

# Smart Employee Tracking System Using IOT

**Industrial/Practical Training Report**

*Submitted to the*

*Department of Electronics and Communication Engineering,*

**SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE**

In partial fulfillment of the requirements,

for the award of the Degree of

***Bachelor of Technology***

in

***Electronics and Communication Engineering***

By

**NAZIYA PARVEEN**

**(20485A0414)**

**Department of Electronics and Communication Engineering**

**SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE**

**SESHADRI RAO KNOWLEDGE VILLAGE**

**GUDLAVALLERU -521356**

**ANDHRA PRADESH**

**2022-23**



## DECLARATION

I am NAZIYA PARVEEN hereby declare that this industrial training report is the record of authentic work carried out by me during the period 26.03.2022 to 30.07.2022 in **Smart Bridge Pvt.Ltd** under the supervision of my training In charge Nikhil Gurram Smart Bridge Pvt.Ltd.

Signature:

Name of the student: NAZIYA PARVEEN

## **Acknowledgement**

I am very glad to express my deep indebtedness to all the employees of Smart Bridge Pvt.Ltd. Without their support and guidance, it would not have been possible for this training to have materialized and taken a concrete shape. I extend my deep gratitude to my training mentor -Nikhil Gurram, Gnaneswar Bandari, Sai Rajeti, who supported us all the time. I owe my personal thanks to our HOD **Dr.Y.Rama Krishna** and our industrial training coordinator **Mr. N.Samba Murthy** for undergoing training at a reputed company like Smart Bridge Pvt.Ltd.

We would like to take this opportunity to express our profound sense of gratitude to our beloved Principal **Dr.G.V.S.N.R.V.Prasad**, for providing us all the required facilities. We express our thanks to Teaching and Non-Teaching staff of electronics and communication engineering department who helped us directly or indirectly in completion of our internship.

**NAZIYA PARVEEN**

**20485A0414**

# CONTENTS

## TITLE

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

### CHAPTER 01:

- 1.0 Introductions to IoT
- 1.1 Introduction to open Hardware Platforms and Tinkercad Circuits
- 1.2 Arduino -Tinkercad
- 1.3 Introduction to Python and Python Basics
  - 1.3.1 Python Introduction
  - 1.3.2 Python Basics
  - 1.3.3 Conditional Statements and Functional basics
  - 1.3.4 Data Variables and Basic Variables
  - 1.3.5 Lists, Tuples and Dictionaries
  - 1.3.6 Functions and Modules
  - 1.3.7 Files IO and Python Inbuild Libraries
  - 1.3.8 Python OOPS Concepts
  - 1.3.9 Networking -Socket Programming

### CHAPTER 02:

- 2.0 IBM Cloud Platforms
  - 2.1.0 Introduction to IBM Cloud and its Architecture
  - 2.1.1 IBM IoT Device creation and connecting Internal Simulator
  - 2.1.2 Connecting to IBM IoT Platform and sending sensor data using python code

### CHAPTER 03:

- 3.0 IBM Cloud Services
  - 3.0.0 IBM Cloudant
  - 3.0.1 IBM Object Storage
- 3.1 Nodered
  - 3.1.0 Retrieving Data from IBM IoT Platform
  - 3.1.1 Sending Command to the Device
- 3.4 Text To Speech and To Text Using Nodered
- 3.5 Watson Assistant using python code

### CHAPTER 04:

- 4.1 Applications
- 4.2 IBM Cloud Functions

**4.2.0** Smart Security Using Nodered Service

**4.2.1** Smart Tracking (Geofence)

**CHAPTER 05:**

**5.0** Project

**5.1** Purpose of the Project

**5.2** Proposed Solution

**5.3** Block Diagram

**5.4** Hardware/Software Designing

**CHAPTER 06:**

**6.0** Conclusion

## ABSTRACT

We have done our internship at **Smart Bridge** in Virtual mode. There are different areas in Electronics and communication Engineering and we have chosen Internet of Things (IoT), newly emerged technology with a group of students.

In our Internship, we have studied various Virtual Platforms like Tinker cad, IBM Cloud Platform and Python Programming. We have learnt how to use Virtual Environment to perform the Hardware Projects using the Tinker cad and ARDUINO. We have done the Smart Employee Tracking System Using IOT.

As an application of these, we have done a project comprised of a Team of 5 members. Our Project name is Smart Employee Tracking System Using IOT. With the help of Smart Internz Platform we started our project with the knowledge we have gained in this Internship. We successfully completed our Project using the Python Programming and IBM Cloud Services.

In this way the knowledge we have gained in the Smart Bridge company is useful to us.

Key words-IoT,Arduino,cloud ,virtual.

## LIST OF FIGURES

<b>Figure No</b>	<b>Figure Name</b>	<b>Page No</b>
1.	Internet of Things (IoT) Application in different Sectors	8
2.	IoT Architecture	9
3.	IoT Technology Stack	9
4.	Arduino UNO Board	10
5.	Experiments in Tinkercad	11
6.	Addition, Subtraction, Multiplication, Division using Python	13
7.	String, Hello World in Python Programming	13
8.	Various Operations using Python.	14
9.	Socket Programming Code- Creating Socket & Waiting for Connection	21
10.	Command Prompt Output-Client Data	22
11.	Output from Python Compiler - From Server.	22
12.	IBM IoT Platform Architecture	29
13.	IBM account Creation & IoT Service	30-32
14.	Python Programming to send data	33
15.	Cloudant in catalog page	37
16.	Cloudant database creating page	38
17.	Cloudant launch dashboard	38
18.	Creating database	39
19.	Editor Page	39
20.	Data is created in j.son format	40
21.	Object storage	40
22.	Creating object storage	41
23.	Creating buckets	41
24.	Data is stored	42
25.	Node-Red flow diagram	43
26.	IoT Simulator	44
27.	Sending Command to device	44
28.	Receiving Data from web	44
29.	Architecture of Watson Visual Recognition	45
30.	Watson assistant AI platform	45
31.	Watson assistant preview	46
32.	Controlling the mobile app application	47
33.	Block Codes	48
34.	Nodered Code for Smart Tracking with Geofence	48
35.	Smart Tracking with Geofence	49

## LIST OF TABLES

<b>S.No</b>	<b>Table Name</b>	<b>Page No</b>
1.	Arduino UNO Feature	10
2.	Modes description in Opening files in python	18
3.	Python File Methods Description	19



## CHAPTER 01

### 1.0 Introductions to IoT:

What is IoT?

- Connecting everyday things embedded with electronics, software and sensors to the internet enabling them to collect and exchange data.

Benefits of IoT:

1. Improved Performance
2. Reduced Cost
3. Improved Data Collection
4. Improved Customer Engagement
5. New Revenue Streams

Applications:

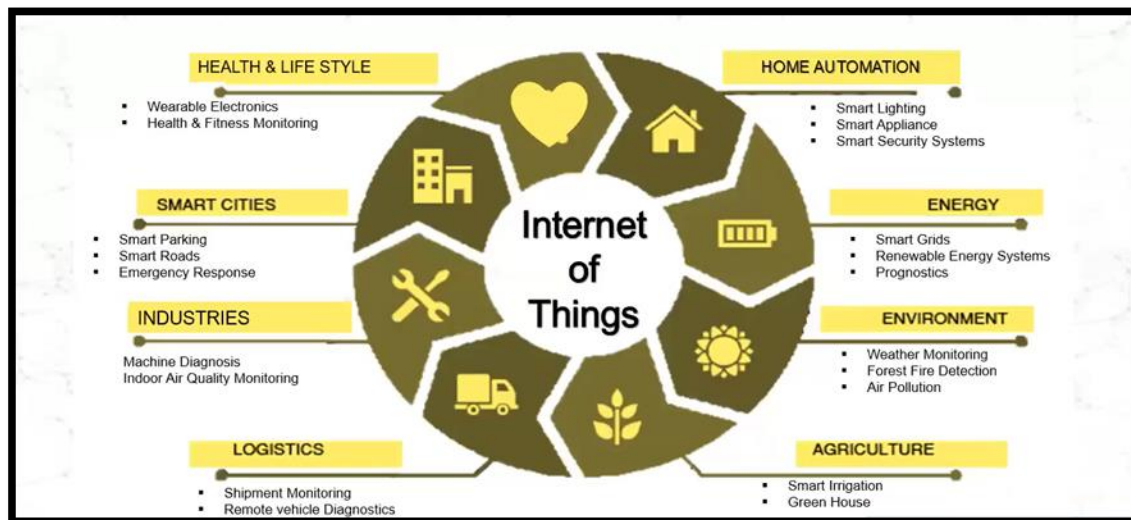


Fig 1: Internet of Things (IoT) Application in different Sectors

## What is IoT Architecture?

IoT architecture is the system of numerous elements: sensors, protocols, actuators, cloud services, and layers. Given its complexity, there exist 4 stages of IoT architecture. Such a number is chosen to steadily include these various types of components into a sophisticated and unified network.

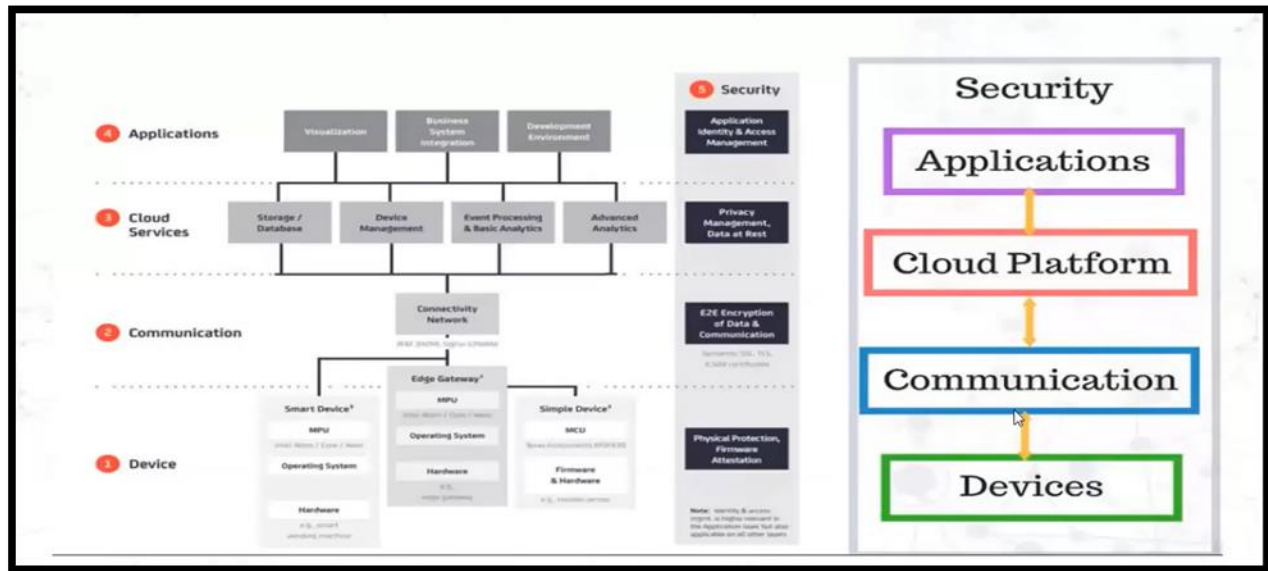


Figure 2: IoT Architecture

## IoT Technology Stack:

- The IoT technology stack is nothing else than a range of technologies, standards and applications, which lead from the simple connection of objects to the Internet to the most easy and most complex applications that use these connected things, the data they gather and communicate and the different steps needed to power these applications.

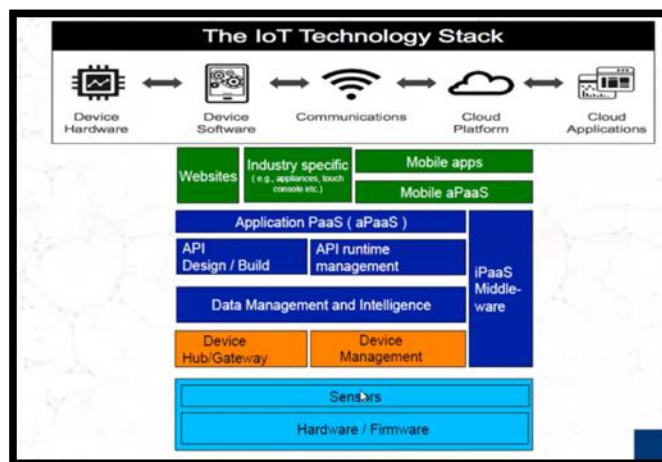


Figure 3: IoT Technology Stack

## 1.1 Introduction to open Hardware Platforms and Tinker cad Circuits:

### What is Arduino?

- Arduino is an open-source platform used for building electronics projects.
- Easy tool for fast prototyping.
- Consists of both a physical programmable circuit board and a piece of software.

### Why Arduino?

- Open-Source Platform
- Inexpensive
- Does not need a separate piece of hardware
- Arduino IDE uses a simplified version of C++
- Cross-Platform
- Provides a standard form factor.

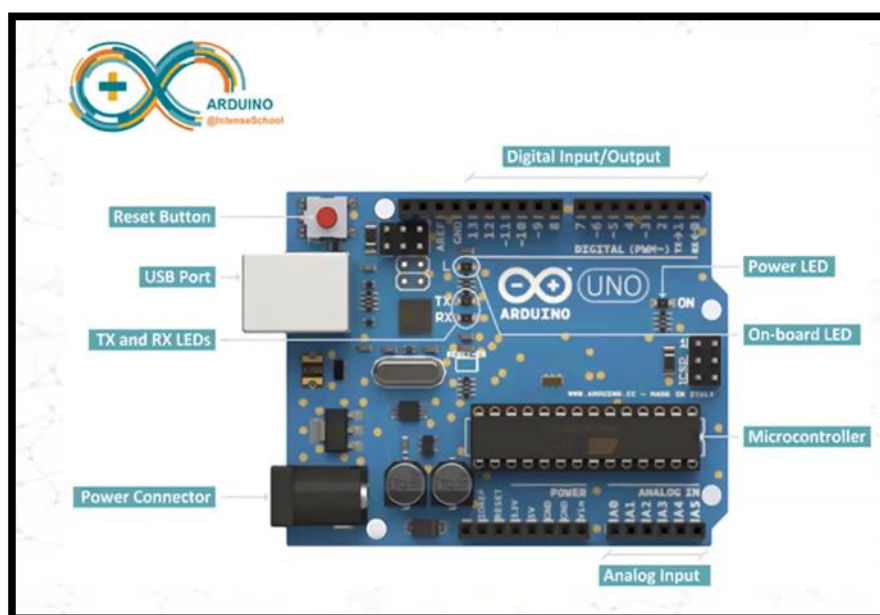


Figure 4: Arduino UNO Board

### Arduino UNO Features:

Operating Voltage	5V and 3.3V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3 V Pin	50 mA
Flash Memory	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEROM	1 KB (ATmega328)

Clock Speed	16 MHz
Microcontroller	ATmega328

Table 1: Arduino UNO Feature

## 1.2 Arduino -Tinkercad:

### Use of Tinkercad

- Tinkercad is a free, easy-to-use app for 3D design, electronics, and coding. It's used by teachers, kids, hobbyists, and designers to imagine, design, and make anything!

List of the Experiments done with Tinkercad:

- Buzz
- Smart Door
- Blinker with TMP
- PIR Sensor
- LED Blink
- Piezo
- Ultrasonic Sensor
- Temp Sensor
- Servo Motor
- LED with Potentiometer
- Digital Input and Output

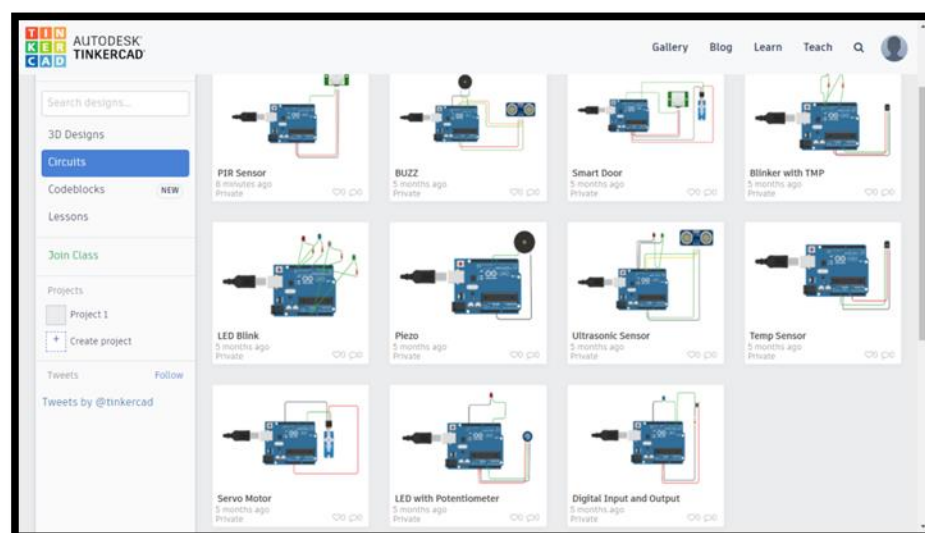


Figure 5: Experiments in Tinkercad

### 1.3 Introduction to Python and Python Basics:

#### 1.3.0 Python Introduction:

##### What is Python?

- General Purpose
- Interpreted
- High Level
- Dynamically Typed

##### General Purpose:

Python language is a general-purpose language because it can be applicable to all domains, such type of languages is generic and not specialized.

##### Interpreted:

An Interpreted language will read the Raw code without being explicitly compiled before execution.

##### High Level:

Python allows programmers to express their logic in a form which is very close to human language. This helps in many computing aspects like memory management and data management.

##### Dynamically Typed:

Python identifies the type of variables on the basis of what kind of data you have assigned to the variable.

##### Why Choose Python?

- Readily available and open source
- Huge Libraries
- Easy to understand and learn
- Big developer Communities
- Fewer code lines and larger Functionalities.

#### 1.3.1 Python Basics:

Here is a list of the Python keywords. Enter any keyword to get more help.

False	class	from	or
None	continue	global	pass
True	def	if	raise
and	del	import	return
as	elif	in	try
assert	else	is	while
async	except	lambda	with
await	finally	nonlocal	yield

break                      for                      not

### Basic Python Programming: -



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> a=70
>>> type(a)
<class 'int'>
>>> b=3.0
>>> type(b)
<class 'float'>
>>> a=5
>>> b=7
>>> a+b
12
>>> a-b
-2
>>> a*b
35
>>> a/b
0.7142857142857143
>>> a=5.0
>>> b=7.0
>>> a*b
35.0
>>> b-a
2.0
>>> x=3+5j
>>> type(x)
<class 'complex'>
>>> y=2-2.2j
>>> x+y
(5+2.8j)
>>> a=True
>>> type(a)
<class 'bool'>
>>> b=False
>>> type(b)
<class 'bool'>
>>> a and b
```

Figure 6: Addition, Subtraction, Multiplication, Division using Python.



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> s1="hello"
>>> type(s1)
<class 'str'>
>>> print(s1)
hello
>>> s1
'hello'
>>> s2="welcome"
>>> s1+s2
'helloworld'
>>> print("Hello world")
Hello world
>>> print("Hello \n World")
Hello
World
>>> print("Hello \nWorld")
Hello
World
>>> print("Hello \t World")
Hello   World
>>> print("Hello\tWorld")
Hello  World
>>> s="Hello World"
>>> s.capitalize()
'Hello world'
>>> len(s)
11
>>> s[0]
'H'
>>> s[4]
'o'
>>> s[0:4]
'Hell'
>>> s[0:5]
'Hello'
>>>
```

Figure 7: String, Hello World in Python Programming.



```

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> name="vinay kumar "
>>> name.lstrip()
'vinay kumar '
>>> name.rstrip()
'vinay kumar'
>>> name.strip()
'vinay kumar'
>>> s="This is python"
>>> s.find('is')
2
>>> s.find('python')
8
>>> s.replace('is','was')
'Thwas was python'
>>> s.upper()
'THIS IS PYTHON'
>>> s.lower()
'this is python'
>>> s.title()
'This Is Python'
>>> s[3]
'g'
>>> s="This is a anIal"
>>>

```

Figure 8: Various Operations using Python.

### 1.3.2 Conditional Statements and Functional basics:

#### Conditional Statements:

A conditional statement in Python is handled by if statements and we saw various other ways we can use conditional statements like if and else over here.

- "if condition" -It is used when you need to print out the result when one of the conditions is true or false.
- "else condition"- it is used when you want to print out the statement when your one condition fails to meet the requirement
- "elif condition" -It is used when you have third possibility as the outcome. You can use multiple elif conditions to check for 4<sup>th</sup>, 5<sup>th</sup>, 6<sup>th</sup> possibilities in your code
- We can use minimal code to execute conditional statements by declaring all condition in single statement to run the code
- If Statement can be nested.

#### Syntax:

```

if expression
    Statement
else expression
    Statement

```

#### Basic Functions:

What is function in Python?

- A function is a group of related statements that performs a specific task.

- Functions help break our program into smaller and modular chunks. As our program grows larger and larger, functions make it more organized and manageable.
- It avoids repetition and makes the code reusable.

Function definition that consists of the following components:

1. Keyword **def** that marks the start of the function header.
2. A function name to uniquely identify the function. Function naming follows the same rules of writing identifiers in python.
3. Parameters (arguments) through which we pass values to a function. They are optional.
4. A colon (:) to mark the end of the function header.
5. Optional documentation string (docstring) to describe what the function does.
6. One or more valid python statements that make up the function body. Statements must have the same indentation level (usually 4 spaces).
7. An optional **return** statement to return a value from the function.

### **Types of Functions:**

1. Built-in functions - Functions that are built into Python.
2. User-defined functions- Functions defined by the users themselves.

#### **1.3.3 Data Variables and Basic Variables:**

Data Types in Python:

Every value in Python has a data type. Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.

There are various data types in Python. Some of the important types are listed below.

#### **1. Python Numbers:**

- Integers, floating point numbers and complex numbers fall under Python numbers category. They are defined as int, float, and complex classes in Python.
- We can use the **type()** function to know which class a variable or a value belongs to. Similarly, the **isinstance()** function is used to check if an object belongs to a particular class.

#### **2. Python Strings:**

- String is a sequence of Unicode characters. We can use single quotes or double quotes to represent strings. Multi-line strings can be denoted using triple quotes, ""

#### **3. Python Set:**

- Set is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

#### **1.3.4 Lists, Tuples and Dictionaries:**

##### **1. Python List:**

- List is an ordered sequence of items. It is one of the most used data type in Python and is very flexible. All the items in a list do not need to be of the same type.
- Declaring a list is pretty straightforward. Items separated by commas are enclosed within brackets [ ].



- We can use the slicing operator [ ] to extract an item or a range of items from a list. The index starts from 0 in Python.

## 2. Python Tuple:

- Tuple is an ordered sequence of items same as a list. The only difference is that tuples are immutable. Tuples once created cannot be modified.
- Tuples are used to write-protect data and are usually faster than lists as they cannot change dynamically.
- It is defined within parentheses ( ) where items are separated by commas.
- We can use the slicing operator [ ] to extract items but we cannot change its value.

## 3. Python Dictionaries:

- Dictionary is an unordered collection of key-value pairs.
- It is generally used when we have a huge amount of data. Dictionaries are optimized for retrieving data. We must know the key to retrieve the value.
- In Python, dictionaries are defined within braces { } with each item being a pair in the form **key:value**. Key and value can be of any type.
- We use key to retrieve the respective value. But not the other way around.

### 1.3.5 Functions and Modules:

#### Python Functions:

- Functions are a handy way to isolate a particular part of your programs are functionality and make it reusable. Modules are a way to collect a number of helpful functions in one file, which you can then use in multiple projects and share with other programmers.
- We are already been using functions extensively in our programs; functions are one of the main building blocks of Python programming. Whenever you are typed something like **len(x)** or **type(y)** or even **random.choice ([1, 2, 3])**, you have been using functions. It is just that these functions come pre-defined by Python.

#### Python Modules:

- A Python module is a file that contains one or more function definitions. Modules are a handy way of keeping related functions together, so that you can easily reuse them between projects, or share them with other programmers.
- Making a module is easy. Just make a file that has a **.py** extension and contains only **import** statements and function definitions. Here is a module called **restaurantutils** that contains many of the functions.

### 1.3.6 Files IO and Python Inbuilt Libraries:

#### Files:

- Files are named locations on disk to store related information. They are used to permanently store data in a non-volatile memory.
- RAM is volatile which loses its data when the computer is turned off, we use files for future use of the data by permanently storing them.

- When we want to read from or write to a file, we need to open it first. When we are done, it needs to be closed so that the resources that are tied with the file are freed.

Hence, in Python, a file operation takes place in the following order:

1. Open a file
2. Write
3. Read (perform operation)
4. Close the file

### 1. Opening Files in Python:

- Python has a built-in **open( )** function to open a file. This function returns a file object, also called a handle, as it is used to read or modify the file accordingly.
- We can specify the mode while opening a file. In mode, we specify whether we want to read **r**, write **w** or append **a** to the file. We can also specify if we want to open the file in text mode or binary mode.

Mode	Description
R	Opens a file for reading. (default)
w	Opens a file for writing. Creates a new file if it does not exist or truncates the file if it exists.
X	Opens a file for exclusive creation. If the file already exists, the operation fails.
A	Opens a file for appending at the end of the file without truncating it. Creates a new file if it does not exist.
T	Opens a file for appending at the end of the file without truncating it. Creates a new file if it does not exist.
B	Opens in binary mode.
+	Opens a file for updating (reading and writing)

Table 2: Modes description in Opening files in python

### 2. Writing to files in Python:

- In order to write into a file in Python, we need to open it in write **w**, append **a** or exclusive creation **x** mode.

- We need to be careful with the **w** mode, as it will overwrite into the file if it already exists.
- Writing a string or sequence of bytes for binary files is done using the **write( )** method. This method returns the number of characters written to the file.
- This program will create a new file named **test.txt** in the current directory if it does not exist. If it does exist, it is overwritten.

### 3. Reading Files in Python:

- To read a file in Python, we must open the file in reading **r** mode.
- There are various methods available for this purpose. We can use the **read(size)** method to read in the **size** number of data. If the **size** parameter is not specified, it reads and returns up to the end of the file.
- We can see that the **read( )** method returns a newline. Once the end of the file reached, we get an empty string on further reading.
- We can change our current file cursor position using the **seek( )** method. Similarly, the **tell( )** method returns our current position in number of bytes.

### 4. Closing Files in Python:

- When we are done with performing operations on the file, we need to properly close the file.
- Closing a file will free up the resources that were tied with the file. It is done using the **close( )** method available in Python.
- Python has a garbage collector to clean up unreferenced objects but we must not rely on it to close the file.

### Python File Methods:

There are various methods available with the file object. Some of them have been used in the above examples.

Here is the complete list of methods in text mode with a brief description:

Method	Description
<b>close()</b>	Closes an opened file. It has no effect if the file is already closed.
<b>detach()</b>	Separates the underlying binary buffer from the <b>TextIOBase</b> and returns it.
<b>fileno()</b>	Returns an integer number (file descriptor) of the file.
<b>flush()</b>	Flushes the write buffer of the file stream.
<b>isatty()</b>	Returns <b>True</b> if the file stream is interactive.
<b>read(n)</b>	Reads at most <b>n</b> characters from the file. Reads till end of file if it is negative or <b>None</b> .
<b>readable()</b>	Returns <b>True</b> if the file stream can be read

	from.
readline( <b>n</b> =-1)	Reads and returns one line from the file. Reads in at most <b>n</b> bytes if specified.
readlines( <b>n</b> =-1)	Reads and returns a list of lines from the file. Reads in at most <b>n</b> bytes/characters if specified.
seek( <b>offset</b> , <b>from</b> , <b>SEEK_SET</b> )	Changes the file position to <b>offset</b> bytes, in reference to <b>from</b> (start, current, end).
seekable()	Returns <b>True</b> if the file stream supports random access.
tell()	Returns the current file location.
truncate ( <b>size</b> = <b>None</b> )	Resizes the file stream to <b>size</b> bytes. If <b>size</b> if not specified, resizes to current location.
writable()	Returns <b>True</b> if the file stream can be written to.
write( <b>s</b> )	Writes the string <b>s</b> to the file and returns the number of characters written.
writelines( <b>lines</b> )	Writes a list of <b>lines</b> to the file.

Table 3: Python File Methods Description

### 1.3.7 Python OOPS Concepts:

#### Python Object Oriented Programming:

- Python is a multi-paradigm programming language. It supports different programming approaches.
- One of the popular approaches to solve a programming problem is by creating objects. This is known as Object-Oriented Programming (OOP)

An object has two characteristics:

- Attributes
- Behavior

In Python, the concept of OOP follows some basic principles:

#### 1) Class:

- A Class is a blueprint for the object.
- We can think of class as a sketch of a parrot with labels. It contains all the details about the name, colors, size etc. Based on these descriptions, we can study about the parrot. Here, a parrot is an object.

#### 2) Object:

- An object (instance) is an instantiation of a class. When class is defined, only the description for the object is defined. Therefore, no memory or storage is allocated.

#### 3) Methods:

- Methods are functions defined inside the body of a class. They are used to define the behaviors of an object.

**4) Inheritance:**

- Inheritance is a way of creating a new class for using details of an existing class without modifying it. The newly formed class is a derived class (or child class). Similarly, the existing class is a base class (or parent class).

**5) Encapsulation:**

- Using OOP in Python, we can restrict access to methods and variables. This prevents data from direct modification which is called encapsulation. In Python, we denote private attributes using underscore as the prefix i.e single \_ or double \_\_.

**6) Polymorphism:**

- Polymorphism is an ability (in OOP) to use a common interface for multiple forms (data types).
- Suppose, we need to color a shape, there are multiple shape options (rectangle, square, circles). However we could use the same method to color any shape. This concept is called Polymorphism.

**Key Points on OOPS:**

- 1) Object-Oriented Programming makes the program easy to understand as well as efficient.
- 2) Since the class is sharable, the code can be reused.
- 3) Data is safe and secure with data abstraction.
- 4) Polymorphism allows the same interface for different objects, so programmers can write code efficient code.

**1.3.8 Networking - Socket Programming:**

**Networking:**

A computer network is a group of computers that use a set of common communication protocols over digital interconnections for the purpose of sharing resources located on or provided by the network nodes.

**OSI Model:**

- 1) Physical
- 2) Data Link
- 3) Network
- 4) Transport
- 5) Session
- 6) Presentation
- 7) Application.

## TCP/IP & Socket Programming:

A Socket Programming interface provides the routines required for interprocess communication between applications, either on the local system or spread in a distributed, TCP/IP based network environment. Socket is uniquely defined by

- Internet Address (IP Address)
- Communication Protocol
- Port

```
ser.py - C:/Users/SmartBridgePC/Desktop/ser.py (3.8.2)
File Edit Format Run Options Window Help

import socket

s=socket.socket() #creation of socket

print("Socket is created")

s.bind(("localhost",3333)) # assign IP and port to socket

s.listen(3) #Listening limit for client

print("Waiting for connection")

while True:
    c,addr=s.accept() #it will accept the connection from client
    print("Connected with",addr)
    data=c.recv(1024).decode() #to recieve the data from client
    c.send(bytes("Welcome",'utf-8'))
    print("Client Data",data)
    c.close()
```

Figure 9: Socket Programming Code- Creating Socket & Waiting for Connection

The screenshot shows a Windows command prompt window titled "Python 3.8.2 Shell". The prompt is at the root directory. The user has run a command to start a Python server script. The output shows the server listening for connections. It receives multiple connections from 127.0.0.1, and for each connection, it prints "Client Data hello" and "Connected with (\'127.0.0.1\', 59783)". The output is as follows:

```

C:\>python ser.py
Client Data hello
Connected with (\'127.0.0.1\', 59783)
Client Data hello
Connected with (\'127.0.0.1\', 59784)
Client Data hello
Connected with (\'127.0.0.1\', 59785)
Client Data hello
Connected with (\'127.0.0.1\', 59786)
Client Data hello
Connected with (\'127.0.0.1\', 59787)
Client Data hello
Connected with (\'127.0.0.1\', 59788)
Client Data hello
Connected with (\'127.0.0.1\', 59789)
Client Data hello
Connected with (\'127.0.0.1\', 59790)
Client Data hello
Connected with (\'127.0.0.1\', 59791)
Client Data hello
Connected with (\'127.0.0.1\', 59792)
Client Data hello
Connected with (\'127.0.0.1\', 59793)
Client Data hello
Connected with (\'127.0.0.1\', 59794)
Client Data hello
Connected with (\'127.0.0.1\', 59795)
Client Data hello
Connected with (\'127.0.0.1\', 59796)
Client Data hello

```

Figure 10: Command Prompt Output-Client Data

A screenshot of a Python 3.8.2 Shell window. The window title is "Python 3.8.2 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The code in the editor is a loop that prints "Message from Server Welcome" 20 times. The output in the shell shows the same message repeated 20 times. The code is as follows:

```
import socket
s=socket.socket()
s.connect(('localhost', 8080))
while True:
    message = s.recv(1024)
    print(message)
```

Figure 11: Output from Python Compiler - From Server.

## 2.1 IBM Cloud Platforms:

### Cloud:

Cloud Computing is simply on-demand delivery of computing services over the internet.

Cloud Computing is categorized into different service models:

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

### Software as a Service (SaaS):

- Simplest to use and easier to deploy
- No software to install or configure
- Everything is available by web browser.
- Microsoft office and slack two popular SaaS products.

### Platform as a Service (PaaS):

- Developers and programmers upload their content to pre-configured servers.

- No installation or maintaining the server software or operating system.

#### **Infrastructure as a Service (IaaS):**

- Dedicated virtual systems that you manage and maintain yourself.
- Greatest degree of flexibility and customization in cloud computing.
- Amazon web services (AWS) and Digital ocean are widely known as IaaS provider.

#### **IBM IoT Platform Architecture:**

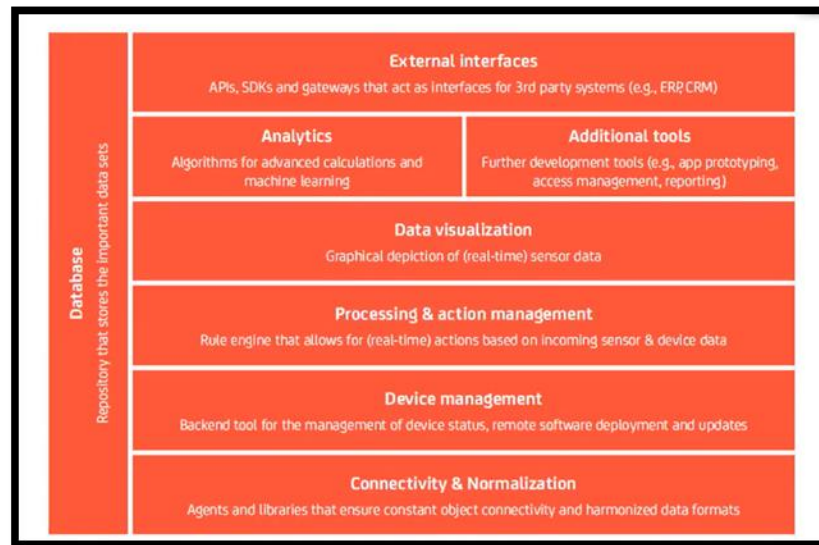


Figure 12: IBM IoT Platform Architecture

1. **Connectivity & Normalization**: Brings different protocols and different data formats into one software interface ensuring accurate data streaming and interaction with all devices.
2. **Device Management**: Ensures the connected things are working properly, seamlessly running patches and updates for software and applications running on the device or edge gateways.
3. **Database**: Scalable storage of device data brings the requirements for hybrid cloud-base data bases to a new level in terms of data volume, variety, velocity and veracity.
4. **Processing & Action Management**: Brings data to life with rule-based event-action-triggers enabling execution of smart actions based on specific sensor data.
5. **Analytics**: Performs a range of complex analysis from basic data clustering and deep machine learning to predictive analytics extracting the most value out of the IoT data-stream.
6. **Visualization**: Enables humans to see patterns and observe trends from visualization dashboards where data is vividly portrayed through line-, stacked-, or pie charts, 2D- or even 3D-models.

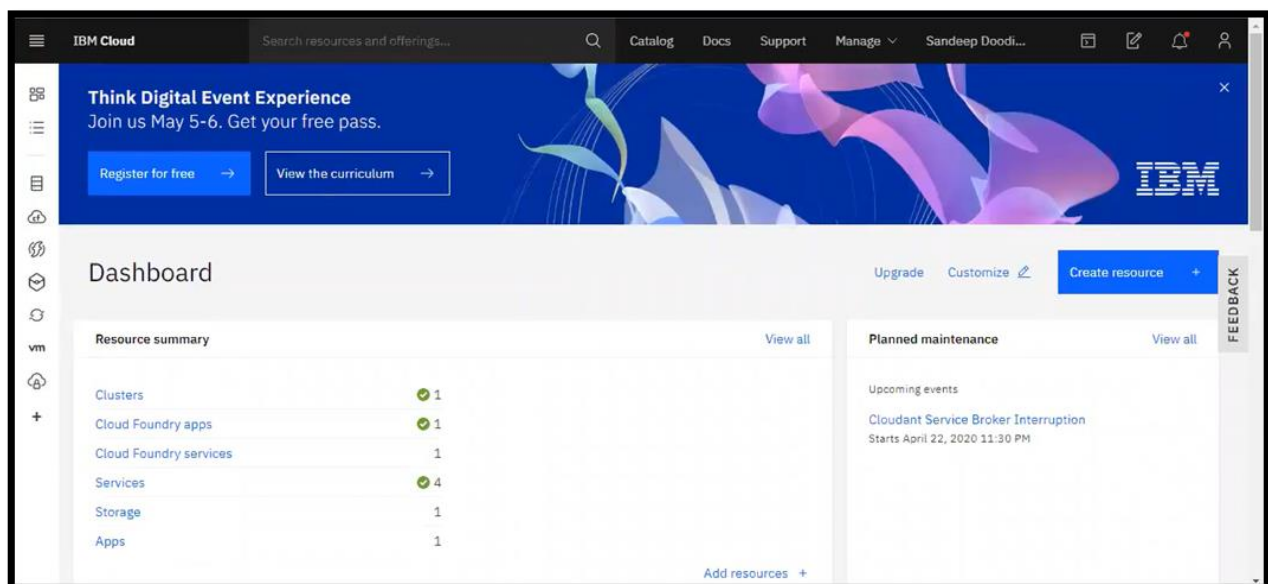


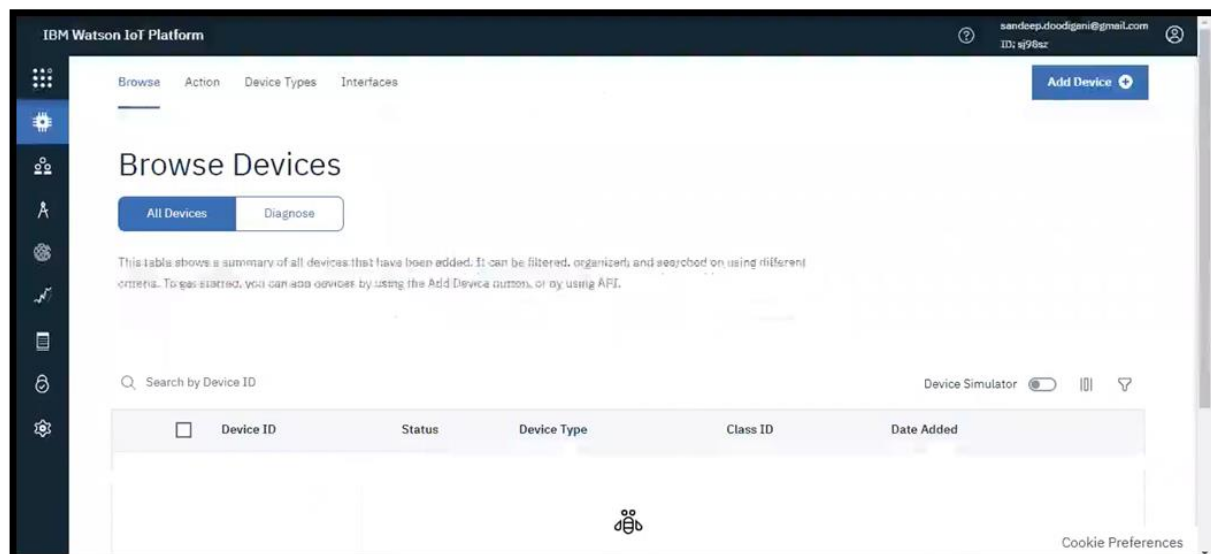
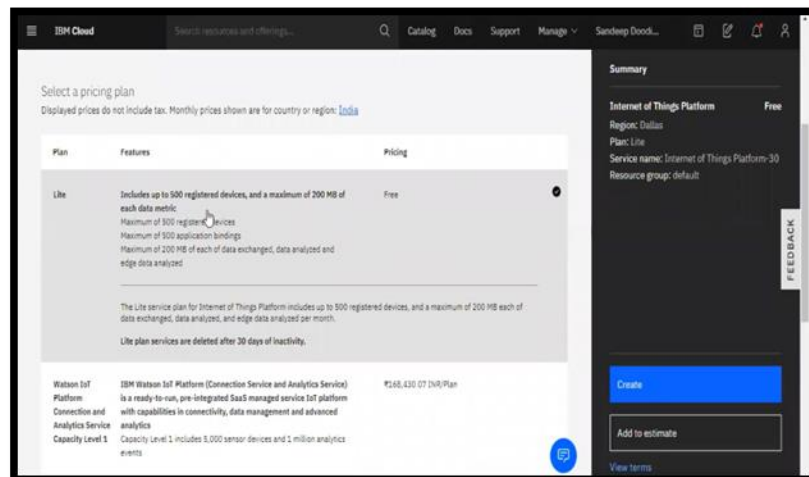
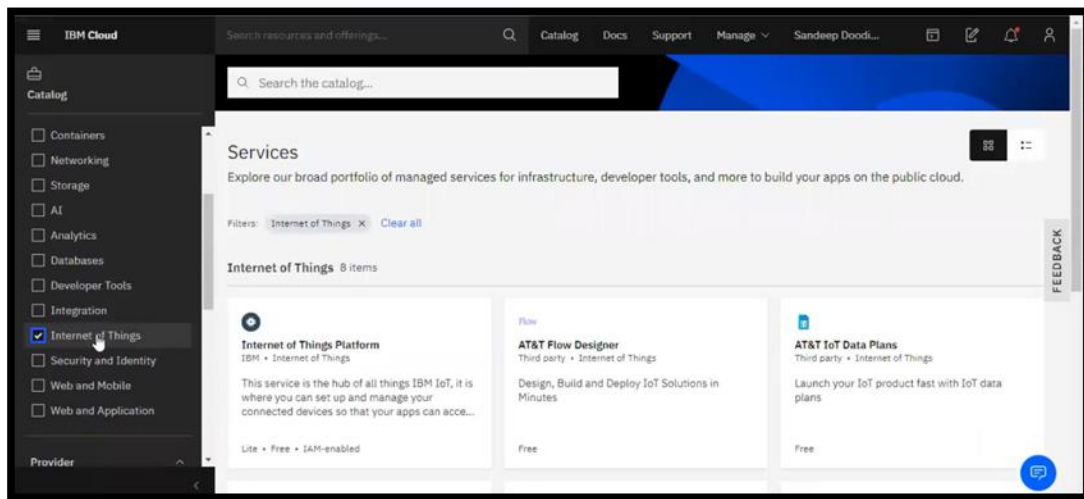
7. Additional Tools: Allow IoT developers prototype, test and market the IoT use case creating platform ecosystem apps for visualizing, managing and controlling connected devices.
8. External Interfaces: Integrate with 3<sup>rd</sup>-party systems and the rest of the wider IT-ecosystem via built-In application programming interfaces (API), software development kits (SDK), and gateways.

### 2.1.1 IBM IoT Device creation and connecting Internal Simulator:

Steps for IBM IoT device creation and connecting Internal Simulator:

1. First we have to Create IBM Cloud Account at <https://www.ibm.com/in-en/cloud> .
2. Next we have login into our account.
3. You will display the IBM dashboard. In that dashboard you can see our active IBM Services.
4. Click on catalog we will get the list of the services provided by the IBM Cloud.
5. Create Internet of Things Service.
6. After service is Created, we have to add devices to the IoT Platform.
7. We can add the devices like Raspberry Pie, Arduino etc.
8. We have to connect device to the IBM virtual IoT simulator <http://watson-iot-sensor-simulator.mybluemix.net/> .





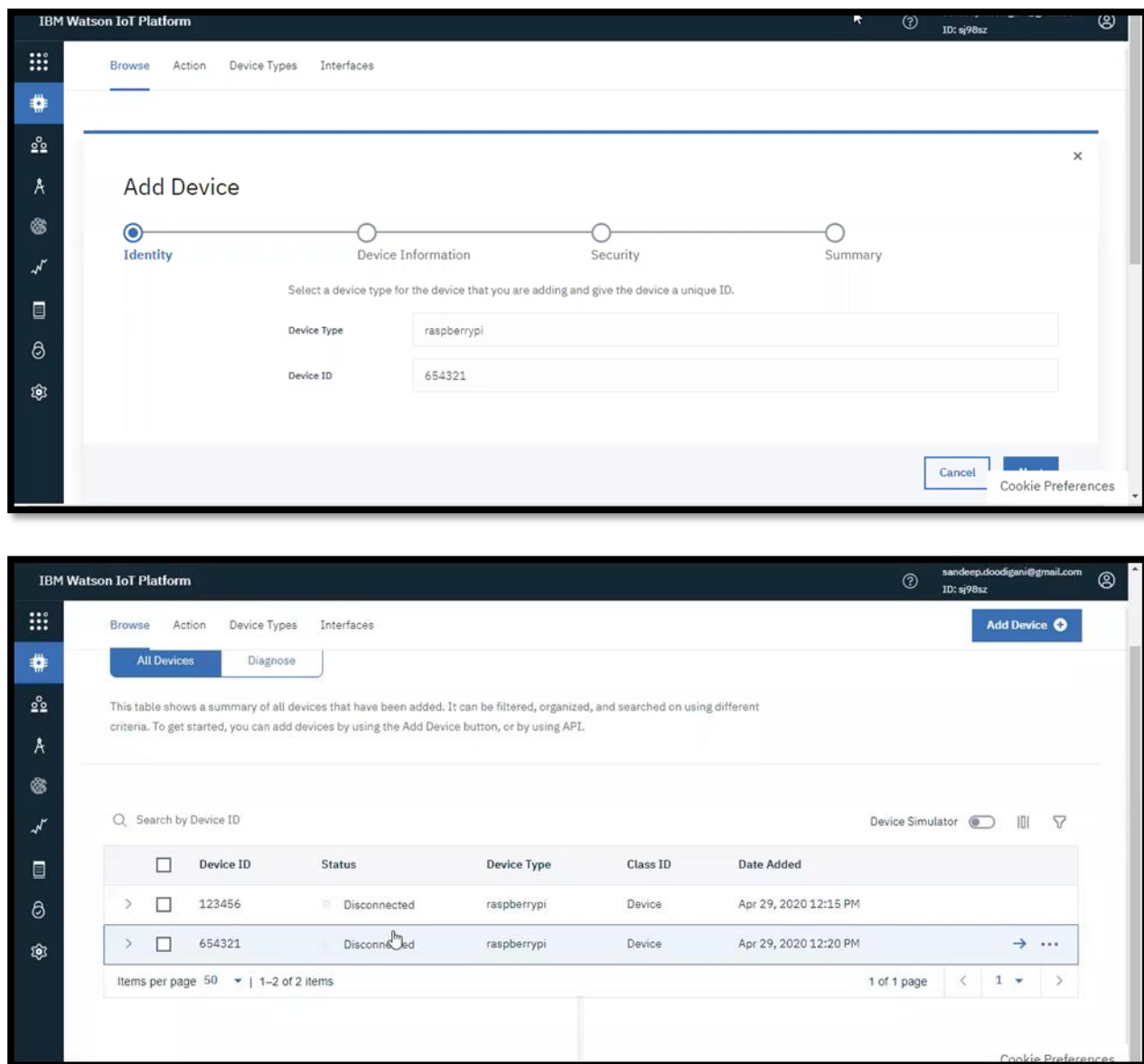
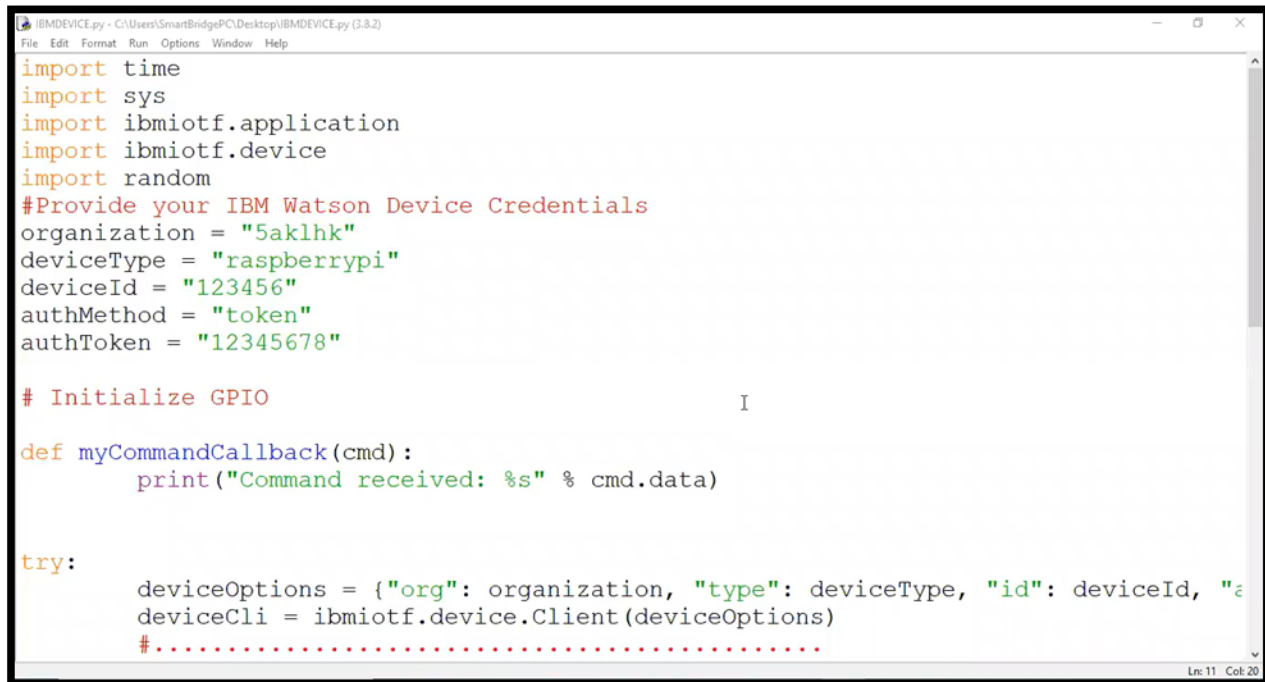


Figure 13: IBM account Creation &amp; IoT Service

### 2.1.2 Connecting to IBM IoT Platform and sending sensor data using python code:

- Previously we created IoT Platform and connected to Virtual IBM IoT simulator.
- Now we using python code to perform IoT
- By using Device data in the IoT Platform we can send data by using the Python code.
- To use Python code, we have to install some library files.

- We can check whether the required library files are available in our laptop/Computer by using command prompt. For installation use **pip install ibmiot** in command prompt
- For installation use **pip install ibmiot** in command prompt



```

IBMDEVICE.py - C:\Users\SmartBridgePC\Desktop\IBMDEVICE.py (3.8.2)
File Edit Format Run Options Window Help

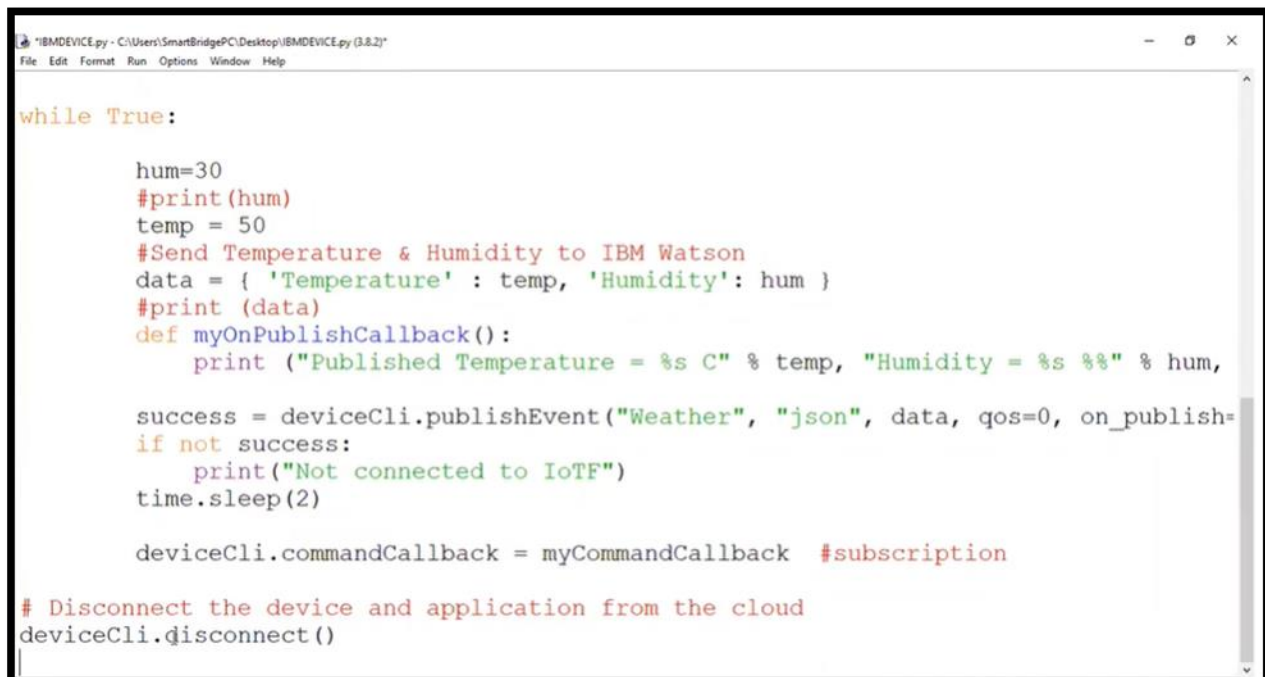
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "5aklhk"
deviceType = "raspberrypi"
deviceId = "123456"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "a
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #.....
  
```



```

IBMDEVICE.py - C:\Users\SmartBridgePC\Desktop\IBMDEVICE.py (3.8.2)
File Edit Format Run Options Window Help

while True:

    hum=30
    #print(hum)
    temp = 50
    #Send Temperature & Humidity to IBM Watson
    data = { 'Temperature' : temp, 'Humidity': hum }
    #print (data)
    def myOnPublishCallback():
        print ("Published Temperature = %s C" % temp, "Humidity = %s %" % hum,

    success = deviceCli.publishEvent("Weather", "json", data, qos=0, on_publish=
    if not success:
        print("Not connected to IoT")
        time.sleep(2)

    deviceCli.commandCallback = myCommandCallback #subscription

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

## CHAPTER 03

### 3.0 IBM Cloud Services

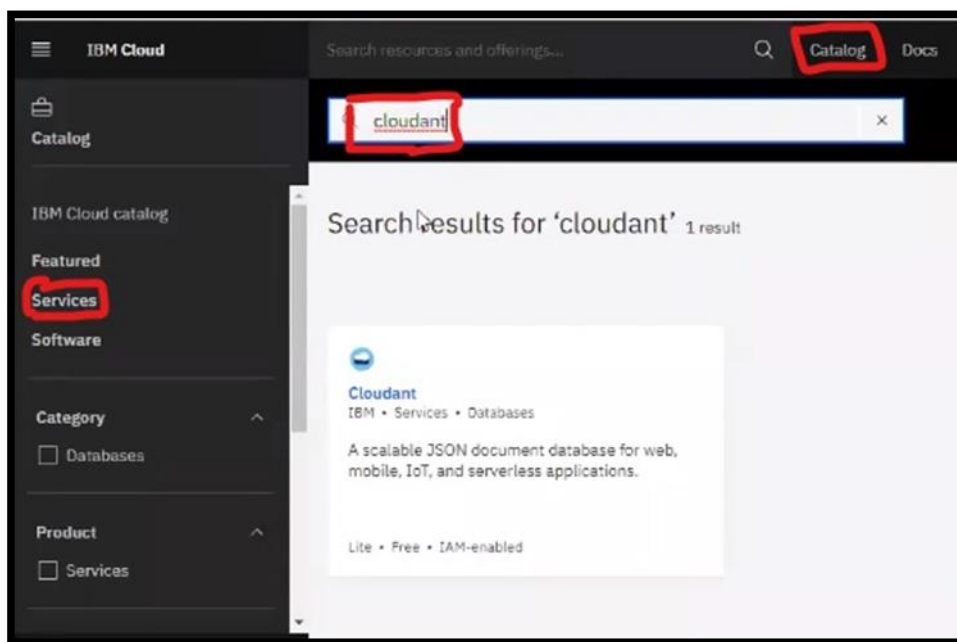
#### 3.0.0 IBM Cloudant:

What is IBM Cloudant?

IBM Cloudant is a fully managed JSON document database that offers independent serverless scaling of throughput capacity and storage.

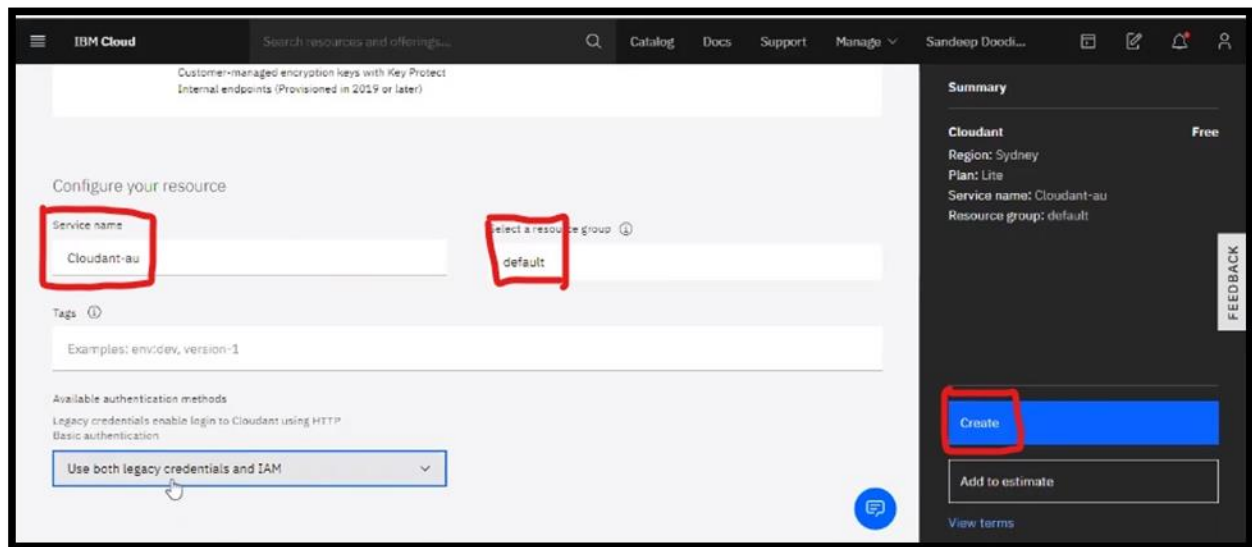
How to open IBM Cloudant and use it?

First fall open IBM cloud and there will open a dashboard page and click on catalog and there will see all the services of IBM. and click on catalog then left side we see services option click on it and search for Cloudant, and there will see a Cloudant data base and click on it.



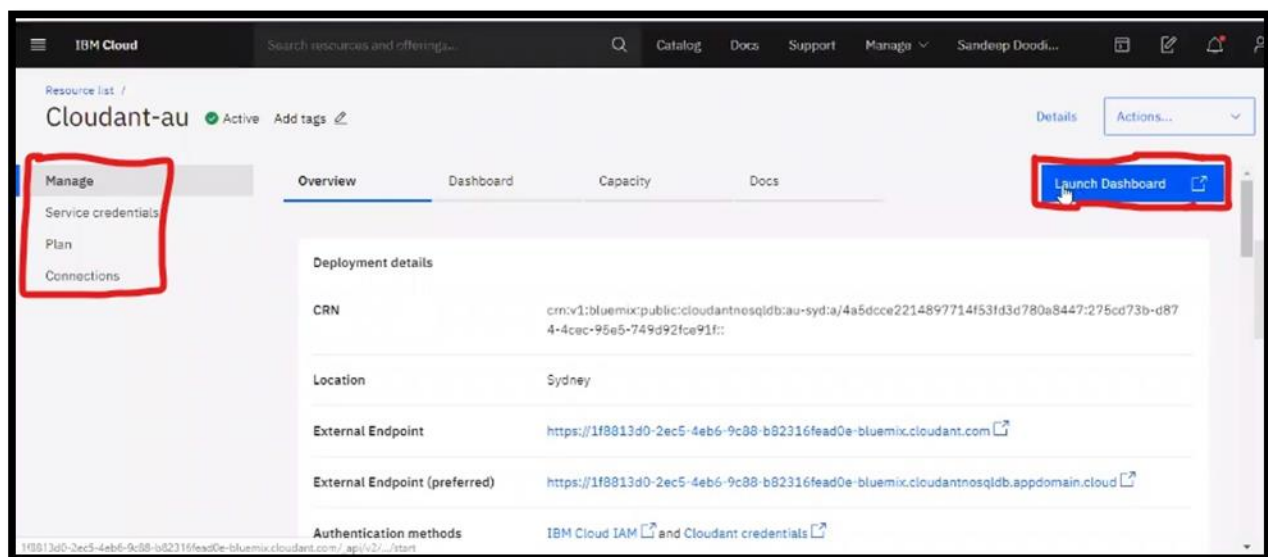
**Fig18:** Cloudant in catalog page

In Cloudant we store the data in JSON format. The Cloudant page will be open and fill necessary credentials and click on create.



**Fig 19:** Cloudant database creating page

Now open Cloudant service and click on launch dashboard page.



**Fig 20:** Cloudant launch dashboard

And database page will be open for us. and click on create database and there a pop page will be open on right side and create a database name and there we have an option is partitioning in that we have to click on non-partitioned and then click on create button.

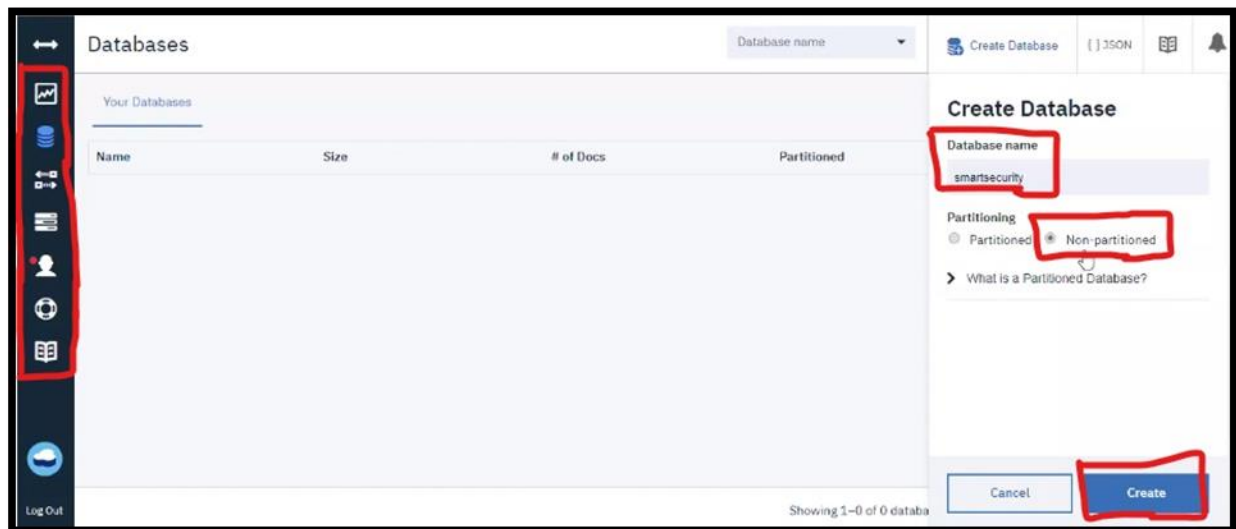


Fig 21: creating database

And we create our data here and document will be created in it. To run the python code in it. We have to open database page and check data is created or not.

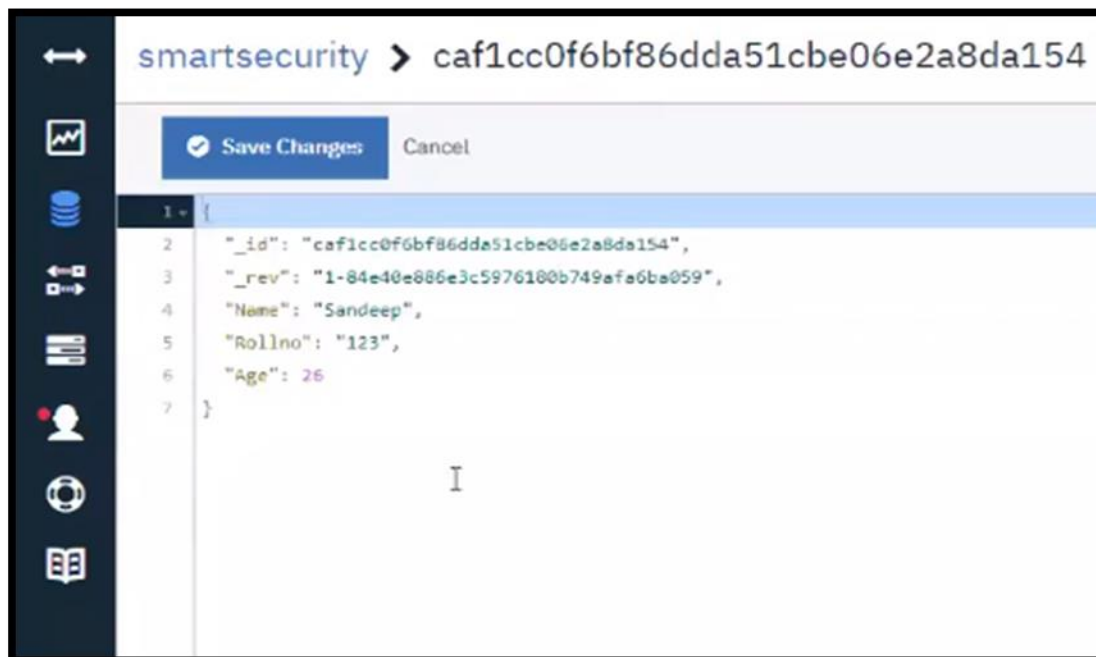
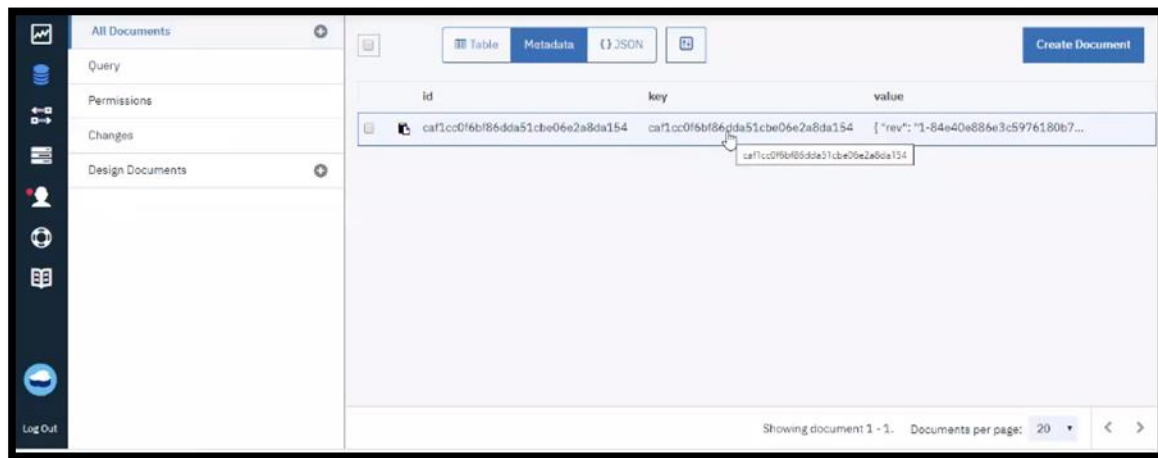


Fig 22: Editor page





The screenshot shows the IBM Cloud Object Storage interface. On the left is a sidebar with navigation options: All Documents, Query, Permissions, Changes, and Design Documents. The main area displays a document in JSON format. The document has a single key-value pair where the key is a long alphanumeric string and the value is a JSON object containing a 'rev' field.

id	key	value
caf1cc0f6bf86dda51cbe06e2a8da154	caf1cc0f6bf86dda51cbe06e2a8da154	{ "rev": "1-84e40e88e3c5976180b7..." }

At the bottom right, it indicates 'Showing document 1 - 1. Documents per page: 20'.

**Fig 23:** Data is created in j.son format

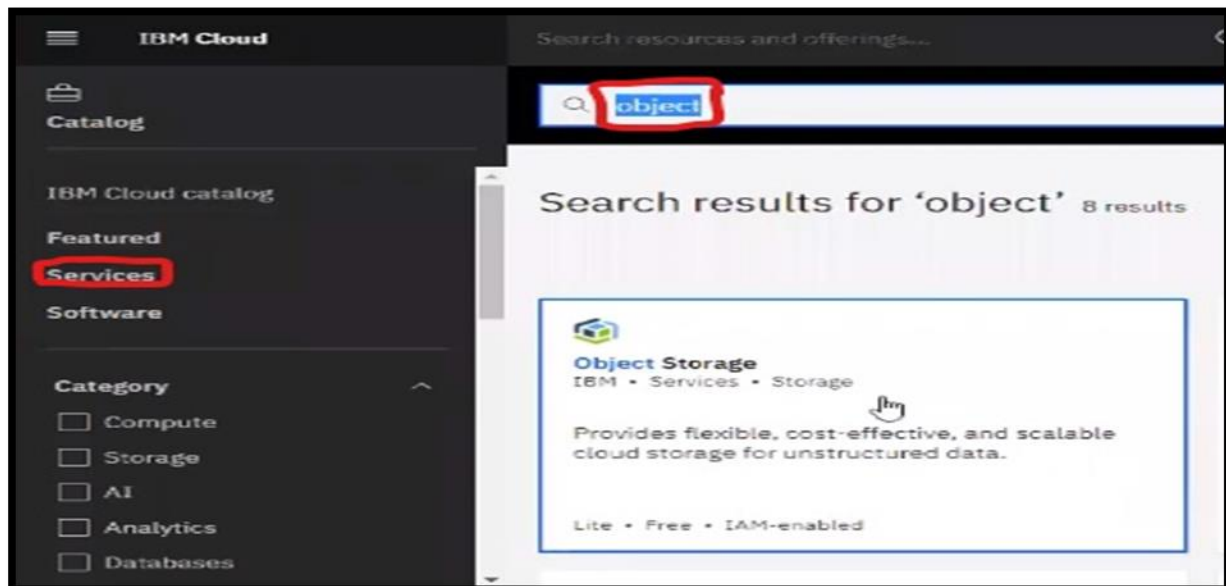
### 3.0.1 IBM Object Storage:

What is IBM object storage?

IBM Object Storage makes it possible to store practically limitless amounts of data, simply and cost effectively. It is commonly used for data archiving and backup; for web and mobile applications; and as scalable, persistent storage for analytics.

How to open IBM Object Storage and use it?

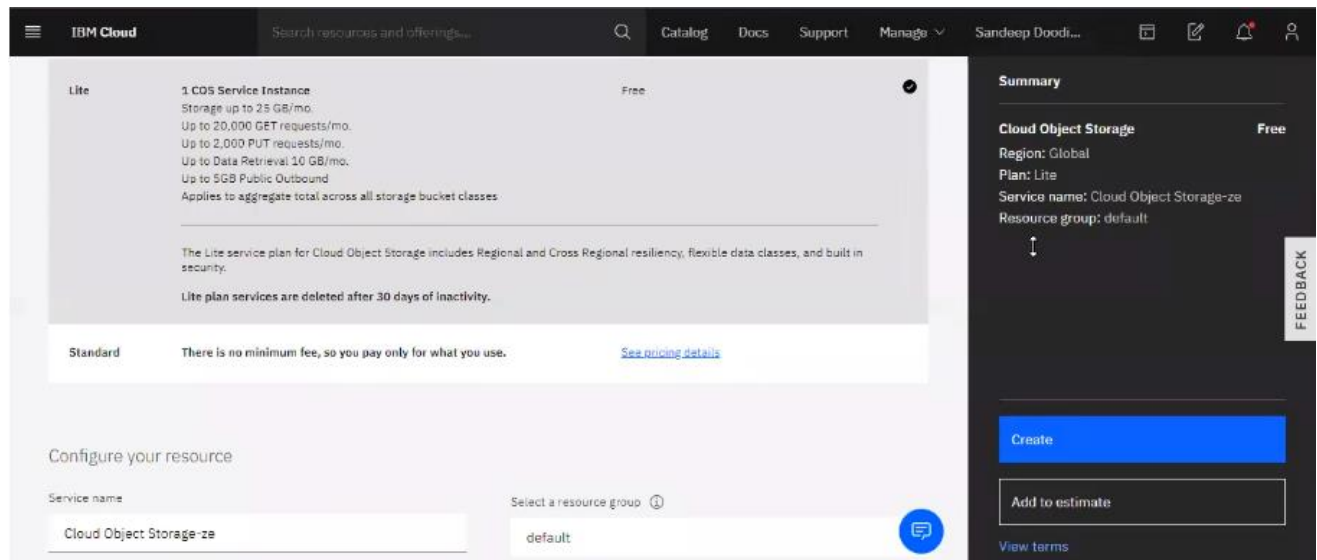
First fall open IBM cloud and there will open a dashboard page and click on catalog and there will see all the services of IBM. and click on catalog then left side we see a services option click on it. Search for Object Storage, there we observe Object Storage then click on it.



**Fig 24:** Object storage

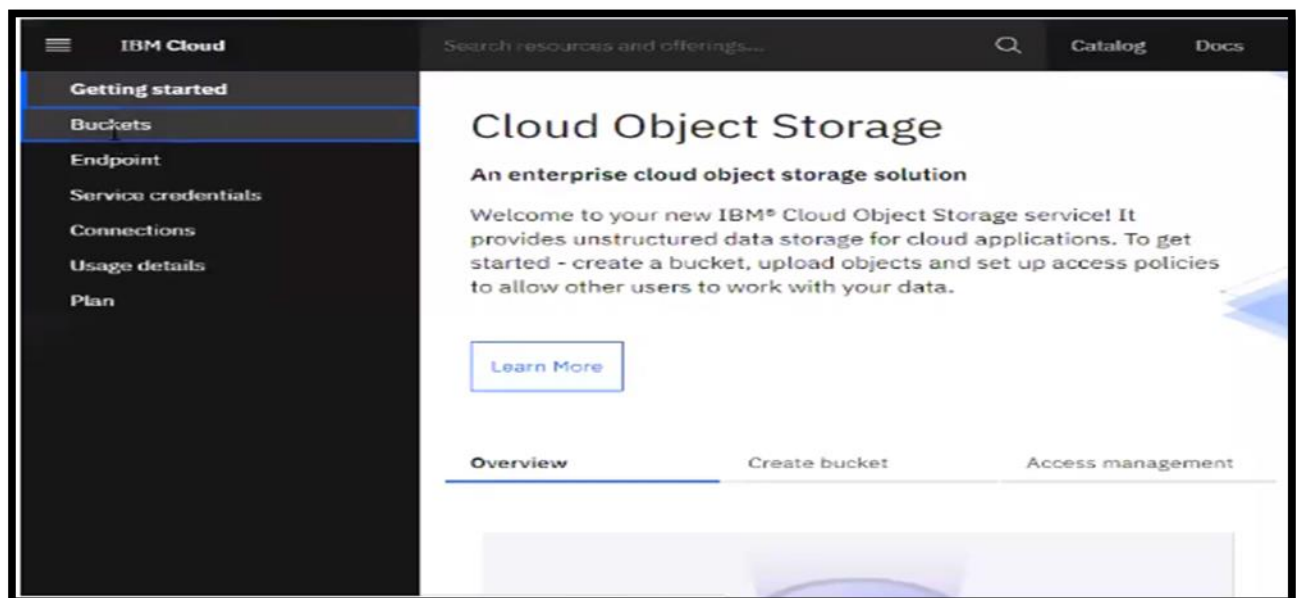
Object storage page will be open then click on create option then service will be created.





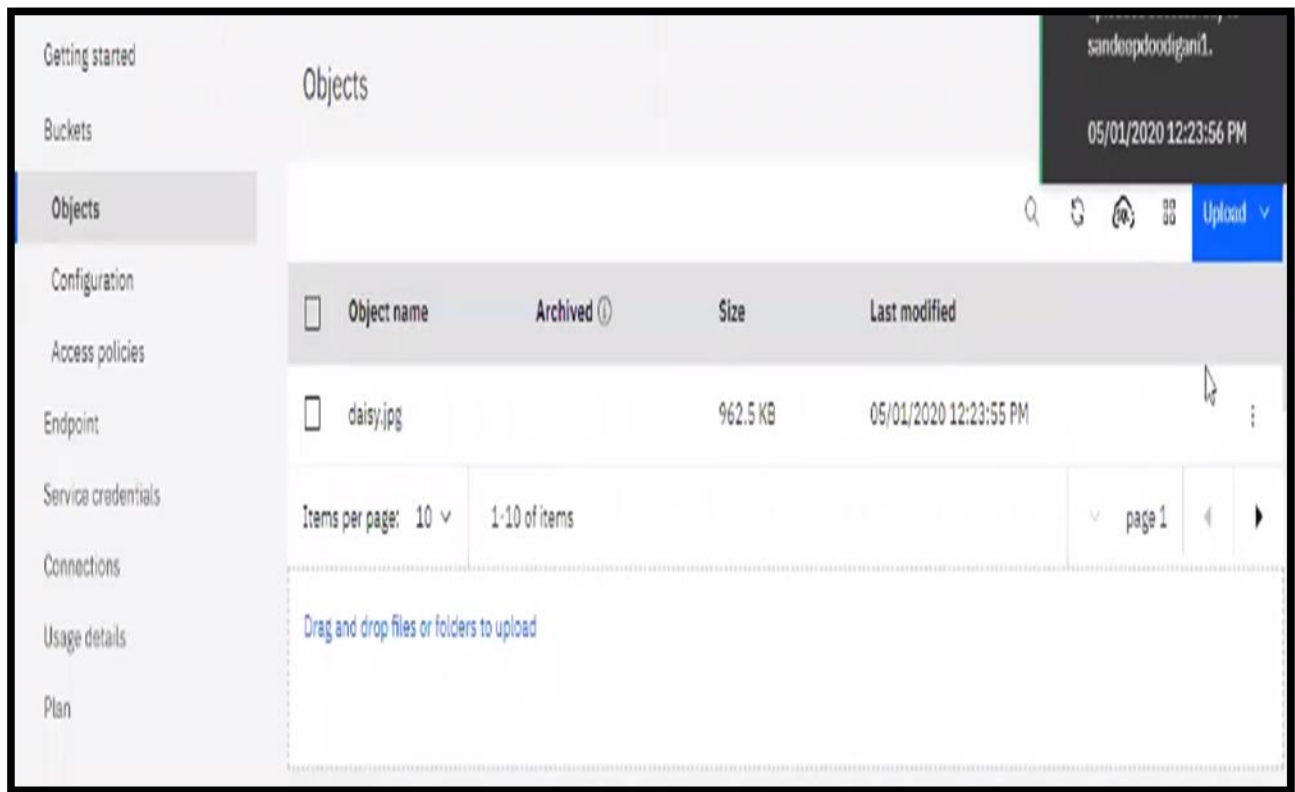
**Fig 25: Creating object storage**

Object storage will be open and click on buckets and then click on create bucket button there are different kind of buckets are there. We have to click on custom bucket and bucket should always be unique name and click on create bucket.



**Fig 26: Creating buckets**

There will have a drag and drop option then click on it. A pop page will be opened and select any file you want to store in it.



**Fig 27: Data is stored**

### 3.1 NODERED

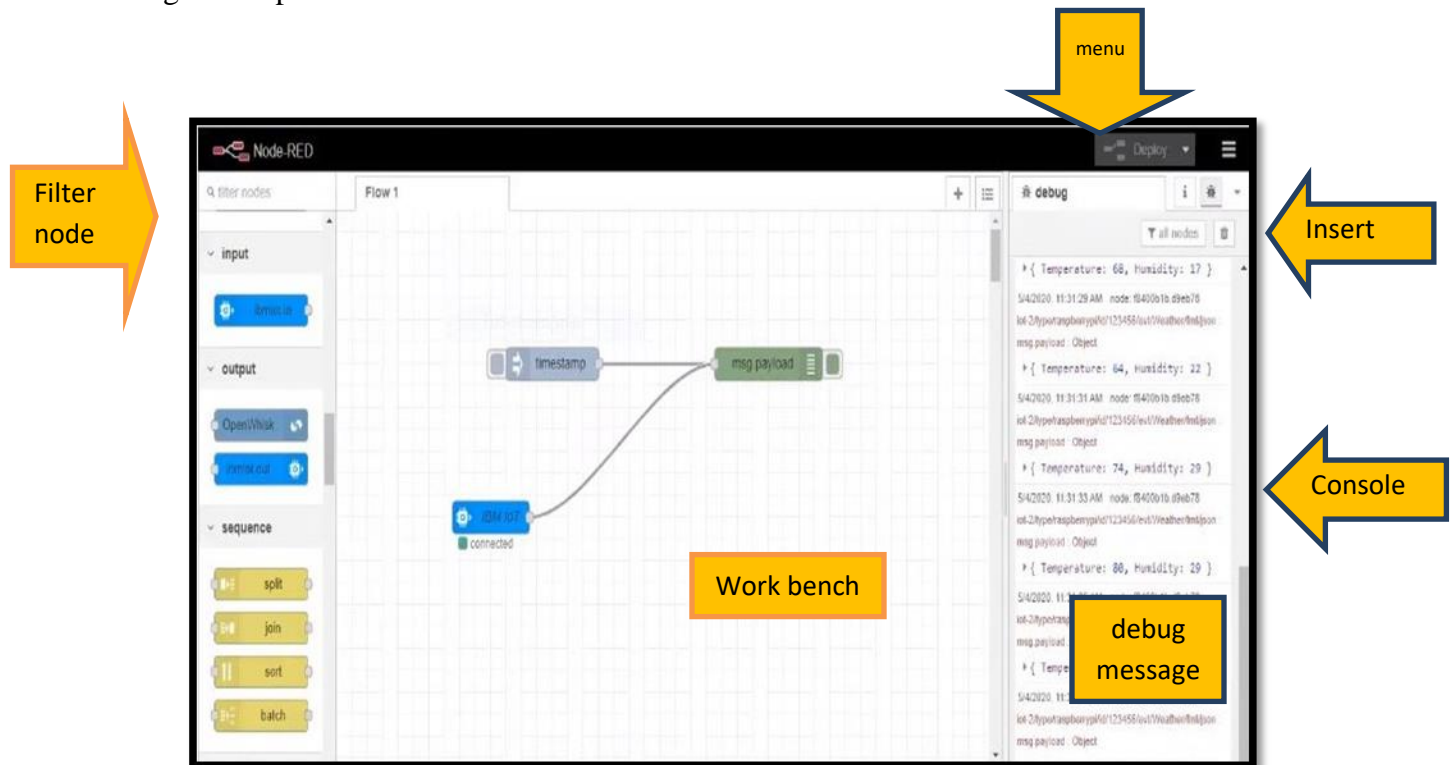
#### WHAT IS Nodered?

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

#### 3.1.0 Retrieving Data from IBM IoT Platform:

First fall open IBM cloud and there will open a dashboard page and click on catalog and there will see all the services of IBM. and click on catalog then left side we see software option click on it. and search for Node-red and there will see a Node-Red App then click on it. Node-Red page will be open and back end it operates under Cloudant database. Node-Red is been developed on Node. JS and click on create and give some necessary credentials in it. Then click on Next and duplicate the same page and come back to dashboard page then click on services in that click on internet of things platform then click on launch button. So, it will be launch IBM IoT platform. In node-red page click on IBM cloud and click on cloud foundry apps in that click on Node-Red and then click on Visit App URL. then welcome page will be open of Node-Red and click on next finally click on finish. And click on Go to Node-Red flow Editor then open the Node-Red tool page. in Node-Red left side we have filter nodes and middle page we have work bench and right side we have console and every time we have to click on Deploy to save the flow. Connect necessary nodes and click on Deploy to save the flow of the Node-Red, in right side of the corner we have two buttons one is Node information and other one is Debug

messages and then click on the button on work bench then we see the debug information on the right side of the debug message area. We have to Retrieving Data from IBM IoT Platform and we have installed some libraries in it. Then click on menu button and go to manage palette and click on install and search for IBM IoT and click on install and then click on again install. and left side we have seen IBM IoT nodes and drag the node to work bench and double tab on the IBM IoT and insert API KEY and API token in it and click on add and come back same page then add some credentials in it. and then connected to the IBM IoT platform and whatever data coming in IoT platform that data comes to node-red flow also.

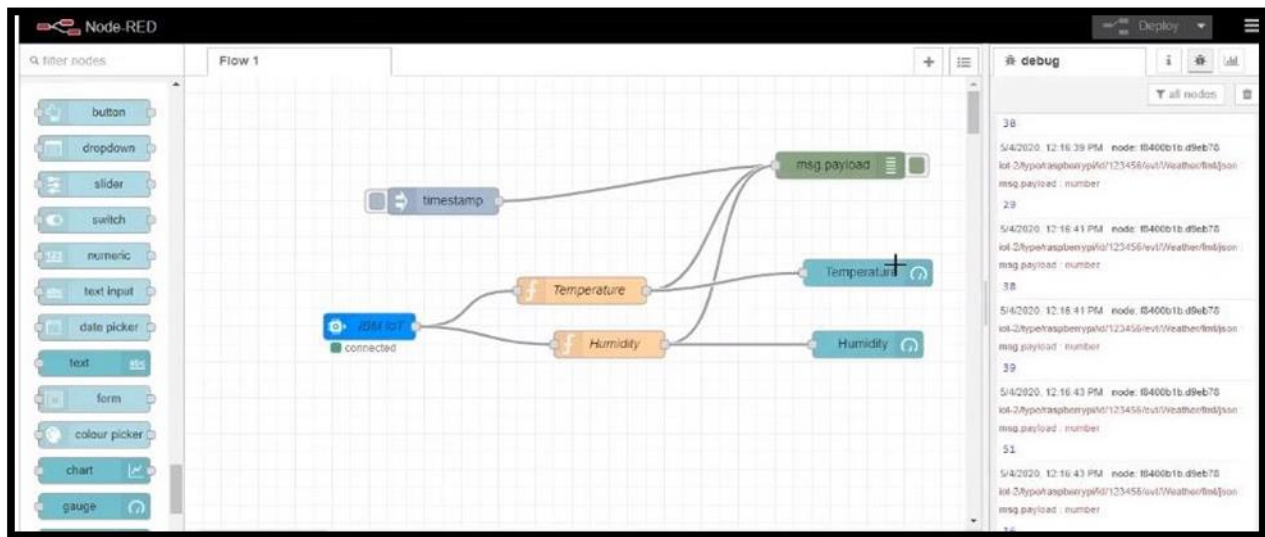
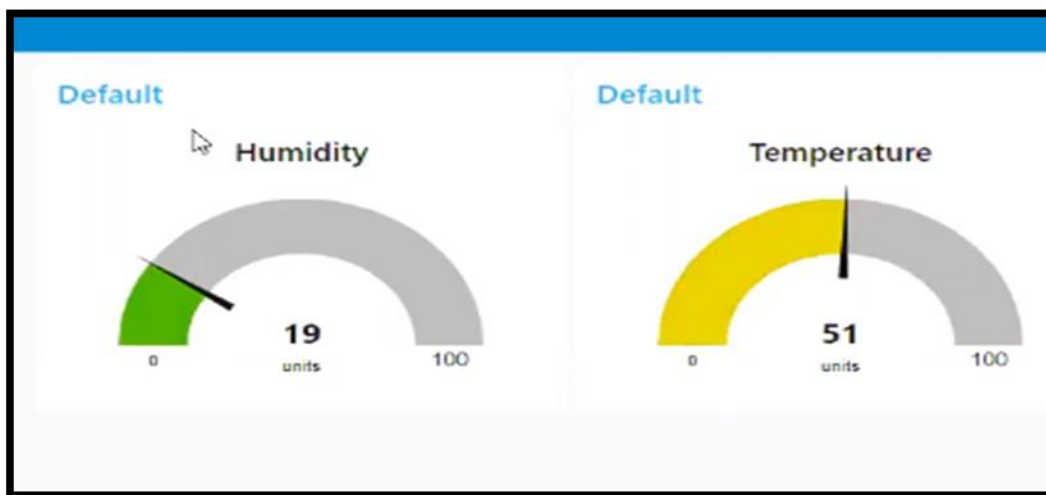


**Fig 28: Node-Red flow diagram**

### 3.1.1 Sending Command to the Device:

Sending commands to connected device by using IoT simulator. By adding additional functions to it, temperature and humidity functions to it to the message. Payload sees same values in the IoT simulator. Whatever information comes in the node-red. If we want to see the interface of UI then copy the node-red URL and open the new tab and paste URL in Web and add additional term UI in the web lastly and then press enter it is shown as figure.



**Fig 29: IoT Simulator****Fig 30: Sending command to device****Fig 31: Receiving data from web**

### 3.5 Watson Assistant using python code

What is Watson assistant?

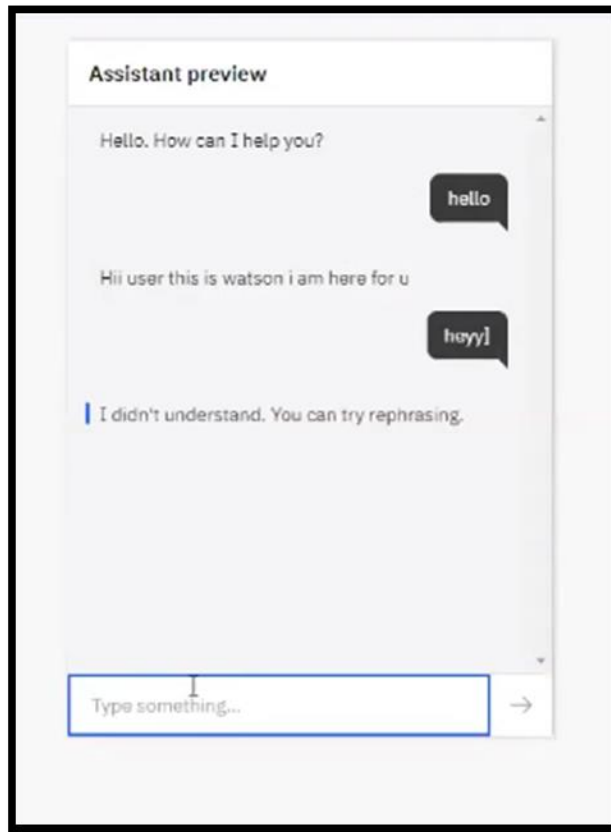
Watson Assistant is a conversation AI platform that helps you provide customers with fast, straightforward and accurate answers to their questions, across any application, device or channel. ... Watson Assistant is more than a Chabot.



**Fig 35: Watson assistant AI platform**

How to open Watson Assistant and work it?

First fall open IBM cloud and there will open a dashboard page there we will see a services button in service button we have Watson assistant click on it Watson assistant page will be open there we have some credentials in it copy them in one side and paste this credentials in the python code when this credentials needed and click on launch Watson assistant. There you see assistant page we want some libraries in it and download the libraries. we have to create a session to use Watson assistant every time and copy Watson assistant code and paste the code in python shell and paste the API keys and Watson assistant keys in it and run the code in python shell then we get session id copy that id in a notepad carefully. In API docs left side we have methods in methods we have messages in messages select send user input to assistant there we have a python code copy that code and paste the code in python shell and give necessary credentials to it, lastly, we have copied session id and place them in that code and run that python code successfully. We provide some skills to our Watson assistant then Watson assistant is trained successfully.



**Fig 36:** Watson assistant preview

## CHAPTER 04

### 4.0 Applications

#### 4.0.0 Smart Security Using Node-RedService

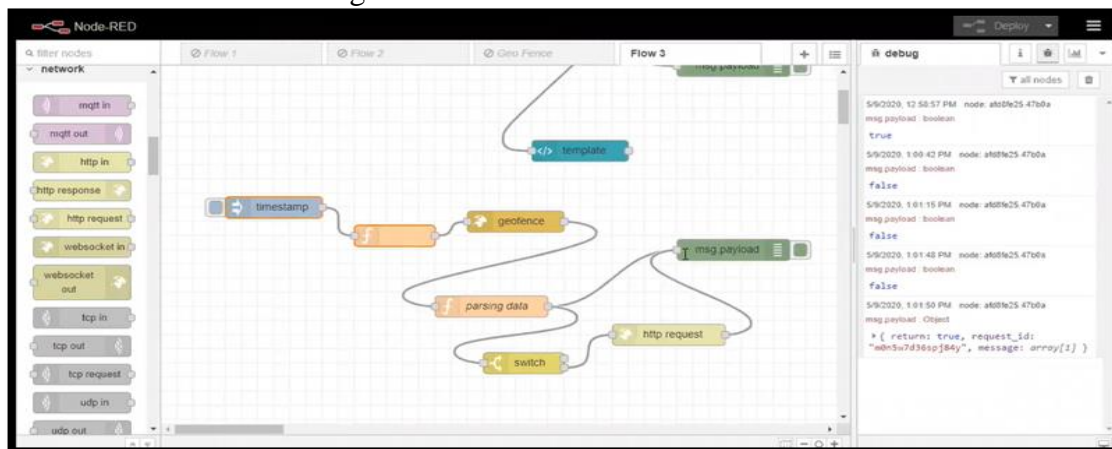
- Login to the IBM Cloud and open the Cloudant database service, Node-Red and object storage. Give the service credentials of the object storage.
- From the end point give the resiliency as regional, location as jp-tok, copy and paste the public URL.
- Give the service credentials to the Cloudant service to the python code. Give the database name =sample.
- Open the Fast2sms website and login. click on Dev API and then open read API documentation. Click on Bulk SMS API and click Get Method.
- Copy the URL and paste it new tab and give the API key from Fast2sms and give the mobile number.

#### 4.1.1 Smart Tracking (Geofence)

- Geofences are a great feature of the EyeQ mapping. By drawing virtual fences (geofences) around areas, you are able to manage when vehicles, assets or personnel have crossed the fence boundary and receive notifications, should they be needed as well.
- By looking at Geofence applications within your own organization, you can utilize improved visibility and reporting metrics to provide a more streamlined and automated process for management of the comings and goings of fleet and assets and the reporting of such.

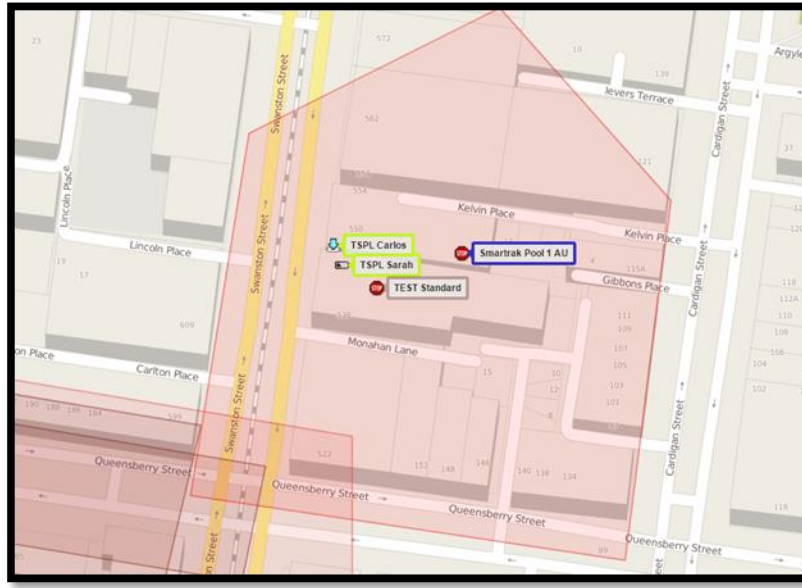
The following are some of the uses you might have for setting up a geofence:

1. Deport
2. Job & Work Sites
3. Restricted Areas
4. Parks/Regions
5. Disaster Zoning



**Figure 42:** Nodered Code for Smart Tracking with Geofence





**Figure 43: Smart Tracking with Geofence**

#### 4.2 IBM Cloud Functions

- IBM Cloud Functions is a distributed compute service that executes application logic in response to requests from web or mobile apps. You can set up specific actions to occur based on HTTP-based API requests from web apps or mobile apps, and from event-based requests from services like Cloudant. Functions can run your code snippets on demand or automatically in response to events.
- All APIs are protected with HTTP Basic authentication. You can use the wskadmin tool to generate a new namespace and authentication. The Basic authentication credentials are in the AUTH property in your ~/.wskprops file, delimited by a colon. You can also retrieve these credentials by using the CLI running ibmcloud wsk property get --auth.

For the {namespace} in the URL, the underscore character (\_) can be used to specify the user's default namespace.



## **CHAPTER 05**

### **PROJECT**

1. Purpose of the Project
2. Block Diagram
3. Hardware/Software Designing
4. Flowchart
5. Advantages & Disadvantages
6. Applications
7. Result
8. Conclusions

#### **5.1 Purpose of the Project**

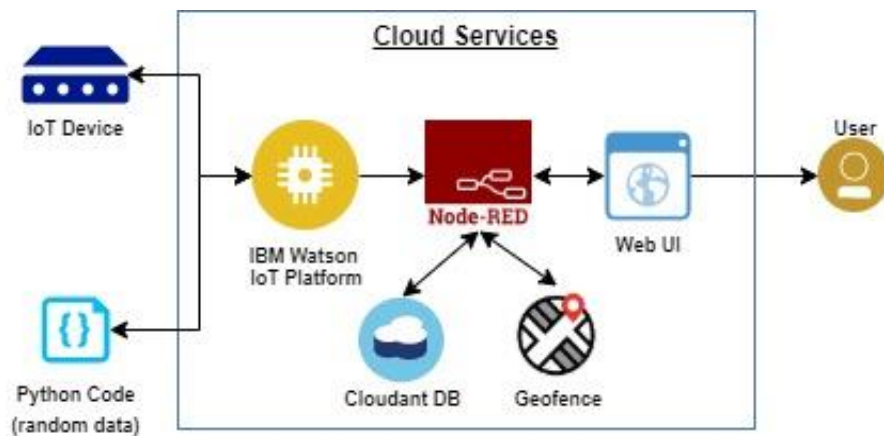
The proposed project is a system that keeps a track of employees Goals along with their salary and assigns goal completion bonus too. The system first allows admin to insert and update employee payroll that includes employee salary, name address and other data into the system. The system then manages this data effectively and also allows admin to assign goals or targets to employees. As soon as the employee achieves his/her goal the admin can see the goals assigned to an employee and mark that particular goal as completed. The system then looks for the stars or importance assigned to that particular goal. After this the system credits some bonus amount in that employee salary who completed that goal. The admin may later assign other goals to employees and system itself tracks those goals. Each employee is assigned a login where they may view their goals.

The main purpose of coming up with this project is to create an easy-to-use and efficient platform on which employers can easily monitor their employees daily work.

## 5.2 Proposed Solution

This Project examines and compares some IOT regression methods. Wireless sensor network makes the employee tracking in easy way. The location of an organization has a specific location, which can be determine by the Geofence. The location of each employee with their ID and time is updated to the cloud and stored in the database. There will be a provision for Admin to track the location of every employee at a particular zone in the Web app.

## 5.3 Block Diagram



## 5.4 Hardware/Software Designing

By using

- Python
- IOT Open Hardware Platforms
- IOT Application Development
- IOT Cloud Platform
- IOT Communication Technologies
- IOT Communication Protocols

### **Python Code:**

```
#IBM Watson IOT Platform

#pip install wiotp-sdk

import wiotp.sdk.device

import time

import random

myConfig = {
    "identity": {
        "orgId": "92xo80",
        "typeId": "node",
        "deviceId": "12357"
    },
    "auth": {
        "token": "employeeid12345"
    }
}

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)

client.connect()

while True:
```

```

emp_dic =
{'AngelBonnie':1780,'Frank':1781,'Joe':1782,'Kimberly':1783,'Lisa':1784,'Michael':1785,'Patrick'
:1786,'Rose':1787,'Todd':1788,'Roja':1789}

print(emp_dic )

empid=random.randint(1780,1789)

longitude = "78.5456505"

latitude = "17.4226372"

myData={'employeeid':empid,'longitude':longitude,'latitude':latitude}

enteremployeeid = int(input("Enter employee id : "))

if enteremployeeid in emp_dic.values():

    print("IT IS VALID EMPLOYEE ID ")

elif enteremployeeid not in emp_dic.values():

    print("INVALID EMPLOYEE ID ")

    print("TRY AGAIN ")

result=[new_k for new_k in emp_dic.items() if new_k[1] == enteremployeeid][0][0]

print("YOU CAN ENTER INTO THE ROOM WELCOME " +result)

client.publishEvent(eventId="status", msgFormat="json", data=myData , qos=0,
onPublish=None)

client.commandCallback = myCommandCallback

time.sleep(1)

client.disconnect()

```

## **5.5 Advantages & Disadvantages**

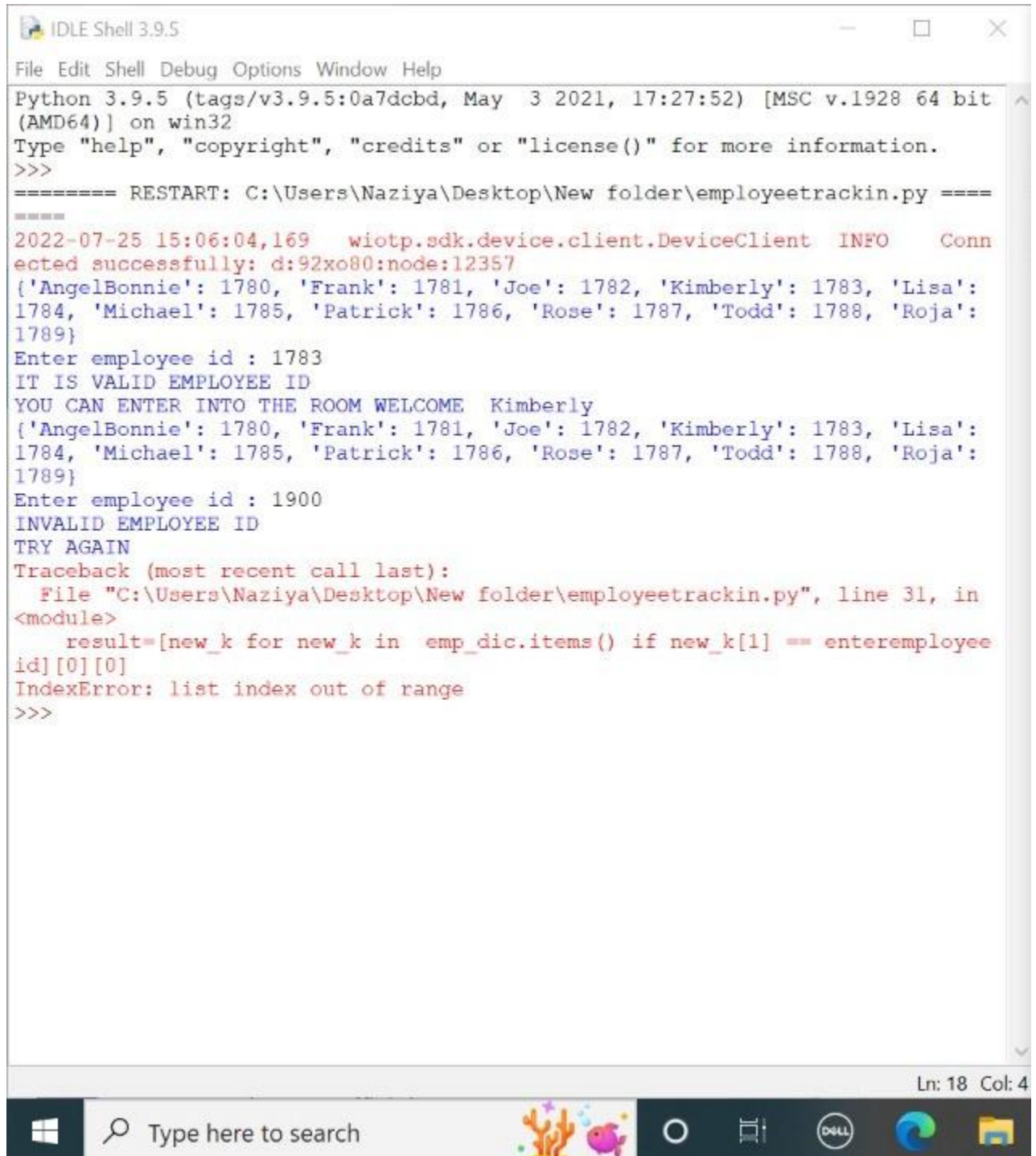
### **ADVANTAGES:**

1. This System helps the admin to keep track of the employee's who go for field work. Since the location of employee is tracked in that Geo-fence area.
2. It helps the employees by restricting them to not go to unsafe zones in the workplace.

### **DISADVANTAGES:**

1. By keen monitoring of the employees, it creates a bit pressure for the employees to work freely.
2. It is difficult to track the location of the employees for the admin, if the device (beacon) at the employee is working not properly.

## RESULTS

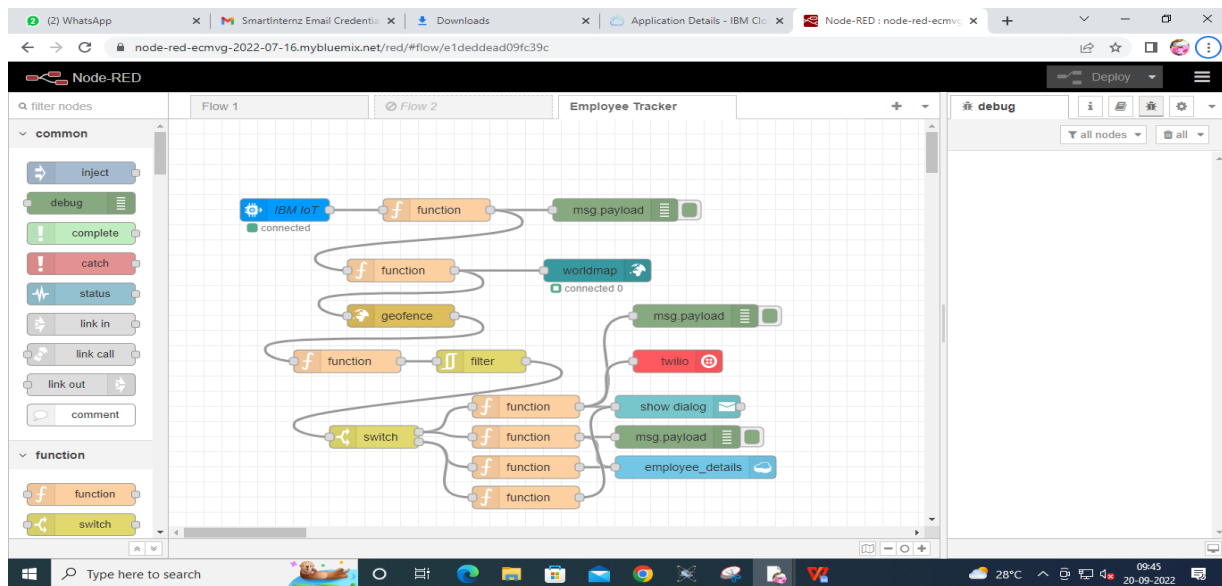
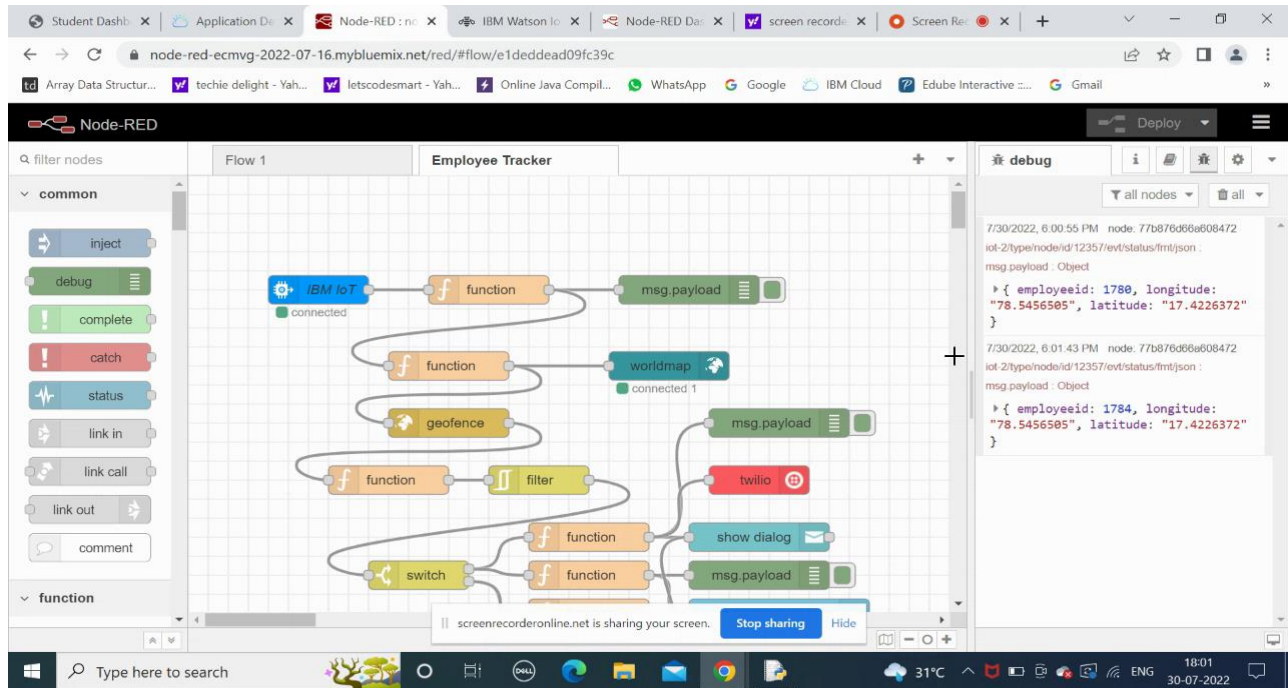


```

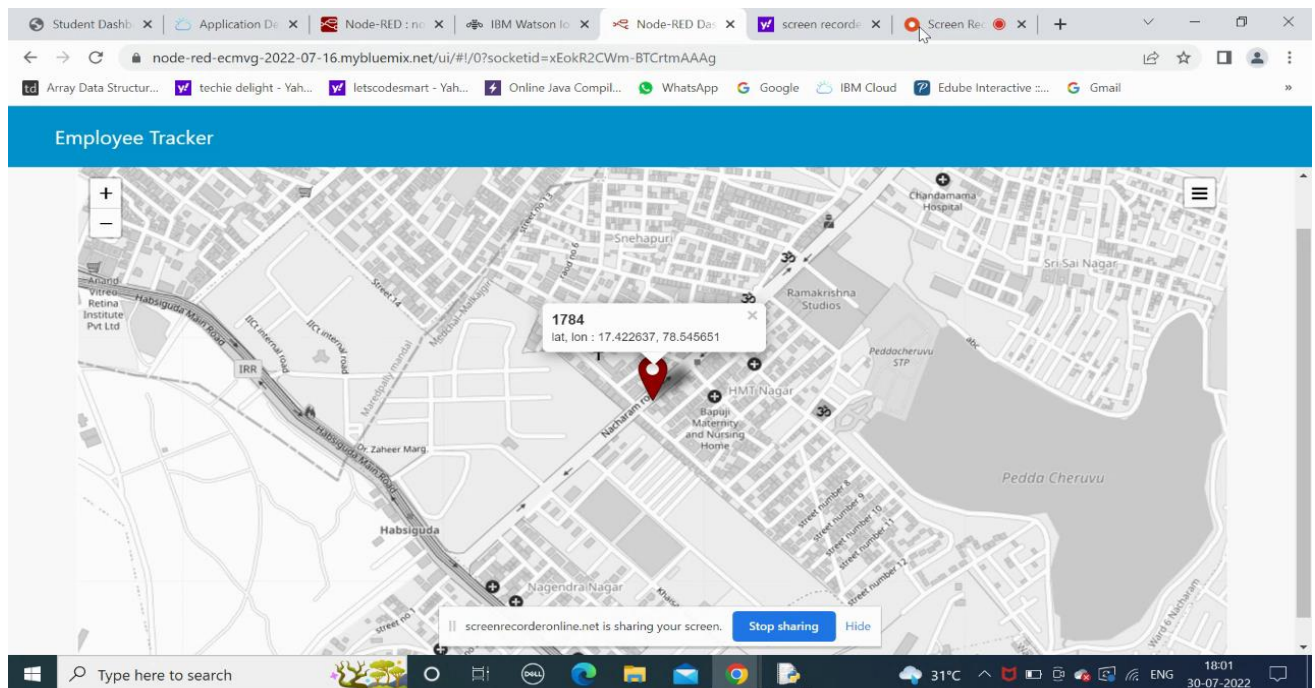
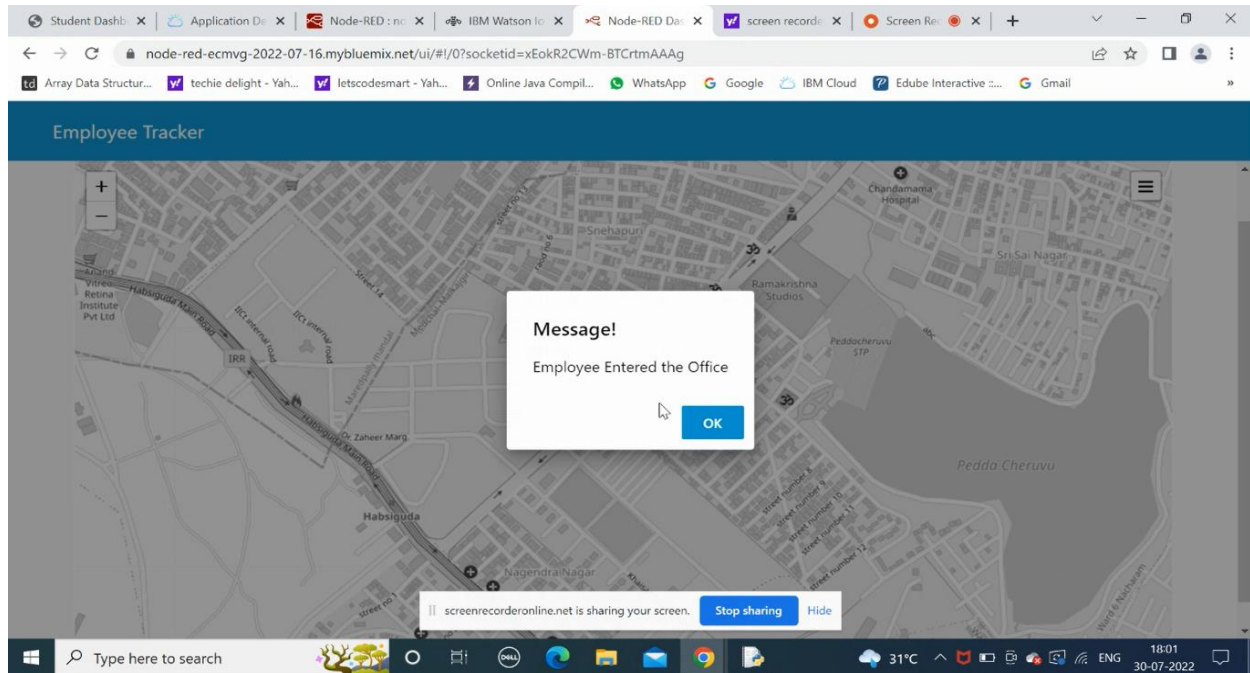
IDLE Shell 3.9.5
File Edit Shell Debug Options Window Help
Python 3.9.5 (tags/v3.9.5:0a7dcdb, May 3 2021, 17:27:52) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Naziya\Desktop\New folder\employeeetrackin.py =====
2022-07-25 15:06:04,169 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:92xo80:node:12357
{'AngelBonnie': 1780, 'Frank': 1781, 'Joe': 1782, 'Kimberly': 1783, 'Lisa': 1784, 'Michael': 1785, 'Patrick': 1786, 'Rose': 1787, 'Todd': 1788, 'Roja': 1789}
Enter employee id : 1783
IT IS VALID EMPLOYEE ID
YOU CAN ENTER INTO THE ROOM WELCOME Kimberly
{'AngelBonnie': 1780, 'Frank': 1781, 'Joe': 1782, 'Kimberly': 1783, 'Lisa': 1784, 'Michael': 1785, 'Patrick': 1786, 'Rose': 1787, 'Todd': 1788, 'Roja': 1789}
Enter employee id : 1900
INVALID EMPLOYEE ID
TRY AGAIN
Traceback (most recent call last):
  File "C:\Users\Naziya\Desktop\New folder\employeeetrackin.py", line 31, in <module>
    result=[new_k for new_k in emp_dic.items() if new_k[1] == enteremployeeid][0][0]
IndexError: list index out of range
>>>
Ln: 18 Col: 4

```

## Node-Red:



## Node-RedUI:





## **CHAPTER 06**

### **6.0 Conclusion**

In conclusion, I am well satisfied with my training. I have learned many new concepts, acquired a number of new technical skills and improved another group of existing skills. What I liked most about my training is that it is very strongly related to new emerging technology. This refutes the common saying that very little of the materials taught in university engineering courses is used by engineers working in the labor market. This dependency (relationship) is clearest in engineering design. I may count the technical skills that I learned or improved at the training site, other than those gained at college. At last, I hereby conclude that I have successfully completed my industrial training in Smart Bridge and gained knowledge.