Department of Electronic & Telecommunication Engineering

University of Moratuwa

EN 2560 – Internet of Things Design and Competition

Tracking the location of the user and sending a notification of the person if he is in a Covid red zone

Group-18

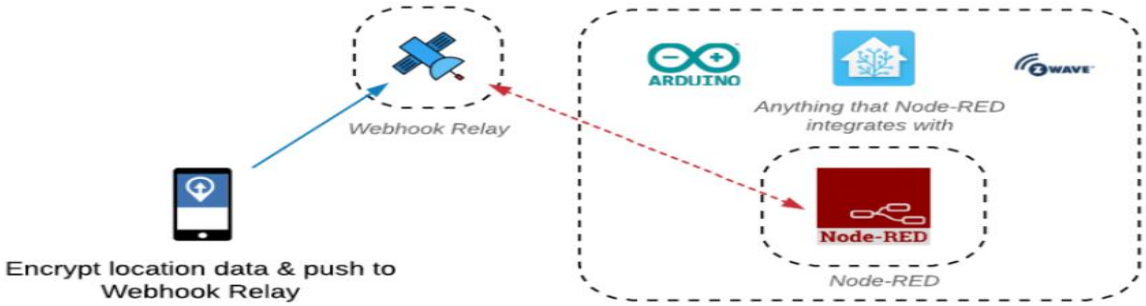
Group Member	Index Number
P.Samarasinghe	180554B
K.A.G.D.Maheekumara	180379R
K.K.D.R.P.Kannangara	180302D

Problem to be solved

Best way to get rid of covid pandemic is staying at home, but we were forced to move out of the home in order to satisfy our basic needs and wants. This application which Track the location of a person and send a notification to the user if he enters or leaves a covid red zone (Location with high density of covid patients) will help to avoid covid spread to a certain extent.

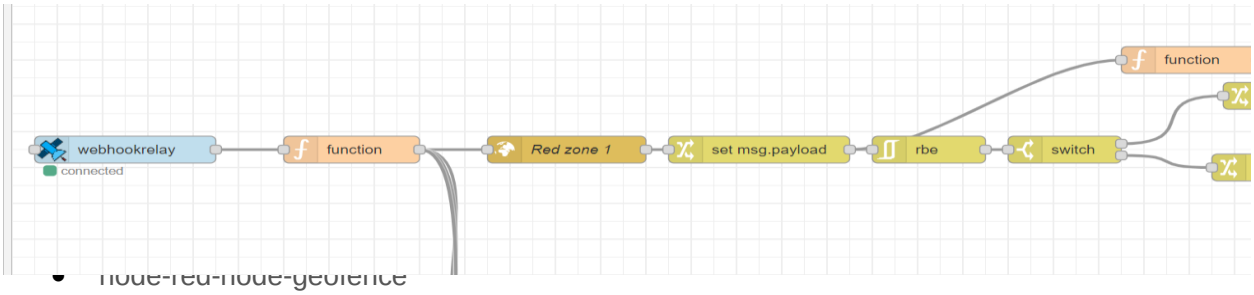
System overview and operation

Open source API to the Node-red



We use a mobile phone application called Owntracks which helps us to track the location of the person. The OwnTracks app runs in the background on the mobile device and waits for the smart phone to tell it that the device has moved, whereupon OwnTracks sends out a message with its current coordinates. OwnTracks can send payloads either to an MQTT server or a standard HTTP endpoint. But here according to the project requirements we use HTTP for sending payloads. Our location data will be relayed to Node-RED through a public Webhook Relay address. Webhook Relay provides public endpoints which can accept webhooks and then, based on user defined rules, forward them to either public or internal destinations.

Node-Red function



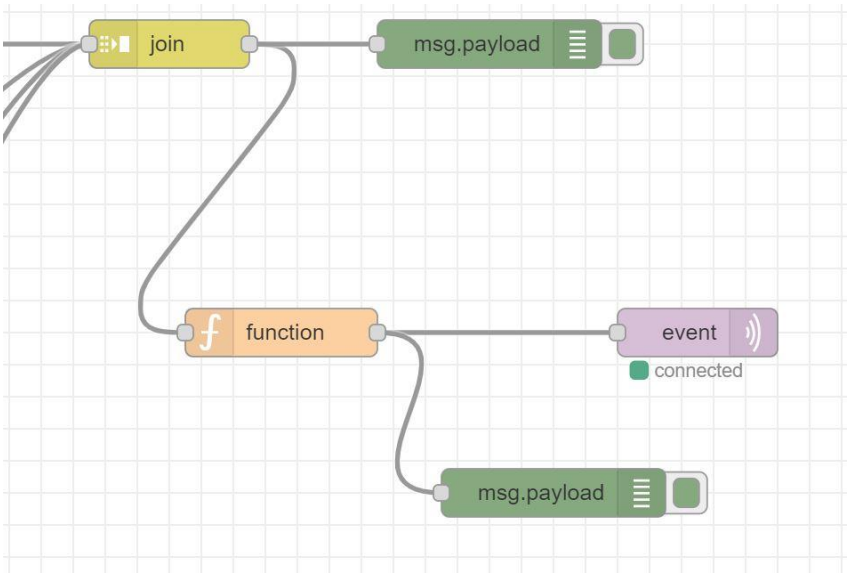
Webhook node which subscribe to webhooks receives the payload as a string the function node will convert it to a JSON format. The Geofence node (Red zone 1) will locate the user's location according to the latitude and longitude value in the payload. If the user is inside the geofence

which was selected according to the covid risky areas in the country msg.inarea sets to true else set to false. With this we can identify whether the user has reached a risky area.

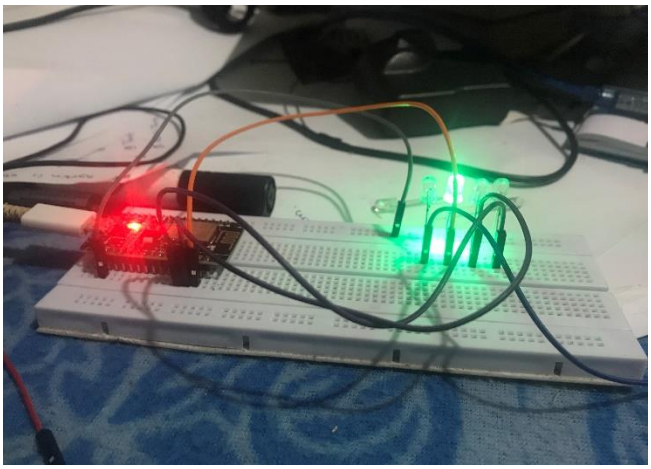
API keys used

Webhooks API: Webhook API allows us to subscribe to ebents happening in the owntracks application. Rather than making API call when event happens this can send HTTP requests to the endpoint. Webhooks are more scalable than regularly polling for changers.

Node Red to Node MCU

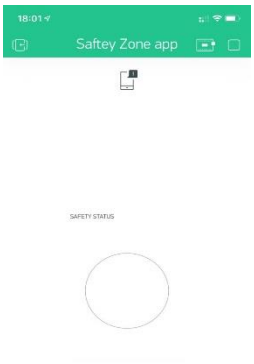


MQTT clients are very small, require minimal resources so can be used on small microcontrollers. MQTT message headers are small to optimize network bandwidth. So we use MQTT as the communication protocol between the node red and Node MCU. Here Node-Red act as the server ESP 8266 Act as the client. And we use Mqtt brokers. Node-red publishes to the topic location and the esp which act as the client subscribe to this topic which helps to receive data to the esp 8266. The function was tested by LEDs connected to the NodeMCU.



NodeMCU to mobile phone

After the payload (Here payload is 0 or 1 ; 0 when the person is not in the red zone and 1 when the person is in the red zone) is received to the NodeMCU , the payload is analyzed to check whether the payload is 1 or 0. If it's 1, then commands are sent to the blynk app created in the phone. Blynk uses a customized TCP/IP protocol. In the blynk app , our custom app is created to indicate the safety by a virtual bulb. The bulb would turn red if the payload received is equal to 1. And the notification is sent as well stating that the person is in a red zone.



Implementation including hardware, software flow, protocols, online resources

Hardware required: User's Mobile phone, Computer, Node MCU, Bread Board

Software required: Owntracks application, Node-red, Arduino , Blynk application

Online resources: Api keys, MQTT brokers

Protocols: HTTP, MQTT,TCP

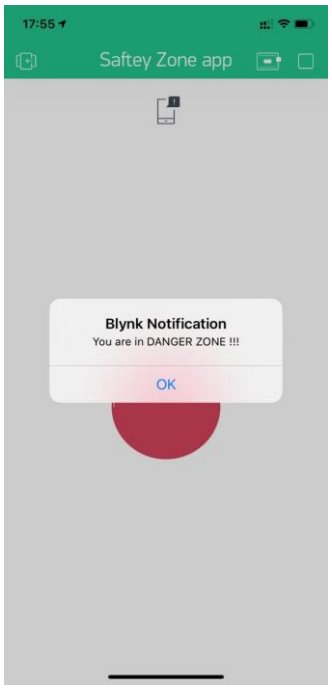
For this implementation we require to access the location of the user through a softaware called owntracks. After that we made a webhook relay account. Our location data will be relayed to Node-RED through a public Webhook Relay address Here we will use HTTP webhooks so we don't have to run a separate service(API keys obtained from the www.webhookrelay.com), Also webhook relay enables us to receive webhooks without having a public IP or configuring NAT. Geofence is created locally in the node red. For the communication between node red and the node-mcu we use light weight and efficient MQTT Publish Subscribe architecture. We used hivemq mqtt broker.

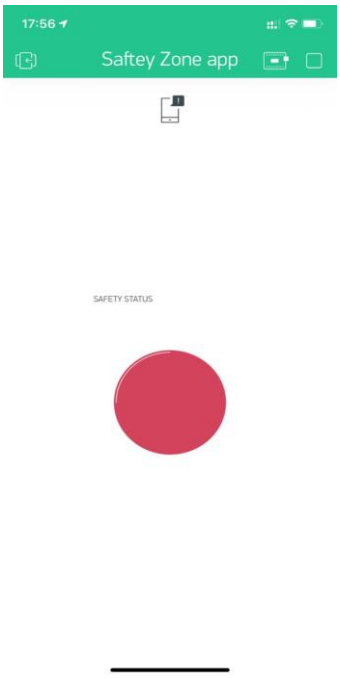
After that , from NodeMCU , the signals are sent using the Blynk platform. The signals are sent to the Blynk server by the NodeMCU and from there they are transferred to the mobile app , so the protocol used here is HTTP. By that , the Safety status bulb would be lit, and a notification would pop up.

Screenshot of the working prototype

1. We use OWNTRACKS app , to update our location.

2. When the person enters the danger zone the functionality of the app will be like this ; A notification will be received and the safety status bulb will be red.





3. The person will receive the message even though the blynk app is offline in his phone.