Module Code & Module Title:
CS4001NT Programming

Assessment Weightage & Type:
30% Individual Coursework

Year and Semester:
2022 Autumn

Student Name: Anjan Khadka
London Met ID: 2207082
College ID: NP05CP4A220018
Assignment Due Date: 2023-May-10
Word Count: 13619

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Introduction to Java

Java is an object oriented programming language which was first developed by James Gosling at Sun Microsystems which is now a part of Oracle Corporation. Java is also used as a computing platform. Java is general purpose programming language which is used for application development as it is secure, fast and reliable. It is being used widely across many platforms as a mean to develop java applications in laptops, game consoles, mobile phone etc. It is popular as it is one of the easy programming language to learn and use and it also supports multi-platform. The java platforms component like Java Development kit (JDK) which is a development environment for building used to write a java program and Java Runtime Environment (JRE) which is needed to run a java program, helps a programmer learn and make programs in easier and faster way. To run a java program, it should be run through compiler and then assembler.

Java programming language consist of class which has multiple methods in it. Every method has its specific work which can be called to run. The main method is a heart of all methods as it is used to call other methods by making an object i.e. real world entity. While initializing value in program we need to declare a variable with its specific data type. As java is case sensitive the name of methods and variable should have a systematic rule while declaring

Java has great portability. The same Java application will function equally on any computer, regardless of its hardware or operating system, as long as it has a Java interpreter. In addition to portability, Java has a number of security features that shield a computer running a Java software from malware (like viruses) as well as issues brought on by incorrect programming. Because Java's security mechanisms prevent these applets from accessing a PC's hard drive or network connections, Java applets obtained from the Internet can be safely launched. An HTML page will frequently contain an applet, a brief Java program. Java can be thought of as both a compiled language and a bytecode language because its source code is first converted to binary Java is a language that can be both compiled and interpreted. The Java Virtual Machine (JVM), a software-based interpreter, runs this bytecode. The fact that Java is an open standard with open-source code is another distinctive quality of the language. Although Sun Microsystems controls the Java language and the tools that go with it, the Internet community has accepted Java as a standard thanks to Sun's permissive license policy. (Hartman, 2022)

1.1 Introduction to GUI

Graphical User Interface (GUI) is a visual representation of communication that is provided for user to have simple interactions with the machine. It contains typical graphical representations like buttons and icons through which communication can be achieved rather than traditional command line based communication. To achieve such visually in java we can use swing and AWT.

Swing is set of interfaces and class that handle wide range of visual components including text fields, labels, buttons, check boxes and many more and with the proper combination of these we can accomplish a GUI with good graphical interfaces. At the beginning in the Java's GUI swing was not available so there was use of Abstract Window Toolkit (AWT). The AWT defines a fundamental collection of components that provide a functional but limited graphical interface. Java Swing, in contrast to AWT, provides platform independent and lightweight components. Also using the Listener interfaces we can have a specific command be done while clicking with mouse or clicking buttons or menu items.

1.3 Introduction to Project

This project acts as a GUI for the 1st coursework java program where we have made use of variables, arrays, along with the concept of OOP and made a working program for the Bank Cards where Credit Card and Debit Card were inheriting the properties of Bank Card class. Using the concept of GUI we were told to make a well working GUI which can handle the methods used in first coursework and the data places on text fields will be used as the variables for previous program and give the required output on both terminal and dialog box.

2. Class Diagram

2.1 Introduction to class diagram

UML class diagrams are a way to show the parts and connections of classes. They can show a class's variables and functions, as well as if it has a relationship with another class. We can see how the source code relies on each other. It's easier to understand the structure of a system using a diagram instead of reading the source code. We can see certain patterns in diagrams, like when classes depend on each other in a loop. We can also see when abstract classes rely on concrete ones and come up with a way to fix the problem. The most common way to show a class is with its name. There are also compartments in the class icon for variables and functions. We use symbols like +, -, and # to show if they are public, private, or protected. After the colon, we can see the type of the variable or argument name and a function's return value. It's not necessary to include everything in the UML diagram, just the important parts. We should keep variable and function declarations in the source code and only use extra details in the diagram when they help us understand the system better. (JavaTPoint, n.d.)

2.2 Class Diagram of BankGUI

BankGUI
- frame: **JFrame**
- pnlBC : **JPanel** - welcomeLabel, fillLabel, balanceAmtLabl, issuerLabel, bankAccLabel, clientNmLabel : JLabel - BAmtf, IBtf, BActf, CNatf : **JTextField** - CredBCButton,DebBCutton : **JButton**
- pnlDC : JPanel - DC_Add_IDtf,DCIDtf,PNtf,PNWtf,WAmtf : **JTextField** - WYears,WMonths,WDays : **JComboBox<String>** - addDebCardLabel, pinNumLabel, withAmtLabel, DOW_Label, wthCardIdLabel, wthPinNumLabel : **JLabel** - withdrawButton,addDebitCardButton,credDfButton,bankDfButton : **JButton**
- pnlCC : **JPanel** - CC_Add_CIDtf, CVCtf,IRtf,CLtf, GPtf,CCIDtf, CancelCreditIDtf **: JTextField** - EYears,EMonths,EDays : **JComboBox<String>** - addCredCardidLabel, cvcNumLabel,interestLabel, DOE_Label, credLimitLabel, graceLabel, setCreLim_CId_label, cancelCC_CId_Label : **JLabel** - addCreditCardButton, setCreditLimiButton, cancelCCButton, debCfButton, bnkCfButton, bankCardClearButton, debitDisplayButton, debitClearButton, creditDisplyButton, creditClearButton : **JButton**
+ INVALID : int
- cardList : ArrayList<BankCard>
- years[] : String - months[] : String - days[] : String
+ BankGUI() : void + initFrame(): void + initBankCard(): void + initDebitCard(): void + initCreditCard(): void + clearButton(Container) : void + addDebitCard(): void

+ withdrawal() : void + addCreditCard : void + cancelCreditCard : void + checkDebitCardUnique : boolean + checkCreditCardUnique : boolean + addCreditLimit :void + Display : void + getBalanceAmount : int + getIssuerBank() : String + getBankAccount() : String + getClientName() : String + getAddCardIDCredit() : int + getCVCNumber() : int + getInterestRate() : int + getExpirationDate() : String + getCreditLimit() : int + getCardIDCredit() : int + getGracePeriod() : int + getCancelCreditCardID() : int + getPinNumber() : int + getAddDebitCardId() : int + getWithdrawalPinNumber() : int + getWithdrawalAmount() ; int + getDateOfWithdrawal() : String + getDebitCardId() : int + main(String[] args) : void

Figure 1: Class Diagram of BankGUI

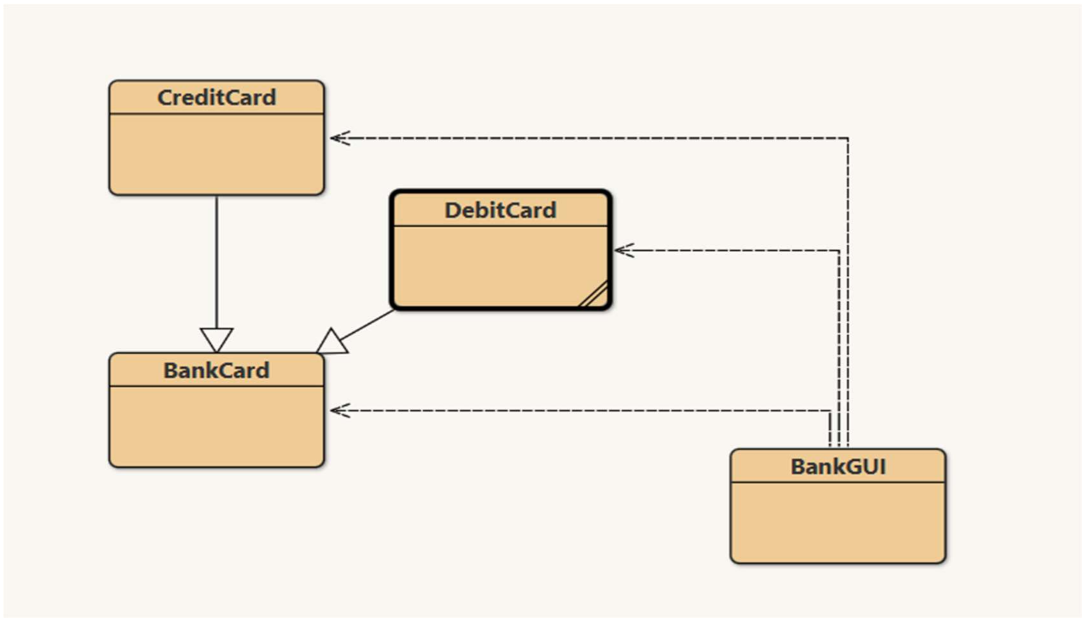


Figure 1: Class Diagram from BLUEJ

3. Pseudo code

The use of pseudocode, as opposed to a particular programming language, allows for the expression of computer programs. It's a tool used to conceptualize and create software before any code is ever written.

Consider it a recipe for a dish. Pseudocode outlines the procedures needed to run a program, much like a cookbook would when outlining how to prepare a food. However, pseudocode employs broader notions and ideas as opposed to particular elements.

For example, a pseudocode statement might look like this:

```
IF user has entered a valid password THEN
  allow access to the system
ELSE
  display an error message
ENDIF
```

This statement describes a decision-making process: The system should grant access if the user entered a valid password; otherwise, it should display an error message. This statement illustrates a decision-making process.

Developers can lay out a program's logic using pseudocode without becoming caught down in the specifics of a given programming language. Before the software is ever developed, this makes sure it functions as anticipated, saving time and minimizing errors.

3.1 Pseudo code for constructor of BankGUI

```
FUNCTION
    CALL initFrame():
    CALL initBankCard():
    CALL initDebitCard():
    CALL initCreditCard():
END FUNCTION
```

3.2 Pseudo code for method initFrame()

```
FUNCTION initFrame():
      CREATE object of JFrame named “frame”
```

```
        SET frame size to (900,900)
        SET frame layout to NULL
        SET frame resizable property to false
        SET frame location relative to NULL
        SET frame default close operation to EXIT_ON_CLOSE
END FUNCTION
```

3.2 Pseudo code for method initBankCard()

```
FUNCTION initBankCard():
        CREATE object of JPanel named “pnlBC”
        SET pnlBC layout to NULL
        SET pnlBC location to (50,30)
        SET pnlBC size to (800,750)
        SET pnlBC background color to Cyan
        CREATE object of TitledBorder named “borderbank”
        SET borderbank title justification to center
        SET borderbank title font to (“Arial”,Bold,”24)
        ADD borderbank in pnlBC
        ADD pnlBC in frame
        CREATE object of JLabel name “welcomeLabel“
        SET welcomeLabel bounds to (140,50,600,80)
        SET welcomeLabel foreground color to red
        SET welcomeLabel font to (“Futura”,Bold,45)
        ADD welcomeLabel to pnlBC

        CREATE object of JLabel name “fillLabel”
        SET fillLabel bounds to (70,470,600,80)
        SET fillLabel foreground color to red
        SET fillLabel font to (“Futura”,Bold,45)
        ADD fillLabel to pnlBC
```

```
CREATE object of JLabel named "balanceAmLabl"
SET balaneAmLabl font to ("Ariel Black",PLAIN,15)
SET balaneAmLabl bound to (70,130,120,50)
ADD balaneAmLabl to pnlBC

CREATE object of JTextField named "BAmtf"
SET BAmtf bounds to (200,130,170,50)
ADD BAmtf to pnlBC

CREATE object of JLabel named "issuerLabel"
SET issuerLabel font to ("Ariel Black",PLAIN,15)
SET issuerLabel bounds to (70,180,120 ,70)
ADD issuerLabel to pnlBC

CREATE object of JTextField named "IBtf"
SET IBtf bounds to (200,190,170,50)
ADD IBtf to pnlBC


CREATE object of JLabel named "bankAccLabel"
SET bankAccLabel font to ("Ariel Black",PLAIN,15)
SET bankAccLabel bounds to (70,240,120,70)
ADD bankAccLabel to pnlBC

CREATE object of JTextField named "BActf"
SET BActf bounds to (200,250,170,50)
ADD BActf to pnlBC

CREATE object of JLabel named "clientNmLabel"
SET clientNmLabel font to ("Ariel Black",PLAIN,15)
SET clientNmLabel bounds to (70,300,120,70)
```

```
    **ADD** clientNmLabel to pnlBC

    **CREATE** object of JTextField named “CNatf”
    **SET** CNatf bounds to (200,310,170,50)
    **ADD** CNatf to pnlBC

    **CREATE** object of JButton named “bankCardClearButton”
    **SET** bankCardClearButton bound to (300,680,100,50)
    **ADD** bankCardClearButton to pnlBC
    **ADD** ActionListener():
        **WHEN CLICKED**
            **CALL** clearButton with parameter (pnlBC)

    **CREATE** object of JButton named “CredBCButton”
    **SET** CredBCButton bound to (140,550,120,50)
    **ADD** ActionListener():
        **WHEN CLICKED**
            **SET** pnlCC visibility to true
            **SET** pnlBC visibility to false
    **ADD** CredBCButton to pnlBC

    **CREATE** object of JButton named “DebBCButton”
    **SET** DebBCButton bound to (300,550,120,50)
    **ADD** ActionListener():
        **WHEN CLICKED**
            **SET** pnlDC visibility to true
            **SET** pnlBC visibility to false
    **ADD** DebBCButton to pnlBC
**END FUNCTION**
```

3.3 Pseudo Code for initDebitCard()

```
FUNCTION initDebitCard():
        CREATE object of JPanel named "pnlDC"
        SET pnlDC layout to NULL
        SET pnlDC location to (50,30)
        SET pnlDC size to (800,750)
        SET pnlDC background color to GREEN
        CREATE object of TitledBorder named "borderDebit"
        SET borderDebit title justification to center
        SET borderDebit title font to ("Arial",Bold,"24)
        ADD borderDebit in pnlDC
        ADD pnlDC in frame

        CREATE object of JLabel named "pinNumLabel"
        SET pinNumLabel bounds to (50,50,120,70)
        SET pinNumLabel font to ("Ariel Black",PLAIN,15)
        ADD pinNumLabel to pnlDC

        CREATE object of JTextField named "PNtf"
        SET PNtf bounds to (170,60,170,50)
        ADD PNtf to pnlDC

        CREATE object of JLabel named "addDebCardLabel"
        SET addDebCardLabel bounds to (50,110,120,70)
        SET addDebCardLabel font to ("Ariel Black",PLAIN,15)
        ADD addDebCardLabel to pnlDC

        CREATE object of JTextField named "DC_Add_IDtf"
        SET DC_Add_IDtf bounds to (170,120,170,50)
        ADD DC_Add_IDtf to pnlDC
```

```
**CREATE** object of JLabel named “withAmtLabel“
**SET** withAmtLabel bounds to (50,280,120,70)
**SET** withAmtLabel font to (“Ariel Black”,PLAIN,15)
**ADD** withAmtLabel to pnlDC

**CREATE** object of JTextField named “WAmtf”
**SET** WAmtf bounds to (170,290,170,50)
**ADD** WAmtf to pnlDC

**CREATE** object of JLabel named “DOW_Label“
**SET** DOW_Label bounds to (50,410,120,70)
**SET** DOW_Label font to (“Ariel Black”,PLAIN,15)
**ADD** DOW_Label to pnlDC

**CREATE** object of JComboBox<String> named “WYears” with parameter “years”
**SET** WYears bounds to (200,430,90,28)
**ADD** WYears to pnlDC

**CREATE** object of JComboBox<String> named “WMonths” with parameter
            “months”
**SET** WMonths bounds to (300,430,90,28)
**ADD** WMonths to pnlDC

**CREATE** object of JComboBox<String> named “WDays” with parameter “days”
**SET** WDays bounds to (400,430,90,28)
**ADD** WDays to pnlDC

**CREATE** object of JLabel named “wthCardIdLabel”
**SET** wthCardIdLabel text to “Card ID”
**SET** wthCardIdLabel bounds to (70,340,120,70)
**SET** wthCardIdLabel font to (“Ariel Black”,PLAIN,15)
```

```
ADD wthCardIdLabel to pnlDC

CREATE object of JTextField named "DCIDtf"
SET DCIDtf bounds to (170,350,170,50)
ADD DCIDtf to pnlDC

CREATE object of JLabel named "wthPinNumLabel"
SET wthPinNumLabel text to "Pin Number"
SET wthPinNumLabel bounds to (50,470,120,70)
SET wthPinNumLabel font to ("Ariel Black",PLAIN,15)
ADD wthPinNumLabel to pnlDC

CREATE object of JTextField named "PNWtf"
SET PNWtf bounds to (170,480,170,50)
ADD PNWtf to pnlDC

CREATE object of JLabel named "lblspam"
SET lblspam bounds to (10,230,600,50)
ADD lblspam to pnlDC

CREATE object of JButton named "withdrawButton"
SET withdrawButton text to "Withdraw"
SET withdrawButton bounds to (140, 580, 120, 50)
ADD ActionListener():
    WHEN CLICKED
            CALL withdrawal()
            ADD withdrawButton to pnlDC

CREATE object of JButton named "addDebitCardButton"
SET addDebitCardButton text to "Add Debit Card"
SET addDebitCardButton bounds to (170,180,170,50)
```

```
ADD ActionListener():
    WHEN CLICKED
        CALL addDebitCard()
ADD addDebitCardButton to pnlDC

CREATE object of JButton named “credDfButton”
SET credDfButton text to "Credit Card"
SET credDfButton bounds to (550,60,120,50)
ADD ActionListener():
    WHEN CLICKED
        SET pnlCC visibility to true
        SET pnlDC visibility to false
ADD credDfButton to pnlDC

CREATE object of JButton named “bankDfButton”
SET bankDfButton text to "Bank Card"
SET bankDfButton bounds to (550,150,120,50)
ADD ActionListener():
    WHEN CLICKED
        SET pnlBC visibility to true
        SET pnlDC visibility to false
ADD bankDfButton to pnlDC

CREATE object of JButton named “debitDisplayButton”
SET debitDisplayButton text to "Display"
SET debitDisplayButton bounds to (670, 600, 90, 50)
ADD ActionListener():
    WHEN CLICKED
        CALL Display()
ADD debitDisplayButton to pnlDC
```

```
          CREATE object of JButton named "debitClearButton"
          SET debitClearButton text to "Clear"
          SET debitClearButton bounds to (540,600,100,50)
          ADD ActionListener():
                    WHEN CLICKED
                              CALL clearButton with parameter (pnlDC)
          ADD debitClearButton to pnlDC
END FUNCTION
```

3.4 Pseudo Code for method initCreditCard()

```
FUNCTION initCreditCard():
          CREATE object of JPanel named "pnlCC"
          SET pnlCC layout to NULL
          SET pnlCC location to (50,30)
          SET pnlCC size to (800,750)
          SET pnlCC background color to ORANGE
          CREATE object of TitledBorder named "borderCredit"
          SET borderCredit title justification to center
          SET borderCredit title font to ("Arial",Bold,"24)
          ADD borderCredit in pnlCC
          ADD pnlCC in frame

          CREATE object of JLabel named "cvcNumLabel"
          SET cvcNumLabel bounds to (50,50,120,70)
          SET cvcNumLabel font to ("Ariel Black",PLAIN,15)
          ADD cvcNumLabel to pnlCC

          CREATE object of JTextField named "CVCtf"
          SET CVCtf bounds to (170,50,170,50)
          ADD CVCtf to pnlCC
```

```
CREATE object of JLabel named “interestLabel“
SET interestLabel bounds to (50,100,120,70)
SET interestLabel font to (“Ariel Black”,PLAIN,15)
ADD interestLabel to pnlCC

CREATE object of JTextField named “IRtf”
SET IRtf bounds to (170,110,170,50)
ADD IRtf to pnlCC

CREATE object of JLabel named “DOE_Label“
SET DOE_Label bounds to (50,160,120,70)
SET DOE_Label font to (“Ariel Black”,PLAIN,15)
ADD DOE_Label to pnlCC

CREATE object of JComboBox named “EYears”
SET EYears items to years
SET EYears bounds to (200,180,90,28)
ADD EYears to pnlCC

CREATE object of JComboBox named “EMonths”
SET EMonths items to months
SET EMonths bounds to (300,180,90,28)
ADD EMonths to pnlCC

CREATE object of JComboBox named “EDays”
SET EDays items to days
SET EDays bounds to (400,180,90,28)
ADD EDays to pnlCC

CREATE object of JLabel named “addCredCardidLabel“
SET addCredCardidLabel bounds to (50, 240, 120, 70)
```

```
**SET** addCredCardidLabel font to (“Futura”,PLAIN,45)
**ADD** addCredCardidLabel to pnlCC

**CREATE** object of JTextField named “CC_Add_CIDtf”
**SET** CC_Add_CIDtf bounds to (170,250,170,50)
**ADD** CC_Add_CIDtf to pnlCC

**CREATE** object of JLabel named “lblspam2“
**SET** lblspam2 bounds to (10,300,600,50)
**ADD** lblspam2 to pnlCC

**CREATE** object of JLabel named “credLimitLabel“
**SET** credLimitLabel bounds to (50, 360, 120, 70)
**SET** credLimitLabel font to (“Ariel Black”,PLAIN,15)
**ADD** credLimitLabel to pnlCC

**CREATE** object of JTextField named “CLtf”
**SET** CLtf bounds to (170,360,170,50)
**ADD** CLtf to pnlCC

**CREATE** object of JLabel named “graceLabel“
**SET** graceLabel bounds to (50, 410, 120, 70)
**SET** graceLabel font to (“Ariel Black”,PLAIN,15)
**ADD** graceLabel to pnlCC

**CREATE** object of JTextField named “GPtf”
**SET** GPtf bounds to (170,420,170,50)
**ADD** GPtf to pnlCC

**CREATE** object of JLabel named “setCreLim_CId_label“
**SET** setCreLim_CId_label bounds to (50, 470,120,70)
```

```
SET setCreLim_CId_label font to ("Ariel Black", Font.PLAIN, 15)
ADD setCreLim_CId_label to pnlCC

CREATE object of JTextField named "CCIDtf"
SET CCIDtf bounds to (170,480,170,50)
ADD CCIDtf to pnlCC

CREATE object of JLabel named "lblspam3"
SET lblspam3 bounds to (10,540,600,50)
ADD lblspam3 to pnlCC

CREATE object of JLabel named "cancelCC_CId_Label"
SET cancelCC_CId_Label bounds to (50, 600, 120, 70)
SET cancelCC_CId_Label font to ("Ariel Black",PLAIN,15)
ADD cancelCC_CId_Label to pnlCC

CREATE object of JTextField named "CancelCreditIDtf"
SET CancelCreditIDtf bounds to (170,610,170,50)
ADD CancelCreditIDtf to pnlCC

CREATE object of JButton named "addCreditCardButton"
SET addCreditCardButton text to "Add Credit Card"
SET addCreditCardButton bounds to (370, 250, 150, 50)
ADD ActionListener():
        WHEN CLICKED
                CALL addCreditCard()
ADD addCreditCardButton to pnlCC

CREATE object of JButton named "setCreditLimiButton"
SET setCreditLimiButton text to "Set Credit Limit"
SET setCreditLimiButton bounds to (430,400,270,50)
```

```
ADD ActionListener():
    WHEN CLICKED
        CALL addcreditLimit()
ADD setCreditLimiButton to pnlCC

CREATE object of JButton named “cancelCCButton”
SET cancelCCButton text to "Cancel Credit Card"
SET cancelCCButton bounds to (130,680,270,50)
ADD ActionListener():
    WHEN CLICKED
        CALL cancelCreditCard()
ADD cancelCCButton to pnlCC

CREATE object of JButton named “bnkCfButton”
SET bnkCfButton text to "Bank Card"
SET bnkCfButton bounds to (550,60,120,50)
ADD ActionListener():
    WHEN CLICKED
        SET pnlBC visibility to true
        SET pnlCC visibility to false
ADD bnkCfButton to pnlCC

CREATE object of JButton named “debCfButton”
SET debCfButton text to "Debit Card"
SET debCfButton bounds to (550,150,120,50)
ADD ActionListener():
    WHEN CLICKED
        SET pnlDC visibility to true
        SET pnlCC visibility to false
ADD debCfButton to pnlCC
```

```
    CREATE object of JButton named “creditDisplyButton”
    SET creditDisplyButton text to "Display"
    SET creditDisplyButton bounds to (500, 600, 90, 50)
    ADD ActionListener():
        WHEN CLICKED
            CALL Display()
    ADD creditDisplyButton to pnlCC

    CREATE object of JButton named “creditClearButton”
    SET creditClearButton text to "Clear"
    SET creditClearButton bounds to (600,600,100,50)
    ADD ActionListener():
        WHEN CLICKED
            CALL clearButton with parameter (pnlCC)
    ADD creditClearButton to pnlCC
END FUNCTION
```

3.5 Pseudo Code for method clearButton()

```
FUNCTION clearButon():
    FOR each Container c in cardList
        IF c is instance of JTextField THEN
            SET JTextField f to (JTextField) c
            SET f text to empty string
        ELSE IF c is instance of Container THEN
            CALL clearButton with parameter ((Container) c)
        END IF
    END FOR
    SET WYears Selected Index to 0
    SET WMonths Selected Index to 0
    SET WDays Selected Index to 0
END FUNCTION
```

3.6 Pseudo Code for method addDebitCard()

```
FUNCTION addDebitCard():
    SET issuerBank to the result of CALLING the getter method getIssuerBank()
    SET bankAccount to the result of CALLING the getter method getBankAccount()
    SET clientName to the result of CALLING the getter method getClientName()
    SET balanceAmount to the result of CALLING the getter method
        getBalanceAmount()
    SET cardID to the result of CALLING the getter method getAddDebitCardId()
    SET pinNumber to the result of CALLING the getter method getPinNumber()

    IF checkDebitCardUnique (cardID) THEN:
        ADD object of DebitCard in cardList
        SHOW message dialog “Debit Card added successfully”
        RETURN
    ELSE:
        SHOW message dialog  “Card not added”
    END IF
END FUNCTION
```

3.7 Pseudo Code for method withdrawal()

```
FUNCTION withdrawal():
    CREATE   a Boolean variable, is_Found and INITIALIZE it with value false
    IF getDebitCardID(),  getWithdrawalAmount() or getWithdrawalPinNumber()  is
      equal to -1 or getDateOfWithdrawal is empty THEN:
        SHOW message dialog with appropriate error message
    ELSE :
        SHOW message dialogue with values of getter methods
        FOR  each object stored in cardList:
            IF that object is instance of DebitCard
                CHANGE that object references to that of DebitCard
                IF getDebitCardId() equals to cardid of DebitCard
                    IF getWithdrawalPinNumber() equals to pinNumber of
                    DebitCard THEN:
                        UPDATE is_found value to true
                        CALL withdraw() method by that object
                        SHOW message dialog “Withdraw successful”
                        BREAK
                    ELSE:
                        SHOW message dialogue with appropriate
                        error message
                    END IF
                END IF
            END IF
        END FOR
        IF is_found is equal to false THEN:
            SHOW message dialogue with appropriate error message
        END IF
    END IF
END FUNCTION
```

3.8 Pseudo Code of method addCreditCard():

```
FUNCTION addCreditCard():
        SET cardID to the result of CALLING the getter method getAddCardIDCredit()
        SET clientName to the result of CALLING the getter method getClientName()
        SET issuerBank to the result of CALLING the getter method getIssuerBank()
        SET bankAccount to the result of CALLING the getter method getBankAccount()
        SET balanceAmount to the result of CALLING the getter method
              getBalanceAmount()
        SET cvcNumber to the result of CALLING the getter method getCVCNumber()
        SET interestRate to the result of CALLING the getter method getInterestRate()
        SET expirationDate to the result of CALLING the getter method
                getExpirationDate()
        IF checkCreditCardUnique(cardID) THEN:
                ADD object of CreditCard in cardList
                SHOW message dialog message with title "Credit Card added
                          successfully"
        ELSE:
                SHOW message dialog with appropriate error message
        END IF
END FUNCTION
```

2.9 Pseudo Code for method cancelCreditCard():

```
FUNCTION cancelCreditCard():
        IF getCardIDCredit() is not equal to -1 THEN:
                FOR each object stored in cardList:
                        IF CHECK object is an instance of CreditCard THEN:
                                SET that object references to that of CreditCard
                                IF CreditCard cardId is equal to getCancelCreditCardID()
                                THEN:
                                        IF isGranted is true THEN:
```

```
                        CALL cancelCreditCard() by that object
                        SHOW message dialog "Credit card is
                        cancelled successfully."
                        RETURN
                    ELSE:
                        SHOW message dialogue with error message
                    END IF
            ELSE:
                SHOW message dialogue with error message
                END IF
        END FOR
    ELSE:
        SHOW message dialogue with appropriate error message
    END IF
END FUNCTION
```

3.10 Pseudo Code for method checkDebitCardUnique(int cardID):

```
FUNCTION checkDebitCardUnique(int cardID):
    CREATE a Boolean variable isUnique and INITIALIZE it to true
    FOR each object stored in cardList:
            IF that object is instance of DebitCard
                CHANGE that object references to that of DebitCard
                IF cardID equals to cardid of DebitCard
                    SHOW message dialog with appropriate warning
                       Message
                    SET value of isUnique to false
                    BREAK
                END IF
            END IF
    END FOR
        RETURN isUnique
```

END FUNCTION

3.11 Pseudo Code for method checkCreditCardUnique(int cardid)

```
FUNCTION checkCreditCardUnique(int cardid):
    CREATE a Boolean variable isUnique and INITIALIZE it to true
    FOR each object stored in cardList:
            IF that object is instance of CreditCard
                CHANGE that object references to that of CreditCard
                IF cardID equals to cardid of CreditCard
                    SHOW message dialog with appropriate warning
                      message
                    SET value of isUnique to false
                    BREAK
                END IF
            END IF
    END FOR
        RETURN isUnique
END FUNCTION
```

3.11 Pseudo Code for method addCreditLimit()

```
FUNCTION addCreditLimit():
    SET cardID to the result of CALLING the getter method getCardIDCredit()
    SET creditLimit to the result of CALLING the getter method getCreditLimit()
    SET gracePeriod to the result of CALLING the getter method getGracePeriod()
    FOR each object in cardList:
        IF CHECK object is an instance of CreditCard THEN:
            SET reference of that object to that of CreditCard
            IF cc.getCardId() is equal to cardID THEN:
                SHOW message dialog “Credit Limit Added.”
                CALL cc.setCreditLimit() with creditLimit and gracePeriod as
                    arguments
            ELSE:
                SHOW message dialog with appropriate error message
            END IF
        END IF
    END FOR
END FUNCTION
```

3.12 Pseudo Code for method Display()

```
FUNCTION Display():
    FOR each object in cardList:
        IF object is an instance of DebitCard THEN:
            SET reference of object to that of DebitCard
            CALL dc.disout() method with that object
            SHOW message dialo "Display Information"
        ELSE IF object is an instance of CreditCard THEN:
            CALL cc.disout() method with that object
            SHOW message dialog "Display Information"
        ELSE:
             SHOW message dialog with appropriate error message
        END IF
    END FOR
END FUNCTION
```

3.13 Pseudo Code for getter methods of Bank Card

```
FUNCTION getBalanceAmount()
    CREATE AND INITIALIZE variable BalanceAmountText with value of BAmtf text
field, trimmed
    CREATE AND INITIALIZE variable BalanceAmount with value of INVALID
    TRY:
        CHANGE the value of BalanceAmountText to integer and INITIALIZE it to
                BalanceAmount
        IF BalanceAmount is less than 0 THEN:
            UPDATE BalanceAmount to INVALID
                SHOW message dialog with appropriate warning message
        END IF
        IF BalanceAmountText is empty THEN
            SHOW message dialog with approprtaie warning message
            END IF
        END TRY
    CATCH NumberFormatException
            SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of BalanceAmount
END FUNCTION


FUNCTION getIssuerBank()
    CREATE AND INITIALIZE variable IssuerBankText with value of IBtf text field,
    trimmed
    IF IssuerBankText is empty THEN:
            SHOW message Dialog with appropriate warning message
    END IF
    RETURN value of IssuerBankText
END FUNCTION


FUNCTION getBankAccount()
    CREATE AND INITIALIZE variable BankAccountText with value of BActf text
    field, trimmed
    IF BankAccountText is empty THEN:
            SHOW message dialog with appropriate warning message
    END IF
    RETURN value of BankAccountText
END FUNCTION
```

```
FUNCTION getClientName()
    CREATE AND INITIALIZE variable ClientNameText with value of CNatf text
    field, trimmed
    IF ClientNameText is empty THEN:
        SHOW message dialog with appropriate warning message
    END IF
    RETURN value of ClientNameText
END FUNCTION
```

3.14 Pseudo Code of getter methods of CreditCard

```
FUNCTION getAddCardIDCredit()
    CREATE AND INITIALIZE variable CardIDText with value of CC_Add_CIDtf text
    field, trimmed
    CREATE AND INITIALIZE variable CardID with value of -1
    TRY:
        IF CardIDText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CardIDText to integer and INITIALIZE it to CardID
        IF CardID is less than or equal to 0 THEN:
            UPDATE CardID to -1
        SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CardID
END FUNCTION


FUNCTION getCVCNumber()
    CREATE AND INITIALIZE variable CVCNumberText with value of CVCtf text field, trimmed
    CREATE AND INITIALIZE variable CVCNumber with value of -1
    TRY:
        IF CVCNumberText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CVCNumberText to integer and INITIALIZE it to CVCNumber
        IF CVCNumber is less than or equal to 0 THEN:
            UPDATE CVCNumber to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CVCNumber
END FUNCTION


FUNCTION getInterestRate()
```

```
    CREATE AND INITIALIZE variable InterestRateText with value of IRtf text field,
    trimmed

    CREATE AND INITIALIZE variable InterestRate with value of -1

    TRY:

        IF InterestRateText is empty THEN:

            SHOW message dialog with appropriate warning message

        END IF

            CHANGE the value of InterestRateText to double and INITIALIZE it
            to InterestRate

        IF InterestRate is less than or equal to 0 THEN:

            UPDATE InterestRate to -1

            SHOW message dialog with appropriate warning message

        END IF

    END TRY

    CATCH NumberFormatException

        SHOW message dialog with appropriate error message

    END CATCH

    RETURN value of InterestRate

END FUNCTION
```

```
FUNCTION getExpirationDate()

    CREATE AND INITIALIZE variable date to empty string

    CREATE variable year and INITIALIZE it to the selected item from EYears

    CREATE variable month and INITIALIZE it to the selected item from EMonths

    CREATE variable day and INITIALIZE it to the selected item from EDays

    TRY:

        IF year equals "Year" OR month equals "Month" OR day equals "Day"
        THEN:
            INITIALIZE date to null
            SHOW message dialog with appropriate warning message
        ELSE
            CONCATENATE year, "-", month, "-", and day together and
            INITIALIZE it to date
        END IF
    END TRY
    CATCH Exception e
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN date
END FUNCTION


FUNCTION getCreditLimit()
    CREATE AND INITIALIZE variable CreditLimitText with value of CLtf text field,
    trimmed
    CREATE AND INITIALIZE variable CreditLimit with value of -1
    TRY:
        IF CreditLimitText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CreditLimitText to integer and INITIALIZE it to
        CreditLimit
        IF CreditLimit is less than or equal to 0 THEN:
            UPDATE CreditLimit to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CreditLimit
END FUNCTION


FUNCTION getCardIDCredit()
    CREATE AND INITIALIZE variable CardIDText with value of CCIDtf text field,
    trimmed
    CREATE AND INITIALIZE variable CardID with value of -1
    TRY:
        IF CardIDText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CardIDText to integer and INITIALIZE it to CardID
        IF CardID is less than or equal to 0 THEN:
            UPDATE CardID to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CardID
END FUNCTION


FUNCTION getGracePeriod()
```

```
    CREATE AND INITIALIZE variable GracePeriodText with value of GPtf text field,
    trimmed
    CREATE AND INITIALIZE variable GracePeriod with value of -1
    TRY:
        CHANGE the value of GracePeriodText to integer and INITIALIZE it to
        GracePeriod
        IF GracePeriodText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        IF GracePeriod is less than or equal to 0 THEN:
            UPDATE GracePeriod to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of GracePeriod
END FUNCTION


FUNCTION getCancelCreditCardID()
    CREATE AND INITIALIZE variable CardIDText with value of CancelCreditIDtf
    text field, trimmed
    CREATE AND INITIALIZE variable CardID with value of -1
    TRY:
        IF CardIDText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CardIDText to integer and INITIALIZE it to CardID
        IF CardID is less than or equal to 0 THEN:
            UPDATE CardID to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CardID
END FUNCTION


FUNCTION getCVCNumber()
    CREATE AND INITIALIZE variable CVCNumberText with value of CVCtf text
    field, trimmed
    CREATE AND INITIALIZE variable CVCNumber with value of -1
    TRY:
        IF CVCNumberText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CVCNumberText to integer and INITIALIZE it to
              CVCNumber
        IF CVCNumber is less than or equal to 0 THEN:
            UPDATE CVCNumber to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
```

```
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CVCNumber
END FUNCTION
```

3.15 Pseudo Code for getter methods of Debit Card

```
FUNCTION getPinNumber()
    CREATE AND INITIALIZE variable PinNumberText with value of PNtf text field,
    trimmed
    CREATE AND INITIALIZE variable PinNumber with value of -1
    TRY:
        CHANGE the value of PinNumberText to integer and INITIALIZE it to
        PinNumber
        IF PinNumberText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        IF PinNumber is less than 0 THEN:
            UPDATE PinNumber to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of PinNumber
END FUNCTION
```

```
FUNCTION getAddDebitCardId()
    CREATE AND INITIALIZE variable DebitCardIDText with value of DC_Add_IDtf
    text field, trimmed
    CREATE AND INITIALIZE variable DebitCardID with value of -1
    TRY:
        IF DebitCardIDText is empty THEN:
                SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of DebitCardIDText to integer and INITIALIZE it to
        DebitCardID
        IF DebitCardID is less than or equal to 0 THEN:
            UPDATE DebitCardID to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of DebitCardID
END FUNCTION


FUNCTION getWithdrawalPinNumber()
    CREATE AND INITIALIZE variable PinNumberText with value of PNWtf text
    field, trimmed
    CREATE AND INITIALIZE variable PinNumber with value of -1
    TRY:
        CHANGE the value of PinNumberText to integer and INITIALIZE it to
        PinNumber
        IF PinNumberText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        IF PinNumber is less than 0 THEN:
            UPDATE PinNumber to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of PinNumber
END FUNCTION


FUNCTION getCancelCreditCardID()
    CREATE AND INITIALIZE variable CardIDText with value of CancelCreditIDtf
    text field, trimmed
    CREATE AND INITIALIZE variable CardID with value of -1
    TRY:
        IF CardIDText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
        CHANGE the value of CardIDText to integer and INITIALIZE it to CardID
        IF CardID is less than or equal to 0 THEN:
            UPDATE CardID to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of CardID
END FUNCTION


FUNCTION getWithdrawalAmount()
    CREATE AND INITIALIZE variable WithdrawalAmountText with value of WAmtf
    text field, trimmed
    CREATE AND INITIALIZE variable WithdrawalAmount with value of -1
    TRY:
        IF WithdrawalAmountText is empty THEN:
            SHOW message dialog with appropriate warning message
        END IF
            CHANGE the value of WithdrawalAmountText to integer and
            INITIALIZE it  to WithdrawalAmount
        IF WithdrawalAmount is less than or equal to 0 THEN:
            UPDATE WithdrawalAmount to -1
            SHOW message dialog with appropriate warning message
        END IF
    END TRY
    CATCH NumberFormatException
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of WithdrawalAmount
END FUNCTION
```

```
FUNCTION getDateOfWithdrawal()
    CREATE AND INITIALIZE variable date with empty string
    CREATE AND INITIALIZE variable year with value of WYears selected item as
    string
    CREATE AND INITIALIZE variable month with value of WMonths selected item
    as string
    CREATE AND INITIALIZE variable day with value of WDays selected item as
    string
    TRY:
        IF year equals "Year" OR month equals "Month" OR day equals "Day"
        THEN:
            UPDATE date to null
        ELSE:
            CONCATENATE year, "-", month, "-", and day together and
            INITIALIZE it to date
        END IF
    END TRY
    CATCH Exception
        SHOW message dialog with appropriate error message
    END CATCH
    RETURN value of date
END FUNCTION

FUNCTION getDebitCardId()
    CREATE AND INITIALIZE variable DebitCardIDText with value of DCIDtf text
    field, trimmed
    CREATE AND INITIALIZE variable DebitCardID with value of -1
    TRY:
        IF DebitCardIDText is empty THEN:
            **SHOW** message dialog with appropriate warning message
        **END IF**
        **CHANGE** the value of DebitCardIDText to integer and **INITIALIZE** it to
            DebitCardID
        **IF** DebitCardID is less than or equal to 0 THEN:
            **UPDATE** DebitCardID to -1
            **SHOW** message dialog with appropriate warning message
        **END IF**
    **END TRY**
    **CATCH** NumberFormatException
        **SHOW** message dialog with appropriate error message
    **END CATCH**
    **RETURN** value of DebitCardID
**END FUNCTION**
```

3.16 Pseudo Code for main method

```
**FUNCTION main (String [] args)**
    **CREATE** an object of constructor BankGUI() and set frame visibility to true
**END FUNCTION**
```

4. Method Description

4.1 Method description of :

- public BankGUI()

  This is a constructor used to call methods that contains all the content pane elements. InitFrame(), initBankCard, initDebitCard() and initCreditCard().

- public void initFrame()

  This method contains all the frame details and initializes the frame where all the work are kept In GUI program.

- public void initBankCard()

  This method contains the labels, text fields, buttons of bank card and also is the gateway to credit card and debit card panels.

- public initDebitCard()

  This method contains labels, text fields, buttons , combo box which is necessary to enter the data of debit card. This also has elements necessary to withdraw balance.

- public initCreditCard()

  This method contains labels, text fields, buttons , combo box which is necessary to enter the data of credit card. This also has elements necessary to set credit limit and cancel credit card.

4.2 Method description of getter methods of Bank card

- getBalanceAmount() is integer return type method that returns Balanceamount, it takes data from JTextfield BAmtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If BAmtf is empty it shows message and if it is less than 0 returns -1.

- getIssuerBank() is String return type method that returns IssuerBankText, it takes data from JTextField IBtf trims it to remove space and checks if it is empty. If it is empty it shows warning message.

- getBankAccount() is String return type method that returns BankAccountText, it takes data from JTextField BActf trims it to remove space and checks if it is empty. If it is empty it shows warning message.

- getClientName is String return type method that returns ClientNameText, it takes data from JTextField CNatf trims it to remove space and checks if it is empty. If it is empty it shows warning message.

4.3 Method description of getter methods of Credit Card

- getAddCardIDCredit() is integer return type method that returns CardID, it takes data from JTextfield CC_Add_CIDtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If CC_Add_CIDtf is empty it shows message and if it is less than 0 returns -1.

- getCVCNumber() is integer return type method that returns CVCNumber, it takes data from JTextfield CVCtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If CVCtf is empty it shows message and if it is less than 0 returns -1.

- getInterestRate() is double return type method that returns InterestRate, it takes data from JTextfield IRtf and converts the string type to integer using parseDouble method which is done inside try and catch block to catch exception. If IRtf is empty it shows message and if it is less than 0 returns -1.

- getExpirationDate() is a string return type method which returns currently selected items from date using getSelectedItem() method which is further parsed into string to avoid any exception and all EYears, EMonths, EDays are initialized to date which is returned in proper format.

- getCreditLimit() is integer return type method that returns CreditLimit, it takes data from JTextfield CLtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If CLtf is empty it shows message and if it is less than 0 returns -1.

- getCardIDCredit() is integer return type method that returns CardID, it takes data from JTextfield CCIDtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If CCIDtf is empty it shows message and if it is less than 0 returns -1.

- getGracePeriod() is integer return type method that returns GracePeriod, it takes data from JTextfield GPtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If GPtf is empty it shows message and if it is less than 0 returns -1.

- getCancelCreditCardID () is integer return type method that returns CardID, it takes data from JTextfield CancelCreditIDtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If CancelCreditIDtf is empty it shows message and if it is less than 0 returns -1.

4.4 Method description of getter methods of Debit Card

- getPinNumber() is integer return type method that returns PinNumber, it takes data from JTextfield Pntf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If PNtf is empty it shows message and if it is less than 0 returns -1.

- getAddDebitCardId() is integer return type method that returns DebitCardID, it takes data from JTextfield DC_Add_IDtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If DC_Add_IDtf is empty it shows message and if it is less than 0 returns -1.

- getWithdrawalPinNumber() is integer return type method that returns PinNumber, it takes data from JTextfield PNWtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If PNWtf is empty it shows message and if it is less than 0 returns -1.

- getWithdrawalAmount() is integer return type method that returns WithdrawalAmount, it takes data from JTextfield WAmtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If WAmtf is empty it shows message and if it is less than 0 returns -1.

- getDateofWithdrawal() is a string return type method which returns currently selected items from date using getSelectedItem() method which is further parsed into string to avoid any exception and all WYears, WMonths, WDays are initialized to date which is returned in proper format.

- getDebitCardId() is integer return type method that returns DebitCardID, it takes data from JTextfield DCIDtf and converts the string type to integer using parseInt method which is done inside try and catch block to catch exception. If DCIDtf is empty it shows message and if it is less than 0 returns -1.

4.5 Method description for method Clear and Display methods

- clearButton() : The void return type clear method accepts an integer Container as an input parameter. It runs for all of the Container's components and tests whether the component is an instance of JTextField and sets its text to null, otherwise it checks if the component is of Container type and calls itself with the current component as argument. In addition it'll set the Index of all the ComboBox to 0.

- Display() : This is a void type method that invokes the display method on DebitCard if the object generated is from object of Debit Card in arraylist. Otherwise it invokes the display from CreditCard if the object is instance of Credit Card and the details are displayed in terminal along with a dialog box if both are not fulfilled then error message is shown that object is not found and to add a card.

4.6 Method description for addDebitCard method:

- addDebitCard(): The addDebitCard method is a void return type method that adds DebitCard objects to the cardList ArrayList. It takes all of the required arguments for initializing DebitCard through the predefined getter methods and then validates all of the variables to check whether they are valid or not. If the entered variable is not valid, it throws an error pop up window and returns the method; otherwise, it checks if the vehicle is unique and adds it to the cardList and displays an appropriate pop up message.

4.7 Method description for withdrawal method:

- withdrawal(): The withdrawal method is void return type method that calls method of Debit Card object which are stired in the array list. It takes all the required arguments that are needed for withdraw() methid through predefined getter methods and validates all of those variables to check if they are valid and if not a suitable dialog message is shown and returns the method. Otherwise, it checks if the cardId matches the cardid that is in cardlist and calls withdraw() method on and also displays an appropriate message.

4.8 Method description for addCreditCard

- addCreditCard() : The addCreditCard() method is a void return type method that adds CreditCard objects to the cardList ArrayList. It takes all of the required arguments for initializing CreditCard through the predefined getter methods and then validates all of the variables to check whether they are valid or not. If the entered variable is not valid, it throws an error pop up window and returns the method; otherwise, it checks if the vehicle is unique and adds it to the cardList and displays an appropriate pop up message.

4.9 Method description for addCreditLimit

- addCreditLimit() : This is a void type method that invokes the setCreditLimit() method on CreditCard if the object generated is from object of Credit Card in arraylist. It takes all of the required arguments by calling their getter methods and if the cardId given is same to the cardid in arraylist object than calls the setCreditLimit with a proper message else shows appropriate error message.

4.10 Method description for cancelCreditCard

- cancelCreditCard() : This is a void type method that invokes the cancelCreditCard() method on CreditCard if the object from the arraylist is instance of CreditCard, It takes a cardId by calling its getter method and checks if that is same to one in object of arraylist and calls the method cancelCreditCard along with proper message. Else shows appropriate error message.

4,11 Method description for methods checkUnique

- checkDebitCardUnique(): This is a Boolean return type method that initializes a Boolean isUnique to true and if the object generated is from object of Debit Card in arraylist then it down casts and checks if the entered cardid is equal to the one in object of arraylist if so then it shows appropriate message and changes the Boolean to false then breaks.

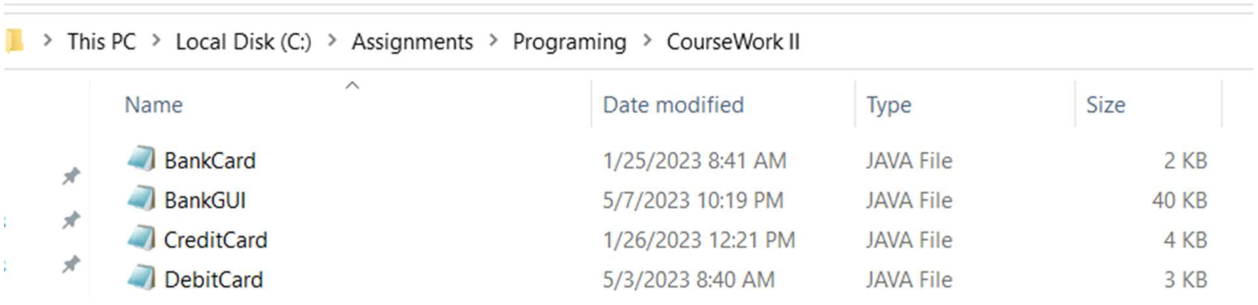
- checkCreditCardUnique(): This is a Boolean return type method that initializes a Boolean isUnique to true and if the object generated is from object of Credit Card in arraylist then it down casts and checks if the entered cardid is equal to the one in object of arraylist if so then it shows appropriate message and changes the Boolean to false then breaks.

5. Testing

5.1 Testing of compilation in terminal and run the program

Table 1: Test 1 (Compile on cmd)

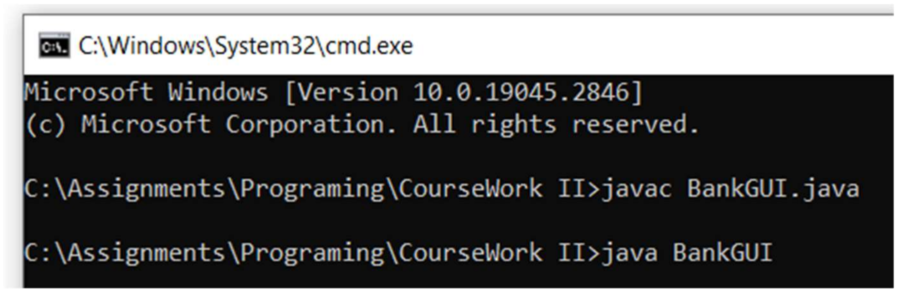
OBJECTIVE	Testing whether the program can be compiled and run using command prompt or not
ACTION	1.Open command prompt from related source code file location 2.Write the following commands: javac BankGUI.java java BankGUI
EXPECTED RESULT	GUI should appear after correctly writing the commands on command prompt.
ACTUAL RESULT	The code is successfully compiled and the GUI appeared without any error.
CONCLUSION	The test is successful.



This PC > Local Disk (C:) > Assignments > Programing > CourseWork II

Name	Date modified	Type	Size
BankCard	1/25/2023 8:41 AM	JAVA File	2 KB
BankGUI	5/7/2023 10:19 PM	JAVA File	40 KB
CreditCard	1/26/2023 12:21 PM	JAVA File	4 KB
DebitCard	5/3/2023 8:40 AM	JAVA File	3 KB

Figure 2: File location of source code

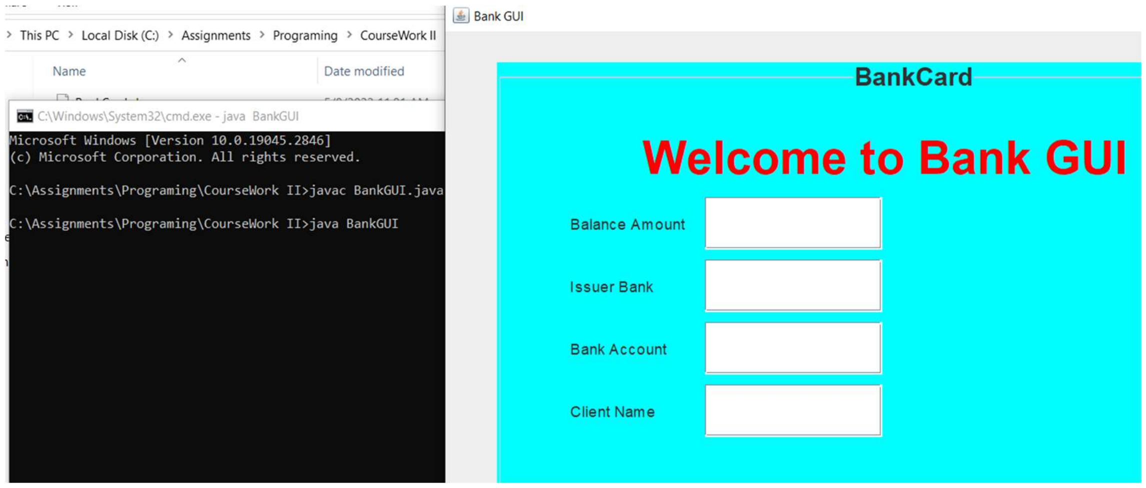

C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Assignments\Programing\CourseWork II>javac BankGUI.java

C:\Assignments\Programing\CourseWork II>java BankGUI
```

Figure 3: File compilation in command prompt


This PC > Local Disk (C:) > Assignments > Programing > CourseWork II
Name
Date modified
C:\Windows\System32\cmd.exe - java BankGUI

```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Assignments\Programing\CourseWork II>javac BankGUI.java

C:\Assignments\Programing\CourseWork II>java BankGUI
```

Bank GUI
BankCard
Welcome to Bank GUI
Balance Amount
Issuer Bank
Bank Account
Client Name

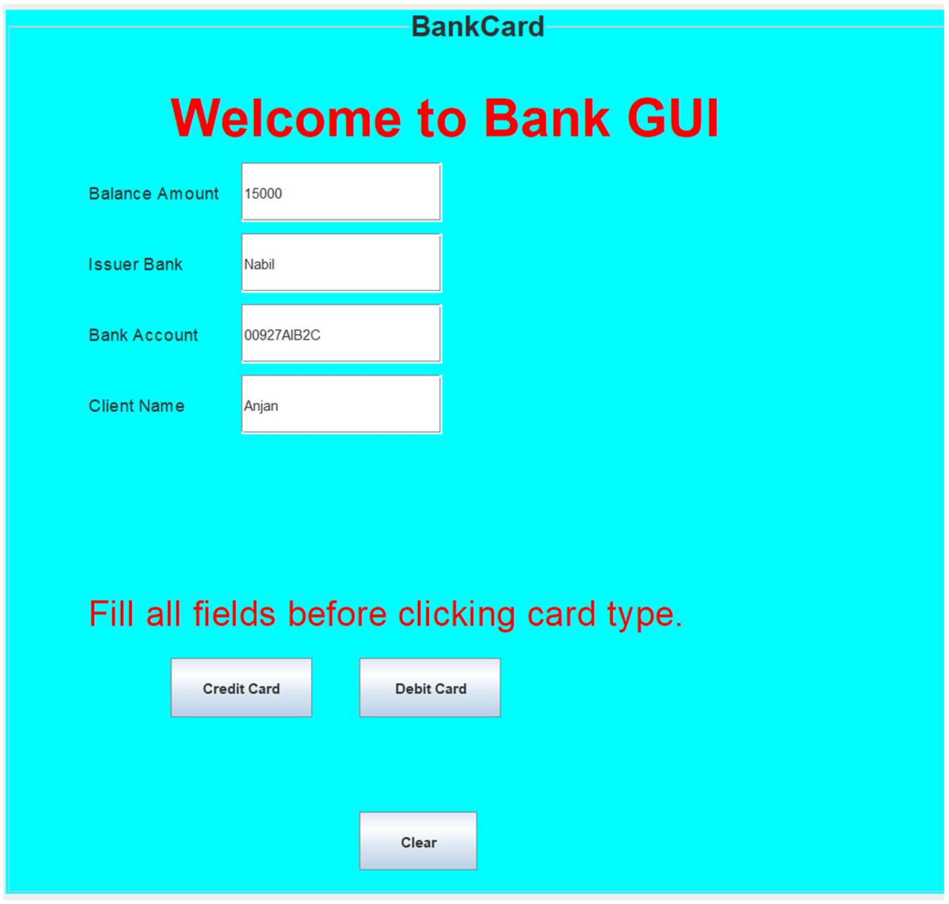
Figure 4: GUI opened through command prompt

5.2 Testing of Buttons

5.2.1 Testing of button to add Debit Card

Table 2: Test 2 (Add debit card)

OBJECTIVE	Testing whether we can add Debit Card or not
ACTION	1. Open BankGUI Class 2. Fill the fields in BankCard panel Balance Amount : 15000 Issuer Bank: Nabil Bank Account: 00927AIB2C Client Name: Anjan 3. Click on DebitCard button and go to DebitDard panel 4. Fill the Fields to add debit card Pin Number : 9967 Card ID: 78921 5. Click the Add Debit Card button
EXPECTED RESULT	The DebitCard should be added successfully, and message box should pop up saying successfully added.
ACTUAL RESULT	The DebitCard is added successfully.
CONCLUSION	The test is successful.



Balance Amount 15000

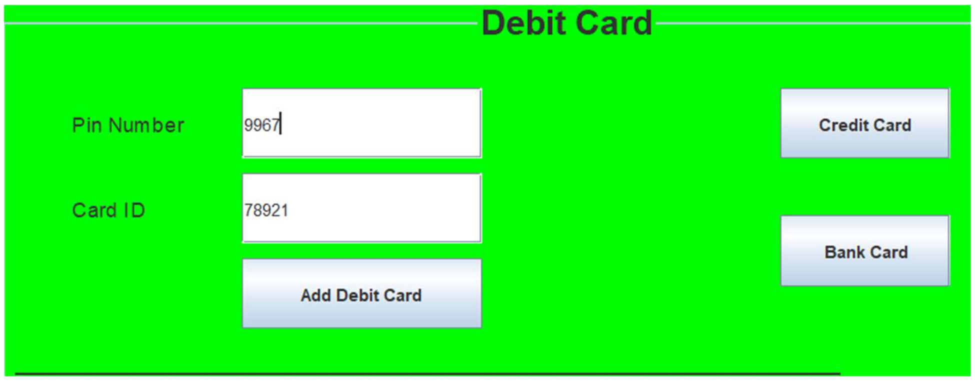
Issuer Bank Nabil

Bank Account 00927AIB2C

Client Name Anjan

Fill all fields before clicking card type.

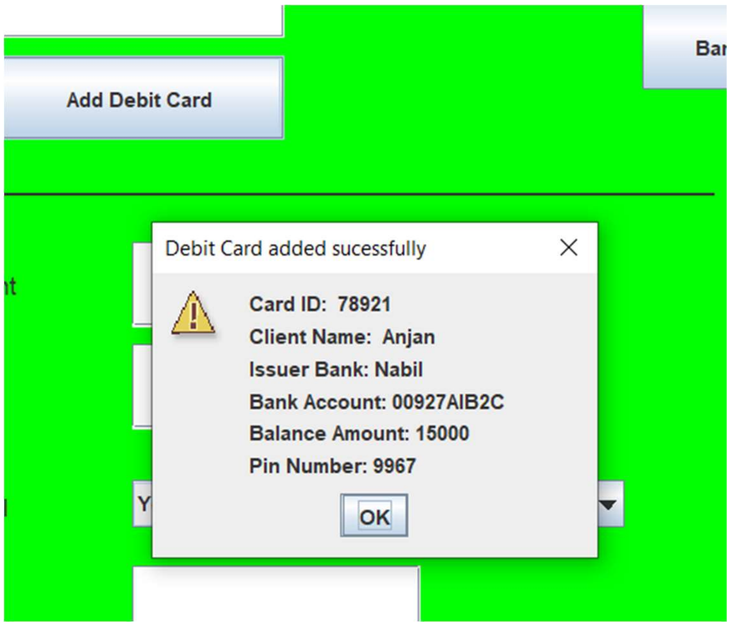
Figure 5: Adding data in bank card fields



Pin Number 9967

Card ID 78921

Figure 6: Adding data in debit card fields



Debit Card added sucessfully

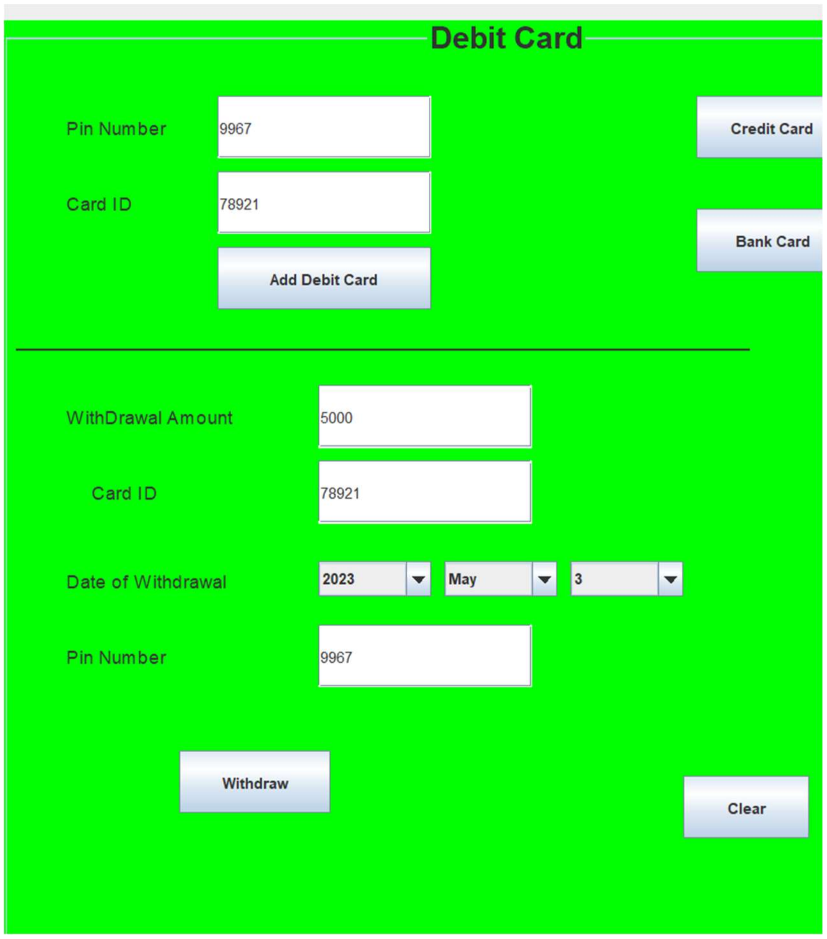
Card ID: 78921
Client Name: Anjan
Issuer Bank: Nabil
Bank Account: 00927AIB2C
Balance Amount: 15000
Pin Number: 9967

Figure 7: Displaying the evidence of adding DebitCard

5.2.2 To withdraw from Debit Card

Table 3: Test to check withdraw button

OBJECTIVE	Testing whether we can withdraw balance or not.
ACTION	1. After adding DebitCard go to withdraw fields and enter the data necessary WithDrawal Amount : 5000 Card ID : 78921 Date of Withdrawal : 2023-May-3 Pin Number: 9967 2.Click on Withdraw button
EXPECTED RESULT	The withdraw method from Debit Card class should be called and the balace should be withdrawn with message and print.
ACTUAL RESULT	The balance is withdrawn and message and print is shown.
CONCLUSION	The test is successful.



Debit Card

Pin Number 9967

Card ID 78921

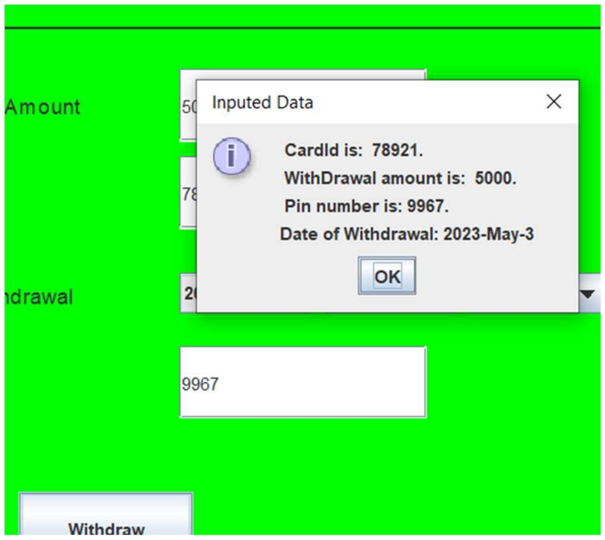
Withdrawal Amount 5000

Card ID 78921

Date of Withdrawal 2023 May 3

Pin Number 9967

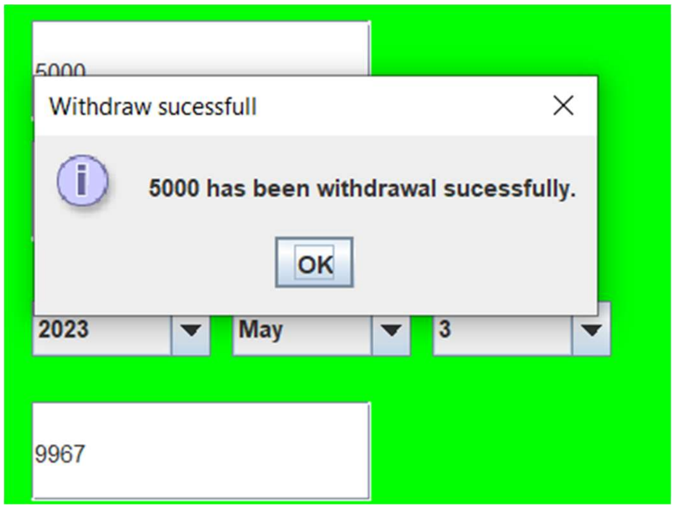
Figure 8: Adding data in withdraw fields



Inputed Data

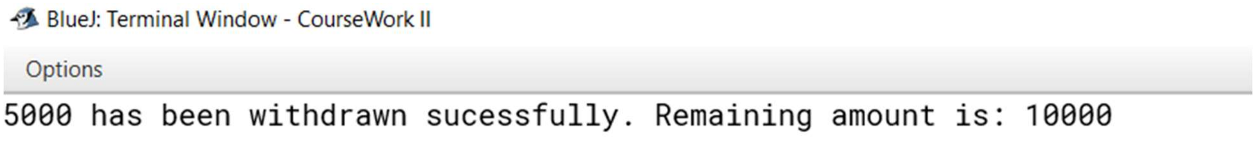
CardId is: 78921.
WithDrawal amount is: 5000.
Pin number is: 9967.
Date of Withdrawal: 2023-May-3

Figure 9: Message to show our input



5000 has been withdrawal sucessfully.

Figure 10: Message confirming our withdraw.



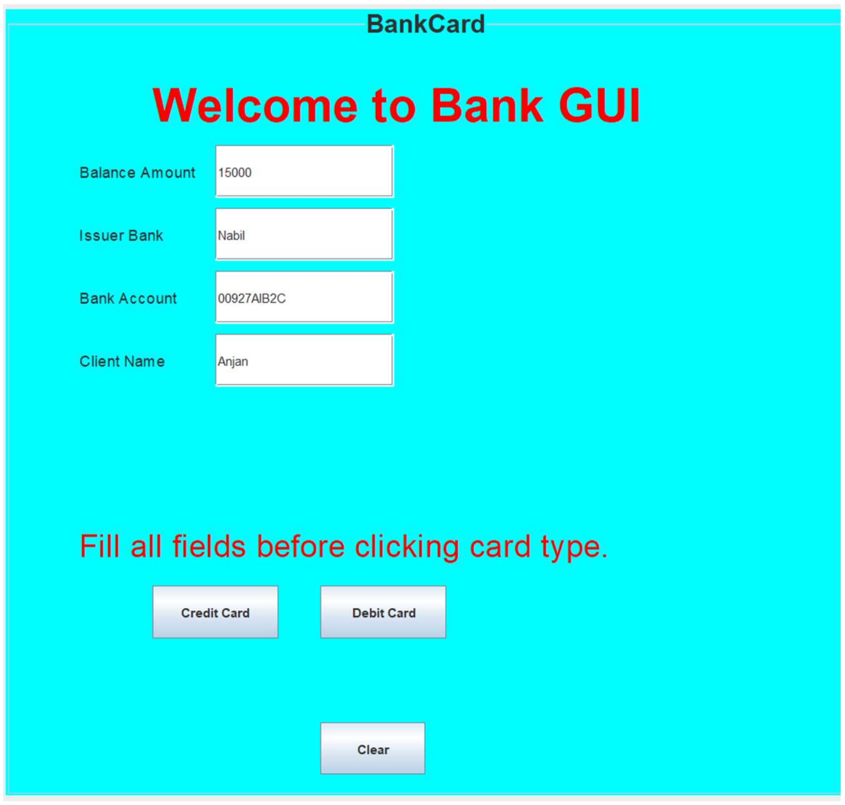
```
5000 has been withdrawn sucessfully. Remaining amount is: 10000
```

Figure 11: Print in terminal from withdraw method of Debit Card class

5.2.3 Testing of button to add CreditCard

Table 4: Testing of Adding Credit Card

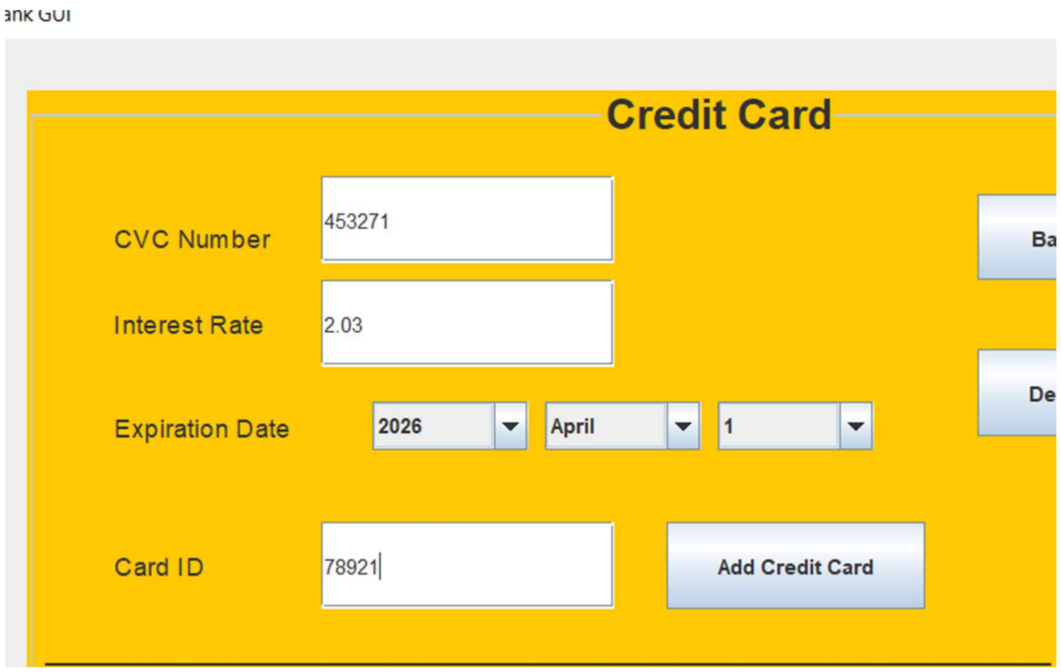
OBJECTIVE	Testing whether we can add Credit Card or not
ACTION	1. Open BankGUI Class 2. Fill the fields in BankCard panel Balance Amount : 15000 Issuer Bank: Nabil Bank Account: 00927AIB2C Client Name: Anjan 3. Click on CreditCard button and go to CreditCard panel 4. Fill the Fields to add debit card CVC Number : 453271 Interest Rate: 2.03 Expiration Date: 2026-April-1 Card ID: 78921 5. Click the Add Debit Card button
EXPECTED RESULT	The CreditCard should be added successfully, and message box should pop up saying successfully added.
ACTUAL RESULT	The CreditCard is added successfully.
CONCLUSION	The test is successful.



Balance Amount: 15000
Issuer Bank: Nabil
Bank Account: 00927AIB2C
Client Name: Anjan

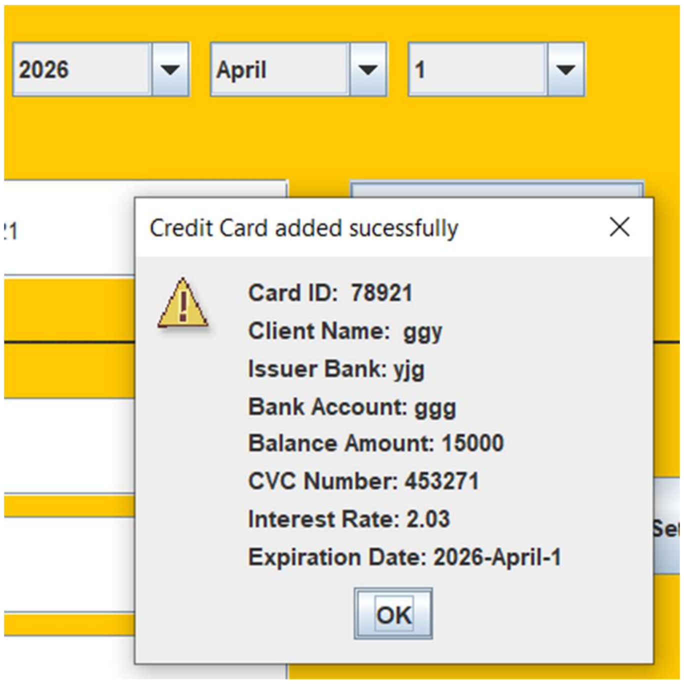
Fill all fields before clicking card type.

Figure 12: Adding data in BankCard Fields



CVC Number: 453271
Interest Rate: 2.03
Expiration Date: 2026 April 1
Card ID: 78921

Figure 13: Adding data in Credit Card fields



Credit Card added sucessfully

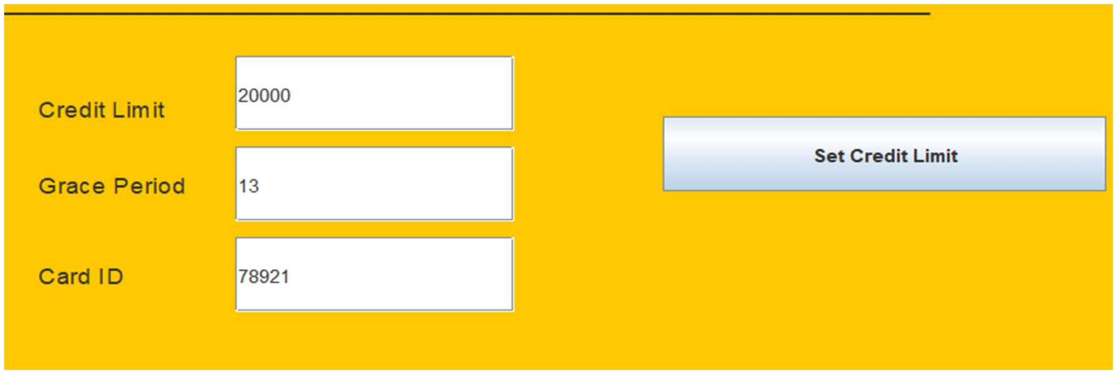
Card ID: 78921
Client Name: ggy
Issuer Bank: yjg
Bank Account: ggg
Balance Amount: 15000
CVC Number: 453271
Interest Rate: 2.03
Expiration Date: 2026-April-1

Figure 14: Evidence of Credit Card being added

5.2.5 Test to Set Credit Limit

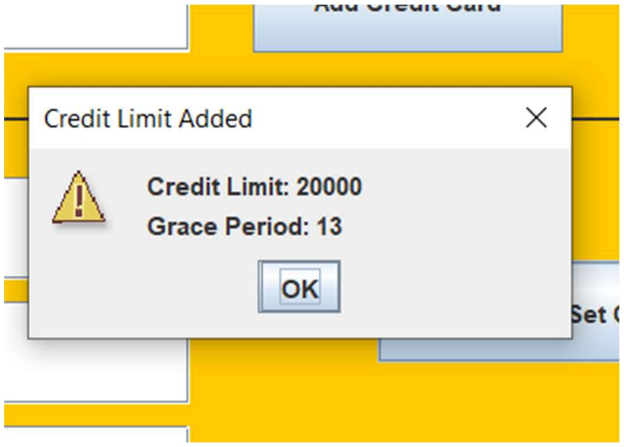
Table 5: Test to set Credit Limit

OBJECTIVE	Testing whether we can set Credit Limit or not.
ACTION	1. After adding CreditCard go to CreditLimit fields and enter the data necessary Credit Limit : 20000 Grace Period: 13 Card ID : 78921 2.Click on Set Credit Limit button
EXPECTED RESULT	The setCreditLimit method from CreditCard class should be called and the CreditLimit should be set with message and print.
ACTUAL RESULT	The CreditLimit is set and message and print is shown.
CONCLUSION	The test is successful.



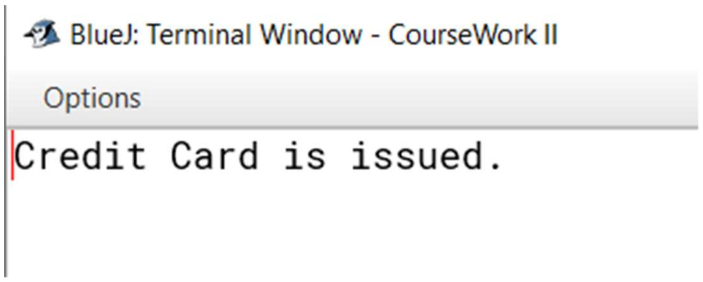
Credit Limit: 20000
Grace Period: 13
Card ID: 78921

Figure 15: Adding data for CreditLimit set



Credit Limit: 20000
Grace Period: 13

Figure 16: Evidence of CreditLimit being added



```
Credit Card is issued.
```

Figure 17: Print from method setCreditLimit of CreditCard class

5.2.6 Test to cancel Credit Card

Table 6: Test to cancel CreditCard

OBJECTIVE	Testing whether we can cancel Credit card or not.
ACTION	1. After adding CreditCard and setting credit limit go to cancel credit card fields and enter the data necessary Card ID : 78921 2.Click on cancel Credit card button
EXPECTED RESULT	The cancelCreditCard method from CreditCard class should be called and the CreditCard should be cancelled with message and print.
ACTUAL RESULT	The CreditCard is cancelled and message and print is shown.
CONCLUSION	The test is successful.

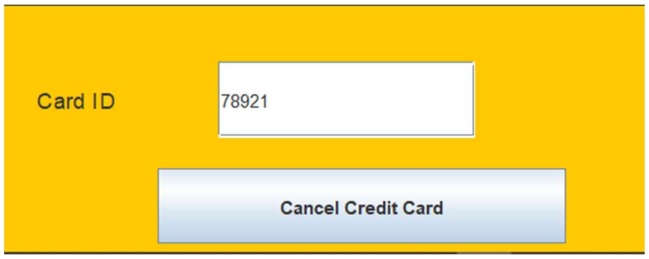
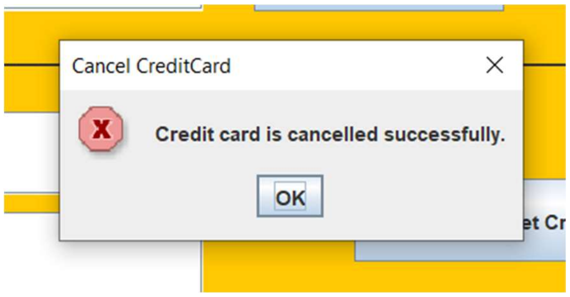
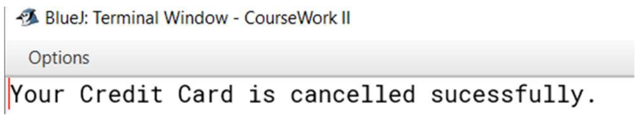


Figure 18: Adding card id for cancelling credit card



Credit card is cancelled successfully.

Figure 19: Evidence of CreditCard being cancelled



```
Your Credit Card is cancelled sucessfully.
```

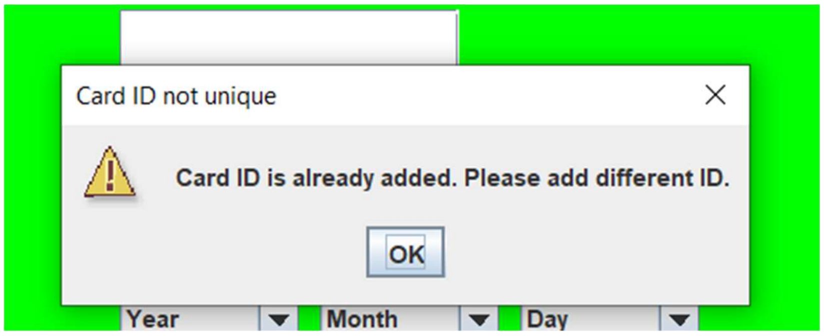
Figure 20: Print from cancelCreditCard method of CreditCard class

5.3 Test 3

5.3.1 To validate unique cardid on DebitCard

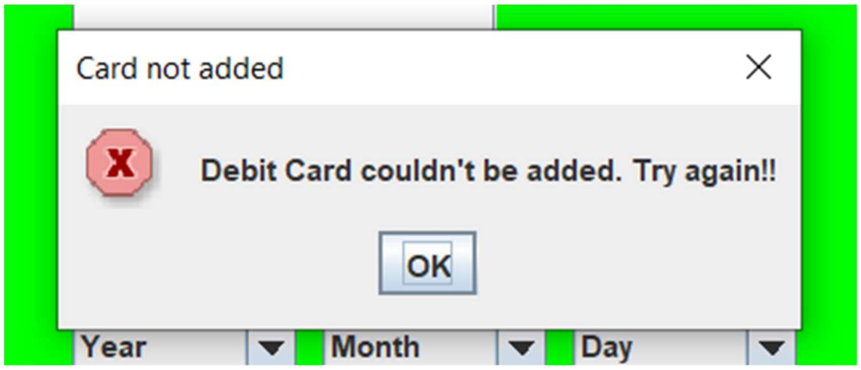
Table 7: Test to validate unique card id

OBJECTIVE	Testing whether the card id must be unique or not.
ACTION	1. Open BankGUI class 2.Enter the fields with appropriate data on BankCard panel 3. Fill all the text fields with appropriate data on DebitCard panel for adding DebitCard 4.Click the Add Debit Card button 5. Again fill the text fields with the same CardId 6. Click the Add button
EXPECTED RESULT	The error message saying "Card id not unique" and debit card not added should pop up.
ACTUAL RESULT	The error message saying "Card id not unique" and "debit card not added" is popped up..
CONCLUSION	The test is successful.


Card ID not unique

Card ID is already added. Please add different ID.

Figure 21: Message saying card id not unique


Card not added

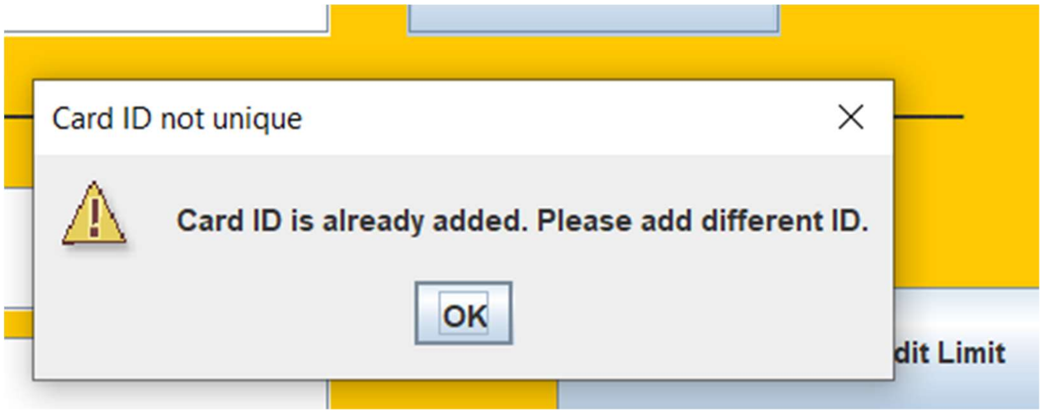
Debit Card couldn't be added. Try again!!

Figure 22: Message saying Debit card not added

5.3.2 To validate unique cardid on CreditCard

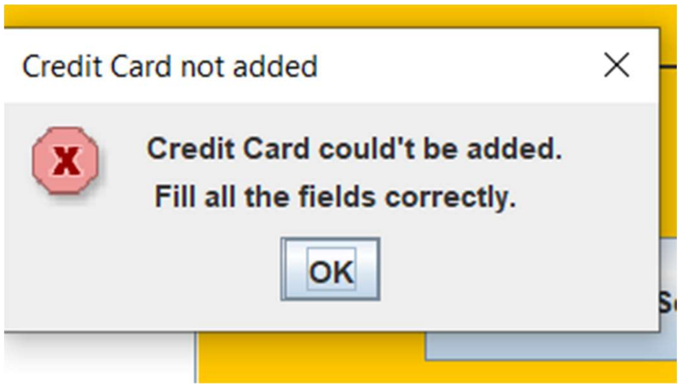
Table 8: Test to see unique cardid on CreditCard

OBJECTIVE	Testing whether the card id must be unique or not.
ACTION	1. Open BankGUI class 2.Enter the fields with appropriate data on BankCard panel 3. Fill all the text fields with appropriate data on CreditCard panel for adding CreditCard 4.Click the Add CreditCard button 5. Again fill the text fields with the same CardId 6. Click the Add button
EXPECTED RESULT	The error message saying “Card id not unique” and CreditCard not added should pop up.
ACTUAL RESULT	The error message saying “Card id not unique” and “CreditCard card not added” is popped up..
CONCLUSION	The test is successful.


Card ID not unique

Card ID is already added. Please add different ID.

Figure 23: Message saying cardid not unique


Credit Card not added

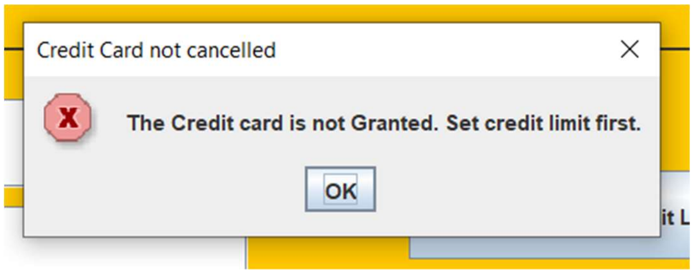
Credit Card could't be added.
Fill all the fields correctly.

Figure 24: Message saying credit card not added

5.3.2 Test to cancel Credit Card without setting Credit limit

Table 9: Test to cancel credit card without setting credit limit

OBJECTIVE	Testing whether we can cancel Credit card without setting credit limit.
ACTION	1. After adding CreditCard and go to cancel credit card fields and enter same card id used to add CreditCard Card ID : 12431 2.Click on cancel Credit card button
EXPECTED RESULT	The method cancel CreditCard checks if the Credit Card is granted and if false then shows appropriate error message saying “Credit Card not cancelled”.
ACTUAL RESULT	The CreditCard is not cancelled and error message is shown.
CONCLUSION	The test is successful.


Credit Card not cancelled

The Credit card is not Granted. Set credit limit first.

Figure 25: Message saying credit card not cancelled

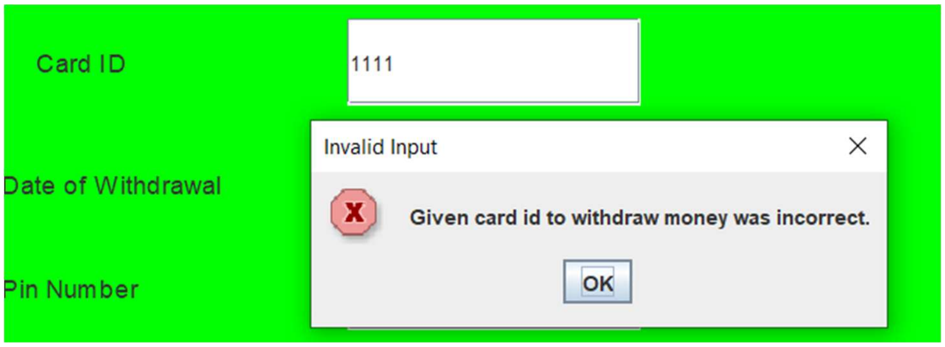
5.3.3 Test to withdraw without incorrect card Id

Table 10: Test to withdraw with incorrect Card ID:

OBJECTIVE	Testing whether we can withdraw with incorrect card id

ACTION	1. After adding DebitCard and go to withdraw fields and enter different card id used to add DebitCard
	for adding Card ID : 12498
	for withdrawing Card ID : 1111
	2.Click on cancel Withdraw button

EXPECTED RESULT	The method withdrawal find card id on object of debit card is not equal to card id used to withdraw and shows appropriate error message.
ACTUAL RESULT	The balance is not withdrawn and error message is shown.
CONCLUSION	The test is successful.


Card ID 1111

Date of Withdrawal

Pin Number

Invalid Input

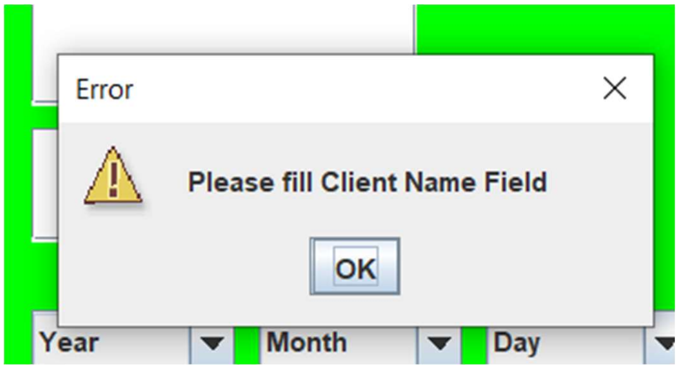
Given card id to withdraw money was incorrect.

Figure 26: Message showing card id is incorrect for withdraw

5.3.4 Test to put String, negative and empty values in int and string and add Debitcard

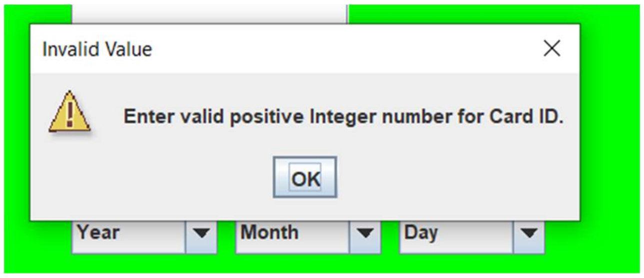
Table 11: Test to see messages of invalid input

OBJECTIVE	Testing whether we can put invalid text on fields or not
ACTION	1. Enter negative , String and empty values on textfields. cardid : -199 pinNumber: Check Client Name : “ ” 2. Click on add debit card button
EXPECTED RESULT	appropriate error message should be shown saying cardid and pin number is invalid and client name is empty and debit card not added.
ACTUAL RESULT	Messages are shown and debit card is not added.
CONCLUSION	The test is successful.



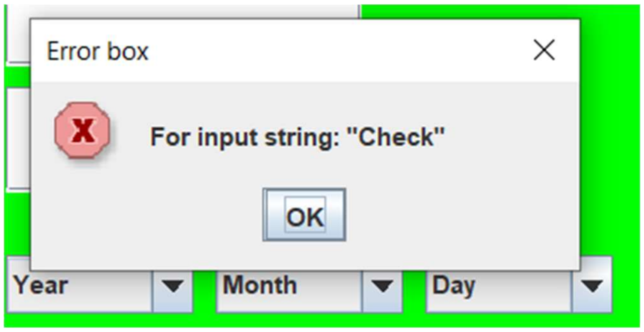
Please fill Client Name Field

Figure 27:Error message for empty client name


Invalid Value

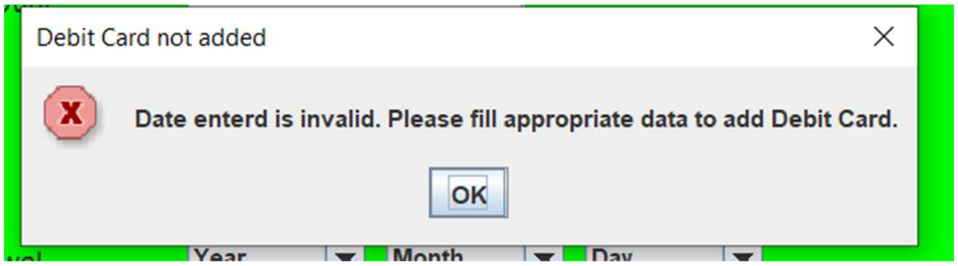
Enter valid positive Integer number for Card ID.

Figure 28: Error message for negative card id


Error box

For input string: "Check"

Figure 29: Error message for String input in integer


Debit Card not added

Date enterd is invalid. Please fill appropriate data to add Debit Card.

Figure 30: Error message for invalid input

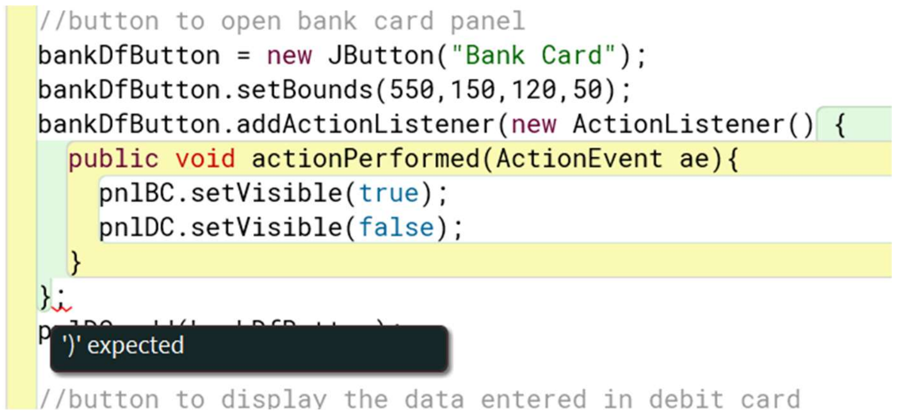
6. Error Detection and solution

6.1 Syntax error

Syntax errors or compile errors are errors identified by the compiler. Syntax errors are caused by mistakes in code construction, such as mistyping a keyword, deleting essential punctuation, or using an opening bracket without a matching closing brackets

6.1.1 Problem

Here the compiler throws an error saying ‘)’ expected as the code is not closed after making an anonymous class for ActionListener at the argument.



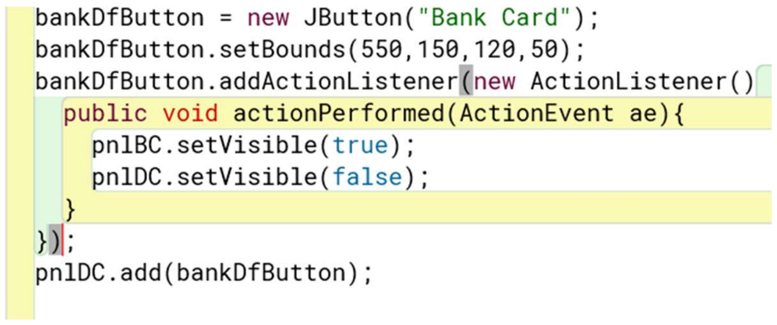
```
//button to open bank card panel
bankDfButton = new JButton("Bank Card");
bankDfButton.setBounds(550,150,120,50);
bankDfButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent ae){
        pnlBC.setVisible(true);
        pnlDC.setVisible(false);
    }
};
')' expected

//button to display the data entered in debit card
```

Figure 31: Syntax error problem

6.1.2 Solution

As the compiler shows me the location of error it was easy to find where error occurred and also the compiler suggested me to close the bracket and the error was solved.



```
bankDfButton = new JButton("Bank Card");
bankDfButton.setBounds(550,150,120,50);
bankDfButton.addActionListener(new ActionListener()
    public void actionPerformed(ActionEvent ae){
        pnlBC.setVisible(true);
        pnlDC.setVisible(false);
    }
});
pnlDC.add(bankDfButton);
```

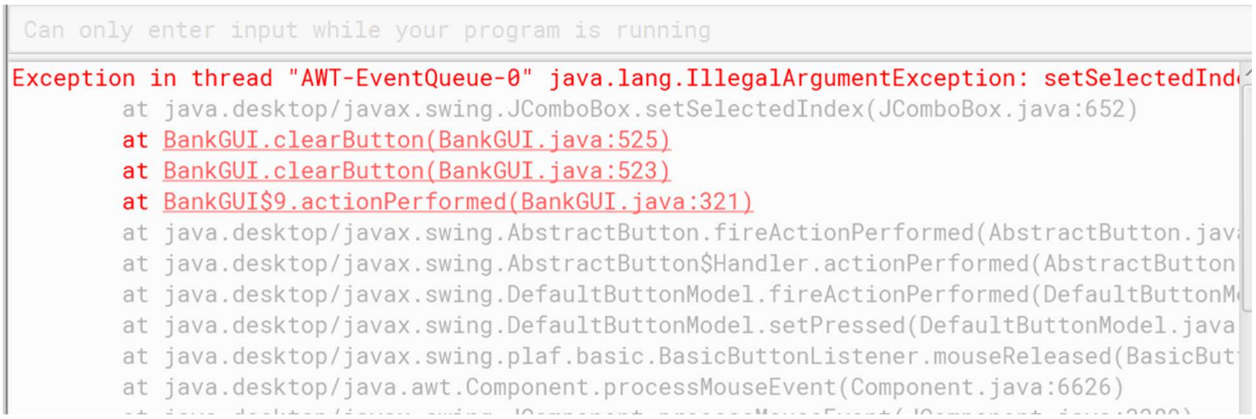
Figure 32: Syntax error solution

6.2 Runtime error

Runtime errors are errors that cause a program to crash while running it because the Java RunTime Enviroment (JRE) detects an operation that cannot be performed hence terminating the flow of program. It is caused by input errors like when user enters the value index which is more than array length hence breaking the runtime.

6.2.1 Problem

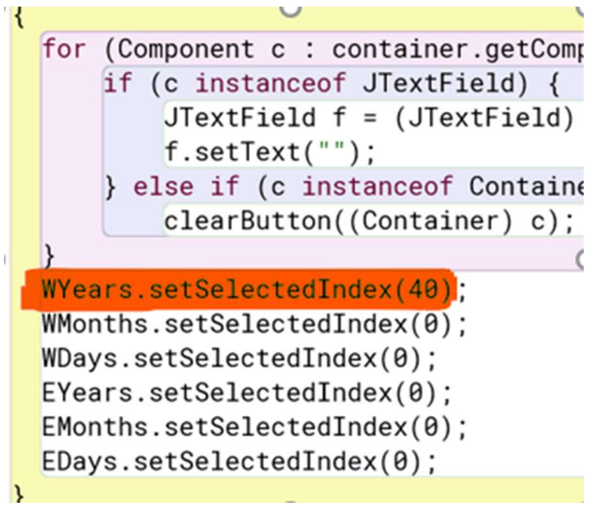
Here, the program compiles without an error but when trying to clear the input data by clicking on clear button the program throws an index out of bound error.



```
Can only enter input while your program is running

Exception in thread "AWT-EventQueue-0" java.lang.IllegalArgumentException: setSelectedInd
        at java.desktop/javax.swing.JComboBox.setSelectedIndex(JComboBox.java:652)
        at BankGUI.clearButton(BankGUI.java:525)
        at BankGUI.clearButton(BankGUI.java:523)
        at BankGUI$9.actionPerformed(BankGUI.java:321)
        at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.jav
        at java.desktop/javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton
        at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonM
        at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java
        at java.desktop/javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicBut
        at java.desktop/java.awt.Component.processMouseEvent(Component.java:6626)
```

Figure 33: Runtime error problem- terminal

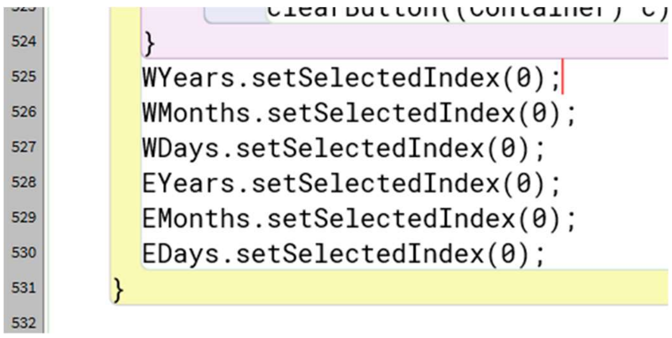


```
{
    for (Component c : container.getComp
        if (c instanceof JTextField) {
            JTextField f = (JTextField)
            f.setText("");
        } else if (c instanceof Containe
            clearButton((Container) c);
    }
    WYears.setSelectedIndex(40);
    WMonths.setSelectedIndex(0);
    WDays.setSelectedIndex(0);
    EYears.setSelectedIndex(0);
    EMonths.setSelectedIndex(0);
    EDays.setSelectedIndex(0);
}
```

Figure 34: Runtime error problem - code

6.2.2 Solution

The Problem was shown to be at line 525 where set selected index was 40. To fix the solution we have to replace the 40 with required number for resetting the combo box which is 0.



```
        clearButton((Container) c)
    }
    WYears.setSelectedIndex(0);
    WMonths.setSelectedIndex(0);
    WDays.setSelectedIndex(0);
    EYears.setSelectedIndex(0);
    EMonths.setSelectedIndex(0);
    EDays.setSelectedIndex(0);
}
```

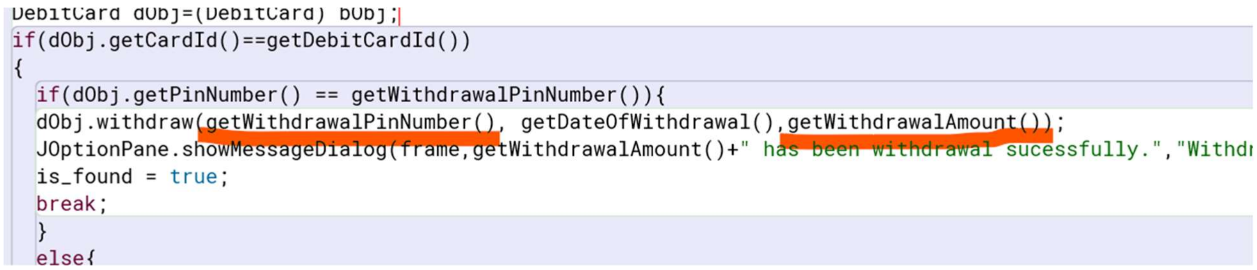
Figure 35: Runtime error solution

6.3 Logical error

Logic errors occur when a program fails to perform as intended. This type of error can occur for a variety of reasons. A logic error is a type of runtime error that can cause a program to produce incorrect output. It may also cause the program to crash while in use.

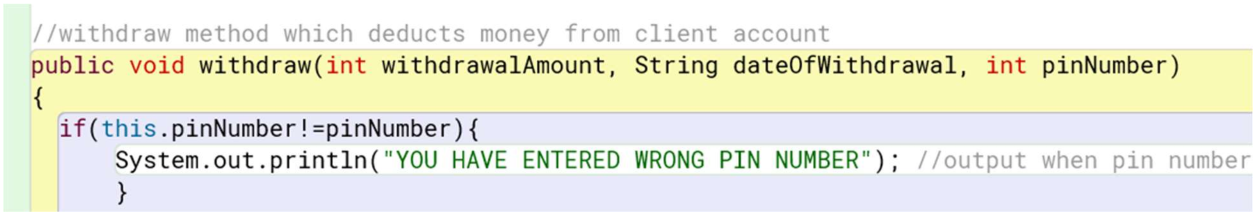
6.3.1 Problem

The program runs without any errors and can even add debit card but while trying to withdraw it doesn't run as intended and after looking and searching to find the error I have made on withdrawal method I found that the arguments that was passed to withdraw method of debit card was not same while calling it on BankGUI. The place of pin number and withdrawal amount was incorrect hence printing wrong pin number message at terminal.



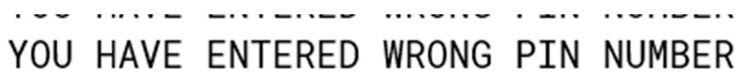
```
DebitCard dObj=(DebitCard) bObj;
if(dObj.getCardId()==getDebitCardId())
{
  if(dObj.getPinNumber() == getWithdrawalPinNumber()){
  dObj.withdraw(getWithdrawalPinNumber(), getDateOfWithdrawal(),getWithdrawalAmount());
  JOptionPane.showMessageDialog(frame,getWithdrawalAmount()+" has been withdrawal sucessfully.","Withdr
  is_found = true;
  break;
  }
  else{
```

Figure 36: Logical error arguments BankGUI



```
//withdraw method which deducts money from client account
public void withdraw(int withdrawalAmount, String dateOfWithdrawal, int pinNumber)
{
  if(this.pinNumber!=pinNumber){
      System.out.println("YOU HAVE ENTERED WRONG PIN NUMBER"); //output when pin number
      }
```

Figure 37: Logical error DebitCard arguments

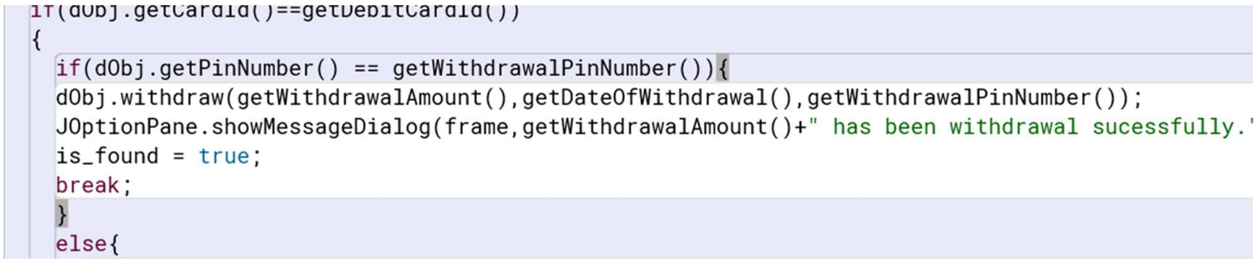


```
YOU HAVE ENTERED WRONG PIN NUMBER
```

Figure 38: Logical error problem -terminal

6.3.2 Solution

It was little longer to find compared to other errors as it doesn’t really gives any clue about the location of error but eventually I found it out and then changed the arguments to its respective location and the balance started to withdraw.



```
if(dObj.getCardId()==getDebitCardId())
{
  if(dObj.getPinNumber() == getWithdrawalPinNumber()){
  dObj.withdraw(getWithdrawalAmount(),getDateOfWithdrawal(),getWithdrawalPinNumber());
  JOptionPane.showMessageDialog(frame,getWithdrawalAmount()+" has been withdrawal sucessfully.'
  is_found = true;
  break;
  }
  else{
```

Figure 39: Logical error solution

7. Conclusion

As the coursework itself was a massive leap of programming for us it was harder to manage time and focus on perfecting the swing and AWT components which are used in this coursework file. This project was done while also learning about the java swing components which had me researching all night long and focusing my time on this large code. There were also many logical and runtime errors which also took a large chunk of my time. The effort alone was not solving as I lacked in my knowledge while starting to code this program. Many hurdles and takebacks later I was finally able to have a decent looking GUI which consisted of labels and text field that could be used to enter the data.

After finally making a GUI there were bunch of Action listener and validation still unfinished which for me was a massive problem to face. Learning new things and different styles to face those problems especially from different website and e-books(like Java Swing by Marc Loy and Robert Eckstein) and many other books in library I was starting to solve those problems one by one. Some of the greater concepts of this program like adding the event handler on add debit card button was solved by asking our very helpful teacher and getting a gist of the idea to solve I further researched on to it and got the solution.

Even the journey was much harder than expected it gave me many skills needed to be a professional programmer one day. The skill of time management and researching was also improved much more than before. The knowledge on Java programming language was also enhanced and gave me a peek on the vast world of Java. After giving my best on this program as it is my last coursework of my 1st year I know how harder I should focus and dedicate myself on coursework and projects that are still to come. This coursework not only gave me skills to code myself on a larger scale with a well working GUI but also mentally improved me and gave me basic skills needed to improve myself on the world of Java programmers.

Finally, I am thankful to my teachers and friends as they helped me overcome some errors and bugs on my program and also thankful to the Java community all around the world as people have faced and solved similar problems and shard the journey and hints to solve them. The E-books about java and books on beginner’s guide on Swing and AWT was very helpful for me to learn and code this project and make it fruitful of all the hard work, patience and the time I spent on this program.

8. Bibliography

Hartman, J. (2022, December 31). *java-platform*. Retrieved from Gruru99: https://www.guru99.com/java-platform.html

JavaTPoint. (n.d.). *uml-class-diagram*. Retrieved from javatpoint: https://www.javatpoint.com/uml-class-diagram

Manning® Java Swing, Second Edition. (2003). In M. Robinson, *Manning® Java Swing, Second Edition* (p. 912). Manning Publicatons.

Marc Loy , Robert Eckstien. (2002). Java Swing. In *Java Swing* (p. 1252). "O'Reilly Media, Inc".

9. Appendix

```
/**
 * This is BankGUI class which holds all the methods and components for GUI
 * @author Anjan Khadka
 * @version 19.0.1
 */




import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.JPanel;
import javax.swing.JLabel;
import javax.swing.border.TitledBorder;

import java.awt.Color;
import java.awt.Component;
import java.awt.Container;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import java.util.ArrayList;
```

```
public class BankGUI{

    private JFrame frame;
    //Bank Card variables
    private JPanel pnlBC;
    private                                                                    JLabel
welcomeLabel,fillLabel,balanceAmtLabl,issuerLabel,bankAccLabel,clientNmLabel;
    private JTextField BAmtf,IBtf,BActf,CNatf;
    private JButton CredBCButton,DebBCutton;



    // Debit Card variables
    private JPanel pnlDC;
    private JTextField DC_Add_IDtf,DCIDtf,PNtf,PNWtf,WAmtf;
    private JComboBox<String> WYears,WMonths,WDays;
    private                                                                    JLabel
addDebCardLabel,pinNumLabel,withAmtLabel,DOW_Label,wthCardIdLabel,wthPinNum
Label;
    private JButton withdrawButton,addDebitCardButton,credDfButton,bankDfButton;



    // Credit Card variables
    private JPanel pnlCC;
    private JTextField CC_Add_CIDtf,CVCtf,IRtf,CLtf,GPtf,CCIDtf,CancelCreditIDtf;
    private JComboBox<String> EYears,EMonths,EDays;
    private                                                                    JLabel
addCredCardidLabel,cvcNumLabel,interestLabel,DOE_Label,credLimitLabel,graceLabe
l,
    setCreLim_CId_label,cancelCC_CId_Label;
    private                                                                   JButton
addCreditCardButton,setCreditLimiButton,cancelCCButton,debCfButton,bnkCfButton,
bankCardClearButton,debitDisplayButton,debitClearButton,creditDisplyButton,creditCle
arButton;


    //variables for checking invalid integers
    public final static int INVALID = -1;

    //Arraylist of bank card to store array objects
    ArrayList<BankCard> cardList = new ArrayList<BankCard>();


    // Instance List varibables for year, month and days to use in date combobox
    private                    String[]                    years                    =
{"Year","2020","2021","2022","2023","2024","2025","2026","2027"};
    private                    String[]                    months                   =
{"Month","January","February","March","April","May","June","July","August",
    "September","October","November","December"};
    private                    String[]                    days                     =
{"Day","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","32"};


    //Constructor of BankGUI
    public BankGUI(){
      initFrame();
      initBankCard();
      initDebitCard();
      initCreditCard();

    }
```

```
  //Method for Creating a Frame
  public void initFrame() {
    frame = new JFrame("Bank GUI");
    frame.setSize(900,900);
    frame.setLayout(null);
    frame.setResizable(false);
    frame.setLocationRelativeTo(null);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }

  //Bank                                    Card                                    Panel
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
  public void initBankCard() {
    pnlBC = new JPanel();
    pnlBC.setLayout(null);
    pnlBC.setLocation(50,30);
    pnlBC.setSize(800, 750);
    pnlBC.setBackground(Color.CYAN);
    TitledBorder borderBank = new TitledBorder("BankCard");
    borderBank.setTitleJustification(TitledBorder.CENTER);
    borderBank.setTitleFont(new Font("Arial", Font.BOLD,24));
    pnlBC.setBorder(borderBank);
    frame.add(pnlBC);

    //Labelling and giving text fields to enter the data
    welcomeLabel = new JLabel("Welcome to Bank GUI");
    welcomeLabel.setBounds(140,50,600,80);
    welcomeLabel.setForeground(Color.RED);
    welcomeLabel.setFont(new Font("Futura", Font.BOLD, 45)); // adding font style to the
label
    pnlBC.add(welcomeLabel);
    //label to show a large text on bank card
    fillLabel = new JLabel("Fill all fields before clicking card type.");
    fillLabel.setBounds(70,470,600,80);
    fillLabel.setForeground(Color.RED);
    fillLabel.setFont(new Font("Futura",Font.PLAIN,30)); // adding font style
    pnlBC.add(fillLabel);

    //Bank Card  BalanceAmount
    balanceAmtLabl = new JLabel("Balance Amount");
    balanceAmtLabl.setFont(new Font("Ariel Black", Font.PLAIN, 15)); // adding font style
to labels
    balanceAmtLabl.setBounds(70,130,120,50);
    pnlBC.add(balanceAmtLabl);

    BAmtf = new JTextField();
    BAmtf.setBounds(200,130,170,50);
    pnlBC.add(BAmtf);

    //BankCard IssuerBank
    issuerLabel = new JLabel("Issuer Bank");
    issuerLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
    issuerLabel.setBounds(70,180,120,70);
    pnlBC.add(issuerLabel);

    IBtf = new JTextField();
    IBtf.setBounds(200,190,170,50);
    pnlBC.add(IBtf);

    //BankCard BankAccount
    bankAccLabel = new JLabel("Bank Account");
```

```
bankAccLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
bankAccLabel.setBounds(70,240,120,70);
pnlBC.add(bankAccLabel);

BActf = new JTextField();
BActf.setBounds(200,250,170,50);
pnlBC.add(BActf);

//BankCard ClientName
clientNmLabel = new JLabel("Client Name");
clientNmLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
clientNmLabel.setBounds(70,300,120,70);
pnlBC.add(clientNmLabel);

CNatf = new JTextField();
CNatf.setBounds(200,310,170,50);
pnlBC.add(CNatf);


// button made to clear all the textfields in bank card panel
bankCardClearButton = new JButton("Clear");
bankCardClearButton.setBounds(300,680,100,50);
pnlBC.add(bankCardClearButton);
bankCardClearButton.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e){
     clearButton(pnlBC);

   }
});

//button made to open credit card panel
```

```
 CredBCButton = new JButton("Credit Card");
 CredBCButton.setBounds(140,550,120,50);
 CredBCButton.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent ae){
      pnlCC.setVisible(true);
      pnlBC.setVisible(false);
   }
 });
 pnlBC.add(CredBCButton);

 //button made to open debit card panel
 DebBCutton = new JButton("Debit Card");
 DebBCutton.setBounds(300,550,120,50);
 DebBCutton.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent ae){
     pnlDC.setVisible(true);
     pnlBC.setVisible(false);
   }
 });
 pnlBC.add(DebBCutton);

}

//Debit card Panel >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
public void initDebitCard() {
 pnlDC = new JPanel();
 pnlDC.setLayout(null);
 pnlDC.setLocation(50,30);
 pnlDC.setSize(800, 750);
 pnlDC.setBackground(Color.GREEN);
 TitledBorder borderDebit = new TitledBorder("Debit Card");
```

```
borderDebit.setTitleJustification(TitledBorder.CENTER);
borderDebit.setTitleFont(new Font("Arial", Font.BOLD,24));
pnlDC.setBorder(borderDebit);
pnlDC.setVisible(false);
frame.add(pnlDC);

//Labelling and giving text fields to enter the data
//addDebit PinNumber
pinNumLabel = new JLabel("Pin Number");
pinNumLabel.setBounds(50 ,50 , 120, 70);
pinNumLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlDC.add(pinNumLabel);

PNtf = new JTextField();
PNtf.setBounds(170,60,170,50);
pnlDC.add(PNtf);

//addDebit CardID
addDebCardLabel = new JLabel("Card ID");
addDebCardLabel.setBounds(50,110,120,70);
addDebCardLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlDC.add(addDebCardLabel);

DC_Add_IDtf = new JTextField();
DC_Add_IDtf.setBounds(170,120,170,50);
pnlDC.add(DC_Add_IDtf);

//Withdraw WithdrawalAmount
withAmtLabel = new JLabel("WithDrawal Amount");
withAmtLabel.setBounds(50, 280, 200, 70);
withAmtLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
```

```
pnlDC.add(withAmtLabel);

WAmtf = new JTextField();
WAmtf.setBounds(250,290,170,50);
pnlDC.add(WAmtf);

//Withdraw Date of Withdrawal
DOW_Label = new JLabel("Date of Withdrawal");
DOW_Label.setBounds(50, 410, 200, 70);
DOW_Label.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlDC.add(DOW_Label);

//Combo box to enter the date of withdrawal
WYears = new JComboBox<String>(years);
WYears.setBounds(250,430,90,28);
pnlDC.add(WYears);



WMonths = new JComboBox<String>(months);
WMonths.setBounds(350,430,90,28);
pnlDC.add(WMonths);

WDays = new JComboBox<String>(days);
WDays.setBounds(450,430,90,28);
pnlDC.add(WDays);

//withdraw CardID
wthCardIdLabel = new JLabel("Card ID");
wthCardIdLabel.setBounds(70,340,120,70);
wthCardIdLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlDC.add(wthCardIdLabel);
```

```
    DCIDtf = new JTextField();
    DCIDtf.setBounds(250,350,170,50);
    pnlDC.add(DCIDtf);

    //withdraw PinNumber
    wthPinNumLabel = new JLabel("Pin Number");
    wthPinNumLabel.setBounds(50 ,470 , 120, 70);
    wthPinNumLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
    pnlDC.add(wthPinNumLabel);

    PNWtf = new JTextField();
    PNWtf.setBounds(250,480,170,50);
    pnlDC.add(PNWtf);

     // Creating a JLabel to make a line and seprate Debit card panel
    JLabel                    lblspam                    =                    new
JLabel("_________________________________________________________________________
______________________________");
    lblspam.setBounds(10,230,600,50);

    pnlDC.add(lblspam);

    //Button to call the withdrawal method
    withdrawButton=new JButton("Withdraw");
    withdrawButton.setBounds(140, 580, 120, 50);
    pnlDC.add(withdrawButton);
    withdrawButton.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent ae){
        withdrawal();
      }
```

```
});

// button to add debit card
addDebitCardButton = new JButton("Add Debit Card");
addDebitCardButton.setBounds(170,180,170,50);
addDebitCardButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent e){
      addDebitCard();
  }
});
pnlDC.add(addDebitCardButton);

//button to open credit card panel
credDfButton = new JButton("Credit Card");
credDfButton.setBounds(550,60,120,50);
credDfButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent ae){
    pnlCC.setVisible(true);
    pnlDC.setVisible(false);

  }
});
pnlDC.add(credDfButton);

//button to open bank card panel
bankDfButton = new JButton("Bank Card");
bankDfButton.setBounds(550,150,120,50);
bankDfButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent ae){
    pnlBC.setVisible(true);
    pnlDC.setVisible(false);
```

```
      }
    });
    pnlDC.add(bankDfButton);

    //button to display the data entered in debit card
    debitDisplayButton = new JButton("Display");
    debitDisplayButton.setBounds(670, 600, 90, 50);
    debitDisplayButton.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent ae){
          Display();
      }
    });
    pnlDC.add(debitDisplayButton);

    // button to clear the data entered in debit card
    debitClearButton = new JButton("Clear");
    debitClearButton.setBounds(540,600,100,50);
    pnlDC.add(debitClearButton);
    debitClearButton.addActionListener(new ActionListener() {
       public void actionPerformed(ActionEvent e){
         clearButton(pnlDC);
       }
    });

  }




    // Credit Card Frame >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   public void initCreditCard(){
```

```
pnlCC = new JPanel();
pnlCC.setLayout(null);
pnlCC.setLocation(50,30);
pnlCC.setSize(800, 750);
pnlCC.setBackground(Color.ORANGE
);
TitledBorder borderCredit =new TitledBorder("Credit Card");
borderCredit.setTitleJustification(TitledBorder.CENTER);
borderCredit.setTitleFont(new Font("Arial", Font.BOLD,24));
pnlCC.setBorder(borderCredit);
pnlCC.setVisible(false);
frame.add(pnlCC);

// Labelling and creating text fields and combobox for credit card
//addCredit CVCNumber
cvcNumLabel = new JLabel("CVC Number");
cvcNumLabel.setBounds(50,50, 120, 70);
cvcNumLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlCC.add(cvcNumLabel);

CVCtf = new JTextField();
CVCtf.setBounds(170,50,170,50);
pnlCC.add(CVCtf);

//addCredit  InterestRate
interestLabel = new JLabel("Interest Rate");
interestLabel.setBounds(50, 100, 120, 70);
interestLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlCC.add(interestLabel);
```

```
IRtf = new JTextField();
IRtf.setBounds(170,110,170,50);
pnlCC.add(IRtf);

//addCredit DateOfExpiration
DOE_Label = new JLabel("Expiration Date");
DOE_Label.setBounds(50,160,120,70);
DOE_Label.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlCC.add(DOE_Label);

// combo box to enter the date of expiration
EYears = new JComboBox<String>(years);
EYears.setBounds(200,180,90,28);
pnlCC.add(EYears);



EMonths = new JComboBox<String>(months);
EMonths.setBounds(300,180,90,28);
pnlCC.add(EMonths);



EDays = new JComboBox<String>(days);
EDays.setBounds(400,180,90,28);
pnlCC.add(EDays);

//addCredit  CardID
addCredCardidLabel = new JLabel("Card ID");
addCredCardidLabel.setBounds(50, 240, 120, 70);
addCredCardidLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
pnlCC.add(addCredCardidLabel);
```

```
    CC_Add_CIDtf = new JTextField();
    CC_Add_CIDtf.setBounds(170,250,170,50);
    pnlCC.add(CC_Add_CIDtf);

    //label made to separate fields of credit card panel
    JLabel                    lblspam2                    =                    new
JLabel("_________________________________________________________________
_______________________________");
    lblspam2.setBounds(10,300,600,50);
    pnlCC.add(lblspam2);

    //setCreditLimit CreditLimit
    credLimitLabel = new JLabel("Credit Limit");
    credLimitLabel.setBounds(50, 360, 120, 70);
    credLimitLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
    pnlCC.add(credLimitLabel);

    CLtf = new JTextField();
    CLtf.setBounds(170,360,170,50);
    pnlCC.add(CLtf);

    //setCreditLimit GracePeriod
    graceLabel = new JLabel("Grace Period");
    graceLabel.setBounds(50, 410, 120, 70);
    graceLabel.setFont(new Font("Ariel Black", Font.PLAIN, 15));
    pnlCC.add(graceLabel);

    GPtf = new JTextField();
    GPtf.setBounds(170,420,170,50);
    pnlCC.add(GPtf);
    //setCreditLimit CardId
    setCreLim_CId_label = new JLabel("Card ID");
    setCreLim_CId_label.setBounds(50, 470, 120, 70);
    setCreLim_CId_label.setFont(new Font("Ariel Black", Font.PLAIN, 15));
    pnlCC.add(setCreLim_CId_label);

    CCIDtf = new JTextField();
    CCIDtf.setBounds(170,480,170,50);
    pnlCC.add(CCIDtf);

    //label to separate panel
    JLabel                    lblspam3                    =                    new
JLabel("____________________________________________________________________
______________________________");
    lblspam3.setBounds(10,540,600,50);
    pnlCC.add(lblspam3);

    //CancelCreditCard CardID
    cancelCC_CId_Label = new JLabel("Card ID");
    cancelCC_CId_Label.setBounds(50, 600, 120, 70);
    cancelCC_CId_Label.setFont(new Font("Ariel Black", Font.PLAIN, 15));
    pnlCC.add(cancelCC_CId_Label);

    CancelCreditIDtf = new JTextField();
    CancelCreditIDtf.setBounds(170,610,170,50);
    pnlCC.add(CancelCreditIDtf);

    //button to add the data of credit card
    addCreditCardButton=new JButton("Add Credit Card" );
    addCreditCardButton.setBounds(370, 250, 150, 50);
    addCreditCardButton.addActionListener(new ActionListener() {
```

```
  public void actionPerformed(ActionEvent ae){
    addCreditCard();
  }
});
pnlCC.add(addCreditCardButton);

//button to set credit limit
setCreditLimiButton = new JButton("Set Credit Limit");
setCreditLimiButton.setBounds(430,400,270,50);
setCreditLimiButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent ce){
    addCreditLimit();
  }
});
pnlCC.add(setCreditLimiButton);

//button to cancel the credit card
cancelCCButton = new JButton("Cancel Credit Card");
cancelCCButton.setBounds(130,680,270,50);
cancelCCButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent e){
    cancelCreditCard();
  }
});
pnlCC.add(cancelCCButton);

//button to open bank card panel
bnkCfButton = new JButton("Bank Card");
bnkCfButton.setBounds(550,60,120,50);
bnkCfButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent ab){
   pnlBC.setVisible(true);
   pnlCC.setVisible(false);

  }
 });
pnlCC.add(bnkCfButton);

//button to open debit card panel
debCfButton= new JButton("Debit Card");
debCfButton.setBounds(550,150,120,50);
debCfButton.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent ad){
    pnlDC.setVisible(true);
    pnlCC.setVisible(false);
  }
 });
pnlCC.add(debCfButton);

 //button to display the data of credit card
creditDisplyButton = new JButton("Display");
creditDisplyButton.setBounds(500, 600, 90, 50);
creditDisplyButton.addActionListener(new ActionListener(){
  public void actionPerformed(ActionEvent ae){
     Display();
  }
});
pnlCC.add(creditDisplyButton);

//button to clear the data of credit card
creditClearButton = new JButton("Clear");
creditClearButton.setBounds(600,600,100,50);
```

```
      pnlCC.add(creditClearButton);
      creditClearButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e){
            clearButton(pnlCC);
        }
      });
    }

//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
      // clearButton method to clear fields
    public void clearButton(Container container)
    {
      for (Component c : container.getComponents()) {
          if (c instanceof JTextField) {
              JTextField f = (JTextField) c;
              f.setText("");
          } else if (c instanceof Container)
              clearButton((Container) c);
      }
      WYears.setSelectedIndex(0);
      WMonths.setSelectedIndex(0);
      WDays.setSelectedIndex(0);
      EYears.setSelectedIndex(0);
      EMonths.setSelectedIndex(0);
      EDays.setSelectedIndex(0);
    }

    //Add Debit Card
    public void addDebitCard()
    {
      String issuerBank = getIssuerBank();
```

```
        String bankAccount = getBankAccount();
        String clientName = getClientName();
        int balanceAmount = getBalanceAmount();
        int cardID = getAddDebitCardId();
        int pinNumber = getPinNumber();
    if(issuerBank.isEmpty()||     bankAccount.isEmpty()||     clientName.isEmpty()     ||
balanceAmount == INVALID || cardID == INVALID || pinNumber == INVALID){
        JOptionPane.showMessageDialog(frame,"Date  enterd  is  invalid.  Please  fill
appropriate    data    to    add    Debit    Card.","Debit    Card    not
added",JOptionPane.ERROR_MESSAGE);
    }

    else if (checkDebitCardUnique(cardID)){
          cardList.add(new DebitCard(cardID, bankAccount, balanceAmount, issuerBank,
clientName, pinNumber));
          String message = "Card ID:  "+ cardID+ "\nClient Name:  " + clientName+ "\nIssuer
Bank: "+ issuerBank+ "\nBank Account: "+ bankAccount + "\nBalance Amount: "+
balanceAmount + "\nPin Number: "+ pinNumber;
          JOptionPane.showMessageDialog(frame,   message,   "Debit   Card   added
sucessfully",JOptionPane.OK_CANCEL_OPTION);
        }

        else{
          JOptionPane.showMessageDialog(frame,  "Debit  Card  couldn't  be  added.  Try
again!!", "Card not added",
           JOptionPane.ERROR_MESSAGE);
        }



    }
     // method to withdraw by calling withdraw method of debit card
```

```
  public void withdrawal()
  {
   boolean is_found = false;
   if(getDebitCardId() == INVALID || getWithdrawalAmount()==INVALID || getWithdrawalPinNumber()== INVALID || getDateOfWithdrawal().isEmpty())
   {
    JOptionPane.showMessageDialog(frame," The data given was not valid. Check and try again.","Invalid Input",JOptionPane.ERROR_MESSAGE);
   }
   else
   {
     JOptionPane.showMessageDialog(frame," CardId is: "+getDebitCardId()+".\n"+" WithDrawal amount is: "+getWithdrawalAmount()+".\n"+" Pin number is: "+getWithdrawalPinNumber()+".\n"+"Date of Withdrawal: "+getDateOfWithdrawal() ," Inputed Data",JOptionPane.INFORMATION_MESSAGE);
    for(BankCard bObj:cardList)
    {
     if(bObj instanceof DebitCard)
     {
      DebitCard dObj=(DebitCard) bObj;
      if(dObj.getCardId()==getDebitCardId())
      {
       if(dObj.getPinNumber() == getWithdrawalPinNumber()){

dObj.withdraw(getWithdrawalAmount(),getDateOfWithdrawal(),getWithdrawalPinNumber());
       JOptionPane.showMessageDialog(frame,getWithdrawalAmount()+" has been withdrawal sucessfully.","Withdraw sucessfull", JOptionPane.INFORMATION_MESSAGE);
       is_found = true;
       break;
```

```
              }
              else{
                JOptionPane.showMessageDialog(frame,"Given Pin number to withdraw
money was incorrect.","Invalid Input", JOptionPane.ERROR_MESSAGE);
              }
            }
            else{
              JOptionPane.showMessageDialog(frame,"Given card id to withdraw money was
incorrect.","Invalid Input", JOptionPane.ERROR_MESSAGE);
            }

          }

        }
      }
      if(is_found== false){
        JOptionPane.showMessageDialog(frame,"Balance is not
Withdrawn.","Error",JOptionPane.ERROR_MESSAGE);
      }
    }



    //add credit card data
    public void addCreditCard()
    {
      int cardID = getAddCardIDCredit();
      String clientName = getClientName();
      String issuerBank = getIssuerBank();
      String bankAccount = getBankAccount();
      int balanceAmount = getBalanceAmount();
      int cvcNumber = getCVCNumber();
      double interestRate = getInterestRate();
      String expirationDate = getExpirationDate();
      if (cardID == INVALID || clientName.isEmpty() || issuerBank.isEmpty() ||
bankAccount.isEmpty() || cvcNumber == INVALID || interestRate == INVALID ||
expirationDate.isEmpty()) {
       JOptionPane.showMessageDialog(frame,"Fields cannot be empty, Please Fill them
before adding Credit Card","Empty Fields",JOptionPane.ERROR_MESSAGE);
      }
      else if (checkCreditCardUnique(cardID)){
       cardList.add(new CreditCard(cardID, clientName, issuerBank, bankAccount,
balanceAmount, cvcNumber, interestRate, expirationDate));
       String message = "Card ID:  "+ cardID+ "\nClient Name:  " + clientName+ "\nIssuer
Bank: "+ issuerBank+ "\nBank Account: "+ bankAccount + "\nBalance Amount: "+
balanceAmount + "\nCVC Number: "+ cvcNumber + "\nInterest Rate: " + interestRate +
"\nExpiration Date: " + expirationDate;
       JOptionPane.showMessageDialog(frame, message, "Credit Card added
sucessfully",JOptionPane.OK_CANCEL_OPTION);
        }
      else{
       JOptionPane.showMessageDialog(frame,"Credit Card could't be added."+"\n Fill all
the fields correctly.","Credit Card not added",JOptionPane.ERROR_MESSAGE);
      }
    }

    //cancel credit card
    public void cancelCreditCard(){
      if(getCancelCreditCardID()!= INVALID){
        for(BankCard bObj: cardList)
        {
         if(bObj instanceof CreditCard)
        {
          CreditCard cObj = (CreditCard) bObj;
          if(cObj.getCardId() == getCancelCreditCardID()){
            if(cObj.getIsGranted() == true){
             cObj.cancelCreditCard();
             JOptionPane.showMessageDialog(frame,"Credit card is cancelled successfully.","Cancel CreditCard",JOptionPane.OK_OPTION);
            }
            else{
              JOptionPane.showMessageDialog(frame,"The Credit card is not Granted. Set credit limit first.","Credit Card not cancelled",JOptionPane.ERROR_MESSAGE);

            }
          }

        else{
            JOptionPane.showMessageDialog(frame,"The card Id provided did not match. Try again.","Wrong Card ID",JOptionPane.ERROR_MESSAGE);
          }
        }
       }
     }

     else{
       JOptionPane.showMessageDialog(frame, "Object of Credit Card not found.", "CreditCard not added",JOptionPane.ERROR_MESSAGE);

     }
   }

   // check if debit card  Card id is unique
```

```
    public boolean checkDebitCardUnique(int cardID)
    {
      boolean isUnique = true;
      for (BankCard bObj : cardList){
        if(bObj instanceof DebitCard){
          DebitCard dc = (DebitCard) bObj;
          if(dc.getCardId() == cardID){
            JOptionPane.showMessageDialog(frame, "Card ID is already added. Please add
different ID.","Card ID not unique",JOptionPane.WARNING_MESSAGE);
            isUnique = false;
            break;
          }
      }
    }
      return isUnique;

  }
    //method to check credit card's card id is unique
    public boolean checkCreditCardUnique(int cardid)
    {
      boolean isUnique = true;
      for (BankCard bObj : cardList){
       if(bObj instanceof CreditCard){
         CreditCard cc = (CreditCard) bObj;
        if(cc.getCardId() == cardid)
        {
          JOptionPane.showMessageDialog(frame, "Card ID is already added. Please add
different ID.","Card ID not unique",JOptionPane.WARNING_MESSAGE);
            isUnique = false;
            break;
        }
      }
    }
      return isUnique;

    }



    // method to set credit limit
    public void addCreditLimit(){
      int cardID = getCardIDCredit();
      int creditLimit = getCreditLimit();
      int GracePeriod = getGracePeriod();
      for(BankCard obj2 : cardList){
        if(obj2 instanceof CreditCard){

          CreditCard cc = (CreditCard) obj2;
          if(cc.getCardId() == cardID){
            JOptionPane.showMessageDialog(frame, "Credit Limit: "+creditLimit+"\nGrace
Period: "+GracePeriod ,"Credit Limit Added",JOptionPane.OK_CANCEL_OPTION);
            cc.setCreditLimit(creditLimit, GracePeriod);

          }
          else{
            JOptionPane.showMessageDialog(frame, "The Card ID provided doesn't
match."+"\n Credit Limit cannot be set","Credit Limit not
set",JOptionPane.ERROR_MESSAGE);
          }
        }
      }

    }
```

```
  // Display method for display buttons
   public void Display()
   {
   for (BankCard obj : cardList){
     if(obj instanceof DebitCard){
     DebitCard dc = (DebitCard) obj;
     dc.disout();
     JOptionPane.showMessageDialog(frame,"The details of Debit card is printed in the
terminal. Please check there.","Display
Information",JOptionPane.INFORMATION_MESSAGE);
     }
     else if(obj instanceof CreditCard){
       CreditCard cc = (CreditCard) obj;
       cc.disout();
       JOptionPane.showMessageDialog(frame,"The details of Credit card is printed in the
terminal. Please check there.","Display
Information",JOptionPane.INFORMATION_MESSAGE);
     }
     else{
       JOptionPane.showMessageDialog(frame,"Cannot find the object in cardList. Add a
card first.","Object not set",JOptionPane.ERROR_MESSAGE);
     }
   }
  }
```

```
    //BankCard                                    getter                                    methods
.............................................................................................
    //BankCard BalanceAmount
    public int getBalanceAmount(){
      String BalanceAmountText = BAmtf.getText().trim();
      int BalanceAmount = INVALID;
      try{

        BalanceAmount = Integer.parseInt(BalanceAmountText);
        if(BalanceAmount < 0 ){
         BalanceAmount = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Balance Amount", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);

        }

        if (BalanceAmountText.isEmpty()){
          JOptionPane.showMessageDialog(frame,  "Please  fill  BalanceAmount  Field",
"Error", JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error  "+  e.getMessage(),  "Error  box",
JOptionPane.ERROR_MESSAGE);
      }
      return BalanceAmount;

    }

    //BankCard IssuerBank
    public String getIssuerBank(){
      String IssuerBankText = IBtf.getText().trim();
      if (IssuerBankText.isEmpty()){
        JOptionPane.showMessageDialog(frame, "Please fill Issuer Bank Field", "Error",
JOptionPane.WARNING_MESSAGE);
      }
      return IssuerBankText;
    }

    //BankCard BankAccount
    public String getBankAccount(){
      String BankAccountText = BActf.getText().trim();
      if (BankAccountText.isEmpty()){
        JOptionPane.showMessageDialog(frame, "Please fill Bank ACcount Field", "Error",
JOptionPane.WARNING_MESSAGE);
      }
      return BankAccountText;
    }

    //BankCard ClientName
    public String getClientName(){
      String ClientNameText = CNatf.getText().trim();
      if (ClientNameText.isEmpty()){
        JOptionPane.showMessageDialog(frame, "Please fill Client Name Field", "Error",
JOptionPane.WARNING_MESSAGE);
      }
      return ClientNameText;
    }
```

```
  //credit card getter methods
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>

  //addCredit CardId
  public int getAddCardIDCredit(){
    String CardIDText = CC_Add_CIDtf.getText().trim();
    int CardID = INVALID;
    try{
      if (CardIDText.isEmpty()){
       JOptionPane.showMessageDialog(frame, "Please fill CardID Text Field", "Error",
JOptionPane.WARNING_MESSAGE);
     }
     CardID = Integer.parseInt(CardIDText);
     if(CardID <= 0 ){
      CardID = INVALID;
      JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Card ID.", "Invalid Value",JOptionPane.WARNING_MESSAGE);
     }
    }
    catch(NumberFormatException e){
     JOptionPane.showMessageDialog(frame,"Error "+  e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);
    }
    return CardID;
  }

  //addCreditCard cvcNumber
  public int getCVCNumber(){
    String CVCNumberText = CVCtf.getText().trim();
```

```
      int CVCNumber = INVALID;
      try{
        if (CVCNumberText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill CVC Number Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }

        CVCNumber = Integer.parseInt(CVCNumberText);
        if(CVCNumber <= 0 ){
         CVCNumber = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
CVC Number", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error  "+  e.getMessage(),  "Error  box",
JOptionPane.ERROR_MESSAGE);
      }
      return CVCNumber;
    }

    //addCreditCard interestRate
    public double getInterestRate(){
      String InterestRateText = IRtf.getText().trim();
      double InterestRate = INVALID;
      try{
        if (InterestRateText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill Interest Rate Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }
        InterestRate = Double.parseDouble(InterestRateText);
        if(InterestRate <= 0 ){
         InterestRate = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Double number for
Interest Rate", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+  e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);
      }
      return InterestRate;
    }

    //addCreditCard ExpirationDate
    public String getExpirationDate(){
      String date = "";
      String year = EYears.getSelectedItem().toString();
      String month = EMonths.getSelectedItem().toString();
      String day = EDays.getSelectedItem().toString();
      try{
      if(year.equals("Year") || month.equals("Month") || day.equals("Day")) {
         date = null;
         JOptionPane.showMessageDialog(frame, "Please choose the Expiration Date",
"Empty value",
         JOptionPane.WARNING_MESSAGE);
      }
       else {
          date = year + "-" + month + "-" + day;
       }
    }
    catch(Exception e){
      JOptionPane.showMessageDialog(frame,"Error "+  e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);


    }

      return date;
    }


    //setCreditLimit creditlimit
    public int getCreditLimit(){
      String CreditLimitText = CLtf.getText().trim();
      int CreditLimit = INVALID;
      try{
        CreditLimit = Integer.parseInt(CreditLimitText);
        if (CreditLimitText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill Credit Limit Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }
        if(CreditLimit <= 0 ){
         CreditLimit = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Double number for
Credit Limit", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+ e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);

      }
      return CreditLimit;

    }

    //setCreditLimit cardid
    public int getCardIDCredit(){
      String CardIDText = CCIDtf.getText().trim();
      int CardID = INVALID;
      try{
         if (CardIDText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill CardID Text Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }
        CardID = Integer.parseInt(CardIDText);
        if(CardID <= 0 ){
         CardID = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Card ID.", "Invalid Value",JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+ e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);
      }
      return CardID;
    }
    //setcreditlimit graceperiod
    public int getGracePeriod(){
      String GracePeriodText = GPtf.getText().trim();
      int GracePeriod = INVALID;
      try{
        GracePeriod = Integer.parseInt(GracePeriodText);
        if (GracePeriodText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill Grace Period Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }
        if(GracePeriod <= 0 ){
         GracePeriod = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Grace Period", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
        }

      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error  "+   e.getMessage(),  "Error  box",
JOptionPane.ERROR_MESSAGE);
      }
      return GracePeriod;
    }

    //cancelCreditCard cardid
    public int getCancelCreditCardID(){
      String CardIDText = CancelCreditIDtf.getText().trim();
      int CardID = INVALID;
      try{
         if (CardIDText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill CardID Text Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }
        CardID = Integer.parseInt(CardIDText);
        if(CardID <= 0 ){
         CardID = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Card ID.", "Invalid Value",        JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error  "+   e.getMessage(),  "Error  box",
JOptionPane.ERROR_MESSAGE);
      }
      return CardID;
    }


    //                    DebitCard                    getter                    methods
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>

    //addDebitCard pinnumber
    public int getPinNumber(){
      String PinNumberText = PNtf.getText().trim();
      int PinNumber = INVALID;
      try{
        PinNumber = Integer.parseInt(PinNumberText);
        if (PinNumberText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill Pin Number Field", "Error",
JOptionPane.WARNING_MESSAGE);
```

```
        }
        if(PinNumber < 0 ){
         PinNumber = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Pin Number", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+ e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);
      }
      return PinNumber;
    }

    //addDebitCard cardid
    public int getAddDebitCardId(){
      String DebitCardIDText = DC_Add_IDtf.getText().trim();
      int DebitCardID = INVALID;
      try{
         if (DebitCardIDText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill DebitCardID Text Field",
"Error", JOptionPane.WARNING_MESSAGE);
        }
        DebitCardID = Integer.parseInt(DebitCardIDText);
        if(DebitCardID <= 0 ){
         DebitCardID = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Card ID.", "Invalid Value",  JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+ e.getMessage(),"Error box",
JOptionPane.ERROR_MESSAGE);
      }
      return DebitCardID;
    }

    //withdrawal pinNumber
    public int getWithdrawalPinNumber(){
      String PinNumberText = PNWtf.getText().trim();
      int PinNumber = INVALID;
      try{
        PinNumber = Integer.parseInt(PinNumberText);
        if (PinNumberText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill Pin Number Field", "Error",
JOptionPane.WARNING_MESSAGE);
        }
        if(PinNumber < 0 ){
         PinNumber = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Pin Number", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+ e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);
      }
      return PinNumber;
    }
    //withdrawal withdrawalAmount
    public int getWithdrawalAmount(){
      String WithdrawalAmountText = WAmtf.getText().trim();
      int WithdrawalAmount = INVALID;
      try{
        WithdrawalAmount = Integer.parseInt(WithdrawalAmountText);
        if (WithdrawalAmountText.isEmpty()){
          JOptionPane.showMessageDialog(frame, "Please fill Withdrawal Amount Field",
"Error", JOptionPane.WARNING_MESSAGE);
        }
        if(WithdrawalAmount <= 0 ){
         WithdrawalAmount = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Withdrawal", "Invalid Value",
         JOptionPane.WARNING_MESSAGE);
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error "+  e.getMessage(), "Error box",
JOptionPane.ERROR_MESSAGE);
      }
      return WithdrawalAmount;
    }

    //withdrwawal dateofWithdrawal
    public String getDateOfWithdrawal(){
      String date = "";
      String year = WYears.getSelectedItem().toString();
      String month = WMonths.getSelectedItem().toString();
      String day = WDays.getSelectedItem().toString();
      try{
      if(year.equals("Year") || month.equals("Month") || day.equals("Day")) {
         date = null;
      }
      else {
         date = year + "-" + month + "-" + day;
      }
    }
    catch(Exception e){
      JOptionPane.showMessageDialog(frame,"Error  "+    e.getMessage(),"Error  box",
JOptionPane.ERROR_MESSAGE);

    }

      return date;
    }

    //withdrawal cardid
    public int getDebitCardId(){
      String DebitCardIDText = DCIDtf.getText().trim();
      int DebitCardID = INVALID;
      try{
         if (DebitCardIDText.isEmpty()){
          JOptionPane.showMessageDialog(frame,  "Please  fill  DebitCardID  Text  Field",
"Error", JOptionPane.WARNING_MESSAGE);
        }
        DebitCardID = Integer.parseInt(DebitCardIDText);
        if(DebitCardID <= 0 ){
         DebitCardID = INVALID;
         JOptionPane.showMessageDialog(frame, "Enter valid positive Integer number for
Card ID.", "Invalid Value",  JOptionPane.WARNING_MESSAGE);
```

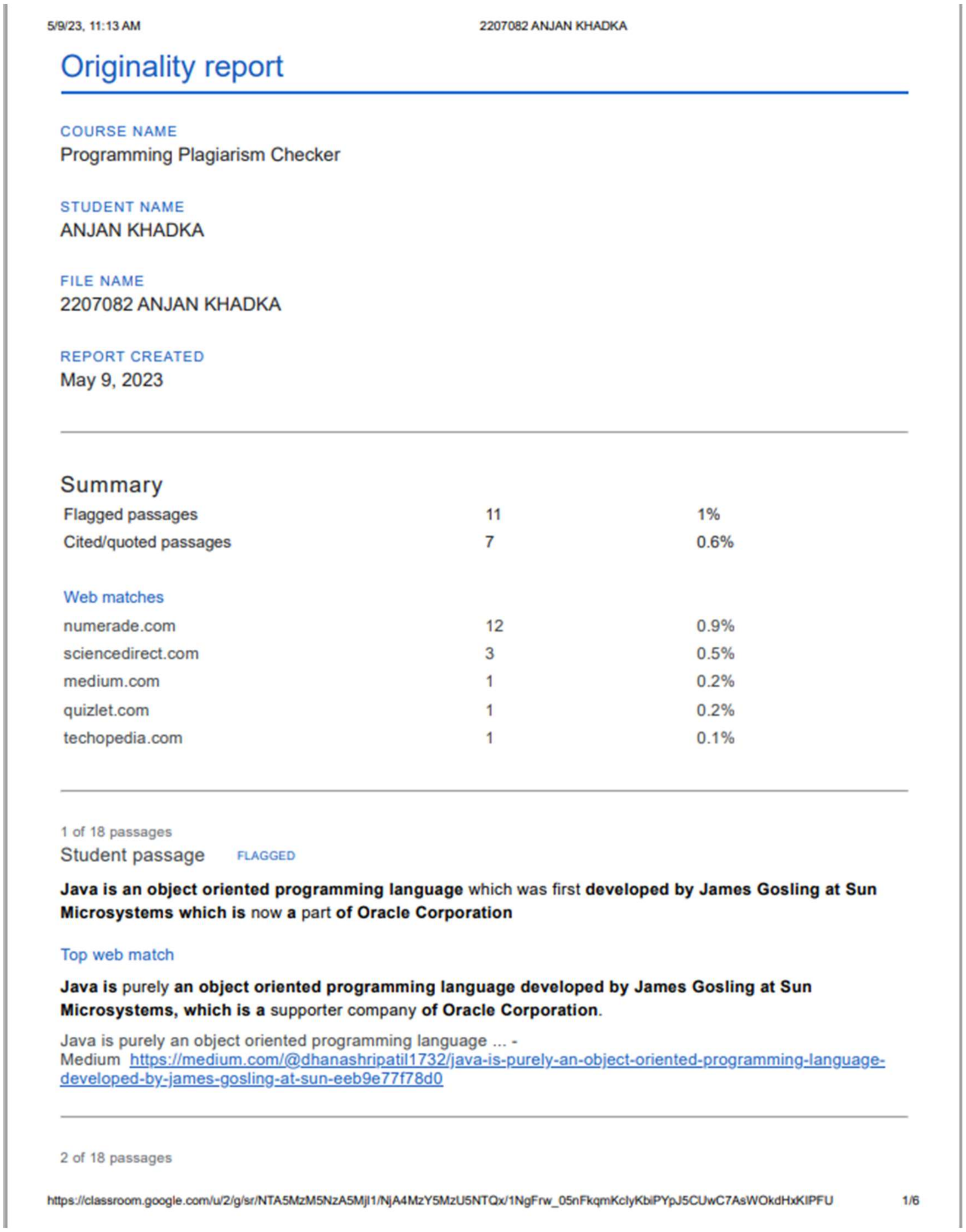
```
        }
      }
      catch(NumberFormatException e){
        JOptionPane.showMessageDialog(frame,"Error  "+   e.getMessage(),"Error  box",
JOptionPane.ERROR_MESSAGE);
      }
      return DebitCardID;
    }


    // main method of BankGUI class
    public static void main(String[] args) {
      // calling the construstor and setting the frame visibility to true
        new BankGUI().frame.setVisible(true);;
    }




}
```

10. Originality Check



5/9/23, 11:13 AM 2207082 ANJAN KHADKA

Originality report

COURSE NAME
Programming Plagiarism Checker

STUDENT NAME
ANJAN KHADKA

FILE NAME
2207082 ANJAN KHADKA

REPORT CREATED
May 9, 2023

Summary

Flagged passages	11	1%
Cited/quoted passages	7	0.6%

Web matches		
numerade.com	12	0.9%
sciencedirect.com	3	0.5%
medium.com	1	0.2%
quizlet.com	1	0.2%
techopedia.com	1	0.1%

1 of 18 passages

Student passage FLAGGED

Java is an object oriented programming language which was first **developed by James Gosling at Sun Microsystems which is** now **a** part **of Oracle Corporation**

Top web match

Java is purely **an object oriented programming language developed by James Gosling at Sun Microsystems, which is a** supporter company **of Oracle Corporation**.

Java is purely an object oriented programming language ... - Medium https://medium.com/@dhanashripatil1732/java-is-purely-an-object-oriented-programming-language-developed-by-james-gosling-at-sun-eeb9e77f78d0

2 of 18 passages

https://classroom.google.com/u/2/g/sr/NTA5MzM5NzA5Mjl1/NjA4MzY5MzU5NTQx/1NgFrw_05nFkqmKclyKbiPYpJ5CUwC7AsWOkdHxKIPFU 1/6

Figure 40: Originality Report