**FINAL DELEIVERABLES**

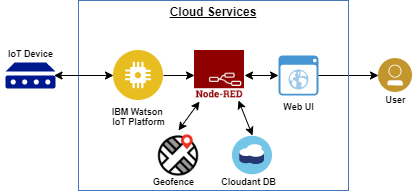
**IoT Based Safety Gadget for Child Safety Monitoring & Notification**

|  |  |
| --- | --- |
| Date | 18 November 2022 |
| Team ID | PNT2022TMID31875 |
| Project Name | IoT Based Safety Gadget for Child Safety Monitoring and Notification |
| Maximum Marks | 1. Marks |

**PROJECT OBJECTIVE:**

Child tracker helps the parents in continuously monitoring the child's location. They can simply leave their children in school or parks and create a geofence around the particular location. By continuously checking the child's location notifications will be generated if the child crosses the geofence. Notifications will be sent according to the child's location to their parents or caretakers. The entire location data will be stored in the database.

**TECHNICAL ARCHITECTURE:**



**PROJECT FLOW:**

|  |  |
| --- | --- |
| 1. | IOT DEVICE |
| 2. | IBM Watson IoT Platform |
| 3. | Node-Red |
| 4. | Cloudant DB |
| 5 | Alert SMS |

**1.IOT DEVICE**

**AIM:**

To get the coordinates of the child using GPS & ESP32

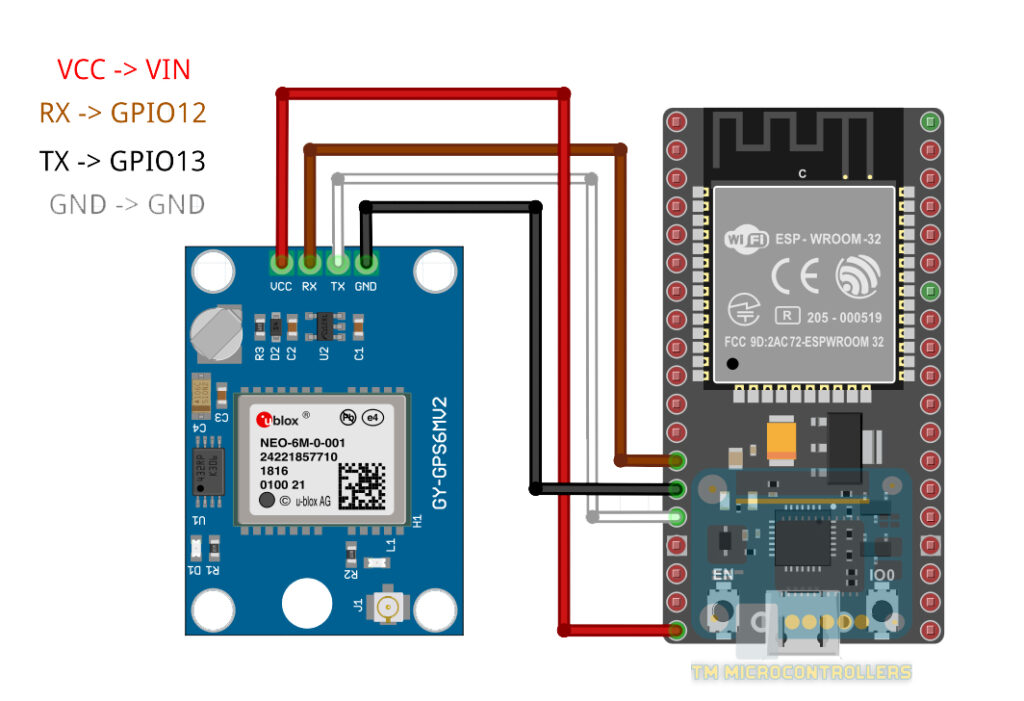
**HARDWARE REQUIRED:**

* ESP32-WROOM-32U
* [NEO-6M GPS Module](https://robu.in/product/ublox-neo-6m-gps-module/)
* Micro-USB Cable
* Connecting wires

**SOFTWARE REQUIRED:**

* Arduino IDE to run the program

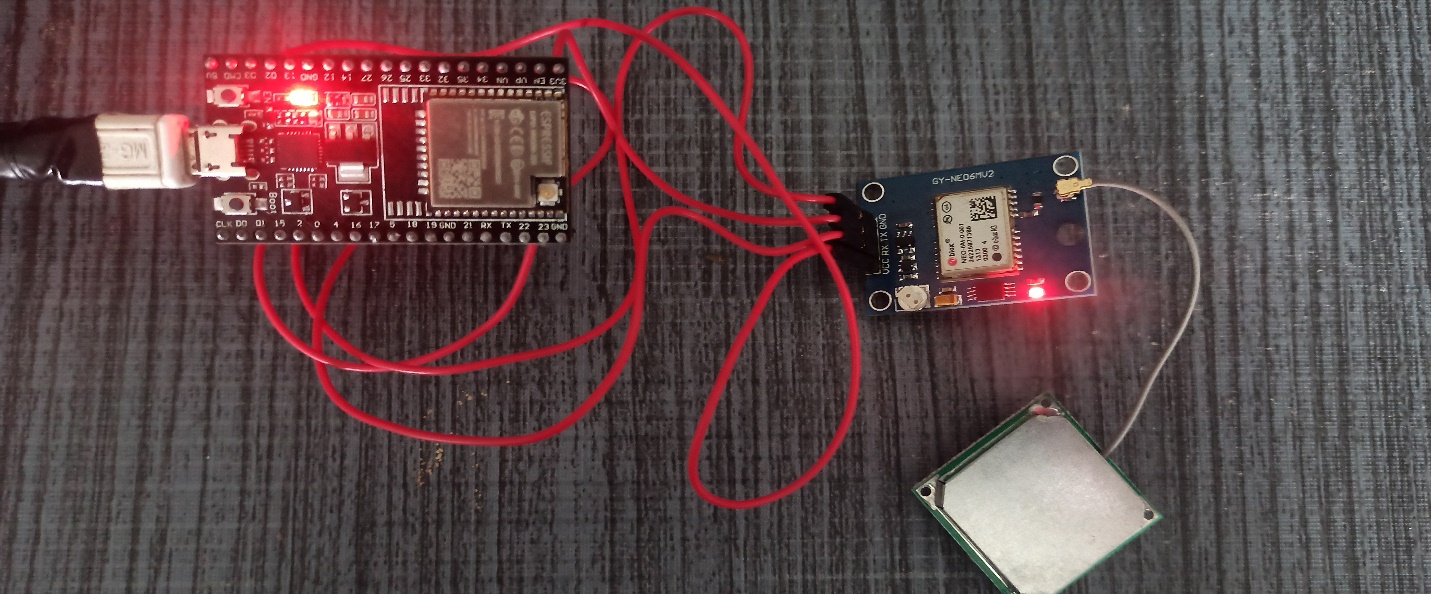
**CIRCUIT DIAGRAM:**



|  |  |
| --- | --- |
| **ESP32 board** | **NEO-6M Module** |
| VCC = 5V | VCC |
| RX0 | TX |
| TX0 | RX |
| GND | GND |

**PIN CONNECTION:**

**WIRE CONNECTION:**



**SOURCE CODE FOR COORDINATES:**

#include <TinyGPSPlus.h>

TinyGPSPlus gps;

void setup()

{

Serial.begin(9600);

Serial2.begin(9600);

delay(3000);

}

void loop()

{

while (Serial2.available() > 0)

if (gps.encode(Serial2.read()))

displayInfo();

if (millis() > 5000 && gps.charsProcessed() < 10)

{

Serial.println(F("No GPS detected: check wiring."));

while (true);

}

}

void displayInfo()

{

Serial.print(F("Location: "));

if (gps.location.isValid())

{

Serial.print("Lat: ");

Serial.print(gps.location.lat(), 6);

Serial.print(F(","));

Serial.print("Lng: ");

Serial.print(gps.location.lng(), 6);

Serial.println();

}

else

{

Serial.print(F("INVALID"));

}

}

void updateSerial()

{

delay(500);

while (Serial.available())

{

Serial2.write(Serial.read());//Forward what Serial received to Software Serial Port

}

while (Serial2.available())

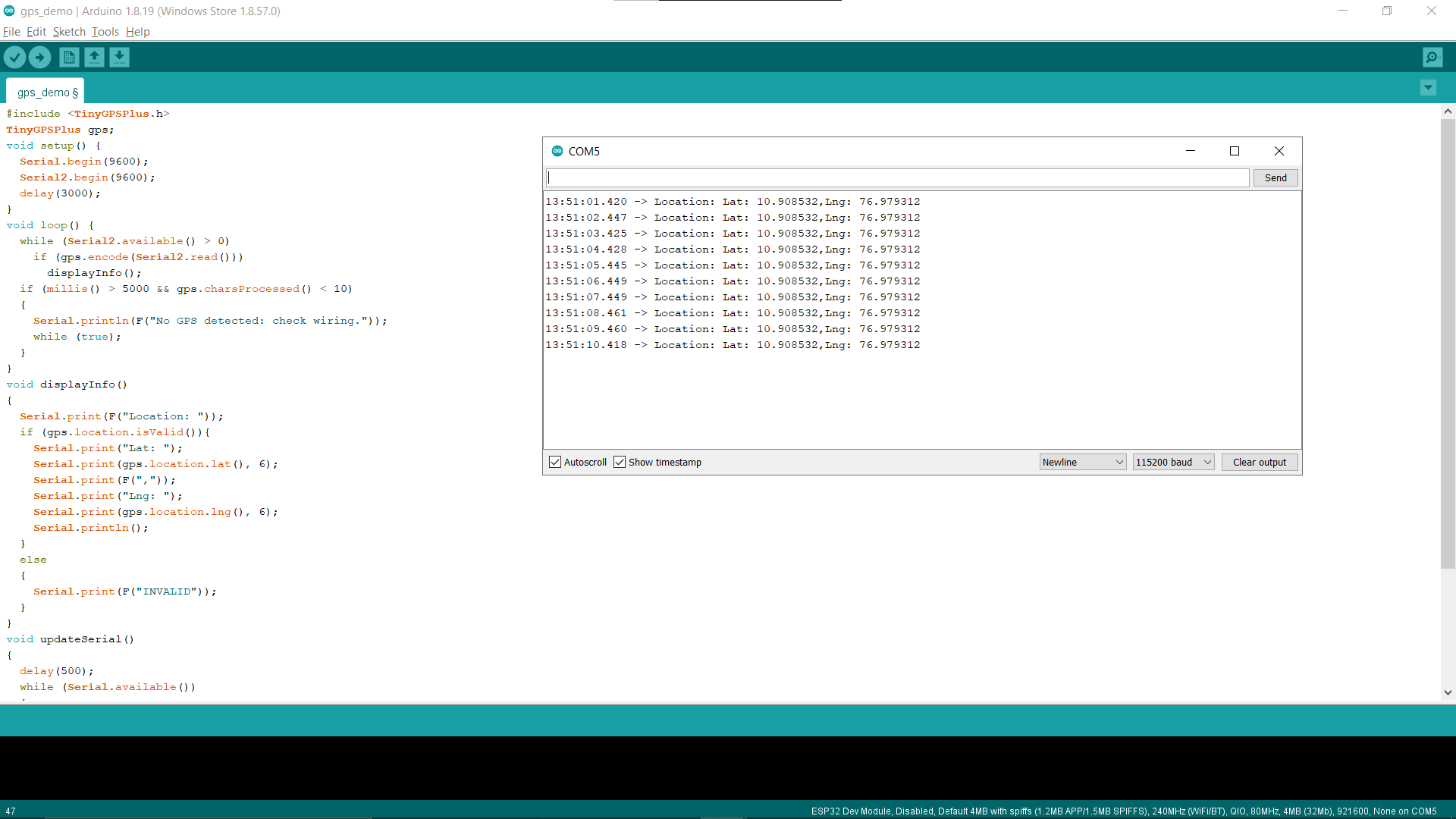
{

Serial.write(Serial2.read());//Forward what Software Serial received to Serial Port

}

}

**Output:**



To connect tracking device or manually programmed python code to IBM Watson IoT Platform.

**Source Code:**

import random as rand

import time

import ibmiotf.application

import ibmiotf.device

import sys

import imdb

#defining credentials of device

organization = "aa13kc"

deviceType = "Vijay2001"

deviceId = "1234567"

authMethod = "token"

authToken = "Yd-6ozY-S6BLhM0vkw"

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data['command'])

try:

deviceOptions = {"org" : organization, "type": deviceType, "id" : deviceId, "auth-method" : authMethod, "auth-token" : authToken}

deviceCli = ibmiotf.device.Client(deviceOptions)

except Exception as e:

print("Caught exception connecting device: %s" %str(e))

sys.exit()

deviceCli.connect()

while True:

name= "Vijay2001"

lat= 10.908532

lon= 76.979312

#Manually sending the coordintes

data = {'name':name,'Latitude' : lat,

'Longitude': lon}

def myOnPublishCallback():

print("Published all data to IBM Watson :",lat," ,",lon)

success = deviceCli.publishEvent("Iottracker","json",data,qos=0,on\_publish=myOnPublishCallback)

if not success:

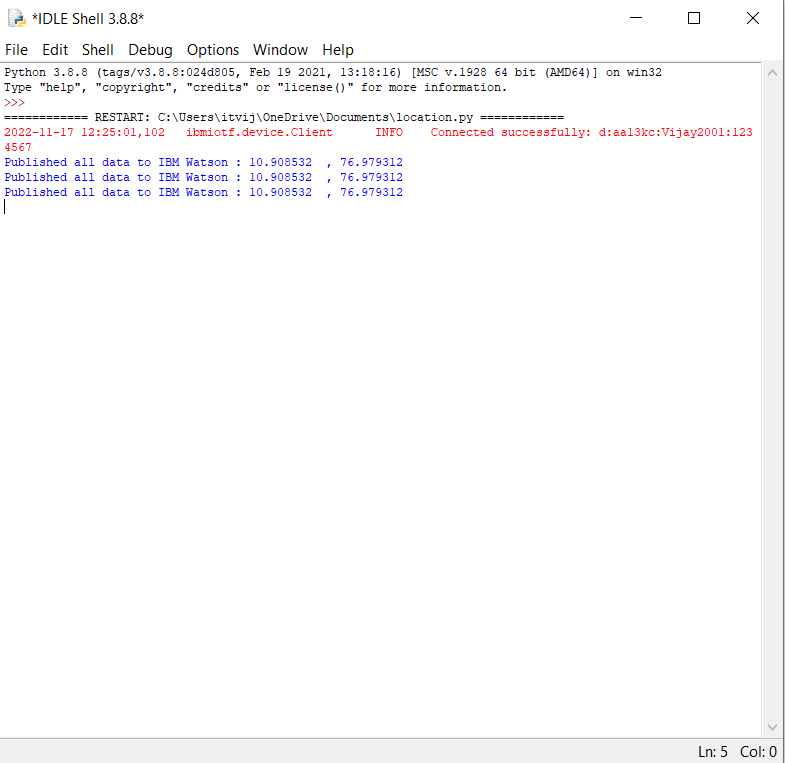
print("Not connected to IoT Device")

time.sleep(5)

deviceCli.commandCallback = myCommandCallback

deviceCli.disconnect()

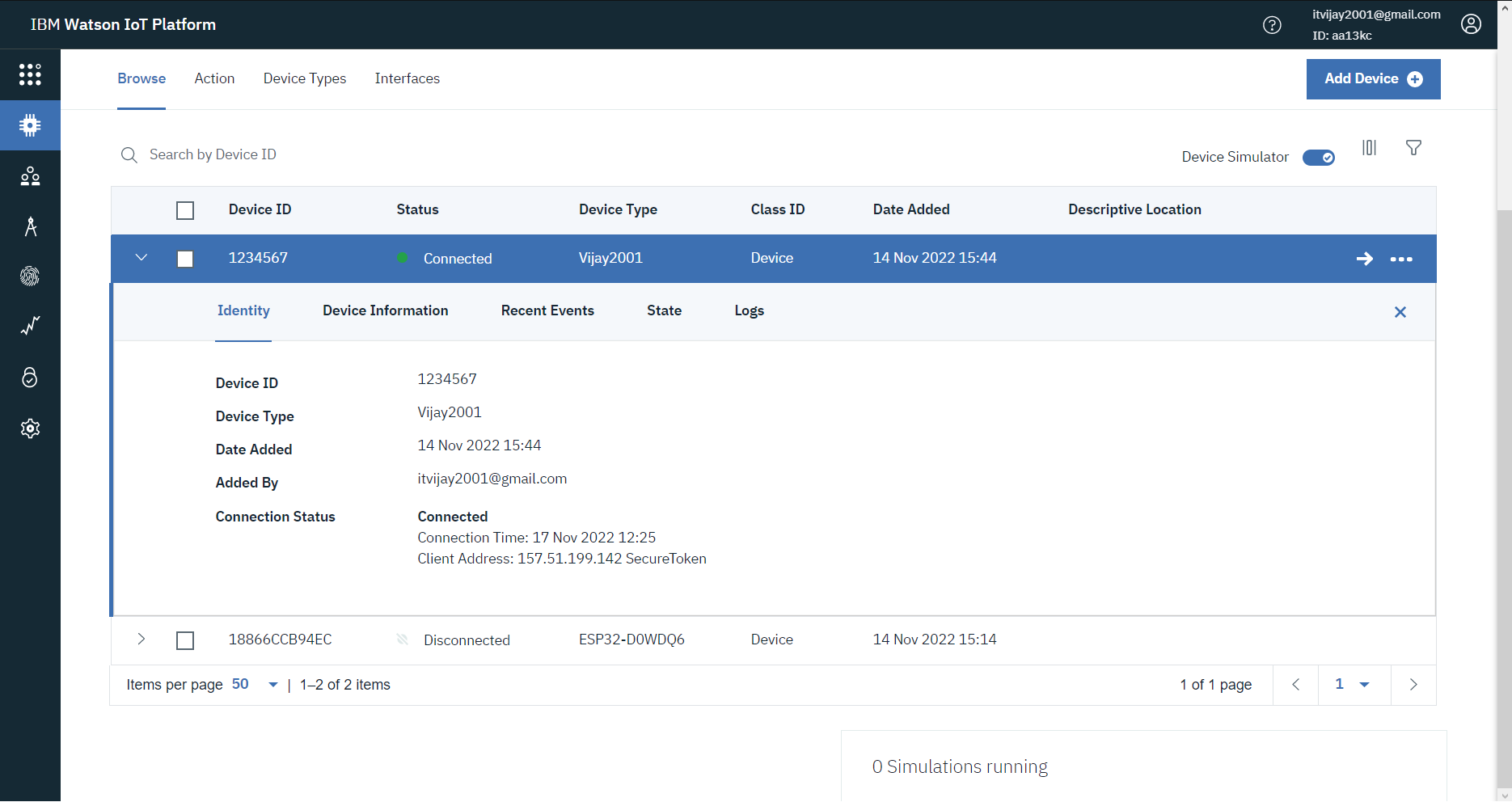
**Program output:**

****

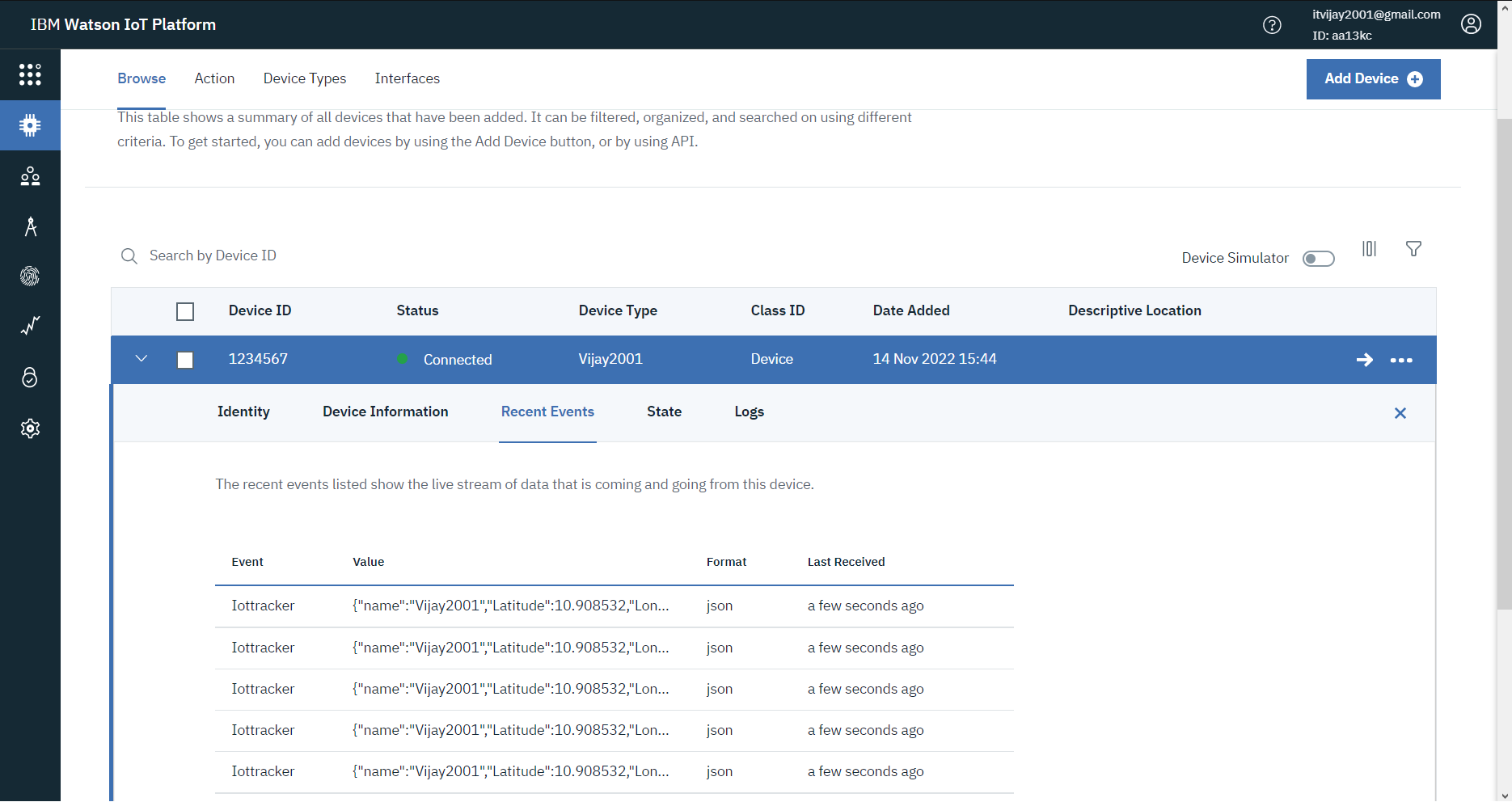
**2.IBM Watson IoT Platform**

**IBM Watson IoT:**

* IBM Watson IoT platform acts as the mediator to connect the web application to IoT device, so create the IBM Watson IoT platform.
* In order to connect the IoT device to the IBM cloud, create a device in the IBM Watson IoT platform and get the device credentials.
* Device connected to IBM Watson IoT Platform.

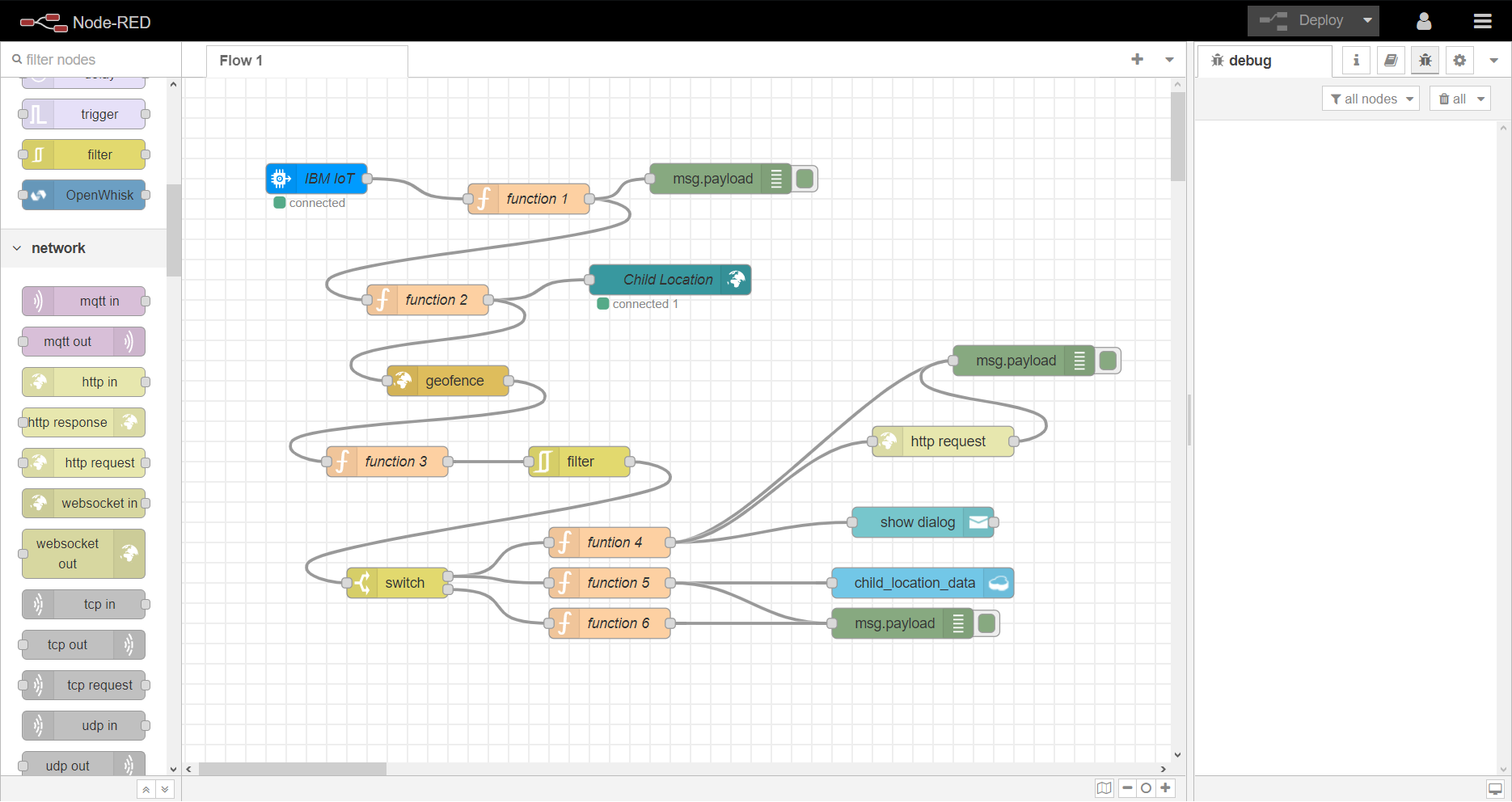
****

* Getting coordinates as output in IBM Watson IoT Platform.

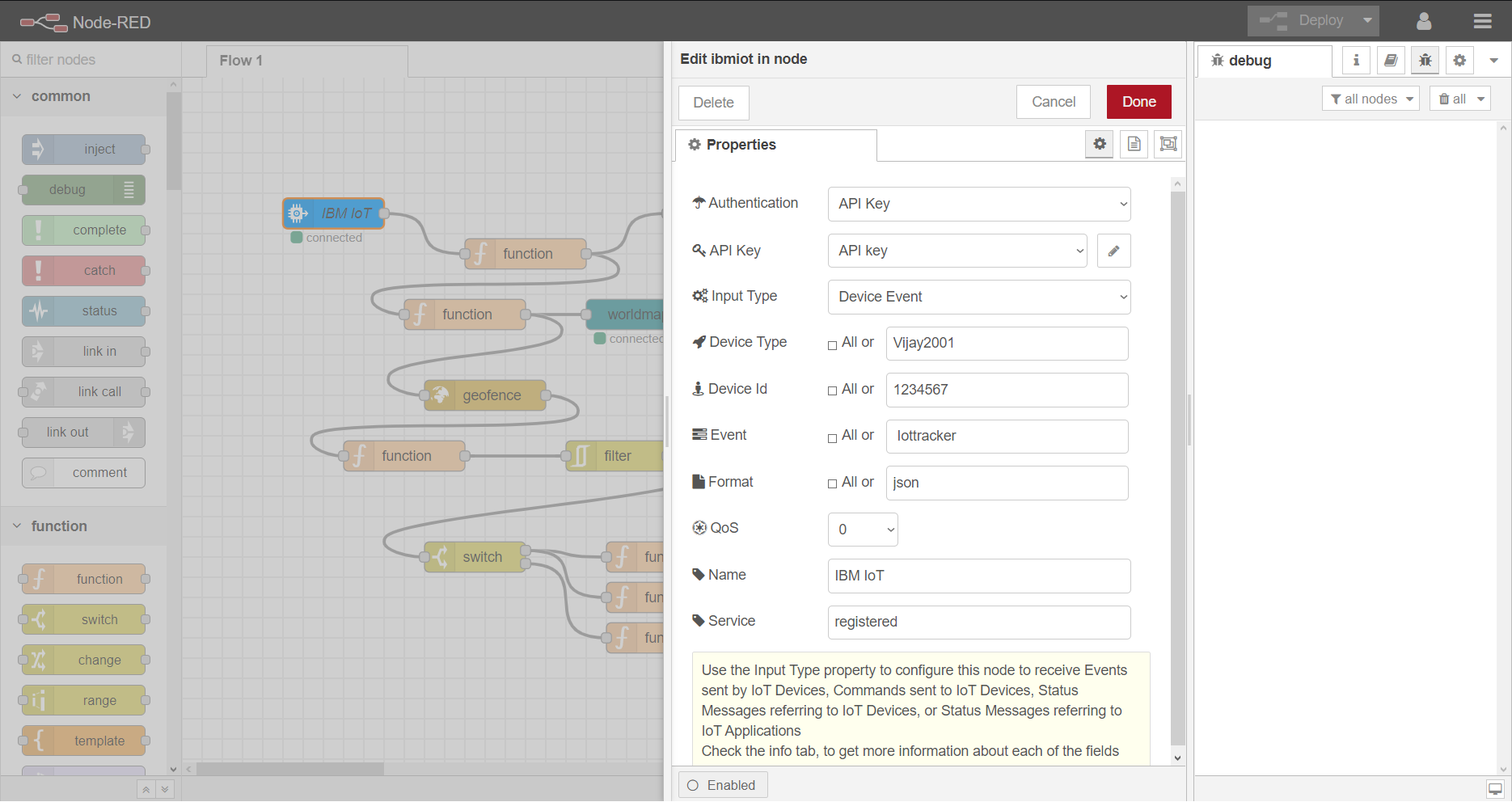
**3.Node-Red:**

To connect IBM Watson IoT Platform with Node-red to locate the child and create geofence on it.

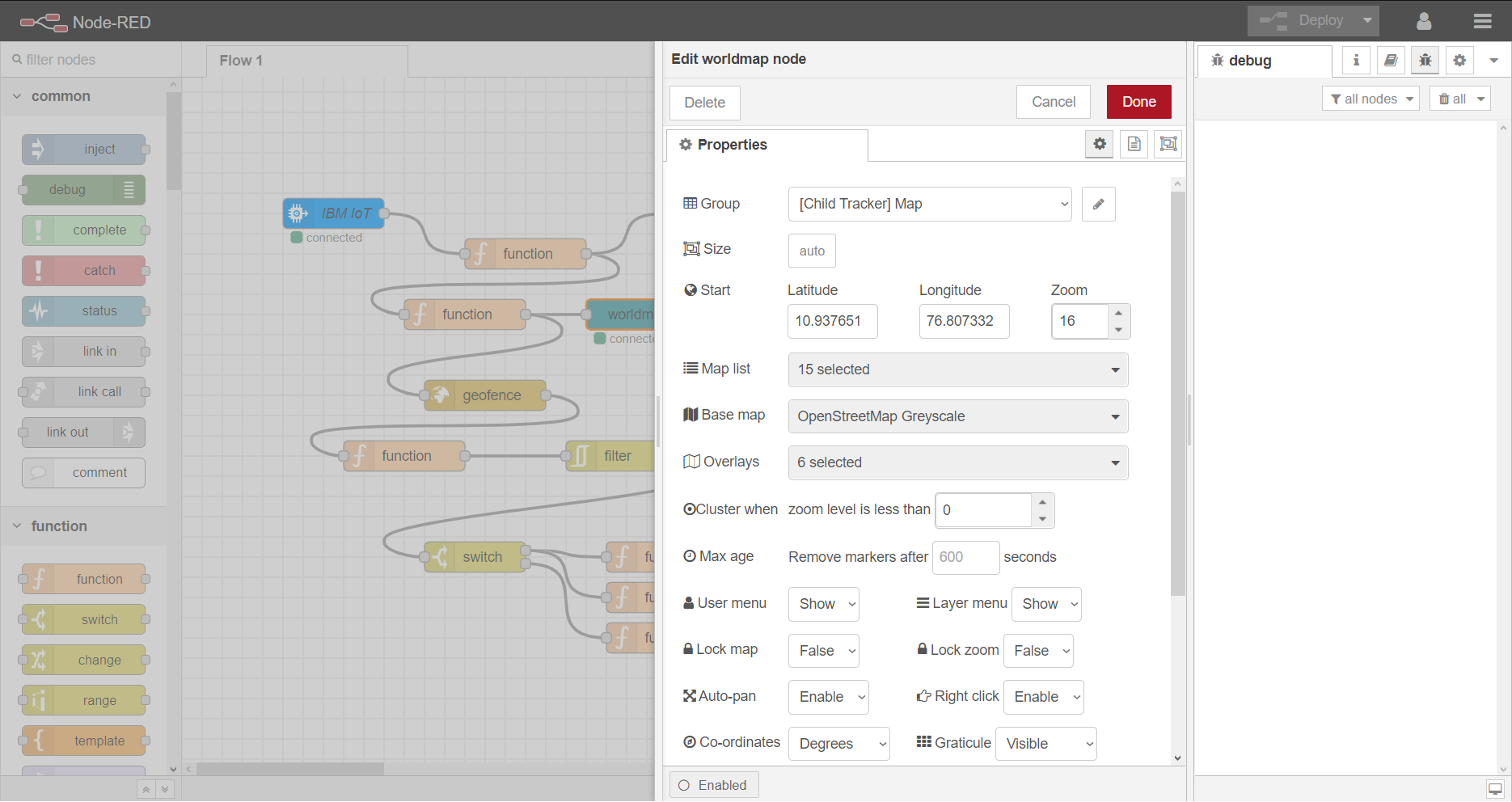
* Created node-red flow using worldmap, geofence, cloudant, and http request to locate the child.



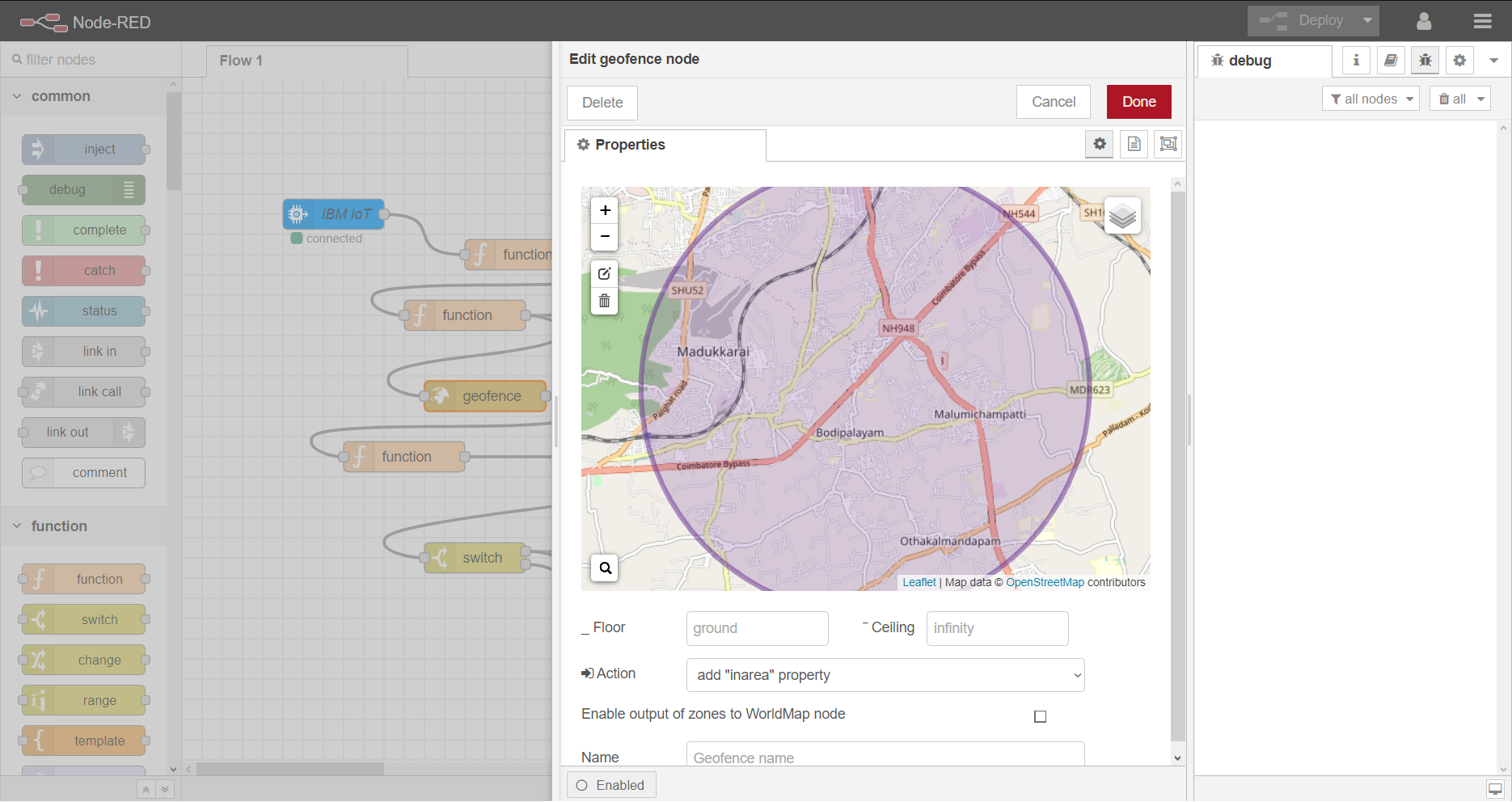
* Connected IBM IoT node in node-red to IBM Watson IoT using device credentials.



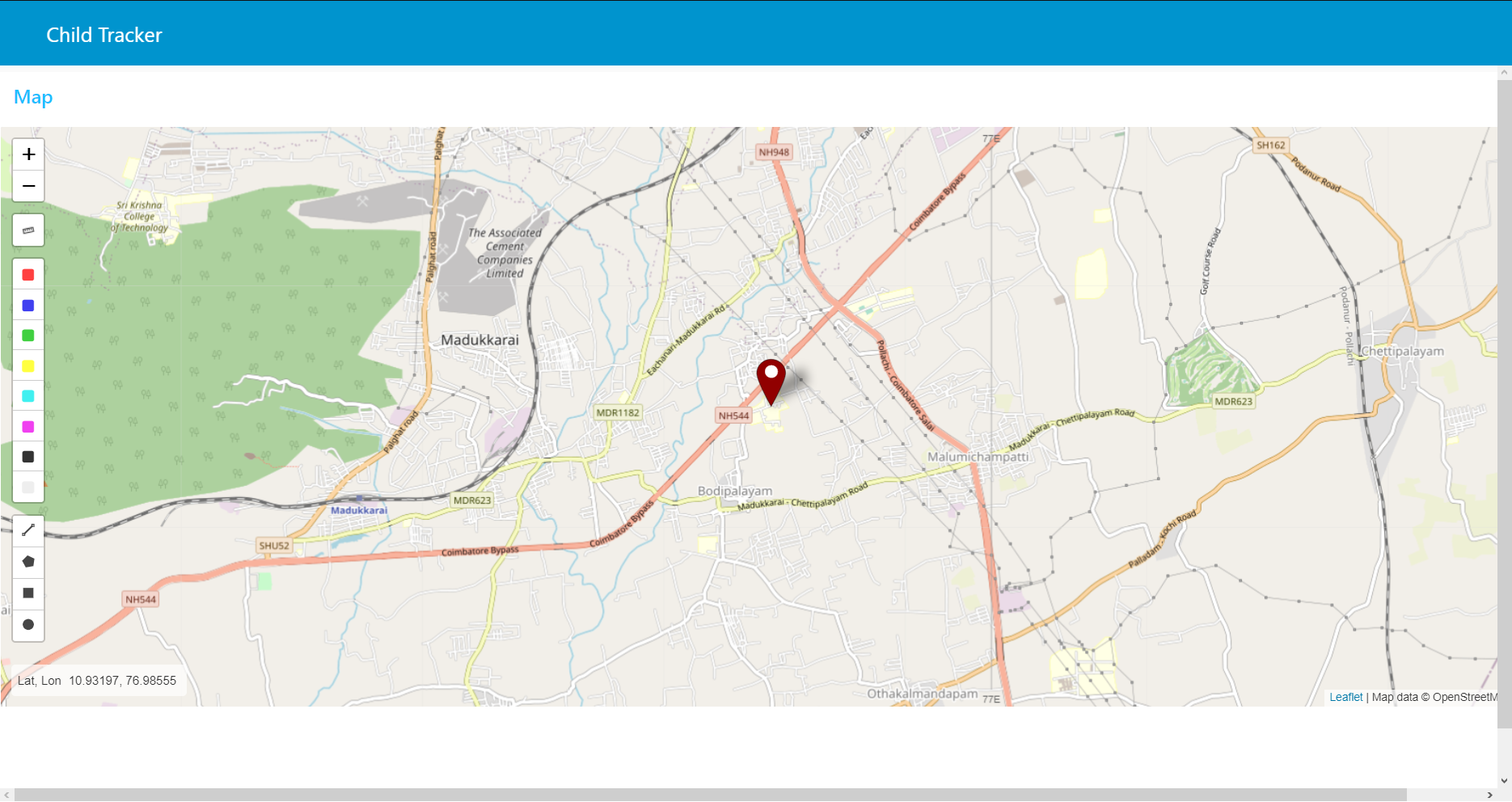
* Created worldmap\_ui to show the child location in user interface page of node-red



* Connected geofence to keep the child safe inside the parent or caretaker’s monitoring

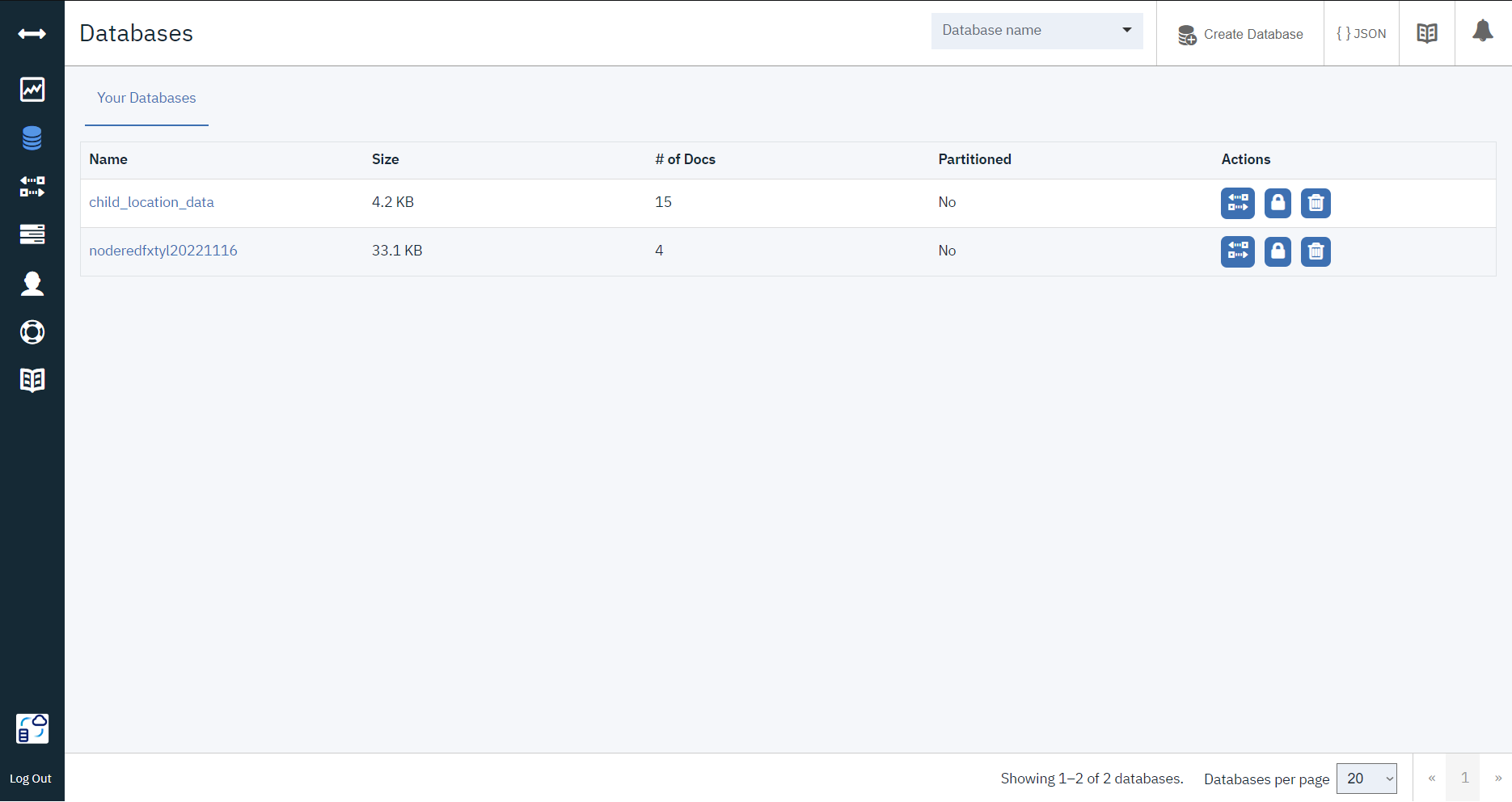


* You can see the child location in the ui page of node-red

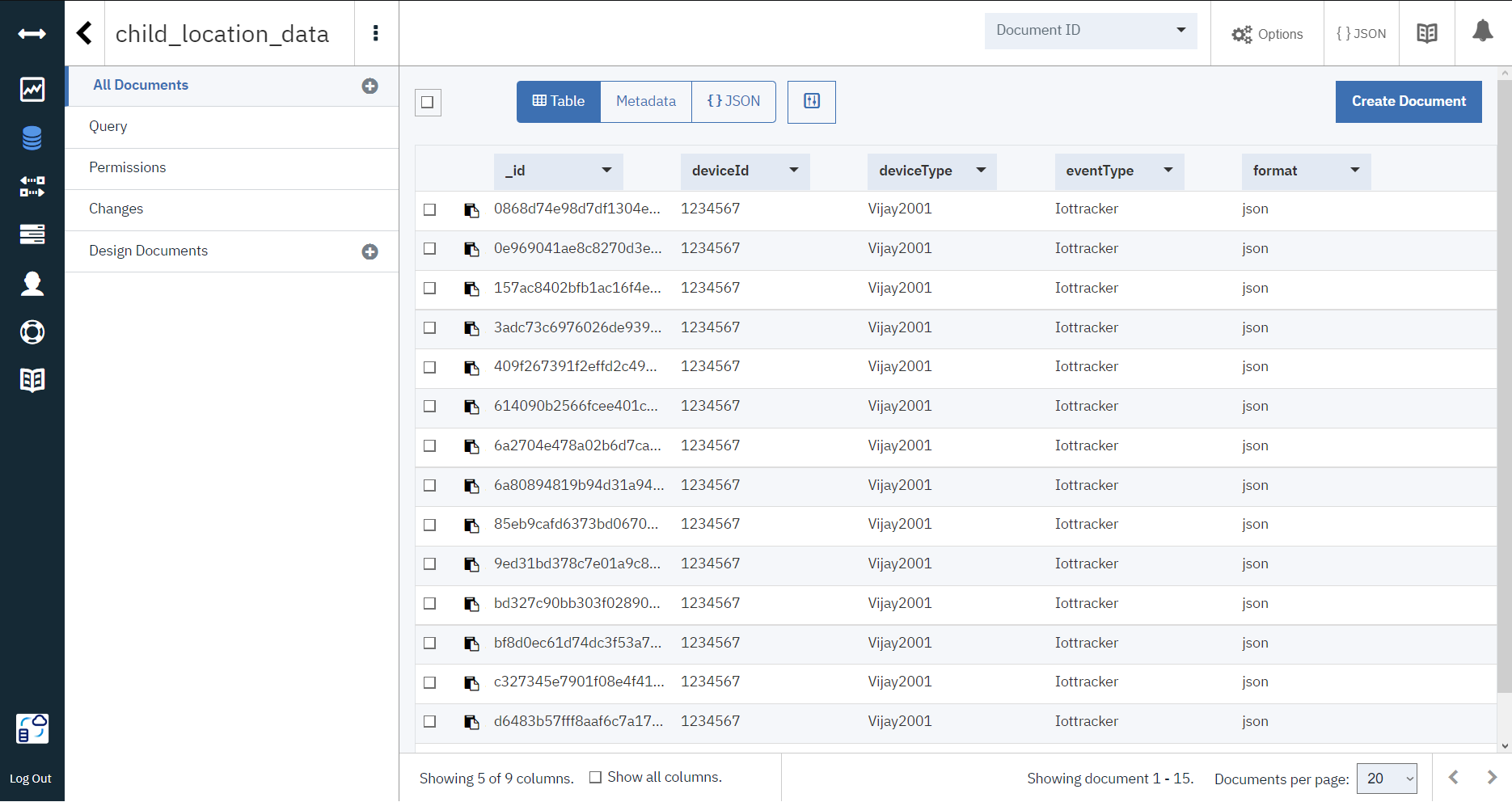


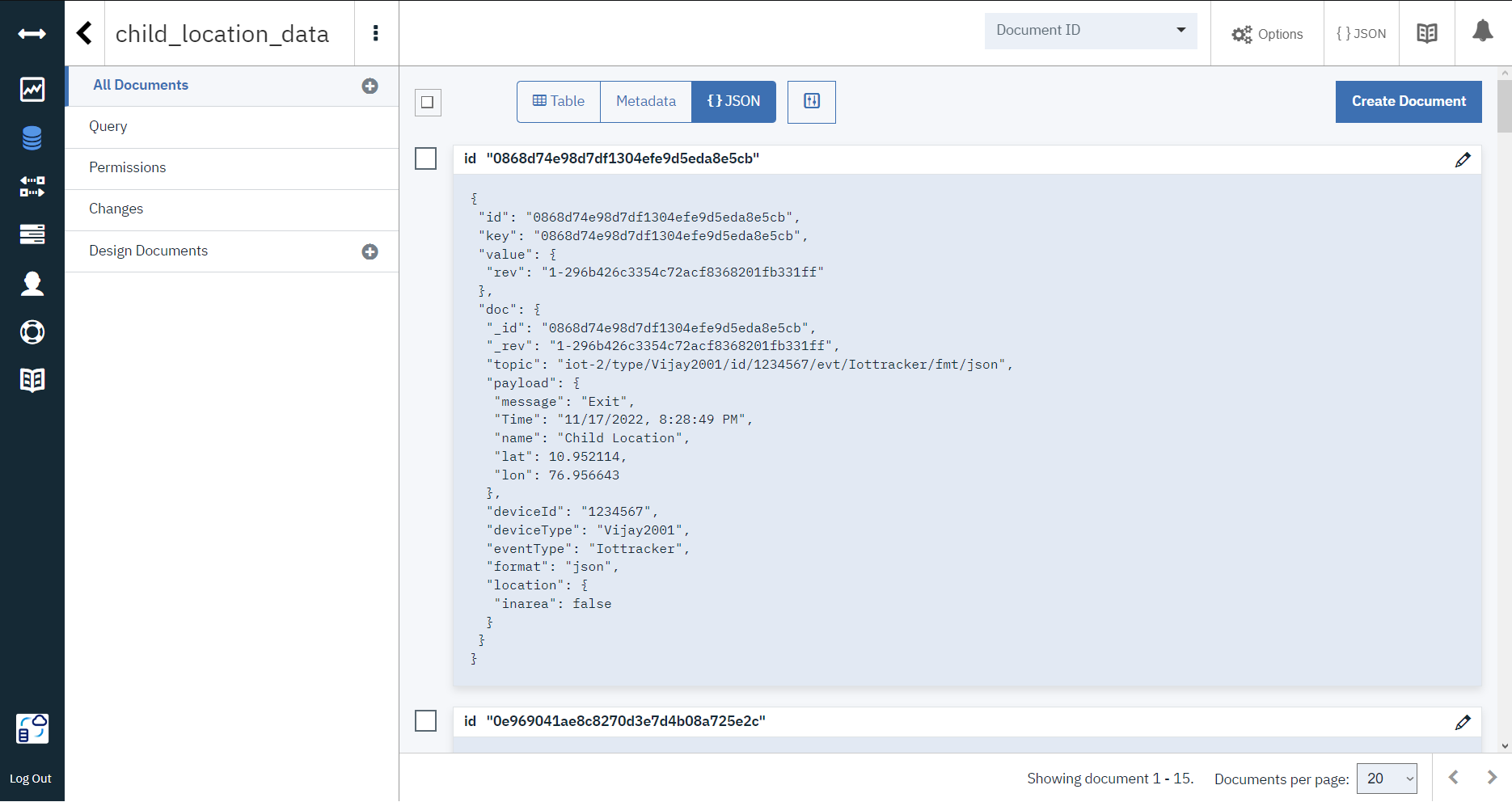
**4.CLOUDANT DB:**

* Created database in IBM cloudant service as child\_location\_data.



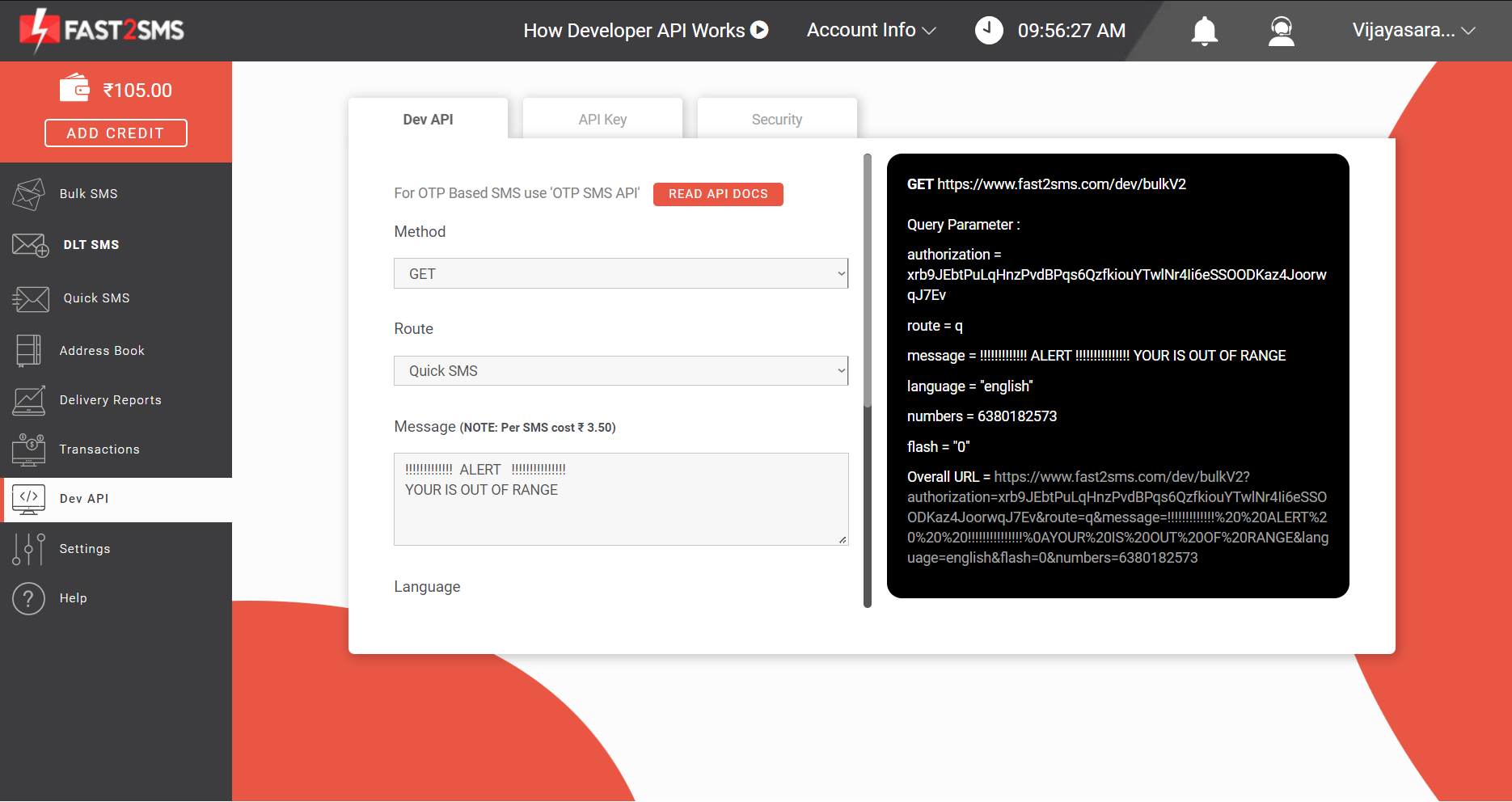
* Store the location coordinates in database



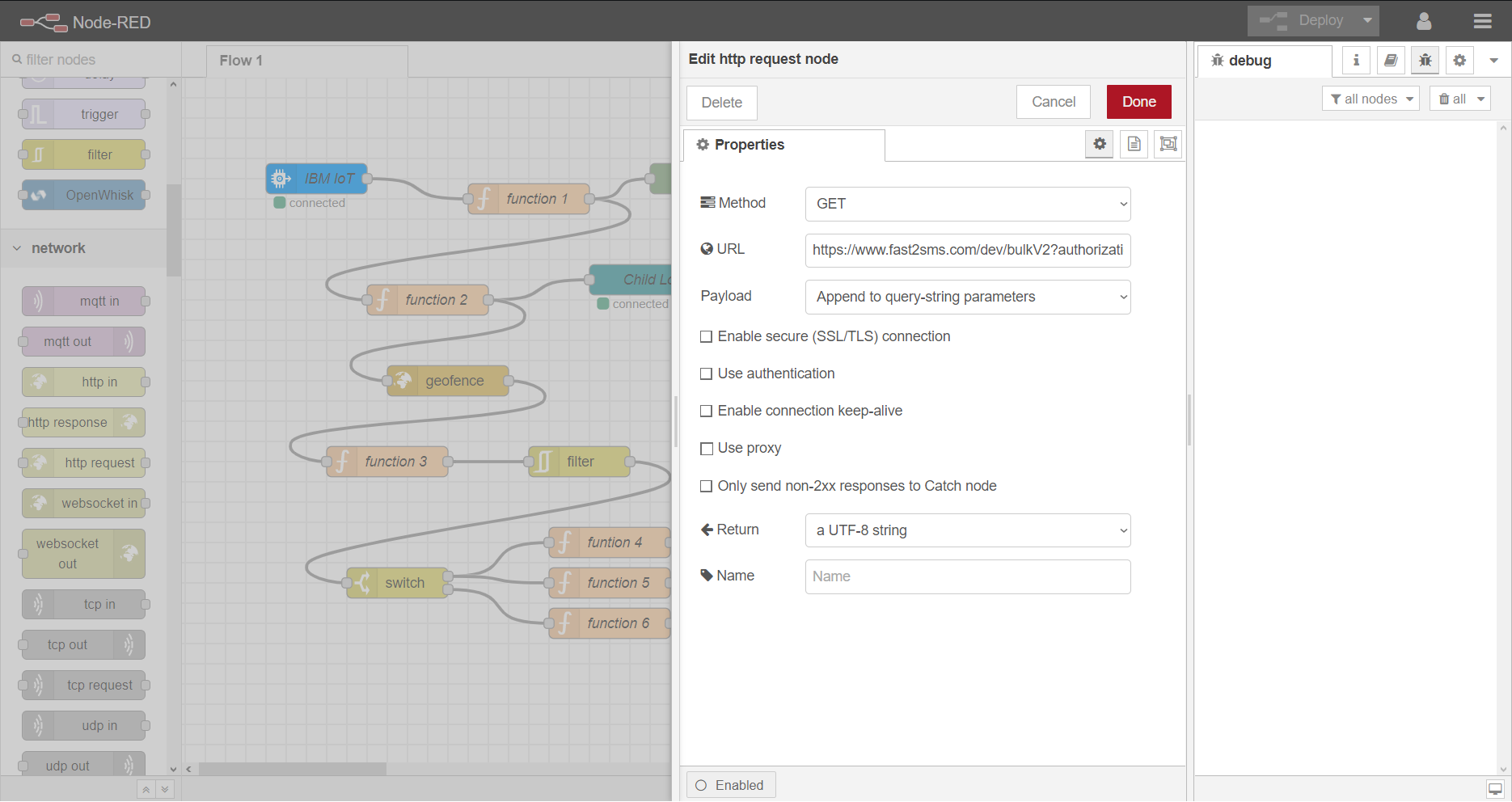


**5.ALERT SMS:**

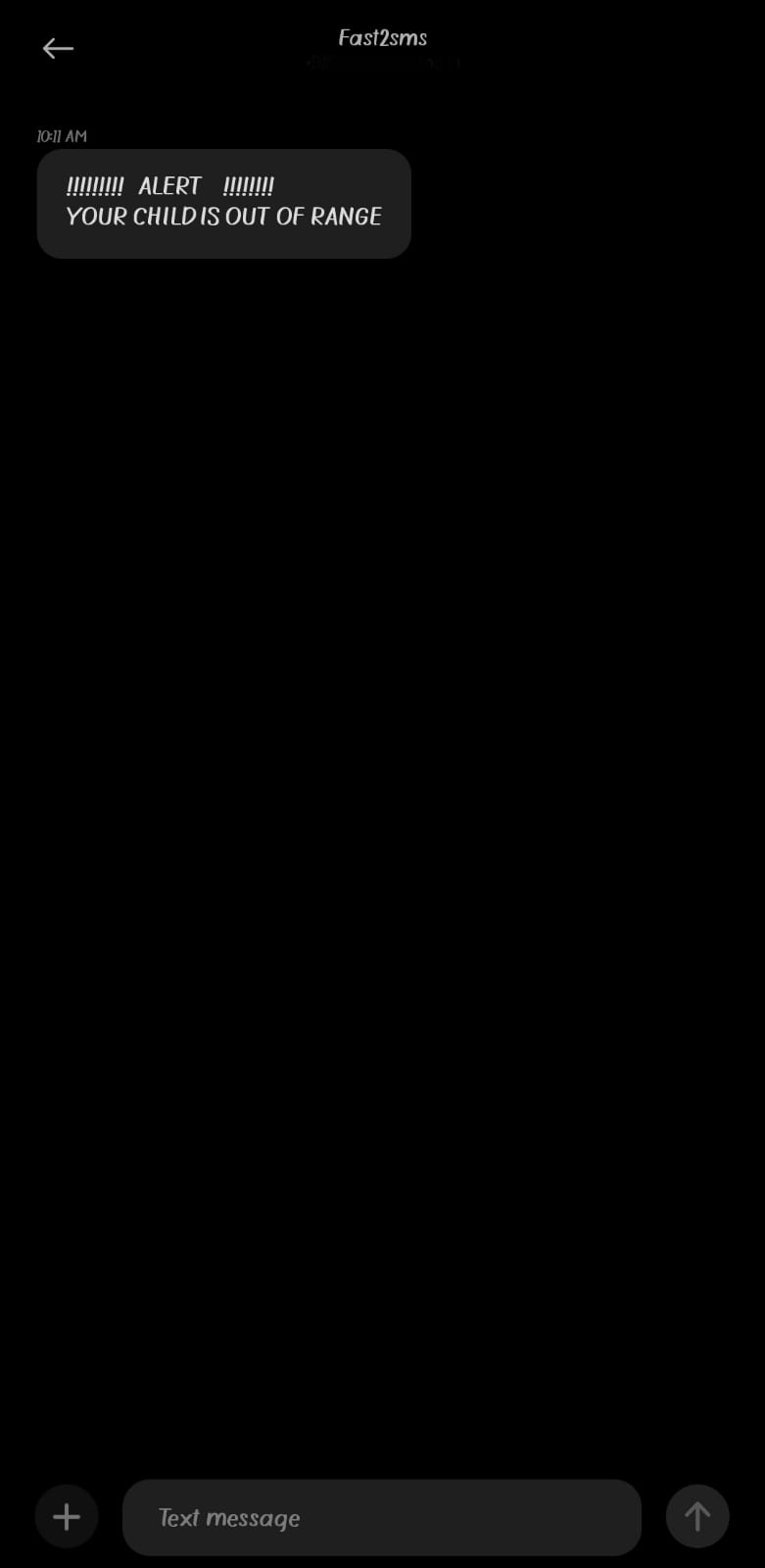
* Created Dev API ULR to send message through Fast2sms services.



* Paste the url in http request node in node red.

****

**SMS RECEIVED TO MOBILE PHONE:**

****