

PROJECT REPORT

Name: Ritika

Project title :Smart parking system for smart cities

Project ID: SPS_PRO_251

Internship title:june 15-IOT 4

CONTENT

1.INTRODUCTION

1.1 Overview

1.2 Purpose

2.LITERATURE SURVEY

2.1 Existing problem

2.2 Proposed solution

3.THEORETICAL ANALYSIS

3.1 Block diagram

3.2 Hardware/Software designing

4.EXPERIMENTAL INSVESTIGATIONS

5.FLOWCHART

6.RESULT

7.ADVANTAGES AND DISADVANTAGES

8.APPLICATIONS

9.CONCLUSIONS

10.FUTURE SCOPE

11.BIBLIOGRAPHY

APPENDIX

A.Source code

INTRODUCTION

Overview

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.

Nodemcu is used as central device for controlling and collecting data from sensors and then it is connected to IBM cloud platform using MQTT protocol.

PURPOSE

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 users.

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 Nodemcu which helps to serve as a medium of communication between those peripherals or devices. Data is then sent to IBM lot platform and finally,we can check availability using web application.

LITERATURE SURVEY

EXISTING PROBLEM

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.

PROPOSED SOLUTION

The existing parking problems create a necessity to introduce an automated smart parking system .A 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.

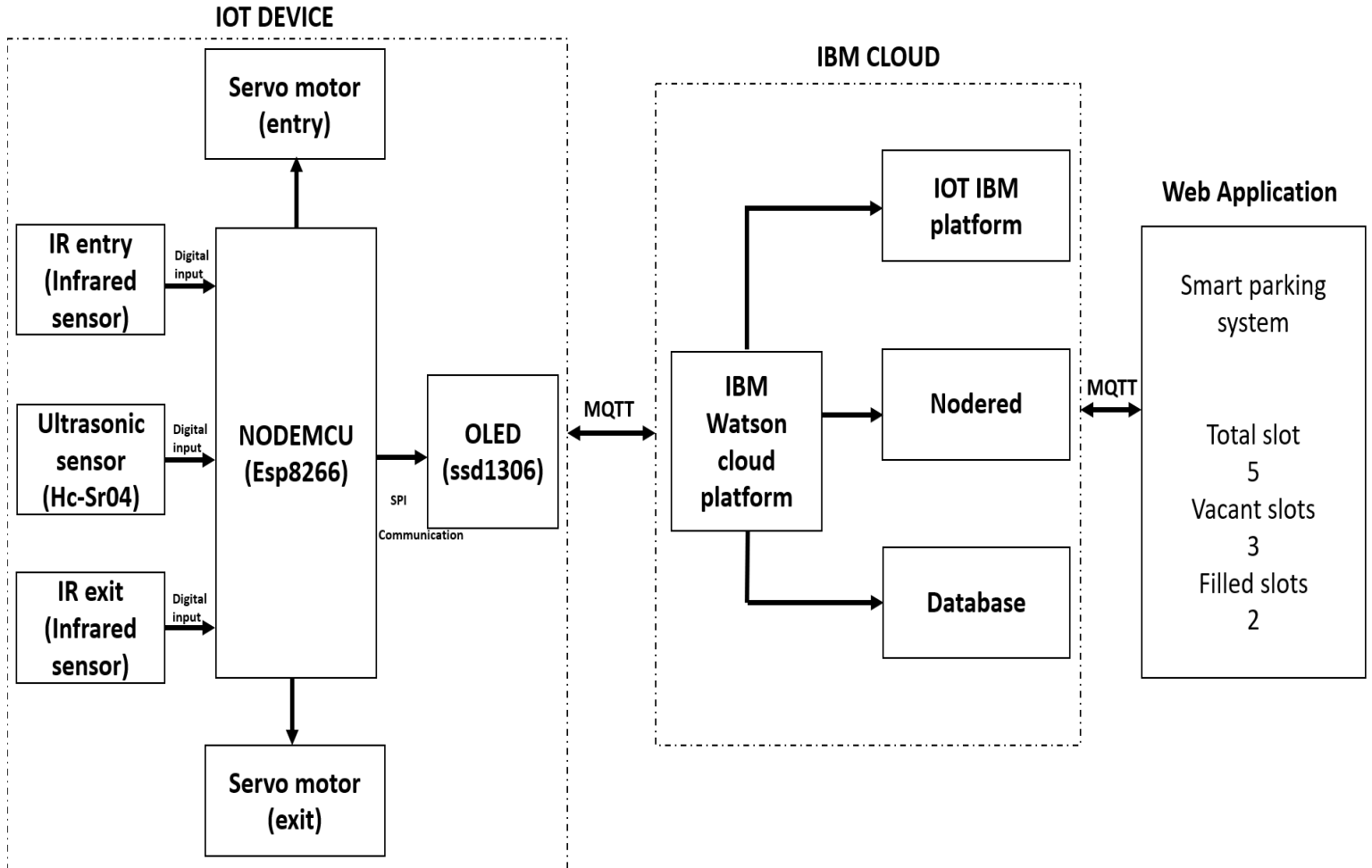
The proposed system works primarily on the detection of parking slots through sensors that are mounted on every parking slots which facilitates the information. This system consists of three parts.The first is the interfacing and development of the sensor which are deployed in the parking spaces.Data 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 lot cloud which is connected to nodered .

The third is web application, which is developed using nodered .It fetch the data from IBM lot 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.

BLOCK DIAGRAM



Hardware/Software Designing

The five main hardware components used are NodeMCU, OLED(ssd1306), Ultrasonic sensors(Hc-Sr04), IR sensors and Servo motors.

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.

IR Sensor

An infrared sensor is basically an electronic device which is used to detect the presence of objects. Infrared light is emitted by this device. If this device does not detect any IR light reflected back that means there is no object present. If the light is detected by the sensor there is an object present.



Ultrasonic Sensor(Hc-Sr04)

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. The presence of vehicle is checked using it. If distance measured is less than or 40cm, then vehicle is present.



Servo Motor

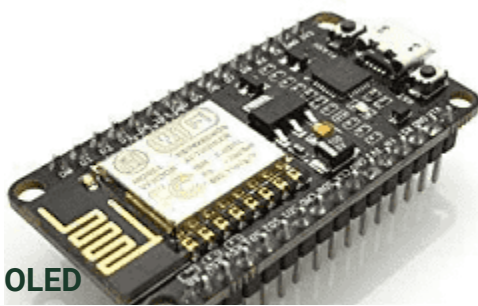
servo motor is an electrical device which can push or rotate an object with great precision. If you want to rotate an object at some specific angles or distance, then you use servo motor. It is just made up of simple motor which runs through **servo mechanism**. In this project, it is used to open and close the gate.



NODEMCU(Esp8266)

NodeMCU is an open source firmware for which open source prototyping board designs are available. Both the firmware and prototyping board designs are open source.^[9]

The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It is used here to fetch data from sensors, drive servo motor and OLED. It is a 32 bit microcontroller with 80kb user data. It contains 16 gpio pins.



OLED

OLED (Organic Light Emitting Diodes) is a flat light emitting technology, made by placing a series of organic thin films between two conductors. When electrical current is applied, a bright light is emitted. OLEDs are emissive displays that do not require a backlight and so are thinner and more efficient than LCD displays (which do require a white backlight). Here it is used to display parking space information.

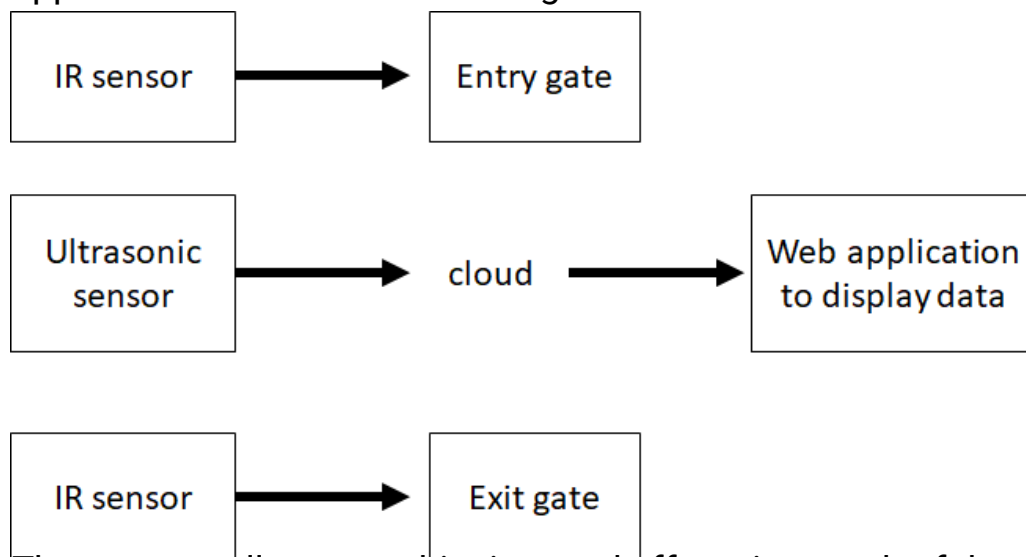
SOFTWARE

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.

EXPERIMENTAL INVESTIGATIONS

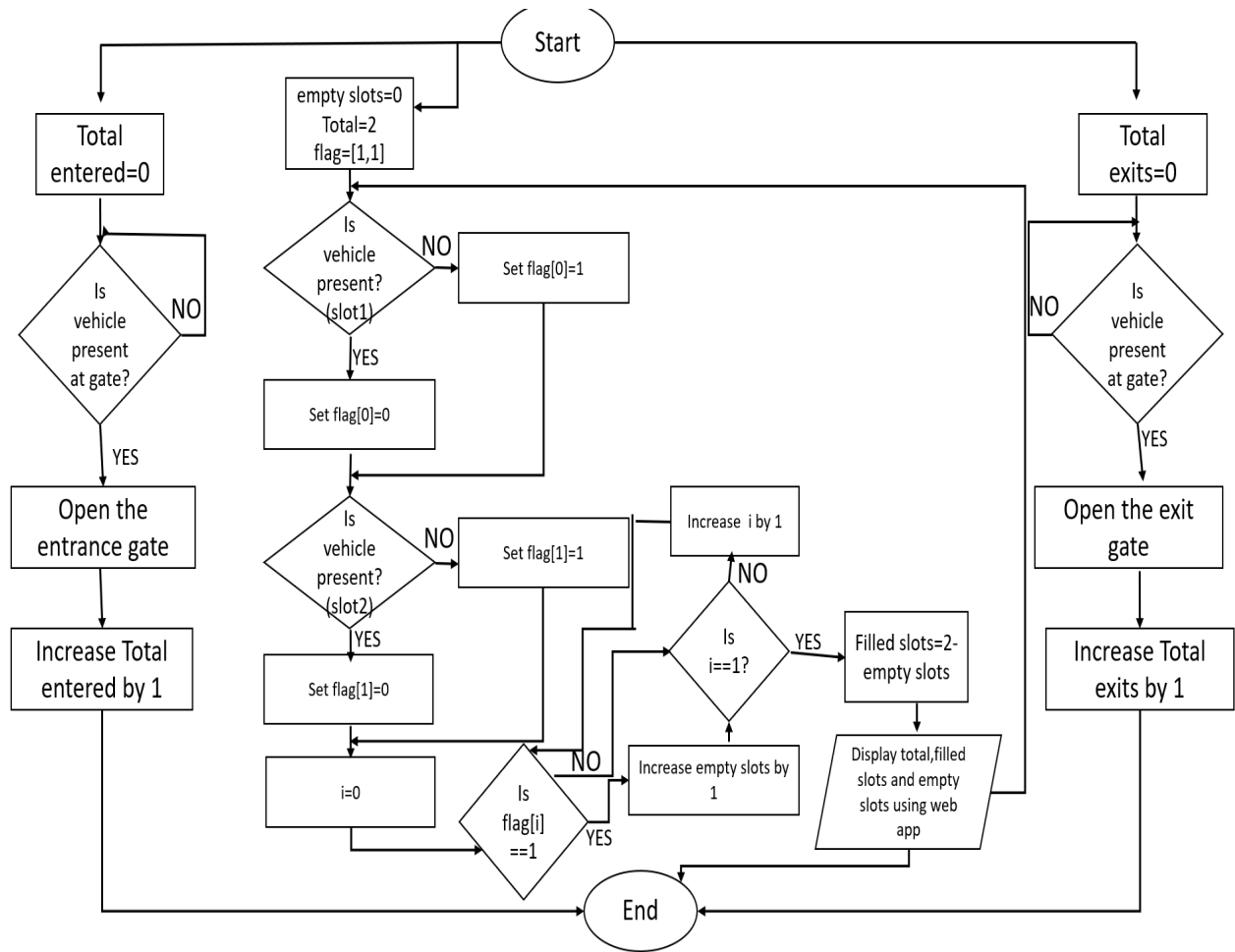
The proposed system is based on the observation of existing system of parking. With increase in the population of the vehicles in metropolitan cities, road congestion is the major problem that is being faced. This system is efficient and smart way to automate the management of the parking system that 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. NodeMCU sends data to cloud and then nodered 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.

FLOW CHART



RESULT

The demand of smart parking system is increasing significantly. The existing system in today's world doesn't contain 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.

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

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

APPLICATIONS

- The smart parking system can be implemented in
 - Shopping malls
 - Theatre
 - Restaurants
- 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 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.

CONCLUSION

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.

FUTURE SCOPE

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 valet to park the vehicles.

We can notify users via SMS through GSM module.

BIBLIOGRAPHY

Paper:Journal of physics-conference series

www.google.com

www.wikipedia.com

Paper:International Journal of Engineering and Advanced Technology

SOURCE CODE

```
1  import time, random, sys
2  import ibmiotf.device
3
4  #Provide your IBM Watson Device Credentials
5
6  organisation="umm3j1"
7  devicetype="NodeMCU"
8  deviceid="1234"
9  authm="token"
10 authtoken="123456789"
11
12 def myCommandCallback(cmd) :
13     print("Command received:%s"% cmd.data["command"])
14     #commands
15     print("Booking a slot.")
16
17 try:
18     dataop={"org":organisation, "type":devicetype, "id":deviceid,
19            "auth-method":authm, "auth-token":authtoken}
20     devicecli=ibmiotf.device.Client(dataop)
21
22 #.....
23 except Exception as e:
24     print("Caught exception connecting device:%s"% str(e))
```



```

22     sys.exit()
23
24 devicecli.connect()
25
26 totalslot=5
27 emptyslot=5                                #total 5 slots empty by
    default
28 slot=[1,1,1,1,1]                          #all 5 slots empty
    initially
29 total_entered=0
30 total_exits=0
31 while True:
32     #entry gate
33     IREntry=random.randint(0,1)            #IRsensor gives 1 if
    there is a vehicle at the gate
34     if IREntry==1:
35         print("Opening the entry gate")
36         total_entered=total_entered+1
37     if emptyslot==0:
38         print("Sorry,parking slots are NOT AVAILABLE!!")
    #print a message if no slots available
39
40     emptyslot=0
41
42     #Check for empty slots
43     ultrasonic1=random.randint(2,400)
44     if ultrasonic1<=40:
45         print("Slot 1 OCCUPIED")
46         slot[0]=0                          #setting flag 0 for
    occupied slot
47     else:
48         print("Slot 1 AVAILABLE")
49         slot[0]=1                          #setting flag 1 for
    available slot
50
51     ultrasonic2=random.randint(2,400)

```

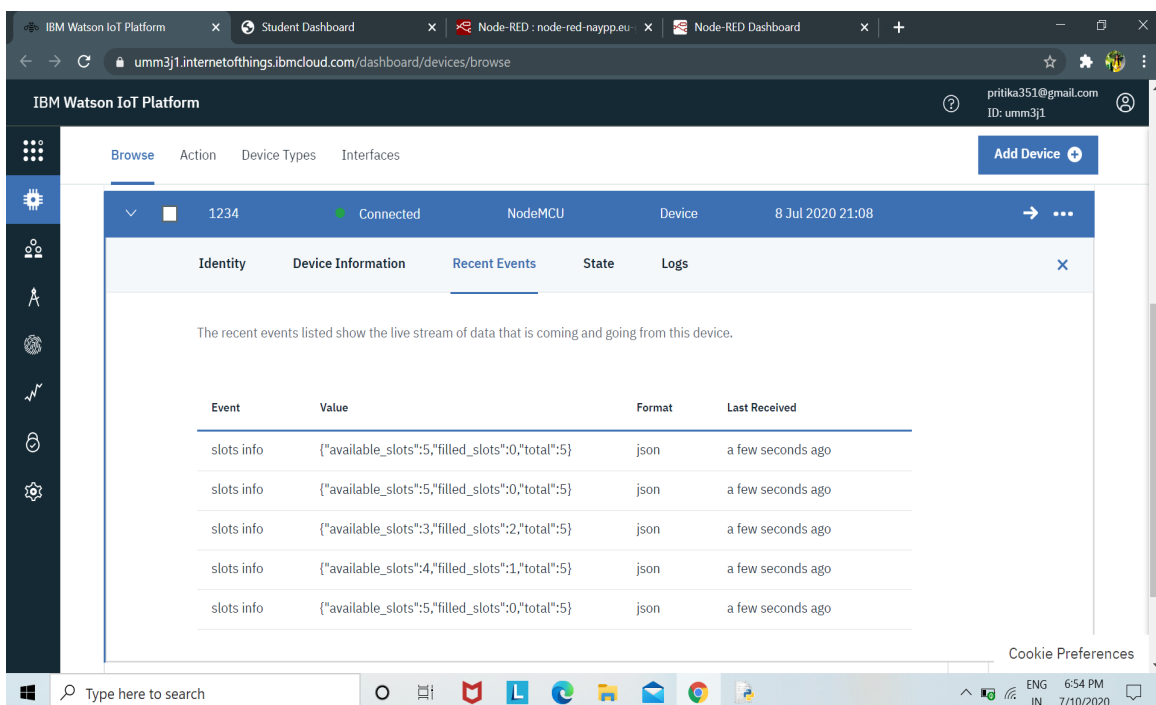
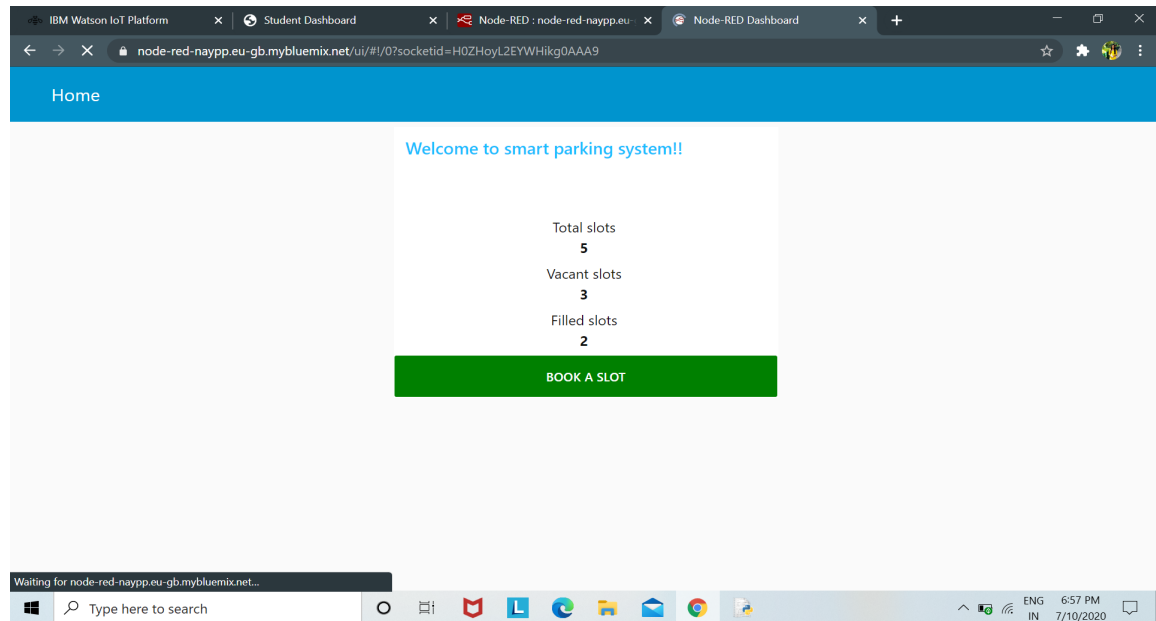
```
52     if ultrasonic2<=40:
53         print("Slot 2 OCCUPIED")
54         slot[1]=0
55     else:
56         print("Slot 2 AVAILABLE")
57         slot[1]=1
58
59
60     ultrasonic3=random.randint(2,400)
61     if ultrasonic1<=40:
62         print("Slot 3 OCCUPIED")
63         slot[2]=0
64     else:
65         print("Slot 3 AVAILABLE")
66         slot[2]=1
67
68     ultrasonic4=random.randint(2,400)
69     if ultrasonic3<=40:
70         print("Slot 4 OCCUPIED")
71         slot[3]=0
72     else:
73         print("Slot 4 AVAILABLE")
74         slot[3]=1
75
76     ultrasonic5=random.randint(2,400)
77     if ultrasonic5<=40:
78         print("Slot 5 OCCUPIED")
79         slot[4]=0
80     else:
81         print("Slot 5 AVAILABLE")
82         slot[4]=1
83
84
85
86     for i in slot:
87         if i==1:
```

```

88         emptyslot=emptyslot+1
89
90     data={"available_slots":emptyslot,"filled_slots":5-emptyslot
    ,"total":totalslot}
91     def myonpublishcallback():
92         print("Number of vacant slots are %s"% emptyslot)
93
94     success=devicecli.publishEvent("slots
    info","json",data,qos=1,on_publish=myonpublishcallback())
    #publishing data to cloud
95
96     if not success:
97         print("Not connected to IOTf.")
98
99
100         #exit gate
101         IRexit=random.randint(0,1)
102         if IRexit==1:
103             print("Opening the exit gate.THANK YOU for
    visiting!!")
104             total_exits==total_exits+1
105
106         time.sleep(5)                                #checks status
    every 5 seconds
107         devicecli.commandCallback=myCommandCallback
108
109     print(total_entered)
110     print(total_exits)
111
112 # Disconnect the device and application from the cloud
113 devicecli.disconnect()

```

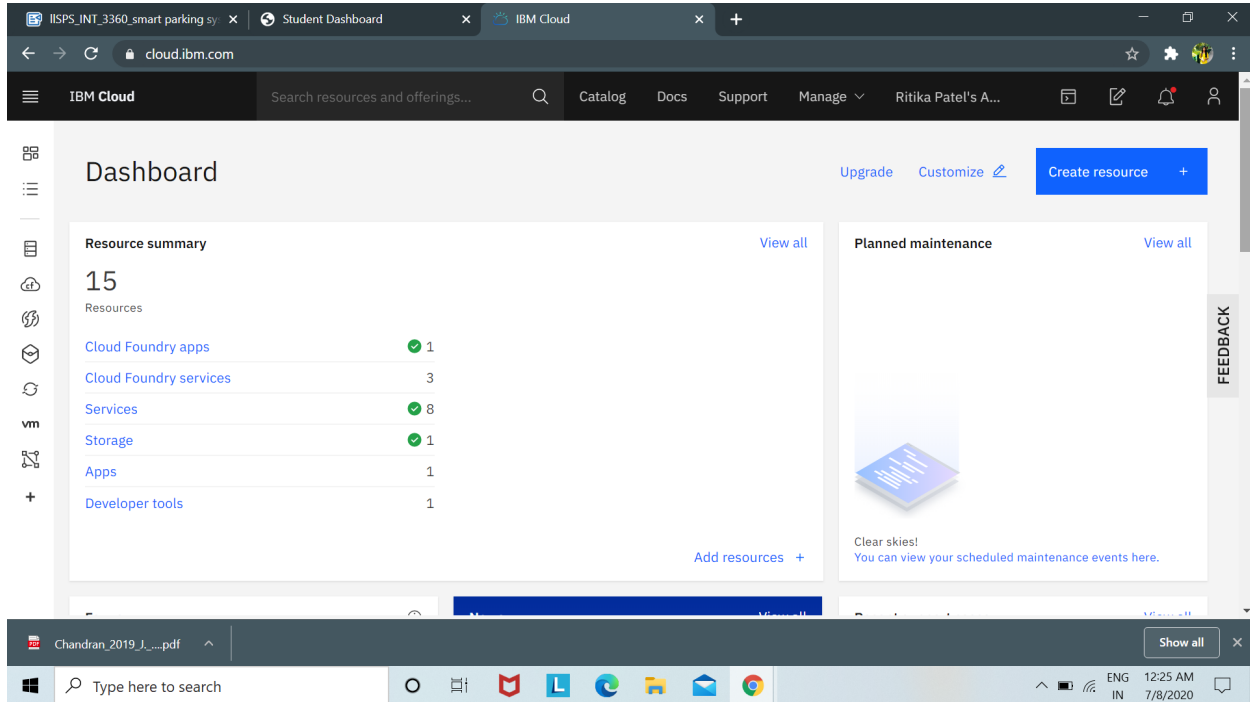
OUTPUT



```
*Python 3.8.2 Shell*
File Edit Shell Debug Options Window Help
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 exit gate.THANK YOU for visiting!!
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 OCCUPIED
Slot 3 AVAILABLE
Slot 4 OCCUPIED
Slot 5 AVAILABLE
Number of vacant slots are 3
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
Command received:book a slot
Booking a slot.
Opening the entry gate
Slot 1 AVAILABLE
Slot 2 AVAILABLE
Slot 3 AVAILABLE
Slot 4 AVAILABLE
Slot 5 OCCUPIED
Number of vacant slots are 4
Opening the exit gate.THANK YOU for visiting!!
|
```

Ln: 5 Col: 0

IBM account created



The screenshot shows the IBM Cloud Dashboard in a web browser. The browser tabs include 'IISPS_INT_3360_smart parking sy', 'Student Dashboard', and 'IBM Cloud'. The address bar shows 'cloud.ibm.com'. The dashboard header includes the IBM Cloud logo, a search bar, and navigation links for Catalog, Docs, Support, and Manage. The user's name 'Ritika Patel's A...' is visible. The main content area is titled 'Dashboard' and features a 'Resource summary' section with a count of 15 resources. Below this, a list of resource categories is shown with their respective counts and status indicators. A 'Planned maintenance' section is also visible on the right. The bottom of the screen shows a Windows taskbar with various application icons and system information.

IBM Cloud

Search resources and offerings...

Catalog Docs Support Manage Ritika Patel's A...

Dashboard

Upgrade Customize Create resource +

Resource summary View all

15 Resources

- Cloud Foundry apps 1
- Cloud Foundry services 3
- Services 8
- Storage 1
- Apps 1
- Developer tools 1

Add resources +

Planned maintenance View all

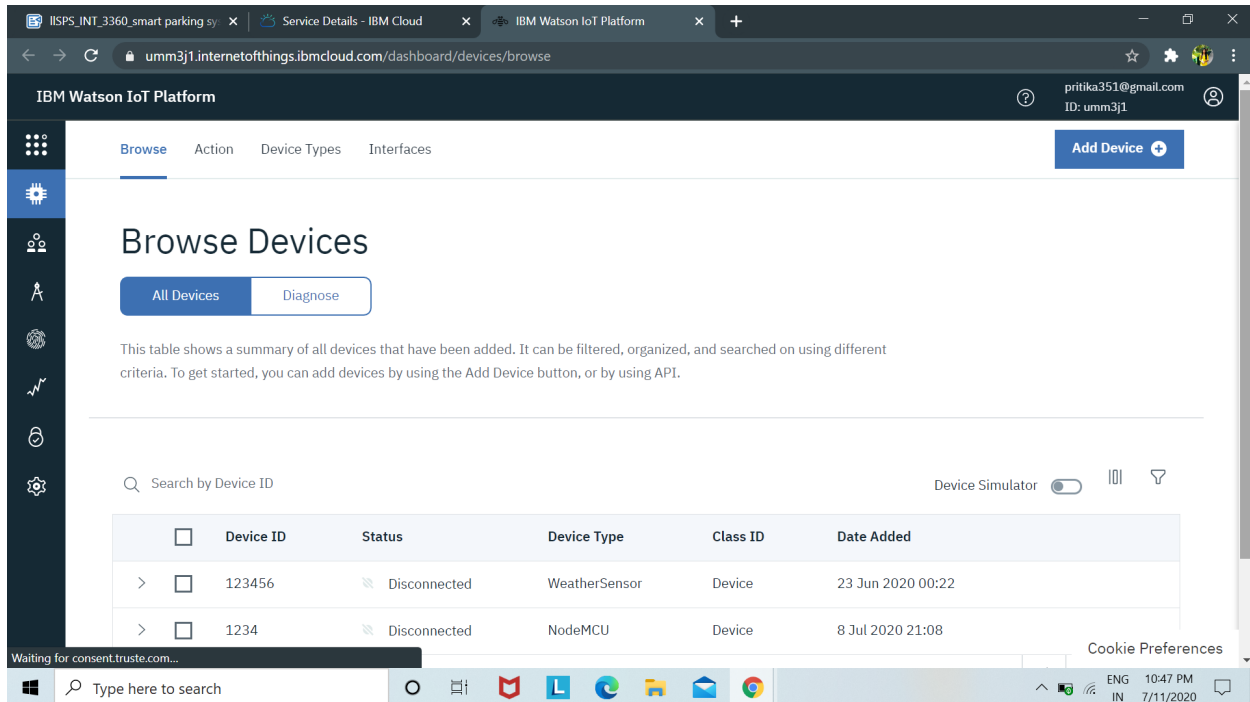
Clear skies! You can view your scheduled maintenance events here.

Chandran_2019_J_...pdf

Type here to search

ENG IN 12:25 AM 7/8/2020

IBM IOT platform



The screenshot shows the IBM Watson IoT Platform dashboard in a web browser. The browser tabs include 'IISPS_INT_3360_smart parking sy', 'Service Details - IBM Cloud', and 'IBM Watson IoT Platform'. The address bar shows 'umm3j1.internetofthings.ibmcloud.com/dashboard/devices/browse'. The dashboard header includes the IBM Watson IoT Platform logo, a search bar, and navigation links for Browse, Action, Device Types, and Interfaces. The user's name 'pratika351@gmail.com' and ID 'ID: umm3j1' are visible. The main content area is titled 'Browse Devices' and features a table showing a summary of all devices that have been added. The table includes columns for Device ID, Status, Device Type, Class ID, and Date Added. A 'Device Simulator' toggle is also present. The bottom of the screen shows a Windows taskbar with various application icons and system information.

IBM Watson IoT Platform

pratika351@gmail.com ID: umm3j1

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

Device ID	Status	Device Type	Class ID	Date Added
123456	Disconnected	WeatherSensor	Device	23 Jun 2020 00:22
1234	Disconnected	NodeMCU	Device	8 Jul 2020 21:08

Waiting for consent.truste.com...

Type here to search

ENG IN 10:47 PM 7/11/2020

Nodered app created

The screenshot shows the IBM Cloud Dashboard. The top navigation bar includes the IBM Cloud logo, a search bar, and links to Catalog, Docs, Support, and Manage. The main content area is titled "Dashboard" and features a "Resource summary" section with a count of 15 resources. Below this, a table lists various resource categories and their counts:

Resource Category	Count
Cloud Foundry apps	1
Cloud Foundry services	3
Services	8
Storage	1
Apps	1
Developer tools	1

Other sections visible include "Planned maintenance" with a "Clear skies!" message, "Recent support cases", and a "For you" section with a link to "Get started with using AI and Cloud Object". The bottom of the dashboard shows a Windows taskbar with various application icons and a system clock indicating 10:46 PM on 7/11/2020.

Flow

The screenshot displays the Node-RED interface within a web browser. The main workspace shows a flow titled "Flow 4" with the following components:

- An "IBM IoT" node (connected) that triggers a sequence of three "function" nodes.
- The first "function" node outputs to a "total" node with the value "abc".
- The second "function" node outputs to a "filled slots" node with the value "abc".
- The third "function" node outputs to an "empty slots" node with the value "abc".
- A "msg.payload" node is also connected to the output of the third function node.
- A "Book a slot" node is connected to the "IBM IoT" node.

The right-hand pane shows the "debug" console with a log of messages received from the IoT node, including timestamps and JSON payloads. The bottom of the interface shows a Windows taskbar with various application icons and a system clock indicating 6:59 PM on 7/10/2020.