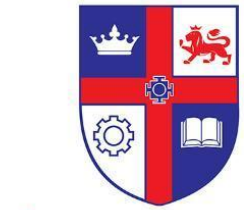
LONDON METROPOLITAN UNIVERSITY



Islington college
(इस्लिङ्टन कलेज)

CS4001NI Programming

30% Individual Coursework

2022-23 Autumn

Student Name: Angana Bhattarai

London Met ID: 22067110

College ID: NP01NT4A220074

Group: N6

Assignment Due Date: Friday, July 28, 2023

Assignment Submission Date: Thursday, July 27, 2023

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

Table of Figures

Table of tables

1. Introduction

1.1. Tools Used

Java is a powerful general-purpose programming language created in 1995 by Sun Microsystems. Java was initially modified in early times to take advantage of the development in the sector of World Wide Web. In 2009, Oracle Corporation acquired Sun Microsystems and took ownership of Java and many other assets. Java is a multi-platform, object-oriented, and network-centric language, which is fast, secure, and reliable. Java code can run on all platforms that support Java without requiring it to be recompiled.

BlueJ is an integrated development environment (IDE) for the JAVA programming language, designed for academic purposes. It is also suitable for small scale software development. It runs with the help of Java Development Kit (JDK). BlueJ's design is unique from other development environments because it was created to help the teaching and learning of object-oriented programming. An application's class structure is graphically displayed on the main screen, and objects can be built and tested in real time. This interactivity facility, when combined with a clear, uncomplicated user interface, makes it straightforward to experiment with developing objects. Visual representations of object-oriented concepts (classes, objects, and communication through method calls) are used in the interface's interaction design.

Draw.io is a free diagram online software for making flowcharts, process diagrams, org charts, UML, ER and network diagrams. It is a diagram editor. They contain a big section of shapes and hundreds of graphic components that may be utilized to make diagrams and charts. The drag-and-drop capability makes creating a professional-looking diagram or chart a breeze. We can make flowcharts, UML diagrams, entity relationship diagrams, network diagrams, mock ups, and more.

Microsoft Word is a popular commercial word processor from Microsoft. It was first released in 1983 and has seen various revisions since then. It’s compatible with both Windows and Mac computers. MS-Word has been so popular in the last two decades because of the ability of users to copy and paste text from one platform to another without major formatting loss.

1.2. About Coursework.

This coursework expects us to make a Graphical User Interface (GUI) where it is imported and extended from the BankCard, DebitCard and CreditCard and then add functionalities within it for the GUI aspects to work. We make class diagrams, Pseudocode, describe the methods used within the program, do a few testing over the result of the program and then check for error detection and rectify them later.

The constructors called in the BankCard, DebitCard and the CreditCard are the elements processed in the BankGUI. In the making of GUI, all the constructors and parameters are called and carried out. JLabel, JTextFields, JComboBox, JButtons, etc are used in this BankGUI. And after the completion of the project, when the GUI is fully formed, when the input information is entered, we get the output accordingly. In this GUI, we have the information of a client with their bank details depending on whether the client wants to withdraw cash or deposit amount. The process in carried by adding the client's name along with the amount, the name of the bank, the balance amount in the particular bank account with the date of withdrawal or deposit.The BankGUI stored as array list of the type BankCard class to hold DebitCard and CreditCard. The following report is discussed respectively.

2. Class Diagram

The links between classes and their attributes, constructors, and methods are shown in a class diagram. On the other hand, a class diagram provides no information regarding the implementation of the constructors and methods. It does this by presenting properties, classes, functions, and relationships to give a general picture of the software system. It organizes class names, traits, and methods into separate sections to facilitate program development. Given that it contains classes, interfaces, affiliations, collaborations, and constraints, it is known as a structural diagram.

Three rows make up the class diagram. The name of the appropriate class appears in the first row. All instance variables are described in the second row. When describing instance variables in the second row, the "+" symbol is used for public access modifiers and the "-" sign for private access modifiers.

Next, the return type and the name of the instance variable are stated. If a method has a public access modifier, the "+" sign is used in the third row's method descriptions, and if a method has a private access modifier, the "-" sign is used. The name of the method and the return type are then stated.

2.1. Class Diagram for Bank GUI

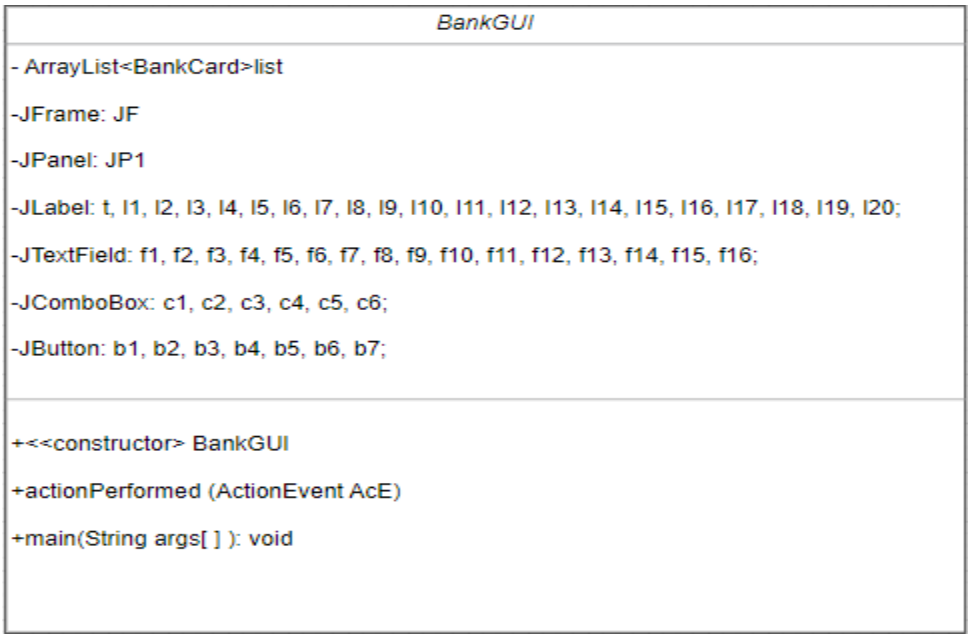


Figure 1: Class Diagram of BankGUI

2.2. Class Diagram for Bank Card Class

Bank Card
-ClientName:String -IssuerBank:String -CardId:int -BalanceAmount:double -BankAccount:String
+<<Constructor>>BankCard(balanceamount.:double,bankaccount:String,cardId:int,issuerbank:String) +getClientName():String +getIssuerBank():String +getCardId():int +getBalanceAmount():double +getBankAccount():String +setClientName(CientName:String):void +setBalanceAmount(balanceamount:double):void +display():void

Figure 2: Class diagram of BankCard

2.3. Class Diagram for Debit Card

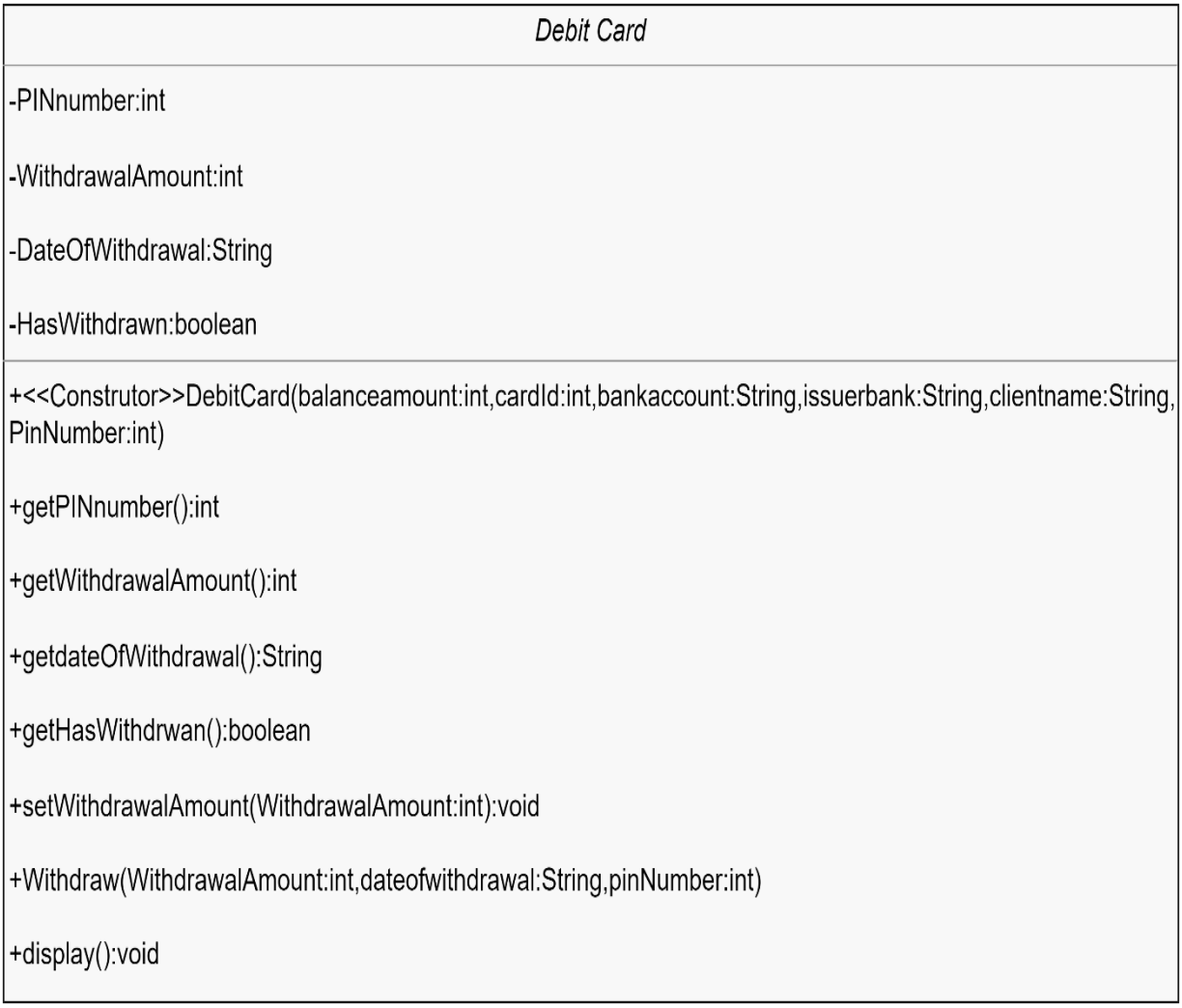


Figure 3: Class Diagram of DebitCard

2.4. Class Diagram for Credit Card

Figure 4: Class diagram of Creditcard

2.5. Combine Class Diagram of the Project

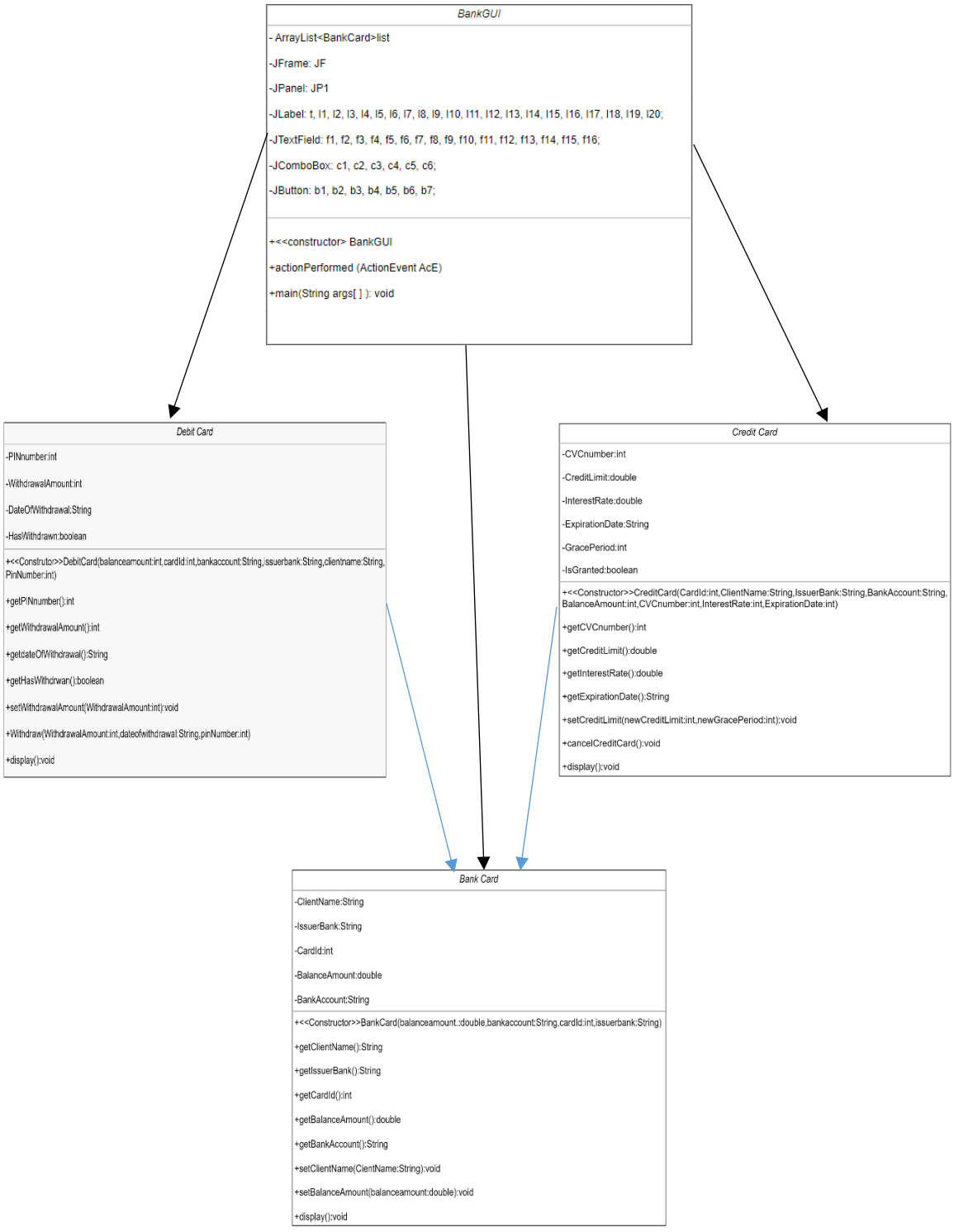


Figure 5: Combined class diagram of BankGUI

2.6. Diagram of The Project in BlueJ and GUI

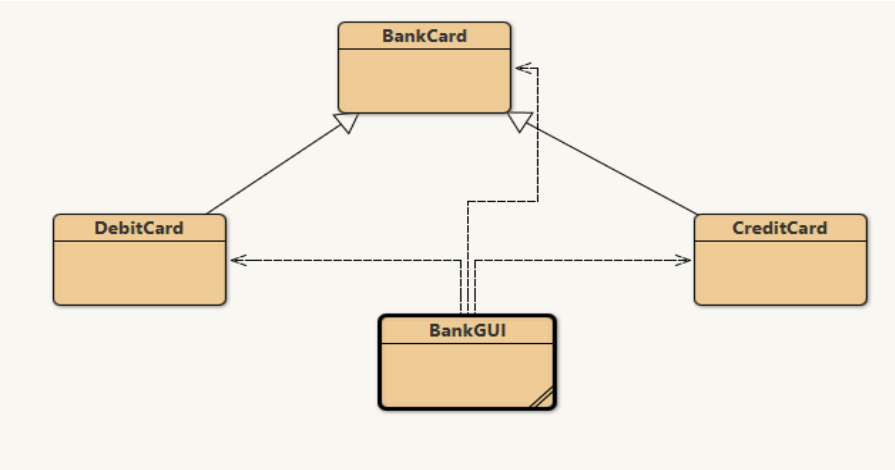


Figure 6: Diagram of BankGUI in BlueJ

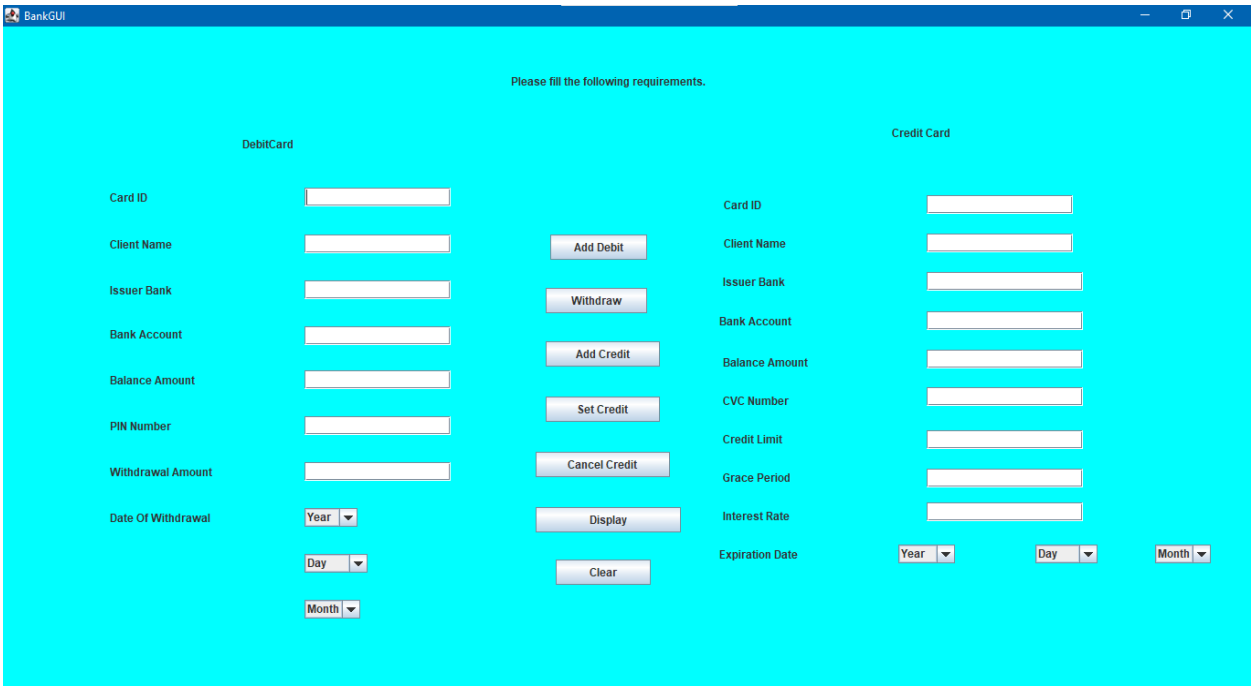


Figure 7: Output Of the GUI

3. Pseudocode

Programmers can construct algorithms artificially and formally using pseudocode. It is a non-technical way of describing programming without using strict syntaxes or other technical issues. It provides a level of comprehension for a program's flow.

By making the code more readable, even for non-technical individuals, programmers may identify and align the code in accordance with the requirements of a software project. It is not, however, a true programming language. Additionally, it offers a basic documentation that makes it easy for you to comprehend how the product is developed. Pseudocode is crucial in this context for describing precisely what each line of the program does. Documentation is a crucial component that explains what has been done, what may be altered, and what should not be done.

The guidelines that had to be adhered to when creating the pseudocode for any of the programs are listed below:

1. First words are bolded and capitalized.

2. By-lines for statements are printed and are not mixed up.

3. A block of code is contained in the keywords DO and ENDDO.

4. Multiline forms have END keywords at the end of their suffix.

5. Use simple, straightforward vocabulary to keep it readable and brief.

6. Correct handling of indentation and separation is required.

3.1. Pseudocode of BankGUI.

IMPORT javax.swing.*;

IMPORT java.awt.event.*;

IMPORT java.awt.*;

IMPORT java.util.ArrayList;

CREATE a class BankGUI which implements ActionListener

DO

DECLARE JFrame, JLabel, JTextField, JComboBox, JButton, JPanel

GENERATE Array List of BankGUI

GENERATE a constructor BankGUI

DO

GENERATE labels and text fields.

SET t label to Please fill the following requirements.

SET l1 label to DebitCard

SET l2 label to Card ID

SET f1 text field for Card ID

SET l3 label to Bank Acoount

SET f2 text field to Bank Account

SET l4 label to Client Name

SET f3 text field to Client Name

SET l5 label to PIN Number

SET f4 text field to PIN Number

SET l6 label to Issuer Bank

SET f5 text field to Issuer Bank

SET l7 label to Balance Amount

SET f6 text field to Balance Amount

SET l8 label to Withdrawal Amount

SET f7 text field to Withdrawal Amount

SET l9 label to Date Of Withdrawal

SET String Year

SET String Day

SET String Month

SET l10 label to Credit Card

SET l11 label to Card ID

SET f8 text field to Card ID

SET l12 label to Client Name

SET f9 text field to Client Name

SET l13 label to Balance Amount

SET f10 text field to Balance Amount

SET l14 label to Credit Limit

SET f11 text field to Credit Limit

SET l15 label to Grace Period

SET f12 text field to Grace Period

SET l16 label to Bank Account

SET f13 text field to Bank Account

SET l17 label to Issuer Bank

SET f14 text field to Issuer Bank

SET l18 label to CVC Number

SET f15 text field to CVC

SET l19 label to Interest Rate

SET f16 text field to Interest Rate

SET l20 label to Expiration Date

SET String Year

SET String Day

SET String month

SET c1 combo box to Year

SET c2 combo box to Day

SET c3 combo box to Month

SET c4 combo box to Year

SET c5 combo box to Day

SET c6 combo box to Month

SET b1 button to Add Debit

SET b2 button to Withdraw

SET b3 button to Add Credit

SET b4 button to Set Credit

SET b5 button to Cancel Credit

SET b6 button to Display

SET b7 button to Clear

ADD labels, text fields, buttons, combo box to the Frame

SET bounds for all labels, text fields, buttons, combo box

ADD ActionListener to all buttons

ADD JPanel to the JFrame

SET Layout as null to JFrame

SET JFrame as visible to true

SET Size of JFrame

SET Layout as null to JPanel

SET Bounds to JPanel

SET Background with colour to JPanel

SET Default Close Operation

END DO

PROCEDURE actionPerformed AcE

DO

IF AcE is equal to b1

DO

IF text fields are Empty **THEN**

DO

DISPLAY the error message

END DO

ELSE

DO

TRY

DO

SET f1 text field value to CardId

SET f2 text field value to BankAccount

SET f3 text field value to ClientName

SET f4 text field value to PinNumber

SET f5 text field value to IssuerBank

SET f6 text field value to BalanceAmount

IF boolean apa condition is true

FOR each x in BankCard list

DO

IF x is an instance of DebitCard

DO

SET x object as DebitCard

IF CardId is equal to object getCardId

DO

HERE the condition refers to apa is equal to false;

END DO

END DO

END DO

IF apa is equal to true

DO

SET DebitCard asap is equal to new object DebitCard

ADD asap to list

DISPLAY information message

END DO

ELSE

DO

DISPLAY Error Message

END DO

END DO

CATCH Number Format Exception

DO

DISPLAY Error Message.

END DO

END DO

END DO

ELSE

IF AcE is equal to b6

DO

FOR each x in list of BankCard

DO

IF x is an instance of DebitCard

DO

SET x as object on DebitCard

DISPLAY object

END DO

ELSE IF x is an instance of CreditCard

DO

SET x as object on CreditCard

DISPLAY object

END DO

END DO

END DO

ELSE IF AcE is equal to b7

DO

SET texts to all the text fields

REMOVE all items for clear in all combo box

END DO

ELSE IF AcE is equal to b3

DO

IF text fields are empty **THEN**

DO

DISPLAY Error Message

END DO

ELSE

DO

TRY

```
DO
  SET f8 text field as CardID
  SET f9 text field as ClientName
  SET f10 text field as BalanceAmount
  SET f13 text field as BankAccount
  SET f14 text field as IssuerBank
  SET f15 text field as CVCnumber
  SET f16 text field as InterestRate
  SET c4 combo box as Year
  SET c5 combo box as Month
  SET c6 combo box as Day
  SET label ExpirationDate is equal to Year and Month and Day

  IF boolean apa is equal to true

  FOR x in each list of BankCard
  DO
    IF x is an instance of CreditCard
    DO
      SET object as CreditCard(x)
      IF x is an instance of CreditCard
      DO
        IF apa is equal to false
      END DO
    END DO
  END DO
  IF apa is equal to true
   DO
```

DECLARE object asap in CreditCard

ADD asap in list

DISPLAY Information Message

END DO

ELSE

DO

DISPLAY Error Message

END DO

END DO

CATCH Number Format Exception

DO

DISPLAY Error Message

END DO

END DO

END DO

ELSE IF AcE is equal to b2

DO

IF text field is Empty

DO

DISPLAY Error Message

END DO

ELSE

DO

TRY

DO

SET f1 text field as CardID

SET f4 text field as PINnumber

SET f7 text field as Withdraw

SET c1 combo box as Year

SET c2 combo box as Month

SET c3 combo box as Day

SET label ExpirationDate is equal to year and month and day

IF boolean psp is equal to false

FOR each x in list of BankCard

DO

IF (x instanceof DebitCard)

DO

SET object as DebitCard(x)

IF x is an instance of DebitCard

DO

IF psp is equal to true;

SET PINnumber object is equal to PINnumber

DO

IF Withdraw is less than Balance Amount

DO

DECLARE object as Withdraw

DISPLAY Information Message

END DO

END DO

END DO

ELSE

DO

IF psp is equal to false;

END DO

END DO

END DO

IF psp is equal to false

DO

DISPLAY Error Message

END DO

END DO

CATCH Number Format Exception

DO

DISPLAY Error Message

END DO

END DO

END DO

ELSE IF AcE is equal to b4

DO

IF the text field is Empty

DO

DISPLAY Error Message

END DO

ELSE

DO

TRY

DO

SET f8 text field as CardId

SET f11 text field as CreditLimit

SET f12 text field as GracePeriod

IF boolean ppt is equal to false

FOR each x in BankCard list

DO

IF x is an instance of CreditCard

DO

SET object as CreditCard(x)

IF x is an instance of CreditCard

DO

IF ppt is true

SET object to CreditLimit

DISPLAY Information Message

DO

ELSE

DO

WHERE ppt is false

END DO

END DO

END DO

IF ppt is false

DO

DISPLAY Error Message

END DO

END DO

CATCH Number Format Exception

DO

DISPLAY Error Message

END DO

END DO

END DO

ELSE IF AcE is equal to b5

DO

IF the text field is Empty

DO

DISPLAY Error Message

END DO

ELSE

DO

TRY

DO

DECALRE CardId as Integer

SET f8 as int text field

WHERE Boolean condition psp is false;

FOR each x in list of BankCard

DO

IF x is an instance of CreditCard

DO

SET x object as CreditCard

IF x is an instance of CreditCard

DO

WHERE psp is true

SET object as CancelCredit

DISPLAY Information Message

END DO

ELSE

DO

WHERE psp is false

END DO

END DO

END DO

IF psp is false

DO

DISPLAY Error Message

END DO

END DO

CATCH Number Format Exception

DO

DISPLAY Error Message

END DO

END DO

END DO

END DO

CREATE main method

DO

CREATE object from class BankGUI

END DO

END DO

4. Method Description

The following table gives a short description of the methods used in this project.

• actionPerformed():	Using this technique, user activities like button clicks are handled. The Buttons are: • Add Debit(b1): When this button is pressed, a new object is created using the values of the CardId, ClientName, BankAccount, IssuerBank, BalanceAmount, and PINnumber. If the required text fields are empty then an error dialog box message will appear asking the client to not leave the text fields empty. If all the required text fields are filled, it enters the 'else' block to process the new debit card. • Withdraw(b2): The CardId input value is compared to the existing CardId when this button is pushed, and if a valid CardId has been supplied, it is used to withdraw money from the Debit Card. With the 'else' and 'if else' conditions, we thoroughly work on the characters necessary. Presented with the ActionEvent, and the given 'else' and 'else if' conditions, 'CardId', 'PINNumber' and 'WithdrawalAmount' text fields and the three combo boxes of Year, Month, Day for Withdrawal Amount, must be filled. If failed to enter, the values will not be further executed. If the appropriate values are entered, the dialog box with 'Withdrawn Successfully' message with appear with the trigger of 'JOptionPane.showMessageDailog' code. • Add Credit(b3): When this button is selected, a new object identical to a Debit Card is created using the input values for the Card, ClientName, IssuerBank, BankAccount, BalanceAmount, CVCnumber, InterestRate, and ExpirationDate. The conditions required are to fill all the give text fields with

	appropriate values. If the text fields are left empty, you will receive a message asking you to not leave the text fields empty. With the 'NumberFormatException' the non-integers values will not be processed. If the text fields are entered with suitable values, the object will be created for the credit card with all the information given. Later, you will receive a dialog box where it refers that the credit card is added successfully. • Set Credit(b4): The CardId input value is compared to the existing CardId when this button is pushed, and if a valid CardId has been supplied, it is used to adjust the Credit Limit of the Credit Card.The 'else if' block used in this button's code it to handle a specific event from a component in the GUI. getSource() checkes whether the text fields are used with proper input values. If any of the three necessary text fields are empty i.e. 'CreditLimit', 'CardId' and 'GracePeriod' , the "JOptionPane.showMessageDialog" code is used to display an error message. With the 'else', it triggers the code under that only after the three text fields required are filled. With the 'NumberFormatException' the non-integer values entered will be denied. • Cancel Credit(b5): The CardId input value is compared to the existing CardId when this button is pushed, and if a valid CardId has been entered, the Credit Card is cancelled. When the button is clicked, the AcE.getSource() == b5 has an event handling block situation which triggers the code when the "Cancel Credit" button is clicked in the GUI. The code then reviews if the text field is filled with the appropriate value or not. If not, the Error message dialog box appears. This button also holds Number Format Exception, which is executed if the text field is entered with an invalid input value i.e. non-integer.

	• Display(b6): When this button is touched, the appropriate class's information is displayed. When the ActionEvent comes from this b6 button, it enterest a loop that that iterates through each element in the list. It reviews the object is the instance of the particular class. Then the ‘display()’ method is called which prints the information of the required details according to the values entered. The loop continues until all the elements in the list are processed. • Clear(b7): The values from text fields are removed when this button is clicked. The code sets the text of multiple text fields to empty strings and removed all the items from the selected dropdown menu from the combo box and resets back to as it was.

Table 1: Method Description Table.

5. Testing

5.1. Test 1

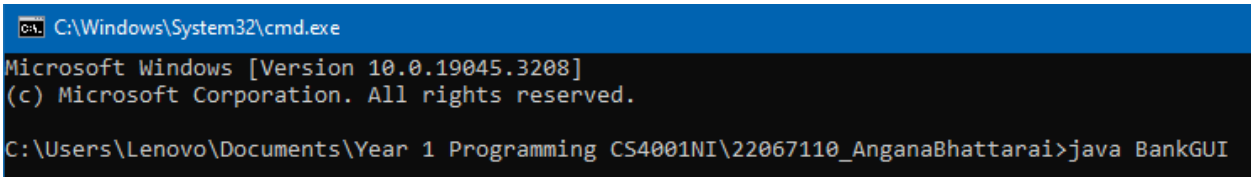


Figure 8: Command prompt for test 1

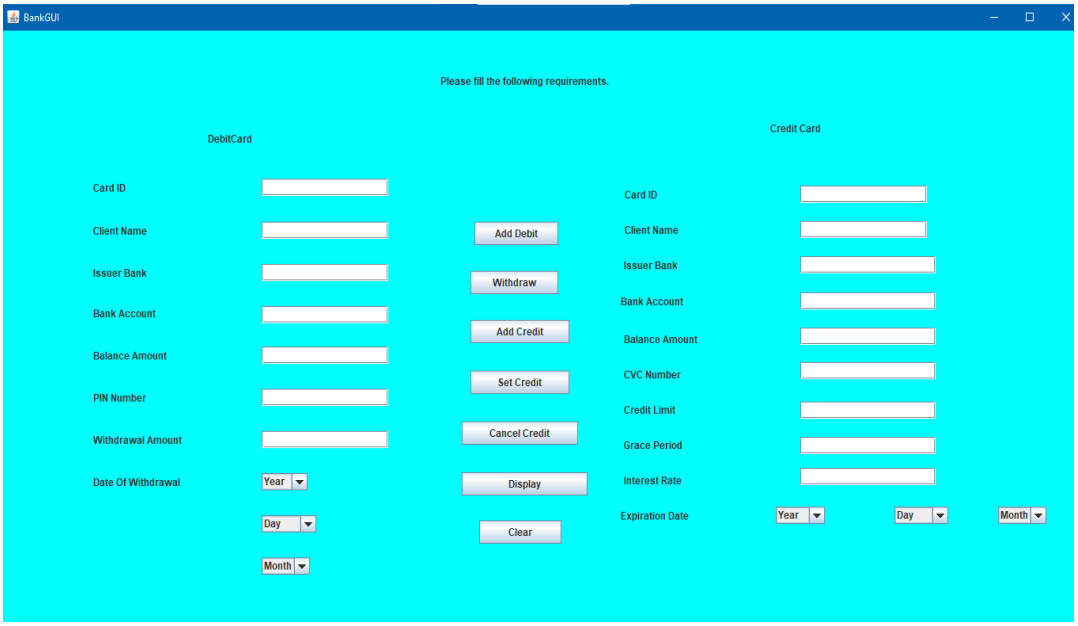


Figure 9: Output for test 1

• Objective	To run GUI through command prompt.
• Action	Go to the folder's location, and enter cmd in the path directory. When the command prompt appears, type 'java BankGUI'.
• Expected Result	The compiled java file of BankGUI should appear.
• Actual Result	BankGUI compiled java file is on display.
• Conclusion	The test is successful.

Table 2 Test 1: Analysis Table

5.2. Test 2

5.2.1. Add Debit Card

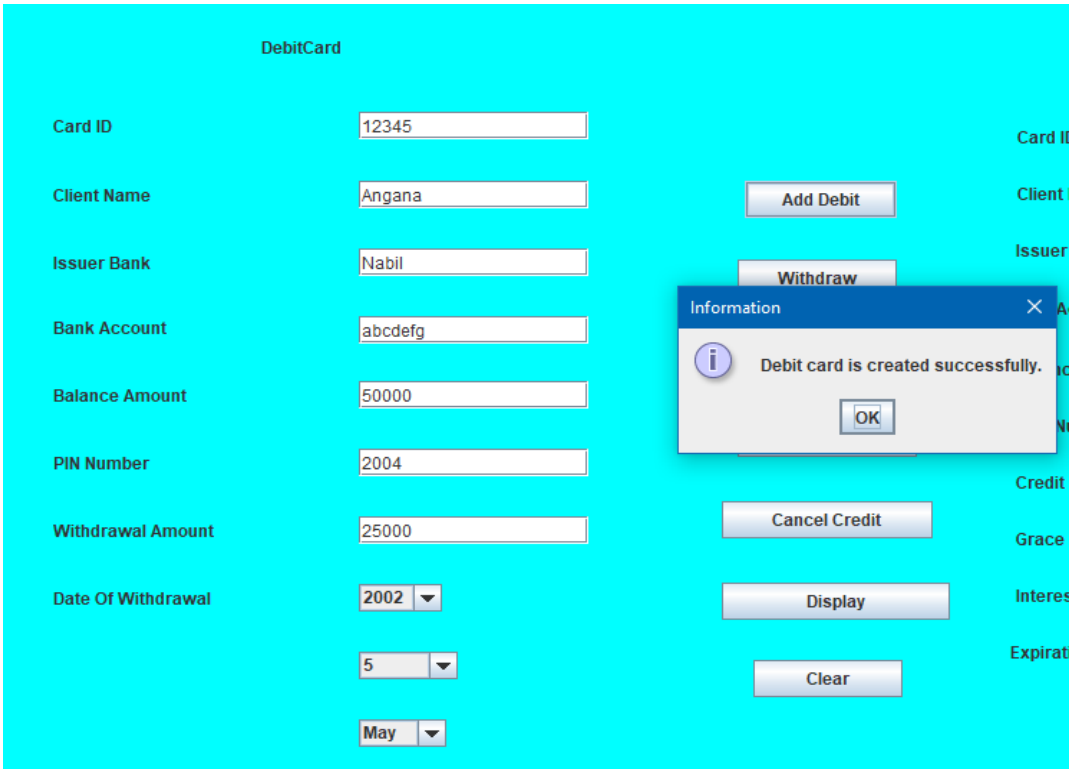


Figure 10: Adding Debit Card

• Objective	To add Debit Card
• Action	The values are entered in the Debit Card text fields and the Add Debit button is clicked.
• Expected Result	Dialog box should appear with “ Debit Card is created successfully”.
• Actual Result	Dialog box appeared with “ Debit card is created successfully” message.
• Conclusion	Test is successful.

Table 3 Test 2: Analysis Table

5.2.2. Add Credit Card

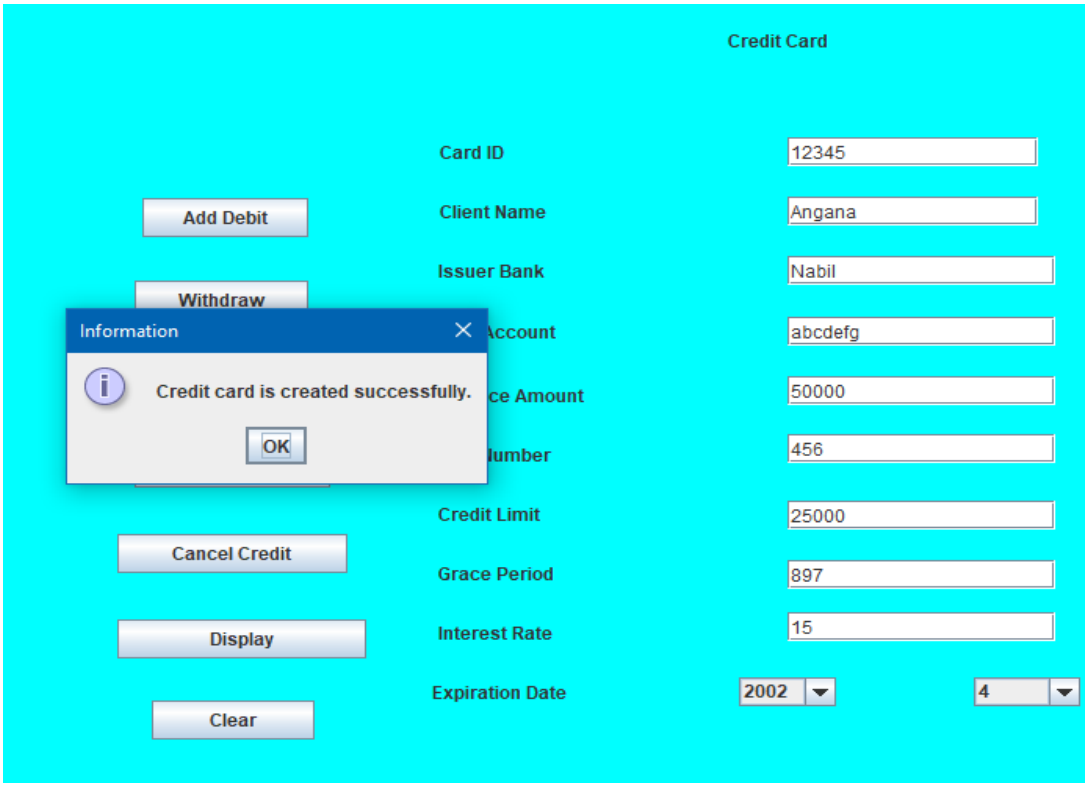


Figure 11: Adding Credit Card

• Objective	To add Credit Card
• Action	The values are entered in the Credit Card text fields and then the Add Credit button is clicked.
• Expected Result	The dialog box should appear with the “Credit Card is successfully created” message.
• Actual Result	The dialog box appeared with the “Credit Card is successfully created” message.
• Conclusion	The test is successful.

Table 4: Test 3: Analysis Table

5.2.3. Withdraw Amount from Debit Card

Figure 12: Withdraw amount from Debit Card

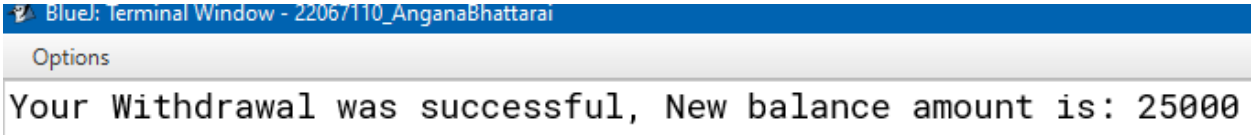


Figure 13: Display of successful message.

• Objective	To withdraw amount from Debit Card
• Action	The values are entered in the Debit Card text fields and then the Withdraw button is clicked.
• Expected Result	The dialog box should appear with the “Withdrawn successfully” message.
• Actual Result	The dialog box appeared with the “Withdrawn Successfully” message.
• Conclusion	The test is successful.

Table 5: Test 4: Analysis Table

5.2.4. Set Credit Limit

Figure 14: Setting Credit Limit

• Objective	To set Credit Limit.
• Action	The values are entered in the Credit Card text fields and the Set Credit Limit button is clicked.
• Expected Result	The dialog box should appear with the message “Credit card has been set.”
• Actual Result	The dialog box appeared with the particular message.
• Conclusion	The test is successful.

Table 6: Test 5: Analysis Table

5.2.5. Remove Credit Limit

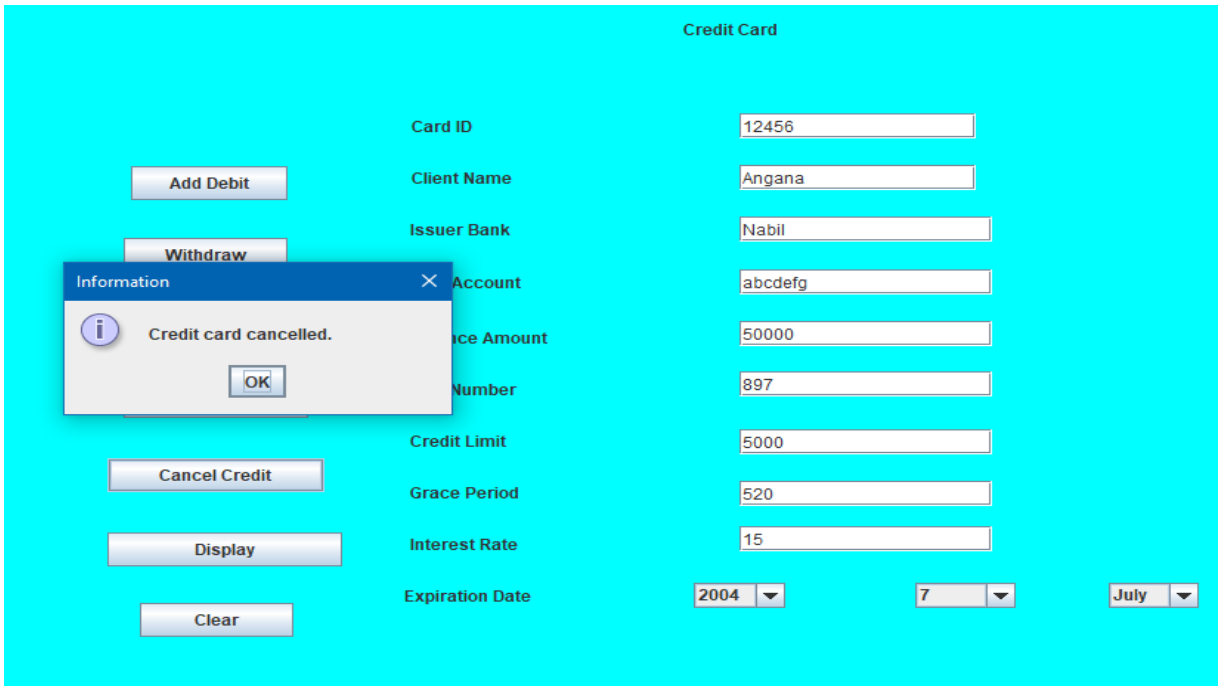


Figure 15: Removing Credit Limit

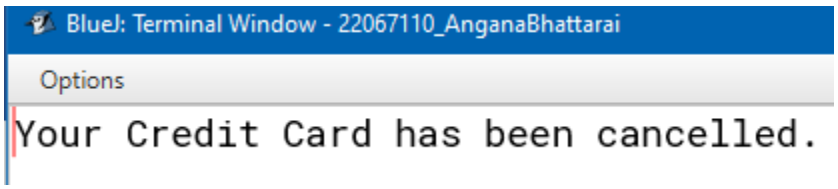


Figure 16: Display of message

Objective	To remove Credit Card
Action	The values are entered in the text fields of Credit Card and the Cancel Credit button is clicked.
Expected Result	The dialog box should appear with the message, “Credit Card Cancelled”.
Actual Result	The dialog box appeared with the particular message.
Conclusion	The test is successful.

Table 7: Test 6: Analysis Table

5.3. Test 3: Card ID

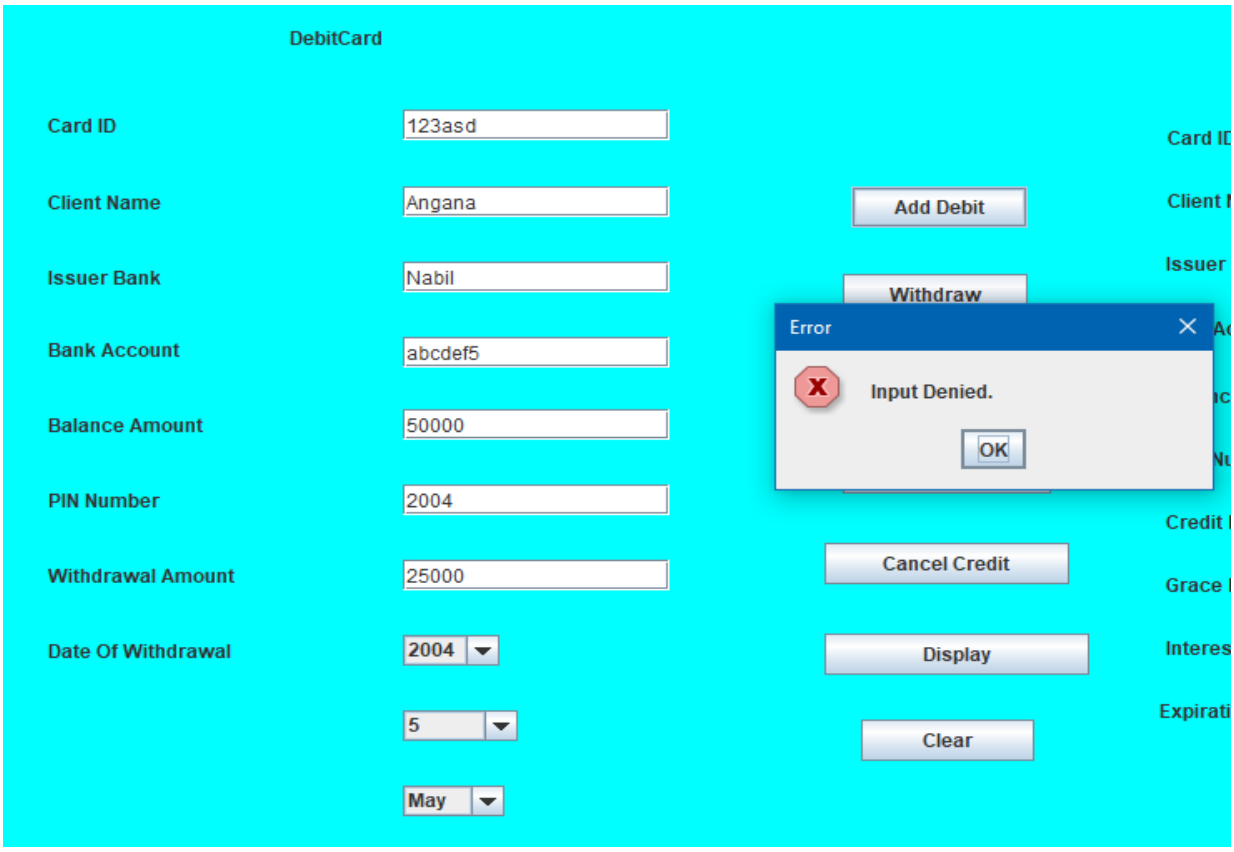


Figure 17: Card ID Test

Objective	To appear the appropriate dialog boxes when unsuitable data are entered.
Action	The value entered in the Card ID was supposed to be numbers as it is called under integer data type but instead double data type was placed.
Expected Result	The dialog box should appear with the message, “Input Denied”.
Actual Result	The dialog box appeared with the “Input Denied” message.
Conclusion	The test is successful.

Table 8: Test 7: Analysis Table

6. Error Detection and Correction

6.1. Syntax Error

Error detection for syntax error.

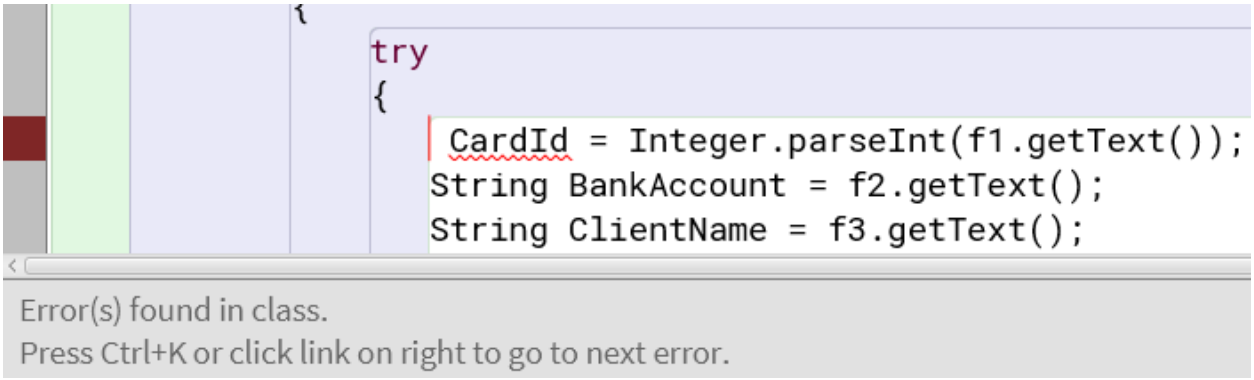


Figure 18: Error detection for syntax error

In the above figure, we can see that the ‘int’ datatype is missing. That is why the error is shown. This is an example of Syntax error.

Error correction for syntax error.

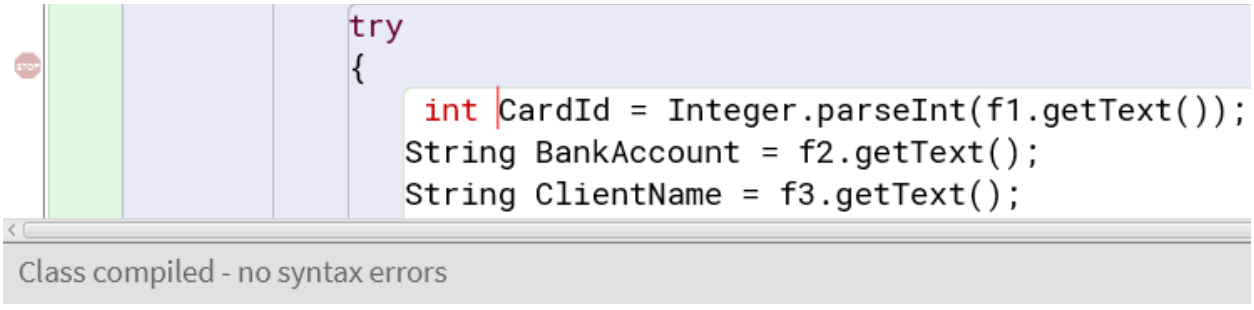


Figure 19: Error correction for syntax error

Now the missing datatype is used, thus we don’t have any error.

6.2. Semantic Error

Error detection for Semantic Error

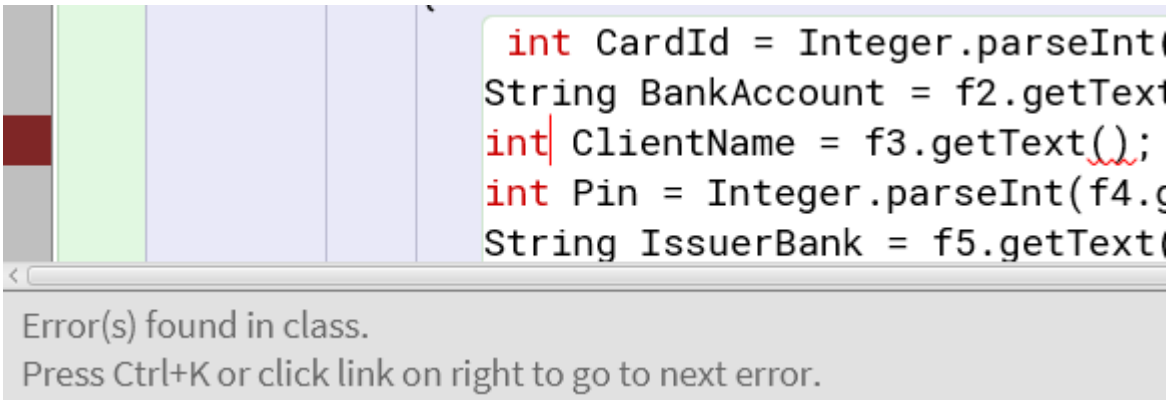


Figure 20: Error detection for semantic error

In the figure above, the syntax datatype is given but an error is still shown in the program. This is an example of semantic error. Here, the datatype should be 'String' not 'int'.

Error correction for Semantic Error

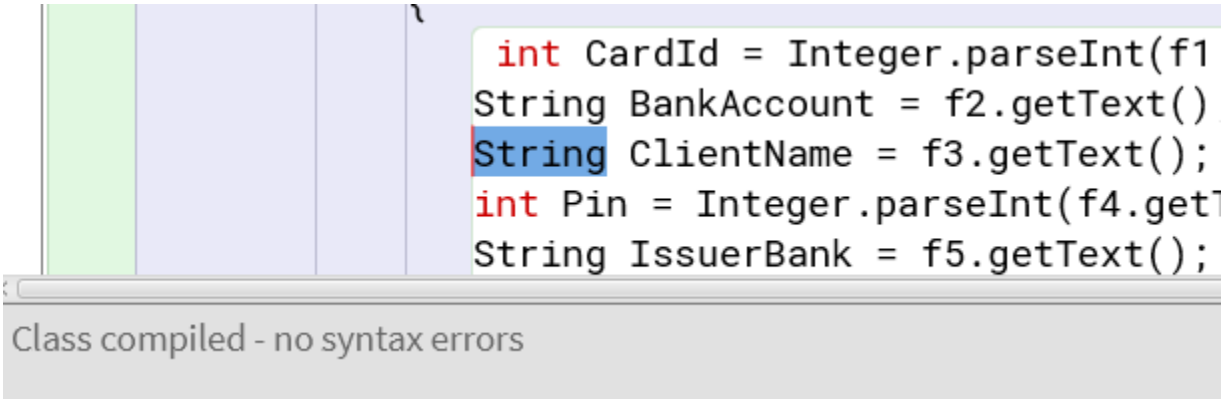


Figure 21: Error correction for semantic error

In the above figure, as we can see the datatype is corrected from 'int' to 'String'. Thus, the error is no longer shown.

6.3. Logical Error

Error detection for logical error.

```
boolean apa = false;

for(BankCard x : list)
{
    if(x instanceof CreditCard)
    {
        CreditCard object= (CreditCard) x;
        if( CardId== object.getCardId())
        {
            apa= false;
        }
    }
}
```

Class compiled - no syntax errors

Figure 22: Error detection for logical error.

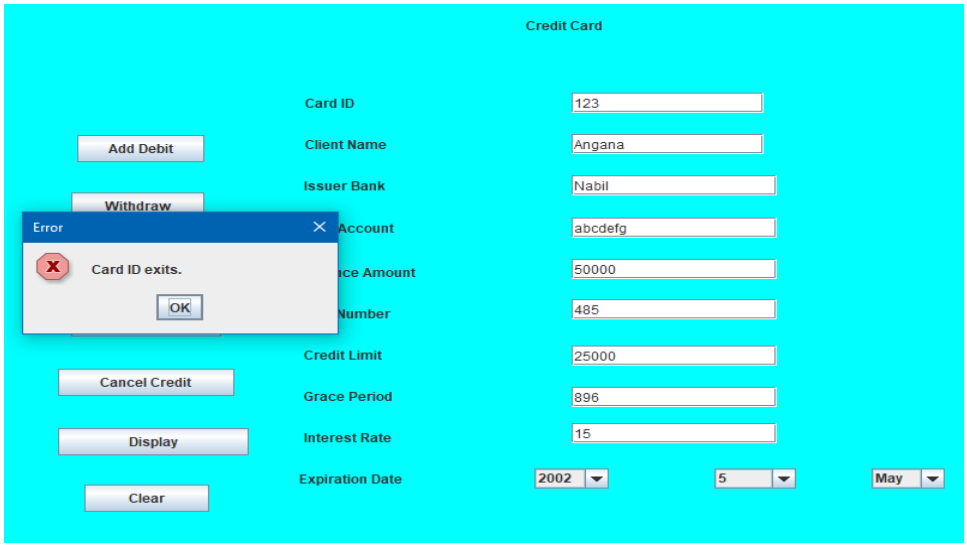


Figure 23: Output of error detection for logical error

Here, the condition was provided as false which led the input details to not process successfully. Thus, the error was shown.

Error correction for logical error.

```
boolean apa = true;

for(BankCard x : list)
{
    if(x instanceof CreditCard)
    {
        CreditCard object= (CreditCard) x;
        if( CardId== object.getCardId())
        {
            apa= false;
        }
    }
```

Class compiled - no syntax errors

Figure 24: Error correction for logical error

Figure 25: Output of error correction for logical error

Here, the condition was changed to true which executed with the expected successful output. Thus, the error was corrected.

7. Conclusion

A programming language that helps in the creation of programs is called Java. Java allows us to create classes by adhering to the identifiers rule. Java programs are created using a program compilation tool called BlueJ. Since java teaches us how to handle failures, I learned the language while working with Blue. I discovered class diagrams in this coursework, which are intended to illustrate static structure. Additionally, I learned how to draw class diagrams, which let us see a class's attributes and methods. In a similar way, I made inheritance diagrams using these class diagrams.

During this project, I created a GUI where all different methods and functions in Java are set as input. I learned about the new methods to use and their purpose in Java program. I also tried creating Pseudocode for this project code.

I discovered numerous types of errors in this program as I was constructing it. Different errors that I was previously unaware of happened, including logical, semantic, and grammatical issues. I did some research on it before figuring out how to troubleshoot and fix these mistakes. In order to correct mistakes and construct this application, I used a variety of sources and assistance from my tutors.

8. Appendix

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
import java.util.ArrayList;

public class BankGUI implements ActionListener
{
    JFrame JF;
    JPanel JP1;
    JLabel t,l1,l2,l3,l4,l5,l6,l7,l8,l9,l10,l11,l12,l13,l14,l15,l16,l17,l18,l19,l20;
    JTextField f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16;
    JComboBox c1,c2,c3,c4,c5,c6;
    JButton b1,b2,b3,b4,b5,b6,b7;
    ArrayList<BankCard> list  = new ArrayList<BankCard>();
     public BankGUI()
    {
        JF = new JFrame("BankGUI");
        JP1 = new JPanel();
        t = new JLabel("Please fill the following requirements.");
        l1 = new JLabel("DebitCard");
        l2 = new JLabel("Card ID");
        f1 = new JTextField();
        l3 = new JLabel("Bank Account");
        f2 = new JTextField();
        l4 = new JLabel("Client Name");
        f3 = new JTextField();
```

```
        l5 = new JLabel("PIN Number");
        f4 = new JTextField();
        l6 = new JLabel("Issuer Bank");
        f5 = new JTextField();
        l7 = new JLabel("Balance Amount");
        f6 = new JTextField();
        l8 = new JLabel("Withdrawal Amount");
        f7 = new JTextField();
        l9 = new JLabel("Date Of Withdrawal");
        String[] year = {"Year","1999","1992","1993"};
        String[] Day =
{"Day","1","2","3","4","5","6","7","9","10","11","12","13","14","15","16","17","18","19","20","
21","22","23","24","25","26","27","29","30","31"};
        String[] Month
={"Month","Jan","Feb","Mar","Apr","May","Jun","July","Aug","Sep","Oc","Nov","Dec"};
        l10 =new JLabel("Credit Card");
        l11 =new JLabel("Card ID");
        f8 =new JTextField();
        l12 =new JLabel("Client Name");
        f9 =new JTextField();
        l13 =new JLabel("Balance Amount");
        f10 =new JTextField();
        l14 =new JLabel("Credit Limit");
        f11 =new JTextField();
        l15 =new JLabel("Grace Period");
        f12 =new JTextField();
        l16 =new JLabel("Bank Account");
        f13 =new JTextField();
        l17 =new JLabel("Issuer Bank");
```

```
        f14 =new JTextField();
        l18 =new JLabel("CVC Number");
        f15 =new JTextField();
        l19 =new JLabel("Interest Rate");
        f16 =new JTextField();
        l20 =new JLabel("Expiration Date");
        String[] Year =
{"Year","2001","2002","2003","2004","2005","2006","2007","2008","2009","2010"};
        String[] day =
{"Day","1","2","3","4","5","7","9","10","11","12","13","14","15","16","17","18","19","20","21"
,"22","23","24","25","26","27","29","30","31"};
        String[] month
={"Month","Jan","Feb","Mar","Apr","May","Jun","July","Aug","Sep","Oct","Nov","Dec"};

        c1 =new JComboBox(Year);
        c2 =new JComboBox(Day);
        c3 =new JComboBox(Month);
        c4 =new JComboBox(Year);
        c5 =new JComboBox(Day);
        c6 =new JComboBox(Month);

        b1 =new JButton("Add Debit");
        b2 =new JButton("Withdraw");
        b3 =new JButton("Add Credit");
        b4 =new JButton("Set Credit");
        b5 =new JButton("Cancel Credit");
        b6 =new JButton("Display");
        b7 =new JButton("Clear");

        JF.add(t);
```

```
JF.add(l1);
JF.add(l2);
JF.add(f1);
JF.add(l3);
JF.add(f2);
JF.add(l4);
JF.add(f3);
JF.add(l5);
JF.add(f4);
JF.add(l6);
JF.add(f5);
JF.add(l7);
JF.add(f6);
JF.add(l8);
JF.add(f7);
JF.add(l9);
JF.add(l10);
JF.add(l11);
JF.add(f8);
JF.add(l12);
JF.add(f9);
JF.add(l13);
JF.add(f10);
JF.add(l14);
JF.add(f11);
JF.add(l15);
JF.add(f12);
JF.add(l16);
```

```
JF.add(f13);
JF.add(l17);
JF.add(f14);
JF.add(l18);
JF.add(f15);
JF.add(l19);
JF.add(f16);
JF.add(l20);


JF.add(c1);
JF.add(c2);
JF.add(c3);
JF.add(c4);
JF.add(c5);
JF.add(c6);

JF.add(b1);
JF.add(b2);
JF.add(b3);
JF.add(b4);
JF.add(b5);
JF.add(b6);
JF.add(b7);

t.setBounds(556,41,312,35);
l1.setBounds(262,115,84,20);
l2.setBounds(117,172,94,20);
```

```
f1.setBounds(330,172,160,20);
l3.setBounds(117,318,109,20);
f2.setBounds(330,320,160,20);
l4.setBounds(117,222,109,20);
f3.setBounds(330,222,160,20);
l5.setBounds(117,416,80,20);
f4.setBounds(330,416,160,20);
l6.setBounds(117,271,109,20);
f5.setBounds(330,271,160,20);
l7.setBounds(117,367,109,20);
f6.setBounds(330,367,160,20);
l8.setBounds(117,465,125,20);
f7.setBounds(330,465,160,20);
l9.setBounds(117,514,121,20);
l10.setBounds(973,103,75,20);
l11.setBounds(789,180,109,20);
f8.setBounds(1011,180,160,20);
l12.setBounds(789,221,109,20);
f9.setBounds(1011,221,160,20);
l13.setBounds(788,348,109,20);
f10.setBounds(1011,345,171,20);
l14.setBounds(788,430,108,20);
f11.setBounds(1011,431,171,20);
l15.setBounds(788,471,108,20);
f12.setBounds(1011,472,171,20);
l16.setBounds(784,304,109,20);
f13.setBounds(1011,304,171,20);
l17.setBounds(788,262,109,20);
```

```
f14.setBounds(1011,262,171,20);
l18.setBounds(788,389,108,20);
f15.setBounds(1011,385,171,20);
l19.setBounds(788,512,108,20);
f16.setBounds(1011,508,171,20);
l20.setBounds(784,553,108,20);



c1.setBounds(330,514,58,20);
c2.setBounds(330,563,69,20);
c3.setBounds(330,612,61,20);
c4.setBounds(980,553,62,20);
c5.setBounds(1130,553,68,20);
c6.setBounds(1261,553,61,20);

b1.setBounds(599,222,106,27);
b2.setBounds(594,279,111,27);
b3.setBounds(594,336,125,27);
b4.setBounds(594,395,125,27);
b5.setBounds(583,454,147,27);
b6.setBounds(583,513,159,27);
b7.setBounds(605,569,104,27);

b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
```

```
        b5.addActionListener(this);
        b6.addActionListener(this);
        b7.addActionListener(this);

        JF.add(JP1);
        JF.setLayout(null);
        JF.setVisible(true);
        JF.setSize(1400,1020);

        JP1.setLayout(null);
        JP1.setBounds(0,0,1400,1020);
        JP1.setBackground(Color.CYAN);

        JF.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
    public void actionPerformed(ActionEvent AcE)
    {
    if(AcE.getSource()==b1)
        {
            if (f1.getText().isEmpty() || f2.getText().isEmpty() || f3.getText().isEmpty() ||
f4.getText().isEmpty() || f5.getText().isEmpty() || f6.getText().isEmpty() )
            {
                JOptionPane.showMessageDialog(JF,"Please don't leave any text
field.","Error",JOptionPane.ERROR_MESSAGE);

            }
            else
            {
                try
```

```
{
    int CardId = Integer.parseInt(f1.getText());
   String BankAccount = f2.getText();
   String ClientName = f3.getText();
   int Pin = Integer.parseInt(f4.getText());
   String IssuerBank = f5.getText();
   int BalanceAmount = Integer.parseInt(f6.getText());
   boolean apa= true;

   for(BankCard x : list)
   {
      if(x instanceof DebitCard)
      {
         DebitCard object= (DebitCard) x;
         if( CardId== object.getCardId())
         {
             apa= false;
         }
      }
   }
   if(apa== true)
    {
        DebitCard asap= new
DebitCard(BalanceAmount,CardId,BankAccount,IssuerBank,ClientName,Pin);
        list.add( asap);
        JOptionPane.showMessageDialog(JF,"Debit card is created
successfully.","Information",JOptionPane.INFORMATION_MESSAGE);
    }
    else
```

```
                {
                    JOptionPane.showMessageDialog(JF,"Card ID
exits.","Error",JOptionPane.ERROR_MESSAGE);
                }
            }catch(NumberFormatException n)
            {
                JOptionPane.showMessageDialog(JF,"Input
Denied.","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    }
    else if(AcE.getSource()==b6)
    {
        for(BankCard x : list)
        {
                if(x instanceof DebitCard)
                {
                    DebitCard object= (DebitCard) x;
                    object.display();
                 }
                 else if(x instanceof CreditCard)
                {
                    CreditCard object= (CreditCard) x;
                    object.display();
                }
        }
    }
    else if(AcE.getSource()==b7)
    {
```

```
            f1.setText("");
            f2.setText("");
            f3.setText("");
            f4.setText("");
            f5.setText("");
            f6.setText("");
            f7.setText("");
            f8.setText("");
            f9.setText("");
            f10.setText("");
            f11.setText("");
            f12.setText("");
            f13.setText("");
            f14.setText("");
            f15.setText("");
            f16.setText("");
            c1.removeAllItems();
            c2.removeAllItems();
            c3.removeAllItems();
            c4.removeAllItems();
            c5.removeAllItems();
            c6.removeAllItems();
        }
        else if(AcE.getSource()==b3)
       {
            if (f8.getText().isEmpty() || f9.getText().isEmpty() || f10.getText().isEmpty() ||
f13.getText().isEmpty() || f14.getText().isEmpty() || f15.getText().isEmpty() ||
f16.getText().isEmpty() )
            {
```

```
            JOptionPane.showMessageDialog(JF,"Please don't leave any text
field.","Error",JOptionPane.ERROR_MESSAGE);


        }
        else
        {
            try
            {
                int CardId = Integer.parseInt(f8.getText());
                String ClientName = f9.getText();
                int BalanceAmount = Integer.parseInt(f10.getText());
                String BankAccount = f13.getText();
                String IssuerBank = f14.getText();
                int PINnumber = Integer.parseInt(f15.getText());
                int CVCnumber = Integer.parseInt(f16.getText());
                double InterestRate = Double.parseDouble(f16.getText());
                String Year =(String)c4.getSelectedItem();
                String Month =(String)c5.getSelectedItem();
                String Day =(String)c6.getSelectedItem();
                String ExpirationDate= Year+ " " + Month+ " " + Day;

                boolean apa = true;

                for(BankCard x : list)
                {
                    if(x instanceof CreditCard)
                    {
                        CreditCard object= (CreditCard) x;
                        if( CardId== object.getCardId())
```

```
                    {
                        apa= false;
                    }
                  }
                }
                if(apa== true)
                {
                    CreditCard asap = new
CreditCard(CardId,ClientName,IssuerBank,BankAccount,BalanceAmount,CVCnumber,I
nterestRate,ExpirationDate);
                    list.add( asap);
                    JOptionPane.showMessageDialog(JF,"Credit card is created
successfully.","Information",JOptionPane.INFORMATION_MESSAGE);
                }
                else
                {
                    JOptionPane.showMessageDialog(JF,"Card ID
exits.","Error",JOptionPane.ERROR_MESSAGE);
                }
            }catch(NumberFormatException n)
            {
                JOptionPane.showMessageDialog(JF,"Input
Denied.","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    }
    else if(AcE.getSource()==b2)
    {
        if (f1.getText().isEmpty() || f4.getText().isEmpty()|| f7.getText().isEmpty() )
        {
```

```
            JOptionPane.showMessageDialog(JF,"Please don't leave any text
field.","Error",JOptionPane.ERROR_MESSAGE);
        }
        else
        {
           try
           {
           int CardId = Integer.parseInt(f1.getText());
           int PINnumber = Integer.parseInt(f4.getText());
           int Withdraw = Integer.parseInt(f7.getText());
           String Year =(String)c1.getSelectedItem();
           String Month =(String)c2.getSelectedItem();
           String Day =(String)c3.getSelectedItem();
           String ExpirationDate= Year+ " " + Month+ " " +Day;
           boolean psp = false;
             for(BankCard x : list)
           {
                if(x instanceof DebitCard)
                {
                   DebitCard object= (DebitCard) x;
                   if (CardId==object.getCardId())
                   {
                      psp=true;
                      if(PINnumber==object.getPINnumber())
                      {
                         if (Withdraw<=object.getBalanceAmount())
                         {
                            object.Withdraw(Withdraw,ExpirationDate,PINnumber);
```

```
                                JOptionPane.showMessageDialog(JF,"Withdrawn
Successfully.","Information",JOptionPane.INFORMATION_MESSAGE);
                            }
                        }
                    }
                    else
                    {
                        psp= false;
                    }
                }
            }
             if (psp==false)
            {
              JOptionPane.showMessageDialog(JF,"Card ID not
found.","Error",JOptionPane.ERROR_MESSAGE);
            }
            }catch(NumberFormatException n)
            {
               JOptionPane.showMessageDialog(JF,"Input
Denied.","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    }
    else if(AcE.getSource()==b4)
    {
      if (f8.getText().isEmpty() ||f11.getText().isEmpty() ||f12.getText().isEmpty() )
        {
            JOptionPane.showMessageDialog(JF,"Please don't leave any text
field.","Error",JOptionPane.ERROR_MESSAGE);
        }
```

```
        else
        {
            try
            {
                int CardId = Integer.parseInt(f8.getText());
                int CreditLimit = Integer.parseInt(f11.getText());
                int GracePeriod = Integer.parseInt(f12.getText());
                boolean ppt= false;
                for(BankCard x : list)
                {
                    if(x instanceof CreditCard)
                    {
                        CreditCard object= (CreditCard) x;
                        if( CardId==object.getCardId())
                        {
                            ppt= true;
                            object.setCreditLimit( CreditLimit, GracePeriod);
                            JOptionPane.showMessageDialog(JF,"Credit card has been
set.","Information",JOptionPane.INFORMATION_MESSAGE);
                        }
                        else
                        {
                            ppt= false;
                        }
                    }
                }
                if ( ppt= false)
                {
```

```
                    JOptionPane.showMessageDialog(JF,"Card ID Not
Found","Error",JOptionPane.ERROR_MESSAGE);
                }
            }catch(NumberFormatException n)
            {
                JOptionPane.showMessageDialog(JF,"Input
Denied","Error",JOptionPane.ERROR_MESSAGE);
            }
        }
    }
    else if(AcE.getSource()==b5)
    {
        if (f8.getText().isEmpty())
        {
            JOptionPane.showMessageDialog(JF,"Please don't leave any text
field.","Error",JOptionPane.ERROR_MESSAGE);
        }
        else
        {
            try
            {
                int CardId = Integer.parseInt(f8.getText());
                boolean psp= false;
                for(BankCard x : list)
                {
                    if(x instanceof CreditCard)
                    {
                        CreditCard object= (CreditCard) x;
                        if( CardId== object.getCardId())
```

```
                    {
                        psp= true;
                        object.cancelCreditCard();
                        JOptionPane.showMessageDialog(JF,"Credit card
cancelled.","Information",JOptionPane.INFORMATION_MESSAGE);
                    }
                    else
                    {
                       psp= false;
                    }
                  }
               }
               if ( psp= false)
               {
                   JOptionPane.showMessageDialog(JF,"Card ID Not
Found0","Error",JOptionPane.ERROR_MESSAGE);
               }
            }catch(NumberFormatException n)
            {
                JOptionPane.showMessageDialog(JF,"Input
Denied","Error",JOptionPane.ERROR_MESSAGE);
            }
         }
      }
   }
    public static void main(String [] args)
   {
        new BankGUI();
   } }
```