**ASSIGNMENT-3**

***3.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*.**

Platform Required: Node Red Services to establish the link between IBM Watson cloud and user input code.

Python IDE(Latest version) for execution of given code.

MIT APP INVENTOR for building of mobile application.

Python code for Water-Level and Light-Intensity

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import json

#Provide your IBM Watson Device Credentials

organization ="kbfeya"

deviceType = "IOTDEVICE"

deviceId = "1010"

authMethod = "token"

authToken = "07\_13\*11&83"

# Initialize the device client.

WL=0

LI=0

def myCommandCallback(cmd):

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

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

print("SWITCH ON IS RECEIVED")

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

print("SWITCH 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:

WL=29.58

LI=35.46

#Send Water-Level & Light Intensity to IBM Watson

data =jsondata={"d":{ 'waterlevel': WL, 'lightintensity': LI}}

print (data)

def myOnPublishCallback():

print ("Published Water Level = %s %%" % WL, "Light Instensity = %s %%" % LI, "to IBM Watson")

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

if not success:

print("Not connected to IoTF")

time.sleep(1)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()

Python Program Output:

Published Water Level = 25.56 % Light Instensity = 34.78 % to IBM Watson

{'d': {'waterlevel': 25.56, 'lightintensity': 34.78}}

Published Water Level = 25.56 % Light Instensity = 34.78 % to IBM Watson

{'d': {'waterlevel': 25.56, 'lightintensity': 34.78}}

Published Water Level = 25.56 % Light Instensity = 34.78 % to IBM Watson

{'d': {'waterlevel': 25.56, 'lightintensity': 34.78}}

Published Water Level = 25.56 % Light Instensity = 34.78 % to IBM Watson

Command received: switchon

SWITCH ON IS RECEIVED

{'d': {'waterlevel': 25.56, 'lightintensity': 34.78}}

Published Water Level = 25.56 % Light Instensity = 34.78 % to IBM Watson

{'d': {'waterlevel': 25.56, 'lightintensity': 34.78}}

Published Water Level = 25.56 % Light Instensity = 34.78 % to IBM Watson

Command received: switchon

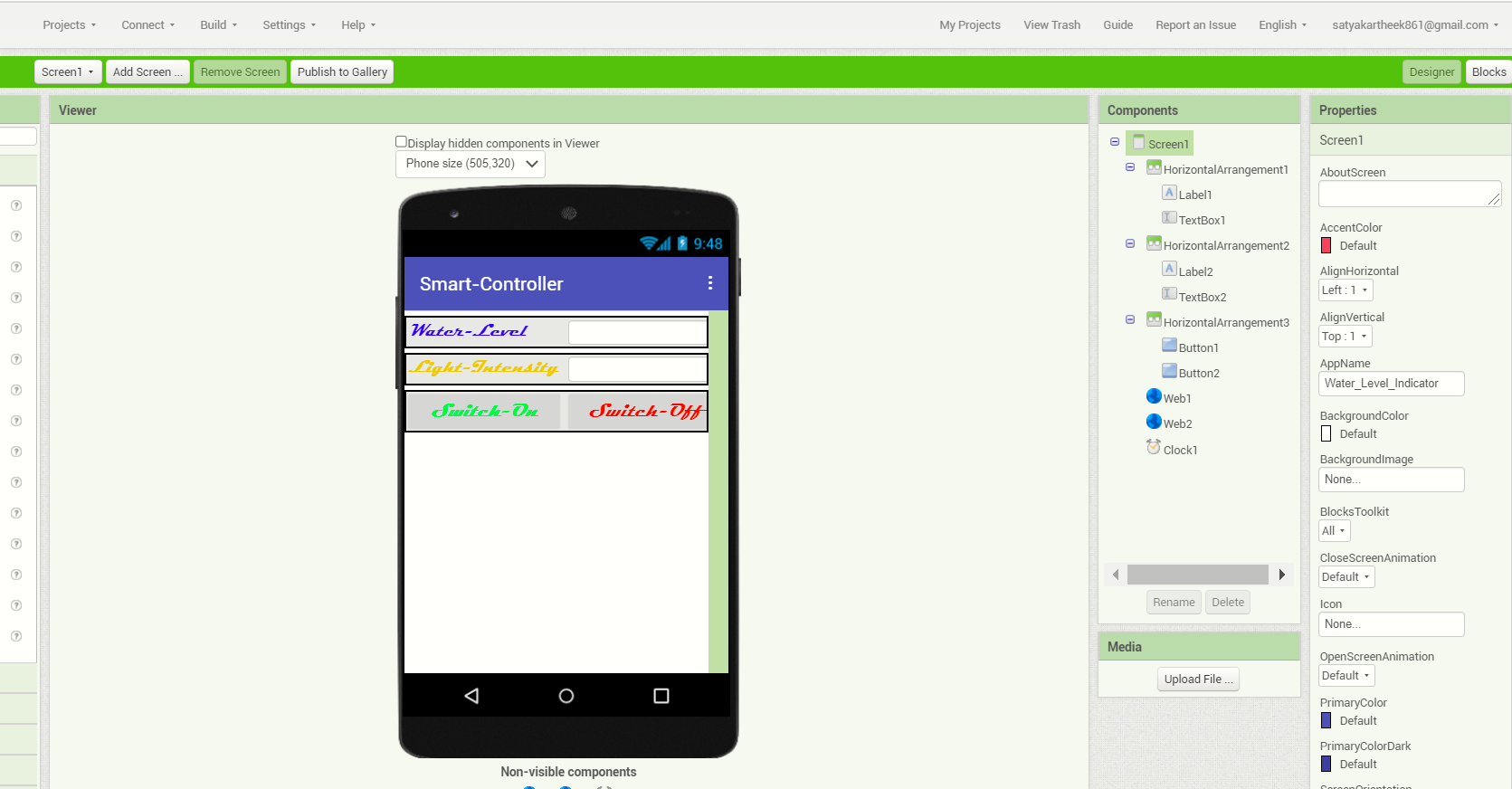
SWITCH ON IS RECEIVED

User Interface Image : 

Node-Red Connections:



MIT DESIGN:



MIT BLOCKS



MOBILE APP OUTPUT:



