Group 4 Project 2 Analysis

Question 1: The GUI Finite State Machine

S1 (Logged Out/Logging In): This is the state that the user will be in before entering the credentials to enter another role-based state (states S2-S4).

- Login requirements for other states
  - S2: ClientID is both login name and password.
  - S3: “salesclerk” is both login name and password.

S2 (Client): View the account, put in an order, check the price of a product, modify the shopping cart, place order etc.

S3 (Salesclerk): Print a list of all products with quantity on hand, add a client, add a product, load database, “become a client,” logout, and any other ops except the ones listed as manager ops.

S4 (Manager): Modify the sale price of an item, receive a shipment, “become a client,” logout, and freeze (block all future order processing)/unfreeze a client’s account. All manager operations need a password for confirmation.

S5 (Become Client): Allows a salesclerk to enact commands as if they were a specified client. Client is chosen by indicating ClientID before moving to the impersonation state.

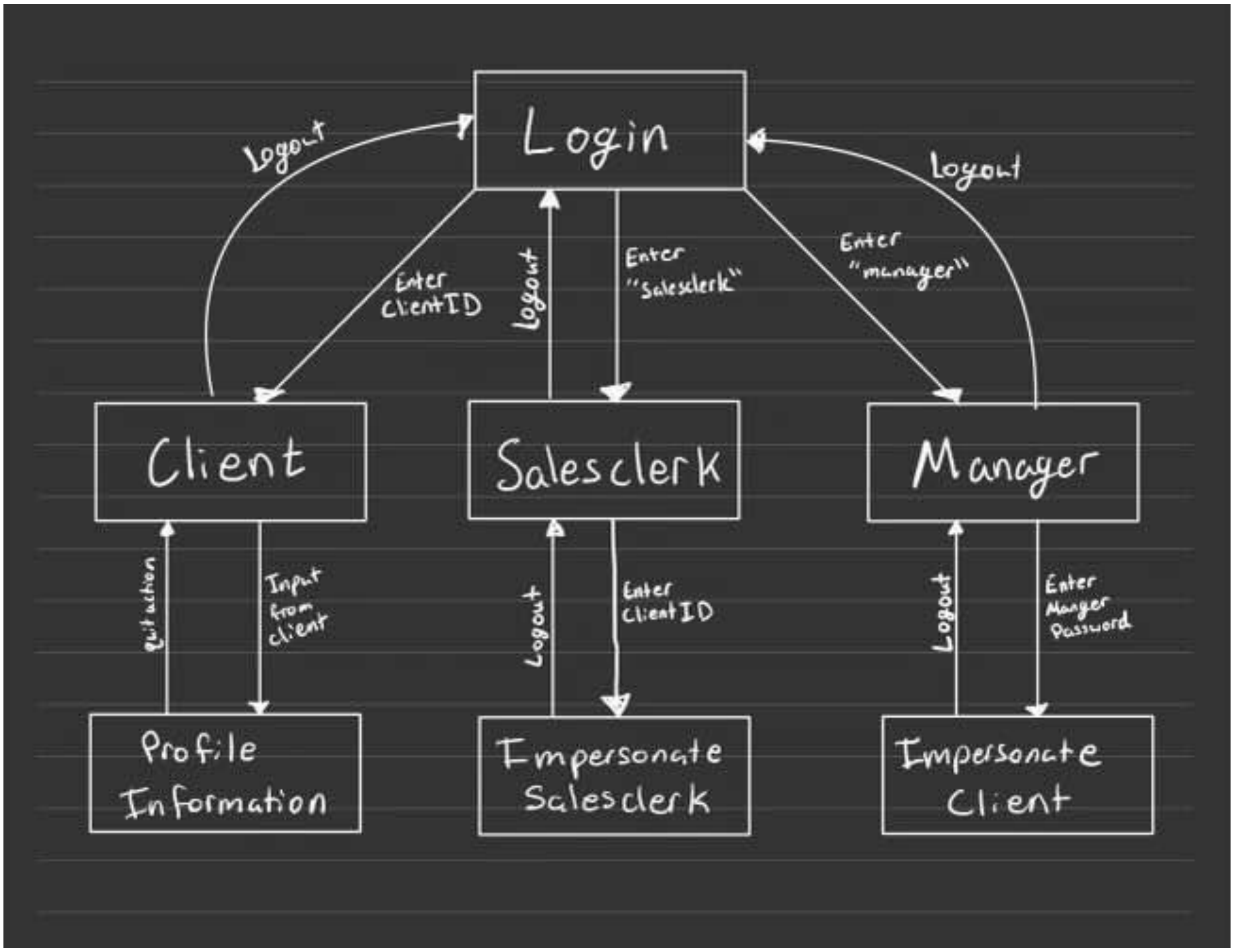
- When the logout action is taken in this state impersonation state ends and you move back to the initially logged in salesclerk state.

S6 (Become Salesclerk): State that allows a manager to enact commands as if they were a specified salesclerk. Client is chosen by indicating ClientID before moving to the impersonation state.

- When the logout action is taken in this state impersonation state ends and you move back to the initially logged in manager.

S7 (Client Profile): This state allows the user to see a menu for editing personal information or displaying documentation such as invoices sent to the client.

- Instead of a logout option this state has a quit option that will move the client back to the original option menu without leaving the current clients logged in state.



Question 2 Aspects of our system that do not fit with the definition of a "Finite State Machine":

The fact that roles can become other roles is an issue because usually, to enter or leave a state, there should be only one entrance to that state. In this case, the user can dynamically switch roles, when usually the movement between states is set in stone by the system. The issue isn't that you can enter client from salesclerk but instead that the logout action now has two separate functions depending on if you are impersonating or just logged in as the client. To account for this the system will keep track of the original salesclerk that was logged in a bit for impersonation. If the bit for impersonation is flipped, when the user logs out, the system will automatically log the user in as the salesclerk after the client has been logged out. The same will happen for the manager to salesperson interaction.

Another is how the password check is handled by a separate "security system" object. This is more procedural and does not naturally align with FSM states, which model system behavior based on user interactions and transitions. To account for this, we will have the security system confirm the password before attempting to move states. In other words, the actions will be completed then the login panel will give the command to move to the next panel. As such the transition will not require the password but rather the login confirmation is abstracted by the "security system" object as to which state it should move to next.