**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

**Q & A Platform for Educators**

|  |  |
| --- | --- |
| **Group 11** | |
| **Group members** | Kha Hoàng Minh – Team Leader – SE60979  Trương Như Khang – Team Member – SE60992 |
| **Supervisor** | Nguyễn Huy Hùng |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | QAPE |

-Ho Chi Minh City, May 11 2015-

*This page is intentionally left blank*

# Table of Contents

[Table of Contents 3](#_Toc427441698)

[List of Tables 6](#_Toc427441699)

[List of Figures 7](#_Toc427441700)

[Definitions, Acronyms, and Abbreviations 10](#_Toc427441701)

[A. Report No. 1 Introduction 11](#_Toc427441702)

[1. Project Information 11](#_Toc427441703)

[2. Introduction 11](#_Toc427441704)

[3. Current Situation 11](#_Toc427441705)

[4. Problem Definition 11](#_Toc427441706)

[5. Proposed Solution 12](#_Toc427441707)

[5.1 Feature functions 12](#_Toc427441708)

[5.2 Advantages and disadvantages 12](#_Toc427441709)

[6. Functional Requirements 12](#_Toc427441710)

[7. Role and Responsibility 13](#_Toc427441711)

[B. Report No.2 Software Project Management Plan 14](#_Toc427441712)

[1. Problem Definition 14](#_Toc427441713)

[1.1 Name of this Capstone Project 14](#_Toc427441715)

[1.2 Problem Abstract 14](#_Toc427441716)

[1.3 Project Overview 14](#_Toc427441717)

[2. Project organization 17](#_Toc427441718)

[2.1 Software Process Model 17](#_Toc427441719)

[2.2 Roles and responsibilities 17](#_Toc427441720)

[2.3 Tools and Techniques 18](#_Toc427441721)

[3. Project Management Plan 19](#_Toc427441722)

[3.1 Product Backlog 19](#_Toc427441723)

[3.2 Sprint Backlog 20](#_Toc427441724)

[3.3 All Meeting Minutes 20](#_Toc427441725)

[4. Coding Convention 20](#_Toc427441726)

[C. Report No. 3 Software Requirement Specification 21](#_Toc427441727)

[1. User Requirement Specification 21](#_Toc427441728)

[1.1 Guest Requirement 21](#_Toc427441729)

[1.2 Student Requirement 21](#_Toc427441730)

[1.3 Teacher Requirement 21](#_Toc427441731)

[1.4 Admin Requirement 22](#_Toc427441732)

[2. System Requirement Specification 22](#_Toc427441733)

[2.1 External Interface Requirement 22](#_Toc427441734)

[2.2 System Overview Use Case 23](#_Toc427441735)

[2.3 List of Use Case 24](#_Toc427441736)

[3. Software System Attribute 92](#_Toc427441737)

[3.1 Usability 92](#_Toc427441738)

[3.2 Reliability 92](#_Toc427441739)

[3.3 Availability 92](#_Toc427441740)

[3.4 Security 92](#_Toc427441741)

[3.5 Maintainability 92](#_Toc427441742)

[3.6 Portability 92](#_Toc427441743)

[3.7 Performance 92](#_Toc427441744)

[4. Conceptual Diagram 93](#_Toc427441745)

[D. Report No. 4 Software Design Description 94](#_Toc427441746)

[1. Design Overview 94](#_Toc427441747)

[2. System architecture design 94](#_Toc427441748)

[2.1 Web architecture design 95](#_Toc427441749)

[3. Component Diagram 96](#_Toc427441750)

[4. Detailed Description 96](#_Toc427441751)

[4.1 Class Diagram 96](#_Toc427441752)

[4.2 Class Diagram Explanation 98](#_Toc427441753)

[4.3 Sequence Diagram 105](#_Toc427441754)

[5. User Interface Design 120](#_Toc427441755)

[5.1 Guest Interface Design 120](#_Toc427441756)

[5.2 Teacher Interface Design 121](#_Toc427441757)

[5.3 Student Interface Design 136](#_Toc427441758)

[5.4 Confirm popup 141](#_Toc427441759)

[6. Database Design 142](#_Toc427441760)

[6.1 Logical Diagram 142](#_Toc427441761)

[6.2 Data Dictionary 144](#_Toc427441762)

[7. Algorithms 144](#_Toc427441763)

[7.1 Calculating question’s score 144](#_Toc427441764)

[7.2 Calculate similar score of two String 145](#_Toc427441765)

[7.3 Arrange strings by similar score 146](#_Toc427441766)

[E. Report No. 5 System Implementation & Test 148](#_Toc427441767)

[1. Introduction 148](#_Toc427441768)

[1.1 Overview 148](#_Toc427441769)

[1.2 Test Approach 148](#_Toc427441770)

[2. Database Relationship Diagram 148](#_Toc427441771)

[2.1 Physical Diagram 148](#_Toc427441772)

[2.2 Data Dictionary 150](#_Toc427441773)

[3. Performance Measurement 152](#_Toc427441774)

[4. Test Plan 153](#_Toc427441775)

[4.1 Features to be tested 153](#_Toc427441776)

[4.2 Features not to be tested 153](#_Toc427441777)

[5. System Testing Test Case 154](#_Toc427441778)

[5.1 Teacher core flow 155](#_Toc427441779)

[5.2 Student core flow 167](#_Toc427441780)

[F. Report No. 6 Software User’s Manual 176](#_Toc427441781)

[1. Installation Guide 176](#_Toc427441782)

[1.1 Setting up environment at server side 176](#_Toc427441783)

[1.2 Deployment at server side 176](#_Toc427441784)

[1.3 Setting up the environment at client side 179](#_Toc427441785)

[2. User Guide 180](#_Toc427441786)

[2.1 Register 180](#_Toc427441787)

[2.2 Create classroom 181](#_Toc427441788)

[2.3 Create post 182](#_Toc427441789)

[2.4 Answer 182](#_Toc427441790)

[2.5 Merge question 183](#_Toc427441791)

[2.6 Invite student 184](#_Toc427441792)

[2.7 Create folder 184](#_Toc427441793)

[2.8 Add to folder 185](#_Toc427441794)

[G. Appendix 186](#_Toc427441795)

# List of Tables

[Table 1: Definitions 10](#_Toc427441796)

[Table 2: Roles and Responsibilities 13](#_Toc427441797)

[Table 3: Hardware Requirement for Server 16](#_Toc427441798)

[Table 4: Roles and Responsibilities Details 18](#_Toc427441799)

[Table 5: Product Backlog Detail 20](#_Toc427441800)

[Table 6: Entity data dictionary 93](#_Toc427441801)

[Table 7: Web architecture design 95](#_Toc427441802)

[Table 8: Component dictionary 96](#_Toc427441803)

[Table 9: Class dictionary 98](#_Toc427441804)

[Table 10 Logical Dictionary 144](#_Toc427441805)

[Table 11 Physical database Dictionary 150](#_Toc427441806)

[Table 12 Attribute Data Dictionary 152](#_Toc427441807)

# List of Figures

[Figure 1: Scrum Development Model 17](#_Toc427441808)

[Figure 2: System Overview Use Case 23](#_Toc427441809)

[Figure 3: <Guest> Overview Use Case 24](#_Toc427441810)

[Figure 4: <Guest> Login Use Case Diagram 24](#_Toc427441811)

[Figure 5: <Guest> Register Use Case Diagram 26](#_Toc427441812)

[Figure 6: <Teacher> Overview Use Case 28](#_Toc427441813)

[Figure 7: <Teacher> Create class Use Case Diagram 29](#_Toc427441814)

[Figure 8: <Teacher> Invite student Use Case Diagram 31](#_Toc427441815)

[Figure 9: <Teacher> Edit class Use Case Diagram 33](#_Toc427441816)

[Figure 10: <Teacher> Close class Use Case Diagram 35](#_Toc427441817)

[Figure 11: <Teacher> Reopen class Use Case Diagram 36](#_Toc427441818)

[Figure 12: <Teacher> Accept request of student Use Case Diagram 38](#_Toc427441819)

[Figure 13: <Teacher> Remove student Use Case Diagram 39](#_Toc427441820)

[Figure 14: <Teacher> Create post Use Case Diagram 41](#_Toc427441821)

[Figure 15: <Teacher> Edit post Use Case Diagram 43](#_Toc427441822)

[Figure 16: <Teacher> Close post Use Case Diagram 44](#_Toc427441823)

[Figure 17: <Teacher> Open post Use Case Diagram 46](#_Toc427441824)

[Figure 18: <Teacher> Delete post Use Case Diagram 47](#_Toc427441825)

[Figure 19: <Teacher> Upload material Use Case Diagram 49](#_Toc427441826)

[Figure 20: <Teacher> Delete material Use Case Diagram 50](#_Toc427441827)

[Figure 21: <Teacher> Invite teacher answering question Use Case Diagram 52](#_Toc427441828)

[Figure 22: <Teacher> Want answer Use Case Diagram 53](#_Toc427441829)

[Figure 23: <Teacher> Un-want answer Use Case Diagram 55](#_Toc427441830)

[Figure 24: <Teacher> Follow teacher Use Case Diagram 56](#_Toc427441831)

[Figure 25: <Teacher> Un-follow teacher Use Case Diagram 57](#_Toc427441832)

[Figure 26: <Teacher> Edit profile Use Case Diagram 58](#_Toc427441833)

[Figure 27: <Teacher> Merge question Use Case Diagram 60](#_Toc427441834)

[Figure 28: <Teacher> Search Use Case Diagram 63](#_Toc427441835)

[Figure 29: <Teacher> Answer Use Case Diagram 64](#_Toc427441836)

[Figure 30: <Teacher> Accept answer Use Case Diagram 66](#_Toc427441837)

[Figure 31: <Student> Overview Use Case 68](#_Toc427441838)

[Figure 32: <Student> Accept Invitation Use Case Diagram 69](#_Toc427441839)

[Figure 33: <Student> Request for joining class Use Case Diagram 70](#_Toc427441840)

[Figure 34: <Student> Leave class Use Case Diagram 71](#_Toc427441841)

[Figure 35: <Student> Create folder Use Case Diagram 73](#_Toc427441842)

[Figure 36: <Student> Delete folder Use Case Diagram 74](#_Toc427441843)

[Figure 37: <Student> Add material to folder Use Case Diagram 76](#_Toc427441844)

[Figure 38: <Student> Delete material Use Case Diagram 77](#_Toc427441845)

[Figure 39: <Student> Create post Use Case Diagram 78](#_Toc427441846)

[Figure 40: <Student> Edit post Use Case Diagram 78](#_Toc427441847)

[Figure 41: <Student> Delete post Use Case Diagram 79](#_Toc427441848)

[Figure 42: < Student> Want answer Use Case Diagram 80](#_Toc427441849)

[Figure 43: < Student> Un-want answer Use Case Diagram 80](#_Toc427441850)

[Figure 44: < Student> Follow teacher Use Case Diagram 81](#_Toc427441851)

[Figure 45: < Student> Un-follow teacher Use Case Diagram 81](#_Toc427441852)

[Figure 46: <Student> Edit profile Use Case Diagram 82](#_Toc427441853)

[Figure 47: <Student> Search Use Case Diagram 82](#_Toc427441854)

[Figure 48: <Student> Answer Use Case Diagram 83](#_Toc427441855)

[Figure 49: <Student> Accept answer Use Case Diagram 83](#_Toc427441856)

[Figure 50: <System> Overview Use Case 84](#_Toc427441857)

[Figure 51: <System> Push Notification Use Case Diagram 84](#_Toc427441858)

[Figure 52: <System> Feed News Use Case Diagram 86](#_Toc427441859)

[Figure 53: <Admin> Overview Use Case 87](#_Toc427441860)

[Figure 54: <Admin> Manage user Use Case Diagram 87](#_Toc427441861)

[Figure 55: <Admin> Search Use Case Diagram 89](#_Toc427441862)

[Figure 56: <Admin> Delete post Use Case Diagram 89](#_Toc427441863)

[Figure 57: <Admin> Close class Use Case Diagram 90](#_Toc427441864)

[Figure 58: <Admin> Reopen class Use Case Diagram 91](#_Toc427441865)

[Figure 59: <Admin> Delete material Use Case Diagram 91](#_Toc427441866)

[Figure 60: Conceptual diagram 93](#_Toc427441867)

[Figure 61: Component diagram 96](#_Toc427441868)

[Figure 62: Class diagram 97](#_Toc427441869)

[Figure 63: <Teacher> Create classroom sequence 105](#_Toc427441870)

[Figure 64: <Teacher> Invite student sequence 106](#_Toc427441871)

[Figure 65: <Teacher> Invite teacher sequence 107](#_Toc427441872)

[Figure 66: <Teacher> Upload material sequence 108](#_Toc427441873)

[Figure 67: <Teacher> Merge questions sequence 109](#_Toc427441874)

[Figure 68: <Teacher> Answer multi questions sequence 110](#_Toc427441875)

[Figure 69: <Teacher> Accept request sequence 111](#_Toc427441876)

[Figure 70: <Student> Request to join sequence 112](#_Toc427441877)

[Figure 71: <Student> Add to folder sequence 113](#_Toc427441878)

[Figure 72: <Teacher, Student> View news feed sequence 114](#_Toc427441879)

[Figure 73: <Teacher, Student> Create post sequence 115](#_Toc427441880)

[Figure 74: <Teacher, Student> Delete post sequence 116](#_Toc427441881)

[Figure 75: <Teacher, Student> Want answer sequence 117](#_Toc427441882)

[Figure 76: <Teacher, Student> Answers sequence 118](#_Toc427441883)

[Figure 77: <Teacher, Student> Accept answer sequence 119](#_Toc427441884)

[Figure 78: Login and register interface 120](#_Toc427441885)

[Figure 79: News feed interface 121](#_Toc427441886)

[Figure 80: Teacher Dashboard interface 123](#_Toc427441887)

[Figure 81: Create classroom interface 124](#_Toc427441888)

[Figure 82: Create post interface 125](#_Toc427441889)

[Figure 83: Teacher Classroom view interface 127](#_Toc427441890)

[Figure 84: Teacher Post view interface 129](#_Toc427441891)

[Figure 85: Update classroom interface 130](#_Toc427441892)

[Figure 86: Edit post interface 131](#_Toc427441893)

[Figure 87: Merge question interface 132](#_Toc427441894)

[Figure 88: Create new merged question interface 133](#_Toc427441895)

[Figure 89: Invite student popup interface 134](#_Toc427441896)

[Figure 90: Invite teacher popup interface 134](#_Toc427441897)

[Figure 91: Upload material popup interface 135](#_Toc427441898)

[Figure 92: Tracking interface 135](#_Toc427441899)

[Figure 93: Student Dashboard interface 136](#_Toc427441900)

[Figure 94: Student Classroom view interface 137](#_Toc427441901)

[Figure 95: Student Post view interface 138](#_Toc427441902)

[Figure 96: Manage material interface 139](#_Toc427441903)

[Figure 97: Folder view interface 139](#_Toc427441904)

[Figure 98: Add to folder interface 140](#_Toc427441905)

[Figure 99: Add to folder interface 141](#_Toc427441906)

[Figure 100: Confirm popup 141](#_Toc427441907)

[Figure 101: Logical diagram 143](#_Toc427441908)

[Figure 102: Physical database diagram 149](#_Toc427441909)

[Figure 103: Teacher core flow 154](#_Toc427441910)

[Figure 104: Student core flow 154](#_Toc427441911)

[Figure 105: Register form 180](#_Toc427441912)

[Figure 106: Create classroom form 181](#_Toc427441913)

[Figure 107: Create post form 182](#_Toc427441914)

[Figure 108: Answer form 182](#_Toc427441915)

[Figure 109: Merge questions form 183](#_Toc427441916)

[Figure 110: Create new merged question form 183](#_Toc427441917)

[Figure 111: Invite student form 184](#_Toc427441918)

[Figure 112: Create folder form 184](#_Toc427441919)

[Figure 113: Add to folder form 185](#_Toc427441920)

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| QAPE | Q & A Platform for Educators |

Table 1: Definitions

# Report No. 1 Introduction

## Project Information

* Project name: **Q&A Platform for Educators**
* Project Code: **QAPE**
* Product Type: **Web Application**
* Start Date: **May 11th, 2015**
* End Date: **August 11th, 2015**

## Introduction

This project is a Q&A Web application where students and teachers can exchange information and other educational materials. Students can ask question, learn more from articles or lectures from teachers or find the answers from a lot of professionals. Teachers can share their knowledge or their material and support their student’s by answering questions.

## Current Situation

Teaching and studying activities take place at school. Students ask for their problems at school time or sometime via email. Teachers teach, share their knowledge, their experience to student and answers student’s questions.

Students also ask a question using existed Q&A websites. There is a lot of popular Q&A websites like Quora, Yahoo! Answers, Stack Overflow… People can post threads or answer the questions pretty easy using its account or connect to other users. These sites also have rating system for questions and responses in order to rate users as well.

Teachers sometimes build their owned blog. In their blog, they post articles, lecturers and materials in order to sharing their knowledge.

## Problem Definition

Because of school time is limited, sometime teachers cannot answer all student’s questions. Moreover, teachers always answer questions that have similar content or answer a lot of emails for same problem. Students are difficult to find teachers when they have not yet understood problem. Some of them also feel shy when asking.

Using existed Q&A website may have some problem. Quora and Yahoo! Answer are known as the Q&A famous website but their objects are about a lot of different issues in life. These websites doesn’t only focus on education so students and teachers will be difficult for sharing their knowledge. Spamming questions are also problems of them. On the other hand, Stack Overflow only focuses on programing rather than diversifying subjects. All of them do not yet support the true studying environment.

## Proposed Solution

We suggest developing a Q&A website that focuses on asking for multiple studying subjects. This website is such as a university in reality that includes many classrooms, teachers and students. It supports teachers share their additional lectures because of lacking of time. It also helps students ask self-confidently because some of them feel shy when asking in school.

### **Feature functions**

* Teachers have their classrooms with lectures and students. They can share additional lectures and studying materials, invites students join their classes. They also can invite other teachers together discuss about lectures or question.
* Students can ask about not yet understanding issues. They also join in classes to learn additional subject, follow their favorite teachers and manage studying material that uploaded by teachers.

### Advantages and disadvantages

The advantages and disadvantages of the proposed solution:

* Advantage:
* Teachers can upload their studying material (.doc, .pdf…) and students can manage it online.
* By supporting real time notification and discussion, students can imagine this website as the university with many classrooms, teachers and friends. They will feel free to ask anything.
* Users will have their right answers faster than they do on their own. That leads to learn better in real classes.
* System has a variety of categories that is better than Stack Overflow but it is not a mess such as Quora or Yahoo! Answers.
* Disadvantage:
* System has not yet supported user’s point to evaluate teachers.
* System has not yet supported suggesting teacher to join threads to answer the questions.
* System has not yet supported private mode for classroom
* Cost time waiting to approve to join a class.

## Functional Requirements

Function requirements of the system are listed as below:

**For Teachers:**

* Create classes.
* Create a post. Post types include: articles, questions, answer, and upload material.
* Merge question (also support answering multi questions)
* Invite other teachers to answer questions.
* Invite students to join the discussion in a classroom.
* Follow a teacher.
* Track students’ questions to see which topics a student is less known about.
* Support real time notification.
* Support real time in discussion.

**For Students:**

* Create a post: articles, questions, answer.
* Manage learning materials: allowing students to create folders to store different kinds of studying documents.
* Support real time notification.
* Support real time in discussion.
* Follow a teacher.
* Send a request to join in a teacher’s class.
* Support an easy way to add mathematic equation, code highlighting besides other normal text formatting for editor.
* Suggest related posts.

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
| **1** | Nguyễn Huy Hùng | Project Manager | Instructor | HungNH@fpt.edu.vn |
| **2** | Kha Hoàng Minh | Developer | Leader | MinhKHSE60979@fpt.edu.vn |
| **3** | Trương Như Khang | Developer | Member | KhangTNSE60992@fpt.edu.vn |

Table 2: Roles and Responsibilities

# Report No.2 Software Project Management Plan

## Problem Definition



### Name of this Capstone Project

* Q&A Platform for Educators (QAPE)

### Problem Abstract

Nowadays, in almost schools and university, the level of exchange knowledge between students and teachers are not much. A result of learning and teaching has not reached the highest efficiency. Teaching hour is pretty little so that teachers cannot answer all questions of students during class. Besides that, some students feel shy to ask about their problems that are not understood.

Therefore, building online Q&A system is urgently needed employment in the current educational context. This system will help teachers and students interact with each other through a Website application. Teachers can create their own classrooms for teaching on the website. These classrooms allow their students join and post question or article. Teachers answer the question in owned classroom or invite other teachers to join and answer that question. They can also upload materials and documents for sharing. Students ask question, answer existed questions if they know, share their knowledge by posting articles or download materials to PC or save to online folder.

### Project Overview

#### Current Situation

Teaching and studying via school or via email:

* Teachers and students exchange information at school or university
* Students send e-mail to ask for not yet understood issues
* Teachers reply e-mail for student’s questions

Teaching and studying via existed Q&A websites or teacher’s blog:

* Students post questions on existed Q&A websites and get answer from lot of users.
* Teachers can answer the questions on existed Q&A websites. Once they want to make clear problems by posting articles or lectures, they must include reference links
* Teachers create owned blogs and post their articles or sharing materials

The above situations have some disadvantages:

* School time is limited
* There’s no place for storing student’s questions or teacher’s articles
* Using email can cause receiving answer lately or duplicate questions
* Teachers cannot manage students
* Teachers can only answer if using existed Q&A websites, they cannot share their knowledge directly.
* Students can only read and ask about the lectures or article if using blogs, they cannot ask other issues directly.
* Teachers cannot share small materials directly.
* Discussion in some websites or blogs sometimes is not real time, teachers and students must refresh page when discussing
* Teachers cannot consult other teacher’s opinions
* There are lot of duplicated questions

This project is developed in order to help teachers and students exchange information conveniently. This is the way that teachers share and answer as most students as possible. This is also the way that students get answer quickly and easily or get more ideas from different teachers and other students. This project also has restrictions. It comes from spamming, missing teacher’s rate, no private classroom. The system not yet supports reporting bad question, article or users.

#### The Proposed System

This project is going to build a website that help teachers and students communicate with each other easier and get the answer from who what they want.

#### Website

**Teachers:**

Create classrooms

Create a post. Post types include: articles, questions, answer, and upload material

Merge similar questions

Invite other teachers to answer questions

Invite students to join the discussion in a classroom

Follow a teacher

Track students’ questions to see which topics a student is less known about

Real time notification

Real time in discussion

**Students:**

Create a post: articles, questions, answer

Manage learning materials: allowing students to create folders to store different kinds of studying documents.

Real time notification

Real time in discussion

Follow a teacher

Send a request to join in a teacher’s class.

Note: the editor must support an easy way to add mathematic equation, code highlighting besides other normal text formatting.

Suggest related posts

**Admin:**

Manage users

Manage classroom

Manage post

#### Mobile application

N/A

#### Boundaries of the System

This system is a Web application that helps student and teacher communicate with each other by Question and Answer.

There are three type of account: Student, Teacher, and Admin. They have different features.

#### Development Environment

##### Hardware requirements

**For server**

|  |  |  |
| --- | --- | --- |
| Windows | Minimum Requirements | Recommended |
| Internet Connection | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| Operating System | Ubuntu 10.04 | Ubuntu 10.04 |
| Computer Processor | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core  (12M Cache, 2.50 GHz) |
| Computer Memory | 1GB RAM | 2GB or more |

Table 3: Hardware Requirement for Server

##### Software requirements

* Handle Products and sprints backlog: MS Excel 2013.
* Implement Website and service: IntelliJ v13.1.6.
* Create and manage database for system: MySQL v5.6.24 and MySQL Workbench v6.6.3.
* Source control: Git hub.
* Create models and diagrams: StarUML.
* Communicate and meeting: Skype 7.0 and Google mail.
* System and platform development: Window 7.

## Project organization

### Software Process Model

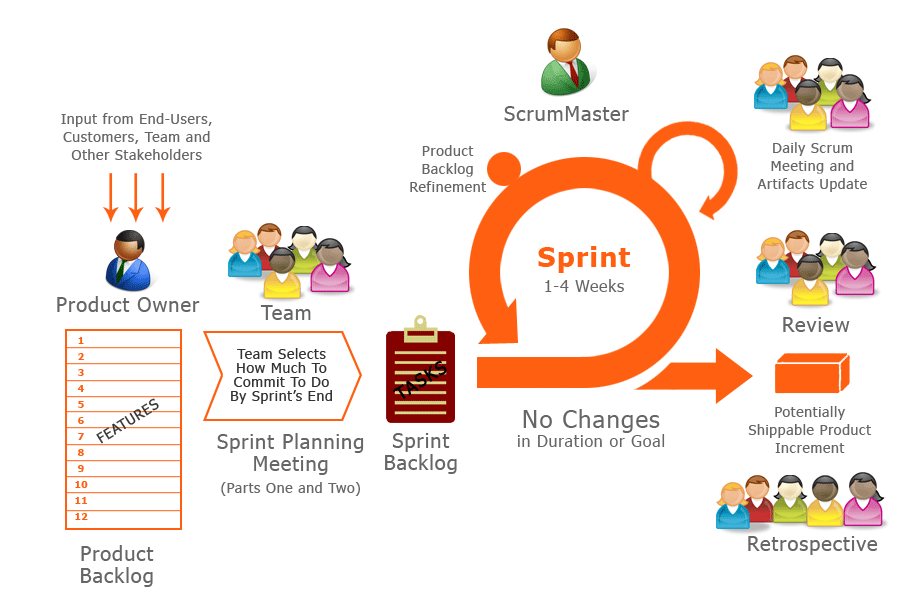


Figure 1: Scrum Development Model

Because of the project characteristics, we decide to make progress in series of sprints which are time boxed iterations in one week. At the beginning of one sprint, through sprint planning meeting, all team members will discuss together to define sprint backlog which is suitable to be completed within a week. Daily meeting and online conference are used to manage all activities and issues troubleshoot. At the end of sprint, product owner and developer team will review completed product to figuring out the necessary changes for products.

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| **1** | Nguyễn Huy Hùng | Scrum Master/Product Owner | * Specify user requirement * Control the development process * Give out technique and business analysis support |
| **2** | Kha Hoàng Minh | Team Leader, BA, DEV, Tester | * Managing process * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |
| **3** | Trương Như Khang | Team Member, BA, DEV, Tester | * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |

Table 4: Roles and Responsibilities Details

### Tools and Techniques

- Front-end technologies: HTML5, CSS3, JavaScript, jQuery, AJAX.

- Back-end technologies:

+Website: Java

+Framework: Spring.

- Web Server: Installed Tomcat 7.0.61.

- Database Management System: MySQL 5.6.

## Project Management Plan

### Product Backlog

|  |  |  |  |
| --- | --- | --- | --- |
| **Project: Q&A Platform for Educators** | | | |
|
| **ID** | **Function** | **Description** | **Added Sprint** |
| 1 | Register | Registering to become member of website |  |
| 2 | Login/Logout | Accessing the account to use their features |  |
| 3 | Edit profile | Users edit their profile |  |
| 4 | Create class | Teachers create class for student's joining |  |
| 5 | Create article, question, answer | User post an article, ask a question or answer a question |  |
| 6 | Upload material | Teachers upload their studying documents |  |
| 7 | Invite teacher | Teachers invite other teachers to answer a question on |  |
| 8 | Invite student | Teachers invite students to join the discussion in class |  |
| 9 | Accept/Deny an invitation | Users accept/deny teachers' invitation |  |
| 10 | Accept/Deny a request | Teachers accept/deny a joining request from student |  |
| 11 | Track student | Teachers track students’ questions to see which topics a student is less known about |  |
| 12 | Manage post | Users manage their post (edit, remove post), Admin only can remove post. |  |
| 13 | Manage class | Teachers manage students in their class |  |
| 14 | Real time notify | System support real time notification when followed teachers have new posts or posts have new reply |  |
| 15 | Real time discuss | New answers or replies will show at real time without refreshing page |  |
| 16 | Manage learning materials | Students to create folders to store different kinds of studying documents |  |
| 17 | Follow teacher | Students follow their favorite teachers |  |
| 18 | Send a request | Students send a request to join in a teacher’s class. |  |
| 19 | Suggest related post | System suggest related posts (related articles, related questions) for users |  |
| 20 | Search | Users search posts or other users |  |
| 21 | Manage users | Admin can activate/deactivate users |  |

Table 5: Product Backlog Detail

### Sprint Backlog

Preference to Appendix

### All Meeting Minutes

Place at Meeting Minute folder in CD.

## Coding Convention

Use Java coding convention to develop website.

Summary:

Naming Convention:

Use camel case for both variable and function name.

Use Pascal case for class name.

Indentation:

Avoid lines longer than 80 characters, since they are not handled well by many terminals and tools.

Declaration:

One declaration per line is recommended since it encourages commenting.

In absolutely no case should variables and functions be declared on the same line.

Do not put different types on the same line.

Code Examples:

Follow “Code Conventions for the Java TM Programming Language, by Sun Microsystems, rev April 20, 1999”.

# Report No. 3 Software Requirement Specification

## User Requirement Specification

### Guest Requirement

Guest is a person who doesn’t access to the system. There are some functions guest can use:

* Register
* Login

### Student Requirement

Student is a person who accesses to the system with role “student”. There are functions student can use:

* Manage owned questions/articles/answers (create new, edit and delete)
* Request to join/Leave classrooms
* Manage materials (add material to folder, delete material in folder)
* Manage folders (create new, delete)
* Want answer the questions
* Accept answers
* Follow/Un-follow teachers
* Edit profile
* Search
* Logout

### Teacher Requirement

Teacher is a person who accesses to the system with role “teacher”. There are functions teacher can use:

* Mange owned classrooms (create new, edit, close)
* Manage questions/articles/answers in owned classrooms (create new, edit, close)
* Manage owned questions/articles/answers (create new, edit, delete)
* Merge questions, answer multi questions
* Upload/Delete materials in owned classrooms
* Accept/Remove students in owned classrooms
* Invite students to join owned classrooms
* Invite teachers to answer questions in owned classrooms
* Merge questions in owned classrooms
* Answer multiple questions at once
* Want answer the questions
* Accept answers
* Follow/Un-follow teachers
* Edit profile
* Search
* Logout

### Admin Requirement

Admin is a person who has the right of administration. There are functions admin can use:

* Manage articles/questions/answers (delete)
* Manage users (activate, deactivate)
* Manage category (create new)

## System Requirement Specification

### External Interface Requirement

#### User Interface

* General requirement for graphics user interface is the GUI should be simple, clear, intuitive, and reminiscent.
* The interface design is an iterate process includes design, sketching, prototyping, user assessment.

#### Hardware Interface

* N/A

#### Software Interface

* Run with Google Chrome (Version 43.0.2357.132 m) and Cốc Cốc (Version 43.0.2357.126) browsers or newer version and must support JavaScript, web socket and html5.
* The screen of browsers must be at least 1024 x 768px to browse complete part of website.

#### Communication Protocol

* Using HTTP/HTTPS protocol

### System Overview Use Case

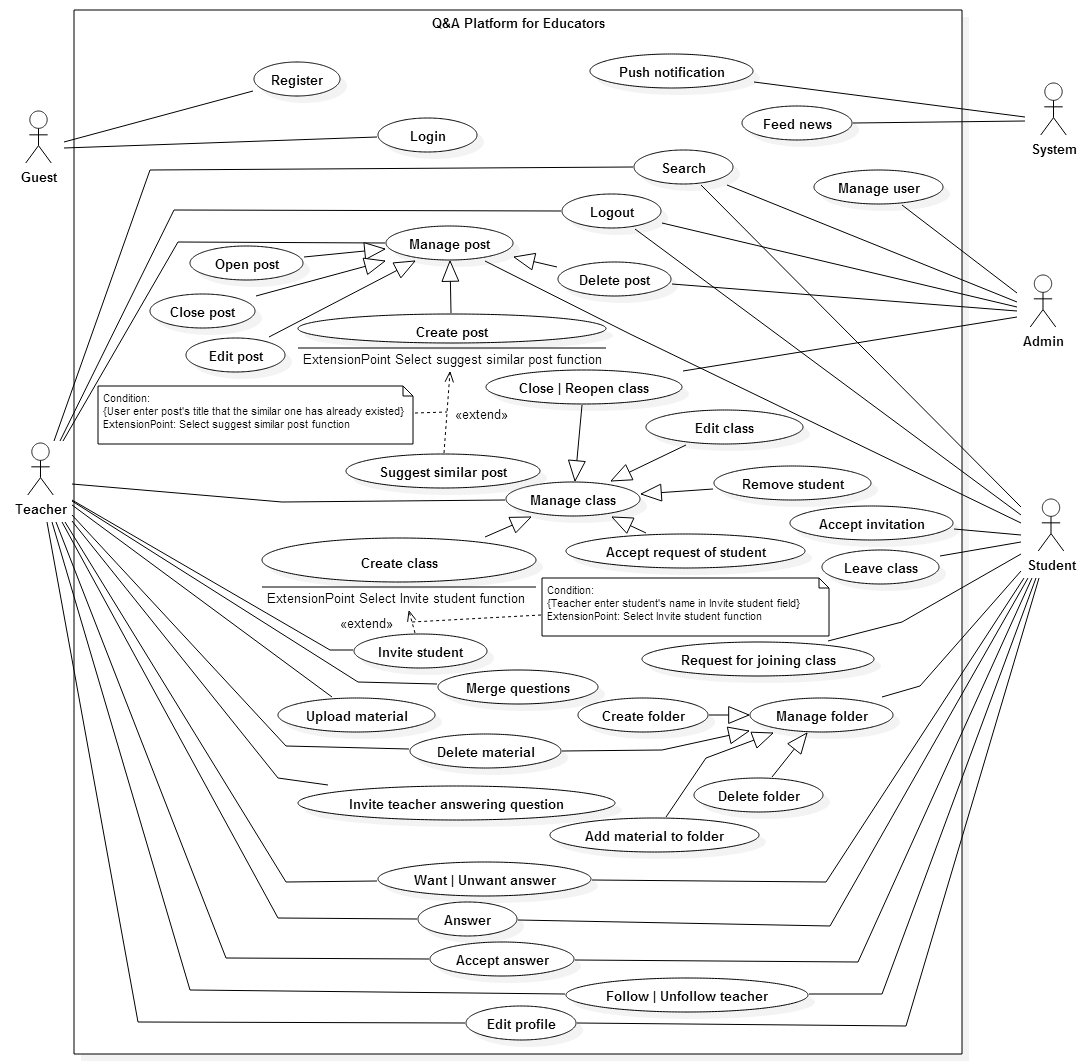
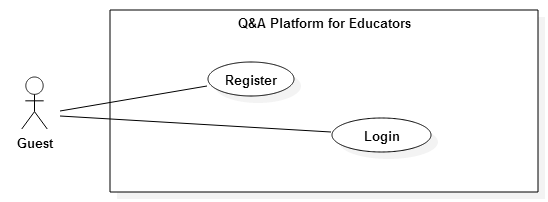


Figure 2: System Overview Use Case

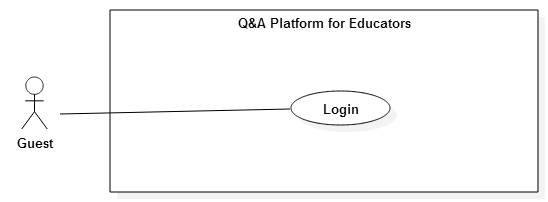
### List of Use Case

#### <Guest> Overview Use Case



**Figure 3: <Guest> Overview Use Case**

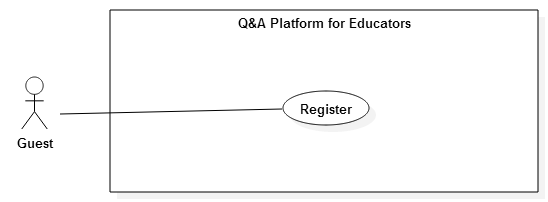
##### <Guest> Login Use Case Diagram

****

**Figure 4: <Guest> Login Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -1 SPECIFICATION** | | | |
| **Use-case No.** | QAPE001 | **Use-case Version** | 2.0 |
| **Use-case Name** | Login | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | High |
| **Actor:**   * Guest   **Summary:**   * This use case allows guest login to the system   **Goal:**   * Allows guest login to the system   **Triggers**   * Guest clicks Login button on Welcome screen.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: Guest logged in to the system. * **On Failure**: Show Login screen with error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 |  | System displays Welcome screen:  - “Email”: textbox, min length 8, max length 255.  - “Password”: textbox, min length: 8, max length 50.  -“Login”: button. | | 2 | Guest inputs email and password |  | | 3 | Click “Login” button | System navigates to News feed screen.  [Exceptions 1, 2, 3] |   **Alternative Scenario:**   * N/A   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Guest leaves “Email” or “Password” textboxes blank. | Show login screen with message: “Invalid email or password”. | | 2 | Guest logins with wrong username or password. | Show login screen with message: “Invalid email or password”. | | 3 | Guest logins with non-existed account. | Show login screen with message: “Invalid email or password”. |   **Relationships:**   * N/A   **Business Rules:**   * Guest have to login with Student account, Teacher account or Admin account to access the web application. | | | |

##### <Guest> Register Use Case Diagram



**Figure 5: <Guest> Register Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -2 SPECIFICATION** | | | |
| **Use-case No.** | QAPE002 | **Use-case Version** | 2.0 |
| **Use-case Name** | Register | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | High |
| **Actor:**   * Guest   **Summary:**   * This use case allows guest register an account   **Goal:**   * Allows guest login register an account   **Triggers**   * Guest clicks Register button on Welcome screen.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: New account is created and guest logged in to the system. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 |  | System displays Welcome screen:  - “Email\*”: textbox, min length 8, max length 255, regular expression: “^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$" required.  - “Password\*”: textbox, min length: 8, max length 50, required.  - “Confirm password\*”: textbox, min length: 8, max length 50, required.  - “Specialty\*”: drop down list, required.  - “You are\*”: select radio button, required.  -“Register”: button. | | 2 | Guest inputs all required fields |  | | 3 | Click “Register” button | New account is created and system navigates to News feed screen.  [Exceptions 1, 2, 3, 4, 5]  [Alternative 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Guest enters existed email address and clicks Register button | Show message: “This email has been used”. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Guest leaves “Email\*” blank or enters one not matched regular expression | Show error message: “Please enter a valid email address.” | | 2 | Guest leaves “Password\*” or “Confirm password\*” blank | Show error message: “Please provide a password. | | 3 | Guest enters “Password\*” or “Confirm password\*” is not matched min and max length | Show error message: “Your password must be between 8 and 50 characters long.” | | 4 | Guest enters “Password\*” and “Confirm password\*” are not matched | Show error message: “Please enter the same password as above.” | | 5 | Guest leaves “Specialty\*” as default | Show error message: “Please choose your specialty.” |   **Relationships:**   * N/A   **Business Rules:**   * All required fields must be filled. * Email address must be unique. * Password and confirm password must be same. | | | |

#### <Teacher> Overview Use Case

****

**Figure 6: <Teacher> Overview Use Case**

##### <Teacher> Create class Use Case Diagram

****

**Figure 7: <Teacher> Create class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -3 SPECIFICATION** | | | |
| **Use-case No.** | QAPE003 | **Use-case Version** | 2.0 |
| **Use-case Name** | Create class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher create the classroom and invite student to join classroom   **Goal:**   * Create new classroom   **Triggers**   * Teacher clicks “Create class” button in Dashboard screen.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: New classroom is created. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Dashboard screen and clicks “Create class” button | System displays Create classroom screen:  - “Class name\*”: textbox, min length 20, max length 255, required.  - “Tag\*”: textbox, required.  - “Invite student”: textbox.  - “Specialty\*”: drop down list, required.  - “Class’s Description\*”: text-area, , min length 20, max length 350, required.  - “Create class”: button. | | 2 | Teacher inputs all required fields |  | | 3 | Click “Create class” button | New classroom created and system navigates to classroom screen.  [Exceptions 1, 2, 3, 4, 5, 6] |   **Alternative Scenario:**   * N/A   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Class name\*” blank | Show error message: “Please provide the class's name.” | | 2 | Teacher enters “Class name\*” is not match min and max length | Show error message: “The class name must be between 20 and 255 characters long.” | | 3 | Teacher leaves “Tag\*” blank | Show error message: “Please provide at least one tag.” | | 4 | Teacher leaves “Specialty\*” as default | Show error message: “Please select classroom’s specialty.” | | 5 | Teacher leaves “Class’s description\*” blank | Show error message: “Please provide the description.” | | 6 | Teacher enters “Class’s description\*” is not match min and max length | Show error message: “The description must be between 20 and 255 characters long.” |   **Relationships:**   * Extend for Invite student (Select invite student function)   **Business Rules:**   * All required fields must be filled. * Number of tag is not over 5. * Duplicated tags in Tag or students in Invite student do not allow. * “Student” field can be blank. | | | |

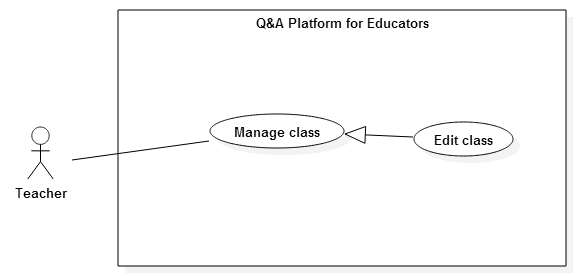
##### <Teacher> Invite student Use Case Diagram

****

**Figure 8: <Teacher> Invite student Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -4 SPECIFICATION** | | | |
| **Use-case No.** | QAPE004 | **Use-case Version** | 2.0 |
| **Use-case Name** | Invite student | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher invite students to join classroom   **Goal:**   * Invite students to join classroom   **Triggers**   * Teacher clicks “Invite” button in Classroom screen. * Teacher enters student’s name in Invite student when Create classroom.   **Preconditions:**   * In case of invite student in Classroom screen, classroom must be already created, in “open” state and owned by teacher.   **Post Conditions:**   * **On Success**: Students are invited received invitation. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks “Invite” button | System displays popup “Invite student to join class”:  - “Student\*”: textbox, required  - “Invite”: button.  - “x” icon: button.  [Alternative 1] | | 2 | User inputs student’s name | System show drop down list with suggest students base on student’s name | | 3 | User chooses student from drop down list |  | | 4 | Click “Invite” button | Popup is closed and invited students receive invitations.  [Alternative 2]  [Exceptions 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User clicks “x” icon | Popup is closed and no student is invited | | 2 | User types in text box instead of chooses from dropdown list | Popup is closed and no student is invited |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User leaves “Student\*” blank | Show error message: “Please provide at least one student.” |   **Relationships:**   * Extend by Create class   **Business Rules:**   * Teacher must choose student from suggest drop down list instead of types text. * If student has already joined in classroom or sent request to join classroom, their name must not be displayed in suggest list. * Suggested students must be active accounts. | | | |

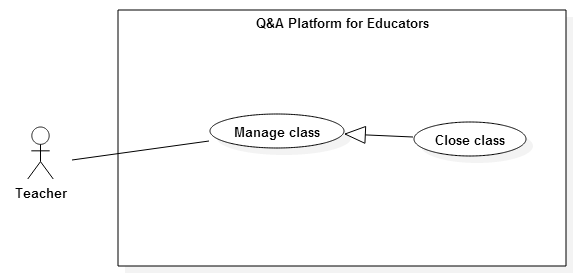
##### <Teacher> Edit class Use Case Diagram

****

**Figure 9: <Teacher> Edit class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -5 SPECIFICATION** | | | |
| **Use-case No.** | QAPE005 | **Use-case Version** | 2.0 |
| **Use-case Name** | Edit class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher update owned classroom’s information   **Goal:**   * Update classroom’s information   **Triggers**   * User clicks setting icon and clicks “Update class” in Classroom screen.   **Preconditions:**   * Classroom must be in “open” state and owned by teacher.   **Post Conditions:**   * **On Success**: Classroom’s information is updated * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks setting icon | System displays drop down list | | 2 | Teacher clicks “Update class”  [Alternative 1] | System displays Update classroom screen:  - “Tag\*”: textbox, required.  - “Specialty\*”: drop down list, required.  - “Class’s Description\*”: text-area, , min length 20, max length 350, required.  - “Update class”: button. | | 3 | Teacher updates classroom’s information |  | | 4 | Click “Update class” button | Classroom’s information is updated and system navigates to classroom screen.  [Exceptions 1, 2, 3, 4] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks different place in page or clicks setting icon again | Drop down list is hidden |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Tag\*” blank | Show error message: “Please provide at least one tag.” | | 2 | Teacher leaves “Specialty\*” as “Select classroom’s specialty” | Show error message: “Please select classroom’s specialty.” | | 3 | Teacher leaves “Class’s description\*” blank | Show error message: “Please provide the description.” | | 4 | Teacher enters “Class’s description\*” is not match min and max length | Show error message: “The description must be between 20 and 255 characters long.” |   **Relationships:**   * N/A   **Business Rules:**   * All required fields must be filled. * Number of tag is not over 5. * Teacher cannot input duplicated tags. * Teacher cannot edit classroom’s name. | | | |

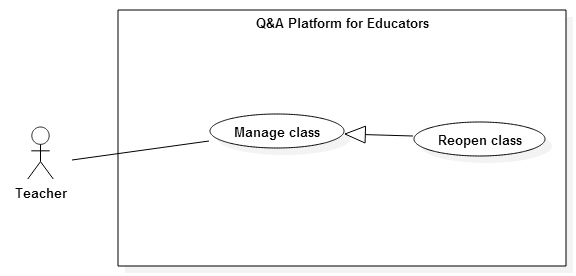
##### <Teacher> Close class Use Case Diagram

****

**Figure 10: <Teacher> Close class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -6 SPECIFICATION** | | | |
| **Use-case No.** | QAPE006 | **Use-case Version** | 2.0 |
| **Use-case Name** | Close class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher close owned classroom   **Goal:**   * Close the classroom to prevent more request, question, article or material   **Triggers**   * Teacher clicks setting icon and clicks “Close class” in Classroom screen.   **Preconditions:**   * Classroom must be in “open” state and owned by teacger.   **Post Conditions:**   * **On Success**: Classroom status is changed to close * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks setting icon | System displays drop down list | | 2 | Teacher clicks “Close class”  [Alternative 1] | Classroom status is changed to close  System displays Classroom screen with no “Create post” button and show “Classroom is closed” in “About classroom” widget |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks different place in page or clicks setting icon again | Drop down list is hidden. Nothing change |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher cannot create new article/question in closed classroom * Student cannot request to join closed classroom * Teacher cannot edit classroom information when it is closed * Teacher can manage question, article, material, request, and student in closed classroom except creating new question or article, uploading material and inviting student. | | | |

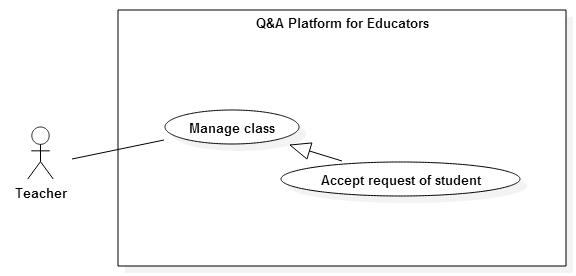
##### <Teacher> Reopen class Use Case Diagram

****

**Figure 11: <Teacher> Reopen class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -7 SPECIFICATION** | | | |
| **Use-case No.** | QAPE007 | **Use-case Version** | 2.0 |
| **Use-case Name** | Reopen class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher reopen closed classroom   **Goal:**   * Reopen closed classroom   **Triggers**   * Teacher clicks setting icon and clicks “Reopen class” in Classroom screen.   **Preconditions:**   * Classroom must be in “close” state and owned by teacher.   **Post Conditions:**   * **On Success**: Classroom status is changed to open * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks setting icon | System displays drop down list | | 2 | Teacher clicks “Reopen class”  [Alternative 1] | System displays Classroom screen with full classroom management functions |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks different place in page or clicks setting icon again | Drop down list is hidden. Nothing change |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher has full management functions with owned opened classroom * All questions, articles, materials, students of classroom after opening is kept stable * Student can send request to join opened classroom * Joined students can create new question or article after classroom is opened | | | |

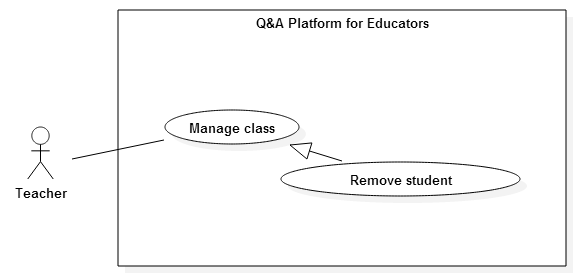
##### <Teacher> Accept request of student Use Case Diagram

****

**Figure 12: <Teacher> Accept request of student Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -8 SPECIFICATION** | | | |
| **Use-case No.** | QAPE008 | **Use-case Version** | 2.0 |
| **Use-case Name** | Accept request of student | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher accept joining requests of students   **Goal:**   * Accept students’ requests   **Triggers**   * Teacher clicks Confirm button on request.   **Preconditions:**   * Classroom must be owned by teacher and have at least one joining request from student.   **Post Conditions:**   * **On Success**: Request is removed and student is added to classroom * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to Classroom screen and clicks “Join Request” tab | System displays list of requests from students. | | 2 | User clicks “Confirm” button on request that user wants to accepts  [Alternative 1] | Message “You have confirmed the request of <student’s name>” is showed and this student is added to classroom. |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User clicks “Ignore” button | Message “You have ignored the request of <student’s name>” is showed and the request is removed |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher can confirm or ignore only with students’ requests to joined owned classrooms * Teacher cannot confirm the request after ignoring * Teacher can invite student after ignoring the request of this student | | | |

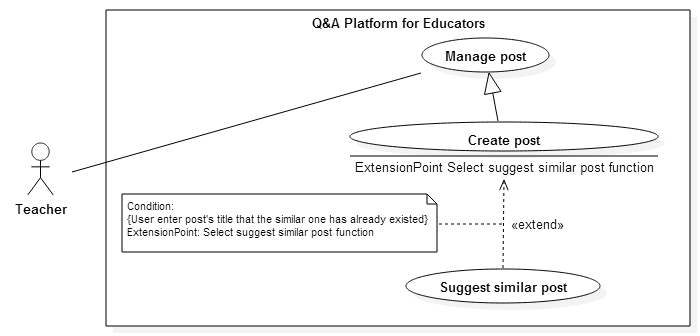
##### <Teacher> Remove student Use Case Diagram

****

**Figure 13: <Teacher> Remove student Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -9 SPECIFICATION** | | | |
| **Use-case No.** | QAPE009 | **Use-case Version** | 2.0 |
| **Use-case Name** | Remove student | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 15/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher remove students from classroom   **Goal:**   * Remove students   **Triggers**   * User clicks “Remove” button on student.   **Preconditions:**   * Classroom must be owned by teacher and have at least one student.   **Post Conditions:**   * **On Success**: Student is removed from classroom * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks tab “Student” | System displays list of students in classroom. | | 2 | Teacher clicks “Remove” button on student that user wants to remove | System displays “Remove student” popup:  - “OK”: button  - “Cancel”: button | | 3 | Teacher clicks “OK” button  [Alternative 1] | Message “You have remove <student’s name> from classroom.” is showed and student is removed from classroom |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks “Cancel” button | Popup disappears and no student is removed. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher can only remove students in owned classrooms. * Teacher cannot add students after removing them. * Teacher can invite students after removing them. * Removed student can sent request to join classroom again | | | |

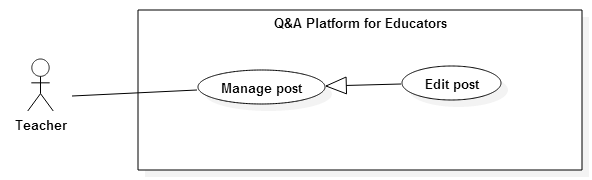
##### <Teacher> Create post Use Case Diagram

****

**Figure 14: <Teacher> Create post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 10 SPECIFICATION** | | | |
| **Use-case No.** | QAPE010 | **Use-case Version** | 2.0 |
| **Use-case Name** | Create post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 16/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher create new question or article in classroom.   **Goal:**   * Create new question or article   **Triggers**   * Teacher clicks “Create post” button in Classroom screen.   **Preconditions:**   * Teacher is owner of classroom * Classroom is in “open” state   **Post Conditions:**   * **On Success**: New question or article is created. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks “Create post” button | System displays “Create post” screen:  - “Title\*”: textbox, min length 20, max length 255, required.  - “Tag\*”: textbox, required.  - “Type\*”: drop down list, required.  - “Detail\*”: text-area, min length 120, required.  - “Publish your post”: button. | | 2 | Teacher inputs all required fields | After inputting “Title\*”, system suggest posts have similar titles  [Alternative 1] | | 3 | Click “Publish your post” button  [Alternative 2] | New article/question (base on Type) is created and system navigates to article/question screen.  [Exceptions 1, 2, 3, 4, 5, 6] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 |  | Nothing is suggested when there are no post have similar title | | 2 | Teacher clicks post’s title in suggested list. | System opens new tab in browser and show this post |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Title\*” blank | Show error message: “Please provide the title.” | | 2 | Teacher enters “Title\*” is not match min and max length | Show error message: “The class name must be between 20 and 255 characters long.” | | 3 | Teacher leaves “Tag\*” blank | Show error message: “Please provide at least one tag.” | | 4 | Teacher leaves “Type\*” as default | Show error message: “Please choose question or article.” | | 5 | Teacher leaves “Details\*” blank | Show error message: “Please provide the details.” | | 6 | Teacher enters “Details \*” is not match min length | Show error message: “The details must be at least 120 characters long.” |   **Relationships:**   * Extend for suggest similar post (Select suggest similar post function)   **Business Rules:**   * Post is only created if classroom is open. * Teacher can only create post in owned classroom where post will be created. * All required fields must be filled. * Number of tag is not over 5. * Teacher cannot input duplicated tags. | | | |

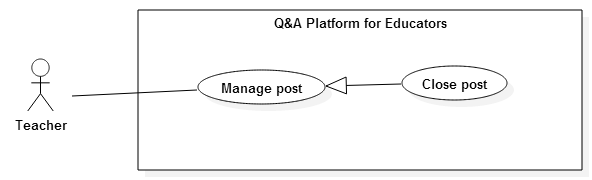
##### <Teacher> Edit post Use Case Diagram

****

**Figure 15: <Teacher> Edit post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 11 SPECIFICATION** | | | |
| **Use-case No.** | QAPE011 | **Use-case Version** | 2.0 |
| **Use-case Name** | Edit post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 16/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher edit owned question or article   **Goal:**   * Edit question or article   **Triggers**   * Teacher clicks setting icon and clicks “Edit” button in Question or Article screen.   **Preconditions:**   * Teacher is owner of question or article. * Article or question is in “Open” state.   **Post Conditions:**   * **On Success**: Question or Article is updated * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks “Create post” button | System displays Update post screen:  - “Title\*”: textbox, min length 20, max length 255, required.  - “Tag\*”: textbox, required.  - “Type\*”: drop down list, required.  - “Detail\*”: text-area, min length 120, required.  - “Save”: button. | | 2 | Teacher changes fields |  | | 3 | Click “Save” button | Question/Article is updated and system navigates to question/article screen  [Exceptions 1, 2, 3, 4, 5, 6] |   **Alternative Scenario:**   * N/A   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Title\*” blank | Show error message: “Please provide the title.” | | 2 | Teacher enters “Title\*” is not match min and max length | Show error message: “The class name must be between 20 and 255 characters long.” | | 3 | Teacher leaves “Tag\*” blank | Show error message: “Please provide at least one tag.” | | 4 | Teacher leaves “Type\*” as default | Show error message: “Please choose question or article.” | | 5 | Teacher leaves “Details\*” blank | Show error message: “Please provide the details.” | | 6 | Teacher enters “Details \*” is not match min length | Show error message: “The details must be at least 120 characters long.” |   **Relationships:**   * N/A   **Business Rules:**   * Teacher can only edit owned post. Teacher can edit student’s post (particularly student’s question) when teacher merges questions. * All required fields must be filled. * Number of tag is not over 5. * User cannot input duplicated tags. | | | |

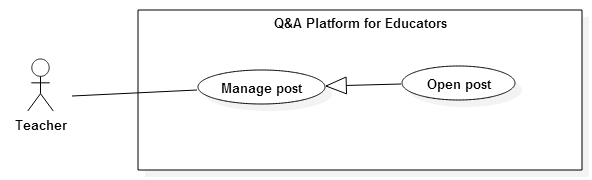
##### <Teacher> Close post Use Case Diagram

****

**Figure 16: <Teacher> Close post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 12 SPECIFICATION** | | | |
| **Use-case No.** | QAPE012 | **Use-case Version** | 2.0 |
| **Use-case Name** | Close post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 16/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher close post (question or article)   **Goal:**   * Close question or article to prevent more answers   **Triggers**   * Teacher clicks setting icon and clicks “Close” in Question or Article screen.   **Preconditions:**   * Question or article is in user’s owned classroom. * Article or question is in “Open” state.   **Post Conditions:**   * **On Success**: Question or Article is closed * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Question/Article screen and clicks setting icon. | System displays drop down list. | | 2 | Teacher clicks “Close”.  [Alternative 1] | System displays question/article screen with no “Leave a reply” area.  [Alternative 2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks different place in page or clicks setting icon again. | Drop down list is hidden. Nothing is changed. | | 2 |  | If post is question, “Ask teacher” widget will be hidden too. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher and student cannot leave answer in closed post or edit closed post. * Teacher cannot invite other teacher to closed question. * Teacher can delete closed owned post. * Teacher can only close posts in owned classroom | | | |

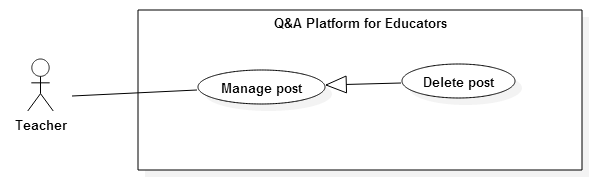
##### <Teacher> Open post Use Case Diagram

****

**Figure 17: <Teacher> Open post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 13 SPECIFICATION** | | | |
| **Use-case No.** | QAPE013 | **Use-case Version** | 2.0 |
| **Use-case Name** | Open post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 16/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows user open question or article   **Goal:**   * Open question or article   **Triggers**   * User clicks setting icon and clicks “Open” in Question or Article screen.   **Preconditions:**   * Question or article is in user’s owned classroom. * Article or question in “Close” state.   **Post Conditions:**   * **On Success**: Question or Article is opened * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to Question/Article screen and clicks setting icon. | System displays drop down list. | | 2 | User clicks “Open”.  [Alternative 1] | System displays question/article screen with “Leave a reply” area.  [Alternative 2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User clicks different place in page or clicks setting icon again. | Drop down list is hidden. Nothing is changed. | | 2 |  | If post is question, “Ask teacher” will be display too. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * User can reply or if post is question, teacher can also invite other teachers to answer it. * All answers or replies of post are kept stable. * Invited teacher can continue discussing in question. | | | |

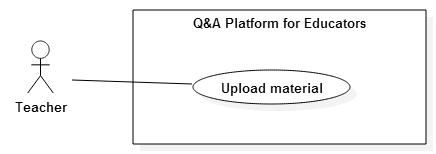
##### <Teacher> Delete post Use Case Diagram

****

**Figure 18: <Teacher> Delete post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 14 SPECIFICATION** | | | |
| **Use-case No.** | QAPE014 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 16/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher delete owned question or article   **Goal:**   * Delete question or article   **Triggers**   * Teacher clicks setting icon and clicks “Delete” button in Question or Article screen.   **Preconditions:**   * Teacher is owner of question or article.   **Post Conditions:**   * **On Success**: Question or Article is deleted * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Question/Article screen and clicks setting icon | System displays drop down list | | 2 | Teacher clicks “Delete” | System displays “Delete post” popup:  - “OK”: button  - “Cancel”: button | | 3 | Teacher clicks “OK” button  [Alternative 1] | Post is removed and system navigates to post’s owned classroom screen. |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks “Cancel” button | Popup disappears and no student is removed. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * All answers or replies of post are also deleted. * Deleting re-used merged question does not affect to other merged questions * Deleted post cannot restore * Teacher cannot delete other users’ posts. | | | |

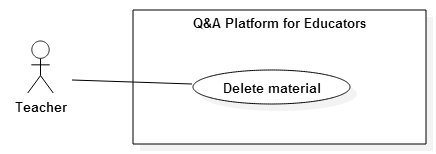
##### <Teacher> Upload material Use Case Diagram

****

**Figure 19: <Teacher> Upload material Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -15 SPECIFICATION** | | | |
| **Use-case No.** | QAPE015 | **Use-case Version** | 2.0 |
| **Use-case Name** | Upload material | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 16/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher upload material in owned classroom   **Goal:**   * Upload material   **Triggers**   * Teacher clicks Upload button in Material tab.   **Preconditions:**   * Classroom must be in “open” state and owned by user.   **Post Conditions:**   * **On Success**: New material is uploaded * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks tab “Material” | System displays list of materials in classroom. | | 2 | Teacher clicks “Upload” button | System displays “Upload material” popup:  - “Tag\*”: textbox, required.  - “Material\*”: file input, required  - “x” icon: button  - “Choose File”: button  - “Upload”: button | | 3 | Teacher inputs “Tag\*”  [Alternative 1] |  | | 4 | Teacher clicks “Choose file” and select file from computer | System displays name of selected material. | | 5 | Teacher clicks “Upload” button | Material is uploaded and systems show it in material list  [Exception 1, 2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks “x” icon button | Popup disappears and no material is uploaded. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Tag\*” blank | Show error message: “Please provide at least one tag.” | | 2 | Teacher doesn’t choose material to upload | Show error message: “Please choose material.” |   **Relationships:**   * N/A   **Business Rules:**   * Teacher can only upload material in owned opened classrooms. * Teacher can only upload one material at time. * If teacher upload material have same name with existed material in classroom, the existed material is updated with the new one. * Number of tag is not over 5. * Duplicated tags in Tag do not allow. | | | |

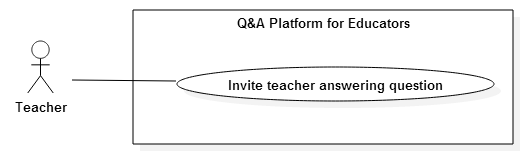
##### <Teacher> Delete material Use Case Diagram

****

**Figure 20: <Teacher> Delete material Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -16 SPECIFICATION** | | | |
| **Use-case No.** | QAPE016 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete material | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 15/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher delete material in owned classroom   **Goal:**   * Delete materials   **Triggers**   * Teacher clicks “Delete” button in Material tab.   **Preconditions:**   * Classroom must be owned by teacher and have at least one material.   **Post Conditions:**   * **On Success**: Material is deleted * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks tab “Material” | System displays list of materials. | | 2 | Teacher clicks “Delete” button | System displays “Delete material” popup:  - “OK”: button  - “Cancel”: button | | 3 | Teacher clicks “OK” button  [Alternative 1] | Material is deleted and disappears in material list |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks “Cancel” button | Popup disappears and no material is deleted. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher can only delete materials in owned classrooms. * Deleted material cannot restore * Student cannot download material that has been added to folder if this material is deleted in classroom | | | |

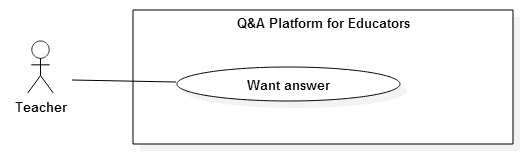
##### <Teacher> Invite teacher answering question Use Case Diagram

****

**Figure 21: <Teacher> Invite teacher answering question Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -17 SPECIFICATION** | | | |
| **Use-case No.** | QAPE017 | **Use-case Version** | 2.0 |
| **Use-case Name** | Invite teacher answering question | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 15/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher invite another teachers to join into discussion in question   **Goal:**   * Invite teacher to answering question   **Triggers**   * Teacher clicks “Ask teacher” button in Question screen.   **Preconditions:**   * Question must be in “open” state and in teacher’s owned classroom.   **Post Conditions:**   * **On Success**: Teachers are invited received invitation and can discuss in question. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Question screen and clicks “Ask teacher” button | System displays popup “Ask teacher student to join class”:  - “Teacher\*”: textbox, required  - “Invite”: button.  - “x” icon: button.  [Alternative 1] | | 2 | Teacher inputs student’s name | System show drop down list with suggest teachers base on teacher’s name | | 3 | Teacher chooses another teacher from drop down list |  | | 4 | Click “Invite” button | Popup is closed, invited teachers receive invitations and they can answer this question.  [Alternative 2]  [Exceptions 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks “x” icon | Popup is closed and no teacher is invited | | 2 | Teacher types in text box instead of chooses from dropdown list | Popup is closed and no teacher is invited |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Teacher\*” blank | Show error message: “Please provide at least one teacher.” |   **Relationships:**   * N/A   **Business Rules:**   * User must choose teacher from suggest drop down list instead of types text * Suggested teachers must be active accounts * User can invite other teachers only in opened question * Invited teacher can be invited again | | | |

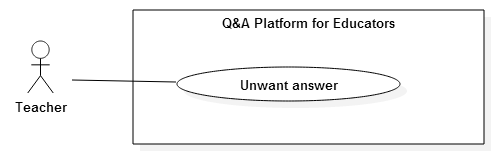
##### <Teacher> Want answer Use Case Diagram

****

**Figure 22: <Teacher> Want answer Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -18 SPECIFICATION** | | | |
| **Use-case No.** | QAPE018 | **Use-case Version** | 2.0 |
| **Use-case Name** | Want answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher want answer to increase the interest of question   **Goal:**   * Want answer to increase the interest of question   **Triggers**   * Teacher checks in un-checked “Want answer” checkbox   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: Number of “Want answer” increases 1 * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to Question screen and checks in “Want answer(s)” check box. | The number of “Want answer” increases 1 and checkbox is checked |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Closed question allows teacher checks “Want answer” * With each question, teacher can only increase the number of “Want answer” 1 unit * Teacher can check “Want answer” with any question in any classroom | | | |

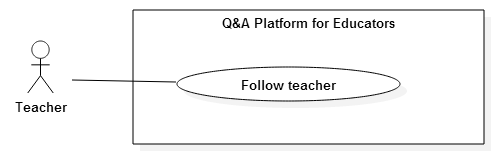
##### <Teacher> Un-want answer Use Case Diagram

****

**Figure 23: <Teacher> Un-want answer Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -19 SPECIFICATION** | | | |
| **Use-case No.** | QAPE019 | **Use-case Version** | 2.0 |
| **Use-case Name** | Un-want answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher un-want answer to decrease the interest of question   **Goal:**   * Un-want answer to increase the interest of question   **Triggers**   * Teacher unchecks in checked “Want answer” checkbox   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: Number of “Want answer” decreases 1 * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to Question screen and unchecks in “Want answer(s)” checkbox. | The number of “Want answer” decreases 1 and checkbox is un-checked |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Closed question allows uncheck “Want answer” * Each teacher can only uncheck “Want answer” if teacher has checked before * Teacher can only uncheck the checked checkbox. | | | |

##### <Teacher> Follow teacher Use Case Diagram

****

**Figure 24: <Teacher> Follow teacher Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -20 SPECIFICATION** | | | |
| **Use-case No.** | QAPE020 | **Use-case Version** | 2.0 |
| **Use-case Name** | Follow teacher | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher follow other teachers to get more interest question for news feed   **Goal:**   * Follow teacher   **Triggers**   * Teacher click “Follow” button in other teacher’s profile screen   **Preconditions:**   * Followed teacher must be activated account   **Post Conditions:**   * **On Success**: Teacher follows another one. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to another one’s Profile screen and clicks “Follow” button | Teacher follows another one and button “Follow” change to “Unfollow” |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Teacher cannot follow himself. * Only teachers can be followed. * Deactivated teacher accounts cannot be followed. * After following, news feed content changed earliest after 30 minutes | | | |

##### <Teacher> Un-follow teacher Use Case Diagram

****

**Figure 25: <Teacher> Un-follow teacher Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -21 SPECIFICATION** | | | |
| **Use-case No.** | QAPE021 | **Use-case Version** | 2.0 |
| **Use-case Name** | Un-follow teacher | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher un-follow other teachers to prevent get more news from them   **Goal:**   * Un-follow teacher   **Triggers**   * Teacher click “Unfollow” button in other teacher’s profile screen   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: Teacher un-follows another one. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to another one’s Profile screen and clicks “Unfollow” button  [Alternative 1] | Teacher un-follows another one and button “Unfollow” change to “Follow” |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher goes to dashboard screen, clicks on tab “Followed teacher”. | System show list of followed teachers | |  | Teacher clicks “Unfollow” button on teacher that user want to un-follow. | Message “You have unfollow <teacher’s name>” is showed and teacher un-follows another one. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Only followed teachers can be un-followed * Deactivated teacher accounts can be un-followed * Teacher can follow other teacher after un-following | | | |

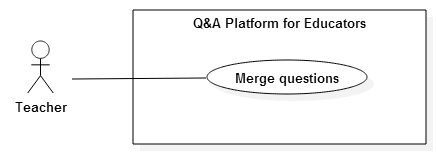
##### <Teacher> Edit profile Use Case Diagram

****

**Figure 26: <Teacher> Edit profile Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -22 SPECIFICATION** | | | |
| **Use-case No.** | QAPE022 | **Use-case Version** | 2.0 |
| **Use-case Name** | Edit profile | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher edit profile to update information   **Goal:**   * Edit profile   **Triggers**   * Teacher clicks “Edit profile” button on Profile screen.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: Profile is updated. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Profile screen and clicks “Edit profile” button | System displays “Edit profile” screen:  - “Profile Picture”: file input.  - “Display Name\*”: textbox, min length: 8, max length: 30, required  - “Password\*”: textbox, min length: 8, max length 50, required.  - “Confirm password\*”: textbox, min length: 8, max length 50, required.  - “Specialty\*”: drop down list, required.  - “Your tags”: textbox  - “About Yourself”: text-area  -“Update”: button. | | 2 | Teacher changes fields |  | | 3 | Click “Update” button | Profile is updated and system refreshes “Edit profile” screen with updated information  [Exception 1, 2, 3, 4, 5, 6] |   **Alternative Scenario:**   * N/A   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Display Name\*” blank | Show error message: “Please provide the display name.” | | 2 | Teacher enters “Display Name\*” not match min and max length | Show error message: “The display name must be between 8 and 30 characters long.” | | 3 | Teacher leaves “Password\*” or “Confirm password\*” blank | Show error message: “Please provide a password.” | | 4 | Teacher enters “Password\*” or “Confirm password\*” is not matched min and max length | Show error message: “Your password must be between 8 and 50 characters long.” | | 5 | Teacher enters “Password\*” and “Confirm password\*” are not matched | Show error message: “Please enter the same password as left.” | | 6 | Teacher leaves “Specialty\*” as default | Show error message: “Please choose your specialty.” |   **Relationships:**   * N/A   **Business Rules:**   * All required fields must be filled. * Password and confirm password must be same. * Teacher can only edit his profile. * After profile is updated, news feed content changed earliest after 30 minutes | | | |

##### <Teacher> Merge question Use Case Diagram

****

**Figure 27: <Teacher> Merge question Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -23 SPECIFICATION** | | | |
| **Use-case No.** | QAPE023 | **Use-case Version** | 2.0 |
| **Use-case Name** | Merge question | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher merge multiple questions into one.   **Goal:**   * Merge multiple questions into one in order to prevent similar question in classroom   **Triggers**   * Teacher clicks “Merge question” button on tab Question.   **Preconditions:**   * Teacher is owner of classroom. * Classroom has at least 2 questions.   **Post Conditions:**   * **On Success**: Main merged question is updated with new information. Other merge questions are closed * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Classroom screen and clicks tab Question | System displays list of questions in classroom | | 2 | Teacher clicks “Merge question”  [Alternative 1] | System navigates to “Merge questions” screen:  - List of similar questions with checkbox: required  - “Merge checked questions”: button  - “Answer checked questions”: text area, min length 30  - “Submit your answer”: button | | 3 | Teacher select questions and clicks “Merge checked questions”  [Alternative 2] | System navigates to “New merged question” screen:  - Reused question: radio button, required.  - “Title\*”: textbox, min length 20, max length 255, required.  - “Tag\*”: textbox, required.  - “Detail\*”: text-area, min length 120, required.  - “Save”: button. | | 4 | Teacher inputs all required field |  | | 5 | Clicks “Save” button | Reused question is saved with new information.  Other merged questions are closed and have commented “This question was merged and answered by <teacher’s name>; you can read it in this Question”.  System navigates to “Question” screen with new information of reused question.  [Exception 1, 2, 3, 4, 5, 6, 7, 9,10] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher clicks “<Number> similar question(s)” on question that user want to merge | [Main success 3] | | 2 | Teacher inputs “Answer checked question” text-area and click “Save” button  [Exception 9, 10] | All checked questions have answered with same answer that user has inputted. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves “Title\*” blank | Show error message: “Please provide the title.” | | 2 | Teacher enters “Title\*” is not match min and max length | Show error message: “The class name must be between 20 and 255 characters long.” | | 3 | Teacher leaves “Tag\*” blank | Show error message: “Please provide at least one tag.” | | 4 | Teacher leaves “Details\*” blank | Show error message: “Please provide the details.” | | 5 | Teacher enters “Details \*” is not match min length | Show error message: “The details must be at least 120 characters long.” | | 6 | Teacher doesn’t select any question | Show error message: “Please select at least two questions.” | | 7 | Teacher selects only one question | Show error message: “Please select at least two questions.” | | 8 | Teacher leaves “Answer checked question” text-area blank | Show error message: “Please provide the answer.” | | 9 | Teacher enters Answer checked question” text-area not match min length | Show error message: “The answer must be at least 30 characters long.” | | 10 | Teacher doesn’t select any question | Show error message: “Please select at least one question.” |   **Relationships:**   * Edit post   **Business Rules:**   * All required fields must be filled * User can only merge question in owned classroom * Merged questions that have not been re-used must be closed. * Resolved questions and closed questions cannot be merged * Students must receive notification if their questions are merged | | | |

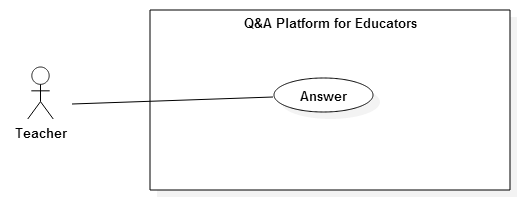
##### <Teacher> Search Use Case Diagram

****

**Figure 28: <Teacher> Search Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -24 SPECIFICATION** | | | |
| **Use-case No.** | QAPE024 | **Use-case Version** | 2.0 |
| **Use-case Name** | Search | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher search question, article, material, classroom, teacher and student.   **Goal:**   * Search question, material, classroom, teacher, student in system   **Triggers**   * Teacher clicks “Search” button on News feed screen.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: Results of searching is showed. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to News feed screen  [Alternative 1] | System displays “News feed” screen with search area:  - Search filter: dropdown list  - Search textbox  - “Search”: button | | 2 | Teacher chooses filter and inputs search key |  | | 3 | Click “Search” button  [Alternative 2] | System navigates to Search result screen and displays search results |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher goes to Classroom screen | System display “Classroom” screen with search textbox | |  | Teacher enters search key and presses “Enter” key | System display search results in this classroom | | 2 | Press “Enter” key | System navigates to Search result screen and displays search results |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Search results also are closed classrooms, questions, article; deactivated teacher, student; | | | |

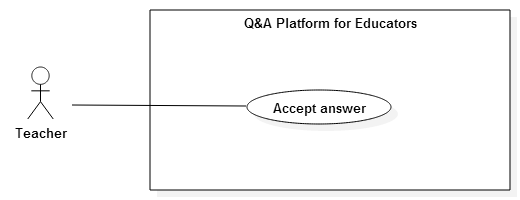
##### <Teacher> Answer Use Case Diagram

****

**Figure 29: <Teacher> Answer Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -25 SPECIFICATION** | | | |
| **Use-case No.** | QAPE025 | **Use-case Version** | 2.0 |
| **Use-case Name** | Answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows teacher answer question   **Goal:**   * Leave an answer for question   **Triggers**   * Teacher clicks “Submit your answer” button.   **Preconditions:**   * Question is in “open” state   **Post Conditions:**   * **On Success**: Answer is showed at real time. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Teacher goes to Question screen | System displays “Question” screen with:  - Leave an answer: text area, required, min length: 30  - “Submit your answer”: button | | 2 | Teacher inputs answer and clicks “Submit your answer” | System displays inputted answer at real time  [Exception 1] |   **Alternative Scenario:**   * N/A   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Teacher leaves answer blank | Show error message: “The answer must be at least 30 characters long.” | | 2 | Teacher leaves answer not match min length | Show error message: “The answer must be at least 30 characters long.” |   **Relationships:**   * N/A   **Business Rules:**   * An answer is submitted only if post is in “open” state * An answer is shown in real time to users that are visiting the question * Newest answer is on top of the answer list | | | |

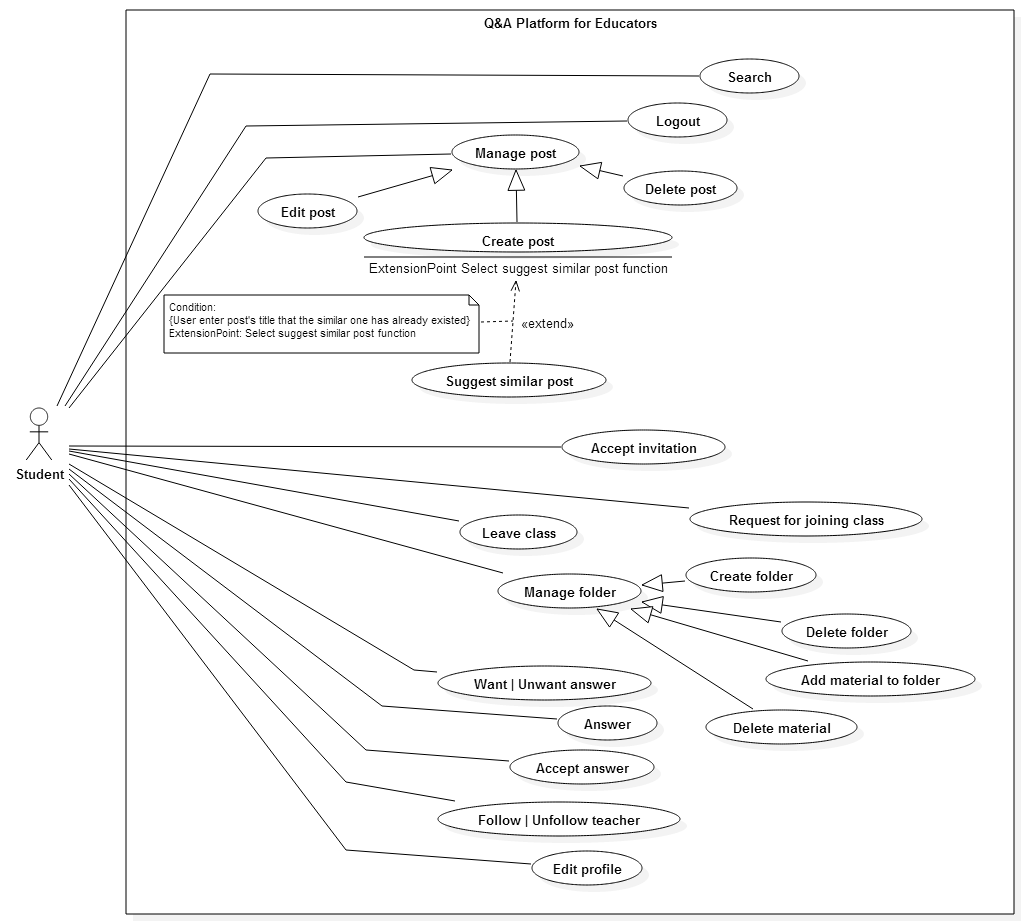
##### <Teacher> Accept answer Use Case Diagram

****

**Figure 30: <Teacher> Accept answer Use Case Diagram**

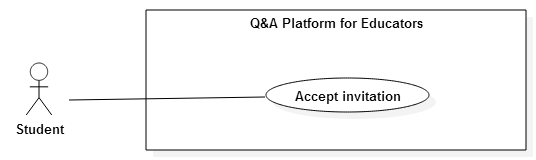
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -26 SPECIFICATION** | | | |
| **Use-case No.** | QAPE026 | **Use-case Version** | 2.0 |
| **Use-case Name** | Accept answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Teacher   **Summary:**   * This use case allows user accept an answer in user’s owned question   **Goal:**   * Answer in user’s owned question is accepted   **Triggers**   * User clicks “Accept” button.   **Preconditions:**   * Question is in “open” state * User is owner of the question. * Question has at least one answer from another user   **Post Conditions:**   * **On Success**: Question is resolved. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User goes to Question screen and click “Accept” button | “Accept” button is changed to “Unaccept” button.  [Alternative 1] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 |  | If accepted answer’s already existed, “Unaccept” button of it is auto changed to “Accept” button |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Only owner of the question can accept answer * Only one answer is accepted * If accepted answer is existed, the owner of the question can accept another answer. The older accepted answer is un-accepted. * The owner of the question cannot accept his owned answers in this question. | | | |

#### <Student> Overview Use Case

****

**Figure 31: <Student> Overview Use Case**

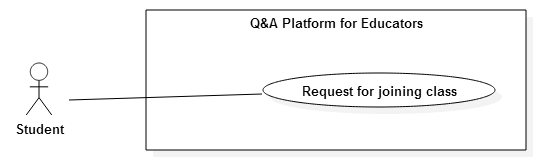
##### <Student> Accept Invitation Use Case Diagram

****

**Figure 32: <Student> Accept Invitation Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -27 SPECIFICATION** | | | |
| **Use-case No.** | QAPE027 | **Use-case Version** | 2.0 |
| **Use-case Name** | Accept Invitation | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Student   **Summary:**   * This use case allows student accept invitation from teacher to join classroom   **Goal:**   * Accept invitation to join classroom   **Triggers**   * Student clicks “Confirm” button on invitation.   **Preconditions:**   * Student has at least one invitation.   **Post Conditions:**   * **On Success**: Student joined in classroom. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Dashboard screen and clicks “Invitation” tab  [Alternative 1] | System displays list of invitations from teachers | | 2 | Student clicks “Confirm” button on invitation that student wants to accept  [Alternative 2] | Message “You have joined <classroom’s name>” is showed and student is added to this classroom. |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Student goes to Classroom screen that is invited and clicks “Accept invitation” button | “Accept invitation” button is hidden  “Leave” button is showed  “Create post” button is showed | | 2 | Student clicks “Ignore” button | Message “You have ignored the invitation to join <classroom’s name>” is showed and the invitation is removed |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Student cannot confirm the invitation after ignoring * Student cannot accept invitation in closed classroom * Student can be invited again to join the classroom that student has ignored * After confirming invitation, news feed content changed earliest after 30 minutes * Student can only create post in classroom after joining classroom (by accepting teacher’s invitation or after teacher accepts student’s request) | | | |

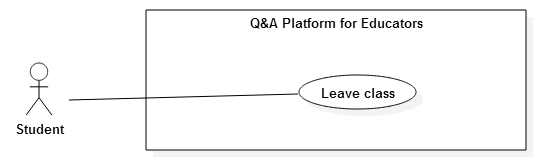
##### <Student> Request for joining class Use Case Diagram

****

**Figure 33: <Student> Request for joining class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -28 SPECIFICATION** | | | |
| **Use-case No.** | QAPE028 | **Use-case Version** | 2.0 |
| **Use-case Name** | Request for joining class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Student   **Summary:**   * This use case allows student request teacher for joining favorite classroom   **Goal:**   * Request for joining classroom   **Triggers**   * Student clicks “Join” button on Classroom screen.   **Preconditions:**   * Student hasn’t joined or received invitation   **Post Conditions:**   * **On Success**: Request is sent to teacher. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Classroom screen and clicks “Join” button | The request for joining classroom is sent to teacher  “Join” button is hidden  “Request sent” button is show |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Student can only request for joining classroom if student has not yet invited by teacher or has not yet joined. * Student cannot request to join closed classroom | | | |

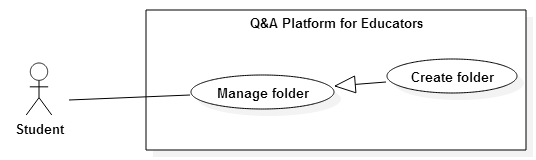
##### <Student> Leave class Use Case Diagram

****

**Figure 34: <Student> Leave class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -29 SPECIFICATION** | | | |
| **Use-case No.** | QAPE029 | **Use-case Version** | 2.0 |
| **Use-case Name** | Leave class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **Actor:**   * Student   **Summary:**   * This use case allows student leave joined classroom   **Goal:**   * Leave joined classroom   **Triggers**   * Student clicks “Leave” button in Classroom screen.   **Preconditions:**   * Student has joined at least one classroom   **Post Conditions:**   * **On Success**: Request is sent to teacher. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Classroom screen and clicks “Leave” button  [Alternative 1] | System displays “Leave classroom” popup:  - “OK”: button  - “Cancel”: button | | 2 | Student clicks “OK” button  [Alternative 2] | “Create post” button is hidden  “Leave” button is hidden  “Join” button is showed |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Student goes to Dashboard screen and clicks “Joined classroom” tab | System displays list of joined classrooms | |  | Student clicks “Leave” button on classroom that user wants to leave | System displays “Leave classroom” popup:  - “OK”: button  - “Cancel”: button | |  | Student clicks “OK” button  [Alternative 2] | “Create post” button is hidden  “Leave” button is hidden  “Join” button is showed | | 2 | Student clicks “Cancel” button | Popup disappears and student has not left classroom. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Student can only leave joined classroom (if student is invited to join or student sends request for joining, student cannot leave classroom) * After leaving, student can request for joining or owner of classroom can invite student to join again. * After leaving, news feed content changed earliest after 30 minutes | | | |

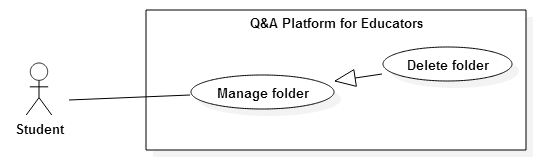
##### <Student> Create folder Use Case Diagram

****

**Figure 35: <Student> Create folder Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -30 SPECIFICATION** | | | |
| **Use-case No.** | QAPE030 | **Use-case Version** | 2.0 |
| **Use-case Name** | Create folder | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Student   **Summary:**   * This use case allows student create folder to manage material   **Goal:**   * Create new folder   **Triggers**   * Student clicks “Create folder” button in Classroom screen.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: New folder is created. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Material screen and clicks “Create folder” button | System displays “Create folder” popup:  - “Folder name\*”: textbox, min length: 8, max length: 30, required  - “Create new folder”: button  - “x” icon: button | | 2 | Student fill “Folder name\*” |  | | 3 | Student click “Create new folder”  [Alternative 1] | Folder is created, popup is hidden and system displays new folder in “Library” tab |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Student click “x” icon button | Popup is hidden and no folder is created |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Student leaves “Folder name\*” blank | Show error message: “Please provide the folder name.” | | 2 | Student enters “Folder Name\*” not match min and max length | Show error message: “The folder name must be between 8 and 30 characters long.” |   **Relationships:**   * N/A   **Business Rules:**   * Student can create maximum 10 folders * Only students can create new folder * Newest folder is showed at first | | | |

##### <Student> Delete folder Use Case Diagram

****

**Figure 36: <Student> Delete folder Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -31 SPECIFICATION** | | | |
| **Use-case No.** | QAPE031 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete folder | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **Actor:**   * Student   **Summary:**   * This use case allows student delete folder   **Goal:**   * Delete owned folder   **Triggers**   * Student clicks “x” icon in folder icon.   **Preconditions:**   * Student must have at least one folder   **Post Conditions:**   * **On Success**: Folder is deleted. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Material screen and clicks “x” icon on folder icon | System displays “Delete folder” popup:  - “OK”: button  - “Cancel”: button | | 2 | Student clicks “OK” button  [Alternative 1] | Folder is deleted and removed from screen |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Student clicks “Cancel” button | Popup disappears and no student is removed. |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Delete folder include deleting all materials in it | | | |

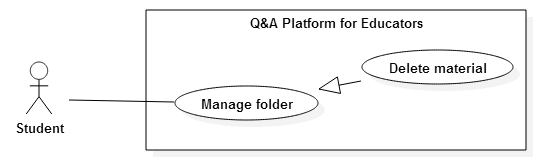
##### <Student> Add material to folder Use Case Diagram

****

**Figure 37: <Student> Add material to folder Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -32 SPECIFICATION** | | | |
| **Use-case No.** | QAPE032 | **Use-case Version** | 2.0 |
| **Use-case Name** | Add material to folder | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **Actor:**   * Student   **Summary:**   * This use case allows student add favorite material to folder   **Goal:**   * Add material to folder   **Triggers**   * Student clicks “Folder” button on material that student want to add.   **Preconditions:**   * User must have at least one folder.   **Post Conditions:**   * **On Success**: Material is added to folder. * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Classroom screen and clicks “Material” tab | System displays list of material of classroom | | 2 | Student clicks “Folder” button on material that student want to added | System displays popup “Add to folder”:  - List of created folder  - “x” icon button | | 3 | Student select folder  [Alternative 1] | Material is added to folder |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Student click “x” icon button | Popup is hidden and material is added to folder |   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * Student can add duplicate material to same folder * Student can add any material in any classroom to folder | | | |

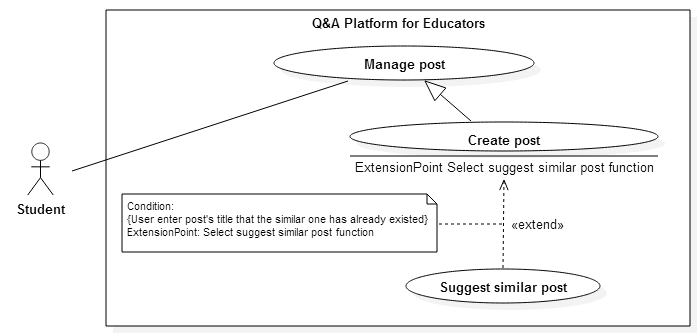
##### <Student> Delete material Use Case Diagram

****

**Figure 38: <Student> Delete material Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -33 SPECIFICATION** | | | |
| **Use-case No.** | QAPE033 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete material | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 16 (QAPE016) WITH DIFFERENT POINTS:**  **Actor:**   * Student   **Triggers**   * Student clicks “Delete” button in Folder screen.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Student goes to Folder screen. | System displays list of material | |  | **Other steps (begin from step 2) reference to QAPE016 USE CASE** | |   **Business Rules:**   * Deleted material is only deleted from folder. Material is still available in owned classroom. | | | |

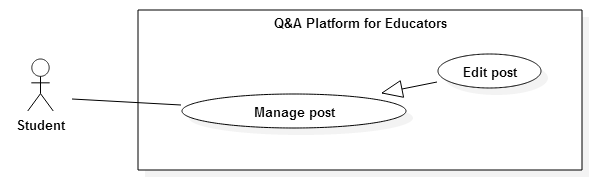
##### <Student> Create post Use Case Diagram

****

**Figure 39: <Student> Create post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 34 SPECIFICATION** | | | |
| **Use-case No.** | QAPE034 | **Use-case Version** | 2.0 |
| **Use-case Name** | Create post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **REFERENCE TO USE CASE – 10 (QAPE010) WITH DIFFERENT POINTS:**  **Actor:**   * Student   **Preconditions:**   * Student has joined classroom where post will be created * Classroom is in “open” state | | | |

##### <Student> Edit post Use Case Diagram

****

**Figure 40: <Student> Edit post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 35 SPECIFICATION** | | | |
| **Use-case No.** | QAPE035 | **Use-case Version** | 2.0 |
| **Use-case Name** | Edit post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 11 (QAPE011) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

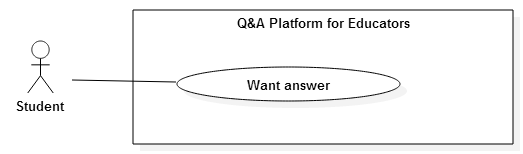
##### <Student> Delete post Use Case Diagram

****

**Figure 41: <Student> Delete post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 36 SPECIFICATION** | | | |
| **Use-case No.** | QAPE036 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 14 (QAPE014) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

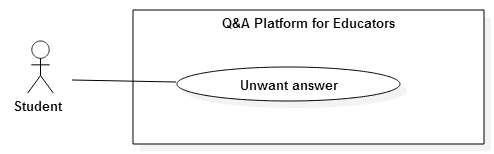
##### <Student> Want answer Use Case Diagram

****

**Figure 42: < Student> Want answer Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -37 SPECIFICATION** | | | |
| **Use-case No.** | QAPE037 | **Use-case Version** | 2.0 |
| **Use-case Name** | Want answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **REFERENCE TO USE CASE – 18 (QAPE018) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

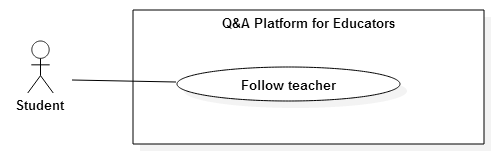
##### < Student> Un-want answer Use Case Diagram

****

**Figure 43: < Student> Un-want answer Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -38 SPECIFICATION** | | | |
| **Use-case No.** | QAPE038 | **Use-case Version** | 2.0 |
| **Use-case Name** | Un-want answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **REFERENCE TO USE CASE – 19 (QAPE019) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

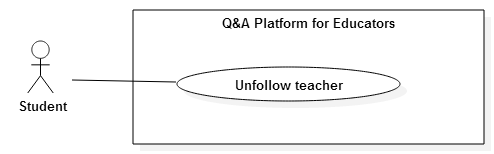
##### < Student> Follow teacher Use Case Diagram

****

**Figure 44: < Student> Follow teacher Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -39 SPECIFICATION** | | | |
| **Use-case No.** | QAPE039 | **Use-case Version** | 2.0 |
| **Use-case Name** | Follow teacher | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **REFERENCE TO USE CASE – 20 (QAPE020) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

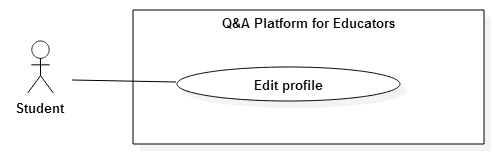
##### < Student> Un-follow teacher Use Case Diagram

****

**Figure 45: < Student> Un-follow teacher Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -40 SPECIFICATION** | | | |
| **Use-case No.** | QAPE040 | **Use-case Version** | 2.0 |
| **Use-case Name** | Un-follow teacher | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 21 (QAPE021) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

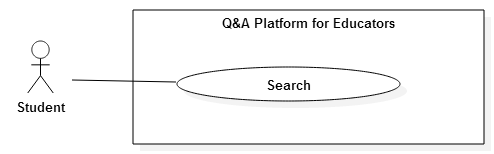
##### < Student> Edit profile Use Case Diagram

****

**Figure 46: <Student> Edit profile Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -41 SPECIFICATION** | | | |
| **Use-case No.** | QAPE041 | **Use-case Version** | 2.0 |
| **Use-case Name** | Edit profile | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 22 (QAPE022) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

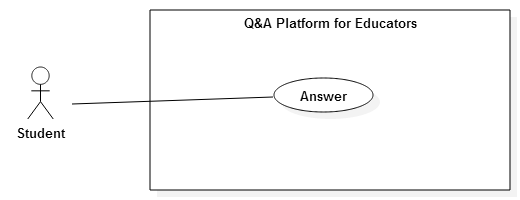
##### <Student> Search Use Case Diagram

****

**Figure 47: <Student> Search Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -42 SPECIFICATION** | | | |
| **Use-case No.** | QAPE042 | **Use-case Version** | 2.0 |
| **Use-case Name** | Search | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 24 (QAPE024) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

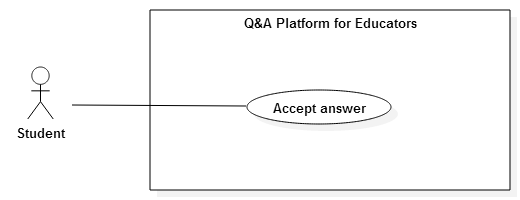
##### <Student> Answer Use Case Diagram

****

**Figure 48: <Student> Answer Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -43 SPECIFICATION** | | | |
| **Use-case No.** | QAPE043 | **Use-case Version** | 2.0 |
| **Use-case Name** | Answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **REFERENCE TO USE CASE – 25 (QAPE025) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

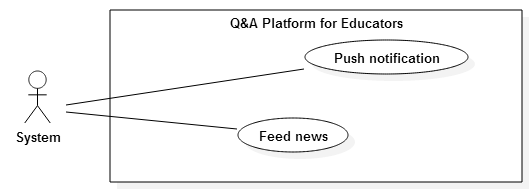
##### <Student> Accept answer Use Case Diagram

****

**Figure 49: <Student> Accept answer Use Case Diagram**

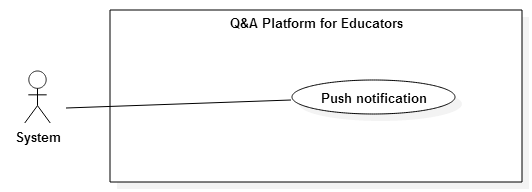
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -44 SPECIFICATION** | | | |
| **Use-case No.** | QAPE044 | **Use-case Version** | 2.0 |
| **Use-case Name** | Accept answer | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | High |
| **REFERENCE TO USE CASE – 26 (QAPE026) WITH DIFFERENT POINTS:**  **Actor:**   * Student | | | |

#### <System> Overview Use Case

****

**Figure 50: <System> Overview Use Case**

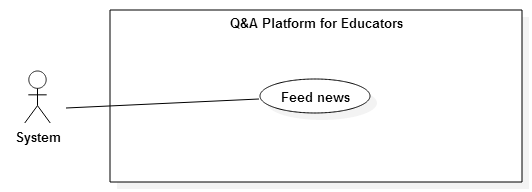
##### <System> Push Notification Use Case Diagram

****

**Figure 51: <System> Push Notification Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -45 SPECIFICATION** | | | |
| **Use-case No.** | QAPE045 | **Use-case Version** | 2.0 |
| **Use-case Name** | Push notification | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 18/07/2015 | **Priority** | High |
| **Actor:**   * System   **Summary:**   * This use case allows system send notification to related users   **Goal:**   * Notify followers when teacher creates classroom * Notify followers and student in classroom when teacher creates post in this classroom * Notify students in classroom and classroom’s owned teacher when student creates post in this classroom. * Notify question owner and all users discuss in question when having new reply * Notify classroom’s owned teacher when having new request * Notify student and teacher when having new invitation   **Triggers**   * One of above events (in **Goal** above) occurs in real time   **Preconditions:**   * Teacher creates classroom * Teacher creates post in classroom * Student creates post in classroom * Teacher or student answers the question * Student requests to join classroom * Teacher invites student to join classroom * Teacher invites others to answering question   **Post Conditions:**   * **On Success**: Notify related user when event occurs * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | System find all users that relate to occurred event | System sends notification to related users. |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * After event occurs, notification will send to related user at the same time (real time notification) * Notification includes hyperlink that linked with occurred event. * User can only see top 10 recent notifications and all unread notifcations | | | |

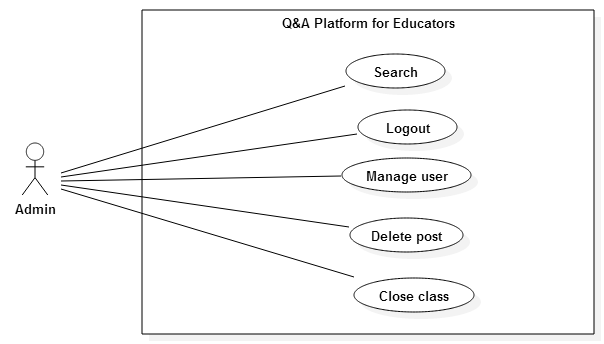
##### <System> Feed News Use Case Diagram

****

**Figure 52: <System> Feed News Use Case Diagram**

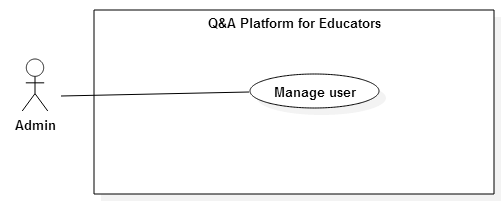
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -46 SPECIFICATION** | | | |
| **Use-case No.** | QAPE046 | **Use-case Version** | 2.0 |
| **Use-case Name** | Feed news | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 18/07/2015 | **Priority** | High |
| **Actor:**   * System   **Summary:**   * This use case allows system calculates scores of questions, articles, materials to order and show on user’s newsfeed for each user.   **Goal:**   * Calculate question score base on view number, want answer number, created date, updated date * Check question is in owned/joined classroom, is owned followed teacher, is in user’s specialty * Check article’s, material’s tags is in user tags * Check article, material is in user’s specialty * Sort questions, articles, materials decreasingly by score and add them to supported database   **Triggers**   * At 30 minutes routine.   **Preconditions:**   * User must be log in as Student or Teacher role   **Post Conditions:**   * **On Success**: Sorted questions, articles, materials for each user are added to supported database * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Server checks time and compares with previous feed news function’s executed times. If distance between two times is 30 minutes, calculate score and sort | System calculates score of questions, articles, materials, and sorted each of three by score decreasingly. |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * In 30 minutes between two execution times, user’s newsfeed has nothing changed. * If user has not owned or joined any classroom, scores of questions that are in classrooms will be passed * If user has not follow any teacher, scores of questions that are owner by followed teachers will be passed * If user has not yet enter his tags in profile, scores of articles and material that their tags match user’s tags will be passed. | | | |

#### <Admin> Overview Use Case

****

**Figure 53: <Admin> Overview Use Case**

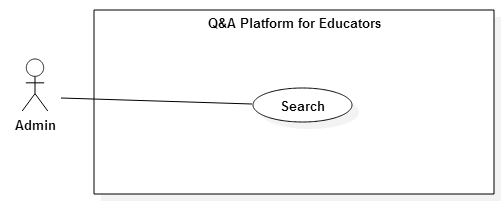
##### <Admin> Manage user Use Case Diagram

****

**Figure 54: <Admin> Manage user Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -47 SPECIFICATION** | | | |
| **Use-case No.** | QAPE047 | **Use-case Version** | 2.0 |
| **Use-case Name** | Manage user | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 18/07/2015 | **Priority** | Medium |
| **Actor:**   * Admin   **Summary:**   * This use case allows admin change users’ status in system   **Goal:**   * Activate/Deactivate user account   **Triggers**   * Admin clicks “Lock” icon / “Unlock” icon on user.   **Preconditions:**   * N/A   **Post Conditions:**   * **On Success**: User is activated or deactivated * **On Failure**: Show error message   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin goes to Manage user screen. | System displays list of users in system.  If user account in row is deactivated: show “Unlock” icon  If user account in row is activated: show “Lock” icon | | 2 | Admin clicks “Lock”/”Unlock” icon on user that admin wants to change status | User’s status is updated  “Lock” icon changes to “Unlock” icon in case of deactivating account  “Unlock” icon change to “Lock” icon in case of activating account |   **Alternative Scenario:**   * N/A   **Exceptions:**   * N/A   **Relationships:**   * N/A   **Business Rules:**   * User account has two status Activated/Deactivated * After account is deactivated, user cannot login to system with this account * Admin cannot activate/deactivate himself | | | |

##### <Admin> Search Use Case Diagram

****

**Figure 55: <Admin> Search Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE -48 SPECIFICATION** | | | |
| **Use-case No.** | QAPE048 | **Use-case Version** | 2.0 |
| **Use-case Name** | Search | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 24 (QAPE024) WITH DIFFERENT POINTS:**  **Actor:**   * Admin | | | |

##### <Admin> Delete post Use Case Diagram

****

**Figure 56: <Admin> Delete post Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 49 SPECIFICATION** | | | |
| **Use-case No.** | QAPE049 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete post | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 14 (QAPE014) WITH DIFFERENT POINTS:**  **Actor:**   * Admin   **Business Rules:**   * User with admin role can delete any posts. | | | |

##### <Admin> Close class Use Case Diagram

****

**Figure 57: <Admin> Close class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 50 SPECIFICATION** | | | |
| **Use-case No.** | QAPE050 | **Use-case Version** | 2.0 |
| **Use-case Name** | Close class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 6 (QAPE06) WITH DIFFERENT POINTS:**  **Actor:**   * Admin   **Preconditions:**   * Classroom must be in “open” state.   **Business Rules:**   * Admin can close any classrooms * Admin cannot create new article/question in closed classroom * Admin can manage question, article, material, request, and student in closed classroom. | | | |

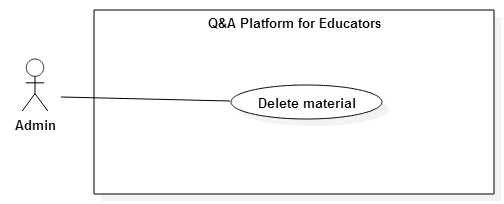
##### <Admin> Reopen class Use Case Diagram

****

**Figure 58: <Admin> Reopen class Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 51 SPECIFICATION** | | | |
| **Use-case No.** | QAPE051 | **Use-case Version** | 2.0 |
| **Use-case Name** | Reopen class | | |
| **Author** | Kha Hoang Minh | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 7 (QAPE07) WITH DIFFERENT POINTS:**  **Actor:**   * Admin   **Preconditions:**   * Classroom must be in “open” state.   **Business Rules:**   * Admin can reopen any classrooms. | | | |

##### <Admin> Delete material Use Case Diagram

****

**Figure 59: <Admin> Delete material Use Case Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE 52 SPECIFICATION** | | | |
| **Use-case No.** | QAPE052 | **Use-case Version** | 2.0 |
| **Use-case Name** | Delete material | | |
| **Author** | Truong Nhu Khang | | |
| **Date** | 17/07/2015 | **Priority** | Medium |
| **REFERENCE TO USE CASE – 16 (QAPE016) WITH DIFFERENT POINTS:**  **Actor:**   * Admin   **Business Rules:**   * Admin can delete materials in any classrooms. * Deleted material cannot restore * Student cannot download material that has been added to folder if this material is deleted in classroom | | | |

## Software System Attribute

### Usability

#### Graphic User Interface

* + All the texts, labels and buttons should be written in English.

#### Usability

* + Admin only need one day of training to use this system.
  + Other users can use system easily without technical training.

#### Installation

* + The system must be easy to deploy. Customer can deploy successfully after following all step in installation guide.

### Reliability

* + N/A

### Availability

* + The system is available 24/7.
  + The user need an internet connection to access the system.

### Security

* + Each user has specific role to interact with system in user’ permission.
  + The system checks authorization and authentication for each user’s action.

### Maintainability

* + The system is divided into separate modules.
  + The code is easy to extend or upgrade.

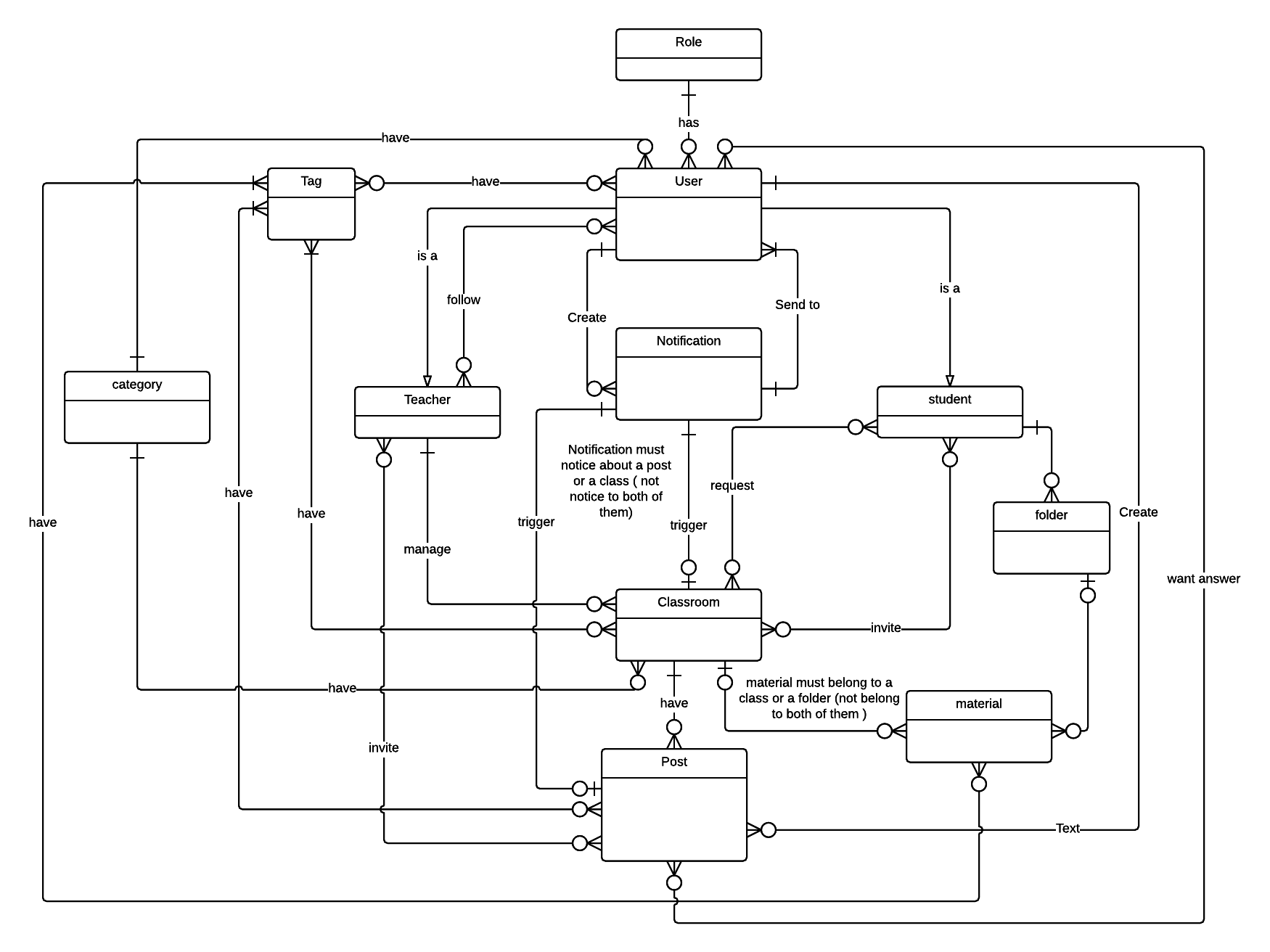
### Portability

* + The system is suitable with many servers that support Tomcat

### Performance

* + N/A

## Conceptual Diagram

****

**Figure 60: Conceptual diagram**

**Data Dictionary**

|  |  |
| --- | --- |
| **Entity Data dictionary: describe content of all entities** | |
| Entity Name | Description |
| User | Abstract describes a user in system |
| Tag | Contain the Tag information |
| Role | Contain the Role information |
| Category | Contain the Category information |
| Class | Contain the Class information |
| Post | Contain the Post information |
| Student | Is a user , contain the Student information |
| Teacher | Is a user , contain the Teacher information |
| Folder | Contain the Folder information |
| Material | Contain the Material information |
| Notification | Contain the Notification information |

Table 6: Entity data dictionary

# Report No. 4 Software Design Description

## Design Overview

* This document describes the technical and user interface design of QAPE System. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.
* The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.
* The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.
* The database design describes the relationships between entities and details of each entity.
* Document overview:
* Section 2: gives an overall description of the system architecture design.
* Section 3: gives component diagrams that describe the connection and integration of the system.
* Section 4: gives the detail design description, which includes class diagram, class explanation, and sequence diagram to details the application functions.
* Section 5: gives the design of user interface.
* Section 6: describe a fully attributed ERD.
* Section 7: gives the details of algorithms which are used.

## System architecture design

* Our system, QAPE is developed based on MVC pattern.
* QAPE is a big website and a kind of social website. MVC architecture separate a product (website, win form, phone app …) to three parts that help us easier to build – develop – test – control project.
* MVC architecture can run with any environment or language, so in the future if we want to another instance for our website on Mobile Phone or Desktop Computer, We can do it easily.

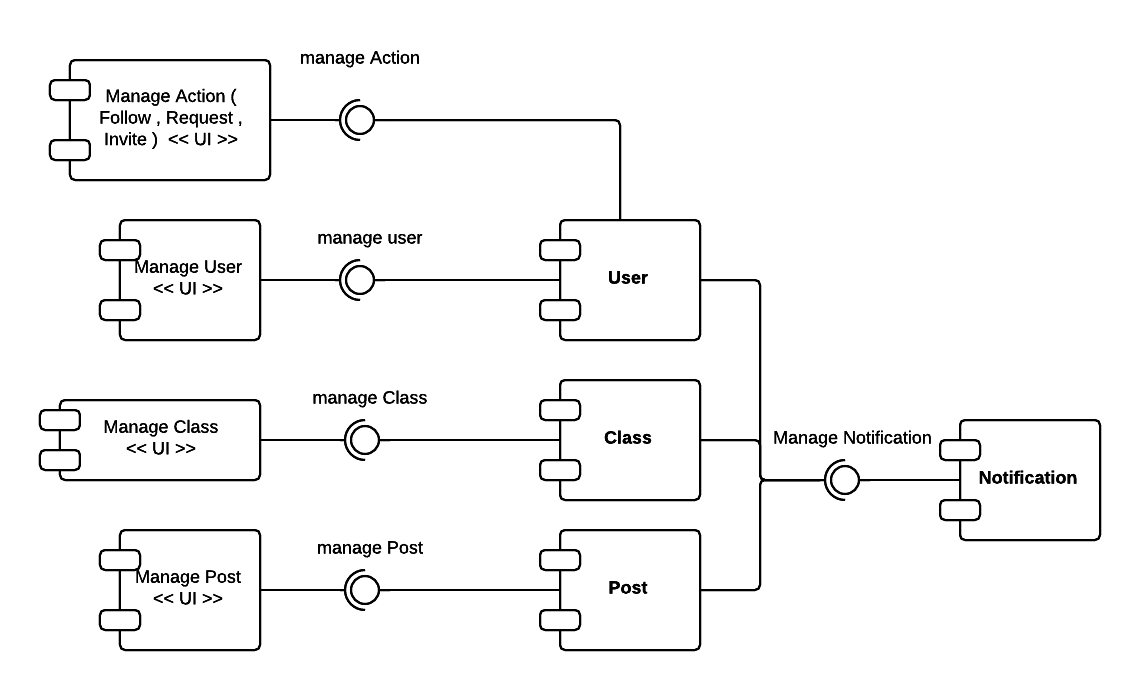
### Web architecture design

|  |  |
| --- | --- |
| Model – View – Controller Architecture | |
| User (Web browser) | Business logic  Send Request  Process Request  Call  Call  View  Select  Result  Forward/Redirect |
| Controller | Forward/Redirect |
| Model |  |
| View |  |

Table 7: Web architecture design

* Model is the part of the application that handles the logic for the application data. Model often receives data from database, process business, and then sends result to Controller.
* View is the part of the application that handles the display of the data. View often receives data from Controller and shows the data to user.
* Controller is the part of the application that handles user interaction. Controller often receives request from user, sends request to suitable Model, receives result from Model, and sends result to suitable View.

## Component Diagram



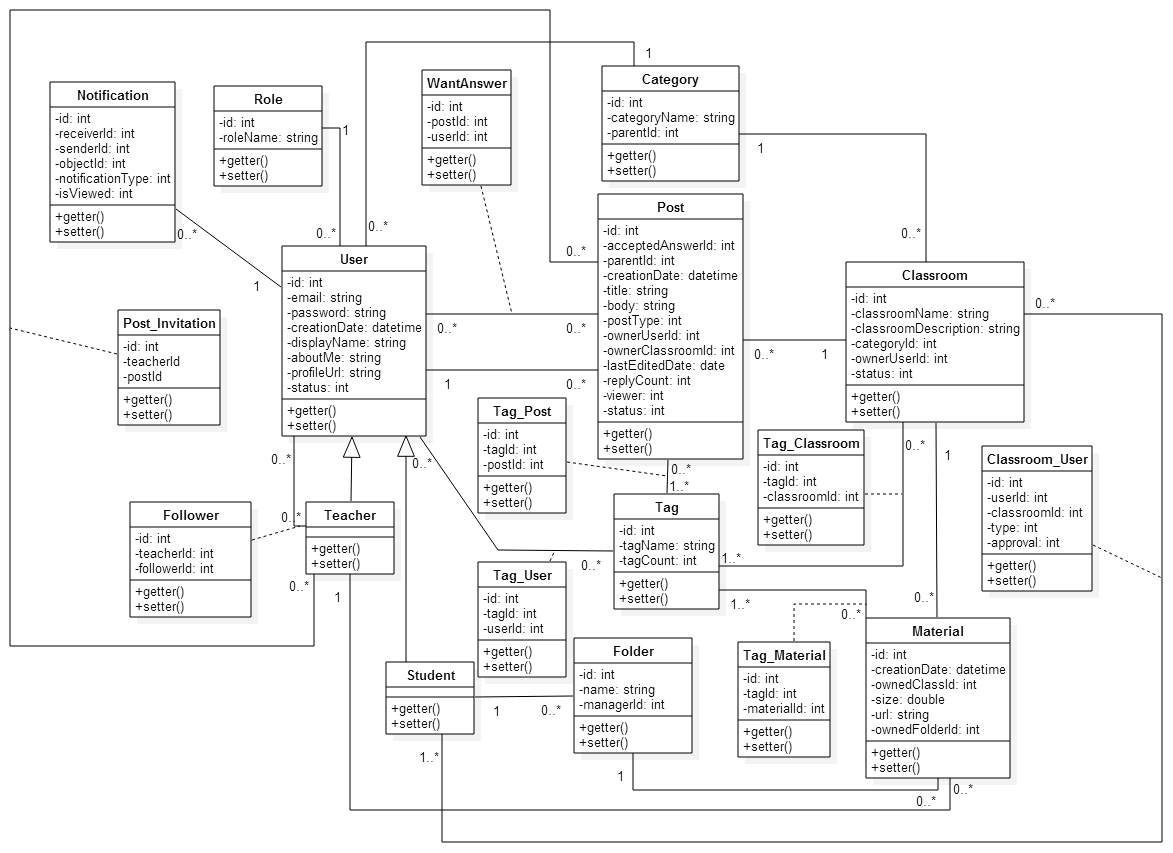
**Figure 61: Component diagram**

|  |  |
| --- | --- |
| **Component dictionary: describe component** | |
| Component Name | Description |
| User | Provide interface manage user action |
| Class | Provide interface manage class action |
| Post | Provide interface manage post action |
| Manage Action | Manage action of user |
| Manage User | Manage user |
| Manage Class | Manage class |
| Manage Post | Manage post |
| Notification | Provide interface send notification |

Table 8: Component dictionary

## Detailed Description

### Class Diagram

****

**Figure 62: Class diagram**

|  |  |
| --- | --- |
| **Class dictionary: describe class** | |
| Class Name | Description |
| Category | Describe attribute and operation of category |
| Classroom | Describe attribute and operation of classroom |
| Classroom\_User | Describe attribute and operation of relationship between classroom and student |
| Folder | Describe attribute and operation of folder |
| Follower | Describe attribute and operation of relationship between teacher and other users |
| Material | Describe attribute and operation of material |
| Notification | Describe attribute and operation of notification |
| Post | Describe attribute and operation of post |
| Post\_Invitation | Describe attribute and operation of relationship between post and teacher |
| Role | Describe attribute and operation of role |
| Student | Describe attribute and operation of student |
| Tag | Describe attribute and operation of tag |
| Tag\_Classroom | Describe attribute and operation of relationship between tag and classroom |
| Tag\_Material | Describe attribute and operation of relationship between tag and material |
| Tag\_Post | Describe attribute and operation of relationship between tag and post |
| Tag\_User | Describe attribute and operation of relationship between tag and user |
| Teacher | Describe attribute and operation of teacher |
| User | Describe attribute and operation of user |
| WantAnswerPost | Describe attribute and operation of relationship between post and user |

Table 9: Class dictionary

### Class Diagram Explanation

#### Category

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a category |
| categoryName | String | private | Name of category |
| parentId | Integer | private | Parent identifier of a category |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Classroom

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a classroom |
| classroomName | String | private | Name of classroom |
| classroomDescription | String | private | Description of classroom |
| categoryId | Integer | private | Identifier of a category that classroom belongs to |
| ownerUserId | Integer | private | Identifier of a user(teacher) that classroom belongs to |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Classroom\_User

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between classroom and user (student) |
| userId | Integer | private | Identifier of a user (student) |
| classroomId | Integer | private | Identifier of a classroom |
| type | Integer | private | Relationship type between student and classroom: 1: student request to join classroom  2: student is invited to join classroom |
| approval | Integer | private | Check if student has joined classroom or not:  0: pending  1: accept  2: ignore |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Folder

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a folder |
| name | String | private | Name of folder |
| managerId | String | private | Identifier of a user (student) owned the folder |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Follower

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between teacher and other users |
| teacherId | Integer | private | Identifier of a teacher that is followed |
| userId | Integer | private | Identifier of a user that is following |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Material

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a material |
| name | String | private | Name of material |
| creationDate | Date | private | Uploaded date of material |
| ownerClassId | Integer | private | Identifier of a classroom owned the material |
| folderId | Integer | private | Identifier of a folder the material belongs to |
| size | double | private | Size of material |
| fileURL | String | private | Download link of material |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Notification

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a notification |
| receiverId | Integer | private | Identifier of a user that received notification |
| senderId | Integer | private | Identifier of a user that makes notification occur |
| ownerClassId | Integer | private | Identifier of a classroom owned the material |
| objectId | Integer | private | Identifier of an object (classroom or post) reference to notification |
| notificationType | Integer | private | Notification type:  1: teacher creates new classroom  2: teacher creates new post  3: student creates post  4: teacher/student replies post  5: student requests to join classroom  6: teacher invites student to join classroom  7: teacher invites others to answer question  8: student accepts to join classroom |
| isViewed | Integer | private | Check if user has viewed notification or not:  0: not yet viewed  1: is viewed |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Post

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a post |
| isAccepted |  |  |  |
| parentId | Integer | private | Parent identifier of post (reply type) |
| creationDate | Date | private | Created date of post |
| title | String | private | Title of post (question/article) |
| body | String | private | Body of post |
| postType | Integer | private | Post type:  1: question  2: article  3: reply |
| ownerUserId | Integer | private | Identifier of user owned the post |
| ownerClassId | Integer | private | Identifier of classroom owned the post |
| lastEditedDate | Date | private | Last edited date of post |
| viewer | Integer | private | Number of viewers of post |
| status | Integer | private | Check if post is closed or opened |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Post\_Invitation

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between question and teacher |
| teacherId | Integer | private | Identifier of a teacher invited to answer question |
| postId | Integer | private | Identifier of a post teacher is invited to |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Role

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between question and teacher |
| roleName | String | private | Name of role |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Student

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Tag

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a tag |
| tagName | String | private | Name of tag |
| tagCount | Integer | private | Number of used times of tag |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Tag\_Classroom

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between tag and classroom |
| tagId | Integer | private | Identifier of tag |
| classroomId | Integer | private | Identifier of classroom |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Tag\_Material

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between tag and material |
| tagId | Integer | private | Identifier of tag |
| materialId | Integer | private | Identifier of material |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Tag\_Post

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between tag and post |
| tagId | Integer | private | Identifier of tag |
| postId | Integer | private | Identifier of post |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Tag\_User

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between tag and user |
| tagId | Integer | private | Identifier of tag |
| userId | Integer | private | Identifier of user |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### Teacher

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### User

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a user |
| email | String | private | Email of user |
| password | String | private | Password of user |
| creationDate | date | private | Created date of user |
| displayName | String | private | Display name of user |
| aboutMe | String | private | Introduction of user |
| profileImageURL | String | private | URL of image of user |
| status | Integer | private | Check if user is activated or deactivated:  0: activated  1: deactivated |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

#### WantAnswerPost

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a relationship between post and user |
| postId | Integer | private | Identifier of post |
| userId | Integer | private | Identifier of user |

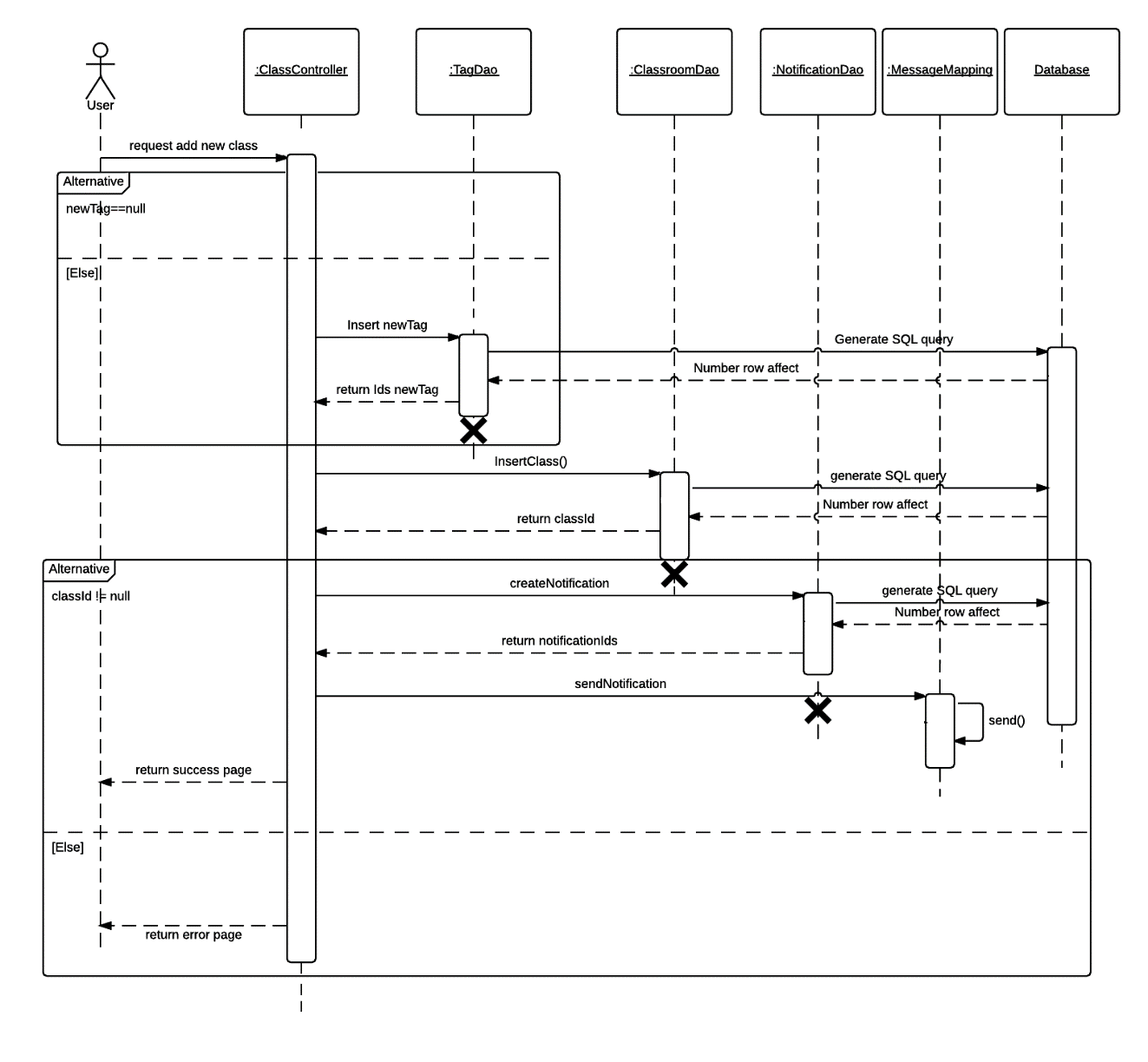
Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getter | attribute type | public | Get attribute value |
| setter | void | public | Set value of attribute |

### Sequence Diagram

#### <Teacher> Create classroom

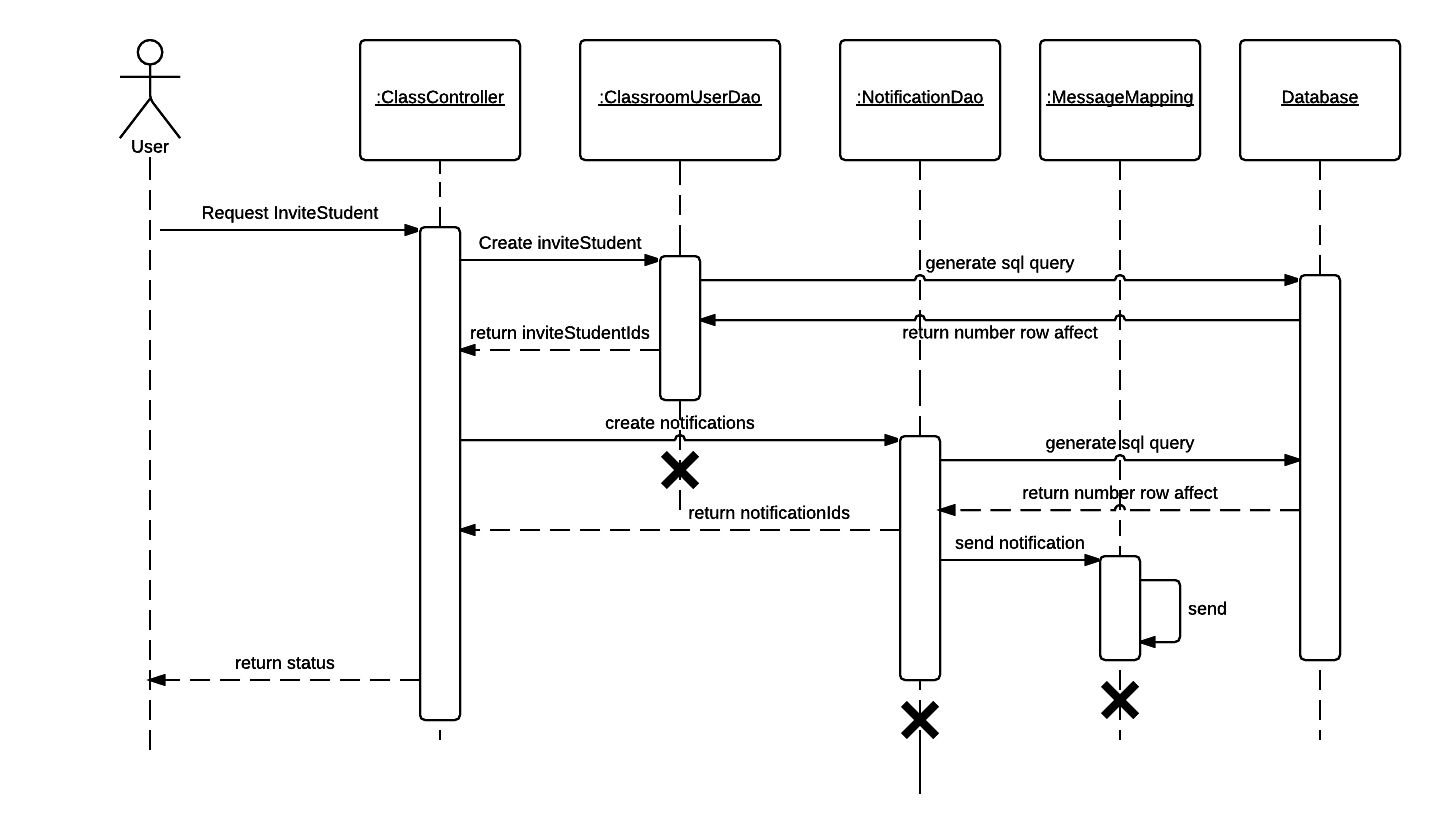
**Summary**: This diagram shows how teacher create a classroom



**Figure 63: <Teacher> Create classroom sequence**

#### <Teacher> Invite student

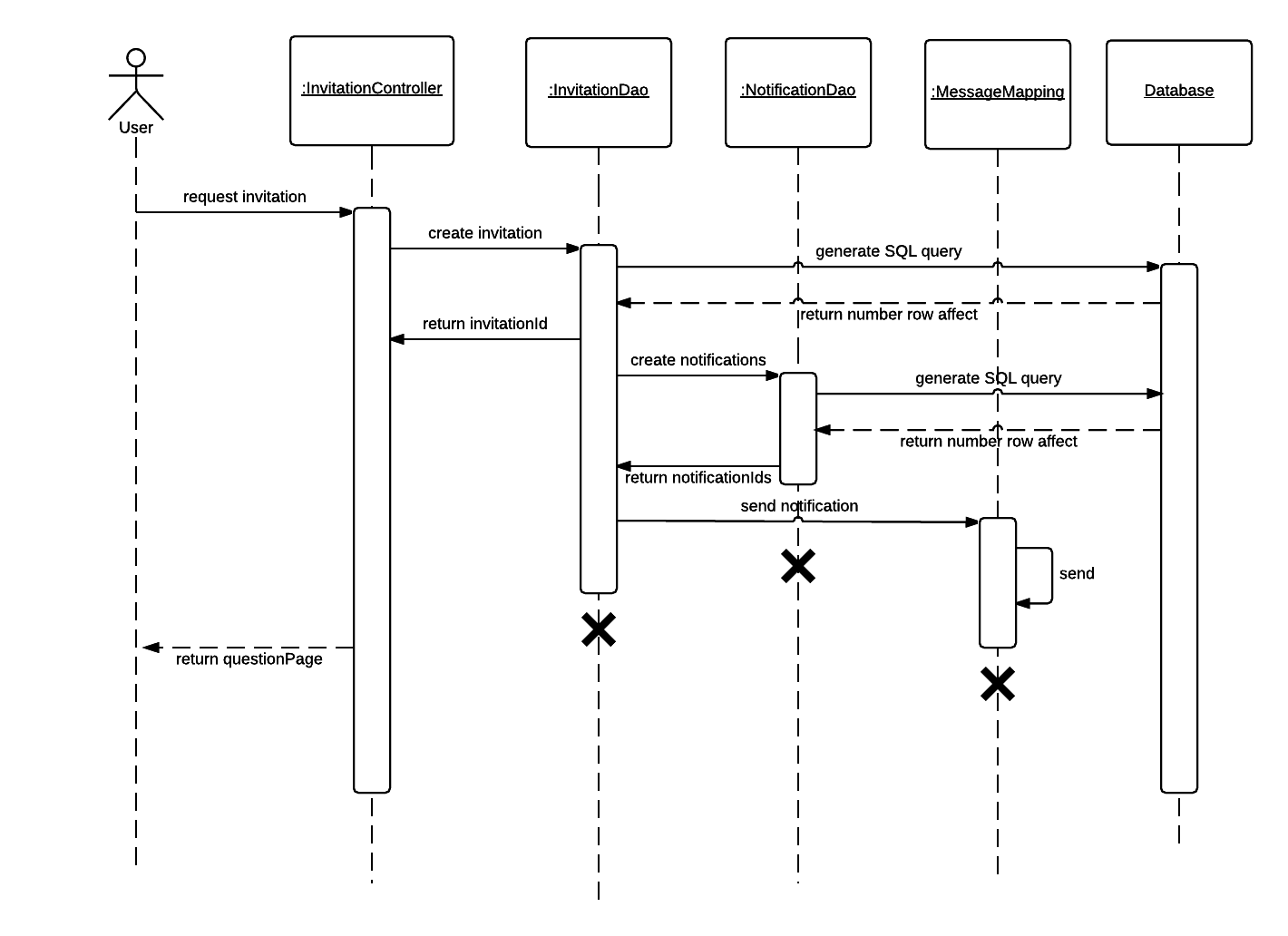
**Summary**: This diagram shows how teacher invites student to join classroom



**Figure 64: <Teacher> Invite student sequence**

#### <Teacher> Invite teacher

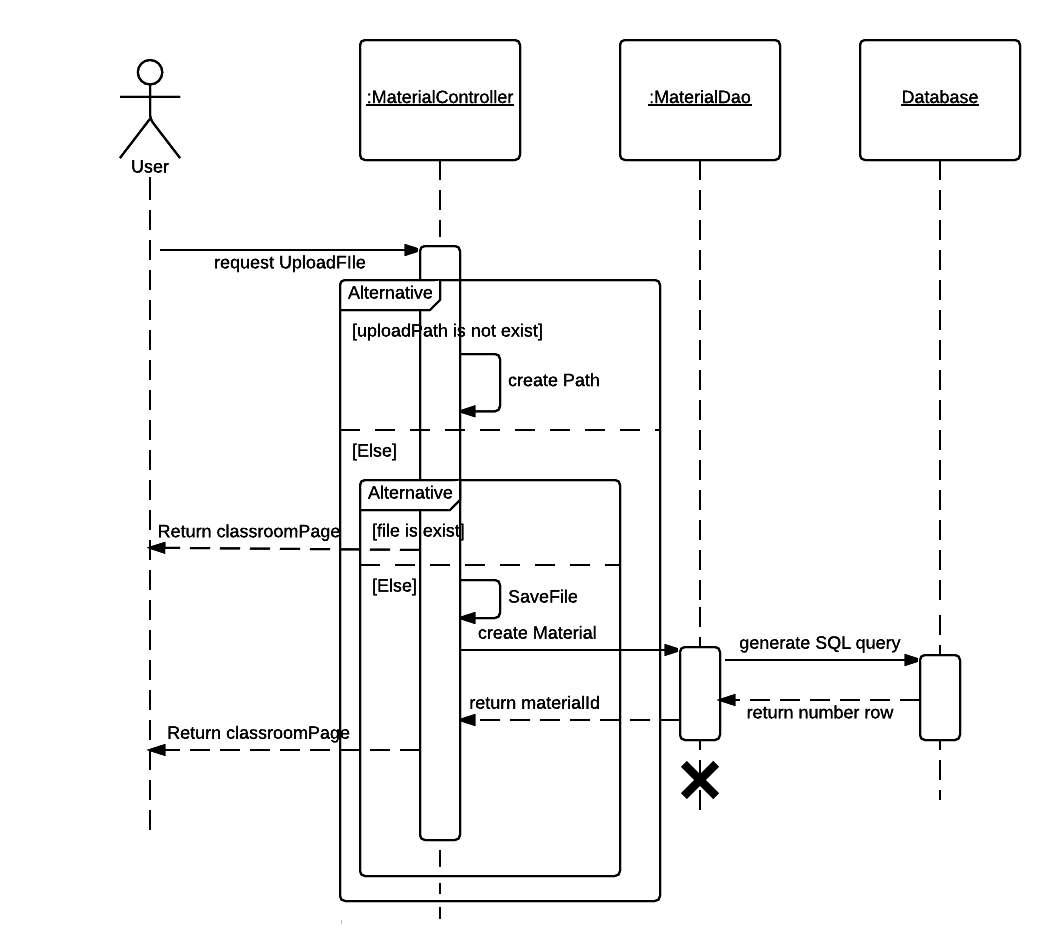
**Summary**: This diagram shows how teacher invites teacher to discuss in question



**Figure 65: <Teacher> Invite teacher sequence**

#### <Teacher> Upload material

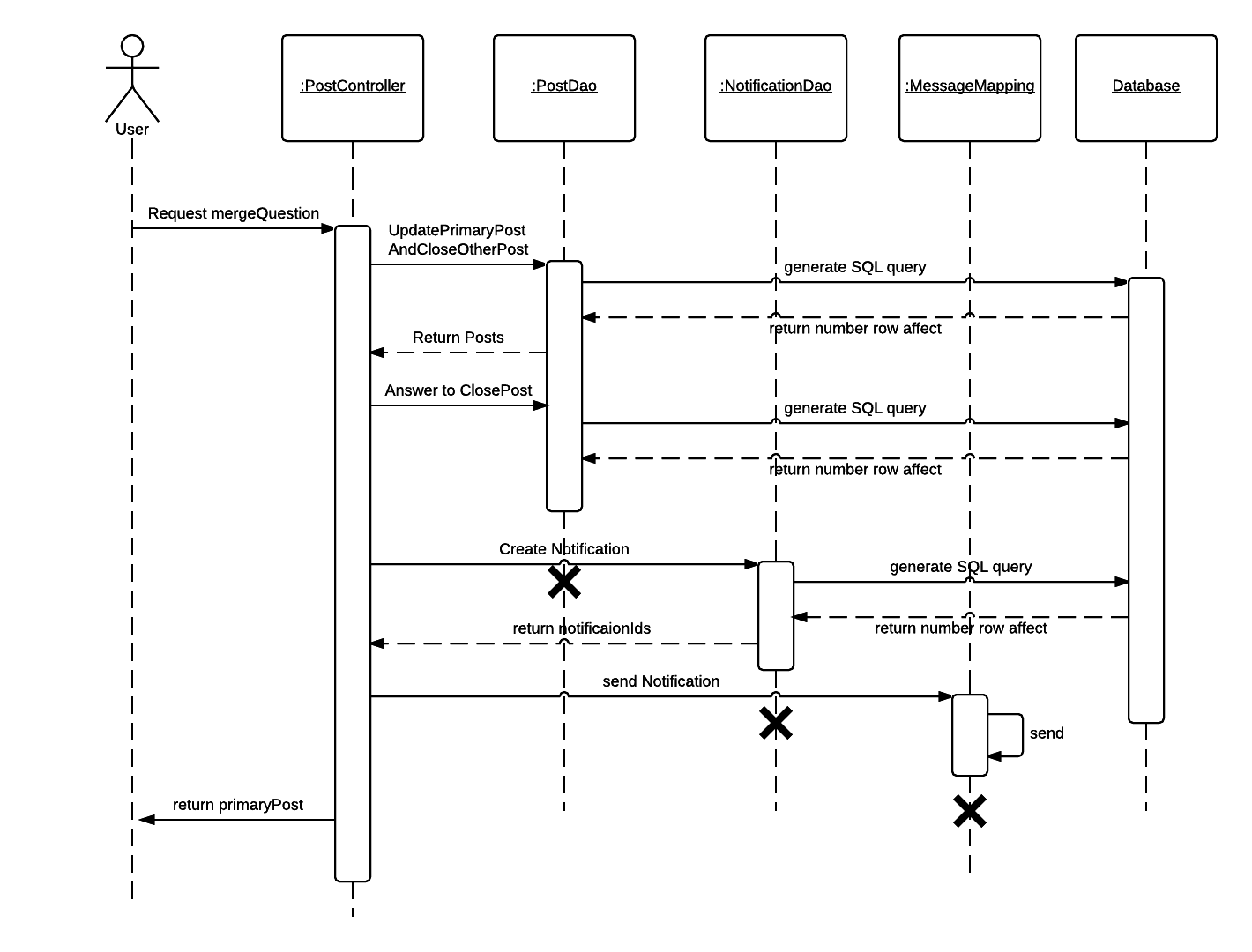
**Summary**: This diagram shows how teacher upload material



**Figure 66: <Teacher> Upload material sequence**

#### <Teacher> Merge questions

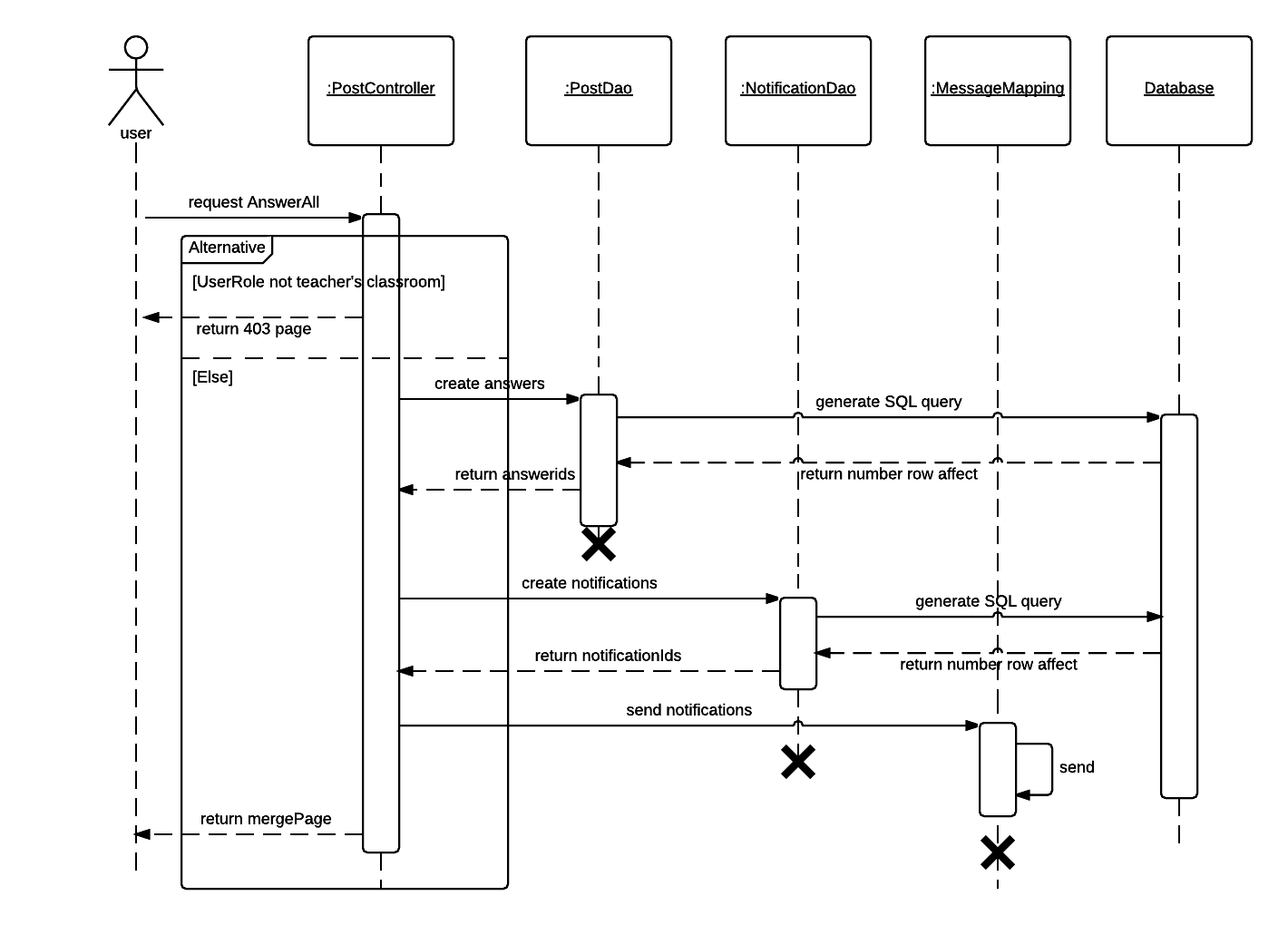
**Summary**: This diagram shows how teacher merges similar questions



**Figure 67: <Teacher> Merge questions sequence**

#### <Teacher> Answer multi questions

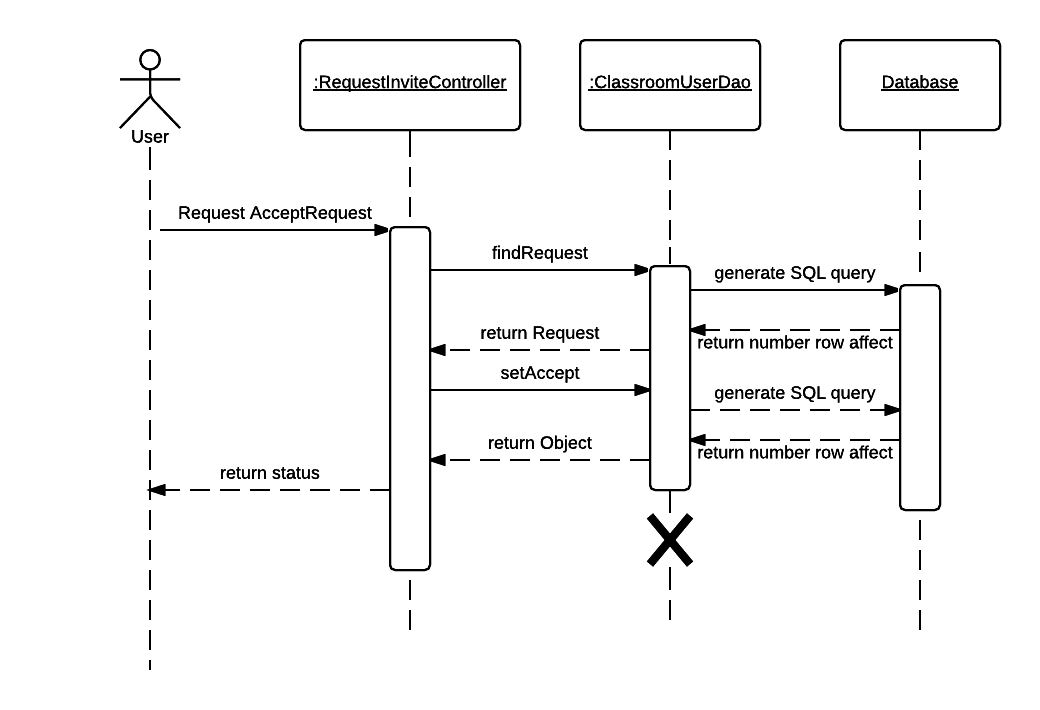
**Summary**: This diagram shows how teacher answers multi similar questions when merging



**Figure 68: <Teacher> Answer multi questions sequence**

#### <Teacher> Accept request

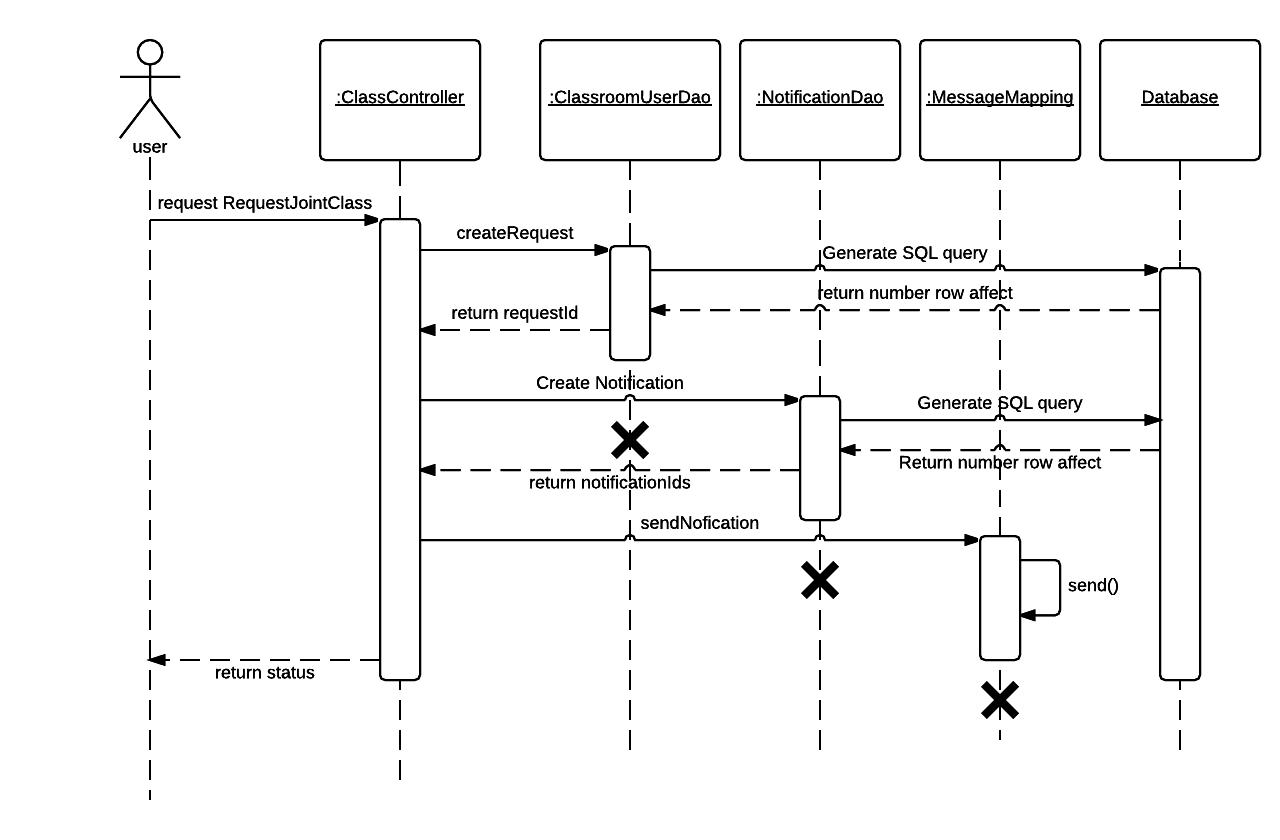
**Summary**: This diagram shows how teacher accepts student’s request



**Figure 69: <Teacher> Accept request sequence**

#### <Student> Request to join

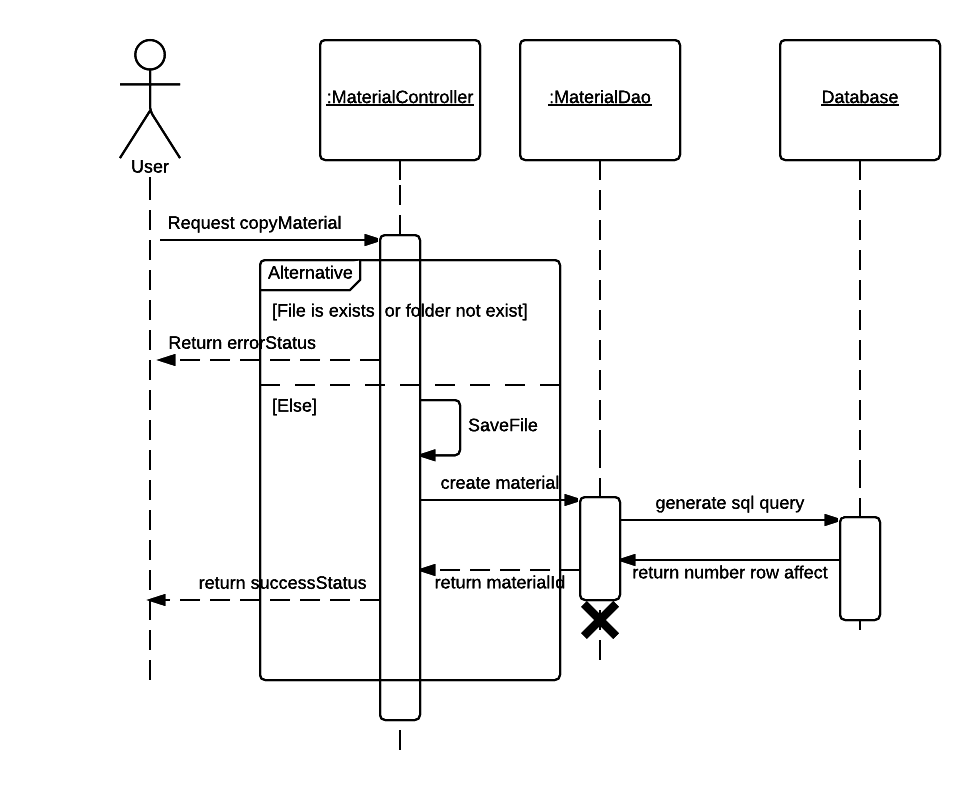
**Summary**: This diagram shows how student requests to join classroom



**Figure 70: <Student> Request to join sequence**

#### <Student> Add to folder

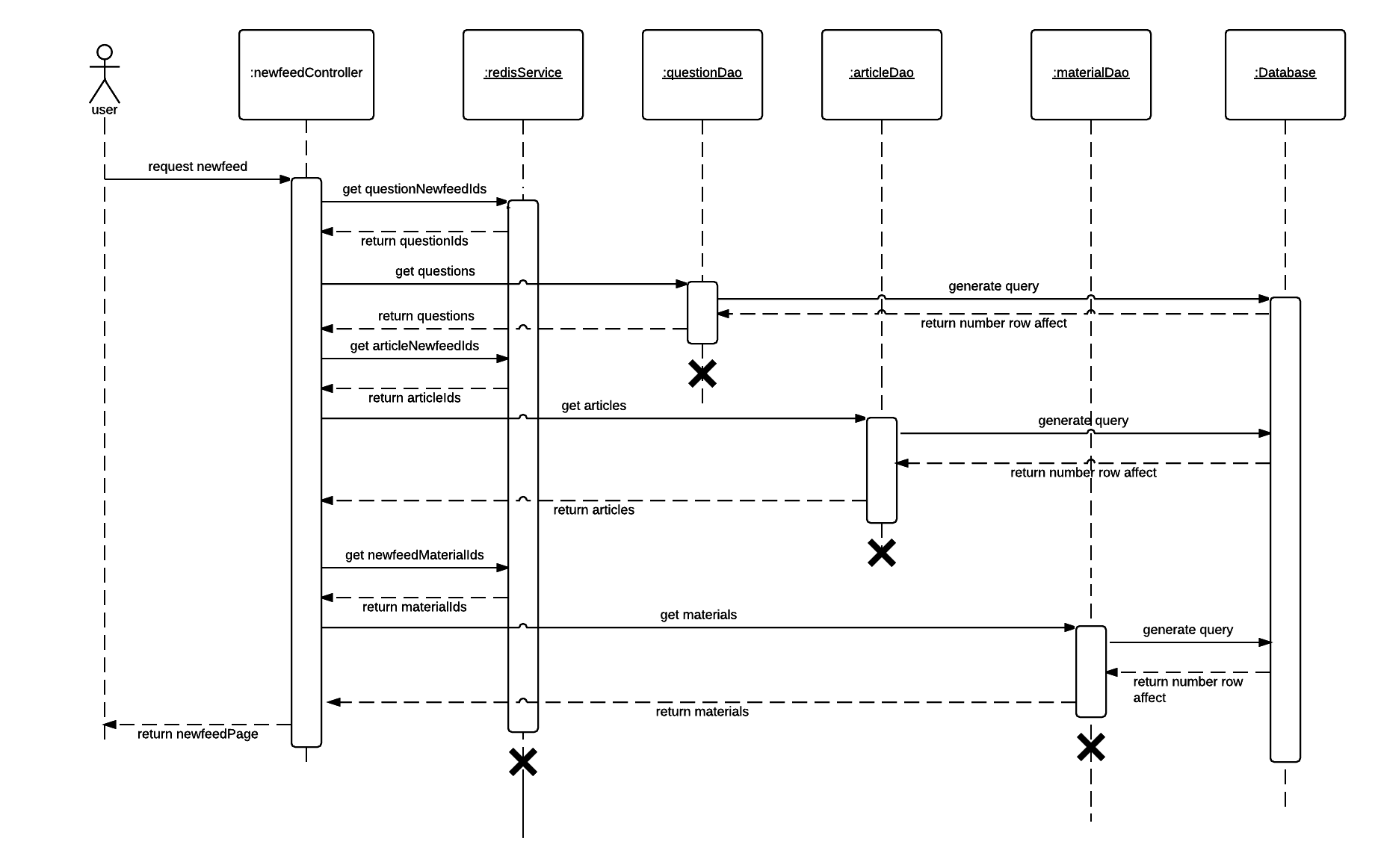
**Summary**: This diagram shows how student adds material to folder



**Figure 71: <Student> Add to folder sequence**

#### <Teacher, Student> View news feed

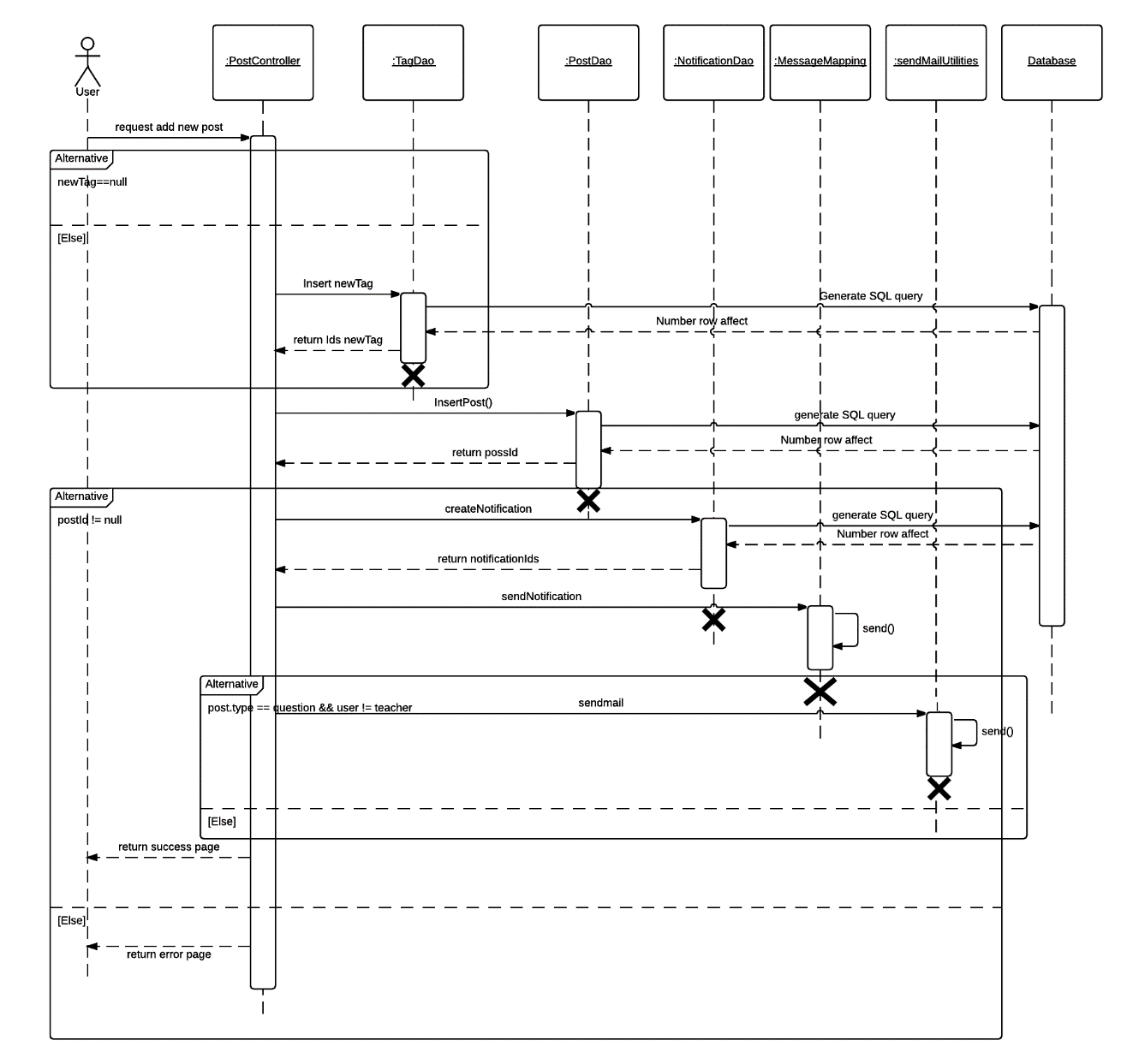
**Summary**: This diagram shows how teacher or student views news feed page



**Figure 72: <Teacher, Student> View news feed sequence**

#### <Teacher, Student> Create post

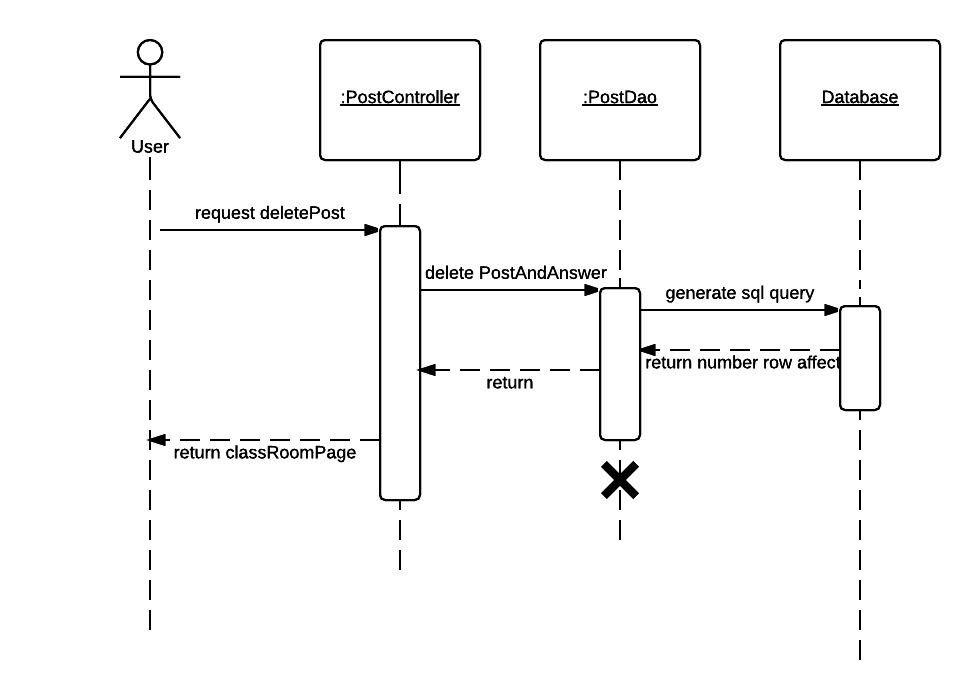
**Summary**: This diagram shows how teacher or student creates new post



**Figure 73: <Teacher, Student> Create post sequence**

#### <Teacher, Student> Delete post

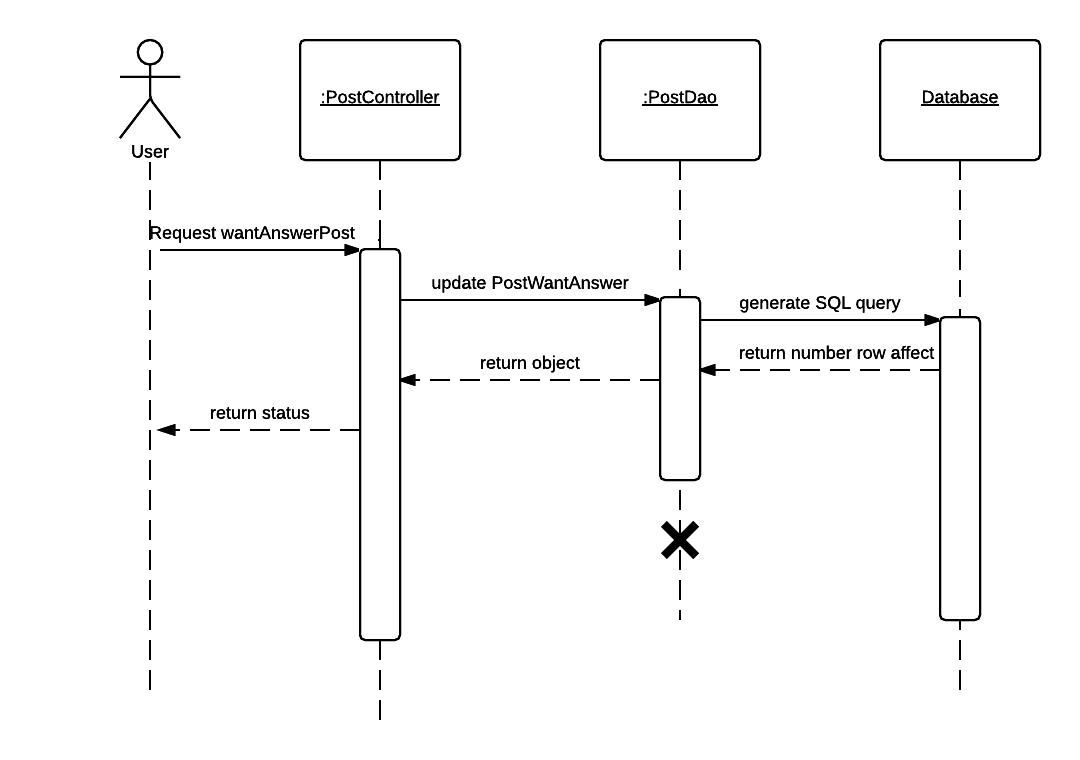
**Summary**: This diagram shows how teacher or student deletes new post



**Figure 74: <Teacher, Student> Delete post sequence**

#### <Teacher, Student> Want answer

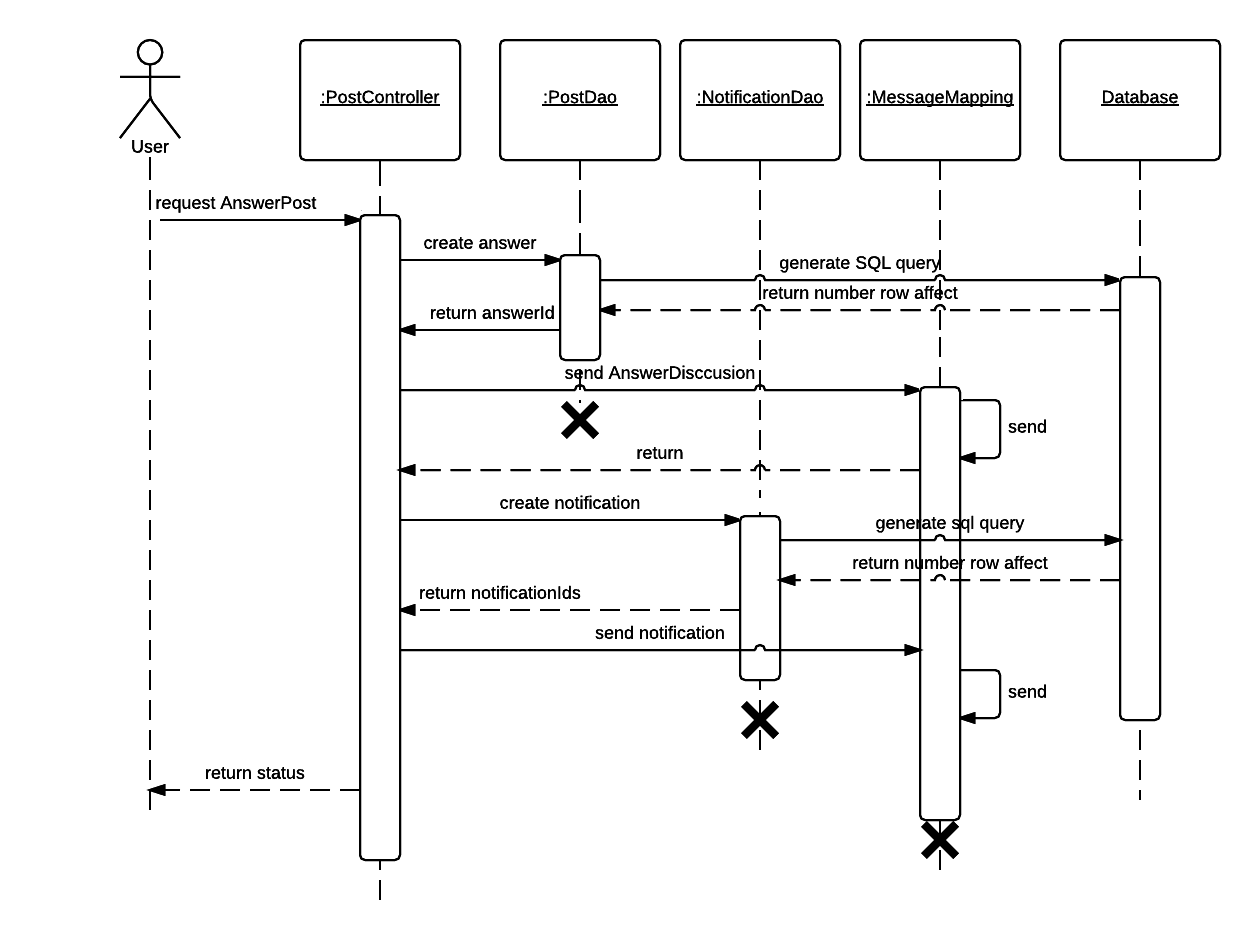
**Summary**: This diagram shows how teacher or student wants answer a question



**Figure 75: <Teacher, Student> Want answer sequence**

#### <Teacher, Student> Answer

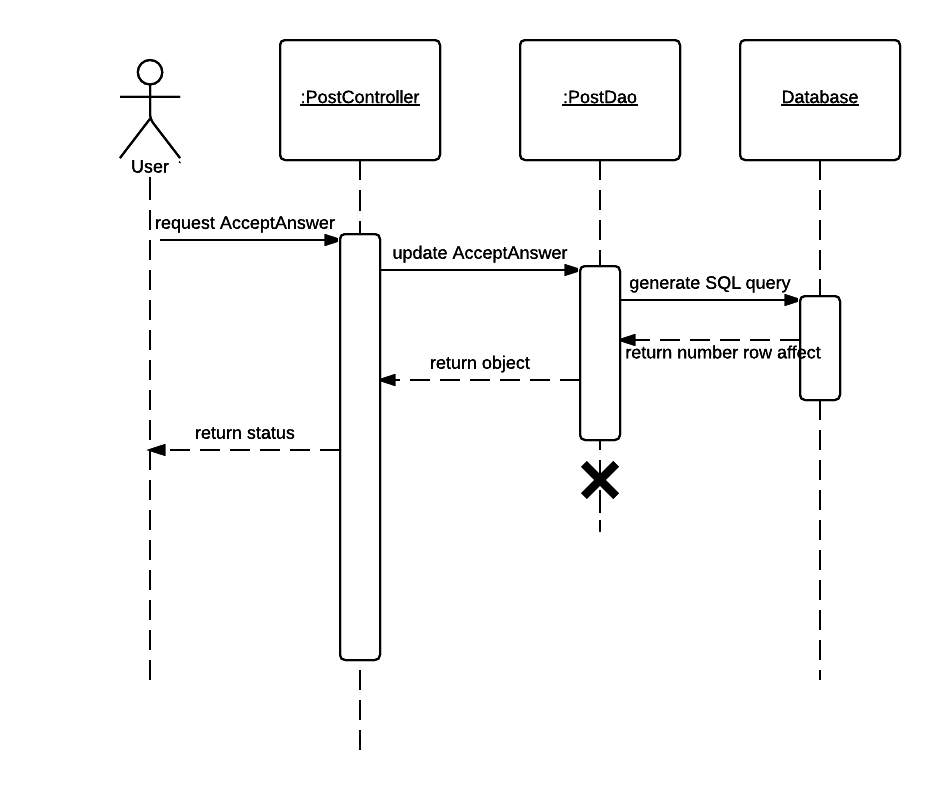
**Summary**: This diagram shows how teacher or student answers a question



**Figure 76: <Teacher, Student> Answers sequence**

#### <Teacher, Student> Accept answer

**Summary**: This diagram shows how teacher or student accepts answer a question

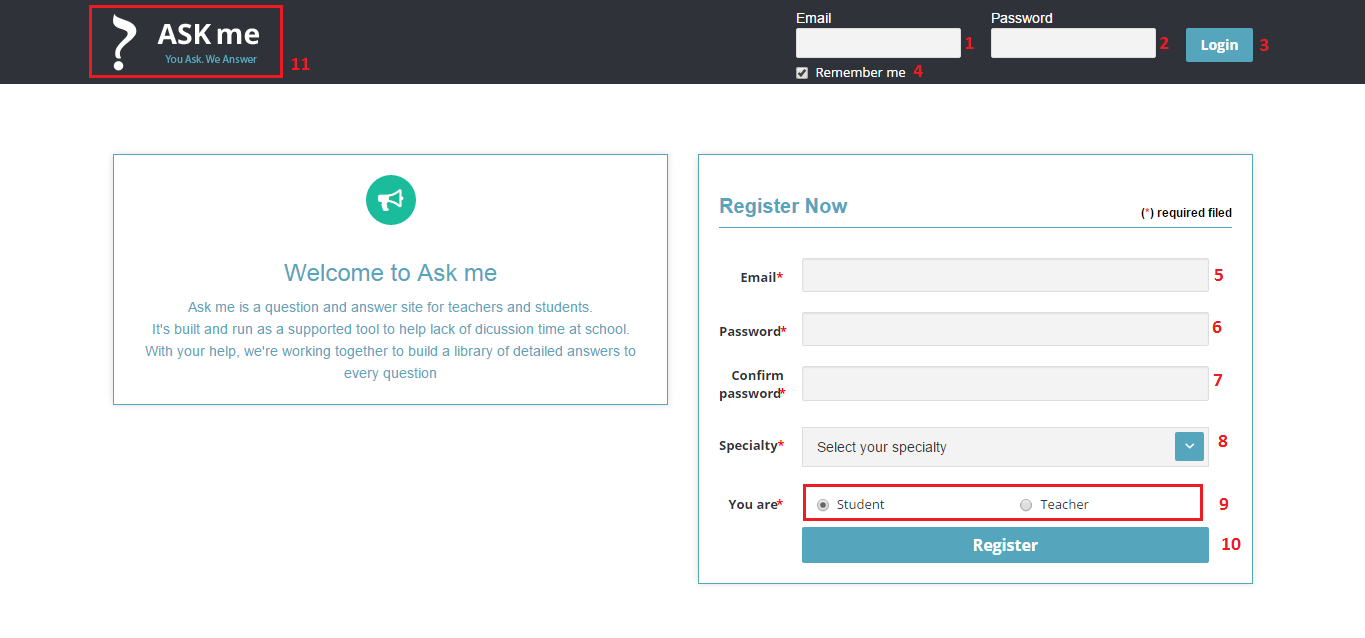


**Figure 77: <Teacher, Student> Accept answer sequence**

## User Interface Design

### Guest Interface Design

**Login and register**



**Figure 78: Login and register interface**

**Fields**

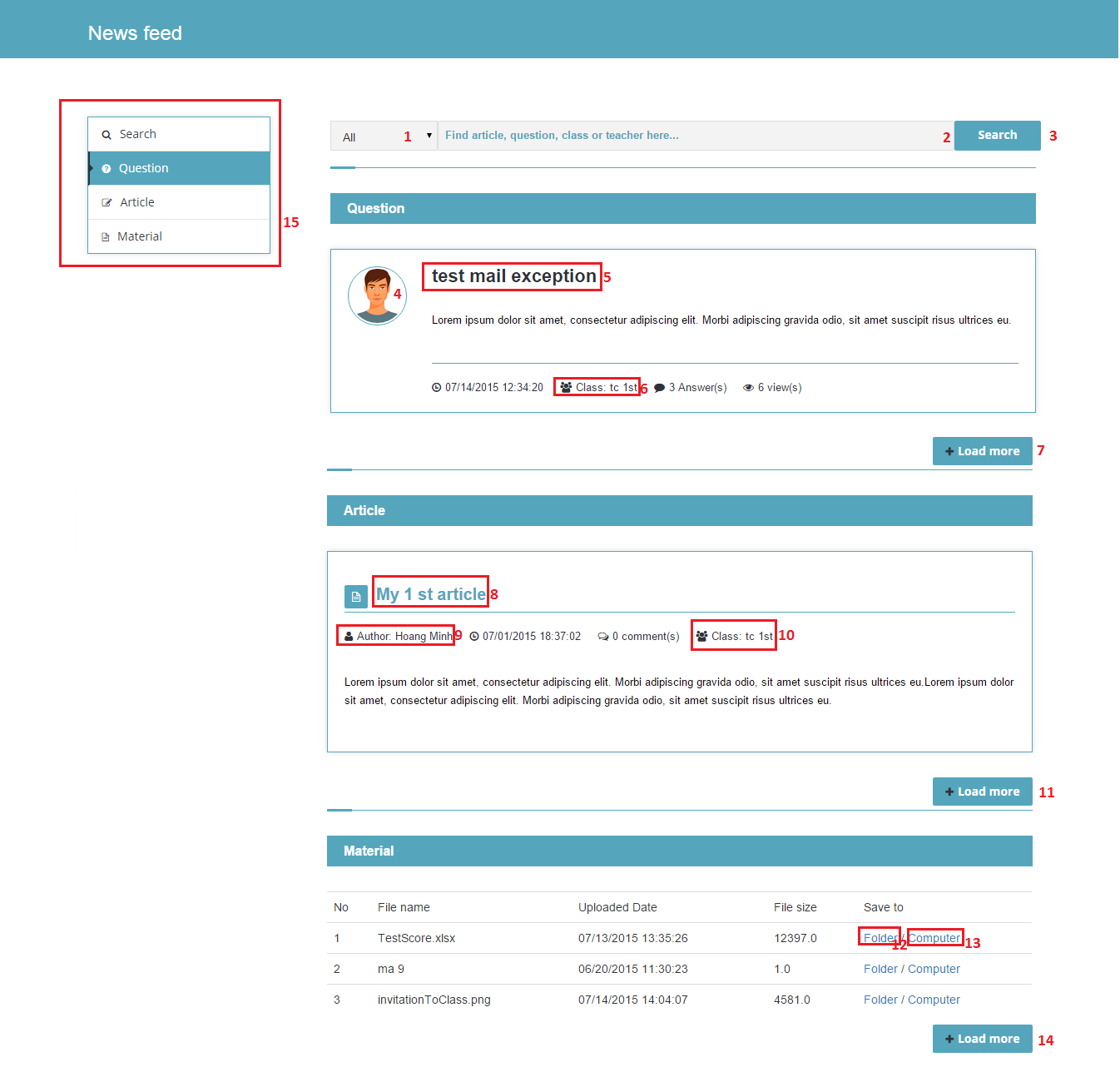
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Email | Fill email | No | Yes | Textbox | String | N/A |
| 2 | Password | Fill password | No | Yes | Password | String | N/A |
| 5 | Email | Fill email | No | Yes | Textbox | String | N/A |
| 6 | Password | Fill password | No | Yes | Password | String | N/A |
| 7 | Confirm password | Fill confirm password | No | Yes | Password | String | N/A |
| 8 | Specialty | Select specialty | No | Yes | Combo box | String | N/A |
| 9 | You are | Select role | No | Yes | Radio button | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Login | Log-in to the system | N/A | Transfer to news feed page |
| 10 | Register | Register an account | N/A | Transfer to dashboard page |
| 11 | Welcome | Go to welcome page | N/A | Transfer to welcome page |

### Teacher Interface Design

#### News feed

****

**Figure 79: News feed interface**

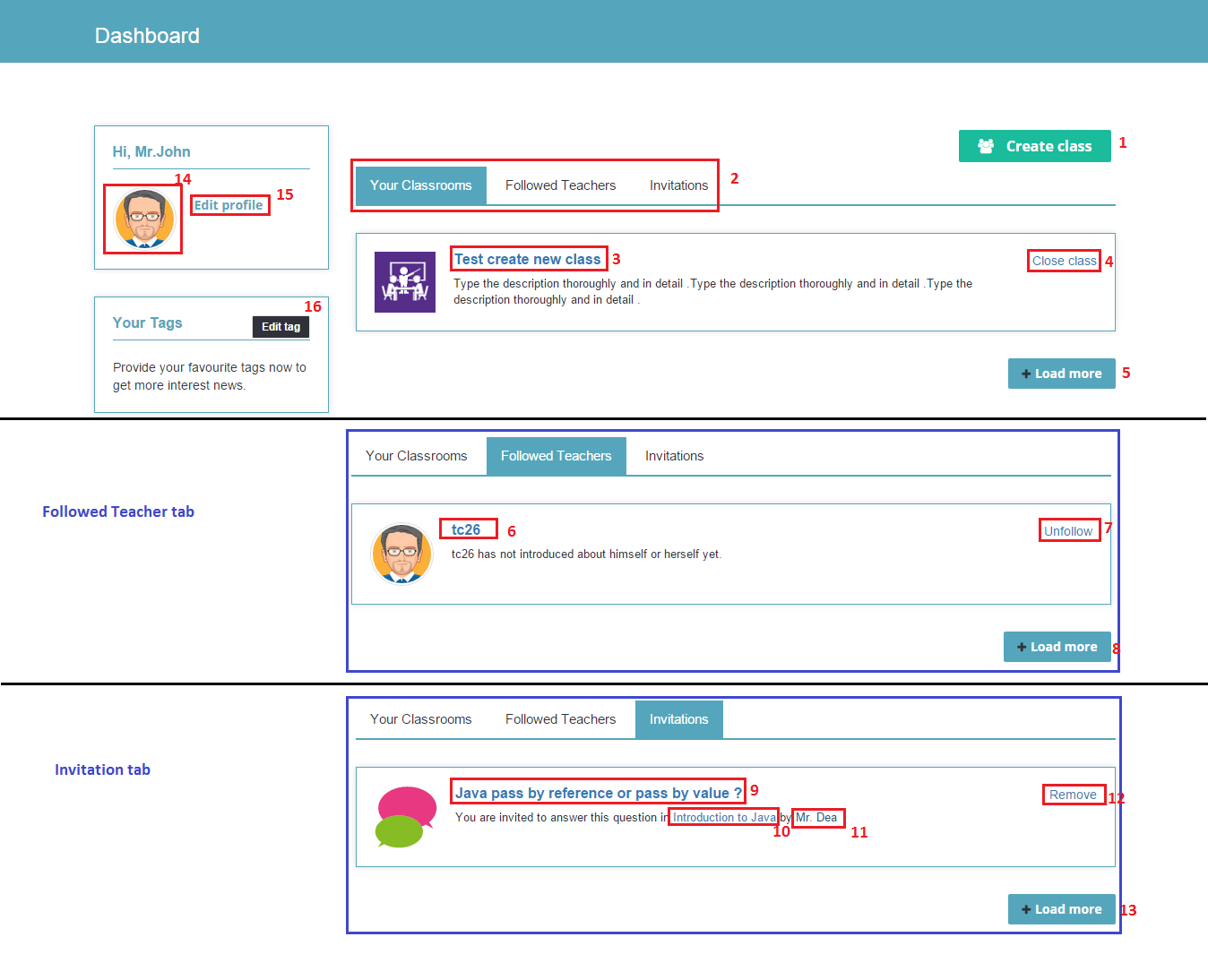
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Filter | Select filter | No | No | Combo box | String | N/A |
| 2 | Search keyword | Fill search keyword | No | No | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Search | Search with selected filter and inputted keyword | N/A | Transfer to search result page |
| 4 | View profile | Display avatar, click to view profile of user who is post owner | N/A | Transfer to profile page |
| 5 | View post | Display post title, click to view question/article | N/A | Transfer to question/article page |
| 6 | View classroom | Display class’s name, click to view classroom that contains post | N/A | Transfer to classroom page |
| 7 | Load more | View more question/article/material | N/A | Show more question/article/material |
| 8 | Referenced to No.5 | | | |
| 9 | Referenced to No.4 | | | |
| 10 | Referenced to No.6 | | | |
| 11 | Referenced to No.7 | | | |
| 12 | View folder list | View folder list of student (only displayed if user is student) | N/A | Show folder list in pop-up |
| 13 | Download | Save material to PC | N/A | Material is downloaded to PC |
| 14 | Referenced to No.7 | | | |
| 15 | Menu | Navigation bar | N/A | Focus to clicked header on menu |

#### Dashboard

****

**Figure 80: Teacher Dashboard interface**

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Create class | Go to create classroom page | N/A | Transfer to Create class page |
| 2 | Tab | Navigation tab | N/A | Show list of clicked tab |
| 3 | View classroom | Display classroom’s name, click to view classroom | N/A | Transfer to classroom page |
| 4 | Close class | Close classroom | N/A | Classroom description is changed to closed |
| 5 | Load more | View more classroom /followed teacher /invitation | N/A | Show more classroom /followed teacher /invitation |
| 6 | View profile | Display teacher’s name, click to view teacher’s profile | N/A | Transfer to profile page |
| 7 | Unfollow | Unfollow teacher | N/A | Show un-followed dialog |
| 8 | Referenced to No.5 | | | |
| 9 | View question | Display question’s title, click to view question | N/A | Transfer to question page |
| 10 | View classroom | Display classroom’s name, click to view classroom. | N/A | Transfer to classroom page |
| 11 | View profile | Display teacher’s name, click to view teacher’s profile | N/A | Transfer to profile page |
| 12 | Remove | Remove invitation | N/A | Show removed dialog |
| 13 | Referenced to No.5 | | | |
| 14 | View profile | Display current user avatar, click to view profile | N/A | Transfer to profile page |
| 15 | Edit profile | Go to edit profile page of logged user | N/A | Transfer to edit profile page |
| 16 | Edit tag | Go to edit profile page of logged user and focus to tag textbox | N/A | Transfer to edit profile page |

#### Create classroom

****

**Figure 81: Create classroom interface**

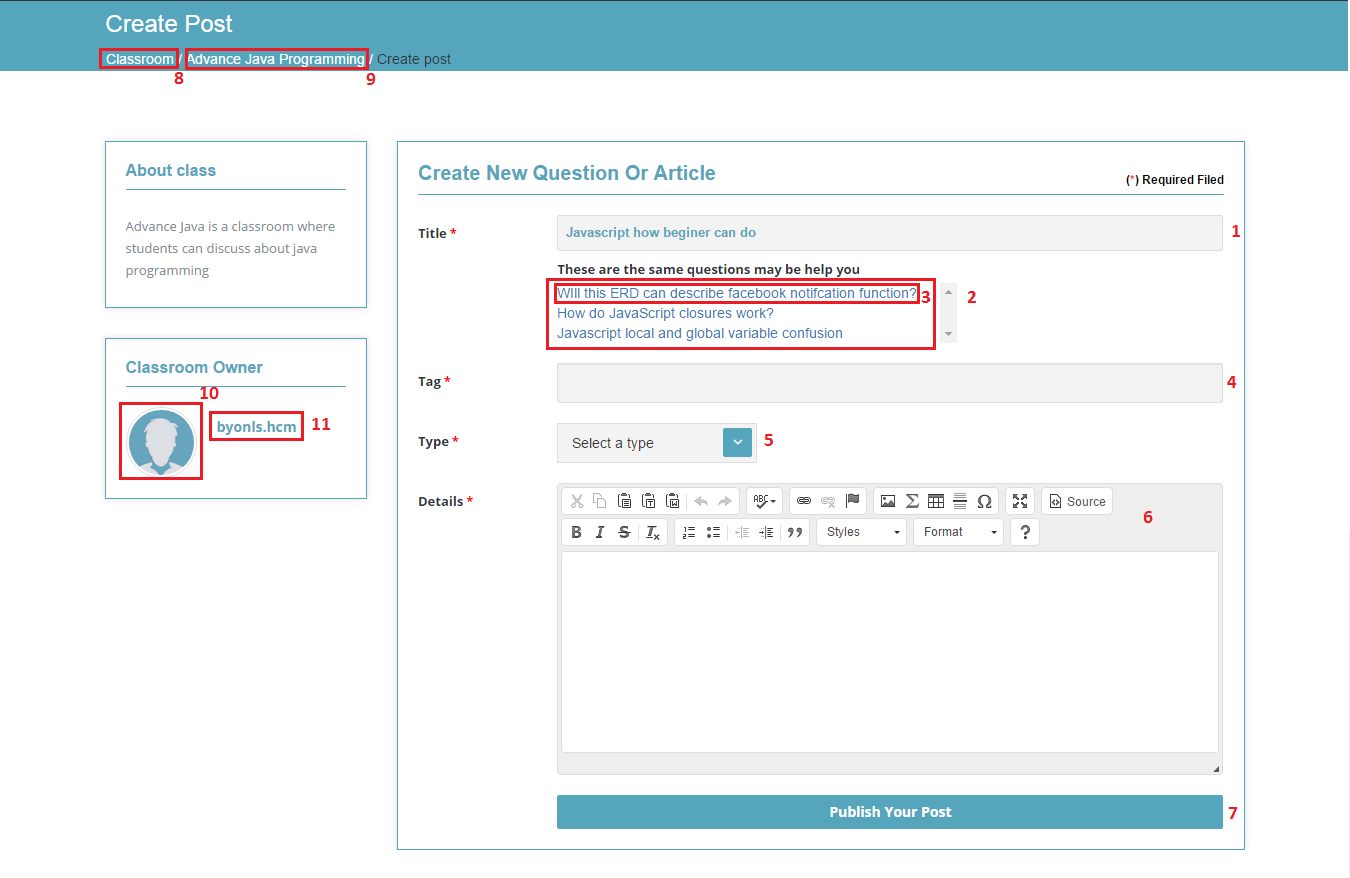
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Class name | Fill classroom’s name | No | Yes | Textbox | String | N/A |
| 2 | Tag | Fill tags | No | Yes | Textbox | String | N/A |
| 3 | Invite student | Fill students’ name | No | No | Textbox | String | N/A |
| 4 | Specialty | Select classroom’s specialty | No | Yes | Combo box | String | N/A |
| 5 | Class’s Description | Fill classroom’s description | No | Yes | Text-area | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 6 | Create class | Create new classroom | N/A | Transfer to classroom page |
| 7 | View profile | Display current user’s avatar, click to view profile | N/A | Transfer to profile page |
| 8 | View profile | Display current user’s name, click to view profile | N/A | Transfer to profile page |

#### Create post

****

**Figure 82: Create post interface**

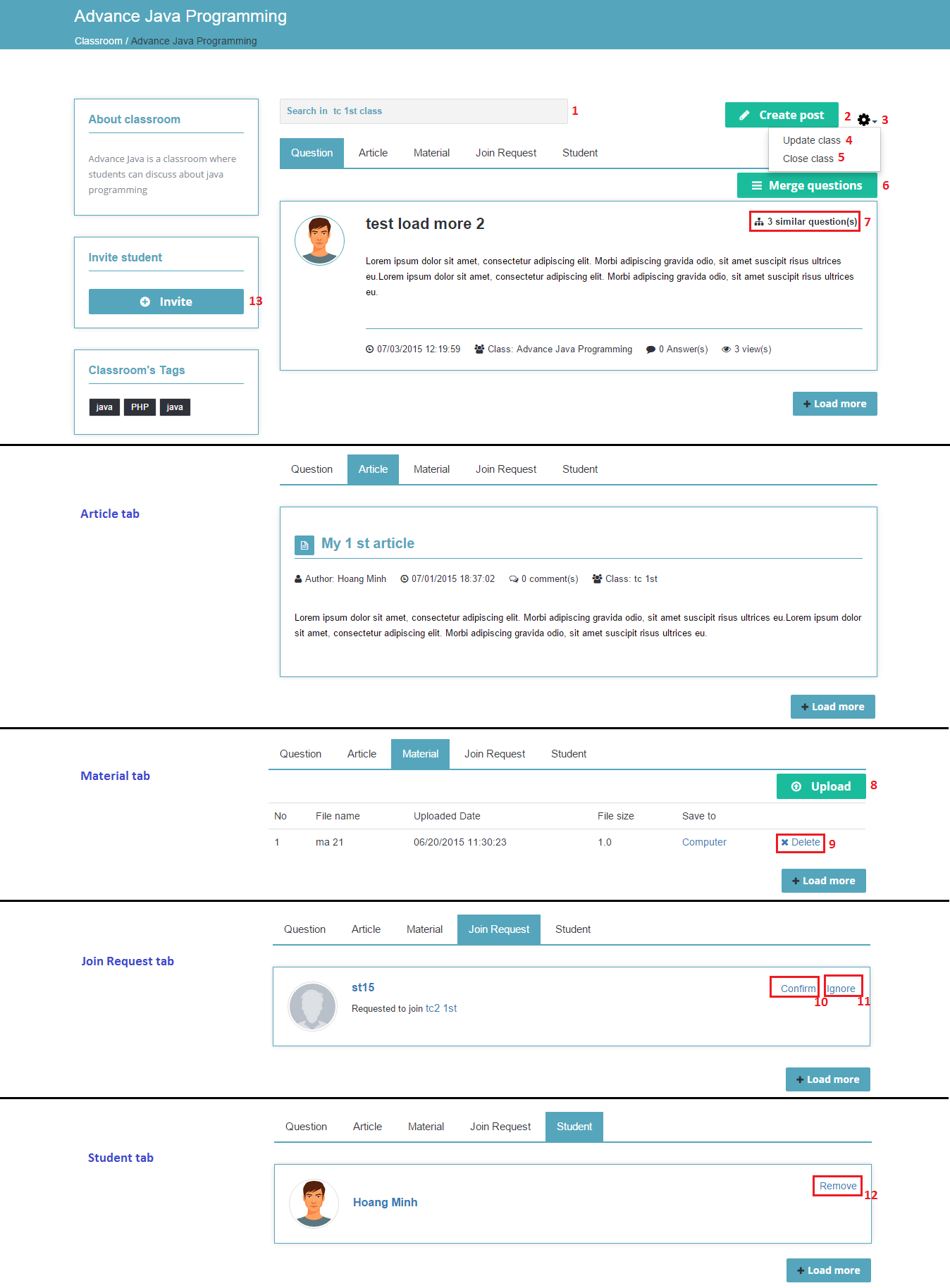
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Title | Fill title | No | Yes | Textbox | String | N/A |
| 2 | Suggested posts | Show suggested related post | Yes | No | Table | String | N/A |
| 4 | Tag | Fill tags | No | Yes | Textbox | String | N/A |
| 5 | Type | Select post’s type | No | Yes | Combo box | String | N/A |
| 6 | Details | Fill details | No | Yes | Textbox with ckeditor plugin | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | View post | Go to clicked post in suggestion list | N/A | Transfer to post view |
| 7 | Publish your post | Create new post | N/A | Transfer to post question/article page |
| 8 | Search classroom | Find all classroom of system | N/A | Transfer to search result page |
| 9 | View classroom | Display classroom’s name, click to view classroom | N/A | Transfer to classroom page |
| 10 | View profile | Display current user’s avatar, click to view profile | N/A | Transfer to profile page |
| 11 | View profile | Display current user’s name, click to view profile | N/A | Transfer to profile page |

#### Classroom view

****

**Figure 83: Teacher Classroom view interface**

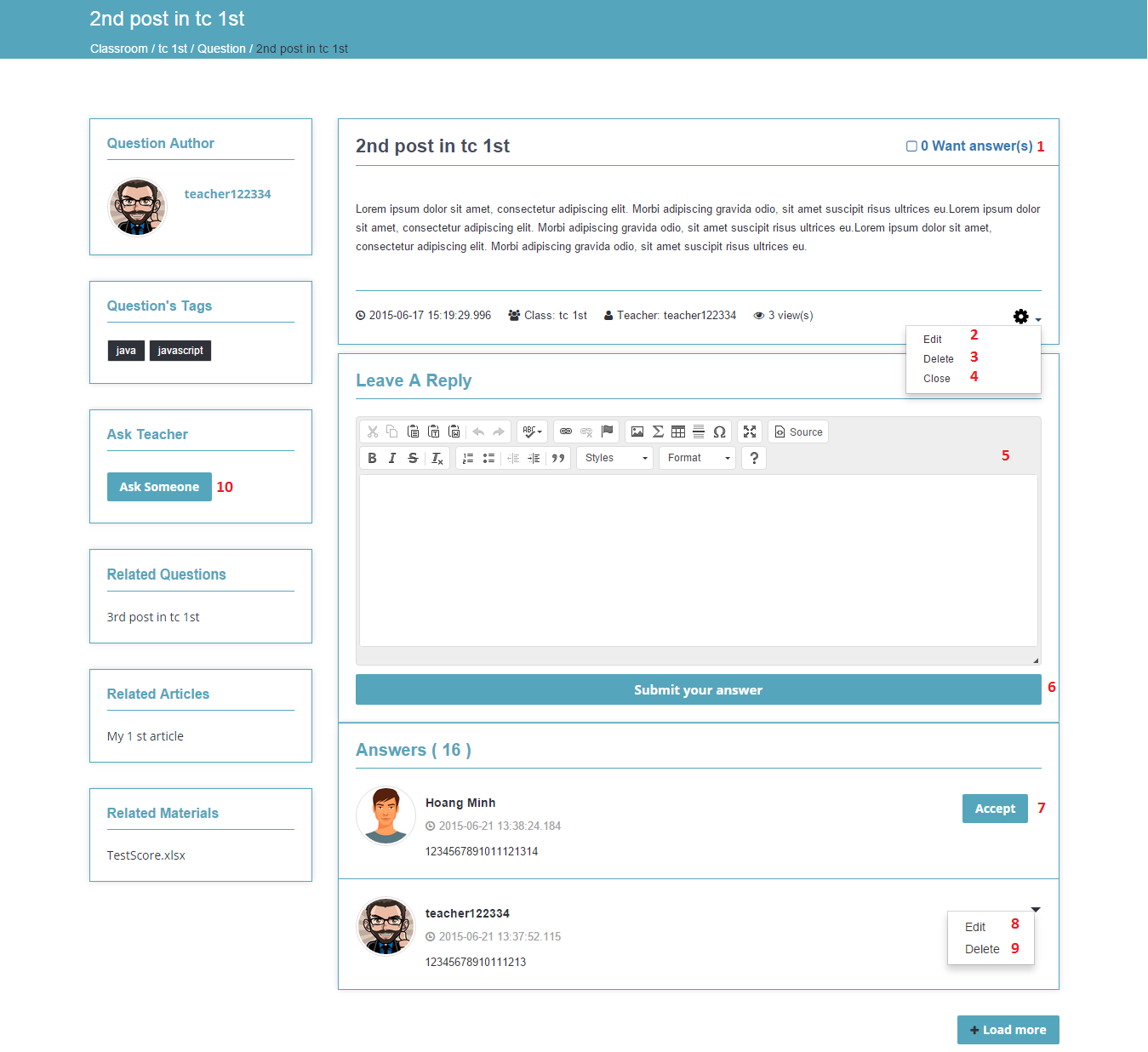
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Search | Fill search keyword, press Enter to search | No | No | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Create post | Go to create new post page | N/A | Transfer to Create post page |
| 3 | Show dropdown | Display dropdown menu action | N/A | Show dropdown menu action |
| 4 | Update class | Go to Update class page | N/A | Transfer to Update class page |
| 5 | Close (Reopen) class | Close classroom (Reopen classroom) | N/A | Classroom description is changed to closed |
| 6 | Merge question | Merge similar question in classroom | N/A | Go to merge question page |
| 7 | Merge question | Merge similar question in classroom | N/A | Go to Create new merged question page |
| 8 | Upload material | View upload material popup | N/A | Show upload material pop-up |
| 9 | Delete material | View confirm delete material pop-up | N/A | Show confirm delete pop-up |
| 10 | Confirm | Confirm the request | N/A | Show confirmed dialog |
| 11 | Ignore | Ignore the request | N/A | Show ignored dialog |
| 12 | Remove | Remove a student | N/A | Show removed dialog |
| 13 | Invite | View invite student pop-up | N/A | Show Invite student pop-up |

#### Post view

****

**Figure 84: Teacher Post view interface**

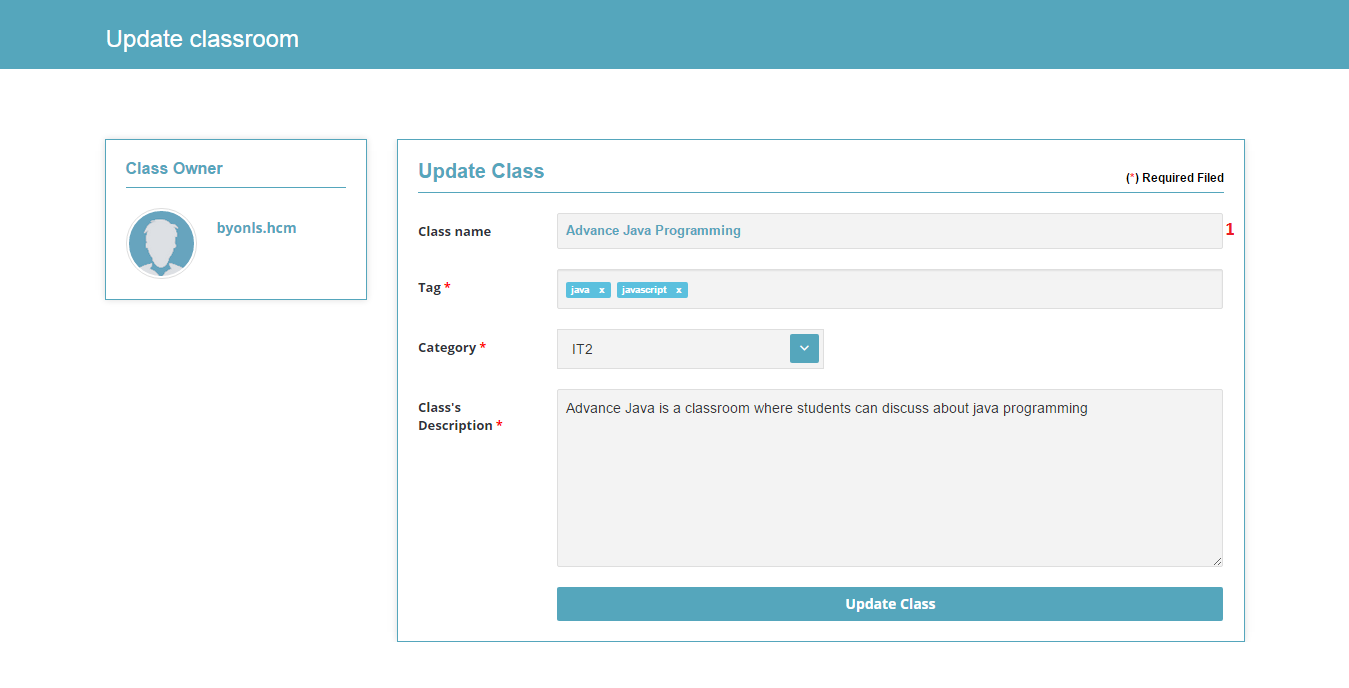
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Want answer | Check to increase number of “want answer” | No | No | Checkbox | N/A | N/A |
| 5 | Leave an answer | Fill answer | No | No | Text-area with ckeditor plugin | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Edit post | Go to edit post page | N/A | Transfer to edit post page |
| 3 | Delete post | Delete post | N/A | Post is deleted.  Transfer to classroom page |
| 4 | Close post | Close post | N/A | Post is closed. “Leave an answer” area is hidden. |
| 6 | Submit answer | Submit inputted answer | N/A | Submitted answer will be shown in “Answer” area |
| 7 | Accept answer | Accept an answer | N/A | “Accept” button is changed to “Unaccept” |
| 8 | Edit answer | Edit answer | N/A | Answer content is putted in text-area with “Update” button to submit |
| 9 | Delete answer | Delete answer | N/A | Answer is removed from “Answer” area |
| 10 | Invite teacher | Invite teacher to answer post | N/A | Show “Invite teacher” popup |

#### Update classroom

****

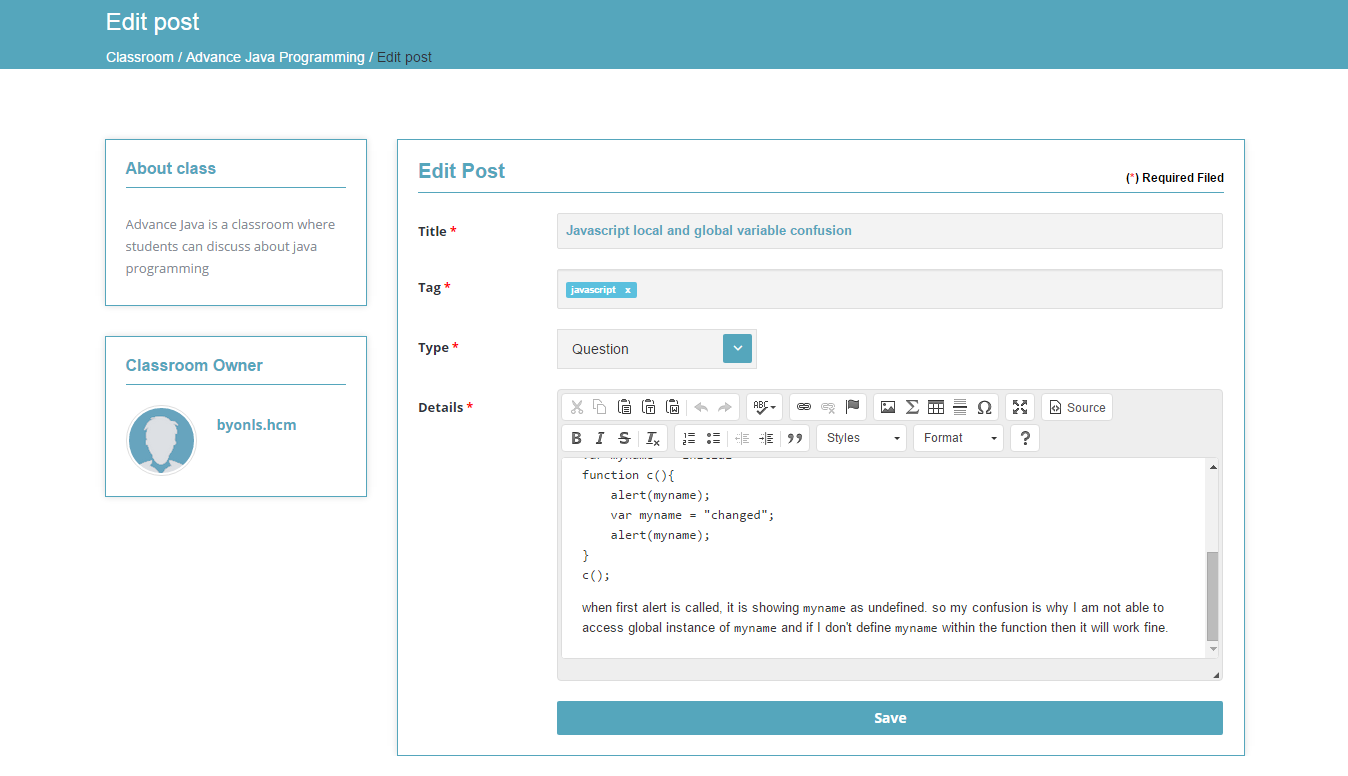
**Figure 85: Update classroom interface**

Fields, Buttons/Hyperlinks are referenced to “Create classroom” (5.2.3) with differences:

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Class name | Show classroom’s name | Yes | No | Textbox | String | N/A |

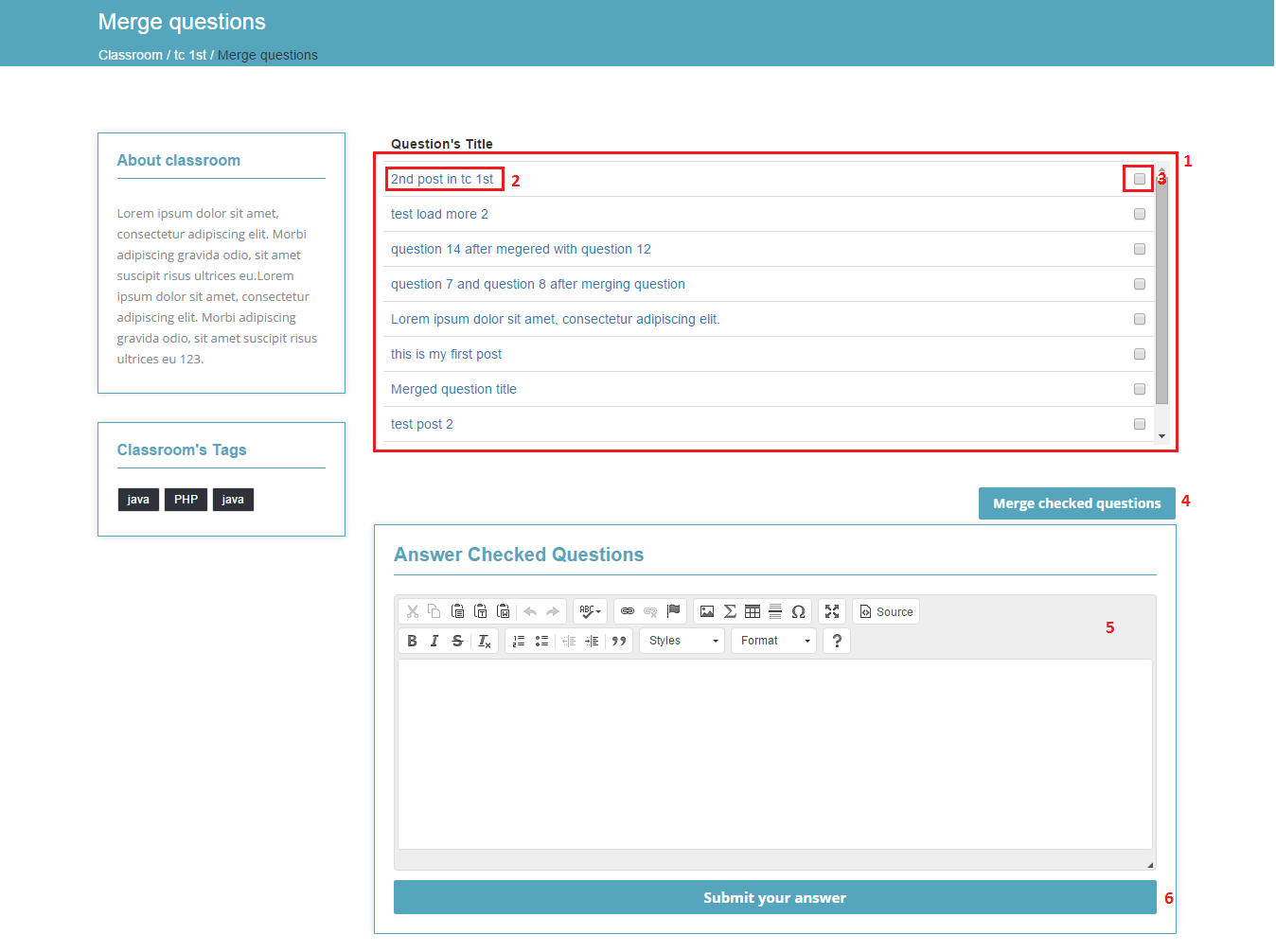
#### Edit post

****

**Figure 86: Edit post interface**

Fields, Buttons/Hyperlinks are referenced to “Create post” (5.2.4)

#### Merge question

****

**Figure 87: Merge question interface**

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Question list | Show list of question | Yes | Yes | Table | String | N/A |
| 3 | Select | Select checkbox to choose question | No | Yes | Checkbox | String | N/A |
| 5 | Answer checked questions | Fill answer | No | Yes | Textbox with ckeditor plugin | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | View post | View post | N/A | Transfer to question page |
| 4 | Merge checked question | Go to Create new merged question | N/A | Transfer to Create new merged question page |
| 6 | Submit your answer | Submit answer to checked question | N/A | Reload Merge question page |

#### Create new merged question

****

**Figure 88: Create new merged question interface**

Fields, Buttons/Hyperlinks are referenced to “Create post” (5.2.4) with differences:

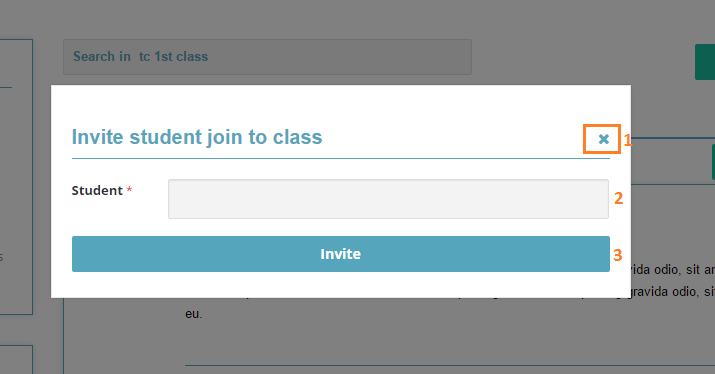
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Reused question | Show list of selected merged question | Yes | Yes | Table | String | N/A |
| 3 | Selected question | Select one reused question | No | Yes | Radio button | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Post view | Display question title, click to view question | N/A | Open question page in new tab |

#### Invite student popup

****

**Figure 89: Invite student popup interface**

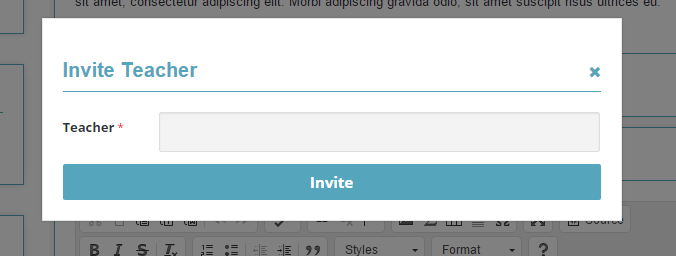
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 2 | Student | Fill student name | No | Yes | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Close | Close popup | N/A | Popup is hidden from screen |
| 3 | Invite | Invite student | N/A | Popup is hidden from screen |

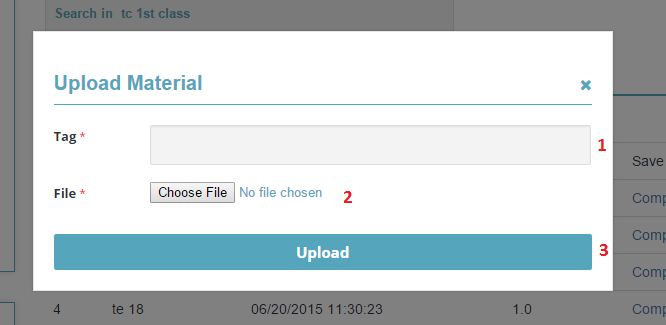
#### Invite teacher popup

****

**Figure 90: Invite teacher popup interface**

Fields, Buttons/Hyperlinks are referenced to “Invite student popup” (5.2.11)

#### Upload material popup

****

**Figure 91: Upload material popup interface**

Fields, Buttons/Hyperlinks are referenced to “Invite student popup” (5.2.11) with differences:

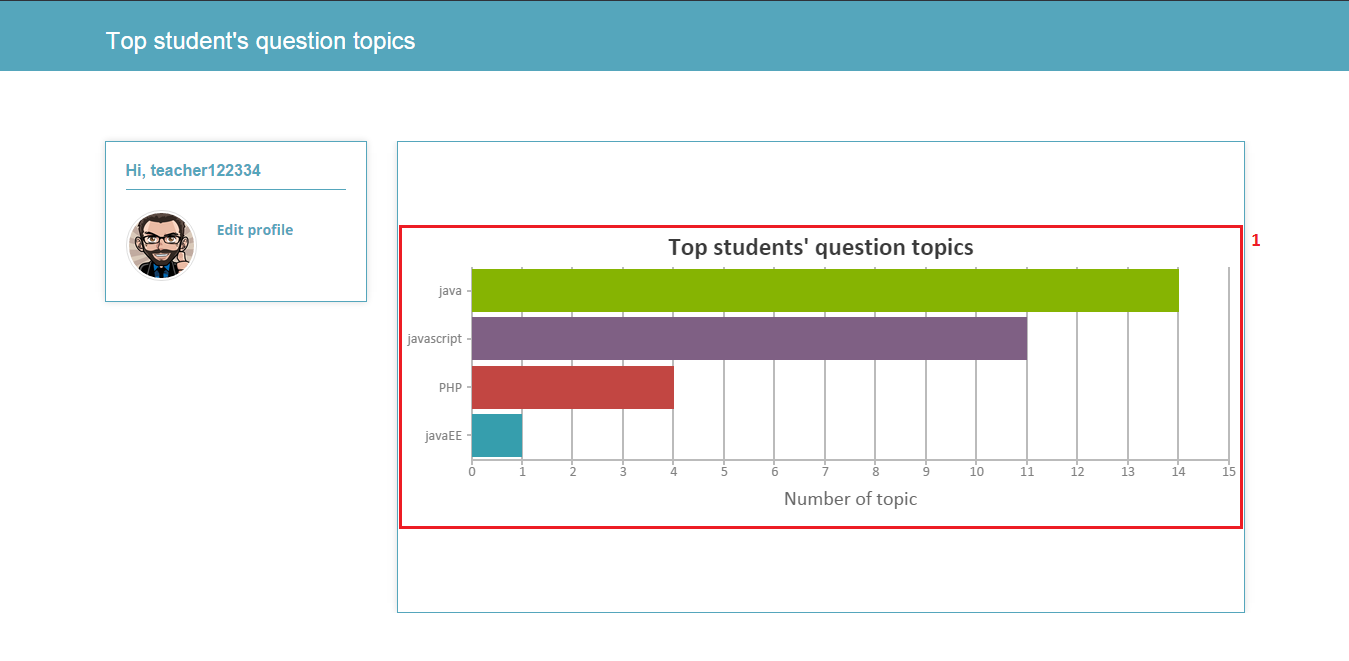
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Tag | Fill tag | No | Yes | Textbox | String | N/A |
| 2 | File | Choose file from PC | No | Yes | File input | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Upload | Upload material | N/A | Popup is hidden from screen |

#### Tracking

****

**Figure 92: Tracking interface**

**Fields**

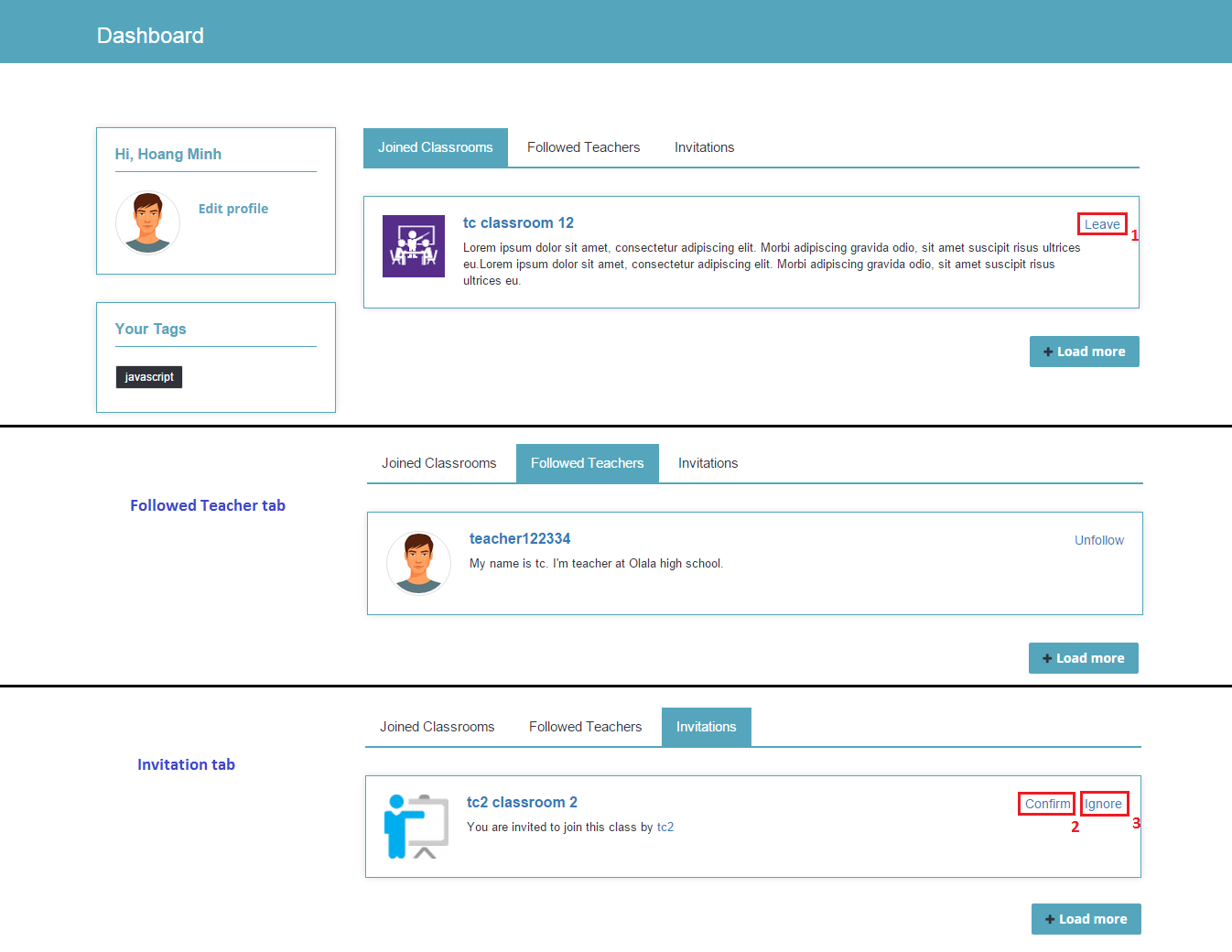
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Top student’s question topic | Show top topic that student usually ask | Yes | Yes | Canvas JS | N/A | N/A |

### Student Interface Design

#### News feed

Interface is referenced to teacher “News feed” (5.2.1)

#### Dashboard

****

**Figure 93: Student Dashboard interface**

Buttons/Hyperlinks are referenced to teacher “Dashboard” (5.2.2) with differences:

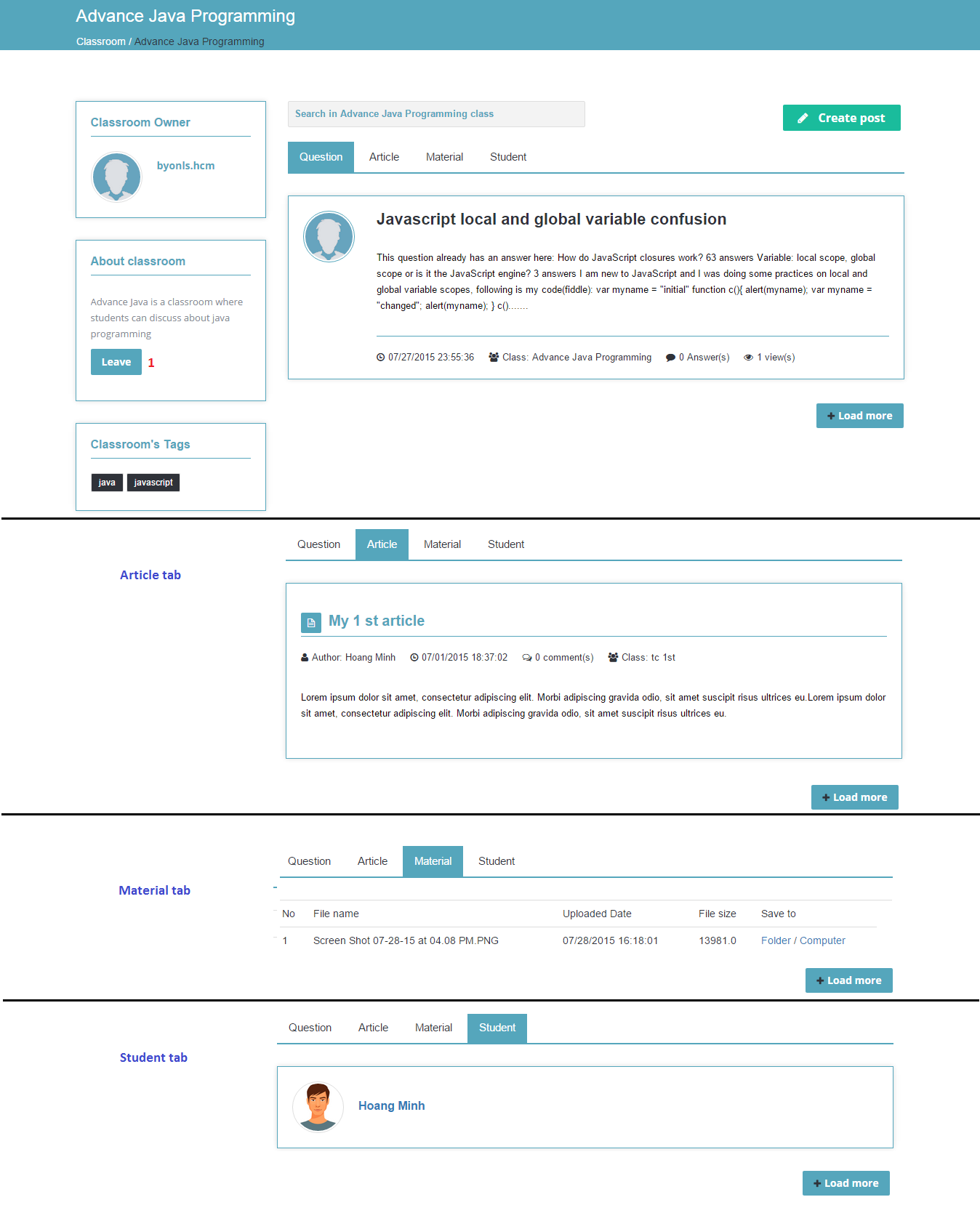
**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Leave | Leave classroom, click to display confirm popup | N/A | Show confirm popup |
| 2 | Confirm | Confirm invitation to join classroom | N/A | Show confirmed dialog |
| 3 | Ignore | Ignore invitation to join classroom | N/A | Show ignored dialog |

#### Create post

Interface is referenced to teacher “Create post” (5.2.4)

#### Classroom view

****

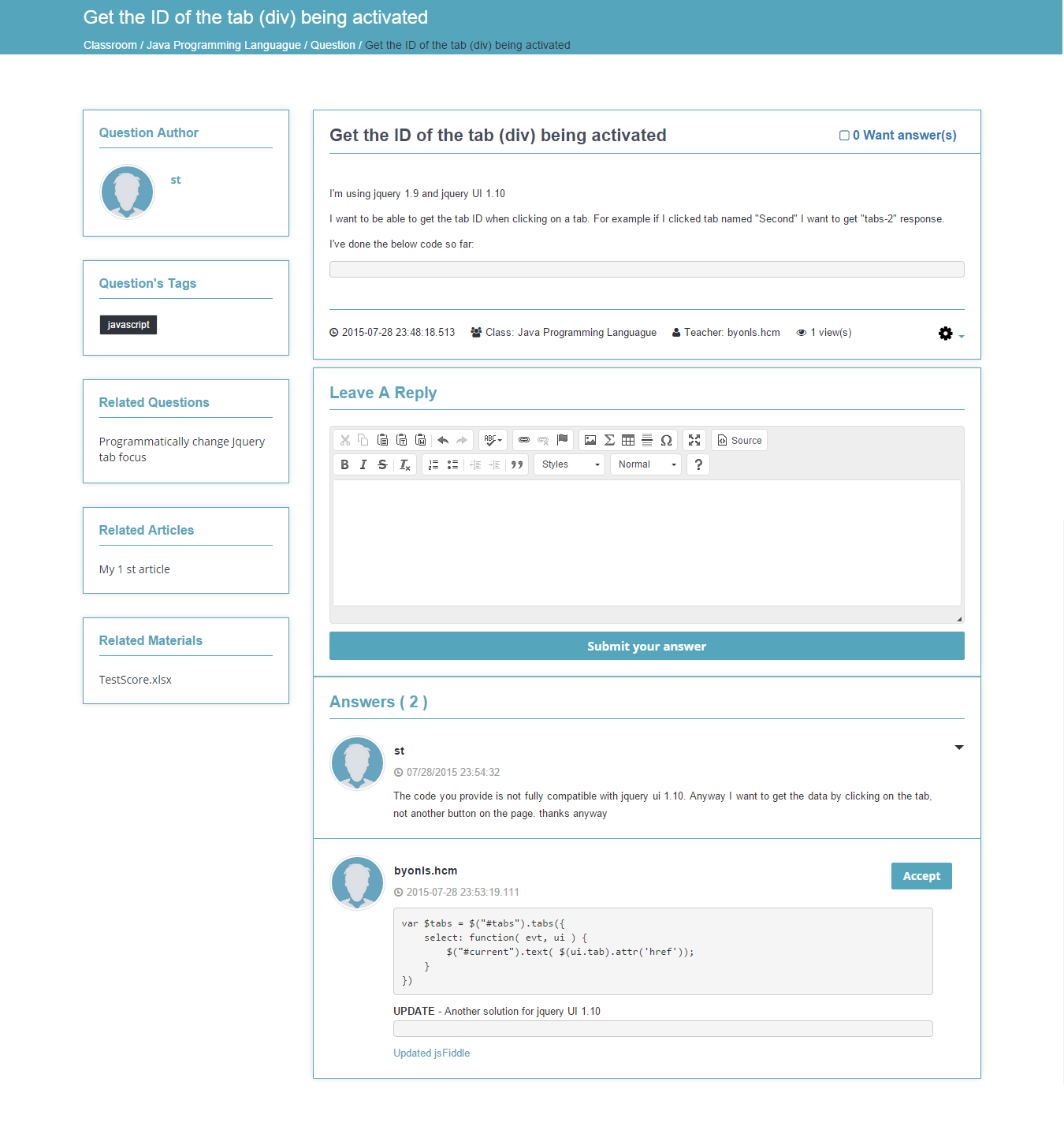
**Figure 94: Student Classroom view interface**

Buttons/Hyperlinks are referenced to teacher “Classroom view” (5.2.5) with differences:

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Leave | Leave classroom, click to display confirm popup. 4 states:  1: “Join”: click to send request 2: “Request sent”: after clicking “Join” 3: “Accept invitation”: after teacher’s invited 4: “Leave” | N/A | Show confirm popup |

#### Post view

****

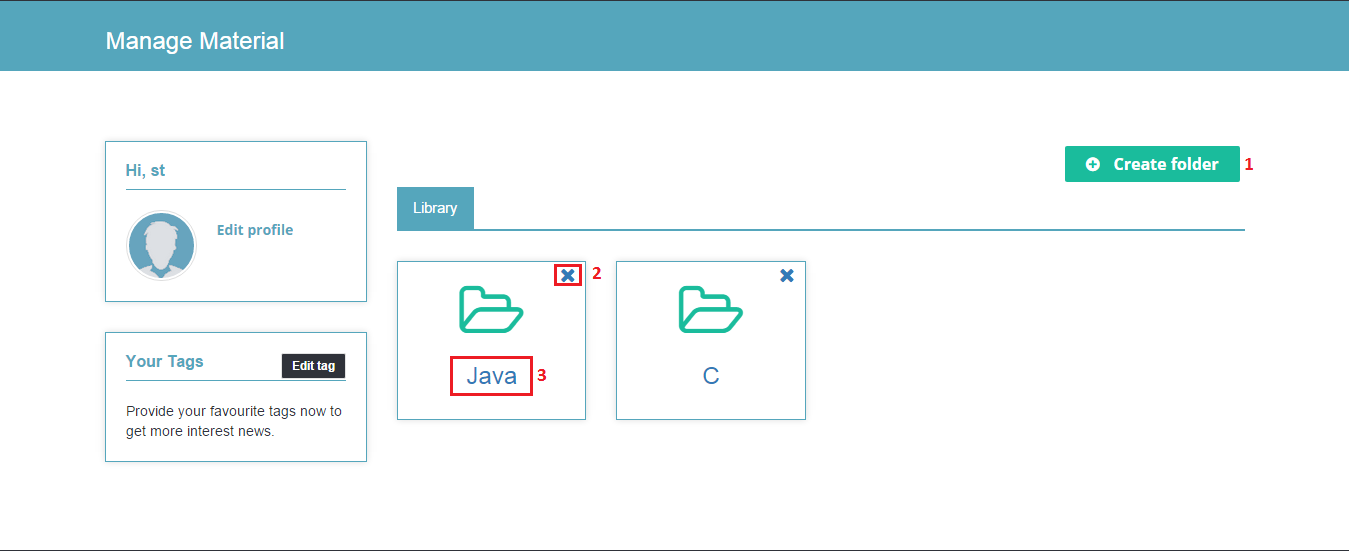
**Figure 95: Student Post view interface**

Buttons/Hyperlinks are referenced to teacher “Post view” (5.2.6)

#### Edit post

Interface is referenced to teacher “Edit post” (5.2.8)

#### Manage material

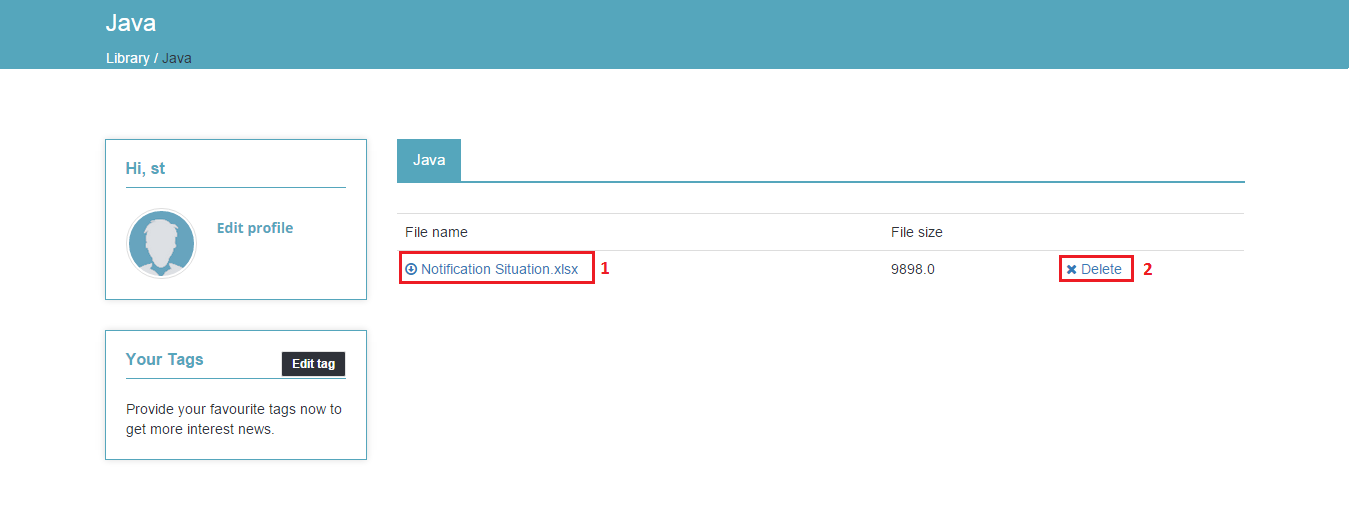
****

**Figure 96: Manage material interface**

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Create folder | View Create folder popup | N/A | Show Create folder popup |
| 2 | Delete folder | Delete folder and all folder’s items, click to show confirm popup | N/A | Show confirm popup |
| 3 | View folder | View folder | N/A | Transfer to Folder view page |

#### Folder view

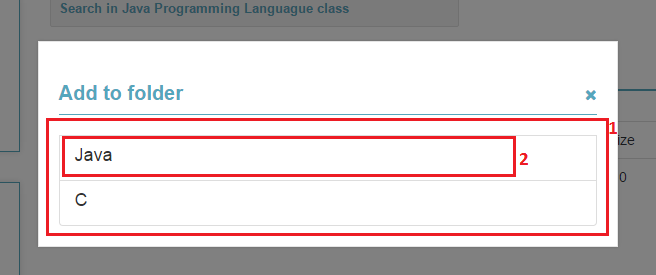
****

**Figure 97: Folder view interface**

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Download material | Click to save material to PC | N/A | N/A |
| 2 | Delete material | Delete material, click to show confirm popup | N/A | Show confirm popup |

#### Add to folder popup

****

**Figure 98: Add to folder interface**

Fields, Buttons/Hyperlinks are referenced to “Invite student popup” (5.2.11) with differences:

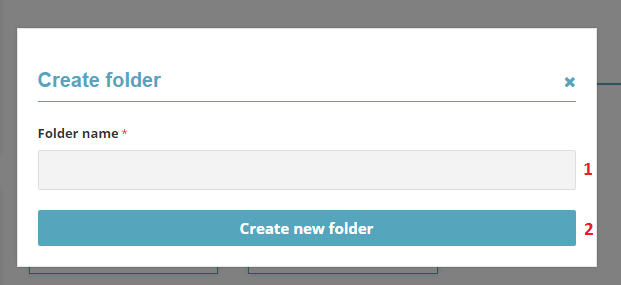
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Folder list | Show list of student’s folders | Yes | Yes | Table | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Add to folder | Select folder to add material | N/A | Popup is hidden from screen |

#### Create folder popup

****

**Figure 99: Add to folder interface**

Fields, Buttons/Hyperlinks are referenced to “Invite student popup” (5.2.11) with differences:

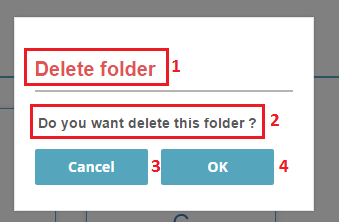
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Folder name | Fill folder name | No | Yes | Textbox | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Create new folder | Create new folder | N/A | Popup is hidden from screen. New folder is displayed. |

### Confirm popup

****

**Figure 100: Confirm popup**

**Fields**

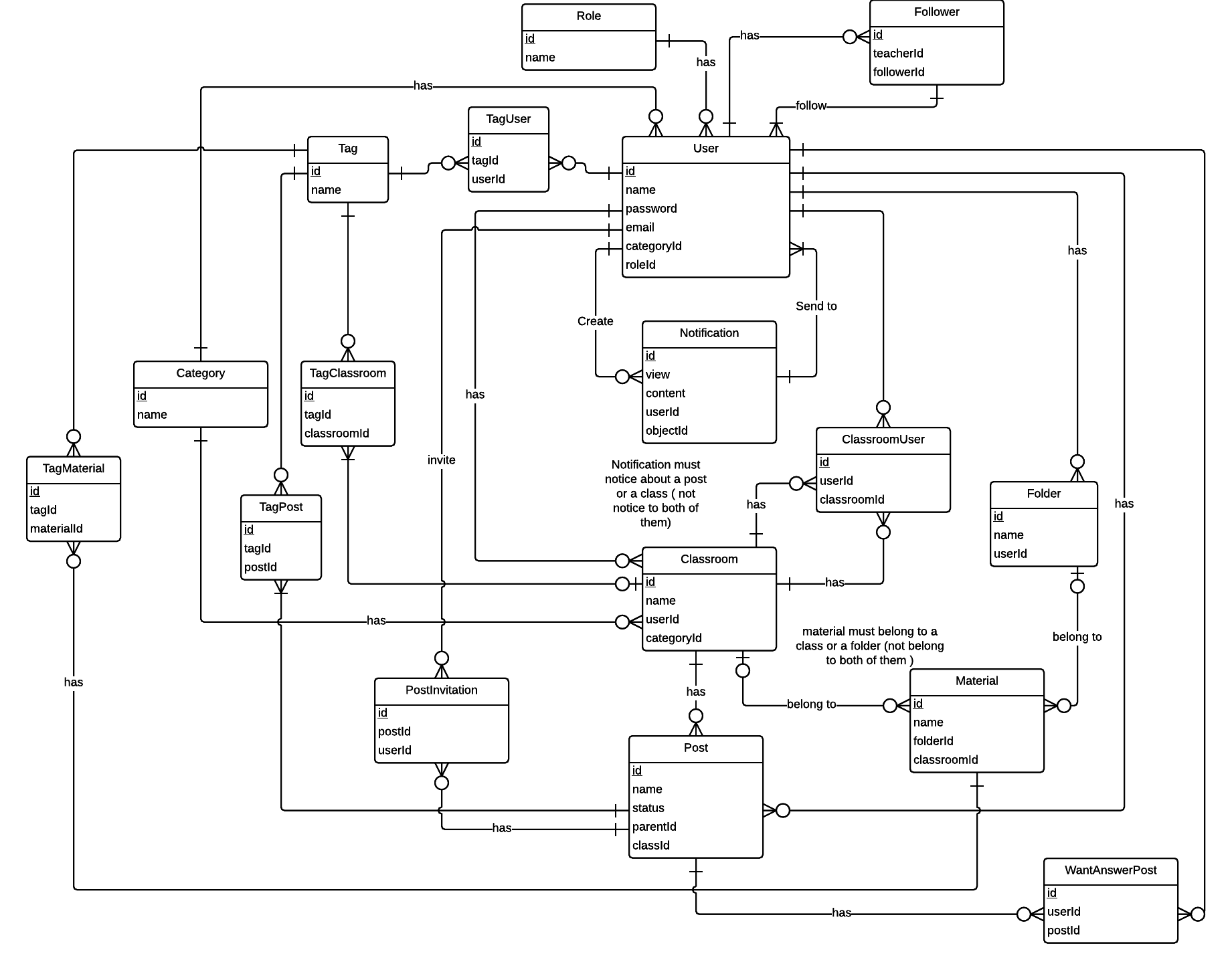
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Popup title | Show title base on delete/remove cases | Yes | Yes | Label | String | N/A |
| 2 | Popup message | Show message base on delete/remove cases | Yes | Yes | Label | String | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Cancel | Cancel removing/deleting action | N/A | Popup is hidden from screen. |
| 4 | OK | Confirm removing/deleting action | N/A | Popup is hidden from screen. |

## Database Design

### Logical Diagram



**Figure 101: Logical diagram**

### Data Dictionary

|  |  |
| --- | --- |
| **Logical Data dictionary: describe content of all entities** | |
| Entity Name | Description |
| Role | Describe all roles in system |
| User | Describe all users in system |
| Classroom | Describe all classes in system |
| Post | Describe all posts in system |
| Tag | Describe all tags in system |
| Category | Describe all categories in system |
| Notification | Describe all notifications in system |
| Material | Describe all materials in system |
| Folder | Describe all folders in system |
| Classroom\_user | Describe status User of one classroom in systems |
| TagClassroom | Describe tags of one classroom in systems |
| TagPost | Describe tags of one Post in systems |
| TagUser | Describe tags of one User in systems |
| TagMaterial | Describe tags of one Material in systems |
| PostInvitation | Describe Users was invite to answer a post |
| WantAnswerPost | Describe Users want to answer Post |
| Follower | Describe relation follower of user |

Table 10 Logical Dictionary

## Algorithms

### Calculating question’s score

#### Definition

Calculating question’s score is the way to evaluate question’s importance. Base on question scores, system knows the order to show questions to user.

#### Definition Problem

It’s important to choose what question will be shown first and what question will be shown after to user. An algorithm that evaluates the user’s interest with questions (question’s scores) is needed.

#### Solution

Question’s score (S) is calculates base on the following attributes:

* *wa*: Number of users that want the answer for the question
* *v*: Number of users that view the question
* *a*: Number of answers for the question
* *a’*: Decrease value: a’=0 if the question hasn’t yet have accepted answer, a’=1 if the question has already have accepted answer
* *h*: distance between current time and last interaction (last edited time of the question or last edited time of answer of the question) time of question by hour

Because of the difference of attributes’ importance, each attribute have different weight: *wa* is 3, *v* is 2 and others is 1.

Define the formula to calculate question’s score:

Example: Given the question that has 30 users that want the answer, 100 views, 5 answers and lasted answer is posted at 5 hours ago. This question already has accepted answer.

Because the question has accepted answer, a’ =1. So question’s score:

#### Complexity

The complexity of algorithms is O(1)

### Calculate similar score of two String

#### Define Problem

* Given two strings. Calculate their matching percent

#### Requirement

* two strings

#### Solution

* + The solution follows “Jaccard index”
  + If a string contains many words, break it into a list of words.
  + For each word, we find out how many adjacent character pairs are contained in it.
  + Create a function pairs(s) which returns a list of adjacent character pairs of string s.
  + Then, we use below formula to calculate matching percent.

#### Example

Calculate the matching percent of 2 strings: *KHANG, KHUONG.*

Break string into list of adjacent character pairs:

KHANG 🡪 = {KH, HA, AN, NG}

KHUONG 🡪 = {KH, HU, UO, ON, NG}

Find number interaction of 2 Strings

Find number union of 2 Strings

Calculate matching percentage

### Arrange strings by similar score

#### Define Problem

* Given list of strings. Arrange them by similar score.

#### Requirement

* List of strings.

#### Solution

* Calculate total similar score of each element in list with other elements.
* Order list by element’s total score.

#### Example

* + - Given List of strings :
    - List A = {What is abstraction in OOP,

How to convert String to Int,

What is inheritance in OOP,

How to parse String to Int}

A1 = What is abstraction in OOP

A2 = How to convert String to Int

A3 = What is inheritance in OOP

A4 = How to parse String to Int

* + - Expected Result :

List B = {How to parse String to Int ,

How to convert String to Int ,

What is abstraction in OOP ,

What is inheritance in OOP }

* + - Calculate total similar score of First Element with other elements :
      * Total(A1) =
    - We do the same thing with other elements. We will get these results :
      * Total(A2) = 0.65
      * Total(A3)= 0.43
      * Total(A4) = 0.689
    - Order list decreasingly by total similar score we get:

List B = {How to parse String to Int ,

How to convert String to Int ,

What is abstraction in OOP ,

What is inheritance in OOP }

# Report No. 5 System Implementation & Test

## Introduction

### Overview

This section contains in detail of all necessary information about system implementation and testing of QAPE System includes all test plans, test cases, test results and test environment.

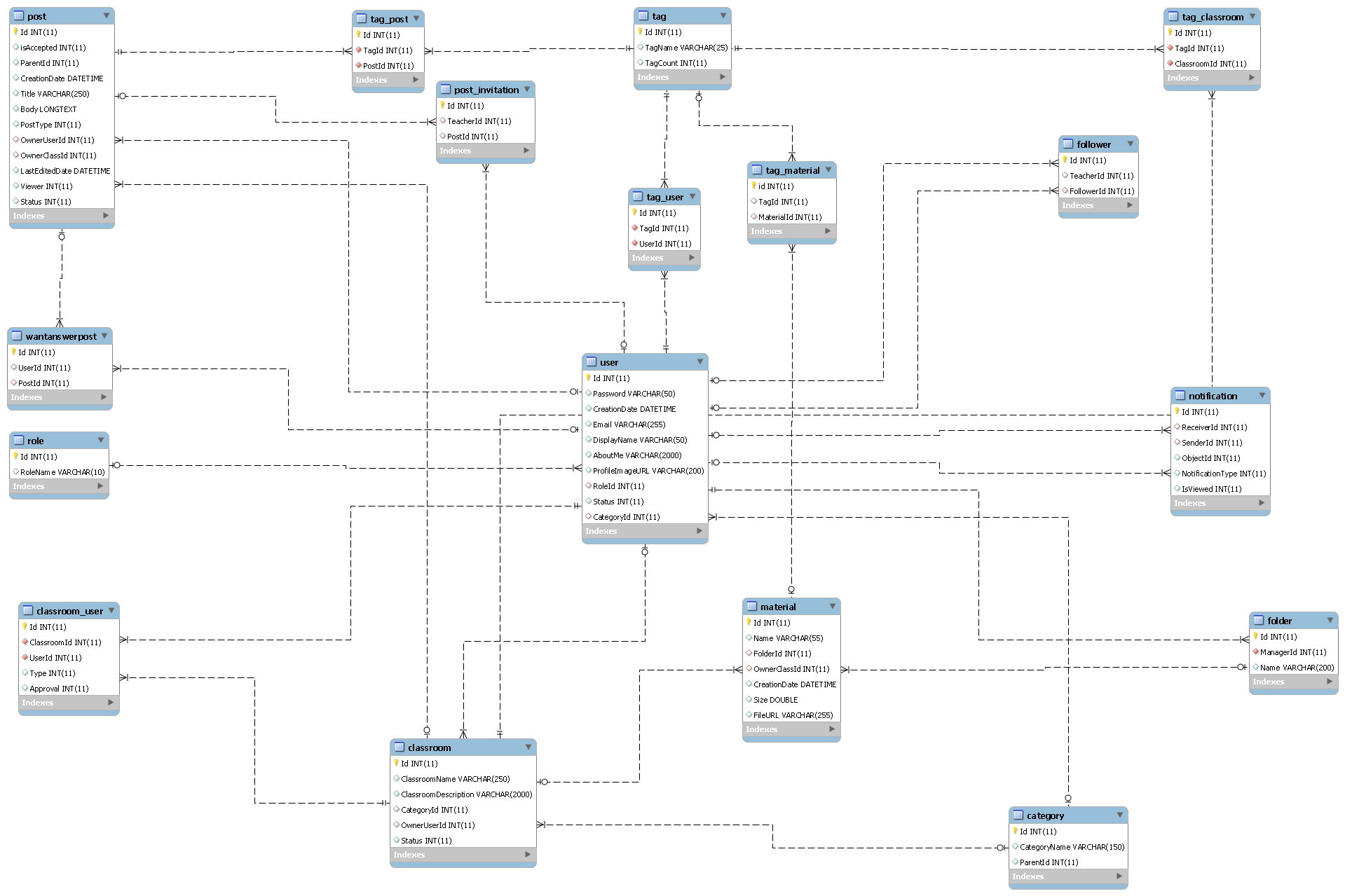
### Test Approach

- Goal: To test the whole system based on the core flow.

- Method: System testing, Black box testing.

## Database Relationship Diagram

### Physical Diagram



**Figure 102: Physical database diagram**

### Data Dictionary

|  |  |
| --- | --- |
| **Physical database dictionary: describe content of all physical database** | |
| Entity Name | Description |
| Role | Describe all roles in system |
| User | Describe all users in system |
| Classroom | Describe all classes in system |
| Post | Describe all posts in system |
| Tag | Describe all tags in system |
| Category | Describe all categories in system |
| Notification | Describe all notifications in system |
| Material | Describe all materials in system |
| Folder | Describe all folders in system |
| Classroom\_User | Describe status User of one classroom in systems |
| Tag\_Classroom | Describe tags of one classroom in systems |
| Tag\_Post | Describe tags of one Post in systems |
| Tag\_User | Describe tags of one User in systems |
| Tag\_Material | Describe tags of one Material in systems |
| Post\_Invitation | Describe Users was invite to answer a post |
| WantAnswerPost | Describe Users want to answer Post |
| Follower | Describe relation follower of user |

Table 11 Physical database Dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity name** | **Attributes** | **Description** | **Domain** | **Null** |
| Role | Id {PK} | Unique identifier | Int | No |
| RoleName | Role name | Varchar(12) | No |
| Category | Id {PK} | Unique identifier, auto increment | Int | No |
| CategoryName | Category name | Varchar(150) | No |
| ParentId | Parent identifier of a category | Int | Yes |
| User | Id {PK} | Unique identifier, auto increment | Int | No |
| Password | Password of User | Int | No |
| CreationDate | Creation Date | DateTime | No |
| Email | Email | Varchar(255) | No |
| DisplayName | Name | Varchar(50) | No |
| AboutMe | About user | Varchar(2000) | Yes |
| ProfileImageUrl | Avatar url | Varchar(200) | No |
| RoleId {FK} | Role of user | Int | No |
| Status | Status of user | Int | No |
| CategoryId {FK} | Category of user | Int | No |
| Classroom | Id {PK} | Unique identifier, auto increment | Int | No |
| ClassroomName | Classroom name | Varchar(250) | No |
| ClassroomDescription | Classroom description | Varchar(250) | No |
| CategoryId {FK} | Id of category of classroom | Int | No |
| OwnerUserId {PK} | Id of user who create classroom | Int | No |
| Status | Status of classroom | Int | No |
| Post | Id {PK} | Unique identifier, auto increment | Int | No |
| isAccepted | Post was accepted or not | Int | No |
| ParentId | Id of Parent post | Int | Yes |
| CreationDate | Creation date | DateTime | No |
| Title | Title of post | Varchar(250) | No |
| Body | Body of post | LongText | No |
| PostType | Type of post | Int | No |
| OwnerUserId {FK} | Id of user who created the post | Int | No |
| OwnerClassId {FK} | Id of classroom that post was belong to | Int | No |
|  | LastEditedDate | Last edit time | DateTime | No |
| Viewer | Number viewer | Int | No |
| Status | Status of Post | Int | No |
| Tag | Id {PK} | Unique identifier, auto increment | Int | No |
| TagName | Tag name | Varchar(25) | No |
| TagCount | Tag count |  |  |
| Notification | Id {PK} | Unique identifier, auto increment | Int | No |
| ReceiverId {FK} | Id of receiver | Int | No |
| SenderId {FK} | Id of sender | Int | No |
| ObjectId | Id of Classroom or Post was noticed | Int | No |
| NotificationType | Type of notification | Int | No |
| IsViewed | Notification is viewed or not | Int | No |
| Material | Id {PK} | Unique identifier, auto increment | Int | No |
| Name | Material name | Int | No |
| FolderId {FK} | Id of Folder contain material | Int | Yes |
| OwnerClassId {FK} | Id of Class contain material | Int | Yes |
| CreationDate | Upload date | DateTime | No |
| Size | Size of material | Double | No |
| FileUrl | url of material | Varchar(250) | No |
| Folder | Id {PK} | Unique identifier, auto increment | Int | No |
| ManagerId | Id of owner folder | Int | No |
| Name | Name of folder | Varchar(200) | No |
| Classroom\_User | Id {PK} | Unique identifier, auto increment | Int | No |
| ClassroomId {FK} | Id of classroom that user take part in | Int | No |
| User {FK} | Id of user take part in classroom | Int | No |
| Status | Status of user in classroom | Int | No |
| Tag\_User | Id {PK} | Unique identifier, auto increment | Int | No |
| TagId {FK} | Id of tag | Int | No |
| UserId {FK} | Id of User | Int | No |
| Tag\_Post | Id {PK} | Unique identifier, auto increment | Int | No |
| TagId {FK} | Id of tag | Int | No |
| PostId {FK} | Id of Post | Int | No |
| Tag\_Classroom | Id {PK} | Unique identifier, auto increment | Int | No |
| TagId {FK} | Id of tag | Int | No |
| ClassroomId {FK} | Id of Classroom | Int | No |
| Tag\_Material | Id {PK} | Unique identifier, auto increment | Int | No |
| TagId {FK} | Id of tag | Int | No |
| MaterialId {FK} | Id of Material | Int | No |
| Post\_Invitation | Id {PK} | Unique identifier, auto increment | Int | No |
| TeacherId {FK} | Id of invited teacher | Int | No |
| PostId {FK} | Id of Post | Int | No |
| WantAnswerPost | Id {PK} | Unique identifier, auto increment | Int | No |
| PostId {FK} | Id of Post | Int | No |
| UserId {FK} | Id of User want answer | Int | No |

Table 12 Attribute Data Dictionary

## Performance Measurement

N/A

## Test Plan

The purpose of this section is to verify and ensure that QAPE meets its design specification and other requirements from user. Based on limited team member, the following part will describe which features to be tested and which will not.

### Features to be tested

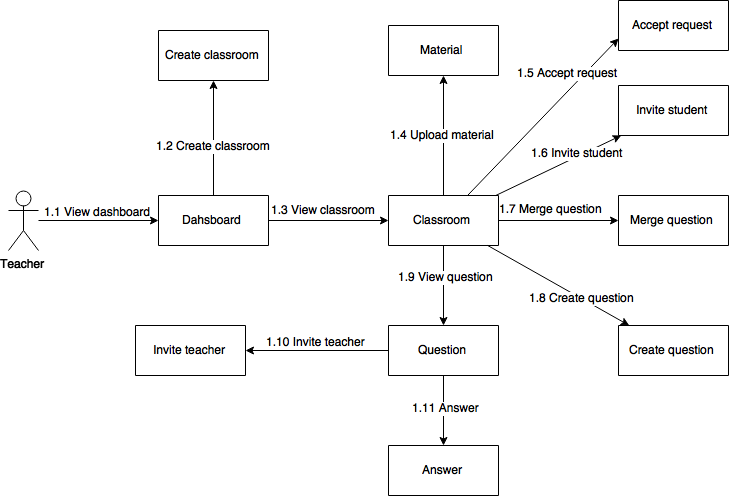
Teacher: Create classroom, create post (question/article), upload material, invite student, invite teacher, accept request and merge question

Student: Join classroom, create post (question/article), create folder, add material to folder, and accept answer

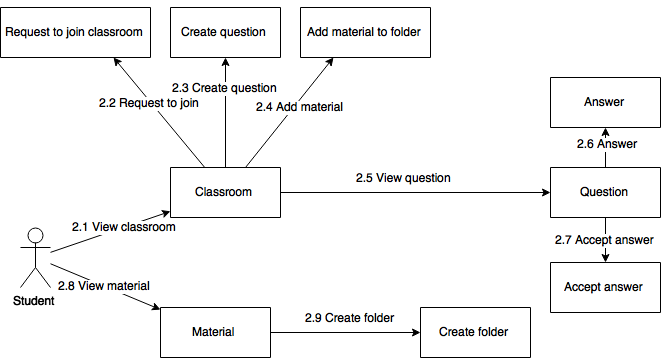
### Features not to be tested

Login, register, logout, search, close classroom, close post, delete post, delete folder, delete material, track question.

## System Testing Test Case



**Figure 103: Teacher core flow**



**Figure 104: Student core flow**

### Teacher core flow

#### View dashboard

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| VD01 | Teacher views dashboard | 1. After login, click on “Dashboard” on menu (teacher has at least 1 classroom or 1 followed teacher or 1 invitation) | - Show “Dashboard” page, include three tabs (Owned classroom, Followed Teacher and Invitation) and “Create class” button |  | Pass | 30/07/2015 |  |
| VD02 | Teacher views dashboard with no item | 1. After login, click on Dashboard” on menu (teacher has no classroom, followed teacher and invitation) | - Show “Dashboard” page, include the suggestion to create first classroom and Create class button |  | Pass | 30/07/2015 |  |

#### Create classroom

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| CC01 | Teacher creates classroom with all valid fields | 1. Go to dashboard, click on “Create class” button.  2. Input valid Class name, Tag, Specialty, Invite student and Class’s description. 3. Click “Create class” button | - After step 1, show “Create classroom” page with required fields have “\*”.     - After step 3, navigate to Classroom page and show created classroom |  | Pass | 30/07/2015 |  |
| CC02 | Teachers create classroom with no field inputted | 1. Go to dashboard, click on “Create class” button.   2. Click “Create class” button | - After step 1, show “Create classroom” page with required fields have “\*”. - After step 2, show “Please provide the class name”, “Please provide at least one tag”, “Please select specialty”, “Please provide the description” error message |  | Pass | 30/07/2015 |  |
| CC03 | Teachers create classroom with no “Invite student” inputted | 1. Go to dashboard, click on “Create class” button.  2. Input valid Class name, Tag, Specialty and Class’s description. 3. Click “Create class” button | - After step 1, show “Create classroom” page with required fields have “\*”.    - After step 3, navigate to Classroom page and show created classroom |  | Pass | 30/07/2015 |  |

#### View classroom

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| VC01 | Teacher views owned opened classroom | 1. Go to dashboard, click on “Owned classroom” tab 2. Click on any opened classroom. | - After step 1, show list of owned classrooms. - After step 2, show Classroom page, includes: 5 tabs(Question, Article, Material, Join Request, Student), Invite student button, Create post button, Setting icon |  | Pass | 30/07/2015 |  |
| VC02 | Teacher views another one’s classroom | 1. Go to another teacher’s classroom | - Show Classroom page and classroom with 4 tabs(Question, Article, Material, Student), no “Create post” and “Invite student” button |  | Pass | 30/07/2015 |  |

#### Upload material

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| UM01 | Teacher upload material with all valid fields | 1. Go to owned opened classroom, click tab Material 2. Click “Upload” button  3. Select file, input tag and click “Upload” button | - After step 1, show list of materials and “Upload” button - After step 2, show Upload material popup -After step 3, hide popup, uploaded material is shown in list |  | Pass | 31/07/2015 |  |
| UM02 | Teacher upload material with no tag | 1. Go to owned opened classroom, click tab Material 2. Click “Upload” button  3. Select file and click “Upload” button | - After step 1, show list of materials and “Upload” button - After step 2, show Upload material popup -After step 3, show “Please provide at least one tag and max 5 tags” error message |  | Pass | 31/07/2015 |  |

#### Accept Request

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| AR01 | Teacher accept student’s request | 1. Go to owned opened classroom, click tab Join Request 2. Click “Confirm” on any request in list | - After step 1, show list of requests  - After step 2, show “You have confirm the request of <student’s name>” message, “Student” tab show student’s name have accepted |  | Pass | 31/07/2015 |  |

#### Invite student

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| IS01 | Teacher invites student with valid inputted student’s name | 1. Go to owned opened classroom, click “Invite” button 2. Input student name and choose student from suggestion list 3. Click “Invite” button | - After step 1, show Invite student popup     - After step 3, popup is hidden. Login with student’s account has invited will receive invitation. |  | Pass | 31/07/2015 |  |
| IS02 | Teacher invites student with no student’s name | 1. Go to owned opened classroom, click “Invite” button 2. Click “Invite” button | - After step 1, show Invite student popup  - After step 2, show “Please provide student’s name” error message |  | Pass | 31/07/2015 |  |

#### Merge question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| MQ01 | Teacher merges questions with all valid fields | 1. Go to owned opened classroom, click tab Question 2. Click “Merge questions” button 3. Select at least 2 questions and click “Merge checked questions” button 4. Input valid Reused Question, Title, Tag and Details 5. Click “Save” button | - After step 1, show list of questions with “Merge questions” button - After step 2, show “merge question” page - After step 3, show “Create new merged question” page     - After step 5, show question page with main merged question info, other merged questions is closed and left a comment with reference to the main one |  | Pass | 31/07/2015 |  |
| MQ02 | Teacher merges questions with no selected question | 1. Go to owned opened classroom, click tab Question 2. Click “Merge questions” button 3. Click “Merge checked questions” button | - After step 1, show list of questions with “Merge questions” button - After step 2, show “merge question” page - After step 3, show “Please select at least two questions” error message |  | Pass | 31/07/2015 |  |
| MQ03 | Teacher merges questions with one selected question | 1. Go to owned opened classroom, click tab Question 2. Click “Merge questions” button 3. Select one question and click “Merge checked questions” button | - After step 1, show list of questions with “Merge questions” button - After step 2, show “merge question” page - After step 3, show “Please select at least two questions” error message |  | Pass | 31/07/2015 |  |

#### Create question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| CQ01 | Teacher creates question with all valid fields | 1. Go to owned opened classroom, click “Create post” button 2. Input valid Title, Tag, Type (select Question) and Details 3. Click “Publish your post” button | - After step 1, show “Create post” page     - After step 3, show question page with created question |  | Pass | 31/07/2015 |  |
| CQ02 | Teacher creates question with no inputted fields | 1. Go to owned opened classroom, click “Create post” button 2. Click “Publish your post” button | - After step 1, show “Create post” page  - After step 2, show “Please provide title”, “Please provide at least one tag”, “Please choose question or article”, “Please provide the details” error messages |  | Pass | 31/07/2015 |  |

#### View question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| VQ01 | Teacher views question belongs to owned classroom | 1. Go to owned classroom, click "Question” tab 2. Click on any opened question | - After step 1, show list of question. - After step 2, show Question page with Setting icon, Ask teacher button and answer text-area |  | Pass | 31/07/2015 |  |
| VQ02 | Teacher views question belongs to not owned classroom | 1. Go to not owned classroom, click "Question” tab 2. Click on any opened question | - After step 1, show list of question. - After step 2, show Question page with no Setting icon, no Ask teacher button and no answer text-area |  | Pass | 31/07/2015 |  |

#### Invite teacher

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| IT01 | Teacher invites others to answer opened question belongs to owned classroom | 1. Go to opened question in owned classroom, click “Ask teacher” button 2. Input teacher name and choose teacher from suggestion list 3. Click “Invite” button | - After step 1, show Invite teacher popup     - After step 3, popup is hidden. Login with teacher’s account has invited will receive invitation. |  | Pass | 31/07/2015 |  |
| IT02 | Teacher invites others to answer a question belongs to owned classroom with no teacher’s name | 1. Go to opened question in owned classroom, click “Ask teacher” button. 2. Click “Invite” button | - After step 1, show Invite teacher popup  - After step 2, show “Please provide teacher’s name” error message |  | Pass | 31/07/2015 |  |

#### Answer question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| AQ01 | Teacher answers opened question belongs to owned classroom | 1. Go to opened question in owned classroom, input answer 2. Click “Submit your answer” button | - After step 1, show question page  - After step 2, show submitted answer in real time |  | Pass | 31/07/2015 |  |
| AQ02 | Teacher answers opened question belongs to owned classroom with no inputted answer | 1. Go to opened question in owned classroom  2. Click “Submit your answer” button | - After step 1, show question page - After step 2, show “The answer must be at least 30 characters long” error message |  | Pass | 31/07/2015 |  |

### Student core flow

#### View classroom

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| VC03 | Student views joined opened classroom | 1. Go to dashboard, click on “Joined classroom” tab 2. Click on any opened classroom. | - After step 1, show list of joined classrooms. - After step 2, show Classroom page, includes: 4 tabs(Question, Article, Material, Student), “Create post” button, “Leave” classroom button |  | Pass | 01/08/2015 |  |
| VC04 | Student views opened classroom that hasn’t joined and hasn’t been invited | 1. Go to opened classroom that hasn’t joined and hasn’t been invited | - Show Classroom page and classroom with 4 tabs(Question, Article, Material, Student), no “Create post” and “Join” button |  | Pass | 01/08/2015 |  |
| VC05 | Student views opened classroom that hasn’t joined and has been invited | 1. Go to opened classroom that hasn’t joined and has been invited | - Show Classroom page and classroom with 4 tabs(Question, Article, Material, Student), no “Create post” and “Accept invitation” button |  | Pass | 01/08/2015 |  |

#### Request to join classroom

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| RJ01 | Student request to join opened classroom | 1. Go to opened classroom that hasn’t joined and hasn’t been invited 2. Click “Join” button | - After step 1, show Classroom page  - After step 2, Hide “Join” button, show “Request sent”. Login with teacher’s account that owned classroom will receive the request |  | Pass | 01/08/2015 |  |
| RJ02 | Student request to join opened classroom that has been invited | 1. Go to opened classroom that hasn’t joined and has been invited 2. Click “Accept Invitation” button | After step 1, show Classroom page  - After step 2, hide “Accept Invitation” button, show “Leave” button. |  | Pass | 01/08/2015 |  |

#### Create question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| CQ03 | Student creates question with all valid fields | 1. Go to joined opened classroom, click “Create post” button 2. Input valid Title, Tag, Type (select Question) and Details 3. Click “Publish your post” button | - After step 1, show “Create post” page     - After step 3, show question page with created question |  | Pass | 01/08/2015 |  |
| CQ04 | Student creates question with no inputted fields | 1. Go to joined opened classroom, click “Create post” button 2. Click “Publish your post” button | - After step 1, show “Create post” page  - After step 2, show “Please provide title”, “Please provide at least one tag”, “Please choose question or article”, “Please provide the details” error messages |  | Pass | 01/08/2015 |  |

#### Add material to folder

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| AM01 | Student has at least one folder and add material to folder | 1. Go to joined opened classroom, click “Material” tab 2. Click “Folder” on any material 3. Select folder | - After step 1, show list of materials  - After step 2, show Add to folder popup - After step 3, popup is hidden. Go to Material page and select folder to view added material |  | Pass | 01/08/2015 |  |

#### View question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| VQ03 | Student views question belongs to joined classroom | 1. Go to joined classroom, click "Question” tab 2. Click on any opened question | - After step 1, show list of questions. - After step 2, show Question page with answer text-area |  | Pass | 01/08/2015 |  |
| VQ04 | Student views question belongs to not joined classroom | 1. Go to joined classroom, click "Question” tab 2. Click on any opened question | - After step 1, show list of question. - After step 2, show Question page with no answer text-area |  | Pass | 01/08/2015 |  |

#### Answer question

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| AQ03 | Student answers opened question belongs to joined classroom | 1. Go to opened question in joined classroom, input answer 2. Click “Submit your answer” button | - After step 1, show question page  - After step 2, show submitted answer in real time |  | Pass | 01/08/2015 |  |
| AQ04 | Student answers opened question belongs to joined classroom with no inputted answer | 1. Go to opened question in joined classroom  2. Click “Submit your answer” button | - After step 1, show question page - After step 2, show “The answer must be at least 30 characters long” error message |  | Pass | 01/08/2015 |  |

#### Accept answer

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| AA01 | Student accepts answer belongs to owned opened question | 1. Go to owned opened question  2. Click “Accept” button | - After step 1, show question page - After step 2, hide “Accept” button, show “Unaccept” button |  | Pass | 01/08/2015 |  |
| AA02 | Student accepts other answer belongs to owned opened question | 1. Go to owned opened question has accepted answer 2. Click “Accept” button on other answer | - After step 1, show question page  - After step 2, Hide “Accept” button of current answer, show “Unaccept” button. Hide “Unaccept” button of older accepted answer, show “Accept” button |  | Pass | 01/08/2015 |  |

#### View material

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| VM01 | Student views material | 1. Go to Material page  2. Select any folder in list | - After step 1, show list of folder. - After step 2, show list of material of folder |  | Pass | 01/08/2015 |  |

#### Create folder

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Inter-test Case Dependence** | **Result** | **Test Date** | **Note** |
| CF01 | Student creates folder with inputted folder name | 1. Go to Material page  2. Click “Create folder” button 3. Input folder name and click “Create folder” button | - After step 1, show list of folder. - After step 2, show Create folder popup - Hide popup and show created folder in list |  | Pass | 01/08/2015 |  |
| CF02 | Student creates folder with no inputted | 1. Go to Material page  2. Click “Create folder” button 3. Click “Create folder” button | - After step 1, show list of folder. - After step 2, show Create folder popup - Show “Please provide folder’s name” error message |  | Pass | 01/08/2015 |  |

# Report No. 6 Software User’s Manual

## Installation Guide

### Setting up environment at server side

The following software must be installed into the server machine:

#### Hardware requirements

* CPU Intel Xeon Quad Core 2.5GHz.
* 2GB of RAM.
* 10GB of hard disk.
* Wi-Fi Internet (> 2Mbps).

#### Software requirements

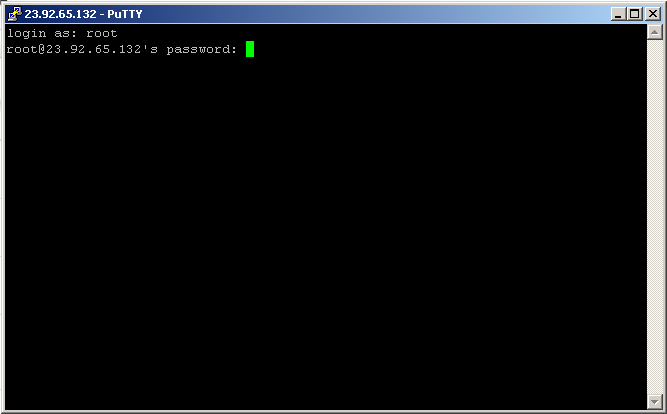
* Apache Tomcat 7.0.61 with JRE 7(7u79).
* Ubuntu 14.04
* MySQL (version 5.6.24).

### Deployment at server side

#### Set up system variable

Use PuTTY to connect to server

* **Login with username and password**

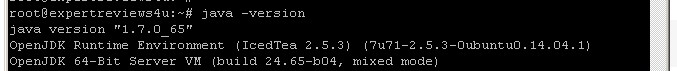


* **Update apt-get package lists (do not enter comma)**Enter “apt-get update”
* **Install JDK (do not enter comma)**

Enter “apt-get install default-jdk”

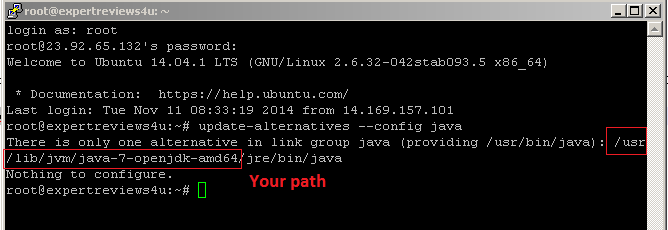
* **Check java version (do not enter comma)**

Enter “java –version”



* **Create environment variable (do not enter comma)**

Enter “update-alternatives --config java”



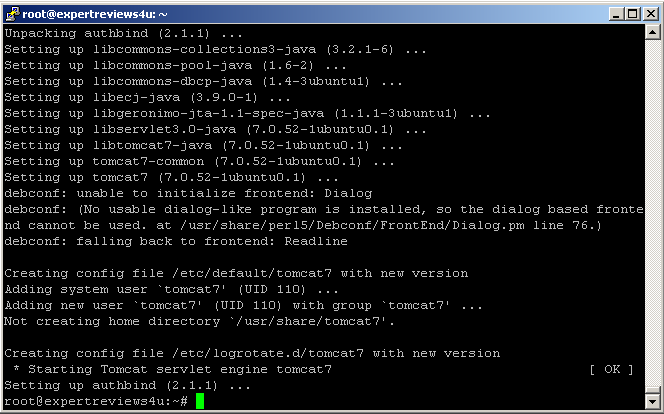
Enter “vi /etc/environment”

Enter “JAVA\_HOME=Your path”

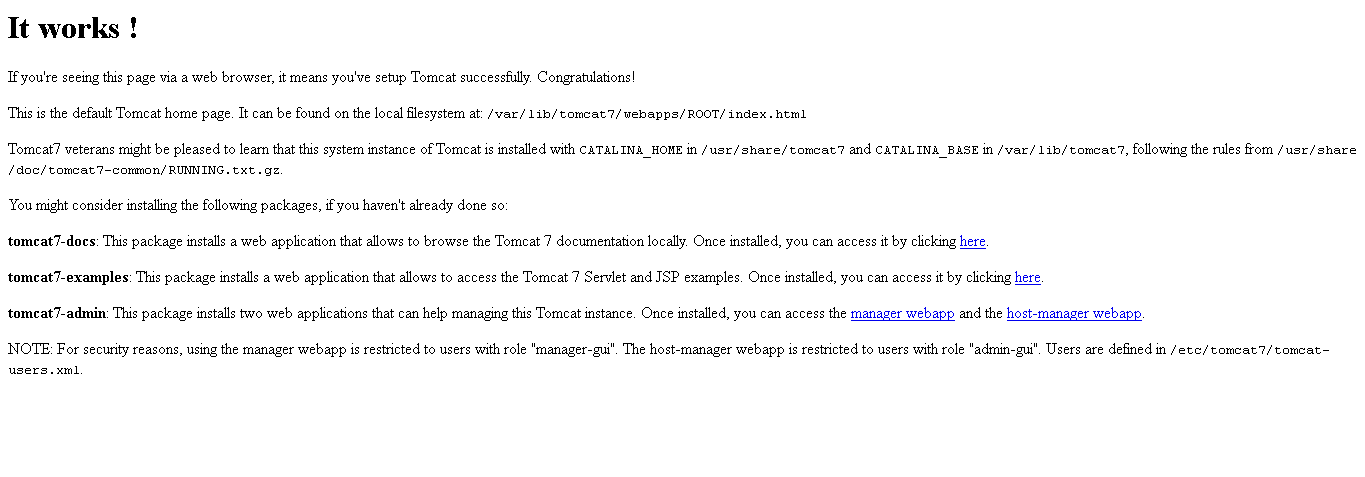
Reload file by entering “source /etc/environment”

Test by entering “echo $JAVA\_HOME”

* **Install Tomcat (do not enter comma)**Enter “apt-get install tomcat7”

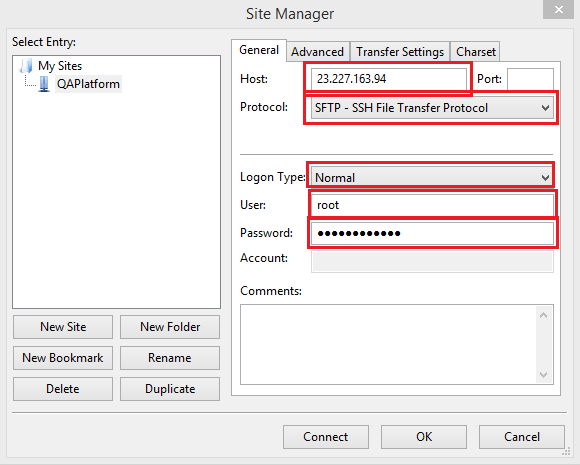
****

* **Check (do not enter comma)**Open browser and enter: “https://[your server IP address]:8080

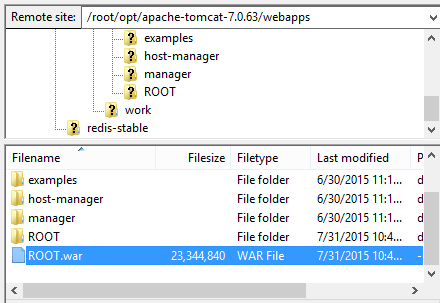
****

#### Deploy package

Use FileZilla to deploy package .war via SFTP-SSH. Go to File/Site Manager, input required fields and then click Connect



Copy .war package to /root/opt/apache-tomcat-7.0.63/webapps

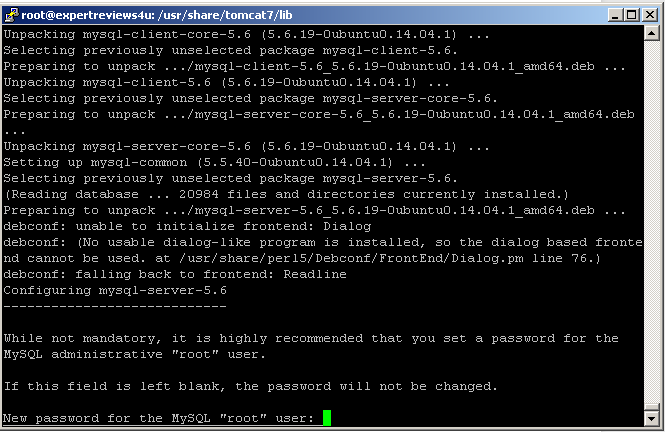


#### Install MySQL

* **Install MySQL and enter new root’s password (do not enter comma)**

Enter “apt-get install mysql-server-5.6”

Enter new root’s password



* **Fix error (if have | do not enter comma)**

Enter “apt-get --reinstall install bsdutils”

#### Install Redis

Follow “How To Install and Use Redis, by Etel Sverdlov, Aug 16 2012”.

### Setting up the environment at client side

#### Setting up for computer

##### Hardware requirements

Reference to recommend requirement of browsers using

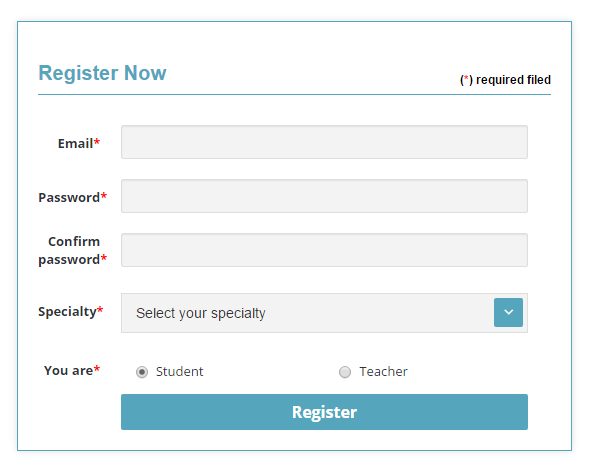
##### Software requirements

Installed at least one web browser:

* Google Chrome.
* Cốc Cốc.

## User Guide

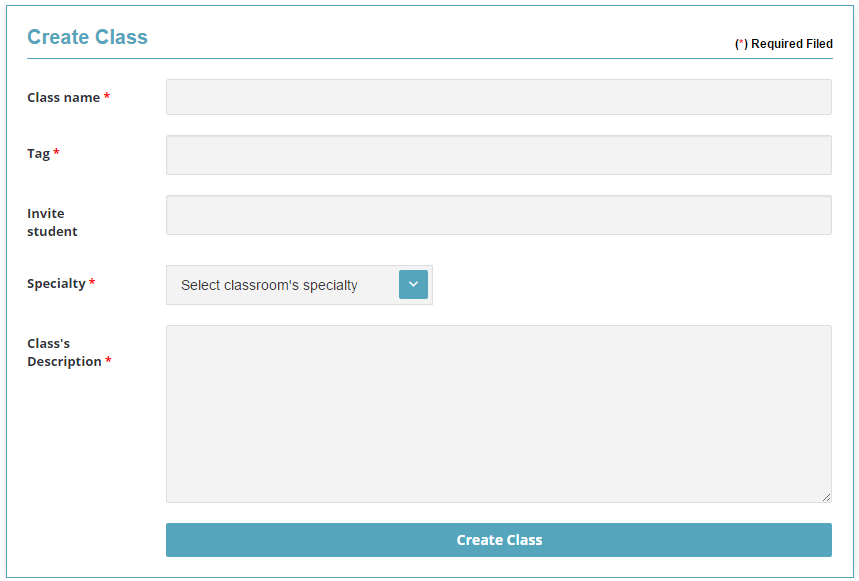
### Register



**Figure 105: Register form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User fills out all of required fields with personal information. |
| 2 | User clicks button “Register” to create an account. |

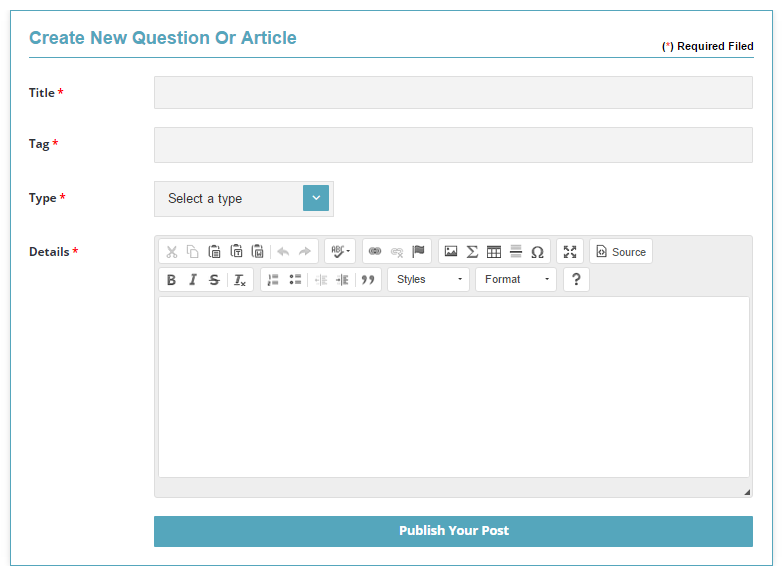
### Create classroom



**Figure 106: Create classroom form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User fills out all of required fields with classroom information. |
| 2 | User clicks button “Create Class” to create a classroom. |

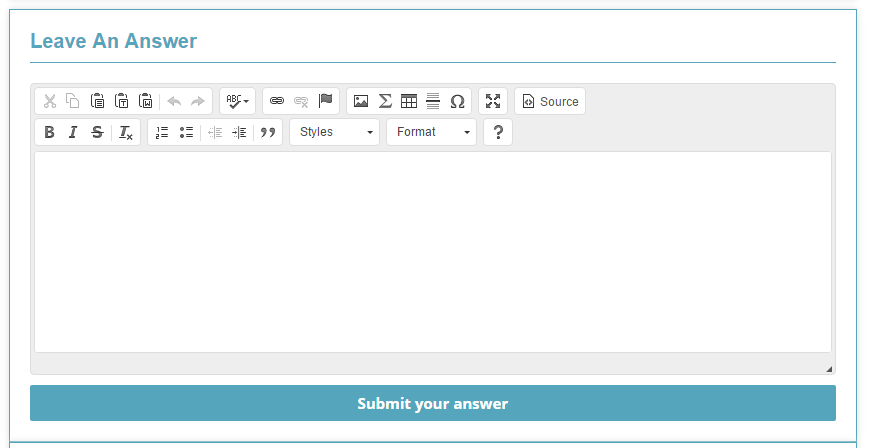
### Create post



**Figure 107: Create post form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User fills out all of required fields with question/article information. |
| 2 | User clicks button “Publish your post” to create a/an question/article. |

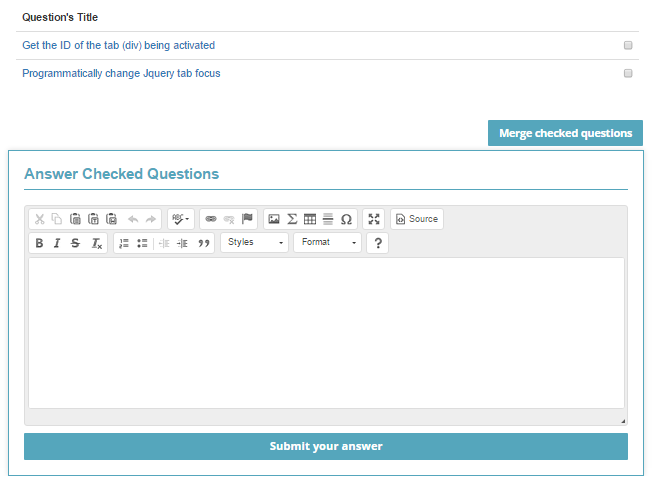
### Answer



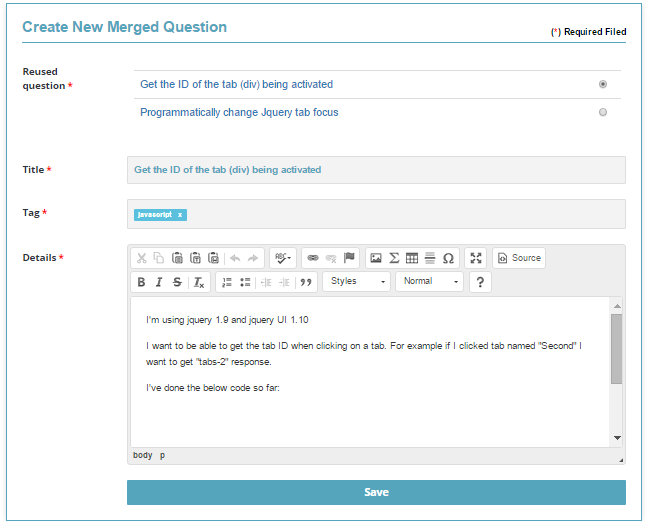
**Figure 108: Answer form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User fills out answer. |
| 2 | User clicks button “Submit your answer” to leave an answer. |

### Merge question



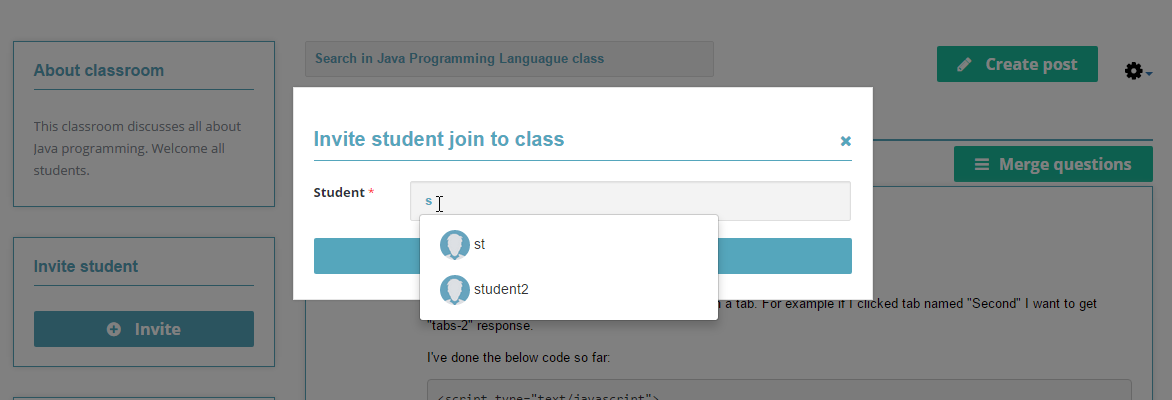
**Figure 109: Merge questions form**



**Figure 110: Create new merged question form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User selects questions that will be merged |
| 2 | (Optional) User fills out text area and click “Submit your answer” |
| 3 | User clicks “Merge checked questions” |
| 4 | User selects one question that will be re-used |
| 5 | User changed others fields if needed |
| 6 | User clicks “Save” button to save re-used question |

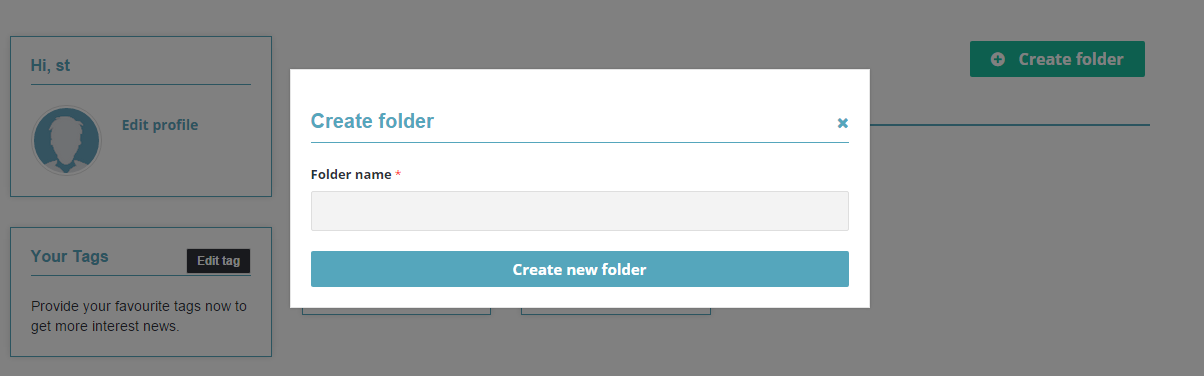
### Invite student



**Figure 111: Invite student form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User clicks “Invite” button in “Invite student” widget |
| 2 | User fills out student’s name |
| 3 | User selects student from list |
| 4 | User clicks “Invite” button in “Invite student to join class” popup |

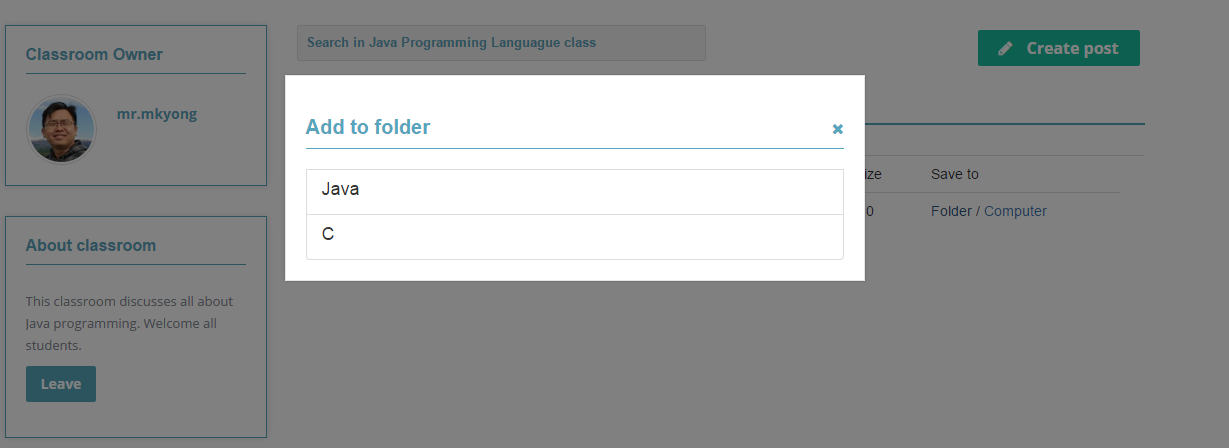
### Create folder



**Figure 112: Create folder form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User clicks “Create folder” button |
| 2 | User fills out folder name |
| 3 | User clicks “Create new folder” button |

### Add to folder



**Figure 113: Add to folder form**

|  |  |
| --- | --- |
| **Step** | **Description** |
| 1 | User clicks “Folder” |
| 2 | User selects folder to add to |

# Appendix

* 1. Code Conventions for the Java TM Programming Language, by Sun Microsystems, rev April 20, 1999.  
     <http://www.oracle.com/technetwork/java/codeconventions-150003.pdf>
  2. Jaccard index

<https://en.wikipedia.org/wiki/Jaccard_index>

* 1. How To Install and Use Redis, by Etel Sverdlov, Aug 16 2012  
     <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-redis>

**Sprint 1 (12/05/2015-18/05/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Design homepage |  | MinhKH | 10 | Completed | 10 | 10 | 10 | 10 | 5 | 0 | 0 |
| 2 | Design register page | Didn't submit | ThiTD | 6 | In process |  |  |  |  |  |  |  |
| 3 | Design login form | Pop-up form, didn't submit | ThiTD | 6 | In process |  |  |  |  |  |  |  |
| 4 | Design get forget password page | Didn't submit | ThiTD | 6 | In process |  |  |  |  |  |  |  |
| 5 | Design student dashboard | Didn't meet deadline &requirement | TungTD | 8 | In process |  |  |  |  |  |  |  |
| 6 | Design teacher dashboard | Didn't meet deadline &requirement | TungTD | 8 | In process |  |  |  |  |  |  |  |
| 7 | Design admin dashboard | Didn't meet deadline &requirement | TungTD | 8 | In process |  |  |  |  |  |  |  |
| 8 | Report 1 | Didn't meet deadline | ThiTD | 4 | Completed |  |  |  |  |  |  |  |
| 9 | Review report 1 |  | MinhKH | 3 | Completed | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | Setup environment tutorial |  | KhangTN | 8 | Completed | 8 | 8 | 4 | 0 | 0 | 0 | 0 |
| 11 | Create overall use-case |  | KhangTN | 15 | Completed | 15 | 15 | 12 | 5 | 0 | 0 | 0 |
| 12 | Create guest use-cases | Didn't submit | ThiTD | 10 | In process |  |  |  |  |  |  |  |
| 13 | Create ERD |  | MinhKH | 12 | Completed | 12 | 12 | 6 | 0 | 0 | 0 | 0 |
| 14 | Create System Architectural Design |  | KhangTN | 10 | Completed |  |  |  |  | 5 | 5 |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | Report 2 | Didn't meet deadline | TungTD | 8 | Completed |  |  |  |  |  |  |  |
| 16 | Review report 2 |  | MinhKH | 3 | Completed | 3 | 3 | 3 | 3 | 3 | 3 | 0 |
| 17 | Review use-case, page design |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 4 | 4 | 4 | 0 |

**Sprint 2 (19/05/2015-25/05/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Design class page |  | MinhKH | 12 | Completed | 6 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | Design post page |  | MinhKH | 12 | Completed | 12 | 12 | 6 | 0 | 0 | 0 | 0 |
| 3 | Design notification |  | KhangTN | 8 | Completed | 8 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | Design teacher's profile viewer | TungTD->MinhKH | TungTD | 8 | Completed |  |  |  |  |  |  |  |
| 5 | Design student's profile viewer | TungTD->MinhKH | TungTD | 8 | Completed |  |  |  |  |  |  |  |
| 6 | Design create class page |  | KhangTN | 10 | Completed | 10 | 10 | 5 | 0 | 0 | 0 | 0 |
| 7 | Design create post page |  | KhangTN | 16 | Completed | 16 | 16 | 16 | 16 | 8 | 0 | 0 |
| 8 | Design register page | Sprint 1 task (ThiTD -> MinhKH) | MinhKH | 6 | Completed | 6 | 6 | 6 | 6 | 0 | 0 | 0 |
| 9 | Design login form | Sprint 1 task (ThiTD -> MinhKH) | MinhKH | 6 | Completed | 6 | 6 | 6 | 6 | 6 | 0 | 0 |
| 10 | Design get forget password page | Sprint 1 task (ThiTD -> MinhKH) | MinhKH | 6 | Completed | 6 | 6 | 6 | 6 | 6 | 6 | 0 |
| 11 | Design student dashboard | Sprint 1 task -> MinhKH | TungTD | 8 | Completed |  |  |  |  |  |  |  |
| 12 | Design teacher dashboard | Sprint 1 task ->MinhKH | TungTD | 8 | Completed |  |  |  |  |  |  |  |
| 13 | Design admin dashboard | Sprint 1 task | TungTD | 8 | In process |  |  |  |  |  |  |  |
| 14 | Fix overall use-case |  | MinhKH | 1 | Completed | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 15 | Fix ERD |  | KhangTN | 2 | Completed | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 16 | Fix System architecture |  | KhangTN | 2 | Completed | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Sprint 3 (26/05/2015-02/06/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Design physical diagram | DB | MinhKH | 8 | Completed | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | Design suggest page |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 4 | 4 | 4 | 0 |
| 3 | Design Admin dashboard | Sprint 1 Task | MinhKH | 2 | Completed | 2 | 2 | 2 | 0 | 0 | 0 | 0 |
| 4 | Create DB | using MySQL | MinhKH | 6 | Completed | 3 | 3 | 3 | 3 | 3 | 0 | 0 |
| 5 | Map DB - Entity |  | KhangTN | 12 | Completed | 12 | 12 | 4 | 0 | 0 | 0 | 0 |
| 6 | Create dao, dto |  | MinhKH | 8 | Completed | 8 | 8 | 4 | 0 | 0 | 0 | 0 |
| 7 | implement login function | include client side validation | TungTD | 10 | Waiting for Approval |  |  |  |  |  |  |  |
| 8 | implement register function | include client side validation | TungTD | 10 | Waiting for Approval |  |  |  |  |  |  |  |
| 9 | implement newsfeed page |  | TungTD | 12 | Waiting for Approval |  |  |  |  |  |  |  |
| 10 | real-time notification & discussion demo | notification count, left notification, real-time discussion | KhangTN | 12 | Completed | 12 | 12 | 8 | 4 | 0 | 0 | 0 |
| 11 | store material using blob demo |  | KhangTN | 8 | Completed | 8 | 8 | 6 | 6 | 6 | 0 | 0 |
| 12 | Design Manage material |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 4 | 4 | 4 | 0 |

**Sprint 4 (03/06/2015-09/06/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Implement search function | solr | MinhKH | 16 | In process | 16 | 16 | 16 | 16 | 16 | 6 | 6 |
| 2 | Implement upload, add to folder and manage material |  | KhangTN | 16 | Waiting for Approval | 16 | 16 | 16 | 16 | 6 | 2 | 2 |
| 3 | Implement follow/unfollow teacher function |  | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 4 | Implement request function |  | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 5 | Implement invitation function |  | TungTD | 6 | In process |  |  |  |  |  |  |  |
| 6 | Implement create class function |  | TungTD | 6 | In process |  |  |  |  |  |  |  |
| 7 | Implement create post function | include article, question and material | KhangTN | 16 | Completed | 16 | 16 | 16 | 8 | 0 | 0 | 0 |
| 8 | Implement edit profile function | teacher profile, student profile | TungTD | 8 | In process |  |  |  |  |  |  |  |
| 9 | Implement suggest related post function |  | MinhKH | 12 | Completed | 12 | 12 | 12 | 6 | 0 | 0 | 0 |
| 10 | design add to folder popup and create folder popup |  | MinhKH | 4 | Completed | 4 | 4 | 0 | 0 | 0 | 0 | 0 |
| 11 | component diagram |  | KhangTN | 4 | Completed | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | package diagram |  | KhangTN | 4 | Completed | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | class diagram |  | MinhKH | 6 | Completed | 2 | 0 | 0 | 0 | 0 | 0 | 0 |

**Sprint 5 (10/06/2015-16/06/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Implement search function | Task Sprint 4 | MinhKH | 10 | In process |  |  |  |  |  |  |  |
| 2 | Implement upload, add to folder and manage material | Task Sprint 4, not yet "delete" function | KhangTN | 2 | In process |  |  |  |  |  |  |  |
| 3 | Implement invitation function | Task Sprint 4 | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 4 | Implement create class function | Task Sprint 4 | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 5 | Implement edit profile function | Task Sprint 4 | TungTD | 8 | Completed |  |  |  |  |  |  |  |
| 6 | Implement classroom page | Note: + left widget at page create Post | MinhKH | 8 | Completed |  |  |  |  |  |  |  |
| 9 | Implement news feed page |  | MinhKH | 12 | Completed |  |  |  |  |  |  |  |
| 8 | Implement manage post |  | KhangTN | 10 | In process |  |  |  |  |  |  |  |
| 9 | Implement reply post |  | KhangTN | 10 | Completed |  |  |  |  |  |  |  |
| 10 | Implement accept, deny request and invitation |  | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 11 | Implement manage class |  | KhangTN | 12 | In process |  |  |  |  |  |  |  |
| 12 | Implement Redis database and schedule |  | MinhKH | 8 | Completed |  |  |  |  |  |  |  |
| 13 | Remove all un-needed Ajax, use controller |  | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| Note: This sprint must implement core flow firstly, all un-completed task will be implement in sprint 6 | | | | | | | | | | | | |

**Sprint 6 (17/06/2015-23/06/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Real time notification |  | KhangTN | 16 | In process | 16 | 16 | 16 | 16 | 12 | 12 | 8 |
| 2 | Implement upload, add to folder and manage material | Task Sprint 5, not yet "delete" function | KhangTN | 4 | Completed | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | Implement manage post | Task sprint 5: edit, delete answer and post | KhangTN | 10 | Completed | 8 | 6 | 0 | 0 | 0 | 0 | 0 |
| 4 | Implement load more in post view |  | KhangTN | 8 | Completed | 8 | 6 | 5 | 3 | 0 | 0 | 0 |
| 5 | Implement Accept answer and Want answer |  | KhangTN | 4 | Completed | 4 | 4 | 4 | 4 | 2 | 0 | 0 |
| 6 | Implement remove student from class |  | TungTD | 4 | Completed |  |  |  |  |  |  |  |
| 7 | Implement Join button | state 1:"Join" if not yet in class, state 2: "Request sent" if request, state 3: "Leave" if joined, state 4: "Accept invitation" if invite | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 8 | Implement load more in classroom |  | TungTD | 6 | Completed |  |  |  |  |  |  |  |
| 9 | Implement manage class | Close class ("create post" hidden), Close -> Reopen, Update classroom | TungTD | 10 | Waiting for Approval |  |  |  |  |  |  |  |
| 10 | Fix classroom page and create class page | load tag, input tag in create Class page | TungTD | 6 | In process |  |  |  |  |  |  |  |
| 11 | Implement notification | Not real-time yet | MinhKH | 12 | Completed | 12 | 6 | 0 | 0 | 0 | 0 | 0 |
| 12 | Fix category |  | MinhKH | 2 | New |  |  |  |  |  |  |  |
| 13 | Implement Student dashboard and teacher dashboard |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 0 | 0 | 0 | 0 |
| 14 | Implement Newsfeed algorithm, load more newsfeed |  | MinhKH | 8 | Completed | 8 | 8 | 8 | 8 | 0 | 0 | 0 |
| 15 | Implement View profile page |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 4 | 4 | 0 | 0 |
| 16 | Tracking question topics |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 4 | 4 | 4 | 0 |

**Sprint 7 (01/07/2015-07/07/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Real time notification (continue) |  | KhangTN | 8 | In process | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | Implement count question's view function |  | KhangTN | 6 | Completed | 6 | 6 | 0 | 0 | 0 | 0 | 0 |
| 3 | Invite student to classroom | TungTD didn't complete task | KhangTN | 4 | Completed | 4 | 4 | 4 | 0 | 0 | 0 | 0 |
| 4 | Fix real-time discussion |  | KhangTN | 2 | Completed | 2 | 2 | 2 | 2 | 0 | 0 | 0 |
| 5 | Fix create classroom | TungTD didn't complete task | KhangTN | 2 | Completed | 2 | 2 | 2 | 2 | 0 | 0 | 0 |
| 6 | Update tag for material |  | KhangTN | 4 | Completed | 4 | 4 | 4 | 4 | 0 | 0 | 0 |
| 7 | Implement article page |  | KhangTN | 6 | Completed | 6 | 6 | 6 | 6 | 6 | 0 | 0 |
| 8 | Design search result page |  | MinhKH | 6 | Completed | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | Implement search function |  | MinhKH | 8 | Completed | 8 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | Implement related articles, materials |  | MinhKH | 6 | Completed | 6 | 6 | 0 | 0 | 0 | 0 | 0 |
| 11 | Implement article & material for newsfeed |  | MinhKH | 4 | Completed | 4 | 4 | 4 | 0 | 0 | 0 | 0 |
| 12 | Fix theme's color | using basic color: black, green, blue, red | MinhKH | 2 | Completed | 2 | 2 | 2 | 0 | 0 | 0 | 0 |
| 13 | Authorize user |  | MinhKH | 6 | Completed | 6 | 6 | 6 | 6 | 6 | 0 | 0 |

**Sprint 8 (08/07/2015-14/07/2015)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Task | Detail information | Implementer | Estimated hours | Status | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 | Day 6 | Day 7 |
| 1 | Implement merge question | merge question, answer multi question | KhangTN | 20 | Completed | 16 | 10 | 6 | 0 | 0 | 0 | 0 |
| 2 | Implement admin function |  | KhangTN | 12 | Completed | 12 | 12 | 12 | 12 | 6 | 0 | 0 |
| 3 | Implement load more using Ajax |  | MinhKH | 20 | Completed | 16 | 10 | 6 | 0 | 0 | 0 | 0 |
| 4 | Fix close class, open class, follow teacher, unfollow teacher |  | MinhKH | 12 | Completed | 12 | 12 | 12 | 12 | 6 | 0 | 0 |

**Sprint 9 (15/07/2015-21/07/2015)**

Fix bugs and test. Implementer: MinhKH, KhangTN

**Sprint 10 (22/07/2015-28/07/2015)**

Fix bugs and test. Implementer: MinhKH, KhangTN

**Sprint 11 (29/07/2015-04/08/2015)**

Fix bugs and test. Implementer: MinhKH, KhangTN

**Sprint 12 (05/08/2015-11/08/2015)**

Deploy to server, fix bugs and test. Implementer: MinhKH, KhangTN

**Sprint 13 (12/08/2015-18/08/2015)**

Fix bugs and test. Implementer: MinhKH, KhangTN

**Sprint 14 (19/08/2015-25/08/2015)**

Prepare for presentation. Implementer: MinhKH, KhangTN