

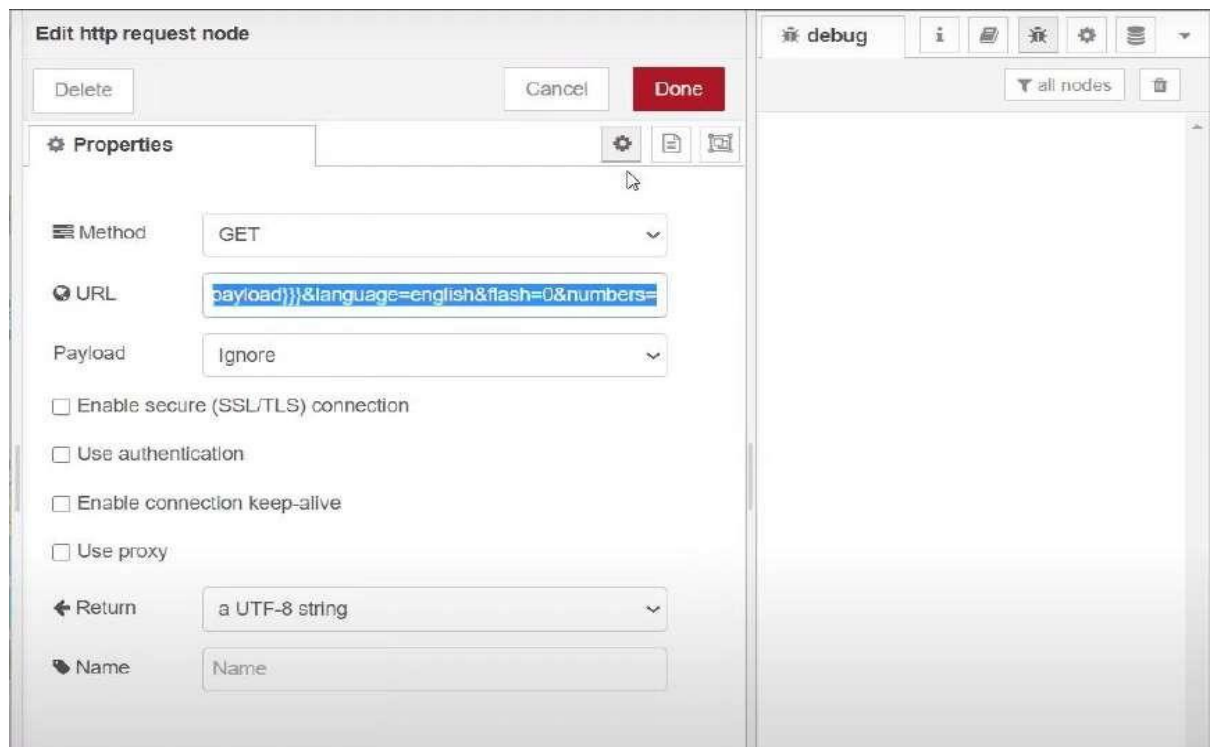
3. Create the GeoFence

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flow named 'Flow 1' with a sequence of nodes: 'IBM IoT' (with a green 'connected' status), followed by four 'function' nodes, and a 'geofence' node. The 'location' category in the left sidebar is expanded, showing various location-related nodes like 'worldmap', 'worldmap in', 'tracks', 'convex - hull', and 'geofence'.

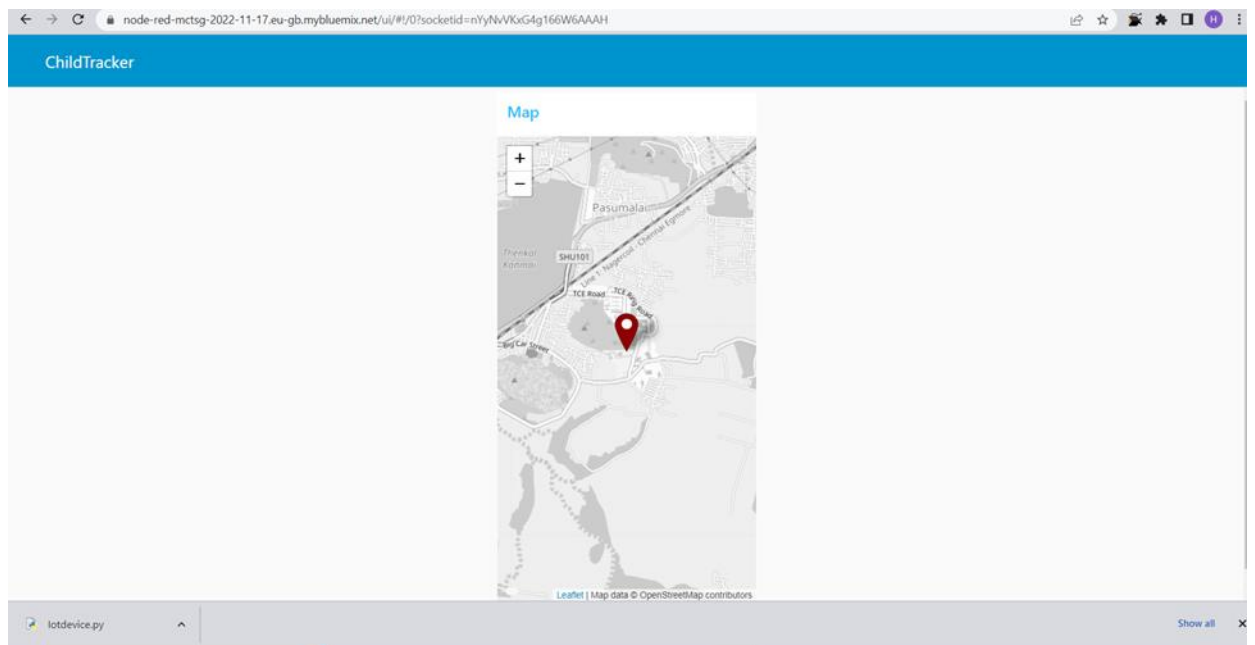
The 'Edit geofence node' configuration panel is open on the right. It features a map of Thiruparankundram with a blue polygonal geofence. The panel includes a 'Delete' button, 'Cancel', and 'Done' buttons. Below the map, there are fields for '_Floor' (set to 'ground') and '_Ceiling' (set to 'infinity'). At the bottom, there is an 'Enabled' checkbox.

The right sidebar shows the 'info' tab with a search bar and a list of flows. The 'geofence' node is selected, showing its ID as 'bad851be662dd060' and its type as 'geofence'. A note at the bottom of the sidebar instructs: 'Move the selected nodes using the ↑ ↓ and → ← keys. Hold ⌘ to nudge them further'.

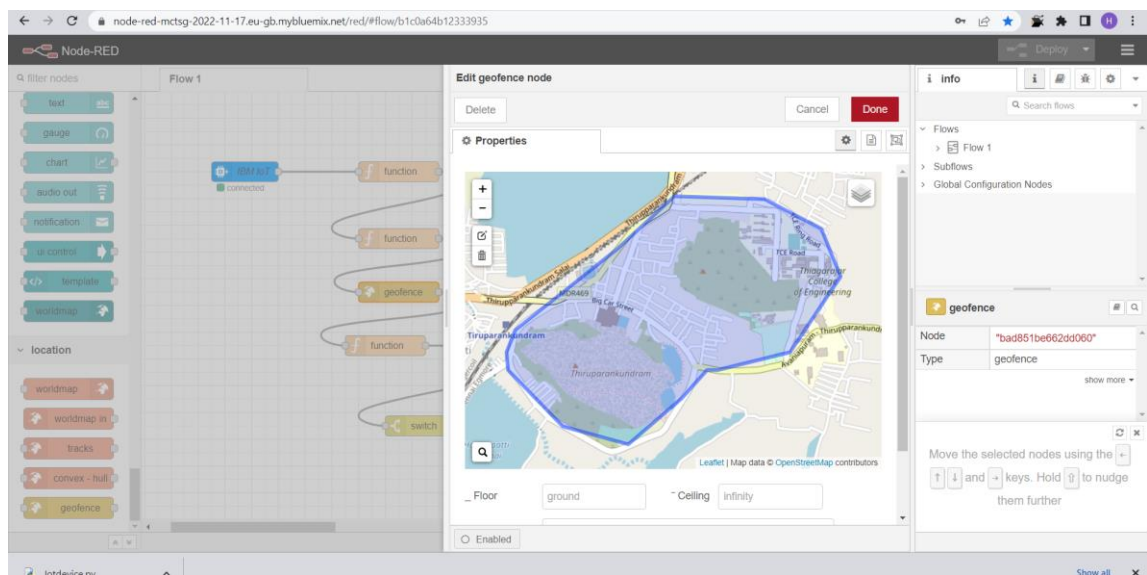
4. Edit the HTTP Request URL



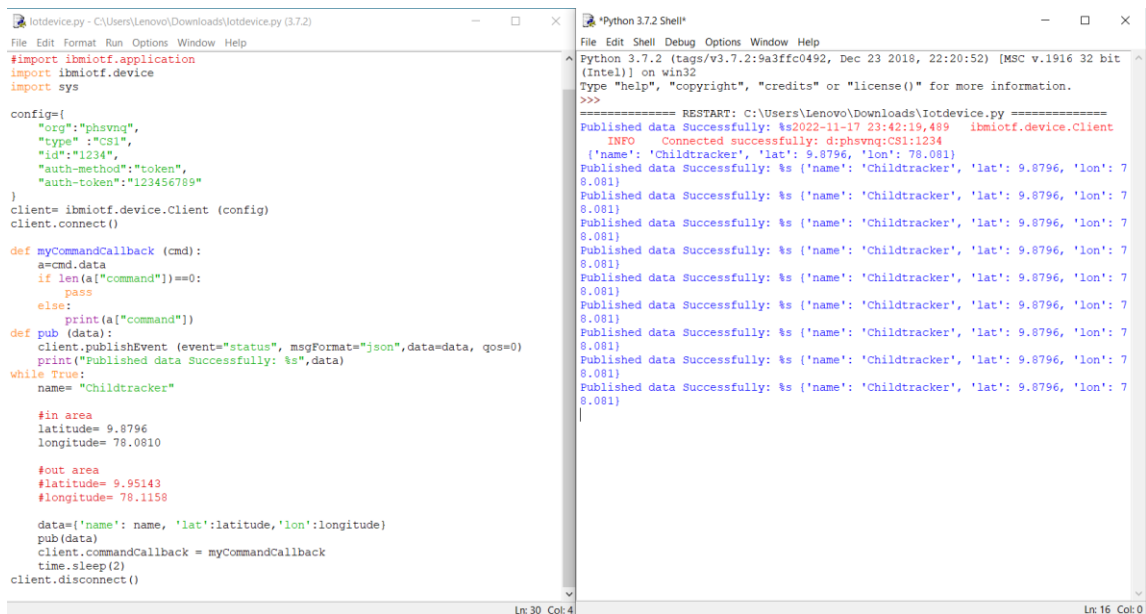
4. Locate the child



5. Create the geofence node



7. Python script send requests to IBM Cloud



The image shows two side-by-side windows. The left window is a text editor showing a Python script named `iotdevice.py`. The script imports `ibmiotf.application`, `ibmiotf.device`, and `sys`. It defines a configuration dictionary with fields like `org`, `type`, `id`, `auth-method`, and `auth-token`. It then creates an `ibmiotf.device.Client` object and calls `connect()`. A `myCommandCallback` function is defined, which checks if a command is received and publishes data. The script enters a `while True` loop, publishing data for a device named 'Childtracker' with specific latitude and longitude coordinates. The right window is a Python 3.7.2 Shell showing the execution of the script. It displays the restart command, connection status, and multiple 'Published data Successfully' messages, each showing the device name and coordinates.

```
iotdevice.py - C:\Users\Lenovo\Downloads\iotdevice.py (3.7.2)
File Edit Format Run Options Window Help
# import ibmiotf.application
# import ibmiotf.device
import sys

config={
    "org":"phsvng",
    "type":"CS1",
    "id":"1234",
    "auth-method":"token",
    "auth-token":"123456789"
}

client= ibmiotf.device.Client (config)
client.connect()

def myCommandCallback (cmd):
    a=cmd.data
    if len(a["command"])==0:
        pass
    else:
        print(a["command"])
def pub (data):
    client.publishEvent (event="status", msgFormat="json",data=data, qos=0)
    print("Published data Successfully: %s",data)
while True:
    name= "Childtracker"

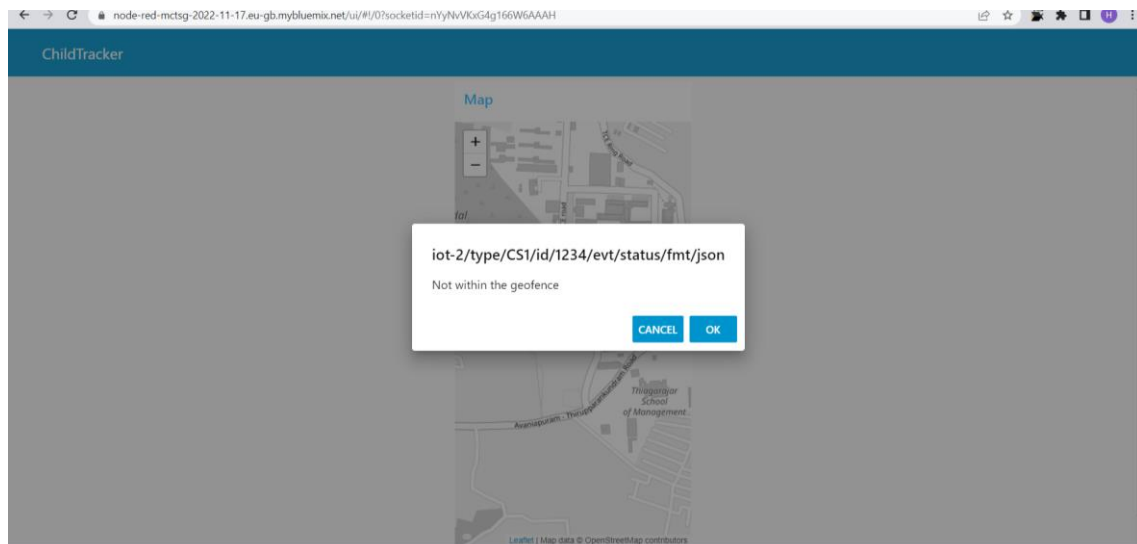
    #in area
    latitude= 9.8796
    longitude= 78.0810

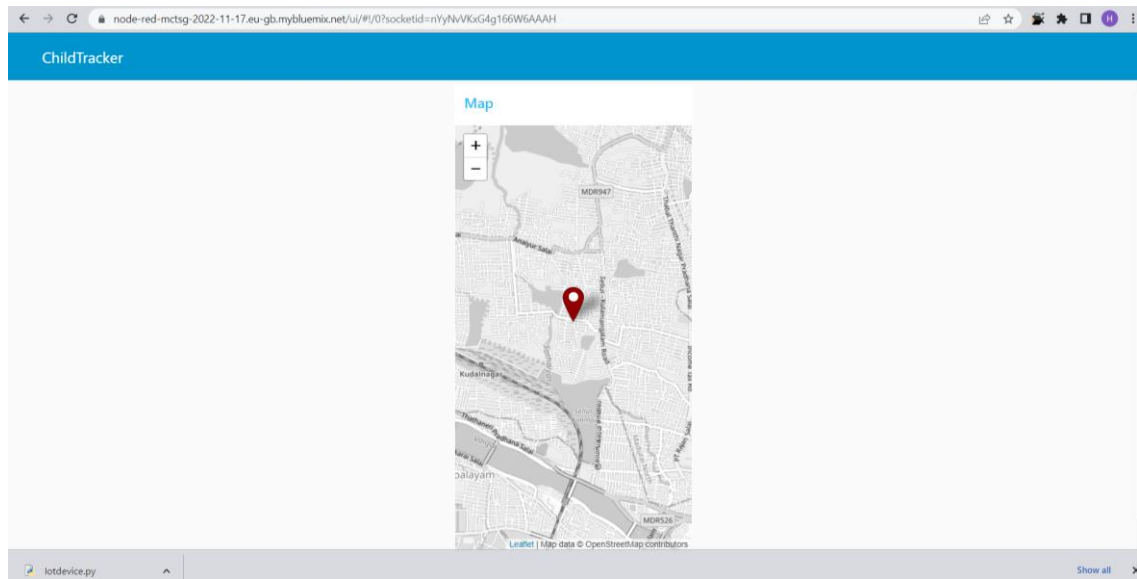
    #out area
    #latitude= 9.95143
    #longitude= 78.1158

    data={'name': name, 'lat':latitude,'lon':longitude}
    pub(data)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
```

```
*Python 3.7.2 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Lenovo\Downloads\iotdevice.py =====
Published data Successfully: %s2022-11-17 23:42:19,489 ibmiotf.device.Client
INFO Connected successfully: d:phsvng:CS1:1234
{'name': 'Childtracker', 'lat': 9.8796, 'lon': 78.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
Published data Successfully: %s {'name': 'Childtracker', 'lat': 9.8796, 'lon': 7
8.081}
|
```

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





Conclusion:

Developed the web application using Node-RED Successfully