

PROJECT DEVELOPMENT PHASE

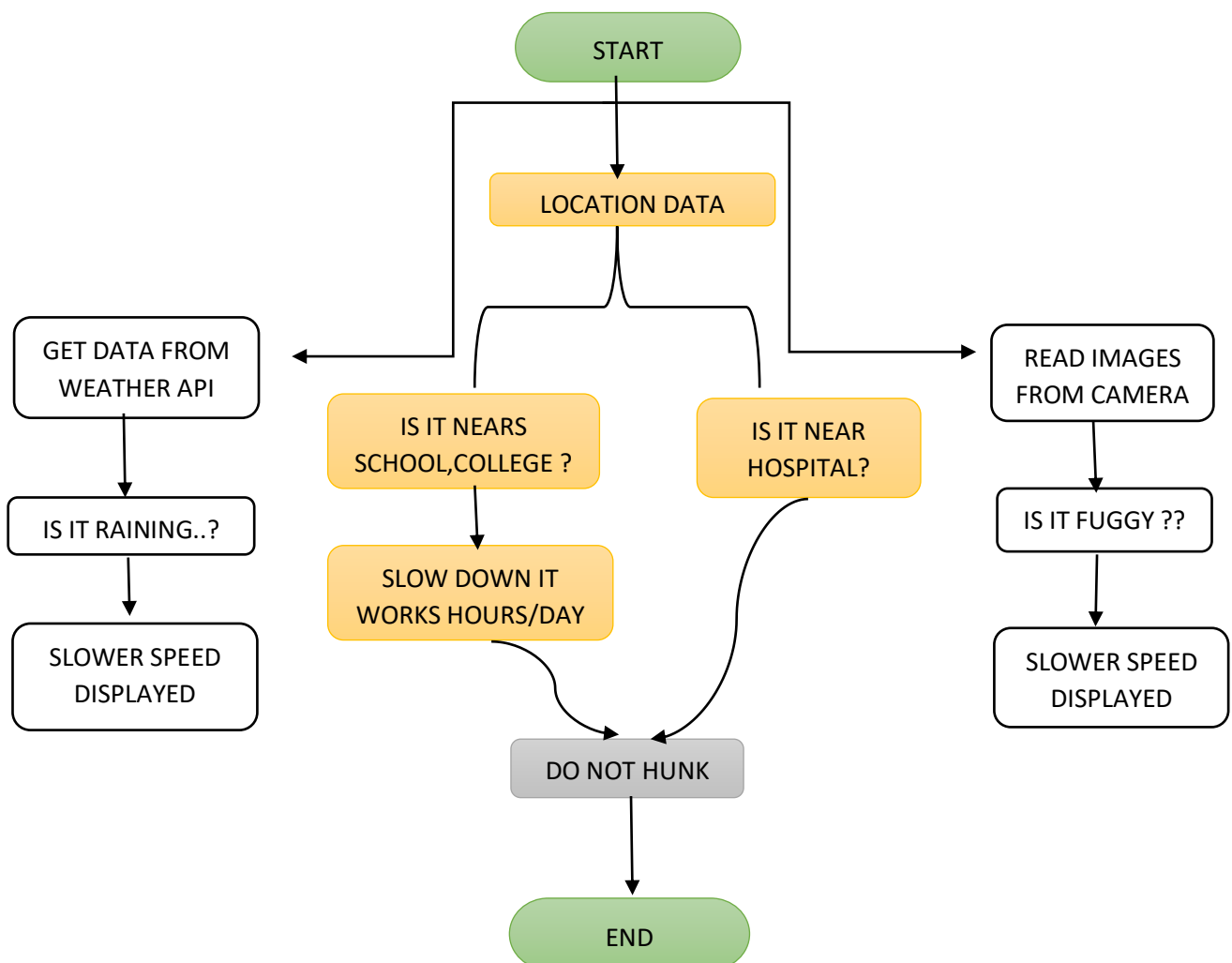
SPRINT 1

TEAM ID	PNT2022TMID44448
PROJECT NAME	Signs With smart connectivity for better road safety

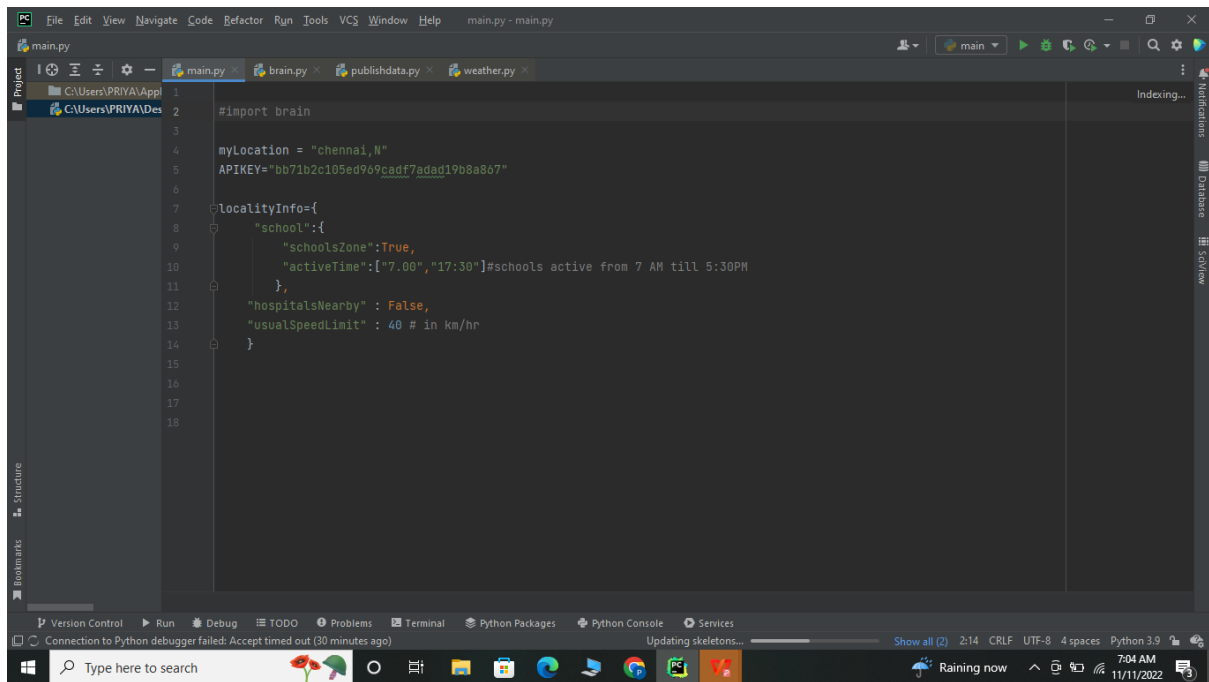
SPRINT GOALS:

1. Create and initialize accounts in various public APIs like Open Weather API.
2. Write a Python program that outputs results given the inputs like weather and location.

COAD FLOW

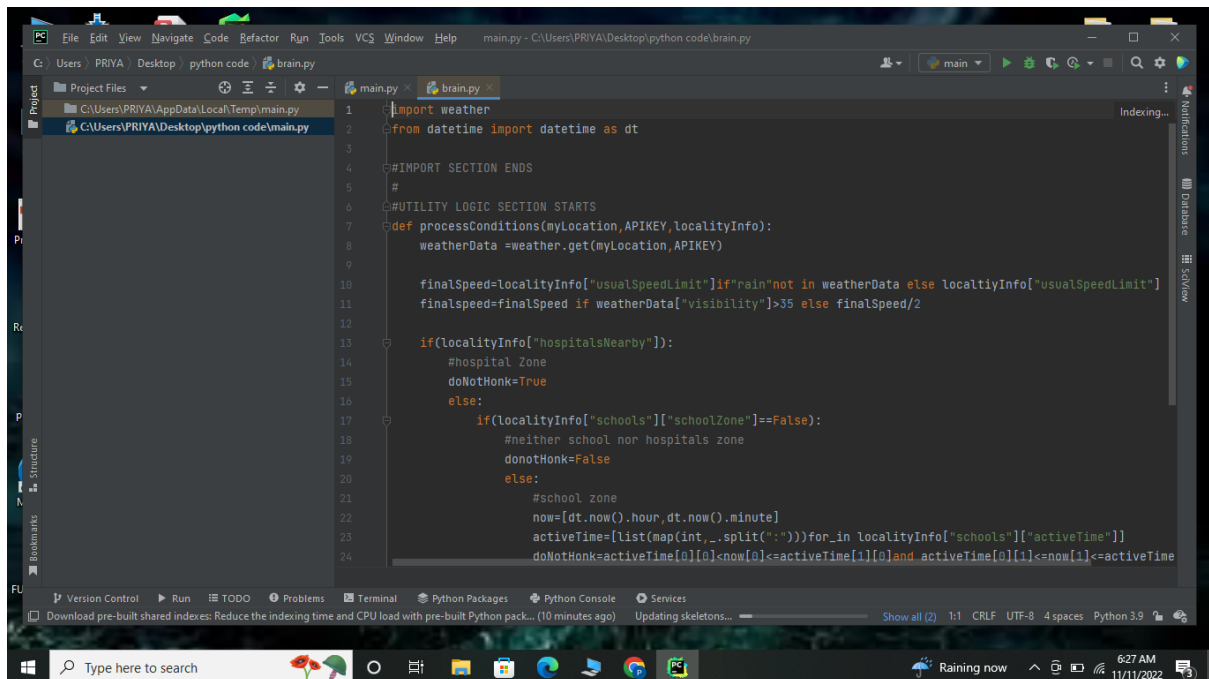


MAIN.PY



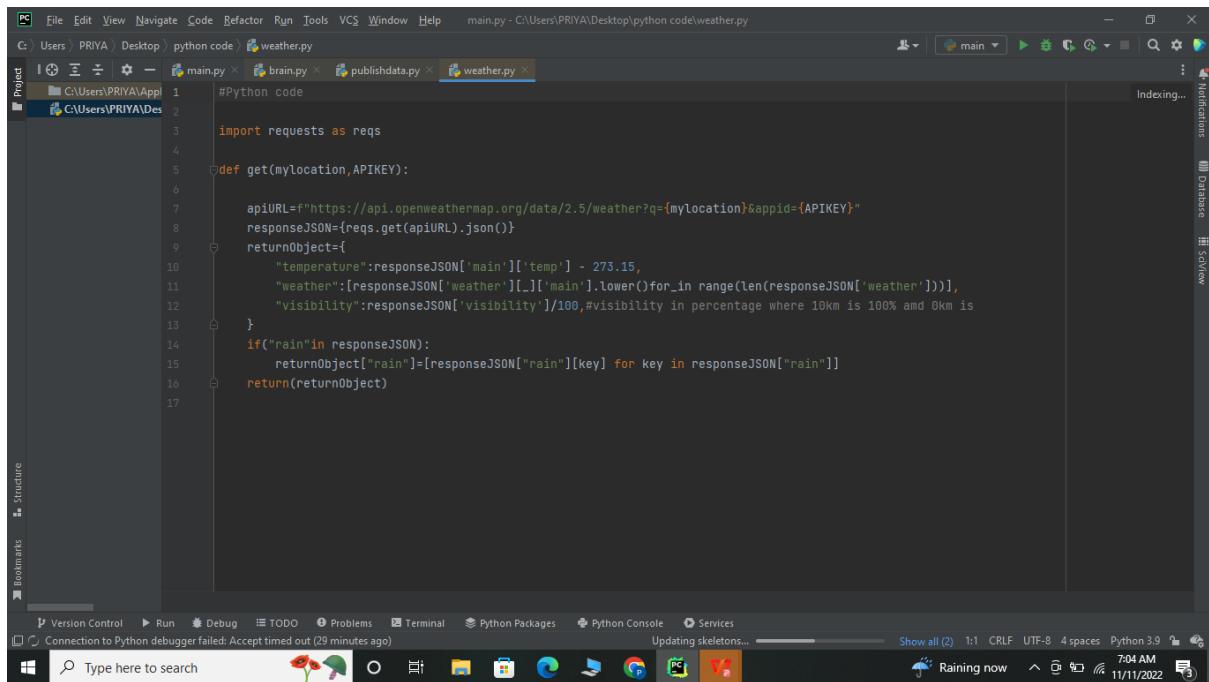
```
1 #import brain
2
3
4 myLocation = "chennai,N"
5 APIKEY="bb71b2c105ed969cadf7adad19b8a867"
6
7 localityInfo={
8     "school":{
9         "schoolsZone":True,
10        "activeTime":["7.00","17:30"]#schools active from 7 AM till 5:30PM
11    },
12    "hospitalsNearby" : False,
13    "usualSpeedLimit" : 40 # in km/hr
14 }
15
16
17
18
```

BRAIN.PY



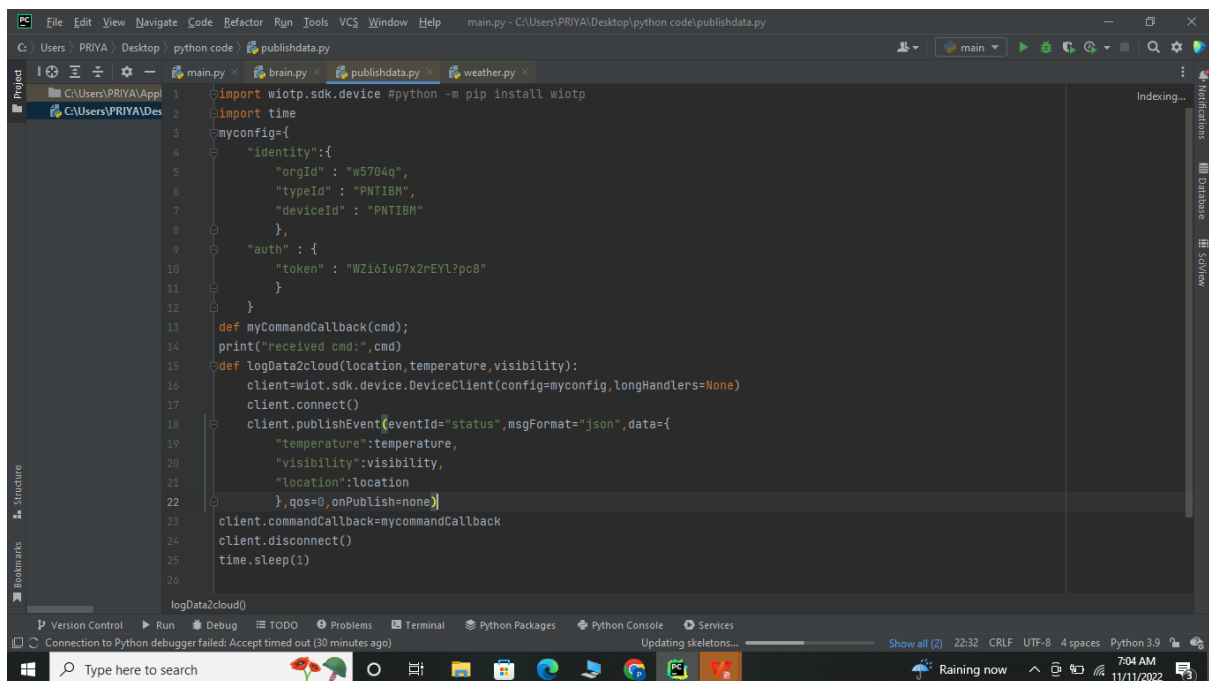
```
1 import weather
2 from datetime import datetime as dt
3
4 #IMPORT SECTION ENDS
5 #
6 #UTILITY LOGIC SECTION STARTS
7 def processConditions(myLocation,APIKEY,localityInfo):
8     weatherData =weather.get(myLocation,APIKEY)
9
10    finalSpeed=localityInfo["usualSpeedLimit"]if"rain"not in weatherData else localityInfo["usualSpeedLimit"]
11    finalSpeed=finalSpeed if weatherData["visibility"]>35 else finalSpeed/2
12
13    if(localityInfo["hospitalsNearby"]):
14        #hospital zone
15        doNotHonk=True
16    else:
17        if(localityInfo["schools"]!="schoolsZone"):
18            #neither school nor hospitals zone
19            doNotHonk=False
20        else:
21            #school zone
22            now=[dt.now().hour,dt.now().minute]
23            activeTime=[list(map(int,_.split(":")))for _ in localityInfo["schools"]['activeTime']]
24            doNotHonk=activeTime[0][0]<now[0]<=activeTime[1][0] and activeTime[0][1]<=now[1]<=activeTime[1][1]
```

WEATHER.PY



```
1 #Python code
2
3 import requests as reqs
4
5 def get(mylocation,APIKEY):
6
7     apiURL=f"https://api.openweathermap.org/data/2.5/weather?q={mylocation}&appid={APIKEY}"
8     responseJSON=(reqs.get(apiURL).json())
9     returnObject={
10         "temperature":responseJSON['main']['temp'] - 273.15,
11         "weather":[responseJSON['weather'][_]['main'].lower()for _in range(len(responseJSON['weather']))],
12         "visibility":responseJSON['visibility']/100,#visibility in percentage where 10km is 100% and 0km is
13     }
14     if("rain" in responseJSON):
15         returnObject["rain"]=[responseJSON["rain"][key] for key in responseJSON["rain"]]
16     return(returnObject)
17
```

PUBLISH DATA.PY



```
1 import wiotsdk.device #python -m pip install wiotsdk
2 import time
3 myconfig={
4     "identity":{
5         "orgId": "w5704q",
6         "typeId": "PNTIBH",
7         "deviceId": "PNTIBH"
8     },
9     "auth": {
10         "token": "WZi6iv67x2rEYL?pc8"
11     }
12 }
13 def myCommandCallback(cmd):
14     print("received cmd:",cmd)
15 def logData2cloud(location,temperature,visibility):
16     client=wiotsdk.device.DeviceClient(config=myconfig,longHandlers=None)
17     client.connect()
18     client.publishEvent(eventId="status",msgFormat="json",data={
19         "temperature":temperature,
20         "visibility":visibility,
21         "location":location
22     },qos=0,onPublish=None)
23     client.commandCallback=myCommandCallback
24     client.disconnect()
25     time.sleep(1)
26
27 logData2cloud()
```

OUTPUT:

Code Output

2022-11-06 21:38:33,452 wiotp.sdk.device.client.DeviceClient INFO

Connected successfully: d:epmoec:testDevice:device0

2022-11-06 21:38:33,452 wiotp.sdk.device.client.DeviceClient INFO

Disconnected from the IBM Watson IoT Platform

2022-11-06 21:38:33,452 wiotp.sdk.device.client.DeviceClient INFO

Closed connection to the IBM Watson IoT Platform

{'speed': 40, 'doNotHonk': False}

2022-11-06 21:38:35,631 wiotp.sdk.device.client.DeviceClient INFO

Connected successfully: d:epmoec:testDevice:device0

2022-11-06 21:38:35,631 wiotp.sdk.device.client.DeviceClient INFO

Disconnected from the IBM Watson IoT Platform

2022-11-06 21:38:35,631 wiotp.sdk.device.client.DeviceClient INFO

Closed connection to the IBM Watson IoT Platform

{'speed': 40, 'doNotHonk': False}

... repeats every 1 sec