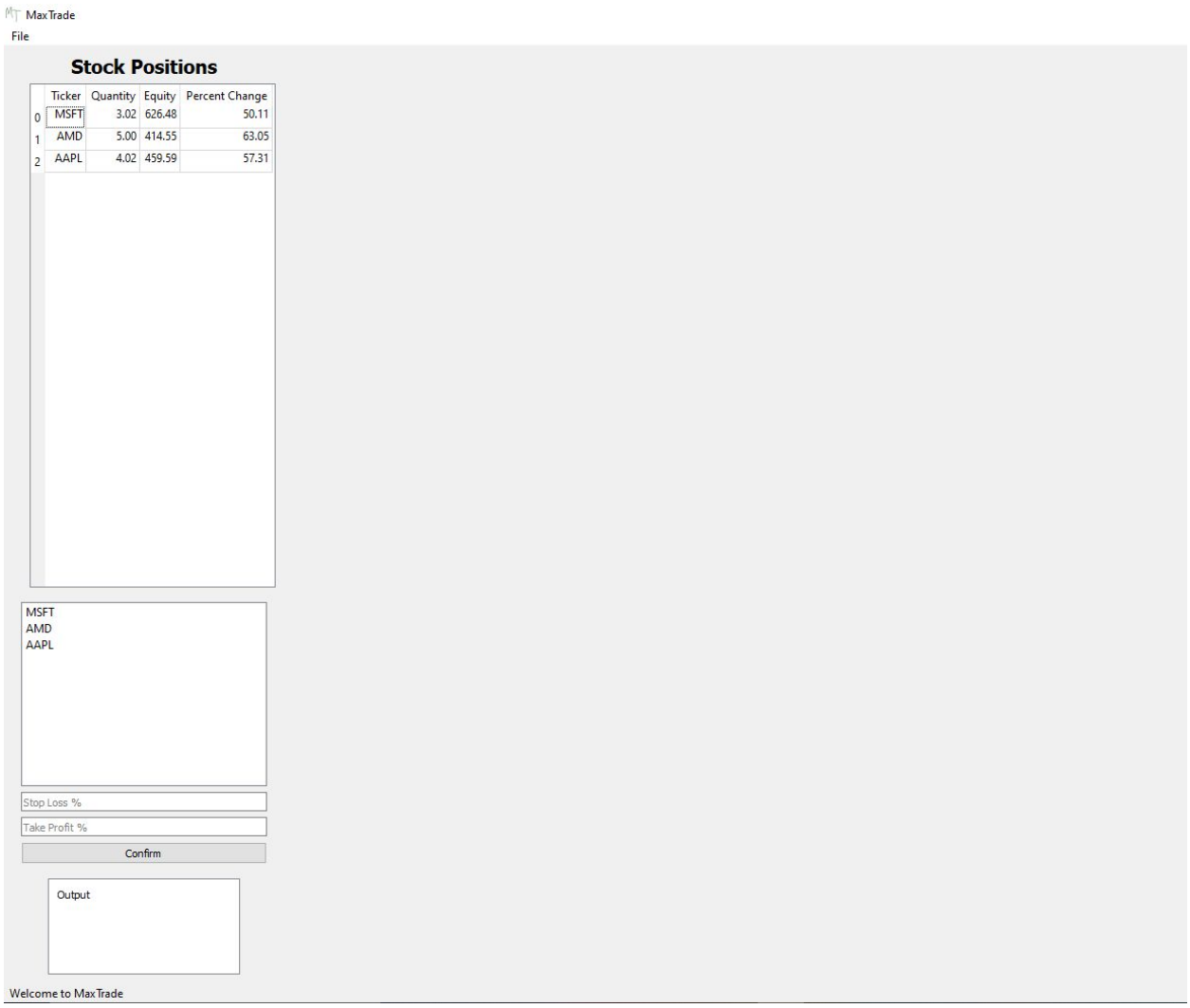
Reflection 1.1

1. What went well?
    a. Setup Robinhood API to connect to user accounts.
        i. Import robin_stocks library
    b. Allow user to securely log in
        i. Implement keyring library to store user password securely.
        ii. Connect to robin_stocks with user info
        iii. Provide way for user to enter MFA token
    c. Find all user positions and display them (not gui)
    d. Allow for manual input of stop losses and take profit % on each position w/ loop.
    e. Features Working: Display feature, secure password feature mostly working, need to add login gui box
    f. Add qt to project virtual env
    g. Display positions in table format in gui window.
        i. Use qt QWidget module
    h. Allow users to select position from table and set loss and gain, as well as for multiple positions
    i. Text box with bot output, prints set stop loss and take profit
    j. Features working: GUI window + positions in table. Confirm button working as well

    Here is a screenshot:



2. What did not go well?
    a. This sprint went pretty well, but there were definitely some things which could have gone better. First of all, I planned my user stories under the impression that the sprint would be 2 weeks long, so I gave myself too many tasks. Adding the GUI and transitioning from cmd output to a window was certainly harder than I expected, because I had to set up the library and merge it into code that I already had.
    b. Unsuccessful User Stories:
        i. Show that bot is running with a loading gif.
        ii. Adding selling in the backend based on user submitted stop loss and take profit. I already have access to the info, so I just need to set up a loop in the background checking if the positions have reached the stop loss and take profit and then submit a sell request with robin_stocks.
        iii. Adding open option positions as well as stock positions. This is just a matter of duplicating the same interface as the stock positions with the option data, which I already have setup in the backend.
        iv. Adding a login popup if the user has not already logged in.
3. How can I improve

    a. One thing I need to improve on is reading through documentation more before I try to dive in on adding a feature. Something which I often find myself doing is being excited to add a feature, and then either trying random things to make it work, or searching google. In the long run I want to make sure my code is well organized and efficient, so I should make sure to reference the documentation, especially the Qt documentation.
    b. Another way I can improve is by not focusing too much on making the program look amazing for its first version. The most important thing is to have functionality, so I need to focus on that before the layout and strict organization.
4. I discovered a few things about the computer science topics identified for this sprint. The first one is that I've already learned a lot about using python to develop applications. The most interesting part of this has been working on its object oriented side, which I did not really know anything about before. Python has much of the same functionality of C++ classes, with much more flexible design. For example, I create a class for important components on the screen, such as the table of positions. Another computer science topic which I heavily focused on is UI design. The focus of this is making an easy to use layout which displays information to the user in a useful manner. This is 1000 times harder than just printing information to the screen, and can be quite infuriating often, but the reward is a marketable product which is friendly to all users. I had to learn to use the QWidget component of Qt, as well as many other inherited subclasses.