



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

**School of Computer Science and
Engineering
CSE2006-Microprocessor and Interfacing**

Fall 2021-22

Slot: - G2

Smart Parking Management System

(A Microcontroller based Parking Management System.)

Submitted By:

Team Member

Prithak Gajurel (20BCE2921)

Pratik Luitel (20BCE2897)

Bijan Shrestha (20BCE2904)

Anurag Karki (20BCE2907)

Sandesh Khatiwada (20BCE2898)

SUBMITTED TO:

PROF. Ms. Saranya K.C

Asst. Prof.

School of Electronics Engineering

VIT University

Vellore-14, INDIA

ACKNOWLEDGEMENTS

We would like to express our deep gratitude to **PROF. Ms. Saranya K.C** (Asst. Prof.) , our project guide, for his patient guidance, enthusiastic encouragement and useful critiques of this project work.

We would also like to thank SCOPE department of VIT University for providing us this course which help us fulfil our Degree of Bachelor of technology in Computer and Science Engineering.

TABLE OF CONTENT

1. ABSTRACT
2. INTRODUCTION
3. LITERATURE SURVEY
4. AIM
5. OBJECTIVE
6. EXISTING AND PROPOSED SYSTEM
7. COMPONENTS OF PROPOSED SYSTEM
8. CIRCUIT DIAGRAM
9. WORKING AND ALGORITHM
10. RESULT
11. CONCLUSION
12. REFERENCE
13. CODE

ABSTRACT

With growing, Vehicle parking increases with the number of users. With the increased use of smartphones and their applications, users prefer mobile phone-based solutions. One of the most important problems facing large cities is congestion and parking. So, using Automated Parking System Management is an efficient technique using the Internet of Things to manage the garage. This paper proposes the Smart Parking Management System (SPMS) that depends on Arduino parts, Android applications, modules, sensor and based on IoT. This gave the client the ability to check available parking spaces and reserve a parking spot. IR sensors are utilized to know if a car park space is allowed. Its area data are transmitted using the WI-FI module to the server and are recovered by the mobile application which offers many options attractively and with no cost to users and lets the user check reservation details. With IoT technology, the smart parking system can be connected wirelessly to easily track available locations.

INTRODUCTION

The number of vehicle client's increases was requested more parking spots, and with the growth of the internet of things causes smart urban areas to have picked up grind popularity. In this way, issues, for example, traffic blockage, constrained vehicle leaving offices, and street security are being tended to by IoT. So, several parking organization systems have been organized to decrease such traffic issues and improve the comfort of vehicle users, it has combined.

Smart parking management system is a classic example demonstrates how the Internet-of-Things will be effectively and efficiently used to make life easy for a common citizen. Smart parking is an electronic tool that enables the user to find vacant parking spaces through information technology and by using appropriate sensors. Sensors are deployed in smart systems, which in turn collect information from the device for processing and analysis .So, Sensors would be deployed in the parking area and through the mobile application for helping the user to know the freedom of parking places on a real-time basis with more efficiency, and less cost. A smart parking system reduces the time to locate available places and reduces fuel consumption. The paper is organized as follows: First, it presents the concept of the smart parking system and its various functions, then its reviews previous research and studies on the implementation of smart parking. Then it describes the system implementation and operation and gives a conclusion of the smart parking application.

LITERATURE SURVEY

[1] Camille Persson et-al proposes a Multi-Agent organization for expressing the governance strategy of such systems. The smart parking management application is illustrated using the MOISE organization framework. The next generation Smart Cities will provide automated service to improve the life of citizens. The Machine-to- Machine (M2M) paradigm involves devices like sensors and actuators interacting together to provide services located in the real world. The Sencity project proposes an infrastructure to enable shared city scale applications. Multi-Agent technologies grant adaptability, flexibility and proactivity properties.

[2] Abhirup Khanna et-al proposed this paper on Smart Parking System that is implemented using a mobile application that is connected to the cloud. The system helps a user know the availability of parking spaces on real time basis. Factors that led to amalgamation of Cloud and IoT are storage capacity, computation power, communication resources, scalability, availability and interoperability. Parking Sensors like Ultrasonic Sensors are used this project. They detect the presence of a car. The ultrasonic sensors are wirelessly connected to raspberry pi using the ESP8266 chip. The advantage of using this mobile application is that users from remote locations could book.

[3] Vanessa W.s Tang et-al proposed a WSN-based intelligent car parking system, low-cost wireless sensors are deployed into a car park field, with each parking lot equipped with one sensor node, which detects and monitors the occupation of the parking lot. The status of the parking field detected by sensor nodes is reported periodically to a database via the deployed wireless sensor network and its gateway. The database can be accessed by the upper layer management system to perform various management functions, such as finding vacant lots, auto-toll, security management, and statistic report.

[4] Mano Suruthi et-al proposes to make a model of a reservation-based framework that is outfitted with sensors, 16x2 LCD (for display of spots) and LEDs. It gives a choice to preserve utilizing a versatile mobile application based on android that can be accessed by the client. The application demonstrates the quantity of accessible and inaccessible openings. On pre- booking an empty slot a notice is sent to the enrolled client. In the course of recent decades, traffic experts and authorities in numerous urban communities have grown purported Parking Guidance and Information (PGI) systems better parking administration.

[5] Sayanti Banerjee et-al proposes a new system for providing parking information and guidance using image processing. The proposed system includes counting the number of parked vehicles, and identifying the stalls available. The system detects cars through images instead of using electronic sensors embedded on the floor. A camera is installed at the entry point of the parking lot. It will capture image sequences. Setting image of a as reference image, the captured images are sequentially matched.

AIM

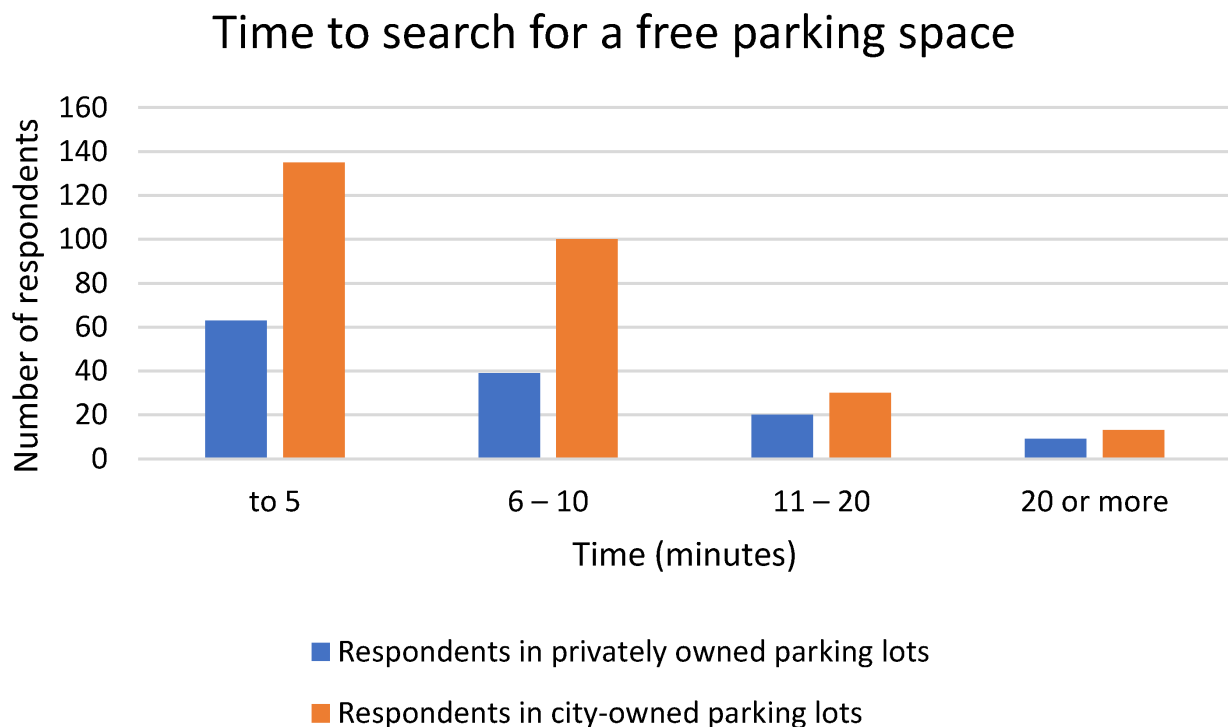
The smart parking Management system is an integrated system to recognize the nearest available parking zone. So, the main purpose of the system is to develop an Arduino based Smart Parking Management System using different modules and sensors.

OBJECTIVE

The main objective of the project is to provide a solution to the parking problem, to reduce the time to search for parking lots, to eliminate unnecessary travel for vehicles, to save money and make it easier for customers.

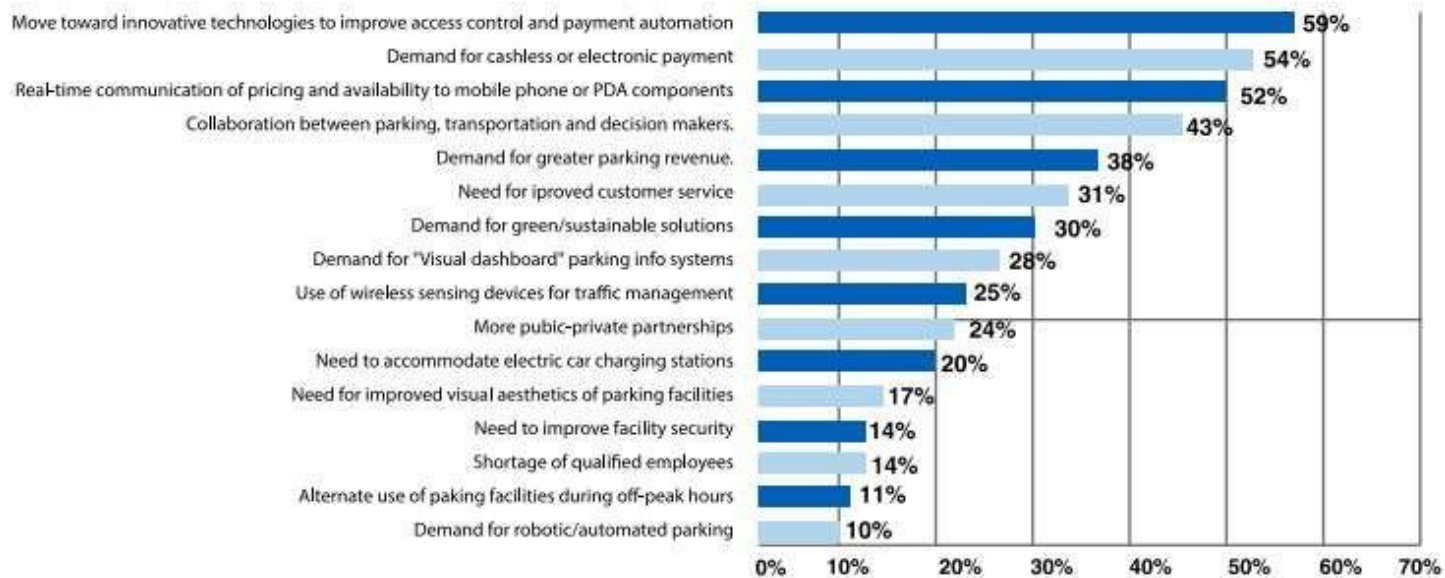
EXISTING AND PROPOSED SYSTEM

Traffic congestion caused by vehicle is an alarming problem at a global scale and it has been growing exponentially. Vehicle parking problem is a major contributor and has been, still a major problem with increasing vehicle size in the luxurious segment and confined parking spaces in urban cities. Searching for a parking space is a routine (and often frustrating) activity for many people in cities around the world. This search burns about one million barrels of the world's oil every day. As the global population continues to urbanize, without a well-planned, convenience-driven retreat from the car problems will worsen.



Source: MDPI (Department of Road and Urban Transport, University of Žilina)

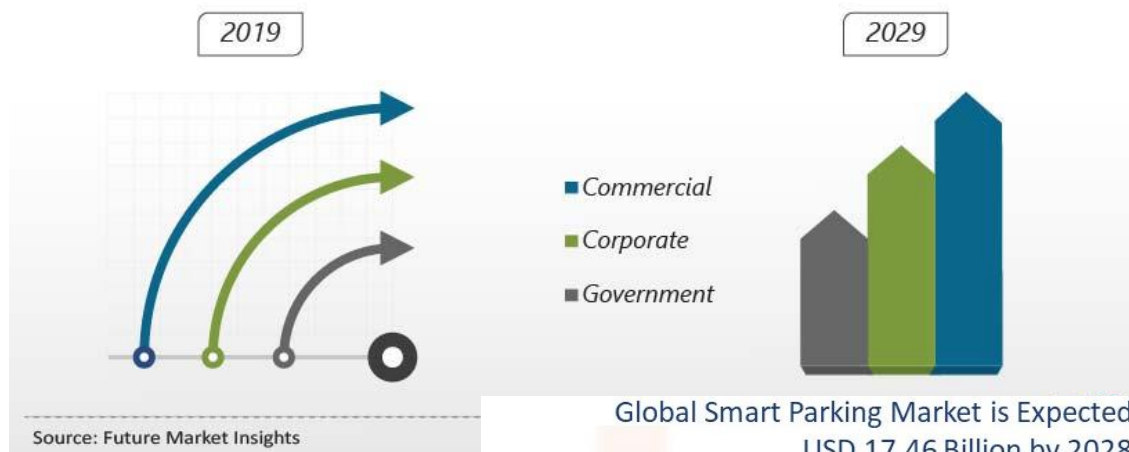
Time Taken to Search Parking



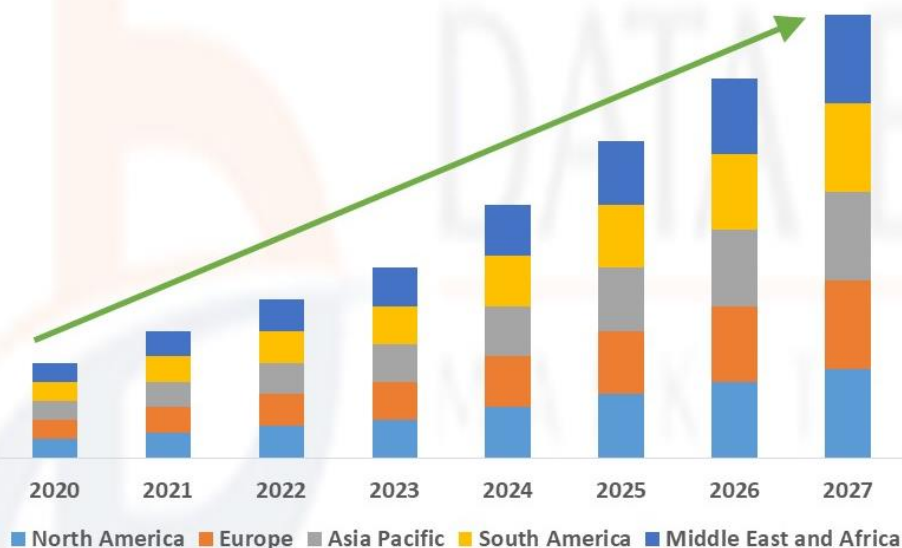
Source: Google

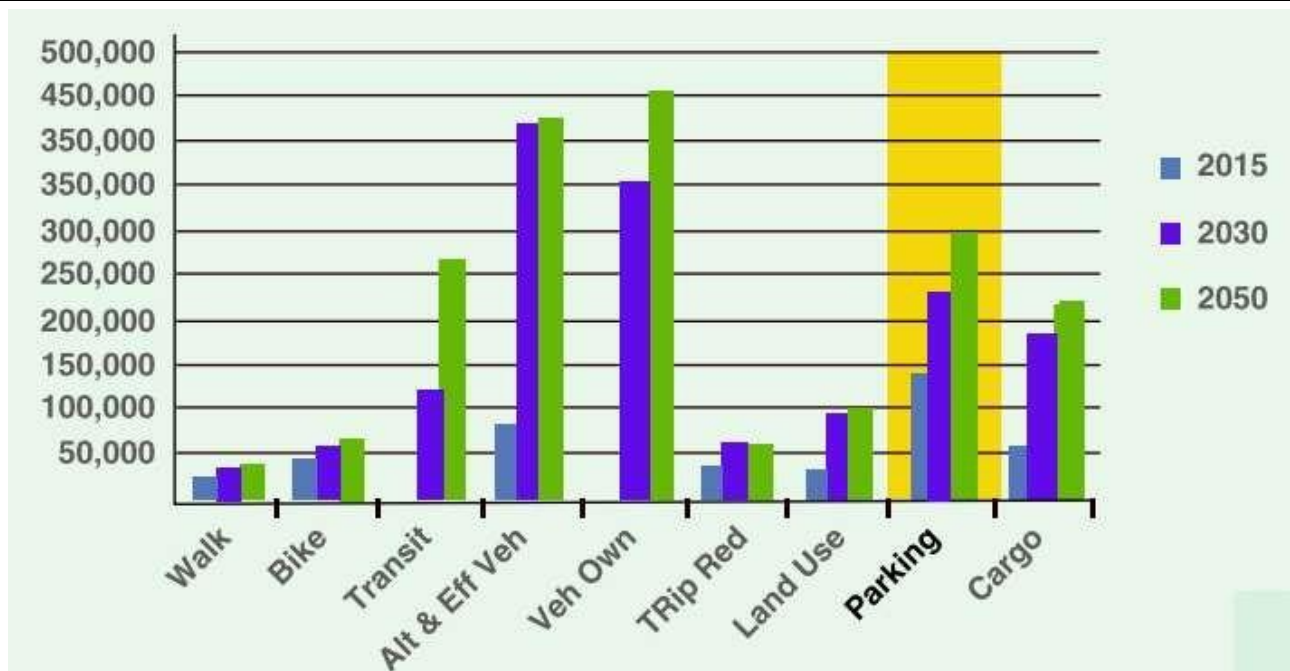
Trends having the greatest effect on Parking Industry

Smart Parking Application Assessment



Future Prediction of having Smart Parking system





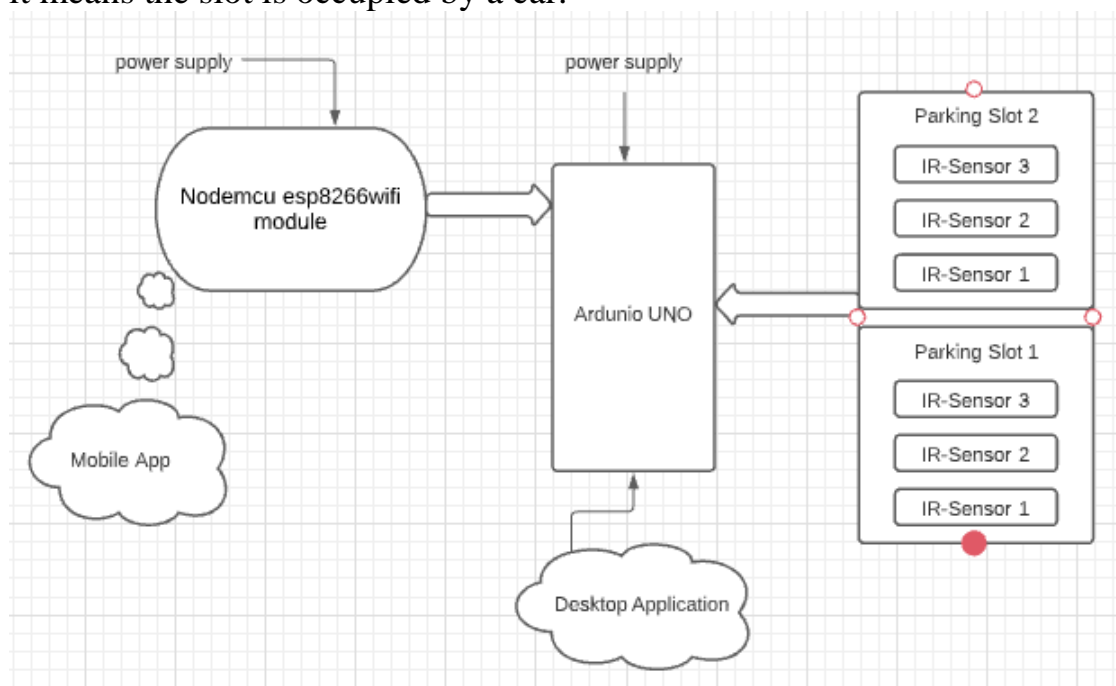
According to a report, Smart Parking could result in 2,20,000 gallons of fuels saving till 2030 and approx. 3,00,000 gallons of fuels saved by 2050, if implemented successfully

The project will accomplish as follows:

In this Smart Parking System, the infrared sensors will detect the presence/absence of a car which is interfaced with the LEDs that displays the availability of vacant spot.

Two LEDs will be used to indicate the Vacant/Taken slots where green light will indicate the vacant spot. This is also a method to calculate how long each user has been parked.


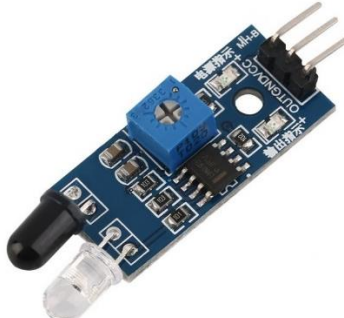
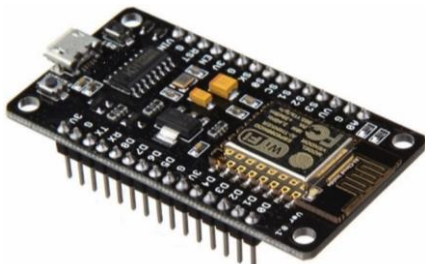
With the help of the Nodemcu esp8266 wifi module and Blynk application, the parking slots can be monitored from anywhere around the world. Car parking Slots is also monitored using a computer application designed in Visual Basic .net which is also known as vb.net. Depending on the detection of the car the box next to the slot is checked or unchecked. If the box is checked it means the slot is occupied by a car.



COMPONENTS OF PROPOSED SYSTEM

The proposed system works through a set of commands within the Arduino and it needs hardware components to work suitably.

Hardware Components

- **Arduino UNO:** It is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.
- 
- The image shows an Arduino UNO microcontroller board. It is a blue printed circuit board (PCB) with various electronic components. Key features include a USB Type-B port on the left, a DC power jack, a reset button, and a central ATmega328P microcontroller. The board has two rows of pins: digital pins on the top and analog pins on the bottom.
- **IR Sensor:** It is an electronic device that emits the light in order to sense some object of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations.
- 
- The image shows an IR sensor module. It is a small blue PCB with a black cylindrical lens at the front. It has three pins: VCC, GND, and OUT. The module is used for detecting objects without physical contact.
- **Nodemcu ESP8266 WIFI Module:** NodeMCU is an open source development board and firmware based in the widely used ESP8266 -12E WiFi module. It allows you to program the ESP8266 WiFi module with the simple and powerful LUA programming language or Arduino IDE. With just a few lines of code you can establish a WiFi connection and define input/output pins according to your needs exactly like arduino, turning your ESP8266 into a web server and a lot more. It is the WiFi equivalent of ethernet module.
- 
- The image shows a NodeMCU ESP8266 WIFI Module. It is a small black PCB with a USB Type-C port on the left. It features a microcontroller, a USB-to-UART bridge, and various other components. The module is used for connecting microcontrollers to a network via WiFi.
- **Others:** Connecting wires, Breadboard, USB cable etc.

Software Component

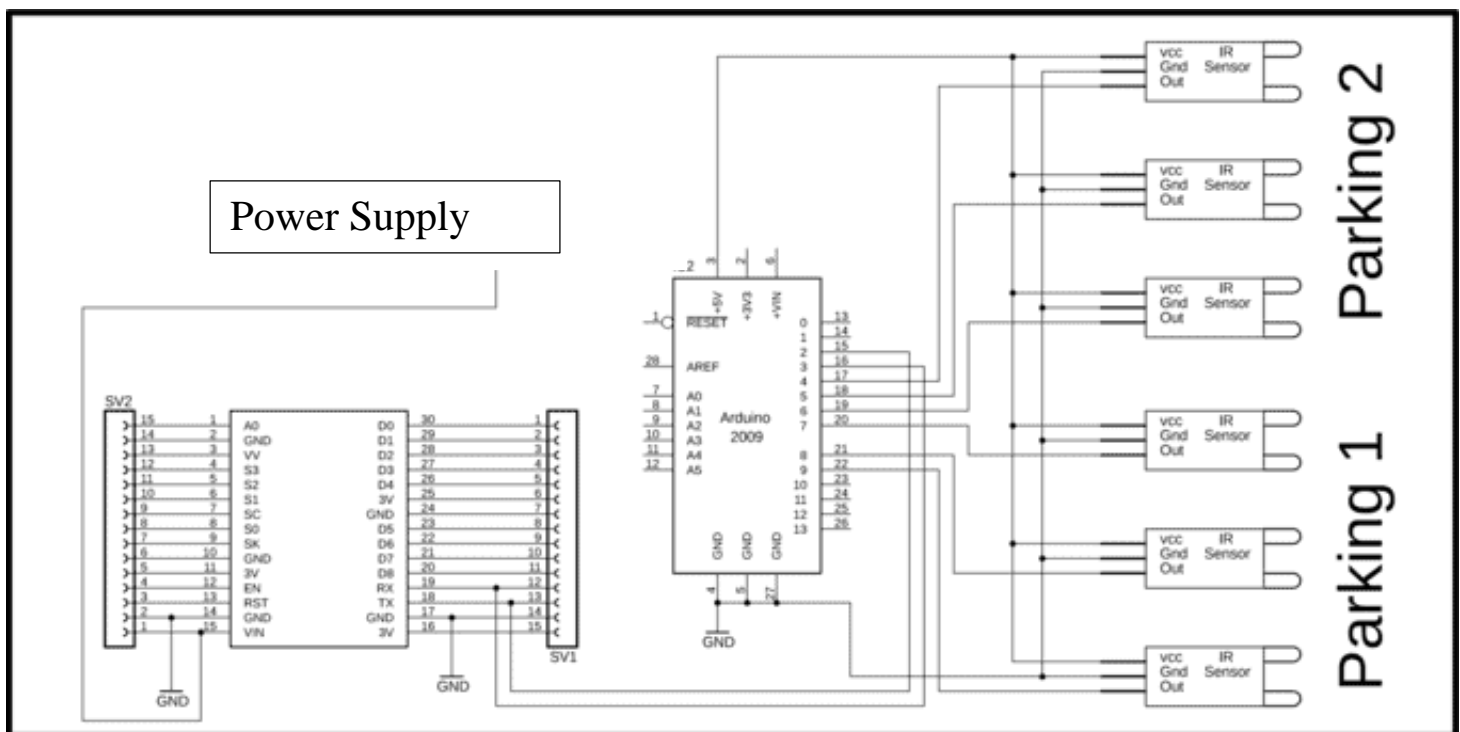
- **Arduino IDE:** It supports the languages C and C++ using distinct guidelines of code architecture, which stores a software library from the wiring project, which runs common input and output procedures.



- **Visual Studio:** Visual Studio is an Integrated Development Environment (IDE) developed by Microsoft to develop GUI(Graphical User Interface), console, Web applications, web apps, mobile apps, cloud, and web services, etc. With the help of this IDE, you can create managed code as well as native code. It uses the various platforms of Microsoft software development software like Windows store, Microsoft Silverlight, and Windows API, etc.



CIRCUIT DIAGRAM



As you can see six infrared sensors are connected with the Arduino pins 4 to 9. The infrared sensor VCC pins are connected with the Arduino's 5v. Grounds are connected with the Arduino's Ground while the out pins of all the infrared sensors are connected with pin 4 to 9.

The Nodemcu module tx and Rx pins are connected with pin2 and pin3 of the Arduino. while the Vin pin of the Nodemcu module is connected with the output of the voltage regulator. This is a regulated 5v power supply based on the lm7805 voltage regulator. We will be using two USB cables, 1 cable will be used for powering up the Arduino and the other USB cable will be used to power up the Nodemcu.

WORKING AND ALGORITHM:

Smart parking suggests an IoT-based system that sends data to free and busy parking places via net/mobile applications. The IoT-network includes sensors and microcontrollers, which are found in each parking place.

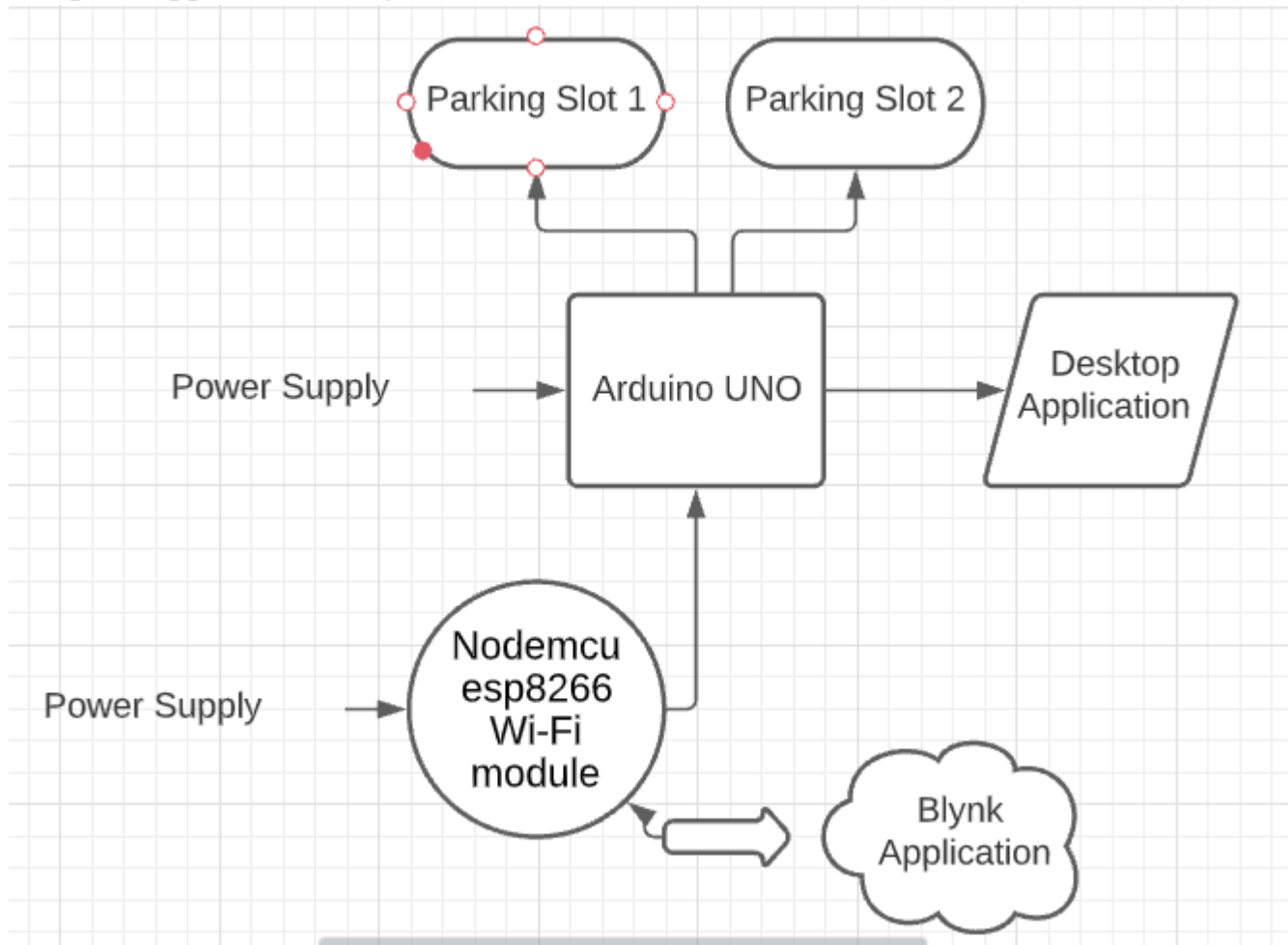
The Parking Area is divided into two Parkings.

Parking 1

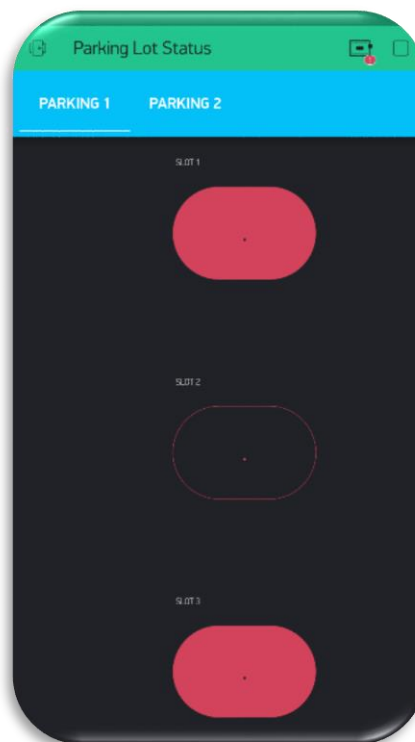
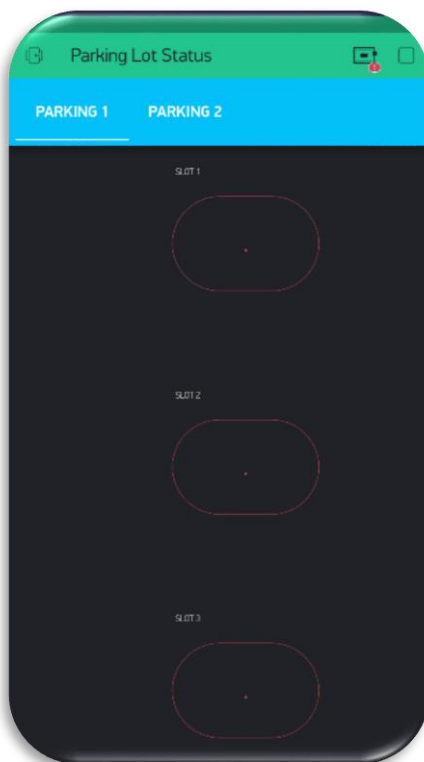
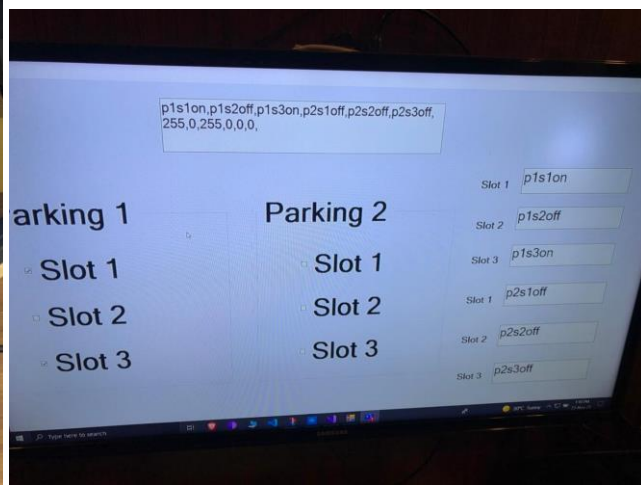
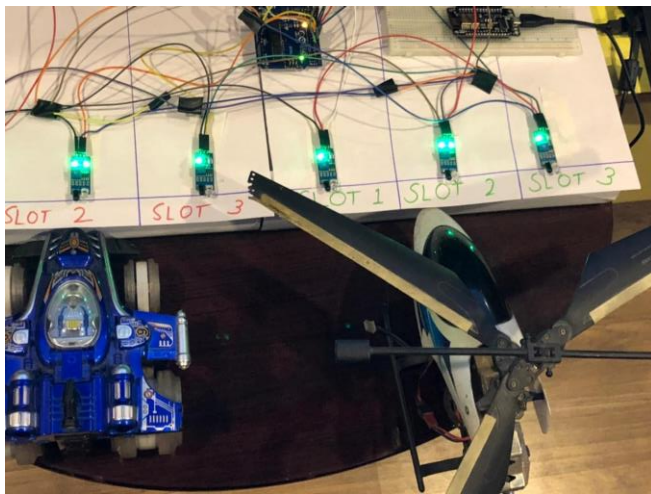
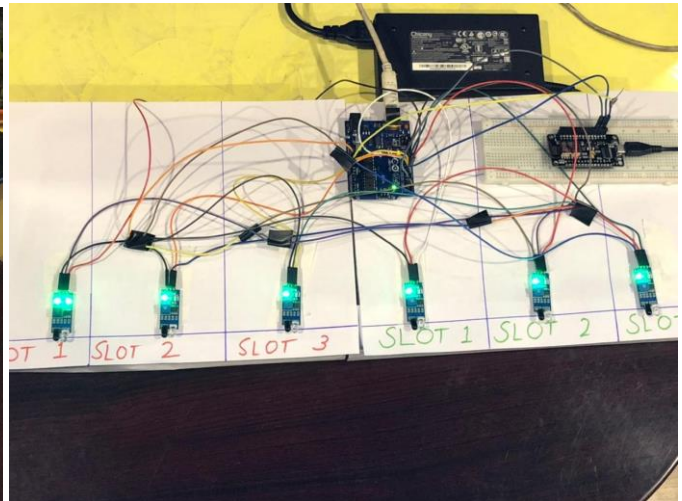
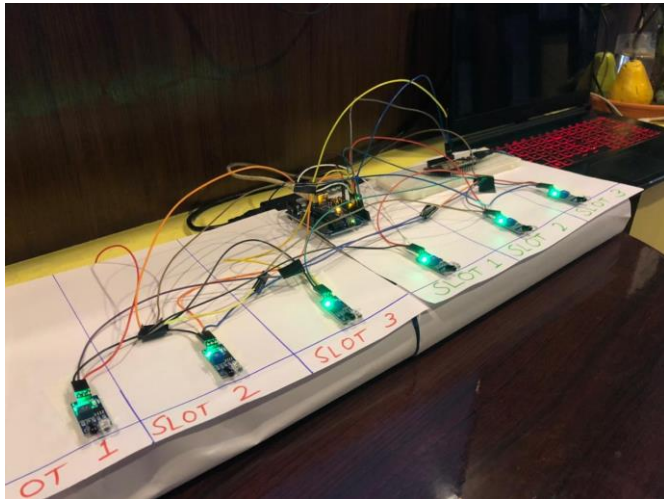
Parking 2

Each Parking has 3 Slots and every slot has one infrared sensor. So we have a total of 6 infrared sensors. Each sensor is used to detect the presence of Car in the Slot. These infrared sensors are connected with the Arduino. So when a car is parked in the slot, the Arduino sends a command to the Nodemcu esp8266 Wi-Fi module, then Nodemcu then sends the command to the Blynk application and show free slots and occupied slots.

As well as Depending on the detection of the car the box next to the slot is checked or unchecked. If the box is checked it means the slot is occupied by a car. It shows the result due to computer application designed in Visual Basic .net which is also known as vb.net.



RESULT



CONCLUSION

The services provided by smart parking have become the essence of building smart cities. This project focused on implementing an integrated solution for smart parking. The proposed system has several advantages, including detecting parking spaces using the Internet of Things and calculating the time of entry and exit and calculating the expected cost. An attractive and effective application was designed for Android mobile phones and desktop application. The project benefits of Smart Parking go well beyond avoiding the needless circling of city blocks. It also enables cities to develop fully integrated multimodal intelligent transportation systems. The system benefits from avoiding wasting time and reducing pollution and fuel consumption.

References:

- [1] [A MULTI-AGENT ORGANIZATION FOR THE GOVERNANCE OF MACHINE-TOMACHINE SYSTEMS, IEEE 2011]
<https://ieeexplore.ieee.org/document/6040668>
- [2][SPARK: A NEW VANET-BASED SMART PARKING SCHEME FOR LARGE PARKING LOTS, IEEE 2009]
<https://ieeexplore.ieee.org/document/5062057>
- [3][A privacy-preserving smart parking system using an IoT elliptic curve based security platform
<https://dl.acm.org/citation.cfm?id=2972908>
- [4][INTEGRATED APPROACH IN THE DESIGN OF CAR PARK OCCUPANCY INFORMATION SYSTEM (COINS), IAENG]
https://www.researchgate.net/publication/26587244_Integrated_Approach_in_the_Design_of_Car_Park_Occupancy_Information_System_COINS
- [5][A MULTIPLE-CRITERIA ALGORITHM FOR SMART PARKING: MAKING FAIR AND PREFERRED PARKING RESERVATIONS IN SMARTCITIES, ACM JOURNAL 2018]
<https://dl.acm.org/citation.cfm?id=32093>
- 18 [6][Smart Parking 26516]
<https://www.happiestminds.com/whitepapers/smart-parking.pdf>

SOURCE CODE:

Arduino Programming Code

```
//ARDUINO PROGRAMMING CODE
#include <SoftwareSerial.h>
SoftwareSerial nodemcu(2,3);

int parking1_slot1_ir_s = 4; // parking slot1 infrared sensor connected with pin number 4 of arduino
int parking1_slot2_ir_s = 5;
int parking1_slot3_ir_s = 6;
int parking2_slot1_ir_s = 7;
int parking2_slot2_ir_s = 8;
int parking2_slot3_ir_s = 9;

String sensor1;
String sensor2;
String sensor3;
String sensor4;
String sensor5;
String sensor6;

String sensorA;
String sensorB;
String sensorC;
String sensorD;
String sensorE;
String sensorF;

void p1slot1();
void p1slot2();
void p1slot3();
void p2slot1();
void p2slot2();
void p2slot3();

String cdata = ""; // complete data, consisting of sensor values for the desktop
String ddata = ""; // complete data, consisting of sensors values for the mobile

void setup()
{
  Serial.begin(9600);
  pinMode(parking1_slot1_ir_s, INPUT);
  pinMode(parking1_slot2_ir_s, INPUT);
  pinMode(parking1_slot3_ir_s, INPUT);
```

```

pinMode(parking2_slot1_ir_s, INPUT);
pinMode(parking2_slot2_ir_s, INPUT);
pinMode(parking2_slot3_ir_s, INPUT);
}
void loop(){

p1slot1();
p1slot2();
p1slot3();

p2slot1();
p2slot2();
p2slot3();

    cdata = cdata + sensor1 + "," + sensor2 + "," + sensor3 + "," + sensor4 + "," + sensor5 + "," + sensor6 + ","; //
comma will be used a delimiter
    ddata = ddata + sensorA + "," + sensorB + "," + sensorC + "," + sensorD + "," + sensorE + "," + sensorF + ","; //
comma will be used a delimiter
    Serial.println(cdata);
    Serial.println(ddata);
    nodemcu.println(ddata);

    delay(2000); // 2 seconds
    cdata = "";
    ddata="";
digitalWrite(parking1_slot1_ir_s, HIGH);
digitalWrite(parking1_slot2_ir_s, HIGH);
digitalWrite(parking1_slot3_ir_s, HIGH);

digitalWrite(parking2_slot1_ir_s, HIGH);
digitalWrite(parking2_slot2_ir_s, HIGH);
digitalWrite(parking2_slot3_ir_s, HIGH);
}

//P1slot1 is a user defined function, it has no return type and it doesn't take any argument as the
input. if there is a car in front of the sensor it gives digital logic 0, and if no car then it gives digital logic 1,
depending on this, then we store p1s1on or p1s1off. The same mechanism is used for all the other infrared
sensors.

void p1slot1() // parking 1 slot1
{
    if( digitalRead(parking1_slot1_ir_s) == LOW)
    {

```

```

    sensor1 = "p1s1on"; // parking1 slot1 (desktop app)
    sensorA = "255"; // parking1 slot1 (mobile app)
    delay(200);
}
if( digitalRead(parking1_slot1_ir_s) == HIGH)
{
    sensor1 = "p1s1off";
    sensorA = "0";
    delay(200);
}
}
void p1slot2() // parkng 1 slot1
{
    if( digitalRead(parking1_slot2_ir_s) == LOW)
    {
        sensor2 = "p1s2on"; // parking1 slot1 (desktop app)
        sensorB = "255"; // parking1 slot1 (mobile app)
        delay(200);
    }
    if( digitalRead(parking1_slot2_ir_s) == HIGH)
    {
        sensor2 = "p1s2off";
        sensorB = "0";
        delay(200);
    }
}
void p1slot3() // parkng 1 slot1
{
    if( digitalRead(parking1_slot3_ir_s) == LOW)
    {
        sensor3 = "p1s3on"; // parking1 slot1 (desktop app)
        sensorC = "255"; // parking1 slot1 (mobile app)
        delay(200);
    }
    if( digitalRead(parking1_slot3_ir_s) == HIGH)
    {
        sensor3 = "p1s3off";
        sensorC = "0";
        delay(200);
    }
}
void p2slot1() // parkng 1 slot1
{
    if( digitalRead(parking2_slot1_ir_s) == LOW)
    {
        sensor4 = "p2s1on"; // parking1 slot1 (desktop app)
        sensorD = "255"; // parking1 slot1 (mobile app)
    }
}

```



```

delay(200);
}
if( digitalRead(parking2_slot1_ir_s) == HIGH)
{
    sensor4 ="p2s1off";
    sensorD = "0";
    delay(200);
}
}
void p2slot2() // parkng 1 slot1
{
    if( digitalRead(parking2_slot2_ir_s) == LOW)
    {
        sensor5 = "p2s2on"; // parking1 slot1 (desktop app)
        sensorE = "255"; // parking1 slot1 (mobile app)
        delay(200);
    }
    if( digitalRead(parking2_slot2_ir_s) == HIGH)
    {
        sensor5 ="p2s2off";
        sensorE = "0";
        delay(200);
    }
}
void p2slot3() // parkng 1 slot1
{
    if( digitalRead(parking2_slot3_ir_s) == LOW)
    {
        sensor6 = "p2s3on"; // parking1 slot1 (desktop app)
        sensorF = "255"; // parking1 slot1 (mobile app)
        delay(200);
    }
    if( digitalRead(parking2_slot3_ir_s) == HIGH)
    {
        sensor6 ="p2s3off";
        sensorF = "0";
        delay(200);
    }
}

```

NODEMCU 8266 programming

```
//NODEMCU 8266 PROGRAMMING (ARDUINO_IDE)
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SoftwareSerial.h>
char auth[] = "h7N5_B-fz7K_51qODgI9pgWBx2zE8IG4";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Basanta_Gajurel";
char pass[] = "password123";
BlynkTimer timer;
String myString; // complete message from arduino, which consists of sensors data
char rdata; // received characters
int firstVal, secondVal, thirdVal; // sensors
int led1, led2, led3, led4, led5, led6;
// This function sends Arduino's up time every second to Virtual Pin (1).
// In the app, Widget's reading frequency should be set to PUSH. This means
// that you define how often to send data to Blynk App.
void myTimerEvent()
{
    // You can send any value at any time.
    Blynk.virtualWrite(V1, millis() / 1000);
}
void setup()
{
    BlynkTimer timer;
    // Debug console
    Serial.begin(9600);
    Blynk.begin(auth, ssid, pass);
    timer.setInterval(1000L, sensorvalue1);
    timer.setInterval(1000L, sensorvalue2);
    timer.setInterval(1000L, sensorvalue3);
    timer.setInterval(1000L, sensorvalue4);
    timer.setInterval(1000L, sensorvalue5);
    timer.setInterval(1000L, sensorvalue6);
}
void loop()
{
    if (Serial.available() == 0 )
    {
        Blynk.run();
        timer.run(); // Initiates BlynkTimer
    }
}
```

```

    }
    if (Serial.available() > 0 )
    {
        rdata = Serial.read();
        myString = myString+ rdata;
        Serial.print(rdata);
        if( rdata == '\n')
        {
            Serial.println(myString);
            Serial.println("hello"); //this shows in serial monitor means the code is working
// new code
String l = getValue(myString, ',', 0);
String m = getValue(myString, ',', 1);
String n = getValue(myString, ',', 2);
String o = getValue(myString, ',', 3);
String p = getValue(myString, ',', 4);
String q = getValue(myString, ',', 5);
// these leds represents the leds used in Blynk application
led1 = l.toInt();
led2 = m.toInt();
led3 = n.toInt();
led4 = o.toInt();
led5 = p.toInt();
led6 = q.toInt();
    myString = "";
// end new code
    }}}
void sensorvalue1()
{
    int sdata = led1;
    // You can send any value at any time.
    Blynk.virtualWrite(V10, sdata);
}
void sensorvalue2()
{
    int sdata = led2;
    // You can send any value at any time.
    Blynk.virtualWrite(V11, sdata);
}
void sensorvalue3()
{
    int sdata = led3;
    // You can send any value at any time.
    Blynk.virtualWrite(V12, sdata);}

```

```

void sensorvalue4()
{
int sdata = led4;
  // You can send any value at any time.
  Blynk.virtualWrite(V13, sdata);
}
void sensorvalue5()
{
int sdata = led5;
  // You can send any value at any time.
  Blynk.virtualWrite(V14, sdata);
}
void sensorvalue6()
{
int sdata = led6;
  // You can send any value at any time.
  Blynk.virtualWrite(V15, sdata);
}
String getValue(String data, char separator, int index)
{
  int found = 0;
  int strIndex[] = { 0, -1 };
  int maxIndex = data.length() - 1;
  for (int i = 0; i <= maxIndex && found <= index; i++) {
    if (data.charAt(i) == separator || i == maxIndex) {
      found++;
      strIndex[0] = strIndex[1] + 1;
      strIndex[1] = (i == maxIndex) ? i+1 : i;
    }
  }
  return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}

```

Desktop Application Framework code

```
Imports System.IO
Imports System.IO.Ports
Public Class Parking_Lot_Status
    Dim value1 As Integer
    Private Sub Parking_Lot_Status_Load(sender As Object, ByVal e As EventArgs) Handles MyBase.Load
        SerialPort1.Close()
        SerialPort1.PortName = "COM3"
        SerialPort1.BaudRate = "9600"
        SerialPort1.DataBits = 8
        SerialPort1.Parity = Parity.None
        SerialPort1.StopBits = StopBits.One
        SerialPort1.Handshake = Handshake.None
        SerialPort1.Encoding = System.Text.Encoding.Default
        SerialPort1.Open()
    End Sub
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Timer1.Tick
        Dim s As String
        s = TextBox1.Text + "," + "," + "," + "," + "," + "," + "," + ""
        Dim somestring() As String
        ' Split string based on comma
        somestring = s.Split(New Char() {","c})

        TextBox2.Text = somestring(0)
        TextBox3.Text = somestring(1)
        TextBox_4.Text = somestring(2)
        TextBox5.Text = somestring(3)
        TextBox6.Text = somestring(4)
        TextBox7.Text = somestring(5)
        TextBox1.Text = ""
    End Sub
    Private Sub DataReceived(ByVal sender As Object, ByVal e As SerialDataReceivedEventArgs) Handles
SerialPort1.DataReceived
        Try
            Dim mydata As String = ""
            mydata = SerialPort1.ReadExisting()
            If TextBox1.InvokeRequired Then
                TextBox1.Invoke(DirectCast(Sub() TextBox1.Text &= mydata, MethodInvoker))
            Else
                TextBox1.Text &= mydata
            End If
        Catch ex As Exception
```

```

        MessageBox.Show(ex.Message)
    End Try
End Sub

Private Sub TextBox2_TextChanged(sender As System.Object, e As System.EventArgs) Handles
TextBox2.TextChanged
    If InStr(TextBox2.Text, "p1s1on") Then
        chkp1slot1.Checked = True
    End If

    If InStr(TextBox2.Text, "p1s1off") Then
        chkp1slot1.Checked = False
    End If
End Sub

Private Sub TextBox3_TextChanged(sender As System.Object, e As System.EventArgs) Handles
TextBox3.TextChanged
    If InStr(TextBox3.Text, "p1s2on") Then
        chkp1slot2.Checked = True
    End If

    If InStr(TextBox3.Text, "p1s2off") Then
        chkp1slot2.Checked = False
    End If
End Sub

Private Sub TextBox_4_TextChanged(sender As System.Object, e As System.EventArgs) Handles
TextBox_4.TextChanged
    If InStr(TextBox_4.Text, "p1s3on") Then
        chkp1slot3.Checked = True
    End If

    If InStr(TextBox_4.Text, "p1s3off") Then
        chkp1slot3.Checked = False
    End If
End Sub

Private Sub TextBox5_TextChanged(sender As System.Object, e As System.EventArgs) Handles
TextBox5.TextChanged
    If InStr(TextBox5.Text, "p2s1on") Then
        chkp2slot1.Checked = True
    End If

    If InStr(TextBox5.Text, "p2s1off") Then
        chkp2slot1.Checked = False
    End If
End Sub

Private Sub TextBox6_TextChanged(sender As System.Object, e As System.EventArgs) Handles
TextBox6.TextChanged
    If InStr(TextBox6.Text, "p2s2on") Then

```

```
        chkp2slot2.Checked = True
    End If
    If InStr(TextBox6.Text, "p2s2off") Then
        chkp2slot2.Checked = False
    End If
End Sub

Private Sub TextBox7_TextChanged(sender As System.Object, e As System.EventArgs) Handles
    TextBox7.TextChanged
    If InStr(TextBox7.Text, "p2s3on") Then
        chkp2slot3.Checked = True
    End If
    If InStr(TextBox7.Text, "p2s3off") Then
        chkp2slot3.Checked = False
    End If
End Sub
End Class
```