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 Spring – 2

Student Name: Aayush Man Tuladhar

London Met ID: 22015636

College ID: NP01CP4S220066

Assignment Due Date: 5th Aug 2022

Assignment Submission Date: 5th Aug 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

1.1. Objective of the Project

The following project is related to the previous projected submitted. The main objective of the coursework is to create a GUI for the previously done coursework on booking Autorickshaw's and purchasing, selling Electric Scooters.

The GUI is to be designed using the Java framework. It can be designed any way you want to as long as it follows the guidelines of the coursework. The system is to store the details of the vehicles in an Array List and call it for the different functions of the program. It needs to contain mean method to run the java file through the help of the command prompt. All the functionalities should be tested, and all the buttons must show the appropriate messages as per the event the occur.

1.2. Applications Used

Multiple tools were used for the completion of this coursework. Some of these tools are as follows:

BlueJ:

The following application is a integrate development environment or IDE for short for a programming language called JAVA. It is free Java programming and development tool for beginners, and it is a visually resounding application.

Draw.io:

The following is application that is used to create any sort of diagram be it flowchart or an entity relational diagram. For this project though the following program was specifically used to create the class diagram of the program.

Ms-Word:

The following is the application used for writing and documentation of the report about the objective and how it was completed and if the requirements were fulfilled. Microsoft Word or Ms-word for short is a writing application that can help with multiple disciplines of studies and research. It is diverse tool used around the world for many reasons.

2. Class Diagram

2.1. For Super-Class Vehicle

The following is class diagram showcases the attributes, constructors, and methods of the super class Vehicle. It records the common attributes found in all vehicles.

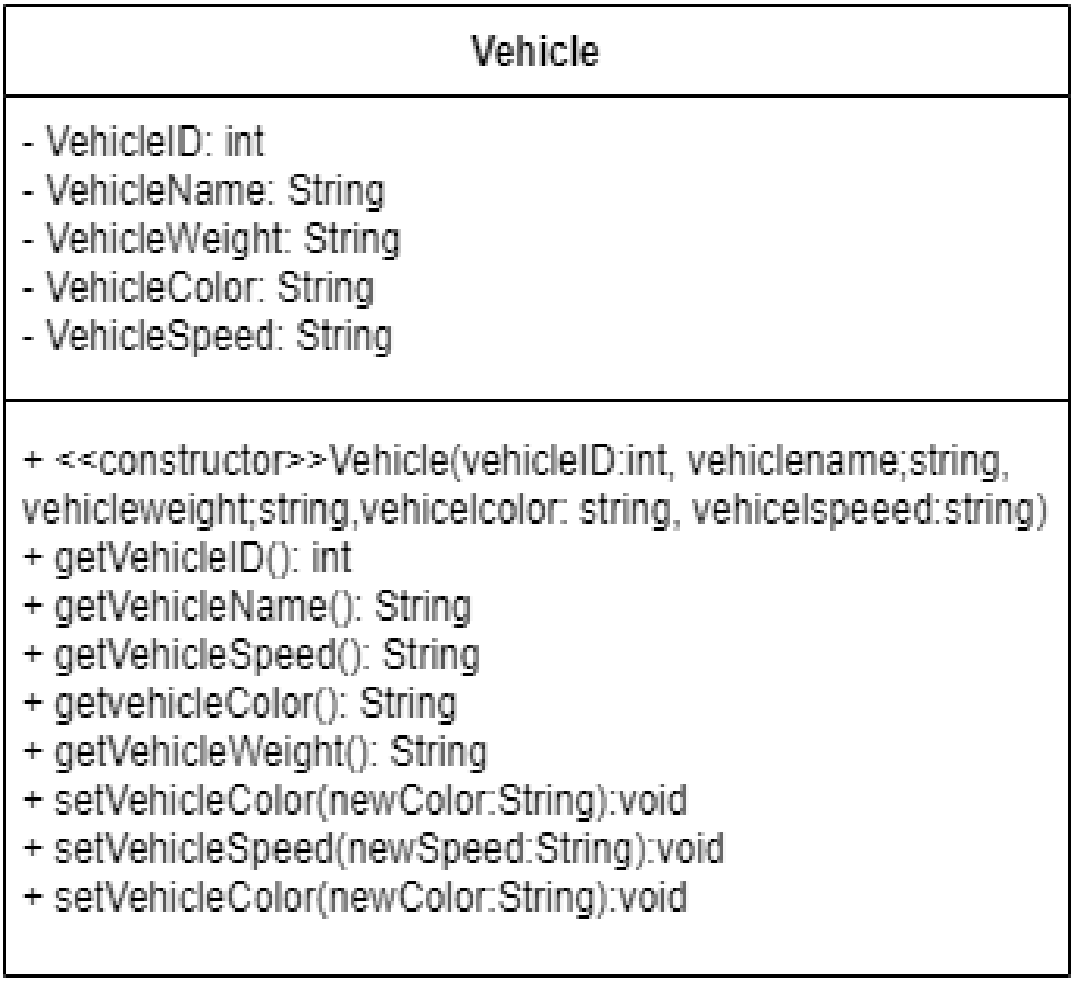

Figure 1: Super-Class Vehicle Class Diagram

2.2. For Sub-Class Autorickshaw

The following is the class diagram of the sub-class Autorickshaw and showcases the attributes, constructor and methods used in it. It has a method to book the rickshaw if needed.

Figure 2: Sub-class Autorickshaw Class Diagram

2.3. For Sub-Class Electric Scooter

The following is the class diagram of the sub-class Electric Scooter and showcases the attributes, constructor and methods used in it. With this the user can purchase or sell the vehicle as they want.

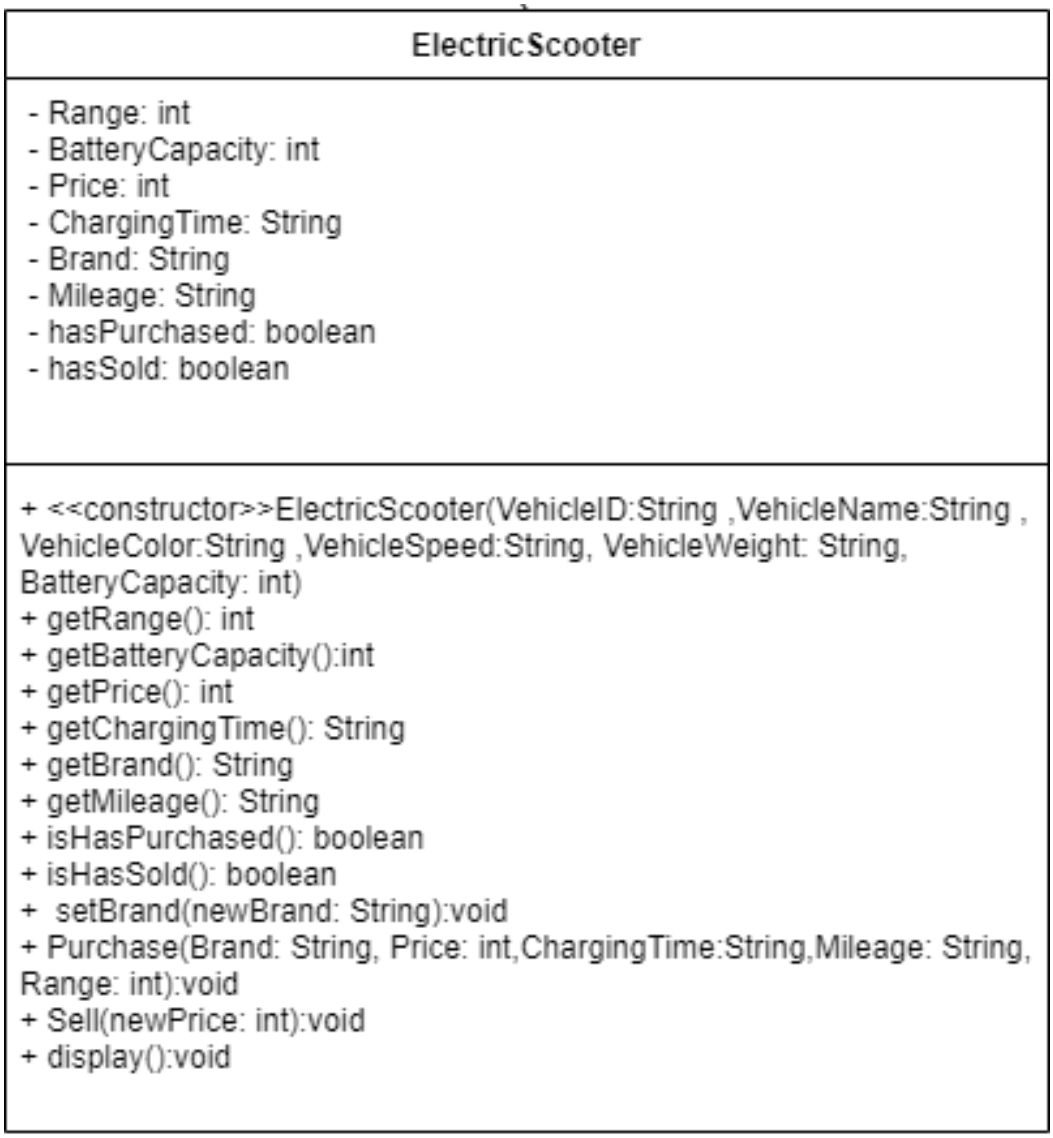

Figure 3: Sub-class Electric Scooter Class Diagram

2.4. For Class TransportGUI

The following is the class diagram of Transport GUI and showcases all the attributes, constructors and methods used in it. It has methods to add, book, purchase and sell Auto Rickshaw and Electric Scooter respectively.

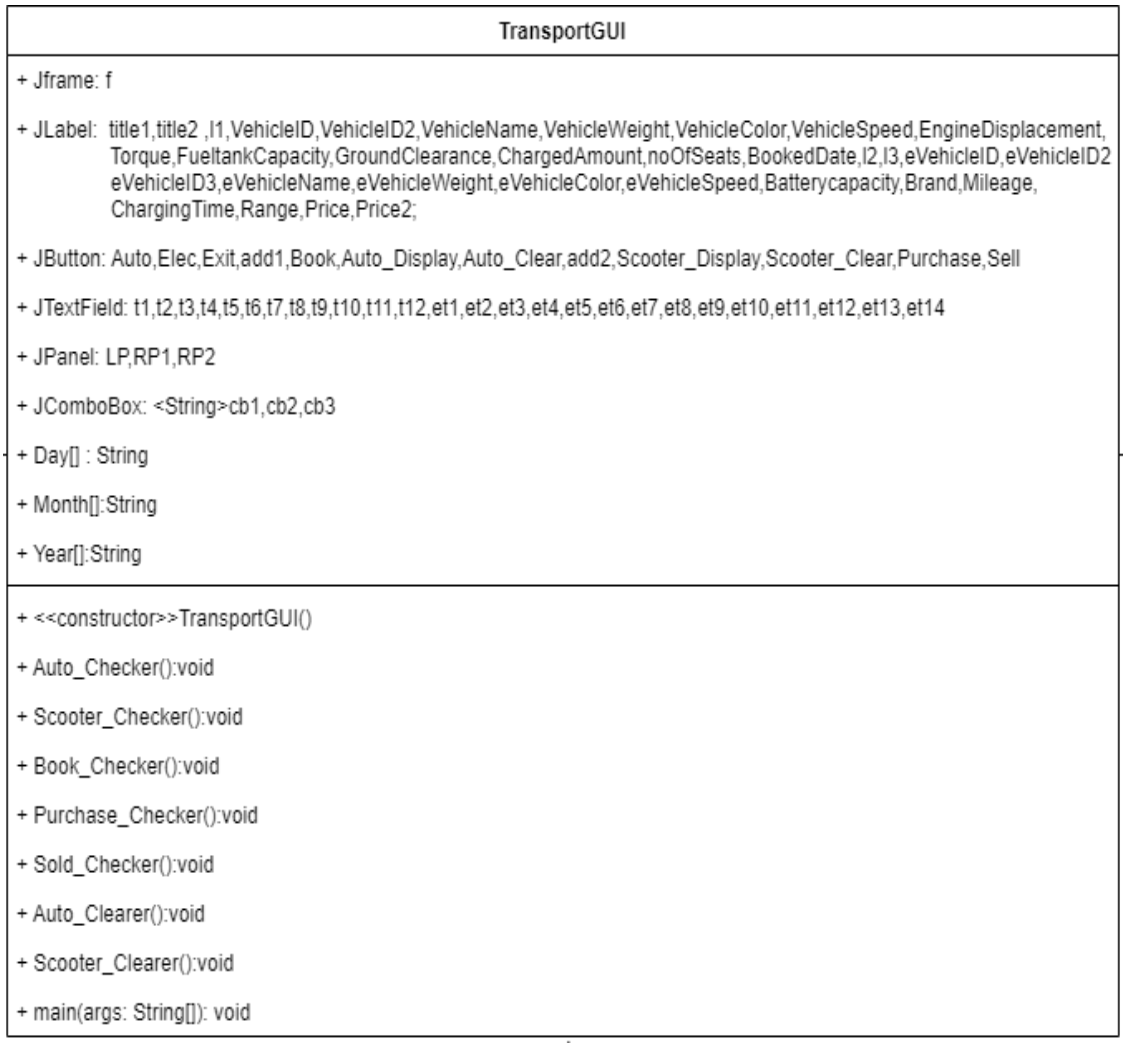

Figure 4: Class Diagram of TransportGUI

2.5. Relation Between all the Classes

The following diagram shows the relationship between all the classes that were mentioned above.

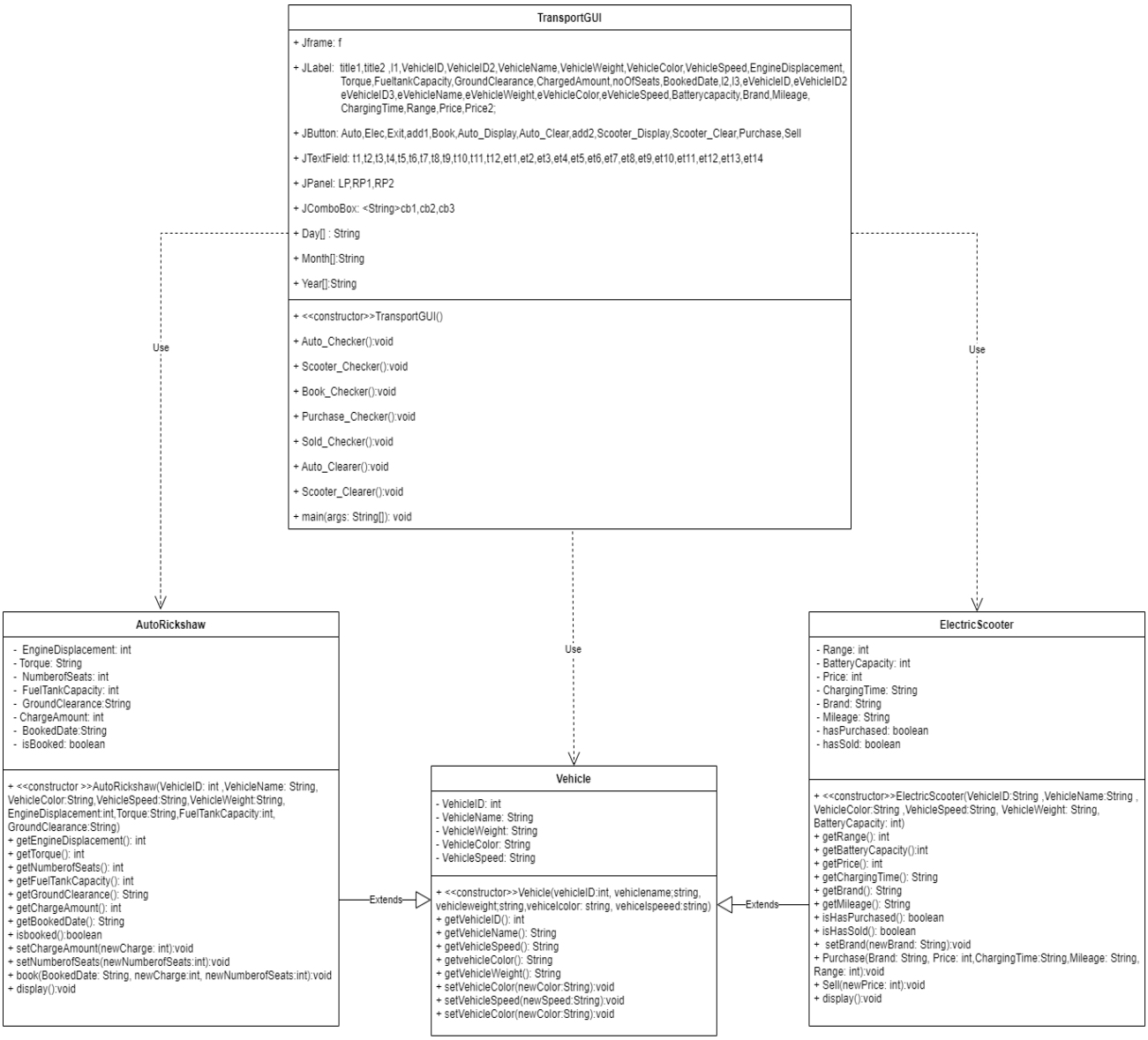

Figure 5:Relation between all the Classes

3. Pseudocode

3.1. Pseudocode for TransportGUI

```
IMPORT all components from javax.swing

IMPORT all components from java.awt

IMPORT all components from java.awt.event

IMPORT all components from java.util


CREATE a class TransportGUI that implements ActionListener

DO

    DECLARE JFrame called f

    DECLARE JPanel called LP, RP1, RP2

    DECLARE JLabel called title1, title2, l1, VehicleID, VehicleID2, VehicleName,
    VehicleWeight, VehicleColor ,VehicleSpeed, EngineDisplacement, Torque,
    FueltankCapacity, GroundClearance ,ChargedAmount ,noOfSeats ,BookedDate
    ,l2 ,l3 ,eVehicleID ,eVehicleID2 ,eVehicleID3 ,eVehicleName ,eVehicleWeight
    ,eVehicleColor ,eVehicleSpeed ,Batterycapacity ,Brand ,Mileage ,ChargingTime
    ,Range ,Price , Price2

    DECLARE JTextField t1 ,t2 ,t3 ,t4 ,t5 ,t6 ,t7 ,t8 ,t9 ,t1 0,t 11 ,t12 ,et1 ,et2 ,et3 ,et4
    ,et5 ,et6,et7,et8,et9,et10,et11,et12,et13,et14

    DECLARE JComboBox cb1, cb2, cb3

    DELCARE JButton Auto ,Elec ,Exit ,add1 ,Book ,Auto_Display ,Auto_Clear ,add2
    ,Scooter_Display ,Scooter_Clear,Purchase,Sell

    DECLARE Color LC, RC1, RC2

    DECLARE String Day, Month, Year

    CREATE an Array List from Vehicle called Vehicle

    CREATE a constructor TransportGUI where no parameters were taken

    DO

        CREATE a new frame named f

        CREATE a new Panel named LP

        CREATE a new Panel named RP1
```

CREATE a new Panel named RP2

CREATE a new JLabel named title1

CREATE a new JLabel named VehicleID

CREATE a new JLabel named VehicleName

CREATE a new JLabel named VehicleWeight

CREATE a new JLabel named VehicleColor

CREATE a new JLabel named VehicleSpeed

CREATE a new JLabel named Engine Displacement

CREATE a new JLabel named Torque

CREATE a new JLabel named Fuel Tank Capacity

CREATE a new JLabel named Ground Clearance

CREATE a new JLabel named l1

CREATE a new JLabel named VehicleID2

CREATE a new JLabel named Charged Amount

CREATE a new JLabel named No. of Seats

CREATE a new JLabel named Booked Date

CREATE a new JTextField name t1

CREATE a new JTextField name t2

CREATE a new JTextField name t3

CREATE a new JTextField name t4

CREATE a new JTextField name t5

CREATE a new JTextField name t6

CREATE a new JTextField name t7

CREATE a new JTextField name t8

CREATE a new JTextField name t9

CREATE a new JTextField name t10

CREATE a new JTextField name t11

CREATE a new JTextField name t12

CREATE a new JButton named add1

SET background color of add1 as LC

CREATE a new JButton named Book

SET background color of Book as LC

CREATE a new JButton named Auto_Display

SET background color of Auto_Display as LC

CREATE a new JButton named Auto_Clear

SET background color of Auto_Clear as LC

CREATE a new JComboBox named cb1 and Put in the values

SET background color of cb1 as LC

CREATE a new JComboBox named cb2 and Put in the values

SET background color of cb2 as LC

CREATE a new JComboBox named cb3 and Put in the values

SET background color of cb3 as LC

CREATE a new JLabel named title2

CREATE a new JLabel named eVehicleID

CREATE a new JLabel named eVehicleName

CREATE a new JLabel named eVehicleWeight

CREATE a new JLabel named eVehicleColor

CREATE a new JLabel named eVehicleSpeed

CREATE a new JLabel named BatteryCapacity

CREATE a new JLabel named l2

CREATE a new JLabel named eVehicleID2

CREATE a new JLabel named Brand

CREATE a new JLabel named ChargingTime

CREATE a new JLabel named Range

CREATE a new JLabel named Mileage

CREATE a new JLabel named Price

CREATE a new JLabel named l3

CREATE a new JLabel named eVehicleID3

CREATE a new JLabel named Price2

CREATE a new JTextField name et1

CREATE a new JTextField name et2

CREATE a new JTextField name et3

CREATE a new JTextField name et4

CREATE a new JTextField name et5

CREATE a new JTextField name et6

CREATE a new JTextField name et7

CREATE a new JTextField name et8

CREATE a new JTextField name et9

CREATE a new JTextField name et10

CREATE a new JTextField name et11

CREATE a new JTextField name et12

CREATE a new JTextField name et13

CREATE a new JTextField name et14

CREATE a new JButton named add2

SET background color of add2 as LC

CREATE a new JButton named Purchase

SET background color of Purchase as LC

CREATE a new JButton named Sell

SET background color of Sell as LC

CREATE a new JButton name Scooter_Display

SET background color of Scooter_Display as LC

CREATE a new JButton named Scooter_Clear

SET background color of Scooter_Clear as LC

CREATE a new JButton named Auto

SET background color of Auto as RC1

CREATE a new JButton named Elec

SET background color of Elec as RC1

CREATE a new JButton named Exit

SET background color of Exit as RC1

SET coordinates, height, and width of title1

SET fonts for the Label title1

SET coordinates, height, and width of Vehicle ID

SET coordinates, height, and width of VehicleName

SET coordinates, height, and width of VehicleWeight

SET coordinates, height, and width of VehicleColor

SET coordinates, height, and width of VehicleSpeed

SET coordinates, height, and width of EngineDisplacement

SET coordinates, height, and width of Torque

SET coordinates, height, and width of FuelTankCapacity

SET coordinates, height, and width of GroundClearance

SET coordinates, height, and width of l1

SET fonts for the Label l1

SET coordinates, height, and width of VehicleID2

SET coordinates, height, and width of ChargedAmount

SET coordinates, height, and width of noOfSeats

SET coordinates, height, and width of BookedDate

SET coordinates, height, and width of t1

SET coordinates, height, and width of t2

SET coordinates, height, and width of t3

SET coordinates, height, and width of t4

SET coordinates, height, and width of t5

SET coordinates, height, and width of t6

SET coordinates, height, and width of t7

SET coordinates, height, and width of t8

SET coordinates, height, and width of t9

SET coordinates, height, and width of t10

SET coordinates, height, and width of t11

SET coordinates, height, and width of t12

SET coordinates, height, and width of add1

SET coordinates, height, and width of cb1

SET coordinates, height, and width of cb2

SET coordinates, height, and width of cb3

SET coordinates, height, and width of Book

SET coordinates, height, and width of Auto_Display

SET coordinates, height, and width of Auto_Clear

SET coordinates, height, and width of title2

SET fonts for the Label title2

SET coordinates, height, and width of eVehicle ID

SET coordinates, height, and width of eVehicleName

SET coordinates, height, and width of eVehicleWeight

SET coordinates, height, and width of eVehicleColor

SET coordinates, height, and width of eVehicleSpeed

SET coordinates, height, and width of et1

SET coordinates, height, and width of et2

SET coordinates, height, and width of et3

SET coordinates, height, and width of et4

SET coordinates, height, and width of et5

SET coordinates, height, and width of et6

SET coordinates, height, and width of et7

SET coordinates, height, and width of et8

SET coordinates, height, and width of et9

SET coordinates, height, and width of et10

SET coordinates, height, and width of et11

SET coordinates, height, and width of et12

SET coordinates, height, and width of et13

SET coordinates, height, and width of et14

SET coordinates, height, and width of l2

SET fonts for the Label l2

SET coordinates, height, and width of add2

SET coordinates, height, and width of l3

SET fonts for the Label l3

SET coordinates, height, and width of eVehicleID2

SET coordinates, height, and width of Brand

SET coordinates, height, and width of Price

SET coordinates, height, and width of Range

SET coordinates, height, and width of ChargingTime

SET coordinates, height, and width of Mileage

SET coordinates, height, and width of eVehicleID3

SET coordinates, height, and width of Price2

SET coordinates, height, and width of Purchase

SET coordinates, height, and width of Sell

SET coordinates, height, and width of Scooter_Clear

SET coordinates, height, and width of LP

SET Background color for the LP with LC

SET coordinates, height, and width of RP1

SET Background color for the RP1 with RC1

SET coordinates, height, and width of RP2

SET Background color for the RP2 with RC2

SET coordinates, height, and width of Auto

SET coordinates, height, and width of Elec

SET coordinates, height, and width of Exit

ADD Elec to the ActionListener

ADD Auto to the ActionListener

ADD Exit to the ActionListener

ADD Auto_Clear to the ActionListener

ADD Scooter_Clear to the ActionListener

ADD add1 to the ActionListener

ADD Book to the ActionListener

ADD add2 to the ActionListener

ADD Purchase to the ActionListener

ADD Sell to the ActionListener

ADD Auto_Display to the ActionListener

ADD Scooter_Display to the ActionListener

SET the layout to null

SET the visibility to true

SET the size of the frame

SET the Relative location to null

ADD LP to the frame f

ADD RP1 to the frame f

ADD RP2 to the frame f

SET the RP panel Layout to null

ADD title1 to the panel RP1

ADD VehicleID to the panel RP1

ADD VehicleName to the panel RP1

ADD VehicleWeight to the panel RP1

ADD VehicleColor to the panel RP1

ADD VehicleSpeed to the panel RP1

ADD EngineDisplacement to the panel RP1

ADD torque to the panel RP1

ADD FuelTankCapacity to the panel RP1

ADD Ground Clearance to the panel RP1

ADD l1 to the panel RP1

ADD VehicleID2 to the panel RP1

ADD ChargedAmount to the panel RP1

ADD noOfSeats to the panel RP1

ADD BookedDate to the panel RP1

ADD t1 to the panel RP1

ADD t2 to the panel RP1

ADD t3 to the panel RP1

ADD t4 to the panel RP1

ADD t5 to the panel RP1

ADD t6 to the panel RP1

ADD t7 to the panel RP1

ADD t8 to the panel RP1

ADD t9 to the panel RP1

ADD t10 to the panel RP1

ADD t11 to the panel RP1

ADD t12 to the panel RP1

ADD add1 to the panel RP1

ADD cb1 to the panel RP1

ADD cb2 to the panel RP1

ADD cb3 to the panel RP1

ADD Auto_Display to the panel RP1

ADD Auto_Clear to the panel RP1

ADD Book to the panel RP1

SET the RP2 panel Layout to null

ADD title2 to the panel RP2

ADD eVehicleID to the panel RP2

ADD eVehicleName to the panel RP2

ADD eVehicleWeight to the panel RP2

ADD eVehicleColor to the panel RP2

ADD eVehicleSpeed to the panel RP2

ADD BatteryCapacity to the panel RP2

ADD et1 to the panel RP2

ADD et2 to the panel RP2

ADD et3 to the panel RP2

```
ADD et4 to the panel RP2
ADD et5 to the panel RP2
ADD et6 to the panel RP2
ADD et7 to the panel RP2
ADD et8 to the panel RP2
ADD et9 to the panel RP2
ADD et10 to the panel RP2
ADD et11 to the panel RP2
ADD et12 to the panel RP2
ADD et13 to the panel RP2
ADD et14 to the panel RP2
ADD eVehicleID2 to the panel RP2
ADD eVehicleID3 to the panel RP2
ADD Range to the panel RP2
ADD Brand to the panel RP2
ADD Price to the panel RP2
ADD Mileage to the panel RP2
ADD ChargingTime to the panel RP2
ADD Price2 to the panel RP2
ADD Purchase to the panel RP2
ADD Sell to the panel RP2
ADD Scooter_Display to the panel RP2
ADD Scooter_Clear to the panel RP2

SET layout of panel LP to null
ADD Auto to panel LP
ADD Elec to panel LP
ADD Exit to panel LP
```

```
END DO

CREATE a method Auto_Checker with no parameters return type void
DO
    GET text from the Text Field t1
    GET text from the Text Field t2
    GET text from the Text Field t3
    GET text from the Text Field t4
    GET text from the Text Field t5
    GET text from the Text Field t6
    GET text from the Text Field t7
    GET text from the Text Field t8
    GET text from the Text Field t9

    DECLARE Boolean added and set as false
    CREATE object named auto from the class Autorickshaw

    FOR each object in the Vehicle Array List
        IF object is the instance of Autorickshaw and If the vehicleID matches
            SET Boolean added to true
        END IF
     IF Boolean added is set as true
        DISPLAY the dialog box “Your AutoRickshaw has already been added."
     ELSE

        DISPLAY the dialog box “Your AutoRickshaw has been added."
      END IF
END DO
```

```
CREATE a method Scooter_Checker with no parameters return type void
DO
    GET text from the Text Field et1
    GET text from the Text Field et2
    GET text from the Text Field et3
    GET text from the Text Field et4
    GET text from the Text Field et5
    GET text from the Text Field et6

    DECLARE Boolean added and set as false
    CREATE object named auto from the class ElectricScooter

    FOR each object in the Vehicle Array List
        IF object is the instance of ElectricScooter and If the vehicleID matches
            SET Boolean added to true
        END IF
     IF Boolean added is set as true
        DISPLAY the dialog box “Your Electric Scooter has already been added."
     ELSE

        DISPLAY the dialog box “Your Electric Scooter has been added."
    END IF
END DO
```

CREATE a method Auto_Clearer with no parameters return type void

DO

SET text of Text Field t1 to empty

SET text of Text Field t2 to empty

SET text of Text Field t3 to empty

SET text of Text Field t4 to empty

SET text of Text Field t5 to empty

SET text of Text Field t6 to empty

SET text of Text Field t7 to empty

SET text of Text Field t8 to empty

SET text of Text Field t9 to empty

SET text of Text Field t10 to empty

SET text of Text Field t11 to empty

SET text of Text Field t12 to empty

SET ComboBox cb1 to 1

SET ComboBox cb2 to January

SET ComboBox cb3 to 2022

END DO

CREATE a method Scooter_Clearer with no parameters return type void

DO

SET text of Text Field et1 to empty

SET text of Text Field et2 to empty

SET text of Text Field et3 to empty

SET text of Text Field et4 to empty

SET text of Text Field et5 to empty

SET text of Text Field et6 to empty

SET text of Text Field et7 to empty

SET text of Text Field et8 to empty

SET text of Text Field et9 to empty

SET text of Text Field et10 to empty

SET text of Text Field et11 to empty

SET text of Text Field et12 to empty

SET text of Text Field et13 to empty

SET text of Text Field et14 to empty

END DO

```
CREATE a method Book_Checker with no parameters return type void
DO
    GET text from the Text Field t10
    GET text from the Text Field t11
    GET text from the Text Field t12
    GET selected items from the cb1, cb2 and cb3
    DECLARE Boolean available as true
    DECLARE Boolean exist as true

    FOR each object in Vehicle Array List
        IF vehicle ID is the same and is instance of AutoRickshaw
            Downcast the object to AutoRickshaw class
            IF the isBooked value is false
                CALL the method from AutoRickshaw to book the AutoRickshaw
                Display message dialog box" The Autorickshaw has been booked"
                SET available value as true
            ELSE
                Display message dialog box" The Autorickshaw has already been booked"
                SET available value as true
            END IF
            BREAK

        ELSE IF vehicleID is not Equal or not instance of AutoRickshaw
            SET value of exist as False
            SET value of Available as False
        END IF
    END FOR
```

```
    IF exist equals to false and available equals to false
        Display message dialog box" The Autorickshaw doesn't exist."
    ELSE IF vehicle array list is empty
         Display message dialog box" The Autorickshaw doesn't exist."
     END IF
END DO

CREATE a method Purchase_Checker with no parameters return type void
DO
    GET text from the Text Field et7
    GET text from the Text Field et8
    GET text from the Text Field et9
    GET text from the Text Field et10
    GET text from the Text Field et11
    GET text from the Text Field et12
    DECLARE Boolean available as true
    DECLARE Boolean exist as true

    FOR each object in Vehicle Array List
        IF vehicle ID is the same and is instance of ElectricScooter
            Downcast the object to ElectricScooter class
            IF the hasPurchased value is false
                CALL the method from ElectricScooter to purchase the ElectricScooter
                Display message dialog box" The Scooter has been Purchased"
                SET available value as true
            ELSE
                Display message dialog box" The Scooter has already been Purchased"
```

```
                SET available value as true
            END IF
            BREAK

        ELSE IF vehicleID is not Equal or not instance of ElectricScooter
            SET value of exist as False
            SET value of Available as False
        END IF
    END FOR

    IF exist equals to false and available equals to false
        Display message dialog box" The ElectricScooter doesn't exist."
    ELSE IF vehicle array list is empty
        Display message dialog box" The ElectricScooter doesn't exist."
    END IF
END DO

CREATE a method Purchase_Checker with no parameters return type void
DO
    GET text from the Text Field et13
    GET text from the Text Field et14
    DECLARE Boolean available as true
    DECLARE Boolean exist as true

    FOR each object in Vehicle Array List
        IF vehicle ID is the same and is instance of ElectricScooter
            Downcast the object to ElectricScooter class
            IF the hasPurchased value is true and hasSold is false
```

```
                CALL the method from ElectricScooter to purchase the ElectricScooter
                Display message dialog box" The Scooter has been Sold."
                SET available value as true
            ELSE IF hasPurchased is false and hasSold is fasle
              Display message dialog box" The Scooter has to br Purchase before Selling"
                SET available value as true
            ELSE
                Display message dialog box" The Scooter has already been Purchased"
                SET available value as true
            END IF
            BREAK

        ELSE IF vehicleID is not Equal or not instance of ElectricScooter
            SET value of exist as False
            SET value of Available as False
        END IF
    END FOR

    IF exist equals to false and available equals to false
        Display message dialog box" The ElectricScooter doesn't exist."
    ELSE IF vehicle array list is empty
          Display message dialog box" The ElectricScooter doesn't exist."
     END IF
END DO
```

CREATE a method ActionPerformed to Handle the Events

```
DO
    IF add1 button is pressed
      TRY
          IF the text fields are empty
             Display message dialog box" The Text Fields are Empty!!"
          ELSE
             CALL the Auto_Checker method
          END IF
      END TRY
      CATCH NumberFormatException
            Display message dialog box" Re-Enter the values again"
      CATCH Exception
            Display message dialog box" Re-Enter the values again"
      END CATCH
   END IF

  IF add2 button is pressed
      TRY
          IF the text fields are empty
             Display message dialog box" The Text Fields are Empty!!"
          ELSE
             CALL the Scooter_Checker method
          END IF
      END TRY
      CATCH NumberFormatException
            Display message dialog box" Re-Enter the values again"
```

```
        CATCH Exception
                Display message dialog box" Re-Enter the values again"
        END CATCH
    END IF

    IF Book button is pressed
        TRY
            IF the text fields are empty
                Display message dialog box" The Text Fields are Empty!!"
            ELSE
                CALL the Book_Checker method
            END IF
        END TRY
        CATCH NumberFormatException
                Display message dialog box" Re-Enter the values again"
        CATCH Exception
                Display message dialog box" Re-Enter the values again"
        END CATCH
    END IF

    IF Purchase button is pressed
        TRY
            IF the text fields are empty
                Display message dialog box" The Text Fields are Empty!!"
            ELSE
                CALL the Purchase_Checker method
            END IF
        END TRY
```

```
    CATCH NumberFormatException
        Display message dialog box" Re-Enter the values again"
    CATCH Exception
        Display message dialog box" Re-Enter the values again"
    END CATCH
END IF

IF Sell button is pressed
    TRY
        IF the text fields are empty
            Display message dialog box" The Text Fields are Empty!!"
        ELSE
            CALL the Sold_Checker method
        END IF
    END TRY
    CATCH NumberFormatException
        Display message dialog box" Re-Enter the values again"
    CATCH Exception
        Display message dialog box" Re-Enter the values again"
    END CATCH
END IF

IF Auto_Display Button is pressed
    FOR each object in the vehicle Array List
        IF object is the instance of Autorickshaw
            Downcast the object to AutoRickshaw
            CALL the display method from the class
        END IF
```

```
      END FOR

  IF Scooter_Display Button is pressed
        FOR each object in the vehicle Array List
            IF object is the instance of ElectricScooter
                Downcast the object to ElectricScooter
                 CALL the display method from the class
            END IF
       END FOR

 IF Elec button is pressed
     SET RP1 visibility to false
     SET RP2 visibility to true
     CALL the method Auto_Clearer

   IF Auto button is pressed
     SET RP1 visibility to true
     SET RP2 visibility to false
     CALL the method Scooter_Clearer

  IF the Auto_Clear button is pressed
    CALL the method Auto_Clearer
  IF the Scooter_Clear button is pressed
    CALL the method Scooter_Clearer
  IF Exit button is pressed
    CALL System.exit(0) to close the program completely
   END IF
END DO
```

4. Methods in TransportGUI

- **public void Auto_Checker**():

  The following method is used to add and re-check the object created and added into the Vehicle array list. This is for the objects of Auto Rickshaws only. This method allows the object to be added only once which reduces the redundancies in the list. It checks if the object with the same Vehicle ID already exists or not and display the appropriate message.

- **public void Scooter_Checker():**

  The following method is used to add and re-check the object created and added into the Vehicle array list. This for the objects of Electric Scooter only. This method allows the object to be added only once which reduces the redundancies in the list. It checks if the object with the same Vehicle ID already exists or not and display the appropriate message.

- **public void Book_Checker():**

  The following method id used for the function of the book button. This method first verifies the existence of the Vehicle by comparing the vehicle ID of the objects in list and the vehicle ID entered in the Text-Field. It also checks if it is the instance of AutoRickshaw and downcast the object so we can get methods from the class AutoRickshaw. If the vehicle exists, it books the vehicle using the Book method form the class of AutoRickshaw. It also doesn't allow the same vehicle to be booked multiple times and will display the appropriate message to inform the user.

- **public void Purchase_Checker():**

  The following method is used for the function of the buy button. This method too first verifies the existence of the vehicle by comparing the vehicle ID of the objects in list and vehicle ID entered in the Text-Field. It also checks if it is the instance of ElectricScooter and downcast the object so we can get methods from the class ElectricScooter. If all the parameters are entered correctly the vehicle can be purchased and the appropriate message will be displayed. The purchasing is carried out by calling the Purchase method form the class ElectricScooter. This method allows the vehicle to be purchased only once as the same vehicle can not be purchased multiple times.

- **public void Sold_Checker():**

  The following method is used for the function of the sell button. This method also first verifies the existence of the vehicle by comparing the vehicle ID of the objects in list and vehicle ID entered in the Text-Field. It also checks if it is the instance of ElectricScooter and downcast the object so we can get methods from the class ElectricScooter. It also allows you to sell the vehicle only if it has been purchased first, if not purchased first it will display the appropriate message. If all the conditions meet the sell method from the Electric Scooter class is called and used to sell the scooter. It allows the scooter to sold only once unless it is purchased again.

- **public void Auto_Clearer():**

  The following method is used for the function of the clearing the Text-Fields. It helps you clear all the Text-Fields once you have filled all the parameters in the GUI. This method is specifically for the AutoRickshaw panel in this GUI.

- **public void Scooter_Clearer():**

  The following method is used for the function of the clearing the Text-Fields. It helps you clear all the Text-Fields once you have filled all the parameters in the GUI. This method is specifically for the Electric-Scooter panel in this GUI.

- **public static void main(String args[]):**

  The following is the main method that helps to run the TransportGUI program without creating an instance of it. It is needed for the program to be run through the help of command prompt.

- **public void actionPerformed(ActionEvent e):**

  The following method is used to provide functionality to the buttons that are present in the GUI.

- **Auto-Rickshaw Add Button:**

  The following button is used to add the Autorickshaw object and make it an instance of Autorickshaw. If the values in the text fields are not entered the appropriate dialog box is shown to the user. When the parameters are met correctly it calls upon the Auto_Checker method to add the object or if already added to display the appropriate dialog box is displayed. If any exceptions such as NumberFormatException a caught an appropriate error box is shown guiding the user to perform the task correctly.

- **Electric-Scooter Add Button:**

  The following button is used to add the object of Electric scooter to the vehicle array list. If the values in the text fields are not entered the appropriate dialog box is shown to the user. When the parameters are met correctly it calls upon the Scooter_Checker method to add the object or if already added to display the appropriate dialog box is displayed. If any exceptions such as NumberFormatException a caught an appropriate error box is shown guiding the user to perform the task correctly.

- **Book Button:**

  The following button is used for the booking of the Autorickshaw. It first checks the existence of the vehicle comparing the vehicle Ids the are given. If the parameter is not satisfied it will give an appropriate dialog box if not it will call upon the method Book_Checker for the booking of the vehicle. If any exceptions such as NumberFormatException a caught an appropriate error box is shown guiding the user to perform the task correctly.

- **Purchase Button:**

  The following button is used for the purchasing of the Electric Scooter. It first checks the existence of the vehicle comparing the vehicle Ids the are given. If the parameter is not satisfied it will give an appropriate dialog box if not it will call upon the method Purchase_Checker for the purchasing of the vehicle. If any exceptions such as NumberFormatException a caught an appropriate error box is shown guiding the user to perform the task correctly.

- **Sell Button:**

  The following button is used for the selling of the Electric Scooter. It first checks the existence of the vehicle comparing the vehicle Ids the are given. If the parameter is not satisfied it will give an appropriate dialog box if not it will call upon the method Sold_Checker for the purchasing of the vehicle. The vehicle cannot be sold until it has been purchased. If any exceptions such as NumberFormatException a caught an appropriate error box is shown guiding the user to perform the task correctly.

- **Display Button:**

  The following button calls upon the display method respective to the panels of the GUI. This displays all the information that has been collected by the program while it was running.

- **Clear Button:**

  The following button calls upon the Auto_Clearer or the Scooter_Clearer method respective of the panel that the button has been used. This button clears all the text fields in the GUI.

- **Exit Button:**

  The following button exits the program entirely.

- **Autorickshaw Button:**

  The following button closes the Electric scooter panel and opens the Autorickshaw panel. It also clears the panel it changes from.

- **Electric Scooter Button:**

  The following button closes the Electric scooter panel and opens the Autorickshaw panel. It also clears the panel it changes from.

5. Testing the GUI

5.1. Test 1

Test no.	1
Objective	Running and compiling the GUI through command prompt
Action	• Typing cmd onto the path where the files exist. • Typing javac filename.java to complie. • Typing java filename to run the file.
Expected Result	The program should compile without error and run smoothly without any errors.
Actual Result	The program compiled without any error and ran smoothly without any error.
Conclusion	Objective completed.

Table 1: Test no. 1

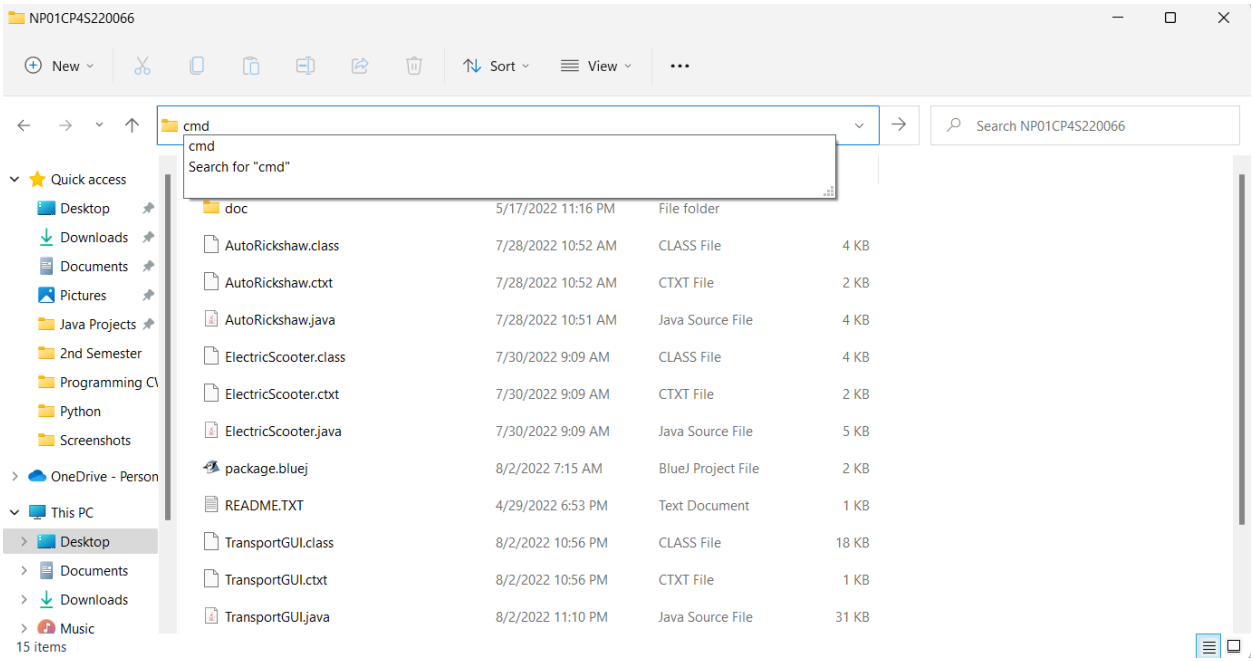

Figure 6:Opening Command Prompt for directory

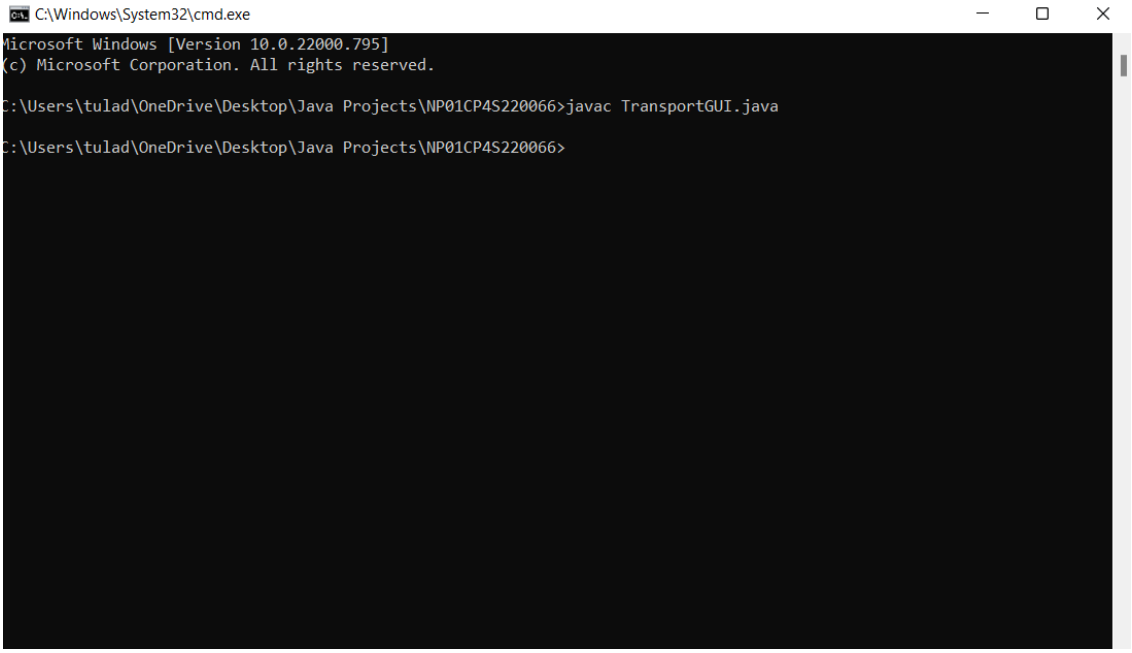

Figure 8:Compiling the GUI in cmd

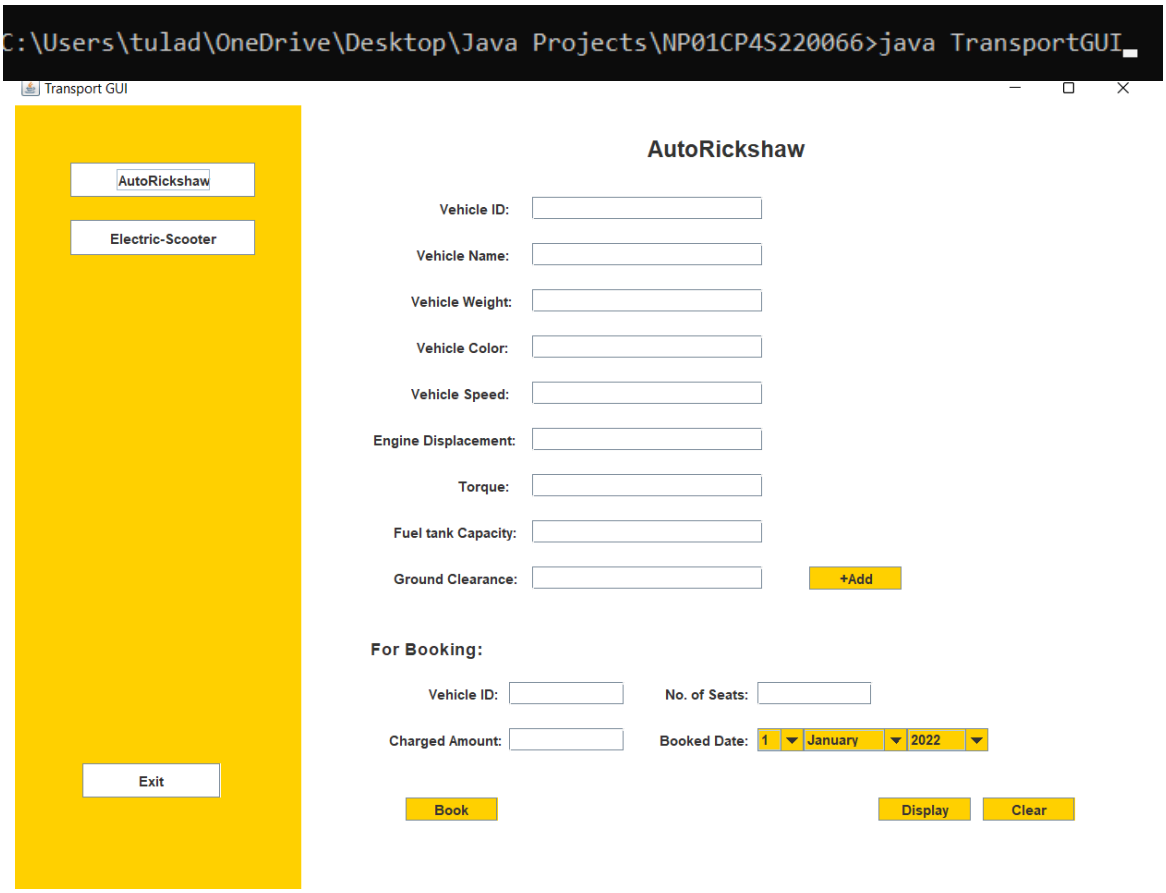

Figure 7: Running the GUI in cmd

5.2. Test 2

Test no.	2
Objective	a) Add Auto-Rickshaw b) Add Electric Scooter c) Book Auto-Rickshaw d) Purchase Electric Scooter e) Sell Electric Scooter
Action	• Enter all the parameters and add both electric scooter and autorickshaw by using the button. • Book the autorickshaw using the book button. • Purchase the electric scooter using the buy button. • Sell the electric scooter using the sell button.
Expected Result	Appropriate dialog boxes should be displayed respective to the actions performed.
Actual Result	Appropriate dialog boxes were displayed respective of the actions performed.
Conclusion	Objective completed

Table 2: Test no. 2

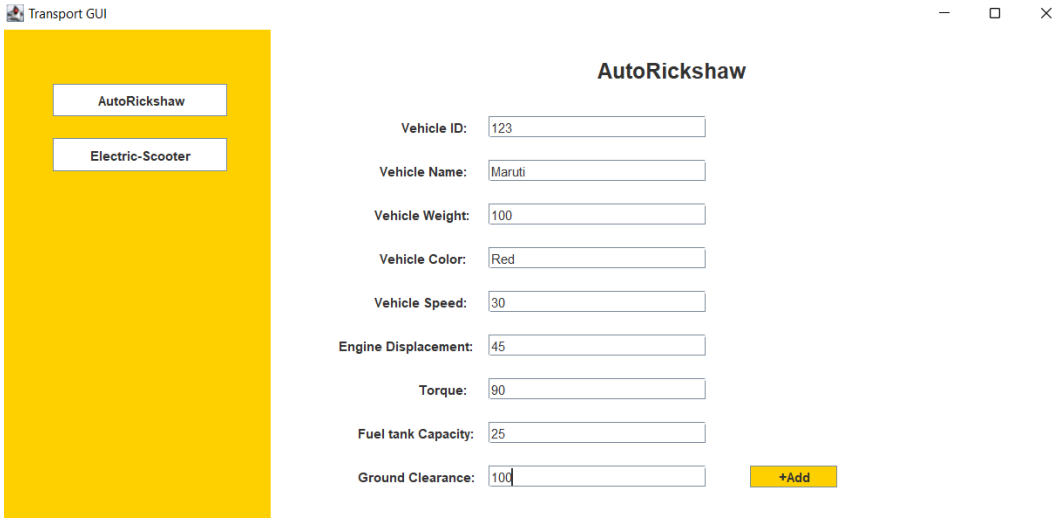

Figure 9:Entering values in the Text Fields

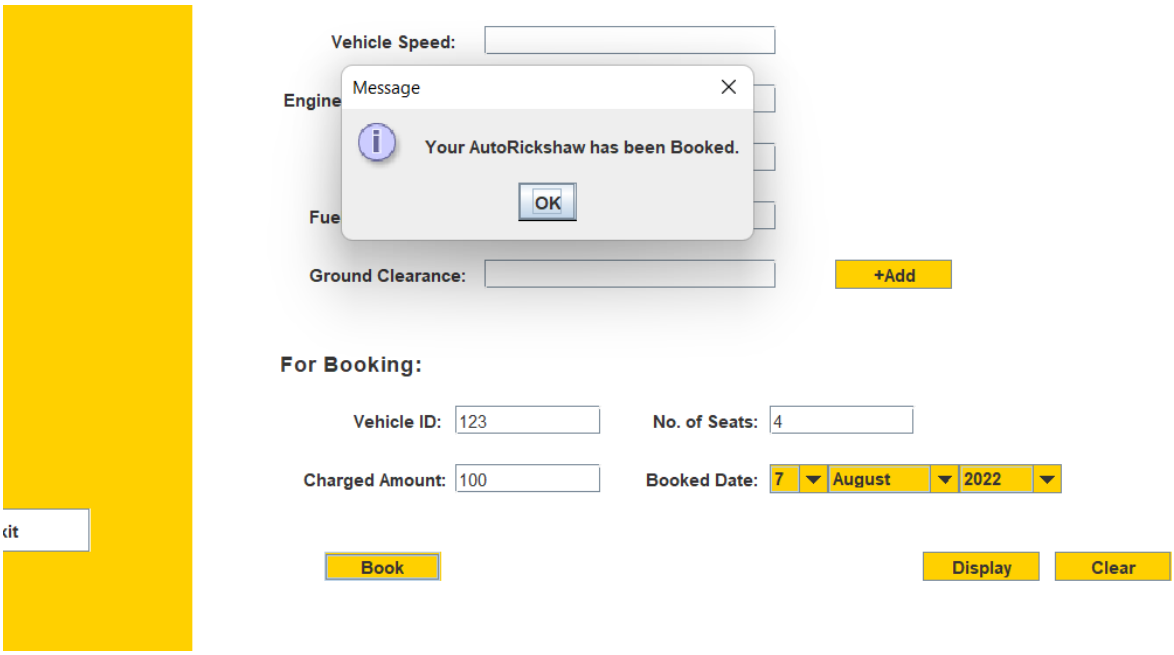

Figure 10: Adding an Autorickshaw successfully

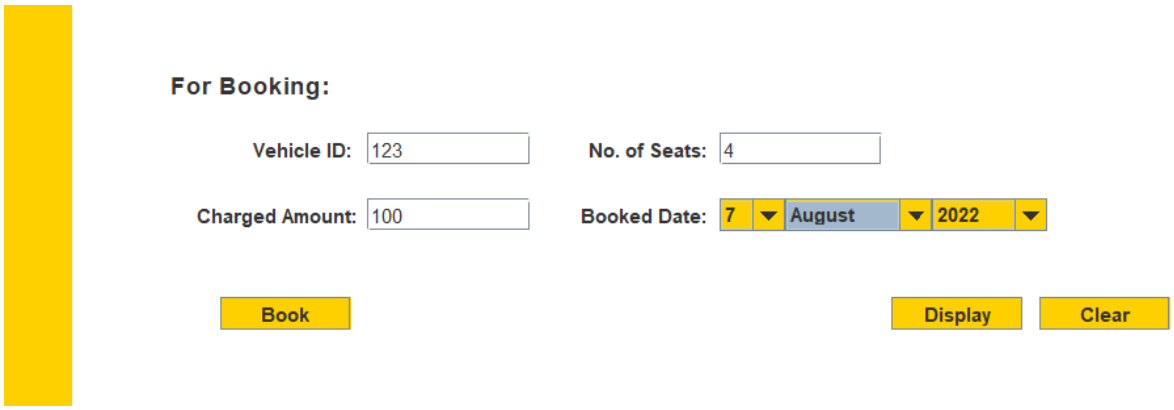

Figure 11:Entering values in the Text Fields for Booking

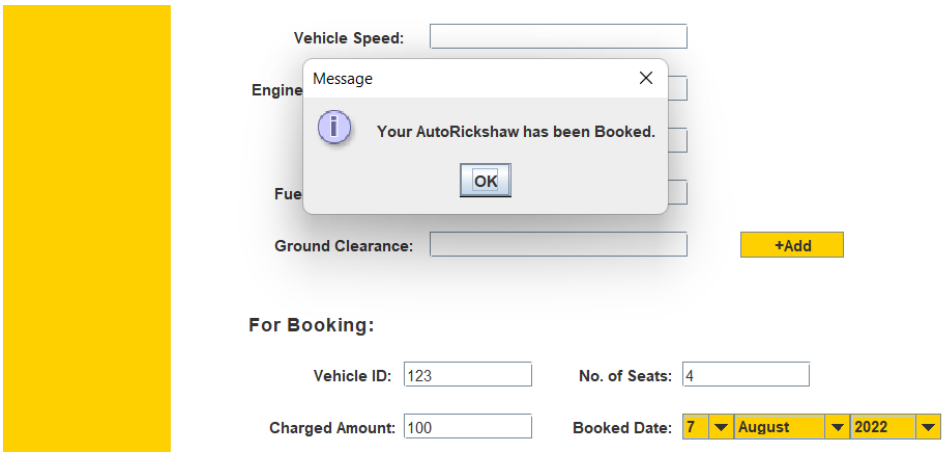

Figure 12: Booking an Autorickshaw successfully

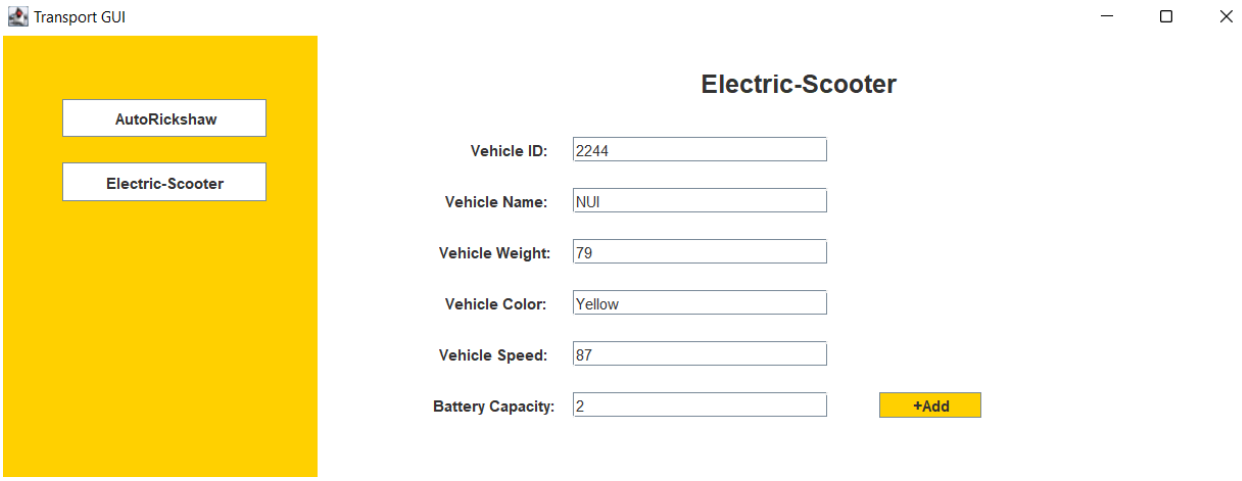

Figure 14:Entering values in Text Fields for Adding Scooter

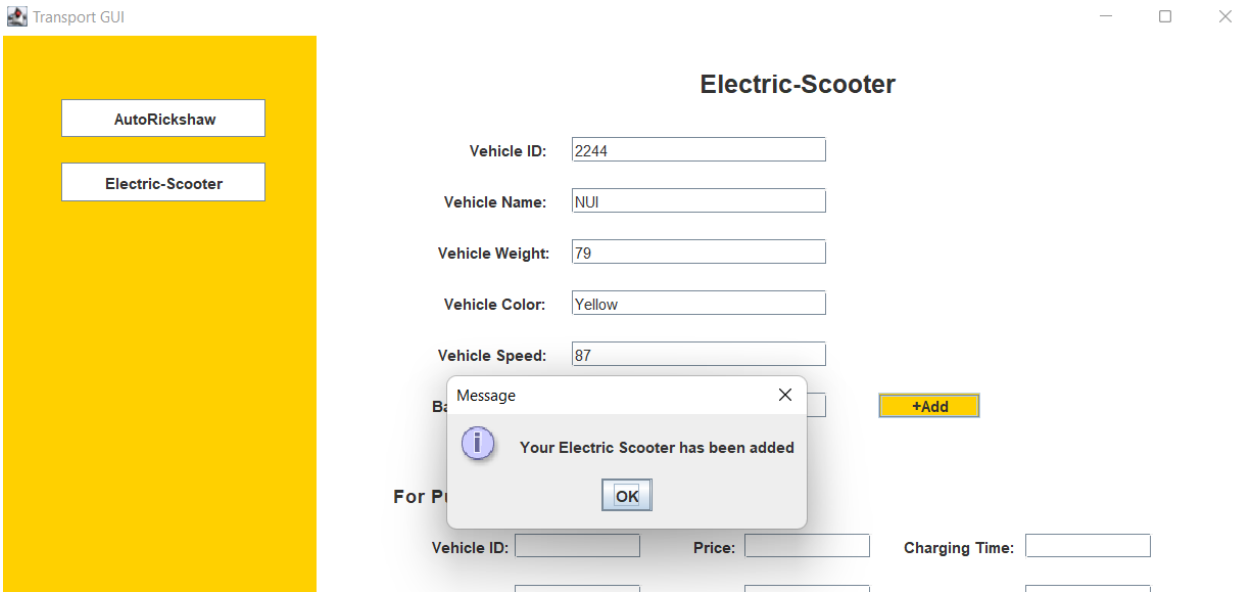

Figure 13: Electric Scooter added successfully

Figure 15: Entering values in Text Fields for Purchasing Scooter

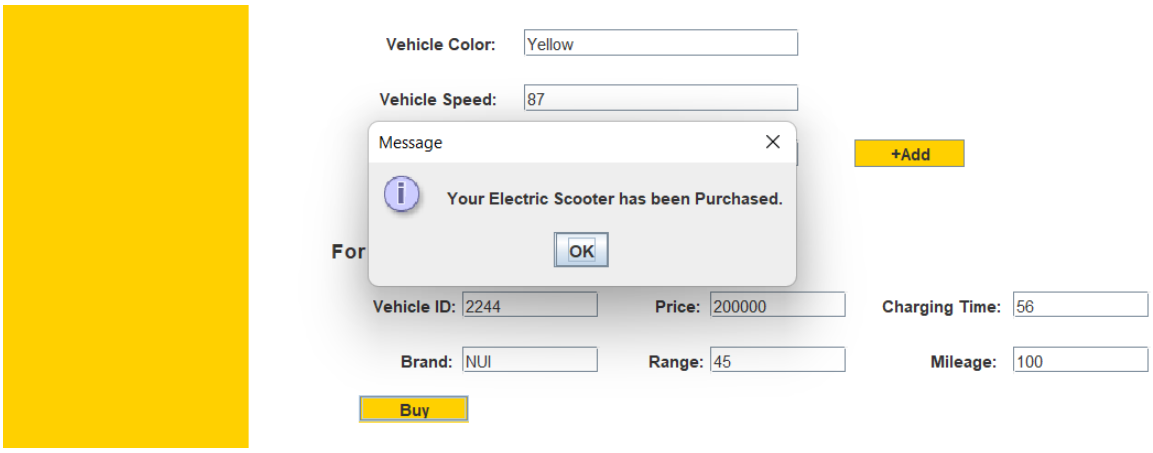

Figure 16:Electric Scooter Purchased Successfully

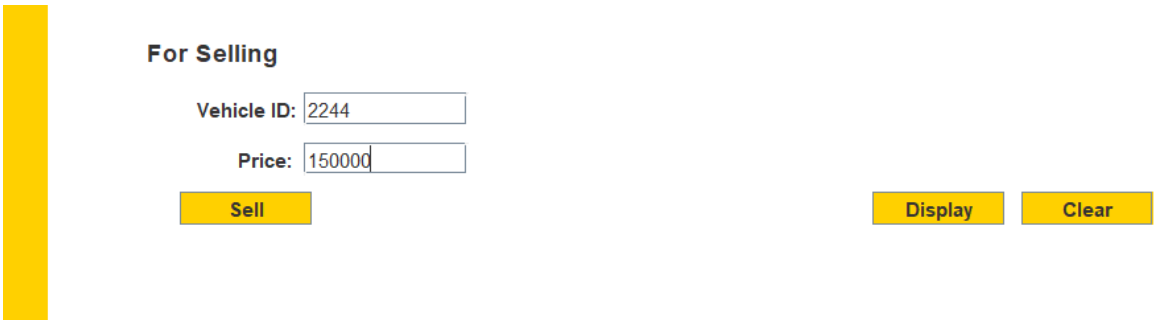

Figure 17: Entering values in Text Fields for Selling Scooter

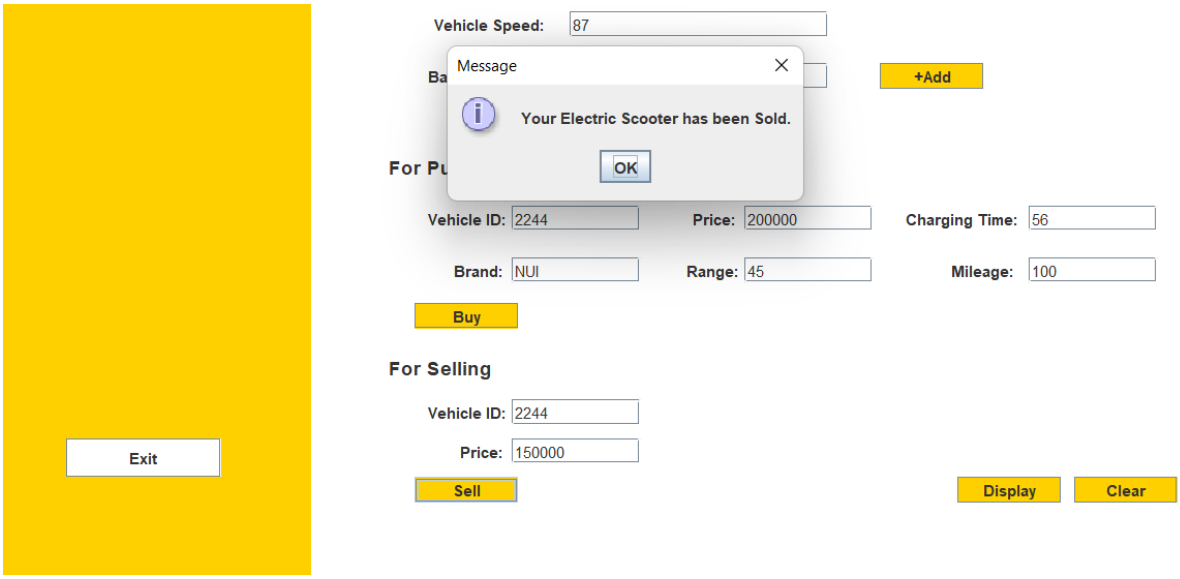

Figure 18:Electric Scooter selling Successfully

5.3. Test 3

Test no.	3
Objective	Display the messages when any error or exceptions occur.
Action	• Adding without providing the values in the parameters • Adding a Vehicle which has already been added. • Selling scooter without purchasing it. • Feed in String Value where int value is required.
Expected Result	It should display the appropriate error message as per the error.
Actual Result	It displayed the appropriate error message as per the error.
Conclusion	Objective Completed

Table 3:Test no.3

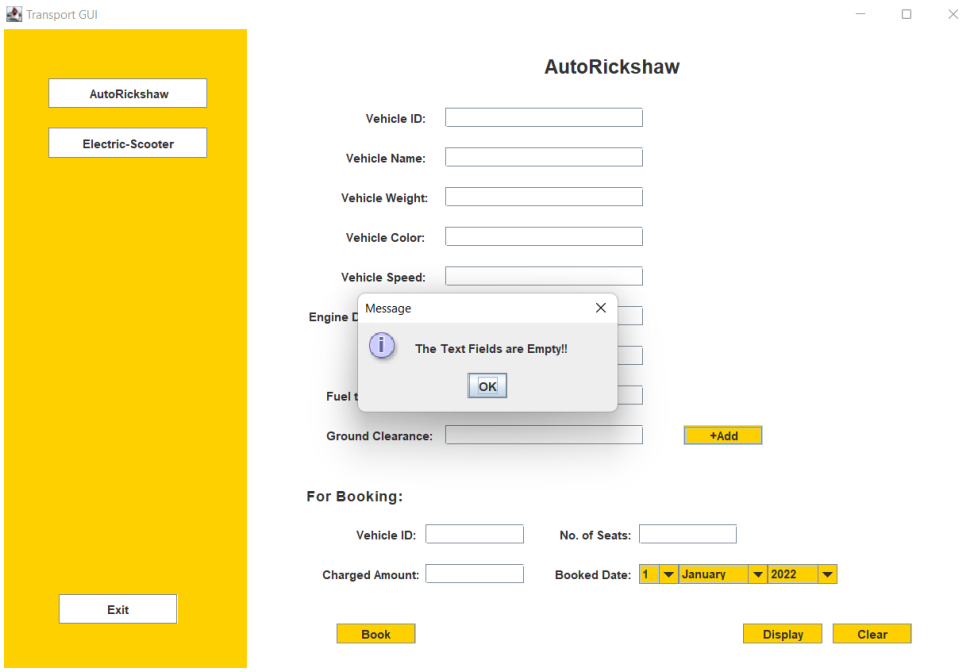

Figure 19: Entering Empty Fields

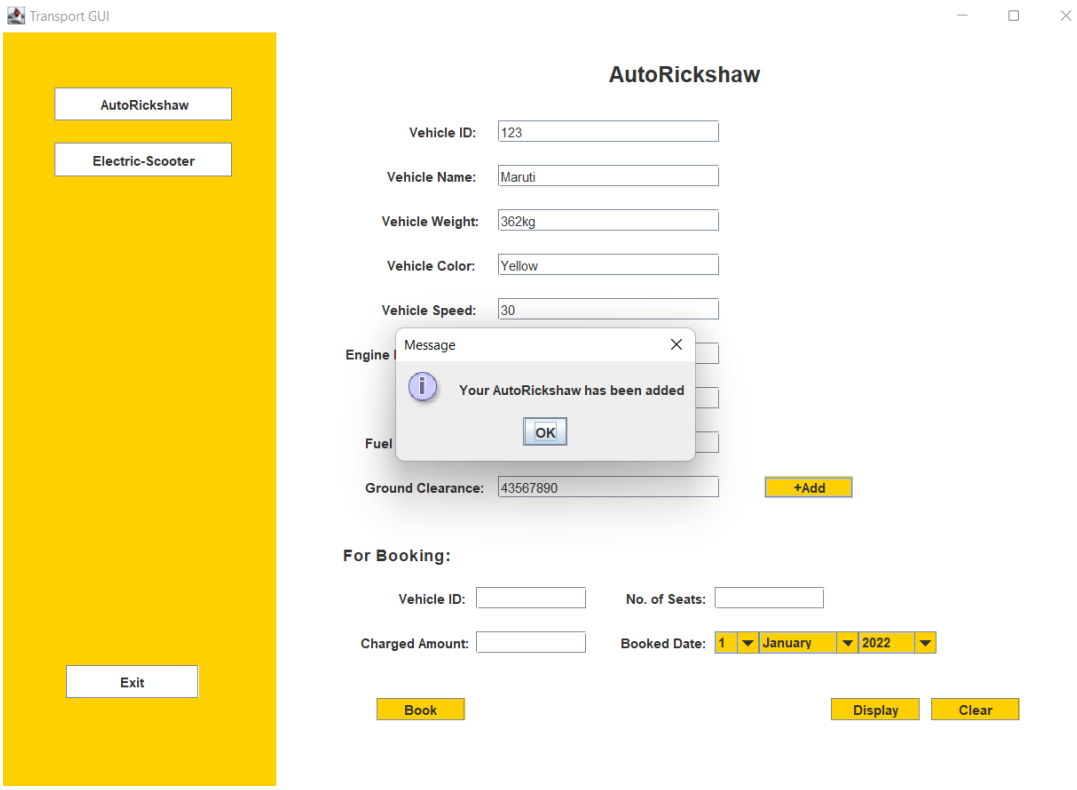

Figure 20:Adding autorickshaw

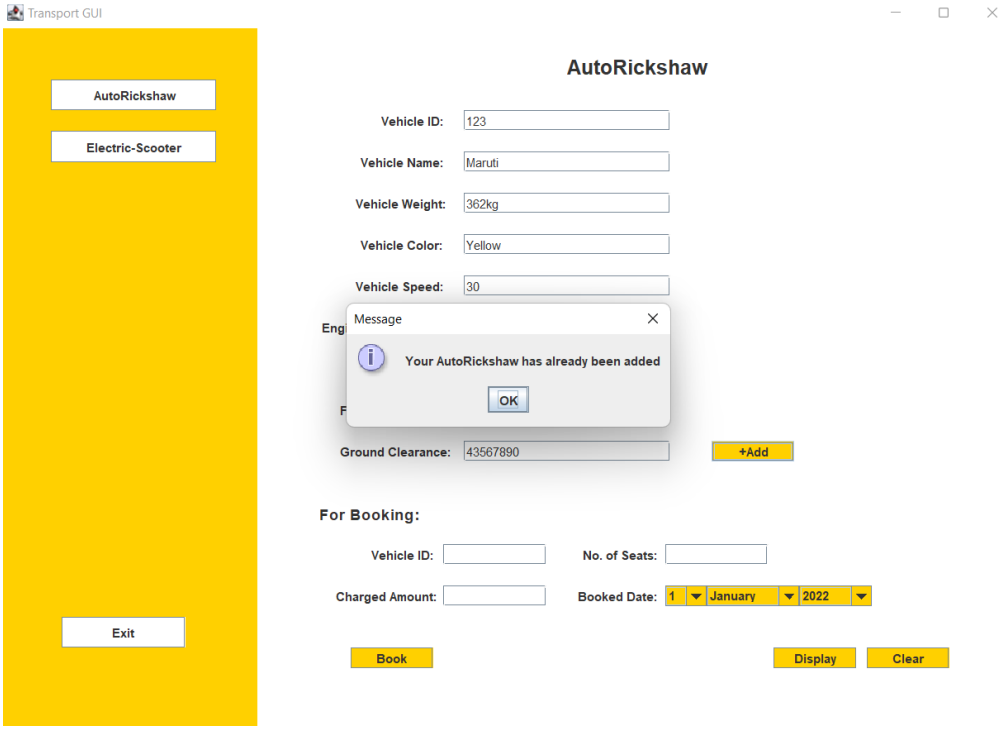

Figure 21: Adding the object again

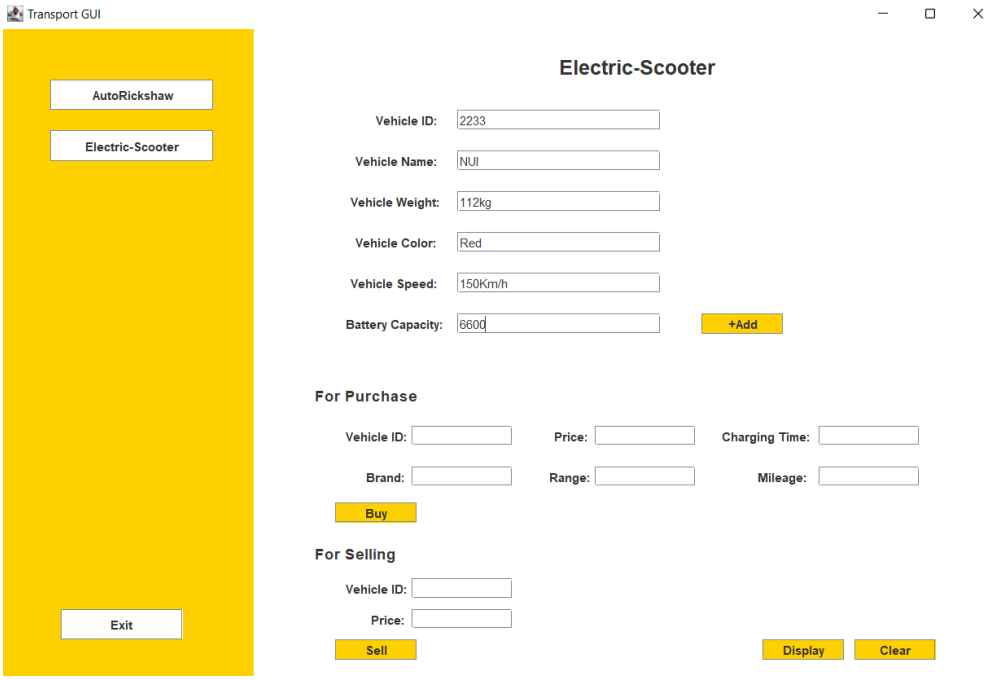

Figure 22:Entering a value in the parameters

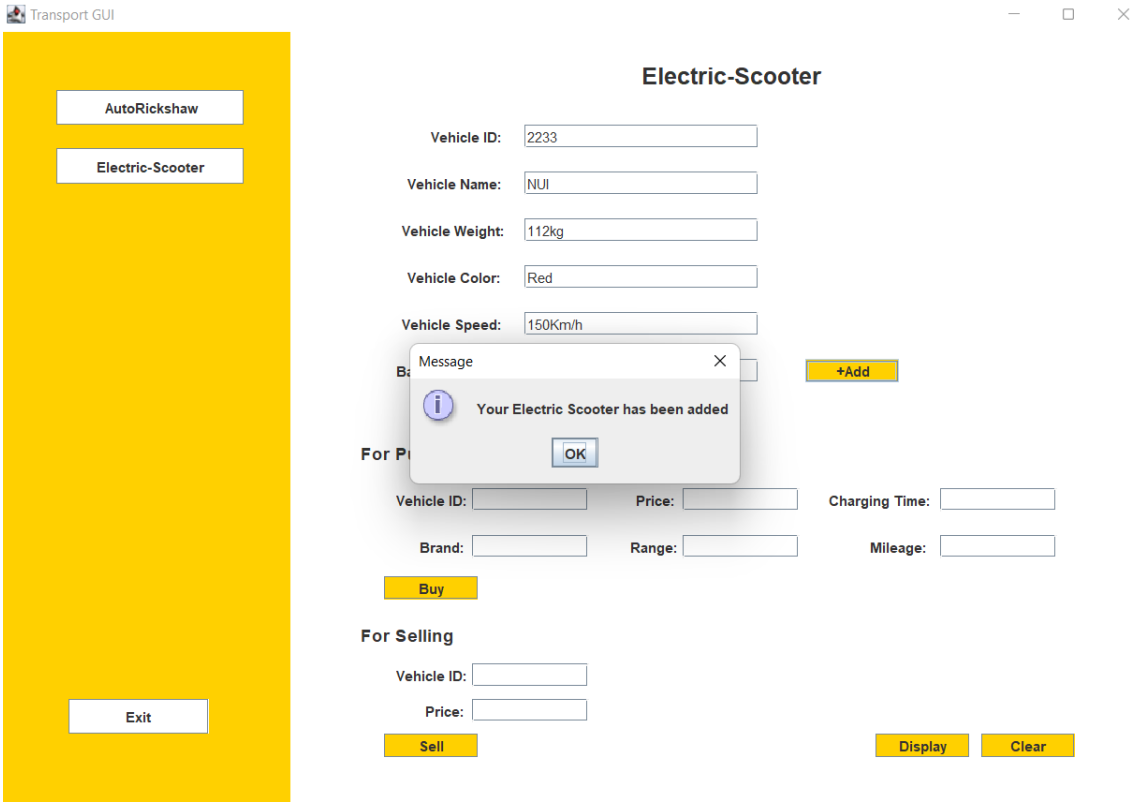

Figure 23:Adding an Electric Scooter

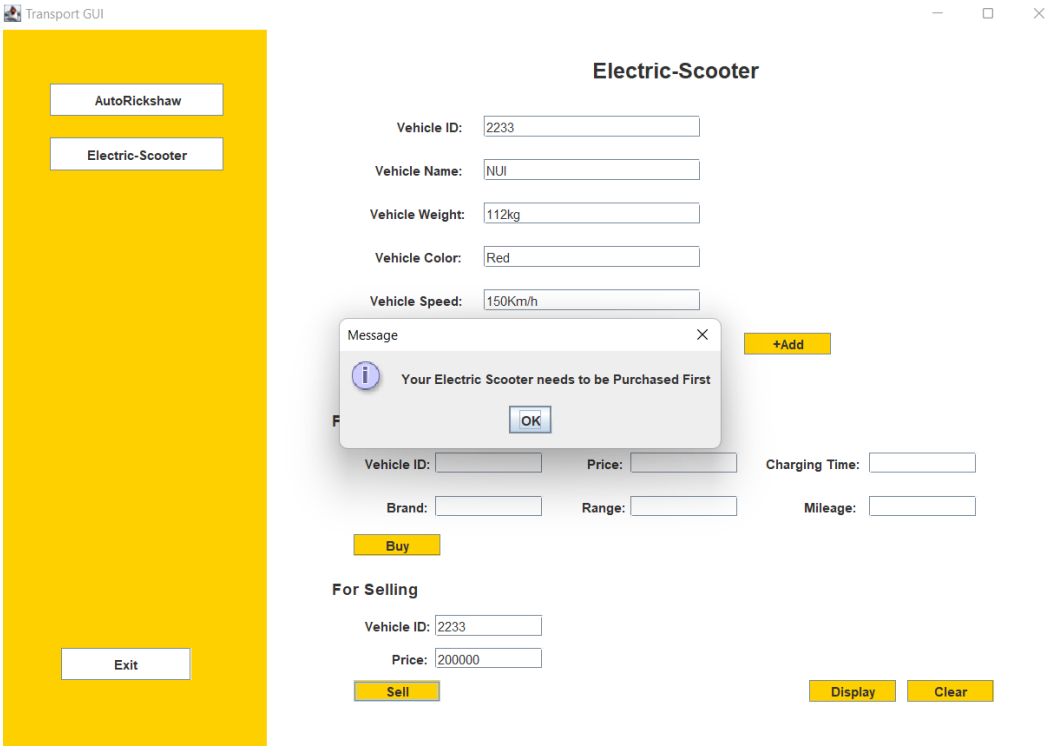

Figure 24:Selling Electric Scooter without purchasing it

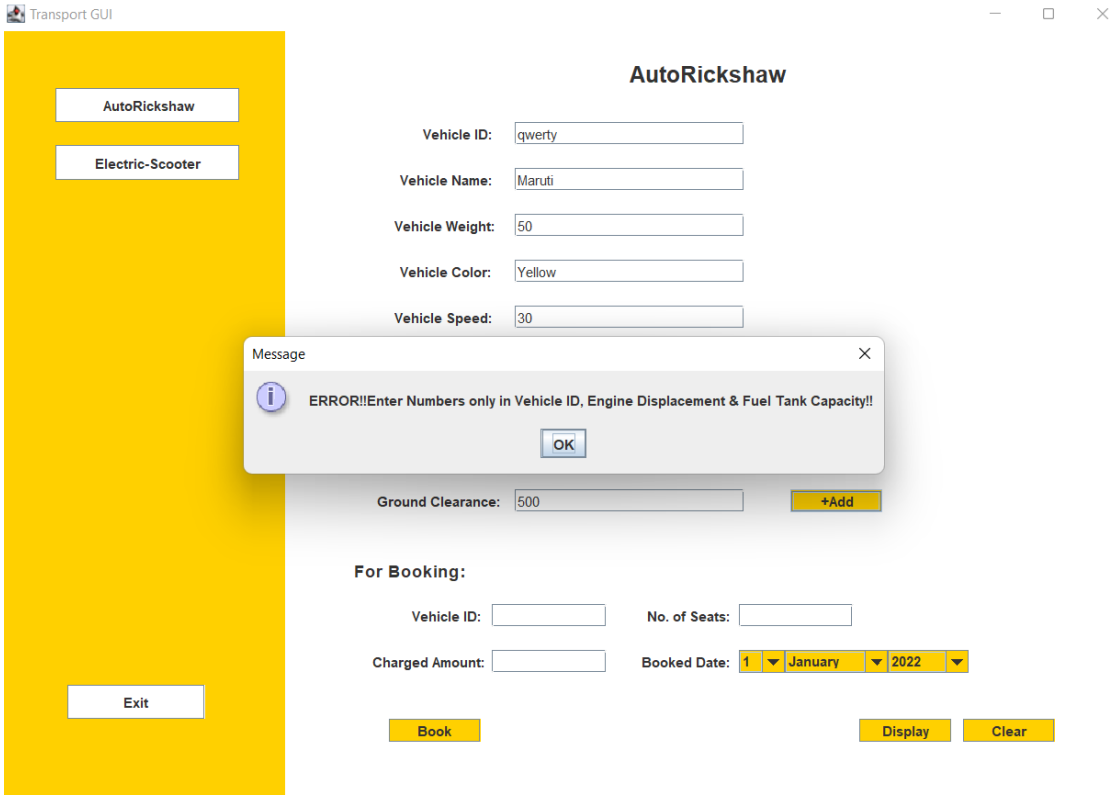

Figure 25: NumberFormatException

6. Error Detection and Correction

6.1. Syntax Error

A syntax error is an error in the source code of a program. Since computer programs must follow strict syntax to compile correctly, any aspects of the code that do not conform to the syntax of the programming language will produce a syntax error. (Techterms, 2022)

Here the Syntax error detected is a missing semi-colon which was easily detected and fixed.

```
359     }
360     //method to check the repeatition of value and adding the value in the ar
361     public void Auto_Checker()
362     {
363         //Creating all the variables and getting all the Values for the TextF
364         int VehicleID = Integer.parseInt(t1.getText());
365         String VehicleName = t2.getText();
366         String VehicleWeight = t3.getText();
367         String VehicleColor = t4.getText();
368         String VehicleSpeed = t5.getText();
369         int EngineDisplacement = Integer.parseInt(t6.getText());
370         String Torque = t7.getText();
371         int FuelTankCapacity = Integer.parseInt(t8.getText());
372         String GroundClearance = t9.getText()
373         boolean added = false;
374         //Creating a AutoRicksahw object named auto
375         AutoRickshaw auto = new AutoRickshaw(VehicleID ,VehicleName ,VehicleC
376         //Using For loop for checking the repeatition of the values entered i
377         for(Vehicle x : Vehicle)
378         {
379             if(x instanceof AutoRickshaw == true && x.getVehicleID()==Vehicle
380             {
381                 added = true;
```

Figure 26:Syntax Error is Detected

```
360     //method to check the repeatition of value and adding the value in the array li
362     public void Auto_Checker()
362     {
363         //Creating all the variables and getting all the Values for the TextField r
364         int VehicleID = Integer.parseInt(t1.getText());
365         String VehicleName = t2.getText();
366         String VehicleWeight = t3.getText();
367         String VehicleColor = t4.getText();
368         String VehicleSpeed = t5.getText();
369         int EngineDisplacement = Integer.parseInt(t6.getText());
370         String Torque = t7.getText();
371         int FuelTankCapacity = Integer.parseInt(t8.getText());
372         String GroundClearance = t9.getText();
373         boolean added = false;
374         //Creating a AutoRicksahw object named auto
375         AutoRickshaw auto = new AutoRickshaw(VehicleID ,VehicleName ,VehicleColor,V
376         //Using For loop for checking the repeatition of the values entered in the
377         for(Vehicle x : Vehicle)
378         {
379             if(x instanceof AutoRickshaw == true && x.getVehicleID()==VehicleID )
380             {
381                 added = true;
```

Figure 27: Syntax Error Corrected

6.2. Logical Error

A logic error is a mistake in a program's source code that results in incorrect or unexpected behaviour. It is a type of runtime error that may simply produce the wrong output or may cause a program to crash while running. (Techterms, 2022)

Here without the break the message is repeated, and the program doesn't run as per the requirements.

```
for(Vehicle x:Vehicle)
{
    //Checks the existence of the object and if it is an instance of AutoRicksahw
    if( x.getVehicleID()==VehicleID && x instanceof AutoRickshaw == true)
    {
        //Downcasting the object using the class Autorickshaw
        AutoRickshaw auto = (AutoRickshaw)x;
        //calling the isBooked from the AutoRickshaw class to varify the booking
        //and Displaying the appropriate messages
        if(auto.getisBooked() == false)
        {
            auto.Book(Date, ChargeAmount,NoOfSeats);
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has been Booked.");
            available = true;
        }
        else
        {
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has already been Booked.");
            available = true;
        }
    }
    //To check the existence of the object in the ArrayList
    else if(x.getVehicleID()!=VehicleID|| x instanceof AutoRickshaw == false)
    {
        exist = false;
        available = false;
    }
```

Figure 28:Logical Error Detected

```
for(Vehicle x:Vehicle)
{
    //Checks the existence of the object and if it is an instance of AutoRicksahw
    if( x.getVehicleID()==VehicleID && x instanceof AutoRickshaw == true)
    {
        //Downcasting the object using the class Autorickshaw
        AutoRickshaw auto = (AutoRickshaw)x;
        //calling the isBooked from the AutoRickshaw class to varify the booking
        //and Displaying the appropriate messages
        if(auto.getisBooked() == false)
        {
            auto.Book(Date, ChargeAmount,NoOfSeats);
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has been Booked.");
            available = true;
        }
        else
        {
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has already been Booked.");
            available = true;
        }
        break;
    }
    //To check the existence of the object in the ArrayList
    else if(x.getVehicleID()!=VehicleID|| x instanceof AutoRickshaw == false)
    {
        exist = false;
        available = false;
    }
```

Figure 29:Logical Error Corrected

6.3. Semantic Error

An error message that is semantic is one that is not generated by a program but does not do the right thing when it runs. An expression may not be evaluated in the order you expect, resulting in an incorrect result. (thesassway, 2022)

```
//Using For Loop to check the different situations when different events occur
for(Vehicle x:Vehicle)
{
    //Checks the existence of the object and if it is an instance of Electric Scooter
    if( x.getVehicleID()==VehicleID && x instanceof ElectricScooter == true)
    {
        //Downcasting the object using the class Electric Scooter
        ElectricScooter scooter = (ElectricScooter)x;
        //calling the hasPurchased from the Electric Scooter class to varify the booking
        //and Displaying the appropriate messages
        if(scooter.gethasPurchased() == false)
        {
            scooter.Purchase(Brand,Price,ChargingTime,Mileage,Range);
            JOptionPane.showMessageDialog(f,"Your Electric Scooter has been Purchased.");
            available = true;
        }
        else
        {
            JOptionPane.showMessageDialog(f,"Your Electric Scooter has been Purchased.");
            available = true;
        }
        break;
    }
    //To check the existence of the object in the ArrayList
    else if(x.getVehicleID()!=VehicleID|| x instanceof ElectricScooter == false)
    {
        exist=false;
        available = false;
    }
```

Figure 30:Semantic Error Detected

```
for(Vehicle x:Vehicle)
{
    //Checks the existence of the object and if it is an instance of Electric Scooter
    if( x.getVehicleID()==VehicleID && x instanceof ElectricScooter == true)
    {
        //Downcasting the object using the class Electric Scooter
        ElectricScooter scooter = (ElectricScooter)x;
        //calling the hasPurchased from the Electric Scooter class to varify the booking
        //and Displaying the appropriate messages
        if(scooter.gethasPurchased() == false)
        {
            scooter.Purchase(Brand,Price,ChargingTime,Mileage,Range);
            JOptionPane.showMessageDialog(f,"Your Electric Scooter has been Purchased.");
            available = true;
        }
        else
        {
            JOptionPane.showMessageDialog(f,"Your Electric Scooter has already been Purchased.");
            available = true;
        }
        break;
    }
    //To check the existence of the object in the ArrayList
    else if(x.getVehicleID()!=VehicleID|| x instanceof ElectricScooter == false)
    {
        exist=false;
        available = false;
    }
}
```

Figure 31: Semantic Error Corrected

7. Conclusion

The coursework was a great help in furthering my knowledge of programming in general and helped be proficient enough in the language of java. My interest in programming has been immense from the beginning of the semester and this work furthered my motivation in being more proficient than my current self.

If I l the project was its problem-solving aspects. At the beginning it was hard for me to understand that question but as I went through my ordeal I got better at understanding and implementing the solutions for my errors, be it syntax, semantic or logical. I faced a lot of problems with syntax errors as I was not familiar to the programming language of java. I faced many logical fallacies during the testing of the project but was able to solve all of it with the help available to me and my problem-solving skills. My teachers helped me understand and reassess my goals for the project. I still have certain doubts that are not clear about the project, but I think these doubts won't last long if I go through the ordeal of learning more about programming in a consistent way. The practical and real-world problem provided to us through the coursework will help us adapt and problem solve other problems well within our future. Due to this my motivation for the next coursework has increased and I. looking forward to more of the same or even different challenges. During this coursework I learned many things like time management, critical thinking and problem solving.

The material provided was clear, concise and to the point. The teachers have been a great guide while going through this unfamiliar terrain. I've learned all that I can here and will continue to do so. The learning for this module has not been a rough one as the content is not very to grasp. It has been a lot of understanding the material or looking at the material and trying to do it myself to understand.

Currently in the module I have not had a problem that has been a roadblock in my learning. There has been some minor in conveniences in the way, but I have understood it better. The part I had the most problem with was not understanding the questions provided to me in the beginning. Although it was something I struggled with it at the beginning I have become more accustomed to it.

Throughout this journey for the coursework, I had a lot of positive experience. The benefit of this coursework far outweighs the cons. This project has reignited my joy of programming. I felt like I was learning something new every step of the way. I feel that I am more ready than ever to progress forward in the field of programming and learning more nuanced ideas about programming.

Appendix

```
import javax.swing.*;

import java.awt.event.*;

import java.awt.*;

import java.util.*;

public class TransportGUI implements ActionListener

{

    //Declaring Instance variable for Jframe

    JFrame f;

    JPanel LP,RP1,RP2;

    JLabel title1,title2
,l1,VehicleID,VehicleID2,VehicleName,VehicleWeight,VehicleColor,VehicleSpeed,Engin
eDisplacement,Torque,FueltankCapacity,GroundClearance,ChargedAmount,noOfSeats
,BookedDate,l2,l3,eVehicleID,eVehicleID2,eVehicleID3,eVehicleName,eVehicleWeight,
eVehicleColor,eVehicleSpeed,Batterycapacity,Brand,Mileage,ChargingTime,Range,Pric
e,Price2;


    JTextField
t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12,et1,et2,et3,et4,et5,et6,et7,et8,et9,et10,et11,et12,et13,
et14;

    JButton
Auto,Elec,Exit,add1,Book,Auto_Display,Auto_Clear,add2,Scooter_Display,Scooter_Cle
ar,Purchase,Sell;

    JComboBox<String> cb1,cb2,cb3;

    Color LC,RC1,RC2;

    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"};

    String Month[] =
{"January","Febuary","March","April","May","June","July","August","September","October
","November","December"};

    String Year[] = {"2022","2023","2024"};
```

```
    ArrayList<Vehicle> Vehicle = new ArrayList<Vehicle>();

    public TransportGUI()
    {
        f = new JFrame("Transport GUI");

        LP = new JPanel();
        RP1 = new JPanel();
        RP2 = new JPanel();
        LC = new Color(255, 208, 0);
        RC1 = new Color(255, 255, 255);
        RC2 = new Color(255, 255, 255);

        title1 = new JLabel("AutoRickshaw");
        VehicleID = new JLabel("Vehicle ID:");
        VehicleName = new JLabel("Vehicle Name:");
        VehicleWeight = new JLabel("Vehicle Weight:");
        VehicleColor = new JLabel("Vehicle Color:");
        VehicleSpeed = new JLabel("Vehicle Speed:");
        EngineDisplacement = new JLabel("Engine Displacement:");
        Torque = new JLabel("Torque:");
        FueltankCapacity = new JLabel("Fuel tank Capacity:");
        GroundClearance = new JLabel("Ground Clearance:");
        l1 = new JLabel("For Booking:");
        VehicleID2 = new JLabel("Vehicle ID:");
        ChargedAmount = new JLabel("Charged Amount:");
        noOfSeats = new JLabel("No. of Seats:");
```

```
BookedDate = new JLabel("Booked Date:");

t1 = new JTextField();
t2 = new JTextField();
t3 = new JTextField();
t4 = new JTextField();
t5 = new JTextField();
t6 = new JTextField();
t7 = new JTextField();
t8 = new JTextField();
t9 = new JTextField();
t10 = new JTextField();
t11 = new JTextField();
t12 = new JTextField();

add1 = new JButton("+Add");
add1.setBackground(LC);
Book = new JButton("Book");
Book.setBackground(LC);
Auto_Display = new JButton("Display");
Auto_Display.setBackground(LC);
Auto_Clear = new JButton("Clear");
Auto_Clear.setBackground(LC);

cb1 = new JComboBox<String>(Day);
cb1.setBackground(LC);
cb2 = new JComboBox<String>(Month);
cb2.setBackground(LC);
```

```
cb3 = new JComboBox<String>(Year);
cb3.setBackground(LC);

title2 = new JLabel("Electric-Scooter");
eVehicleID = new JLabel("Vehicle ID:");
eVehicleName = new JLabel("Vehicle Name:");
eVehicleWeight = new JLabel("Vehicle Weight:");
eVehicleColor = new JLabel("Vehicle Color:");
eVehicleSpeed = new JLabel("Vehicle Speed:");
Batterycapacity = new JLabel("Battery Capacity:");
l2 = new JLabel("For Purchase");
eVehicleID2 = new JLabel("Vehicle ID:");
Brand = new JLabel("Brand:");
ChargingTime = new JLabel("Charging Time:");
Range = new JLabel("Range:");
Mileage = new JLabel("Mileage:");
Price = new JLabel("Price:");
l3 = new JLabel("For Selling");
eVehicleID3 = new JLabel("Vehicle ID:");
Price2 = new JLabel("Price:");

et1 = new JTextField();
et2 = new JTextField();
et3 = new JTextField();
et4 = new JTextField();
et5 = new JTextField();
et6 = new JTextField();
et7 = new JTextField();
```

```
et8 = new JTextField();
et9 = new JTextField();
et10 = new JTextField();
et11 = new JTextField();
et12 = new JTextField();
et13 = new JTextField();
et14 = new JTextField();

add2 = new JButton("+Add");
add2.setBackground(LC);
Purchase = new JButton("Buy");
Purchase.setBackground(LC);
Sell = new JButton("Sell");
Sell.setBackground(LC);
Scooter_Display = new JButton("Display");
Scooter_Display.setBackground(LC);
Scooter_Clear = new JButton("Clear");
Scooter_Clear.setBackground(LC);


Auto = new JButton("AutoRickshaw");
Auto.setBackground(RC1);
Elec = new JButton("Electric-Scooter");
Elec.setBackground(RC1);
Exit = new JButton("Exit");
Exit.setBackground(RC1);
```

```
title1.setBounds(300,15,200,45);
title1.setFont(new Font("San-Serif",Font.BOLD,20));
VehicleID.setBounds(120,70,180,40);
VehicleName.setBounds(100,110,180,40);
VehicleWeight.setBounds(95,150,180,40);
VehicleColor.setBounds(100,190,180,40);
VehicleSpeed.setBounds(95,230,180,40);
EngineDisplacement.setBounds(62,270,180,40);
Torque.setBounds(135,310,180,40);
FueltankCapacity.setBounds(80,350,180,40);
GroundClearance.setBounds(80,390,180,40);
l1.setBounds(60,450,180,40);
l1.setFont(new Font("San-Serif",Font.BOLD,15));
VehicleID2.setBounds(110,490,180,40);
ChargedAmount.setBounds(76,530,180,40);
noOfSeats.setBounds(315,490,180,40);
BookedDate.setBounds(310,530,180,40);

t1.setBounds(200,80,200,20);
t2.setBounds(200,120,200,20);
t3.setBounds(200,160,200,20);
t4.setBounds(200,200,200,20);
t5.setBounds(200,240,200,20);
t6.setBounds(200,280,200,20);
t7.setBounds(200,320,200,20);
t8.setBounds(200,360,200,20);
t9.setBounds(200,400,200,20);
t10.setBounds(180,500,100,20);
```

```
t11.setBounds(180,540,100,20);
t12.setBounds(395,500,100,20);
add1.setBounds(440,400,80,20);
cb1.setBounds(395,540,40,20);
cb2.setBounds(435,540,90,20);
cb3.setBounds(525,540,70,20);
Book.setBounds(90,600,80,20);
Auto_Display.setBounds(500,600,80,20);
Auto_Clear.setBounds(590,600,80,20);

title2.setBounds(300,15,200,45);
title2.setFont(new Font("San-Serif",Font.BOLD,20));
eVehicleID.setBounds(120,70,180,40);
eVehicleName.setBounds(100,110,180,40);
eVehicleWeight.setBounds(95,150,180,40);
eVehicleColor.setBounds(100,190,180,40);
eVehicleSpeed.setBounds(95,230,180,40);
Batterycapacity.setBounds(90,270,180,40);
et1.setBounds(200,80,200,20);
et2.setBounds(200,120,200,20);
et3.setBounds(200,160,200,20);
et4.setBounds(200,200,200,20);
et5.setBounds(200,240,200,20);
et6.setBounds(200,280,200,20);
add2.setBounds(440,280,80,20);
l2.setBounds(60,340,200,40);
l2.setFont(new Font("San-Serif",Font.BOLD,15));
eVehicleID2.setBounds(90,380,180,40);
```

```
Brand.setBounds(110,420,180,40);
Price.setBounds(295,380,180,40);
Range.setBounds(290,420,180,40);
ChargingTime.setBounds(460,380,180,40);
Mileage.setBounds(495,420,180,40);
l3.setBounds(60,495,200,40);
l3.setFont(new Font("San-Serif",Font.BOLD,15));
eVehicleID3.setBounds(90,530,180,40);
Price2.setBounds(115,560,180,40);
Scooter_Display.setBounds(500,600,80,20);
et7.setBounds(155,390,100,20);
et8.setBounds(155,430,100,20);
et9.setBounds(335,390,100,20);
et10.setBounds(335,430,100,20);
et11.setBounds(555,390,100,20);
et12.setBounds(555,430,100,20);
et13.setBounds(155,540,100,20);
et14.setBounds(155,570,100,20);
Purchase.setBounds(80,465,80,20);
Sell.setBounds(80,600,80,20);
Scooter_Clear.setBounds(590,600,80,20);

LP.setBackground(LC);
LP.setBounds(0,0,250,800);
RP1.setBackground(RC1);
RP1.setBounds(250,0,950,800);

RP2.setBackground(RC2);
```

```
RP2.setBounds(250,0,950,800);

Auto.setBounds(50,50,160,30);
Elec.setBounds(50,100,160,30);
Exit.setBounds(60,570,120,30);


Elec.addActionListener(this);
Auto.addActionListener(this);
Exit.addActionListener(this);
Auto_Clear.addActionListener(this);
Scooter_Clear.addActionListener(this);
add1.addActionListener(this);
Book.addActionListener(this);
add2.addActionListener(this);
Purchase.addActionListener(this);
Sell.addActionListener(this);
Auto_Display.addActionListener(this);
Scooter_Display.addActionListener(this);

f.setLayout(null);
f.setVisible(true);
f.setSize(1000,720);
//when the frame appears it puts it in the middle of the screen
f.setLocationRelativeTo(null);
//Adding the the Panels to the frame
f.add(LP);
f.add(RP1);
```

```
f.add(RP2);

RP1.setLayout(null);
RP1.add(title1);
RP1.add(VehicleID);
RP1.add(VehicleName);
RP1.add(VehicleWeight);
RP1.add(VehicleColor);
RP1.add(VehicleSpeed);
RP1.add(EngineDisplacement);
RP1.add(Torque);
RP1.add(FueltankCapacity);
RP1.add(GroundClearance);
RP1.add(l1);
RP1.add(VehicleID2);
RP1.add(ChargedAmount);
RP1.add(noOfSeats);
RP1.add(BookedDate);
RP1.add(t1);
RP1.add(t2);
RP1.add(t3);
RP1.add(t4);
RP1.add(t5);
RP1.add(t6);
RP1.add(t7);
RP1.add(t8);
RP1.add(t9);
RP1.add(t10);
```

```
RP1.add(t11);
RP1.add(t12);
RP1.add(add1);
RP1.add(cb1);
RP1.add(cb2);
RP1.add(cb3);
RP1.add(Auto_Display);
RP1.add(Auto_Clear);
RP1.add(Book);

RP2.setLayout(null);
RP2.add(title2);
RP2.add(eVehicleID);
RP2.add(eVehicleName);
RP2.add(eVehicleWeight);
RP2.add(eVehicleColor);
RP2.add(eVehicleSpeed);
RP2.add(Batterycapacity);
RP2.add(et1);
RP2.add(et2);
RP2.add(et3);
RP2.add(et4);
RP2.add(et5);
RP2.add(et6);
RP2.add(add2);
RP2.add(l2);
RP2.add(eVehicleID2);
RP2.add(Brand);
```

```
        RP2.add(Price);
        RP2.add(Range);
        RP2.add(Mileage);
        RP2.add(ChargingTime);
        RP2.add(et7);
        RP2.add(et8);
        RP2.add(et9);
        RP2.add(et10);
        RP2.add(et11);
        RP2.add(et12);
        RP2.add(et13);
        RP2.add(et14);
        RP2.add(l3);
        RP2.add(Purchase);
        RP2.add(eVehicleID3);
        RP2.add(Price2);
        RP2.add(Sell);
        RP2.add(Scooter_Display);
        RP2.add(Scooter_Clear);

        LP.setLayout(null);
        LP.add(Auto);
        LP.add(Elec);
        LP.add(Exit);
    }

    public void Auto_Checker()
    {
```

```
int VehicleID = Integer.parseInt(t1.getText());
String VehicleName = t2.getText();
String VehicleWeight = t3.getText();
String VehicleColor = t4.getText();
String VehicleSpeed = t5.getText();
int EngineDisplacement = Integer.parseInt(t6.getText());
String Torque = t7.getText();
int FuelTankCapacity = Integer.parseInt(t8.getText());
String GroundClearance = t9.getText();
boolean added = false;

AutoRickshaw auto = new AutoRickshaw(VehicleID ,VehicleName
,VehicleColor,VehicleSpeed,VehicleWeight,EngineDisplacement,Torque,FuelTankCapa
city,GroundClearance);
//Using For loop for checking the repeatition of the values entered in the Text Field
for(Vehicle x : Vehicle)
{
    if(x instanceof AutoRickshaw == true && x.getVehicleID()==VehicleID )
    {
        added = true;
    }
}

if(added == true)
{
    JOptionPane.showMessageDialog(f,"Your AutoRickshaw has already been
added");
}
```

```
        else
        {
            Vehicle.add(auto);
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has been added");
        }
    }

    public void Scooter_Checker()
    {

        int VehicleID = Integer.parseInt(et1.getText());
        String VehicleName = et2.getText();
        String VehicleWeight = et3.getText();
        String VehicleColor = et4.getText();
        String VehicleSpeed = et5.getText();
        int BatteryCapacity = Integer.parseInt(et6.getText());
        boolean added = false;
        ElectricScooter scooter = new ElectricScooter(VehicleID ,VehicleName
,VehicleColor,VehicleSpeed,VehicleWeight,BatteryCapacity);

        for(Vehicle x : Vehicle)
        {
            if(x instanceof ElectricScooter == true && x.getVehicleID()==VehicleID)
            {
                added = true;
            }
        }

        if(added == true)
```

```
        {
            JOptionPane.showMessageDialog(f,"Your Electric Scooter has already been added");
        }
        else
        {
            Vehicle.add(scooter);
            JOptionPane.showMessageDialog(f,"Your Electric Scooter has been added");
        }
    }

    public void Auto_Clearer()
    {
        t1.setText("");
        t2.setText("");
        t3.setText("");
        t4.setText("");
        t5.setText("");
        t6.setText("");
        t7.setText("");
        t8.setText("");
        t9.setText("");
        t10.setText("");
        t11.setText("");
        t12.setText("");
        cb1.setSelectedItem("1");
        cb2.setSelectedItem("January");
        cb3.setSelectedItem("2022");
    }
```

```
    public void Scooter_Clearer()
    {
        et1.setText("");
        et2.setText("");
        et3.setText("");
        et4.setText("");
        et5.setText("");
        et6.setText("");
        et7.setText("");
        et8.setText("");
        et9.setText("");
        et10.setText("");
        et11.setText("");
        et12.setText("");
        et13.setText("");
        et14.setText("");
    }

    public void Book_Checker()
    {

        int  VehicleID = Integer.parseInt(t10.getText());
        int  NoOfSeats= Integer.parseInt(t11.getText());
        int  ChargeAmount= Integer.parseInt(t12.getText());
        String Date = cb1.getSelectedItem()+" "+cb2.getSelectedItem()+"
"+cb3.getSelectedItem();
        boolean available = true;
        boolean exist = true;
```

```
for(Vehicle x:Vehicle)
{
    //Checks the existence of the object and if it is an instance of AutoRicksahw
    if( x.getVehicleID()==VehicleID && x instanceof AutoRickshaw == true)
    {

        AutoRickshaw auto = (AutoRickshaw)x;

        if(auto.getisBooked() == false)
        {
            auto.Book(Date, ChargeAmount,NoOfSeats);
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has been
Booked.");
            available = true;
        }
        else
        {
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw has already been
Booked.");
            available = true;
        }
        break;
    }

    else if(x.getVehicleID()!=VehicleID|| x instanceof AutoRickshaw == false)
    {
        exist = false;
        available = false;
```

```
            }
        }

        if(exist == false && available == false)
        {
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw dosen't Exist!!");
        }
        else if(Vehicle.isEmpty())
        {
            JOptionPane.showMessageDialog(f,"Your AutoRickshaw dosen't Exist!!");
        }
    }

    public void Purchase_Checker()
    {

        int  VehicleID = Integer.parseInt(et7.getText());
        String Brand = et8.getText();
        int  Price = Integer.parseInt(et9.getText());
        String ChargingTime = et10.getText();
        String Mileage = et11.getText();
        int  Range = Integer.parseInt(et12.getText());
        boolean available = true;
        boolean exist = true;
        for(Vehicle x:Vehicle)
        {

            if( x.getVehicleID()==VehicleID && x instanceof ElectricScooter == true)
```

```
        {

            ElectricScooter scooter = (ElectricScooter)x;

            if(scooter.gethasPurchased() == false)
            {
                scooter.Purchase(Brand,Price,ChargingTime,Mileage,Range);
                JOptionPane.showMessageDialog(f,"Your Electric Scooter has been
Purchased.");
                available = true;
            }
            else
            {
                JOptionPane.showMessageDialog(f,"Your Electric Scooter has already
been Purchased.");
                available = true;
            }
            break;
        }
        else if(x.getVehicleID()!=VehicleID|| x instanceof ElectricScooter == false)
        {
            exist=false;
            available = false;
        }
    }

    if(exist==false && available == false)
    {
        JOptionPane.showMessageDialog(f,"Your Electric Scooter doesn't exist!!");
```

```
        }
        else if(Vehicle.isEmpty())
        {
          JOptionPane.showMessageDialog(f,"Your Electric Scooter doesn't exist!!");
        }
    }

    public void Sold_Checker()
    {
        int  VehicleID = Integer.parseInt(et13.getText());
        int  Price = Integer.parseInt(et14.getText());
        boolean available = true;
        boolean exist = true;

        for(Vehicle x:Vehicle)
        {

            if( x.getVehicleID()==VehicleID )
            {
                if(x instanceof ElectricScooter == true )
                {

                    ElectricScooter scooter = (ElectricScooter)x;

                    if (scooter.gethasSold()==false && scooter.gethasPurchased() == true)
                    {
                        JOptionPane.showMessageDialog(f,"Your Electric Scooter has been
Sold.");
                        scooter.sell(Price);
```

```
                    available = true;
                }
                else if(scooter.gethasPurchased() == false &&
scooter.gethasSold()==false)
                {
                    JOptionPane.showMessageDialog(f,"Your Electric Scooter needs to be
Purchased First");
                    available = true;
                }
                else
                {
                    JOptionPane.showMessageDialog(f,"Your Electric Scooter has already
been Sold.");
                    available = true;
                }
            }
            break;
        }
        else if(x.getVehicleID()!=VehicleID|| x instanceof ElectricScooter == false)
        {
            exist=false;
            available = false;
        }
    }

    if(exist==false && available == false)
    {
        JOptionPane.showMessageDialog(f,"Your Electric Scooter doesn't exist!!");
    }
```

```
        else if(Vehicle.isEmpty())
        {
            JOptionPane.showMessageDialog(f,"Your Electric Scooter doesn't exist!!");
        }
    }

    public void actionPerformed(ActionEvent e)
    {
        //adding functionality to the add Button
        if(e.getSource() == add1)
        {

            try
            {

if(t1.getText().trim().isEmpty()||t2.getText().trim().isEmpty()||t3.getText().trim().isEmpty()||t4.getText().trim().isEmpty()||t5.getText().trim().isEmpty()||t6.getText().trim().isEmpty()||t7.getText().trim().isEmpty()||t8.getText().trim().isEmpty()||t9.getText().trim().isEmpty())
                {
                    JOptionPane.showMessageDialog(f,"The Text Fields are Empty!!");
                }
                else
                {
                    //calling the Auto_Checker to check the values
                    Auto_Checker();
                }
            }
            catch(NumberFormatException error)
```

```
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Enter Numbers only in Vehicle
ID, Engine Displacement & Fuel Tank Capacity!!");
            }
            catch(Exception error)
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Re-enter the values again!!");
            }
        }
        if(e.getSource() == add2)
        {
            //using try catch the catch the exceptions
            try
            {
                //checking if the Text Fields are Empty

if(et1.getText().trim().isEmpty()||et2.getText().trim().isEmpty()||et3.getText().trim().isEmp
ty()||et4.getText().trim().isEmpty()||et5.getText().trim().isEmpty()||et6.getText().trim().isE
mpty())
                {
                    JOptionPane.showMessageDialog(f,"The Text Fields are Empty!!");
                }
                else
                {
                    Scooter_Checker();
                }
            }

            catch(NumberFormatException error)
            {
```

```
            JOptionPane.showMessageDialog(f,"ERROR!!Enter Numbers only in Vehicle
ID & Battery Capacity!!");
        }
        catch(Exception error)
        {
            JOptionPane.showMessageDialog(f,"ERROR!!Re-enter the values again!!");
        }
    }
    if(e.getSource() == Book)
    {
        try
        {

if(t10.getText().trim().isEmpty()||t11.getText().trim().isEmpty()||t12.getText().trim().isEmp
ty())
            {
                JOptionPane.showMessageDialog(f,"The Text Fields are Empty!!");
            }
            else
            {
                Book_Checker();
            }
        }


        catch(NumberFormatException error)
        {
            JOptionPane.showMessageDialog(f,"ERROR!!Enter Numbers only in Vehicle
ID, No. of Seats & Amount!!");
        }
        catch(Exception error)
```

```
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Re-enter the values again!!");
            }
        }
        if(e.getSource() == Purchase)
        {
             try
            {

if(et7.getText().trim().isEmpty()||et8.getText().trim().isEmpty()||et9.getText().trim().isEmp
ty()||et10.getText().trim().isEmpty()||et11.getText().trim().isEmpty()||et12.getText().trim().
isEmpty())
                {
                    JOptionPane.showMessageDialog(f,"The Text Fields are Empty!!");
                }
                else
                {
                    Purchase_Checker();
                }
            }


            catch(NumberFormatException error)
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Enter Numbers only in Vehicle
ID, Price & Range!!");
            }
            catch(Exception error)
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Re-enter the values again!!");
            }
```

```
        }
        if(e.getSource() == Sell)
        {
            try
            {
                if(et13.getText().trim().isEmpty()||et14.getText().trim().isEmpty())
                {
                    JOptionPane.showMessageDialog(f,"The Text Fields are Empty!!");
                }
                else
                {
                    Sold_Checker();
                }
            }
            //Catching differnt types of Errors and Displaying appropriate messages
            catch(NumberFormatException error)
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Enter Numbers only in Vehicle
ID & Price!!");
            }
            catch(Exception error)
            {
                JOptionPane.showMessageDialog(f,"ERROR!!Re-enter the values again!!");
            }
        }
        if(e.getSource() == Auto_Display)
        {
            for(Vehicle x:Vehicle)
            {
```

```
            if(x instanceof AutoRickshaw == true)
            {
                AutoRickshaw auto = (AutoRickshaw)x;
                auto.display();
            }
        }
    }
    if(e.getSource() == Scooter_Display)
    {
        for(Vehicle x:Vehicle)
        {
            if(x instanceof ElectricScooter == true)
            {
                ElectricScooter scooter = (ElectricScooter)x;
                scooter.display();
            }
        }
    }
    if(e.getSource() == Elec)
    {
        RP1.setVisible(false);
        RP2.setVisible(true);
        Auto_Clearer();
    }
    if(e.getSource() == Auto)
    {
        RP2.setVisible(false);
        RP1.setVisible(true);
```

```
            Scooter_Clearer();
        }
        if(e.getSource() == Auto_Clear )
        {
            Auto_Clearer();
        }
        if(e.getSource() == Scooter_Clear)
        {
            Scooter_Clearer();
        }
        if(e.getSource() == Exit)
        {
            System.exit(0);
        }
    }



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

References

Techterms, 2022. *techterms.* [Online]
Available at: https://techterms.com/definition/compile
[Accessed Thusrday August 2022].

thesassway, 2022. *thesassway.* [Online]
Available at: https://thesassway.com/what-is-semantic-error-in-computer-science/#1
[Accessed Thursday August 2022].