



## Task:

You are working for a small manufacturing organization that has outgrown its current inventory system. They have been using a spreadsheet program to manually enter inventory additions, deletions, and other data from a paper-based system but would now like you to develop a more sophisticated inventory program.

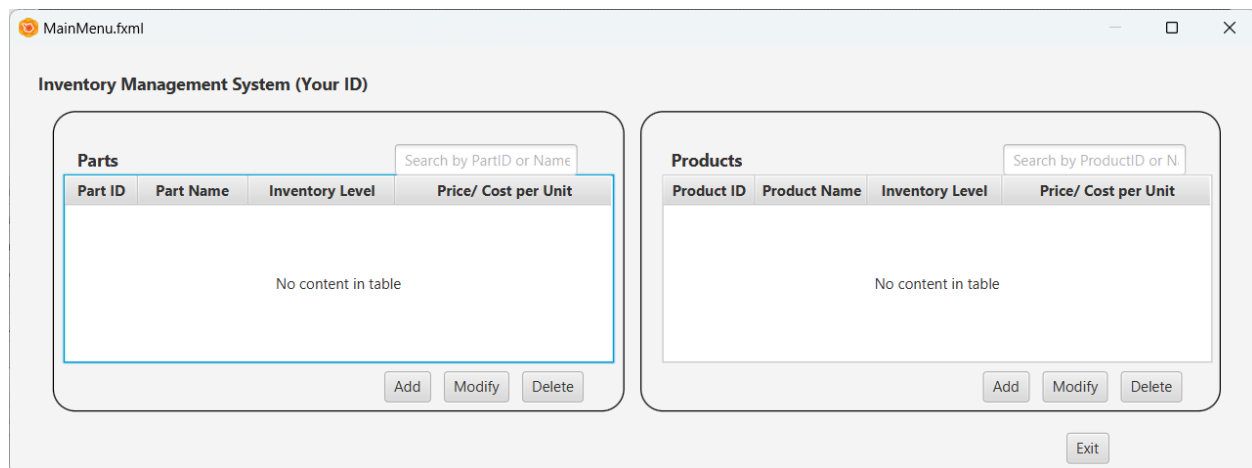
The organization also has specific business requirements that must be included as part of the application. A system analyst from your company created the solution statements outlined in the requirements section based on the manufacturing organization's business requirements. You will use these solution statements to develop your application.

Create a JavaFX application with a graphical user interface (GUI). Write code to display each of the following screens in the GUI:

## Front-End

### Main Screen

- A. A main screen, showing the following controls:
  - i. Buttons for “Add”, “Modify”, “Delete”, “Search” for parts and products, and “Exit”
  - ii. Lists for parts and products
  - iii. Text boxes for searching for parts and products
  - iv. Title labels for parts, products, and the application title



### Parts Screen

- B. An add part screen, showing the following controls:
  - i. Radio buttons for “In-House” and “Outsourced” parts
  - ii. Buttons for “Save” and “Cancel”

- iii. Text fields for ID, name, inventory level, price, max and min values, and company name or machine ID
- iv. Labels for ID, name, inventory level, price/cost, max and min values, the application title, and company name or machine ID

The screenshot shows a window titled 'addPartForm.fxml' with standard window controls (minimize, maximize, close). The form is titled 'Add Part' and features two radio buttons: 'In-House' (selected) and 'Outsourced'. Below these are several text input fields: 'ID' (with a disabled 'Auto Gen - Disabled' button), 'Name', 'Inv', 'Price/Cost', 'Max', and 'MachineID'. A 'Min' label is positioned next to a text field. At the bottom are 'Save' and 'Cancel' buttons.

**Add Part** ☒ In-House ☐ Outsourced

ID

Name

Inv

Price/Cost

Max  Min

MachineID

- C. A modify part screen, with fields that populate with pre-saved data, showing the following controls:
- i. Radio buttons for “In-House” and “Outsourced” parts
  - ii. Buttons for “Save” and “Cancel”
  - iii. Text fields for ID, name, inventory level, price, max and min values, and company name or machine ID

- iv. Labels for ID, name, inventory level, price, max and min values, the application title, and company name or machine ID

**Modify Part** ☒ In-House ☐ Outsourced

ID

Name

Inv

Price/Cost

Max  Min

MachineID

## Product Screen

- D. An add product screen, showing the following controls:
  - i. Buttons for “Save”, “Cancel”, “Add” part, and “Delete” part
  - ii. Text fields for ID, name, inventory level, price, and max and min values

- iii. Labels for ID, name, inventory level, price, max and min values, and the application
- iv. A list for associated parts and their products
- v. A “Search” button and a text field with an associated list for displaying the results of the search

The screenshot shows a window titled 'addProductform.fxml' with a standard macOS-style title bar. Inside the window is a form titled 'Add Product'. The form contains several input fields: 'ID' with a dropdown menu currently showing 'Auto Gen - Disabled', 'Name' with a text input field, 'Inv' with a text input field, 'Price' with a text input field, and 'Max' and 'Min' with separate text input fields. To the right of these fields is a search bar labeled 'Search by Part ID or Nam'. Below the search bar is a table with the following headers: 'Part ID', 'Part Name', 'Inventory Level', and 'Price/ Cost per Unit'. The table is currently empty, displaying 'No content in table'. Below this table is an 'Add' button. Further down is another identical table with the same headers, also empty and displaying 'No content in table'. At the bottom right of the form are three buttons: 'Remove Associated Part', 'Save', and 'Cancel'.

- E. A modify product screen, with fields that populate with pre-saved data, showing the following controls:
- i. Buttons for “Save”, “Cancel”, “Add” part, and “Delete” part
  - ii. Text fields for ID, name, inventory level, price, and max and min values
  - iii. Labels for ID, name, inventory level, price, max and min values, and the application
  - iv. A list for associated parts and their products
  - v. A “Search” button and a text field with associated list for displaying the results of the search

**Modify Product**

Search by Product ID or N

ID

Name

Inv

Price

Max  Min

Part ID	Part Name	Inventory Level	Price/ Cost per Unit
No content in table			

Add

Part ID	Part Name	Inventory Level	Price/ Cost per Unit
No content in table			

Remove Associated Part

Save Cancel

## Backend

Now that you've created the GUI, write code to create the class structure provided in the attached "UML Class Diagram". Enable each of the following capabilities in the application:

- F. Using the attached "UML Class Diagram", create appropriate classes and instance variables with the following criteria:
- Five classes with the associated instance variables.
  - Variables are only accessible through getter methods.
  - Variables are only modifiable through setter methods.

Note: The UML Class Diagram may be altered so long as the aspects of the current UML diagram are intact and the changes applied do not provide a work around for key aspects, such as

- Inheritance.
- Abstraction.
- Encapsulation of the data.

G. Add the following functionalities to the main screen, using the methods provided in the attached “UML Class Diagram”:

- i. Redirect the user to the “Add Part”, “Modify Part”, “Add Product”, or “Modify Product” screens
- ii. Delete a selected part or product from the list
- iii. Search for a part or product and display matching results
- iv. Exit the main screen

H. Add the following functionalities to the part screens, using the methods provided in the attached “UML Class Diagram”:

1. “Add Part” screen

- a. Select “In-House” or “Outsourced”
- b. Enter name, inventory level, price, max and min values, and company name or machine ID
- c. Save the data and then redirect to the main screen
- d. Cancel or exit out of this screen and go back to the main screen

2. “Modify Part” screen

- a. Select “In-House” or “Outsourced”
- b. Modify or change data values
- c. Save modifications to the data and then redirect to the main screen
- d. Cancel or exit out of this screen and go back to the main screen

I. Add the following functionalities to the product screens, using the methods provided in the attached “UML Class Diagram”:

1. “Add Product” screen

- a. Enter name, inventory level, price, max and min values, and company name or machine ID
- b. Save the data and then redirect to the main screen
- c. Associate one or more parts with a product
- d. Remove or disassociate a part from a product
- e. Cancel or exit out of this screen and go back to the main screen

2. “Modify Product” screen

- a. Modify or change data values
- b. Save modifications to the data and then redirect to the main screen
- c. Associate one or more parts with a product
- d. Remove or disassociate a part from a product
- e. Cancel or exit out of this screen and go back to the main screen

- J. Write code to implement exception controls with custom error messages for each of the following sets:

Set 1

- Entering an inventory value greater than the maximum value for a part or product, or lower than the minimum value for a part or product
- Preventing the minimum field from having a value above the maximum field
- Preventing the maximum field from having a value below the minimum field
- Ensuring that a product must always have at least one part

Set 2

- Preventing the user from deleting a product that has a part assigned to it
- Including a confirm dialogue for all “Delete” and “Cancel” buttons
- Ensuring that the price of a product cannot be less than the cost of the parts
- Ensuring that a product must have a name, price, and inventory level (default 0)

Note: UML diagram is attached as a PDF.





