Final Analysis Documentation

### Test Cases:

For test cases I tested

1. Registering a User and then logging in with those account details.

Entered in testUser1 with password bonkers1234. Then logged in with those credentials and received a log in successfully message.

1. Logging in as Administrator and adding a user. Looking at user table.

Logged in with Admin and password. Got admin panel. Clicked the add account button. Add account with username testUser2 with password, 56243. Click on the print table button and it displays Admin and testUser2. The passwords are hashed and the salt values are also shown.

1. Registering a User and then logging in as Administrator and deleting the user. Looking at user table.

Register a username testUser3 with password, pass. Log in with admin and password. Click the delete account button. Enter testUser3. Accept confirmation. Click on the print table button. Only admin is present.

### Analysis of algorithm:

My project determines that you either try to login or click the register button. Clicking the register button takes you to a signup page. You can either try to register a user or generate a password. If you tried to register a user it makes sure the fields aren’t blank and then looks at the username entered. It looks at the database and makes sure the username isn’t already taken. Then creates if not, hashes the password with a randomly generated salt value and adds the user info to the database. Clicking the generate password button takes you to a window where you can choose length and what type of characters are in the password and then click generate password to get a password displayed that is randomly generated from your selection .

If you tried to login it makes sure the fields aren’t blank and then looks at the username entered. It looks in the database file to find the username and if it isn’t found it spits out an error message. If it is it takes the hashed password and salt value for that user. Then it takes the password you entered and uses the salt to hash it and compares it to the stored password. If they match you login and if not you get an error.

If you login in with the credentials ‘admin’ and ‘password’ you get an admin panel instead of the program just closing. Here you can add a user like the register button does, delete a username by entering a username, modify a user by entering a username and then a new password, print the database table of user info, or exit.

Deleting a user searches the database to make sure the user exists and then asks the admin for confirmation before deleting the user’s entry in the table. Modifying accounts has the admin enter a username which searches the database to confirm existence. Then you enter a new password for the user and confirm the password. Print table displays the Username, hashed password, and salt value of users. Exit ends the program.

### Lessons learned:

Some lessons I learned from this project is that I wrote my project to work command line and then converted it to a GUI. The conversion was a lot of work and I probably should have just gone for the GUI at the beginning of project. I wasn’t sure I could do it so I started with the command line version for my, just in case, option. Also, I learned I’m susceptible to scope creep as I also want to make my project better. If I had given myself more time I could clean up my code from looking so bad.