Name: Love & Dharmik Patel

GUI Database Application Development of Manufacturing Assembly Using Object Oriented Programming

Time Estimate

Setting up the Project File	: 10 min
Creating the GUI	: 60 min
Developing the Controller for the GUI	: 80 min
Testing the Application with Examples from Simulation	: 30 min

GUI Database Application Development of Manufacturing Assembly Using Object Oriented Programming (50 pts total)

ACME Manufacturing has hired you as a software engineer to develop an information system for their manufacturing assembly to track their items and worker performance. The manufacturing assembly has an assembly line, multiple storage racks, a rework area, a packaging area, and two docks. Your team is assigned to observe the manufacturing assembly line and develop a simple GUI that will allow users to perform CRUD functionalities. You will be provided two tables and their attributes. You will have to create the tables through flyway using SQL commands. The following figure shows a simulation model of the manufacturing assembly, which we will treat as the "real" system in this assignment.

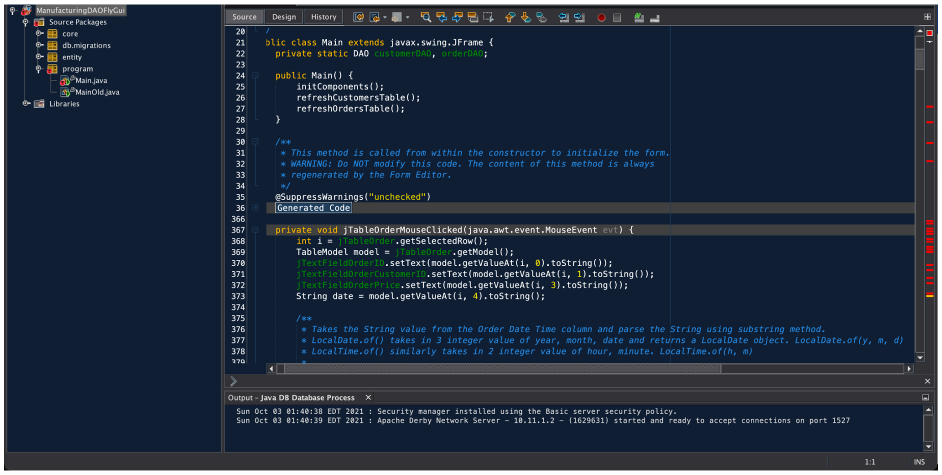
Rework Station
Station Yellow
Station Red

Your project export should include your existing database (created by flyway) and all the other necessary source code/libraries to compile and run the project successfully. Test your exported project zip file in another computer before submitting and see if it is working correctly on that computer when you imported it. In your report, you must show your work with screenshots and descriptions/captions about those screenshots. If you do not show your work, you will not get that point (there is no partial credit).

<Link to the starter project>

Part 1: <u>Setting Up the Project File (2 pts)</u>

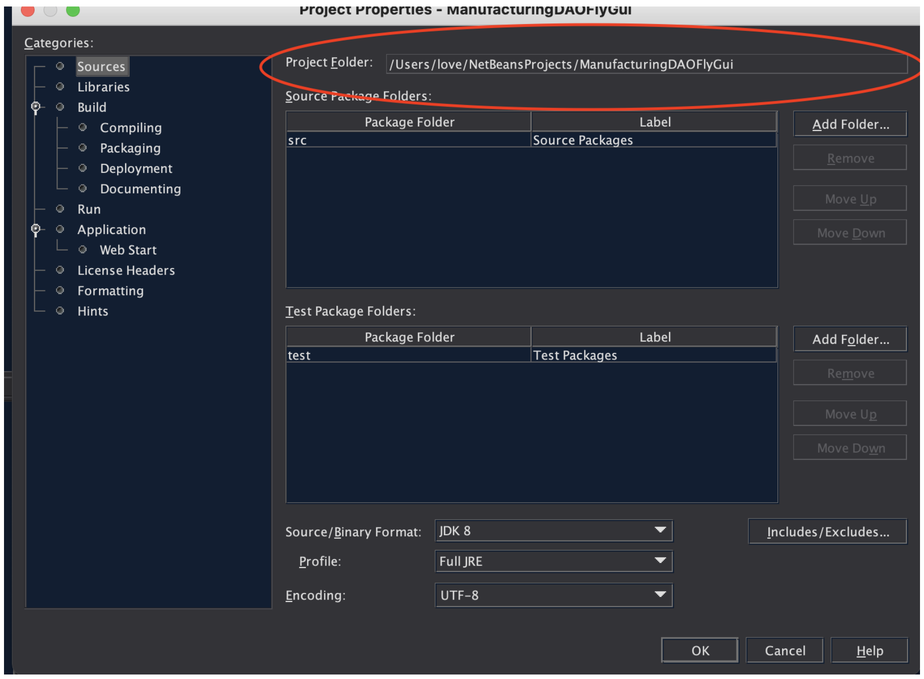
Download the starter project. Rename the project as ManufacturingCRUD. Copy the project into your using NetBeans copy project function (right click to your ManufacturingCRUD and select Copy, please do not copy it any other way). Now delete the Main.java file and add a new Java Frame (called Main.java) into your program. Include a screenshot of your folder directory.



```
/
blic class Main extends javax.swing.JFrame {
  private static DAO customerDAO, orderDAO;

  public Main() {
      initComponents();
      refreshCustomersTable();
      refreshOrdersTable();
  }

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

  private void jTableOrderMouseClicked(java.awt.event.MouseEvent evt) {
      int i = jTableOrder.getSelectedRow();
      TableModel model = jTableOrder.getModel();
      jTextFieldOrderID.setText(model.getValueAt(i, 0).toString());
      jTextFieldOrderCustomerID.setText(model.getValueAt(i, 1).toString());
      jTextFieldOrderPrice.setText(model.getValueAt(i, 3).toString());
      String date = model.getValueAt(i, 4).toString();

      /**
       * Takes the String value from the Order Date Time column and parse the String using substring method.
       * LocalDate.of() takes in 3 integer value of year, month, date and returns a LocalDate object. LocalDate.of(y, m, d)
       * LocalTime.of() similarly takes in 2 integer value of hour, minute. LocalTime.of(h, m)
```

Output - Java DB Database Process

```
Sun Oct 03 01:40:38 EDT 2021 : Security manager installed using the Basic server security policy.
Sun Oct 03 01:40:39 EDT 2021 : Apache Derby Network Server - 10.11.1.2 - (1629631) started and ready to accept connections on port 1527
```



Project Folder: /Users/love/NetBeansProjects/ManufacturingDAOFlyGui

Source Package Folders:

Package Folder	Label
src	Source Packages

Test Package Folders:

Package Folder	Label
test	Test Packages

Source/Binary Format: JDK 8

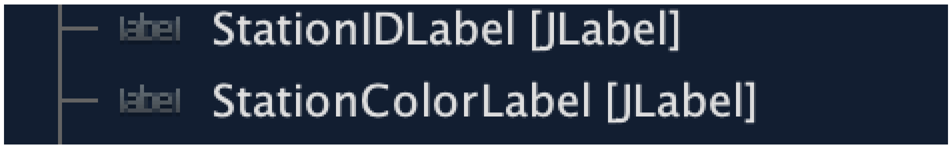
Profile: Full JRE

Encoding: UTF-8

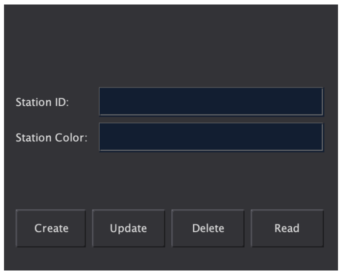
Part 2: Creating the GUI (10 pts)

Create a CRUD GUI application for Manufacturing Assembly database. You can use split pane, tabbed pane, or scroll pane to layout your GUI palette. In your Main.java using Java Swing create a shell graphical interface.

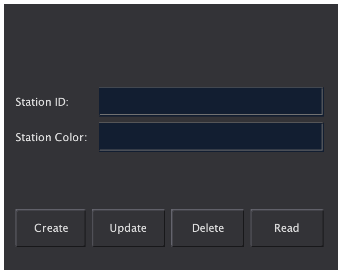
1. Create necessary labels for the Station with screenshots. **1 point.**

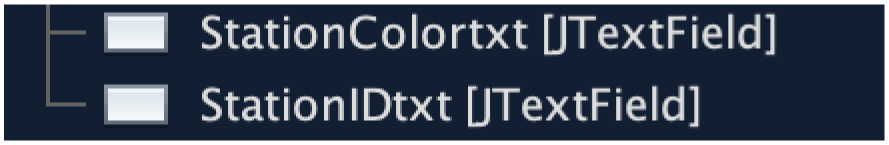


StationIDLabel [JLabel]
StationColorLabel [JLabel]



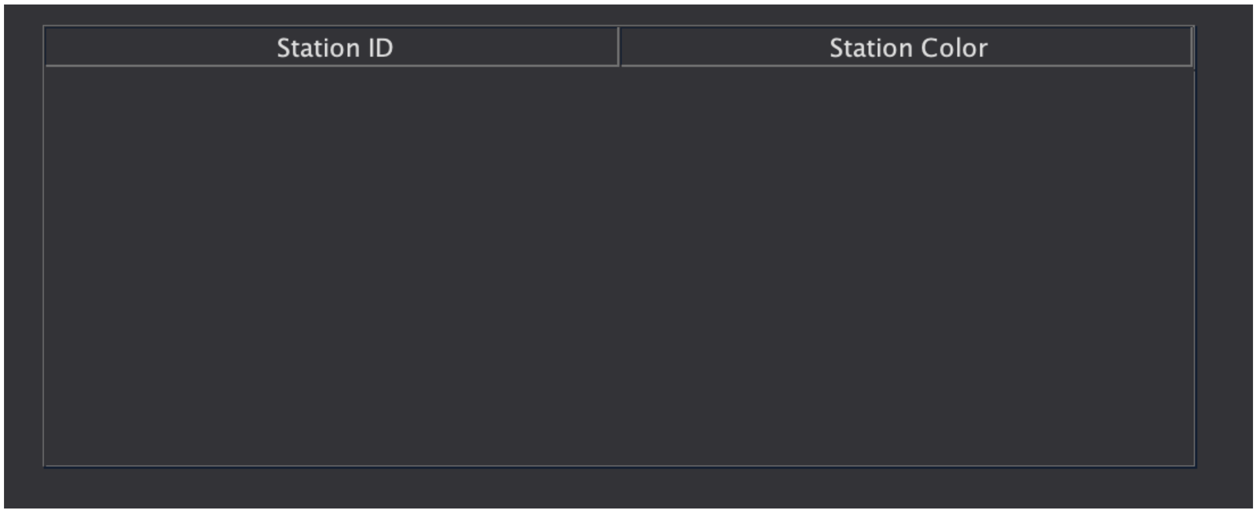
2. Create necessary text fields for Station with screenshots. **1 point.**





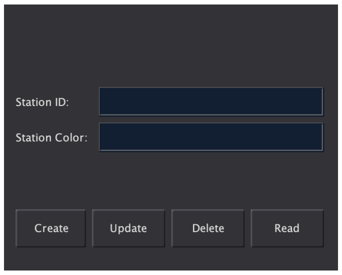
StationColortxt [JTextField]
StationIDtxt [JTextField]

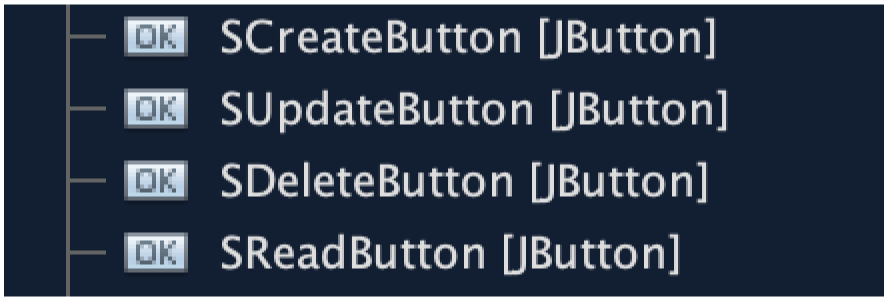
3. Create/Edit the JTable to include attributes from the Station table. Provide screenshots. **1 point.**



Station ID	Station Color

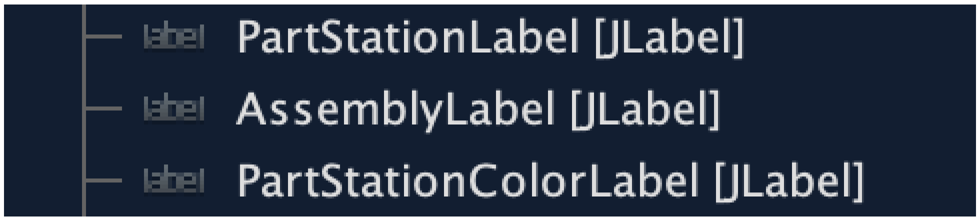
4. Create buttons for Station CRUD functionalities with screenshots. (*i.e.*, CREATE, UPDATE, DELETE) **1 point.** **DONE**

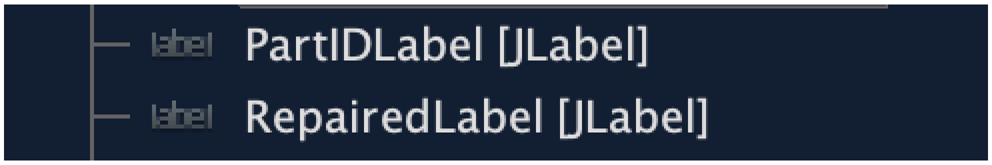


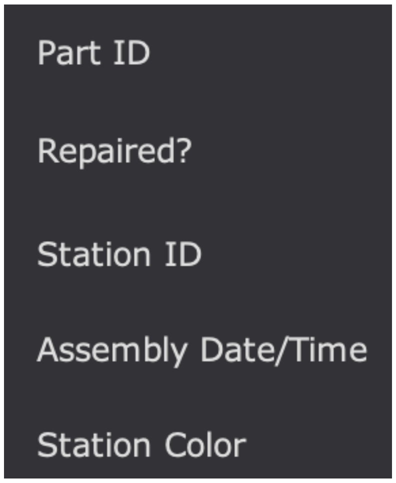


SCreateButton [JButton]
SUpdateButton [JButton]
SDeleteButton [JButton]
SReadButton [JButton]

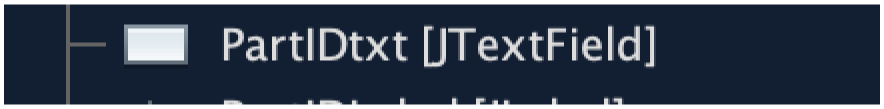
5. Create necessary labels for Part with screenshots. **1 point.**

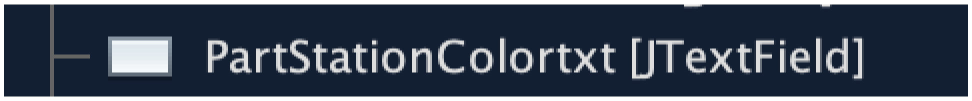

PartStationLabel [JLabel]
AssemblyLabel [JLabel]
PartStationColorLabel [JLabel]

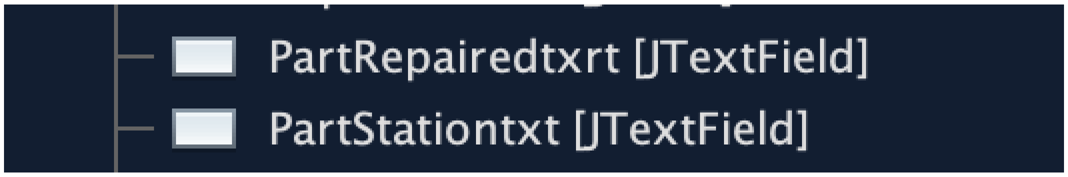

PartIDLabel [JLabel]
RepairedLabel [JLabel]

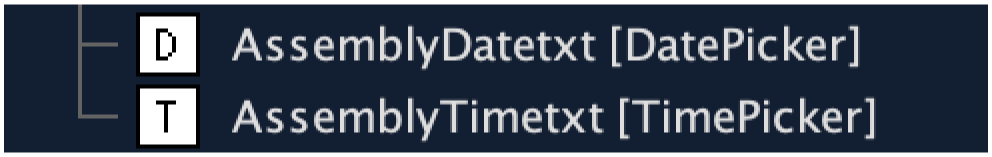

Part ID
Repaired?
Station ID
Assembly Date/Time
Station Color

6. Create necessary text fields for Part with screenshots. **1 point.**


PartIDtxt [JTextField]

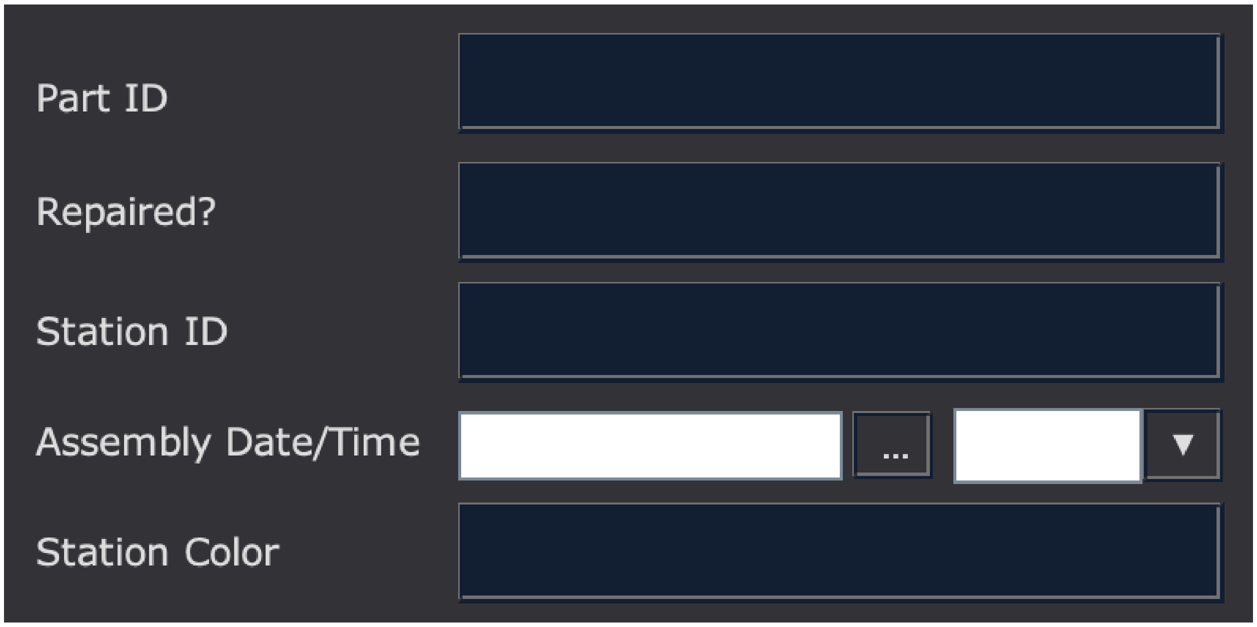

PartStationColortxt [JTextField]


PartRepairedtxrt [JTextField]
PartStationtxt [JTextField]



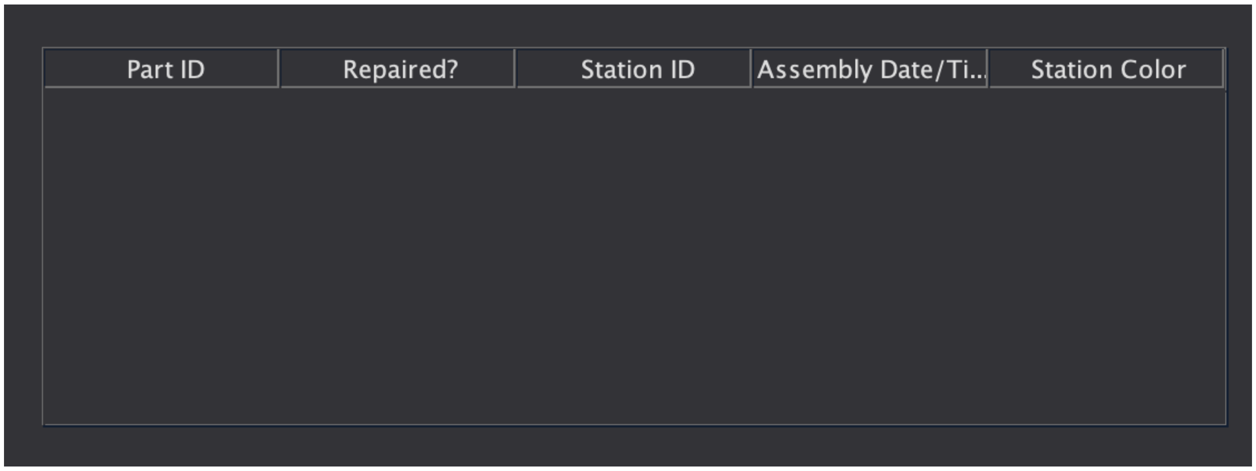
AssemblyDatetxt [DatePicker]

AssemblyTimetxt [TimePicker]



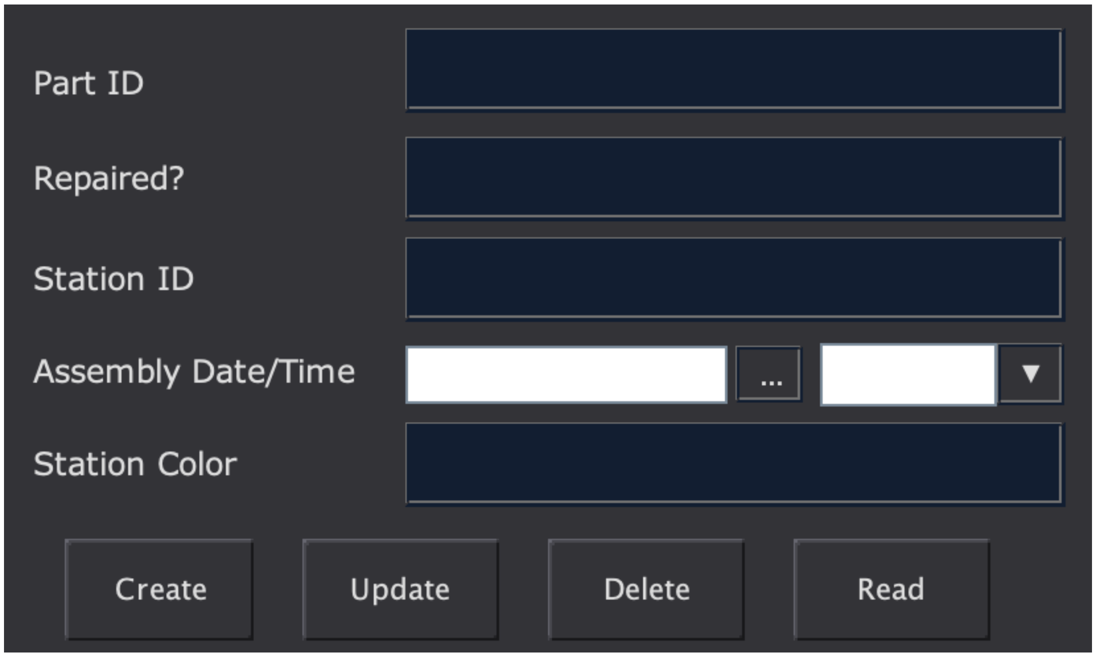
Part ID	
Repaired?	
Station ID	
Assembly Date/Time	
Station Color	

7. Create/Edit the JTable to include attributes from the Part table. Provide screenshots. **1 point.**



Part ID	Repaired?	Station ID	Assembly Date/Ti...	Station Color

8. Create buttons for Part CRUD functionalities with screenshots. (*i.e.*, CREATE, UPDATE, DELETE) **1 point.**



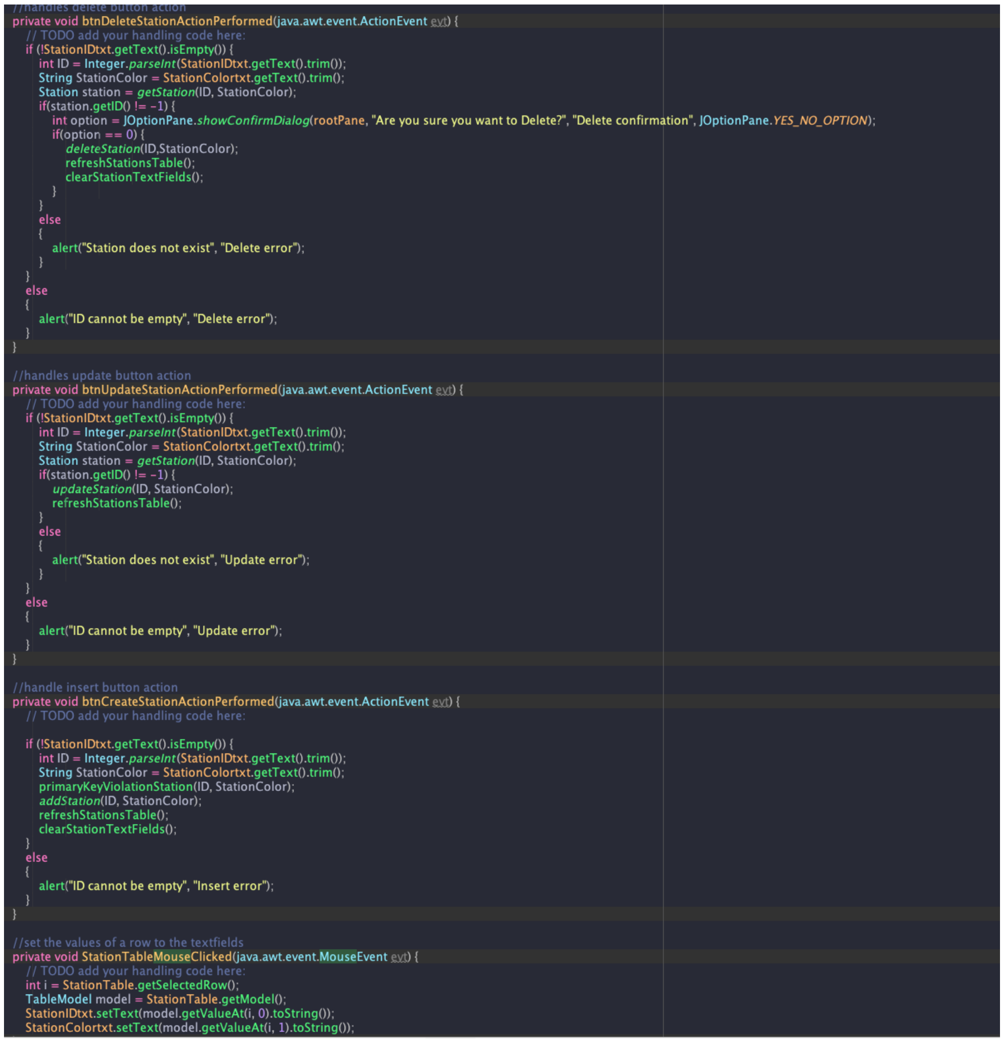
9. Explain the layout used in your GUI and why you choose to use such layout. **2 points.**

   For the Station part, Upon examining the Simio Model. There are multiple Stations which can be represented by an ID, and then each station has a color that represents it. The station color would connect to station ID. For the part of the GUI, I looked at that table in part 4 in order to meet the requirements of that chart when testing the application, along with examining the simio model. The part ID is PK, with the use of Station ID which is FK and Station Color which is also FK that connects to the Station table.

Part 3: Developing the Controller for the GUI (8 pts)

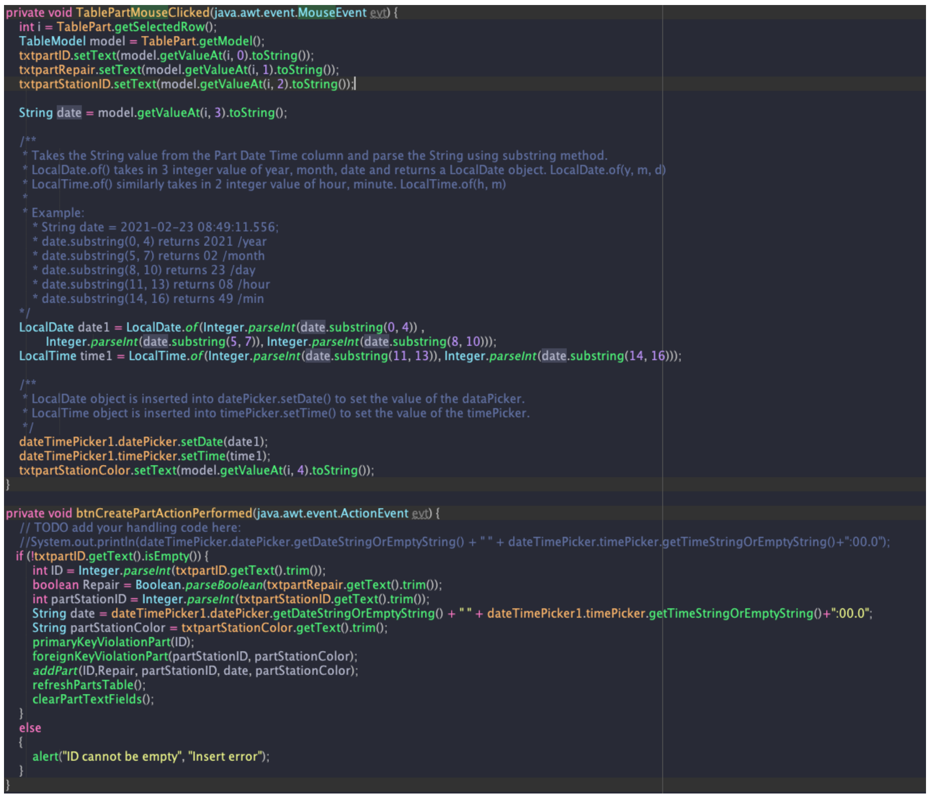
In your Main.java, write the necessary methods and action events for your GUI interface to function. Be sure to use comments to label what each method does and output.

1. Provide a screenshot for each CRUD function method for Station. **4 points.**

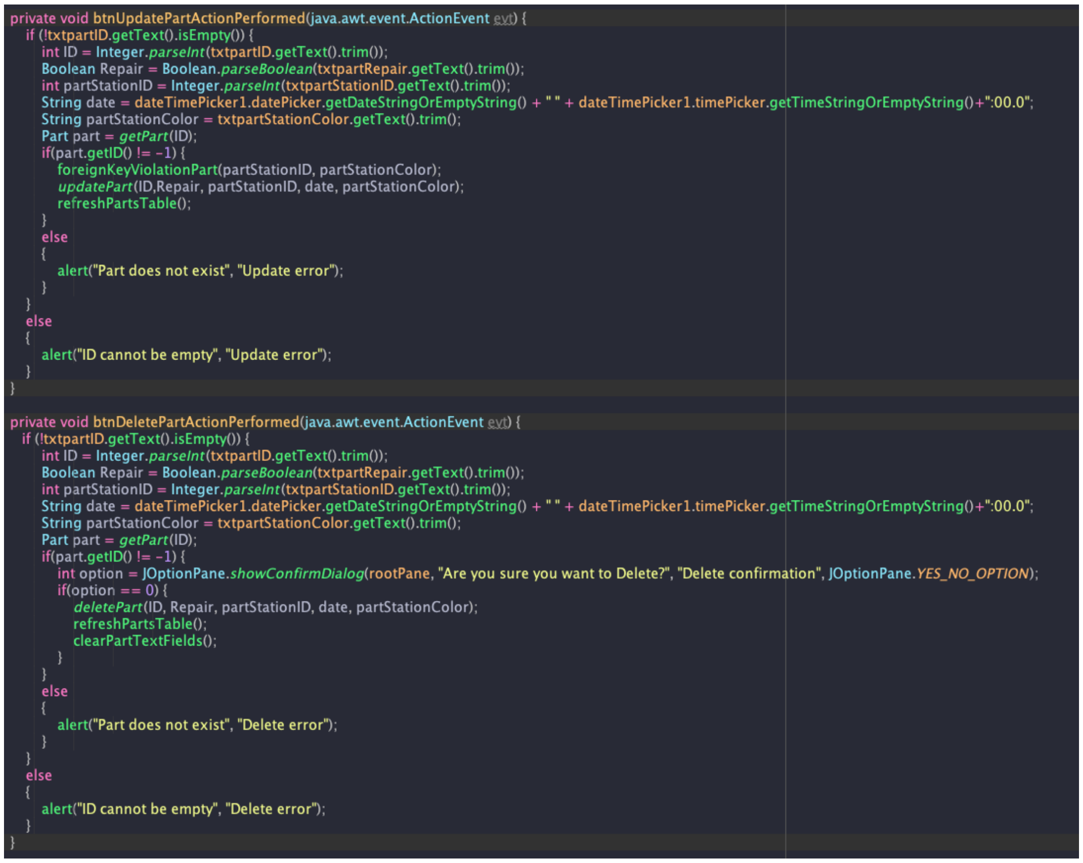


```
//handles delete button action
private void btnDeleteStationActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (!StationIDtxt.getText().isEmpty()) {
        int ID = Integer.parseInt(StationIDtxt.getText().trim());
        String StationColor = StationColortxt.getText().trim();
        Station station = getStation(ID, StationColor);
        if(station.getID() != -1) {
            int option = JOptionPane.showConfirmDialog(rootPane, "Are you sure you want to Delete?", "Delete confirmation", JOptionPane.YES_NO_OPTION);
            if(option == 0) {
                deleteStation(ID,StationColor);
                refreshStationsTable();
                clearStationTextFields();
            }
        }
        else
        {
            alert("Station does not exist", "Delete error");
        }
    }
    else
    {
        alert("ID cannot be empty", "Delete error");
    }
}

//handles update button action
private void btnUpdateStationActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (!StationIDtxt.getText().isEmpty()) {
        int ID = Integer.parseInt(StationIDtxt.getText().trim());
        String StationColor = StationColortxt.getText().trim();
        Station station = getStation(ID, StationColor);
        if(station.getID() != -1) {
            updateStation(ID, StationColor);
            refreshStationsTable();
        }
        else
        {
            alert("Station does not exist", "Update error");
        }
    }
    else
    {
        alert("ID cannot be empty", "Update error");
    }
}

//handle insert button action
private void btnCreateStationActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    if (!StationIDtxt.getText().isEmpty()) {
        int ID = Integer.parseInt(StationIDtxt.getText().trim());
        String StationColor = StationColortxt.getText().trim();
        primaryKeyViolationStation(ID, StationColor);
        addStation(ID, StationColor);
        refreshStationsTable();
        clearStationTextFields();
    }
    else
    {
        alert("ID cannot be empty", "Insert error");
    }
}

//set the values of a row to the textfields
private void StationTableMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int i = StationTable.getSelectedRow();
    TableModel model = StationTable.getModel();
    StationIDtxt.setText(model.getValueAt(i, 0).toString());
    StationColortxt.setText(model.getValueAt(i, 1).toString());
```

2. Provide a screenshot for each CRUD function method for Part. **4 points.**



```java
private void TablePartMouseClicked(java.awt.event.MouseEvent evt) {
   int i = TablePart.getSelectedRow();
   TableModel model = TablePart.getModel();
   txtpartID.setText(model.getValueAt(i, 0).toString());
   txtpartRepair.setText(model.getValueAt(i, 1).toString());
   txtpartStationID.setText(model.getValueAt(i, 2).toString());

   String date = model.getValueAt(i, 3).toString();

   /**
    * Takes the String value from the Part Date Time column and parse the String using substring method.
    * LocalDate.of() takes in 3 integer value of year, month, date and returns a LocalDate object. LocalDate.of(y, m, d)
    * LocalTime.of() similarly takes in 2 integer value of hour, minute. LocalTime.of(h, m)
    *
    * Example:
      * String date = 2021-02-23 08:49:11.556;
      * date.substring(0, 4) returns 2021 /year
      * date.substring(5, 7) returns 02 /month
      * date.substring(8, 10) returns 23 /day
      * date.substring(11, 13) returns 08 /hour
      * date.substring(14, 16) returns 49 /min
   */
   LocalDate date1 = LocalDate.of(Integer.parseInt(date.substring(0, 4)) ,
         Integer.parseInt(date.substring(5, 7)), Integer.parseInt(date.substring(8, 10)));
   LocalTime time1 = LocalTime.of(Integer.parseInt(date.substring(11, 13)), Integer.parseInt(date.substring(14, 16)));

   /**
    * LocalDate object is inserted into datePicker.setDate() to set the value of the dataPicker.
    * LocalTime object is inserted into timePicker.setTime() to set the value of the timePicker.
    */
   dateTimePicker1.datePicker.setDate(date1);
   dateTimePicker1.timePicker.setTime(time1);
   txtpartStationColor.setText(model.getValueAt(i, 4).toString());
}

private void btnCreatePartActionPerformed(java.awt.event.ActionEvent evt) {
   // TODO add your handling code here:
   //System.out.println(dateTimePicker.datePicker.getDateStringOrEmptyString() + " " + dateTimePicker.timePicker.getTimeStringOrEmptyString()+":00.0");
  if (!txtpartID.getText().isEmpty()) {
      int ID = Integer.parseInt(txtpartID.getText().trim());
      boolean Repair = Boolean.parseBoolean(txtpartRepair.getText().trim());
      int partStationID = Integer.parseInt(txtpartStationID.getText().trim());
      String date = dateTimePicker1.datePicker.getDateStringOrEmptyString() + " " + dateTimePicker1.timePicker.getTimeStringOrEmptyString()+":00.0";
      String partStationColor = txtpartStationColor.getText().trim();
      primaryKeyViolationPart(ID);
      foreignKeyViolationPart(partStationID, partStationColor);
      addPart(ID,Repair, partStationID, date, partStationColor);
      refreshPartsTable();
      clearPartTextFields();
   }
   else
   {
      alert("ID cannot be empty", "Insert error");
   }
}
```



```java
private void btnUpdatePartActionPerformed(java.awt.event.ActionEvent evt) {
   if (!txtpartID.getText().isEmpty()) {
      int ID = Integer.parseInt(txtpartID.getText().trim());
      Boolean Repair = Boolean.parseBoolean(txtpartRepair.getText().trim());
      int partStationID = Integer.parseInt(txtpartStationID.getText().trim());
      String date = dateTimePicker1.datePicker.getDateStringOrEmptyString() + " " + dateTimePicker1.timePicker.getTimeStringOrEmptyString()+":00.0";
      String partStationColor = txtpartStationColor.getText().trim();
      Part part = getPart(ID);
      if(part.getID() != -1) {
         foreignKeyViolationPart(partStationID, partStationColor);
         updatePart(ID,Repair, partStationID, date, partStationColor);
         refreshPartsTable();
      }
      else
      {
         alert("Part does not exist", "Update error");
      }
   }
   else
   {
      alert("ID cannot be empty", "Update error");
   }
}

private void btnDeletePartActionPerformed(java.awt.event.ActionEvent evt) {
  if (!txtpartID.getText().isEmpty()) {
      int ID = Integer.parseInt(txtpartID.getText().trim());
      Boolean Repair = Boolean.parseBoolean(txtpartRepair.getText().trim());
      int partStationID = Integer.parseInt(txtpartStationID.getText().trim());
      String date = dateTimePicker1.datePicker.getDateStringOrEmptyString() + " " + dateTimePicker1.timePicker.getTimeStringOrEmptyString()+":00.0";
      String partStationColor = txtpartStationColor.getText().trim();
      Part part = getPart(ID);
      if(part.getID() != -1) {
         int option = JOptionPane.showConfirmDialog(rootPane, "Are you sure you want to Delete?", "Delete confirmation", JOptionPane.YES_NO_OPTION);
         if(option == 0) {
            deletePart(ID, Repair, partStationID, date, partStationColor);
            refreshPartsTable();
            clearPartTextFields();
         }
      }
      else
      {
         alert("Part does not exist", "Delete error");
      }
   }
   else
   {
      alert("ID cannot be empty", "Delete error");
   }
}
```

Part 4: Testing the Application with Simulation Data (30 pts)

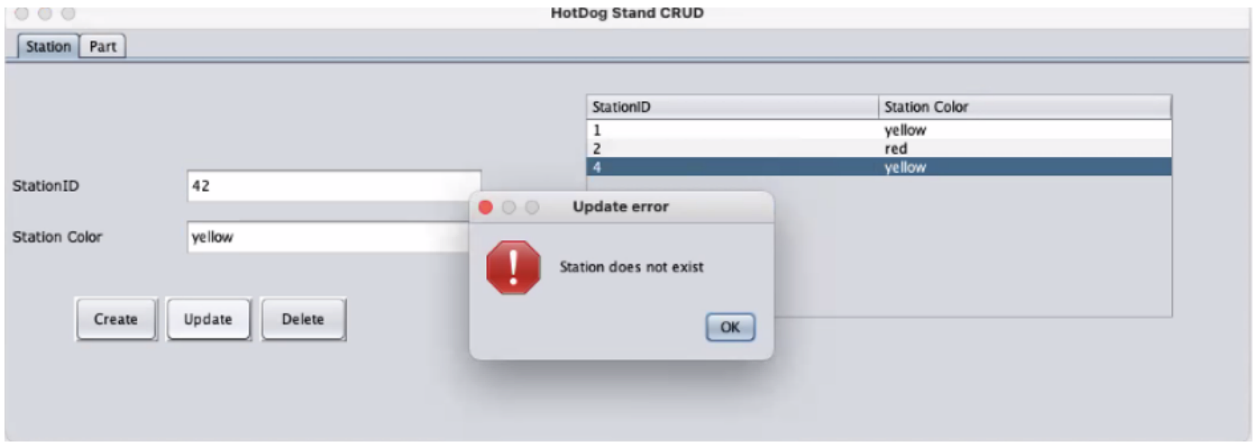
Part ID (Given in Simulation)	Is it Repaired	Which Station Part Has Visited	Date Time the Part Has Assembled	Station Color
84	false	1	01/01/08 12:03:12	yellow
85	fasle	1	01/01/08 12:04:05	yellow

https://psu.zoom.us/rec/share/3FO2nDB5kCmkYrVi3OXhfghjYPTrgwnKnh7l9fMVEFGATxOgTMQ8nvs6P8D35WsD.BzkwWeJJ7tWGQl58?startTime=1637036762000

https://psu.zoom.us/rec/share/3FO2nDB5kCmkYrVi3OXhfghjYPTrgwnKnh7l9fMVEFGATxOgTMQ8nvs6P8D35WsD.BzkwWeJJ7tWGQl58?startTime=1637037866000

1. Pick two Parts and record its information and Station data into the table above. **2 points.**
2. Show how you add a new Station with screenshots. **3 points.**
3. Show how you update an existing Station with screenshots. **3 points.**
4. Show how you print/list Stations with screenshots. **3 points.**
5. Show how you delete an existing Station with screenshots. **3 points.**
6. Show how you add a new Part with screenshots. **3 points.**
7. Show how you update an existing Part with screenshots. **3 points.**
8. Show how you print/list Parts with screenshots. **3 points.**
9. Show how you delete an existing Part with screenshots. **3 points.**
10. Show a primary key, how you print the error to the screen with screenshots. You must use ShowDialog to print the error or exception. Command line errors are not accepted. **2 points.**

Station PK Violation



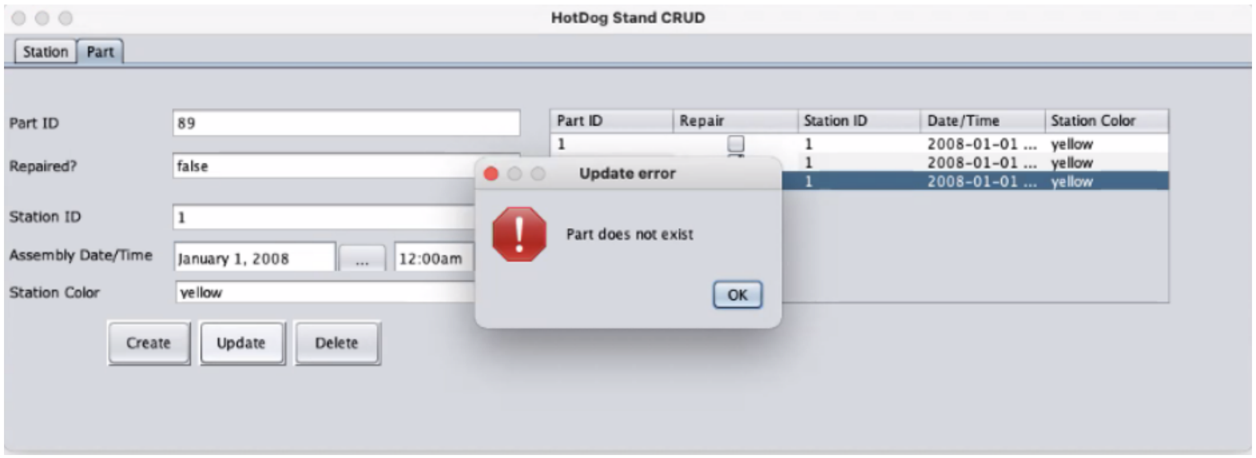
StationID: 42
Station Color: yellow

StationID	Station Color
1	yellow
2	red
4	yellow

Update error

Station does not exist

Part PK Violation


HotDog Stand CRUD

Station | Part

Part ID	89
Repaired?	false
Station ID	1
Assembly Date/Time	January 1, 2008 ... 12:00am
Station Color	yellow

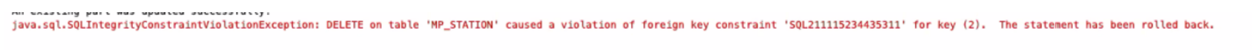
Part ID	Repair	Station ID	Date/Time	Station Color
1	☐	1	2008-01-01 ...	yellow
		1	2008-01-01 ...	yellow
		1	2008-01-01 ...	yellow

Update error

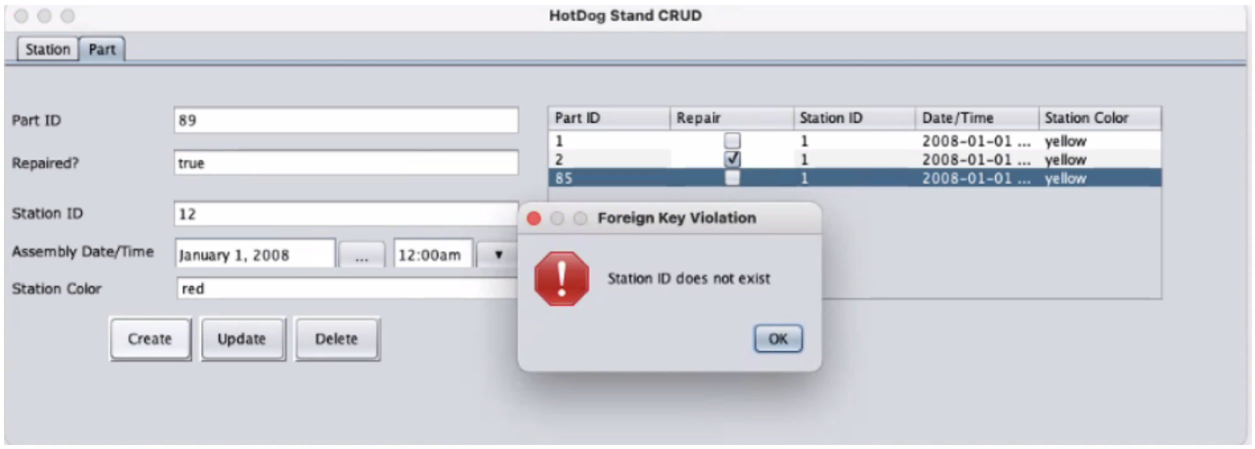
Part does not exist

11. Show a foreign key violation, how you print the error to the screen with screenshots. You must use ShowDialog to print the error or exception. Command line errors are not accepted. **2 points.**

Station FK Cascade Violation


```
java.sql.SQLIntegrityConstraintViolationException: DELETE on table 'MP_STATION' caused a violation of foreign key constraint 'SQL211115234435311' for key (2).  The statement has been rolled back.
```

Part FK Violation


HotDog Stand CRUD

Station | Part

Part ID	89
Repaired?	true
Station ID	12
Assembly Date/Time	January 1, 2008 ... 12:00am
Station Color	red

Part ID	Repair	Station ID	Date/Time	Station Color
1	☐	1	2008-01-01 ...	yellow
2	✓	1	2008-01-01 ...	yellow
85	☐	1	2008-01-01 ...	yellow

Foreign Key Violation

Station ID does not exist