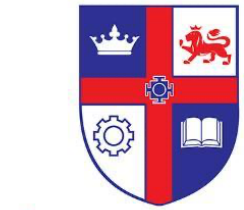
LONDON METROPOLITAN UNIVERSITY



Islington college
(इस्लिङ्टन कलेज)

CS4001NI Programming

30% Individual Coursework

2022-23 Autumn

Student Name: Rounak Pradhan

London Met ID: 22066975

College ID: NP01NT4A220198

Group: N6

Assignment Due Date: Wednesday, May 10, 2023

Assignment Submission Date: Wednesday, May 10, 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

1. Introduction 1

1.1. About the coursework: 1

1.2. Tools used in project: 1

2. Class Diagram: 2

2.1. Bankcard: 3

2.2. Debitcard: 4

2.3. Creditcard: 5

2.4. BankGUI: 5

2.5. Parent, child and BankGUI class diagram: 6

3. Pseudocode: 6

4. Method description: 15

5. Testing 17

5.1. Test1: 17

5.2. Test 2 18

5.3. Test 3 19

5.4. Test 4 20

5.5. Test 5 21

5.6. Test 6 22

5.7. Test 7 23

6. Error detection 25

6.1. Syntax Error 25

6.2. Semantic Error 26

6.3. Logical Error 26

7. Conclusion: 28

8. References 28

9. Appendix 29

Table of figures

Figure 1: Blue j 2
Figure 2: Ms-word 2
Figure 3: Draw.io 2
Figure 4: Class diagram in Blue j 3
Figure 5: Class diagram of Bankcard 3
Figure 6: Class diagram of Debitcard 4
Figure 7: Class diagram of Creditcard 5
Figure 8: Class diagram of BankGUI 5
Figure 9: Parent, child and BankGUI 6
Figure 10: run BankGUI 17
Figure 11: Test 1 18
Figure 12: Test 2 19
Figure 13: Test 3 20
Figure 14: Test 4 21
Figure 15: Test 5 22
Figure 16: Test 6 23
Figure 17: Test 7 24
Figure 18: Test 7.1 25
Figure 19: Syntax error 1.1 25
Figure 20: Syntax error 1.2 26
Figure 21: Semantic error 1.1 26
Figure 22: Semantic error 1.2 26
Figure 23: Logical erorr 1.1 27
Figure 24: Logical error 1.2 27

List of tables

Table 1: Test 1 17
Table 2: Test 2 18
Table 3: Test 3 19
Table 4: Test 4 20
Table 5: Test 5 21
Table 6: Test 6 22
Table 7: Test 7 23

1. Introduction

1.1. About the coursework:

The goal of this project is to create a graphical user interface(GUI) for a banking application. Customers will be able to manage their debit and credit cards, examine their account information, and make various banking operations using the GUI. An Array List of Bankcard objects will be used to hold the customer data, such as card details, account balances, and transaction history.
The GUI will have text boxes and buttons that will allow users to add new debit and credit cars, withdraw money from their accounts, set credit limits, and cancel credit cards. The program will also display relevant card information such as balances and interest rates, as well as a simple and easy-to-use interface with which users may interact.

It aims to provide a stable and secure banking system that satisfies the expectations of customers in an ever-changing financial landscape. I will utilize java's features and application capable of handling a big volume of transactions and data. I will guarantee that the application is extensively tested and optimized for performance, ensuring a smooth and trouble-free customer experience.

1.2. Tools used in project:

To complete this project many tools are used some of them are listed below:

- Blue j
- Ms-word
- Draw.io

Blue j:

A blue j java program is one that was created using the Blue j development environment and written in the java programming language. The Blue j program is free and easy to use. The program can created by creating a new project and then adding one or more classes to it. These classes can have methods and fields that determine the behaviour and data of the program. After defining the classes, they may be created and their methods to called carry out the program's functionally (java, 2020).

Figure 1: Blue j (Rouse, Techopedia, 2013)

Ms-word:

Another important tool for this curriculum is Ms-word. It is a component of the Microsoft office product line and is used for the document creation, editing and formatting (Rouse, 2020).

Figure 2: Ms-word (Rouse, Techopedia, 2022)

Draw.io:

A cross-platform graph drawing called Draw.io was produced using HTML and JavaScript. You can utilize interface to generate flowcharts, wireframes, and UML diagrams (Computer Hope, 2020).

Figure 3: Draw.io (Computer Hope, 2020)

2. Class Diagram:

Class diagram illustrate the internal organization of a system, compared to the static organization or behaviours of an object-oriented system. Support for design elements in class diagram. They specify various classes, linkages, and interactions.

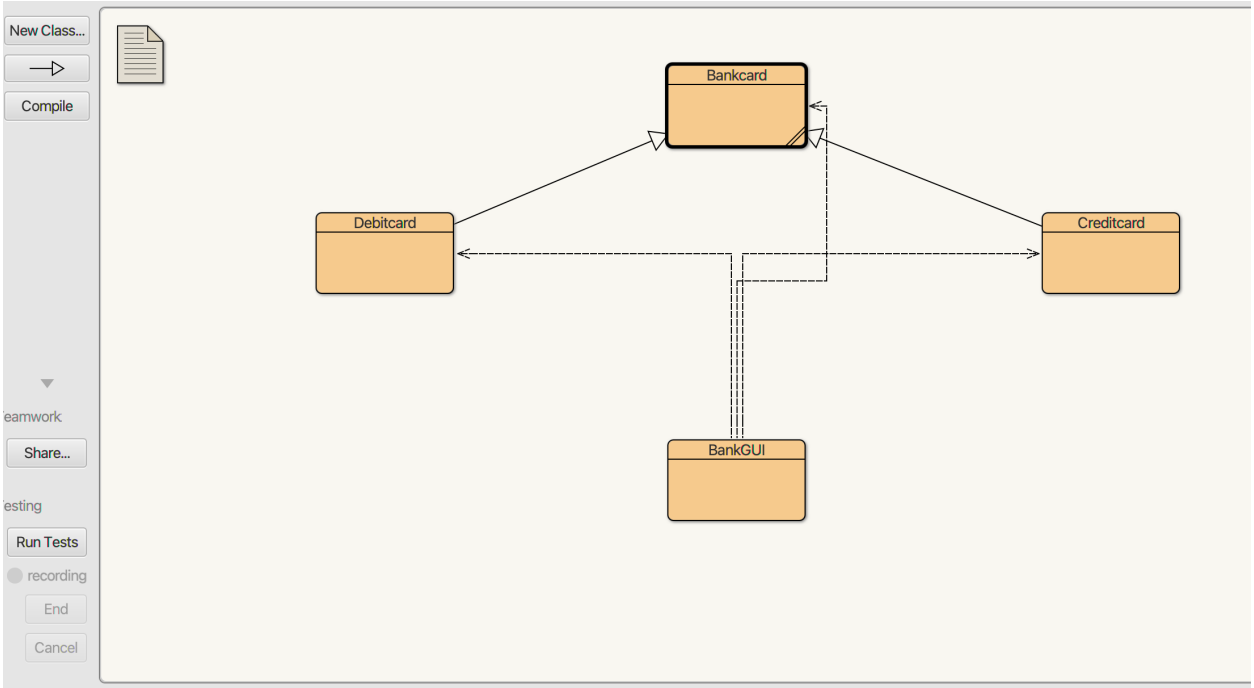


Figure 4: Class diagram in Blue j

2.1. Bankcard:

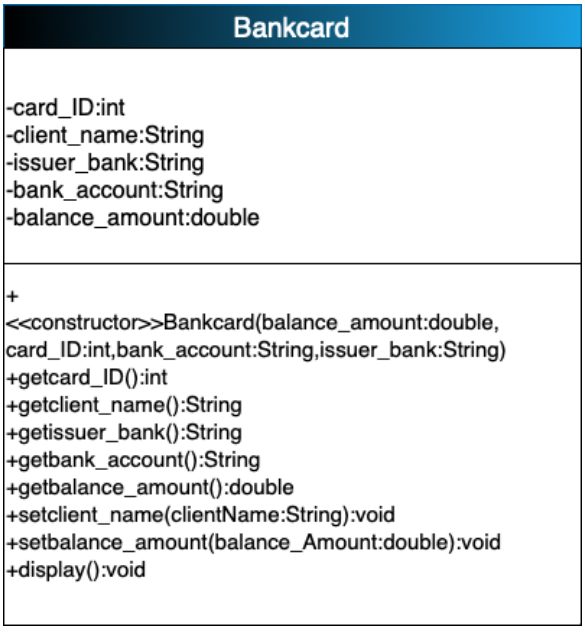


Figure 5: Class diagram of Bankcard

2.2. Debitcard:

Figure 6: Class diagram of Debitcard

2.3. Creditcard:

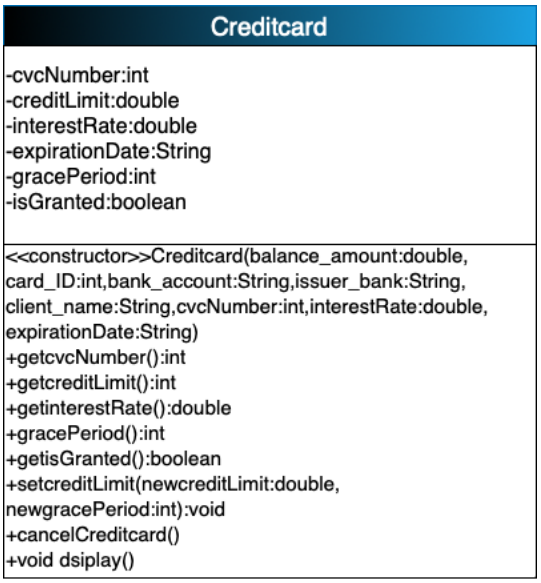


Figure 7: Class diagram of Creditcard

2.4. BankGUI:

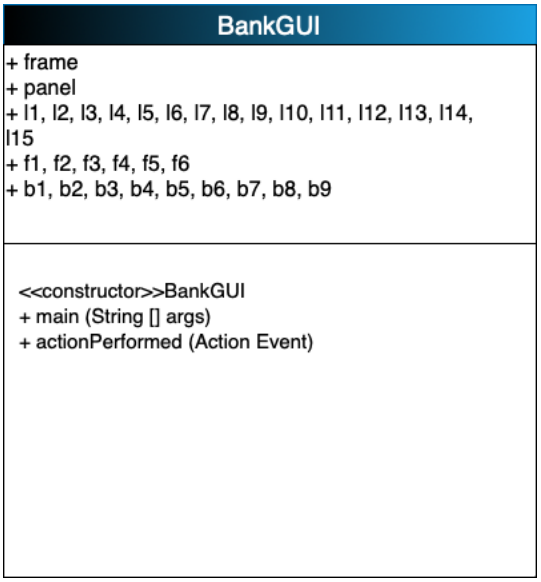


Figure 8: Class diagram of BankGUI

2.5. Parent, child and BankGUI class diagram:

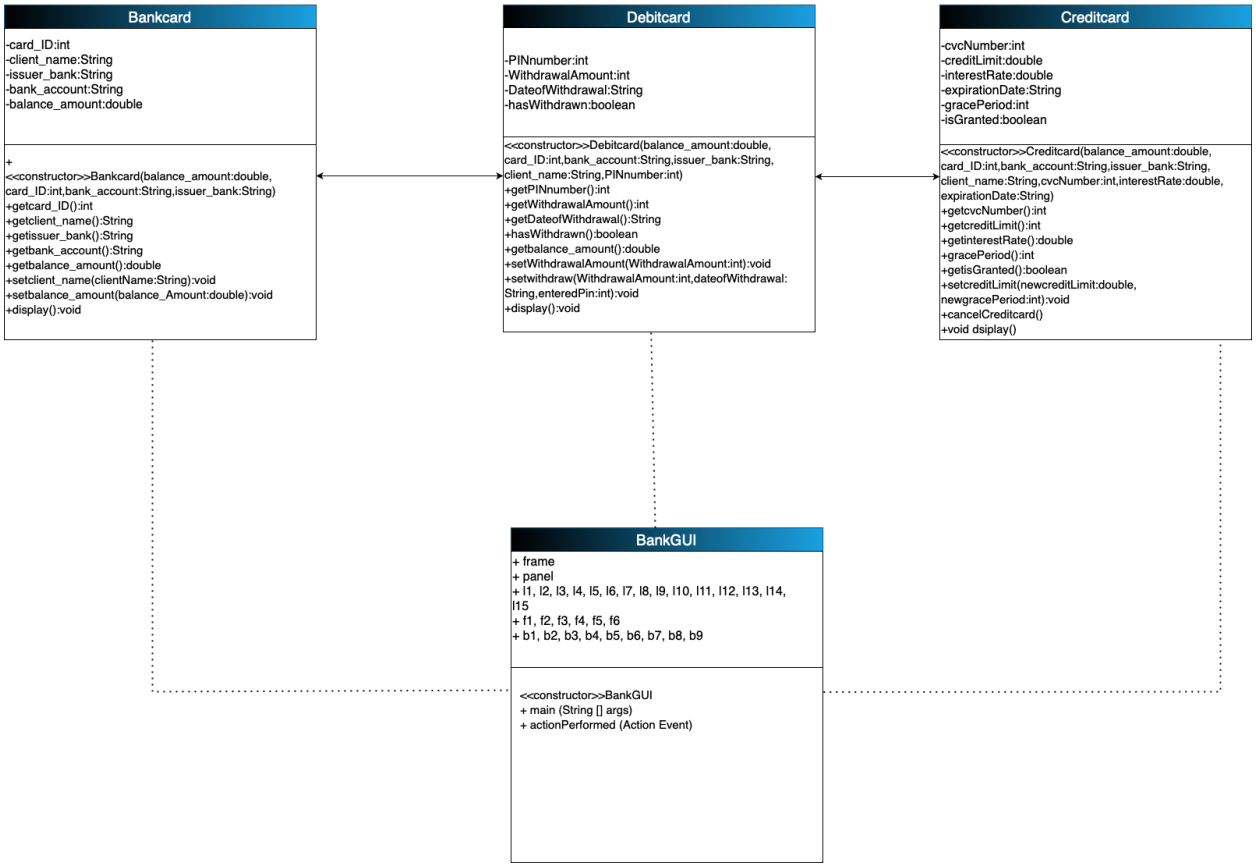


Figure 9: Parent, child and BankGUI

3. Pseudocode:

CREATE class BankGUI

CREATE jframe
CREATE jpanel
CREATE jlabel, combobox, jtextfield, jbutton
CREATE arraylist named Rounak_li

CREATE constructor BankGUI
DECLARE jframe
DECLARE jpanel
DECLARE jlabel, jcombobox, jtextfield, jbutton
DECLARE array named year, month, date

ADD jlabel, jcombobox, jtextfield, jbutton in jpanel
ADD array named year, month, date
SET BOUND FOR jframe
SET BOUND FOR jpanel
SET BOUND FOR jlabel, jcombox, jtextfield, jbutton
SET BOUND FOR array named year, month day
SET font for labels
SET layout, visible, sie for jframe

CREATE method actionPerformed(ActionEvent a)

DO

IF (a.getSource() is equals to b3)

DO

f1.setText(empty);
f2.setText(empty);
f3.setText(empty);
f4.setText(empty);
f5.setText(empty);
f6.setText(empty);
f7.setText(empty);
f8.setText(empty);
f9.setText(empty);
f10.setText(empty);
f11.setText(empty);
c1.setSelectedIndex(0);
c2.setSelectedIndex(0);
c3.setSelectedIndex(0);
c4.setSelectedIndex(0);
c5.setSelectedIndex(0);
c6.setSelectedIndex(0);

END DO

END DO

DO

ELSE IF (a.getSource() is equals to b2)
DO

```
        FOR ( Bankcard R : Rounak_li )
        DO
                IF ( R instanceof Debitcard ) DO
                        SET Debitcard object is equals to (Debitcard) R;
                        Object.display();

                END DO
        END DO
END DO

ELSE IF ( a.getSource() is equals to b7 )
DO
        FOR ( Bankcard R : Rounak_li )
        DO
                IF ( R instance of Creditcard ) DO
                Creditcard object is equals to (Creditcard) R;
                Object.display();
                END DO
        END DO
END DO

ELSE IF ( a.getSource() is equals to b6 )
DO
        IF (f1.getText().isEmpty())
        DO
                DISPLAY message

        END DO
        ELSE
        DO
                TRY
        DO
                SET Int Cardid is equals to Integer.parseInt(f1.getText());
                SET Boolean x is equals to false;

                FOR (Bankcard R : Rounak_li)
                DO
                        IF (R instanceof Creditcard) DO
                        Creditcard object is equals to (Creditcard) R;
                        IF ( Cardid is equals to object.getcard_Id())
                        DO
                                Object.cancelCreditcard();
```

```
                        SET X is equals to true;
                            DISPLAY message
                    END DO
                    ELSE
                    DO
                        SET X is equals to false;
                    END DO
                END DO
            END DO
            IF ( X is equals to false)
            DO
                DISPLAY message
            END DO
            END DO
        CATCH (NumberFromatException b) DO
                DISPLAY message
            END DO
        END DO
        ELSE IF (a.getsource() is equals to b4)
        DO
            IF (f1.getText().isEmpty() || f4.getText().isEmpty() || f7.getText().isEmpty())
            DO
                DISPLAY message
            END DO
        ELSE
        DO
            TRY
        DO
            SET int Cardid is equals to Integer.parseInt(f1.getText());
            SET int withdraw is equals to Integer.parseInt(f7.getText());
            SET int pin is equals to Integer.parseInt(f4.getText());
            SET String year is equals to (String) c1.getSelectedItem();
            SET String day is equals to (String) c2.getSelectedItem();
            SET String time is equals to year + month + day;
            SET Boolean x is equals to false;

            FOR (Bankcard R : Rounak_li)
            DO
                IF (R instanceof Debitcard) DO
                Debitcard object is equals to (Debitcard) R;
                IF (Cardid is equals to object.getcard_Id())
                DO
                    IF (pin is equals to object.getPINnumber())
```

```
                    DO
                        X is equals to true;
                        IF (withdraw is less than or equal to
object.getbalance_amount())
                        DO
                            Object.withdraw(withdraw, time, pin);
                            DISPLAY message
                        END DO
                    END DO
                    ELSE
                    DO
                        DISPLAY message
                    END DO

                        Break;
                END DO
                ELSE
                DO
                    SET X is equals to false;
                END DO
            END DO
            IF SET(x is equals to false)
            DO
                DISPLAY message
            END DO
            END DO CATCH (NumberFormatException b)
            DO
                DISPLAY message
            END DO
END DO
END DO
ELSE IF (a.getSource() is equals to b5)
DO
        IF (f1.getText().isEmpty() || f10.getText().isEmpty() ||
f9.getText().isEmpty())
        DO
            DISPLAY message
        END DO
ELSE
DO
        TRY
        DO
            SET Int Cardid is equals to Integer.parseInt(f1.getText());
```

```
                    SET Boolean x is equals to false;
                    SET Double credit is equals to Double.parseDouble(f10.getText());
                    SET Int Grace is equals to Integer.parseInt(f9.getText());
                    FOR (Bankcard R : Rounak_li)
                    DO
                        IF (R instanceof Creditcard)
                        DO
                            Creditcard object is equals to (Creditcard) R;
                            IF ( Cardid is equals to object.getcard_Id())
                            DO
                                X is equals to true;
                            IF ( credit is less than or equals to 2.5 *
    object.getbalance_amount())
                            DO
                                Object.setcreditLimit(credit, Grace);
                                DISPLAY message
                            END DO
                        ELSE
                            DO
                                DISPLAY message
                            END DO
                        Break;
                    END DO
                    ELSE
                    DO
                        SET X is equals to false;
                    END DO
                END DO
            END DO
            IF SET  ( x is equals to false)
            DO
                DISPLAY message
            END DO

END DO CATCH (NumberFormatException b)

            DO

                DISPLAY message

            END DO

END DO

END DO
```

END DO

ELSE IF (a.getSource() is equals to b1)

DO

IF (f1.getText().isEmpty() || f2.getText().isEmpty() || f3.getText().isEmpty() || f4.getText().isEmpty() || f5.getText().isEmpty() || f6.getText().isEmpty())

DO

DISPLAY message

END DO

ELSE

DO

TRY

DO

SET Double balanceamt is equalS to Double.parseDouble(f3.getText());

SET Int Cardid is equals to Integer.parseInt(f1.getText());

SET Int pin is equals to Integer.parseInt(f4.getText());

SET String Clientname is equals to f2.getText();

SET String issuerbank is equals to f5.getText();

SET String Bankacc is equals to f6.getText();

SET Boolean x is equals to true;

FOR (Bankcard R : Rounak_li)

DO

IF (R instanceof Debitcard) **DO**

Debitcard object is equals to (Debitcard) R;

IF (Cardid is equals to object.getcard_Id())

DO

X is equals to false;

END DO

END DO

```
            END DO
            IF SET ( x is equals to true)
            DO
                SET Debitcard apple is equals to new Debitcard(balanceamt,
Cardid, Bankacc, issuerbank, Clientname, pin);
                Rounak_li.add(apple);
                DISPLAY message;
            END DO
            ELSE
            DO
                DISPLAY message
            END DO
        END DO CATCH (NumberFormatException b)
            DO
                DISPLAY message
            END DO
        END DO
END DO
ELSE IF ( a.getSource() is equals to b8)
DO
    IF (f1.getText().isEmpty() || f2.getText().isEmpty() || f3.getText().isEmpty() ||
f8.getText().isEmpty() || f5.getText().isEmpty() || f6.getText().isEmpty() ||
f11.getText().isEmpty())
    DO
        DISPLAY message
    END DO
    ELSE
    DO
        TRY
```

DO

SET Double balanceamt is equals to Double.parseDouble(f3.getText());

SET Int Cardid is equals to Integer.parseInt(f1.getText());

SET int cvcnum is equals to Integer.parseInt(f8.getText());

SET String Clientname is equals to f2.getText();

SET String issuerbank is equals to f5.getText();

SET String Bankacc is equals to f6.getText();

SET double interest is equals to Double.parseDouble(f11.getText());

SET String year is equals to (String) c4.getSelectedItem();

SET String month is equals to (String) c5.getSelectedItem();

SET String day is equals to (String) c6.getSelectedItem();

SET String time is equals to year+month+day;

SET boolean x is equals to true;

FOR (Bankcard R : Rounak_li)

DO

IF (R instanceof Debitcard) **DO**

Creditcard object is equals to (Creditcard) R;

IF (Cardid is equals to object.getcard_Id))

DO

X is equals to false;

END DO

END DO

END DO

IF SET (x is equals to true)

DO

Creditcard apple is equals to new Creditcard(balanceamt, Cardid, Bankacc, issuerbank, Clientname, cvcnum, interest, time);

```
                Rounak_li.add(apple);
                DISPLAY message
            END DO
            ELSE
            DO
                DISPLAY message
            END DO
        END DO CATCH (NumberFormatException b)
            DO
                DISPLAY message
            END DO
        END DO
END DO
END DO
        CREATE main method
        DO
            CREATE object
        END DO
END DO
```

4. Method description:

The types of methods used in this report are:

- Public BankGUI () : A method is a primary component of any java application that requires a graphical user interface. This constructor method, which is called when a GUI class object is formed, is responsible for generating and configuring the user interface elements that will be displayed on the screen. To do this, the GUI() approach generally

produces on the needs of the application window's top-level container. It then adds a variety of various elements to the Jframe, depending on the demands of the application, such as JButtons, Jlabels, JTextFields. JButtons, JLabels, JTextFields, JPanels, JComboBox have been added to this project.

- actionPerformed(ActionEvent a): The actionPerformed method is a method of the ActionListener interface, which is used to handle events in Java Swing. The actionPerformed(ActionEvent a) method is called when a button is pressed. It contains the action that the buttons are expected to do when called upon. When a button is pressed, the actionPerformed(ActionEvent a) method is used. The action that the buttons are supposed to take cleared by clicking upon is contained in it.

- Static void main(String args[]): Main (String args[]) is the main method used to call the constructor. The main method of the Java program is public static void main(String args[]). To start a java program, the Java virtual machine calls the function. In the main method of this piece of code, the keyword "new" is used to invoke the constructor of the GUI class and create an instance of the GUI class.

- Add Debit: This button is used to insert debit card information into an arraylist. This button is pressed once the fields of card id, client name, issuer bank, bank account, balance amount, and pin number have been completed. When you select this button, exception handling and empty field validation are performed before checking for card duplication. If all validations are successful, a new object of the Debitcard class is created with these values and added to an arraylist called Bankcard.

- Add credit: This button is used to insert debit card information into an arraylist. This button is pressed once the fields of card id, client name, issuer bank, bank account, balance amount, and pin number have been completed. After pressing this button, error handling and empty field validation are performed before checking for card duplication. If all validations pass, a new object of the Debitcard class is created with these values and added to an arraylist called bankcard.

- Set Credit: By clicking on this button, you can change the credit limit. Before you can click on it, you must provide the necessary information, such as the card ID, credit limit, and grace period. If the card Id box is left blank, the credit limit amount entered must not be larger than the balance

amount since it will be confirmed against previously supplied information if any differences exist. If everything is in order, the Creditcard setcreditlimit function is activated. Depending on whether the validation was successful or not, the dialog box displays a message indicating success or failure.

- Cancelcreditcard: If there is a match, the credit card is revoked and a notification is displayed. An error message is given if no match is discovered. An error message is also displayed if there is an error parsing the text field.

- Clear: This adds a text box and a "Clear" button. When the button is pressed, the text in the text field is replaced with an empty string.

5. Testing

5.1. Test1:

Table 1: Test 1

Objective	To run BankGUI
Action	Terminal is opened and all java files are compiled and GUI runs
Excepted Result	The program would be compiled and run
Actual Result	The program was compiled and run
Conclusion	Test successful

```
ronakgod@Rounaks-MacBook-Air abc % ls
BankGUI$1.class         BankGUI.ctxt            Debitcard.class
BankGUI$2.class         BankGUI.java            Debitcard.ctxt
BankGUI$3.class         Bankcard.class          Debitcard.java
BankGUI$4.class         Bankcard.ctxt           README.TXT
BankGUI$5.class         Bankcard.java           doc
BankGUI$6.class         Creditcard.class        package.bluej
BankGUI$MyFrame.class   Creditcard.ctxt
BankGUI.class           Creditcard.java
ronakgod@Rounaks-MacBook-Air abc % javac BankGUI.java
ronakgod@Rounaks-MacBook-Air abc % java BankGUI.java
```

Figure 10: run BankGUI

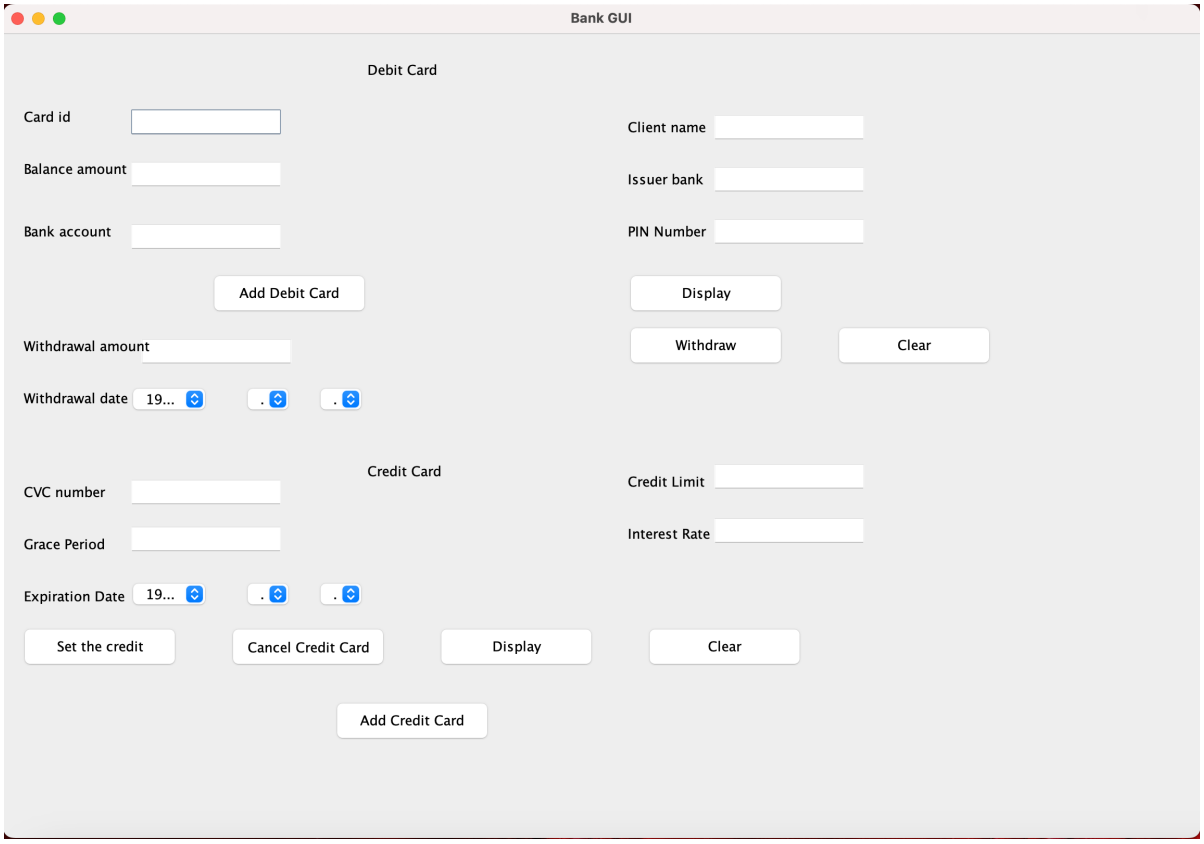


Figure 11: Test 1

5.2. Test 2

Table 2: Test 2

Objective	To add debit card
Action	After running the program Debit card is added in the GUI. The related fields are filled up
Excepted result	The debit card would be added to GUI and success would was pop up.
Actual result	The debit card was added successfully and success message was popped up.
Conclusion	The test was successful

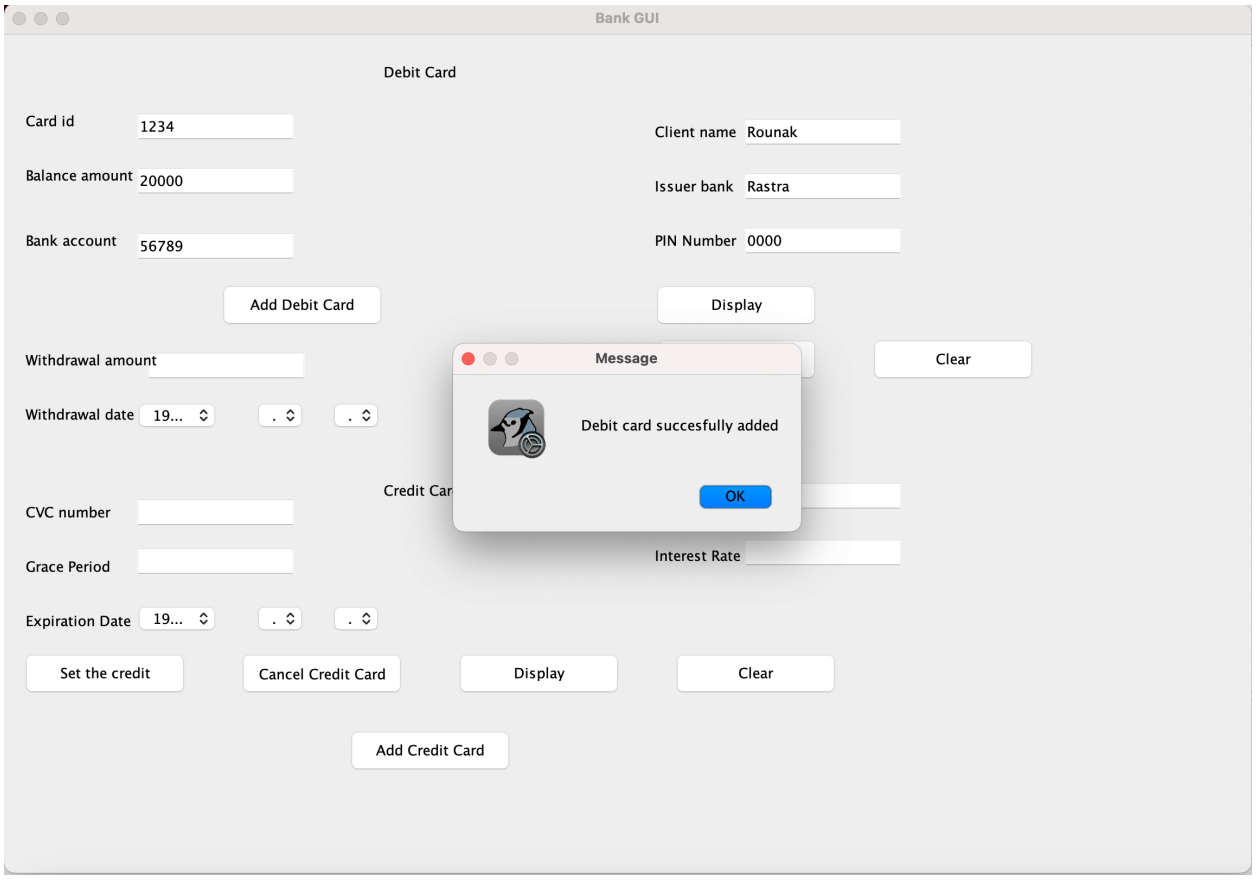


Figure 12: Test 2

5.3. Test 3

Table 3: Test 3

Objective	To add credit card
Action	After running the program credit card is added in the GUI and The related fields are filled up
Expected result	The credit card would be added and a success message would be pop up
Actual result	The credit card was added successfully and a success message popped up.
Conclusion	Test was successful

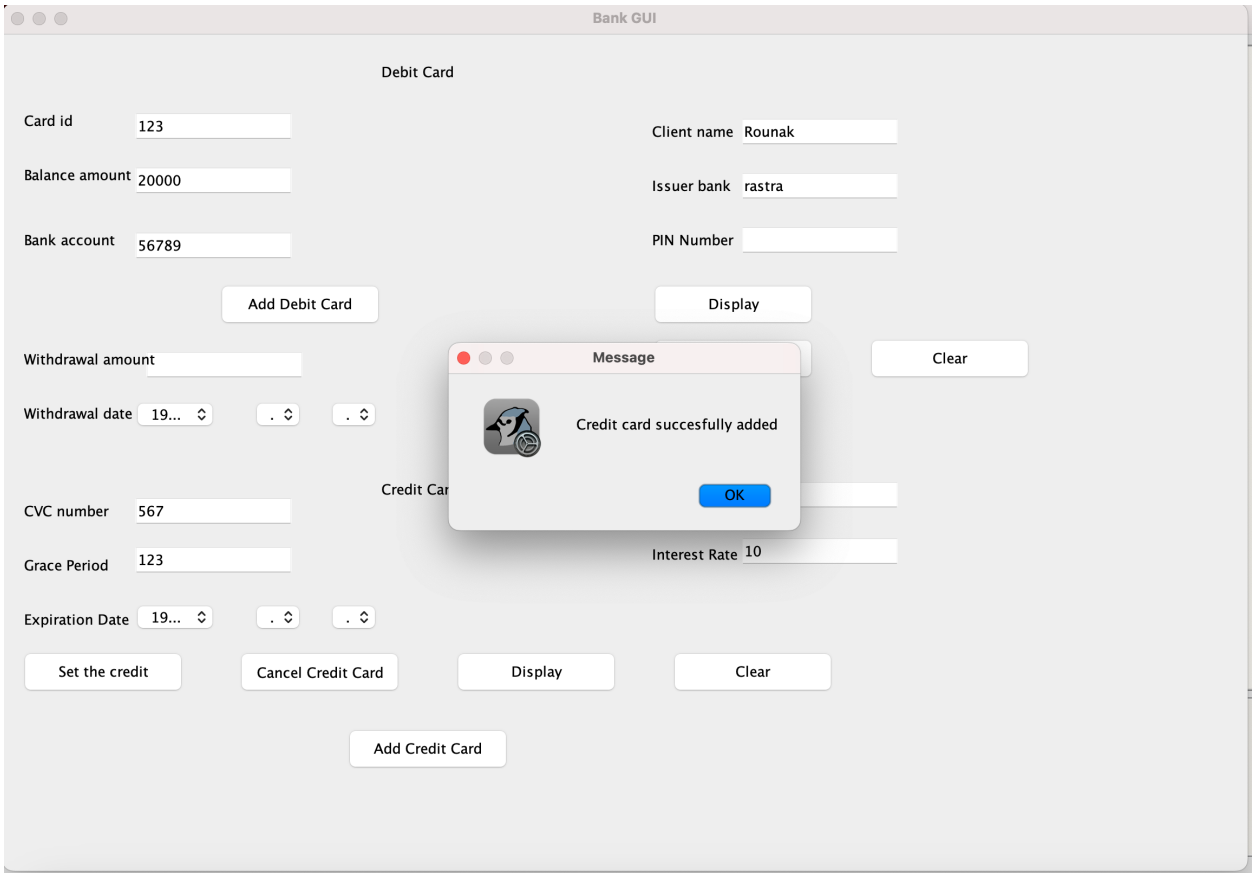


Figure 13: Test 3

5.4. Test 4

Table 4: Test 4

Objective	To withdraw amount from debit card
Action	The related fields are filled up and “Withdrawn”
Expected result	The withdraw should be display succesfully
Actual result	The withdraw was display successfully
Conclusion	Test successful

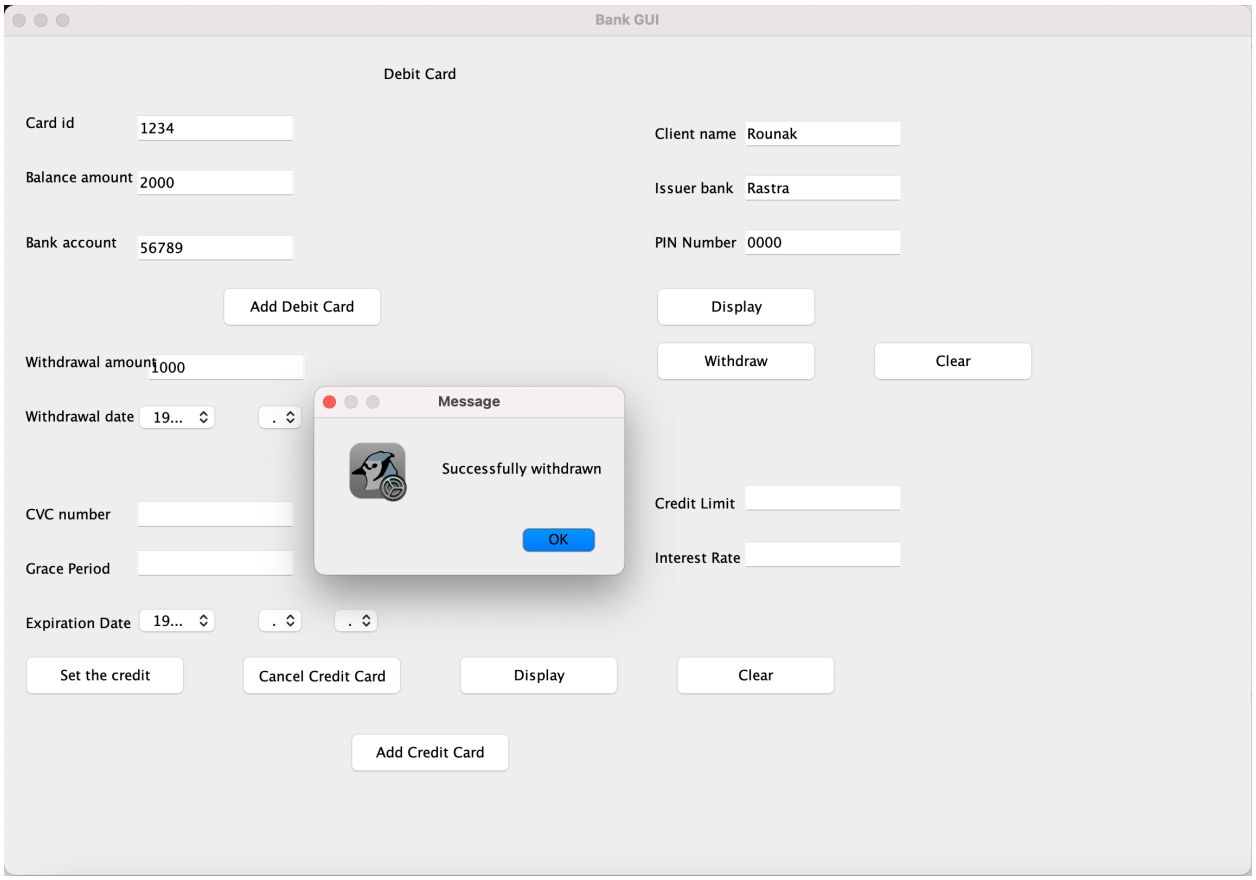


Figure 14: Test 4

5.5. Test 5

Table 5: Test 5

Objective	To set Credit limit
Action	The related fields are filled up and
Expected result	The set credit should display
Actual result	Set credit was display successfully
Conclusion	Test successful

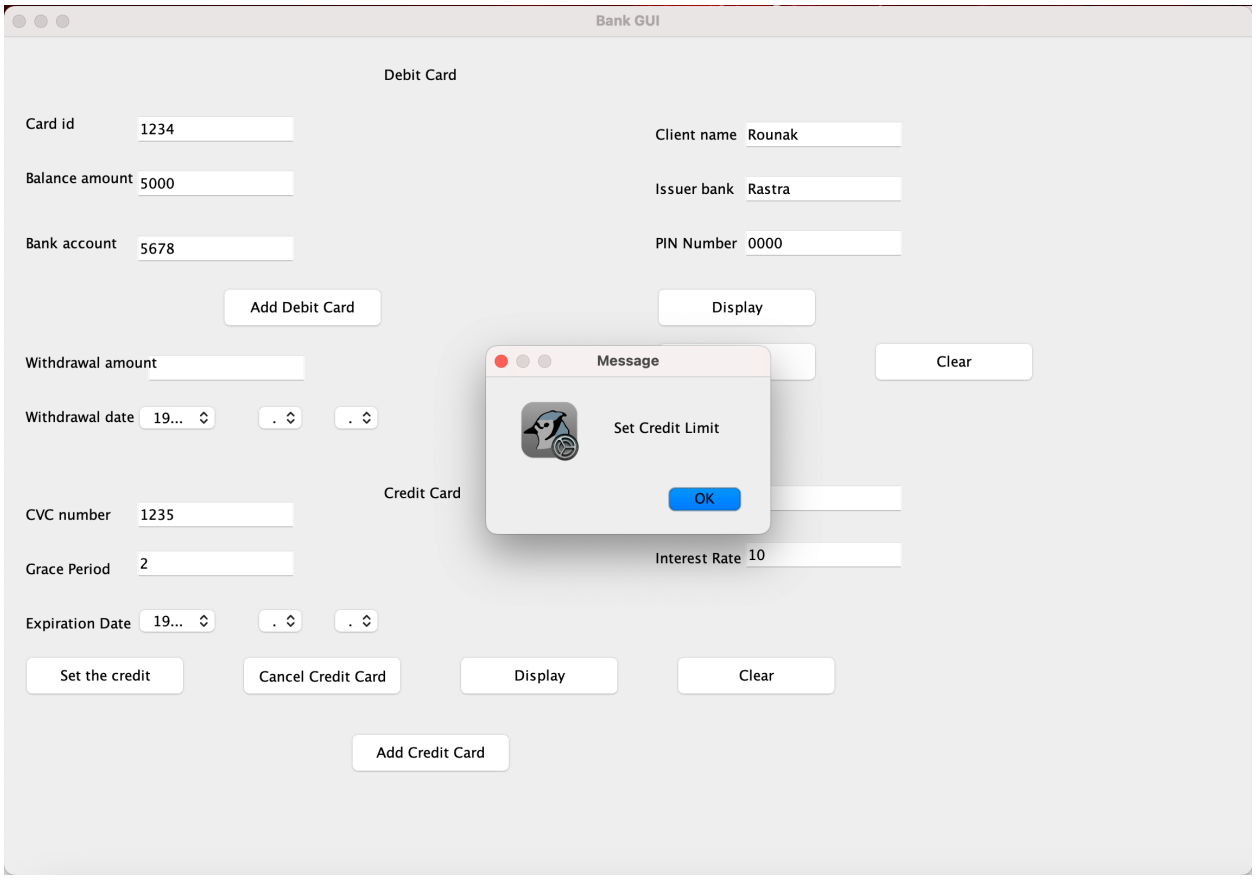


Figure 15: Test 5

5.6. Test 6

Table 6: Test 6

Objective	To cancel credit card
Action	The card is filled and Cancel Credit card
Expected result	The credit card would be removed
Action result	The credit card was removed successfully
Conclusion	Test successful

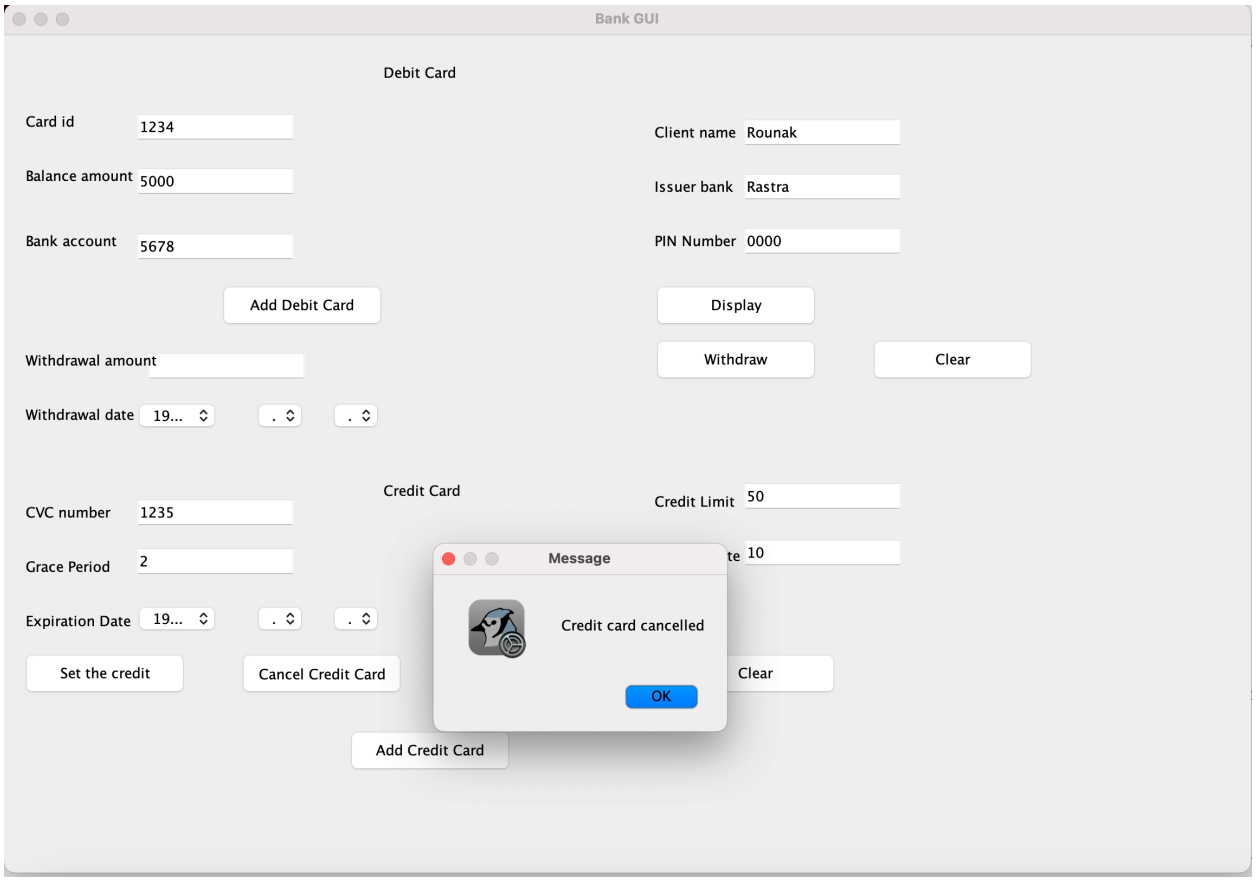


Figure 16: Test 6

5.7. Test 7

Table 7: Test 7

Objective	To check number format exception for Debit card and Credit card
Action	When button is pressed dialog box should appear with error message.
Expected result	The error message should pop up
Actual result	The error message should popped up
Conclusion	Test was successful

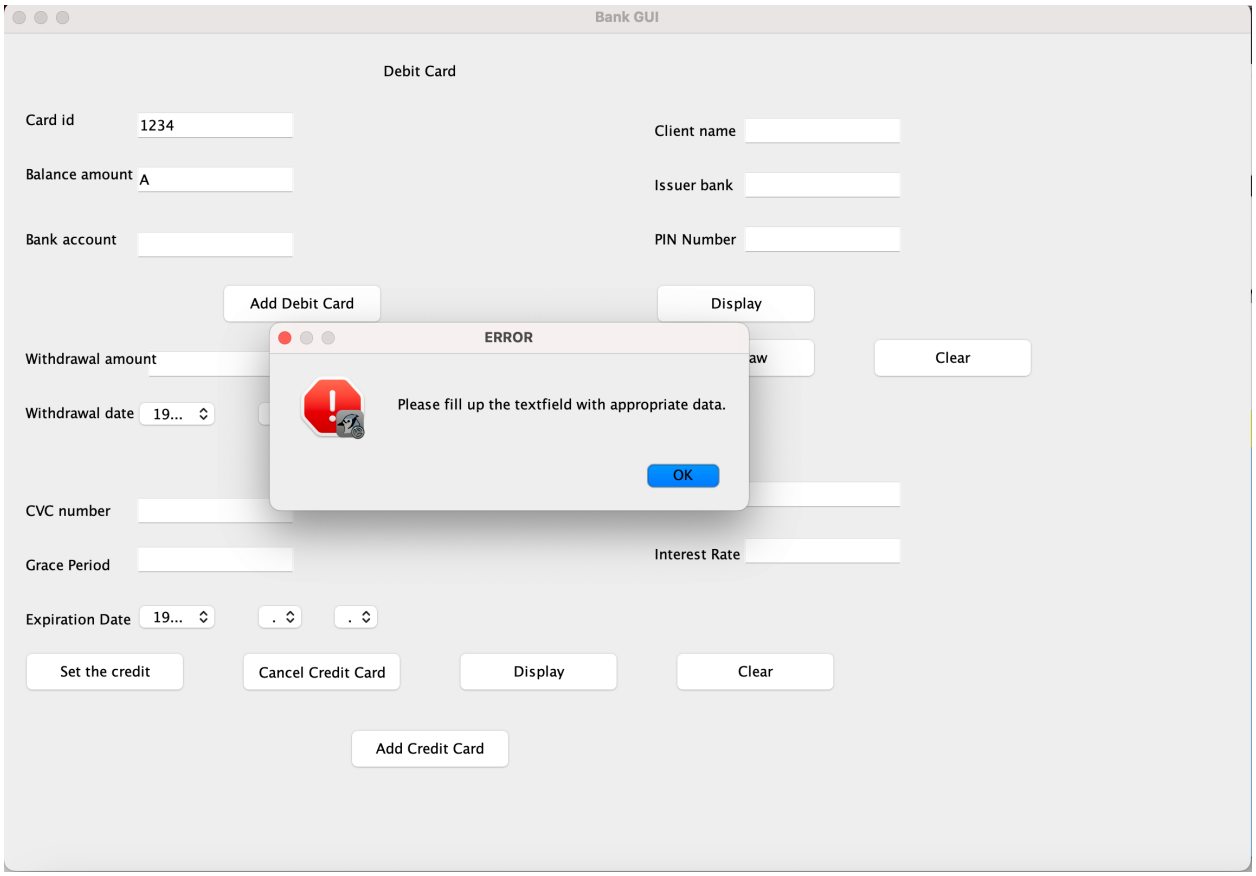


Figure 17: Test 7

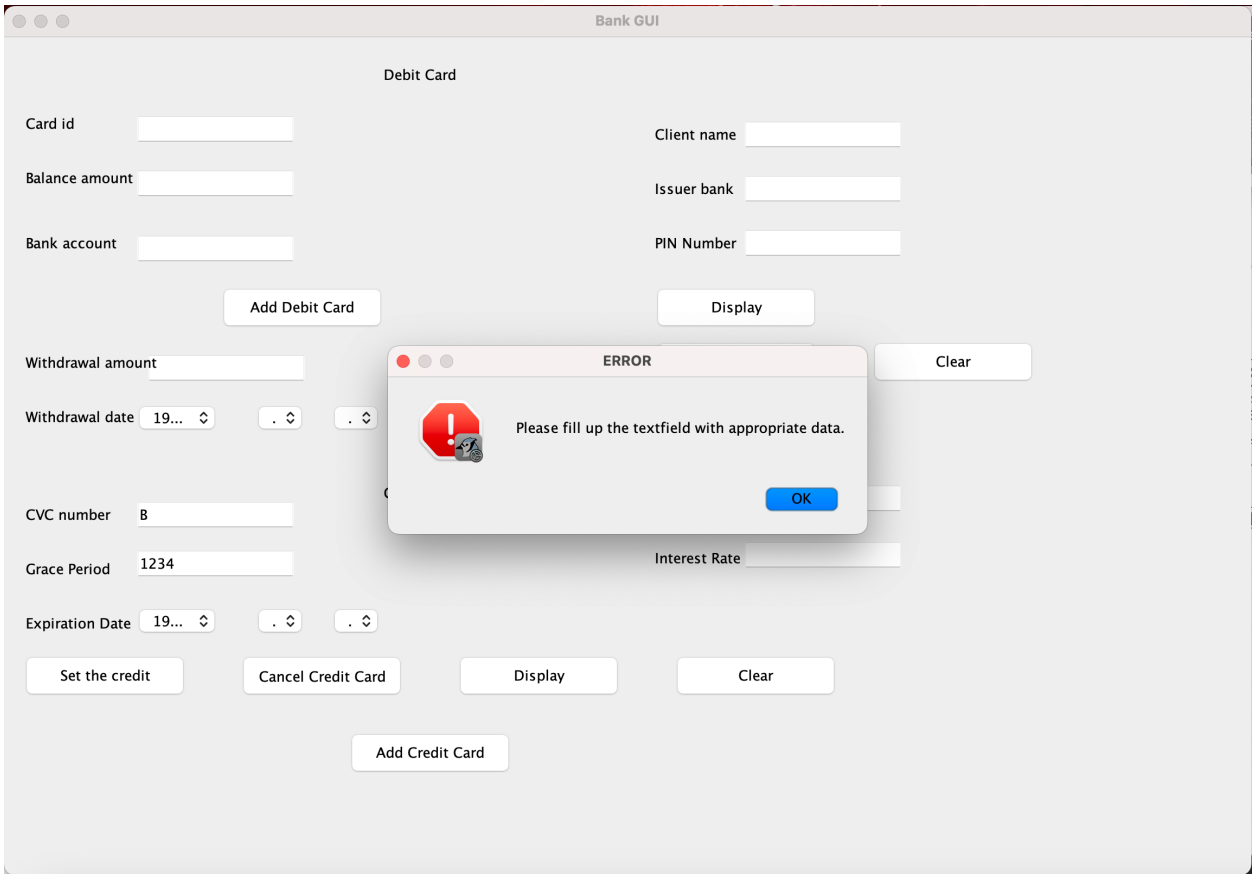


Figure 18: Test 7.1

6. Error detection

6.1. Syntax Error

Error1: Datatype in [ ] month was not written

```
l9 = new JLabel("Withdrawal date ");
String[] Year = {"1999","1992","1993","1994","1995","1996","1997","1998","1999","2000","2001"
[] Month ={"Jan","Feb","Mar","apr","May","Jun","July","Aug","Sep","Oc","Nov","Dec"};
String[] Day = {"1","2","3","4","5","7","9","10","11","12","13","14","15","16","17","18","19"
 c1 =new JComboBox<>(Year);
```

Figure 19: Syntax error 1.1

Correction of Error by written its datatype.

```
l9 = new JLabel("Withdrawal date ");
String[] Year = {"1999","1992","1993","1994","1995","1996","1997","1998","1999","2000","2001",
String[] Month ={"Jan","Feb","Mar","apr","May","Jun","July","Aug","Sep","Oc","Nov","Dec"};
String[] Day = {"1","2","3","4","5","7","9","10","11","12","13","14","15","16","17","18","19",
```

Figure 20: Syntax error 1.2

6.2. Semantic Error

Error2: datatype in Cardid was initialized as String

```
try
{
    String Cardid = Integer.parseInt(f1.getText());
    boolean x = false;
```

Figure 21: Semantic error 1.1

Correction of Error2 by initializing the datatype as int in Cardid

```
try
{
    int Cardid = Integer.parseInt(f1.getText());
    boolean x = false;

    for (Bankcard R : Rounak_li)
```

Figure 22: Semantic error 1.2

6.3. Logical Error

Errror3: The error was caused as a result of not keeping two == which are caused to assign value.

```
public void actionPerformed(ActionEvent a)
 {
        // clear button
        if (a.getSource() = b3) {
            f1.setText("");
            f2.setText("");
            f3.setText("");
            f4.setText("");
            f5.setText("");
            f6.setText("");
            f7.setText("");
```

Figure 23: Logical erorr 1.1

The Error3 solved by ==

```
public void actionPerformed(ActionEvent a)
 {
        // clear button
        if (a.getSource() == b3) {
            f1.setText("");
            f2.setText("");
            f3.setText("");
            f4.setText("");
            f5.setText("");
            f6.setText("");
            f7.setText("");


            c1.setSelectedIndex(0);
            c2.setSelectedIndex(0);
            c3.setSelectedIndex(0);
```

Figure 24: Logical error 1.2

7. Conclusion:

BankGUI This report includes a description of this class as well as class diagram that shows how it relates to previous classes. We've studied a lot about java over the last two semesters, including array list, errors, ActionListeners, event handling, and so on. This report summarizes the majority of the knowledge taught to us by our wonderful professors, who were quite helpful in completing these assignments. We learned a lot throughout this time period. It simplified the creation and use of applications for me. After finishing this course, I'd like to learn more about coding.

When working on these jobs, I personally encountered a number of challenges due to coding errors. Because I understood nothing about programming, the class diagrams and pseudo code perplexed me. I had no idea what pseudo code was at first, but with my teacher's help, I was able to understand it and use it to help me with my homework. Despite the numerous obstacles, I persisted in pushing myself to learn new things.

8. References

java. (2020). Retrieved from java: https://www.java.com/en/java_in_action/bluej.jsp

Rouse, M. (2020, August 10). *Techopedia*. Retrieved from Techopedia: https://www.techopedia.com/definition/3840/microsoft-word

Computer Hope. (2020). Retrieved from Computer Hope: https://www.computerhope.com/jargon/d/drawio.htm

Computer Hope. (2020). Retrieved from Computer Hope: https://www.computerhope.com/jargon/d/drawio.htm

Rouse, M. (2013, October). *Techopedia*. Retrieved from Techopedia: https://www.techopedia.com/definition/29530/bluej

Rouse, M. (2022, August 10). *Techopedia*. Retrieved from techopedia: https://www.techopedia.com/definition/3840/microsoft-word

9. Appendix

```
import javax.swing.*;
import javax.swing.JLabel;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
/**
 * Write a description of class BankGUI here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */


public class BankGUI implements ActionListener
{
JFrame frame;
JPanel panel;
JLabel l1, l2, l3, l4, l5, l6, l7, l8, l9, l10, l11, l12, l13, l14, l15;
JTextField f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11;
JComboBox <String> c1, c2, c3, c4, c5, c6;
JButton b1, b2, b3, b4, b5, b6, b7, b8, b9;
ArrayList<Bankcard> Rounak_li = new ArrayList<Bankcard>();
```

```
    public BankGUI()
    {

     frame = new JFrame("Bank GUI");
     panel = new JPanel();

     l1 = new JLabel("Debit Card");
     l2 = new JLabel("Card id");
          f1 = new JTextField();

         l3 = new JLabel("Client name");
         f2 = new JTextField();
         l4 = new JLabel("Balance amount");
         f3 = new JTextField();
         l5 = new JLabel("PIN Number");
         f4 = new JTextField();
         l6 = new JLabel("Issuer bank");
         f5 = new JTextField();
         l7 = new JLabel("Bank account");
         f6 = new JTextField();

          l8= new JLabel("Withdrawal amount");
         f7 = new JTextField();
        l9 = new JLabel("Withdrawal date ");
        String[] Year =
{"1999","1992","1993","1994","1995","1996","1997","1998","1999","2000","2001","2002",
"2003","2004","2005","2006","2007","2008","2009","2010","2011","2012","2013","2014","
2015","2016","2017","2018","2019","2020","2021","2022","2023"};
        String[] Month
={"Jan","Feb","Mar","apr","May","Jun","July","Aug","Sep","Oc","Nov","Dec"};
```

```
        String[] 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"};

         c1 =new JComboBox<>(Year);

         c2 =new JComboBox<>(Month);

         c3 =new JComboBox<>(Day);

          b1 =new JButton("Add Debit Card");

          b2 =new JButton("Display");

          b3 =new JButton("Clear");

          b4 =new JButton("Withdraw");



          l10 = new JLabel("Credit Card");

          l11 = new JLabel("CVC number");

          f8 = new JTextField();

          l12 = new JLabel("Grace Period");

          f9 = new JTextField();

          l13 = new JLabel("Credit Limit");

          f10 = new JTextField();

          l14 = new JLabel("Interest Rate");

          f11 = new JTextField();

          l15 = new JLabel("Expiration Date");

         String[] year1 =
{"1999","1992","1993","1994","1995","1996","1997","1998","1999","2000","2001","2002",
"2003","2004","2005","2006","2007","2008","2009","2010","2011","2012","2013","2014","
2015","2016","2017","2018","2019","2020","2021","2022","2023"};

        String[] Day1 =
{"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"};

        String[] Month1
={"Jan","Feb","Mar","Apr","May","Jun","July","Aug","Sep","Oct","Nov","Dec"};
```

```
c4 =new JComboBox<>(year1);
c5 =new JComboBox<>(Month1);
c6 =new JComboBox<>(Day1);
b5 =new JButton("Set the credit");
b6 =new JButton("Cancel Credit Card");
b7 =new JButton("Display");
b8 =new JButton("Add Credit Card");
b9 =new JButton("Clear");

frame.add(l1);
frame.add(l2);
frame.add(l3);
frame.add(l4);
frame.add(l5);
frame.add(l6);
frame.add(l7);
frame.add(l8);
frame.add(l9);
frame.add(l10);
frame.add(l11);
frame.add(l12);
frame.add(l13);
frame.add(l14);
frame.add(l15);
frame.add(f1);
frame.add(f2);
frame.add(f3);
frame.add(f4);
```

```
frame.add(f5);
frame.add(f6);
frame.add(f7);
frame.add(f8);
frame.add(f9);
frame.add(f10);
frame.add(f11);
frame.add(c1);
frame.add(c2);
frame.add(c3);
frame.add(c4);
frame.add(c5);
frame.add(c6);
frame.add(b1);
frame.add(b2);
frame.add(b3);
frame.add(b4);
frame.add(b5);
frame.add(b6);
frame.add(b7);
frame.add(b8);
frame.add(b9);



l11.setBounds(20,420,200,40);
f8.setBounds(120,425,150,30);
```

```
l12.setBounds(20,470,200,40);
f9.setBounds(120,470,150,30);
l13.setBounds(600,410,200,40);
f10.setBounds(680,410,150,30);
l14.setBounds(600,460,200,40);
f11.setBounds(680,462,150,30);


l15.setBounds(20,520,200,40);
c4.setBounds(120,527,80,25);
c5.setBounds(230,527,50,25);
c6.setBounds(300,527,50,25);


b5.setBounds(18,569,150,40);
b6.setBounds(218,569,150,40);
b7.setBounds(418,569,150,40);
b9.setBounds(618,569,150,40);
b8.setBounds(318,640,150,40);


l1.setBounds(350,15,200,40);
l10.setBounds(350,400,200,40);
l2.setBounds(20,70,100,20);
f1.setBounds(120,70,150,30);
l3.setBounds(600,70,200,40);
f2.setBounds(680,75,150,30);
```

```
l4.setBounds(20,120,100,20);
f3.setBounds(120,120,150,30);
l5.setBounds(600,170,200,40);
f4.setBounds(680,175,150,30);
l6.setBounds(600,120,200,40);
f5.setBounds(680,125,150,30);
l7.setBounds(20,170,200,40);
f6.setBounds(120,180,150,30);
 l8.setBounds(20,280,200,40);
f7.setBounds(130,290,150,30);

 l9.setBounds(20,330,200,40);
c1.setBounds(120,340,80,25);
c2.setBounds(230,340,50,25);
c3.setBounds(300,340,50,25);

b1.setBounds(200,230,150,40);
   b2.setBounds(600,230,150,40);
   b3.setBounds(600,280,150,40);
   b3.setBounds(800,280,150,40);
   b4.setBounds(600,280,150,40);


b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
```

```
        b6.addActionListener(this);
        b7.addActionListener(this);
        b8.addActionListener(this);
        b9.addActionListener(this);




        frame.setLayout(null);
        frame.setVisible(true);
        frame.setSize(1150,800);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


}
public void actionPerformed(ActionEvent a)
 {
        // clear button
        if (a.getSource() == b3) {
            f1.setText("");
            f2.setText("");
            f3.setText("");
            f4.setText("");
            f5.setText("");
            f6.setText("");
            f7.setText("");
```

```
        c1.setSelectedIndex(0);
        c2.setSelectedIndex(0);
        c3.setSelectedIndex(0);


    }
    else if (a.getSource() == b9) {
        f8.setText("");
        f9.setText("");
        f10.setText("");
        f11.setText("");
        f5.setText("");
        f6.setText("");
        f7.setText("");
        f8.setText("");
        f9.setText("");
        f10.setText("");
        f11.setText("");

        c4.setSelectedIndex(0);
        c5.setSelectedIndex(0);
        c6.setSelectedIndex(0);


    }

    else if (a.getSource() == b2)
    {
```

```
            for (Bankcard R : Rounak_li)
            {
                if(R instanceof Debitcard){
                    Debitcard object = (Debitcard) R;
                    object.display();
                }
            }
        }
        else if (a.getSource() == b7)
        {
            for (Bankcard R : Rounak_li)
            {
                if(R instanceof Creditcard){
                    Creditcard object = (Creditcard) R;
                    object.display();
                }
            }
        }
        else if (a.getSource() == b6)
        {
            if (f1.getText().isEmpty())
            {
                JOptionPane.showMessageDialog(frame, "Please fill up the textfield with
appropriate data.","ERROR", JOptionPane.ERROR_MESSAGE);
            }
            else
            {
              try
              {
```

```
            int Cardid = Integer.parseInt(f1.getText());
            boolean x = false;

            for (Bankcard R : Rounak_li)
            {
               if(R instanceof Creditcard){
                  Creditcard object = (Creditcard) R;
                  if ( Cardid == object.getcard_Id())
                  {
                     object.cancelCreditcard();
                     x = true;
                     JOptionPane.showMessageDialog(frame, "Credit card
cancelled","Message", JOptionPane.INFORMATION_MESSAGE);
                  }
                  else
                  {
                     x = false;
                  }
               }
            }
            if (x = false)
            {
               JOptionPane.showMessageDialog(frame, "Card id not found",
"Error",JOptionPane.ERROR_MESSAGE);
            }
          }catch (NumberFormatException b) {
               JOptionPane.showMessageDialog(frame, "Please enter the appropriate
values in the textfield", "Error",JOptionPane.ERROR_MESSAGE);
                  }
```

```
        }
    }
    else if (a.getSource() == b4)
    {
        if (f1.getText().isEmpty() || f4.getText().isEmpty() || f7.getText().isEmpty())
        {
            JOptionPane.showMessageDialog(frame, "Please fill up the textfield with
appropriate data.","ERROR", JOptionPane.ERROR_MESSAGE);
        }
        else
        {
          try
          {
            int Cardid = Integer.parseInt(f1.getText());
            int withdraw = Integer.parseInt(f7.getText());
            int pin = Integer.parseInt(f4.getText());
            String year = (String) c1.getSelectedItem();
            String month = (String) c2.getSelectedItem();
            String day = (String) c3.getSelectedItem();
            String time = year+month+day;
            boolean x = false;

            for (Bankcard R : Rounak_li)
           {
              if(R instanceof Debitcard){
                 Debitcard object = (Debitcard) R;
                 if ( Cardid == object.getcard_Id())
                 {
                    if (pin == object.getPINnumber())
```

```
                    {
                        x = true;
                        if (withdraw <= object.getbalance_amount())
                        {
                            object.withdraw(withdraw, time, pin);
                            JOptionPane.showMessageDialog(frame, "Successfully
withdrawn","Message", JOptionPane.INFORMATION_MESSAGE);
                        }
                        else
                        {
                            JOptionPane.showMessageDialog(frame, "Low Balance
Amount", "Error",JOptionPane.ERROR_MESSAGE);
                        }
                    }
                    else
                    {
                        JOptionPane.showMessageDialog(frame, "Wrong Pin number",
"Error",JOptionPane.ERROR_MESSAGE);
                    }
                    break;
                }
                else
                {
                    x = false;
                }
            }
        }
        if (x = false)
        {
```

```
                JOptionPane.showMessageDialog(frame, "Card id not found",
"Error",JOptionPane.ERROR_MESSAGE);
              }
            }catch (NumberFormatException b)
            {
                 JOptionPane.showMessageDialog(frame, "Please enter the appropriate
values in the textfield", "Error",JOptionPane.ERROR_MESSAGE);
            }
        }
        }
        else if (a.getSource() == b5)
        {
           if (f1.getText().isEmpty() || f10.getText().isEmpty() || f9.getText().isEmpty())
           {
              JOptionPane.showMessageDialog(frame, "Please fill up the textfield with
appropriate data.","ERROR", JOptionPane.ERROR_MESSAGE);
           }
           else
           {
              try
              {
                  int Cardid = Integer.parseInt(f1.getText());
                  boolean x = false;
                  double credit = Double.parseDouble(f10.getText());
                  int Grace = Integer.parseInt(f9.getText());
                 for (Bankcard R : Rounak_li)
                 {
                    if(R instanceof Creditcard)
                    {
```

```
                    Creditcard object = (Creditcard) R;
                    if ( Cardid == object.getcard_Id())
                    {
                        x = true;
                        if ( credit <= 2.5*object.getbalance_amount())
                        {
                            object.setcreditLimit(credit, Grace);
                            JOptionPane.showMessageDialog(frame, "Set Credit
Limit","Message", JOptionPane.INFORMATION_MESSAGE);


                        }
                        else
                        {
                            JOptionPane.showMessageDialog(frame, "Credit limit too high",
"Error",JOptionPane.ERROR_MESSAGE);
                        }
                        break;
                    }
                    else
                    {
                        x = false;
                    }
                }
            }
            if (x = false)
            {
                JOptionPane.showMessageDialog(frame, "Card id not found",
"Error",JOptionPane.ERROR_MESSAGE);
            }
```

```
            }catch (NumberFormatException b)
           {
                JOptionPane.showMessageDialog(frame, "Please enter the appropriate values in the textfield", "Error",JOptionPane.ERROR_MESSAGE);
           }
          }
       }
       else if (a.getSource() == b1)
       {
          if (f1.getText().isEmpty() || f2.getText().isEmpty() || f3.getText().isEmpty() || f4.getText().isEmpty() || f5.getText().isEmpty() || f6.getText().isEmpty())
          {
              JOptionPane.showMessageDialog(frame, "Please fill up the textfield with appropriate data.","ERROR", JOptionPane.ERROR_MESSAGE);
          }
          else
          {
             try
             {
                double balanceamt = Double.parseDouble(f3.getText());
                int Cardid = Integer.parseInt(f1.getText());
                int pin = Integer.parseInt(f4.getText());
                String Clientname = f2.getText();
                String issuerbank = f5.getText();
                String Bankacc = f6.getText();
                boolean x = true;
                for (Bankcard R : Rounak_li)
                {
                   if(R instanceof Debitcard){
```

```
                    Debitcard object = (Debitcard) R;
                    if ( Cardid == object.getcard_Id())
                    {
                        x = false;
                    }
                  }
               }
               if ( x == true)
               {
                  Debitcard apple = new Debitcard(balanceamt, Cardid, Bankacc,
issuerbank, Clientname, pin);
                  Rounak_li.add(apple);
                  JOptionPane.showMessageDialog(frame, "Debit card succesfully
added","Message", JOptionPane.INFORMATION_MESSAGE);
               }
               else
               {
                  JOptionPane.showMessageDialog(frame, "Card id already exist",
"Error",JOptionPane.ERROR_MESSAGE);
               }
            }catch (NumberFormatException b)
             {
               JOptionPane.showMessageDialog(frame, "Please enter the appropriate
values in the textfield", "Error",JOptionPane.ERROR_MESSAGE);
             }


         }
      }
      else if (a.getSource() == b8)
      {
```

```
        if (f1.getText().isEmpty() || f2.getText().isEmpty() || f3.getText().isEmpty() ||
f8.getText().isEmpty() || f5.getText().isEmpty() || f6.getText().isEmpty() ||
f11.getText().isEmpty())
        {
            JOptionPane.showMessageDialog(frame, "Please fill up the textfield with
appropriate data.","ERROR", JOptionPane.ERROR_MESSAGE);
        }
        else
        {
           try
           {
              double balanceamt = Double.parseDouble(f3.getText());
              int Cardid = Integer.parseInt(f1.getText());
              int cvcnum = Integer.parseInt(f8.getText());
              String Clientname = f2.getText();
              String issuerbank = f5.getText();
              String Bankacc = f6.getText();
              double interest = Double.parseDouble(f11.getText());
              String year = (String) c4.getSelectedItem();
              String month = (String) c5.getSelectedItem();
              String day = (String) c6.getSelectedItem();
              String time = year+month+day;
              boolean x = true;
              for (Bankcard R : Rounak_li)
              {
                 if(R instanceof Debitcard){
                    Creditcard object = (Creditcard) R;
                    if ( Cardid == object.getcard_Id())
                    {
```

```
                        x = false;
                    }
                }
            }
            if ( x == true)
            {
                Creditcard apple = new Creditcard(balanceamt, Cardid, Bankacc,
issuerbank, Clientname, cvcnum, interest, time );
                Rounak_li.add(apple);
                JOptionPane.showMessageDialog(frame, "Credit card succesfully
added","Message", JOptionPane.INFORMATION_MESSAGE);
            }
            else
            {
                JOptionPane.showMessageDialog(frame, "Card id already exist",
"Error",JOptionPane.ERROR_MESSAGE);
            }
        }catch (NumberFormatException b)
         {
            JOptionPane.showMessageDialog(frame, "Please enter the appropriate
values in the textfield", "Error",JOptionPane.ERROR_MESSAGE);
         }


    }
  }
}


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

```
}
```