Second Milestone: Implementation Guidelines

1. Implementation

Continue with a BlueJ project created for first milestone named as **LMU StudentName (Example: 24046344 Aaditya Sharma)** and implement the following task:

2. GymGUI Class

Create **GymGUI** class that includes an **ArrayList** (not a simple array) to store objects of both **RegularMember** and **PremiumMember** objects.

There should be following text fields for entering:

1. Id
2. Name
3. Location
4. Phone
5. Email
6. Gender
7. DOB – dobYearComboBox, dobMonthComboBox, dobDayComboBox
8. Membership Start Date - msYearComboBox, msMonthComboBox, msDayComboBox
9. Referral Source
10. Paid Amount
11. Removal Reason
12. Trainer's Name

Also, there should be **a non-editable text field** for regular plan price, premium plan charge and discount amount.

The gender should be displayed as a **radio button.**

The **Plan (Basic, Standard, Deluxe)** attribute should be displayed as a **JComboBox** for plan selection for regular members only.

The GUI should have the following buttons to perform specific task:

1. **Add a Regular Member**

   When this button is pressed, the input values from the text fields (ID, Name, Location, Phone, Email, Gender, DOB, Membership Start Date,Referral Source) will be used to create a RegularMember object. This object will then be added to an ArrayList of the GymMember class.
   Note: Member ID duplication is prohibited. Each member, whether Regular or Premium, must have a unique Member ID]

2. **Add a Premium Member**

   When this button is pressed, the input values from the **text fields** (ID, Name, Location, Phone, Email, Gender, DOB, Membership Start Date, Personal Trainer) will be used to create an object of Premium member and this object will then be added to an ArrayList of the GymMember class.

[Note: Member ID duplication is prohibited. Each member, whether Regular or Premium, must have a unique Member ID.]

3. **Activate Membership**

The **Member Id** should be entered in the GUI to activate membership. When this button is pressed, the input value of Member Id is **compared** to the existing Member Id. If the entered Member Id is **valid** and **exists** in the system, then **call** the method to activate membership from GymMember class.

4. **Deactivate Membership**

The **Member Id** should be entered in the GUI to deactivate membership. When this button is pressed, the input value of Member Id is **compared** to the existing Member Id. If the entered Member Id is **valid** and **exists** in the system, then **call** the method to deactivate membership from GymMember class.

5. Mark Attendance

First, the **Member Id** should be entered in the GUI to mark attendance. When this button is pressed, the input value of Member Id is **compared** to the existing Member Id. If the entered Member Id is **valid** and **exists** in the system, then call the method to mark attendance from GymMember class.

[Note: if only active status of member is true, proceed with mark attendance]

8. Revert Member

To remove a member, the MemberID is entered in the GUI. When this button is pressed the, the input value of Member Id is **compared** to the existing Member Id. If the entered Member Id is **valid** and **exists** in the system and then **call** the method to revert member from both RegularMember and PremiumMember class.

[Note: there should be two separate buttons for reverting the members of each class]

10. Display

When this button is pressed, the information of all the members of the respective class should be displayed in a separate frame/panel in the GUI.

11. Clear

When this button is pressed the entered values from text fields should be cleared and the check box should be unchecked.

Additional Information to be included:

- Retrieve the values from each text field by using the getText() method.
- For numeric variables, extract the text from the text field, convert it to an integer, and return the result as a whole number.
- Use try and catch blocks to handle potential NumberFomatException with that may occur during the conversion of the string to an integer or double.
- In case of successful operations, display an appropriate success message to the user using a message dialog box.
- In case of incorrect input, display an appropriate error message to the user using a message dialog box.

3. Testing

You should give evidence (through inspection tables and appropriate screenshots) of the following testing that you carried out on your program:

Test 1: Compile and run the program using command prompt/terminal.
Test 2: Adding regular member and premium member respectively.
Test 3: Test case for mark attendance

For Example:

Table 1: Test-1: Add a regular student.

Objective	To add a regular student.
Action	All the required fields were filled, and 'Add Student' button was clicked.
Expected Output	The regular student details should be added successfully with an appropriate message dialog box.
Actual Output	The regular student details were added successfully with an appropriate message dialog box.
Result	The test was successful.

4. Report

Prepare report for Second Milestone including following topics:

- Cover Page
- Turnitin Similarity Report
- Table of Contents
- Screenshot of Developed GUI
- Testing – Test 1, Test 2, Test 3

5. Submission:

- Submit a **Word document via the Turnitin submission link**, ensuring that the file **excludes** the cover page, table of contents, and references.
- Create and submit **a ZIP file via MST portal** named **LMU StudentName**, containing the following:
  1. **Java Files:**
     - GymGUI.java
     - GymMember.java
     - RegularMember.java
     - PremiumMember.java
  2. **Report:**
     - A PDF version of the second milestone report.