An IoT BASED SMART CAR PARKING SYSTEM

Report Submitted to

Purnea College of Engineering, Purnea

For the award of the degree

of

Bachelor of Technology

In

Electronics & Communication Engineering

by

ASHISH KUMAR GUPTA

17104131023

Under the Supervisions of

Prof.Vikas kumar

DEPARTMENT OF ECE

PURNEA COLLEGE OF ENGINEERING, PURNEA

SEPTEMBER 2021

DECLARATION

I declare that

a) The work contained in this project is original and has been done by me under the guidance of my supervisor(s).

b) The work has not been submitted to any other Institute for any degree or diploma.

c) I have followed the guidelines provide by the Institute in preparing the report.

d) I have conformed to the norms and guidelines given in the Ethical code of Conduct of the Institute.

e) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the report and given their details in the references.

Signature of the Student

.....

Ashish Kumar Gupta

CERTIFICATE

This is certified that the Report entitled, **"IOT Based car parking system"** submitted by **"Ashish Kumar Gupta"** to Purnea College of Engineering, Purnea, is a record of bonafide Major Project work carried out by him under our supervision and guidance and worthy of consideration for the award of degree of Bachelor of Technology in Electronics and Communication Engineering.

Prof.-Vikas Kumar
Project Supervisor

Prof.-Md Iftekhar Alam
Head of the Department

Appeared for the end of the semester Project viva-voice examination held on……..

Internal Examiner

External Examiner

ABSTRACT

Recently parking has become a serious issue and even worsen, because of the increasing number of automobiles everywhere. In this paper we propose an IoT based guidance for user to monitor and book the parking space for the vehicle and for managing and monitoring free parking space, it provides an intelligent solution. It aims at implementing smarter and better parking guidance mechanism which significantly reduces difficulty in conventional parking system. The system can monitor the state of every parking slot by deploying a sensor node on the slot. Accordingly sensor senses the status of parking slot and send status to central node server controller. The Node MCU collect the data from all sensor node and upload to the server where user can check the parking status from anywhere using internet and any browser. User can also book a parking slot in by crating there profile on the server.

Keywords:- Node MCU, LDR, Resistor, LED....etc

Contents

Introduction

Now a days, main problem in malls, function halls and etc., is parking. It is due to the lack of sufficient parking space. Now a days the vehicles in a family are greater than the head count of the family members, and due to this the vehicles are also increased in the country, which leads to the parking scenario which is unhappily falling short to the current requirements in the country. Due to this parking is difficult and it also increases the time needed to park the vehicle with increase in the fuel consumption of the vehicle. And during the working days the companies and offices are facing the problem of the parking in urban areas. Now a days vehicles are most affordable to the low income group families also and the vehicles especially the cars are taking lot of space. Due to the increase in vehicles the parking space is also not sufficient in this congested cities. Whether at a shopping malls, stations and airport, problems with parking is a big issue. Most of the time people spend their time on searching parking, to park their vehicles. Thus, lot of congestion occurs in the traffic which leads to a tedious job to find the parking space to park their vehicle. The most traffic occurs only because of vehicle congestion in the urban areas thus people are wasting time in searching the parking area abnormally to park their vehicles. And one more issue is also added to this is pollution, which effects the entire environment due to this increase in vehicles.

PROPOSED WORK

Recently, with the explosive increase of automobiles in cities, parking problems are serious and even worsen in many cities. The aim of the project is to design and provide:-

- A simple web application for parking vehicles.
- Booking parking slot from home.
- Can search nearby places using google map.
- Easy payment system.
- Parking owners can add their won parking palaces.
- Make easy to automate parking owners and customers.

User can search their destination parking facilities and according the availablity they can choose vehicle either 4- wheeler, 2-wheeler or public transport to reach their destination.

Smart parking development implies an IoT-based system that sends data about free and occupied parking places via web/mobile application. The IoT-device, including sensors and microcontrollers, is located in each parking place. The user receives a live update about the availability of all parking places and chooses the best one.

In order to investigate technologies behind the smart parking solution, we implemented an internal research project. The main idea was the creation of smart parking using the Internet of Things and ultrasonic sensors, where available parking places could be displayed in a web application.

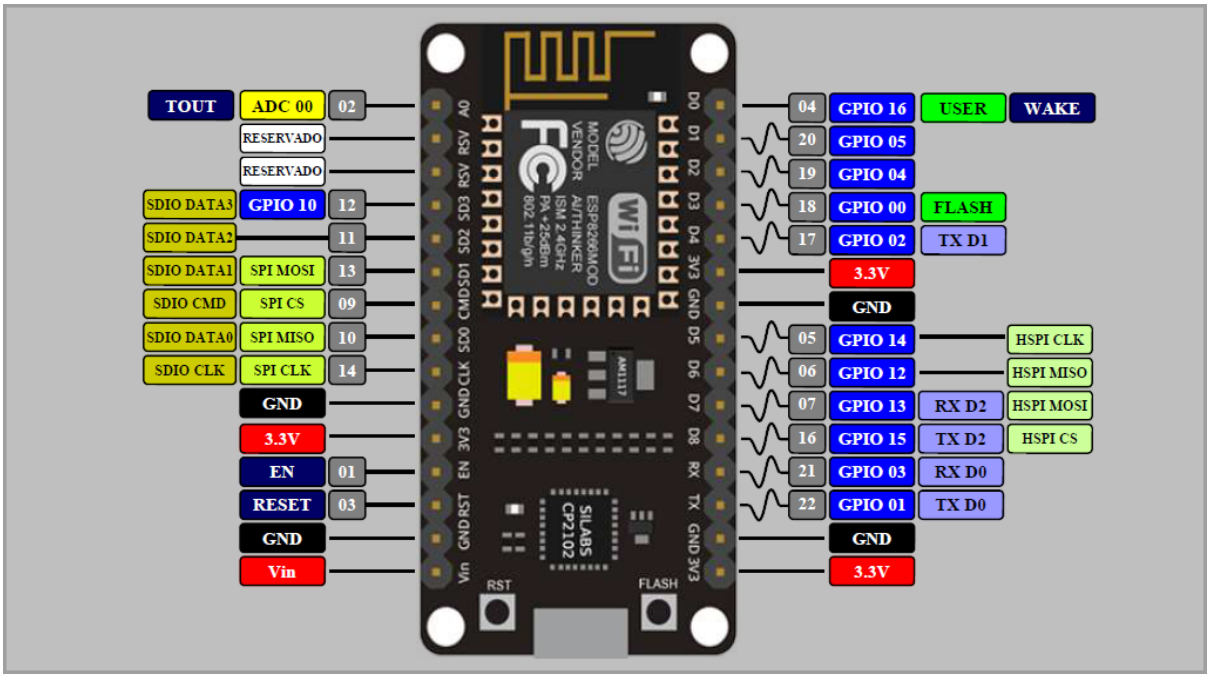
THEORY

Internet of Things

The concept of Internet of Things (IoT) started with things and identity communication devices. The devices could be tracked, controlled or monitored using remote computers connected through Internet. The internet of things has different definitions. In Short it is defined as the things present in the physical world or in an environment are attached with sensors or with any embedded systems and made connected to network via wired or wireless connections. These connected devices are called as smart devices or smart objects. And it consists of smart machines, which communicate, interact with other machines, environment, objects etc. And these can be processing by using some processors such as network processor, hybrid processor MCU/MPU etc. And the devices are connected by using some technologies called GPS, Wi-Fi, BT/BTLE, RFID etc. Internet of things was first introduced in 1999 at auto-ID centre and first used by Kevin ashton. This latest technology promises to connect all our surrounding things to a network and communicating with each other with less human involvement. Still internet of things is in beginning stage and there is no common architecture exist still today.

Node MCU:-

In this project we have used Node MCU (Micro Controller Unit). The Node MCU contains Wi-Fi as well as a Node microcontroller. Node Microcontroller has 3Mb of ROM which can be used to upload a program.



Pin Diagram of Node MCU

Node MCU is an impressive, low cost Wi-Fi module suitable for adding Wi-Fi functionality to an existing microcontroller project via a UART serial connection. The module can even be reprogrammed to act as a standalone Wi-Fi connected device–just add power.

The feature list is impressive and includes:

- 802.11 b/g/n protocol
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack

The Node MCU Wi-Fi Module is a self-contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your Wi-Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266

module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Arduino device and get about as much Wi-Fi-ability as a Wi-Fi Shield offers (and that's just out of the box)! The ESP8266 module is an extremely cost effective board with a huge, and ever growing, community. This module has a powerful enough on-board processing and storage capability that allows it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, including the front-end module, is designed to occupy minimal PCB area.

Light Dependent Resistor (LDR)

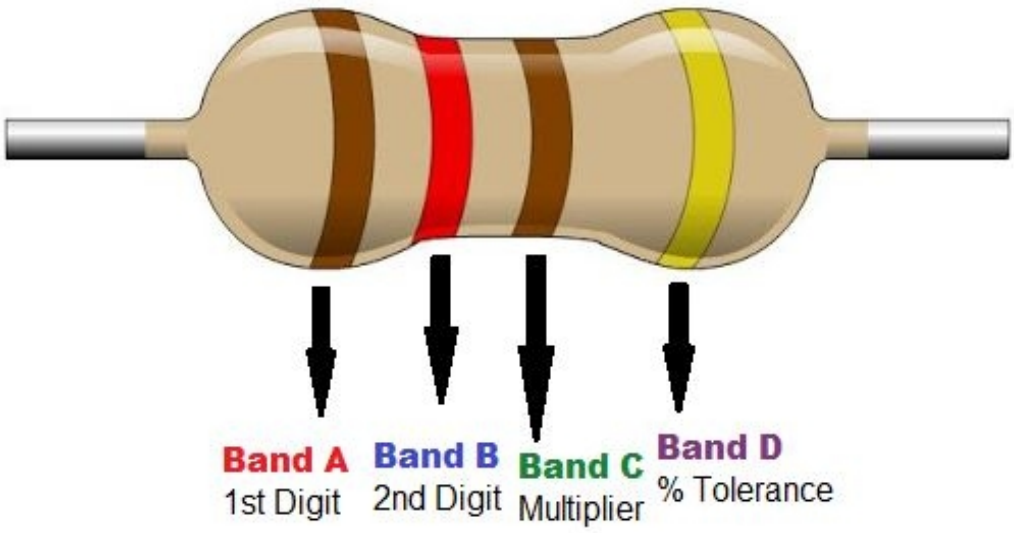
A light-dependent resistor, LDR (also known as a , photoresistor or photo-conductive cell) is a passive component that decreases resistance with respect to receiving luminosity (light) on the component's sensitive surface.
The resistance of a photoresistor decreases with increase in incident light intensity; in other words, it exhibits photoconductivity. A photoresistor can be applied in light-sensitive detector circuits and light-activated and dark-activated switching circuits acting as a resistance semiconductor. In the dark, a photoresistor can have a resistance as high as several megaohms (MΩ), while in the light, a photoresistor can have a resistance as low as a few hundred ohms. If incident light on a photoresistor exceeds a certain frequency, photons absorbed by the semiconductor give bound electrons enough energy to jump into the conduction band. The resulting free electrons (and their hole partners) conduct electricity, thereby lowering resistance. The resistance range and sensitivity of a photoresistor can substantially differ among dissimilar devices. Moreover, unique photoresistors may react substantially differently to photons within certain wavelength bands.

A photoelectric device can be either intrinsic or extrinsic. An intrinsic semiconductor has its own charge carriers and is not an efficient semiconductor, for example, silicon. In intrinsic devices, the only available electrons are in the valence band, and hence the photon must have enough energy to excite the electron across the entire bandgap. Extrinsic devices have impurities, also called dopants, added whose ground state energy is closer to the conduction band; since the electrons do not have as far to jump, lower energy photons (that is, longer wavelengths and lower frequencies) are sufficient to trigger the device. If a sample of silicon has some of its atoms replaced by phosphorus atoms (impurities), there will be extra electrons available for conduction. This is an example of an extrinsic semiconductor.

RESISTOR

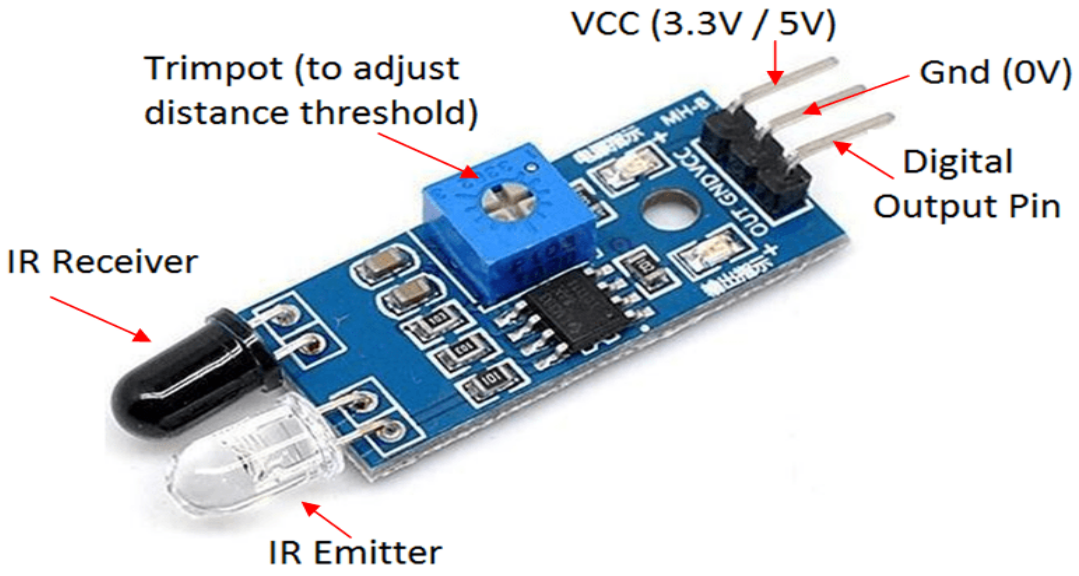
The resistor is a passive electrical component that creates resistance in the flow of electric current. In almost all electrical networks and electronic circuits they can be found. The resistance is measured in ohms (Ω). An ohm is the resistance that occurs when a current of one ampere (A) passes through a resistor with a one volt (V) drop across its

terminals. The current is proportional to the voltage across the terminal ends. This ratio is represented by Ohm's Law.



A resistor works by restricting the flow of current, it can do this in one of three ways: firstly, by using a less conductive material, secondly by making the conductive material thinner and finally by making the conductive material longer.

IR SENSOR



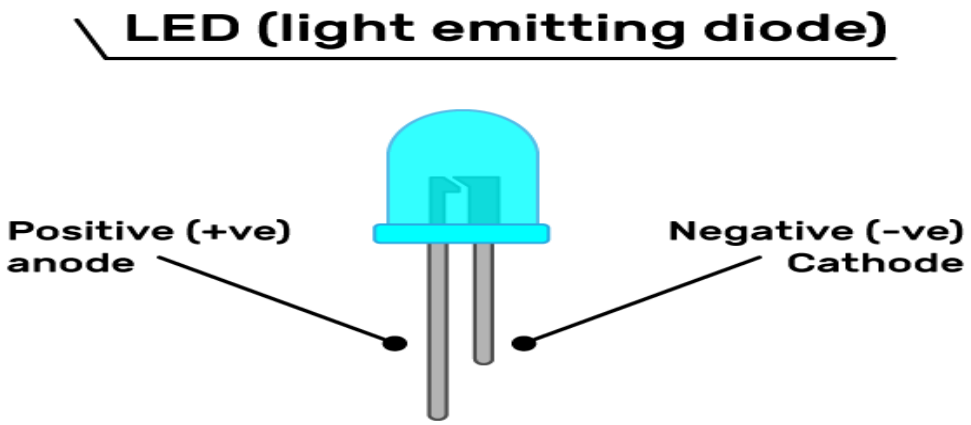
An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. Infrared radiation was accidentally discovered by an astronomer named William Herchel in 1800. While measuring the temperature of each color of light (separated by a prism), he noticed that the temperature just beyond the red light was highest. IR is invisible to the human eye, as its wavelength is longer than that of visible light.

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.

LED

In the simplest terms, a light-emitting diode (LED) is a semiconductor device that emits light when an electric current is passed through it. Light is produced when the particles that carry the current (known as electrons and holes) combine together within the semiconductor material.

Since light is generated within the solid semiconductor material, LEDs are described as solid-state devices. The term solid-state lighting, which also encompasses organic LEDs (OLEDs), distinguishes this lighting technology from other sources that use heated filaments (incandescent and tungsten halogen lamps) or gas discharge (fluorescent lamps).



SERVER

"A computer or computer program which manages access to a centralized resource or service in a network".

Basically server computer is a centralised computer which has static IP that is mapped with the DNS server. The server computer always connected with high speed internet connection with static IP. Whenever any client device is request for any webpage using URL and web browser the server computer is responsible to manage the request and provide the related data.

Server computer also contain data base which is used to store the data of the user and also used to identify the valid user info.

In this project we have used server from pcecarparking.com for testing purpose but according to the requirement we can take any server from any places on lease or we can setup own server.

We have used following web development languages to develop our web application.

- HTML
- Java Script
- PHP
- Jquery
- SQL

METHODOLOGY

Methodology is a model to explain the methods or techniques used to design, develop or plan a project. This chapter explains about the software and hardware that will be used for developing this project further. The results are going to be analyzed to achieve the objective of this project.

Smart parking platform is designed to avoid traffic congestion problems by guiding drivers the information about the empty parking slots. Ultrasonic sensors are equipped in each parking slot to detect whether a vehicle is parked in that particular parking slot or not. This information about the status of the parking lots is displayed in web browser to make it easy for the user as soon as they head nearby the area. The prototype is designed using Arduino UNO

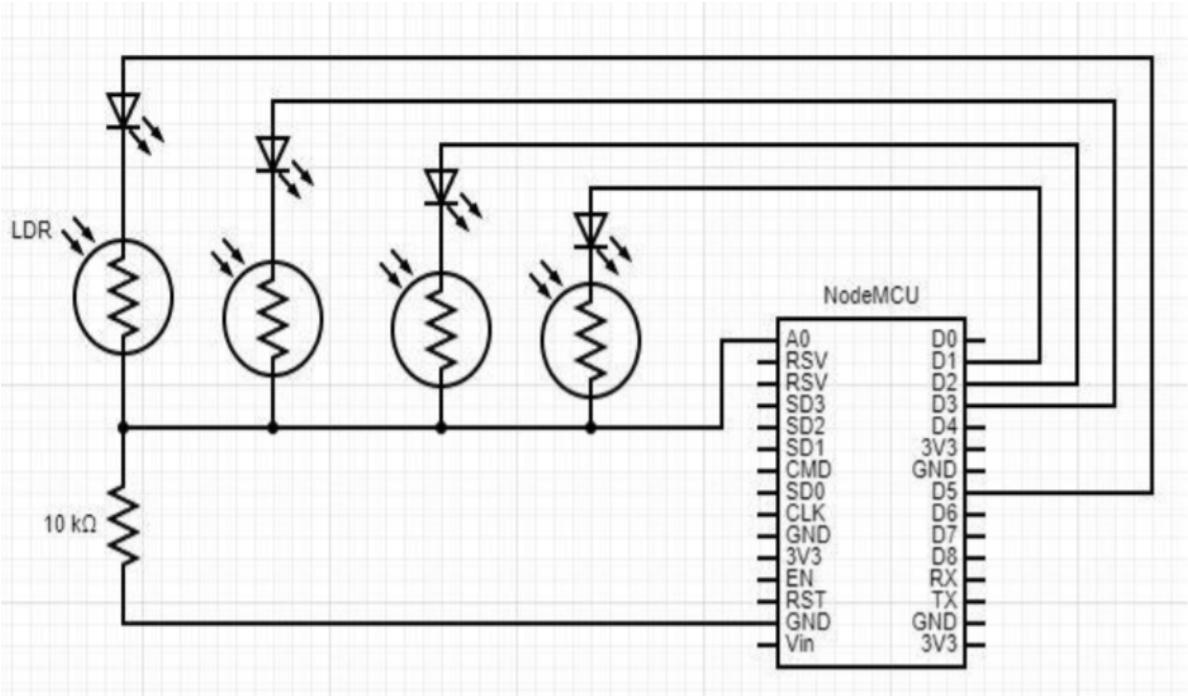
microcontroller and node MCU for Wi-Fi connectivity. The block diagram of the proposed project ‘smart parking platform’ is as shown in figure 1. There are three parking lots. Each of the parking lot has three parking spaces. Each of the parking space is equipped with one ultrasonic sensor. Each ultrasonic sensor is interfaced to microcontroller. Microcontroller sends high signal to trigger pin of ultrasonic sensor. The ultrasonic sensor will transmit eight pulses of 40kHz sound wave for 10µs. Now the echo pin of the sensor will go low. Sensor keeps sensing continuously for the obstacle. When an obstacle is detected, the sound waves reflect to the receiver and the echo pin goes high. This means that the parking slot is filled, else the sensor continues sensing for obstacle and the status of the parking slot is considered as empty Microcontroller is programmed in such a way that it sends a high pulse if the parking is filled to the Node MCU. The Node MCU keeps on checking the status of its GPIO pins and updates the webpage accordingly. The user can access the information regarding parking lots by entering the IP address of the respective node MCU in the web browser. The status of the parking slots is displayed in the webpage graphically. If a parking slot is empty, the block in the webpage is represented by green colour. If it is filled, it is represented by red block.

Overview

System Overview The proposed system is used to indicate the user about the vacancy of the parking slots. A user can choose the parking slot in advance, instead of waiting in area of the parking, where the parking availability are shown through user’s smart phones. IR Sensors will be attached in each slot for detecting the vacancy. The signal from the sensors captured by Arduino and these signal is then converted from electrical signal into another form to detect presence of vehicle in terms of the amount of light reflected back from the obstacle such as wall of the parking lot. The output from Arduino depends on the measurement of amount of light and based on that, slot’s allocation is done. On the other hand, the output from Arduino is changed into text format and sent to the smart phones through a developed Android application. Now the users are provided with the parking details and can choose the appropriate slots to reserve. Besides reservation, user also will be notified on details of parking such as extending or making payment via a simple text message with the help of GSM. The parking area are sensed by using the sensors which are placed in each slot. The sensors will detect each slot as input and the output of the sensors is preceded to the Arduino. Arduino will process the input of the

sensors, analog to digital conversion are made and by tracking the user using the details of the parking slots given to the user. The components needed for this project are sensing device, communication platform and mobile application. Figure shows the proposed block diagram of smart parking system.

BLOCK DIAGRAM



Referring to Fig above, I have shown only four LDRs, for simplicity. Go ahead and complete the circuit for 8 LDRs. Just connect each pin to each LDR, eg. D1 for first LDR, D2 for second LDR etc. In Fig 1 , I skipped D4 and used D5 instead. For your project you should also use D4 if you are going to build 8 parking lots.

In the actual carpark model for this project, we use 8 LDR.

ALGORITHM

Step 1: Send high trigger input signal to ultrasonic sensor.

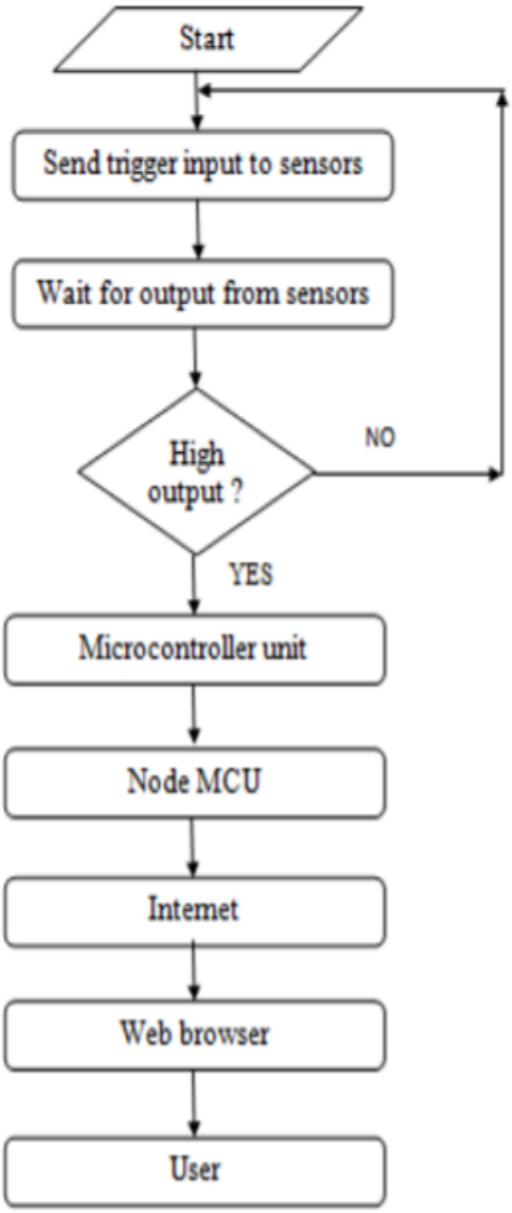
Step 2: Wait for echo output signal to go high.

Step 3: Send the output of echo to microcontroller.

Step 4: High pulse is sent to Node MCU.

Step 5: Node MCU checks for the status of its GPIO pins continuously.

Step 6: Update this status of the parking lot in the webpage accordingly.



SOFTWARE(CODE)

Sample Code:- 1

```
/*Reads values between 0-1023 for each LDR.
 * In calibration mode, it gets that max value
 * In non-calibration mode, it normalizes the value to 0.0 - 1.0
 * and sends data to server
*/
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

const char* ssid     = "iot-lab";
const char* password = "pervasive";
String ServerIP = "192.168.43.11"; //Paul Asus  MAC: 40:E2:30:CC:1E:8D
//String ServerIP = "192.168.1.88"; //Darren PC MAC: CC:B0:DA:CA:EA:D1


#define P1 D1
#define P2 D2
#define P3 D3
#define P4 D4
#define P5 D5
#define P6 D6
#define P7 D7
```

```
#define P8 D8



int P1MAX = 10000, P2MAX  = 10000, P3MAX  = 10000, P4MAX  =
10000, P5MAX = 10000, P6MAX  = 10000, P7MAX  = 10000, P8MAX  =
10000;
float P1NORM = 0, P2NORM = 0, P3NORM = 0, P4NORM = 0,
P5NORM = 0, P6NORM = 0, P7NORM = 0, P8NORM = 0;


int sensorValue = 0;


//--- Set by getCommand() ---
int CALSTATE = 0;
int cmd = 0;


void setup() {
  Serial.begin(9600);
  //MAC: B4:E6:2D:34:A4:A2
  //IP: 192.168.1.35


  Serial.println("MAC: " + WiFi.macAddress());
  pinMode(P1, OUTPUT);
  pinMode(P2, OUTPUT);
  pinMode(P3, OUTPUT);
  pinMode(P4, OUTPUT);
```

```
  pinMode(P5, OUTPUT);
  pinMode(P6, OUTPUT);
  pinMode(P7, OUTPUT);
  pinMode(P8, OUTPUT);
}


void setAllLow(){
  digitalWrite(P1, LOW);
  digitalWrite(P2, LOW);
  digitalWrite(P3, LOW);
  digitalWrite(P4, LOW);
  digitalWrite(P5, LOW);
  digitalWrite(P6, LOW);
  digitalWrite(P7, LOW);
  digitalWrite(P8, LOW);
}

void readPin(int pin){
  setAllLow();
  if(pin == 1) digitalWrite(P1, HIGH);
  if(pin == 2) digitalWrite(P2, HIGH);
  if(pin == 3) digitalWrite(P3, HIGH);
  if(pin == 4) digitalWrite(P4, HIGH);
  if(pin == 5) digitalWrite(P5, HIGH);
```

```
  if(pin == 6) digitalWrite(P6, HIGH);
  if(pin == 7) digitalWrite(P7, HIGH);
  if(pin == 8) digitalWrite(P8, HIGH);
  delay(50);
  sensorValue = analogRead(A0);
  if(CALSTATE == 1) {
    if(pin == 1) P1MAX = sensorValue;
    if(pin == 2) P2MAX = sensorValue;
    if(pin == 3) P3MAX = sensorValue;
    if(pin == 4) P4MAX = sensorValue;
    if(pin == 5) P5MAX = sensorValue;
    if(pin == 6) P6MAX = sensorValue;
    if(pin == 7) P7MAX = sensorValue;
    if(pin == 8) P8MAX = sensorValue;
  }

  if(pin == 1) {
    P1NORM = (float)sensorValue/P1MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P1MAX,
P1NORM);
  }
  if(pin == 2) {
    P2NORM = (float)sensorValue/P2MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P2MAX,
P2NORM);
```

```
  }
  if(pin == 3) {
    P3NORM = (float)sensorValue/P3MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P3MAX,
P3NORM);
  }
  if(pin == 4) {
    P4NORM = (float)sensorValue/P4MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P4MAX,
P4NORM);
  }
  if(pin == 5) {
    P5NORM = (float)sensorValue/P5MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P5MAX,
P5NORM);
  }
  if(pin == 6) {
    P6NORM = (float)sensorValue/P6MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P6MAX,
P6NORM);
  }
  if(pin == 7) {
    P7NORM = (float)sensorValue/P7MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P7MAX,
P7NORM);
  }
```

```
  if(pin == 8) {
    P8NORM = (float)sensorValue/P8MAX;
    Serial.printf("P%d: %d/%d %.2f\n", pin, sensorValue, P8MAX,
P8NORM);
  }
}

void loop() {
  //getCommandFromSerialMonitor();
  getCommandFromServer();
  for(int i=1; i<=8; i++) readPin(i);
  Serial.println("----------------------");

  sendToServer();
}

void sendToServer(){
  if(WiFi.status() != WL_CONNECTED ){
    Serial.println("Wifi not connected...");
    WiFi.begin(ssid, password);
  }

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
```

```
}

Serial.print("Connected to: "); Serial.println(WiFi.SSID());
Serial.print("Your IP: "); Serial.println(WiFi.localIP());
HTTPClient http;

String datatosend = "/parking/setdata.php?"
+ String("a=") + String(P1NORM) + String("&")
+ String("b=") + String(P2NORM) + String("&")
+ String("c=") + String(P3NORM) + String("&")
+ String("d=") + String(P4NORM) + String("&")
+ String("e=") + String(P5NORM) + String("&")
+ String("f=") + String(P6NORM) + String("&")
+ String("g=") + String(P7NORM) + String("&")
+ String("h=") + String(P8NORM);

Serial.print("Sensor values: "); Serial.println(datatosend);

http.begin(ServerIP, 80, datatosend);

int httpCode = http.GET();

if(httpCode > 0){
  Serial.printf("GET code: %d\n", httpCode);
  if(httpCode == HTTP_CODE_OK){
```

```
      String response = http.getString();
      Serial.println(response);
    }
  } else {
    Serial.printf("GET failed: error: %s\n",
http.errorToString(httpCode).c_str());
  }
  http.end();
}

void getCommandFromSerialMonitor(){
  String inStr = Serial.readStringUntil('\n');
   if(inStr.equals("caloff")){
        Serial.println("CAL OFF");
        CALSTATE = 0;
   }else if(inStr.equals("calon")){
       Serial.println("CAL ON");
       CALSTATE = 1;
   }
}

void getCommandFromServer(){
   if(WiFi.status() != WL_CONNECTED ){
    Serial.println("Wifi not connected...");
    WiFi.begin(ssid, password);
```

```
}

while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}

Serial.print("Connected to: "); Serial.println(WiFi.SSID());
Serial.print("Your IP: "); Serial.println(WiFi.localIP());
HTTPClient http;

String datatosend = "/parking/getcmd.php";

Serial.print("Get cmd: "); Serial.println(datatosend);

http.begin(ServerIP, 80, datatosend);

int httpCode = http.GET();

if(httpCode > 0){
  Serial.printf("GET code: %d\n", httpCode);
  if(httpCode == HTTP_CODE_OK){
    String response = http.getString();
    CALSTATE = response.toInt();
    Serial.println(response);
```

```
      if(CALSTATE==0){
        Serial.println("CAL OFF");
       }else if(CALSTATE==1){
        Serial.println("CAL ON");
      }
    }
  } else {
   Serial.printf("GET failed: error: %s\n",
http.errorToString(httpCode).c_str());
  }
  http.end();
}
```

Sample Code:-2

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>IOT Car Park System</title>

    <style>
      .item1 {
```

```
  grid-area: header;
 }
 .item2 {
  grid-area: menu;
 }
 .item3 {
  grid-area: main;
 }
 .item4 {
  grid-area: right;
 }
 .item5 {
  grid-area: footer;
 }

 .grid-container {
  display: grid;
  grid-template-areas:
    'header header header header header header header header header header header'
    'menu main main main main main main main main main main'
    'menu main main main main main main main main main main';
  grid-gap: 10px;
  background-color: lightblue;
  padding: 10px;
```

```
  margin-left: 5vw;
  margin-right: 5vw;
}

.grid-container > div {
  background-color: rgba(255, 255, 255, 0.8);
  text-align: center;
  font-size: 18x;
}
.parking-grid-container {
  display: grid;
  grid-template-columns: auto auto auto auto;
  margin-left: 3vw;
  padding-top: 0px;
  padding-bottom: 3vw;
  padding-left: 3vw;
}
.parking-grid-item {
  background-color: lightgreen;
  border: 2px solid rgba(0, 0, 0, 0.8);
  width: 6vw;
  height: 8vw;
  font-size: 3vw;
  text-align: center;
  padding-top: 3vw;
```

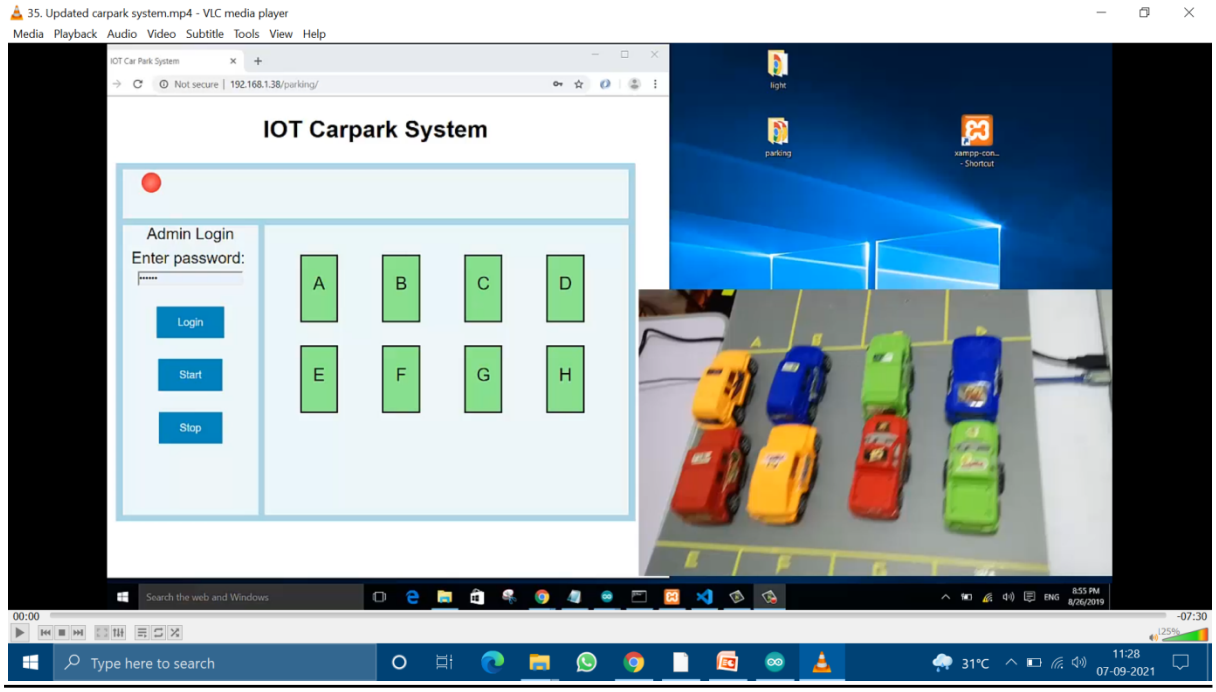
```
}
.lane {
  height: 1vw;
}
.column {
  float: left;
  width: 10%;
  padding: 5px;
}

/* Clearfix (clear floats) */
.row::after {
  content: '';
  clear: both;
  display: table;
}
body {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 24px;
}

.button {
  border: none;
  color: white;
  padding: 15px 32px;
```

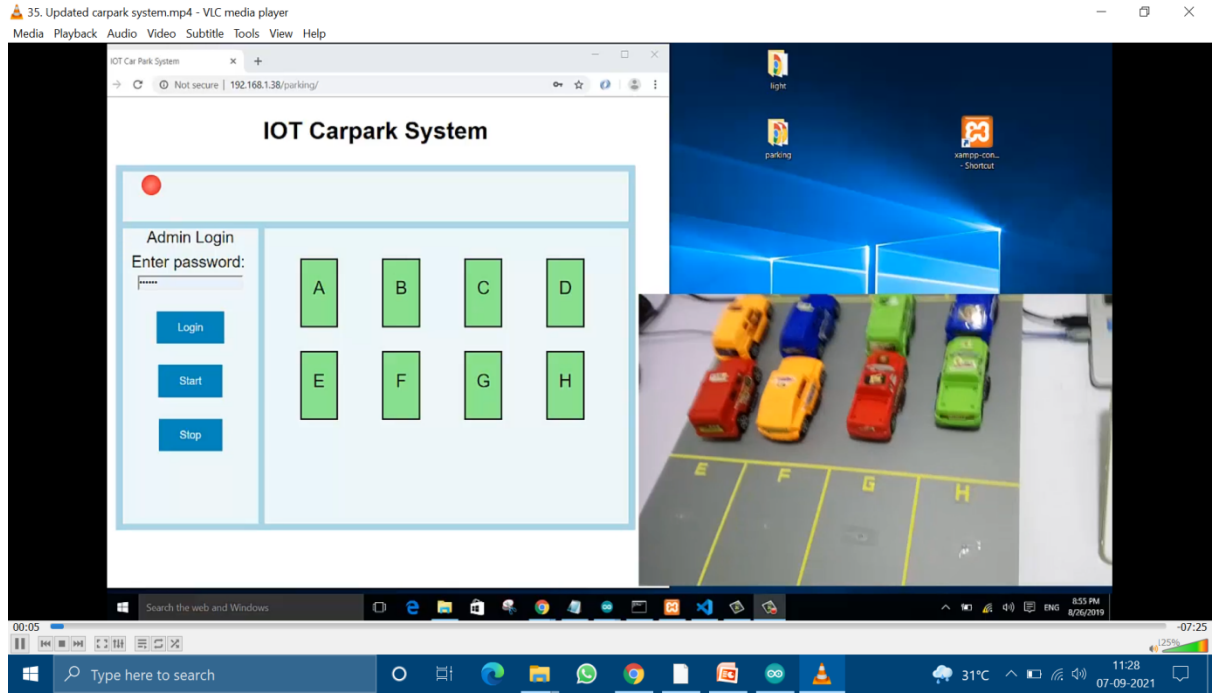
```
      text-align: center;
      text-decoration: none;
      display: inline-block;
      font-size: 16px;
      margin: 4px 2px;
      background-color: #008CBA;
    }
    .controlbutton, .statustext{
      visibility: hidden;
    }


  </style>
 </head>
 <body>
  <div style="text-align: center"><h2>IOT Carpark System</h2></div>

  <div class="grid-container">
    <div class="item1">
      <div class="row">
        <div class="column" id="light-status">
          <img src="images/red.png" alt="" width="30vw" id="light-img"/>
        </div>
```

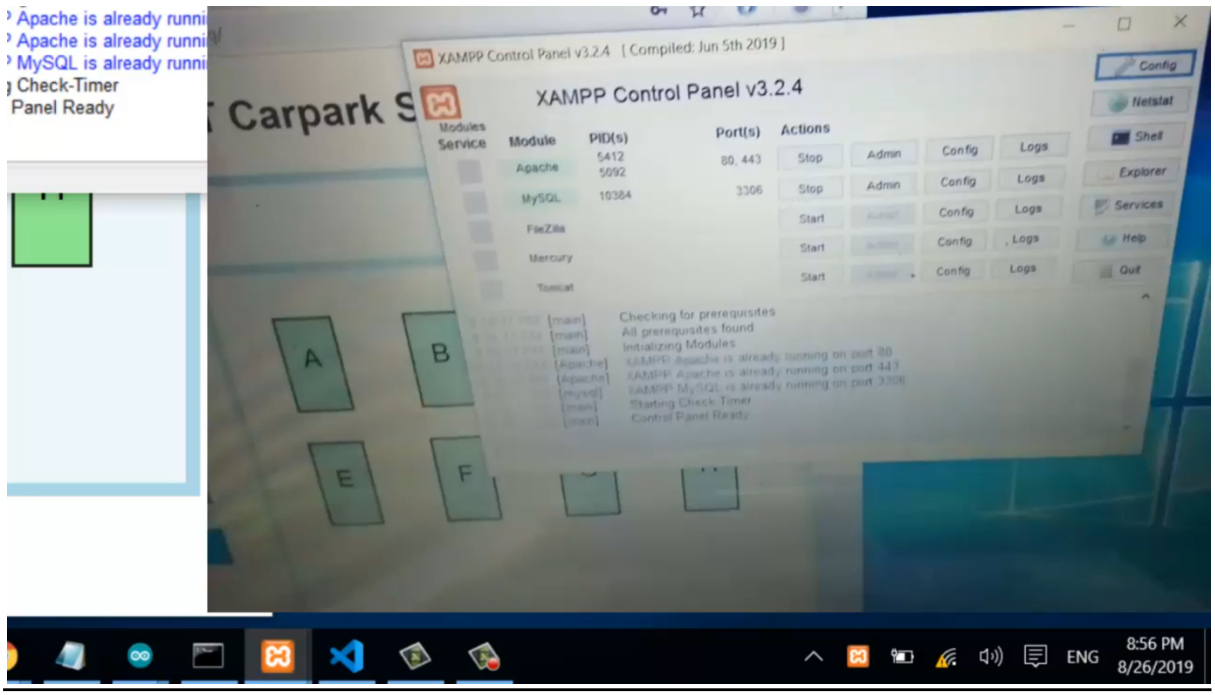
```
        <div class="column statustext" id="status-a">a</div>
        <div class="column statustext" id="status-b">b</div>
        <div class="column statustext" id="status-c">c</div>
        <div class="column statustext" id="status-d">d</div>
        <div class="column statustext" id="status-e">e</div>
        <div class="column statustext" id="status-f">f</div>
        <div class="column statustext" id="status-g">g</div>
        <div class="column statustext" id="status-h">h</div>
        <div class="column statustext" id="status-cmd">cal</div>
        <div class="column statustext" id="status-sens">sens</div>
      </div>
    </div>
    <div class="item2">
      Admin Login<br />
      <div style="margin-top: 10px" id="passworddiv">
        Enter password:
        <br><input type="password" id="passwordinput">
      </div>
      <br><button class="button" id="loginbutton"
onclick="doLoginLogoff()">Login</button>

      <p><button class="button"
onclick="startTimer()">Start</button></p>
      <p><button class="button"
onclick="stopTimer()">Stop</button></p>
```

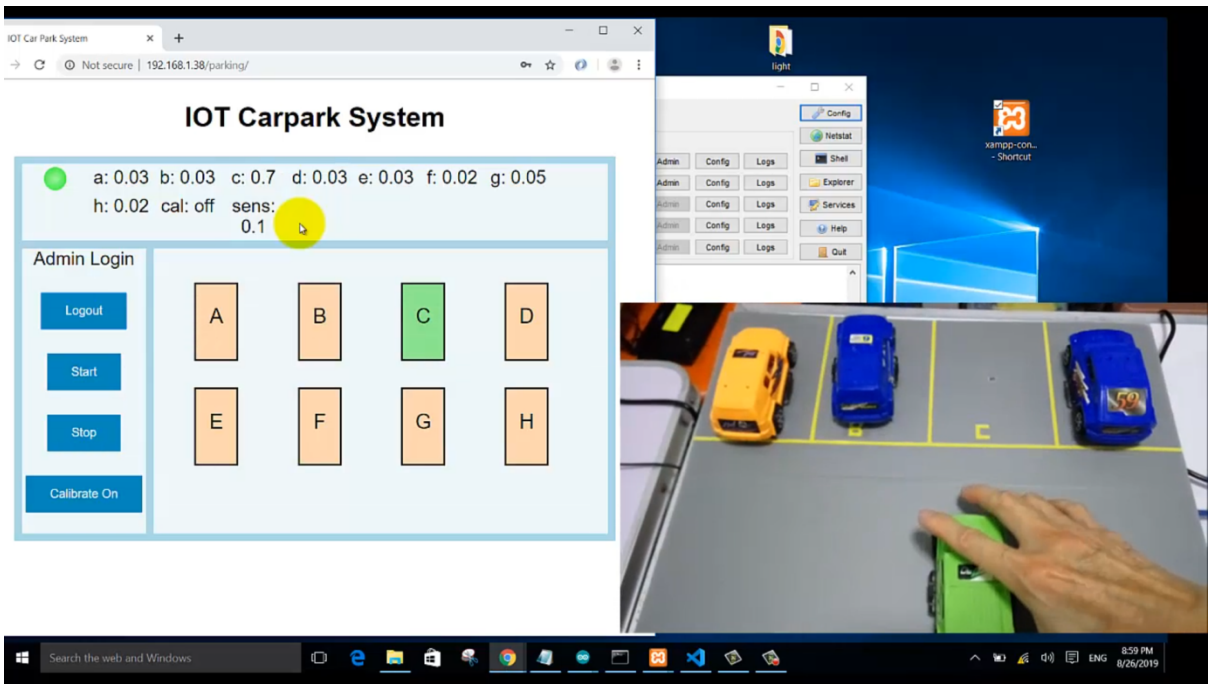
```
      <p><button class="button controlbutton" id="calbutton"
onclick="calibrateOnOff()">Calibrate On</button></p>


    </div>
    <div class="item3">
     <div class="parking-grid-container" style="margin-top: 5vw">
      <div class="parking-grid-item" id="a">A</div>
      <div class="parking-grid-item" id="b">B</div>
      <div class="parking-grid-item" id="c">C</div>
      <div class="parking-grid-item" id="d">D</div>
     </div>
     <div class="lane"></div>
     <div class="parking-grid-container">
      <div class="parking-grid-item" id="e">E</div>
      <div class="parking-grid-item" id="f">F</div>
      <div class="parking-grid-item" id="g">G</div>
      <div class="parking-grid-item" id="h">H</div>
     </div>
    </div>
   </div>
   <script src="jquery-3.4.0.min.js"></script>
   <script src="app.js"></script>
  </body>
</html>
```

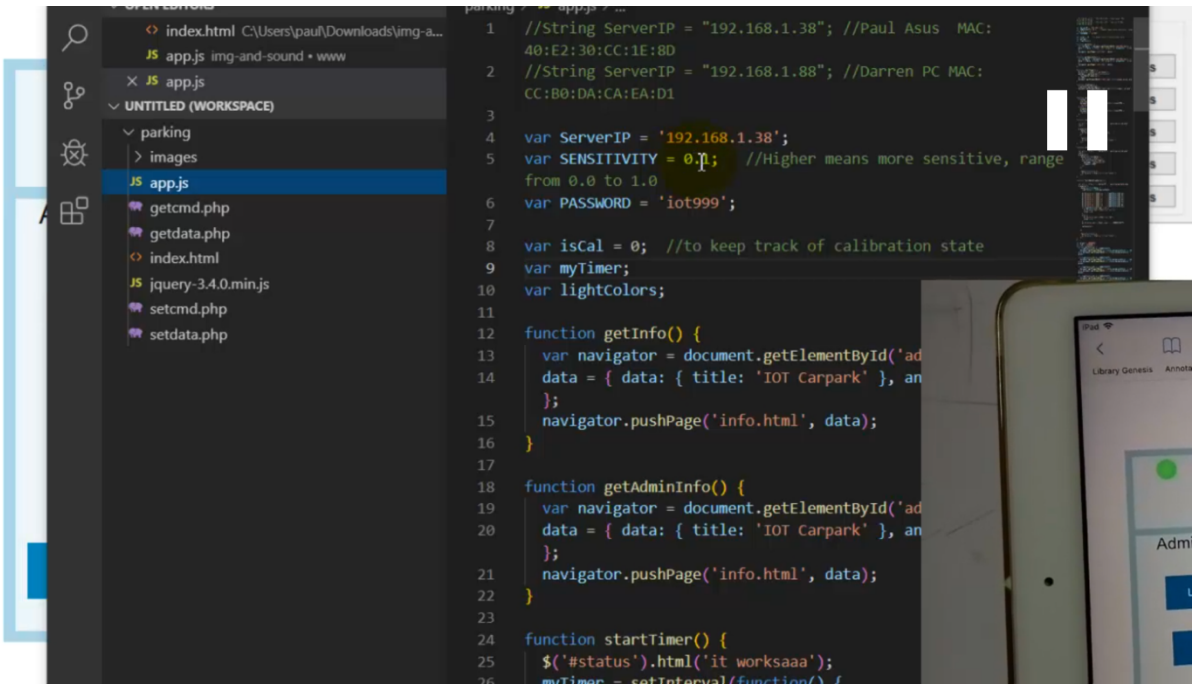
PROJECT OUTPUT

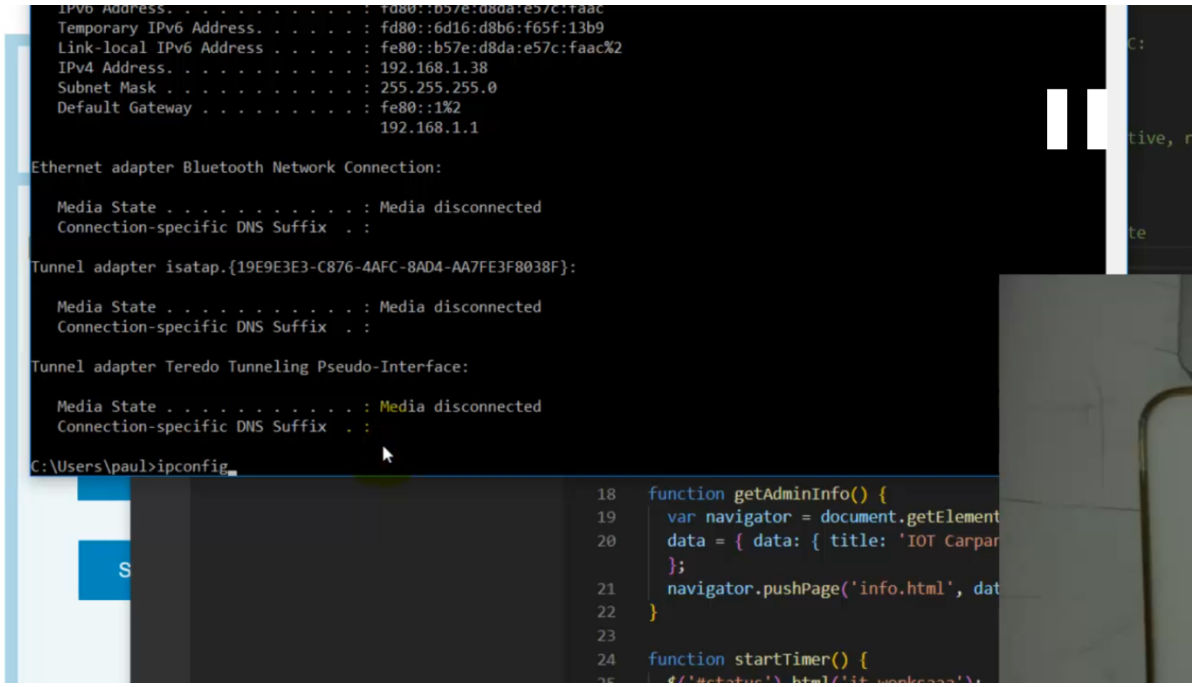





Apache is already runni
Apache is already runni
MySQL is already runni
Check-Timer
Panel Ready
XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]
XAMPP Control Panel v3.2.4
Module
PID(s)
Port(s)
Actions
Apache
MySQL
Stop
Admin
Config
Logs
Netstat
Shell
Explorer
Services
Help
Quit
A
B
E
F
ENG
8:56 PM
8/26/2019


IOT Car Park System
Not secure | 192.168.1.38/parking/
IOT Carpark System
a: 0.03 b: 0.03 c: 0.7 d: 0.03 e: 0.03 f: 0.02 g: 0.05
h: 0.02 cal: off sens:
0.1
Admin Login
Logout
Start
Stop
Calibrate On
A
B
C
D
E
F
G
H
Search the web and Windows
ENG
8:59 PM
8/26/2019


index.html C:\Users\paul\Downloads\img-a...
app.js img-and-sound • www
app.js
UNTITLED (WORKSPACE)
parking
images
app.js
getcmd.php
getdata.php
index.html
jquery-3.4.0.min.js
setcmd.php
setdata.php
//String ServerIP = "192.168.1.38"; //Paul Asus MAC: 40:E2:30:CC:1E:8D
//String ServerIP = "192.168.1.88"; //Darren PC MAC: CC:B0:DA:CA:EA:D1
var ServerIP = '192.168.1.38';
var SENSITIVITY = 0.1; //Higher means more sensitive, range from 0.0 to 1.0
var PASSWORD = 'iot999';
var isCal = 0; //to keep track of calibration state
var myTimer;
var lightColors;
function getInfo() {
var navigator = document.getElementById('ad
data = { data: { title: 'IOT Carpark' }, an
};
navigator.pushPage('info.html', data);
}
function getAdminInfo() {
var navigator = document.getElementById('ad
data = { data: { title: 'IOT Carpark' }, an
};
navigator.pushPage('info.html', data);
}
function startTimer() {
$('#status').html('it worksaaa');


Temporary IPv6 Address. . . . . : fd80::6d16:d8b6:f65f:13b9
Link-local IPv6 Address . . . . . : fe80::b57e:d8da:e57c:faac%2
IPv4 Address. . . . . : 192.168.1.38
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : fe80::1%2
192.168.1.1
Ethernet adapter Bluetooth Network Connection:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
Tunnel adapter isatap.{19E9E3E3-C876-4AFC-8AD4-AA7FE3F8038F}:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
Tunnel adapter Teredo Tunneling Pseudo-Interface:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
C:\Users\paul>ipconfig
function getAdminInfo() {
var navigator = document.getElement
data = { data: { title: 'IOT Carpar
};
navigator.pushPage('info.html', dat
}
function startTimer() {

Results & Discussion

The below table indicates the status of the parking slots filled or vacant. Depending on the remark table status, parking place can be chosen without wasting searching time for the parking place. From the above table 1 Indicates parking slots 2 and 3 are filled and parking slot 1 is vacant for the parking. Similarly, in table 2 parking slot 2 is filled and 1, 3 are not filled they are available for the parking. The table 3 indicates all the parking slots are filled and there is no vacant place for parking.

Table 1: Display of Parking slots status and Remarks: Case1

Parking Slots	Status	Remark
1	Not Filled	Vacant Place at Slot 1
2	Filled	No Vacant Place
3	Filled	No Vacant Place

Table 2: Display of Parking slots status and Remarks: Case2

Parking Slots	Status	Remark
1	Not Filled	Vacant Place at Slot 1
2	Filled	No Vacant Place
3	Not Filled	Vacant Place at Slot 3

Table 3: Display of Parking slots status and Remarks: Case3

Parking Slots	Status	Remark
1	Filled	No Vacant Place
2	Filled	No Vacant Place
3	Filled	No Vacant Place

Advantage

It is **well managed to access and map the status of parking slots from any remote location through web app**. Thus, it reduces the time of finding the parking slots in any parking area and also it eliminates unnecessary travelling of vehicles across the filled parking slots in a city.

Limitations

In contrast to traditional PMS that provided only basic services, smart PMS are envisioned to offer a whole range of advanced services with a highly cross-functional management tool. Some features include:

- Parking availability monitoring ranging from general to granular information.
- Space optimization.

- Parking guidance and search time reduction based on operator interest, user preference, and so on.
- Parking reservation.
- Parking demand management
- Parking price and policy optimization (mostly dynamic).
- Parking enforcement such as detection of zones, payment, and overstay violations.
- Routine services such as reporting, integrated payment support (through mobile apps featuring pay-by-text, pay-by-voice, payby-phone, and other such functionalities), and system management in terms of configuration and maintenance.

Conclusion

Business 4.0, which is integral to Parking 4.0, works on the principle of ecosystems and a sense of abundance. In this context, the future of parking will emerge as an ecosystem play, comprising individuals, parking operators, automotive OEMs, solution integrators, and other stakeholders. The sense of abundance will come from the provision of 'right parking at the right place at the right price'. We believe that Parking 4.0 has the potential to be a major game-changer not only in the smart parking segment, but also in the overall mobility ecosystem of present cities and smart cities of the future. By leveraging IPX, Parking 4.0 is poised to lead to more advanced outcomes.

References

1) https://www.google.com\

2) https://www.udemy.com/course/iot-internet-of-things-automation-with-esp8266/learn/lecture/12065946?start=15#content

3) https://nodemcu.readthedocs.io/en/release/

4) https://www.elprocus.com/infrared-ir-sensor-circuit-and-working/

5) https://en.wikipedia.org/wiki/Photoresistor

6) https://en.wikipedia.org/wiki/Light-emitting_diode

7) https://en.wikipedia.org/wiki/Internet_of_things