CS 200 | **Project 4** | **120 points**
Fall 2017 | **Implementation** | Due Dec 1, 2017 - 11:59pm

Practical information

- Ask your questions on Slack or make an appointment to meet with the TA.
- Teams: As Assigned.
- **There is no submission in this assignment.** The TA will clone your repository when it is due and run the application, later commits than the deadline will not be pulled.
- 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.

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 (55%)
  - GUI (5%) (Optional/Bonus)
  - Persistence (5%) (Optional/Bonus)
- JUnit tests (25%)
- Generated Javadocs (5%)
- Short user manual (5%)
- Correct (unmessy) repo structure (5%) (Optional/Bonus)

A sample correct repo structure

The repo structure may be in the following format to be counted as "correct".

- `https://bitbucket.org/syue2/cs200fall2017teamX/`

`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 visual paradigm files here
`manual/`	:	Put the short user manual pdf or doc here
`release/`	:	Put the auto-generated jar file here using ANT
`old/`	:	Put the all of the old stuff here about projects 1,2,3
`build.xml`	:	The ant script of the project

Assignments
Given the requirement, analysis, and design 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 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. 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. Note that you are not required to provide a graphical user interface. **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 -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 a method you did not implement**. If Bob implemented method m() and Jane implemented method n(), then Bob should unit test n() and Jane should unit test m(). 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.**

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 let the TA run the program and test various features without asking you**. 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 **under release** folder. Be sure jar file is running successfully.

ALL TEAM MEMBERS SHOULD DO COMMITS TO THE TEAM REPOSITORY.
Additional Resources
The IDE to use is Eclipse.