Project 4
Implementation

Fall 2023
Deadline: 11/12/2023, 11:59 PM CST

Practical Information

Question:
Ask your questions on Slack or make an appointment to meet with the TA.

Teams:
As Assigned

- **There is no Blackboard submission for this assignment.** The TAs will clone your repository when it is due and run the application, later commits than the deadline will not be pulled.
- After your submission, there will be a demo in which you will run and demonstrate the functionalities of your code. The demos will take place in weeks 12 and 14. Scheduling for the demos will be implemented with ample time for preparation.
- You will receive two grades for this project: 90 points for the whole implementation tasks as a team and 30 points for the unit tests as an individual for a total of 120 points. There will also be an additional 18 bonus points for optional features. Additional extra credit points will be given at TA's discretion.

Goals

This assignment will make you familiar with the object-oriented implementation of a complete software. The grading scheme is as follows:

1. ANT configuration + modified to create JAR and put under release (10%)
2. Java source code (55%)
    a. GUI (5%) (Optional/Bonus 6pts)
    b. Persistence (5%) (Optional/Bonus 6pts)
3. JUnit tests (25%)
4. Generated JavaDocs (5%)
5. Short user manual (5%)
6. Correct (unmessy) repo structure (5%) (Optional/Bonus 6pts)

A Sample Correct Repo Structure

- The repo structure may be in the following format to be counted as "correct". https://bitbucket.org/azaman2/fall23teamx

Repo Contents and Assignments

- src/ - Source code for the project + JUnit tests
- doc/ - Javadoc auto-generated

- design/ - If you have updated any of your class/sequence/activity diagrams. (Meaning they are different than the first three assignments, put the UMLentino files here)
- manual/ - Put the short user manual pdf or doc here
- release/ - Put the auto-generated jar file here using ANT
- old/- Put all previous project files from Projects 1, 2, and 3 here.
- build.xml - The ant script of the project

Given the previous Requirement, Analysis, and Design phases you have performed for the Chocoholics Anonymous (ChocAn) project, in this assignment, you will implement the system.

Task 1

Implement the software such that it supports the functionalities of all the use cases you have identified in Java. Your **class diagram must reflect the implementation**. The functions to implement are defined by the sequence diagrams. Make sure you have one public method per sequence diagram. Use coding conventions you mutually agree with your teammates. Don't forget to **put comments in your code**. Programming should be **spread evenly** among all team members: each team member must be assigned nearly the same number of classes to implement (remember to use documentation and knowledge transfer meetings for others to understand your code). In the comment above **each class, specify who the author of that class is**. Communication among you is very important, especially if your module is interacting with someone else's. The code must be written using the Eclipse IDE which offers great support for debugging. If you decide to use Visual Studio Code or any other editors/IDEs, you are responsible for understanding the intricacies related to coding through them. Note that you are not required to provide a graphical user interface (GUI). **A command line menu is sufficient; however, a GUI is 5% bonus**. **Persisting the users/records etc. between sessions is another 5% bonus.** *Submitting a program that does not compile will lead to an automatic ZERO (0) for the Java Source Code score.*

Task 2

Choose one feature per team member (from sequence diagrams) to unit test using JUnit. **Each team member must write one set of unit tests.** You must **write a unit test for at least 2 methods you implement and 1 method you did not implement**. If Bob implemented method m() and n(), while Jane implemented method x() and y(), then Bob should unit test m(),n() and one of x() or y(). Jane should unit test x(), y() and one of m(),n(). This also means that **you will loose at least 30 points (possibly more) if you did not contribute to any of the code**. While writing unit tests, try to be as detailed as possible. You should at least test for success and failure. **Each member should have one unit test file with at least 3 test cases in it. When each test is performed during the Demo, a green "Pass" line for each test must be displayed.**

Task 3

After your code has been written, tested, and completed, you should **generate the JavaDocs** from the code. This will help future programmers locate and understand better the functionalities you implemented. This will also facilitate them to make changes and interact with your program. **Each team member** should have **at least 1 class (file) fully commented**. Produce a **short user manual** (1 to 3 pages) on how to use the ChocAn program you developed over the semester. **The manual should describe/explain to the TA "how to run the program" and "how to test each feature as in a step-by-step guide. Explain each step from cloning the repo to creating the JAR, to running the tests.** Note that you may need to revise any artifact (use cases, activity diagram, class

diagram, sequence diagram) you previously produced during the past assignments as you are actually implementing the system. Additionally, this manual must **include a *Task Distribution* section**. This section should outline all the tasks that were performed and what percentage each team member contributed.

Other Tasks

Build the ANT script from your project and modify it. Let the **ANT script create a JAR** and put it **without asking you under a "Release"** folder. Be sure the JAR file is running successfully. This is required and will be put to the test during the Demo. **ALL TEAM MEMBERS SHOULD DO COMMITS TO THE TEAM REPOSITORY. COMMITS WILL BE CHECKED AS USUAL.**

Additional Resources

The IDE to use is Eclipse. If using any other editor/IDE, you are responsible to understand its functionalities. Please refer to Eclipse Documentation for a User Guide of Eclipse functionality.