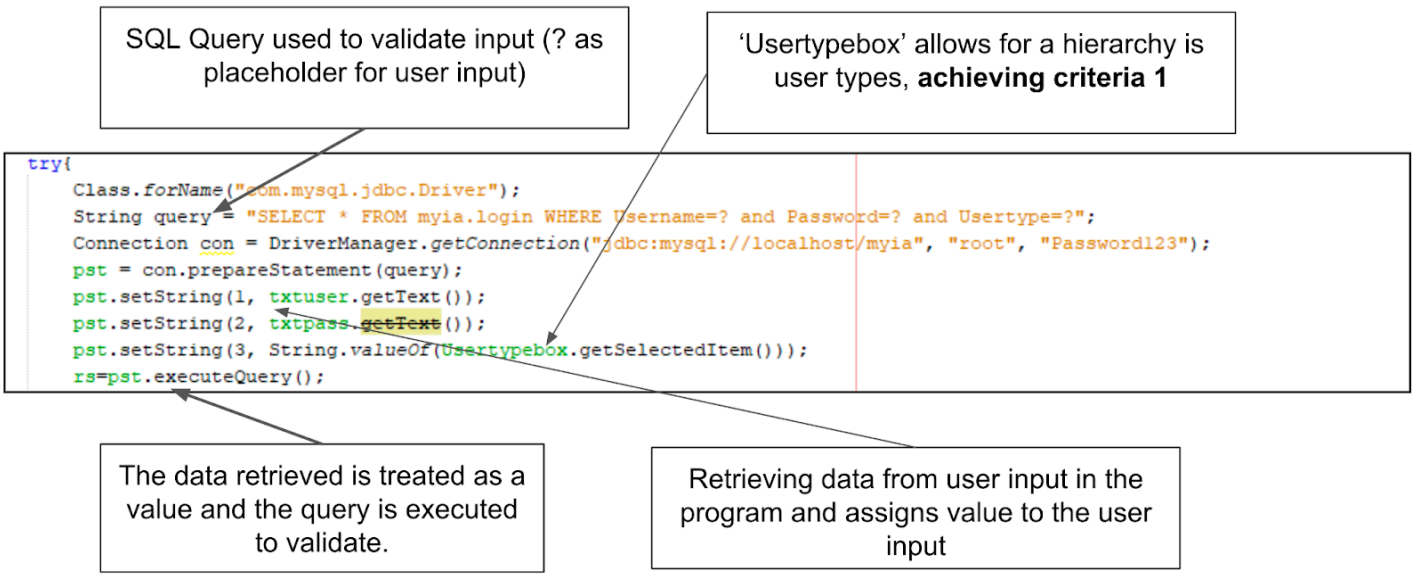
Criteria C: Development

Table of contents

Section C1: Connecting to MySQL Database

C1.1: Database connection

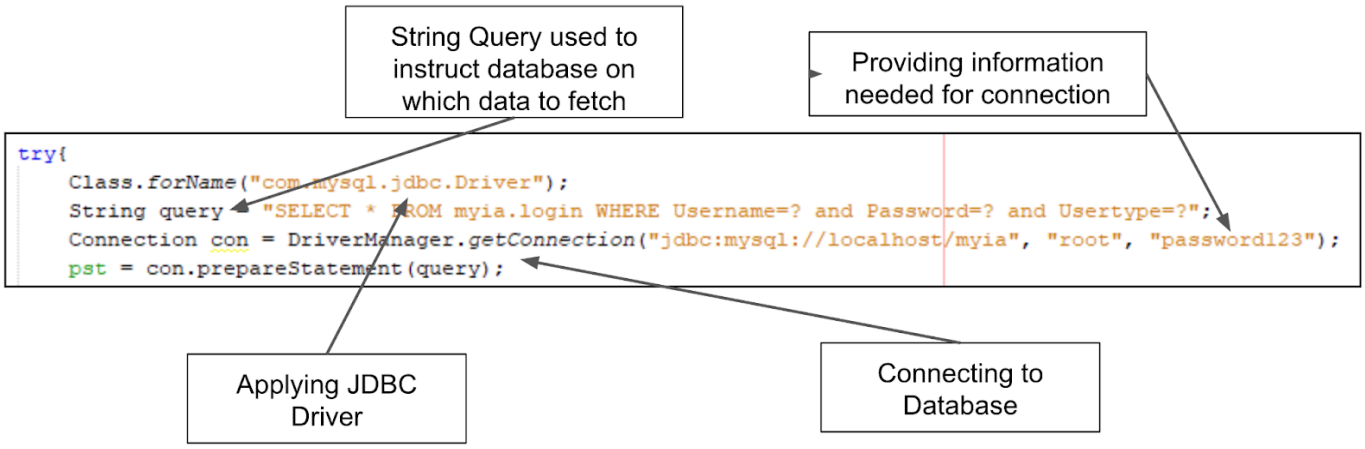
A connection is established with the MySQL database using the Driver. Manager. getConnection method. The database table "Login" stores the login credentials for the drivers and the managers used to validate the user's input. The JDBC driver acts as a small piece of software that allows the connection of a data source to Netbeans allowing the sending and updating of query statements from Netbeans (DbSchema, n.d.).



The pst.setString statement sets the user input to a numerical value. It's then compared against the stored login credentials from the MySQL database. This allows for both the Managers and Drivers to use the same platform for different purposes. The Usertype box allows the driver and the manager to input their user type, determining where they will be redirected, **achieving criteria 1**.

C1.2: Retrieving stored login details

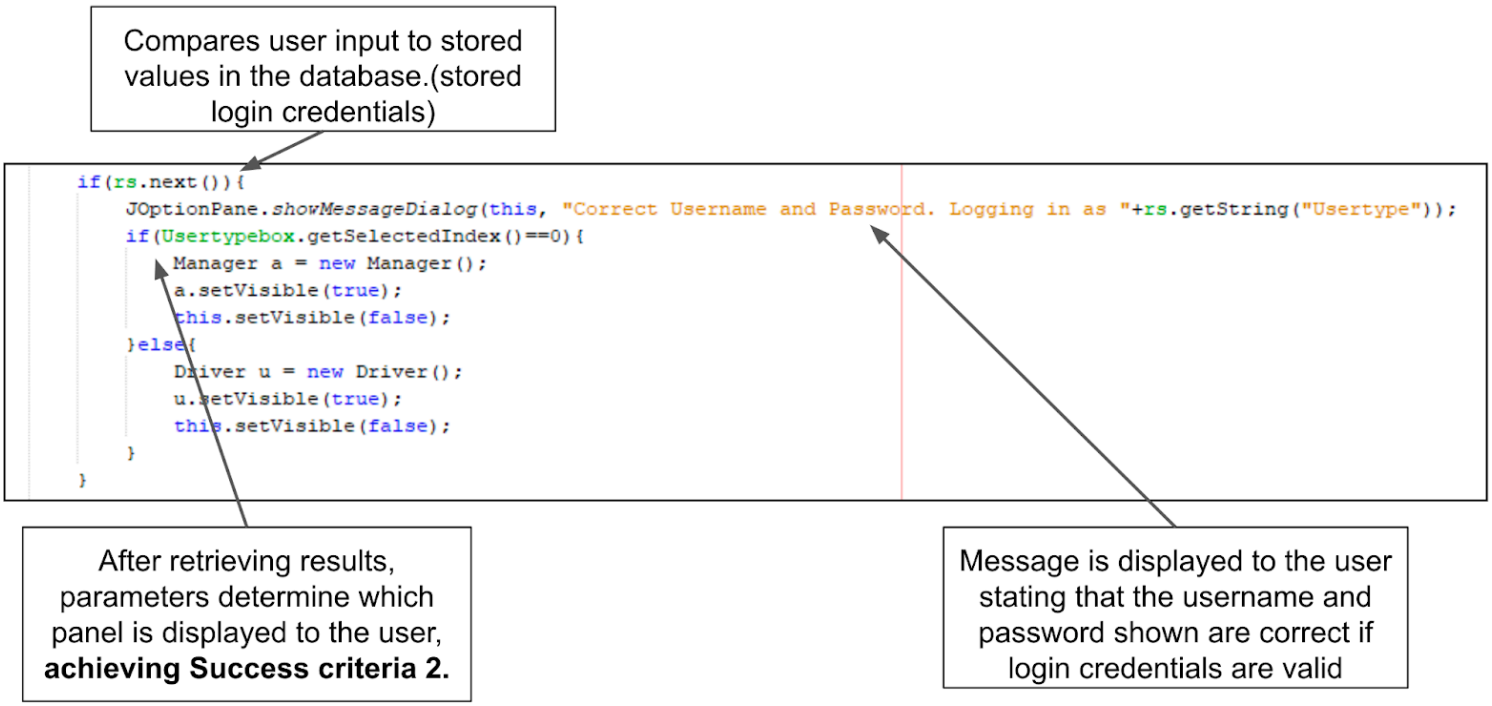
The program fetches the data input from the user and validates it with the login credentials stored in the MySQL database. This ensures that there are no unauthorised logins and that the appropriate panel is displayed depending on the login inputted by either the drivers or the managers. The "SELECT * FROM" statement retrieves the Username, Password and User type columns from the table that match with '?' which acts as a placeholder.



The 'preparedStatement(query)' fetches all the data columns from the MySQL table before the comparison used to validate the login.

C1.3: Validating Login details

The user's input is validated through a query that compares it to the stored values in the MySQL database. After which, a message is displayed notifying the driver or manager of the correct entry of the login credentials and redirects them to either the driver or manager user panels depending on their designated user type, **achieving Success Criteria 2.**



C1.4: Displaying incorrect log-in Error message

Using JOptionPane import that provides a simple dialogue block, the driver/manager will be prompted with an error message, if their input is incorrect, which will prompt them to log in again.

```
}
else{
JOptionPane.showMessageDialog(this, "Username and Password incorrect.");
}
```

Error message displayed stating that the username and password provided is incorrect

C1.5 Use of Try-Catch Statement

The Try-Catch statement allows the code within the try-catch statement to be executed except for errors (Simplilearn, 2023). If an expectation occurs that cannot be addressed by the code in the try block, the code in the catch block is executed to address the exception.

```
try{
    Class.forName("com.mysql.jdbc.Driver");
    String query = "SELECT * FROM myia.login WHERE Username=? and Password=? and Usertype=?";
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost/myia", "root", "Deathlbyl");
    pst = con.prepareStatement(query);
    pst.setString(1, txtuser.getText());
    pst.setString(2, txtpass.getText());
    pst.setString(3, String.valueOf(Usertypebox.getSelectedItem()));
    rs=pst.executeQuery();
```

Try Block containing the code that is executed

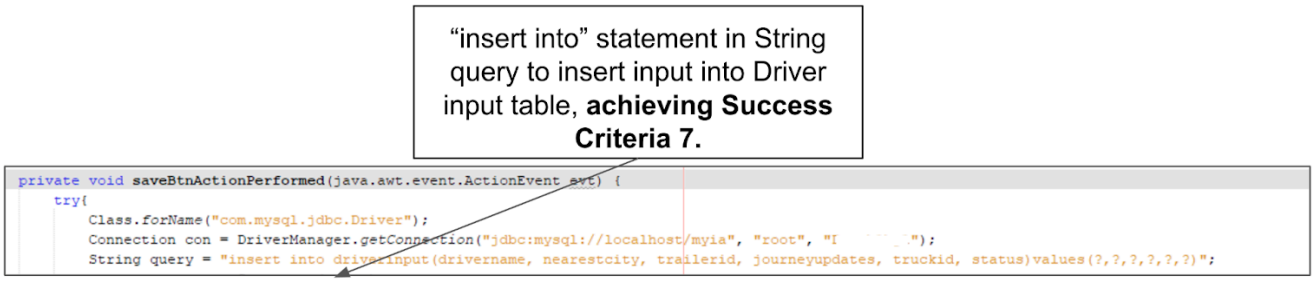
```
    }
    else{
    JOptionPane.showMessageDialog(this, "Username and Password incorrect.");
    }
}catch (Exception ex){
JOptionPane.showMessageDialog(this, ex.getMessage());
```

Code in Catch block is run in the event of an exception. In this case, it displays an error message.

Section C2: Collecting and displaying of data

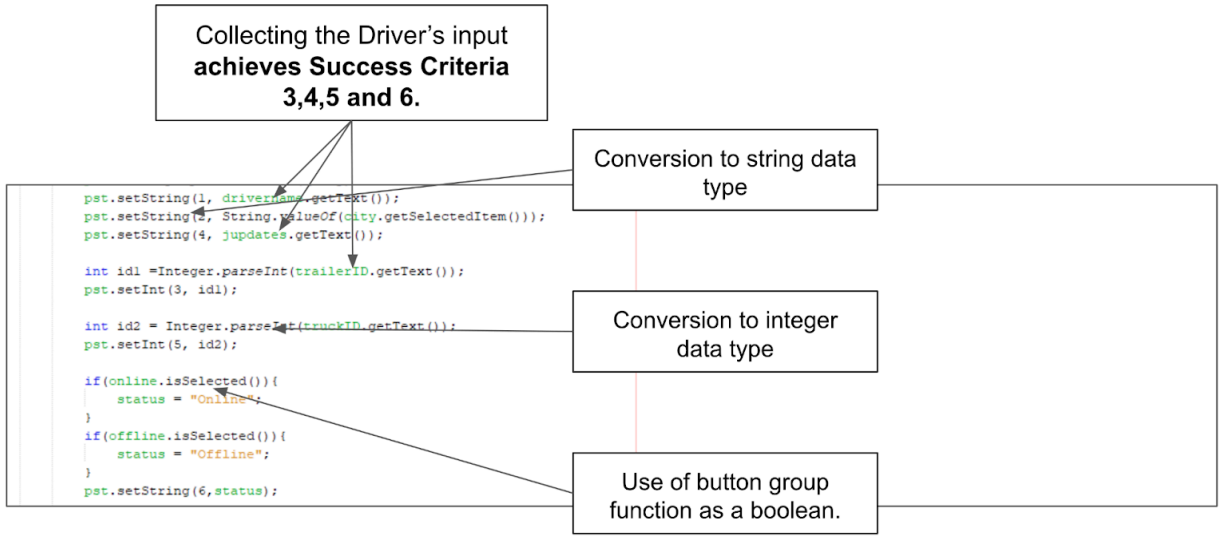
C2.1 Data collection and storing

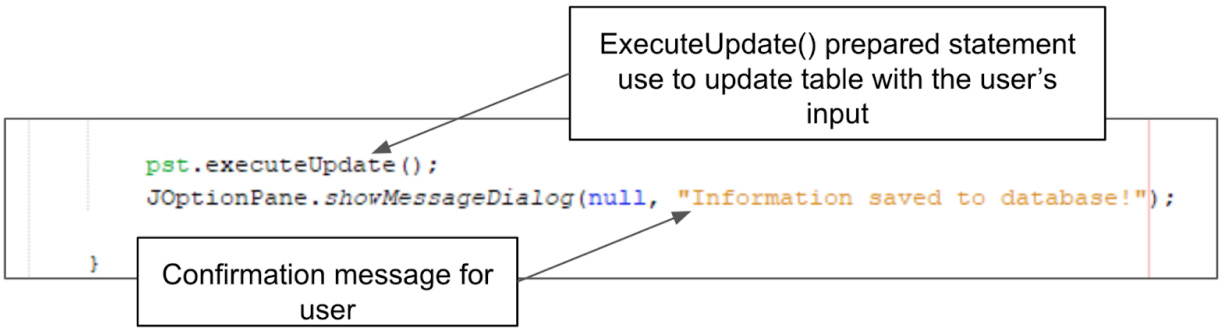
The information provided by the user's input is collected and stored in the MySQL database. To insert the driver's input to the table in the database, an "insert into" statement is used in the string query to store the input of the driver, **achieving success criteria 7.** Through this, the driver can input; their name, nearest city, trailer ID, truck ID and activity status, which the manager can later view tables in the manager user panel.



Before the data is stored, it is first converted to the desired data type to prevent conflict between the user's input and the set data type in the MySQL database using the SetString() and SetInt() functions, allowing for collection of the driver's input in the correct format as the driver name, nearest city, journey updates and activity status are stored as string while the truck and trailer ID are stored as integers. **achieving success criteria 3,4,5 and 6.**

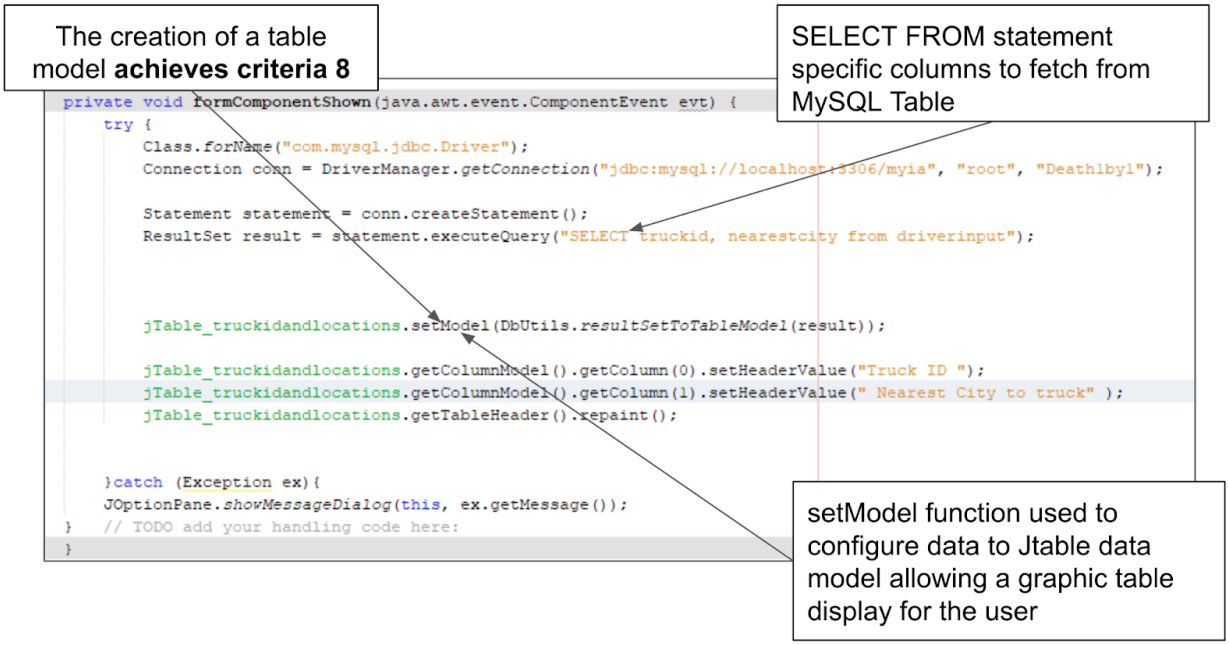
After the data is set to the desired data type, the program stores the data using the "executeUpdate" prepared statement. This updates the table in the MySQL database with the driver's inputs.





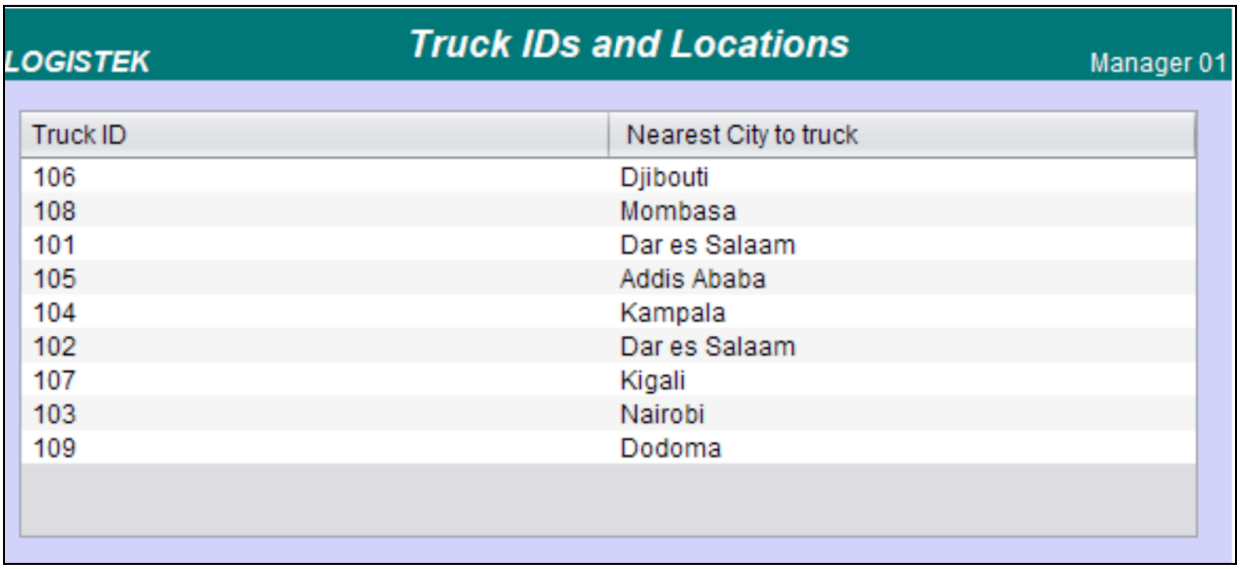
C2.2 Displaying sets of data in table form

An SQL query selects the 'truckid' and 'nearest city' columns from the MySQL table. The S statement object is used to execute the query and the results obtained are set to ResultSet. The program allows the manager to view the data inputted by the drivers using a table displayed in Netbeans, **achieving Success Criteria 8**.



After that, the 'setModel' function populates the table with the data fetched from the MySQL database. The DbUtilis external library uses resultSet and converts to XML data which can be

formatted easily to the NetBeans jTable. The data presented on the Manager panel depends on the filter selected by the manager. In this instance, the truck ID and Nearest city filter has been selected - populating the table with those two sets of data.



Truck ID	Nearest City to truck
106	Djibouti
108	Mombasa
101	Dar es Salaam
105	Addis Ababa
104	Kampala
102	Dar es Salaam
107	Kigali
103	Nairobi
109	Dodoma

C2.4 Graphic display of data set (Pie Chart)

The program employs the JFreeChart library to create a visual chart, specifically, the 3D pie chart used to calculate the available capacity of trucks by looking at the activity status provided by the driver's input (Bodnar, n.d.).

the JFreeChart and Chart Factory imports used to create 3D Pie Chart, **achieving Success Criteria 9.**

```
package login;

import java.awt.Dimension;
import javax.swing.JPanel;
import javax.swing.table.DefaultTableModel;

import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PiePlot3D;
import org.jfree.data.general.DefaultPieDataset;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
```

The data for the chart is extracted from the 'Status' column in the MySQL table through a query, where segments of the pie chart correspond to the tally of 'Online' and 'Offline' statuses, **achieving success criteria 9.**

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/myia?useSSL=false", "root", "Deathlbyl");

        Statement st = con.createStatement();
        String sql = "SELECT * FROM driverinput";
        ResultSet rs = st.executeQuery(sql);

        int onlineCount = 0;
        int offlineCount = 0;

        while (rs.next()) {
            String status = rs.getString("status");
            if (status.equals("Offline")) {
                offlineCount++;
            } else if (status.equals("Online")) {
                onlineCount++;
            }
        }

        DefaultPieDataset pieDataset = new DefaultPieDataset();
        pieDataset.setValue("Online", onlineCount);
        pieDataset.setValue("Offline", offlineCount);

        JFreeChart chart = ChartFactory.createPieChart3D("Driver Status", pieDataset, true, true, true);
        PiePlot3D plot = (PiePlot3D) chart.getPlot();

        // Create a ChartPanel and add the chart to it
        ChartPanel chartPanel2 = new ChartPanel(chart);
        chartPanel2.setPreferredSize(new Dimension(400, 300));

    }catch (Exception e) {
        System.out.println(e.getMessage());
    }
}
```

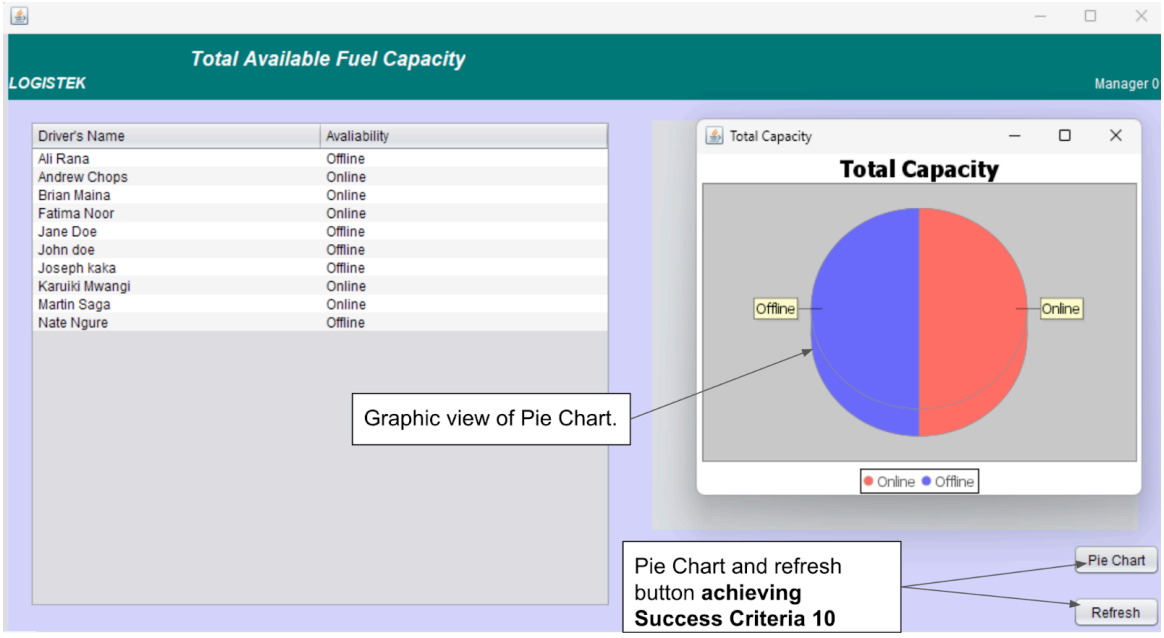
Declaration of count

Iteration used to tally number of Available trucks

Setting Pie chart value to tally in the count variables

Use of JFreeChart to create 3D pie chart.

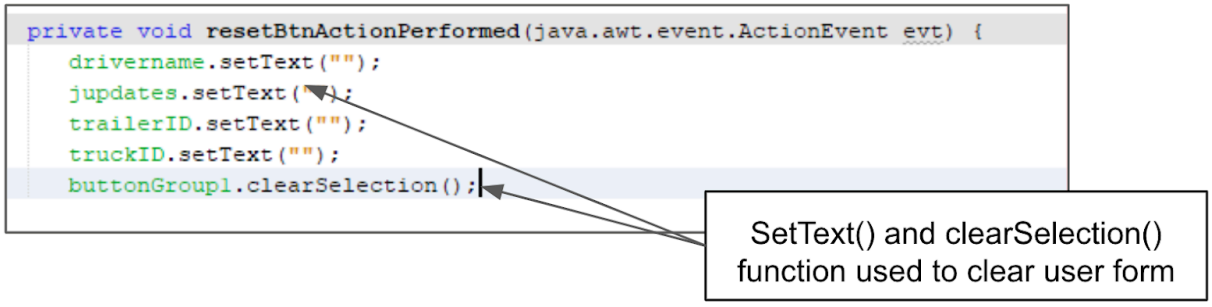
The tally is retrieved by iterating through each row of data using the *while(rs. next())*; loop where the string value of the "status" column is retrieved and the tally(onlineCount and offlineCount) with the corresponding string value is increased by 1.



Additional Features

Reset function for user input

The setText(“”) and clearSelection() are used to clear the (drivername, jupdates,trailerID, truckID and status) fields when the reset button is clicked, providing an easier way to reset the driver’s inputs.



Bibliography

Bodnar, J. (n.d.). Java JFreeChart - creating charts in Java with JFreeChart. Zetcode.com. https://zetcode.com/java/jfreechart/#:~:text=JFreeChart%20is%20a%20popular%20Java

Simplilearn. (2023, February 23). Try Catch in Java - Exception handling (With Examples) | https://www.simplilearn.com/tutorials/java-tutorial/try-catch-in-java#:~:text=The%20Try%20Catch%20in%20java

DbSchema. (n.d.). MySql JDBC Driver Download & Connectivity. Dbschema.com. Retrieved October 2, 2023, from https://dbschema.com/jdbc-driver/mysql.html#:~:text=JDBC%20drivers%20are%20Java%20library