What went well

* Once our app had access to users’ information, it was relatively easy to request and receive that information.
* Using the Facebook and Twitter SDKs for Flutter made the process of signing in easy and understandable.
* Implementing Firebase backend with Flutter was almost seamless and provides our app with a lot of functionality.
* Creating and finalizing a mock UI before coding, saved time and was easier to quickly make changes.
* The Firebase Cloud Firestore NoSQL database is easy to work with.

What didn’t go well

* Figuring out how Facebook/Instagram API works because their docs are spread all over the place and Instagram API is being deprecated.
* Converting UI from Adobe XD to Flutter Code, details put in XD were lost and not ended up being useful in final implementation.

What could be improved

* The structure and design of the Post and related classes could be improved to be more cohesive and user-friendly, since they were created during original testing of API requests.
* How our user access tokens are stored needs to be discussed to ensure user privacy.
* The structure and code for the UI pages. There is a lot of repetition of objects and styling code, maybe condensed into a different file/ reused. The appbar and few pages like the map page could use work with formatting.
* Additional rules could be imposed on the backend to restrict the ability of end users to modify the database.

Challenges

* Once we reach the stage of app review we will have to be diligent in making sure our app adheres to the terms and conditions set out by the social media companies.
* The Twitter API doesn’t allow you to choose which fields you want returned and just sends all of it, most of which we don’t need for our app currently. It also has many subcategories and arrays returned in the response, so parsing the correct data needed out of the long, complicated response was time consuming and difficult.
* Twitter also has separate responses for simple 140 character tweets, extended 280 character tweets, retweets, and quote tweets so making sure we covered each case when creating the classes took a lot of planning and testing.
* Enforcing unique usernames across the platform and ensuring that each user has the minimum amount of database access necessary to use the platform will need to be handled carefully.