Introduction and Project description:

Now a days mobile banking is so popular which saves lot of time for both user and bank employees and most of banking needs are being done in mobile itself. With this project one can create an account if they are not existing customer. They can retrieve their account balance from the database and can add balance to their bank account and withdraw amount or transfer it to another account. There are different types of Loans providing by the bank and they can access details like rate of interest, monthly amount, total interest one is going to pay and tenure. Customers can check their eligibility for getting a credit card (we do that by analyzing credit score which will be included in the persons database) if they have required credit score, we inform him about his eligibility. They can create account by providing first name, last name and age and system creates account number and provides a secure passcode and a unique account number which can be used to login to the account and access other functionality.

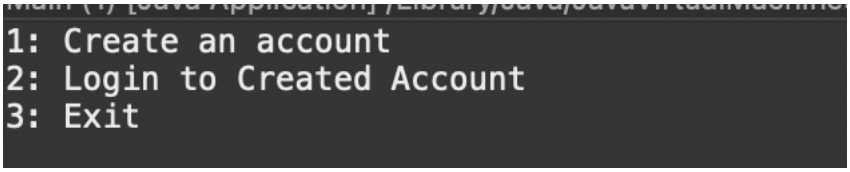
Feature of the code:

- Create Account
- Login to account
- Balance
- Withdraw
- Deposit
- Transfer
- Interest calculator
- Emi calculator
- Credit card eligibility checker

Running Instructions:

1.When we run the code first, we get is the main menu which contains three options, they are

1. Create an account
2. Login into created account
3. Exit



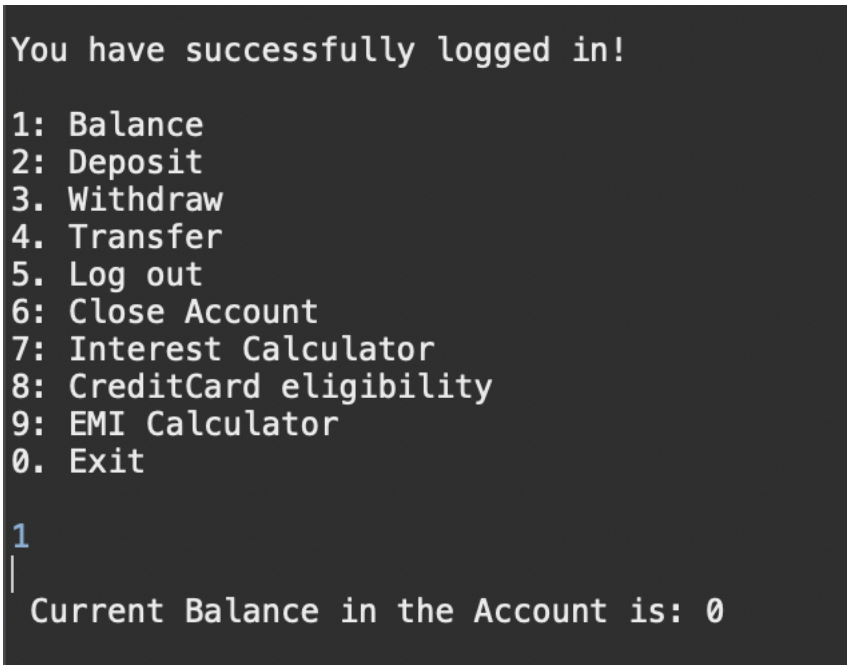
2.When we select create account code asks to input First name, last name, and age and then it created account in the database and gives us the account number and passcode to log into the account to perform different functions.

```
1
Enter your first name:
Preetam
Enter your Last name:
Beeravelli
Enter your Age:
23
Your Account has been created
Your Account number is:
7030097120
Your Account SecurePasscode is:
8926
```

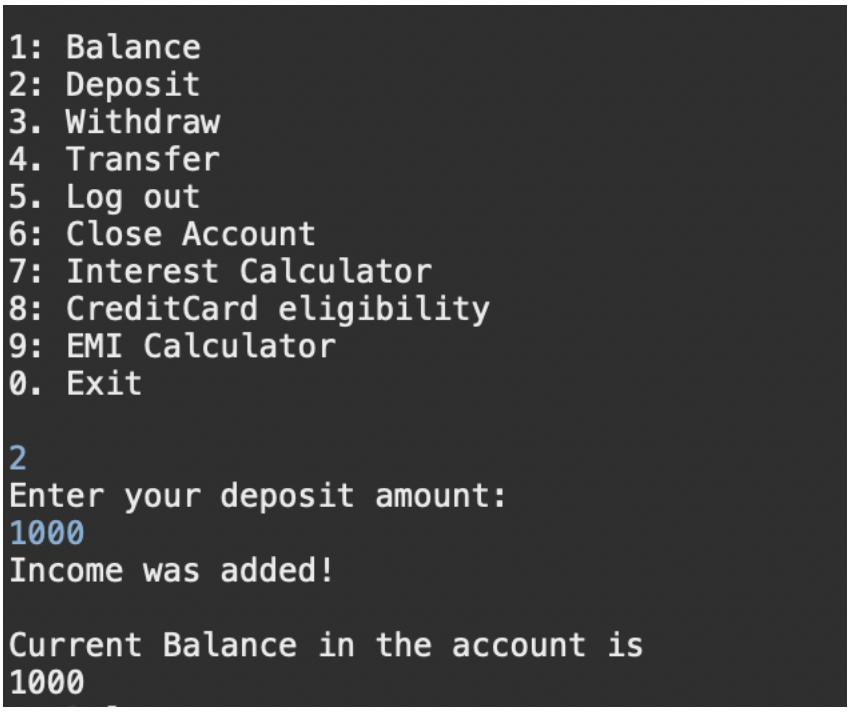
3.when we select option 2 from main menu it prompts to enter the account number and passcode to enter second menu, only if credentials are correct otherwise, we get an error and asks to input correct account number or passcode if typed wrong. If entered correctly it displays, you have successfully logged into the account.

```
1: Create an account
2: Login to Created Account
3: Exit

2

Enter your Account number:
7030097120
Enter your SecurePin:
8926

You have successfully logged in!

1: Balance
2: Deposit
3. Withdraw
4. Transfer
5. Log out
6: Close Account
7: Interest Calculator
8: CreditCard eligibility
9: EMI Calculator
0. Exit
```

4.When we press 1 for balance, we get the balance in the account, newly created accountd have 0 balance in then which displays 0.



5.To deposit balance in the account press 2 which prompts user to enter the amount they want to deposit and then updates the balance and displays the updated balance in that account.



6. To withdraw balance from the account we need to select option 3, it prompts to enter the amount they want to withdraw, if the entered amount is available in the account the balance will be withdrawn and remaining balance will be displayed.

```
3
Enter your Withdraw amount:
50
You have Succesfully Withdraw 50 from the Account

Current Balance in the account is 950
```

7.To transfer balance between two account we need at least two accounts created and after it we can transfer the amount by entering the account number to which we want to transfer and next prompts to enter the amount needed to be transferred. If transfer is successful, it displays the confirmation.

```
4
Transfer
Enter Account number:

7030097120
Please enter the amount you want to transfer:
500
Success!
```

8.if we want to logout of the account, we need to select option 5 which logs out of the account and displays main menu.

```
5

You have successfully logged out of your account!

1: Create an account
2: Login to Created Account
3: Exit
```

9.we can even delete the account by selecting close account option.

```
The account has been closed!
```

10. To get details about interest we need to select option 7, first it prompts to enter the amount user wants to invest and prompts to enter the number of years they are going to invest and displays the results with regards to each year:

```
7
Enter The amount you want to deposit
1000
The interest rate will be 0.06%
Enter number of years you want to invest
10
year   Amount on deposit

 1            1,060.00
 2            1,123.60
 3            1,191.02
 4            1,262.48
 5            1,338.23
 6            1,418.52
 7            1,503.63
 8            1,593.85
 9            1,689.48
10            1,790.85
```

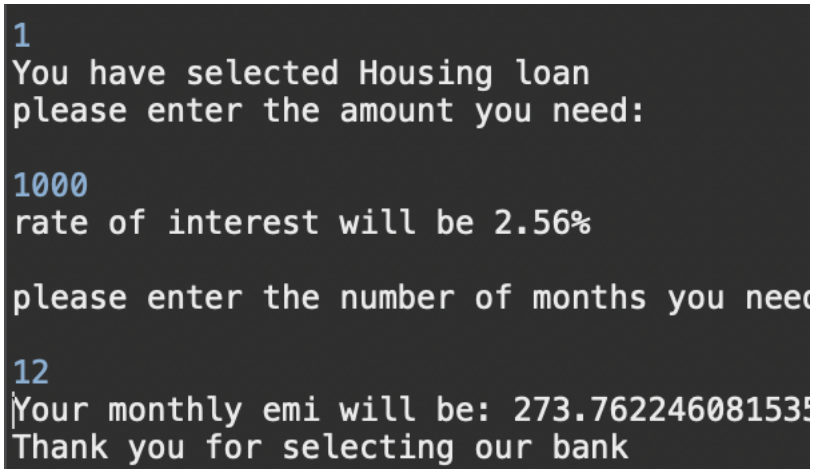
11. we can even know the credit card eligibility; system prompts to enter the credit score and displays results based on the credit score and displays credit limit also

```
8
Please Enter Your credit Score:
Enter values in the range of 0-1000
560
YOU ARE ELIGIBLE FOR CREDIT CARD
Your credit limit will be 5000$
```

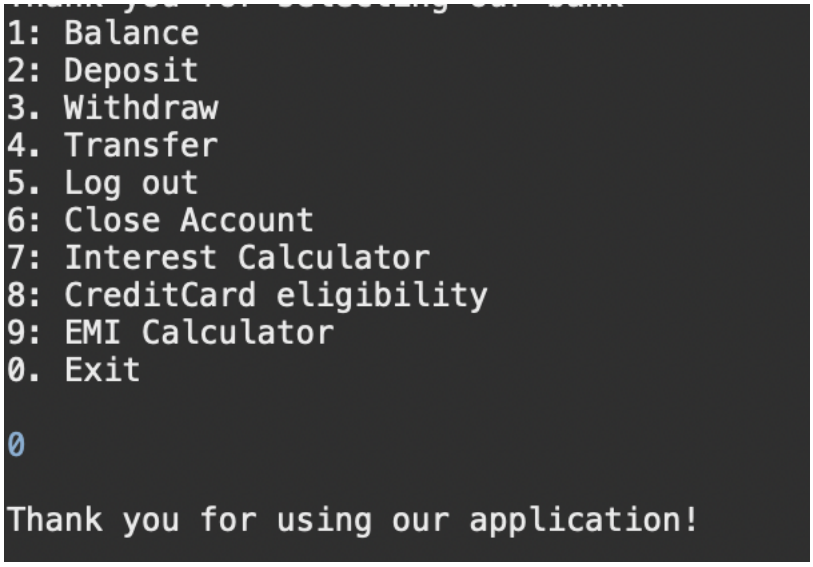
12. we can even know the EMI for select amount by selecting option 9:

```
9
Please select the type of load you need
1: Housing Loan
2: Auto Loan
3: personal Loan
4: Education Loan
```

Different types of loans have different rate and system prompts user to enter the amount they need and number of months they need the amount and displays the monthly interest amount.



13. Last option will be exit when selected will exit the system and closes the application.



Code Structure:

Our application has a total of 7 classes, They are:

1. CreateAccount
2. DbConnection
3. CreditCard
4. Interest
5. Loan
6. Mainmenu
7. Main

1. MainMenu Class:

Contains two methods to display menu options in the program
First is displayed when we we run the program
Second menu is displayed when user logins to the account

```java
public class Mainmenu {

    public static void mainPage() {

        System.out.println(
                "1: Create an account\n" +
                "2: Login to Created Account\n" +
                "3: Exit\n");
    }

    public static void secondPage() {
        System.out.println(
                "1: Balance\n"
              + "2: Deposit\n"
              + "3. Withdraw\n"
              + "4. Transfer\n"
              + "5. Log out\n"
              + "6: Close Account\n"
              + "7: Interest Calculator\n"
              + "8: CreditCard eligibility\n"
              + "9: EMI Calculator\n"
              + "0. Exit\n"

                );
    }

}
```

1.**CreateAccount class:**
Various static variables are created in this class.

```java
public class CreateAccount {

    private String accountNumber;
    private String password;
    public int balance;
    static Scanner input = new Scanner(System.in);
    private static final Random rand = new Random();
    static DbConnection dbConnection = new DbConnection();
    static CreateAccount myAccount= new CreateAccount();
```

accCre method:

Takes input from the user like First name, Last name and age using input object of Scanner class and inserts into the database using dbConnection object of DbConnection Class.

```java
    public static void accCre() {

    System.out.println("Enter your first name:");
    String firstName = input.next();

    System.out.println("Enter your Last name:");
    String lastName = input.next();

    System.out.println("Enter your Age:");
    int age = input.nextInt();

    dbConnection.insertdetails(firstName, lastName, age);
    myAccount.setAccountNumber();

    System.out.println(
            "Your Account has been created\n" +
            "Your Account number is:");
    String account = myAccount.getAccountNumber();

    System.out.println(account);

    System.out.println("Your Account SecurePasscode is:");

    myAccount.setSecurePasscode();

    String pin = myAccount.getSecurePasscode();

    System.out.println(pin + "\n");

    int balance = myAccount.getBalance();

    dbConnection.insert(account, pin, balance);
}
```

Creacc method:

creAcc method generates random account number using arrays of length 9 digits.

```
public static String createAcc() {

    String BIN = "703";
    int controlNumber = 0;
    int checkSum = 0;

    String card = BIN.concat(String.format("%06d", rand.nextInt(10000)));

    String[] cardArr = card.split("");

    // Converting the String to an array of Integers

    int[] cardNumber = new int[9];

    for (int i = 0; i < 9; i++) {

        cardNumber[i] = Integer.parseInt(cardArr[i]);

    }
```

There are set and get methods to set account number, passcode, and balance

```
public void setAccountNumber() {

        this.accountNumber = createAcc();
    }


public String getAccountNumber() {

        return accountNumber;
    }

    public void setSecurePasscode() {

        this.password = String.format("%04d", rand.nextInt(8999) +
    }

    public String getSecurePasscode() {

        return password;
    }


    public int getBalance() {

        return balance;
    }
```

3. CreditCard Class:

This class contains cardApply method which takes credit score input from user and used switch statement to evaluate credit score and provide decision and credit limit to the user.

```
public static void cardApply() throws SQLException {

    Scanner input = new Scanner(System.in);

    System.out.println("Please Enter Your credit Score:");
    System.out.println("Enter values in the range of 0-1000");

    int creditScore = input.nextInt();

    switch(creditScore  / 100) {

    case 1:
        System.out.println("YOU ARE NOT ELIGIBLE FOR CREDIT CARD");
        break;
    case 2:
        System.out.println("YOU ARE NOT ELIGIBLE FOR CREDIT CARD");
        break;
    case 3:
        System.out.println("YOU ARE NOT ELIGIBLE FOR CREDIT CARD");
        break;
    case 4:
        System.out.println("YOU ARE NOT ELIGIBLE FOR CREDIT CARD");
        break;
    case 5:
        System.out.println("YOU ARE ELIGIBLE FOR CREDIT CARD");
        System.out.println("Your credit limit will be 5000$");
        break;
    case 6:
        System.out.println("YOU ARE ELIGIBLE FOR CREDIT CARD");
        System.out.println("Your credit limit will be 10000$");
        break;
    case 7:
        System.out.println("YOU ARE ELIGIBLE FOR CREDIT CARD");
        System.out.println("Your credit limit will be 15000$");
        break;
    case 8:
        System.out.println("YOU ARE ELIGIBLE FOR CREDIT CARD");
        System.out.println("Your credit limit will be 25000$");
        break;
    case 9:
        System.out.println("YOU ARE ELIGIBLE FOR CREDIT CARD");
        System.out.println("Your credit limit will be 5000$");
        break;
    case 10:
            System.out.println("YOU ARE ELIGIBLE FOR CREDIT CARD");
            System.out.println("Your credit limit will be 5000$");
            break;
            default:
                System.out.println("You have entered wrong creditScore,"
                        + " please try again with coreect range ");
    }
    }
```

4.DbConnection class:

Checks for connection with database and throws an error if there is no connection

```java
package BankProject;

import java.sql.*;

public class DbConnection {

    private  Connection connect() {

            String url = "jdbc:mysql://localhost:3306/BankData";
            String user = "bank";
            String password = "Securepassword";



        Connection connection = null;

        try {

            connection = DriverManager.getConnection(url,user,password);

        } catch (SQLException e) {

            System.out.println(e.getMessage());

        }

        return connection;
    }

```

Insert method for inserting data into database i.e.. accountDetails Table.

```java
public void insert(String number, String pin, Integer balance) {

        String sql = "INSERT INTO accountDetails(number, pin, balance) VALUES(?, ?, ?)";

        try (Connection conn = this.connect();
             PreparedStatement preparedStatement = conn.prepareStatement(sql)) {

            preparedStatement.setString(1, number);
            preparedStatement.setString(2, pin);
            preparedStatement.setInt(3, balance);

            preparedStatement.executeUpdate();

        } catch (SQLException e) {

            System.out.println(e.getMessage());
        }
    }
```

Update method for updating data into database i.e.. accountDetails Table.

```
    public void update(String number, String pin, Integer balance) {

        String sql = "UPDATE accountDetails SET balance = ? WHERE number = ? AND pin = ?";

        try (Connection conn = this.connect();
             PreparedStatement preparedStatement = conn.prepareStatement(sql)) {

            preparedStatement.setInt(1, balance);
            preparedStatement.setString(2, number);
            preparedStatement.setString(3, pin);

            preparedStatement.executeUpdate();

        } catch (SQLException e) {

            System.out.println(e.getMessage());
        }
    }
```

CheckLogin method checks data from database and compares it with details entered by user using conditional and logins if details match it grants access.

```
    public boolean checkLogin(String userAccount, String userPin) {

        // This method is used to verify if the credentials exist in the database

        String savedAccount, savedPass;

        String query = "SELECT * FROM accountDetails";

        try (Connection conn = this.connect();

             Statement statement = conn.createStatement();
             ResultSet resultSet = statement.executeQuery(query)) {

            while (resultSet.next()) {

                savedAccount = resultSet.getString("number");

                savedPass = resultSet.getString("pin");

                if (userAccount.equals(savedAccount) && userPin.equals(savedPass)) {

                    return true;

                }

            }


        } catch (SQLException exception) {

            exception.printStackTrace();
        }

        return false;
    }
```

checkAccount method checks if Account is present in the database.

```
    public boolean checkAccount(String cardTransfer) {

        // This method is used to check if an Account exists in the database

        String savedCard;

        String query = "SELECT * FROM accountDetails";

        try (Connection conn = this.connect();

             Statement statement = conn.createStatement();
             ResultSet resultSet = statement.executeQuery(query)) {

            while (resultSet.next()) {

                savedCard = resultSet.getString("number");

                if (cardTransfer.equals(savedCard)) {

                    return true;

                }

            }

        } catch (SQLException exception) {

            exception.printStackTrace();
        }

        return false;
```

5.Interest class:

This class contains method getInterestDetails method which displays interest you will be getting at the end of each year for principal amount entered and number of years entered.

```
    public static void getInterestDetails() {

    Scanner input = new Scanner(System.in);

    double principal;

        System.out.println("Enter The amount you want to deposit");
        principal = input.nextDouble();

        System.out.println("The interest rate will be 0.06%");
        System.out.println("Enter number of years you want to invest");
        int tenure = input.nextInt();

        System.out.printf("%s%20s%n","year","Amount on deposit");
        System.out.println();

        for(int year =1 ; year <= tenure ; ++year) {
             double amount = principal * Math.pow(1.00 + 0.06, year);
             System.out.printf("%2d%,20.2f%n",year,amount);
        }

    }
```

6. **Loan Class:**

This class contains emiCal method to calculate EMI for different types of loans for entered loan amount and number of months term.
Uses switch statement to select between different types of loans.

```
public static void emiCal() {


    System.out.println("Please select the type of load you need");
    System.out.println("1: Housing Loan \n" +
    "2: Auto Loan \n" + "3: personal Loan \n" + "4: Education Loan \n");
    Scanner input = new Scanner(System.in);

int choice = input.nextInt();

    switch(choice) {

    case 1:
        System.out.println("You have selected Housing loan");
        System.out.println("please enter the amount you need: \n");

        double amount = input.nextDouble();
        System.out.println("rate of interest will be 2.56% \n");

        System.out.println("please enter the number of"
                + " months you need loan: \n");
        int month = input.nextInt();

        double emi1 = Math.pow(1 + 0.256, month);
        double emi2 = emi1 - 1;
        double emi3 = emi1/emi2;
        double emi = emi3 * amount * 0.256;

        System.out.println("Your monthly emi will be: " + emi);
        System.out.println("Thank you for selecting our bank");
        break;


    case 2:
        System.out.println("You have selected Auto loan");
        System.out.println("please enter the amount you need: \n");

        double amount1 = input.nextDouble();
        System.out.println("rate of interest will be 3.56% \n");

        System.out.println("please enter the number of "
                + "months you need loan: \n");
        int month1 = input.nextInt();

        double emi11 = Math.pow(1 + 0.356, month1);
        double emi22 = emi11 - 1;
        double emi33 = emi11/emi22;
        double emii = emi33 * amount1 * 0.356;

        System.out.println("Your monthly emi will be: " + emii);
        System.out.println("Thank you for selecting our bank");
        break;
```

```
        case 3:
            System.out.println("You have selected Personal loan");
            System.out.println("please enter the amount you need: \n");

            double amount2 = input.nextDouble();
            System.out.println("rate of interest will be 4.8% \n");

            System.out.println("please enter the number of "
                    + "months you need loan: \n");
            int month2 = input.nextInt();

            double emi111 = Math.pow(1 + 0.48, month2);
            double emi222 = emi111 - 1;
            double emi333 = emi111/emi222;
            double emiii = emi333 * amount2 * 0.48;

            System.out.println("Your monthly emi will be: " + emiii);
            System.out.println("Thank you for selecting our bank");
            break;

        case 4:
            System.out.println("You have Education loan");
            System.out.println("please enter the amount you need: \n");

            double amount3 = input.nextDouble();
            System.out.println("rate of interest will be 1.3% \n");

            System.out.println("please enter the number of"
                    + " months you need loan: \n");
            int month3 = input.nextInt();

            double emi1111 = Math.pow(1 + 0.13, month3);
            double emi2222 = emi1111 - 1;
            double emi3333 = emi1111/emi2222;
            double emiiii = emi3333 * amount3 * 0.13;

            System.out.println("Your monthly emi will be: " + emiiii);
            System.out.println("Thank you for selecting our bank");
            break;
            default:
                System.out.println("Please select correct option");
        }
```

7. Main class

Contains main method to execute the program and uses switch statement and nested switch statements to switch between different options of the class.

Outer Switch for main menu (first menu):

Case 1 calls accCre method of CreateAccount class.
Case 2 enter second switch statement if login is successful
Case 0 exists the system

```
case 1:
    CreateAccount.accCre();
    int balance = myAccount.getBalance();
    break;

case 2:

    System.out.println("\nEnter your Account number:");

    String userAccount = input.next();

    System.out.println("Enter your SecurePin:");

    String userPin = input.next();


case 0:

    System.out.println("\nThank you for using our application \n GoodBye!");

    input.close();

    System.exit(0);
```

Inner Switch menu after logging in (Second menu):
Case 1: displays balance
Case 2: deposits amount
Case 3: withdraws amount

```
    case 1:// Gets Balance of the created Account

        System.out.println("\n Current Balance in the Account is: " + balance + "\n" );

        break;

    case 2: // Deposit amount


        System.out.println("Enter your deposit amount:");

        balance += input.nextInt();

        dbConnection.update(userAccount, userPin, balance);

        System.out.println("Income was added!\n");
        System.out.println("Current Balance in the account is \n" + balance );



        break;

    case 3: // Withdraw Amount


        System.out.println("Enter your Withdraw amount:");
        int withdrawAmount = input.nextInt();

        balance -= withdrawAmount;

        dbConnection.update(userAccount, userPin, balance);

        System.out.println("You have Succesfully Withdraw " + withdrawAmount+ " from the Account\n");
        System.out.println("Current Balance in the account is " + balance + "\n" );


        break;
```

Case 7: gets details about interest
Case 8: displays credit card eligibility

```
        case 7:// details about interest

            Interest.getInterestDetails();
            break;

        case 8:// displays credit card eligibility

        try {
            CreditCard.cardApply();
        } catch (SQLException e) {

            e.printStackTrace();
        }
            break;
```

Case:4 transfers balance between two accounts

```
case 4:

    while (true) {

        System.out.println("Transfer\n" +
                "Enter Account number:");

        String accNum = input.next();

        if (!CreateAccount.verifyAcc(accNum)) {

            System.out.println("There is no Account with this number." +
                    " Please check the account number and try again!\n");
            break;
        }

        if (!dbConnection.checkAccount(accNum) && CreateAccount.verifyAcc(accNum)) {

            System.out.println("No Account does exist with that number.\n");

            break; }

        if (accNum.equals(userAccount)) {

            System.out.println("You can't transfer money to the same account!\n");

            break;}

        if (CreateAccount.verifyAcc(accNum) &&
                !(!dbConnection.checkAccount(accNum) && CreateAccount.verifyAcc(accNum))) {

            System.out.println("Please enter the amount you want to transfer:");

            int amountTransfer = input.nextInt();

            if (balance < amountTransfer) {

                System.out.println("Not enough Balance in your account!\n");

                break;}

            // Money to be transferred

            int targetBalance = dbConnection.returnBalance(accNum);

            dbConnection.update(accNum, amountTransfer + targetBalance);

            // Subtract amount from the actual account

            dbConnection.update(userAccount, balance - amountTransfer);

            System.out.println("Success!\n");

            break;} }
            break;
```

Case 9: calls emiCal method

```
case 9: //calls emi method
    Loan.emiCal();
    break;
```

Case0: exists system

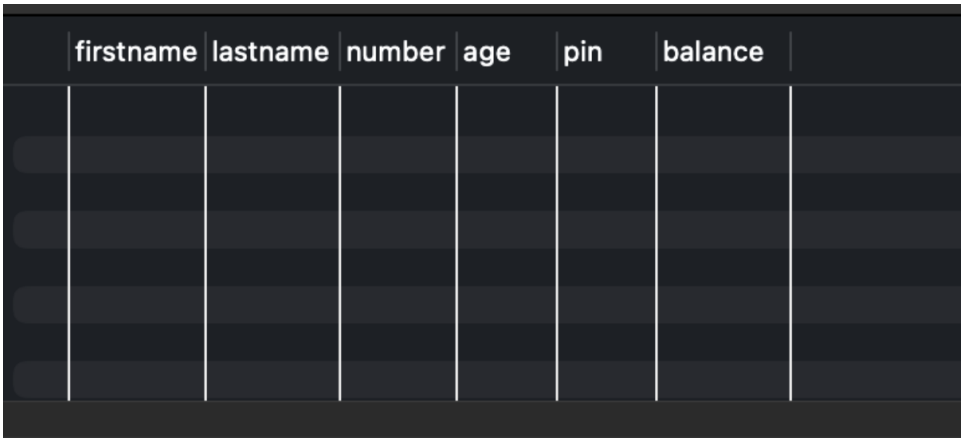
```
case 0:
    System.out.println("\nThank you for using our application!");
    input.close();
    System.exit(0);
```

Steps in developing code:

1. First we have created table in my SQL using the following command

```
CREATE TABLE accountDetails (
    firstname varchar(20) Default NULL,
    lastname  varchar(20) Default NULL,
    number bigint Default NULL,
    age smallint DEFAULT NULL,
    pin smallint DEFAULT NULL,
    balance smallint DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

2. Then the table is accountDetails table is created:



3. Then we have connected the database using the following commands

```
private  Connection connect() {

        String url = "jdbc:mysql://localhost:3306/BankData";
        String user = "bank";
        String password = "Securepassword";



    Connection connection = null;

    try {

        connection = DriverManager.getConnection(url,user,password);

    } catch (SQLException e) {

        System.out.println(e.getMessage());

    }

    return connection;
}
```

4. Then we have created classes like DbConnection, Mainmenu, CreditCard, Loan, Interest classes respectively.
5. We have connected program to mysql using jdbc driver.
6. Then by using all these classes as base we have created Main class which contains main methods to run the program.
7. Then we have checked for the errors.

Structures Used:

- If else statements
- Switch Statements
- While statements
- Try and catch blocks
- Nested switch statements

References:

- Java How to program early objects
- https://alvinalexander.com/java/java-mysql-update-query-example/
- https://www.mysqltutorial.org/mysql-generated-columns/
- https://www.youtube.com/watch?v=293M9-QRZ0c
- https://www.businesstoday.in/magazine/banking/story/how-to-calculate-emi-on-your-loans-formula-explanation-74916-2017-01-02#:~:text=The%20mathematical%20formula%20for%20calculating,the%20number%20of%20monthly%20instalments.
- https://alvinalexander.com/jdbc/
- https://www.tutorialspoint.com/generate-a-random-array-of-integers-in-java#:~:text=In%20order%20to%20generate%20random,this%20random%20number%20generator%20sequence.
- https://www.javatpoint.com/example-to-connect-to-the-mysql-database