Leadership & Excellence

Sri Eshwar College of Engineering

An Autonomous Institution

Affiliated to Anna University, Chennai

Accredited by NAAC with 'A' Grade

NATIONAL BOARD of ACCREDITATION For CSE, ECE, EEE, MECH

DEPARTMENT OF ELECTROINICS AND COMMUNICATION ENGINEERING

Academic Year

2023 – 2024 (Odd Semester)

MINI PROJECT REPORT

R19CS652 Database Management Systems

R19CS203 Object-Oriented Programming

------ Automated Food Ordering System (Swiggy) -----

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

ABSTRACT

The Automated Food Ordering System for Swiggy is a comprehensive project designed to revolutionize the food delivery industry by leveraging cutting-edge technology to enhance efficiency and user experience. Swiggy, a leading online food delivery platform, has witnessed unprecedented growth in recent years, necessitating the development of innovative solutions to meet the increasing demands of its user base.

This project focuses on automating key processes within Swiggy's food ordering system to streamline operations, reduce manual intervention, and improve overall service quality. The system encompasses various modules, including intelligent order processing, route optimization, and personalized recommendation engines.

The intelligent order processing module utilizes advanced algorithms to optimize order assignment, ensuring timely and efficient delivery. Real-time data analysis and machine learning techniques are employed to predict order demand, enabling proactive resource allocation and minimizing delivery times.

The route optimization module aims to enhance the delivery logistics by employing geospatial algorithms to find the most optimal routes for delivery executives. This not only reduces delivery times but also contributes to environmental sustainability by minimizing fuel consumption.

The personalized recommendation engine is designed to enhance the user experience by analyzing customer preferences and behavior patterns. Through collaborative filtering and deep learning algorithms, the system suggests personalized menus, promoting upselling and cross-selling while increasing customer satisfaction.

Furthermore, the Automated Food Ordering System integrates seamlessly with Swiggy's existing platform, ensuring a smooth transition and minimal disruption to current operations. The system is designed to be scalable, accommodating Swiggy's future growth and evolving technological landscape.

In conclusion, the Automated Food Ordering System for Swiggy represents a forward-thinking approach to enhance efficiency, reduce operational costs, and elevate the overall user experience. By embracing automation and leveraging advanced technologies, this project aims to position Swiggy as a leader in the competitive online food delivery market, catering to the dynamic needs of its diverse customer base.

CHAPTER 1

INTRODUCTION

1.1OBJECTIVES

- Implement intelligent order processing for optimal assignment, reducing processing times and minimizing errors.
- Develop automated workflows for order confirmation, dispatch, and delivery tracking to streamline the overall food delivery process.
- Utilize geospatial algorithms for route optimization, minimizing delivery times, and reducing fuel consumption.
- Implement real-time tracking and adaptive routing functionalities to address dynamic changes in traffic and delivery conditions.
- Integrate a personalized recommendation engine using machine learning techniques to enhance the overall ordering experience.
- Design a scalable architecture to accommodate the growing user base and increasing transaction volumes.

1.1 SCOPE OF THE PROJECT

- Implement intelligent algorithms for efficient and error-free order processing, from confirmation to delivery tracking.
- Utilize geospatial algorithms to optimize delivery routes, minimizing delivery times and reducing fuel consumption.
- Enhance user experience with a personalized recommendation engine using machine learning for menu suggestions.
- Design a scalable system architecture capable of handling the growing user base and increasing transaction volumes.

CHAPTER 2
SYSTEM ANALYSIS AND SPECIFICATION

2.1 PROBLEM DESCRIPTION

The rapid growth of the online food delivery industry, epitomized by platforms such as Swiggy, has introduced new challenges related to operational efficiency and user experience. As the number of users and restaurants on the platform continues to surge, traditional manual processes in food ordering and delivery are proving to be increasingly cumbersome and prone to inefficiencies. Several key problems warrant attention and innovative solutions to sustain and enhance Swiggy's market position.

2.2 FUNCTIONAL REQUIREMENT –

SOFTWARE AND HARDWARE REQUIREMENT

The hardware required is good PC and Laptop.

Software required are Java compiler, Mongo DB, 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 retrieval.

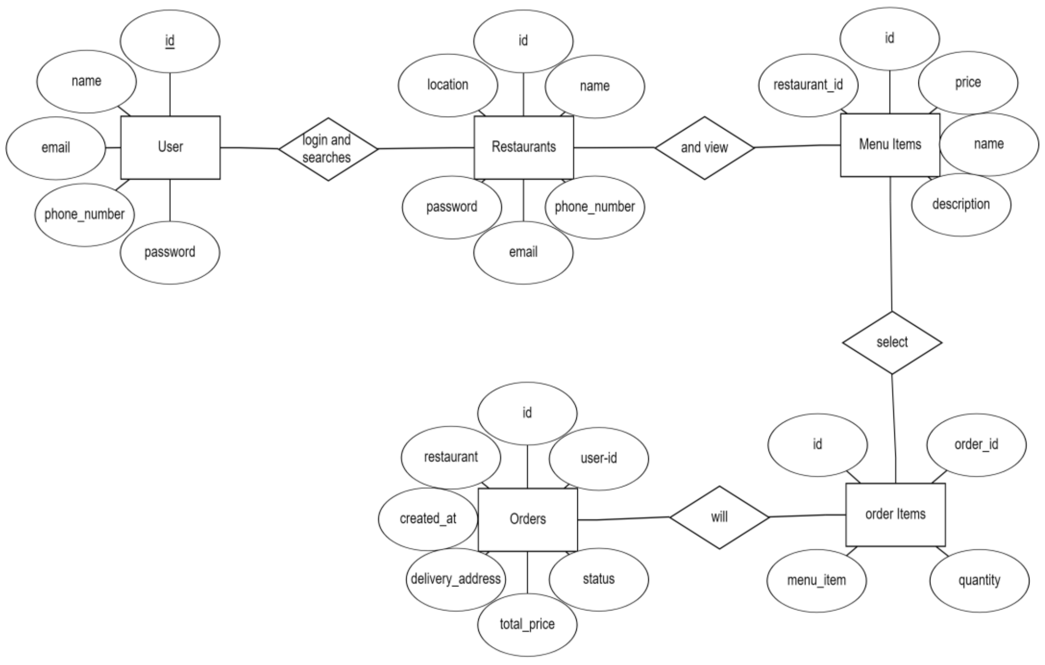
Usability:

- 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: MongoDB queries should follow a syntax that is familiar to developers and easily understandable.

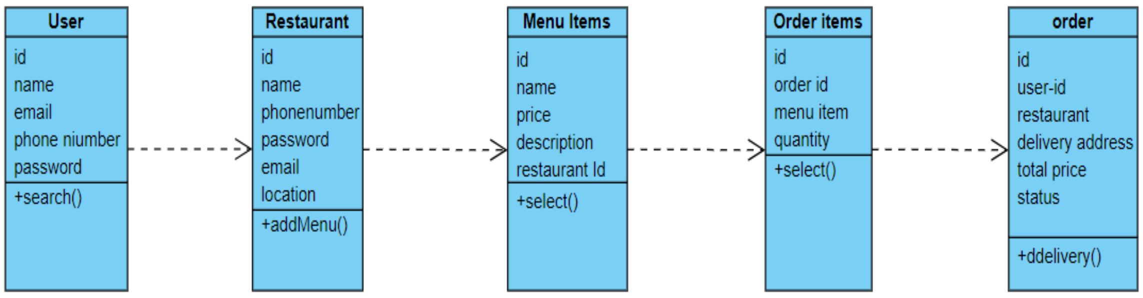
CHAPTER 3

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 hotels and restaurants and the most meals spots.
- A simple page has created to book the selected hotel 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 hotel bookings.
- The separate database to store booked seats and persons belongs to the bookings.

CHAPTER 5
PROJECT DESCRIPTION

5.1 MODULE DESCRIPTION

5.1.1 User Access:

- Objective: Allow secure user registration and login.
- Features: Account creation, login, and profile management.

5.1.2 Ordering:

- Objective: Let users browse, select items, and place orders.
- Features: Restaurant browsing, menu selection, and order placement.

5.1.3 Delivery Tracking:

- Objective: Facilitate delivery and provide tracking.
- Features: Assign deliveries, real-time tracking.

5.1.4 Database Integration:

- Objective: Use MongoDB for efficient data storage.
- Features: Designing collections, handling data.

5.2 JDBC CONNECTIVITY

To use the MongoDB Java Driver in your Java project, you'll need to include the MongoDB Java Driver dependency in your project's build configuration. If you are using a build tool like Maven or Gradle, you can add the dependency as follows:

```
<dependencies>
  <!-- MongoDB Java Driver -->
  <dependency>
    <groupId>org.mongodb</groupId>
    <artifactId>mongodb-driver-sync</artifactId>
    <version>4.4.3</version> <!-- Replace with the latest version
-->
  </dependency>
</dependencies>
```

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() {

        // Optional: Perform initialization tasks when the controller is created

        // You can access the MenuButton here

        // For example, set an event handler for menu item selection

        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">
```

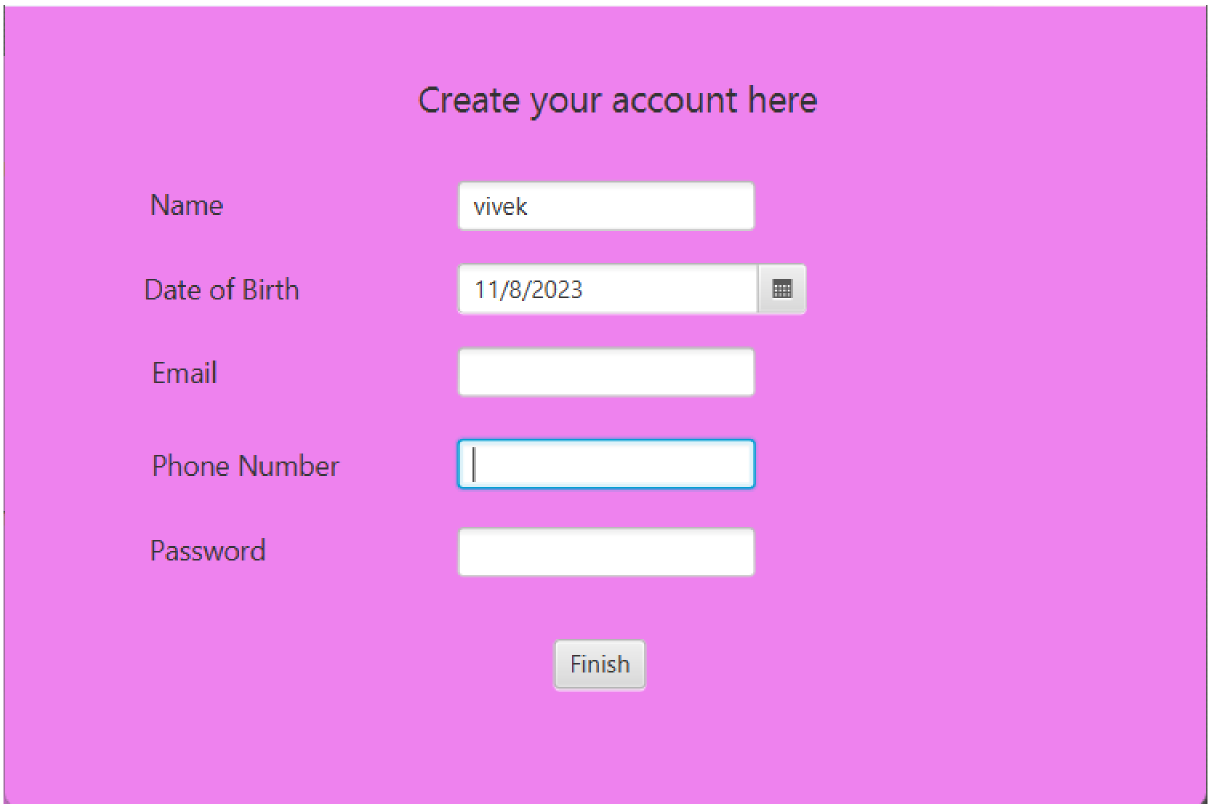
```
<children>
  <Label fx:id="lname" layoutX="59.0" layoutY="29.0" text="Name">
    <font>
      <Font size="14.0" />
    </font>
  </Label>
  <Label fx:id="ldob" layoutX="56.0" layoutY="71.0" text="Date of Birth">
    <font>
      <Font size="14.0" />
    </font>
  </Label>
  <Label fx:id="lemail" 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="AC" />

          <MenuItem mnemonicParsing="false" text="NON-AC" />

        </items>

      </MenuButton>

   </children>

</AnchorPane>
```

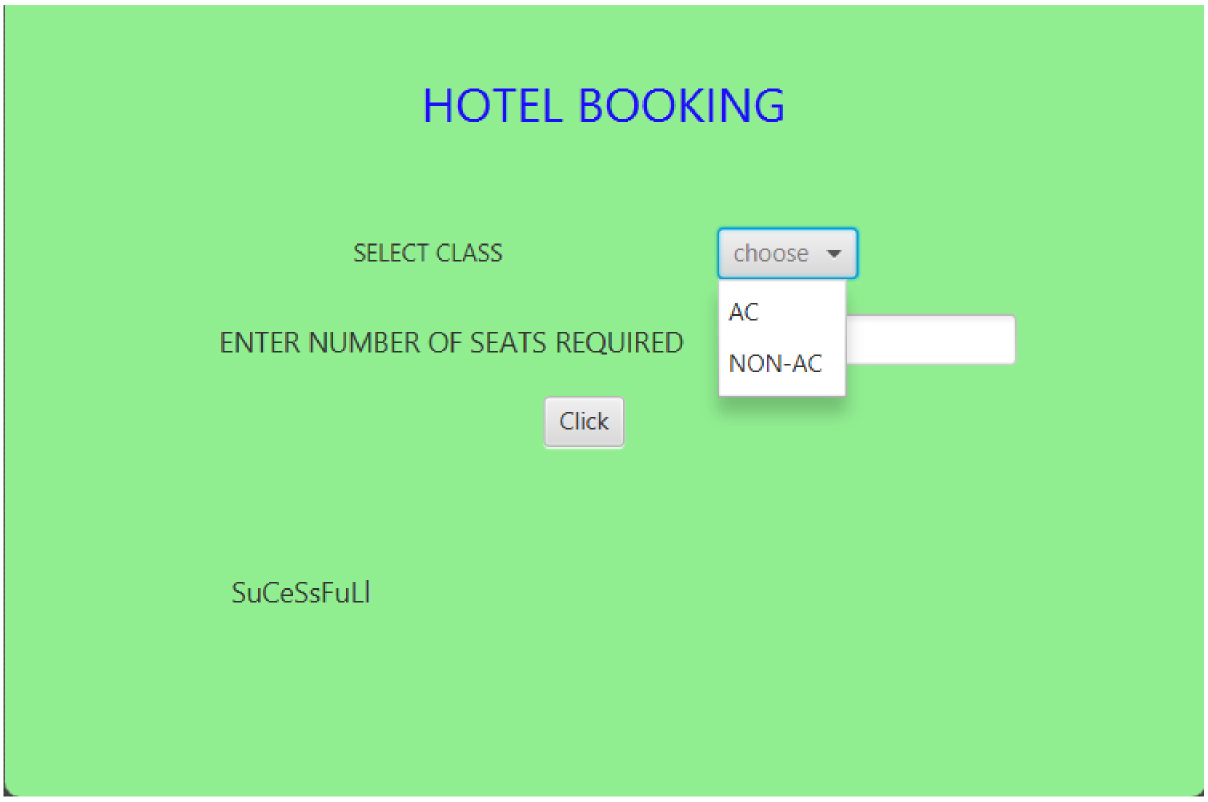
CHAPTER 7
RESULTS AND DISCUSSIONS

SCREENSHOTS:



Create your account here

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

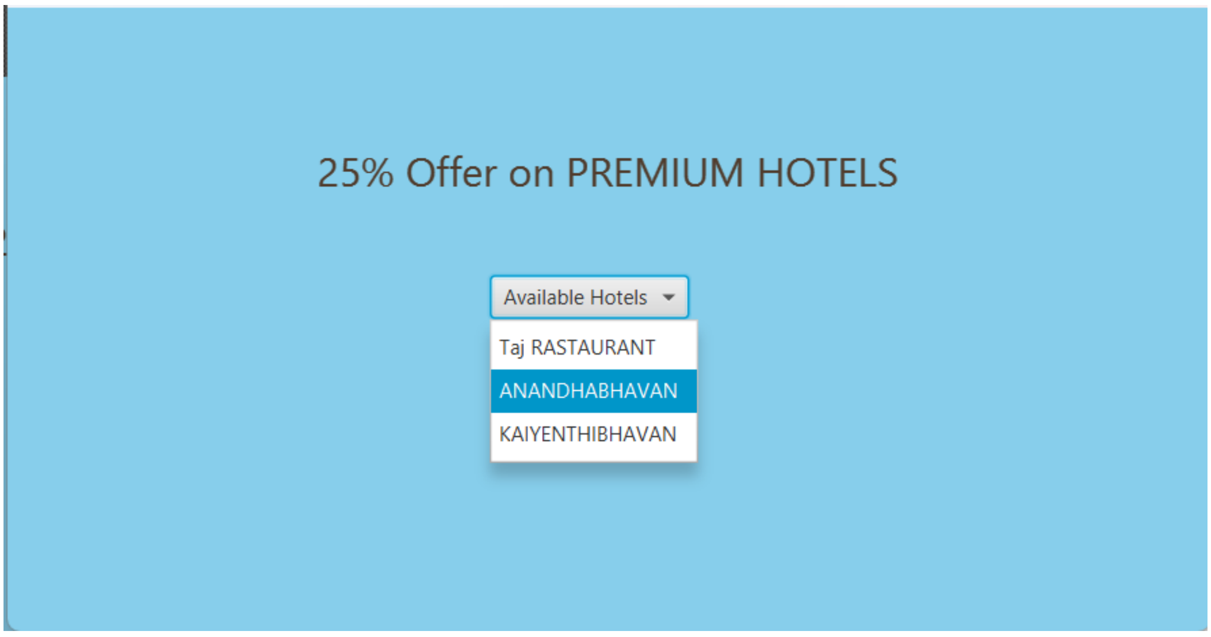


HOTEL BOOKING

SELECT CLASS

ENTER NUMBER OF SEATS REQUIRED

SuCeSsFuLl



25% Offer on PREMIUM HOTELS

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