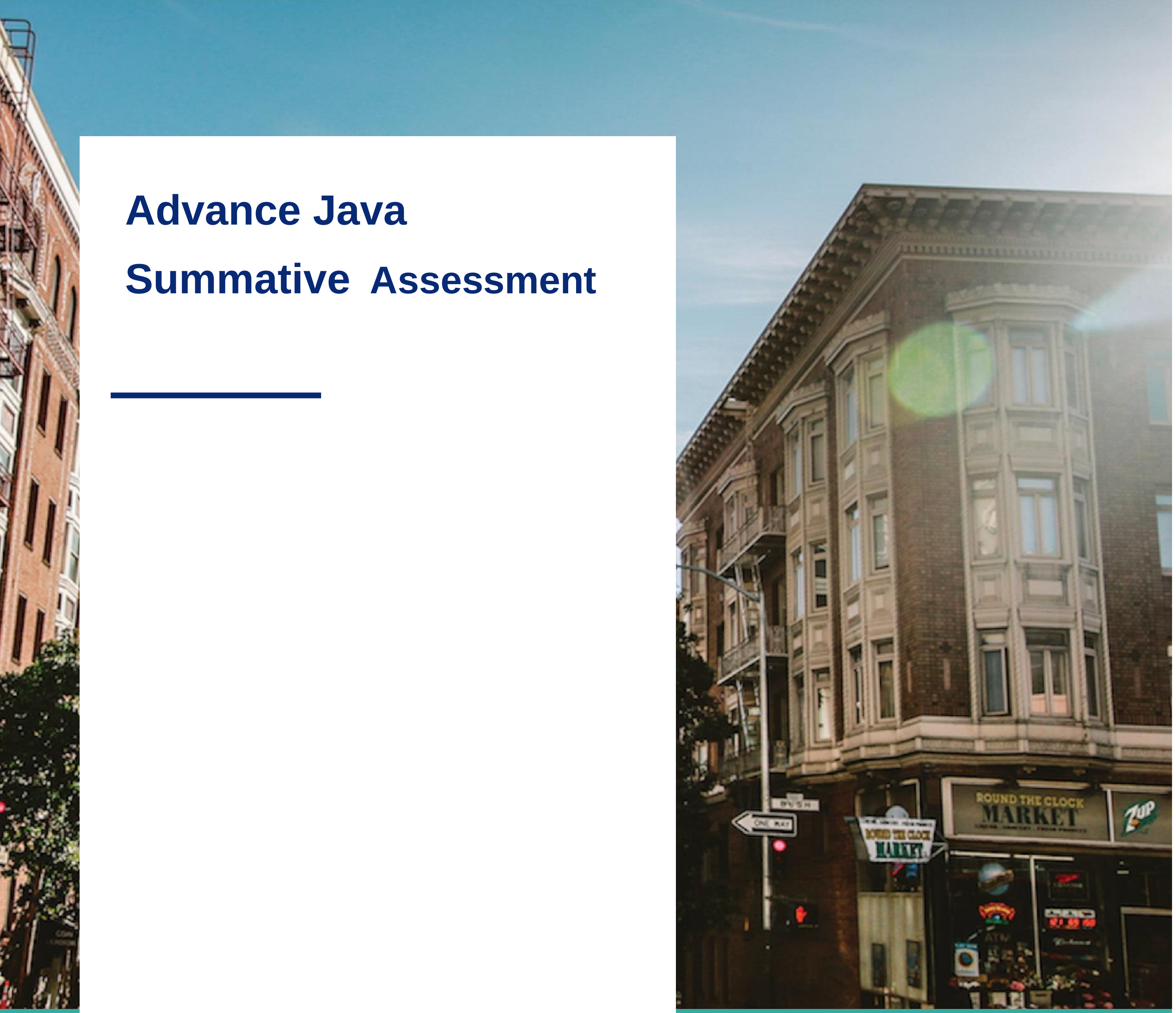


Advance Java

Summative Assessment

OCTOBER 26

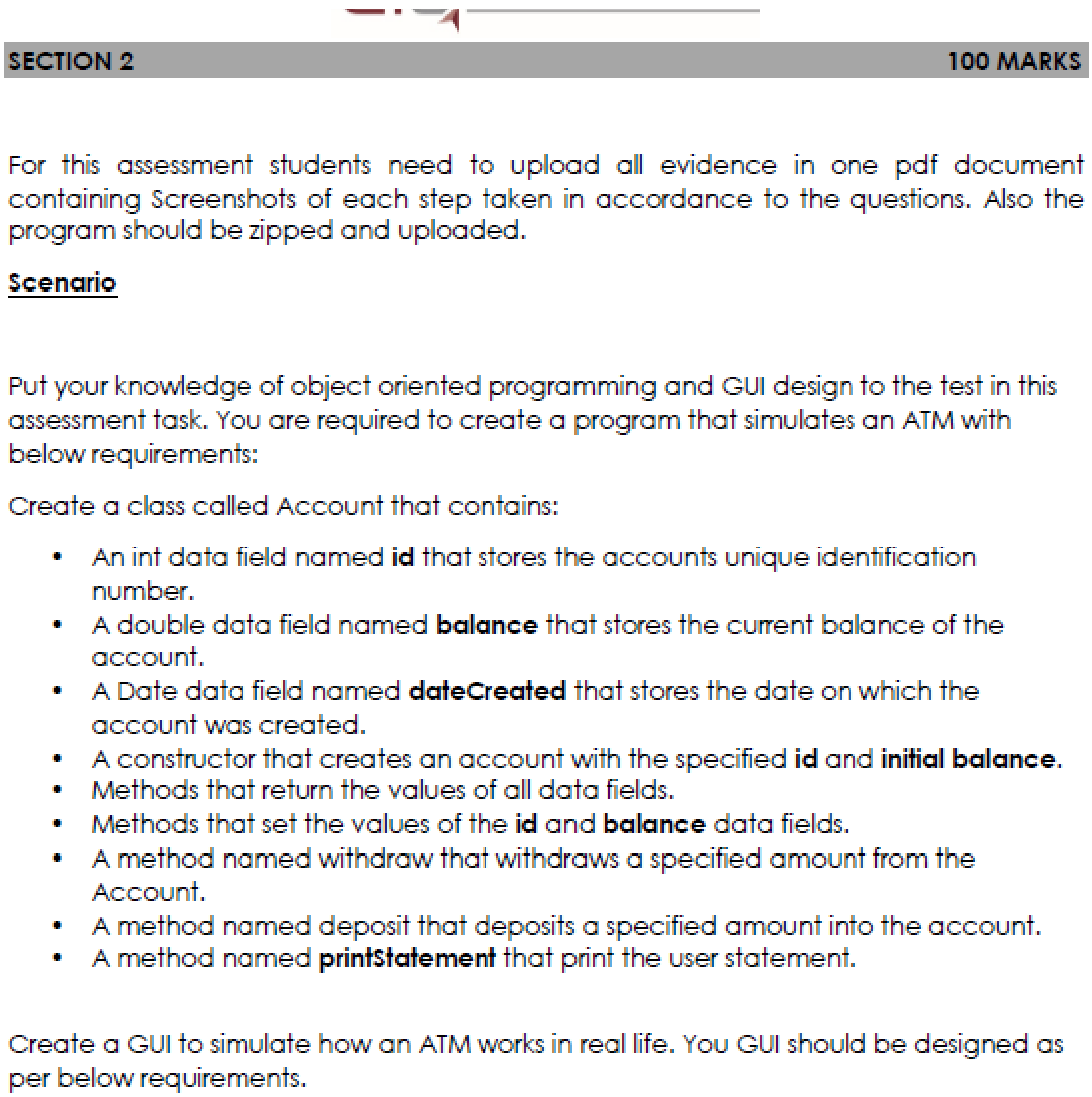
COMPANY NAME

Authored by: Your Name

Logo
Name

Table of Contents

Introduction:



SECTION 2 100 MARKS

For this assessment students need to upload all evidence in one pdf document containing Screenshots of each step taken in accordance to the questions. Also the program should be zipped and uploaded.

Scenario

Put your knowledge of object oriented programming and GUI design to the test in this assessment task. You are required to create a program that simulates an ATM with below requirements:

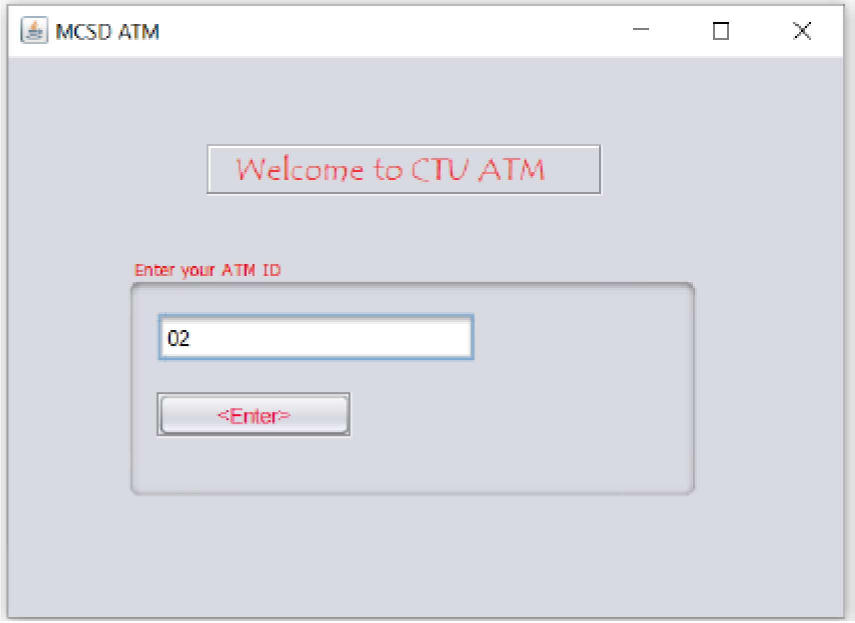
Create a class called Account that contains:

- An int data field named **id** that stores the accounts unique identification number.
- A double data field named **balance** that stores the current balance of the account.
- A Date data field named **dateCreated** that stores the date on which the account was created.
- A constructor that creates an account with the specified **id** and **initial balance**.
- Methods that return the values of all data fields.
- Methods that set the values of the **id** and **balance** data fields.
- A method named withdraw that withdraws a specified amount from the Account.
- A method named deposit that deposits a specified amount into the account.
- A method named **printStatement** that print the user statement.

Create a GUI to simulate how an ATM works in real life. You GUI should be designed as per below requirements.

Window 1

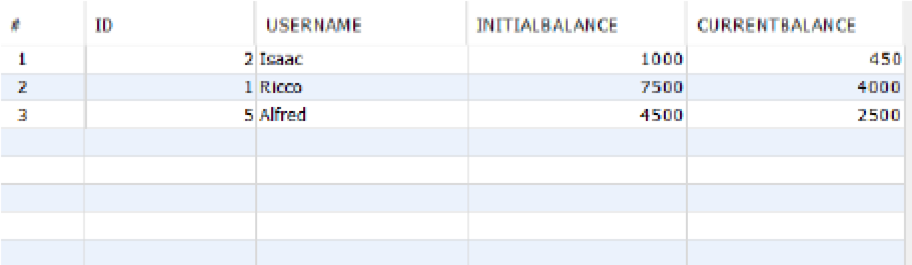
Should have a welcome message, a textbox to receive AN EXISTING ID of the user and a button to access the data base. All users ID should be saved in a data base (Don't use access). Once the Id is verified then the program will give the user access to the main menu window.



Enter your ATM ID

02

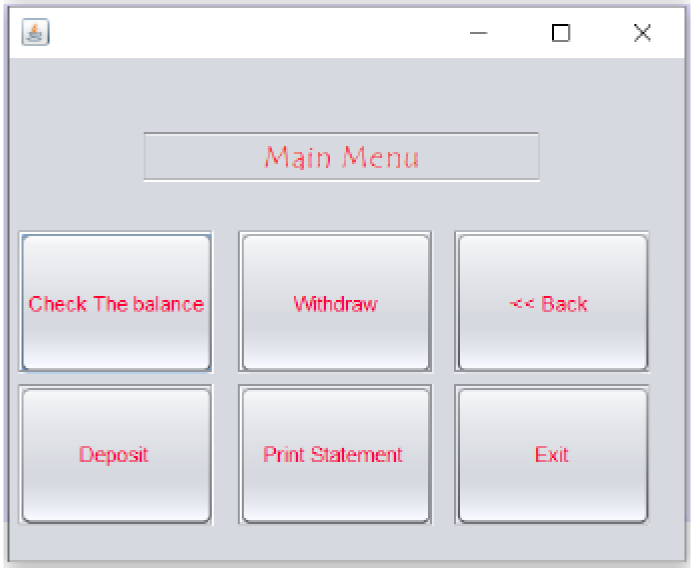
Your database should have the following columns.



#	ID	USERNAME	INITIALBALANCE	CURRENTBALANCE
1	2	Isaac	1000	450
2	1	Ricco	7500	4000
3	5	Alfred	4500	2500

Window 2

Should have a heading stating "Main Menu" and six options as shown below.



Option 1:

Check Balance: This will allow the user to view the current balance. Use **JOptionPane** to display the balance from the data base.

Option 2:

Withdraw: This will provide another window (Do not use JOptionPane) with a textbox to enter the amount of money the user wants to withdraw. The ATM will not allow the user to withdraw an amount greater than the current balance.

Option 3

Deposit: This will provide another window (Do not use JOptionPane) to allow the user to deposit no amount greater than 10.000 should be deposited on the ATM. In a case the user wants to deposit an amount greater than 10.000 a message box should be displayed Stating "No amount greater than 10.000 will be deposited. Please contact the bank".

Option 4

Print: this option should print the current balance, the name of the user, and the current date.

Option 5

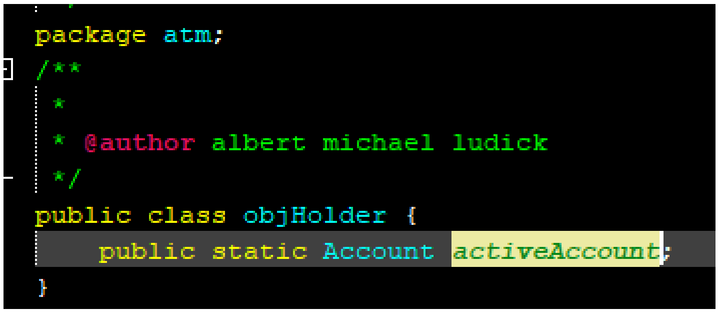
Exit: This option should terminate the program.

Option 6

Back: This option will take back the user to the welcome window

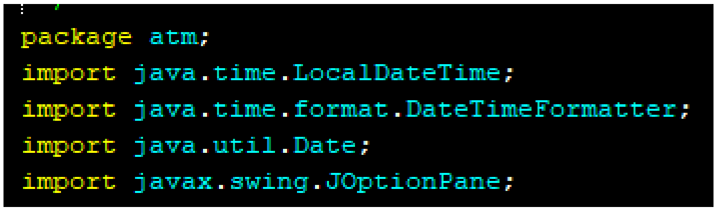
SourceCode:

Objholder Class:

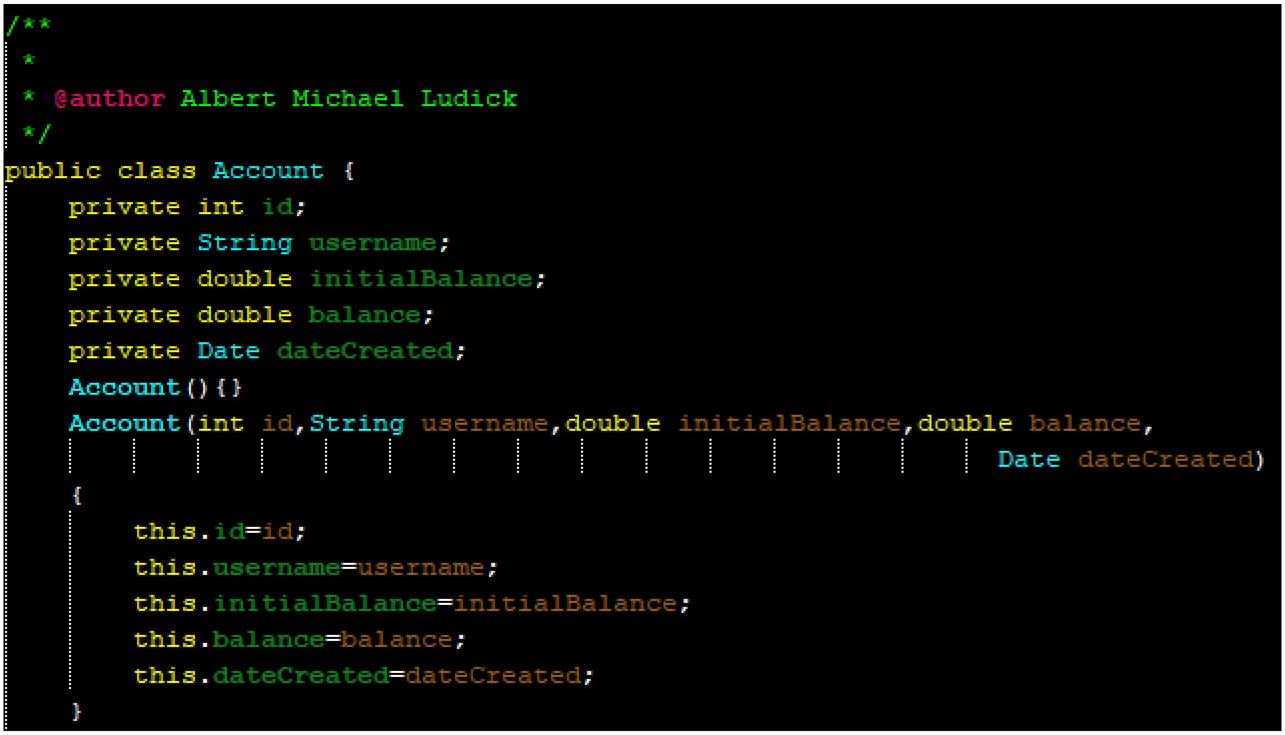


```
package atm;
/**
 *
 * @author albert michael ludick
 */
public class objHolder {
    public static Account activeAccount;
}
```

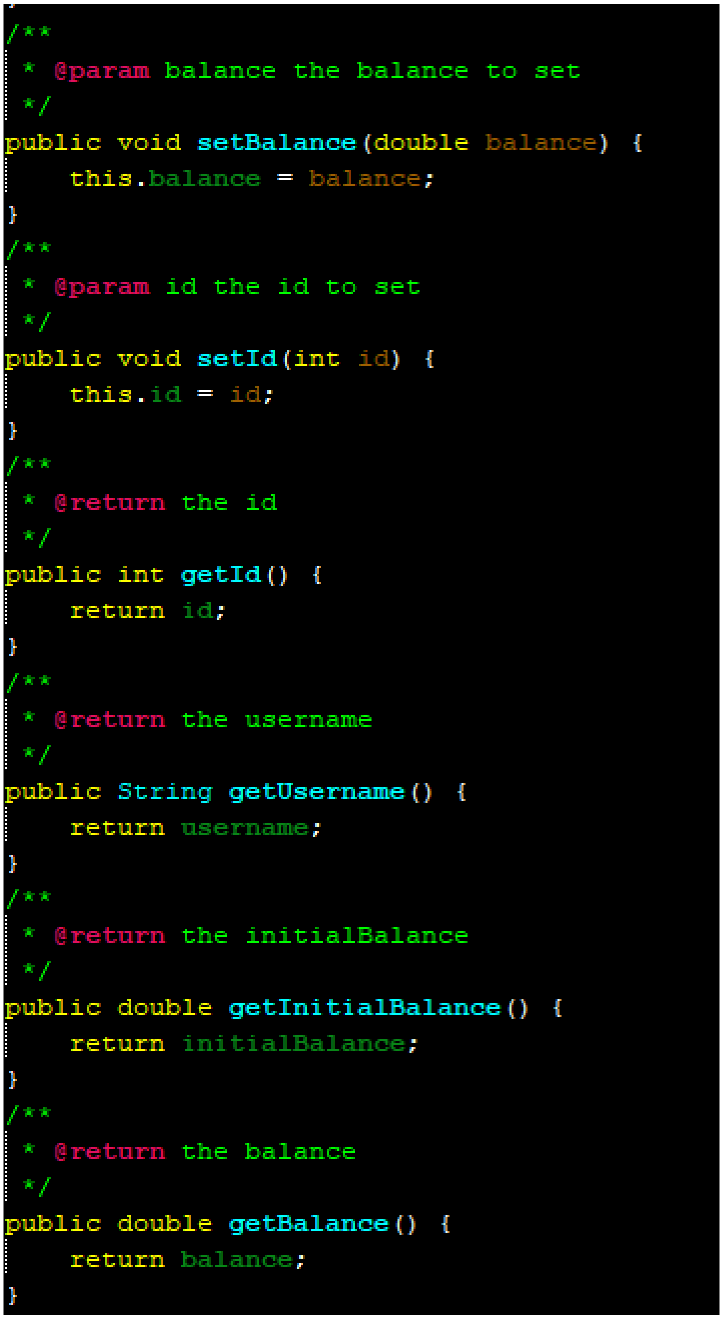
Account class:



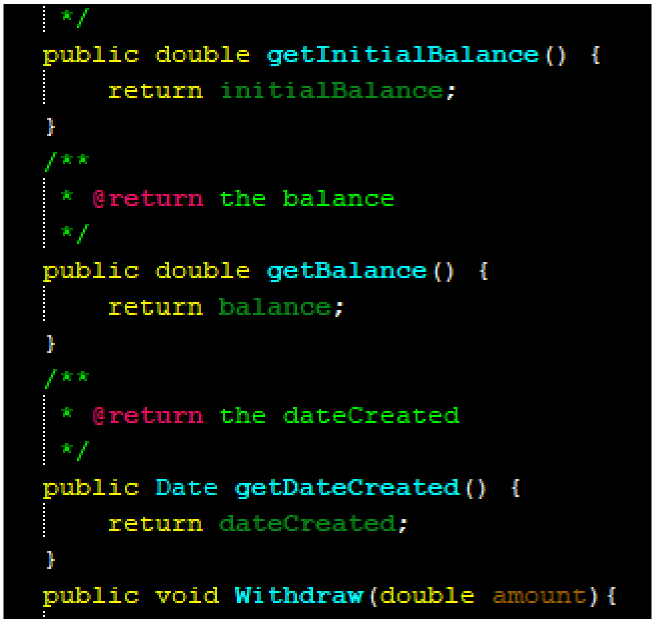
```
package atm;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.Date;
import javax.swing.JOptionPane;
```



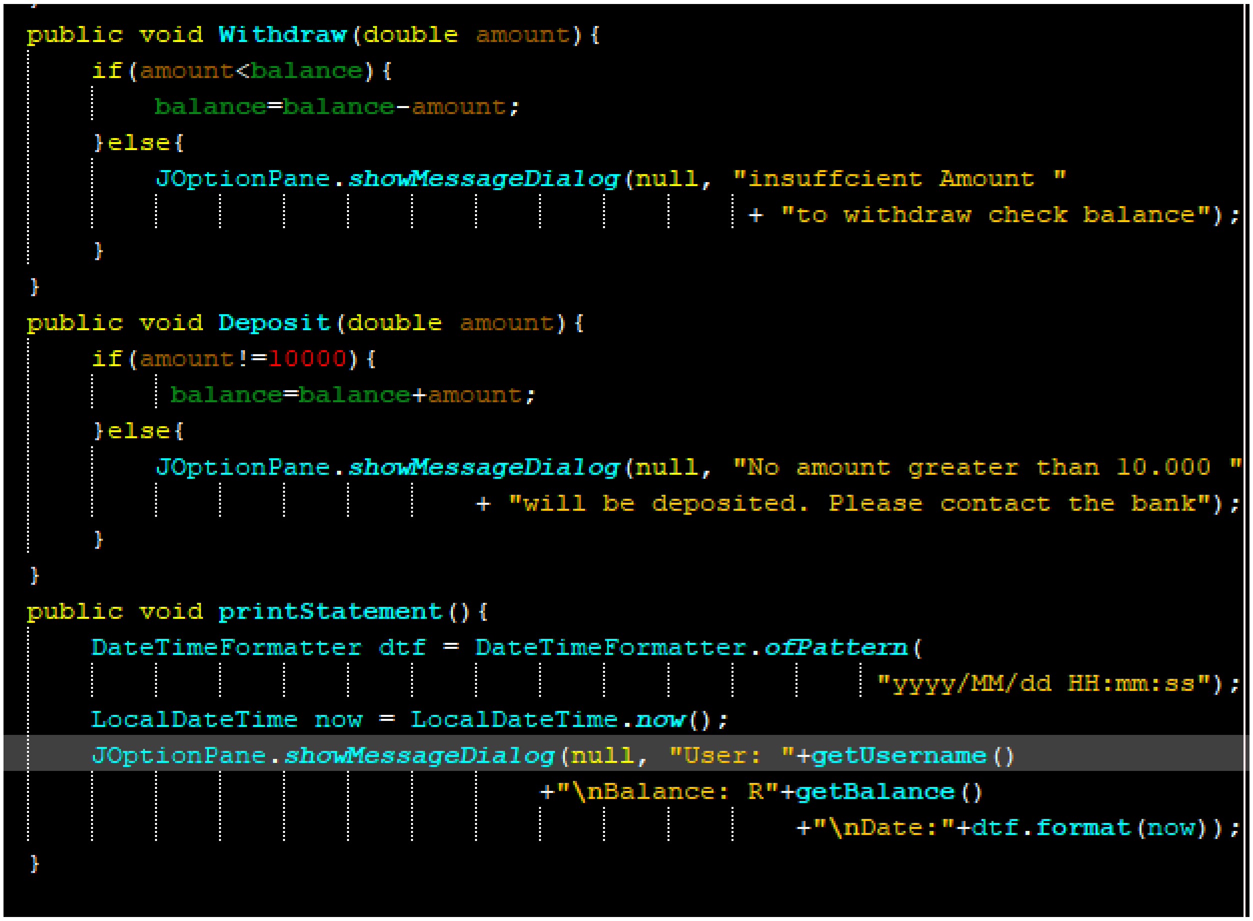
```
/**
 *
 * @author Albert Michael Ludick
 */
public class Account {
    private int id;
    private String username;
    private double initialBalance;
    private double balance;
    private Date dateCreated;
    Account(){}
    Account(int id,String username,double initialBalance,double balance,
                                                         Date dateCreated)
    {
        this.id=id;
        this.username=username;
        this.initialBalance=initialBalance;
        this.balance=balance;
        this.dateCreated=dateCreated;
    }
```



```
/**
 * @param balance the balance to set
 */
public void setBalance(double balance) {
    this.balance = balance;
}
/**
 * @param id the id to set
 */
public void setId(int id) {
    this.id = id;
}
/**
 * @return the id
 */
public int getId() {
    return id;
}
/**
 * @return the username
 */
public String getUsername() {
    return username;
}
/**
 * @return the initialBalance
 */
public double getInitialBalance() {
    return initialBalance;
}
/**
 * @return the balance
 */
public double getBalance() {
    return balance;
}
```

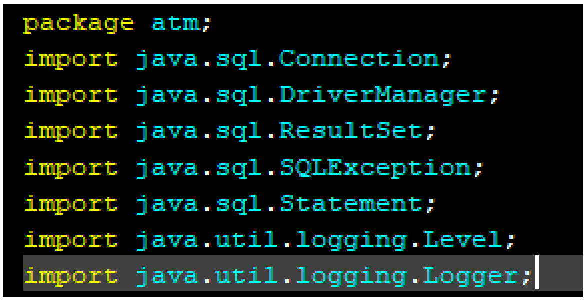


```
 */
public double getInitialBalance() {
    return initialBalance;
}
/**
 * @return the balance
 */
public double getBalance() {
    return balance;
}
/**
 * @return the dateCreated
 */
public Date getDateCreated() {
    return dateCreated;
}
public void Withdraw(double amount){
```

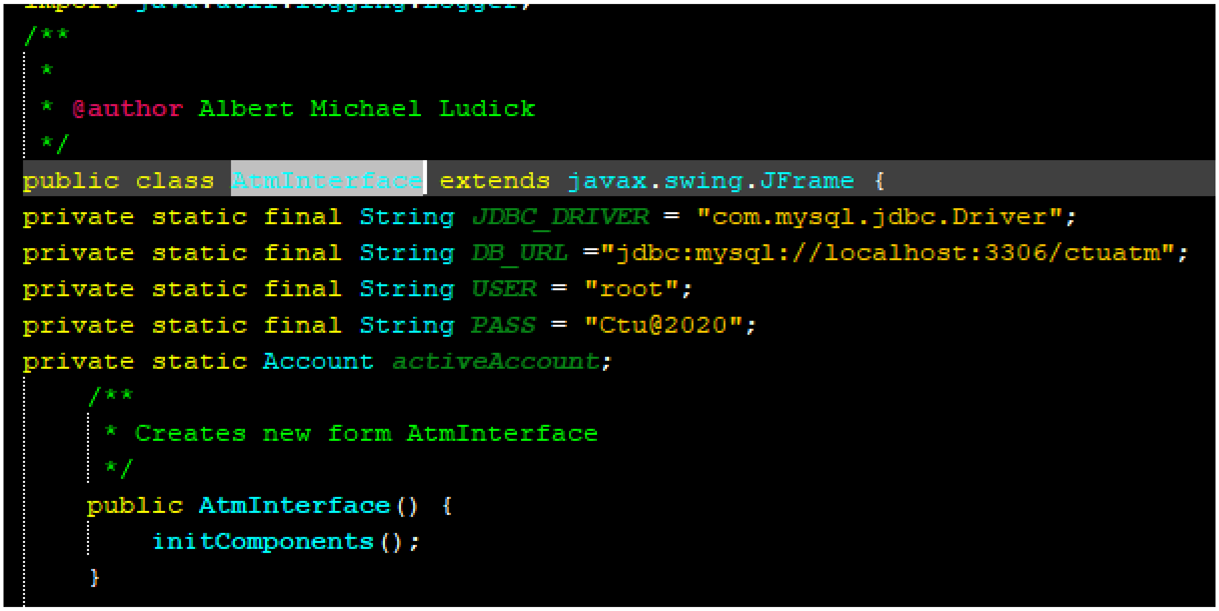


```
public void Withdraw(double amount){
    if(amount<balance){
        balance=balance-amount;
    }else{
        JOptionPane.showMessageDialog(null, "insuffcient Amount "
                                                    + "to withdraw check balance");
    }
}
public void Deposit(double amount){
    if(amount!=10000){
        balance=balance+amount;
    }else{
        JOptionPane.showMessageDialog(null, "No amount greater than 10.000 "
                                    + "will be deposited. Please contact the bank");
    }
}
public void printStatement(){
    DateTimeFormatter dtf = DateTimeFormatter.ofPattern(
                                                        "yyyy/MM/dd HH:mm:ss");
    LocalDateTime now = LocalDateTime.now();
    JOptionPane.showMessageDialog(null, "User:  "+getUsername()
                                        +"\nBalance: R"+getBalance()
                                                +"\nDate:"+dtf.format(now));
}
```

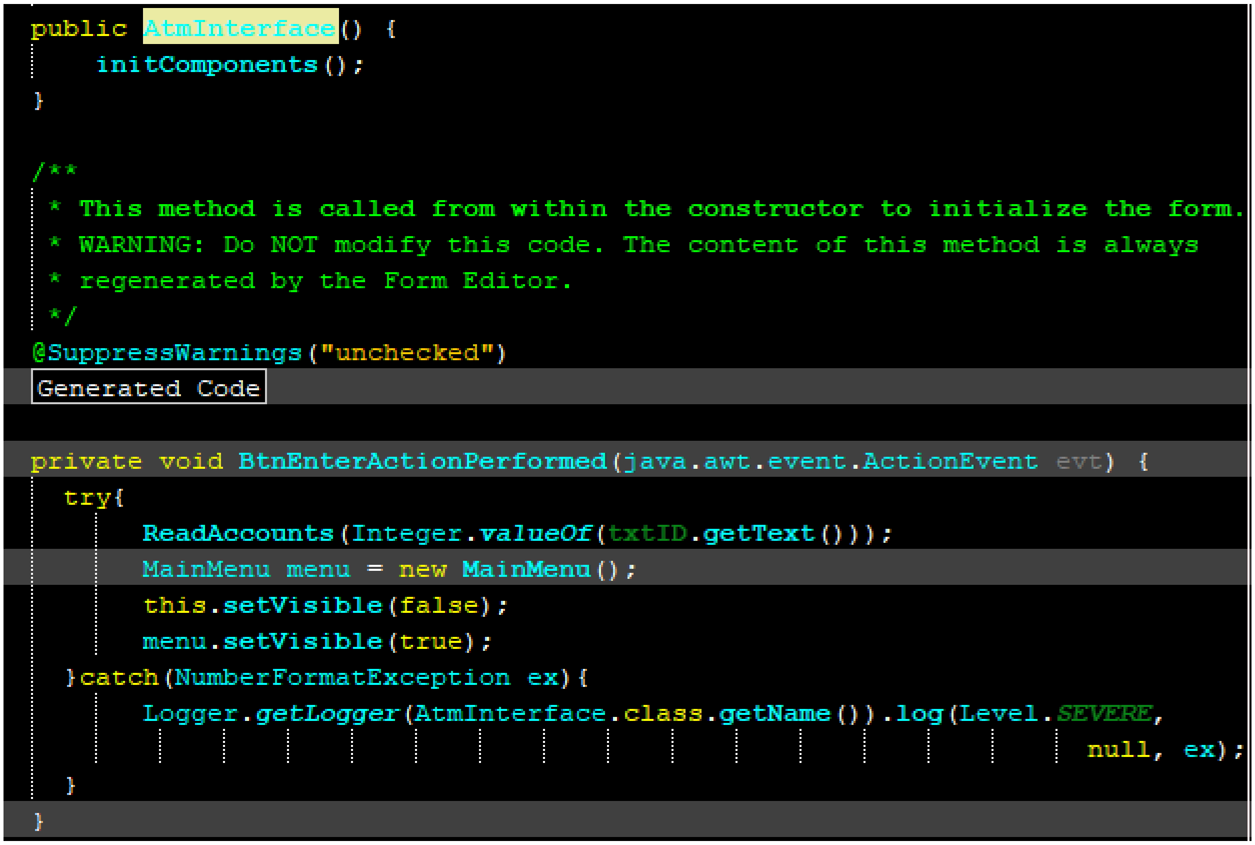
AtmInterface class:



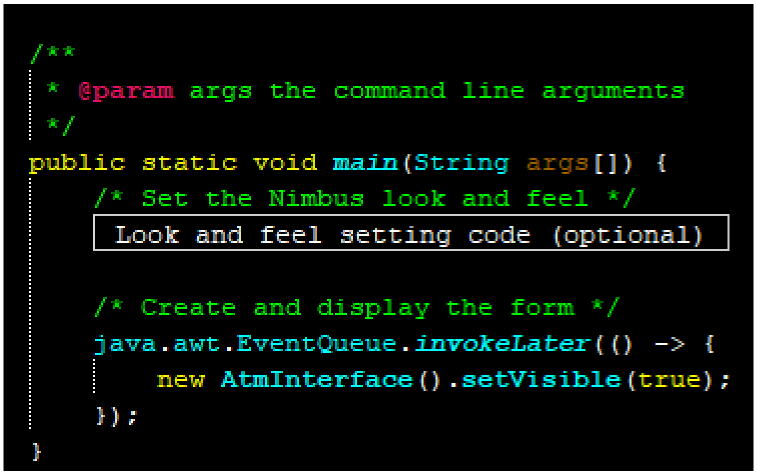
```
package atm;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
```



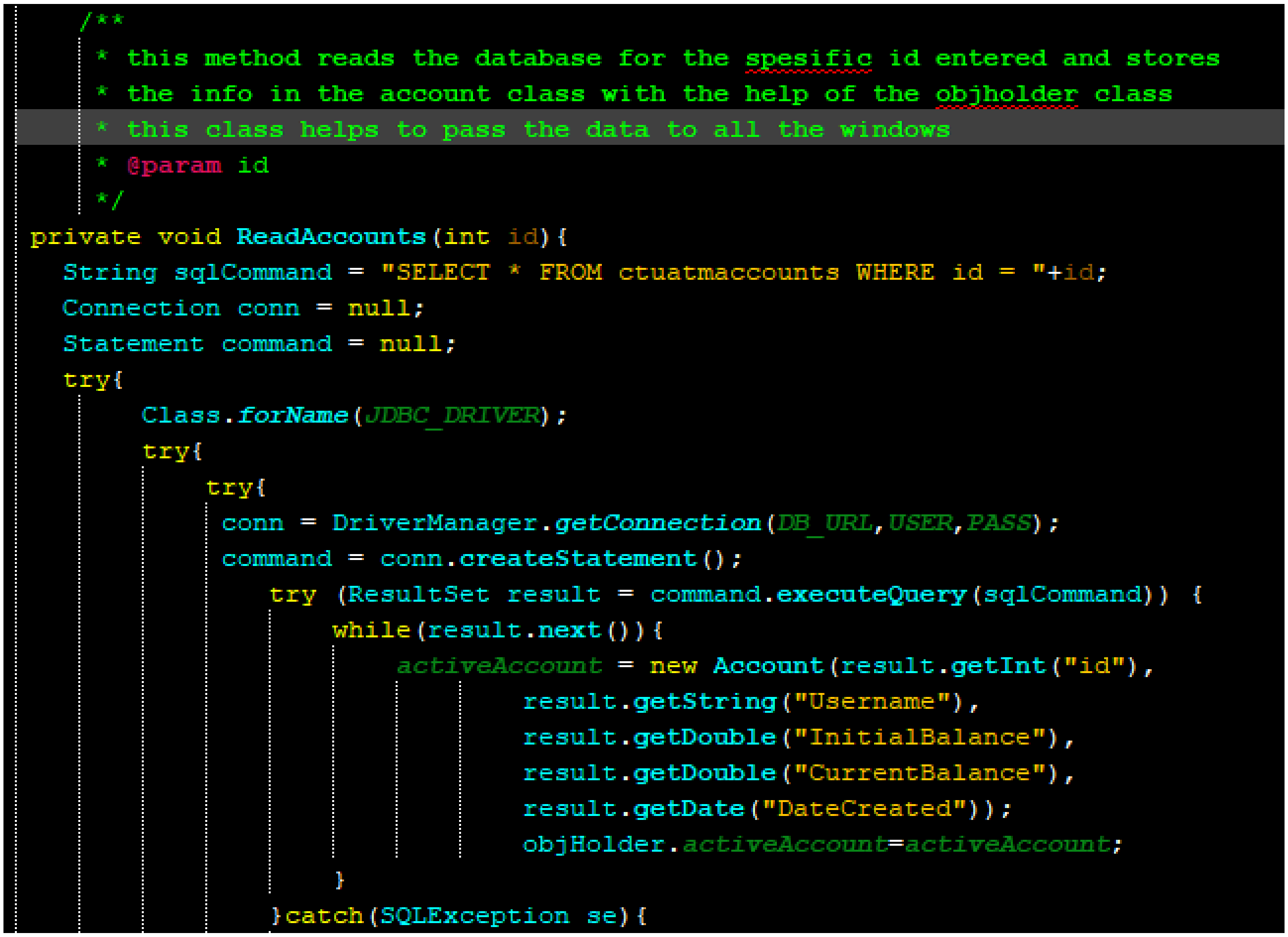
```
/**
 *
 * @author Albert Michael Ludick
 */
public class AtmInterface extends javax.swing.JFrame {
private static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
private static final String DB_URL ="jdbc:mysql://localhost:3306/ctuatm";
private static final String USER = "root";
private static final String PASS = "Ctu@2020";
private static Account activeAccount;
    /**
     * Creates new form AtmInterface
     */
    public AtmInterface() {
        initComponents();
    }
```



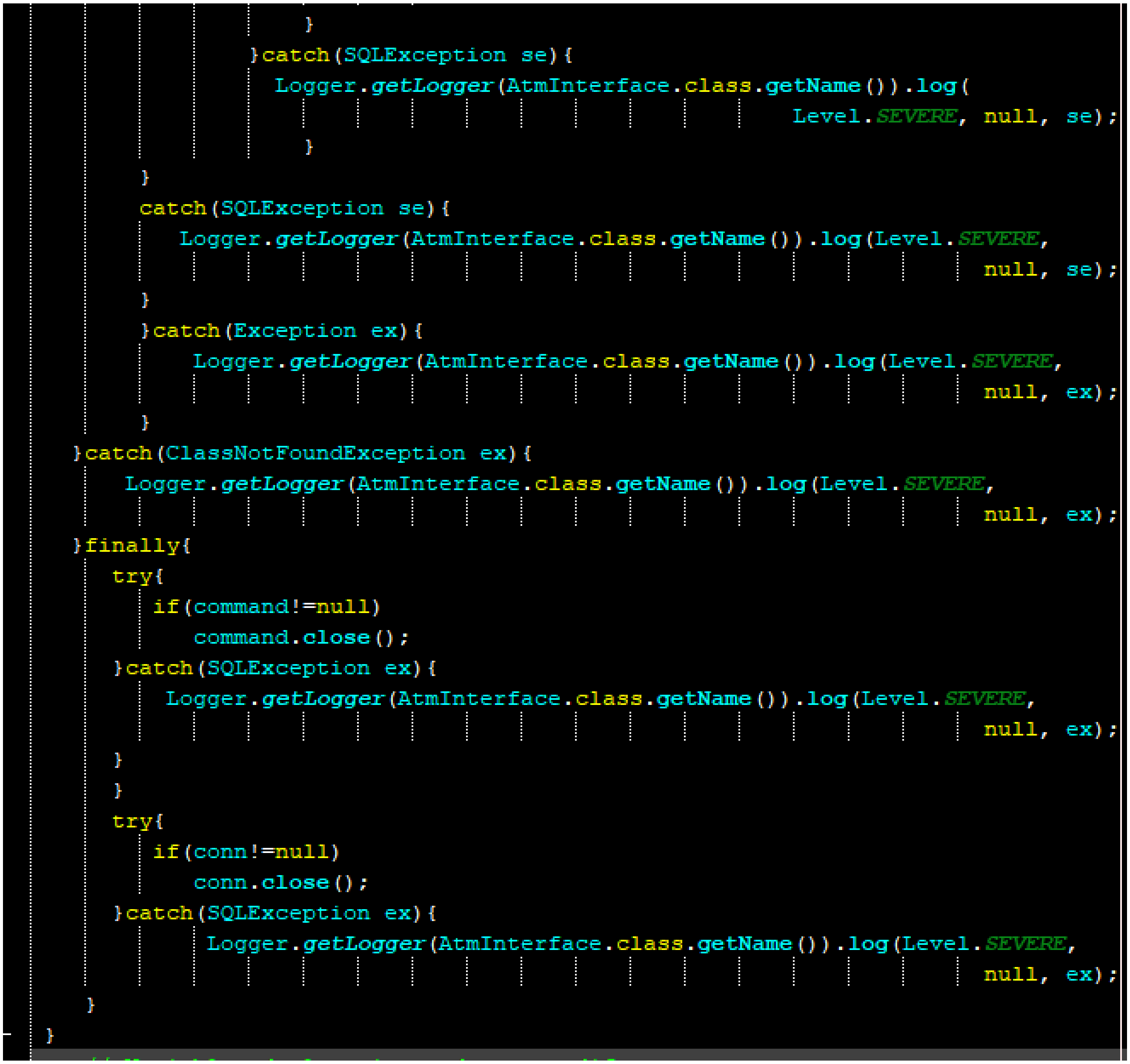
```
public AtmInterface() {
    initComponents();
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
Generated Code

private void BtnEnterActionPerformed(java.awt.event.ActionEvent evt) {
  try{
       ReadAccounts(Integer.valueOf(txtID.getText()));
       MainMenu menu = new MainMenu();
       this.setVisible(false);
       menu.setVisible(true);
  }catch(NumberFormatException ex){
       Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                null, ex);
  }
}
```



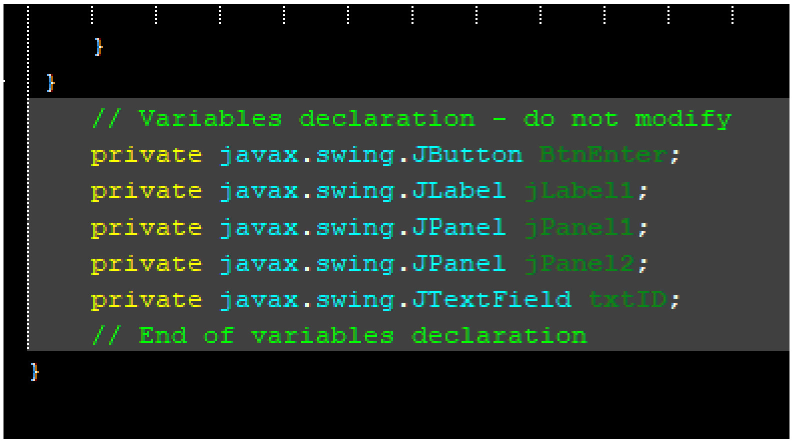
```
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
     Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(() -> {
        new AtmInterface().setVisible(true);
    });
}
```



```
   /**
    * this method reads the database for the spesific id entered and stores
    * the info in the account class with the help of the objholder class
    * this class helps to pass the data to all the windows
    * @param id
    */
private void ReadAccounts(int id){
  String sqlCommand = "SELECT * FROM ctuatmaccounts WHERE id = "+id;
  Connection conn = null;
  Statement command = null;
  try{
       Class.forName(JDBC_DRIVER);
       try{
           try{
            conn = DriverManager.getConnection(DB_URL,USER,PASS);
            command = conn.createStatement();
               try (ResultSet result = command.executeQuery(sqlCommand)) {
                   while(result.next()){
                       activeAccount = new Account(result.getInt("id"),
                               result.getString("Username"),
                               result.getDouble("InitialBalance"),
                               result.getDouble("CurrentBalance"),
                               result.getDate("DateCreated"));
                               objHolder.activeAccount=activeAccount;
                   }
               }catch(SQLException se){
```

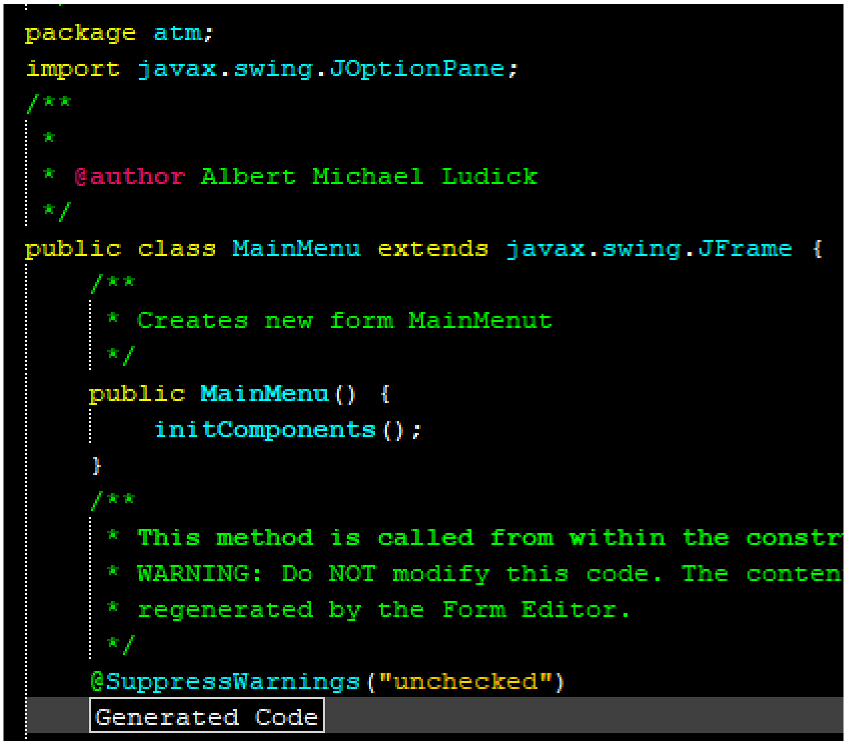


```
                    }
                }catch(SQLException se){
                  Logger.getLogger(AtmInterface.class.getName()).log(
                                                        Level.SEVERE, null, se);
                    }
        }
        catch(SQLException se){
           Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                      null, se);
        }
        }catch(Exception ex){
            Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                      null, ex);
        }
   }catch(ClassNotFoundException ex){
       Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                      null, ex);
   }finally{
      try{
         if(command!=null)
            command.close();
      }catch(SQLException ex){
          Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                      null, ex);
      }
      }
      try{
         if(conn!=null)
            conn.close();
      }catch(SQLException ex){
             Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                      null, ex);
    }
 }
```

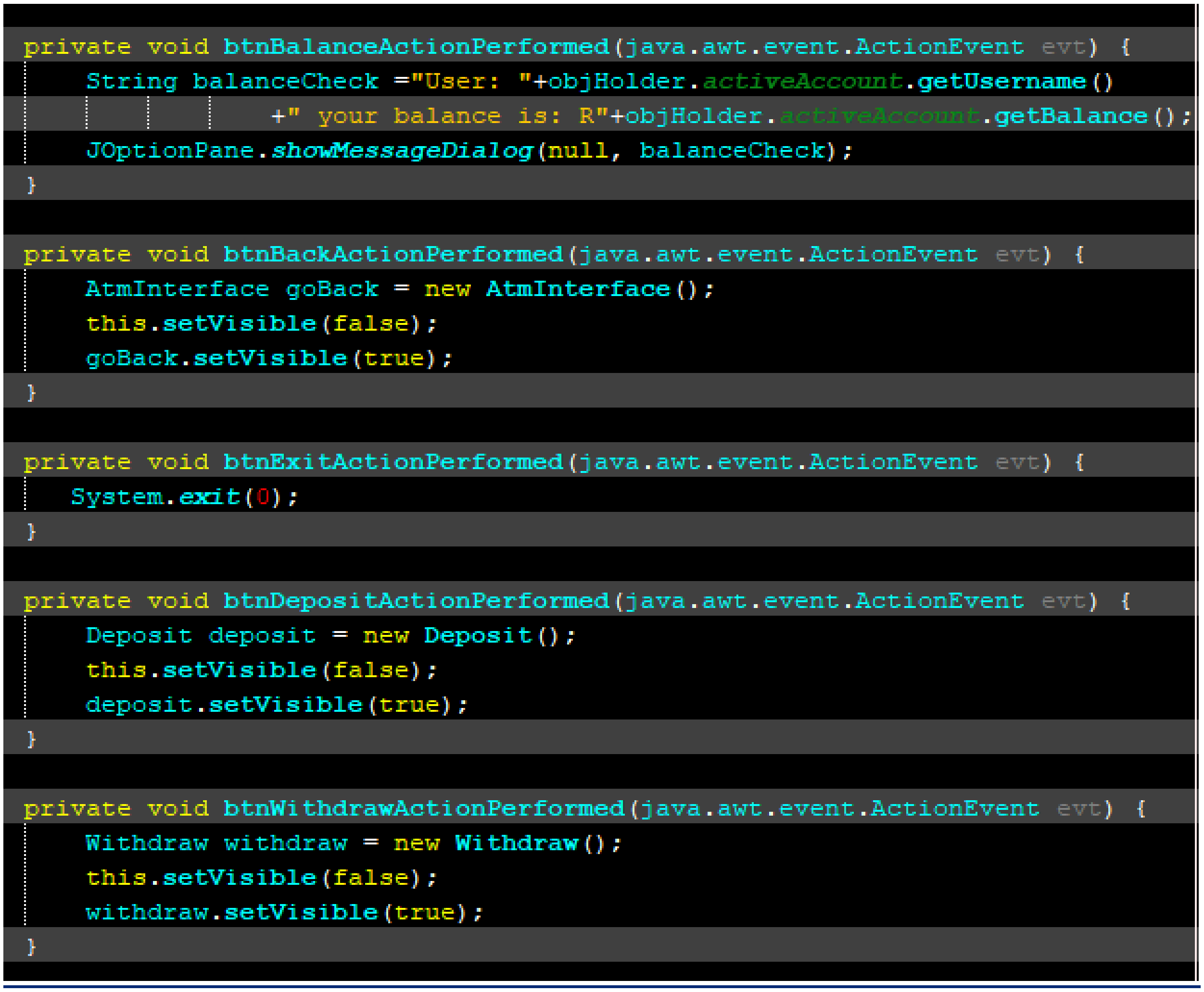


```
    }
 }
    // Variables declaration - do not modify
    private javax.swing.JButton BtnEnter;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JTextField txtID;
    // End of variables declaration
}
```

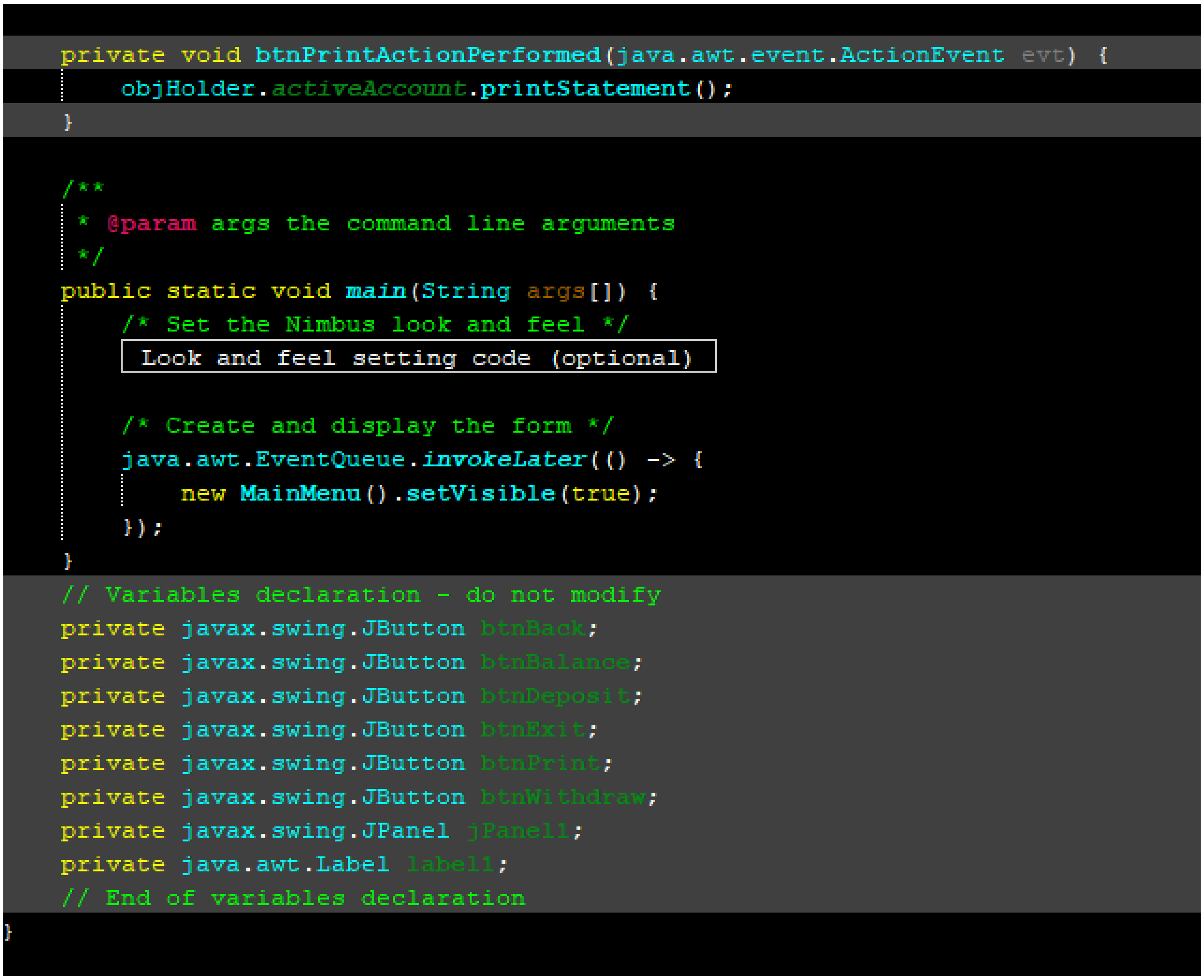
MainMenu Class:



```
package atm;
import javax.swing.JOptionPane;
/**
 *
 * @author Albert Michael Ludick
 */
public class MainMenu extends javax.swing.JFrame {
    /**
     * Creates new form MainMenut
     */
    public MainMenu() {
        initComponents();
    }
    /**
     * This method is called from within the constr
     * WARNING: Do NOT modify this code. The conten
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code
```

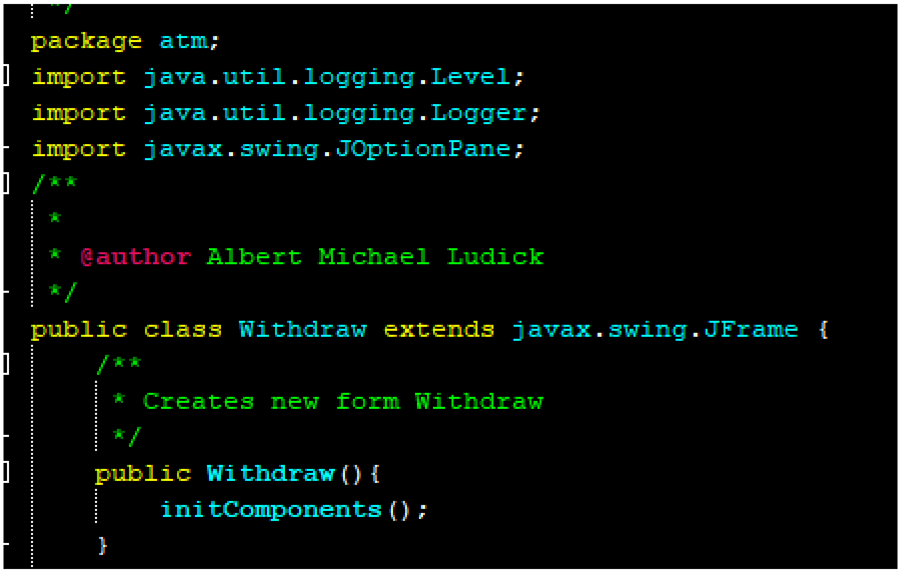


```java
private void btnBalanceActionPerformed(java.awt.event.ActionEvent evt) {
    String balanceCheck ="User: "+objHolder.activeAccount.getUsername()
                +" your balance is: R"+objHolder.activeAccount.getBalance();
    JOptionPane.showMessageDialog(null, balanceCheck);
}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    AtmInterface goBack = new AtmInterface();
    this.setVisible(false);
    goBack.setVisible(true);
}

private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
   System.exit(0);
}

private void btnDepositActionPerformed(java.awt.event.ActionEvent evt) {
    Deposit deposit = new Deposit();
    this.setVisible(false);
    deposit.setVisible(true);
}

private void btnWithdrawActionPerformed(java.awt.event.ActionEvent evt) {
    Withdraw withdraw = new Withdraw();
    this.setVisible(false);
    withdraw.setVisible(true);
}
```

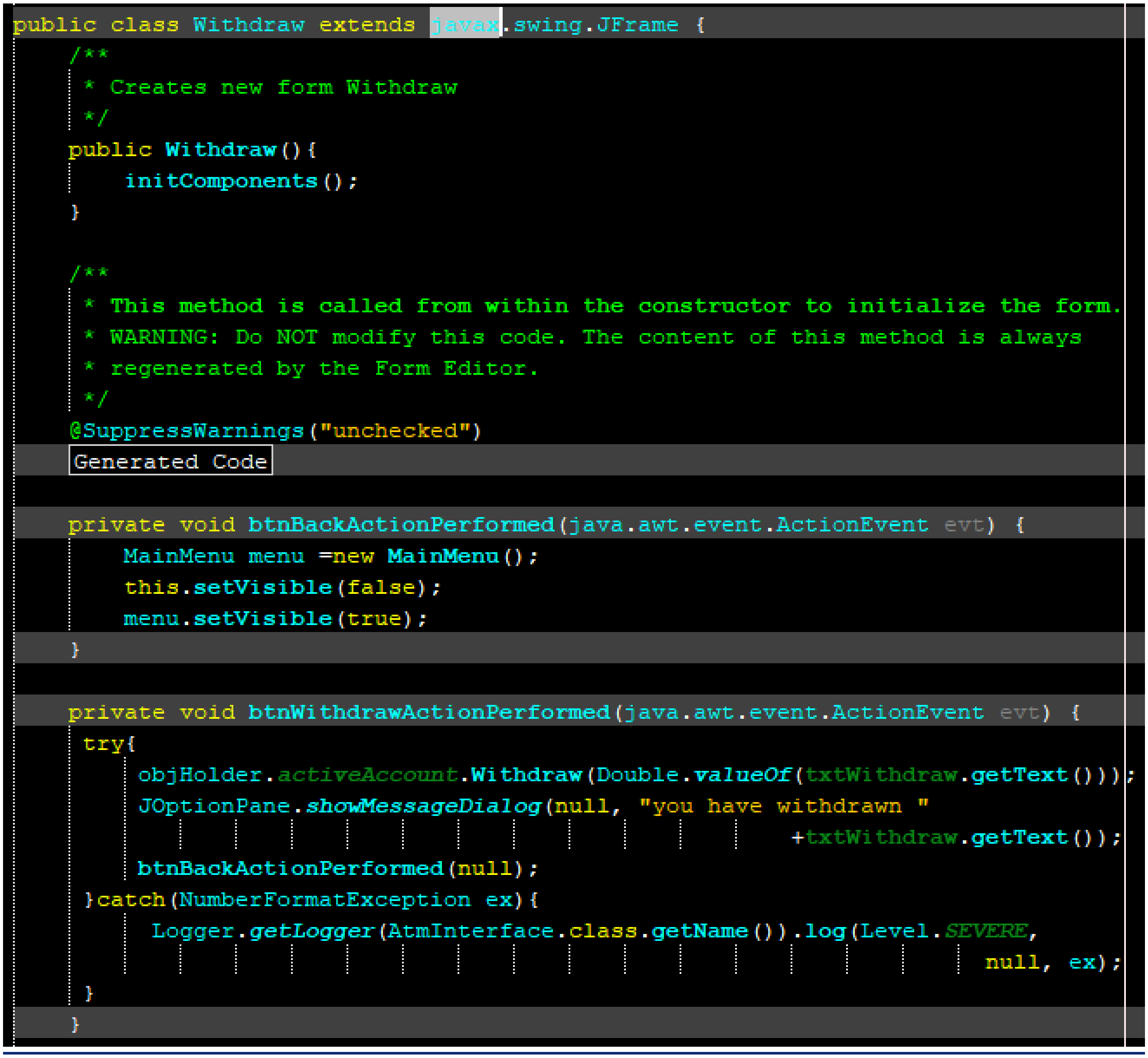


```
    private void btnPrintActionPerformed(java.awt.event.ActionEvent evt) {
        objHolder.activeAccount.printStatement();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> {
            new MainMenu().setVisible(true);
        });
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnBack;
    private javax.swing.JButton btnBalance;
    private javax.swing.JButton btnDeposit;
    private javax.swing.JButton btnExit;
    private javax.swing.JButton btnPrint;
    private javax.swing.JButton btnWithdraw;
    private javax.swing.JPanel jPanel1;
    private java.awt.Label label1;
    // End of variables declaration
}
```

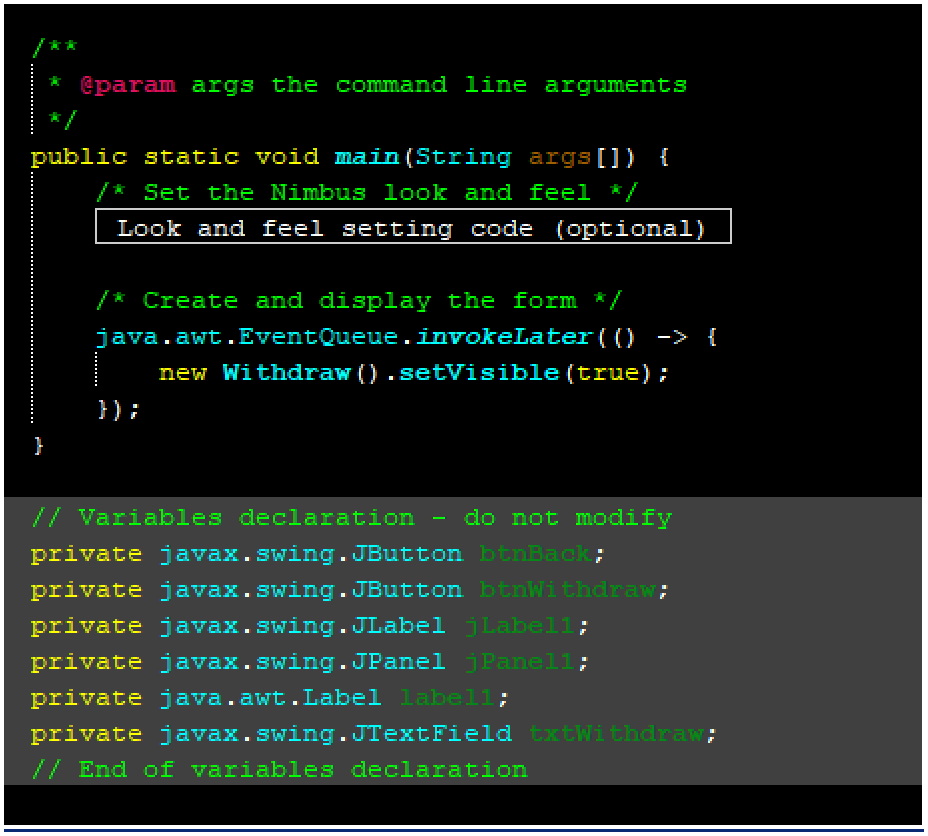
Withdraw class:



```
 */
package atm;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
/**
 *
 * @author Albert Michael Ludick
 */
public class Withdraw extends javax.swing.JFrame {
    /**
     * Creates new form Withdraw
     */
    public Withdraw(){
        initComponents();
    }
```

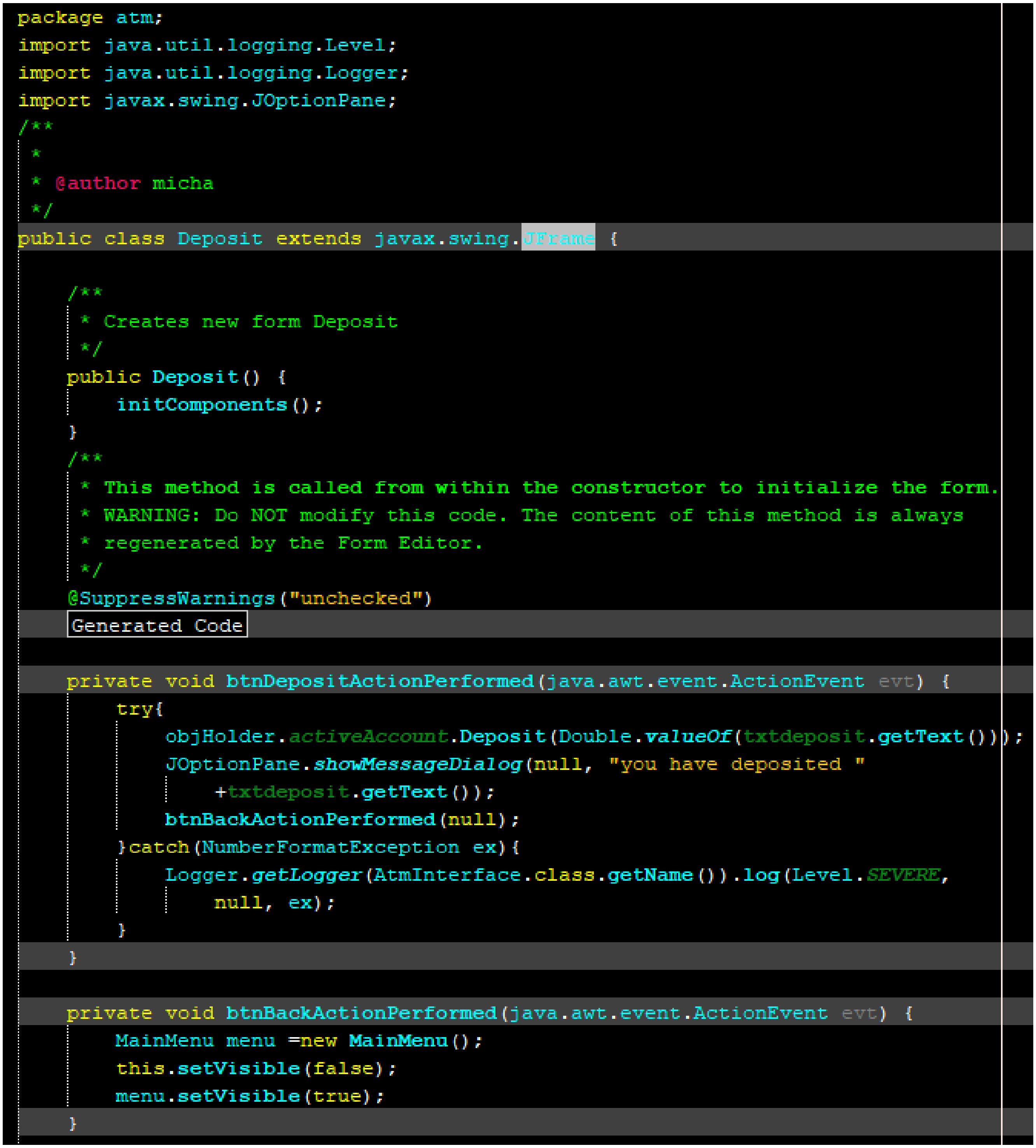


```
public class Withdraw extends javax.swing.JFrame {
    /**
     * Creates new form Withdraw
     */
    public Withdraw(){
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
        MainMenu menu =new MainMenu();
        this.setVisible(false);
        menu.setVisible(true);
    }

    private void btnWithdrawActionPerformed(java.awt.event.ActionEvent evt) {
     try{
         objHolder.activeAccount.Withdraw(Double.valueOf(txtWithdraw.getText()));
         JOptionPane.showMessageDialog(null, "you have withdrawn "
                                                        +txtWithdraw.getText());
         btnBackActionPerformed(null);
     }catch(NumberFormatException ex){
          Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                                      null, ex);
     }
    }
```

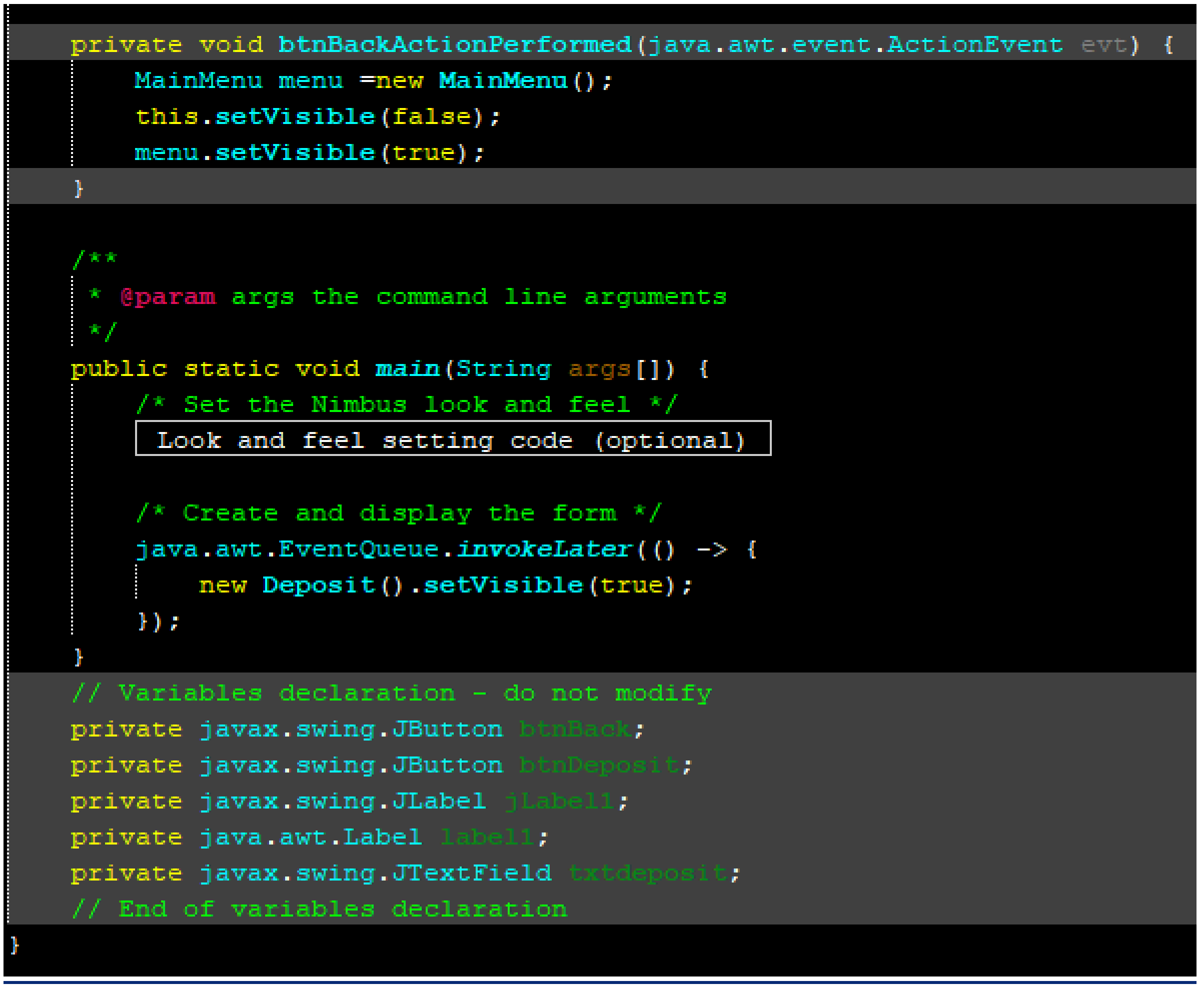


```
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(() -> {
        new Withdraw().setVisible(true);
    });
}

// Variables declaration - do not modify
private javax.swing.JButton btnBack;
private javax.swing.JButton btnWithdraw;
private javax.swing.JLabel jLabel1;
private javax.swing.JPanel jPanel1;
private java.awt.Label label1;
private javax.swing.JTextField txtWithdraw;
// End of variables declaration
```

Deposit class:

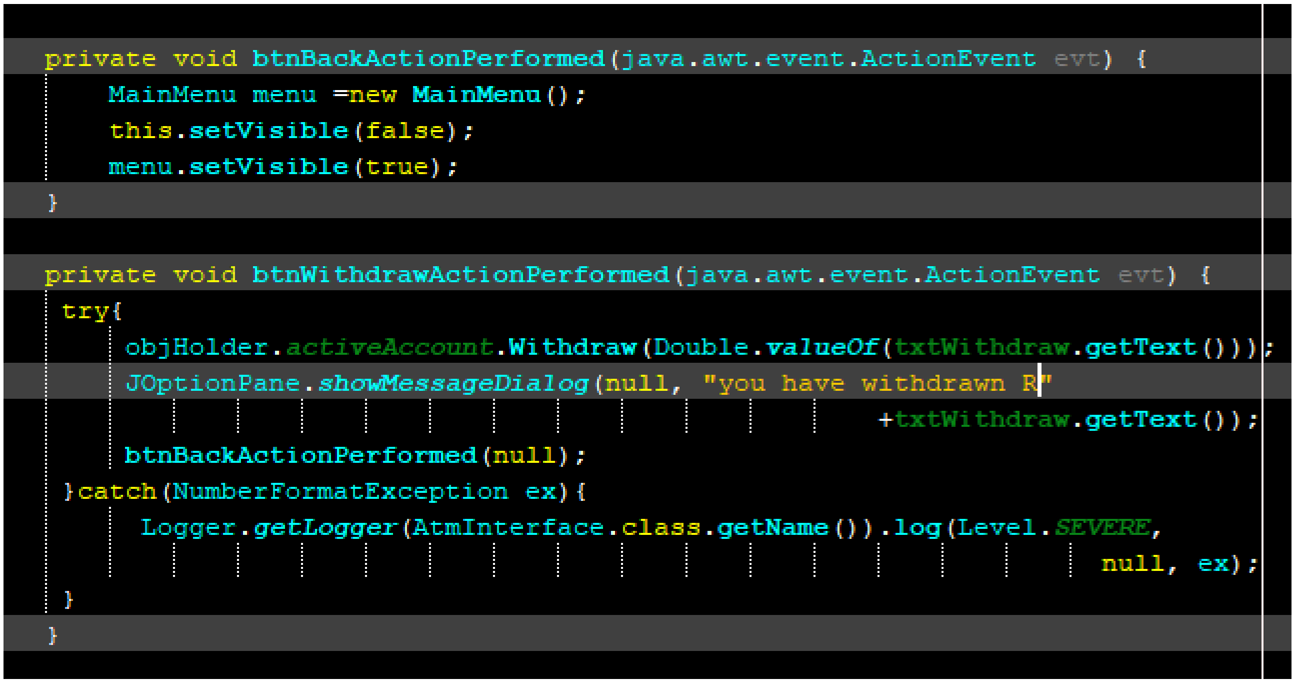


```
package atm;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
/**
 *
 * @author micha
 */
public class Deposit extends javax.swing.JFrame {

    /**
     * Creates new form Deposit
     */
    public Deposit() {
        initComponents();
    }
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnDepositActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            objHolder.activeAccount.Deposit(Double.valueOf(txtdeposit.getText()));
            JOptionPane.showMessageDialog(null, "you have deposited "
                +txtdeposit.getText());
            btnBackActionPerformed(null);
        }catch(NumberFormatException ex){
            Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                null, ex);
        }
    }

    private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
        MainMenu menu =new MainMenu();
        this.setVisible(false);
        menu.setVisible(true);
    }
```

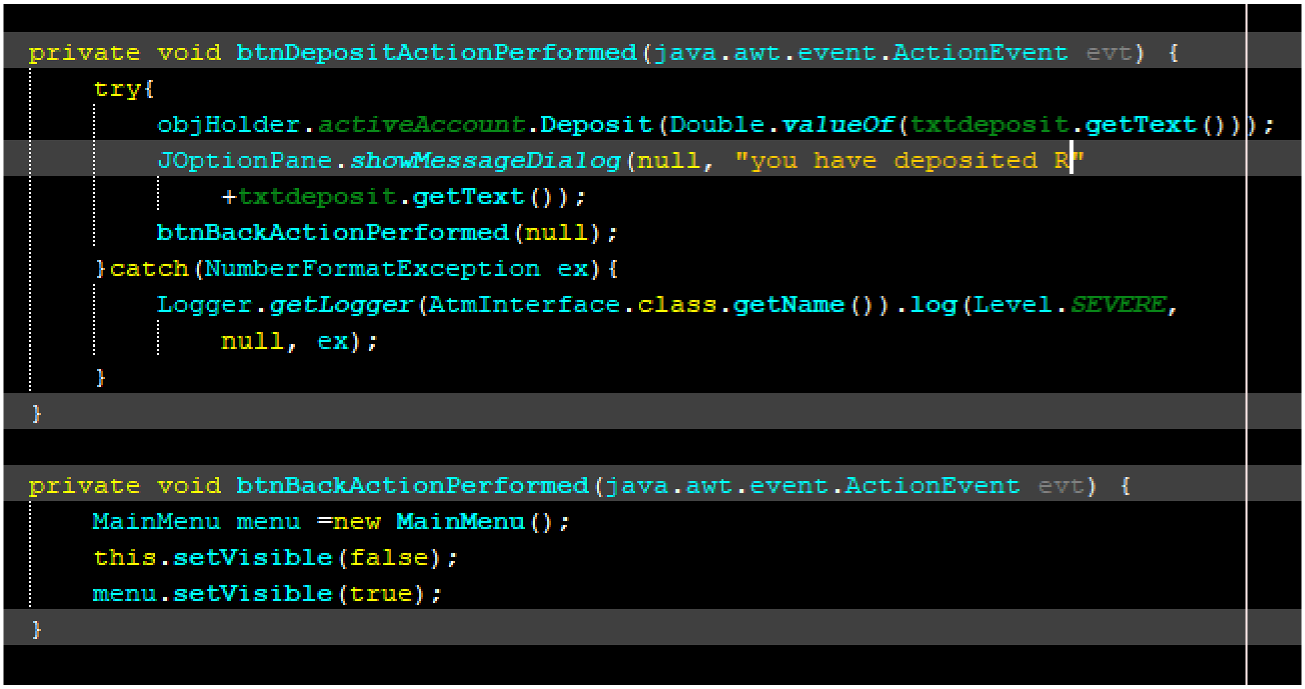


```
    private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
        MainMenu menu =new MainMenu();
        this.setVisible(false);
        menu.setVisible(true);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> {
            new Deposit().setVisible(true);
        });
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnBack;
    private javax.swing.JButton btnDeposit;
    private javax.swing.JLabel jLabel1;
    private java.awt.Label label1;
    private javax.swing.JTextField txtdeposit;
    // End of variables declaration
}
```

I changed these two class found a mistake:

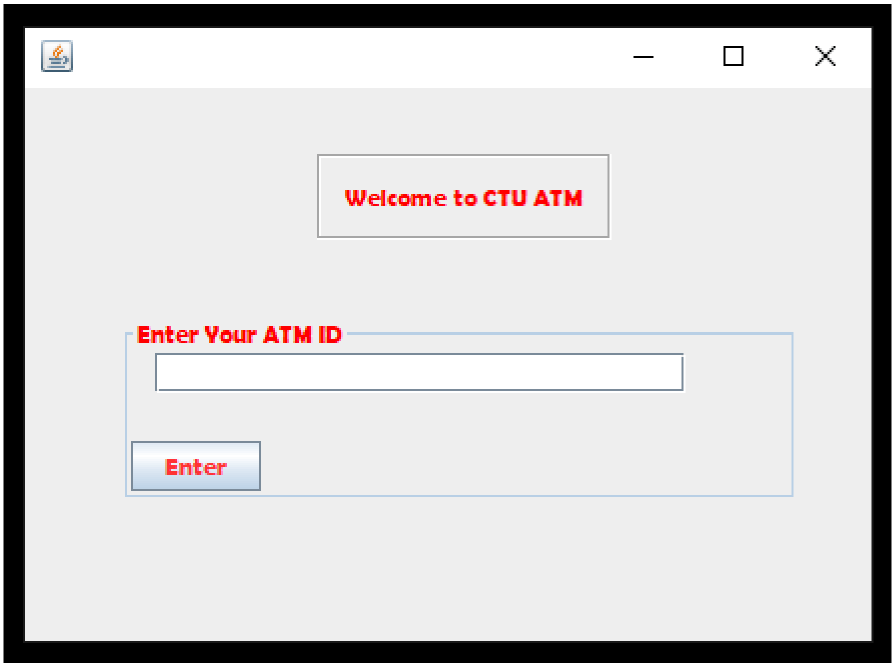


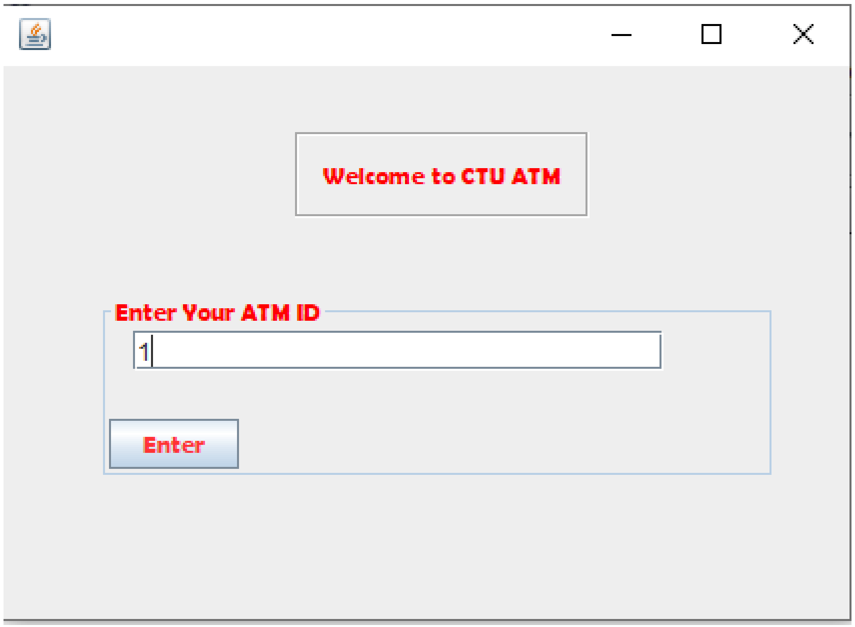
```
private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    MainMenu menu =new MainMenu();
    this.setVisible(false);
    menu.setVisible(true);
}

private void btnWithdrawActionPerformed(java.awt.event.ActionEvent evt) {
 try{
     objHolder.activeAccount.Withdraw(Double.valueOf(txtWithdraw.getText()));
     JOptionPane.showMessageDialog(null, "you have withdrawn R"
                                                      +txtWithdraw.getText());
     btnBackActionPerformed(null);
 }catch(NumberFormatException ex){
      Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
                                                              null, ex);
 }
}
```



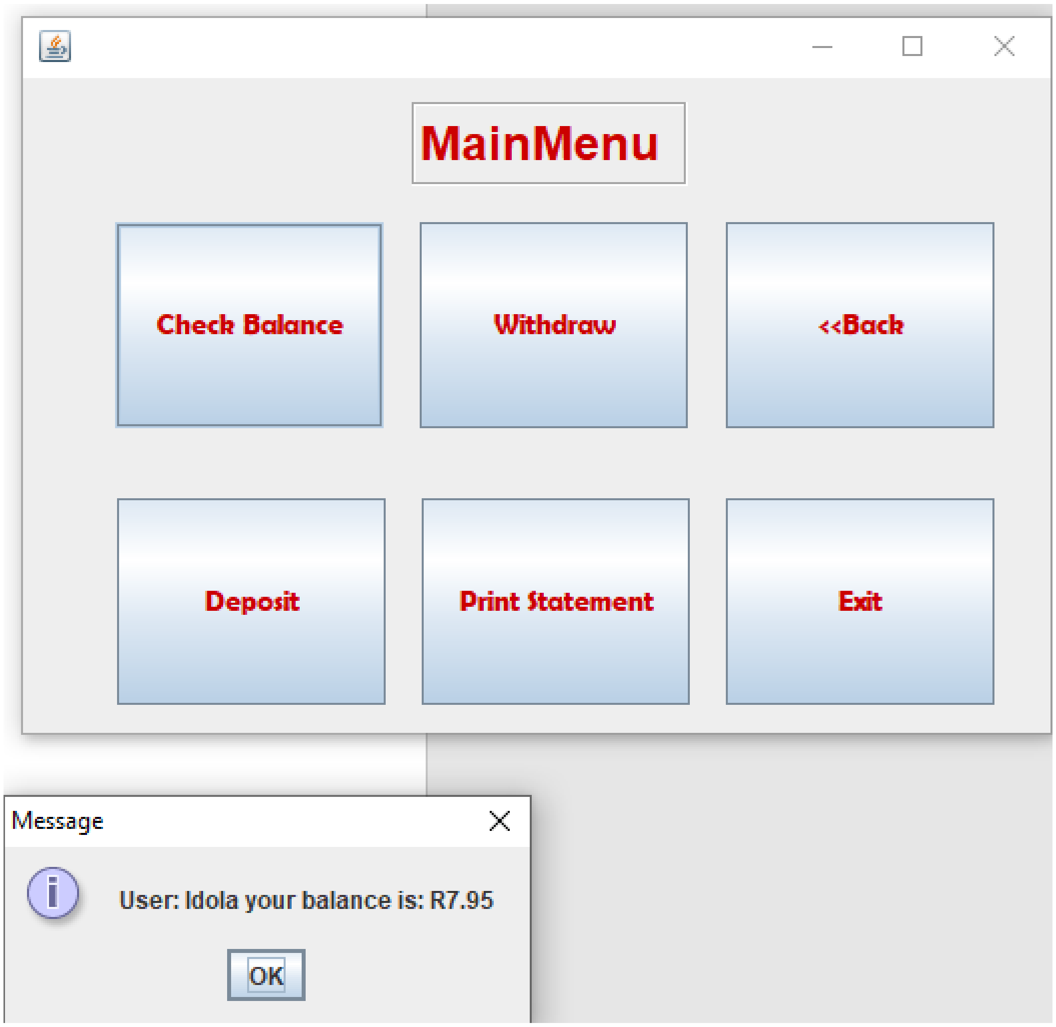
```
private void btnDepositActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        objHolder.activeAccount.Deposit(Double.valueOf(txtdeposit.getText()));
        JOptionPane.showMessageDialog(null, "you have deposited R"
            +txtdeposit.getText());
        btnBackActionPerformed(null);
    }catch(NumberFormatException ex){
        Logger.getLogger(AtmInterface.class.getName()).log(Level.SEVERE,
            null, ex);
    }
}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    MainMenu menu =new MainMenu();
    this.setVisible(false);
    menu.setVisible(true);
}
```

Running App:

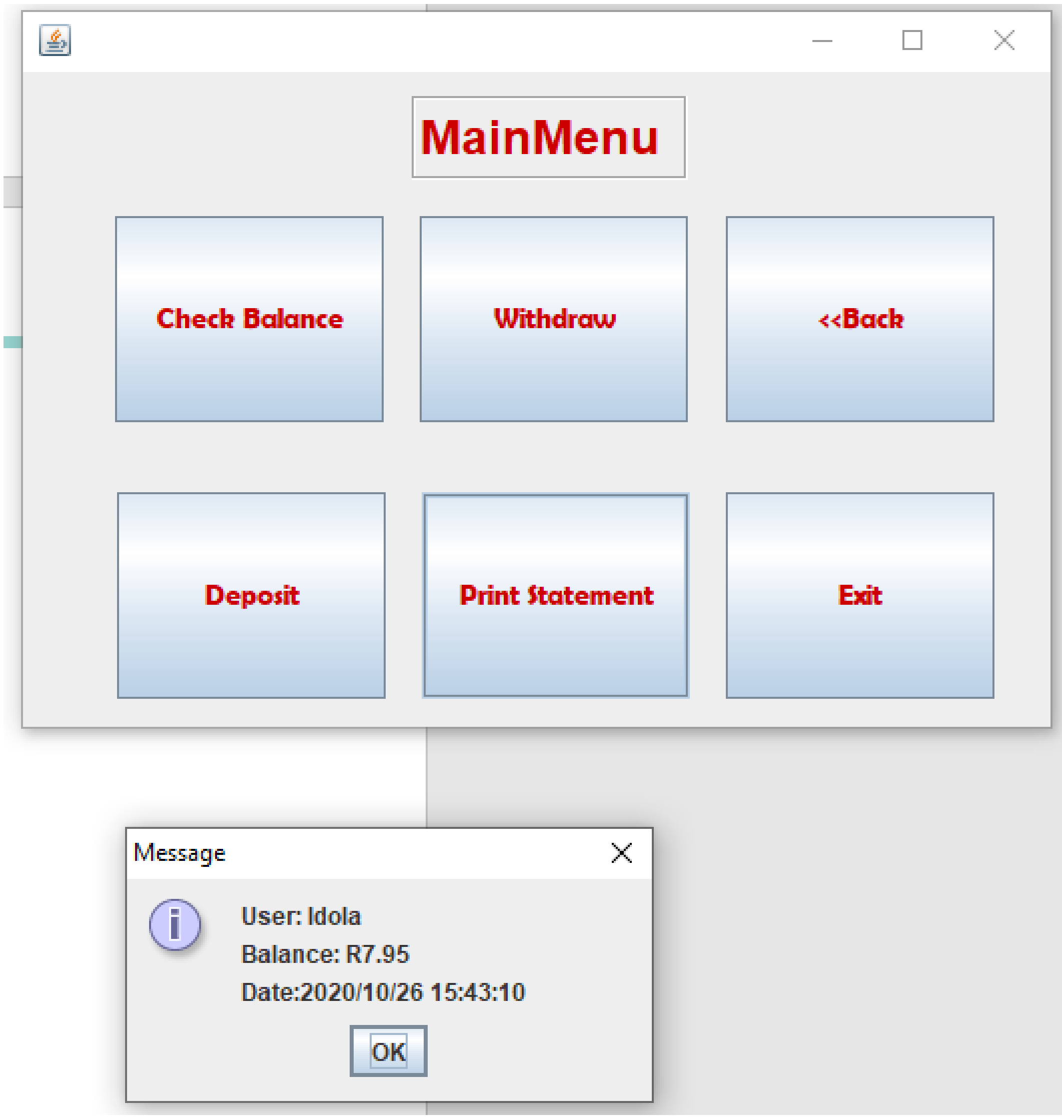




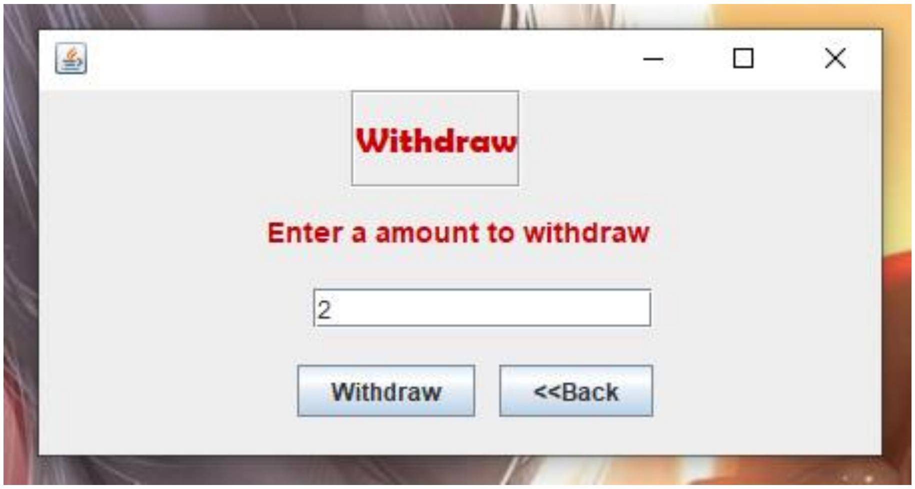
Enter Your ATM ID: 1



User: Idola your balance is: R7.95

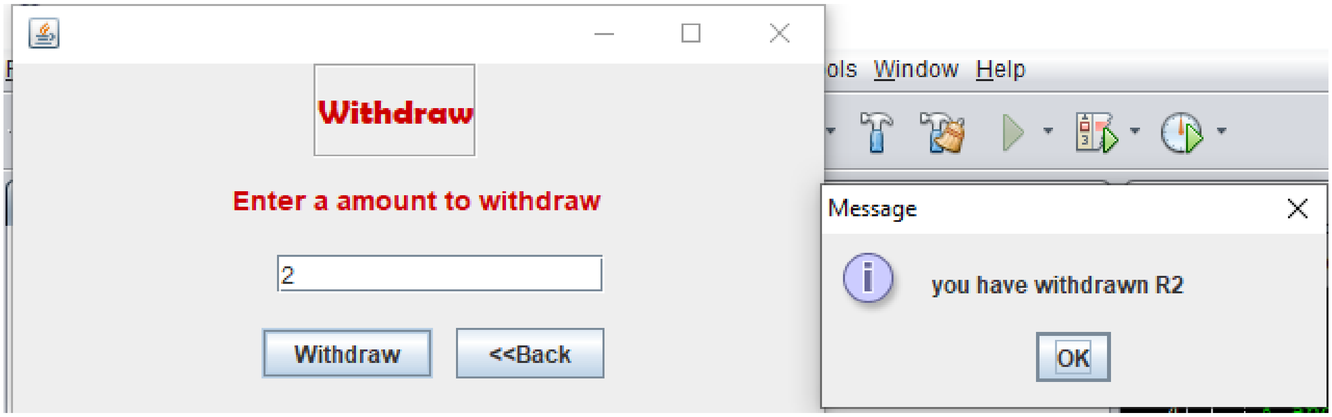


User: Idola
Balance: R7.95
Date:2020/10/26 15:43:10

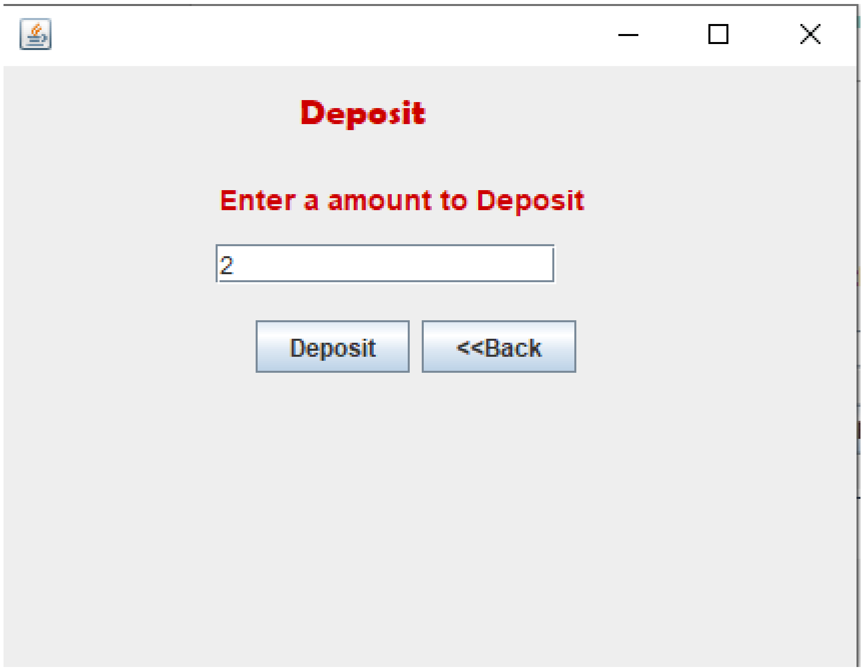


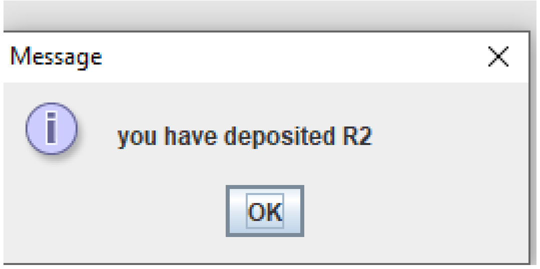
Enter a amount to withdraw

2



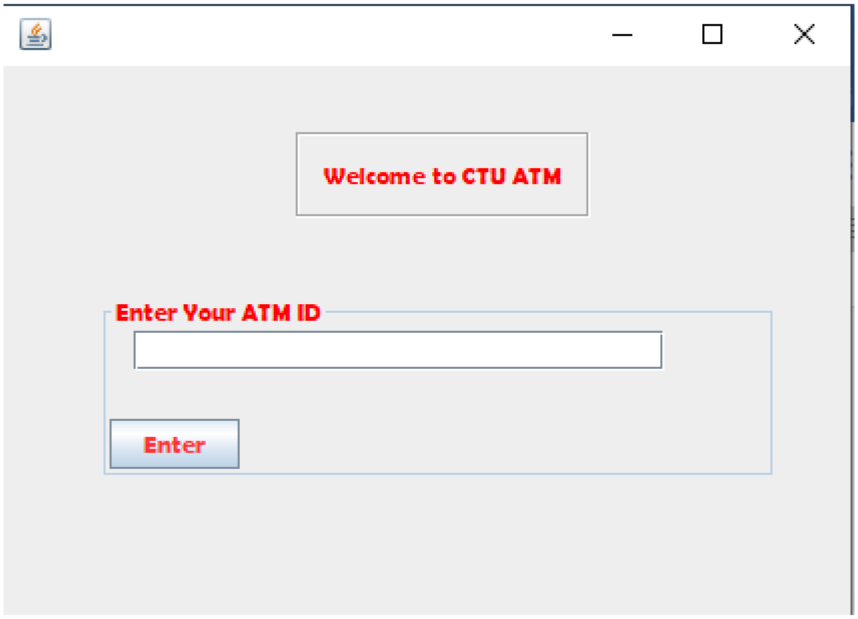
you have withdrawn R2





you have deposited R2

When pressing back

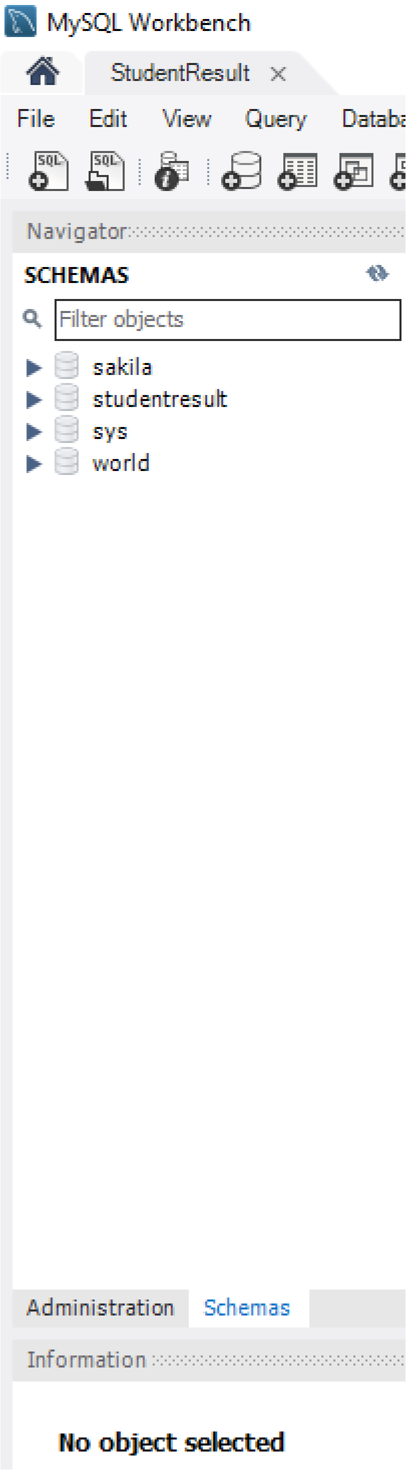


Welcome to CTU ATM

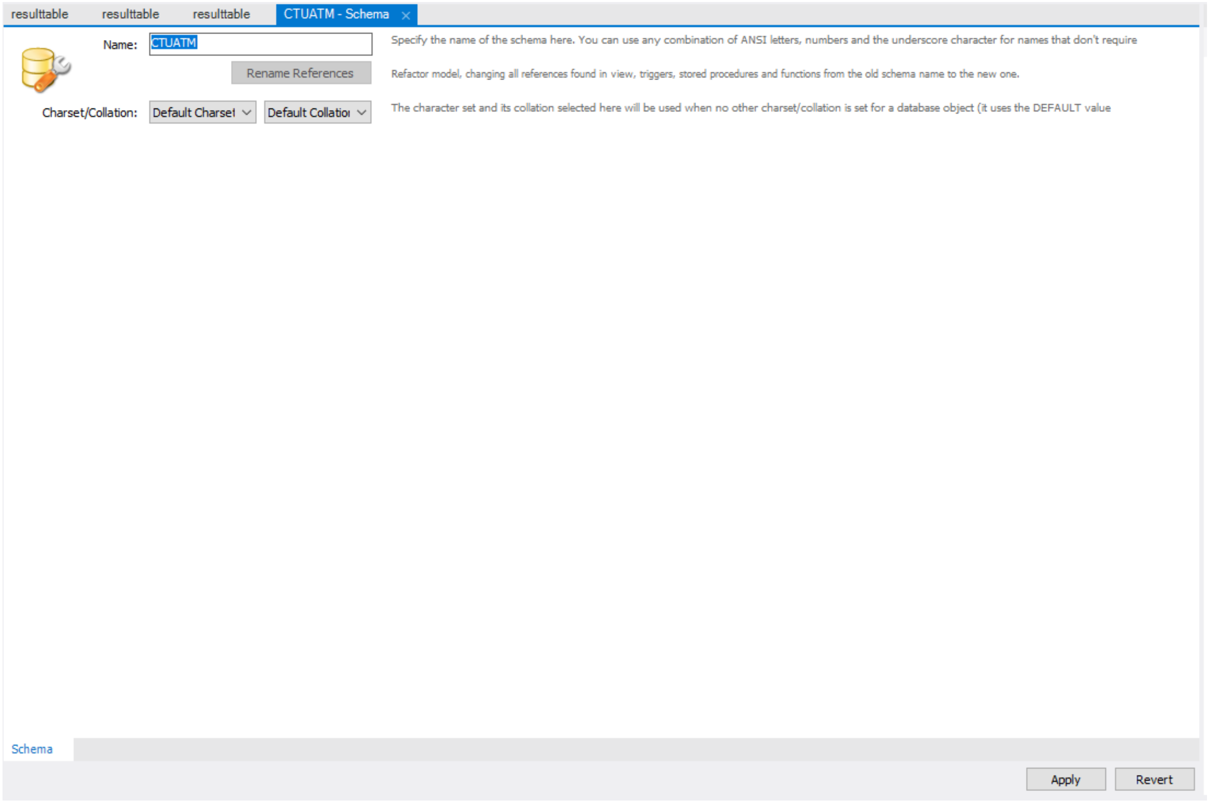
Enter Your ATM ID

When pressing exit the app exits.

The sql stuff



No object selected



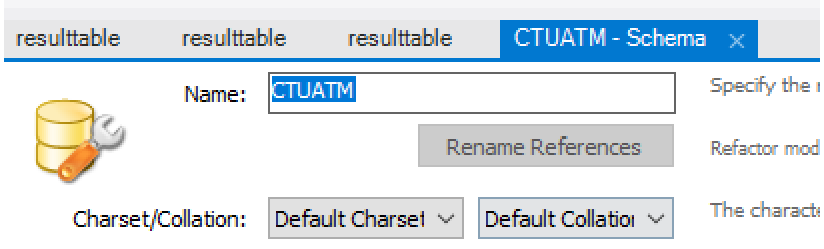
Name: CTUATM

Specify the name of the schema here. You can use any combination of ANSI letters, numbers and the underscore character for names that don't require

Refactor model, changing all references found in view, triggers, stored procedures and functions from the old schema name to the new one.

Charset/Collation: Default Charset, Default Collation

The character set and its collation selected here will be used when no other charset/collation is set for a database object (it uses the DEFAULT value



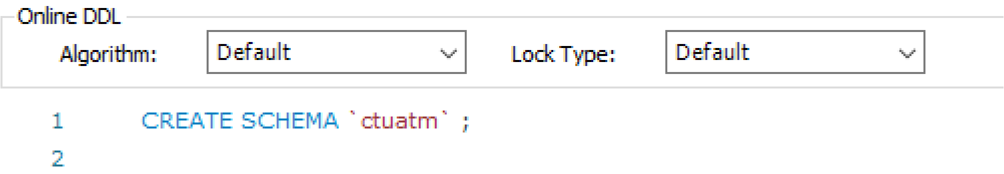
Name: CTUATM

Specify the ı

Refactor mod

Charset/Collation: Default Charset, Default Collation

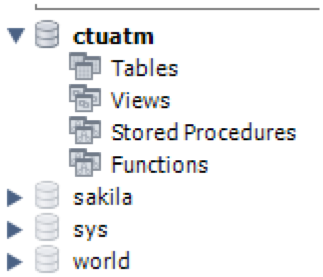
The characte



Algorithm: Default

Lock Type: Default

```
CREATE SCHEMA `ctuatm` ;
```



ctuatm
- Tables
- Views
- Stored Procedures
- Functions

sakila

sys

world

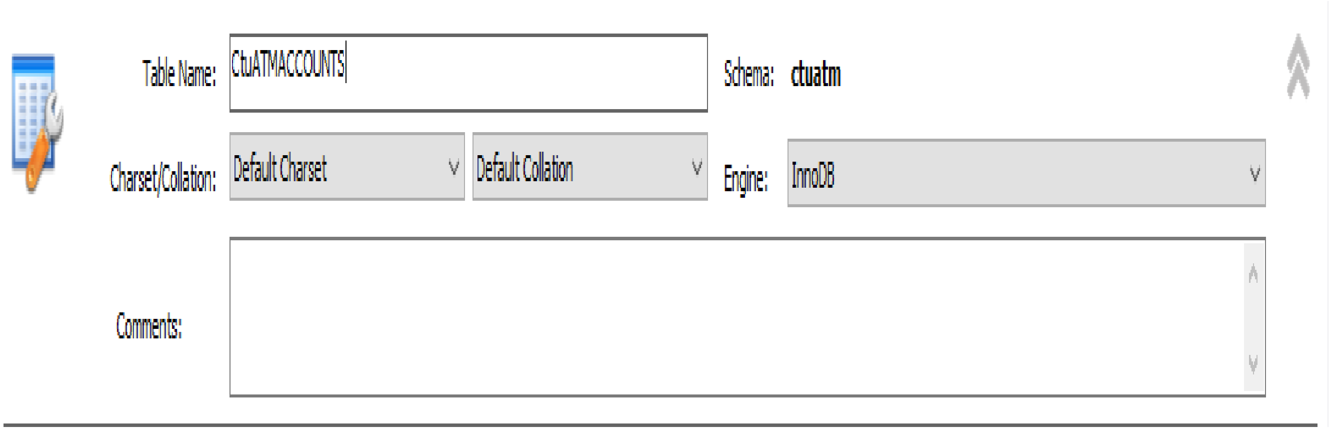


Table Name: CtuATMACCOUNTS

Schema: ctuatm

Charset/Collation: Default Charset | Default Collation

Engine: InnoDB

Comments:

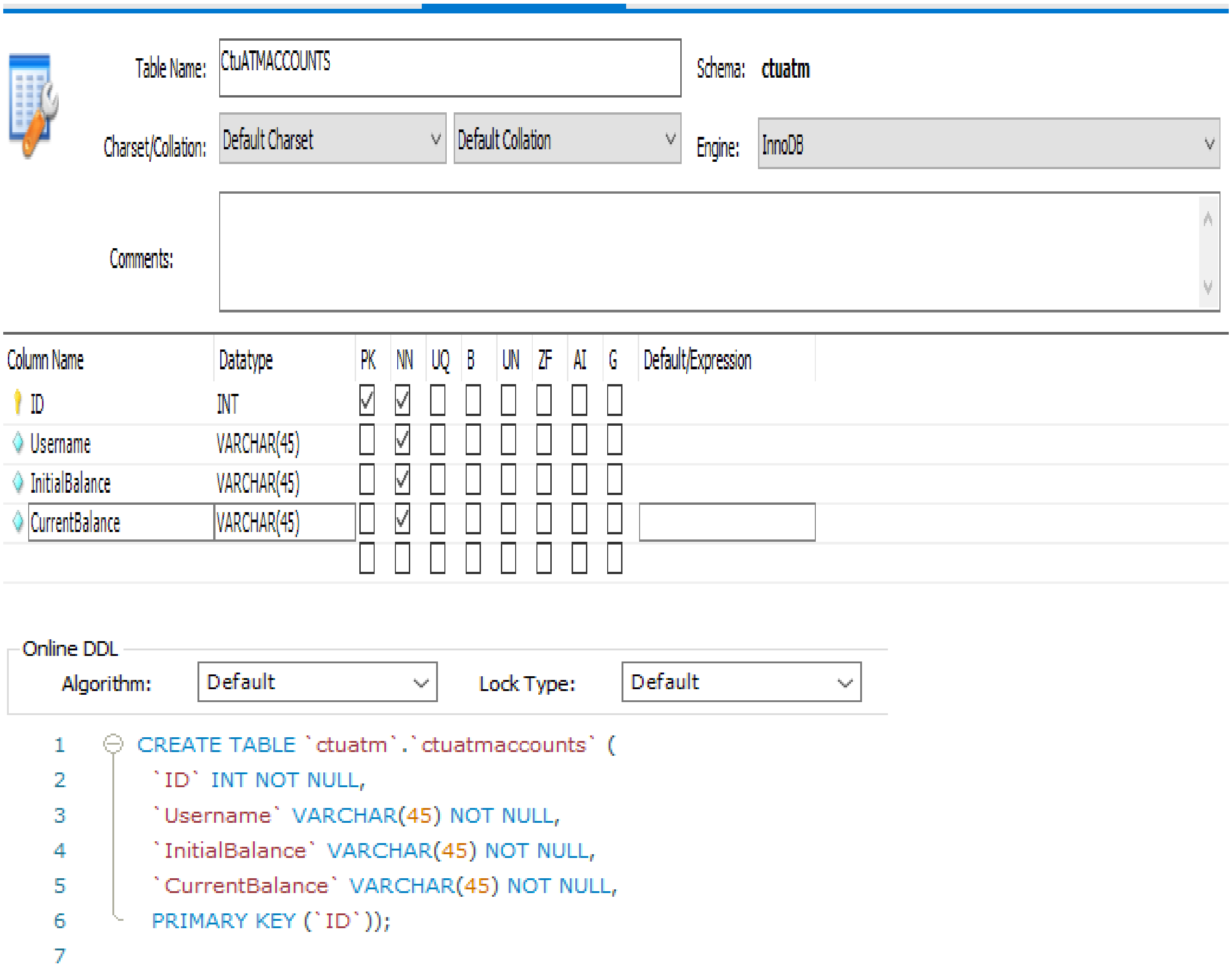


Table Name: CtuATMACCOUNTS

Schema: ctuatm

Charset/Collation: Default Charset | Default Collation

Engine: InnoDB

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
ID	INT	☑	☑	□	□	□	□	□	□	
Username	VARCHAR(45)	□	☑	□	□	□	□	□	□	
InitialBalance	VARCHAR(45)	□	☑	□	□	□	□	□	□	
CurrentBalance	VARCHAR(45)	□	☑	□	□	□	□	□	□	
		□	□	□	□	□	□	□	□	

Online DDL

Algorithm: Default

Lock Type: Default

```
CREATE TABLE `ctuatm`.`ctuatmaccounts` (
  `ID` INT NOT NULL,
  `Username` VARCHAR(45) NOT NULL,
  `InitialBalance` VARCHAR(45) NOT NULL,
  `CurrentBalance` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`ID`));
```

Conclusion:

My coinclusion is there is no coinclusion to this coinclusion.

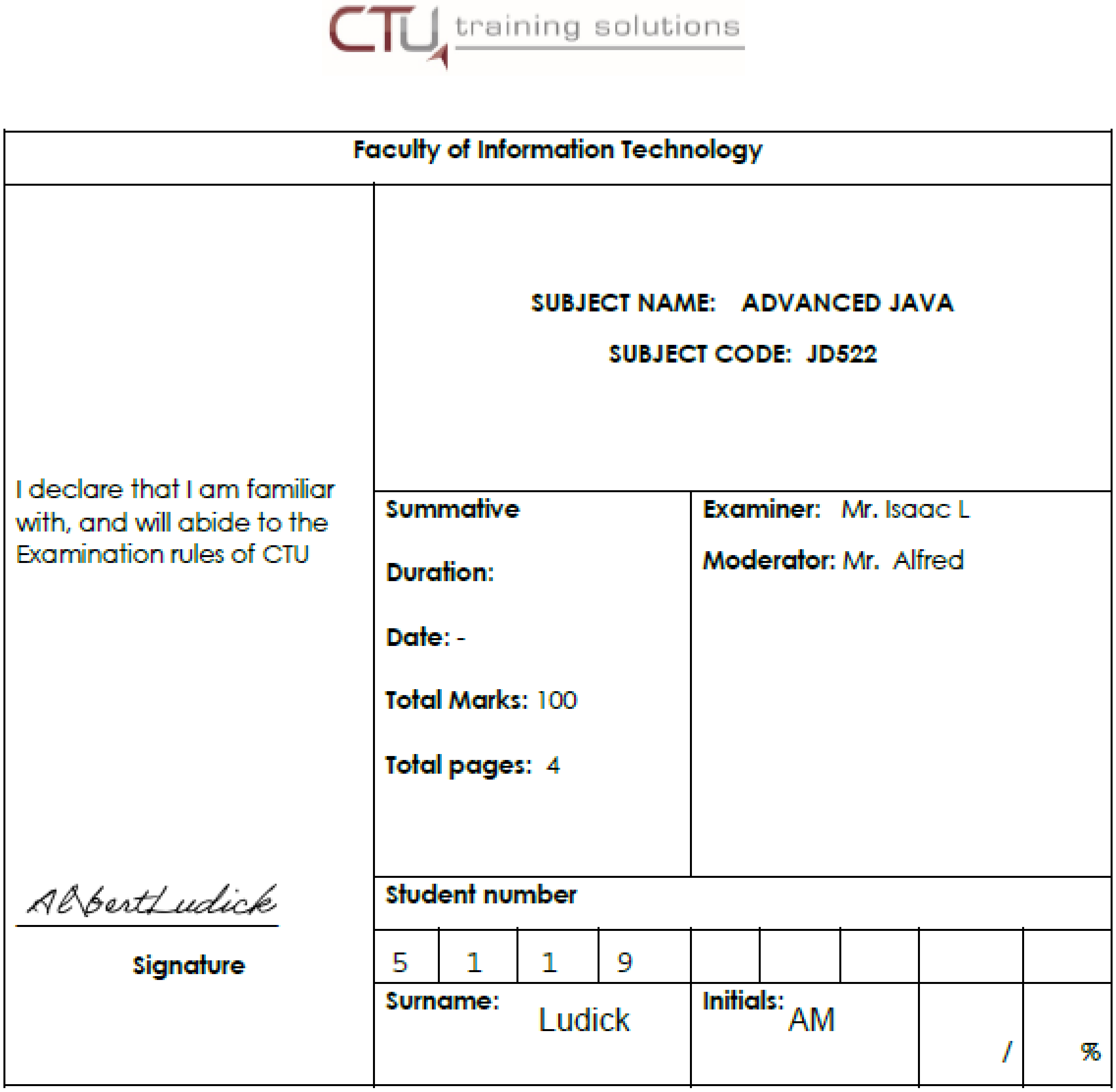
References:

Website:

https://stackoverflow.com/questions/5455794/removing-whitespace-from-strings-in-java

https://www.generatedata.com/

Appendix:



CTU training solutions

Faculty of Information Technology

SUBJECT NAME: ADVANCED JAVA

SUBJECT CODE: JD522

I declare that I am familiar with, and will abide to the Examination rules of CTU

AlbertLudick

Signature

Summative

Duration:

Date: -

Total Marks: 100

Total pages: 4

Examiner: Mr. Isaac L

Moderator: Mr. Alfred

Student number

5	1	1	9					

Surname: Ludick

Initials: AM

/ %