Software Engineering 14:332:452

Group: 9

Project Title: Minerva

Project Site URL:
https://rutgerssoftwareengineering.github.io/Minerva/

Date Submitted: April 14, 2019

Team Members: Jacob Battipaglia, Jonathan Hong, Skyler Lee, Brian Ma, Justin May, Salman Omer, Joshua Olazo, Phurushotham Shekar

Individual Contributions Breakdown

Jacob	Jon	Skyler	Brian	Justin	Salman	Joshua	Phuru
12.5	12.5	12.5	12.5	12.5	12.5	12.5	12.5

Table Of Contents

Summary of Changes

1. Customer Problem Statement :

Common complaints of existing classroom technology outside of class expanded upon.

2. Glossary of Terms :

Formatting Fix

3. System Requirements :

Split up REQ-3 into two requirements (REQ-3 and REQ-32)
REQ-10,11, and 12 priority weights have been adjusted.

4. Functional Requirements Specification:

REQ – 16, TA can now also be able to view the forum posts.
REQ – 14, TA can now have access to manage members of the class.
REQ – 22, TA can now be one of the involved actors.

Added more fully dressed use cases for features that will be completed for the final demo

6. Domain Analysis (25 Points):

150 seconds timing threshold and 20 question threshold explained. This is expanded upon in Section 10.

9. System Architecture and System Design:

Updated/provided a brief description on the UML diagram.
Updated system specifications.

10. Algorithms and Data Structures:

Description was edited to more accurately describe the search algorithm process.

Algorithm for matching resources to keywords explained more in depth. It is also now described as potential future work since we did not implement it for our final demo. The data structure for this algorithm is also described in the "data structures" section

Hashing algorithm explained more in depth. We identified what libraries we will use to accomplish these hashing techniques and what kind of transformations we will subsequently perform on that data.

13. History of Work, Current Status, and Future Work:

Section completely rewritten to reflect our current state at that end of this project as well as highlighting what future work can be done to build on what we have.

14. References (Round 5 Points):

4th Reference, Hyperlink was removed.

References Added

1. Customer Problem Statement

This section will contain the problem as described by the customer and what type of system they want in order to solve their issues.

a. Problem Statement

With rising University education costs in the United States, the question of the efficacy of a bachelor's degree is put more and more into question. A Georgetown Public Policy Institute study puts the percentage of American jobs that require post-secondary education beyond high school at 65 *per cent* by the year 2020; however, as more and more Americans attend universities across the country, professor to student ratios will rise and university infrastructure will struggle.[1] In addition, as class sizes continue to increase new challenges and barriers—such as decreasing student in-class interaction—arise that prevent the retention of information and the educational process.[2] A report from the Cornell School of Industrial Labor Relations indicated "that both class size and student load negatively impact student assessments of courses and instructor"

Skyrocketing university tuition, in conjunction with ballooning classroom sizes, could positively benefit from the introduction of innovative software in the classroom.

One of the largest areas of concern is with classroom size. College attendance has increased by 28 *per cent* since 2000, and various studies have demonstrated a negative impact on student performance from classroom size.[3,4] In fact, the National Center for Academic Transformation estimated that only

[1] Anthony P Carnevale, Nicole Smith, and Jeff Strohl, "Recovery: Job Growth and Education Requirements Through 2020," *Georgetown Public Policy Institute*, last accessed 10 February 2019, https://cew.georgetown.edu/wp-content/uploads/2014/11/Recovery2020.ES_.Web_.pdf.
[2] Monks, J. & Schmidt, R. (2010). The impact of class size and number of students on outcomes in higher education[Electronic version]. Retrieved 10 February 2019, from Cornell University, School of Industrial and Labor Relations site: http://digitalcommons.ilr.cornell.edu/workingpapers/114/
[3] Ibid.
[4] "Undergraduate Enrollment," *National Center for Education Statistics*, las accessed 10 February 2019, https://nces.ed.gov/programs/coe/indicator_cha.asp

25 of the most attended college courses contributed to over 35 *percent* of four-year college attendance annually.[5]

Common complaints from large classroom (Students):

1. **Hard to have a voice**
    a. **Difficult to ask questions to the professor without disrupting the lecture**
        i. **Sometimes class time is wasted on less important questions**
        ii. **Speaking in front of the entire class is intimidating**
        iii. **Especially hard to ask for clarifications when it is your fault for not paying attention or missing the last class**
        iv. **Even "good questions" could lead to a back and forth which could waste class time**
        v. **Asking questions could lead to judgement from classmates**
    b. **Awkward to give feedback to professor during class**
        i. **Hard to ask for special accommodations such as professor speaking louder**
        ii. **Individual problems may not be worth it to disrupt lecture**
        iii. **May not want to give negative feedback in a non anonymous way**
2. **Seating can be important based on the professor's style of teaching**
    a. **For lectures that depend on complex diagrams and equations that are presented on the blackboards, it is often difficult for students farther from the front to understand and copy what is being written down**
    b. **Some professors possess poor or no equipment to project their voice, making it much more difficult to listen to students not in the front**

Common complaints from large classrooms (Professors):

1. **Difficult to keep students engaged**

[5] "Monstrous class sizes unavoidable at colleges," *NBC news,* last modified 24 November 2007, http://www.nbcnews.com/id/21951104/ns/us_news-education/t/monstrous-class-sizes-unavoidable-colleges/

    a. **Cannot keep students attention for students in the back**

2. **Hard to assess students' understanding**
    a. **Paper and Pen assessments take limited lecture time and need to be graded by hand**
3. **Lack of feedback from students**
    a. **Negative feedback usually only comes in when semester in anonymous survey**
4. **Discussion amongst students interrupts lecture**
    a. **Cannot tell meaningful discussion apart from idle chat**
5. **Often held back after class so that students can ask questions**
    a. **Long queues can take away too much time from professors**
    b. **Some professors have another class to teach, and must redirect students to email him/her, unable to give an immediate response**
6. **Misuse of emails and online forums**
    a. **Students may send emails with inquiries that can be solved through Sakai forums/resources, cluttering the professors inbox**
    b. **Some inquiries posted on online resources end up being unanswered due to the nature of the problem only being possibly solved by the professor him/herself**

A recent Kennesaw State University study using student gaze as a proxy for attention found specific in-class methodologies and factors that increased student attention. Some factors were out of the control of the professor; students that sat in the back of the classroom showed less overall attention that students that sat in the middle and front, which would be exacerbated with increased classroom size.[6] The study found that students who sat on the sides and back of the classroom would

[6] David Rosengrant, Doug Hearrington, Kerriann Alvarado, and Danielle Keeble, "Following Student Gaze Patterns in Physical Science Lectures," *University of Kennesaw*, last accessed 10 February 2019, https://www.usnews.com/pubfiles/PERC2011_Rosengrant.pdf.

inadvertently see the computer screens and phone screens of students in front of them, of which texts and Facebook seemed to be the largest contributor to distractions.[7] In addition, student and professorial back and forths seemed to indicate a large increase in attention, yet was polarizing in that some students would quickly lose attention.[8] Finally, a large predictor of student distraction was pre-printed or pre-released class notes, suggesting that students that could only access notes in class tended to pay more attention.[9]

Where other industries, such as transportation and the hotel industry, have greatly innovated over the past few decades, education continues to lag behind. The college classroom has seldom changed over the past few decades and the pedagogical methods professors adopt with it. Technology has the potential to solve various problems plaguing the classroom. Most students already the technology needed, "83% own a laptop, and over 50% have a Smartphone"[10]. Additionally, classrooms are already equipped to interact with students through technology, "97% of classrooms had one or more computers, and 93% of classroom computers had Internet access."[11]

Currently some college lectures are using IClickers for in class polling and quizzes. This solution is bad for students because it does not leverage the smartphones and laptops that students already carry. Requiring students to buy specialized hardware and carry it to class everyday is less than ideal

1. **Problems with IClickers**
    a. **Cost money to buy and app needs to be paid per semester**
    b. **Have to carry the remote to class**
    c. **University has to buy receivers**
    d. **Can only answer simple multiple choice questions**
    e. **Cannot see questions and answers after class for review**

[7] Ibid.
[8] Ibid.
[9] Ibid.
[10] "The Evolution of Technology in the Classroom." *Purdue University Online*, 1 Aug. 2017, online.purdue.edu/ldt/learning-design-technology/resources/evolution-technology-classroom.
[11] Ibid.

There exists demonstrable and specific actions that can greatly increase the attentiveness of students in class that can counteract the trend of rising classroom numbers, giving greater benefit from the rising costs of colleges. The aforementioned studies outline key moments in the classroom that can be eliminated or improved upon with technology. Instead of spending their in-class time on Facebook, if students could spend their in-class time using the mobile application, answering quizzes, asking questions on the material, providing the Professor feedback, they would be more engaged in the material and there would be more interaction and interest in the material. Utilising these tendencies to look at our phones during class, but essentially leading the students to use it for classroom purposes instead of social media should be the goal. Bridging the generation gap between antiquated classrooms and 21st century technology can provide a panacea for the modern ailments of education.

Another domain of this problem is out of class information retention. For example, there exists an asymmetry between the homework knowledge and student knowledge and thus students will often revert to cheating or copying homework answers when it gets too challenging. If the purpose of homework and quizzes is to test student knowledge and encourage out of class exploration, it fails because the friction is too high (student frustrations often trump the desire to learn). Thus, a system that is knowledgeable of student flaws and weaknesses, possibly attaining this knowledge from homework and quiz answers, would suppose the place of a tutor. This type of personalized learning would increase a students willingness to learn. For example, specific Laplace transforms presuppose knowledge about integration by parts. A good tutor would have identified a student weakness with integration by parts, perhaps by a low grade in a prior quiz, and provide a integration by parts problem before attempting the Laplace transform problem.

Common complaints of existing classroom technology:

1. **Sakai/Canvas/Piazza**
    a. **Hassle to log on and switch between different sites for different classes**
    b. **Lack of connectivity and integration between sites**
2. **Quiz feedback can be inadequate**

a. After taking a quiz administered through Sakai, the feedback only indicates the correct answers and sometimes (although rare) has an explanation. However, it is difficult to know the larger concept to study to improve rather than just knowing how to answer the particular problem

b. No indication of what topics to focus on

3. Some forums are difficult to use

a. Forum on Sakai does not have an intuitive interface. For example, a class with an available forum option simply displays "No forum has been created". This does not give the user information on how to create threads or participate in the forum, and there is seemingly no option to do so.

2. Glossary of Terms

The following is a list of terms which may appear throughout our report and are necessary for complete understanding of our design and development process. These terms may describe technical features as well as functional capabilities of our software.

- Polls - This feature allows professors to ask short, multiple choice questions during class. Students then have the ability to answer the questions from their phones. After the professor closes the question, the application will show the distribution of answers from the students in class.
- Mobile App - a separate downloadable app that links with the website to provide extra functionality not available on the web page, but does not contain the same features as the web page
- Public Forum - a collection of messages that students can reply to or ask questions; a place of discussion for both the instructor and his/her students
- Private Forum - The student-only forum that professors and TAs do not have access to
- Upvote - a quantifiable value that represents the approval or agreement from fellow classmates
- Downvote - an expression of disapproval or disagreement about a post or comment
- Large Classroom - a learning environment where the student to professor ratio exceeds at least 50:1, making teaching almost impractical
- Active Feedback - Feedback given to the professor during the class, through the mobile app, generally about more specific actions. E.g. "talk louder"
- Passive Feedback - Feedback given to the professor at any time, through the web site, generally about more broad actions. E.g. "Plan more lectures around the blackboard"
- Platforms - the devices that users will be able to access our features through (smartphones or desktops/laptops)
- I-Clicker - small handheld device used to answer multiple choice questions the professor posts during class. Sometimes mandatory for classes

- Account - a collection of user-related information (username, class-enrollment, professor or student, etc.) that is only accessible to the owner of that information through username and password authentication
- Online Office Hours - live video recordings for professors to provide additional academic help or clarification.

3. System Requirements

a. Enumerated Functional Requirements

1. Out of Class Requirements (Web Page)

Label	Priority Weight	User Story
REQ-1	4	As a student or professor, I should have the ability to post and answer questions, and communicate with the other members of my class via an online forum
REQ-30	4	As a student or professor, I should have the ability to view announcements
REQ-31	3	As a professor, I should have the ability to create announcements
REQ-2	3	As a student, I would also like to have access to a private, student-only forum for discussing class topics with my peers and arranging things like tutoring sessions or study group meetings
REQ-3	2	As a user, I will have the ability to access and view grades for assignments
REQ-31	2	As a professor, I will have the ability to update grades for students and classes for which I have permission
REQ-4	2	As a professor, I will be able to upload files such as slides, additional reading, and other class resources for my students
REQ-5	4	As a student, I will be able to access class resources uploaded by my professor, as well as view my current performance in the class and on individual assignments
REQ-6	1	As a professor, I will have the ability to post quizzes and gather specific information on what topics my students have fully mastered or identify weak areas.
REQ-7	1	As a professor, I will have the ability to host online lectures to provide additional out-of-class help for my students.
REQ-8	5	As any user, I do not want to go to multiple sources in order to access class information. (Want to combine Piazza + iClickers + Sakai)

2. In class Requirements (Mobile App for Students, and Web Page for Professors)

Label	Priority Weight	User Story
REQ-9	5	As a student, I can use the polling feature of Minerva to participate in in-class quizzes and questionnaires
REQ-10	4	As a student, I can use the app to give live feedback and anonymously ask questions to my professors during lecture
REQ-11	2	As a student, I can view questions that my peers have asked and use a voting system to vote on questions that I agree with
REQ-12	2	As a professor, I can view questions that are submitted by my students and see which have been "upvoted" by other students for visibility.

b. Enumerated Non-Functional Requirements

1. Out of Class Requirements (Web Page)

Label	PW	User Story
REQ-13	5	As a professor, I should not be able to view the student-only forum
REQ-14	4	As a professor, I shall have the ability to control what students are registered for my particular class.
REQ-15	3	As a user, I should be able to change my password if I forget it.
REQ-16	3	As a user, I should be able to access past chat and forum information.
REQ-17	2	As a user, my personal data, such as what classes I am enrolled in, shall persist across all platforms.
REQ-18	1	As a user, I will have the option to have the website and app remember my password and log me in automatically

2. In class Requirements (Mobile App for Students, and Web Page for Professors)

Label	PW	User Story
REQ-19	5	As a user, I should be able to view questions asked in class in real time

REQ-20	1	As a user, I will have the option to have the website and app remember my password and log me in automatically

c. On-Screen Appearance Requirements

1. Out of Class Requirements (Web Page)

Label	**Priority Weight**	**User Story**
REQ - 21	5	There needs to be a log in screen that logs me into the relevant type of account, professor or student.
REQ - 22	1	As a user, I must not be bogged down in features that are cluttered and poorly organized

2. In class Requirements (Mobile App for Students, and Web Page for Professors)

Label	**Priority Weight**	**User Story**
REQ - 23	5	There needs to be a log in screen that logs me into the relevant type of account, professor or student.
REQ - 24	5	As a student in class, I need to be able to provide feedback to the professor at the touch of a button.
REQ - 25	3	As a student in class, I need to be able to ask questions easily and anonymously without too much navigation.
REQ - 26	4	As a student in class, I need a screen where I can view other students questions and upvote/downvote them.
REQ - 27	2	As a student in class, I want to be able to quickly answer the in-class quizzes at the touch of a button.
REQ - 28	3	As a professor in lecture, I need highly upvoted questions to pop up on my screen without me having to navigate anywhere else during lecture
REQ - 29	1	As a student in class, I need to be able to easily read poll questions and answers and select them intuitively

4. Functional Requirements Specification

a. Stakeholders:

Professors, Teaching Assistants, Learning Assistants, and University Faculty: this group of people are interested in seeing the benefits of this technology in the classroom to improve student educational retention.

University Governance: this group of people are interested in seeing better student outcomes over time, improved student performance would boost University statistics.

Students: this group of people are interested in benefiting by being able to interact with the learning material in novel ways, as well as give immediate feedback in class.

Prospective Parents and Students, Current Parents: would be attracted to novel uses of technology in the classroom, especially if there is a demonstrable boon in student education.

Academic Researchers: The current academia surrounding student focus is mostly centered around studying the effects in primary and secondary education. Controlled studies could be conducted on post-secondary education, to study and reproduce the existing findings in post-secondary education metrics.

b. Actors and Goals

Related Req's	Actor	Goal	Use Case Name	Priority
REQ-6 REQ-9 REQ-27 REQ-29	Student	View and answer quiz questions on the web site	UC-1 Take Quiz	5
REQ-6, REQ-9, REQ-27, REQ-29	Professor	Create and post a quiz on the website for the class to see	UC-2 Create Quiz	5

REQ-30,	Student Professor	See any past announcements the professor posted about the class	UC-3 View Announcement	3
REQ-30, REQ-31	Professor or TA	Create and post an announcement on the website for everyone to see	UC-4 Create Announcement	3
REQ-9, REQ-10, REQ-11, REQ-12, REQ-19, REQ-24, REQ-25, REQ-26, REQ-28	Student	Create a question for other students to see and vote on in the mobile app	UC-5 Create Question (in-class)	5
REQ-9, REQ-10, REQ-11, REQ-12, REQ-19, REQ-24, REQ-25, REQ-26, REQ-28	Student	Allows student to move questions they see useful higher on the professor's list	UC-6 Vote on Question (in-class)	2
REQ-9, REQ-10, REQ-11, REQ-12, REQ-19, REQ-24, REQ-25, REQ-26, REQ-28	Professor	Address and delete questions or feedback from the professor's list	UC-7 Dismiss Question (in-class)	2
REQ-16	Student Professor TA	View all public forum posts	UC-8 Read Public Forum	4
REQ-17 REQ-3 REQ-14	Student Professor TA	Look at profile and data related to user	UC-9 Profile View	3
REQ-18 REQ-20	Student Professor	Log into web or mobile app without entering password	UC-10 Remembered	1

REQ-23	TA		Log-in	
REQ-15	Student Professor TA	Change password used to authenticate and access account	UC-11 Change Password	1
REQ-1 REQ-8	Student Professor TA	Post a reply and new thread for everyone else to see and respond to	UC-12 Post in Public Forum	5
REQ-1 REQ-2 REQ-13	Student	View all past private forum posts	UC-13 Read Private Forum	1
REQ-4	Professor TA	Post class materials such as lecture slides and videos in one location that students can view and download from	UC-14 Upload Class Materials	3
REQ-5	Student Professor TA	View and download class materials uploaded by the professor	UC-15 View Class Materials	3
REQ-21,	Student Professor TA	Authenticate account details to allow the user to access their relevant features	UC-16 Log-In	5
REQ-3	Professor TA	Access and update grades for specific assignment or student	UC-17 Modify Grades	3
REQ-3	Student Professor	View grade summary for all assignments in a class	UC-18 View Gradebook	2
REQ-7	Professor TA	Host live stream to teach or review material for students	UC-19 Host Lecture	1
REQ-14	Professor Student TA	Manage the members registered for a class and the classes that the user is a member of	UC-20 Modify Classes	2
REQ-22	Professor Student TA	See all the different sections of the website easily	UC-21 View Hub	2

c. Use Cases

i. Casual Descriptions

UC-1: Take quiz - to solve timed quizzes posted by professor

UC-2: Create quiz - professors can create multiple choice or open-ended questions

UC-3: View Announcement - students can see the announcements that professors post

UC-4: Create Announcement - professors post important announcements on the website

UC-5: Create Question (in-class) - students can propose a question to be asked in class to the professor

UC-6: Vote on Question (in-class) - other students can vote on whether a question is important enough to be asked

UC-7: Dismiss Question (in-class) - professors can choose to ignore a question

UC-8: Read Public Forum - students, professors, and TA's can discuss with each other on forum posts

UC-9: Profile View - everyone can view information about a particular user

UC-10: Remembered Log-in - users can stay logged in without having to enter a password all the time

UC-11: Change Password - users can change the password they use to log in with

UC-12: Post in Public Forum - anyone can post in the public forum to ask a question, or bring up a topic of interest for further discussion

UC-13: Read Private Forum - only students can read any posts in this forum

UC-14: Upload Class Materials - professors and TAs can post any additional resources students may need

UC-15: View Class Materials - students can access and download any files hosted on the class's resources page

UC-16: Log In - authentication for every single user that every other feature uses to identify the students

UC-17: Modify Grades - professors can modify the grades of their students

UC-18: View Gradebook - professors and students can view grades

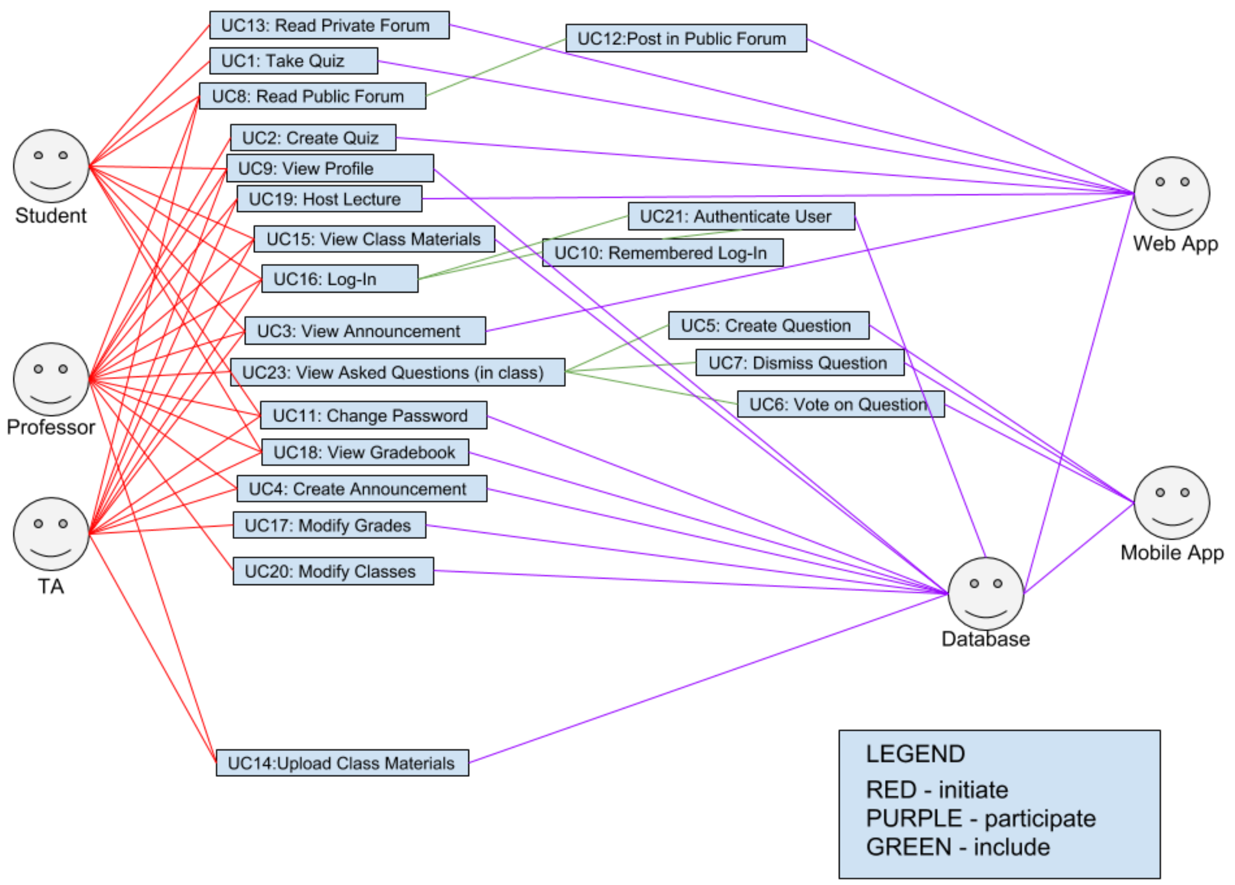
UC-19: Host Lecture - professors/TAs can host online lectures through the use of video calling and virtual whiteboard that students can see in real time.

UC-20: Modify Classes - professors can control the students registered for their classes and students can

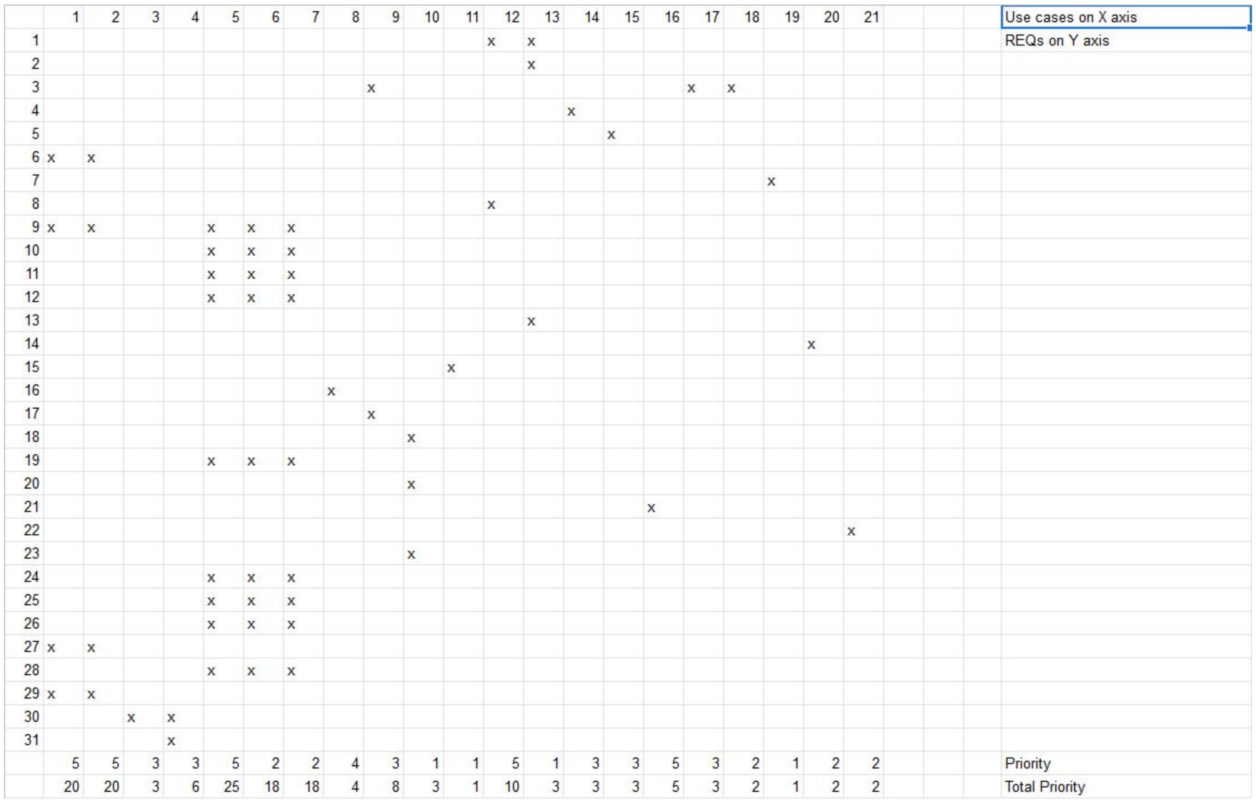
manage all the classes they are registered for

<u>UC-21:</u> View Hub - Users can clearly see exactly where to go for whatever they want to do on the website

ii. Use Case Diagram



iii. Traceability Matrix



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Use cases on X axis
1												x	x									REQs on Y axis
2													x									
3									x								x	x				
4														x								
5															x							
6	x	x																				
7																			x			
8												x										
9	x	x			x	x	x															
10					x	x	x															
11					x	x	x															
12					x	x	x															
13													x									
14																				x		
15											x											
16								x														
17									x													
18										x												
19					x	x	x															
20										x												
21																x						
22																					x	
23										x												
24					x	x	x															
25					x	x	x															
26					x	x	x															
27	x	x																				
28					x	x	x															
29	x	x																				
30			x	x																		
31				x																		
	5	5	3	3	5	2	2	4	3	1	1	5	1	3	3	5	3	2	1	2	2	Priority
	20	20	3	6	25	18	18	4	8	3	1	10	3	3	3	5	3	2	1	2	2	Total Priority

iv. Fully Dressed Descriptions

UC-1	**Take Quiz and Analyze Results**
Related Requirement	REQ-6, REQ-9, REQ-28, REQ-30
Initiating Actor	Student
Actor's Goal	To take quizzes through the app as well as online quizzes and be able to view and analyze results
Participating Actors	Professor, Web App, Phone App, Database
Pre Conditions	The student is logged in and authenticated. The particular class is specified
Post Conditions	None

Flow of Events for Main Success Scenario

->	1.	Student either answers a poll question in class or through an online quiz
<-	2.	System checks the student response(s), and marks them as correct or incorrect. System associates each question with a certain topic.
->	3.	Through the webapp, student can view specific question to see results at any time in the future. The student can also search a particular topic in the database to see all quiz questions related to that topic.

UC-2	**Create and Administer Quiz/In Class Questions**
Related Requirement	REQ-6, REQ-9, REQ-28, REQ-30
Initiating Actor	Professor or TA
Actor's Goal	To create a quiz questions that the students can take in class or a full quiz that the students can take out of class.
Participating Actors	Students, Database, Web App
Pre Conditions	Professor/TA is authenticated and the particular class is specified
Post Conditions	If the quiz question is an in-class question, the question can be triggered at any time by the professor. If the quiz is to be taken outside of class, the quiz form persists for the specified amount of time.

Flow of Events for Main Success Scenario

->	1.	Professor selects to make an out-of- class quiz administered through the web app. The professor specifies the time frame for which this quiz should be available.
<-	2.	The system creates an in-progress data store for this quiz. System displays a page that allows for user inputs in quiz questions and quiz answers
->	3.	Professor can write questions and associated answers in a multiple choice format. The professor can mark questions with a topic. The professor can add or remove questions as desired. Professor saves quiz.
<-	4.	System saves the quiz questions and answers. The system indicates to the professor that the quiz is in-progress.
->	5.	Professor adds more questions or removes questions from the quiz. The Professor publishes the quiz.
<-	6.	System marks the quiz as published. The system sends an announcement to all students that the quiz will be available at the originally specified time. At the specified time, the quiz becomes available to take by students, leading into UC-1.

Flow of Events for Alternate Scenario

->	1.	Professor selects to make an in-class quiz administered through the phone app. The professor specifies the date of the lecture for which this quiz question corresponds to.
<-	2.	The system creates an in-progress data store for this quiz. System displays a page that allows for user inputs in quiz questions and quiz answers
->	3.	Professor can write questions and associated answers in a multiple choice format. The professor can mark questions with a topic. The professor can add or remove questions as desired. Professor saves quiz.
<-	4.	System saves the quiz questions and answers. The system indicates to the professor that the quiz is in-progress.
->	5.	Professor adds more questions or removes questions from the quiz. The Professor publishes the quiz.
<-	6.	System marks the quiz as published. On the specified date, the quiz becomes viewable on the professor's in-class dashboard.
->	7.	Professors triggers individual questions on the day of the lecture.
<-	8.	System recognizes that question was triggered by professor. The system now accepts student responses for that particular question, leading into UC-1.

UC-3	**View Announcement**
Related Requirement	REQ-31, REQ-32
Initiating Actor	Student, Professor, and TAs
Actor's Goal	View announcement published by professor or TA
Participating Actors	Student, Professor, TAs, Database, Web App, Mobile App
Pre Conditions	Professor or TA previously published an announcement. Actor is logged in.
Post Conditions	Announcement is marked as read for actor.

Flow of Events for Main Success Scenario

->	1.	Student, Professor, or TA goes to the announcement tab
<-	2.	System displays all recent announcements and flags unread ones
->	3.	Student, Professor, or TA open an announcement
<-	4.	System flags announcement as read

UC-4	**Create Announcement**
Related Requirement	REQ-31, REQ-32
Initiating Actor	Professor or TA
Actor's Goal	Create announcement viewable to everyone in the class and send notifications via mobile app and or email.
Participating Actors	Students, Database, Web App, Mobile App
Pre Conditions	Professor/TA is authenticated and the particular class is specified
Post Conditions	Send announcements via email and display on mobile and web app.

Flow of Events for Main Success Scenario

->	1.	Professor selects to make an announcement though the web app.
<-	2.	The system creates an in-progress data store for the announcement. System displays a HTML text editor.
->	3.	Professor can write and format the announcement. Professor saves announcement.
<-	4.	System saves the in-progress announcement. The system indicates to the professor that the announcement is being edited and saved.
->	5.	Professor continues to edit until it is ready to publish. Professor publishes announcement immediately or at a certain time.
<-	6.	System marks the announcement as published. The system sends a notification to all students, TAs, and the professor about the announcement at the specified time. The announcement becomes viewable, leading into UC-3

Flow of Events for Alternate Scenario

->	1.	Professor selects to make an announcement though the web app.
<-	2.	The system creates an in-progress data store for the announcement. System displays a HTML text editor.
->	3.	Professor can write and format the announcement. Professor saves

		announcement and professor stops editing.
<-	4.	System saves the in-progress announcement. The system indicates to the professor that the announcement is being edited and saved. Announcement can be edited in the future.

UC-5	Create Question/Feedback (In Class)
Related Requirement	REQ-9, REQ-10, REQ-11, REQ-12, REQ - 19, REQ-25, REQ - 26, REQ - 27, REQ-29
Initiating Actor	Student
Actor's Goal	Create a question for the professor to answer in class
Participating Actors	Professor, System, Database
Pre Conditions	Class is in session. Student is logged in
Post Conditions	Students may vote on question and professor may read and answer question

Flow of Events for Main Success Scenario

->	1.	Student initiates question creation.
<-	2.	System opens question creation workspace with plain text box and option to include name with question.
->	3.	Student writes and edits question until satisfied. Student may choose to remain anonymous when publishing question
<-	4.	System saves question on database, and publishes question to view by other students and the instructor.

UC-6	Vote on Question/Feedback (In Class)
Related Requirement	REQ-9, REQ-10, REQ-11, REQ-12, REQ - 19, REQ-25, REQ - 26, REQ - 27, REQ-29
Initiating Actor	Student
Actor’s Goal	Move better questions higher on professor’s list
Participating Actors	Student, Professor, System, Database
Pre Conditions	Class is in session. Question is previously asked. Student is logged in and authenticated.
Post Conditions	Question priority is updated.

Flow of Events for Main Success Scenario

->	1.	Student opens live question and feedback tab and votes on a question/feedback already made.
<-	2.	System updates the question/feedback score and displays update for all users.
->	3.	Student can see the vote by highlight and can change or take away the vote.

UC-7	**Dismiss Question/Feedback (In Class)**
Related Requirement	REQ-9, REQ-10, REQ-11, REQ-12, REQ - 19, REQ-25, REQ - 26, REQ - 27, REQ-29
Initiating Actor	Professor
Actor's Goal	Address and delete question or feedback from list
Participating Actors	Professor, System, Database
Pre Conditions	Class is in session. Question is previously asked. Professor is logged in and authenticated.
Post Conditions	Question priority is updated.

Flow of Events for Main Success Scenario

->	1.	Professor opens live question and feedback tab on web app and may address a question on the list. Professor dismisses a question.
<-	2.	System deletes question/feedback, but keeps a System updates the question/feedback score and displays update for all users.
->	3.	Student can see the vote by highlight and can change or take away the vote.

UC-8	**Read Forum**
Related Requirement	REQ-1, REQ-16, REQ-8
Initiating Actor	Student, Professor
Actor’s Goal	View past forum posts from other students and the professor
Participating Actors	Student, Professor, System, Database
Pre Conditions	Student or Professor are logged into the web page
Post Conditions	Forum page is updated with the new post

Flow of Events for Main Success Scenario

->	1.	Professor or student opens the window to submit a new forum post and correctly submits it
<-	2.	System updates the new post on the public forum and displays this new post for everyone visiting the page
->	3.	Students and the professor can view the new post and respond to it

UC-14	**Upload and Manage Resources**
Related Requirement	REQ-4, REQ-5, REQ-8,
Initiating Actor	Professor or TA
Actor's Goal	To upload and manage class resources such as lecture slides, notes, homework, etc.
Participating Actors	Students, Database, Web App
Pre Conditions	Professor/TA is authenticated and the particular class is specified
Post Conditions	Uploaded data persists until deleted by the initiating actor

Flow of Events for Main Success Scenario

->	1.	Professor selects documents from local file system to upload to resources application.
<-	2.	The system copies the uploaded documents and populates the resources with them.
->	3.	The professor may move the resources around within the application by creating subdirectories organized in a traditional file system as seen on regular computers.
<-	4.	The system keeps track of file locations to update its UI without making multiple copies of the data.

UC-15	**View Class Materials**
Related Requirement	REQ-5
Initiating Actor	Student
Actor’s Goal	To access and download any files hosted on the class’s resource page
Participating Actors	Students, Database, Web App
Pre Conditions	There are files currently uploaded on the database
Post Conditions	The student can access an offline copy of the file

Flow of Events for Main Success Scenario

->	1.	Student selects a file from a list of uploaded files for a class
<-	2.	The system downloads the file onto the device to a given path

UC-16	**Log In**
Related Requirement	REQ-21, REQ-22,
Initiating Actor	Professor, Student or TA
Actor's Goal	To enter an individualized username and password and be presented with the appropriate web page depending on what kind of user the actor is.
Participating Actors	Database, Web App
Pre Conditions	An account for the user information exists
Post Conditions	The account is logged in until the account is logged out or cookies are cleared

Flow of Events for Main Success Scenario

->	1.	Actor enters login information
<-	2.	The system searches the database for an account with matching user info
<-	3.	The system finds the account that matches the entered info
<-	4.	The system creates and displays the appropriate home page depending on account type

Flow of Events for Alternate Scenario

->	1.	Actor enters login information
<-	2.	The system searches the database for an account with matching user info
<-	3.	The system does not find an account that matches the entered info
<-	4.	The web page notifies the actor that the login information is invalid and does not change otherwise

UC-16	**Log In**
Related Requirement	REQ-21, REQ-22,
Initiating Actor	Professor, Student or TA
Actor's Goal	To enter an individualized username and password and be presented with the appropriate web page depending on what kind of user the actor is.
Participating Actors	Database, Web App
Pre Conditions	An account for the user information exists
Post Conditions	The account is logged in until the account is logged out or cookies are cleared

Flow of Events for Main Success Scenario

->	1.	Actor enters login information
<-	2.	The system searches the database for an account with matching user info
<-	3.	The system finds the account that matches the entered info
<-	4.	The system creates and displays the appropriate home page depending on account type

5. Project Size Estimation

We use the following formula to calculate the Use Case Points for our project:

$$UCP = (UUCW + UAW) * TCF * ECF$$

i. The Unadjusted Use Case Weight (UUCW) is calculated from the sum of total priorities of each of our use cases, which can be found in the traceability matrix from section 3iii above.

$$UUCW = 20 + 20 + 3 + 6 + 25 + 18 + 18 + 4 + 8 + 3 + 1 + 10 + 3 + 3 + 5 + 3 + 3 + 2 + 1 + 2 + 2 = 160$$

ii. The Unadjusted Actor Weight is based on the complexity of all the relevant actors for our use cases.

Actor	Complexity	Weight
Student	Complex	3
Professor	Complex	3
Web Application	Average	2
Mobile Application	Average	2

$$UAW = 3 + 3 + 2 + 2 = 10$$

iii. The Technical Complexity Factor (TCF) is calculated based on the weights and perceived complexity of the following factors:

Technical Factor	Description	Weight	Perceived Complexity	Calculated Factor
T1	In-class features must update in real time with a reasonable response time	1	2	2
T2	Application must be an efficient replacement to already existent software	1	3	3
T3	Processing complexity is fairly minimal	1	1	1

T4	There is very little code reusability, as this is not a continuation of an existing/previous project	1	4	4
T5	Our project will exist on multiple platforms (mobile and web applications)	2	2	4
T6	Ease of use is very important	0.5	5	2.5
T7	Security is not a major concern	1	1	1
T8	We expect minimal system maintenance	1	1	1
			Total:	18.5

We use the formula $TCF = 0.6 + (TF/100)$ to calculate our TCF value of 0.785.

iv. Environment Complexity Factor (ECF) is based on the relevant skills of our team members and how they cater to the needs of our project.

Environmental Factor	Description	Weight	Perceived Impact	Calculated Factor
E1	Not all team members have familiarity with development processes used in our project	1.5	3	4.5
E2	Not all team members have familiarity with applications used in our project	0.5	3	1.5
E3	Our team generally has a solid foundation with object-oriented programming, but not at an expert/professional level	1	4	4
E4	Our team is generally motivated about the problem our program plans to address	1	2	2
E5	Almost all team members are familiar with the programming languages they will be using	1	4	4
			Total:	16

We us the formula $ECF = 1.4 + (-0.03 * EF)$ to calculate our ECF value of 0.92

Finally, we calculate the final UCP value $(160 + 10) * 0.785 * 0.92$ = 123 Use Case Points.

6. Domain Analysis

a. Domain Model

UC-1: Take Quiz and Analyze Results

UC-2: Create and Administer Quiz/In-Class Questions

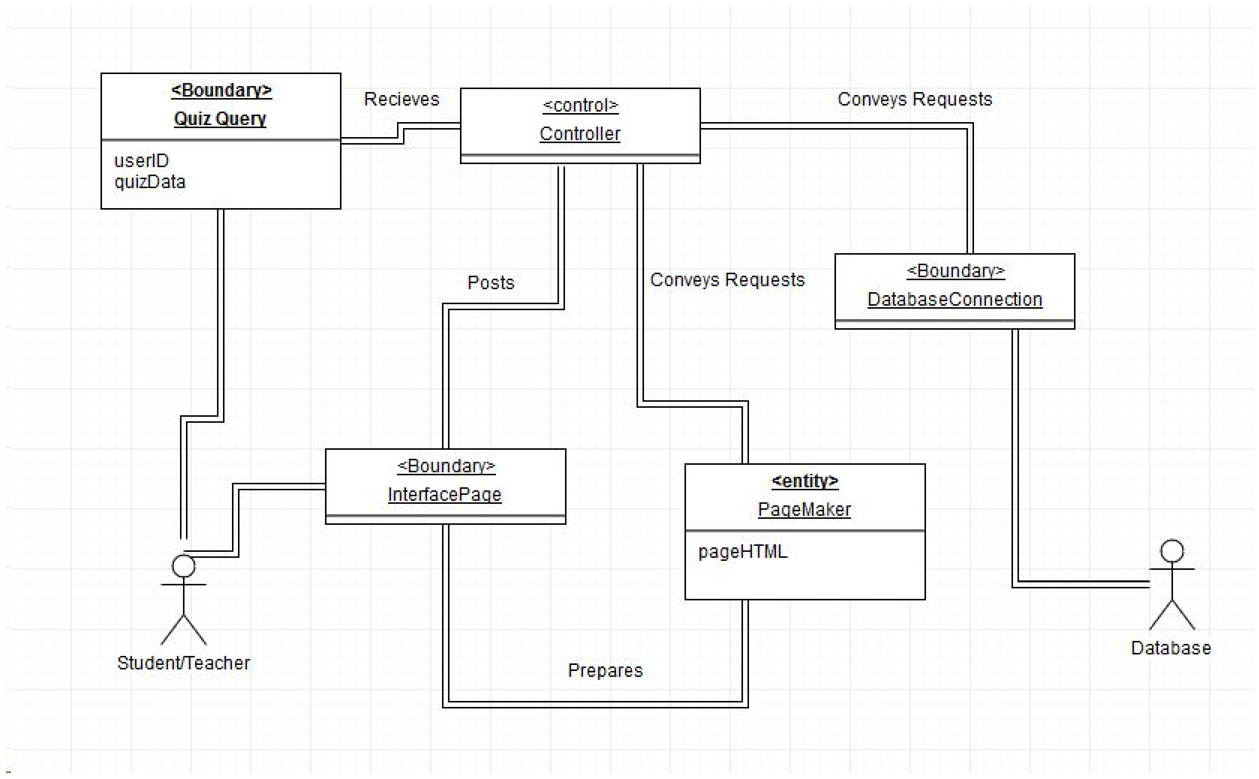
The following concepts, responsibility descriptions, and attribute descriptions can be applied to both use cases. In practice, both use cases will have a different implementation of how the concepts will interact in regards to the specific data they will store and exchange. However, the basic concept to concept structure is kept the same between both use cases, making this design model versatile for many problems in this project.

Responsibility Description	**Type D/K**	**Concept Name**
Coordinates the actions of all concepts associated with this use case and delegate work to other concepts	D	Controller
HTML Document that shows current actions that can be done	K	Interface Page
Render the retrieved records into an HTML for the web browser to display	D	Page Maker
Prepare a query that matches the actor's intended action, either for storage or retrieval	D	Database Connection
Storage of current information regarding the status of the quiz	K	Quiz Request

Concept Pair	**Association Description**	**Association Name**

Controller <-> Page Maker	Controller passes request to Page Maker and receives the HTML to display	Conveys requests
Controller <-> Interface Page	Conveys info from "conveys requests" to the interface page to display	Control interface
Controller <-> Database Connection	Controller makes queries to database connection either storing or retrieving information and database sends confirmation back	Make query
Page Maker <-> Interface Page	Page maker makes page for interface page display	Page display
Interface Page <-> Quiz Request	Quiz request is updated based on user interaction with the interface page	Take quiz
Quiz Request <-> Controller	Controller handles quiz request and conveys to database connection	Quiz query

Concept	**Attributes**	**Attribute Description**
Quiz Query	User Identity	Used to determine the actor's credentials which determines what data the actor can access as well as what data they can submit
	Quiz data	Quiz answers data submitted or viewed by the actor for communication with data storage. Includes any other relevant quiz data
Page maker	Page html	Actual HTML to display on the page



UC-3: View Announcement

UC-4: Create Announcement

The Domain of creating and viewing announcement is similar to that of creating a quiz. In both cases, the user will be verified based on ID if the user can create an announcement. Announcements and quizzes will be edited on an HTML textbox, but the announcements will have less functionality as it does not require input.

UC-5: Create Question/Feedback

UC-6: Vote on Question

UC-7: Dismiss Question/Feedback

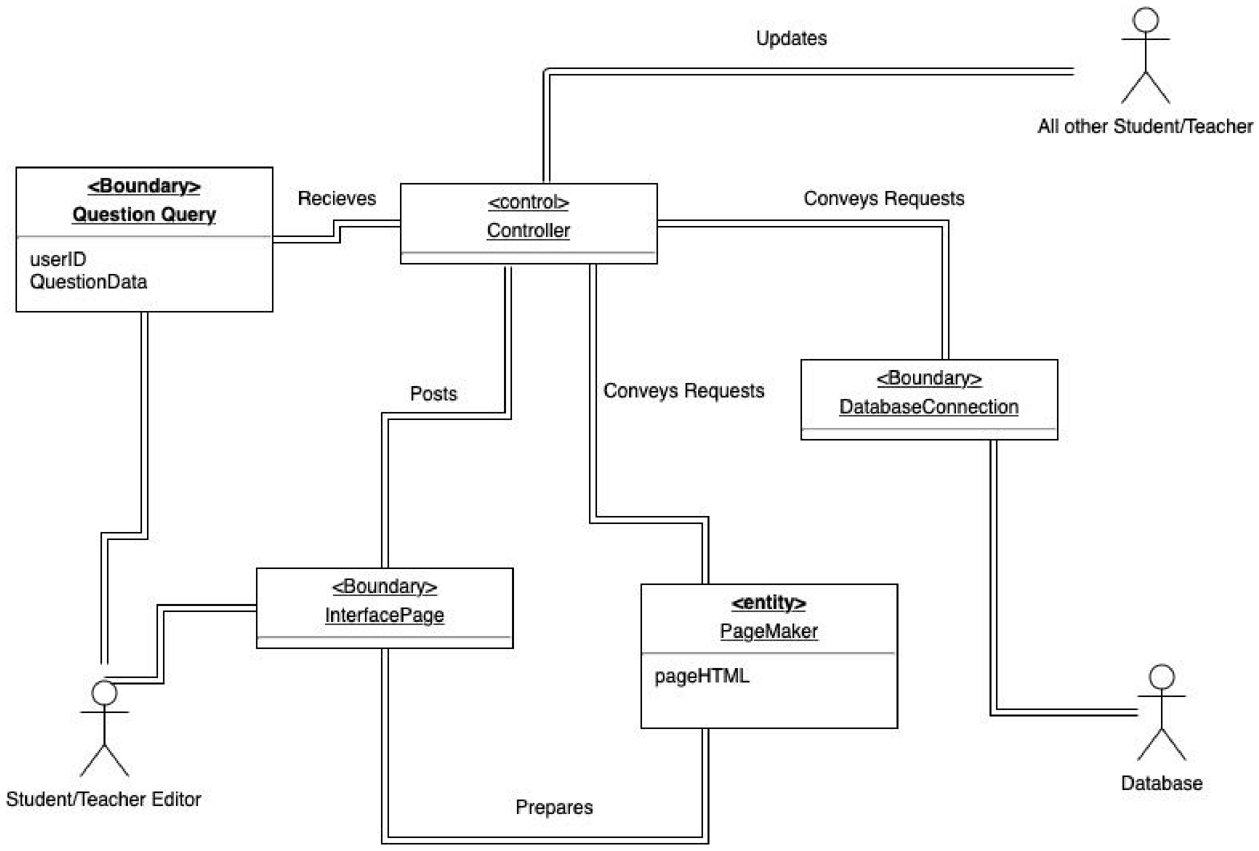
The process of creating, voting, and dismissing a question happens with a similar structure to quizzes and announcements. Questions and Feedback are updated live for all users on the page whenever a vote, creation, or dismissal happens from a user. The capabilities of the user are determined by the userID. If this is not feasible, a simple prompt will appear on the bottom of the poll indicating the user to refresh the page to see updates. *The Domain will vary between the webapp and the mobile app.

Responsibility Description	**Type D/K**	**Concept Name**
Coordinates the actions of all concepts associated with this use case and delegate work to other concepts	D	Controller
HTML Document that shows the current Question/Feedback list. Includes buttons to create and vote.	K	Interface Page
Render the page into an HTML for the web browser to display for the users.	D	Page Maker
Prepare a query that matches the actor's intended action, either for storage, update, or retrieval.	D	Database Connection
Storage of current information regarding the status of the question. Can be votes, dismissal, or creation.	K	Question Request

Concept Pair	**Association Description**	**Association Name**
Controller <-> Page Maker	Controller passes request to Page Maker and receives the HTML to display	Conveys requests
Controller <-> Interface Page	Conveys info from "conveys requests" to the interface page to display.	Control interface
Controller <-> Database Connection	Controller makes queries to database connection either storing, retrieving, or updating information. Database sends data on question back.	Make query

Page Maker <-> Interface Page	Page maker makes page for interface page display. Updates whenever a command is sent that the database has been updated (Live updates).	Page display
Interface Page <-> Announcement Request	Editing user interaction with the interface page. Can be creation of question/feedback or vote. In the case of the teacher, a dismissal.	Publish Announcement
Question Request <-> Controller	Controller handles question request and conveys to database connection. Request does not go directly to the database.	Announcement query
Database Connection <-> Controller	Send command to update question list for all users when update is made for live feedback.	Update request

Concept	**Attributes**	**Attribute Description**
Question Query	User Identity	Used to determine the actor's credentials which determines what data the actor can access as well as what data they can submit
	Question data	Announcement answers data submitted or viewed by the actor for communication with data storage. Includes Question, Votes, and Creator (can be anonymous)
Page maker	Page html	Actual HTML to display on the page, includes HTML textbox



UC-14: Upload Class Resources

These descriptions are similar to the descriptions for other use cases. They are modified to accommodate the use of "resources" which refers to files that an actor may want to access or manage within the software. This is a type of data that will be stored and transferred through the use of other concepts.

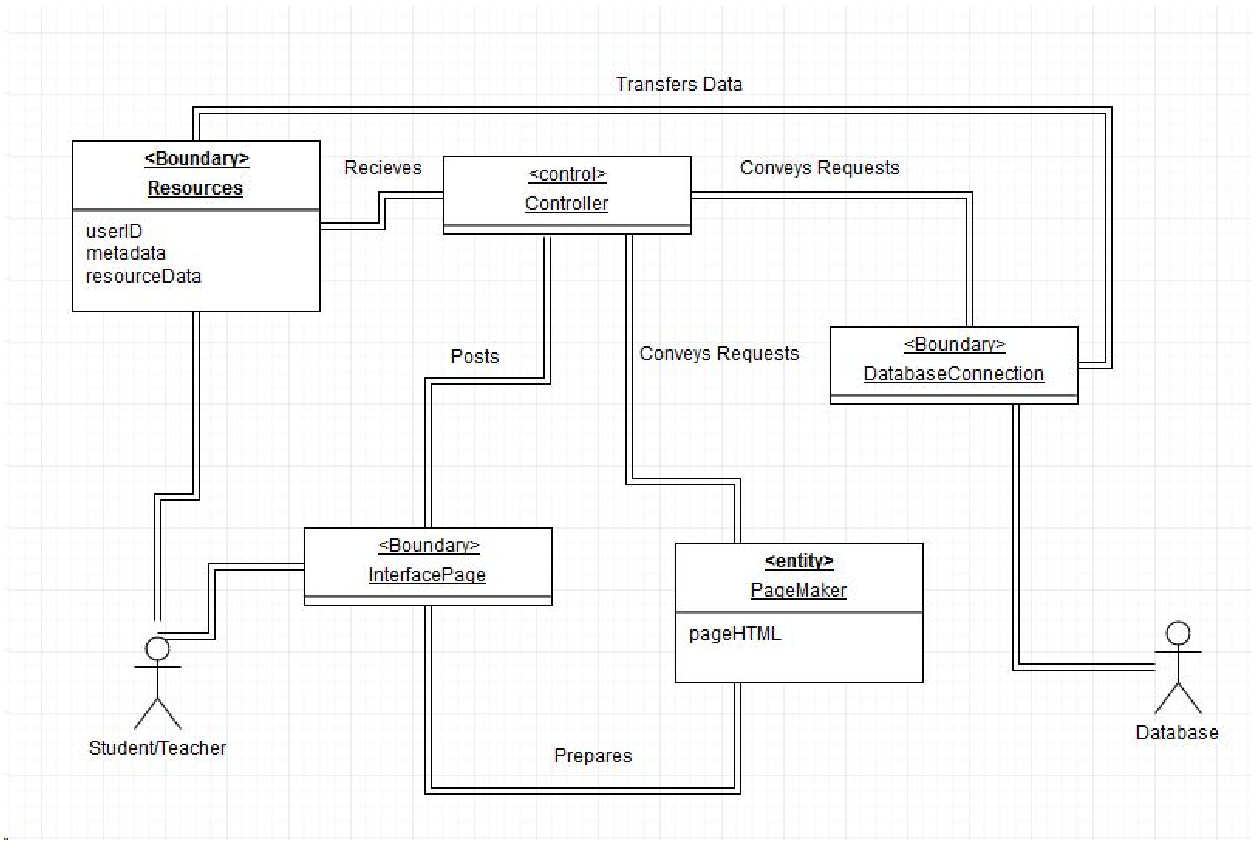
Responsibility Description	T	Concept Name

Coordinates the actions of all concepts associated with this use case and delegate work to other concepts	D	Controller
HTML Document that shows current actions that can be done	K	Interface Page
Render the retrieved records into an HTML for the web browser to display	D	Page Maker
Prepare a query that matches the actor's intended action, either for storage or retrieval	D	Database Connection
Resources that can be accessed, uploaded, and managed	K	Resources

Concept Pair	**Association Description**	**Association Name**
Controller <-> Page Maker	Controller passes request to Page Maker and receives the HTML to display	Conveys requests
Controller <-> Interface Page	Conveys info from "conveys requests" to the interface page to display	Control interface
Controller <-> Database Connection	Controller makes queries to database connection either storing or retrieving information and database sends confirmation back	Make query
Page Maker <-> Interface Page	Page maker makes page for interface page display	Page display
Interface Page <-> Database Connection	Interface interactions trigger database requests via the controller.	Interact Resource
Resources <-> Database Connection	Resources are stored, conveyed, and managed through the database	Access Resource

	connection	

Concept	Attributes	Attribute Description
Resources	User Identity	Used to determine the actor's credentials which determines what data the actor can access as well as what data they can submit
	metadata	Metadata information of the resource such as upload date, file type, etc
	data	The resource data itself
Page maker	Page html	Actual HTML to display on the page



b. System Operation Contracts

UC-1: Take Quiz and Analyze Results

- Precondition: The student is logged in and authenticated. The particular class is specified
- Postcondition: Results are stored and accessible by relevant actors.

UC-2: Create and Administer Quiz/In-Class Questions

- Precondition: Professor/TA is authenticated and the particular class is specified
- Postcondition: If the quiz question is an in-class question, the question can be triggered at any time by the professor. If the quiz is to be taken outside of class, the quiz form persists for the specified amount of time.

UC-3: View Announcement

- Precondition: Professor or TA previously published an announcement. Actor is logged in
- Postcondition: Announcement is marked as read for actor.

UC-4: Create Announcement

- Precondition: Professor/TA is authenticated and the particular class is specified
- Postcondition: Send announcements via email and display on mobile and web app.

UC-5: Create Question/Feedback

- Precondition: Class is in session. Student is logged in
- Postcondition: Students may vote on question and professor may read and answer question

UC-6: Vote on Question

- Precondition: Class is in session. Question is previously asked. Student is logged in and authenticated.
- Postcondition: Question priority is updated.

UC-7: Dismiss Question/Feedback

- Precondition: Class is in session. Question is previously asked. Professor is logged in and authenticated.
- Postcondition: Question priority is updated and is dismissed from screen.

UC-8: Read Forum

- Precondition: Student or Professor are logged into the web page.
- Postcondition: Forum page is updated with the new post and comments.

UC-14: Upload Class Resources

- Precondition: Professor is logged in and authenticated, and has prepared material to share with class
- Postcondition: Class material is made available for everyone to view and download.

c. Mathematical Model

For the in-class questions, the algorithm to determine if the question will be displayed on the professors screen will be determined by Number of upvotes within a certain time frame.

Considerations:

- It is understood that it will take a few moments for the student to come up with a question after the material is discussed and to type it.
- There will also be some time for students to view the said question and decide whether or not they want to vote on it.
- However, having a question pop-up about a topic that was discussed a while back in lecture would inhibit progress in the lecture and slow it down.

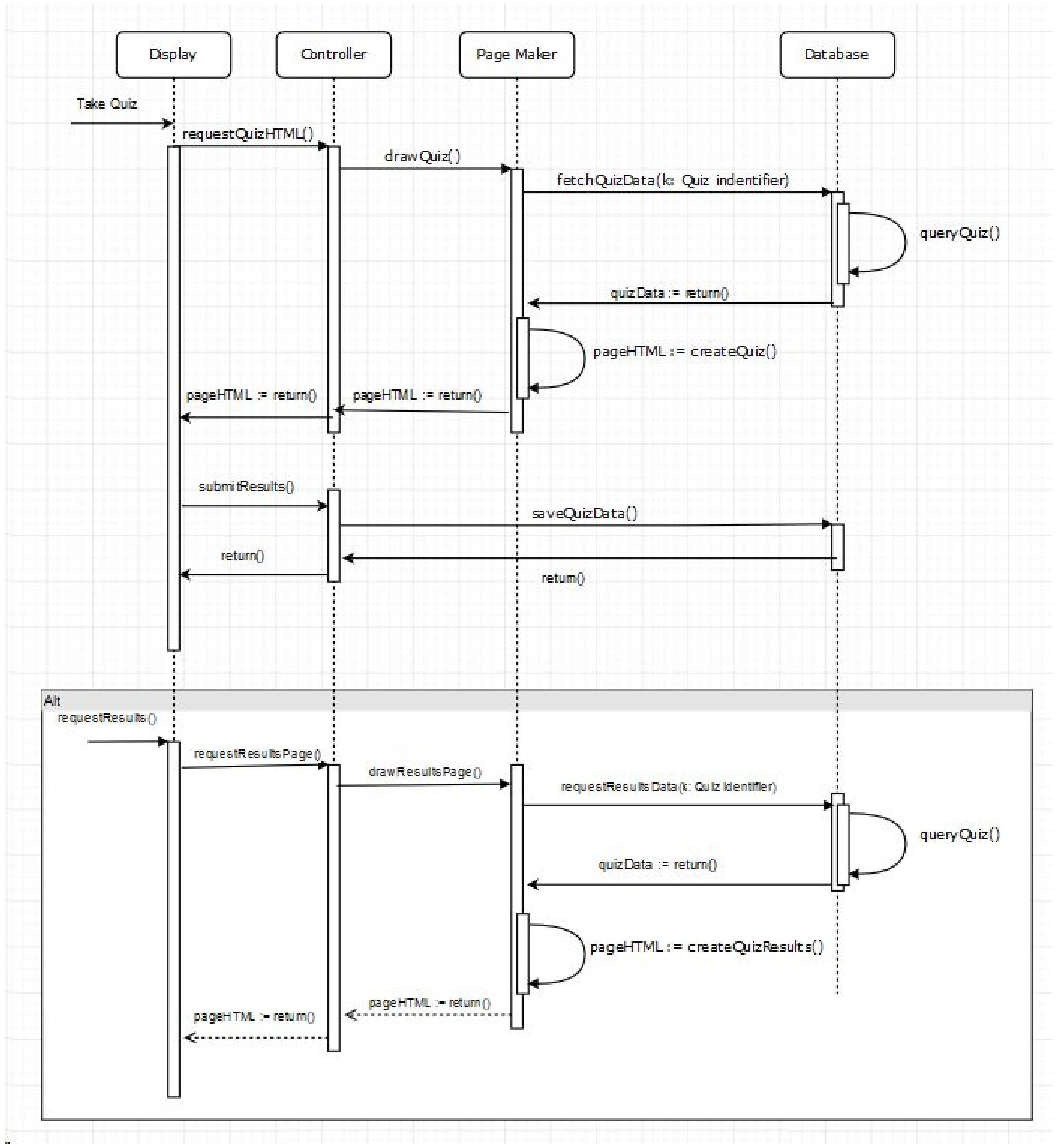
Due to this, there will be a time limit of 150 seconds after a question is asked. If the question reaches a threshold of 10% upvotes within this time, then the Professor will get a notification on their screen in order to answer the question. So for example, if the class size is 200, and a Student asks a question through the app, there is a 150 second internal timer within which the question must get 20 more upvotes than downvotes for it to be sent to the professor in-class as a notification. This time interval is selected because it gives students in the class ample time to check the questions that have been asked through the app. Furthermore, it helps to ensure that the question itself is still topical to the class discussion at hand and would not divert the lecture. In this example, 20 upvotes is selected as the threshold because it represents 10% of the class which is a significant enough portion of the class that ensures that the question is a common question among students.

The decisions and research done to arrive at the 150 second time limit and the %threshold are explained in more detail in the Algorithms section, section 10.

7. Interaction Diagrams

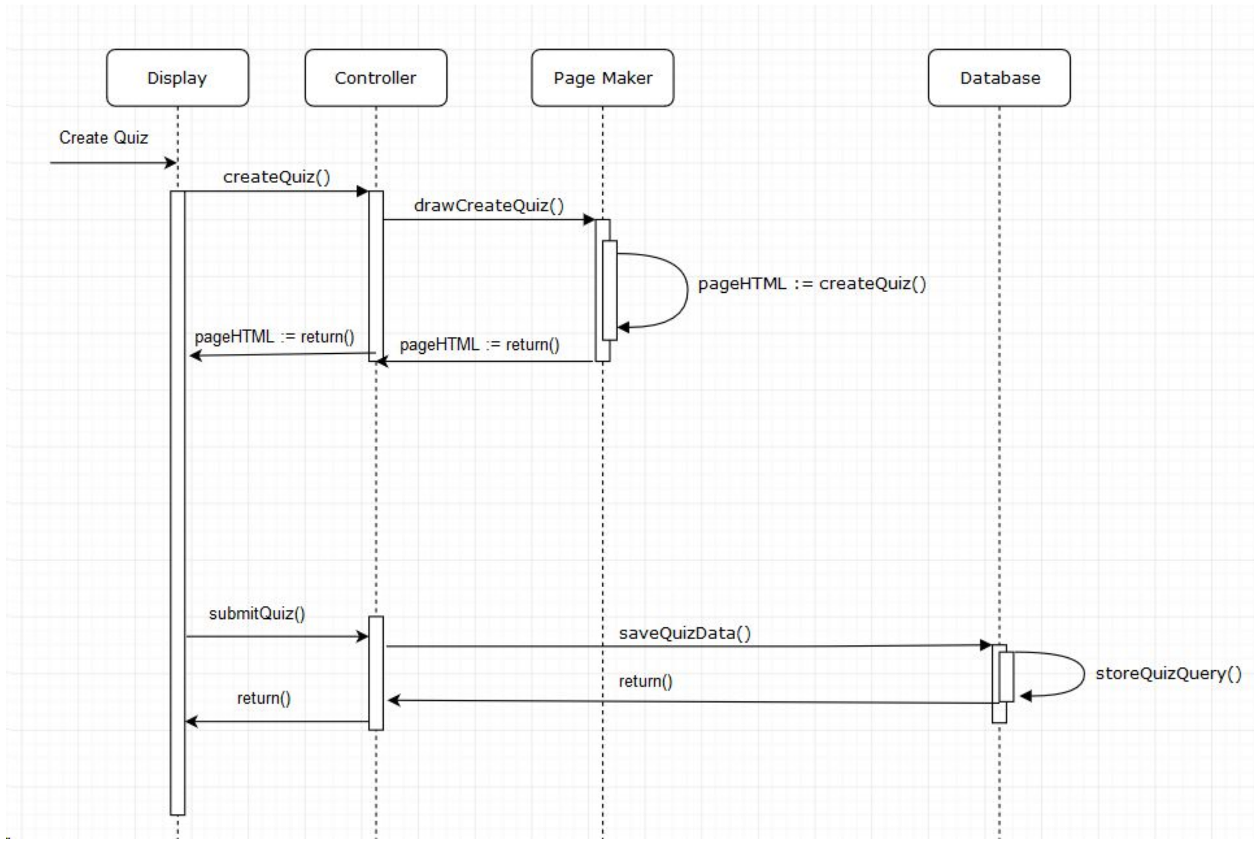
Sequence diagram for UC 1: Take quiz and analyze results

The main scenario shown is for taking the quiz. The alternate scenario shown is for when a student requests quiz results. Both use the same domain concepts in similar ways, so they are shown in the same sequence diagram.



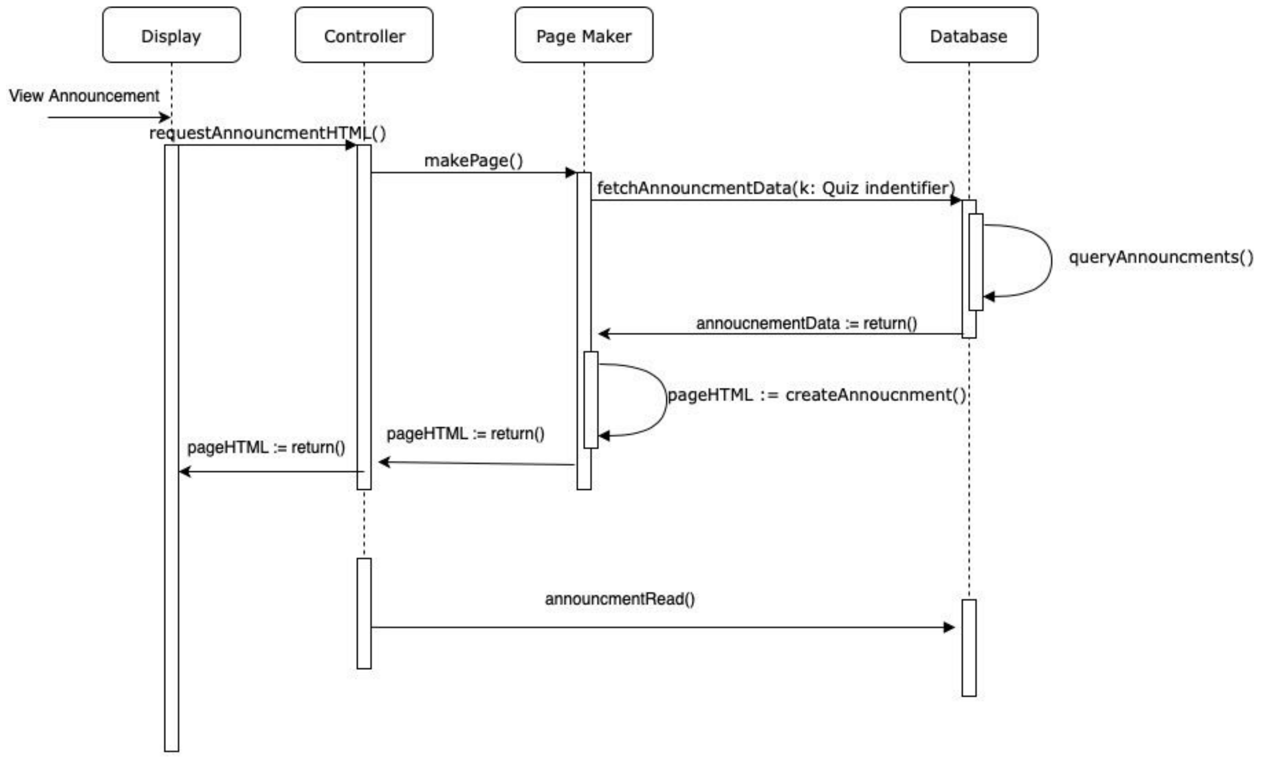
<u>Sequence diagram for UC 2: Create and Administer Quiz</u>

The sequence diagram below shows the sequence for creating a quiz. This includes the page maker which draws the appropriate page for quiz creation based on the particular initial request.



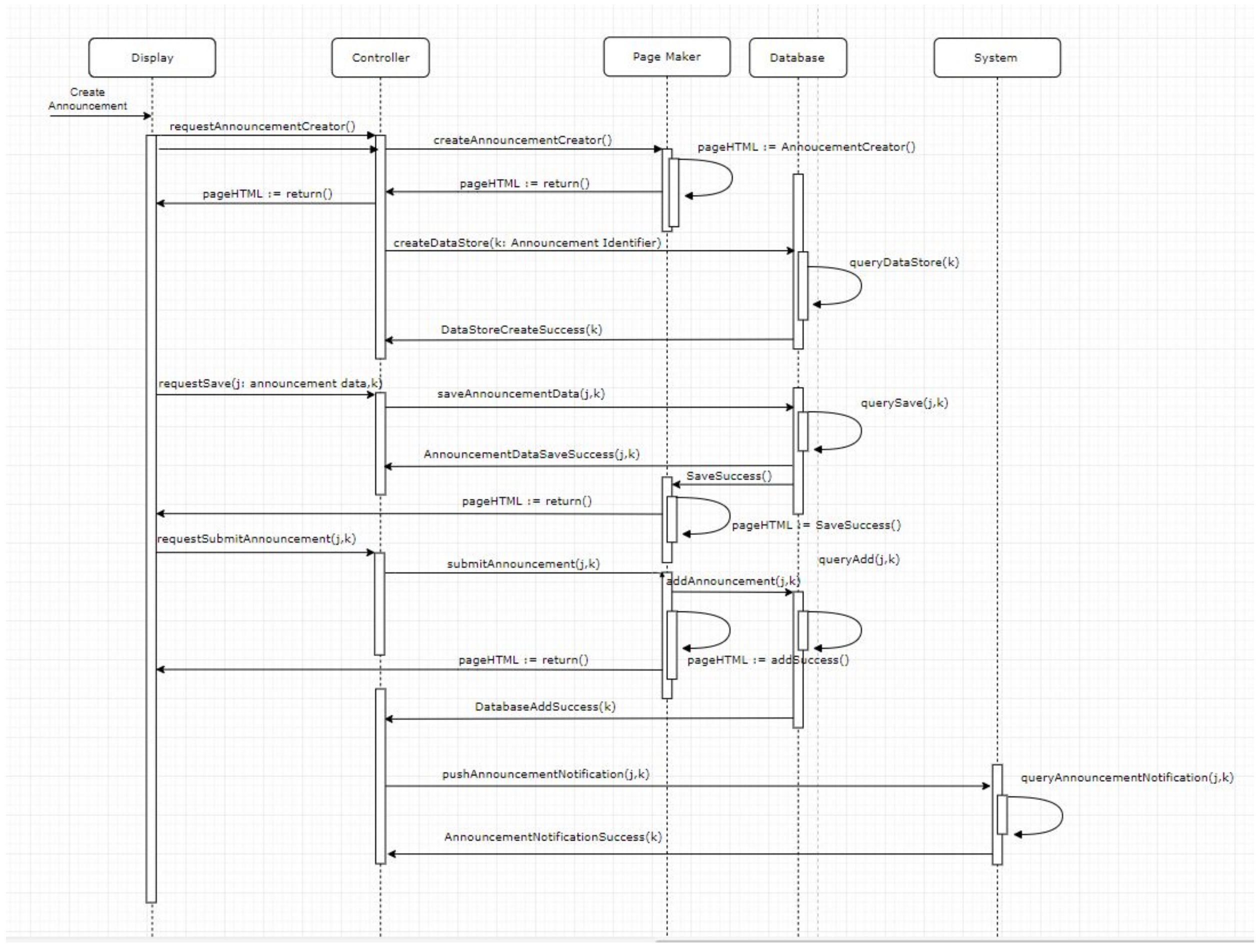
Sequence diagram for UC 3: View Announcement

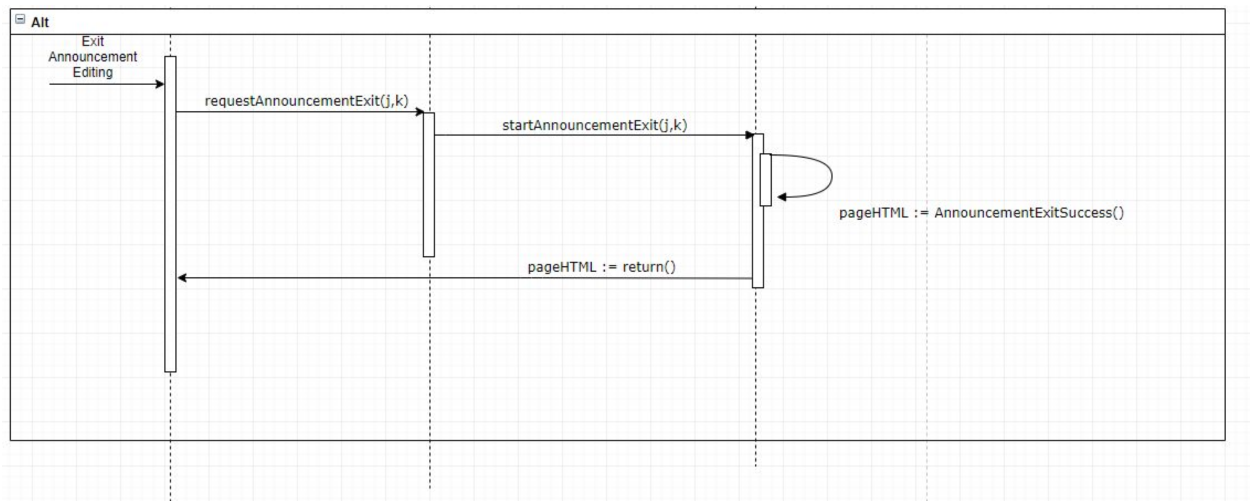
The sequence diagram below shows the sequence for viewing an Announcement.



<u>Sequence diagram for UC 4: Create Announcement</u>

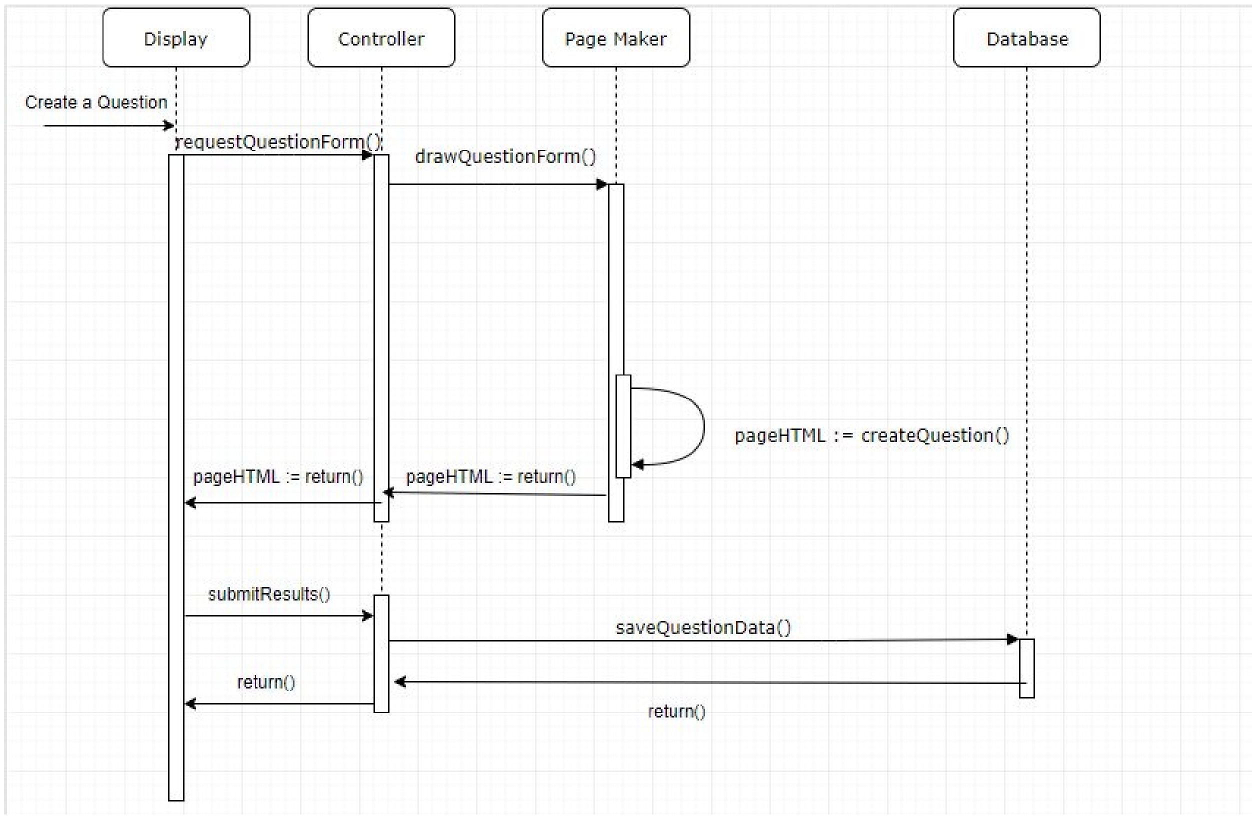
The following sequence diagram shows the the flow of object responsibility distribution when an authorized user (professors and teaching assistants) decides to create an announcement. Instances such as saving the announcement for another time, exiting the announcement creation process, and creating notifications through the system for announcements made have been included.


Display
Controller
Page Maker
Database
System
Create Announcement
requestAnnouncementCreator()
createAnnouncementCreator()
pageHTML := AnnoucementCreator()
pageHTML := return()
pageHTML := return()
createDataStore(k: Announcement Identifier)
queryDataStore(k)
DataStoreCreateSuccess(k)
requestSave(j: announcement data,k)
saveAnnouncementData(j,k)
querySave(j,k)
AnnouncementDataSaveSuccess(j,k)
SaveSuccess()
pageHTML := return()
pageHTML := SaveSuccess()
requestSubmitAnnouncement(j,k)
submitAnnouncement(j,k)
queryAdd(j,k)
addAnnouncement(j,k)
pageHTML := return()
pageHTML := addSuccess()
DatabaseAddSuccess(k)
pushAnnouncementNotification(j,k)
queryAnnouncementNotification(j,k)
AnnouncementNotificationSuccess(k)


Alt
Exit Announcement Editing
requestAnnouncementExit(j,k)
startAnnouncementExit(j,k)
pageHTML := AnnouncementExitSuccess()
pageHTML := return()

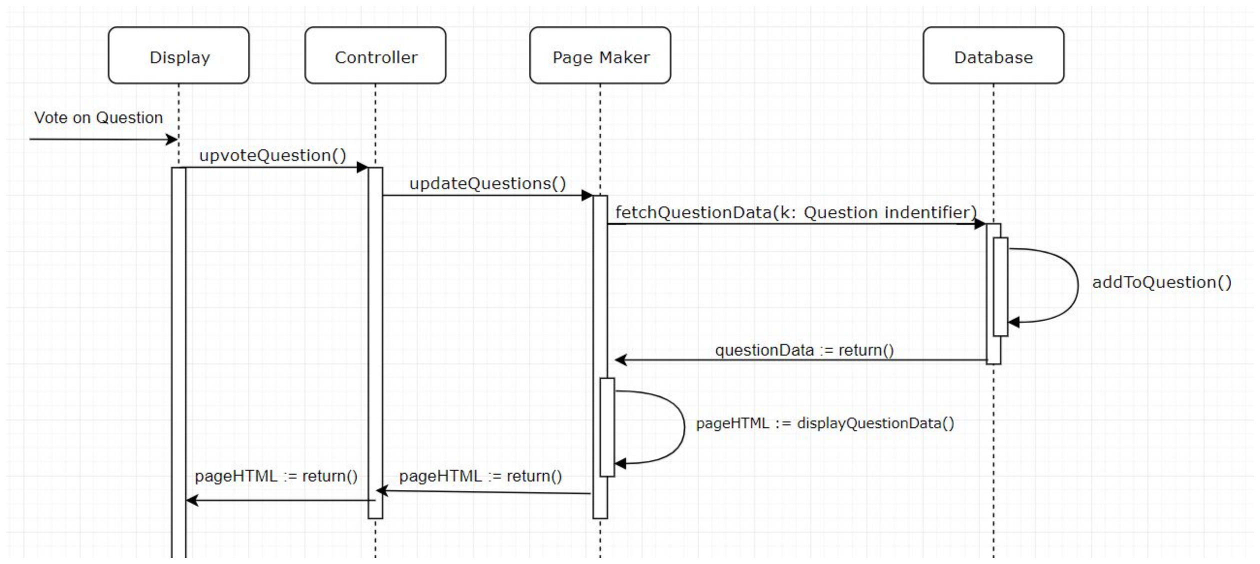
Sequence diagram for UC 5: Create Question/Feedback

The sequence diagram below shows the sequence for a student creating a question for the professor. This includes the page maker which draws the form for question creation once requested.



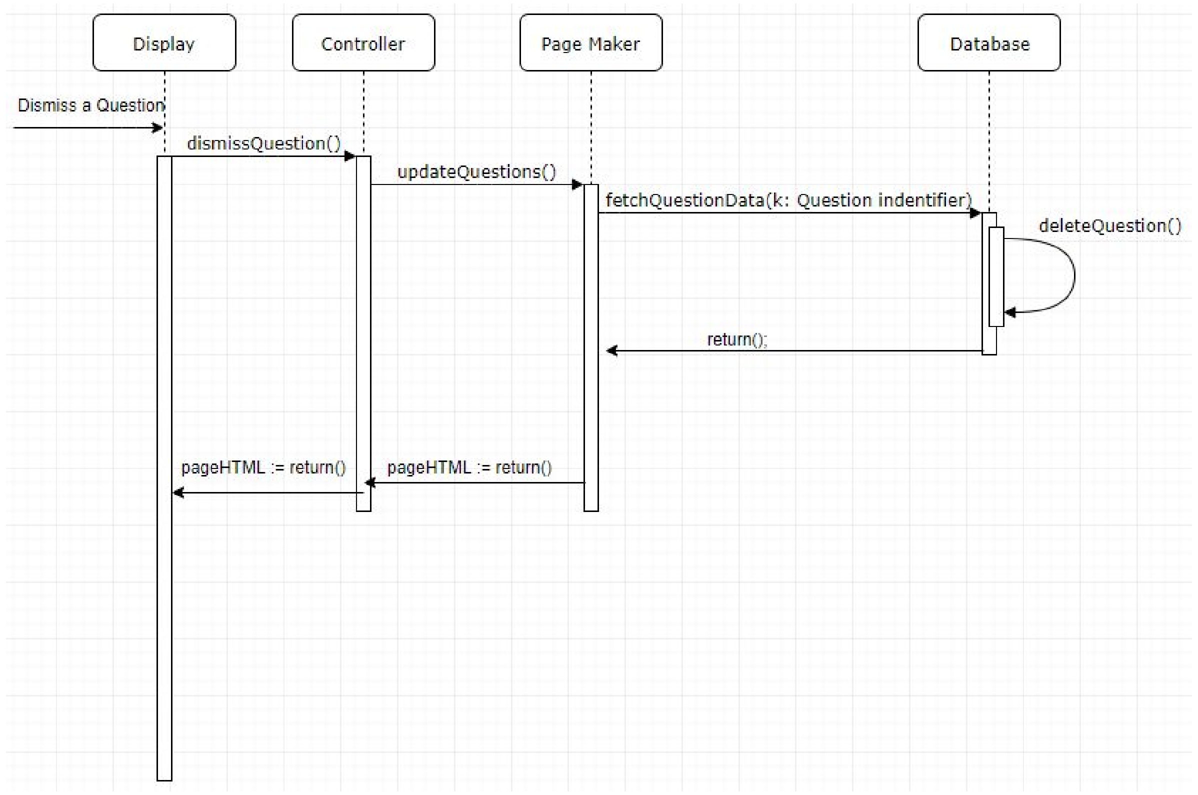
Sequence Diagram for UC-6: Vote on Question:

The diagram below shows the sequence for voting on a question. The user will vote on a question they deem relevant or helpful, and a request is made to update the score for the question in the database, and display the updated score on the page.



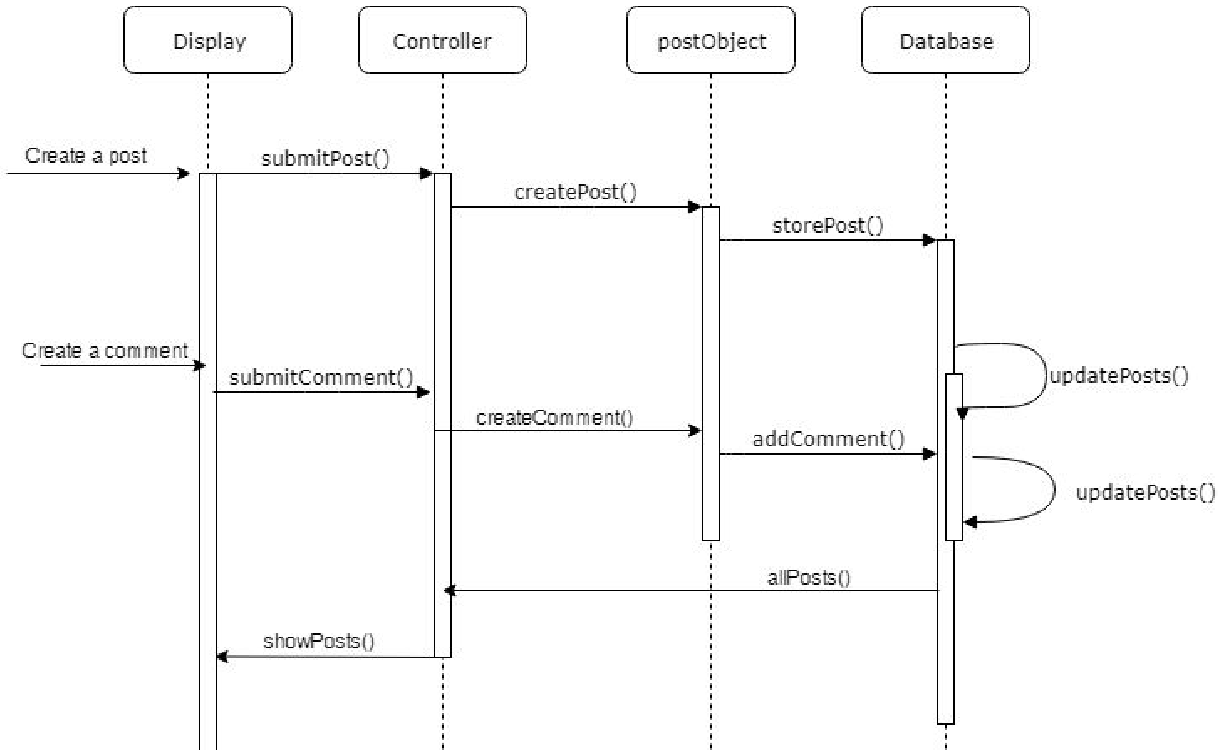
<u>Sequence diagram for UC 7 : Dismiss Question</u>

The sequence diagram below shows the sequence for dismissing an asked question. When a professor dismisses a question, he does not feel that it is necessary to address it, so the question is deleted.



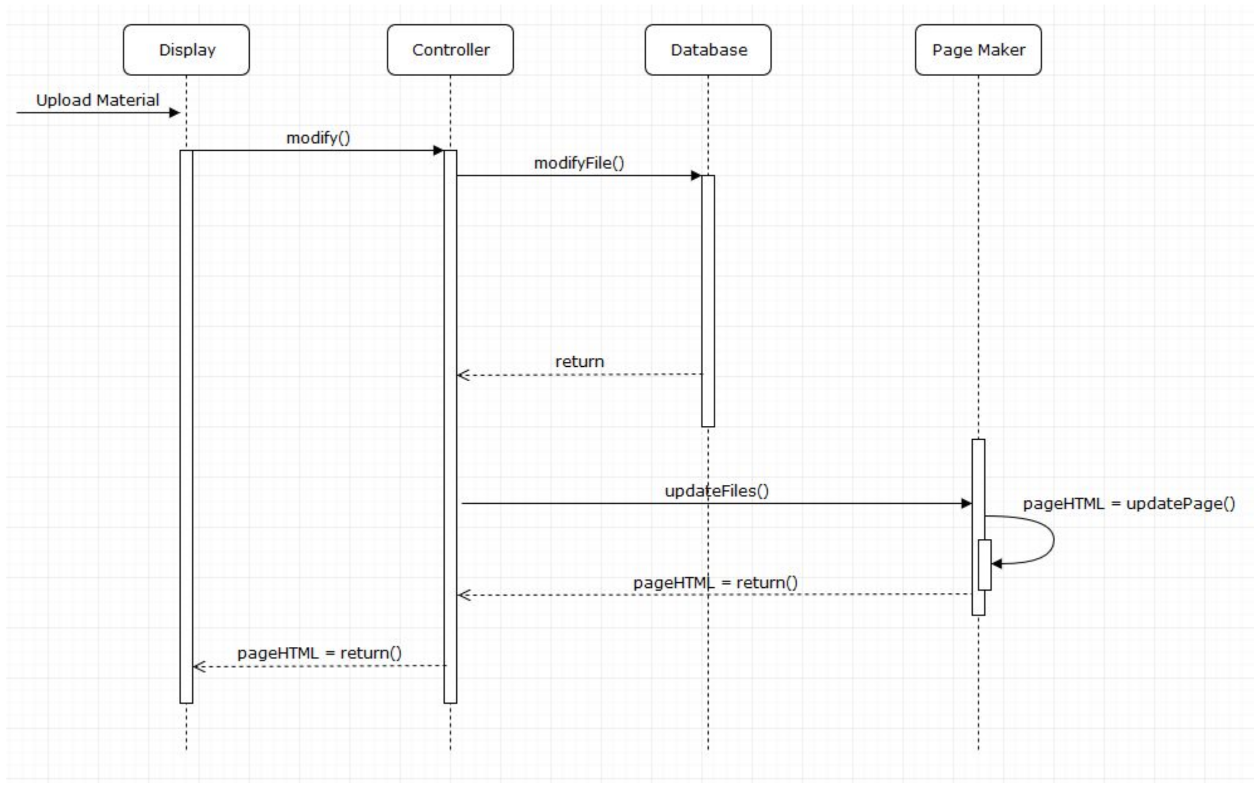
<u>Sequence Diagram for UC-8: Read Forum:</u>

The diagram below displays the sequence in which the users will interact with and read the forum. The user can submit a new forum post which will then be viewable by other users. The other users can then respond to the forum post.



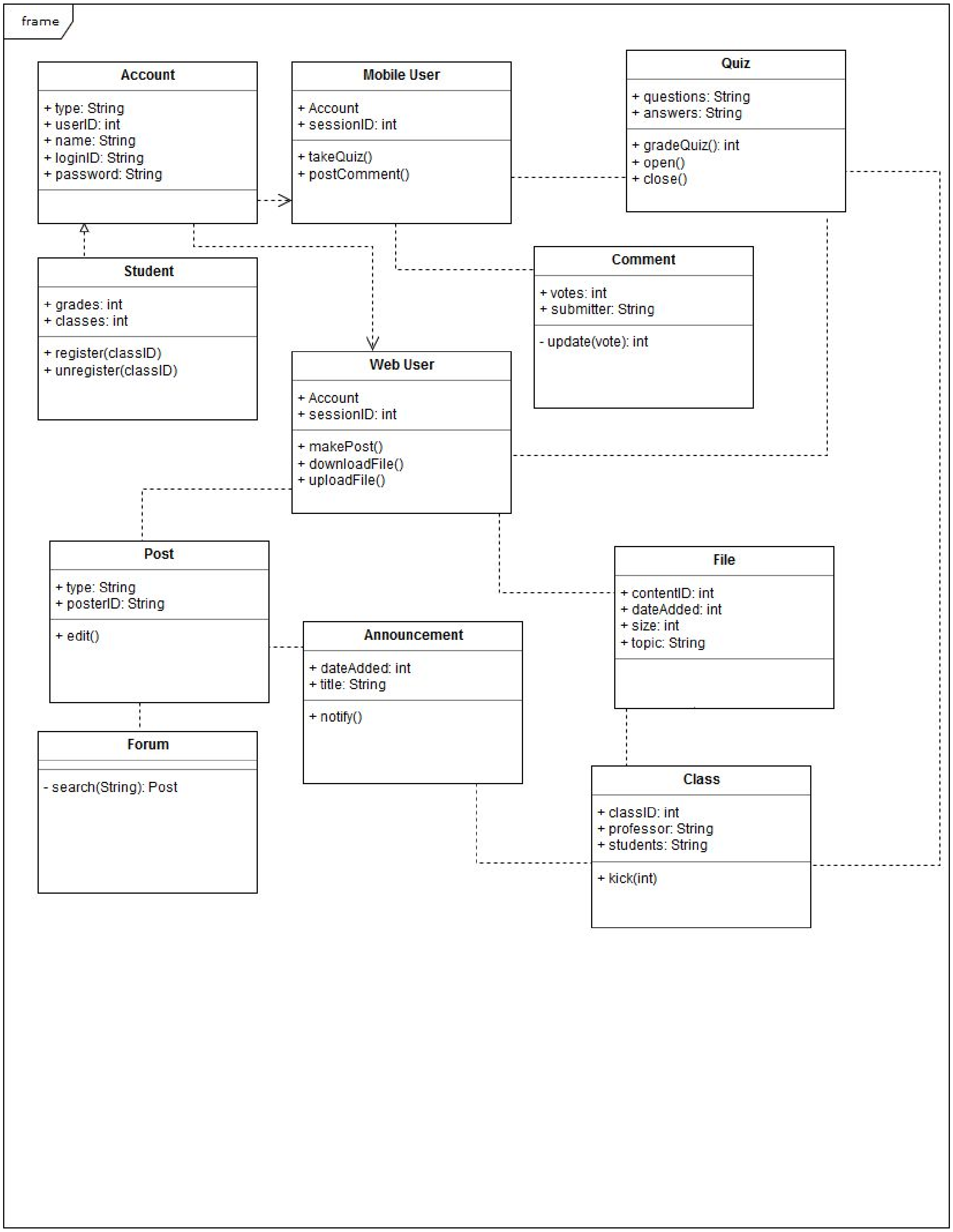
<u>Sequence Diagram for UC-14: Upload and Manage Resources:</u>

The diagram below shows the sequence for modifying course materials uploaded online. The professor has to ability to upload, delete, and move files that are available to all other users. The most up to date version of the files will be displayed and updated as modifications are made.



8. Class Diagram and Interface Specification

a. Class Diagram



b. Data Types and Operation Signatures

1. Account
    a. Attributes
        i. Type: whether this account is professor, TA, or student
        ii. userID: number identification for the account
        iii. Name: name registered to account
        iv. loginID: login user name
        v. Password: login password
2. Student
    a. Attributes
        i. Grades: grades of students classes
        ii. Classes: classes student is registered in
    b. Methods
        i. register(classID): registered a student for the class with classID
        ii. unregister(classID): unregisters a student for the class
3. Mobile User
    a. Attributes
        i. Account: The account that is currently logged into the mobile device
        ii. sessionID: unique session ID for identification
    b. Methods
        i. takeQuiz(): starts a quiz to be taken on the app
        ii. postComment(): creates a comment for other students and professor to view
4. Web user
    a. Attributes
        i. Account: account currently logged into web page
        ii. sessionID: unique session ID for identifications
    b. Methods
        i. makePost(): makes a post to upload to forum or announcement if professor
        ii. downloadFile(): downloads a file from the database
        iii. uploadFile(): uploads a file to the database if professor
5. Post
    a. Attributes
        i. Type: what kind of account created the post
        ii. posterID: id of the user that created the post
    b. Methods
        i. edit(): edits the post
6. Announcement
    a. Attributes
        i. dateAdded: the date the announcement was added
        ii. Title: title of the announcement
    b. Methods

        i. notifty(): sends out a notification to students about the announcement

7. Forum
    a. Methods
        i. search(): searches for a specific post
8. Class
    a. Attributes
        i. classID: unique class ID
        ii. Professor: the professor of the clas
        iii. Students: list of students currently registered to take the class
    b. Methods
        i. kick(userID): kicks the user with the userID from the class
9. Quiz
    a. Attributes
        i. Questions: the questions of the quiz
        ii. Answers: the answers of the quiz
    b. Methods
        i. gradeQuiz(): returns the grade of the quiz with the given answers
        ii. open(): opens the quiz to be available to complete
        iii. close(): closes the quiz and prevents answers to be submitted
10. File
    a. Attributes
        i. contentID: unique content ID
        ii. dateAdded: the date the file was added
        iii. Size: the size of the file
        iv. Topic: a topic the professor says the file pertains to
11. Comment
    a. Attributes
        i. Votes: the current voting score the students have given the comment
        ii. Submitter: who submitted the comment
    b. Methods
        i. update(vote): updates the vote count of the comment

c. Traceability Matrix

Use Cases	Account	Mobile User	Quiz	Student	Comment	Web User	Post	File	Announcement	Forum	Class
UC-1		x	x	x							x
UC-2	x		x								x
UC-3				x					x		
UC-4									x		
UC-5		x			x						x
UC-6		x		x							x
UC-7	x										x
UC-8						x	x			x	
UC-9	x										x
UC-10	x										
UC-11	x										
UC-12						x	x			x	
UC-13				x		x	x			x	
UC-14	x					x		x			
UC-15				x		x		x			x
UC-16	x			x							
UC-17	x					x					
UC-18	x			x		x					
UC-19	x	x									x
UC-20	x			x							x
UC-21	x			x		x					

d. Design Patterns

For the most part, almost all of the initial sequence diagrams already followed the command design pattern. This was to be expected as most of the sequences already started with a similar framework in there being a client, server and middle man. There is the display which serves as the client, which sends the necessary commands to the controller, which serves as the middle man, which finally calls the necessary methods in the page maker which serves as the receiver. The sequences do not exactly mirror what was actually done in the project to a tee in terms of class separation, however the process and the pattern is still there because it was decided that this would be the best way to organize these interactions due to the foundation already being there. There was a little refinement to more closely follow the design pattern, however not much had to change.

9. System Architecture and System Design

a. Architectural Styles

For this project we are using a *layered* architectural design. Our layers consist of:

1. UI Layer / Presentation Layer

   This layer will be solely be used for UI and UX concepts. It will be easy to change and edit without impacting the application as a whole.

2. Application Layer / Service Layer

   Beginning here, all of our services will exist on this layer and above. Concepts and features like polling and announcements are such applications.

3. Domain Layer / Logic Layer

   This layer will control the logic of the application and how they work together.

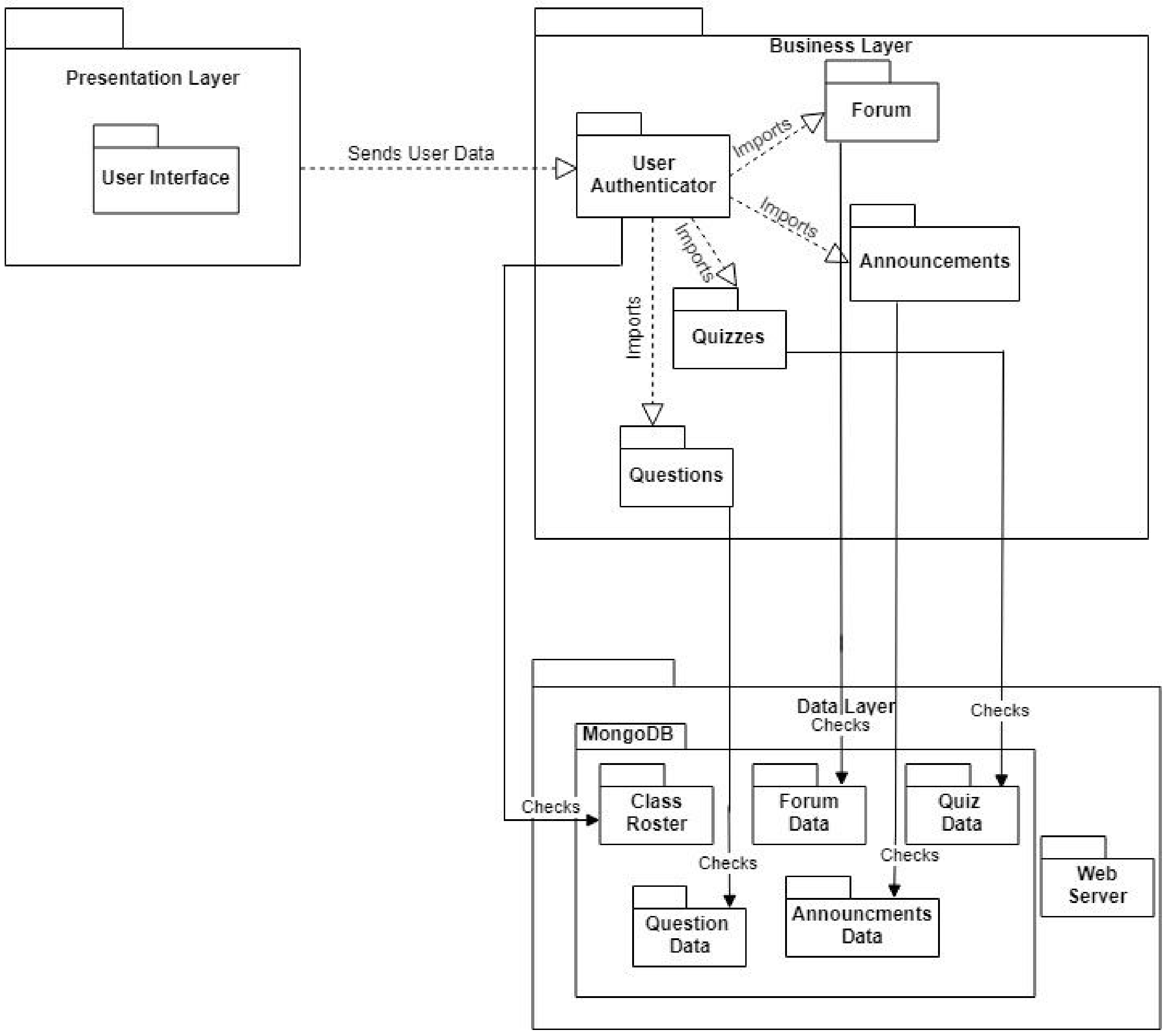
4. Persistence Layer / Data Access Layer

   Our database will exist on this layer.

The benefits of using a layered style for our app are numerous. Firstly, lack of dependence on other layers will allow for more code reuse, which is key in a team of this size and on an app on multiple platforms. Second, the encapsulation and abstractions of each layer makes testing easier. Additionally, as the app gets more and more featured, it will be easier to maintain. Lastly, performance is not a big requirement for this application, which is the main drawback of a layered system.

b. Identifying Subsystems

UML Package Diagram



The UML diagram above shows the connections of the subsystems of our application. The User Authenticator is responsible for importing the necessary packages from the business layer, which are subsequently displayed on the user interface. Our data layer is primarily powered by our MongoDB database, which stores user information, quizzes, forum posts, etc.

c. Mapping Subsystems to Hardware

1. Mobile App - Android and IOS

2. Professor Web app - Web Browser
3. Student Web app - Web Browser
4. Database - AWS EC2 + S3

d. Persistent Data Storage

We decided to use a MongoDB-powered database to manage data for the different components of our application. The main actors in our application include students and professors, both of which will be stored as users in our database. Each user will have a unique user ID and password that will be used to authenticate the users to access their relevant courses, files, and grades through their Minerva portal. Our database will allow us to retain permissions for different users, such as allowing teachers to edit grades and upload materials, and for students to view only their grades for assignments and assessments. Some components of our project will rely heavily on database for storage, such as our forums, and gradebook features. Forums will be stored as threads of posts, which will store relevant information such as the user that created them, the time at which they were created, and of course the actual content of the post. Gradebooks will exist for each unique course, and a numerical grade for each assignment will be given to each unique student in the class. A major reason we chose MongoDB to implement our database was because of its flexibility and ability to easily store many different file types of varying sizes. This will be incredibly useful for the "resource folder" portion of our application, where teachers will be able to upload relevant course materials such as syllabi, assignments, lecture slides, etc. which could be many different file formats (.pdf, .docx, .ppt). Other SQL database management systems would likely make this much more difficult as they are very strict about data types and sizes of elements stored, unlike MongoDB.

e. Network Protocol

Our project has four different components to keep track of: the webserver, the frontend website, the mobile app, and the server for the app information. Thus, we need multiple different channels of communication. We will consolidate the webserver and app server into one server written in Node.js hosted on Amazon Web Services. The server will act as a RESTful endpoint for communication with the frontend for the website and mobile app through the HTTP protocol. This will allow us to have a consistent protocol for data transfer across both of our interfaces. The use of a Node.js server will give us access to many publicly available packages such as code to make sure data is correctly encrypted or making sure that the data transfer between the database is reliable. The server will maintain a connection to a MongoDB database as described in the above "Persistent Data Storage" section of this report.

f. Global Control Flow

For the webapp, the system as a whole is event driven for deciding which functionality the user wants to use. However, specific functionalities are procedure driven. For example, when taking a quiz in the webapp, the user must select their options for the quiz questions and submit the form in order to continue. Similarly, for chat and forum functions, the user must input certain messages in order to an outcome to appear, which is a message being sent in this case. Since there are different functions to choose from, the initial interface will be event driven. Once a specific functionality is chosen, the app will be procedure driven until the procedure is complete, at which point it will return to being event driven.

Certain functions of the system are time dependent, specifically the quiz features. For the quizzes administered through the webapp, a quiz remains open for an amount of time as determined by the instructor. Once that time has elapsed, the student is locked from the page and the results are automatically submitted. Furthermore, there is also a timing feature in how long the quiz should be available for. For example, the instructor might specify a 24 hour window in which the student can take the quiz. Once that window has elapsed, the quiz can no longer be taken by the students and the option will not be available. Both of these are real-time constrained systems for a time frame specified by the instructor.

There are no concurrency specifications for this system.

g. Hardware Requirements

1. Screen Colored Display 1920 x 1080 (1080p resolution) 400 ppi minimum
2. Web server (AWS free tier)
    a. Amazon S3:
        i. 5 GB of Standard Storage
        ii. 20,000 Get Requests
        iii. 2,000 Put Requests
    b. Amazon EC3
        i. 750 hours of usage a month
3. Touch Screen enabled Smart Phone (latest Android or iOS) for Mobile app
4. Network Bandwidth of 256kbs
5. 2 GB ram
6. 50 MB hard drive space

10. Algorithms and Data Structures

a. Algorithms

The time limit of 150 seconds (2 mins 30 sec) after a question is asked by a student in class for it to be voted on is based on a study that states, “around 20% of a students time in class is spent on their phone" [1]. In an 80 minute lecture this would amount to 16 minutes. Assuming that the 16 minutes of phone time would be a random linear distribution throughout class, each student would check their phone an average of once per 5 minutes. Assuming a random distribution among all of the students of when they check their phones in 5 minute intervals, it can safely be assumed that in the 2 min 30 sec time interval, approx. ½ of the class may check their phone and see a new question that was posted. Since 50% of the class would potentially view a question, if ⅕ of them or a total 10% of the class thought a question had merit or was wondering the same thing and upvoted the question, it would pop up as a notification for the Professor.

This timer is set by default, only the 10% value will be calculated separately for each class based on number of students in lecture.

One possible feature for future work that we would implement given more time is an automated system that would find related documents in a particular class's resources. This would be used to create associations between certain topics and their related resources. For example, when a student would search the resources for documents related to "unit testing", the algorithm would output all documents that include "unit testing" or related terms. Our backend resource search will use fuzzy string matching algorithms to match search terms to related documents. Fuzzy string matching is computed using Levenshtein distances between two words, or the shortest number of insertions, deletions, or letter swaps it takes to get from one word to another. It allows us to find similar terms among multiple documents and group them so when a student searches for a document about "LaPlace Transform" they will be directed to documents containing the term and other related terms.

b. Data Structures

The in-class feedback/question will be a utilizing an AVL tree. The priority of each feedback/question will be 0 and will be updated whenever a user votes on the question. Feedback needs to be ordered so that the most voted one will be at the top. Updating the priority of a feedback/question needs to be fast for the page to update in real time for the users will take O(log n). Inserting, Deleting feedback/questions will take also O(log n) time.

For the purpose of online quizzes, arrays will be implemented to store the questions in our database. Quizzes will be formatted as an array of questions, which will contain an array of possible answers for each question. The first possible answer stored in the array will always be the correct answer, but the answers will be listed in random order when displayed to the students. This will combat sharing answers among students, simplify the quiz creation process a bit, and reduce space needed in the database to label the correct answer for each question. Quizzes will always be static, so arrays are ideal for this purpose. They allow for O(1) access and can easily be created and removed from the database.

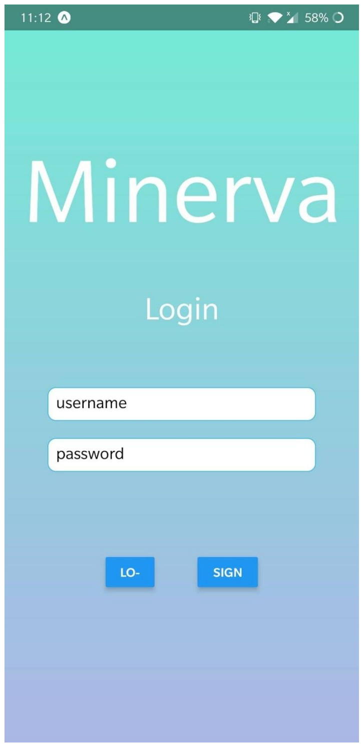
In the previous section, the algorithm that would match search terms to particular resources was described. In order to potentially implement that algorithm, there needs to be a good data infrastructure in place to ensure quick searching and consistency of the data. Any and all resources uploaded to the backend will be pre-filtered and inserted into a word frequency hashmap (key: "word", value: "resource"). We will generate a predetermined number of hashmaps, representing the frequency of words. For example, if a document called "BipolarJunctionTriodes.pdf" has the following frequency of words in descending order [it, the, a, BJT, Gaussian Curve, to, diode], the list would first be filtered of non meaningful words such as "it, the, and to" resulting in [BJT, Gaussian Curve, diode]. Then in the most-frequent-word-hashmap, (key: BJT, value: "BipolarJunctionTidode.pdf") would be entered, in the second-most-frequent-word-hashmap (key: Gaussian Curve, value: BipolarJunctionTriode.pdf) would be entered. This greatly decreases lookup time for the end user. We would use a map library to hash the keys then serialize the hashmap object and store it as user data, specifically the Map object provided by ES6. The next time a search is performed, the webapp will fetch the serialized object.

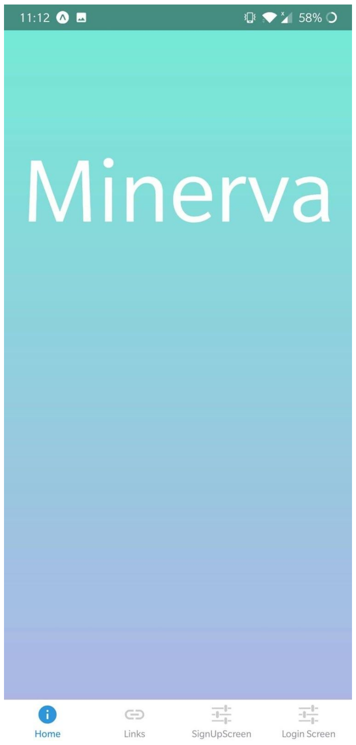
11. User Interface Design and Implementation

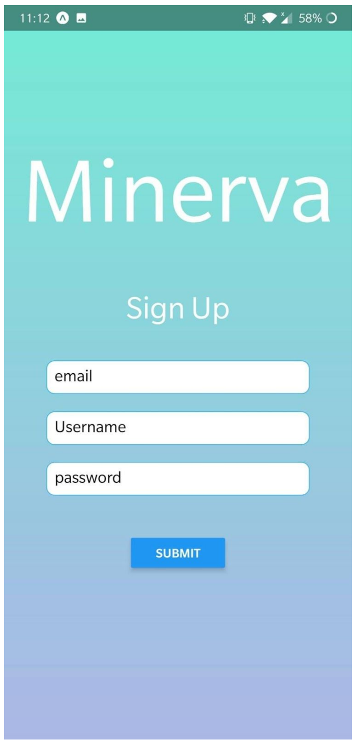
a. Mobile:

We took the designs from report 1 and implemented them using React Native so that it deploys in both iOS and Android. We used javascript to code some React Native components which define the styling and UI of the front end. For example, the input for net-id and passwords are handled with textinput component, where the values are stored as props. To navigate from screen to screen, we used the react native navigator, a built in library that helps navigate between screens (segues in iOS).

The user effort should be reduced in most cases as the react native navigator reduces the number of taps required to navigate between pages as there is a persistent navbar at the bottom of the screen.







b. Web App:

There are no significant changes to the UI mocked up in the initial report.

12. Design of Tests

For sign-in, the tests will be conducted by determining which portions of the sign-in are working. Since we are using React and React Native, it is easy to see which portions of the components are storing the login string and password and check if they are properly being authenticated.

In order to log in we will be testing many scenarios including:

- User creates a new account
- User tries to create an account with an already registered email
- User registers for a class (both when they aren't registered to any and if they already are registered to one)
- User tries signing in with incorrect credentials
- User tries signing in with insufficient credentials (missing required information)
- User signs in successfully (both professor case and student case)
- Unknown error
- Too many sign-in attempts

For the forum, there are many individual components that must be tested to ensure the application is working correctly.

In the below test cases, the user is assumed to be logged in as an existing user in the database

- When the user submits a post on a brand new thread, the database is populated with a new thread containing that post

- When the user attempts to view a specific thread, no other threads or posts from other threads will be shown to the user
- When the user submits a post on a specific thread, that post is associated with that thread in the database only
- When the user submits a post on a specific thread, and is not allowed to post on that thread, the post will not be associated anything in the database, and will be deleted
- When the user attempts to view a post on a specific thread, and is not allowed to view that thread, they will not be able to see that thread
- When the user is creating a post, the post will not be submitted to the database until the user explicitly submits the post

Teaching assistants and teachers should have all existing privileges of users as well as additional permissions. These test cases outline the additional actions and information TA's and teachers should have access to:

- Users with administrative powers (teachers and teaching assistants) try to delete a post, thread, or comment
- Authorized users try to undo the deletion of a post or comment
- Authorized users try to endorse an answer as the correct answer
- Authorized users try to access the edit history of a post or comment
- Authorized users try to "pin" a post (have said post appear first at the top of its category) for a set amount of time
- Authorized users try to "unpin" a post, regardless of which original authorized user pinned it
- Authorized users try to set tags to posts
- Authorized users try to edit their pinned announcements, and have a subsequent announcement pushed for it

For the quizzing features, there are various test cases that need to be passed to ensure that the application is working correctly. However, not all of these test cases are programmable in unit test cases. The unit test cases are similar to the actual usability test cases, so they will be defined jointly.

In the below test cases, the user is assumed to be an administrator who has the permissions to make a quiz. The test cases for making a quiz are defined as:

- User is an administrator
- When user submits a quiz with one question, the database is populated with a one question quiz
- When a user submits a question with one answer, one answer is associated with that question in the question table in the database.
- When the user submits a quiz with multiple questions, all questions are associated with the quiz in the quiz table and are not associated with other quizzes.
- When the user submits multiple answers for one question, the answers are only related to that particular question in the question table.
- When the user is trying to make quiz, the data is not submitted to the database until the user explicitly submits it or until the auto-save clock saves the quiz.
- When the user makes a second quiz, all fields are empty on the quiz creation page.

- When the user submits the information for a second quiz, all questions are confined to their particular quiz and all answers are confined to their particular question.
- When the user tries to edit a quiz, only the questions and answers for that particular quiz are displayed.

The following are test cases for *accessing* a quiz:

- User is a student
- When user opens a quiz, a call to the database will return the quiz data necessary to be rendered.
- When the user is taking a quiz, a timer counts down to indicate how much time a user has remaining.
- When a user submits a quiz without answer all the questions, a confirmation popup is displayed to notify the user that some answers were left blank.
- When a user submits a quiz with all the answers selected, the answers are automatically graded, and the results are submitted to the database (final score, answer selections, etc.).
- When a user fails to submit within the time-limit, the current state of the quiz will graded and submitted to the database.
- If a user begins a quiz within a timeframe of less than the given time-limit, the user will not have the full time limit to complete the quiz. All current answers at the time of the close date will be submitted promptly.
- If a user does not attempt the quiz within the given timeframe, the user will receive an automatic 'incomplete' or a grade of 0.
- A user cannot see a quiz until it is released.
- A user cannot alter their answer choice selections after it has been submitted.
- A user will receive feedback after the quiz deadline has passed.

For the announcement page, there will be a good mix of programmable unit tests and manual tests. Tests cases are as defined:

- User is anyone
- When user clicks on announcement page, a call to the database will be made for the current announcements completed and another call to mark the ones read for the user.
- Creators are Professors and TAs
- When the creator types in the textbox an option to submit and save the announcement will be available.
- When the announcement gets saved it gets populated in the announcement table in the database as in-progress.
- When the announcement gets submitted it gets populated in the announcement table in the database as completed.

For the resources folder and page, there are some considerations that need to be made to maintain consistent access for all users. Test cases are as follows:

- Users is anyone
- Only professors and TAs have permission to upload new materials
- Creators include professors and TAs
- When a creator attempts to upload a new document or file, the file will be stored in the database and be visible in the resources folder on the web application

- When a creator attempts to upload a file that already exists with the same name, the file will not be uploaded.
- When a creator attempts to upload a file greater than 25MB, the upload will fail (This is only for our current version, which has limited database storage space)
- When a student attempts to upload a file, the application will not allow them.
- The resource folder will only show files for the class specified.

13. History of Work, Current Status, and Future Work

Throughout the semester we were able to keep to the timeline outlined in the gantt charts used in reports 1 and 2. Due to changing conditions, some goals were completed ahead of time, and some were accomplished later. All the basic functionality of both sides of the web page, meaning the log in, registration, and basic individual features, have been completed. Further, the mobile frontend was integrated with a backend to allow it to communicate with the web page.

After the first demo, we had implemented good infrastructure to build upon for both the web page and mobile app. So, our focus shifted to adding more advanced features and refining the features that have already been added to improve quality of life. We also overhauled the UI for many of these features to make them more intuitive and visually appealing to use. These features and changes are highlighted below.

The following are completed features

- Web page login/registration
- Web page view/create announcements
- Web page take/create quizzes
- Web page assign/view grades
- Web page participate in forum
- Web page choosing active class
- Web page online office hours which include video calling and screen sharing by the teacher
- Web page teacher hub
    - Allows teacher to administer a quiz in class, view real-time student feedback, and view real-time student questions
- Mobile app Questions
- Mobile app Navigation Bar

- Mobile app Login

Throughout the semester, we were able to accomplish many of the goals we set out for ourselves. However, there are still many ways to expand this project and further work on features that currently exist that we did not get around to refining. In the web page, we could implement a way to search through resources based on keywords that show up in the documents. The possible implementation for this feature is described earlier in section 10 (Algorithm and Data Structures). Another feature that we could implement is a quiz analysis tool that generates statistics of student performances on certain topics. This would include statistics individualized for each student as well as statistics that pertain to the entire class. Furthermore, this could use many of the structures and algorithms that the resource-search feature would use, keeping features on the platform consistent. The last important feature that could be implemented for the web page is an administration tool. This tool would allow the administrator to assign students and teachers to certain classes as well as have a more robust way to interact with all the data on the platform.

In the mobile app, there were a few features in terms of usability that could be done in the future such as a logout function. Also, we were unable to complete the Upvote/Downvote functionality for the questions as since each of the question cards was dynamically generated, having the upvote and downvote buttons do different things was difficult, especially considering that the documentation for MongoDB Stitch, the REST API used to talk to the database, was lacking and there was no source online for using it with React Native. In the future, this feature would be completed and working.

Also, there is an issue with one of the npm modules where it used an older version of the braces dependency. This caused a lot of issues that could not be fixed by using "npm audit fix". The braces dependency had to be manually replaced with the newest version in the package-lock.json file in order to remove the error. This cannot be changed by us and has to be updated by the creator of the package. The only workaround is as stated above, manually changing package-lock.json.

Currently, the questions pop up and are organized by number of upvotes, but this could be remade to only show the questions asked in the past 5 minutes or even have upvotes be anonymous and have the most recent question show up on top (this could be done by what is preferred by professors and surveying them). Changing the order of the questions can easily be done with MongoDB Stitch as it allows the query to return based on certain parameters.

Currently there is a dissonance between the backend of the Mobile software and the webapp software; the mobile infrastructure uses MongoDB Stitch as a backend-as-a-service technology while the webapp uses a Node.js server hosted locally. This creates problems because Stitch natively handles authentication, of which is so abstracted away we currently log in anonymously and then authenticate with the "login" collection. This obviously isn't ideal and given more time we would host a server somewhere (e.g. horoku) and every service would connect to the server.

14. References

Anthony P Carnevale, Nicole Smith, and Jeff Strohl, "Recovery: Job Growth and Education Requirements Through 2020," *Georgetown Public Policy Institute*, last accessed 10 February 2019,https://cew.georgetown.edu/wp-content/uploads/2014/11/Recovery2020.ES_.Web_.pdf.

Monks, J. & Schmidt, R. (2010). The impact of class size and number of students on outcomes in higher education[Electronic version]. Retrieved 10 February 2019, from Cornell University, School of Industrial and Labor Relations site: http://digitalcommons.ilr.cornell.edu/workingpapers/114/

Undergraduate Enrollment," *National Center for Education Statistics*, las accessed 10 February 2019, https://nces.ed.gov/programs/coe/indicator_cha.asp

"Monstrous class sizes unavoidable at colleges," *NBC news,* last modified 24 November 2007, http://www.nbcnews.com/id/21951104/ns/us_news-education/t/monstrous-class-sizes-unavidable-colleges/

David Rosengrant, Doug Hearrington, Kerriann Alvarado, and Danielle Keeble, "Following Student Gaze Patterns in Physical Science Lectures," *University of Kennesaw*, last accessed 10 February 2019, https://www.usnews.com/pubfiles/PERC2011_Rosengrant.pdf.

"The Evolution of Technology in the Classroom." *Purdue University Online*, 1 Aug. 2017,

online.purdue.edu/ldt/learning-design-technology/resources/evolution-technology-classroom
.

“Barney McCoy Publishes New Digital Distractions Study.” *Barney McCoy Publishes New Digital Distractions Study | CoJMC | Nebraska*, Jan. 2016, journalism.unl.edu/news/barney-mccoy-publishes-new-digital-distractions-stud