NKUNet Design Document

Last Modified: April 17th, 2017

Tables of Contents:

1. Introduction

1.1 Background

NKUNet is a forum system designed for people within the NKU community. The ideal outcome of this is to provide a forum for faculty, students and registrars which are able to ask questions and get answers from others within the community. The idea behind this is to grant a social area for both faculty and students to share their questions and thoughts and to be able to communicate with people on a wider network.

1.2 Design Goals

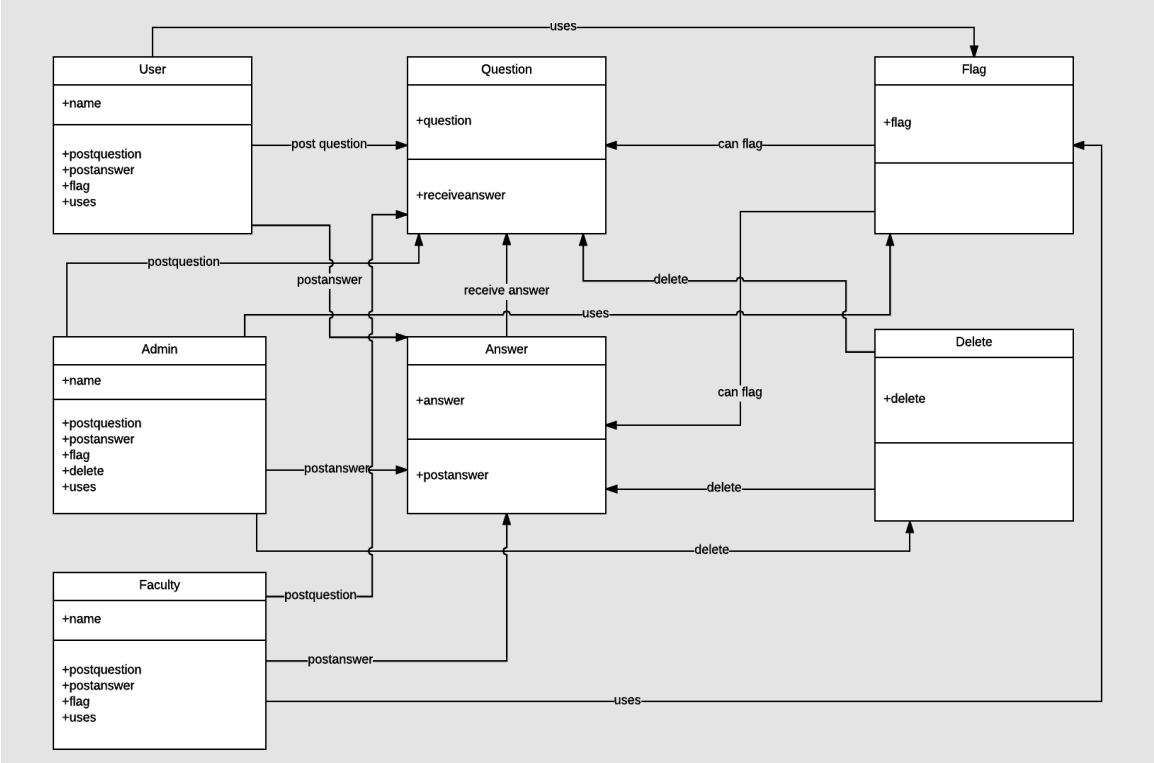
The basic design goal is the creation of a forum for those within the NKU community. The users should have the ability to post questions, receive answers, upvote/downvote responses, have their schedules attached to their user account and add friends, for personal messaging. The idea has potential to expand further beyond this, given development time and resources available should initial implementation be done ahead of schedule.

1.3 Changes From Iteration 1

A large portion of the changes made this iteration revolved around functionality and operations. The basic premise from our original design was to implement features and controls. As a result of that, users are now able to be added/removed from the system,

login/logout of the system, upvote or downvote posts and answer questions. Registrars were added and given the ability, on top of some listed above, to create courses, add users to courses and remove users from courses. Lastly, admins were given the option of being able to un-flag a post if required.

2. Class Diagram



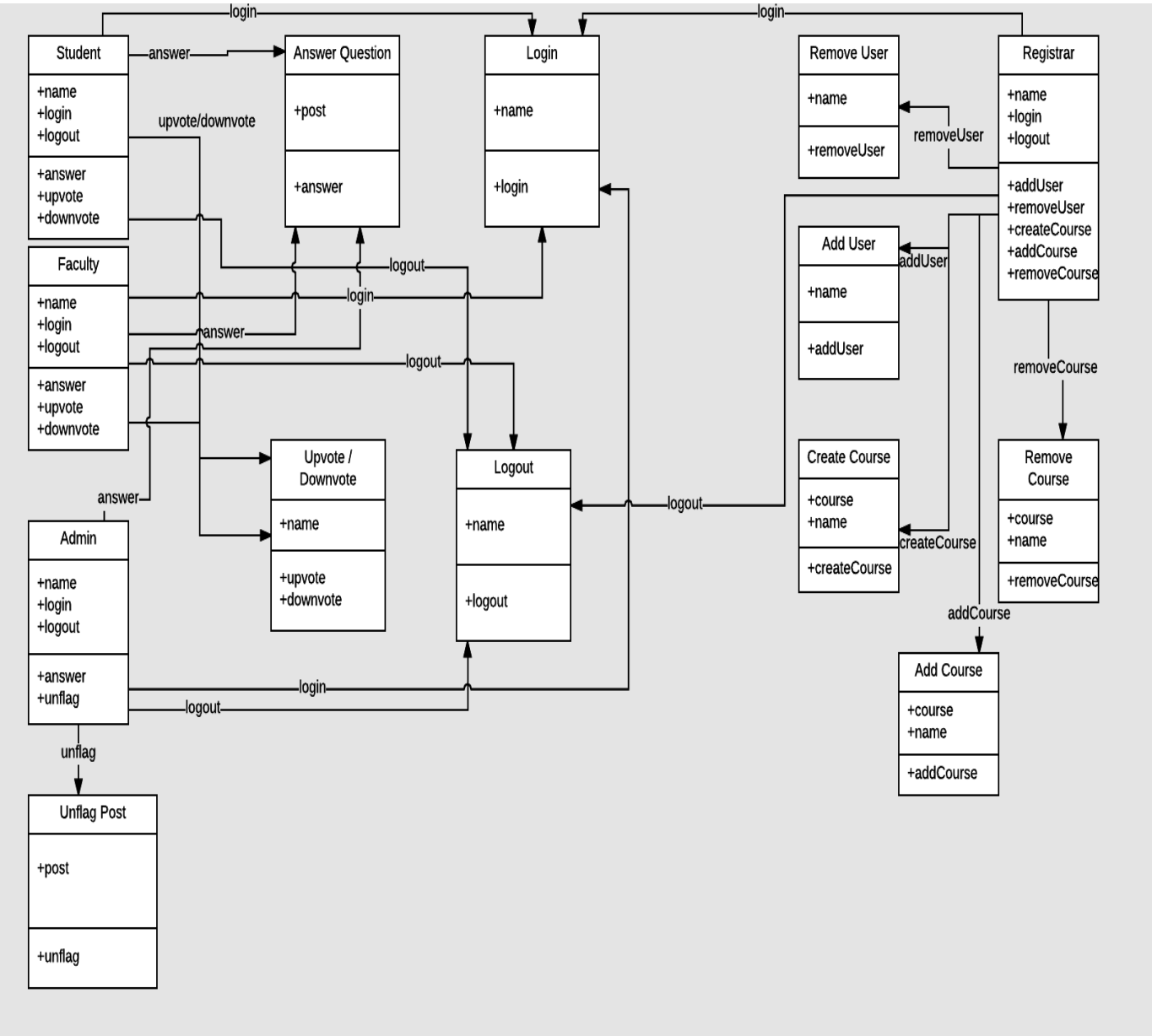
The basic premise behind the class choices here is to get a functional, early model of the system up and running. The user, when posting, will have their names displayed and the ability to post questions or answers on NKUNet. They will also have the ability to use the flag command for posts that need possible looking into for various reasons. Faculty will follow the same permissions as the user currently. The Admin will have the ability to see everything, including posts that are flagged. Admin will also be the class that is granted the delete functionality, for removing either an answer or an entire question. These will likely be expanded upon as time progresses forward.

Iteration 1 → Iteration 2

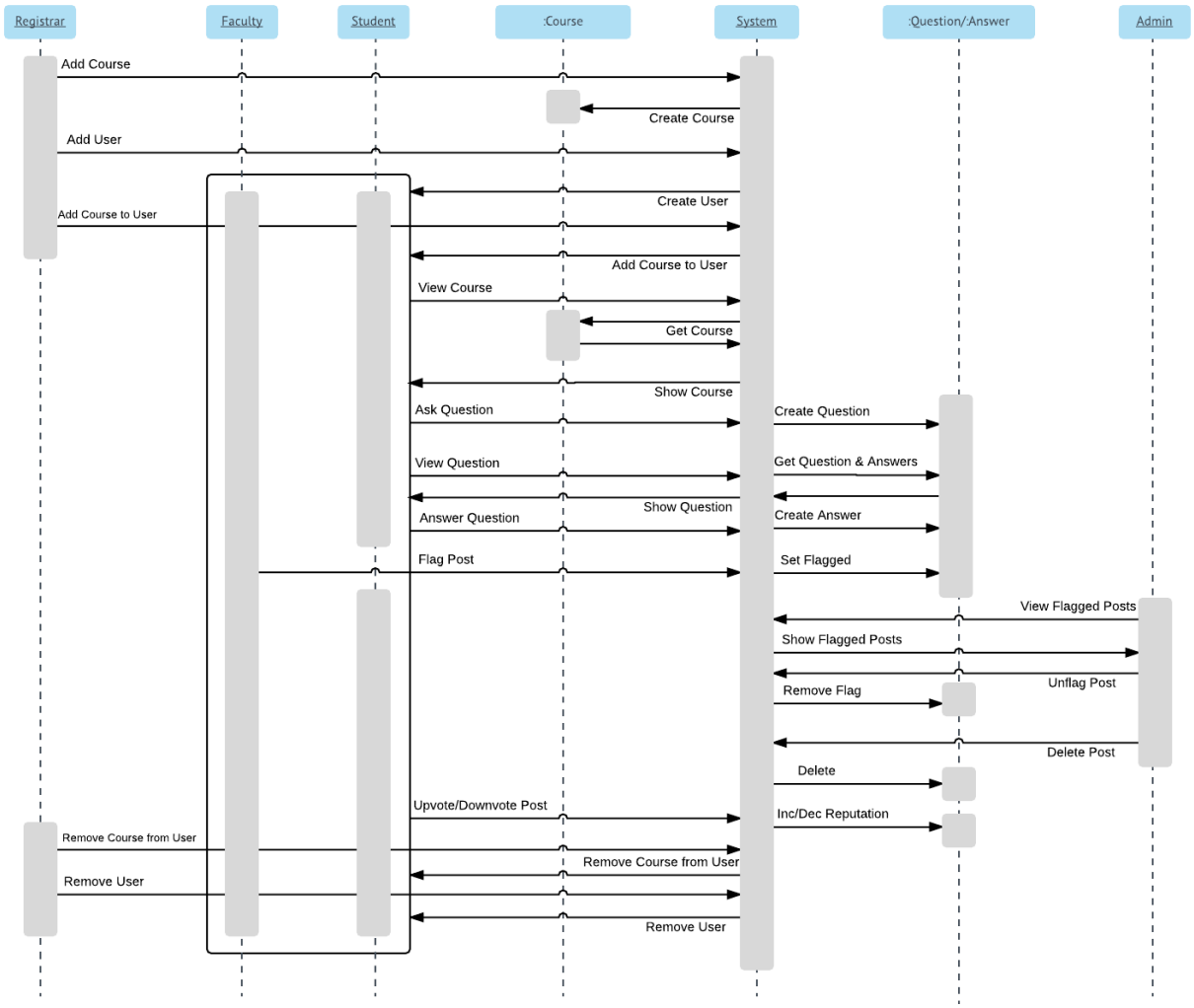
For this iteration, we added a lot of functionality. The registrar was added into the system, allowing users to be added and removed, along with having the functionality to add courses to a given user's schedule as well. This design in essence helps to give users access to certain forums and open up the forum as a

whole. Also added in this iteration was a large chunk of missing functionality. We have added the ability to answer questions, upvote or downvote posts, login and logout, create/add/remove courses from a user and unflagging a post. The result is that the forum has more sorting potential, accessibility and comes off as a more polished version of the original design.

Class Diagram Iteration 2:



3. Design Sequence Diagram



The Sequence diagram is a current basic flow chart for the current design. Registrars and Admins will likely already be added to the system upon implementation; however, if more admins/registrars are required they can be added by any admin or registrar. The Registrar is responsible for adding users and courses to the system. Admins have this ability as well, though it is not a task that they will likely be performing often. Registrars are also responsible for "connection" users and courses. This allows the system to show each user only relevant information and also allows the system to prevent users from commenting in forums where they do not belong. Students and Faculty are able to post in courses in which they are enrolled/teach. If a question or answer is inappropriate or misplaced then it can be flagged by a Faculty member to be examined by an admin. Flagging a post hides the content (temporarily) until an Admin either unflags it or deletes it. Admins, upon logging in, are shown a list of all flagged posts. An admin may choose to unflag a post or delete it permanently.

4. Rails Commands

4.1 – rails generate scaffold User name: string email: string

This command has a name field and an email field.

4.2 – rails generate scaffold Question content: text user_id: integer

This command we have a content field and an id to differentiate questions

4.3 - rails generate model Question content: text user: references –force

This command we have a content field and a user reference foreign key so that we can easily see who asked. The –force means we overwrite the db table.

4.4 - rails generate scaffold Answer content: text user_id: integer

This command we have a content field and an id to differentiate answers.

4.5 – rails generate model Answer content: text user: references question: references –force

This command we have a content field and a user reference foreign key so that we can easily see who asked. The question reference also has a key to the question foreign key. The –force means we overwrite the db table.

5. GRASP Patterns

The session class is largely a pattern based on the "expert" class. This is largely in part due to the nature of the class and it requiring to know information and functionality of other classes. Aside from this, most other classes feature a job or task of some nature to be performed, so all of them have purposes for the overarching design of the system.

6. Database Design

The database design is currently fairly simplistic in nature. It consists of four tables, labeled:

a. Courses

b. Users

c. Questions

d. Answers

It also has a junction table, aptly named course2users. This junction table is what allows the system to add courses to a user while also having users attached to a course. While simple in design, this manages the entirety of the forum and is efficient at this iteration.