Testing Document:

Test 1: testConstructorFunctionalityUsername() is meant to test the user class to see if it can properly create a new username. The test succeeds if the username does not return null after using the user.getUser() function.

Test 2: testConstructorFunctionalityPassword() tests if the user password can be created properly. If the password does not return null after using the user.getPass() function then the test is valid..

Test 3: testConstructorFunctionalityfName() tests the system's ability to correctly enter the fields into a user object. The test is successful if it uses the user.getName() function and returns a non null answer signifying it was able to label the user's first name correctly.

Test 4: testConstructorFunctionalityName() is meant to test the system's ability to correctly enter the fields into a newly created user object. The test is successful if after using the user.getName() function returns a non null answer signifying it was able to label the user's last name correctly

// ********** User Object Types.**********

Test 5: testUsernameType() test the login class examines the overall functionality of a specific username, and whether it is valid. If the sample username is successful, then it is valid. It does not, however, see if the password is valid, and will return null if the username is invalid.

Test 6: TestPasswordType() test whether the password is valid. If the sample password is successful, then it is valid. It returns null when invalid.

Test 7: TestfNameType() tests the getfName() function to make sure users use correct typing on first name. The test is successful if it returns true when it calls the FirstName variable. The first name is a string type. So, it must assign a string to the FirstName variable.

Test8: TestlNameType() tests if the system can properly set the last name to the correct typing. The test is successful if it returns true when calling the LastName variable. The Last name is a string type so it must properly assign a string to the LastName variable.

// ********** Getters & Setters **********

Test 1: TestUsernameGetter() test if the first name is valid and stored within the entire database of the server. It will see if the server stored a valid Username within it's database and will return true if and only if it does exist. It will, however, return false, if the name is invalid, and not stored within the server's database.

Test 2: TestPasswordGetter() tests the getPass() functionality. By using the sample password, if the function is able to return the password properly then the test is successful.

Test 3: TestfNameGetter() examines whether or not the first name of a user object can be correctly called. This test is determined to be successful if it returns "John" when calling the user.getfName() function. The test needs to be equal to the string "John" in order for it to deem correct.

Test 4: TestlNameGetter() tests the getlName() functionality. The test is successful if the function is able to return the sample last name when calling the user.getlName() function.

Test 5: TestUserSetter() will test the mutator function, to see if we can change an existing username within its database. It will then use a getter to see if the new username is successfully changed, and will return true if it went through. This will occur by comparing the two username strings together. It will, however, return false if it is unsuccessful.

Test 6 TestPassSetter() will test the functionality of setting the password for users. It will use the setPass() function where the password is the parameter. After setting the password, assertEquals checks if the sample password matches the result of the getPass() function. If it matches then the test is successful.

Test 7 TestfNameSetter() will test the mutator functionality, and will return true if the new modified first name. It does this by using the setter function to modify the existing name, and a getter function in order to compare the two strings together. This will return true if and only if both of the strings are the same, but will return false if they are both different (case sensitive).

Test 8 TestlNameSetter() will test the mutator functionality, and will return true if there is a new modified last name. It does this by using the setter function to modify an existing last name, and

a getter function in order to compare the two strings together. This will return true if and only if both of the strings are the same, but will return false if they are both different (case sensitive).

// ********** Testing toString() ********** //

Test 1: TesttoString() tests to see if it can properly take all the fields of an user object and properly put them into a string. The test is successful if it can properly return a string that is equal to the string “username/password/John/Smith”. If the strings do not equal each other, the test fails.