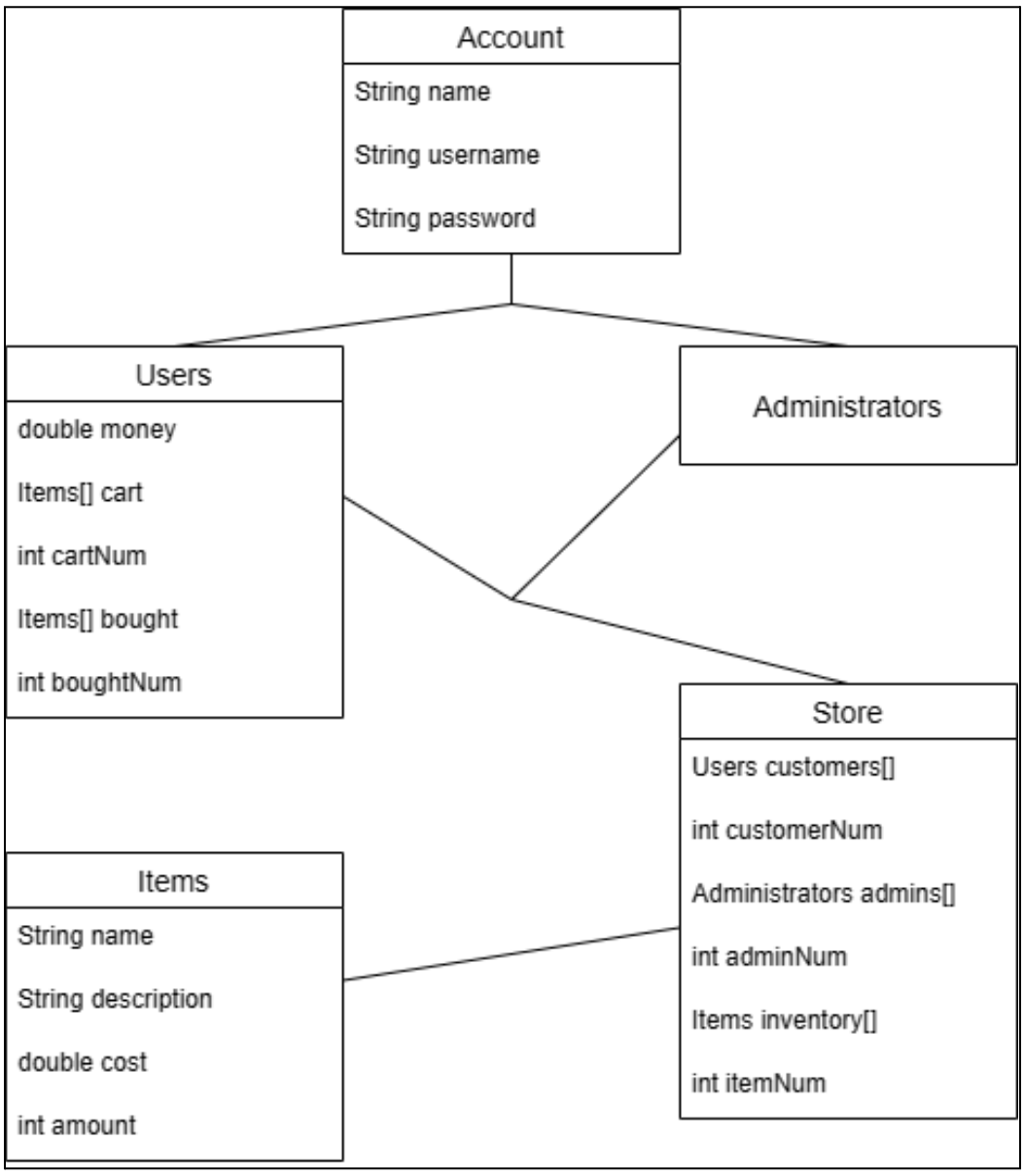
Introduction:

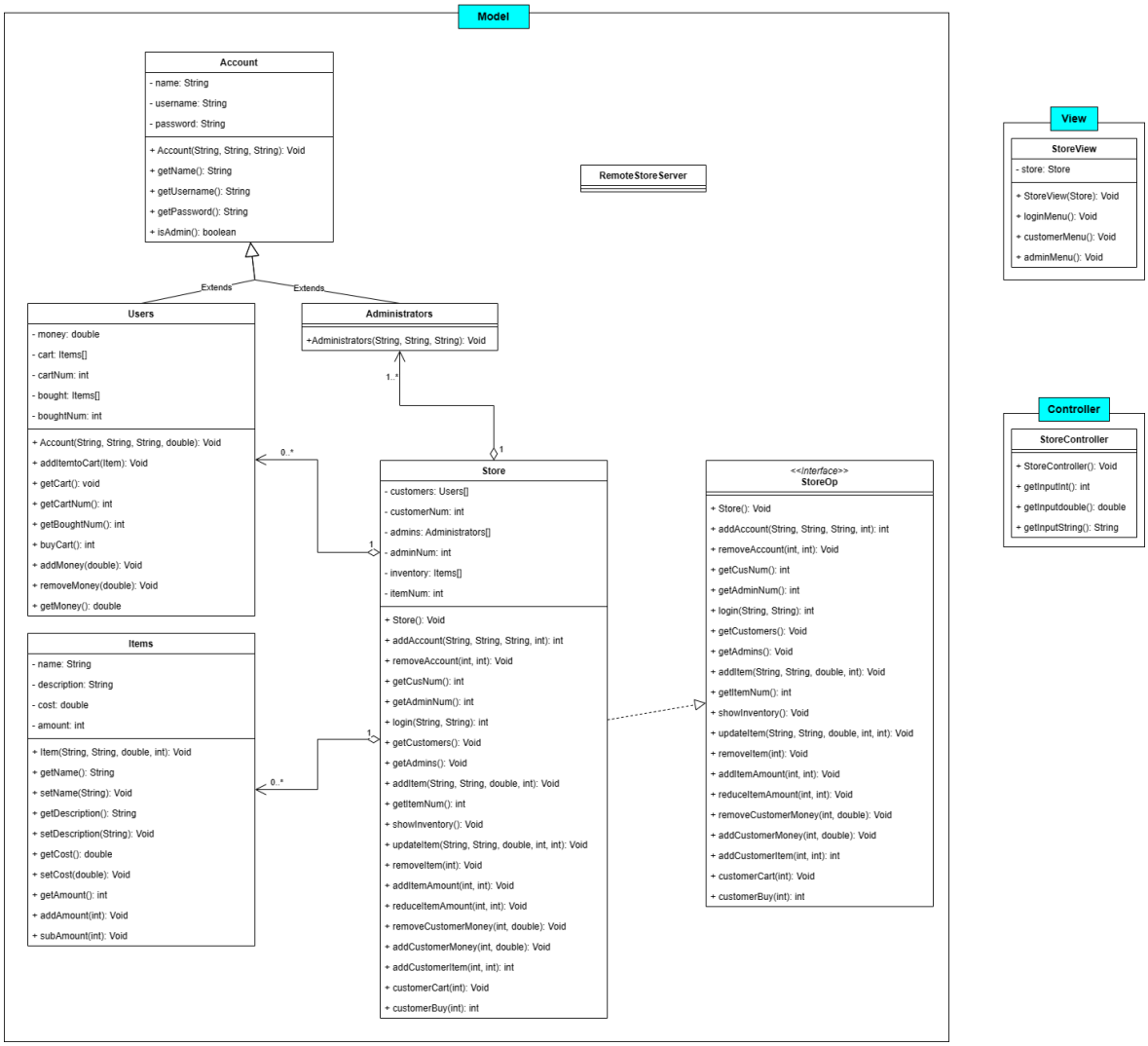
For this assignment we needed to be able to create an online store which had the capability to have multiple users, which could be customers or administrators, using it simultaneously. In order to create this online store I decided to use the MVC architecture to make the code more readable and so each of the three logical operations are independent of each other. To make the store able to allow multiple users to access it at the same time I used Java RMI to create a remote server that can have multiple clients contact it at the same time. Once I had the basic idea of how I wanted to create the store I then began to work on the domain model.

Domain Model:



The domain model is a simple and straightforward diagram roughly showing what will be in the model part of the MVC. As can be seen in the image above the model has three main parts which are the accounts, items, and the store itself. The accounts can be either customers or administrators so inheritance is used to further specify details about the two classes while keeping the universal details the same. In the Users class the details about the individual costumes are held such as the items in their cart, the amount of money they have, and all the items they have purchased. The Items class is a standalone class which is used to create products that will be in the store and they are composed of four attributes: name, description, cost, product amount. The main class in the model is the Store class because it incorporates all of the previously stated classes into one class that is used to operate the store. In the Store class an array of customers, admins, and items are kept along with three variables keeping track of their lengths.

Class Diagram:



In the class diagram of assignment 2 the classes and relationships are further defined along with showing the full MVC architecture. Every class is fully encapsulated with getter and setter methods to ensure the data is fully protected. The Store class has an interface called StoreOp that is used for Java RMI to establish a remote connection for the program. The Store class also has three aggregation relationships between the Items, Users, and Administrators classes. Both the Items and Users classes can have zero or more objects in one Store object, but the Store must have at least one Administrator object to run the online store. The Users and Administrators classes inheritance to the Account class is also further defined in this diagram. The only class without a connection in the model is the RemoteStoreServer class which is used to set up the server that contains the Store object that all the clients use when visiting the online store.

Java RMI:

In order to allow multiple clients to visit the online store at the same time I used Java RMI to create a remote Store object in a server which clients could connect to. The first step in this process was to first establish the Store object to be a remote object which was done in the StoreOp interface that the Store class references. Then once Store is able to be accessed remotely the online store's server is created in the RemoteStoreServer class. In this class a remote Store object is created on the specified virtual machine and can be accessed by any client that connects to it. To connect clients to the server a Naming.lookup method is used on the server's pathway to find the remote Store object. Once the remote object is found operations can be performed on the remote object. Multiple client virtual machines can be connected to the server using the remote object at the same time sharing and manipulating the same Store object.

Additional:

I was unfortunately unable to get the JAR file to work correctly after multiple attempts to do so. Not having the time to find a fix for this I zipped all the files up in a standard zip file. Sorry for the inconvenience.