IT145 Final Project

Randall Rowland

Southern New Hampshire University

IT 145: Foundation in Application Development

Feb 2019

**Problem Statement and Scenario**

For security-minded professionals, it is important that only the appropriate people gain access to data in a computer system. This is called authentication. Once users gain entry, it is also important that they only see data related to their role in a computer system. This is called authorization.

For the zoo scenario, I developed an authentication system that manages both authentication and authorization. Given a credentials file that contains credential information for authorized users and also given three user files, one for each role: zookeeper, veterinarian, and admin. Each role file describes the data the particular role should be authorized to access.

The program asks for a username and password. The password is than converted into a MD5 hash where both the username and hash are compared against the credentials file. If everything matches, the user’s file is displayed on screen. However, if the login fails then the user is returned back to the login screen. After three unsuccessful attempts, the program terminates. The user is also able to quit the program from the login screen if they do not wish to login.

**Overall Process**

Provide a short narrative that shows your progression from problem statement to breakdown to implementation strategies. In other words, describe the process you took to work from problem statement (your starting point) to the final product. Your process description should align to your end resulting program and include sufficient detail to show the step-by-step progress from your problem statement analysis.

I took the problem statement and broke it down into smaller problems. For example, one of the smaller problems I identified was I needed to get and store a username. I took each of these smaller problems and created issues on a GitHub repository to ensure I was tracking all of them. Within one of the issues I started a UML process but never officially formalized it into actual UML. I used that to create a flow chart of possible classes and how they will interact with each other.

From those issues, I created a milestone that cannot be complete until all the issues are completed. This included additional bugs and planning I had previously did not realize. These issues were also put on a Kanban board so between days I could remember what I was working on and what was left. The Kanban board was connected directly to the issue tracker. I could then write notes to myself about where I left off, what I think I needed to do, and other notes.

Once all the code was written and the issues checked off, I created a new branch and make a few unit testing classes. I begin testing my file reader to open and close files, followed by my implementation of the MD5 class. I then started to test the main class to see how the objects would interact with each other. When it called the Security Account Manager class, I would get a null pointer error. I opened this as a bug on my tracker.

I began to test the Security Account Manager class, I would get test failures. I set up a breakpoint and began to debug the class. I discovered that as I pulled in each string and filling the respective arrays, the passwords with spaces were shifting the arrays. I could see that the credentials were separated by a ‘\t’ so I tried to implement a delimiter using the ‘\t’. This fixed the passwords with a space in between but when the reader got to the end of the line it would combine the last string and the first string of the next line since they were separated by a ‘\r\n’ and not the ‘\t’. I then tried to set a ‘\r\n’ delimiter before the last string. The ‘\r\n’ then showed up at the beginning of the first string and the ‘\t’ was still showing up with the last string.

The next day, I remembered in zyBooks how you could read in a whole line as string and then use another scanner to split that line into individual stings separated by the ‘\t’ delimiter. This worked! I no longer was getting escape characters in the arrays.

The next issue I had written down was a parser. I thought I was gonna have to create a method to remove the quotes from around the plain text password. However, after thinking about it for a bit, the MD5 class was working, main class was working, and now the hash portion of the Security Account Manager class was working. In in main, I swapped to the hashed methods from the plain text methods. I reran the tests and was successful at logging in and closed the issue about the parser making it invalid requirement.

The last issue I had to open was displaying the user’s file. It would only show the first line and then would terminate with a NoSuchElementException. For some reason the reader did not like the blank space in between, so I caught the exception and ignored it so the rest of the file would continue to print to the screen.

This closed out all the issues, bugs, and the milestone with the program. With everything recorded in GitHub, it has made documenting my process easy for this paper.

**Pseudocode**

OBTAIN username  
OBTAIN password

WHILE LOOP (user selection not equal to exit or 3 failed attempts)

CHECK username against credential file

IF username does not exists go back to WHILE LOOP beginning

INCREMENT failed attempt by 1

IF username does exist,

convert plaintext password into MD5 hash

COMPARE MD5 hash against credential file

IF MD5 hash does not match, go back to WHILE LOOP beginning

INCREMENT failed attempt by 1

FIND text file that matches username

PRINT text file to screen

QUIT Program

END WHILE LOOP

QUIT Program

**Methods and Classes**

I don’t care for the use of pseudocode but I had written down some UML. When I was writing down the problem statement, created the following.

**Classes with fields**

UserCredential

String username

String password

PasswordConverter

no variables

SecurityAccountManager

bool allowAccess

TextReader

unknown variables

**Classes with methods**

UserCredential

final String getUsername()

final String getPassword()

void setUsername(String)

void setPassword(String)

PasswordConverter

final String convertPassword(String)

SecurityAccountManager

void setCredentialsFile(String)

void searchForUsername(String)

void comparePassword(String)

bool isAllowedAccess()

TextReader  
 unknown methods

**Differences**

When I created the first pseudocode, I had it split into these classes with the while loop calling all the methods. I recreated it following the example provided and make a few changes to match more what my program was doing. However, I have everything in the loop to include obtaining the username and password. Otherwise, it functions very closely to what the pseudocode has.

**Error Documentation**

As stated previously, the major errors I had was in dealing with the credential file and reading in the strings correctly. The delimiter cause a different issue but still related.

The second error was only having the first line of text displayed of the user’s file and it throwing an exception.

**Solution Documentation**

To be able to solve the problems, I had to use unit tests and the integrated development environment's debugger. I was able to see where things failed with the unit tests. Once the classes or methods were identified, I could then set up a breakpoint and see how that method or class was working. This was how I was able to see the array and determine my delimiter.

Really what I learned from this problem solving is unit testing. This is one of my weakness and a portion that I would like to improve upon.