UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Kum-Ladi

Buzz Discussion Forum

Requirements and Design Specifications

Compiled By

Nkosinathi Mothoa - u12077420
Nkosenhle Ncube - u13247914
Nathan Ngobale - u15110045
Kamogelo Tsipa - u13010931
Melvin Zitha - u12138747

Contents

1 Vision and Scope **2**

1.1 Project Vision 2

1.2 Architecture Design of Buzz System 2

1.3 Project Scope 3

1.4 Design Requirement 4

2 Functional Requirements **4**

3 Modules **6**

3.1 Users 6

3.1.1 Service contracts 6

3.1.2 Technologies 6

3.2 Notifications 6

3.2.1 Service Contracts 6

3.2.2 Requirements 7

3.2.3 Sequence Diagram 7

3.2.4 Technologies 8

3.3 Messages 9

3.3.1 Requirements 9

3.3.2 Use Case: Post 9

3.3.3 Use Case: CRUD 9

3.4 Status 10

1 Vision and Scope

1.1 Project Vision

Buzz is aimed at finding ways to enhance teaching and improve the learning of students through the use of online discussion. Through the system, there will be formations of collaborative communities within student groups, which is essential to enhance education. The forum will allow students to express their views and explore what they are learning, this will result in creating a collaborative community in which students can excel.

The Buzz project also aims to create an online space where students, teaching assistants and lecturers can engage in activities related to learning the content of our module while applying game concepts to motivate students to increase the quality of their participation and consequently experience deeper learning of the course.

1.2 Architecture Design of Buzz System

At the highest level of granularity the Buzz system is based on Service Oriented Architecture(SOA). Second level of granularity can be visualized as to be based on model-view controller.

On a high-level view, the system must be an Online Transcation Processing system. This system will be most beneficial because it will allow the users to receive near instant responses to their requests. It can accommodate multiple concurrent users at the same time. The system will be able to handle all kinds of processes (mainly CRUD operations).

Architecture Design

- Micro services
- Model-view-control architecture patterns
- Layered architecture patterns

Quality Requirements

The quality requirements of the Buzz system will be, but not limited to:

- Security
- Performance
- Accessibility
- Integrability
- Maintainability
- Reliability
- Availability

1.3 Project Scope

.The core of the system is an online discussion board which applies game concepts. A basic Use case diagram of the system is show below.

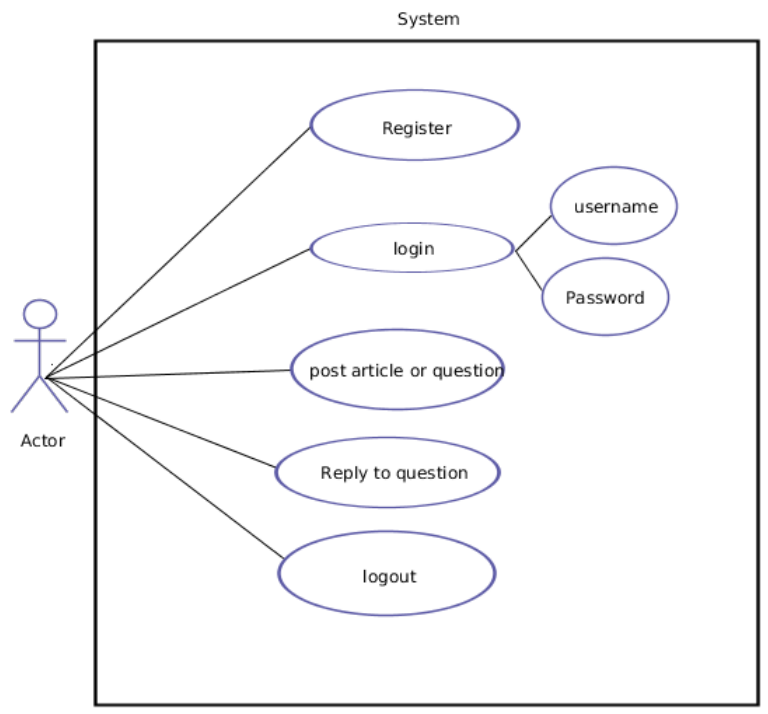

Figure 1: Use case Diagram of the Buzz system.

1.4 Design Requirement

2 Functional Requirements

- Users must be able to create,read,update and delete (CRUD) posts. certain users will be granted power to CRUD other user's posts in a highly controlled fashion.
- Keep track of who has read what and highlight unread messages for each user.
- Restrict the length of messages and the type of content allowed in messages based on the level where it is posted as well as on the status of the user posting the message.

- Restrict users to post on specified levels based on their status of the user posting the message.
- Allow staff to manage content i.e summaries,close or hide threads and move things around.
- Provide functionality to support semi-automatic creation of thread summaries.
- Create automated template based messages to individual users or specified groups.
- Automatically change the status of a user based on participation.
- Integrate seamlessly with any host site.
- Provide functions such as searching and filtering.
- Provide functionality to evaluate posts and vote for posts.
- Use evaluation to create statistical information such as average mark of each student within a given time range. Visual reporting of a participants evaluation in relation to the average of the evaluation of all the users of a certain groups of users is required for the gratification concept.
- Enhancement of the post editor for example text formatting and automatic pretty-printing of code in posts.
- Provide functions to apply social tagging. Allow users to view content based on personal structure according to their own tags or according to the administration's structure and share their tags.
- Apply self-organization based on social tagging and allow the user to view according to the base structure, owns structure or public structure.
- Detect if a post is plagiarized.
- Detect violation of etiquette rules.

3 Modules

3.1 Users

The user management module is responsible for maintaining information about registered users of the system. This includes different levels of authority and restrictions for each user. The Adminitrator can manage and control the content of posts whilst the tutors/TA's can manage the content and the violation of netiquette rules. Users who have logged in my requests services to persist content from various modules based on their needs.

3.1.1 Service contracts

3.1.2 Technologies

To address the need for a loosely coupled and high cohension system, a micro service architecture will be used.

- REST Proxy, which is an open source HTTP-based proxy for Kafka cluster,

3.2 Notifications

This module will be responsible for the delivering and receiving of notifications of the system. All notification messages will need to pass through this module in order for the necessary operations to take place.

3.2.1 Service Contracts

The notification requests will need to be of a specific type and properly validated before being processed. The request will fail if does not meet its specified prerequisites. In order for a notification request to be passed on to the server, the user needs to already exist on the database. if this requirement is not met, a notification request will fail.

- Notify user after posting

- Notify user of a posting
- Remove posted notification
- Read posted notification

3.2.2 Requirements

Functional

- Showing appropriate messages according to the user
- User friendly and accurate

Non-functional

- User friendly and accurate
- The notification should be able to direct the user to the appropriate thread or posted
- Allow for an expansion of a post or thread

3.2.3 Sequence Diagram

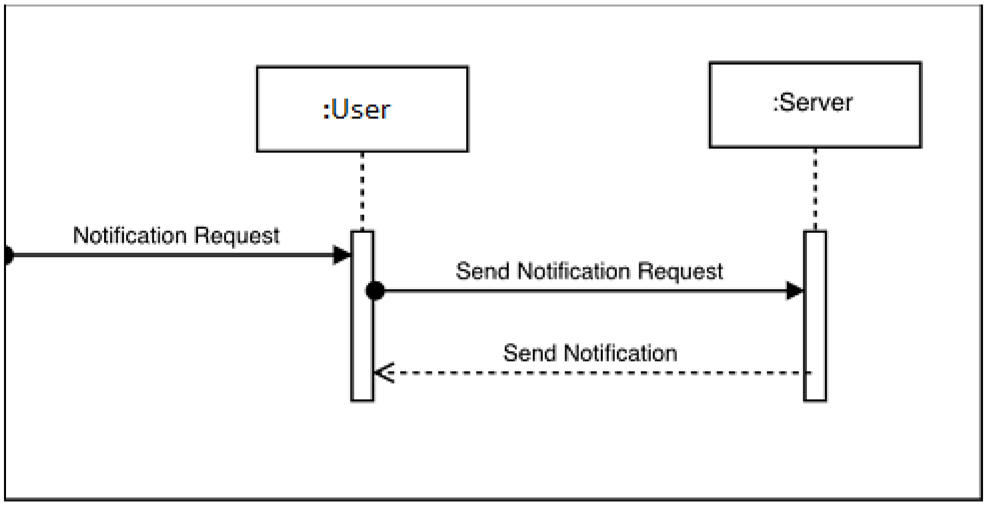

Figure 2: Sequence diagram of the notifications module

The sequence diagram (Figure 2) shows the manner in which all interactions between the user's preferred device and the server will occur. The user's device will request notifications from the server and the server will provide the notifications, if any.

3.2.4 Technologies

- Apache Kafka will be used for data streaming of the notifications. This will allow the user's page to be updated as the information changes in the background, without the user having to refresh their page to check if any changes have been made.

- The linking of the notification with the thread or post it is associated with will be done using JavaScript.This will allow for dynamic posting of notifications, without the adminstrator having to manually create every notification.

- Google Email is an API with a capacity to service a number of email within a limit. The API will be ideal to provide messaging capacity for the system without having to implement email servers internally.

3.3 Messages

This module will model the functionality involved with the posting of messages/comments/feedback on the forum.

3.3.1 Requirements

Functional

- Handle the posting and accessing of forum content asd well as structuring forum threads in an orderly manner.
- User friendly and accurate

Non-functional

- User friendly and accurate
- The notification should be able to direct the user to the appropriate thread or posted
- Allow for an expansion of a p

3.3.2 Use Case: Post

- Each user will be able to make posts about the current topic they are in. The posts will be seen on the Internet forum message board, which enables them to participate in the discussion within the current topic / thread. The effectiveness of this is that every user is able to engage on their own free will and unlike group chats on whats app people would be able to give url links, images, code snippet suggestions, and also other valuable information that may help with their understanding and development in the current thread.

3.3.3 Use Case: CRUD

- All users (admin and non-admin) will be able to create their own threads and posts but non-admin users will be able to delete their posts based on certain restrictions , example if they had made a post or a thread that has not been given any comment or follow up post, the create will be able to delete it incase they feel it will useless. Non-admin users will be restricted as to the things they, will not be able to delete other peoples posts. Administrators will be able to delete posts also viewing statistics on individual users.

3.4 Status