Mobile Hackathon Project 0:

Android/iOS Patient Tracker App

We propose to develop an Android and iOS application to aid in dispensary patient data management and viewing. The system is aimed to help doctors to enter as well as view patient history as well as other patient details. Our system is a standalone system that can be installed on doctor Android or iOS phone to be used for further login. On installation the application allows a doctor to open application and enter the details of any patient that used his service. The application allows doctor to insert various data fields regarding a patient including patient name, disease, medication provided, date of arrival, cost etc. The system saves this patient related data in the mobile phone or web service or realtime DB (see below stages below). The doctor may now view this data as and when needed. The doctor may check the details whenever needed. The application allows doctor to search patients by name as well as appointment date.

Modules:

- **Patient entry:** Doctor may enter details of any patient with related data.
- **Entry saving:** The system saves this entry in its mobile data base for later viewing.
- **History viewing:** The system allows doctor to view patient history as and when needed.
- **Search by name:** The application allows doctor to search any patient by name.
- **Search by date:** The application allows doctor to search patients by appointment date.

App Development in Four Stages:

To fulfill the requirements of this project the developer will basically deliver four apps, which we are calling stages with different data architectures:

Stage I: Save and access data in Local Mobile Phone. The disadvantage is that doctor will not only be able to view the data in another device.

Stage II: Save and access data by using REST API. The REST API will be developed using Node.js, Express and MongoDB and deployed on Heroku as a Express App and MongoDB on Atlas. The developer will design and document the API using Swagger and build the API code manually. The disadvantage of this kind of backend is that it is not infinitely scalable, and the server is always on for which you will be charged.

References:
https://expressjs.com/
https://www.mongodb.com/cloud/atlas
https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction
https://swagger.io/

Stage III: Save and access data by using Serverless REST API. We will port the REST API developed in Stage II to Firebase Cloud Functions. In this stage we will continue to use MongoDB Atlas service as our database. This means we are basically just changing the middle tier from Express to Firebase Cloud Functions. This architecture is infinitely scalable and you are only charged for what server time you use, with a very large free tier.
References:
https://firebase.google.com/docs/functions/

Stage IV: Save and access data by using a Real-time database. We will not be using the API's but directly accessing data from the client. You are required to use the Firebase real-time database. You are also required to write security rules to secure your data. We will test the real-time capabilities of your app by running two instances of the app simultaneously on two different devices.
References:
https://firebase.google.com/docs/database/
https://firebase.googleblog.com/2014/04/best-practices-arrays-in-firebase.html
https://firebase.google.com/docs/database/security/
https://firebase.googleblog.com/2016/01/the-beginners-guide-to-react-native-and_84.html

Development Stack Requirements:

You can develop the mobile app in any of the following technologies: React Native, Ionic 2, Native Android using Java or Native iOS using Swift. Those who are developing the app in Native Android using Java and iOS using Swift are not required to do stage II.

Advantages:

- The system can be used anytime and from anywhere by the doctor.
- It excludes the use of paper entries/registers.
- Doctors can view patient whenever needed in their application.
- Saves time and reduces human intervention.
- The system is flexible and secured to be used.

Disadvantages:

- Every doctor must have an Android or iOS device for using the system.
- Data cannot be viewed in another app or device (Only Stage I).
- Does not contain dispensary stock details.

- Is a standalone system (Only Stage I).