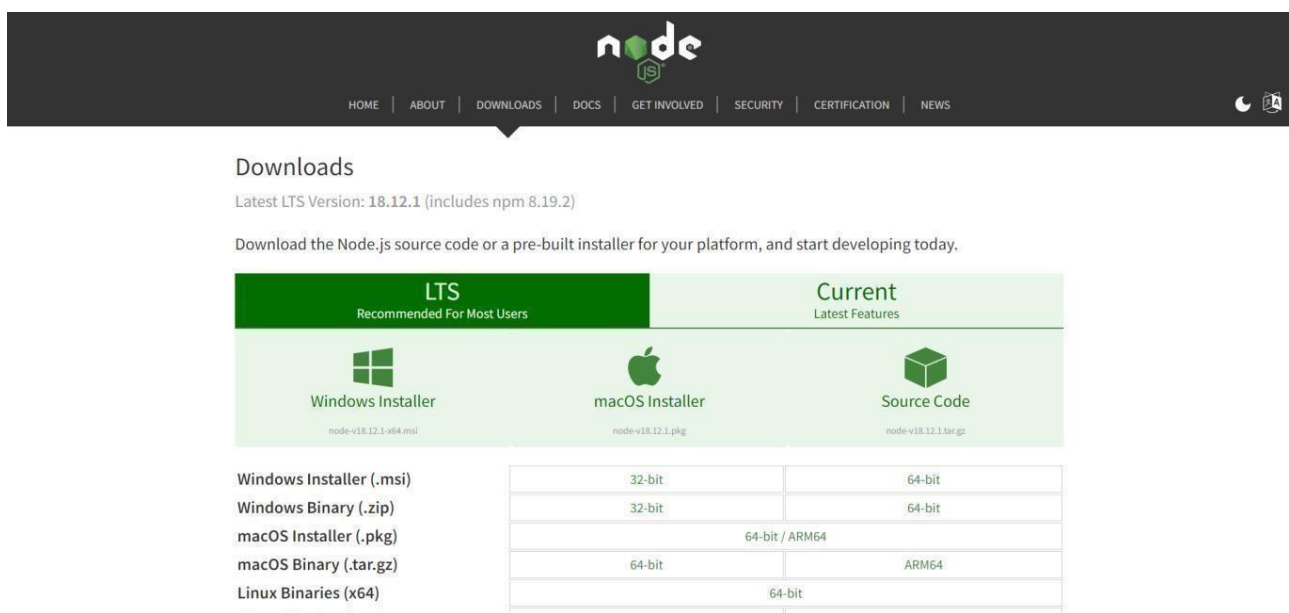


SPRINT 2

TEAM ID	PNT2022TMID04613
Project Name	IoT based smart crop protection system for Agriculture

STEP1: Download and Install NODE JS.



Downloads

Latest LTS Version: 18.12.1 (includes npm 8.19.2)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS
Recommended For Most Users

Current
Latest Features

Windows Installer
node-v18.12.1-x64.msi

macOS Installer
node-v18.12.1.pkg

Source Code
node-v18.12.1.tar.gz

Windows Installer (.msi)

Windows Binary (.zip)

macOS Installer (.pkg)

macOS Binary (.tar.gz)

Linux Binaries (x64)

32-bit	64-bit
32-bit	64-bit
64-bit / ARM64	
64-bit	ARM64
64-bit	

STEP2: Setup node.js and configure command prompt for error check. open node-red from the generated link.

```
node-red
4 Nov 18:48:05 - [info] Node-RED version: v3.0.2
4 Nov 18:48:05 - [info] Node.js version: v18.12.0
4 Nov 18:48:05 - [info] Windows_NT 10.0.19044 x64 LE
4 Nov 18:48:26 - [info] Loading palette nodes
4 Nov 18:48:44 - [info] Settings file : C:\Users\ELCOT\.node-red\settings.js
4 Nov 18:48:45 - [info] Context store : 'default' [module=memory]
4 Nov 18:48:45 - [info] User directory : \Users\ELCOT\.node-red
4 Nov 18:48:45 - [warn] Projects disabled : editorTheme.projects.enabled=false
4 Nov 18:48:45 - [info] Flows file : \Users\ELCOT\.node-red\flows.json
4 Nov 18:48:45 - [info] Creating new flow file
4 Nov 18:48:45 - [warn]

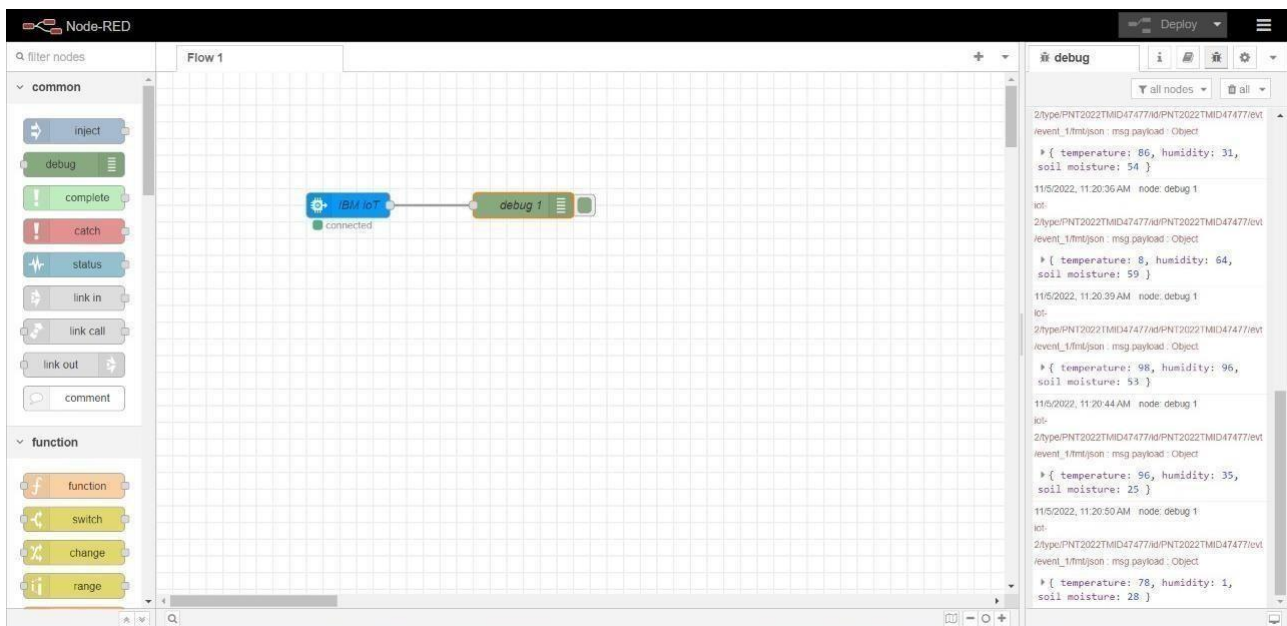
-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

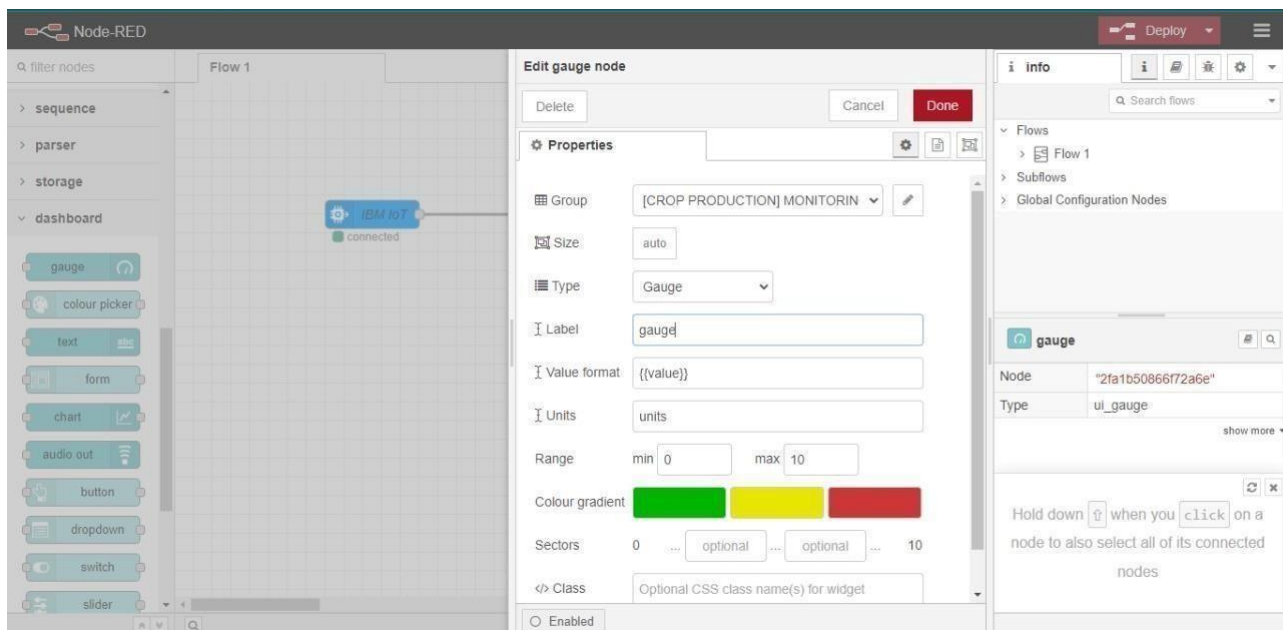
You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----

4 Nov 18:48:45 - [warn] Encrypted credentials not found
4 Nov 18:48:45 - [info] Starting flows
4 Nov 18:48:46 - [info] Started flows
4 Nov 18:48:46 - [info] Server now running at http://127.0.0.1:1880/
```

STEP3: Connect IBM IOT in and Debug 1 and Deploy.



STEP4: Edit gauge node (Here the gauge nodes are named as Temperature, Humidity and Soil moisture).



Node-RED

filter nodes

dashboard

button

dropdown

switch

slider

numeric

text input

data picker

colour picker

form

text

gauge

chart

audio out

notification

ui control

Flow 1

IBM IoT

connected

gauge

debug 1

Delete

Cancel

Done

Properties

Group

[CROP] MONITORING

Size

auto

Type

Gauge

Label

TEMPERATURE

Value format

{{value}}

Units

C

Range

min 0

max 100

Colour gradient

Sectors

0

optional

optional

100

Class

Optional CSS class name(s) for widget

Name

Enabled

debug

all nodes

all

2type/PNT2022TMD47477/Id/PNT2022TMD47477/evi
/event_tfmt/json : msg.payload : Object
» { temperature: 26, humidity: 26,
soil moisture: 75 }
11/5/2022, 11:24:38 AM node: debug 1
id:
2type/PNT2022TMD47477/Id/PNT2022TMD47477/evi
/event_tfmt/json : msg.payload : Object
» { temperature: 2, humidity: 82,
soil moisture: 53 }
11/5/2022, 11:24:44 AM node: debug 1
id:
2type/PNT2022TMD47477/Id/PNT2022TMD47477/evi
/event_tfmt/json : msg.payload : Object
» { temperature: 48, humidity: 95,
soil moisture: 82 }
11/5/2022, 11:24:50 AM node: debug 1
id:
2type/PNT2022TMD47477/Id/PNT2022TMD47477/evi
/event_tfmt/json : msg.payload : Object
» { temperature: 33, humidity: 40,
soil moisture: 90 }
11/5/2022, 11:24:56 AM node: debug 1
id:
2type/PNT2022TMD47477/Id/PNT2022TMD47477/evi
/event_tfmt/json : msg.payload : Object
» { temperature: 43, humidity: 2,
soil moisture: 86 }

STEP 5: PYTHON CODE:

```
1 Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep 5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on win32
2 Type "help", "copyright", "credits" or "license()" for more information.
3 import cv2
4 import numpy as np
5 import wiot.sdk.device
6 import playsound
7 import random
8 import time
9 import datetime
10 import ibm_boto3
11 from ibm_botocore.client import Config, ClientError
12
13 #CloudantDB
14 from cloudant.client import Cloudant
15 from cloudant.error import CloudantException
16 from cloudant.result import Result, ResultByKey
17 from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
18 from clarifai_grpc.grpc.api import service_pb2_grpc
19 stub = service_pb2_grpc.V2Stub(ClarifaiChannel.get_grpc_channel())
20 from clarifai_grpc.grpc.api import service_pb2, resource_pb2
21 from clarifai_grpc.grpc.api.status import status_code_pb2
22
23 #This is how you authenticate
24 metadata = (('authorization', 'key 0620e202302b4508b90eab7efe7475e4'),)
25 COS_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud"
26 COS_API_KEY_ID = "g5d4q08E1gv4TWUCJj4hfEzgalqEjrDbE82AJDWL1AOHo"
27 COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"
28 COS_RESOURCE_CRN = "crn:vl:bluemix:public:cloud-object-storage:global:a/c2fa2836eaf3434bbcb8b5b58fefff3f0:62e450fd-4c82-4153-ba41-ccb53adb8111::"
29 clientdb = cloudant("apikey-W2nj1dnwtj016V53LAVUCqPwc2aHTLmj1xXvtdGK3Bn", "88cc5f47c1a28afbfb8ad16161583f5a", url="https://d6c89f97-cf91-48b7-b14b-c99b2fe27c2f-bluemix.clouda
30 clientdb.connect()
31
32 #Create resource
33 cos = ibm_boto3.resource("s3",
34                          ibm_api_key_id=COS_API_KEY_ID,
35                          ibm_service_instance_id=COS_RESOURCE_CRN,
36                          ibm_auth_endpoint=COS_AUTH_ENDPOINT,
37                          config=Config(signature_version="oauth"),
38
39
40 def = multi_part_upload(bucket_name, item_name, file_path):
41     try:
42         print("Starting file transfer for {0} to bucket: {1}\n".format(item_name, bucket_name))
43         #set 5 MB chunks
44         part_size = 1024 * 1024 * 5
45         #set threshold to 15 MB
46         file_threshold = 1024 * 1024 * 15
47         #set the transfer threshold and chunk size
48         transfer_config = ibm_boto3.s3.transfer.TransferConfig(
49             multipart_threshold=file_threshold,
50             multipart_chunksize=part_size
51         )
52         #the upload_fileobj method will automatically execute a multi-part upload
53         #in 5 MB chunks size
54         with open(file_path, "rb") as file_data:
55             cos.Object(bucket_name, item_name).upload_fileobj(
56                 Fileobj=file_data,
57                 Config=transfer_config
58             )
59         print("Transfer for {0} Complete!\n".format(item_name))
60     except ClientError as be:
61         print("CLIENT ERROR: {0}\n".format(be))
62     except Exception as e:
63         print("Unable to complete multi-part upload: {0}".format(e))
64
65 def myCommandCallback(cmd):
66     print("Command received: %s" % cmd.data)
67     command=cmd.data['command']
68     print(command)
69     if(command=="lighton"):
70         print('lighton')
71     elif(command=="lightoff"):
72         print('lightoff')
73     elif(command=="motoron"):
74         print('motoron')
```



```

74     print('motoron')
75     elif(command=="motoroff"):
76         print('motoroff')
77     myConfig = {
78         "identity": {
79             "orgId": "chytun",
80             "typeId": "NodeMCU",
81             "deviceId": "12345"
82         },
83         "auth": {
84             "token": "12345678"
85         }
86     }
87     client = wiot.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
88     client.connect()
89
90     database_name = "sample"
91     my_database = clientdb.create_database(database_name)
92     if my_database.exists():
93         print(f'"{database_name}" successfully created.')
94     cap=cv2.VideoCapture("garden.mp4")
95     if(cap.isOpened()==True):
96         print('File opened')
97     else:
98         print('File not found')
99
100    while(cap.isOpened()):
101        ret, frame = cap.read()
102        gray = cv3.cvtColor(frame, cv2.COLOR_BGR@GRAY)
103        imS= cv2.resize(frame, (960,540))
104        cv2.imwrite('ex.jpg',imS)
105        with open("ex.jpg", "rb") as f:
106            file_bytes = f.read()
107        #This is the model ID of a publicly available General model. You may use any other public or custom model ID.
108        request = service_pb2.PostModelOutputsRequest(
109            model_id='e9359dbe6ee44dbc8842ebe97247b201',
110            inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.Image(base64=file_bytes))

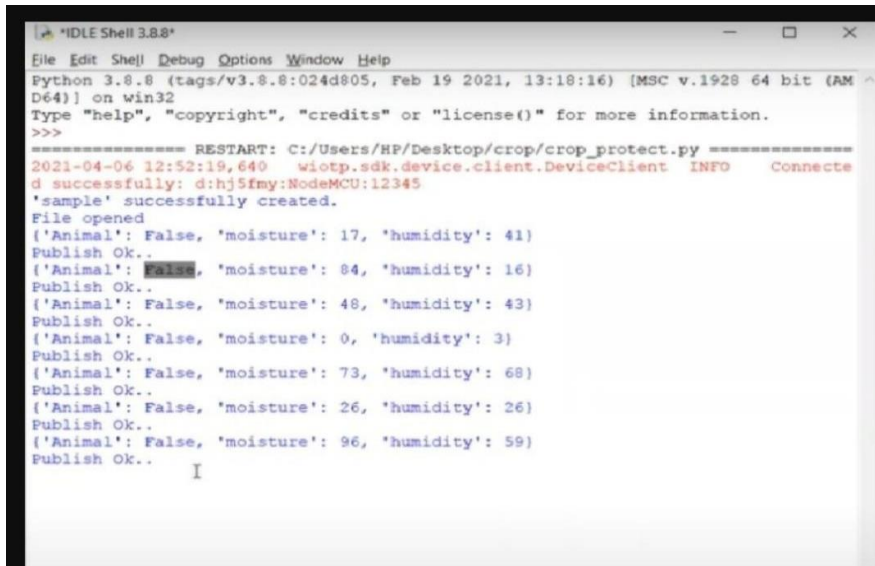
```

```

110    ...     inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.Image(base64=file_bytes))
111    ...                                     ))]
112    ...     response = stub.PostModelOutputs(request, metadata=metadata)
113    ...     if response.status.code != status_code_pb2.SUCCESS:
114    ...         raise Exception("Request failed, status code: " + str(response.status.code))
115    ...     detect=False
116    ...     for concept in response.outputs[0].data.concepts:
117    ...         #print('%12s: %.f' % (concept.name, concept.value))
118    ...         if(concept.value>0.98):
119    ...             #print(concept.name)
120    ...             if(concept.name=="animal"):
121    ...                 print("Alert! Alert! animal detected")
122    ...                 playsound('alert.mp3')
123    ...                 picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")
124    ...                 cv2.imwrite(picname+'.jpg',frame)
125    ...                 multi_part_upload('Dhakshesh', picname+'.jpg', picname+'.jpg')
126    ...                 json_document={"link":COS_ENDPOINT+'/'+'Dhakshesh'+'/'+'picname+'.jpg'}
127    ...                 new_document = my_database.create_document(json_document)
128    ...                 if new_document.exists():
129    ...                     print(f"Document successfully created.")
130    ...                     time.sleep(5)
131    ...                     detect=True
132    ...                     moist=random.randint(0,100)
133    ...                     humidity=random.randint(0,100)
134    ...                     myData={'Animal':detect,'moisture':moist,'humidity':humidity}
135    ...                     print(myData)
136    ...                     if(humidity!=None):
137    ...                         client.publishEvent(eventId="status",msgFormat="json", daya=myData, qos=0, onPublish=None)
138    ...                         print("Publish Ok..")
139    ...                         client.commandCallback = myCommandCallback
140    ...                         cv2.imshow('frame',imS)
141    ...                         if cv2.waitKey(1) & 0xFF == ord('q'):
142    ...                             break
143    ...     client.disconnect()
144    ...     cap.release()

```

STEP 6: OUTPUT



```
IDLE Shell 3.8.8
File Edit Shell Debug Options Window Help
Python 3.8.8 (tags/v3.8.8:024d805, Feb 19 2021, 13:18:16) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/HP/Desktop/crop/crop_protect.py =====
2021-04-06 12:52:19,640 wiotp.sdk.device.client.DeviceClient INFO Connecte
d successfully: d:hj5fmy:NodeMCU:12345
'sample' successfully created.
File opened
{'Animal': False, 'moisture': 17, 'humidity': 41}
Publish Ok..
{'Animal': False, 'moisture': 84, 'humidity': 16}
Publish Ok..
{'Animal': False, 'moisture': 48, 'humidity': 43}
Publish Ok..
{'Animal': False, 'moisture': 0, 'humidity': 3}
Publish Ok..
{'Animal': False, 'moisture': 73, 'humidity': 68}
Publish Ok..
{'Animal': False, 'moisture': 26, 'humidity': 26}
Publish Ok..
{'Animal': False, 'moisture': 96, 'humidity': 59}
Publish Ok..
I
```