FOOD DELIVERY MANAGEMENT SYSTEM

A MINI-PROJECT REPORT

Submitted by

SARVESH S	**240701479**
SHERIN KATHERINA D	**240701495**

in partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"Food-Delivery-Management-System"** is the bonafide work of **"SARVESH S, SHERIN KATHERINA D"** who carried out the project work under my supervision.

SIGNATURE

Dr. V. JANANEE

ASSISTANT PROFESSOR (SG)

Dept. of Computer Science and Engg,
Rajalakshmi Engineering College
Chennai

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

_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

In our state, the food service industry plays a major role in daily life. Although there are large-scale online platforms for food ordering, many **local restaurants** still lack an efficient and organized digital system to manage their operations. To address this gap, our team developed a **Restaurant Management System** — a database-driven application designed to streamline restaurant operations and improve coordination between customers, restaurants, and delivery agents.

The main objective of this project is to **simplify the process of order management and food delivery** within the local market. The system enables customers to place orders easily, restaurants to manage menus and monitor orders, and delivery agents to track and update delivery statuses. This organized approach allows local restaurants to **enhance efficiency, reduce delays, and compete effectively** with large-scale food delivery services.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson **DR. M.THANGAM MEGANATHAN** for their timely support and encouragement.We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.No words of gratitude will suffice for the unquestioning support extended to us by our Head Of The Department **Dr. E.M. MALATHY** and our Deputy Head Of The Department **Dr. J. MANORANJINI** for being ever supporting force during our project work.We also extend our sincere and hearty thanks to our internal guide **Dr. V. JANANEE ,** for her valuable guidance and motivation during the completion of this project. Our sincere thanks to our family members, friends and other staff members of computer science engineering.

1. SARVESH S

2. SHERIN KATHERINA D

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The **Restaurant Management System** simplifies food ordering, menu management, and delivery tracking. It enables customers to view dishes, place orders, and monitor delivery status, while restaurants can efficiently manage menus and orders. The admin oversees all system activities through a centralized interface.

1.2 SCOPE OF THE WORK

This system automates food ordering and delivery operations. It allows customers to browse menus, restaurants to update dishes, and delivery agents to track orders. The platform improves efficiency, saves time, and ensures transparency among all users. It is scalable for both small and large restaurants.

1.3 PROBLEM STATEMENT

Manual restaurant operations often lead to delays, errors, and poor coordination. Customers lack real-time order updates, and restaurants face challenges in managing orders and deliveries. This project resolves these issues by developing an automated, centralized system using **JavaFX** and **MySQL**.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The project aims to create a unified platform that connects customers, restaurants, delivery agents, and administrators within a single application. It focuses on simplifying the entire food ordering and delivery process while ensuring a seamless experience for all users. The system provides secure login and registration features, automates order tracking and delivery updates, and enables restaurants to efficiently manage menus and customer orders. Additionally, it allows administrators to monitor, control, and maintain the overall functionality of the system effectively.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

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

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 10
Front – End	:	JavaFX
Back - End	:	MySql-Wampserver
Language	:	Java,SQL

CHAPTER 3

MODULE DESCRIPTION

This application consists of **four major modules** — Customer, Restaurant, Delivery Agent, and Administrator. Each module has specific roles and permissions as described below.

1. Customer Module

Customers can register, log in, and browse available menu items from different restaurants. They can place orders, view their order history, and check the delivery status in real-time. The system automatically assigns a delivery agent to each order.

2. Restaurant Module

The restaurant user can log in and manage their menu by adding, updating, or deleting food items. They can also view all customer orders placed for their restaurant and track their progress until delivery.

3. Delivery Agent Module

Delivery agents can log in to view the list of assigned orders. They can update the delivery status from “Pending” to “Delivered” as the order progresses. This ensures that customers receive real-time updates on their order.

4. Admin Module

The administrator oversees all operations within the system. They have access to all user accounts, order details, and menu data. The admin can add or delete users, view overall reports, and ensure smooth coordination between customers, restaurants, and delivery agents.

CHAPTER 4

SAMPLE CODING

```
import javafx.application.Application;

import javafx.geometry.*;

import javafx.scene.Scene;

import javafx.scene.control.*;

import javafx.scene.layout.*;

import javafx.stage.Stage;

import java.sql.*;

public class Main extends Application {

private static final String URL = "jdbc:mysql://localhost:3306/";

private static final String USER = "root";

private static final String PASSWORD = "";

private Stage stage;

public static void main(String[] args) {

setupDatabase();

launch(args);

}

// ---------------- DATABASE SETUP ----------------

private static void setupDatabase() {
```

```
try {
Class.forName("com.mysql.cj.jdbc.Driver");
Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
Statement stmt = conn.createStatement();
stmt.executeUpdate("CREATE DATABASE IF NOT EXISTS Restaurant");
stmt.executeUpdate("USE Restaurant");
// Users
stmt.executeUpdate("""
CREATE TABLE IF NOT EXISTS Users (
user_id INT PRIMARY KEY AUTO_INCREMENT,
name VARCHAR(100),
email VARCHAR(100) UNIQUE,
password VARCHAR(100),
phone VARCHAR(15),
role ENUM('customer','restaurant','delivery','admin')
)
""");
// Menu
stmt.executeUpdate("""
CREATE TABLE IF NOT EXISTS Menu (
item_id INT PRIMARY KEY AUTO_INCREMENT,
```

```
item_name VARCHAR(100),

price DOUBLE,

restaurant_name VARCHAR(100)

)

""");

// Orders

stmt.executeUpdate("""

CREATE TABLE IF NOT EXISTS Orders (

order_id INT PRIMARY KEY AUTO_INCREMENT,

customer_name VARCHAR(100),

item_name VARCHAR(100),

status ENUM('Pending','Preparing','Out for Delivery','Delivered') DEFAULT 'Pending',

delivery_agent VARCHAR(100)

)

""");

System.out.println("✅ Database ready!");

conn.close();

} catch (Exception e) {

System.out.println("❌ Database setup failed");

e.printStackTrace();
```

```
}

}

// ---------------- START APP ----------------

@Override

public void start(Stage stage) {

this.stage = stage;

stage.setTitle("🍴 Restaurant Management System");

showLoginPage();

stage.show();

}

// ---------------- LOGIN PAGE ----------------

private void showLoginPage() {

Label lblTitle = new Label("🔐 Login");

lblTitle.setStyle("-fx-font-size: 22px; -fx-font-weight: bold; -fx-text-fill: white;");

TextField tfEmail = new TextField();

tfEmail.setPromptText("Enter Email");

PasswordField tfPass = new PasswordField();

tfPass.setPromptText("Enter Password");

Button btnLogin = new Button("Login");

Button btnRegister = new Button("Register");

Label lblStatus = new Label();
```

```
btnLogin.setStyle("-fx-background-color: #007bff; -fx-text-fill: white;");

btnRegister.setStyle("-fx-background-color: #28a745; -fx-text-fill: white;");

btnLogin.setOnAction(e -> {

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/Restaurant", USER, PASSWORD)) {

PreparedStatement ps = conn.prepareStatement("SELECT * FROM Users WHERE email=? AND password=?");

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

ps.setString(2, tfPass.getText());

ResultSet rs = ps.executeQuery();

if (rs.next()) {

String name = rs.getString("name");

String role = rs.getString("role");

switch (role) {

case "customer" -> showCustomerDashboard(name);

case "restaurant" -> showRestaurantDashboard(name);

case "admin" -> showAdminDashboard(name);

case "delivery" -> showDeliveryDashboard(name);

}

} else {

lblStatus.setText("❌ Invalid email or password!");
```

```
lblStatus.setStyle("-fx-text-fill: yellow;");

}

} catch (Exception ex) {

lblStatus.setText("❌ Database error!");

lblStatus.setStyle("-fx-text-fill: red;");

ex.printStackTrace();

}

});

btnRegister.setOnAction(e -> showRegisterPage());

VBox box = new VBox(12, lblTitle, tfEmail, tfPass, btnLogin, btnRegister, lblStatus);

box.setAlignment(Pos.CENTER);

box.setPadding(new Insets(20));

box.setStyle("-fx-background-color: linear-gradient(to bottom right, #0062E6, #33AEFF);");

stage.setScene(new Scene(box, 400, 350));

}

// ---------- REGISTER PAGE ----------

private void showRegisterPage() {

Label lblTitle = new Label("📝 Register New User");

lblTitle.setStyle("-fx-font-size: 20px; -fx-font-weight: bold; -fx-text-fill: white;");
```

```
TextField tfName = new TextField(); tfName.setPromptText("Name");

TextField tfEmail = new TextField(); tfEmail.setPromptText("Email");

PasswordField tfPass = new PasswordField(); tfPass.setPromptText("Password");

TextField tfPhone = new TextField(); tfPhone.setPromptText("Phone Number");

ComboBox<String> cbRole = new ComboBox<>();

cbRole.getItems().addAll("customer", "restaurant", "delivery", "admin");

cbRole.setPromptText("Select Role");

Button btnRegister = new Button("Register");

Button btnBack = new Button("← Back");

Label lblStatus = new Label();

btnRegister.setStyle("-fx-background-color: #20c997; -fx-text-fill: white;");

btnBack.setStyle("-fx-background-color: #ffc107; -fx-text-fill: black;");

btnRegister.setOnAction(e -> {

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/Restaurant", USER, PASSWORD)) {

String sql = "INSERT INTO Users (name, email, password, phone, role) VALUES (?,?,?,?,?)";

PreparedStatement ps = conn.prepareStatement(sql);

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

ps.setString(2, tfEmail.getText());

ps.setString(3, tfPass.getText());
```

```
ps.setString(4, tfPhone.getText());

ps.setString(5, cbRole.getValue());

ps.executeUpdate();

lblStatus.setText("✅ Registered Successfully!");

lblStatus.setStyle("-fx-text-fill: white;");

tfName.clear(); tfEmail.clear(); tfPass.clear(); tfPhone.clear(); cbRole.setValue(null);

} catch (SQLException ex) {

lblStatus.setText("❌ Error: " + ex.getMessage());

lblStatus.setStyle("-fx-text-fill: red;");

}

});

btnBack.setOnAction(e -> showLoginPage());

VBox box = new VBox(10, lblTitle, tfName, tfEmail, tfPass, tfPhone, cbRole, btnRegister, btnBack, lblStatus);

box.setAlignment(Pos.CENTER);

box.setPadding(new Insets(25));

box.setStyle("-fx-background-color: linear-gradient(to bottom right, #00C9A7, #92FE9D);");

stage.setScene(new Scene(box, 420, 450));

}

// ---------- CUSTOMER DASHBOARD ----------
```

```
private void showCustomerDashboard(String name) {

Label lbl = new Label("👋 Welcome " + name + " (Customer)");

lbl.setStyle("-fx-font-size: 20px; -fx-text-fill: white; -fx-font-weight: bold;");

ComboBox<String> cbItems = new ComboBox<>();

Label lblStatus = new Label();

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/Restaurant", USER, PASSWORD)) {

Statement st = conn.createStatement();

ResultSet rs = st.executeQuery("SELECT item_name, restaurant_name, price FROM Menu");

while (rs.next()) {

cbItems.getItems().add(rs.getString("item_name") + " - ₹" + rs.getDouble("price") + " (" + rs.getString("restaurant_name") + ")");

}

} catch (SQLException ex) {

lblStatus.setText("❌ Failed to load menu: " + ex.getMessage());

lblStatus.setStyle("-fx-text-fill: yellow;");

}

Button btnOrder = new Button("🛒 Place Order");

Button btnHistory = new Button("📜 View Orders");

Button logout = new Button("Logout");
```

```
TextArea txtOrders = new TextArea(); txtOrders.setPrefHeight(200); txtOrders.setEditable(false);

btnOrder.setOnAction(e -> {

if (cbItems.getValue() == null) {

lblStatus.setText("⚠️ Select an item first!");

lblStatus.setStyle("-fx-text-fill: yellow;");

return;

}

String item = cbItems.getValue().split(" - ")[0];

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/Restaurant", USER, PASSWORD)) {

// auto-assign delivery agent

Statement st = conn.createStatement();

ResultSet rs = st.executeQuery("SELECT name FROM Users WHERE role='delivery' ORDER BY RAND() LIMIT 1");

String agent = rs.next() ? rs.getString("name") : "Unassigned";

PreparedStatement ps = conn.prepareStatement("INSERT INTO Orders (customer_name, item_name, delivery_agent) VALUES (?,?,?)");

ps.setString(1, name);

ps.setString(2, item);

ps.setString(3, agent);
```

```
ps.executeUpdate();

lblStatus.setText("✅ Order placed! Assigned to: " + agent);

lblStatus.setStyle("-fx-text-fill: white;");

} catch (SQLException ex) {

lblStatus.setText("❌ " + ex.getMessage());

lblStatus.setStyle("-fx-text-fill: red;");

}

});

btnHistory.setOnAction(e -> {

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/Restaurant", USER, PASSWORD)) {

PreparedStatement ps = conn.prepareStatement("SELECT * FROM Orders WHERE customer_name=?");

ps.setString(1, name);

ResultSet rs = ps.executeQuery();

StringBuilder sb = new StringBuilder();

while (rs.next()) {

sb.append("#").append(rs.getInt("order_id")).append(" - ")

.append(rs.getString("item_name")).append(" [")

.append(rs.getString("status")).append("] (Agent: ")

.append(rs.getString("delivery_agent")).append(")\n");
```

```
}
txtOrders.setText(sb.toString());
} catch (SQLException ex) {
txtOrders.setText("❌ Error fetching orders");
}
});
logout.setOnAction(e -> showLoginPage());
VBox box = new VBox(10, lbl, cbItems, btnOrder, btnHistory, txtOrders, lblStatus, logout);
box.setAlignment(Pos.CENTER);
box.setPadding(new Insets(25));
box.setStyle("-fx-background-color: linear-gradient(to bottom right, #74ABE2, #5563DE);");
stage.setScene(new Scene(box, 480, 500));
}
```

CHAPTER 5

SCREENSHOTS

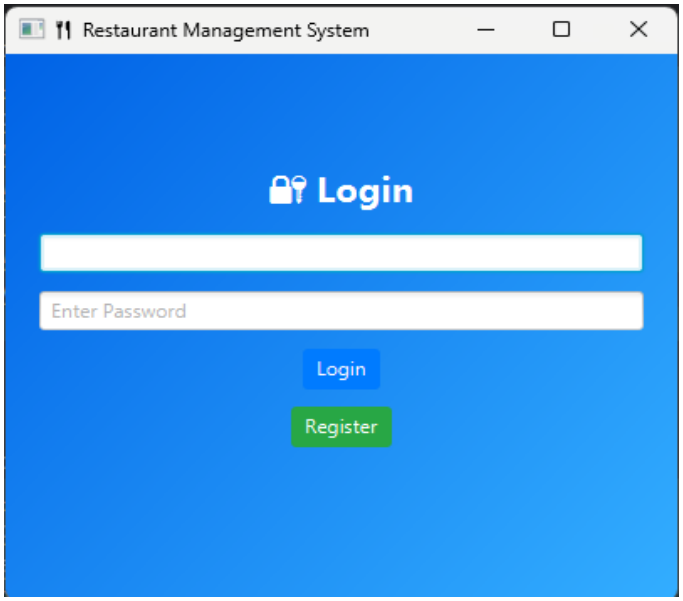


Fig 5.1 Login page

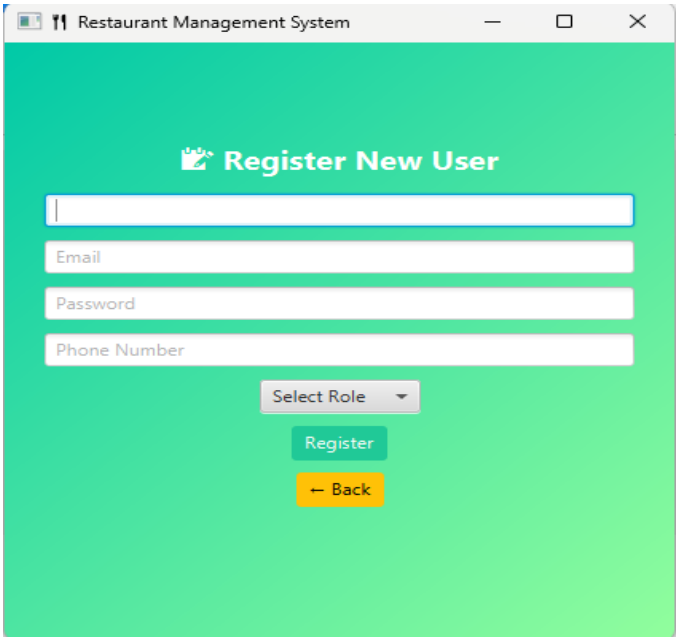


Fig 5.2 Registration Page

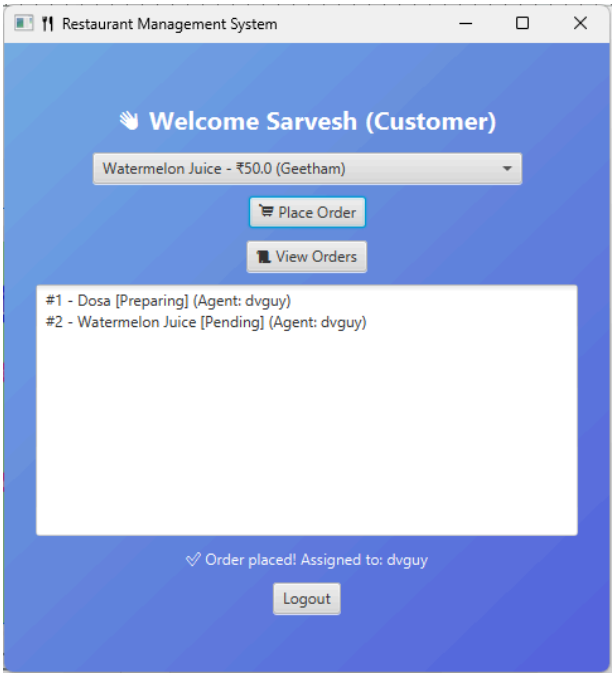


Fig 5.3 Customer Page

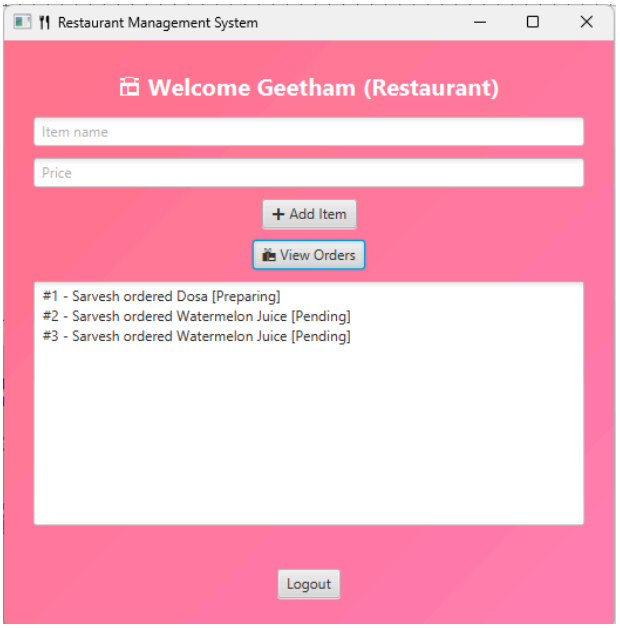


Fig 5.4 Restaurant page

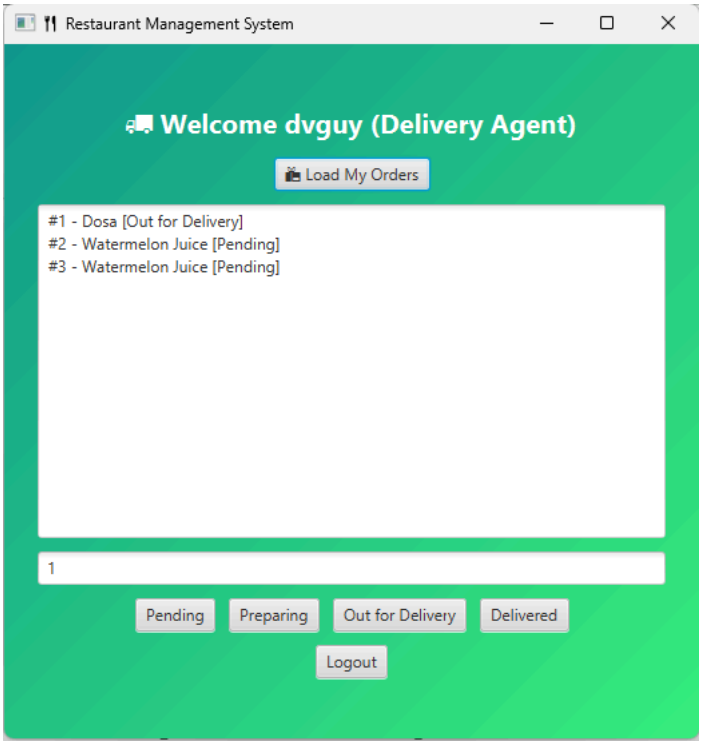


Fig 5.5 Delivery Agent Page

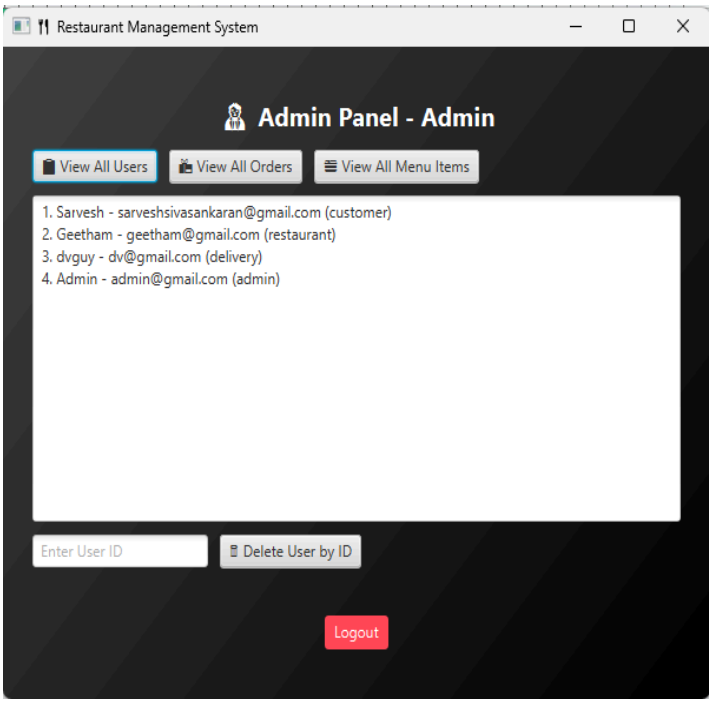
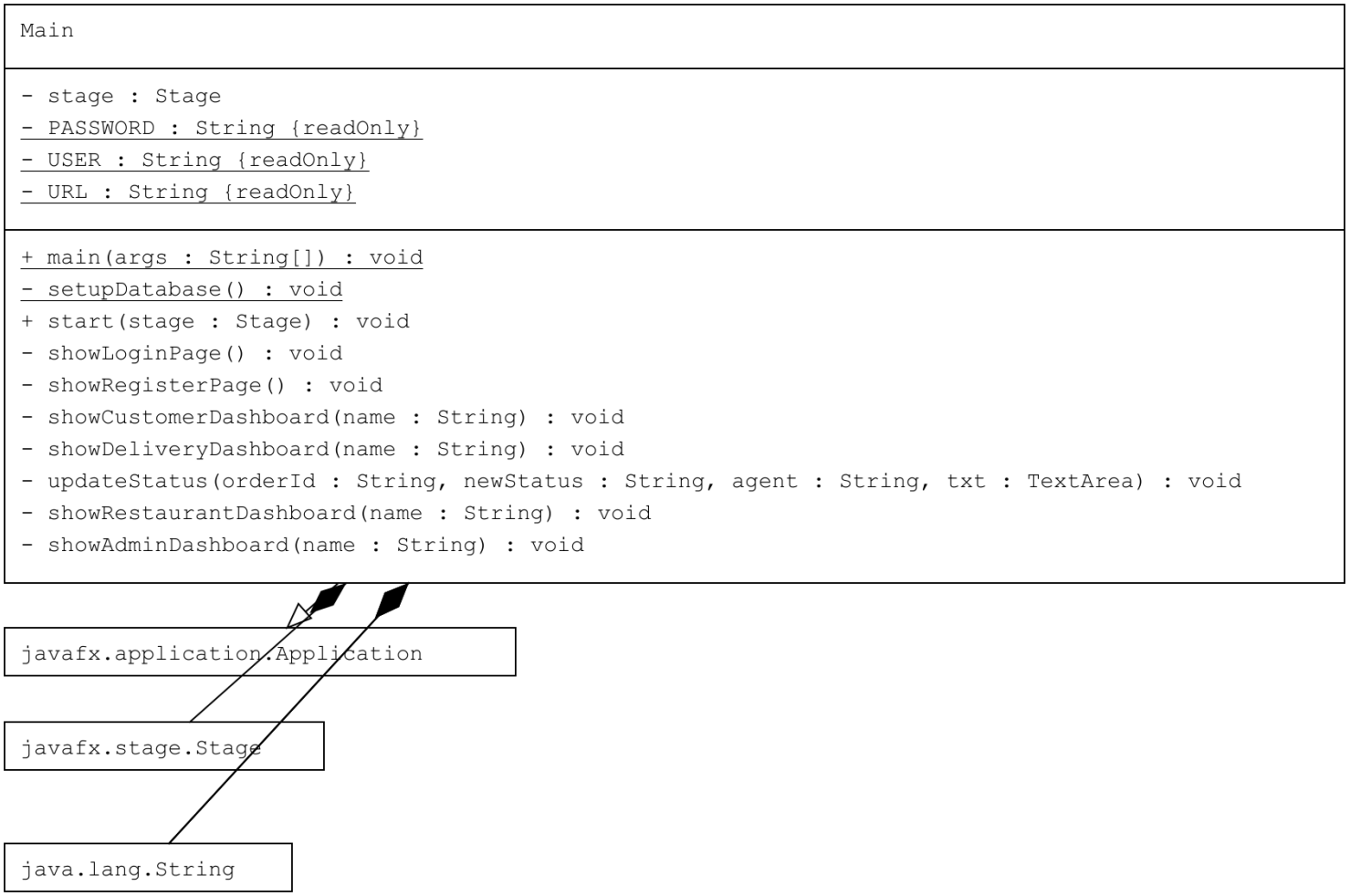


Fig 5.6 Admin Page

ER DIAGRAM


Main
- stage : Stage
- PASSWORD : String {readOnly}
- USER : String {readOnly}
- URL : String {readOnly}
+ main(args : String[]) : void
- setupDatabase() : void
+ start(stage : Stage) : void
- showLoginPage() : void
- showRegisterPage() : void
- showCustomerDashboard(name : String) : void
- showDeliveryDashboard(name : String) : void
- updateStatus(orderId : String, newStatus : String, agent : String, txt : TextArea) : void
- showRestaurantDashboard(name : String) : void
- showAdminDashboard(name : String) : void
javafx.application.Application
javafx.stage.Stage
java.lang.String

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

The **Restaurant Management System** successfully automates the food ordering and delivery process using an interactive and user-friendly interface built with **JavaFX** and **MySQL**. The system ensures data consistency, improves efficiency, and bridges the gap between restaurants and customers. It offers secure authentication, real-time updates, and role-based functionalities that streamline restaurant operations.

In the future, the system can be enhanced by:

- Integrating **online payment gateways** for secure transactions.
- Implementing **real-time delivery tracking** using GPS APIs.
- Providing a **mobile app version** for Android and iOS.
- Adding **AI-based recommendations** for customers based on previous orders.
- Enabling **multi-branch restaurant support** with centralized analytics.

REFERENCES

1. https://openjfx.io/
2. https://www.geeksforgeeks.org/java/javafx-tutorial/
3. Designing Data-Intensive Applications By-Martin Kleppmann
4. Database Internals: A deep-dive into how distributed data systems work By-Alex petrov
5. Database Systems Implementation By-Hector Gracia-Molina,Jeffrey D. Ullman,Jennifer widom