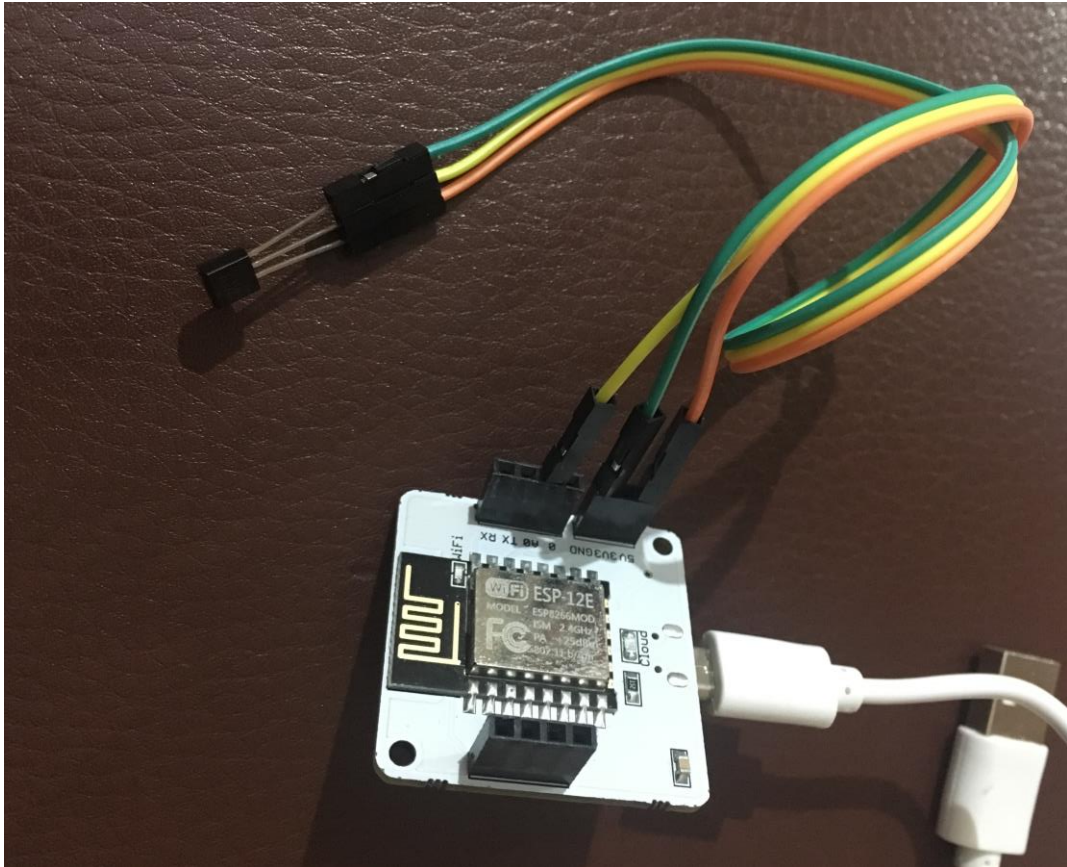


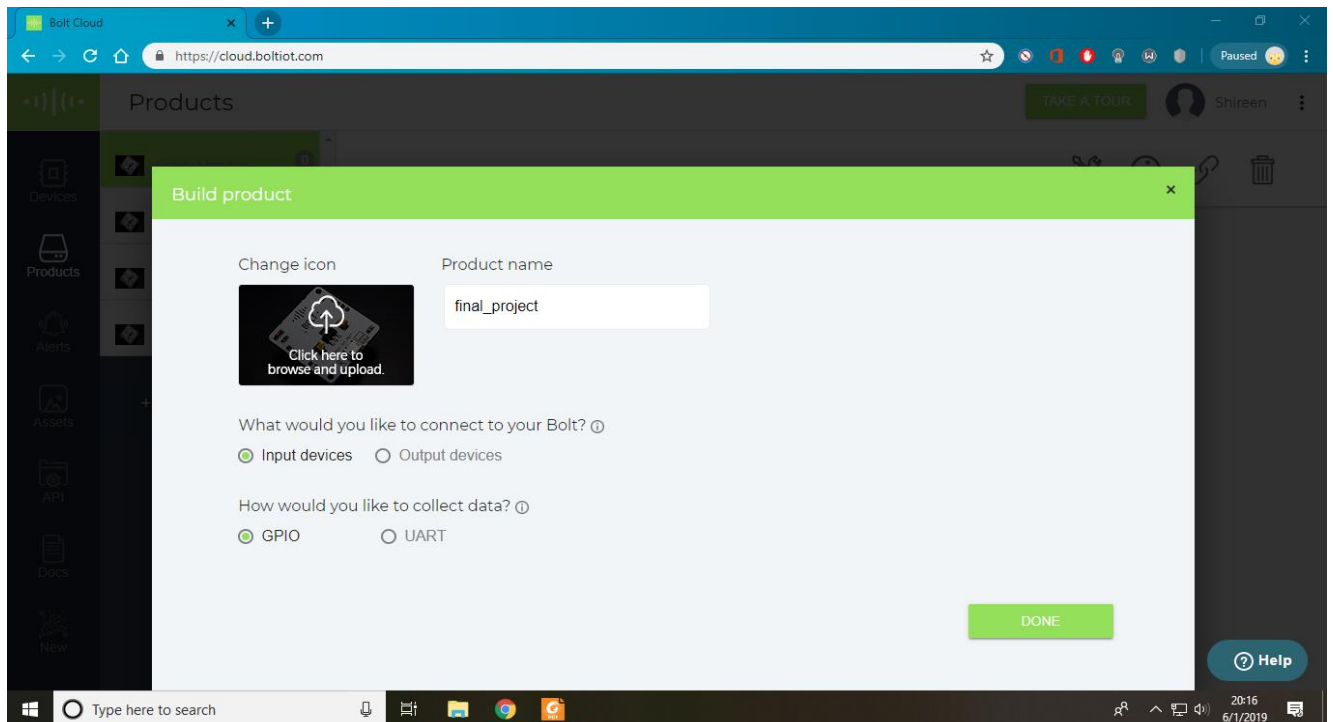
CAPSTONE PROJECT:

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

Pin1 is connected to 5V supply. Pin2(Output) is connected to 'A0'.
Pin3 is connected to GND pin of the Bolt Wifi module.



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



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

Step 1: Assemble the circuit using Bolt hardware module as per your requirement.
Step 2: Select the pins as per circuit designed and assign a unique variable name to them.
Step 3: Data collection rate: 5 Minutes

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

Pin	Variable Name
A0 Analog	temp

The chart type is chosen as **predictionGraph**.

Write your code in the code window below.

```
predict_data
```

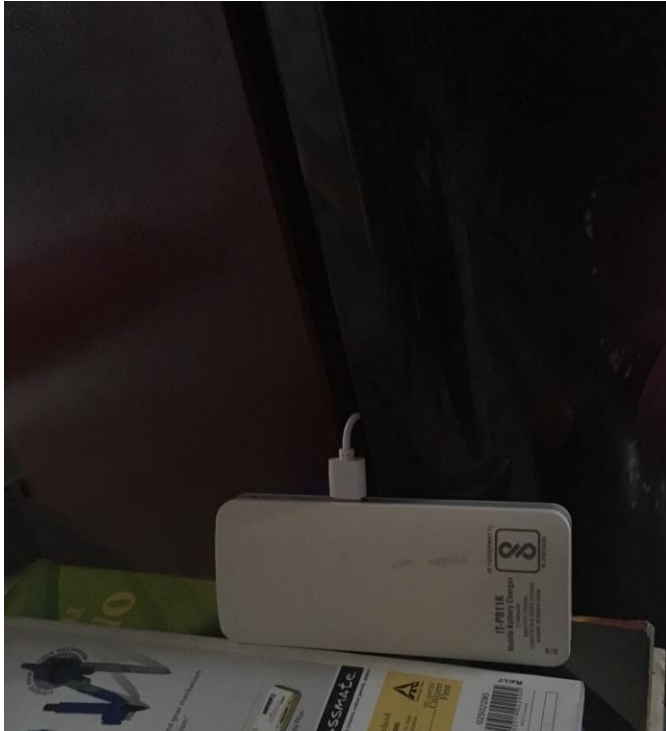
```
1 setChartLibrary('google-chart');  
2 setChartTitle('Polynomial Regression');  
3 setChartType('predictionGraph');  
4 setAxisName('time_stamp', 'temp');  
5 mul(0.0977);  
6 plotChart('time_stamp', 'temp');  
7
```

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

Pin	Variable Name
A0 Analog	temp

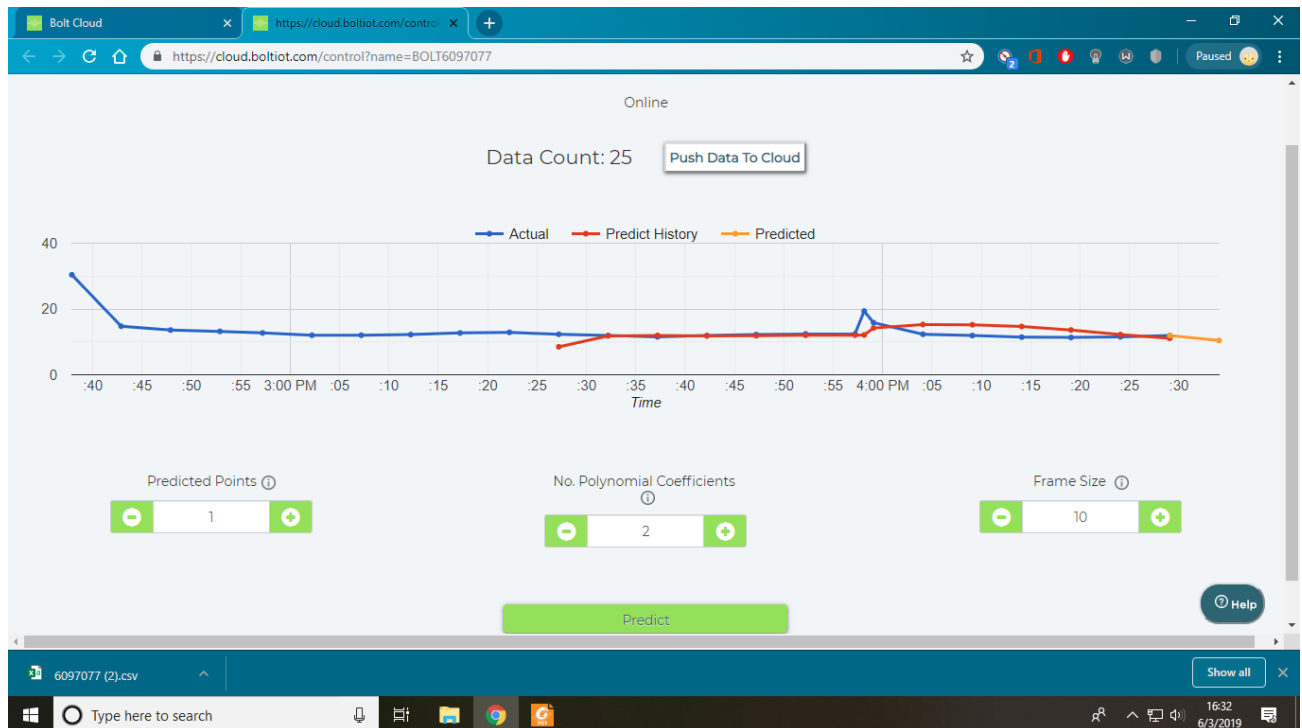
d) Keep the temperature monitoring circuit inside your fridge with the door of the fridge closed, and let the system record the temperature readings for about 2 hours.

The Bolt module is connected to 5V-1Amp of my powerbank. The circuit is inside the refrigerator.



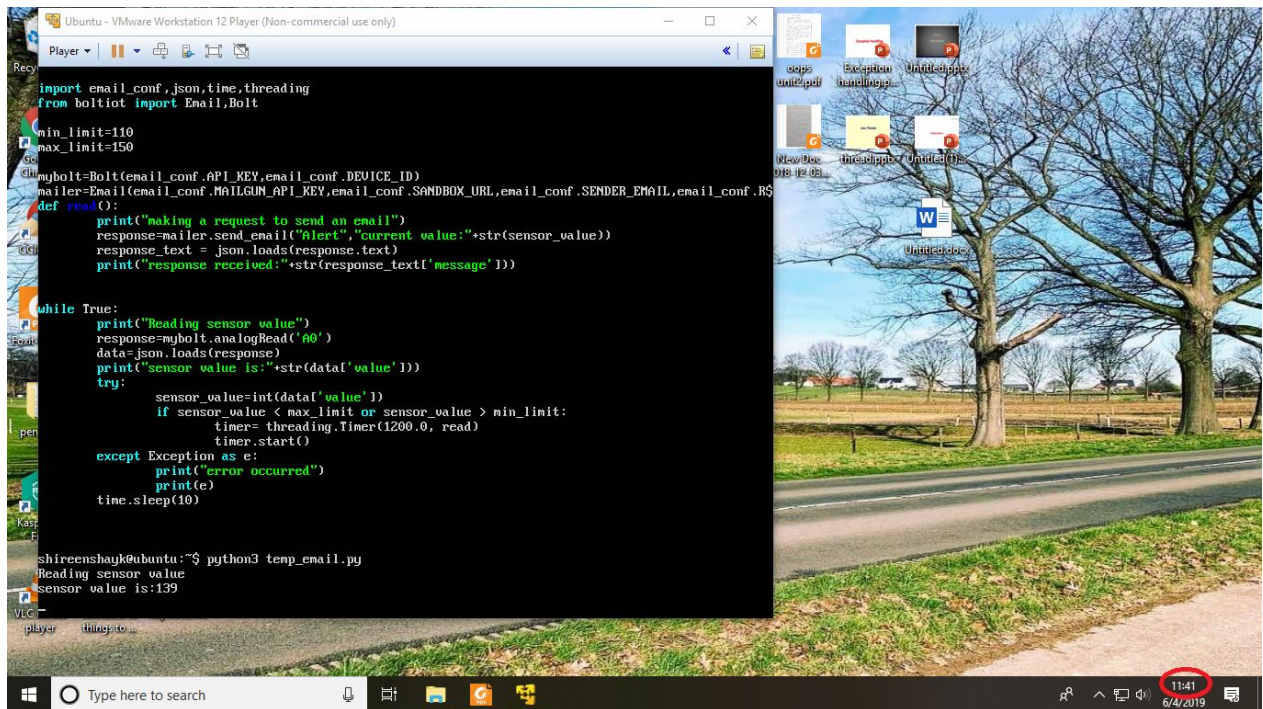
e) Using the reading that you received in the 2 hours, set boundaries for the temperature within the fridge.

This screenshot was taken after 2 hrs of keeping the Bolt module inside the refrigerator. The peak on the right side is when the door was opened for a while.

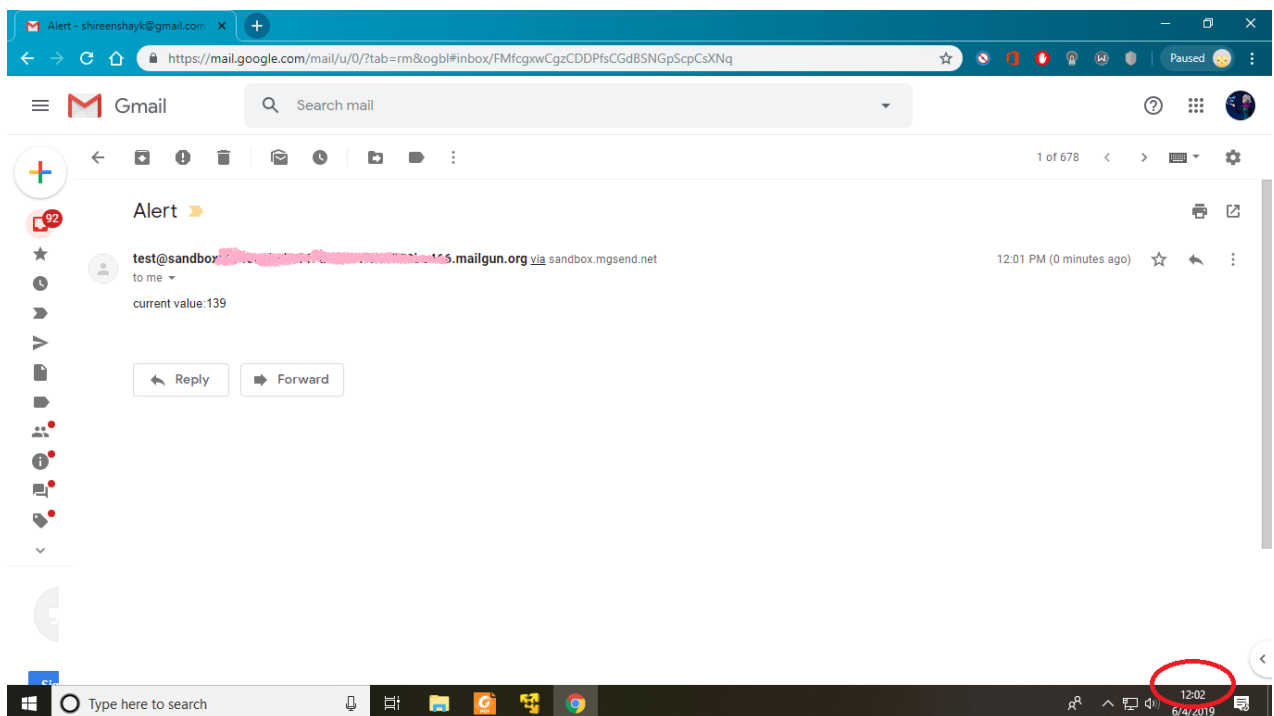


f) Write a python code which will fetch the temperature data, every 10 seconds, and send out an email alert, if the temperature goes beyond the temperature thresholds you decided on in objective 'e'.

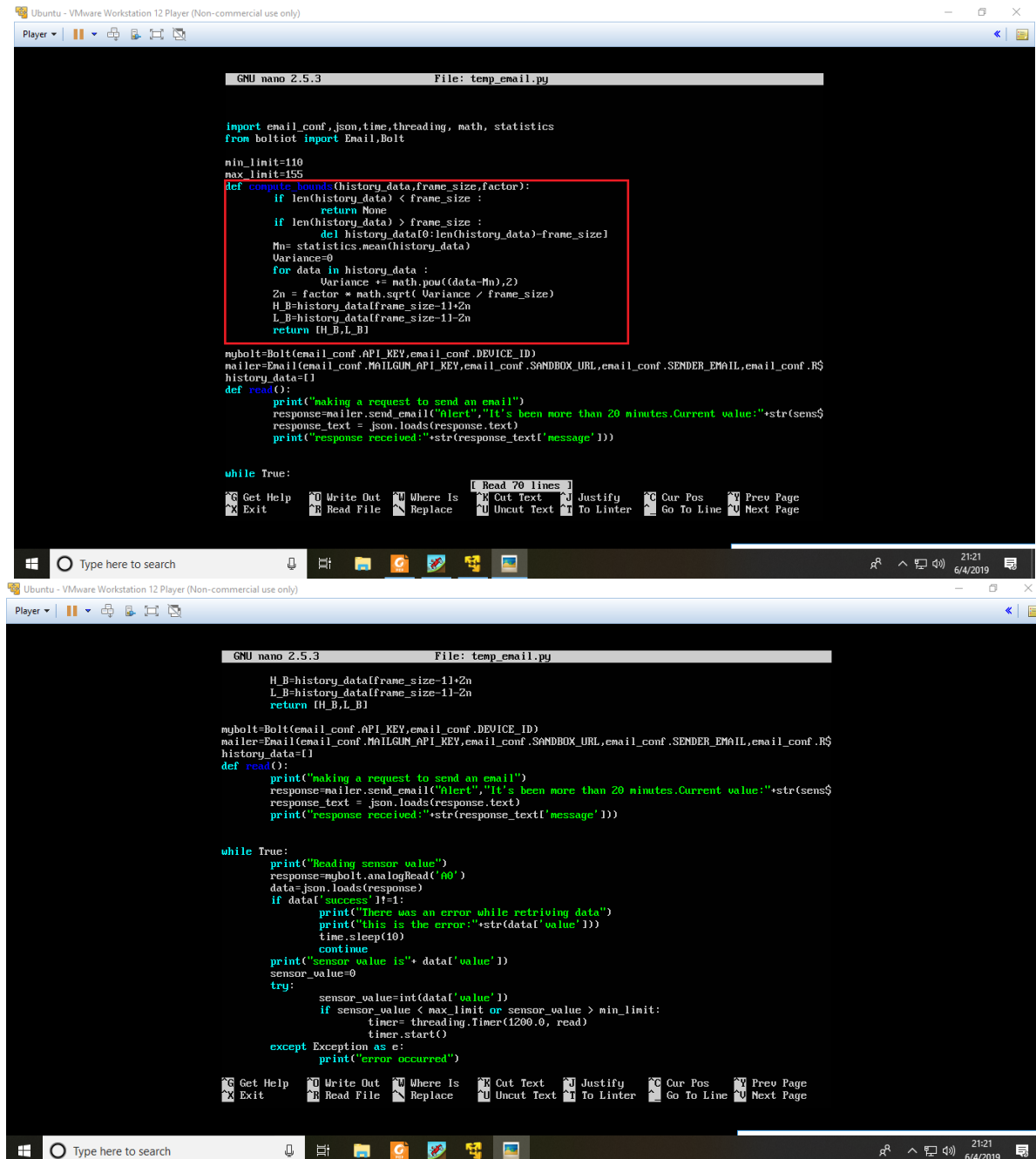
The maximum and minimum limits are 110 and 150 respectively. The function `read()` is called when the temperature stays within the specified limits for 1200 sec.(20 min.). It is responsible for sending an email.



The emails I received once the set condition was met.



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.



```
GNU nano 2.5.3 File: temp_email.py

import email_conf,json,time,threading, math, statistics
from boltiot import Email,Bolt

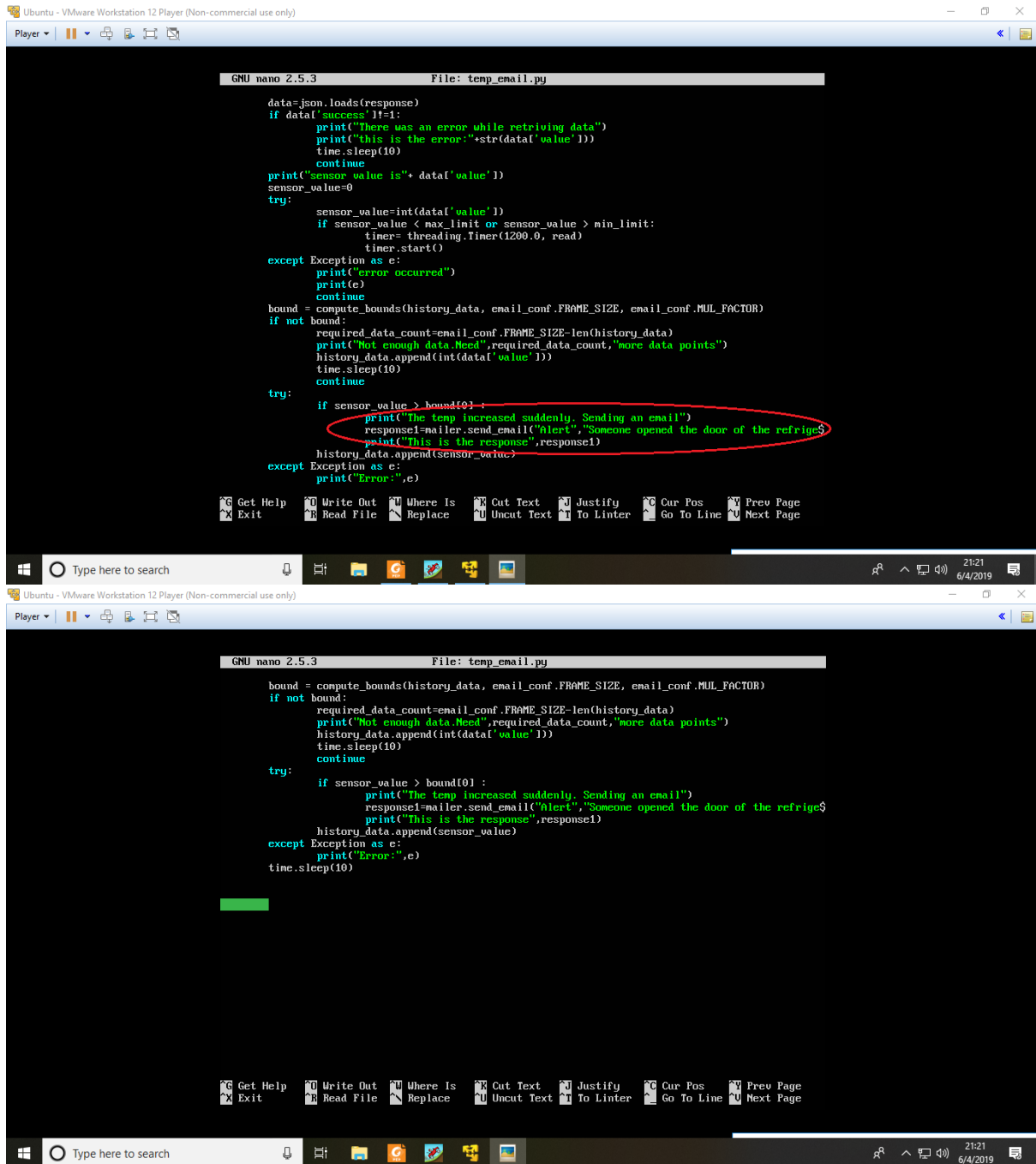
min_limit=110
max_limit=155
def compute_bounds(history_data,frame_size,factor):
    if len(history_data) < frame_size :
        return None
    if len(history_data) > frame_size :
        del history_data[0:len(history_data)-frame_size]
    Mn= statistics.mean(history_data)
    Variance=0
    for data in history_data :
        Variance += math.pow((data-Mn),2)
    Zn = factor * math.sqrt( Variance / frame_size)
    H_B=history_data[frame_size-1]*Zn
    L_B=history_data[frame_size-1]-Zn
    return [H_B,L_B]

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.RS)
history_data=[]
def read():
    print("making a request to send an email")
    response=mailer.send_email("Alert","It's been more than 20 minutes.Current value:"+str(sens$
    response_text = json.loads(response.text)
    print("response received:"+str(response_text['message']))

while True:
    print("Reading sensor value")
    response=mybolt.analogRead('A0')
    data=json.loads(response)
    if data['success']==1:
        print("There was an error while retriving data")
        print("this is the error:"+str(data['value']))
        time.sleep(10)
        continue
    print("sensor value is"+ data['value'])
    sensor_value=0
    try:
        sensor_value=int(data['value'])
        if sensor_value < max_limit or sensor_value > min_limit:
            timer= threading.Timer(1200.0, read)
            timer.start()
    except Exception as e:
        print("error occurred")

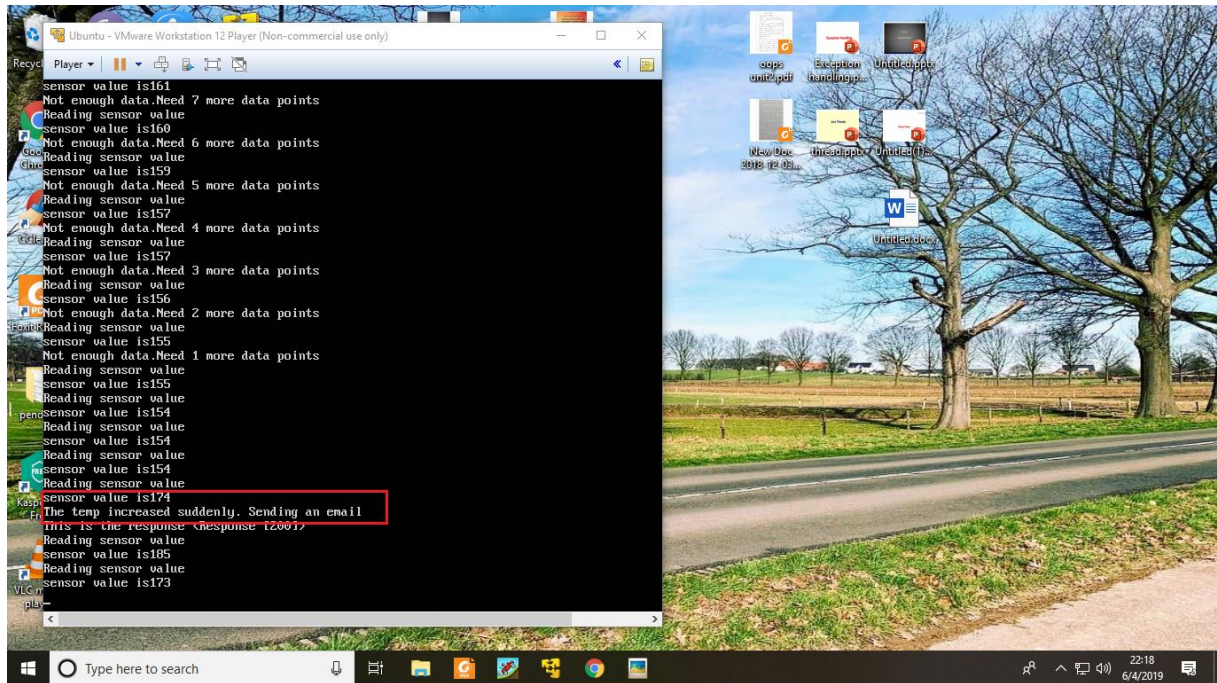
mybolt=Bolt(email_conf.API_KEY,email_conf.DEVICE_ID)
mailer=Email(email_conf.MAILGUN_API_KEY,email_conf.SANDBOX_URL,email_conf.SENDER_EMAIL,email_conf.RS)
history_data=[]
def read():
    print("making a request to send an email")
    response=mailer.send_email("Alert","It's been more than 20 minutes.Current value:"+str(sens$
    response_text = json.loads(response.text)
    print("response received:"+str(response_text['message']))

while True:
    print("Reading sensor value")
    response=mybolt.analogRead('A0')
    data=json.loads(response)
    if data['success']==1:
        print("There was an error while retriving data")
        print("this is the error:"+str(data['value']))
        time.sleep(10)
        continue
    print("sensor value is"+ data['value'])
    sensor_value=0
    try:
        sensor_value=int(data['value'])
        if sensor_value < max_limit or sensor_value > min_limit:
            timer= threading.Timer(1200.0, read)
            timer.start()
    except Exception as e:
        print("error occurred")
```

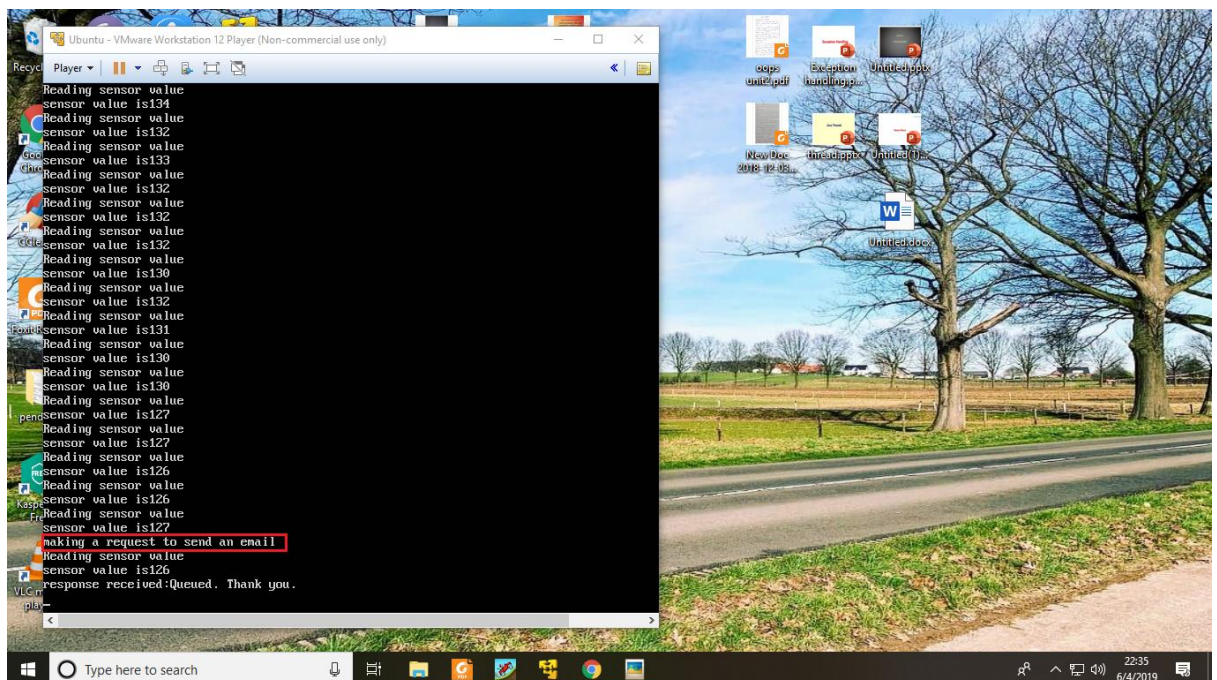


h) Tune the Z-score analysis code, such that, it detects an anomaly when someone opens the door of the fridge.

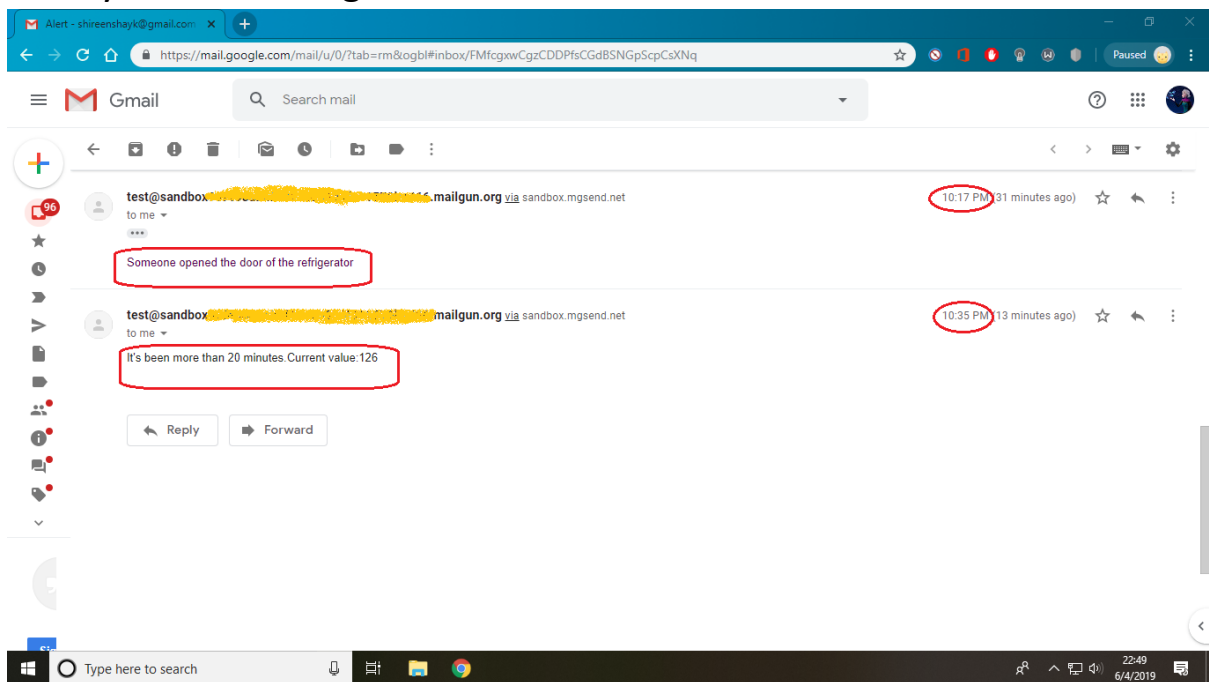
This image has 'opening the refrigerator door' case.



This image has 'temperature in the specified range for more than 20 minutes' case.



Finally, the last image has emails for both the above said cases.



By Sheikh Zubeena Shireen.