MICROFINANCE LOANS & REPAYMENTS – SPECIFICATION DOCUMENT

Problem Statement:

Design and implement a Java console application for a Microfinance system that

manages customers, loan products, loans, disbursements, EMIs, repayments, and

penalties. The application should demonstrate object-oriented principles and ensure

accurate schedules and balances.

Class Requirements:

1. Customer

2. Loan Product

3. Loan

4. Disbursement

5. Repayment Schedule

6. Repayment

7. Penalty

Business Rules:

1. Loan eligibility must be verified before approval.

2. Disbursement activates the repayment schedule immediately.

3. Repayments reduce outstanding principal and interest as per schedule.

4. Late or partial payments may trigger penalties.

5. Each repayment must link to a loan and scheduled installment.

Console Interface Requirements:

1. Menu-driven program: Add Customer / Add Loan Product / Approve Loan /

Disburse Loan / Generate Schedule / Record Repayment / View Outstanding /

Exit

2. Input validations must be performed for all user entries.

3. Encapsulation must be followed for all attributes.

Expected Output Behavior:

• Show amortization schedule and current outstanding.

• Show repayment receipt with principal/interest split and penalties (if any).

• Show loan status (active/closed/in arrears).

Questions for Students:

1. Draw the UML Class Diagram for the above system.

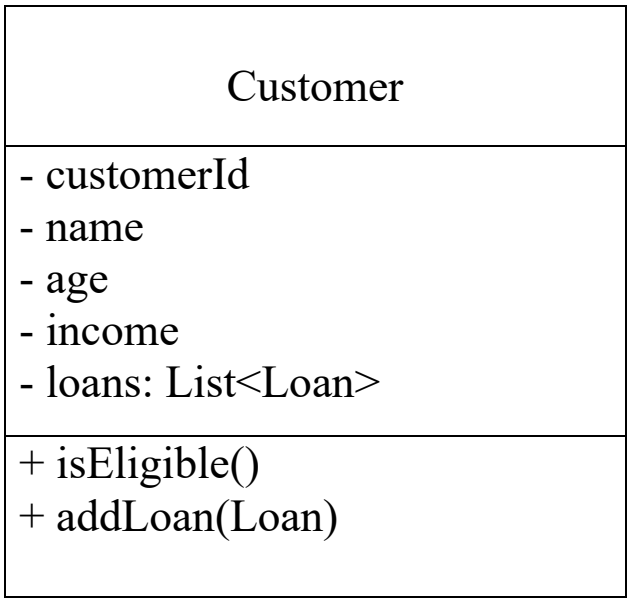
2. Implement the Classes with necessary Data Members and Methods for System

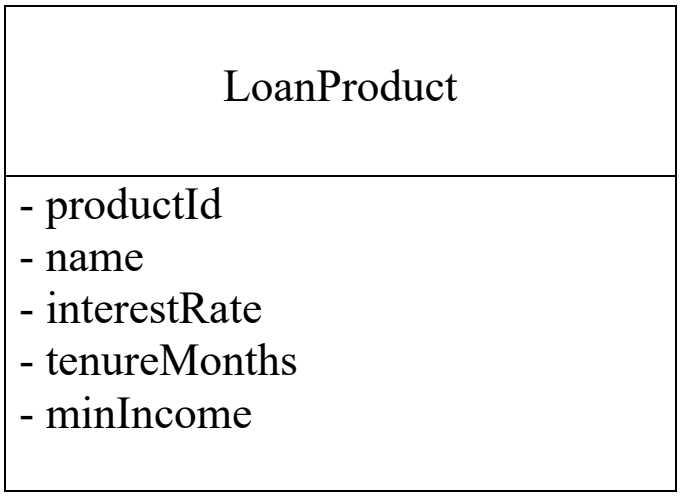
and Business Rules.

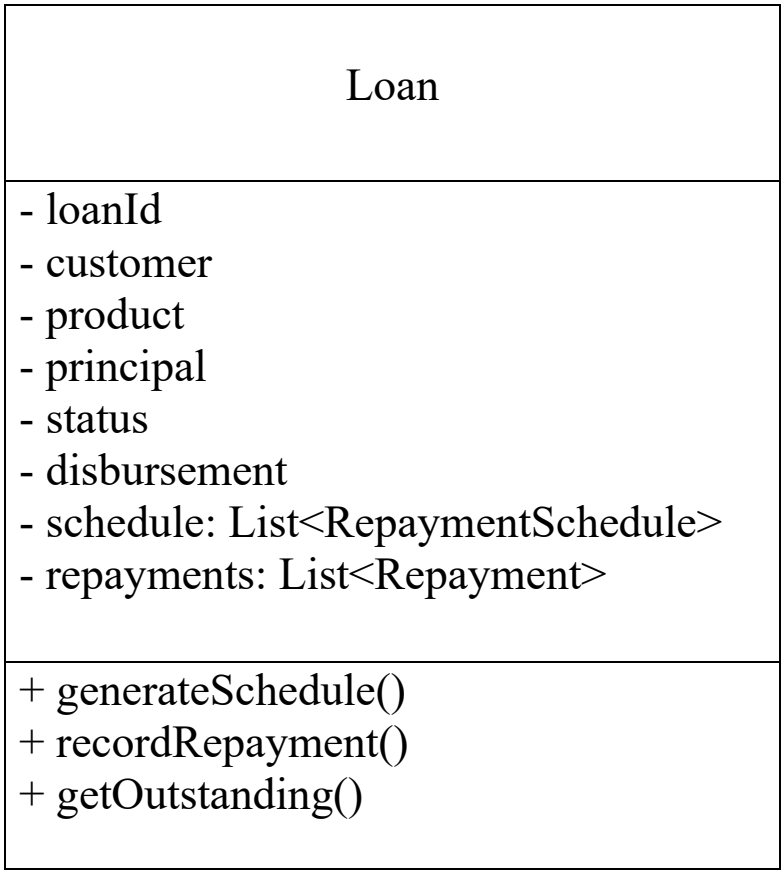
3. Use Aggregation, Inheritance and Polymorphism wherever required.

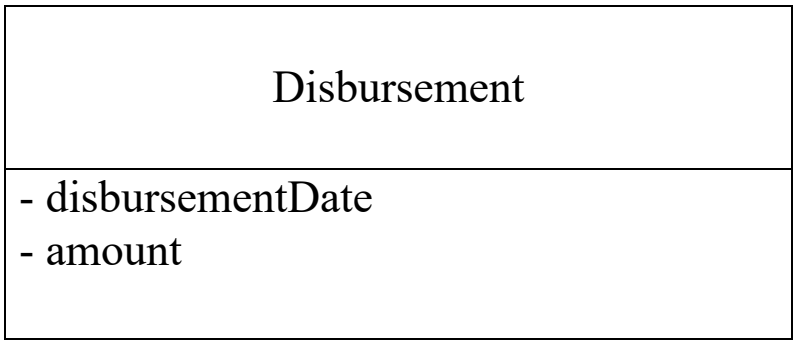
4. Implement the main method for Menu Driven System.

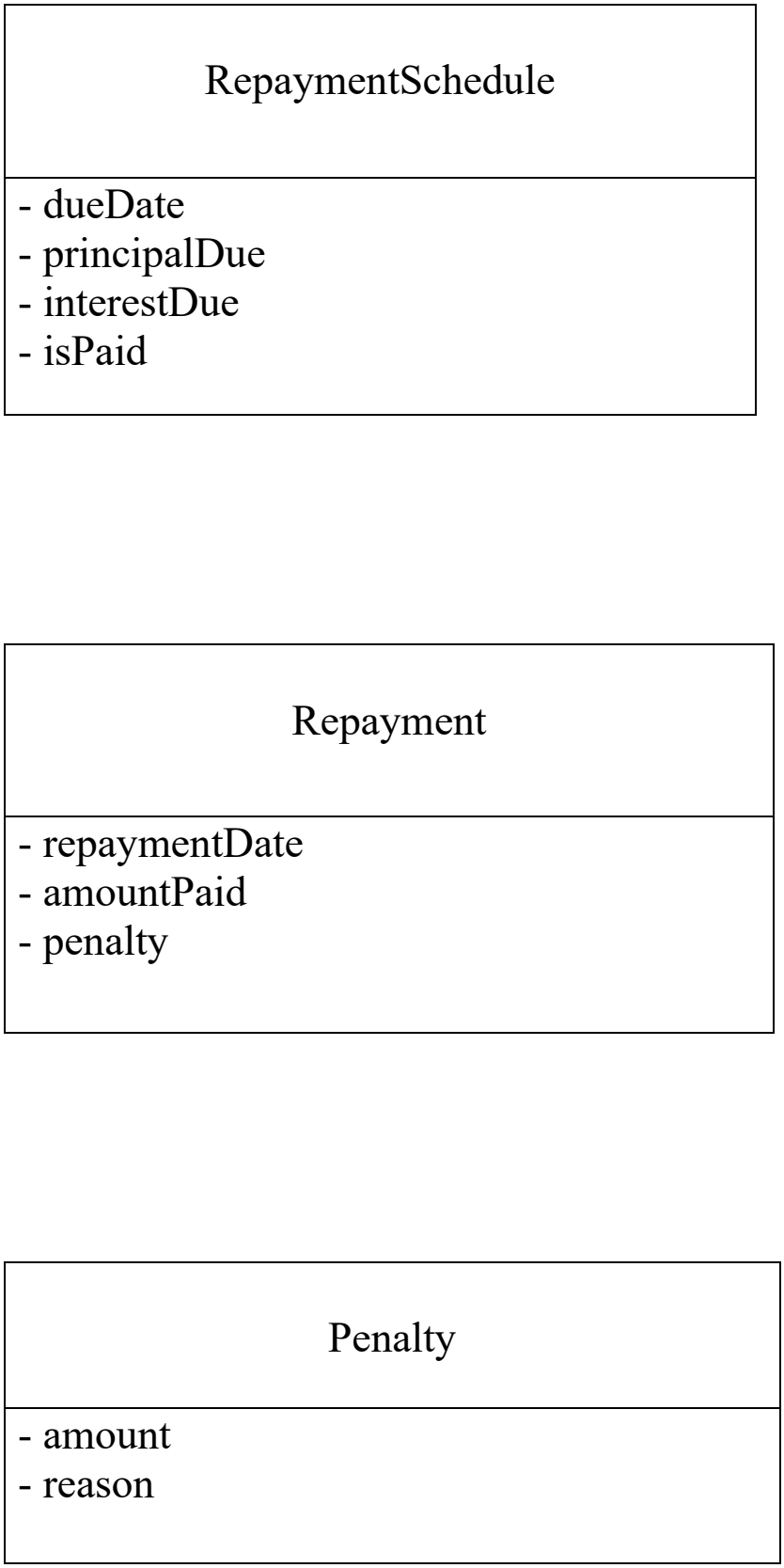
//==========UML Class Diagram==========//






Loan
- loanId
- customer
- product
- principal
- status
- disbursement
- schedule: List<RepaymentSchedule>
- repayments: List<Repayment>
+ generateSchedule()
+ recordRepayment()
+ getOutstanding()


Disbursement
- disbursementDate
- amount


RepaymentSchedule
- dueDate
- principalDue
- interestDue
- isPaid
Repayment
- repaymentDate
- amountPaid
- penalty
Penalty
- amount
- reason

//==========Java Class Implementations==========//

1. Customer.java

```
import java.util.*;
public class Customer {
private String customerId;
private String name;
private int age;
private double income;
private List<Loan> loans;
public Customer(String customerId, String name, int age, double
income) {
this.customerId = customerId;
this.name = name;
this.age = age;
this.income = income;
this.loans = new ArrayList<>();
}
public boolean isEligible(LoanProduct product) {
return this.income >= product.getMinIncome();
}
public void addLoan(Loan loan) {
loans.add(loan);
```

```
}
// Getters
public String getCustomerId() { return customerId; }
public String getName() { return name; }
public double getIncome() { return income; }
}
```

2. LoanProduct.java

```
public class LoanProduct {
    private String productId;
    private String name;
    private double interestRate;
    private int tenureMonths;
    private double minIncome;
    public LoanProduct(String productId, String name, double
interestRate, int tenureMonths, double minIncome) {
        this.productId = productId;
        this.name = name;
        this.interestRate = interestRate;
        this.tenureMonths = tenureMonths;
        this.minIncome = minIncome;
    }
```

```
    // Getters
    public double getInterestRate() { return interestRate; }
    public int getTenureMonths() { return tenureMonths; }
    public double getMinIncome() { return minIncome; }
    public String getProductId() { return productId; }
}
```

3. Loan.java

```
import java.util.*;
public class Loan {
    private String loanId;
    private Customer customer;
    private LoanProduct product;
    private double principal;
    private String status; // active, closed, in_arrears
    private Disbursement disbursement;
    private List<RepaymentSchedule> schedule = new
ArrayList<>();
    private List<Repayment> repayments = new ArrayList<>();
    public Loan(String loanId, Customer customer, LoanProduct
product, double principal) {
        this.loanId = loanId;
        this.customer = customer;
```

```
        this.product = product;
        this.principal = principal;
        this.status = "Pending";
    }
    public void disburse(Date disbursementDate) {
        this.disbursement = new Disbursement(disbursementDate,
principal);
        generateSchedule(disbursementDate);
        this.status = "Active";
    }
    private void generateSchedule(Date startDate) {
        double monthlyInterest = (principal *
product.getInterestRate()) / 12 / 100;
        double monthlyPrincipal = principal /
product.getTenureMonths();
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(startDate);
        for (int i = 0; i < product.getTenureMonths(); i++) {
            calendar.add(Calendar.MONTH, 1);
            schedule.add(new
RepaymentSchedule(calendar.getTime(), monthlyPrincipal,
monthlyInterest));
        }
    }
    public void recordRepayment(double amountPaid, Date date) {
```

```
        for (RepaymentSchedule rs : schedule) {
            if (!rs.isPaid()) {
                double totalDue = rs.getPrincipalDue() +
rs.getInterestDue();
                Penalty penalty = null;
                if (amountPaid < totalDue) {
                    penalty = new Penalty(50.0, "Partial Payment");
                }
                rs.setPaid(true);
                repayments.add(new Repayment(date, amountPaid,
penalty));
                break;
            }
        }
    }
    public double getOutstanding() {
        double totalOutstanding = 0;
        for (RepaymentSchedule rs : schedule) {
            if (!rs.isPaid()) {
                totalOutstanding += rs.getPrincipalDue() +
rs.getInterestDue();
            }
        }
        return totalOutstanding;
    }
```

```
    public String getStatus() { return status; }
    public String getLoanId() { return loanId; }
    public Customer getCustomer() { return customer; }
}
```

4. Disbursement.java

```
import java.util.Date;
public class Disbursement {
    private Date disbursementDate;
    private double amount;
    public Disbursement(Date disbursementDate, double amount) {
        this.disbursementDate = disbursementDate;
        this.amount = amount;
    }
}
```

5. RepaymentSchedule.java

```
import java.util.Date;
public class RepaymentSchedule {
    private Date dueDate;
    private double principalDue;
    private double interestDue;
```

```
    private boolean isPaid;
    public RepaymentSchedule(Date dueDate, double
principalDue, double interestDue) {
        this.dueDate = dueDate;
        this.principalDue = principalDue;
        this.interestDue = interestDue;
        this.isPaid = false;
    }
    public boolean isPaid() { return isPaid; }
    public void setPaid(boolean paid) { this.isPaid = paid; }
    public double getPrincipalDue() { return principalDue; }
    public double getInterestDue() { return interestDue; }
}
```

6. Repayment.java

```
import java.util.Date;
public class Repayment {
    private Date repaymentDate;
    private double amountPaid;
    private Penalty penalty;
    public Repayment(Date repaymentDate, double amountPaid,
Penalty penalty) {
        this.repaymentDate = repaymentDate;
```

```
        this.amountPaid = amountPaid;
        this.penalty = penalty;
    }
}
```

7. Penalty.java

```
public class Penalty {
    private double amount;
    private String reason;
    public Penalty(double amount, String reason) {
        this.amount = amount;
        this.reason = reason;
    }
}
```

//==========Menu-Driven Main Method==========//

MicrofinanceApp.java

```
import java.util.*;

public class MicrofinanceApp {
    static Scanner sc = new Scanner(System.in);
    static List<Customer> customers = new ArrayList<>();
    static List<LoanProduct> products = new ArrayList<>();
    static List<Loan> loans = new ArrayList<>();

    public static void main(String[] args) {
        while (true) {
            System.out.println("\n--- Microfinance Loan System ---");
            System.out.println("1. Add Customer");
            System.out.println("2. Add Loan Product");
            System.out.println("3. Approve Loan");
            System.out.println("4. Disburse Loan");
            System.out.println("5. Generate Schedule");
            System.out.println("6. Record Repayment");
            System.out.println("7. View Outstanding");
            System.out.println("8. Exit");
```

```
            int choice = sc.nextInt();
            sc.nextLine(); // consume newline

            switch (choice) {
                case 1 -> addCustomer();
                case 2 -> addLoanProduct();
                case 3 -> approveLoan();
                case 4 -> disburseLoan();
                case 5 -> generateSchedule();
                case 6 -> recordRepayment();
                case 7 -> viewOutstanding();
                case 8 -> System.exit(0);
                default -> System.out.println("Invalid choice!");
            }
        }
    }

    private static void addCustomer() {
        System.out.print("Enter ID: ");
        String id = sc.nextLine();
        System.out.print("Enter name: ");
        String name = sc.nextLine();
        System.out.print("Enter age: ");
        int age = sc.nextInt();
```

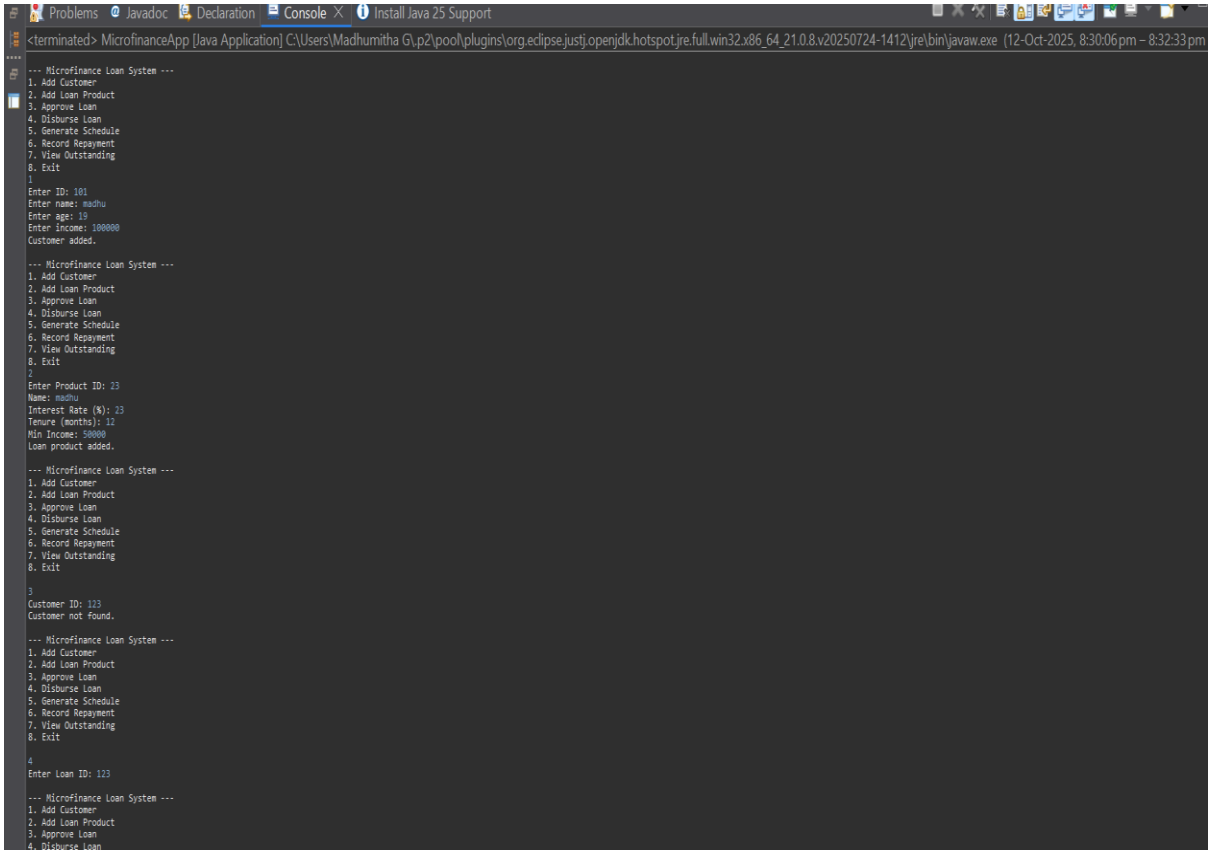
```
        System.out.print("Enter income: ");
        double income = sc.nextDouble();
        customers.add(new Customer(id, name, age, income));
        System.out.println("Customer added.");
    }

    private static void addLoanProduct() {
        System.out.print("Enter Product ID: ");
        String pid = sc.nextLine();
        System.out.print("Name: ");
        String name = sc.nextLine();
        System.out.print("Interest Rate (%): ");
        double rate = sc.nextDouble();
        System.out.print("Tenure (months): ");
        int tenure = sc.nextInt();
        System.out.print("Min Income: ");
        double minIncome = sc.nextDouble();
        products.add(new LoanProduct(pid, name, rate, tenure,
minIncome));
        System.out.println("Loan product added.");
    }

    private static void approveLoan() {
        System.out.print("Customer ID: ");
```

```
        String cid = sc.nextLine();
        Customer customer = customers.stream().filter(c ->
c.getCustomerId().equals(cid)).findFirst().orElse(null);
        if (customer == null) {
            System.out.println("Customer not found.");
            return;
        }

        System.out.print("Loan Product ID: ");
        String pid = sc.nextLine();
        LoanProduct product = products.stream().filter(p ->
p.getProductId().equals(pid)).findFirst().orElse(null);
        if (product == null) {
            System.out.println("Product not found.");
            return;
        }

        if (!customer.isEligible(product)) {
            System.out.println("Customer is not eligible.");
            return;
        }

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

```
        Loan loan = new Loan("L" + (loans.size() + 1), customer,
product, amount);
        loans.add(loan);
        customer.addLoan(loan);
        System.out.println("Loan approved.");
    }

    private static void disburseLoan() {
        Loan loan = selectLoan();
        if (loan != null) {
            loan.disburse(new Date());
            System.out.println("Loan disbursed and schedule
activated.");
        }
    }

    private static void generateSchedule() {
        Loan loan = selectLoan();
        if (loan != null) {
            System.out.println("Outstanding: " +
loan.getOutstanding());
        }
    }

    private static void recordRepayment() {
```

```
        Loan loan = selectLoan();
        if (loan != null) {
            System.out.print("Amount Paid: ");
            double amt = sc.nextDouble();
            loan.recordRepayment(amt, new Date());
            System.out.println("Repayment recorded.");
        }
    }

    private static void viewOutstanding() {
        Loan loan = selectLoan();
        if (loan != null) {
            System.out.println("Outstanding balance: " +
loan.getOutstanding());
        }
    }

    private static Loan selectLoan() {
        System.out.print("Enter Loan ID: ");
        String lid = sc.nextLine();
        return loans.stream().filter(l ->
l.getLoanId().equals(lid)).findFirst().orElse(null);
    }
}
```

//====================OUTPUT====================//


Problems
Javadoc
Declaration
Console
Install Java 25 Support
<terminated> MicrofinanceApp [Java Application] C:\Users\Madhumitha G\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20250724-1412\jre\bin\javaw.exe (12-Oct-2025, 8:30:06 pm – 8:32:33 pm
--- Microfinance Loan System ---
1. Add Customer
2. Add Loan Product
3. Approve Loan
4. Disburse Loan
5. Generate Schedule
6. Record Repayment
7. View Outstanding
8. Exit
1
Enter ID: 101
Enter name: madhu
Enter age: 19
Enter income: 100000
Customer added.
--- Microfinance Loan System ---
1. Add Customer
2. Add Loan Product
3. Approve Loan
4. Disburse Loan
5. Generate Schedule
6. Record Repayment
7. View Outstanding
8. Exit
2
Enter Product ID: 23
Name: madhu
Interest Rate (%): 23
Tenure (months): 12
Min Income: 50000
Loan product added.
--- Microfinance Loan System ---
1. Add Customer
2. Add Loan Product
3. Approve Loan
4. Disburse Loan
5. Generate Schedule
6. Record Repayment
7. View Outstanding
8. Exit
3
Customer ID: 123
Customer not found.
--- Microfinance Loan System ---
1. Add Customer
2. Add Loan Product
3. Approve Loan
4. Disburse Loan
5. Generate Schedule
6. Record Repayment
7. View Outstanding
8. Exit
4
Enter Loan ID: 123
--- Microfinance Loan System ---
1. Add Customer
2. Add Loan Product
3. Approve Loan
4. Disburse Loan