UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Buzz Discussion Forum

Requirements and Design Specifications

Compiled By

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

Contents

1 Vision and Scope

1.1 Project Vision

Buzz is aimed to provide an observable communication space for University modules where students can raise questions, share knowledge add comments and mark up each other's contributions. It should also enable students to follow threads of conversation and be notified of any events occurring in Buzz.

The proposed system is a discussion board which is to be integrated into the computer science department's website. It should enable lecturers to set up a discussion board for a module which provides an observable communication infrastructure for a course that can be accessed by the students and teaching staff for that module.

1.2 Project Scope

The core of the system is an online discussion board with functionality for tagging, appraisals and status building, notification and reporting. A deployment diagram of the overall system is show below.

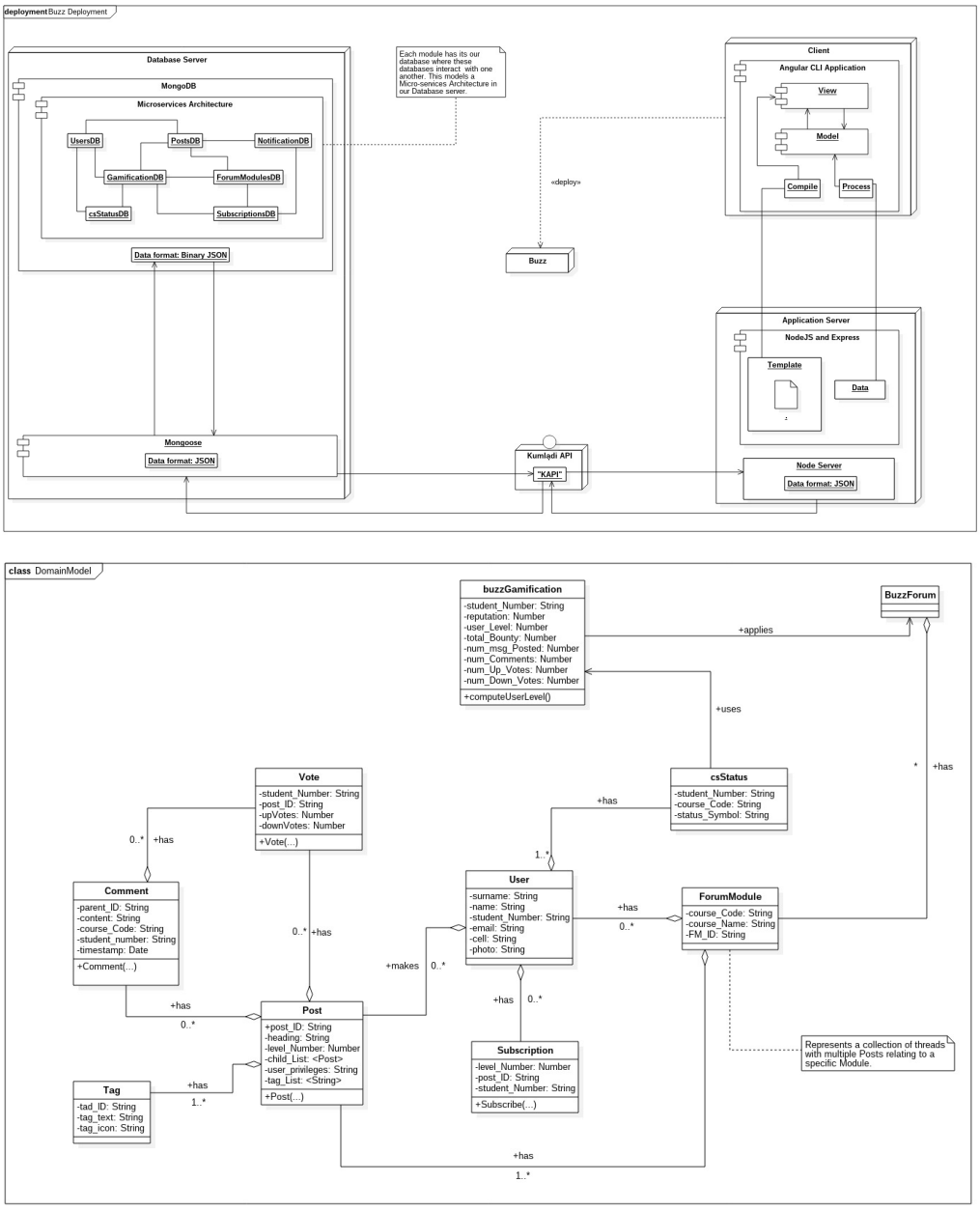

Figure 1: Deployment diagram and Buzz domain model.

1.3 Architecture Design of Buzz System

At the highest level of granularity the Buzz system is based on Service Ori-

ented Architecture(SOA). Second level of granularity can be visualized as to be based on model-view controller (MVC), which further transcends to micro services in relation to SOA.

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).

1.3.1 Architecture Design

- Service oriented architecture patterns (Micro services) on the database server.
- Model-view-control incorporated with the Blackboard pattern, is applied across the application server and client.
- Publish and Subscribe pattern will be applied for users to receive notifications based on the content or topic they subscribe toe

1.3.2 Non Functional Requirements

The non-functional requirements of the Buzz system will be, but not limited to:

- Security: Users will need to authenticate themselves upon login in order for them to interact in any forum activity. user passwords are not stored in our databases, but a combination of a user name and password is encrypted using RSA encryption before being sent across the serve.
- Coupling: The design of all database schema is done in effort to attain low coupling. The architecture applied in our database-server compliments the independence but unity of both our databases and system modules.
- Documentation: The development process includes the proper documentation of each module and/or DB-schema as well as a set of coding standards that must be adhered to by all contributers to the development of the system.

- Integrability: The MEAN Stack is being used to develop the Buzz system and integration follows a systematic approach across all tiers of the system.

- Maintainability: The main reason for aiming to produce a system with low coupling among its modules is to make the maintaining of this system as lucid as it can possibly be. It should not be a challenge for a person seeing the implementation for the first time to be able to grasp what a said module does and be able to add and/or alter the functionality successfully without breaking anything. This can be accomplished through high cohesion of our modules/schema and proper documentation.

2 Application requirements and design

- 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.

2.1 Users Module

The user management module is responsible for maintaining information about registered users of the system.

2.1.1 Scope

This includes different levels of authority and restrictions for each user. The depending on your csStatus and level of authority, can manage and control the content of posts and the violation of netiquette rules. Users who have logged in, my requests services to persist content from various modules based on their needs.

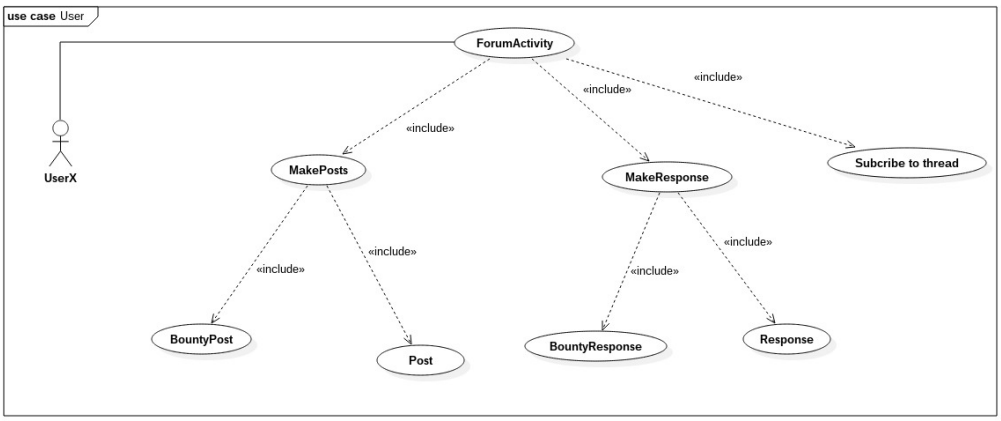

Figure 2: Scope of the Buzz-Users Module.

2.1.2 Domain model

The domain model for the user module is represented in the figure below

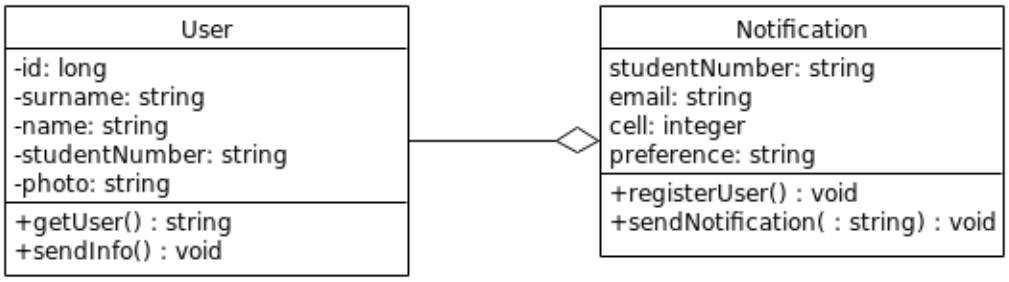

Figure 3: User domain model.

2.2 CSStatus Module

CSStatus is the module which provides the functionality to assess a number of measures around individual's contributions, to use these to calculate a status for a profile (i.e. for a user and a specific module), and to restrict access to system functionality based on user status and user role.

2.2.1 Scope

This will be the module in which the user's status will be maintained. With every change made, the user's status will be immediately assessed to see if it needs to changed. This module will be a volatile one. When a user logs in, the information from the LDAP services will replace the information that the user currently has stored.

2.2.2 Use cases

2.2.3 Create CS Status

This is a simple query service that will allow a new CS-Status to be created for a user. Once the status is created then it is returned to the user.

2.2.4 Get Status Symbol

This is a query service that will return the user status, e.g. a black belt or white belt.

2.2.5 Domain model

The figure of the CS-Status domain model is in figure 4.

2.3 Notifications Module

This module will be responsible for the handling of User notifications.

2.3.1 Scope

Users will have the option of subscribing to Forum content and receiving notifications on this specific content via email.

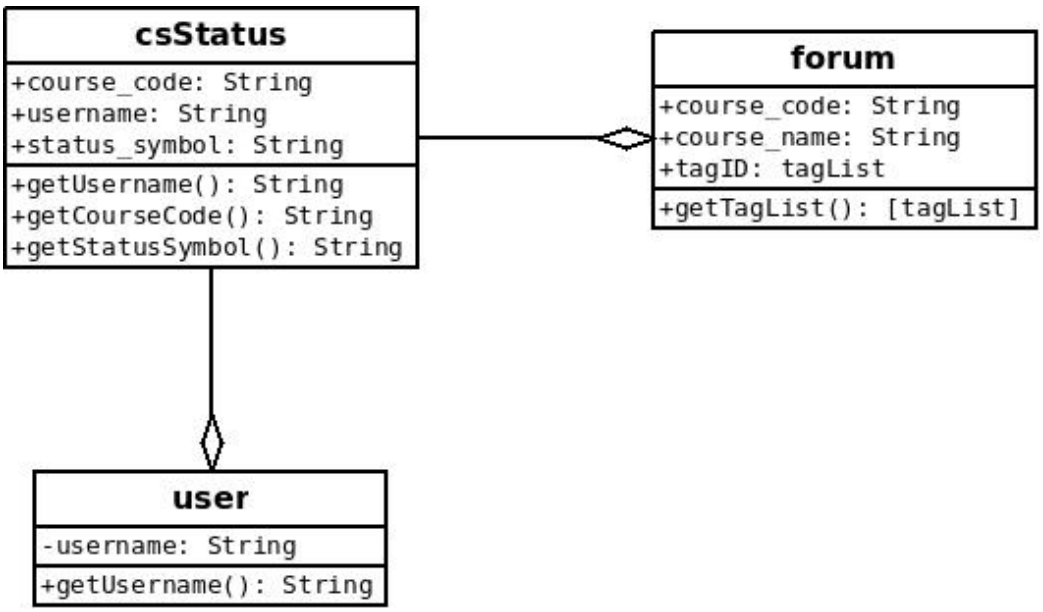

Figure 4: Domain model representing the message model

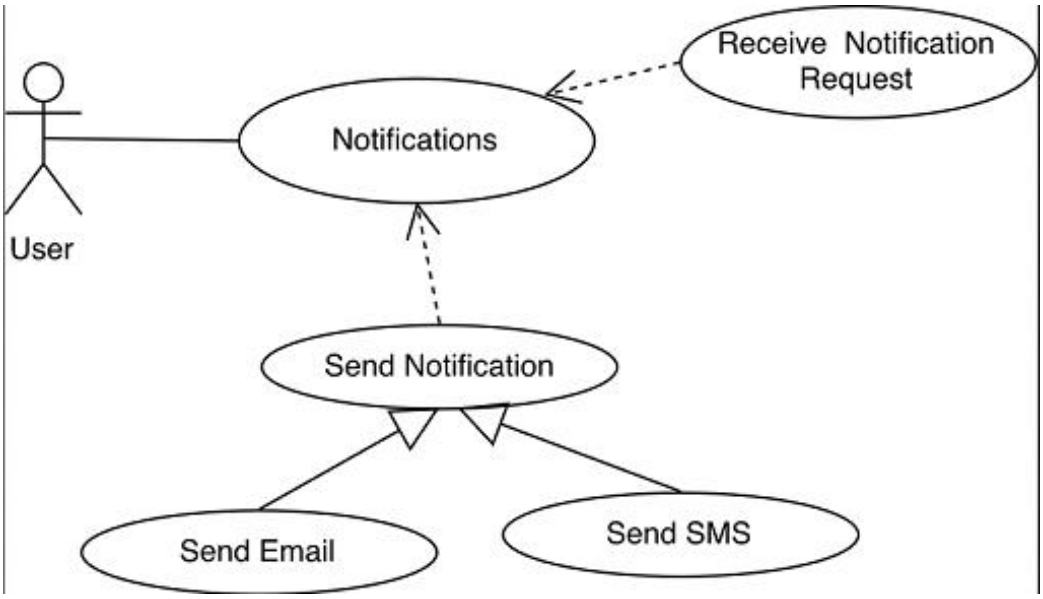

Figure 5: Scope of the Buzz-Notification Module.

2.3.2 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

2.3.3 Technologies

- 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 administrator 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.

2.3.4 Domain Model

The domain model for the notifications module is represented in figure below

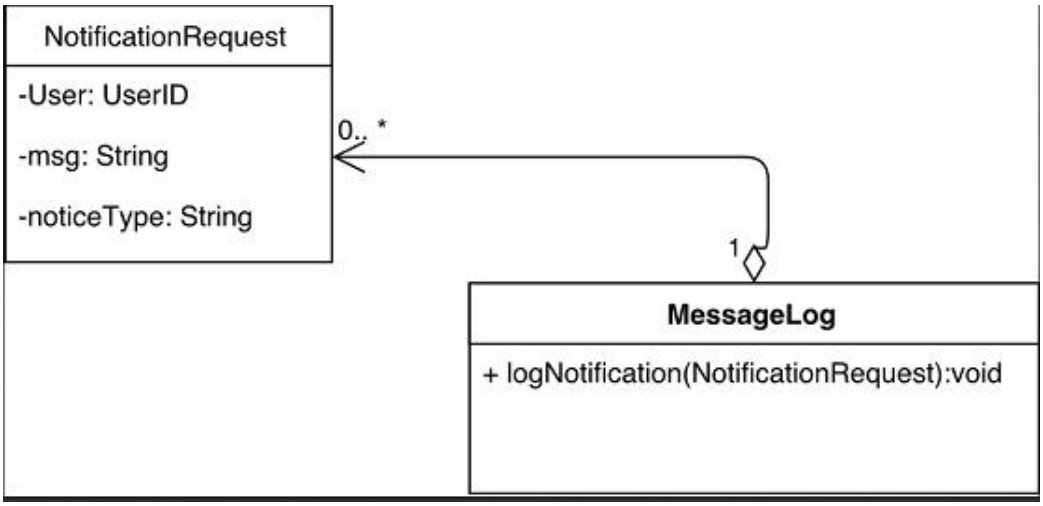

Figure 6: Domain model of Notifications

The domain model of the notification is simply a description of the type of notification the system will handle. Notifications will be requested based on the threads the user has subscribed to.

2.3.5 Service contracts

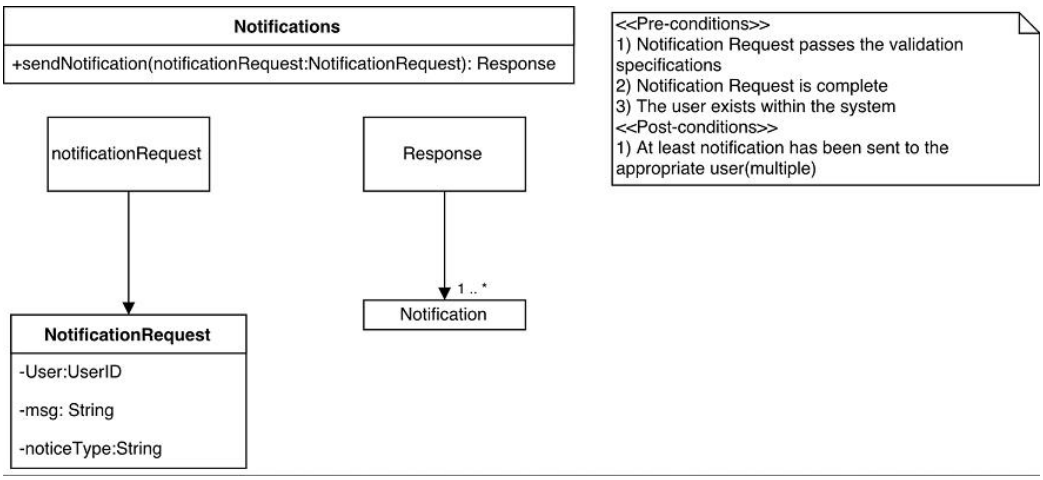

Figure 7: Domain model of Notifications

2.4 Messaging Module

The messaging model will be a service used to post messages to the forum. The messages will be in the form of either posts and comments. The design pattern used for this will be a composite pattern.

2.4.1 Requirements

2.4.2 Functional

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

2.4.3 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

2.4.4 Use cases

2.4.5 Create Post

This will be a functionality which will allow users to create new posts for the course module they belong to.

2.4.6 Create Comment

This will be a functionality which will allow users, that have the right privileges, to create new comment for a post.

2.4.7 Edit Post

This will be a service for users with higher privileges. It will allow them to edit various posts.

2.4.8 Edit Comment

This will be a service for users with higher privileges. It will allow them to edit various comments.

2.4.9 Remove Post

This will be a service for users with higher privileges. It will allow them to remove various posts. When a post is removed, all the posts and comments that accompany and fall under it will also be removed.

2.4.10 Remove Comment

This will be a service for users with higher privileges. It will allow them to remove various comments.

2.4.11 Move Post

This service will allow users with higher levels of clearance to move posts between different levels, in an upward direction only.

2.4.12 Domain Model

The domain model will follow a design pattern similar to that of a composite design pattern, in the following ways:

- A new post will be on level 0, making its heading the name of the thread.
- The level of a new post, under another post, will be an increment of the post that it falls under.
- A post cannot be created under a comment, and a comment can only be created under a post.

Figure 8 shows the diagram of the domain model, for the messaging module.

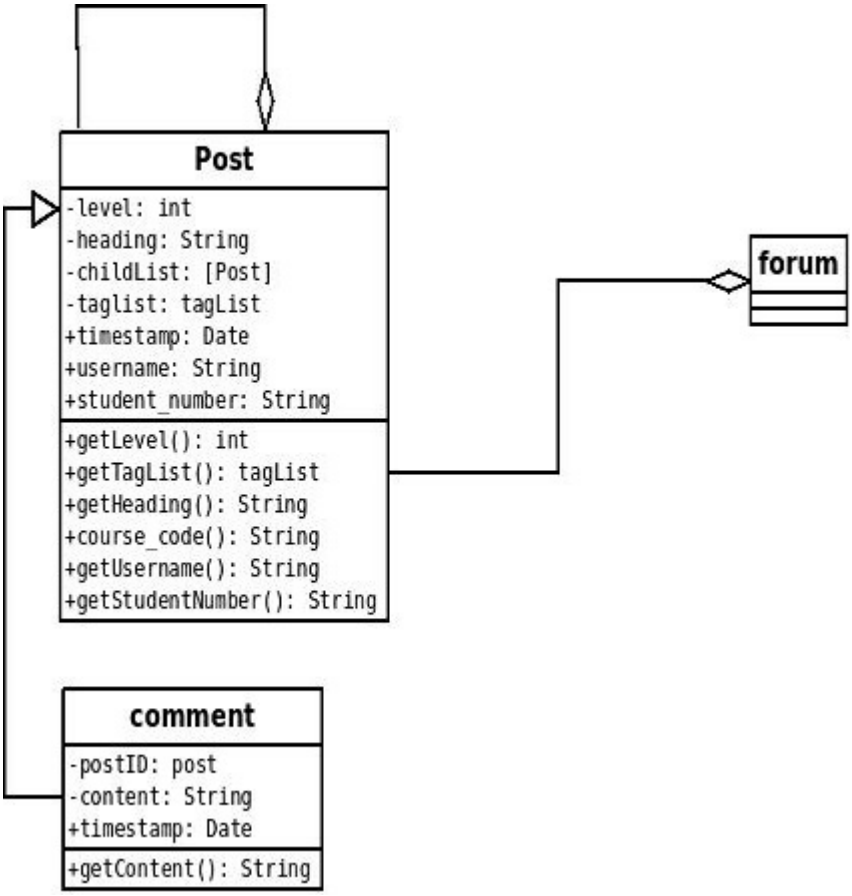

Figure 8: Domain model representing the message model

2.5 Report module

The buzz Report module is used to provide statistical information that can be used by the lecture to observe overall usage of the forum and award rewards to users that participate constructively to the forum. Management can also use the data collected to make updates and maintain the system.

2.5.1 Use case

The reporting module provides services to gather statistical information for each user and export a report back to the lecture, who can assess the user's overall usage of the forum.

2.5.2 getTotalPosts

The system will store the assessed user profile to generate a total for the number of posts a user has made. This statistical information will be saved and exported to the lecture/management.

2.5.3 Service contracts

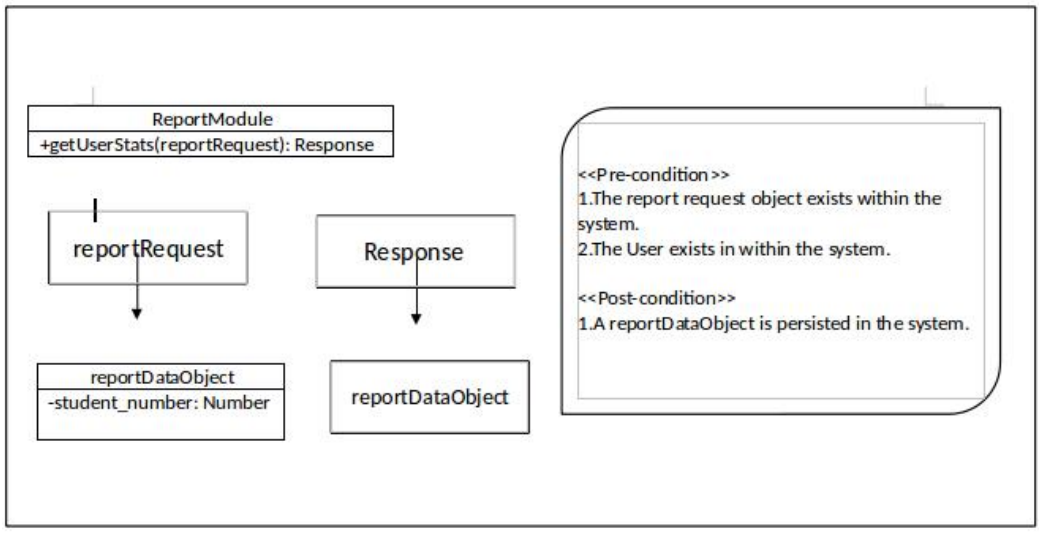

Figure 9: Service contracts of getTotalPosts.

2.5.4 ActiveUsers

The system will store and return the number of active users in the forum. Management can use this statistical information to perform maintenance if server overloading occurs and also manage the resources the system uses.

2.5.5 Domain Model

2.6 Buzz Forum-Modules Module

The Buzz Forum-Modules module models the course spaces of the forum and separates forum topics into their respective course modules (ie: COS 121 and COS 212 will have their own Buzz Forum-Module)

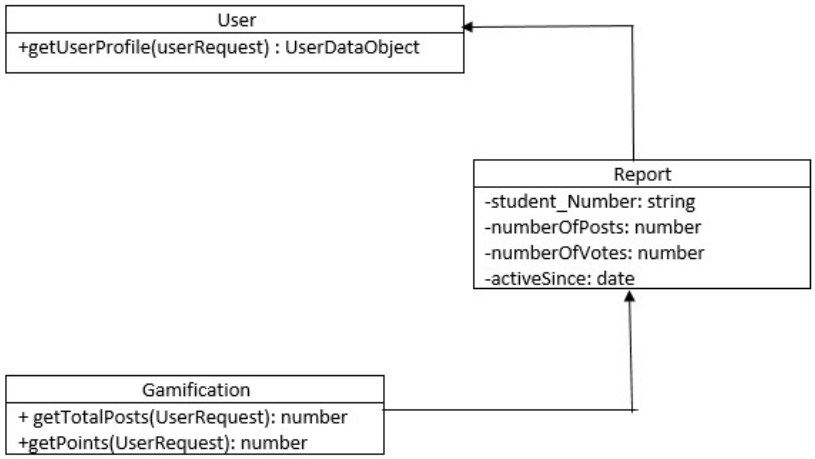

Figure 10: Domain model of the Buzz-Report module.

2.6.1 Scope

The scope of the Buzz-Subscriptions Module is modeled by Figure 11. Users with the appropriate privilege will be able to CRUD and manage Forum-Modules

2.7 Authorization Module

This module will be used to verify all requests a user makes, therefore acting as a bilayer between modules allowing or denying permission. This includes all types of users by viewing their status level and deducing what they are capable of doing. The result from this module can either be granted or denied this is represented by what it returns a boolean value "true" representing granted and an exception representing access being denied.

2.7.1 Use cases

The module will be used to allow or disallow users who have no access to certain feature by helping segregate peoples capabilities. The module will

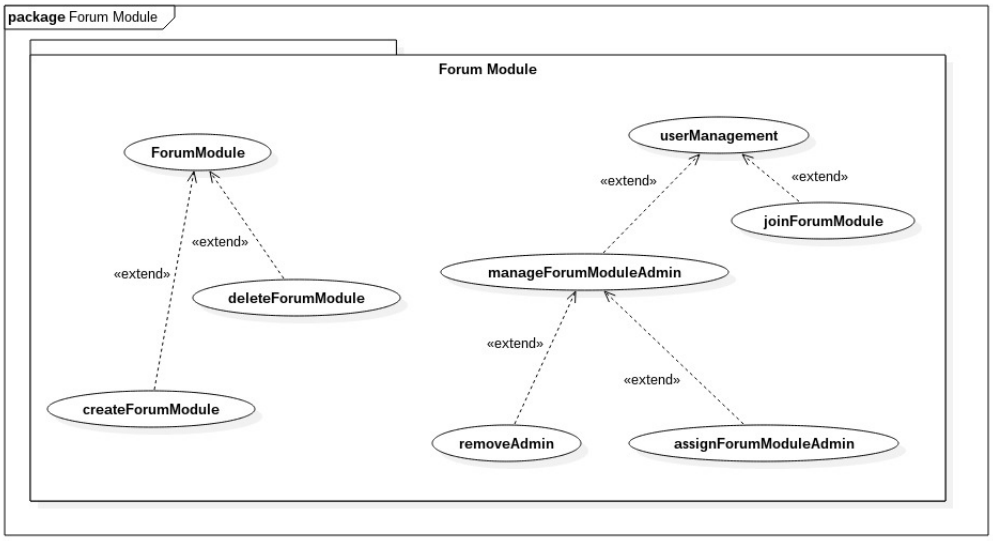

Figure 11: Scope of the Buzz-ForumModule Module.

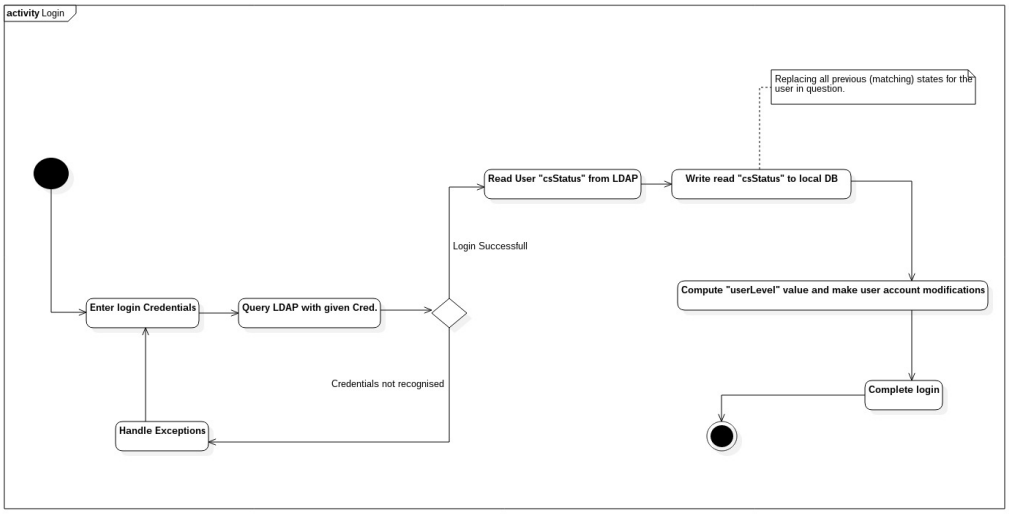

Figure 12: Sequence diagram.

be used to allow or disallow users who have no access to certain feature by helping segregate peoples capabilities. This will be used in a flexible manner

as certain aspects of the system will be able to be given restrictive features, or determining the amount of points must be accumulated as assigned by the administrator.

2.7.2 increaseAuthorizationLevel

2.7.3 Service contracts

This is in the event that the administrator finds that a feature of the website is too great or needs to be further earned before usage.

Service Contract This is achieved by increasing the level required to access it as a whole so one would have to be of a certain standard before utilizing this function.

2.7.4 decreaseAuthorizationLevel

This is in the event that a feature is to become available for complete public use by the people by decreasing the level of access required to use this said feature such as a lecturer reducing the points in the event that they are not enough events / practicals / online activities for students to accumulate enough points to use the features.

2.7.5 lockAuthorizationLevel

This is done to prevent any users from utilizing this feature by increasing it's authorization degree to the point where it is inaccessible to any non-administrative user aka making it part of the administrators features only.

2.7.6 isAuthorized

This is done to actually verify if a user may be permitted to utilize this feature by getting their status level as a parameter to which it is decided if they have access to the feature or denied and if so, what constraints also apply it such as amount of times they can use the feature etc.

2.8 Buzz-Gamification Module

This module will be used to add another dimension to the online chat platform by including a gamifying factor which will help make the system as a

whole interactive as one can measure the progress, not only does it make it exciting but also displays participation in an active form.

2.8.1 Scope

Majority of this performed in the back end of the system, to which the administrator has the ability to change/assign titles as well as bounty requirements.

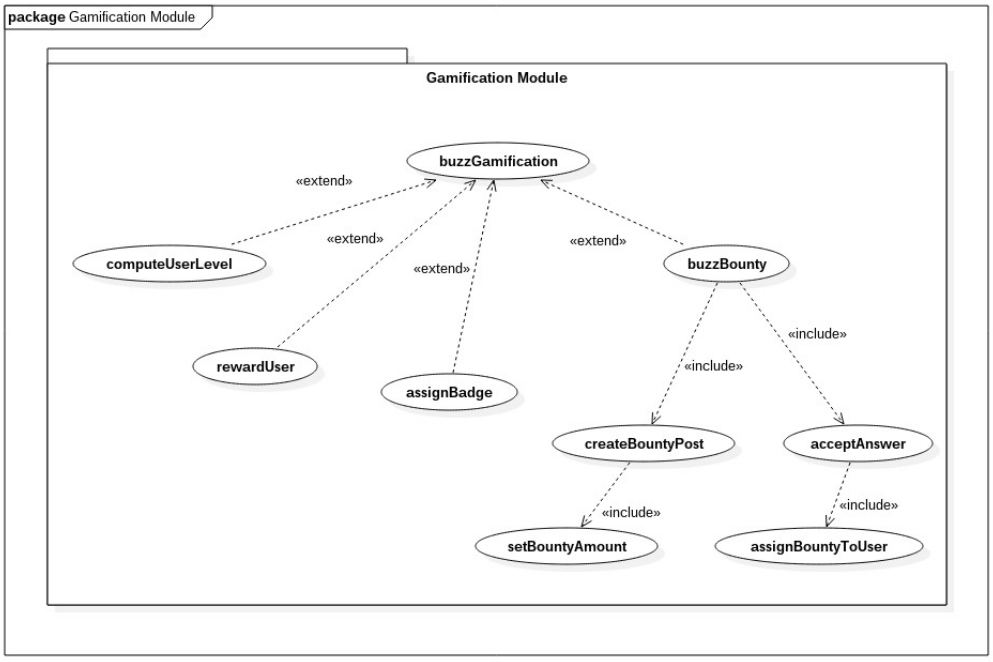

Figure 13: Scope of the Buzz-Gamification Module.

2.8.2 Use cases and Service Contracts

2.8.2.1 Use case and Service Contract: calculateStatus

Use Case- calculateStatus This use case calculates the user's status using their bounty points

Service Contract- calculateStatus
This is used to determine what is the current status of a user, using a

formula that is based on the current level their currently on

2.8.2.2 Use case and Service Contract: statusPromotion

Use Case- statusPromotion
This is in the event that a user's status is to be promoted.

Service Contract- statusPromotion
A person is given a new status due to them accumulating enough points to deserve a promotion, after their status is calculated.

2.8.2.3 Use case and Service Contract: statusPromotion

Use Case- statusPromotion
This is in the event that a user's status is to be promoted.

Service Contract- statusPromotion
A person is given a new status due to them accumulating enough points to deserve a promotion, after their status is calculated.

2.8.2.4 Use case and Service Contract: statusDemotion

Use Case- statusDemotion
This is in the event that a user's status is to be demoted.

Service Contract- statusDemotion
A person is given a prior status, after their status is calculated, due to them loosing enough points to deserve a demotion.

2.8.2.5 Use case and Service Contract: increaseBounty

Use Case- increaseBounty
This is in the event that a user has done something to have their points increase.

Service Contract- increaseBounty
A person has their bounty points increased due to them performing a positive act of participation on the site, such as posting, or having another user do something in relation to their action such as like their post.

2.8.2.6 Use case and Service Contract: decreaseBounty

Use Case- decreaseBounty
This is in the event that something has occurred that must decrease the bounty points of the user.

Service Contract- decreaseBounty
A person has their bounty points decreased due to a negative event taking place such as having another user do something such as dislike their post.

2.8.2.7 Use case and Service Contract: assignTitles

Use Case- assignTitles
This is an administrative feature that enables him to further adjust aspects of the game.

Service Contract- assignTitles The administrator is able to set the name and number of different titles (levels) the users can obtain as well the number of points required to achieve each.

2.8.3 Domain Model

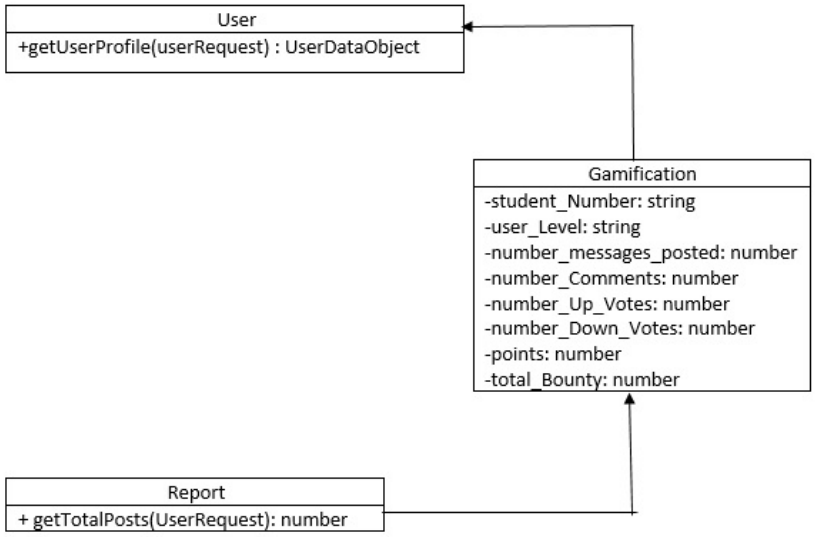

Figure 14: Domain model of the Gamification module.

2.9 Buzz-Subscriptions Module

The Buzz-Subscriptions module handles content subscriptions made by the user. These include subscriptions to Posts and/or forum topics.

2.9.1 Scope

The scope of the Buzz-Subscriptions Module is modeled by Figure 25. Users have the option of subscribing to particular Posts or Forum topics. The User will then have the option to receive Notifications via email on the content of their subscriptions.

3 Glossary

- **Software Architecture** - Structured solution that meets all of the technical and operational requirements of the system.
- **Service oriented software** - Style of software design where services are provided to the other components by application components, through

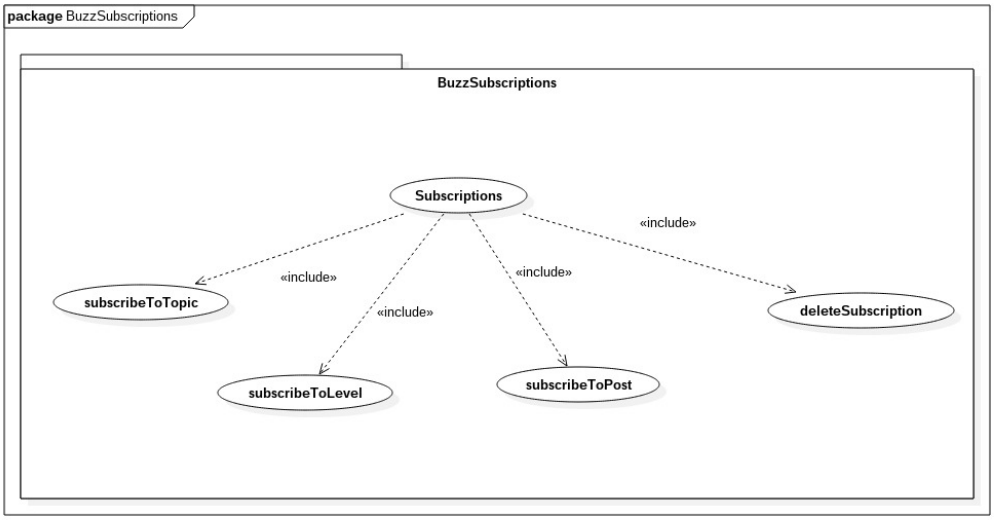

Figure 15: Scope of the Buzz-Subscriptions Module.

a communication protocol over a network.

- **Coupling** - a measure of how closely connected two routines or modules are.
- **Model-View Controller** - Software architectural pattern for implementing user interfaces on computers by dividing a given application into three interconnected parts.
- **RSA encryption** - A public key cryptography algorithm used in the encrypting and decrypting of messages.
- **DB-Schema** - Database Schema is a structure used to describe the organization of data within the database.
- **csStatus** - The role of the user for a specific module i.e. lecture, student, tutor
- **Netiquette** - The correct or acceptable way of communicating on the Internet.