CS4001NI Programming

30% Individual Coursework

2022-23 autumn

Student Name: Rajita Maharjan

London Met ID: 22067335

College ID: np01ai4a220052

Group: AI2

Assignment Due Date: Wednesday, May 10, 2023

Assignment Submission Date: Wednesday, May 10, 2023

Table of Contents

Table of figures

Table of tables

1. Introduction

1.1 About the coursework

The given coursework is to add a class to the existing project developed in coursework1 to make the graphical user imterface (GUI) for the system that stored details of the Bank Card in an ArrayList. The primary objective of this coursework is to improve the current project by adding a GUI. The major objective is to provide a user friendly interface which allows users to connect with the database of the bank card information.

1.2 Tools used in the coursework

BlueJ is a Java integrated development environment (IDE) built basically for educational purposes yet also useful for small-scale software development. It is operated by the Java Development Kit (JDK). It was created to help the learning the teaching of the object oriented programming and as a result it varies from other development environments.

Figure 1 : Blue j

Java is a popular object-oriented programming language and software platform used by billions of devices that includes laptop computers, mobile devices, gaming consoles medical equipment, and many more. Java's principles and grammar are based on the C and C++ programming languages.

Figure 2 : Java

Draw.io is an unique tool for creating diagrams and charts. You can use the software's automatic layout option or design a custom layout. They provide a wide range of shapes and hundreds of graphic components to let you create a one-of-a-kind diagram or chart. The drag-and-drop tool makes it easy to construct a great looking diagram or chart.

Figure 3 : Draw.io

Ms Word

Finally, the tool used for the documentation part of cousework is MS Word. Microsoft Word is a word processor that was developed by Microsoft. It is one of the Microsoft Office suite's office productivity programs but can also be purchased as a stand-alone product.

Figure 4 : Ms Word

2. Class Diagram

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. (paradigm, n.d.)

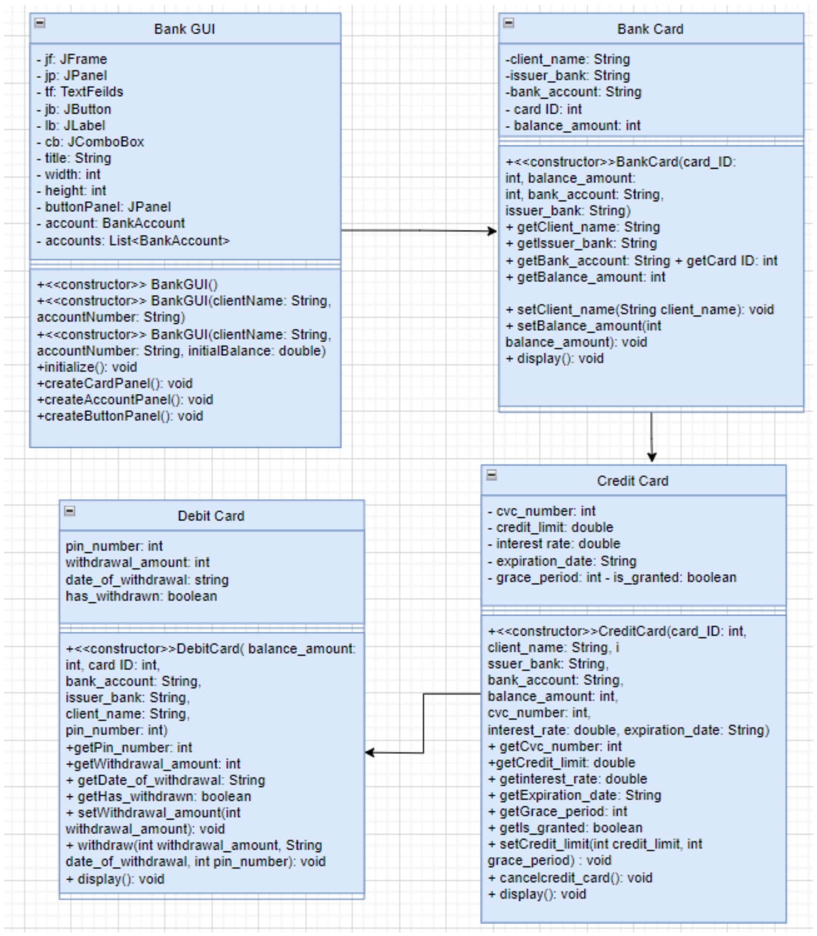

Bank GUI
- jf: JFrame
- jp: JPanel
- tf: TextFeilds
- jb: JButton
- lb: JLabel
- cb: JComboBox
- title: String
- width: int
- height: int
- buttonPanel: JPanel
- account: BankAccount
- accounts: List<BankAccount>
+<<constructor>> BankGUI()
+<<constructor>> BankGUI(clientName: String, accountNumber: String)
+<<constructor>> BankGUI(clientName: String, accountNumber: String, initialBalance: double)
+initialize(): void
+createCardPanel(): void
+createAccountPanel(): void
+createButtonPanel(): void
Bank Card
-client_name: String
-issuer_bank: String
-bank_account: String
- card ID: int
- balance_amount: int
+<<constructor>>BankCard(card_ID: int, balance_amount: int, bank_account: String, issuer_bank: String)
+ getClient_name: String
+ getIssuer_bank: String
+ getBank_account: String + getCard ID: int
+ getBalance_amount: int
+ setClient_name(String client_name): void
+ setBalance_amount(int balance_amount): void
+ display(): void
Credit Card
- cvc_number: int
- credit_limit: double
- interest rate: double
- expiration_date: String
- grace_period: int - is_granted: boolean
+<<constructor>>CreditCard(card_ID: int, client_name: String, i ssuer_bank: String, bank_account: String, balance_amount: int, cvc_number: int, interest_rate: double, expiration_date: String)
+ getCvc_number: int
+getCredit_limit: double
+ getinterest_rate: double
+ getExpiration_date: String
+ getGrace_period: int
+ getIs_granted: boolean
+ setCredit_limit(int credit_limit, int grace_period) : void
+ cancelcredit_card(): void
+ display(): void
Debit Card
pin_number: int
withdrawal_amount: int
date_of_withdrawal: string
has_withdrawn: boolean
+<<constructor>>DebitCard(balance_amount: int, card ID: int, bank_account: String, issuer_bank: String, client_name: String, pin_number: int)
+getPin_number: int
+getWithdrawal_amount: int
+ getDate_of_withdrawal: String
+ getHas_withdrawn: boolean
+ setWithdrawal_amount(int withdrawal_amount): void
+ withdraw(int withdrawal_amount, String date_of_withdrawal, int pin_number): void
+ display(): void

Figure 5: Class Diagram

3. Pseudocode:

Pseudocode is a accessible explanation of what a computer program or algorithm must perform, written in normal language rather than programming language. Pseudocode is occasionally used as a detailed step in the development process of a program. It enables designers or lead programmers to communicate the concept in great detail, while also providing programmers with a thorough template for the next step of developing code in a certain programming language.

3.1 Pseudocode of BankGui

```
CREATE  a class BankGUI
DO
   // Declare instance variables
   DECLARE jf as JFrame
   DECLARE jp as JPanel
   DECLARE tf1, tf2, tf3, tf4, tf5, tf6, tf7, tf8, tf9, tf11, tf12, tf13, tf14, tf15, tf16,
tf17, tf19, tf20, tf21, tf22 AS JTextField
   DECLARE jb1, jb2, jb3, jb4, jb5, jb6, jb7, jb8, jb9, jb10, jb11 AS JButton
   DECLARE lb1, lb2, lb3, lb4, lb5, lb6, lb7, lb8, lb9, lb10, lb11, lb12, lb13,
lb14, lb15, lb16, lb17, lb19, lb20, lb21, lb22, heading1, heading2 AS JLabel
   DECLARE cb1, cb2, cb3, cb4, cb5, cb6 AS JComboBox
     CREATE  a constructor BankGUI()
DO
     CREATE  a JFrame and set its title
     CREATE  a JPanel
     SET the layout of the JPanel to null
     SET the bankground color of the JPanel
     CREATE a JLabel for “Debit Card”
     SET the font and size of the heading
     SET the position and size of the heading
     SET the text color of the heading
     ADD the heading label to the JPanel
     CREATE a JLabel for “Credirt Card
     SET the font and size of the heading
     SET the position and size of the heading
     SET the text color of the heading
     ADD the heading label to the JPanel
     ADD the JPanel to the JFrame
     INITIALIZE JTextField variables
     CREATE JTextField instances
     SET JTextFields with parameters
     SET JTextFields background color to new Color
     ADD TextFields to the JPanel
     SET lb1 text to " Name "
     SET lb2 text to " Card ID "
     SET lb3 text to " Bank Account "
     SET lb4 text to " Issuer Bank "
     SET lb5 text to " Pin Number "
     SET lb6 text to " Balance Amount "
     SET lb7 text to " Card ID "
     SET lb8 text to " Withdrawal Amount "
     SET lb9 text to " Pin Number "
     SET lb10 text to " Date of Withdrawal "
     SET lb11 text to " Name "
     SET lb12 text to " Card ID "
     SET lb13 text to " Bank Account "
     SET lb14 text to " Issuer Bank "
     SET lb15  text to " Interest Rate "
```

```
    SET lb16  text to " Balance Amount "
    SET lb17  text to " CVC Number "
    SET lb18  text to " Expiry Date "
    SET lb19  text to " Card ID "
    SET lb20  text to " Grace period "
    SET lb21  text to " Credit Limit "
    SET lb22  text to " Card ID "
    SET all the JLabel foreground color to Color.BLACK
    SET JLabel bounds with parameters
    ADD all JLabel to the JPanel
    SET JButtons
    SET  JButton bounds with parameters
    ADD JButton to JPanel
    DECLARE years as String
    DECLARE months as String
    DECLARE  days as String
    SET JComboBox
    SET JComboBox bounds with parameters
    ADD JComboBox to the JPanel
    SET JPanel visibility to true
    SET JPanel setSize with parameters
    SET JFrame setSize with parameters
    SET JFrame visibility to true
    SET JFrame setDefaultCloseOperation with parameter
    DECLARE bankCards as ArrayList of BankCard objects
    ADD action listener for "Add" button
    CALL jb2 addActionListener with parameter (new ActionListener()
DO
```

```
DECLARE balanceAmount AS integer
DECLARE cardId AS integer
DECLARE pinNumber AS integer
DECLARE bankAccount AS string
DECLARE issuerBank AS string
DECLARE clientName AS string

   SET bankAccount to value of tf3.getText()
   SET issuerBank to value of tf4.getText()
   SET clientName to value of tf1.getText()

   TRY
      SET balanceAmount to integer value of tf6.getText()
      SET cardId to integer value of tf2.getText()
      SET pinNumber to integer value of tf5.getText()
   CATCH NumberFormatException n
      CALL JOptionPane.showMessageDialog with parameters
      RETURN
   END TRY
CREATE new DebitCard object with input values
   DECLARE debitCard as DebitCard
   SET debitCard to new DebitCard with parameters (balanceAmount,
cardId, bankAccount, issuerBank, clientName, pinNumber)

   ADD new DebitCard object to ArrayList of BankCard objects
   CALL bankCards.add with parameter (debitCard)

CALL JOptionPane.showMessageDialog with parameters (null, "Debit
card added successfully.")
```

```
END DO)
CALL jb1 addActionListener with parameter (new ActionListener()
DO
        CALL TextFeilds setText with parameter ("")
END DO)

CALL jb3 addActionListener with parameter (new ActionListener()
DO
        DECLARE cardId as integer
        DECLARE clientName as string
      SET clientName to value of tf1.getText()
   TRY
      SET cardId to integer value of tf2.getText()
   CATCH NumberFormatException n
      CALL JOptionPane.showMessageDialog with parameters
      RETURN
   END TRY

   DECLARE debitCard as DebitCard
   SET debitCard to null

   FOR each card IN bankCards
      IF card is an instance of DebitCard and card.getcardID() is equal to
cardId
         SET debitCard to card
         CALL debitCard setclientName with parameter (clientName)
         BREAK
      END IF
   END FOR
```

```
    IF debitCard is not null
        CALL JOptionPane.showMessageDialog with parameter (null, debitCard.display())
    ELSE
        CALL JOptionPane.showMessageDialog with parameter (null, "Debit card not found")
    END IF
END DO)
DO
    SET tf11 to ""
    SET tf12 to ""
    SET tf13 to ""
    SET tf14 to ""
    SET tf15 to ""
    SET tf16 to ""
    SET tf17 to ""
    SET cb1.selectedIndex to 0
    SET cb2.selectedIndex to 0
    SET cb3.selectedIndex to 0
END DO

ADD ActionListener to jb5
    WHEN actionPerformed event occurs
        CALL ClearButtonClicked()
    END WHEN
DO DisplayCreditCardButtonClicked()
    DECLARE cardId as integer
    DECLARE clientName as string
    TRY
        SET clientName to tf11.getText()
        SET cardId to convert to integer(tf12.getText())
    CATCH NumberFormatException
        DISPLAY "Invalid input values. Please check and try again."
        RETURN
    END TRY

    DECLARE creditCard AS CreditCard
    SET creditCard to null

    FOR each card in bankCards
        IF card is instance of CreditCard and card.getcardID() equals cardId
THEN
            SET creditCard to card
            BREAK
        END IF
    END FOR

    IF creditCard is not null then
        creditCard.setclientName(clientName)
        DISPLAY creditCard.display()
    ELSE
        DISPLAY "Credit card not found"
    END IF
END DO
ADD ActionListener to jb6
    WHEN actionPerformed event occurs
```

```
        CALL DisplayCreditCardButtonClicked()
    END WHEN
DO
    DECLARE cardId as integer
    DECLARE balanceAmount  as integer
    DECLARE cvcNumber  as integer
    DECLARE interestRate  as double
    DECLARE clientName  as string
    DECLARE issuerBank  as string
    DECLARE bankAccount  as string
    DECLARE expirationDate  as string

    TRY
        SET clientName  to  tf11.getText()
        SET issuerBank   to  tf13.getText()
        SET bankAccount  to  tf14.getText()
        SET expirationDate  to  cb1.getSelectedItem().toString() + "-" +
cb2.getSelectedItem().toString() + "-" + cb3.getSelectedItem().toString()
        SET cardId  to convert to integer (tf12.getText())
        SET balanceAmount  to convert to integer (tf16.getText())
        SET cvcNumber  to convert to integer (tf17.getText())
        SET interestRate  to convert to integer (tf15.getText())
    CATCH NumberFormatException
        DISPLAY "Invalid input values. Please check and try again."
        RETURN
    END TRY

    DECLARE creditCard as CreditCard
    SET creditCard to new CreditCard(balanceAmount, clientName,
bankAccount, issuerBank, cardId, cvcNumber, interestRate,
expirationDate)


    bankCards.add(creditCard)


    DISPLAY "Credit card added successfully"
END DO


ADD ActionListener to jb7
    WHEN actionPerformed event occurs
        CALL AddCreditCardButtonClicked()
    END WHEN
DO
    DECLARE cardId as integer
    DECLARE creditLimit  as integer
    DECLARE gracePeriod  as integer


    TRY
        SET cardId to convert to integer (tf19.getText())
        SET creditLimit to convert to integer (tf21.getText())
        SET gracePeriod to convert to integer (tf20.getText())
    CATCH NumberFormatException
        DISPLAY "Invalid input values. Please check and try again."
        RETURN
    END TRY


    DECLARE creditCard as CreditCard
```

```
            BREAK from loop
        END IF
    END FOR

    IF creditCard is  not null THEN
        creditCard.setcreditLimit(creditLimit, gracePeriod)
        DISPLAY "Credit limit updated to " + creditLimit + " and grace period updated to " + gracePeriod + "."
    ELSE
        DISPLAY "Invalid card ID."
    END IF
END DO

ADD ActionListener to jb8
    WHEN actionPerformed event occurs

        CALL UpdateCreditLimitButtonClicked()
    END WHEN
DO ActionListener for jb9
    WHEN actionPerformed event occurs
        GET cardId from tf22
        TRY
            CONVERT cardId tp Integer
        CATCH NumberFormatException
            SHOW "Invalid input values. Please check and try again."
            RETURN
        END TRY

        FOR  each card in bankCards
```

```
            IF card.getcardID() equals cardId
               CreditCard creditCard =  cast card as CreditCard
               creditCard.cancelcreditCard()
               SHOW "Credit card " + cardId + " has been cancelled."
               RETURN
            END IF
         END FOR
         SHOW "Invalid card ID."
      END WHEN
   END DO

   DO ActionListener for jb10
      WHEN actionPerformed event occurs

         SET tf7 to ""
         SET  tf8 to ""
         SET  tf9 to ""
         SET  cb4.selectedIndex to 0
         SET  cb5.selectedIndex to 0
         SET  cb6.selectedIndex to 0
      END WHEN
   END DO
   DO ActionListener for jb11
      WHEN actionPerformed event occurs

         SET  tf19 to ""
         SET  tf20 to ""
         SET  tf21 to ""
```

```
            END WHEN
          END DO
     END DO
```

4. Description of Methods of all the buttons.

4.1 Add a Debit Card:

This button is used to register a Debit Card with the system. When the button is clicked, the information given in the card details text fields (balance amount, card ID, bank account, issuer bank, client name, and PIN number) are used for creating a new Debit Card object. After that, the new Debit Card object is added to the BankCard ArrayList.

4.2 Add a credit card:

This button is used to enter a Credit Card into the system. When you click the information you give in the text fields for card details (card ID, client name, issuer bank, bank account, balance amount, CVC number, interest rate, and expiration date) are used for creating a new Credit Card object. The new Credit Card object is then added to the BankCard ArrayList.

4.3 Withdraw from debit card:

This button is used to make a withdrawal from a Debit Card. In the proper text fields the user must input the card ID, withdrawal amount, date of withdrawal, and PIN number. When you click it, the program checks the card ID and PIN number you entered. If a valid card ID and PIN number are supplied, the requested withdrawal amount will be deducted from the Debit Card object's balance.

4.4 SET the credit limit:

This button is used to SET a Credit Card's credit limit and grace period. In the proper text fields, the user must input the card ID, new credit limit, and new grace period. When you click it, the program checks the card ID you entered.

If a valid card ID is entered, that particular Credit Card object's credit limit and grace period are updated with the new values.

4.5 Cancel credit card:

This button is used to cancel a credit card transaction. The card ID must be entered into the text feild by the user. When you click it, the program checks the card ID you entered. If a valid card ID is entered, the related Credit Card object from the BankCard ArrayList is deleted.

4.6 Display:

This button is used to display information about the relevant class (Debit Card or Credit Card) based on the details input or actions completed. Based on the input, the application obtains and shows the right details about the selected card (Debit Card or Credit Card) when it is clicked.

4.7 Clear:

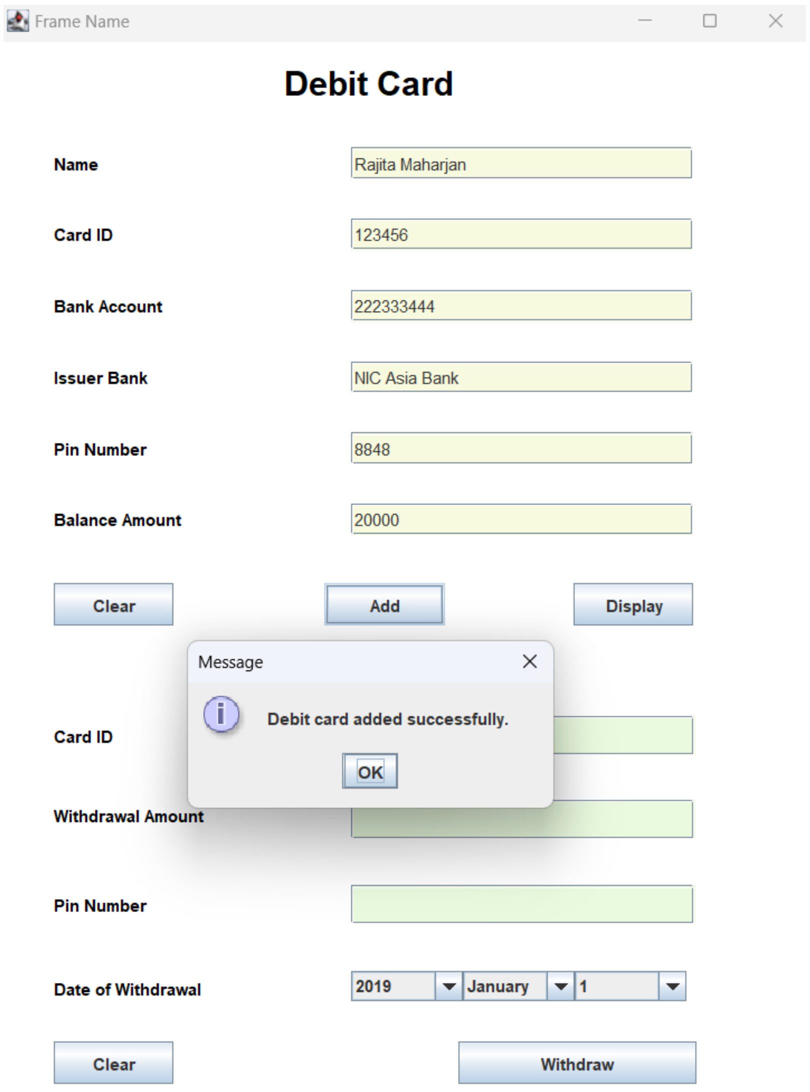
This button is used to remove any previously entered values from the text fields. When this button is pressed, all text fields in the GUI are cleared, leaving a blank input field for new card data or operations.

5. Testing

5.1 Test 1: To test that the users can add new debit card to their account using ‘Add debit card ’ button..

Objective	To test that the users can add new debit card to their account using ‘Add debit card ’ button.**.**
Action	Enter the valid input details in the respective text fields for card details. • Name: Rajita Maharjan • Card Id: 123456 • Bank Account: 222333444 • Issuer Bank: NIC Asia Bank • PIN Number: 8848 • Balance Amount: 20000 Click on the ‘add’ button to add debit card.
Expected result	The debit card with the given details should be added to the system successfully.
Actual result	The debit card with the details is successfully added to the system without any errors.
Conclusion	The test is successful.

***Table 1**: Test 1 – To test that the users can add new debit card.*



Name	Rajita Maharjan
Card ID	123456
Bank Account	222333444
Issuer Bank	NIC Asia Bank
Pin Number	8848
Balance Amount	20000
Card ID	
Withdrawal Amount	
Pin Number	
Date of Withdrawal	2019 January 1

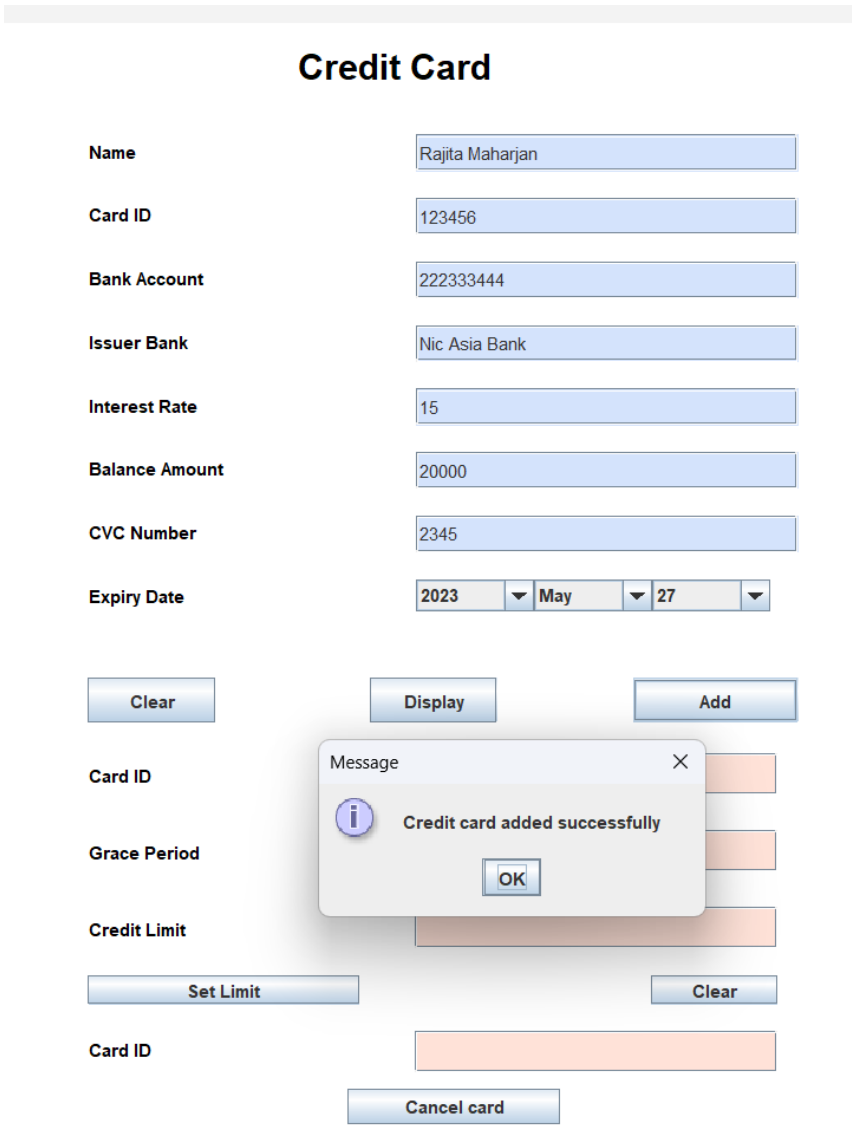
Debit card added successfully.

Figure 6: Debit Card added.

5.2 Test 2: To test that the users can add new credit card to their account using ‘Add credit card ’ button.

Objective	To test that the users can add new credit card to their account using ‘Add credit card ’ button.
Action	Fill out the details in the respective text fields. • Name: Rajita Maharjan • Card Id: 123456 • Bank Account: 222333444 • Issuer Bank: NIC Asia Bank • Interest Rate:15 • Balance Amount: 20000 • CVC Number: 2345 • Expiration Date: 2024- May- 27 Click on the ‘add’ button to add the credit card.
Expected result	The credit card with the given details should be added to the system.
Actual result	The credit card with the details is successfully added to the system without any errors.
Conclusion	The test is successful.

Table 2: Test 2 – To test that the users can add new credit card to their account.



Credit Card

Name	Rajita Maharjan
Card ID	123456
Bank Account	222333444
Issuer Bank	Nic Asia Bank
Interest Rate	15
Balance Amount	20000
CVC Number	2345
Expiry Date	2023 May 27

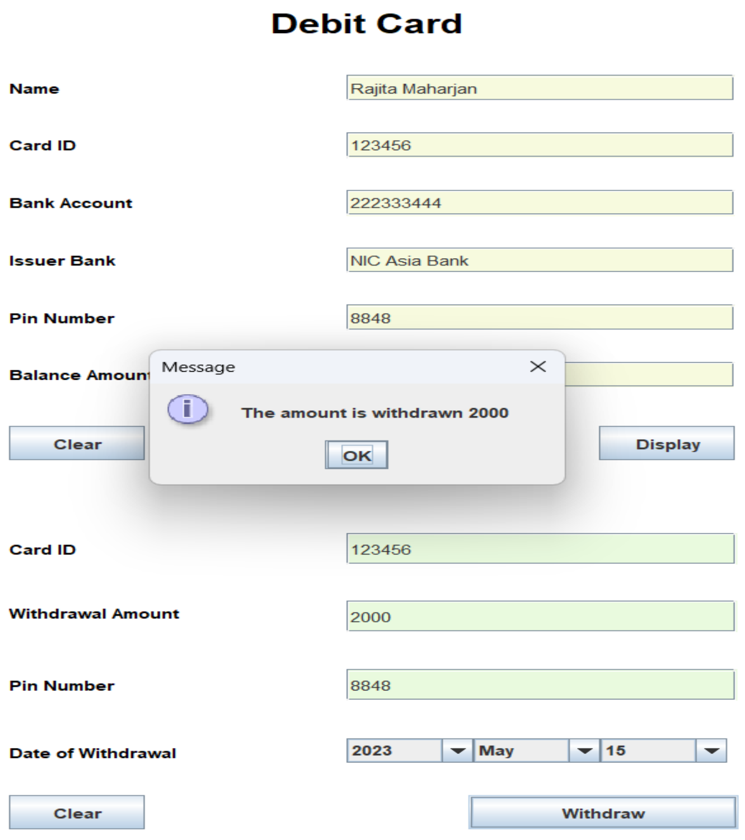
Credit card added successfully

Figure 7: Credit Card added.

5.3 Test 3: To test that user can withdraw money from their debit card by using ‘withdraw’ button.

Objective	To test that user can withdraw money from their debit card by using ‘withdraw’ button.
Action	Fill out the details in the respective text fields. Click on the ‘withdraw’ button start the withdrawal process.
Expected result	The stated amount should be successfully withdrawn from the debit card balance, and the transaction should be recorded.
Actual result	The stated amount was successfully withdrawn.
Conclusion	The test is successful.

Table 3: Test 3 – To test that user can withdraw money from debit card.



Debit Card

Name: Rajita Maharjan
Card ID: 123456
Bank Account: 222333444
Issuer Bank: NIC Asia Bank
Pin Number: 8848
Balance Amount:

Message
The amount is withdrawn 2000
OK

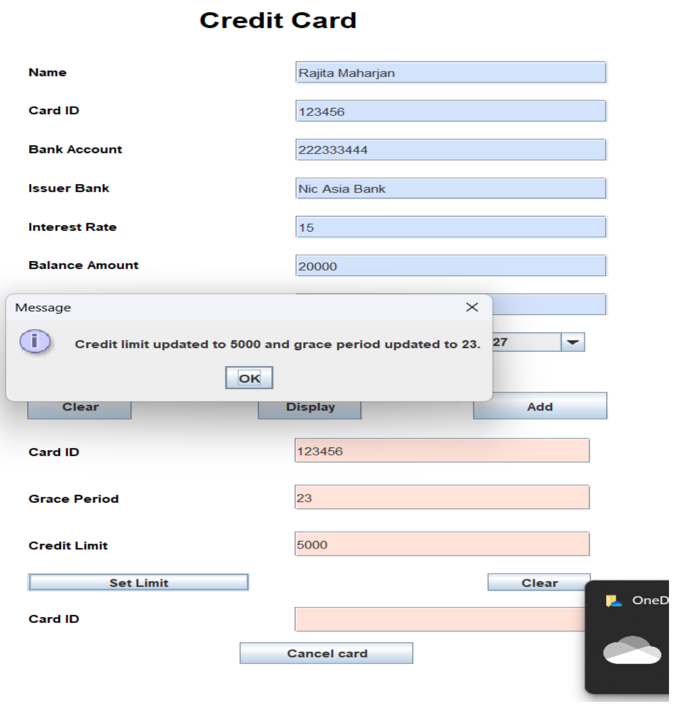
Card ID: 123456
Withdrawal Amount: 2000
Pin Number: 8848
Date of Withdrawal: 2023 May 15

Figure 8: Withdraw amount from debit card

5.4 Test 4: To test that user can SET the credit limit for their credit card by using 'Set Limit' button.

Objective	To test that user can SET the credit limit for their credit card by using 'Set Limit' button.
Action	Fill out the details in the respective text field. To save the changes in credit card limit, click on 'SET limit' button.
Expected result	The given credit card's credit limit and grace period should be successfully updated.
Actual result	The credit card's credit limit and grace period is successfully updated.
Conclusion	The test is successful.

Table 4: Test4 – To test that user can SET the credit limit for their credit card.



Rajita Maharjan
123456
222333444
Nic Asia Bank
15
20000
27

Credit limit updated to 5000 and grace period updated to 23.

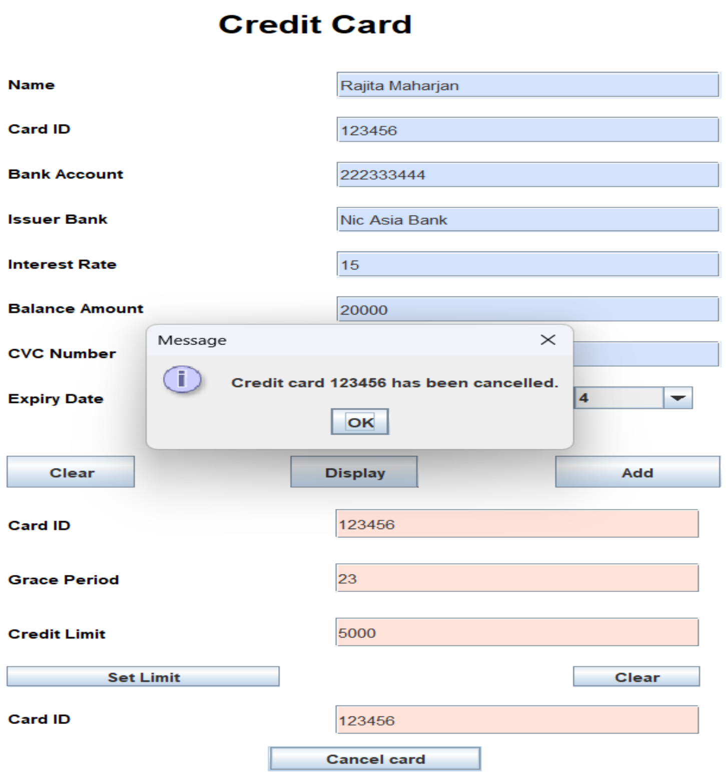
123456
23
5000

Figure 9: Set The credit Limit

5.5 Test 5: To test that user can cancel a card by using ‘Cancel ’ button.

Objective	To test that user can cancel a card by using ‘Cancel’ button.
Action	Enter the Card Id in the respective text field. Click ‘cancel credit card’ button.
Expected result	The credit card should be cancelled from the users account.
Actual result	The credit card is cancelled from the users account successfully as expected.
Conclusion	The test is successful.

Table 5: Test 5 - To test that the user cancel a credit card.


Credit Card

Name	Rajita Maharjan
Card ID	123456
Bank Account	222333444
Issuer Bank	Nic Asia Bank
Interest Rate	15
Balance Amount	20000
CVC Number	
Expiry Date	4

Message

Credit card 123456 has been cancelled.

Card ID	123456
Grace Period	23
Credit Limit	5000

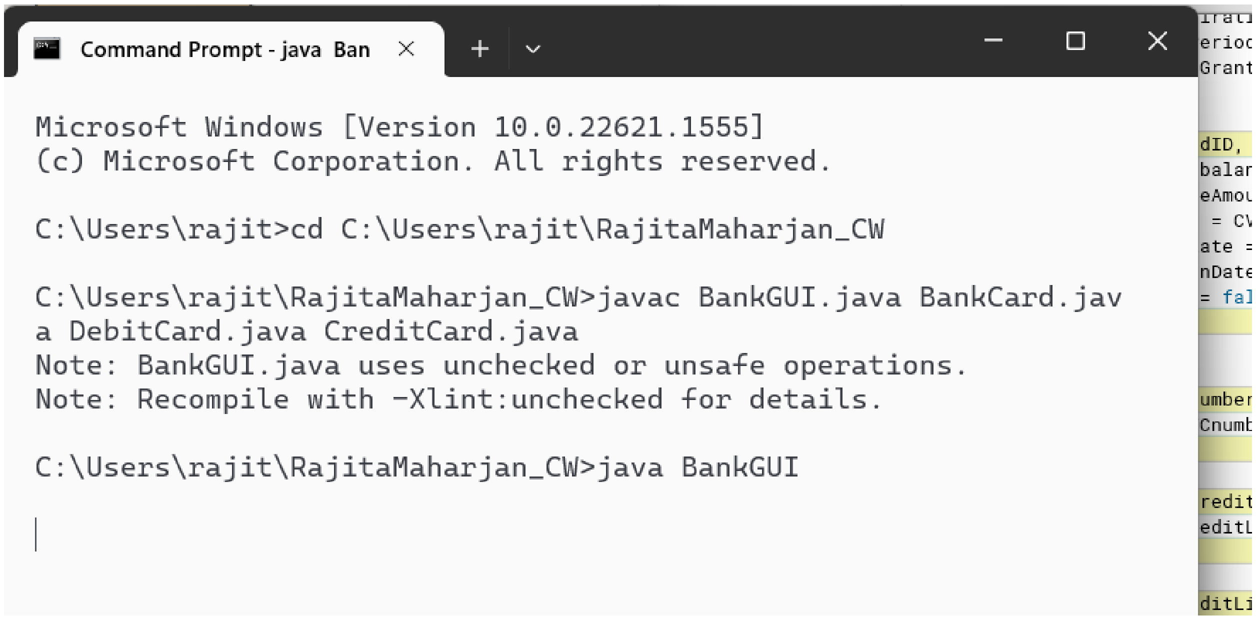
Card ID	123456

Figure 10: Cancel the credit card.

5.6 Test 6: To Test that the program can be compiled and run using the command prompt.

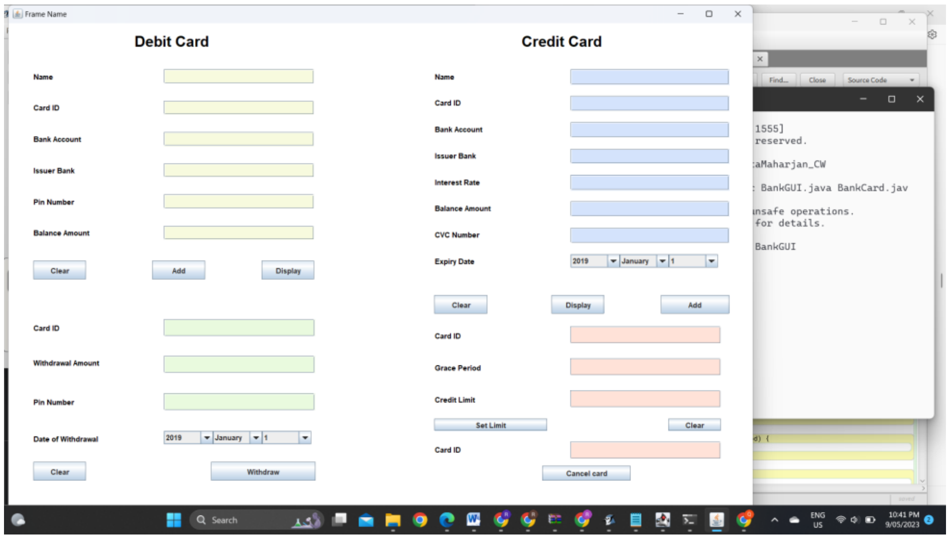
Objective	To test that the program can be compiled and run using the command prompt.
Expected result	The program should be compiled and run through command prompt.
Actual result	The program was successfully run and compiled through command prompt.
Conclusion	The test is successful.

Table 6: Test 6 - To test that the program can be compiled using command prompt.



```
Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rajit>cd C:\Users\rajit\RajitaMaharjan_CW

C:\Users\rajit\RajitaMaharjan_CW>javac BankGUI.java BankCard.jav
a DebitCard.java CreditCard.java
Note: BankGUI.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

C:\Users\rajit\RajitaMaharjan_CW>java BankGUI
```

Figure 11: Command Prompt



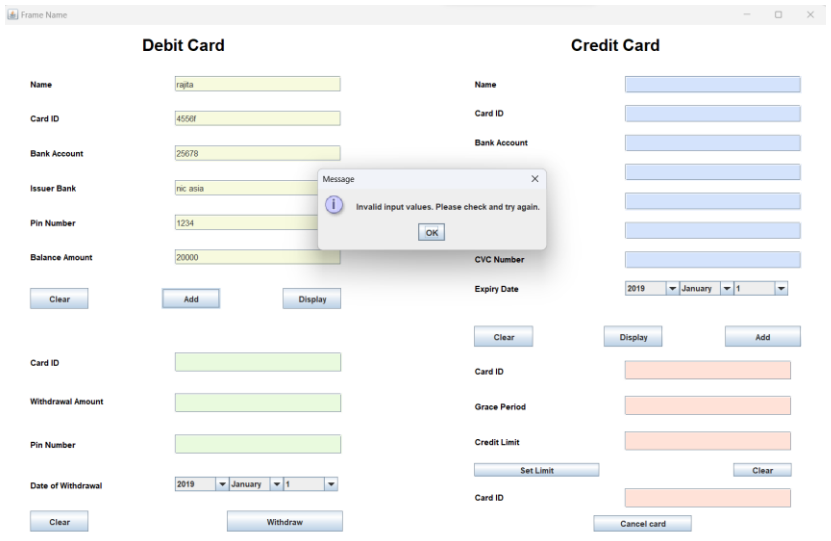
```
1555]
reserved.

aMaharjan_CW

: BankGUI.java BankCard.jav

unsafe operations.
for details.

BankGUI
```

Figure 12: Bank GUI

5.7 Test 7: To test that appropriate dialog boxes appear when unsuitable values are entered for the Card ID.

Objective	To test that appropriate dialog boxes appear when unsuitable values are entered for the Card ID.
Expected result	Dialog boxes should appear when unsuitable values are entered in Card ID.
Actual result	Dialog boxes appears when unsuitable values are entered in CardID.
Conclusion	The test is successful.

Table 7: Test 7 – To test appropriate dialog boxes appear when unsuitable values are entered for Card ID.



Debit Card

Name	rajita
Card ID	4556f
Bank Account	25678
Issuer Bank	nic asia
Pin Number	1234
Balance Amount	20000
Card ID	
Withdrawal Amount	
Pin Number	
Date of Withdrawal	2019 January 1

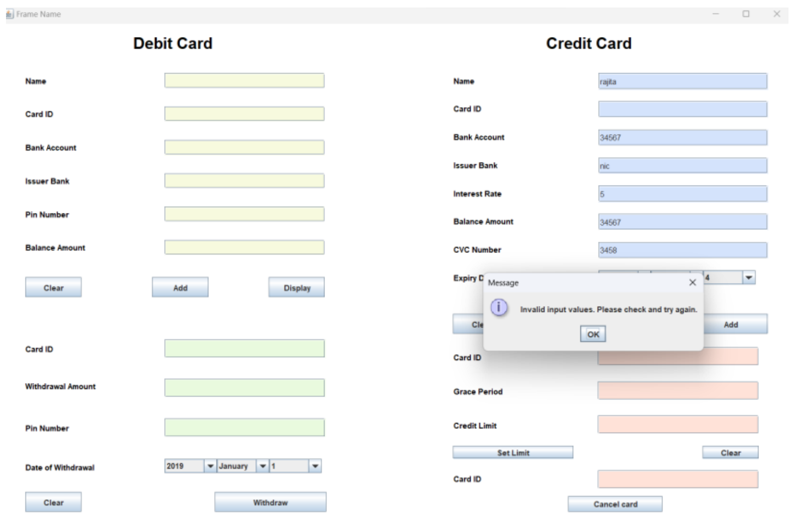
Credit Card

Name	
Card ID	
Bank Account	
CVC Number	
Expiry Date	2019 January 1
Card ID	
Grace Period	
Credit Limit	
Card ID	

Message

Invalid input values. Please check and try again.

Figure 13: Dialog box when inappropriate value is entered.



Debit Card

Name	
Card ID	
Bank Account	
Issuer Bank	
Pin Number	
Balance Amount	
Card ID	
Withdrawal Amount	
Pin Number	
Date of Withdrawal	2019 January 1

Credit Card

Name	rajita
Card ID	
Bank Account	34567
Issuer Bank	nic
Interest Rate	5
Balance Amount	34567
CVC Number	3458
Expiry D	4
Card ID	
Grace Period	
Credit Limit	
Card ID	

Message

Invalid input values. Please check and try again.

Figure 14: Dialog box appears when text feild is empty.

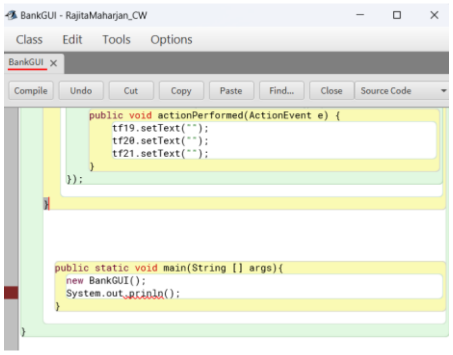
6. Error detection and correction:

6.1 Syntax Error:

Syntax errors are caught by a software program called a compiler, and the programmer must fix them before the program is compiled and then run.

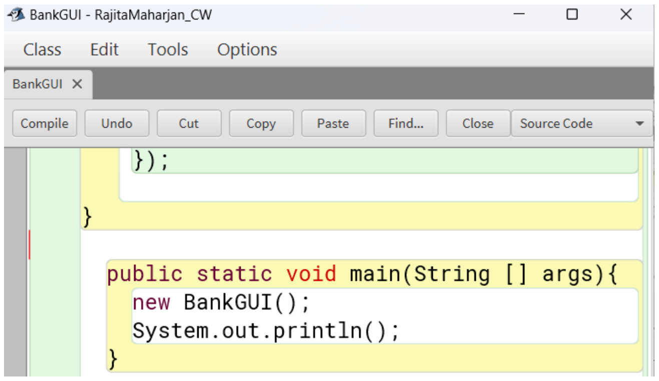
The following figure shows a syntax error, where semicolon (;) is missing. It occurs when the language used by a user is incorrect.

These type of errors are the most common type of errors in programming.


BankGUI - RajitaMaharjan_CW
Class Edit Tools Options
BankGUI
Compile Undo Cut Copy Paste Find... Close Source Code

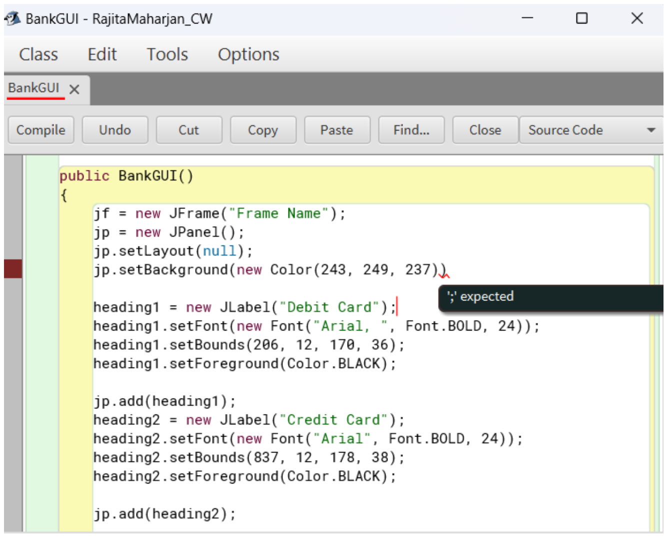
```
            public void actionPerformed(ActionEvent e) {
                tf19.setText("");
                tf20.setText("");
                tf21.setText("");
            }
        });

    }

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

Figure 15: Syntax Error


BankGUI - RajitaMaharjan_CW
Class Edit Tools Options
BankGUI
Compile Undo Cut Copy Paste Find... Close Source Code

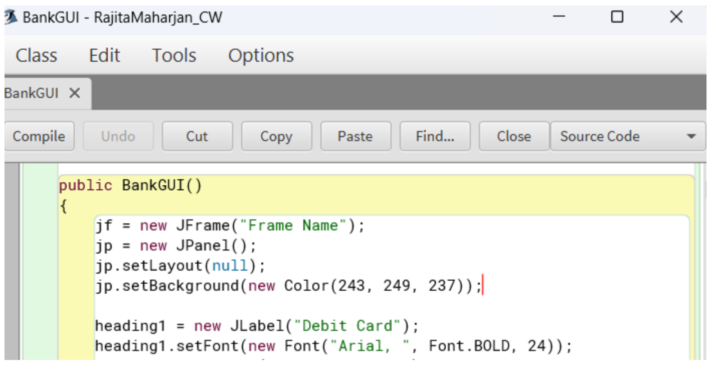
```
        });

    }

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

Figure 16: Syntax error corrected


BankGUI - RajitaMaharjan_CW
Class Edit Tools Options
BankGUI
Compile Undo Cut Copy Paste Find... Close Source Code
';' expected

```
public BankGUI()
{
    jf = new JFrame("Frame Name");
    jp = new JPanel();
    jp.setLayout(null);
    jp.setBackground(new Color(243, 249, 237))

    heading1 = new JLabel("Debit Card");
    heading1.setFont(new Font("Arial, ", Font.BOLD, 24));
    heading1.setBounds(206, 12, 170, 36);
    heading1.setForeground(Color.BLACK);

    jp.add(heading1);
    heading2 = new JLabel("Credit Card");
    heading2.setFont(new Font("Arial", Font.BOLD, 24));
    heading2.setBounds(837, 12, 178, 38);
    heading2.setForeground(Color.BLACK);

    jp.add(heading2);
```

Figure 17: Syntax Error semicolon(;)


BankGUI - RajitaMaharjan_CW
Class Edit Tools Options
BankGUI
Compile Undo Cut Copy Paste Find... Close Source Code

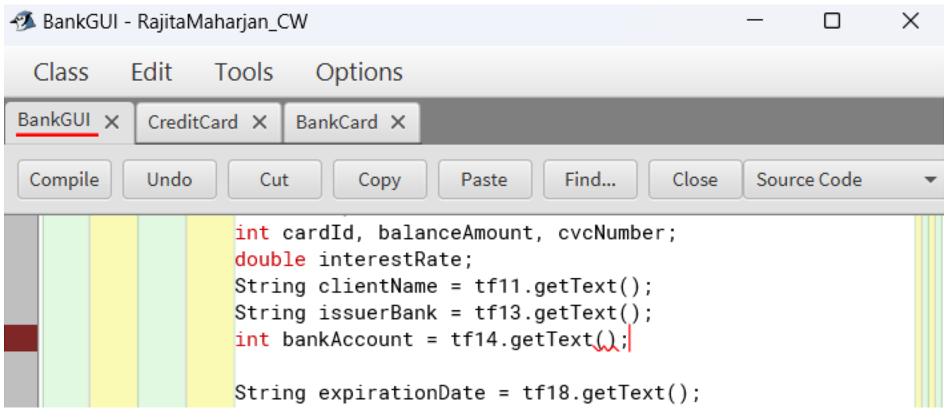
```
public BankGUI()
{
    jf = new JFrame("Frame Name");
    jp = new JPanel();
    jp.setLayout(null);
    jp.setBackground(new Color(243, 249, 237));

    heading1 = new JLabel("Debit Card");
    heading1.setFont(new Font("Arial, ", Font.BOLD, 24));
```

Figure 18: Syntax error corrected.

6.2 Semantics Error:

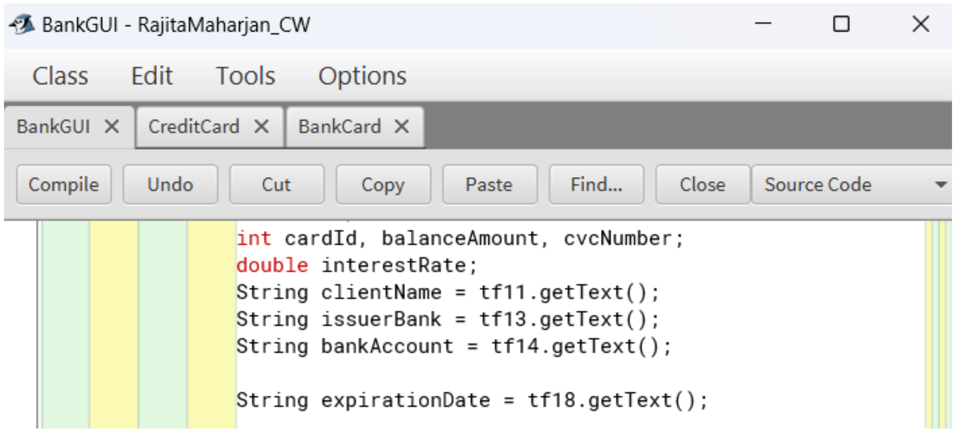
Semantics is a linguistic concept different from syntax, which is also related to the attributes of programming languages. Semantics holds that the linguistic representations or symbols.

The following figure is semantics error.



```
int cardId, balanceAmount, cvcNumber;
double interestRate;
String clientName = tf11.getText();
String issuerBank = tf13.getText();
int bankAccount = tf14.getText();

String expirationDate = tf18.getText();
```

Figure 19: Semantics Error



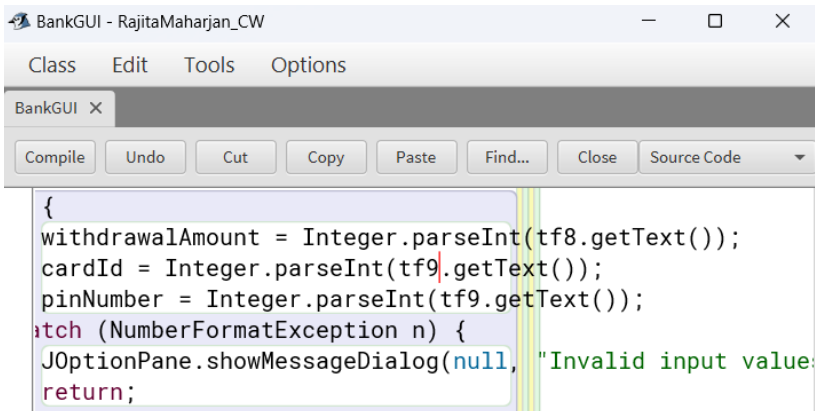
```
int cardId, balanceAmount, cvcNumber;
double interestRate;
String clientName = tf11.getText();
String issuerBank = tf13.getText();
String bankAccount = tf14.getText();

String expirationDate = tf18.getText();
```

Figure 20: Semantics error corrected.

6.3 Logical Error:

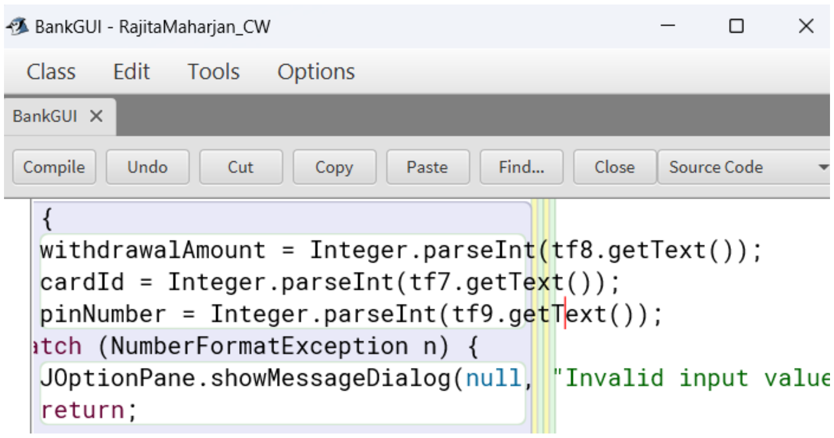
Logical error is an error in a program's source code that causes unexpected and improper action. A logic error is a sort of runtime error that can cause a program to provide incorrect output. It may also cause the program to crash while running.

Logical errors are not always simple to identify right away. This is because, unlike syntax faults, such errors are correct when examined in the language yet do not achieve the intended result. These may occur in both compiled and interpreted languages.



```
{
withdrawalAmount = Integer.parseInt(tf8.getText());
cardId = Integer.parseInt(tf9.getText());
pinNumber = Integer.parseInt(tf9.getText());
atch (NumberFormatException n) {
JOptionPane.showMessageDialog(null, "Invalid input value
return;
```

Figure 21: Logical Error



```
{
withdrawalAmount = Integer.parseInt(tf8.getText());
cardId = Integer.parseInt(tf7.getText());
pinNumber = Integer.parseInt(tf9.getText());
atch (NumberFormatException n) {
JOptionPane.showMessageDialog(null, "Invalid input value
return;
```

Figure 22: Logical Error corrected.

7. Conclusion

A graphical user interface (GUI) for the Bank Card system has been made in this coursework. The BankGUI class has been successfully created, together with the BankCard, DebitCard, and CreditCard classes, to store and manage debit and credit card information using an ArrayList.

The GUI incorporates numerous components, including as text fields, combo boxes, and buttons, to create a user-friendly interface for adding debit and credit cards, withdrawing amounts from debit cards, setting credit limits, and canceling credit cards. try-catch blocks, have been implemented to enable correct input validation and display appropriate error messages.

While working on the coursework, we noticed various issues and faults, including grammatical, logical, and semantic errors. Notes and other resources from lecture, tutorial, and workshop classes are easily accessible to all students via mySecondTeacher, which helped us in gaining additional programming knowledge and resolving mistakes we faced while completing the assignment.

In conclusion, the Bank Card GUI system displays effective implementation, improved Java programming and GUI development abilities, and the capacity to overcome problems through problem-solving strategies. This course has truly helped us to understand the fundamentals of java programming and create a foundation for future learning.

8. References

Bluej. (2023). Retrieved from wikipedia: https://en.wikipedia.org/wiki/BlueJ

Semantic Errors (Syntax Parsing Engine). (2023). Retrieved from INFRAGISTICS: https://www.infragistics.com/help/wpf/ig-spe-semantic-errors

What is Java? (2023). Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/java/

Contributor, T. (1999-2023). TechTarget. *pseudocode*, https://www.techtarget.com.

GeeksforGeeks. (2023). *GeeksforGeeks.* Retrieved from GeeksforGeeks.: https://www.geeksforgeeks.org/java/

Hope, C. (2023). *Draw.io*. Retrieved from Computer Hope: https://www.computerhope.com/jargon/d/drawio.htm

Rouse, M. (2023). *Logic Error*. Retrieved from Techopedia: https://www.techopedia.com/definition/8122/logic-error

Rouse, M. (2023). *Syntax Error*. Retrieved from techopedia: https://www.techopedia.com/definition/13391/syntax-error#:~:text=10%20February%2C%202017-,What%20Does%20Syntax%20Error%20Mean%3F,is%20compiled%20and%20then%20run.

9. Appendix:

```
public class BankGUI
{
    JFrame jf;
    JPanel jp;
    JTextField tf1, tf2, tf3, tf4,tf5, tf6, tf7, tf8, tf9, tf11, tf12, tf13, tf14, tf15, tf16, tf17,
tf19, tf20, tf21, tf22;
    JButton jb1, jb2, jb3, jb4, jb5, jb6, jb7, jb8, jb9, jb10, jb11;
    JLabel lb1, lb2, lb3, lb4, lb5, lb6, lb7, lb8, lb9, lb10, lb11, lb12, lb13, lb14, lb15,
lb16, lb17, lb18, lb19, lb20, lb21, lb22, heading1, heading2;
    JComboBox cb1, cb2, cb3, cb4, cb5, cb6;
    public BankGUI()
    {
        jf = new JFrame("Frame Name");
        jp = new JPanel();
        jp.setLayout(null);
        jp.setBackground(Color.WHITE);

        heading1 = new JLabel("Debit Card");
        heading1.setFont(new Font("Arial, ", Font.BOLD, 24));
        heading1.setBounds(206, 12, 170, 36);
        heading1.setForeground(Color.BLACK);

        jp.add(heading1);
        heading2 = new JLabel("Credit Card");
        heading2.setFont(new Font("Arial", Font.BOLD, 24));
        heading2.setBounds(837, 12, 178, 38);
        heading2.setForeground(Color.BLACK);

        jp.add(heading2);


        jf.add(jp);


        tf1 = new JTextField();
        tf2 = new JTextField();
        tf3 = new JTextField();
        tf4 = new JTextField();
        tf5 = new JTextField();
        tf6 = new JTextField();
```

```
tf7 = new JTextField();
tf8 = new JTextField();
tf9 = new JTextField();
tf11 = new JTextField();
tf12 = new JTextField();
tf13 = new JTextField();
tf14 = new JTextField();
tf15 = new JTextField();
tf16 = new JTextField();
tf17 = new JTextField();
tf19 = new JTextField();
tf20 = new JTextField();
tf21 = new JTextField();
tf22 = new JTextField();


tf1.setBounds(254, 76, 245, 23);
tf1.setBackground(new Color(247, 250, 222));

tf2.setBounds(254, 127, 245, 23);
tf2.setBackground(new Color(247, 250, 222));

tf3.setBounds(254, 178, 245, 23);
tf3.setBackground(new Color(247, 250, 222));

tf4.setBounds(254, 229, 245, 23);
tf4.setBackground(new Color(247, 250, 222));

tf5.setBounds(254, 280, 245, 23);
tf5.setBackground(new Color(247, 250, 222));

tf6.setBounds(254, 331, 245, 23);
tf6.setBackground(new Color(247, 250, 222));

tf7.setBounds(254, 483, 246, 28);
tf7.setBackground(new Color(233, 250, 222));

tf8.setBounds(254, 543, 246, 28);
tf8.setBackground(new Color(233, 250, 222));

tf9.setBounds(254, 604, 246, 28);
tf9.setBackground(new Color(233, 250, 222));
```

```
tf11.setBounds(917, 76, 259, 25);
tf11.setBackground(new Color(212, 227, 252));

tf12.setBounds(917, 119, 259, 25);
tf12.setBackground(new Color(212, 227, 252));

tf13.setBounds(917, 162, 259, 25);
tf13.setBackground(new Color(212, 227, 252));

tf14.setBounds(917, 205, 259, 25);
tf14.setBackground(new Color(212, 227, 252));

tf15.setBounds(917, 248, 259, 25);
tf15.setBackground(new Color(212, 227, 252));

tf16.setBounds(917, 291, 259, 25);
tf16.setBackground(new Color(212, 227, 252));

tf17.setBounds(917, 334, 259, 25);
tf17.setBackground(new Color(212, 227, 252));

tf19.setBounds(916, 495, 246, 28);
tf19.setBackground(new Color(255, 226, 216));

tf20.setBounds(916, 547, 246, 28);
tf20.setBackground(new Color(255, 226, 216));

tf21.setBounds(916, 599, 246, 28);
tf21.setBackground(new Color(255, 226, 216));

tf22.setBounds(916, 683, 246, 28);
tf22.setBackground(new Color(255, 226, 216));



jp.add(tf1);
jp.add(tf2);
jp.add(tf3);
jp.add(tf4);
jp.add(tf5);
jp.add(tf6);
jp.add(tf7);
jp.add(tf8);
jp.add(tf9);
jp.add(tf11);
jp.add(tf12);
jp.add(tf13);
jp.add(tf14);
jp.add(tf15);
jp.add(tf16);
jp.add(tf17);
jp.add(tf19);
jp.add(tf20);
jp.add(tf21);
jp.add(tf22);

lb1 = new JLabel("Name");
lb1.setForeground(Color.BLACK);

lb2 = new JLabel("Card ID");
lb2.setForeground(Color.BLACK);

lb3 = new JLabel("Bank Account");
lb3.setForeground(Color.BLACK);

lb4 = new JLabel("Issuer Bank ");
lb4.setForeground(Color.BLACK);

lb5 = new JLabel("Pin Number");
lb5.setForeground(Color.BLACK);

lb6 = new JLabel("Balance Amount");
lb6.setForeground(Color.BLACK);

lb7 = new JLabel("Card ID");
lb7.setForeground(Color.BLACK);

lb8 = new JLabel("Withdrawal Amount");
lb8.setForeground(Color.BLACK);

lb9 = new JLabel("Pin Number");
lb9.setForeground(Color.BLACK);

lb10 = new JLabel("Date of Withdrawal");
lb10.setForeground(Color.BLACK);
```

```
lb11 = new JLabel("Name");
lb11.setForeground(Color.BLACK);

lb12 = new JLabel("Card ID");
lb12.setForeground(Color.BLACK);

lb13 = new JLabel("Bank Account");
lb13.setForeground(Color.BLACK);

lb14 = new JLabel("Issuer Bank");
lb14.setForeground(Color.BLACK);

lb15 = new JLabel("Interest Rate");
lb15.setForeground(Color.BLACK);

lb16 = new JLabel("Balance Amount");
lb16.setForeground(Color.BLACK);

lb17 = new JLabel("CVC Number");
lb17.setForeground(Color.BLACK);

lb18 = new JLabel("Expiry Date");
lb18.setForeground(Color.BLACK);

lb19 = new JLabel("Card ID");
lb19.setForeground(Color.BLACK);

lb20 = new JLabel("Grace Period");
lb20.setForeground(Color.BLACK);

lb21 = new JLabel("Credit Limit");
lb21.setForeground(Color.BLACK);

lb22 = new JLabel("Card ID");
lb22.setForeground(Color.BLACK);


lb1.setBounds(41, 76, 105, 22);
lb2.setBounds(41, 127, 105, 22);
lb3.setBounds(41, 178, 105, 22);
lb4.setBounds(41, 229, 105, 22);
lb5.setBounds(41, 280, 105, 22);
```

```
lb6.setBounds(41, 331, 105, 22);
lb7.setBounds(41, 483, 106, 28);
lb8.setBounds(41, 543, 160, 22);
lb9.setBounds(41, 604, 106, 28);
lb10.setBounds(41, 665, 146, 27);
lb11.setBounds(695, 76, 112, 23);
lb12.setBounds(695, 119, 112, 23);
lb13.setBounds(695, 162, 112, 23);
lb14.setBounds(695, 205, 112, 23);
lb15.setBounds(695, 248, 112, 23);
lb16.setBounds(695, 291, 112, 23);
lb17.setBounds(695, 334, 112, 23);
lb18.setBounds(695, 377, 112, 23);
lb19.setBounds(695, 495, 113, 31);
lb20.setBounds(695, 547, 113, 31);
lb21.setBounds(695, 599, 113, 31);
lb22.setBounds(695, 680, 113, 31);



jp.add(lb1);
jp.add(lb2);
jp.add(lb3);
jp.add(lb4);
jp.add(lb5);
jp.add(lb6);
jp.add(lb7);
jp.add(lb8);
jp.add(lb9);
jp.add(lb10);
jp.add(lb11);
jp.add(lb12);
jp.add(lb13);
jp.add(lb14);
jp.add(lb15);
jp.add(lb16);
jp.add(lb17);
jp.add(lb18);
jp.add(lb19);
jp.add(lb20);
jp.add(lb21);
jp.add(lb22);
```

```
        jb1 = new JButton("Clear");
        jb2 = new JButton("Add");
        jb3 = new JButton("Display");
        jb4 = new JButton("Withdraw");
        jb5 = new JButton("Clear");
        jb6 = new JButton("Display");
        jb7 = new JButton("Add");
        jb8 = new JButton("Set Limit");
        jb9 = new JButton("Cancel card");
        jb10 = new JButton("Clear");
        jb11 = new JButton("Clear");


        jb1.setBounds(41, 388, 86,30 );
        jb2.setBounds(235, 388, 86, 30);
        jb3.setBounds(413, 388, 86, 30);
        jb4.setBounds(331, 716, 170, 30);
        jb5.setBounds(695, 444, 86, 30);
        jb6.setBounds(886, 444, 86, 30);
        jb7.setBounds(1064, 444, 112, 30);
        jb8.setBounds(695, 645, 184, 20);
        jb9.setBounds(871, 722, 144, 24);
        jb10.setBounds(41, 716, 86, 30);
        jb11.setBounds(1076, 645, 86, 20);

        jp.add(jb1);
        jp.add(jb2);
        jp.add(jb3);
        jp.add(jb4);
        jp.add(jb5);
        jp.add(jb6);
        jp.add(jb7);
        jp.add(jb8);
        jp.add(jb9);
        jp.add(jb10);
        jp.add(jb11);

        //cb1 will contain list of years
        String[] years = {"2019", "2020", "2021", "2022",
    "2023","2024","2025","2026","2027","2028","2029","2030"};
```

```
        String[] months = {"January", "February", "March",
    "April","May","June","July","August","September","October","November","December"
    };
        String[] days = {"1", "2", "3", "4", "5",
    "6","7","8","9","10","11","12","13","14","15","16","17","18","19","20",
            "21","22","23","24","25","26","27","28","29","30","31"};
        cb1 = new JComboBox(years);
        cb2 = new JComboBox(months);
        cb3 = new JComboBox(days);
        cb4 = new JComboBox(years);
        cb5 = new JComboBox(months);
        cb6 = new JComboBox(days);

        cb1.setBounds(917, 377, 80, 22);
        cb2.setBounds(997, 377, 80, 22);
        cb3.setBounds(1077, 377, 80, 22);
        cb4.setBounds(254, 665, 80, 22);
        cb5.setBounds(334, 665, 80, 22);
        cb6.setBounds(414, 665, 80, 22);

        jp.add(cb1);
        jp.add(cb2);
        jp.add(cb3);
        jp.add(cb4);
        jp.add(cb5);
        jp.add(cb6);

        jp.setVisible(true);
        jp.setSize(500,900);
        jf.setSize(1228, 1080);
        jf.setVisible(true);
        jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        ArrayList<BankCard> bankCards = new ArrayList<BankCard>();
        //add debit card
        jb2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // Get input values from GUI

                int balanceAmount;
                int cardId;
                int pinNumber;
                String bankAccount = tf3.getText();
                String issuerBank = tf4.getText();
                String clientName = tf1.getText();
                try {
                    balanceAmount = Integer.parseInt(tf6.getText());
                    cardId = Integer.parseInt(tf2.getText());
                    pinNumber = Integer.parseInt(tf5.getText());
                } catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please
check and try again.");
                    return;
                }

                // Create new DebitCard object with input values
                DebitCard debitCard = new DebitCard(balanceAmount, cardId,
bankAccount, issuerBank, clientName, pinNumber);

                // Add new DebitCard object to ArrayList of BankCard objects
                bankCards.add(debitCard);
                JOptionPane.showMessageDialog(null, "Debit card added successfully.");
            }
        });

        // clear values from debit card fields
        jb1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                tf1.setText("");
                tf2.setText("");
                tf3.setText("");
                tf4.setText("");
                tf5.setText("");
                tf6.setText("");
            }
        });
        //debit card display
        jb3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                int cardId;
                String clientName = tf1.getText();
                try{
                    cardId = Integer.parseInt(tf2.getText());
                }catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please
check and try again.");
```

```
                    return;
                }

                DebitCard debitCard = null;
                for (BankCard card : bankCards) {
                    if (card instanceof DebitCard &&   card.getcardID() == cardId) {
                        debitCard = (DebitCard) card;
                        debitCard.setclientName(clientName);
                        break;
                    }
                }

            // If a matching DebitCard object is found, display its information
                if (debitCard != null) {
                    JOptionPane.showMessageDialog(null, debitCard.display());
                } else {
                    JOptionPane.showMessageDialog(null, "Debit card not found");
                }
            }
        });
        //withdraw
        jb4.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String dateOfWithdrawal=cb4.getSelectedItem().toString()+"-
    "+cb5.getSelectedItem().toString()+"-"+cb6.getSelectedItem().toString();
                int cardId;
                int pinNumber;
                int withdrawalAmount;
                try {
                    withdrawalAmount = Integer.parseInt(tf8.getText());
                    cardId = Integer.parseInt(tf7.getText());
                    pinNumber = Integer.parseInt(tf9.getText());
                } catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please
    check and try again.");
                    return;
                }

                DebitCard debitCard = null;
                for (BankCard bankCard : bankCards) {
                    if (bankCard instanceof DebitCard && bankCard.getcardID() == cardId)
    {
                        debitCard = (DebitCard) bankCard;
                        break;
                    }
                }
                if (debitCard != null) {
                    String message =
    debitCard.Withdraw(withdrawalAmount,dateOfWithdrawal,pinNumber);
                    JOptionPane.showMessageDialog(null, message);
                } else {
                    JOptionPane.showMessageDialog(null, "Debit card not found");
                }

            }
        });


        // clear values from debit card fields
        jb5.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                tf11.setText("");
                tf12.setText("");
                tf13.setText("");
                tf14.setText("");
                tf15.setText("");
                tf16.setText("");
                tf17.setText("");
                cb1.setSelectedIndex(0);
                cb2.setSelectedIndex(0);
                cb3.setSelectedIndex(0);
            }
        });
        //display credit card
        jb6.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                int cardId;
                String clientName = tf11.getText();
                try{
                    cardId = Integer.parseInt(tf12.getText());
                }catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please
    check and try again.");
                    return;
                }
```

```
                CreditCard creditCard = null;
                for (BankCard card : bankCards) {
                    if (card instanceof CreditCard && card.getcardID() == cardId) {
                        creditCard = (CreditCard) card;
                        break;
                    }
                }

                // If a matching DebitCard object is found, display its information
                if (creditCard != null) {
                    creditCard.setclientName(clientName);
                    JOptionPane.showMessageDialog(null, creditCard.display());
                } else {
                    JOptionPane.showMessageDialog(null, "Credit card not found");
                }
            }
        });
        //add credit card
        jb7.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // Get inputs from text fields
                int cardId, balanceAmount, cvcNumber;
                double interestRate;
                String clientName = tf11.getText();
                String issuerBank = tf13.getText();
                String bankAccount = tf14.getText();

                String expirationDate = cb1.getSelectedItem().toString() + "-" +
cb2.getSelectedItem().toString() + "-" + cb3.getSelectedItem().toString();

                try {
                    cardId = Integer.parseInt(tf12.getText());
                    balanceAmount = Integer.parseInt(tf16.getText());
                    cvcNumber = Integer.parseInt(tf17.getText());
                    interestRate = Double.parseDouble(tf15.getText());
                } catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please
check and try again.");
                    return;
                }

                // Create a new CreditCard object
                CreditCard creditCard = new CreditCard(balanceAmount, clientName, 
    bankAccount, issuerBank, cardId, cvcNumber, interestRate, expirationDate);

                // Add the new CreditCard object to the array list of BankCard class
                bankCards.add(creditCard);

                // Display a success message
                JOptionPane.showMessageDialog(null, "Credit card added successfully");
            }
        });

        //set limit to credit card
        jb8.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // get the input values
                int cardId,creditLimit,gracePeriod;
                try{
                     cardId = Integer.parseInt(tf19.getText());
                     creditLimit = Integer.parseInt(tf21.getText());
                     gracePeriod = Integer.parseInt(tf20.getText());
                }catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please 
    check and try again.");
                    return;
                }


                // find the credit card with the given card ID
                CreditCard creditCard = null;
                for (BankCard bankCard : bankCards) {
                    if (bankCard instanceof CreditCard && bankCard.getcardID() == cardId) 
    {
                        creditCard = (CreditCard) bankCard;
                        break;
                    }
                }

                // update the credit limit if a valid credit card is found
                if (creditCard != null) {
                    creditCard.setcreditLimit(creditLimit,gracePeriod);

                    // show a message dialog with the updated credit limit and grace period
                    JOptionPane.showMessageDialog(null, "Credit limit updated to " +
    creditLimit
                        + " and grace period updated to " + gracePeriod + ".");
                } else {
                    // show an error message if the card ID is invalid
                    JOptionPane.showMessageDialog(null, "Invalid card ID.");
                }
            }
        });
        //cancel credit card
        jb9.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                int cardId;
                try{
                    cardId = Integer.parseInt(tf22.getText());
                }catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Invalid input values. Please
    check and try again.");
                    return;
                }
                boolean foundCard = false;
                for (BankCard card : bankCards) {
                    if (card.getcardID() == cardId) {
                        foundCard = true;
                        // Cast the BankCard object as a CreditCard object
                        CreditCard creditCard = (CreditCard) card;
                        creditCard.cancelcreditCard();
                        JOptionPane.showMessageDialog(null, "Credit card " + cardId + "
    has been cancelled.");
                        break;
                    }
                }
                if (!foundCard) {
                    JOptionPane.showMessageDialog(null, "Invalid card ID.");
                }
            }
        });
        // clear fields related to withdraw
        jb10.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                tf7.setText("");
                tf8.setText("");
                tf9.setText("");
```

```
                cb4.setSelectedIndex(0);
                cb5.setSelectedIndex(0);
                cb6.setSelectedIndex(0);
            }
        });
        //clear fields related to set limit
        jb11.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                tf19.setText("");
                tf20.setText("");
                tf21.setText("");
            }
        });

    }
      public static void main(String [] args){
        new BankGUI();
        System.out.println();
      }

}
```