C. V. Raman Global University
ODISHA BHUBANESWAR INDIA

Cranes Varsity
'Where Technology Meets Excellence'

RESTAURANT MANAGEMENT SYSTEM

JAVA

A PROJECT REPORT

Submitted by

NAME	REGD. NO.	CRANES REGD.
ABHISHEK SAHOO	2201020689	CL2025010601949635
DEBANSHU DEVGOSWAMI	2201020452	CL2025010601875158
SATYAM MOHANTY	2201020489	CL2025010601949130
MRINALINI PATNAIK	2201020792	CL2025010601914379
ADITYA DASH	2201020496	CL2025010601951857

Guided by

Vishnu Reddy

CRANES VARSITY

CERTIFICATE OF APPROVAL

This is to certify that we have examined the project entitled " **RESTAURANT MANAGEMENT SYSTEM** " submitted by *Abhishek Sahoo, Debanshu Devgoswami, Satyam Mohanty, Mrinalini Patnaik,* and *Aditya Dash*. We hereby accord our approval of it as a project work carried out and presented in a manner required for its acceptance for the partial fulfilment for the **Computer Science and Engineering** for which it has been submitted. This approval does not necessarily endorse or accept every statement made, opinion expressed or conclusions drawn as recorded in this project, it only signifies the acceptance of the project for the purpose it has been submitted.

ACKNOWLEDGEMENT

We would like to articulate our deep gratitude to our project guide **Vishnu Reddy**, Professor who has always been source of motivation and firm support for carrying out the project.

We would also like to convey our sincerest gratitude and indebtedness to all other faculty members, who bestowed their great effort and guidance at appropriate times without it would have been very difficult on our project work.

An assemblage of this nature could never have been attempted with our reference to and inspiration from the works of others whose details are mentioned in references section. We acknowledge our indebtedness to all of them. Further, we would like to express our feeling towards our parents and God who directly or indirectly encouraged and motivated us during Assertion.

CONTENTS

ABSTRACT

The **Restaurant Management System (RMS)** is a software solution designed to streamline the operations of restaurants by automating tasks such as order processing, billing, inventory management, and customer relationship management. Traditional restaurant operations involve manual record-keeping, leading to inefficiencies, errors, and delays. This system aims to improve the efficiency, accuracy, and overall customer experience by integrating modern technologies into restaurant management.

The RMS consists of several key modules, including **order management, menu management, billing system, inventory tracking, and staff management**. The order management module allows customers to place orders, either through a waiter or via an integrated online platform. The menu management module enables restaurant administrators to update and modify menu items dynamically, reflecting real-time availability. The billing system automates invoice generation, tax calculations, and payment processing, reducing errors and enhancing transparency.

The **inventory management module** helps track stock levels of raw materials and ingredients, alerting restaurant managers when supplies are low. This prevents shortages and ensures smooth operations. Additionally, the **staff management module** allows restaurant owners to manage employee schedules, attendance, and payroll efficiently.

The **Restaurant Management System** is developed using a **Java-based backend with MySQL as the database**, ensuring robust data management and security. The system supports **both online and offline operations**, making it suitable for restaurants of all sizes. Additionally, it features an **intuitive user interface**, allowing restaurant staff to operate the system with minimal training.

The benefits of implementing an RMS include **increased efficiency, reduced operational costs, improved customer satisfaction, and better decision-making through data analytics**. By automating routine tasks, restaurant managers can focus more on customer service and business growth. Furthermore, integration with **POS (Point of Sale) systems** and digital payment gateways enhances the customer experience by offering seamless transactions.

INTRODUCTION

A **Restaurant Management System** is a specialized software designed to streamline and optimize the operations of a restaurant, café, or food service establishment. It acts as the central hub for managing various aspects of restaurant operations, including order processing, billing, inventory tracking, and customer management. By integrating different functions into a single platform, an RMS helps restaurant owners and staff improve efficiency, reduce errors, and enhance the overall dining experience.

In today's fast-paced food industry, restaurants face numerous challenges such as high customer expectations, fluctuating demand, staff management issues, and supply chain complexities. A well-implemented RMS addresses these challenges by automating routine tasks, improving accuracy in order processing, and maintaining seamless coordination between kitchen staff, waiters, and management. With digital transformation taking over various industries, restaurants are also adopting technology-driven solutions to stay competitive and meet modern customer preferences.

One of the primary functions of a restaurant is taking and processing orders efficiently. A Restaurant Management System simplifies this process by allowing orders to be placed digitally, whether through a Point of Sale (POS) system, online ordering platforms, or self-service kiosks. This reduces human errors, minimizes wait times, and ensures better coordination between the front-end staff and kitchen operations.

Another crucial aspect of restaurant operations is handling finances and billing. Traditional paper-based billing systems are prone to errors and inefficiencies. An RMS automates the billing process, calculates taxes, applies discounts, and generates receipts accurately, ensuring a smooth checkout experience for customers. Moreover, it can integrate with digital payment methods, including credit cards, mobile wallets, and online payment gateways, providing greater convenience to customers.

Keeping track of ingredients, monitoring stock levels, and reducing food wastage are essential for maintaining profitability. An RMS helps restaurant owners track inventory in real time, send alerts for low-stock items, and optimize procurement processes. This prevents shortages and excess stock, ensuring a well-balanced supply chain.

RELEVANT TECHNOLOGIES

A Restaurant Management System integrates various technologies to streamline operations such as order processing, inventory management, customer service, and financial transactions. Below are the key technologies required:

1. Database Management System (DBMS)

- **MySQL, PostgreSQL, MongoDB** – Used to store customer orders, menu details, employee records, and inventory.
- **Firebase** – A cloud-based alternative for real-time database synchronization.

2. Backend Technologies

- **Java (Spring Boot), Node.js (Express), Python (Django/Flask), PHP (Laravel)** – Used to manage business logic, API communication, and database interactions.
- **GraphQL/REST APIs** – Enables seamless integration between mobile apps, web platforms, and third-party services.

3. Mobile Technologies

- **Android (Kotlin/Java), iOS (Swift), Flutter, React Native** – Used to develop mobile applications for online ordering, table reservations, and staff management.

4. POS (Point of Sale) System

- **Square POS, Toast POS, Lightspeed POS** – Helps in billing, managing payments, and tracking sales.
- **Integration with barcode scanners, receipt printers, and card readers** for seamless transactions.

5. Cloud Computing & SaaS Solutions

- **AWS, Google Cloud, Microsoft Azure** – Used for hosting cloud-based restaurant management systems.

- **SaaS Platforms (Toast, Square, Oracle MICROS)** – Ready-made solutions for restaurants to manage orders and inventory.

6. Payment Processing Technologies

- **Stripe, PayPal, Razorpay, Google Pay, Apple Pay** – Secure online payment gateways for processing transactions.
- **NFC & QR Code Payments** – Contactless payment technology to improve customer convenience.

7. IoT & Smart Devices

- **IoT Sensors** – Used in kitchen appliances to monitor temperature and energy usage.
- **Smart Kiosks & Digital Menu Boards** – Self-service ordering systems for customer convenience.

8. AI & Data Analytics

- **AI Chatbots (Dialogflow, IBM Watson)** – Handles reservations and customer queries.
- **Data Analytics (Power BI, Google Analytics)** – Tracks sales, customer preferences, and demand forecasting.

METHODOLOGY

1. Requirement Analysis

Objectives:

- Identify the needs of the restaurant (e.g., order management, billing, inventory, reservations).
- Define user roles (e.g., Admin, Waiter, Chef, Customer).
- Gather requirements from stakeholders (restaurant owners, staff, customers).

Deliverables:

- Requirement Specification Document

- Use Case Diagrams
- Functional and Non-Functional Requirements

2. Feasibility Study

Aspects Considered:

- **Technical Feasibility:** Identify the best technology stack (e.g., web-based, mobile app, cloud-based).
- **Economic Feasibility:** Estimate budget, development costs, and return on investment (ROI).
- **Operational Feasibility:** Assess usability for staff and customers.

Deliverables:

- Feasibility Report
- Technology & Platform Selection

3. System Design

Key Activities:

- **Architectural Design:** Define system architecture (e.g., client-server, cloud-based).
- **Database Design:** Develop an ER diagram and database schema for tables (e.g., Orders, Customers, Menu, Inventory).
- **UI/UX Design:** Design wireframes and mockups for a user-friendly interface.

Deliverables:

- System Architecture Diagram
- Database Schema
- UI/UX Wireframes

4. Development & Implementation

Development Approach:

- **Agile Methodology** (Iterative Development with Feedback Loops).
- **Technology Stack:**
    - **Frontend:** React, Angular, or Vue.js
    - **Backend:** Node.js, Django, or Laravel
    - **Database:** MySQL, PostgreSQL, or Firebase
    - **Cloud Services:** AWS, Google Cloud, or Azure

Implementation Phases:

- Develop core modules:
    1. User Authentication & Role Management
    2. Menu & Order Management
    3. Table Reservations
    4. Payment Integration
    5. Inventory Management
    6. Reports & Analytics

Deliverables:

- Fully Functional System
- API Documentation

5. Testing & Quality Assurance

Testing Strategies:

- **Unit Testing:** Test individual modules (e.g., order placement, payments).
- **Integration Testing:** Ensure seamless communication between different system components.
- **User Acceptance Testing (UAT):** Conduct testing with restaurant staff and customers.
- **Performance Testing:** Assess load handling and response times.

Deliverables:

- Test Cases & Reports
- Bug Fixes & Refinements

6. Maintenance & Support

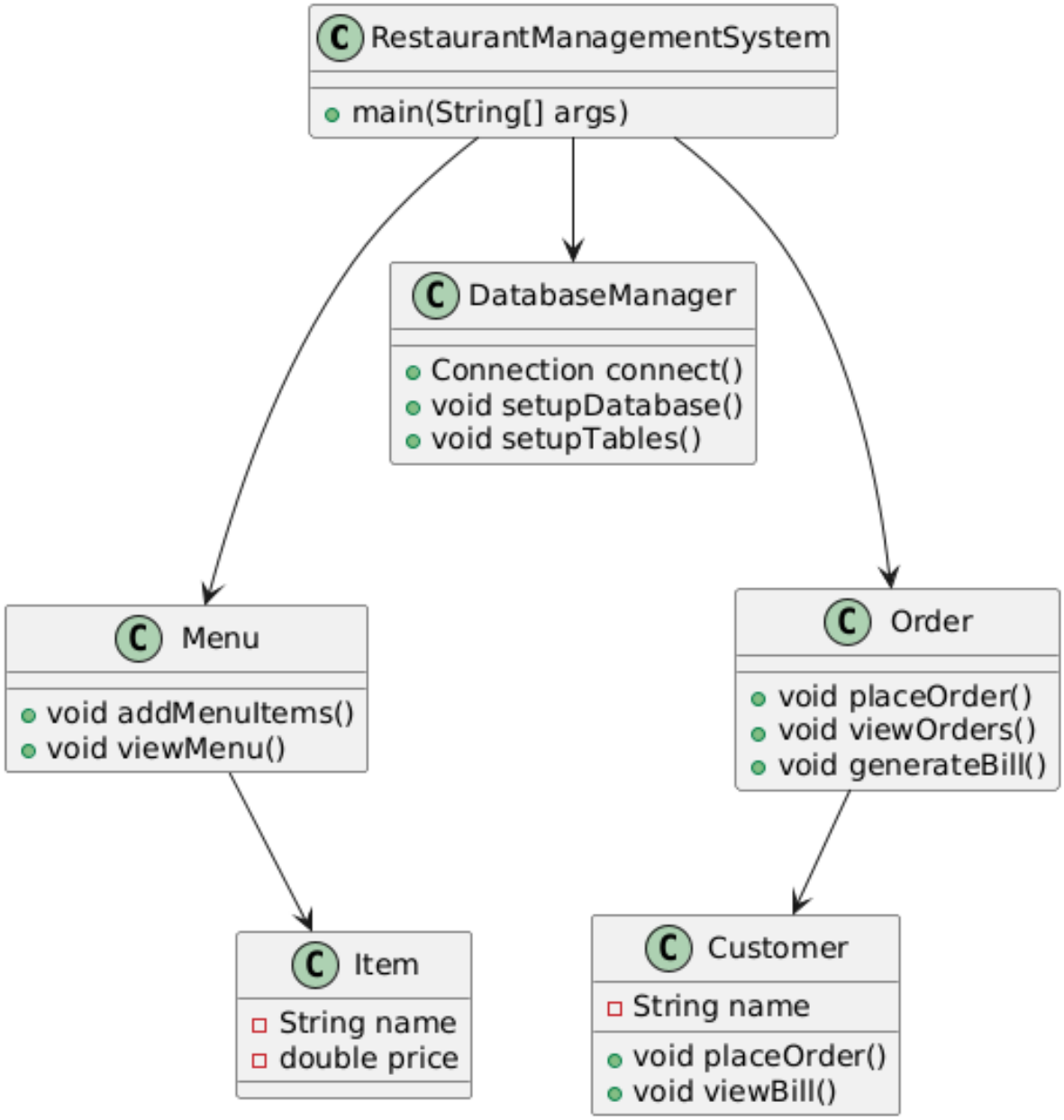
Ongoing Support:

- Regular updates & security patches.
- Feature enhancements based on user feedback.
- 24/7 technical support for issues.

Deliverables:

- Maintenance Logs
- Future Roadmap for Enhancements

FLOWCHART


RestaurantManagementSystem
main(String[] args)
DatabaseManager
Connection connect()
void setupDatabase()
void setupTables()
Menu
void addMenuItems()
void viewMenu()
Order
void placeOrder()
void viewOrders()
void generateBill()
Item
String name
double price
Customer
String name
void placeOrder()
void viewBill()

IMPLEMENTATION

```
import java.sql.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class RestaurantManagementSystem {
    public static void main(String[] args) {
        String databaseName = "restaurant";
        String menuTableName = "menu";
        String orderTableName = "orders";

        try {
            String url = "jdbc:mysql://localhost:3306/";
            String userName = "root";
            String password = "1234@@*";
            Connection connection = DriverManager.getConnection(url, userName, password);
            Statement statement = connection.createStatement();

            String checkDatabaseExistenceSQL = "SELECT SCHEMA_NAME FROM
INFORMATION_SCHEMA.SCHEMATA WHERE SCHEMA_NAME = '" + databaseName + "';";
            boolean databaseExists = statement.executeQuery(checkDatabaseExisten-
ceSQL).next();

            if (!databaseExists) {
                String createDatabaseSQL = "CREATE DATABASE " + databaseName;
                statement.executeUpdate(createDatabaseSQL);
            }

            String useDatabaseSQL = "USE " + databaseName;
            statement.executeUpdate(useDatabaseSQL);

            String createMenuTableSQL = "CREATE TABLE IF NOT EXISTS " + menuTableName + " ("
+
                    "ItemName VARCHAR(255) PRIMARY KEY, " +
                    "Price DECIMAL(10,2));";

            statement.executeUpdate(createMenuTableSQL);

            addMenuItems(connection, menuTableName);

            String createOrderTableSQL = "CREATE TABLE IF NOT EXISTS " + orderTableName + "
(" +
                    "CustomerName VARCHAR(255)," +
                    "ItemName VARCHAR(255)," +
                    "Quantity INT," +
                    "TableNumber INT);";

            statement.executeUpdate(createOrderTableSQL);
```

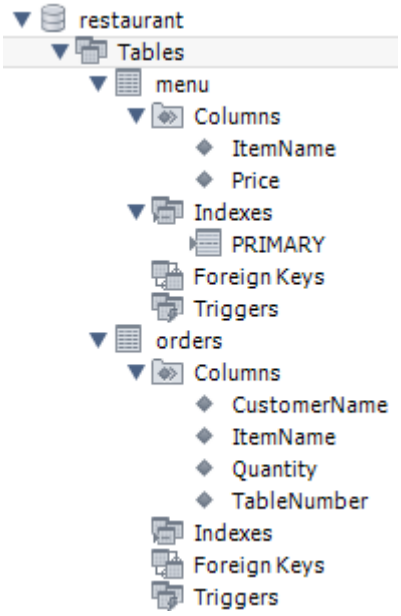
```
            boolean exit = false;
            Scanner scanner = new Scanner(System.in);
            while (!exit) {
                System.out.println("Restaurant Management System\n" +
                        "1. View Menu\n" +
                        "2. Place Order\n" +
                        "3. View Orders\n" +
                        "4. Generate Bill\n" +
                        "5. Exit\n" +
                        "Enter your choice (1-5):");
                String choice = scanner.nextLine();
                switch (choice) {
                    case "1":
                        viewMenu(connection, menuTableName);
                        break;
                    case "2":
                        placeOrder(connection, menuTableName, orderTableName, scanner);
                        break;
                    case "3":
                        viewOrders(connection, orderTableName);
                        break;
                    case "4":
                        generateBill(connection, orderTableName, scanner);
                        break;
                    case "5":
                        exit = true;
                        break;
                    default:
                        System.out.println("Invalid choice. Please enter a number between 1
and 5.");
                        break;
                }
            }

            statement.close();
            connection.close();
            System.out.println("Exiting Restaurant Management System.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static void addMenuItems(Connection connection, String tableName) throws
SQLException {
        String[] menuItems = {"Pizza", "Burger", "Pasta", "Salad", "Soda"};
        double[] prices = {12.99, 8.50, 10.75, 6.99, 2.50};
        for (int i = 0; i < menuItems.length; i++) {
            ResultSet resultSet = connection.createStatement().executeQuery(
                "SELECT * FROM " + tableName + " WHERE ItemName = '" + menuItems[i] + "'");
```

```
            if (!resultSet.next()) {
                String insertMenuItemSQL = "INSERT INTO " + tableName + " VALUES ('" +
                    menuItems[i] + "','" +
                    prices[i] + "')";
                connection.createStatement().executeUpdate(insertMenuItemSQL);
            }
        }
    }

    private static void viewMenu(Connection connection, String tableName) throws SQLException
{
        ResultSet resultSet = connection.createStatement().executeQuery(
            "SELECT * FROM " + tableName);
        System.out.println("Menu:");
        while (resultSet.next()) {
            String itemName = resultSet.getString("ItemName");
            double price = resultSet.getDouble("Price");
            System.out.println(itemName + " - $" + price);
        }
        System.out.println();
    }

    private static void placeOrder(Connection connection, String menuTableName, String
orderTableName, Scanner scanner) throws SQLException {
        System.out.println("Enter Customer Name:");
        String customerName = scanner.nextLine();

        List<String> items = new ArrayList<>();
        List<Integer> quantities = new ArrayList<>();

        while (true) {
            System.out.println("Enter Item Name (or type 'done' to finish):");
            String itemName = scanner.nextLine();
            if (itemName.equalsIgnoreCase("done")) {
                break;
            }
            items.add(itemName);
            System.out.println("Enter Quantity:");
            int quantity = Integer.parseInt(scanner.nextLine());
            quantities.add(quantity);
        }

        System.out.println("Enter Table Number:");
        int tableNumber = Integer.parseInt(scanner.nextLine());

        for (int i = 0; i < items.size(); i++) {
            String itemName = items.get(i);
            ResultSet resultSet = connection.createStatement().executeQuery(
                "SELECT * FROM " + menuTableName + " WHERE ItemName = '" + itemName + "'");
            if (resultSet.next()) {
```

```
                String insertOrderSQL = "INSERT INTO " + orderTableName + " VALUES ('" +
                    customerName + "','" +
                    itemName + "','" +
                    quantities.get(i) + "'," +
                    tableNumber + ")";
                connection.createStatement().executeUpdate(insertOrderSQL);
            } else {
                System.out.println("Item " + itemName + " not found in the menu.
Skipping...");
                continue;
            }
        }

        System.out.println("Orders placed successfully.");
    }

    private static void viewOrders(Connection connection, String tableName) throws
SQLException {
        ResultSet resultSet = connection.createStatement().executeQuery("SELECT * FROM " +
tableName);
        System.out.println("Orders:");
        while (resultSet.next()) {
            String customerName = resultSet.getString("CustomerName");
            String itemName = resultSet.getString("ItemName");
            int quantity = resultSet.getInt("Quantity");
            int tableNumber = resultSet.getInt("TableNumber");
            System.out.println("Customer: " + customerName + ", Item: " + itemName + ",
Quantity: " + quantity + ", Table: " + tableNumber);
        }
        System.out.println();
    }

    private static void generateBill(Connection connection, String tableName, Scanner
scanner) throws SQLException {
        System.out.println("Enter Customer Name to generate bill:");
        String customerName = scanner.nextLine();
        ResultSet resultSet = connection.createStatement().executeQuery("SELECT * FROM " +
tableName + " WHERE CustomerName = '" + customerName + "'");

        double totalBill = 0;
        System.out.println("Bill for " + customerName + ":");
        while (resultSet.next()) {
            String itemName = resultSet.getString("ItemName");
            int quantity = resultSet.getInt("Quantity");
            ResultSet itemResultSet = connection.createStatement().executeQuery("SELECT Price
FROM menu WHERE ItemName = '" + itemName + "'");
            if (itemResultSet.next()) {
                double price = itemResultSet.getDouble("Price");
                double subtotal = price * quantity;
                System.out.println(itemName + " (" + quantity + "): $" + subtotal);
                totalBill += subtotal;
```

```
            }
        }
        System.out.println("Total Bill: $" + totalBill);
    }
}
```

OUTPUT



ItemName	Price
Burger	8.50
Pasta	10.75
Pizza	12.99
Salad	6.99
Soda	2.50
NULL	NULL

CustomerName	ItemName	Quantity	TableNumber
Abhishek	Burger	3	34

APPLICATIONS

1. Point of Sale (POS) System

- Manages orders, payments, and billing
- Supports multiple payment methods (cash, credit, digital wallets)
- Integrates with kitchen display systems (KDS) and inventory

2. Inventory Management

- Tracks stock levels and ingredient usage
- Sends alerts for low stock
- Automates purchase orders with suppliers

3. Table Reservation System

- Online booking and real-time availability updates
- Waitlist management
- Automated reminders and confirmations

4. Kitchen Display System (KDS)

- Digital order tracking for kitchen staff
- Reduces order errors and improves efficiency
- Tracks preparation times

5. Customer Relationship Management (CRM)

- Stores customer preferences and order history
- Manages loyalty programs and rewards
- Sends personalized promotions and feedback requests

6. Employee Management

- Schedules shifts and tracks attendance
- Monitors performance and payroll
- Reduces labor costs through data-driven decisions

7. Reporting & Analytics

- Provides sales, expense, and profit reports
- Identifies best-selling items and peak hours
- Tracks customer trends and staff performance

8. Online Ordering & Delivery Integration

- Connects with third-party delivery apps (UberEats, DoorDash, etc.)

- Manages in-house delivery operations
- Provides real-time tracking for customers

9. Menu Management

- Updates digital menus across all platforms (POS, website, apps)
- Analyzes sales data to optimize menu offerings
- Supports dynamic pricing and seasonal changes

10. Feedback & Review Management

- Collects and analyzes customer feedback
- Responds to online reviews
- Enhances service based on customer insights

CHALLENGES AND LIMITATIONS

1. Customization Complexity – Some restaurants have unique workflows that generic RMS solutions may not support without expensive customization.
2. Internet Dependence – Cloud-based systems require stable internet; downtime can halt operations.
3. Regulatory Compliance – The system must comply with tax regulations, food safety, and labor laws, which vary by region.
4. Hidden Costs – Subscription fees, transaction fees, and updates can add to the total cost.
5. Hardware Compatibility – Some systems require proprietary hardware, increasing costs.
6. Limited AI & Analytics – Not all RMS solutions offer advanced data analytics for customer behavior and forecasting.

CONCLUSION

A **Restaurant Management System** is essential for modern food service businesses to streamline operations, enhance customer experience, and improve overall efficiency. By integrating key features such as order management, inventory tracking, billing, and customer relationship management, an RMS helps restaurant owners and staff optimize workflows, reduce human error, and increase profitability.

Implementing an RMS leads to **faster service, better resource allocation, and improved financial tracking**, ultimately contributing to higher customer satisfaction and business growth. As technology advances, adopting cloud-based and AI-driven solutions can further enhance restaurant management, providing insights into customer preferences and operational efficiency.

In conclusion, investing in a **well-designed and efficient Restaurant Management System** is a strategic decision that not only simplifies daily operations but also positions a restaurant for long-term success in an increasingly competitive industry.

REFERENCES

- **"Restaurant Success by the Numbers"** – Roger Fields
- **"Setting the Table: The Transforming Power of Hospitality in Business"** – Danny Meyer
- **"The Restaurant Manager's Handbook"** – Douglas Robert Brown
- Upserve Restaurant Management Resources – Industry trends and tools
- Modern Restaurant Management – News, insights, and guides
- **A Survey on Restaurant Management Systems and Their Evolution** – IEEE Xplore
- **Design and Implementation of a Restaurant Ordering Management System** – ResearchGate
- **POS Systems and Their Impact on Restaurant Efficiency** – Journal of Hospitality & Tourism Research