ESE343 Final Project Report

Student Zach Doctor

ID:112204138

Student Jiale Chen

ID: 112295228

Student Jinxing Yin

ID 112263441

**Introduction**

As a wrapper for the whole semester ‘s study of mobile cloud computing, our group proposed to build up an application that could not only glue up what we had learned but also extend our perspective on other techniques used in mobile cloud computing. The motivation came from Zach in our group, he was planning a trip with his high school friends after his college graduation. He and one other person were planning the trip for a group of 8, however, he found it difficult to keep everyone in the loop while they were away at college. Group chats were made but they would get clogged up with random messages and his friends would keep asking questions that were already addressed in the chat. This app aims to eliminate those worries and keep an organized view of the whole trip. A polling feature that would allow people in the group to vote on things to do during the event, a group chat specific to each event, and a separate Itinerary for each event which allows the host to keep a detailed schedule of the whole event. That is the app our team envisions.

**Implementation**

As we researched online about some fresh user interfaces designed for applications, it turned out that we could design our interface with frame layouts for some of the view objects.

For us, it was really interested as all the projects we encountered before did not have a hierarchy for the layout. They were mostly just a single level so we had to squeeze all the views into a single layout. In comparison to this, the frame layout gave us the opportunity to settle some of the buttons/icons into different levels which gave us more space for the mainstream of the UI.

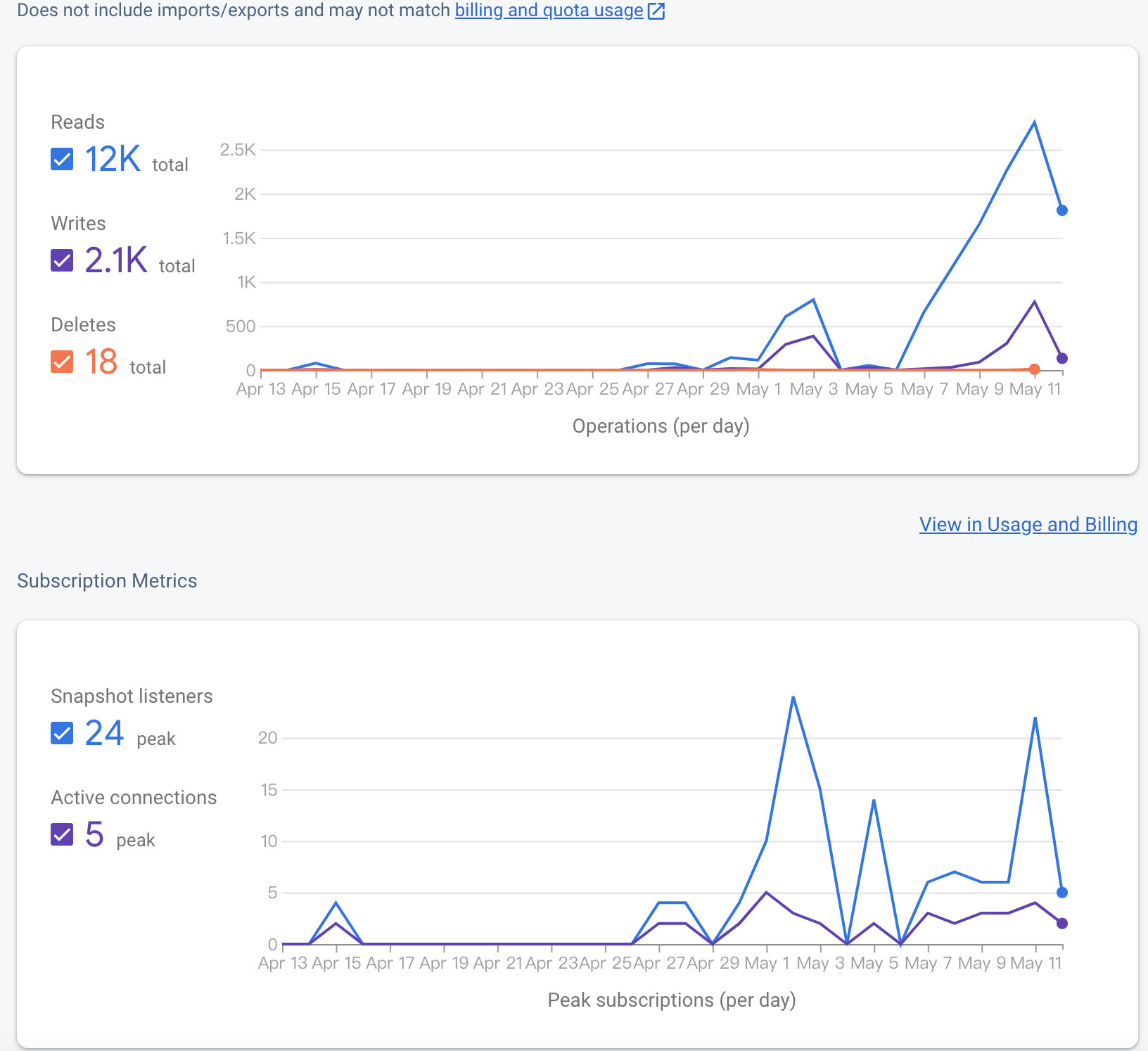
Meanwhile, we also utilized the recycle view instead of the traditional list view. We found out that, the recycler view had better animations and performances as we updated our user interface. However, we did encounter some obstacles in implementing our UI this way. We had a hard time properly working with the recycler view as the “onClickListener” method for this was not that straightforward. After some research, we overrode these functions so that they eventually got to work. These are some uniquenesses of our UI design.

For the design of the application activity, we split the works into different portions with respect to the different features of our application. By the way, we also exploited the preference manager to pass some data locally throughout activities. With the help of this implementation trick, we eliminated some unnecessary remote queries with Firebase FireStore.

For the chattings and creating group events we mostly utilized the Firebase Firestore database to interact with our data. There was massive data we needed to handle within some applications with chatting as one of the features. We had to manage every field required by the messages and the events and properly stored them in some places for later updating and querying. After looking up some tutorials on using the firebase fire store, we were able to correctly update the documents and queried them when we needed such as the documents for the latest conversations and the event list. Furthermore, as we mentioned before about implementing the recycler view for our UI, we were also required to bind our data with each of the view objects. We implicitly built some adapters for some of our models such as users and events. These adapters helped us integrate our objects with views with the help of the view holders. As a result, we had a more readable source code and less execution time in comparison to some traditional ways of implementing the view objects with the data-bound to them.

**Evaluation**

The app successfully runs to allow users to interact in group chats and one-on-one chats. The user's events and user information can be made and stored in Firebase Firestore. The itinerary is used to make plans for the time and the polling feature allows the users to decide on what they want to do and have a majority vote. The app's activities code is approximately 2420 lines of code long while the layout files total about 1750 lines long. The app storage and cache use 25.99MB with App size using 24.88MB, User data use 1.08MB, and Cache has shown to use 32.77kB. The app uses 1.43MB for Mobile data and WiFi mostly for foreground uses. The images below indicate the amount of reading and writing that is being done in the database and app along with the amount of listeners getting triggered while running the app.



**Conclusion**

Indeed, we did not meet all of our specifications as we planned at the beginning of the proposal. We aimed to keep working on this during our break. We should find a way to share the app with friends such that people can easily invite their friends to private group events. Also, we planned to integrate our application with other applications such that it could have more interesting features related to trip planning or any outdoor activities. We thought these are feasible by applying some pre-trained AI models to these design features of planning. Generally speaking, we successfully built our group project and extended our knowledge on mobile cloud computing topics. We are looking forward to improving our application and expect it to be published one day with all our efforts.