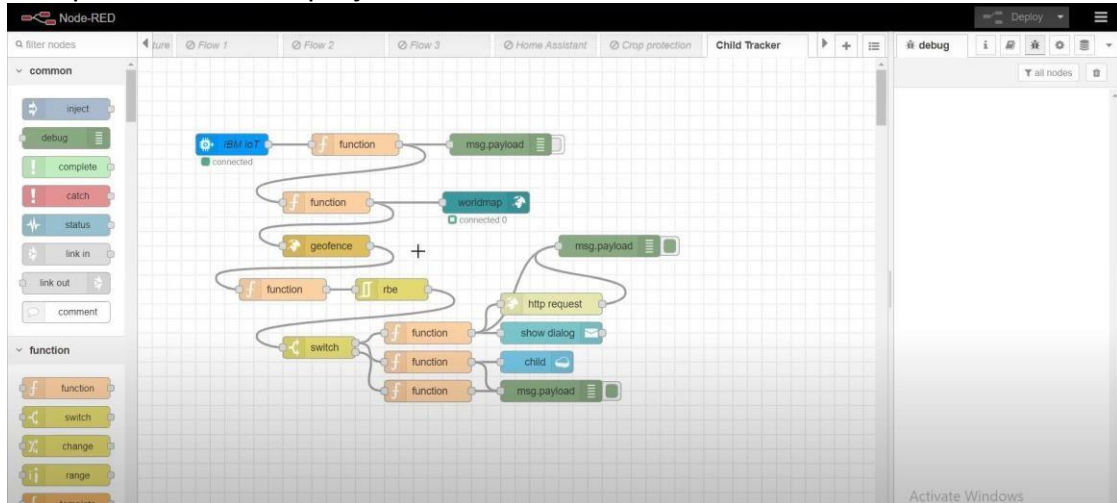


Develop The Web Application Using Node-RED

1. To Develop the web application using Node-RED

Steps :

1. Open a Node-RED project



2. Add code to get child location in python

```
import json
import wiotp.sdk.device
import time

myConfig = {
    "identity": {
        "orgId": "hj5fmy",
        "typeId": "NodeMCU",
        "deviceId": "12345"
    },
    "auth": {
        "token": "12345678"
    }
}

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

while True:
    name= "Smartbridge"
    #in area location

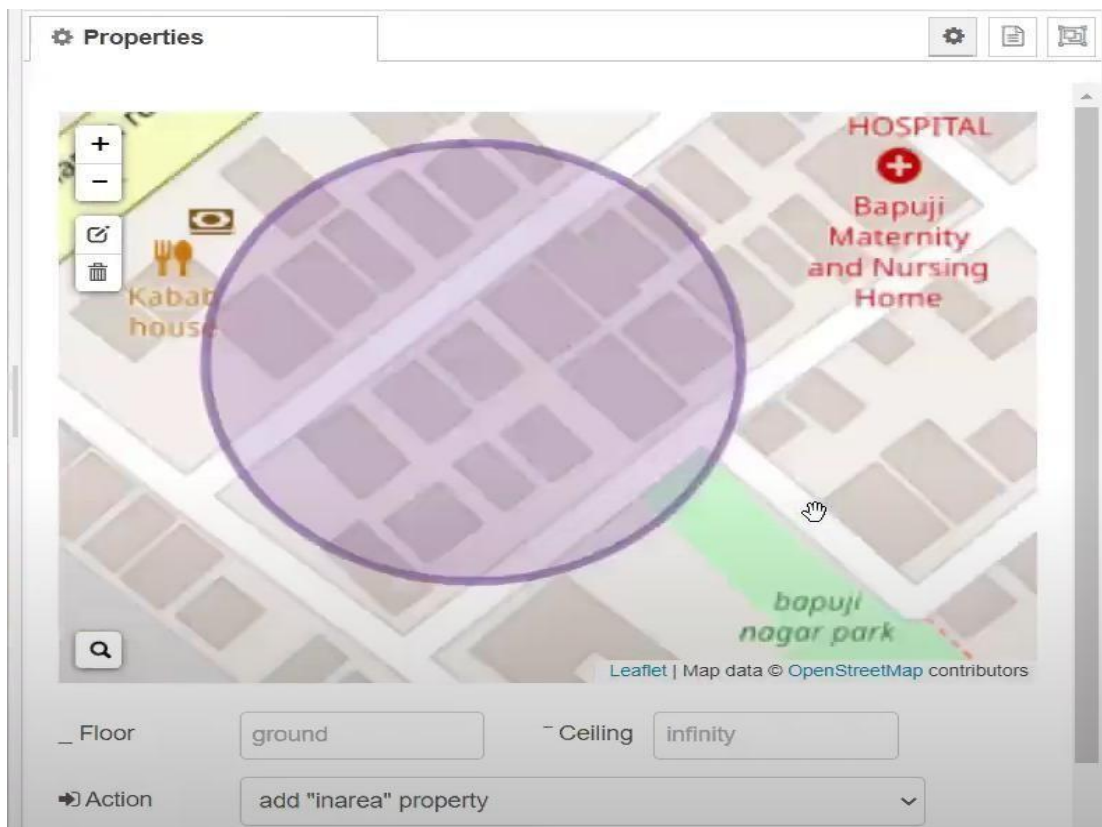
    latitude= 17.4225176
    longitude= 78.5458842

    #out area location

    #latitude= 17.4219272
    #longitude= 78.5488783
    myData={'name': name, 'lat':latitude,'lon':longitude}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
    print("Data published to IBM IoT platform: ",myData)
    time.sleep(5)

client.disconnect()
```

3. Create the GeoFence



4. Edit the HTTP Request URL

The screenshot shows the 'Edit http request node' dialog box. At the top, there are buttons for 'Delete', 'Cancel', and 'Done'. Below these is a 'Properties' section with a gear icon and three sub-panels: 'Method', 'URL', and 'Payload'. The 'Method' dropdown is set to 'GET'. The 'URL' field is highlighted with a blue selection box containing the text 'payload}}&language=english&flash=0&numbers='. The 'Payload' dropdown is set to 'Ignore'. Below these are four checkboxes: 'Enable secure (SSL/TLS) connection', 'Use authentication', 'Enable connection keep-alive', and 'Use proxy', all of which are unchecked. At the bottom, there are two more fields: 'Return' set to 'a UTF-8 string' and 'Name' set to 'Name'. To the right of the dialog is a 'debug' panel with a toolbar containing icons for 'i', 'f', 'g', 's', and a dropdown arrow. Below the toolbar is a search bar with the text 'all nodes' and a trash icon.

Edit http request node

Delete Cancel Done

Properties

Method GET

URL payload}}&language=english&flash=0&numbers=

Payload Ignore

☐ Enable secure (SSL/TLS) connection

☐ Use authentication

☐ Enable connection keep-alive

☐ Use proxy

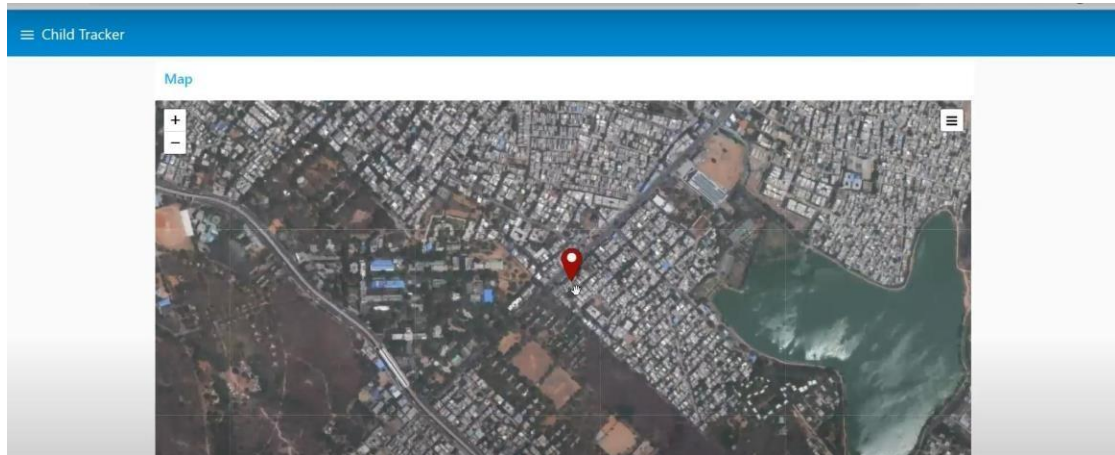
Return a UTF-8 string

Name Name

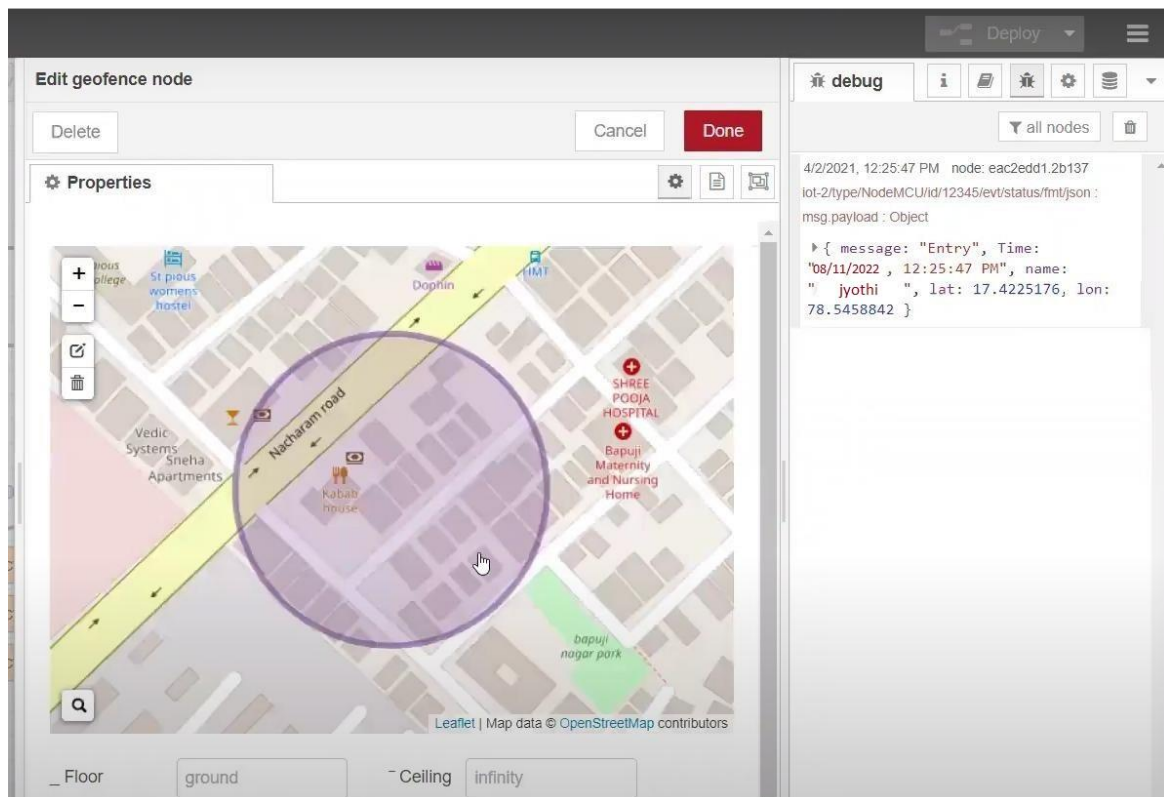
debug

all nodes

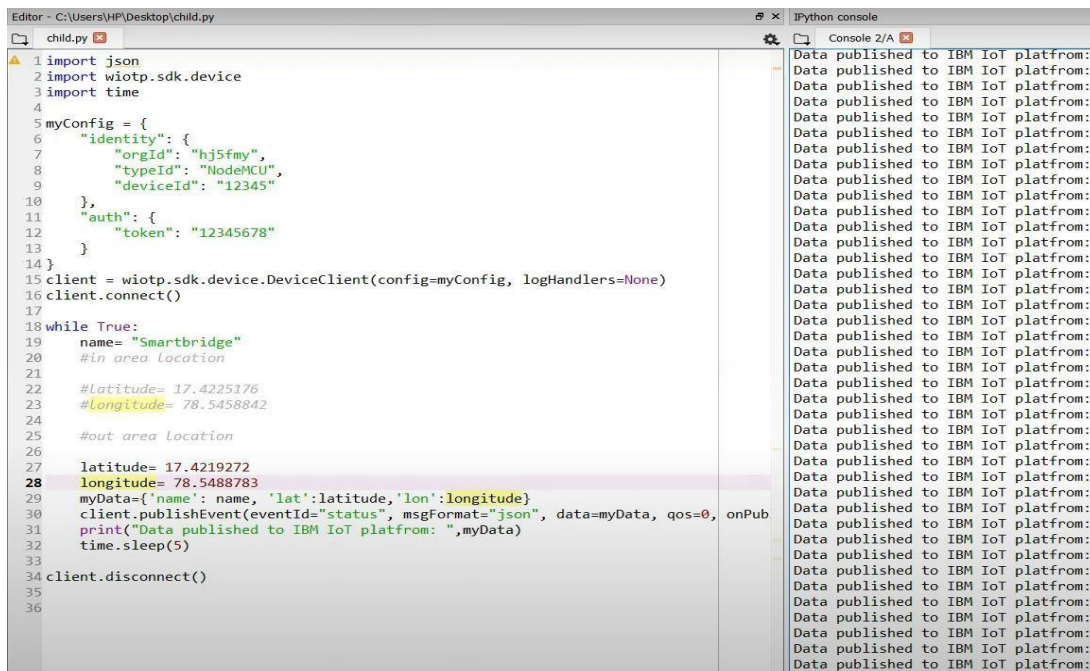
5. Locate the child



6. Create the geofence node



7. Python script send requests to IBM Cloud



The image shows a screenshot of a Python script in an IDE (Editor - C:\Users\HP\Desktop\child.py) and its corresponding console output (IPython console).

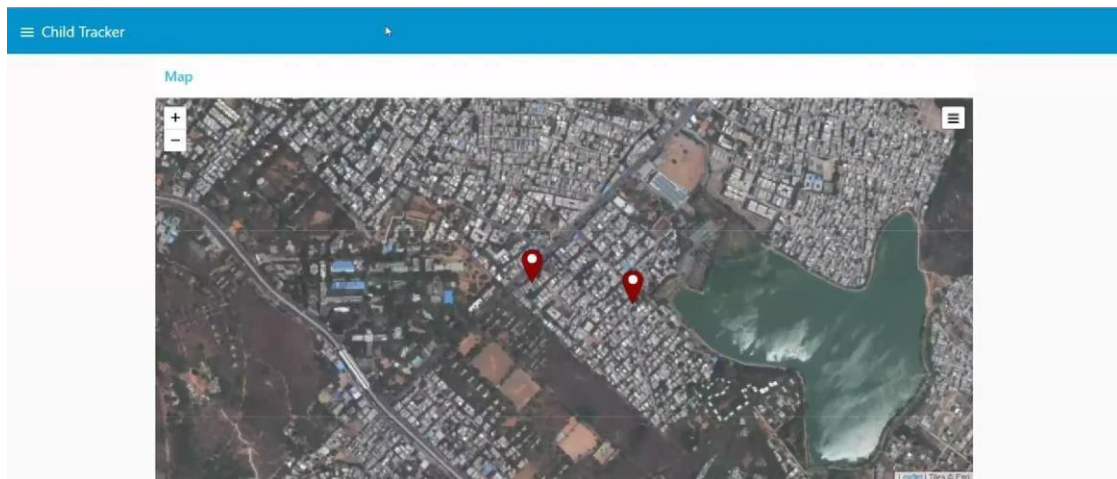
Python Script (child.py):

```
1 import json
2 import wiotp.sdk.device
3 import time
4
5 myConfig = {
6     "identity": {
7         "orgId": "hj5fmy",
8         "typeId": "NodeMCU",
9         "deviceId": "12345"
10    },
11    "auth": {
12        "token": "12345678"
13    }
14}
15 client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
16 client.connect()
17
18 while True:
19     name= "Smartbridge"
20     #in area location
21
22     #Latitude= 17.4225176
23     #Longitude= 78.5458842
24
25     #out area location
26
27     latitude= 17.4219272
28     longitude= 78.5488783
29     myData={'name': name, 'lat':latitude, 'lon':longitude}
30     client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPub
31     print("Data published to IBM IoT platfrom: ",myData)
32     time.sleep(5)
33
34 client.disconnect()
35
36
```

Console Output (IPython console):

The console shows a continuous stream of messages: "Data published to IBM IoT platfrom:". This indicates that the script is successfully publishing data to the IBM IoT platform in a loop.

7. After running the script, the web UI shows “Person is not in the particular area”



Conclusion:

Developed the web application using Node-RED Successfully