Drew Waller

11/10/2018

COMP 2710

**Project 2 – Design**

**Analysis (Use Cases):**

Use Case 1: System Logins Screen

Assumptions: The user has already been added to the system. (if not, we will select the add branch staff member(Another use case, see use case 2)).

Case 1.1: The user logins in as a system administrator or branch staff member.

System interaction: The system will prompt a login menu with a user ID and password.

User interaction: The user inputs the correct login information and enters the administrator options screen or branch staff options screen.

Case 1.2: The user enters the incorrect user ID or password.

System interaction: The system prompts the user to login with user ID and password.

User interaction: The user enters the incorrect user ID or password.

System interaction: The system recognizes the mismatch in the user ID or the password and outputs an error message “Invalid ID. Please re-enter you user ID: “ or “Invalid password. Please re-enter your password: “.

Extra feature: User does not know their username of password and must reset it. (using the security questions that they create in the next use case).

Use Case 2: Adding a branch staff member(option number 2)

Assumptions: We are signed in as a system administrator.

Case 2.1 administrator add the new staff member and has them set up two security question and create a 4-digit security pin.

System interaction: The system displays the administrator option menu and the user selects “add a branch staff member”. Then the user is prompted to enter a user ID, and reads it in. Then, asks the user for a password, reads it, asks for them to please re-enter their password.

User interaction: The user inputs a user ID and then the new branch staff passwords. The following cases can occur while entering the password.

System interaction: System compares two inputs:

Case 2.1.1: The two passwords match and the user moves on to the security questions.

Case 2.1.2: The passwords do not match, and error message is displayed, and the user must re-enter the password again. Once inputs match move to the security questions.

System interaction: Now the system prompts the user to select a security question from a list, then create a secure 4-digit PIN that will be used to retrieve user ID or to reset password. The system will read in the security question answers and PIN. If the PIN is not 4-digits exactly the system will prompt the user to re-enter the PIN.

User interaction: Selects the security questions and enters the answers. After that the user will create the secure 4-digid PIN.

Use Case 3: Adding a client within the account management class.

Assumption: A branch staff member has successfully signed in and is now in the account management class and selects to add a client.

Case 3.1: User enters the clients name and they are already in the system.

System interaction: Prompts the user for the clients first and last name. Then, searches the list of current accounts for the client’s name. In this case the client already exists so the system prints “<client’s name> already has an account open”.

User interaction: Inputs the client’s first and last name.

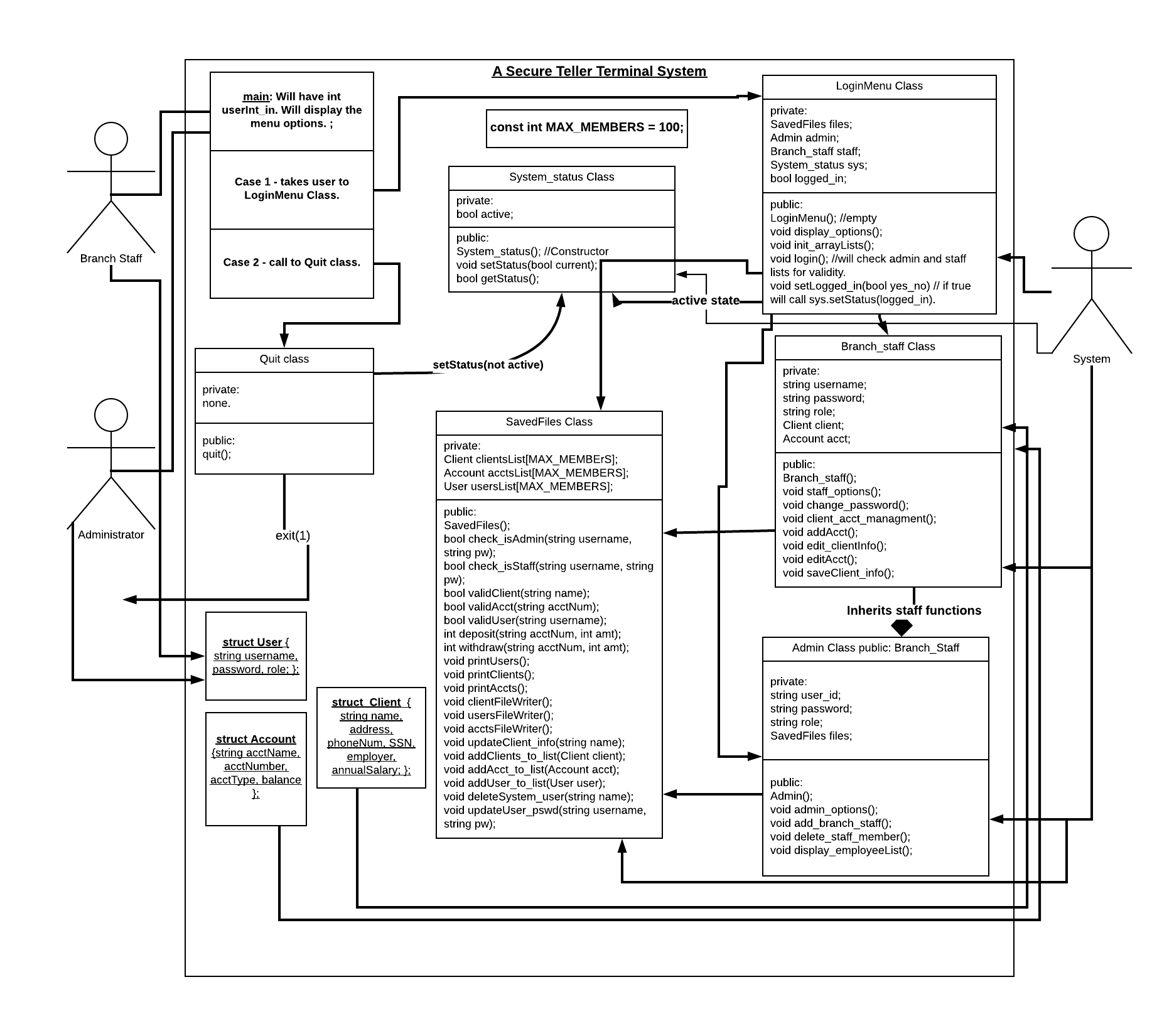
Case 3.2: New client does not already exist and the system.

System interaction: After the searching the system and not finding the client’s name the system asks for the address, social security number, employer, and annual income.

User interaction: Inputs all the information asked by the system to add a new client

**Design (UML – Diagram)**

\*\*NOTE: Did not include the additional Testing class as its purpose is trivial.



**Test Cases(Testing)**

1. The first test case should test the functionality of the deposit function. Using a shell client, the test should pass in a standard value (e.g. $100) and deposit a fund of an additional $100. After this is complete, an assertion is done to check that the balances truly reflect from the account afterwards. In addition to testing the deposit function, this will be a good place to assure that the display account balance is accurately getting the correct values from an individual instantiation of a client.
2. For the second test case, there should be a function that tests correct password input versus incorrect password input. For this test, there will be two separate modules that test both aforementioned cases, making sure that they both can successfully authenticate a password input.
3. For the third test case, there should be something to test adding a staff member to the list of valid branch\_staff. A shell list of branch staff will be established, and the function display\_employeeList will be called. After this initialization is completed, a call to add\_branch\_staff will instantiate a new branch staff member, and display\_employeeList will be called again to ensure that the addition was done properly.