The following write up will discuss the results of this assignment in which we did "exploratory programming". The various tasks we were asked to accomplish is as follows:

- Authenticate users with Email sign-in, as well as Google
- Implement CRUD functionality
- Image uploading and hosting

Challenges faced when Authentication:

We did not realize how complicated authentication was before we started implementing it. At first we planned to get a lot of information from the user in order to provide a more personalized user experience; however, quickly we realized that the more personal information about a user we store, the less secure the app is. In addition to this having all this information was not crucial to the functionality of our app. Thus **we made the decision to only authenticate using email and password**.

Challenges faced during CRUD Implementation:

The biggest challenge we faced while implementing CRUD was figuring out how to appropriately nest data items without nesting too much. Originally we planned to nest our data as follows:

User

- User name
  - Own
    - Lipstick
      - Brand
      - Name
      - Color
      - Price
      - Finish
  - Wish List
    - Lipstick
      - Brand
      - Name
      - Color
      - Price
      - Finish

However according to Firebase specifications:

"Avoid nesting data

Because the Firebase Realtime Database allows nesting data up to 32 levels deep, you might be tempted to think that this should be the default structure. However, when you fetch data at a location in your database, you also retrieve all of its child nodes. In addition, when you grant someone read or write access at a node in your database, you also grant them access to all data under that node. Therefore, in practice, it's best to keep your data structure as flat as possible."

Because of this we decided to restructure our database to try and improve speed and decrease complexity of implementation. Our database structure is now as follows:

- Username
    - Lipstick
        - Brand
        - Name
        - Color
        - Price
        - Own
- Username
    - Lipstick
        - Brand
        - Name
        - Color
        - Price
        - Own

This new structure takes us from 5 nested levels to 3 thus simplifying our code a lot.

Another challenge we ran into was updating entries in our database. As part of our implementation, we decided to assign unique keys to each lipstick in the format name_brand. The problem with this was when users update this, we were having issues deleting the old lipstick entry. At first we implemented a solution where if users could edit the name and brand we would delete the first entry and create a new one, giving the user the impression of "editing" – however after some discussion we decided it would not make sense for users to edit a name or brand of a lipstick, since it didn't make sense to change it like that so we made those immutable. This reduced our code length and thus total file size by quite a bit.

We tried to optimize our app the best we could for speed; however, since we are still in the exploratory/learning phase we did not have too much time to really implement all the different ways in which we could make our app faster.

We hope to encode images as strings for the next part in order to speed up the download process in the future as well as place image file size constraints that users can upload.