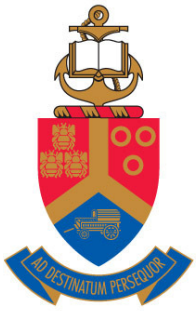


UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science

COS 301

Mini Project Assignment 1

Software Requirements Specification and Technology Neutral Process Design

---

Team Bravo

Student:	*Student number:*
Daniel King	u13307607
Azhar Mohungoo	u12239799
Andreas du Preez	u12207871
Banele Nxumalo	u12201911
Frederic Ehlers	u11061112
Diana Obo	u13134885
Bilal Muhammad	u13080335

Contents

Bravo github repository link

For further references, please click on this link.

1 Introduction

This document aims to specify the functional and non-functional requirements of a document archiving system, as specified by Ms Vreda Pieterse of the Computer Science Department.

It will serve as a means of communication between the client and developers as well as providing an elaboration and a clear discription of it's implementation specifications.

2 Vision

We intend to create a system that will allow authors and their co-authors to work on their research papers in an environment that reassures collaborative work, which in turn diminishes the time spent on papers with multiple authors. The following are what we plan to achieve:

- Keep track of research papers.
- View meta-data of research papers.
- Allow multiple authors to collaborate on the same research paper.
- Different levels of authority, i.e. Admin, (Co)Author, User.
- View and edit the details, of a text based profile, of different researchers.
- Implemention as a website and an android application.

3 Background

We live in a world where time is valuable. We would like to do as much as we possibly can in the shortest amount of time. And if that's not possible, we work in teams to ensure that we achieve that goal.

Reseacher papers tend to be fairly lengthy, and if completed by only one author, it could be quite a tedious process. Hence we propose a system which would make the storage and collaboration of research articles and papers effortless by producing an archive system.

4 Architecture Requirements

4.1 Access channel requirements

The different access channels for the system will be as follows:

4.1.1 Web Application

The system can be used via a web application that uses RESTful web services and bootstrap technology. The system will be fully supported on the following web browsers:

- Chrome 7.0.517 and up
- Firefox 3.6 and up
- Safari 4 and up
- Edge

Bootstrap will be used so that the web application will also be accessible by mobile web browsers that support HTML5 and JavaScript technology.

4.1.2 Mobile Application

The system will also be accessible via a mobile application and will be operational on the following mobile operating systems:

- Android 4.0 (Ice Cream Sandwich) and up.

4.2 Quality requirements

- Security - The system shall identify all of its client applications before allowing them access to those enities. Possible measurement methods: Success rate in authentication, percentage of successful attacks, encryption level and probability/time/resources needed to attack the system.
- Usability - The degree of ease of use and training needed for end users. Possible measurement methods: Time it took a user to find a report (search functionality), percentage of deadlines met (notification functionality) and time it took a user to perform certain tasks.
- Auditability - The degree to which transactions can be traced. Possible measurement methods: The number and precision of logs generated in a period of time (metadata accessed/deleted/added).

- Portability - Measure ability of the system to run under different computing environments. Possible measurement methods: Number of targeted software environments (ie different browsers and operating systems) and proportion of platform specific functionality.
- Maintainability - Measures ability to make changes quickly and cost effectively. Possible measurement methods: Degree of complexity to make changes to the format of the metadata and mean time to add or change certain functionalities (ie new report types and new types of users).
- Availability - Percentage of time that the system is up and running correctly. Possible measurement methods: Length of time between failures and length of time needed to resume operation after a failure.
- Performance - Possible measurement methods: How well the system perform under high workload and number of events processed/denied in some interval of time.

4.3 Integration requirements

4.3.1 Integration channels to be used

- System will be integrated on two platforms. A website and an android application.

4.3.2 Protocols to be integrated

- Protocols needed for the website: HTTPS, Data Query Protocol
- Protocols needed for the application: SIP

4.3.3 API specification (UML form) and technology specific API specification

- UML diagram here

4.3.4 Integration quality requirements

- Reliability - The service must not crash, it must run effectively and efficiently allowing users to make use of the service at all times.
- Auditability - Have logs record all activity that occurs thus all actions and occurrences can be traced.
- Security - Giving the users reassurance that the system will protect their details and content. Allow secure and safe communication between user and the system.
- Maintainability - Provide easy, effective and efficient maintenance to the system allowing minimal or no downtime of the system.

- Usability - Allow users to use the system with ease and minimal need for tutorials. Ensure that the system is self-explanatory, efficient and effective.
- Portability - To provide compatibility with as many systems as possible without integration failure or function failure.
- Availability - Minimise downtime of the system. Quick recoveries from crashes, preventive measures to minimise chances of system crash.
- Performance - The rate of work done when the system has either a high or low workload and minimise amount of denied events.
- Seamless transitions between platforms - Implementation between the integrated platforms must not vary greatly to avoid confusion and allow ease of use.

4.4 Architecture constraints

- The platform must exist in 2 mediums, namely web and android application.
- It is an internal network to be used by the Computer Science department, so it need not to source any information from the web.
- Any form of database can be used, but the team has decided that MySQL would be the best option from the current skillset.
- PHP and JavaScript will be used for the web and Android Studio for the android application.
- The system should cater for and including 100 users as a maximum.
- There should be a log that records all activity on the system

5 Functional requirements and application design

5.1 Use case prioritization

This section specifies the level of importance of different use cases, prioritized in terms of the following 3 catogories: Critical, Important and Nice-To-Have. Each of which has a lesser importance than the previous one.

5.1.1 Citical:

- Server connection
- Database connection
- User gateaway
- Research paper management
- User account management
- Log services

5.1.2 Important:

- Notification Services
- Search Services
- Archival Services

5.1.3 Nice-To-Have:

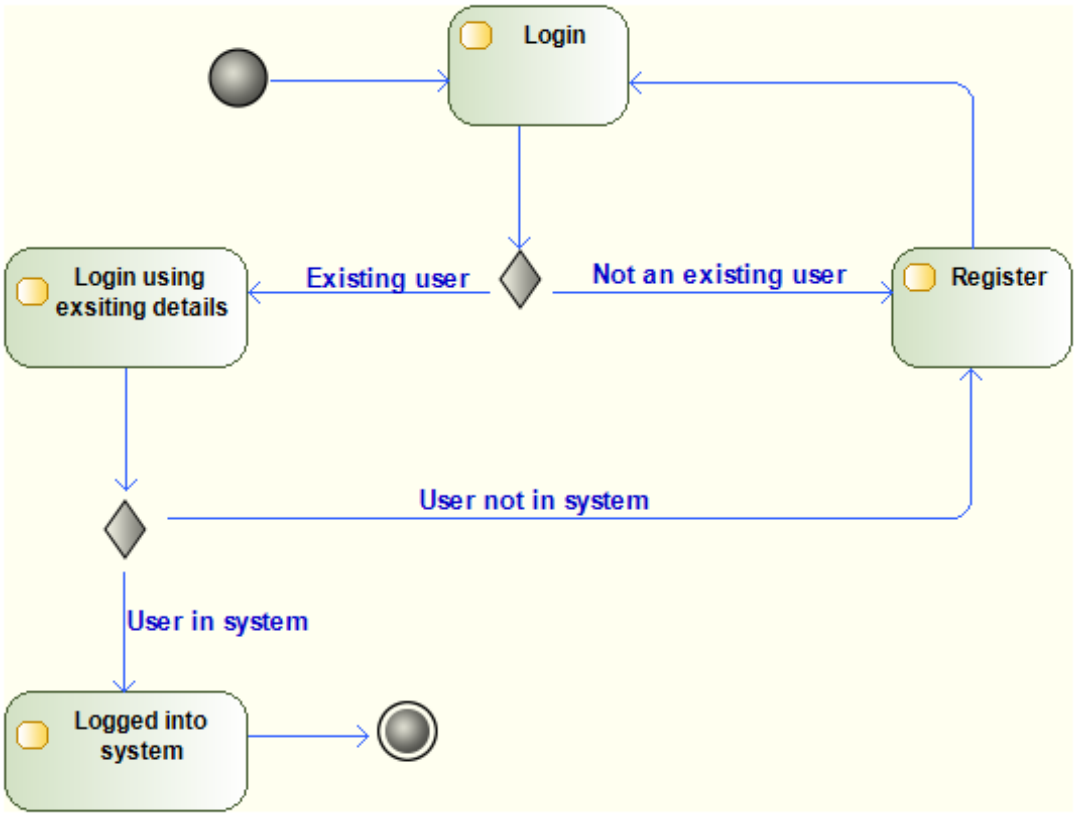
- ???

5.2 Use case/Services contracts

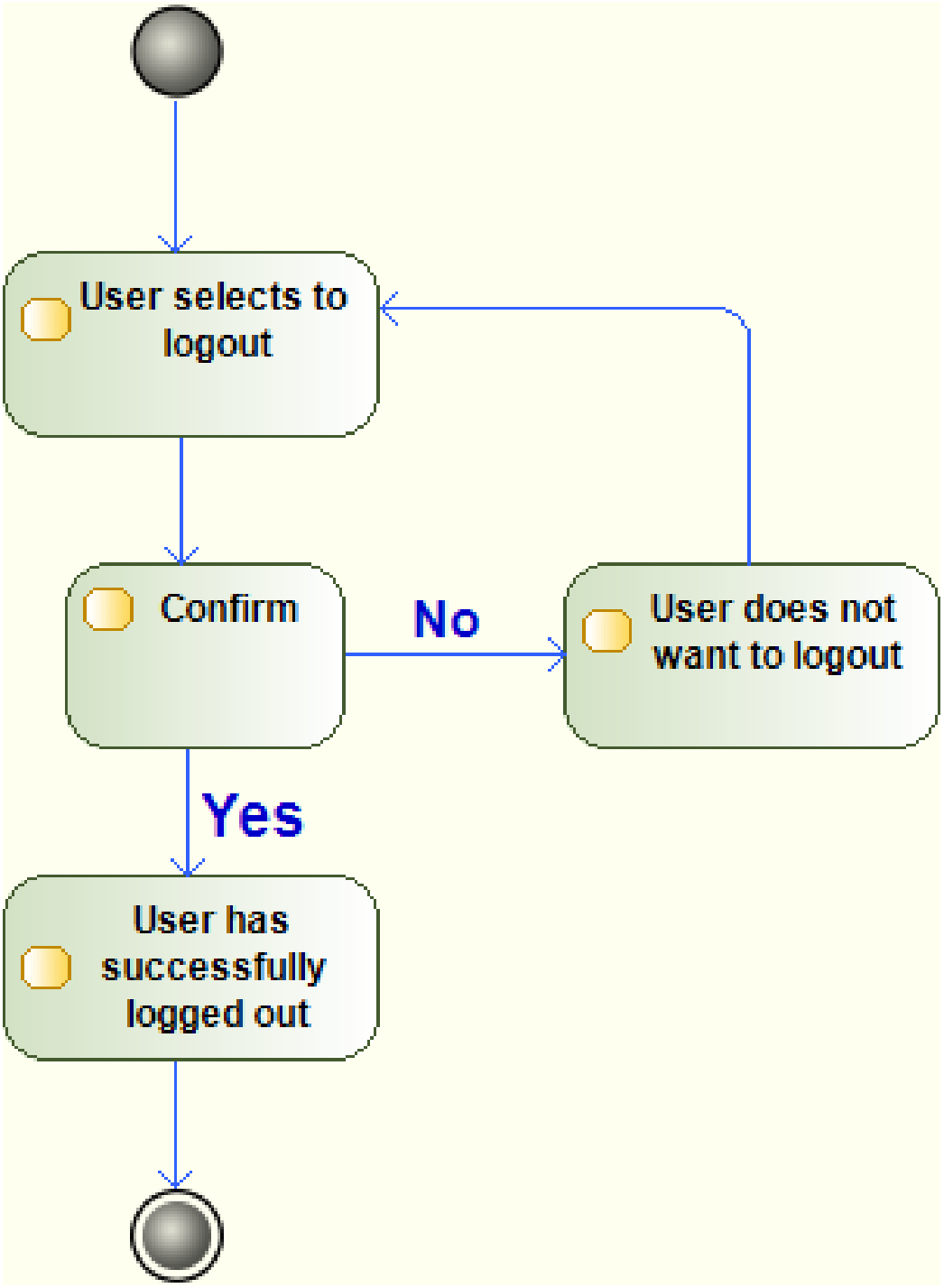
5.3 Required functionality

5.4 Process specifications

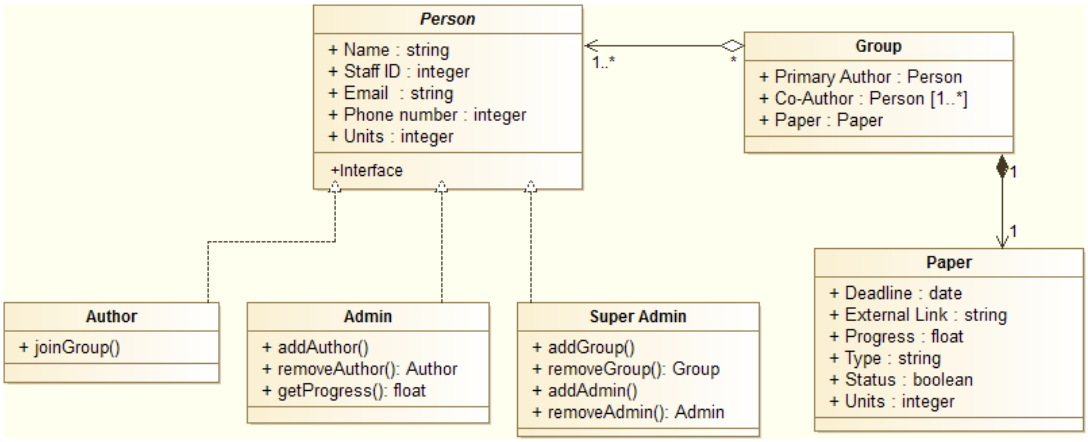
LOGIN



Logout


User selects to logout
Confirm
No
User does not want to logout
Yes
User has successfully logged out

5.5 Domain Model



6 Open Issues

References

[1] Mrs Vreda Pieterse. Cos 301 mini project: Client specification. COS 301 lecture notes and template documents via the Computer Science website, February 2016. Specifications also provided during COS 301 lectures at the University Of Pretoria.