CS 4720 - F17 - Final Project Proposal

Members: Tarun Saharya (ts4pe), Salman Guliwala (sg7va)
Device: Samsung Galaxy Note 8 (Android)
App Name: AskYourProfessor

Project Description:
Our app, AskYourProfessor, will provide an instant messaging service that enables college students to more effectively ask questions during lecture. Traditionally, a college student asks questions during lecture by raising their hand, waiting for some amount of time, directing every other student's attention to them, and then pausing the lecture to have the professor answer their questions.

This format has numerous flaws:

1. It is frustrating for the student. The student must raise their hand and keep it raised for a potentially frustrating amount of time so that they can be recognized by the professor, which might disinterest the student in asking that question and force them to give up on it.
2. It breaks the flow of the lecture. Asking spontaneous questions creates spontaneous interruptions for the professor, thereby distracting them and generally ruining the flow of the lecture, leading to a more disjointed and less engaging experience.
3. It discourages asking questions. In addition to students trying to compete to have their questions answered first in class, asking questions in general is seen as an inconvenience to other students, since doing so interrupts and pauses the lecture. Few students want to be "that person" by asking questions, and thus are less likely to do so.
4. It wastes time. Communicating a query verbally to a professor is often slower, harder to understand, and more error-prone than, for instance, communicating via reasonably well-written text messages.

AskYourProfessor would fix these problems by allowing a student to effortlessly, quickly, and anonymously type their query on their device and send it to the professor's phone, who would intermittently pause the lecture when they think it's convenient to view the query feed on their phone and answer them in a prioritized and rapid-pace fashion, while still keeping a steady train of thought, uninterrupted by spontaneous questions.

In more precise terms, we propose that:

1. Our system shall allow the professor to initiate a lecture session and thereby start receiving queries
2. Our system shall allow a student to search and join such (active) lecture sessions
3. Our system shall allow a student to type a query into an input field and submit that query to the professor

4. Our system shall allow the professor to read through a feed (list) of all submitted queries, with the feed updating as soon as new queries are received, just like a typical instant messaging chat window.
5. Our system shall allow sections to be added from the Lou's List API.

We plan to incorporate the following features:

- Camera/Image Gallery - users can take a picture of their face and set that picture as their profile picture and icon (which will be shown in the home activity screen)
- Firebase Cloud Services - messages are sent and stored on the Realtime Database, and they are retrieved by the professor's version of the app to display in a feed. When a professor adds a section to their section list (list of sections that they teach), the section data is also added to the remote database. Also, Firebase Authentication will handle sign ups and logins by both students and professors
- SQLite Database - used for writing section data - course name, instructor, location, etc. - to the phone for fast, local access
- Consume a pre-built web service - Used the Lou's List JSON API to pull course and section data for the purposes of adding these sections to the user list
- SharedPreferences to save particular user data like the profile image URL path and computing ID.

Wireframe Description:

Our wireframe shows the basic layout we expect in our app. On launch, you will be directed to the sign-in page. If an account does not exist, you have the ability to create one. During account creation, you will provide a university e-mail and password, and have the option of uploading a profile picture that will appear as your user icon. For the purposes of this app, the university email is a UVa email (e.g. computingID@virginia.edu) You may also choose the role of a student or instructor, although generally, the student or instructor status would be preset from another master database.
Upon login, you will see the list of class sections you have current access to. You may sign out from this page, and can add classes. We might also implement dropping classes, although that wouldn't be necessary considering you should have access to whichever classes you will have access to, similar to how Collab gives you access to each page of the classes you are enrolled in.
If you click on a section as a student, you will access the message feed where you can send questions for the instructor to view ONLY IF the section is active. As an instructor, if you click a section, the section will be marked as active, and the instructor will gain access to the message feed for the section.
Both roles may also add sections to the list of sections they currently have under their profile, which is retained through the local database on the phone.

Platform Justification - What are the benefits to the platform you chose?

Kotlin is very much compatible with Java libraries and capabilities. We also had more experience with Android as developers, and our personal devices happened to be Android as well. There isn't much more reason for us to have used Android over iOS for our particular app aside from these reasons.

Major Features/Screens - Include short descriptions of each (at least 3 of these)

SignInActivity - The app begins here on launch, and allows you to either sign in with an existing account or create an account if you have not already done so. If the account entered does not exist, the relevant error message (shown as an Android toast) will pop up saying your e-mail or password is incorrect.

SignUpActivity - Enter your university e-mail (computingID@virginia.edu), and a confirmed password to access the application. You must also specify whether you are a student or a professor. (For the purposes of this assignment, we are assuming that each user is only one or the other, and that all instructor users also have computing IDs) You also have the option of uploading an image to your profile to use as your profile icon. Pressing the account creation button will redirect you to the SignInActivity page. (Note that you are still not signed in yet)

HomeActivity - This is where you are directed after signing in. The top left displays your profile picture, and next to it is your computing ID and the number of sections you currently have under your profile. The list of sections in this activity is the list that you either currently attend (as a student) or teach (as an instructor). You can click on each to enter the section lecture session (where you can either start sending messages or receiving them).

SectionPickerActivity - Here you can specify a course by its department ID and course number (e.g. CS 4720, COMM 4710, etc.) to have the app access the Lou's List web service API to retrieve all the sections that are part of the course. The user may go on to select one of these sections and add it to their personal list of sections (the sections that they either attend or teach) and display all the sections underneath in a RecyclerView.

MessageFeedActivity - This is the activity that lets instructors view all the questions submitted by students for a particular section (the list of questions is a feed). Leaving this activity would mean ending the session and thus preventing students from submitting further questions.

SendMessageActivity - This is the activity that lets students submit questions or comments to instructors under the section they have currently selected. They can only enter this activity if the lecture session is currently active (which only the instructor can initiate).

Optional Features - Include specific directions on how to test/demo each feature and declare the exact set that adds up to ~60 pts
The set adding up to 60 pts is:

- Camera (15 pts)

    - On the sign up screen, you can tap the camera icon on the top to take a photo. Once you take a photo, the image will appear in the frame that the camera icon used to be.
    - Once you have signed up with a profile image, that image will appear in the top left corner of the HomeActivity screen, or the screen with the list of sections on it.
- Firebase (15 pts)
    - On the sign up screen, you can press the “sign up” button, which will create an account (assuming the input is valid) using Firebase. The computing ID and whether the user is a student or a professor is stored in the database.
    - On the sign in screen, pressing the “sign in” button will send an authentication request to the Firebase server, letting you know if you are properly authenticated or not (you can try signing up with an account and then signing in to make sure that you did in fact make an account).
    - If you are an instructor user, and if you add a section to your list, the section data (id, instructor, location, meeting time, etc.) will be pushed to the Realtime Database (if the section didn’t already exist).
    - If you are an instructor user, clicking on a section in the section list will take you to the MessageFeedActivity, thereby changing the “active” value of the section in the Firebase database to true. This essentially means the lecture has started and the instructor will now be accepting questions. When the instructor leaves the activity, the “active” value turns back into false. You can test whether or not a section session is active by having another phone (signed in as a student) try to go into that section (a student cannot enter a section that is inactive)
    - If you are a student user, you may add messages and send them in the SendMessageActivity. These messages are stored in the Firebase database, and the instructor’s activity will be monitoring that section’s list of messages. If a new message is added by a student, it will appear immediately in the instructor’s feed. You can test this by having 2 Android phones, one acting as the instructor and one as the student.
- Lou’s List API (10 pts)
    - In the SectionPickerActivity, you can enter the department code (e.g. CS or COMM) and the course number (e.g. 4720) to specify the course from which you want to pick sections. Once you click the submission button, you will immediately see a lists of sections on the bottom of the screen, each with information pulled from the Lou’s List web API.
- SQLite Database (20 pts)
    - Whenever a student or an instructor add a section from the SectionPickerActivity, that section is saved into the section list of that user. You can test the persistence of that section list by exiting out of the app and trying to enter back in (after signing in with the same account). Also, signing into another account shows a different list of sections (which means there is a unique database table for each user).

Testing Methodologies - What did you do to test the app?
There was a lot of back and forth with the database, as we checked whether things appeared in the database or not after a given action was implemented. We used TextViews and logs to determine whether certain calls (such as Lou's List) were extracting the proper information, and then parsed through that information to get the details relevant to the usage of our app. We ran the app alongside Firebase to make sure things were saving and changing as intended. Most of our testing methods were unit tests, so as we introduced newer functions into the application, such as adding sections to a user profile, we caught the results with visual feedback and log statements in areas where there was no visual feedback. Log statements were particularly helpful in our case because the bulk of the heavy lifting in this app was on the backend.

Usage - Include any special info we need to run the app (username/passwords, etc.)

1. You must enter a UVA-style email (i.e. address ending in @virginia.edu)
2. After you sign up, you must sign in to actually start using the app.
3. You must click the add button (+) on the top right corner of the action bar in the HomeActivity screen to go to the SectionPicker screen and add a section to your list.
4. To sign out, you need to click the "Sign out" button on the top right of the action toolbar in the HomeActivity
5. You cannot enter incorrect course information in the SectionPickerActivity (such as CS 1095)
6. It is recommended that you have another phone/simulator to properly test the simultaneous interaction between students and a professor (sending messages and then receiving them on the other phone)

Lessons Learned - What did you learn about mobile development through this process?

We learned a lot about managing expectations about how long it would take to implement seemingly simple features. Additionally, it was very beneficial for us to have a vision for the project through the wireframe, and it helped immensely to have a reference to what we would like to build in order to streamline the tasks we would need to accomplish.
There was also a lot of focus on usability, so even if our artistic elements were not the most noteworthy, the actual seamlessness of the app was a huge focus for us. We learned how ugly a lot of the default styles are in Android, and that allowed us to appreciate UX and graphic designers a lot more.
Additionally, we learned how important it was to focus on the most important features first rather than dwell too long on the minor features or aesthetics of the app, which allowed us to get relatively extensive amounts of work done on the side of core functionality.