FINAL DELIVERABLES

PROJECT REPORT

TEAM ID	PNT2022TMID49419
PROJECT TITLE	IoT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

OBJECTIVES:

Agriculture is the backbone of the economy but because of animal interface in agricultural lands, there will be huge loss of crops. This paper monitor and protection procedure for farm safety against animal attacks. It is proposed to develop a smart agriculture system that makes use of technologies such as ARDUINO, IOT and WSN. The feature of this paper incudes development of a system which can monitor temperature, humidity, water level and even the movement of animals through SENSORS using ARDUINO board. In case of any obstacle detected sends a notification to the application developed for the same to the farmer's smartphone using Wi-Fi. Hence, every time farmer will get to know the changes his farm. This project will be more helpful for the farmer's welfare.

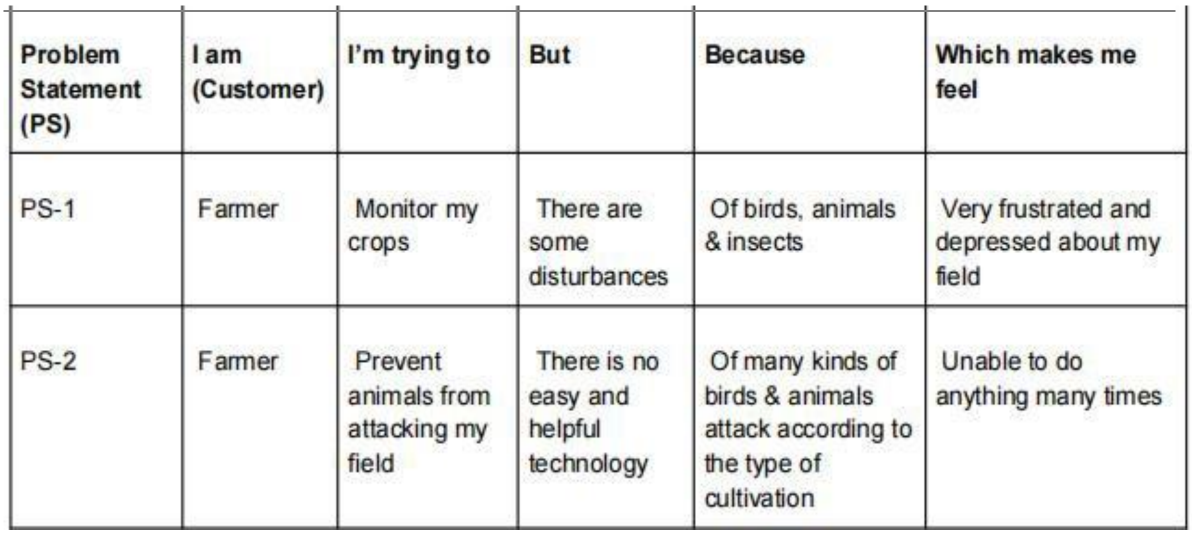
Design Methodology:

The most important factors needed are Internet, Arduino UNO, sensors such as temperature sensor, humidity sensor, u v sensor, u v cameras for image processing, wi-fi module, GSM module, PIR sensor, motion detector and a smartphone that is connected to the internet and the Arduino. All the sensors are connected to the Arduino and are placed at its specific coverage distances. The temperature and humidity levels are monitored and graphs are updated every 1 hour. If the humidity level and temperature increase or decreases from its normal level an intimation is sent in the form of message and mail to the connected smartphone. The graphs of these are stored in the cloud for future references.
The PIR sensor and UV sensors detect the motion of animals and birds for a particular range. The thermal radiation temperature of humans at different ages is fed to the system so there won't be any false alarm. If any invasion of animals is found, the u v camera focuses on the region and the processed image is sent to the farmer. After seeing the image of the animal that entered, they can decide to take any actions. A fence is built around the field to prevent large animals from entering where the sensors are placed at all the corners of the field fully covering the entire region.

Implementation:

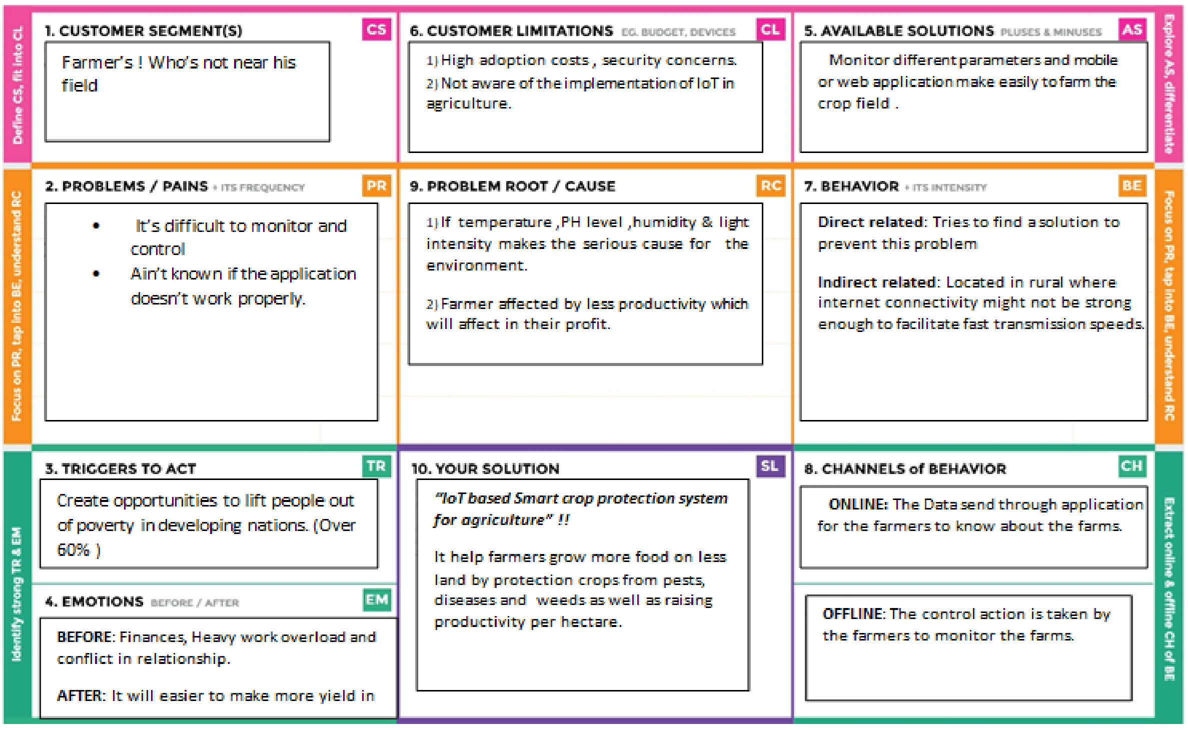
Firstly, we should create codes for connecting the sensors to the Arduino and connecting the Arduino to the Wi-fi module and connecting them to the Internet. Then we should create codes to monitor and intimate messages about humidity and temperature on a regular basis, and codes should be written for PIR sensor and UV sensor to make sure that the motion detection of animals is being intimidated and preventive measures are taken. The preventive measure for every problem should be given according to the problem that arose and the codes for every problem and their solution should be fed on the cloud to access and as a result if the person doesn't know what to do in this type of situation then they can refer to the solutions. Codes should be written to not to intimate humans and also there should be power backup for the system to function efficiently. The backup system is solar and all the products used should consume less power and function more efficiently. The system should be made in a way that it can function more effectively even when there is very low data rate. The program should be coded in such a way.

PROBLEM STATEMENT:



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Farmer	Monitor my crops	There are some disturbances	Of birds, animals & insects	Very frustrated and depressed about my field
PS-2	Farmer	Prevent animals from attacking my field	There is no easy and helpful technology	Of many kinds of birds & animals attack according to the type of cultivation	Unable to do anything many times

PROPOSED SOLUTION FIT:



Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS

Farmer's ! Who's not near his field

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL

1) High adoption costs , security concerns.
2) Not aware of the implementation of IoT in agriculture.

5. AVAILABLE SOLUTIONS PLUSES & MINUSES AS

Monitor different parameters and mobile or web application make easily to farm the crop field .

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR

- It's difficult to monitor and control
- Ain't known if the application doesn't work properly.

9. PROBLEM ROOT / CAUSE RC

1) If temperature ,PH level ,humidity & light intensity makes the serious cause for the environment.

2) Farmer affected by less productivity which will affect in their profit.

7. BEHAVIOR + ITS INTENSITY BE

Direct related: Tries to find a solution to prevent this problem

Indirect related: Located in rural where internet connectivity might not be strong enough to facilitate fast transmission speeds.

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR

Create opportunities to lift people out of poverty in developing nations. (Over 60%)

4. EMOTIONS BEFORE / AFTER EM

BEFORE: Finances, Heavy work overload and conflict in relationship.

AFTER: It will easier to make more yield in

10. YOUR SOLUTION SL

"IoT based Smart crop protection system for agriculture" !!

It help farmers grow more food on less land by protection crops from pests, diseases and weeds as well as raising productivity per hectare.

8. CHANNELS of BEHAVIOR CH

ONLINE: The Data send through application for the farmers to know about the farms.

OFFLINE: The control action is taken by the farmers to monitor the farms.

Extract online & offline CH of BE

REQUIRED SOFTWARE:

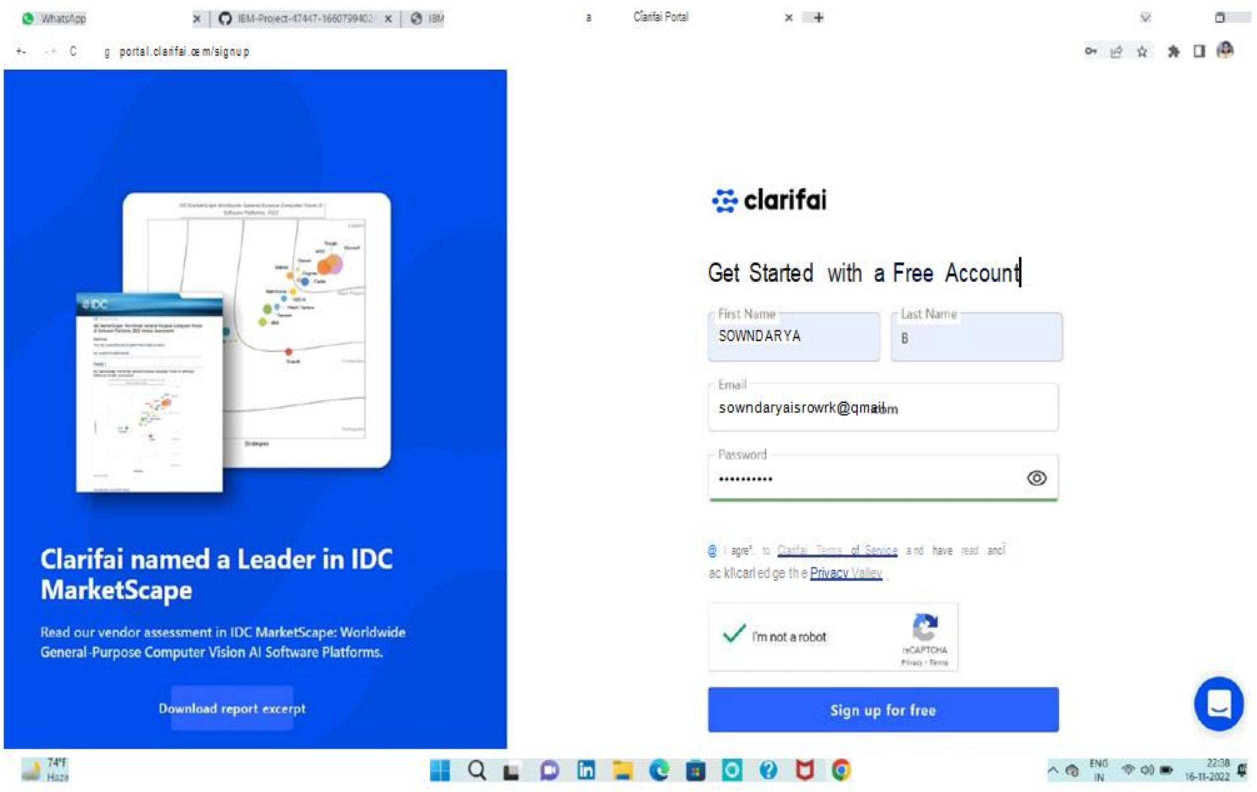
- **CLARIFAI**
- **IBM WATSON IOT PLATFORM**
- **PYTHON IDLE**
- **NODE RED**
- **MIT APP INVENTOR**

CLARIFAI:

Clarifai provides an end-to-end platform with the easiest to use UI and API in the market. Clarifai Inc. is an artificial intelligence (AI) company that specializes in computer vision and uses machine learning and deep neural networks to identify and images and videos. The company offers its solution via API, mobile SDK, and on-premise solutions.

STEP 1:

- Open Clarifai portal in web browser.



portal.clarifai.com/signup

Clarifai named a Leader in IDC MarketScape

Read our vendor assessment in IDC MarketScape: Worldwide General-Purpose Computer Vision AI Software Platforms.

Download report excerpt

clarifai

Get Started with a Free Account

First Name: SOWNDARYA

Last Name: B

Email: sowndaryaisrowrk@qmail.com

Password: ••••••••••

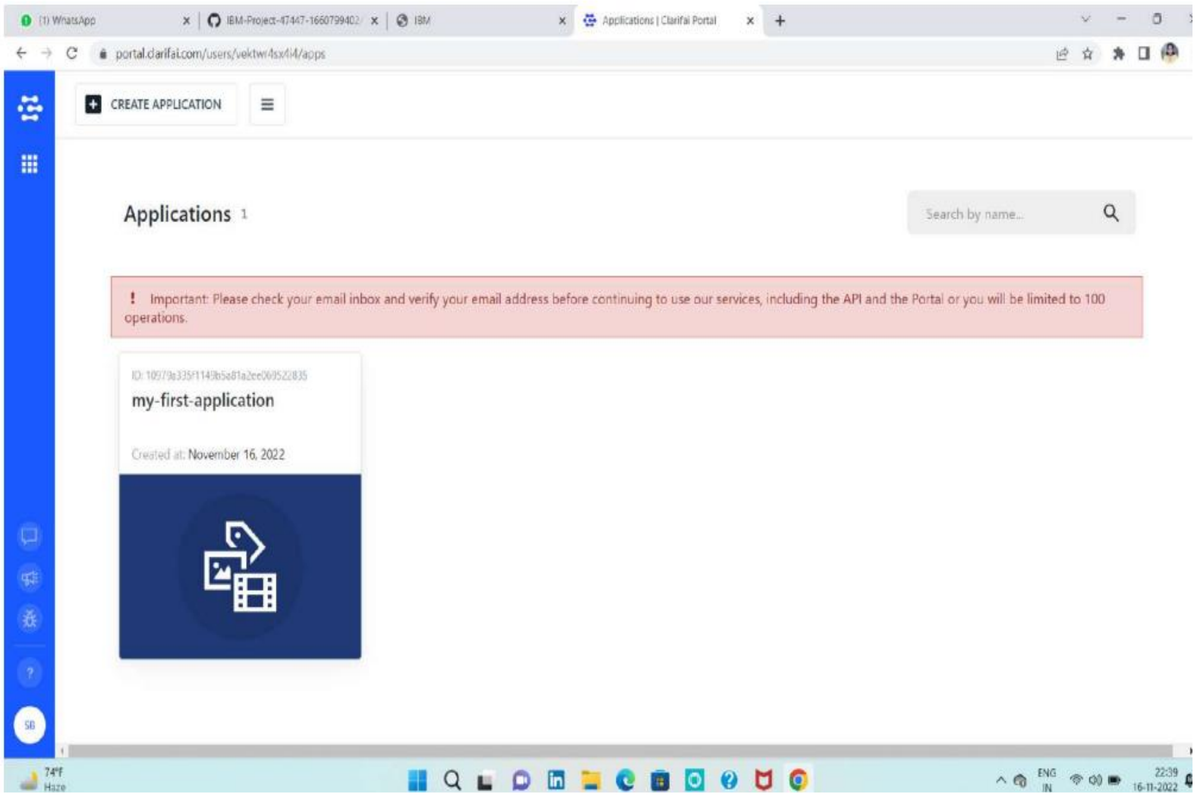
I agree to Clarifai Terms of Service and have read and acknowledge the Privacy Policy

I'm not a robot

Sign up for free

STEP 2:

Finally, Created an account


(1) WhatsApp
IBM-Project-47447-1660799402
IBM
Applications | Clarifai Portal
portal.clarifai.com/users/vektwr4sx4i4/apps
CREATE APPLICATION
Search by name...

Applications 1

! Important: Please check your email inbox and verify your email address before continuing to use our services, including the API and the Portal or you will be limited to 100 operations.

ID: 10979a335f1149b5a81a2ee069522835

my-first-application

Created at: November 16, 2022

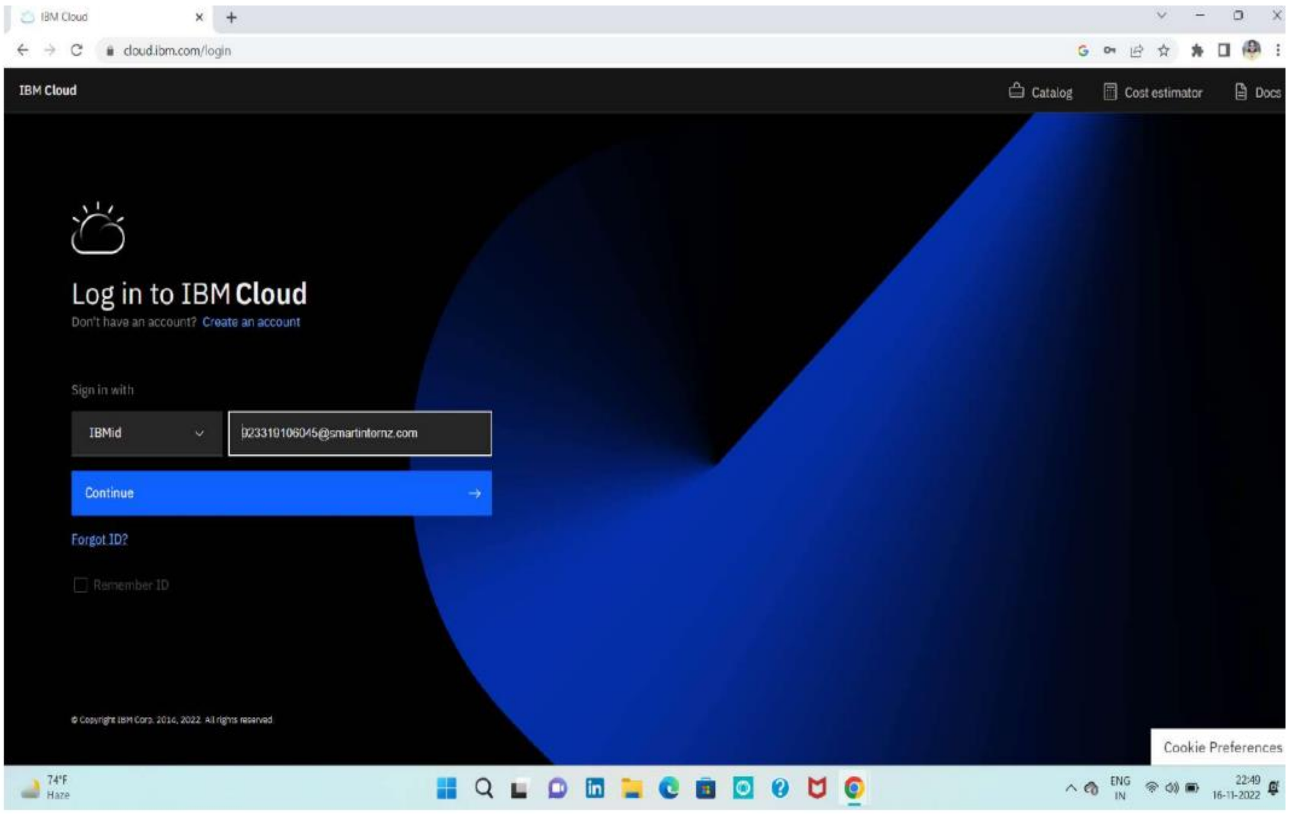
IBM WATSON IoT PLATFORM:

We need to have basic knowledge of the following cloud services:

- IBM Watson IoT Platform
- Node-RED Service
- Cloudant DB

We need to create an IBM Cloud Account to complete this project.

LOGIN:



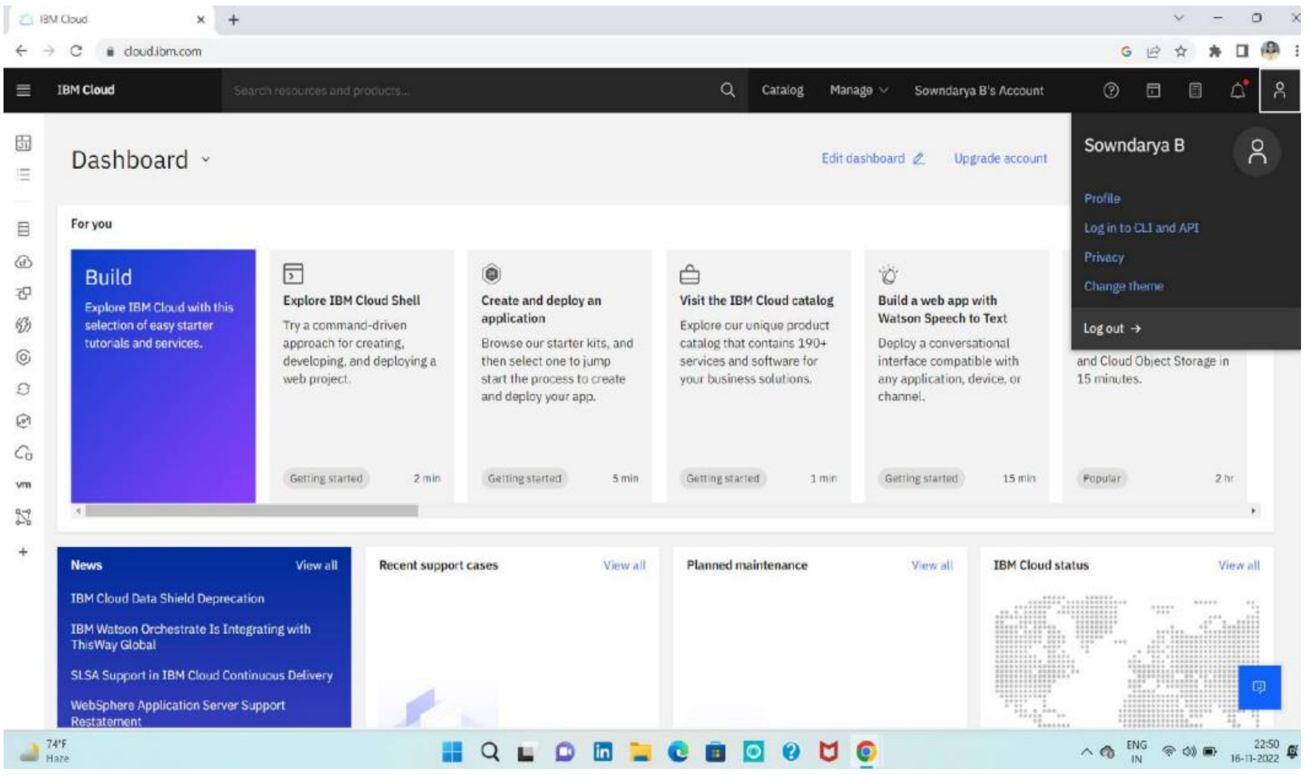
Log in to IBM Cloud

Don't have an account? Create an account

Sign in with: IBMid

923319106045@smartinternz.com

© Copyright IBM Corp. 2014, 2022. All rights reserved.



Build: Explore IBM Cloud with this selection of easy starter tutorials and services.

Explore IBM Cloud Shell: Try a command-driven approach for creating, developing, and deploying a web project. Getting started 2 min

Create and deploy an application: Browse our starter kits, and then select one to jump start the process to create and deploy your app. Getting started 5 min

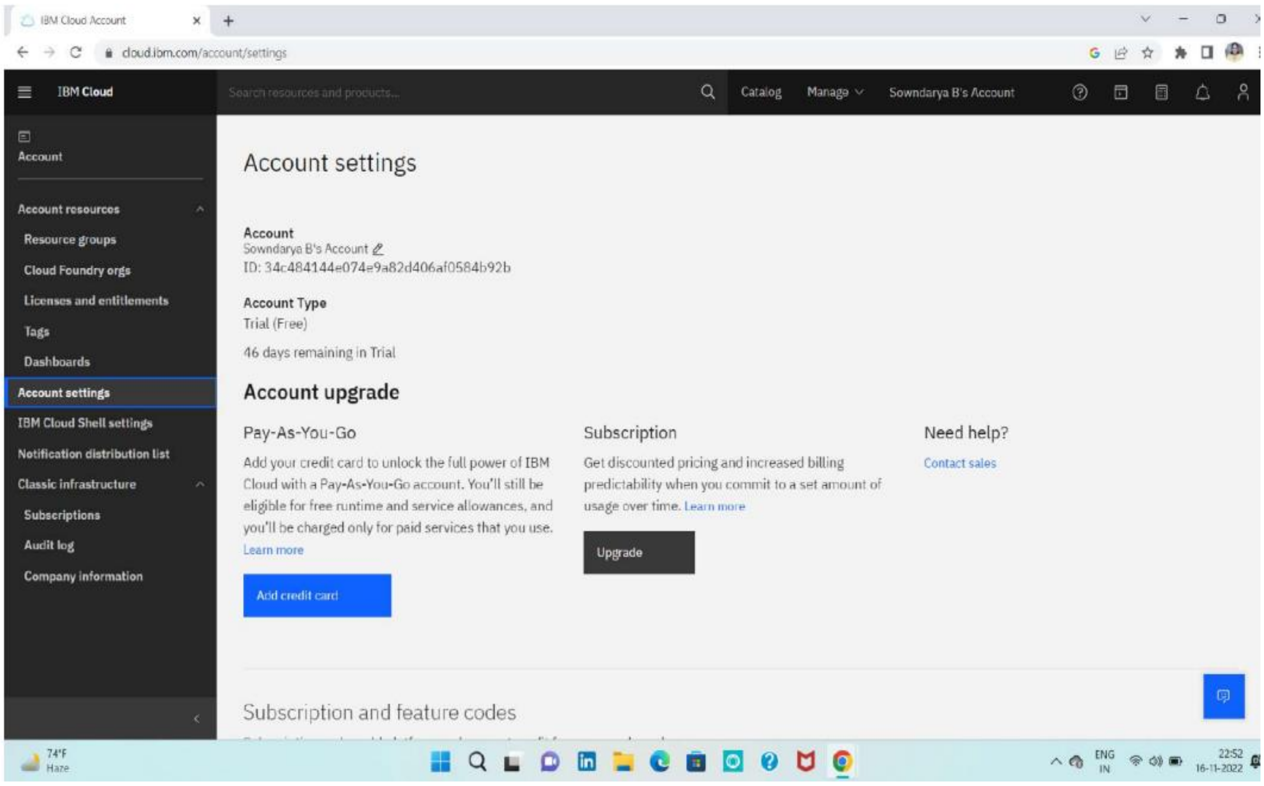
Visit the IBM Cloud catalog: Explore our unique product catalog that contains 190+ services and software for your business solutions. Getting started 1 min

Build a web app with Watson Speech to Text: Deploy a conversational interface compatible with any application, device, or channel. Getting started 15 min

and Cloud Object Storage in 15 minutes. Popular 2 hr

News — View all
- IBM Cloud Data Shield Deprecation
- IBM Watson Orchestrate Is Integrating with ThisWay Global
- SLSA Support in IBM Cloud Continuous Delivery
- WebSphere Application Server Support Restatement

Recent support cases — View all
Planned maintenance — View all
IBM Cloud status — View all
74°F Haze
ENG IN
22:50 16-11-2022


IBM Cloud Account

cloud.ibm.com/account/settings

IBM Cloud

Account settings

Account
Sowndarya B's Account
ID: 34c484144e074e9a82d406af0584b92b

Account Type
Trial (Free)

46 days remaining in Trial

Account upgrade

Pay-As-You-Go
Add your credit card to unlock the full power of IBM Cloud with a Pay-As-You-Go account. You'll still be eligible for free runtime and service allowances, and you'll be charged only for paid services that you use. Learn more

Subscription
Get discounted pricing and increased billing predictability when you commit to a set amount of usage over time. Learn more

Need help?
Contact sales

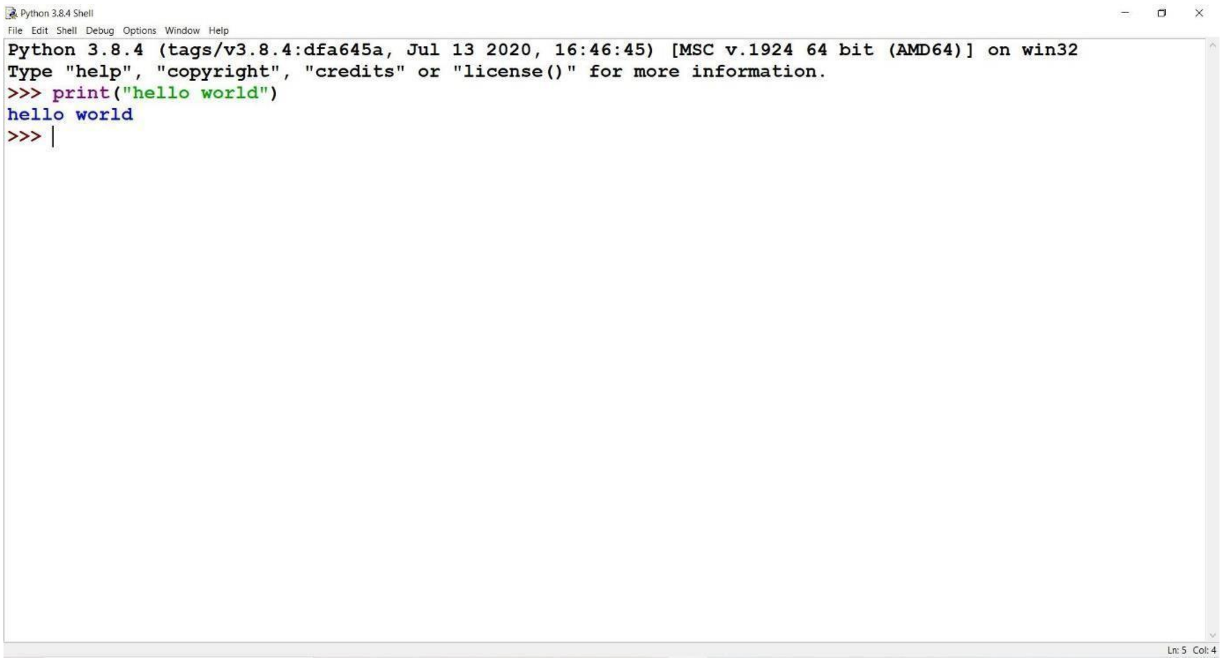
Subscription and feature codes

PYTHON IDLE INSTALLATION:

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general- purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems.

STEP 1:

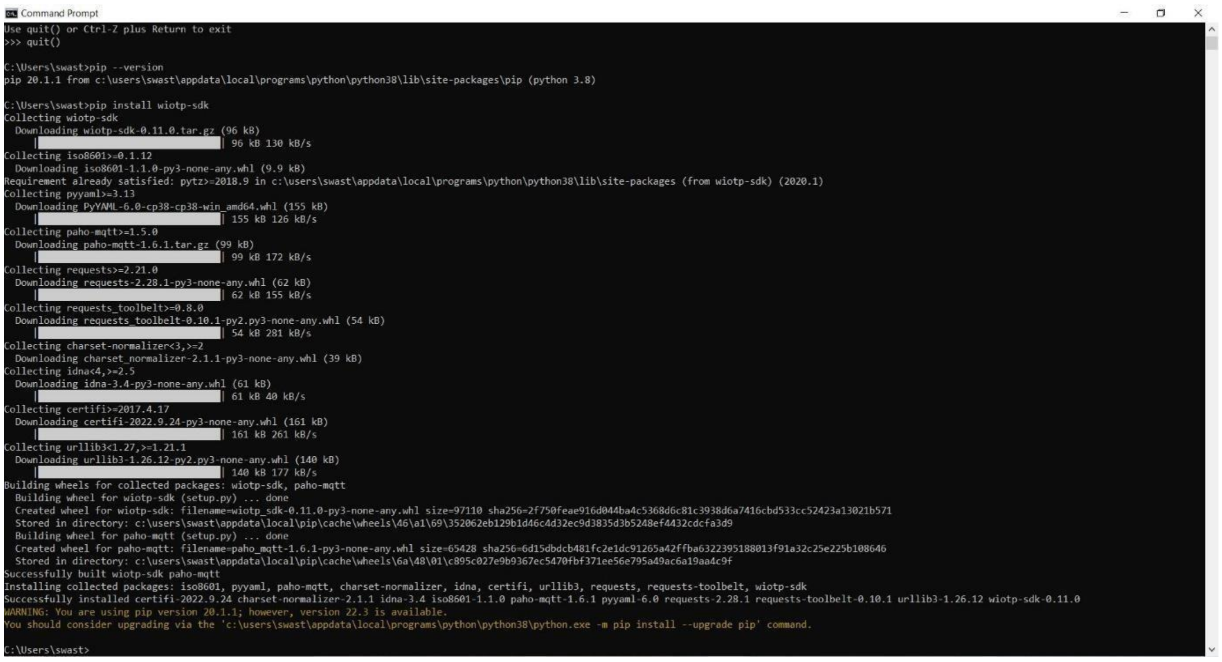
- Python is installed successfully



```
Python 3.8.4 (tags/v3.8.4:dfa645a, Jul 13 2020, 16:46:45) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("hello world")
hello world
>>> 
```

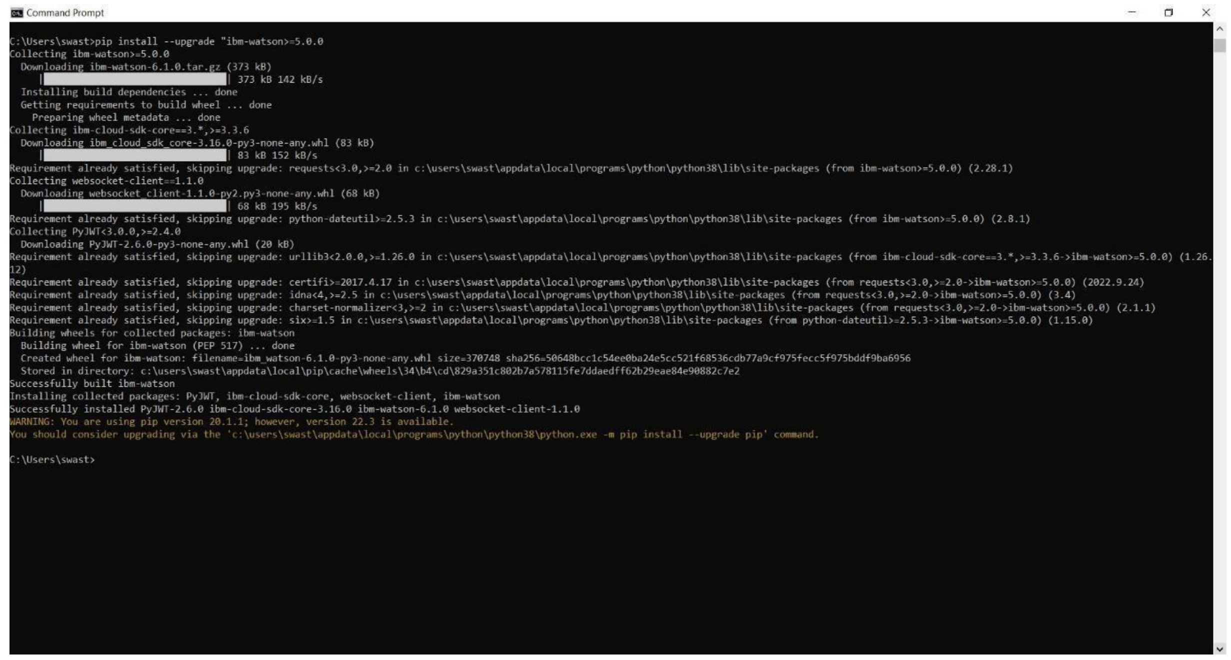
STEP 2:

- The required python libraries are installed.
- Watson IoT Python SDK to connect to IBM Watson IoT Platform using python code is installed
- pip install wiotp-sdk



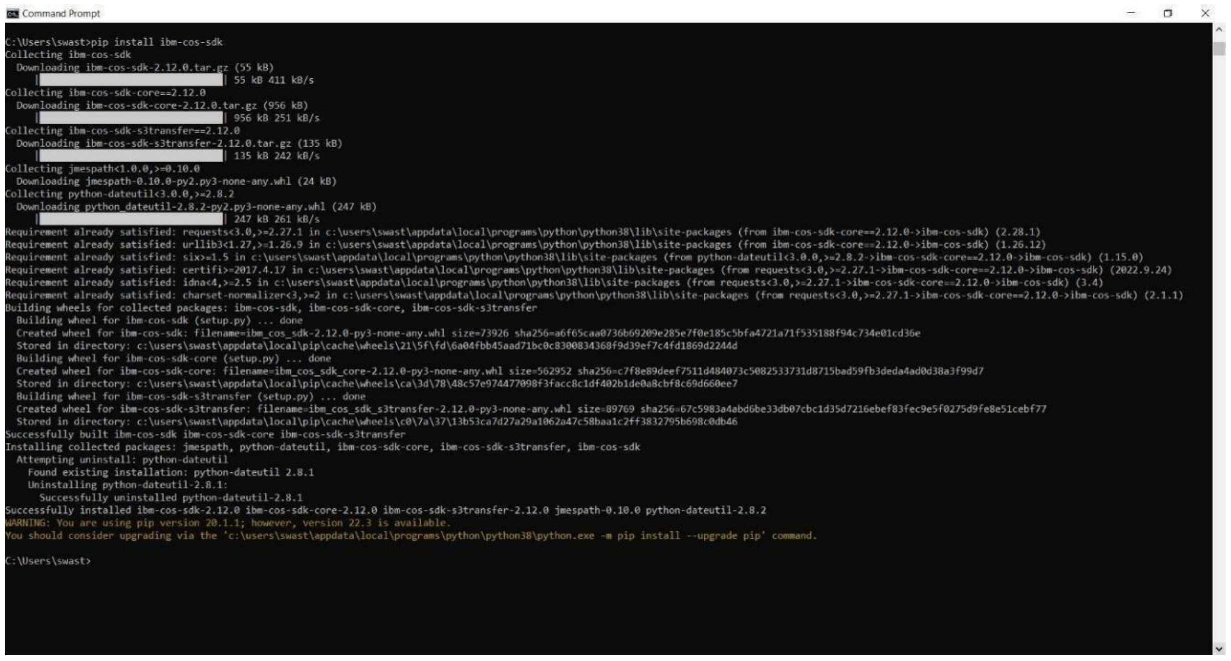
```
Use quit() or Ctrl-Z plus Return to exit
>>> quit()

C:\Users\swast>pip --version
pip 20.1.1 from c:\users\swast\appdata\local\programs\python\python38\lib\site-packages\pip (python 3.8)

C:\Users\swast>pip install wiotp-sdk
Collecting wiotp-sdk
  Downloading wiotp-sdk-0.11.0.tar.gz (96 kB)
     |████████████████████████████████| 96 kB 130 kB/s
Collecting iso8601>=0.1.12
  Downloading iso8601-1.1.0-py3-none-any.whl (9.9 kB)
Requirement already satisfied: pytz>=2018.9 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from wiotp-sdk) (2020.1)
Collecting pyyaml>=3.13
  Downloading PyYAML-6.0-cp38-cp38-win_amd64.whl (155 kB)
     |████████████████████████████████| 155 kB 126 kB/s
Collecting paho-mqtt>=1.5.0
  Downloading paho-mqtt-1.6.1.tar.gz (99 kB)
     |████████████████████████████████| 99 kB 172 kB/s
Collecting requests>=2.21.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     |████████████████████████████████| 62 kB 155 kB/s
Collecting requests_toolbelt>=0.8.0
  Downloading requests_toolbelt-0.10.1-py2.py3-none-any.whl (54 kB)
     |████████████████████████████████| 54 kB 281 kB/s
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.1-py3-none-any.whl (39 kB)
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
     |████████████████████████████████| 61 kB 40 kB/s
Collecting certifi>=2017.4.17
  Downloading certifi-2022.9.24-py3-none-any.whl (161 kB)
     |████████████████████████████████| 161 kB 261 kB/s
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)
     |████████████████████████████████| 140 kB 177 kB/s
Building wheels for collected packages: wiotp-sdk, paho-mqtt
  Building wheel for wiotp-sdk (setup.py) ... done
  Created wheel for wiotp-sdk: filename=wiotp_sdk-0.11.0-py3-none-any.whl size=97110 sha256=2f750feae916d044ba4c5368d6c81c3938d6a7416cbd533cc52423a13021b571
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\46\a1\69\352062eb129b1d46c4d32ec9d3835d3b5248ef4432cdcfa3d9
  Building wheel for paho-mqtt (setup.py) ... done
  Created wheel for paho-mqtt: filename=paho_mqtt-1.6.1-py3-none-any.whl size=65428 sha256=6d15dbdcb481fc2e1dc91265a42ffba6322395188013f91a32c25e225b108646
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\6a\48\01\c895c027e9b9367ec5470fbf371ee56e795a49ac6a19aa4c9f
Successfully built wiotp-sdk paho-mqtt
Installing collected packages: iso8601, pyyaml, paho-mqtt, charset-normalizer, idna, certifi, urllib3, requests, requests-toolbelt, wiotp-sdk
Successfully installed certifi-2022.9.24 charset-normalizer-2.1.1 idna-3.4 iso8601-1.1.0 paho-mqtt-1.6.1 pyyaml-6.0 requests-2.28.1 requests-toolbelt-0.10.1 urllib3-1.26.12 wiotp-sdk-0.11.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- Python client library for IBM Text to Speech is installed
- pip install –upgrade "ibm-watson>=5.0.0



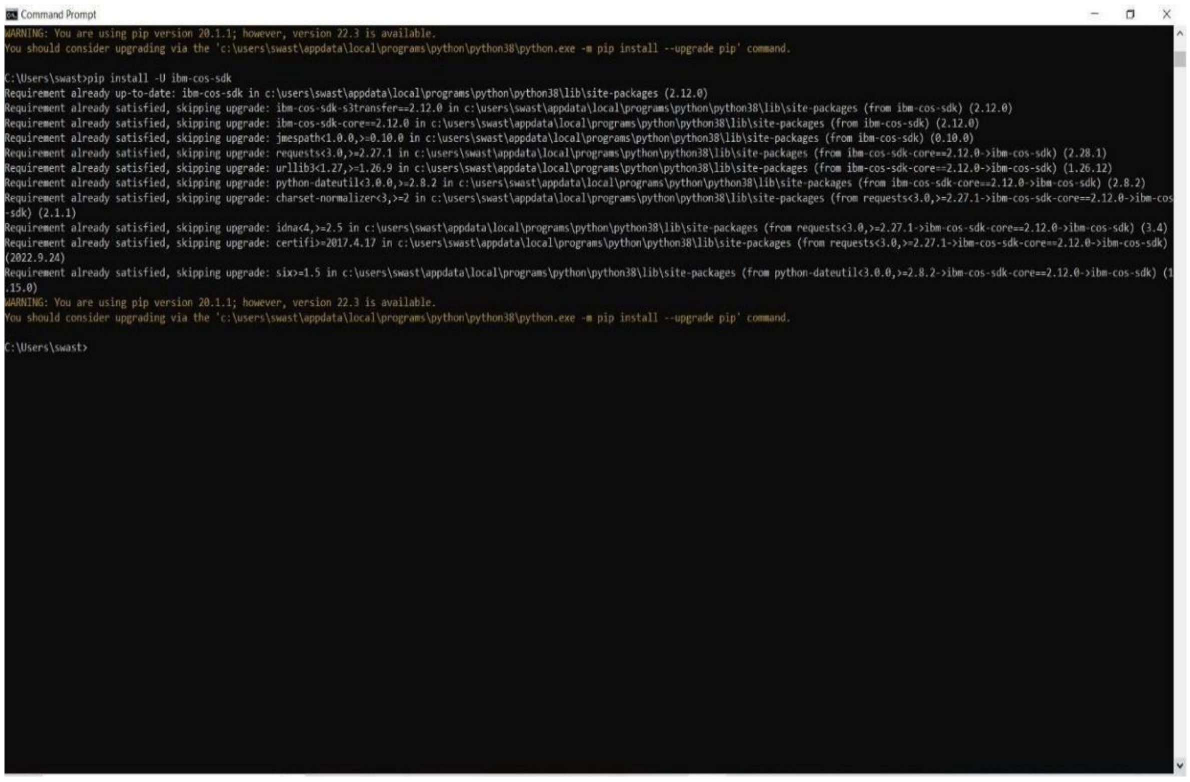
```
C:\Users\swast>pip install --upgrade "ibm-watson>=5.0.0
Collecting ibm-watson>=5.0.0
  Downloading ibm-watson-6.1.0.tar.gz (373 kB)
     |████████████████████████████████| 373 kB 142 kB/s
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
    Preparing wheel metadata ... done
Collecting ibm-cloud-sdk-core==3.*,>=3.3.6
  Downloading ibm_cloud_sdk_core-3.16.0-py3-none-any.whl (83 kB)
     |████████████████████████████████| 83 kB 152 kB/s
Requirement already satisfied, skipping upgrade: requests<3.0,>=2.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-watson>=5.0.0) (2.28.1)
Collecting websocket-client==1.1.0
  Downloading websocket_client-1.1.0-py2.py3-none-any.whl (68 kB)
     |████████████████████████████████| 68 kB 195 kB/s
Requirement already satisfied, skipping upgrade: python-dateutil>=2.5.3 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-watson>=5.0.0) (2.8.1)
Collecting PyJWT<3.0.0,>=2.4.0
  Downloading PyJWT-2.6.0-py3-none-any.whl (20 kB)
Requirement already satisfied, skipping upgrade: urllib3<2.0.0,>=1.26.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cloud-sdk-core==3.*,>=3.3.6->ibm-watson>=5.0.0) (1.26.
12)
Requirement already satisfied, skipping upgrade: certifi>=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.0->ibm-watson>=5.0.0) (2022.9.24)
Requirement already satisfied, skipping upgrade: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.0->ibm-watson>=5.0.0) (3.4)
Requirement already satisfied, skipping upgrade: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.0->ibm-watson>=5.0.0) (2.1.1)
Requirement already satisfied, skipping upgrade: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil>=2.5.3->ibm-watson>=5.0.0) (1.15.0)
Building wheels for collected packages: ibm-watson
  Building wheel for ibm-watson (PEP 517) ... done
  Created wheel for ibm-watson: filename=ibm_watson-6.1.0-py3-none-any.whl size=370748 sha256=50648bcc1c54ee0ba24e5cc521f68536cdb77a9cf975fecc5f975bddf9ba6956
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\34\b4\cd\829a351c802b7a578115fe7ddaedff62b29eae84e90882c7e2
Successfully built ibm-watson
Installing collected packages: PyJWT, ibm-cloud-sdk-core, websocket-client, ibm-watson
Successfully installed PyJWT-2.6.0 ibm-cloud-sdk-core-3.16.0 ibm-watson-6.1.0 websocket-client-1.1.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- Required Libraries for cloud object storage is installed.
- pip install ibm-cos-sdk



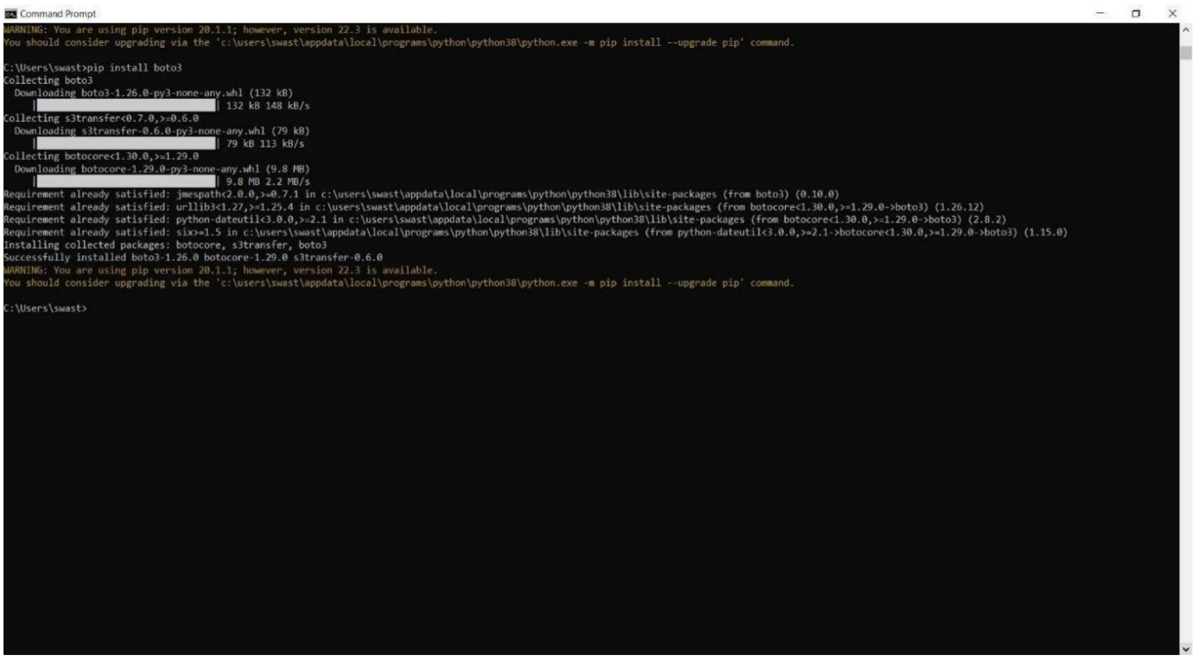
```
C:\Users\swast>pip install ibm-cos-sdk
Collecting ibm-cos-sdk
  Downloading ibm-cos-sdk-2.12.0.tar.gz (55 kB)
     |████████████████████████████████| 55 kB 411 kB/s
Collecting ibm-cos-sdk-core==2.12.0
  Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)
     |████████████████████████████████| 956 kB 251 kB/s
Collecting ibm-cos-sdk-s3transfer==2.12.0
  Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB)
     |████████████████████████████████| 135 kB 242 kB/s
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 kB 261 kB/s
Requirement already satisfied: requests<3.0,>=2.27.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.28.1)
Requirement already satisfied: urllib3<1.27,>=1.26.9 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.26.12)
Requirement already satisfied: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil<3.0.0,>=2.8.2->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.15.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2022.9.24)
Requirement already satisfied: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (3.4)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.1.1)
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=73926 sha256=a6f65caa0736b69209e285e7f0e185c5bfa4721a71f535188f94c734e01cd36e
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\21\5f\fd\6a04fbb45aad71bc0c8300834368f9d39ef7c4fd1869d2244d
  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=562952 sha256=c7f8e89deef7511d484073c5082533731d8715bad59fb3deda4ad0d38a3f99d7
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\ca\3d\78\48c57e974477098f3facc8c1df402b1de0a8cbf8c69d660ee7
  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=89769 sha256=67c5983a4abd6be33db07cbc1d35d7216ebef83fec9e5f0275d9fe8e51cebf77
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\c0\7a\37\13b53ca7d27a29a1062a47c58baa1c2ff3832795b698c0db46
Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer
Installing collected packages: jmespath, python-dateutil, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer, ibm-cos-sdk
  Attempting uninstall: python-dateutil
    Found existing installation: python-dateutil 2.8.1
    Uninstalling python-dateutil-2.8.1:
      Successfully uninstalled python-dateutil-2.8.1
Successfully installed ibm-cos-sdk-2.12.0 ibm-cos-sdk-core-2.12.0 ibm-cos-sdk-s3transfer-2.12.0 jmespath-0.10.0 python-dateutil-2.8.2
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- pip install -U ibm-cos-sdk



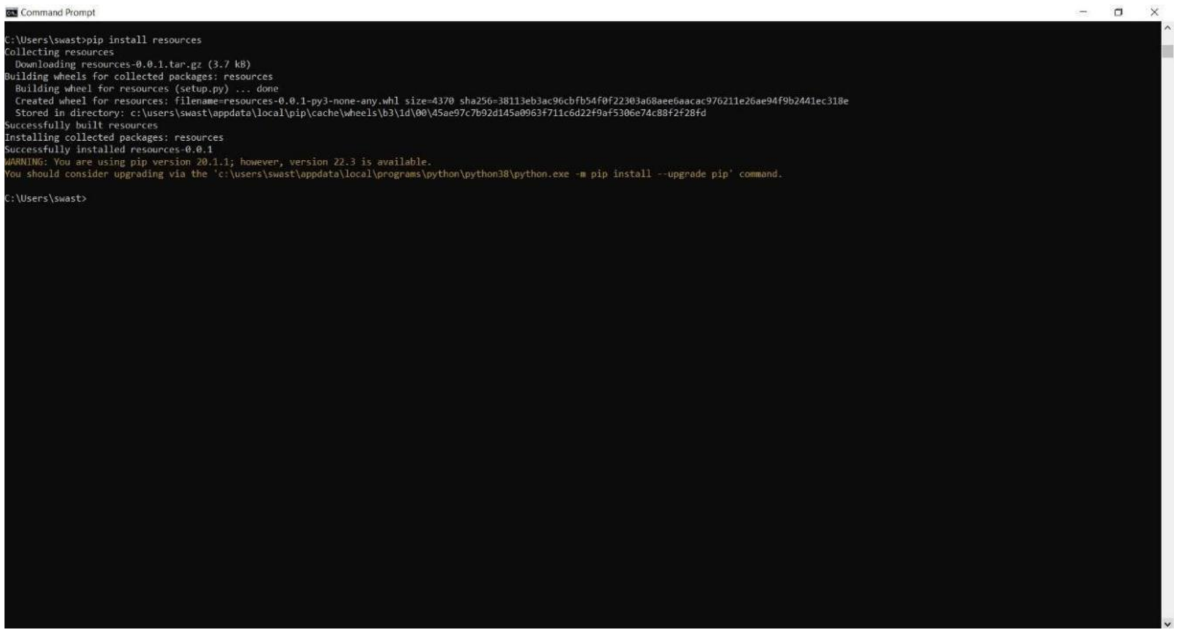
```
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>pip install -U ibm-cos-sdk
Requirement already up-to-date: ibm-cos-sdk in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (2.12.0)
Requirement already satisfied, skipping upgrade: ibm-cos-sdk-s3transfer==2.12.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk) (2.12.0)
Requirement already satisfied, skipping upgrade: ibm-cos-sdk-core==2.12.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk) (2.12.0)
Requirement already satisfied, skipping upgrade: jmespath<1.0.0,>=0.10.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk) (0.10.0)
Requirement already satisfied, skipping upgrade: requests<3.0,>=2.27.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.28.1)
Requirement already satisfied, skipping upgrade: urllib3<1.27,>=1.26.9 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1.26.12)
Requirement already satisfied, skipping upgrade: python-dateutil<3.0.0,>=2.8.2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (2.8.2)
Requirement already satisfied, skipping upgrade: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos
-sdk) (2.1.1)
Requirement already satisfied, skipping upgrade: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (3.4)
Requirement already satisfied, skipping upgrade: certifi>=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0,>=2.27.1->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk)
(2022.9.24)
Requirement already satisfied, skipping upgrade: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil<3.0.0,>=2.8.2->ibm-cos-sdk-core==2.12.0->ibm-cos-sdk) (1
.15.0)
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- pip install boto3



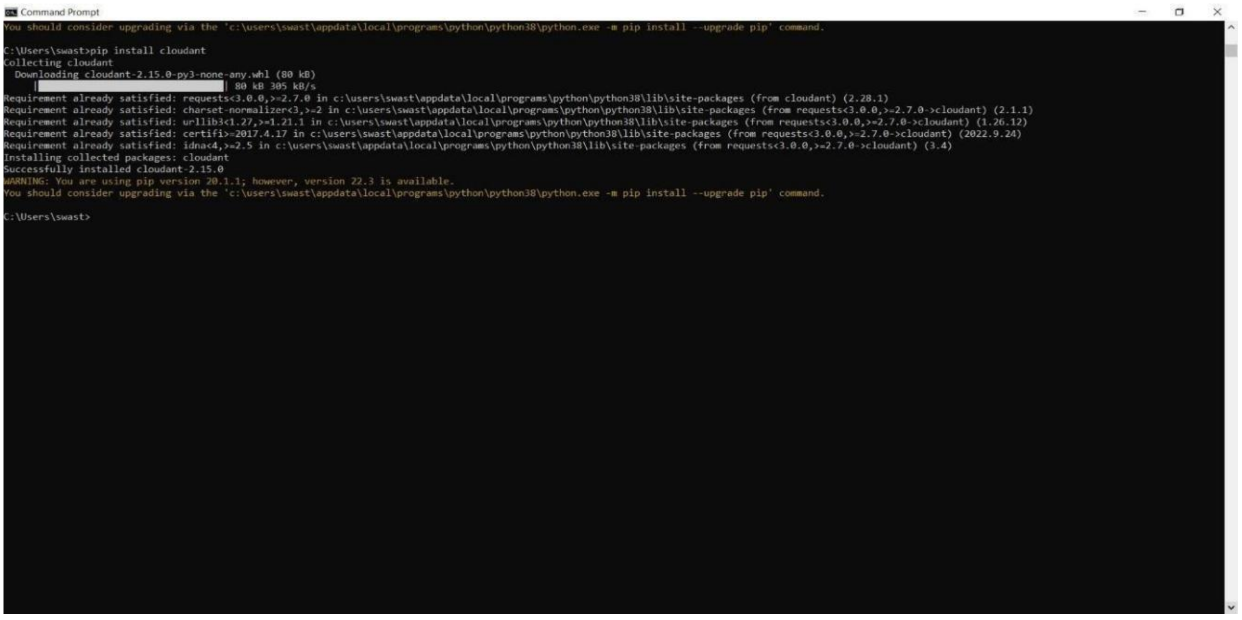
```
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>pip install boto3
Collecting boto3
  Downloading boto3-1.26.0-py3-none-any.whl (132 kB)
     |████████████████████████████████| 132 kB 148 kB/s
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
     |████████████████████████████████| 79 kB 113 kB/s
Collecting botocore<1.30.0,>=1.29.0
  Downloading botocore-1.29.0-py3-none-any.whl (9.8 MB)
     |████████████████████████████████| 9.8 MB 2.2 MB/s
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from boto3) (0.10.0)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from botocore<1.30.0,>=1.29.0->boto3) (1.26.12)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from botocore<1.30.0,>=1.29.0->boto3) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.30.0,>=1.29.0->boto3) (1.15.0)
Installing collected packages: botocore, s3transfer, boto3
Successfully installed boto3-1.26.0 botocore-1.29.0 s3transfer-0.6.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- pip install resources



```
C:\Users\swast>pip install resources
Collecting resources
  Downloading resources-0.0.1.tar.gz (3.7 kB)
Building wheels for collected packages: resources
  Building wheel for resources (setup.py) ... done
  Created wheel for resources: filename=resources-0.0.1-py3-none-any.whl size=4370 sha256=38113eb3ac96cbfb54f0f22303a68aee6aacac976211e26ae94f9b2441ec318e
  Stored in directory: c:\users\swast\appdata\local\pip\cache\wheels\b3\1d\00\45ae97c7b92d145a0963f711c6d22f9af5306e74c88f2f28fd
Successfully built resources
Installing collected packages: resources
Successfully installed resources-0.0.1
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

- pip install cloudant



```
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>pip install cloudant
Collecting cloudant
  Downloading cloudant-2.15.0-py3-none-any.whl (80 kB)
     |████████████████████████████████| 80 kB 305 kB/s
Requirement already satisfied: requests<3.0.0,>=2.7.0 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from cloudant) (2.28.1)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (2.1.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (1.26.12)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (2022.9.24)
Requirement already satisfied: idna<4,>=2.5 in c:\users\swast\appdata\local\programs\python\python38\lib\site-packages (from requests<3.0.0,>=2.7.0->cloudant) (3.4)
Installing collected packages: cloudant
Successfully installed cloudant-2.15.0
WARNING: You are using pip version 20.1.1; however, version 22.3 is available.
You should consider upgrading via the 'c:\users\swast\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.

C:\Users\swast>
```

DATA FROM PYTHON TO IBM:

Python code to generate random data and pass it to IBM Watson IoT platform

Source Code:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device import
random


            #Provide your IBM Watson Device
            Credentialsorganization = "wu5b55"

            deviceType = "crop1"
            deviceId = "1234"
            authMethod =
            "token" authToken =
            "1234567890"


            # Initialize
            GPIOtry:

                    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.conn

            ect()while

            True:


                temp=random.randint(0,
                100)
```

```
        Hum=random.randint(0,1
        00)
        moisture=random.randint
        (0,100)

        data = { 'temperature' : temp, 'Humidity': Hum, 'Moisture':moisture }



def myOnPublishCallback():
          print ("Temperature = " + str(temp)+" C Humidity = " +
      str(hum)+ " moisture = " +str(moisture) + "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor",
      "json", data, qos=0,on_publish=myOnPublishCallback)
        if not success:
          print("Not connected to
        IoTF")time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

      # Disconnect the device and application
      from the clouddeviceCli.disconnect()
```

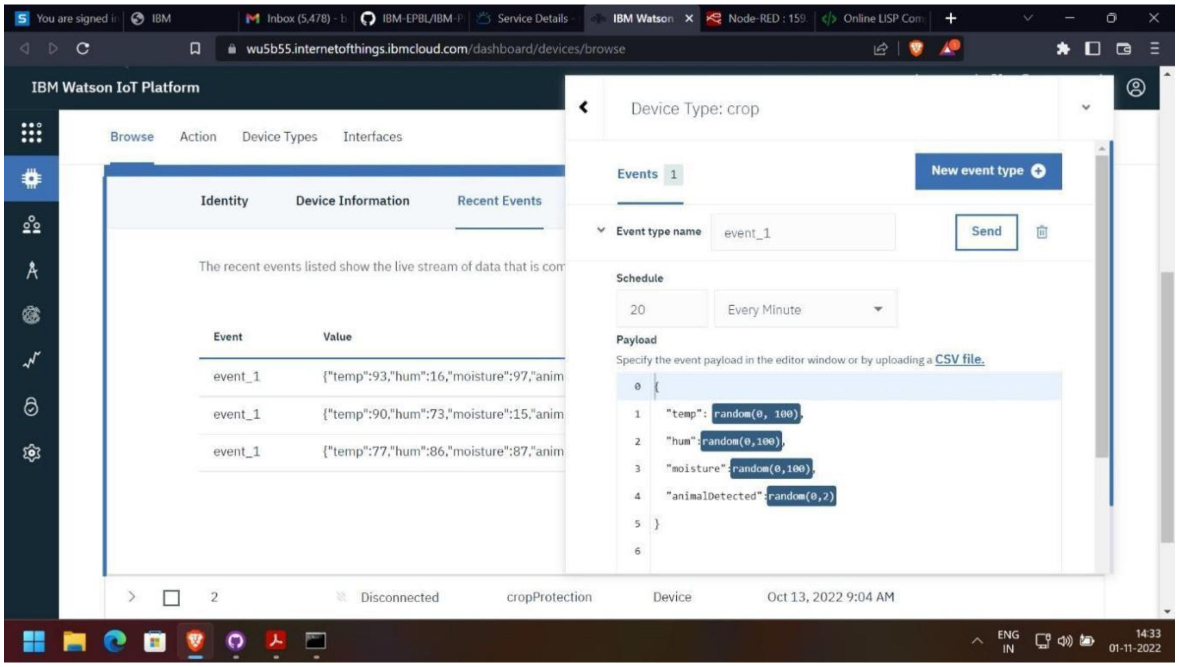
DATA GENERATION IOT PLATFORM:

Source code is deployed on IBM Watson IoT platform to generate sensor data.

Source Code:

```
{
        "temperature": random(0, 100),
        "humidity": random(0, 100),
        "moisture": random(0, 100),
        “animalDetected”:random(0,2)
}
```

Output:



wu5b55.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

Device Type: crop

The recent events listed show the live stream of data that is com

Event	Value
event_1	{"temp":93,"hum":16,"moisture":97,"anim
event_1	{"temp":90,"hum":73,"moisture":15,"anim
event_1	{"temp":77,"hum":86,"moisture":87,"anim

Events 1

Event type name: event_1

Schedule: 20 Every Minute

Payload

Specify the event payload in the editor window or by uploading a CSV file.

```
{
  "temp": random(0, 100),
  "hum":random(0,100),
  "moisture":random(0,100),
  "animalDetected":random(0,2)
}
```

2 Disconnected cropProtection Device Oct 13, 2022 9:04 AM

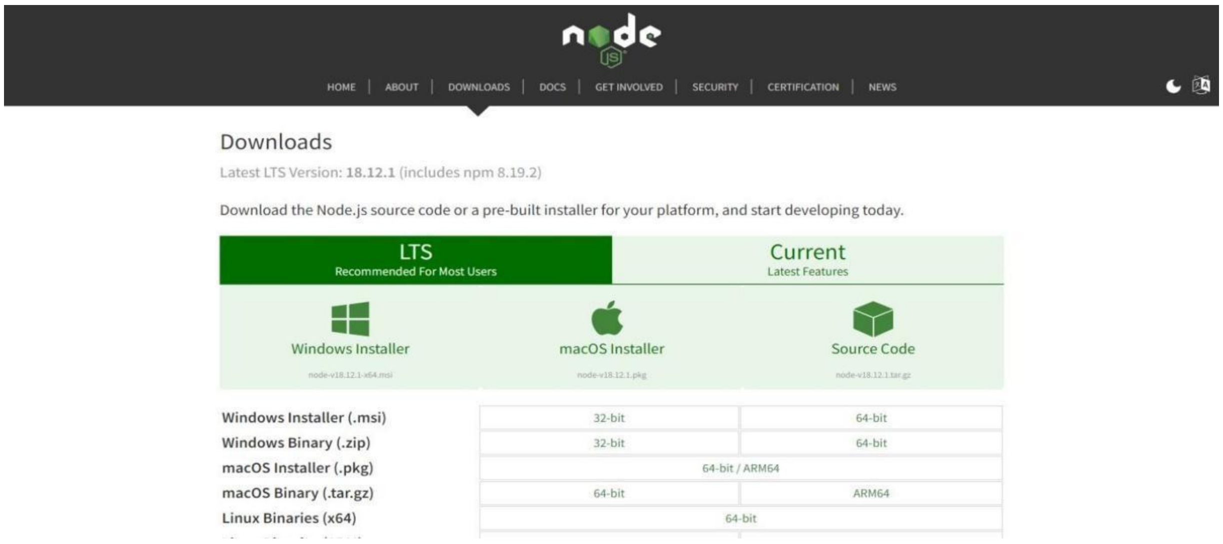
PYTHON CODE TO IBM:

```
import time
import sys

import ibmiotf.application
import ibmiotf.device import
random



#Provide your IBM Watson Device Credentials
organization = "wu5b55"

deviceType = "crop1"
deviceId = "1234"
authMethod = "token"
authToken = "1234567890"


# Initialize GPIO



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 thecloud as
an event of type "greeting" 10 times

deviceCli.connect()
```

```
while True:

            #Get Sensor Data from DHT11


            temp=random.randint(0,100)
            Hum=random.randint(0,100)
            moisture=random.randint(0,100)


            data = { 'temperature' : temp, 'Humidity': Hum,
'Moisture':moisture }
#print data
            def myOnPublishCallback():
                  print ("Temperature = " + str(temp)+" C Humidity = " +
str(hum)+ " moisture = " + str(moisture) + "to IBM Watson")


            success = deviceCli.publishEvent("IoTSensor", "json", data,qos=0,
on_publish=myOnPublishCallback)
            if not success:
                  print("Not connected to IoTF")
            time.sleep(10)


            deviceCli.commandCallback = myCommandCallback


# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

NODE-JS CONNECTION:

STEP1: Download and Install NODE JS.


HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Downloads

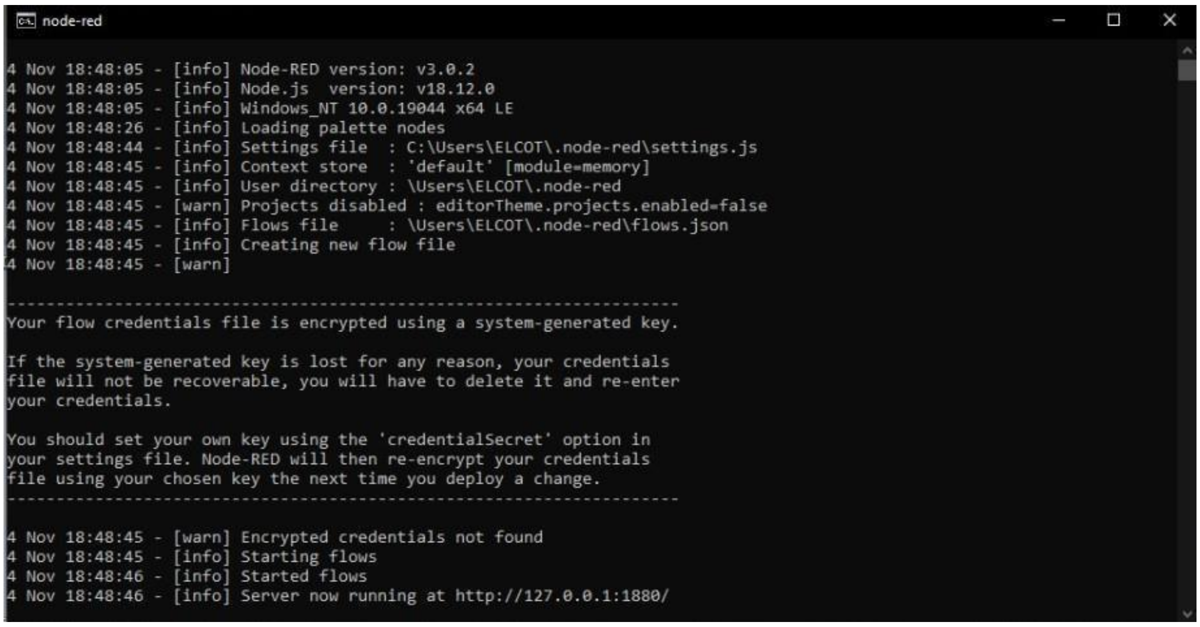
Latest LTS Version: 18.12.1 (includes npm 8.19.2)

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

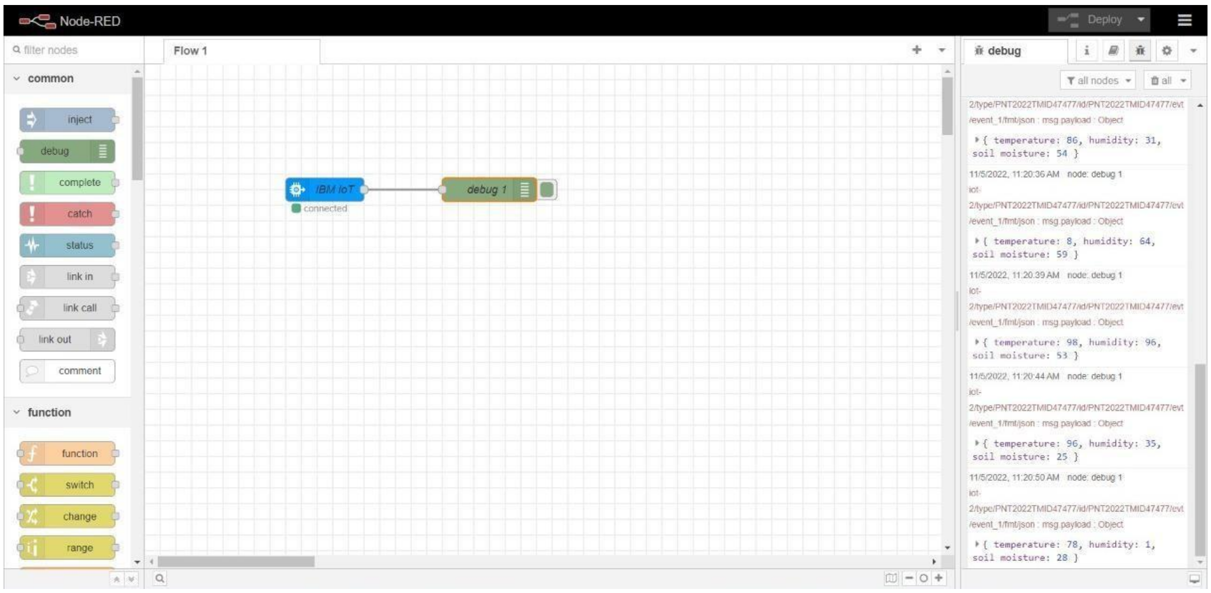
Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	

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



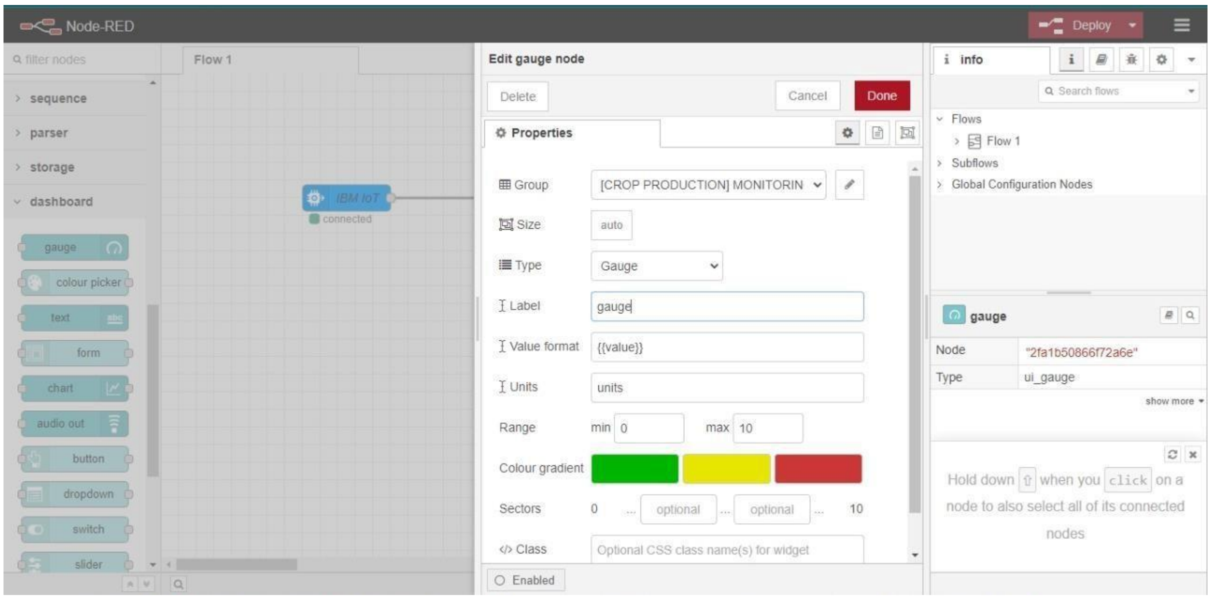
```
4 Nov 18:48:05 - [info] Node-RED version: v3.0.2
4 Nov 18:48:05 - [info] Node.js  version: v18.12.0
4 Nov 18:48:05 - [info] Windows_NT 10.0.19044 x64 LE
4 Nov 18:48:26 - [info] Loading palette nodes
4 Nov 18:48:44 - [info] Settings file  : C:\Users\ELCOT\.node-red\settings.js
4 Nov 18:48:45 - [info] 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 .



```
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
 ▸{ temperature: 86, humidity: 31,
soil moisture: 54 }
11/5/2022, 11:20:36 AM   node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
 ▸{ temperature: 8, humidity: 64,
soil moisture: 59 }
11/5/2022, 11:20:39 AM   node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
 ▸{ temperature: 98, humidity: 96,
soil moisture: 53 }
11/5/2022, 11:20:44 AM   node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
 ▸{ temperature: 96, humidity: 35,
soil moisture: 25 }
11/5/2022, 11:20:50 AM   node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
 ▸{ temperature: 78, humidity: 1,
soil moisture: 28 }
```

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

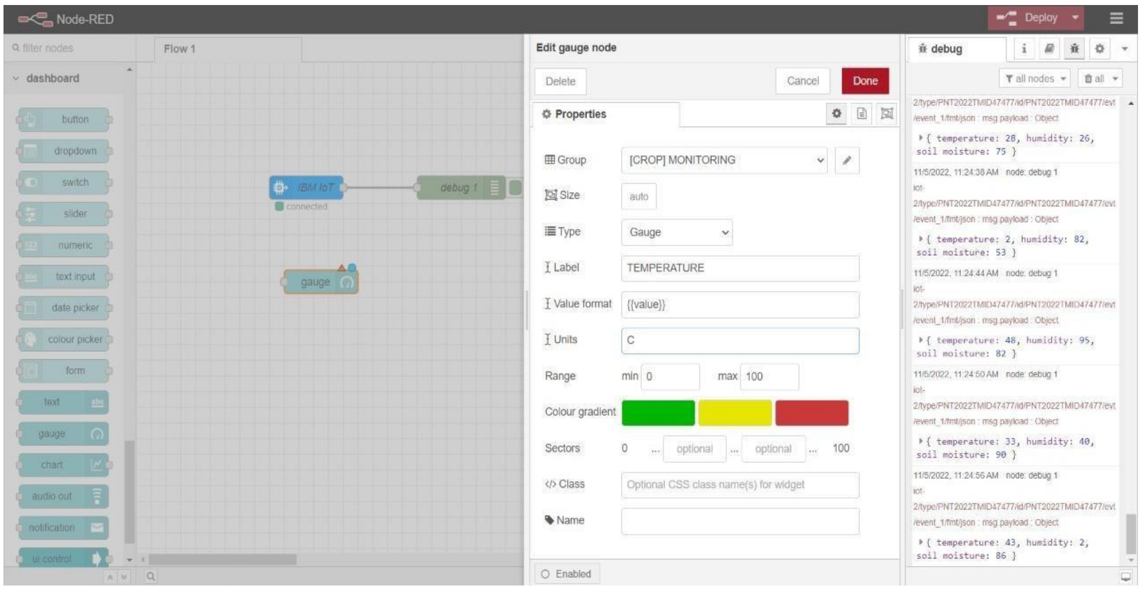


Group: [CROP PRODUCTION] MONITORIN
Size: auto
Type: Gauge
Label: gauge
Value format: {{value}}
Units: units
Range: min 0 max 10
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 10
Class: Optional CSS class name(s) for widget

gauge

Node	"2fa1b50866f72a6e"
Type	ui_gauge

Hold down ⇧ when you click on a node to also select all of its connected nodes



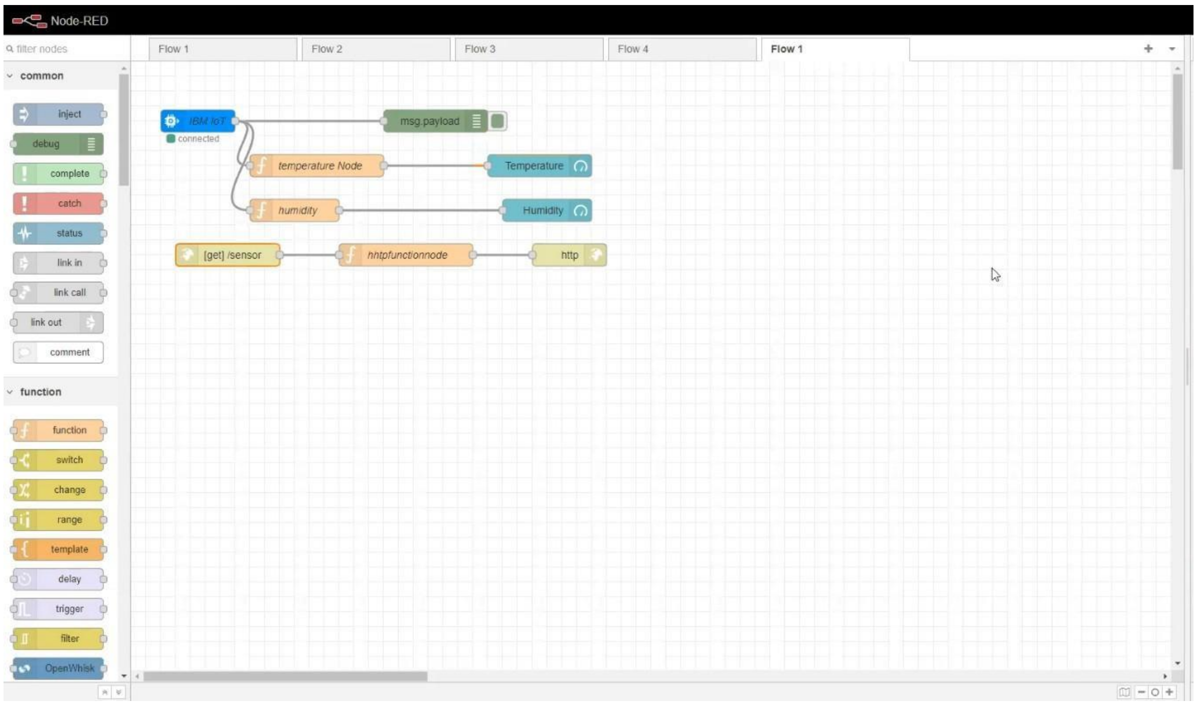
Edit gauge node

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:

```
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 28, humidity: 26,
soil moisture: 75 }
11/5/2022, 11:24:38 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 2, humidity: 82,
soil moisture: 53 }
11/5/2022, 11:24:44 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 48, humidity: 95,
soil moisture: 82 }
11/5/2022, 11:24:50 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 33, humidity: 40,
soil moisture: 90 }
11/5/2022, 11:24:56 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 43, humidity: 2,
soil moisture: 86 }
```

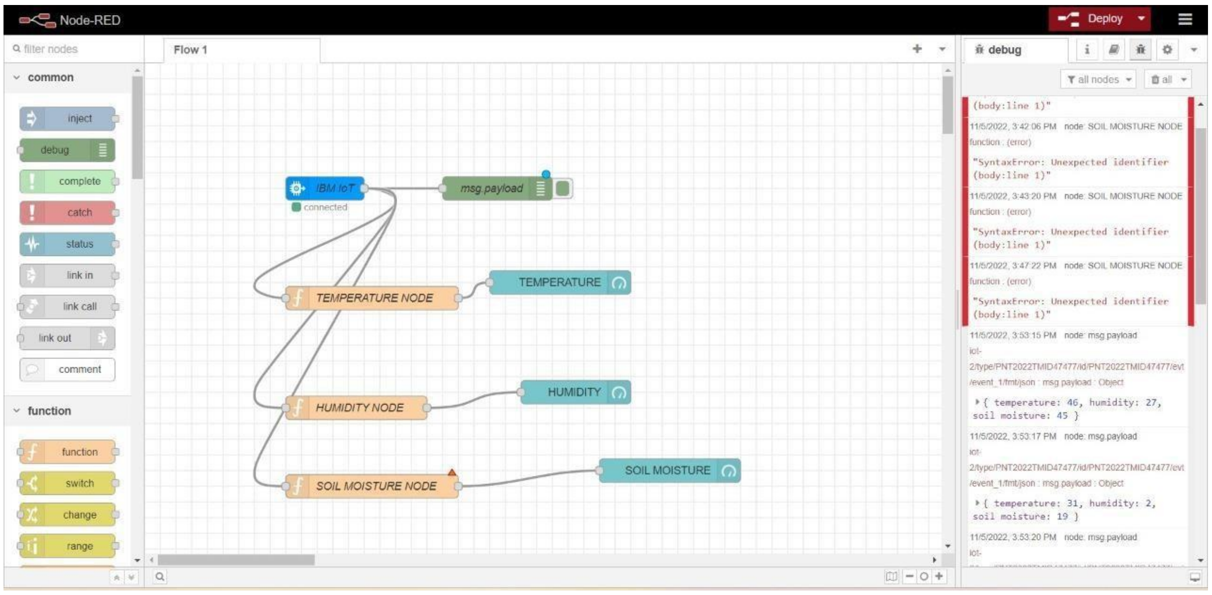
SIMULATION:

STEP1: Simulated program to get the random values



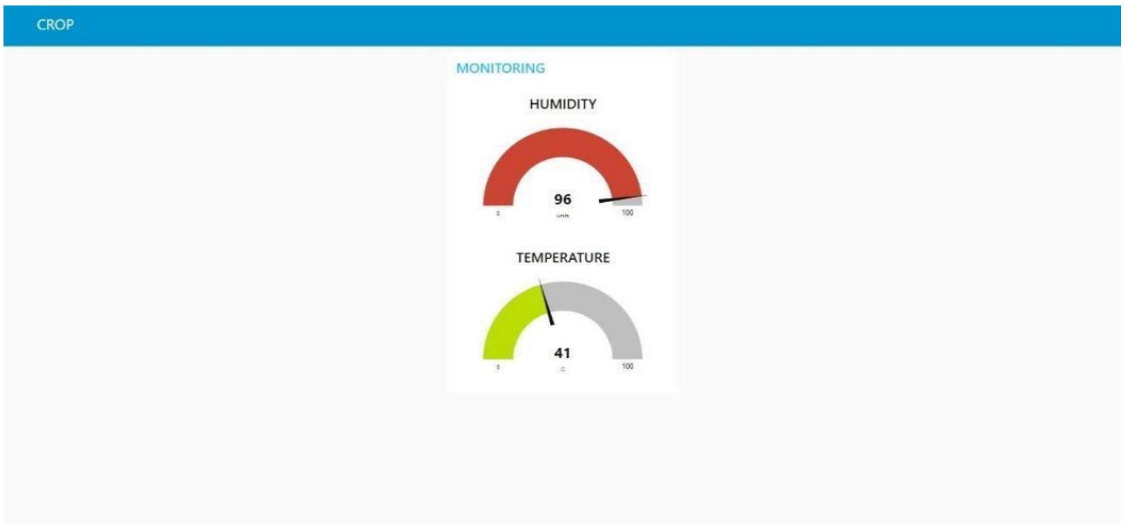
STEP2:

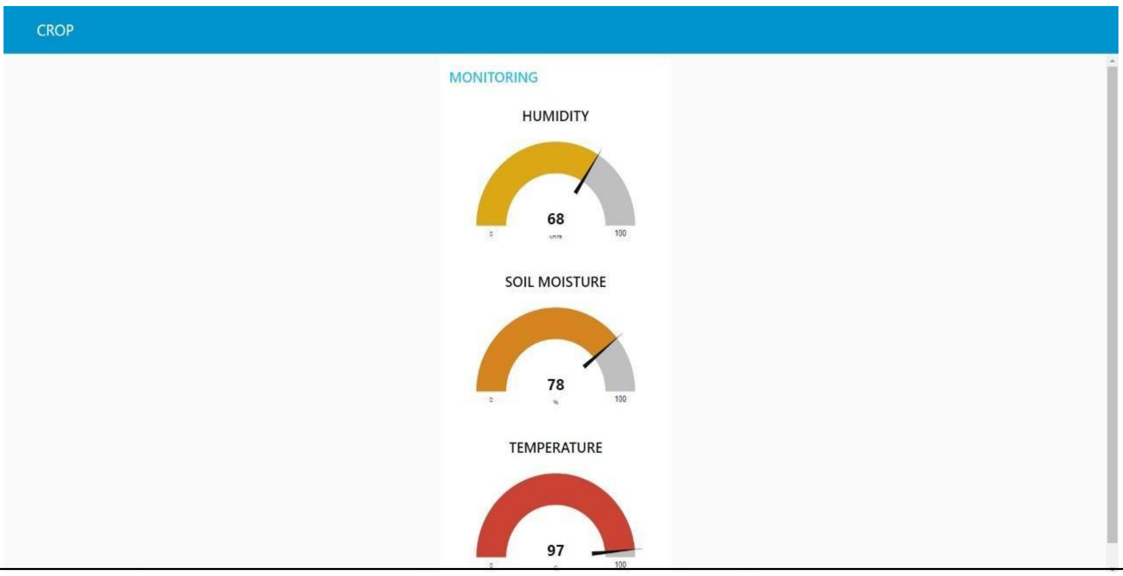
Generate debug message from IBM Watson IoT Platform and connect the nodes.



```
(body:line 1)"
11/5/2022, 3:42:06 PM   node: SOIL MOISTURE NODE
function : (error)
"SyntaxError: Unexpected identifier
(body:line 1)"
11/5/2022, 3:43:20 PM   node: SOIL MOISTURE NODE
function : (error)
"SyntaxError: Unexpected identifier
(body:line 1)"
11/5/2022, 3:47:22 PM   node: SOIL MOISTURE NODE
function : (error)
"SyntaxError: Unexpected identifier
(body:line 1)"
11/5/2022, 3:53:15 PM   node: msg.payload
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 46, humidity: 27,
soil moisture: 45 }
11/5/2022, 3:53:17 PM   node: msg.payload
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 31, humidity: 2,
soil moisture: 19 }
11/5/2022, 3:53:20 PM   node: msg.payload
iot-
```

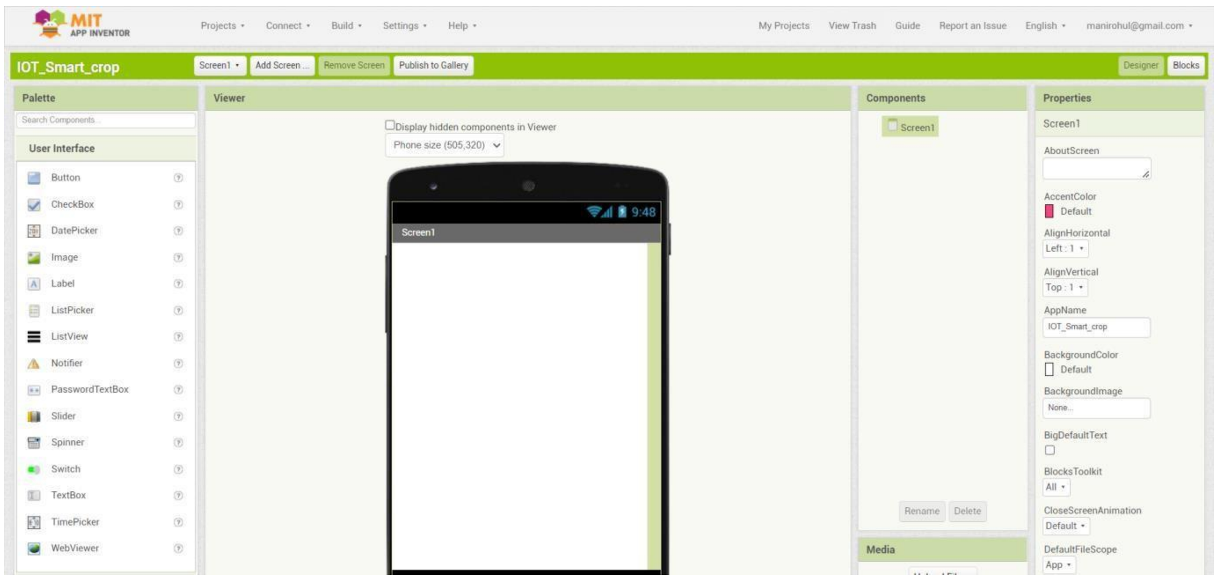
STEP3: Generate the some output from recent events.





MIT APP INVENTOR:

STEP 1: MIT APP inventor to design the APP.



Screen1

AboutScreen

AccentColor: Default

AlignHorizontal: Left : 1

AlignVertical: Top : 1

AppName: IOT_Smart_crop

BackgroundColor: Default

BackgroundImage: None...

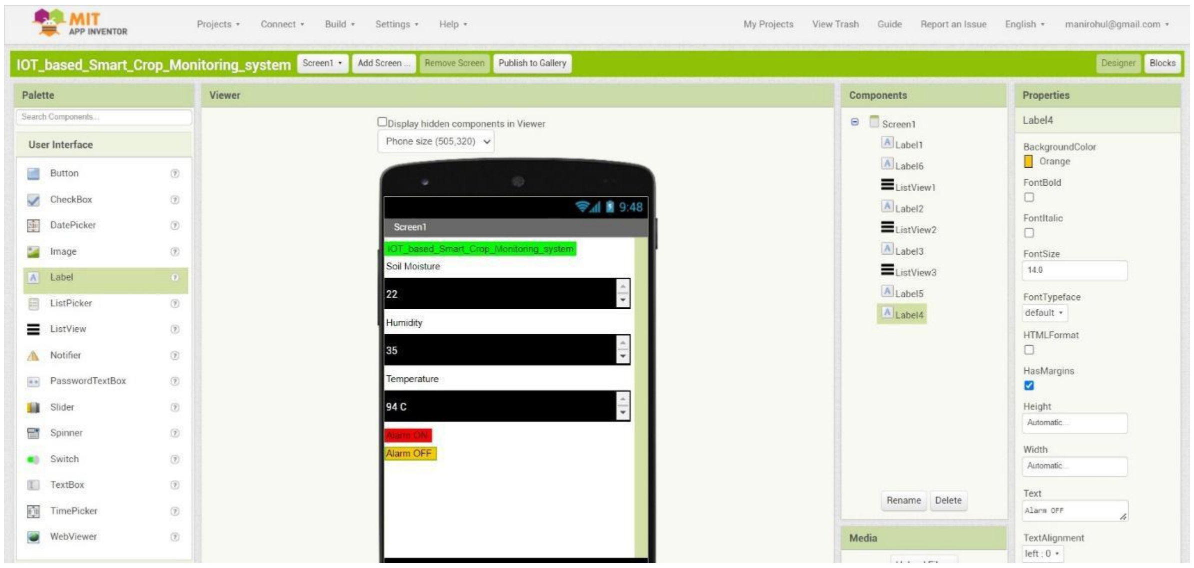
BigDefaultText: ☐

BlocksToolkit: All

CloseScreenAnimation: Default

DefaultFileScope: App

STEP 2: Customize the App interface to Display the Values.



Label4

BackgroundColor: Orange

FontBold: ☐

FontItalic: ☐

FontSize: 14.0

FontTypeface: default

HTMLFormat: ☐

HasMargins: ☑

Height: Automatic...

Width: Automatic...

Text: Alarm OFF

TextAlignment: left : 0

ADVANTAGES:

- Farmers can monitor the health of farm animals closely, even if they are physically distant.
- Smart farming systems reduce waste, improve productivity and enable management of a greater number of resources through remote sensing.
- High reliance.
- Enhanced Security.

DISADVANTAGES:

- Farms are located in remote areas and are far from access to the internet.
- A farmer needs to have access to crop data reliably at any time from any location, so connection issues would cause an advanced monitoring system to be useless.
- High Cost
- Equipment needed to implement IoT in agriculture is expensive.

APPLICATIONS:

- Monitoring the crop field with the help of sensors (light, humidity, temperature, soil moisture, etc.)
- Automating the irrigation system
- Soil Moisture Monitoring (including conductivity and pH)

CONCLUSION:

AS a result of this system, we can detect the changes in the field easily and intimate the farmers about it and also we can take precautions and do remedies accordingly. Here we use very low power consuming highly efficient components that give us accurate results and also they perform at low data rate conditions without any lag and help in finding the remedies. This crop protection system helps in detection of all kinds of external dangers and it saves time and money to the farmers before any loss that may occur. With the help of this system the farmers can be in a peaceful environment at ease without any pressure.

GITHUB LINK: IBM-Project-10070-1659090292/LITERATURE SURYVEY.docx at main · IBM-EPBL/IBM-Project-10070-1659090292 (github.com)

DEMO VIDEO: