CERTIFICATE

This is to certify that [Your Name], a student of Class XII-[Section] at [Your School Name], has successfully completed the project titled **“Bank Management System”** under the supervision of [Project Supervisor Name] during the academic year [2025-26].

This project has been completed in fulfillment of the requirements for the Computer Science course and adheres to the guidelines issued by CBSE, New Delhi.

The project work meets the expected standards and has been evaluated by the undersigned.

Signature of Principal: __________________________

Signature of Supervisor: _________________________

Signature of External Examiner: ________________

Date: ______________

Place: ______________

Student Details:

Student Names:

Class : XII

Subject: Information Technology

School Name: Torch Bearers Convent School

Affiliation By.: CBSE Board

ABSTRACT

The Bank Management System is a Java-based console application designed to simulate the core operations of a real-world banking system. It uses fundamental programming concepts such as classes, objects, switch-case statements, and loops to manage bank accounts and transactions efficiently. The system allows users to create new accounts, deposit and withdraw funds, view account details, and perform basic transaction management in a user-friendly, menu-driven environment.

This project aims to provide a practical understanding of object-oriented programming principles and control structures while delivering a functional banking tool for educational purposes. It models the essential features of a bank, enabling users to interact with account data securely and effectively. The modular design promotes code clarity and extensibility, allowing future enhancements such as transaction history or multi-user login.

Developed by team members Riya (Nagar), Vandana, Vanshika, and Yashika, this system serves as a foundational learning project demonstrating how programming constructs can be applied to solve real-world problems in banking operations.

Table of Contents

Introduction

This Bank Management System project serves as a basic simulation of banking operations. It provides functionalities for managing multiple customer accounts, performing deposits, withdrawals, and viewing balances. The system uses classes and objects to model real-world entities. Switch-case statements and loops are employed to handle user menu navigation and repetitive tasks efficiently.

Objectives

- To implement a user-friendly banking system using Java programming language.
- To understand and apply fundamental concepts like classes, objects, switch-case, and loops.
- To simulate key banking operations: create accounts, deposit money, withdraw money, check balance.
- To develop modular and organized code promoting reusability and clarity.
- To provide transaction management mimicking real-world banking scenarios.

System Design and Classes

Main Classes:

- **Account:** Stores customer details like name, account number, account type, and balance. It provides methods for deposit, withdrawal, and displaying account details.
- **Bank:** Manages all accounts, enabling addition, deletion, search, and transaction operations across accounts.
  Key Features:
- Use of constructors to initialize object state.
- Encapsulation of data with private attributes and public getter/setter methods.
- Utilization of switch-case for menu-driven operations.
- Loops for repeated menu display and batch processing of accounts.

Core Components and Code Snippets

- Account

```java
/**
 * Represents a bank account with basic operations.
 */
public class Account {
    private String accountNumber;
    private String accountHolder;
    private double balance;

    // Constructor to initialize account details
    public Account(String accountNumber, String accountHolder, double balance) {
        this.accountNumber = accountNumber;
        this.accountHolder = accountHolder;
        this.balance = balance;
    }

    // Getters
    public String getAccountNumber() { return accountNumber; }
    public String getAccountHolder() { return accountHolder; }
    public double getBalance() { return balance; }

    // Deposit money to account
    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount);
        } else {
            System.out.println("Enter a positive amount.");
        }
    }

    // Withdraw money from account if sufficient balance
    public boolean withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
            return true;
        }
        System.out.println("Insufficient balance or invalid amount.");
        return false;
    }

    // Display account info
    public String getAccountInfo() {
        return "Account No: " + accountNumber + ", Holder: " + accountHolder + ",
Balance: " + balance;
    }
}
```

- Bank

```java
import java.util.ArrayList;
import java.util.List;

/**
 * Manages multiple accounts and their operations.
 */
public class Bank {
    private List<Account> accounts;

    public Bank() {
        accounts = new ArrayList<>();
    }

    // Add new account
    public void addAccount(Account account) {
        accounts.add(account);
        System.out.println("Account added: " + account.getAccountNumber());
    }

    // Search account by account number
    public Account findAccount(String accountNumber) {
        for (Account acc : accounts) {
            if (acc.getAccountNumber().equals(accountNumber)) {
                return acc;
            }
        }
        return null;
    }

    // Show all accounts
    public void displayAllAccounts() {
        for (Account acc : accounts) {
            System.out.println(acc.getAccountInfo());
        }
    }
}
```

- Transaction

```java
/**
 * Handles transactions like deposit and withdrawal.
 */
public class Transaction {

    // Perform deposit transaction
    public static void deposit(Account account, double amount) {
        account.deposit(amount);
    }

    // Perform withdrawal transaction
    public static void withdraw(Account account, double amount) {
        account.withdraw(amount);
    }
}
```

-
- User interface

```java
import java.util.Scanner;

/**
 * Handles user interaction and menu display.
 */
public class UserInterface {
    private Bank bank;
    private Scanner scanner;

    public UserInterface(Bank bank) {
        this.bank = bank;
        scanner = new Scanner(System.in);
    }

    public void start() {
        while (true) {
            System.out.println("\nBank Management System Menu");
            System.out.println("1. Create Account");
            System.out.println("2. Deposit Money");
            System.out.println("3. Withdraw Money");
            System.out.println("4. Display All Accounts");
            System.out.println("5. Exit");
            System.out.print("Enter choice: ");

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

            switch (choice) {
                case 1: createAccount(); break;
                case 2: depositMoney(); break;
                case 3: withdrawMoney(); break;
                case 4: bank.displayAllAccounts(); break;
                case 5: System.out.println("Exiting..."); scanner.close();
System.exit(0);
                default: System.out.println("Invalid choice.");
            }
        }
    }

    private void createAccount() {
        System.out.print("Enter Account Number: ");
        String accNum = scanner.nextLine();
        System.out.print("Enter Account Holder Name: ");
        String accHolder = scanner.nextLine();
        System.out.print("Enter Initial Deposit: ");
        double initialDeposit = scanner.nextDouble();
        scanner.nextLine();

        Account account = new Account(accNum, accHolder, initialDeposit);
        bank.addAccount(account);
    }

    private void depositMoney() {
        System.out.print("Enter Account Number: ");
        String accNum = scanner.nextLine();
        Account acc = bank.findAccount(accNum);
        if (acc != null) {
            System.out.print("Enter amount to deposit: ");
            double amount = scanner.nextDouble();
            scanner.nextLine();
            Transaction.deposit(acc, amount);
        } else {
            System.out.println("Account not found.");
        }
    }

    private void withdrawMoney() {
        System.out.print("Enter Account Number: ");
        String accNum = scanner.nextLine();
        Account acc = bank.findAccount(accNum);
        if (acc != null) {
            System.out.print("Enter amount to withdraw: ");
            double amount = scanner.nextDouble();
            scanner.nextLine();
            Transaction.withdraw(acc, amount);
        } else {
            System.out.println("Account not found.");
        }
    }
}
```

- Main

```
/**
 * Main application class to launch the Bank Management System.
 */
public class BankManagementApp {
        public static void main(String[] args) {
            Bank bank = new Bank();
            UserInterface ui = new UserInterface(bank);
            ui.start();
        }
    }
```

-

Transaction Handling

- Input validation during deposit and withdrawal to prevent invalid operations.
- Looping structure allows continuous operations until user opts to exit.
- Searching an account by account number before any transaction.
- Error handling for insufficient balance during withdrawals.

Conclusion and Team Contribution

The project successfully simulates key banking operations in a simple console-based Java application. It demonstrates usage of classes, objects, switch-case statements, and loops to create a modular and functional banking system.

Team Contributions:

- Riya (Nagar) - Account class design and implementation
- Vandana - Banking operations and transaction handling
- Vanshika - Menu design and input validation
- Yashika - Testing, debugging, and report compilation