DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY OF PRETORIA

NavUp

Broadsword Users Testing

Ndung'u Brian, Ncube Nkosenhle, Letsoalo Joseph

Contents

May 2017

1 Introduction

We as the Broadsword's user team were tasked with testing team Longsword user's code implementation of the user's module. This code will be tested on a high language level to determine whether they adhered to the functional and non-functional requirements of their module.

2 Functional Requirements

- Add user to database
- Remove user from database
- Logging in specified user
- Getting specified user's details
- Updating user details
- Checking if user is activated
- Activating the user account
- Grant user admin rights
- Reseting password
- Update user password

Use Case	Description	Input	Expected Outcome
1	Inserting a new user to the database	Username, Password, Lastname, Firstname, Email, Expected Outcome	Data is inserted
2	Removing a user	Username	The user is removed
3	Logging in	Username and password	User ID should be returned
4	Getting user details	Username	The user's details
5	Updating user details	Username	The user's details should be changed
6	Checking if the user is activated	Username	It should show if the user is activated
7	Activating the user account	Username, password and activation key	It should activate the account
8	Grant specified user admin rights	Username	Admin rights field in database set to true
9	Reset password	Username and reset key	User given ability to update password.
10	Update password	Username and password	New password of user persisted to database

3 Test Cases

3.1 Add User

Test Case	Username	Password	Lastname	Firstname	Email	Expected Output
1	“Brian”	“pass”	“Ndungu”	“Brian”	“brian.ka@gmail.com”	Success
2	“Brian”	“pass”	“Ndungu”	“Brian”	“brian.ka@gmail.com”	Failed
3	“Brian”	“cos”	“Tern”	“Brian”	“brian@gmail.com”	Failed
4	“Eric”	“pass”	“Mich”	“Eric”	“eric@gmail.com”	Failed
5	“Erica”	“past”	“Michaela”	“Erica”	“eric@gmail.com”	Failed

The add user function works effectively with all the required test cases and conditions, such as checking if there exist duplicate users with the same username or password. It also provides the automated required information for the necessary fields, such as the reset key, reset date activated key. The function also provides added functionality by hashing the user’s password and persisting the hashed password to the database. There is a possibility of improvement which would be validation of the data required for each of the parameters such as checking whether the email or password is in the correct format.

Mark: 8/10

3.2 Remove User

Test Case	Username	Expected Output
1	Valid	Successful
2	Invalid	Failed

The remove user function provided effectively removes the specified user as stated if it exists.

10/10

3.3 Login User

Test Case	Username	Expected Output
1	Valid	Successful
2	Invalid	Failed

The remove user function provided effectively removes the specified user as stated if it exists.

6/10

3.4 Grant Admin Rights

Test Case	Username	Expected Output
1	Valid	Successful
2	Invalid	Failed

The function that grants admin rights did as stated but left room for possible security risks as it did not check for the necessary permissions to grant a user admin rights such as checking whether the request to grant admin rights came from someone who has admin rights.

6/10

3.5 Updating The Password

Test Case	username	password	New password	Expected Output
1	Valid username	Valid current password	New password	Password is updated
2	Valid username	invalid current password	New password	Show error message
3	Invalid username	Valid current password	New password	Show error message
4	Invalid username	Invalid password	New password	Show error message
5	Valid username	Valid password	Type nothing	Show error message

It does not allow the use to update the password if the username or the password is incorrect. This function is therefore accomodated for.

10/10

3.6 Get User Email

Test Case	username	Expected Output
1	Valid username	user email address
2	Invalid username	Show error message

This function accepts only the username to get the user email while the insert user allows multiple user to have same username this means it can return wrong email, the first one encountered email will be returned.

6/10

3.7 Update User Email

Test Case	username	Expected Output
1	Valid username	user email address
2	Invalid username	Show error message

This function does not validate if the person who is updating the email is valid user by password or something secondly it is affected by the insert since it does not restrict multiple to have same username it will eventually update the email of the first username will find in the database.

4/10

3.8 Check If User Is Activated

Test Case	username	password	Expected utput
1	Valid username	Valid current password	true/false
2	Valid username	invalid current password	Show error message
3	Invalid username	Valid current password	Show error message
4	Invalid username	Invalid password	Show error message

This function validates very well if the user is activated and denies the user to fully access the app with being activated.

10/10

3.9 Activate User Account

Test Case	username	password	ActivatedKey	Expected Output
1	Valid username	Valid current password	Valid activation key	User activated
2	Valid username	invalid current password	Valid activation key	Show error message
3	Invalid username	Valid current password	Valid activation key	Show error message
4	Invalid username	Invalid password	Valid activation key	Show error message
5	Valid username	Valid password	inValid activation key	Show error message

This function meets all the requirements that is one of the three is invalid then the program shows error message.

10/10

3.10 Overall Mark

Some of the basic functional requirements were met but there were still some that were not met and it is for this reason the appropriate mark was given

6/10

4 Non-Functional Requirements

Requirements	Mark (10)	Comments
Security	7	The subsystem is relatively secure due to their provisions for hashing the password. This results in the ability for unauthorised personnel gaining access to restricted user information being minimised. But there still exists a risk of other forms of possible unauthorised access due to the inefficient use of SQL queries, which allow the possibility of entry into the database through SQL injection.
Interface	4	Once the subsystem is up and running the interface and the provided functions are rather intuitive but the required steps to get the subsystem to start are rather cumbersome. The system relies on the use of SQL which does not give the user control over their infrastructure as it does not have cross platform support.
Quality	5	The security of the data depends on how you have encrypted your data to keep it safe and this will reduce the performance. The reliability of the data on the other hand is not of good standard due to their lack of data validation when inputting data into the databases.
Performance	5	The security of the data depends on how you have encrypted your data to keep it safe and this will reduce the performance So in this subsystem the used SQL. Total speed 3.707s just to add a user into the database. Which is significantly less than a lot of the technology's alternatives.
Safety	10	What the user has stored in the database is what the user will retrieve. That is the data integrity is not compromised.

4.1 Overall Mark

Some of the non-functional requirements were met well but at the same time others were not, based on this the average mark was taken from all the different non-functional requirements.

6.5/10