**Assignment -3 18481A0478**

Develop a code to upload the water tank level and light intensity values to the IBM IoT platform and visualize them in the web application.

Python code:

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import json

#Provide your IBM Watson Device Credentials

organization = "gaxcrt"

deviceType = "iotdevice"

deviceId = "1001"

authMethod = "token"

authToken = "Chinna@9966"

# Initialize the device client.

w=0

L=0

def myCommandCallback(cmd):

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

if cmd.data['command']=='lighton':

print("LIGHT ON IS RECEIVED")

elif cmd.data['command']=='lightoff':

print("LIGHT OFF IS RECEIVED")

if cmd.command == "setInterval":

if 'interval' not in cmd.data:

print("Error - command is missing required information: 'interval'")

else:

interval = cmd.data['interval']

elif cmd.command == "print":

if 'message' not in cmd.data:

print("Error - command is missing required information: 'message'")

else:

print(cmd.data['message'])

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:

w=89

L=44

#Send waterlevel & light intensity to IBM Watson

data = {"d":{ 'Waterlevel' : w, 'lightintensity': L, }}

#print data

def myOnPublishCallback():

print ("Published waterlevel = %s units" % w, "Light intensity = %s %%" % L, "to IBM Watson")

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

if not success:

print("Not connected to IoTF")0

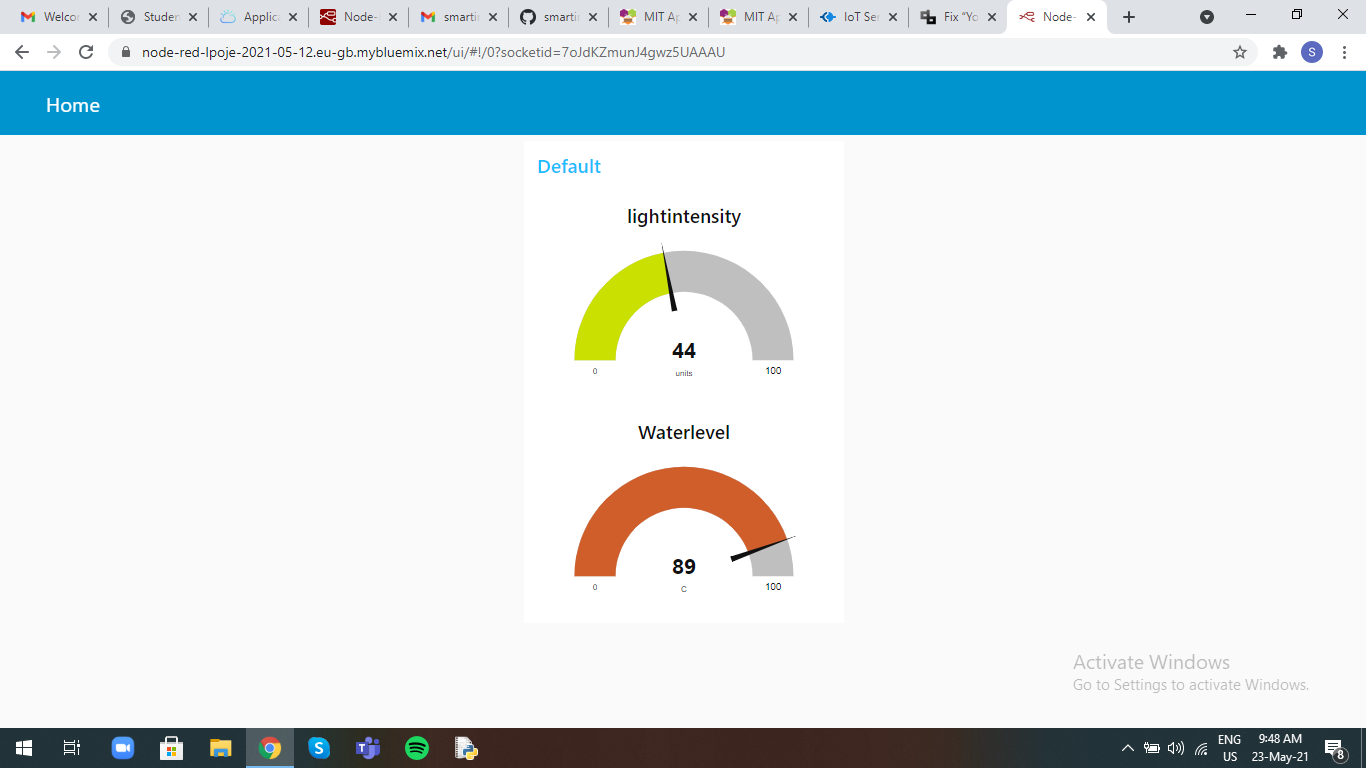
time.sleep(1)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()

**UI:**

****

**Node Red Blocks:**

