LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CC4057NI Introduction to Information Systems

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2021-22 Spring

Student Name: Saugat Basnet

Group: N7

London Met ID:22015870

College ID:np01nt4s220042

Assignment Due Date:8/5/2022

Assignment Submission Date:8/5/220042

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.

Acknowledgment

Firstly, I would like to be very thankful to our module lecturer Mr. Pramod Tuladhar for providing us with a platform to help develop skills. This project has given us the opportunity to improve our skills in problem-solving and doing research for solving the problems which arise while working on the project. Secondly, I would like to thank Islington college and London met university for providing us with an excellent environment for learning new skills and also helping us to learn about new technology. Lastly, I would like to thank my friends who suggested to me and helped me solve some of the problems that arise during my project.

Purpose

The purpose of the coursework was to make us familiar with how the GUI is made and how they work. In this coursework, we were tasked to make a GUI for
the system program which was created in the first coursework. Here I created a GUI using different components of java like Text Field, Button,
Label, Combo Box, Panel, Frame, ETC. this coursework helped us to get familiarized with how certain components of java works.

Tools used

While doing this there was much software used. for the coding portion, Bluej was used as it is created for educational purposes. Students who are new
to the Java environment will find BlueJ to be the ideal introduction. It is simple to learn and has every feature a Java program could have.
BlueJ is an integrated development environment (IDE) for the Java programming language, developed mainly for educational purposes, but also suitable
for small-scale software development. It runs with the help of the Java Development KIt.
For the report writing portion, i used Microsoft word as it is easy to use
and had all the features that were needed for the complication of the coursework.

Contents

List of Figures

List of Table

1. Introduction

The given coursework is designed for us to develop the graphical user interface (GUI) and add the functionality to the GUI for a certain vehicle company using the programming language called Java. The vehicle company GUI consists of booking, adding, and selling a system for Autorickshaw and Electric scooters. The GUI includes components like labels, text fields, buttons, combo box, etc while developing. Every button present in the GUI is assigned with different functionality which is required while booking or adding the vehicle for this coursework. The developed GUI is supposed to store the details of the vehicles including both Autorickshaw and electric scooters.

1.2 Java

Java is a programming language and a platform. Java is a high-level, robust, object-oriented, and secure programming language. ava was developed by *Sun Microsystems* (which is now the subsidiary of Oracle) in the year 1995. *James Gosling* is known as the father of Java. Before Java, its name was *Oak*. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java. (Java TPOINT, n.d.)

Java is used to create programs that run on a single device, such as a desktop or smartphone, and supports programs that run on a variety of platforms that support JRE. The creation of distributed applications using Java is another option. That implies that a single program can run synchronously while being spread among servers or clients in a network. As an additional component of web pages, Java can be used to create application modules or applets.

Java can be used for the following things:

- Web development
- Mobile application development
- GUI application development
- Middleware Application
- Embedded System
- Enterprises Application

1.3 Tools Required

Basically, very few numbers tools were required while doing this coursework

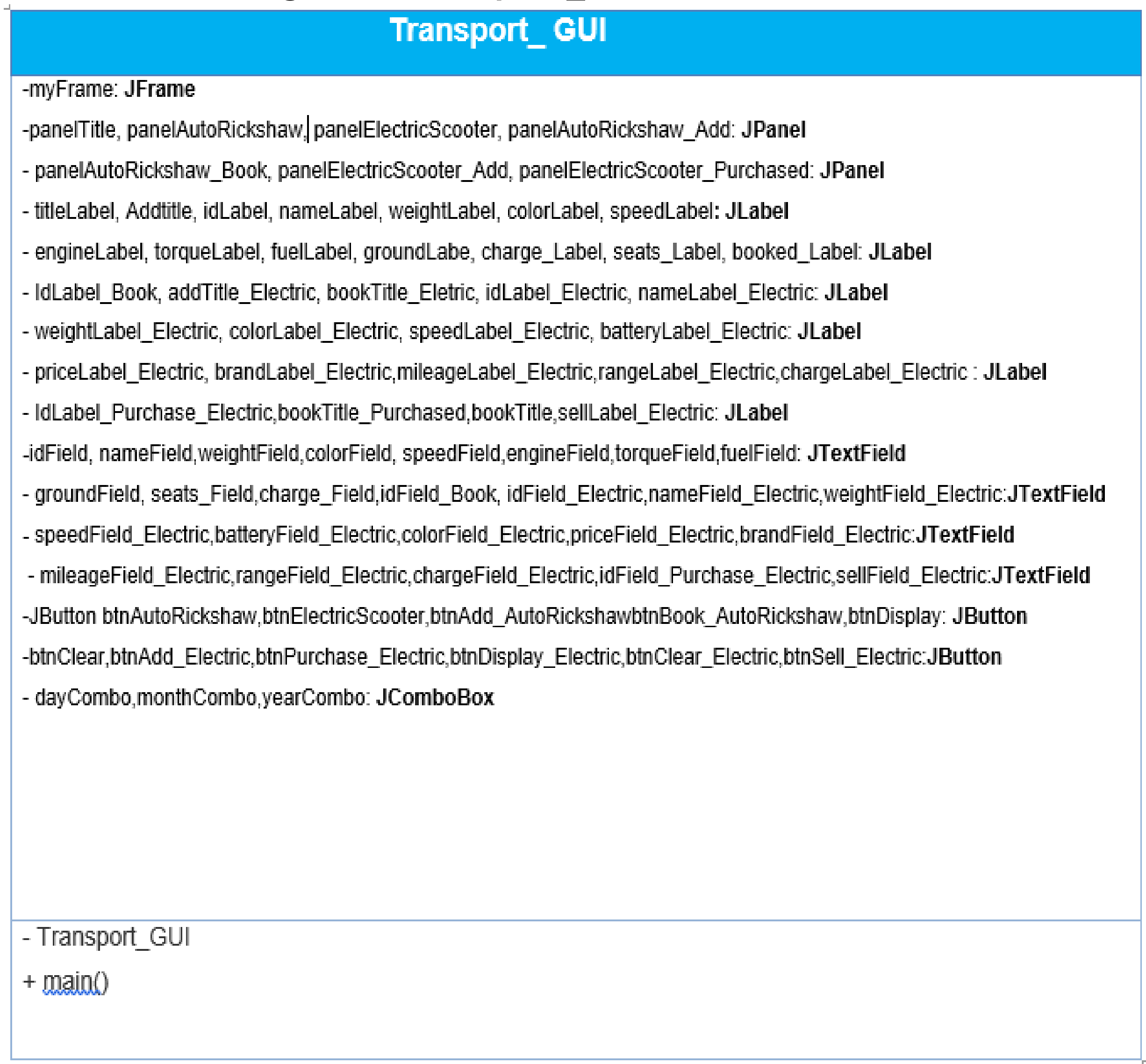
They are:

- BlueJ: Developed primarily for instructional reasons but also appropriate for small-scale software development, BlueJ is an integrated development environment (IDE) for the Java programming language. It is powered by the Java Development Kit (JDK) .
- Ms Word: In 1983, Microsoft created the word processing program known as Microsoft Word. It is the word processing program that is most frequently used. It is used to generate papers, letters, reports, resumes, etc. of a professional caliber and also enables you to edit or alter a new or existing document. The Word document has been saved.
- Moqups: Moqups is an online platform that integrates whiteboard, diagram, and design features into one visual collaboration tool.

2.1 Class Diagram

The characteristics and functions of a class are described in a class diagram, along with the restrictions placed on the system. Because they are the only UML diagrams that can be directly mapped with object-oriented languages, class diagrams are frequently employed in the modelling of object-oriented systems.

I. Class Diagram of Transport _GUI



Transport_ GUI

-myFrame: **JFrame**

-panelTitle, panelAutoRickshaw, panelElectricScooter, panelAutoRickshaw_Add: **JPanel**

- panelAutoRickshaw_Book, panelElectricScooter_Add, panelElectricScooter_Purchased: **JPanel**

- titleLabel, Addtitle, idLabel, nameLabel, weightLabel, colorLabel, speedLabel**: JLabel**

- engineLabel, torqueLabel, fuelLabel, groundLabe, charge_Label, seats_Label, booked_Label: **JLabel**

- IdLabel_Book, addTitle_Electric, bookTitle_Eletric, idLabel_Electric, nameLabel_Electric: **JLabel**

- weightLabel_Electric, colorLabel_Electric, speedLabel_Electric, batteryLabel_Electric: **JLabel**

- priceLabel_Electric, brandLabel_Electric,mileageLabel_Electric,rangeLabel_Electric,chargeLabel_Electric : **JLabel**

- IdLabel_Purchase_Electric,bookTitle_Purchased,bookTitle,sellLabel_Electric: **JLabel**

-idField, nameField,weightField,colorField, speedField,engineField,torqueField,fuelField: **JTextField**

- groundField, seats_Field,charge_Field,idField_Book, idField_Electric,nameField_Electric,weightField_Electric:**JTextField**

- speedField_Electric,batteryField_Electric,colorField_Electric,priceField_Electric,brandField_Electric:**JTextField**

- mileageField_Electric,rangeField_Electric,chargeField_Electric,idField_Purchase_Electric,sellField_Electric:**JTextField**

-JButton btnAutoRickshaw,btnElectricScooter,btnAdd_AutoRickshawbtnBook_AutoRickshaw,btnDisplay: **JButton**

-btnClear,btnAdd_Electric,btnPurchase_Electric,btnDisplay_Electric,btnClear_Electric,btnSell_Electric:**JButton**

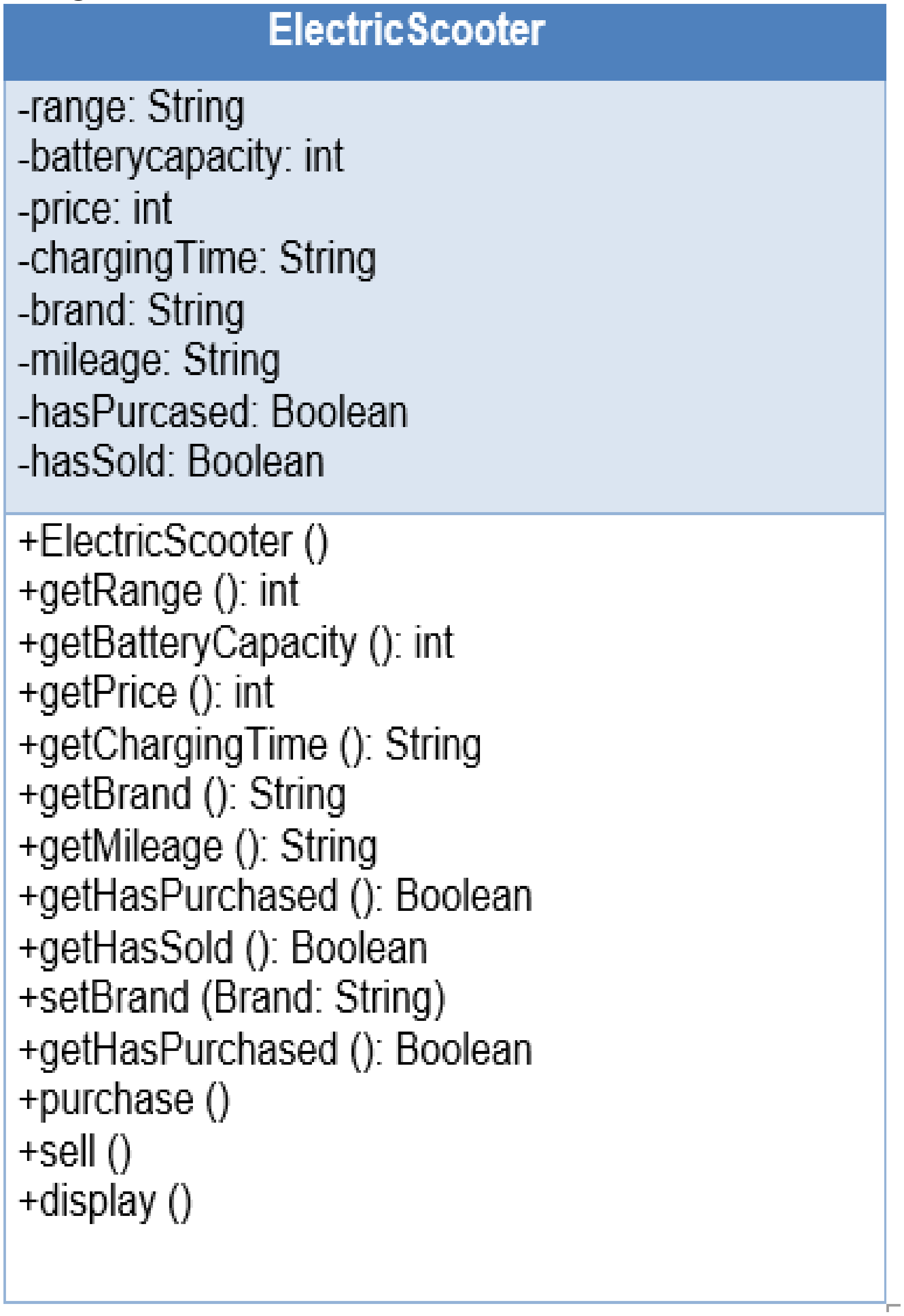
- dayCombo,monthCombo,yearCombo: **JComboBox**

- Transport_GUI

+ main()

Figure 1 CLASS Diagram Transport GUI

II.Class Diagram of class ElectricScooter

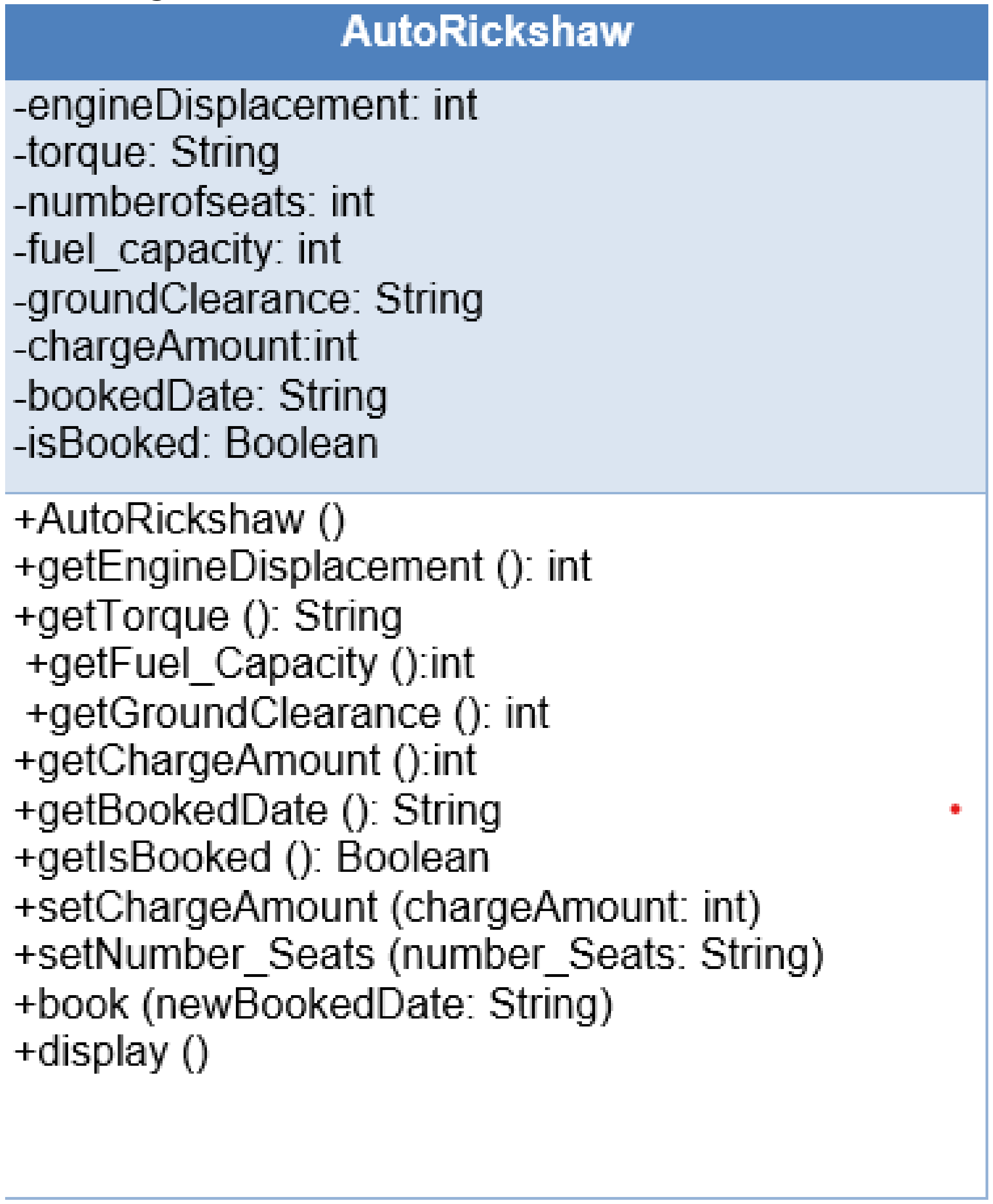


ElectricScooter

-range: String
-batterycapacity: int
-price: int
-chargingTime: String
-brand: String
-mileage: String
-hasPurcased: Boolean
-hasSold: Boolean

+ElectricScooter ()
+getRange (): int
+getBatteryCapacity (): int
+getPrice (): int
+getChargingTime (): String
+getBrand (): String
+getMileage (): String
+getHasPurchased (): Boolean
+getHasSold (): Boolean
+setBrand (Brand: String)
+getHasPurchased (): Boolean
+purchase ()
+sell ()
+display ()

I. Class Diagram of AutoRickshaw



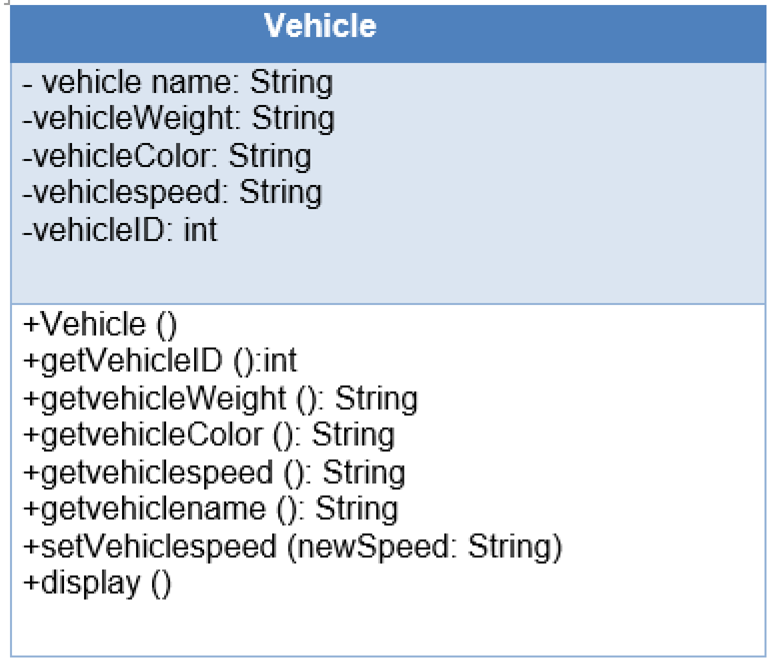
AutoRickshaw

-engineDisplacement: int
-torque: String
-numberofseats: int
-fuel_capacity: int
-groundClearance: String
-chargeAmount:int
-bookedDate: String
-isBooked: Boolean

+AutoRickshaw ()
+getEngineDisplacement (): int
+getTorque (): String
+getFuel_Capacity ():int
+getGroundClearance (): int
+getChargeAmount ():int
+getBookedDate (): String
+getIsBooked (): Boolean
+setChargeAmount (chargeAmount: int)
+setNumber_Seats (number_Seats: String)
+book (newBookedDate: String)
+display ()

Figure 2 Class diagram of AutoRickshaw

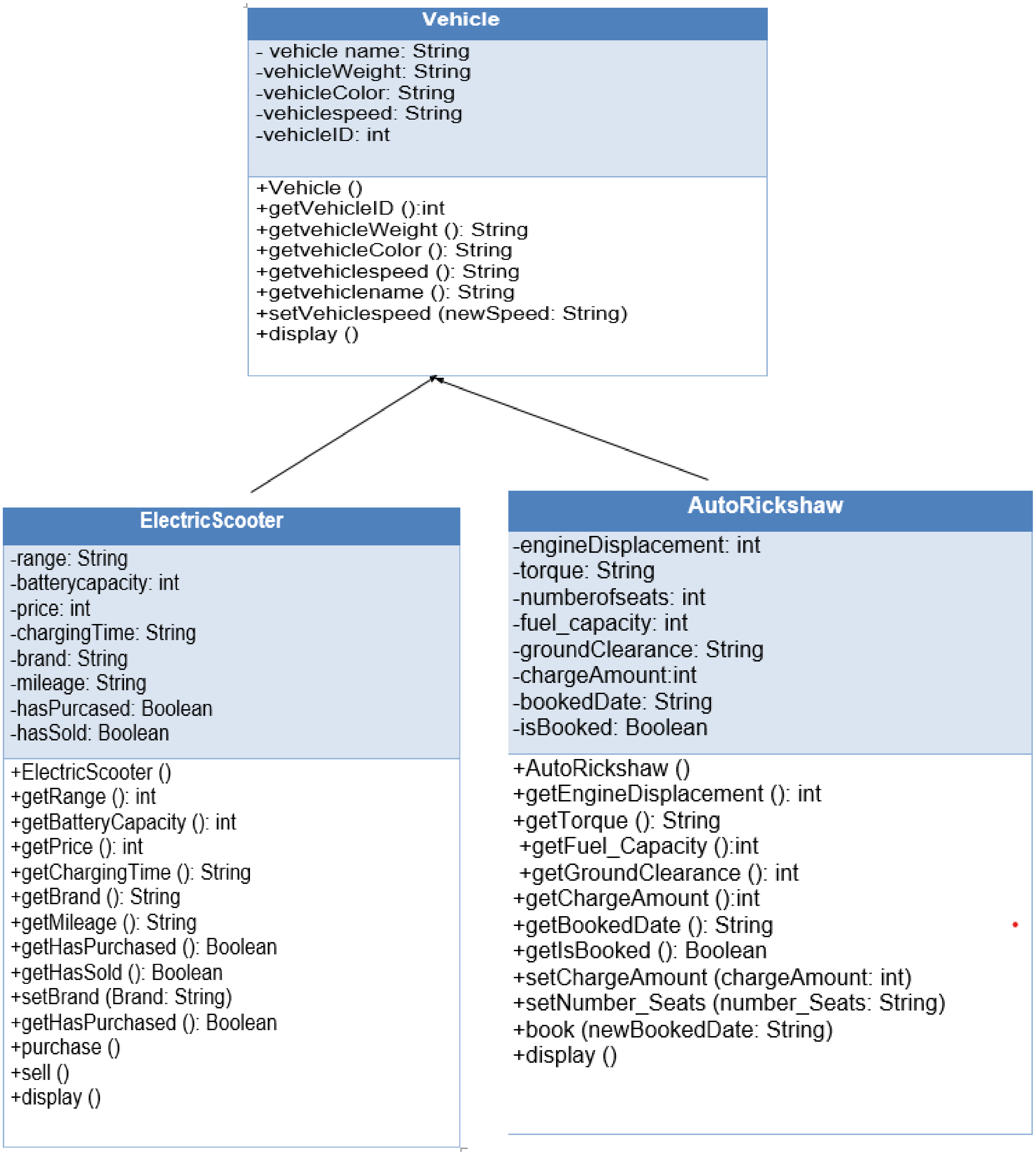
II. Class Diagram of class Vehicle



Vehicle
- vehicle name: String -vehicleWeight: String -vehicleColor: String -vehiclespeed: String -vehicleID: int
+Vehicle () +getVehicleID ():int +getvehicleWeight (): String +getvehicleColor (): String +getvehiclespeed (): String +getvehiclename (): String +setVehiclespeed (newSpeed: String) +display ()

Figure 3 Class Diagram of ElectricScooter

III. Class Diagram



3.1 Pseudocode

Pseudocode is a loose way of describing programming that doesn't need to adhere to any particular rules of syntax or underlying technology. It is used to develop a program's rough draft or blueprint. Pseudocode condenses a program's flow but omits supporting information. To make sure that programmers comprehend the specifications of a software project and align their code correctly, system designers create pseudocode.

Advantages of Pseudocode:

- Pseudocode is accessible to all types of programmers.
- It allows the programmer to focus exclusively on the algorithmic portion of the code creation.

Pseudocode for class Transport_GUI

```
IMPORT Packages in program

CREATE class Transport_GUI
DEFINE frame
DEFINE panel to book an AutoRickshaw, panel to add an AutoRickshaw, panel to
             add an ElectricScooter, panel to purchase an ElectricScooter
DEFINE Label for AutoRickshaw Add panel and book panel
DEFINE Label for ElectricScooter Add panel and Purchased panel
DEFINE Button for display, clear, add, book, sell and purchase
DEFINE Text Field for AutoRickshaw book panel and add panel
DEFINE Text Field for ElectricScooter add panel and purchase panel
DEFINE Combo Box for panel AutoRickshaw
DEFINE int variables for autoRickshaw and ElectricScooter
 DEFINE arrayList named autoRickshawList
DEFINE arrayList named ElectricScooterList
DEFINE object named autoRickshawObj and electricScooterOBJ

CREATE constructor named Transport_GUI
  CREATE frame and add its attributes
   CREATE panel for Title, panel for AutoRickshaw and panel for ElectricScooter
    CREATE panel for adding and booking AutoRickshaw and panel for adding and
                 Purchasing ElectricScooter
        DEFINE fonts
      CREATE Button for switching autoRickshaw and ElectricScooter
          CREATE buttons for panels
           CREATE labels for panels
            CREATE combo Box for panel AutoRickshaw
            CREATE text Field for Panels

    DEFINE actionPerformed method for AutoRickshaw Button
       HIDE ElectricScooter panel
       SHOW AutoRickshaw Panel
       CHANGE AutoRickshaw button background red and forecolor
        CHANGE the colour of the ElectricScooter button

    DEFINE Action Performed methods for ElectricScooter
       HIDE     AutoRickshaw Panel
       SHOW ElectricScooter Panel
       CHANGE AutoRickshaw button   Colour
       CHANGE ElectricScooter Button Colour

    DEFINE actionPerformed method for add Button of AutoRickshaw
      IF (GET all the values from AutoRickshaw add Text Field) is Empty
           SHOW (Empty Field is Found)
         ELSE

             IF (getvehicle_Id () == idField)
                SHOW (This vehicle is already added)
                 RETURN
             END IF
              DO
                TRY
                  CONVERT engineField and fuelField to int form
                CATCH
                   SHOW (Please Enter in int form)
                END DO
          CALL AutoRickshaw with input parameter
                  SHOW (Vehicle Successfully added)
```

```
DEFINE Action Performed for book button on AutoRickshaw panel
 IF (GET all the values from AutoRickshaw book Text Field) is Empty
    SHOW (Empty Field is Found)
 ELSE
        IF (autoRickshawList size is equal to 0)
            SHOW (Please Add a List First)
        ELSE
             DEFINE Boolean check
                SET check = false
        FOR I form 0 to AutoRickshawList.size
           IF AutoRickshawList.getVehicleID = entered vehicle id
          CALL book method from class AutoRickshaw
            SHOW (Vehicle booked Successfully
            ELSE
             SHOW (This Vehicle is already booked)
              END IF
           UPDATE check = true
              END IF
             IF check == false
                   SHOW (Vehicle doesn't exist)
            END IF
     SET all the field to empty
 END IF
```

```
DEFINE ActionPerformed on Display of panel AutoRickshaw
       FOR i in the range from 0 to AutoRickshawList.size
           CREATE Object of class AutoRickshaw using AutoRickshawList.i
                CALL display method form AutoRickshaw

DEFINE ActionPerformed on button clear of panel AutoRickshaw
            SET all the fields to empty

DEFINE ActionPerformed on button Add of panel ElectricScooter
    IF (GET all the values from the ElectricScooter Add Text Field) is Empty
         SHOW (Empty Field Found)
      ELSE
       DO
           TRY Id field in integer form
            CATCH SHOW (Vehicle id needed in Integer form)
             TRY battery capacity in integer form
              CATCH SHOW (Battery Capacity needed in Integer form)
         END DO
         FOR I in range from 0 to ElectricScooterList. Size
                CREATE object from ElectricScooter
                 IF (CALL method getVehicleID ()) == entered Id
                            SHOW (vehicle already exists)
                             SET id field to empty
                                    Return;
                 END IF
             CALL ElectricScooter with input parameter
    DEFINE ActionPerformed for button purchase of ElectricScooter
                IF (GET all the values from ElectricScooter Purchases Text Field) is
Empty
    SHOW (Empty Field is Found)
 ELSE
      IF (ElectricScooterList size is equal to 0)
         SHOW (Please Add a List First)
      ELSE
         DEFINE Boolean check
           SET check = false
      FOR I form 0 to ElectricScootrList.size
        IF ElectricScooterList. GetVehicleID = entered vehicle id
        CALL Purchased method from class ElectricScooter
         SHOW (Vehicle purchased Successfully
         ELSE
          SHOW (This Vehicle is already Purchased)
           END IF
        UPDATE check = true
           END IF
          IF check == false
               SHOW (Vehicle doesn't exist)
         END IF
    SET all the field to empty
 END IF
```

```
        DEFINE ActionPerformed for button Sell in Panel ElectricScooter
            IF (GET all the values from ElectricScooter sell Text Field) is Empty
      SHOW (Empty Field is Found)
  ELSE
        IF (ElectricScooterList size is equal to 0)
           SHOW (Please Add a List First)
        ELSE
            DEFINE Boolean check
              SET check = false
        FOR I form 0 to ElectricScootrList.size
          IF ElectricScooterList. GetVehicleID = entered vehicle id
          CALL Sell method from class ElectricScooter
           SHOW (Vehicle Sold Successfully
           ELSE
            SHOW (This Vehicle is already Sold)
             END IF
          UPDATE check = true
             END IF
            IF check == false
                 SHOW (Vehicle doesn't exist)
           END IF
     SET all the field to empty
  END IF
```

```
   DEFINE ActionPerformed for  button Display of panel ElectricScooter
            FOR i in the range from 0 to ElectricScooter. Size
      CREATE Object of class ElectricScooter using ElectricScooterList.i
           CALL display method form ElectricScooter
DEFINE ActionPerformed for button Clear of panel ElectricScooter
      SET all the field to Empty
```

4.1 Method Description

A method is a block of code that only runs when it is called. You can pass data, known as parameters, into a method. Methods are used to perform certain actions, and they are also known as functions**.** (W3.School, n.d.)

The methods included in class Transport_GUI are as follows:

4.1Constructor Method (Transport_GUI ())

All the elements needed for this program to run are in the constructor procedure. It includes GUI elements together with their features, such as font, boundaries, color, and layout. A frame with 7 panels (Title panel, panel AutoRickshaw Add and book panel for AutoRickshaw panel, panel ElectricScooter, Add and purchase the panel for ElectricScooter.). Choose buttons to alter the other two panels, which contain the components to add the classes. One panel is buried beneath another that is the same in terms of its bound attributes, and the AutoRickshaw and ElectricScooter buttons are in charge additional elements included in this Label and text field are the methods. This procedure includes an anonymous class to execute actions. Each button executes a distinct action using a distinct action listener as necessary for the coursework.

AutoRickshaw Button

This button enables the user to change the panel to AutoRickshaw. One panel is buried in another panel, when the user clicks the button the panel AutoRickshaw is set to visible.

ElectricScooter Button

This button enables the user to change the panel to ElectricScooter. One panel is buried in another panel, when the user clicks the button the panel ElectricScooter is set to visible.

Add Vehicle Button

The action listener on both AutoRickshaw and ElectricScooter panels for this button works similarly. When the button is pressed the action listener checks whether the field is empty or not. If the fields are not empty the action listener creates the object using the values which are given and added it to the ArrayList of the vehicle.

Book and purchase Button

Action listener of book button of panel AutoRickshaw has similar functions as an add button. The text field is checked, and an error message is displayed saying empty field found if the fields are empty and if no vehicles are added an error message saying no vehicle is added. A loop is run to check if the ArrayList has the vehicle having Id as given in Id field of book or purchase. If the id matches to the any id in array list message saying the vehicle is successfully booked is displayed and if the vehicle is already booked a message saying the vehicle is already booked is displayed.

Display Button

The action listener for the Display button displays the data from an array list into a table. Both the academic and non-academic panels respond to this button in a similar manner. The user sees an error message if the array list is empty. The display method is called the class AutoRickshaw or ElectricScooter.

Clear Button

The clear button is the most basic of all. This button's action listener displays a confirmation dialog to the user and, upon confirmation, clears all of the panel's text fields by assigning an empty string to each one.

Sell Button of panel ElectricScooter

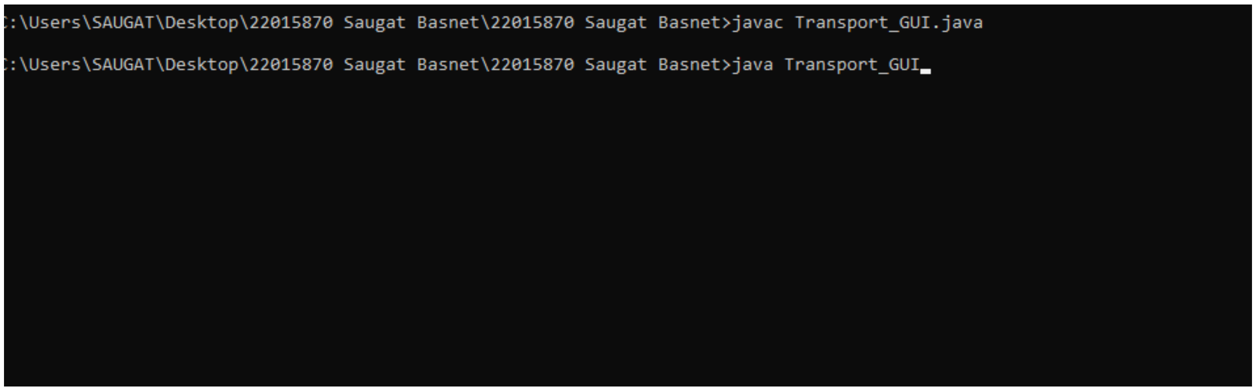
Action listener on sell button on panel ElectricScooter works very similar to the book and purchase button. But after when the fields are checked, the action listener calls the selling method from class ElectricScooter with parameter

5.Testing

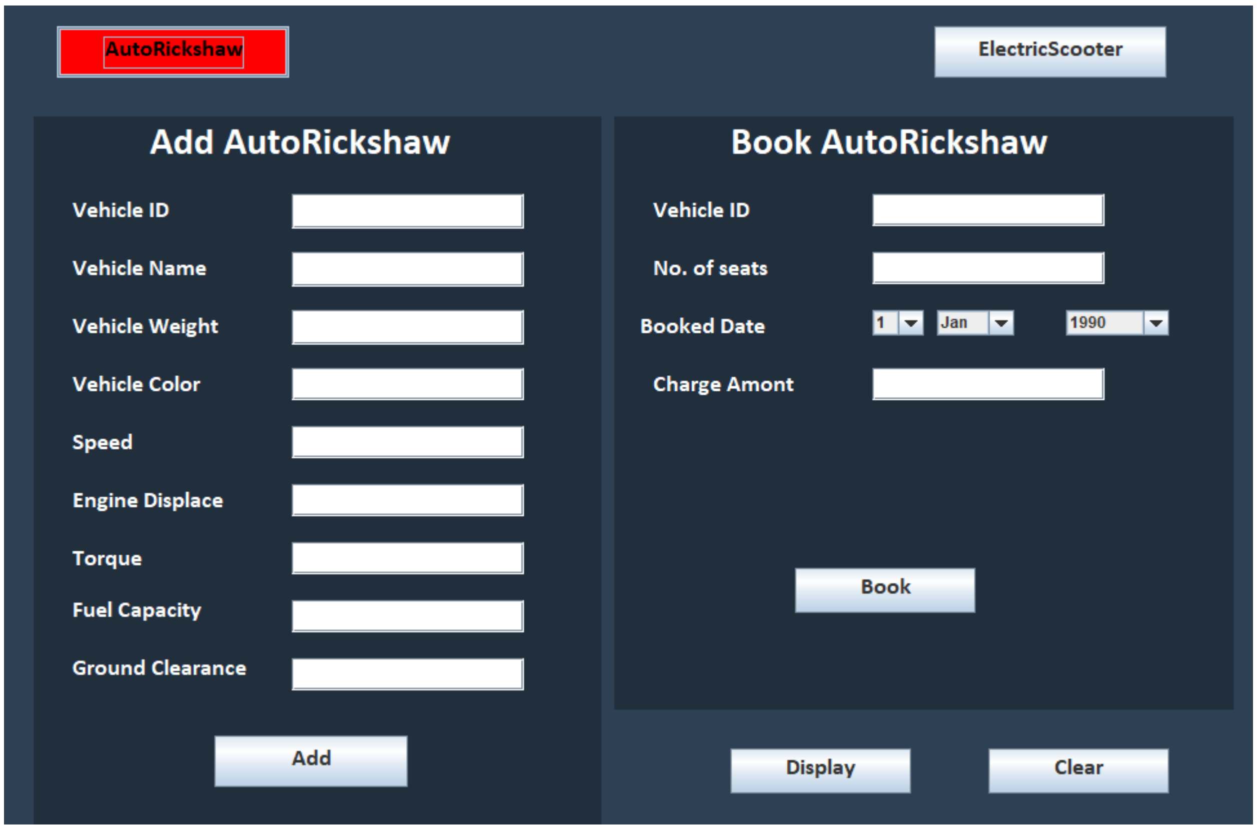
5.1Test 1

Table 1 Test 1 Table

TEST No.	1
Objectives	Test that the program can be compiled and run using the command prompt, including a screenshot like a Figure 1 from the command prompt learning aid.
Action	Open a command prompt on java file Program is compiled using Javac Transport_GUI.java Program is run by entering the command
Expected Result	Java should successfully run using cmd
Actual Result	Java was successfully run using cmd
Conclusion	Test Successfully



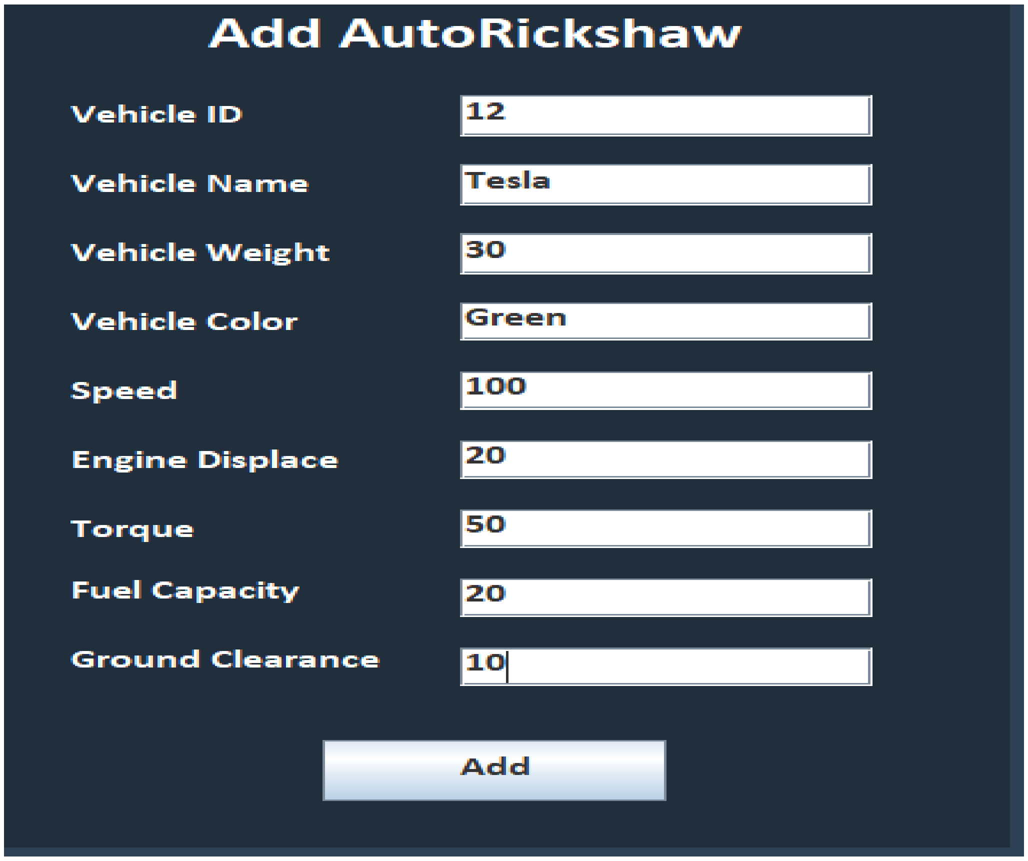
```
:\Users\SAUGAT\Desktop\22015870 Saugat Basnet\22015870 Saugat Basnet>javac Transport_GUI.java

:\Users\SAUGAT\Desktop\22015870 Saugat Basnet\22015870 Saugat Basnet>java Transport_GUI_
```



5.2 Test 2 a

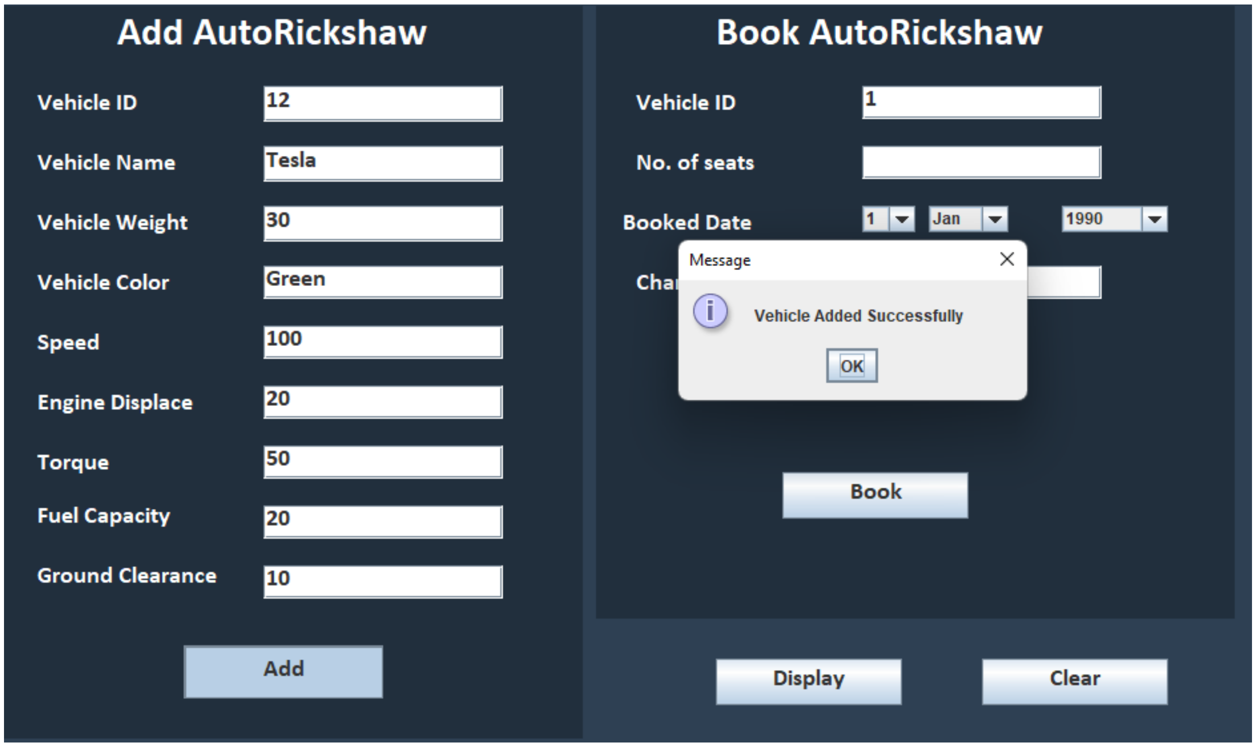
Table 2 Test 2 a Table

TEST No.	2
Objectives	Add the AutoRickshaw
Action	Values are inserted in text field of panel ADD AutoRickshaw Add button is clicked
Expected Result	Message showing vehicle Successfully booked is displayed
Actual Result	Vehicle Successfully booked is displayed
Conclusion	Test Successfully booked


Add AutoRickshaw

Vehicle ID	12
Vehicle Name	Tesla
Vehicle Weight	30
Vehicle Color	Green
Speed	100
Engine Displace	20
Torque	50
Fuel Capacity	20
Ground Clearance	10

Figure 4 Test 2a



Message

Vehicle Added Successfully

Vehicle ID	12
Vehicle Name	Tesla
Vehicle Weight	30
Vehicle Color	Green
Speed	100
Engine Displace	20
Torque	50
Fuel Capacity	20
Ground Clearance	10

Vehicle ID	1
No. of seats	
Booked Date	1 Jan 1990
Cha	

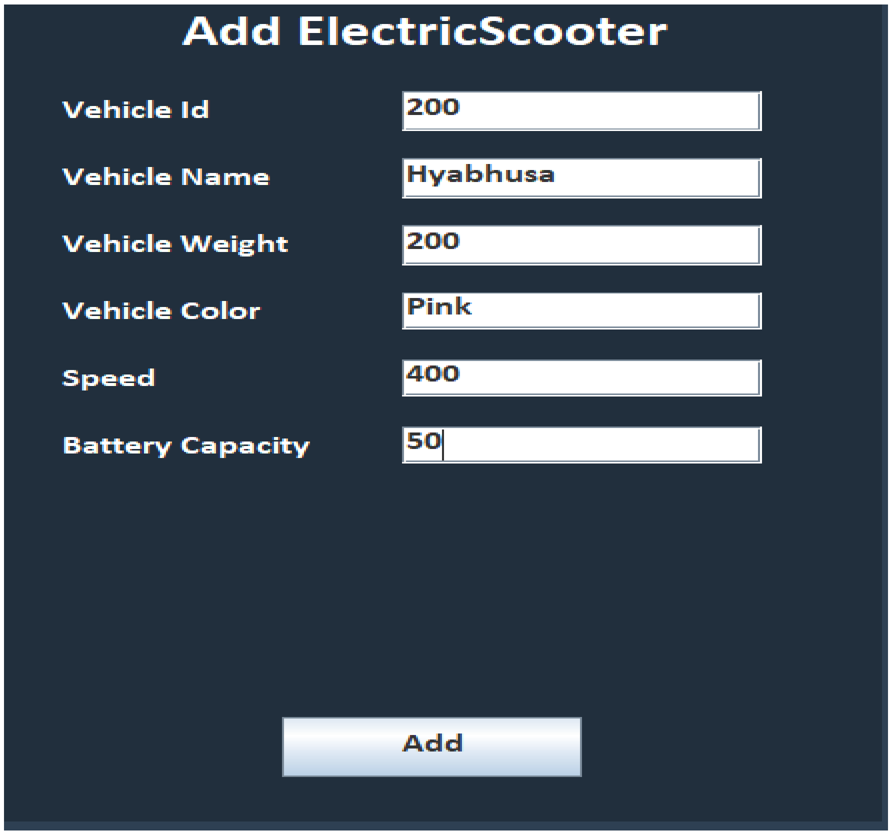
Figure 5 Test 2 a Result

5.3 Test 2.b

Table 3 Test 2 b Table

TEST No.	2.b
Objectives	Add the ElectricScooter
Action	Values are inserted in text field of panel ADD ElectricScooter Add button is clicked
Expected Result	Message showing vehicle Successfully booked is displayed
Actual Result	Vehicle Successfully booked is displayed
Conclusion	Test Successful

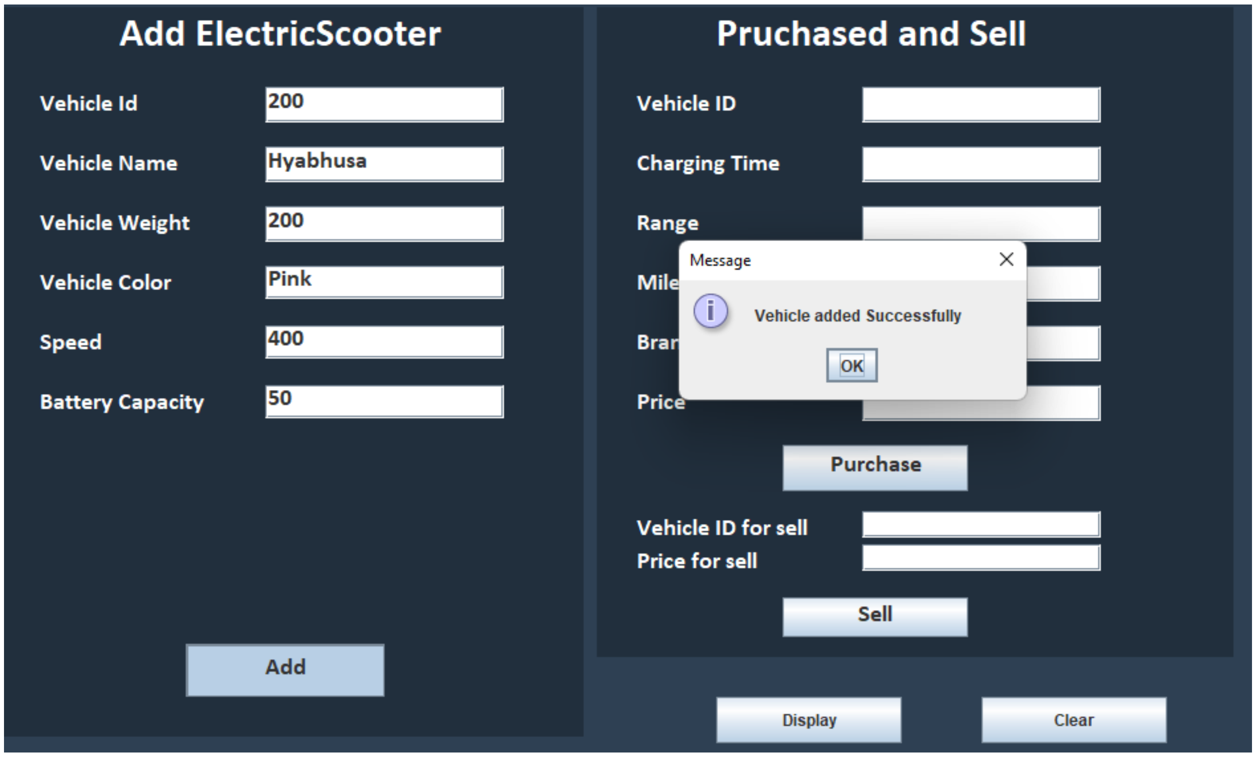
Figure 6 Test 2c



Add ElectricScooter

Vehicle Id	200
Vehicle Name	Hyabhusa
Vehicle Weight	200
Vehicle Color	Pink
Speed	400
Battery Capacity	50

Figure 7 Test 2b



Vehicle Id	200
Vehicle Name	Hyabhusa
Vehicle Weight	200
Vehicle Color	Pink
Speed	400
Battery Capacity	50

Vehicle ID	
Charging Time	
Range	
Mile	
Brar	
Price	
Vehicle ID for sell	
Price for sell	

Message

Vehicle added Successfully

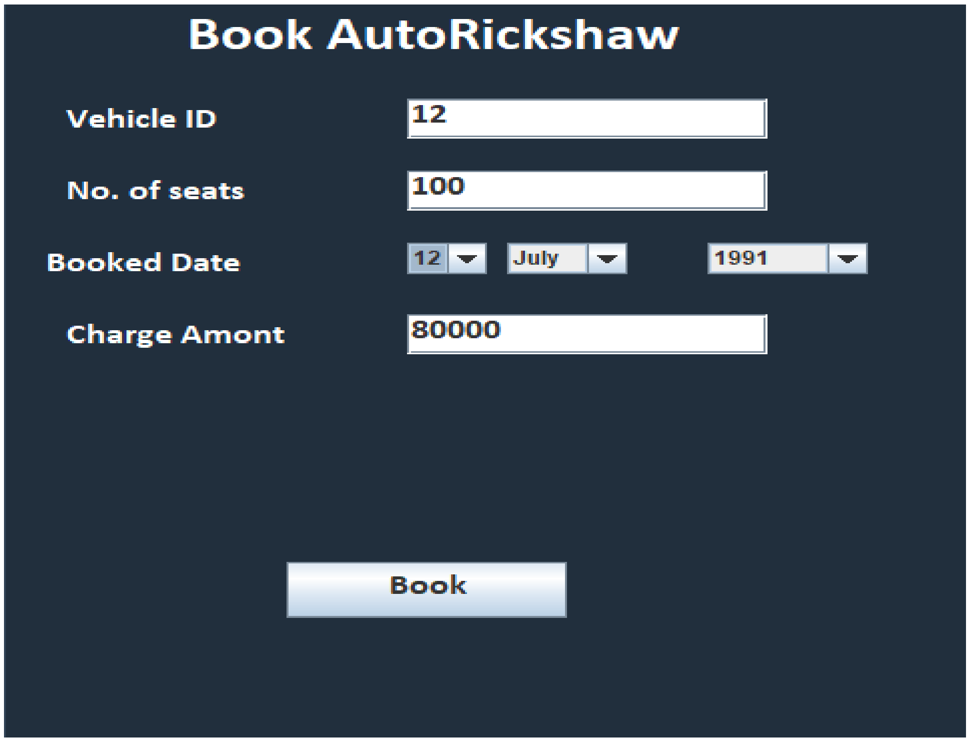
Figure 8 Test 2 b Result

5.4 Test 2 c

Table 4 Test 2 c Table

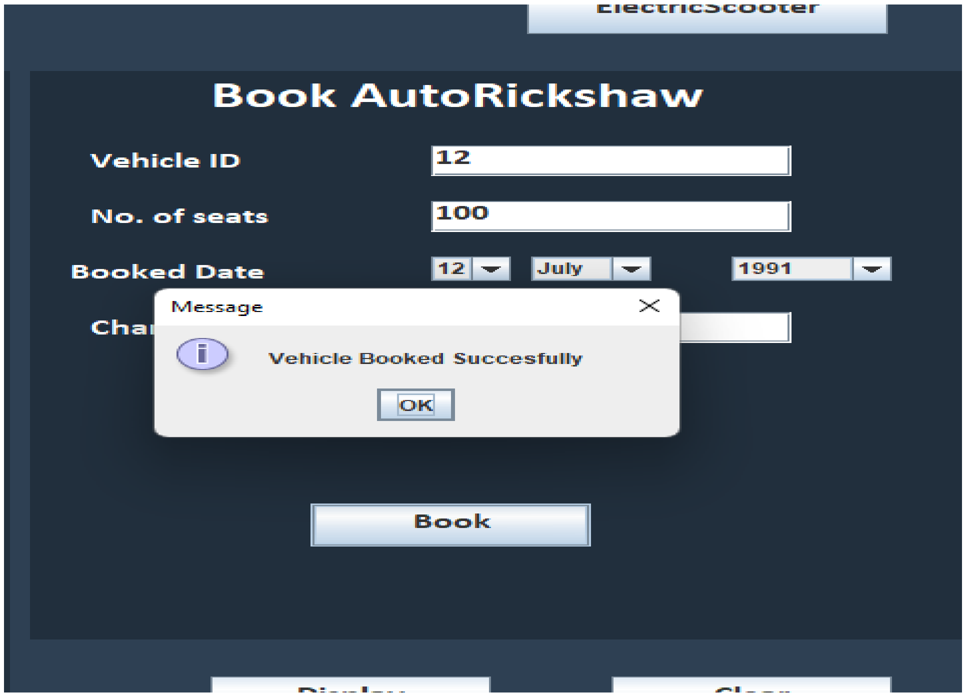
TEST No.	2 c
Objectives	Book the AutoRickshaw
Action	Values are inserted in the text field the of panel book AutoRickshaw Book Button is clicked
Expected Result	Message showing successfully booked

Actual Result	Message showing successfully booked is displayed
Conclusion	Test successful


Book AutoRickshaw

Vehicle ID: 12
No. of seats: 100
Booked Date: 12 July 1991
Charge Amont: 80000

Figure 9 Test 2c



Book AutoRickshaw

Vehicle ID: 12

No. of seats: 100

Booked Date: 12 July 1991

Message

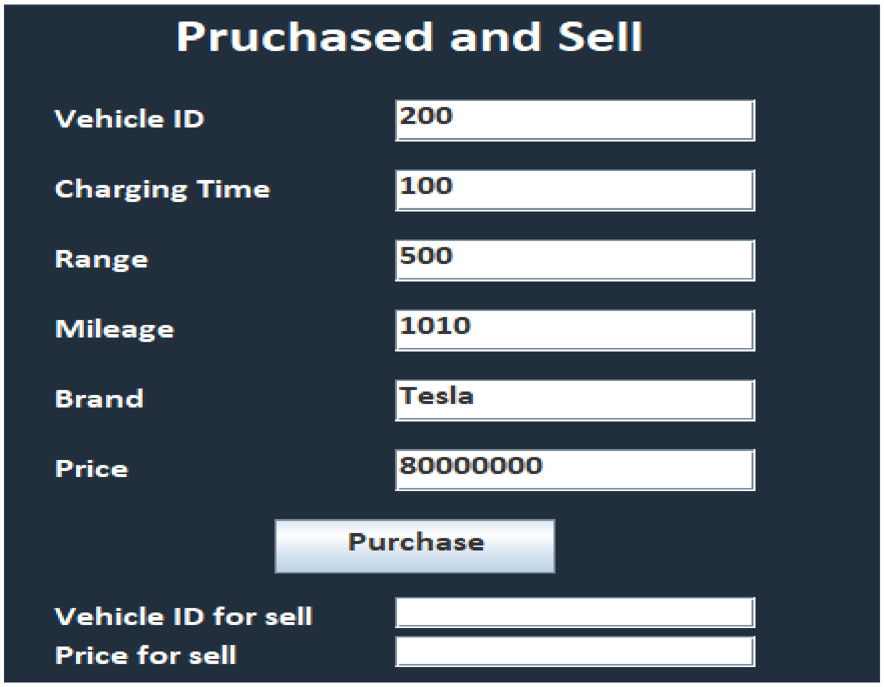
Vehicle Booked Succesfully

Figure 10 Test 2 c Result

5.5 Test 2 d

Table 5 Test 2 d table

TEST No.	2d
Objectives	Purchase the ElectricScooter
Action	Values are inserted in the text field the of panel purchase ElectricScooter The purchased button is clicked
Expected Result	Message showing successfully purchased is displayed
Actual Result	Message showing successfully purchased was displayed
Conclusion	Test Successful

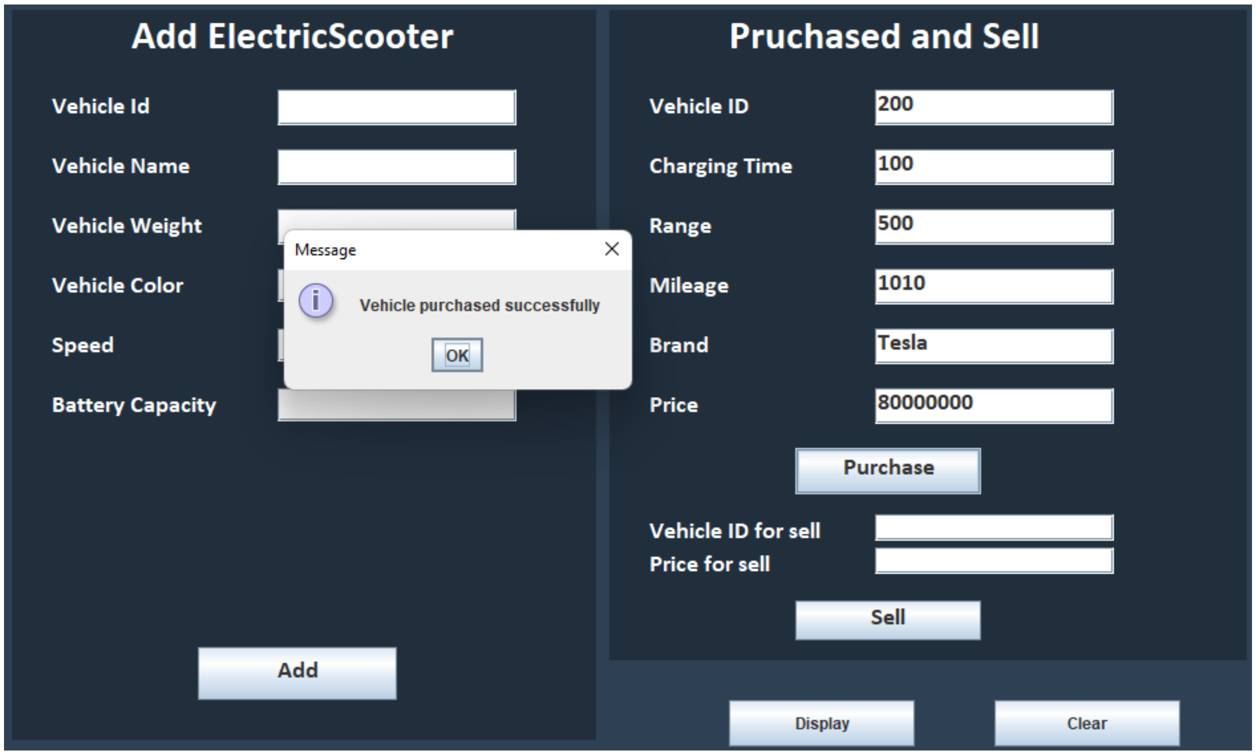

Pruchased and Sell

Vehicle ID	200
Charging Time	100
Range	500
Mileage	1010
Brand	Tesla
Price	80000000

Vehicle ID for sell	
Price for sell	

Figure 11 Test 2 d Result

(J.ECK)



Vehicle purchased successfully

Vehicle ID 200
Charging Time 100
Range 500
Mileage 1010
Brand Tesla
Price 80000000

Figure 12 Test 2 D

5.6 Test 2 e

Table 6 Test 2 e table

TEST No.	2 e
Objectives	Sell the ElectricScooter
Action	Values are inserted in the text field the of panel sell ElectricScooter The sell button is clicked
Expected Result	Message showing successfully Sold is displayed
Actual Result	Message showing successfully Sold was displayed
Conclusion	Test Successfully

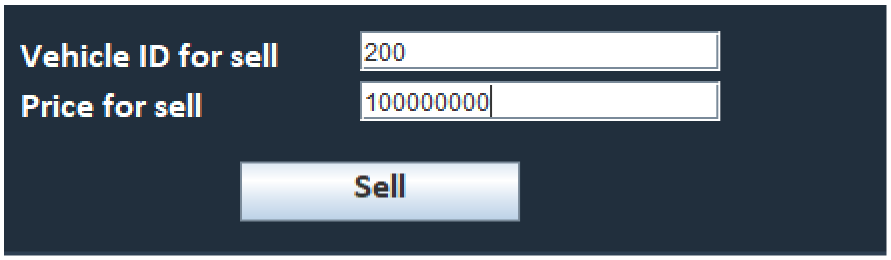
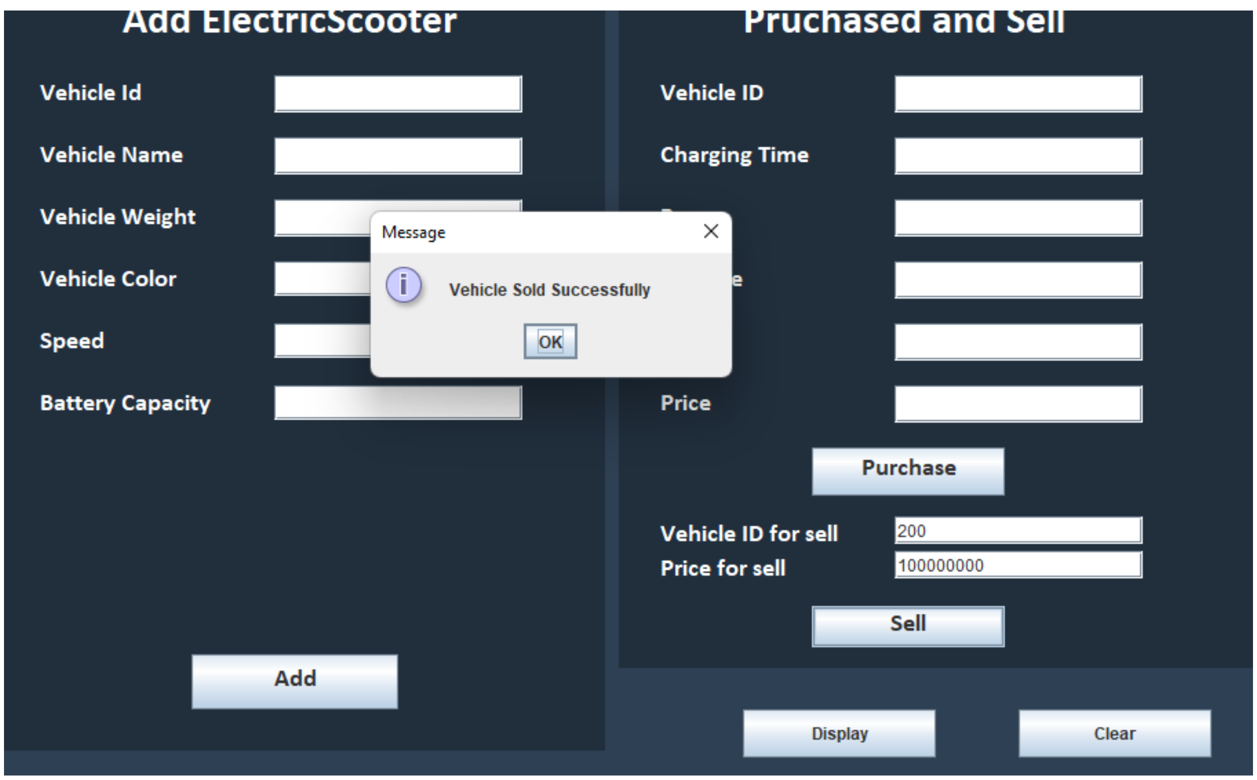

Vehicle ID for sell: 200
Price for sell: 100000000

Figure 13 Test 2 e



Vehicle ID for sell: 200
Price for sell: 100000000

Vehicle Sold Successfully

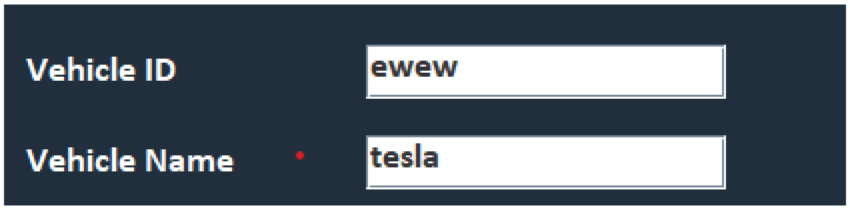
Figure 14 Test 2 e Result\

5.7 Test 3

Table 7 Test 3 Table

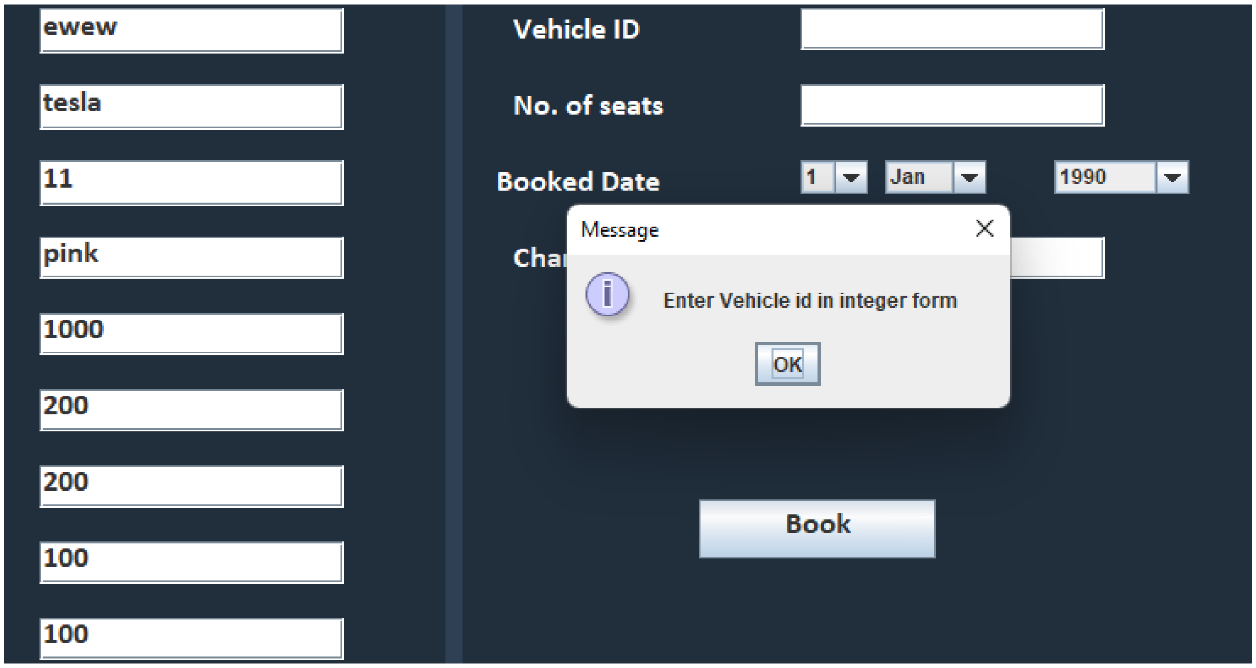
TEST No.	3
Objectives	Show dialog box message when False value in inserted
Action	False values were inserted
Expected Result	Message showing Invalid input should be displayed
Actual Result	Message showing invalid input was displayed

Conclusion	Test Successfully


Vehicle ID ewew

Vehicle Name tesla

Figure 15 Test 3


ewew

tesla

11

pink

1000

200

200

100

100

Vehicle ID

No. of seats

Booked Date 1 Jan 1990

Message

Enter Vehicle id in integer form

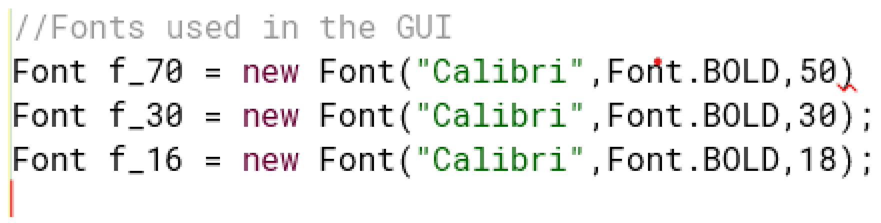
Figure 16Test 3 Result

6. Error Detection and Correction

The given course work of Java was different and more complicated from other course work and assignment which we had done previously. It was had long coding lines and tons of validation process, so facing a error was obvious. To solve such errors and mistake I took help from our module teacher Mr. Prabodh Tuladhar and my friends. The major faced errors and mistakes are syntax errors, semantic errors and logical errors:

6.1 Error 1: Syntax Error

The most common type of error is syntax error which occurs when a syntax and command is missing from the code which can generally be called as violation of the java rules.

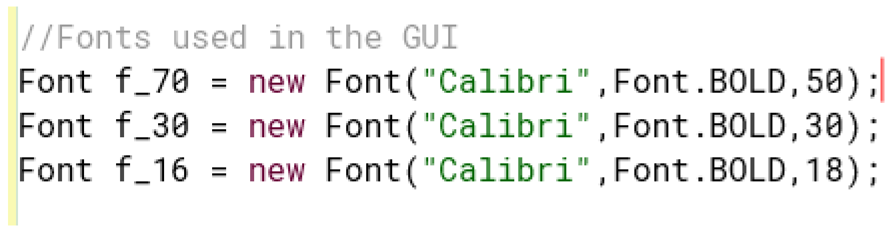


```
//Fonts used in the GUI
Font f_70 = new Font("Calibri",Font.BOLD,50)
Font f_30 = new Font("Calibri",Font.BOLD,30);
Font f_16 = new Font("Calibri",Font.BOLD,18);
```

Figure 17 Syntax Error

The error was occurred due to missing of semicolon from the code which violates the rules of java.

This error was corrected y adding the semicolon in the code where it is necessary.

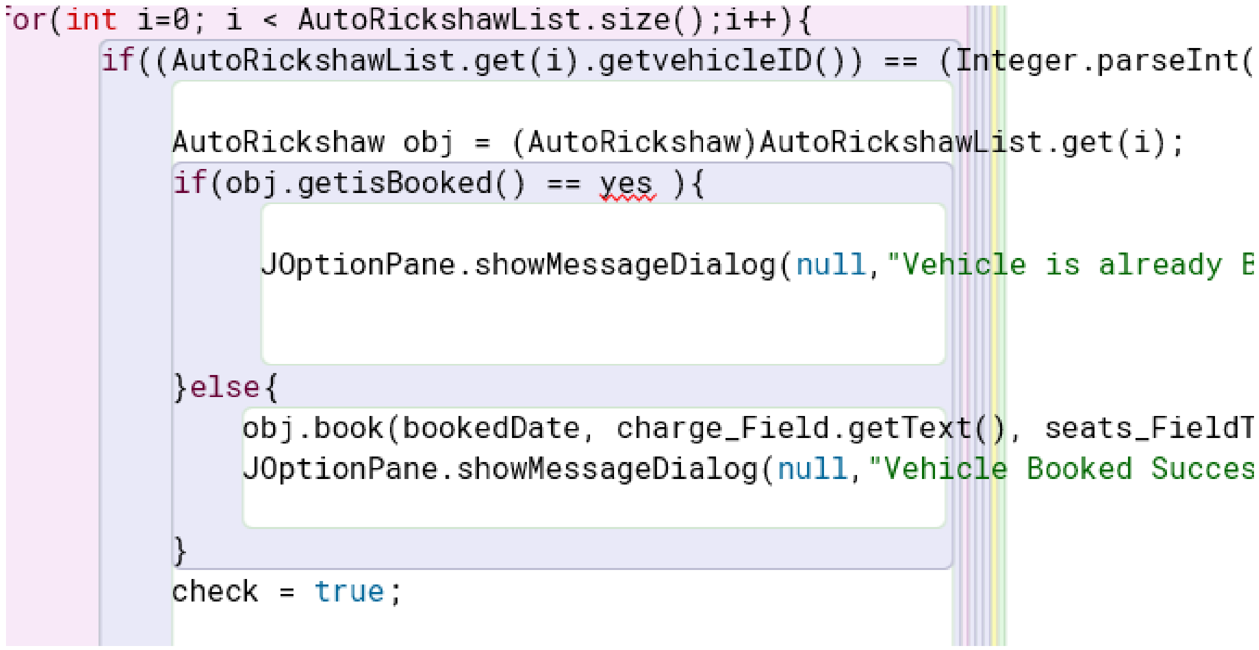


```
//Fonts used in the GUI
Font f_70 = new Font("Calibri",Font.BOLD,50);
Font f_30 = new Font("Calibri",Font.BOLD,30);
Font f_16 = new Font("Calibri",Font.BOLD,18);
```

Figure 18 Syntax Error Correction

6.2 Error 2. Semantic Error

Semantic error can be said as improper use of java command. This kind of error occurs when there is misuse of values in java code.

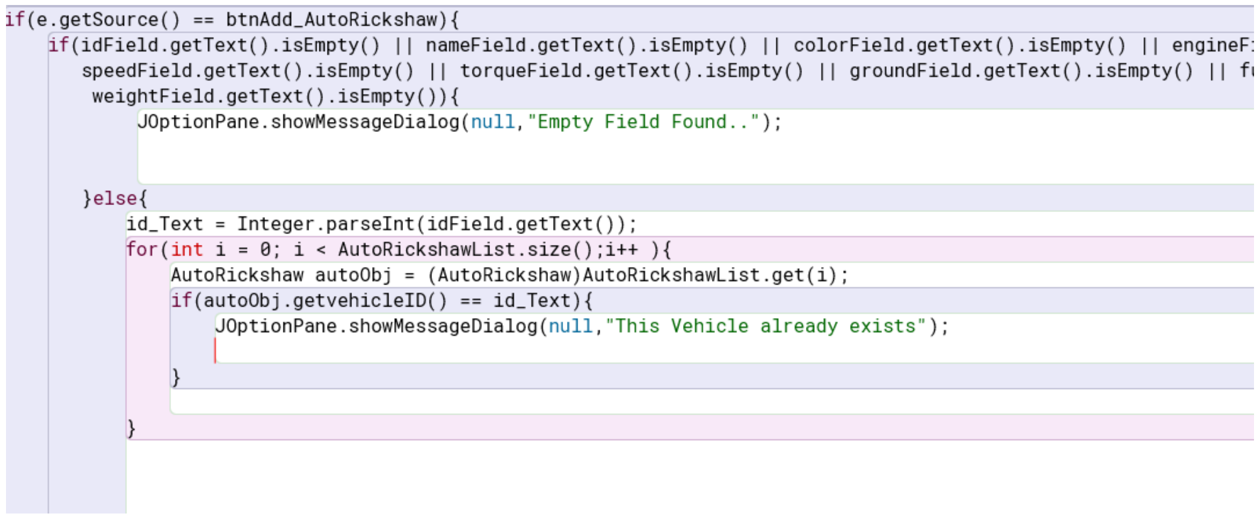


```
for(int i=0; i < AutoRickshawList.size();i++){
    if((AutoRickshawList.get(i).getvehicleID()) == (Integer.parseInt(

        AutoRickshaw obj = (AutoRickshaw)AutoRickshawList.get(i);
        if(obj.getisBooked() == yes ){

            JOptionPane.showMessageDialog(null,"Vehicle is already B


        }else{
            obj.book(bookedDate, charge_Field.getText(), seats_FieldT
            JOptionPane.showMessageDialog(null,"Vehicle Booked Succes

        }
        check = true;
```

Figure 19 Semantic Error

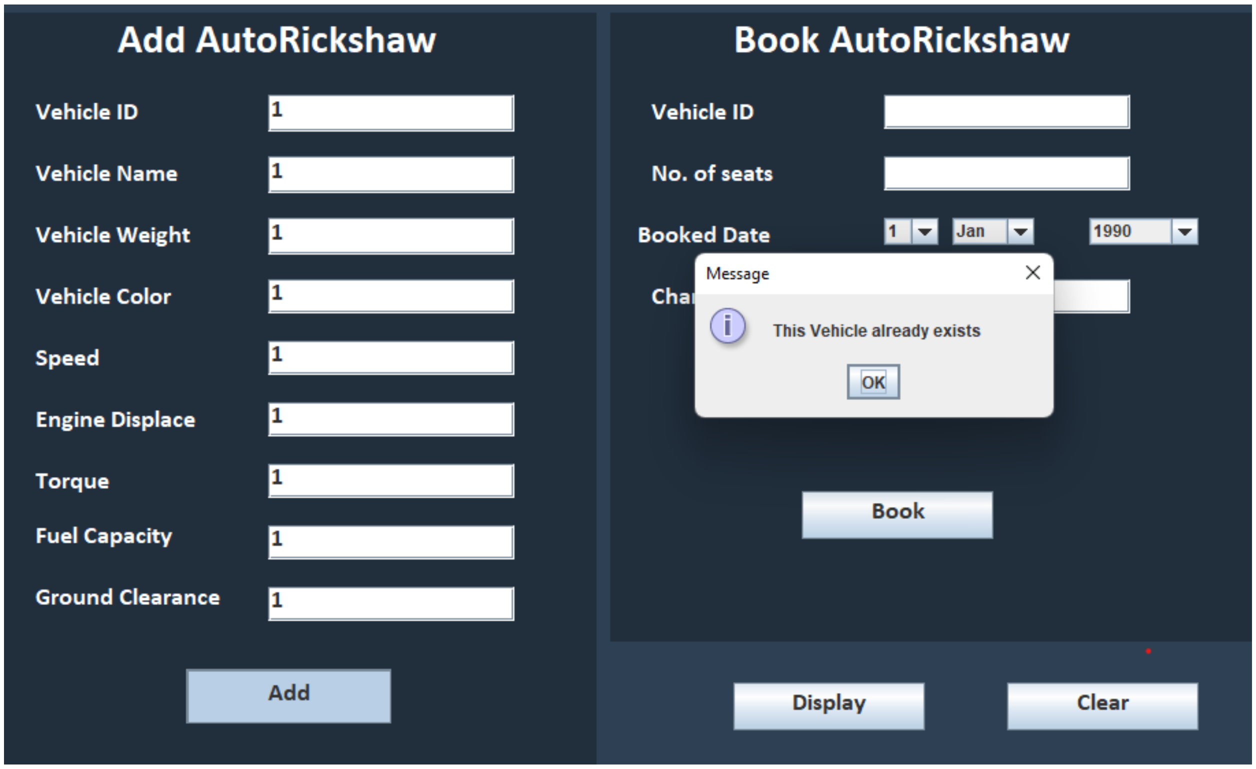
6.3 Error 3: Logical Error

A logic mistake occurs when your software compiles and runs, but performs the improper thing, provides an inaccurate result, or produces nothing when it should. Neither JVM nor the compiler can identify these mistakes.



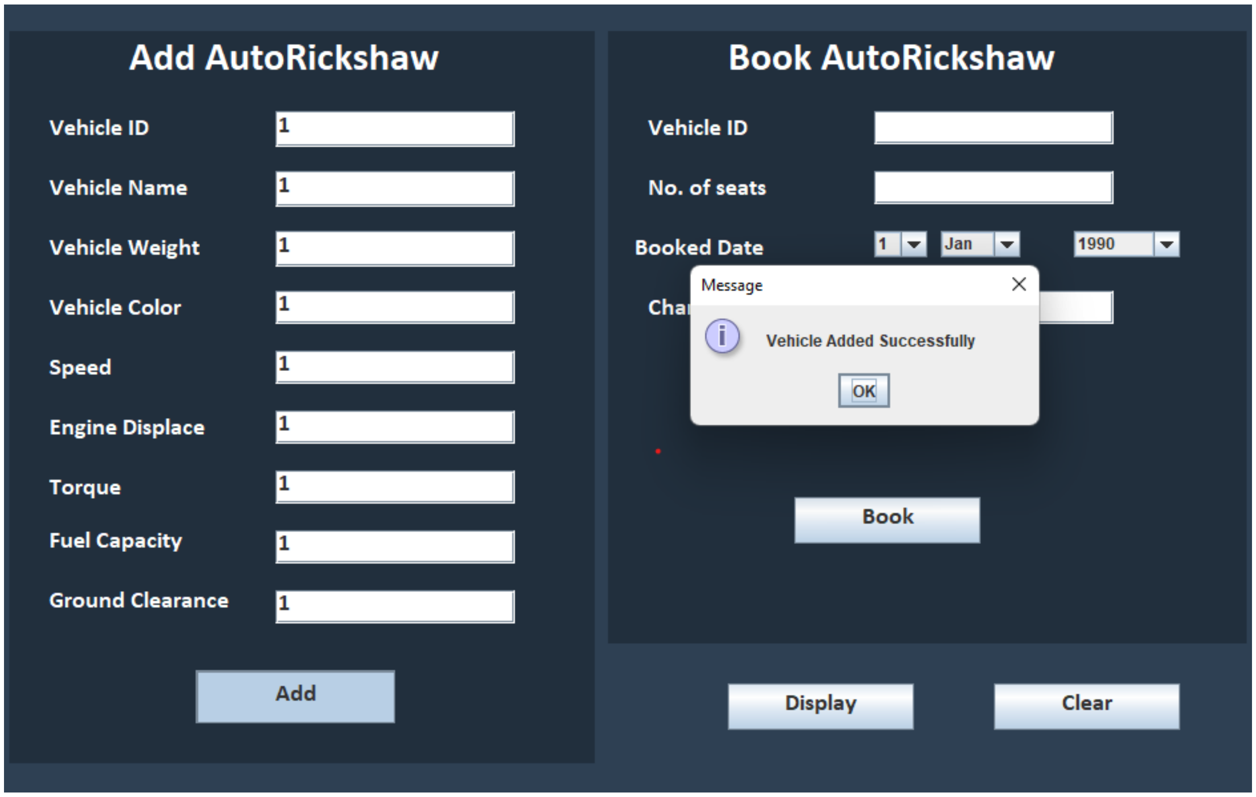
```
if(e.getSource() == btnAdd_AutoRickshaw){
    if(idField.getText().isEmpty() || nameField.getText().isEmpty() || colorField.getText().isEmpty() || engineF
       speedField.getText().isEmpty() || torqueField.getText().isEmpty() || groundField.getText().isEmpty() || fu
        weightField.getText().isEmpty()){
            JOptionPane.showMessageDialog(null,"Empty Field Found..");


    }else{
        id_Text = Integer.parseInt(idField.getText());
        for(int i = 0; i < AutoRickshawList.size();i++ ){
            AutoRickshaw autoObj = (AutoRickshaw)AutoRickshawList.get(i);
            if(autoObj.getvehicleID() == id_Text){
                JOptionPane.showMessageDialog(null,"This Vehicle already exists");

            }

        }
```

Figure 20 Logical Error



This Vehicle already exists

Figure 21 Logical Error 2



Vehicle ID: 1
Vehicle Name: 1
Vehicle Weight: 1
Vehicle Color: 1
Speed: 1
Engine Displace: 1
Torque: 1
Fuel Capacity: 1
Ground Clearance: 1

Message

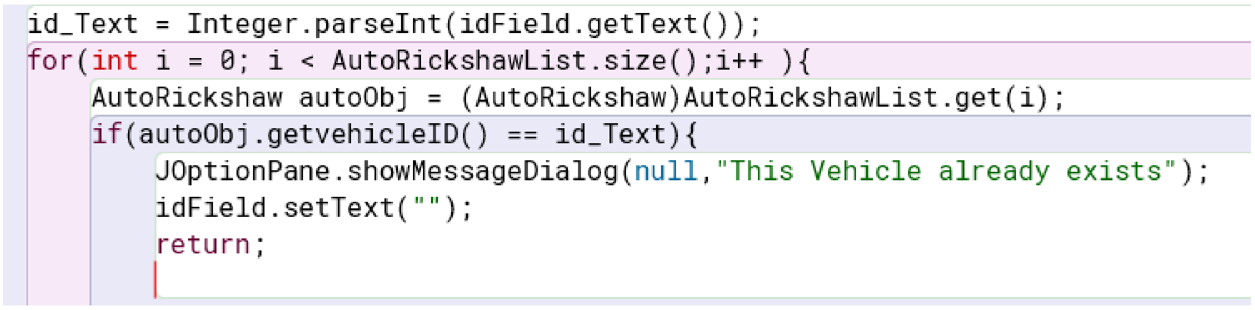
Vehicle Added Successfully

OK

Figure 22 Logical Error 3

Correction of Logical Error

The error occurred because when the vehicle is already added but after displaying the vehicle has already been added a message was shown saying the vehicle was successfully booked because of the logical error.



```
id_Text = Integer.parseInt(idField.getText());
for(int i = 0; i < AutoRickshawList.size();i++ ){
    AutoRickshaw autoObj = (AutoRickshaw)AutoRickshawList.get(i);
    if(autoObj.getvehicleID() == id_Text){
        JOptionPane.showMessageDialog(null,"This Vehicle already exists");
        idField.setText("");
        return;
```

Figure 23 Logical Error Correction

7. Conclusion

The GUI of the class vehicle was successfully developed before the deadline of the coursework submission. This course helped me to learn and work with different components and functions of the programming language Java. Our module teacher and my friends played an important role in helping with the development of the GUI for the class Vehicle.

The GUI contains two panels named AutoRickshaw panel and ElectricScooter panel. Inside both panels, there are two panels developed one for adding vehicles and the other for purchasing or booking. Two buttons named as ElectricScooter button and the AutoRickshaw button were developed which on click switches the panel AutoRickshaw and ElectricScooter. According to the question different Text Fields, labels, and buttons were added inside the panel which performed different functions. The add button inside the panel is used for adding the vehicle to the ArrayList with inserted information on the vehicle. The book and purchase button performs a similar task in the GUI by booking or purchasing the vehicle which is already added to the list. The display button is used to display all the information about the vehicle.

With the completion of all the tasks, the coursework was submitted on time without any problems and errors, although the coursework was a bit hard and took a long time. But this has helped me to learn various things and work with different components of java and solve the real-time problem that occurs in the real world. I am grateful to our module teacher for this.

8. Appendix

8.1 Transport_GUI

```
//Importing the packages


import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.awt.Font;
import java.awt.Color;
import java.awt.BorderLayout;
import java.util.ArrayList;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import javax.swing.JLabel;

/**
 * @author (Lonmetid Firstname Lastname)
 * @version (1.0.0)
 *
 */



public class Transport_GUI implements ActionListener{
   //Defining the UI components in the class Vehicle
   private JFrame myFrame;
   private JPanel
panelTitle,panelAutoRickshaw,panelElectricScooter,panelAutoRickshaw_Add,

panelAutoRickshaw_Book,panelElectricScooter_Add,panelElectricScooter_Purchased;
```

```
private JLabel titleLabel,Addtitle,
            idLabel,nameLabel,weightLabel,
            colorLabel,speedLabel,engineLabel,
            torqueLabel,fuelLabel,groundLabel
            ,charge_Label,seats_Label,booked_Label,idLabel_Book,
            addTitle_Electric,bookTitle_Eletric,




            idLabel_Electric,nameLabel_Electric,weightLabel_Electric,
            colorLabel_Electric,speedLabel_Electric,batteryLabel_Electric,
            priceLabel_Electric,brandLabel_Electric,mileageLabel_Electric,
            rangeLabel_Electric,chargeLabel_Electric,idLabel_Purchase_Electric,
            bookTitle_Purchased,bookTitle,sellLabel_Electric,priceSell_Label
            ;
private JTextField
             idField,nameField,weightField,colorField,
             speedField,engineField,torqueField,fuelField,
             groundField,seats_Field,charge_Field,idField_Book,

             idField_Electric,nameField_Electric,weightField_Electric,
             speedField_Electric,batteryField_Electric,colorField_Electric,
             priceField_Electric,brandField_Electric,mileageField_Electric,
             rangeField_Electric,chargeField_Electric,idField_Purchase_Electric,
             sellField_Electric,priceSell_Field



             ;
```

```
private JButton btnAutoRickshaw,btnElectricScooter,btnAdd_AutoRickshaw,
       btnBook_AutoRickshaw,btnDisplay,btnClear,btnAdd_Electric
       ,btnPurchase_Electric,btnDisplay_Electric,btnClear_Electric,btnSell_Electric
       ;


private JComboBox dayCombo,monthCombo,yearCombo;

ArrayList<Vehicle> AutoRickshawList = new ArrayList<Vehicle>();
 ArrayList<Vehicle> ElectricScooterList = new ArrayList<Vehicle>();


 int fuelTxt,engineTxt,id_Text,id_Electric,battery_Electric,
      seats_FieldTxt,rangeTxt_Electric,priceTxt_Electric,priceSell_Txt
     ;


 Vehicle AutoRickshawObj,ElectricScooterObj;








Transport_GUI(){

    //Frame
    myFrame= new JFrame("Vehicle Details");
    myFrame.setLayout(null);
    myFrame.setVisible(true);
    myFrame.setSize(1000, 720);
    myFrame.setResizable(false);//Unable to change the size of frame
    myFrame.getContentPane().setBackground(Color.decode("#2E4053"));

    //Panel for Title
    panelTitle = new JPanel();
    panelTitle.setBounds(0,0,980,70);
    panelTitle.setLayout(null);
    panelTitle.setBackground(Color.decode("#2E4053"));
    myFrame.add(panelTitle);

    //Panel for AutoRickshaw
    panelAutoRickshaw = new JPanel();
    panelAutoRickshaw.setBounds(22,50,950,600);
    panelAutoRickshaw.setLayout(null);
    panelAutoRickshaw.setVisible(true);
    panelAutoRickshaw.setBackground(Color.decode("#2E4053"));
    myFrame.add(panelAutoRickshaw);
      //Panel to Add an AutoRickshaw
      panelAutoRickshaw_Add = new JPanel();
      panelAutoRickshaw_Add.setBounds(10,40,440,550);
      panelAutoRickshaw_Add.setLayout(null);
      panelAutoRickshaw_Add.setBackground(Color.decode("#212F3D"));
      panelAutoRickshaw.add(panelAutoRickshaw_Add);
      //Panel to Book an Autorickshaw
      panelAutoRickshaw_Book = new JPanel();
      panelAutoRickshaw_Book.setBounds(460,40,480,460);
      panelAutoRickshaw_Book.setLayout(null);
      panelAutoRickshaw_Book.setBackground(Color.decode("#212F3D"));
      panelAutoRickshaw.add(panelAutoRickshaw_Book);
```

```
//Fonts used in the GUI
Font f_70 = new Font("Calibri",Font.BOLD,50);
Font f_30 = new Font("Calibri",Font.BOLD,30);
Font f_16 = new Font("Calibri",Font.BOLD,18);

//Button for TitlePanel
btnAutoRickshaw =new JButton("AutoRickshaw");
btnAutoRickshaw.setBounds(50,20,180,40);
btnAutoRickshaw.setFont(f_16);
btnAutoRickshaw.setLayout(null);
btnAutoRickshaw.addActionListener(this);
panelTitle.add(btnAutoRickshaw);

btnElectricScooter = new JButton("ElectricScooter");
btnElectricScooter.setBounds(730,20,180,40);
btnElectricScooter.setFont(f_16);
btnElectricScooter.setLayout(null);
btnElectricScooter.addActionListener(this);
panelTitle.add(btnElectricScooter);




//Title label add
Addtitle = new JLabel("Add AutoRickshaw");
Addtitle.setBounds(90,10,300,30);
Addtitle.setFont(f_30);
Addtitle.setForeground(Color.white);
panelAutoRickshaw_Add.add(Addtitle);
```

```
                                    //AutoRickshaw Add
    idLabel = new JLabel("Vehicle ID");
    idField = new JTextField();
    idLabel.setBounds(30,60,120,30);
    idField.setBounds(200,60,180,27);
    idLabel.setFont(f_16);
    idField.setFont(f_16);
    idLabel.setForeground(Color.white);
    panelAutoRickshaw_Add.add(idLabel);
    panelAutoRickshaw_Add.add(idField);

    //Vehcile Name
    nameLabel = new JLabel("Vehicle Name");
    nameField = new JTextField();
    nameLabel.setBounds(30,105,120,30);
    nameField.setBounds(200,105,180,27);
    nameLabel.setFont(f_16);
    nameField.setFont(f_16);
    nameLabel.setForeground(Color.white);
    panelAutoRickshaw_Add.add(nameLabel);
    panelAutoRickshaw_Add.add(nameField);

    // Vehicle Weight
    weightLabel = new JLabel("Vehicle Weight");
    weightField = new JTextField();
    weightLabel.setBounds(30,150,120,30);
    weightField.setBounds(200,150,180,27);
    weightLabel.setForeground(Color.white);
    weightLabel.setFont(f_16);
```

```
weightField.setFont(f_16);
panelAutoRickshaw_Add.add(weightLabel);
panelAutoRickshaw_Add.add(weightField);


colorLabel = new JLabel("Vehicle Color");
colorField = new JTextField();
colorLabel.setBounds(30,195,100,30);
colorField.setBounds(200,195,180,25);
colorLabel.setFont(f_16);
colorField.setFont(f_16);
colorLabel.setForeground(Color.white);
panelAutoRickshaw_Add.add(colorLabel);
panelAutoRickshaw_Add.add(colorField);



//Vehicle Speed
speedLabel = new JLabel("Speed");
speedField = new JTextField();
speedLabel.setBounds(30,240,100,30);
speedField.setBounds(200,240,180,25);
speedLabel.setFont(f_16);
speedField.setFont(f_16);
speedLabel.setForeground(Color.white);
panelAutoRickshaw_Add.add(speedLabel);
panelAutoRickshaw_Add.add(speedField);

//vehicle Engine
engineLabel = new JLabel("Engine Displace");
engineField = new JTextField();
```

```
engineLabel.setBounds(30,285,150,30);
engineField.setBounds(200,285,180,25);
engineLabel.setFont(f_16);
engineField.setFont(f_16);
engineLabel.setForeground(Color.white);
panelAutoRickshaw_Add.add(engineLabel);
 panelAutoRickshaw_Add.add(engineField);

//Vehicle Torque
torqueLabel = new JLabel("Torque");
torqueField = new JTextField();
torqueLabel.setBounds(30,330,100,30);
torqueField.setBounds(200,330,180,25);
torqueLabel.setFont(f_16);
torqueField.setFont(f_16);
torqueLabel.setForeground(Color.white);
panelAutoRickshaw_Add.add(torqueLabel);
panelAutoRickshaw_Add.add(torqueField);

fuelLabel  = new JLabel("Fuel Capacity");
fuelField = new JTextField();
fuelLabel.setBounds(30,375,100,20);
fuelField.setBounds(200,375,180,25);
fuelLabel.setFont(f_16);
fuelField.setFont(f_16);
fuelLabel.setForeground(Color.white);
panelAutoRickshaw_Add.add(fuelLabel);
panelAutoRickshaw_Add.add(fuelField);

// Ground clearance of the vehicle
groundLabel =  new JLabel("Ground Clearance");
```

```
 groundField = new JTextField();
 groundLabel.setBounds(30,420,150,20);
 groundField.setBounds(200,420,180,25);
 groundLabel.setFont(f_16);
 groundField.setFont(f_16);
 groundLabel.setForeground(Color.white);
 panelAutoRickshaw_Add.add(groundLabel);
 panelAutoRickshaw_Add.add(groundField);

 //Add Button in AutoRickshaw
btnAdd_AutoRickshaw = new JButton("Add");
btnAdd_AutoRickshaw.setBounds(140,480,150,40);
btnAdd_AutoRickshaw.setFont(f_16);
btnAdd_AutoRickshaw.setFocusable(false);
btnAdd_AutoRickshaw.addActionListener(this);
panelAutoRickshaw_Add.add( btnAdd_AutoRickshaw);




 //Booking part in the vehicle GUI
idLabel_Book = new JLabel("Vehicle ID");
idField_Book = new JTextField();
idLabel_Book.setBounds(30,60,120,30);
idField_Book.setBounds(200,60,180,25);
idLabel_Book.setFont(f_16);
idField_Book.setFont(f_16);
idLabel_Book.setForeground(Color.WHITE);
panelAutoRickshaw_Book.add(idLabel_Book);
panelAutoRickshaw_Book.add(idField_Book);
```

```
    //Number of Seats
    seats_Label = new JLabel("No. of seats");
    seats_Field = new JTextField();
    seats_Label.setBounds(30,105,120,30);
    seats_Field.setBounds(200,105,180,25);
    seats_Label.setFont(f_16);
    seats_Field.setFont(f_16);
    seats_Label.setForeground(Color.white);
    panelAutoRickshaw_Book.add(seats_Label);
    panelAutoRickshaw_Book.add(seats_Field);

    //Booking date
      //booking Label
       booked_Label =new JLabel("Booked Date");
       booked_Label.setBounds(20,150,120,30);
       booked_Label.setFont(f_16);
       booked_Label.setForeground(Color.white);
       panelAutoRickshaw_Book.add(booked_Label);

       //Year Combo Box
       String year[] ={"1990","1991"};
       yearCombo = new JComboBox(year);
       yearCombo.setBounds(350,150,80,20);
       panelAutoRickshaw_Book.add(yearCombo);

       //Month Combo Box
       String month[] =
{"Jan","Feb","March","April","Jun","July","August","September","Octuber","November"};
       monthCombo = new JComboBox(month);
       monthCombo.setBounds(250,150,60,20);
    panelAutoRickshaw_Book.add(monthCombo);

    //Day Combo Box
    String day[] = {"1","2","3","4","5","6","7","8","9","10","12"};
    dayCombo = new JComboBox(day);
    dayCombo.setBounds(200,150,40,20);
    panelAutoRickshaw_Book.add(dayCombo);


  //Booking button
  btnBook_AutoRickshaw =new JButton("Book");
  btnBook_AutoRickshaw.setBounds(140,350,140,35);
  btnBook_AutoRickshaw.setFont(f_16);
  btnBook_AutoRickshaw.addActionListener(this);
  panelAutoRickshaw_Book.add(btnBook_AutoRickshaw);

  //Charge Amount for Booking

  charge_Label = new JLabel("Charge Amont");
  charge_Field = new JTextField();
  charge_Label.setBounds(30,195,120,30);
  charge_Field.setBounds(200,195 ,180,25);
  charge_Label.setFont(f_16);
  charge_Field.setFont(f_16);
  charge_Label.setForeground(Color.white);
  panelAutoRickshaw_Book.add(charge_Label);
  panelAutoRickshaw_Book.add(charge_Field);


  //Title of the Book panel
  bookTitle = new JLabel("Book AutoRickshaw");
```

```
bookTitle.setBounds(90,10,300,30);
bookTitle.setFont(f_30);
bookTitle.setForeground(Color.white);
panelAutoRickshaw_Book.add(bookTitle);

//Display Button
btnDisplay = new JButton("Display");
btnDisplay.setBounds(550,530,140,35);
btnDisplay.addActionListener(this);
panelAutoRickshaw.add(btnDisplay);

//Clear Button
btnClear = new JButton("Clear");
btnClear.setBounds(750,530,140,35);
btnClear.addActionListener(this);
panelAutoRickshaw.add(btnClear);



//For ElectricScooter
panelElectricScooter  = new JPanel();
panelElectricScooter.setBounds(22,50,950,600);
panelElectricScooter.setLayout(null);
panelElectricScooter.setVisible(false);
panelElectricScooter.setBackground(Color.decode("#2E4053"));
myFrame.add(panelElectricScooter);

  //Panel Add for ElectricScooter
  panelElectricScooter_Add = new JPanel();
  panelElectricScooter_Add.setBounds(10,40,440,550);
  panelElectricScooter_Add.setLayout(null);
```

```
 panelElectricScooter_Add.setBackground(Color.decode("#212F3D"));
 panelElectricScooter.add(panelElectricScooter_Add);
 //Panel to Book an Autorickshaw
 panelElectricScooter_Purchased = new JPanel();
 panelElectricScooter_Purchased .setBounds(460,40,480,490);
 panelElectricScooter_Purchased .setLayout(null);
 panelElectricScooter_Purchased .setBackground(Color.decode("#212F3D"));
 panelElectricScooter.add(panelElectricScooter_Purchased );

//Component inside the Add panle in electricScooter



idLabel_Electric = new JLabel("Vehicle Id");
idField_Electric = new JTextField();
idLabel_Electric.setBounds(30,60,120,30);
idField_Electric.setBounds(200,60,180,27);
idLabel_Electric.setFont(f_16);
idField_Electric.setFont(f_16);
idLabel_Electric.setForeground(Color.white);
panelElectricScooter_Add.add(idLabel_Electric);
panelElectricScooter_Add.add(idField_Electric);

//Vehcile Name
nameLabel_Electric = new JLabel("Vehicle Name");
nameField_Electric = new JTextField();
nameLabel_Electric.setBounds(30,105,120,30);
nameField_Electric.setBounds(200,105,180,27);
nameLabel_Electric.setFont(f_16);
nameField_Electric.setFont(f_16);
nameLabel_Electric.setForeground(Color.white);
```

```
panelElectricScooter_Add.add(nameLabel_Electric);
panelElectricScooter_Add.add(nameField_Electric);

// Vehicle Weight
weightLabel_Electric = new JLabel("Vehicle Weight");
weightField_Electric= new JTextField();
weightLabel_Electric.setBounds(30,150,120,30);
weightField_Electric.setBounds(200,150,180,27);
weightLabel_Electric.setForeground(Color.white);
weightLabel_Electric.setFont(f_16);
weightField_Electric.setFont(f_16);
panelElectricScooter_Add.add(weightLabel_Electric);
panelElectricScooter_Add.add(weightField_Electric);


colorLabel_Electric = new JLabel("Vehicle Color");
colorField_Electric = new JTextField();
colorLabel_Electric.setBounds(30,195,100,30);
colorField_Electric.setBounds(200,195,180,25);
colorLabel_Electric.setFont(f_16);
colorField_Electric.setFont(f_16);
colorLabel_Electric.setForeground(Color.white);
panelElectricScooter_Add.add(colorLabel_Electric);
panelElectricScooter_Add.add(colorField_Electric);



//Vehicle Speed
speedLabel_Electric = new JLabel("Speed");
speedField_Electric = new JTextField();
speedLabel_Electric.setBounds(30,240,100,30);
```

```
 speedField_Electric.setBounds(200,240,180,25);
 speedLabel_Electric.setFont(f_16);
 speedField_Electric.setFont(f_16);
 speedLabel_Electric.setForeground(Color.white);
 panelElectricScooter_Add.add(speedLabel_Electric);
 panelElectricScooter_Add.add(speedField_Electric);

 //vehicle Engine
 batteryLabel_Electric = new JLabel("Battery Capacity");
 batteryField_Electric = new JTextField();
 batteryLabel_Electric.setBounds(30,285,150,30);
 batteryField_Electric.setBounds(200,285,180,25);
 batteryLabel_Electric.setFont(f_16);
 batteryField_Electric.setFont(f_16);
 batteryLabel_Electric.setForeground(Color.white);
 panelElectricScooter_Add.add(batteryLabel_Electric);
 panelElectricScooter_Add.add(batteryField_Electric);

 //Title in ElectricScooter Add
addTitle_Electric = new JLabel("Add ElectricScooter");
addTitle_Electric.setBounds(90,10,300,30);
addTitle_Electric.setFont(f_30);
addTitle_Electric.setForeground(Color.white);
panelElectricScooter_Add.add(addTitle_Electric);

 //Button for Adding ElectricScooter in panel add of electric Scooter
btnAdd_Electric = new JButton("Add");
btnAdd_Electric.setBounds(140,480,150,40);
btnAdd_Electric.setFont(f_16);
btnAdd_Electric.setFocusable(false);
btnAdd_Electric.addActionListener(this);
```

```
panelElectricScooter_Add.add( btnAdd_Electric);




//Purchased and selling in Electric Scooter
 mileageLabel_Electric= new JLabel("Mileage");
mileageField_Electric = new JTextField();
mileageLabel_Electric.setBounds(30,195,120,30);
mileageField_Electric.setBounds(200,195,180,27);
mileageLabel_Electric.setFont(f_16);
mileageField_Electric.setFont(f_16);
mileageLabel_Electric.setForeground(Color.white);
panelElectricScooter_Purchased.add(mileageLabel_Electric);
panelElectricScooter_Purchased.add(mileageField_Electric);

//range of Electric SCOOTER
rangeLabel_Electric = new JLabel("Range");
rangeField_Electric = new JTextField();
rangeLabel_Electric.setBounds(30,150,120,30);
rangeField_Electric.setBounds(200,150,180,27);
rangeLabel_Electric.setFont(f_16);
rangeField_Electric.setFont(f_16);
rangeLabel_Electric.setForeground(Color.white);
panelElectricScooter_Purchased.add(rangeLabel_Electric);
panelElectricScooter_Purchased.add(rangeField_Electric);

//Charge Time of Electric
chargeLabel_Electric = new JLabel("Charging Time");
chargeField_Electric = new JTextField();
chargeLabel_Electric.setBounds(30,105,120,30);
```

```
chargeField_Electric.setBounds(200,105,180,27);
chargeLabel_Electric.setFont(f_16);
chargeField_Electric.setFont(f_16);
chargeLabel_Electric.setForeground(Color.white);
panelElectricScooter_Purchased.add(chargeLabel_Electric);
panelElectricScooter_Purchased.add(chargeField_Electric);



//VehicleID for purchasing the electric Scooter
idLabel_Purchase_Electric= new JLabel("Vehicle ID");
idField_Purchase_Electric  = new JTextField();
idLabel_Purchase_Electric.setBounds(30,60,120,30);
idField_Purchase_Electric.setBounds(200,60,180,27);
idLabel_Purchase_Electric.setFont(f_16);
idField_Purchase_Electric.setFont(f_16);
idLabel_Purchase_Electric.setForeground(Color.white);
panelElectricScooter_Purchased.add(idLabel_Purchase_Electric);
panelElectricScooter_Purchased.add(idField_Purchase_Electric);


//Brand of Electric Vehicle
brandLabel_Electric = new JLabel("Brand  ");
brandField_Electric = new JTextField();
brandLabel_Electric.setBounds(30,240,120,30);
brandField_Electric.setBounds(200,240,180,27);
brandLabel_Electric.setFont(f_16);
brandField_Electric.setFont(f_16);
brandLabel_Electric.setForeground(Color.WHITE);
panelElectricScooter_Purchased.add(brandLabel_Electric);
panelElectricScooter_Purchased.add(brandField_Electric);
```

```
// Price of of the Electric Scooter
priceLabel_Electric = new JLabel("Price");
priceField_Electric = new JTextField();
priceLabel_Electric.setBounds(30,285,120,30);
priceField_Electric.setBounds(200,285,180,27);
priceLabel_Electric.setFont(f_16);
priceField_Electric.setFont(f_16);
priceLabel_Electric.setForeground(Color.WHITE);
panelElectricScooter_Purchased.add(priceLabel_Electric);
panelElectricScooter_Purchased.add(priceField_Electric);

// Purchase Button of Electric Scooter
btnPurchase_Electric =new JButton("Purchase");
btnPurchase_Electric.setBounds(140,330,140,35);
btnPurchase_Electric.setFont(f_16);
btnPurchase_Electric.setLayout(null);
btnPurchase_Electric.addActionListener(this);

panelElectricScooter_Purchased.add(btnPurchase_Electric);

//Sell of ElectricScooter
  sellLabel_Electric = new JLabel("Vehicle ID for sell");
  sellField_Electric = new JTextField();
  sellLabel_Electric.setBounds(30,380,150,30);
  sellField_Electric.setBounds(200,380,180,20);
  sellLabel_Electric.setFont(f_16);

  sellLabel_Electric.setForeground(Color.white);
  panelElectricScooter_Purchased.add(sellLabel_Electric);
  panelElectricScooter_Purchased.add(sellField_Electric);
```

```
 //Price for Selling
 priceSell_Label = new JLabel("Price for sell");
 priceSell_Field = new JTextField();
 priceSell_Label .setBounds(30,405,150,30);
 priceSell_Field.setBounds(200,405,180,20);
 priceSell_Label .setFont(f_16);
 priceSell_Label.setForeground(Color.white);
 panelElectricScooter_Purchased.add(priceSell_Label );
 panelElectricScooter_Purchased.add(priceSell_Field);
 //Button for selling the the vehicle
 btnSell_Electric = new JButton("Sell");
 btnSell_Electric.setFont(f_16);
 btnSell_Electric.setBounds(140,445,140,30);
 btnSell_Electric.addActionListener(this);
 panelElectricScooter_Purchased.add(btnSell_Electric);




//Title label add
bookTitle_Purchased = new JLabel("Pruchased and Sell ");
bookTitle_Purchased.setBounds(90,10,300,30);
bookTitle_Purchased.setFont(f_30);
bookTitle_Purchased.setForeground(Color.white);
panelElectricScooter_Purchased.add(bookTitle_Purchased);


 //Display Button
btnDisplay_Electric = new JButton("Display");
```

```
        btnDisplay.setFont(f_16);
        btnDisplay_Electric.setBounds(550,560,140,35);
        btnDisplay_Electric.addActionListener(this);
        panelElectricScooter.add(btnDisplay_Electric);

        //Clear Button
        btnClear_Electric = new JButton("Clear");
        btnClear.setFont(f_16);
        btnClear_Electric.setBounds(750,560,140,35);
        btnClear_Electric.addActionListener(this);
        panelElectricScooter.add(btnClear_Electric);






        }
    public static void main(String[] args){



        new Transport_GUI();





    }

    //Implements the methods of the ActionListner
     public void actionPerformed(ActionEvent e){
          if(e.getSource() == btnAutoRickshaw){
            panelAutoRickshaw.setVisible(true);
            panelElectricScooter.setVisible(false);
            btnAutoRickshaw.setBackground(Color.red);
            btnAutoRickshaw.setForeground(Color.BLACK);
            btnElectricScooter.setBackground(new JButton().getBackground());
            btnElectricScooter.setForeground(new JButton().getForeground());
        }
         if(e.getSource() == btnElectricScooter){
            panelElectricScooter.setVisible(true);
            panelAutoRickshaw.setVisible(false);
            btnElectricScooter.setBackground(Color.red);
            btnElectricScooter.setForeground(Color.BLACK);
            btnAutoRickshaw.setBackground(new JButton().getBackground());
            btnAutoRickshaw.setForeground(new JButton().getForeground());

        }
        if(e.getSource() == btnAdd_AutoRickshaw){
            if(idField.getText().isEmpty() || nameField.getText().isEmpty() ||
colorField.getText().isEmpty() || engineField.getText().isEmpty() ||
               speedField.getText().isEmpty() || torqueField.getText().isEmpty() ||
groundField.getText().isEmpty() || fuelField.getText().isEmpty() ||
                weightField.getText().isEmpty()){
                    JOptionPane.showMessageDialog(null,"Empty Field Found..");



               }else{
                  try{
                  id_Text = Integer.parseInt(idField.getText());
                }catch(Exception ex){
                    JOptionPane.showMessageDialog(null,"Enter Vehicle id in integer form");
                    idField.setText("");
                    return;
                }
                  for(int i = 0; i < AutoRickshawList.size();i++ ){
                      AutoRickshaw autoObj = (AutoRickshaw)AutoRickshawList.get(i);
                      if(autoObj.getvehicleID() == id_Text){
                          JOptionPane.showMessageDialog(null,"This Vehicle already exists");
                          idField.setText("");
                          return;

                      }

                  }




                  try{
                      engineTxt = Integer.parseInt(engineField.getText());

                  }catch(Exception ex){
                      JOptionPane.showMessageDialog(null,"Enter engine Displacement in
integer form");
                      engineField.setText("");
                      return;

                  }
                  try{
                      fuelTxt = Integer.parseInt(fuelField.getText());

                  }catch(Exception ex){
                      JOptionPane.showMessageDialog(null,"Enter fuel capacity in integer
form");
```

```
                }

                AutoRickshawObj = new
AutoRickshaw(id_Text,nameField.getText(),weightField.getText(),colorField.getText(),sp
eedField.getText(),

engineTxt,torqueField.getText(),groundField.getText(),fuelTxt);

                AutoRickshawList.add(AutoRickshawObj);
                JOptionPane.showMessageDialog(null,"Vehicle Added Successfully");
                idField.setText("");
                      nameField.setText("");
                      speedField.setText("");
                      engineField.setText("");
                      torqueField.setText("");
                      weightField.setText("");
                      fuelField.setText("");
                      groundField.setText("");
                      colorField.setText("");




            }
        }
        if(e.getSource() ==btnBook_AutoRickshaw){
           if(idField_Book.getText().isEmpty() || seats_Field.getText().isEmpty() ||
charge_Field.getText().isEmpty() ||
             dayCombo.getSelectedItem() == null || monthCombo.getSelectedItem() == null
|| yearCombo.getSelectedItem() == null){
              JOptionPane.showMessageDialog(null,"Empty Field Found");
        }
         else{
              if(AutoRickshawList.size() == 0){
               JOptionPane.showMessageDialog(null,"Please Add a List First..");
            }else{
               try {
                    seats_FieldTxt = Integer.parseInt(seats_Field.getText());


                     }catch(Exception ex){
                      JOptionPane.showMessageDialog(null,"Please Enter number of seats
in int form");
                      seats_Field.setText("");
                    }
                String bookedDate = yearCombo.getSelectedItem() + "/"  +
monthCombo.getSelectedItem() + "/" + dayCombo.getSelectedItem();
                  boolean check =false;
                 for(int i=0; i < AutoRickshawList.size();i++){
                      if((AutoRickshawList.get(i).getvehicleID()) ==
(Integer.parseInt(idField_Book.getText()))){

                            AutoRickshaw obj = (AutoRickshaw)AutoRickshawList.get(i);
                            if(obj.getisBooked() == true ){

                                 JOptionPane.showMessageDialog(null,"Vehicle is already
Booked");



                            }else{
                                obj.book(bookedDate, charge_Field.getText(), seats_FieldTxt);
                                JOptionPane.showMessageDialog(null,"Vehicle Booked
Succesfully");
```

```
                        }
                        check = true;



                     }
                     }
                  if(check ==false){
                        JOptionPane.showMessageDialog(null,"Vehicle doesn't exists");
                }
                  idField_Book.setText("");
                  charge_Field.setText("");
                  brandField_Electric.setText("");
                  seats_Field.setText("");
                   }
             }

        }

        if(e.getSource() == btnDisplay){

            for(int i =0;i<AutoRickshawList.size();i++){
                AutoRickshaw Obj = (AutoRickshaw)AutoRickshawList.get(i);
                Obj.AutoRickshaw_display();

System.out.println("______________________________________________________________
____");
            }

        }
        if(e.getSource() == btnClear){
            idField_Book.setText("");
            charge_Field.setText("");
            brandField_Electric.setText("");
            seats_Field.setText("");
            idField.setText("");
            nameField.setText("");
            speedField.setText("");
            engineField.setText("");
            torqueField.setText("");
            weightField.setText("");
            fuelField.setText("");
            groundField.setText("");
            colorField.setText("");

        }

        if(e.getSource() == btnAdd_Electric){

if(idField_Electric.getText().isEmpty()||nameField_Electric.getText().isEmpty()||colorField
_Electric.getText().isEmpty()||speedField_Electric.getText().isEmpty()||

weightField_Electric.getText().isEmpty()||batteryField_Electric.getText().isEmpty()){
            JOptionPane.showMessageDialog(null,"Empty Field Found");



            }
            else{
            try{
            id_Electric = Integer.parseInt(idField_Electric.getText());
        }catch(Exception ex){
        JOptionPane.showMessageDialog(null,"Vehicle id needed in Integer form");
        idField_Electric.setText("");
        return;
        }

        try{
           battery_Electric = Integer.parseInt(batteryField_Electric.getText());
        }catch(Exception ex){
              JOptionPane.showMessageDialog(null,"Battery Capacity needed in Intger
form");
              batteryField_Electric.setText("");
              return;

        }
        for(int i =0 ; i < ElectricScooterList.size() ; i++){
           Electric_Scooter electricObj = (Electric_Scooter)ElectricScooterList.get(i);
           if(electricObj.getvehicleID() == Integer.parseInt(idField_Electric.getText())){
               JOptionPane.showMessageDialog(null,"Vehicle already exist");
               idField_Electric.setText("");
               return;

            }
        }
        ElectricScooterObj = new
Electric_Scooter(id_Electric,nameField_Electric.getText(),colorField_Electric.getText(),s
peedField_Electric.getText(),weightField_Electric.getText(),
                                        battery_Electric);
        ElectricScooterList.add(ElectricScooterObj);
        JOptionPane.showMessageDialog(null,"Vehicle added Successfully");
        idField_Electric.setText("");
        nameField_Electric.setText("");
        colorField_Electric.setText("");
        speedField_Electric.setText("");

        batteryField_Electric.setText("");
        weightField_Electric.setText("");









      }

    }



        if(e.getSource() == btnPurchase_Electric){
            if(idField_Purchase_Electric.getText().isEmpty() ||
chargeField_Electric.getText().isEmpty() || rangeField_Electric.getText().isEmpty()||
                brandField_Electric.getText().isEmpty() ||
mileageField_Electric.getText().isEmpty() || priceField_Electric.getText().isEmpty()){
                    JOptionPane.showMessageDialog(null,"Empty Field found");


                }else{
                    if(ElectricScooterList.size() ==0){
                        JOptionPane.showMessageDialog(null,"No Vehicle is added");
                    }else{
                        try{
                           int priceTxt_Electric = Integer.parseInt(priceField_Electric.getText());

                        }catch(Exception ex){
                           JOptionPane.showMessageDialog(null,"Price in needed in
Integerform");
                           priceField_Electric.setText("");
                           return;
                        }
                        try{
                           int rangeTxt_Electric = Integer.parseInt(rangeField_Electric.getText());

                        }catch(Exception ex){
                           JOptionPane.showMessageDialog(null,"Range is needed in Integer
form");

                           rangeField_Electric.setText("");
                           return;
                        }
                        boolean check = false;
                        for(int i = 0;i < ElectricScooterList.size(); i++){
                           Electric_Scooter electricObj =
(Electric_Scooter)ElectricScooterList.get(i);
                           if(electricObj.getvehicleID()==
(Integer.parseInt(idField_Purchase_Electric.getText()))){

                  if(electricObj.gethasPurchased() == true){
```

```
                                JOptionPane.showMessageDialog(null,"Vehicle is already
Purchased");

                            }
                        else{
                            electricObj.purchase(brandField_Electric.getText(),
priceTxt_Electric,chargeField_Electric.getText() , mileageField_Electric.getText(),
rangeTxt_Electric);
                            JOptionPane.showMessageDialog(null,"Vehicle purchased
successfully");

                        }
                        check = true;
                      }

                      }
                      if(check ==false){
                          JOptionPane.showMessageDialog(null,"Vehicle doesn't exists");

                      }



                        }
                        idField_Purchase_Electric.setText("");
                        chargeField_Electric.setText("");
                        rangeField_Electric.setText("");
                        brandField_Electric.setText("");
                        mileageField_Electric.setText("");
                        priceField_Electric.setText("");
                  }
              }
```

```
    if(e.getSource() ==btnSell_Electric){
        if(sellField_Electric.getText().isEmpty() || priceSell_Field.getText().isEmpty()){
            JOptionPane.showMessageDialog(null,"Empty Field Found");
        }else{
            if(ElectricScooterList.size() == 0){
                JOptionPane.showMessageDialog(null,"No vehicle is added");
            }else{
                try{
                    priceSell_Txt = Integer.parseInt(priceSell_Field.getText());
                }catch(Exception ex){
                     JOptionPane.showMessageDialog(null,"Price in needed in intger form");

                }
                boolean check = false;
                for(int i =0;i < ElectricScooterList.size();i++){
                    Electric_Scooter electricObj =
(Electric_Scooter)ElectricScooterList.get(i);
                    if(electricObj.getvehicleID() ==
Integer.parseInt(sellField_Electric.getText())){
                        if(electricObj.gethasSold() == true){
                            JOptionPane.showMessageDialog(null,"This Vehicle is already
sold");
                        }else{
                            electricObj.sell(priceSell_Txt);
                            JOptionPane.showMessageDialog(null,"Vehicle Sold
Successfully");
                        }
                        check = true;
                    }
                }
                if(check ==false){

                    JOptionPane.showMessageDialog(null,"This Vehicle doesn't exists");

            }
            sellField_Electric.setText("");
            priceSell_Field.setText("");




        }



    }


  }
  if(e.getSource() ==btnDisplay_Electric){
     if(ElectricScooterList.size() == 0){
        JOptionPane.showMessageDialog(null,"No vehicle is addes");
     }else{
     for(int i = 0;i < ElectricScooterList.size();i++){
        Electric_Scooter Obj = (Electric_Scooter)ElectricScooterList.get(i);
        Obj.ElectricScooter_display();

System.out.println("_______________________________________________________
___________");
     }
  }
```

```
    }
    if(e.getSource() == btnClear_Electric){
        sellField_Electric.setText("");
        priceSell_Field.setText("");
        idField_Purchase_Electric.setText("");
        chargeField_Electric.setText("");
        rangeField_Electric.setText("");
        brandField_Electric.setText("");
        mileageField_Electric.setText("");
        priceField_Electric.setText("");
         idField_Electric.setText("");
        nameField_Electric.setText("");
        colorField_Electric.setText("");
        speedField_Electric.setText("");

        batteryField_Electric.setText("");
        weightField_Electric.setText("");
    }



}
}
```

8.2 Vehicle

```
//

/**
 Name:Saugat Basnet
 Id = np014s220042

 The class Vehicle is call following attributes and to set accersor and
 mutator method
 within the class
**/
public class Vehicle {
    private int vehicleID;// Id assigned to the Vehicle
    private String vehiclename;// Name of the Vehicle
    private String vehicleWeight;//Weight of the Vehicle
    private String vehicleColor;//Color of the Vehicle
    private String vehiclespeed;// Speed of the Vehicle


    /* The class vehicles attribute are assigned with value using
     * Constructor parameter*/
    public Vehicle
    (int vehicleID,String vehiclename,String vehicleWeight,String vehicleColor)
    {
    this.vehicleID = vehicleID;
    this.vehicleWeight = vehicleWeight;
    this.vehicleColor = vehicleColor;
    this.vehiclespeed = vehiclespeed;
    this.vehiclename = vehiclename;
   }
   //Getter method is used tp reads the value of the variable or retrieve the value
```

```
  public int getvehicleID(){
     return this.vehicleID;
  }
  public String getvehicleweight(){
     return this.vehicleWeight;
  }
  public String getvehiclecolor(){
     return this.vehicleColor;
  }
  public String getvehicle_Speed(){
     return this.vehiclespeed;
  }
  public String getvehicleName(){
     return this.vehiclename;
  }
  //Setter methos takes a parameter and assigned it to the attribute
  //Setter method to set Speed of the vehicle
  public void setVehicle_Speed(String newSpeed){
     vehiclespeed = newSpeed;
  }
  //Setter method to set Color of the Vehicle
  public void setVehicleColor(String newvehicle_Color){
     vehicleColor = newvehicle_Color;
  }
  ////Method display is used to display the attributes of vehicle class with suitable
notation
  public void display(){
     System.out.println("Vehicle Id =" + vehicleID);
     System.out.println("Vehicle Name = " + vehiclename);
     System.out.println("Speed of the Vehicle = " + vehiclespeed);
     System.out.println("Colour of the Vehicle = " + vehicleColor);
```

```
      if (vehicleWeight == " "){
         System.out.println("Empty");

      }else{
         System.out.println("Weight of the Vehicle=" + vehicleWeight);
       }
   }


   }
```

8.3 AutoRickshaw

```
//The class AutoRickShaw is called which is the sub class of class Vehicle
public class AutoRickshaw extends Vehicle{
   private int engine_Displacement;// Private methis are declare because it can be
accesed otside the class
   private String torque;
   private int number_Seats;
   private int fuel_capacity;
   private String groundClearance;
   private int charge_Amount;
   private String bookedDate;
   private boolean  isBooked;
   // Constructor method AutoRickshaw is called to set parameter on the attribute
   public AutoRickshaw
   (int vehicleID,String vehiclename,String vehicleWeight,String vehicleColor,
   String vehicleSpeed, int engine_Displacement, String torque,String groundClearance,
int fuelCapacity){
      super(vehicleID,vehiclename, vehicleWeight,vehicleColor);
      super.setVehicleColor(vehicleColor);
      super.setVehicle_Speed(vehicleSpeed);
      this.engine_Displacement = engine_Displacement;
      this.torque  = torque;
      this.fuel_capacity = fuelCapacity;
      this.groundClearance = groundClearance;
      this.isBooked  = false;
   }
    //Getter method is used tp reads the value of the variable or retrive the value
   public int getengine_Displacement(){
      return this.engine_Displacement;
   }

   public  String gettorque(){
      return this.torque;
   }




   public int getnumber_Seats(){
      return this.number_Seats;
   }

   public int getfuel_capacity(){
      return this.fuel_capacity;
   }

   public String getgroundClearance(){
      return this.groundClearance;
   }

   public int getcharge_Amount(){
      return this.charge_Amount;
   }

   public boolean getisBooked(){
      return this.isBooked;
   }
   //Setter methos takes a parameter and assigned it to the attribute
   public  void Setcharge_Amount(int charge_Amount){
      this.charge_Amount= charge_Amount;
   }

   public void SetNumber_Seats(int number_Seats){
      this.number_Seats = number_Seats;
   }
   public void setIsBooked(boolean isbooked){
      this.isBooked = isBooked;
   }
   // Method book is called to set the bookeddate,seat numbers and charge amount if
autorickshaw is not booked
   public void book(String newBookedDate,String chargeAmount,int number_Seats){
      if (isBooked == false){
          this.bookedDate= newBookedDate;
          SetNumber_Seats(number_Seats);
          Setcharge_Amount(charge_Amount);
          isBooked = true;
          System.out.println("You have booked the Vehicle");
      }else{
          System.out.println("The AutoRickshaw is already Booked " +
super.getvehicleID() + " is Booked");
      }
   }

   //Method display is used to display the attributes of autorickshaw class with suitable
notation
   public void AutoRickshaw_display(){
      super.display();
       if (isBooked == true){
          System.out.println("Engine Displacement of AutoRickshaw  " +
engine_Displacement);
          System.out.println("The Torque of AutoRickshaw is " + torque);
          System.out.println("The Fuel Tank Capacity is " + fuel_capacity);
          System.out.println("The Ground Clearnce is " + groundClearance);
          System.out.println("The booking date of AutoRickshaw  " + bookedDate);
          if (charge_Amount == 0   ){
              System.out.println("Empty Charge Amount");

          }else{

              System.out.println("Charged Amount  " + charge_Amount);

          }
          if (number_Seats == 0){
              System.out.println("Empty number of seats");
          }
          else{
              System.out.println("Number of Seats  " + number_Seats);
          }

      }
   }



}
```

8.4 ElectricScooter

```
// Eletric_Scooter class is called which set the attributes the of the electric scooter
public class Electric_Scooter extends Vehicle{
   private int Range;
   private int Battery_Capacity;
   private int Price;
   private String Charging_Time;
   private String Brand;
   private String Mileage;
   private boolean hasPurchased;
   private boolean hasSold;
   // Constructor method is used to set the attributes with following parameter
   public Electric_Scooter
   (int vehicleID,String vehiclename,String vehicleWeight,String vehicleColor,String
vehiclespeed,int Battery_Capacity){
      super( vehicleID, vehiclename, vehicleWeight, vehicleColor);
      super.setVehicle_Speed(vehiclespeed);
      super.setVehicleColor(vehicleColor);
      this.Battery_Capacity = Battery_Capacity;
      Range = 0;
      Price = 0;
      Brand = "";
      Charging_Time = "";
      hasPurchased = false;
      hasSold=false;
      }
      //Above attributes are assigned with accesor method

      public int getRange(){
          return this.Range;
      }
      public int getBattery_Capacit(){
          return this.Battery_Capacity;
      }
      public int getPrice(){
          return this.Price;
      }
      public String getCharging_Time(){
          return this.Charging_Time;
      }
      public String getBrand(){
          return this.Brand;
      }
      public String getMileage(){
          return this.Mileage;
      }
      public boolean gethasPurchased(){
          return this.hasPurchased;
      }
      public boolean gethasSold(){
          return this.hasSold;
      }


      public void setBrand(String Brand){
          if (hasPurchased = false){
              this.Brand = Brand;
          }else{
              System.out.println("Already Purchased");
          }
      }
      //Method purchase is called to called some attributes and if hasPurchased is false
  set the brand name

      public void purchase(String Brand,int Price,String Charging_Time,String
  Mileage,int Range){
          this.Brand = Brand;
          this.Price = Price;
          this.Charging_Time = Charging_Time;
          this.Mileage = Mileage;
          this.Range = Range;



          if (hasPurchased = false){
              setBrand(Brand);

              }
          hasPurchased  = true;
      }
      //Method sell values to attribute when the hasSold is false

      public void sell(int newPrice){

          if(hasSold == false){
              Price = newPrice;
              Range = 0;
              Charging_Time = "";
              Mileage = "";
              Battery_Capacity = 0;
              hasSold = true;
```

```
            hasPurchased = true;



         }else {
            System.out.println("The Scooter is already sold");
         }


         }
         ///Method display is used to display the attributes of vehicle class with suitable
notation
      public void ElectricScooter_display(){
         super.display();
         if ( hasPurchased == true){
            System.out.println("The brand of electric Scooter: " + Brand);
            System.out.println("The battery capacity of electric scooty is " +
Battery_Capacity);
            System.out.println("Thec mileage of the Electric scooty is " + Mileage);
            System.out.println("The range of Electric Scooty is " + Range);
            System.out.println("The recharge time of electric Scooty is " +
Charging_Time);

         }

      }




}
```

10. Reference

J.ECK, D. (n.d.). Introuction to programming using Jaa.

Java TPOINT. (n.d.). Retrieved from https://www.javatpoint.com/java-tutorial

JAVA TPOINT. (n.d.). *JavatPOINT*. Retrieved from https://www.javatpoint.com/java-tutorial

W3.School. (n.d.). Retrieved from https://www.w3schools.com/java/java_methods.asp

11. Bibliography

J.ECK, D. (n.d.). Introuction to programming using Jaa.

Java TPOINT. (n.d.). Retrieved from https://www.javatpoint.com/java-tutorial

JAVA TPOINT. (n.d.). *JavatPOINT*. Retrieved from https://www.javatpoint.com/java-tutorial

W3.School. (n.d.). Retrieved from https://www.w3schools.com/java/java_methods.asp