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 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.

1.4 Design Requirement

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 1: Scope of the Buzz-Users Module.

2.1.2 Use cases

getUserProfile

2.1.3 Service contracts

2.1.4 Functional Req's

2.1.5 Domain Model

2.1.6 Service contracts

2.1.7 Technologies

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

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

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

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.

Get Status Symbol

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

2.2.3 Domain model

The figure of the CS-Status domain model is in figure 2.

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.

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.

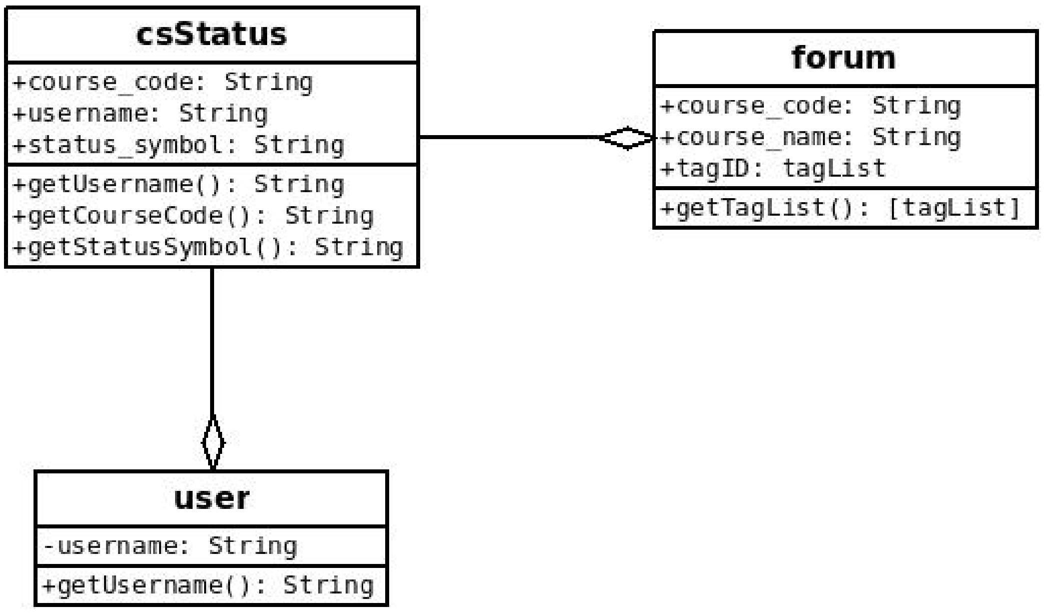


Figure 2: Domain model representing the message model

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

2.3.3 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 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 *insert figure*

2.4 Messaging model

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

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

2.4.2 Use cases

Create Post

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

Create Comment

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

Edit Post

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

Edit Comment

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

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.

Remove Comment

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

Move Post

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

2.4.3 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 3 shows the diagram of the domain model, for the messaging module.

2.5 Gamification module

The buzz Gamification module is used to manage the game-like presentation of the discussion board by assessing the user contribution, award rewards, and to restrict functionality in the system based on the user level.

2.6 —

Scope

The scope of the Gamification module is shown in the figure below

2.6.1 Use case

The use case of the gamification module provide functionality around assessing profiles, awarding rewards, bounty and user level calculation.

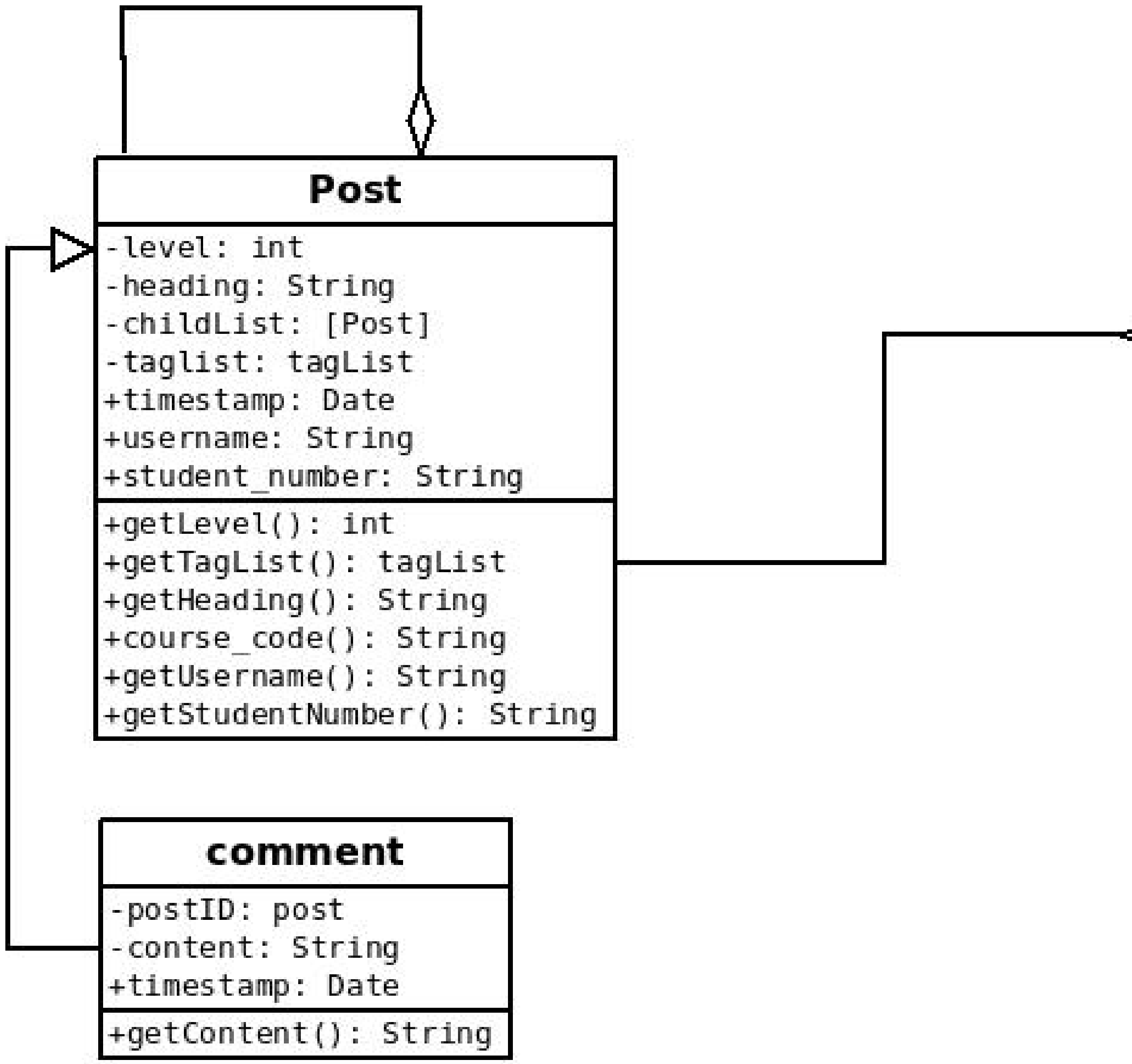
2.6.2 computeUserLevel

The system assesses the user profile to calculate the user level in order to restrict the systems functionality and award rewards to a user. The user is assigned a badge according to their level, calculated from the user's activity, within the system.

2.6.3 Service contracts

2.6.4 rewardUser

The system will allow the user to be rewarded based on their participation within the system, proper use of the system and comments made. A user post receives feedback from posts they make, with the management overseeing the feedback and approval of rewards.



mesg

Figure 3: Domain model representing the message model

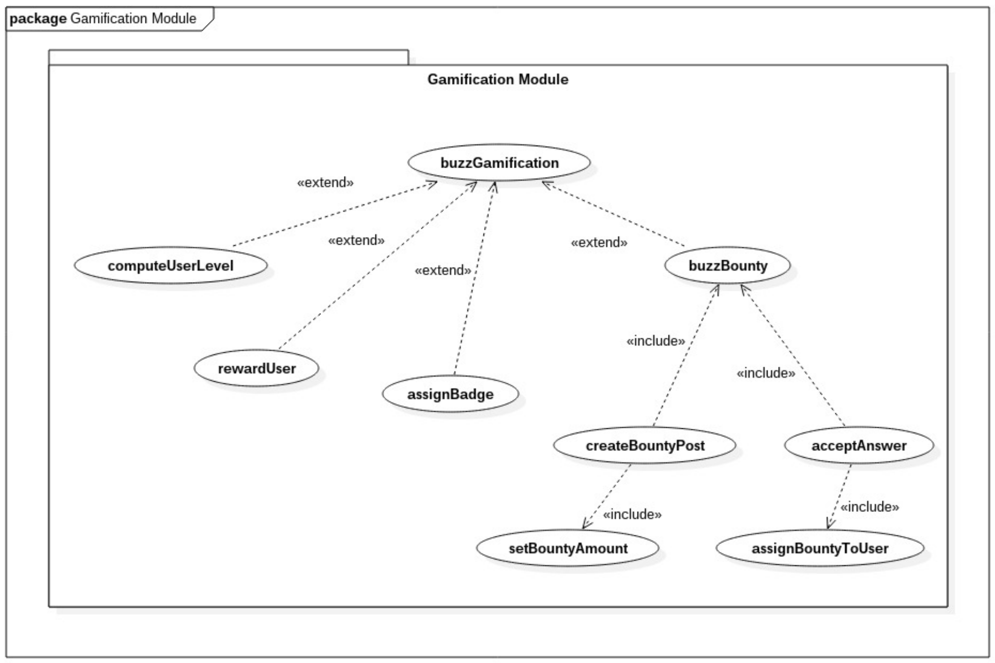


Figure 4: Scope of the Buzz-Gamification module.

2.6.5 Service contracts

2.6.6 createBountyPost

The system will allow the user to create a post and attach a bounty. The bounty functions as points and are given to the user that answers the question correctly. User points can be converted to bounty when creating a post.

2.6.7 acceptAnswer

The system will allow the management of the system oversee and approve an answer to a question posted. Based on his approval, a user may gain or lose their bounty.

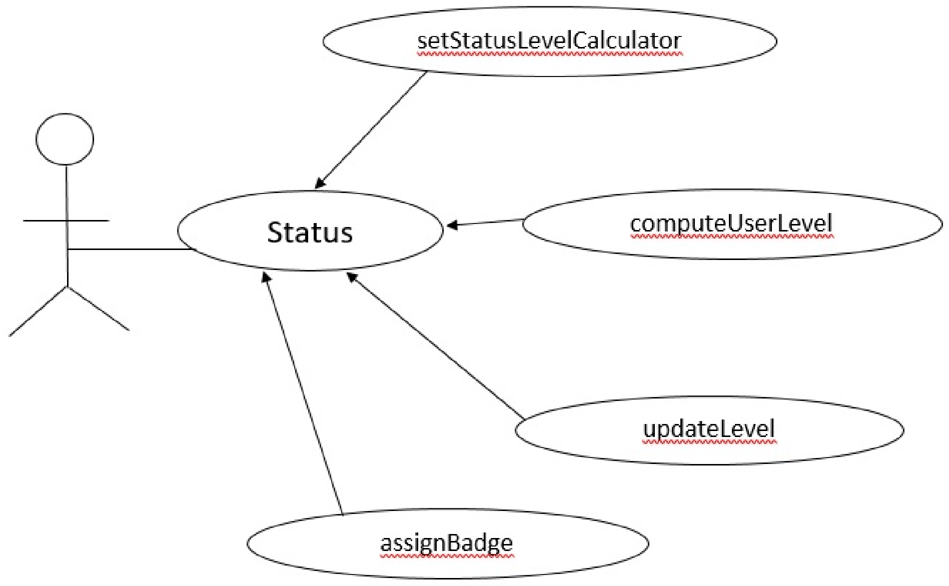


Figure 5: Service contracts of computeUserLevel.

2.6.8 Domain Model

2.7 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.

2.8 —

Scope

The scope of the Report module is shown in the figure below

2.8.1 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.

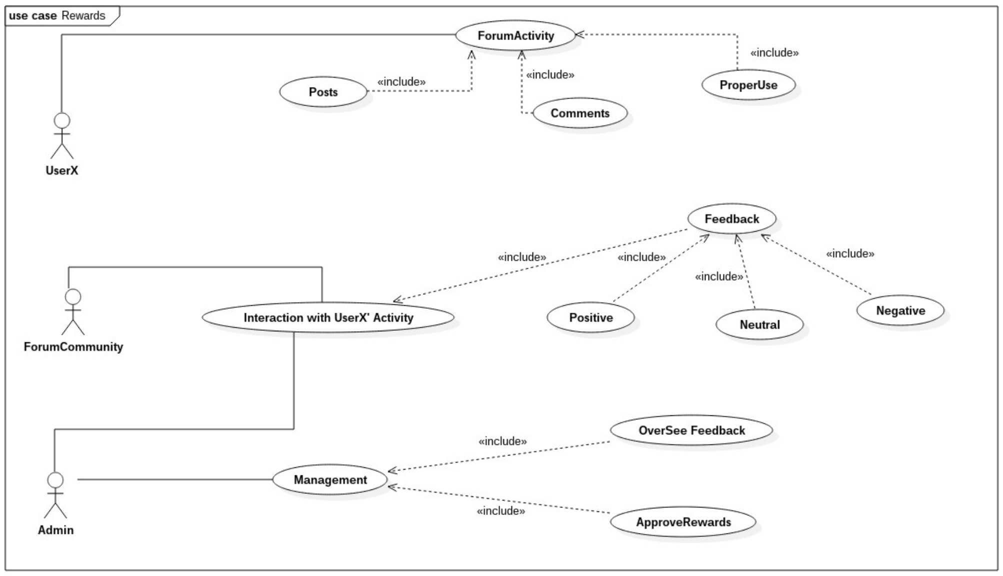


Figure 6: Service contracts of rewardUser.

2.8.2 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.

2.8.3 Service contracts

2.8.4 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.

2.8.5 Domain Model

2.9 Buzz Forum-Modules Module

The Buzz Forum-Modules module models the course spaces of the forum and separates forum topics into their respective course mudules (ie: COS 121 and

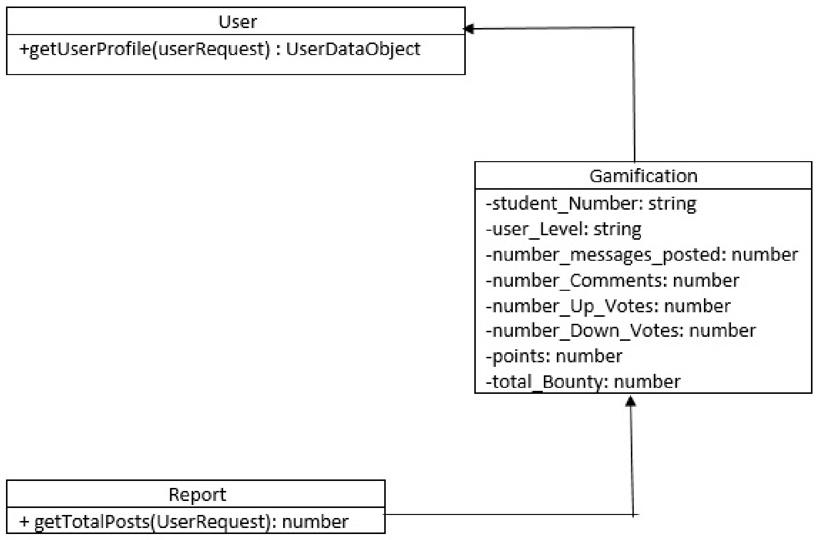


Figure 7: Domain model of the Buzz-Gamification module.

Figure 8: Scope of the Buzz-Report Module.

COS 212 will have their own Buzz Forum-Module)

2.9.1 Scope

The scope of the Buzz-Subscriptions Module is modeled by Figure 22. Users with the appropriate privilege will be able to CRUD and manage Forum-Modules

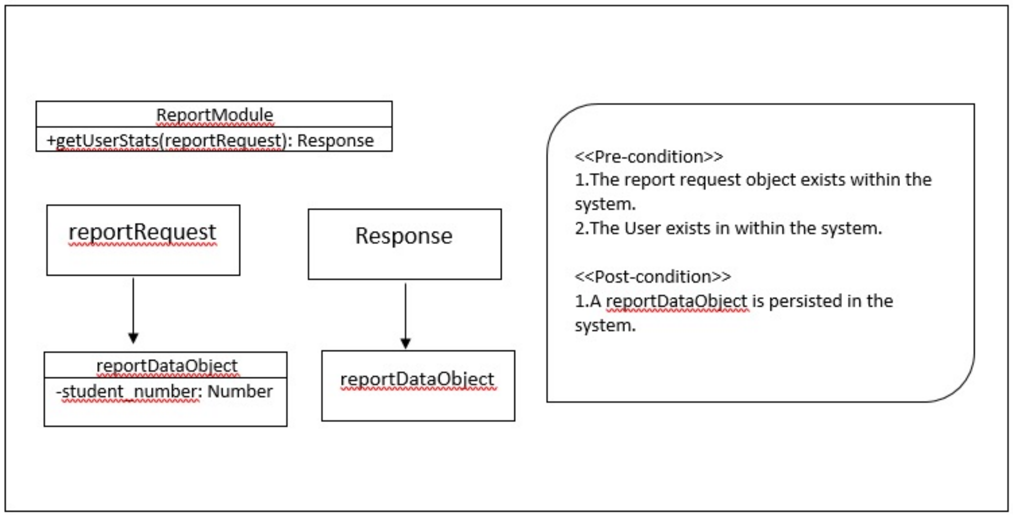

ReportModule
+getUserStats(reportRequest): Response
reportRequest
Response
reportDataObject
-student_number: Number
reportDataObject
<<Pre-condition>>
1.The report request object exists within the system.
2.The User exists in within the system.
<<Post-condition>>
1.A reportDataObject is persisted in the system.

Figure 9: Service contracts of getTotalPosts.

2.9.2 Use cases

2.9.3 Service contracts

2.9.4 Functional Req's

2.9.5 Domain Model

2.10 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.

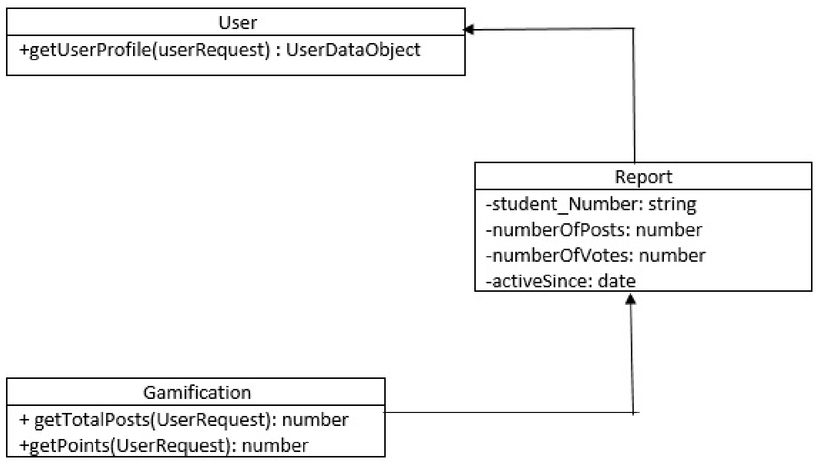

User
+getUserProfile(userRequest) : UserDataObject
Report
-student_Number: string
-numberOfPosts: number
-numberOfVotes: number
-activeSince: date
Gamification
+ getTotalPosts(UserRequest): number
+getPoints(UserRequest): number

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

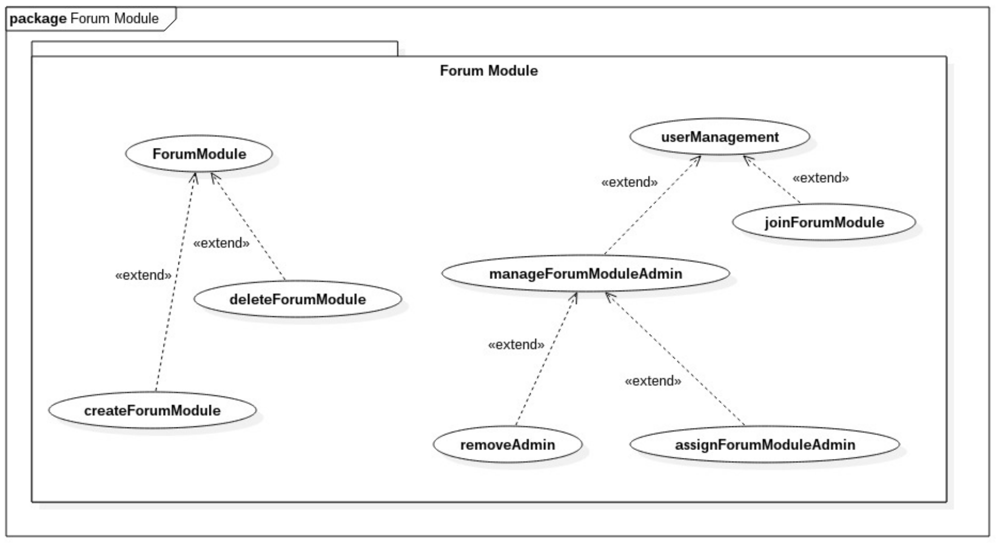


Figure 11: Scope of the Buzz-ForumModule Module.

Figure 12: Scope of the Buzz-Authorization Module.

2.10.1 Scope

2.10.2 Use cases

2.10.3 Service contracts

2.10.4 Functional Req's

2.10.5 Domain Model

2.11 Data-Sources Module

Intro to Module

2.11.1 Scope

Figure 13: Scope of the Buzz-DataSources Module.

2.11.2 Use cases

The module will be used to allow or disallow users who have no access to certain feature by helping segragate peoples capabilities. The module will be used to allow or disallow users who have no access to certain feature by helping segragate peoples capabilities. This will be used in a flexibile manner as certain aspects of the system will be able to be given restrictive features, or determining the amount of points must be accumilated as assignmed by the administrator.

2.11.3 increaseAuthorizationLevel

2.11.4 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.

2.11.5 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 accumilate enough points to use the features.

2.11.6 lockAuthorizationLevel

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

2.11.7 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.

2.11.8 Functional Req's

2.11.9 Domain Model

2.12 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.

2.12.1 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.

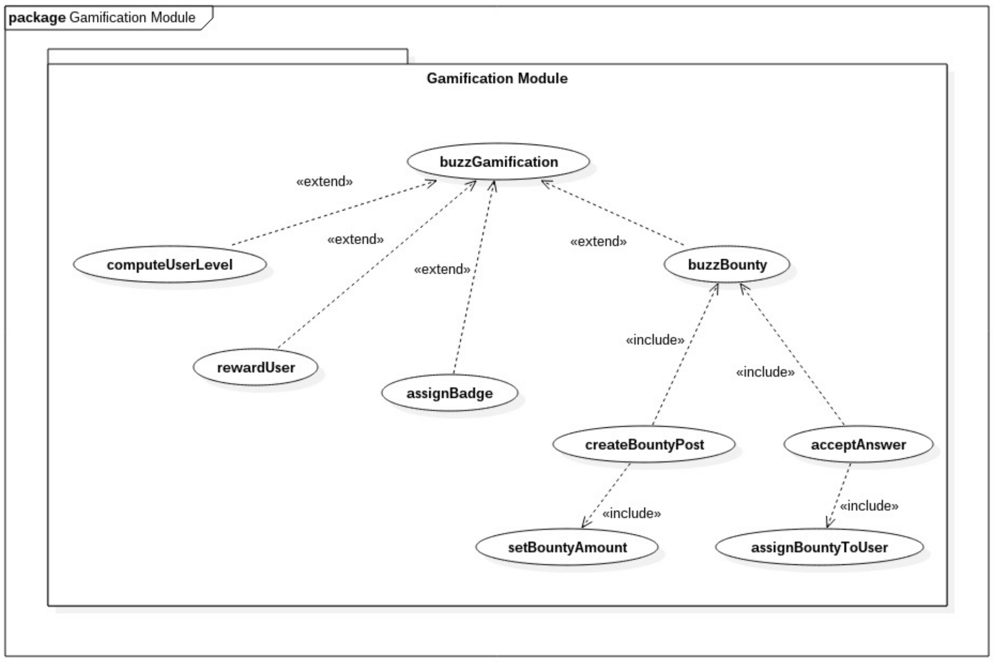


Figure 14: Scope of the Buzz-Gamification Module.

2.12.2 Use cases and Service Contracts

2.12.2.1 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

2.12.2.2 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.

2.12.2.3 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.

2.12.2.4 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.

2.12.2.5 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.

2.12.2.6 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.

2.12.2.7 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.

2.12.3 Functional Req's

2.12.4 Domain Model

2.13 Reporting and Data Module

Intro to Module

2.13.1 Scope

2.13.2 Use cases

2.13.3 Service contracts

2.13.4 Functional Req's

2.13.5 Domain Model

2.14 Buzz-Subscriptions Module

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

2.14.1 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

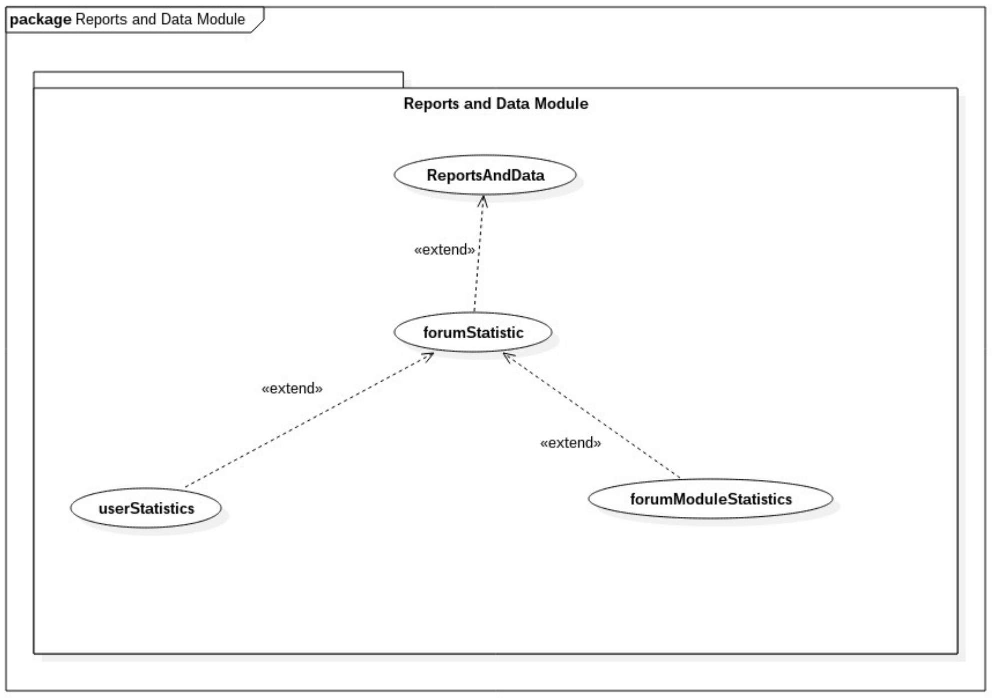


Figure 15: Scope of the Buzz Reports and Data Module.

will then have the option to receive Notifications via email on the content of their subscriptions.

2.14.2 Use cases

2.14.3 Service contracts

2.14.4 Functional Req's

2.14.5 Domain Model

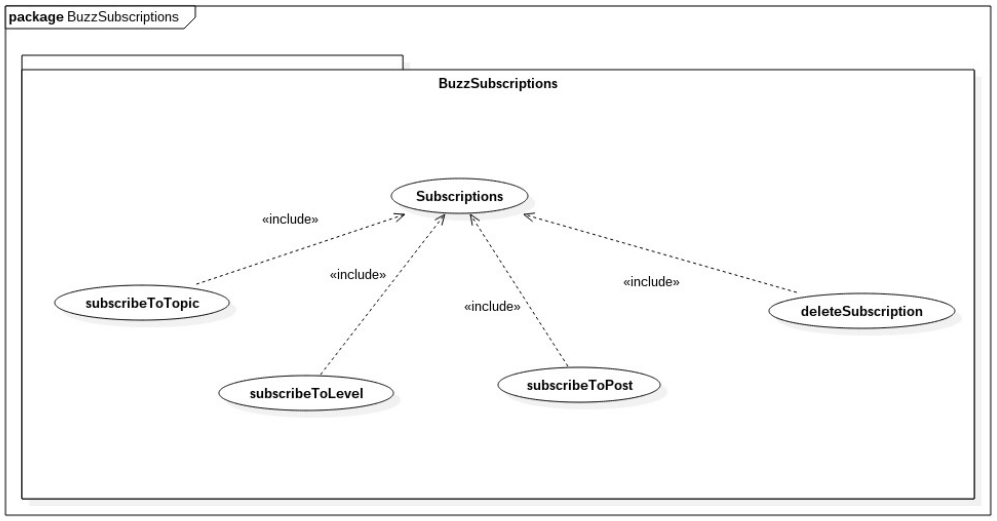


Figure 16: Scope of the Buzz-Subscriptions Module.