HandRaise

Final Report

Team Name: sdmay22-40

Client: Dr. Md Maruf Ahamed

Team Members: Nick Oswald, Michael Kies, Brian Sayre,
Vance Kaw, Daniel King, Robert Walling, Jeremy Tracz

Team Email: sdmay22-40@iastate.edu

Team Website: sdmay22-40.sd.ece.iastate.edu

Revised: 4/23/2022

Table of Contents

1 Introduction

1.1 Team

Nick Oswald, Software Engineering
Michael Kies, Software Engineering
Brian Sayre, Software Engineering
Robert Walling, Software Engineering
Jeremy Tracz, Software Engineering
Daniel King, Computer Engineering
Vance Kaw, Computer Engineering

1.2 Required Skill Sets for your Project

Our project is dedicated to creating a web application to improve class participation and assist professors and TAs in facilitating class operations and communicating with students. Therefore, skill sets required for our project include web development, app development, frontend development, backend development, programming, creating concrete system design patterns, basic security, software testing, documentation, presentation skills, and more.

Due to the nature of the project, all team members will contribute towards web and app development, programming, testing, documentation, presentations, and more. Therefore, our team has divided primarily into frontend and backend project groups, but our project contributions are not necessarily restricted to these groups. Our frontend team includes Robert Walling, Nick Oswald, Jeremy Tracz, and Daniel King. Our backend team includes Brian Sayre, Vance Kaw, and Michael Kies.

1.3 Problem Statement

Our project looks to increase participation in large class settings. In large classrooms it can be hard for students to ask questions or stay engaged if they are sitting far away, and with so many other students in a lecture hall, students are not able to receive much, if any, personalized attention to assist their learning. Along with this, being called out in a large lecture hall can be intimidating. Also, it is difficult for professors to gauge how involved students are in lectures as well as get valuable feedback on what topics people are struggling with. The problem to be addressed is developing an app to allow professors to see statistics for class participation, assess TA and student performance and participation, and ask students questions to increase class participation. The application should allow students to feel like they are having one-to-one communication with the professor and TA. The introduction of more hybrid and online class settings, in addition to the traditional face-to-face classes, indicates the need for increased flexibility in learning environments and requires more flexible learning solutions, and this application seeks to provide a solution.

1.4 Intended Users and Uses

Our problem statement necessitates the app to support three different levels of users: students, TAs, and professors. Intended app uses for each level of user are displayed below.

Students:

- Sign up
- Ask questions in lecture with text, image, audio, and/or video to be answered by TAs and professors either anonymously or by name.
- Reply to conversations/discussions
- Participate in polls
- View archived discussions and polls

TAs:

- Sign up
- Reply to conversations/discussions
- View/grade responses to poll questions
- View archived discussions and polls
- See student contribution data

Professors:

- Sign up
- Open discussions
- Open polls
- Reply to conversations
- See student and TA contribution data
- View archived discussions and polls

2 Design

2.1 Design Context

As our design evolved, we focused our design to be ideal for a classroom setting at a university. We kept the priorities of students, TAs, and professors in mind when designing an app that would be accessible, easy to use, and fast. We also wanted to avoid complexity and design an app that would require minimal explanation to use and intuitive for all users, minimizing the time needed to learn how to use the app and maximize the time spent using the app features, including responding to students and increasing class participation.

2.1.1 Broader Context

Area	**Description**	**Examples**
Public health, safety, and welfare	The project will increase welfare by increasing the opportunity for students to ask questions in class, hopefully increasing their knowledge in the subject and helping their GPA.	A student is too nervous to ask questions in class with over one hundred people, so they ask anonymously through HandRaise.
Global, cultural, and social	Our project reflects the values of students by giving them an extra opportunity to participate in larger	For example, students with disabilities who are unable to

	classes where they would otherwise be more hesitant to reach out.	participate in class traditionally will be able to ask/answer questions through HandRaise.
Environmental	The cost of electricity for the servers that run HandRaise and the devices that access it will affect the environment based on the method the electricity was obtained.	If electricity was obtained by burning fossil fuels, there is an impact on the environment because of that.
Economic	Iowa State University can provide this application as a resource for students and faculty (i.e. this will be included with the cost of tuition). This will hopefully lessen (at least in some small part) the financial burden of higher education.	Instead of requiring students to purchase TopHat, classes can use our free application.

2.1.2 User Needs

Professors:

- Need a way to create different types of polls and quizzes that can/will be answered by students.
- Need to be able to view student questions and either resolve them in class or start a discussion thread.
- Need a way to gauge student participation and attention level in class.
- Need a way to see student participation metrics and grades on past quizzes.

Teaching Assistants:

- Need a way to respond to questions that students ask in class without interrupting the lecture.
- Need a way to view and grade polls and quizzes that the professor publishes and creates on the fly.

Students:

- Need a way to ask anonymous questions in a large lecture hall.
- Need a way to engage in a class even when a professor can't interact with them personally.
- Need a platform where they can interact with other students in a class in a controlled environment.

2.1.3 Prior Work/Solutions

The HandRaise application falls in the same category as a learning management app (LMS), so there are many previous solutions in the space. Some comparable applications include TopHat, PackBack, and Piazza, which all attempt to promote class communication. However, the HandRaise application differs from the elements of all these applications in several ways and builds upon the positive features of each of these applications to create a free, easy-to-use application for Iowa State University students.

Pros

- Extend the functionality of the current products
- Get direct feedback from students about what we could do to improve our product and add those desired features
- Students don't have to buy the third-party service

- The ability to respond to another student's question is like Piazza but can be done more informally (like during lecture).
- An application that can be easier to use and customized to the university

Cons

- There will be no Interactive Textbook like Top Hat (tophat.com)
- TopHat has a built-in Exam feature that will not be available to HandRaise (tophat.com)
- Labs will not be available in HandRaise (tophat.com)
- We will not have 24/7 support for HandRaise (tophat.com)
- There will be no automated participation scoring (packback.co)

2.1.4 Technical Complexity

Our project requirements include several features which will increase the technical complexity of the project. The primary cause for this complexity is the requirement that all messages, polls, and responses in the app need to be nearly instantaneous, and instructors need to be able to view student engagement statistics while still keeping the data secure. Some notable factors which increase technical complexity are listed below:

1. The project requires fast responses from the server after user input with hundreds of concurrent users.
2. To compete with currently used software the product must look professional and be intuitive to the students.
3. Storing user data means we have an extra responsibility of maintaining security standards for our product.
4. There is a challenge in building a product that is easily maintainable and extendable in the future.

2.2 Design Evolution and Revised Project Design

Over the course of our project, our client expectations remained mostly the same, so the design decisions set in CPRE 491 were mostly adhered to.

After considering the options, as well as the client's initial preference for a web-based application, we decided to move forward with developing the web application. If there was more time and additional resources, we expect that this app could be transitioned to a mobile application as well, although there are no expected difficulties with using this web app on mobile devices.

This would mean that in a classroom setting, students would be easily able to load the app on an internet browser, and the professor would be able to access questions and load polls to assess participation, while students would also be easily able to use the application on a laptop to type any longer responses that might be required for a poll or app response.

2.2.1 Design Evolution Since CPRE 491

There were some design changes made since CPRE 491 as the semester progressed. These changes involved UI differences and some evolution of features. For practicality purposes, it was not feasible to implement certain features, including implementing the app as a mobile application as well, but we did adhere to the client guidelines throughout the semester.

2.2.2 Backend Design Evolution since CPRE 491

- **Dedicated ISU Servers** - These servers will be dedicated to our service. The client-server architecture will be used so that many clients (user groups) can be connected to the server in-class session. Since the fall, we were aided by ETG to get a signed SSL certificate to enable HTTPS and WSS in our Spring Boot applications. This means all connections and requests to these servers are encrypted and secure for the user.
- **Spring Boot / Java** - This is the language/framework that we used for the backend side of HandRaise. This allows us an easy connection to MySQL and implementation of WebSockets for live class sessions. Since the fall we have made a few changes to the design. We opted to create two seperate Spring Boot applications, one for handling the MySQL connect and REST API endpoints and another for handling live class sessions using WebSockets. This allowed for more modularity when it came to development and allowed us to uphold the security measures we had in place for the REST API.
- **Spring Boot Security** - Along with adding HTTPS to our applications, we also made use of a Web Security package to use JSON Web Tokens, JWT, for user authorization. We created a service in our Spring Boot application that created unique, encrypted JWT's for users on login. This JWT is then required in the Authorization header of all HTTPS requests to authorize that the user making the request has permission to read or write to the application. For enhanced security, we also only allow one active token for each user and can set expiration dates for all tokens.

2.3 Requirements and Constraints

- Ability to run on different browsers and OS. (Functional requirement)
- Handle up to 2000 concurrent users. (Estimated using the top five largest lecture halls at Iowa State) (Functional requirement)
- Maintain security with multiple levels of access. (Realistic constraint)
- Code should be well documented and documented on a (minimum) weekly basis. (qualitative)
- Project must be completed within 1,500 person hours. (Resource constraint)
- Project should be testable by anyone working on it. (Realistic constraint)
- Web pages should be consistent across the site. Buttons, navigation aids, and other data should have the same feel and location as previous pages. (UI requirement)
- All code shall be either archived, deleted, or pushed to the dev branch each week. Once working, code will be pushed to master. (Resource constraint)
- Front end shall be intuitive and easy to navigate for every user. (Qualitative aesthetic requirement)

2.4 Engineering Standards and their Applicability

IEEE 1008-1987 (Standard for Software Unit Testing): This standard helped us in our testing phase of the project. This ensured that we were testing our app efficiently by covering as many cases as we could to prevent faults in our code. Reference: https://ieeexplore.ieee.org/document/27763

IEEE 23026-2006 (Standard for Software Engineering -- Website Engineering, Website Management, and Website Life Cycle): This falls perfectly into our project plan, as it has good and detailed constraints for the development, use, maintenance, and life cycle of our website.
Java coding standards are used for the backend through Spring Boot.
Flutter coding standards are used for the frontend development of the application.
GIT standards: Git will be our main hub for connecting code and managing tasks that need completed.

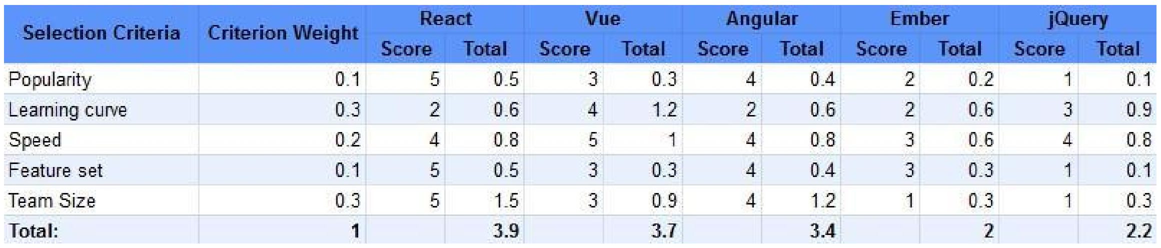
2.5 Technology Considerations

Our group considered that end users would be using the application on personal computers, laptops, and mobile devices, as well as our constraints and capabilities when deciding which frameworks to use for the frontend and backend development.

The lotus blossom for our decision-making for the frontend framework is displayed below:

Very popular (i.e. lots of online resources)	Open-source	Most Robust	Used before by team members	Made by Google	TypeScript	Open-source	TypeScript / JavaScript	Support libraries
Great for high traffic	**React**	Made by Facebook	Good for large applications	**Angular**	Steep learning curve	Light-weight	**Vue**	Smaller community
Dependable	JavaScript	Good for small teams						Component - based
			React	**Angular**	**Vue**			
				Frontend Framework				
			Ember	**Flutter**	**jQuery**			
Fastest development framework	Two-way data binding	Good for extensive projects	Fast development framework	Easier to learn than other options, but still includes powerful features	Flexible and fast for web development, including the ability to adapt to different screen sizes	It has Plugins	It comes with an MIT license and is Open Source	jQuery is flexible and fast for web develop-ment
Not fit for small development teams	**Ember**	Preliminary cost is high	Practical for small development teams	**Flutter**	Android studio compatibility	jQuery JavaScript file required	**jQuery**	Large library
Difficult learning curve	Easy add-ons	Very large community	Faster to develop and maintain a web application	Little ios support	Can be difficult to design the UI without experience	Function-ality maybe limited		

Initially, we chose to go with React as our frontend framework. To make this decision we considered popularity, the learning curve associated with the framework, the speed, the feature set, and the recommended team size. Pictured below is the weighted decision matrix to help quantify those criteria.



Selection Criteria	Criterion Weight	React		Vue		Angular		Ember		jQuery	
		Score	Total	Score	Total	Score	Total	Score	Total	Score	Total
Popularity	0.1	5	0.5	3	0.3	4	0.4	2	0.2	1	0.1
Learning curve	0.3	2	0.6	4	1.2	2	0.6	2	0.6	3	0.9
Speed	0.2	4	0.8	5	1	4	0.8	3	0.6	4	0.8
Feature set	0.1	5	0.5	3	0.3	4	0.4	3	0.3	1	0.1
Team Size	0.3	5	1.5	3	0.9	4	1.2	1	0.3	1	0.3
Total:	**1**		**3.9**		**3.7**		**3.4**		**2**		**2.2**

We decided to transition our frontend framework to Flutter after considering the difficulty with quickly learning and adopting React to develop a frontend application without previous experience. Therefore, we moved forward with Flutter as the semester continued.

For backend, using MySQL and Spring Boot allowed us to use the skills and technology learned in Com S 309 and other previous experiences.

2.6 Security Considerations and Countermeasures

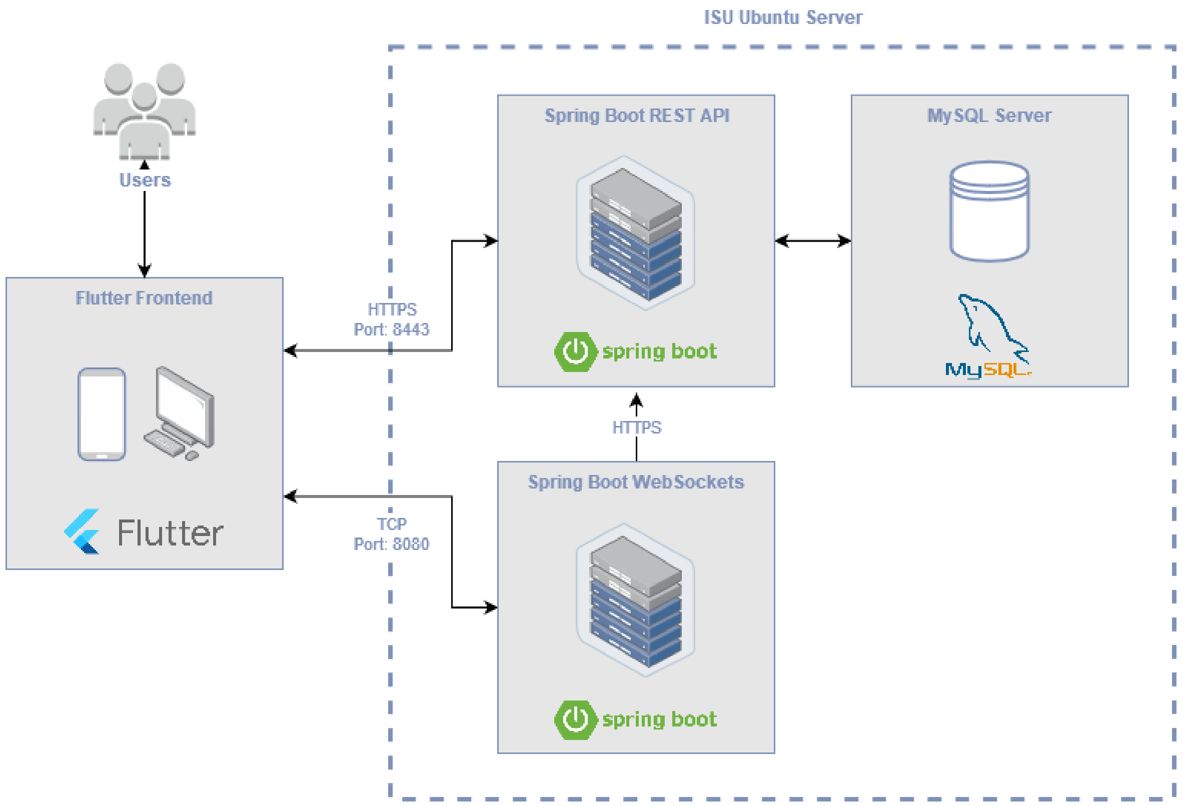
Our primary concerns were the following:

- Iowa State server's capacity to store all necessary data.
- Privacy of users' data and information.
- To work efficiently with minimal lag times.

During the design we addressed these concerns and questions with the following steps:

- Developing a backend that can handle a load of a couple of hundred users with sufficient response times.
  - Creating a sound design based on previous experience and researching the tools we will be using.
  - Load testing on our servers frequently during development to ensure we are meeting the requirements
- Making sure that the Iowa State server can handle the large number of archived discussions that will need to be saved for the project.
  - Researching ways to archive the data that will have a minimal data cost to save in the database.
  - Finding how much data the servers are able to handle so we have a guideline.
- Having a secure backend and frontend with minimal risk of data breaches and having an architecture that ensures privacy of our users' data (i.e a secure application).
  - Researching encryption and good security practices for online applications.
  - Having a continuous development environment so that any security concerns or bugs may be found and dealt with quickly and efficiently.
  - Setting up a good testing environment so bugs and issues may be routinely looked for and resolved.

3 Implementation



Final application architecture diagram

3.1 Back End Implementation

3.1.1 Spring Boot

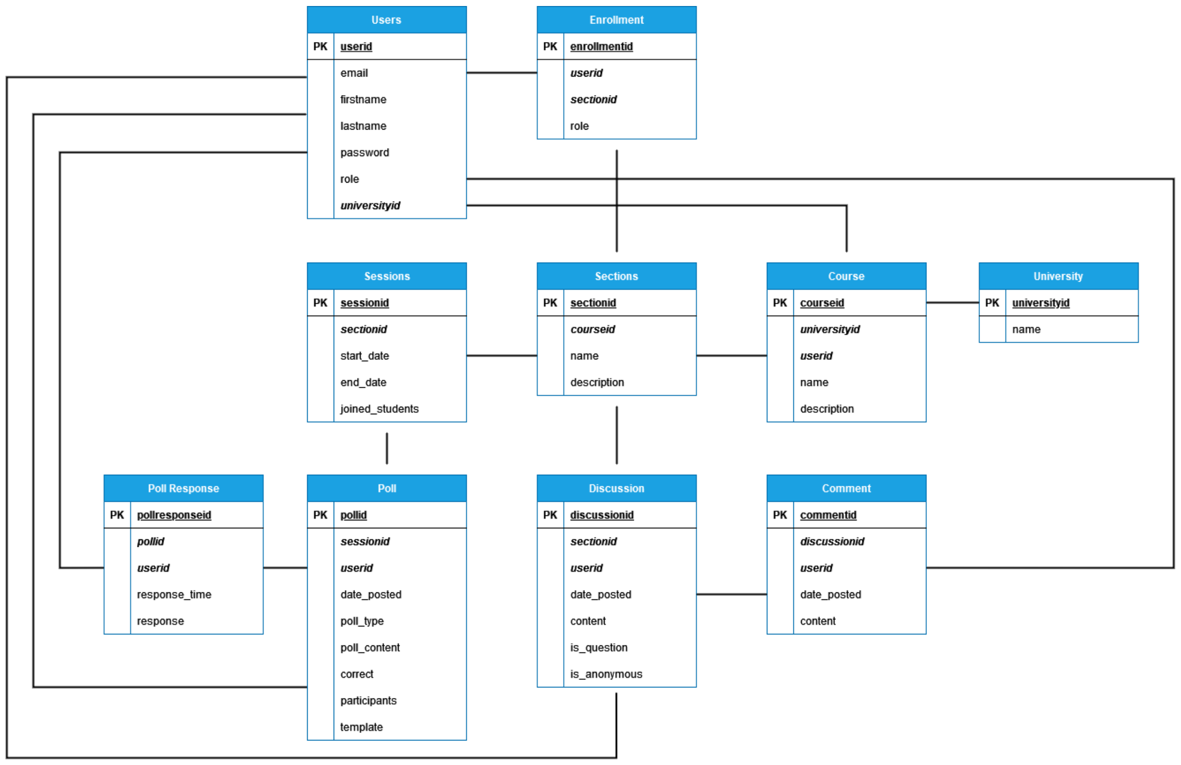
We chose to use Spring Boot to create a back end for our application. Our application required a basic RESTful API as well as a live session for polling in classes and live discussions. Initially, we thought we could do this all in one application but soon found that making two separate Spring Boot applications would be more beneficial for us. We found that separating the REST API from the WebSocket application allowed for easier development. Two separate applications meant one application could be developed and tested separately from the other. Along with that, we found that having the application run separate meant we were able to uphold the security measures we put into place for the REST API.

The first application we made was to create a REST API. This Spring Boot application created endpoints for the frontend to use to create, read, update, and delete from our MySQL server. We ended up with just over fifty individual endpoints over the ten String Boot controllers we made. Once the endpoints were implemented we had to think about user authorization and security of the REST API. For user authorization, we implemented the use of JSON Web Token to encrypt users' information into a token and gave each user that unique, expirable token to verify their identity with each request. This meant every request to our server must contain this token in the Authorization header. From there, the application would decrypt the token to verify the user is accessing permissible data. A new token is generated every time a user logs in and only the most recent token is valid for each user and expires every 10 days. This prevents malicious users from using an old token to gain access to unauthorized data. For an extra layer of security we also received a signed SSL certificate from ETG for the ISU server we were using. We were able to require HTTPS for every request using this certificate. Now every request to and from the server is encrypted for the users' security.

The second application we made was to add live polling and discussions to class sessions. To accomplish this, we created another Spring Boot application to run on the ISU server that uses WebSockets to create live sessions for class sections. Users of the app will connect to the WebSocket via the Flutter frontend. When a professor creates a session for a section it will allow students enrolled in the class to join the session. We verify all users and enrollments by sending HTTPS requests to our other backend application. Once the session has started the professor is able to open a poll, close a poll, and answer questions. Students are allowed to respond to the open polls, ask questions to the rest of the class (anonymous or not), upvote other questions, and reply to other questions. After a class period (session) has ended, the polls and responses from the session are then sent to the database to be stored for analysis later. The application utilizes the same SSL certificate as the first application to encrypt all incoming and outgoing messages to the server.

3.1.2 MySQL Server

As mentioned above, we used a MySQL server database for storing data for our application. We chose MySQL server due to familiarity within the team. easy integration with Spring Boot, and a relational database was necessary for an application of this size. We created a total of ten tables in our database. The data we are storing can be seen in the entity relationship diagram below.



Entity Relationship Diagram

3.1.3 ISU Servers

To host the two Spring Boot applications we used an Ubuntu server provided by Iowa State.We chose this option because it was free and had the necessary resources for development. On the server we had to open up ports 8080 for the live application and port 8443 for the REST API. We also had to install and set up a MySQL server on the server. Once the server was up and running, we were able to run the Spring Boot applications with no issues.

3.2 Front End

We chose to use Flutter for the frontend framework for our application. This decision was made by considering the limitations of the project and the decision was made in conjunction with the backend team to ensure that the frontend and backend connection would work as well as possible. Given these criteria, and the positive aspects of Flutter, we decided to choose Flutter as the frontend framework. These positive characteristics include a fast development framework, which is easier to learn than other options, but still with powerful features, and increased flexibility. These reasons also included android studio compatibility and that it is faster to develop and maintain a web application. One area in which the frontend team experienced difficulty was in the expected area that it is difficult to design the Flutter UI without previous knowledge and experience.

Required tasks as part of the design process included the UI mockup, which we completed in CPRE 491, and required tasks for implementation included deciding on necessary components, fetching data from the back end, displaying data correctly, sending data to the backend, and writing tests, including through a third party review process. Our implementation first focused on developing pages for login, chat, course selection, and more, including individual pages for students, professors, and TAs. After creating these template pages, implementation moved to developing the frontend and backend connection, verifying the chat functionality and the login pages, implementing the sign up pages, and then moving to features. These features included chat features, class addition and search features, polling features, and additional features, including developing user interfaces for professors, students, and TAs.

4 Testing

4.1 Testing approach

Our testing approach includes testing under the categories of unit testing, interface testing, integration testing, system testing, regression testing, and acceptance testing. Also, we initially set out to test our product in a classroom environment. Since development was not completed in time to run these full in class tests, we were unable to complete this testing. However, results from other testing can provide reasonable insight that the application would function well in a classroom environment. Actual test data would be needed to learn how many students, TAs, and professors can use the application in real-time without any performance reductions or problems. The testing approach was split into a frontend and backend focus and each of these is explained in the below section.

We worked closely with our client to ensure that both functional and non-functional requirements were being met. We also regularly performed regression testing after making significant changes to HandRaise to ensure it did not lose any desired functionality.

4.2 Frontend testing

We unit tested our Flutter frontend using the standard Flutter testing, which includes unit testing, widget testing, and integration testing. Every feature that we added was unit tested for each of its functionalities. For example:

- Page switching
- User submissions
- User login
- Error outputs

We used interface testing for the frontend to mock fake users to sign into that do not directly come from the database. This way, when there were any problems with the round-trip testing, the frontend was still able to develop features without losing too much time. This step was critical since each page needs to be working correctly and displaying the given information, so having a mock user sped up the process significantly.

We used manual testing to get responses that a computer was not able to provide. This included how the flow of the program felt and what changes needed to be made to the appearance of the application.

4.3 Backend testing

For the backend testing, the main components we tested in our Spring Boot application consisted of the model-view-controllers (MVC). Testing the MVC of our app ensures that every operation works as intended. The backend team has created a model for various categories such as

Course, Users, and University. These model classes dictated what parameters needed to be passed from the repository. We also created a repository (repo) class for every category to store data and perform operations such as finding the university name or user's email address. For instance, to test the Users category, we make use of integration testing. In here, we first test by adding a sample user with an email address, first and last name, password, and university id number and save it to the repository class for Users. Once added, we compare the results with what we expect through the use of assertEquals statements. In the same integration test, we also test the methods contained in UsersRepository. For example, the UsersRepository class contains several methods such as findByEmail and findByUniversityid. We test the functionality of these methods by calling one method and comparing them with what we expect. If we find that the result is what we expected, then we have successfully tested and verified the correctness of the method.

For the controller classes in our application, we use Mockito to mock objects. We achieve this by mocking the database and service methods and testing the results of the controller. We first test by mocking the findAll method in every repository class. Next, we mock the save() method in every repository class to determine whether the argument we passed was successfully added to the list. We also mock HTTP requests by using MockMvc. In our controller classes, we have HTTP requests (Get, Post, Put, and Delete) and we mock these requests to perform certain requests. This was done to determine whether these requests are done in the right way.

4.4 Roundtrip testing

We used Postman for simulating the connection between backend and frontend. This was critical to make sure that speedy development continues while both front and backend were being built initially. When testing the combination of multiple interfaces, error handling and Mockito were very important. It is easy to tell if the round-trip connection is working, as we had error handling that would catch anything that went wrong. For example, if there is a wrong email address, there would be an error returned. Or, if the frontend cannot get information from the database, there would be an error as well. Another example would be any CRUD features that we implement. The IDE debugger was used a lot during the testing phase so that we could see exactly where there are issues from the connection. The roundtrip testing step is critical as if there are problems getting information from the backend, the frontend cannot continue making features for the project.

4.5 Results

As mentioned previously, real-life testing in a university classroom environment was unable to be completed due to time and resource constraints. Results of the testing that took place were used to improve and verify the application and ensure that the application was meeting client requirements.

5 Professionalism and Context

This discussion is with respect to the paper titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416–424, 2012.

5.1 Areas of Responsibility

Our group analyzed our areas of responsibility using the IEEE code of ethics.

Area of responsibility	**Definition**	**NSPE Canon**	**IEEE differences**
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts.	The IEEE code of ethics includes a focus on work competence, namely code 6, "to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations." In addition, the IEEE code includes sections about disclosing information promptly and completing professional work.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.	IEEE code 5 states, "to be honest, and realistic in stating claims and estimates based on available data." In addition, IEEE code requires the rejection of all bribery, which falls under the umbrella of financial responsibility.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	IEEE code 5 states, "to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest, and realistic" and also to "credit properly the contributions of others." In addition, the IEEE code of ethics also includes a section on avoiding false or malicious actions, rumors, or any other statements contradictory to general engineering ethical standards that could be detrimental to yourself and others.
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	The IEEE code holds "paramount the safety, health, and welfare of the public" and strives to comply with ethical design. The IEEE includes this as its first code of ethics and places an emphasis on health, safety, and well-being.

Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.	The IEEE code seeks to “avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist.” In addition, the code seeks “to avoid unlawful conduct in professional activities.” These IEEE codes, 3 and 4, apply to the concepts of respecting property and ideas of clients and others.
Sustainability	Protect the environment and natural resources locally and globally.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	The IEEE code 1 “strives to comply with ethical design and sustainable development practices…and to promptly disclose factors that might endanger the environment.”
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	The IEEE code 2: “improve the understanding by individuals of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems.” IEEE code also places an emphasis on avoiding harm to others or engaging in harassment or other illegal and unethical behaviors of any kind.

The IEEE code of ethics differs from the NSPE version only slightly in each area, and most of the concepts are the same. The IEEE code of ethics also includes an important clause, reading, “to support colleagues and co-workers in following this code of ethics, to strive to ensure the code is upheld, and to not retaliate against individuals reporting a violation”, which relates to every aspect of the code of ethics.

Based on the analysis of these areas of responsibility, our group assessed how we did in the context of our project with adherence to each area of responsibility.

Area of responsibility	**Definition**	**Does it Apply? How are we doing in the context of our project?**
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	This certainly applies to our project and we are seeking to meet this responsibility in the context of our project by meeting and checking-in with our client and completing all class assignments on time, as well as moving ahead with frontend and backend app development.

Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	This applies because we must understand the need of educators and students to have access to this type of classroom management software at low costs while not sacrificing value. For this reason and others, we seek to offer this application free of cost to Iowa State students, faculty, teaching assistants, and more.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	We adhere to this area of responsibility by being understandable and upfront with our client and also being completely honest when completing each of the class assignments.
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	This does apply to our app development and we will always minimize risks to safety, health, and well-being of our users and stakeholders as our top priority.
Property Ownership	Respect property, ideas, and information of clients and others.	This strongly applies to our project because we must be careful not to rip off any of the ideas implemented by prior solutions in this space, including TopHat. We are planning to carefully implement new features which do not fully overlap with solutions already implemented. In addition, our app processes student questions and class information, which the app needs to keep confidential and only viewable to students, TAs, and professors.
Sustainability	Protect the environment and natural resources locally and globally.	These concepts do apply to our project and we always seek to leave the smallest environmental impact possible during app development. In addition, we seek to produce a low/no-cost product that will benefit students, TAs, and professors with a high level of accessibility and a positive impact on education.
Social Responsibility	Produce products and services that benefit society and communities.	

5.2 Project in context of related products and literature

As discussed previously, this project is implemented in a space with several previous solutions. Our group did not seek to compete with these solutions, but instead to produce a high-quality solution that would be free of charge for Iowa State students and easy to use and accessible for students, TAs, and professors. Although it is difficult to compete with the speed and features of these world class applications with our current limited resources, our group was able to create a high-quality product during the course of our project.

6 Closing Material

6.1 Discussion

Appendix I provides operation manuals for students, professors, and TAs using the HandRaise app. Appendix II provides discussion of other initial versions of the design including more details about the decision-making process and the design evolution. Appendix III provides other considerations.

6.2 Conclusion

Our group has worked diligently to complete all lightning talks, course assignments, and application development to a high-quality standard in CPRE 491, and with design ideas and application development in CPRE 492. Although we were unable to test the application in a large-scale classroom environment, we believe that the application is in a good place to move forward and be used in a university setting. We are hopeful that Iowa State students, TAs, and professors will find the app useful, allowing them to enhance the high-quality education offered at the university free of charge.

6.3 References

"The Active Learning Platform for Online, in-Person and Blended Courses." Top Hat, 14 Sept. 2021, https://tophat.com/.

"Angular vs React 2021: Which JS Framework Your Project Requires?" Insights on Latest Technologies - Simform Blog, 18 Nov. 2021, https://www.simform.com/blog/angular-vs-react/.

Author Bio Jeel Patel Designation: Founder - Monocubed Jeel Patel is the Founder of Monocubed and is the main curator & writer of the content found on this site. With ideals , et al. "10 Best Front End Frameworks for Web Development in 2021." Monocubed, 30 Nov. 2021, https://www.monocubed.com/best-front-end-frameworks/.

Hibbard, James. "The 5 Most Popular Front-End Frameworks Compared." SitePoint, SitePoint, 13 Apr. 2021, https://www.sitepoint.com/most-popular-frontend-frameworks-compared/.

Hibbard, James. "The 5 Most Popular Front-End Frameworks Compared." SitePoint, SitePoint, 13 Apr. 2021, https://www.sitepoint.com/most-popular-frontend-frameworks-compared/.

How Can We Help You? - Packback. https://help.packback.co/hc/en-us.

"JScripters.com." Jquery Disadvantages And Advantages | JScripters.com: Developing a Site with Javascript, https://jscripters.com/jquery-disadvantages-and-advantages/.

Thomas, Gavin. LORECENTRAL, 14 Nov. 2017, https://www.lorecentral.org/2017/11/advantages-disadvantages-jquery.html.

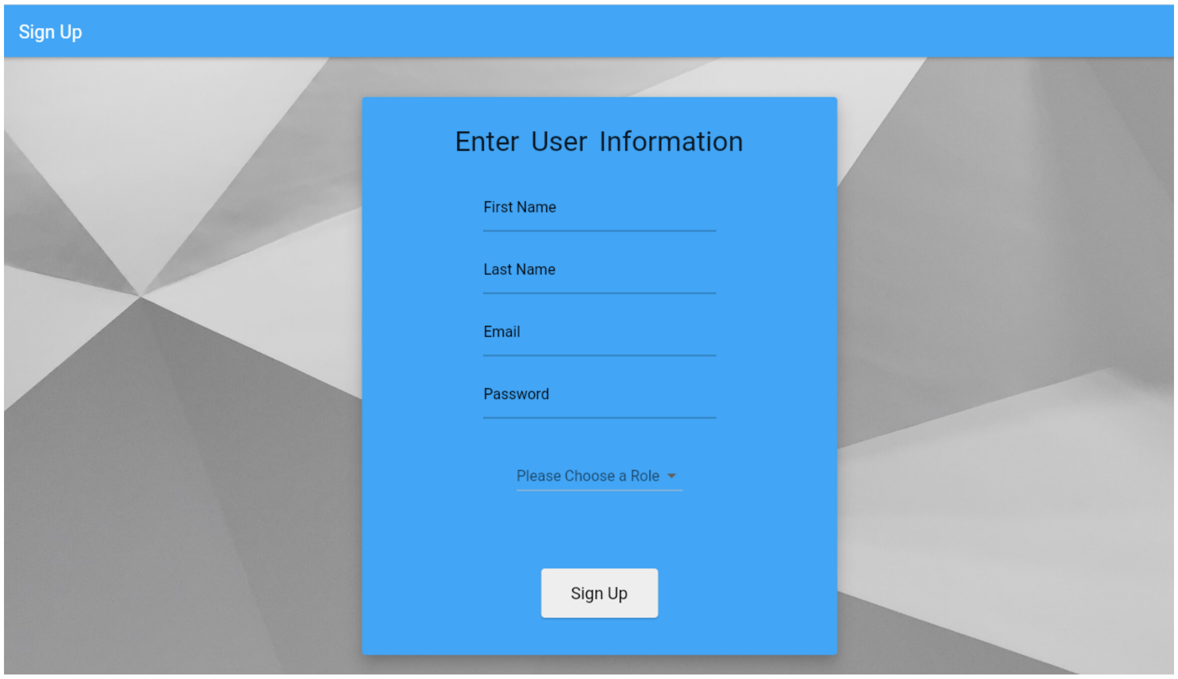
"Vue.js." Wikipedia, Wikimedia Foundation, 24 Nov. 2021, https://en.wikipedia.org/wiki/Vue.js.

7 Appendices

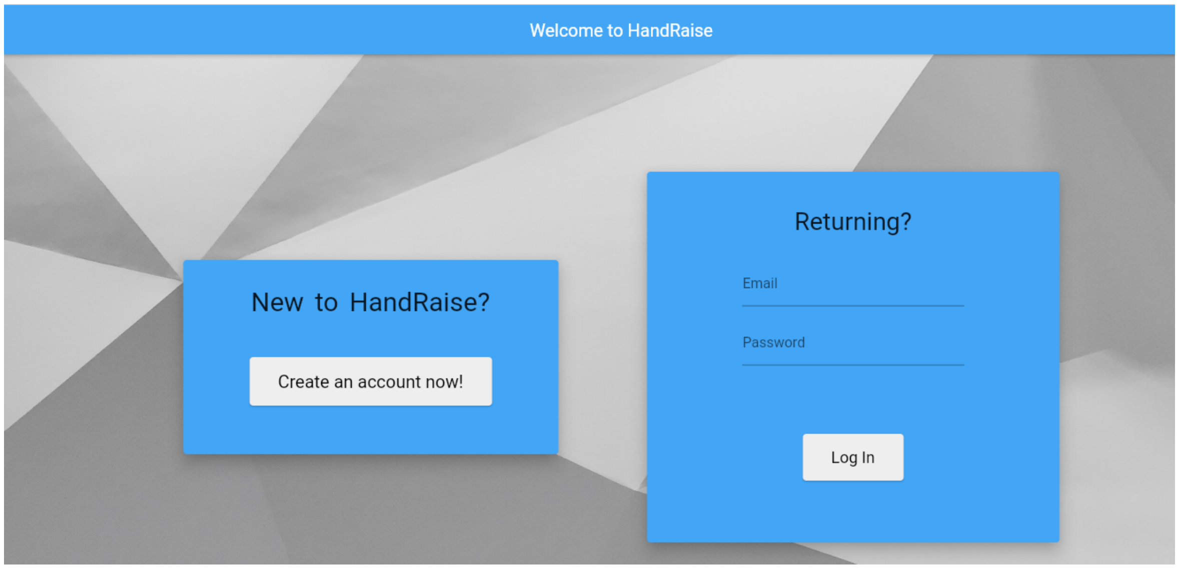
Appendix I – Operation Manual

Student, TA, and professor Instruction Manual

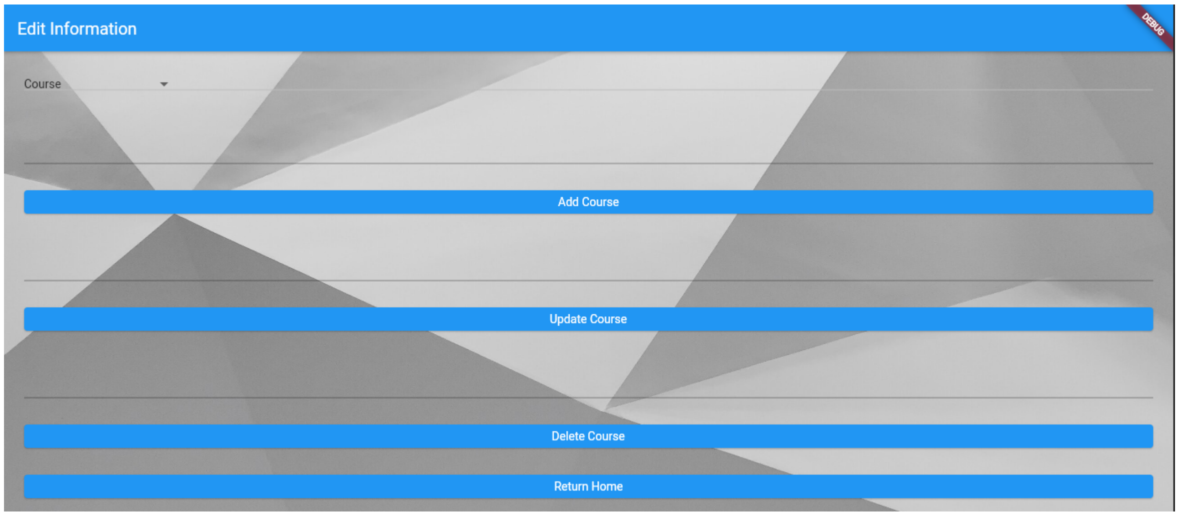
New users are able to create either a professor, TA, or student account on the introduction page. Students should select "Student" from the drop down menu. TAs should select "TA" from the drop down menu. Professors should select "Professor" from the drop down menu.



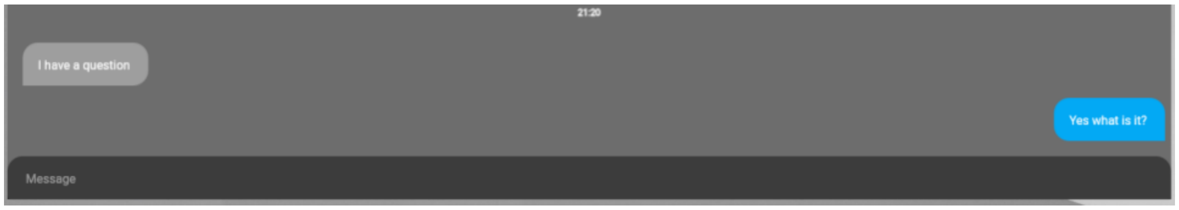
All users can log in at the introduction page displayed below.



Students are able to access existing courses, as well as access an interface to add, modify, delete, or otherwise make changes to their enrolled courses.



In a course environment, students are able to access chat features with the TAs, professors, and other students in the course.

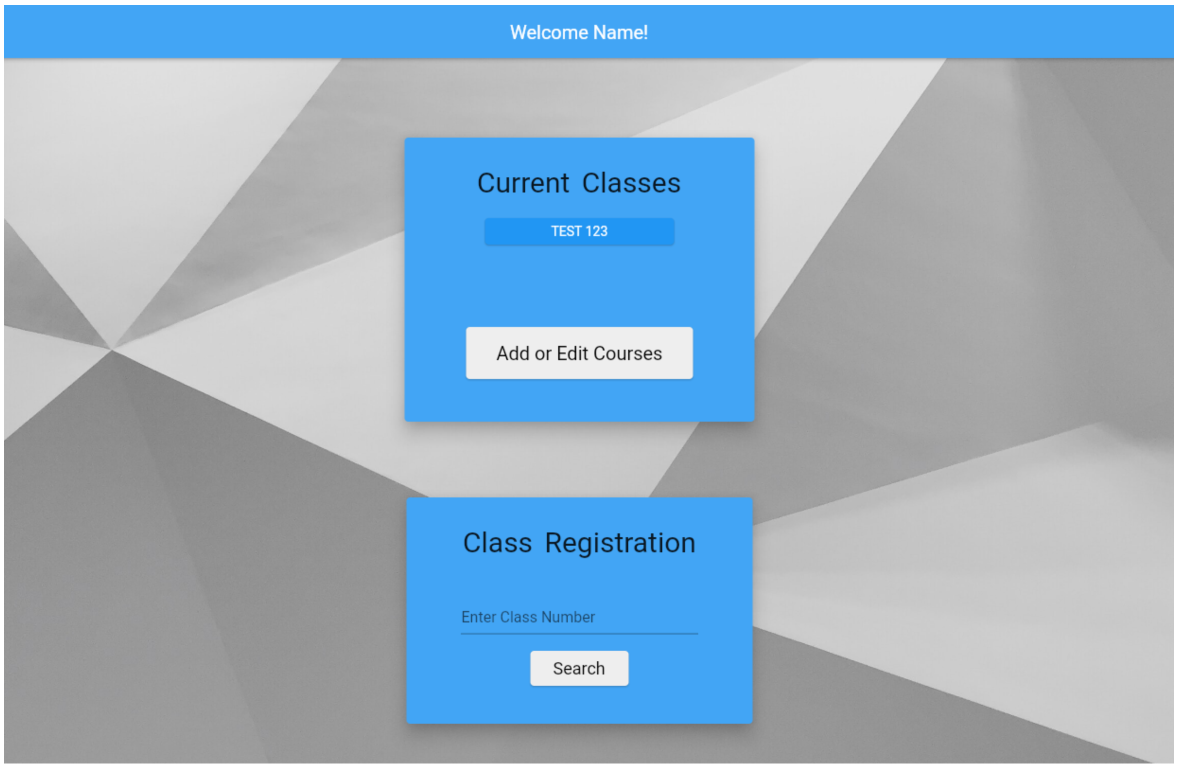

21:20

I have a question

Yes what is it?

Students can access poll features and participate.

Students, TAs, and professors can view current classes and register for new classes.



Appendix II – Alternative/Other Initial Versions of the Design

We considered a version of the application as a mobile application instead of a web application. Our primary reasons for keeping the application as a web application were client preference and increased flexibility by starting with a web application. This would allow users with mobile devices in a classroom setting to connect to the internet to use the application as well as allowing a professor and TA to use a laptop or personal computer to use the application either in an in-person class or during a virtual class lecture. Another factor in this decision was that it would be possible after developing the web application to transition the app to a mobile application.

In our decision matrix, it is depicted that we took several frontend and backend frameworks into consideration before making our decision. Initially, the frontend framework chosen was React, which provided positive features including it being open-source, dependable, good for high traffic, and the most robust option of the frontend frameworks. However, our group transitioned to Flutter after considering the limitations of creating a project using a framework which our group was previously unfamiliar with, and the need to learn it quickly while still having the sufficient features to meet the requirements of the client and advisor.

Appendix III – Other Considerations

It was difficult to create an application using a new framework, which our group was unfamiliar with. As is noted by the change in frontend frameworks during the project, our group was not sure about which framework to use, given that we wanted to balance a framework that had sufficient features but was also relatively simple to learn, which was necessary given the limited time to implement the project. Moving forward, our group would look to implement some additional features, complete the comprehensive in-person testing in a classroom environment as planned prior to the semester, and continue gaining user input to improve the application and hopefully see it be implemented in a real-life university environment eventually.