RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM-602 105

CS23333 OOPS Using Java

Laboratory Record Note Book

Name :	ROSHAN NUR H
Year / Branch / Section :	II / CSE
University Register No. :	2116240701440
College Roll No. :	240701440
Semester :	III
Academic Year :	2024-2028

RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution

BONAFIDE CERTIFICATE

Name: ROSHAN NUR H ……

Academic Year: 2024-2028 **Semester:** III **Branch:** CSE

Register No. 240701440

Certified that this is the bonafide record of work done by the above student in the OOPS USING JAVA ***Laboratory***

during the academic year 2025- 2026

Signature of Faculty in-charge

Submitted for the Practical Examination held on……

Internal Examiner **External Examiner**

INDEX

EX.NO	DATE	NAME OF THE EXPERIMENT	GITHUB QR
1		I/O, Data Types, Operators	https://github.com/Roshan-440/240701440-OOPS-Java
2		Control Structures	
3		Arrays	
4		Strings	
5		Classes & Objects	
6		Inheritance	
7		Interface	
8		Exceptions	
9		Collections	
10		Collections	
11		Project	
12		Lambda	

ABSTRACT

The Food Delivery Web Application is a comprehensive full-stack Java-based web platform designed to streamline the process of ordering food from restaurants and delivering it to customers' doorsteps. In today's digital economy, online food delivery has become an essential service connecting customers, restaurants, and delivery partners seamlessly. This project addresses the growing demand for efficient food delivery services by providing a robust, user-friendly platform that manages the entire lifecycle of food orders—from browsing restaurants and menus to placing orders, processing payments, and tracking deliveries in real-time.

The application implements a multi-user architecture supporting four key actors: customers who browse and order food, restaurant owners who manage their menus and view orders, delivery partners who execute deliveries, and administrators who oversee the entire system. Built using Java technologies including JSP, Servlets, JDBC, and MySQL, the application follows the Model-View- Controller (MVC) architectural pattern deployed on Apache Tomcat Server. The primary objective of this project is to create an efficient, scalable, and secure platform that enhances the user experience while reducing operational complexity for restaurants and delivery services. This system provides an alternative to existing platforms, particularly benefiting local restaurants and regional food delivery services seeking to establish their digital presence without extensive infrastructure investment

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. Our heartfelt thanks to our Head of Department Dr. E. M. MALATHY and Deputy Head Dr. J. MANORANJINI for their constant encouragement throughout our project. We also extend our sincere gratitude to our internal guide Mrs B Deepa for his valuable guidance and motivation during the completion of this project. Finally, we thank our family members, friends, and all staff members of the Computer Science and Engineering department for their continuous support.

240701621 – Muhammed Qaiser J

240701440 – Roshan Nur H

240701611 – Yaswanth Sreenivas D

TABLE OF CONTENTS

CHAPTER 1 - INTRODUCTION

1.1 INTRODUCTION

The Food Delivery Web Application is a comprehensive platform designed to revolutionize how customers access food from restaurants in their locality. This web-based solution enables customers to browse multiple restaurants, view detailed menus with food descriptions and images, add items to a shopping cart, proceed through a secure checkout process, and track their orders in real-time. The application serves as a bridge connecting customers seeking convenient food delivery with restaurants aiming to expand their digital footprint and reach a broader customer base. By centralizing restaurant information, menu management, order processing, and delivery logistics on a single unified platform, the application dramatically improves operational efficiency and enhances the overall user experience for all stakeholders involved.

1.2 SCOPE OF THE WORK

The scope of this project encompasses the development of a full-stack web application capable of handling multiple concurrent users across different roles. The system provides comprehensive features including user authentication and registration, restaurant discovery and browsing, dynamic menu display with filtering and searching capabilities, shopping cart management, multiple payment gateway integration, real-time order tracking, and administrative controls for managing restaurants, users, and orders. The application supports both desktop and responsive mobile browsing, ensuring

accessibility across various devices. The backend infrastructure utilizes relational database management systems to maintain data integrity and consistency. The application is designed to scale with growing user bases and can accommodate additional features such as restaurant ratings and reviews, coupon management, and advanced analytics in future iterations.

1.3 PROBLEM STATEMENT

The existing landscape of local food delivery services faces significant challenges in the modern digital economy. While major platforms like Zomato and Swiggy dominate urban

markets with massive infrastructure investments, small to medium-sized restaurants struggle to compete and reach customers beyond their immediate geographical vicinity. Many restaurants lack the resources or technical expertise to develop their own delivery platforms, while customers in secondary cities and rural areas have limited options for convenient food ordering. Manual order management through phone calls or in-person visits creates inefficiencies, increases the likelihood of order errors, and prevents restaurants from scaling their operations effectively. Additionally, customers frequently lack transparency regarding order status and delivery timelines, leading to frustration and reduced customer satisfaction. There is a critical need for an accessible, cost-effective, yet robust food delivery platform that can serve local restaurants and their customers efficiently, bridging the gap between market demand and available technological solutions.

1.4 AIM AND OBJECTIVES OF THE PROJECT

Primary Aim:

The primary aim of this project is to develop a comprehensive, user-friendly food delivery web application that facilitates seamless interactions between customers, restaurants, and delivery personnel while maintaining data integrity, security, and scalability.

Specific Objectives:

1. Customer-Centric Design: Create an intuitive interface enabling customers to easily search restaurants, browse menus, place orders, and track deliveries in real-time.

2. Restaurant Empowerment: Provide restaurant owners with tools to manage their profiles, update menus, monitor incoming orders, and view performance analytics.

3. Secure Payment Processing: Implement robust payment gateway integration with encryption and validation to ensure secure financial transactions.

4. Real-Time Order Management: Develop a system capable of tracking orders through multiple states—placed, confirmed, preparing, dispatched, and delivered—with real-time notifications to all stakeholders.

5. Administrative Control: Establish administrative functionalities for user management, restaurant verification, order monitoring, and dispute resolution.

6. Scalability and Performance: Build the application using best practices in software architecture to ensure it can handle increasing user loads and transaction volumes without performance degradation.

7. Data Security: Implement authentication mechanisms, data encryption, and secure session management to protect user information and transaction details.

CHAPTER 2 - SYSTEM SPECIFICATIONS

Hardware Specifications:
Processor: Intel i5
RAM: 8 GB minimum
Hard Disk: 500 GB minimum

Software Specifications:
Operating System: Windows 10
Front-End: Java
Back-End: File Handling / Database (if extended)
Language: Java

CHAPTER 3 - MODULE DESCRIPTION

The Food Delivery Web Application consists of eight primary modules, each handling specific functional requirements:

Module 1: User Authentication & Registration

This module manages user signup, login, profile creation, and role-based access control. Users can register as customers, restaurant owners, or delivery partners. The system implements password hashing, session management, and secure token-based authentication to prevent unauthorized access.

Module 2: Restaurant Management

Restaurant owners can register their businesses, update operating hours, manage restaurant details (name, location, cuisine type, contact information), upload logos/images, and view performance metrics including total orders and customer ratings.

Module 3: Menu Management

This module allows restaurant owners to create, read, update, and delete menu items. Each food item includes attributes such as name, description, price, availability status, dietary tags (vegetarian/non- vegetarian), and high-quality images.

Module 4: Customer Browsing & Search

Customers can search restaurants by location, cuisine type, ratings, or keyword. The system implements filtering and sorting capabilities, displaying restaurant details including ratings, delivery time, and cuisine specialization.

Module 5: Shopping Cart & Checkout

This module manages the shopping cart where customers add items, modify quantities, view real-time totals, select delivery addresses, and apply coupon codes before proceeding to payment.

Module 6: Payment Processing

The payment module integrates with payment gateways to handle secure transactions. It supports multiple payment methods (credit/debit cards, digital wallets, cash on delivery) with proper validation, encryption, and transaction logging.

Module 7: Order Management & Tracking

After order confirmation, customers receive real-time updates on order status (placed, confirmed, preparing, dispatched, delivered). The system sends automated notifications via email or SMS at each stage, and displays estimated delivery time.

Module 8: Admin Dashboard & Analytics

The administrative module provides tools for user management, restaurant verification, order monitoring, revenue analytics, complaint resolution, and system configuration.

CHAPTER – 4 ARCHITECTURE DIAGRAM & ER DIAGRAM

4.1 ARCHITECTURE

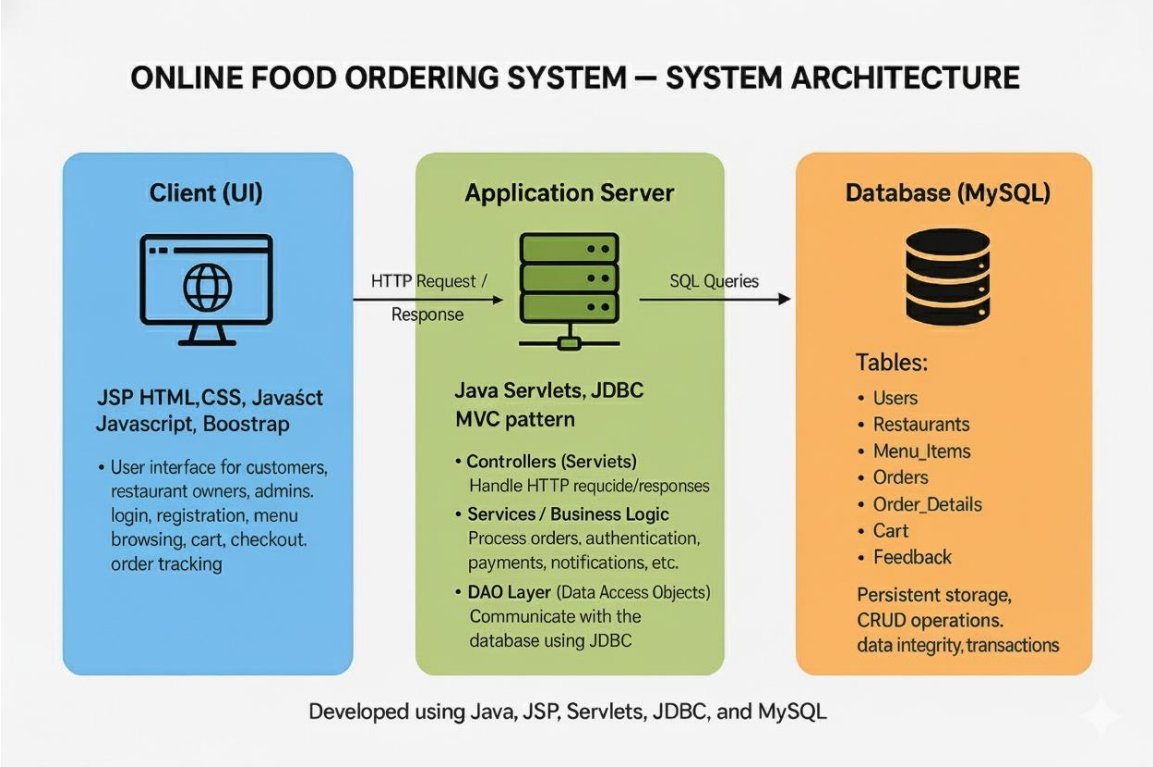


FIG NO: 4.1 ARCHITECTURE DIAGRAM

4.1 System Architecture Overview

Architecture Style: MVC (Model–View–Controller)

Deployment Model: 3-Tier Web Architecture

1. Client Layer (Presentation Layer)

Technologies: JSP, HTML, CSS, JavaScript, Bootstrap

Responsibilities:

User interface for customers, admins, and restaurant owners

Pages for login, registration, menu browsing, cart, checkout, and tracking

Handles user input and displays server responses

→ Communicates with Servlet Controller via HTTP requests.

2. Application Layer (Business Logic Layer)

Technologies: Java Servlets, JDBC, RESTful APIs

Responsibilities:

Acts as the Controller in MVC

Processes requests (e.g., Add to Cart, Place Order)

Validates inputs and applies business logic

Manages sessions, authentication, and authorization

Communicates with the Model (Database) via DAO classes

→ Uses JDBC to interact with MySQL.

3. Database Layer (Data Layer)

Technology: MySQL

Entities:

User

Restaurant

Menu Item

Order

Order Details

Cart

Responsibilities:

Stores and retrieves persistent data

Maintains data integrity and relationships

Supports CRUD operations via DAO

4.2 E-R DIAGRAM :

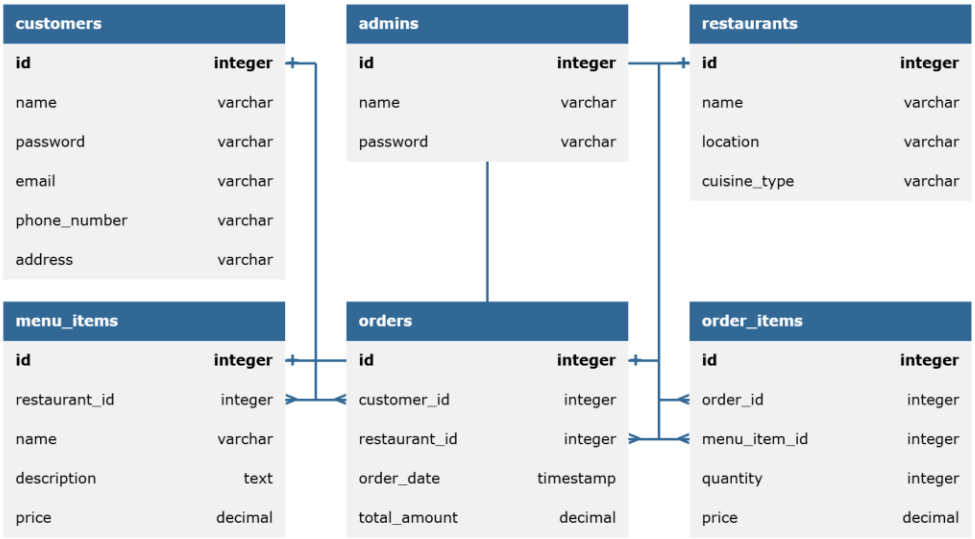


FIG NO : 4.2 ENTITY RELATIONSHIP DIAGRAM.

Entities

Customer – Represents people who register and place orders in the application.
Attributes: Customer_ID, Name, Password, Email, Phone_Number, Address.

Restaurant – Represents restaurants offering food for ordering.
Attributes: Restaurant_ID, Name, Location, Cuisine_Type.

Menu_Item – Details food items available for purchase from restaurants.
Attributes: Item_ID, Restaurant_ID, Name, Description, Price.

Admin – Manages system operations, restaurant details, and user access.
Attributes: Admin_ID, Name, Password.

Order – Stores information about customer food purchase orders.
Attributes: Order_ID, Customer_ID, Restaurant_ID, Order_Date, Total_Amount.

Order_Item – Details individual food items within each order.
Attributes: Order_Item_ID, Order_ID, Item_ID, Quantity, Price.

Relationships

Customer – Order → One-to-Many
A single customer can place multiple orders; each order is linked to one customer.

Restaurant – Menu_Item → One-to-Many
A restaurant offers multiple menu items; each menu item belongs to one restaurant.

Restaurant – Order → One-to-Many
A restaurant receives multiple orders; each order belongs to one restaurant.

Order – Order_Item → One-to-Many
Each order contains multiple food items; each order item is part of one order.

Menu_Item – Order_Item → One-to-Many
A menu item can be part of multiple order items; each order item refers to one menu item.

Admin – Restaurant/Menu_Item→ One-to-Many
Admin manages records of restaurants, menu items

Connections

Customer ↔ Order ↔ Order_Item ↔ Menu_Item ↔ Restaurant:
Customers select menu items from restaurants and place orders containing multiple items.

Admin ↔ Restaurant/Menu_Item:
Admins add, update, or moderate restaurants and manage menu inventory.

CHAPTER 5 - IMPLEMENTATION

Sample 1: User Login Servlet

java

```
import java.io.;
import javax.servlet.;
import javax.servlet.http.;
import java.sql.;
public class LoginServlet extends HttpServlet {
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        String email = request.getParameter("email");
String password = request.getParameter("password");
String userType = request.getParameter("userType");
try {
Class.forName("com.mysql.jdbc.Driver");
Connection conn = DriverManager.getConnection(
"jdbc:mysql://localhost:3306/fooddeliverydb", "root", "password");
String query = "SELECT * FROM users WHERE email = ? AND user_type = ?";
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, email);
ps.setString(2, userType);
ResultSet rs = ps.executeQuery();
if (rs.next()) {
String hashedPassword = rs.getString("password");
// Validate password using BCrypt or similar
if (verifyPassword(password, hashedPassword)) {
```

```
HttpSession session = request.getSession();
session.setAttribute("userId", rs.getInt("user_id"));
session.setAttribute("email", email);
session.setAttribute("userType", userType);
response.sendRedirect("dashboard.jsp");
} else {
response.sendRedirect("login.jsp?error=invalid_credentials");
}
} else {
response.sendRedirect("login.jsp?error=user_not_found");
}
conn.close();
} catch (Exception e) {
e.printStackTrace();
}
}
private boolean verifyPassword(String password, String hashedPassword) {
// Implementation of password verification logic
return true;
 }
}
```

Sample 2: Restaurant Listing JSP

jsp

```
<%@ page import="java.sql.*" %>
<%@ page import="java.util.List" %>
<%
```

```
try {
Class.forName("com.mysql.jdbc.Driver");
Connection conn = DriverManager.getConnection(
"jdbc:mysql://localhost:3306/fooddeliverydb", "root", "password");
String query = "SELECT restaurant_id, name, location, cuisine_type, rating, image_url
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery(query);
%>
<% while (rs.next()) { %>
" alt="<%= rs.getString("name") %>">
Cuisine: <%= rs.getString("cuisine_type") %>
Location: <%= rs.getString("location") %>
Rating: <%= rs.getFloat("rating") %> ☆
" class="btn btn-primary">View Menu
<% } %>
<%
rs.close();
stmt.close();
conn.close();
} catch (Exception e) {
e.printStackTrace();
}
%>
```

CHAPTER 6 - SCREENSHOTS

Fig 5.1 Login Page

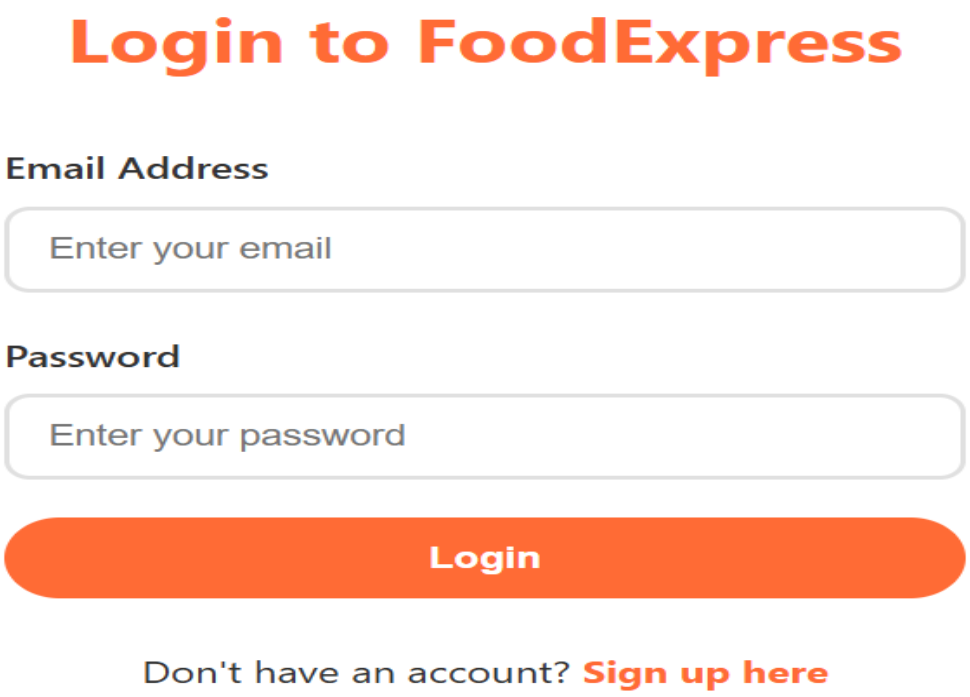


Fig 5.2 Restaurant Listings

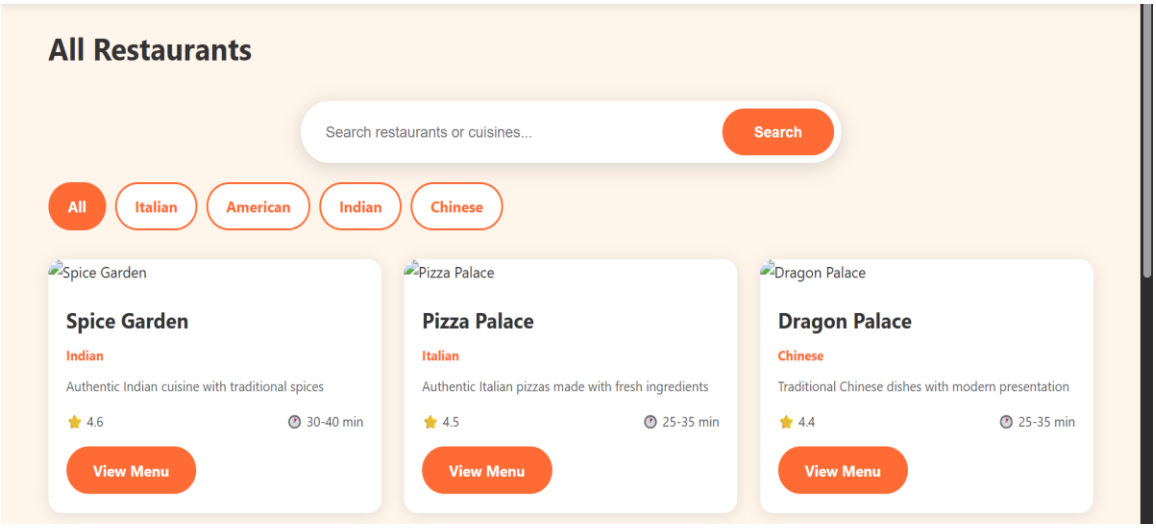


Fig 5.3 Menu Page

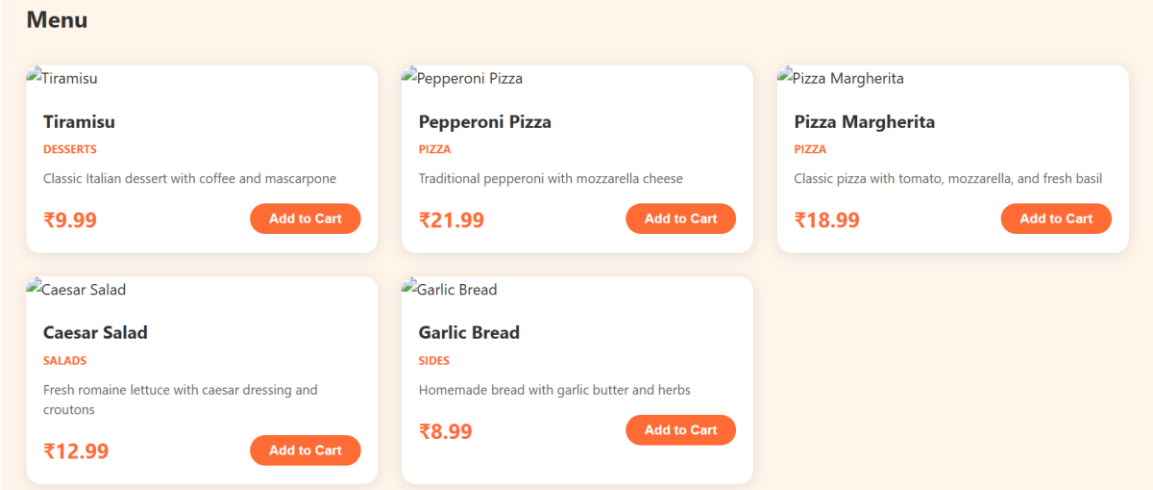
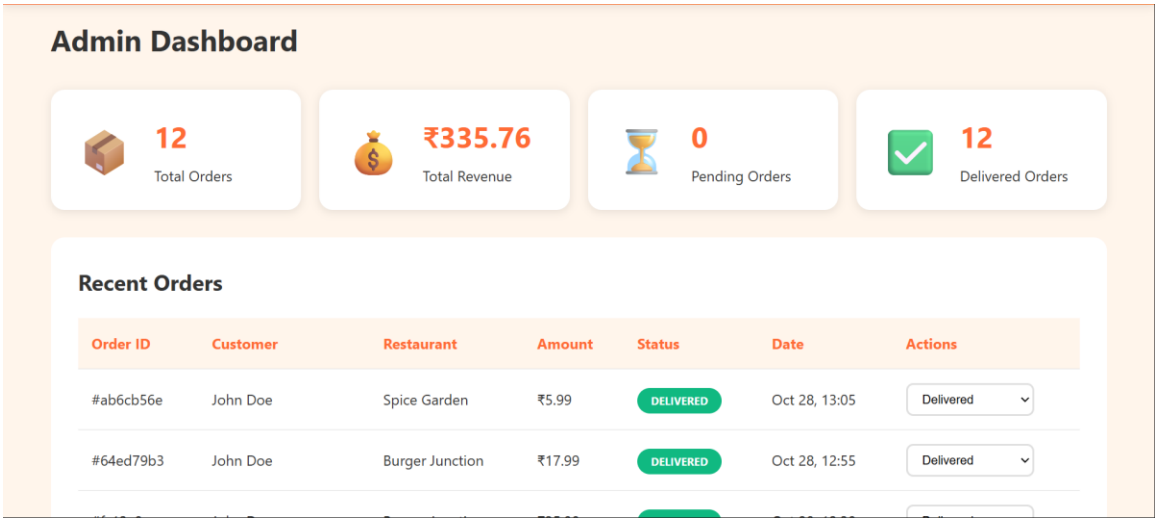


Fig 5.4 Admin Dashboard



CHAPTER 7 - CONCLUSION AND FUTURE ENHANCEMENT

Conclusion

The Food Delivery Web Application successfully demonstrates the practical implementation of a comprehensive full-stack web solution addressing real-world requirements of modern food delivery services. Through the integration of Java technologies, relational databases, and web development best practices, the application provides an efficient platform connecting customers, restaurants, and delivery partners. The MVC architecture ensures clean code separation, maintainability, and scalability, while security features including password hashing and session management protect user data. The project showcases proficiency in backend development using Servlets and JSP, database management with MySQL and JDBC, and frontend development with HTML, CSS, and JavaScript. By successfully implementing core features such as user authentication, restaurant browsing, order management, and real-time order tracking, the application meets its primary objectives and provides a solid foundation for future enhancements.

Future Enhancement Possibilities

1. Mobile Application Development: Extend the platform to native iOS and Android applications using technologies like React Native or Flutter for improved user engagement and accessibility.

2. AI-Based Recommendations: Implement machine learning algorithms to provide personalized restaurant and menu item recommendations based on user browsing and ordering history.

3. Advanced Payment Integration: Integrate cryptocurrency payment options and blockchain-based transaction verification for enhanced security and emerging payment trends. 4. Geolocation & Route Optimization: Implement real-time GPS tracking for delivery partners

with advanced route optimization algorithms to reduce delivery times and fuel costs.

5. Analytics & Business Intelligence: Develop comprehensive analytics dashboards providing restaurants with detailed insights into customer behavior, peak ordering times, and revenue trends.

6. Multi-Language Support: Add internationalization capabilities to support multiple languages, expanding the platform's reach to diverse user populations.

7. API Marketplace: Create public APIs allowing third-party developers to build extensions and integrations, fostering an ecosystem around the platform.

8. Subscription Plans: Implement tiered subscription models for restaurants offering premium features like priority customer support and enhanced menu customization.

REFERENCES

1. https://www.w3schools.com/java/

2. https://www.tutorialspoint.com/java/

3. https://www.geeksforgeeks.org/java/

4. https://www.oracle.com/java/technologies/

5. https://www.github.com/