FILE MANAGEMENT

sudo nano device\_status.py   
   
api\_key = "92c94f06-a681-4ef4-9bff-f4c024fba905"  
device\_id = "BOLT6094955"  
mybolt = Bolt(api\_key, device\_id)  
response = mybolt.isOnline()  
print (response)

sudo python3 device\_status.py

sudo nano device\_control.py

api\_key = "92c94f06-a681-4ef4-9bff-f4c024fba905"  
device\_id = "BOLT6094955"  
mybolt = Bolt(api\_key, device\_id)  
response = mybolt.restart()  
print (response)

Run the file

sudo python3 device\_control.py

Type the below command to create a Python file.  
sudo nano led\_control.py

api\_key = "92c94f06-a681-4ef4-9bff-f4c024fba905"  
device\_id = "BOLT6094955"  
mybolt = Bolt(api\_key, device\_id)  
response = mybolt.digitalWrite('0', 'HIGH')  
print (response)

Save the above file using and run it in the terminal.  
sudo python3 led\_control.py

api\_key = "92c94f06-a681-4ef4-9bff-f4c024fba905"  
device\_id = "BOLT6094955"  
mybolt = Bolt(api\_key, device\_id)  
response = mybolt.digitalWrite('0', 'LOW')  
  
sudo python3 led\_control.py   
and the above command will switch off the LED.

sudo nano led\_brightness.py

api\_key = "92c94f06-a681-4ef4-9bff-f4c024fba905"  
device\_id = "BOLT6094955"  
mybolt = Bolt(api\_key, device\_id)  
response = mybolt.analogWrite('0', '10')  
print (response)

sudo python3 led\_brightness.py

SMS Enabling

<https://www.twilio.com/docs/libraries/python>

sudo nano conf.py

SSID = AC8932d97b090e6d0aa675153a067fb6cd  
AUTH\_TOKEN =2fbdc9a437fd46fa65d7f8dd2b68510e  
FROM\_NUMBER = '+17014190892'  
TO\_NUMBER = '+919833754878'  
API\_KEY = '92c94f06-a681-4ef4-9bff-f4c024fba905'  
DEVICE\_ID = 'BOLT6094955'

SMS SETUP

<https://www.twilio.com/console>

sudo nano temp\_sms.py

import conf  
from boltiot import Sms, Bolt  
import json, time  
  
minimum\_limit = 300   
maximum\_limit = 600   
  
  
mybolt = Bolt(conf.API\_KEY, conf.DEVICE\_ID)  
sms = Sms(conf.SSID, conf.AUTH\_TOKEN, conf.TO\_NUMBER, conf.FROM\_NUMBER)  
  
  
while True:   
 response = mybolt.analogRead('A0')   
 data = json.loads(response)   
 print (data['value'])  
 try:   
 sensor\_value = int(data['value'])

Temperature=(100\*sensor\_value)/1024  
 print (sensor\_value)  
 if sensor\_value > maximum\_limit or sensor\_value < minimum\_limit:  
 response = sms.send\_sms("The Current temperature sensor value is " +str(Temperature))   
 except Exception as e:   
 print ("Error",e)  
 time.sleep(10)

EMAIL SETUP

<https://app.mailgun.com/app/domains>

sudo nano email\_conf.py

MAILGUN\_API\_KEY = 'd64a88fc7118f1b49e9588651968e5a6-49a2671e-d3f02f83'   
SANDBOX\_URL= 'sandboxb60e0c4c0dce4fe685b2f2f38a4ca7c7.mailgun.org'   
SENDER\_EMAIL = 'test@sandboxb60e0c4c0dce4fe685b2f2f38a4ca7c7.mailgun.org'  
RECIPIENT\_EMAIL = 'aashayphirke5798@gmail.com'  
API\_KEY = '92c94f06-a681-4ef4-9bff-f4c024fba905'  
DEVICE\_ID = 'BOLT6094955'

import email\_conf  
from boltiot import Email, Bolt  
import json, time  
  
minimum\_limit = 300 #the minimum threshold of light value   
maximum\_limit = 600 #the maximum threshold of light 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:   
 response = mybolt.analogRead('A0')   
 data = json.loads(response)   
 print (data['value'])  
 try:   
 sensor\_value = int(data['value'])   
 Temperature=(100\*sensor\_value)/1024

print (Temperature)  
 if sensor\_value > maximum\_limit or sensor\_value < minimum\_limit:  
 response = mailer.send\_email("Alert", "The Current temperature sensor value is " +str(Temperature))  
 except Exception as e:   
 print ("Error",e)  
 time.sleep(10)

TWITTER SETUP

<https://developer.twitter.com/en/docs/accounts-and-users/create-manage-lists/api-reference/get-lists-members-show>

<https://developer.twitter.com/en/docs/tweets/post-and-engage/api-reference/post-statuses-update>

<https://developer.twitter.com/en/account/get-started>

consumer\_key = 'j489DTMfw8GoegV96qsrvtFXC'  
consumer\_secret = 'vxbvtHhoZzlD4EJeHNVpL1THDuPyZsMkgMPM3rGg9PkBfoEKcM'  
access\_token = '1079266091426230272-mLdkNKLqsa4rsFjiBU5F12BYrfHLHh'  
access\_secret ='izBLCprKORQSPKWP0pX0d6ovWlWJzZMBaEeMVvYg1nAHV'  
API\_KEY = '92c94f06-a681-4ef4-9bff-f4c024fba905'  
DEVICE\_ID = 'BOLT6094955'

import tweepy  
import conf, json, time  
from boltiot import Bolt  
  
# Dictionary to store credentials as key-value pairs.  
config = {  
"consumer\_key" : conf.consumer\_key,  
"consumer\_secret" : conf.consumer\_secret,  
"access\_token" : conf.access\_token,  
"access\_token\_secret" : conf.access\_token\_secret  
}  
  
# Method to authenticate user via Tweepy and return API object  
def get\_api\_object(cfg):  
 auth =tweepy.OAuthHandler(cfg['consumer\_key'],  
 cfg['consumer\_secret'])  
 auth.set\_access\_token(cfg['access\_token'],  
 cfg['access\_token\_secret'])  
 return tweepy.API(auth)  
  
mybolt = Bolt(conf.bolt\_cloud\_api\_key, conf.device\_id)  
temperature\_threshold = 59  
while True:  
 response = mybolt.analogRead('A0')  
 data = json.loads(response)  
 print (data['value'])  
 try:  
 sensor\_value = int(data['value'])  
 if sensor\_value > temperature\_threshold:  
 print "Temperature has crossed the threshold."  
 # Call get\_api\_object to authenticate user and get the API object  
 api\_object = get\_api\_object(config)  
 # Store the tweet message in the variable  
 tweet = "Temperature has crossed the threshold."  
 # Post the tweet on your Twitter account using the update\_status method.  
 status = api\_object.update\_status(status=tweet)  
 except Exception as e:  
 print ("An error occurred ", e)  
 time.sleep(10)