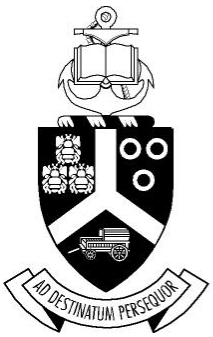

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Information Technology

COS 301 - Software Engineering

COS 301 - Mini Project

Author:	*Student number:*
Jason Richard Evans	u13032608
Abrie van Aardt	uXXXXXXXX
Anrich van Schalkwyk	uXXXXXXXX
Baruch Molefe	uXXXXXXXX
Jimmy Peleha	uXXXXXXXX
Johannes Coetzee	u10693077
Liz Joseph	u10075268
Maret Stoffberg	u11071762
Sebastian Gerber	uXXXXXXXX
Hugo Greyvenstein	uXXXXXXXX

April 22, 2015

Threads Testing Report

Buzz Space Mini Project

Version: Version 0.1 Alpha For further references see gitHub. April 22, 2015

Contents

1 Company A

1.1 Functional Requirements.

1.2 Architectural Requirements

2 Company B

2.1 Functional Requirements

SubmitPost

Test Status: *Partial Pass.*
Details: The submit post use-case was divided into seperate smaller functions. One function *create(mUser, mParent, mPostType, mHeading, mContent, mMimeType)* is used to initialize the process of creating a new thread as well as its embedded post object. This function calls *createNewThread(mUser, mParent, mLevel, mPostType, mHeading, mContent, mMimeType, mSubject)* which parses the current thread and post object to JSON strings and then persists them in a remote database.
The problem however is that when child threads are created using the *createThread(...)* function call, the child thread is never persisted to the remote datase.
The functional requirements also state clearly that a thread (with its related post) should have been allocated a certain space, yet this functionality is never provided.

MarkPostAsRead

Test Status: *Complete Failure.*
Details: The teams approach to the reading events are not that well planned and clearly thought through. It simply sets a flag for the thread and persists that flag to the database. This however will then not be user based, as everyone who then accesses that specific thread from the database will have "read" it.
This function is hence not working as required and thus fails completely. It is however a nice-to-have function and does not break or affect any other part of the thread module.

2.2 Architectural Requirements

3 Conclusion