Tocca – Technical assessment

Roderick Burkhardt

1. Coding question:

Develop an API that accesses a Firebase real-time database to extract and return groups of 50 records from the document *user_profile*
user_profile structure is the following:

```
User_profile: {
   $user_id: { misc. profile information }
}
```

The API must be designed in such a way that it returns the profiles in a JSON object that contains no more than 50 profiles at one time. Many requests to this API can be performed to download all the profiles in the database

Technologies to be used: Javascript, Firebase function and or any other tools from the FIrebase or GCloud environments. Preferred framework: Express - if proposing to use a different one or not to use this one, please provide the reason why.

https://github.com/r-burkhardt/tocca-technical-assessment

2. How would you secure the above API?

I would see the use of token validating process used to the secure the API. The token can be passed in the header from the site requesting data from the API, and should a valid token not be passed the request would be rejected and an error message returned. This token could be through a login process on a website/app, that would then be store local for the site to use when requesting data access. The API would than take the token compare it to the tokens it has issues at logins and determine if the token exists and if it is unexpired.

When a valid token is received, the API will then complete the user request and return the appropriate data. The request and response could also be limited according to the user privileges granted to the user of the token.

3. Architecture question:

Design the architecture of an API or set of APIs to collect customer information from a CRM (let's say SalesForce) and aggregate this information into a Firestore database. Note: the proposed architecture must take into account the fact that the application is used in a multitenant environment.

CRMs are often made up of complex data tables, combining multiple field into what may seem as a single input, ex: firstName, lastName, middleName are often collapsed together into a single field called Name. This can make aggregation of data more difficult handle.

Step on is to determine what data might be more beneficial it have in firestone, and evaluate what data might me more of a security risk in one than the other. It would be best to place things such as the a user id, name(first, middle, last), and ids of the user's data stored in the CRM. Since Firestore is more useful for quick access, I would store minimal data sets of information in Firestore, with references to larger data set in the CRM.

I would go as far as to determine what users that are more frequent consumers of the the API and aggregate larger amounts of their data to Firestone. With this data stored in a faster service, it will lessen the latency of frequent users, as a user become less frequent their data can be monitor and removed from Firestore to make room for new more frequent users that would be moving to Firestore for quicker access.

New and update data that would added to Firestore would be merged into the CRM as it is added and updated so that when a user is removed from firestore there would no need to push the data to the CRM. Also when the data is being cloned to the CRM from Firstore it can be setup in a way so as to not affect the user's experience. This data cloning can be initiated upon completion of the user's event on Firestorm