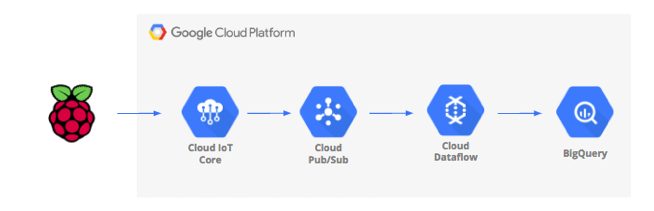
**GOOGLE IOT CORE TO STREAM HEART RATE DATA using RASPBERRY PI**

Internet of things , or IoT refers to a millions of physical devices around the world that are now connected to internet connecting and sharing data. Anyways there are a few basic requirements such as security maintenance , handling the data ingestion requirements of many distributed devices remain as significant challenges. Google Cloud's IoT Core addresses simplifies these tasks which makes it easier to focus on gaining value and insight from an IoT project. In this project we are going to build a Raspberry Pi with a heart rate sensor and several components of cloud platform are also used to form the pipeline.



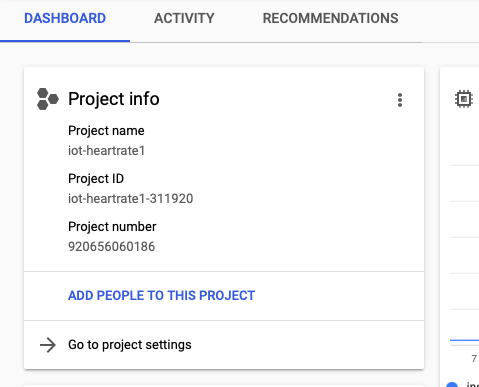
In order to build a IoT device , we need to set-up the hardware. The required hardware are

1. Raspberry Pi Z with power supply
2. Sd memory card and USB card reader
3. A heart rate transmitter or a polar heart rate receiver
4. A USB hub set up ( USB cables for keyboard, mouse etc)
5. Female to male wires
6. Access to a computer monitor or a TV with HDMI cable

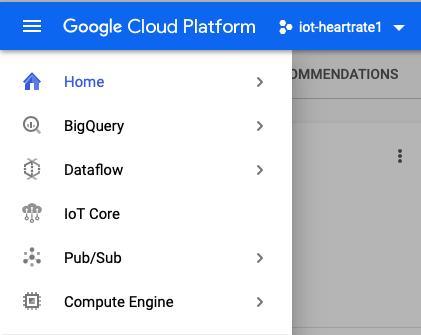
**SETTING UP THE CLOUD**

**Step 1** : Sign in to your google cloud console using your google account and create if you don’t have one. [console.cloud.google.com](http://console.cloud.google.com/),

**Step 2** : Go to cloud console and create a new project with a unique project ID. Your cloud console will be as shown below



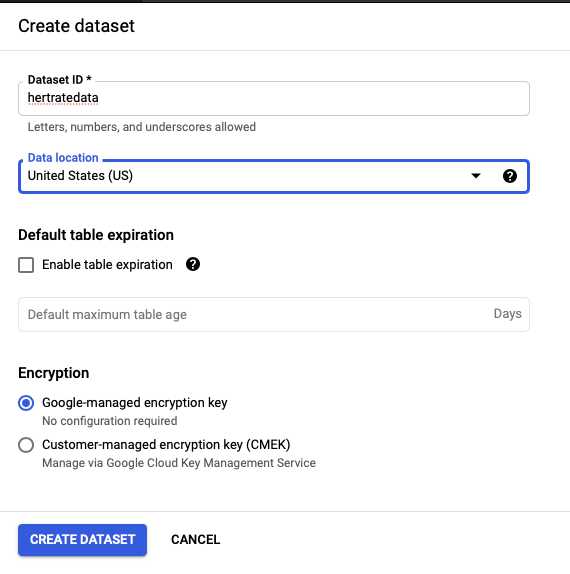
**Step 3**: The next most important step to di is enabling the API services. In the cloud console , select API’s and Services and enable them. We will be using Pub/Sub subscription, dataflow, compute engine, IoT core , so we must enable all of these API’s.



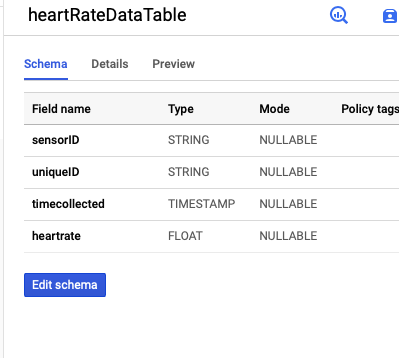
**Step 4**: **Creating a Big Query Table**

A BigQuery is a data warehouse which is for storing the data streamed from IoT devices. A table must be created to hold all of the IoT heartrate data.

Initially , go to big query , create a new dataset , name it ‘ heartratedata’ , select your location and create dataset.



Click on the ‘+’ sign , create a new empty table and the table name as ‘heartratedatatable. Under schema, click on add fields and add 4 new fields and fill in the fields as SensorID, uniqueID, timecollected , heartrate and select the appropriate type as shown below. Click on create table button. The result will be as follows



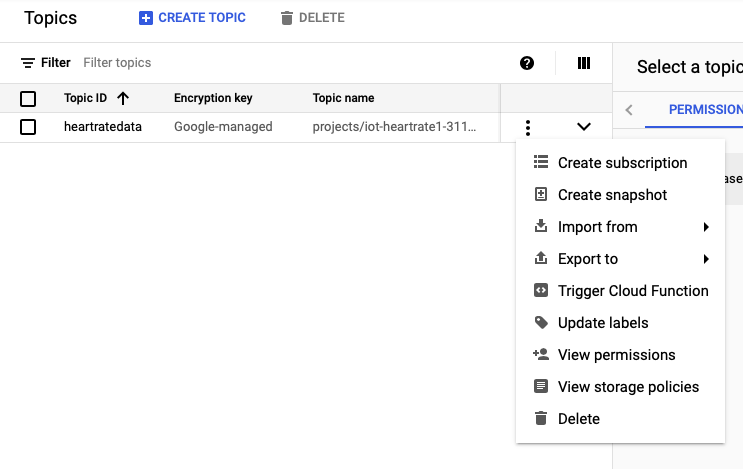
We now have a warehouse set up for receiving the heart rate data.

**Step 5: Creating a Pub/Sub topic and subscription**

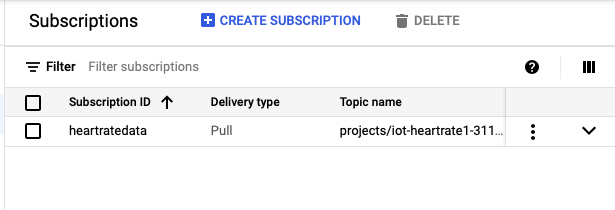
Cloud Pub/Sub is a simple , reliable foundation for streaming data. It is perfect for handling incoming messages and then process them to the linked Cloud IoT core.

Select Pub/Sub and then Topics. Create a topic with name ‘heartratedata’.

We now have a Pub/Sub set to publish IoT messages. Next a subscription needs to be created which allows other users to access this data.



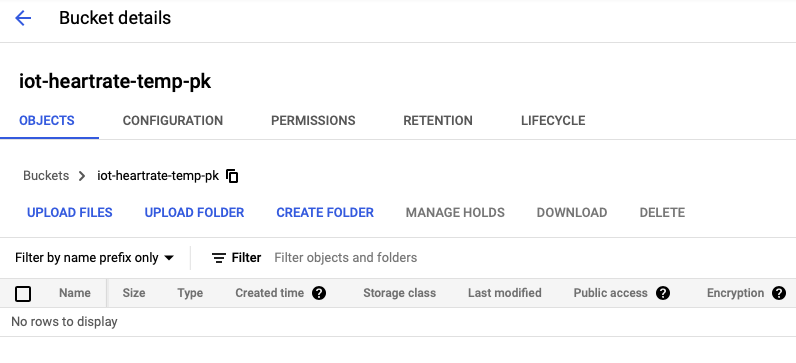
Enter ‘heartratedata’ as the subscription name , select PULL as the delivery type and create. We now have a Pub/Sub subscription ready to pull the IoT messages.



**Step 6: Using a Dataflow template**

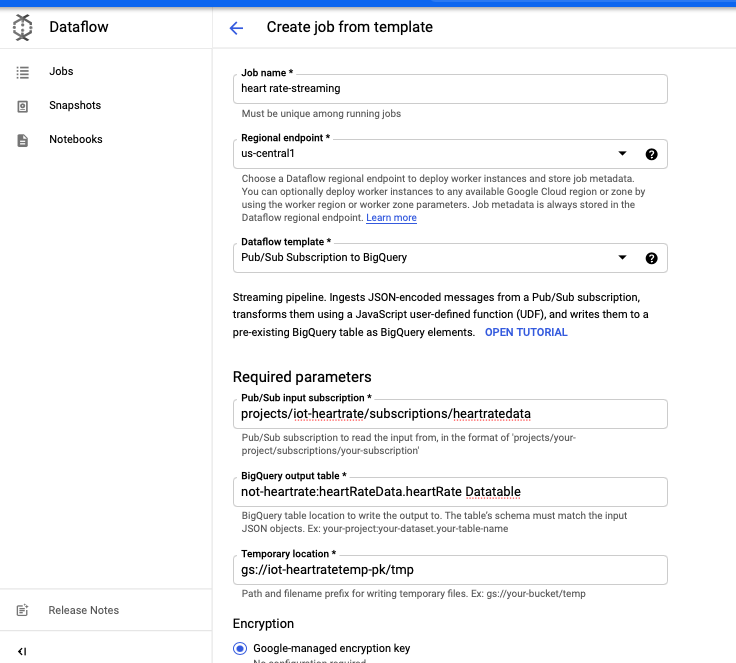
We will be using a dataflow template to create a process that monitors a Pub/Sub topic for messages and then transmits them to BigQuery. This will need a location for storing data temporarily , we need to provide a location in cloud.

Go to Cloud console—> Storage🡪 browser 🡪 Create bucket. (The bucket name should be unique globally across the Google Cloud).



Now , go to cloud console -> Dataflow -> create job from template. Name the job , select Pub/Sub subscription to Big Query dataset and tablename.

Enter the details as shown in following figure.



The dataflow job is started.

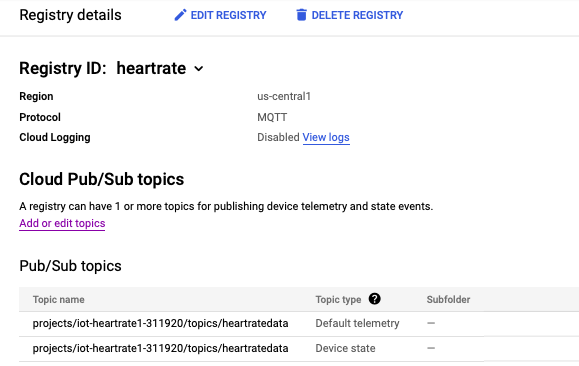


**Step 7 : The IoT core registry**

This is a completely managed service which gives us a chance to easily connect , manage data , ingest data securely from millions of other devices.

Go to Cloud IoT core from the cloud console . Create a Registry. Enter ‘heartrate’ as Registry ID , region , enable the MQTT protocol and select the Pub/Sub topic we created.

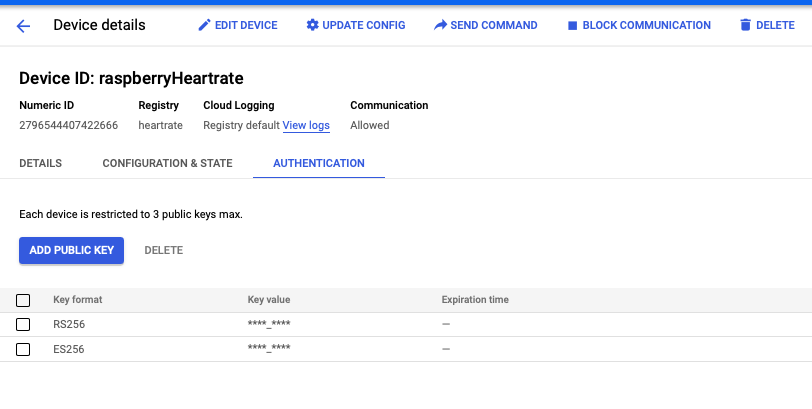
Create the Registry.



**Step 8: Adding the device to the Registry**

Go to your registry -> add device. Enter ‘raspberryheartrate’ as device name and select the ES 256 Public key format. Upload the public key by generating the key and add the device. The following link can be used for generating keys.

<https://cloud.google.com/iot/docs/how-tos/credentials/keys>



The core IoT is now completely ready to receive data from Raspberry Pi.

**Step 9: Streaming the data from Pi**

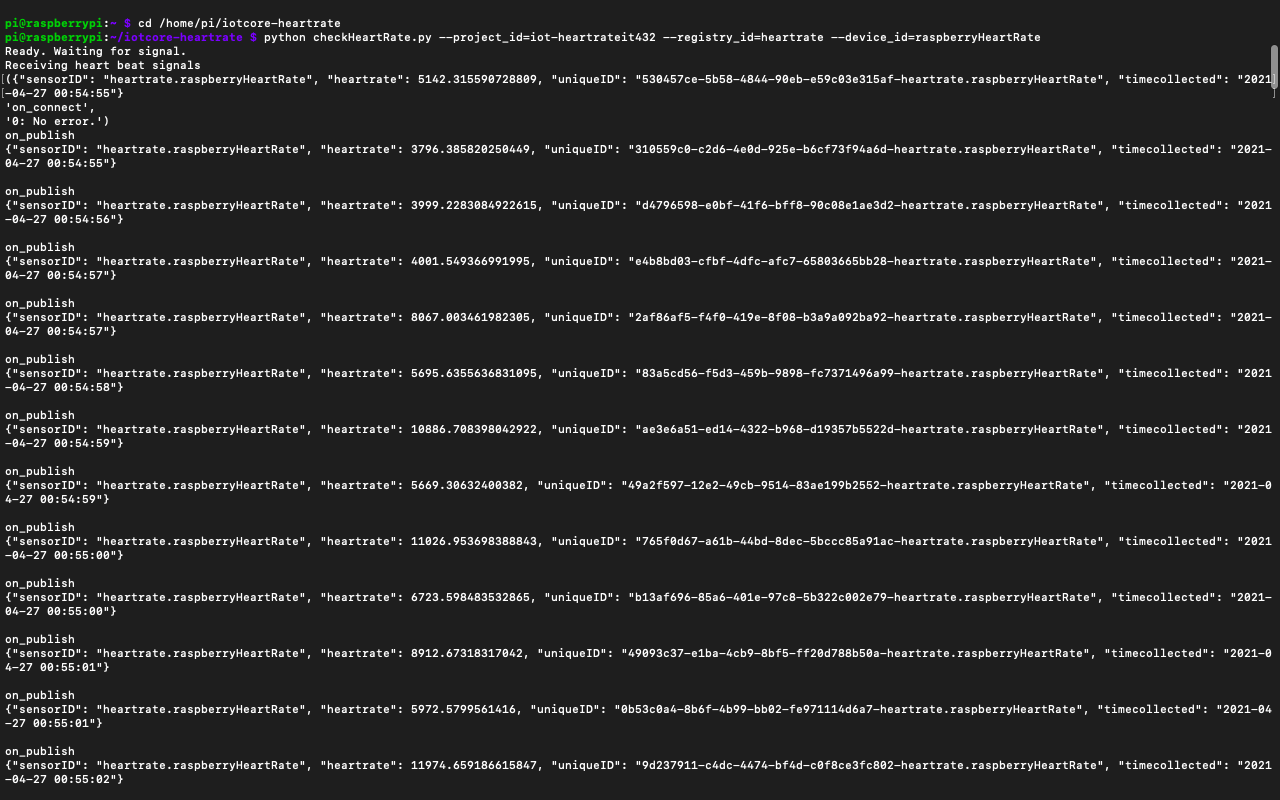
After completing the hardware set up , put on the strap and start the script to receive the heartrate signals.

Use the following commands



Match your project , registry and device and start the heartrate script with the following code

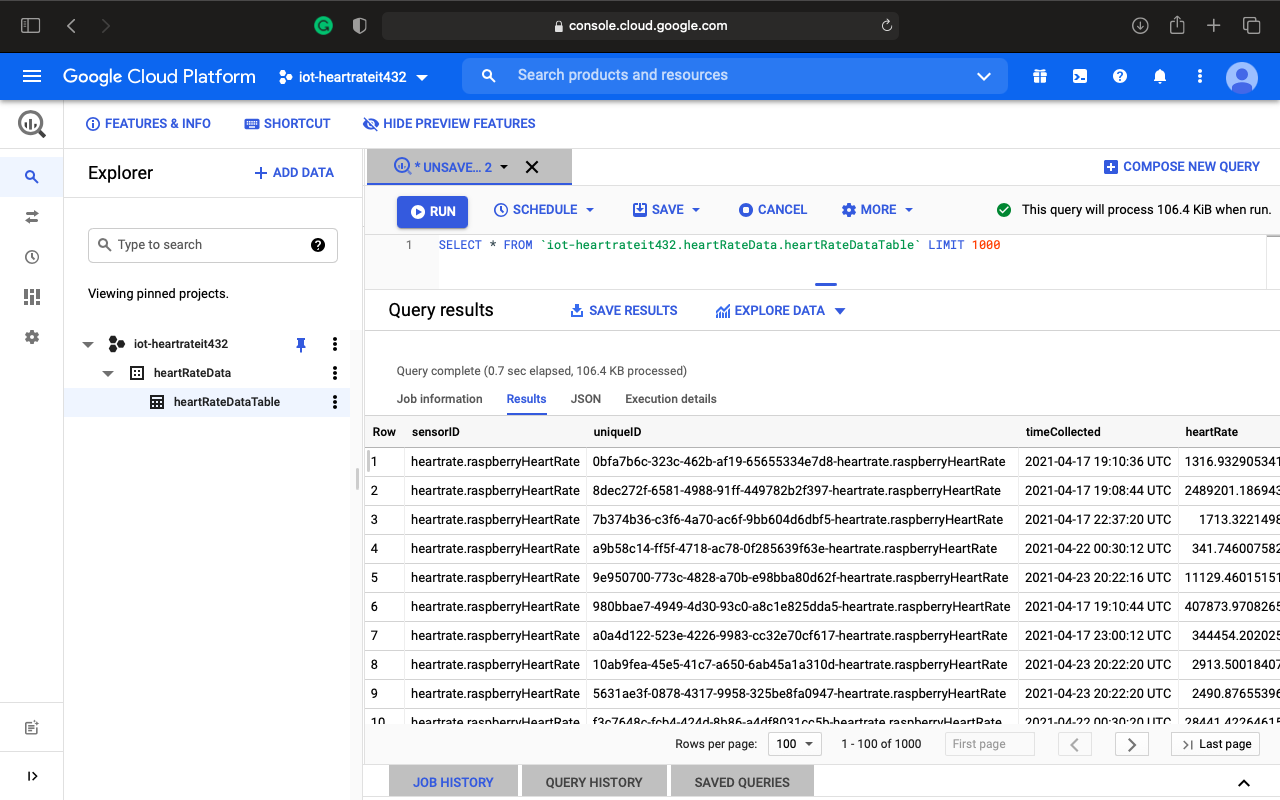




**Step 10 : Check if data is flowing**

To check, Go to Big Query-> under project click on dataset -> click on query table. The only change we have to make is adding an \* to SQL query in the file. Run query.

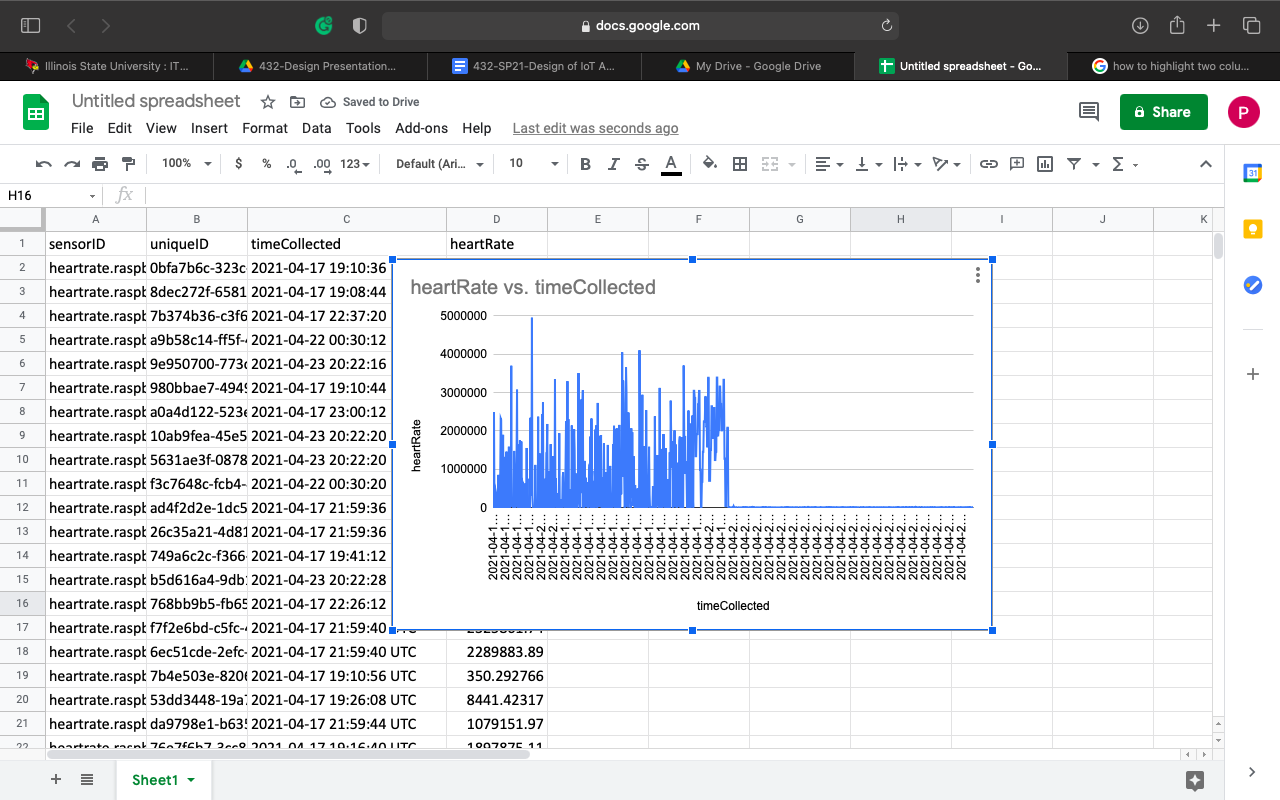
If you see results , dataflow is proper.



**Step 11: Visualization**

As the data is read into the cloud , we next visualize it. Save the data to Google sheets.

Highlight the ‘timecollected’ and ‘heartrate’ columns. Go to Insert-> chart-> Line graph



The entire data pipeline creation is done. We have successfully used the Google cloud to secure IoT devices and allowed the data to flow into Google Pub/Sub , dataflow from a template and sent the data to Big Query and got appropriate results. This documentation shows the complete set up for google IoT core to stream heart rate data using raspberry pi device along with various final visualizations.