Buzz Discussion Board

Version: 1.0

Client: Vreda Pieterse

Author: Group 5a

ARCHITECTURE REQUIREMENTS DOCUMENT

**Contents**

**1. Document Description**

**2. System Context**

**2.1 Core Purpose**

**2.2 Who requires the system?**

**2.3 Stakeholders**

**3. Architecture Scope**

**4. Quality Requirements**

**4.1 Scalabilty**

**4.2 Performance Requirements**

**4.3 Maintability**

**4.4 Reliability and Audibility**

**4.5 Testability**

**4.6 Usability**

**4.7 Integrebility**

**5. Access and integration requirements**

**6. Architecture Constraints**

**1. Document Description**

**2. System Context**

**3. Architecture Scope**

1. A web access channel

2. Providing and hosting a usable system environment for the services and business logic

3. Persisting and providing access to domain objects

4. Providing an infrastructure for specifying and executing reports

5. Integrating with the LDAP repository

6. Provide an infrastructure for notifying users

**4. Quality Requirements**

**5. Access and integration requirements**

This section specifies the different channels through which the system can be accessed by humans and other systems. It also specifies the integration channels which must be supported by the system. Last but not least, the channels through which the architectural components can integrate.

**5.1 Access Channels**

The system will be accessible to humans through the following channels:

* A web browser that provides an easily usable and rich interface. The system must be accessible from any widely used web browser including all the recent versions of Google Chrome, Opera Mini, Mozilla Firefox, Safari, and Internet Explorer.
* Restful or SOAP based web services

**5.2 Integration Channels**

The system should be able to access:

* Computer Data Source Adapter (source module, student & lecturer information from the Computer Science LDAP repository)
* Computer Science MySQL database to access course or module information
* Gmail API for sending email notifications when posted specified threads, status changed, moderation failure, appraisal received.
* Exporting of reports as csv file, pdf file, excel file, or word document
* Alter record sets in a Buzz space by uploading relevant information that is stored in a csv file

**CS LDAP**

**CSMySQL**

The following table is a mock table for the database structure of the Computer Science department module/course table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Key** | **Default** | **Null** |
| id | int | PrimaryKey | NULL | NO |
| code | varchar(20) |  | NULL | YES |
| name | varchar(255) |  | NULL | NO |
| lecturer | varchar(255) |  | 0 | NO |
| semester | int |  | 0 | NO |
| description | text |  | NULL | YES |
| year | int |  | NULL | YES |
| discussion\_board | tinyint |  | NULL | YES |
| Tutors\_allowed | tinyint |  | NULL | YES |

**Gmail API**

The Gmail API gives you flexible, RESTful access with a JSON payload to the user’s inbox, with a natural interface to threads, messages, labels, drafts, and history from any widely used updated version of the language of your choice. Gmail API allows you to:

* read messages from Gmail
* Send email messages
* Modify the labels applied to messages and threads
* Search for specific messages and threads

It uses OAuth 2.0 protocol to handle authentication and authorization. Gmail API also supports the standard IMAP and SMTP protocol for non-Gmail clients. IMAP and SMTP use the standard SASL( Simple Authentication and Security Layer) via the native IMAP authenticate and SMTO Auth commands, to authenticate users.

Google API including the Gmail API provide a variety of libraries for modern programming languages such as java and PHP for using and connecting to their API’s.

**6. Architecture Constraints**