CS 200 **Project 4** **120 points**
Fall 2023 **Implementation & Testing** **Due November 20, 2023 @ Noon**

Practical Information

- 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.
- There will be a Demo after your submission where you will run and demonstrate the functionalities of your code. The Demo will be the last week of class. Scheduling of demos will be implemented with ample time to prepare. Teams will meet with their designated TA via Zoom to complete the demo online.
- 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 can 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:

- ANT configuration + modified to create JAR and put under release (10%)
- Java source code (50%)
  - GUI (5%) (Optional/Bonus 6pts)
  - Persistence (5%) (Optional/Bonus 6pts)
- JUnit tests (25%)
- Generated JavaDocs (5%)
- Short User Manual (5%)
- Assignments and ReadMe Team repo folders (5%)
  - Team Repo folders should include: Assignments, ReadMe, Project 1 – Analysis, Project 2 – Analysis, Project 3 – Design, Project 4 – Implementation and Testing
- Correct (unmessy) Project 4 Eclipse structure (5%) (Optional/Bonus 6pts)

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 – Java Source Code

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 upon with your team mates.

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 the team 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 – JUnit Testing

Choose one feature per team member (from sequence diagrams) to unit testing 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(). **You will lose 30 points if you do not contribute any test 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.**

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**.

Task 3

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 user "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, etc.) you previously produced during the past assignments as you are actually implementing the system.

Other Tasks

A. ANT/JAR

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. Points can be deducted for an insufficient number of commits.**

B. Correct Repo Name

The repo name must be in the following format to be counted as "correct".
`https://bitbucket.org/tklocklear/Fall2023TeamX.git`

C. Team Repo Structure

- Assignments – Upload ChocAn.pdf, Project 1 – 4 Assignments, and Secure Agile Development Lifecycle.
- Project 1 Requirements
- Project 2 Analysis
- Project 3 Design
- Project 4 Implementation and Testing
- Readme – Explains each phase of your Project throughout the Software Development Lifecycle (SDLC). Should have the following sections:
    - Overview – ChocAn software program for CS200 Software Engineering and Design class for Fall 2023. Over the course of the semester, we completed the Requirements (Project 1), Analysis (Project 2), Design (Project 3), and Implementation and Testing (Project 4) phases of the Software Development Lifecycle (SDLC). The project implemented Secure and Agile Software Development processes. Project 1 – 4 assignments can be viewed under the Assignments folder. Solutions for Project 1 – 3 can be viewed via each Report.html file. The final software program can be run by the JAR under the release folder in Project 4. (PROVIDED)
    - Project 1 Requirements – List each artifact in the Report.html. (TODO)
    - Project 2 Analysis– List each artifact in the Report.html. (TODO)
    - Project 3 Design – List each artifact in the Report.html. (TODO)
    - Project 4 Implementation and Testing - List each folder and briefly describe their contents. (TODO)

D. Project 4 Eclipse Folders

- `src/` – Source code for the project + JUnit tests.
- `header/` - Header table for Project 4.
- `taskdistribution/` - Task distribution table for Project 4. Points can be deducted for inequal task distribution and percentages for individuals.
- `doc/` – Javadoc auto-generated.
- `manual/` – Put the short user manual pdf or doc here.
- `release/` - Put the auto-generated JAR file here using ANT.
- `build.xml` – The ANT script of the project.

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.