Project Overview: A Summary of My Project.

My application is an app which makes Math Olympiad Questions more accessible, if it wasn’t accessible enough already. It essentially works like a forum, allowing users to upload questions, solve questions, et cetra.

Functionalities: A Showcase of My Application.

We first note that the whole application is in dark theme; noting that my target audience are students, who generally prefer dark theme. It also enables the application’s functionalities to be very clear, due to the simplicity of the application’s colour scheme. On top of this, I make the assumption that the user has a stable wifi connection; otherwise, the app’s functionalities would be locked behind the login page.

Splash Screen

When the app launches, the user is presented with a splash screen. While this is occurring, the app checks if the user is logged in (via offline storage). If so, the user is directed to the main application. Otherwise, the user is directed to the onboarding.



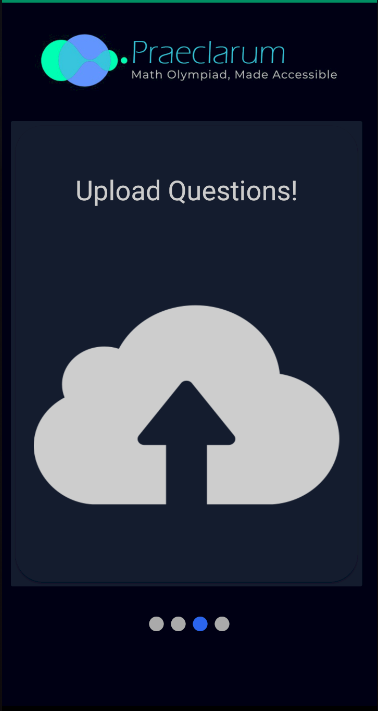
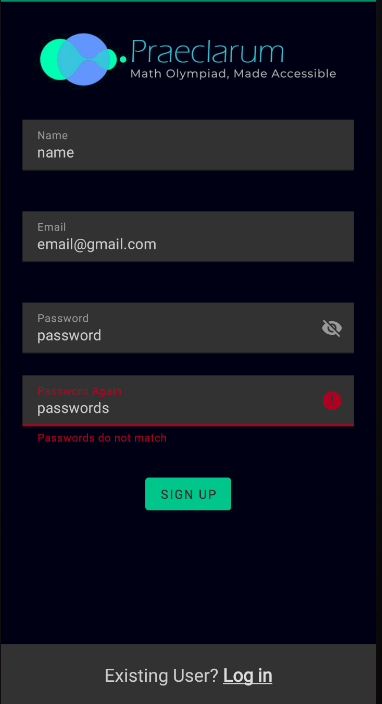
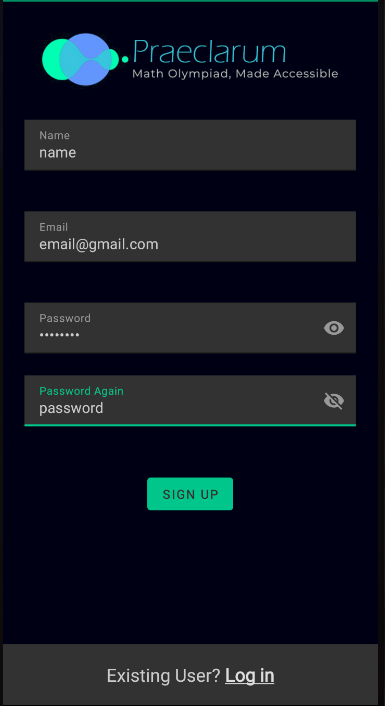
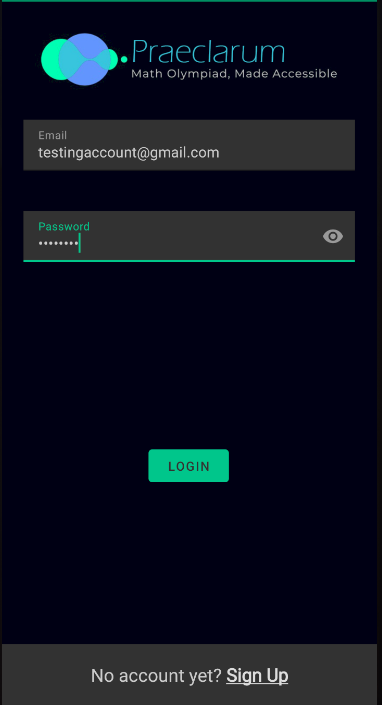
Image 1: The image displayed on the splash screen, designed from wix logo maker. No, WIX is not a sponsor for this application. I repeat, WIX is not a sponsor.

Onboarding

While designing the application, I felt that all new users should be introduced to the main functionalities of the application: what if multiple users are using the same device? As such, the login and signup functionalities are located in the last page of the onboarding.



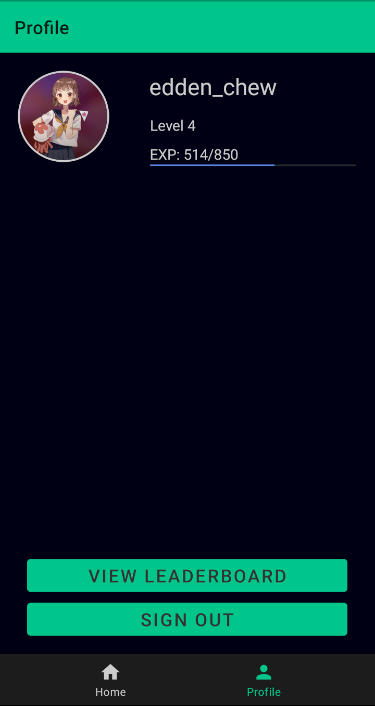
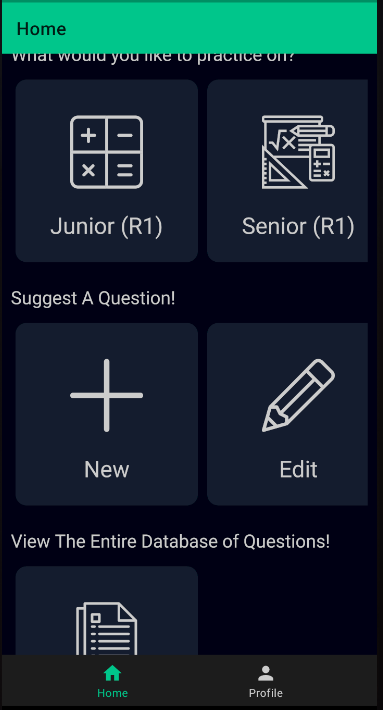
Image 2: The app’s top banner

Main Application

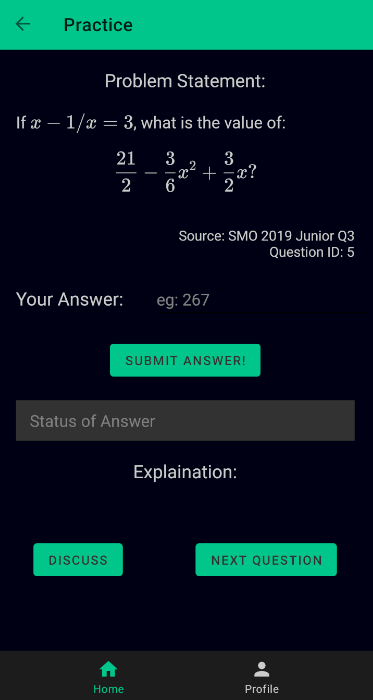
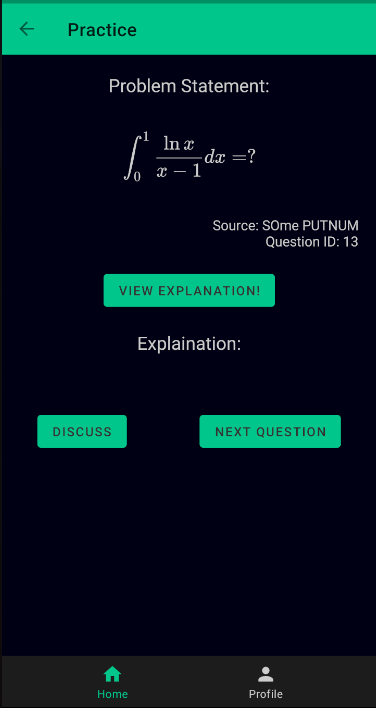
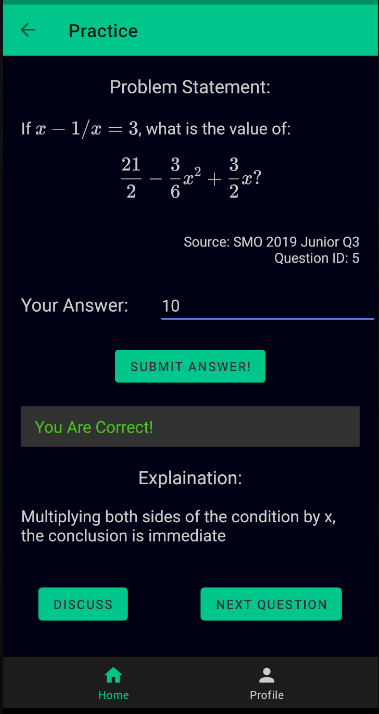
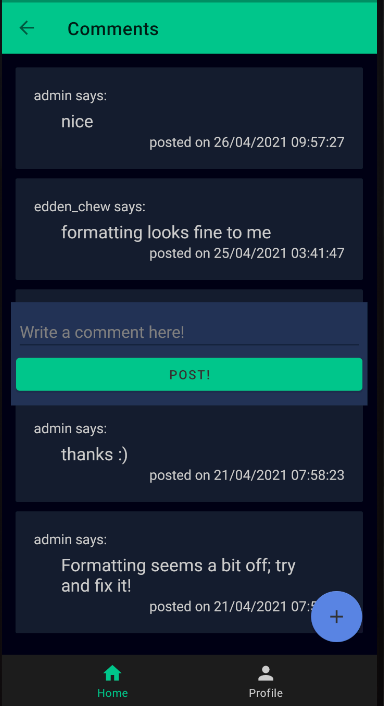
There are a few main functionalities here.

When the main activity is launched, the user is presented with a bottom navigation system, containing 2 main parts; the home fragment, and the profile fragment.



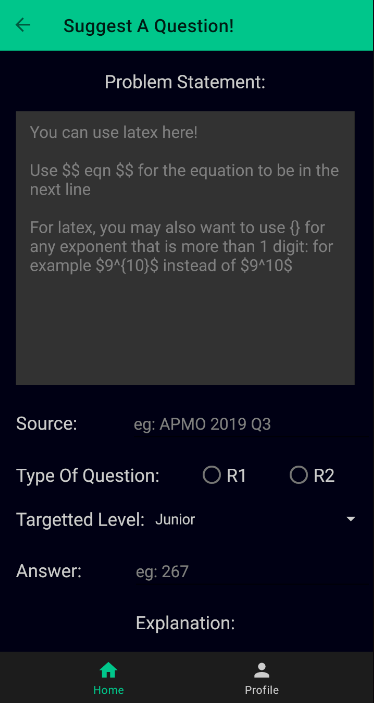
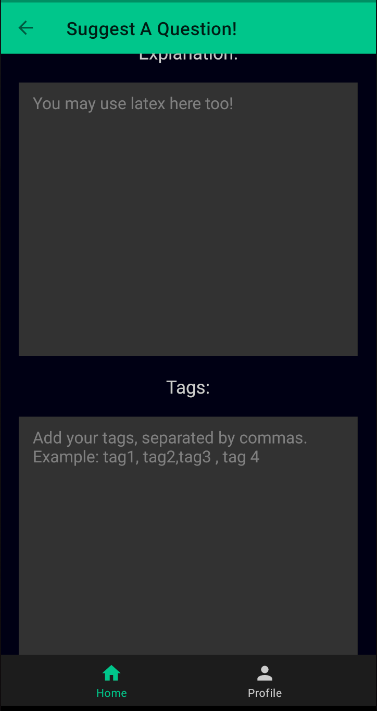
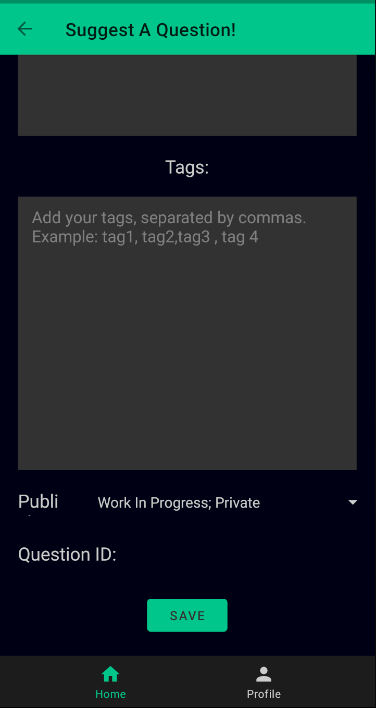
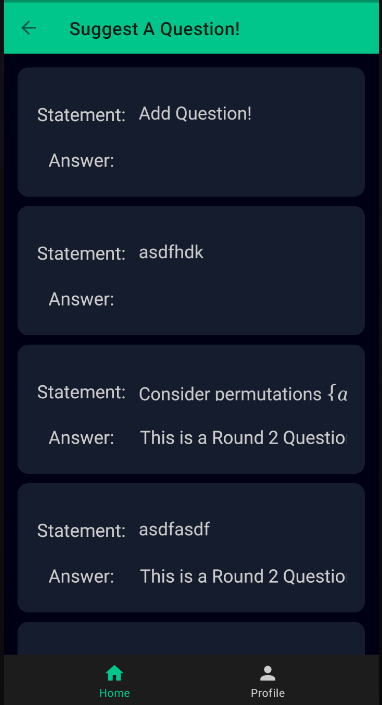
Home Fragment: Question Selection

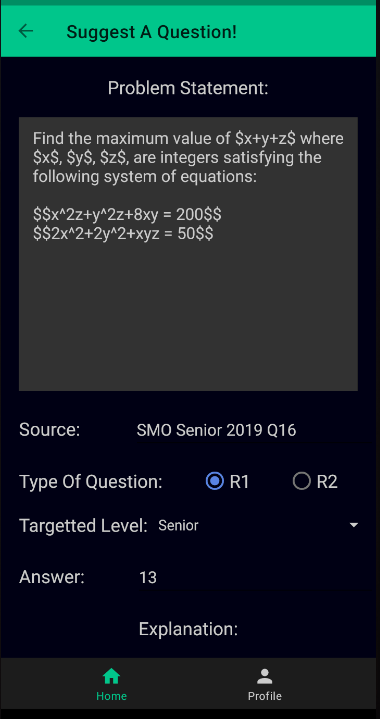
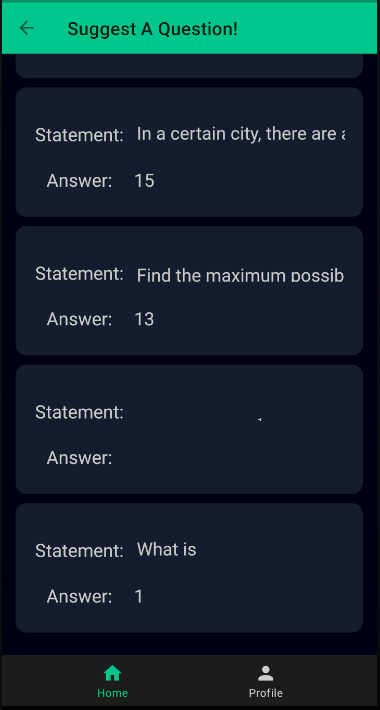
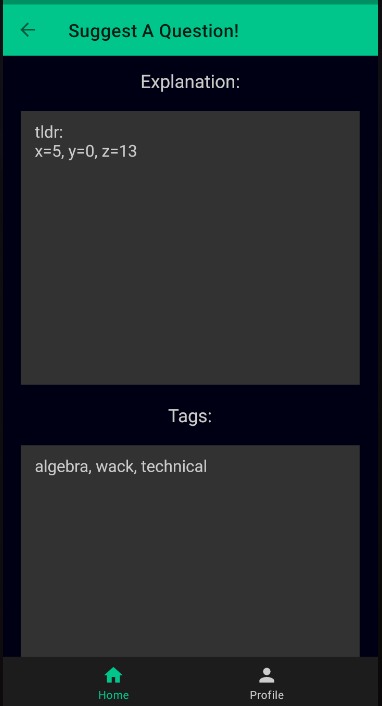
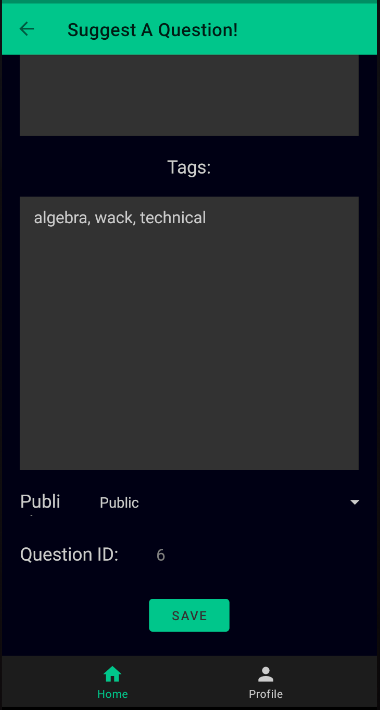
Each question has a certain difficulty assigned to it by the question’s creator. Then, by clicking on a difficulty, a question of that difficulty is loaded in by random (Priority is given to unfinished questions first, then to all questions of that difficulty if the user has already attempted all the questions). From here, the user can also discuss the question with other users. Note that animations are used between the navigation of fragments.

Home Fragment: Question Creation

Users can create new questions and edit previously created questions. Due to the long amount of time taken to load the questions, using any sorts of animations seems to break what the user can see.

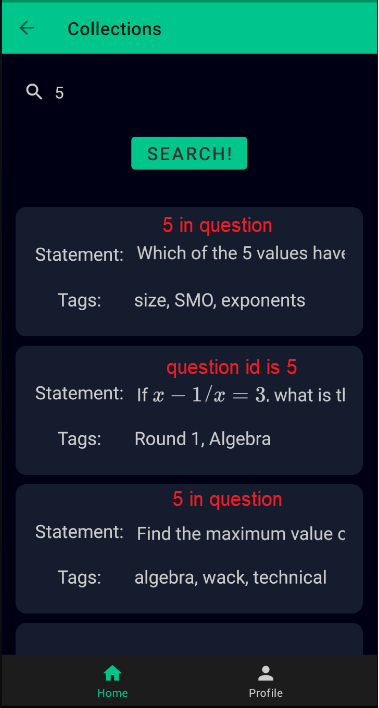
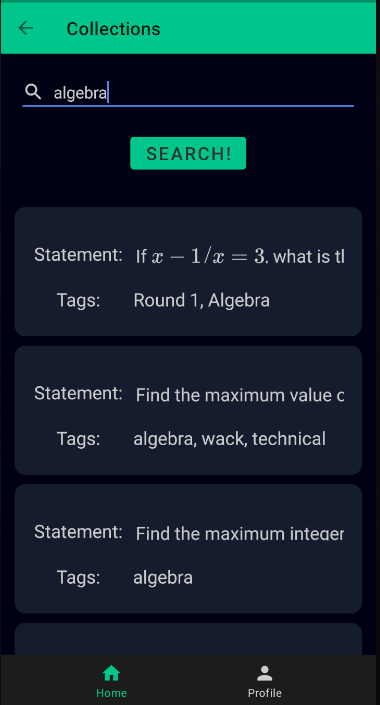
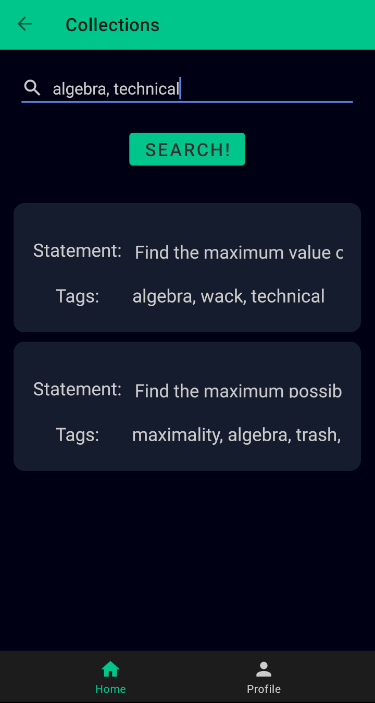
   

Home Fragment: Search Functionality

Users can search for questions in any one of three ways, namely:

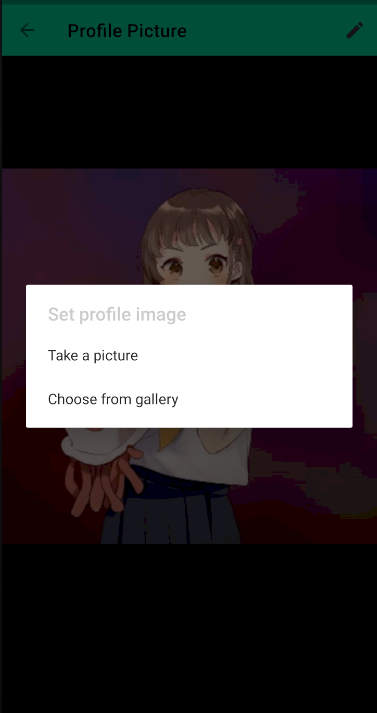
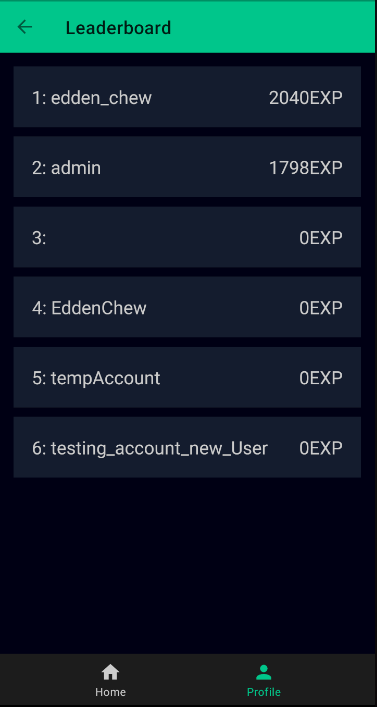
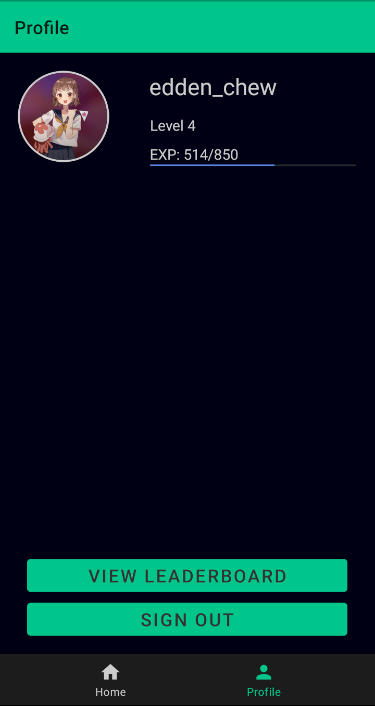
1. By Question (fuzzy wuzzy libary used, substring checked)
2. By Question id (exact)
3. By Tags (exact, separated by commas. Capitalisation is not important.)

Profile Fragment

There are a few functionalities in this fragment.

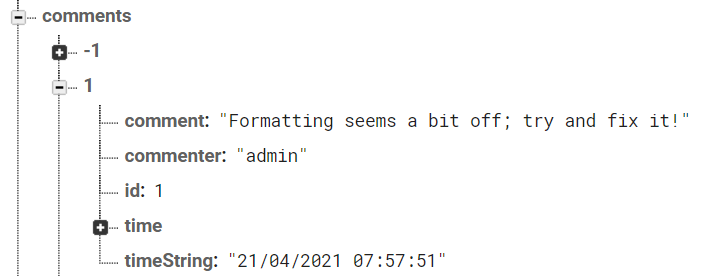
1. Viewing of leaderboard (sorted by total exp)
2. Changing of profile picture (Similar to my Assignment 2)
3. Signing out (redirects you to the onboarding)



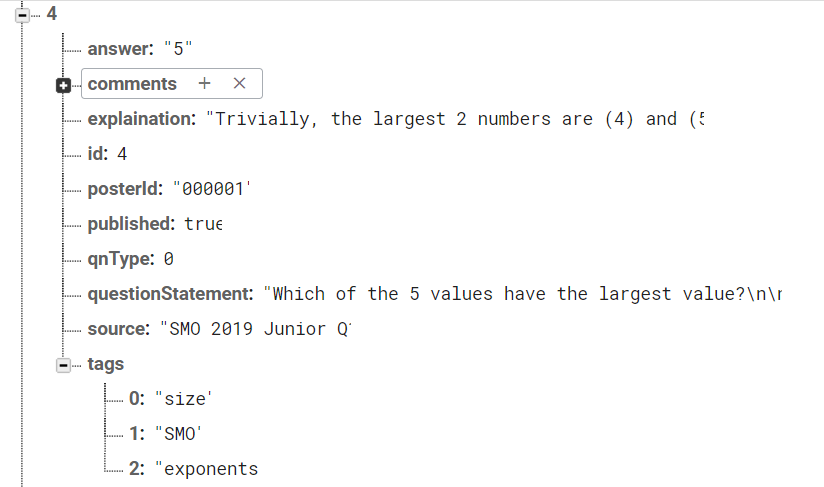
Code Explanation: The Deep Dive Into Firebase.

I will first cover the part of my project that is arguably the most confusing (due to undesirable coding practices and the nature of coding for async tasks: Firebase). Before we dive into the code however, we first have to understand the structure of the real time database.

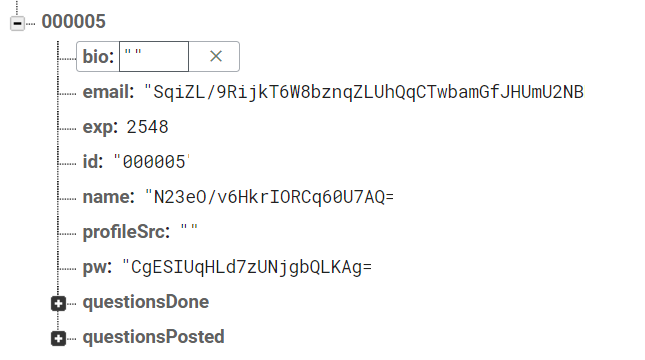
RTD Tree Structure



Comments are stored with the above data.



Posts are stored with the above data. Of note is the fact that the comments and tags are arraylists of strings, and not hash maps (which I found out while asking for peer reviews). This has a few implications, the most notable being that to add even a single element to the arraylist, I would have to retrieve the entire arraylist and add the single element. However, noting that this application was not meant to cater to many users, this should not be much of a problem. However, one notes that by storing the id of the comments, I have effectively flattened the tree structure, which is ideal.



Users are stored with the following data. Of note is the fact that the email, name, and password of the user are encrypted with AES, to ensure the security of the application. questionsDone and questionsPosted are arraylists storing the ids of the completed and uploaded questions (respectively)

FirebaseUtil.kt

Inside my application, all the firebase commands were settled via the object FirebaseUtil.kt. This settles all the needs of the app, for instance:

1. Retrieving user data on login
2. Checking that names and emails used in account reaction are unique
3. Account creation and Login
4. Comment adding and retrieval
5. Updating and retrieving completing questions and uploaded questions
6. Retrieving an question of a certain difficulty, at random
7. Retrieving questions from the search functionality
8. Retrieving all users.

On top of all of these, the application also allows users to change their profile picture. These images are uploaded to Firebase Storage, and retrieved upon login. This also explains why if one instantly taps into the profile fragment on the app launching, the profile picture will not update properly, and will instead show a default profile picture. This is also handled in FirebaseUtil.kt

Some Interesting Implementations.

* The EXP bar is a progress bar!
* The nav graph is way too large to be healthy!
* Every view had to have a text view at the bottom to prevent the bottom navigation from covering the bottommost content (especially on scrollviews)
* The latex library I used rendered latex via webviews so rendering the latex does not work half the time.

Usability Survey: Proof That My App Isn’t Completely Trash.

Pranaya: 5/5

The application is very functional, and the idea of the comment based system makes it very interesting. Similar to LeetCode, wherein users can share their new and fresh solutions, I think the application achieves all the purpose that it needs to achieve. The LaTeX addition and the tags and all this is very functional and not that many bugs were found either. I think this goes without saying but this application is way more functional that my own application, and I feel that it is very interesting how well-made this application is, and the firebase integration was very impressive.

Justin: 3.5/5

The app satisfies its functionality reasonably, with each screen being relevant to the project's requirements. The app also interoperates the content taught in the module sufficiently. However, some of the functionalities in the app can be ambiguous and distracting at times, notably the EXP system and the comments section, with the commenting functionality looking lacklustre. Finally, being able to revisit previously done questions would be ideal and should be implemented.

Jude: 4/5

(Context, this was before I implemented the Leader board System)

UI is minimalistic, functional, and beautiful. The premise and concept of this app is good and fulfils a niche for MO students. The XP functionality seems a bit off as there is nothing else tied to the XP function. Otherwise, the functionality of this app is focused and and seems to fulfil the purpose of this app, other than a few minor details.

Zane: 5/5

The app is very cool. It seems very practical for people who want to get good at SMO. The XP can be used to motivate the students to work on more questions and seems useful. It fulfils the purpose quite well.

Dway: 4.5/5

The latex is super cool.

James: 5/5

UI design is very intriguing with a nice colour scheme. UI is very intuitive, and the app is very easy to use. Very motivational for students to do MO. It fulfils the purpose quite well.

Eason: 5/5

Complex app design with lots of features but also rather well executed. Some statements are also not rendered properly in the overview page due to the Latex, perhaps showing the source + question number would be better. On top of this, a hash map could be used to store the data: AES is insecure and storing data after AES encryption is effectively the same as storing data as plaintext.

Reflection: Why My App Is Completely Trash.

Obstacles

I faced 2 main obstacles in this project, namely the sheer size of the project and the nature of Firebase.

Firstly, forum apps are not a small task at all. A lot of consideration has to be put into the creation of the app, specifically I took a lot of time to allow the user to be able to key in latex. On top of this, I actually wanted the user to be able to insert an image or use latex; however I was unable to think of an implementation that facilitated the answering of these questions, let alone the displaying of these images in the recylerviews. Despite these and many other functions (such as the changing of password) not being implemented, my whole project still incorporated a massive navigration graph as well as 30 layouts in total.

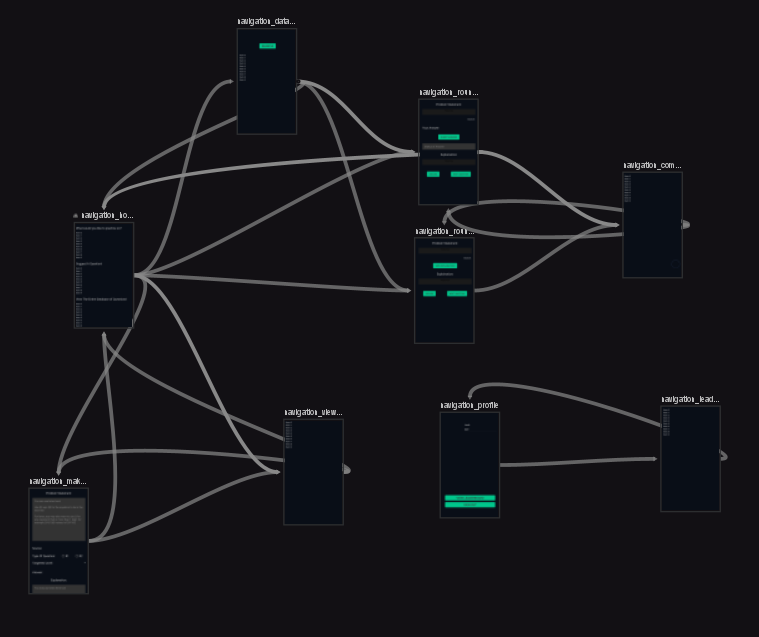


Image 3: Main application Nav Graph be like

On top of all of this, Firebase is by definition, an asynchronous task, all the retrieval of data is mostly done via observers instead of returning a function. This caused the difficulty of the entire project to increase effectively exponentially, and also caused all the code to be very messy. Luckily for me however, once I have retrieved the data snapshot, I no longer have to use async tasks to retrieve the data, and can just retrieve it as I would typically.

Learning Points

Through this project, I learned two things. Firstly, I gained a new appreciation for forum and game developers (notably, those that require the use of a database.) A lot of features which they have implemented seem unfathomable to me, let alone the completeness of their implementations. As an illustration, I wanted to implement a tag system similar to that in cytoid.io, where one can key in a tag which they want, and then select from pre-existing tags on the database. However, I simply could not think of an efficient way to do it, and ended up scrapping the idea.



Image 4: An illustration of the cytoid.io tag system

On top of this, I also learned the definition of good firebase coding by, ironically enough, using a real time database really inappropriately. The only good programming practice that I implemented was flattening my tree structure. Other than that, I did not use a hash map, but instead opted for Arraylists. On top of this, when retrieving data, I did not use event listeners, nor did I retrieve the data in packets (or in small chunks, to prevent lagging the device). Instead, I just retrieve all the data, which works as a proof of concept for smaller databases, but as the number of questions and users increase, my application will become more and more ineffective in retrieving the data.

Extensions to the Project

There are many extensions which I can apply to the project. For instance, I can allow the user to edit their comments, or even implement the aforementioned tag system. The small functionalities in a fully fledged forum, such as reddit, could also be implemented, allowing this project to have a lot more functions. Finally, I could actually try and implement data packaging, or possibly implement a latex library which allows latex to be rendered without the use of webviews.