

Connected Devices

Lab Module Assignment - Final Project

Name and Course

- Name: Dharani Thirumalaisamy
- Course: Connected Devices
- Semester and Year: Spring'19

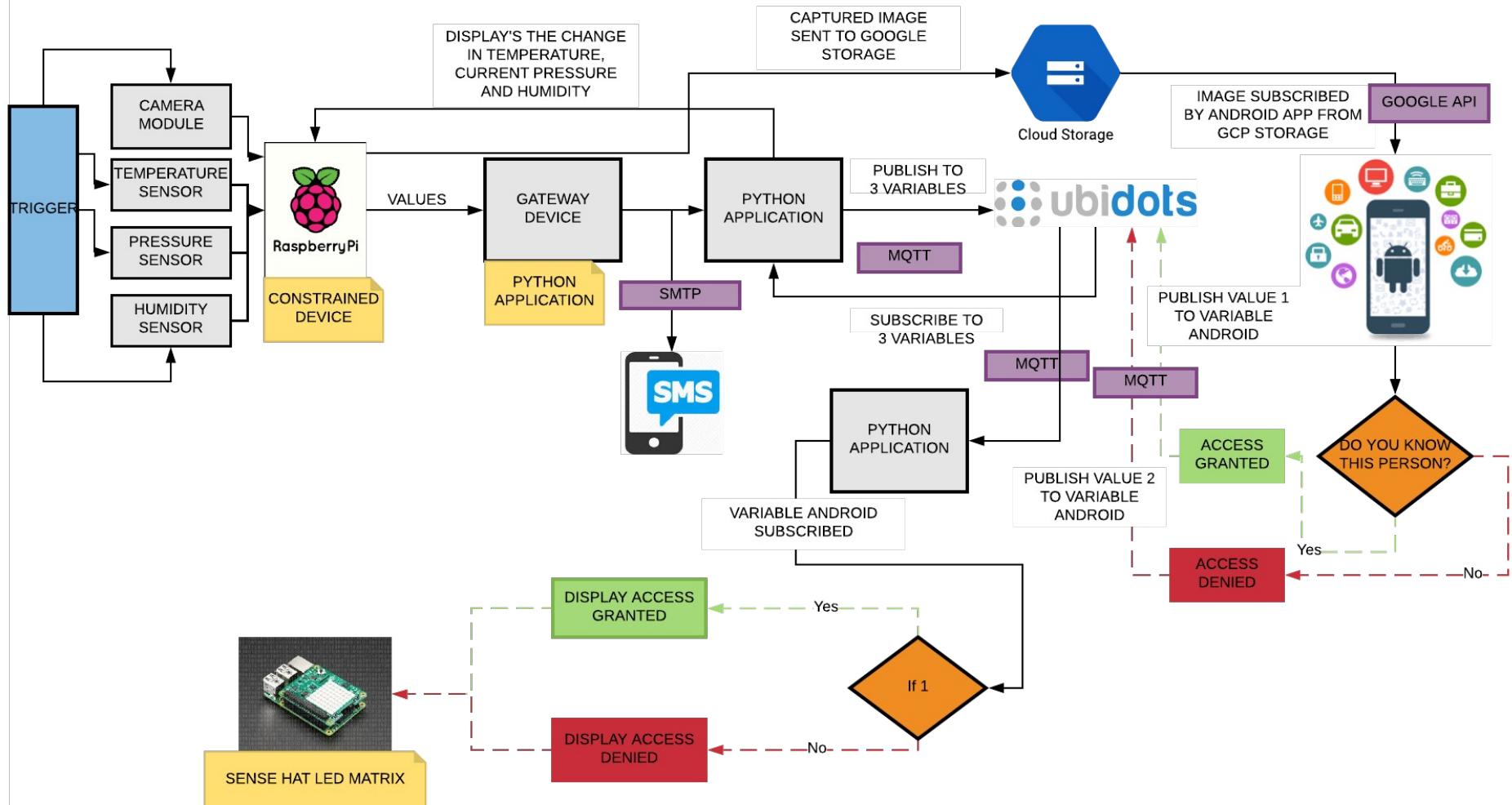
Description

- Description: An End-to-End Facial Recognition based Security System.

URL TO PROJECT DEMO : <https://youtu.be/LZh16xVoErY>

URL TO PROJECT SCRIPTS AND REPORT : <https://github.com/Dharani-Thirumalaisamy/IOT-Workspace/tree/master>

FLOW DIAGRAM



Problem Statement :

Robery has become a common incident in many places around the world today. As IOT is trying to solve many problems that is present in the society, my idea is to implement a IOT based security door which will be able to capture pictures of people who stand outside the door and send a warm welcome message to the owner and will notify the owner that there is a intruder and ask if it should provide access or not.

Protocols Used :

1. Mqtt
2. SMTP

Sensors Used :

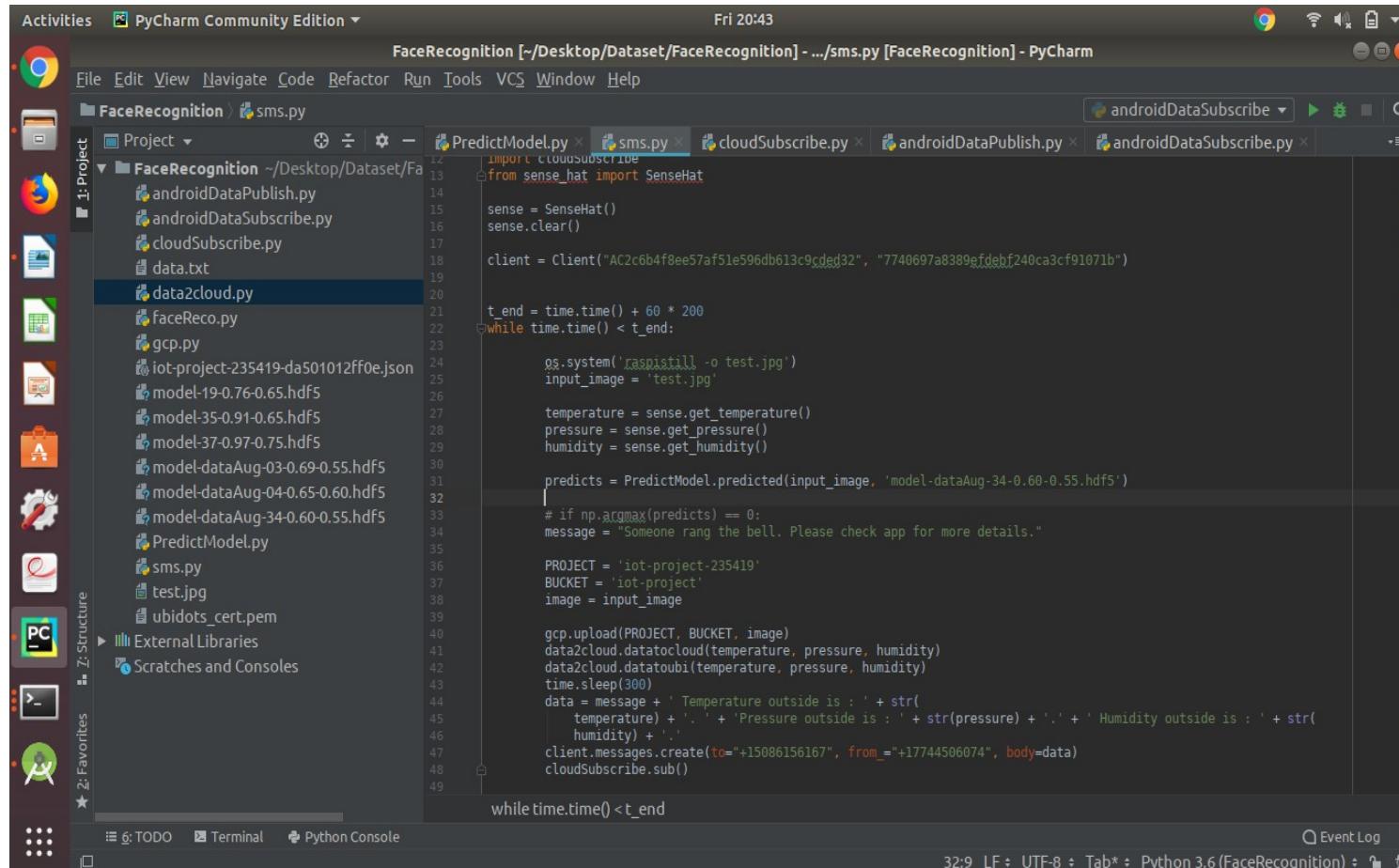
1. Raspberry pi camera module/sensor
2. Humidity Sensor
3. Temperature Sensor
4. Presssure Sensor

Cloud Services:

1. Google Cloud Platform
2. UBIDOTS

Application Script :

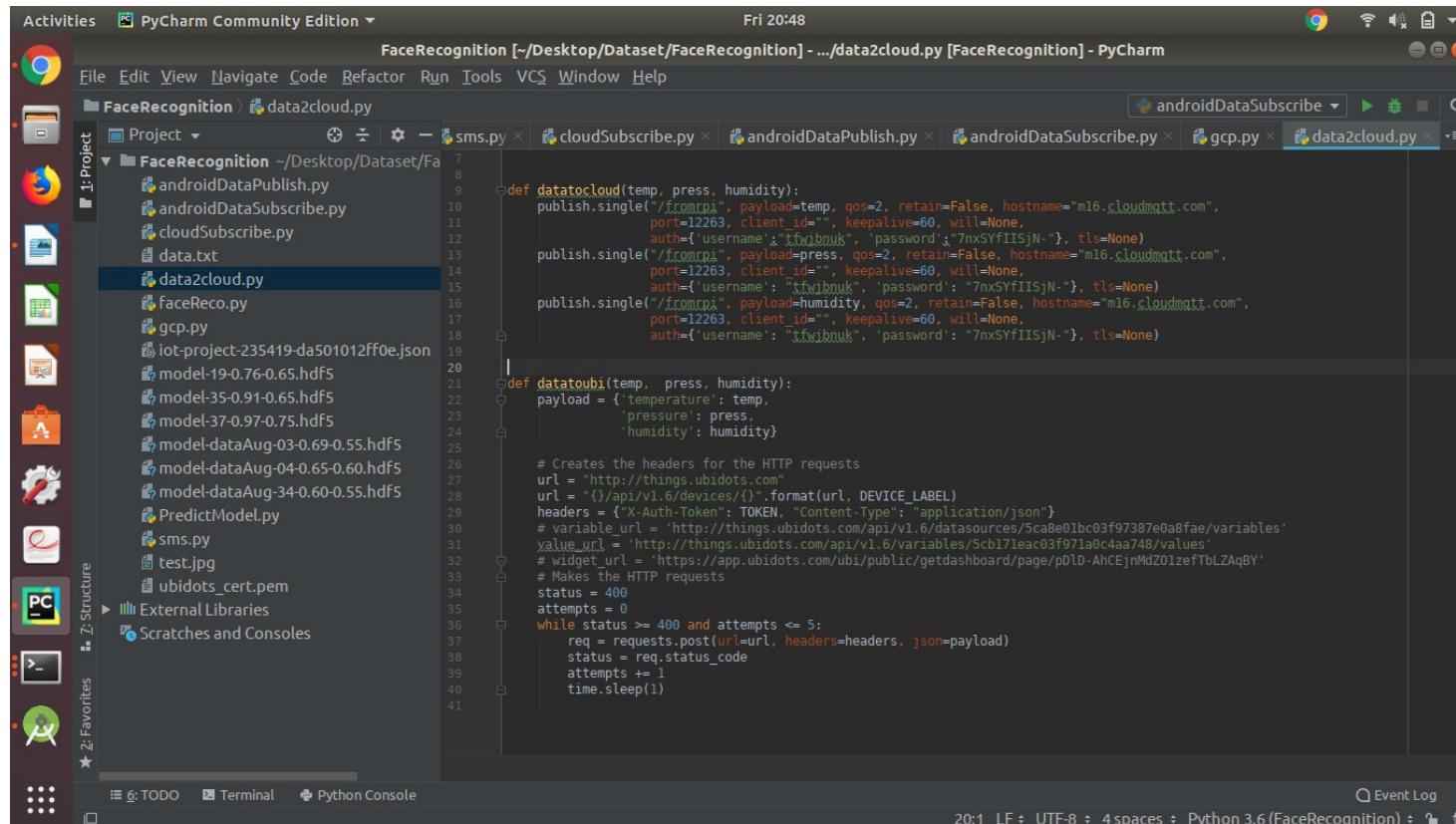
1. Trigger the sensor



The screenshot shows the PyCharm Community Edition interface with the following details:

- Title Bar:** FaceRecognition [~/Desktop/Dataset/FaceRecognition] - .../sms.py [FaceRecognition] - PyCharm
- File Menu:** File Edit View Navigate Code Refactor Run Tools VCS Window Help
- Project Tree:** FaceRecognition (~/Desktop/Dataset/FaceRecognition) contains:
 - Project
 - FaceRecognition (~/Desktop/Dataset/FaceRecognition)
 - androidDataPublish.py
 - androidDataSubscribe.py
 - cloudSubscribe.py
 - data.txt
 - data2cloud.py
 - faceReco.py
 - gcp.py
 - iot-project-235419-da501012ff0e.json
 - model-19-0.76-0.65.hdfs
 - model-35-0.91-0.65.hdfs
 - model-37-0.97-0.75.hdfs
 - model-dataAug-03-0.69-0.55.hdfs
 - model-dataAug-04-0.65-0.60.hdfs
 - model-dataAug-34-0.60-0.55.hdfs
 - PredictModel.py
 - sms.py
 - test.jpg
 - ubidots_cert.pem
 - External Libraries
 - Scratches and Consoles
- Code Editor:** The sms.py file is open, showing Python code for triggering a sensor. The code imports cloudSubscribe, sense_hat, and PredictModel. It initializes a SenseHat object, sets a client, and defines a loop to take a photo, get sensor data, predict, and upload to Google Cloud Storage. It also sends an SMS message and subscribes to a cloud topic.
- Status Bar:** Fri 20:43, 32:9 LF, UTF-8, Tab*, Python 3.6 (FaceRecognition)

2. Send data to gateway and cloud :



The screenshot shows the PyCharm Community Edition interface with the following details:

- Title Bar:** Activities PyCharm Community Edition ▾ FaceRecognition [-/Desktop/Dataset/FaceRecognition] - .../data2cloud.py [FaceRecognition] - PyCharm
- File Menu:** File Edit View Navigate Code Refactor Run Tools VCS Window Help
- Toolbar:** Includes icons for browser, file, search, and others.
- Project Tree:** Shows the project structure under "FaceRecognition":
 - Project
 - FaceRecognition (~/Desktop/Dataset/FaceRecognition)
 - androidDataPublish.py
 - androidDataSubscribe.py
 - cloudSubscribe.py
 - data.txt
 - data2cloud.py
 - faceReco.py
 - gcp.py
 - iot-project-235419-da501012ff0e.json
 - model-19-0.76-0.65.hdf5
 - model-35-0.91-0.65.hdf5
 - model-37-0.97-0.75.hdf5
 - model-dataAug-03-0.69-0.55.hdf5
 - model-dataAug-04-0.65-0.60.hdf5
 - model-dataAug-34-0.60-0.55.hdf5
 - PredictModel.py
 - sms.py
 - test.jpg
 - ubidots_cert.pem
 - External Libraries
 - Scratches and Consoles
- Code Editor:** Displays the Python script "data2cloud.py". The code defines two functions: `dataToCloud` and `dataToUbidot`. The `dataToCloud` function publishes messages to a MQTT broker. The `dataToUbidot` function creates HTTP requests to the Ubidots API to send data.
- Bottom Status Bar:** Shows the current file is "data2cloud.py", encoding is "UTF-8", and the Python version is "Python 3.6 (FaceRecognition)".

3. Image to Google Cloud :

The screenshot shows the PyCharm Community Edition interface with the following details:

- Title Bar:** Activities PyCharm Community Edition ▾ FaceRecognition [~/Desktop/Dataset/FaceRecognition] - .../gcp.py [FaceRecognition] - PyCharm Fri 20:50
- File Menu:** File Edit View Navigate Code Refactor Run Tools VCS Window Help
- Toolbar:** androidDataSubscribe ▾
- Project Explorer:** Project ▾ FaceRecognition ~/Desktop/Dataset/FaceRecognition
 - AndroidDataPublish.py
 - AndroidDataSubscribe.py
 - cloudSubscribe.py
 - data.txt
 - data2cloud.py
 - faceReco.py
 - gcp.py
 - iot-project-235419-da501012ff0e.json
 - model-19-0.76-0.65.hdf5
 - model-35-0.91-0.65.hdf5
 - model-37-0.97-0.75.hdf5
 - model-dataAug-03-0.69-0.55.hdf5
 - model-dataAug-04-0.65-0.60.hdf5
 - model-dataAug-34-0.60-0.55.hdf5
 - PredictModel.py
 - sms.py
 - test.jpg
 - ubidots_cert.pem
- Code Editor:** gcp.py (selected)

```
from google.cloud import storage
import os
import json
import time

os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "iot-project-235419-da501012ff0e.json"
storage_client = storage.Client()
buckets = list(storage_client.list_buckets())
bucket = storage_client.get_bucket("iot-project")

def upload(project, bucket, image):
    FILENAME = image
    client = storage.Client(project)
    bucket = client.get_bucket(bucket)
    blob = bucket.blob('Test Image')
    blob.upload_from_filename(FILENAME, content_type='image/jpeg')
    blob.make_public()
    url = blob.public_url
    print("Image Uploaded")
    time.sleep(300)
```
- Bottom Status Bar:** TODO Terminal Python Console Event Log 21:19 LF + UTF-8 + 4 spaces + Python 3.6 (FaceRecognition) +

4. Image to Android App:

The screenshot shows the Android Studio interface with the following details:

- Title Bar:** Activities JetBrains-studio ▾ Fri 20:51
- Toolbar:** File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
- Project Navigators:** Project, Layout Captures, Favorites, Build Variants.
- Code Editor:** Displays the `MainActivity.java` file under the `com.example.myapplication` package. The code implements an `ImageView` to download an image from the internet.

```
22     public class MainActivity extends AppCompatActivity {
23
24         @Override
25         protected void onCreate(Bundle savedInstanceState) {
26             super.onCreate(savedInstanceState);
27             setContentView(R.layout.activity_main);
28
29             new DownloadImageFromInternet((ImageView) findViewById(R.id.imageView))
30                 .execute("https://00e9e64bacea57b5e675e5667f37ba48dfc1df846e4fe30c67-apidata.googleapis.com");
31
32         }
33         private class DownloadImageFromInternet extends AsyncTask<String, Void, Bitmap> {
34             ImageView imageView;
35
36             public DownloadImageFromInternet(ImageView imageView) {
37                 this.imageView = imageView;
38                 Toast.makeText(getApplicationContext(), "Please wait, it may take a few minute...", Toast.LENGTH_SHORT).show();
39             }
40
41             protected Bitmap doInBackground(String... urls) {
42                 String imageURL = urls[0];
43                 Bitmap bimage = null;
44                 try {
45                     InputStream in = new java.net.URL(imageURL).openStream();
46                     bimage = BitmapFactory.decodeStream(in);
47                 } catch (Exception e) {
48                     Log.e("Error Message", e.getMessage());
49                     e.printStackTrace();
50                 }
51                 return bimage;
52             }
53         }
54     }
```

- Terminal:** Shows the command `dharani@jarvis:~/Android/Sdk/platform-tools$`.
- Bottom Status Bar:** Event Log, Gradle build finished in 1 s 966 ms (4 minutes ago), 30:80 CRLF, UTF-8, Context: <no context>, 7.

Activities jetbrains-studio ▾

Fri 20:51

MyApplication [~/AndroidStudioProjects/MyApplication] - .../app/src/main/java/com/example/myapplication/MainActivity.java [app] - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

1: Project

1:1: app

Android manifests AndroidManifest.xml

java com.example.myapplication

Main2Activity Main3Activity MainActivity

com.example.myapplication (androidTest)

com.example.myapplication (test)

generatedJava com.example.myapplication

BuildConfig

res drawable

layout activity_main.xml activity_main2.xml activity_main3.xml

mipmap values

Gradle Scripts

Gradle

MainActivity.java

AndroidManifest.xml

Main2Activity.java

Main3Activity.java

Fri 20:51

```
protected Bitmap doInBackground(String... urls) {  
    String imageURL = urls[0];  
    Bitmap bimage = null;  
    try {  
        InputStream in = new java.net.URL(imageURL).openStream();  
        bimage = BitmapFactory.decodeStream(in);  
    } catch (Exception e) {  
        Log.e(tag, "Error Message", e.getMessage());  
        e.printStackTrace();  
    }  
    return bimage;  
}  
  
protected void onPostExecute(Bitmap result){ imageView.setImageBitmap(result); }  
  
public void onButtonClick(View v){  
    Intent intent = new Intent(getApplicationContext(), Main2Activity.class);  
    intent.putExtra(name, "Yes", value: "Access Granted");  
    Log.d(tag, "STATE", msg: "Access Granted");  
}  
public void onButtonClick2(View v){  
    Intent intent = new Intent(getApplicationContext(), Main3Activity.class);  
    intent.putExtra(name, "No", value: "Access Denied");  
    Log.d(tag, "STATE", msg: "Access Denied");  
}  
startActivity(intent);  
}  
}
```

MainActivity > onCreate()

Terminal

dharani@jarvis:~/Android/Sdk/platform-tools\$

4: Run 6: Logcat 8: TODO 9: Terminal 10: Build 11: Profiler

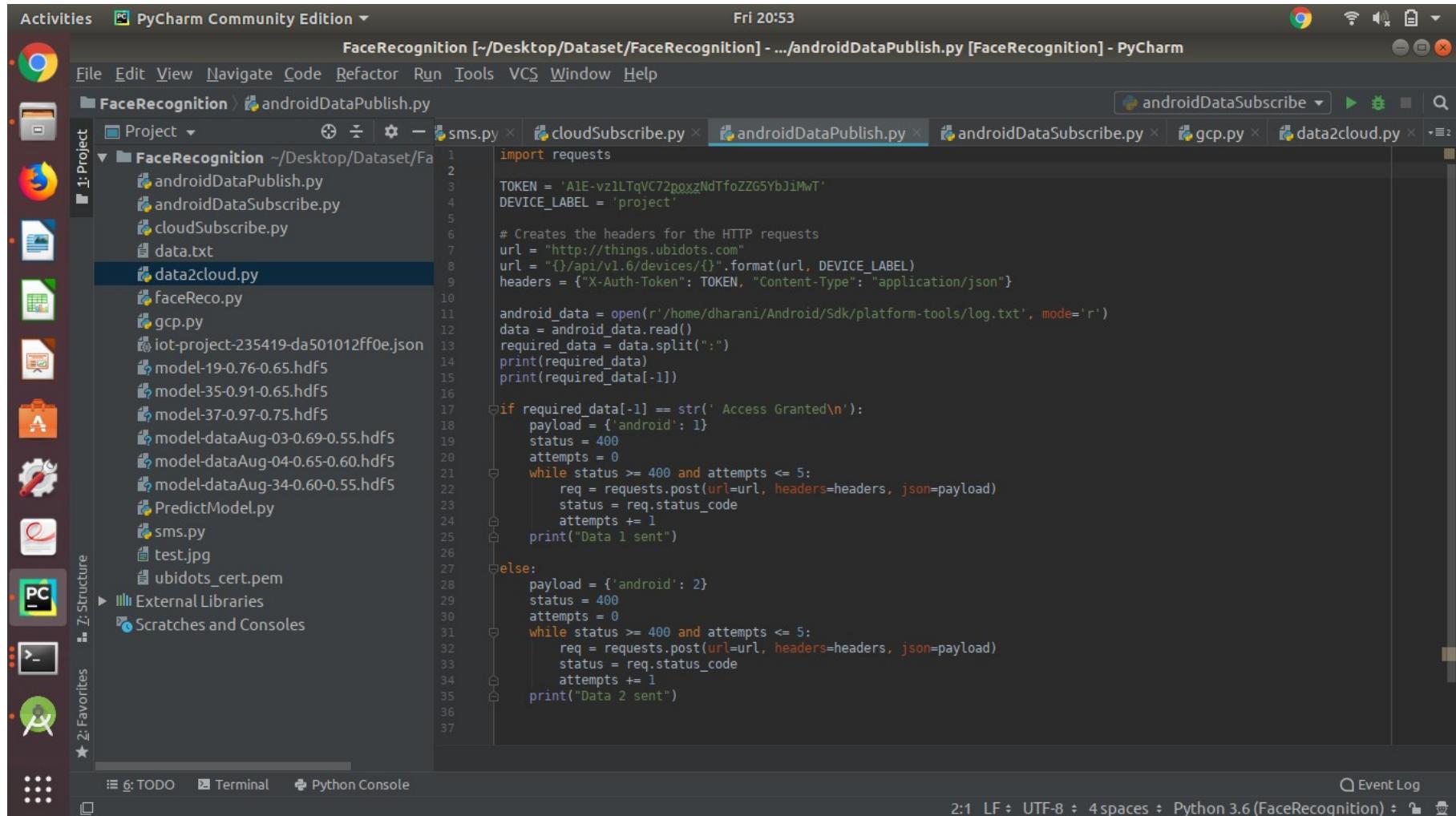
Gradle build finished in 1 s 966 ms (4 minutes ago)

Event Log

30:80 CRLF 10: UTF-8 Context: <no context>

Device File Explorer

5. Android data publish and subscribe :



The screenshot shows the PyCharm Community Edition interface with the following details:

- Title Bar:** Activities PyCharm Community Edition ▾ FaceRecognition [~/Desktop/Dataset/FaceRecognition] - .../androidDataPublish.py [FaceRecognition] - PyCharm
- Toolbar:** File Edit View Navigate Code Refactor Run Tools VCS Window Help
- Project Tree (1: Project):** FaceRecognition (~/Desktop/Dataset/FaceRecognition) containing files: androidDataPublish.py, androidDataSubscribe.py, cloudSubscribe.py, data.txt, data2cloud.py, FaceReco.py, gcp.py, iot-project-235419-da501012ff0e.json, model-19-0.76-0.65.hdfs, model-35-0.91-0.65.hdfs, model-37-0.97-0.75.hdfs, model-dataAug-03-0.69-0.55.hdfs, model-dataAug-04-0.65-0.60.hdfs, model-dataAug-34-0.60-0.55.hdfs, PredictModel.py, sms.py, test.jpg, ubidots_cert.pem.
- Code Editor:** The 'androidDataPublish.py' file is open, showing Python code for publishing data to Ubidots. The code imports requests, defines TOKEN and DEVICE_LABEL, creates headers, reads android_data from log.txt, splits it into required_data, and then attempts to post data to the Ubidots API (http://things.ubidots.com/api/v1.6/devices/DEVICE_LABEL) with payload {'android': 1} or {'android': 2} until status_code is 200. It includes print statements for each attempt.
- Status Bar:** Fri 20:53, 2:1 LF, UTF-8, 4 spaces, Python 3.6 (FaceRecognition), Event Log.

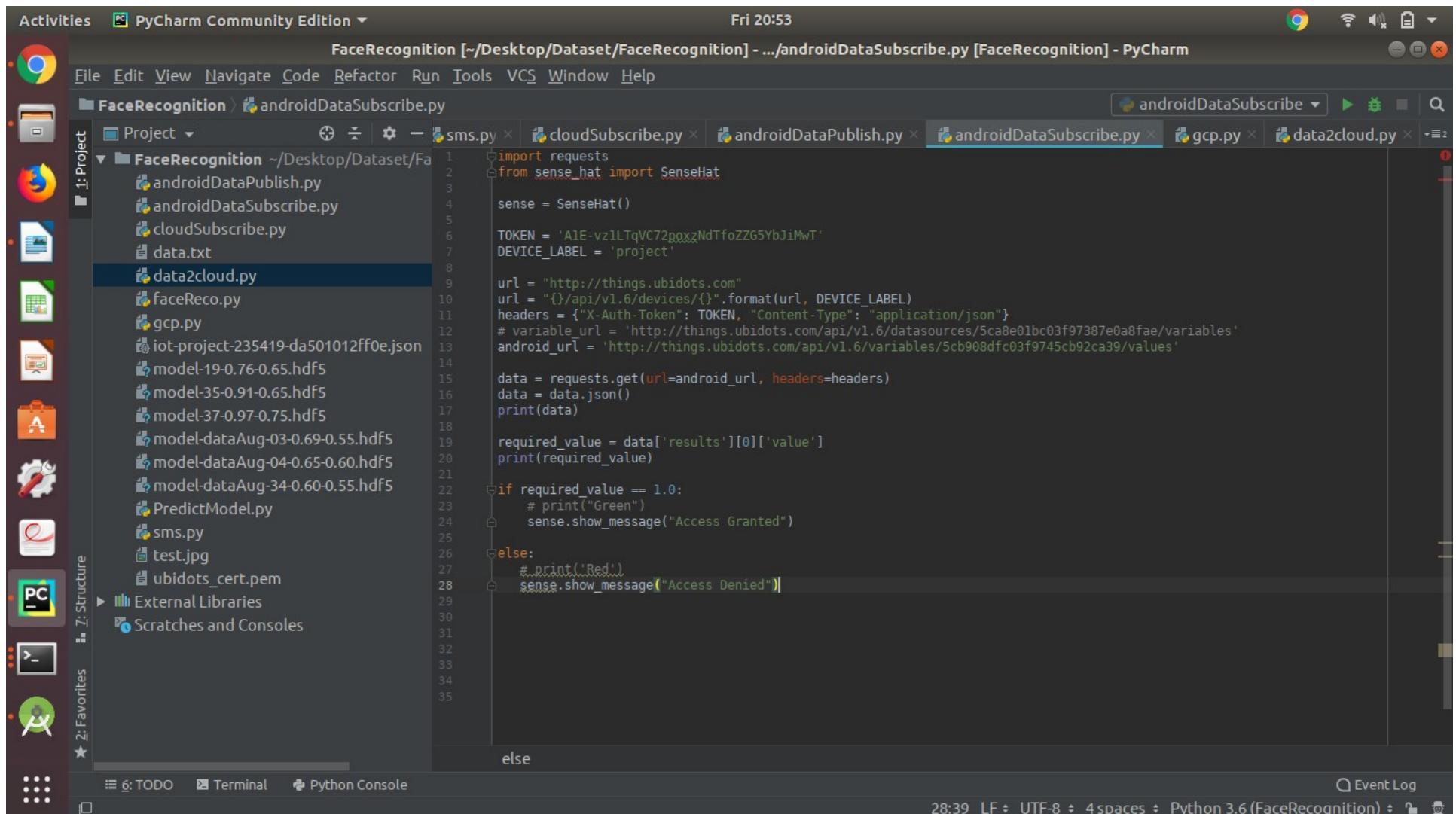
```
import requests
TOKEN = 'A1E-vz1LTqVC72poxxNdTfoZZG5YbJiMwT'
DEVICE_LABEL = 'project'

# Creates the headers for the HTTP requests
url = "http://things.ubidots.com"
url = "{}api/v1.6/devices/{}".format(url, DEVICE_LABEL)
headers = {"X-Auth-Token": TOKEN, "Content-Type": "application/json"}

android_data = open(r'/home/dharani/Android/Sdk/platform-tools/log.txt', mode='r')
data = android_data.read()
required_data = data.split(":")
print(required_data)
print(required_data[-1])

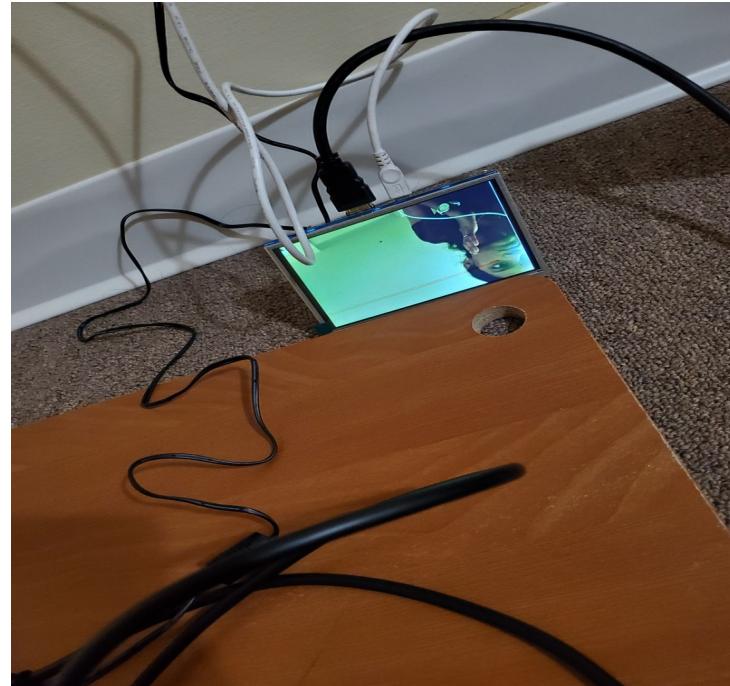
if required_data[-1] == str(' Access Granted\n'):
    payload = {'android': 1}
    status = 400
    attempts = 0
    while status >= 400 and attempts <= 5:
        req = requests.post(url=url, headers=headers, json=payload)
        status = req.status_code
        attempts += 1
        print("Data 1 sent")

else:
    payload = {'android': 2}
    status = 400
    attempts = 0
    while status >= 400 and attempts <= 5:
        req = requests.post(url=url, headers=headers, json=payload)
        status = req.status_code
        attempts += 1
        print("Data 2 sent")
```

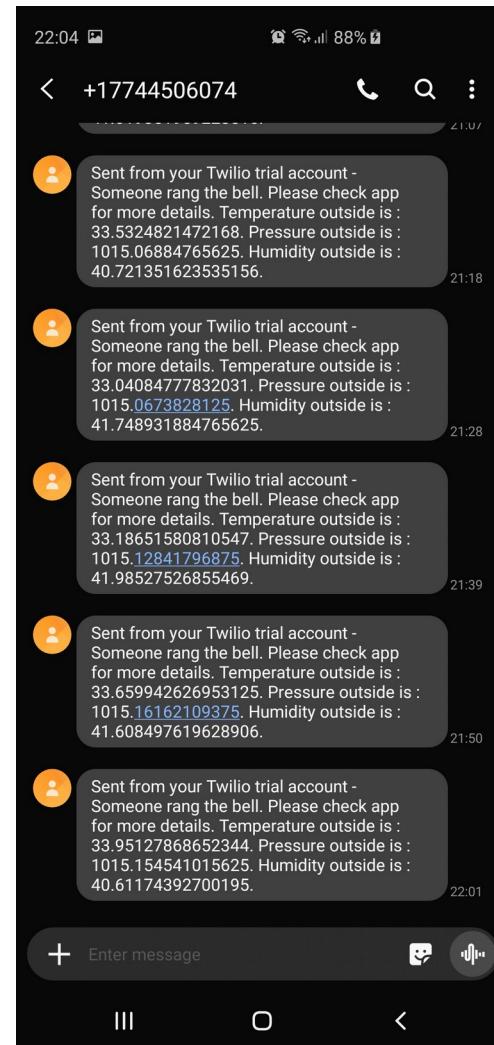
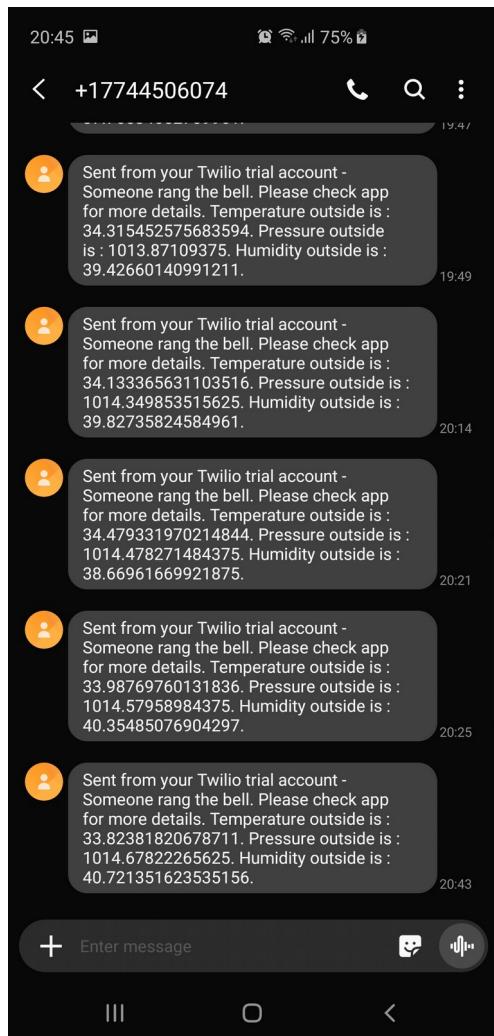


OUTPUT : (2 hours) :

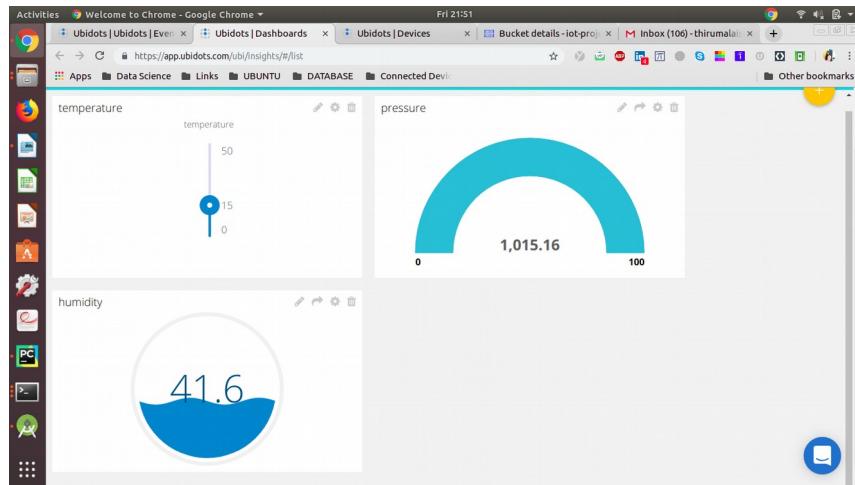
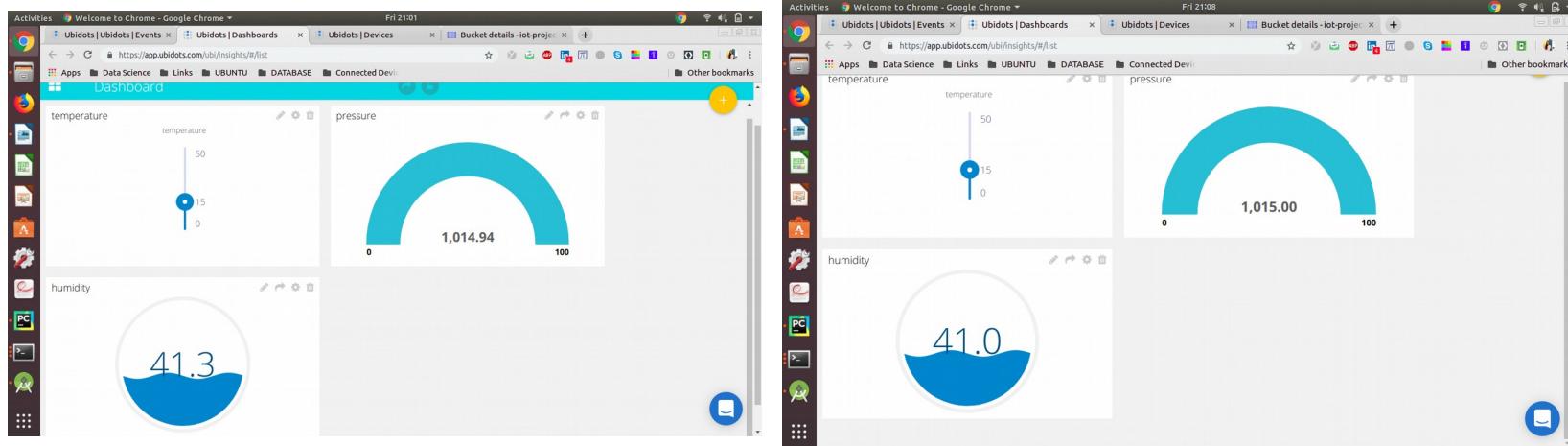
1. Activating Camera module

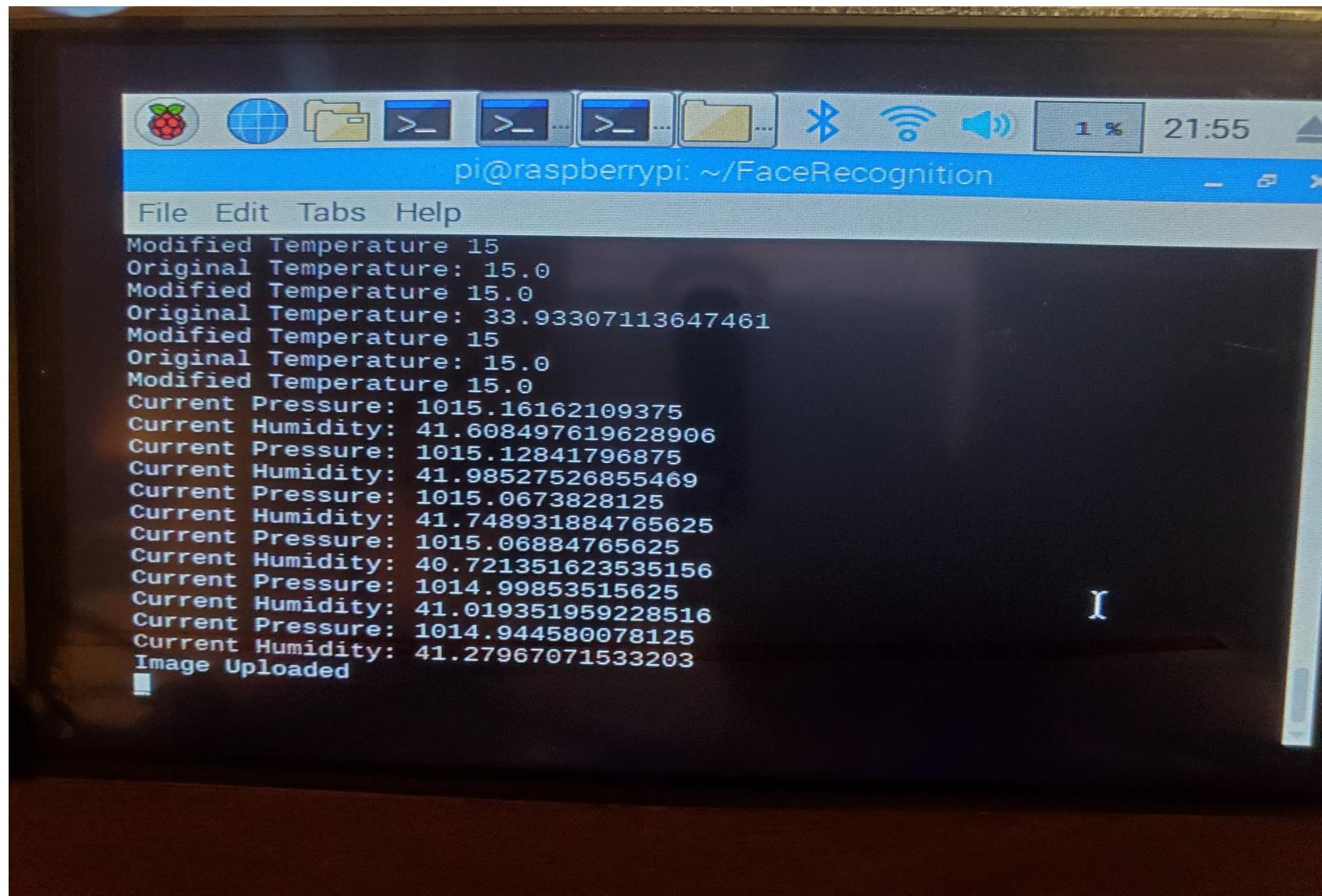


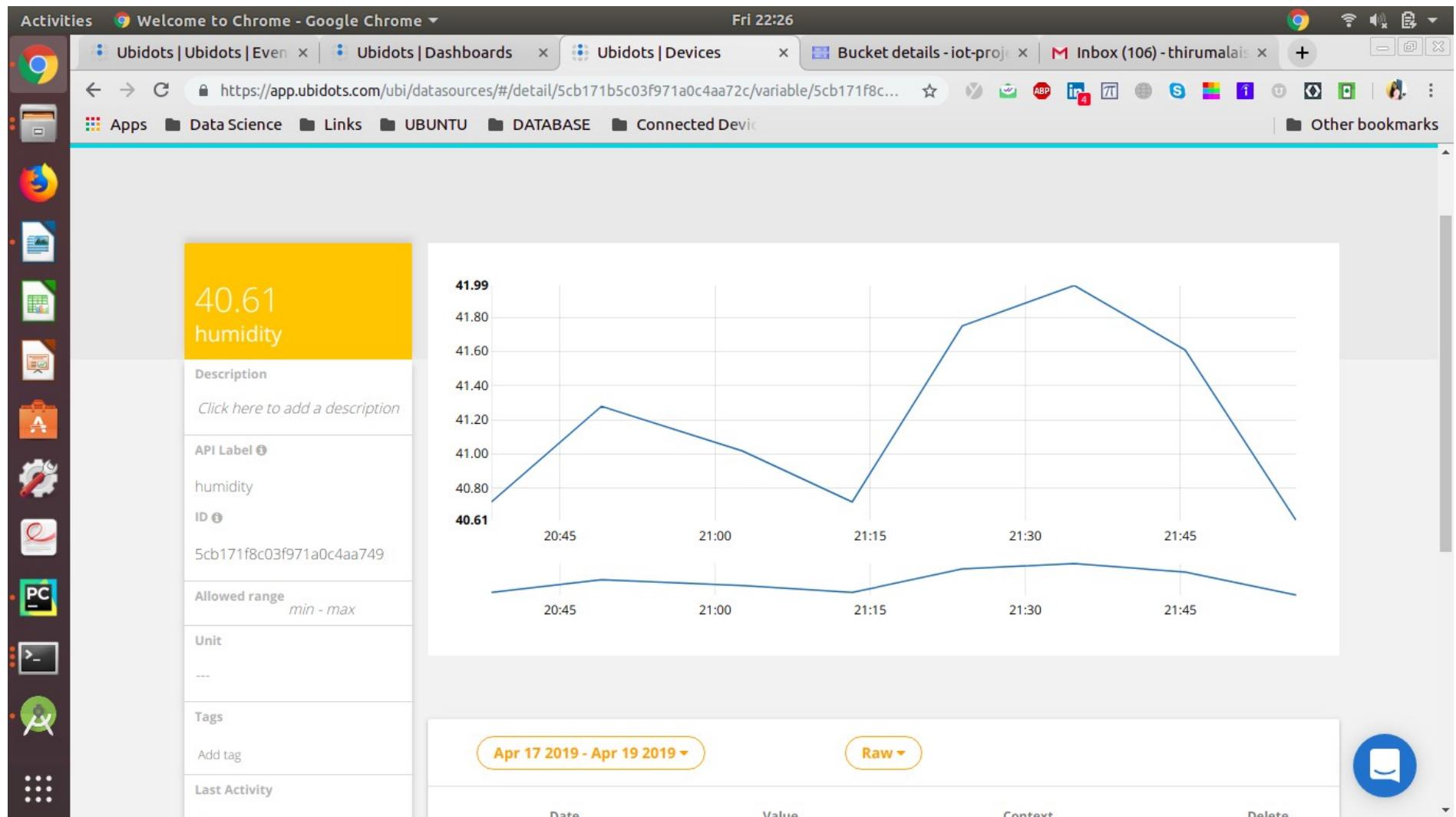
2. Sending Text Message :

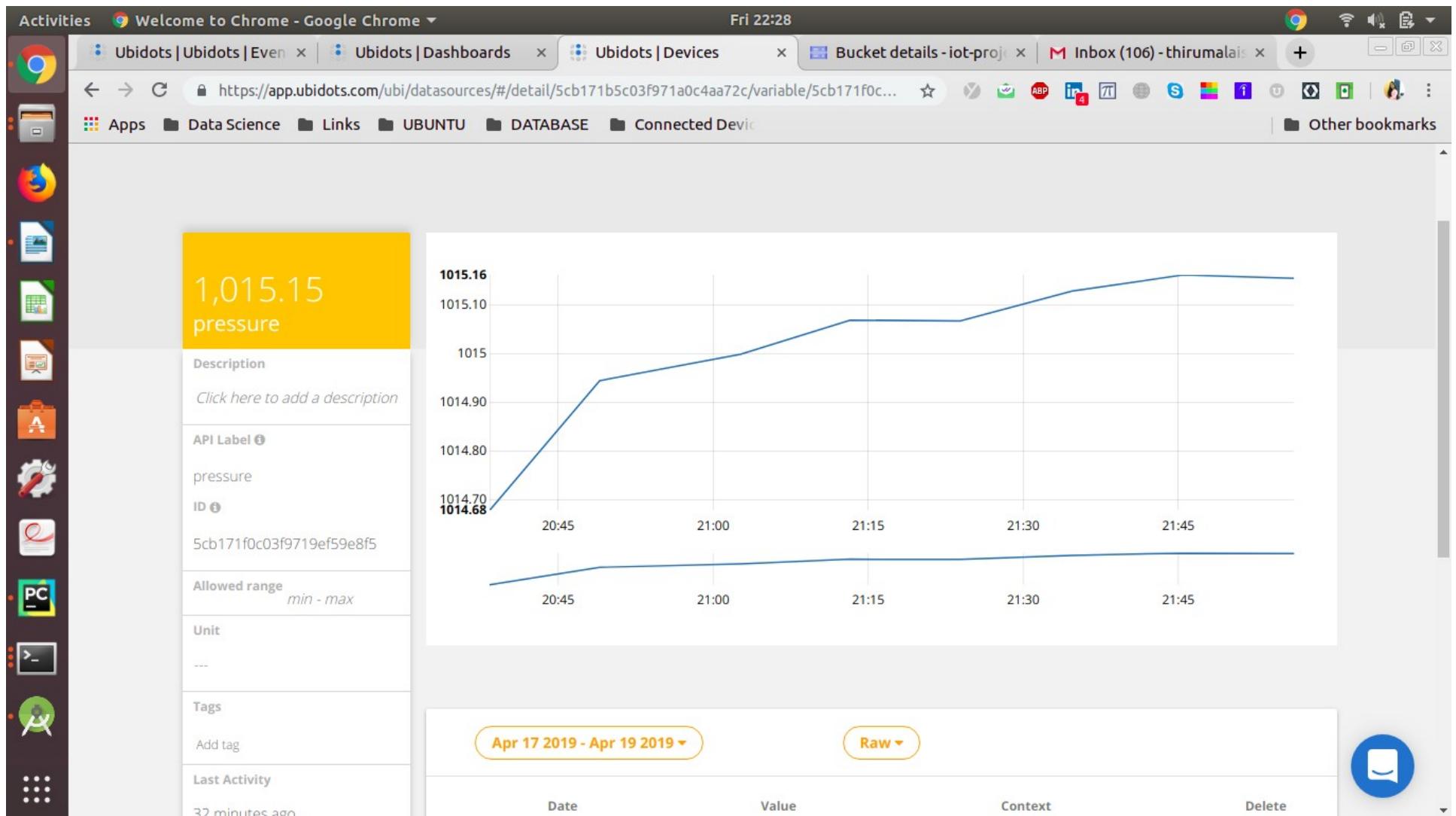


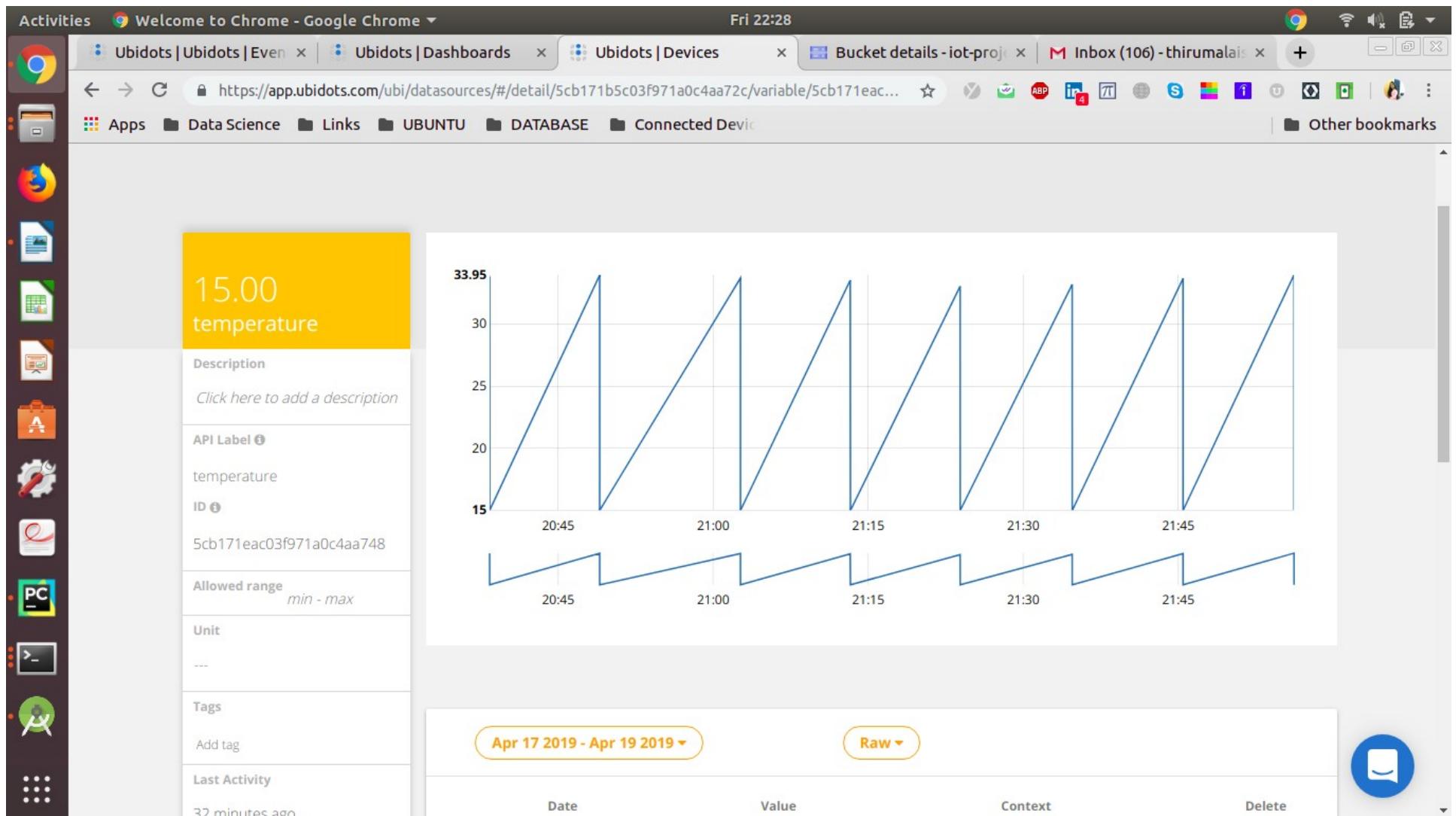
3. Sending data to ubidots and back:









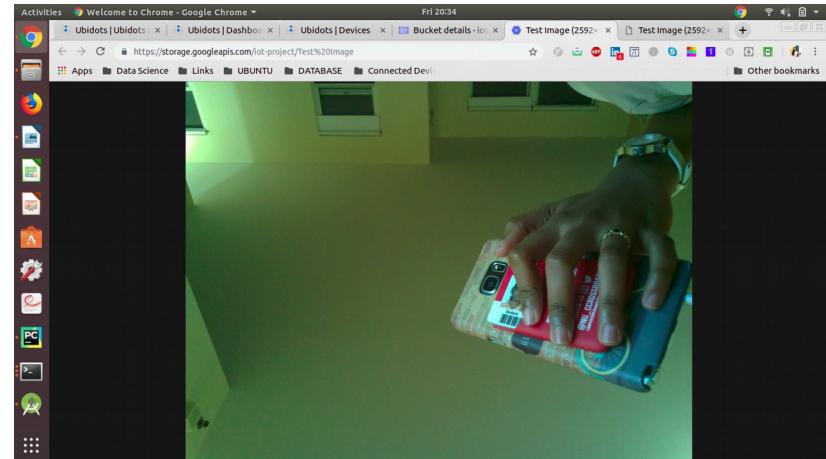


4. Sending image to Google Cloud Storage[f] :

The screenshot shows the Google Cloud Platform Storage interface. On the left, there's a sidebar with icons for Storage, Browser, Transfer, Transfer Appliance, and Settings. The main area is titled 'Bucket details' for 'iot-project'. It has tabs for Objects, Overview, Permissions, and Bucket Lock. Below these are buttons for Upload files, Upload folder, Create folder, Manage holds, and Delete. A search bar labeled 'Filter by prefix...' is present. A table lists objects in the bucket:

Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiration date	Holds
Test Image	2.6 MB	image/jpeg	Multi-Regional	4/19/19, 8:43:57 PM	Public	Google-managed key	-	None

A message 'Bucket refreshed.' is displayed at the bottom left.



Activities Fri 21:34

Ubidots | Ubidots x | Ubidots | Dashboard x | Ubidots | Devices x | Bucket details - io... x | Test Image (2592 x | Drafts (40) - thiru...

https://console.cloud.google.com/storage/browser/iot-project?project=iot-project-235419&authu...

Apps Data Science Links UBUNTU DATABASE Connected Devic...

Google Cloud Platform iot-project

Storage Bucket details EDIT BUCKET REFRESH BUCKET

iot-project

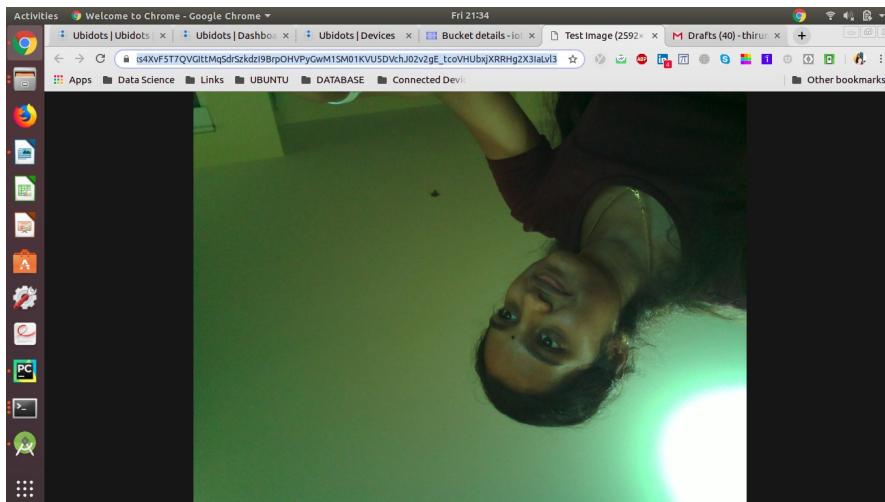
Objects Overview Permissions Bucket Lock

Upload files Upload folder Create folder Manage holds Delete

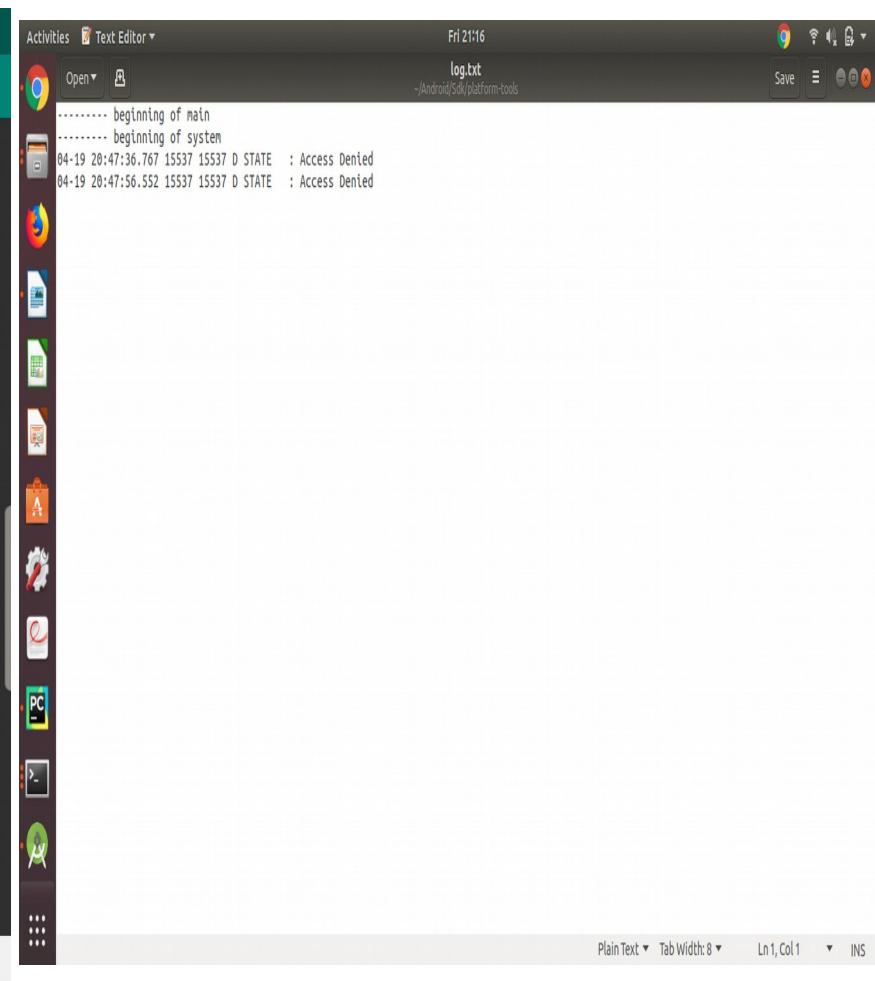
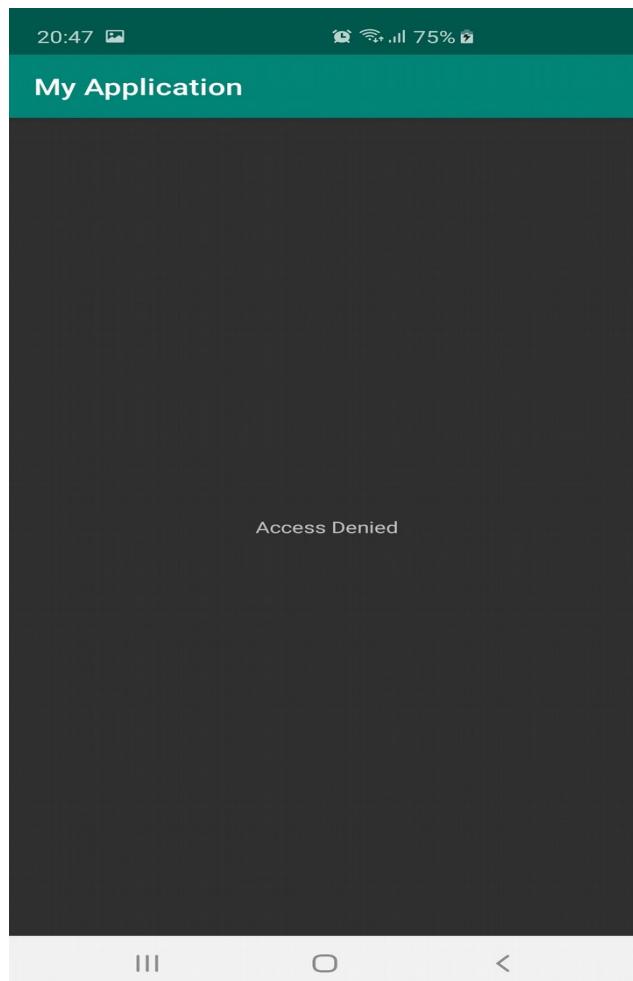
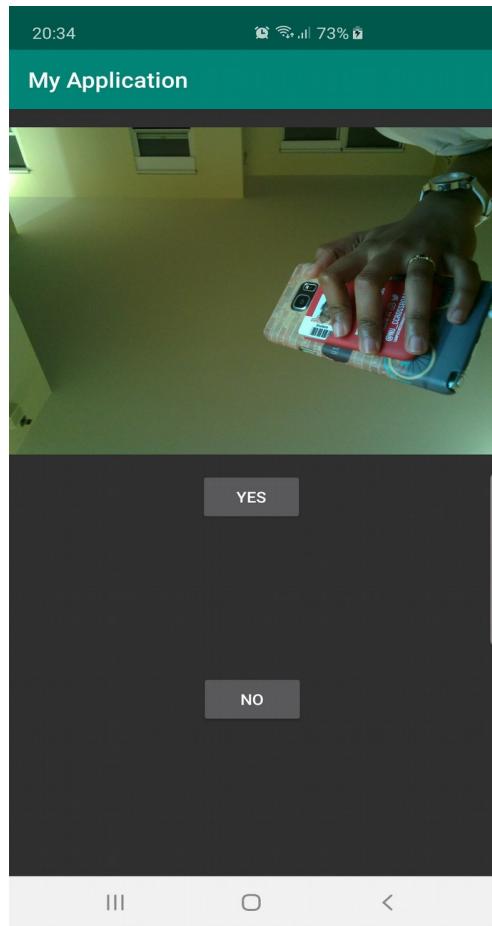
Filter by prefix...

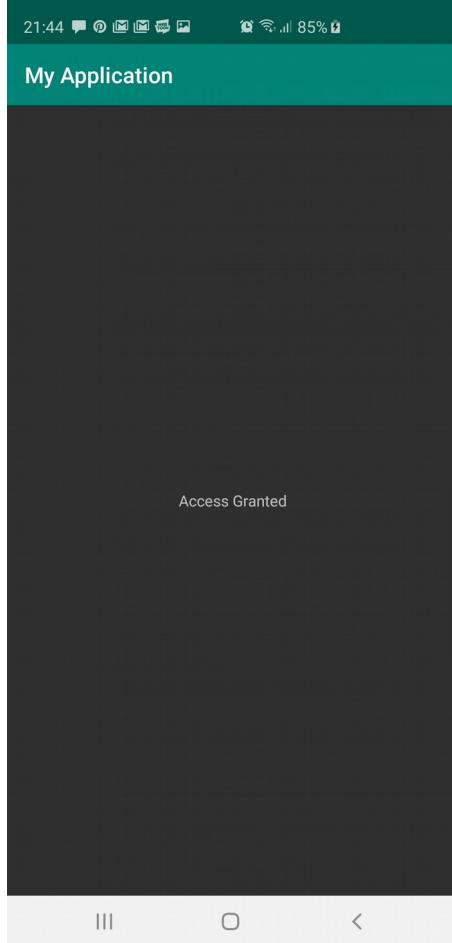
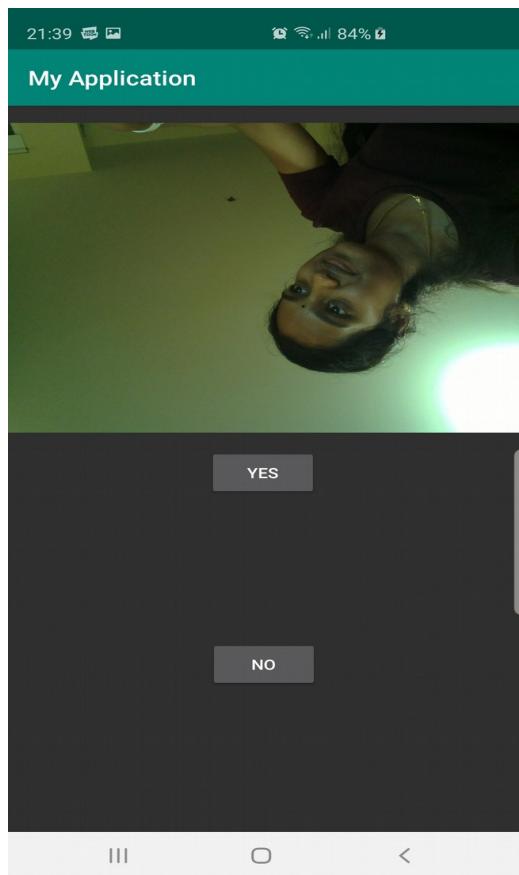
Buckets / iot-project

Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiration date	Holds
Test Image	2.59 MB	image/jpeg	Multi-Regional	4/19/19, 9:29:43 PM UTC-4	Public	Google-managed key	-	None



5. Android App accessing the data [b]:

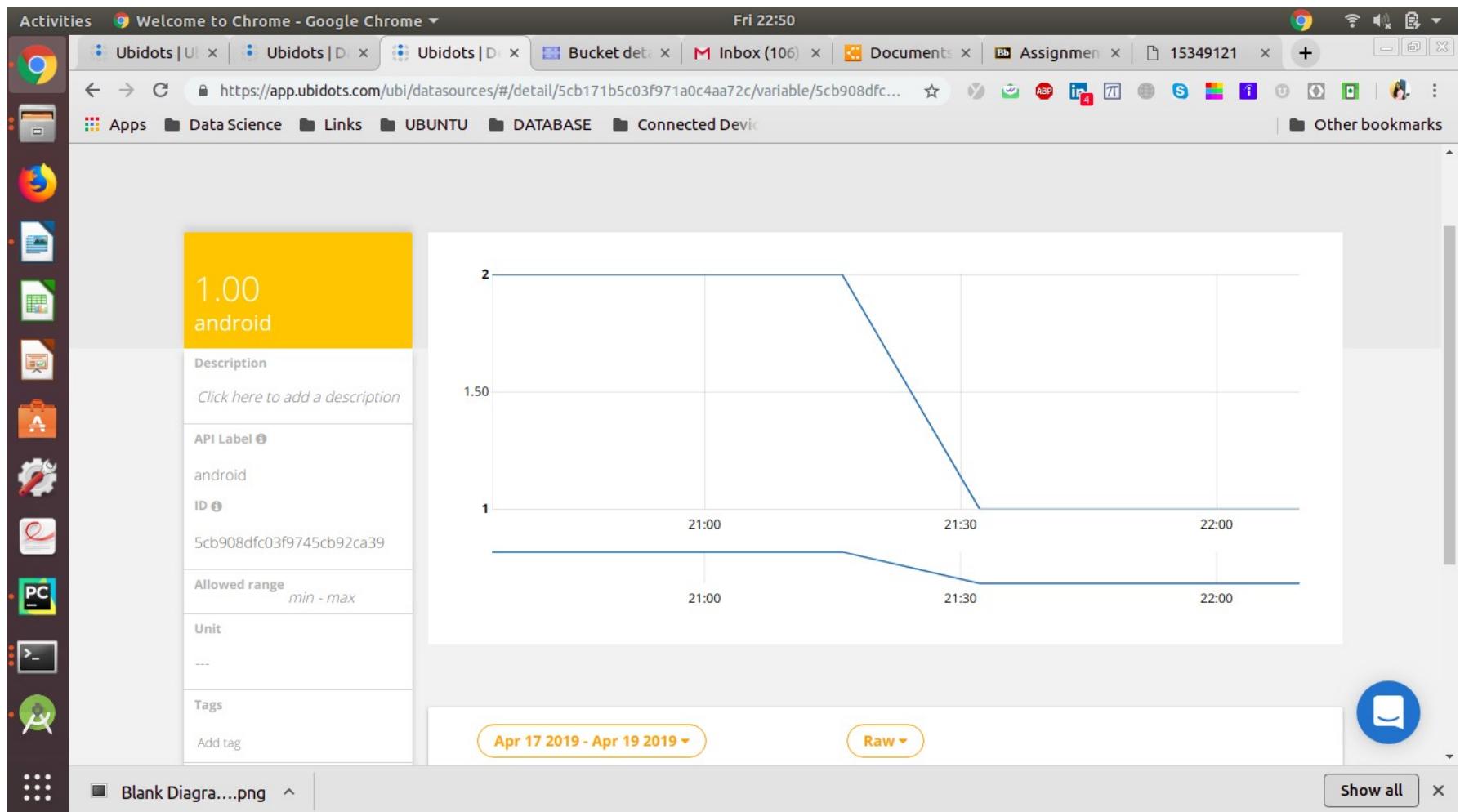




A screenshot of a terminal window titled "Text Editor". The window shows log output from a file named "log.txt" located at "/Android/Sdk/platform-tools". The log contains the following entries:

```
..... beginning of main
04-19 21:15:55.334 21780 21780 D STATE : Access Denied
..... beginning of system
04-19 21:31:57.202 24058 24058 D STATE : Access Granted
04-19 21:32:04.750 24058 24058 D STATE : Access Granted
```

The terminal window also shows a vertical list of icons on the left and various settings at the bottom.



6. Sense Hat matrix Displaying images [a]:

https://youtu.be/xa_9HWD0yGU

REFERENCES :

[a] projects.raspberrypi.org

[b] developer.android.com

[c] sites.google.com

[d] wildanmsyah.wordpress.com

[e] projects.raspberrypi.org

[f] console.cloud.google.com