CS4001NI Programming

30% Individual Coursework

2022-23 Autumn

Student Name: Krish Bhattarai

London Met ID: 22067570

College ID: NP01CP4A220071

Group: L1C3

Assignment Due Date: Wednesday, May 10, 2023

Assignment Submission Date: Monday, May 8, 2023

Contents

Table of Figures

Table of Tables

1. Introduction

Java is a popular object-oriented programming language that runs on almost any device. Java is easy to learn for developers because the syntax is similar to C and C++.

Java permits developers to reuse the same code again and again, simlifing the development and maintenance of a program. Java is platform independent which means that the code can be moved to different devices which saves time and effort. Codes in Java needs to be compiled before can can be run. (IBM, 2023).

1.1 Introduction to this project

This project aims to develop a graphical user interface (GUI) for a system that stores details of Bank Cards in an ArrayList. The previous part of the coursework involved implementing the core functionality of the system using Java classes, such as the BankCard class and the CardList class.

The program will allow the users to add, delete and edit the details in bank by interacting with the GUI through buttons, combo box and text fields. The class BankGUI contains a main method that will be tested using the command prompt.

2. Class Diagram

Class Diagram also known as structural diagram is like a flowchart. It is generally used for construction purposes. It provides a conceptual and architectural model of a system that is being developed. It is used for describing, visualizing different aspects of the program (tutorialspoint, 2023).

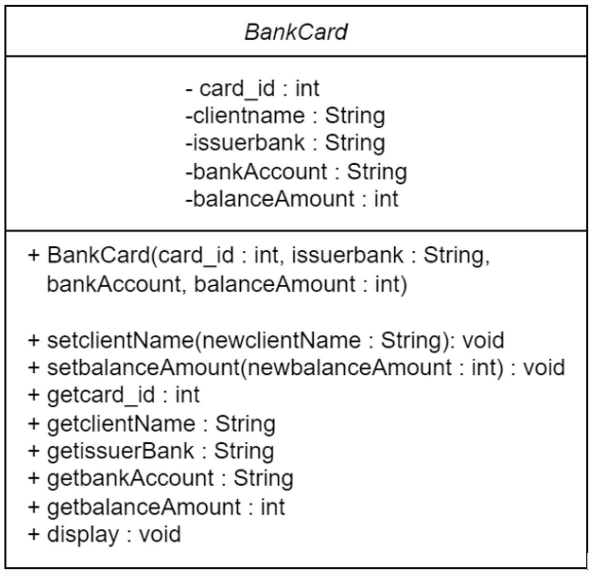


Figure 1: Class Diagram BankCard

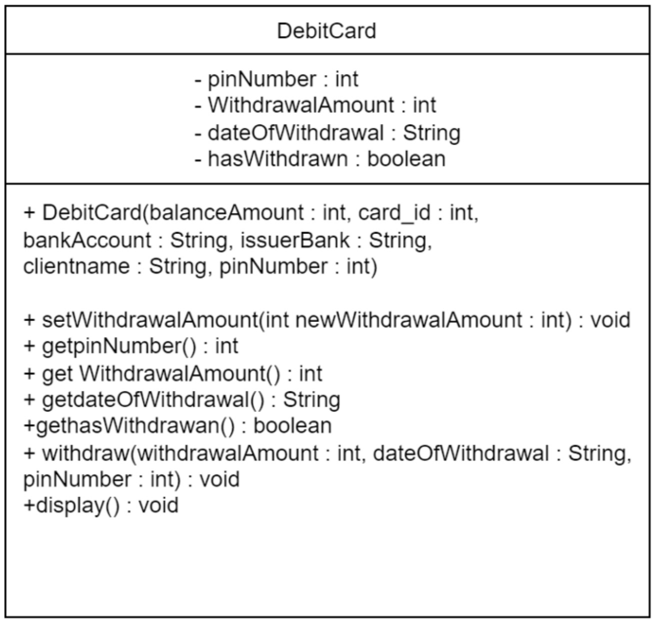

DebitCard
- pinNumber : int
- WithdrawalAmount : int
- dateOfWithdrawal : String
- hasWithdrawn : boolean
+ DebitCard(balanceAmount : int, card_id : int, bankAccount : String, issuerBank : String, clientname : String, pinNumber : int)
+ setWithdrawalAmount(int newWithdrawalAmount : int) : void
+ getpinNumber() : int
+ get WithdrawalAmount() : int
+ getdateOfWithdrawal() : String
+gethasWithdrawan() : boolean
+ withdraw(withdrawalAmount : int, dateOfWithdrawal : String, pinNumber : int) : void
+display() : void

Figure 2:: Class Diagram DebitCard

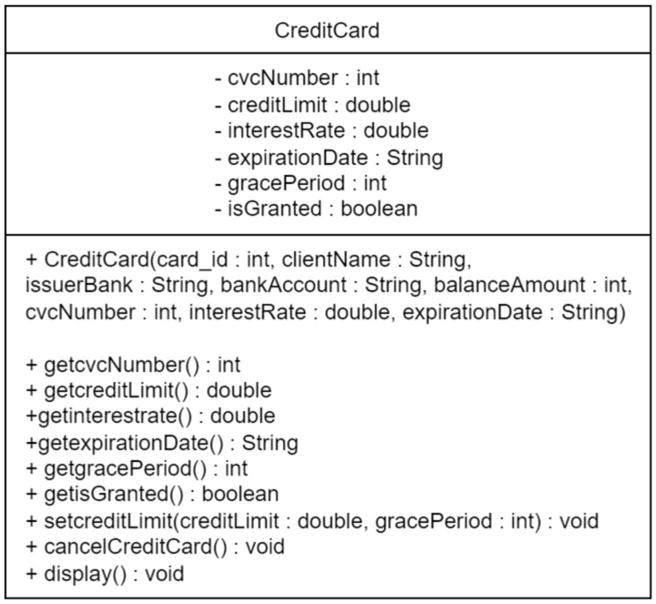

CreditCard
- cvcNumber : int
- creditLimit : double
- interestRate : double
- expirationDate : String
- gracePeriod : int
- isGranted : boolean
+ CreditCard(card_id : int, clientName : String, issuerBank : String, bankAccount : String, balanceAmount : int, cvcNumber : int, interestRate : double, expirationDate : String)
+ getcvcNumber() : int
+ getcreditLimit() : double
+getinterestrate() : double
+getexpirationDate() : String
+ getgracePeriod() : int
+ getisGranted() : boolean
+ setcreditLimit(creditLimit : double, gracePeriod : int) : void
+ cancelCreditCard() : void
+ display() : void

Figure 3:: Class Diagram CreditCard

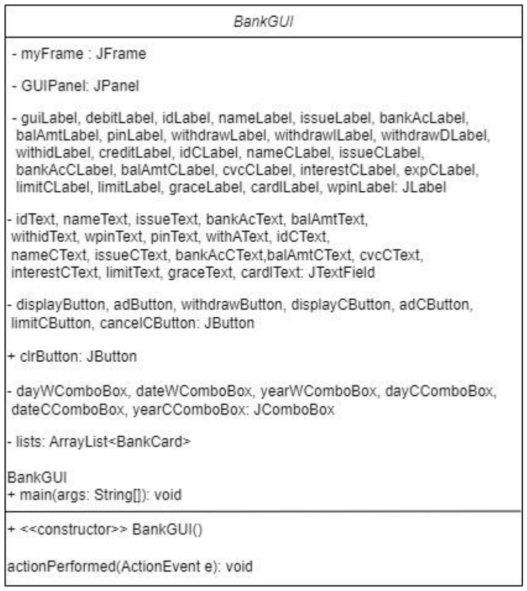

BankGUI
- myFrame : JFrame
- GUIPanel: JPanel
- guiLabel, debitLabel, idLabel, nameLabel, issueLabel, bankAcLabel, balAmtLabel, pinLabel, withdrawLabel, withdrawlLabel, withdrawDLabel, withidLabel, creditLabel, idCLabel, nameCLabel, issueCLabel, bankAcCLabel, balAmtCLabel, cvcCLabel, interestCLabel, expCLabel, limitCLabel, limitLabel, graceLabel, cardlLabel, wpinLabel: JLabel
- idText, nameText, issueText, bankAcText, balAmtText, withidText, wpinText, pinText, withAText, idCText, nameCText, issueCText, bankAcCText,balAmtCText, cvcCText, interestCText, limitText, graceText, cardlText: JTextField
- displayButton, adButton, withdrawButton, displayCButton, adCButton, limitCButton, cancelCButton: JButton
+ clrButton: JButton
- dayWComboBox, dateWComboBox, yearWComboBox, dayCComboBox, dateCComboBox, yearCComboBox: JComboBox
- lists: ArrayList<BankCard>
BankGUI
+ main(args: String[]): void
+ <<constructor>> BankGUI()
actionPerformed(ActionEvent e): void

Figure 4: Class Diagram BankGUI

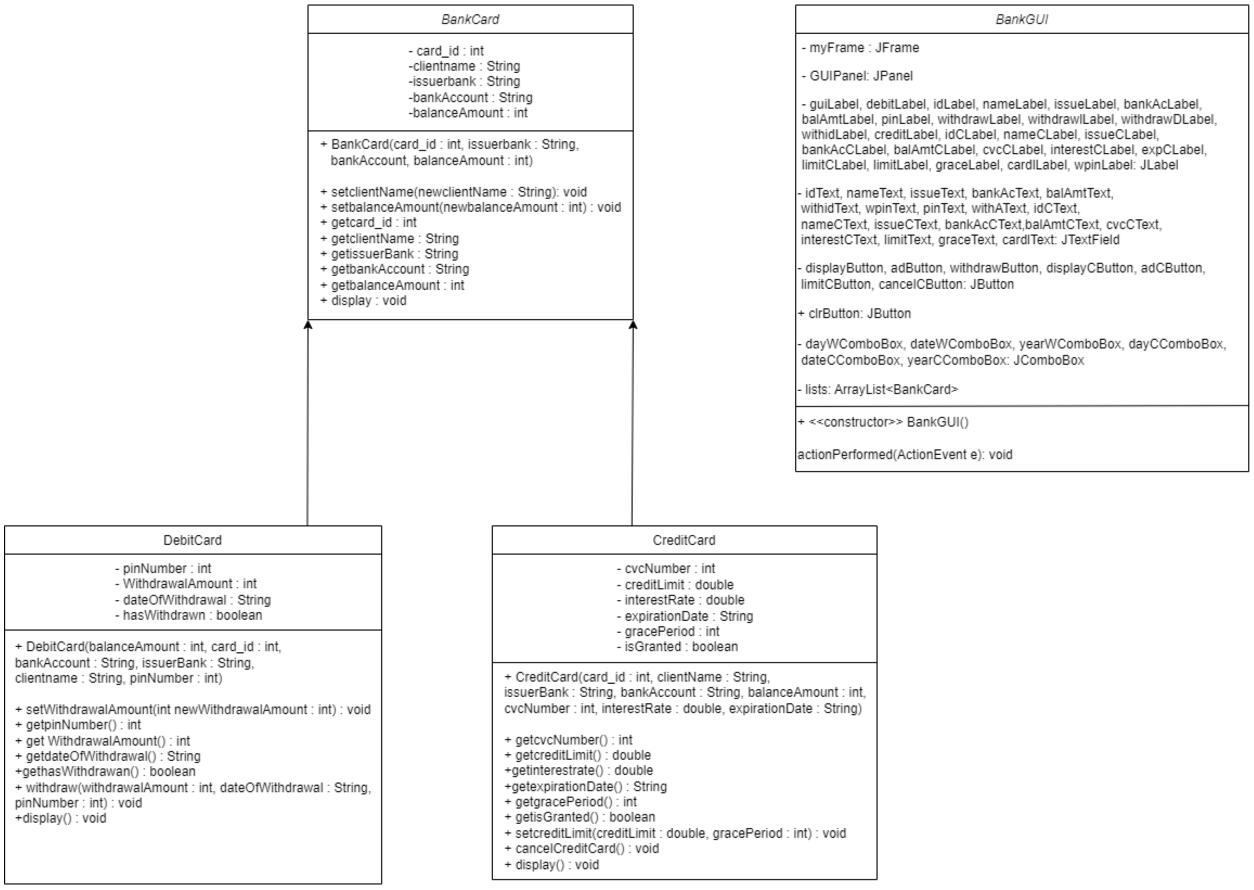

BankCard
- card_id : int
-clientname : String
-issuerbank : String
-bankAccount : String
-balanceAmount : int
+ BankCard(card_id : int, issuerbank : String, bankAccount, balanceAmount : int)
+ setclientName(newclientName : String): void
+ setbalanceAmount(newbalanceAmount : int) : void
+ getcard_id : int
+ getclientName : String
+ getissuerBank : String
+ getbankAccount : String
+ getbalanceAmount : int
+ display : void
BankGUI
- myFrame : JFrame
- GUIPanel: JPanel
- guiLabel, debitLabel, idLabel, nameLabel, issueLabel, bankAcLabel, balAmtLabel, pinLabel, withdrawLabel, withdrawlLabel, withdrawDLabel, withidLabel, creditLabel, idCLabel, nameCLabel, issueCLabel, bankAcCLabel, balAmtCLabel, cvcCLabel, interestCLabel, expCLabel, limitCLabel, limitLabel, graceLabel, cardlLabel, wpinLabel: JLabel
- idText, nameText, issueText, bankAcText, balAmtText, withidText, wpinText, pinText, withAText, idCText, nameCText, issueCText, bankAcCText,balAmtCText, cvcCText, interestCText, limitText, graceText, cardlText: JTextField
- displayButton, adButton, withdrawButton, displayCButton, adCButton, limitCButton, cancelCButton: JButton
+ clrButton: JButton
- dayWComboBox, dateWComboBox, yearWComboBox, dayCComboBox, dateCComboBox, yearCComboBox: JComboBox
- lists: ArrayList<BankCard>
+ <<constructor>> BankGUI()
actionPerformed(ActionEvent e): void
DebitCard
- pinNumber : int
- WithdrawalAmount : int
- dateOfWithdrawal : String
- hasWithdrawn : boolean
+ DebitCard(balanceAmount : int, card_id : int, bankAccount : String, issuerBank : String, clientname : String, pinNumber : int)
+ setWithdrawalAmount(int newWithdrawalAmount : int) : void
+ getpinNumber() : int
+ get WithdrawalAmount() : int
+ getdateOfWithdrawal() : String
+gethasWithdrawan() : boolean
+ withdraw(withdrawalAmount : int, dateOfWithdrawal : String, pinNumber : int) : void
+display() : void
CreditCard
- cvcNumber : int
- creditLimit : double
- interestRate : double
- expirationDate : String
- gracePeriod : int
- isGranted : boolean
+ CreditCard(card_id : int, clientName : String, issuerBank : String, bankAccount : String, balanceAmount : int, cvcNumber : int, interestRate : double, expirationDate : String)
+ getcvcNumber() : int
+ getcreditLimit() : double
+getinterestrate() : double
+getexpirationDate() : String
+ getgracePeriod() : int
+ getisGranted() : boolean
+ setcreditLimit(creditLimit : double, gracePeriod : int) : void
+ cancelCreditCard() : void
+ display() : void

Figure 5: Combined Class Diagram

3. Pseudocode

Pseudocode also known as fake code uses simple English language which makes it easier to understand for both programmers and non-programmers.

3.1 Pseudocode of BankGUI

```
IMPORT javax.swing.JFrame;

IMPORT java.swing.*;

IMPORT java.awt.event.*;

IMPORT java. util.ArrayList;

CREATE a class BankGUI which implements ActionListener

DO

        DECLARE instance variable myFrame as JFrame using private access modifier

        DECLARE instance variable lists as ArrayList<BankCard> using private access
        modifier.

DECLARE instance variables guiLabel, debitLabel, idLabel, nameLabel, issueLabel,
bankAcLabel, balAmtLabel, pinLabel, withdrawLabel, withdrawlLabel, withdrawDLabel,
withidLabel, creditLabel, idCLabel, nameCLabel, issueCLabel, bankAcCLabel,
balAmtCLabel, cvcCLabel, interestCLabel, expCLabel, limitCLabel, limitLabel,
graceLabel, cardlLabel, wpinLabel as JLabel using private access modifier.

DECLARE instance variable idText, nameText, issueText, bankAcText, balAmt,
withidText, wpinText, pinText, withAtext, dayComboBox, dateComboBox,
yearComboBox as JTextField using private access modifier.

DECLARE instance variable dayComboBox, dateComboBox, yearComboBox as
JComboBox using private access modifier.

DECLARE instance variable displayButton, adButton, withdrawButton as JButton using
private access modifier.
```

```
DECLARE instance variable idCText, nameCText, issueCText, bankAcCText, balAmtCText, cvcCText, interestCText, dayCComboBox, dateCComboBox, yearCComboBox, limitText, graceText, cardlText as JTextField using private access modifier.

DECLARE instance variable interestCText, dayCComboBox, dateCComboBox, yearCComboBox as JComboBox using private access modifier.

DECLARE instance variable displayCButton, adCButton, limitCButton, cancelCButton as JButton using private access modifier.

CREATE a method called BankGUI

        INITIALIZE myFrame as JFrame

        INITIALIZE guiLabel as JLabel

        INITIALIZE debitLabel as JLabel

        INITIALIZE idLabel as JLabel

        INITIALIZE nameLabel as JLabel

        INITIALIZE issueLabel as JLabel

        INITIALIZE bankAcLabel as JLabel

        INITIALIZE balAmtLabel as JLabel

        INITIALIZE pinLabel as JLabel

        INITIALIZE withdrawLabel as JLabel

        INITIALIZE withdrawDLabel as JLabel

        INITIALIZE withdrawlLabel as JLabel

        INITIALIZE withidLabel as JLabel

        INITIALIZE wpinLabel as JLabel
```

```
ADD guiLabel to myFrame
ADD debitLabel to myFrame
ADD idLabel to myFrame

ADD nameLabel to myFrame
ADD issueLabel to myFrame
ADD bankAcLabel to myFrame
ADD balAmtLabel to myFrame
ADD pinLabel to myFrame
ADD withdrawLabel to myFrame
ADD withdrawDLabel to myFrame
ADD withdrawlLabel to myFrame
ADD withidLabel to myFrame
ADD wpinLabel to myFrame

ADD idText to myFrame
ADD nameText to myFrame
ADD issueText to myFrame
ADD bankAcText to myFrame
ADD balAmtText to myFrame
ADD withidText to myFrame
ADD pinText to myFrame
ADD dayWComboBox to myFrame
ADD dateWComboBox to myFrame
ADD yearWComboBox to myFrame
ADD withAText to myFrame
ADD wpinText to myFrame

ADD adButton to myFrame
ADD displayButton to myFrame
ADD withdrawButton to myFrame

SET bounds to guiLabel
SET bounds to debitLabel
SET bounds to idLabel
SET bounds to nameLabel
SET bounds to issueLabel
SET bounds to bankAcLabel
SET bounds to balAmtLabel
SET bounds to pinLabel
SET bounds to withdrawLabel
SET bounds to withdrawDLabel
SET bounds to withdrawlLabel
SET bounds to withidLabel

SET bounds to idText
```

```
**SET** bounds to nameText

**SET** bounds to issueText

**SET** bounds to bankAcText

**SET** bounds to balAmtText

**SET** bounds to pinText

**SET** bounds to withAText

**SET** bounds to withidText

**SET** bounds to dayWComboBox

**SET** bounds to dateWComboBox

**SET** bounds to yearWComboBox

**SET** bounds to wpinText



**SET** bounds to adButton

**SET** bounds to displayButton

**SET** bounds to withdrawButton


**INITIALIZE** creditLabel as JLabel

**INITIALIZE** idCLabel as JLabel

**INITIALIZE** nameCLabel as JLabel

**INITIALIZE** issueCLabel as JLabel

**INITIALIZE** bankAcCLabel as JLabel

**INITIALIZE** balAmtCLabel as JLabel
```

```
INITIALIZE cvcCLabel as JLabel

INITIALIZE interestCLabel as JLabel

INITIALIZE expCLabel as JLabel

INITIALIZE limitCLabel as JLabel

INITIALIZE limitLabel as JLabel

INITIALIZE graceLabel as JLabel

INITIALIZE cardlLabel as JLabel


ADD creditLabel to myFrame

ADD idCLabel to myFrame

ADD nameCLabel to myFrame

ADD issueCLabel to myFrame

ADD bankAcCLabel to myFrame

ADD balAmtCLabel to myFrame

ADD cvcCLabel to myFrame

ADD interestCLabel to myFrame

ADD expCLabel to myFrame

ADD limitCLabel to myFrame

ADD limitLabel to myFrame

ADD graceLabel to myFrame

ADD cardlLabel to myFrame


ADD adCButton to myFrame
ADD displayCButton to myFrame

ADD limitCButton to myFrame

ADD cancelCButton to myFrame

SET bound to creditLabel

SET bound to idCLabel

SET bound to nameCLabel

SET bound to issueCLabel

SET bound to bankAcCLabel

SET bound to balAmtCLabel

SET bound to cvcCLabel

SET bound to interestCLabel

SET bound to expCLabel

SET bound to limitCLabel

SET bound to limitLabel

SET bound to graceLabel

SET bound to cardlLabel


SET bounds to idCText

SET bounds to nameCText

SET bounds to issueCText

SET bounds to bankAcCText

SET bounds to balAmtCText

SET bounds to cvcCText
SET bounds to interestCText

SET bounds to dayCComboBox

SET bounds to dateCComboBox

SET bounds to yearCComboBox

SET bounds to limitText

SET bounds to graceText

SET bounds to cardlText


SET bounds to adCButton

SET bounds to displayCButton

SET bounds to limitCButton

SET bounds to cancelCButton


INITIALIZE clrButton as JButton


FOR dayCComboBox.addItem add item (i) in dayCComboBox

FOR dateCComboBox.addItem add item (i) in dateCComboBox

FOR yearCComboBox.addItem add item (i) in yearCComboBox


INITIALIZE ArrayList BankCard as new ArrayList


SET ActionListener to adButton

SET ActionListener to addCButton
```

```
    SET ActionListener to displayButton
    SET ActionListener to displayCButton
    SET ActionListener to limitCButton
    SET ActionListener to cancelCButton
    SET ActionListener to withdrawButton

    SET size to myFrame
    SET Layout to myFrame
    SET setDefaultCloseOperation to myFrame
    SET Visible to true

CREATE a method called actionPerformed which takes no PARAMETER with a RETURN type of void
    IF e.getSource() is clrButton
        SET idText to empty String
        SET nameText to empty String
        SET issueText to empty String
        SET bankAcText to empty String
        SET balAmtText to empty String
        SET withidText to empty String
        SET pinText to empty String
        SET withAText to empty String
        SET idCText to empty String
        SET nameCText to empty String
        SET issueCText to empty String
        SET bankAcCText to empty String
        SET balAmtCText to empty String
        SET cvcCText to empty String
        SET interestCText to empty String
        SET limitText to empty String
        SET graceText to empty String
        SET cardlText to empty String
    END FOR
    IF e.getSource() is displayButton
        FOR BankCard obj: lists
            IF object not instance of DebitCard
            Continue;
        obj.display();
        END FOR
    END IF

    IF e.getSource() is adButton
        TRY
            DECLARE card_id as Integer
            DECLARE clientName as String
            DECLARE issuerBank as String
```

```
DECLARE bankAccount as String

DECLARE balanceAmount as Integer

DECLARE pinNumber as Integer

DECLARE debit_card object from class DebitCard with parameters passing balanceAmount, card_id, bankAccount, issuerBank, clientName, pinNumber passed in the constructor

DECLARE verify as boolean to value false

FOR BankCard obj: lists

    IF obj not instanceof DebitCard

        DECLARE verify to true

        Continue;

    END IF

    IF  (DebitCard)obj.getcard_id () is true

        DISPLAY “Card ID is already present”

        DECLARE verify to false

        Break

    ELSE

        DECLARE Verify to true

    END IF

END FOR

IF (list.isEmpty()) or (verify)

    Lists.add(debit_card)
```

```
                DISPLAY “Debit Card has been added”
            END IF
        CATCH NumberFormatException ex
        DISPLAY Inaccurate Data
        END TRY
    END IF


    IF e.getSource() is withdrawButton
        TRY
            DECLARE verify as Boolean to value False
            DECLARE card_id to Integer
            DECLARE date as String
            DECLARE day as String
            DECLARE year as String
            DECLARE calendar as String to value date / day / year
            DECLARE balance_amount as integer
            DECLARE pin_number as integer
            FOR BankCard obj: lists
                IF obj not instanceof DebitCard
                Continue
                END IF
                IF obj.getcard_id() is card_id
                    DECLARE verify to true
                    CALL ((DebitCard)obj).withdraw(balance_amount, date, pin_number)
                    IF draw is 1
                        DISPLAY “Amount has been successfully withdrawn.”
                    ELSE draw is 2
                        DISPLAY “Incorrect Pin Number”
                    ELSE IF draw is 3
                        DISPLAY “Insufficient Balance.”
                    END IF
            END FOR
            IF list.isEmpty
                DISPLAY “ERROR!! Please confirm if Debit Card is Present.”

            END IF
            IF not verify
                DISPLAY “Could not find the Card ID.”
            END IF
        CATCH
            DISPLAY “ERROR!! Verify if the data inserted is correct.”
        END TRY
    END IF
    IF e.getSource() == displayCButton
        FOR BankCard obj: lists
            IF object not instance of CreditCard
            Continue;
        obj.display();
        END FOR
    END IF

    IF e.getSource() is adCButton
        TRY
            DECLARE card_id as Integer
            DECLARE clientName as String
            DECLARE issuerBank as String
            DECLARE bankAccount as String
            DECLARE balanceAmount as Integer
            DECLARE cvcNumber as Integer
            DECLARE interestRate as double
```

```
DECLARE date as String
DECLARE day as String
DECLARE year as String
DECLARE expirationDate as String to value date / day / year
DECLARE verify as boolean to value false
FOR BankCard obj: lists
    IF obj not instanceof CreditCard
        DECLEAR verify to true
        Continue;
    END IF
    IF ((CreditCard)obj).getcard_id () is true
        DISPLAY "Card ID is already present".
        DECLARE verify to false
        Break
    ELSE
        DECLARE Verify to true
    END IF
END FOR
IF list.isEmpty() or (verify)
    Lists.add(debit_card)
    DISPLAY "Successfully added the CreditCard."
```

```
                END IF
            CATCH NumberFormatException ex
            DISPLAY "Invalid Data."
            END TRY
    END IF
    IF e.getSource() is limitCButton
            TRY
                DECLARE verify as boolean to value false
                DECLARE card_id as Integer
                DECLARE creditLimit ad Integer
                DECLARE gracePeriod as Integer
                FOR BankCard obj: lists
                    IF obj not instanceof CreditCard
                        Continue
                    END IF
                    IF (CreditCard)obj).getcard_id() is card_id
                        DECLARE verify to true
                        CALL((CreditCard)obj).setcreditLimit(creditLimit,grac
                        ePeriod)
                        DISPLAY "Credit Limit has been successfully set."
                    END IF
                    IF list.isEmpty
                        DISPLAY "Empty Creditcard."
                    END IF
                    IF not verify
                        DISPLAY "Could not find the Card ID."
                    END IF
            CATCH
                DISPLAY "Incorrect data"
            END TRY
    END IF

    IF e.getSource() is limitCButton
            TRY
                DECLARE verify as boolean to value false
                DECLARE card_id as Integer
                FOR BankCard obj: lists
                    IF obj not instanceof CreditCard
                        Continue
                    END IF
                    IF obj.getcard_id() is card_id
                        **DECLARE** verify to true
                        **CALL**((CreditCard)obj).cancelCreditCard()
                        **DISPLAY** “Credit has been Cancelled.”
                    **END IF**
                **IF** not verify
                    **DISPLAY** “Could not find the Card ID.”
                **END IF**
            **CATCH**
                **DISPLAY** “Input the Card ID.”
            **END TRY**
    **END IF**
```

4. Method Description

Methods also referred to as functions in Java is a collection of code that performs a specific task or operation. Values and parameters can be inserted into methods which will only be executed when called (javatpoint, 2023).

4.1 Describing the Methods

Methods	Description
BankGUI()	BankGUI is a method that is calledwhen the BankGUI class is executed.
actionPerformed(ActionEvent e)	actionPerformed(ActionEvent e) an event is called when a button is clicked.
clrButton	clrButton clears the data that is filled in the Text fields in the GUI.
adButton	adButton Adds card_id, issuerBank, bankAccount, clientName and balanceAmount.
withdrawButton	The withdrawButton displays the data filled in card_id, issuerBank, bankAccount, clientName, balanceAmount, pinNumber, withdrawalAmount and dateOfWithdrawal.
displayButton	The displayButton displays the data filled in Card ID, Issuer Bank, Bank Account, Client Name and Balance Amount from DebitCard, and withdrawalAmount and dateOfWithdrawal from Withdrawal.
adCButton	The adCButton adds the data filled in card_id, issuerBank, bankAccount, clientName, cvcNumber, interestRate, ExpirationDate.

limitCButton	The limitCButton sets the Credit limit and the Grace Period.
cancelCButton	The cancelCButton cancels the credit limit that was set.
displayCButton	The displayCButton displays the data filled in card_id, issuerBank, bankAccount, clientName, cvcNumber, interestRate, ExpirationDate.

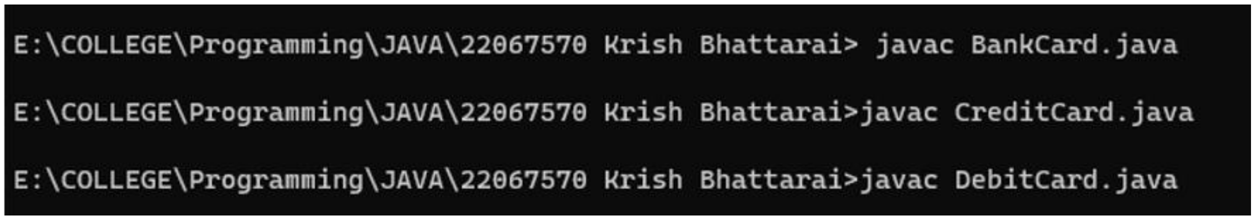
Table 1: Method Description

5. Testing

5.1 Test 1 - To Compile and Run using command prompt.

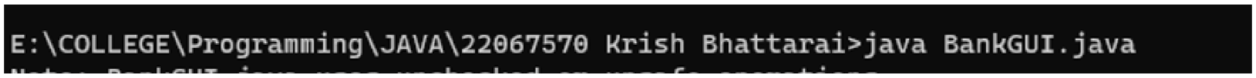
Test No:	1
Objective:	To compile and run the program using command prompt.
Action:	• The BankCard class is compiled using the command prompt. • The CreditCard class is compiled using the command prompt. • The DebitCard class is compiled using the command prompt. • The BankGUI class is compiled using the command prompt. • Inspect BankCard, CreditCard, DebitCard, BankGUI.
Expected Result:	The BankGUI class should be compiled and displayed.
Actual Result:	The BankGUI class was compiled and displayed.
Conclusion:	The test is successful.

Table 2: To Compile and Run using Command Prompt



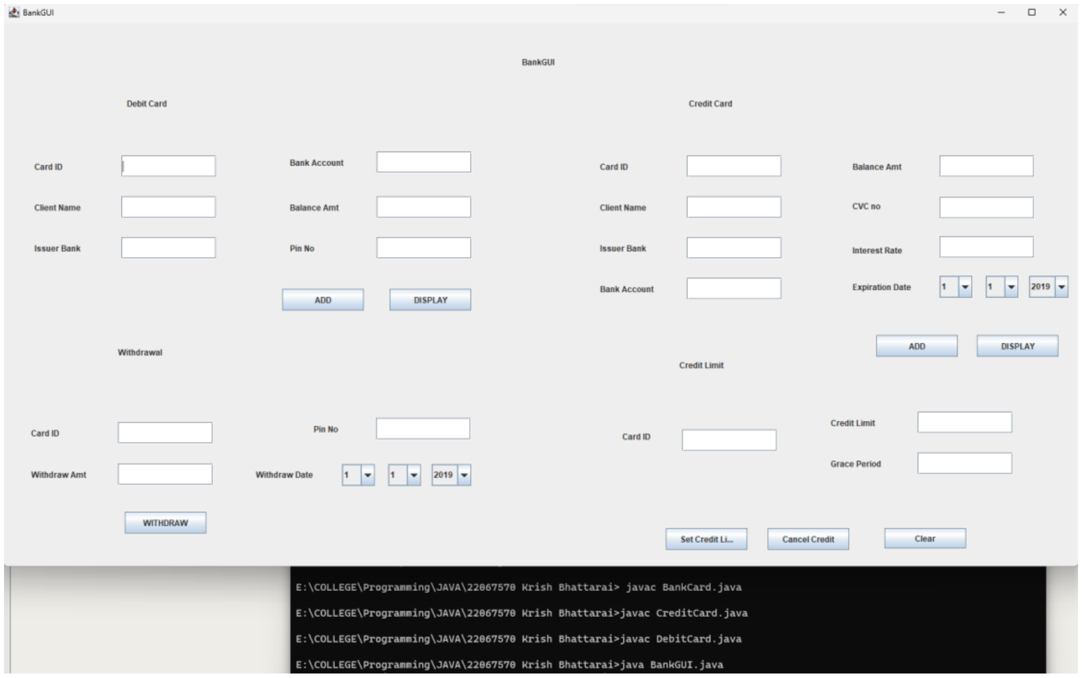
```
E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai> javac BankCard.java

E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai>javac CreditCard.java

E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai>javac DebitCard.java
```

Figure 6: Screenshot of Classes being compiled:



```
E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai>java BankGUI.java
```

Figure 7: Screenshot of running BankGUI.



```
E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai> javac BankCard.java

E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai>javac CreditCard.java

E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai>javac DebitCard.java

E:\COLLEGE\Programming\JAVA\22067570 Krish Bhattarai>java BankGUI.java
```

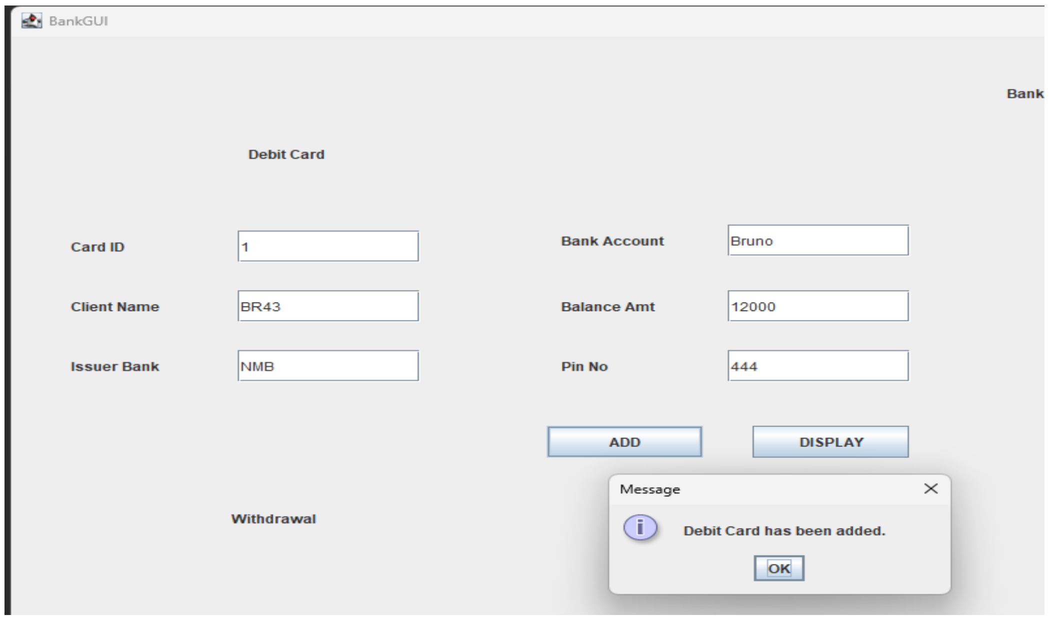
Figure 8: Screenshot of the GUI

5.2 Test 2 Adding objects of DebitCard and CreditCard, withdrawing amount from debit card, setting the credit limit, and removing the credit card)

Test 2 - Test the ADD button in Debit Card.

Test No:	2
Objective:	Test the ADD button in Debit Card.
Action:	• Displaying BankGUI • Data is added in the Debit Card text fields. • Add Button is clicked. • Pop up is displayed.
Expected Result:	The ADD button should work.
Actual result:	The ADD button worked.
Conclusion:	Test Successful.

Table 3: Test the ADD button in Debit Card.



Card ID: 1
Client Name: BR43
Issuer Bank: NMB
Bank Account: Bruno
Balance Amt: 12000
Pin No: 444

Debit Card has been added.

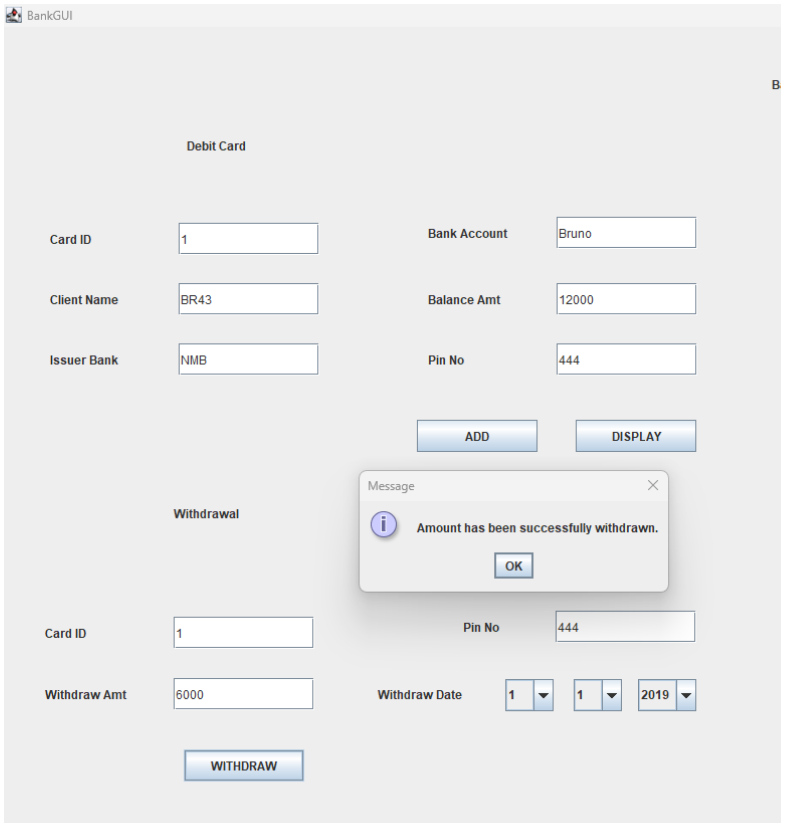
Figure 9: Screenshot of Testing the ADD button in Debit Card.

.

Test 3 – Test the WITHDRAW button.

Test No:	3
Objective:	Test the WITHDRAW button.
Action:	• Data is added in the Withdrawal text fields. • Withdraw button is pressed. • Appropriate Pop Up is displayed.
Expected Result:	The WITHDRAW button should work.
Actual result:	The WITHDRAW button worked.
Conclusion:	Test Successful.

Table 4: Test the WITHDRAW button.



Card ID: 1
Bank Account: Bruno
Client Name: BR43
Balance Amt: 12000
Issuer Bank: NMB
Pin No: 444

Amount has been successfully withdrawn.

Card ID: 1
Pin No: 444
Withdraw Amt: 6000
Withdraw Date: 1 1 2019

Figure 10: Screenshot of testing the WITHDRAW button.

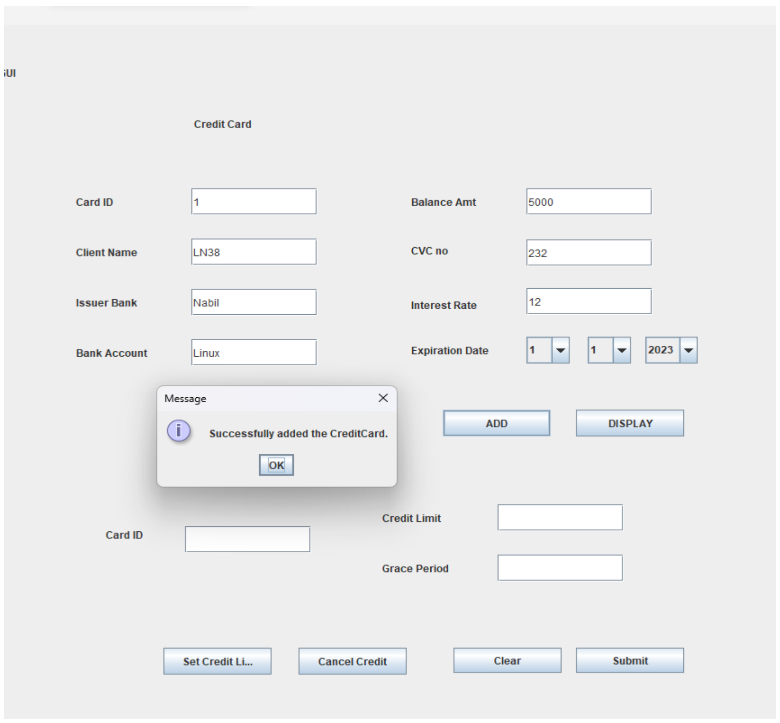
Test 4 - Test the Add Credit Card function.

Test No:	4

Objective:	Test the Add Credit Card function.
Action:	• Data is added in the Credit Card test fields. • Add Button is clicked. • Appropriate Pop Up is displayed
Expected Result:	The ADD button should work.
Actual result:	The ADD button worked.
Conclusion:	Test Successful.

Table 5: Test the Add Credit Card function.

.



Card ID: 1
Client Name: LN38
Issuer Bank: Nabil
Bank Account: Linux
Balance Amt: 5000
CVC no: 232
Interest Rate: 12
Expiration Date: 1 1 2023

Message
Successfully added the CreditCard.
OK

Figure 11: Screenshot of testing the Add Credit Card function.

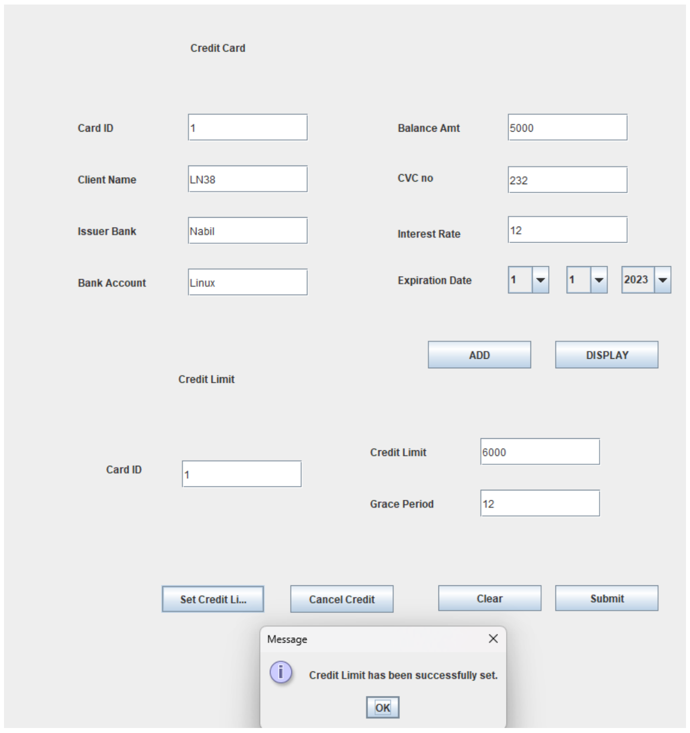
Test 5 - Test the Set Credit Limit Button.

Test No:	5
Objective:	Test the Set Credit Limit Button.
Action:	• Data is added in the Credit Limit text fields.

	• Set Credit Limit Button is clicked. • Appropriate Pop Up is displayed.
Expected Result:	The Set Credit Limit Button should work.
Actual result:	The Set Credit Limit Button worked as expected.
Conclusion:	Test Successful.

Table 6: Test the Set Credit Limit Button.

.



Credit Limit has been successfully set.

Figure 12: Screenshot of testing the Set Credit Limit Button.

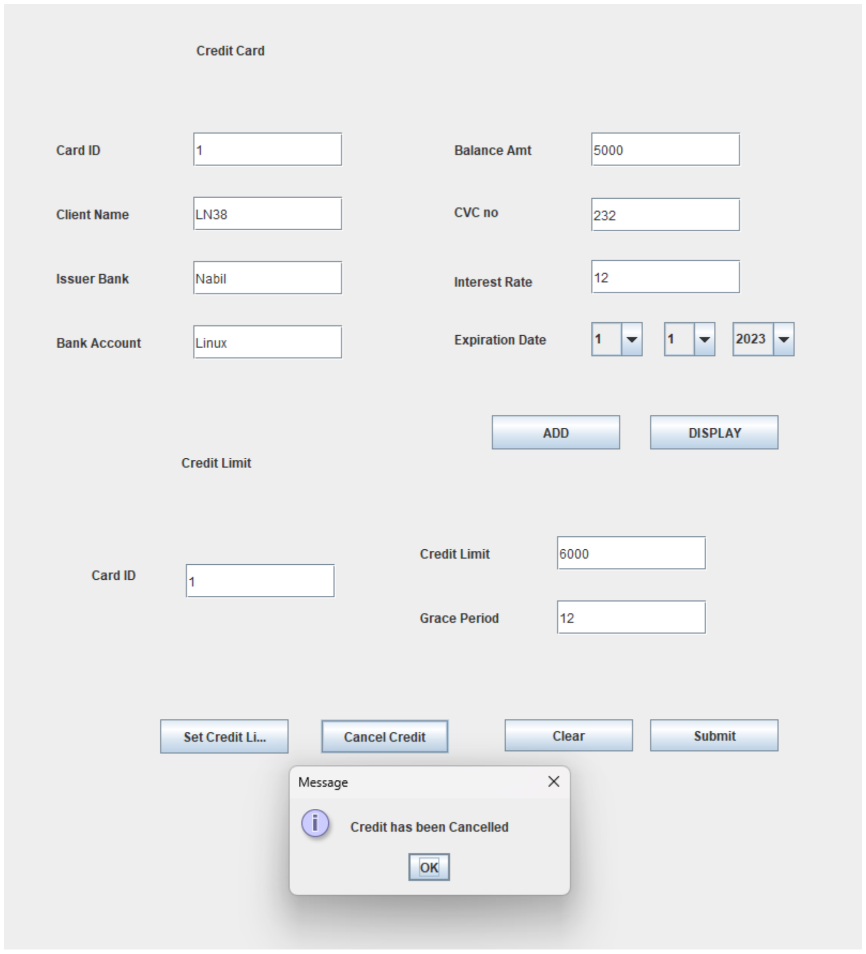
Test 6 - Cancel the Credit Limit.

Test No:	6
Objective:	Cancel the Credit Limit.
Action:	• Cancel Credit button is pressed. • Appropriate Pop Up is displayed.

Expected Result:	The Cancel Credit button should work.
Actual result:	The Cancel Credit button worked as expected.
Conclusion:	Test Successful.

Table 7: Remove the Credit Limit.

.


Credit Card

Card ID: 1
Client Name: LN38
Issuer Bank: Nabil
Bank Account: Linux
Balance Amt: 5000
CVC no: 232
Interest Rate: 12
Expiration Date: 1 1 2023

Credit Limit

Card ID: 1
Credit Limit: 6000
Grace Period: 12

Message

Credit has been Cancelled

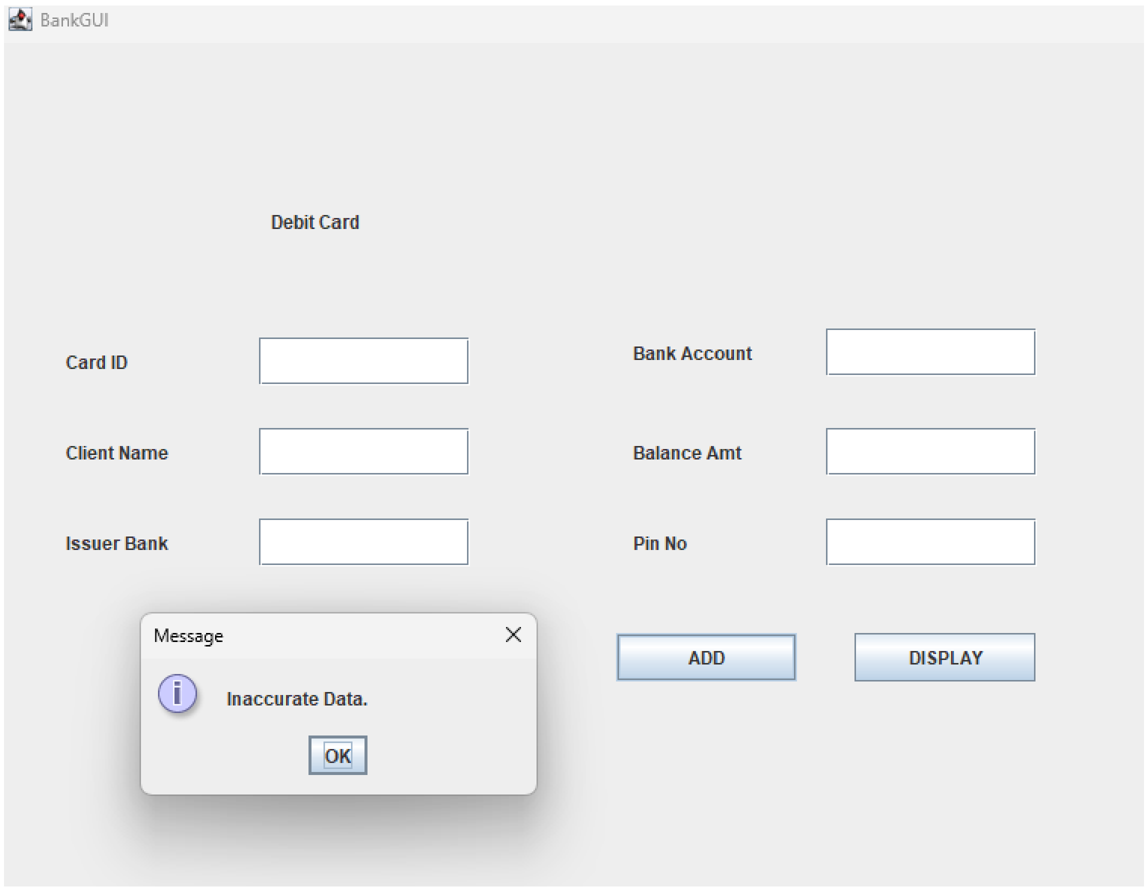
Figure 13: Testing the Cancel the Credit Limit.

Test 3 (Testing the Appropriate Dialog boxes when unsuitable values are entered)

Test 7 – Test the Add button in Debit Card when the text fields are empty.

Test No:	7
Objective:	Test the Add button in Debit Card when the text fields are empty.
Action:	• Leaving the Text fields empty. • Clicking the ADD button. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up saying Inaccurate Data should appear.
Actual result:	A Pop Up saying Inaccurate Data appeared.
Conclusion:	Test Successful.

Table 8: Test the Add button in Debit Card when the text fields are empty.



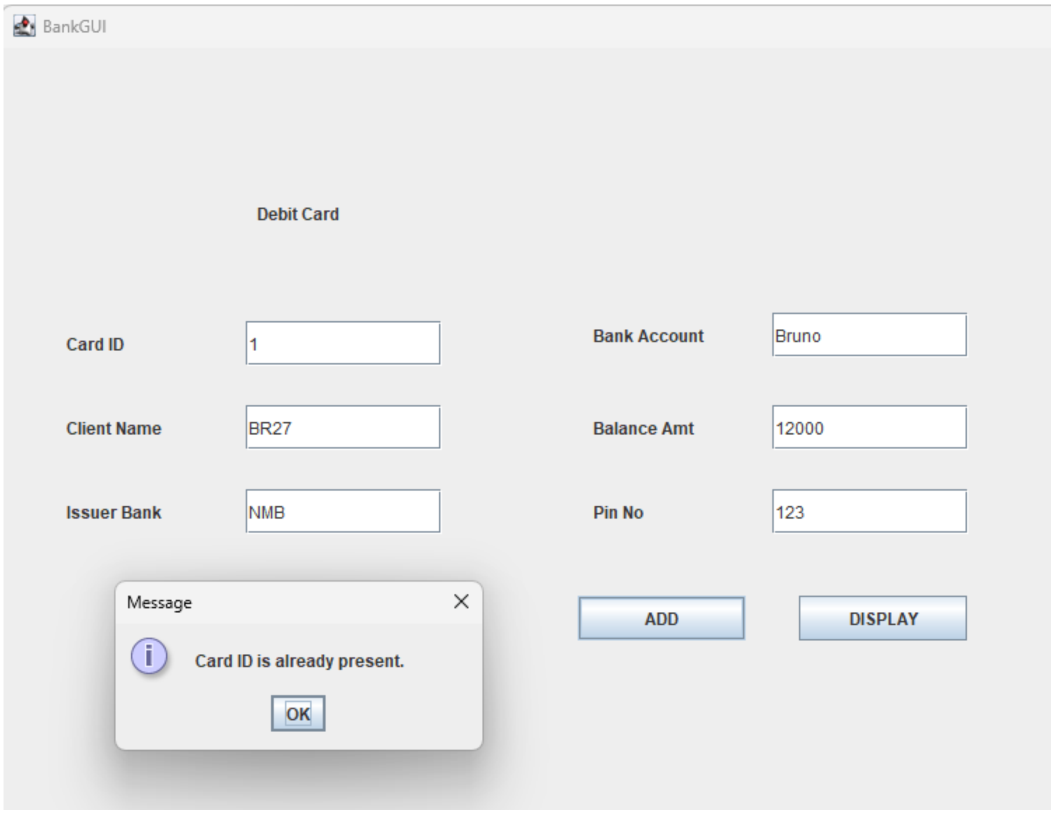
Inaccurate Data.

Figure 14: Screenshot of Testing the Add button in Debit Card when the text fields are empty.

Test 8 - Test by clicking the Add button twice in Debit Card twice.

Test No:	8
Objective:	Test by clicking the Add button twice in Debit Card twice.
Action:	• Data is added in the text fields. • ADD button is pressed twice. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up with an error message should appear.
Actual result:	A Pop Up with an error message appeared.
Conclusion:	Test Successful.

Table 9: Test by clicking the Add button twice in Debit Card twice



Card ID: 1
Client Name: BR27
Issuer Bank: NMB
Bank Account: Bruno
Balance Amt: 12000
Pin No: 123

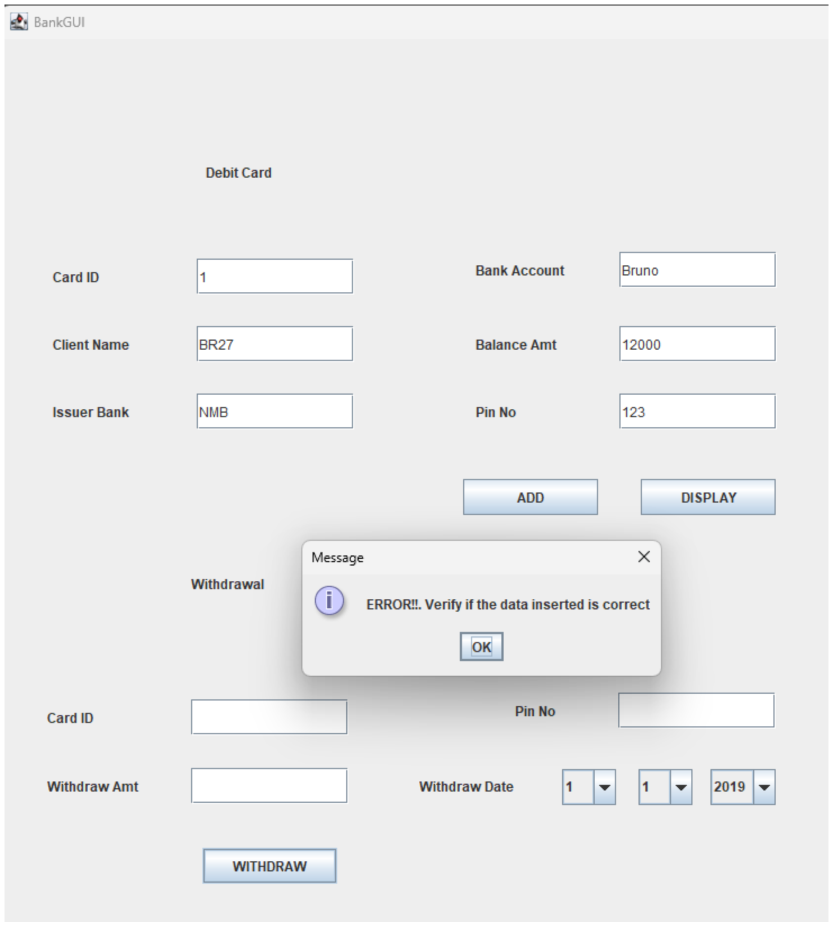
Card ID is already present.

Figure 15: Screenshot of Testing by clicking the Add button twice in Debit Card twice

Table 9 - Test the withdrawal when the text fields are empty.

Test No:	9
Objective:	Test the withdrawal when the text fields are empty.
Action:	• Leaving the Text fields empty. • Clicking the WITHDRAW button. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up with an error message should be displayed.
Actual result:	A Pop Up with an error message is displayed.
Conclusion:	Test is Successful.

Table 10: Test the withdrawal when the text fields are empty.



Card ID: 1
Bank Account: Bruno
Client Name: BR27
Balance Amt: 12000
Issuer Bank: NMB
Pin No: 123

ERROR!!. Verify if the data inserted is correct

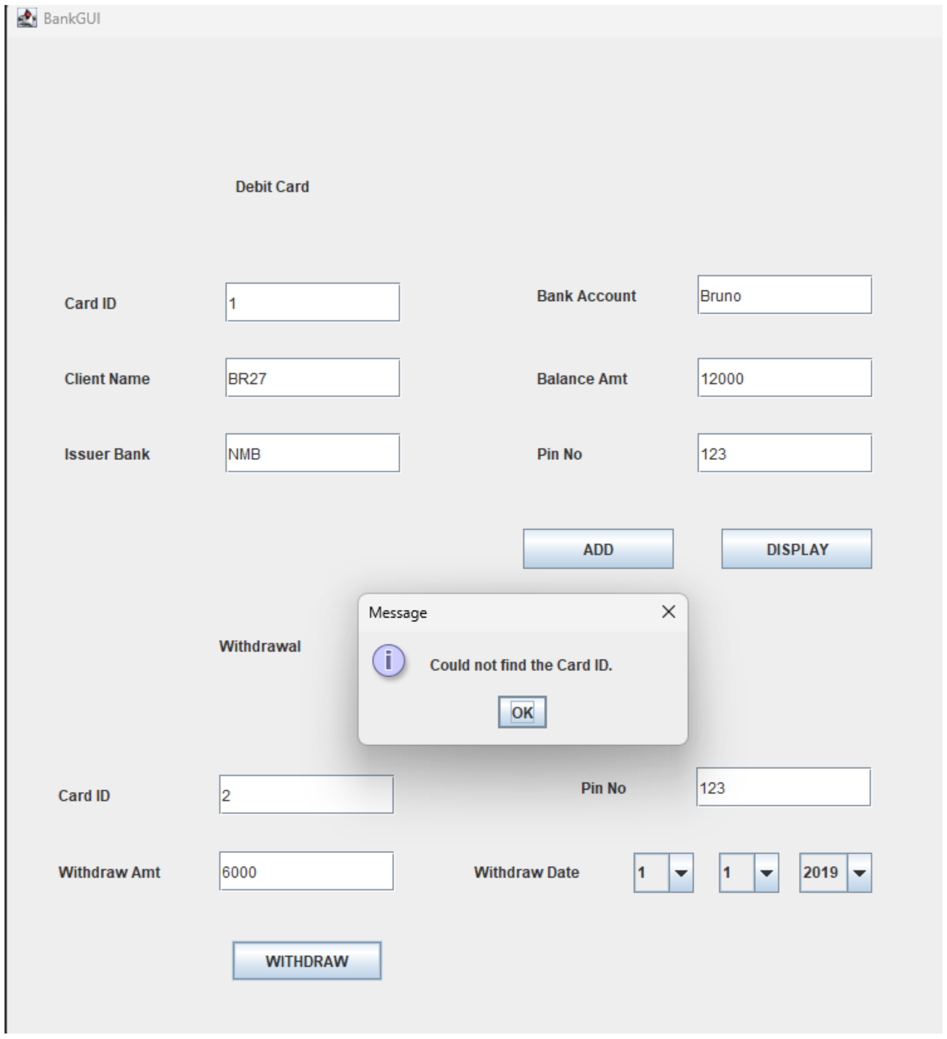
Card ID:
Pin No:
Withdraw Amt:
Withdraw Date: 1 1 2019

Figure 16: Screenshot of Testing the withdrawal when the text fields are empty.

Table 10 - Testing when an inappropriate Card ID is inserted.

Test No:	10
Objective:	Testing when an inappropriate Card ID is inserted.
Action:	• Data is added in the text fields with an inappropriate Card ID. • WITHDRAW button is pressed. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up with an error message should be displayed.
Actual result:	A Pop Up with an error message is displayed.
Conclusion:	Test Successful.

Table 11: Testing when an inappropriate Card ID is inserted.



Card ID: 1
Client Name: BR27
Issuer Bank: NMB
Bank Account: Bruno
Balance Amt: 12000
Pin No: 123

Could not find the Card ID.

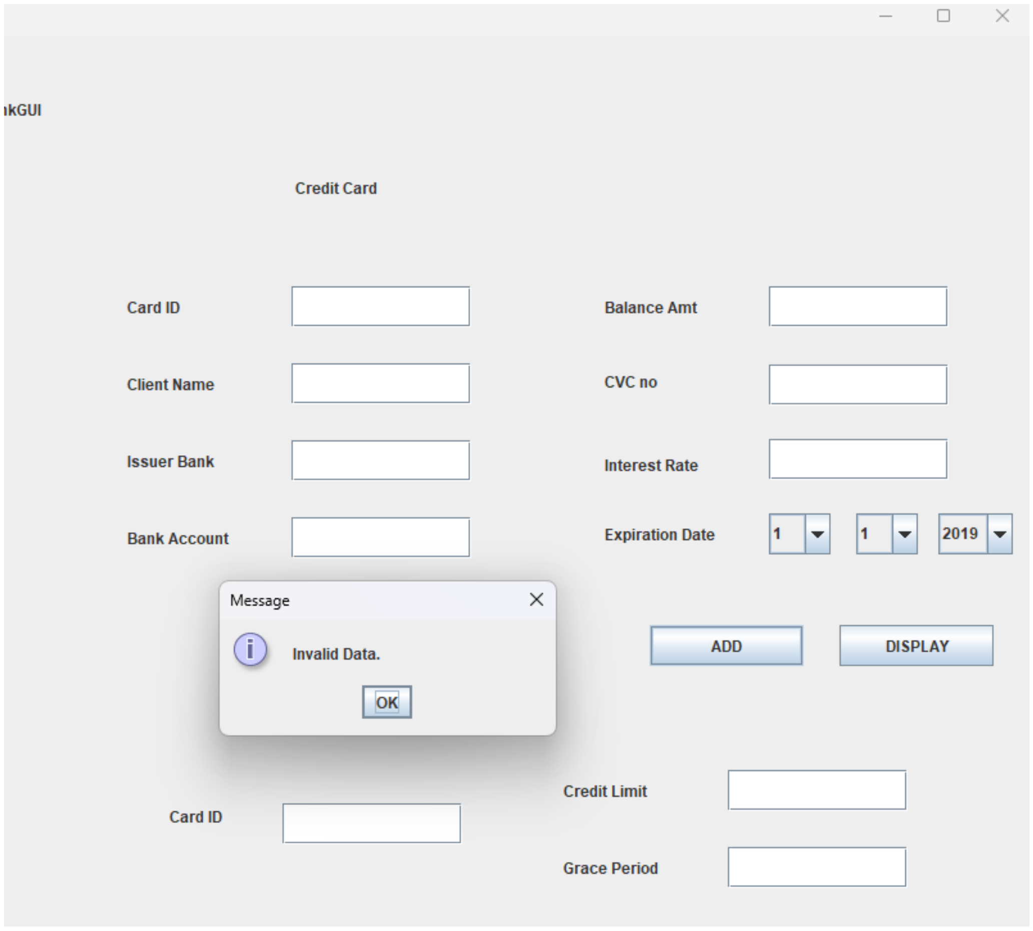
Card ID: 2
Withdraw Amt: 6000
Pin No: 123
Withdraw Date: 1 1 2019

Figure 17: Screenshot of Testing when an inappropriate Card ID is inserted.

Table 11 - Testing the Credit Card when the text fields are empty.

Test No:	11
Objective:	Testing the Credit Card when the text fields are empty.
Action:	• Leaving the Text fields empty. • Clicking the ADD button. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up with an error message should be displayed.
Actual result:	A Pop Up with an error message is displayed.
Conclusion:	Test Successful.

Table 12: Testing the Credit Card when the text fields are empty.



Message

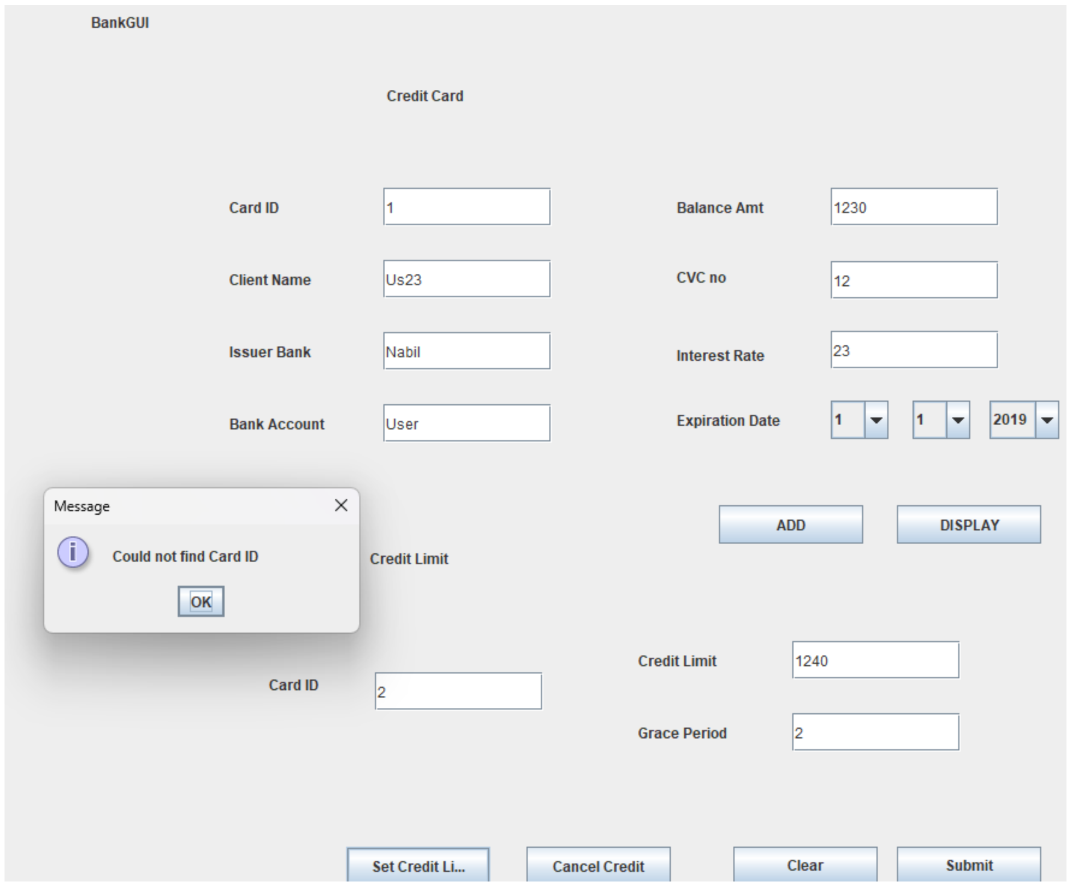
Invalid Data.

Figure 18: Screenshot of Testing the Credit Card when the text fields are empty.

Test 12 - Insert an invalid Card ID in Credit Limit.

Test No:	12
Objective:	Insert an invalid Card ID in Credit Limit.
Action:	• Data is added in the text fields with an inappropriate Card ID. • Set Credit Limit button is pressed. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up with an error message should be displayed.
Actual result:	A Pop Up with an error message is displayed.
Conclusion:	Test Successful.

Table 13: Insert an invalid Card ID in Credit Limit.



Card ID: 1
Client Name: Us23
Issuer Bank: Nabil
Bank Account: User
Balance Amt: 1230
CVC no: 12
Interest Rate: 23
Expiration Date: 1 1 2019

Could not find Card ID

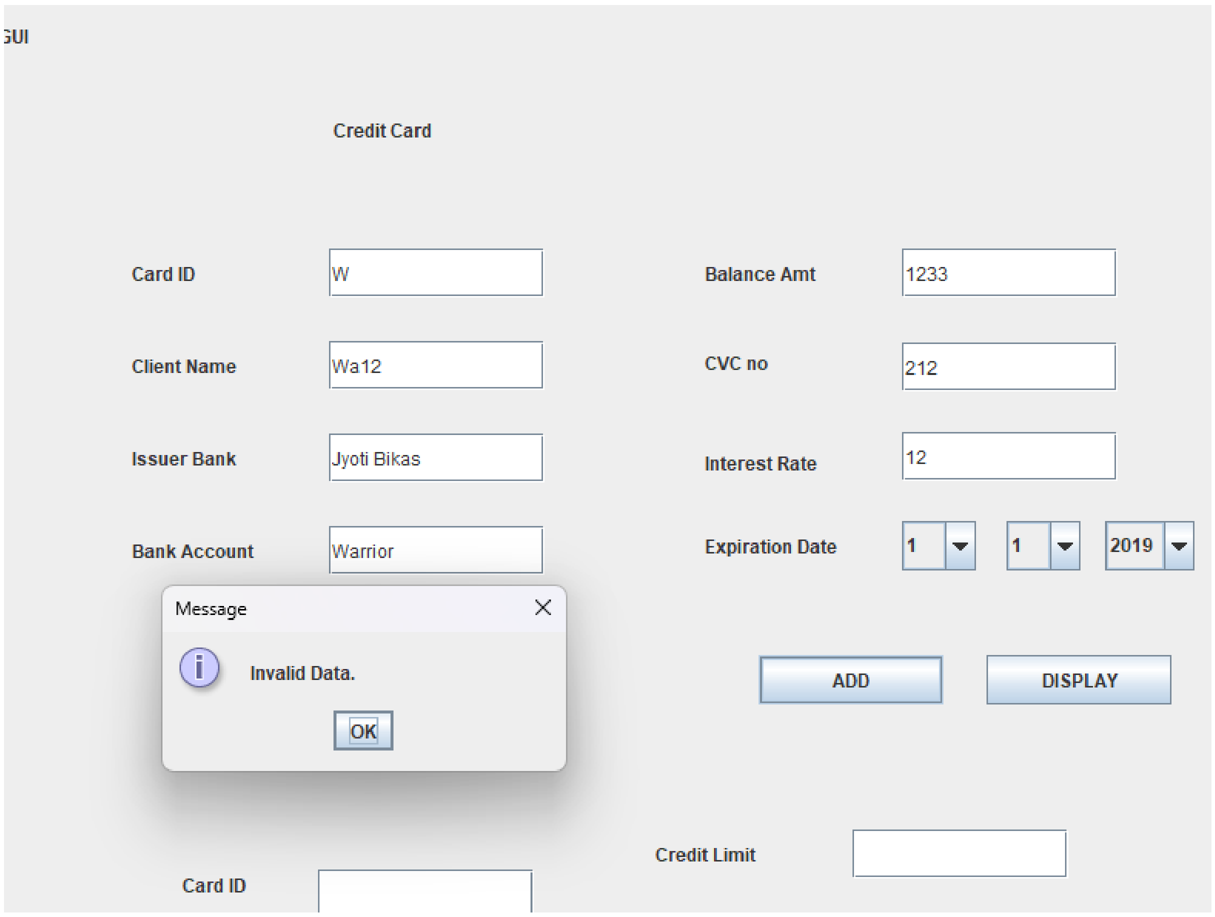
Card ID: 2
Credit Limit: 1240
Grace Period: 2

Figure 19: Screenshot of testing by Inserting an invalid Card ID in Credit Limit.

Test 13 - Insert an invalid data in Credit Card, Card ID.

Test No:	13
Objective:	Insert an invalid data in Credit Card, Card ID.
Action:	• Data is added in the text fields with an inappropriate Card ID. • ADD button is pressed. • A Pop Up with an error message is displayed.
Expected Result:	A Pop Up with an error message should be displayed.
Actual result:	A Pop Up with an error message is displayed.
Conclusion:	Test Successful.

Table 14: Insert an invalid data in Credit Card, Card ID.



Card ID: W
Client Name: Wa12
Issuer Bank: Jyoti Bikas
Bank Account: Warrior
Balance Amt: 1233
CVC no: 212
Interest Rate: 12
Expiration Date: 1 1 2019
Invalid Data.
Credit Limit:
Card ID:

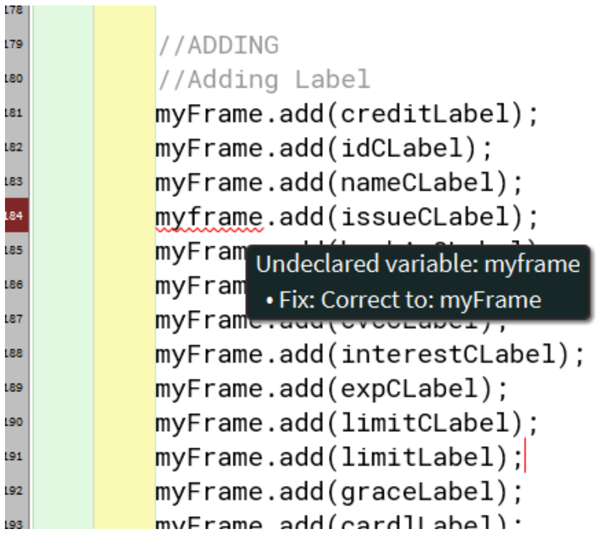
Figure 20: Screenshot of testing by Inserting an invalid data in Credit Card, Card ID.

6. Error Detection

6.1 Syntax Error

A syntax Error occurs when there is an error in the syntax. For example, misspelling a word or missing a comma or a quotation mark. A syntax error is flagged by the editor if there is a syntax error in the code (Elsevier B.V., 2023).

There was a syntax error in the code. myFrame was accidentally misspelled to myframe.

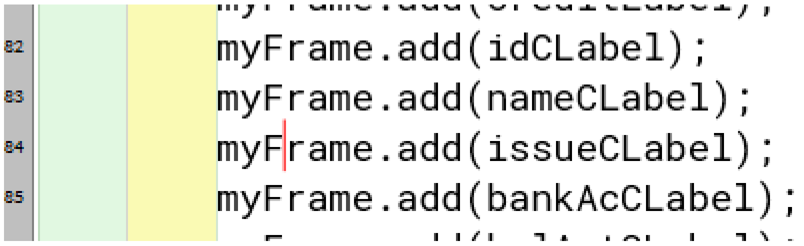


```
//ADDING
//Adding Label
myFrame.add(creditLabel);
myFrame.add(idCLabel);
myFrame.add(nameCLabel);
myframe.add(issueCLabel);
myFram
myFram
myFrame
myFrame.add(interestCLabel);
myFrame.add(expCLabel);
myFrame.add(limitCLabel);
myFrame.add(limitLabel);
myFrame.add(graceLabel);
```

Undeclared variable: myframe
• Fix: Correct to: myFrame

Figure 21: Screenshot of a syntax error.

The syntax error was fixed. myFrame is now correctly spelled.



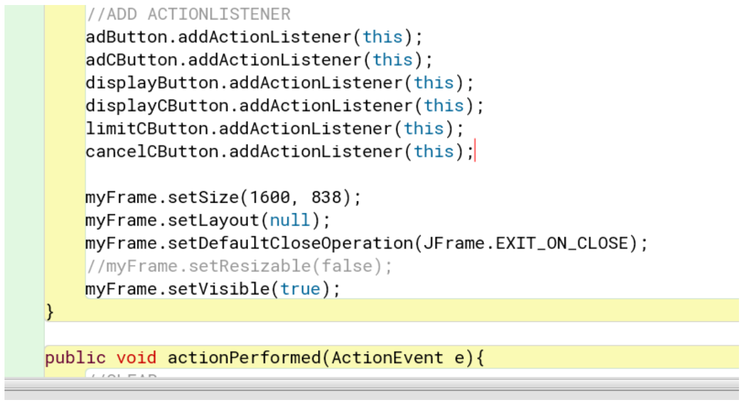
```
myFrame.add(idCLabel);
myFrame.add(nameCLabel);
myFrame.add(issueCLabel);
myFrame.add(bankAcCLabel);
```

Figure 22: Screenshot of fixing the syntax error.

6.2 Runtime Error

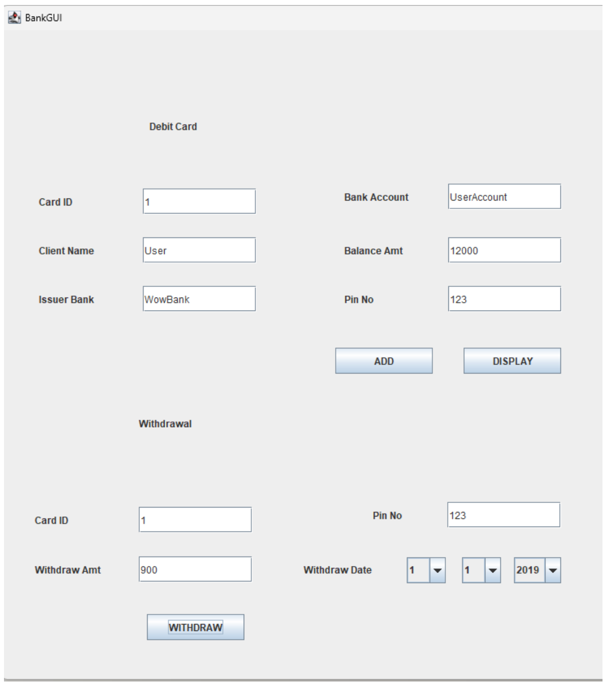
When a program is syntactically correct but contains an issue which is only detected when the program is executed is known as Runtime error. These issues are not caught while compiling a code but are only detected when the program is running (Rollbar, 2023).

There was a Runtime Error in the code. The WITHDRAW button is not working because the ActionListener for the withdraw button is not present.



```
    //ADD ACTIONLISTENER
    adButton.addActionListener(this);
    adCButton.addActionListener(this);
    displayButton.addActionListener(this);
    displayCButton.addActionListener(this);
    limitCButton.addActionListener(this);
    cancelCButton.addActionListener(this);

    myFrame.setSize(1600, 838);
    myFrame.setLayout(null);
    myFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    //myFrame.setResizable(false);
    myFrame.setVisible(true);
}

public void actionPerformed(ActionEvent e){
```

Figure 23: Screenshot of a Runtime Error.

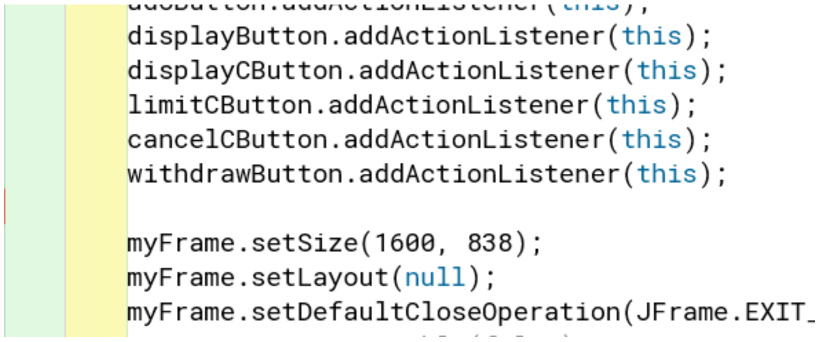


Card ID: 1
Client Name: User
Issuer Bank: WowBank
Bank Account: UserAccount
Balance Amt: 12000
Pin No: 123

Card ID: 1
Pin No: 123
Withdraw Amt: 900
Withdraw Date: 1 1 2019

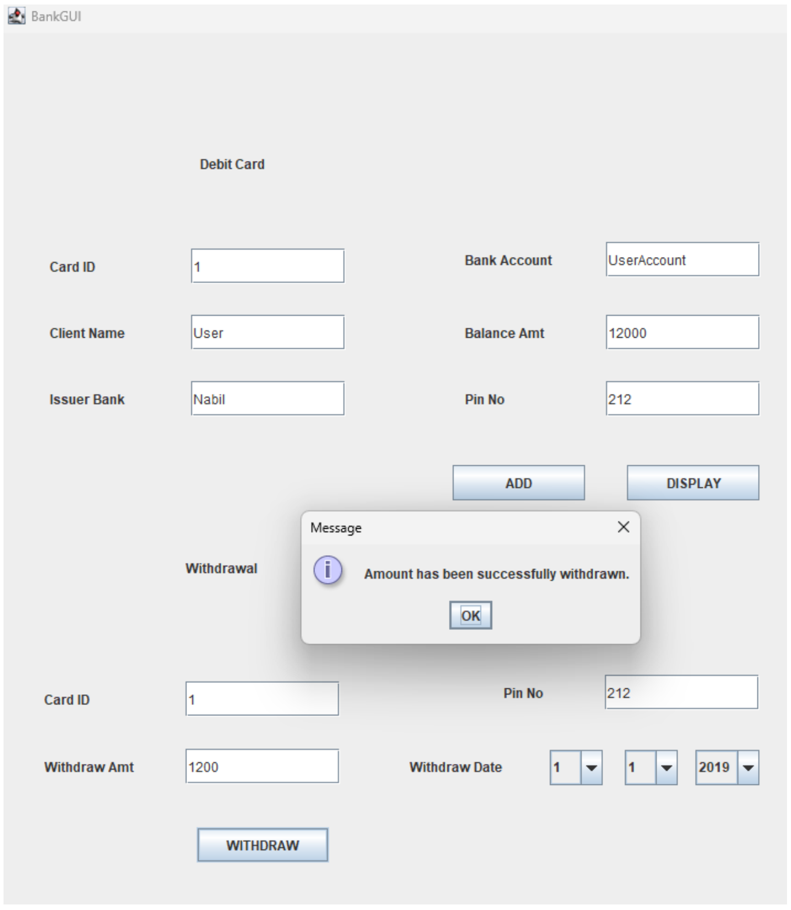
Figure 24: Screenshot of a Runtime error.

The Runtime Error is now fixed. The WITHDRAW button now works because the ActionListener for the withdraw button is present.



```
displayButton.addActionListener(this);
displayCButton.addActionListener(this);
limitCButton.addActionListener(this);
cancelCButton.addActionListener(this);
withdrawButton.addActionListener(this);

myFrame.setSize(1600, 838);
myFrame.setLayout(null);
myFrame.setDefaultCloseOperation(JFrame.EXIT_
```

Figure 25: Screenshot of fixing the Runtime Error.



BankGUI

Debit Card

Card ID	1	Bank Account	UserAccount
Client Name	User	Balance Amt	12000
Issuer Bank	Nabil	Pin No	212

Withdrawal

Message

Amount has been successfully withdrawn.

Card ID	1	Pin No	212
Withdraw Amt	1200	Withdraw Date	1 / 1 / 2019

Figure 26: Screenshot of fixing the Runtime Error.

7. Conclusion

This coursework covers the creation of a class called BankGUI in IDE BlueJ where all the code was compiled. It required creating a GUI which allows users to register the values of different attributes. The coursework encompassed various concepts such as creating a GUI, Action Events, Action listener, Array Lists, and other relevant topics. This project provided me with an opportunity to reflect on what I have learned. It has helped me acquire various skills such as creating a creating a GUI, creating an Array List, using the ActionListener for creating usable buttons. The class ‘BankGUI’ was created to input data in the previously created classes, ‘CreditCard’, ’DebitCard’ and ‘BankCard’. This coursework helped improve my research skill which assisted in overcoming the various difficulties I faced. Additionally, the study materials that were provided by the teacher rendered substantial assistance. Various concepts that were needed to be learned to do this coursework were writing a code, creating a class diagram, writing a pseudocode, testing the GUI, finding, and fixing the errors that occurred while writing the code. Furthermore, this coursework has helped me acquire different skills and gain valuable knowledge.

Bibliography

Elsevier B.V., 2023. *Syntax Error.* [Online]
Available at: https://www.sciencedirect.com/topics/engineering/syntax-error#:~:text=A%20syntax%20error%20occurs%20when%20the%20programmer%20writes%20an%20instruction,cannot%20be%20assigned%20as%20variables
[Accessed 7 5 2023].

IBM, 2023. *Java - IBM Topics.* [Online]
Available at: https://www.ibm.com/topics/java
[Accessed 7 5 2023].

javatpoint, 2023. *Java Method - javatpoint.* [Online]
Available at: https://www.javatpoint.com/method-in-java
[Accessed 7 5 2023].

Rollbar, 2023. *The Most Common Java Runtime Errors. Rollbar Blog.* [Online]
Available at: https://rollbar.com/blog/most-common-java-runtime-errors/
[Accessed 7 5 2023].

tutorialspoint, 2023. *Tutorials Point.* [Online]
Available at: https://www.tutorialspoint.com/uml/uml_class_diagram.htm
[Accessed 7 5 2023].

Appendix

Appendix of BankCard

```
/**
 * Write a description of class BankCard here.
 *
 * @author (22067570 Krish Bhattarai)
 * @version (1.0.0)
 */
public class BankCard
//main class
{
    //Declaring the Attributes
    private int card_id;
    private int balanceAmount;
    private String clientName;
    private String issuerBank;
    private String bankAccount;

    //Creating constructor method
    public BankCard(int card_id, String issuerBank, String bankAccount, int
balanceAmount){
        //Assigning values of parameter to attributes
        this.card_id = card_id;
        this.issuerBank = issuerBank;
        this.bankAccount = bankAccount;
        this.balanceAmount = balanceAmount;

        //Setting the value of clientName to an empty String
        this.clientName = " ";
    }
    //Providing setter method for clientName
    public void setclientName(String newclientName){
        clientName = newclientName;
    }

    //Providing setter method for balanceAmount
    public void setbalanceAmount(int newbalanceAmount){
        balanceAmount = newbalanceAmount;
    }



    //providing getter method for clientName
    public String getclientName(){
        return this.clientName;
```

```
}


//Providing getter method for issuerBank
public String getissuerBank(){
    return this.issuerBank;
}


//Providing getter method for bankAccount
public String getbankAccount(){
    return this.bankAccount;
}


//Providing getter method for balanceAmount
public int getbalanceAmount(){
    return this.balanceAmount;
}


public int getcard_id(){
    return this.card_id;
}


//Creating method
public void display(){
        //Displaying the values

        System.out.println("card_id:" +this.card_id);

        System.out.println("issuerBank:" +this.issuerBank);

        System.out.println("bankAccount:" +this.bankAccount);

        //System.out.println("balanceAmount:" +this.balanceAmount);


        //Checking if clientName has a value or is empty

        if(clientName!=" "){

            System.out.println("clientName: "+this.clientName);

        }

        else{

            System.out.println("This field is empty");

        }

    }



}
```

Appendix of DebitCard

```
/**

 * Write a description of class DebitCard here.

 *
 * @author (22067570 Krish Bhattarai)
 * @version (1.0.0)
 */
public class DebitCard extends BankCard
{
    //Declaring the Attributes
    private int pinNumber;
    private int WithdrawalAmount;
    private String dateOfWithdrawal;
    private boolean hasWithdrawn;

    //Creating constructor method
    public DebitCard(int balanceAmount, int card_id, String bankAccount, String
issuerBank, String clientName, int pinNumber){

        //Creating a super constructor
        super(card_id,issuerBank,bankAccount,balanceAmount);
        //Assigning value of parameter to PINnumber
        setclientName(clientName);
        this.pinNumber = pinNumber;

        //Setting the value of hasWithdrawn to False
        this.hasWithdrawn = false;
```

```
}
//Providing setter method for withdrawal amount
public void setWithdrawalAmount(int newWithdrawalAmount){
    WithdrawalAmount = newWithdrawalAmount;
}

//Providing getter method for PINnumber
public int pinNumber(){
    return this.pinNumber;
}

//Providing getter method for WithdrawalAmount
public int WithdrawalAmount(){
    return this.WithdrawalAmount;
}

//Providing getter method for dateOfWithdrawal
public String dateOfWithdrawal(){
    return this.dateOfWithdrawal;
}

//Providing getter method for hasWithdrawn
public boolean hasWithdrawn(){
        return this.hasWithdrawn;
    }


    //Creating a method called withdraw that verifies if the pin number is correct and checks
whether there is enough balance before completing the withdrawal
    public int withdraw(int WithdrawalAmount, String dateOfWithdrawal, int pinNumber){
        if(pinNumber == this.pinNumber && WithdrawalAmount<=getbalanceAmount())
        {
            super.setbalanceAmount(super.getbalanceAmount() - WithdrawalAmount);
            this.WithdrawalAmount = WithdrawalAmount;
            this.dateOfWithdrawal = dateOfWithdrawal;
            this.hasWithdrawn = true;
        }
        else if(pinNumber != this.pinNumber){
            System.out.println("The PIN number is incorrect.");
            return 2;
        }
        else{
            System.out.println("Your balance is insufficient.");
            return 3;
        }
        return 1;
    }
    //Creating a display method
    public void display(){
        super.display();
        if(hasWithdrawn == true){
            System.out.println("balanceAmount: "+getbalanceAmount());
            System.out.println("pinNumber: " +pinNumber);
            System.out.println("withdrawalAmount: "+WithdrawalAmount);
            System.out.println("dateOfWithdrawal: "+dateOfWithdrawal);
        }
        else{
            System.out.println("balanceAmount: "+getbalanceAmount());
        }
    }
}
```

Appendix of CreditCard

```
/**
 * Write a description of class CreditCard here.
 *
 * @author (22067570 Krish Bhattarai)
 * @version (1.0.0)
 */
public class CreditCard extends BankCard
{
    //Declaring the Attributes
    private int cvcNumber;
    private double creditLimit;
    private double interestRate;
    private String expirationDate;
    private int gracePeriod;
    private boolean isGranted;

    //Creating constructor that takes eight parameters
    public CreditCard(int card_id,String clientName,String issuerBank,String bankAccount,int balanceAmount,int cvcNumber, double interestRate, String expirationDate){ //Expirationdate

        //Creating super constructor
        super(card_id, issuerBank, bankAccount, balanceAmount);
        super.setclientName(clientName);

        //Assigning parameter values to the corresponding class
        this.cvcNumber = cvcNumber;
        this.interestRate = interestRate;

        this.expirationDate = expirationDate;


        //Setting the value of isGranted to False

        this.isGranted = false;

    }


    //Providing getter method for cvcNumber

    public int getcvcNumber(){

        return this.cvcNumber;

    }


    //Providing getter method for creditLimit

    public double getcreditLimit(){

        return this.creditLimit;

    }


    //Providing getter method for interestRate

    public double getinterestRate(){

        return this.interestRate;

    }


    //Providing getter method for expirationDate
```

```
public String getexpirationDate(){
    return this.expirationDate;
}


//Providing getter method for gracePeriod
public int getgracePeriod(){
    return this.gracePeriod;
}
//Providing getter method for isGranted
public boolean getisGranted(){
    return this.isGranted;
}


//Creating a method that sets credit limit.
public void setcreditLimit(double creditLimit,int gracePeriod){
    if(creditLimit <= 2.5 * getbalanceAmount()){
        this.creditLimit = creditLimit;
        this.gracePeriod = gracePeriod;
        this.isGranted = true;
    } else {
        System.out.println("Credit can't be issued.");
    }
}
```

```
//Creating method for cancellingCreditCard
public void cancelCreditCard(){
    if(isGranted){
        cvcNumber = 0;
        creditLimit = 0;
        gracePeriod = 0;
        isGranted =false;
    }
}
//Creating display method for details of creditCard
public void display(){

    if(isGranted == true){
        super.display();

        System.out.println("cvcNumber: "+this.cvcNumber);
        System.out.println("creditLimit: "+this.creditLimit);
        System.out.println("interestRate: "+this.interestRate);
        System.out.println("ExpirationDate: "+this.expirationDate);
        System.out.println("gracePeriod: "+this.gracePeriod);
    }else{
        super.display();
```

```
            System.out.println("cvcNumber: "+this.cvcNumber);

            System.out.println("interestRate: "+this.interestRate);

            System.out.println("ExpirationDate: "+this.expirationDate);

            System.out.println("creditLimit: "+this.creditLimit);

            System.out.println("gracePeriod: "+this.gracePeriod);

        }

    }

}
```

Appenix of BankGUI

```
/**

 * Write a description of class CreditCard here.

 *

 * @author (22067570 Krish Bhattarai)

 * @version (1.0.0)

 */

import javax.swing.JFrame;

import javax.swing.*;

import java.awt.event.*;

import java.util.ArrayList;

public class BankGUI implements ActionListener

{

    public JButton clrButton;
    //Private
    private JFrame myFrame;
    private ArrayList<BankCard>lists = new ArrayList<>();

    //Declare
    private JLabel guiLabel, debitLabel, idLabel, nameLabel, issueLabel, bankAcLabel, balAmtLabel, pinLabel, withdrawLabel, withdrawlLabel, withdrawDLabel, withidLabel, creditLabel, idCLabel, nameCLabel, issueCLabel, bankAcCLabel, balAmtCLabel, cvcCLabel, interestCLabel, expCLabel, limitCLabel, limitLabel, graceLabel, cardlLabel, wpinLabel;

    //----DEBITCARD----
    //----JTextfield
    private JTextField idText = new JTextField();
    private JTextField nameText = new JTextField();
    private JTextField issueText = new JTextField();
    private JTextField bankAcText = new JTextField();
    private JTextField balAmtText = new JTextField();
    private JTextField withidText = new JTextField();
    private JTextField wpinText = new JTextField();
    private JTextField pinText = new JTextField();
    private JTextField withAText = new JTextField(); //Withdraw
```

```
private JComboBox dayWComboBox = new JComboBox();

private JComboBox dateWComboBox = new JComboBox();

private JComboBox yearWComboBox = new JComboBox();


//JButton

//JButton aButton = new JButton("ADD");

private JButton displayButton = new JButton("DISPLAY");

private JButton adButton = new JButton("ADD");

private JButton withdrawButton = new JButton("WITHDRAW");


//----CREDITCARD----

//TEXTFIELD


//JTextField

private JTextField idCText = new JTextField();

private JTextField nameCText = new JTextField();

private JTextField issueCText = new JTextField();

private JTextField bankAcCText = new JTextField();

private JTextField balAmtCText = new JTextField();

private JTextField cvcCText = new JTextField();

private JTextField interestCText = new JTextField();

private JTextField limitText = new JTextField();

private JTextField graceText = new JTextField();
private JTextField cardlText = new JTextField();


private JComboBox dayCComboBox = new JComboBox();

private JComboBox dateCComboBox = new JComboBox();

private JComboBox yearCComboBox = new JComboBox();


//JButton

//JButton aButton = new JButton("ADD");

private JButton displayCButton = new JButton("DISPLAY");

private JButton adCButton = new JButton("ADD");

private JButton limitCButton = new JButton("Set Credit Limit");

private JButton cancelCButton = new JButton("Cancel Credit");


//private JTextField

public BankGUI(){

    //Creating a JFrame using a constructor

    JFrame myFrame = new JFrame("BankGUI");


    //DEBIT CARD

    //JLabel

    guiLabel = new JLabel("BankGUI");

    debitLabel = new JLabel("Debit Card");

    idLabel = new JLabel("Card ID");
```

```
nameLabel = new JLabel("Client Name");

issueLabel = new JLabel("Issuer Bank");

bankAcLabel = new JLabel("Bank Account");

balAmtLabel = new JLabel("Balance Amt");

pinLabel = new JLabel("Pin No");

withdrawLabel = new JLabel("Withdraw Amt");

withdrawDLabel = new JLabel("Withdraw Date");

withdrawlLabel = new JLabel("Withdrawal");

withidLabel = new JLabel("Card ID");

wpinLabel = new JLabel ("Pin No");


//Adding Label

myFrame.add(guiLabel);

myFrame.add(debitLabel);

myFrame.add(idLabel);

myFrame.add(nameLabel);

myFrame.add(issueLabel);

myFrame.add(bankAcLabel);

myFrame.add(balAmtLabel);

myFrame.add(pinLabel);

myFrame.add(withdrawLabel);

myFrame.add(withdrawDLabel);

myFrame.add(withdrawlLabel);
myFrame.add(withidLabel);
myFrame.add(wpinLabel);

//Adding Textfield
myFrame.add(idText);
myFrame.add(nameText);
myFrame.add(issueText);
myFrame.add(bankAcText);
myFrame.add(balAmtText);
myFrame.add(withidText);
myFrame.add(pinText);
myFrame.add(dayWComboBox);
myFrame.add(dateWComboBox);
myFrame.add(yearWComboBox);
myFrame.add(withAText);
myFrame.add(wpinText);

//Adding Buttons
myFrame.add(adButton);
myFrame.add(displayButton);
myFrame.add(withdrawButton);
```

```
//Setting Bounds

//Setting Bounds to Label

guiLabel.setBounds(764, 42, 124, 31);

debitLabel.setBounds(182, 106, 91, 25);

idLabel.setBounds(46, 201, 48, 20);

nameLabel.setBounds(46, 261, 77, 20);

issueLabel.setBounds(46, 321, 74, 20);

bankAcLabel.setBounds(422, 195, 86, 20);

balAmtLabel.setBounds(422, 261, 79, 20);

pinLabel.setBounds(422, 321, 42, 20);

withdrawLabel.setBounds(41, 654, 97, 20);

withdrawDLabel.setBounds(372, 654, 104, 20);

withdrawlLabel.setBounds(169, 472, 81, 25);

withidLabel.setBounds(41, 593, 48, 20);

wpinLabel.setBounds(457, 587, 42, 20);


//Setting Bounds to Textfield(DebitCard)

idText.setBounds(174, 195, 140, 32);

nameText.setBounds(174, 255, 140, 32);

issueText.setBounds(174, 315, 140, 32);

bankAcText.setBounds(550, 189, 140, 32);

balAmtText.setBounds(550, 255, 140, 32);

pinText.setBounds(550, 315, 140, 32);
```

```
withAText.setBounds(169, 648, 140, 32); //Withdrawal Date

withidText.setBounds(169, 587, 140, 32);

dayWComboBox.setBounds(499, 649, 48, 32);

dateWComboBox.setBounds(567, 649, 48, 32);

yearWComboBox.setBounds(631, 649, 58, 32);

wpinText.setBounds(549, 581, 140, 32);


//Setting Bounds to Buttons

adButton.setBounds(411, 391, 120, 32);

displayButton.setBounds(569, 391, 120, 32);

withdrawButton.setBounds(179, 719, 120, 32);


//END OF DEBITCARD


//CREDITCARD

//LABEL

//JLabel

creditLabel = new JLabel("Credit Card");

idCLabel = new JLabel("Card ID");

nameCLabel = new JLabel("Client Name");

issueCLabel = new JLabel("Issuer Bank");

bankAcCLabel = new JLabel("Bank Account");

balAmtCLabel = new JLabel("Balance Amt");
```

```
cvcCLabel = new JLabel("CVC no");

interestCLabel = new JLabel("Interest Rate");

expCLabel = new JLabel("Expiration Date");

limitCLabel = new JLabel("Credit Limit");

limitLabel = new JLabel("Credit Limit");

graceLabel = new JLabel("Grace Period");

cardlLabel = new JLabel("Card ID");


//ADDING

//Adding Label

myFrame.add(creditLabel);

myFrame.add(idCLabel);

myFrame.add(nameCLabel);

myFrame.add(issueCLabel);

myFrame.add(bankAcCLabel);

myFrame.add(balAmtCLabel);

myFrame.add(cvcCLabel);

myFrame.add(interestCLabel);

myFrame.add(expCLabel);

myFrame.add(limitCLabel);

myFrame.add(limitLabel);

myFrame.add(graceLabel);

myFrame.add(cardlLabel);
```

```
//Adding TextField

myFrame.add(idCText);

myFrame.add(nameCText);

myFrame.add(issueCText);

myFrame.add(bankAcCText);

myFrame.add(balAmtCText);

myFrame.add(cvcCText);

myFrame.add(interestCText);

myFrame.add(dayCComboBox);

myFrame.add(dateCComboBox);

myFrame.add(yearCComboBox);

myFrame.add(limitText);

myFrame.add(graceText);

myFrame.add(cardlText);


//Adding Buttons

myFrame.add(adCButton);

myFrame.add(displayCButton);

myFrame.add(limitCButton);

myFrame.add(cancelCButton);


//SETTING BOUNDS
```

```
//Setting Bounds to Label

creditLabel.setBounds(1010, 106, 98, 25);

idCLabel.setBounds(879, 201, 48, 20);

nameCLabel.setBounds(879, 261, 77, 20);

issueCLabel.setBounds(879, 321, 74, 20);

bankAcCLabel.setBounds(879, 381, 86, 20);

balAmtCLabel.setBounds(1251, 201, 79, 20);

cvcCLabel.setBounds(1251, 256, 44, 26);

interestCLabel.setBounds(1251, 325, 80, 19);

expCLabel.setBounds(1251, 378, 96, 20);

limitCLabel.setBounds(996, 491, 100, 25);

limitLabel.setBounds(1219, 578, 71, 20);

graceLabel.setBounds(1219, 638, 82, 20);

cardlLabel.setBounds(912, 598, 48, 20);


//Setting Bounds to TextFields

idCText.setBounds(1007, 195, 140, 32);

nameCText.setBounds(1007, 255, 140, 32);

issueCText.setBounds(1007, 315, 140, 32);

bankAcCText.setBounds(1007, 375, 140, 32);

balAmtCText.setBounds(1379,195, 140, 32);

cvcCText.setBounds(1379, 256, 140, 32);

interestCText.setBounds(1379, 314, 140, 32);
```

```
dayCComboBox.setBounds(1379, 372, 48, 32);
dateCComboBox.setBounds(1447, 372, 48, 32);
yearCComboBox.setBounds(1511, 372, 58, 32);
limitText.setBounds(1347, 572, 140, 32);
graceText.setBounds(1347, 632, 140, 32);
cardlText.setBounds(1000, 598, 140, 32);

//Setting Bounds to Buttons
adCButton.setBounds(1286, 459, 120, 32);
displayCButton.setBounds(1434, 459, 120, 32);
limitCButton.setBounds(976, 743, 120, 32);
cancelCButton.setBounds(1126, 743, 120, 32);
//END OF CREDITCARD

//*********
//Submit and clear button
//JButton submitButton = new JButton("Submit");
clrButton = new JButton("Clear");

//Connect event listener to all source
clrButton.addActionListener(this);

//Add submit and clear button
```

```
//myFrame.add(submitButton);

myFrame.add(clrButton);


//setting Bounds to submit and clear button

//submitButton.setBounds(1434, 743, 120, 30);

clrButton.setBounds(1298, 743, 120, 30);

//combobox EXPIRATION DATE

for (int i = 1; i <=31; i++){

    dayCComboBox.addItem(i);

}

for (int j = 1; j <=12; j++){

    dateCComboBox.addItem(j);

}

for (int k = 2019; k <=2023; k++){

    yearCComboBox.addItem(k);

}


//combobox WITHDRAWL DATE

for (int i = 1; i <=31; i++){

    dayWComboBox.addItem(i);

}

for (int j = 1; j <=12; j++){

    dateWComboBox.addItem(j);
```

```
        }
        for (int k = 2019; k <=2023; k++){
            yearWComboBox.addItem(k);
        }

        //CREATING ARRAYLIST
        ArrayList BankCard = new ArrayList();

        //ADD ACTIONLISTENER
        adButton.addActionListener(this);
        adCButton.addActionListener(this);
        displayButton.addActionListener(this);
        displayCButton.addActionListener(this);
        limitCButton.addActionListener(this);
        cancelCButton.addActionListener(this);
        withdrawButton.addActionListener(this);

        myFrame.setSize(1600, 838);
        myFrame.setLayout(null);
        myFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        //myFrame.setResizable(false);
        myFrame.setVisible(true);
    }
```

```
public void actionPerformed(ActionEvent e){
    //CLEAR
    if (e.getSource() == clrButton){
        idText.setText("");
        nameText.setText("");
        issueText.setText("");
        bankAcText.setText("");
        balAmtText.setText("");
        withidText.setText("");
        pinText.setText("");
        withAText.setText("");
        idCText.setText("");
        nameCText.setText("");
        issueCText.setText("");
        bankAcCText.setText("");
        balAmtCText.setText("");
        cvcCText.setText("");
        interestCText.setText("");
        limitText.setText("");
        graceText.setText("");
        cardlText.setText("");
    }
```

```
//DEBITCARD
//DEBITCARD DISPLAY
if(e.getSource() == displayButton){
    for(BankCard obj: lists){
        if(!(obj instanceof DebitCard)){
            continue;
        }
        obj.display();
    }
}

//DebitCard ADD
if(e.getSource() == adButton){
    try{
        //verifying if any text field is empty or not
        int card_id = Integer.parseInt(idText.getText());
        String clientName = nameText.getText();
        String issuerBank = issueText.getText();
        String bankAccount = bankAcText.getText();
        int balanceAmount = Integer.parseInt(balAmtText.getText());
        int pinNumber = Integer.parseInt(pinText.getText());
        DebitCard debit_card = new DebitCard(balanceAmount, card_id, bankAccount,
issuerBank, clientName, pinNumber);

        boolean verify =false;

        for (BankCard obj: lists){

            if(!(obj instanceof DebitCard)){

                verify = true;

                continue;

            }

            if(((DebitCard)obj).getcard_id() == card_id){

                JOptionPane.showMessageDialog(null, "Card ID is already present.");

                verify = false;

                break;

            }

            else{

                verify = true;

            }

        }

        if((lists.isEmpty()) || (verify)){

            lists.add(debit_card);

            JOptionPane.showMessageDialog(null, "Debit Card has been added.");

        }

    }

    catch(NumberFormatException ex){
```

```
                JOptionPane.showMessageDialog(null, "Inaccurate Data.");
            }
        }

        //WITHDRAW
        if(e.getSource() == withdrawButton){
            try{
                boolean verify = false;
                int card_id = Integer.parseInt(withidText.getText());
                String date = dateWComboBox.getSelectedItem().toString();
                String day = dayWComboBox.getSelectedItem().toString();
                String year = yearWComboBox.getSelectedItem().toString();
                String calender = date+ "/" +day+ "/" +year;
                int balance_amount = Integer.parseInt(withAText.getText()); //balanceAmount,
pinNumber j rakda ni huncha
                int pin_number = Integer.parseInt(wpinText.getText());
                for (BankCard obj: lists){
                    if (!(obj instanceof DebitCard)) {
                        continue;
                    }
                    if (obj.getcard_id() == card_id){
                        verify = true;
                        int draw = ((DebitCard)obj).withdraw(balance_amount, date,
pin_number);
                        if(draw == 1){
                            JOptionPane.showMessageDialog(null, "Amount has been
successfully withdrawn.");
                        }
                        else if(draw == 2){
                            JOptionPane.showMessageDialog(null, "Incorrect Pin Number");
                        }
                        else if(draw == 3){
                            JOptionPane.showMessageDialog(null, "Insufficient Balance.");
                        }
                    }
                }
                if(lists.isEmpty()){
                    JOptionPane.showMessageDialog(null, "ERROR!! Please confirm if Debit
Card is Present.");
                }
                if (! verify) {
                    JOptionPane.showMessageDialog(null, "Could not find the Card ID.");
                }
            }
            catch(NumberFormatException ex){
                JOptionPane.showMessageDialog(null, "ERROR!! Verify if the data inserted is
correct");
            }
        }

        //CREDITCARD
        //CREDITCARD DISPLAY
        if(e.getSource() == displayCButton){
            for(BankCard obj: lists){
                if(!(obj instanceof CreditCard)){
                    continue;
                }
                obj.display();
            }
        }

        //ADD CREDITCARD
        if(e.getSource() == adCButton){
            try{
                int card_id = Integer.parseInt(idCText.getText());
                String clientName = nameCText.getText();
                String issuerBank = issueCText.getText();
                String bankAccount = bankAcCText.getText();
```

```
int balanceAmount = Integer.parseInt(balAmtCText.getText());
int cvcNumber = Integer.parseInt(cvcCText.getText());
double interestRate = Double.parseDouble(interestCText.getText());

String day = dayCComboBox.getSelectedItem().toString();
String date = dateCComboBox.getSelectedItem().toString();
String year = yearCComboBox.getSelectedItem().toString();

String expirationDate = day+ "/" +date+ "/" +year;
CreditCard credit_card = new CreditCard(card_id, clientName, issuerBank,
bankAccount, balanceAmount, cvcNumber, interestRate, expirationDate);
boolean verify =false;
for (BankCard obj: lists){
   if(!(obj instanceof CreditCard)){
      verify = true;
      continue;
   }
   if(((CreditCard)obj).getcard_id() == card_id){
      JOptionPane.showMessageDialog(null, "Card ID is already present.");
      verify = false;
      break;
   }
   else{
```

```
                    verify = true;
                }
            }
            if((lists.isEmpty()) || (verify)){
                lists.add(credit_card);
                JOptionPane.showMessageDialog(null, "Successfully added the
CreditCard.");
            }
        }
        catch(NumberFormatException ex){
            JOptionPane.showMessageDialog(null, "Invalid Data.");
        }
    }

    //CREDIT LIMIT BUTTON
    if(e.getSource() == limitCButton){
        try{
            boolean verify = false; //change verify variable name

            int card_id = Integer.parseInt(cardlText.getText());

            int creditLimit = Integer.parseInt(limitText.getText());
            int gracePeriod = Integer.parseInt(graceText.getText());
            for (BankCard obj: lists){
                if(!(obj instanceof CreditCard)){
                    continue;
                }
                if(((CreditCard)obj).getcard_id() == card_id){
                    verify = true;
                    ((CreditCard)obj).setcreditLimit(creditLimit, gracePeriod);
                    JOptionPane.showMessageDialog(null, "Credit Limit has been
successfully set.");
                }
            }

            if(lists.isEmpty()){
                JOptionPane.showMessageDialog(null, "Empty Creditcard.");
            }

            if(! verify){
                JOptionPane.showMessageDialog(null, "Could not find Card ID");
            }
        }
        catch(NumberFormatException ex){
            JOptionPane.showMessageDialog(null, "Incorrect data");
        }
        //public static void main(String []args){

        //BankGUI obj = new BankGUI();

        //public static void main(String[] args){

        //BankGUI obj = new BankGUI();

    }


    //CANCEL CREDIT BUTTON

    if(e.getSource() == cancelCButton){

        try{

            boolean verify = false;

            int card_id = Integer.parseInt(idCText.getText());

            for(BankCard obj: lists){

                if (!(obj instanceof CreditCard)) {

                    continue;

                }

                if (obj.getcard_id() == card_id){

                    verify = true;

                    ((CreditCard)obj).cancelCreditCard();


                    JOptionPane.showMessageDialog(null, "Credit has been Cancelled");

                }

            }
```

```
                if (! verify) {
                    JOptionPane.showMessageDialog(null, "Could not find the Card ID.");
                }
            }
            catch(NumberFormatException ex){
                JOptionPane.showMessageDialog(null, "Input the Card ID.");
            }
        }
    }

    public static void main(String[] args){
        BankGUI obj = new BankGUI();
    }
}
```