LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS4001NI Programming

Assessment Weightage & Type

30% Individual Coursework 2

Year and Semester

2021-22 Autumn

Student Name: Rohit Ratna Shakya

London Met ID: 21049578

College ID: NP01CP4A210237

Assignment Due Date: 5th August 2022

Assignment Submission Date: 5th August 2022

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

Table of Contents

List of Figures

List of Tables

1. Introduction

The following coursework is extension of the coursework we completed previously where our main aim was to add a class to create a Graphical User Interface (GUI) for the system that used an array list to store all the information of our rented instrument and purchased instrument. The coursework's primary goal was to make our company's program for renting and selling musical instruments easier to use. So, a new class called “Sarangi Sansar” was created and was linked to our previous classes “Instrument”, “Instrument to Rent” and “Instrument to sell” using object casting. The following assignment was done in Blue J Text Editor, Draw.io, and MS-Word. The description of the tools used is given below:

- **Blue J Text Editor**
  Blue J is a Java development environment that makes it simple and quick to create Java apps. We can interact with objects using Blue J by inspecting their value, calling methods on them, passing them as parameters, and more. (Anon., n.d.)

- **Draw.io**
  Draw.io is a customized program that we can use to generate custom layouts or select from an automatic layout function when creating diagrams and charts. They offer a wide variety of shapes and several visual components to help you create a unique diagram or chart. (Anon., n.d.)

- **MS-Word**
  Microsoft Word is a word processor published by Microsoft which allows us to create documents, reports, and letters, and provides features such as spell checking, grammar checking, text formatting, font formatting, etc. (Anon., n.d.)

2. Class Diagram

A class Diagram is a static structure diagram that helps us to describe the structure of the system by showing the classes, attributes methods, and relations within the system's class. (Anon., n.d.)

2.1 Class Diagram of Instrument Class

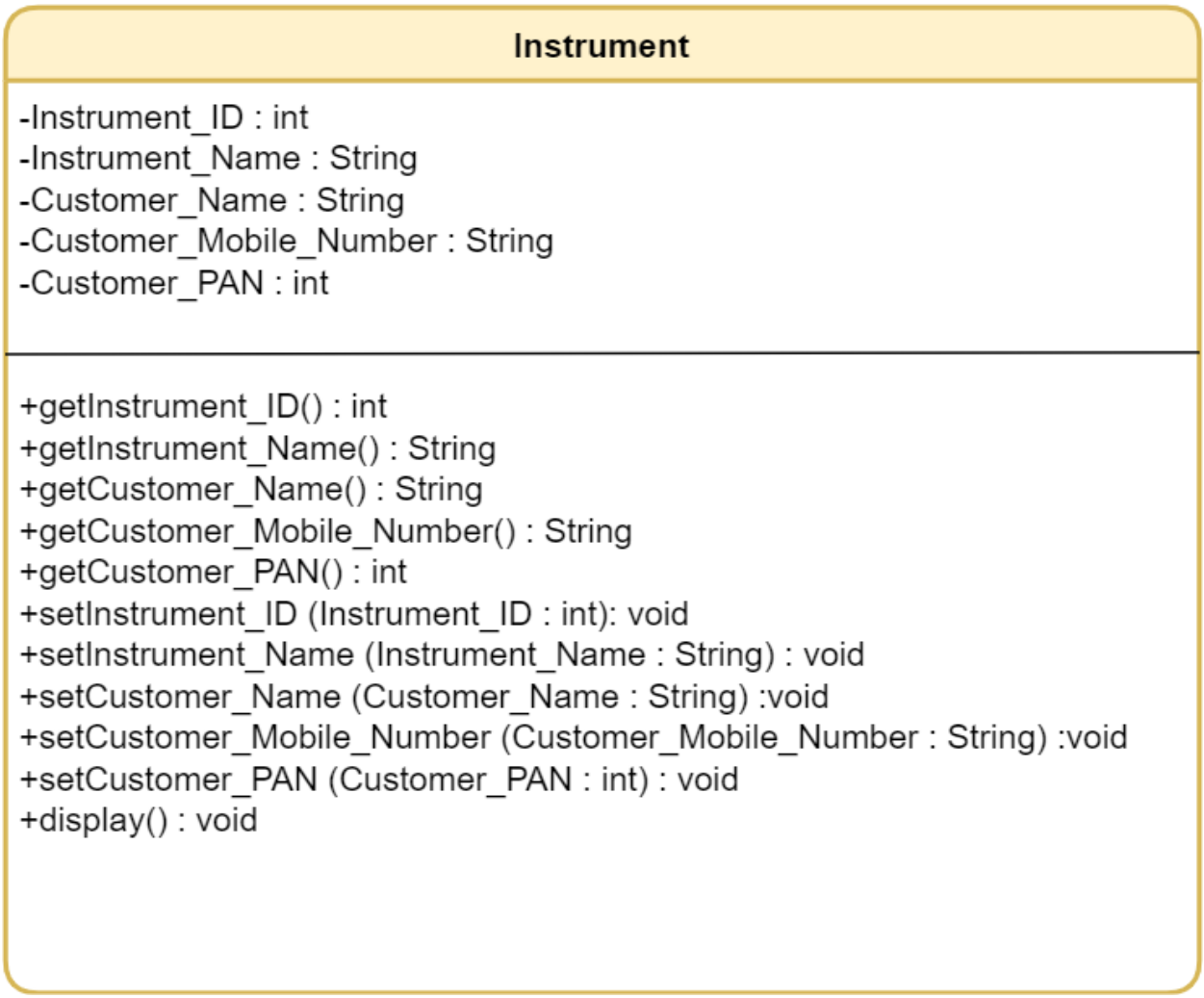

Figure 1: Class Diagram of Instrument

2.2 Class Diagram of Instrument to Rent Class

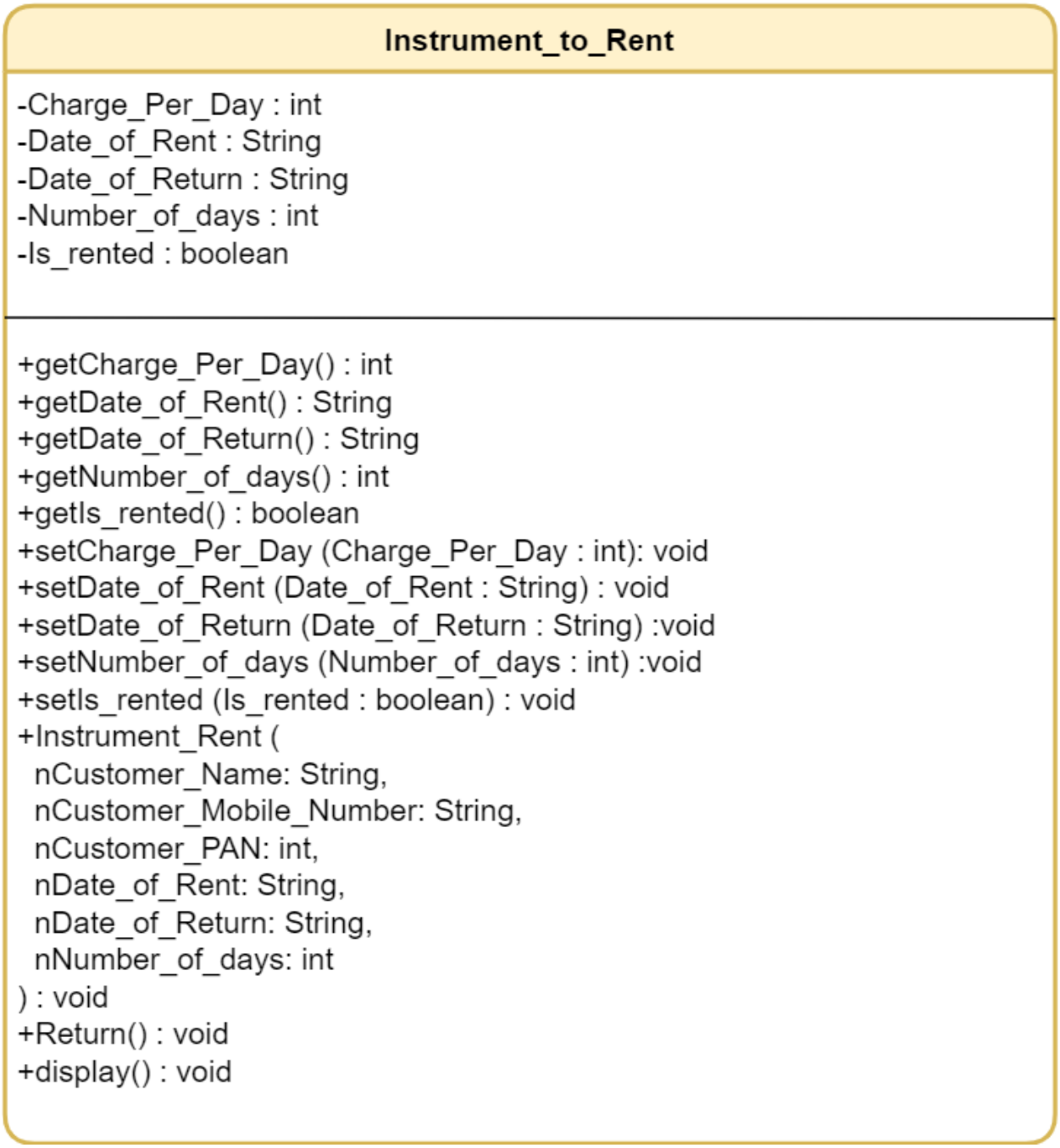

Figure 2: Class Diagram of Instrument to Rent

2.3 Class Diagram of Instrument to Sell Class

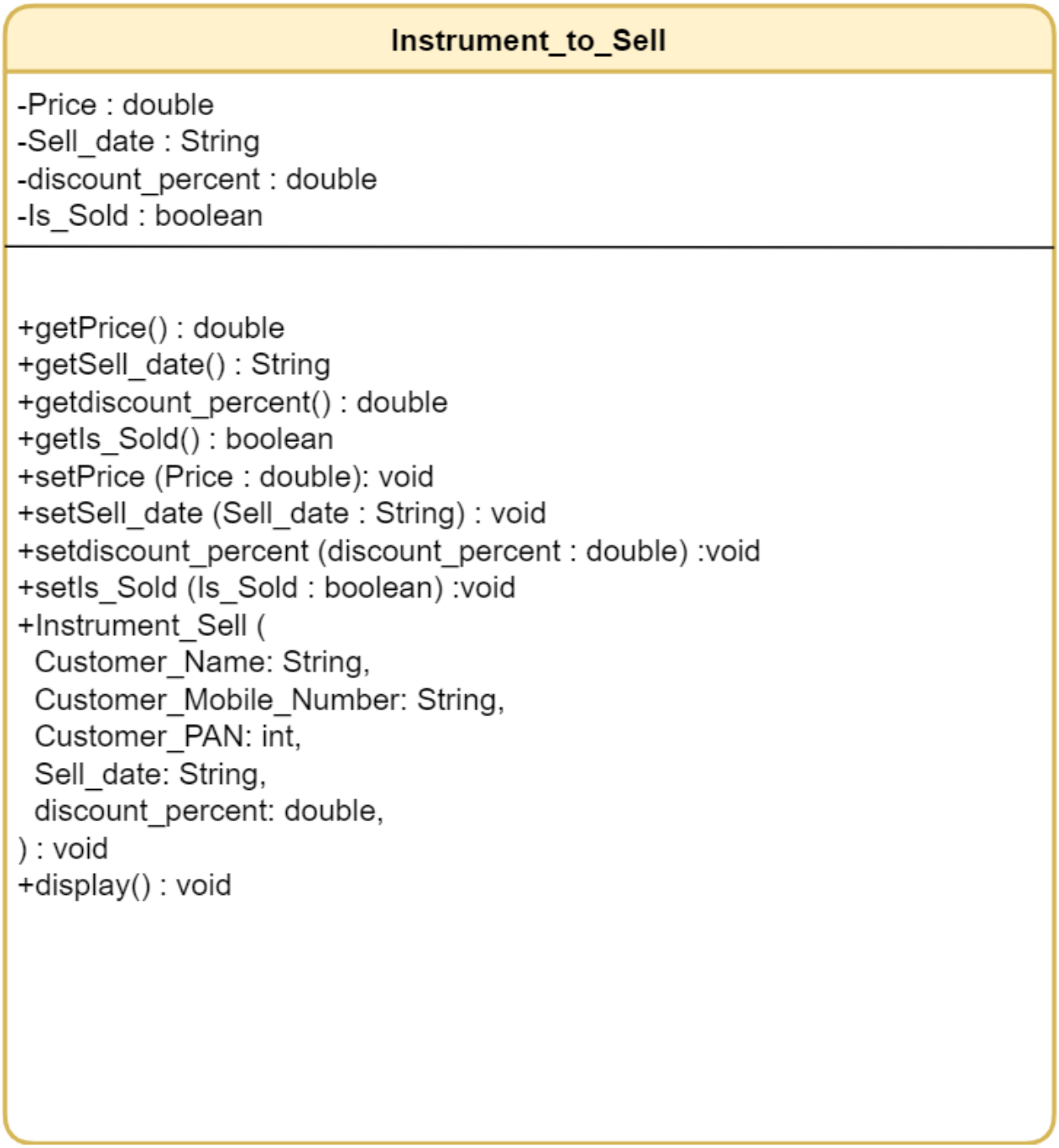

Figure 3: Class Diagram of Instrument to Sell

2.4 Class Diagram of Sarangi Sansar Class

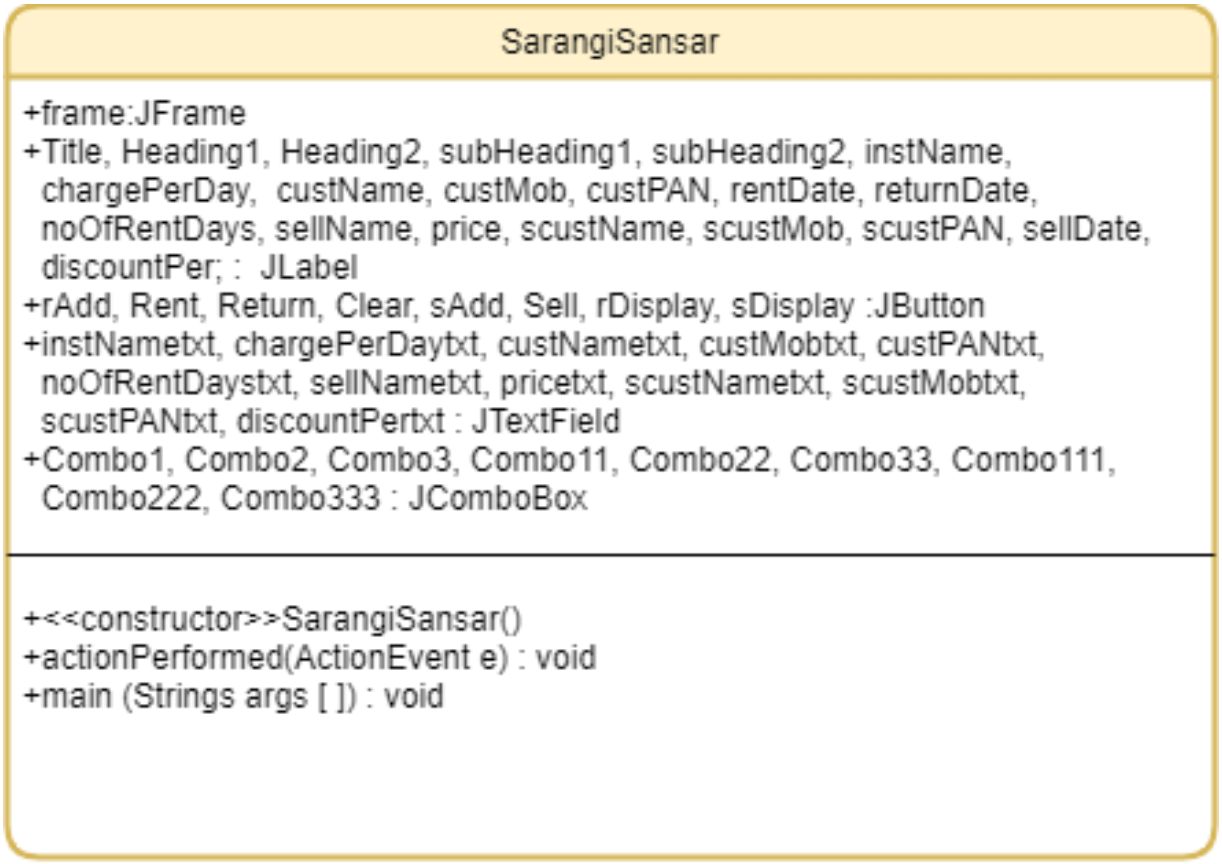

Figure 4: Class Diagram of Sarangi Sansar Class

2.5 Relationship Diagram of Sarangi Sansar Class

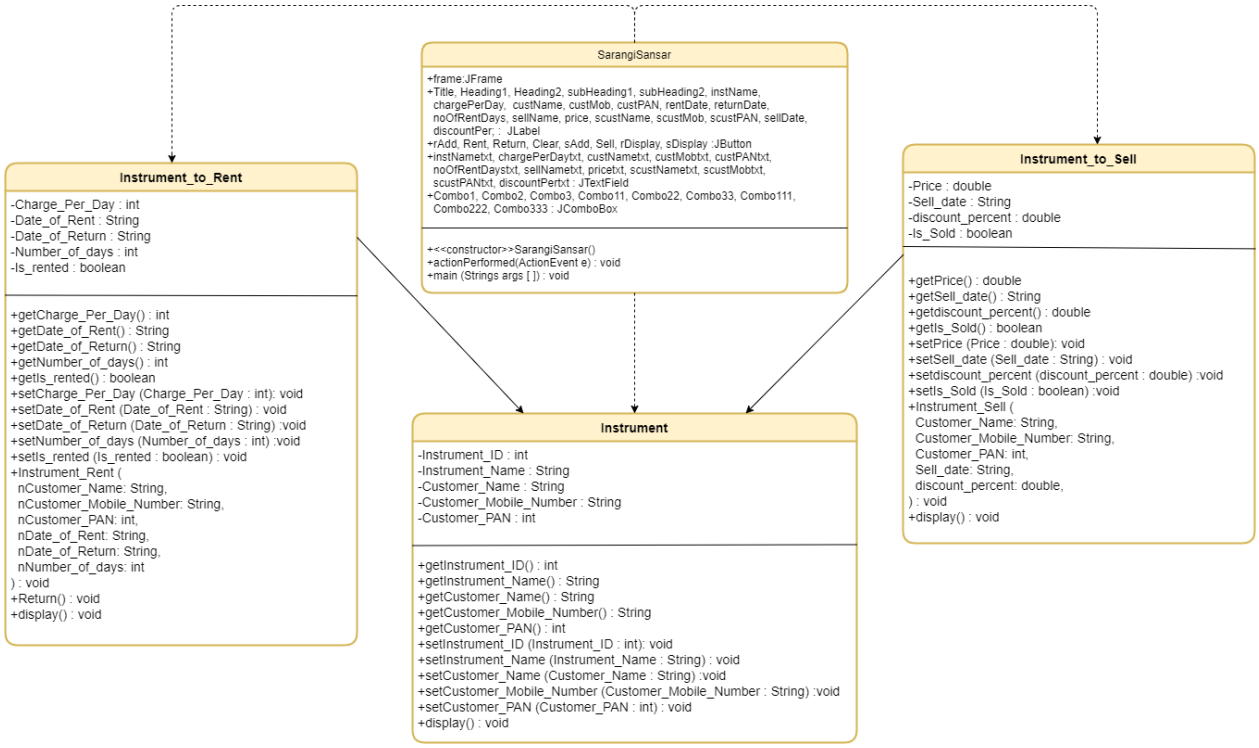

Figure 5: Relationship Diagram of Sarangi Sansar Class

3. Pseudocode

```
IMPORT java.swing.*
IMPORT java.awt.Color
IMPORT java.awt.Dimension
IMPORT java.awt.Font
IMPORT java.awt.event.ActionListener
IMPORT java.awt.event.ActionEvent*
IMPORT java.util.ArrayList

CREATE a public class SarangiSansar() that implements
            ActionListener interface
DO

        DECLARE public component of JFrame
        DECLARE public component of JLabel
        DECLARE public component of JButton
        DECLARE public component of JTextField
        DECLARE public component of JComboBox

CREATE a constructor SarangiSansar()
DO

        INITIALIZE frame as a new JFrame
        SET visible for Jframe
        SET size for Jframe
        SET layout for Jframe
        SET Sarangi Sansar - Instrument Sell and Rent as title for Jframe



        INITIALIZE JLabels
```

```
        INITIALIZE JTextFields
        INITIALIZE JComboBox
        INITIALIZE JButton,

        SET color for JFrame,
        SET font for JLabels,
        SET ActionListener for TextFields and for Buttons

        ADD required components to JFrame

        SET setBounds for JLabels, JTextFields, JButtons

    END DO

    INITIALIZE arraylist named List



    CREATE a method actionPerformed(ActionEvent e)
    DO
        IF event source is equal to rAdd button
        DO
            SET boolean ItemAdd as false

            IF textfields instNametxt and chargePerDaytxt is empty
            DO
                DISPLAY message
            END DO
        END IF
```

```
ELSE
DO
    TRY
    DO
        Instrument_Name is equal to instNametxt
        CONVERT textfield into int and store value

        IF List is empty
        DO
            CREATE object named ObjA with 2 parameters
            ADD ObjA to List
            DISPLAY message
        END DO
        END IF

        ELSE
            FOR instrument stored in arraylist
            DO
                IF object is instance of Instrument_To_Rent
                class
                DO

                IF instrument already exists
                DO
                    SET ItemAdd as true
                    DISPLAY message
                END DO
                END IF
```

```
                    IF ItemAdd boolean is false

                    CREATE object named ObjA with 2
                    parameters
                    ADD ObjA to List

                    END IF
                END DO
                END FOR
            CATCH a number format exception ex
            DO
                DISPLAY message
            END DO
        END DO


IF event source is equal to Rent button
    DO
        SET boolean ItemRent as false

        IF textfields custNametxt, custMobtxt, custPANtxt,and
        noOfRentDaystxt is empty
        DO
            DISPLAY message
        END DO
    END IF
```

```
ELSE
DO
    FOR instrument stored in arraylist
    DO
        IF object is instance of Instrument_To_Rent class
        DO
            IF instrument already exists
            DO
                ItemRent boolean is true
            END DO
            END IF

        TRY
        DO
            IF ItemRent boolean is false
            DO
                DISPLAY message
            END DO
            END IF

            ELSE
                Customer_Name is equal to custNametxt
                Customer_Mobile_Number is equal to
                custMobtxt
                CONVERT Customer_PAN into int and store
                value IN custPANtxt
                Date_of_Rent is equal to selected item
                Date_of_Return is equal to selected item
                CONVERT Number_of_days into int and
                store value in noOfRentDaystxt
```

```
                FOR instrument stored in arraylist
                DO
                        IF object is instance of Instrument_To_Rent class
                        DO

                        IF instrument already exists
                        DO
                                IF Is_rented is not true
                                DO

                                Add parameters to a new object ObjR
                                DISPLAY message

                                END DO
                                END IF

                                ELSE
                                        DISPLAY message

        CATCH a number format exception ex
        DO
                DISPLAY message
        END DO
END DO
```

```
IF event source is equal to Return button
DO
        IF textfields instNametxt is empty
        DO
                DISPLAY message
        END DO
END DO
END IF

ELSE
DO
        FOR instrument stored in arraylist
        DO
                Insturment_Name is equal to instNametxt
                IF object is instance of Instrument_To_Rent class
                DO
                                IF instrument already exists
                                IF Is_rented is not true
                                DO
                                        DISPLAY message
                                END DO
                                END IF

                                ELSE
                                        RETURN object ObjRe
                                        DISPLAY message
                END DO
        END DO
END DO
```

```
            IF event source is equal to Clear button
        DO
            Instrument_Name is equal to NULL
            Charge_Per_Day is equal to NULL
            Customer_Name is equal to NULL
            Customer_Mobile_Number is equal to NULL
            Customer_PAN is equal to NULL
            Number_of_days is equal to NULL
            Selling_Instrument_Name is equal to NULL
            Price is equal to NULL
            Selling_Customer_Name is equal to NULL
            Selling_Customer_Mobile_Number is equal to NULL
            Selling_Customer_PAN is equal to NULL
            Discount_Percentage is equal to NULL

    END DO
        END IF

    IF event source is equal to sAdd button
        DO
            SET boolean ItemASell as false

            IF textfields sellNametxt and pricetxt is empty
            DO
                DISPLAY message
            END DO
        END IF



        ELSE
```

```
DO
    TRY
    DO
        Instrument_Name is equal to sellNametxt
        CONVERT price into int and store value in pricetxt

        IF List is empty
        DO
            CREATE object named ObjS with 2 parameters
            ADD ObjS to List
            DISPLAY message
        END DO
        END IF

        ELSE
            FOR instrument stored in arraylist
            DO
                IF object is instance of Instrument_To_Sell
                class
                DO

                IF instrument already exists
                DO
                    SET ItemSell as true
                    DISPLAY message
                END DO
                END IF



                IF ItemSell boolean is false
```

```
                    CREATE object named ObjS with 2
                    parameters
                    ADD ObjS to List

                    END IF
                END DO
                END FOR
            CATCH a number format exception ex
            DO
                DISPLAY message
            END DO
        END DO


IF event source is equal to Sell button
    DO
        SET boolean ItemSell as false

        IF textfields scustNametxt, scustMobtxt, scustPANtxt,and
        discountPertxt is empty
        DO
            DISPLAY message
        END DO
    END IF

    ELSE
    DO
        FOR instrument stored in arraylist
        DO
            IF object is instance of Instrument_To_Sell class
```

```
DO
        IF instrument already exists
        DO
                ItemSell boolean is true
        END DO
        END IF

TRY
DO
        IF ItemSell boolean is false
        DO
                DISPLAY message
        END DO
        END IF

        ELSE
                Customer_Name is equal to custNametxt
                Customer_Mobile_Number is equal to
                custMobtxt
                CONVERT Customer_PAN into int and store
                value IN custPANtxt
                Sell_Date is equal to selected item
                CONVERT discount_percent into int and
                store value in discountPertxt



FOR instrument stored in arraylist
DO
        IF object is instance of Instrument_To_Sell class
        DO
```

```
                    IF instrument already exists
                    DO
                        IF Is_Sold is not true
                        DO

                        Add parameters to a new object SoldObj
                        DISPLAY message

                        END DO
                        END IF

                        ELSE
                            DISPLAY message

            CATCH a number format exception ex
            DO
                DISPLAY message
            END DO
        END DO

    IF event source is equal to rDisplay button
        DO
            SET boolean disR as false

            IF textfields instNametxt is empty
            DO
                DISPLAY message
            END DO
        END IF
```

```
        ELSE
        DO
            FOR instrument stored in arraylist
            DO
                IF object is instance of Instrument_To_Rent class
                DO
                    IF instrument already exists
                    DO
                        disR boolean is true
                    END DO
                    END IF

            IF boolean disR is true
            DO
                Instrument_Name is equal to instNametxt
                FOR instrument stored in arraylist
                DO
                    IF object is instance of Instrument_To_Rent class
                    DO

                    IF instrument already exists
                    DO
                        CREATE new object rDisplay for
                        Instrument_to_Rent class
                        DISPLAY message
                    END DO
                    END IF
            ELSE
                DISPLAY message
    END IF
```

```
IF event source is equal to sDisplay button
    DO
        SET boolean disS as false

        IF textfields sellNametxt is empty
        DO
            DISPLAY message
        END DO
    END IF

    ELSE
    DO
        FOR instrument stored in arraylist
        DO
            IF object is instance of Instrument_To_Sell class
            DO
                IF instrument already exists
                DO
                    disS boolean is true
                END DO
                END IF

        IF boolean disS is true
        DO
            Instrument_Name is equal to sellNametxt
            FOR instrument stored in arraylist
            DO
                IF object is instance of Instrument_To_Sell class
                DO
```

```
                    IF instrument already exists
                    DO
                        CREATE new object sDisplay for
                        Instrument_to_Sell class
                        DISPLAY message
                    END DO
                    END IF
            ELSE
                DISPLAY message
    END IF


        CREATE main method main(String[ ] args)
        DO
            CALL constructor SarangiSansar
        END DO
END DO
```

4. Description of Method

Method	Description
actionPerformed(ActionEvent e)	actionPerformed handles all button-related events in the GUI. Simply the functionality of the buttons is defined here. For example, whenever user clicks a button a message box with appropriate message is displayed. The details of all the buttons are given below
Buttons	**Description of Buttons**
Button: rAdd	When the button is clicked the input values of Instrument Name and Charge Per day are stored in their respected text fields. If the input value of Charge Per Day is not entered in integer, an error message is displayed. If the text fields are left blank after pressing the “Add to Rent” button, then a message box is shown. If the input values are met with the conditions given then values are stored in a new object which is then added to the array list of Instrument class.
Button: Rent	The values of Customer Name, Customer Mobile Number, Customer PAN Number, Rent and Return date, and Number of Renting Days are stored after entered in the GUI. After the “Rent” button is pressed the input value of the Instrument name is compared to

	the existing instrument name and if a valid instrument name has been submitted, it is used to rent the right instrument If the input value of Customer PAN Number and Number of Rent Days are not entered in integer, an error message is displayed. If the text fields are left blank after pressing the “Rent” button, then a message box is shown. If the input values are met with the conditions given then values are stored in a new object which is then added to the array list of Instrument class and a message “Instrument Rented” will be displayed.
Button: Return	The input value of the Instrument is compared to the existing instrument name when the "Return" button is pushed, and if a valid instrument name has been entered, it is used to return the right instrument from the array list of Instruments.
Button: Clear	When this button is pressed, the values from all the text fields in the GUI are cleared.
Button: sAdd	When the button is clicked the input values of Instrument Name and Price are stored in their respected text fields. If the input value of Price is not entered in integer, an error message is displayed. If the text fields are left blank after pressing the “Put on Sell” button, then a message box is shown.

	If the input values are met with the conditions given then values are stored in a new object which is then added to the array list of Instrument class.
Button: Sell	The values of Customer Name, Customer Mobile Number, Customer PAN Number, Return Date, and Discount Percentage are stored after entered in the GUI. When the sale button is pushed, the instrument name input value is checked with the input instrument name, and if it matches, the corresponding instrument from the list then a message box “Instrument Sold” will be displayed. If the input value of Customer PAN Number and Discount Percentage are not entered in integer, an error message is displayed. If the text fields are left blank after pressing the “Sell” button, then a message box is shown. If the input values are met with the conditions given then values are stored in a new object which is then added to the array list of Instrument class and a message “Instrument Sold Successfully” will be displayed.

Button: rDispaly	When this button is pressed, the information of the Renting Customer with the respective Instrument Name and other details is displayed.
Button: sDisplay	When this button is pressed, the information of the Selling Customer with the respective Instrument Name and other details is displayed.

Table 1: Method Description of SarangiSansar class

5. Testing

5.1 Test 1: To test that the program can be compiled and run using the command prompt.

Test No:	**1**
Objective:	To test that the program can be compiled and run using the command prompt.
Action:	Run Command Prompt and go into the directory of the program. Use the command “javac SarangiSansar.java” to compile the program Use the command “java SarangiSansar” to run the program.
Expected Result:	The required GUI pops up after running the program.
Actual Result:	The required GUI popped up after running the program.
Conclusion:	The test was carried out successfully.

Table 2: Compiling and Running Program Using Command Prompt

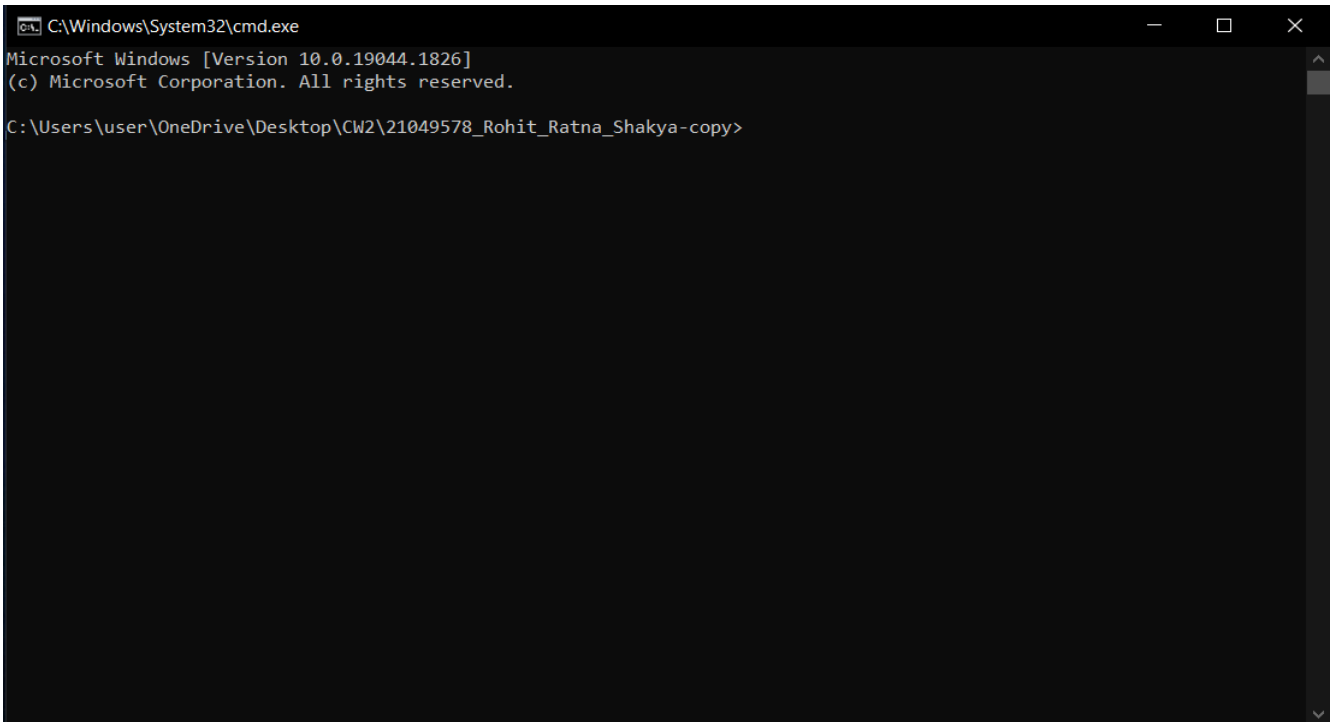

Figure 6:Running Command Prompt in Program's Directory

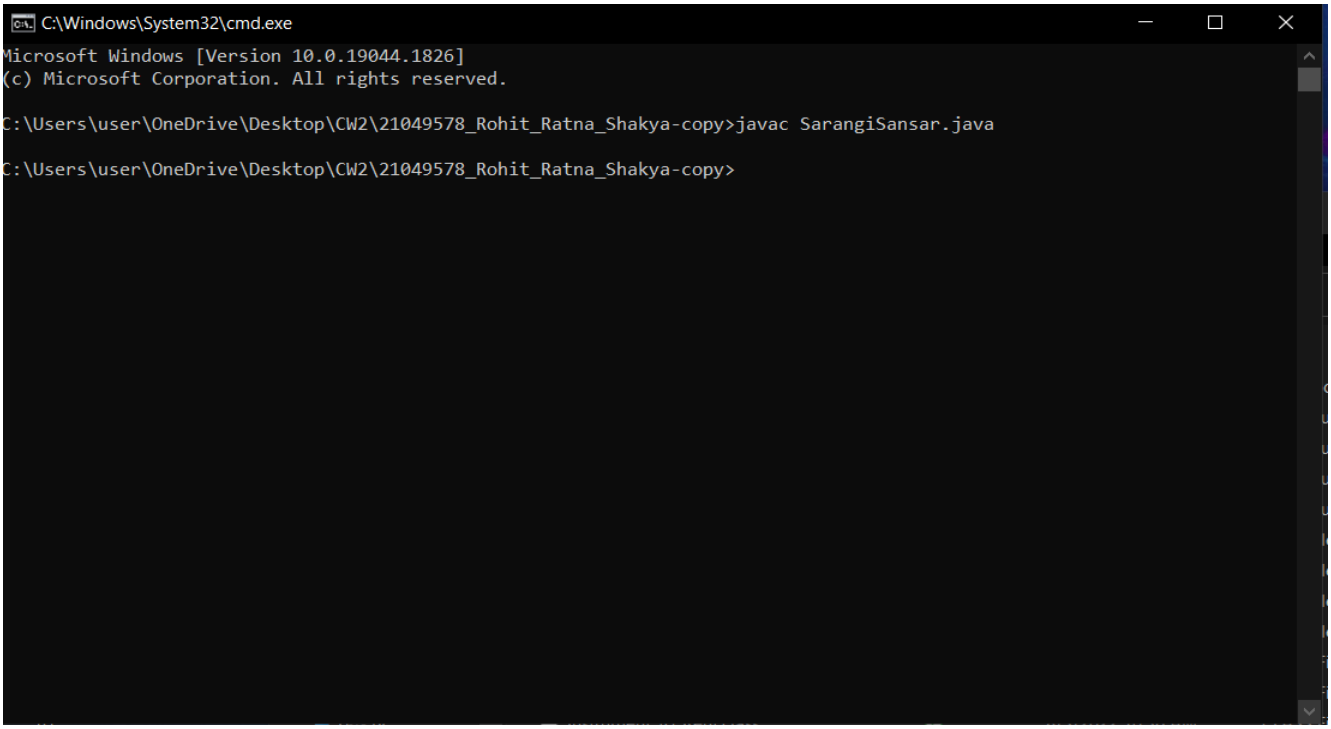

Figure 7:Compiling the Program using Command Prompt

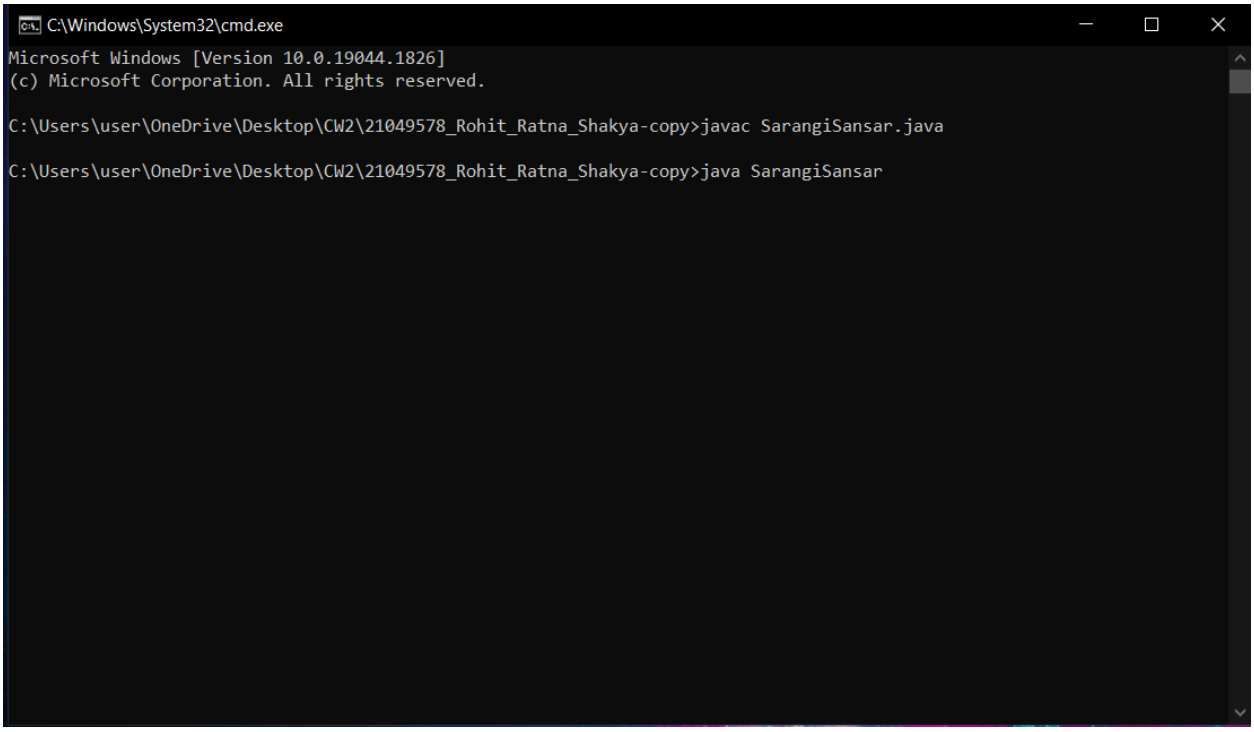

Figure 8: Running the program using Command Prompt

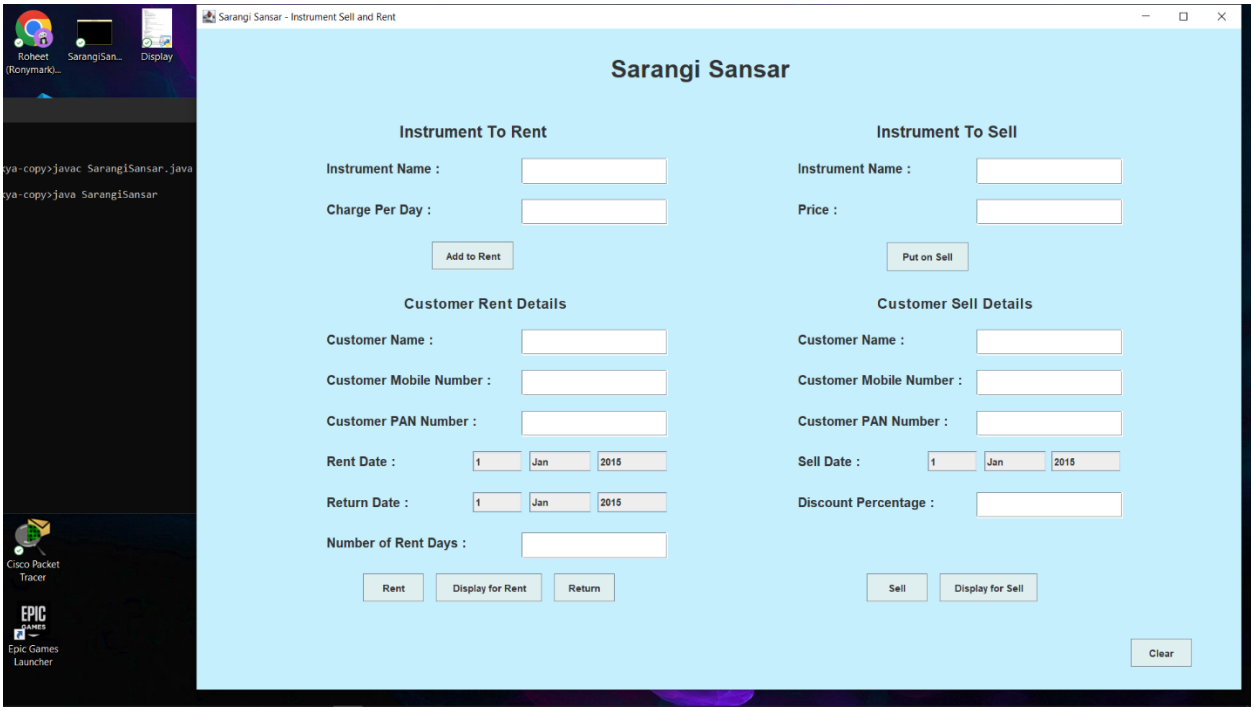

Figure 9: GUI pops up without any Error

5.2 Test 2: To Add objects of Instrument To Rent and Instrument To Sell, Rent, Sell and Return the Instrument

5.2.1 Adding Instrument to Rent

Test No:	**2.1**
Objective:	To Add Instrument to Rent
Action:	Instrument Name and Charge Per Day was added in “Instrument to Rent” section and the following values were added: ➢ Instrument Name: Drum ➢ Charge Per Day: 800 “Add to Rent” button was clicked
Expected Result:	A message box pops up confirming that the instrument is added to rent.
Actual Result:	A message box popped up confirming that the instrument is added to rent.
Conclusion:	The test was carried out successfully.

Table 3: Adding Instrument to Rent

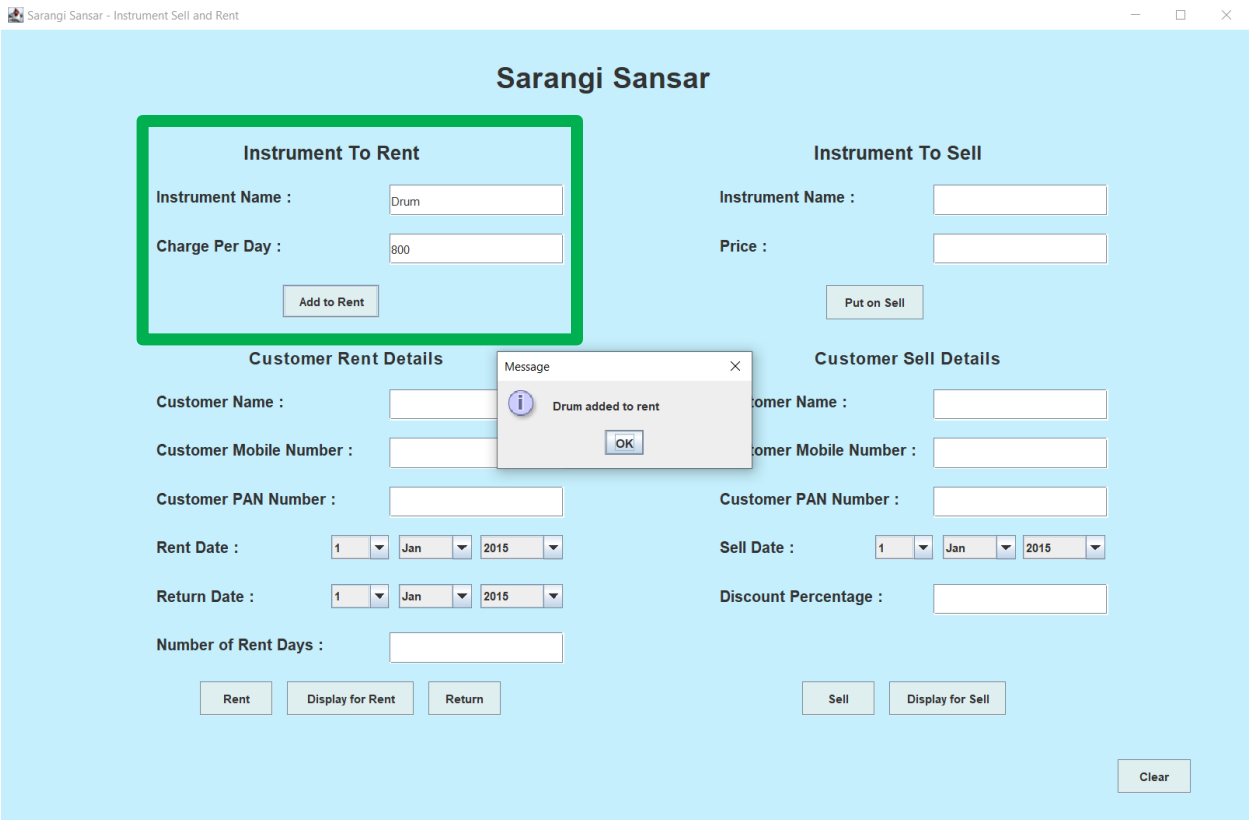

Figure 10: Adding Instrument to Rent

5.2.2 Adding Instrument to Sell

Test No:	**2.2**
Objective:	To Add Instrument to Sell
Action:	Instrument Name and Price was added in “Instrument to Sell” section and the following values were added: ➢ Instrument Name: Piano ➢ Charge Per Day: 2000 “Put on Sell” button was clicked
Expected Result:	A message box pops up confirming that the instrument is added to sell.
Actual Result:	A message box popped up confirming that the instrument is added to sell.
Conclusion:	The test was carried out successfully.

Table 4: Adding Instrument to Sell

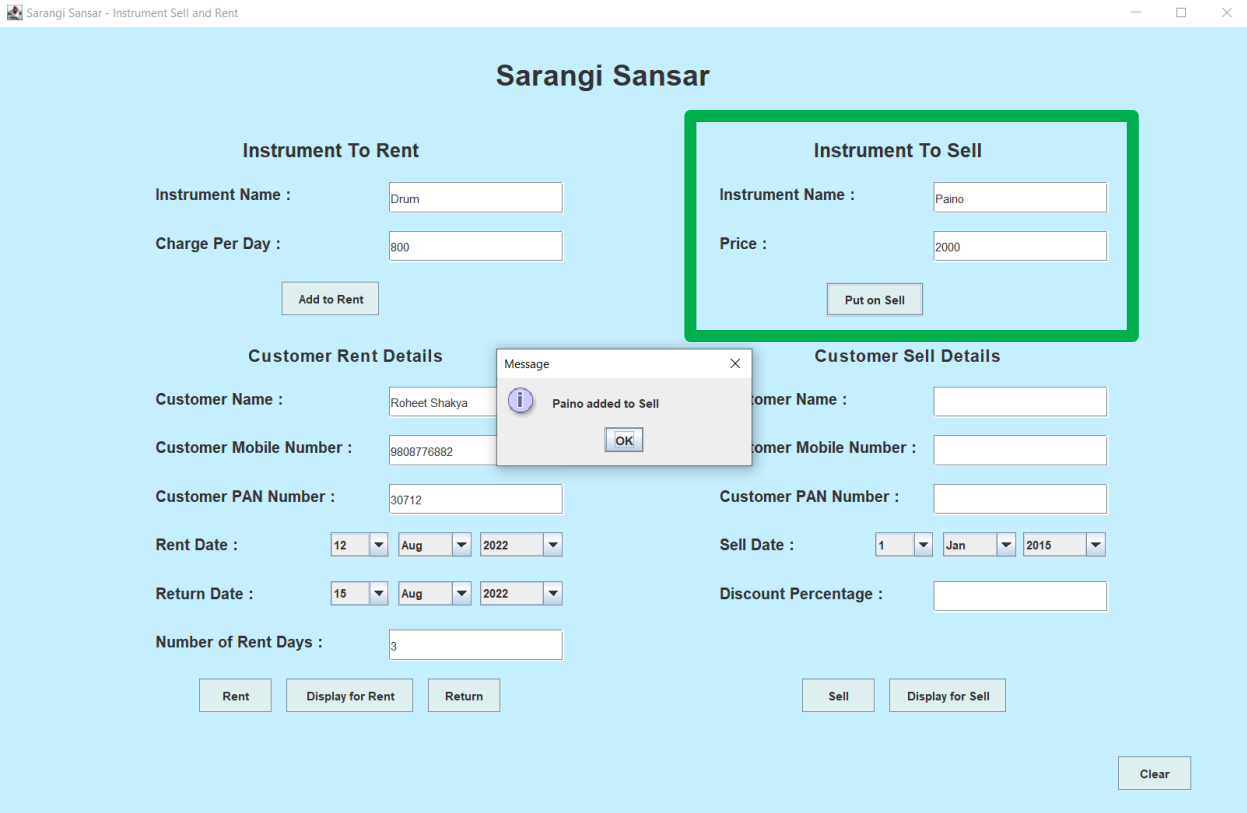

Figure 11: Adding Instrument to Sell

5.2.3 Renting the Instrument

Test No:	**2.3**
Objective:	To rent the Instrument
Action:	Customer Name, Customer Mobile Number, Customer PAN Number, Rent date, Return Date and Number of Renting days rented was added in “Customer Rent Details” section and the following values were added: ➢ Customer Name: Roheet Shakya ➢ Customer Mobile Number: 9808776882 ➢ Customer PAN Number: 30712 ➢ Date of Rent: 12th August 2022 ➢ Date of Return: 15th August 2022 ➢ Number of Renting Days: 3 “Rent” button was clicked
Expected Result:	A message box pops up confirming that the instrument has been successfully rented by the customer.
Actual Result:	A message box popped up confirming that the instrument has been successfully rented by the customer.
Conclusion:	The test was carried out successfully

Table 5: Renting the Instrument

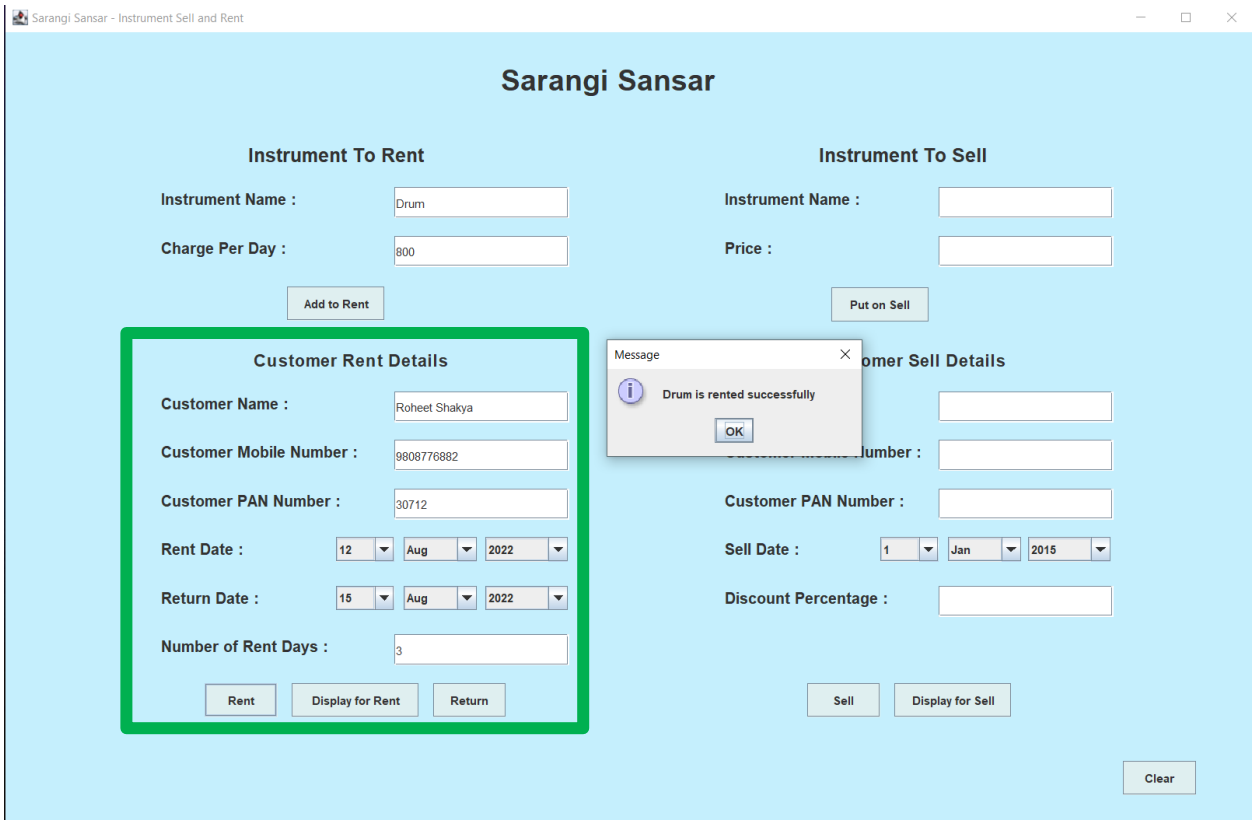

Figure 12: Renting the Instrument

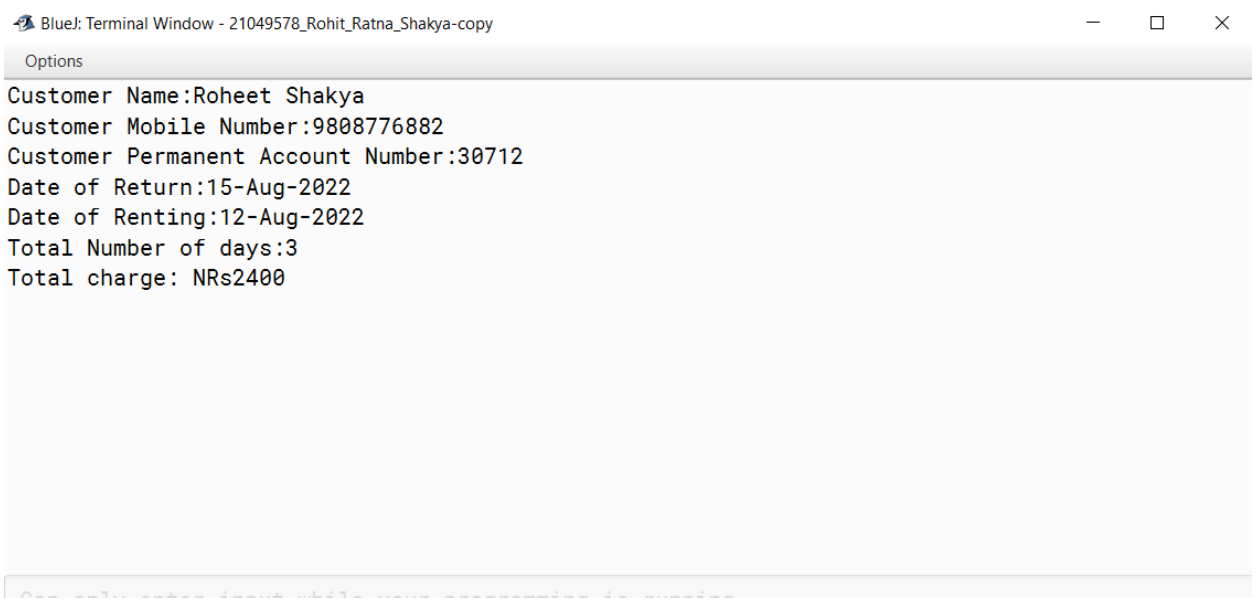

Figure 13: Rented Instrument details displayed in the terminal

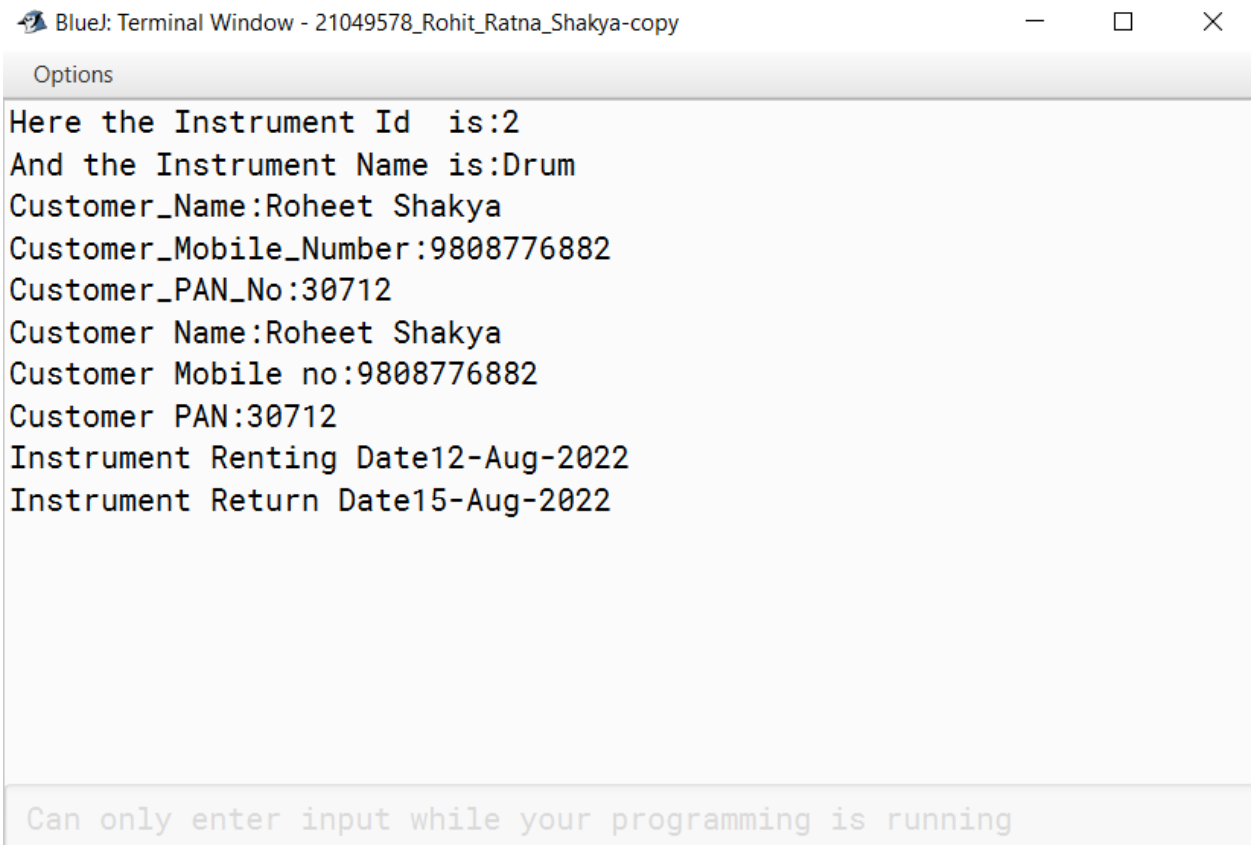

Figure 14: Rented Instrument details displayed after clicking Display Button

5.2.4 Selling the Instrument

Test No:	**2.3**
Objective:	To sell the Instrument
Action:	Customer Name, Customer Mobile Number, Customer PAN Number, Sell Date and Discount Percentage was added in “Customer Sell Details” section and the following values were added: ➢ Customer Name: Roheet Shakya ➢ Customer Mobile Number: 9808776882 ➢ Customer PAN Number: 30712 ➢ Sell Date: 12th August 2022 ➢ Discount Percentage: 20 “Sell” button was clicked
Expected Result:	A message box pops up confirming that the instrument has been successfully sold to the customer.
Actual Result:	A message box popped up confirming that the instrument has been successfully sold to the customer.
Conclusion:	The test was carried out successfully

Table 6: Selling the Instrument

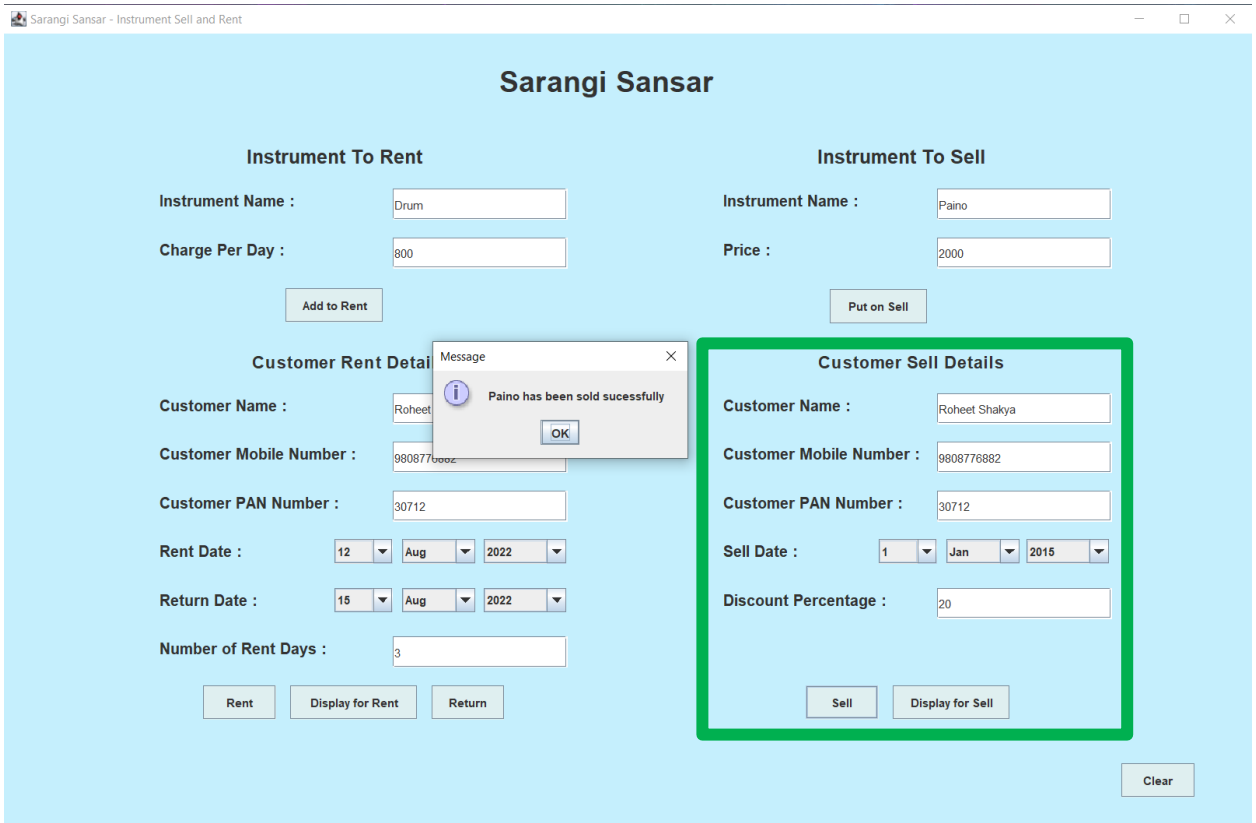

Figure 15: Selling the Instrument

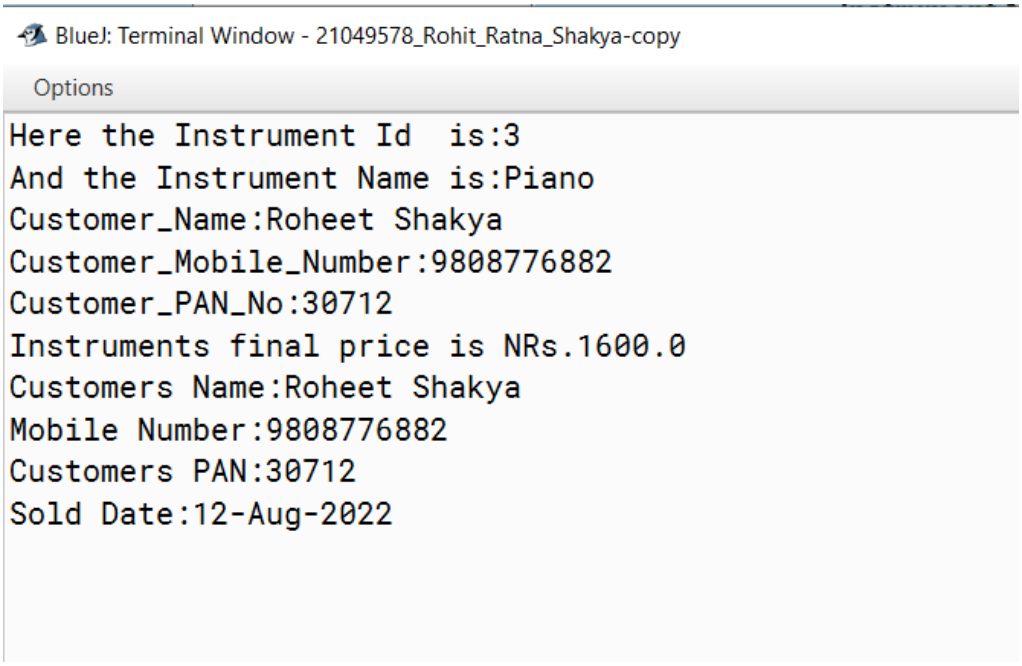

Figure 16: Sold Instrument details displayed after clicking Display Button

5.2.5 Returning the Instrument

Test No:	**2.5**
Objective:	To return the Instrument that was rented.
Action:	The Instrument Name which was rented is added on the “Instrument to Rent” section as: ➢ Instrument Name: Drum “Return” button was clicked.
Expected Result:	A message box pops up confirming that the instrument has been successfully returned.
Actual Result:	A message box popped up confirming that the instrument has been successfully returned.
Conclusion:	The test was carried out successfully

Table 7: Returning the Instrument

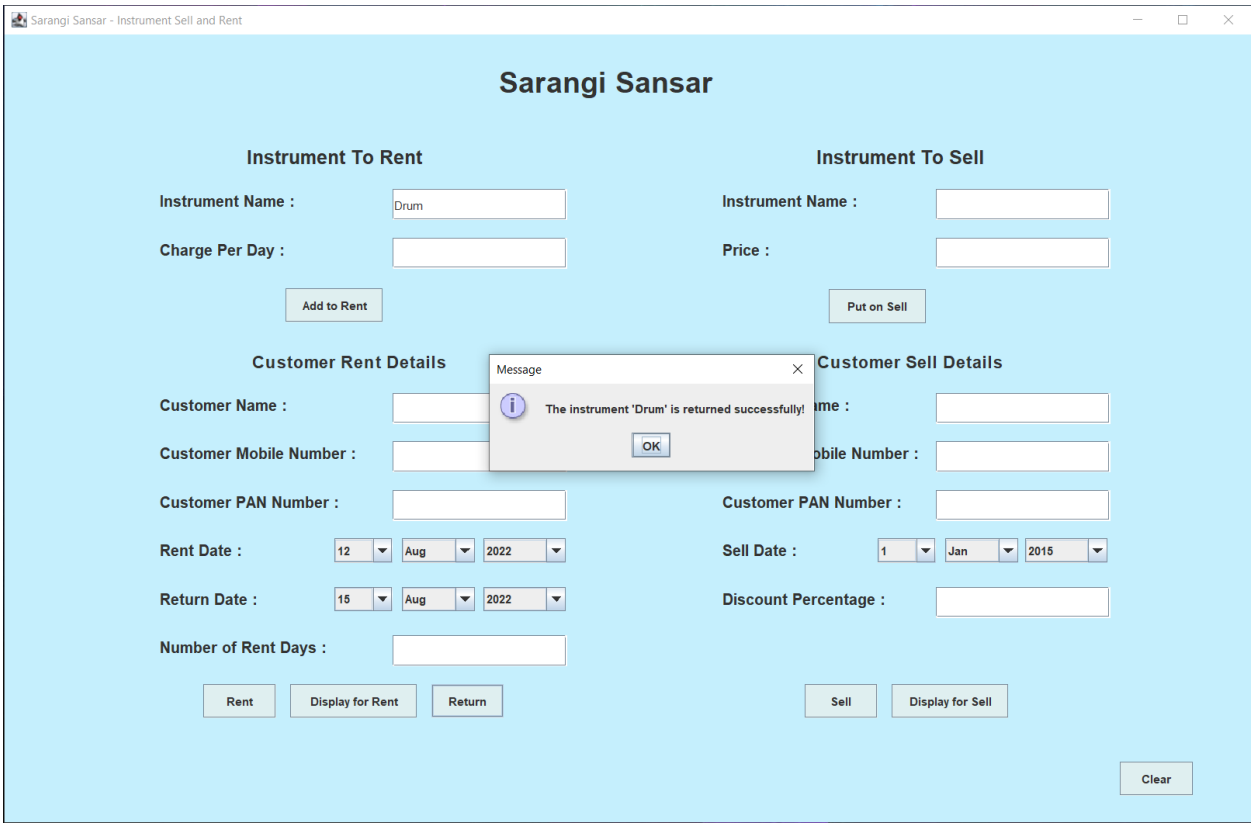

Figure 17: Returning the Instrument

5.3 Test 3: To show appropriate dialog box when unsuitable values are entered

Test No:	**3**
Objective:	To show appropriate dialog box when unsuitable values are entered
Action:	➢ Click Add button without entering any input of Instrument Name. ➢ To use string value in Charge Per Day of Instrument to Rent. ➢ To Rent and Instrument which is not added yet. ➢ To Rent an already rented Instrument. ➢ To Return an Instrument without any Instrument Name. ➢ To add the same Instrument Twice in Instrument to Sell.
Expected Result:	Dialog Box should appear with specific message for specific errors.
Actual Result:	Dialog Box appeared with specific message for specific errors.
Conclusion:	The test was carried out successfully

Table 8: To show appropriate dialog box when unsuitable values are entered

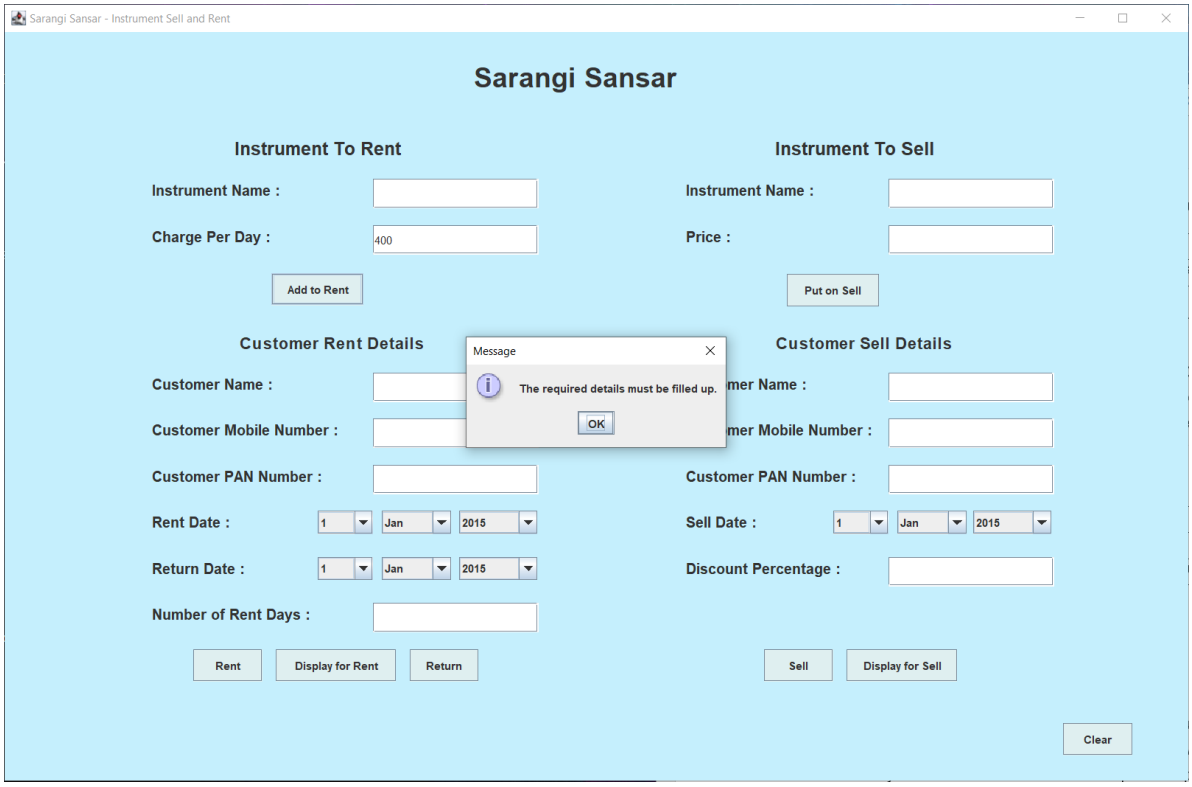

Figure 18: Click Add button without entering any input of Instrument Name.

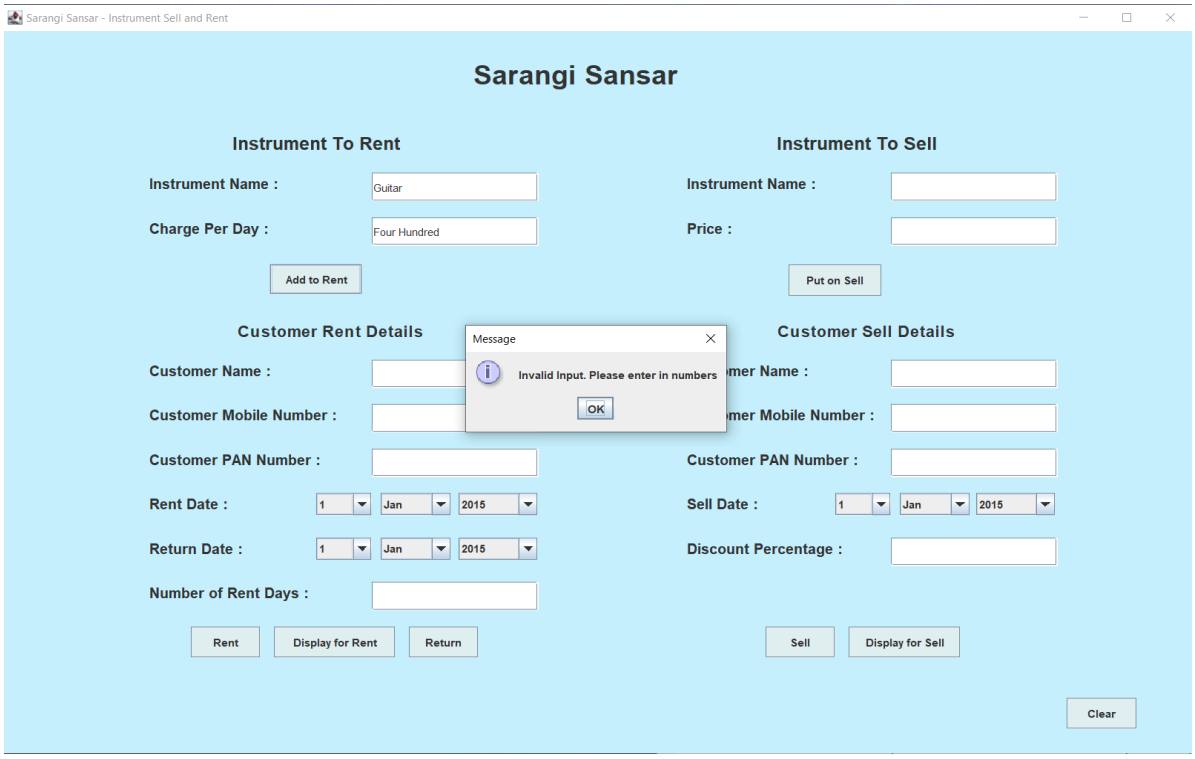

Figure 19: To use string value in Charge Per Day of Instrument to Rent.

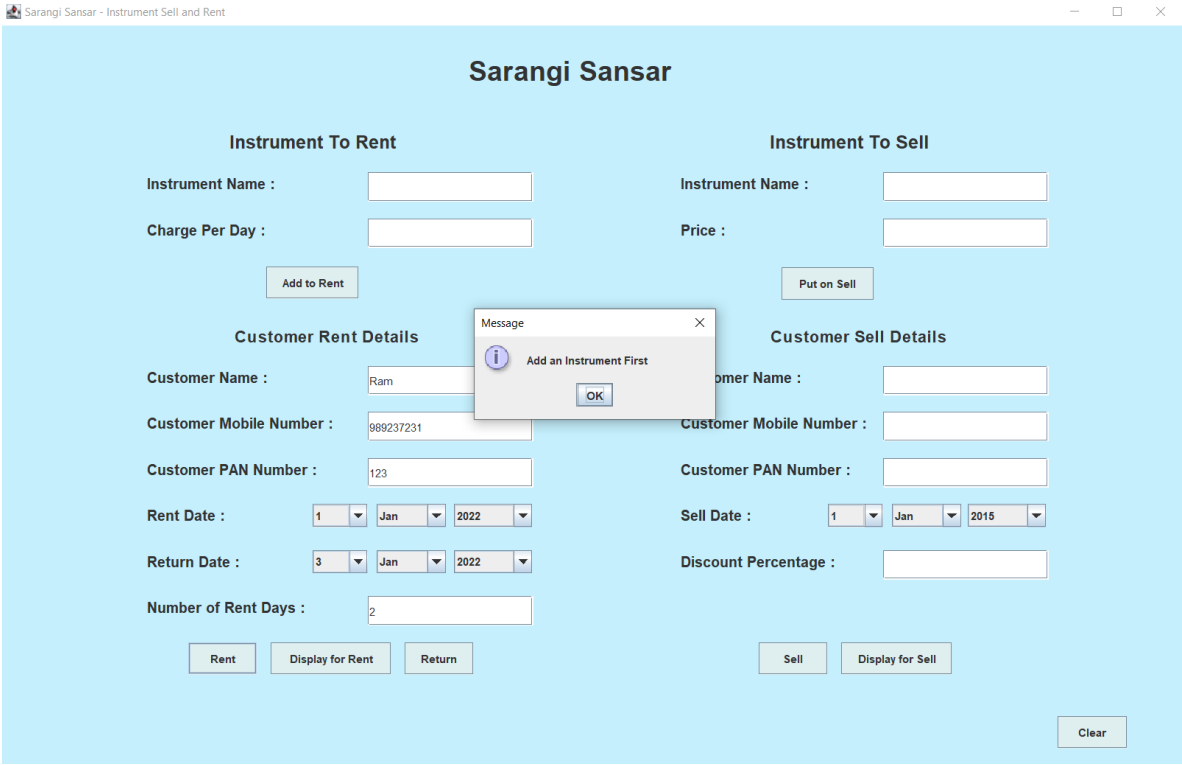

Figure 20: To Rent and Instrument which is not added yet.

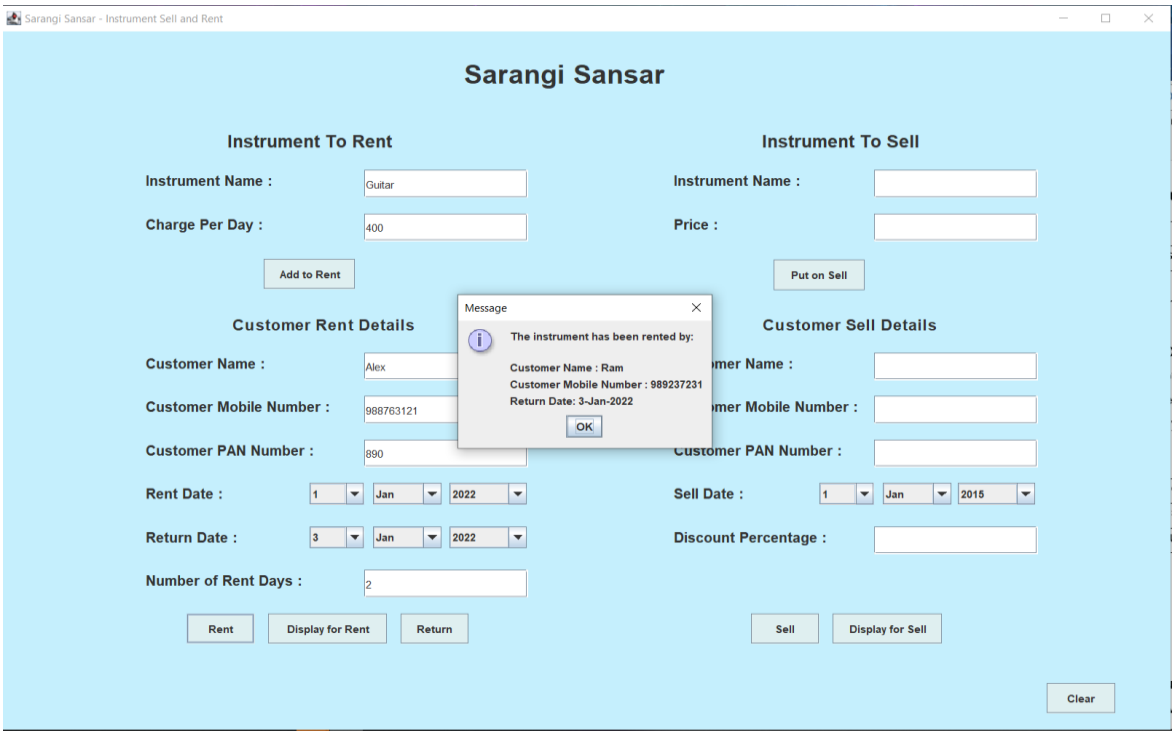

Figure 21: To Rent an already rented Instrument.

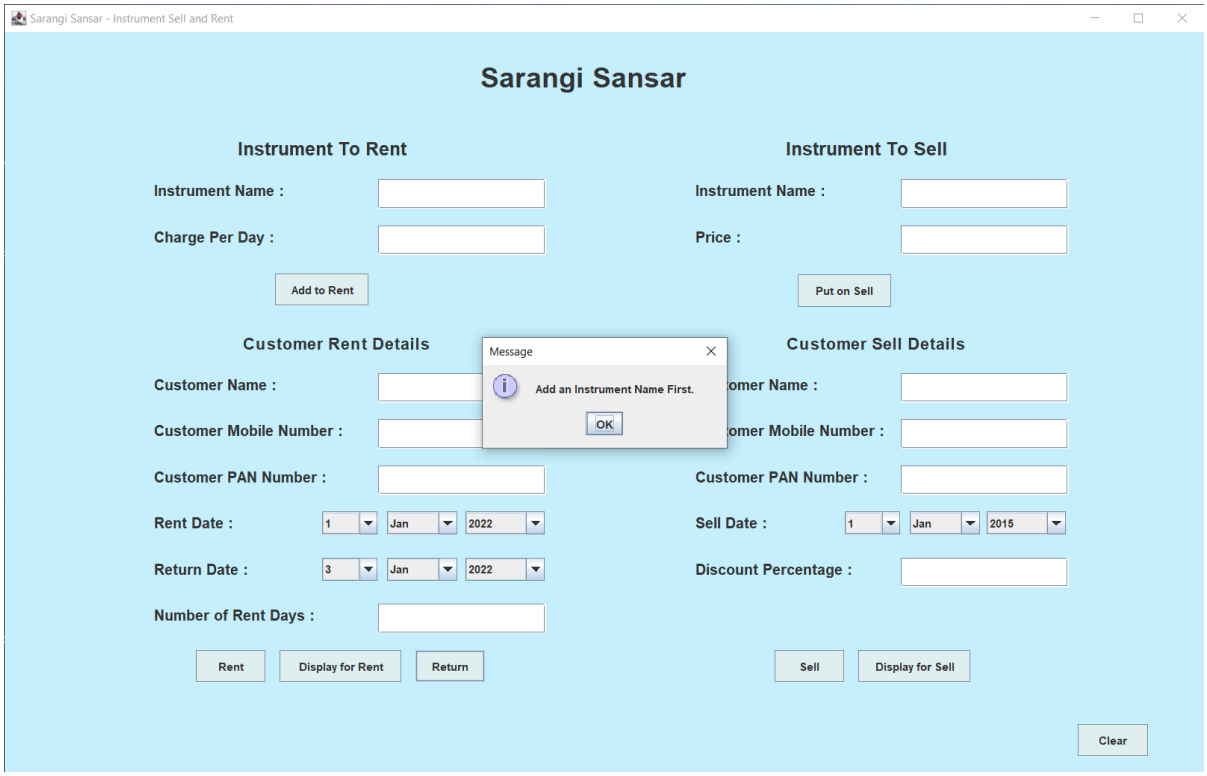

Figure 22: To Return an Instrument without any Instrument Name.

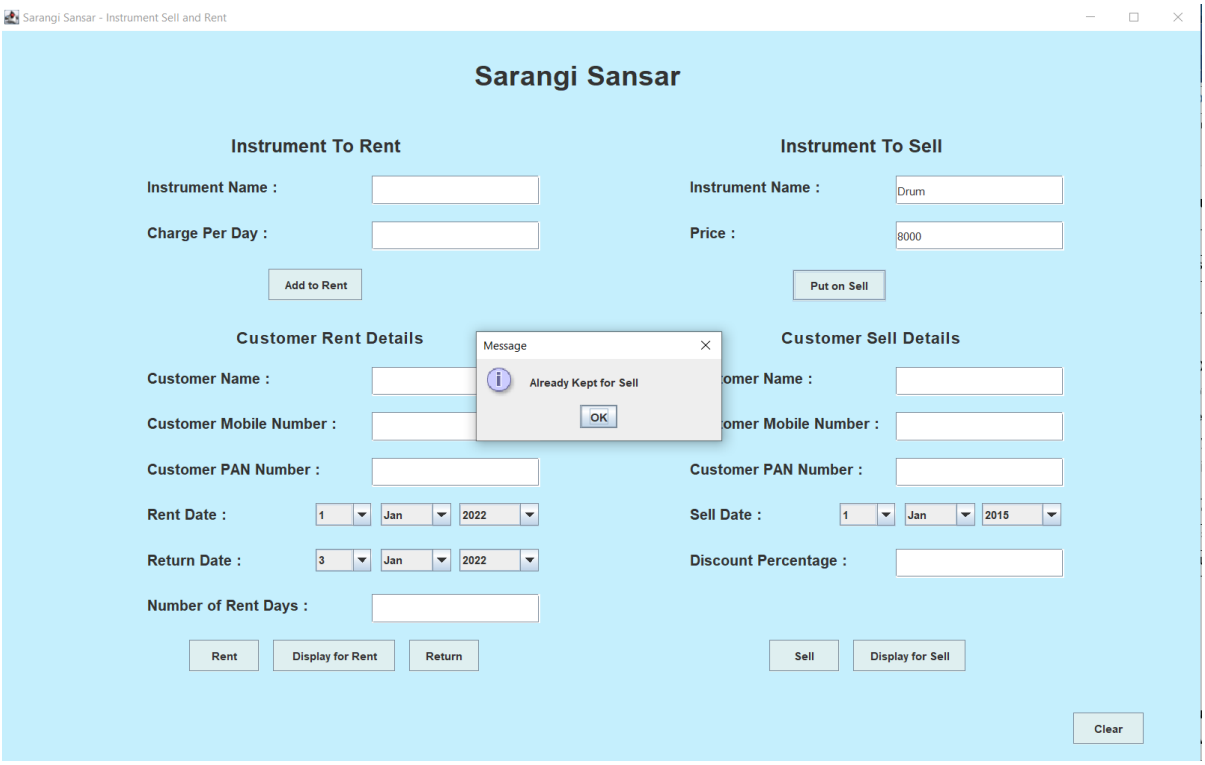

Figure 23: To add the same Instrument Twice in Instrument to Sell.

6. Error Detection

6.1 Syntax Error

Syntax error can be referred to as an error in the syntax during programming. Hence the small mistake made in the programming while writing the syntax code is called a Syntax Error. The code does not compile if the program detects a syntax error.

Error:

In the screenshot below, a semicolon (;) is missing which is preventing the program to compile.

```
public class SarangiSansar implements ActionListener
{
    JFrame frame;
    JButton rAdd, Rent, Return, Clear, sAdd, Sell, rDisplay, sDisplay;
    JTextField instNametxt, chargePerDaytxt, custNametxt, custMobtxt, cu
    JComboBox Combo1, Combo2, Combo3, Combo11, Combo22, Combo33, Combo11
    public SarangiSansar()
    {
        /*
        Assigning Frame
         */
        frame = new JFrame()
        frame.setVisible(tr
        frame.setSize(1300,
        frame.setLayout(null);
        frame.setTitle("Sarangi Sansar - Instrument Sell and Rent");
```

';' expected

Figure 24: Syntax Error

Solution

The error can be solved by adding a semicolon (;) at the end of the line which then compiles the code.

```
public class SarangiSansar implements ActionListener
{
    JFrame frame;
    JButton rAdd, Rent, Return, Clear, sAdd, Sell, rDisplay, sDisplay;
    JTextField instNametxt, chargePerDaytxt, custNametxt, custMobtxt,
    JComboBox Combo1, Combo2, Combo3, Combo11, Combo22, Combo33, Combo
    public SarangiSansar()
    {
        /*
        Assigning Frame
         */
        frame = new JFrame();
        frame.setVisible(true);
        frame.setSize(1300,850);
        frame.setLayout(null);
        frame.setTitle("Sarangi Sansar - Instrument Sell and Rent");
```

Figure 25: Correction for Syntax Error

6.2 Semantic Error

The semantic error can be referred as the error that happens while there is an interchange between data types for the given input. For example, a data which can only be stored in String data type is assigned to integer data type. Hence such errors are defined as semantic errors.

Error:

In the screenshot below, the values of the parameter passed are interchanged and the value of the parameter does not match the data type.

```
argePerDaytxt.getText().isEmpty())

 "The required details must be filled up."); //Displays the message if the TextField


etxt.getText();
seInt(chargePerDaytxt.getText());//getter methods

 Instrument_to_Ren(Charge_Per_Day, Instrument_Name);//Creating Object and Passing P
ct to the List
g(frame, instNamet t.getT                                                    ent added mes

eating Lopp
```

incompatible types: int cannot be converted to java.lang.String

Figure 26: Semantic error

Solution :

The error can be solved by interchanging the values of the parameter as the data type of the following parameter gets compatible.

```
etxt.getText();
seInt(chargePerDaytxt.getText());//getter methods

 Instrument_to_Rent(Instrument_Name, Charge_Per_Day) //Creating
ct to the List
g(frame, instNametxt.getText()+ " added to rent"); //Dispalying

eating Lopp
```

Figure 27: Correction for Semantic Error

6.3 Logical Error

Logical error is those error where the mistakes are made by the programmers while writing the logic of the program. Even the slightest of the mistake may cause this error to happen.

Error:

In the screenshot below, when the rent button is clicked, a message box saying it has been sold is displayed.

```
 Instrument_to_Rent(Instrument_Name, Charge_Per_Day);//Cre
ct to the List
g(frame, instNametxt.getText()+ " added to Sell"); //Dispa
```

Figure 28: Logical Error in Code

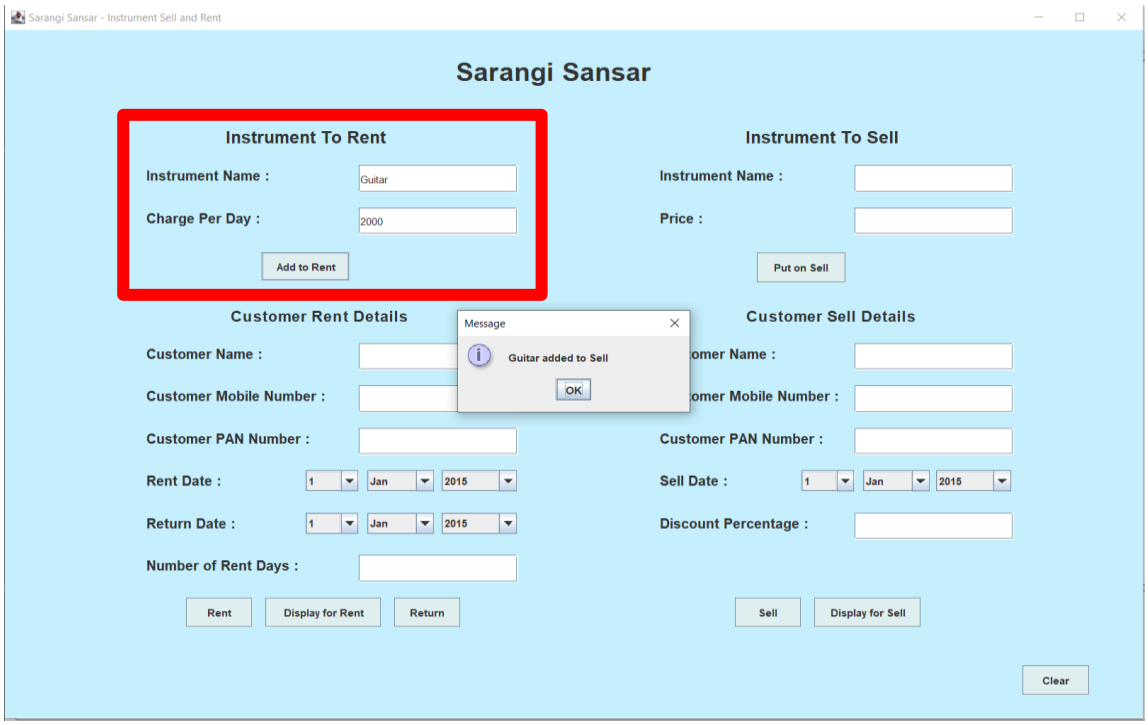

Figure 29: Logical Error

Solution:

Hence swapping back the value from “added to rent” to “added to sell” can solve this error.

```
jA= new Instrument_to_Rent(Instrument_Name, Charge_Per_Day);
ing Object to the List
geDialog(frame, instNametxt.getText()+ " added to rent"); //
```

Figure 30: Solution for Logical Error in Code

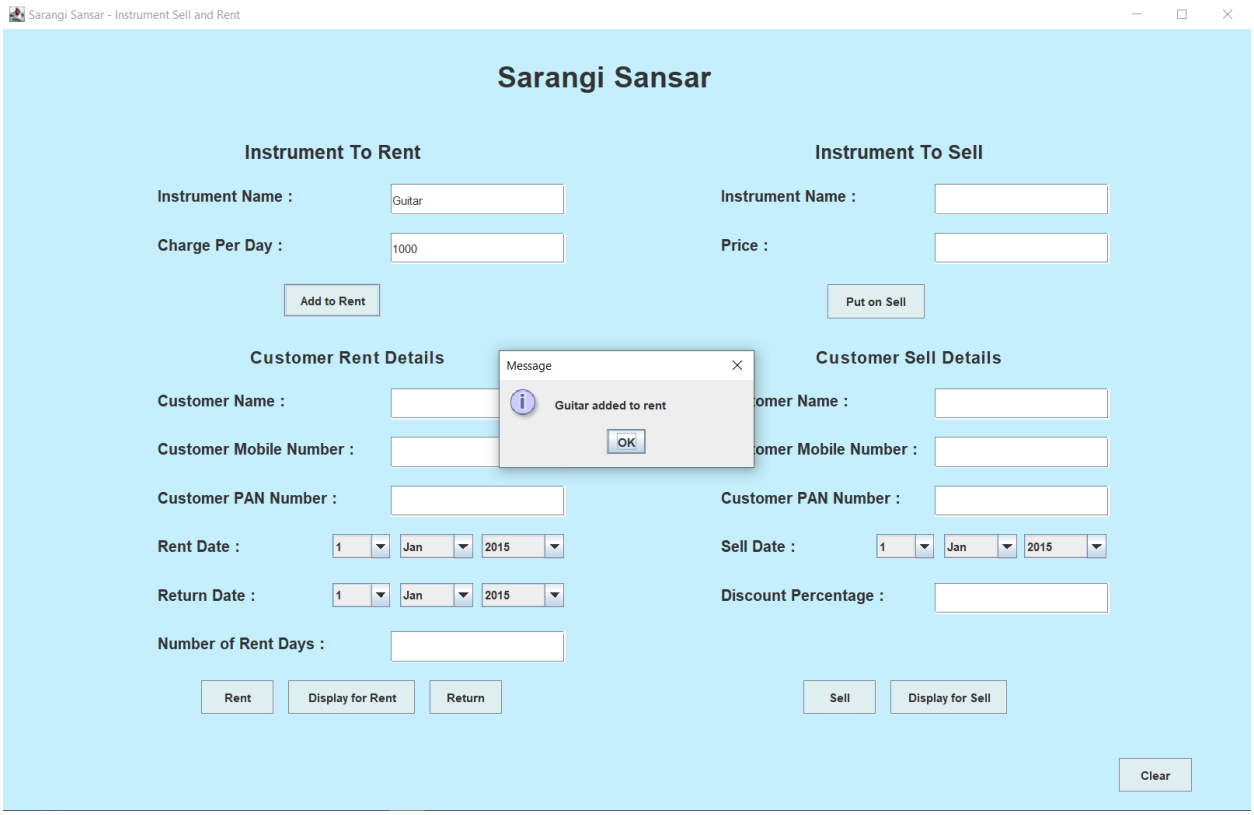

Figure 31: Solution for Logical Error

Conclusion

My understanding of Java programming and the practicality of its properties improved thanks to this assignment. It has helped me increase my knowledge about Java programming and Applicability of Graphical User Interface (GUI). After completion of this coursework, we found that developing a Graphical User Interface (GUI) in Java programming wasn't all that tough, and we became familiar with the different GUI components and their functioning. We got to know the concept of various function on how to use a GUI properly in Java.

We came across a number of difficulties while developing the GUI for the courses. Many problems are encountered while object-casting between various classes and few problems during description of methods. We found some errors while we did my testing. We did some research and we corrected my mistakes. In addition, the study materials provided by the lecturer and tutor greatly helped me a lot in gaining more knowledge and ideas on how to solve the problems and complete my coursework.

We learned many things such as abstraction, about JFrame, JPanel, JOptionPane, Event Handling, Exception Handling and Object-Casting of Java from this coursework. We also learned about the errors that may occur frequently in programming and which we were unaware of. We became familiar with various concepts relating GUI and Java and we hope that we did well in this coursework.

Appendix

Sarangi Sansar

```
import javax.swing.*;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.ArrayList;
public class SarangiSansar implements ActionListener
{
   JFrame frame;
   JLabel Title, Heading1, Heading2, subHeading1, subHeading2, instName,
chargePerDay, custName, custMob, custPAN, rentDate, returnDate, noOfRentDays,
sellName, price, scustName, scustMob, scustPAN, sellDate, discountPer;
   JButton rAdd, Rent, Return, Clear, sAdd, Sell, rDisplay, sDisplay;
   JTextField instNametxt, chargePerDaytxt, custNametxt, custMobtxt, custPANtxt,
noOfRentDaystxt, sellNametxt, pricetxt, scustNametxt, scustMobtxt, scustPANtxt,
discountPertxt ;
   JComboBox Combo1, Combo2, Combo3, Combo11, Combo22, Combo33,
Combo111, Combo222, Combo333;
   public SarangiSansar()
   {

      frame = new JFrame();
      frame.setVisible(true);
      frame.setSize(1300,850);
      frame.setLayout(null);
      frame.setTitle("Sarangi Sansar - Instrument Sell and Rent");

      Title = new JLabel("Sarangi Sansar");
      Heading1 = new JLabel("Instrument To Rent");
      Heading2 = new JLabel("Instrument To Sell");
      subHeading1 = new JLabel("Customer Rent Details");
      subHeading2 = new JLabel("Customer Sell Details");


      instName = new JLabel("Instrument Name :");
      chargePerDay = new JLabel("Charge Per Day :");
      custName= new JLabel("Customer Name :");
      custMob = new JLabel("Customer Mobile Number :");
      custPAN = new JLabel("Customer PAN Number :");
      rentDate = new JLabel("Rent Date :");
```

```
        returnDate = new JLabel("Return Date :");
        noOfRentDays = new JLabel("Number of Rent Days :");


        sellName = new JLabel("Instrument Name :");
        price = new JLabel("Price :");
        scustName= new JLabel("Customer Name :");
        scustMob = new JLabel("Customer Mobile Number :");
        scustPAN = new JLabel("Customer PAN Number :");
        sellDate = new JLabel("Sell Date :");
        discountPer = new JLabel("Discount Percentage :");


        instNametxt = new JTextField();
        chargePerDaytxt = new JTextField();
        custNametxt = new JTextField();
        custMobtxt = new JTextField();
        custPANtxt = new JTextField();
        noOfRentDaystxt = new JTextField();

        sellNametxt = new JTextField();
        pricetxt = new JTextField();
        scustNametxt = new JTextField();
        scustMobtxt = new JTextField();
        scustPANtxt = new JTextField();
        discountPertxt = new JTextField();


        String [] 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"};
        Combo1 = new JComboBox<String>(Day);


        String [] Date =
{"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"};
        Combo2 = new JComboBox<String>(Date);

        String [] Year = {"2015","2016","2017","2018","2019","2020","2021","2022"};
        Combo3 = new JComboBox<String>(Year);


        Combo11 = new JComboBox<String>(Day);//Combo Box for Day
        Combo22 = new JComboBox<String>(Date);//Combo Box for Months
        Combo33 = new JComboBox<String>(Year);//Combo Box for Year
```

```
Combo111 = new JComboBox<String>(Day);//Combo Box for Day
Combo222 = new JComboBox<String>(Date);//Combo Box for Months
Combo333 = new JComboBox<String>(Year);//Combo Box for Year



rAdd = new JButton("Add to Rent");
Rent = new JButton("Rent");
Return = new JButton("Return");
Clear = new JButton("Clear");
sAdd = new JButton("Put on Sell");
Sell = new JButton("Sell");
rDisplay = new JButton("Display for Rent");
sDisplay = new JButton("Display for Sell");



Color bgColor = new Color(197, 239, 253);
frame.getContentPane().setBackground(bgColor);//Changing BgColor of Frame's
Background

Color rAddBtn = new Color(223, 239, 240);
rAdd.setBackground(rAddBtn);//Changing BgColor of Add to Rent Button

Color RentBtn = new Color(223, 239, 240);
Rent.setBackground(RentBtn);//Changing BgColor of Rent Button

Color ReturnBtn = new Color(223, 239, 240);
Return.setBackground(ReturnBtn);//Changing BgColor of Rent Button

Color ClearBtn = new Color(223, 239, 240);
Clear.setBackground(ClearBtn);//Changing BgColor of Clear Button

Color sAddBtn = new Color(223, 239, 240);
sAdd.setBackground(sAddBtn);//Changing BgColor of Put on Sell Button

Color SellBtn = new Color(223, 239, 240);
Sell.setBackground(SellBtn);//Changing BgColor of Sell Button

Color rDisplayBtn = new Color(223, 239, 240);
rDisplay.setBackground(rDisplayBtn);//Changing BgColor of Display for Rent
Button

Color sDisplayBtn = new Color(223, 239, 240);
sDisplay.setBackground(sDisplayBtn);//Changing BgColor of Display for Sell
Button
```

```
Title.setFont(new Font("Helvetica BOLD", Font.BOLD, 30));
Heading1.setFont(new Font("Helvetica BOLD", Font.BOLD, 20));
Heading2.setFont(new Font("Helvetica BOLD", Font.BOLD, 20));
subHeading1.setFont(new Font("Helvetica", Font.BOLD, 18));
subHeading2.setFont(new Font("Helvetica", Font.BOLD, 18));

instName.setFont(new Font("Helvetica", Font.BOLD, 16));
chargePerDay.setFont(new Font("Helvetica", Font.BOLD, 16));
custName.setFont(new Font("Helvetica", Font.BOLD, 16));
custMob.setFont(new Font("Helvetica", Font.BOLD, 16));
custPAN.setFont(new Font("Helvetica", Font.BOLD, 16));
rentDate.setFont(new Font("Helvetica", Font.BOLD, 16));
returnDate.setFont(new Font("Helvetica", Font.BOLD, 16));
noOfRentDays.setFont(new Font("Helvetica", Font.BOLD, 16));

sellName.setFont(new Font("Helvetica", Font.BOLD, 16));
price.setFont(new Font("Helvetica", Font.BOLD, 16));
scustName.setFont(new Font("Helvetica", Font.BOLD, 16));
scustMob.setFont(new Font("Helvetica", Font.BOLD, 16));
scustPAN.setFont(new Font("Helvetica", Font.BOLD, 16));
sellDate.setFont(new Font("Helvetica", Font.BOLD, 16));
discountPer.setFont(new Font("Helvetica", Font.BOLD, 16));

chargePerDaytxt.addActionListener(this);

rAdd.addActionListener(this);
Rent.addActionListener(this);
Return.addActionListener(this);
Clear.addActionListener(this);
sAdd.addActionListener(this);
Sell.addActionListener(this);
rDisplay.addActionListener(this);
sDisplay.addActionListener(this);

frame.add(Title);
frame.add(Heading1);
frame.add(Heading2);
frame.add(subHeading1);
```

```
frame.add(subHeading2);


frame.add(instName);
frame.add(instNametxt);
frame.add(chargePerDay);
frame.add(chargePerDaytxt);
frame.add(rAdd);
frame.add(custName);
frame.add(custNametxt);
frame.add(custMob);
frame.add(custMobtxt);
frame.add(custPAN);
frame.add(custPANtxt);
frame.add(rentDate);
frame.add(returnDate);
frame.add(Combo1);
frame.add(Combo2);
frame.add(Combo3);
frame.add(noOfRentDays);
frame.add(Combo11);
frame.add(Combo22);
frame.add(Combo33);
frame.add(noOfRentDaystxt);
frame.add(Rent);
frame.add(rDisplay);
frame.add(Return);


frame.add(sellName);
frame.add(sellNametxt);
frame.add(price);
frame.add(pricetxt);
frame.add(sAdd);
frame.add(scustName);
frame.add(scustNametxt);
frame.add(scustMob);
frame.add(scustMobtxt);
frame.add(scustPAN);
frame.add(scustPANtxt);
frame.add(sellDate);
frame.add(Combo111);
frame.add(Combo222);
frame.add(Combo333);
frame.add(discountPer);
frame.add(discountPertxt);
```

```
frame.add(Sell);
frame.add(sDisplay);
frame.add(Clear);


Title.setBounds(510, 20, 329, 57);
Heading1.setBounds(250, 110, 235, 33);
Heading2.setBounds(837, 110, 225, 33);
subHeading1.setBounds(255, 320, 225, 34);
subHeading2.setBounds(838, 320, 225, 34);

instName.setBounds(160, 160, 180, 22);
instNametxt.setBounds(400, 160, 180, 32);
chargePerDay.setBounds(160, 210, 180, 22);
chargePerDaytxt.setBounds(400, 210, 180, 32);
rAdd.setBounds(290, 262, 100, 35);
Return.setBounds(340, 262, 75, 35);

custName.setBounds(160, 370, 180, 22);
custNametxt.setBounds(400, 370, 180, 32);
custMob.setBounds(160, 420, 240, 22);
custMobtxt.setBounds(400, 420, 180, 32);
custPAN.setBounds(160, 470, 240, 22);
custPANtxt.setBounds(400, 470, 180, 32);
rentDate.setBounds(160, 520, 180, 22);
Combo1.setBounds(340, 520, 60, 25);
Combo2.setBounds(410, 520, 75, 25);
Combo3.setBounds(494, 520, 85, 25);
returnDate.setBounds(160, 570, 180, 22);
Combo11.setBounds(340, 570, 60, 25);
Combo22.setBounds(410, 570, 75, 25);
Combo33.setBounds(494, 570, 85, 25);
noOfRentDays.setBounds(160, 620, 180,22);
noOfRentDaystxt.setBounds(400, 620, 180, 32);
Rent.setBounds(205, 670, 75, 35);
rDisplay.setBounds(295, 670 ,130, 35);
Return.setBounds(440, 670, 75, 35);


sellName.setBounds(740, 160, 180, 22);
sellNametxt.setBounds(960, 160, 180, 32);
price.setBounds(740, 210, 180, 22);
pricetxt.setBounds(960, 210, 180, 32);
sAdd.setBounds(850, 263, 100, 35);
```

```
        scustName.setBounds(740, 370, 180, 22);
        scustNametxt.setBounds(960, 370, 180, 32);
        scustMob.setBounds(740, 420, 240, 22);
        scustMobtxt.setBounds(960, 420, 180, 32);
        scustPAN.setBounds(740, 470, 240, 22);
        scustPANtxt.setBounds(960, 470, 180, 32);
        sellDate.setBounds(740, 520 ,180, 22);
        Combo111.setBounds(900, 520, 60, 25);
        Combo222.setBounds(970, 520, 75, 25);
        Combo333.setBounds(1052, 520, 85, 25);
        discountPer.setBounds(740, 570, 180, 22);
        discountPertxt.setBounds(960, 570, 180, 32);
        Sell.setBounds(825, 670, 75, 35);
        sDisplay.setBounds(915, 670 ,120, 35);
        Clear.setBounds(1150, 750, 75, 35);
    }


    ArrayList<Instrument> List = new ArrayList<Instrument>();

    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource() == rAdd)
        {
            boolean ItemAdd = false; exectuion...

            if(instNametxt.getText().isEmpty() || chargePerDaytxt.getText().isEmpty())
            {
                JOptionPane.showMessageDialog(frame, "The required details must be filled
up.");
            }
            else
            {

                try
                {
                    String Instrument_Name = instNametxt.getText();
                    int Charge_Per_Day = Integer.parseInt(chargePerDaytxt.getText());
methods

                    if(List.isEmpty())
                    {
                        Instrument_to_Rent ObjA= new Instrument_to_Rent(Instrument_Name,
Charge_Per_Day);
                        List.add(ObjA);
```

```
                    JOptionPane.showMessageDialog(frame, instNametxt.getText()+ "
added to rent");
                }
                else
                {
                    for(Instrument i : List)//Creating Lopp
                    {
                        if(i instanceof Instrument_to_Rent)
                        {
                            Instrument_to_Rent ObjAdd = (Instrument_to_Rent) i;
                            if(i.getInstrument_Name().equals(instNametxt.getText()))
                            {
                                ItemAdd = true;
                                JOptionPane.showMessageDialog(frame, "Already Added to
rent");
                                break;
                            }
                        }
                    }
                    if(ItemAdd == false)
                    {
                        Instrument_to_Rent ObjA = new
Instrument_to_Rent(Instrument_Name, Charge_Per_Day);
                        List.add(ObjA);
                        JOptionPane.showMessageDialog(frame, instNametxt.getText()+ "
added to rent");
                    }
                }
            }

            catch(NumberFormatException ex)
            {
                JOptionPane.showMessageDialog(frame,"Invalid Input. Please enter in
numbers");
            }
        }
    }


    if(e.getSource() == Rent)
    {
        boolean ItemRent = false;
        if(custNametxt.getText().isEmpty() || custMobtxt.getText().isEmpty() ||
custPANtxt.getText().isEmpty() || noOfRentDaystxt.getText().isEmpty())
        {
            JOptionPane.showMessageDialog(frame, "Enter all the details.");
```

```
        }
        else
        {
            for(Instrument i : List)
            {
                if(i instanceof Instrument_to_Rent)
                {
                    Instrument_to_Rent ObjR = (Instrument_to_Rent) i;
                    if((i.getInstrument_Name()).equals(instNametxt.getText()))
                    {
                        ItemRent = true;
                    }
                }
            }
            try
            {
                if(ItemRent == false)
                {
                    JOptionPane.showMessageDialog(frame, "Add an Instrument First");
//Dispalying That the list is empty
                }
                else
                {
                    String Customer_Name = custNametxt.getText();
                    String Customer_Mobile_Number = custMobtxt.getText();
                    int Customer_PAN = Integer.parseInt(custPANtxt.getText());
                    String Date_of_Rent = Combo1.getSelectedItem()+"-
"+Combo2.getSelectedItem()+"-"+Combo3.getSelectedItem();
                    String Date_of_Return = Combo11.getSelectedItem()+"-
"+Combo22.getSelectedItem()+"-"+Combo33.getSelectedItem();
                    int Number_of_days = Integer.parseInt(noOfRentDaystxt.getText());
                    for(Instrument i : List)
                    {
                        if(i instanceof Instrument_to_Rent)//checking object
                        {
                            if((i.getInstrument_Name()).equals(instNametxt.getText()))
                            {
                                Instrument_to_Rent ObjR  = (Instrument_to_Rent)i;
//Downcasting
                                if(ObjR.getIs_rented() == false)
                                {
                                    ObjR.Instrument_Rent(Customer_Name,
Customer_Mobile_Number, Customer_PAN, Date_of_Rent, Date_of_Return,
Number_of_days);
```

```
                                JOptionPane.showMessageDialog(frame,
ObjR.getInstrument_Name()+" is rented successfully");
                            }
                            else
                            {
                                //process to show that the item is already rented by another
customer
                                JOptionPane.showMessageDialog(frame, "The instrument has
been rented by:" + "\n" + "\n"+
                                "Customer Name : "+ObjR.getCustomer_Name()+ "\n"+
                                "Customer Mobile Number :
"+ObjR.getCustomer_Mobile_Number()+ "\n"+
                                "Return Date: "+ObjR.getDate_of_Return());
                            }
                        }
                    }
                }
            }
        }
        catch(NumberFormatException ex)//exception handaling
        {
            JOptionPane.showMessageDialog(frame,"Invalid Input. Please enter in
numbers");
        }
    }
}


if(e.getSource() == Return)
{
    {
    if((instNametxt.getText()).isEmpty())
    {
        JOptionPane.showMessageDialog(frame, "Add an Instrument Name First.");
    }
    else
    {
        for(Instrument i : List)
        {
            String Instrument_Name = instNametxt.getText();
            if(i instanceof Instrument_to_Rent)
            {
                if((i.getInstrument_Name()).equals(instNametxt.getText()))
                {
                    Instrument_to_Rent ObjRe = (Instrument_to_Rent) i;
```

```
                    if(ObjRe.getIs_rented() == false)
                    {
                        JOptionPane.showMessageDialog(frame, "The instrument '"+
ObjRe.getInstrument_Name()+"' is not yet rented.");
                    }
                    else
                    {
                        ObjRe.Return();
                        JOptionPane.showMessageDialog(frame, "The instrument
'"+ObjRe.getInstrument_Name()+"' is returned successfully!");
                    }
                }
            }
        }
    }
    }

    if(e.getSource() == Clear)
    {
        instNametxt.setText("");
        chargePerDaytxt.setText("");
        custNametxt.setText("");
        custMobtxt.setText("");
        custPANtxt.setText("");
        noOfRentDaystxt.setText("");
        sellNametxt.setText("");
        pricetxt.setText("");
        scustNametxt.setText("");
        scustMobtxt.setText("");
        scustPANtxt.setText("");
        discountPertxt.setText("");
    }

    if(e.getSource() == sAdd)
    {
        boolean ItemASell = false; exectuion...

        /*
         Checking if the TextFields are filled or not
         */
        if(sellNametxt.getText().isEmpty() || pricetxt.getText().isEmpty())
        {
            JOptionPane.showMessageDialog(frame, "The required details must be filled
up."); //Displays the message if the TextFields are left empty
        }
```

```
        else
        {
            try
            {
                String Instrument_Name = sellNametxt.getText();
                double Price = Double.parseDouble(pricetxt.getText());//getter methods

                if(List.isEmpty())
                {
                    Instrument_to_Sell ObjS= new Instrument_to_Sell(Instrument_Name,
Price);//Creating Object and Passing Parameters to the Object
                    List.add(ObjS);//Adding Object to the List
                    JOptionPane.showMessageDialog(frame, sellNametxt.getText()+ " kept
for Sell"); //Dispalying Insturment added message
                }
                else
                {
                    for(Instrument i : List)
                    {
                        if(i instanceof Instrument_to_Sell)
                        {
                            Instrument_to_Sell ObjSell = (Instrument_to_Sell) i;
                            if(i.getInstrument_Name().equals(sellNametxt.getText()))
                            {
                                ItemASell = true;
                                JOptionPane.showMessageDialog(frame, "Already Kept for
Sell");
                                break;
                            }
                        }
                    }
                    if(ItemASell == false)
                    {
                        Instrument_to_Sell ObjS = new Instrument_to_Sell(Instrument_Name,
Price);
                        List.add(ObjS);
                        JOptionPane.showMessageDialog(frame, sellNametxt.getText()+"
added to Sell");
                    }
                }
            }

            catch(NumberFormatException ex)
            {
                JOptionPane.showMessageDialog(frame,"Invalid Input. Please enter in
numbers");
```

```
            }
        }
    }

    if(e.getSource() == Sell)
    {
        boolean ItemSell = false;

        /*
         Checking if the TextFields are filled or not
         */
        if((scustNametxt.getText()).isEmpty() || (scustMobtxt.getText()).isEmpty() ||
(scustPANtxt.getText()).isEmpty() || (discountPertxt.getText()).isEmpty())
        {
            JOptionPane.showMessageDialog(frame, "Enter all the details.");
        }
        else
        {
            for(Instrument i : List)
            {
                if(i instanceof Instrument_to_Sell)
                {
                    Instrument_to_Sell SoldObj = (Instrument_to_Sell) i;
                    if((i.getInstrument_Name()).equals(sellNametxt.getText()))
                    {
                        ItemSell = true;
                    }
                }
            }
            try
            {
                if(ItemSell == false)
                {
                    JOptionPane.showMessageDialog(frame, "Add a Selling Instrument
First"); //Dispalying That the list is empty
                }
                else
                {
                    String Customer_Name = scustNametxt.getText();
                    String Customer_Mobile_Number = scustMobtxt.getText();
                    int Customer_PAN = Integer.parseInt(scustPANtxt.getText());
                    String Sell_date = Combo111.getSelectedItem()+"-
"+Combo222.getSelectedItem()+"-"+Combo333.getSelectedItem();
                    double discount_percent =
Double.parseDouble(discountPertxt.getText());
                    for(Instrument i : List)
```

```
                    {
                        if(i instanceof Instrument_to_Sell)
                        {

                            if((i.getInstrument_Name()).equals(sellNametxt.getText()))
                            {
                                Instrument_to_Sell SoldObj  = (Instrument_to_Sell)i;
                                if(SoldObj.getIs_Sold() == false)
                                {
                                    SoldObj.Instrument_Sell(Customer_Name,
Customer_Mobile_Number, Customer_PAN,Sell_date, discount_percent);

                                    JOptionPane.showMessageDialog(frame,
SoldObj.getInstrument_Name()+" has been sold sucessfully");
                                }
                                else
                                {
                                    JOptionPane.showMessageDialog(frame, "The instrument has
been sold to:" + "\n"+ "\n"+
                                    "Customer Name : " + SoldObj.getCustomer_Name()+ "\n"+
                                    "Customer Mobile Number : " +
SoldObj.getCustomer_Mobile_Number()+ "\n"+
                                    "Customer PAN Number : " + SoldObj.getCustomer_PAN());
                                }
                            }
                        }
                    }
                }
            }
            catch(NumberFormatException ex)
            {
                JOptionPane.showMessageDialog(frame, "Invalid Input. Please enter in
numbers");
            }
        }
    }

    //for dispaly button
    if(e.getSource() == rDisplay)
    {
        boolean disR = false;
        if((instNametxt.getText()).isEmpty())
        {
            JOptionPane.showMessageDialog(frame, "Please enter the Instrument
name.");
        }
```

```
            else
            {
                for(Instrument i : List)
                {
                    if(i instanceof Instrument_to_Rent)
                    {
                        Instrument_to_Rent ObjR = (Instrument_to_Rent) i;
                        if((ObjR.getInstrument_Name()).equals(instNametxt.getText()))
                        {
                            disR = true;
                        }
                    }
                }
                if(disR == true)
                {
                    String Instrument_Name = instNametxt.getText();
                    for(Instrument i : List)
                    {
                        if(i instanceof Instrument_to_Rent)
                        {
                            if((i.getInstrument_Name()).equals(instNametxt.getText()))
                            {
                                Instrument_to_Rent rDisplay = (Instrument_to_Rent)i;
                                rDisplay.display();
                            }
                        }
                    }
                }
                else
                {
                    JOptionPane.showMessageDialog(frame, instNametxt.getText()+"' is not
added");
                }
            }
        }

         if(e.getSource() == sDisplay)
        {
            boolean disS = false;
            if((sellNametxt.getText()).isEmpty())
            {
                JOptionPane.showMessageDialog(frame, "Please enter an Selling Instrument
name.");
            }
            else
            {
```

```
            for(Instrument i : List)
            {
                if(i instanceof Instrument_to_Sell)
                {
                    Instrument_to_Sell ObjR = (Instrument_to_Sell) i;
                    if((ObjR.getInstrument_Name()).equals(sellNametxt.getText()))
                    {
                        disS = true;
                    }
                }
            }
            if(disS == true)
            {
                String Instrument_Name = sellNametxt.getText();
                for(Instrument i : List)
                {
                    if(i instanceof Instrument_to_Sell)
                    {
                        if((i.getInstrument_Name()).equals(sellNametxt.getText()))
                        {
                            Instrument_to_Sell sDisplay = (Instrument_to_Sell)i;
                            sDisplay.display();
                        }
                    }
                }
            }
            else
            {
                JOptionPane.showMessageDialog(frame, sellNametxt.getText()+"' is not
added on sell");
            }
        }
    }
  }
  public static void main(String[]args)
  {
    new SarangiSansar();
  }
}
```

Bibliography

Anon., n.d. *ComputerHope.* [Online]

Available at: https://www.computerhope.com/jargon/m/microsoft-word.htm

Anon., n.d. *AboutDraw.io.* [Online]

Available at: https://www.computerhope.com/jargon/d/drawio.htm

Anon., n.d. *Java Editor BlueJ.* [Online]

Available at: https://www.bluej.org/about.html

Anon., n.d. *Visual Paradigm.* [Online]

Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/