FOOD DISPATCH SYSTEM

A MINI-PROJECT REPORT

Submitted by

RAHUL G	**241901087**
VISHAL SK	**241901128**

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 DISPATCH SYSTEM"** is the bonafide work of **"VISHAL SK , RAHUL G"**
who carried out the project work under my supervision.

SIGNATURE

Ms.RUPMALA

ASSISTANT PROFESSOR

Dept. of Computer Science and
Engeneering (Cyber security),
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 city, food delivery plays a major role. Even though there are various multinational companies like Swiggy and Zomato, the local restaurant market is often bereft of a simple, direct-to-consumer application system. To address this gap, our team developed a database-driven application to help local restaurants maintain their menu data and organize orders efficiently.

The main objective of this project is to provide a complete, end-to-end user experience. This includes user registration and login for customer authentication. Once logged in, a customer can select a local restaurant, browse their complete menu, add items to a cart, and place a confirmed order. This system helps maintain restaurant menus, customer data, and order history in a centralized database, allowing local restaurants to compete by providing an efficient, direct service.

ACKNOWLEDGEMENT

We like to convey our sincere appreciation to all who have supported and mentored us during the successful completion of this project work.

We express our profound gratitude to **Mr. Benedict J.N., Associate Professor (SG**) and Head of the Department of Computer Science Engineering and Cyber Security at Rajalakshmi Engineering College, for furnishing us with essential resources, support, and an enabling environment to execute this project.

We express our profound gratitude to **Ms. R. Rupmala**, Assistant Professor in the Department of Computer Science Engineering and Cyber Security at Rajalakshmi Engineering College, for her unwavering support, invaluable insights, and collaboration throughout our endeavor.

We would like to convey our gratitude to our faculty members and colleagues for their valuable feedback and encouragement.

We express our gratitude to our families and friends for their steadfast support, patience, and encouragement, which were instrumental in the effective execution of this seminar.

1. Rahul G

2. Vishal SK

LIST OF FIGURES

CHAPTER 1

1.1 INTRODUCTION

The project provides a platform for customers to find local restaurants and order food directly. It begins by requiring users to authenticate themselves through a secure login or registration system. Once validated, the user can access a list of restaurants, browse menus, and place orders, with all data being fetched from and sent to a central MySQL database.

1.2 SCOPE OF THE WORK

The food dispatch system will help people access local restaurants directly. It creates a complete user workflow, starting from account creation and login to final order placement, bypassing the high fees of third-party food aggregator services.

1.3 PROBLEM STATEMENT

The need for this project arises as many corporate food aggregators have dominated the market. Their high commission rates and complex systems are not always ideal for small, local restaurants. Furthermore, these platforms lack a direct relationship between the customer and the restaurant. This creates a need for a simple, direct, and low-cost platform where users can create accounts and order directly from their favorite local eateries.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The main objective of this project is to allow customers to create an account, log in, easily select a restaurant, build an order, and submit it directly to the restaurant's database.

This system helps to:

Manage user accounts, including secure registration and login.

Maintain a centralized database of restaurants and their menus.

Provide a clean, user-friendly JavaFX interface for customers.

Allow logged-in users to add/remove items from a cart that calculates the total cost automatically.

Submit the finalized order and link it to the customer's account in the database.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel i5 8GB (Minimum)
Memory Size	:	8 GB (Minimum)
HDD	:	500 GB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 10
Front – End	:	Java FX (from org.openfx library
Back - End	:	MySql (Version 8.0 +)
Language	:	Java ,SQL
Runtime.	:	JDK 17
Build Tool.	:	Apache maven
IDE	:	IntelliJ IDEA

CHAPTER 3

MODULE DESCRIPTION

This application consists of several interconnected modules that form a complete user workflow.

1. User Authentication (Login & Register)
This is the new entry point of the application.
Login: The user provides a username and password. The DatabaseConnector queries the users table to verify the credentials. If successful, the application transitions to the Restaurant Selection module.

Register: A new user can enter a username and password. The system checks if the username already exists. If not, it hashes the password and inserts the new user record into the users table.

2. Restaurant Selection (createRestaurantListScene)
After a successful login, this module is displayed. It uses the DatabaseConnector to fetch a list of all restaurants from the restaurants table. These are displayed in a JavaFX ListView. When a user clicks a restaurant, the application transitions to the Menu module.

3. Menu Browsing (createMenuScene)
This module displays the menu for the specific restaurant selected. It fetches data from the menuitems table based on the restaurant_id.Each menu item is displayed in a ListView with a custom ListCell factory.Each cell contains the item's name, price, and an "Add" button.A StringBinding automatically updates a "Total" label whenever the cart's total value changes.

4. Cart Management (createCartScene) This module displays the contents of the shoppingCart.

A TableView is used to show the item name, price, and quantity.A custom TableCell factory adds a "Remove" button to each row.A "Proceed to Payment" button is bound to the cart's state and is disabled if the cart is empty.

5. Payment and Order Submission (createPaymentScene)
This module simulates the final checkout process.
It displays the final total price from the cart.It provides RadioButton options for payment (Credit Card, PayPal, Google Pay).
When the user clicks "Confirm Payment," the application calls the dbConnector.submitOrder(shoppingCart) method.

6. Backend Transaction (DatabaseConnector.java)
This is the core backend module. In addition to getRestaurants and getMenuItems, it now handles user authentication and order submission.registerUser(user, pass): Securely inserts a new user.
loginUser(user, pass): Validates user credentials.
submitOrder(cart): Performs a database transaction to save the order, linking it to the logged-in user's ID.

CHAPTER 4

SAMPLE CODING

The application is built in Java and uses JavaFX for the frontend and JDBC for the backend.

Sample 1: DatabaseConnector.java (User Login Logic) This snippet shows a conceptual loginUser method. It uses a PreparedStatement to safely query the users table and check if the provided username and password are valid. (Note: Passwords should be hashed in a real application).

```
/*
 * Assumes a new 'users' table exists with
 * user_id (INT), username (VARCHAR), password_hash (VARCHAR)
 */

public boolean loginUser(String username, String password) {

    // In a real app, 'password' would be hashed first,
    // and the query would check against the stored hash.
    // For simplicity here, we assume plaintext (WHICH IS INSECURE).

    String sql = "SELECT * FROM users WHERE username = ? AND password_hash =
?";

    try (Connection conn = getConnection();
         PreparedStatement pstmt = conn.prepareStatement(sql)) {

        pstmt.setString(1, username);
        pstmt.setString(2, password);    // Should be hashOf(password)

        try (ResultSet rs = pstmt.executeQuery()) {

            // If rs.next() is true, a matching user was found
            return rs.next();
        }

    } catch (SQLException e) {

        System.err.println("Error during login: " + e.getMessage());
        return false;
    }
}
```

Sample 2: FoodApp.java (Switching Scenes on Login) This snippet shows how the start method would be modified. It no longer shows the restaurant list directly. Instead, it shows the createLoginScene. The "Login" button within that scene is what triggers the transition to the createRestaurantListScene.

```
// In FoodApp.java

private Scene loginScene;
private Scene restaurantListScene;
// ... other scenes

@Override
public void start(Stage stage) {
    primaryStage = stage;
    primaryStage.setTitle("Fooddiispatch");

    // Create all scenes upfront
    loginScene = createLoginScene(); // <-- NEW SCENE
    restaurantListScene = createRestaurantListScene();

    // Start by showing the login scene
    primaryStage.setScene(loginScene);
    primaryStage.show();
}

public Scene createLoginScene() {
    // ... (Layout code for login screen)

    Button loginButton = new Button("Login");
    loginButton.setOnAction(e -> {
        // 1. Get text from username/password fields
        String user = usernameField.getText();
        String pass = passwordField.getText();

        // 2. Validate with database
        boolean loggedIn = dbConnector.loginUser(user, pass);

        // 3. If valid, switch scene
        if (loggedIn) {
            primaryStage.setScene(restaurantListScene);
        } else {
            // Show an error alert
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setHeaderText("Login Failed");
            alert.setContentText("Invalid username or password.");
            alert.showAndWait();
        }
    });

    // ... (rest of layout)
    return new Scene(layout, 500, 600);
}
```

CHAPTER 5

SCREEN SHOTS

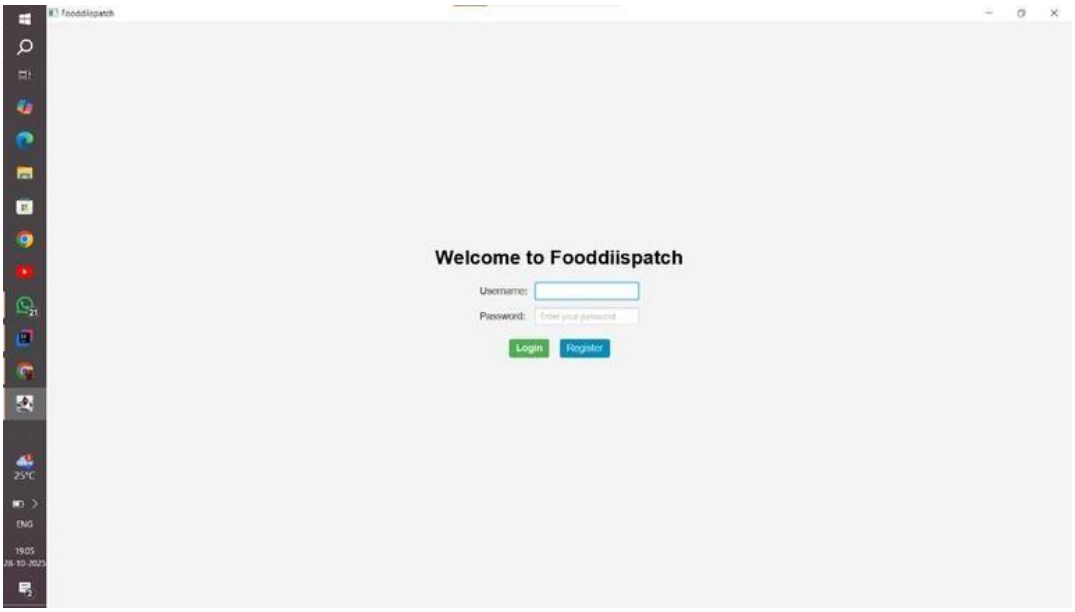

Fig 5.1 User Login and Registration Screen

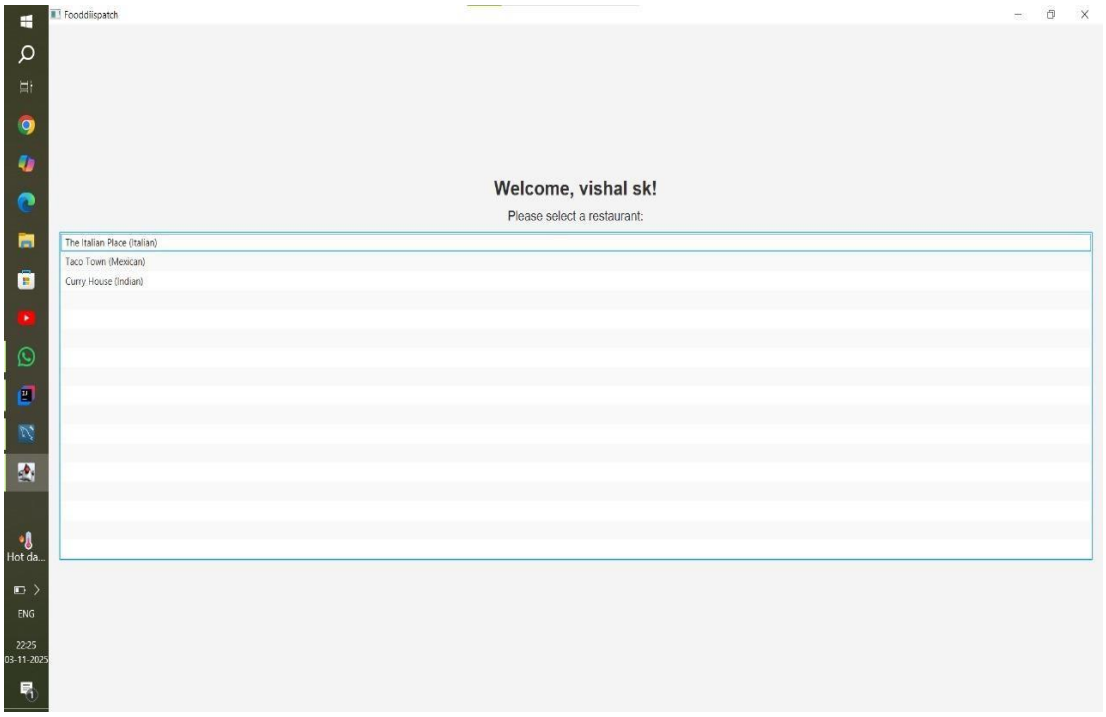

Fig 5.2 Restaurant Selection Screen

Fig 5.3 Restaurant Menu Screen

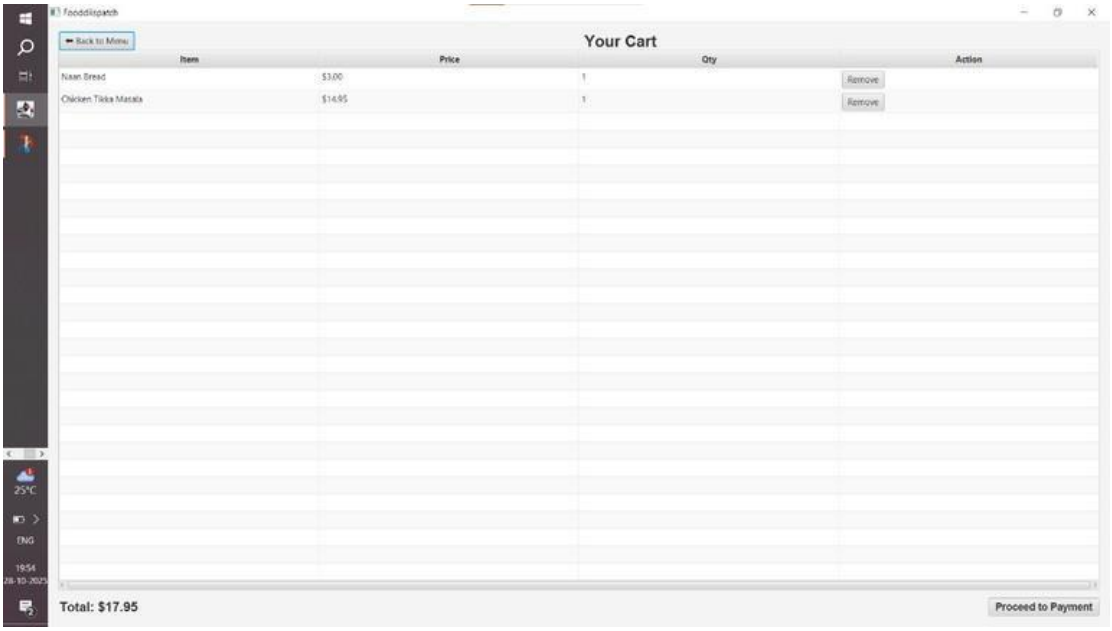

Fig 5.4 Shopping Cart Screen

Fig 5.5 Payment Screen (With QR Code)

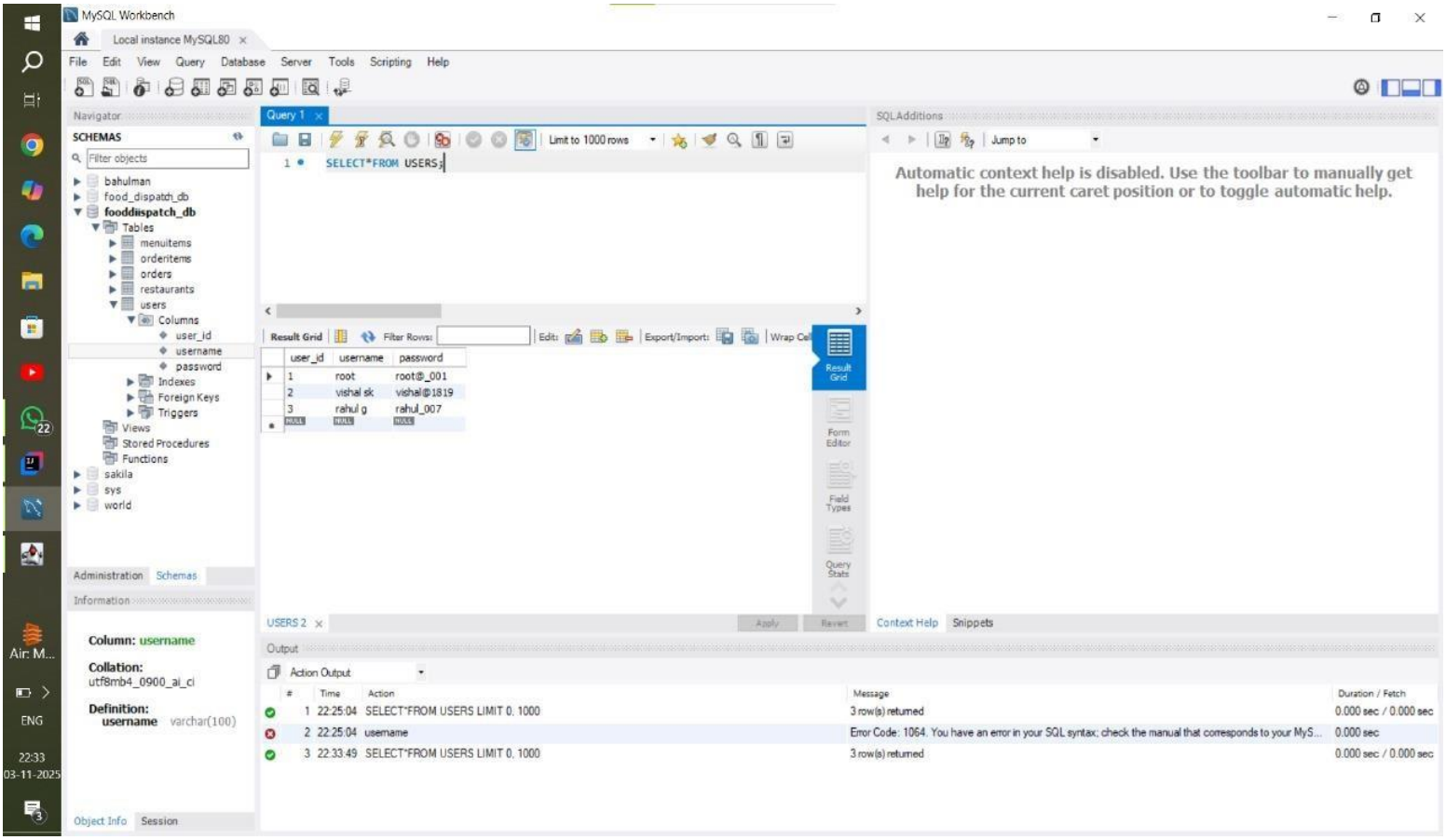

Fig 5.6 Database: users Table

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

In such a way, with the help of our project, customers can create an account, log in, check the list of local restaurants, browse their menus, and place an order. The ordering system clearly represents the available menu items, and the order management becomes easier for the restaurant as it is logged directly into their database and associated with a customer account.

In the future, this project could be enhanced to improve the user account functionality, such as allowing users to save favorite items and view their past order history (linked by their user_id). A new admin module could be added for restaurants to log in and update their own menu items or mark items as "out of stock."

REFERENCES

JavaFX Documentation: https://openjfx.io/openjfx-docs/

MySQL Official Documentation: https://dev.mysql.com/doc/

W3Schools SQL Tutorial: https://www.w3schools.com/sql/

Oracle Java JDBC Tutorial: https://docs.oracle.com/javase/tutorial/jdbc/basics/

GeeksforGeeks JavaFX: https://www.geeksforgeeks.org/javafx/

Baeldung JavaFX: https://www.baeldung.com/javafx-basics