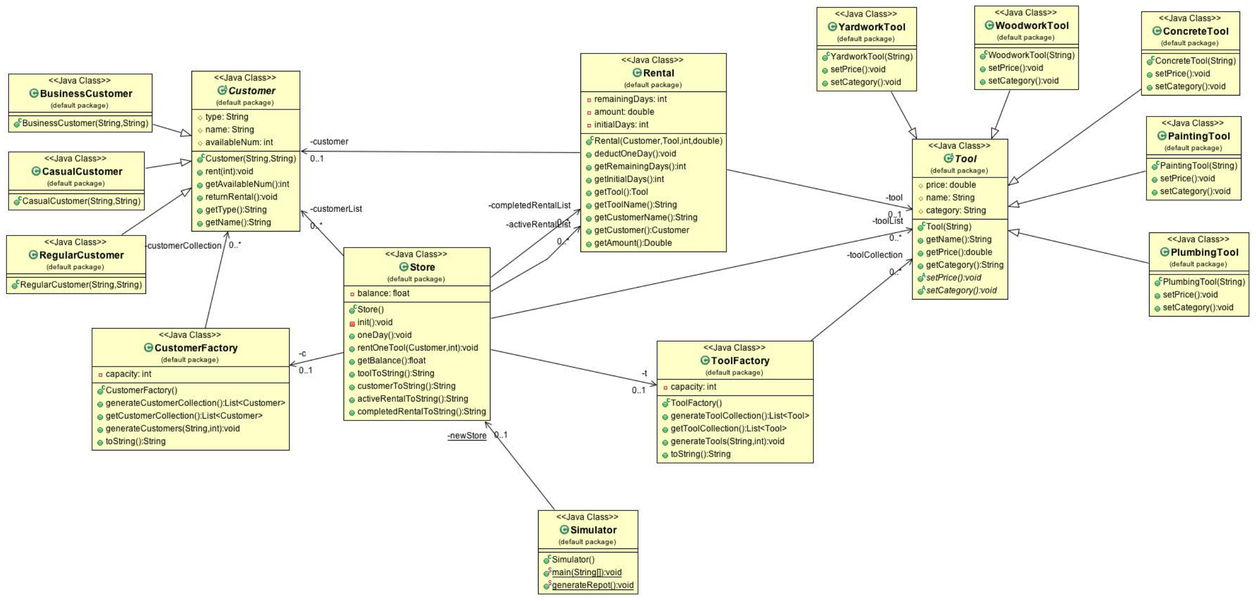
HW3

Team members:
Mutian Yan
Evan Dorrough

UML:



Description:
In this project, we separate the design abstraction from the actual code implementation. In the beginning, we write the skeleton and go over the functionalities that we are supposed to support and then come up with this hierarchical structure. The simulator class is the entrance of the program, which simply has an instance of the Store object. In the future, if we tend to run several stores at the same time by the simulator, it will be just very minimal code change. The store class has all the functionalities that we expected the store to run for a day: when customer come in and rent customer tools, at the end of the day check the due tools and help them process return. The Store class has a collection of tools and a collection of customers. The instantiation of those two collections is through invoking the CustomerFactory class or ToolFactory class separately. Inside the CustomerFactory class, the Customer objects are generated. The Customer classes is an abstract class which cannot be instantiated directly. Same rules apply to the parallel structure, the Tool class. Rental class documents the detail of each rental transaction. The Store holds two collections (list) of Rentals. Specifically, one named activeRentalList is to help keep track of all the active rentals. The other one named completedRentalList is aiming to trace all the rentals.

Design Pattern used: simple Factory, Façade