

Week 9 Blog

Sree Ram Boyapati	a1775690@student.adelaide.edu.au
Course	Comp Sci 7092 - Mobile and Wireless Systems
Group	CovidGuard-F
Supervisor	Zach Wang

What have I done this week?

In the previous week, I have mostly worked on implementation of Upload and Download of TEKs from the server and the related work at the backend of the application.

The week's work has validated our concerns regarding the choice of server infrastructure as it provided us all the functionality regarding the scalability of the app.

The server architecture now takes multiple stakeholders into account like health authorities and separates user's interaction with backend as verification server to register the app and exposure notification to upload/download teks.

Challenges faced in this week

TimeZone Aware Datetimes

In Australia, There are 3 main time zones and the timestamps that we use have to be aware of this for deletion of TEKs from the server and compare the timestamps from various regions.

I chose to work with UTC based timestamp for comparison of timezone aware timestamps and delete the teks based on the ENIN of the TEK.

Idempotent Upload of TEKs

Firestore is still a database in making and there are several features that lack for example, unique indexes on columns. To test the Apps and also reduce storage space, Duplicates of TEKs should be erased from the backend.

All individual columns are indexed for easy read access in FireStore. So I chose to make TEK as the ID of the document in Exposure Notification servers. However, TEKs are base 64 encoded and the firestore id does not accept '/' character in Id. So I have urlencoded the ID and stored TEK as a separate column to have the best of both worlds at the cost of increased space.

Bug Fixes

Every time we download the TEKs, We store them in the datastore and truncate the previous copy. Truncate was buggy when implemented in a work manager because we returned the work manager as Failure if there are no TEKs to be downloaded. This bug was later fixed. Network Connectivity checks are failing when testing using a mobile device. We have side-loaded the app without an internet connection and the app notification failed.

Deletion of the Tasks

Due to the support of idempotent TAN uploads, We have chosen ENIN as the interval in the ENS server to base our deletion of TEKs. ENIN format can change if the RPI Interval is changed. So, client-side and server-side has to operate the same RPI Interval that we have chosen. For migrations, users can also upload the RPI Interval and TEK Interval to support various protocols and matching with various protocols.

Implementations

Request TAN and Upload TEKs

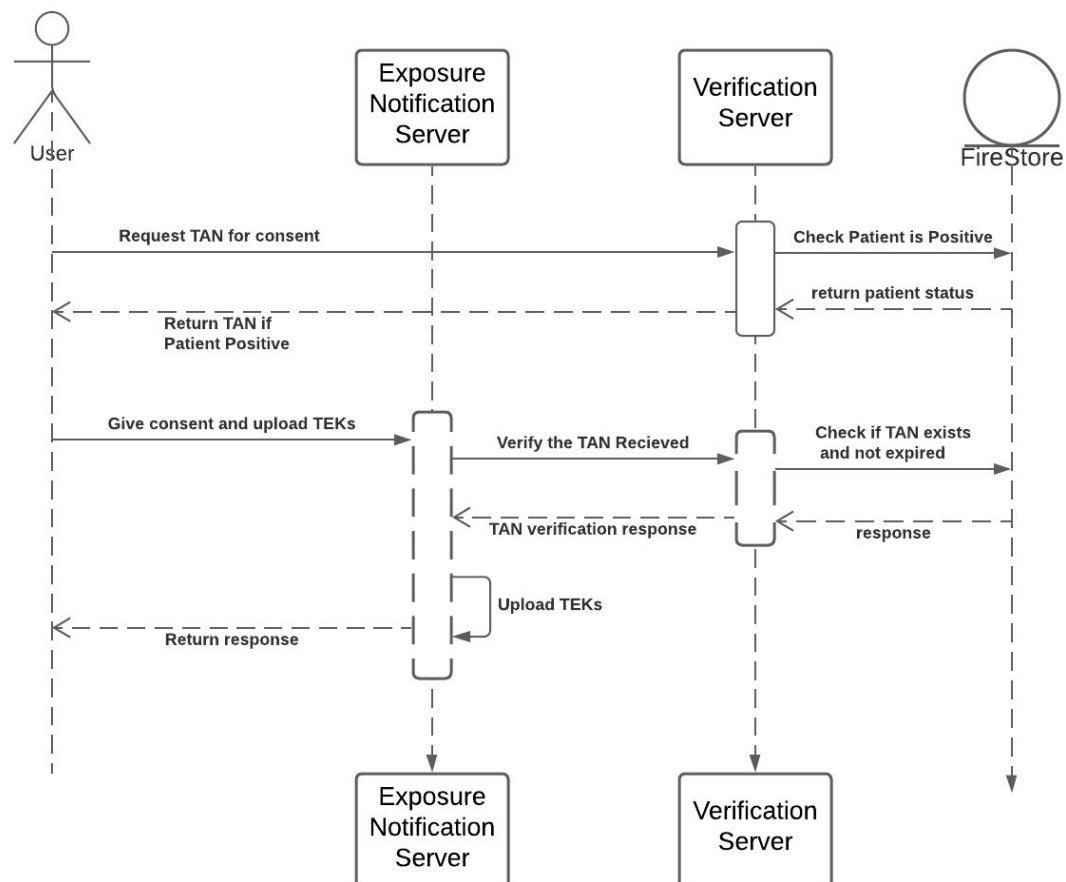


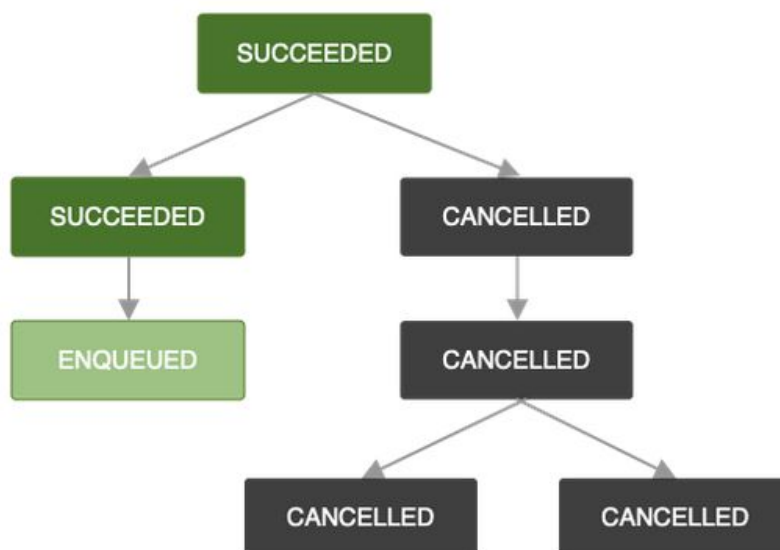
Figure 1: Request TAN and Upload TEKs

1. User requests for TAN to upload TEKs to the verification server. Verification Server checks if the user has been diagnosed as covid-19 positive patient. On Positive Result, Server sends a TAN to the user.
2. User bundles his teks and TAN and sends a request to the exposure notification server
3. Exposure notification server accepts the payload if the TAN is verified.

Client Side Implementation

I have leveraged Android WorkManager ¹ to make this process smooth and Intuitive. It supports chaining of different work tasks and support to pass input from one task to another. It allows us to listen to WorkInfo to act on the result.

1. Users can request TAN and manually give the TAN to upload TEKs for the first time. Next time, The TAN workflow can be chained to upload the tasks.
 2. Download of TEKs to Regeneration of RPIs to matching of TEKs can be third different tasks chained together.
- WorkManager supports complex DAG patterns for chaining tasks in the diagram below and also allow us to set constraints like only download on Wifi or only download on Wifi.



¹ "Getting started with WorkManager | Android Developers." 19 Aug. 2020, <https://developer.android.com/topic/libraries/architecture/workmanager/basics>. Accessed 11 Oct. 2020.

New Backend Architecture

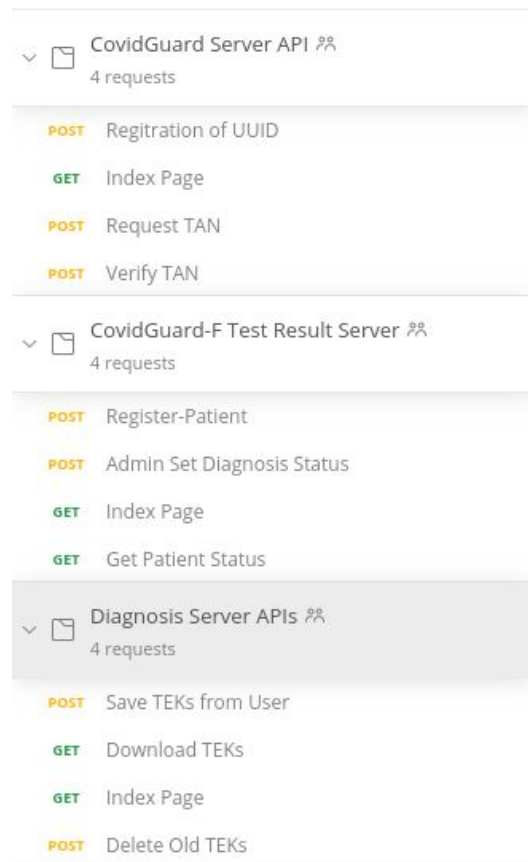


Figure 2: List of All APIs supported on the backend and documented using PostMan to share with Teammates

I have implemented the backend server APIs of LIS Server this week, Download and Upload of TEKs on the ENS Server and Request and Verification of TANs on the verification server. Each server is backed by its own firestore database as per the microservices paradigm. Each server is hosted on a different google project and separate logging pipelines.

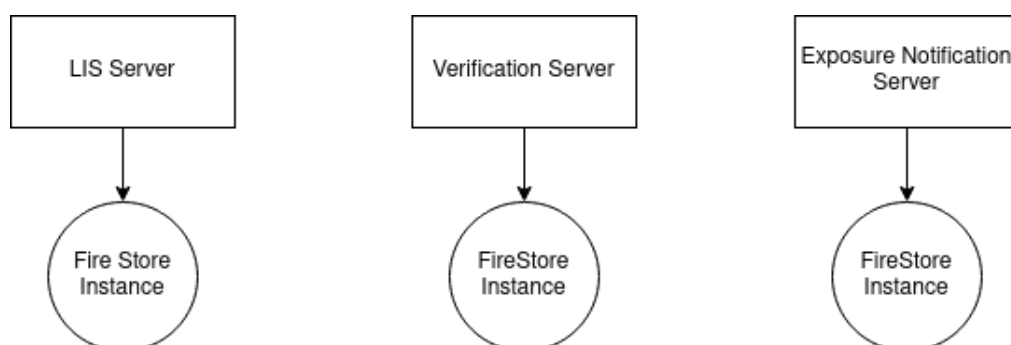


















Figure 3: High Level Design of the backend of CovidGuard-F

PRs Merged this week:

Ref:

<https://github.cs.adelaide.edu.au/2020-Mobile-and-Wireless-Systems/CovidGuard-F/pulls?q=is%3Apr+is%3Aclosed+author%3Aa1775690>

<input type="checkbox"/>	 Feat/cron ens - Add cron job to remove teks #63 by a1775690 was merged 23 hours ago • Approved	 1
<input type="checkbox"/>	 [LISServer] Add patient resources #62 by a1775690 was merged 23 hours ago • Approved	 1
<input type="checkbox"/>	 Refactor/upload tek - Separate RequestTAN and UploadTeK and chain them #61 by a1775690 was merged 23 hours ago • Approved	 1
<input type="checkbox"/>	 Remove livedata - #60 by a1775690 was merged 2 days ago	
<input type="checkbox"/>	 Feat/enhance download teks #58 by a1775690 was merged 2 days ago	 1
<input type="checkbox"/>	 Client Side Implementation of Downloading TEKs #52 by a1775690 was merged 2 days ago	 1
<input type="checkbox"/>	 [ens-server] view to download data #51 by a1775690 was merged 2 days ago	
<input type="checkbox"/>	 Feat/upload teks #50 by a1775690 was merged 3 days ago • Approved	 1
<input type="checkbox"/>	 Feat/ens - Client and Server code for uploading TEKs #49 by a1775690 was merged 3 days ago • Approved	 2

Plans for the Next Week:

1. Work with Lokesh on the consent flow of the User.
2. Fix few bugs in the network connectivity issues
3. Start working on the Product testing for battery efficiency and correctness of the implementation.
4. Explore Microservice patterns to make distributed transactions and hystrix circuits² to retry calls from one microservice to another.

² "Netflix/Hystrix - GitHub." <https://github.com/Netflix/Hystrix>. Accessed 12 Oct. 2020.