Testing Report

Coursierge

June 5th, 2018

Screen-Sharing Test:

Stream Publishing/Stream Subscribing Test:

First we required a test for Publishing, as we cannot subscribe to a stream without publishing one first.

This was tested by using Toasts and by instrumentation testing. To move into the streaming activity, it is required to create and connect to a tokbox session. Otherwise, the activity would generate an error Toast and remain on the Professor activity. Once on the activity, we prompt for permissions for camera and audio. If those pass and the camera/microphone are activated, then we are streaming properly.

Next, we need to test the Subscribing. To view a stream in the subscribing activity, it is required to create and connect to the session id generated by the published stream. Otherwise, the activity would generate a error Toast and remain on the Student activity. Once of the activity, the stream should generate a live stream of the published stream. This is checked by seeing if the frame layout is populated and audio is being produced.

Our main test was using a real android phone to publish the stream and the emulator view the stream. This allowed us to get a real world feel for how the app could be used and any bugs/crashes we could find.

Audio-Only Mode Test:

File: ProfSession.java
Method: onClickToggleVideo(View view)

Using the instrumentation testing, we could see if the video could be toggled by using the emulator. Once streaming, if the button cut the video and retained audio responses then we moved on to subscribing to said stream. This allowed us to test whether the viewer saw the video cut and could still hear the audio. If these two passed, then the method was functional.

Professor/Student Question Functions Test:

We initially needed to test if our database was correctly storing our objects containing Question and QuestionScores. To test this we added dummy variables into the database and verified it by looking at the database. We also stepped through each possible path of execution to validate that we built the functions correctly. All data at that time were correctly updating and being placed into the table. Perfect!

Once that was done, we needed to be able to populate a RecyclerView to display all these questions and question scores. We simply grabbed all these objects and had them automatically be populated into a RecyclerView. Verified that the expected output into the list was the same as the program output and all was done.

Next, we also needed to create an interactivity in this list of questions. So, we went with an OnClick interactivity. Where if a professor/student clicked on the question, they would be taken to a new screen displaying the contents. We ran the emulator and verified that it was indeed accepting the necessary input to open the new activity. Once that was verified, we made sure that all the choices were being populated correctly with its corresponding question. Once that was finished, we had a perfect interactivity for quizzing students and professor viewing questions.

Lastly, I wanted to be able to display all the question results from students as a bar graph so it is easier to view all responses. First, we needed to make sure that the graph was building correctly on the app. Simply, created a fake graph with some data and verified that everything was responsive. Next, we made sure all data was being populated correctly into this graph. Then voila we had a correctly working and functional graph of student responses.