

# **PROJECT REPORT**

INTERNSHIP TITLE : JUNE 29(IOT) - 5

PROJECT ID : SPS\_PRO\_251

PROJECT TITLE : SMART PARKING SYSTEM FOR SMART CITIES

DURATION :- 9 DAYS

PROJECT MANAGER : GUNJAN MOHOD

## **Contents**

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 INVESTIGATIONS	
5 FLOWCHART	
7 ADVANTAGES AND DISADVANTAGES	
8 APPLICATIONS	
9 CONCLUSION	
10 FUTURE SCOPE	
11 BIBLIOGRAPHY	
12 APPENDIX	
A. Source	
cod	
B. UI output Screenshot.	

### **1. INTRODUCTION**

#### **1.1 OVERVIEW**

This report presents design, manufacturing, simulations conducted by me, I've worked in the field of IoT. I have worked for strengthening concepts related to IOT. In this project I have used various softwares likewise IBM Watson Platform, node-red, IBM cloud platform to build this application.

#### **1.2 PURPOSE**

The objective of this project is to build an application based on smart parking system. This application uses real time data 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 slot.

## 2. LITERATURE SURVEY

### 2.1 EXISTING PROBLEM

1. Vehicle production has grown considerably in last 30 years.
2. Having more vehicles around the streets implies more fuel and time consumption and a growing demand for parking spaces.

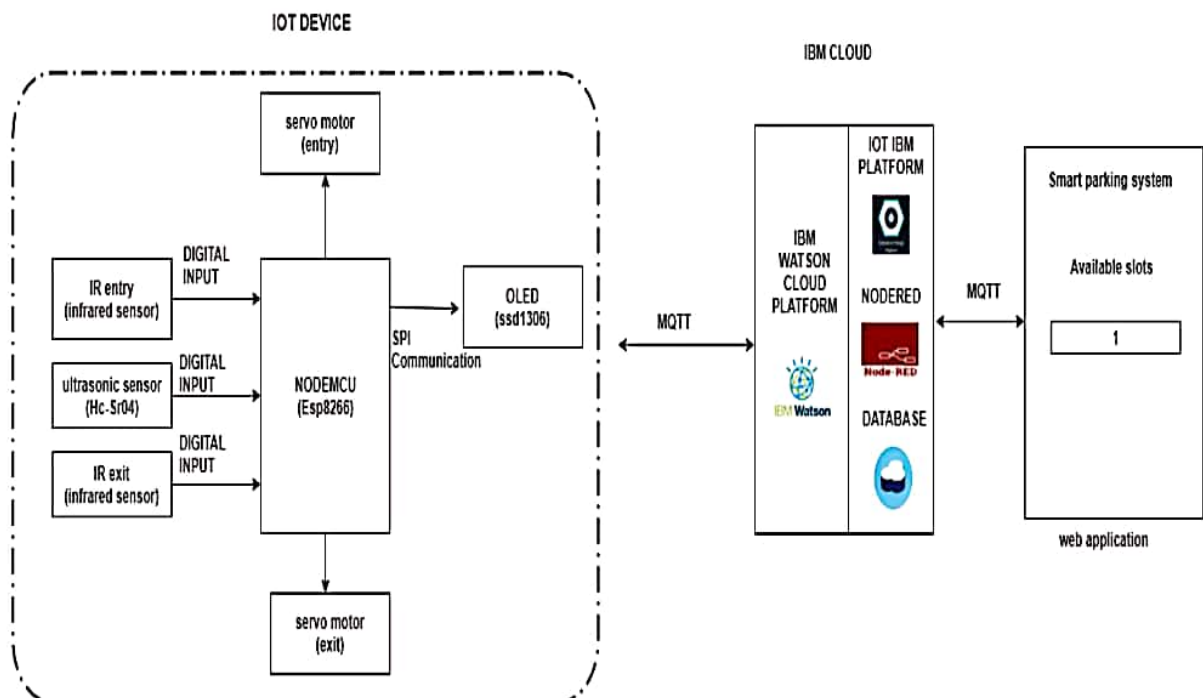
### 2.2 PROPOSED SOLUTION

1. This application uses real time data that allow users to monitor the available and unavailable parking spaces.

2.This application is designed to provide drivers an ultimate solution on their journey from beginning to end without searching for parking, cost, travel time etc.

### 3.THEOROTICAL ANALYSIS

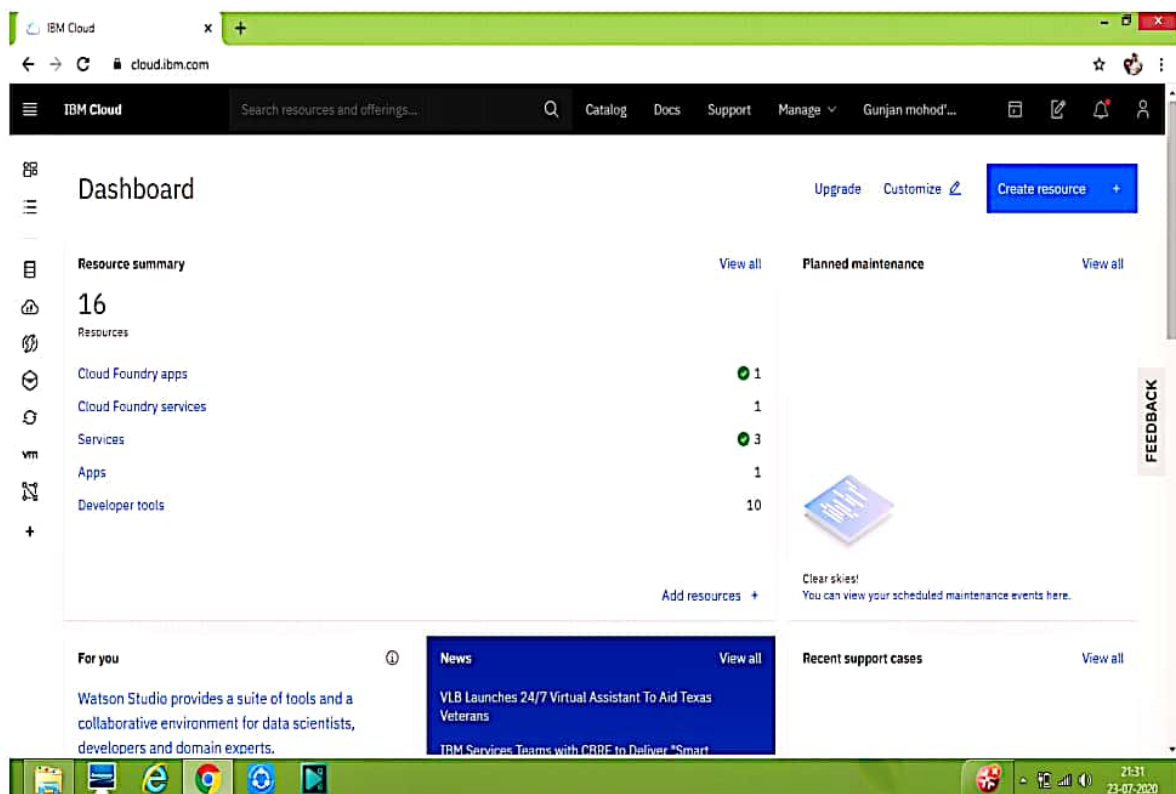
#### 3.1BLOCK DIAGRAM



## 3.2 SOFTWARE DESIGNING

### ❖ IBM CLOUD

- IBM Cloud provides a lot of services which are more reliable, easily accessible and cost effective.
- Under this Cloud, Internet of things platform is used to build an application.
- Device has been created using IBM Watson IoT Platform.



## ➤ WATSON IoT PLATFORM

- IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices

[Resource list](#) /

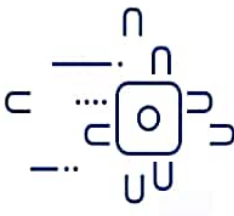
Internet of Things Platform-io ● Active [Add tags](#) [🔗](#)

[Details](#) [Actions...](#)

Manage

Plan

Connections




### Let's get started with IBM Watson IoT Platform

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

[Launch](#) [Docs](#)


Ready for the next level?

#### IBM Watson IoT Platform Journey




##### Lite

The Lite service plan provides a lightweight development environment to get you started with the connectivity capabilities of Watson IoT Platform.



##### Non-Production

The Non-Production service plan is a full-featured, fully-integrated offering that enables you to explore Watson IoT Platform to see how the service can fit into your IoT environment.



##### Production

The Production service is a fully managed SaaS offering that enables you to manage and analyze enterprise IoT data.

Scanned with CamScanner

## ❖ NODE-RED

- Node-red is a workflow based virtual programming tool that is relying a lot on input/output with json as data exchange format which is developed by IBM.
- Installing Node-red locally gives you access to the flow editor.
- For connecting the device you have to install IBM nodes in the node-red

## ❖ USER INTERFACE

- The user interface(UI) in the industrial design field of human-computer interaction, is the space where interactions between humans and machines occur
- Install the dashboard node from manage pallet to create a UI
- For motor configuring commands, configure the nodes to create buttons.
- Display the command using UI.



## ❖ SLACK

- Slack is a proprietary business communication platform
- Slack is a communication tool that is created to streamline and simplify conversations. It is similar to the other messaging applications.
- A channel for every organized conversations. It is a best platform that is used to communicate with with the mentors and getting the instructions related to the project
- Work faster with your tools in one place.

## ❖ GITHUB

- Comapny that provides hosting for software developement version control using Git.
- It makes a lot easier for individuals and teams to use Git for version control and collaboration.
- Github lets you work together on a project
- Github Repository used to store your project developement
- Git is an open-source that is its main advantage
- Github is used in our project to store and upload files and get reviews through it.

#### 4. EXPERIMENTAL INVESTIGATION

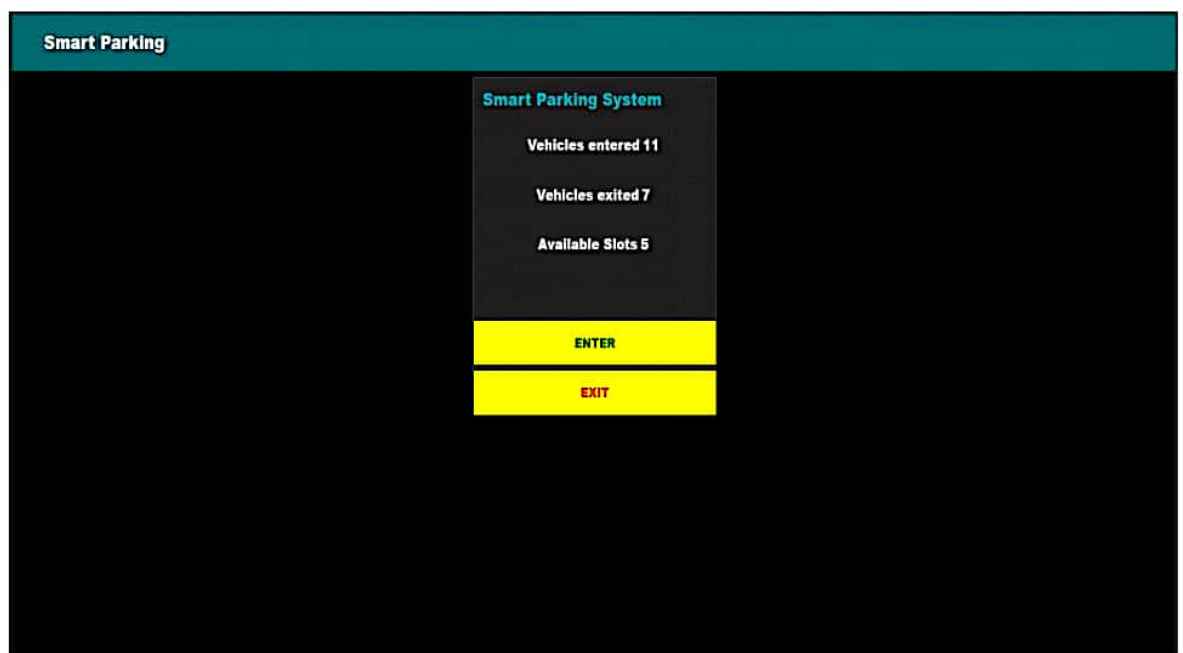
- The scalable and robust nature of cloud computing is allowing developers to create and host their applications on it. cloud acts as a perfect partner for IoT as it acts as a platform where all the sensor data can be stored and accessed from remote locations
- For this application we have made use of sensors like Infrared and Ultrasonic sensors. The work of these sensors is the same i.e. to sense the parking area and determine whether a parking slot is vacant or not. In this case we are using ultrasonic sensors to detect the presence of a car. The ultrasonic sensor is wirelessly connected to raspberry pi.
- The processing unit comprises of Raspberry processor on chip. The processing unit acts like an intermediate between the sensor and cloud.  
All the sensors are wirelessly connected to the processing unit.

- The data collected from various sensors is sent to the raspberry pi. The Raspberry pi then transmit the data to the IBM MQTT server through MQTT protocol.
- MQTT protocol is a publish-subscribe based "Light weight" messaging protocol.
- The IBM MQTT server is hosted on cloud. Cloud acts as a data base to store all the records related to parking areas and end users that have access to the system.
- The IBM cloud keeps a track of every user connected to the system and maintains information such as time at which the car was parked, time duration for parking a car etc. It is due to flexible nature of cloud which permits the system to add any number of users at any time of the day.
- For this application first create a nodered application and a watson iot platform. Then we have to create a Node-red flow to get the data from the device.

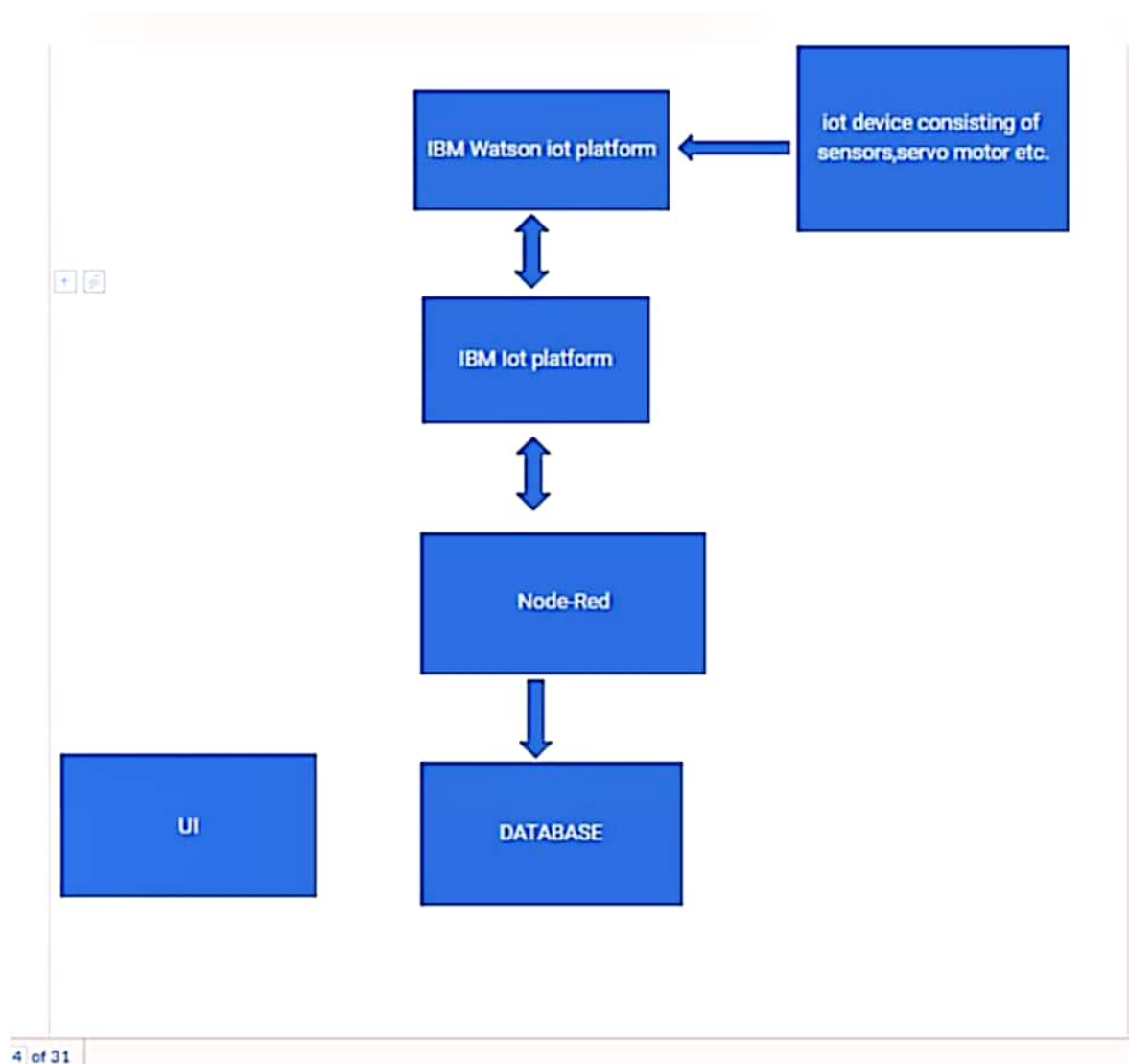
## ❖ RESULT:

- UI will be created in node-red which will display the sensor data, motor commands etc.
- You can get the information about the parking slot, weather it is available or not.

## 🌈 UI:



## 5.FLOWCHART:



## **7.ADVANTAGES AND DISADVANTAGES**

### **❖ ADVANTAGES**

- Optimized Parking
- Reduced Traffic
- Reduced pollution
- Enhanced User Experience
- Increased Safety
- Real-Time Data and Trend Insight.
- Less fuel is wasted
- Smart parking reducing stress while searching for a parking space

### **• DISADVANTAGES**

- Network issue may cause problem for operating the application
- Developing the application may take more time.
- Difficult to use in mixed traffic.
- The control system software could be hacked by the hackers

## 8.Applications:

- Real-Time monitoring of parking space.
- smart parking system promote safety by notifying drivers when they are about to leave a vehicle in a no-parking area.
- Optimizing space and time in a tight and busy urban environment.
- Monitoring can be possible from any location

## 9.CONCLUSION:

- Software designing is easy to develop and understand
- smart parking facilities and traffic management systems have always been at the core of constructing smart cities.
- This application provides real-time information regarding availability of parking slots in a parking area.
- Users from remote location can book their parking slot through this application.
- All the softwares needed to build the application are accessible and cost effective



## 10. FUTURE SCOPE

- Creating Smarter Mobility Networks
- The future of smart parking system is expected to be significantly influenced by the arrival of automated vehicles. Several cities around the world are already beginning to trial self-parking vehicles, specialized AV parking lots, and robotic parking valets.

# 11. BIBILOGRAPHY

## Reference links

☐ <https://www.w3trainingschool.com/future-scope-of-android-app-developement-india>

☐ <https://nodered.org/>

☐ [https://www.w3schools.com/whatis/whatis\\_github.asp](https://www.w3schools.com/whatis/whatis_github.asp)

☐ [https://en.wikipedia.org/wiki/Slack\(software\)](https://en.wikipedia.org/wiki/Slack(software))

- <https://blog.nuclino.com/how-to-write-a-technical-specification-or-software-design-document-sdd>

# APPENDIX

## A. Source Code:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "*****"
deviceType = "*****"
deviceId = "*****"
authMethod = "token"
authToken = "*****"

#initialize GPIO

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)#Commands received from nodered
    print(type(cmd.data))
    i=cmd.data['command']
    if i == 'lighton':
        print("Light is on")
    elif i == 'lightoff':
        print("light is off")

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
    #JSON format
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #connecting the client

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

deviceCli.connect()
# connecting to the platform

while True:
    enter=random.randint(1, 40)
    exits=random.randint(1, 40)
    available=random.randint(1, 40)
    .....
```

```

#Send entry and exit status to IBM Watson
data = { 'Entry' : enter, 'Exit': exits, 'Available slots': available}
#print (data)
def myOnPublishCallback():
    print ("Publish Entries = %s " % enter, "Exits = %s " % exits, "Available slots = %s " %available, "to IBM Watson")

#publish the data
success = deviceCli.publishEvent("Smart Parking", "json", data, qos=0, on_publish=myOnPublishCallback)
if not success:
    print("Not connected to IoT")
time.sleep(2)

deviceCli.commandCallback = myCommandCallback
#subscribing

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

```

# B.UI OUTPUT SCREENSHOT:

