**Proof of Concept Youtube Link**

<https://www.youtube.com/watch?v=0kVYKptMjds>

**Project Requirements** – [Checklist]

√ Register for a New Account

√ Login

√ Encryption

Customer:

√ Update User Information

√ Update Password

√ Place an Order

√ Update an Order

√ Delete an Order

√ Search for Products

Manager:

√ Manage Product (Add, Update, Remove)

**Overview**:

In our final COMP 3700 Project, we were assigned with writing an OOD program which would interact with and manage information stored in an SQL remote client/server environment. The assignment was to create an interface in which a user could interact with a remote server from a local client. It specifically contained 2, arguably 3 user roles. A “New User”, a “Customer” and a “Manager”, each with specific privileges and abilities per the COMP 3700 project requirements.

**Design**:

**[LOGIN SCREEN]**

* There are 3 total user roles, with 2 being declared and the other inherent.
* Upon launching the client (assuming the remote server is running), a user is met with a login screen allowing them to login with a Username and encrypted Password or Create a New Account.

**[NEW USER]**

* The inherent role is “New User”. These users may create an account from the local client, connecting to the server and storing a username and password in the server. *If this username is already taken, the client will notify the user that the username is not available and to try again*. Once a username and password are set, the user inherently becomes a “Customer” and can login.
* **Logging In**:  
   When a user logs in, the system will automatically grab their role and determine the screens displayed. *If a user attempts to login with invalid credentials, a pop-up is displayed notifying the user.*

**[MANAGER]**

* When a “Manager” logs in they are able to add, update, and delete products.   
  THIS WAS THE ONLY STATED REQUIREMENT FOR MANAGER PER THE RUBRIC.  
  My intention was to add additional privileges included the management of all orders and users. In users the manager would be able to manage roles (promote users to manager or employee, as well as manage lockouts from too man failed password attempts).

* **[CUSTOMER]**

When a “Customer” logs int they are given a screen that allows them to update their personal account information including password *(I forgot to show this capability in the video, but it works)*. The only restriction is that they cannot update their username – this is by design and is why the username field is disabled. Current information is displayed, however the password is encrypted. They are also given the option to search products and place or manage orders.   
  
If the customer chooses the “Products and Orders” button, they are presented with a screen that allows them to place a new order or view/update an existing order.

New Orders allows the customer to search any product in the remote server by name and its likeness. This will display the product information allowing them to enter in the product ID and quantity, as well as place the order. When the order is placed the database creates an order which includes the orderID (auto-generated), orderDate (auto-filled), and the userID (auto-filled). It simultaneously creates an orderLine with the same orderID, and includes productid and quantity.

If the customer chooses to view Existing Orders, they can see the orders they have placed and choose to delete them or update them by orderID. By deleted an orderID, both the order and orderline are simultaneously removed from the database.

* **Logging Out**:

Both the Manager and Customer have the ability to “Logout” which brings them back to the cleared Login Screen, and breaks the stored user information from the client-server. This was not required, but it felt appropriate.

**Technical**:

JAVA was the OOD language used in this project

IntelliJ was used to as the coding environment

SQLite was the database management environment

JDK 1.8 was used as the latest Development Kit

Hardware Environment was MacOS

**Acknowledgements**:

The Starter Code released by Tam was used in beginning the new version of this project.

No outside material was used for this project.

Previous material (Project\_1 and assignment\_3 were referenced).

Several office hours with Tam Nguyen assisted greatly.

***A special shoutout to Tam Nguyen (is he a Dr now?) for his help.***

**Comments**:

This project was significantly more challenging than expected. As a student who legitimately did this on his own originally, and then restarted using the TA’s code for readability and simplicity sake, I easily put more than 120 hours into this project alone. I did NOT use GitHub, Chegg, or other students code. I legitimately wanted to do this on my own and, of course with the exception of the help I received from Tam Nguyen during office hours, was successfully able to do so. The complexity of this code, especially with its desired functions using the Java language cannot be understated. Whereas Project1 and Project2 may have taken 2 hours, and assignment 3 upwards of 6 or more, this project was monstrously involved. It serves to say that I can only pray grading is lenient considering the weight of the final project is 30% and the involvement is stark. As of April 28th, many students have NOT started this project. I am unsure how they will complete it “honestly”.