audio reality

Augmented reality for educational equality

Team:

Manuel Rolon-Osuna: Web Developer and Project Design
Kylie Holland: App Developer and Project Design
Pramod Kotipalli, Augmented Reality Developer and Project Design

Table of Contents

Problem and Solution Overview

Our group wanted to work on a problem that would assist those with hearing impairments. To get a holistic view of the problems hard of hearing folk face, we conducted various interviews with people close to the issue at hand. We found that the things that accompany what's said in the lecture are critical to the learning experience. Existing technology provides captions in lectures. However, these captions have significant short-comings, which impedes equal education access for deaf students. Augmented reality will complement existing captioning technology to create equal learning spaces. The implementation is feasible with the current infrastructure. We decided upon two mediums: a digital app for the management of classes and smart lenses for displaying captions.

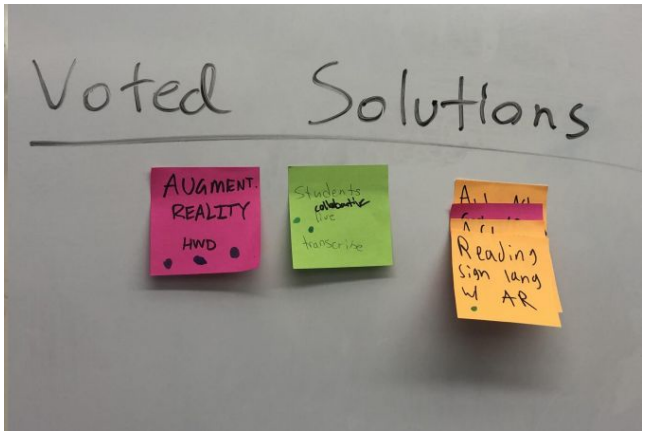

Figure 1: Brainstorming led to many strong potential projects, which we narrowed down to three before making our decision.

Tasks & Final Interface

Task 1 (Simple): Understand what's said in a lecture

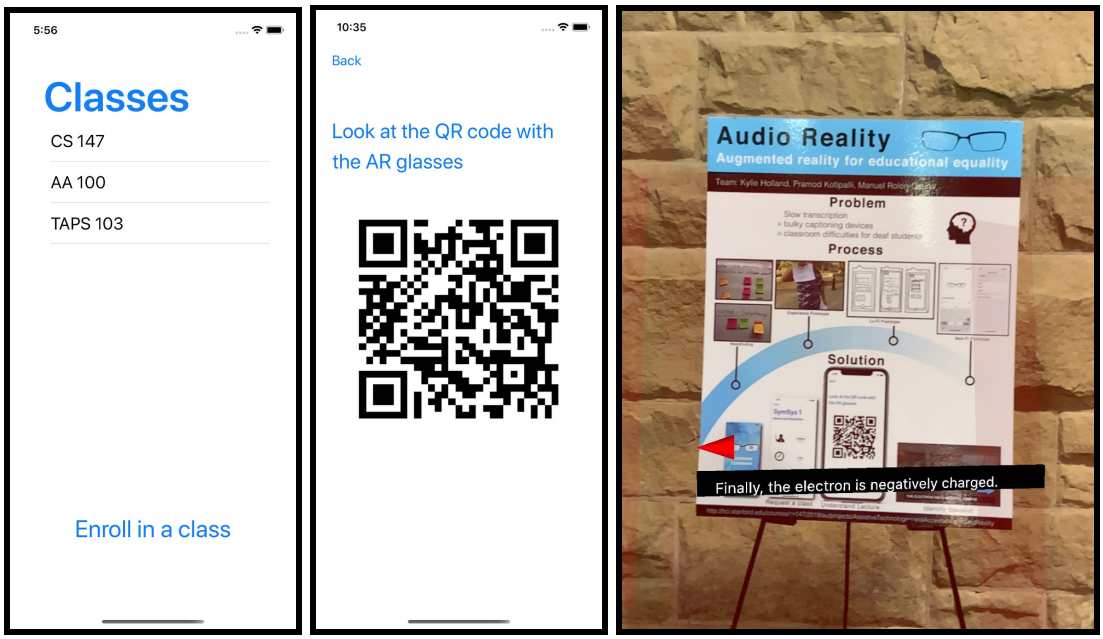

Figure 2: Our final interface. The companion app is shown in the first two screenshots. The ARKit prototype is shown in the third screenshot.

The first task we assumed every user would want to achieve using Audio Reality is to be able to understand what is being said in the lecture. Thus we designed an augmented reality approach to closed captioning in class. While casting the captions onto a plane was difficult to code, the user task of reading the captions is easy as the captions will always be positioned in the line-of-sight. Any end-user would want to be able to see captions in a class if they are using Audio Reality, and achieving this would be a user's simplest task.

Task 2 (Medium): Identify who is speaking

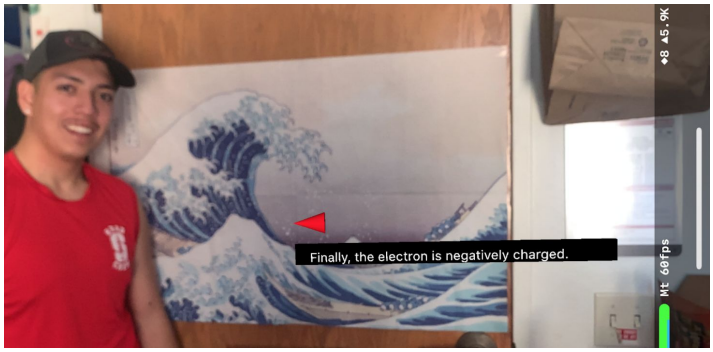

Figure 3: The speaker is identified by the red location marker.

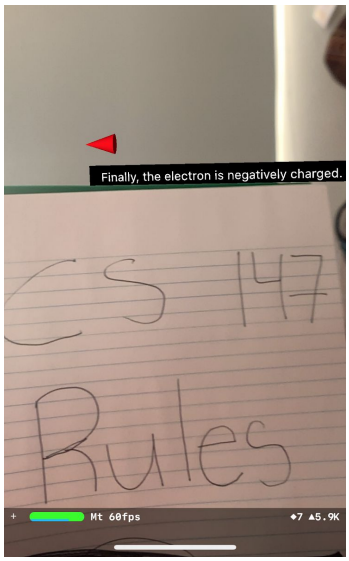

Figure 4: Even when a user is looking away the location marker will track the speaker.

Along with understanding what is being said in a lecture, a user will want to know who said what. Our need-finding revealed that tablets used to display captions in class require the user to look down and force them to lose track of who is speaking. Included in the augmented reality captioning is a red cone that serves as a location guide. It points in the direction of whoever is speaking. While watching the lecture, a user can glance at their captions and track the speaker.

Task 3 (Complex): Ensure a class is Audio Reality compatible

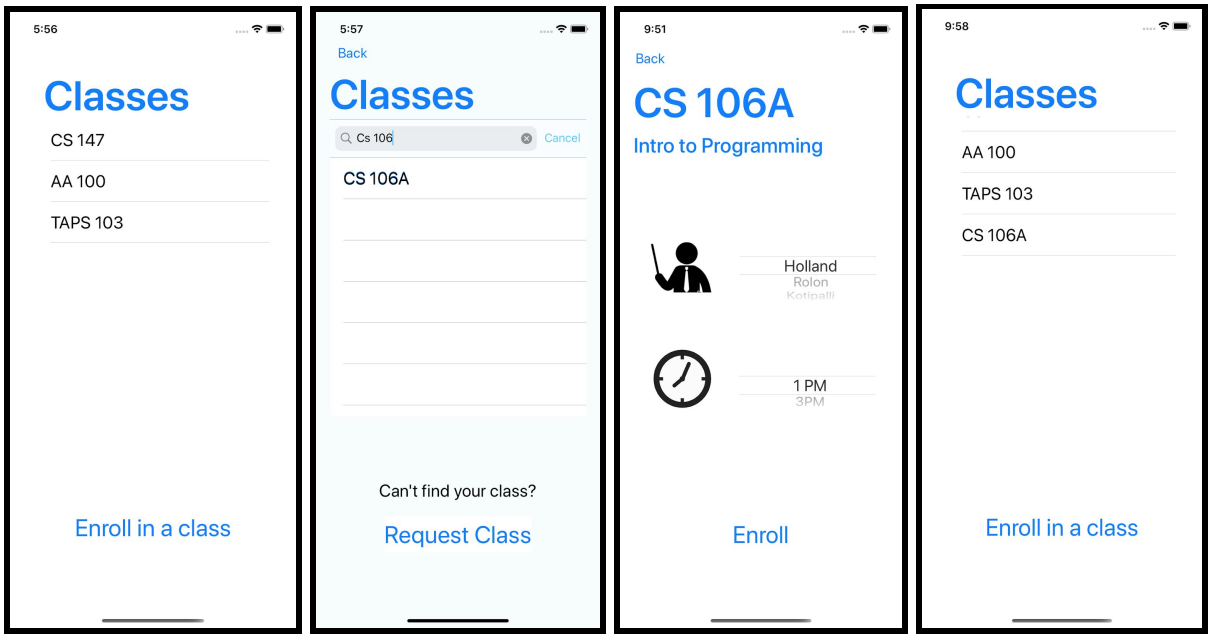

Figure 5: (1) The listing of available classes. (2) Searching through the list of available classes. (3) The enrollment page for a new class. (4) The updated class listing after enrolling in CS 106A.

The third and most complex task is ensuring that a class supports Audio Reality. To do this a user will first login to the app, and then be shown the list of classes they are actively enrolled in. By clicking “Enroll in a class” a student will be taken to a new view that presents a list of classes that already support Audio Reality. Users can then search through the available classes or search for a specific one. Clicking a class from the list will open its enrollment page. From the enrollment page, users can indicate their preferred instructor and time, and after enrolling in it the class will be displayed on the student’s home page. Figure 5 illustrates the task flow for a student enrolling in CS106A.

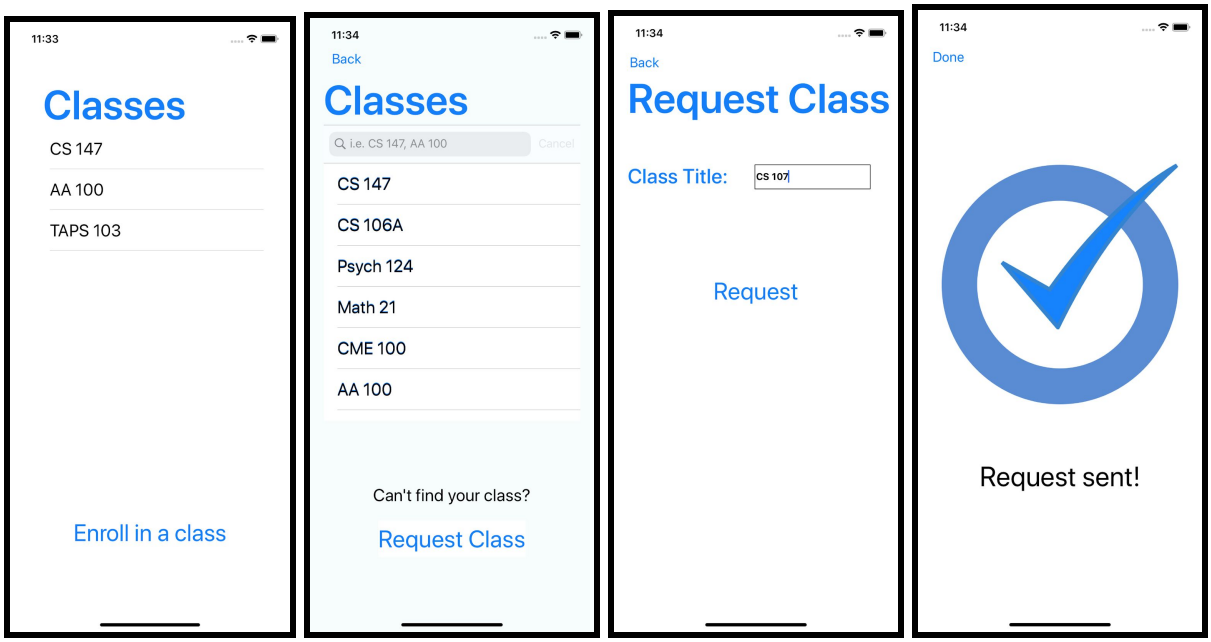

Figure 6: The workflow for requesting a new class to support Audio Reality.

Users can also request classes if it is not available in the database. From the class enrollment screen, a user can tap “Request Class” and transition to a new view. This new page will present a text field in which a user can enter the class they want Audio Reality compatibility for. Pressing “Request” will send the inquiry to the school’s Office of Accessible Education, and then the user will be contacted once their request has been resolved.

Audio Reality is meant to create accessible learning spaces. Users of our product will want to have Audio Reality available in any classroom they will find themselves in. Should Audio Reality not be available in a classroom, a user will be limited and feel undervalued. To avoid this we offered an option to request a class. This design model allows Audio Reality to be scalable with a growing class database. Given the plan to be cooperative with existing infrastructure, it was critical to design for the task of adding classes and requesting them.

Design Evolution

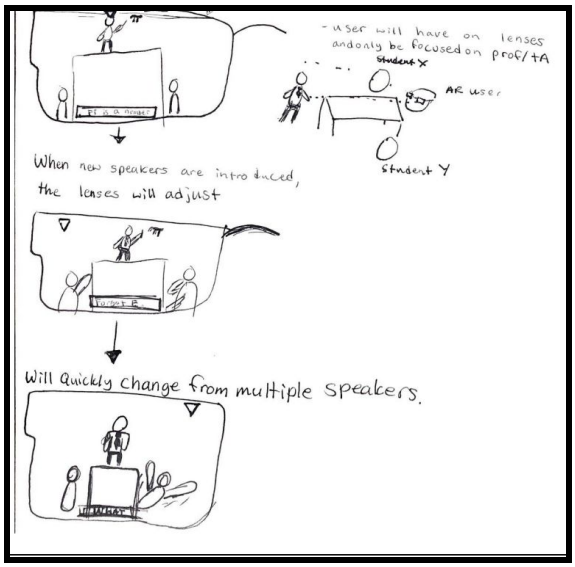

Figure 7: This storyboard illustrates how the lenses would operate in the presence of a second speaker when in lecture or discussion.

Our conversations with hard of hearing folk and advisers in the OAE sparked an interest within our group to make an augmented reality approach to captioning. To begin prototyping our idea we turned to paper versions of our implementations. Our prototyping began with us testing the user experience. We contacted Stanford students and local citizens who identified as hard of hearing to provide feedback. Captions were written on slips of paper and held up to a user during a mock lecture. One of the testers noted that it would be helpful to include an explanation. Following the experience prototype, it was determined that a companion mobile app would assist the augmented reality lenses with class management.

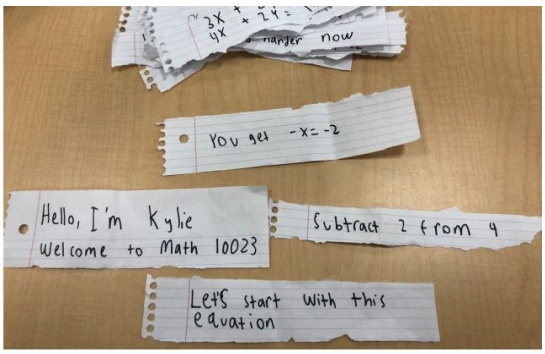

Figure 8: (Left) A hearing aid user tests our experience prototype of class captioning displayed in the line of view. (Right) Strips of paper were held up by a team member simulating the augmented reality captions.

We determined that the smart glasses interface would be cumbersome for class management, so we designed a companion registration app. Using a lo-fi approach of drawings on paper, we created the layout for the user interface of the mobile app. The first view designed was a login page prompting separate logins for users and administrators. User interactions on a view transitioned to another paper view of the app. Strips of papers were used as pre-filled inserts for search bars and text fields. Rather than have a keyboard they would use our paper cutouts. To demonstrate the augmented reality captioning, two team members reenacted a lecture while the last member played the role of “Audio Reality.” During the lecture, the member acting as Audio Reality would hold up pieces of paper labeled with what the lecturer was saying.

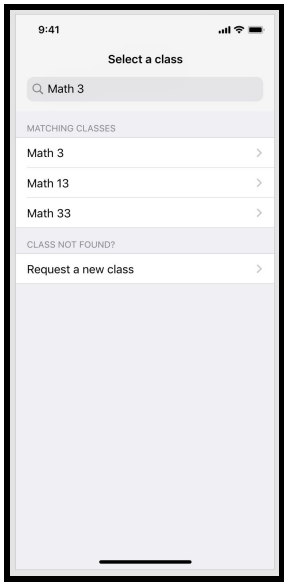

Figure 9: The updated class listing interface in our med-fi prototype.

The next step in our design process was to generate a digital rendering of our paper prototype for our med-fi prototype. In order to do this, we used Figma and Adobe tools to transfer the paper user interface to a digital medium. The app offered the functionality of searching for classes that had Audio Reality. If a class was not in the Audio Reality database, users could submit a request to get the class registered. In order to keep class management separate from the lens interface, a QR code would be used so the two interfaces could communicate.

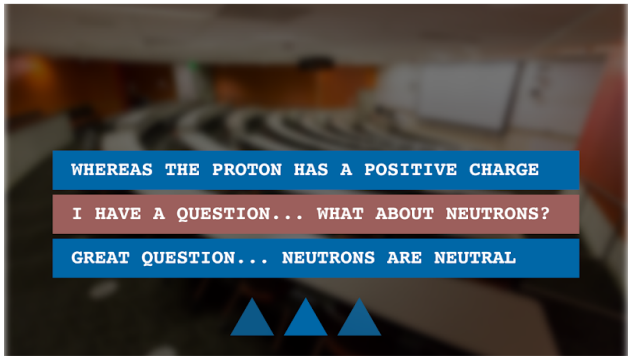

Figure 10: The Augmented Reality interface in the med-fi prototype illustrating the live-transcribed captions and indicators for where the active speaker is in a lecture hall.

To simulate the med-fi prototype of augmented reality we used a view within our Adobe XD app. Clicking on a class in the app loads a QR code. When tapped, this code transitioned to a view of a classroom with a dark tint. Displayed were some captions as well as arrows meant to indicate where the speaker was. This was the design our group had in mind for the interface within the smart lenses. It was to be realized in a similar fashion as Google Glass, displaying the captions in the top corner of the lens. Having two digital renderings of our interface, the next step was to develop functional hi-fi prototypes of Audio Reality.

To create the final, high-fi model of our project we used Xcode for both augmented reality and mobile components. Using the UIKit components we transitioned our Adobe XD prototype to a medium that would be functional on our devices. Since our mobile app was primarily for class management, the first view a user would see when logging in would be their currently enrolled classes. If a class was not listed in the student's home, they would need to enroll in it. To do so they can navigate to our class search page from the home page. The class search is designed to show all classes compatible with Audio Reality. Should a user try to enroll in a class not be compatible with Audio Reality, they have the ability to request it. Once back at the home page, they can select one of their enrolled classes and proceed to pair it with augmented lenses. Tapping on a class would open a QR code that would open up our augmented reality interface.

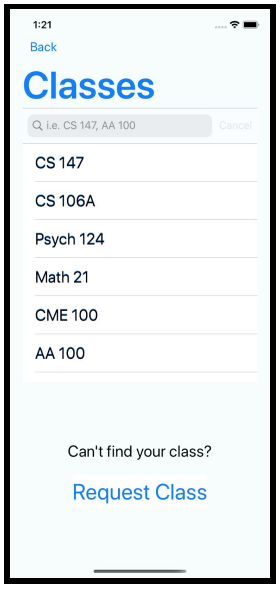

Figure 11: The listing of available classes supporting Audio Reality as seen in our high-fi prototype.

The final design of our augmented reality interface was built using the ARKit component of Xcode. Using this we designed how the augmented reality captions will appear. Since our app is not concerned with auto-generating captions we used hardcoded captions. The captions were simulated using AR Kit and were designed similarly to closed captioning. The background of the text field was black with white font, avoiding the color scheme of captioning used in the med-fi prototype. As mentioned, the captions were hard coded as Audio Reality would be broadcasting a caption stream. An additional affordance in the final design is a red cone that indicates the location of the speaker. This single position marker alleviates additional confusion by indicating the main speaker in a room. The marker is fixed in our final design as, in reality, it would rely on classroom technology like microphones.

The final realization of Audio Reality was completed using two implementations: an augmented captioning app and a companion registration app. The companion app is user friendly and simplifies the class registration process. Additionally, the augmented reality app highlights how captioning will be inputted to the Audio Reality lenses.

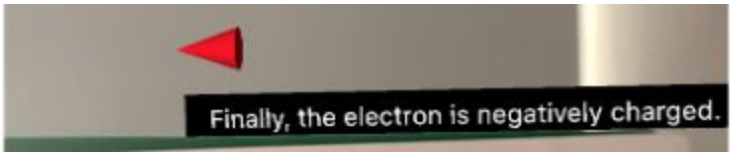

Figure 12: Captions and an arrow for identifying the speaker in a room. Displayed in Augmented Reality with ARKit.

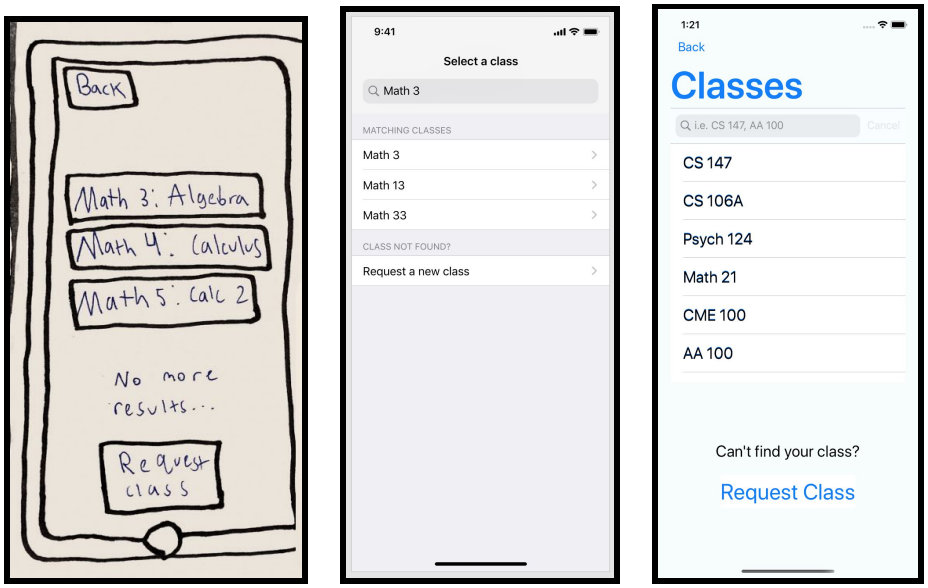

Figure 13: The evolution of the Class Search view of Audio Reality from the initial to the final design.

Major Usability Problems Addressed

4. Evaluation Statistics

Severity / Evaluator	Evaluator A	Evaluator B	Evaluator C
sev. 0	0%	0%	0%
sev. 1	66%	33%	33%
sev. 2	62%	23%	54%
sev. 3	77%	31%	23%
sev. 4	0%	0%	0%
total (sev. 3 & 4)	77%	31%	23%
total (all severity levels)	69%	29%	37%

***Note that the bottom rows are *not* calculated by adding the numbers above it.**

Figure 14: The results of our heuristic evaluation from three evaluators.

Quotes from our evaluators are presented in black boxes.

Mobile App - Searching for a Class

B. *H4 Consistency and Standards / Severity: 3 / Found by B When I arrived on the main screen, my first instinct (and what I did) was click Request a New Class, since that was the only button present, expecting it to be the functionality of finding a class that was already supported by the app (which is done by search). Fix: Don't show the Request a new class button until the user has actually searched for a specific class. Or, make the "Class not found?" text more prominent and associated with the Request a new class button, so it's more clear I should search first.*

The way the first pages were laid out in our medium-fi prototype confused evaluators, especially the purpose of the Request a New Class button. We fixed this by having the first page be a list of enrolled classes that took the user directly to the QR code. The user could then click to enroll in a new class, which would bring them to the search page. On the search page, we added a label clarifying that the Request Class button was for asking classes to be added to Audio Reality, not for enrolling in classes.

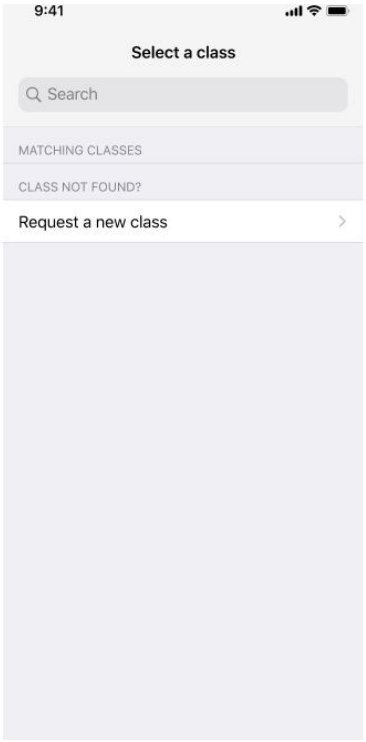

Figure 15: Med-fi: The search function is on the first page after login. The request button is the only visible function.

Figure 16: Hi-fi: The search function is on the second page after login. We clarify the purpose of the Request Class button by adding a label that says "Can't find your class?"

C. *H3 / Severity: 3 / Found by A, B In Select a class, I cannot delete or go back. This limits my user control since I have to keep going forward or restart the app. Allow me to delete or to exit search.*

A*. H3 / Severity: 3 / Found by A, C There is no way for me to go back from Select a class on the app which limits my control. I would have to use arrow keys which is available through Figma. Add a back button or home button.*

Heuristic evaluators pointed out that the lack of a back button limited user control when searching for classes. In our hi-fi, we added a cancel button in the search bar so that users could cancel their searches.

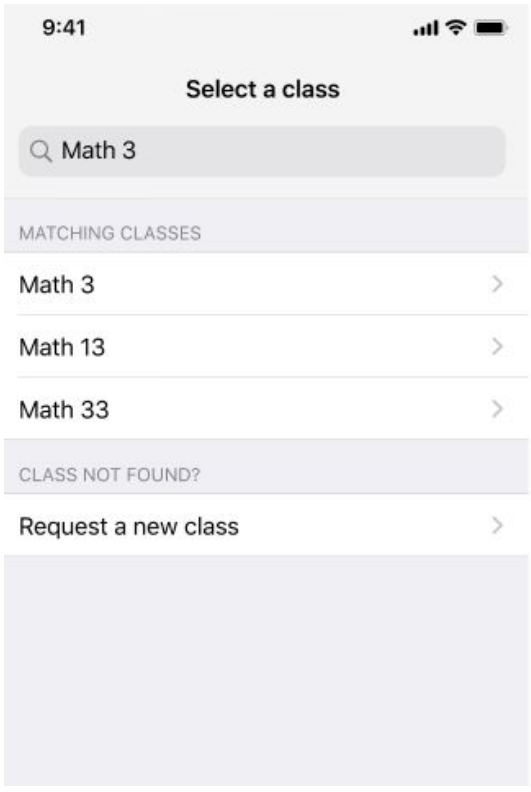

Figure 17: Med-fi: No way to cancel or exit search.

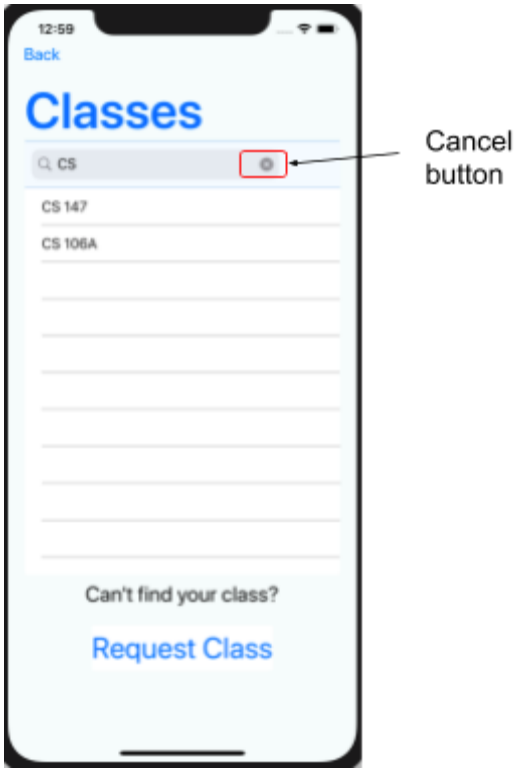

Figure 18: Hi-fi: Users can click the X in the search bar to cancel the search.

F. *H3 / Severity: 3 / Found by A, B, C I selected Math 3 but I cannot go back in case I chose it by accident. This limits my freedom. Add back or exit button.*

Heuristic evaluators pointed out that the lack of a back button limited user control when choosing classes. In our hi-fi, we added a back button after users selected a class. This allows users to go back and search again if they make a mistake. Additionally, we changed the class selection process so that users only search and select a class when they enroll, rather than choosing classes every time they want to launch captions. When selecting a class, students enter teacher and section-time information rather than going straight to the QR code.

Figure 19: Med-fi: The user goes straight from selecting a class to the QR code. No back button.

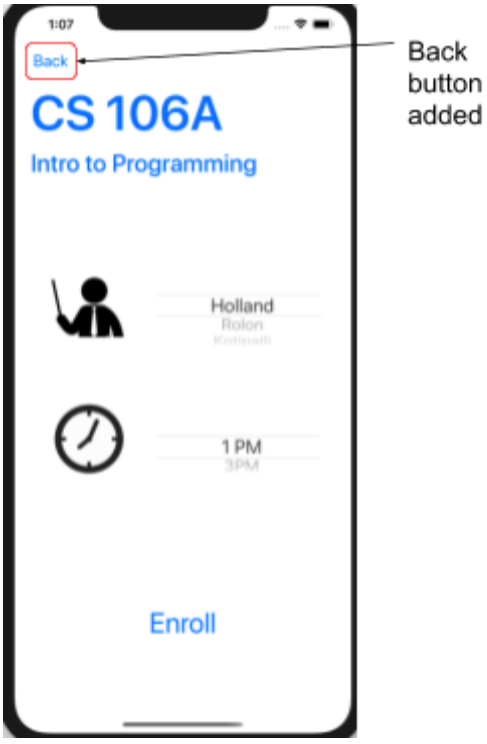

Figure 20: Hi-fi: After selection, the user sets up their class by selecting section times and teachers. There is a back button as well so that users can go back.

***D**. H10 / Severity: 3 / Found by A It is not clear what the app does since after I logged in, I just see select a class. It should provide a reason for searching for a class which is to be able to get captions. Such documentation would help and enhance the user experience.*

We responded to this violation by putting the search bar on the second page after login rather than the first. To get to the search bar, students must click “enroll class”, which indicates the reason for searching. We did not put in further notes, as a real user would be trained on this app and the glasses and would know that the purpose of selecting classes is to get captions.

Figure 21: Med-fi: The first page is the search page. There is no indication as to why the user is searching.

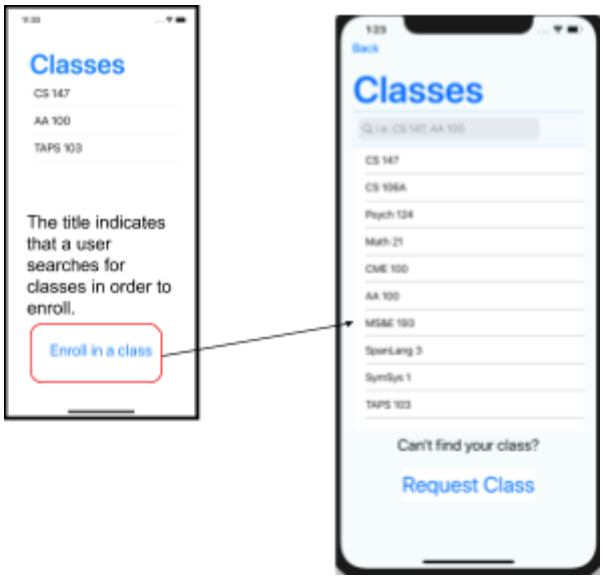

Figure 22: Hi-fi: The button label indicates that a user searches for classes in order to enroll in them.

Augmented Reality caption presentation

***B**. H3 / Severity: 3 / Found by A I cannot close the text on the glasses which is something I may want to. This limits my user control. Add an exit button on the app.*

We did not put in an exit button. We decided to put the captions on a separate app. The user would shut down the captions by quitting the app. We decided that having a cancel button floating in the user's field-of-view would distract them from the captions and understanding class content.

E. H10 / Severity: 3 / The directional cues are not obvious that it is indicating who is speaking. Add a note in the beginning that that is the purpose. This documentation would ensure that the user knows how to use the directional cues. Otherwise, it might be a distraction.

To avoid confusing directional cues, a single red cone/arrow highlights the location of a speaker. The cone will rotate as the active speaker changes in a room. Should a student raise their hand and become the primary speaker, the cone will react and point at them.

Students using this application will be working with the Office of Accessible Education which will provide smart glasses with captioning. As such, students will work with the Office and be oriented on how to use the device; in these conversations, the students will learn about what the directional cues mean and how to interpret them. Therefore, this feedback is not relevant to our application.

Mobile App - Requesting a Class

***D**. H10 / Severity: 3 / Found by A When on Request Class, I am not sure what is Code and Number. Perhaps give an example of where to find Code and Number and the format.*

In response to this feedback, we removed the course number request and changed the course code to the course title. We also present placeholder text with an example of what the users should enter.

Figure 23: Med-fi: Class code and number are not explained.

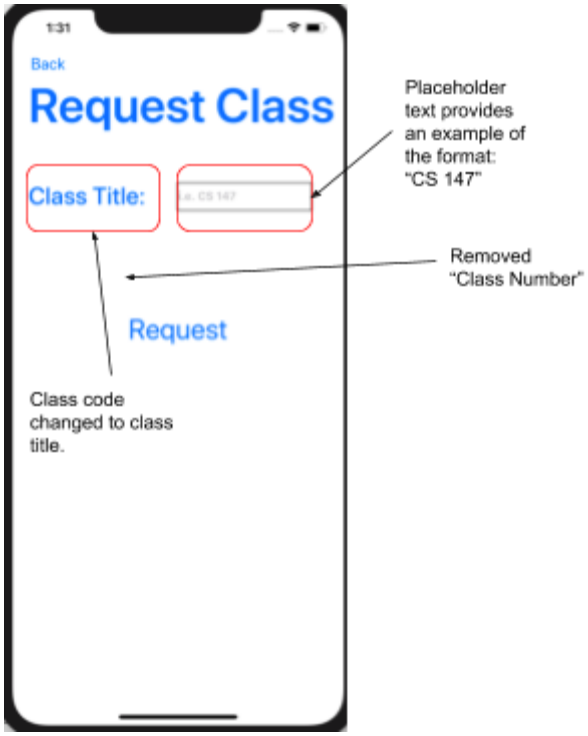

Figure 24: Hi-fi: Class code switched to "class title". Class number field removed, and an example added to the Class Title request.

***F**. H7 Flexibility and efficiency of use / Severity: 3 / Found by B If a user is enrolled in a class, they don't necessarily want to search for it every time they get to lecture. Provide functionality for a regular user to jump directly to a class they are enrolled in. Fix: landing screen could be QR code for that specific class, or there could be a list of enrolled classes rather than a blank list under the "Matching Classes" header.*

In response to this feedback, we made our first page a list of enrolled classes. The search bar is now used only to enroll in a class.

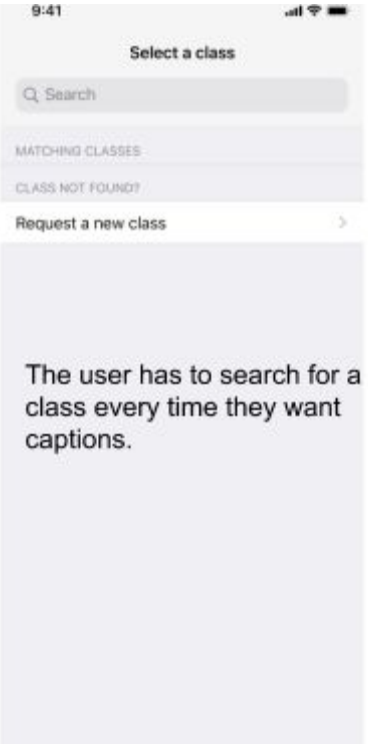

Figure 25: Med-fi: Long process every time the user wants to access captions.

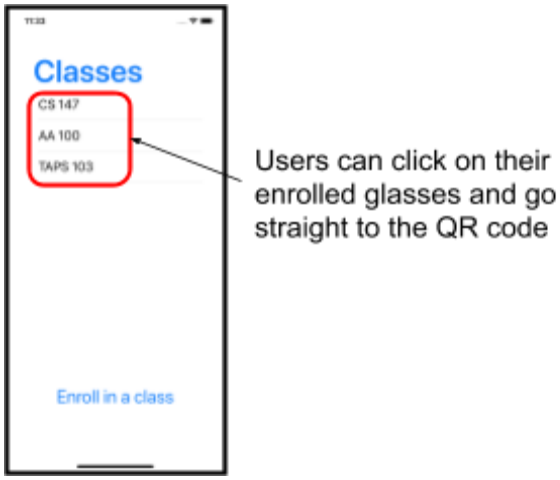

Figure 26: Hi-fi: The user can access their enrolled classes on the front page and go directly to the QR code.

Mobile App - Logging in as an Admin

H4 / Severity: 3 / Found by A I had to click on the password field to log in, instead of Login, which is not consistent with how I logged in as a student. I was confused by how to login as an admin. This shows a lack of consistency

H3 / Severity: 3 / Found by A, C After I filled out the student in Add Student, I could not click on Add Student or exit. Add both these options so I can have user control

H3 / Severity: 3 / Found by A After I left Device Edit, I am brought back to Device Detail where only Edit button works. I cannot go back to the home page which limits my user control. Make <Devices button work.

This is valuable feedback. For the high-fi prototype, we decided to omit the admin functionality so that the application only operates from the point-of-view of the student, who is our end

user. By doing so, we addressed this feedback and improved the consistency of the application.

Figure 27: Mid-fi: The username text field takes the user to a student account. The password text field takes the user to an admin account.

Figure 28: Hi-fi: Clicking login takes the user to a student account. The admin account no longer exists, as it was not useful to students (our end users) and did not help fulfill our three tasks.

Summary of Heuristic Evaluation

The major concern expressed in our heuristic evaluation was the lack of some functional back buttons and other buttons that are important for having an easy-to-use application. These were implementation flaws in our med-fi prototype that were addressed in our high-fi prototype. In addition, we removed the admin functionality which made some points of feedback irrelevant for the high-fi prototype. Based on other feedback from our evaluators, we implemented a few other features to improve the users' experience of the application as a whole.

Prototype Implementation

Tools

In order to build the hi-fi prototype of Audio Reality, we used Xcode and some of its components. The first tool we used was coding in Swift and UIKit in iOS. This tool allowed us to develop an iOS product while making use of existing packages to facilitate some steps in the process. Secondly, we used the ARKit library to develop the augmented reality component of our project. While the learning curve was steep due to our lack of familiarity, using Xcode was ultimately the best platform to build our app.

As mentioned, the existing packages in Xcode facilitated the app-building process. Existing tools such as labels and buttons were accessible through drag and drop. The storyboard functionality of the view controller also helped with the initial layout and saved us time by allowing us to drag and drop components rather than coding them. That being said, switching between storyboard and code was at times a frustrating process. Sometimes layouts and functionality that worked well on storyboards failed when the app was launched, and debugging was more difficult with storyboards than with code.

Having the iOS application work with devices of different sizes was a challenge. Working with "layout constraints" in Xcode's storyboard editor was quite difficult so we hard-coded the positioning of many UI components in the application.

Wizard of Oz

The transcribing of captions was not actually based on the ambient sound in the room. We simulated these captions. Also, identifying the speaker location was not implemented because it would require a lot of signal processing techniques that are outside the scope of this class.

Hard-coded Data

The QR code used to pair with the smart glasses is hard-coded. In addition, the listing of available classes was also hard-coded and did not integrate with systems managed by the Registrar or Office of Accessible Education.

Future Additions

Presenting captions in students' line-of-sight provides many usability and learning benefits. We would like to port the ARKit captioning application to a head-worn display instead of an iPad. Deploying such a solution to hardware like Google Glass or Hololens will be more usable and useful for students.

We'd like to add integrations with Stanford's systems maintained by the Registrar (e.g. for a list of courses) and the Office of Accessible Education (e.g.to ensure the proper infrastructure exists to transcribe a lecturer's voice).

In case a classroom doesn't have microphones available, we'd like to be able to use the microphone on the students' phones to transcribe captions.

Finally, we'd like to re-introduce the admin application that we removed between the med-fi and high-fi prototypes. This application will allow administrators in the Office of Accessible Education to receive and process student requests.

Summary

Our group's shared interest in an accessible project for hard of hearing folks saw the realization of Audio Reality. Our interviews revealed that hearing problems are diverse and are often unique per individual. Despite the disparity in hearing problems, the community shared that they experienced issues aside from captioning in class. Rather than build a model from scratch, our group decided the best solution would be to integrate augmented reality into existing classroom infrastructure. Our application aims to provide valuable captioning in students' line-of-sight so they can focus on learning instead of just understanding the words said in class.