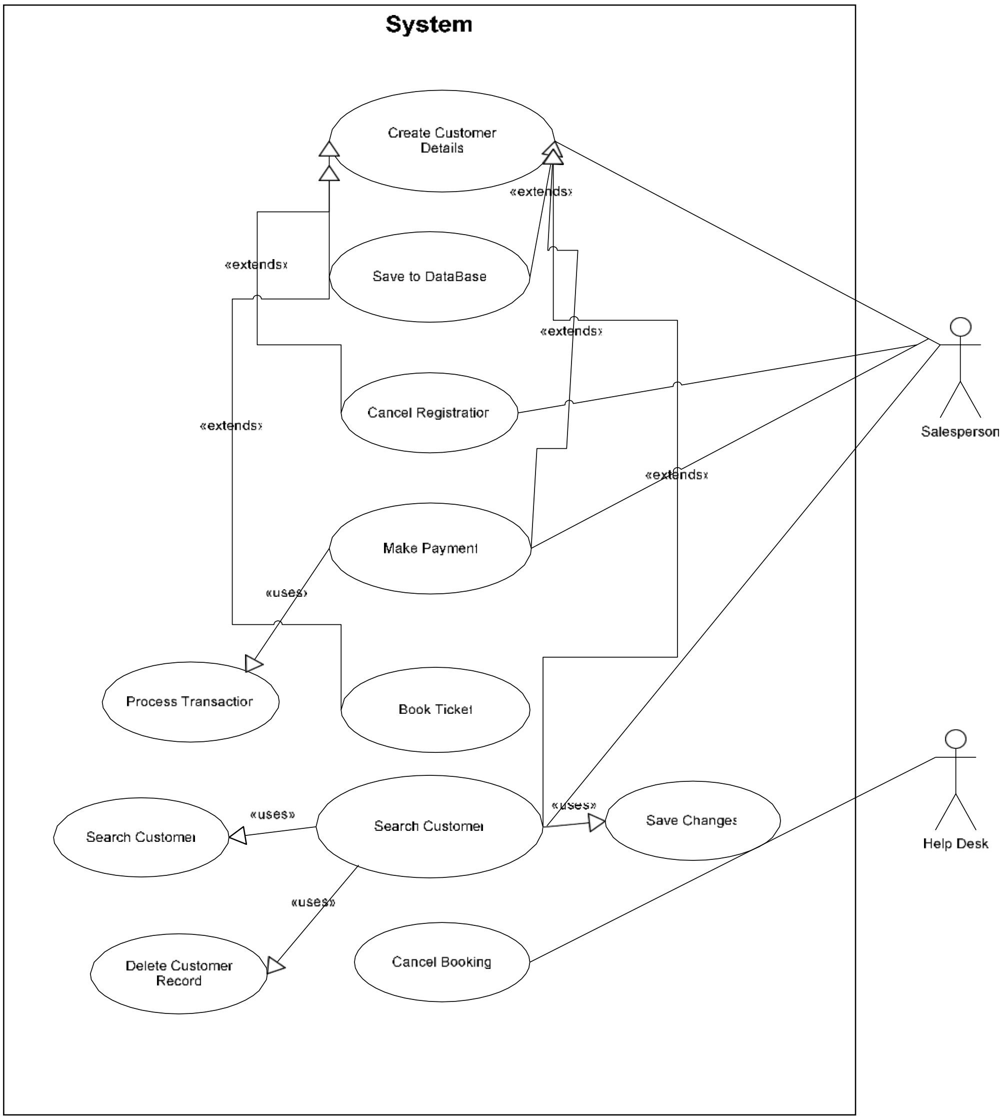


Figure 3 Use Case showing the System process

3.5 Application Output

The output of the application can be inform of a report but mainly will be in the table. The following are the possible tables that will be presumably designed.

Customer Table: showing the customer records
Flight Ticket: showing the booked flight tickets
Payment: showing the payment records.
Cancel: showing cancellation records.
Search: the customer details will be displayed when search is button is clicked and update will be stored in the customer detail records.

3.6 Business Requirements

This system is aimed to be Helpdesk-Driven rather than Internet-Driven due to some security issues that come with internet based using 2012 Olympic online booking ticket technical failure (Evening Standard, 2011), which causes transaction chaos and inconvenient to some customers, as evidence to how important helpdesk-driven or manual booking system. . Also this can be used as a parallel application with the online booking which will help in the case of internet related problems (e.g. heavy network traffic, server overheating etc).

3.7 User Requirements

The user should have been able to log onto the system to access the program.

3.8 Functional Requirements

These are what the system should process:

Enter new customers' details
Update customers' details
Search for customers using their passport number
Enter flight tickets
Enter payment details
Cancel booking.

Delete record.

3.9 Non-Functional Requirements

The application performance requirement is basic computer system, which are the system or hardware requirements.

3.10 System Requirements

The system requirements are of twofold, first fold will be focused on minimum hardware specification and software specification will be the second fold.

3.10.1 Hardware requirements

- Processor; Pentium Dual core processor minimum, Intel or AMD
- Hard drive; 60 GB minimum
- RAM; 3GB
- Keyboard
- Mouse
- Disc drive
- Monitor

3.10.2 Software Specification

Front-End

- HTML
- CSS
- JavaScript
- BootStrap

Back-End

- .NET
- MySQL

4 Design

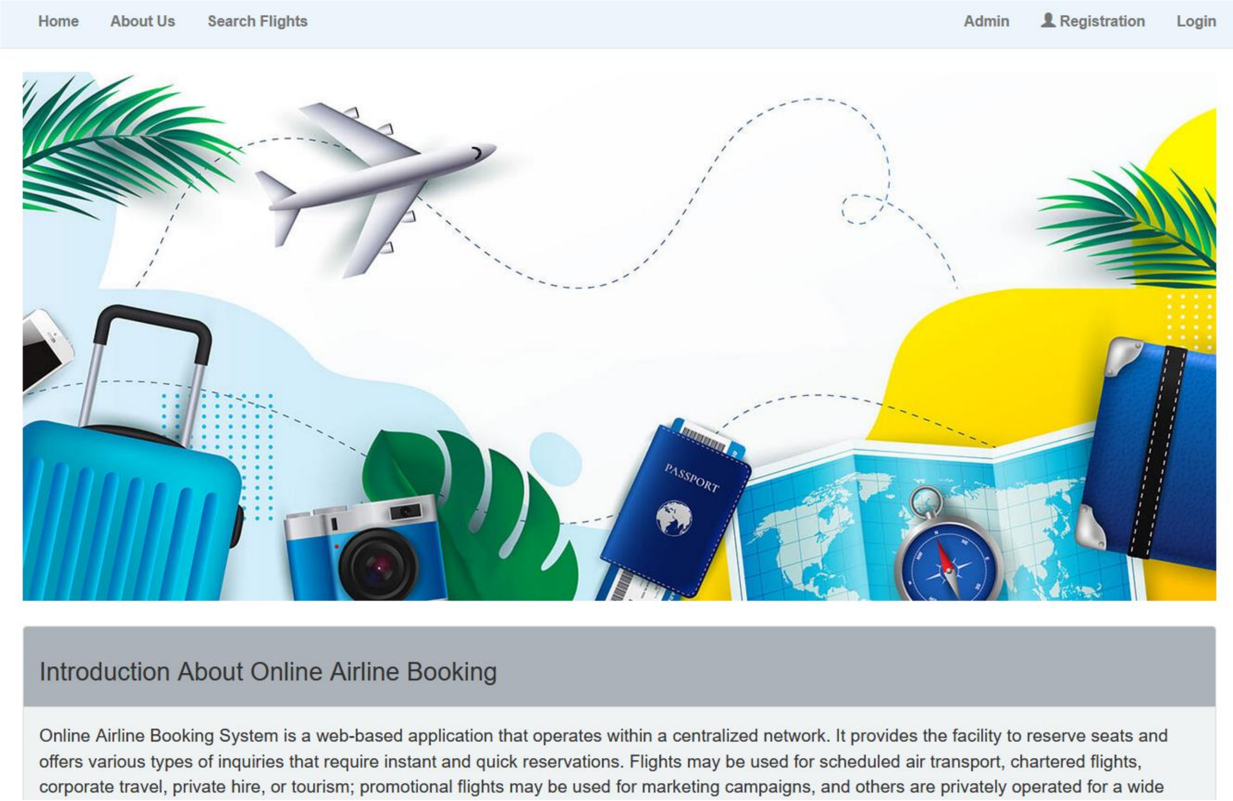
4.1 Introduction

This chapter focus on the design model adopted from the analysis stage of the project, which will be blue print for the implementation stage.

The design will be of twofold which will be hierarchically selected and executed base on there dependency. The first fold will be brainstorming on how the interface and input prototype will be designed, following by the methods to use and tools that will help in achieving the design objectives.

4.2 Input interface

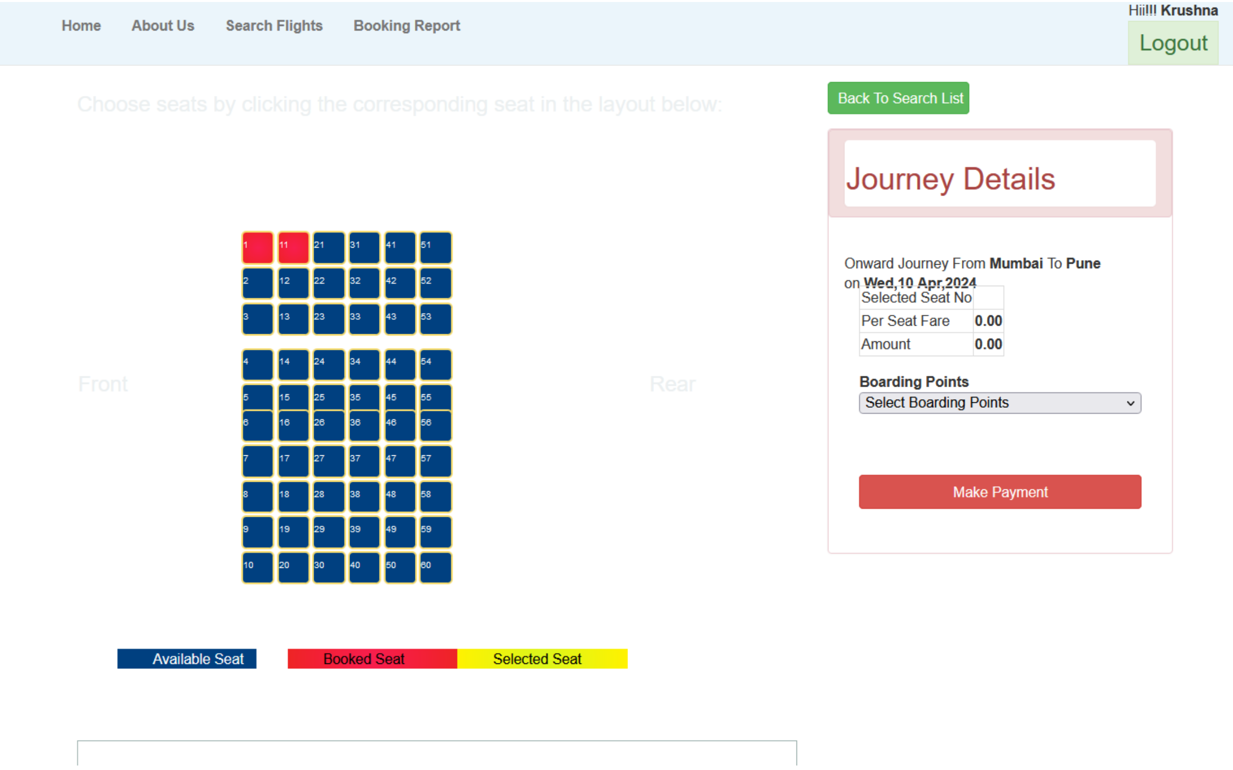
There are many tools available to design user interface prototype and considering the limited time available for this project, Microsoft Visio will be used for professionally looking design. Project inputs are shown below:



Introduction About Online Airline Booking

Online Airline Booking System is a web-based application that operates within a centralized network. It provides the facility to reserve seats and offers various types of inquiries that require instant and quick reservations. Flights may be used for scheduled air transport, chartered flights, corporate travel, private hire, or tourism; promotional flights may be used for marketing campaigns, and others are privately operated for a wide

Figure 4 Customer Interface prototype



Choose seats by clicking the corresponding seat in the layout below:

1	11	21	31	41	51
2	12	22	32	42	52
3	13	23	33	43	53
4	14	24	34	44	54
5	15	25	35	45	55
6	16	26	36	46	56
7	17	27	37	47	57
8	18	28	38	48	58
9	19	29	39	49	59
10	20	30	40	50	60

Available Seat Booked Seat Selected Seat

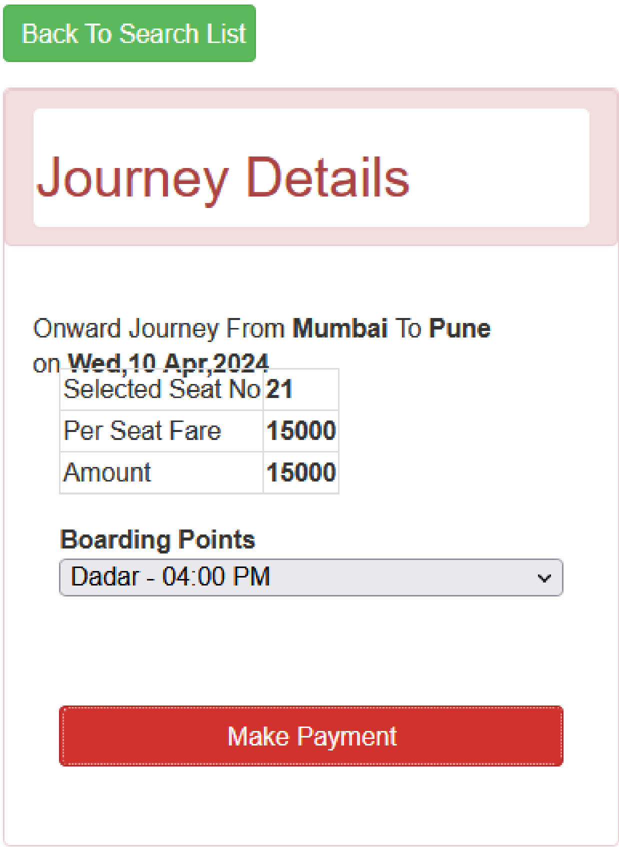
Journey Details

Onward Journey From **Mumbai** To **Pune**
on **Wed,10 Apr,2024**

Selected Seat No	
Per Seat Fare	**0.00**
Amount	**0.00**

Boarding Points
Select Boarding Points

Figure 5 Flight Ticket Booking Interface



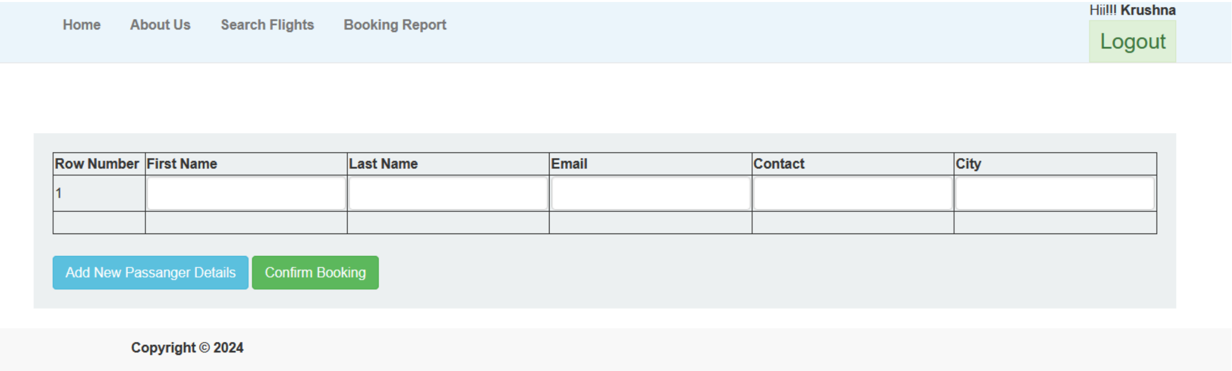
Journey Details

Onward Journey From **Mumbai** To **Pune**
on **Wed,10 Apr,2024**

Selected Seat No	**21**
Per Seat Fare	**15000**
Amount	**15000**

Boarding Points
Dadar - 04:00 PM

Figure 6 Payment Interface prototype



Row Number	First Name	Last Name	Email	Contact	City
1					

Figure 7 Cancel Booking Interface

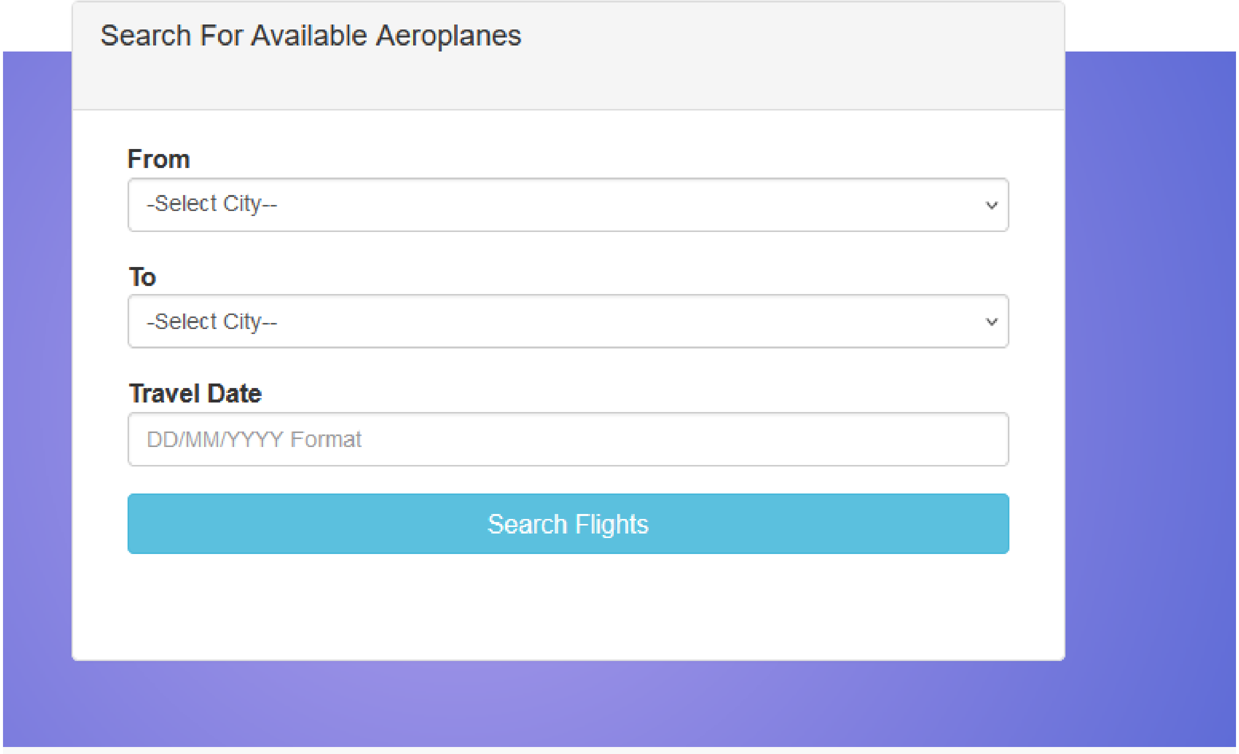
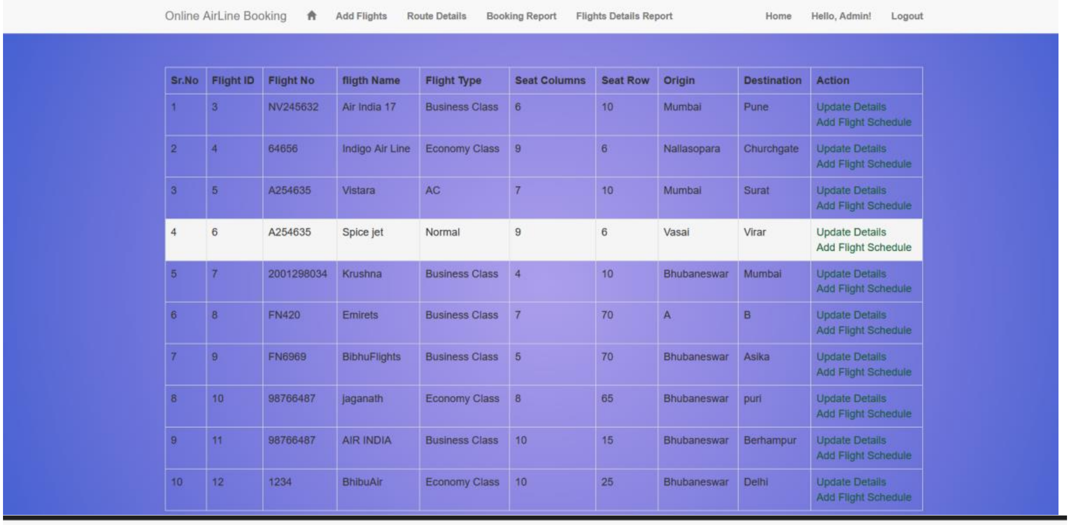
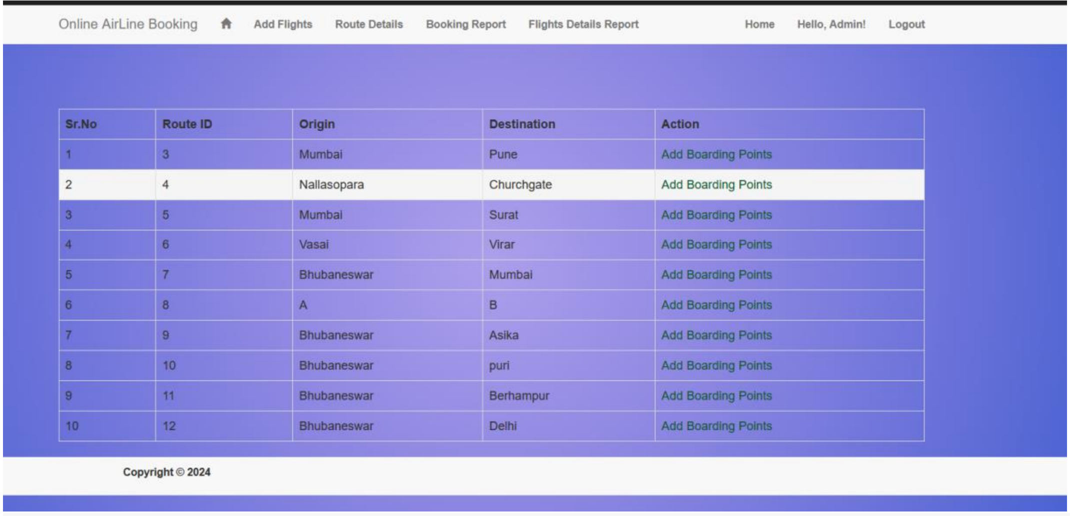


Figure 8 Search Interface

Figure of Admin Side

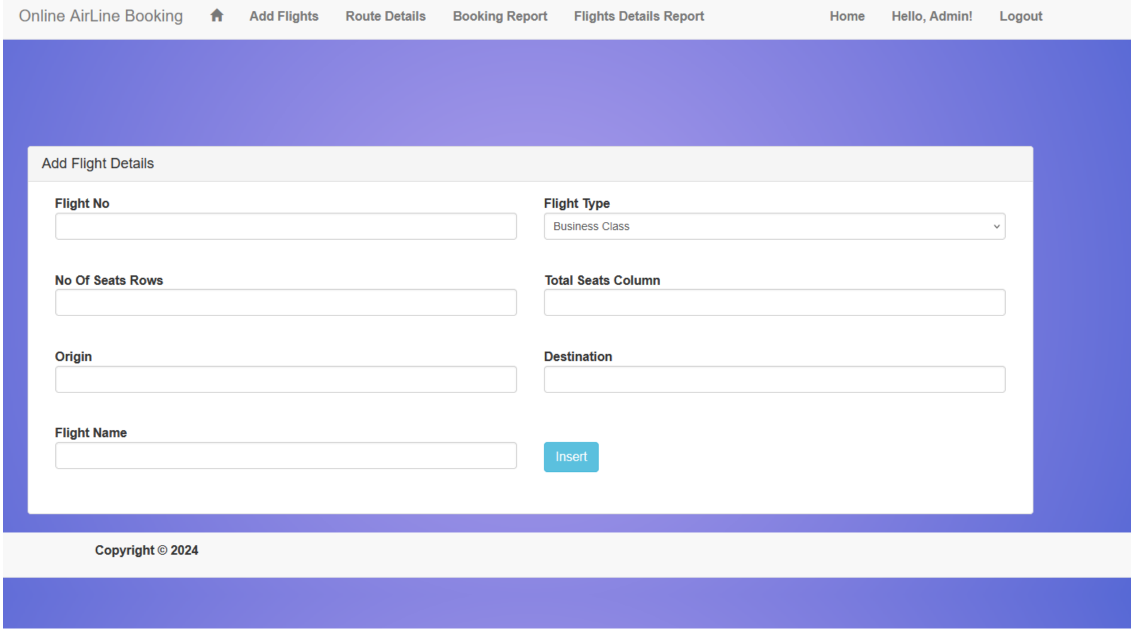


Sr.No	Flight ID	Flight No	fligth Name	Flight Type	Seat Columns	Seat Row	Origin	Destination	Action
1	3	NV245632	Air India 17	Business Class	6	10	Mumbai	Pune	Update Details Add Flight Schedule
2	4	64656	Indigo Air Line	Economy Class	9	6	Nallasopara	Churchgate	Update Details Add Flight Schedule
3	5	A254635	Vistara	AC	7	10	Mumbai	Surat	Update Details Add Flight Schedule
4	6	A254635	Spice jet	Normal	9	6	Vasai	Virar	Update Details Add Flight Schedule
5	7	2001298034	Krushna	Business Class	4	10	Bhubaneswar	Mumbai	Update Details Add Flight Schedule
6	8	FN420	Emirets	Business Class	7	70	A	B	Update Details Add Flight Schedule
7	9	FN6969	BibhuFlights	Business Class	5	70	Bhubaneswar	Asika	Update Details Add Flight Schedule
8	10	98766487	jaganath	Economy Class	8	65	Bhubaneswar	puri	Update Details Add Flight Schedule
9	11	98766487	AIR INDIA	Business Class	10	15	Bhubaneswar	Berhampur	Update Details Add Flight Schedule
10	12	1234	BhibuAir	Economy Class	10	25	Bhubaneswar	Delhi	Update Details Add Flight Schedule



Sr.No	Route ID	Origin	Destination	Action
1	3	Mumbai	Pune	Add Boarding Points
2	4	Nallasopara	Churchgate	Add Boarding Points
3	5	Mumbai	Surat	Add Boarding Points
4	6	Vasai	Virar	Add Boarding Points
5	7	Bhubaneswar	Mumbai	Add Boarding Points
6	8	A	B	Add Boarding Points
7	9	Bhubaneswar	Asika	Add Boarding Points
8	10	Bhubaneswar	puri	Add Boarding Points
9	11	Bhubaneswar	Berhampur	Add Boarding Points
10	12	Bhubaneswar	Delhi	Add Boarding Points

Copyright © 2024



Add Flight Details

Flight No

Flight Type: Business Class

No Of Seats Rows

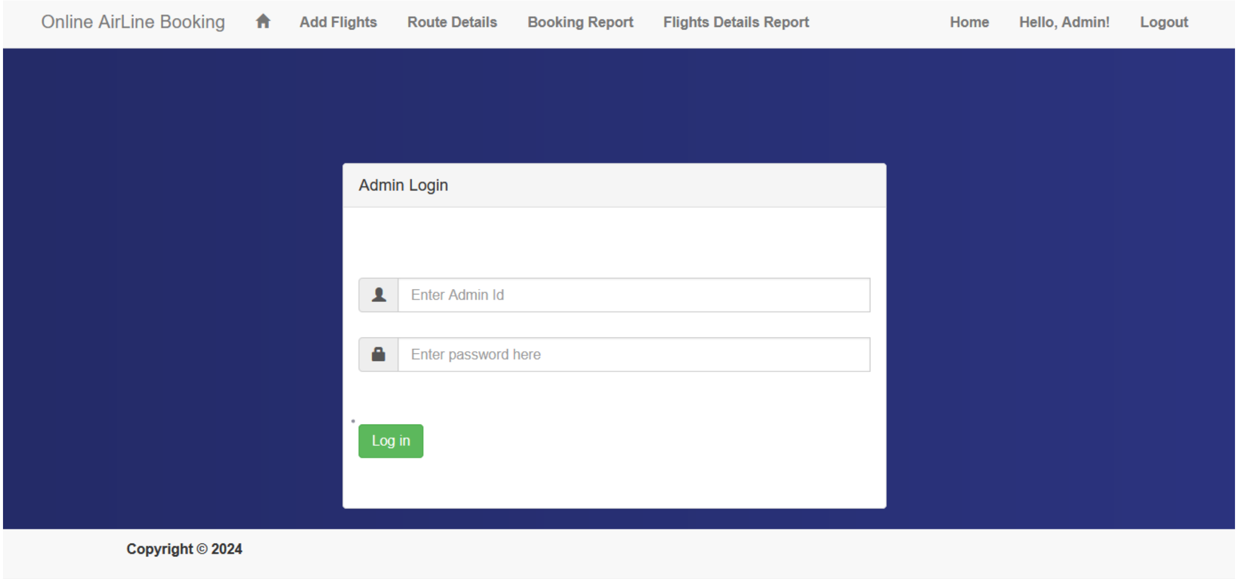
Total Seats Column

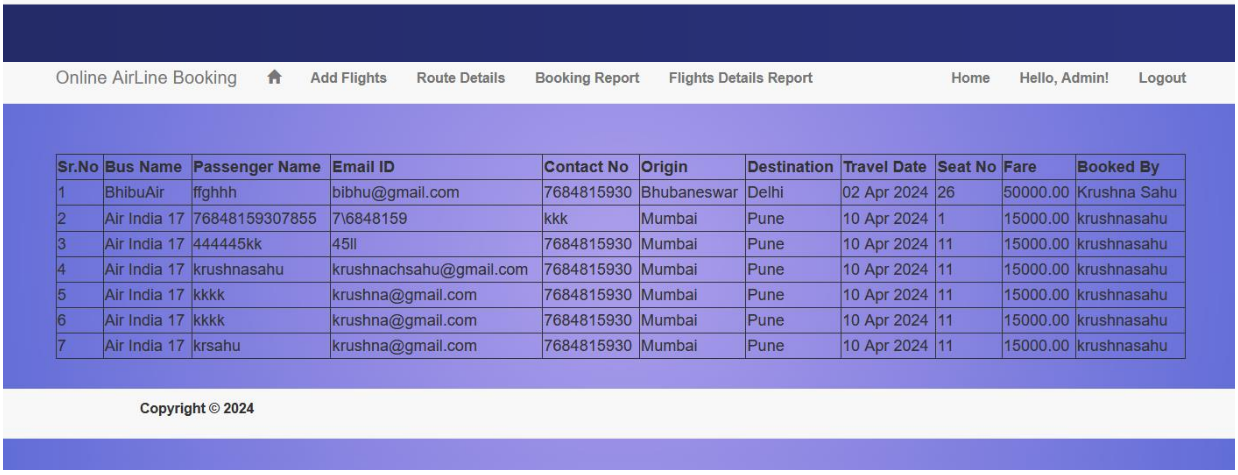
Origin

Destination

Flight Name

Copyright © 2024





Sr.No	Bus Name	Passenger Name	Email ID	Contact No	Origin	Destination	Travel Date	Seat No	Fare	Booked By
1	BhibuAir	ffghhh	bibhu@gmail.com	7684815930	Bhubaneswar	Delhi	02 Apr 2024	26	50000.00	Krushna Sahu
2	Air India 17	76848159307855	7\6848159	kkk	Mumbai	Pune	10 Apr 2024	1	15000.00	krushnasahu
3	Air India 17	444445kk	45ll	7684815930	Mumbai	Pune	10 Apr 2024	11	15000.00	krushnasahu
4	Air India 17	krushnasahu	krushnachsahu@gmail.com	7684815930	Mumbai	Pune	10 Apr 2024	11	15000.00	krushnasahu
5	Air India 17	kkkk	krushna@gmail.com	7684815930	Mumbai	Pune	10 Apr 2024	11	15000.00	krushnasahu
6	Air India 17	kkkk	krushna@gmail.com	7684815930	Mumbai	Pune	10 Apr 2024	11	15000.00	krushnasahu
7	Air India 17	krsahu	krushna@gmail.com	7684815930	Mumbai	Pune	10 Apr 2024	11	15000.00	krushnasahu

4.3 Tools and Software choice

Having successfully created input/interface prototype, the next line of action will be choice of tools and software to accomplish the task efficiently.

Compatibility between the tools and software choice should be prioritised to avoid unnecessary delay. The project platform will be Java 1.2 version and the tool will be J-Creator. I chose J-Creator as my IDE because it is free but can be upgraded to the enterprise version which is not a concern in this project and also the interface created will be professional as opposed to the Text Pad which is a freeware too. UML is one of software developer tools and is highly recommended to adopt it at the design stage. There are many tools available for creating UML related diagrams and in this project, Microsoft Visio studio will be first choice having familiar with it and that will save times on brainstorming around third party tools.

4.4 Class Diagram

It is necessary to have the clear understanding of how the class will be created when coding and fore-think about the possible methods, interface, attributes and variables that can be possibly used. The internationally and professionally recognised too for achieving this task is Unified Modeling Language (UML) which splits into class or object diagram. The following is the class diagram for my project and there associations:

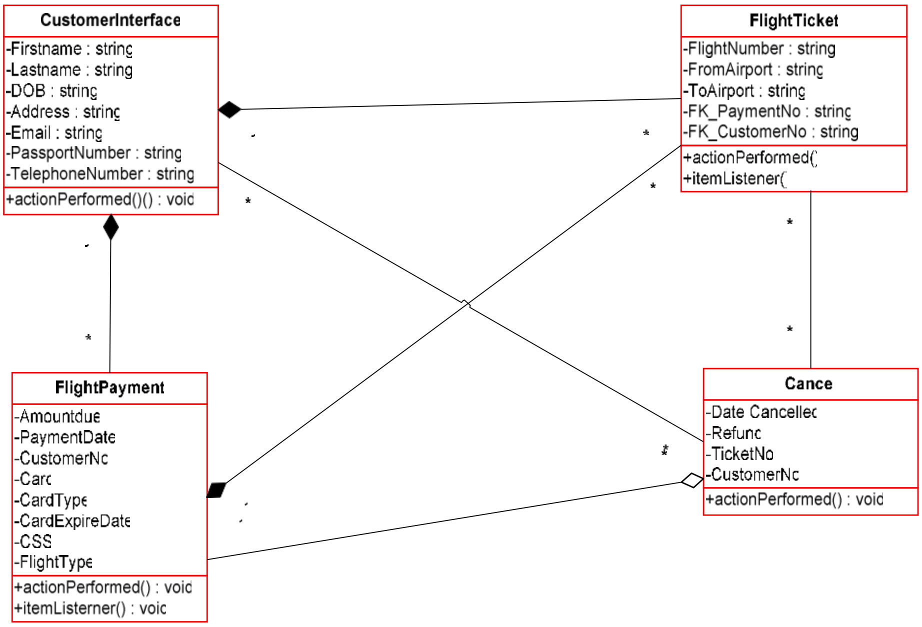


Figure 9 Class diagram and there associations

4.5 Database relation schema

It is understandable, if there is more than one table in a database, then they must be somehow related and this necessitate to show the entity relational diagram of the database.

The ERD is not just about links, but the flow of communications between the table and also constraints.

Example; a customer can book many flight tickets, many payment can be made by one customer, many customers can cancel many bookings etc. Also, one of the advantages of creating relationship between the tables is to enforce referential integrity which ensures there are no data replications and primary keys properties are not over ride by mistake.

The diagram below shows the relationship between the tables

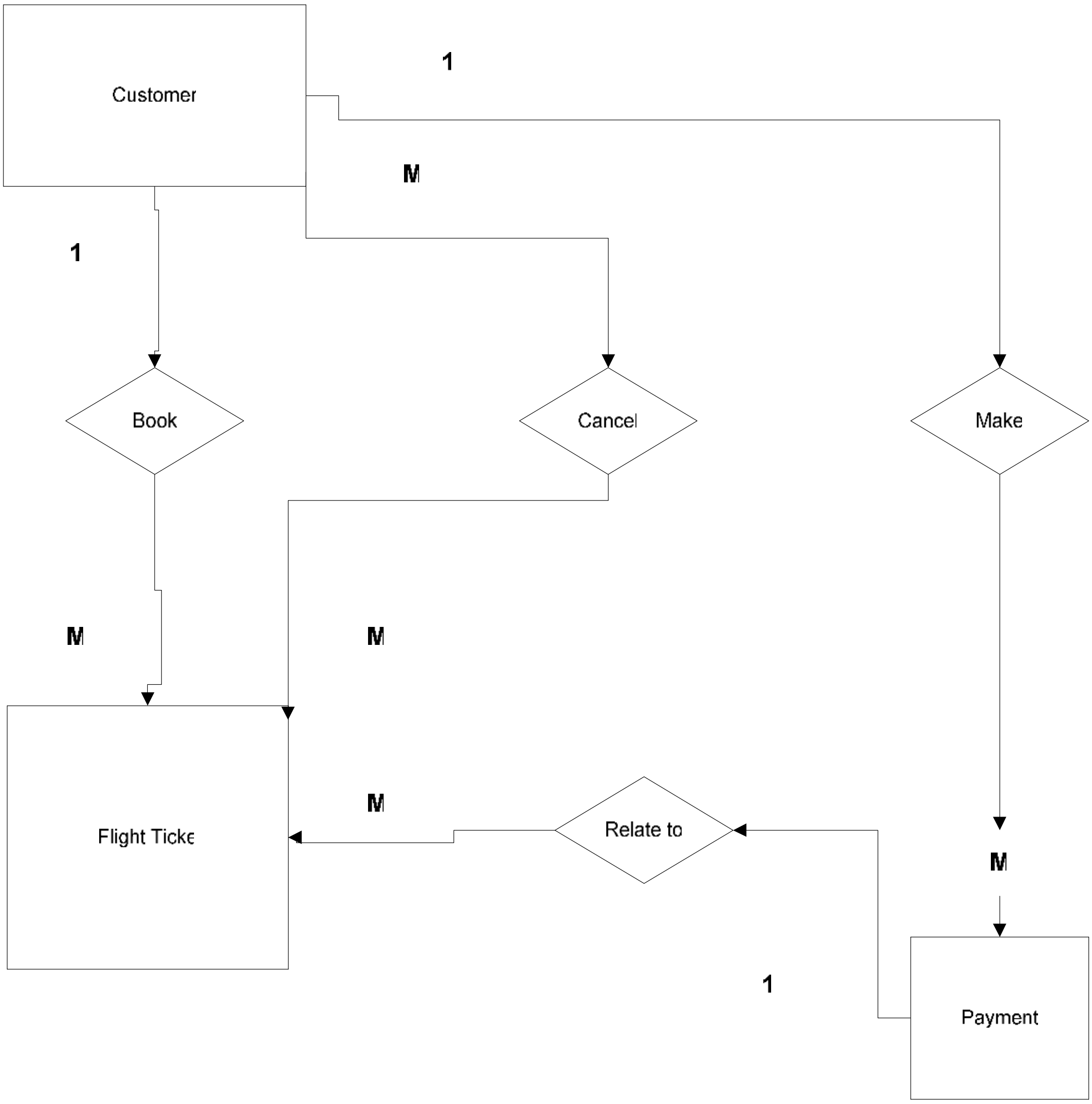


Figure 10 DATABSE ERD

4.6 Process

The process that occurs in the project or application is diagrammatically shown below;

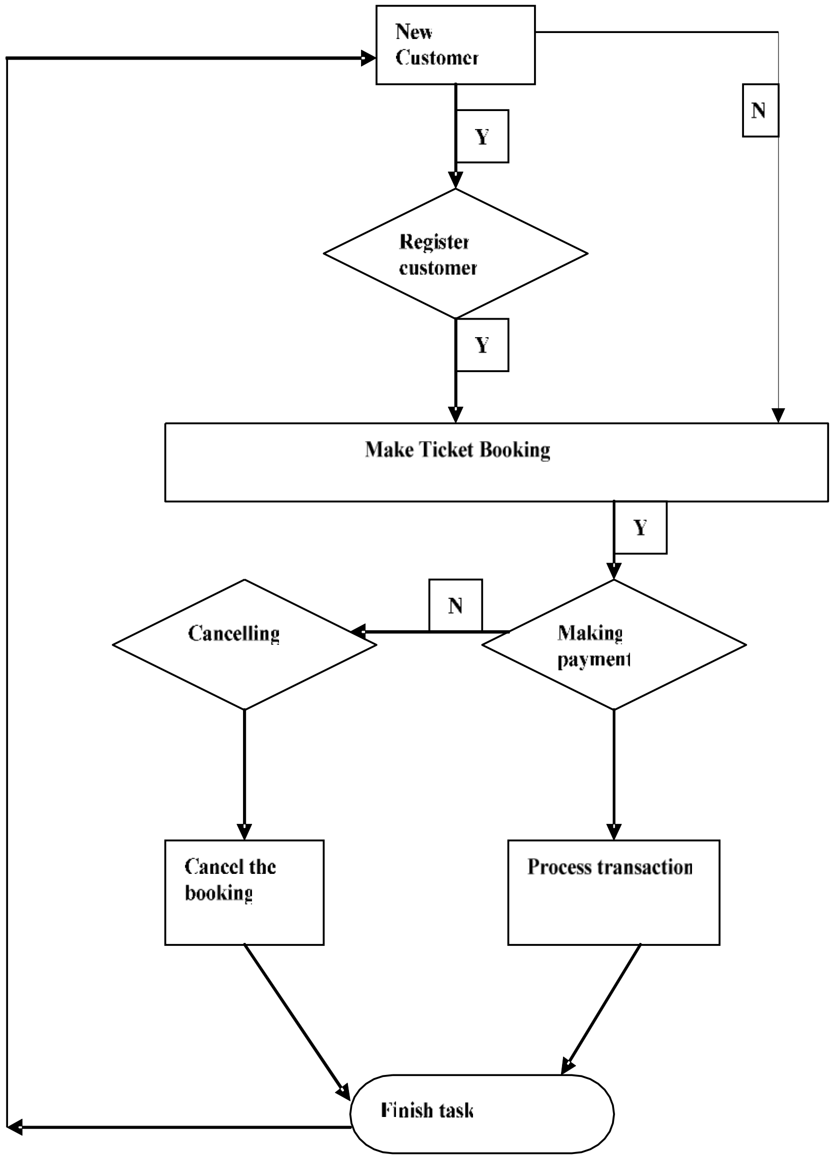


Figure 11 Flow of Process

4.7 Output

The out put of system will be displayed in GUI and/or DOS screen

Most of the output for customer details, flight/booking records and payment records can be viewed in the tables, if the program is run in the text pad with DOS screen in the background, the record or data inserted into the database will show as shown in figure12 below.

5 Implementation

5.1 Introduction

This section will focus on the development of actual system having layout it out in the design stage, however there is possibility that a change erupt during this stage as changes at the implementation are sometime inevitable.

The stage will range from input design which will be of twofold, follow by the database schema design and coding as unarguably part of implementation.

5.2 Software choice

Software Choice for Airline Booking System Project

For an airline booking system project, the following software choices can be considered:

Frontend:

HTML/CSS: For building the user interface and user experience of the website.

JavaScript: For adding interactive elements and dynamic functionality to the website.

Backend:

.NET: A framework for building web applications, providing a robust and scalable architecture.

ASP.NET: A web application framework that allows for building dynamic web pages and web services.

Database:

MySQL: A relational database management system for storing and managing data.

Additional Tools:

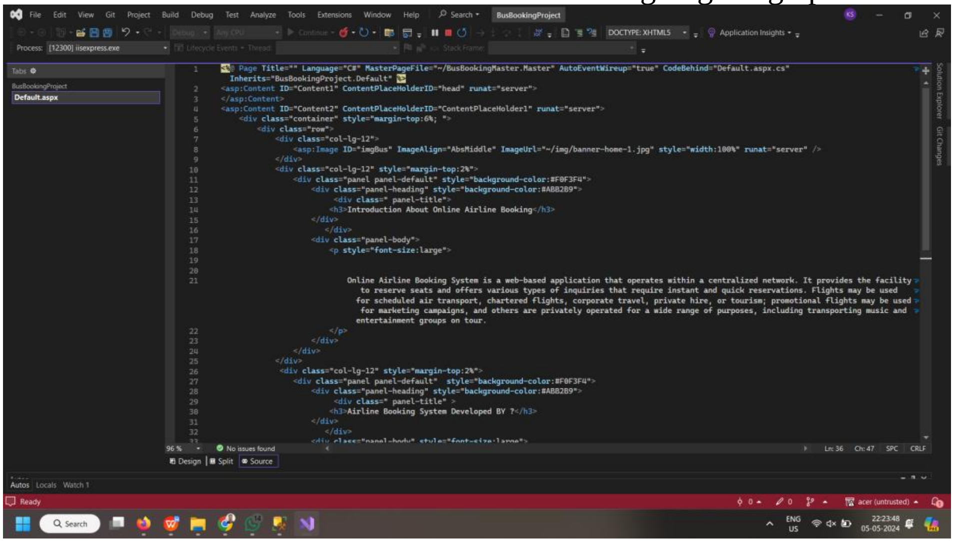
Visual Studio: An integrated development environment (IDE) for building, debugging, and testing the application.

Entity Framework: An object-relational mapping (ORM) framework for interacting with the database.

5.3 User interface (screen shots)

Using the appropriate and carefully chosen tools, the following are the screenshots of the user interface and this is achieved with the implementation of swing package in the java class. The swing is one of the important packages to import when designing user interface in java, just like importing SQL if SQL statements are to be written in .net . Each line ofthe code has import role to play e.g. the frame.setVisible (true) makes the whole frame visible to the user when ran it otherwise no event will be performed.

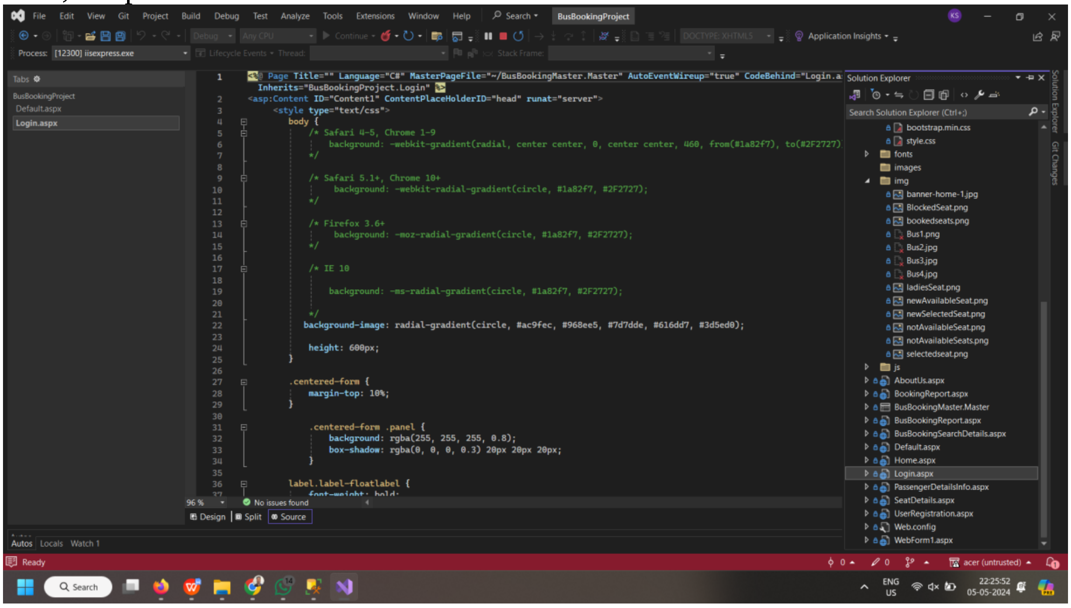
Below is the screen shot of the code used when designing the graphical user interface;



```
<%@ Page Title="" Language="C#" MasterPageFile="~/BusBookingMaster.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
  Inherits="BusBookingProject.Default" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="container" style="margin-top:6%; ">
        <div class="row">
            <div class="col-lg-12">
                <asp:Image ID="imgBus" ImageAlign="AbsMiddle" ImageUrl="~/img/banner-home-1.jpg" style="width:100%" runat="server" />
            </div>
            <div class="col-lg-12" style="margin-top:2%">
                <div class="panel panel-default" style="background-color:#F0F3F4">
                    <div class="panel-heading" style="background-color:#AB82B9">
                         <div class="  panel-title">
                        <h3>Introduction About Online Airline Booking</h3>
                    </div>
                        </div>
                    <div class="panel-body">
                        <p style="font-size:large">


                            Online Airline Booking System is a web-based application that operates within a centralized network. It provides the facility
                               to reserve seats and offers various types of inquiries that require instant and quick reservations. Flights may be used
                             for scheduled air transport, chartered flights, corporate travel, private hire, or tourism; promotional flights may be used
                               for marketing campaigns, and others are privately operated for a wide range of purposes, including transporting music and
                             entertainment groups on tour.
                        </p>
                    </div>
                </div>
            </div>
             <div class="col-lg-12" style="margin-top:2%">
                <div class="panel panel-default"  style="background-color:#F0F3F4">
                    <div class="panel-heading" style="background-color:#AB82B9">
                         <div class="  panel-title" >
                        <h3>Airline Booking System Developed BY ?</h3>
                    </div>
                        </div>
                    <div class="panel-body" style="font-size:large">
```

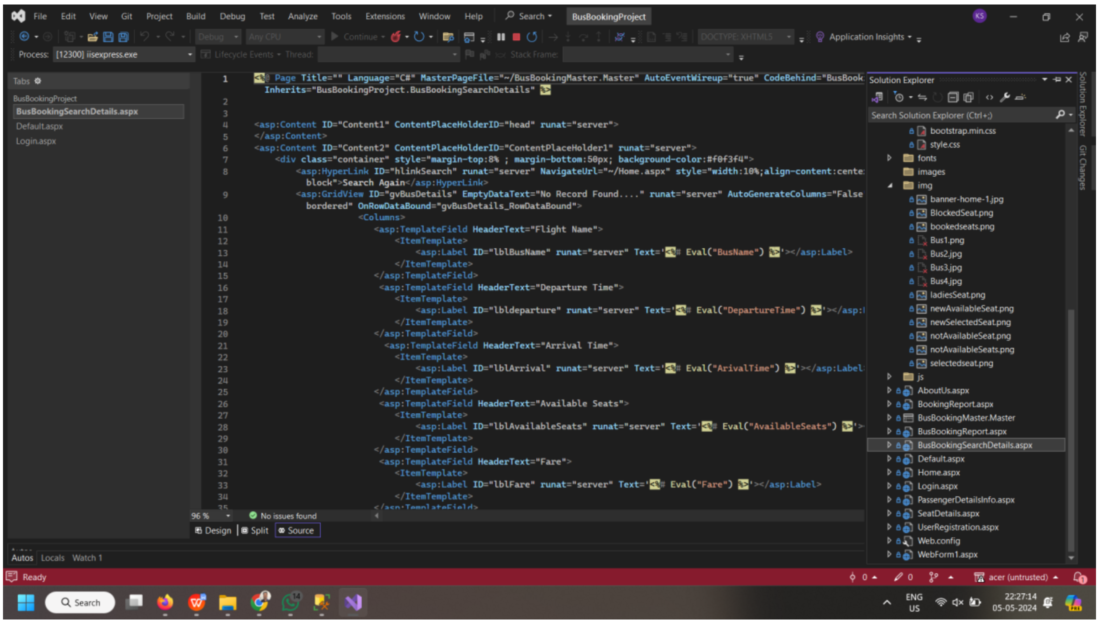
Figure 14 Customer Interface using .NET

The code details can be seen in the interface actual class where the buttons, labels, text field, are pined onto the frame



```
<%@ Page Title="" Language="C#" MasterPageFile="~/BusBookingMaster.Master" AutoEventWireup="true" CodeBehind="Login.a
  Inherits="BusBookingProject.Login" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
    <style type="text/css">
        body {
            /* Safari 4-5, Chrome 1-9
                background: -webkit-gradient(radial, center center, 0, center center, 460, from(#1a82f7), to(#2F2727)
            */

            /* Safari 5.1+, Chrome 10+
                 background: -webkit-radial-gradient(circle, #1a82f7, #2F2727);
            */

            /* Firefox 3.6+
                 background: -moz-radial-gradient(circle, #1a82f7, #2F2727);
            */

            /* IE 10

                background: -ms-radial-gradient(circle, #1a82f7, #2F2727);

            */
           background-image: radial-gradient(circle, #ac9fec, #968ee5, #7d7dde, #616dd7, #3d5ed0);

            height: 600px;
        }

        .centered-form {
            margin-top: 10%;
        }

            .centered-form .panel {
                background: rgba(255, 255, 255, 0.8);
                box-shadow: rgba(0, 0, 0, 0.3) 20px 20px 20px;
            }

        label.label-floatlabel {
            font-weight: bold;
```

Figure 15 Customer Record Input Interface



```
<%@ Page Title="" Language="C#" MasterPageFile="~/BusBookingMaster.Master" AutoEventWireup="true" CodeBehind="BusBook
  Inherits="BusBookingProject.BusBookingSearchDetails" %>


<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="container" style="margin-top:8% ; margin-bottom:50px; background-color:#f0f3f4">
        <asp:HyperLink ID="hlinkSearch" runat="server" NavigateUrl="~/Home.aspx" style="width:10%;align-content:cente
          block">Search Again</asp:HyperLink>
        <asp:GridView ID="gvBusDetails" EmptyDataText="No Record Found...." runat="server" AutoGenerateColumns="False
          bordered" OnRowDataBound="gvBusDetails_RowDataBound">
                    <Columns>
                       <asp:TemplateField HeaderText="Flight Name">
                           <ItemTemplate>
                               <asp:Label ID="lblBusName" runat="server" Text='<%# Eval("BusName") %>'></asp:Label>
                           </ItemTemplate>
                       </asp:TemplateField>
                        <asp:TemplateField HeaderText="Departure Time">
                           <ItemTemplate>
                               <asp:Label ID="lbldeparture" runat="server" Text='<%# Eval("DepartureTime") %>'></asp:
                           </ItemTemplate>
                       </asp:TemplateField>
                         <asp:TemplateField HeaderText="Arrival Time">
                           <ItemTemplate>
                               <asp:Label ID="lblArrival" runat="server" Text='<%# Eval("ArivalTime") %>'></asp:Label
                           </ItemTemplate>
                       </asp:TemplateField>
                        <asp:TemplateField HeaderText="Available Seats">
                           <ItemTemplate>
                               <asp:Label ID="lblAvailableSeats" runat="server" Text='<%# Eval("AvailableSeats") %>'>
                           </ItemTemplate>
                       </asp:TemplateField>
                        <asp:TemplateField HeaderText="Fare">
                           <ItemTemplate>
                               <asp:Label ID="lblFare" runat="server" Text='<%# Eval("Fare") %>'></asp:Label>
                           </ItemTemplate>
                       </asp:TemplateField>
```

Figure 16 Search Detail Interface

5.4 Database schema

Using the database schema (Figure 10) at the design stage as a prototype, helps is achieving effective actual database schema shown below.

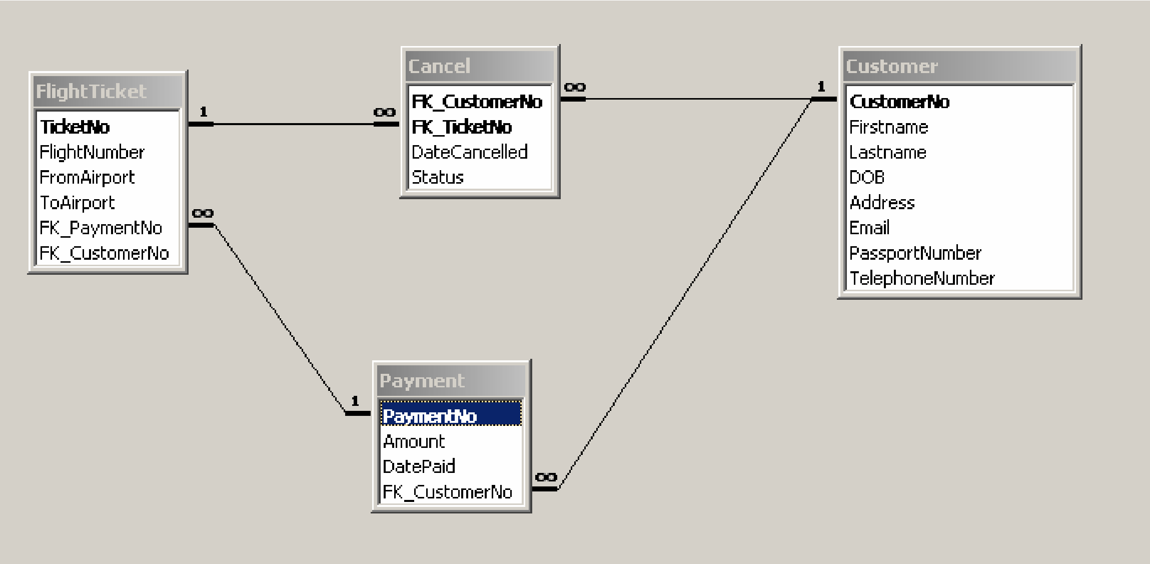
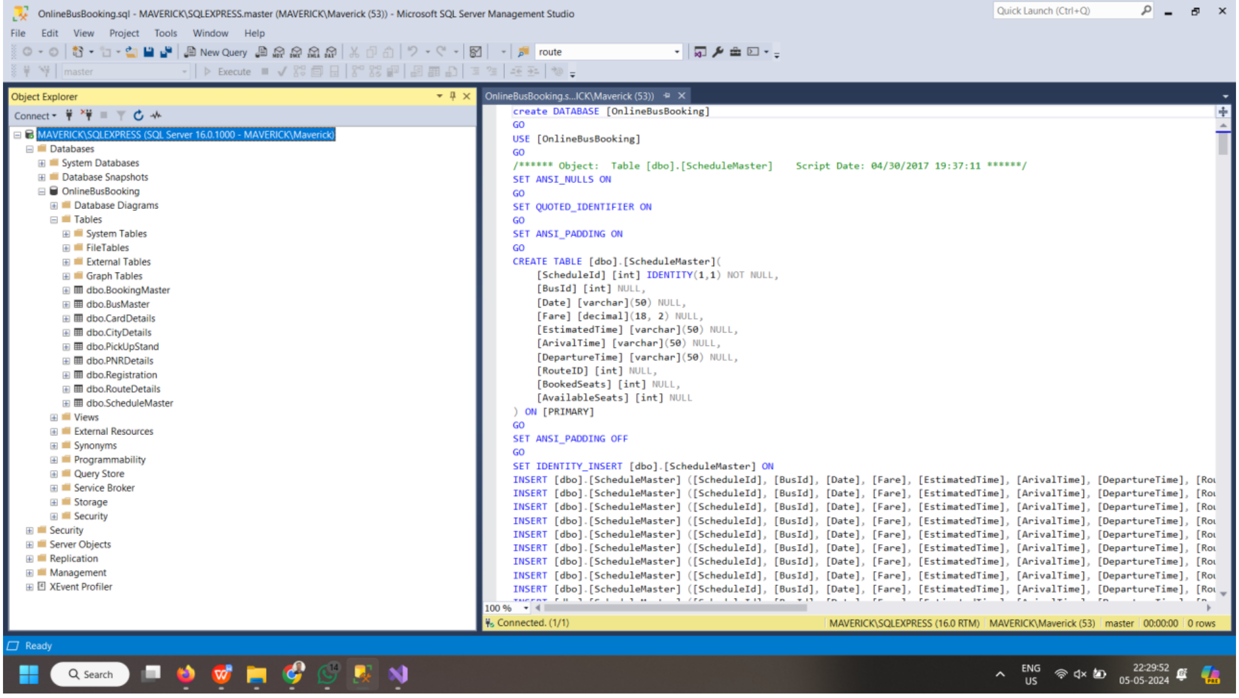


Figure 20 DB Schema screenshot

DBC Connection is created using Administrative tool in control panel.Screen shot showing the connection

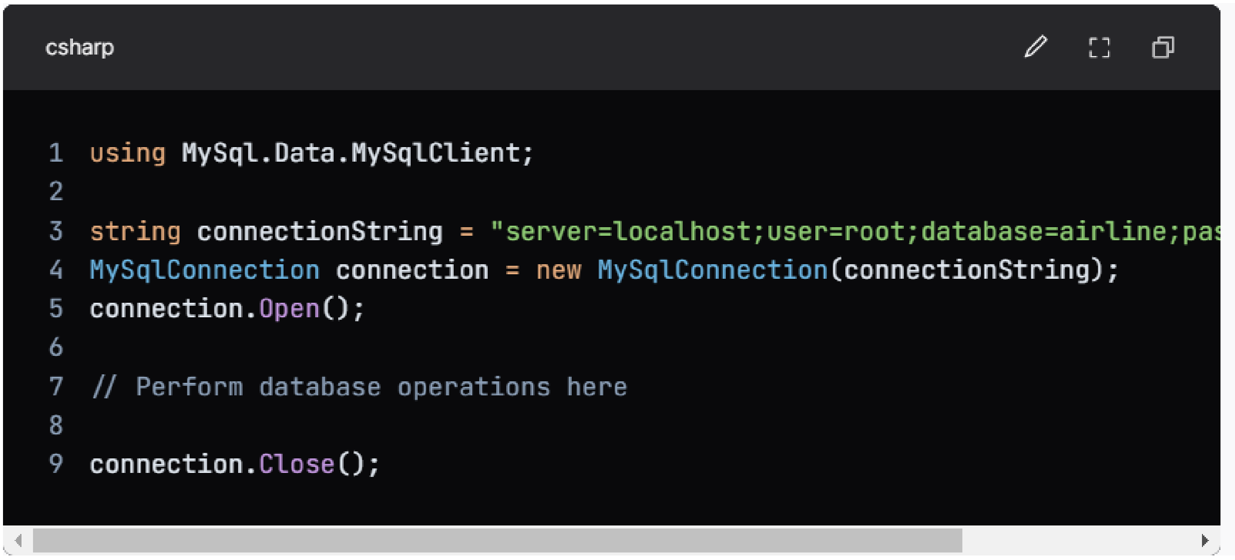


```sql
create DATABASE [OnlineBusBooking]
GO
USE [OnlineBusBooking]
GO
/****** Object:  Table [dbo].[ScheduleMaster]    Script Date: 04/30/2017 19:37:11 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[ScheduleMaster](
	[ScheduleId] [int] IDENTITY(1,1) NOT NULL,
	[BusId] [int] NULL,
	[Date] [varchar](50) NULL,
	[Fare] [decimal](18, 2) NULL,
	[EstimatedTime] [varchar](50) NULL,
	[ArivalTime] [varchar](50) NULL,
	[DepartureTime] [varchar](50) NULL,
	[RouteID] [int] NULL,
	[BookedSeats] [int] NULL,
	[AvailableSeats] [int] NULL
) ON [PRIMARY]
GO
SET ANSI_PADDING OFF
GO
SET IDENTITY_INSERT [dbo].[ScheduleMaster] ON
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
INSERT [dbo].[ScheduleMaster] ([ScheduleId], [BusId], [Date], [Fare], [EstimatedTime], [ArivalTime], [DepartureTime], [Rou
```

Figure 21 DBC DSN screenshot

5.5 Database Connection

To connect the .NET backend to the MySQL database, you can use the MySQL Connector/NET, which is a library that enables .NET applications to connect to MySQL databases. Here is an example of how to establish a connection to the database in C#:



```csharp
using MySql.Data.MySqlClient;

string connectionString = "server=localhost;user=root;database=airline;pas
MySqlConnection connection = new MySqlConnection(connectionString);
connection.Open();

// Perform database operations here

connection.Close();
```

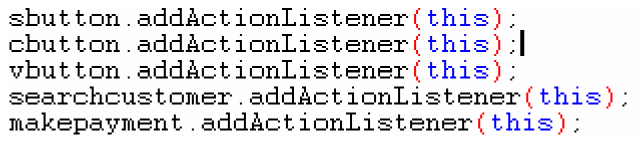
Referencing to the interface above, there are buttons, and textbox pined onto the interface and they are useless without the help of ActionListener () interface in Java class.

The ActionListerner interface uses actionperformed method which applies to the TextField, TextField, ComboBox and Button used in this project, with the If Statement whichcarries out the conditions and statements for each button pressed.

Below is the screen shot of “Save to Database” button and the customer record is inserted into the customer table using the SQL statement.

The absence of customer number is intentional because it is assigned automatically in the database which is set to be primary key in the table and also in payment table and Flight ticket table.

The following screen shot showing when the each button is added to the ActionListener Method, if by any mistake the button is not added to the method, no action will perform regales of complex codes or SQL queries.



```
sbutton.addActionListener(this);
cbutton.addActionListener(this);
vbutton.addActionListener(this);
searchcustomer.addActionListener(this);
makepayment.addActionListener(this);
```

Considering the time, the above codes are adopted in other classes (flight ticket, cancel and payment).

6 Testing

6.1 Introduction

This chapter is centred on checking if the system is doing what is suppose to be doing. The testing is crucial stage in any development and this project it will be inevitable but also necessary for checking system functionalities. The testing will be of twofold:

6.2 Test Plan:

To test the airline booking system project, the following test plan can be used:

6.3 Functional Testing:

Verify that users can search for flights based on various parameters (origin, destination, date, etc.)

Verify that users can select and book a flight, and that the booking is saved in the database

Verify that users can view their bookings and cancel or modify them if necessary

Verify that payment processing is working correctly and that bookings are marked as paid or unpaid accordingly

Verify that users receive confirmation emails after making a booking

6.4 Usability Testing:

Verify that the user interface is intuitive and easy to use

Verify that the layout and styling are consistent and visually appealing

Verify that the system is responsive and works well on different devices and screen sizes

Verify that error messages and prompts are clear and helpful

6.5 Performance Testing:

Verify that the system can handle a high volume of traffic and requests

Verify that the system responds quickly and efficiently, even with a large number of users

Verify that the system can handle multiple bookings and payments at the same time

6.6 Security Testing:

Verify that the system is secure and protects sensitive user data

Verify that the system is resistant to common attacks, such as SQL injection or cross-site scripting

Verify that the system uses secure protocols, such as HTTPS, for transmitting data

6.7 Compatibility Testing:

Verify that the system is compatible with different web browsers, such as Chrome, Firefox, and Safari

Verify that the system works well with different operating systems, such as Windows, macOS, and Linux

Verify that the system is compatible with different devices, such as desktop computers, laptops, and mobile phones.

7 Conclusion

7.1 Introduction

This chapter is focused on both the work carried out so far in this project and the system functionality evaluation.

7.2 Analysis and Design evaluation

An airline booking system is a complex and critical application that enables customers to search, select, and book flights online. The system should provide a user-friendly interface, fast and accurate search results, and secure payment processing.

The following are some key aspects to consider in the analysis of an airline booking system:

User interface design: The system should have an intuitive and easy-to-use interface that allows customers to quickly find and book flights.

Flight search: The system should provide fast and accurate search results based on various parameters, such as departure and arrival cities, dates, and flight times.

Booking process: The system should provide a streamlined and secure booking process that guides customers through the steps of selecting flights, entering passenger information, and making payments.

Payment processing: The system should support various payment methods, such as credit cards, debit cards, and online payment systems, and provide secure and reliable payment processing.

Customer management: The system should allow customers to manage their bookings, view their flight status, and access their travel history.

Data management: The system should provide robust data management capabilities, including data storage, retrieval, and analysis, to support business operations and decision-making.

7.2.1 Design stage

The design of an airline booking system should be based on the analysis of the system requirements and the user needs. The following are some key aspects to consider in the design evaluation of an airline booking system:

Usability: The system should be easy to use and navigate, with clear and concise instructions and feedback.

Performance: The system should provide fast and accurate search results and booking processing, with minimal downtime and errors.

Security: The system should provide secure payment processing and data management, with robust encryption and authentication mechanisms.

Scalability: The system should be scalable and flexible, with the ability to handle a large volume of traffic and support future growth and expansion.

Integration: The system should be able to integrate with other systems and services, such as payment gateways, airline reservation systems, and customer relationship management systems.

Testing: The system should be thoroughly tested and validated, with regular updates and maintenance to ensure optimal performance and security.

7.2.2 Possible Improvement

There many features that can be added to the system to make it better.

- Card payment system can be used instead of manual type. Also, the system can be linked to company website so that customer can access and use the system independently of the staff.

- Security; The system's security is not very good which is a loop for unauthorised access and this can be enhanced by incorporating authentication system with the system so that access or operation will base on user's permission and this will give

  the administrator to manage the system efficiently, effectively and less vulnerableto unauthorised access and system malfunctions.
- Communication; the business to customer communication channels can be improved via auto email for booking ticket and invoice confirmation. For instance, using Enterprise JavaBeans which is server side software but can be implemented to enhance this system to solve business related problems e.g. complex Database queries, pricing engine with a tax calculator for a website's shopping cart.
- Print; the system can be enhanced to be able to print records for filing purpose.

8 References

Books

1. Connolly, T and Begg, C, 2002, Database System (A practical Approach to Design,Implementation and Management), Third Edition, Prentice Hall.

2. Schildt, H, 2007, Java The Complete Reference, Seventh Edition (Osborne CompleteReference series), McGraw-Hill Osborne.

3. Hamilton, G, Cattell, R and Fisher, M, 1997, JDBC Database Access with Java,Addison Wesley Publishing Company.

Websites

1. Microsoft: http://technet.microsoft.com
2. Microsoft: http://msdn.microsoft.com
3. Oracle: http://www.oracle.com
4. Java: http://www.java.com