**Google IoT configuration for sending GPS data to firebase**

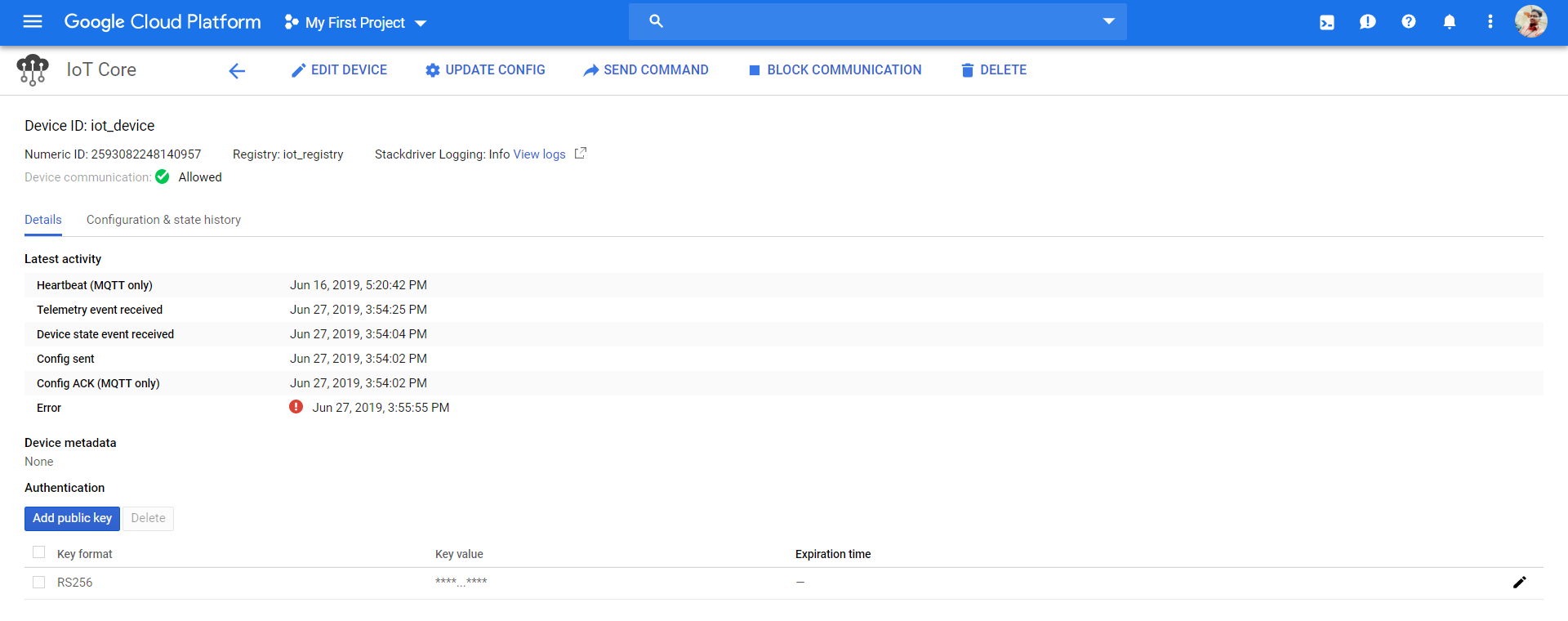
Date worked: 06/17 (8 hrs), 06/18 (8 hrs), 06/19 (8 hrs)

Summary of task undertaken:

1. Google cloud package (<https://www.st.com/en/embedded-software/x-cube-gcp.html>) was used to send the GPS data over the google firebase
2. Initially, I had to study the user manual for the GCP package to understand the working and installation steps. I installed the GCP package using the steps mentioned in sections 5.2 to 5.5 of the user manual (<https://www.st.com/content/ccc/resource/technical/document/user_manual/group1/e9/8a/9b/73/5c/ff/4d/10/DM00522079/files/DM00522079.pdf/jcr:content/translations/en.DM00522079.pdf>)
3. After unzipping the package, I imported the project in Eclipse with the given sample project “B-L475E-IOT01\_GoogleIoT”
4. I checked the github link shared by the professor (<https://github.com/lastjediluke/firefighterFinderGPS>) which mentions the implementation of Google IoT for the GPS. I studied the Readme file to understand the code and steps to be followed next for configuring the firebase and google cloud
5. As per the github link (inside Luke430 Complete folder), I modified the main.c, googleiot.c files to make it GPS specific. As per the readme, GPS was communicating with STM Discovery Kit at UART4.
6. After making the appropriate changes in code and running in debug mode, we get the output over teraterm as explained in section 5.5 of user manual. In teraterm, go to terminal->receive->AUTO, transmit->LF, Enable Local Echo. Go to serial port-->Baud rate: 115200

Perform the following steps as the output is shown in teraterm

1. Enter Wifi SSID, Password and security type. **Make sure to use your Mobile Hotspot to establish data connection.**
2. Update the TLS certifications by adding the public key and device key generated while configuring Google cloud.
3. Enter the Google cloud core connection string (project id, device id, registry id, region)
4. After socket connection is established, press the blue button twice to send data over the google cloud. Basically, it will subscribe to a pub/sub topic and publish the data over that topic. After successful sending of data, the google cloud indicates the date and time of data published.

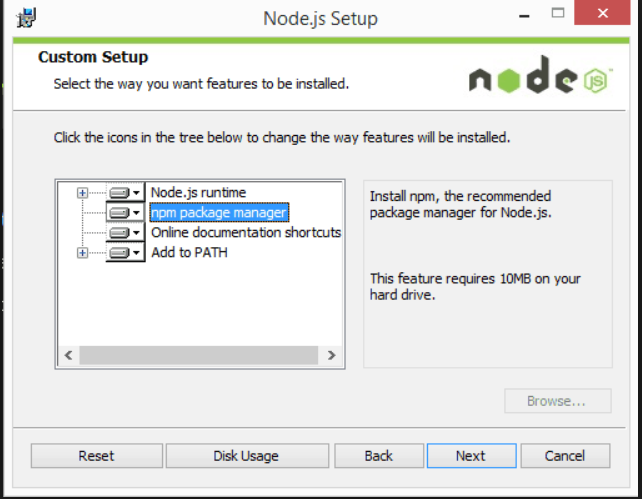


Note: The above error indicate that MQTT connection was lost. I took this screenshot when the discovery board was not plugged in.

7. Once the data was published in the google cloud, the next step was to send the data from Google cloud to Firebase. The firebase works with the google cloud project to display the data or send the data to the database (Ref: <https://www.youtube.com/watch?time_continue=263&v=DYfP-UIKxH0>)

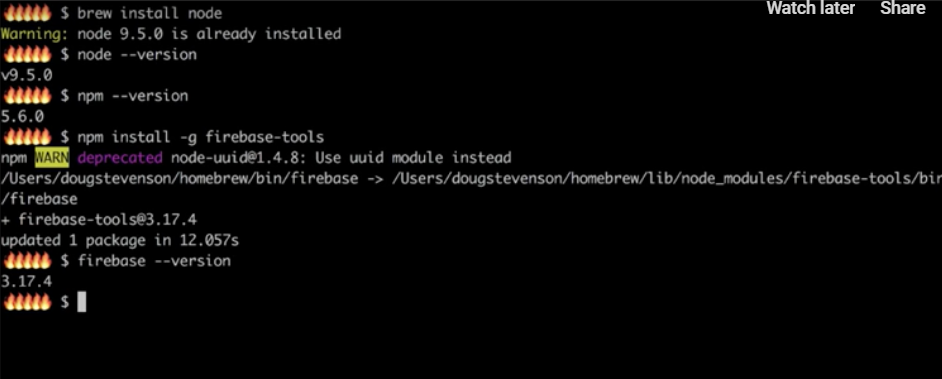
8. Go to firebase website (<https://firebase.google.com/docs>), login using same gmail as google cloud, add project (ensure that project id is same as that configured for google cloud) The following steps were followed to configure the firebase.

1. Download the node.js environment from the official website. Make sure while installing the node js, install the npm package as well.



2) Open the Node js command prompt and type “node --version” and it will show node js version. Type “npm --version” and it shows npm version. This step is to verify whether node js and npm are installed properly or not.

3) Install the Firebase CLI using command “npm install -g firebase-tools”. The path mentioned can be the path where you will install firebase. You can verify the firebase installation by typing “firebase --version”



4) Create a new directory named “firebase” and go inside the firebase folder using “cd firebase”

5) Type “firebase login” to login into google firebase account. It will provide a link to login and complete the authentication. You need to login using same gmail id as you used during configuring the google cloud, since google cloud shall be linked.



6) Type “firebase init” to initialize the firebase. It will ask a few questions to configure the firebase. First shall be to select CLI feature, select all the features: Functions, Database, firestore, hosting, storage.

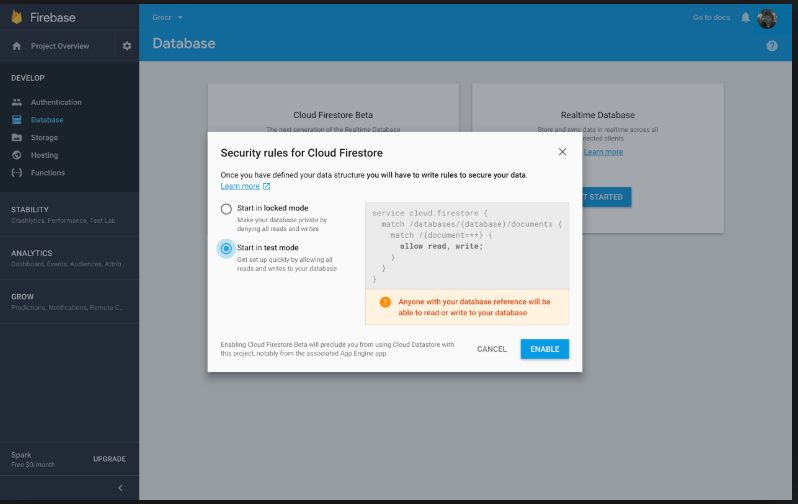


7) Next is select a firebase project: Select the google cloud project id from the list.

8) Configuring the firebase functions. Next question is the type of language used to write cloud functions: Javascript

9) Do you want to use TSLint for catching bugs--YES

10) Follow the similar steps to configure database, storage, firestore, hosting. Once the configuration is complete, go to firebase console-->Database--> Start in test mode and enable



11) Go to the firebase folder-->functions-->Index.js. Copy the index.js from the github link(Luke430Complete) into index.js of the current folder.

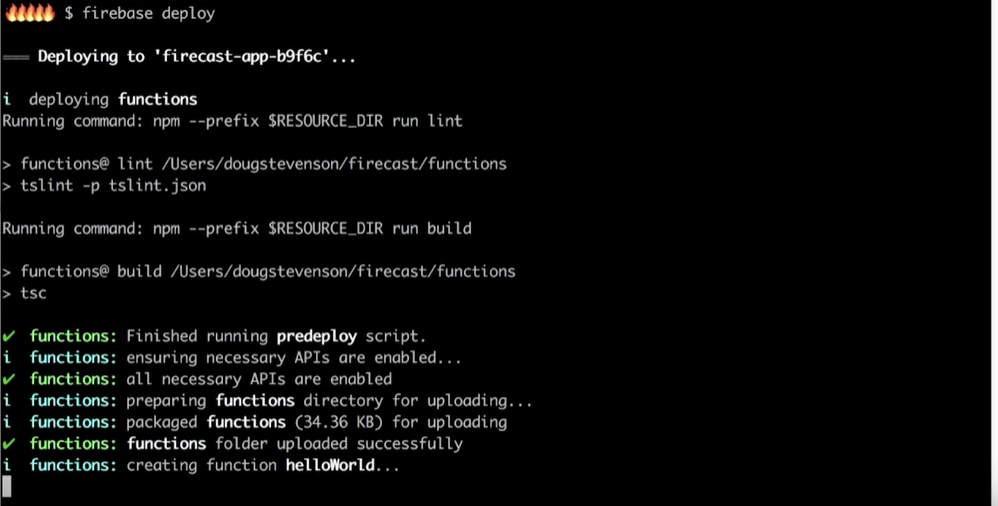
12) Go to nodejs command prompt, type

*cd functions*

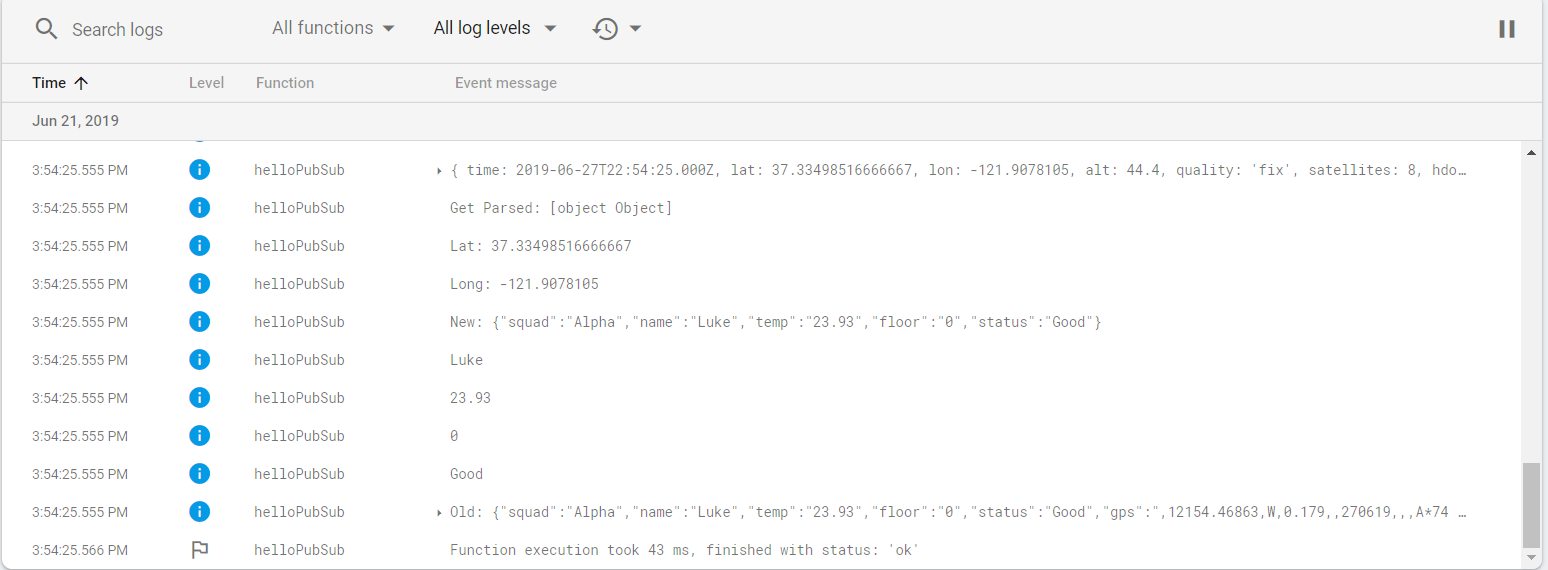
*npm install firebase-admin@latest firebase-functions@latest*

13)Add gps dependencies by typing “npm install gps”

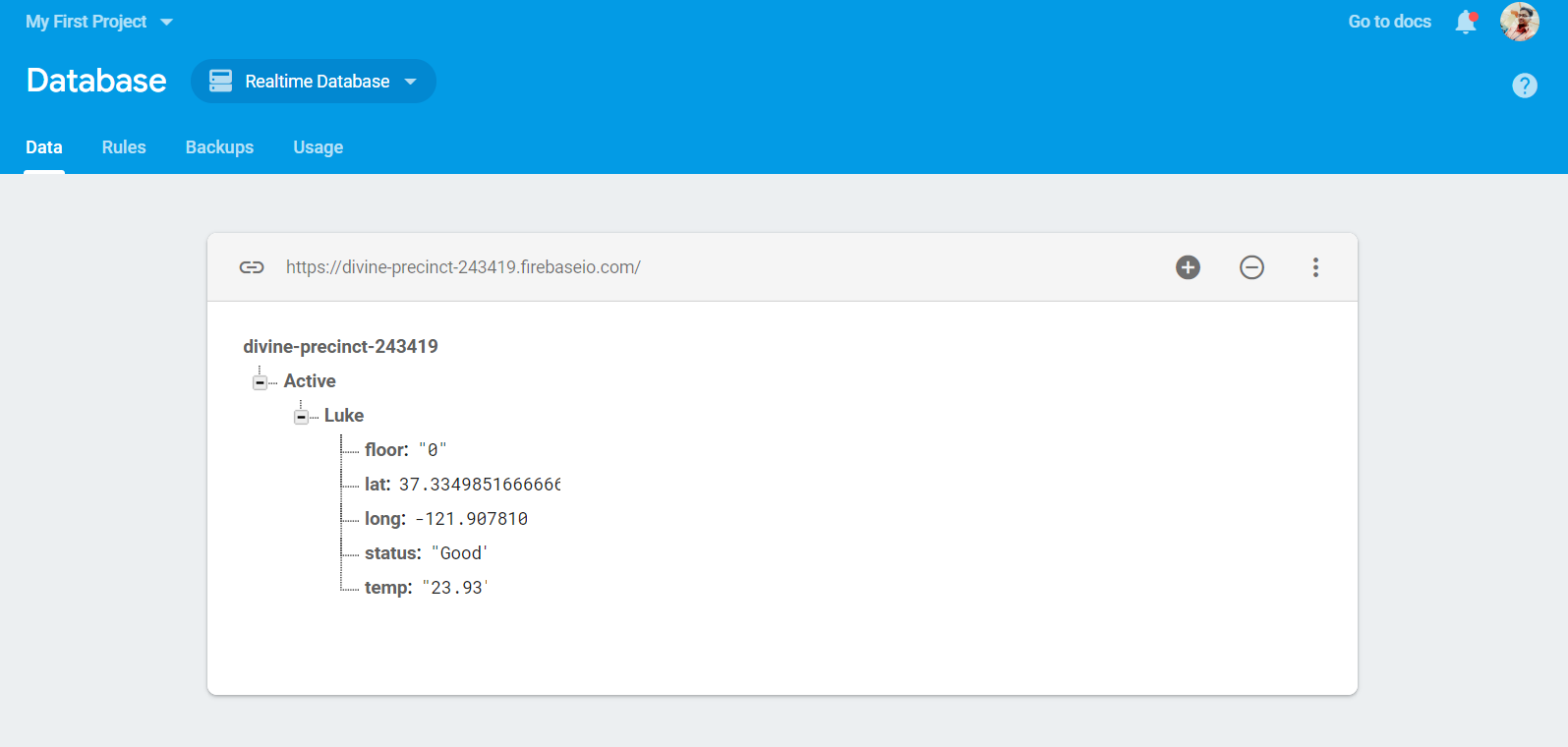
14) Type “firebase deploy” to run the index.js code. The below screenshot shows a hello world project. In our case, the project name is different. After deploying, it will mention as “Deploy complete”



15) Go to firebase console-->functions-->logs. You will be able to see the logs of the data received from google cloud to firebase.



16) Go to database-->realtime data and you will be able to see the GPS coordinates, status and temperature, which updated itself every 3-5 seconds



**CHALLENGES:**

1. **I was not able to configure the firebase using the steps mentioned in firebase website. The data was not getting published to the firebase**

Solution: I did not configure the database as CLI feature in nodejs command prompt. After configuring it, i had to enable the database from the firebase console. It took a day to solve this issue

2. **After firebase deploy, it was throwing an error of gps dependencies not**

**Installed. I tried the command line “npm install gps” and still it was**

**throwing an error.**

Solution: Changed the directory and installed the gps inside node module folder

and functions folder. It deployed fine

3.  **During configuring TLS certificates in teraterm, authentication failed**

**multiple times.**

Solution: Generating a new device key solved the issue.

**MILESTONES:**

1. GPS data was successfully sent from microcontroller to google cloud and firebase.
2. Real time GPS data monitoring achieved

**NEXT MILESTONES:**

1. Working with CO sensor
2. Integrating CO Sensor with GPS in same board--changing the uart4 to uart2
3. Integrating Freertos in google iot code and combining GPS and CO sensor data