UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Faculty of Engineering, Built Environment and Information Technology

Longsword Notification

Phase 4 Testing Broadsword Notification

Frederick Ehlers	11061112
Kyle Erwin	15015302
Tshepo Malesela	14211582
Quinton Swanepoel	15245510

Stakeholders

Computer Science Department of University of Pretoria:	Vreda Pieterse
Team being tested:	Broadsword Notification

Contents

1 Create a list of service contracts tested

1.1 Pre-conditions

- User details such as email address do exist and are valid.
- Notification request is complete with details of the email for example the subject and body of the email.
- The user receiving the email exists in the system

1.1.1 Post-conditions

- At least one notification has been sent to the appropriate user/users.
- System should log who Notifications are sent to, if the Notification request is not successful, the error message should be logged as well.

The Notifications services should only fail under Three conditios that are:

- Request has failed to meet the validation criteria specified in the post conditions.
- The request information is incomplete or has an error.
- There actually is no notification for the user.

1.2 Team Broadsword Notification Serices

For this section we inspected the code of the Team Broadsword email services.

- In the code, the user details are supplied to send an email, this includes email, subject and body.
- The email is not verified that it truelly is a real email address, therefore for fake user email, the code will produce exceptions.
- The notification services of the Broadsword team does work, There are however no batch emails have been implemented.
- The system does have a method to log the email notifications, this is done by saving he notifications request object.

2 Create a list of non-functional requirements tested

3 Evaluate the test cases of the previous team for non-functional requirements using your own criteria

3.1 Modularity

We felt that the broadsword notification team developed well written and modularized code. Each class has designed for a unique purpose and did nothing outside of that purpose. The functions themselves were not over complicated However they could have broken them up to into separate tasks.
8/10

3.2 In-code Comments

The comments for their implementation were decent however there were places were a comment could have been added or expanded upon. The classes themselves each have a descrption of what the classes are supposed to do but again were not exapned upon.
6/10

3.3 Capacity

The team implemented the ability to send a single a email however they didn't provide the ability to send batch emails.
6/10

3.4 Error Handling

The error handling was implemented very well with in their python script, which is the dominant language they used. Every error thrown was caught and handled such as unsupported http requests. Their java script, however, did not provided any error handling on some object creations. This could lead to the function ending unexpectedly.
7/10

3.5 Use Cases

4 Add more tests if required