IOT ENABLED SMART FARMING APPLICATION

Sprint Delivery – 1

Date	17.11.2022
Team ID	PNT2022TMID06965
Project Name	SMART FARMER - IOT ENABLED SMART FARMINGAPPLICATION SYSTEM

Introduction

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the fieldlike Temperature, soil moisture, humidity and etc and control the equipment like water motor and other devices remotely via internet without their actual presencein the field.

Problem Statement

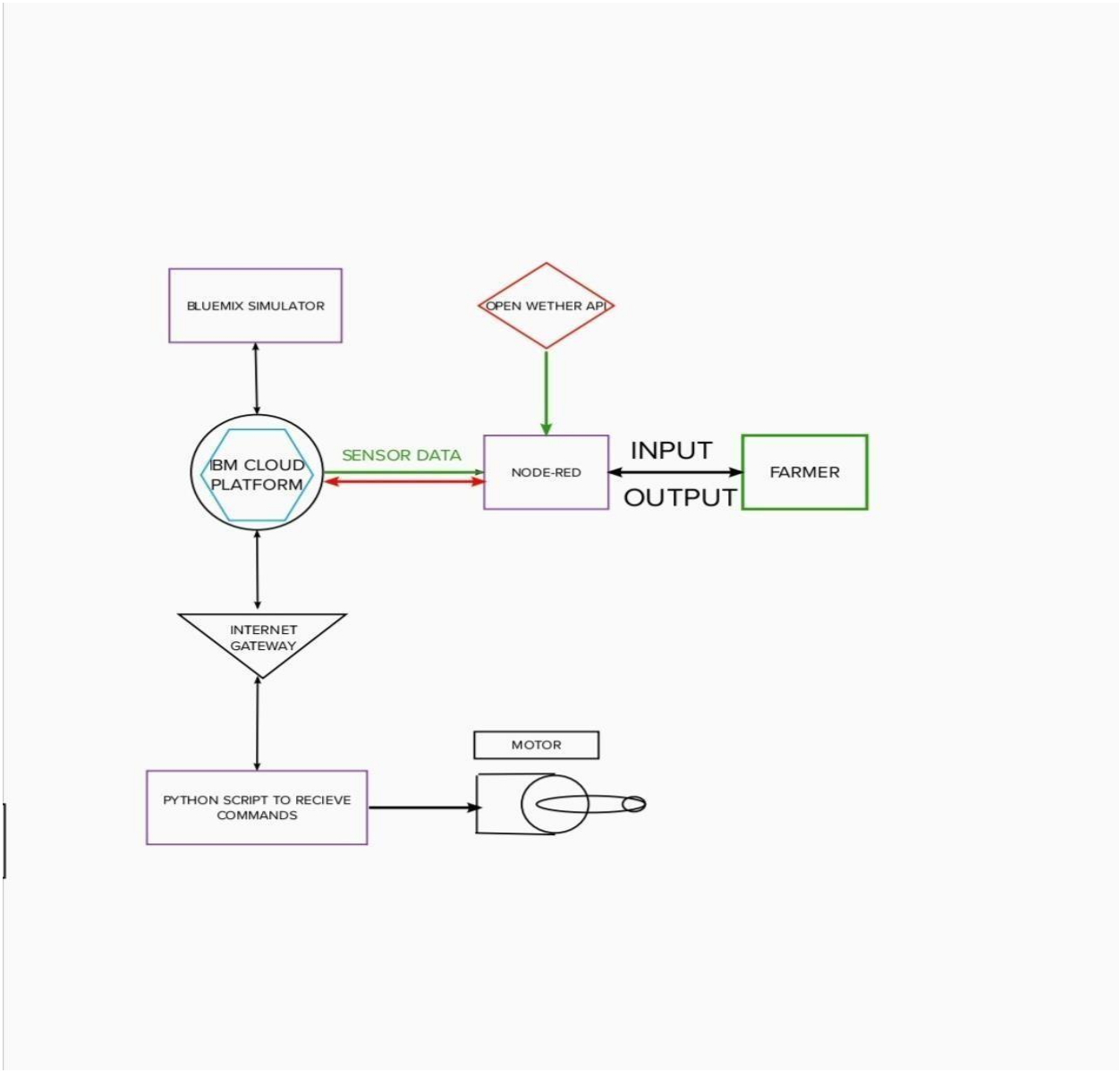
Farmers are to be present at farm for its maintenance irrespective of the weather conditions. They have to ensure that the crops are well watered and the farm status is monitored by them physically. Farmer have to stay most of the time in field in order to get a good yield. In difficult times like in the presence of pandemic also they have to work hard in their fields risking their lives to provide food for the country.

Proposed Solution

In order to improve the farmer's working conditions and make them easier, we introduce IoT services to him in which we use cloud services and internet to enable farmer to continue his work remotely via internet. He can monitor the field parameters and control the devices in farm.

Block Diagram

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

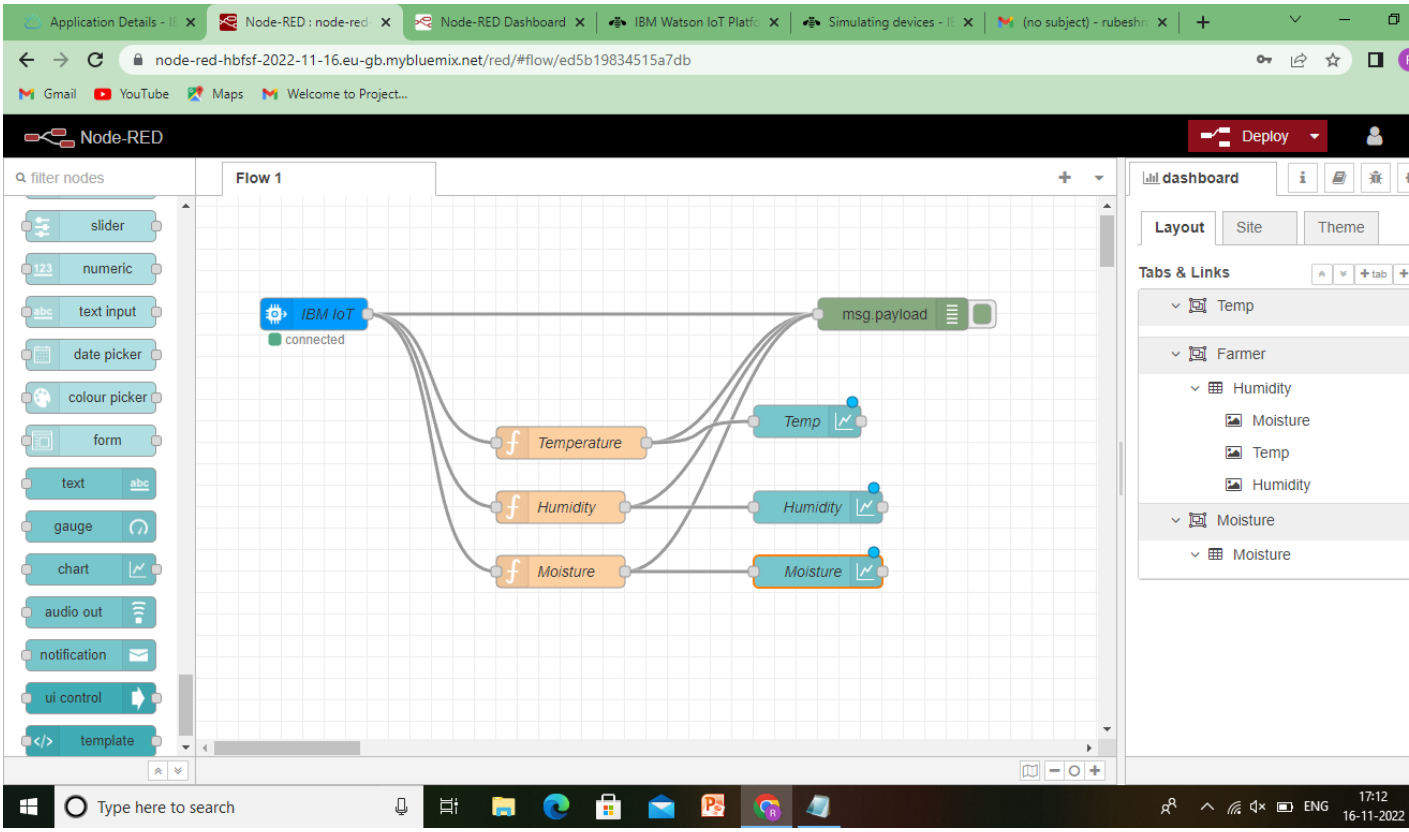


Required Software Installation

Node-Red

Node-RED is a flow-based development tool for visual programming developedoriginally 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

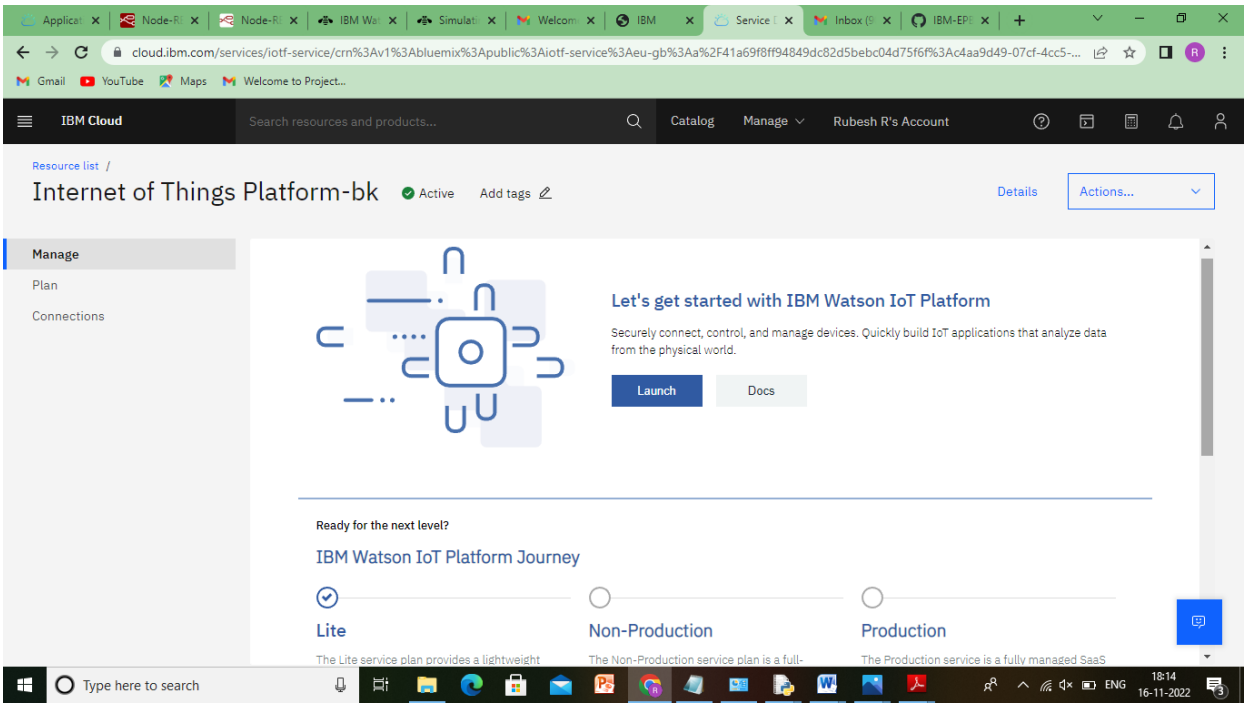
Type=>node-red
Then open http://localhost:1880/ in browser

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 node2. Dashboard node

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 Platformis a managed, cloud-hosted service designed to make it simple to derive value fromyour IoT devices.

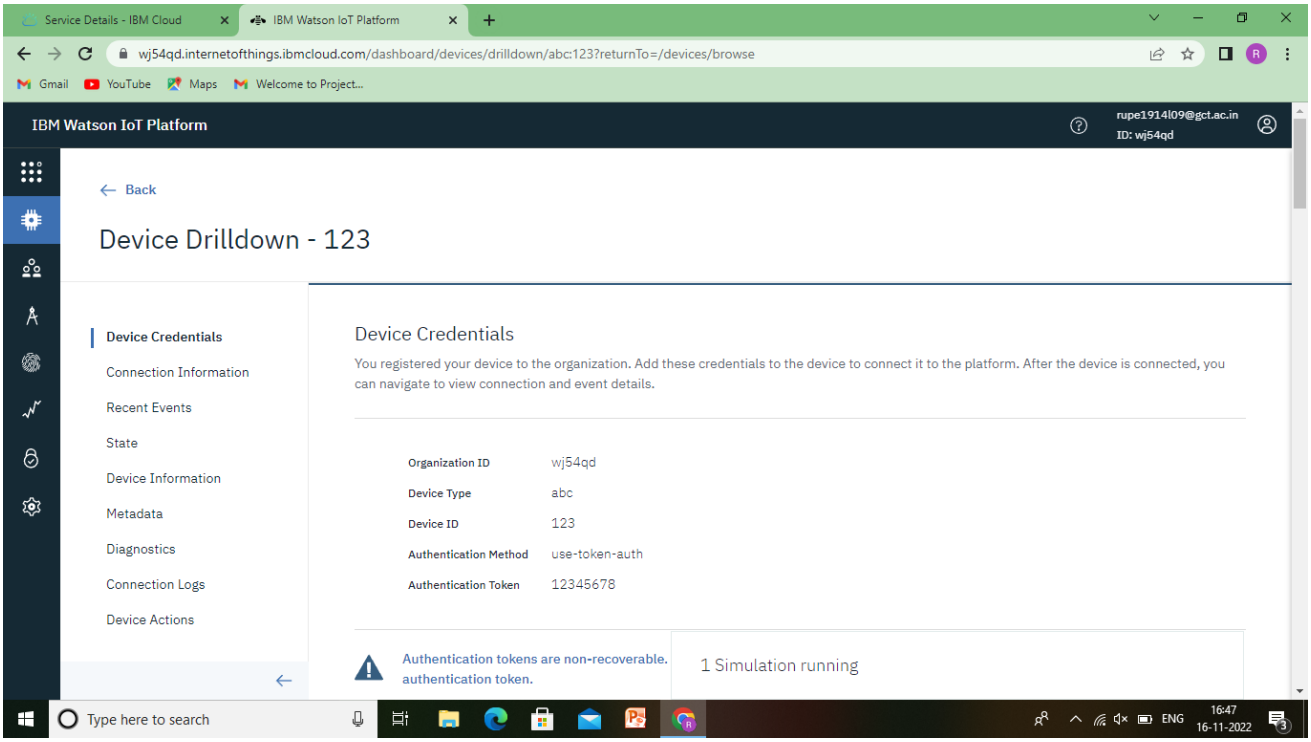


Steps to configure:

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.



Python IDE

Install Python3 compiler

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

CODE

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device
organization = "wj54qd"
deviceType="abc"
deviceId = "123"
authMethod = "token"
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()
deviceCli.connect()
while True:
   temp=random.randint(90,110)
   Humid=random.randint(60,100)
   Mois=random.randint(20,120)
   data = { 'temp' : temp, 'hum': Humid, 'mois' :Mois}
   #print data
   def myOnPublishCallback():
       print ("Published data",data, "to IBM Watson")


   success = deviceCli.publishEvent("event", "json", data,0,
myOnPublishCallback)
   if not success:
      print("Not connected to IoTF")
   time.sleep(5)
   deviceCli.commandCallback = myCommandCallback
#Disconnect the device and application from the cloud
deviceCli.disconnect()
```

Output