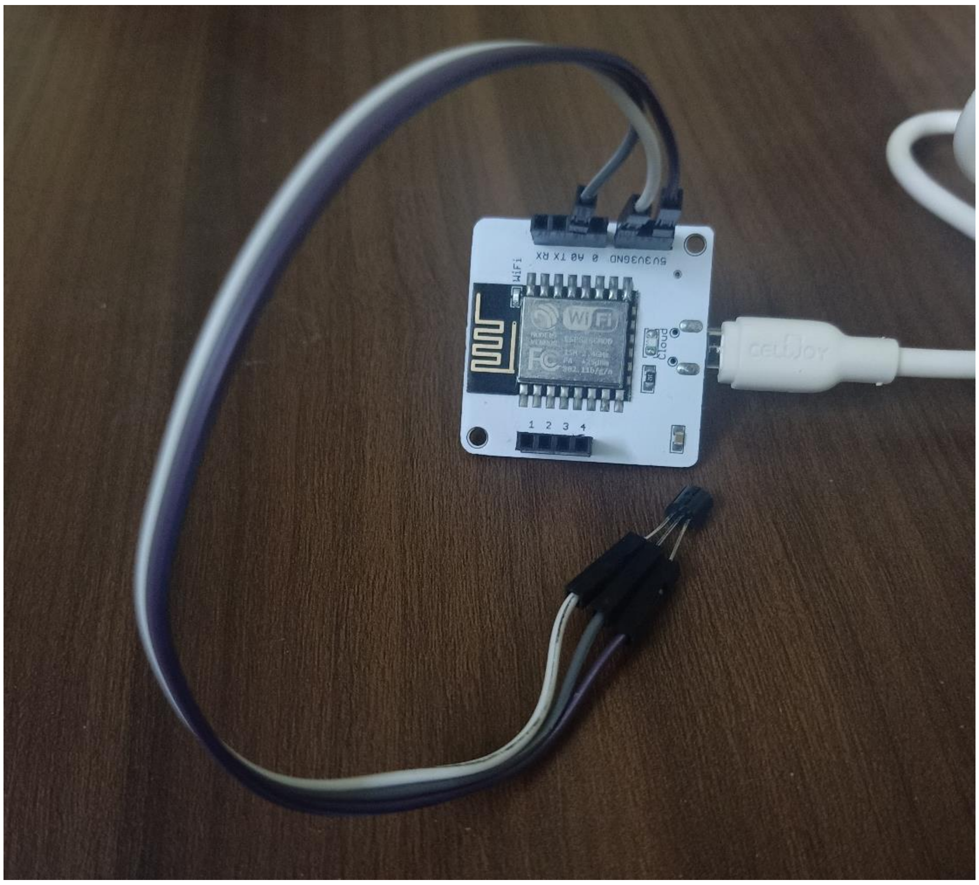
BOLT IOT Capstone Project

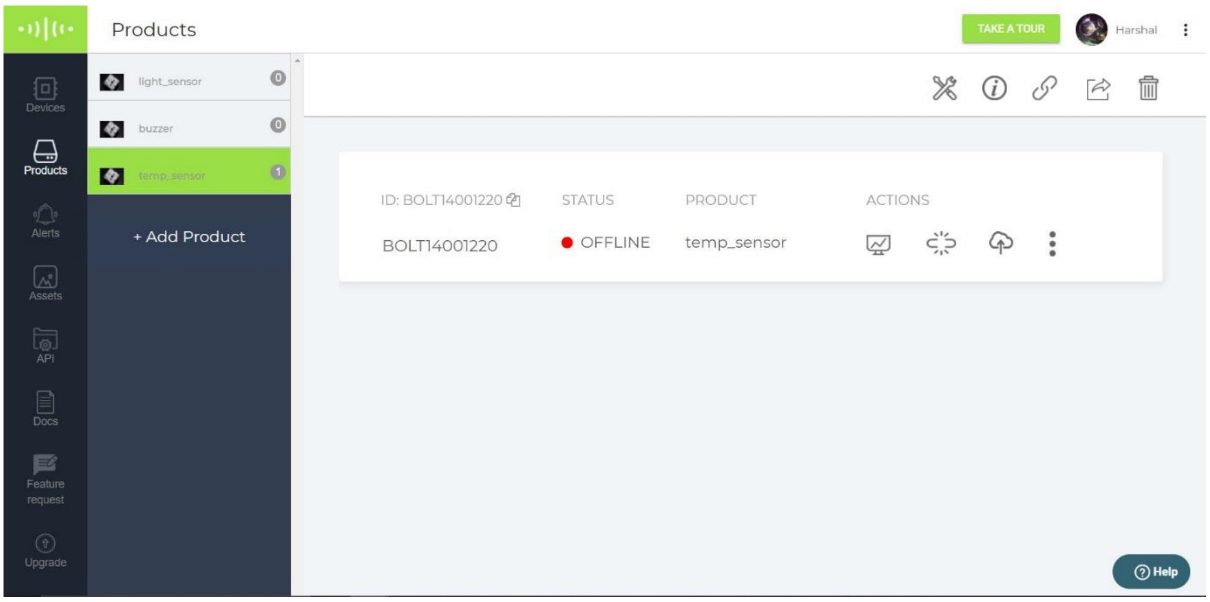
Temperature Anomaly Detection and Alert System

Name: Harshal Abhyankar

a) Build the circuit for temperature monitoring system, using the Bolt and LM35 sensor.

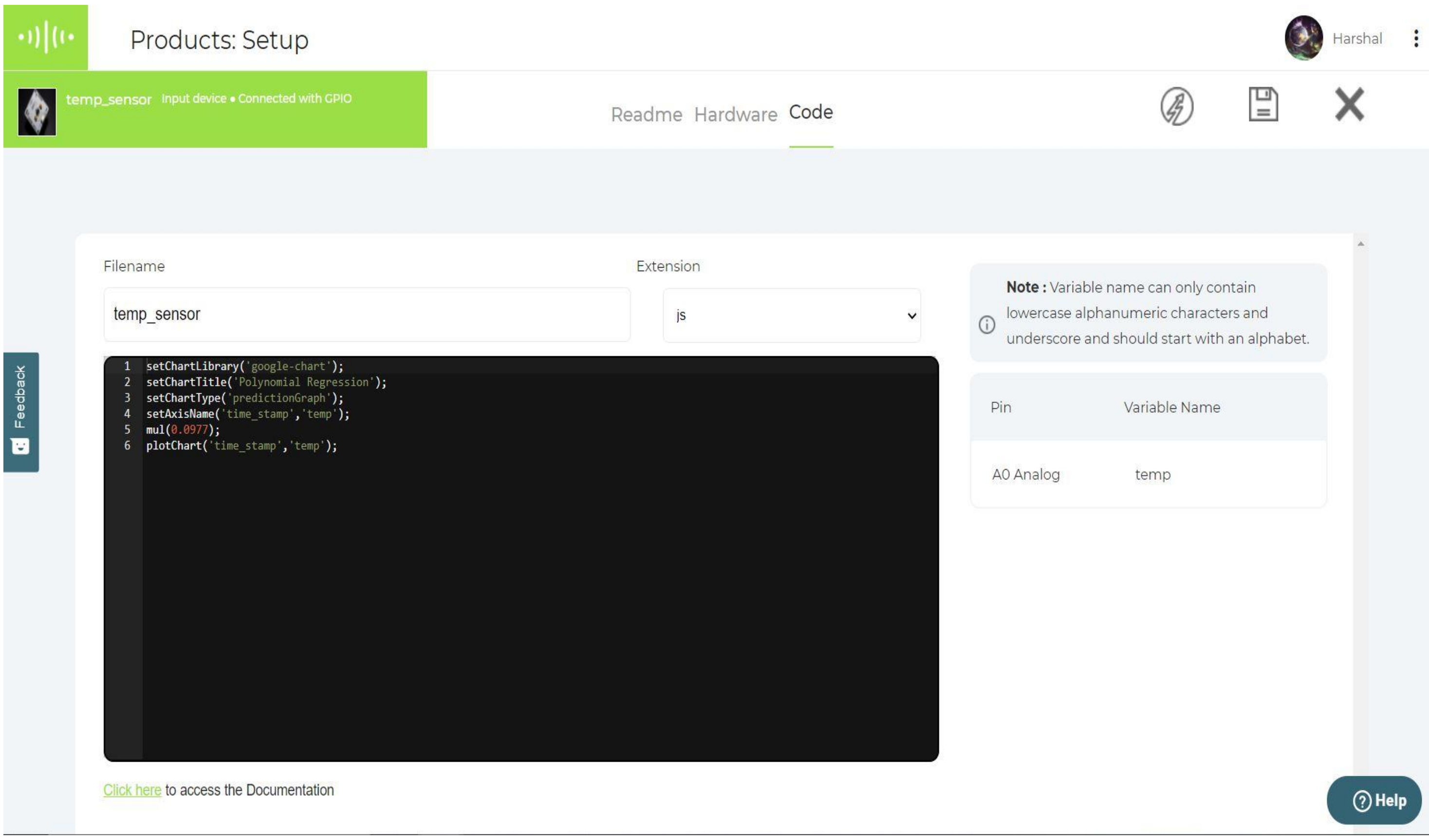


b) Create a product on the Bolt Cloud, to monitor the data from the LM35, and link it to your Bolt.



ID: BOLT14001220	STATUS	PRODUCT	ACTIONS
BOLT14001220	● OFFLINE	temp_sensor	

c) Write the product code, required to run the polynomial regression algorithm on the data sent by the Bolt.



Filename: temp_sensor

Extension: js

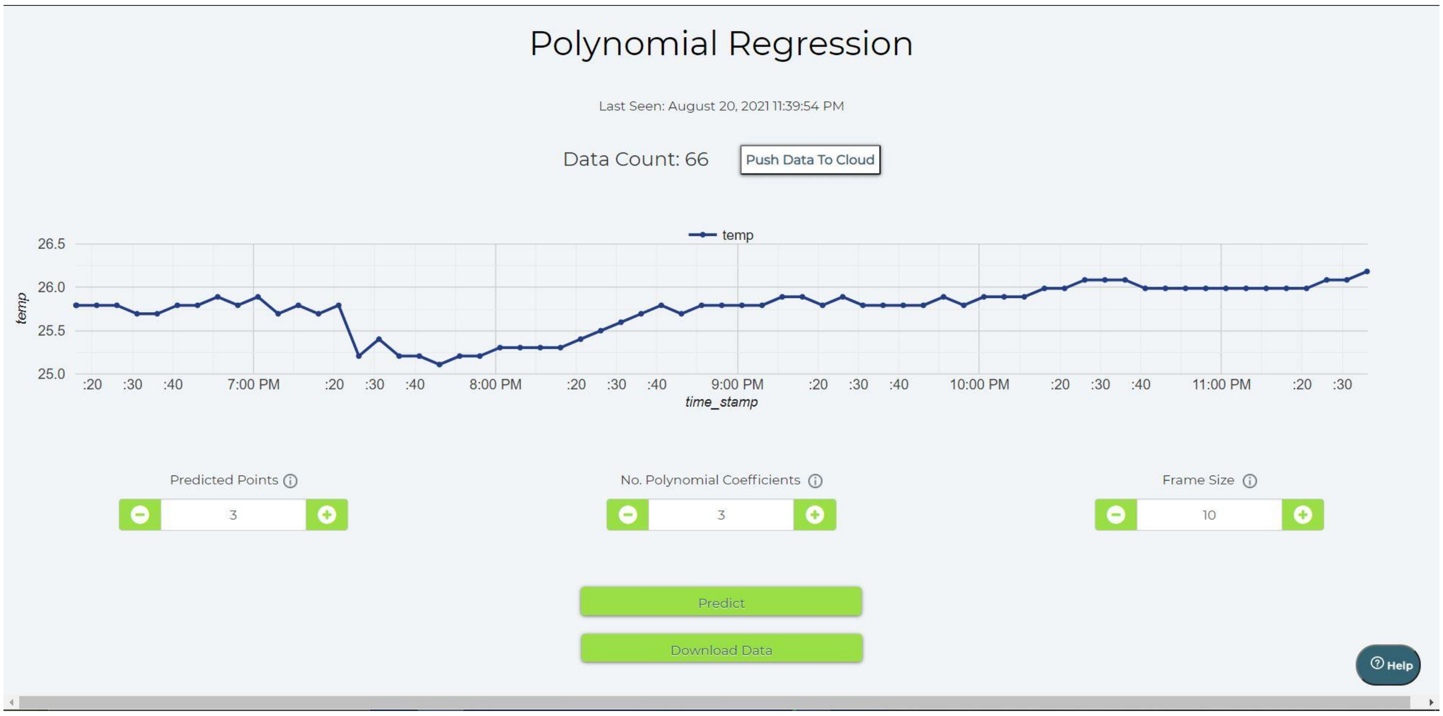
```
setChartLibrary('google-chart');
setChartTitle('Polynomial Regression');
setChartType('predictionGraph');
setAxisName('time_stamp','temp');
mul(0.0977);
plotChart('time_stamp','temp');
```

Click here to access the Documentation

Note : Variable name can only contain lowercase alphanumeric characters and underscore and should start with an alphabet.

Pin	Variable Name
A0 Analog	temp

d) Keep the temperature monitoring circuit inside your fridge with the door of the fridge closed, and let the system record the temperature readings for about 2 hours.



Polynomial Regression

Last Seen: August 20, 2021 11:39:54 PM

Data Count: 66

Predicted Points: 3

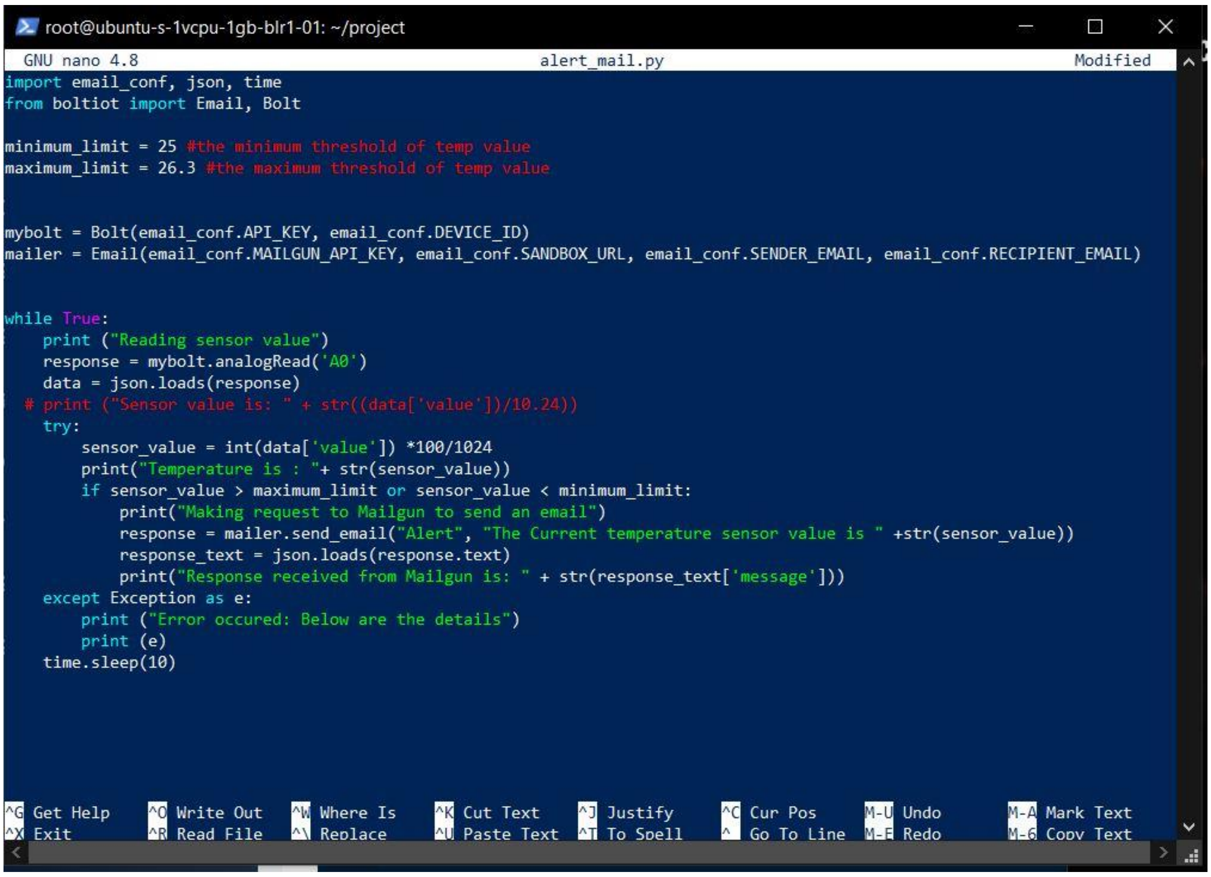
No. Polynomial Coefficients: 3

Frame Size: 10

e) Using the reading that you received in the 2 hours, set boundaries for the temperature within the fridge.

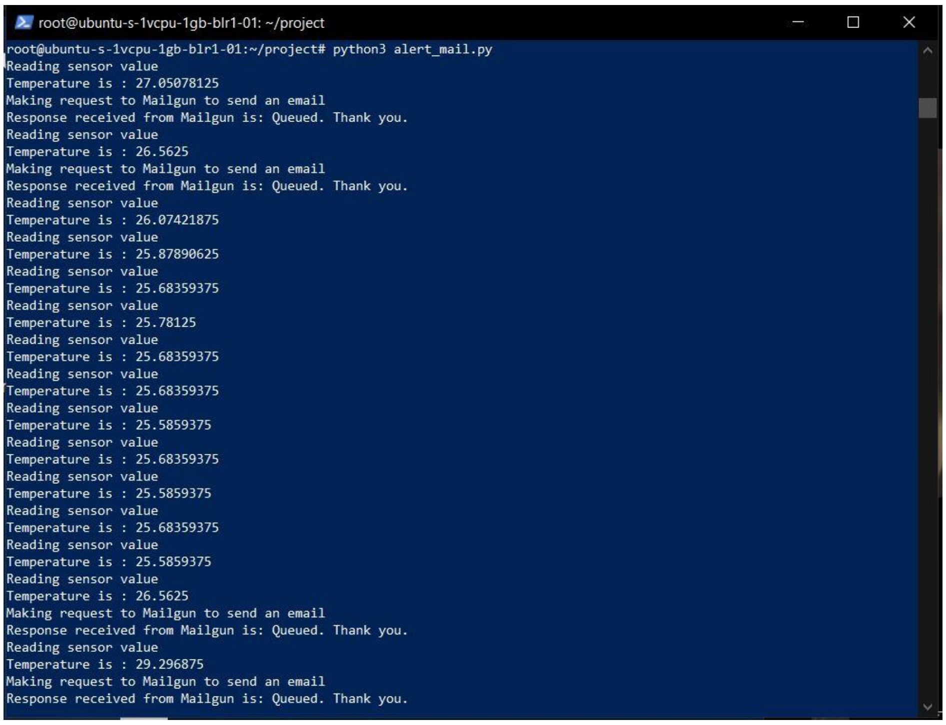
The boundaries set are 25 degrees and 26.3 degrees Celsius. (I recorded room temperature since I didn't have a battery to run the module without direct connection and simulated rise in temperature using heater.)

f) Write a python code that will fetch the temperature data, every 10 seconds, and send out an email alert, if the temperature goes beyond the temperature thresholds you decided on in objective 'e'.



root@ubuntu-s-1vcpu-1gb-blr1-01: ~/project

```
  GNU nano 4.8                              alert_mail.py                              Modified
import email_conf, json, time
from boltiot import Email, Bolt

minimum_limit = 25 #the minimum threshold of temp value
maximum_limit = 26.3 #the maximum threshold of temp value


mybolt = Bolt(email_conf.API_KEY, email_conf.DEVICE_ID)
mailer = Email(email_conf.MAILGUN_API_KEY, email_conf.SANDBOX_URL, email_conf.SENDER_EMAIL, email_conf.RECIPIENT_EMAIL)


while True:
    print ("Reading sensor value")
    response = mybolt.analogRead('A0')
    data = json.loads(response)
  # print ("Sensor value is: " + str((data['value'])/10.24))
    try:
        sensor_value = int(data['value']) *100/1024
        print("Temperature is : "+ str(sensor_value))
        if sensor_value > maximum_limit or sensor_value < minimum_limit:
            print("Making request to Mailgun to send an email")
            response = mailer.send_email("Alert", "The Current temperature sensor value is " +str(sensor_value))
            response_text = json.loads(response.text)
            print("Response received from Mailgun is: " + str(response_text['message']))
    except Exception as e:
        print ("Error occured: Below are the details")
        print (e)
    time.sleep(10)




^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo       M-A Mark Text
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line  M-E Redo       M-6 Copy Text
```

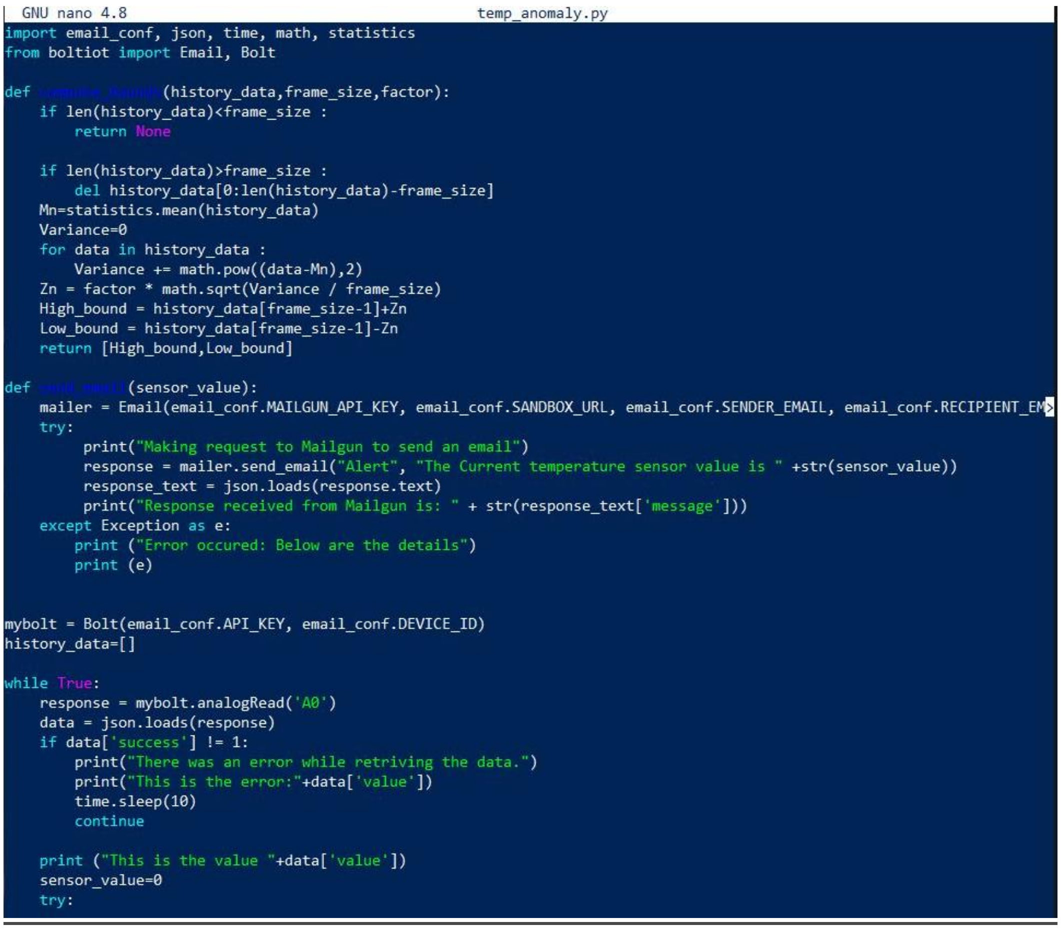


root@ubuntu-s-1vcpu-1gb-blr1-01: ~/project

```
root@ubuntu-s-1vcpu-1gb-blr1-01:~/project# python3 alert_mail.py
Reading sensor value
Temperature is : 27.05078125
Making request to Mailgun to send an email
Response received from Mailgun is: Queued. Thank you.
Reading sensor value
Temperature is : 26.5625
Making request to Mailgun to send an email
Response received from Mailgun is: Queued. Thank you.
Reading sensor value
Temperature is : 26.07421875
Reading sensor value
Temperature is : 25.87890625
Reading sensor value
Temperature is : 25.68359375
Reading sensor value
Temperature is : 25.78125
Reading sensor value
Temperature is : 25.68359375
Reading sensor value
Temperature is : 25.68359375
Reading sensor value
Temperature is : 25.5859375
Reading sensor value
Temperature is : 25.68359375
Reading sensor value
Temperature is : 25.5859375
Reading sensor value
Temperature is : 25.68359375
Reading sensor value
Temperature is : 25.5859375
Reading sensor value
Temperature is : 26.5625
Making request to Mailgun to send an email
Response received from Mailgun is: Queued. Thank you.
Reading sensor value
Temperature is : 29.296875
Making request to Mailgun to send an email
Response received from Mailgun is: Queued. Thank you.
```

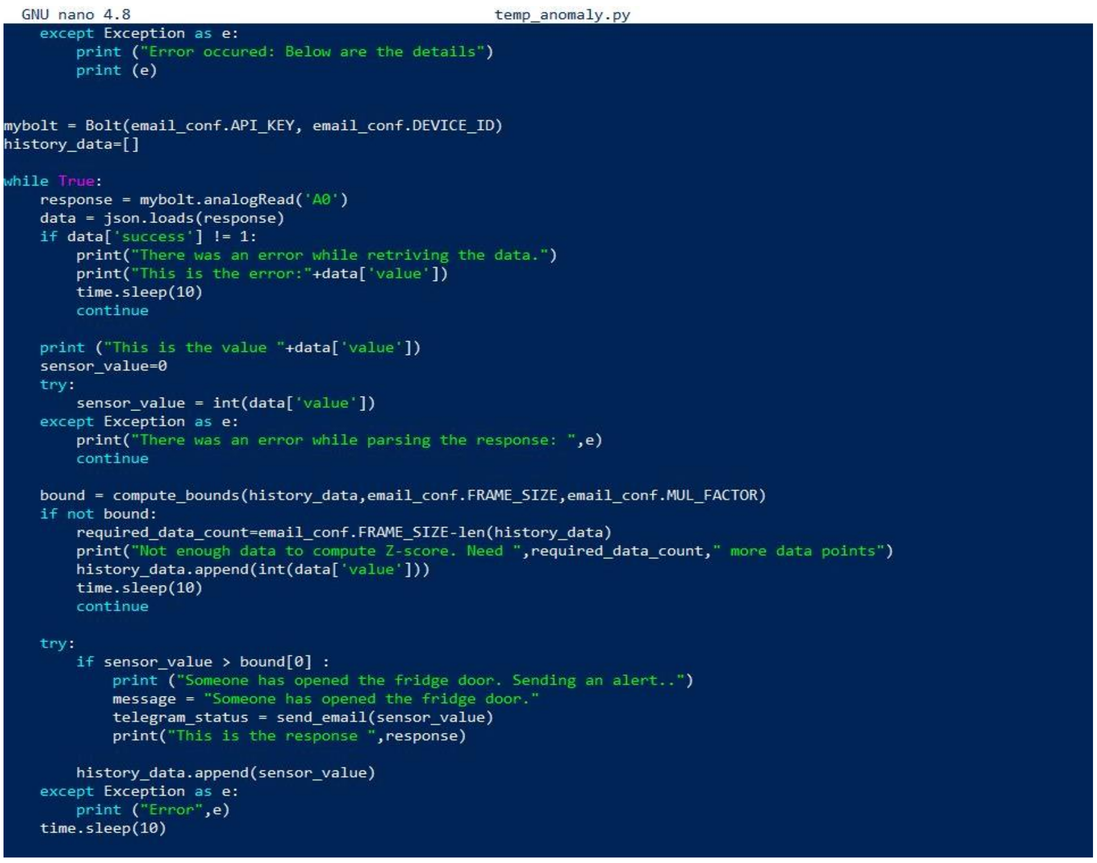
g) Modify the python code, to also do a Z-score analysis and print the line “Someone has opened the fridge door” when an anomaly is detected.

Part 1 of code:



```
  GNU nano 4.8                                            temp_anomaly.py
import email_conf, json, time, math, statistics
from boltiot import Email, Bolt

def compute_bounds(history_data,frame_size,factor):
    if len(history_data)<frame_size :
        return None

    if len(history_data)>frame_size :
        del history_data[0:len(history_data)-frame_size]
    Mn=statistics.mean(history_data)
    Variance=0
    for data in history_data :
        Variance += math.pow((data-Mn),2)
    Zn = factor * math.sqrt(Variance / frame_size)
    High_bound = history_data[frame_size-1]+Zn
    Low_bound = history_data[frame_size-1]-Zn
    return [High_bound,Low_bound]

def send_email(sensor_value):
    mailer = Email(email_conf.MAILGUN_API_KEY, email_conf.SANDBOX_URL, email_conf.SENDER_EMAIL, email_conf.RECIPIENT_EM>
    try:
         print("Making request to Mailgun to send an email")
         response = mailer.send_email("Alert", "The Current temperature sensor value is " +str(sensor_value))
         response_text = json.loads(response.text)
         print("Response received from Mailgun is: " + str(response_text['message']))
    except Exception as e:
        print ("Error occured: Below are the details")
        print (e)


mybolt = Bolt(email_conf.API_KEY, email_conf.DEVICE_ID)
history_data=[]

while True:
    response = mybolt.analogRead('A0')
    data = json.loads(response)
    if data['success'] != 1:
        print("There was an error while retriving the data.")
        print("This is the error:"+data['value'])
        time.sleep(10)
        continue

    print ("This is the value "+data['value'])
    sensor_value=0
    try:
```

Part 2 of code:

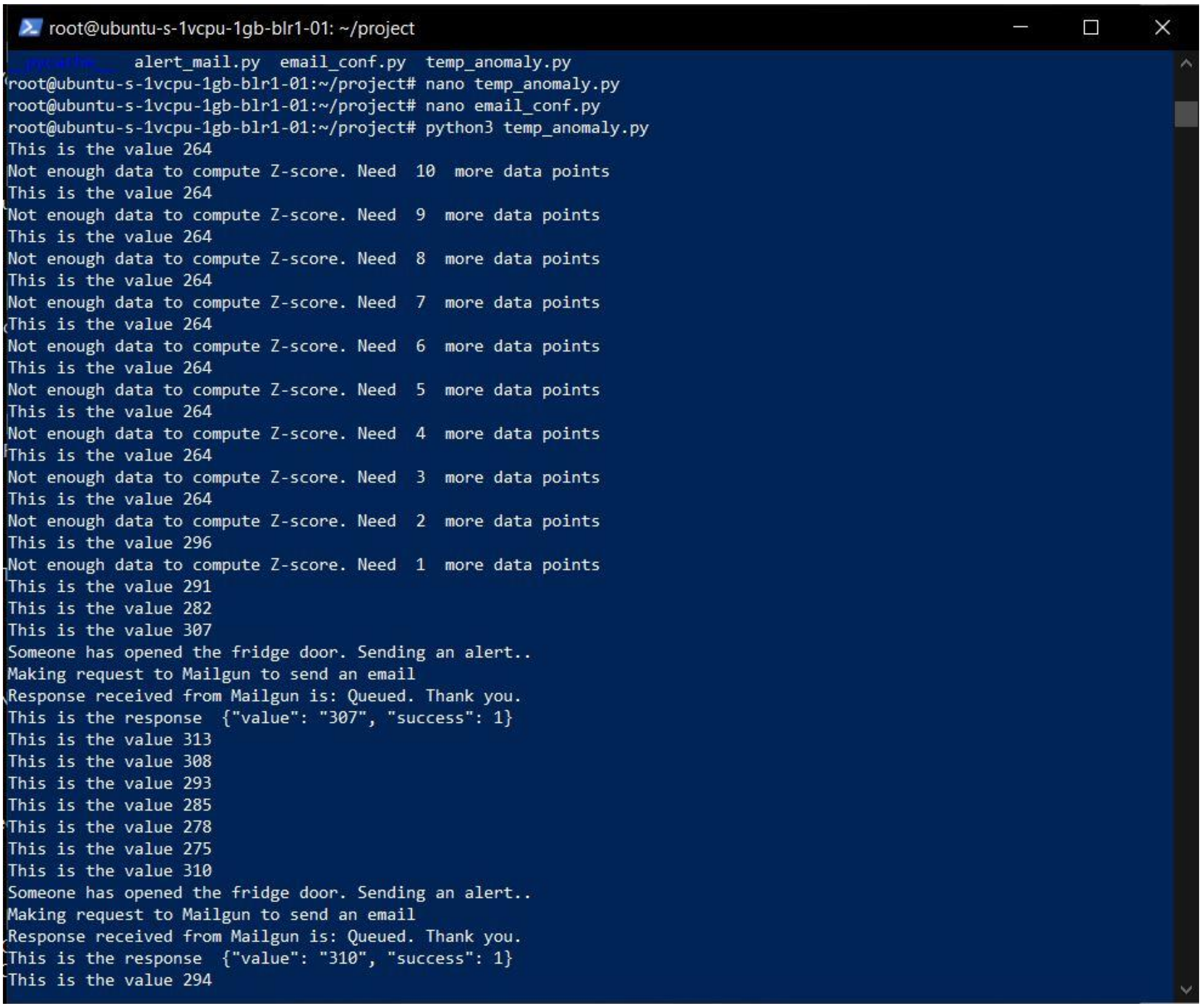


```
  GNU nano 4.8                                            temp_anomaly.py
    except Exception as e:
        print ("Error occured: Below are the details")
        print (e)


mybolt = Bolt(email_conf.API_KEY, email_conf.DEVICE_ID)
history_data=[]

while True:
    response = mybolt.analogRead('A0')
    data = json.loads(response)
    if data['success'] != 1:
        print("There was an error while retriving the data.")
        print("This is the error:"+data['value'])
        time.sleep(10)
        continue

    print ("This is the value "+data['value'])
    sensor_value=0
    try:
        sensor_value = int(data['value'])
    except Exception as e:
        print("There was an error while parsing the response: ",e)
        continue

    bound = compute_bounds(history_data,email_conf.FRAME_SIZE,email_conf.MUL_FACTOR)
    if not bound:
        required_data_count=email_conf.FRAME_SIZE-len(history_data)
        print("Not enough data to compute Z-score. Need ",required_data_count," more data points")
        history_data.append(int(data['value']))
        time.sleep(10)
        continue

    try:
        if sensor_value > bound[0] :
            print ("Someone has opened the fridge door. Sending an alert..")
            message = "Someone has opened the fridge door."
            telegram_status = send_email(sensor_value)
            print("This is the response ",response)

        history_data.append(sensor_value)
    except Exception as e:
        print ("Error",e)
    time.sleep(10)
```

h) Tune the Z-score analysis code, such that, it detects an anomaly when someone opens the door of the fridge.

I tuned the Z-score analysis code by reducing the MUL_FACTOR value to 1 to make it more sensitive to temperature change.

Results:



root@ubuntu-s-1vcpu-1gb-blr1-01: ~/project

```
__pycache__   alert_mail.py  email_conf.py  temp_anomaly.py
root@ubuntu-s-1vcpu-1gb-blr1-01:~/project# nano temp_anomaly.py
root@ubuntu-s-1vcpu-1gb-blr1-01:~/project# nano email_conf.py
root@ubuntu-s-1vcpu-1gb-blr1-01:~/project# python3 temp_anomaly.py
This is the value 264
Not enough data to compute Z-score. Need  10  more data points
This is the value 264
Not enough data to compute Z-score. Need  9  more data points
This is the value 264
Not enough data to compute Z-score. Need  8  more data points
This is the value 264
Not enough data to compute Z-score. Need  7  more data points
This is the value 264
Not enough data to compute Z-score. Need  6  more data points
This is the value 264
Not enough data to compute Z-score. Need  5  more data points
This is the value 264
Not enough data to compute Z-score. Need  4  more data points
This is the value 264
Not enough data to compute Z-score. Need  3  more data points
This is the value 264
Not enough data to compute Z-score. Need  2  more data points
This is the value 296
Not enough data to compute Z-score. Need  1  more data points
This is the value 291
This is the value 282
This is the value 307
Someone has opened the fridge door. Sending an alert..
Making request to Mailgun to send an email
Response received from Mailgun is: Queued. Thank you.
This is the response  {"value": "307", "success": 1}
This is the value 313
This is the value 308
This is the value 293
This is the value 285
This is the value 278
This is the value 275
This is the value 310
Someone has opened the fridge door. Sending an alert..
Making request to Mailgun to send an email
Response received from Mailgun is: Queued. Thank you.
This is the response  {"value": "310", "success": 1}
This is the value 294
```