Mitchell Stewart, Matthew Hotchkiss,
Elayne Trimble, Sanjay Ramanathan,
Gaven Robertson

Project Vision and Scope

1 Background

Once finished, this project will provide a free, open-source, polling website for use in classrooms by students and professors.

1.1 Beneficiaries

This open source app will positively impact a variety of users, primarily educators and students. However, the project could also extend to benefit entire universities, companies, or other organizations that wish to utilize Open Educational Resources. In particular, this project is beneficial to all parties because it is free to use and aims to fill gaps left by other similar free resources. In other words, it fulfills the needs of students, instructors, and other stakeholders with high quality features that have been implemented in successful paid resources, but at no cost to users.

1.2 Sector

The sector of the economy this project is located in is education. This is a tool that is intended to help boost the efficiency and ease of online student responses to questions in a classroom setting - for both the student and the instructor. As a result, the project will be involved with schools and universities, therefore placing it in the educational sector of the economy. The current scope of this project is for a singular set of courses. However, because it is open-source, this could be scalable to a larger group of users as it has applications in many fields.

1.3 Problem Statement

Polling software is used by teachers at Oregon State University (OSU) to facilitate learning for students through the use of live questions that allow students to evaluate their comprehension of course material. It is also helpful for teachers, who can use it to get a sense of where their students are at in their understanding, as well as their areas of strength and weakness. The most notable use case for polling software at OSU is in the Physics 20x series.

The problem with current solutions for polling software is the cost. Paying for the ability to use the polling service is the only non-tuition cost for students taking the Physics 20x courses. In total, the cost is around $6,000 per year for OSU students. The primary polling software that is currently used is Learning Catalytics, which is fully featured and great aside from its cost. There exist other polling services in the market that are free to use, but these services lack the full range of features and flexibility offered by paid software such as Learning Catalytics. Thus, there is a need for an open-source polling service for classrooms that is free to use and still possesses all of the functionality found in paid services. This project will fill this need for OSU, and will serve as a stepping stone for future work which could further improve the polling software for OSU and universities around the world.

2 Vision

The Open Source Classroom Polling Software will be designed to solve a very specific problem, which is a lack of free, individually tailored, polling classroom tools for students. The Oregon State University's Intro Physics programs use Learning Catalytics; a polling software from Pearson. The cost of this software is not included in the student's tuition, so students have to pay for it out of pocket. Once the Open Source Classroom Polling Software is designed, implemented, and released, students will have a classroom tool that is equal in functionality but with no out of pocket expense. Additionally, the tool will be something that can easily scale and improve over time.

2.1 Benefits and Value

Our user base will initially be instructor Evan M. Thatcher and the courses he instructs (the physics 20x series at OSU). The open source aspect of this project will then allow a wider adoption of this tool, ranging to more instructors and professors within OSU and possibly beyond. This will happen because, assuming project success, Evan Thatcher will share the software with other instructors around and beyond OSU. One of the main benefits of this product is that it is open source, which differentiates it from similar products of its type which require some sort of fee or membership to use. Other free polling services do not have the full functionality this project will have, such as intelligent grouping of students and interfacing with Canvas. Regarding the adoption of future users, we are hoping to make this tool robust enough to allow for an easy adoption, using software such as Docker so that it is available anywhere.

As touched on, one of the beneficial aspects of this tool is that it is open source. This will allow schools and students to save money by not having to purchase a subscription to similar tools (such as Learning Catalytics, Top Hat, etc). While these tools are developed by professional corporations, we are hoping to develop our tool well enough to provide the same functionality with the same security measures in place. In total, this should be a similar tool with more accessibility for students and schools at a lower cost. This is valuable to the core users of the system, which are college students who can often struggle financially and for

whom the financial burdens of school are already prodigious. Just amongst students in the 20x physics series at OSU, this project will save an estimated $6,000 per year.

2.2 Requirements

At its most basic level, the system will do a few things. First, it will have a front end web page that is accessible to both students and instructors. The user interface will be simple enough that students who have no prior experience with polling services can use it with only a brief introduction as to how it works - as most physics series students are incoming freshmen. The primary pages present will be a login page, landing pages for teachers and students, live session pages, course pages, and gradebook viewing pages. Interfacing with the front end will be a back end that will do things such as general data storage, API calls to different webpages, gradebook management, session creation, and security handling. Through the back end, teachers will be able to create courses and questions, and students will be able to join them.

There will be a number of question types available for use in our polling service, including standard multiple choice, multiple select (where more than one answer may be correct), matching, free response, sketching, and ranking. To get class responses for polling questions, a live session will open up for students enrolled in a course to join. Students can answer these questions, and then instructors will have access to their responses for analysis and future course planning. Additionally, answers to questions will be stored so that students can be compiled into discussion groups based on their responses - with the goal being grouping of students with different answers together. Finally, there will be back end interfacing between this polling service and Canvas, so that classes can be registered through Canvas and gradebooks can be exported from the service to Canvas.

The software is open-source, and will thus contain a few requirements that can be more accurately identified as stretch goals. These are features that can be implemented in the future if time constraints prevent them from being introduced this year. Among such requirements include visualization of student progress or understanding - including hot-cold tracking of individual students, “I don’t understand” buttons, and confidence ratings. Other potential features include in-class social networking to discuss current or past problems, help request buttons, and links to relevant pre-lecture material.

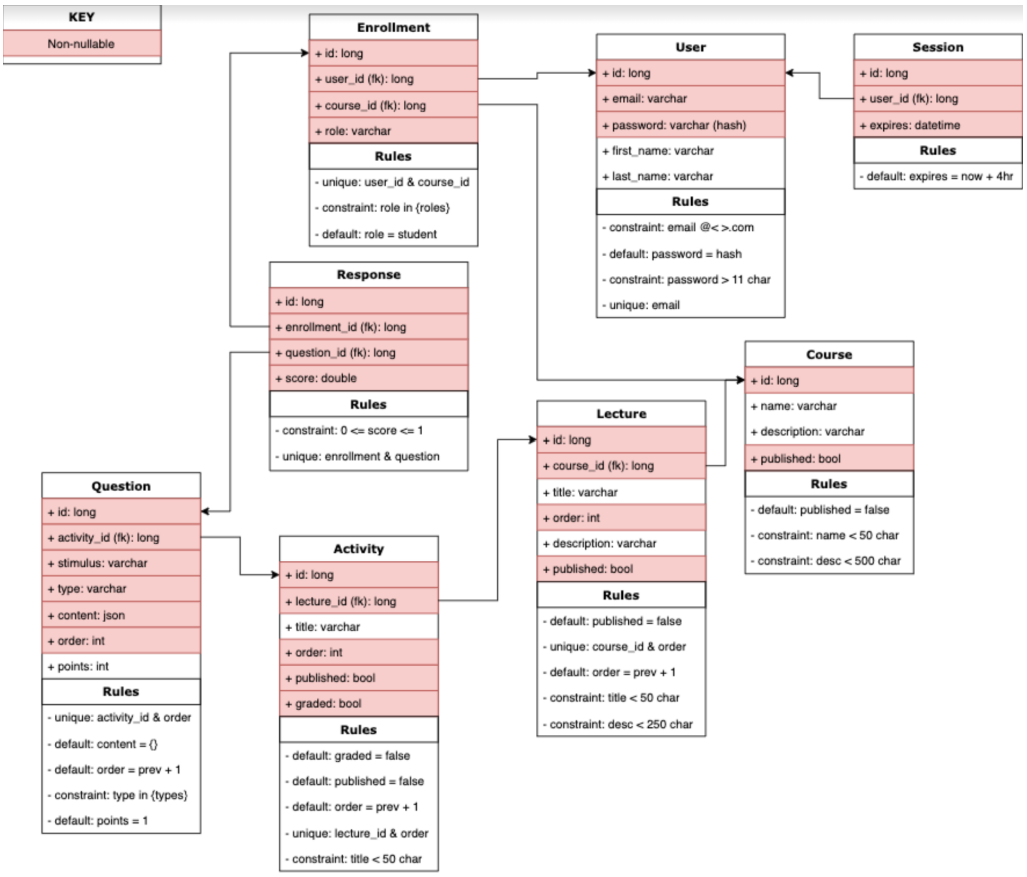
2.3 Logical and Data Requirements

Our database will be MySQL, and will include entities such as User, Enrollment, Session, Response, Lecture, Course, Lecture, Activity, Question, and so on. The names of users as well as their passwords will be used for login. During session creation, primary key IDs will be used. Similarly, these primary keys will be used when accessing the gradebook.

The database will be used frequently to do things such as record student answers to questions, save login credentials, and manage the gradebook.

Access to viewing the database will be granted only to administrators of the system. The database must be capable of handling hundreds to thousands of transactions per day at minimum, as this will be the frequency with which it is used by the physics 20x series.

Below is a rough draft of our database entity-relationship diagram:



3 Success Measures and Stakeholders

3.1 Success Measures

Measures of success for the project largely relate to behavioral dependencies, seeing that statistical measurements are not imperative to the core features of the project - outlined in the scope, Section 4.1.3. However, when it comes to performance metrics, the application should exhibit low failure rates and high fault tolerance rates, so as to limit cases of issues surfaced to the user without explanation or reasonable adaptability by the application. One great preventative measure to account for smooth fault tolerance and error handling is implementing tests with high code coverage, so as to catch issues before they're discovered by

users. Implementing a build integrated into GitHub would be an additional measure taken to ensure that as developers contribute to the project, errors are detected if test cases are failing.

As for behavioral and milestone success measures, the outcome is simply those described as barebones for the scope assessment of the project; A working application that instructors can utilize and universities can support must work. Furthermore, instructors and universities can expect that students are happy with the product, and they can achieve the classroom environment that they are envisioning.

That being said, success extends beyond that of project deliverables and behaviors. The developer stakeholders (Section 3.2.4), for example, have goals that extend beyond those of mere project deliverables. These include technical skill development, project management experience, practice operating in a team, communication & organization skills, and more. For a student, the expectation is that they will be able to participate in class by interacting with the software, in an environment that allows them to respond within the time constraints of the question and without connectivity, response time, or other issues. A culmination of the aforementioned quantitative and qualitative goals of the stakeholders and application behavior amounts to a successful project.

3.2 Stakeholders

The stakeholders in this project – as defined currently – are universities, instructors, students, and the development team.

3.2.1 Universities

"Universities" in this capacity refers to the administration and those that have a say over the licensing of tools such as Learning Catalytics and their adoption into curriculum. These stakeholders are looking to reduce costs for students, but they also have extra emphasis on functionality of the system. To justify its use, universities will have to be sure the system works at least close to as well as its paid counterparts. This group will have a wide reach and will be involved in many things with the university – this tool being a small part of their agenda. As such, the satisfaction of these stakeholders will be largely influenced by the opinion of students and teachers who have used the system and can validate its usefulness.

3.2.2 Instructors

Instructors are the people who teach the courses that will use the final system built during this project. This group includes the project proposer,

Evan Thatcher. These stakeholders teach courses where constant feedback surrounding student understanding is critical, and where students may benefit from knowledge check-ins. This includes areas such as math and physics, for example. This group expects first and foremost that the system has all the functionality of paid polling services, as there are already free services that exist which are not used because they are not fully featured. However, the open-source and free aspect of the project is also important to these stakeholders, who want to get the best deal for students. The opinion they have on the system will be a result of feedback they get from students on it, as well as its usability and usefulness for them in their teaching.

3.2.3 Students

The students are potentially the most important stakeholders, as they will be the ones who can most influence whether or not the service is adopted. In the end, students will be the primary group using the product, and if they do not like the service it will not be used. This group is generally very strapped for time in their day. They have school, work, and extracurriculars pulling at them in every direction. For many of them, college is the first time they are responsible for feeding themselves and handling that task financially. Moreover, students typically do not have large reserves of discretionary funds to spend on extra costs for class such as those incurred by the use of Learning Catalytics. As such, the most important aspects of the system for students are ease of use, basic functionality, and a cheap price. The students want to learn and want the tool to be functional so that they can use it easily, get a good check-in on their comprehension, and let their teacher know where they're at. However, they are also college students, thus the free aspect of the project is likely the most appealing part.

3.2.4 Developers

The development team consists of senior students at Oregon State University. The team's stake in the project is that they are responsible for its implementation, documentation, and for communication with the project proposer. Their input to the project is in their own design decisions and how they go about creating the software. The developers are in their final year of schooling and are looking to build their portfolio for their futures. The project presents something concrete to show prospective employers, as well as an opportunity to put together all the skills they've learned in their time at OSU. The team wants the project to be a success, and hopes to implement a fully functional polling service that can be used by universities all over the country.

4 Project Constraints and Risk

4.1 Prioritized Project Constraints

This project will take course over a 9-month period, with the expectation that the first 3-months will be dedicated to getting oriented, the middle 3 to 4-months will be for building on the project, and the last 3 to 2-months will be for finishing everything up. Time is a big limitation for this project. Besides just time, we are limited by resources, scope, and the risks we may face when working on this project. In this document, we will explore the constraints of this project and how we may navigate them in the next several months.

4.1.1 Time

The project team will generally follow the typical Capstone timeline. Planning for the project will be done in the first term, implementation will be done mostly in the second term, and the final term will be primarily finishing up implementation and delivering the product. For our team, these will not be overly rigid guidelines, as we would prefer to start on implementation whenever we have a plan we are happy with. Our weekly time allowance falls between unlimited time and severely limited time. Some of our members maintain active work schedules in addition to their schooling, which limit the time they can work on the project. Conversely, as this is the final year of school, some team members are ahead on credits enough that their course load is not overwhelming. In general, we expect each team member to have roughly 10 hours a week to which they can devote to the project, especially when implementation comes into full swing during the Winter.

The one area in which our team is slightly limited with regard to time is in availability to meet. We have experienced some struggles with finding a common time for everybody to meet to discuss the project, as during the beginning of this term we have been busy and have not had aligning schedules. That being said, we do not expect any serious problems related to this, and have still managed to implement regular meeting times. Furthermore, our communication is primarily done through a team Discord channel, making meetings less crucial.

4.1.2 Resources

The primary need for resources in our project comes from the hosting aspect of it. While there is not yet a concrete plan for how we are going to host this application, there are a few platforms for which we intend on

using. In terms of development purposes we plan on initially using localhost, however would like to scale up in the near future for other people at OSU to see and use. For this purpose we have considered using the Open Source Lab at OSU, however were turned down due to a current lack of a product. This is a resource that could become available in the future as we progress. Beyond this, we intend to use AWS EB (Elastic Beanstalk) to scale even further. We have been informed of a worker in Kelley who could get us acquainted with this platform, another potential resource. Lastly, we are considering purchasing a domain name for our site once it is published. While this is a decision that will be finalized down the road, we have considered either obtaining a domain name (which is very low-cost), or just leaving our site as an IP address if it's only to be used in small scale environments.

The other consideration of hosting comes to our actual usage. Being a student response tool, we expect students at a quantity of ~200 to be on this platform all at once. In other words, we will have a specified load at concentrated times of the day, and then presumably a very little load at other times. This is something else that may need to be taken into consideration when selecting our resource of hosting.

Lastly, we have some resources that are non-hosting related as well. This mainly consists of last year's capstone project on this same tool. We have an existing code base and a partially complete application to base ours on, as well as one of the team members who has generously dedicated some time to help our team with the project. As an aside, we were informed that professor Rob Hess was a good resource for the previous team for working through the technical details of the project, and could potentially be helpful for us as well. Altogether, these are the resources we have and plan to have going forward.

4.1.3 Scope

There are a number of features that are considered "bare-bones", and should be delivered within the time and resource constraints of the project. These include a front end interface for the polling service and a back end that does data storage, handles the gradebook, implements security, and interfaces with the front end. At the most basic level, the team should be able to implement a system where users can first login as either a teacher or a student. Teachers should then be able to create a question, lecture, or course, while students should be able to join these. Teacher's should then be able to create live sessions to answer question that the students respond to in real time. Finally, all data should be stored securely. These are the basic features that time should fairly easily permit

completion of. Another feature that should be doable is intelligent grouping of students based on their answers to questions.

Other desired features that are better considered “stretch goals” include bi-directional interfacing with Canvas, visualization of student progress of understanding, in-class social networking, and the implementation of things such as “ask-for-help” buttons and links to pre-lecture material. Of these, the main focus is interfacing with Canvas, but this will likely be a difficult process and will take some time. That being said, it will be the primary focus amongst stretch goals. Other stretch goals that may get done if time permits include the “ask-for-help” button and links to pre-lecture material. The stretch goals that are unlikely to get done are the in-class social network and discussion of the current problem within a question live session.

In totality – given the groups time and resource constraints – minimal deliverable functionality should definitely be met, while some of the stretch goals should be met albeit likely not every conceivable feature.

4.1.4 Risk Management

Risk ID	Description	Category	Probability	Impact	Performance indicator	Responsible party	Action Plan
R1	Group Member Falls behind on work	Internal	40% (We are all busy students, it could reasonably happen)	M	Not hearing from one team member often, not seeing changes in github/google drive from them	Group Member	Communicate with team members. See if they need some help with their assigned task.
R2	The server we choose for hosting may not be compatible with our existing framework (i.e, database server in mysql may not be	Technical	20%	H	Research on Hosts for web servers to host a specific database/web site is not turning any results that work without changing a significant portion of our existing site.	Development Group (Not the Host)	Change a portion of the site to fit with a host. Or use the OSU servers for the meantime which reliably host what we need.

	supported by hosting parties)						
R3	We take longer than expected to build off previous team's code	Technical	25%	M	Meeting project milestones, keeping pace with the project plan.	Development Group	Make sure we take time studying the previous team's code before we begin our own implementation.
R4	Team struggles with project organization	Internal	10%	H	Project planning resources being updated frequently, good communication with the project proposer.	Development Group	Keep the Trello updated and maintain weekly meetings with the group, the project proposer, and the TA.

6 Iteration Plan

The Open Source Classroom Polling Software will be developed and built over the course of 10 weeks during the winter term of 2023. Below is a chart which roughly provides a timeline for the tasks we expect to complete within that time frame. It is expected that each of these tasks will likely depend on each other, so a dependency list has been included as well to explain why some tasks must wait until later in the project.

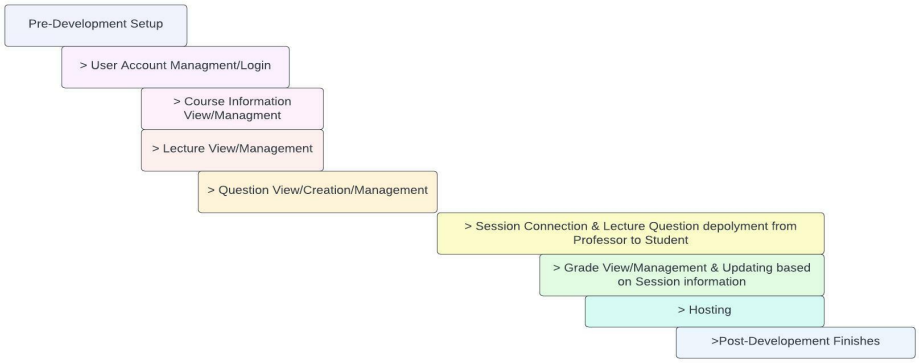
6.1 Gantt Chart

Task	Description	Pre-Winter Term	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Post-Winter Term
1	Set Up GitHub Project												
2	Set Up React App												
3	Set Up Node App, MySQL DB												
4	Create Docker deployment												
5	Establish FE/BE Communications												
6	Create Automated Testing Environments												
7	Create Home View												
8	Users API (includes Login)												
9	Email Notifications												
10	User/Login Views												
11	User Management View												
12	Connect to User/Login API												
13	Storing Session Information (FE Cache)												
14	Courses API												
15	Course Index View												
16	Connect Courses API												
17	Lectures API												
18	Course Page/Lecture Index View												
19	Connect Lectures API												
20	Questions API												
21	Lecture View												
22	Question View (Multiple Choice)												
23	Connect Questions API												
24	Grades API												
25	Grades View												
26	Connect Grades API												
27	Create AWS Instance												
28	Configure Environment												
29	Connect Domain Name												
30	MVP Beta Testing												
31	Integration												
32	Bug Fixes												

Task	Dependencies
1	None
2	1
3	1
4	2, 3
5	4
6	2, 3
7	2
8	3
9	8
10	7
11	7
12	8, 10, 11
13	12
14	3
15	2
16	14, 15
17	16
18	2
19	17, 18
20	3
21	19
22	2
23	20, 21, 22
24	23
25	2
26	24, 25
27	4, 6, 9, 13
28	27
29	27
30	Everything
31	Everything
32	Everything

6.2 Gantt Chart Explanation

The above figure depicts a chart of the expected timeframe to complete each project requirement. Each of these tasks can be categorized into nine different main parts of our product, which can simplify the diagram above for easier understanding. The figure below illustrates a simplified version of the requirements shown in the gantt chart.



The indentation of each encompassing requirement demonstrates what parts of each functionality will depend on each other. The majority of the end parts of the project require us to first have the skeleton of the product finished. Much of the backend surrounding users, courses, lectures, and

questions must first be implemented before we can use those parts to create sessions and have professors use lectures to give questions to students. Grades, which can be partially finished without the session connection part, must be connected to the results of those sessions eventually. This creates a need for these requirements to be mostly implemented sequentially rather than in bits and pieces at the same time.

The Gantt chart also clarifies how long it'll take to complete each requirement, as some will surely take longer than others. For example, the session creation and question response will likely take much longer than other parts of the project like the grade interface and course management. Nearing the end, some interfaces may also take less time compared to the beginning stages, as much of the database structures, frontend, and FE/BE connections have already been built.

As we progress through development, we may find that certain small project parts can be further broken down, and completed earlier. The timeline we've developed however focuses more on the big picture of when work will really start to happen for each functionality. It is expected that viewing this chart does not serve as a replacement for more detailed project planning down the road.