MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

D.Y. PATIL POLYTECHNIC

MICRO PROJECT

Academic year: 2024-25

TITLE OF PROJECT : Payment Management System

Subject : Java Programming **Subject code: 314317**

Course : Computer Engineering **Course code: CO4K**

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr. **Lokare Dnyaneshwari Devidas** Roll No.**31** of Fourth Semester diploma in **Computer Engineering** of Institute, **D.Y. Patil Polytechnic** (Instt.Code: 0996) has completed the Micro-Project in course **Java Programming (314317)** for the academic year 2024-2025 as prescribed in the MSBTE curriculum of K Scheme.

Place: Ambi Enrollment No:**23212350299**

Date: …… Exam Seat No: **240706**

Subject Teacher	**Head of the Department**	**Principal**
(Prof.Pruthviraj Mankape)	(Prof.S.Shiwankar)	(Prof. S.V. Awachar)

Seal of institute

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr. **Borhade Soham Navnath** Roll No.**32** of Fourth Semester diploma in **Computer Engineering** of Institute, **D.Y. Patil Polytechnic** (Instt.Code: 0996) has completed the Micro-Project in course **Java Programming (314317)** for the academic year 2024-2025 as prescribed in the MSBTE curriculum of K Scheme.

Place: Ambi Enrollment No:**23212350300**

Date: …… Exam Seat No:**240707**

Subject Teacher	**Head of the Department**	**Principal**
(Prof.Pruthviraj Mankape)	(Prof.S.Shiwankar)	(Prof. S.V. Awachar)

Seal of institute

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr. **Tidke Sachin Ramesh** Roll No.**33** of Fourth Semester diploma in **Computer Engineering** of Institute, **D.Y. Patil Polytechnic** (Instt.Code: 0996) has completed the Micro-Project in course **Java Programming (314317)** for the academic year 2024-2025 as prescribed in the MSBTE curriculum of K Scheme.

Place: Ambi Enrollment No:**23212350301**

Date: …… Exam Seat No:**240708**

Subject Teacher	**Head of the Department**	**Principal**
(Prof.Pruthviraj Mankape)	(Prof.S.Shiwankar)	(Prof. S.V. Awachar)

Seal of institute

ACKNOWLEDGEMENT

It is a matter of great pleasure by getting the opportunity of highlighting. A fraction of knowledge, I acquired during our technical education through this project. This would not have been possible without the guidance and help of many people. This is the only page where we have opportunity of expressing our emotions and gratitude from the core of our heart to them. This project not have been success without enlightened ideas, timely suggestions and interest of our most respected guide "Prof. Pruthviraj Mankape " without his best guidance this would have been an impossible task to complete.

I would like to thank "Prof.S.Shiwankar" Head of our department for providing necessary facility using the period of working on this project work. I would also like to thank our Principal "Prof. S. V. Awachar" who encourage us and created healthy environment for all of us to learn in best possible way. Finally I would pay my respect and love to my parents and all other family members as we as friends for their love and encouragement throughout my career.

Student Names

1. Dnyaneshwari D. Lokare
2. Soham N. Borhade
3. Sachin R. Tidke

INDEX

Abstract

The Payment Management System is a Java-based application designed to manage and streamline the process of handling payments for individuals or organizations. This project aims to provide a simple, secure, and user-friendly interface for managing transactions, maintaining records, and generating payment reports.

The system is developed using Java as the core programming language, along with JDBC for database connectivity and MySQL as the backend database. The application enables users to perform operations such as adding customer details, processing payments, viewing transaction history, and generating summaries. It also includes features for admin-level access to manage user data and oversee payment activities.

This system enhances accuracy, reduces manual effort, and ensures efficient tracking of payments. It is especially useful for small businesses, educational institutions, or service providers who require a reliable and customizable solution for handling their payment processes.

Overall, this project demonstrates the use of Object-Oriented Programming, file/database handling, and GUI development (optional: using Swing or JavaFX) in Java, making it a practical and educational tool for students and developers alike.

Introduction

In today’s fast-paced digital world, managing payments efficiently essential for the smooth operation of businesses, institutions, and services. A Payment Management System is a software application that facilitates the handling of various types of financial transactions. It automates the process of recording, tracking, and managing payments, making it more reliable and less prone to human error.

This project, titled "Payment Management System", is developed using the Java programming language, which offers a robust, object-oriented approach to building flexible and scalable applications. The system is designed to provide users with functionalities such as registering customer details, processing payments, storing transaction data, and generating reports.

By integrating Java with a MySQL database using JDBC (Java Database Connectivity), the system ensures secure and consistent data handling. Additionally, the use of Java allows for future enhancements, such as adding a graphical user interface (GUI) using Swing or JavaFX, or expanding the system to support online payment gateways.

This project not only helps in understanding key Java concepts such as classes, objects, exception handling, and database connectivity but also provides a real-world use case that demonstrates how software can simplify complex financial tasks. The Payment Management System is a step toward creating efficient, accurate, and user-friendly payment solutions.

Software Requirement

1. Programming Language:

- **Java (JDK 8 or above)**
  - Used to develop the core logic and backend functionality of the application.

2. Database:

- **MySQL (Version 5.7 or above)**
  - Relational database used to store user data, payment details, and transaction history.

3. Database Connectivity:

- **JDBC (Java Database Connectivity)**
  - API used to connect the Java application to the MySQL database.

4. Development Environment / IDE:

- **Eclipse IDE** or **IntelliJ IDEA** or **NetBeans**
  - Used for writing, debugging, and running the Java code.
  - NetBeans also supports GUI development with Swing if needed.

5. Text Editor (Optional):

- **Notepad++**, **VS Code**, or any preferred text editor
  - For editing configuration files or SQL scripts.

6. MySQL Workbench or phpMyAdmin:

- Tools for managing the MySQL database, creating tables, and writing SQL queries.

7. Java Runtime Environment (JRE):

- Required to run the compiled Java application on any system.

8. Operating System:

- **Windows 7/10/11**, **Linux**, or **macOS**
  - The project is platform-independent as long as Java and MySQL are installed.

Methodology

The development of the **Payment Management System** follows a structured and step-by-step approach to ensure the application is efficient, reliable, and easy to use. The methodology adopted for this project is based on the **Software Development** Life Cycle **(SDLC),** specifically using the Waterfall Model, which includes the following key phases:

1. **Requirement Analysis:**
    - Understanding the core objectives of the system, such as storing customer information, managing payment transactions, and generating reports.
    - Identifying the users of the system (e.g., admin and users) and their roles.

2. **System Design:**
    - Designing the overall architecture of the application using object-oriented programming (OOP) concepts in Java.
    - Creating class diagrams and database schemas to represent entities such as Customer, Payment, and Transaction.
    - Planning the user interface (CLI or GUI) for smooth interaction.

3. **Implementation:**
    - Developing the application using Java for the backend logic and JDBC for connectivity with the MySQL database.
    - Writing modular and reusable code for different operations like adding customers, processing payments, and viewing reports.
    - Optional: Implementing a Graphical User Interface (GUI) using Java Swing or JavaFX for better usability.

4. **Database Design:**
    - Creating relational tables in MySQL to store customer records, payment details, and transaction history.
    - Ensuring proper normalization and indexing for efficient querying and data retrieval.

5. **Testing:**
    - Performing unit testing for individual modules and integration testing for the overall system.
    - Ensuring the application handles invalid inputs, exceptions, and edge cases properly.

6. **Deployment:**
    - Running the application in a real or simulated environment to demonstrate its functionalities.
    - Ensuring that the software runs smoothly on systems with Java Runtime Environment (JRE) installed.

7. **Documentation & Maintenance:**
    - Preparing proper documentation for users and developers for future reference.
    - Making the system flexible for updates like adding new payment modes or report types.

Code

```
import java.util.*;

class Customer {
    int id;
    String name;
    String email;

    Customer(int id, String name, String email) {
        this.id = id;
        this.name = name;
        this.email = email;
    }
}

class Payment {
    int paymentId;
    int customerId;
    double amount;
    Date date;

    Payment(int paymentId, int customerId, double amount, Date date) {
        this.paymentId = paymentId;
        this.customerId = customerId;
        this.amount = amount;
        this.date = date;
    }
```

```
}

public class PaymentManagementSystem {
    static Scanner sc = new Scanner(System.in);
    static List<Customer> customers = new ArrayList<>();
    static List<Payment> payments = new ArrayList<>();
    static int customerIdCounter = 1;
    static int paymentIdCounter = 1;

    public static void main(String[] args) {
        int choice;
        do {
            System.out.println("\n--- Payment Management System ---");
            System.out.println("1. Add Customer");
            System.out.println("2. Make Payment");
            System.out.println("3. View Payment History");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch (choice) {
                case 1:
                    addCustomer();
                    break;
                case 2:
                    makePayment();
                    break;
                case 3:
                    viewPayments();
                    break;
                case 4:
                    System.out.println("Exiting... Goodbye!");
```

```
                break;
            default:
                System.out.println("Invalid choice. Try again.");
        }
    } while (choice != 4);
}

static void addCustomer() {
    sc.nextLine(); // consume newline
    System.out.print("Enter customer name: ");
    String name = sc.nextLine();
    System.out.print("Enter customer email: ");
    String email = sc.nextLine();

    Customer customer = new Customer(customerIdCounter++, name, email);
    customers.add(customer);
    System.out.println("Customer added successfully! ID: " + customer.id);
}

static void makePayment() {
    System.out.print("Enter customer ID: ");
    int id = sc.nextInt();
    Customer customer = findCustomerById(id);
    if (customer == null) {
        System.out.println("Customer not found!");
        return;
    }

    System.out.print("Enter amount: ");
    double amount = sc.nextDouble();

    Payment payment = new Payment(paymentIdCounter++, id, amount, new
```

```
Date());
        payments.add(payment);
        System.out.println("Payment successful!");
    }

    static void viewPayments() {
        if (payments.isEmpty()) {
            System.out.println("No payments found.");
            return;
        }

        System.out.println("\n--- Payment History ---");
        for (Payment p : payments) {
            Customer c = findCustomerById(p.customerId);
            String name = (c != null) ? c.name : "Unknown";
            System.out.println("Payment ID: " + p.paymentId);
            System.out.println("Customer Name: " + name);
            System.out.println("Amount: ₹" + p.amount);
            System.out.println("Date: " + p.date);
            System.out.println("---------------------- ");
        }
    }

    static Customer findCustomerById(int id) {
        for (Customer c : customers) {
            if (c.id == id) return c;
        }
        return null;
    }
}
```

Output

```
--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice: 1
Enter customer name: Soham Borhade
Enter customer email: soham@gmail.com
Customer added successfully! ID: 1

--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice: 1
Enter customer name: Sachin
Enter customer email: Sachin@gmail.com
Customer added successfully! ID: 2

--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice: 1
Enter customer name: Dnyaneshwari
Enter customer email: Dnyan@gmail.com
Customer added successfully! ID: 3

--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice: 2
Enter customer ID: 1
```

```
--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice: 2
Enter customer ID: 1
Enter amount: 2500
Payment successful!

--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice: 3

--- Payment History ---
Payment ID: 1
Customer Name: Soham Borhade
Amount: ?2500.0
Date: Thu Apr 10 20:10:05 IST 2025
------------------------

--- Payment Management System ---
1. Add Customer
2. Make Payment
3. View Payment History
4. Exit
Enter your choice:
```

References

1. Software Resources:

- **Java Development Kit (JDK 8 or above):**
  Required to compile and run Java programs.

- **Visual Studio Code (VS Code):**
  A lightweight and powerful code editor used for writing, editing, and debugging Java code.

- **Java Extension Pack for VS Code:**
  Includes essential extensions such as:
  - Language Support for Java™ by Red Hat
  - Debugger for Java
  - Java Test Runner
  - Maven for Java *(optional)*

- **MySQL Server:**
  A relational database system used to store and manage payment-related data.

- **MySQL Command Line Client / MySQL Workbench:**
  Tools used to interact with the database for creating tables and managing data.

- **JDBC (Java Database Connectivity):**
  Java API used to connect the Java program with the MySQL database.

2. Hardware Resources:

- **Computer/Laptop** with minimum specifications:
  - **Processor:** Intel i3 or above
  - **RAM:** 4 GB or more
  - **Storage:** Minimum 500 MB of free space
  - **Operating System:** Windows, Linux, or macOS

3. Documentation & Learning Platforms:

- **Oracle Java Documentation:**
  Official reference for Java syntax, libraries, and API usage.
  https://docs.oracle.com/javase/
- **MySQL Documentation:**
  For SQL syntax and MySQL usage guidelines.
  https://dev.mysql.com/doc/
- **Online Learning Platforms:**
  - GeeksforGeeks – Java + JDBC Tutorials
  - W3Schools – SQL Basics
  - Stack Overflow – for debugging and issue resolution

Conclusion

The Payment Management System project successfully demonstrates the use of core Java concepts such as classes, objects, collections, control structures, and user input handling to build a functional application without relying on external databases. It provides a simple yet effective way to manage customers and their payments through a console-based interface.

This project helped in understanding real-world application logic, data organization, and how to interact with users through a menu-driven approach. It also emphasized the importance of modular design and clean coding practices.

Overall, the system meets its basic objectives of:

- Adding and managing customers
- Recording payment transactions
- Displaying a history of payments

The project lays a strong foundation for future enhancements such as file storage, GUI development, or integration with a database system for persistent data management.

Weekly Progress Report

Sr.No.	Week	Activity Performed	Sign of Guide	Date
1	1st	Discussion and finalization of topic		
2	2nd	Preparation and submission of Abstract		
3	3rd	Literature Review		
4	4th	Collection of Data		
5	5th	Collection of Data		
6	6th	Discussion and outline of Content		
7	7th	Formulation of Content		
8	8th	Editing and proof Reading ofContent		
9	9th	Compilation of Report And Presentation		
10	10th	Seminar		
11	11th	Viva voce		
12	12th	Final submission of Micro-Project		

Sign of the student

Sign of the faculty

(Prof. Pruthviraj Mankape)

ANNEXURE II

Evaluation Sheet for the Micro Project

Academic Year: 2024-2025 **Name of Faculty: Prof.Pruthviraj Mankape**

Course : Computer Engineering **Course code: CO4K**

Semester: IV

Subject Name: Java Programming (314317)

Major learning outcomes achieved by students by doing the project

- **Practical outcome:**
  The practical outcome is selective data routing, where a single input is directed to one of four outputs based on control signals.
- **Course outcome:**
  Understanding English language for speaking

Outcomes in Affective domain: Function as team member Follow Ethics

Comments/suggestions about team work /leadership/inter-personal communication (if any)

……3

Roll No	Student Name	Marks out of 6 for performance in group activity	Marks out of 4 for performance in oral/ presentation	Total out of 10
31	Lokare Dnyneshwari Devidas			
32	Borhade Soham Navnath			
33	Tidke Sachin Ramesh			

Signature of Faculty
(Prof. Pruthviraj Mankape)