

ASSIGNMENT-2

ASSIGNMENT : Build Python code, Generate Temperature and Humidity values (Use Random function to generate values) and write a condition to detect an alarm in case of high temperature and high Humidity

NAME:RAJAPPA R

PROGARM

```
#1 import
```

```
try:
```

```
    import configparser except:
```

```
    from six.moves import configparser
```

```
import smtplib from email.mime.multipart import
```

```
MIMEMultipart from email.mime.text import
```

```
MIMEText import requests
```

```
#2 variable related to weather API
```

```
weather_dict = {'freezing_rain_heavy': 'Heavy rain and snow', 'freezing_rain': 'Rain and snow',  
'freezing_rain_light': 'Light rain and snow', 'freezing_drizzle': 'Light drizzle and snow',  
'ice_pellets_heavy': 'Heavy ice pellets', 'ice_pellets': 'Normal ice pellets', 'ice_pellets_light': 'Light ice  
pellets', 'snow_heavy': 'Heavy snow', 'snow': 'Normal snow', 'snow_light': 'Light snow', 'tstorm':  
'Thunder storm', 'rain_heavy': 'Heavy rain', 'rain': 'Normal rain', 'rain_light': 'Light rain'}  
url = "https://api.climacell.co/v3/weather/nowcast"
```

```
querystring =  
{ "lat": "1.29027", "lon": "103.851959", "unit_system": "si", "timestep": "60", "start_time": "now", "fields": "temp, humidity, weather_code", "apikey": "xxxx" }
```

```
#3 class class
```

```
EmailSender(): #4
```

```
initialization def
```

```
__init__(self):
```

```
    self.cf = configparser.ConfigParser()
```

```
self.cf.read('./config.ini')    self.sec =
```

```
'email'
```

```
    self.email = self.cf.get(self.sec, 'email')
```

```
self.host = self.cf.get(self.sec, 'host')    self.port
```

```
= self.cf.get(self.sec, 'port')    self.password =
```

```
self.cf.get(self.sec, 'password')
```

```
#5 main function to send email
```

```
def SendEmail(self, recipient):
```

```
title = "Home Sweet Home"
```

```
#6 create a new multipart mime object
```

```
msg = MIMEMultipart()    msg['Subject'] =
```

```
'[Weather Notification]'    msg['From'] =
```

```
self.email    msg['To'] = ', '.join(recipient)
```

```

#7 call weather API using requests      response =
requests.request("GET", url, params=querystring)

result = ""

json_data = response.json()

#print(json_data)

#8 loop over each data and check for abnormal weather (rain, snow)
for i in range(len(json_data)):
    if(json_data[i]['weather_code']['value'] in weather_dict):
        if(i == 0):
            result = "%s at the moment. Current temperature is " %
(weather_dict[json_data[i]['weather_code']['value']])
        else:
            result = "%s in %s hour(s) time. Forecasted temperature is " %
(weather_dict[json_data[i]['weather_code']['value']], i)

        result += '%s%s while the humidity is about %s%s' % (json_data[i]['temp']['value'],
json_data[i]['temp']['units'], json_data[i]['humidity']['value'], json_data[i]['humidity']['units'])

    msgText = MIMEText('<b>%s</b><p>%s</p>' % (title, result), 'html')
msg.attach(msgText)

#9 authenticate and send email
with smtplib.SMTP(self.host, self.port) as smtpObj:

```

```
        smtpObj.ehlo()          smtpObj.starttls()
smtpObj.login(self.email, self.password)
smtpObj.sendmail(self.email, recipient, msg.as_string())
return "Success"

        return "Failed"

break
```