import java.nio.file.\*;

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

import java.io.IOException;

import java.io.PrintWriter;

class Bank {

private Map<Integer, User> accounts;

private int accountCounter;

private Scanner scanner;

private final String FILE\_NAME = "bank\_data.txt";

public Bank() {

this.accounts = new HashMap<>();

this.accountCounter = 1000;

this.scanner = new Scanner(System.in);

loadUserData();

}

private void loadUserData() {

try {

if (Files.exists(Paths.get(FILE\_NAME))) {

Scanner fileScanner = new Scanner(Paths.get(FILE\_NAME));

while (fileScanner.hasNext()) {

String[] userData = fileScanner.nextLine().split(",");

int accountNumber = Integer.parseInt(userData[0]);

String accountHolderName = userData[1];

String password = userData[2];

double balance = Double.parseDouble(userData[3]);

String accountType = userData[4];

User loadedUser = new User(accountNumber, accountHolderName, password, balance, accountType);

accounts.put(accountNumber, loadedUser);

}

fileScanner.close();

}

} catch (IOException | java.util.InputMismatchException | NumberFormatException e) {

e.printStackTrace();

}

}

private void saveUserData() {

try {

PrintWriter writer = new PrintWriter(FILE\_NAME);

for (User user : accounts.values()) {

writer.println(user.getAccountNumber() + "," +

user.getAccountHolderName() + "," +

user.getPassword() + "," +

user.getBalance() + "," +

user.getAccountType());

}

writer.close();

} catch (IOException e) {

e.printStackTrace();

}

}

public void createAccount() {

int accountNumber = ++accountCounter;

System.out.print("Enter account holder's name: ");

String accountHolderName = scanner.next();

System.out.print("Choose account type (savings or current): ");

String accountType = scanner.next();

System.out.print("Set a password for your account: ");

String password = scanner.next();

double initialBalance = accountType.equalsIgnoreCase("savings") ? 500 : 0;

User newUser = new User(accountNumber, accountHolderName, password, initialBalance, accountType);

accounts.put(accountNumber, newUser);

saveUserData();

System.out.println("=============================================");

System.out.println("Account created successfully. Your account number is: " + accountNumber);

System.out.println("=============================================");

}

public boolean login() {

System.out.print("Enter your account number: ");

int accountNumber = scanner.nextInt();

if (accounts.containsKey(accountNumber)) {

User user = accounts.get(accountNumber);

System.out.print("Enter your password: ");

String passwordAttempt = scanner.next();

if (user.authenticate(passwordAttempt)) {

System.out.println("=============================================");

System.out.println("Login successful. Welcome, " + user.getAccountHolderName() + "!");

System.out.println("=============================================");

return true;

} else {

System.out.println("=============================================");

System.out.println("Incorrect password. Login failed.");

System.out.println("=============================================");

}

} else {

System.out.println("=============================================");

System.out.println("Account not found. Please check your account number.");

System.out.println("=============================================");

}

return false;

}

public void viewAccountInfo(int accountNumber) {

if (accounts.containsKey(accountNumber)) {

User user = accounts.get(accountNumber);

System.out.println("=============================================");

System.out.println("Account Information for Account Number " + accountNumber);

System.out.println("Account Holder: " + user.getAccountHolderName());

System.out.println("Account Type: " + user.getAccountType());

System.out.println("Balance: Rs " + user.getBalance());

System.out.println("Transaction History:\n" + user.getTransactionHistory());

System.out.println("=============================================");

// Clear the transaction history after printing it

user.clearTransactionHistory();

} else {

System.out.println("=============================================");

System.out.println("Account not found.");

System.out.println("=============================================");

}

}

public void applyForLoan(int accountNumber, double loanAmount) {

if (accounts.containsKey(accountNumber)) {

User user = accounts.get(accountNumber);

System.out.print("Enter your loan application password: ");

String passwordAttempt = scanner.next();

if (user.authenticate(passwordAttempt)) {

double interestRate = 0.065; // 6.5%

double interest = loanAmount \* interestRate;

double totalLoanAmount = loanAmount + interest;

user.addTransaction(loanAmount);

System.out.println("=============================================");

System.out.println("Loan of Rs " + loanAmount + " approved.");

System.out.println("Interest charged: Rs " + interest);

System.out.println("Total Loan Amount (including interest): Rs " + totalLoanAmount);

System.out.println("=============================================");

} else {

System.out.println("=============================================");

System.out.println("Incorrect password. Loan application failed.");

System.out.println("=============================================");

}

} else {

System.out.println("=============================================");

System.out.println("Account not found.");

System.out.println("=============================================");

}

}

public void withdraw(int accountNumber, double amount) {

if (accounts.containsKey(accountNumber)) {

User user = accounts.get(accountNumber);

System.out.print("Enter your withdrawal password: ");

String passwordAttempt = scanner.next();

if (user.authenticate(passwordAttempt)) {

if (user.getBalance() >= amount) {

user.addTransaction(-amount);

System.out.println("=============================================");

System.out.println("Withdrawal successful. New balance: Rs " + user.getBalance());

System.out.println("=============================================");

} else {

System.out.println("=============================================");

System.out.println("Insufficient funds for withdrawal.");

System.out.println("=============================================");

}

} else {

System.out.println("=============================================");

System.out.println("Incorrect password. Withdrawal failed.");

System.out.println("=============================================");

}

} else {

System.out.println("=============================================");

System.out.println("Account not found.");

System.out.println("=============================================");

}

}

public void credit(int accountNumber, double amount) {

if (accounts.containsKey(accountNumber)) {

User user = accounts.get(accountNumber);

// No password check for credit operation

user.addTransaction(amount);

System.out.println("=============================================");

System.out.println("Credit successful. New balance: Rs " + user.getBalance());

System.out.println("=============================================");

} else {

System.out.println("=============================================");

System.out.println("Account not found.");

System.out.println("=============================================");

}

}

}

class User {

private int accountNumber;

private String accountHolderName;

private String password;

private double balance;

private String accountType;

private StringBuilder transactionHistory;

public User(int accountNumber, String accountHolderName, String password, double initialBalance, String accountType) {

this.accountNumber = accountNumber;

this.accountHolderName = accountHolderName;

this.password = password;

this.balance = initialBalance;

this.accountType = accountType;

this.transactionHistory = new StringBuilder("Transaction History:\n");

}

public int getAccountNumber() {

return accountNumber;

}

public String getAccountHolderName() {

return accountHolderName;

}

public double getBalance() {

return balance;

}

public String getAccountType() {

return accountType;

}

public StringBuilder getTransactionHistory() {

return transactionHistory;

}

public void addTransaction(double amount) {

balance += amount;

transactionHistory.append("Transaction: +").append(amount).append(", New Balance: Rs ").append(balance).append("\n");

}

public boolean authenticate(String passwordAttempt) {

return password.equals(passwordAttempt);

}

public String getPassword() {

return password;

}

public void clearTransactionHistory() {

transactionHistory = new StringBuilder("Transaction History:\n");

}

}

public class Main {

public static void main(String[] args) {

Bank bank = new Bank();

Scanner scanner = new Scanner(System.in);

System.out.println("=============================================");

System.out.println("Welcome to the Online Banking System!");

System.out.println("=============================================");

int choice;

do {

System.out.println("\nChoose an option:");

System.out.println("1. Create Account");

System.out.println("2. Login");

System.out.println("3. Apply for Loan");

System.out.println("4. Withdraw");

System.out.println("5. Credit");

System.out.println("6. Exit");

System.out.print("Enter your choice: ");

choice = scanner.nextInt();

switch (choice) {

case 1:

bank.createAccount();

break;

case 2:

boolean loggedIn = bank.login();

if (loggedIn) {

System.out.print("Do you want to view account information? (y/n): ");

char viewInfoChoice = scanner.next().charAt(0);

if (viewInfoChoice == 'y' || viewInfoChoice == 'Y') {

System.out.print("Enter your account number: ");

int accountNumber = scanner.nextInt();

bank.viewAccountInfo(accountNumber);

}

}

break;

case 3:

System.out.print("Enter your account number: ");

int loanAccountNumber = scanner.nextInt();

System.out.print("Enter loan amount: Rs ");

double loanAmount = scanner.nextDouble();

bank.applyForLoan(loanAccountNumber, loanAmount);

break;

case 4:

System.out.print("Enter your account number: ");

int withdrawAccountNumber = scanner.nextInt();

System.out.print("Enter withdrawal amount: Rs ");

double withdrawAmount = scanner.nextDouble();

bank.withdraw(withdrawAccountNumber, withdrawAmount);

break;

case 5:

System.out.print("Enter your account number: ");

int creditAccountNumber = scanner.nextInt();

System.out.print("Enter credit amount: Rs ");

double creditAmount = scanner.nextDouble();

bank.credit(creditAccountNumber, creditAmount);

break;

case 6:

System.out.println("=============================================");

System.out.println("Exiting the Online Banking System. Thank you!");

System.out.println("=============================================");

break;

default:

System.out.println("Invalid choice. Please enter a valid option.");

}

} while (choice != 6);

}

}