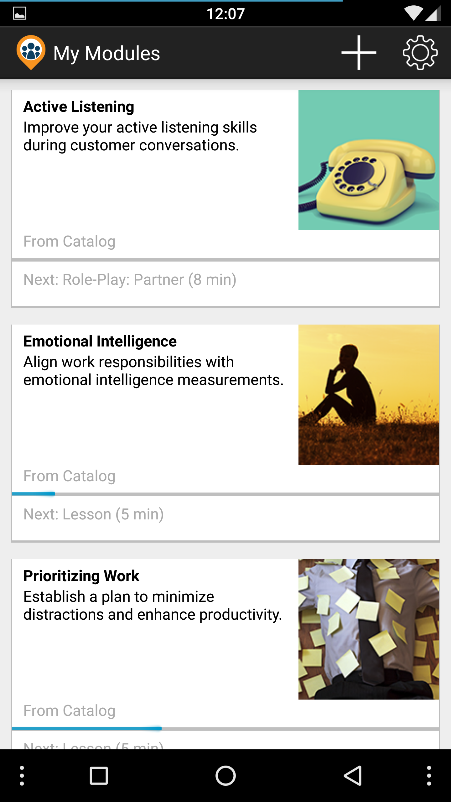
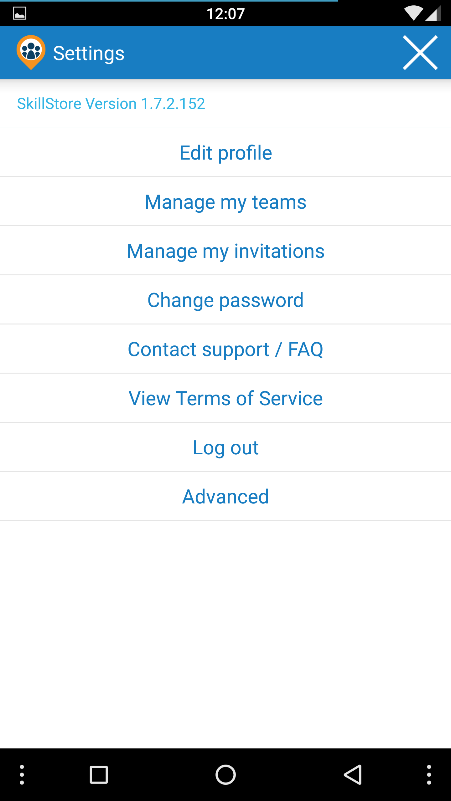
# Skillstore Product Overview

The following is an overview of the Skillstore Product. I spent 10 months working for Skillstore, and it’s unfortunate that the company has gone under. For intellectual property reasons, I cannot share the Skillstore codebase, but what follows is a walk-through of the product that I worked on, and some descriptions of the work that I did.

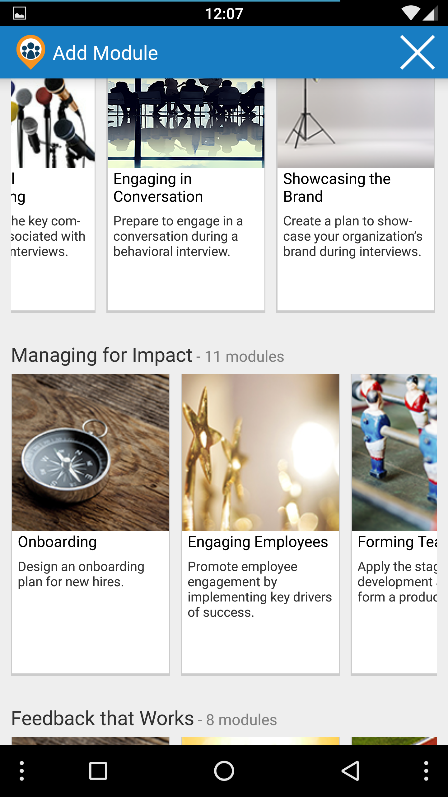
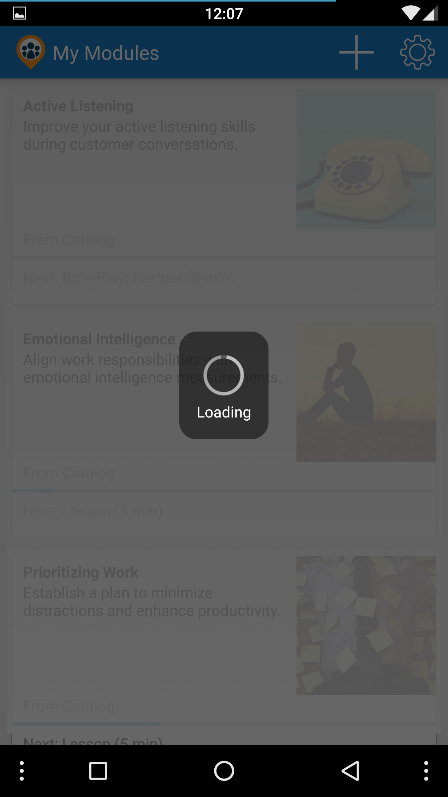
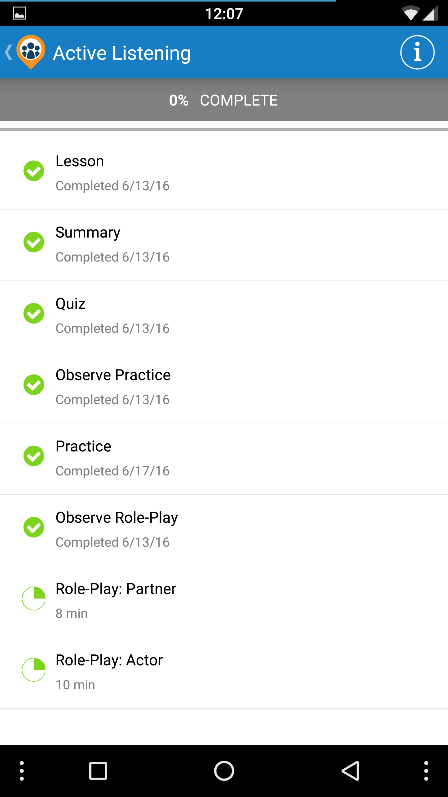
I was originally hired as an Android developer, but that quickly expanded to encompass iOS, and web app work. I’ll start with the Xamarin-based mobile app.

# Mobile

The Skillstore app was originally written with Xamarin Forms, however after many stability and quality issues, the decision was made to move the app over to the Xamarin native support. I spent much of my time with the mobile app “native-izing” existing views while fixing stability issues, updating features and documenting. “Native-izing” is essentially re-writing pages from scratch for both iOS and Android.

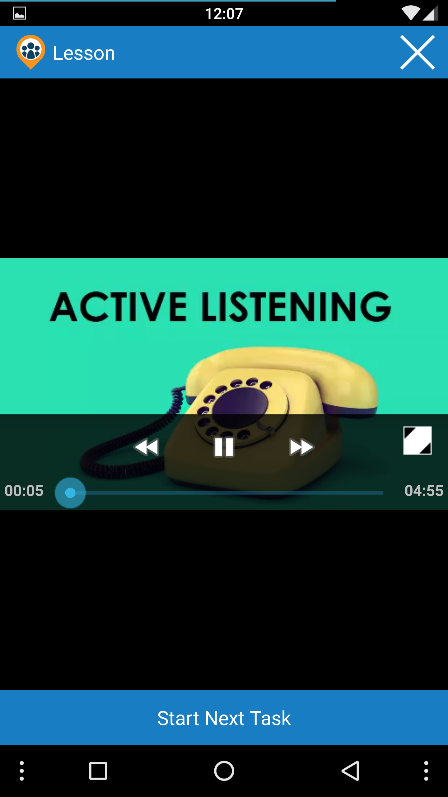
  

These are the first 3 screens in the app: Login, My Modules, and Settings. (Register/Forgot Password, and Settings pages are all simple and roughly the same.) The highlight here is the center image. I updated the My Modules page to be native, dramatically improving performance by changing the code to asynchronously load costly resources like the module images. This page also supports pull-to-refresh.

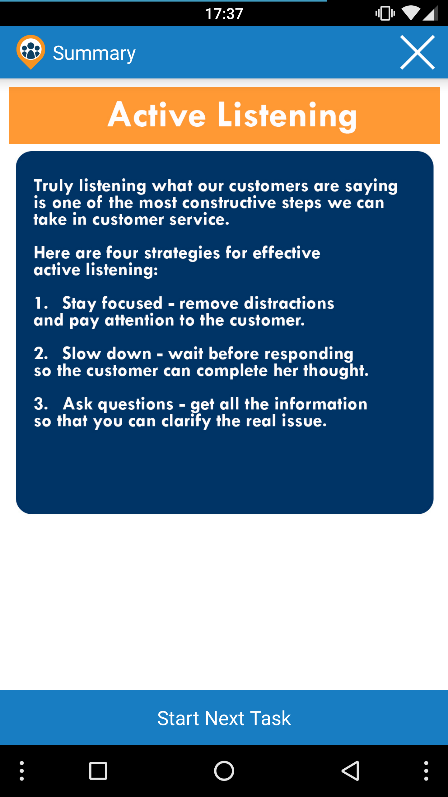
  

If you hit that plus sign on the My Modules page, the app would take you to the module catalog where the user can add new modules. I also streamlined the image asset loading for the catalog. The picture on the right is the inside of a Module. Modules are broken down into components, each of which is listed below. The module summary page shows the completion status of each component, and has a pull-to-refresh functionality. This module detail screen shot was taken on the dev branch, which was broken by a recent Xamarin Forms update – that’s why the completion numbers are broken.

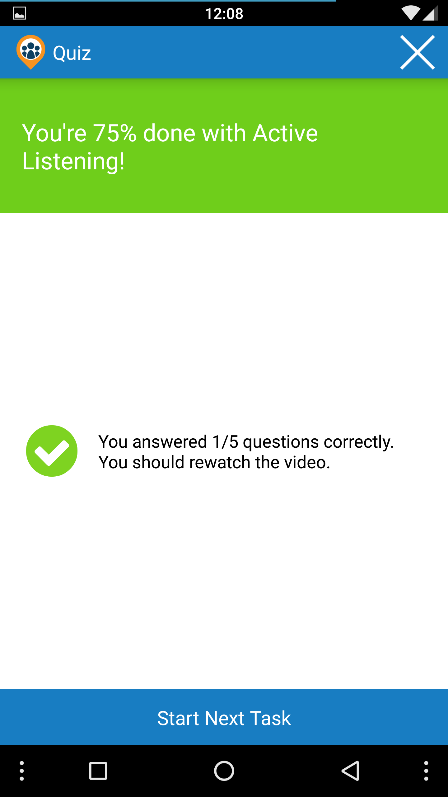
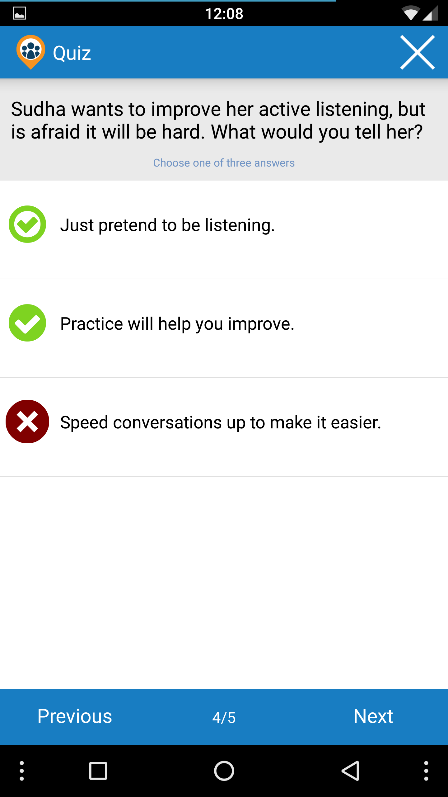
## Lesson

Skillstore was lesson based, where each lesson was a roughly 6min video explaining some aspect of business soft skills. The lesson view was totally re-written from the ground up by myself. The lesson view is Xamarin Native, not Xamarin Forms, and uses a custom video player widget I made to encapsulate video player boiler plate code. On iOS, the video player widget is accompanied by a custom video controls widget, to replace the terrible default iOS video player controls.

## Summary

I also rewrote the summary view from the ground up. This is a view that can display HTML, plain text, or video – depending on what the server hands it.

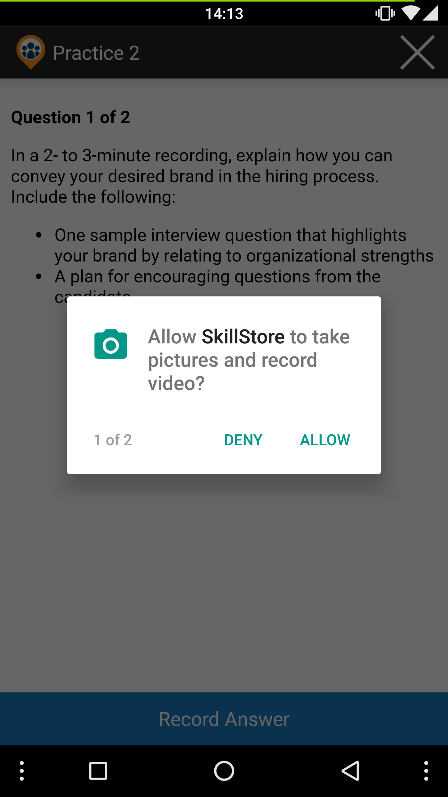
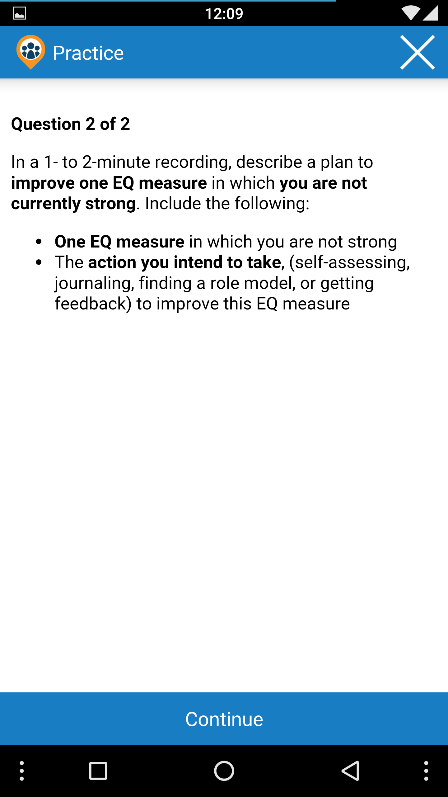
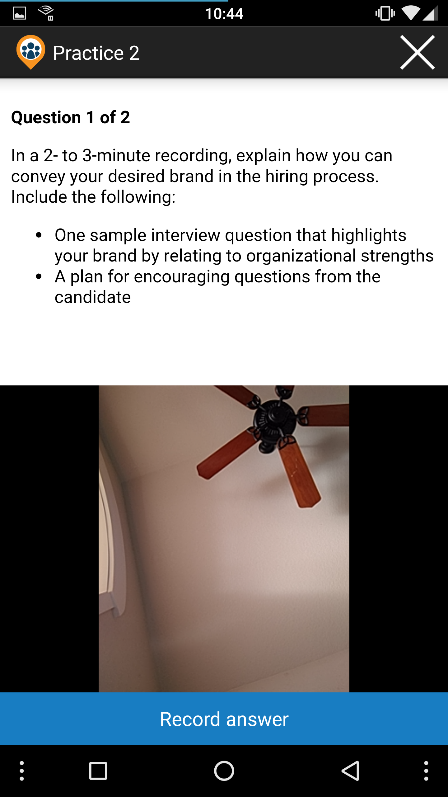
## Quiz

Quizzes are for the learner to quickly check their understanding of a topic. For pedagogical reasons, the quiz gives immediate feedback when the user selects an answer.

I also rewrote the quiz views from the ground up as Xamarin Native views. This involved creating a native Carousel widget to approximate the behavior of the Xamarin Forms carousel.

I particularly like the Carousel I wrote, as it uses reflection to allow a Carousel ViewModel to change the carousel’s child views by manipulating a list of child ViewModels. While reflection is generally a no-no, this keeps the views clean, and only concerned with presentation. The ViewModels should manage the actual view logic.

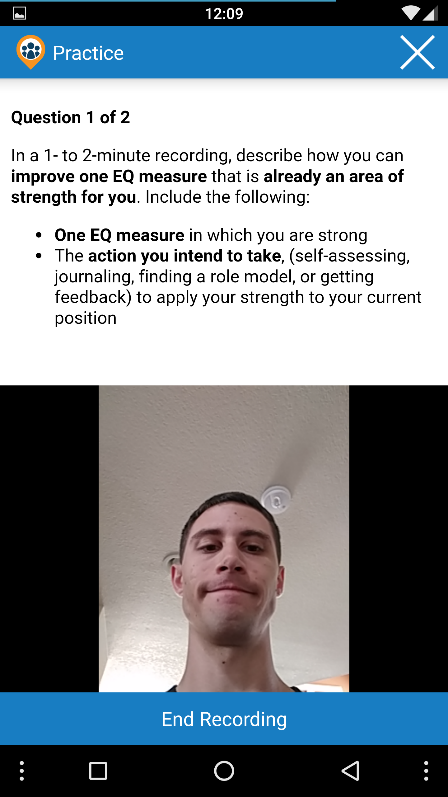
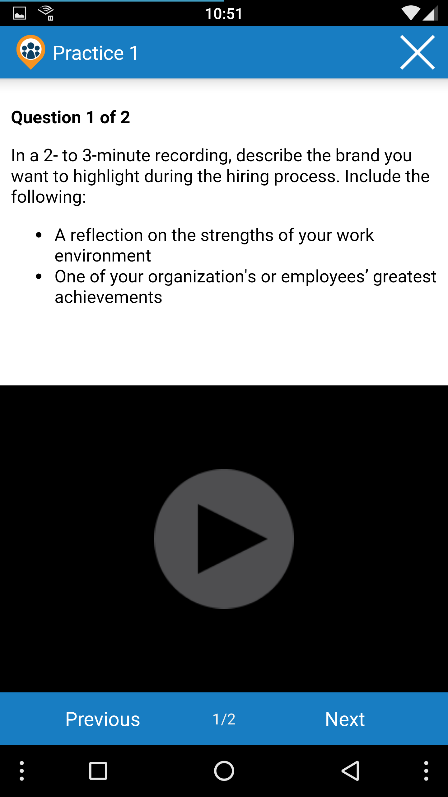
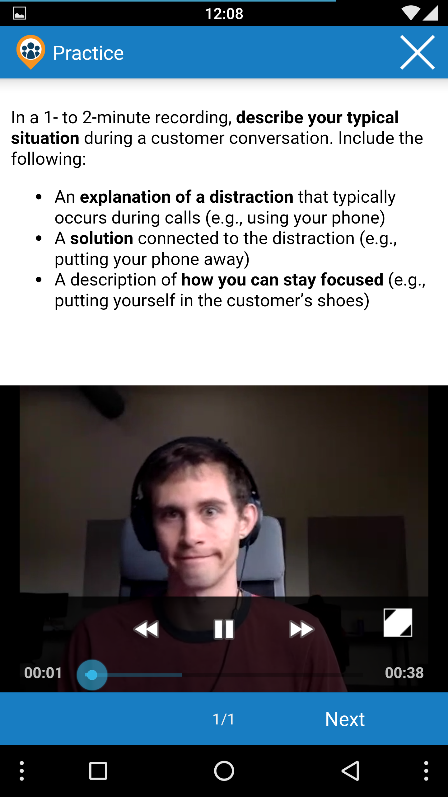
## Practice

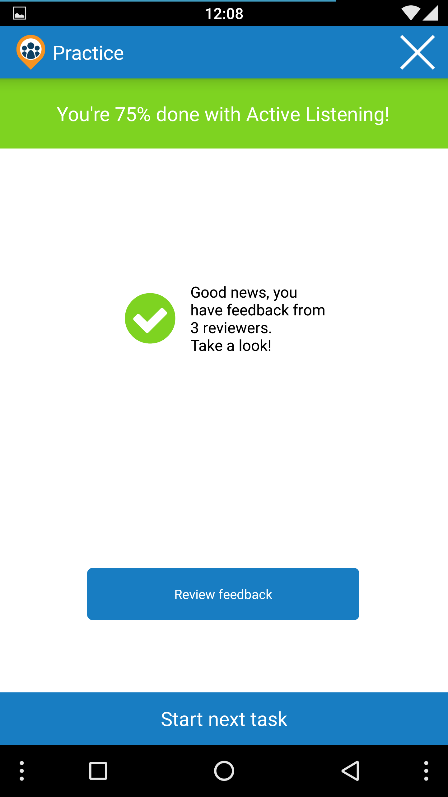
Practice is where a lot of the value of Skillstore was to be found. The idea was that the learner would see some prompt, record a response to that prompt, and get feedback from other users on that recording. This is in following with the architecture of Massive Open Online Courses, where students are leveraged as graders thus resolving the scalability problem of a single professor. Practices are inside a carousel, and can have multiple parts before submission to peer review.

In this first practice screen shot, you can see the very first commit I ever made at Skillstore: it was adding support for the Android 6.0 permissions system. This was a bit of an unusual task, as the Android permissions system relies on callbacks in the main Activity; however, Xamarin Forms apps sit in a single Activity and abstract out most of the Android system which mucked up the whole affair. I ended up using a semaphore system to handle in progress permissions requests.

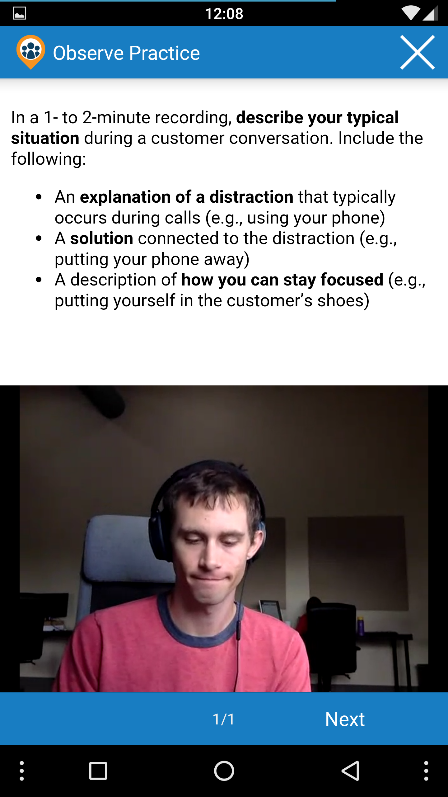
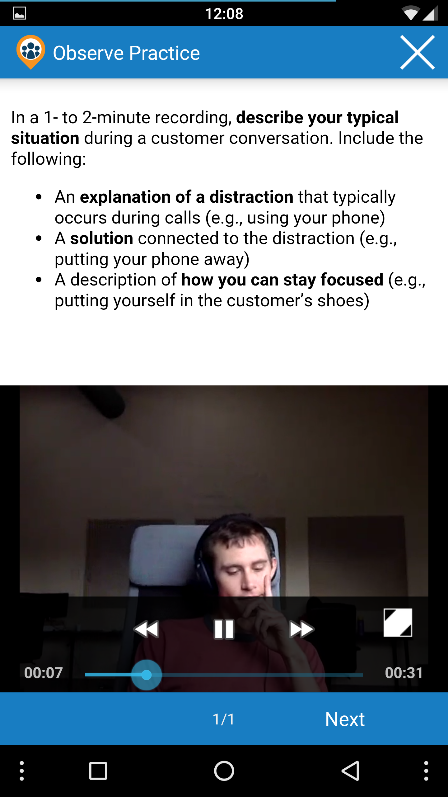
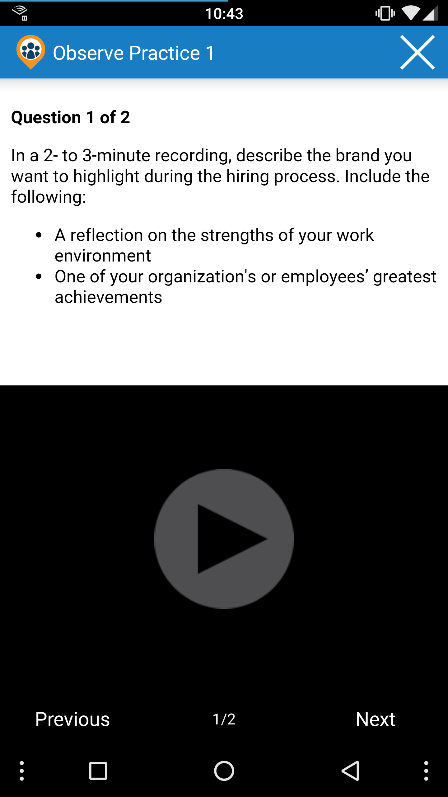
Picture number two shows a prompt for the user, and picture number three is what video capturing looks like. Recording is managed by that single button along the bottom.

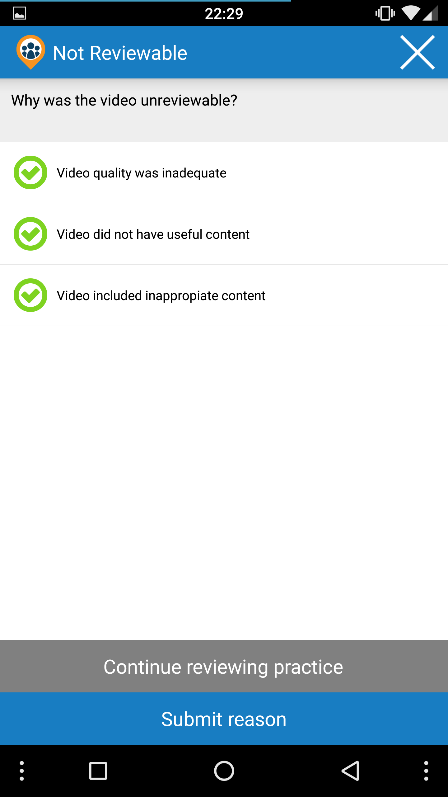
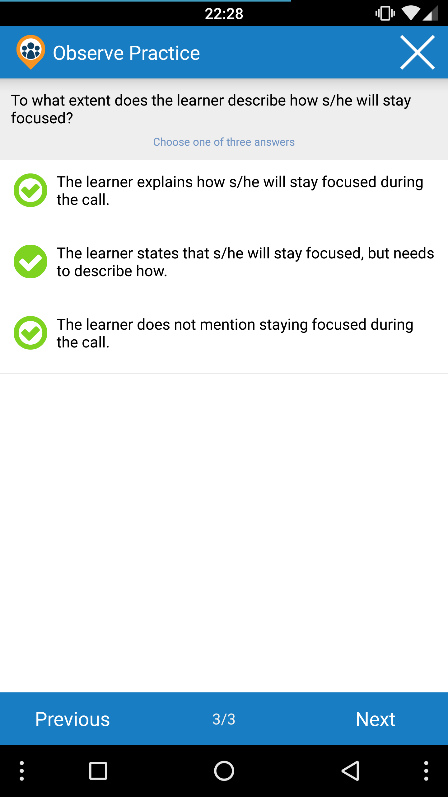
  

Here you see me in the middle of a recording in the picture on the left. After the user has made a recording, they can obviously review the recording. The recording review leverages the same video player widget that I made earlier.

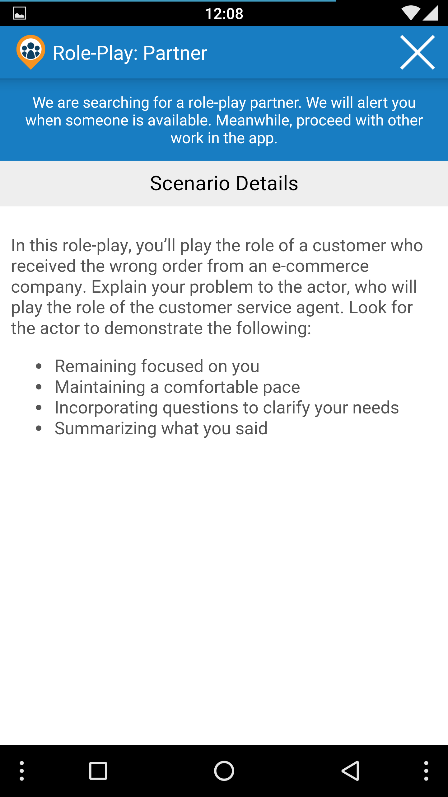
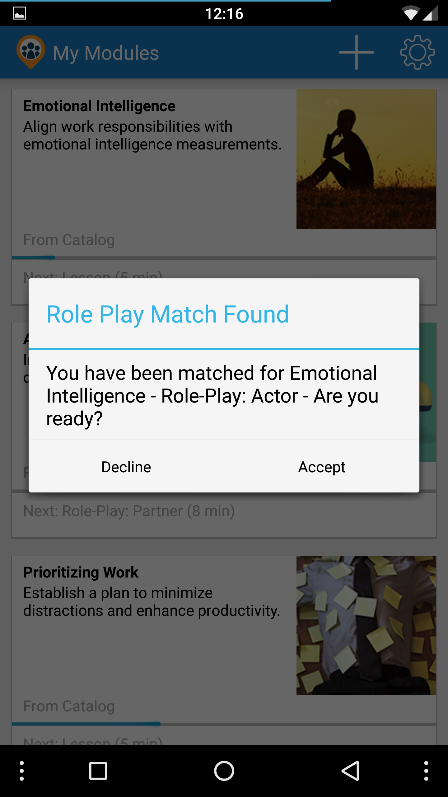
In fact, this is another view where I tossed a bunch of code and rewrote everything from scratch. The original code was a massive, comment-less state machine, which I refactored down into a much more reliable, faster, and simpler state machine spread across more files with many comments.   
  
And of course, when the user has received feedback, they’ll want to see it. That blue button appears when the user has feedback. The feedback looks like it does in role play – which looks a lot like quiz.

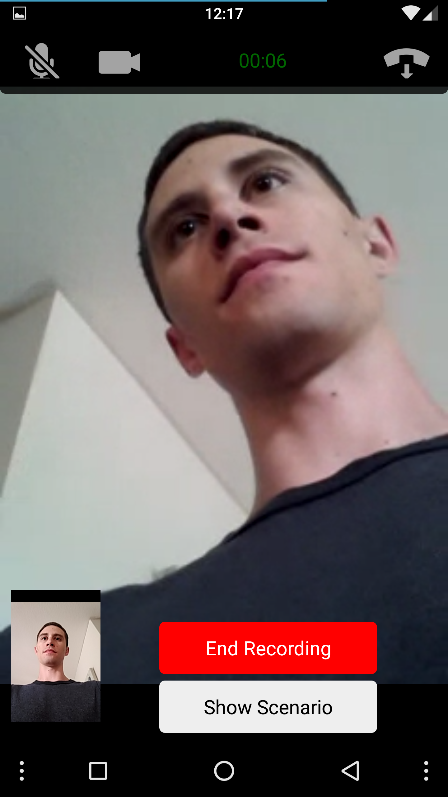
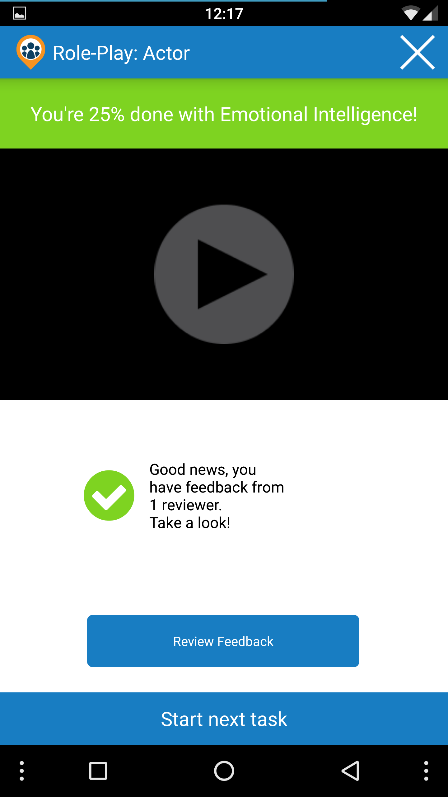
## Observe Practice

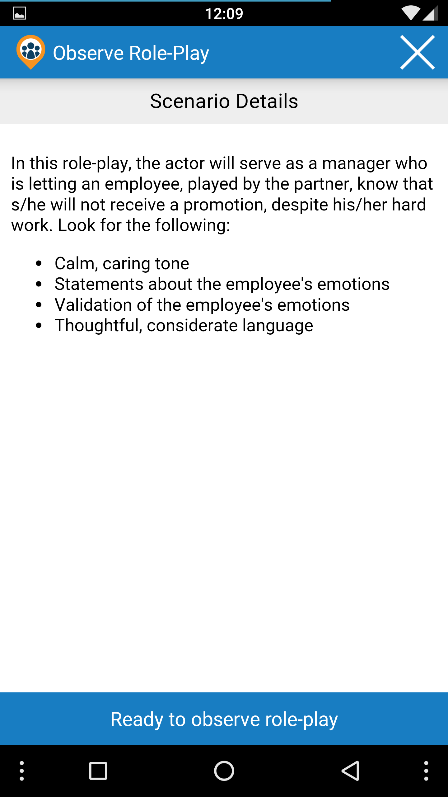
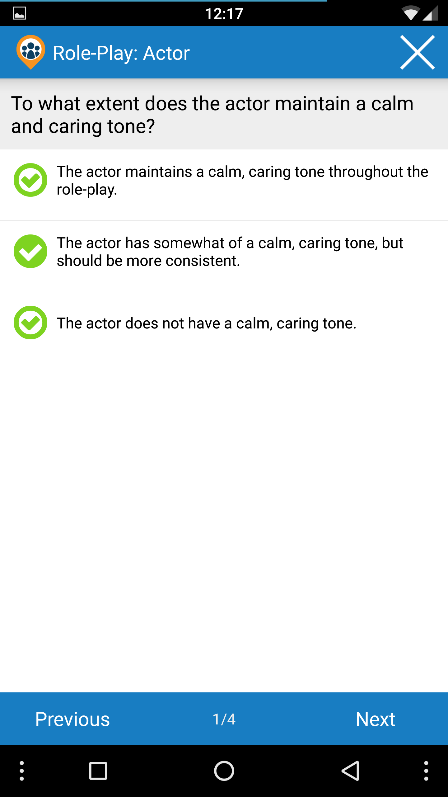
Here you can see what it looks like to observe someone else’s practice. Each practice question is presented as a carousel page, and a video player plays the video. At the end is essentially a quiz for the user to fill out what they thought of the practice. There is also an option to report a practice unreviewable. I set the unreviewable practice logic to reset the component, and kick the user back up to the Module Overview view. Like practice, Observe Practice was largely redone to use Native Xamarin by yours truly.

## Role Play

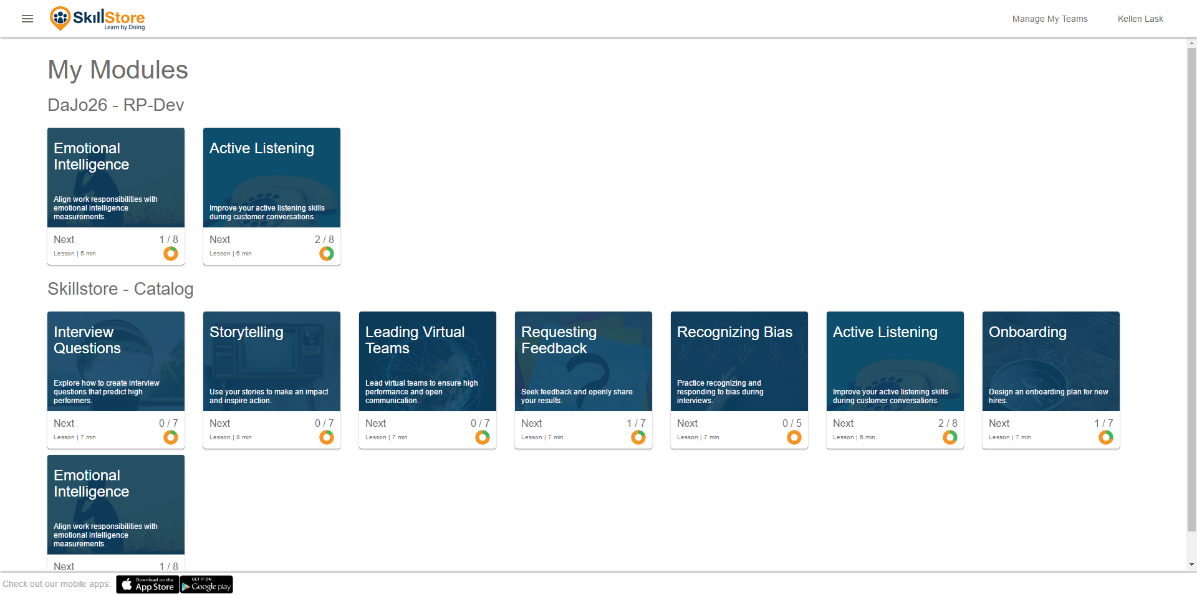
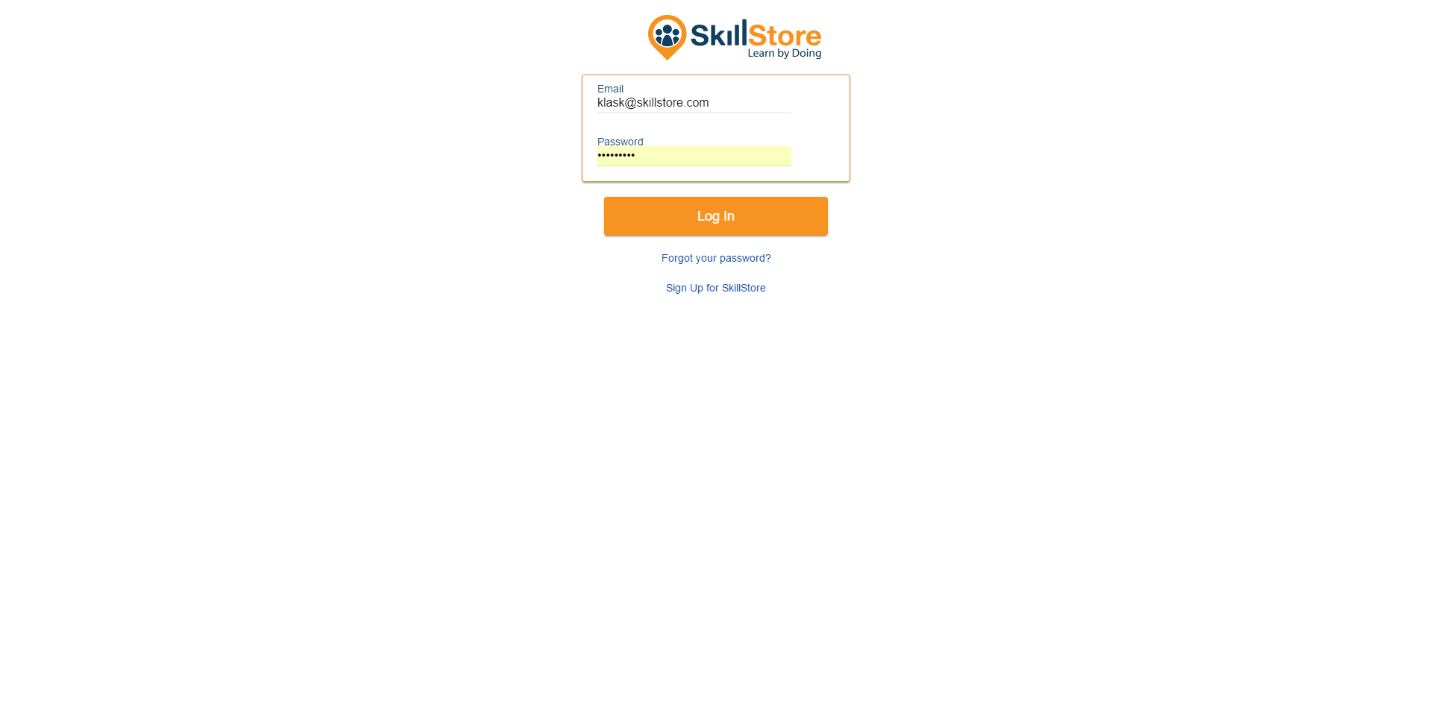
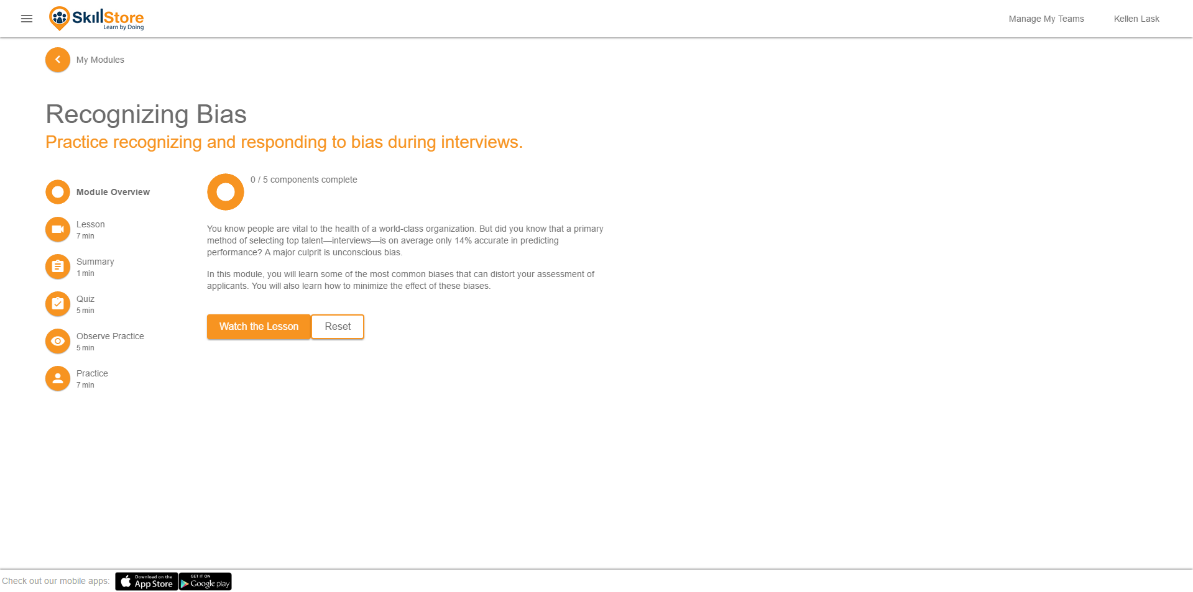
The other big attraction for Skillstore was interactive role play. The basic concept is that users in a module would be matched for role play (only once they’d said they were ready), and then they would enter into a video conference with another person. In the conference, they would act out the scenario presented in the component. This was not an area that I did much work in on the mobile app; however, I did several months’ work with the web app version of interactive role play.

## Observe Role Play

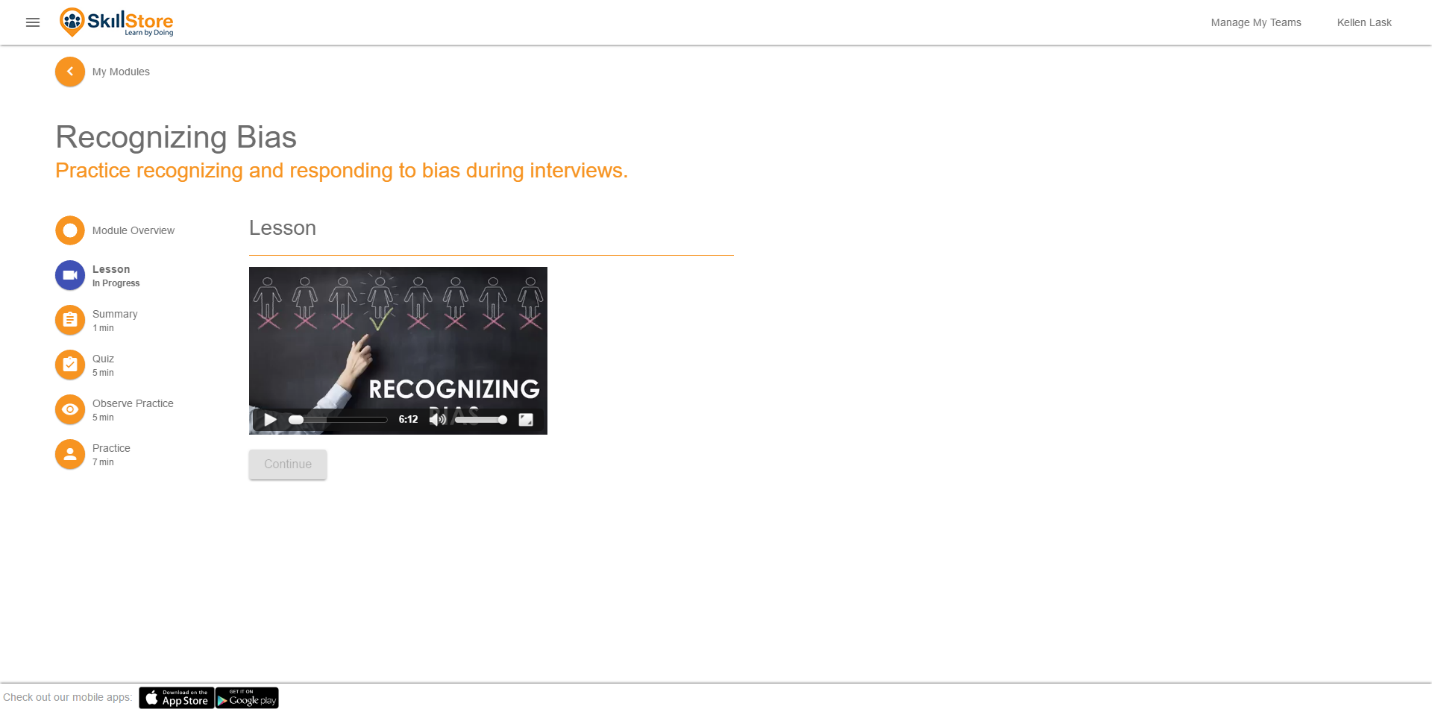
As with practice, users would review other role plays. The two separate video feeds would appear side-by-side in the video player, and then the user would fill out a little quiz afterward.

# Web App

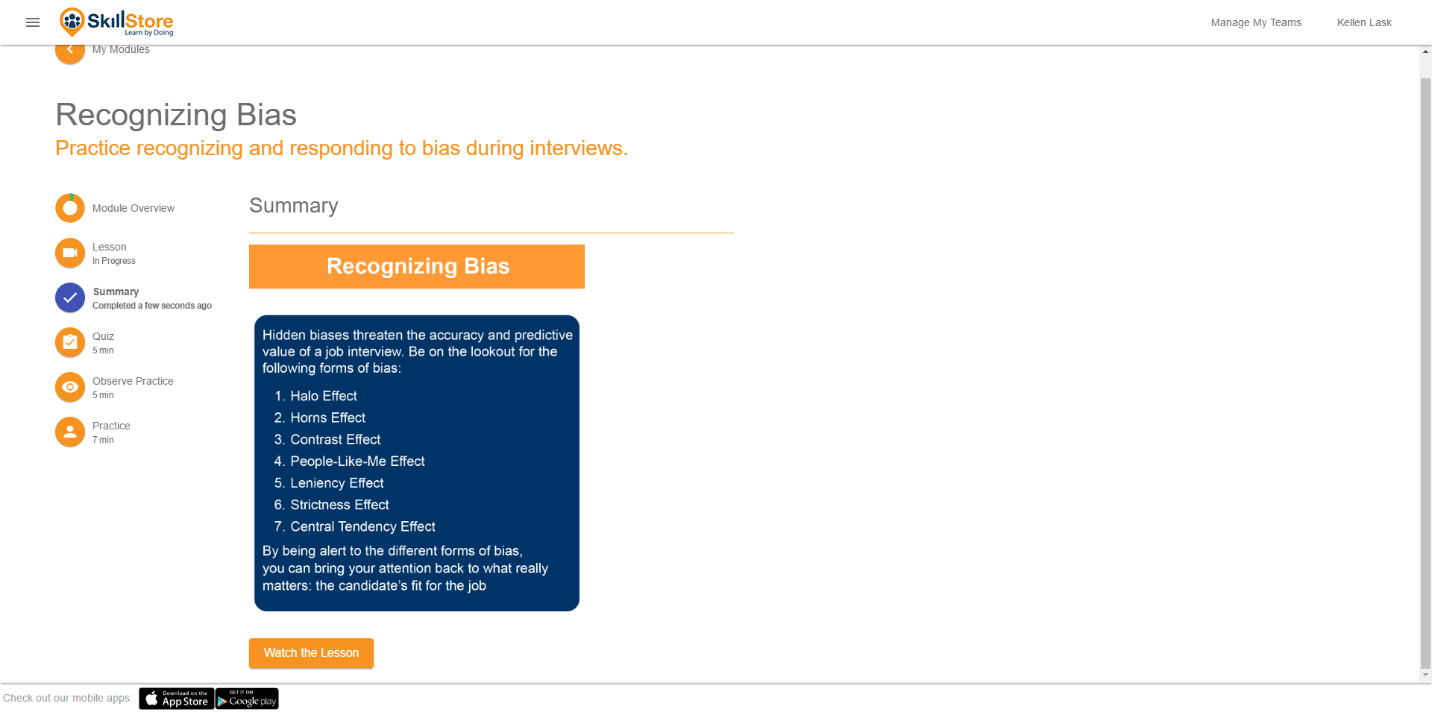
 

Above are a few screenshots of the web app. With the changes I was working on just before Skillstore went under, the web app reached parity with the mobile app. Now every feature in the mobile app was also in the web app.

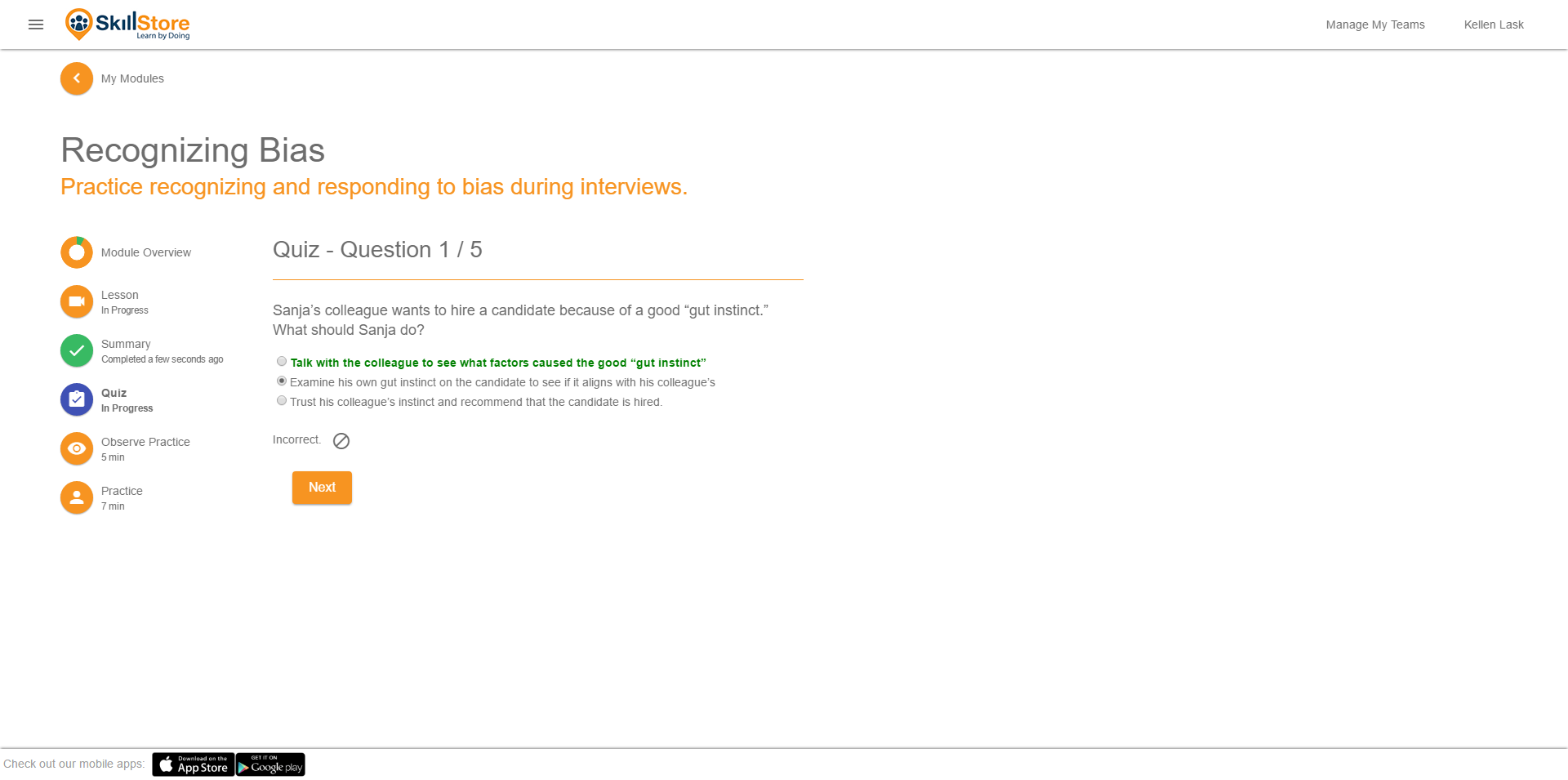
## Lesson

Lesson worked the same way as in the mobile app.

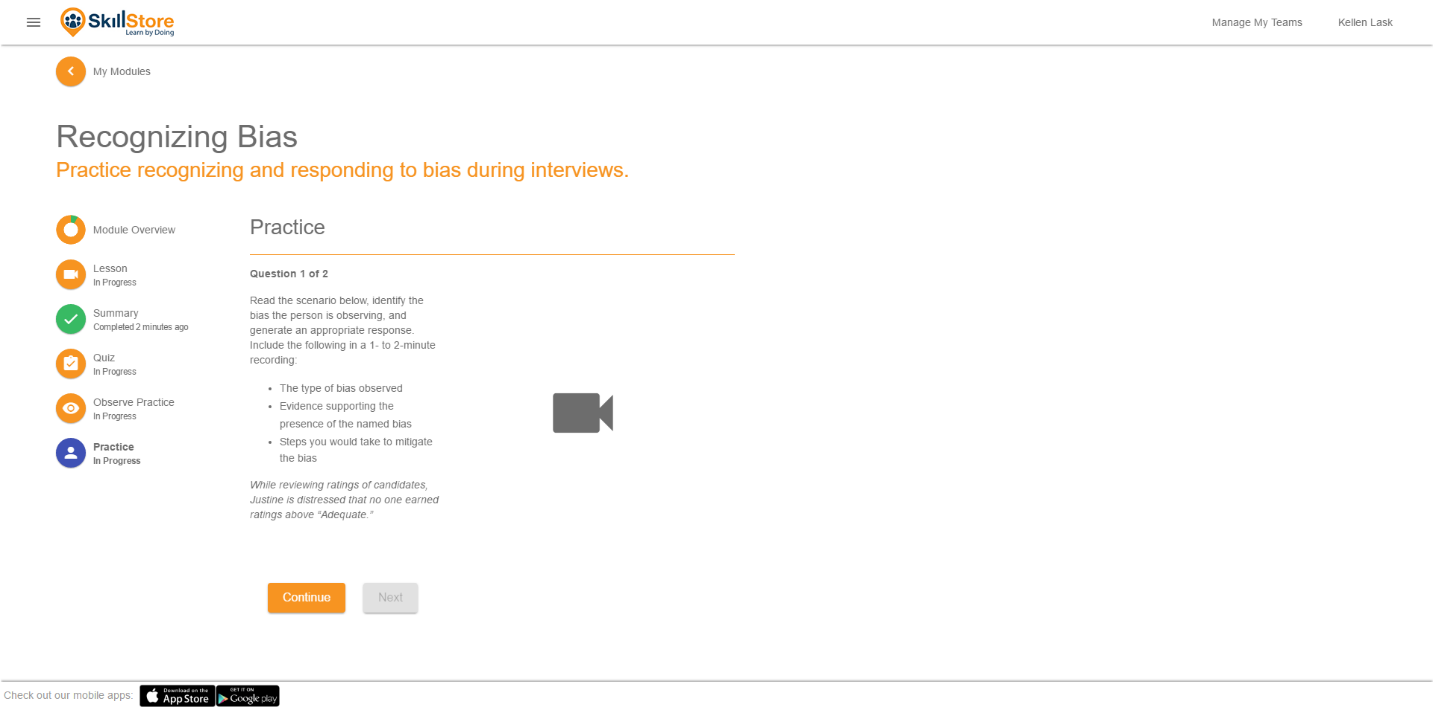
## Summary

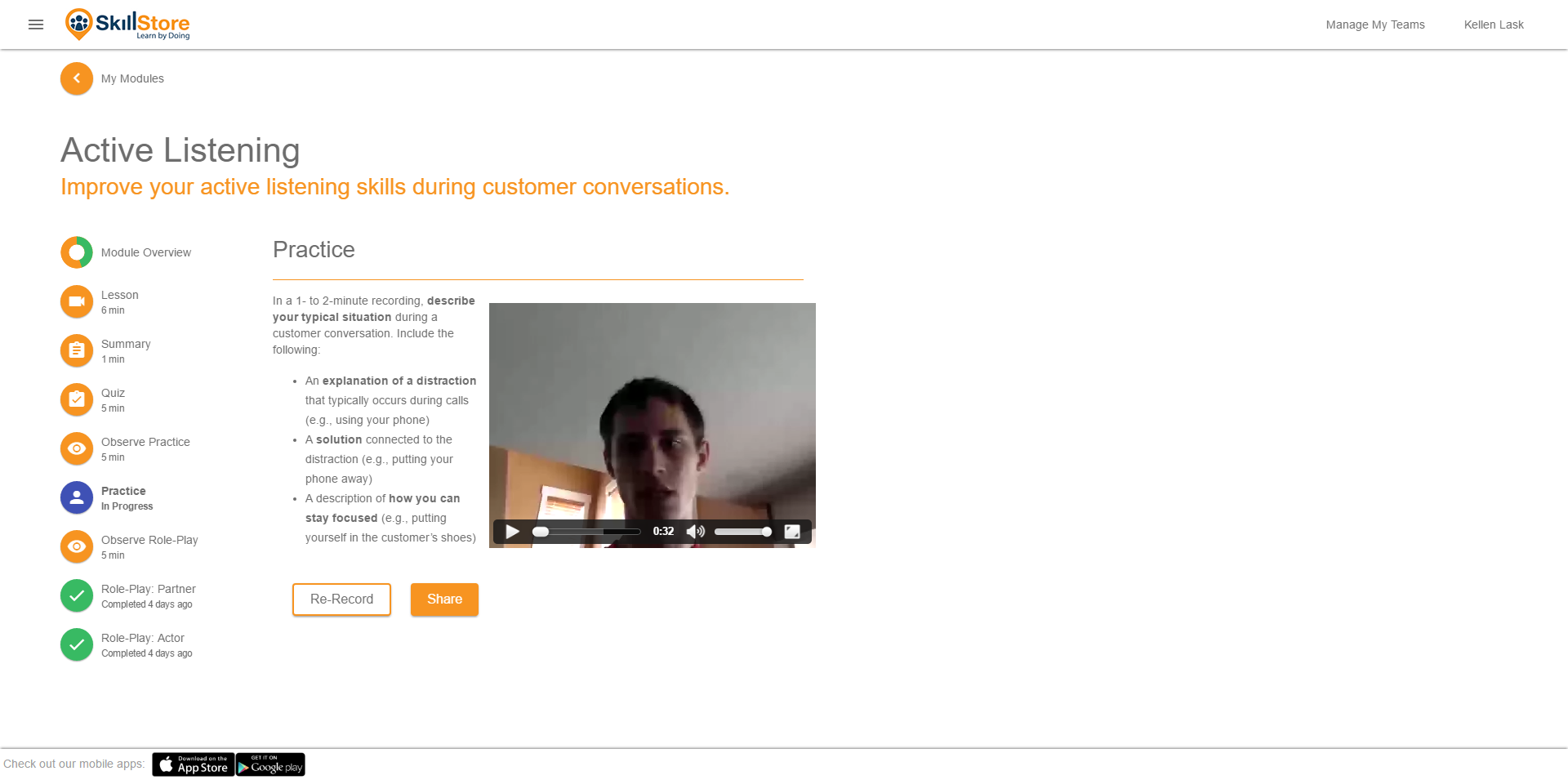
Summary was also the same as in the mobile app.

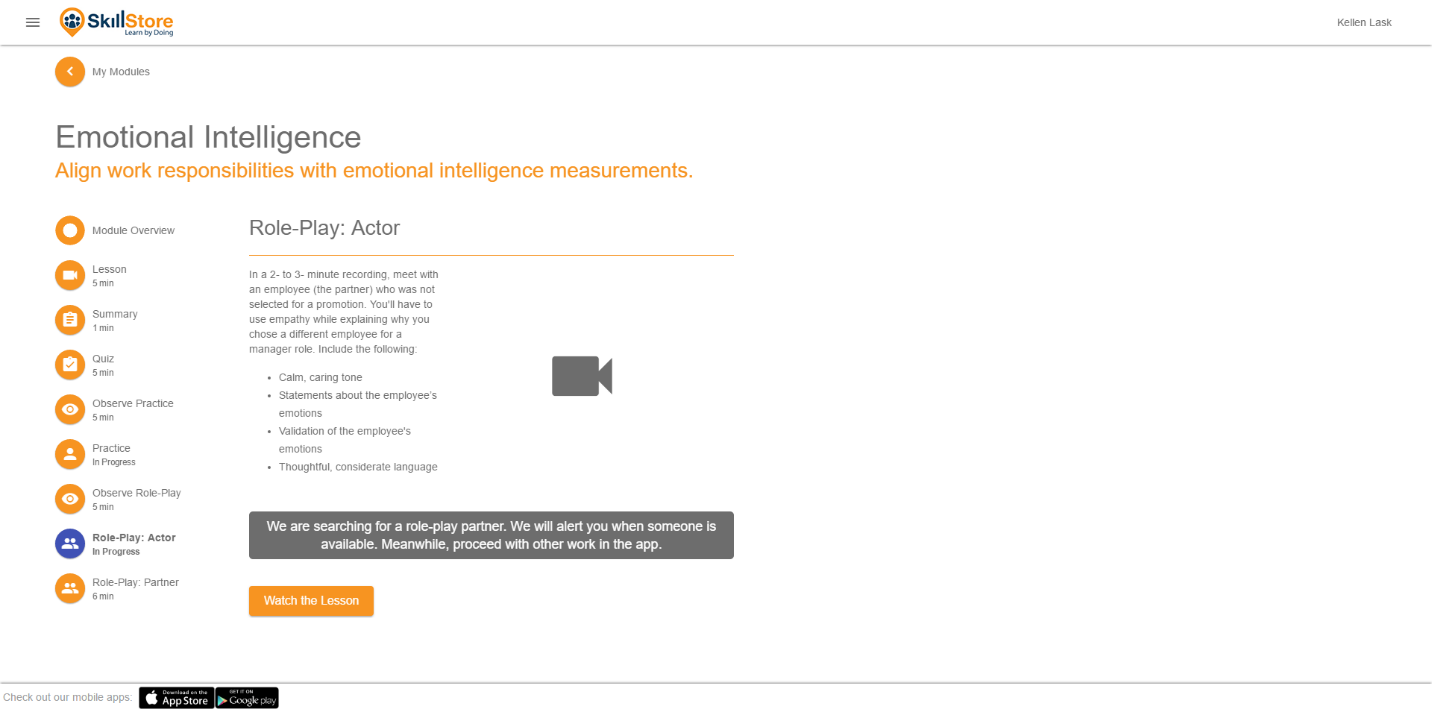
## Quiz

Quiz worked the same as in the mobile app. Interestingly, here was where a feature moved from web to mobile – the web app debuted the instant quiz feedback feature, which I later brought to the mobile app.

## Practice

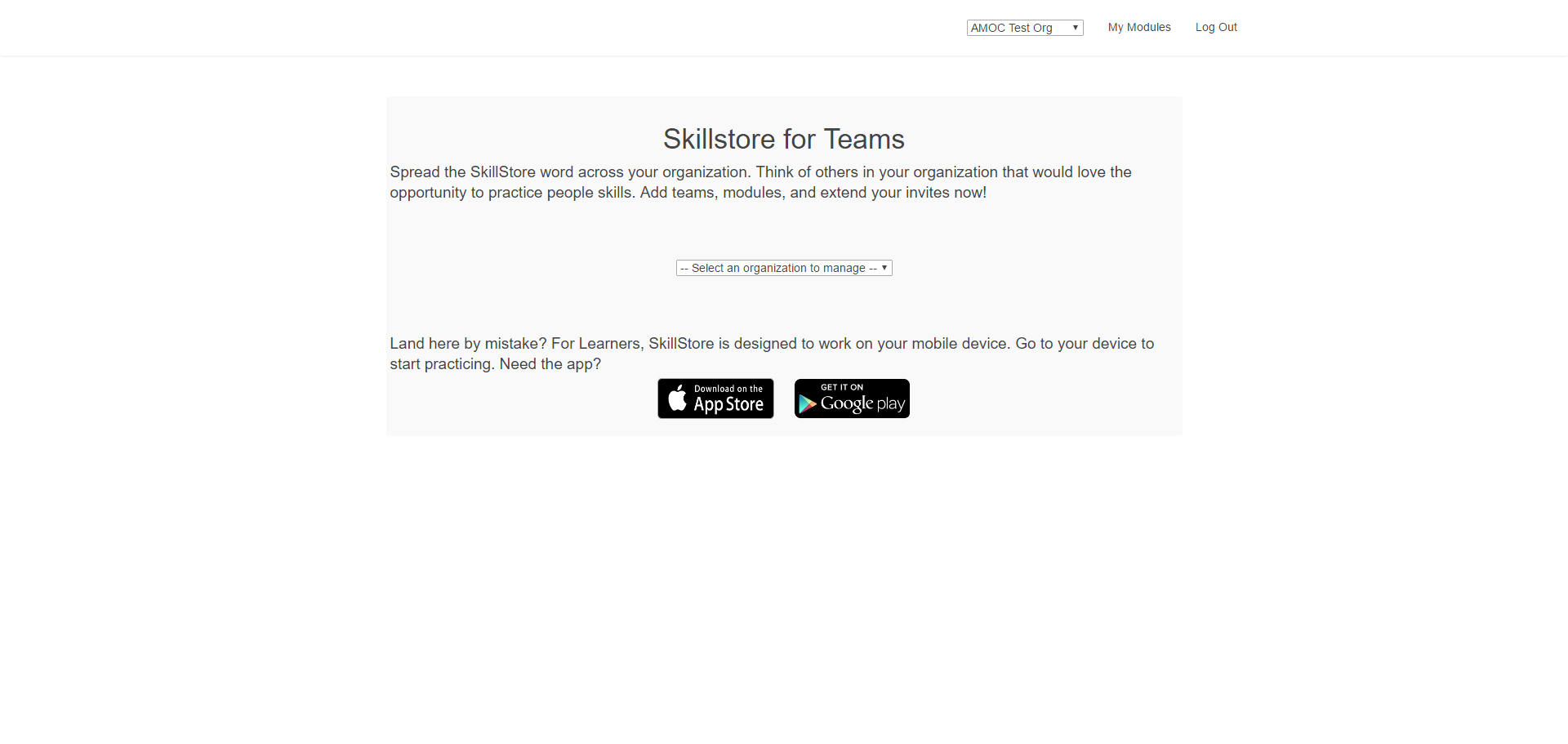
Practice in the web app was very similar to the mobile app version. Very recently, however, I completely re-architected the practice code to reuse new code written for the interactive role play features.

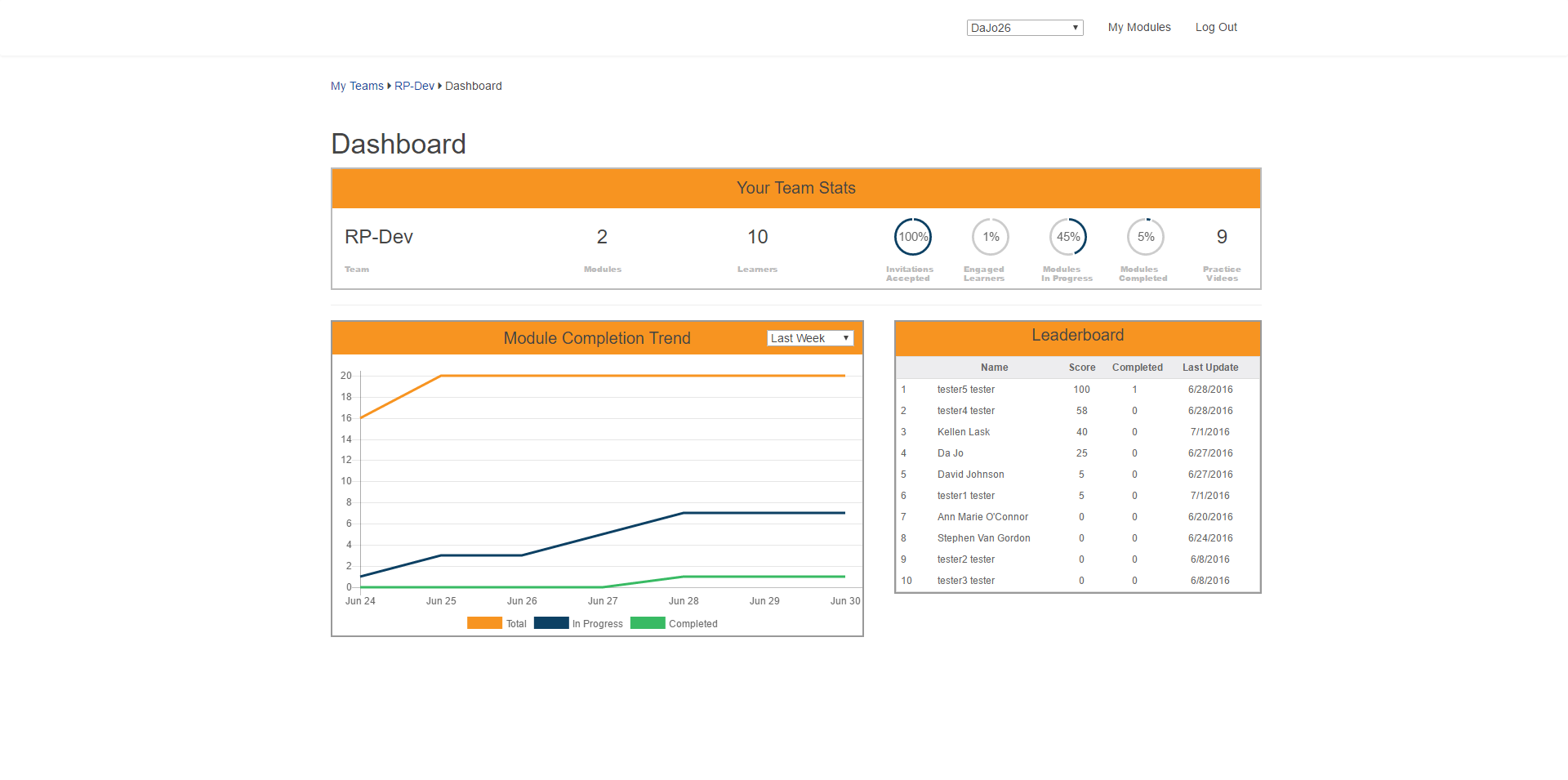
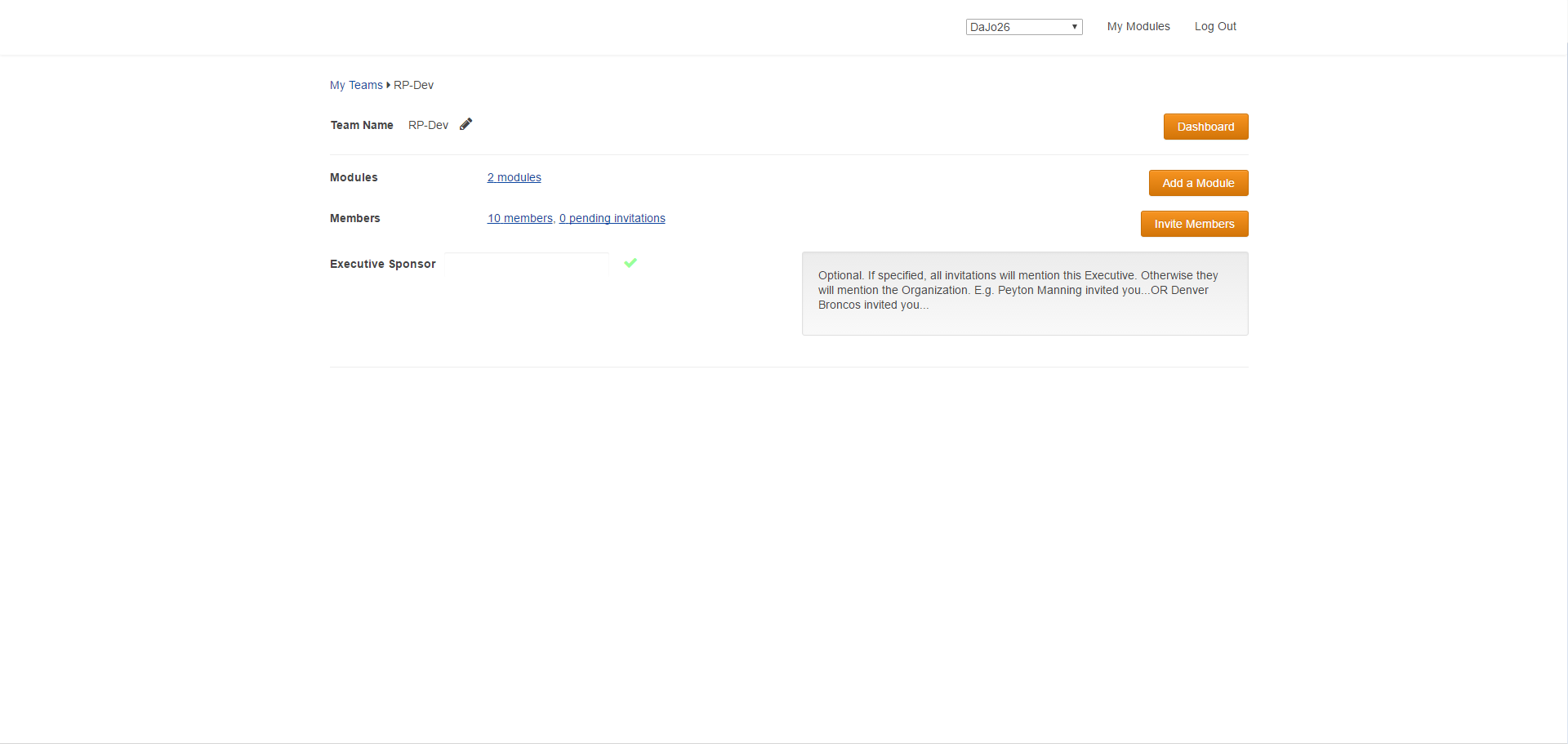
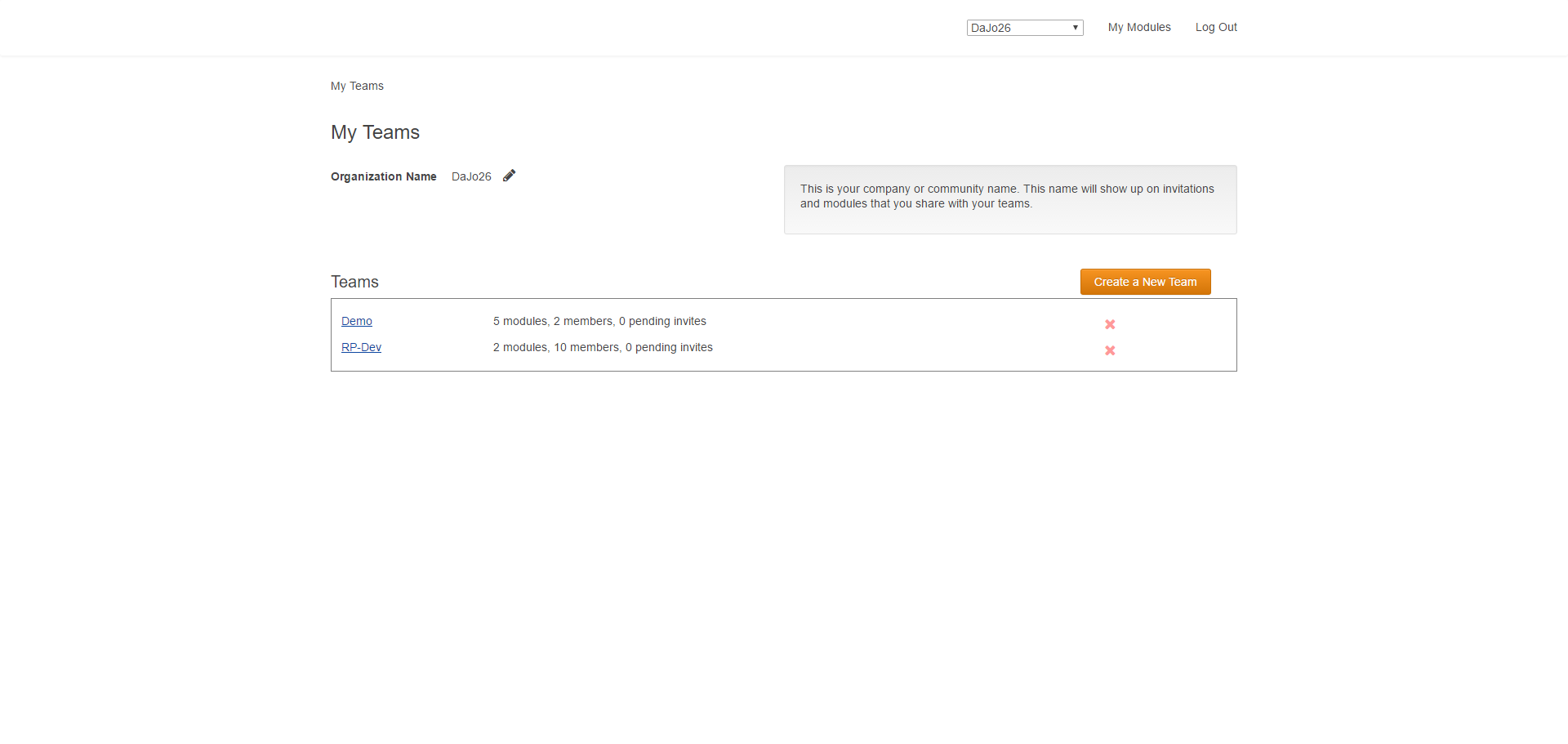
Here you can see playback post-recording. At the time of these screenshots, parts of the web app were already down as Skillstore was being disassembled. I unfortunately couldn’t get a screenshot of recording or role play.

Here you can see one of the role play screens. This is pre-match messaging. As role play didn’t work, I couldn’t get screenshots of the rest. Role play looked about the same as practice, with a second person’s feed included – the classic two box video conferencing look.

Role play was a lot of work to bring to fruition. It was many times more complicated than individual practice.

## Team Admin

Team Admin was made to help clients manage their teams of learners. It offered many reporting tools, and various other team-forming tools. Had Skillstore not shut down, Team Admin was next on the list for a big overhaul.



Team admin was the last thing I was working on before Skillstore decided to close down. The morning I found out, I was actually talking with a co-worker about how to make that top orange box in the dashboard look less awkward. Then, as 9:00am rolled around, we went to have stand up, and Srikant Vasan, the CEO, delivered the bad news. I then reset my mac, helped pack the office up, and left.

On the dashboard there, I created the Leaderboard, and totally refactored the Module Completion Trend chart. The Leaderboard, when given a large enough team, converted into a leader/laggard board. The Module Completion chart gave various filters, and displayed data points on mouse-over.