

# DEVELOP A WEB-BASED APPLICATION

## USING NODE-RED

### SOFTWARE REQUIREMENTS

The web-based application is deployed by consolidating the essential requirements to build IOT web-based platform. The following credential are needed to deploy the web application using NODE-RED which are stated as below

#### 1. Node-Red

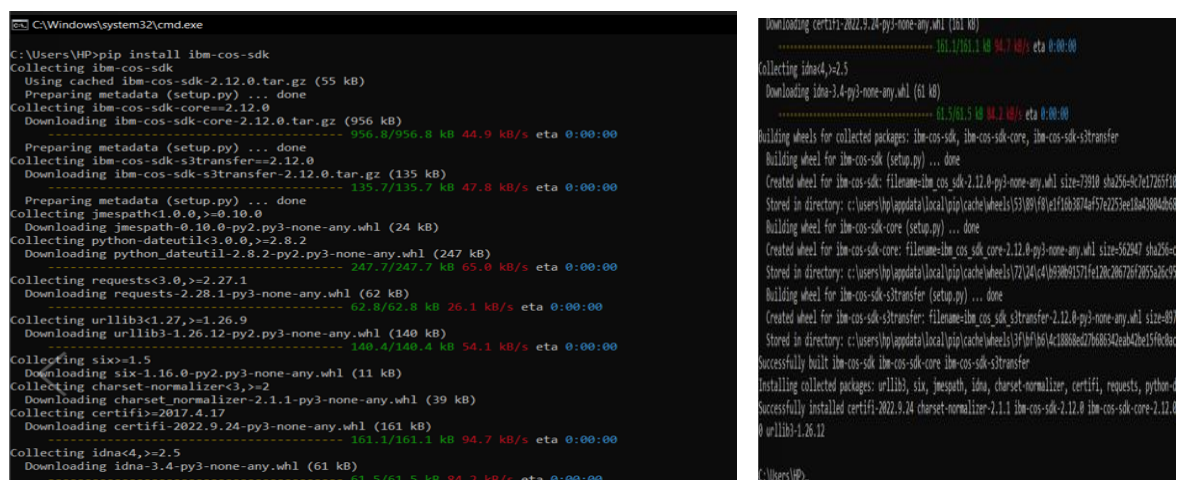
Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs, and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.

#### 2. 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.

#### 3. Python Editor

Various packages are installed to work in IBM platform and for deployment of IOT device



```
C:\Windows\system32\cmd.exe
C:\Users\HP>pip install ibm-cos-sdk
Collecting ibm-cos-sdk
  Using cached ibm-cos-sdk-2.12.0.tar.gz (55 kB)
  Preparing metadata (setup.py) ... done
Collecting ibm-cos-sdk-core==2.12.0
  Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)
    ----- 956.0/956.0 kB 44.9 kB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting ibm-cos-sdk-s3transfer==2.12.0
  Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB)
    ----- 135.7/135.7 kB 47.8 kB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting jmespath<1.0.0,>=0.10.0
  Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)
Collecting python-dateutil<3.0.0,>=2.8.2
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
    ----- 247.7/247.7 kB 65.0 kB/s eta 0:00:00
Collecting requests<3.0,>=2.27.1
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
    ----- 62.8/62.8 kB 26.1 kB/s eta 0:00:00
Collecting urllib3<1.27,>=1.26.9
  Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)
    ----- 140.4/140.4 kB 54.1 kB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting charset-normalizer<3,>=2
  Downloading charset-normalizer-2.1.1-py3-none-any.whl (39 kB)
Collecting certifi>=2017.4.17
  Downloading certifi-2022.9.24-py3-none-any.whl (161 kB)
    ----- 161.1/161.1 kB 94.7 kB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
    ----- 61.5/61.5 kB 84.2 kB/s eta 0:00:00

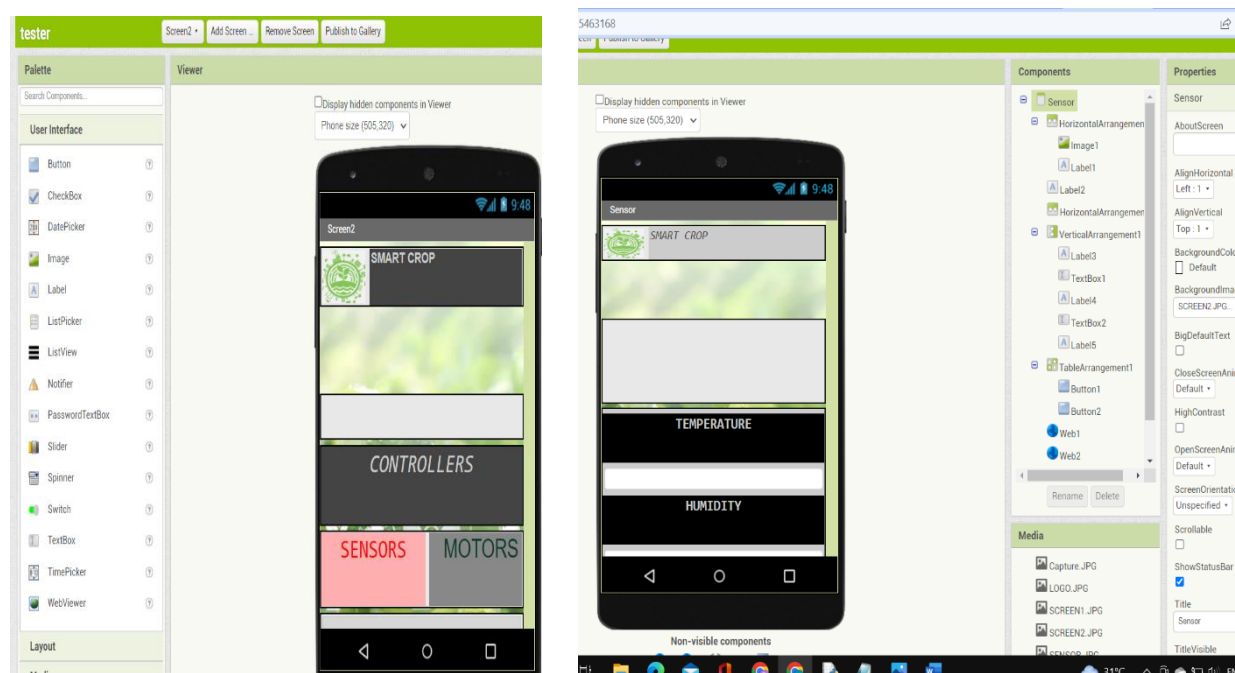
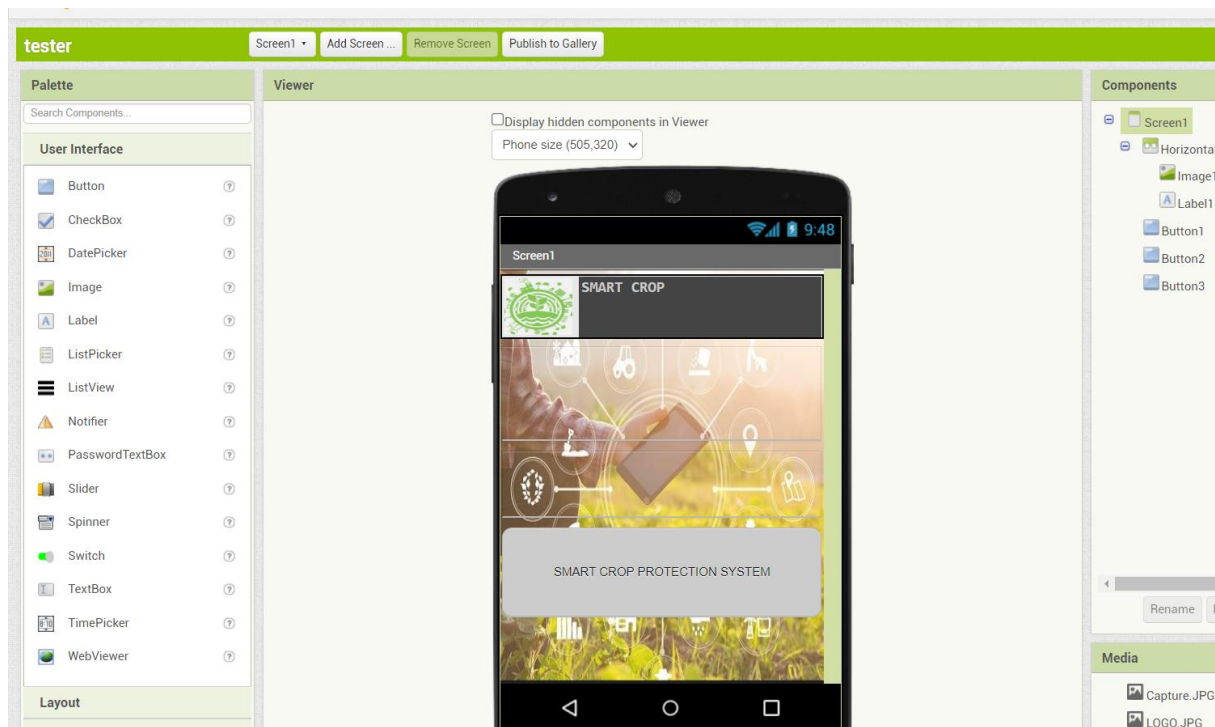
download: certifi-2022.9.24-py3-none-any.whl (161 kB)
----- 161.1/161.1 kB 94.7 kB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
    ----- 61.5/61.5 kB 84.2 kB/s eta 0:00:00
Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer
Building wheel for ibm-cos-sdk (setup.py) ... done
Created wheel for ibm-cos-sdk: filename=ibm_cos_sdk-2.12.0-py3-none-any.whl size=73910 sha256=8c1e1765f10
Stored in directory: c:\users\hp\appdata\local\pip\cache\wheels\53\09\49\ef16b3874a57e225ee1ba43884666
Building wheel for ibm-cos-sdk-core (setup.py) ... done
Created wheel for ibm-cos-sdk-core: filename=ibm_cos_sdk_core-2.12.0-py3-none-any.whl size=562047 sha256=
Stored in directory: c:\users\hp\appdata\local\pip\cache\wheels\72\34\c4\63b0b0157f4120c206776f785a26c9f
Building wheel for ibm-cos-sdk-s3transfer (setup.py) ... done
Created wheel for ibm-cos-sdk-s3transfer: filename=ibm_cos_sdk_s3transfer-2.12.0-py3-none-any.whl size=83
Stored in directory: c:\users\hp\appdata\local\pip\cache\wheels\3f\bf\63\4c188b6e2766874eab32be15f0eb
Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-
Successfully installed certifi-2022.9.24 charset-normalizer-2.1.1 ibm-cos-sdk-2.12.0 ibm-cos-sdk-core-2.12.0
urllib3-1.26.12
C:\Users\HP>
```

#### 4. MIT-APP INVENTOR

Simulation is carried on by using MIT-App inventor for Node-Red web application we have created a simulator with three modules they are:

- HOME SCREEN
- CONTROLLER SCREEN
- DATA SCREEN

There will be a designer and block panel separately and features and styles are being done in designer panel where we use to drag and drop desired icons. Similarly block panel here we give functionality to the features which are done by drag and drop method





## 5. Open Weather API

It is an online service that provides weather data. It provides current weather data, forecasts, and historical data to more than 2 million customers. Website link: <https://openweathermap.org/guide>

To configure: Create account in Open Weather and find the name of your city by searching then create API key to your account after that replace “city name” and “your API key” with your city and API key

The screenshot shows the OpenWeather API website. The top navigation bar includes links for Guide, API, Dashboard, Marketplace, Pricing, Maps, Our Initiatives, Partners, and Blog. Below this, there's a section for API keys with a table listing existing keys. A 'Create key' button is visible. Below the table, a sample JSON response is shown for a weather query for Coimbatore.

Key	Name	Status	Actions
ef901ce25a1088271fc257f6b02c27a3	Default	Active	 

Create key

API key name

```
{
  "coord": {
    "lon": 76.9667,
    "lat": 11
  },
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "few clouds",
      "icon": "02n"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 302.03,
    "feels_like": 301.82,
    "temp_min": 302.03,
    "temp_max": 302.03,
    "pressure": 1013,
    "humidity": 42,
    "visibility": 6000,
    "wind": {
      "speed": 1.54,
      "deg": 0
    },
    "clouds": {
      "all": 20
    },
    "dt": 1667824988,
    "sys": {
      "type": 1,
      "id": 9206,
      "country": "IN",
      "sunrise": 1667781913,
      "sunset": 1667823983,
      "timezone": 19800,
      "id": 1273865,
      "name": "Coimbatore",
      "cod": 200
    }
  }
}
```

## 6. IBM Cloudant

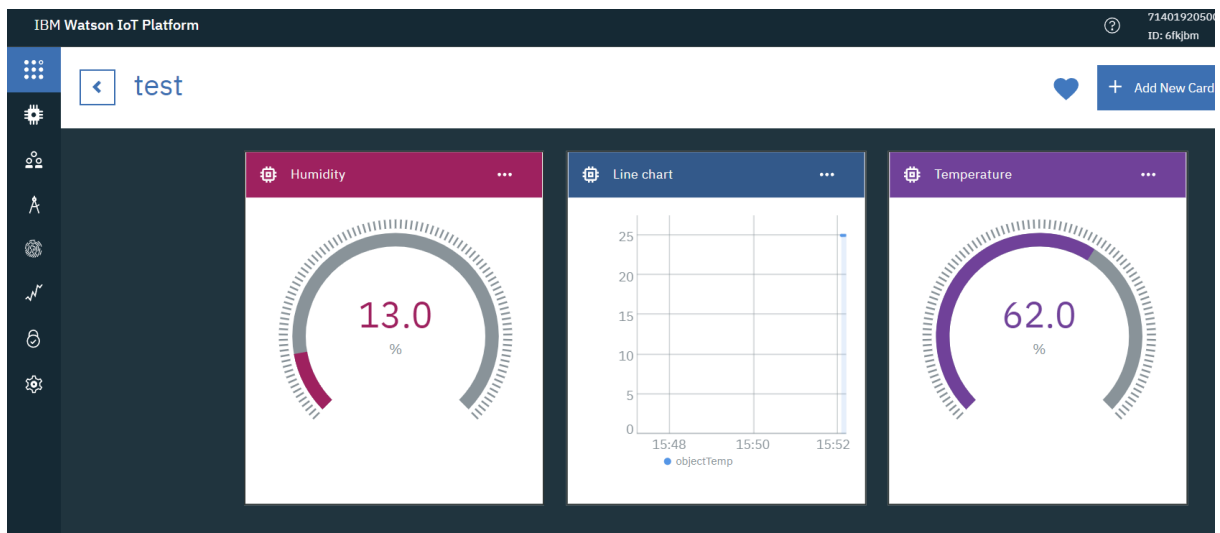
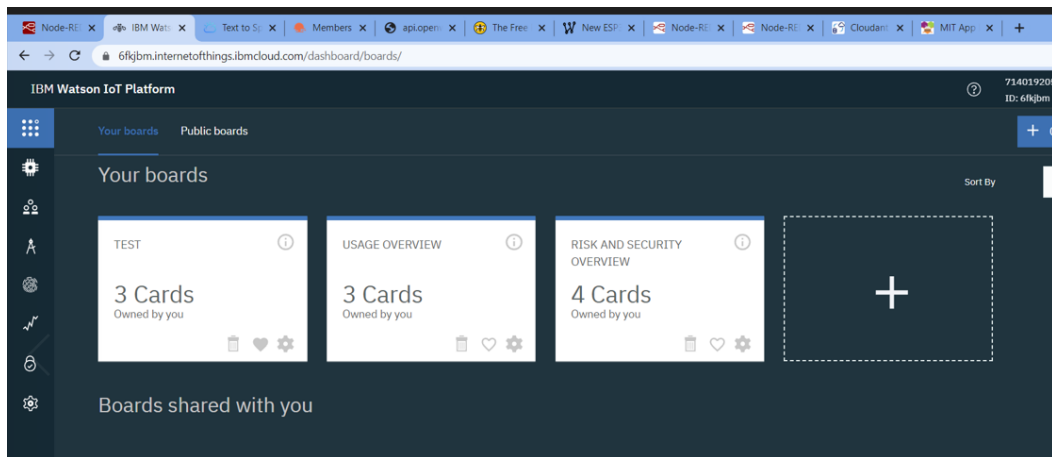
In order to store the NOD-RED value or any service provider we need to have a Database here the cloudant database is created inorder to store the values of the sensor reading of the devices

The screenshot shows the IBM Cloudant dashboard. The left sidebar contains navigation options like All Documents, Query, Permissions, Changes, and Design Documents. The main area displays a table of documents with columns for id, key, and value. The table shows a list of documents with their respective keys and values.

id	key	value
02c9db35bd912af5aeb4c9108645d540	02c9db35bd912af5aeb4c9108645d540	{ "rev": "1-6c0d8a0ab163ca3e5ea48e9480...
03fc1a9006d3f6c8392d971209e6511	03fc1a9006d3f6c8392d971209e6511	{ "rev": "1-b4ae960207938960e733cf1ad255...
07335b582c951f2e2c2faaee31056d31	07335b582c951f2e2c2faaee31056d31	{ "rev": "1-e2ad7802fec1e79086d607041c0...
07335b582c951f2e2c2faaee3109e67a	07335b582c951f2e2c2faaee3109e67a	{ "rev": "1-f7b0d07e948f8e721a37ea21035e...
07335b582c951f2e2c2faaee313745d1	07335b582c951f2e2c2faaee313745d1	{ "rev": "1-881a8696ecf3ee8f5a28de977f7e...
07335b582c951f2e2c2faaee314bbabd	07335b582c951f2e2c2faaee314bbabd	{ "rev": "1-e5c2b8951f863f5f3cd864657a7...
07335b582c951f2e2c2faaee3171e386	07335b582c951f2e2c2faaee3171e386	{ "rev": "1-b064de1870f95a5d59de0fb1ca...
07335b582c951f2e2c2faaee3173b632	07335b582c951f2e2c2faaee3173b632	{ "rev": "1-20a2b6b133a7ed92e66c845d35b...
07335b582c951f2e2c2faaee31768ac0	07335b582c951f2e2c2faaee31768ac0	{ "rev": "1-29bfa44a24c7f4e45d6f17189c7b2...
07335b582c951f2e2c2faaee31769426	07335b582c951f2e2c2faaee31769426	{ "rev": "1-7e5162ad616c53e3854b69c5474f...
07335b582c951f2e2c2faaee317a46de	07335b582c951f2e2c2faaee317a46de	{ "rev": "1-354ea9a1aa0b32b1e5bd3362eff...
07335b582c951f2e2c2faaee317b2734	07335b582c951f2e2c2faaee317b2734	{ "rev": "1-1a36bf3c8add94ecc212bbc872be...
07335b582c951f2e2c2faaee317b7691	07335b582c951f2e2c2faaee317b7691	{ "rev": "1-e41e68f23f2f9969d390cc9aad77f...

Showing document 1 - 20. Documents per page: 20

## OUTPUT OF WEB-BASED APPLICATION USING NODE-RED

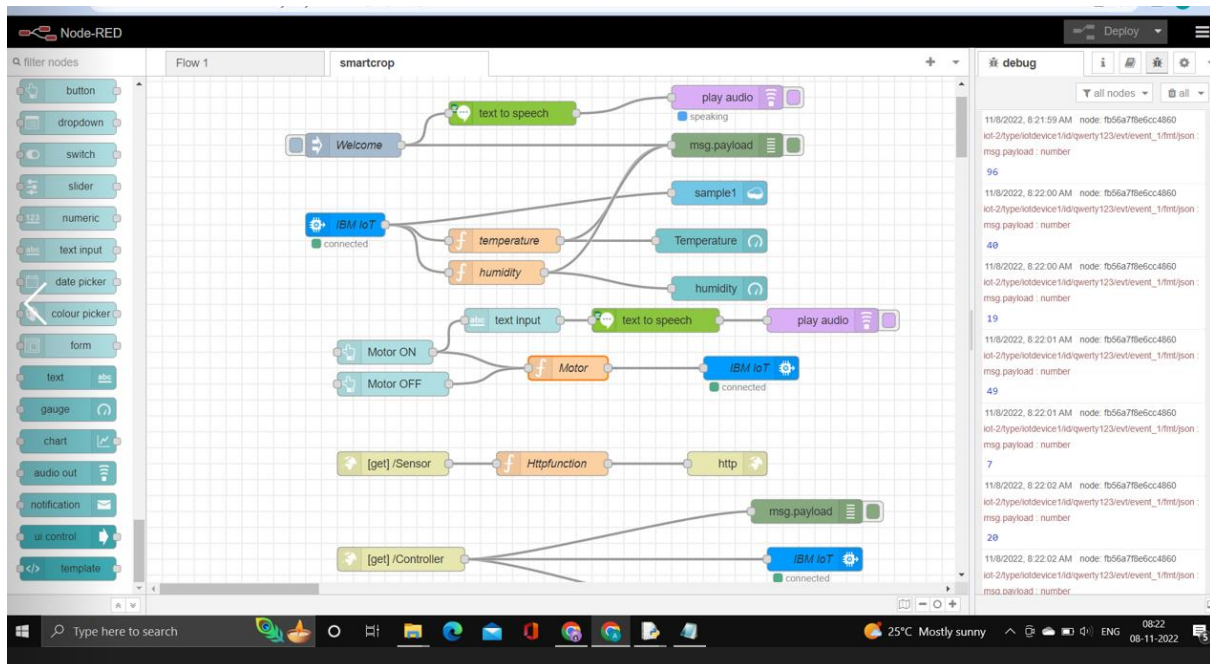


This screenshot shows the 'Browse Devices' page in the IBM Watson IoT Platform. It includes a search bar, a table of devices, and a detailed view for a specific device.

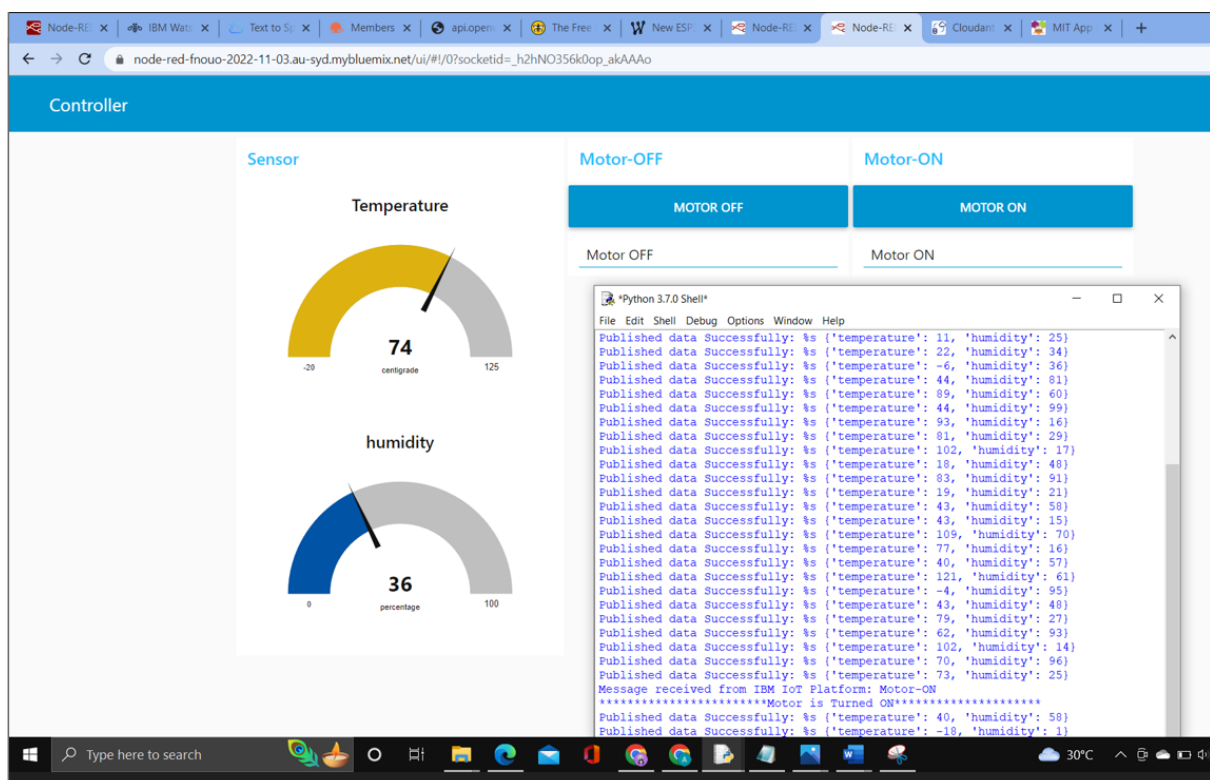
Device ID	Status	Device Type	Class ID	Date Added
qwerty123	Connected	iotdevice1	Device	Nov 7, 2022 4:39 PM

Identity	Device Information	Recent Events	State	Logs
Device ID	qwerty123			
Device Type	iotdevice1			
Date Added	Nov 7, 2022 4:39 PM			
Added By	714019205002@smartinternz.com			



## NODE-RED FLOW EDITOR WITH TEXT TO SPEECH



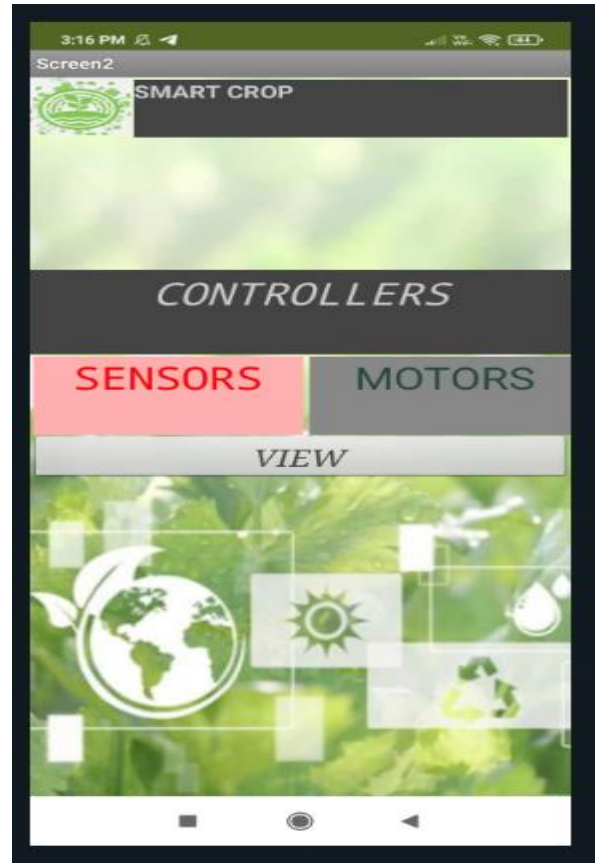
## NODE-RED WEB GUI



## SIMULATOR DEPLOYMENT OUTPUT



HOME SCREEN

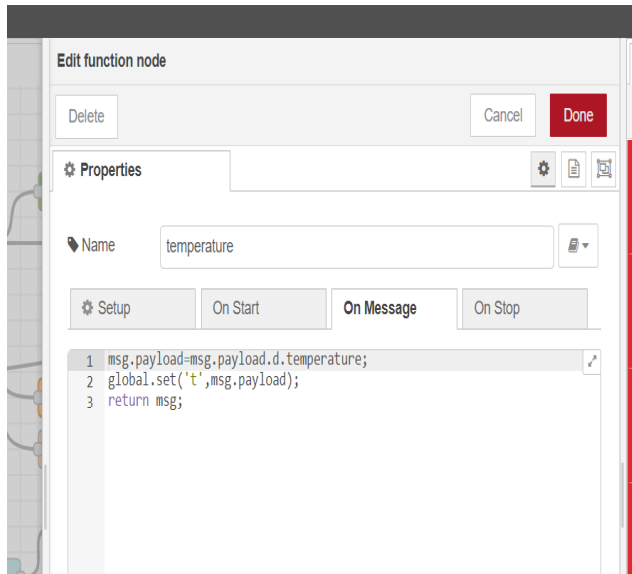


CONTROLLER SCREEN

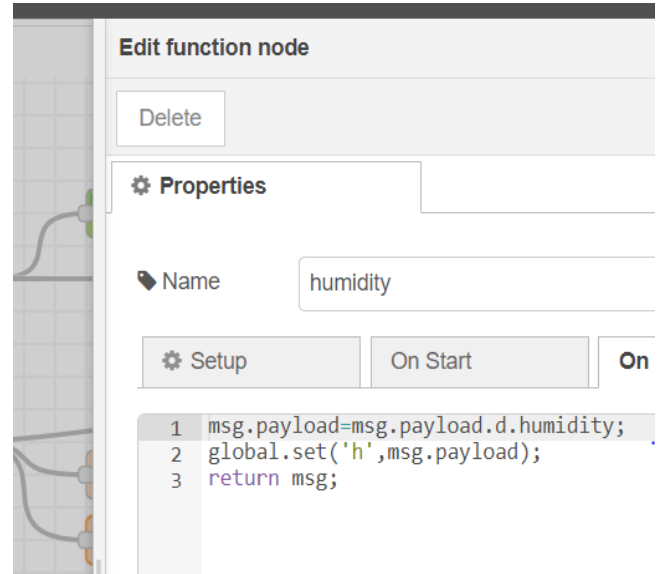


## CODE FOR NODE-RED DEPLOYMENT

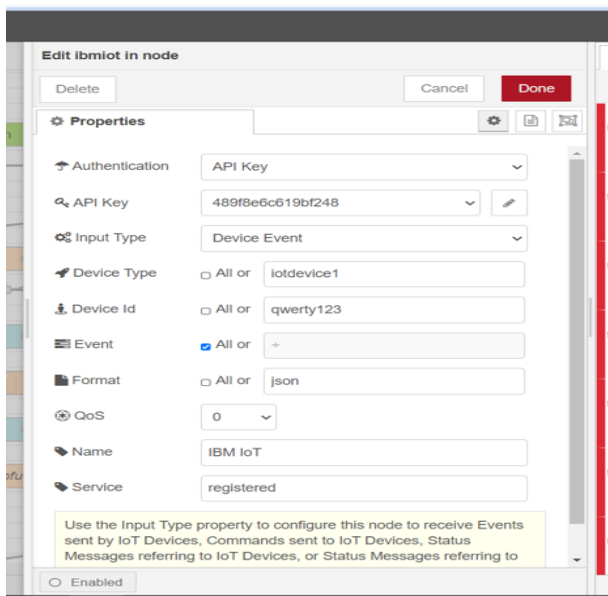
### TEMPERATURE READING



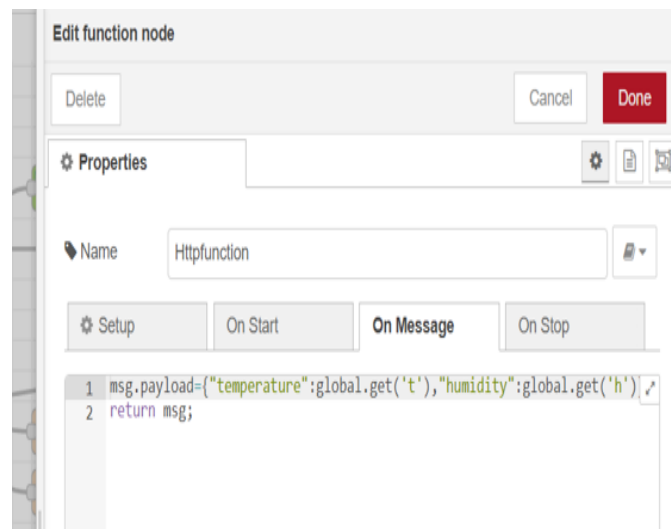
### HUMIDITY READING



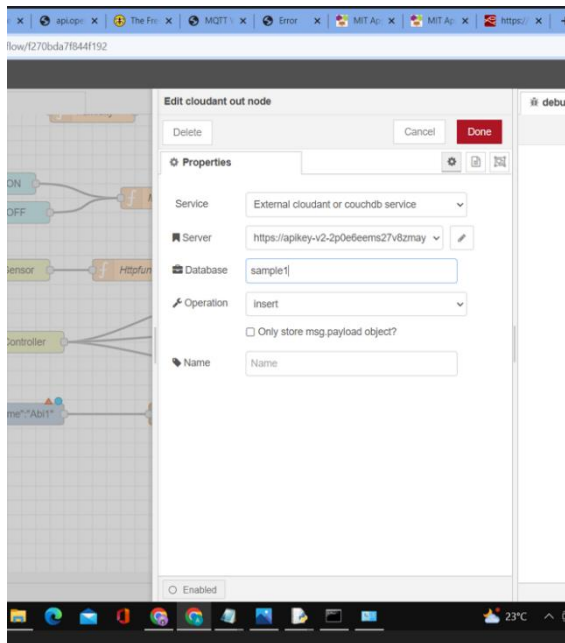
### NODE-RED TO IOT WATSON



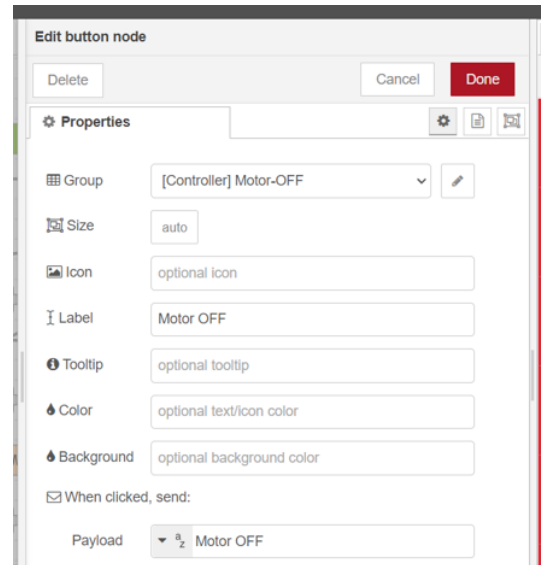
### HTTP CONNECTION TO MIT-APP



## NODE-RED TO CLOUDANT CONNECTIVITY



## MOTOR ON AND OFF



## CODE FOR PYTHON EDITOR

#IBM Watson IOT Platform

```
import wiotp.sdk.device
```

```
import time
```

```
import random
```

```
myConfig = {
```

```
    "identity": {
```

```
        "orgId": "6fkjbm",
```

```
        "typeId": "iotdevice1",
```

```
        "deviceId": "qwerty123"
```

```
    },
```

```
    "auth": {
```

```
        "token": "johnyjohnnyespapa"
```

```
    }
```

```
}
```

```
def myCommandCallback(cmd):
```



```
print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
m=cmd.data['command']
if(m=="Motor-ON"):
    print("*****Motor is Turned ON*****")
else:
    print("*****Motor is Turned OFF*****")
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

while True:
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'temperature':temp, 'humidity':hum}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
onPublish=None)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
```

## IBM TEXT TO SPEECH

```
from ibm_watson import TextToSpeechV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
authenticator =
IAMAuthenticator('M_u6yEvEGJylj_ysbL_pG0ZOKuRCQW1LgXUtv_IcBPCR')
text_to_speech = TextToSpeechV1(
    authenticator=authenticator
)
text_to_speech.set_service_url('https://api.au-syd.text-to-
speech.watson.cloud.ibm.com/instances/23724eb6-a096-4a3a-b914-da0e442c1c5f')
with open('hello_world.wav', 'wb') as audio_file:
```

```
audio_file.write(  
    text_to_speech.synthesize(  
        'Alert',  
        voice='en-US_AllisonV3Voice',  
        accept='audio/wav'  
    ).get_result().content)
```

## CODE FOR MIT-APP INVENTOR (SCREEN 3)

