

SENDING DATA FROM RASPBERRY-PI TO IBM WATSON

Date	3 NOVEMBER 2022
Team ID	PNT2022TMID39560
Project Name	GAS LEAKAGE MONITORING AND ALERTING SYSTEM FOR INDUSTRIES

AIM:

To send sensor data (or any dummy data) from Raspberry –Pi to IBM Watson .In our case it is DHT sensors Data.

REQUIREMENTS:

HARDWARE:

- RASPBERRY-PI (3B)(WITH ETHERNET CABLE OR WIFI CONNECTED)
- USB MOUSE
- USB KEYBOARD
- VGA TO HDMI CABLE
- A MONITOR
- RASPBERRY’S POWER SUPPLY
- DHT-11 Sensor
- Connecting Wires

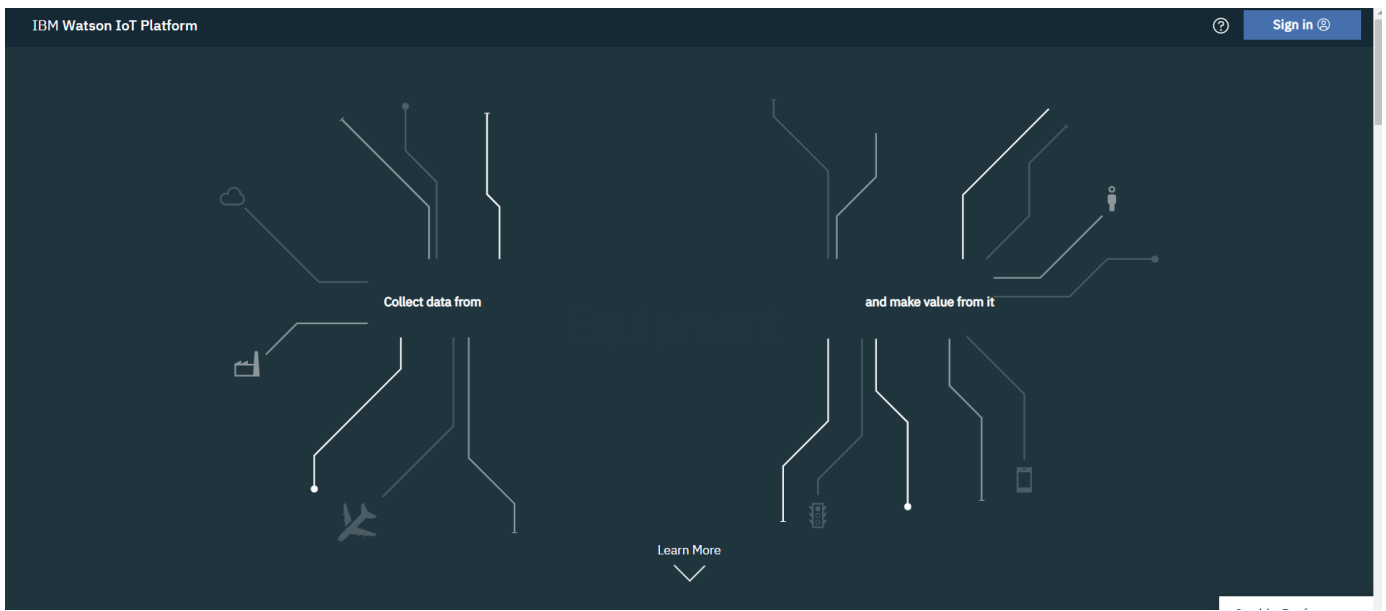
SOFTWARE:

- IBM BLUEMIX ACCOUNT

STEPS TO BE FOLLOWED:

Step-1: Create a device in IBM Watson:

- Firstly, login into your IBM-Bluemix account with your e-mail ID and Password.



IBM

Log in to IBM

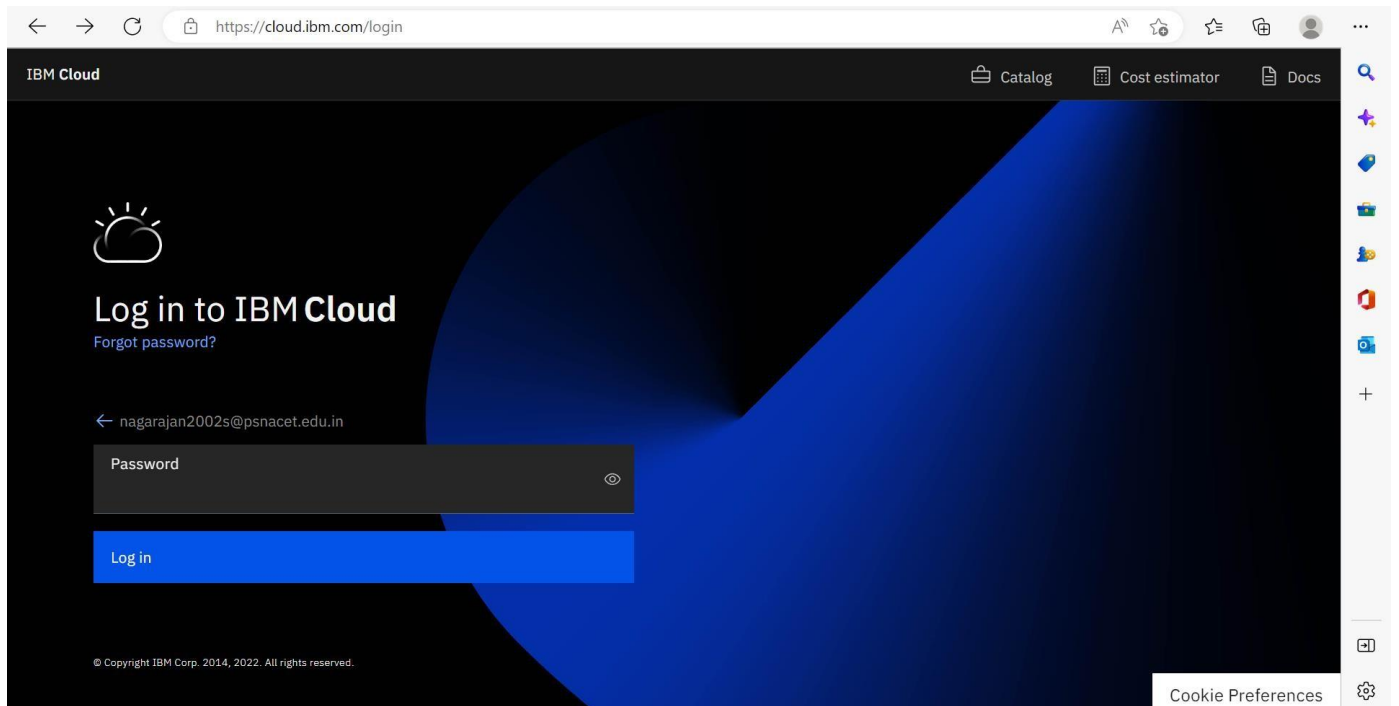
IBMid [Forgot IBMId?](#)

☒ Remember me ⓘ

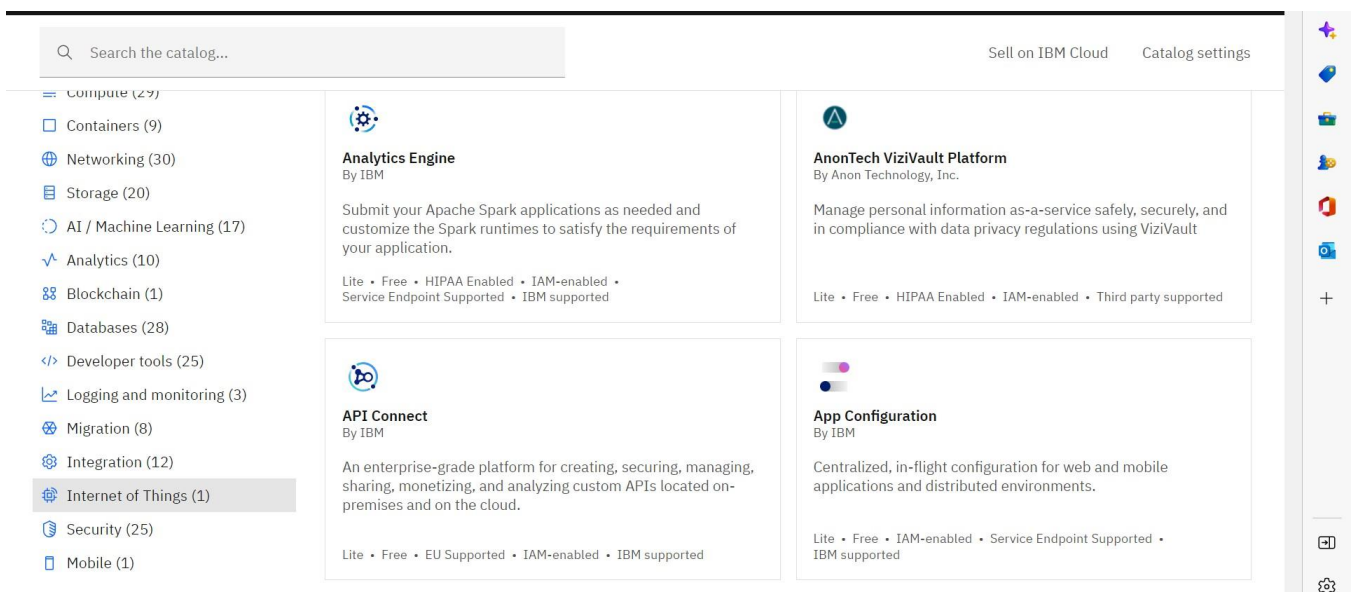
[Continue](#) →

Don't have an account? [Create an IBMId](#)

Need help? [Contact the IBMId help desk](#)




➤ Click on catalog on your dashboard screen, then under platform go IoT.



➤ Check all details and click on create.

➤ click on Launch



Catalog /

Internet of Things Platform

This service is the hub of all things IBM IoT, it is where you can set up and manage your connected devices so that your apps can access their live and historical data.

CreateAbout

Type
Service

Provider
IBM

Last updated
08/15/2022

Category
Internet of Things

Compliance
IAM-enabled

Location
Frankfurt

Select a location

Frankfurt (eu-de)

Select a pricing plan

Displayed prices do not include tax. Monthly prices shown are for country or location: [United States](#)

Plan	Features	Pricing
Lite	Includes up to 500 registered devices, and a maximum of 200 MB of each data metric Maximum of 500 registered devices	Free

Summary

Internet of Things Platform **Free**

Location: Frankfurt

Plan: Lite

Service name: Internet of Things Platform-0g

Resource group: Default

☒ I have read and agree to the following license agreements:
[Terms](#)

Create

Add to estimate

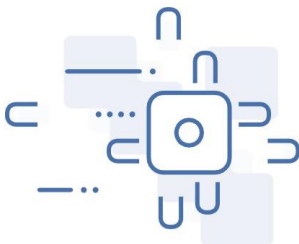
Resource list /

Internet of Things Platform-0g

Active [Add tags](#)

DetailsActions...

ManagePlanConnections



Let's get started with IBM Watson IoT Platform

Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.

LaunchDocs

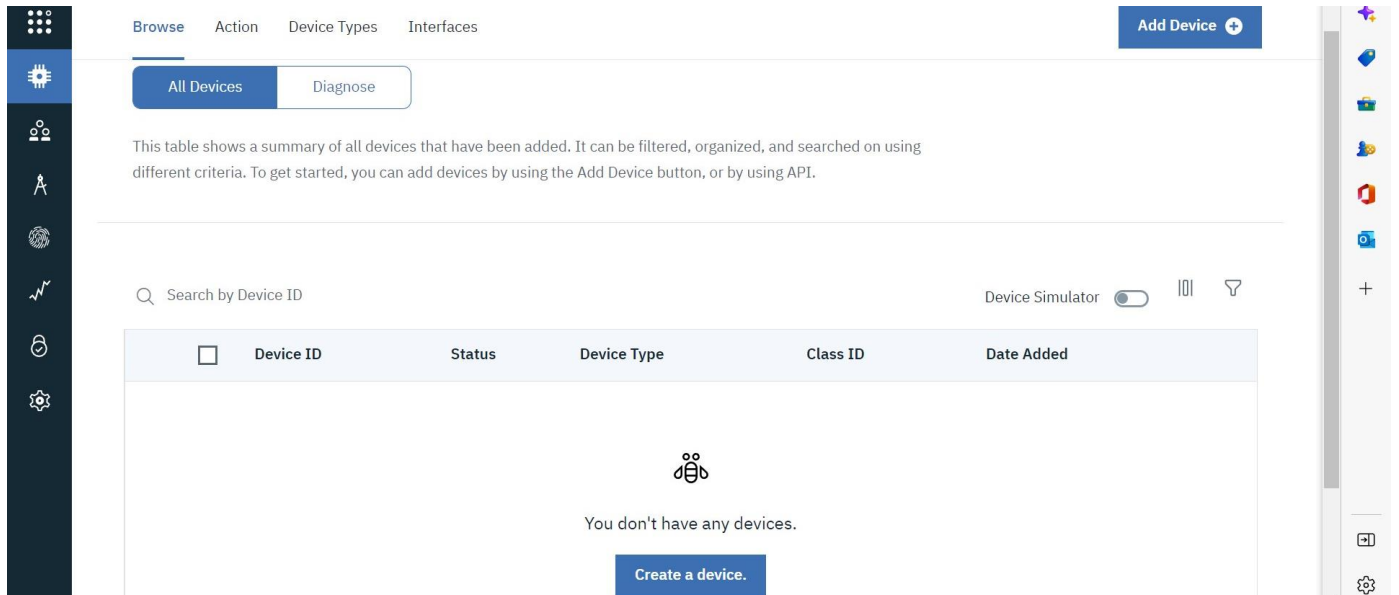
Ready for the next level?

IBM Watson IoT Platform Journey

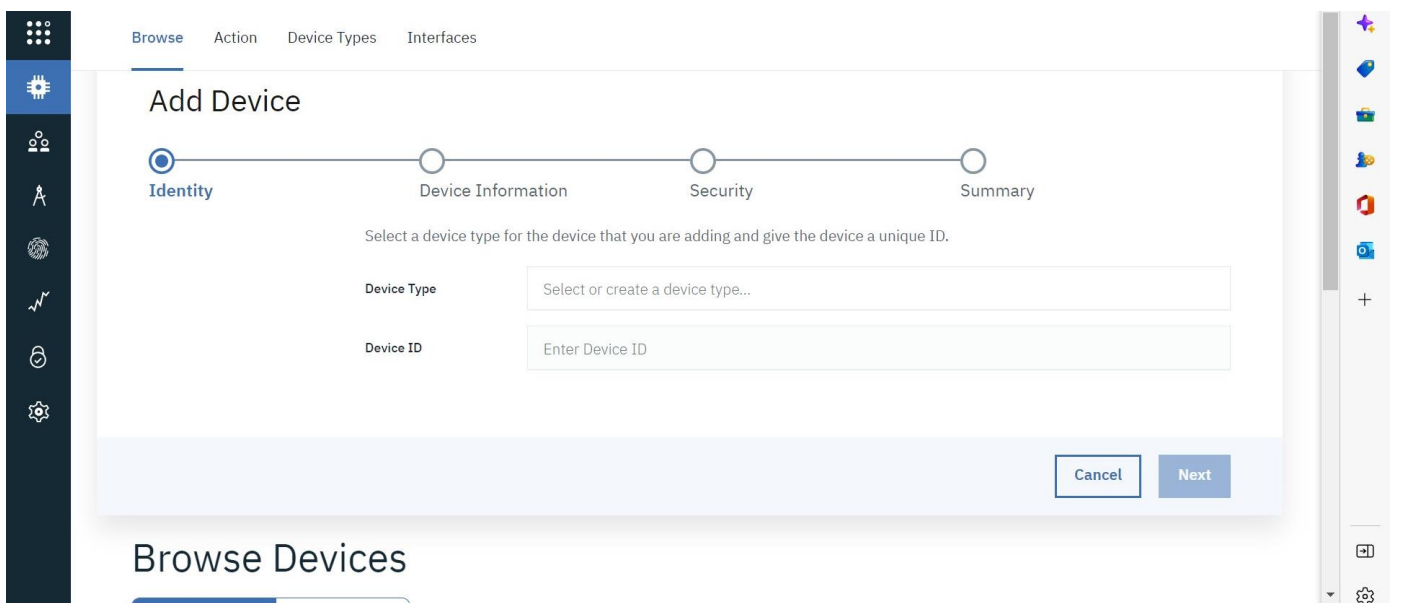
☒ Lite

☐ Non-Production

- Dashboard of IBM Watson IoT platform,
- Click on Add device



- After click on Add device this page will open



- Go to device type and fill the details.

Browse Action **Device Types** Interfaces

Add type

Identity Device Information

Device types group devices that have similar characteristics, such as model number, firmware version, or location. Give the device type a unique name and a description that identifies characteristics that are shared by devices of this type.

Type Or

Name

The device type name is used to identify the device type uniquely and uses a restricted set of characters to make it suitable for API use.

Description

Cancel Next

- Click on Finish

Browse Action **Device Types** Interfaces

Add type

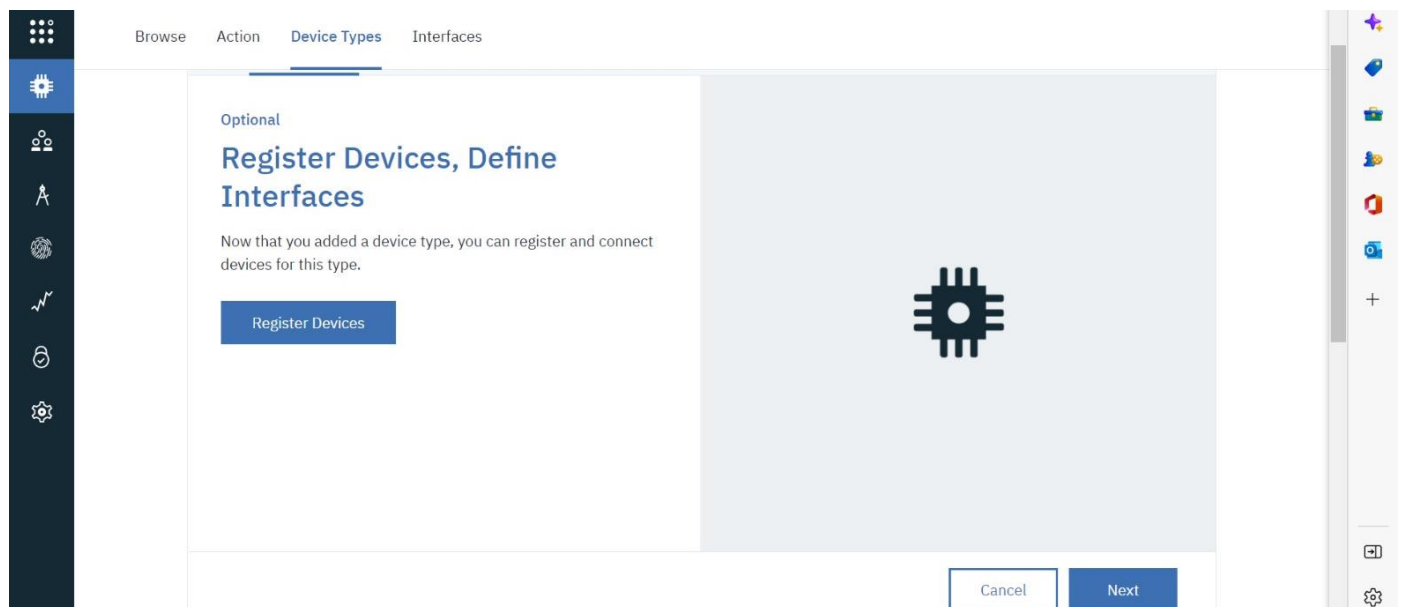
Identity Device Information

These attributes will be used as a template for new devices that are assigned this device type [Edit Metadata](#)

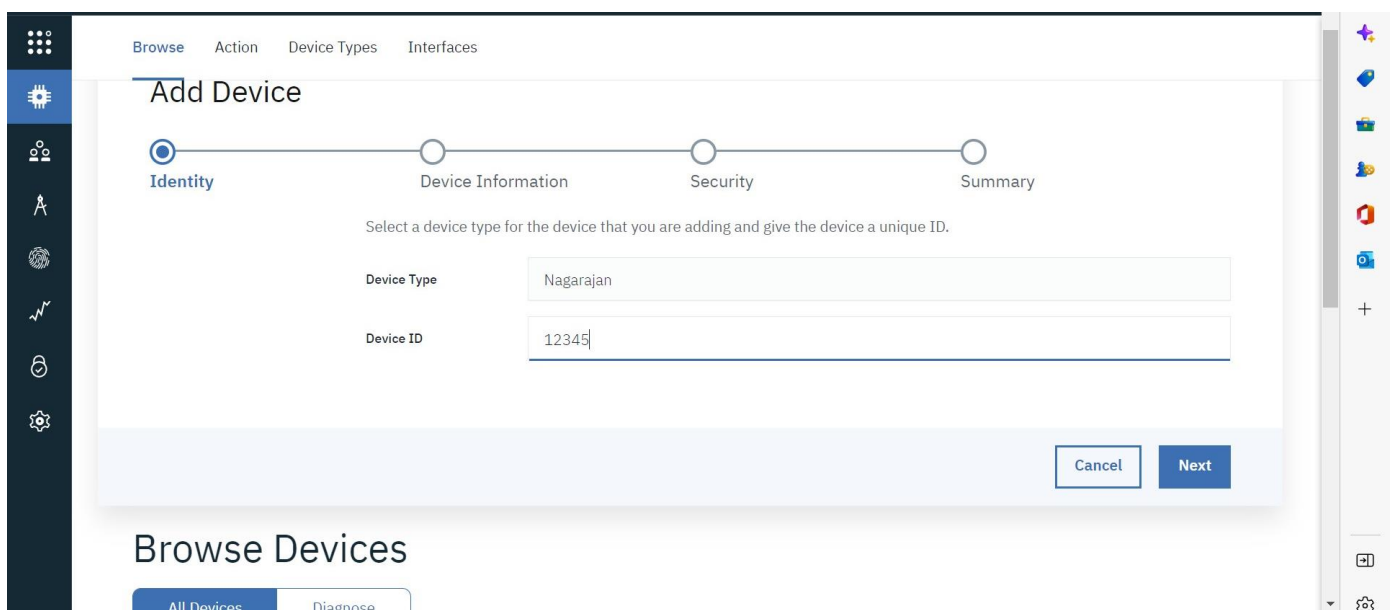
Serial Number	<input type="text" value="Enter Serial Number"/>	Manufacturer	<input type="text" value="Enter Manufacturer"/>
Model	<input type="text" value="Enter Model"/>	Device Class	<input type="text" value="Enter Device Class"/>
Description	<input type="text" value="Enter Description"/>	Firmware Version	<input type="text" value="Enter Firmware Version"/>
Hardware Version	<input type="text" value="Enter Hardware Version"/>	Descriptive Location	<input type="text" value="Enter Descriptive Location"/>

Back Finish

- Click on Register Device.



- Choose the device and give device ID and then click on next.



➤ Click on Next

The screenshot shows the 'Add Device' form in a web application. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. The left sidebar contains various icons for navigation. The main content area is titled 'Add Device' and features a progress bar with four steps: 'Identity' (checked), 'Device Information' (active), 'Security', and 'Summary'. Below the progress bar, a message states: 'You can modify the default device information and enter more information about the device for identification purposes.' The form contains several input fields: 'Serial Number', 'Model', 'Description', 'Hardware Version', 'Manufacturer', 'Device Class', 'Firmware Version', and 'Descriptive Location'. Each field has a placeholder text 'Enter [field name]'. At the bottom left of the form, there is a button labeled 'Add Metadata' with a plus icon.

➤ Click on Next

The screenshot shows the 'Add Device' form in a web application, specifically the 'Security' step. The top navigation bar and left sidebar are consistent with the previous screenshot. The progress bar shows 'Identity' and 'Device Information' as completed steps, 'Security' as the active step, and 'Summary' as the next step. The main content area is titled 'Security' and contains two columns of text. The left column is titled 'Auto-generated authentication token (default)' and explains that the service will generate an 18-character alphanumeric token. The right column is titled 'Self-provided authentication token' and explains that the user must provide their own token, which must be between 8 and 36 characters and contain a mix of lowercase and uppercase letters, numbers, and symbols. Below the text, there is an input field labeled 'Authentication Token' with a placeholder 'Enter an optional token' and an information icon. At the bottom, a note states: 'Make a note of the generated token. Lost authentication tokens cannot be recovered. Tokens are encrypted before being stored.' and 'Authentication token are encrypted before we store them.'

➤ Click on Finish

Browse Action Device Types Interfaces

Add Device

Identity Device Information Security Summary

Verify that the following information is correct then select Finish

Device Type
Nagarajan

Device ID
12345

[View Metadata](#)

Security Token
To be generated

Back Finish

➤ Device is created

Browse Action Device Types Interfaces

Add Device +

Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device Simulator ☒

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
> <input type="checkbox"/>	12345	Disconnected	Nagarajan	Device	Oct 31, 2022 11:38 AM	

Items per page 50 | 1-1 of 1 item

1 of 1 page

1 Simulation running

Activate Windows
Go to Settings to activate Windows.

STEP-2: INSTALLING NECESSARY PACKAGES ON YOUR PI:

- Now we are going to install necessary packages on your pi.
- Open your terminal in your pi and type the following commands
- `curl -LO https://github.com/ibm-messaging/iot-raspberrypi/releases/download/1.0.2.1/iot_1.0-2_armhf.deb`
- `sudo dpkg -i iot_1.0-2_armhf.deb`
- `service iot status`

Following are the images as to what appears on your pi's terminal when u type these commands

```
File Edit Tabs Help
--2017-10-23 06:55:22-- http://ftp.nl.debian.org/debian/pool/main/o/openssl/libssl1.0.0_1.0.1t-1-deb8u6_armhf.deb
Resolving ftp.nl.debian.org (ftp.nl.debian.org)... 130.89.149.21, 2001:67c:2564:a120::21
Connecting to ftp.nl.debian.org (ftp.nl.debian.org)[130.89.149.21]:80... connect
ed.
HTTP request sent, awaiting response... 200 OK
Length: 867950 (848K) [application/x-debian-package]
Saving to: 'libssl1.0.0_1.0.1t-1-deb8u6_armhf.deb'

libssl1.0.0_1.0.1t- 100%[=====] 847.61K  358KB/s   in 2.4s

2017-10-23 06:55:25 (358 KB/s) - 'libssl1.0.0_1.0.1t-1-deb8u6_armhf.deb' saved [
867950/867950]

pi@raspberrypi:~$ sudo dpkg -i libssl1.0.0_1.0.1t-1-deb8u6_armhf.deb
Selecting previously unselected package libssl1.0.0:armhf.
(Reading database ... 115606 files and directories currently installed.)
Preparing to unpack libssl1.0.0_1.0.1t-1-deb8u6_armhf.deb ...
Unpacking libssl1.0.0:armhf (1.0.1t-1-deb8u6) ...
Setting up libssl1.0.0:armhf (1.0.1t-1-deb8u6) ...
pi@raspberrypi:~$ curl -LO https://github.com/ibm-messaging/iot-raspberrypi/rel
eases/download/1.0.2.1/iot_1.0-2_armhf.deb
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           0      0     0    0             0      0      0      0
100 164  0 164  0  0 157  0  0:00:01  0:00:01  0:00:00  157
100 609  0 609  0  0 457  0  0:00:01  0:00:01  0:00:00  457
100 110k 100 110k  0  0 29117  0  0:00:03  0:00:03  0:00:00 48190
pi@raspberrypi:~$ sudo dpkg -i iot_1.0-2_armhf.deb
(Reading database ... 115626 files and directories currently installed.)
Preparing to unpack iot_1.0-2_armhf.deb ...
Unpacking iot (1.0-1) over (1.0-1) ...
Setting up iot (1.0-1) ...
Processing triggers for systemd (232-25+deb9u1) ...
pi@raspberrypi:~$ service iot status
* iot.service - LSB: iot service
   Loaded: loaded (/etc/init.d/iot; generated; vendor preset: enabled)
   Active: active (running) since Mon 2017-10-23 06:56:25 UTC; 17s ago
   Docs: man:systemd-sysv-generator(8)
   CGroup: /system.slice/iot.service
           └─562 /opt/iot/iot /dev/null

Oct 23 06:56:24 raspberrypi systemd[1]: Starting LSB: iot service...
Oct 23 06:56:24 raspberrypi iot[2567]: Starting the iot program
Oct 23 06:56:25 raspberrypi iot[2562]: **** IoT Raspberry Pi Sample has started ****
Oct 23 06:56:25 raspberrypi iot[2562]: Config file not found. Going to Quickstart mode
Oct 23 06:56:25 raspberrypi iot[2562]: Running in Quickstart mode
Oct 23 06:56:25 raspberrypi systemd[1]: Started LSB: iot service
```

- Then open your terminal and type `pip install ibmiotf`

```
File Edit Tabs Help
pi@raspberrypi:~$ pip install ibmiotf
Collecting ibmiotf
  Downloading ibmiotf-0.3.0.tar.gz (59kB)
    100% |#####| 61kB 510kB/s
Collecting dicttoxml>=1.7.4 (from ibmiotf)
  Downloading dicttoxml-1.7.4.tar.gz
Collecting iso8601>=0.1.10 (from ibmiotf)
  Downloading iso8601-0.1.12-py2.py3-none-any.whl
Collecting paho-mqtt>=1.2 (from ibmiotf)
  Downloading paho-mqtt-1.3.1.tar.gz (80kB)
    100% |#####| 81kB 916kB/s
Collecting pytz>=2014.7 (from ibmiotf)
  Using cached pytz-2017.2-py2.py3-none-any.whl
Collecting requests>=2.5.0 (from ibmiotf)
  Downloading requests-2.18.4-py2.py3-none-any.whl (88kB)
    100% |#####| 92kB 1.6MB/s
Collecting requests-toolbelt>=0.7.0 (from ibmiotf)
  Downloading requests-toolbelt-0.8.0-py2.py3-none-any.whl (54kB)
    100% |#####| 61kB 1.6MB/s
Collecting xmldict>=0.18.2 (from ibmiotf)
  Downloading xmldict-0.11.0-py2.py3-none-any.whl
Collecting urllib3<1.23,>=1.21.1 (from requests>=2.5.0->ibmiotf)
  Downloading urllib3-1.22-py2.py3-none-any.whl (132kB)
    100% |#####| 133kB 1.4MB/s
Collecting idna<2.7,>=2.5 (from requests>=2.5.0->ibmiotf)
  Downloading idna-2.6-py2.py3-none-any.whl (56kB)
    100% |#####| 61kB 1.7MB/s
Collecting chardet<3.1.0,>=3.0.2 (from requests>=2.5.0->ibmiotf)
  Downloading chardet-3.0.4-py2.py3-none-any.whl (133kB)
    100% |#####| 143kB 1.9MB/s
Collecting certifi>=2017.4.17 (from requests>=2.5.0->ibmiotf)
  Using cached certifi-2017.7.27-py2.py3-none-any.whl
Building wheels for collected packages: ibmiotf, dicttoxml, paho-mqtt
Running setup.py bdist_wheel for ibmiotf
  Stored in directory: /home/pi/.cache/pip/wheels/7e/f9/45/bbc33ad957e02f7b71ba80e316d65a83d9d735a0d12e0c0418
Running setup.py bdist_wheel for dicttoxml
  Stored in directory: /home/pi/.cache/pip/wheels/45/62/59/96910b33ec6a7b2ae66a13765491b50def5468024078e12cce
Running setup.py bdist_wheel for paho-mqtt
  Stored in directory: /home/pi/.cache/pip/wheels/28/d8/0d/acdc8f289011b7be7de71deebef6642fb3be9313dfff0493
Successfully built ibmiotf dicttoxml paho-mqtt
Installing collected packages: dicttoxml, iso8601, paho-mqtt, pytz, urllib3, idna, chardet, certifi, requests, requests-toolbelt, xmldict, ibmiotf
Successfully installed certifi-2017.7.27.4 chardet-3.0.4 dicttoxml-1.7.4 ibmiotf-0.3.0 idna-2.6 iso8601-0.1.12 paho-mqtt-1.3.1 pytz-2017.2 requests-2.18.4 requests-toolbelt-0.8.0 urllib3-1.22 xmldict-0.11.0
pi@raspberrypi:~$
```

- I have sent DHT-11 Sensors data to ibm bluemix .To get the code u need to login into IOT GYAN.
- Then I get the image as follows in my pi's shell:

```
Python 2.7.13 (default, Jan 19 2017, 14:48:08)
[GCC 6.3.0 20170124] on linux2
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/Downloads/dht11toibmiot.py =====
2017-10-23 07:10:37,768 ibmiotf.device.Client INFO Connected successfully: d:gegtl4:mydevice:mydevice
Published Temperature = 28 C Humidity = 50 % to IBM Watson
SensorData Invalid
Published Temperature = 28 C Humidity = 50 % to IBM Watson
SensorData Invalid
Published Temperature = 28 C Humidity = 50 % to IBM Watson
SensorData Invalid
Published Temperature = 28 C Humidity = 50 % to IBM Watson
Published Temperature = 28 C Humidity = 50 % to IBM Watson
Published Temperature = 29 C Humidity = 50 % to IBM Watson
```

Step-3: checking your data sent on IBM Bluemix:

- After you have sent your sensors data you can check whether it is received at your iot platform Just look at the image below and if u see the same wifi kind of symbol on your created device then your data is being received.

IBM Watson IoT Platform

Search by Device ID

Device Simulator ☒

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
12345	Disconnected	Nagarajan	Device	Oct 31, 2022 11:38 AM	

1 Simulation running

- After double clicking on your created device you can see the received data as shown in image

The screenshot displays the Watson IoT platform interface. At the top, there are tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area shows a table of devices. The selected device, ID 12345, is shown in detail. It is a 'Nagarajan' device, status 'Disconnected', added on 'Oct 31, 2022 11:38 AM'. Below the device details, there is a section for 'Recent Events' showing a live stream of data. The events are listed in a table with columns: Event, Value, Format, and Last Received. The events are all 'event_1' and contain JSON data for 'Hazardous Gas', 'Temperature', and 'Humidity'. The status bar at the bottom indicates '1 Simulation running'.

Event	Value	Format	Last Received
event_1	{"Hazardous Gas":61,"Temperature":88,"Humidit..."}	json	a few seconds ago
event_1	{"Hazardous Gas":20,"Temperature":36,"Humidit..."}	json	a few seconds ago
event_1	{"Hazardous Gas":79,"Temperature":56,"Humidit..."}	json	a few seconds ago
event_1	{"Hazardous Gas":52,"Temperature":82,"Humidit..."}	json	a few seconds ago
event_1	{"Hazardous Gas":26,"Temperature":33,"Humidit..."}	json	a few seconds ago

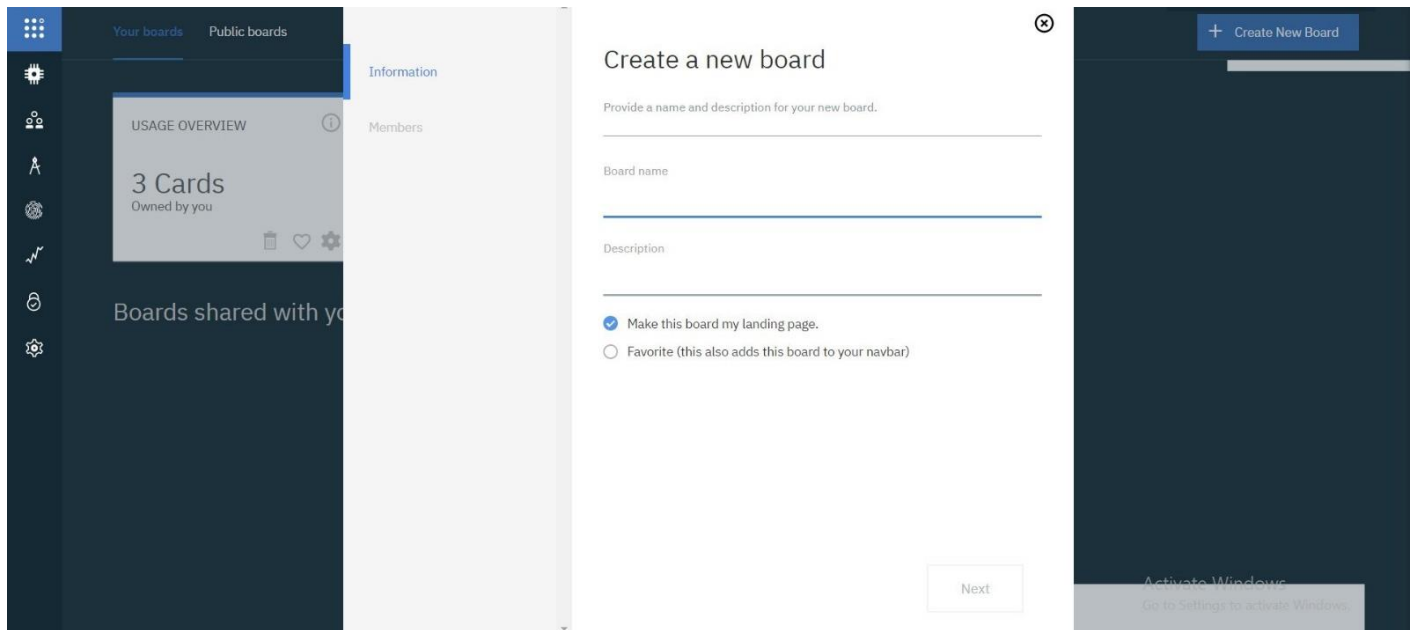
Step-4: Creating boards and cards for visualization of data:

- In your Watson platform you have an option called board .Click on it and you get the following window on your screen

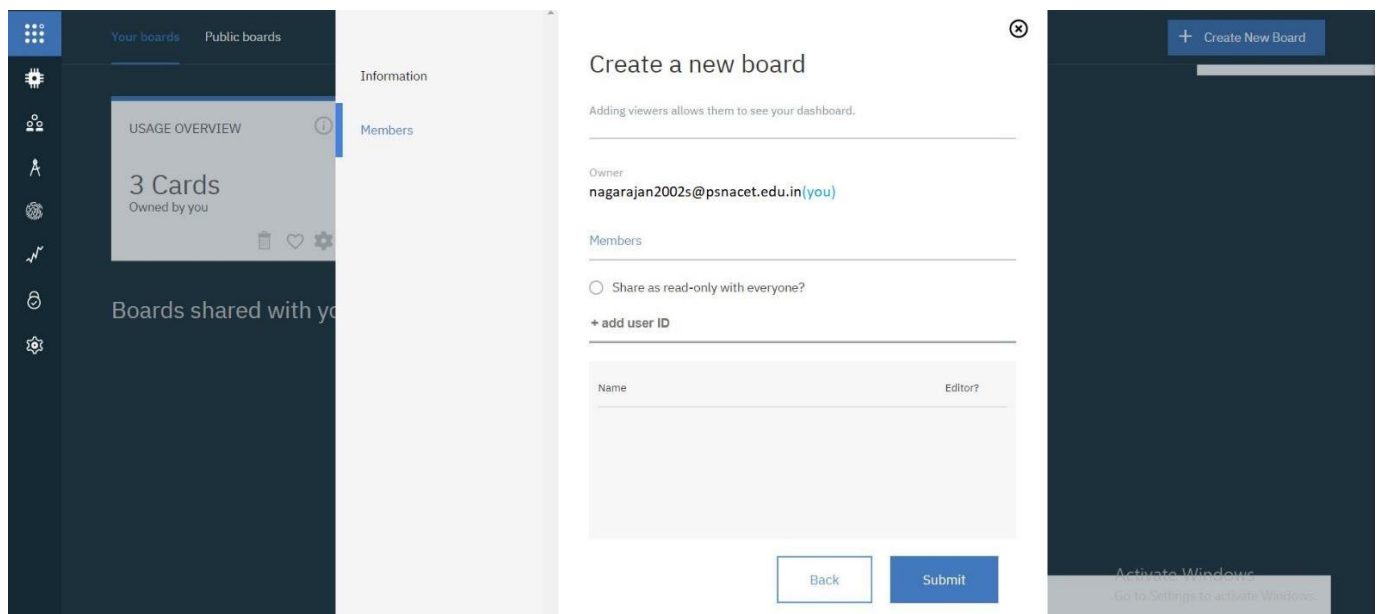
The screenshot shows the 'Your boards' section of the Watson IoT platform. There are two boards displayed: 'USAGE OVERVIEW' with 3 cards and 'RISK AND SECURITY OVERVIEW' with 4 cards. Both boards are owned by the user. A large dashed box with a plus sign indicates where to click to create a new board. The status bar at the bottom indicates '1 Simulation running'.

- Click on Create a new board to create a board .

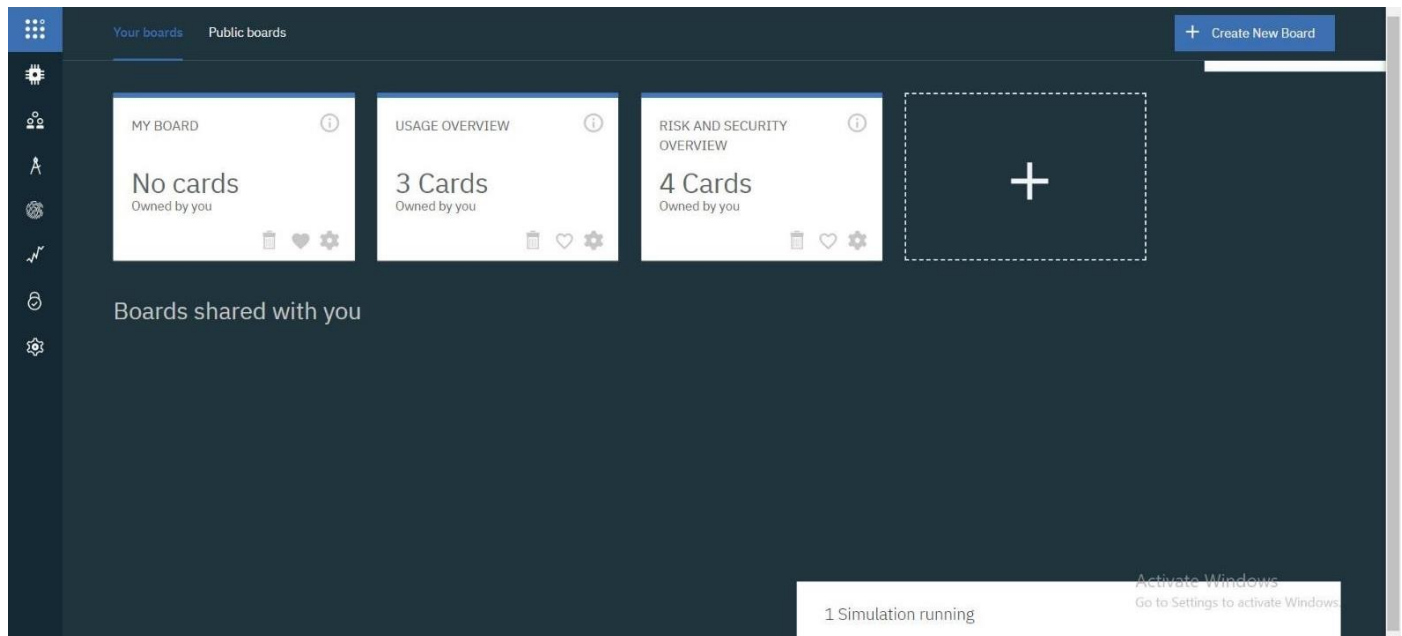
The given below window appears give a name and description to your board as shown in the window below.



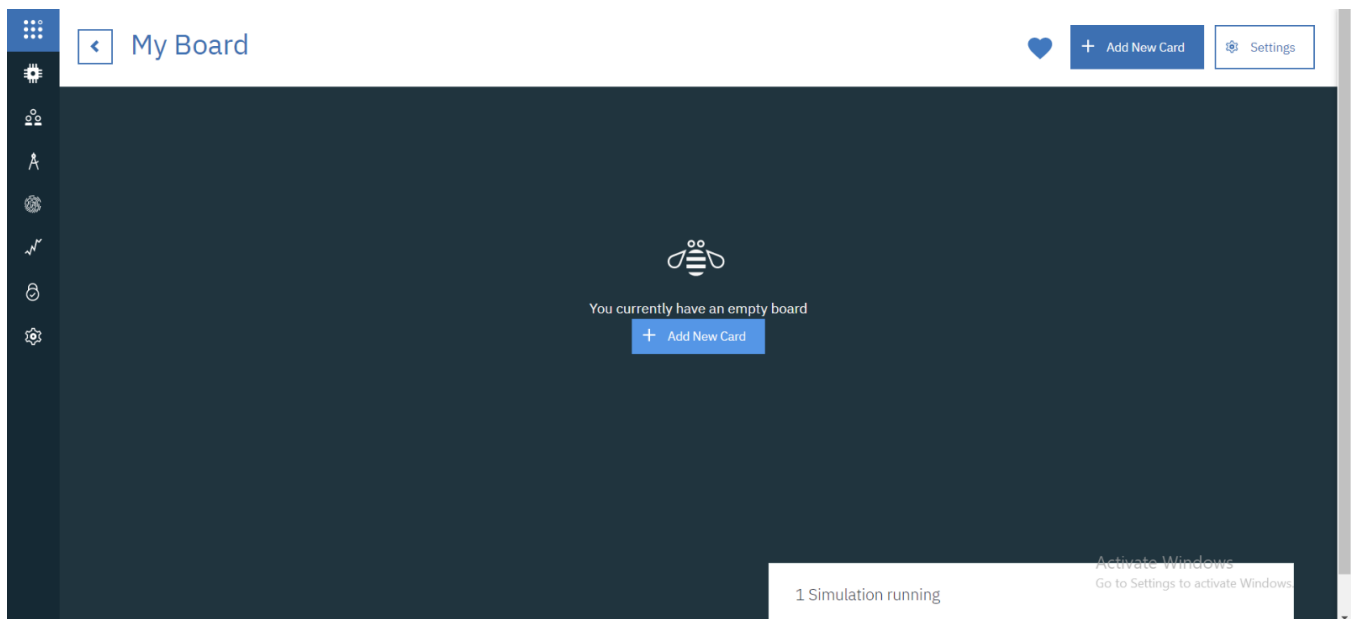
- Then click on Next you get the below window then again click on Submit



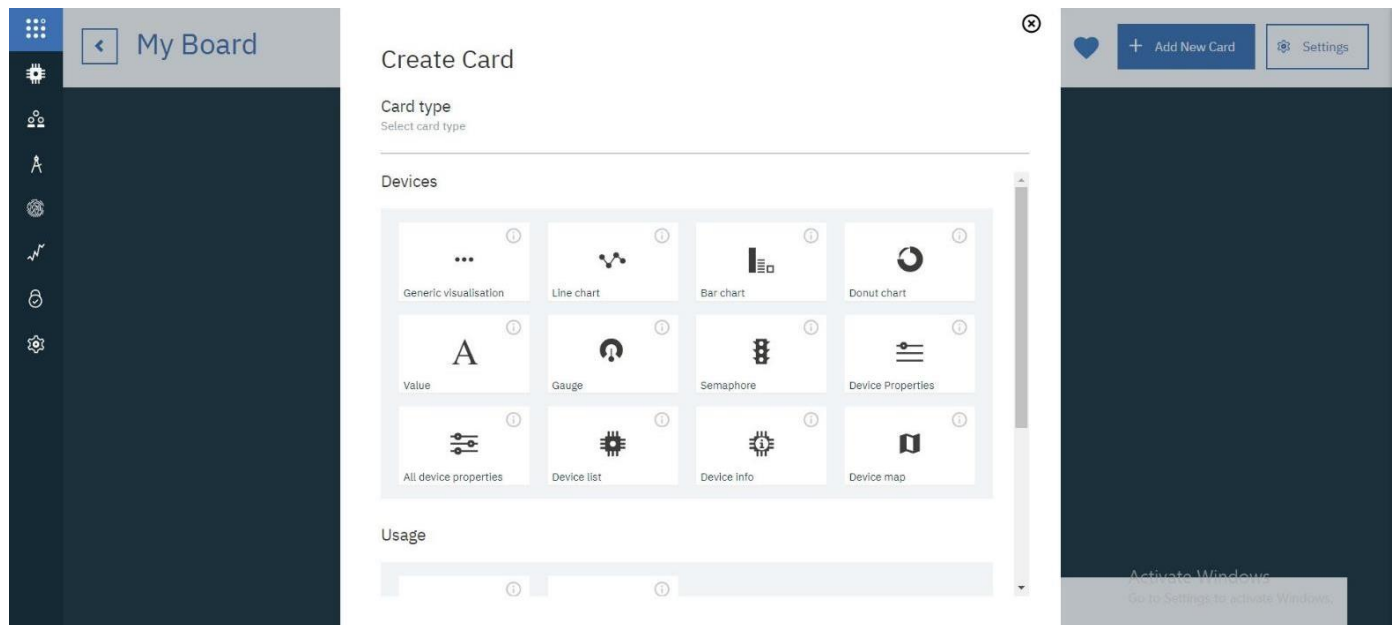
- Then double click on your boards name which you have created.



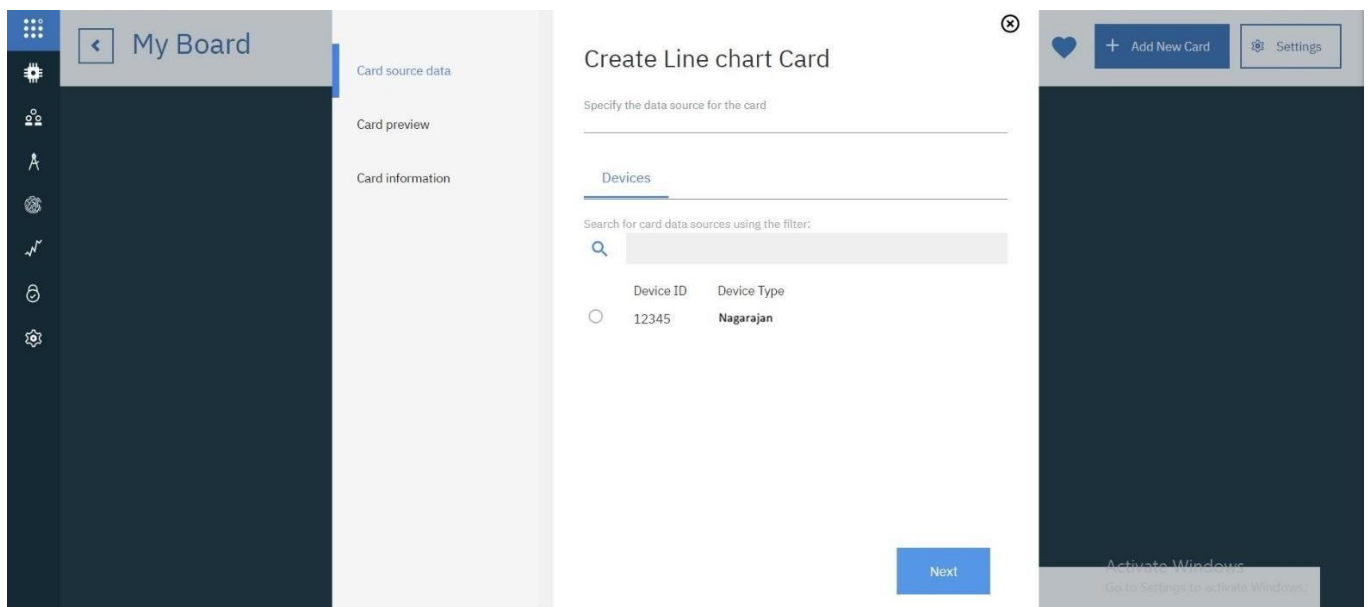
- Click on Add New Card



- Select the type of Graph u want accordingly and click next



- You get the below window, choose the Device and click on Next.



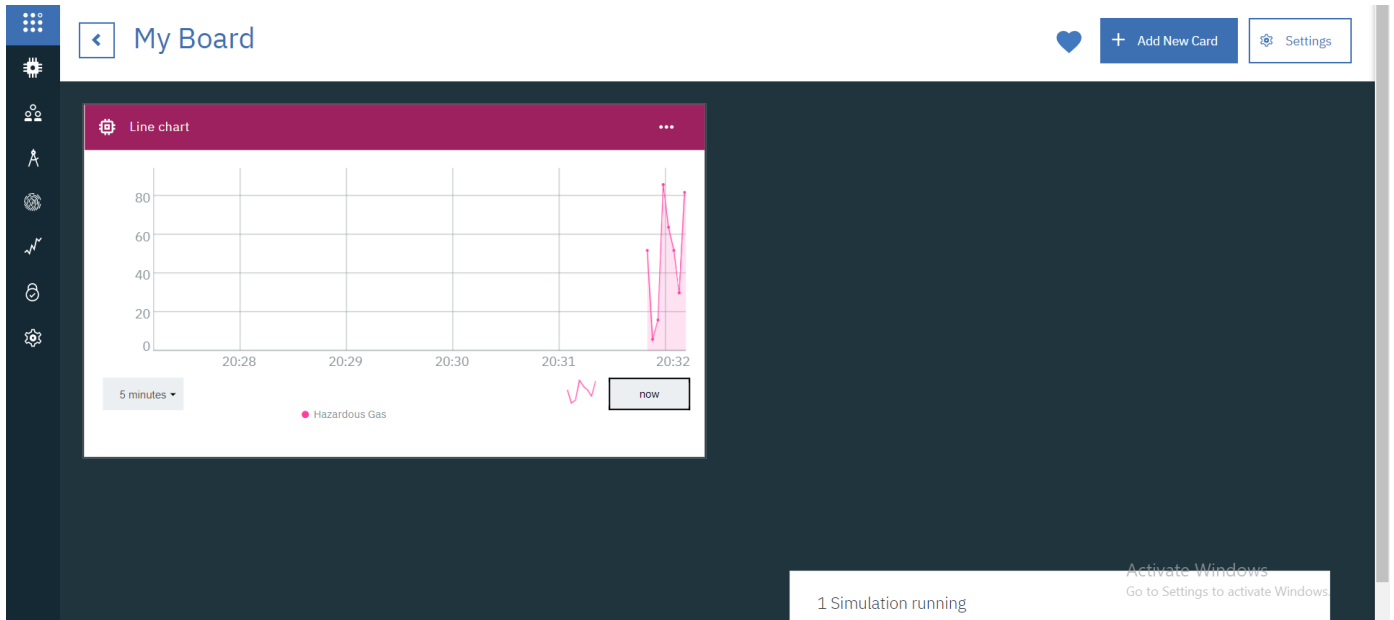
- Select the event, properly to be visualized on your graph and click next. In my case it is humidity

The screenshot shows the 'Create Line chart Card' interface. On the left, a sidebar contains icons for various data sources. The main panel is titled 'Temp & Hum' and has a sidebar with 'Card source data' (12345), 'Card preview', and 'Card information'. The 'Card source data' section is active, showing a list of data sources. The 'Card preview' section shows a line chart. The 'Card information' section shows the card's configuration. The 'Create Line chart Card' form is open, showing the 'Connect data set' section. The 'Event' field is set to 'event_1'. The 'Property' field is set to 'Temperature'. The 'Name' field is set to 'Temperature'. The 'Type' dropdown is set to 'Number'. The 'Unit' field is empty. The 'Max' field is set to '100'. The 'Back' and 'Next' buttons are at the bottom.

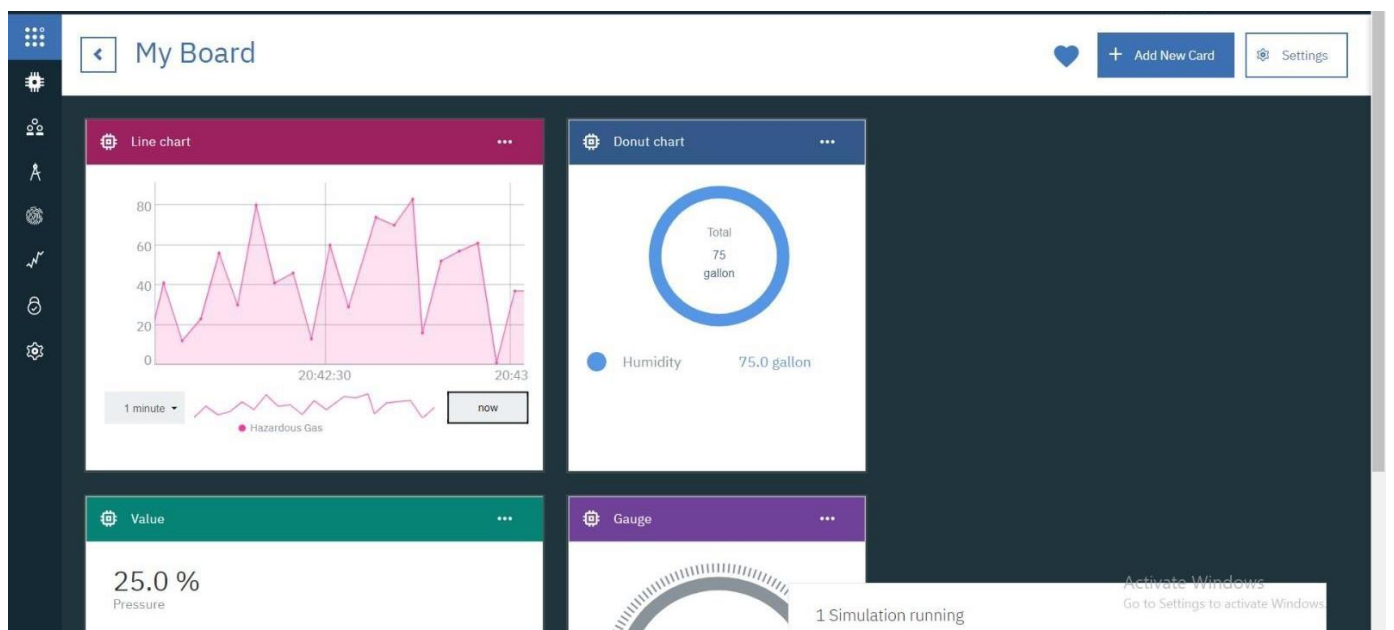
- Then select the size of the graph and color of the graph board you want and click next

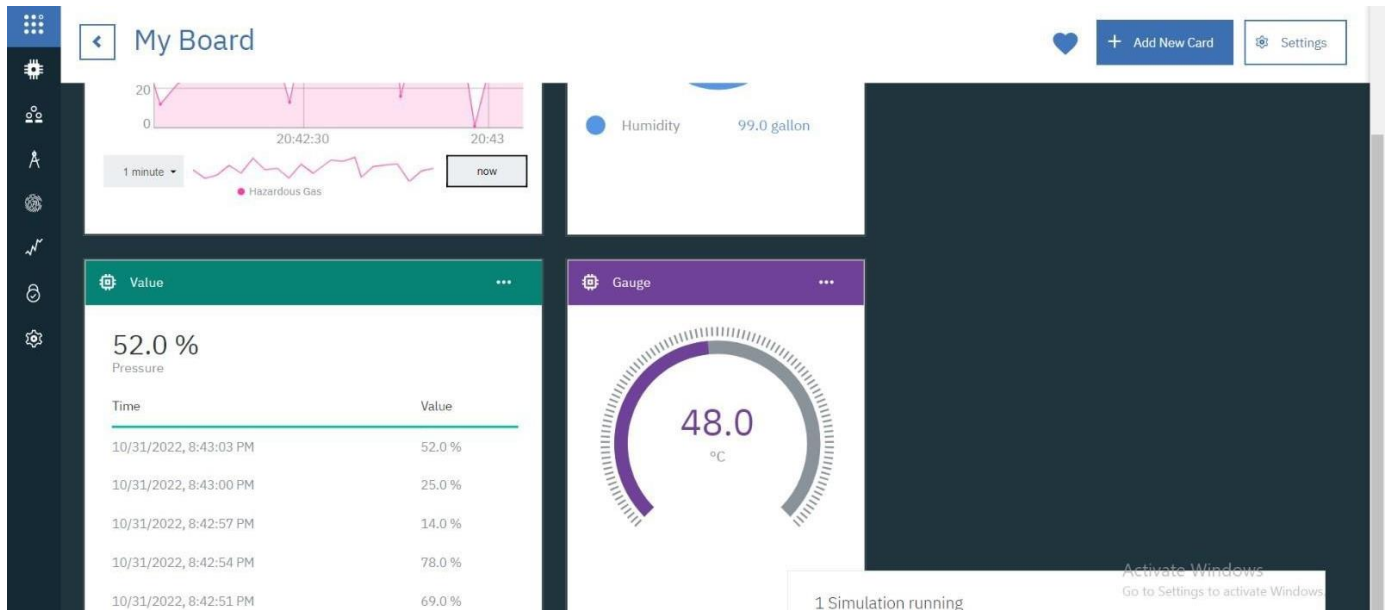
The screenshot shows the 'Create Line chart Card' interface. On the left, a sidebar contains icons for various data sources. The main panel is titled 'Temp & Hum' and has a sidebar with 'Card source data' (12345), 'Card preview', and 'Card information'. The 'Card source data' section is active, showing a list of data sources. The 'Card preview' section shows a line chart. The 'Card information' section shows the card's configuration. The 'Create Line chart Card' form is open, showing the 'Enter title and description of the card' section. The 'Title' field is set to 'Line chart'. The 'Color scheme' section shows a selection of color schemes. The 'A line chart to display time series information with historic and live data' text is displayed. The 'Back' and 'Submit' buttons are at the bottom.

➤ Here is the graph



➤ Repeat the process to get different graphs.





RESULT:

Hence, we were able to send data from our pi to IBM Watson and visualize it on a graph.