

# **Sign with smart connectivity for better road safety**

## **Sprint Delivery 1**

**Team id-PNT2022TMID47661**

### **1. Introduction**

In present Systems the road signs and the speed limits are Static. But the road signs can be changed in some cases. We can consider some cases when there are some road diversions due to heavy traffic or due to accidents then we can change the road signs accordingly if they are digitalized. This project proposes a system which has digital sign boards on which the signs can be changed dynamically. If there is rainfall then the roads will be slippery and the speed limit would be decreased. There is a web app through which you can enter the data of the road diversions, accident prone areas and the information sign boards can be entered through web app. This data is retrieved and displayed on the sign boards accordingly.

Software use    Arduino IDE

## 2. Problem Statement

The early effects to prevent road accidents and to ensure road safety includes the use of speed detection devices, CCTVs, speed limiters and emergency accident units. Old approaches emphasize the concept of problem-solving in Road safety, but it is more correct to recognize that Road safety activities doesn't solve problems. For instance, when a safer road design is implemented, hopefully the number of crashes, or their seriousness, will go down, but they will not disappear. It is more correct to say the implementation of correct policies, programs and measures will reduce numbers or consequences of crashes, but they will no be solved. This realization is important, because it changes the focus from a problem that will go away if we devote enough resources to it, to a situation requiring on-going management. This management in turn requires the development of scientifically based techniques, witch will enable us to predict with confidence that safety resources are well-spent and likely to be effective. The standard measures used in assessing road safety interventions are fatalities and killed or seriously injured (KSI) rates, usually per billion (10<sup>9</sup>) passenger kilometres. Vehicle speed within the human tolerances for avoiding serious injury and death is a key goal of modern road design because impact speed affects the severity of injury to both occupants and pedestrians.

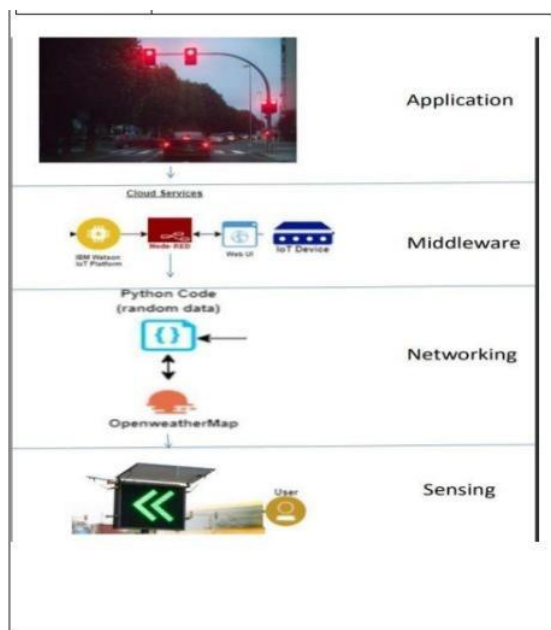
## 3. Proposed Solution

smart device to be used at accident-prone zones on the roads and highways. The intelligent road sensors monitor critical parameters like road visibility, road surface temperature, and rainfall in real-time. [Oizom Polludrone](#) needs to be installed at crossroads, tunnels and multi-level parking facilities to monitor vehicular emission. Weathercom and Polludrone push the real-time data obtained to the Oizom cloud. The top-speed limit is calculated based on weather speed index. The dynamic speed limit is then served to the drivers in the form of push notifications through maps, and Visual Messaging Displays. Effective vehicular route management is performed in case of higher pollution levels. Tunnel and parking ventilation are automated based on the pollution level.

## 4. Theoretical Analysis

### 4.1 Block Diagram

In order to implement the solution , the following approach as shown in the block diagram is used



### 4.2 Required Software Installation

## 4.2.A Node-Red

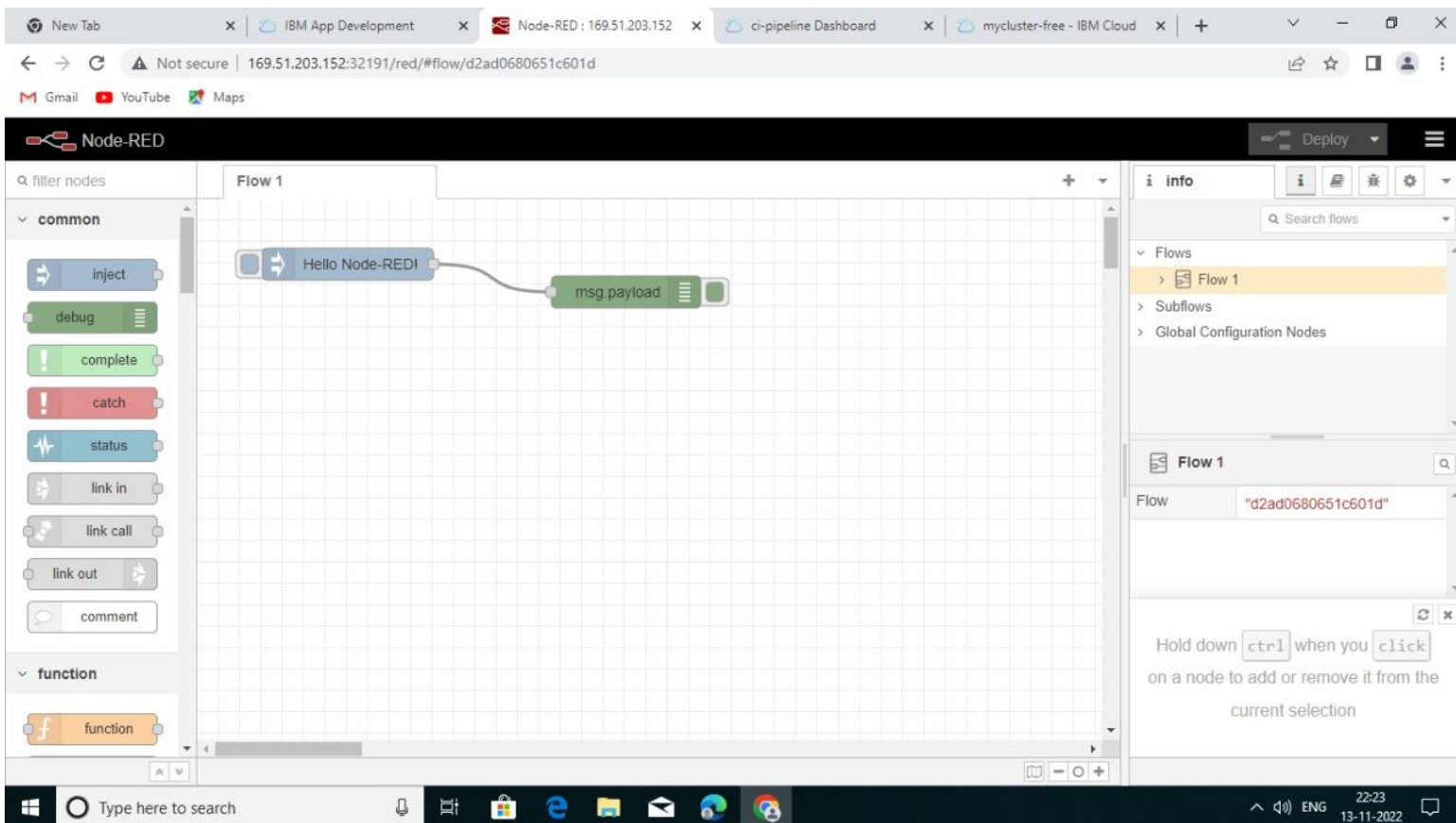
Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.

### Installation :

- First install npm/node.js
- Open cmd prompt
- Type => npm install node-red

### To run the application :

- Open cmd prompt
- Type=>node-red



- Then open <http://localhost:1880/> in browser

## Installation of IBM IoT and Dashboard nodes for Node-Red

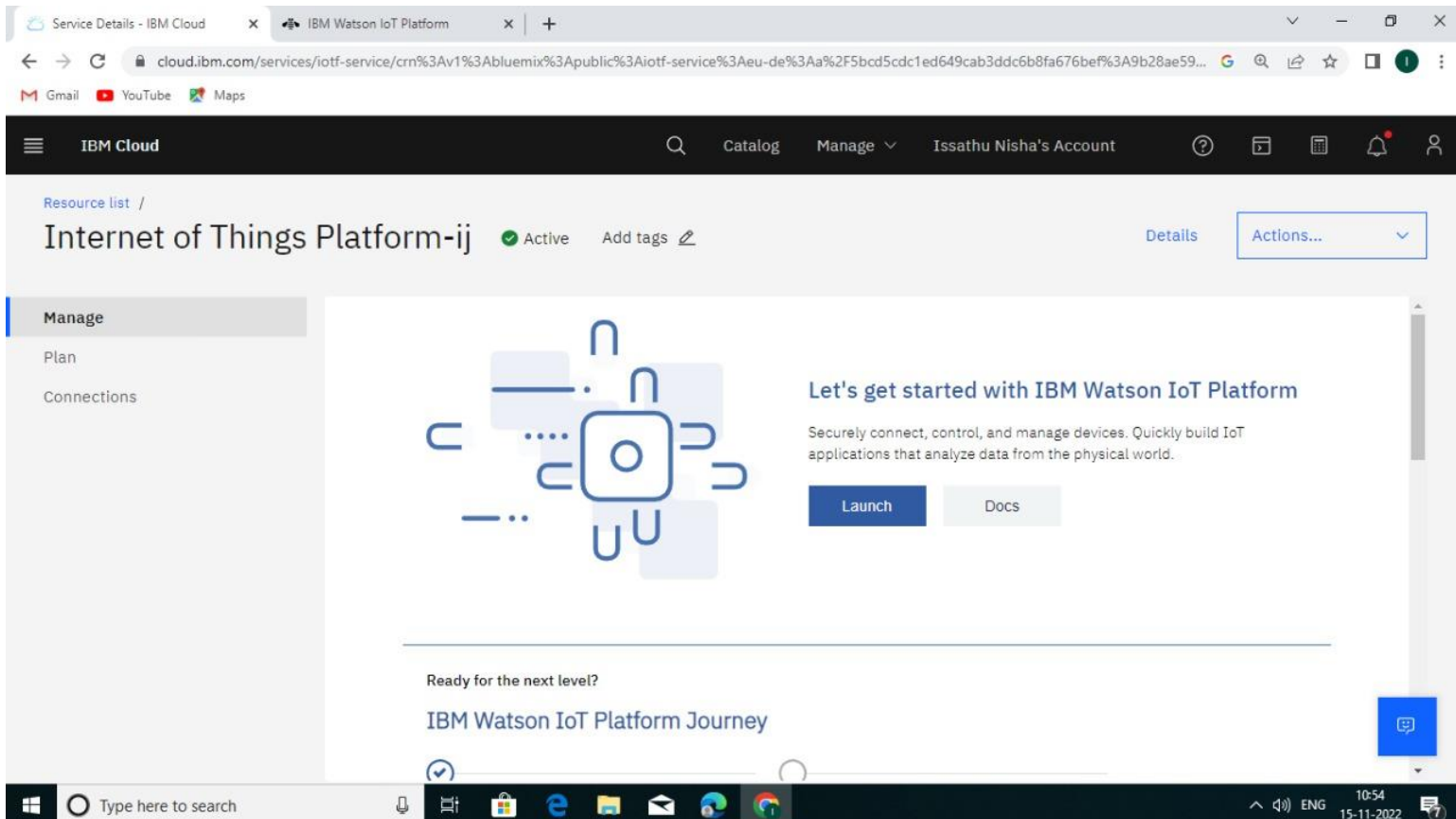
In order to connect to IBM Watson IoT platform and create the Web App UI these nodes are required

1. IBM IoT node

2. Dashboard node

### 4.2.B IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization and data storage. IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices.



The screenshot displays the IBM Cloud console for the 'Internet of Things Platform-ij' service. The interface includes a top navigation bar with 'IBM Cloud', a search bar, and user account information. The main content area features a large graphic of a central node connected to various devices, with the heading 'Let's get started with IBM Watson IoT Platform'. Below this, there is a 'Launch' button and a 'Docs' button. A sidebar on the left contains 'Manage', 'Plan', and 'Connections' options. The bottom of the screen shows the Windows taskbar with the search bar and several application icons.

**Steps to configure:**

- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloud account
- Launch the IBM Watson IoT Platform
- Create a new device
- Give credentials like device type, device ID, Auth. Token
- Create API key and store API key and token elsewhere.



lldown - 12



authentication token.

Find out how to add these credentials to your device [↗](#)

## Connection Information

Basic connection information about this device.

Device ID	12
Device Type	abcd
Date Added	Oct 29, 2022 12:21 AM
Added By	911519104014min@msec.org.in
Connection Status	Disconnected



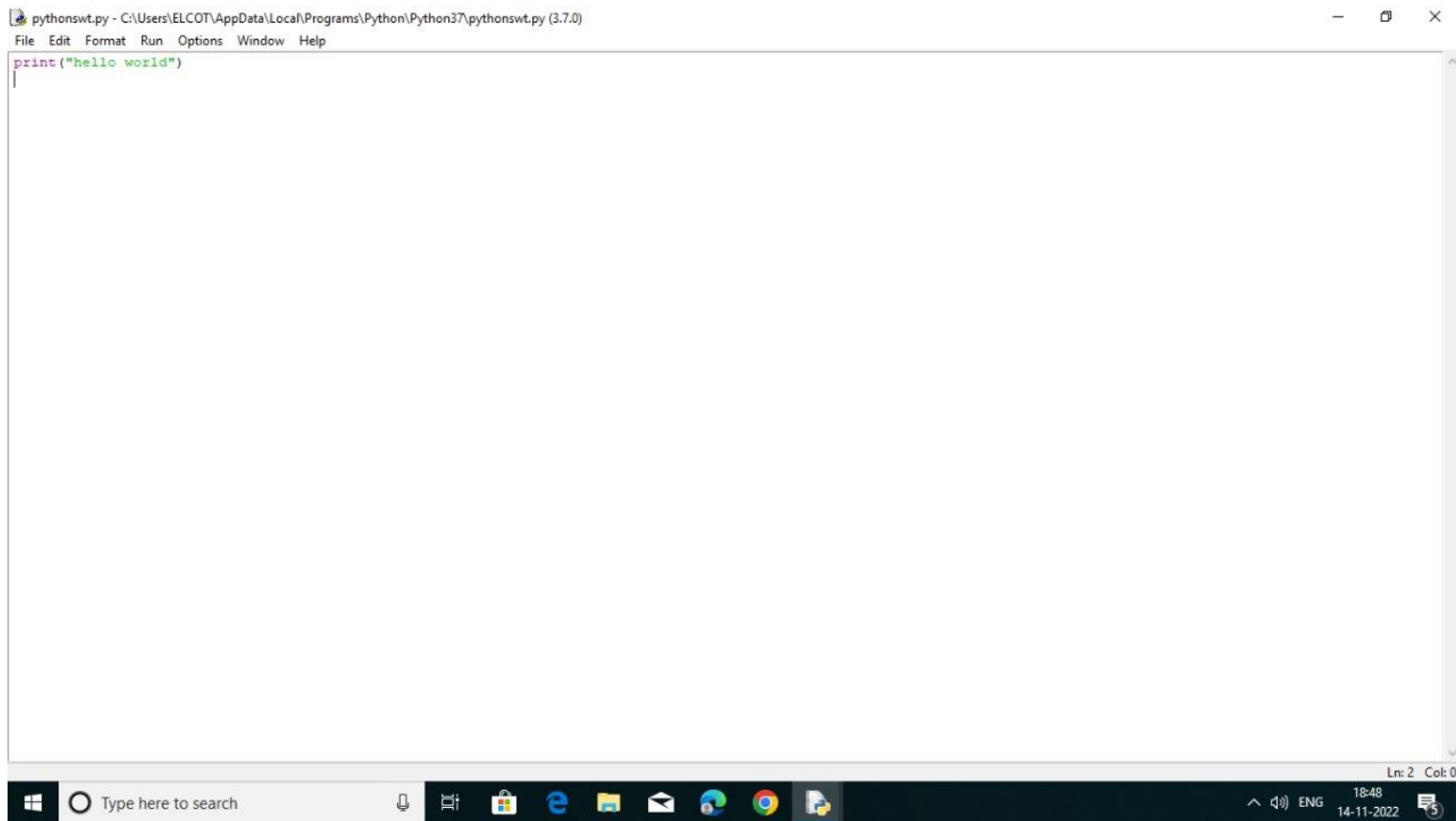
arch



## 4.2.C Python IDE

Install Python3 compiler

Install any python IDE to execute python scripts, in my case I used Spyder to execute the code.





**Code:** import time

import sys import

ibmiotf.application import

ibmiotf.device import

random

#Provide your IBM Watson Device Credentials

organization ID= "Onyujc" deviceType =

"abcd" = "13" authMethod = "usetoken-auth"

authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd): print("Command

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

status=cmd.data['command'] if status=="motoron":

print ("motor is on")

elif status == "motoroff":

print ("motor is off")

else :

print ("please send proper 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()

# Connect and send a datapoint "hello" with value "world" into the cloud as an
event of type "greeting" 10 times deviceCli.connect()

while True:

    #Get Sensor Data from DHT11

    temp=random.randint(90,110)
    Humid=random.randint(60,100)

    Mois=random.randint(20,120)

    data = { 'temp' : temp, 'Humid': Humid, 'Mois' :Mois}
    #print      data      def
    myOnPublishCallback():
    print      ("Published
    Temperature = %s C" %
    temp, "Humidity = %s %"
    % Humid, "Moisture =%s
    deg c" %Mois, "to IBM
    Watson")

    success  =  deviceCli.publishEvent("IoTSensor",  "json",  data,  qos=0,
on_publish=myOnPublishCallback) if
    not success:

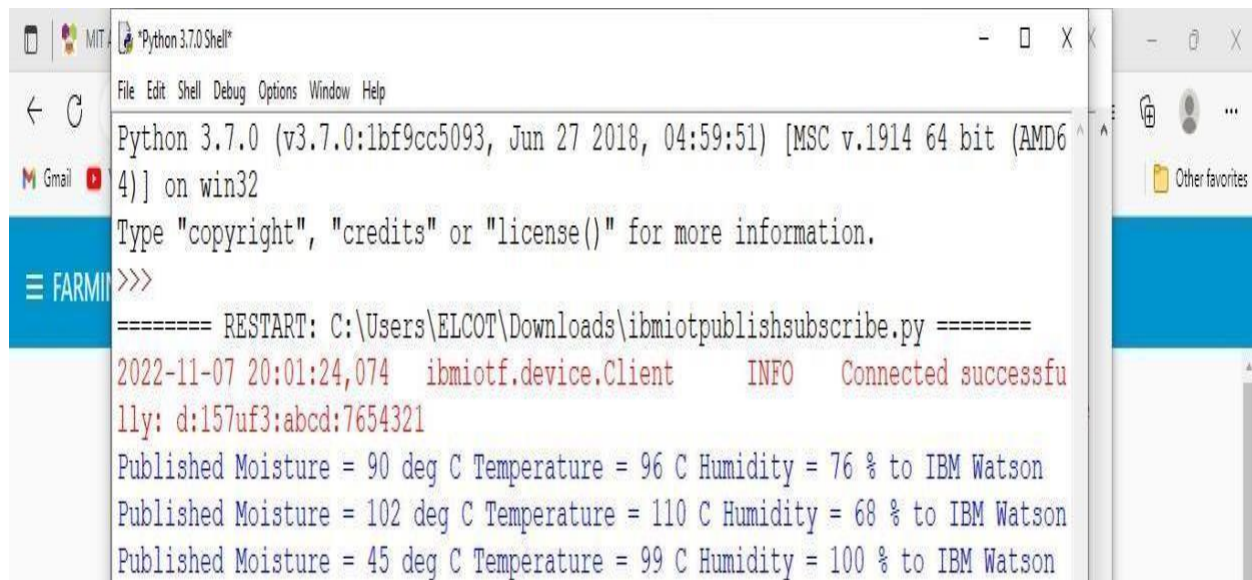
        print("Not connected to IoT")

```

```
time.sleep(10)
```

```
deviceCli.commandCallback = myCommandCallback
```

```
# Disconnect the device and application from the cloud deviceCli.disconnect()
```



```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py =====
2022-11-07 20:01:24,074 ibmiotf.device.Client INFO Connected successfully: d:157uf3:abcd:7654321
Published Moisture = 90 deg C Temperature = 96 C Humidity = 76 % to IBM Watson
Published Moisture = 102 deg C Temperature = 110 C Humidity = 68 % to IBM Watson
Published Moisture = 45 deg C Temperature = 99 C Humidity = 100 % to IBM Watson
```

### 4.3 IoT Simulator

In our project in the place of sensors we are going to use IoT sensor simulator which give random readings to the connected cloud.

The link to simulator:

<https://watson-iot-sensor-simulator.mybluemix.net/>

We need to give the credentials of the created device in IBM Watson IoT Platform to connect cloud to simulator.

### 4.4 OpenWeather API

OpenWeatherMap is an online service that provides weather data. It provides current weather data, forecasts and historical data to more than 2 million customer.

Website link: <https://openweathermap.org/guide> **Steps**

**to configure:**

- o Create account in OpenWeather
- o Find the name of your city by searching
- o Create API key to your account
- o Replace “city name” and “your api key” with your city and API key in below red text

[api.openweathermap.org/data/2.5/weather?q={city name}&appid={your api key}](https://api.openweathermap.org/data/2.5/weather?q={city name}&appid={your api key})