clickED

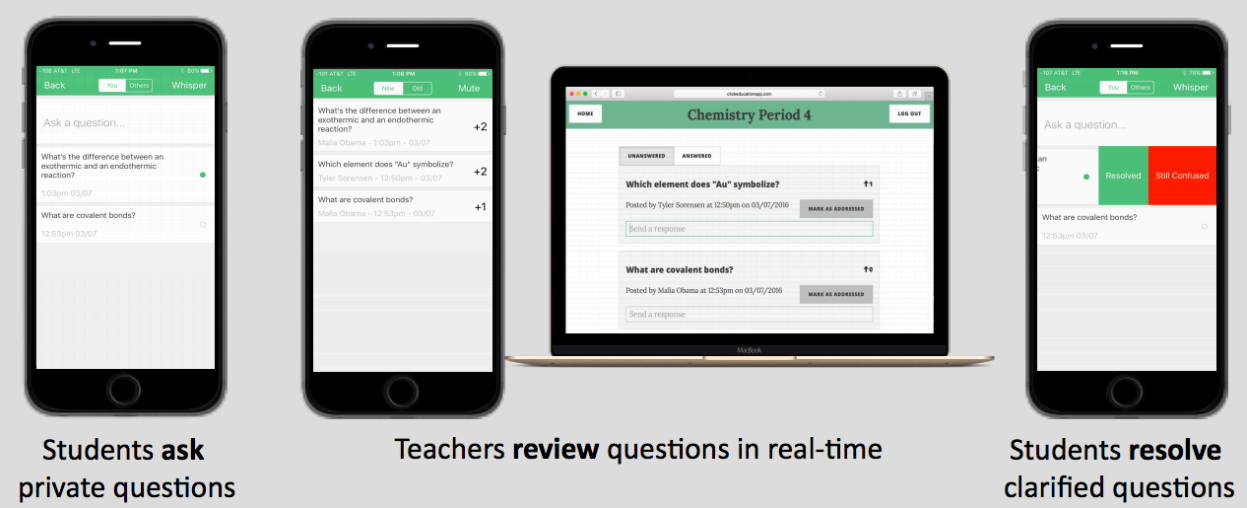
Never Leave Class Confused

Gordon Dean (Product Management), Jake McKinnon (Front End Development), Krister Johnson (iOS Development), Tyler Sorensen (Community Outreach), Arshin Jain (Back End Development)

Problem and Solution Overview

Problem: Students, especially middle school and high school students, don't want to publicly admit that they don't understand class material. Thus, teachers don't have a reliable way to determine their students' comprehension until it's too late (e.g. at the end-of-unit test).

Solution: clickED removes this communication barrier by creating direct communication channels between students and teachers, enabling teachers to revisit topics that students don't understand either during or after class. Students can send and resolve questions through the iOS app, and teachers can review and address questions through the iOS app or through the web app.



Tasks

Simple-Complexity Task: Students need to ask questions when they are confused.

We chose this task - which is arguably the most important task and our core insight - because it came up repeatedly in our need-finding talking to students and especially teachers. Students often mentioned that when they didn't know topics especially well, they would just shrug it off because they didn't feel like being "that kid" who holds up the class or doesn't understand something they should. On the other side, teachers knew that there were many students who were confused, but they couldn't identify them.

Our solution to this task gives students a clear means of resolving their confusion, with a very low barrier to entry (previously, the perceived social cost and extra effort required to ask teachers during work time or after class prevented many students from resolving confusion). We added multiple paths to complete this task: 1) writing out their question and 2) "+1'ing" a question you also have from another student. For a similar comparison, we looked at the impact that Piazza had on student participation in college courses. Although Piazza doesn't improve in-class participation, the number of questions posted on the app vastly outweighs the number of questions asked during class. We hope to see a similar increase in student participation.

Moderate-Complexity Task: Teachers need to effectively judge whether or not students understand the current lesson.

This task was chosen largely as corollary to the aforementioned task, but from the teacher's perspective. Nearly every teacher said they struggled to judge whether students understood material in real-time. They acknowledged that their top students would immediately answer every question and lure them into a false sense of their class's understanding.

Our solution for this task fell out of what teachers said they already try to do: use coarse-grained signals. By making it much easier for students to give feedback, we hope to give teachers a stronger signal with which to "read" the class and judge student understanding, by essentially giving them a newsfeed of students' thoughts about class material.

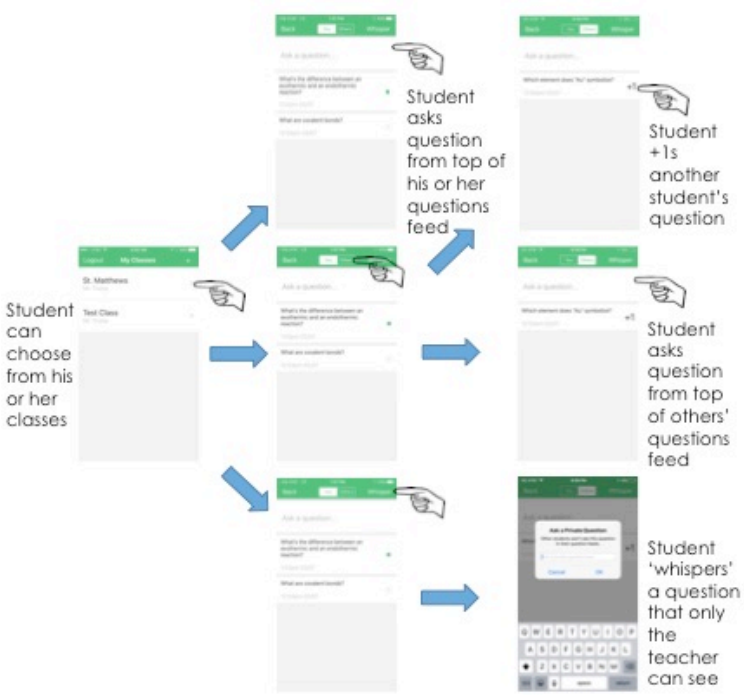
High-Complexity Task: Students need to resolve addressed questions.

If and only if a student feels that his or her question has been answered, it should no longer stay in the teachers' feed of new questions, so that the teacher can

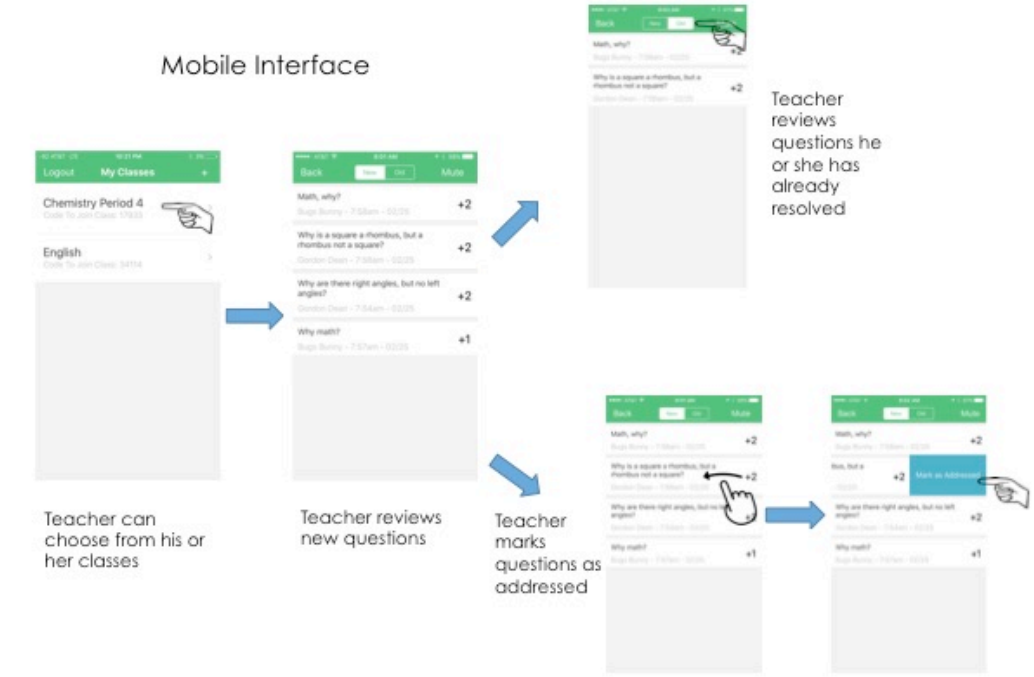
focus on questions that still need answering. Students always have the ultimate power to mark a question as resolved. But during our lab usability study, we found that the teacher wanted the ability to remove questions from her question feed. Thus, teachers now have the ability to mark a question as answered in order to de-clutter their new questions feed. The student then has the ability to either confirm that his or her question has been answered, or tell the teacher that he or she is still confused, which puts the question back into the new question feed for the teacher. Because students have the final say, teachers will get an accurate picture of what her students actually understand, what still needs to be clarified, and what needs to be addressed again.

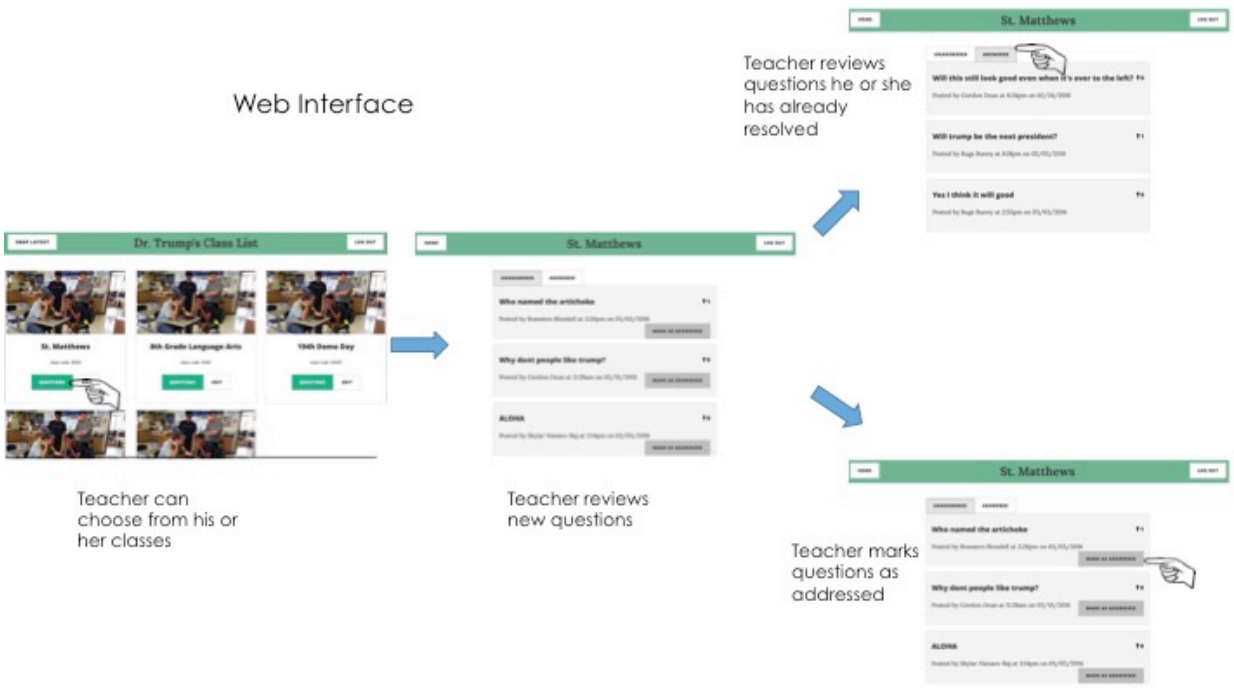
Tasks Flows

Simple Task Storyboard: Students need to ask questions when they are confused.

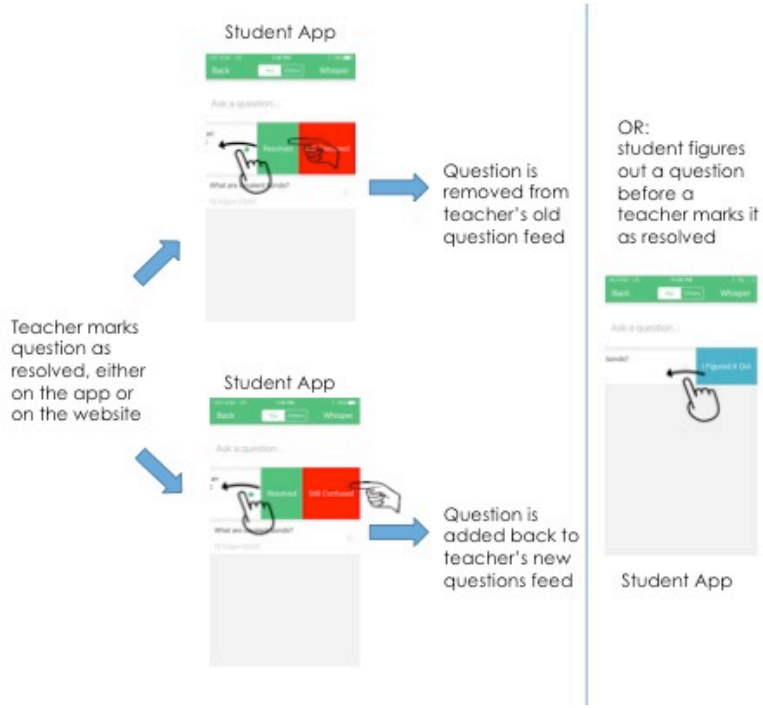


Moderate-Complexity Task Storyboard: Teachers need to effectively judge whether or not students are understanding the current lesson while in class.



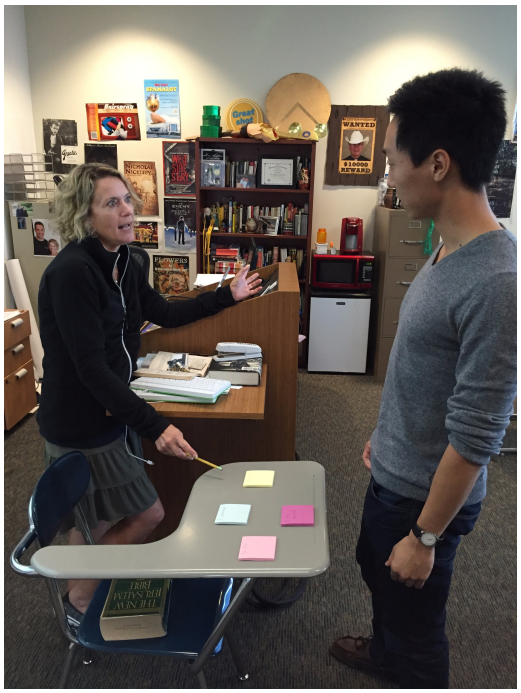


High-Complexity Task Storyboard: Teachers need to address unresolved questions after class or in the next lesson.



Design Evolution

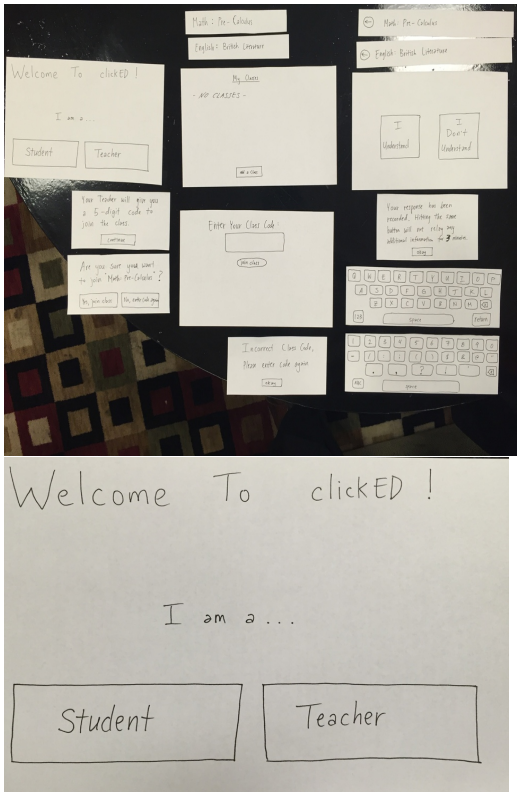
Mobile Prototype Evolution



The Experience Prototype modeled a system that gave teachers real time feedback from the class about their understanding, placing post-it notes for different levels of understanding.

Key Takeaways:

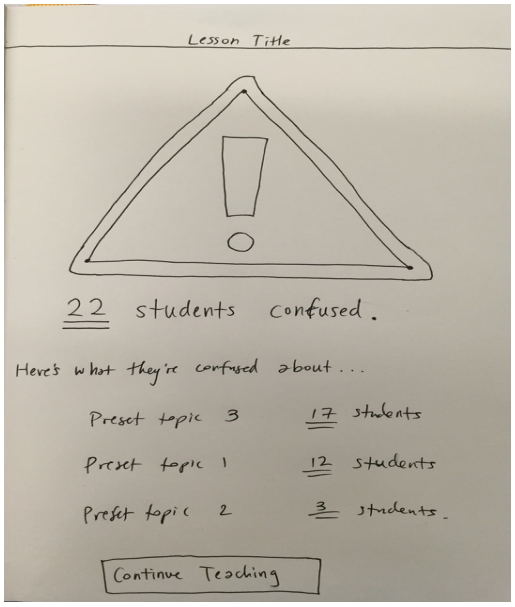
- Frequent pinging and interruption is distracting to teachers.
    - Solution: Use a threshold and only notify the teacher if enough students are confused.
- Negative feedback can often be detrimental to a teacher's confidence
    - Solution: Offer an "I Understand" button so students can undo previous confusion.

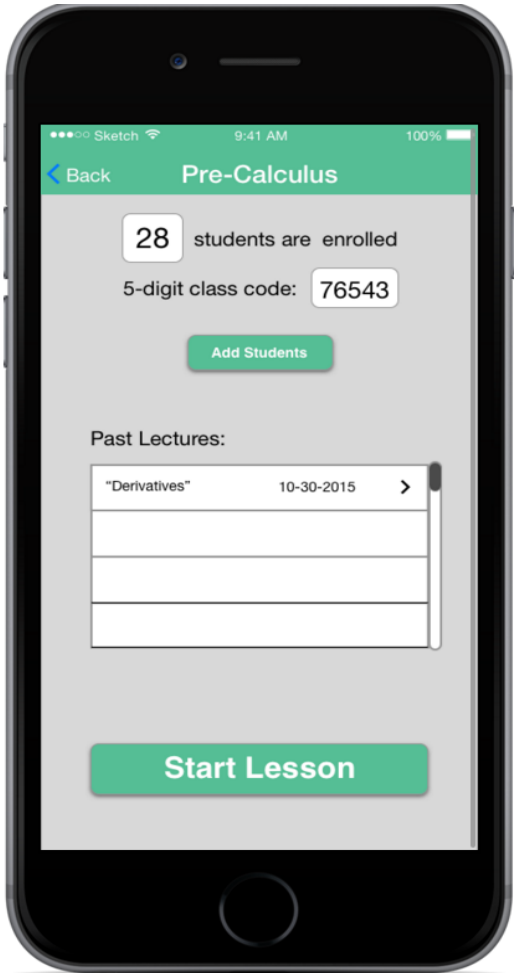


Our first paper prototype was done horizontally because we intended to develop the app for the iPads that are used at Bellarmine High School.

Key Takeaways:

- Designing for iPad was difficult as only 1 member in our group had an iPad.
    - Solution: Switch to iPhone, as more users are likely to have iPhones than iPads
- Data presented to teachers had to be detailed and ideally unanonymous.
    - Solution: We made it so the in-class feedback was anonymous until the end of class, when the teacher could see which students had

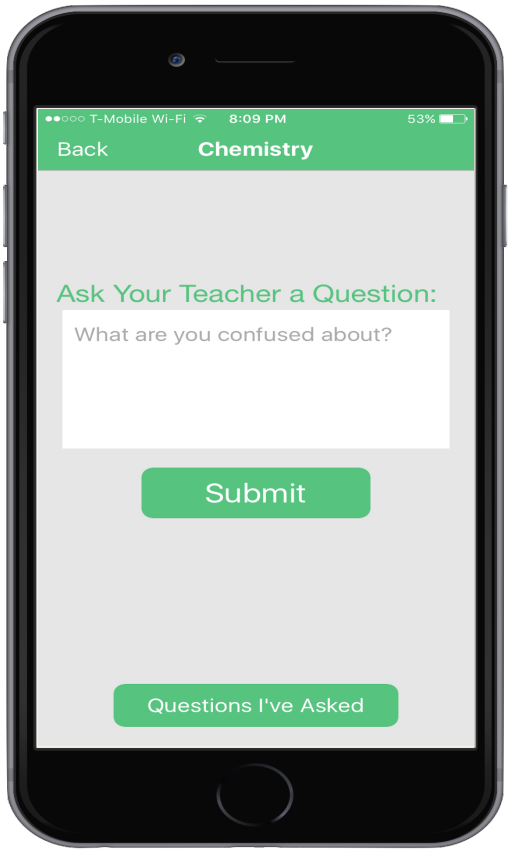
	responded and follow up. ● The main screen was too simplified, to the point where it was sometimes unclear how to proceed ○ Solution: Pop ups and tutorials to guide people through the app.
	Our revised paper prototype included elements from the first, like having a "threshold exceeded" warning and giving students 2 buttons to signal understanding. **Key Takeaways:** ● Teachers felt that just knowing the number of those who hit "I don't understand" wasn't very helpful. Students also wanted to specify their misunderstandings/confusions. ○ Solution: We decided to give teachers the option to input "Preset topics" so that students could specify what they were confused about.



Our medium-fi prototype was done on Marvel and made a lot of the visual elements more important. We chose to use a simple color scheme and keep all the screens relatively hassle free.

Key Takeaways:

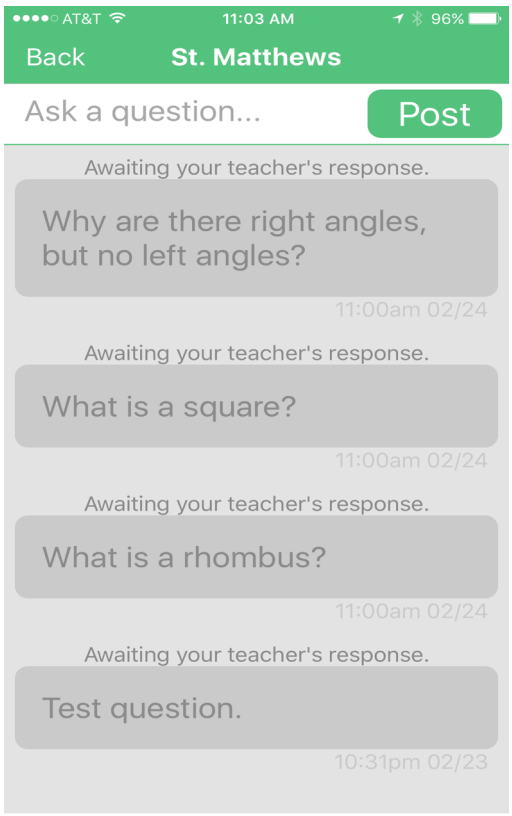
- We used a lot of popups with the intention of guiding users through the app and reducing the number of screens to navigate, but sometimes the popups were excessive.
    - Solution: We modified a few of our popups into new screens.
- Teachers didn't have the time to add topics in advance for students to select
    - Solution: We enabled students to ask their questions directly to their teacher
- The "Threshold Exceeded" alert really threw teachers off and made them not want to use the app
    - Solution: We created a feed of student questions that teachers could address during natural pauses in their lesson



Our final prototype from cs147. This included "Wizard of Oz" data. Devices displayed their own questions, but didn't communicate with others (e.g. student to teacher) at this point.

Key Takeaways from Lab Test:

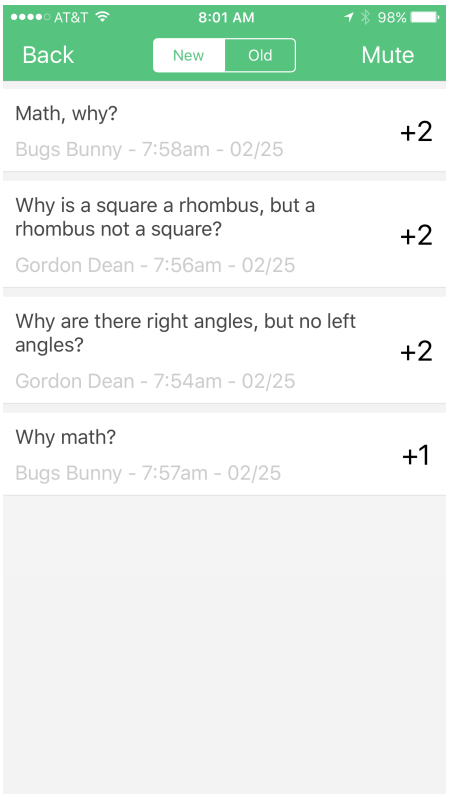
- Submission screen being on separate page from the question feed was confusing to some users – they had difficulty jumping back and forth.
    - Solution: Consolidate the list of asked questions onto the "Ask a question" page
- Students wanted to delete/resolve questions, but couldn't figure out how.
    - Solution: Add swipe to resolve – which students naturally wanted to do.
- By the test, we had a basic backend implemented, but it was horribly slow.
    - Solution: add caching and multi-threading to speed up the question load time.



For this prototype, we greatly improved the backend performance and updated the student to teacher interactions, indicating status changes with changing colors.

Key Takeaways from Field Test:

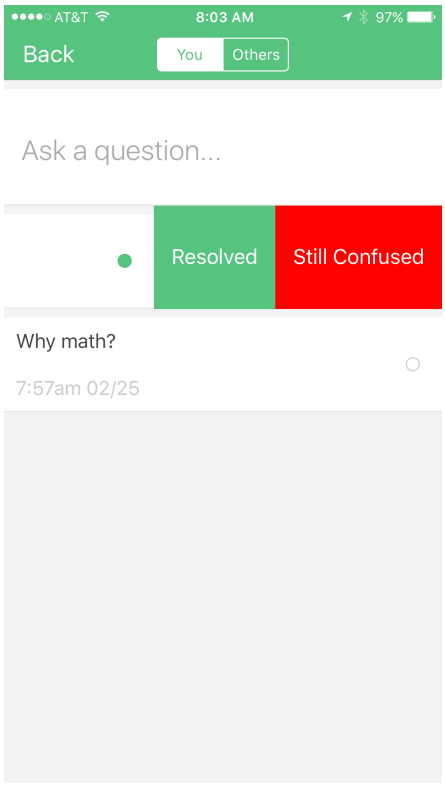
- Users wanted to see each others' questions – and teachers got too many redundant questions
    - Solution: Make an anonymous public feed of student questions.
    - Solution: Add upvote functionality to reduce the time to ask a question for students and to help teachers prioritize questions based on demand.
- Design still not intuitive– text bubbles without messaging, swiping round assets
    - Solution: Switch to a card layout and make UI more professional, adhering to iOS style guidelines.



When we added the public feed, we were concerned that students would use it to "troll" each other. When we initially tested the app with just one user account, students could see each other's questions, and they used the app to inappropriately communicate with each other.

Key Takeaways from Field Test:

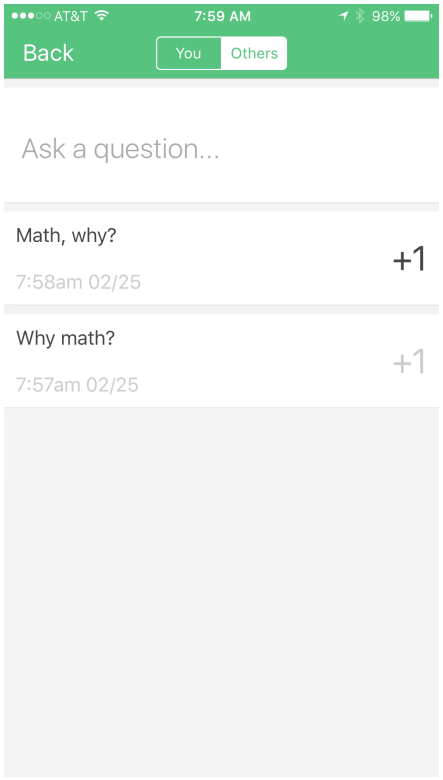
- Teacher wanted ability to shut down the public feed if it was being misused
    - Solution: We added the "mute" function, which enables teachers to close off students' access to the public feed. This worked perfectly when we tested the app as students improved their behavior in response to the teacher muting the thread.



In our field test, we tested with two very different classrooms, and the less tech-literate one didn't notice our old teacher response indicator.

Key Takeaways from Field Test:

- Students were not resolving questions because they did not understand the teacher response indicator
    - Solution: We added the green indicator dot, using email inbox notifications as an inspiration, to alert students that their teacher has addressed their question. Students immediately understood what the indicator meant, and they immediately responded.

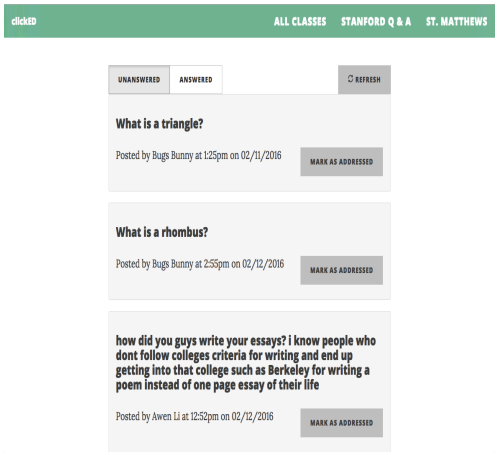


This was our final product, which we actually got to test when we were filming our Hi-Fi video. Users were really enthusiastic that we had incorporated their feedback, and had no areas of improvement. The teacher even said, "Man I wish I had had this in my last class."

Key Takeaways from Field Test:

- Before the demo day, we addressed one last piece of feedback in which students requested a way to send totally private questions.
    - Solution: Add whisper functionality so that students can send private questions, as well as the normal anonymous public questions feed. We ended up testing this functionality a bit on the demo day.

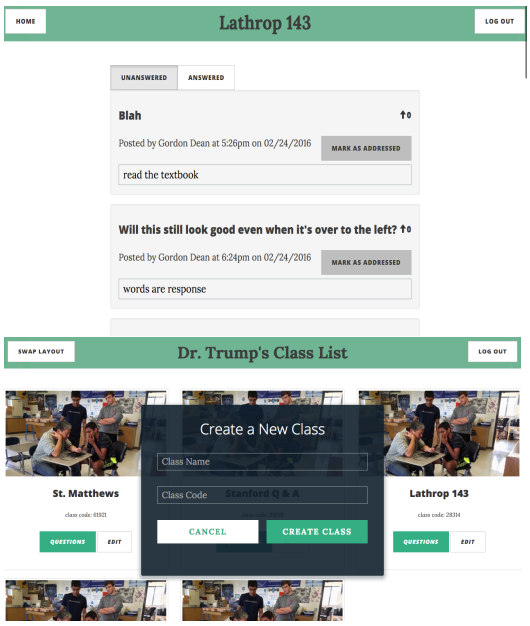
Web Prototype Evolution



Our initial website was created for Hi-Fi 2 and had the feed of questions updating periodically by pulling from parse every few seconds and a force-refresh button, with classes displayed across the top to switch between.

Key Takeaways from Field Test:

- The teacher was confused about how to select a class until it was pointed out. They also didn't realize which class was selected once they selected one.
- The update button was confusing – teachers kept clicking it thinking it was necessary to refresh the feed.



For Hi-Fi #3, we tried to address those previous usability concerns as well as add much of the missing functionality. We added the ability to log in/out or create a user, a home page where a teacher could create or edit classes, and changed the feed.

Solutions to Previous Field Test Problems:

- Create a home page to select classes, and change the top bar to display the current class. We found classes don't need to be changed frequently, so an extra "home" click doesn't hurt the teacher's workflow.
- Remove the refresh button and rely on automatic refresh every few seconds. We found this delay doesn't matter, as teachers only occasionally glance at the feed anyway (every 5 or so minutes)

Over the two quarters, we used lab usability tests, heuristic evaluations, and field usability tests to refine our design. Each evaluation technique gave us different feedback: in CS147, we did a lot of lab usability tests with students and teachers (giving them individual tasks to do rather than focus on more natural use in field tests, which required getting class time). However, it was difficult to decide which feedback was most important, particularly when some of the suggestions contradicted each other (for example, having more specific feedback vs having users spend less time typing questions are competing goals). Since we were asking students and teachers to look for errors, they might find different problems then they would in natural use. Similarly in 147, the in-section heuristic evaluation gave us a great deal of suggestions to work with, and helped us fix a lot of small problems. However, again, the recommendations/problems were somewhat conflicting. Moreover, information overload meant we had a lot of changes to work through, but they weren't necessarily ordered by frequency or importance – in particular, a "severe" error was not necessarily a common one.

By the end of quarter 194h, we really favored field tests with the actual users

themselves, in a realistic environment (which we were lucky enough to be able to do with 6 total classes). This tended to focus the feedback on what mattered (e.g. ease of in-class use on both teacher and student end), rather than what didn't (e.g. the few seconds delay between a user submitting a question on their phone and the website or teacher's phone updating). Further, this gave us a smaller number of more important changes to work on – and it was only after doing this a few times that we got to the point where we could go to classes without having a radical design overhaul afterwards. Heuristic evaluation and testing with non-core users (particularly on the teacher end) just wasn't as effective at highlighting what mattered or making the correct tradeoffs that a teacher might know are useful to make.

Final Interface

Mobile (iOS)

Final UI design Description (I. Functionality)

Users are able to create an account or authenticate themselves as Teachers or Students.

When authenticated as a Teacher, the user is able to create classes with a 5-digit class code, and view questions from the students in two views (Figure 3). The primary view displays questions from students in the selected class sorted by student-votes. The second view, namely private view, lists all questions asked to the teacher privately. In both views, the teacher has the ability to remove answered questions by categorizing a particular question as 'answered' which removes the question from the teacher's view but notifies the student about the teacher's response and allows the student to re-ask or resolve the question. The user authenticated as teacher has an additional feature: "Mute". Mute option (Figure 3 & 9) restricts students to only ask questions privately by disabling the public questions view for students.

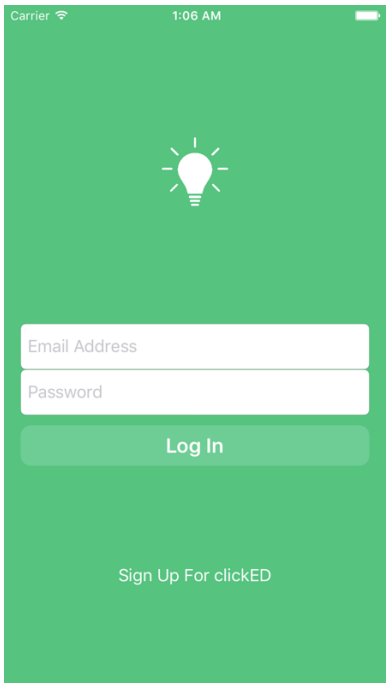


Fig. 1: Login Screen

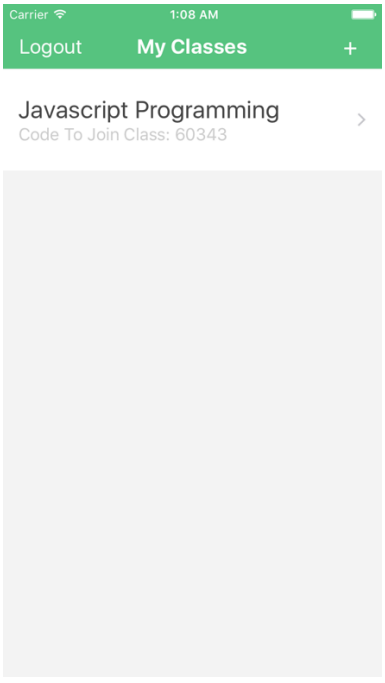


Fig. 2: Teacher Class List

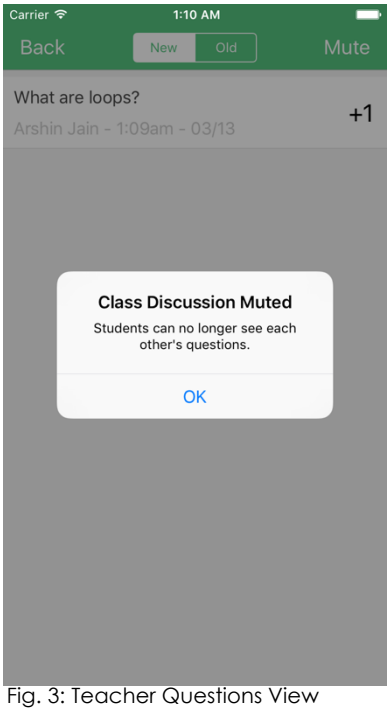


Fig. 3: Teacher Questions View

When authenticated as a Student, user(s) are able to enroll in courses via class code (Figure 10) provided by teacher. The user selects the desired class and they notice two views: "You" and "Others". The default view, titled 'You', allows the user to ask question(s) and view the questions they asked. The user also has the ability to mark a question as "I figured it out" which removes the question from the students' and teacher's view. The second view, namely, "Others" displays all questions asked by all students in the class. In this case however, the student names are invisible to other students but are visible to the teacher. This view also allows up-voting questions by other students to help sort questions by frequency for teachers. Another interesting feature is the "Whisper" button that allows students to ask questions privately.

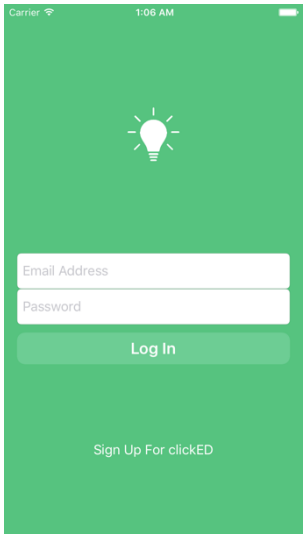


Fig. 4: ClickED Login Screen

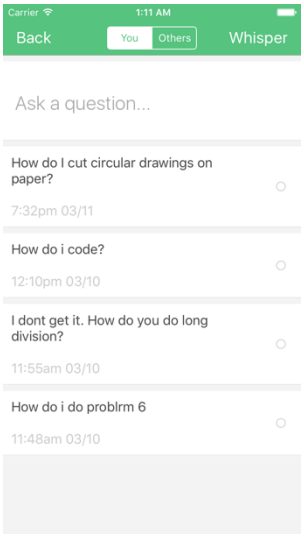


Fig. 5: Student - Ask Question

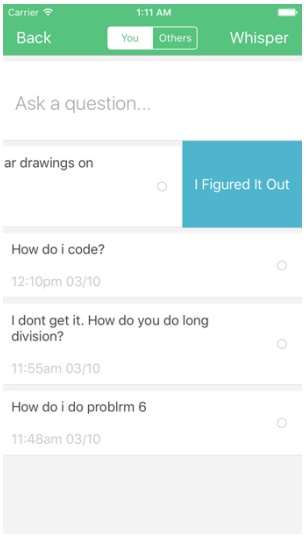


Fig. 6: Student: Swipe-dismiss

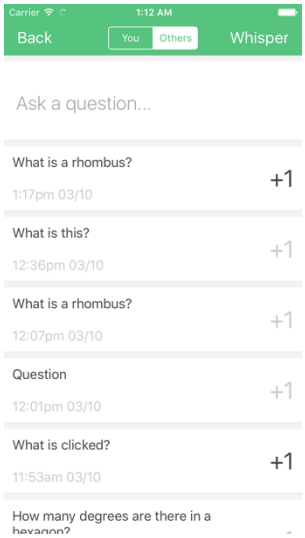


Fig. 7: Student: Up-Voting

Final UI design Description (II. User Interface)

A solid green splash screen greets users with a ClickED Logo (Figure 4) and title then a sleek login screen appears. There is an option to Sign Up below the login button, in a smaller rectangular shape. The teacher signup form is slightly elaborate but well spaced and each form utilizes rounded-corner style (Figure 4) for a better student-friendly outlook.

When authenticated as Students, the user is redirected to their class-list which is nicely displayed as cards. Class title encompasses a larger text size while the teacher's name is displayed in a smaller font with a lighter shade. The navigation bar has our green background, providing a consistent look throughout the app to indicate the user that they are navigating within our app. When in the class

selection view, the navigation bar offers a small "+" button to allow users to add a class. In order to delete a class, users need to gently slide the class (left) and a delete option would appear.

When the user selects a class, they notice a toggle to select the desired view ("You" or "Others"). The default view (Figure 5) allows users to type a question and submit (using the text "Done"). The questions asked by the user appear below and populate as a list with card design. The users swipe the questions to dismiss them (Figure 6), should they have understood the topic. The "Others" view allows the user to see questions listed by other students and they have the option to up-vote the questions (Figure 7), by tapping on a "+1" aligned right of the question over the same card. The "+1" prevents students from being influenced about the importance of any particular question. There is a nice Whisper button (Figures 5, 6 & 7) on the navigation bar for both aforementioned views that allows Students to ask a private question with a popup prompt "Ask a private question" (Figure 8) accompanied by relevant description and text input field.

The teacher view allows the user to select or create a class using a "+" button with similar interface. After selecting a class, the teacher selects between two tabs: "Private" and "All". The private tab allows teacher to view all questions asked privately whereas the "All" tab displays all questions asked by students. The teacher has the ability to swipe and mark questions as answered.

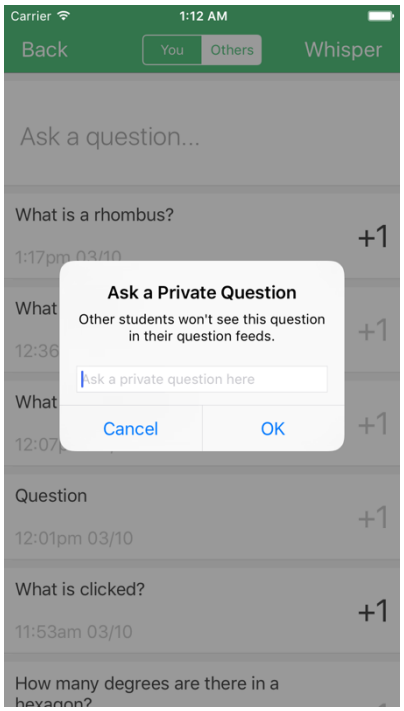


Fig. 8: Student: Private Question

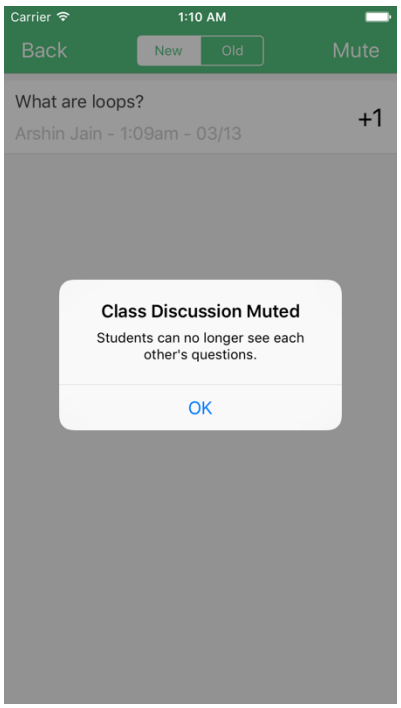


Fig. 9: Student – Muted Class

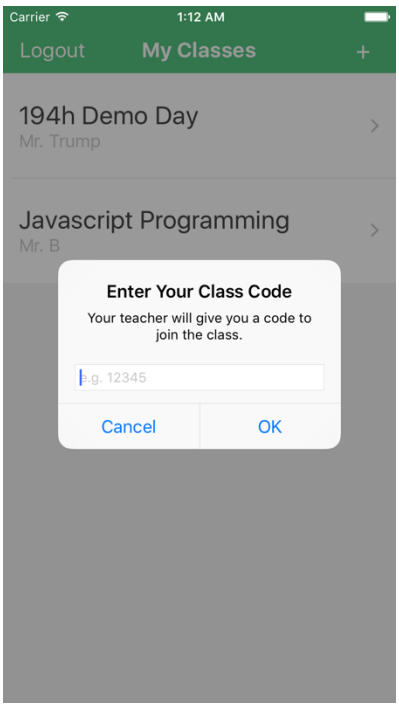


Fig. 10: Student – Add Class | Enroll

Web App (Hosted on Microsoft Azure)

Final UI design Description (I. Functionality and User Interface)

Teachers are able to create an account and authenticate into the Web App (Figure 11). When authenticated, all the classes created by the teacher appear as elements in a grid layout. The teacher has the ability to create or delete a class. Every class has a bright and colorful student image to provide a warm yet professional feeling.

When a class is selected, there are two tabs: 'Unanswered' and 'Answered' that display questions. The default tab, 'Unanswered' displays questions that are not answered by the teacher or resolved by the student. The teacher has the option to mark a question as answered, which would move the question to 'Answered' tab. The 'Answered' tab displays all questions that have been marked as answered. All questions display the time and student name as well. The questions are displayed in an intuitive stacked list with a light background for clear visibility.

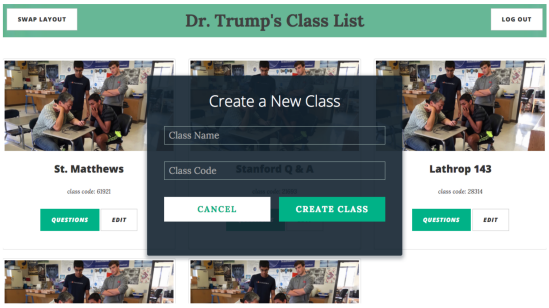
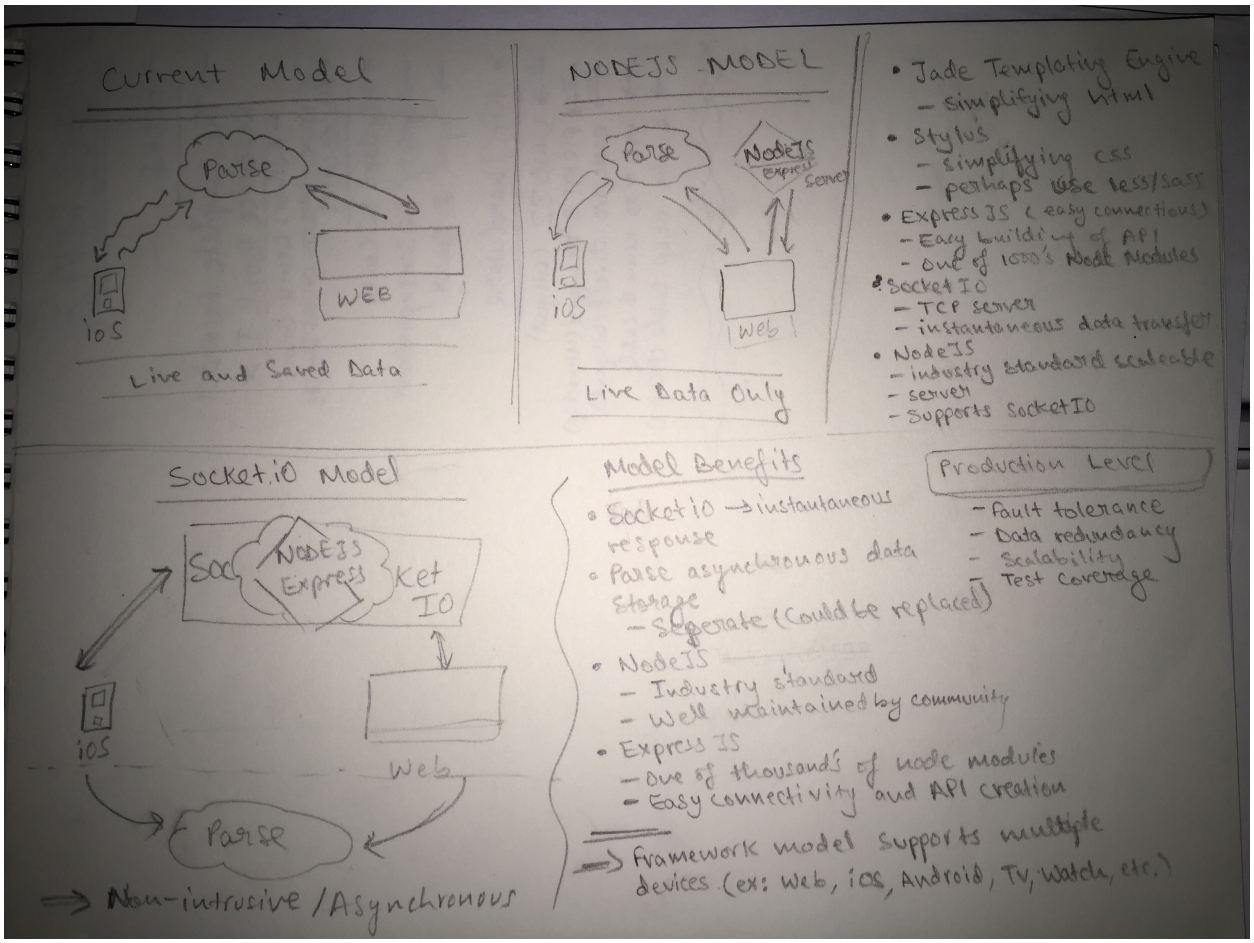


Fig. 11: ClickED Web App Teacher 'Create New Class'

Final UI design Description (II. Architecture)

We developed an initial website that relied on AngularJS to retrieve data from Parse (Backend-As-A-Service) and after HiFi 2, began migrating to a Web App using NodeJS, Express and multiple libraries to allow better control over the website and have a seamless UI. In order to resolve constant refreshing, we decided to implement Web Sockets to our Web App and iOS app. Web Sockets helped improve performance by a significant amount. Express allows us to easily build APIs for our future sustainability and problem-solving needs. We also set up a MongoDB database for speed and reliability—completing our migration to MEAN stack. Parse remains as a backup connection, storing backup of all the data and caching the data on the iOS app. We have successfully segregated Parse connection in order to discard it when necessary provided its retirement in the near future.

Our new implementation allows us to easily scale and deploy our application using Microsoft Azure (presently being used) or Amazon's AWS to serve a large consumer base. The migration to MEAN stack and SocketIO also enables us to seamlessly connect to multiple platforms with little, if any compatibility issues, meaning that this backend is widely supported by a majority of platforms.



Backend Migration and Benefits

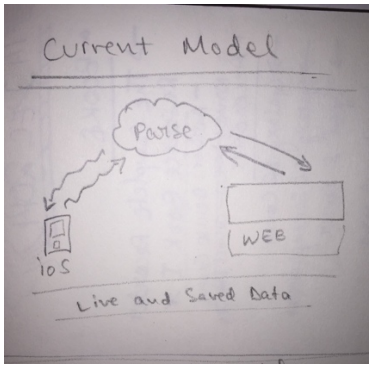


Figure 12: Initial Model

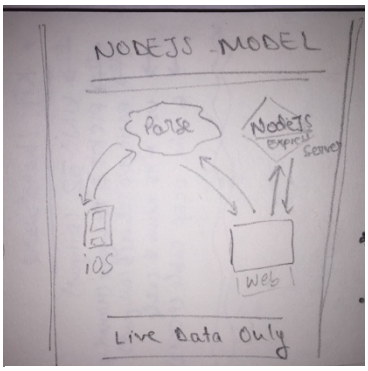


Figure 13: NodeJS Base Model

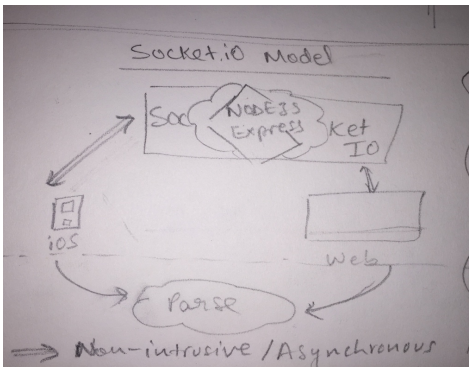


Figure 14: SocketIO – NodeJS - Express Architecture with standalone Parse

Unimplemented or Needed in Future

iOS

- *Viewing data transmitted instantly via SocketIO; The data is being received but users are unable to see it*
- *Restricted signup and security*
- *Thorough error testing*

Web

- *Displaying data transmitted directly by SocketIO; The data is being received but users are unable to see it*
- *Better Landing Page*
- *Better Animations*
- *Email verification for teachers*
- *School Signup and verification*
- *Student View*
    - *Login/Signup*
    - *Viewing questions*
    - *Indicating responses, ex: I figured it out.*
- *Security*
- *Thorough Error testing*

We did not implement the aforementioned features because they were not essential elements to the core usability of our in-class use case that we were optimizing for. We decided that due to constraints of time and necessity, that these nice to have features could be developed if we pursued the app further.

Note: No Wizard of Oz Techniques required or needed to make the product function as it is completely functional.

Tools Used and their Advantages/Disadvantages

- ***Parse*** *– Powerful framework for analytics, data storage and management but is retiring in the near future.*
- ***MongoDB*** *– Migration partially complete, industry standard with strong support.*
- ***XCode*** *– Seamless development and debugging; Displayed clear view of each screen.*
- ***TestFlight*** *– Permitted live testing of app in classroom setting and among team.*
- ***NodeJS*** *– Allows hosting as a Web App.*
- ***ExpressJS*** *– Easily build or incorporate APIs.*
- ***SocketIO*** *– Instantaneous data transfer seamlessly between numerous devices.*
- ***Microsoft Azure*** *– Cloud service by Microsoft to host the web app; very fast and reliable.*
- ***Lookback*** *– Wasted time—failed to support integration with Swift based projects.*
- ***Quicktime*** *– Used to record usage and user-testing as video.*
- ***iOS simulator*** *– Testing on multiple simulated devices and easily obtain screenshots of multiple sizes without device stored locally.*
- ***Marvel*** *– Initial Prototype built rapidly without code.*
- ***Adobe Suite*** *– Design and Video Production/Editing.*

Making It Real

Team

Our team consists of an iOS engineer, three web developers and a product manager. Krister will be able to iterate on the iOS app as we continue to get feedback from students and teachers actually using the app in the classroom. Because of the shift to Chromebook usage in classrooms, we know that we will need to flesh out the web version and build out the student side of the app. Jake, Tyler and Arshin will be able to build out the missing functionality of the web app and continue to iterate on it. Gordon has multiple years of experience in the edtech sector, and he knows many teachers across the country who would be willing to test out the app. He also has experience starting a company, and he will be able to guide clickED from a project to a real organization.

Business Model and Market Size

The clickED app will always be free. Schools and teachers don't have large enough budgets to be able to afford an app like clickED, so charging users for it would only prevent us from being used in schools that could really benefit from the app. Our goal would be to accumulate as many users as possible. This would enable us to improve the app and help us customize it to work in as many different types of school systems as possible. Based on data from 2009 from the National Center for Education Statistics, 40% of K-12 schools often use technology during instructional time and 29% sometimes use it. This number has gone up and will continue to rise, which benefits us because we succeed most in this type of tech-centric environment. We already have a significant audience that we can initially target to begin beta testing the app. Because the

app is free, we will be able to rapidly expand. Our key advantage won't be our technology, so a large, dedicated user base will be the key to our success against potential competitors.

Monetization

Once we have a foothold, we would be able to monetize through a premium version of the app that utilizes the question data that we collect on each student. We would sell a premium subscription to school leaders and district administrators in order to secure year over year revenue. We would provide teachers with a dashboard that helps them visualize what the most asked questions are and which students are struggling the most. This would help teachers address immediate issues, improve their lessons for other sections of the same class, and update future versions of the class. Administrators would be able to use this data to help teachers improve their teaching. Professional development is currently one of the highest spending areas for schools and districts. Admins can see what questions students had and personalize their professional development instruction to help teachers better address those topics.

Impact

Our app has the potential for both immediate and long-term impact. If we can help teachers address students' confusion in real-time, then we can avoid the trap of teacher's only finding out students don't the understand the material until summative tests. Teachers will be able to tell quickly within a few weeks if the app is helping students understand the material better. This will hopefully make the app an essential element of every classroom. Then after we have collected sufficient data, we will be able to offer a free trial of the premium

version in order to get teachers and admins hooked on the impact of our data. Teachers often complain that generic professional development seminars don't actually help them improve their teaching. With our app's data, admins can provide targeted instruction about how to teach specific lessons based on the concepts that generated the most questions. We can enable admins to provide data driven instruction to their teachers and provide the same thing for teachers with their students.

Summary

What is your key innovation?

We received great feedback about the potential of clickED from both students and teachers. We knew that we were solving a real need for both students and teachers based on our needfinding and personal experience, but we didn't know if our solution would actually solve the problem. We learned from teachers that in order for them to adopt technology it needed to work seamlessly within their current teaching style, and it needed to be simple. The first impression is key, and teachers don't give technology second chances. We made sure that every new feature and UI change stayed true to these two principles. Our final tasks reflect our desire to maintain the natural classroom paradigm rather than disrupt it. Students ask questions, teachers address those questions and students either figure it out or ask follow-up questions. We created a digital version of these typical interactions in order to give students who are currently not succeeding in the current paradigm a different medium through which to participate. This makes our app a natural fit into any classroom because it doesn't require students and teachers to drastically change their current behavior.

What will your key impact on the world be?

Students are currently silently slipping through the cracks, and teachers don't have an effective means of identifying them before it's too late. If we provide a model in which every student can participate, then we can empower teachers to help their students succeed. If a teacher isn't able to address every question in class, they still have access to each student's unanswered questions, and they can follow-up with them after class. We are providing teachers with critical data that they don't currently have, and we can do this by taking advantage of the technology these students are already using in class. Currently, most students are using iPads and Chromebooks as glorified notebooks. Students would be able to use our app side-by-side with their note-taking app and easily transition between the two. We can help classrooms better utilize their technology and improve the level of participation without increasing the level of distraction.