PROJECT REPORT

TITLE: SMART PARKING SYSTEM FOR SMART CITIES

SUBMITTED BY:

KARTHIK BHARADWAJ R M

1. INTRODUCTION

1.1 Overview:

Smart Parking System in Smart Cities project is about a new parking system called Smart Parking System. This is proposed to assist which enables the user to find the nearest parking area. And gives availability of parking slots in that respective parking area and it mainly focuses on reducing the time in finding the parking lots. It also avoids the unnecessary traveling through filled parking lots in a parking area. We can check the status of the parking slot by using sensors. With the advent of Internet of Things, the concept of smart cities can be readily achievable. An extensive research is ongoing in the field of Internet of Things to increase the quality of services. It offered in cities and to improve the productivity and reliability of urban infrastructure.

IoT is addressing the most common problems faced in cities like availability of car parking and traffic jams. This paper presents an Internet of Things based Parking system for Smart Cities. The proposed parking system contains an IoT module deployed on-site for managing the available parking spaces. A platform provided in the form of portal for booking the parking spaces.

With increase in the population of the vehicles in metropolitan cities, road congestion is the major problem that is being faced. The aim of this system is to resolve this issue. In this project, IoT based cloud integrated smart parking system has been developed. This system monitors and signalizes the state of availability of each single parking space. In this system, entry and exit gates are automated using IR sensors and servo motor. The occupancy of parking slot is monitored using Ultrasonic sensor and status of parking slots is displayed on the screen at the time of entry. Customers can also check the availability of slots using the web application which has been developed using node red. Node-MCU is used as central device for controlling and collecting data from sensors and then it is connected to IBM cloud platform using MQTT protocol.

1.2 Purpose:

The concept of Smart Cities have always been a dream for humanity. Since the past couple of years large advancements have been made in making smart cities a reality. The growth of IoT and Cloud technologies have given rise to new possibilities in terms of smart cities. Smart parking facilities and traffic management systems have always been at the core of constructing smart cities. The system that we propose provides real time information regarding availability of parking slots in a parking area. Users from remote locations could book a parking slot for them by the use of our mobile application. In this current era of modern world, almost everyone owns a personal vehicle and it has become a basic need for the humans. Hence, it has been proven statistically that the usage of vehicles is increasing rapidly yearly. Due to the growth, it is very difficult to find parking slots in cities, especially during the peak time. This creates a necessity to introduce an automated system that allows users to look for the available parking spaces in the city with a few clicks through a web Application. This serves to hassle free situation for each and every user. The main motivation behind the Smart Parking System is to help the drivers to find where parking is available in that area. The system works primarily on the detection of parking slots through sensors that are mounted on every parking slots which facilitates the information. This is then processed by Node-MCU which helps to serve as a medium of communication between those peripherals or devices. Data is then sent to IBM IoT platform and finally, we can check availability using web application.

2. LITERATURE SURVEY

2.1 Existing Problem:

Traffic management and car parking on modern cities continues to be a problem both for citizens and for city officials. The increasing number of vehicles flowing into the city drain the existing scarce parking resources, and the increase in time spent looking for a parking spot leads to more congestions, parasitic traffic, whilst augmenting fuel consumption and air pollution. Due to rapid economic and population growth, finding a parking space is a routine activity and it is estimated that nearly 30% of urban congestion is created by drivers cruising for parking space, according to ITS America's Market Analysis. Also resulting in oil wastage, almost one million barrels of world's oil every day. Many cities are suffering from lacking of car parking areas with imbalance between parking supply and demand. This imbalance is partially due to ineffective land use planning and miscalculations of space requirements during first stages of planning. This ever growing traffic congestion and uncertainty in the parking availability and payment have thus enforced the need for a Smart Parking systems.

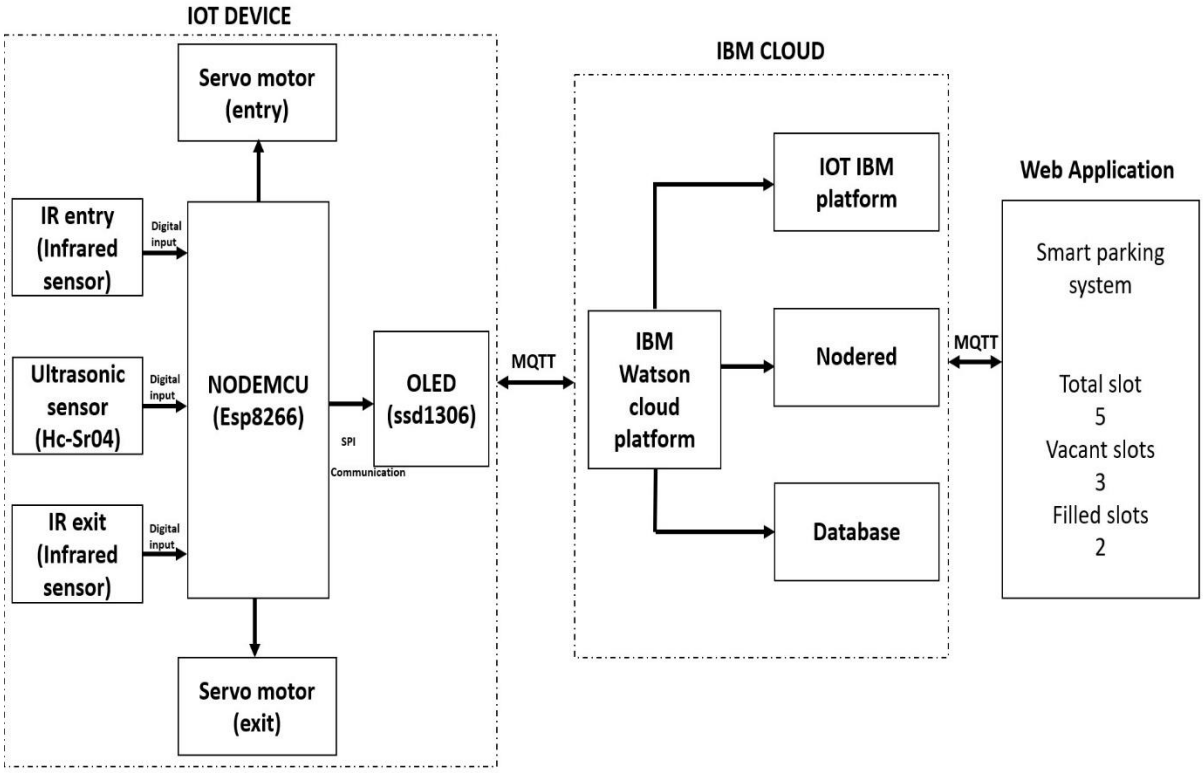
2.2 Proposed Solution:

This project proposes a smart car parking system that will assist users to solve the issue of finding a parking space and to minimise the time spent in searching for the nearest available car park. In addition, it provides users with roads traffic congestion status. The existing parking problems create a necessity to introduce an automated smart parking system. Smart parking technology that will help optimize parking space usage, improve the efficiency of the parking operations and help smoother traffic flow. In this work, a smart parking system is developed. The proposed system works primarily on the detection of parking slots through sensors that are mounted on every parking slots which facilitates the information. This system consists of three parts. First is the interfacing and development of the sensor which

are deployed in the parking spaces. Data from sensors is then processed by NodeMCU(esp8266) which helps to serve as a medium of communication between those peripherals or devices. The occupancy of slots is thus checked. The second is IBM IOT Cloud. The total number of slots available is sent to IBM IoT cloud which is connected to node red. The third is web application, which is developed using node red. It fetches the data from IBM IoT cloud and display it in web application using UI nodes. The user can check if the parking space is available or not through web application.

3. THEORITICAL ANALYSIS

3.1 Block Diagram:



3.2 Hardware and Software Designing

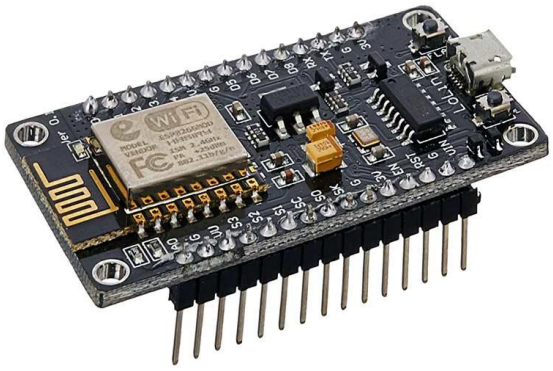
Hardware Components used:

- NodeMCU (ESP8266)
- IR Sensor
- Ultrasonic Sensor
- Servo Motor

The entry and exit gates are opened when vehicle is present. The presence of vehicles is checked using IR sensors and servo motor opens the gate. Ultrasonic sensors check the availability of slots and send data NodeMCU which sends it to cloud and it is finally displayed at the entrance and exit gates through web application.

NodeMCU (ESP8266):

NodeMCU is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.



IR Sensor:

An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. There are two types of infrared sensors: active and passive. Active infrared sensors both emit and detect infrared radiation. Active IR sensors have two parts: a light emitting diode (LED) and a receiver. When an object comes close to the sensor, the infrared light from the LED reflects off of the object and is detected by the receiver. Active IR sensors act as proximity sensors, and they are commonly used in obstacle detection systems.

Ultrasonic Sensor:

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).

Servo Motor:

A servo motor is a rotary actuator or a motor that allows for a precise control in terms of the angular position, acceleration, and velocity. Basically it has certain capabilities that a regular motor does not have. Consequently it makes use of a regular motor and pairs it with a sensor for position feedback. A typical servo motor comprises of three wires namely- power, control, and ground. The shape and size of these motors depends on their applications.

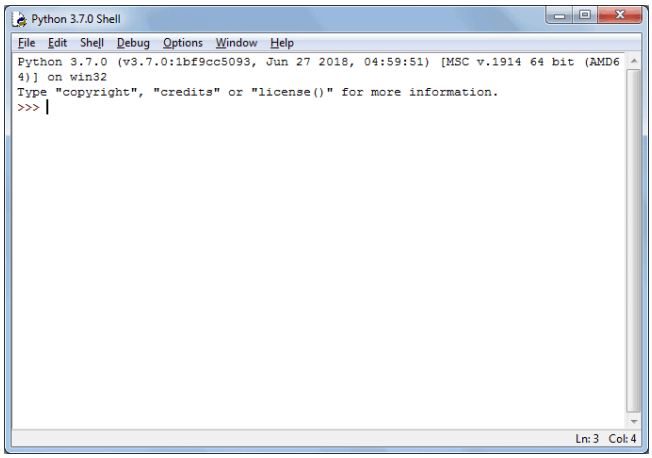
Software Used:

- Python Shell
- IBM Cloud
- Node-RED

IDLE is used to write the code in python. IBM cloud is used to fetch commands and send data. The cloud server acts as a mediator between the modules. The cloud server is connected to the Wi-Fi module. The Node-Red fetches data from IBM IOT cloud platform and dashboard nodes are used to build the web application. The parking space information is displayed at the entrance and exit gates. Total number of vehicles entered and exited are also calculated.

Python Shell:

Python provides a Python Shell (also known as Python Interactive Shell) which is used to execute a single python command and get the result. Python Shell waits for the input command from the user.
IDLE is used to write the code in python.



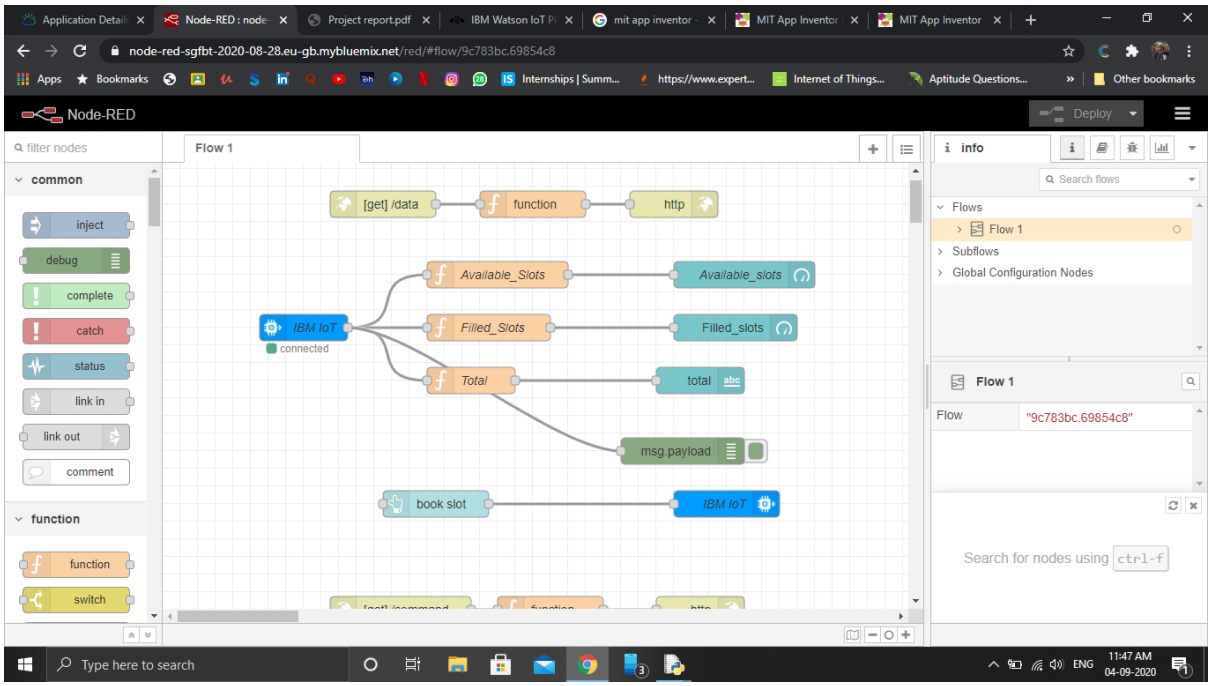
IBM Cloud:

IBM Cloud is a suite of cloud computing services from IBM that offers both Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). With IBM Cloud IaaS, organisations can deploy and access virtualized IT resources such as compute power, storage and networking- over the internet.

Node-RED:

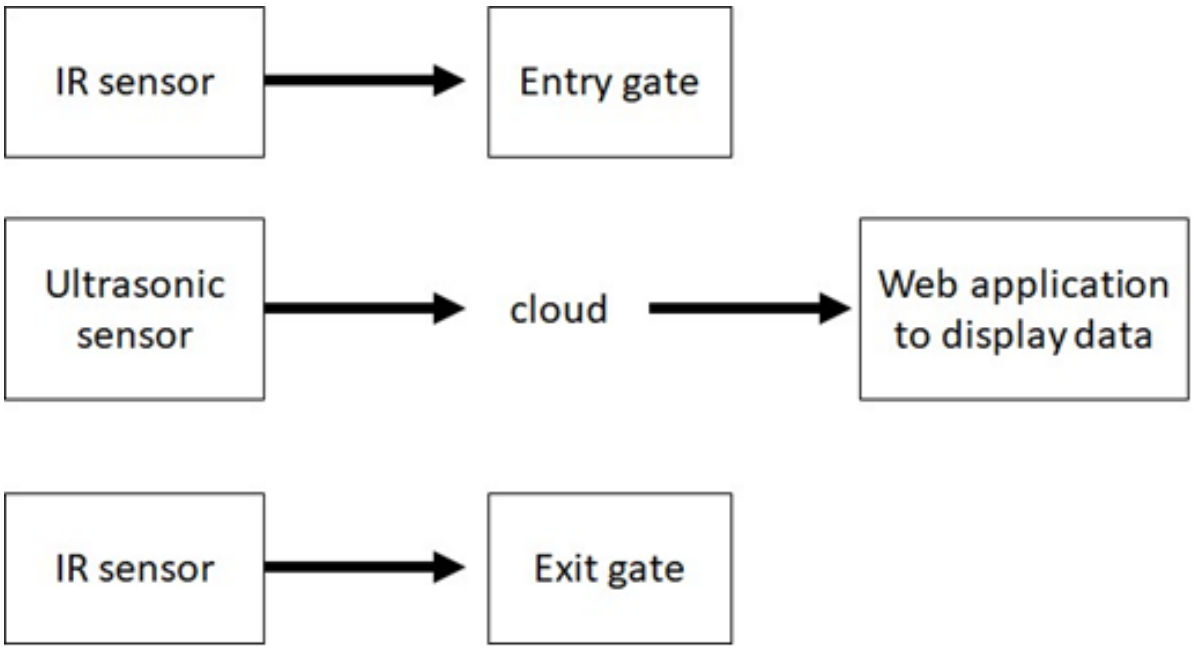
NodeRED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click. JavaScript functions can be created within the editor using a rich text editor. A built-in library allows you to save useful functions, templates or flows for re-use. The flows created in Node-RED are stored using JSON which can be easily imported and exported for sharing with others.



4. EXPERIMENTAL INVESTIGATION

The proposed system is based on the observation of existing system of parking. With increase in the population of the vehicles in metropolitan cities, road congestion is the major problem that is being faced. This system is efficient and smart way to automate the management of the parking system that allocates an efficient parking space using internet of things technology. The IoT provides a wireless access to the system and the user can keep a track of the availability of the parking area. This system uses IR sensors to detect vehicles and servo motor is used to open and close the gate. Ultrasonic sensors are used to detect the presence of vehicles. Node-MCU sends data to cloud and then node-red is used to display data on web application at entrance and exit gates.



The user usually wastes his time and efforts in search of the availability of the free space in a specified parking area. The parking information is sent to the user via notification. Thus, the waiting time for the user in search of parking space is minimised. The problems of traffic congestion, parking on the streets etc. can be avoided using this system. Results of this system are described below. With few advancements ,this system can be easily deployed anywhere.

The IoT based Cold Smart Parking system is structured as below:

- **Setup Environment:**

1. Create an IBM Account
2. Create a Node-Red Application
3. Create An IBM Watson IoT Platform

- **Setup Hardware and Develop the Code:**

1. Code Snippet for publishing data using MQTT Communication

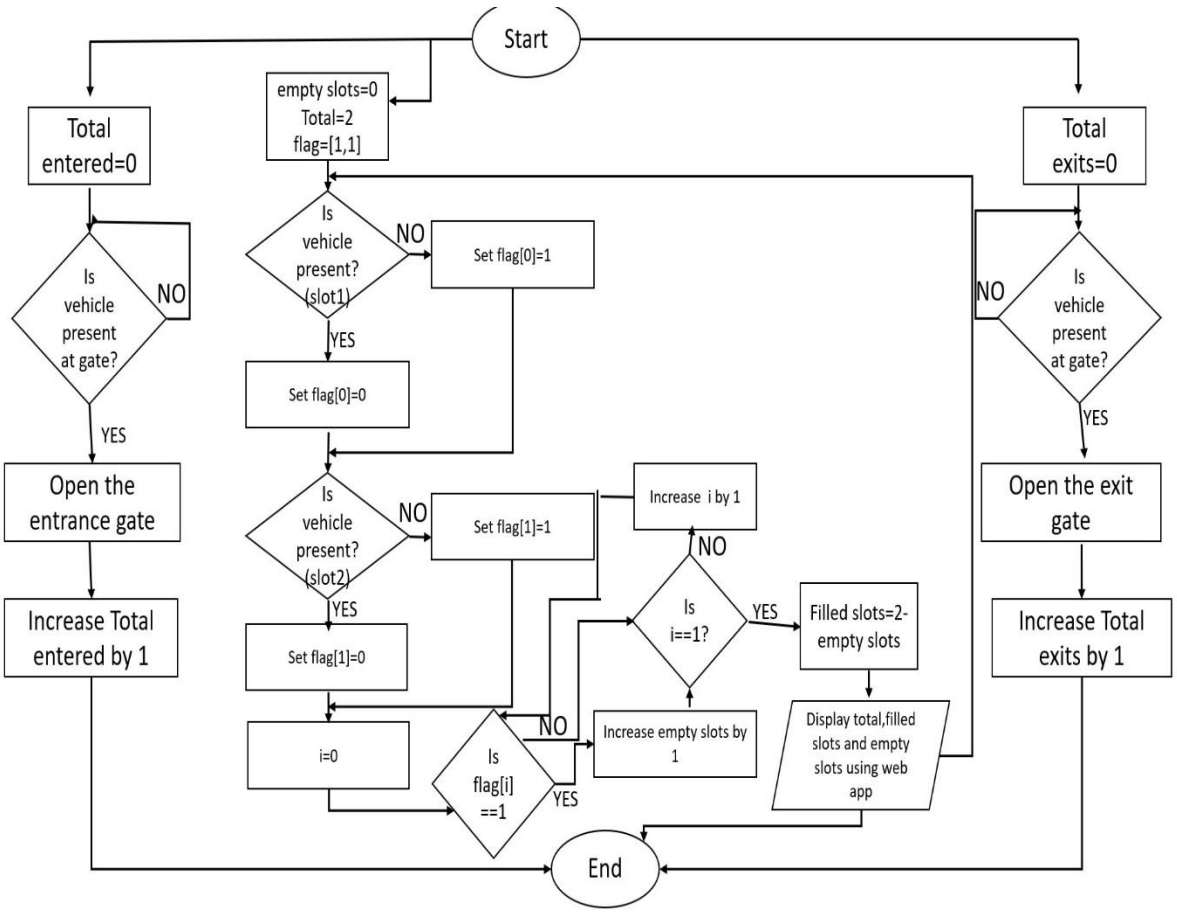
- **Building a Web:**

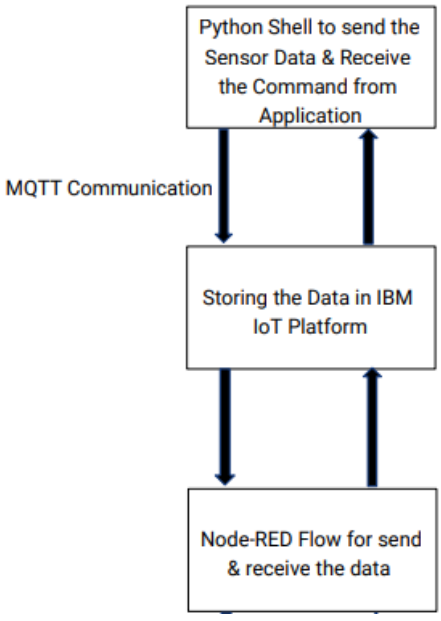
1. Create a node-red flow to get data from the device

- **Use dashboard nodes for creating UI (Web App):**

1. UI Nodes Installation

5. FLOWCHART


Start
Total entered=0
empty slots=0
Total=2
flag=[1,1]
Total exits=0
Is vehicle present at gate?
NO
YES
Open the entrance gate
Increase Total entered by 1
Is vehicle present? (slot1)
Set flag[0]=1
Set flag[0]=0
Is vehicle present? (slot2)
Set flag[1]=1
Set flag[1]=0
i=0
Is flag[i] ==1
Increase i by 1
Is i==1?
Filled slots=2-empty slots
Increase empty slots by 1
Display total,filled slots and empty slots using web app
Open the exit gate
Increase Total exits by 1
End


Python Shell to send the Sensor Data & Receive the Command from Application
MQTT Communication
Storing the Data in IBM IoT Platform
Node-RED Flow for send & receive the data

6. RESULT

The Smart Parking System for Smart cities using IoT was successfully built where the data was sent using the MQTT communication protocol to the IBM IoT Platform Cloud then to the Node-RED flow, from Node-RED using HTTP Communication protocol to the Application where the user can monitor the parking using the Mobile Application. The demand of smart parking system is increasing significantly. The existing system in today's world doesn't contains the facilities of parking reservation and parking slot availability checker. The existing system is vision-based monitoring system which estimates the number of the parking slots available in the area by counting the number of incoming and outgoing cars which consumes lot of time and efforts. This system allows user to involve real time access of the availability of the parking space. The result of the system is that this system makes the parking area connected with the real world. The development of a prototype of the internet based smart parking system works well in detecting, processing parking data, and sending parking data to the cloud. This system will solve the present issues of not finding parking spaces, pollution, fuel wastage etc. This system enhances user experience and with added features like reservation etc., this system can be very effective.

7. ADVANTAGES AND DISADVANTAGES

Advantages:

- Smart parking will reduce search traffic on the streets.
- Smart Parking takes away the unpredictability of finding a parking spot
- Smart parking reduces stress while searching for a parking space
- Optimised parking
- Lowering individual environmental footprint
- Saves time
- Saves money
- Less fuel is wasted
- Reduced pollution
- Decreased management costs
- New Revenue Streams
- Enhanced user experience
- There is high parking efficiency.
- There is no need for driving while looking for an available space.
- Emissions are greatly brought down and reduced.
- The patrons wait for their car in a highly controlled environment.
- There are less chances for vehicle vandalism.
- There is a minimal staff requirement if it is used by known parkers.
- There is a greater sense of security

Disadvantages:

- Initial installation cost of system is high
- Continuous Power supply is required for monitoring
- Damaged sensors or motor require immediate replacement
- Maintenance is required
- More automated things might lead to unemployment
- Ultrasonic sensors do not work well under a cover plate
- Use of redundant systems will result in a greater cost.
- It may be a bit confusing for unfamiliar users.

- It is not recommended for high peak hour volume facilities.
- There may be a fear of breakdown.
- There is an uncertain building department review and approval process.
- It requires a maintenance contract with the supplier

8. APPLICATIONS

Smart Parking involves the use of low cost sensors, real-time data and applications that allow users to monitor available and unavailable parking spots. The goal is to automate and decrease time spent manually searching for the optimal parking floor, spot and even lot. Some solutions will encompass a complete suite of services such as online payments, parking time notifications and even car searching functionalities for very large lots. A parking solution can greatly benefit both the user and the lot owner. Here are some of the top benefits:

- Optimised Parking
- Reduced traffic
- Reduced pollution
- Enhanced User Experience
- New Revenue Streams
- Increased Safety
- Decreased Management Costs
- Real Time Data and Trend Insight
- Integrated Payments
- Increased Service and Brand Image
- It is very beneficial in crowded cities
- Drivers can know in advance about the availability of parking spaces.
- Smart parking system can be used in universities, organizations, companies, shopping malls, theatres etc.
- This system is useful when there is going to be a concert, cricket

match etc, where a lot of vehicles are expected.

- This system can be used to Foresee the flow of vehicles by analysing parking routines in malls, business stores, airports.

9. CONCLUSION

The implementation of the smart parking system being presented, its efficiency in alleviating the traffic problem that arises especially in the city area where traffic congestion and the insufficient parking spaces are undeniable. It does so by directing patrons and optimizing the use of parking spaces. The system benefit of smart parking go well beyond avoiding time wasting. Developing smart parking system within the city also solves the pollution problem. This system is to ease the drivers to find parking slots during peak hours by using web Application. This is an efficient system as it helps to solve heavy traffic congestion and reduces the driver's frustrations. In future we would develop application for iOS and also with virtual reality and test its workability in a real time environment. We infer that our future work would facilitate parking issues and decrease traffic congestion and pollution created by the search for parking. So it is recommended to implement Smart Parking System in Smart Cities.

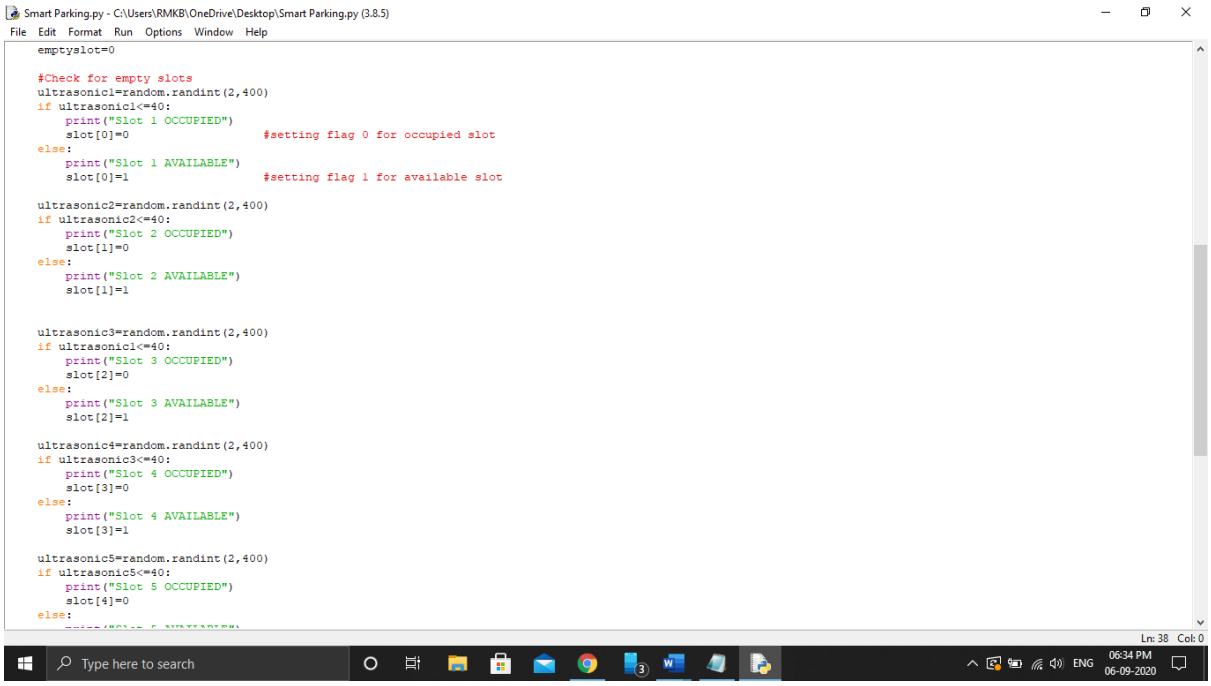
10. FUTURE SCOPE

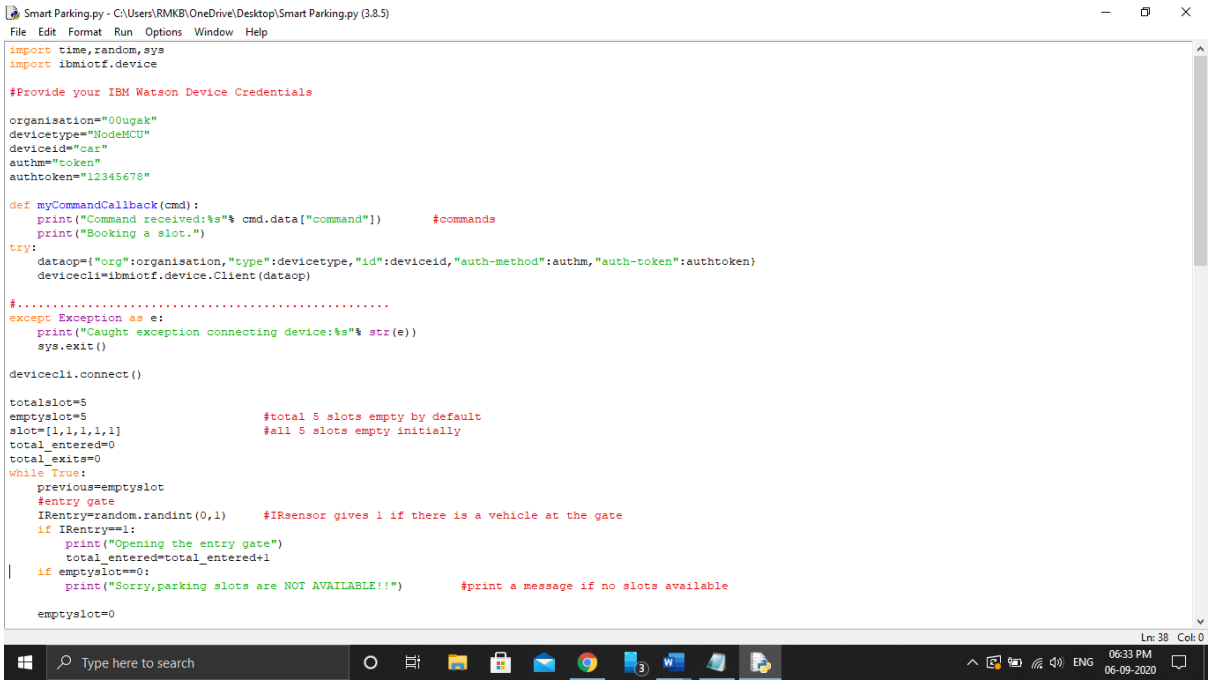
The smart parking industry continues to evolve as an increasing number of cities struggle with traffic congestion and inadequate parking availability. While the deployment of sensor technologies continues to be core to the development of smart parking, a wide variety of other technology innovations are also enabling more adaptable systems—including cameras, wireless communications, data analytics, induction loops, smart parking meters, and advanced algorithms. As for the future work the users can book a parking space from a remote location. GPS, reservation facilities and license plate scanner can be included in the future. The system can be more enhanced by providing the route to the selected parking location with the help of Global Position Search (GPS) System. We can also add features like advanced payment. A mobile application can also be built. We can use robotic vale to park the vehicles. We can notify users via SMS through GSM module.

11. BIBLIOGRAPHY

- https://www.researchgate.net/publication/329686583_IoT_Based_Smart_Parking_System
- https://iotdesignpro.com/projects/iot-based-smart-parking-using-esp8266
- https://www.digiteum.com/iot-smart-parking-solutions
- www.google.com
- www.wikipedia.com
- Paper: International Journal of Engineering and Advanced Technology

APPENDIX:


Smart Parking.py - C:\Users\RMKB\OneDrive\Desktop\Smart Parking.py (3.8.5)
File Edit Format Run Options Window Help
emptyslot=0
#Check for empty slots
ultrasonic1=random.randint(2,400)
if ultrasonic1<=40:
print("Slot 1 OCCUPIED")
slot[0]=0 #setting flag 0 for occupied slot
else:
print("Slot 1 AVAILABLE")
slot[0]=1 #setting flag 1 for available slot
ultrasonic2=random.randint(2,400)
if ultrasonic2<=40:
print("Slot 2 OCCUPIED")
slot[1]=0
else:
print("Slot 2 AVAILABLE")
slot[1]=1
ultrasonic3=random.randint(2,400)
if ultrasonic1<=40:
print("Slot 3 OCCUPIED")
slot[2]=0
else:
print("Slot 3 AVAILABLE")
slot[2]=1
ultrasonic4=random.randint(2,400)
if ultrasonic3<=40:
print("Slot 4 OCCUPIED")
slot[3]=0
else:
print("Slot 4 AVAILABLE")
slot[3]=1
ultrasonic5=random.randint(2,400)
if ultrasonic5<=40:
print("Slot 5 OCCUPIED")
slot[4]=0
else:
Ln: 38 Col: 0
Type here to search
ENG
06:34 PM
06-09-2020


Smart Parking.py - C:\Users\RMKB\OneDrive\Desktop\Smart Parking.py (3.8.5)
File Edit Format Run Options Window Help
import time,random,sys
import ibmiotf.device
#Provide your IBM Watson Device Credentials
organisation="00ugak"
devicetype="NodeMCU"
deviceid="car"
authm="token"
authtoken="12345678"
def myCommandCallback(cmd):
print("Command received:%s"% cmd.data["command"]) #commands
print("Booking a slot.")
try:
dataop={"org":organisation,"type":devicetype,"id":deviceid,"auth-method":authm,"auth-token":authtoken}
devicecli=ibmiotf.device.Client(dataop)
#.....
except Exception as e:
print("Caught exception connecting device:%s"% str(e))
sys.exit()
devicecli.connect()
totalslot=5
emptyslot=5 #total 5 slots empty by default
slot=[1,1,1,1,1] #all 5 slots empty initially
total_entered=0
total_exits=0
while True:
previous=emptyslot
#entry gate
IRentry=random.randint(0,1) #IRsensor gives 1 if there is a vehicle at the gate
if IRentry==1:
print("Opening the entry gate")
total_entered=total_entered+1
if emptyslot==0:
print("Sorry,parking slots are NOT AVAILABLE!!") #print a message if no slots available
emptyslot=0
Ln: 38 Col: 0
Type here to search
ENG
06:33 PM
06-09-2020

Smart Parking.py - C:\Users\RMKB\OneDrive\Desktop\Smart Parking.py (3.8.5)

File Edit Format Run Options Window Help

```
        slot[3]=1

    ultrasonic5=random.randint(2,400)
    if ultrasonic5<=40:
        print("Slot 5 OCCUPIED")
        slot[4]=0
    else:
        print("Slot 5 AVAILABLE")
        slot[4]=1



    for i in slot:
        if i==1:
            emptyslot=emptyslot+1

    data={"available_slots":emptyslot,"filled_slots":5-emptyslot,"total":totalslot}
    def myonpublishcallback():
        print("Number of vacant slots are %s"% emptyslot)

    success=devicecli.publishEvent("slots info","json",data,qos=1,on_publish=myonpublishcallback())       #publishing data to cloud

    if not success:
        print("Not connected to IOTf.")


         #exit gate
    IRexit=random.randint(0,1)
    if IRexit==1:
        print("Opening the exit gate.THANK YOU for visiting!!")
        total_exits==total_exits+1

    time.sleep(5)                       #checks status every 5 seconds
    devicecli.commandCallback=myCommandCallback

print(total_entered)
print(total_exits)

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

Ln: 38 Col: 0

Type here to search

ENG 06:34 PM 06-09-2020

Smart Parking.py - C:/Users/RMKB/AppData/Local/Programs/Python/Python38-32/Smart Parking.py (3.8.5)

File Edit Format Run Options Window Help

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

#Provide your IBM Watson Device Credentials

organisation="00ugak"
devicetype="NodeMCU"
deviceid="car"
authm="token"
authtoken="12345678"

def myCommandCallback(cmd):
    print("Command received:%s"% cmd.data["command"])       #commands
    print("Booking a slot.")
try:
    dataop={"org":organisation,"type":devicetype,"id":deviceid,"auth-method
    devicecli=ibmiotf.device.Client(dataop)

#.....................................................
except Exception as e:
    print("Caught exception connecting device:%s"% str(e))
    sys.exit()

devicecli.connect()

totalslot=5
emptyslot=5                         #total 5 slots empty by default
slot=[1,1,1,1,1]                    #all 5 slots empty initially
total_entered=0
total_exits=0
while True:
    previous=emptyslot
    #entry gate
    IRentry=random.randint(0,1)     #IRsensor gives 1 if there is a vehicle
    if IRentry==1:
        print("Opening the entry gate")
        total_entered=total_entered+1
    if emptyslot==0:
        print("Sorry,parking slots are NOT AVAILABLE!!")        #print a me

    emptyslot=0
```

Python 3.8.5 Shell

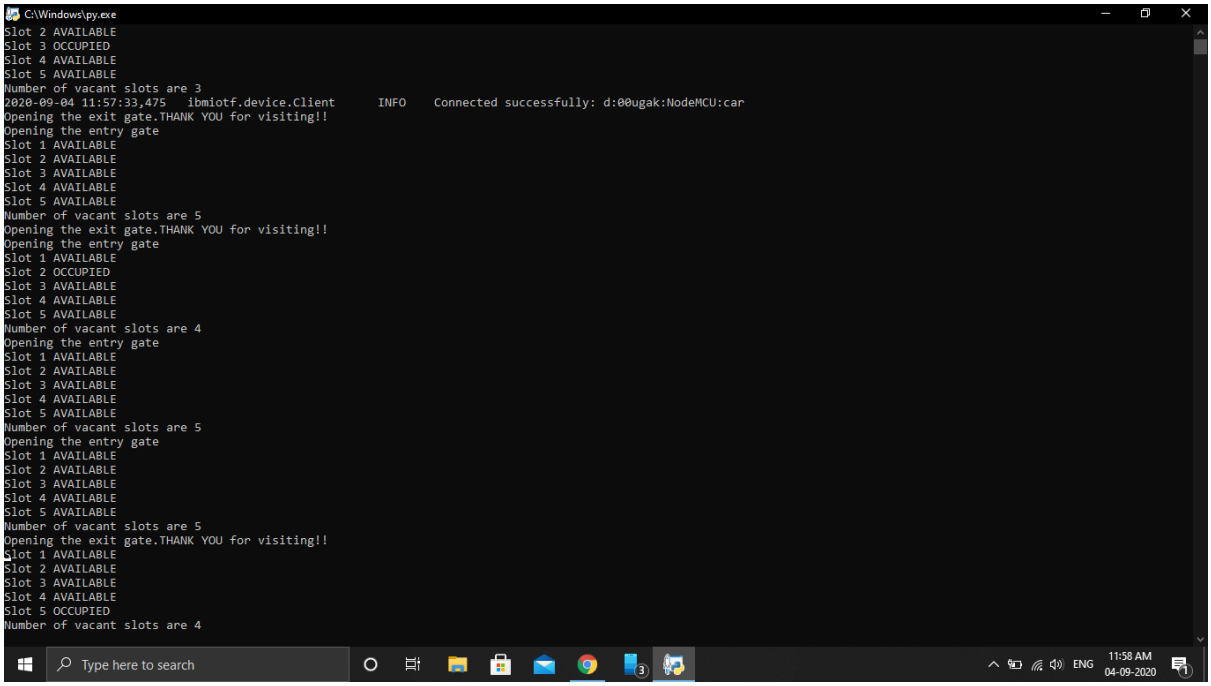
File Edit Shell Debug Options Window Help

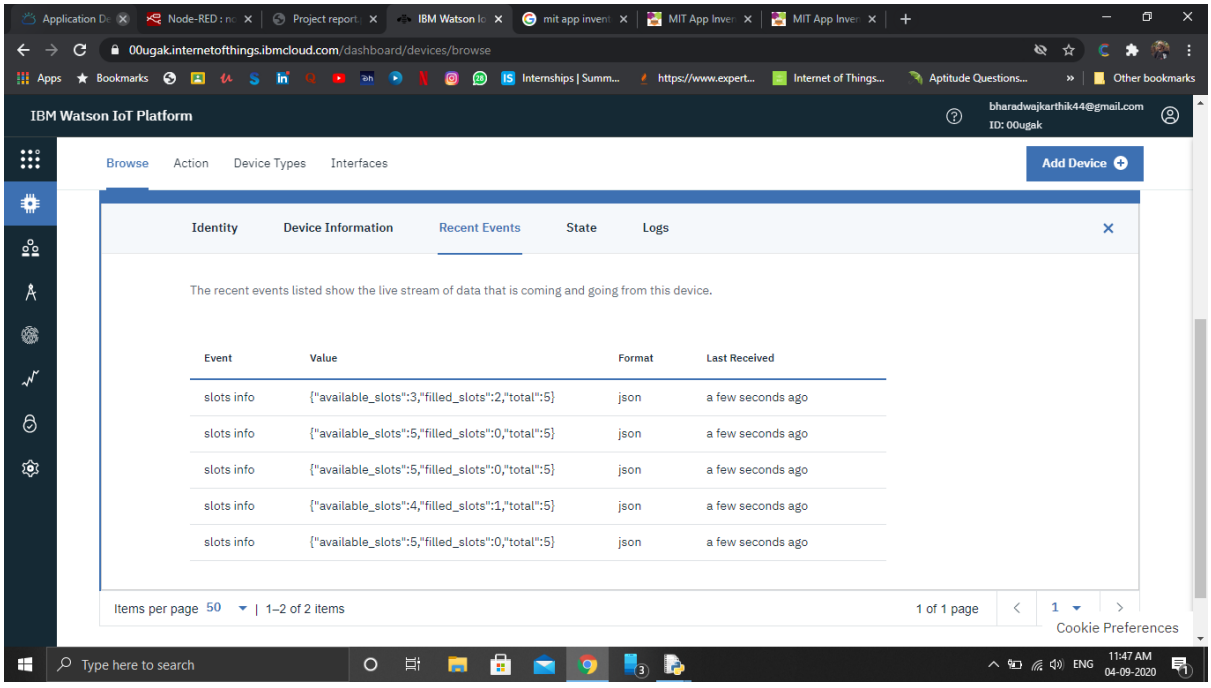
```
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Slot 1 AVAILABLE
Slot 2 OCCUPIED
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 4
Opening the exit gate.THANK YOU for visiting!!
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Opening the exit gate.THANK YOU for visiting!!
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 OCCUPIED
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 4
```

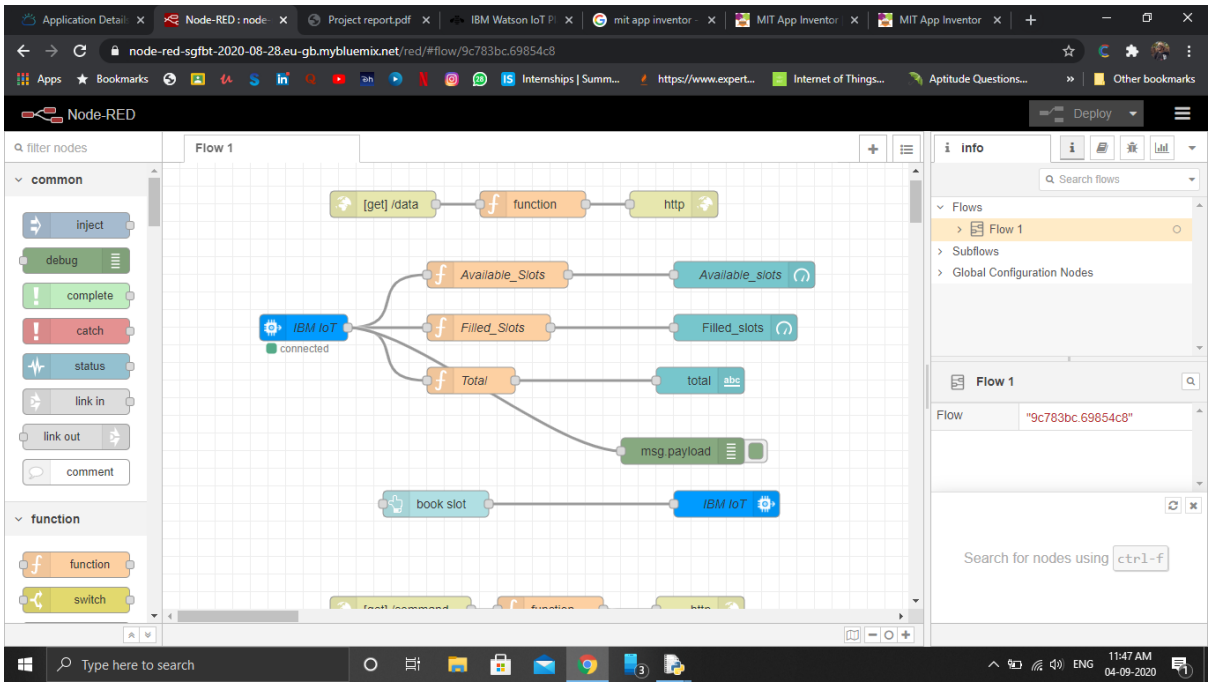
Ln: 114 Col: 22

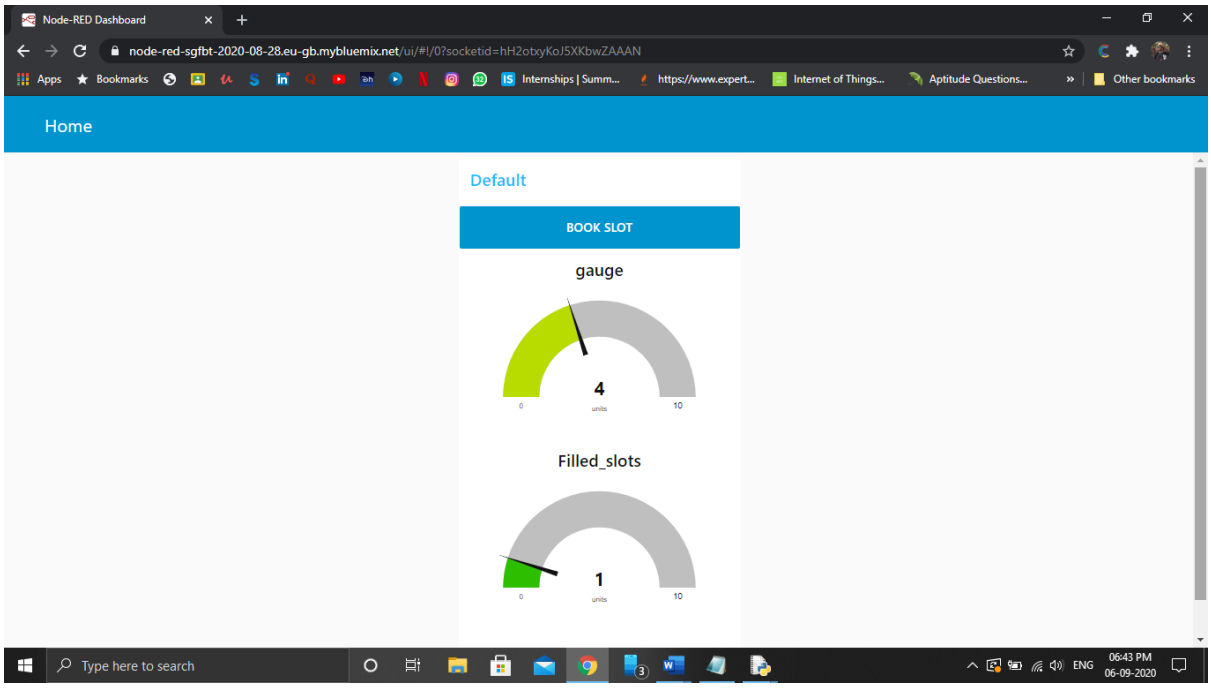
Type here to search

ENG 11:49 AM 04-09-2020


C:\Windows\py.exe
Slot 2 AVAILABLE
Slot 3 OCCUPIED
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 3
2020-09-04 11:57:33,475 ibmiotf.device.Client INFO Connected successfully: d:00ugak:NodeMCU:car
Opening the exit gate.THANK YOU for visiting!!
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Opening the exit gate.THANK YOU for visiting!!
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 OCCUPIED
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 4
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 AVAILABLE
Number of vacant slots are 5
Opening the exit gate.THANK YOU for visiting!!
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 OCCUPIED
Number of vacant slots are 4
Type here to search
ENG
11:58 AM
04-09-2020


00ugak.internetofthings.ibmcloud.com/dashboard/devices/browse
IBM Watson IoT Platform
bharadwajkarthik44@gmail.com
ID: 00ugak
Browse
Action
Device Types
Interfaces
Add Device
Identity
Device Information
Recent Events
State
Logs
The recent events listed show the live stream of data that is coming and going from this device.
Event
Value
Format
Last Received
slots info
{"available_slots":3,"filled_slots":2,"total":5}
json
a few seconds ago
slots info
{"available_slots":5,"filled_slots":0,"total":5}
json
a few seconds ago
slots info
{"available_slots":5,"filled_slots":0,"total":5}
json
a few seconds ago
slots info
{"available_slots":4,"filled_slots":1,"total":5}
json
a few seconds ago
slots info
{"available_slots":5,"filled_slots":0,"total":5}
json
a few seconds ago
Items per page 50
1–2 of 2 items
1 of 1 page
Cookie Preferences
Type here to search
ENG
11:47 AM
04-09-2020


Node-RED
Flow 1
[get] /data
function
http
Available_Slots
Available_slots
IBM IoT
connected
Filled_Slots
Filled_slots
Total
total
msg.payload
book slot
IBM IoT
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
info
Flows
Flow 1
Subflows
Global Configuration Nodes
Flow
"9c783bc.69854c8"
Search for nodes using ctrl-f


Node-RED Dashboard
Home
Default
BOOK SLOT
gauge
4
Filled_slots
1

THANK YOU