جامعة الإمام عبد الرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY

Topic: Digital Bank

Computer Science

1443 هـ

The name:	*University ID:*
Fatimah ALharz	*2200004362*
Dema alghamdi	*2200001218*
Amal Alotaibib	*2200001746*
Khaowlah zakariya	*2200002105*

Table of content

Introduction:

Our Java program is linked to banking transactions and the ease and speed of dealing with the services provided by the bank such as withdrawal, deposit, loans and others. Users will create an account and choose the services they want and all services will be displayed on the screen.

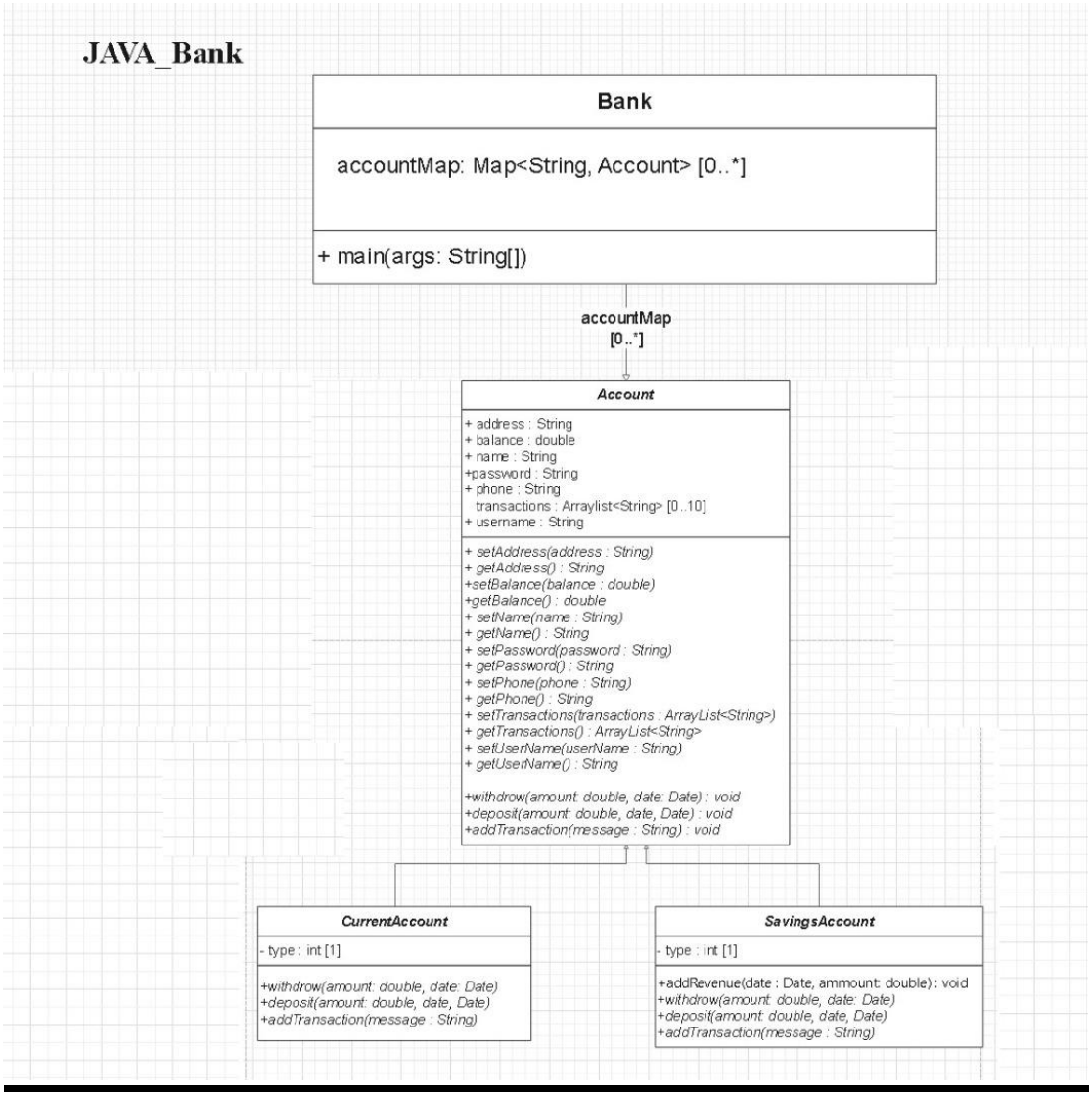
The problem statement:

The difficulty in conducting banking transactions and the obvious delay in making and completing electronic transfers. In addition to maintenance and update problems, they are no incentive to practice online service.

The target of the project:

The goal of creating a digital bank is to facilitate financial transactions for the user, in addition to other services provided by the bank and to address delays in electronic transactions. All transactions made by the user are displayed on the screen.

UML :


JAVA_Bank
Bank
accountMap: Map<String, Account> [0..*]
+ main(args: String[])
accountMap
[0..*]
Account
+ address : String
+ balance : double
+ name : String
+password : String
+ phone : String
transactions : Arraylist<String> [0..10]
+ username : String
+ setAddress(address : String)
+ getAddress() : String
+setBalance(balance : double)
+getBalance() : double
+ setName(name : String)
+ getName() : String
+ setPassword(password : String)
+ getPassword() : String
+ setPhone(phone : String)
+ getPhone() : String
+ setTransactions(transactions : ArrayList<String>)
+ getTransactions() : ArrayList<String>
+ setUserName(userName : String)
+ getUserName() : String
+withdrow(amount: double, date: Date) : void
+deposit(amount: double, date, Date) : void
+addTransaction(message : String) : void
CurrentAccount
- type : int [1]
+withdrow(amount: double, date: Date)
+deposit(amount: double, date, Date)
+addTransaction(message : String)
SavingsAccount
- type : int [1]
+addRevenue(date : Date, ammount: double) : void
+withdrow(amount: double, date: Date)
+deposit(amount: double, date, Date)
+addTransaction(message : String)

Full code:

Class Bank:

```
package com.company;

import java.util.*;


class Bank
{
    Map<String, Account> accountMap;
    Bank()
    {
        accountMap = new HashMap <String, Account>();
    }
    public static void main(String []args)
    {
        Scanner sc  =  new Scanner(System.in);
        Account account;
        String username,password;
        double amount;
        int accountType;
        Bank bank  =  new Bank();
        int choice;
        outer: while(true)
        {
            System.out.println("\n--------------------");
            System.out.println("     JAVA Bank     ");
            System.out.println("--------------------\n");
            System.out.println("1. Register account.");
            System.out.println("2. Login.");
            System.out.println("3. add Revenue to a saving account.");
            System.out.println("4. display user accounts");
            System.out.println("5. Exit.");
            System.out.print("\nEnter your choice : ");
            choice = sc.nextInt();
            sc.nextLine();
            switch(choice)
            {
                case 1:
                    System.out.print("choose account type : \n1 - current account\n2 - savings
account ");
                    accountType = sc.nextInt();
                    while(accountType < 1 || accountType > 2)
                    {
                        System.out.println("Invalid account type. Enter again :");
                        accountType = sc.nextInt() ;
                    }
                    sc.nextLine();
```

```
                System.out.print("Enter name : ");
                String name = sc.nextLine();
                System.out.print("Enter address : ");
                String address = sc.nextLine();
                System.out.print("Enter contact number : ");
                String phone = sc.nextLine();
                System.out.println("Set username : ");
                username = sc.next();
                while(bank.accountMap.containsKey(username))
                {
                   System.out.println("Username already exists. Set again : ");
                   username = sc.next();
                }
                System.out.println("Set a password (minimum 6 characters) :");
                password = sc.next();
                sc.nextLine();
                while(password.length() < 6)
                {
                   System.out.println("Invalid password, Enter again :");
                   password=sc.next();
                }
                System.out.print("Enter initial deposit : ");
                while(!sc.hasNextDouble())
                {
                   System.out.println("Invalid amount. Enter again :");
                   sc.nextLine();
                }
                amount=sc.nextDouble();
                if(accountType == 2){
                   while (amount < SavingsAccount.MIN_LIMIT){
                      System.out.println("Savings accounts have a 5000 SAR on the initial deposit
you can not create account with this amount, enter again :");
                      amount = sc.nextDouble();
                   }
                }
                sc.nextLine();
                if(accountType == 1){
                   account = new
CurrentAccount(username,password,name,address,phone,amount,new Date());
                   bank.accountMap.put(username, account);
                }
                else if(accountType == 2){
                   account = new
SavingsAccount(username,password,name,address,phone,amount,new Date());
                   bank.accountMap.put(username, account);
                }
                break;
             case 2:
                System.out.println("Enter username : ");
                username = sc.next();
                sc.nextLine();
                System.out.println("Enter password : ");
```

```
password = sc.next();
sc.nextLine();
if(bank.accountMap.containsKey(username))
{
    account = bank.accountMap.get(username);
    SavingsAccount savingsAccount;
    CurrentAccount currentAccount;
    if(account instanceof CurrentAccount){
        currentAccount = (CurrentAccount) account;
        if(account.password.equals(password))
        {

            while(true)
            {
                System.out.println("\n-------------------");
                System.out.println("W  E  L  C  O  M  E");
                System.out.println("-------------------\n");
                System.out.println("1. Deposit.");
                System.out.println("2. Transfer.");
                System.out.println("3. Last 5 transactions.");
                System.out.println("4. User information.");
                System.out.println("5. Log out.");
                System.out.print("\nEnter your choice : ");
                choice = sc.nextInt();
                sc.nextLine();
                switch(choice)
                {
                    case 1:
                        System.out.print("Enter amount : ");
                        while(!sc.hasNextDouble())
                        {
                            System.out.println("Invalid amount. Enter again :");
                            sc.nextLine();
                        }
                        amount = sc.nextDouble();
                        sc.nextLine();
                        currentAccount.deposit(amount,new Date());
                        break;
                    case 2:
                        System.out.print("Enter payee username : ");
                        username = sc.next();
                        sc.nextLine();
                        System.out.println("Enter amount : ");
                        while(!sc.hasNextDouble())
                        {
                            System.out.println("Invalid amount. Enter again :");
                            sc.nextLine();
                        }
                        amount = sc.nextDouble();
                        sc.nextLine();
                        if(amount > 300000)
                        {
```

```
                        System.out.println("Transfer limit exceeded. Contact bank
manager.");
                        break;
                    }
                    if(bank.accountMap.containsKey(username))
                    {
                        Account payee = bank.accountMap.get(username);
                        SavingsAccount savingsPayee;
                        CurrentAccount currentPayee;
                        if(payee instanceof SavingsAccount){
                            savingsPayee = (SavingsAccount) payee;
                            savingsPayee.deposit(amount,new Date());
                            currentAccount.withdraw(amount,new Date());
                        }else {
                            currentPayee = (CurrentAccount) payee;
                            currentPayee.deposit(amount,new Date());
                            currentAccount.withdraw(amount,new Date());
                        }

                    }
                    else
                    {
                        System.out.println("Username doesn't exist.");
                    }
                    break;
                case 3:
                    for(String transactions : account.transactions)
                    {
                        System.out.println(transactions);
                    }
                    break;
                case 4:
                    System.out.println("name : "+ account.name);
                    System.out.println("address : "+ account.address);
                    System.out.println("contact : "+ account.phone);
                    break;
                case 5:
                    continue outer;
                default:
                    System.out.println("Wrong choice !");
                }
            }
        }
        else
        {
            System.out.println("Wrong username/password.");
        }

    }else if(account instanceof SavingsAccount){
        savingsAccount = (SavingsAccount) account;
        if(account.password.equals(password))
        {
```

```
                while(true)
                {
                    System.out.println("\n**********************");
                    System.out.println("W  E  L  C  O  M  E");
                    System.out.println("\n**********************\n");
                    System.out.println("1. Deposit.");
                    System.out.println("2. Transfer.");
                    System.out.println("3. Last 5 transactions.");
                    System.out.println("4. User information.");
                    System.out.println("5. Log out.");
                    System.out.print("\nEnter your choice : ");
                    choice = sc.nextInt();
                    sc.nextLine();
                    switch(choice)
                    {
                        case 1:
                            System.out.print("Enter amount : ");
                            while(!sc.hasNextDouble())
                            {
                                System.out.println("Invalid amount. Enter again :");
                                sc.nextLine();
                            }
                            amount = sc.nextDouble();
                            sc.nextLine();
                            savingsAccount.deposit(amount,new Date());
                            break;
                        case 2:
                            System.out.print("Enter payee username : ");
                            username = sc.next();
                            sc.nextLine();
                            System.out.println("Enter amount : ");
                            while(!sc.hasNextDouble())
                            {
                                System.out.println("Invalid amount. Enter again :");
                                sc.nextLine();
                            }
                            amount = sc.nextDouble();
                            sc.nextLine();
                            if(amount > 300000)
                            {
                                System.out.println("Transfer limit exceeded. Contact bank
manager.");
                                break;
                            }
                            if(bank.accountMap.containsKey(username))
                            {
                                Account payee = bank.accountMap.get(username);
                                SavingsAccount savingsPayee;
                                CurrentAccount currentPayee;
                                if(payee instanceof SavingsAccount){
                                    savingsPayee = (SavingsAccount) payee;
```

```
                        savingsPayee.deposit(amount,new Date());
                        savingsAccount.withdraw(amount,new Date());
                    }else {
                        currentPayee = (CurrentAccount) payee;
                        currentPayee.deposit(amount,new Date());
                        savingsAccount.withdraw(amount,new Date());
                    }
                }
                else
                {
                    System.out.println("Username doesn't exist.");
                }
                break;
            case 3:
                for(String transactions : account.transactions)
                {
                    System.out.println(transactions);
                }
                break;
            case 4:
                System.out.println("name : "+ account.name);
                System.out.println("address : "+ account.address);
                System.out.println("contact : "+ account.phone);
                break;
            case 5:
                continue outer;
            default:
                System.out.println("Wrong choice !");
        }
      }
    }
    else
    {
        System.out.println("Wrong username/password.");
    }
  }


 }
 else
 {
    System.out.println("Wrong username/password.");
 }
 break;
case 3:
 System.out.println("Enter username : ");
 username = sc.next();
 sc.nextLine();
 double ammount = 0.00;
 if(bank.accountMap.containsKey(username))
 {
    Account user = bank.accountMap.get(username);
```

```
                    if(user instanceof SavingsAccount){
                        SavingsAccount savingsAccount = (SavingsAccount) user;
                        System.out.println("Enter amount : ");
                        while(!sc.hasNextDouble())
                        {
                            System.out.println("Invalid amount. Enter again :");
                            sc.nextLine();
                        }
                        ammount = sc.nextDouble();
                        savingsAccount.addRevenue(new Date(), ammount);
                    }else {
                        System.out.println("this is not a savings account, please choose savings
account first");
                    }

                }
                else
                {
                    System.out.println("Username doesn't exist.");
                }
                break;
            case 4:
                Set<String> keys = bank.accountMap.keySet();
                final String SEPARATOR = "    |    ";
                System.out.println("user name" + SEPARATOR + "address" + SEPARATOR +
"phone number" + SEPARATOR + "balance");
                for(String userName : keys){
                    Account details = bank.accountMap.get(userName);
                    System.out.println(details.name + SEPARATOR + details.address + SEPARATOR
+ details.phone + SEPARATOR + details.balance);

                }
                break;
            case 5:
                System.out.println("\nThank you for choosing JAVA Bank.");
                System.exit(1);
                break;
            default:
                System.out.println("Wrong choice !");
        }
    }
  }
}
```

Class Account:

```
package com.company;

import java.util.ArrayList;
import java.util.Date;

abstract class Account
{
    public String username,password,name,address,phone;
    public double balance;
    ArrayList<String> transactions;
    Account(String username, String password, String name, String address, String phone,
double balance, Date date)
    {
        this.username = username;
        this.password = password;
        this.name = name;
        this.address = address;
        this.phone = phone;
        this.balance = balance;
        transactions  =  new ArrayList<String>(5);
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAddress() {
        return address;
    }
```

```
    public void setAddress(String address) {
        this.address = address;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public double getBalance() {
        return balance;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }


    public abstract void addTransaction(String message);
    public abstract void withdraw(double amount,Date date);
    public abstract void deposit(double amount,Date date);
}
```

Class CurrentAccount:

```
package com.company;

import java.text.NumberFormat;
import java.util.Date;
import java.util.Locale;

class CurrentAccount extends Account {

    public static final int MIN_LIMIT = 0;
    private static final int TRANSACTIONS_LIMIT = 5;
    private static final int WITHDRAW_COST = 0;
    private final int type = 1;

    CurrentAccount(String username, String password, String name, String address, String
phone, double balance, Date date) {
        super(username, password, name, address, phone, balance, date);
        addTransaction(String.format("Initial deposit for current account - " +
NumberFormat.getCurrencyInstance(Locale.forLanguageTag("ar-SA")).format(balance)+" as on
" + "%1$tD"+" at "+"%1$tT.",date));

    }

    @Override
    public void deposit(double amount,Date date)
    {
        balance += amount;
        addTransaction(String.format(NumberFormat.getCurrencyInstance().format(amount)+"
credited to your account. Balance - "
+NumberFormat.getCurrencyInstance().format(balance)+" as on " + "%1$tD"+" at
"+"%1$tT.",date));
    }

    @Override
    public void withdraw(double amount,Date date)
    {
        if(amount>(balance-WITHDRAW_COST))
        {
            System.out.println("Insufficient balance.");
            return;
        }
        balance -= amount;
        addTransaction(String.format(NumberFormat.getCurrencyInstance().format(amount)+"
debited from your account. Balance - "
+NumberFormat.getCurrencyInstance().format(balance)+" as on " + "%1$tD"+" at
"+"%1$tT.",date));
    }

    @Override
```

```
    public void addTransaction(String message)
    {
        transactions.add(0,message);
        if(transactions.size()>TRANSACTIONS_LIMIT)
        {
            transactions.remove(TRANSACTIONS_LIMIT);
            transactions.trimToSize();
        }
    }

    public int getType() {
        return type;
    }


}
```

Class SavaingsAccount:

```
package com.company;

import java.text.NumberFormat;
import java.util.Date;

class SavingsAccount extends Account {

    public static final int MIN_LIMIT = 5000;
    private static final int TRANSACTIONS_LIMIT = 10;
    private static final int WITHDRAW_COST = 200;
    private final int type = 2;

    SavingsAccount(String username, String password, String name, String address, String
phone, double balance, Date date) {
        super(username, password, name, address, phone, balance, date);
        addTransaction(String.format("Initial deposit for savings account - " +
NumberFormat.getCurrencyInstance().format(balance)+" as on " + "%1$tD"+" at
"+"%1$tT.",date));
    }

    //only for savings account
    void addRevenue(Date date, double ammount)
    {
        deposit(ammount, date);
    }

    @Override
    public void deposit(double amount,Date date)
    {
        balance += amount;
        addTransaction(String.format(NumberFormat.getCurrencyInstance().format(amount)+"
credited to your account. Balance - "
```

```
+NumberFormat.getCurrencyInstance().format(balance)+" as on " + "%1$tD"+" at
"+"%1$tT.",date));
    }

    @Override
    public void withdraw(double amount,Date date)
    {
        if(amount>(balance-WITHDRAW_COST))
        {
            System.out.println("Insufficient balance.");
            return;
        }
        balance -= amount;
        balance -= WITHDRAW_COST;
        addTransaction(String.format(NumberFormat.getCurrencyInstance().format(amount)+"
debited from your account. Balance - "
+NumberFormat.getCurrencyInstance().format(balance)+" as on " + "%1$tD"+" at
"+"%1$tT.",date));
    }

    @Override
    public void addTransaction(String message)
    {
        transactions.add(0,message);
        if(transactions.size()>TRANSACTIONS_LIMIT)
        {
            transactions.remove(TRANSACTIONS_LIMIT);
            transactions.trimToSize();
        }
    }

    public int getType() {
        return type;
    }
}
```

Conclusion:

Finally, we designed a digital bank that facilitates all banking transactions such as withdrawals, deposits and other banking transactions and all transactions are displayed on the screen.