CSCI 430 Project 1. Implementation for Stage 1.

Due date: Monday September 30.

In Stage 1, we implement the following operations: *add client, add product* and *add product to client's wishlist,* **plus** *any queries needed to verify (test) that these features have been implemented correctly.*
Designing the Classes. *For each class, create two items: (1) a table like shown in Chapter 5, page 34 (Table 5.2) explaining how the methods were found in the sequence diagrams. (2) A class diagram like the ones shown in Fig 5.12 on Page 35.*
Notes. To implement this, each member in the group must take responsibility for a few (1 or 2, maybe 3) of the classes. A situation where everyone works on all the classes generally causes confusion and duplication. Once all the sequence diagrams have been finalized for the features that need to be implemented, then we can easily deduce what methods are needed in each class. After coding the class, each group member can do some **unit testing,** i.e., testing one class separately with a driver, or **integration testing,** i.e., testing a few classes together with a driver class.

Individual Code. Each group member should test their own classes separately using a separate driver. In the Github repository there is a **BookTest** folder that shows how the Book and Catalog classes for Library can be tested using a "dummy member class". This is an example of an **integration test** for Book and Catalog.
Group Code. This has to be submitted by the group leader (or whoever in the group takes the responsibility for code submission); leader has to compile and test all the code in their Github account in a newly created **WarehouseStage1** folder. See the code **LibraryStage1** as an example. There is a lot of code in there, that you can "reuse" for your implementation.

What to submit:
1. **Group code in Github.** Only one per group. This will be the entire implementation for all the warehouse operations listed above. Once your code is working correctly, start a script session and test the code thoroughly. The testing should include populating the system with several clients and products, multiple tests to add products to wishlists, and several queries verifying that the tests were done accurately
2. **Individual code in Github.** Every student should write the code for one or two classes and do a unit/integration test. Once your code is working correctly, start a script session and test the code thoroughly.
3. **Individual report. Submitted in D2L**. Each student has to write a report describing what classes they coded, when they completed the coding, and how they contributed to the group. List your **starid** and how the code is to be compiled and tested in Github.
4. **Group Report and Documentation. Submitted in D2L.** The documentation consists of the items described above under "**Designing the Classes".** Person responsible for submitting code has to write a report, giving **starid** and how the code is to be compiled and tested in Github. Please include any comments about individual (non-)performance, and work-sharing percentages if needed. (If very unequal work-sharing points will be re-adjusted accordingly.)