Sri Eshwar College of Engineering
An Autonomous Institution
Affiliated to Anna University, Chennai

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Academic Year

2023 – 2024 (Odd Semester)

MINI PROJECT REPORT

R19CS652 Database Management Systems

R19CS203 Object-Oriented Programming

------ Airline Reservation System -----

Submitted by,

[22EC014]	Avinath S
[22EC027]	Dinesh Kumar L
[22EC035]	Gokulnath D

Mentored by,

DR. Sreemathy J
Assistant Professor,
Department of CSE

DR.Rajesha Narasimha Murthy,
Associate professor,
Department of ECE

	TABLE OF CONTENTS		
CHAPTER	**TITLE**		**PAGE NO.**
	ABSTRACT		**2**
	LIST OF TABLES		
	LIST OF FIGURES		
1	**INTRODUCTION**		**3**
	1.1	OBJECTIVES	
	1.2	SCOPE OF THE PROJECT	**4**
2	**SYSTEM ANALYSIS AND SPECIFICATION**		**5**
	2.1	PROBLEM DESCRIPTION	
	2.2	FUNCTIONAL REQUIREMENT - HARDWARE & SOFTWARE	
	2.3	NON – FUNCTIONAL REQUIREMENT	
3	**SYSTEM DESIGN**		**6**
	3.1	ER DIAGRAM	
	3.2	SCHEMA DIAGRAM	
4	**PROPOSED SOLUTION**		**7**
	4.1	USER INTERFACE DESIGN	
	4.2	CLASS CONSTRUCTION	
	4.3	DATABASE CREATION	
5	**PROJECT DESCRIPTION**		**8**
	5.1	<<MODULE DESCRIPTION – Use your own Module Name>>	
		<<Add your module Description numbered like 5.1.1, 5.1.2, 5.1.3, etc >>	
	5.2	JDBC CONNECTIVITY	
6	**IMPLEMENTATION**		**9**
7	**RESULTS AND DISCUSSION**		**25**
8	**CONCLUSION & FUTURE ENHANCEMENT**		**27**
	REFERENCES		

ABSTRACT

The "JavaFX-Based Airline Reservation System with SQL Integration" project represents a comprehensive solution for modernizing and streamlining airline reservation processes. Utilizing JavaFX for the graphical user interface and integrating SQL for efficient data management, this system is designed to offer a user-friendly experience while ensuring secure and reliable storage of crucial information.

The graphical user interface, developed using JavaFX, provides an aesthetically pleasing and intuitive platform for users to interact with the reservation system. The modern design principles incorporated into the interface enhance the overall user experience, making it accessible and engaging for both customers and airline staff. The system facilitates easy navigation, allowing users to search for flights based on various parameters such as destination, date, and seating preferences.

Flight management is a key feature, enabling the efficient handling of flight details, including schedules, availability, and pricing. The system empowers customers to make, modify, or cancel reservations seamlessly, with the added convenience of generating electronic tickets. This ensures a secure and straightforward process for passengers to access their booking information.

One of the system's strengths lies in its integration of SQL for database management. SQL facilitates the creation of a robust database system, ensuring the secure storage and retrieval of data. This includes sensitive information such as user credentials and booking details. The utilization of SQL not only enhances data integrity but also enables the system to scale seamlessly as the volume of data and user interactions grow.

Additionally, the system provides analytical tools for generating reports on flight occupancy, revenue, and other key performance indicators. This feature empowers airline management with valuable insights, aiding in data-driven decision-making and operational optimization.

In conclusion, the "JavaFX-Based Airline Reservation System with SQL Integration" project amalgamates the strengths of JavaFX and SQL to deliver a comprehensive solution for airline reservation management. The system not only addresses the diverse needs of customers and airline staff but also provides a foundation for efficient, secure, and user-friendly processes in the dynamic domain of airline reservations.

CHAPTER 1
INTRODUCTION

1.1 OBJECTIVES

1. **User-Centric Interface:** Develop an intuitive and visually appealing user interface using JavaFX to enhance the overall user experience for customers and airline staff, ensuring ease of navigation and interaction.
2. **Efficient Flight Management:** Implement robust algorithms and data structures to manage flight details, schedules, availability, and pricing, providing a seamless and efficient booking process for users.
3. **Secure Data Management with SQL:** Integrate SQL for secure and reliable database management, ensuring the safe storage, retrieval, and manipulation of sensitive data such as user credentials, flight details, and reservations.
4. **Authentication and Access Control:** Implement stringent authentication mechanisms and role-based access control to safeguard the system, allowing only authorized personnel to access and modify critical functions.
5. **Analytics and Reporting:** Integrate analytical tools to generate insightful reports on flight occupancy, revenue, and key performance indicators, empowering airline management with data-driven insights for decision-making and operational optimization.

1.2SCOPE OF THE PROJECT

The scope of a development project on an Airline Reservation System (ARS) is quite extensive. It involves creating a user-friendly interface that facilitates easy communication between customers and the airline administration through software. The ARS is designed to automate the registration process of airlines, making it more efficient and accessible for users.

Here are some key points regarding the scope of our project:

1. Modernization: The ARS is a modern method that allows clients to access flight information without manual efforts.
2. Efficiency : It aims to solve the drawbacks of the manual system, such as time consumption and errors, by implementing programming languages like javaFX and database management systems like MySQL.
3. User Experience: The system provides details of flight schedules, costs, time, seats, and check-in details, enhancing the overall user experience.
4. Global Access : It serves as an interface for enabling reservations from any place.
5. Convenience: Customers can make reservations, cancel, or modify them without the need to visit airport.
6. Record Keeping: The ARS keeps track of passenger reservations, ticket data, and informs passengers promptly if there are changes in the flight schedule even flight delay.
7. Benefits : Special offers and discounts are provided for regular users.

The development of an ARS can significantly improve the efficiency and effectiveness of airline operations, providing a better experience for both customers and the airline staff.

CHAPTER 2
SYSTEM ANALYSIS AND SPECIFICATION

2.1 PROBLEM DESCRIPTION

Airline reservation systems play a crucial role in the efficient management of flight bookings, yet many existing systems face challenges related to usability, security, and scalability. The current systems may lack a modern, user-friendly interface, making it cumbersome for customers to navigate and book flights. Additionally, security concerns, especially regarding data breaches and unauthorized access, pose significant risks to sensitive information stored in the system. Moreover, some systems may struggle to handle increasing volumes of data and user interactions, limiting their scalability.This project aims to address these challenges by developing a JavaFX-based Airline Reservation System integrated with SQL

2.2 FUNCTIONAL REQUIREMENT –

SOFTWARE AND HARDWARE REQUIREMENT

The hardware required is good PC and Laptop.

Software required are Java compiler, MySQL, Scene Builder, JavaFx.

2.3 NON FUNCTIONAL REQUIREMENT

Performance:

- Rendering Speed: JavaFX should provide smooth rendering of graphical elements, and MongoDB queries should be optimized for efficient data retrieval and storage.
- Load Time: The JavaFX application should load within a reasonable time frame, and MongoDB queries should not cause significant delays in data.

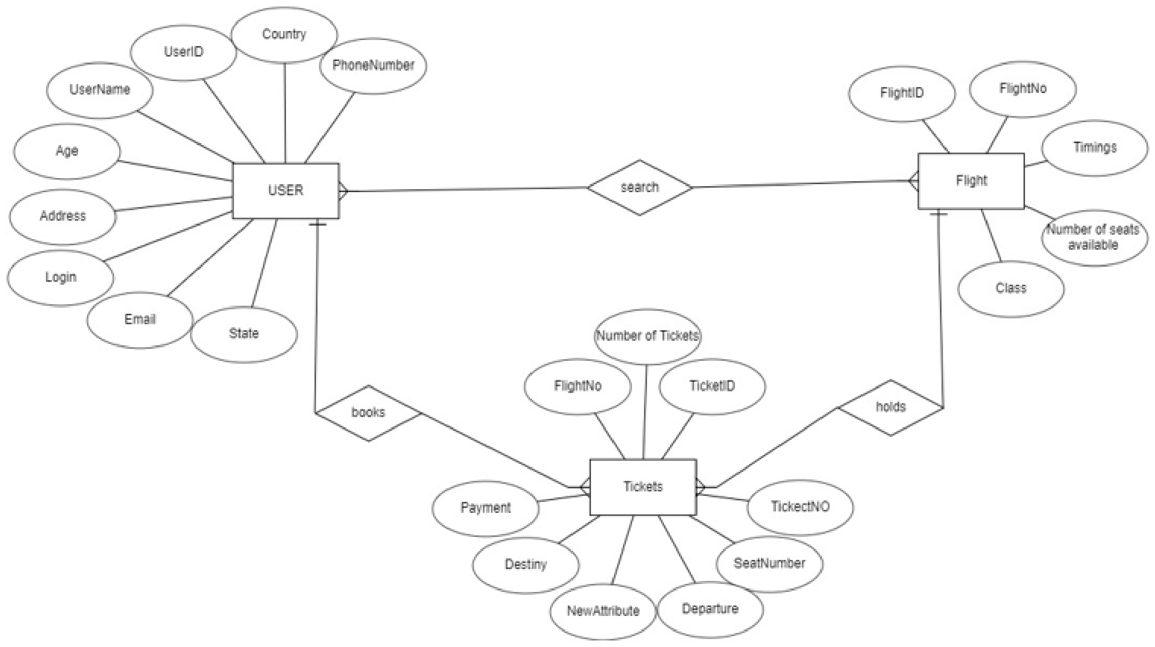
Usability:

- Intuitive User Interface: Design the JavaFX user interface to be intuitive and user-friendly, with a focus on ease of navigation and understanding.
- Query Language Familiarity: MySQL queries should follow a syntax that is familiar to developers and easily understandable.

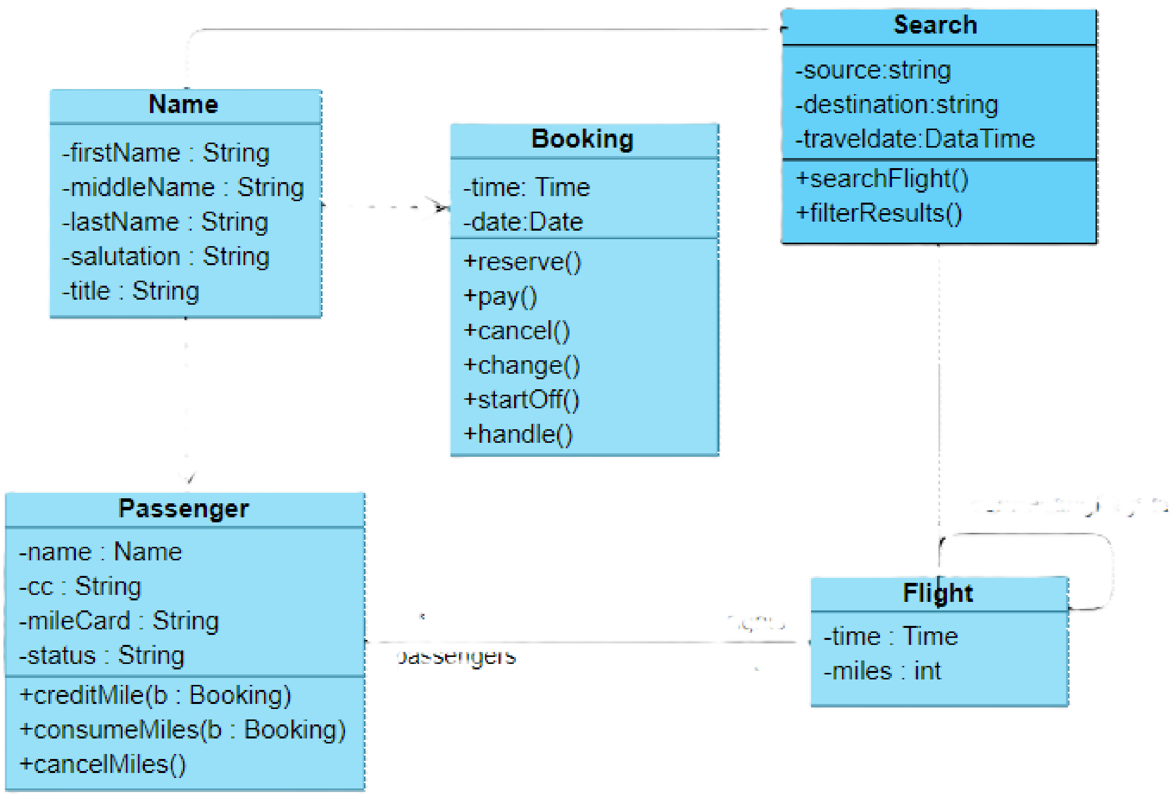
CHAPTER 3

SYSTEM DESIGN

3.1 ER DIAGRAM



3.2 SCHEMA DIAGRAM



CHAPTER 4

PROPOSED SOLUTION

4.1 USER INTERFACE DESIGN

- A login and Account creation page has created where the user exists login and use the application and the new user can create their account to enjoy the application.
- A enhanced page is create to search the flights available and the offer to countries.
- A simple page has created to book the selected flight and payment processing.

4.2 CLASS CONSTRUCTION

The different classes are created the do the seamless application

- An application class is created to handle the fxml pages
- The Sample controller class also created to do the actions on pages.

4.3 DATABASE CREATION

The database created are listed below:

- The user database is created to store the user details and user activities.
- The hotel database is created to maintain the hotel log and seats filling and much details required for airline bookings.
- The separate database to store booked seats and persons belongs to the bookings.

CHAPTER 5
PROJECT DESCRIPTION

5.1 MODULE DESCRIPTION

1. **User Interface (UI) Module:**

   module focuses on creating a visually appealing and user-friendly interface using JavaFX. It includes components for flight search, reservation, and management, providing an intuitive platform for both customers and airline staff to interact with the system.

2. **Flight Management Module:**

   Responsible for implementing algorithms and data structures to manage flight details. This module handles flight schedules, seat availability, and pricing, ensuring accurate and up-to-date information for users during the booking process.

3.Database Management Module (SQL Integration):

The SQL integration module focuses on creating and managing databases to store essential data. It includes tables for user profiles, flight details, reservations, and other relevant information. SQL queries are optimized for efficient data retrieval, insertion, and updates, ensuring database integrity and system performance.

5.2 JDBC CONNECTIVITY

1. **Import JDBC Libraries:** Import the necessary JDBC libraries into your Java project.
2. **Define Database Connection Parameters:** Specify database connection parameters like URL, username, and password.
3. **Establish Database Connection:** Use DriverManager.getConnection() to establish a connection to the database.
4. **Perform Database Operations:** Utilize the Connection object to create statements and execute queries.
5. **Handle Exceptions and Close Resources:** Implement error handling for robustness and close resources when they are no longer needed.

CHAPTER 6

IMPLEMENTATION

Code:

```
package application;

import java.io.IOException;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.scene.Node;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.ChoiceBox;

import javafx.scene.control.ContextMenu;

import javafx.scene.control.DatePicker;

import javafx.scene.control.Label;

import javafx.scene.control.MenuButton;

import javafx.scene.control.MenuItem;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.stage.Stage;


public class SampleController {

	@FXML
```

```
private Button btcreatenow;

        @FXML

        private Button btlogin;

        @FXML

        private PasswordField pass1;

        @FXML

        private TextField userid1;

        @FXML

        private Button btfinish;

        @FXML

        private DatePicker dob2;

        @FXML

        private Label Offer;

        @FXML

        private Label dest;

        @FXML

        private TextField count4;

        @FXML

        private Label passengercount;

        @FXML

        private Label passengercount1;

        @FXML

        private Label classmenu;

        @FXML

        private Label textbook;
```

```
@FXML
        private Label clicktext;
        @FXML
        private ContextMenu ticketmenu;
        @FXML
        private Label email5;
        @FXML
        private Label name3;
        @FXML
        private Label password8;
        @FXML
        private Label phone;
        @FXML
        private Button book;
        @FXML
        private MenuButton searchf;
        @FXML
        private TextField email2;
        @FXML
        private Label ldob;
        @FXML
        private Label lemail;
        @FXML
        private Label lname;
        @FXML
```

```
private Label lpass;

        @FXML

        private Label lphone;

        @FXML

        private TextField name2;

        @FXML

        private PasswordField pass2;

        @FXML

        private ChoiceBox<?> choicebox;

        @FXML

        private MenuButton choose;

        @FXML

        private Label messageeee;

        @FXML

        private TextField phone2;

        @FXML

        private MenuButton m;

        @FXML

        private MenuButton m1;

        @FXML

        private MenuButton m2;

        @FXML

        private MenuButton m3;


        public void initialize() {
                if (searchf != null) {
                        // Your initialization code here
                        this.searchf.getItems().forEach(menuItem -> {
                                menuItem.setOnAction(event ->
handleMenuItemSelection((MenuItem) event.getSource()));
                        });
                }
                if (choose != null) {
                        // Your initialization code here
                        this.choose.getItems().forEach(menuItem -> {
                                menuItem.setOnAction(event ->
handleMenuItemSelection((MenuItem) event.getSource()));
                        });
                }
        }
@FXML
        void Oncreatenow(ActionEvent event) throws IOException {
                Parent root = FXMLLoader.load(getClass().getResource("Create.fxml"));
                Stage stage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                Scene scene = new Scene(root);
                stage.setScene(scene);
                stage.show();
        }
@FXML
        void Onlogin(ActionEvent event) throws IOException {
                Parent root = FXMLLoader.load(getClass().getResource("searchflight.fxml"));
        Stage stage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                Scene scene = new Scene(root);
                stage.setScene(scene);
                stage.show();
                userid1.getText();
                pass1.getText();
        }


@FXML
        private void SearchflightonAction(ActionEvent event) {
                searchf.getText();
        }

        @FXML
        private void handleMenuItem(ActionEvent event) {
                MenuItem selectedMenuItem = (MenuItem) event.getSource();
                String selectedAction = selectedMenuItem.getText();
                System.out.println("Selected Actio    n: " + selectedAction);
        }

        @FXML
        void clickticketonAction(ActionEvent event) {
                choose.getText();
                count4.getText();
```

```
passengercount1.setText("Your ticket booked sucessfully.\nTo cancel contact us");
        }

        @FXML
        void btbook(ActionEvent event) throws IOException {
                Parent root = FXMLLoader.load(getClass().getResource("ticket.fxml"));
                Stage stage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                Scene scene = new Scene(root);
                stage.setScene(scene);
                stage.show();
        }
        private void handleMenuItemSelection(MenuItem selectedItem) {
                System.out.println("Selected menu item: " + selectedItem.getText());
        }
        public void Onfinish(ActionEvent event) throws IOException {
            // Other code...
                Parent root = FXMLLoader.load(getClass().getResource("searchflight.fxml"));
                Stage stage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                Scene scene = new Scene(root);
                stage.setScene(scene);
                stage.show();
                name2.getText();
                String dobText = dob2.getPromptText();
                email2.getText();
```

```
phone2.getText();
                    pass1.getText();
              String pass1Text = pass1 != null ? pass1.getText() : "";
              String pass2Text = pass2 != null ? pass2.getText() : "";
              if (pass1 != null) {
                  String password = pass1.getText();
              }
          }
}
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.DatePicker?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.text.Font?>
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" style="-fx-background-color:
violet;" xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="application.SampleController">
   <children>
      <Label layoutX="206.0" layoutY="33.0" text="Create your account here">
         <font>
```

```
<Font size="18.0" />
  </font>
</Label>
<Pane layoutX="13.0" layoutY="60.0" prefHeight="318.0" prefWidth="573.0">
  <children>
    <Label fx:id="lname" layoutX="59.0" layoutY="29.0" text="Name">
      <font>
        <Font size="14.0" />
      </font>
    </Label>
    <Label fx:id="ldob" layoutX="56.0" layoutY="71.0" text="Date of Birth">
      <font>
        <Font size="14.0" />
      </font>
    </Label>
    <Label fx:id="lemail" layoutX="60.0" layoutY="113.0" text="Email">
      <font>
        <Font size="14.0" />
      </font>
    </Label>
    <Label fx:id="lphone" layoutX="60.0" layoutY="159.0" text="Phone Number">
      <font>
        <Font size="14.0" />
      </font>
```

```
            </Label>

            <Label fx:id="lpass" layoutX="59.0" layoutY="202.0" text="Password">



       <font>

                  <Font size="14.0" />

                </font>

            </Label>

            <TextField fx:id="name2" layoutX="213.0" layoutY="27.0" />

            <DatePicker fx:id="dob2" layoutX="213.0" layoutY="68.0" />

            <TextField fx:id="email2" layoutX="213.0" layoutY="110.0" />

            <TextField fx:id="phone2" layoutX="213.0" layoutY="156.0" />

            <PasswordField fx:id="pass2" layoutX="213.0" layoutY="200.0" />

            <Button fx:id="btfinish" layoutX="261.0" layoutY="256.0"
mnemonicParsing="false" onAction="#Onfinish" text="Finish" />

         </children>

      </Pane>

   </children>

</AnchorPane>

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.DatePicker?>

<?import javafx.scene.control.Label?>

<?import javafx.scene.control.PasswordField?>

<?import javafx.scene.control.TextField?>

<?import javafx.scene.layout.AnchorPane?>

<?import javafx.scene.layout.Pane?>
```

```
<?import javafx.scene.text.Font?>



<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" style="-fx-background-color:
violet;" xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="application.SampleController">
   <children>
      <Label layoutX="206.0" layoutY="33.0" text="Create your account here">
         <font>
            <Font size="18.0" />
         </font>
      </Label>
      <Pane layoutX="13.0" layoutY="60.0" prefHeight="318.0" prefWidth="573.0">
         <children>
            <Label fx:id="lname" layoutX="59.0" layoutY="29.0" text="Name">
               <font>
                  <Font size="14.0" />
               </font>
            </Label>
            <Label fx:id="ldob" layoutX="56.0" layoutY="71.0" text="Date of Birth">
               <font>
                  <Font size="14.0" />
               </font>
            </Label>
            <Label fx:id="lemail" layoutX="60.0" layoutY="113.0" text="Email">
```

```
              <font>
                 <Font size="14.0" />
              </font>



  </Label>
            <Label fx:id="lphone" layoutX="60.0" layoutY="159.0" text="Phone Number">
              <font>
                 <Font size="14.0" />
              </font>
            </Label>
            <Label fx:id="lpass" layoutX="59.0" layoutY="202.0" text="Password">
              <font>
                 <Font size="14.0" />
              </font>
            </Label>
            <TextField fx:id="name2" layoutX="213.0" layoutY="27.0" />
            <DatePicker fx:id="dob2" layoutX="213.0" layoutY="68.0" />
            <TextField fx:id="email2" layoutX="213.0" layoutY="110.0" />
            <TextField fx:id="phone2" layoutX="213.0" layoutY="156.0" />
            <PasswordField fx:id="pass2" layoutX="213.0" layoutY="200.0" />
            <Button fx:id="btfinish" layoutX="261.0" layoutY="256.0"
mnemonicParsing="false" onAction="#Onfinish" text="Finish" />
         </children>
      </Pane>
   </children>
</AnchorPane>
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.Label?>

<?import javafx.scene.control.MenuButton?>


<?import javafx.scene.control.MenuItem?>

<?import javafx.scene.effect.Bloom?>

<?import javafx.scene.effect.SepiaTone?>

<?import javafx.scene.layout.AnchorPane?>

<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="368.0" prefWidth="739.0" style="-fx-background-color:
skyblue;" xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="application.SampleController">

  <children>

    <Label fx:id="Offer" layoutX="230.0" layoutY="68.0" text="25% Offer on pre-
booking">

      <font>

        <Font size="24.0" />

      </font>

      <effect>

        <Bloom>

          <input>

            <SepiaTone />

          </input>

        </Bloom>

      </effect>

    </Label>
```

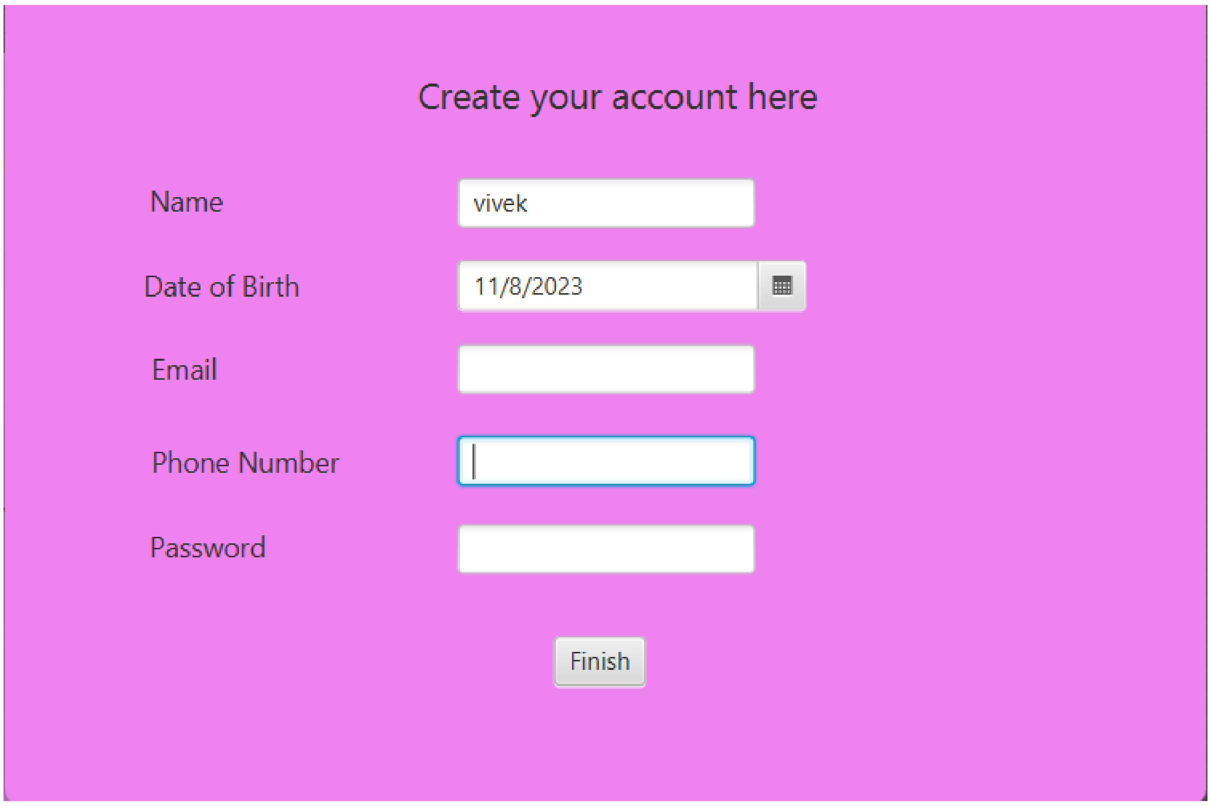
```
     <Label fx:id="dest" layoutX="188.0" layoutY="139.0" text="Enter your destination to
search flight available">

        <font>

           <Font size="18.0" />


</font>

     </Label>

     <MenuButton fx:id="searchf" layoutX="316.0" layoutY="193.0"
mnemonicParsing="false" onAction="#SearchflightonAction" text="Available Flights">

              <items>

                  <MenuItem onAction="#handleMenuItem" mnemonicParsing="false"
text="USA" />

                  <MenuItem onAction="#handleMenuItem" mnemonicParsing="false"
text="France" />

                   <MenuItem onAction="#handleMenuItem" mnemonicParsing="false"
text="Japan" />

              </items>

          </MenuButton>

     <Button fx:id="book" layoutX="334.0" layoutY="270.0" mnemonicParsing="false"
onAction="#btbook" text="Book Tickets" />

   </children>

</AnchorPane>

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>

<?import javafx.scene.control.Label?>

<?import javafx.scene.control.MenuButton?>

<?import javafx.scene.control.MenuItem?>

<?import javafx.scene.control.TextField?>
```

```
<?import javafx.scene.layout.AnchorPane?>

<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" style="-fx-background-color:


lightgreen;" xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="application.SampleController">

  <children>

    <Label fx:id="textbook" layoutX="208.0" layoutY="37.0" text="TICKET BOOKING"
textFill="#1d0dff">

      <font>

        <Font size="24.0" />

      </font>

    </Label>

    <Label fx:id="classmenu" layoutX="107.0" layoutY="119.0" text="Select the class you
prefer to travel">

      <font>

        <Font size="14.0" />

      </font>

    </Label>

    <Label fx:id="passengercount" layoutX="107.0" layoutY="163.0" text="Number of
tickets needed">

      <font>

        <Font size="14.0" />

      </font>

    </Label>

    <Label fx:id="passengercount1" layoutX="113.0" layoutY="245.0" prefHeight="106.0"
prefWidth="366.0" text="Number of tickets needed">
```

```
          <font>
            <Font size="14.0" />
          </font>
        </Label>


 <TextField fx:id="count4" layoutX="356.0" layoutY="159.0"
onAction="#clickticketonAction" promptText="count" />
        <Button fx:id="clickticket" layoutX="269.0" layoutY="200.0" mnemonicParsing="false"
onAction="#clickticketonAction" text="Click" />
        <MenuButton fx:id="choose" layoutX="356.0" layoutY="116.0"
mnemonicParsing="false" onAction="#clickticketonAction" text="choose"
textFill="#817979">
          <items>
            <MenuItem mnemonicParsing="false" text="Class 1" />
            <MenuItem mnemonicParsing="false" text="Class 2" />
            <MenuItem mnemonicParsing="false" text="Class 3" />
          </items>
        </MenuButton>
    </children>
</AnchorPane>
```

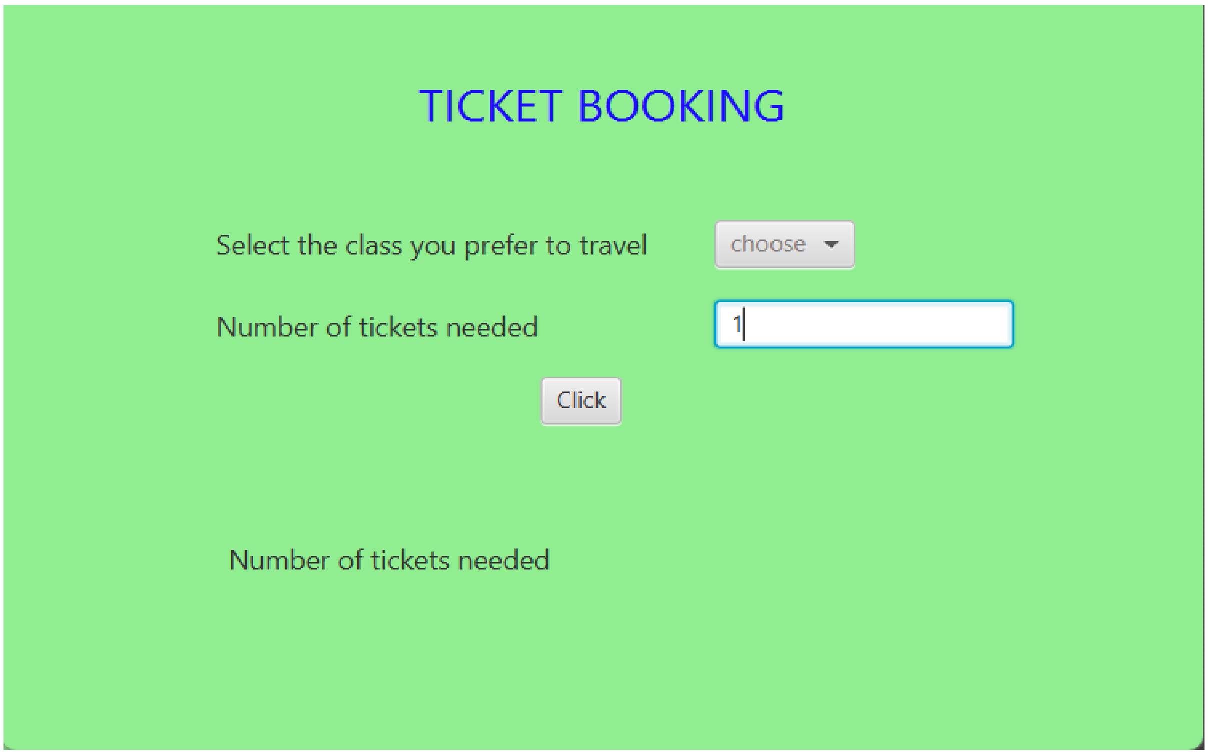
CHAPTER 7
RESULTS AND DISCUSSIONS

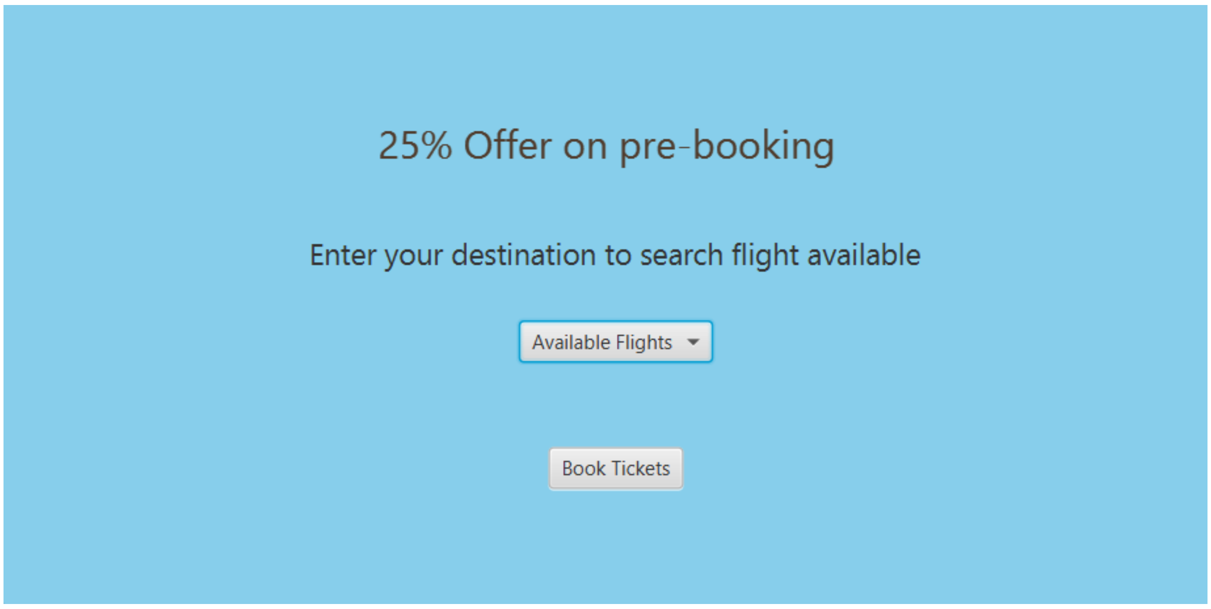
SCREENSHOTS:



Create your account here

Name	vivek
Date of Birth	11/8/2023
Email	
Phone Number	
Password	


TICKET BOOKING
Select the class you prefer to travel
choose
Number of tickets needed
1
Click
Number of tickets needed


25% Offer on pre-booking
Enter your destination to search flight available
Available Flights
Book Tickets

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

AI virtual assistants are evolving quickly. Companies are enabling them to provide more capabilities like speech recognition and natural language processing advances. It will enable them to understand and perform requests. Furthermore, improvements in voice recognition technology are allowing them to move deeper into business workflows. In the future, AI assistants will have more advanced cognitive computing technologies. These will help them carry out multi-step requests and perform more complex tasks.

REFERENCES

[1] https://youtu.be/C-KZO_dLr-A
[2] https://www.codewithharry.com/videos/python-tutorials-for-absolute-beginners-120
[3] https://github.com/Ram-Deepak/Natasha.git
[4] https://towardsdatascience.com/how-to-build-your-own-ai-personal-assistant-using-python-f57247b4494b?gi=847571c7aacd
[5] https://youtu.be/tSjR7bk1Y9U