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 System

(A Microcontroller based Parking Monitoring System.)

<u>Submitted By:</u>

Team Member

Prithak Gajurel (20BCE2921)
Pratik Luitel (20BCE2897)
Bijan Shrestha (20BCE2904)
Anurag Karki (20BCE2907)
Sandesh Khatiwada(20BCE2898)

<u>SUBMITTED TO:</u>

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. Saranya K.C**, for her patient guidance, enthusiastic encouragement and useful critiques for this project work.

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

TABLE OF CONTENTS

OBJECTIVE

The main objective of the project is to make the clients aware about the status of the parking lot at real time. This will help the clients to recognize the nearest available parking zone making it time and cost efficient parking system.

INTRODUCTION

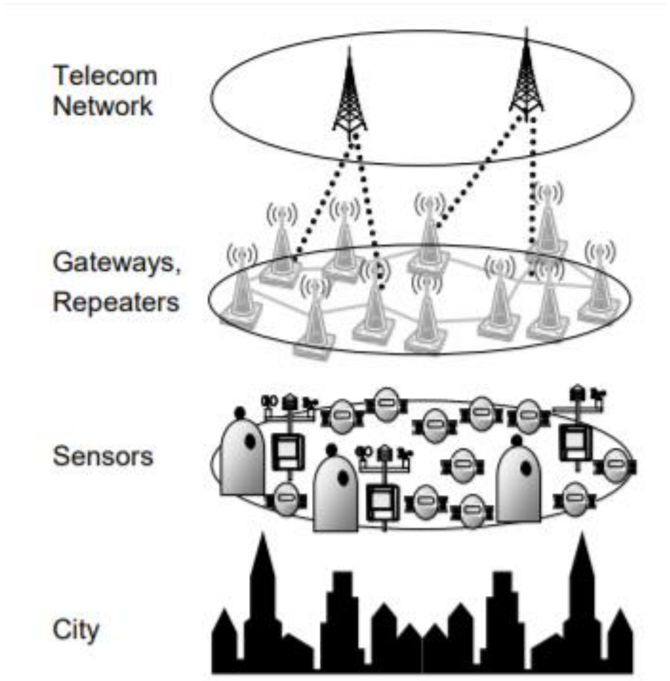
With growing, Vehicle parking increases with the number of users. With the increased use of smartphones and their applications, users prefer mobile phone-based solutions. One of the most important problems facing large cities is congestion and parking. So, using Automated Parking System Management is an efficient technique using the Internet of Things to manage the garage.This paper proposes the Smart Parking system (vehicle monitoring), that depends on Arduino parts, Android/iphone applications, desktop application (through serial connections), nodemcu esp8266 wifi module and sensors. This gave the client the ability to check available parking spaces at real time. 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.. With IoT technology, the smart parking system can be connected wirelessly to easily track available locations. Furthermore, with the help of serial communication of the management controlling desktop and the main microcontroller (Arduino in this case), we can also know the status of the parking lots in real time. This will mainly help the parking lot manager take and grant reservations of parking lots from the client.

LITERATURE SURVEY

[1] The Senscity Project:

Camille Persson et-al proposed the blueprint a smart parking management system namely 'The Senscity Project'. 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 Senscity project proposes an infrastructure to enable shared city scale applications. Multi-Agent technologies grant adaptability, flexibility and productivity properties.

It was a project of having network and iot based devices in order to make the entire city smart. However, the module parking systems were only implemented public places (as it was funded by the French government). Another limitation of this system was it was only for sensing and documenting the presence of four wheeler vehicles. Our prototype has adopted the primitive network configuration and iot implementation from this huge nation-wide built project.



[2] Wireless cloud implemented real time parking system using raspberry pi

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.

This parking system is designed for private companies who own a parking lot. We have used the ESP8266 chip wifi chip from this project. Some of the limitation of this design can be that it is only implemented on one platform i.e mobile phones (unlike ours which is available on more than one) and the wireless and cloud nature of this project can result in latency in its processing. To overcome that, a better processor/ microcontroller can be used instead of a raspberry pi.

[3] Parking Monitoring System for the parking lot manager

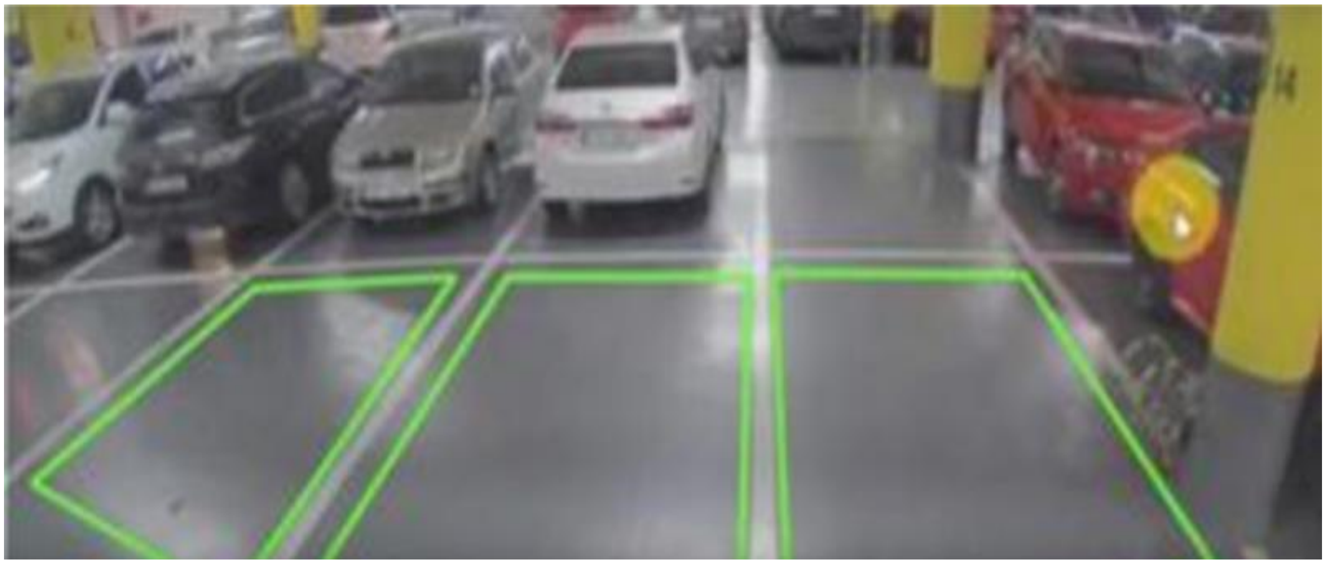
Vanessa W.s Tang et-al proposed a WSN-based intelligent car parking system, low-cost wireless sensors are deployed into a car park field, with each parking lot equipped with one sensor node, which detects and monitors the occupation of the parking lot. The status of the parking field detected by sensor 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.

This design is very good and efficient for the parking lot manager as it makes him aware about the status of the parking lots. It also records the entry and the exit time of the clients and maintains a sophisticated database. However, it lacks the scalability of its facilities. The sole user of this system is the manager himself. Further development of desktop apps and mobile applications for the user so that they can know and book a parking lot can make this design highly desirable

[4] Parking System designed by the utility of Image Processing

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.

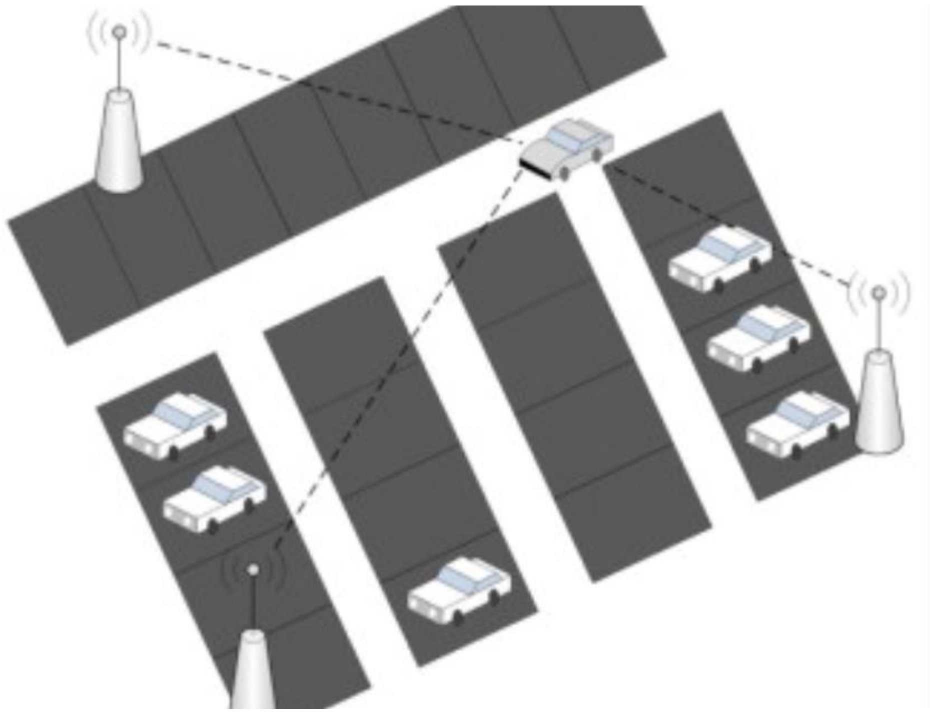
This design is an advanced design. Some of its merits include, it avoids the failure of sensors, the presence of car is made more certain and clear than that of sensor parking systems and the level of security is increased due to rapid picture taking from the detection camaras. Some of its demerits are: it is a very costly design, it is only applicable for four wheeler vehicles and it lacks the user scalability as the only user of this system is the parking lot manager.



[5] Parking System using Vehicular Communication:

In this technique of parking system, few vehicles are used as the communication node for the purpose of data transmission as shown in the figure. The node vehicles can be taxi or some other public vehicles that tracks the same route every day. These vehicles communicate with the parking lot status towers that send the data of the parking lot status, This way, the node vehicles will know the status of the parking lots at real time.

This design is very effective for the node vehicles as they seamlessly receive the required data at real time. However, its limitations is, the facility isn't available for any other normal users who do not do the same route everyday.



[6] A privacy-preserving smart parking system using an IoT elliptic curve based security platform

Today, there is a large variety of hardware and software to choose from that is easy to set up and use. Even though there is an increasing number of real-world applications that employ large deployments of IoT devices, the wireless nature of communication in combination with the low-end capabilities of the devices raises security and privacy issues that have not been properly addressed. In this paper Ioannis Chatzigiannakis and his team has adopted Elliptic Curve Cryptography (ECC) as an attractive alternative to conventional public key cryptography, such as RSA. ECC is an ideal candidate for implementation on constrained devices where the major computational resources, i.e., speed, memory are limited and low-power wireless communication protocols are employed. That is because it attains the same security levels with traditional cryptosystems using smaller parameter sizes. They provide a generic implementation of ECC that runs on different host operating systems, such as Contiki, TinyOS, iSenseOS, ScatterWeb and Arduino. Furthermore, it runs on smartphone platforms such as Android and iPhone and also any linux based systems (e.g., raspberryPi).

[7] Smart Parking Applications Using RFID Technology

There has been a considerable amount of reduction in transaction costs and decrease in stock shortage with the use of Radio Frequency Identification (RFID) technology in automation. In this study, a solution has been provided for the problems encountered in parking-lot management systems via RFID technology. RFID readers, RFID labels, computers, barriers and software are used as for the main components of the RFID technology. The software has been handled for the management, controlling, transaction reporting and operation tasks for parking lots located on various parts of the city. Check-ins and check-outs of the parking-lots will be under control with RFID readers, labels and barriers. Personnel costs will be reduced considerably using this technology. Instead of cars' parking on streets, a more modern and a fast operating parking-lot system have been developed.

[8] Vision based SPS (Smart Parking System)

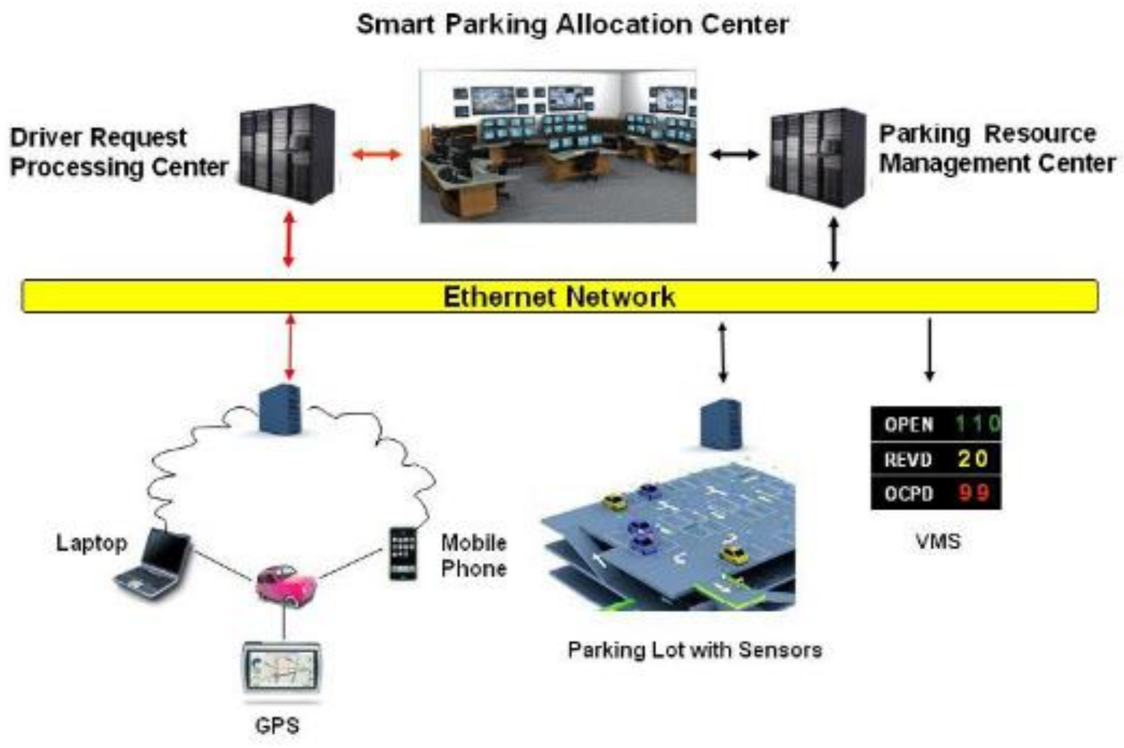
Monitoring parking lot vacancy is a significant technology which can be used for guiding cars to vacant spaces and for efficient use of parking spaces. Monitoring detection technology can be divided into two categories. The first estimates the number of remaining vacant spaces for the entire parking lot by counting incoming and outgoing vehicles. The second monitors the status of each individual space and can be used to guide a car to a vacant space. To help drivers find a vacant parking space without much effort, intelligent parking systems should provide the specific location of vacant spaces and not just the total number of spaces.

The limitation of this project is its scalability as it is only for the parking lot manager and it does not consider the improper parking condition of some cars. It is also only applicable for four wheelers.

Figure 1. Improper Parking

[9] Parking System that solves the MILP (Mixed Integer Linear Program)

The system assigns and reserves an optimal parking space for a driver based on the user's requirements that combine proximity to destination and parking cost, while also ensuring that the overall parking capacity is efficiently utilized. This approach solves a Mixed Integer Linear Program (MILP) problem at each decision point in a time-driven sequence. The solution of each MILP is an optimal allocation based on current state information and subject to random events such as new user requests or parking spaces becoming available. The allocation is updated at the next decision point ensuring that there is no resource reservation conflict and that no user is ever assigned a resource with higher than the current cost function value. Implementation issues including parking detection, reservation guarantee and Vehicle-to-Infrastructure (V2I) or Infrastructure-to-Vehicle (I2 V) communication are resolved in the paper.


Smart Parking Allocation Center
Driver Request Processing Center
Parking Resource Management Center
Ethernet Network
Laptop
Mobile Phone
GPS
Parking Lot with Sensors
OPEN 110
REVD 20
OCPD 99
VMS

[10] Automatic Parking Management System and Parking Fee Collection Based on Number Plate Recognition

This paper discussed on automatic parking system and electronic parking fee collection based on vehicle number plate recognition.

The aim of this research is to develop and implement an automatic parking system that will increase convenience and security of the public parking lot as well as collecting parking fee without hassles of using magnetic card. The auto parking system will able to have less interaction of humans and use no magnetic card and its devices. In additions to that, it has parking guidance system that can show and guide user towards a parking space. The system used image processing of recognizing number plates for operation of parking and billing system. Overall, the systems run with pre-programmed controller to make minimum human involvement in parking system and ensure access control in restricted places. This paper presents algorithm technology based method for license plate extraction from car images followed by the segmentation of characters and reorganization and also develop electronics parking fee collection system based on number plate information. It is a terrific paper. However, the only drawback is that it is very expensive.

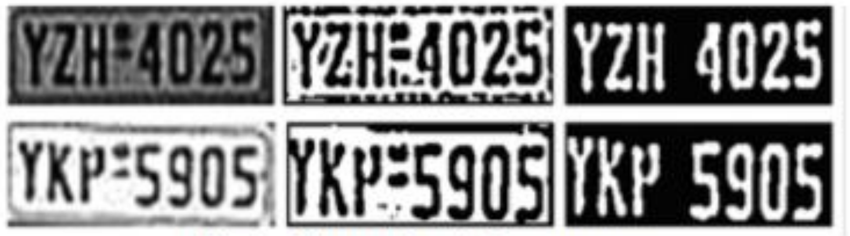


Fig. 1. License plate sizing sequence

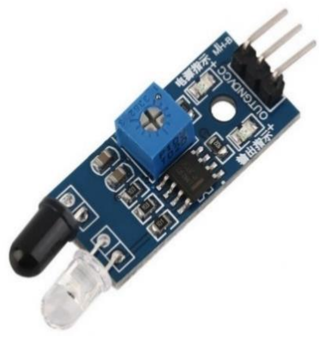
COMPONENTS OF PROPOSED SYSTEM

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

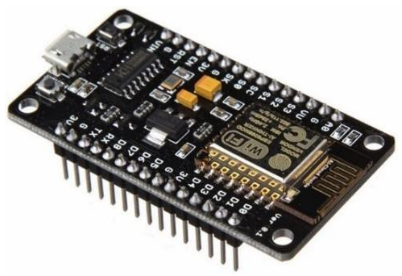
Hardware Components

- **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.

- **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.



- **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.



- **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.

- **Proteus 8:** The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

- **Blynk Application:** Blynk app for iOS and Android is the easiest way to build your own mobile app that work with the hardware of your choice. ... It handles all the connection routines and data exchange between your hardware, Blynk Cloud, and your app project.

Reasons for the utility of softwares, programming language and framework:

Softwares we used and why:

Arduino IDE: Simply in order to have serial communication between desktop and for the interfacing of Nodemcu esp8266 wifi module.
Visual studio: For the purpose of coding in vb.net, making form based desktop app (prototype) and for the displaying of the results of interfacing between the desktop and Arduino microcontroller.
Protheus: To make the sketch of how hardware is being assembled
Blynk mobile application: To create a user interface in order to know the status of the parking lot using sensor datas.

Programming languages and frameworks we used why:

C based language: it is used in Arduino IDE as it is the protocol to control the arduino and nodemcu microcontrollers
Visual Basic with dot net framework: Visual basic is one of the pioneers of programming languages hence it is great for the direct communication with the simple hardwares such as arduino. It supports .NET framework which is great and easy to develop tabs and widgets manually.

Methodology:

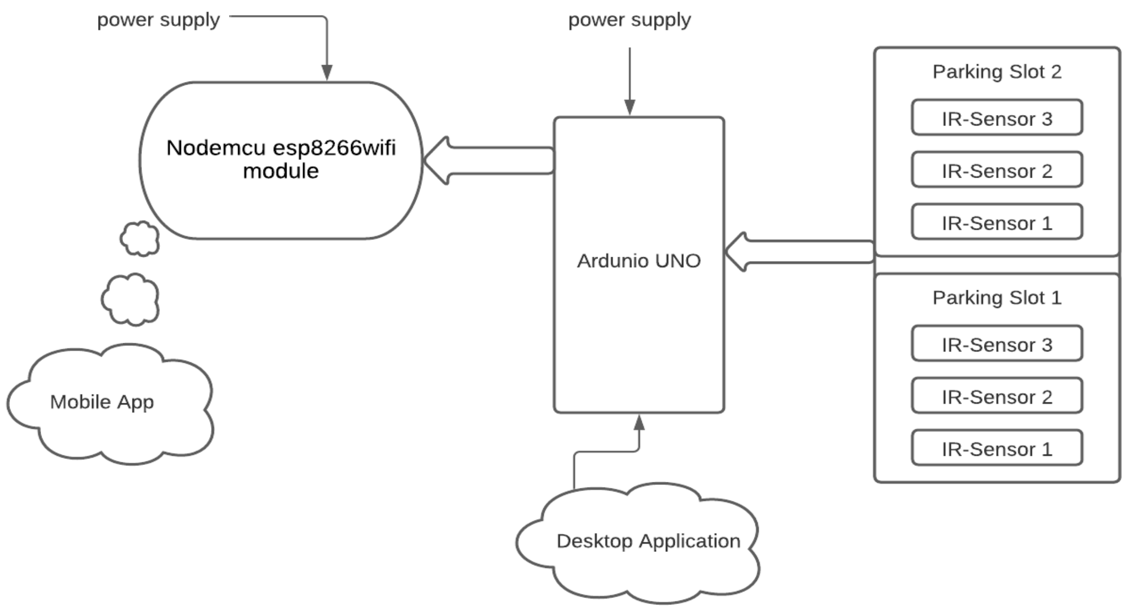
Blocks of the Project:

Smart Parking system (vehicle monitoring)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**.**



Software implementation:

As mentioned already, all the softwares along with their drivers and library were instralled in a device. The arduino codes were compiled and fed to the arduino using arduino ide and data cable. The codes of Nodmcu esp8266 wifi module were compled in arduino ide and fed to the chip using data cable. Now, dot net (.NET) framework of visual basic programming language (vb.net) was installed in visual studio. The form mode of template was designed using the ‘design’ option of vb.net. Most importantly a timer and serial port was set up for the serial and periodic communication between the desktop and the microcontroller. For the implementation of the project in mobile devices, blynk application was installed. Appropriate widgets and tabs were set for interfacing. Blynk was connected with nodemcu wifi model after giving the internet and authorization token in the nodemcu program.

Hardware Architecture and its Implementation:

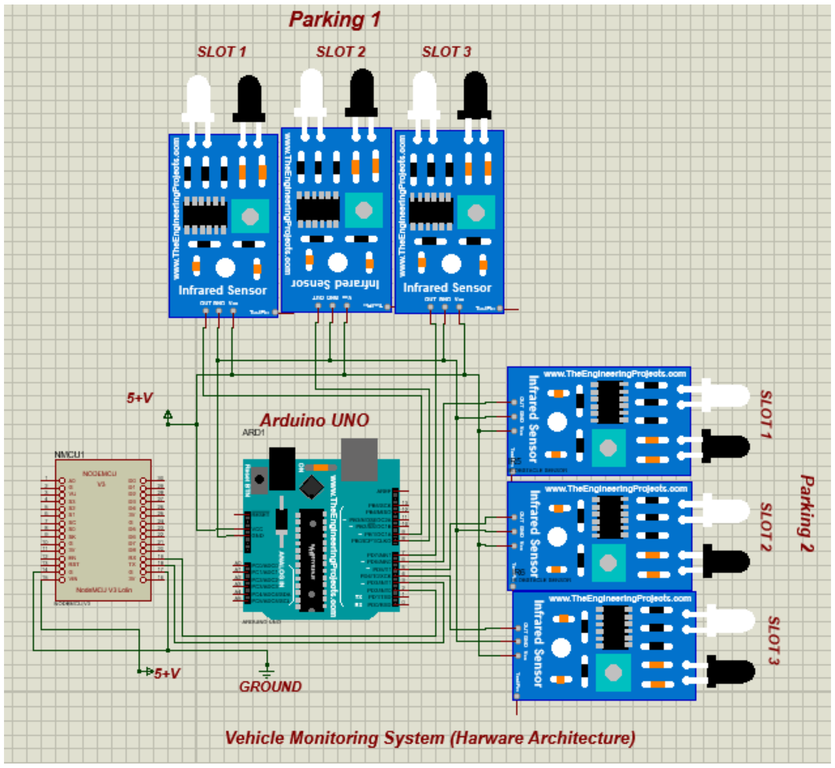
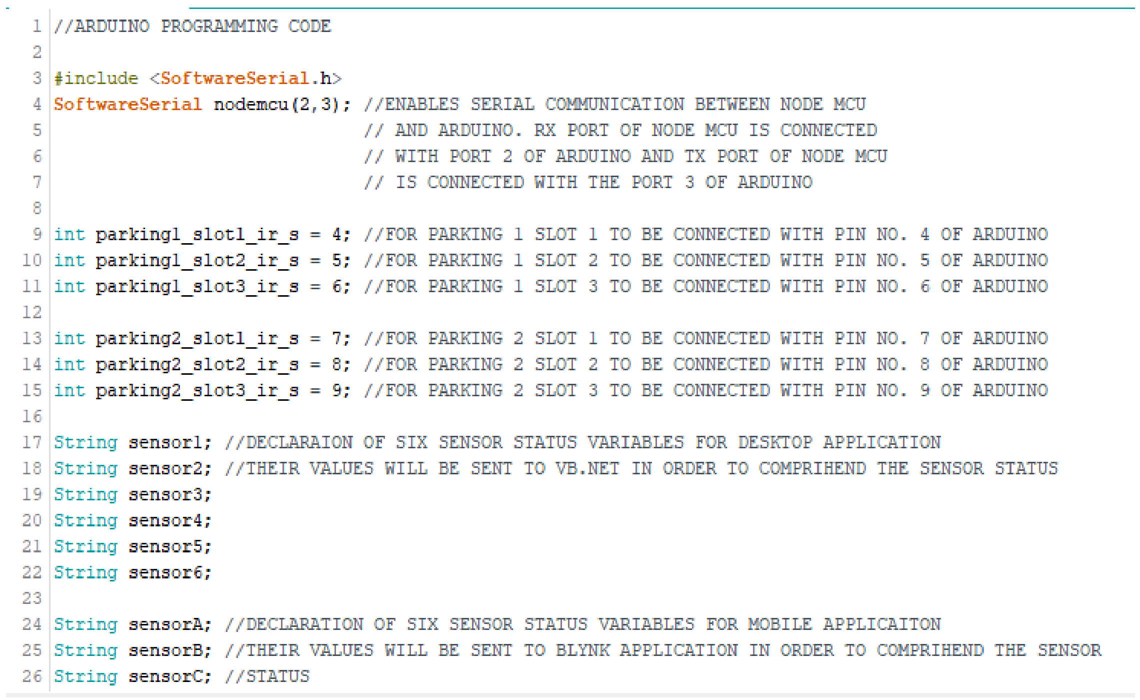


Fig: Hardware Architecture designed in Protheus

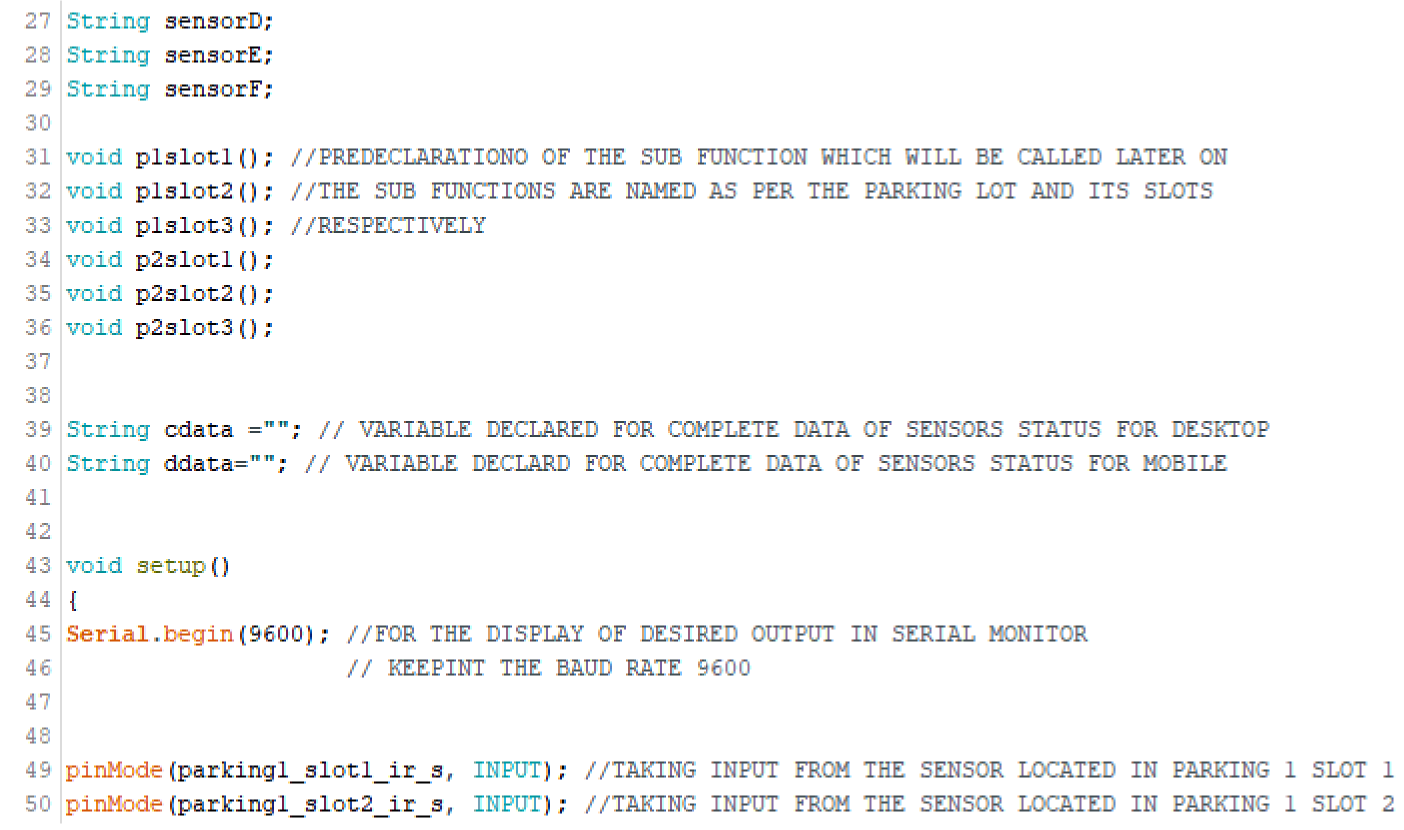
As already mentioned above, first all the hardware components were assembled. The 6 infrared sensors are considered as to be 6 parking lot slots (one sensor for each).. The 6 out pins (in total) were connect to the arduino pins 4,5,6,7,8 and 9 respectively. All the gcc (ground) pins of the sensors were merged together and connected to the gcc pin of the arduino. Similarly all the vcc pins of the sensors (6 in total) were merged using connecting wires and connected to the single Vcc pin of the arduino. Pins 2 and 3 were connected to the Rx and Tx respectively of the nodmcu wifi module.

CODES AND ITS EXPLANATIONI IN DETAIL:

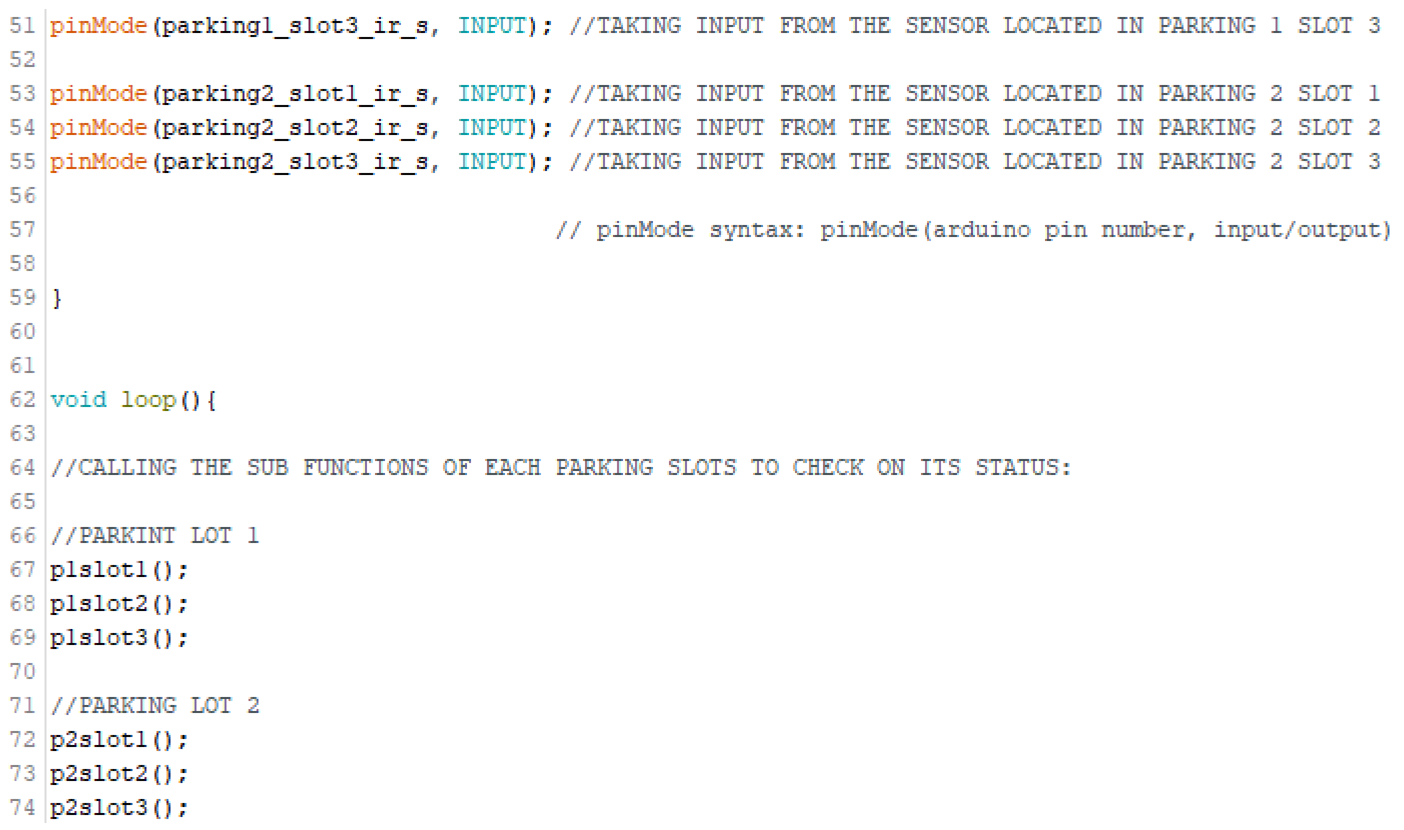
Arduino programming code:



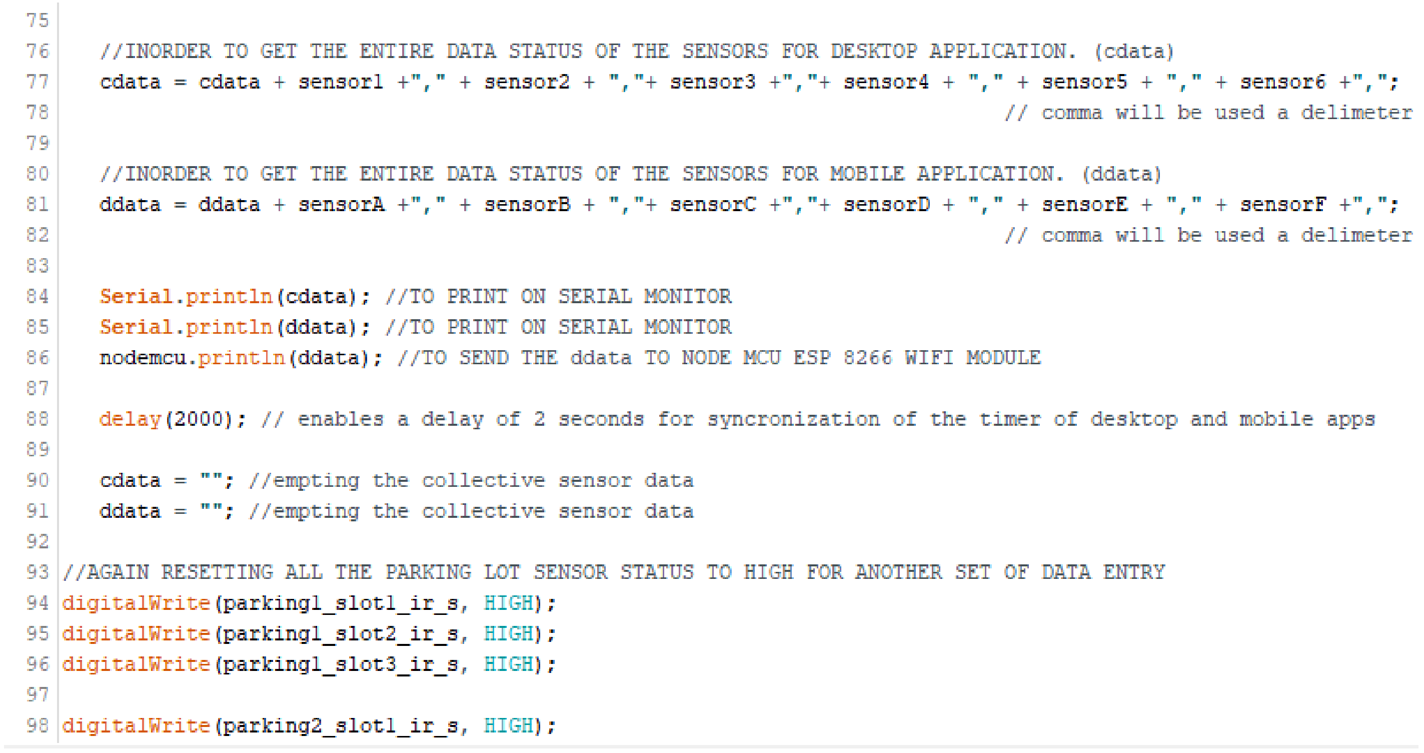
```
//ARDUINO PROGRAMMING CODE

#include <SoftwareSerial.h>
SoftwareSerial nodemcu(2,3); //ENABLES SERIAL COMMUNICATION BETWEEN NODE MCU
                             // AND ARDUINO. RX PORT OF NODE MCU IS CONNECTED
                             // WITH PORT 2 OF ARDUINO AND TX PORT OF NODE MCU
                             // IS CONNECTED WITH THE PORT 3 OF ARDUINO

int parking1_slot1_ir_s = 4; //FOR PARKING 1 SLOT 1 TO BE CONNECTED WITH PIN NO. 4 OF ARDUINO
int parking1_slot2_ir_s = 5; //FOR PARKING 1 SLOT 2 TO BE CONNECTED WITH PIN NO. 5 OF ARDUINO
int parking1_slot3_ir_s = 6; //FOR PARKING 1 SLOT 3 TO BE CONNECTED WITH PIN NO. 6 OF ARDUINO

int parking2_slot1_ir_s = 7; //FOR PARKING 2 SLOT 1 TO BE CONNECTED WITH PIN NO. 7 OF ARDUINO
int parking2_slot2_ir_s = 8; //FOR PARKING 2 SLOT 2 TO BE CONNECTED WITH PIN NO. 8 OF ARDUINO
int parking2_slot3_ir_s = 9; //FOR PARKING 2 SLOT 3 TO BE CONNECTED WITH PIN NO. 9 OF ARDUINO

String sensor1; //DECLARAION OF SIX SENSOR STATUS VARIABLES FOR DESKTOP APPLICATION
String sensor2; //THEIR VALUES WILL BE SENT TO VB.NET IN ORDER TO COMPRIHEND THE SENSOR STATUS
String sensor3;
String sensor4;
String sensor5;
String sensor6;

String sensorA; //DECLARATION OF SIX SENSOR STATUS VARIABLES FOR MOBILE APPLICAITON
String sensorB; //THEIR VALUES WILL BE SENT TO BLYNK APPLICATION IN ORDER TO COMPRIHEND THE SENSOR
String sensorC; //STATUS
```



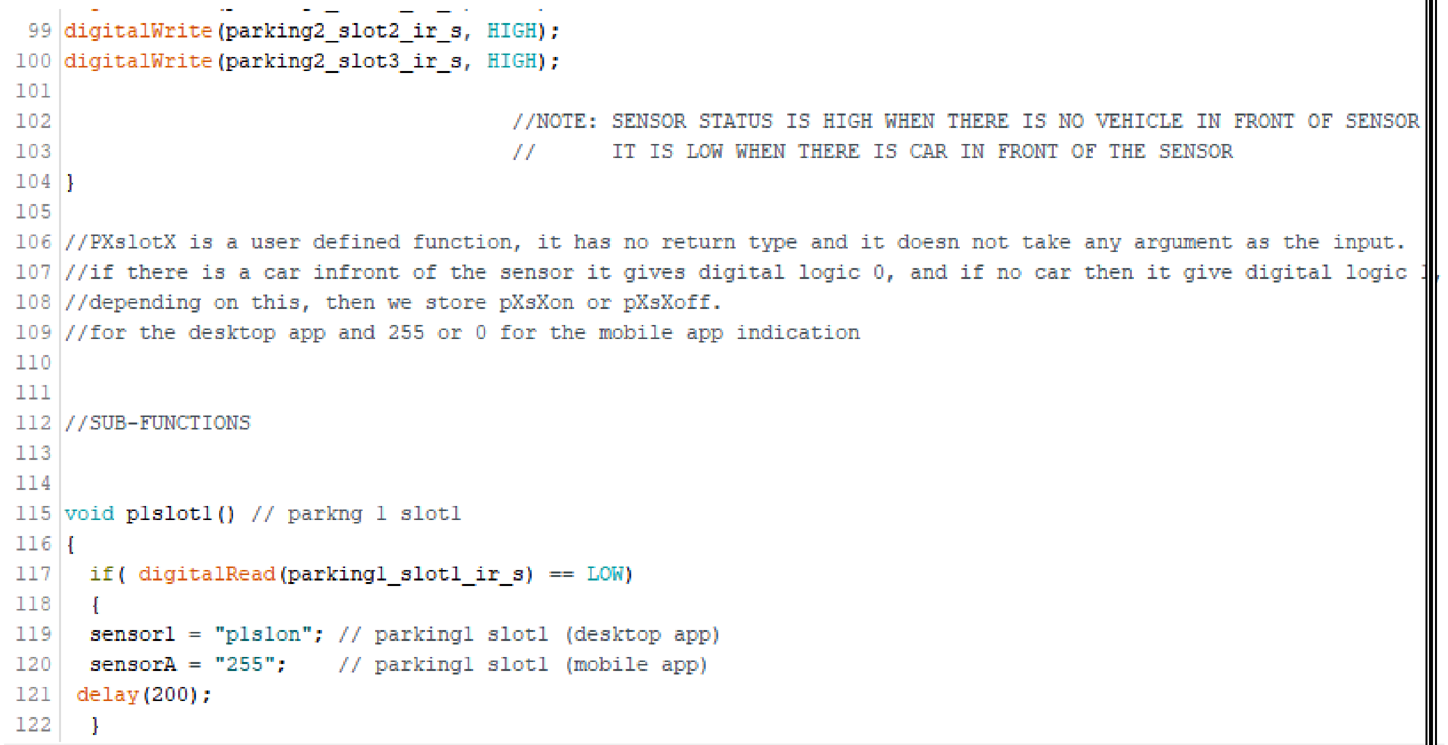
```
String sensorD;
String sensorE;
String sensorF;

void p1slot1(); //PREDECLARATIONO OF THE SUB FUNCTION WHICH WILL BE CALLED LATER ON
void p1slot2(); //THE SUB FUNCTIONS ARE NAMED AS PER THE PARKING LOT AND ITS SLOTS
void p1slot3(); //RESPECTIVELY
void p2slot1();
void p2slot2();
void p2slot3();


String cdata =""; // VARIABLE DECLARED FOR COMPLETE DATA OF SENSORS STATUS FOR DESKTOP
String ddata=""; // VARIABLE DECLARD FOR COMPLETE DATA OF SENSORS STATUS FOR MOBILE


void setup()
{
Serial.begin(9600); //FOR THE DISPLAY OF DESIRED OUTPUT IN SERIAL MONITOR
                    // KEEPINT THE BAUD RATE 9600


pinMode(parking1_slot1_ir_s, INPUT); //TAKING INPUT FROM THE SENSOR LOCATED IN PARKING 1 SLOT 1
pinMode(parking1_slot2_ir_s, INPUT); //TAKING INPUT FROM THE SENSOR LOCATED IN PARKING 1 SLOT 2
```



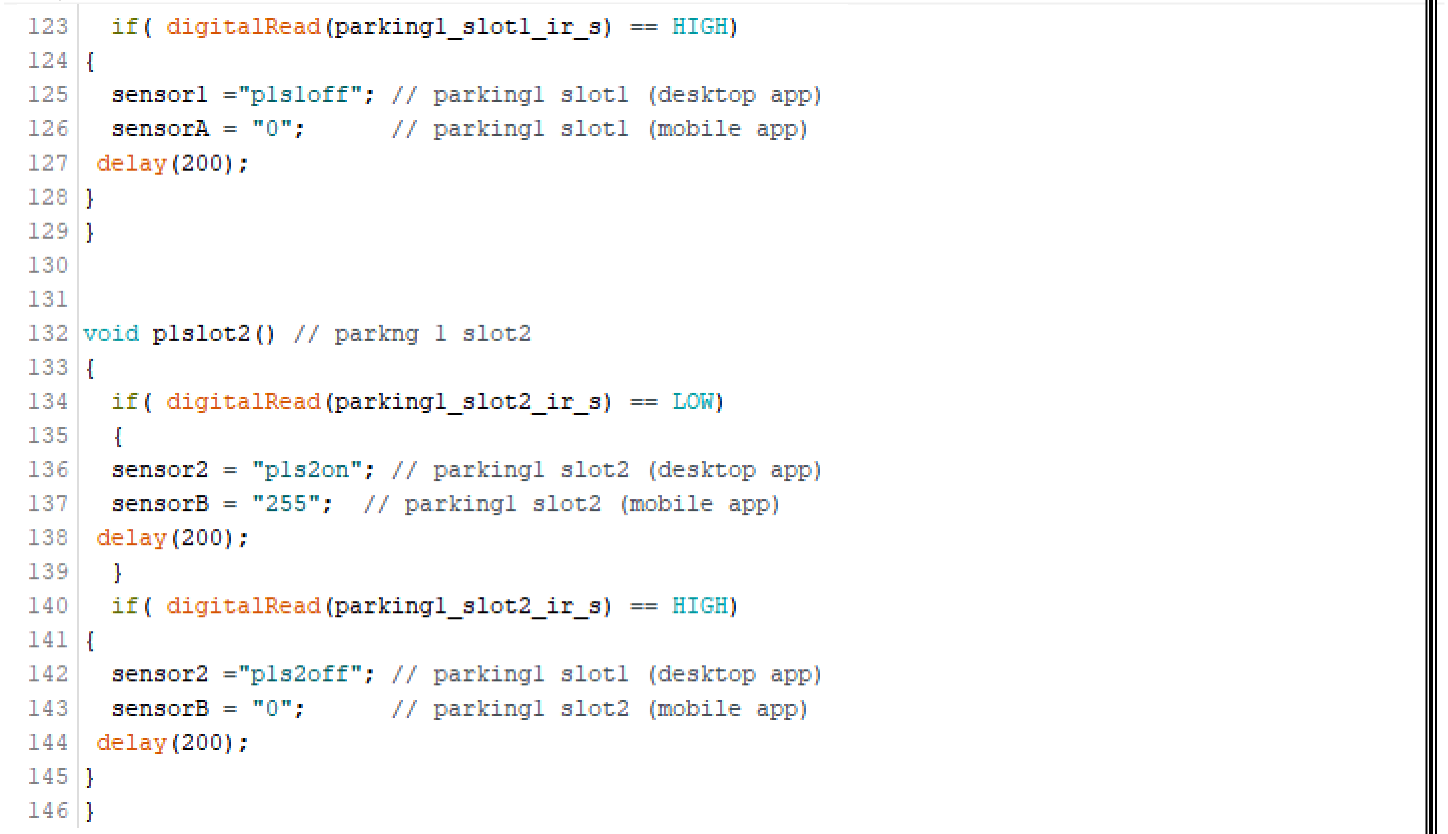
```
pinMode(parking1_slot3_ir_s, INPUT); //TAKING INPUT FROM THE SENSOR LOCATED IN PARKING 1 SLOT 3

pinMode(parking2_slot1_ir_s, INPUT); //TAKING INPUT FROM THE SENSOR LOCATED IN PARKING 2 SLOT 1
pinMode(parking2_slot2_ir_s, INPUT); //TAKING INPUT FROM THE SENSOR LOCATED IN PARKING 2 SLOT 2
pinMode(parking2_slot3_ir_s, INPUT); //TAKING INPUT FROM THE SENSOR LOCATED IN PARKING 2 SLOT 3

                                    // pinMode syntax: pinMode(arduino pin number, input/output)

}


void loop(){

//CALLING THE SUB FUNCTIONS OF EACH PARKING SLOTS TO CHECK ON ITS STATUS:

//PARKINT LOT 1
p1slot1();
p1slot2();
p1slot3();

//PARKING LOT 2
p2slot1();
p2slot2();
p2slot3();
```



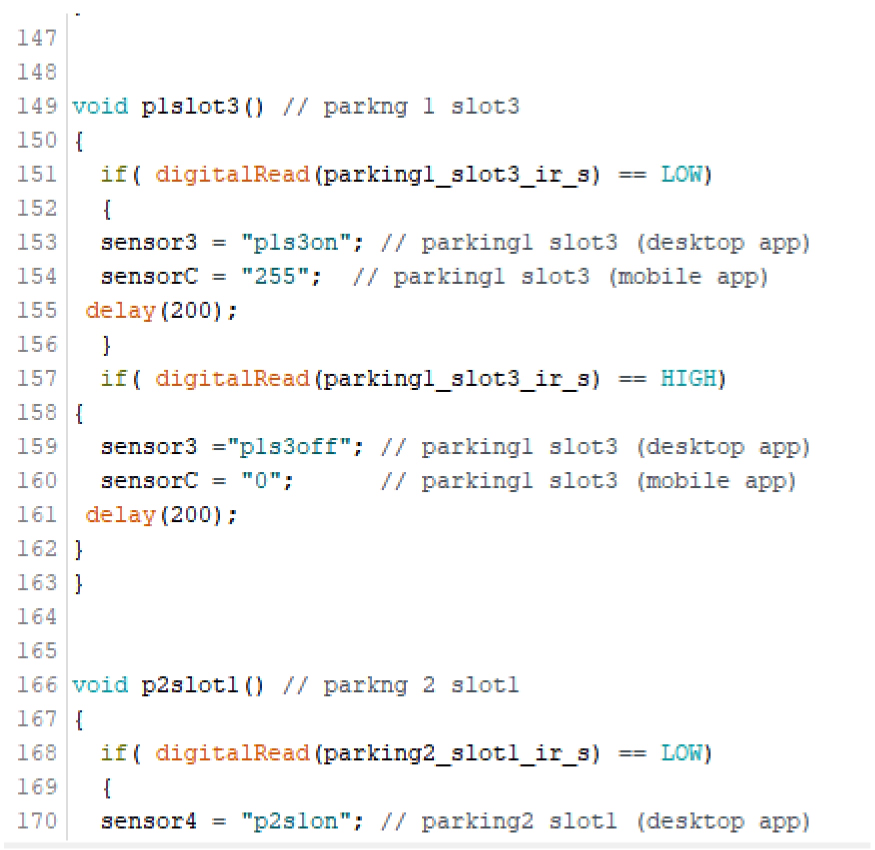
```

   //INORDER TO GET THE ENTIRE DATA STATUS OF THE SENSORS FOR DESKTOP APPLICATION. (cdata)
   cdata = cdata + sensor1 +"," + sensor2 + ","+ sensor3 +","+ sensor4 + "," + sensor5 + "," + sensor6 +",";
                                                                           // comma will be used a delimeter

   //INORDER TO GET THE ENTIRE DATA STATUS OF THE SENSORS FOR MOBILE APPLICATION. (ddata)
   ddata = ddata + sensorA +"," + sensorB + ","+ sensorC +","+ sensorD + "," + sensorE + "," + sensorF +",";
                                                                           // comma will be used a delimeter

   Serial.println(cdata); //TO PRINT ON SERIAL MONITOR
   Serial.println(ddata); //TO PRINT ON SERIAL MONITOR
   nodemcu.println(ddata); //TO SEND THE ddata TO NODE MCU ESP 8266 WIFI MODULE

   delay(2000); // enables a delay of 2 seconds for syncronization of the timer of desktop and mobile apps

   cdata = ""; //empting the collective sensor data
   ddata = ""; //empting the collective sensor data

//AGAIN RESETTING ALL THE PARKING LOT SENSOR STATUS TO HIGH FOR ANOTHER SET OF DATA ENTRY
digitalWrite(parking1_slot1_ir_s, HIGH);
digitalWrite(parking1_slot2_ir_s, HIGH);
digitalWrite(parking1_slot3_ir_s, HIGH);

digitalWrite(parking2_slot1_ir_s, HIGH);
```



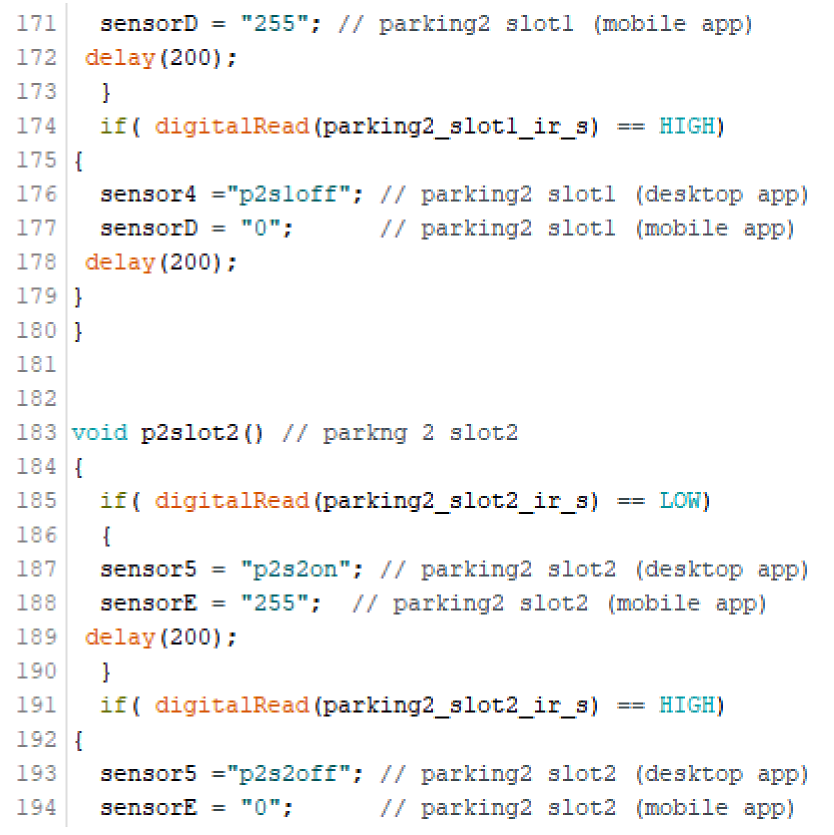
```
digitalWrite(parking2_slot2_ir_s, HIGH);
digitalWrite(parking2_slot3_ir_s, HIGH);

                                    //NOTE: SENSOR STATUS IS HIGH WHEN THERE IS NO VEHICLE IN FRONT OF SENSOR
                                    //      IT IS LOW WHEN THERE IS CAR IN FRONT OF THE SENSOR
}

//PXslotX is a user defined function, it has no return type and it doesn not take any argument as the input.
//if there is a car infront of the sensor it gives digital logic 0, and if no car then it give digital logic 1,
//depending on this, then we store pXsXon or pXsXoff.
//for the desktop app and 255 or 0 for the mobile app indication


//SUB-FUNCTIONS


void p1slot1() // parkng 1 slot1
{
  if( digitalRead(parking1_slot1_ir_s) == LOW)
  {
  sensor1 = "p1s1on"; // parking1 slot1 (desktop app)
  sensorA = "255";    // parking1 slot1 (mobile app)
 delay(200);
  }
```



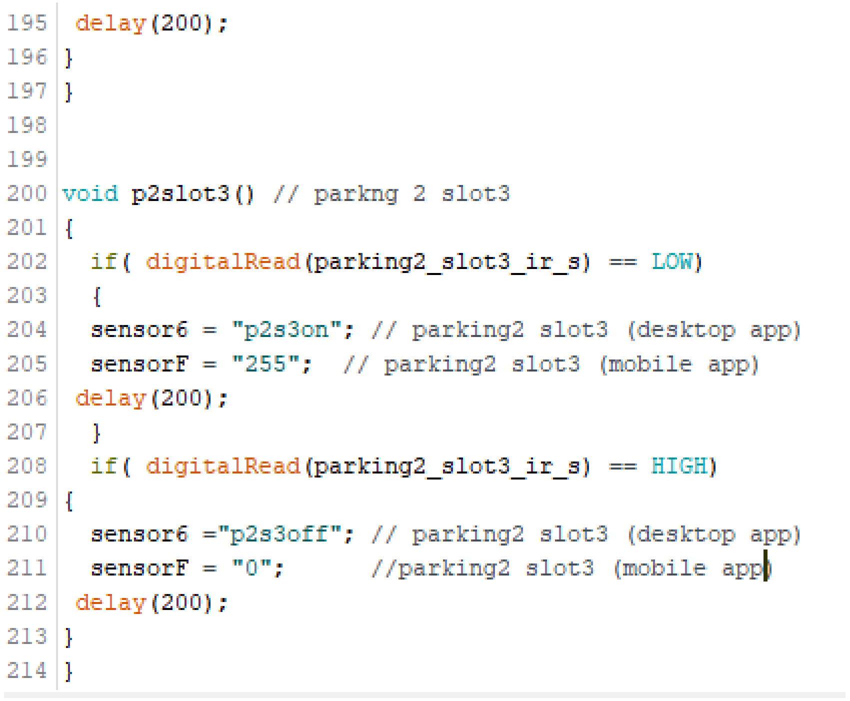
```
  if( digitalRead(parking1_slot1_ir_s) == HIGH)
{
  sensor1 ="p1s1off"; // parking1 slot1 (desktop app)
  sensorA = "0";      // parking1 slot1 (mobile app)
 delay(200);
}
}


void p1slot2() // parkng 1 slot2
{
  if( digitalRead(parking1_slot2_ir_s) == LOW)
  {
  sensor2 = "p1s2on"; // parking1 slot2 (desktop app)
  sensorB = "255";  // parking1 slot2 (mobile app)
 delay(200);
  }
  if( digitalRead(parking1_slot2_ir_s) == HIGH)
{
  sensor2 ="p1s2off"; // parking1 slot1 (desktop app)
  sensorB = "0";      // parking1 slot2 (mobile app)
 delay(200);
}
}
```



```


void p1slot3() // parkng 1 slot3
{
  if( digitalRead(parking1_slot3_ir_s) == LOW)
  {
  sensor3 = "p1s3on"; // parking1 slot3 (desktop app)
  sensorC = "255";  // parking1 slot3 (mobile app)
 delay(200);
  }
  if( digitalRead(parking1_slot3_ir_s) == HIGH)
{
  sensor3 ="p1s3off"; // parking1 slot3 (desktop app)
  sensorC = "0";      // parking1 slot3 (mobile app)
 delay(200);
}
}


void p2slot1() // parkng 2 slot1
{
  if( digitalRead(parking2_slot1_ir_s) == LOW)
  {
  sensor4 = "p2s1on"; // parking2 slot1 (desktop app)
```

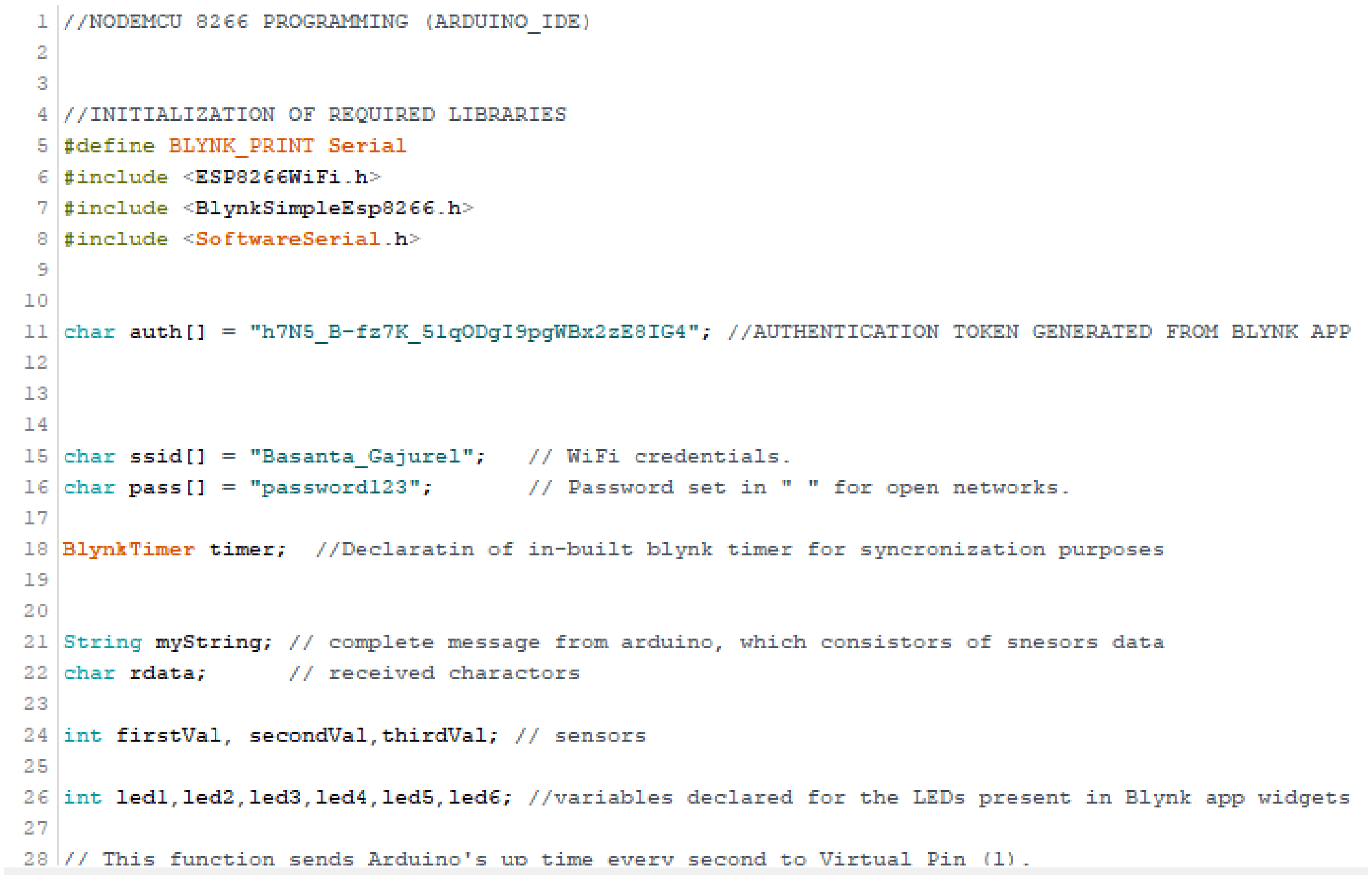


```
  sensorD = "255"; // parking2 slot1 (mobile app)
 delay(200);
  }
  if( digitalRead(parking2_slot1_ir_s) == HIGH)
{
  sensor4 ="p2s1off"; // parking2 slot1 (desktop app)
  sensorD = "0";      // parking2 slot1 (mobile app)
 delay(200);
}
}


void p2slot2() // parkng 2 slot2
{
  if( digitalRead(parking2_slot2_ir_s) == LOW)
  {
  sensor5 = "p2s2on"; // parking2 slot2 (desktop app)
  sensorE = "255";  // parking2 slot2 (mobile app)
 delay(200);
  }
  if( digitalRead(parking2_slot2_ir_s) == HIGH)
{
  sensor5 ="p2s2off"; // parking2 slot2 (desktop app)
  sensorE = "0";      // parking2 slot2 (mobile app)
```

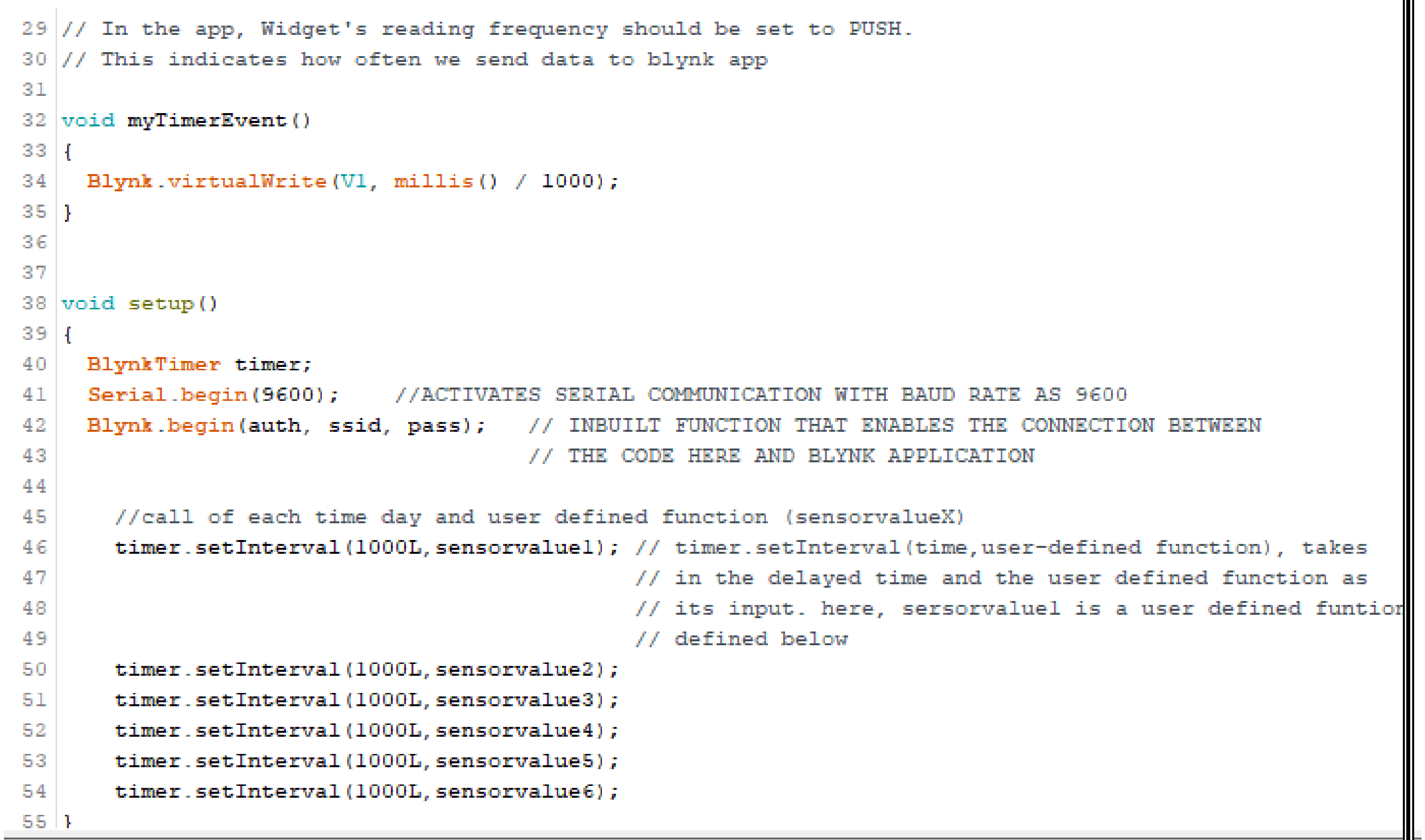


```
 delay(200);
}
}


void p2slot3() // parkng 2 slot3
{
  if( digitalRead(parking2_slot3_ir_s) == LOW)
  {
  sensor6 = "p2s3on"; // parking2 slot3 (desktop app)
  sensorF = "255";  // parking2 slot3 (mobile app)
 delay(200);
  }
  if( digitalRead(parking2_slot3_ir_s) == HIGH)
{
  sensor6 ="p2s3off"; // parking2 slot3 (desktop app)
  sensorF = "0";      //parking2 slot3 (mobile app)
 delay(200);
}
}
```

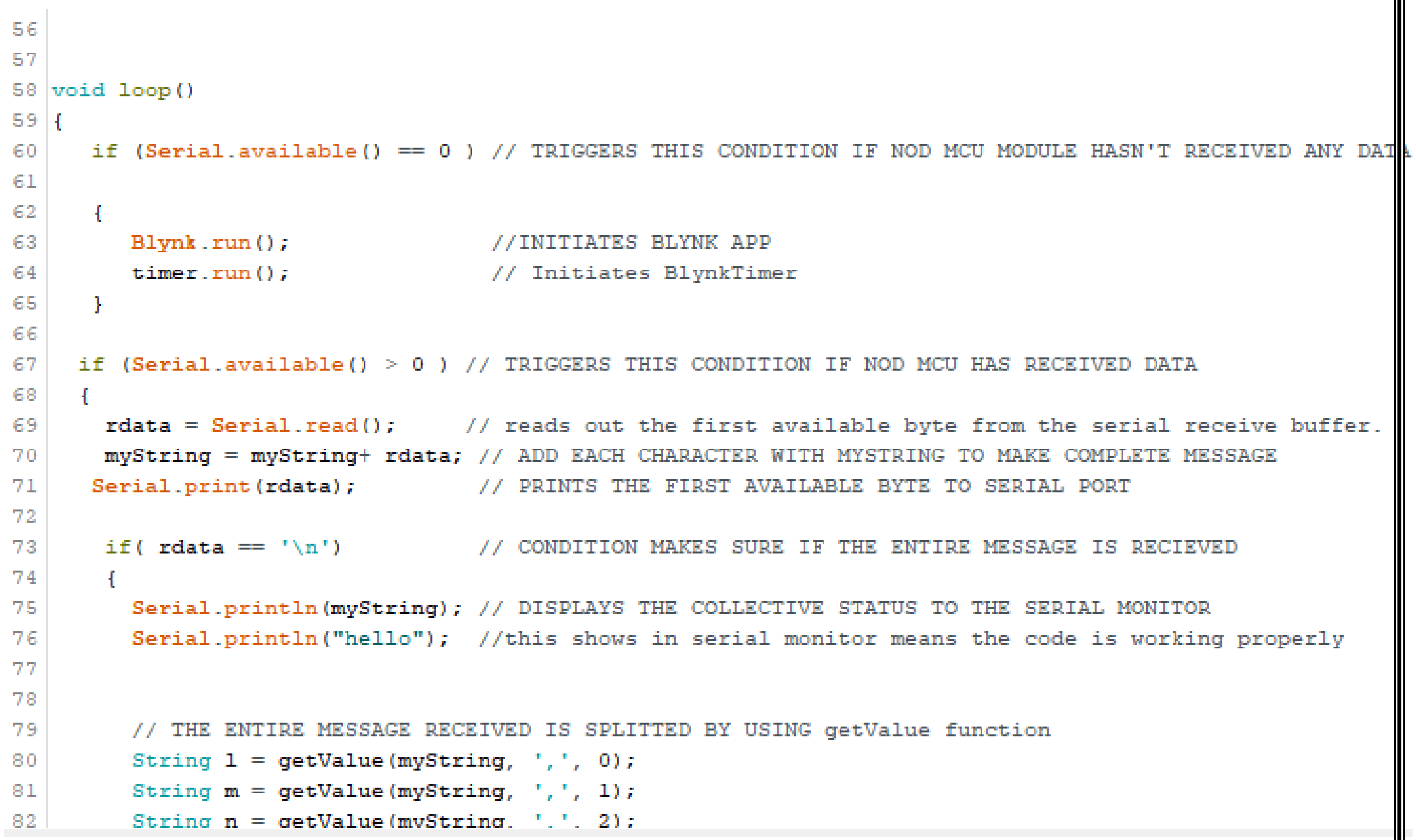
NODE MCU 8266 WIFI MODULE programming code in arduino IDE:



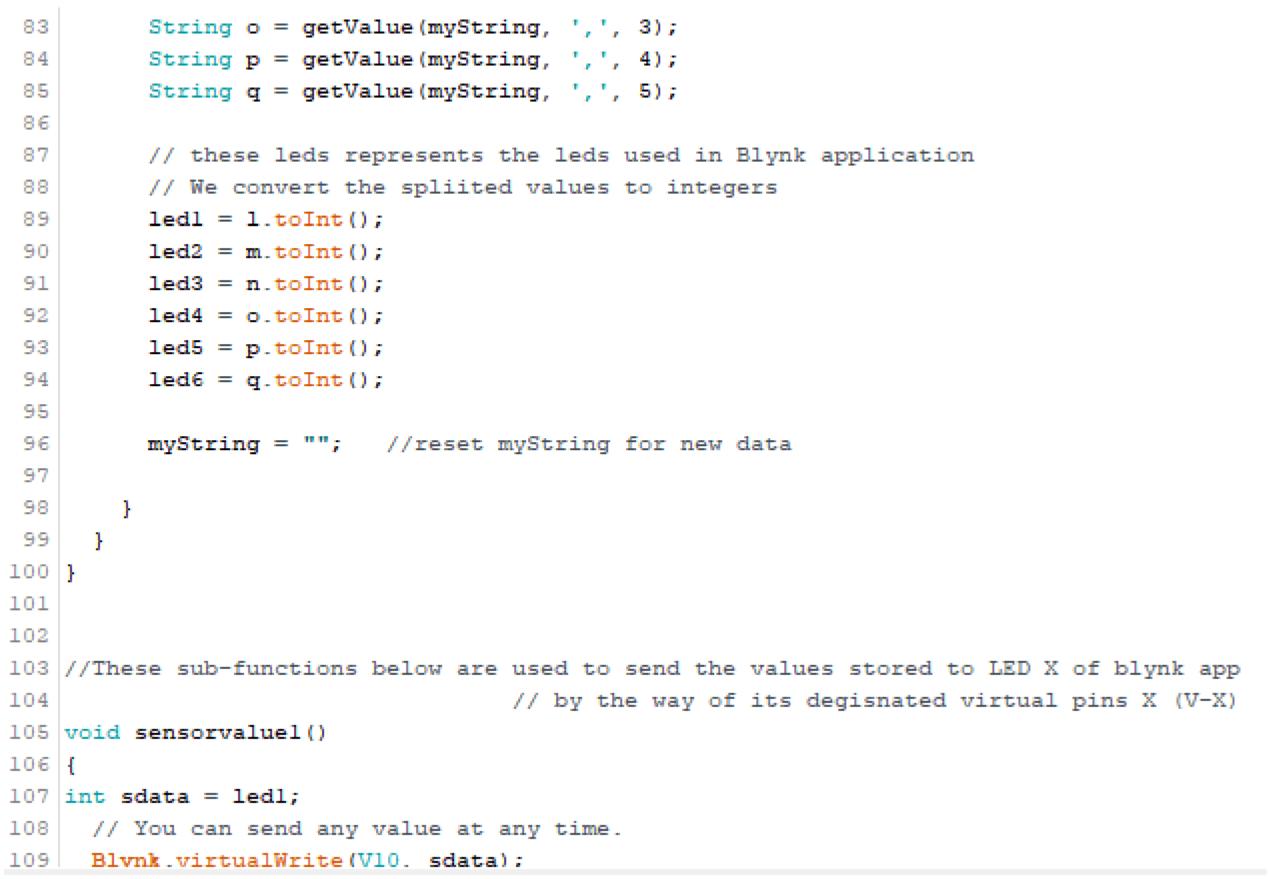
```
//NODEMCU 8266 PROGRAMMING (ARDUINO_IDE)


//INITIALIZATION OF REQUIRED LIBRARIES
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SoftwareSerial.h>


char auth[] = "h7N5_B-fz7K_51qODgI9pgWBx2zE8IG4"; //AUTHENTICATION TOKEN GENERATED FROM BLYNK APP



char ssid[] = "Basanta_Gajurel";   // WiFi credentials.
char pass[] = "password123";       // Password set in " " for open networks.

BlynkTimer timer;  //Declaratin of in-built blynk timer for syncronization purposes


String myString; // complete message from arduino, which consistors of snesors data
char rdata;      // received charactors

int firstVal, secondVal,thirdVal; // sensors

int led1,led2,led3,led4,led5,led6; //variables declared for the LEDs present in Blynk app widgets

// This function sends Arduino's up time every second to Virtual Pin (1).
```



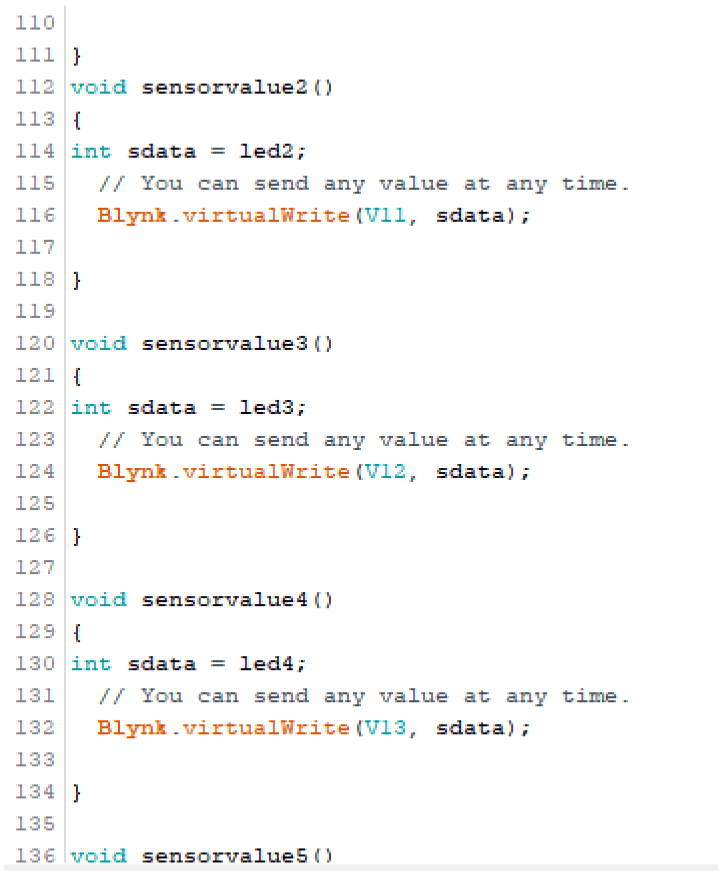
```
// In the app, Widget's reading frequency should be set to PUSH.
// This indicates how often we send data to blynk app

void myTimerEvent()
{
  Blynk.virtualWrite(V1, millis() / 1000);
}


void setup()
{
  BlynkTimer timer;
  Serial.begin(9600);    //ACTIVATES SERIAL COMMUNICATION WITH BAUD RATE AS 9600
  Blynk.begin(auth, ssid, pass);   // INBUILT FUNCTION THAT ENABLES THE CONNECTION BETWEEN
                                   // THE CODE HERE AND BLYNK APPLICATION

    //call of each time day and user defined function (sensorvalueX)
    timer.setInterval(1000L,sensorvalue1); // timer.setInterval(time,user-defined function), takes
                                           // in the delayed time and the user defined function as
                                           // its input. here, sersorvalue1 is a user defined funtio
                                           // defined below
    timer.setInterval(1000L,sensorvalue2);
    timer.setInterval(1000L,sensorvalue3);
    timer.setInterval(1000L,sensorvalue4);
    timer.setInterval(1000L,sensorvalue5);
    timer.setInterval(1000L,sensorvalue6);
}
```



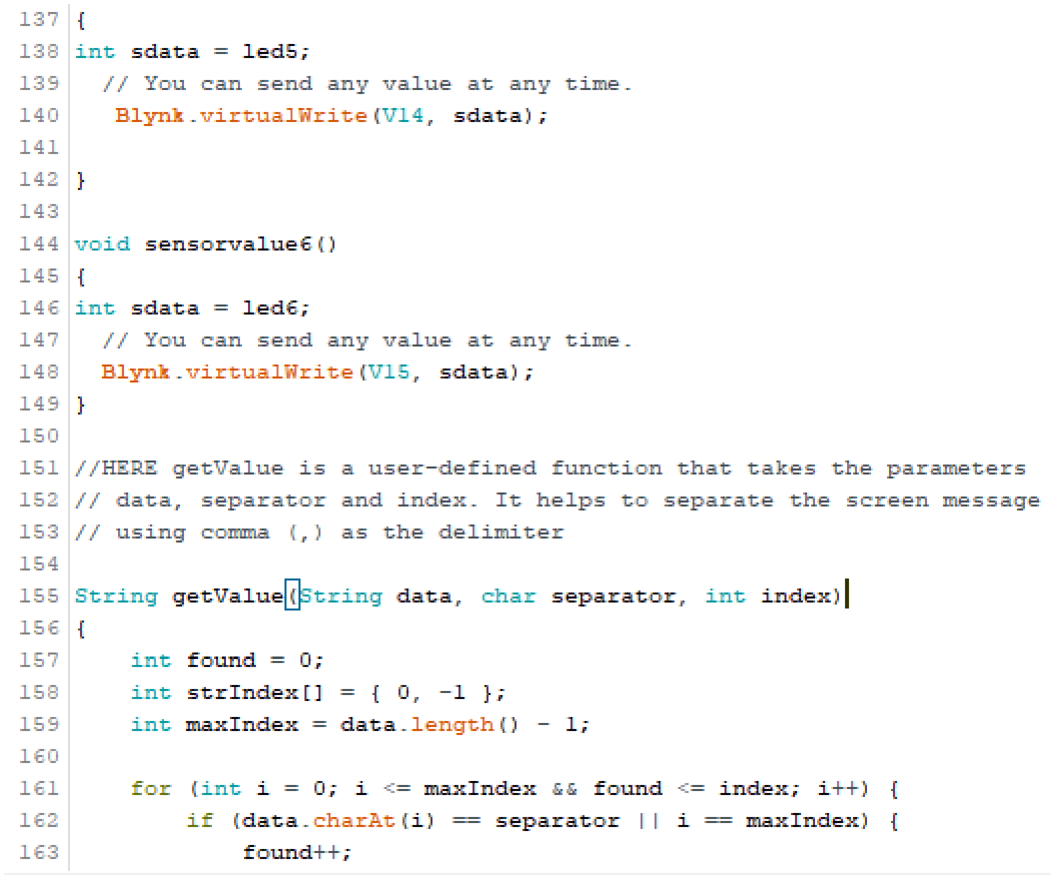
```


void loop()
{
   if (Serial.available() == 0 ) // TRIGGERS THIS CONDITION IF NOD MCU MODULE HASN'T RECEIVED ANY DATA

   {
      Blynk.run();                //INITIATES BLYNK APP
      timer.run();                // Initiates BlynkTimer
   }

  if (Serial.available() > 0 ) // TRIGGERS THIS CONDITION IF NOD MCU HAS RECEIVED DATA
  {
    rdata = Serial.read();     // reads out the first available byte from the serial receive buffer.
    myString = myString+ rdata; // ADD EACH CHARACTER WITH MYSTRING TO MAKE COMPLETE MESSAGE
   Serial.print(rdata);         // PRINTS THE FIRST AVAILABLE BYTE TO SERIAL PORT

    if( rdata == '\n')          // CONDITION MAKES SURE IF THE ENTIRE MESSAGE IS RECIEVED
    {
      Serial.println(myString); // DISPLAYS THE COLLECTIVE STATUS TO THE SERIAL MONITOR
      Serial.println("hello");  //this shows in serial monitor means the code is working properly


      // THE ENTIRE MESSAGE RECEIVED IS SPLITTED BY USING getValue function
      String l = getValue(myString, ',', 0);
      String m = getValue(myString, ',', 1);
      String n = getValue(myString, ',', 2);
```



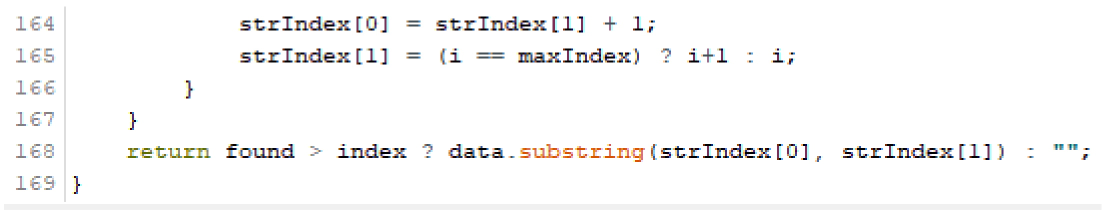
```
      String o = getValue(myString, ',', 3);
      String p = getValue(myString, ',', 4);
      String q = getValue(myString, ',', 5);

      // these leds represents the leds used in Blynk application
      // We convert the spliited values to integers
      led1 = l.toInt();
      led2 = m.toInt();
      led3 = n.toInt();
      led4 = o.toInt();
      led5 = p.toInt();
      led6 = q.toInt();

      myString = "";   //reset myString for new data

    }
  }
}


//These sub-functions below are used to send the values stored to LED X of blynk app
                                // by the way of its degisnated virtual pins X (V-X)
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()
```

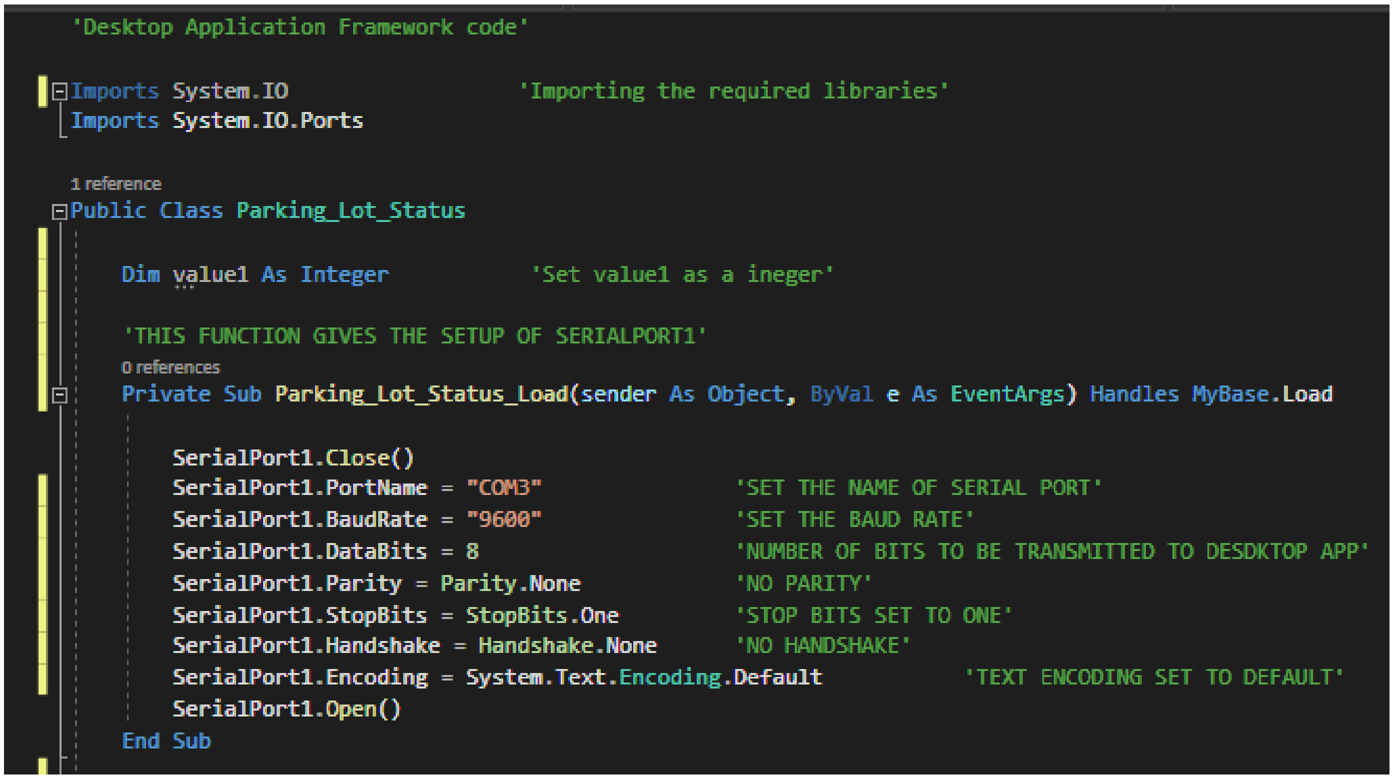


```
{
int sdata = led5;
  // You can send any value at any time.
   Blynk.virtualWrite(V14, sdata);

}

void sensorvalue6()
{
int sdata = led6;
  // You can send any value at any time.
  Blynk.virtualWrite(V15, sdata);
}

//HERE getValue is a user-defined function that takes the parameters
// data, separator and index. It helps to separate the screen message
// using comma (,) as the delimiter

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++;
```

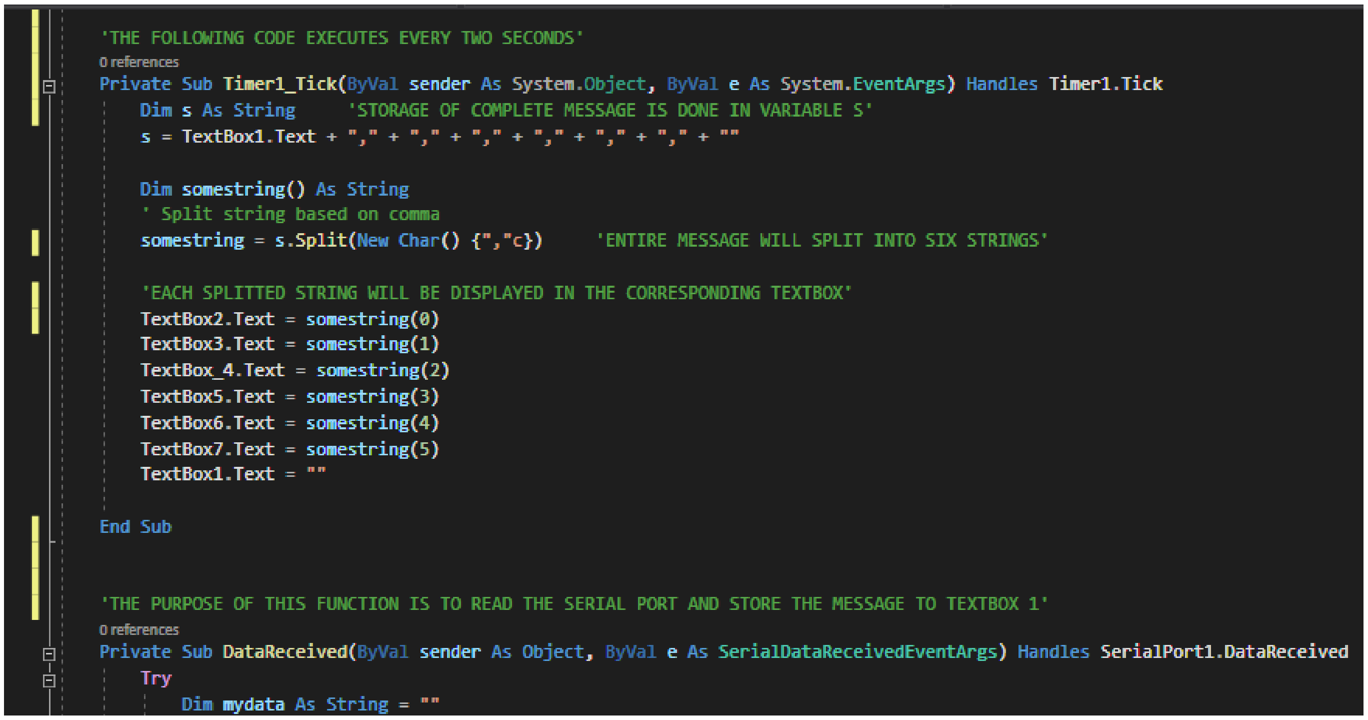


```
            strIndex[0] = strIndex[1] + 1;
            strIndex[1] = (i == maxIndex) ? i+1 : i;
        }
    }
    return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}
```

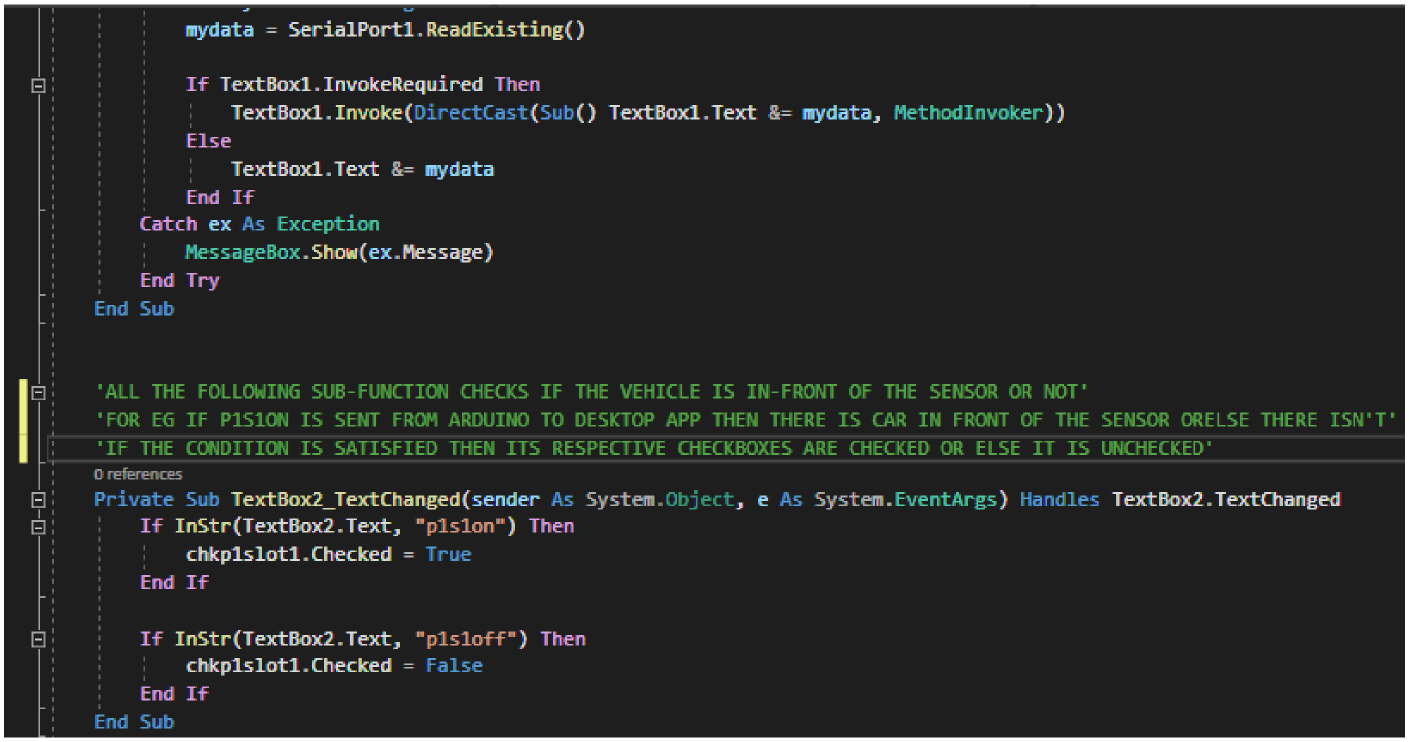
VB.NET CODE FOR DESKTOP APPLICATION:



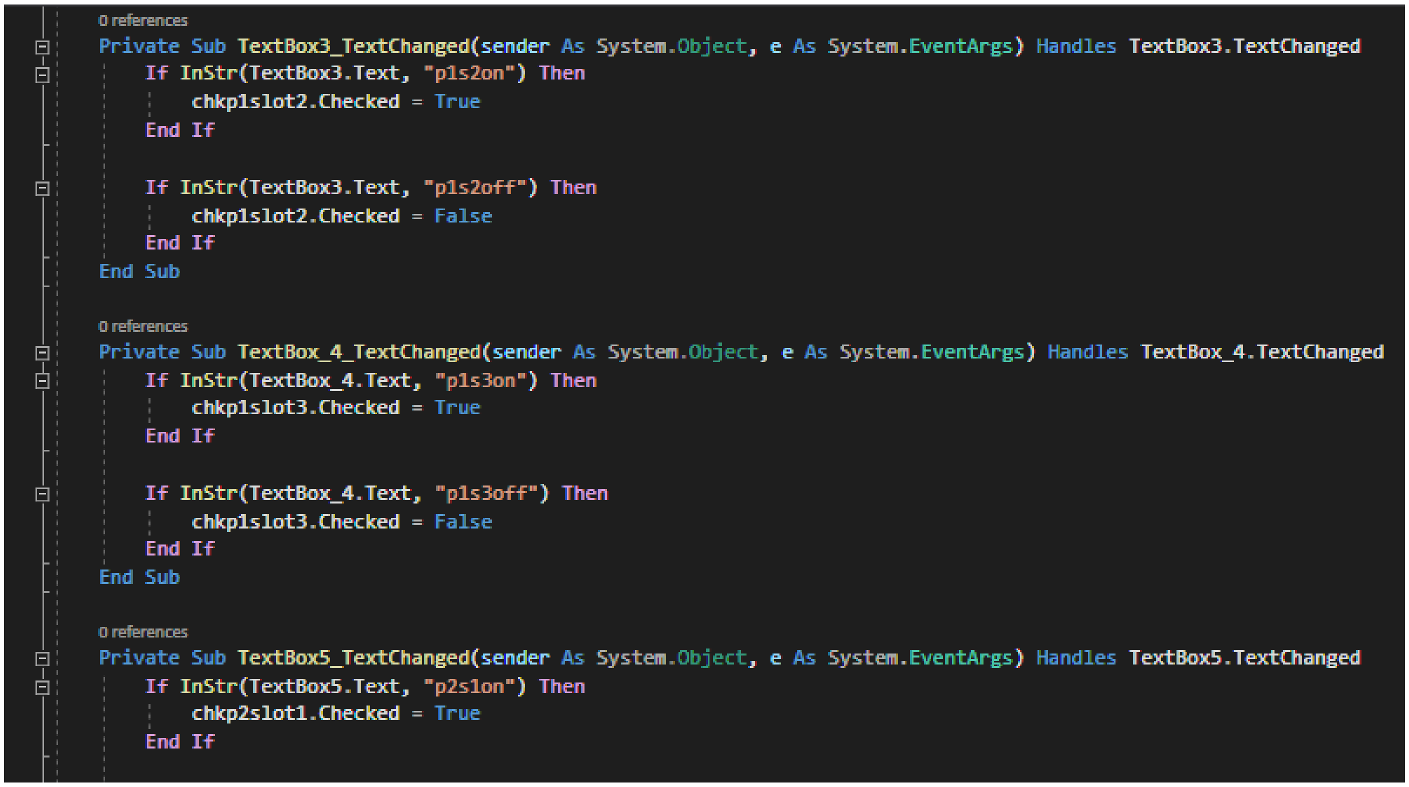
```
    'Desktop Application Framework code'

Imports System.IO                         'Importing the required libraries'
Imports System.IO.Ports

1 reference
Public Class Parking_Lot_Status

    Dim value1 As Integer             'Set value1 as a ineger'

    'THIS FUNCTION GIVES THE SETUP OF SERIALPORT1'
    0 references
    Private Sub Parking_Lot_Status_Load(sender As Object, ByVal e As EventArgs) Handles MyBase.Load

        SerialPort1.Close()
        SerialPort1.PortName = "COM3"               'SET THE NAME OF SERIAL PORT'
        SerialPort1.BaudRate = "9600"               'SET THE BAUD RATE'
        SerialPort1.DataBits = 8                    'NUMBER OF BITS TO BE TRANSMITTED TO DESDKTOP APP'
        SerialPort1.Parity = Parity.None            'NO PARITY'
        SerialPort1.StopBits = StopBits.One         'STOP BITS SET TO ONE'
        SerialPort1.Handshake = Handshake.None      'NO HANDSHAKE'
        SerialPort1.Encoding = System.Text.Encoding.Default          'TEXT ENCODING SET TO DEFAULT'
        SerialPort1.Open()
    End Sub
```



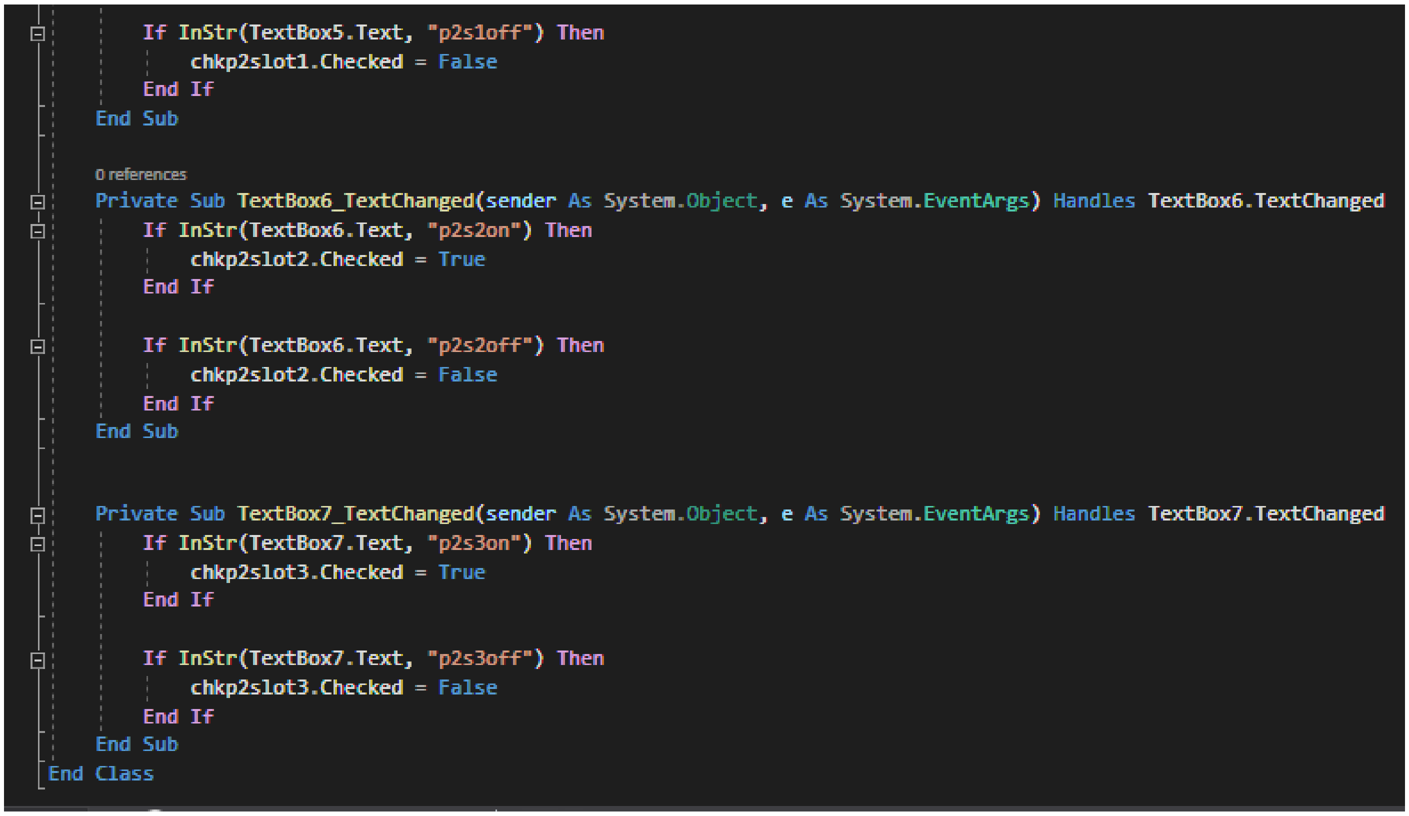
```
    'THE FOLLOWING CODE EXECUTES EVERY TWO SECONDS'
    0 references
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        Dim s As String      'STORAGE OF COMPLETE MESSAGE IS DONE IN VARIABLE S'
        s = TextBox1.Text + "," + "," + "," + "," + "," + "," + ""

        Dim somestring() As String
        ' Split string based on comma
        somestring = s.Split(New Char() {","c})      'ENTIRE MESSAGE WILL SPLIT INTO SIX STRINGS'

        'EACH SPLITTED STRING WILL BE DISPLAYED IN THE CORRESPONDING TEXTBOX'
        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


    'THE PURPOSE OF THIS FUNCTION IS TO READ THE SERIAL PORT AND STORE THE MESSAGE TO TEXTBOX 1'
    0 references
    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


    'ALL THE FOLLOWING SUB-FUNCTION CHECKS IF THE VEHICLE IS IN-FRONT OF THE SENSOR OR NOT'
    'FOR EG IF P1S1ON IS SENT FROM ARDUINO TO DESKTOP APP THEN THERE IS CAR IN FRONT OF THE SENSOR ORELSE THERE ISN'T'
    'IF THE CONDITION IS SATISFIED THEN ITS RESPECTIVE CHECKBOXES ARE CHECKED OR ELSE IT IS UNCHECKED'
    0 references
    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
```



```
    0 references
    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

    0 references
    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

    0 references
    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

    0 references
    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
```

EXECUTION AND DISCUSSION

The result of this project Smart Parking system (vehicle monitoring), one of which is creating a parking monitoring app and desktop application that is connected to the prototype that was created as shown below

Here all the slots of the parking lots are empty.

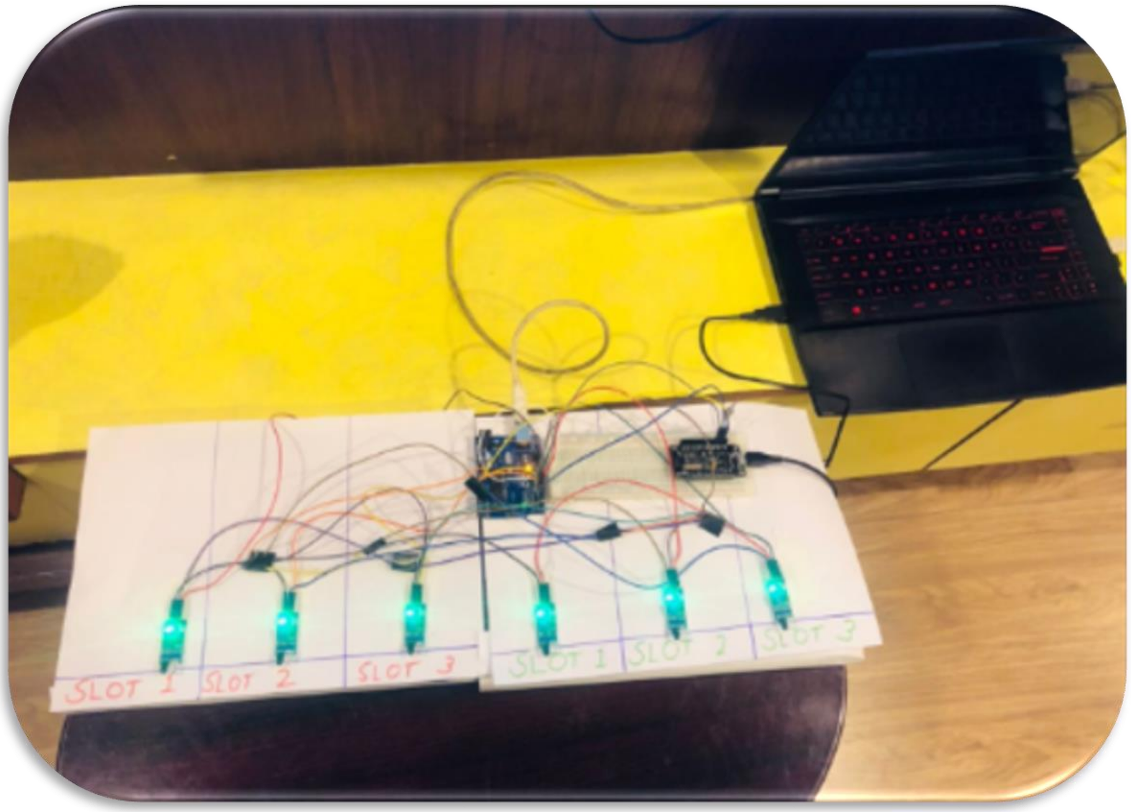


Fig 1: Prototype of Smart Parking System

The initial status of the blynk application is shown in figure 2 and that of desktop application is shown is in figure 3 respectively.

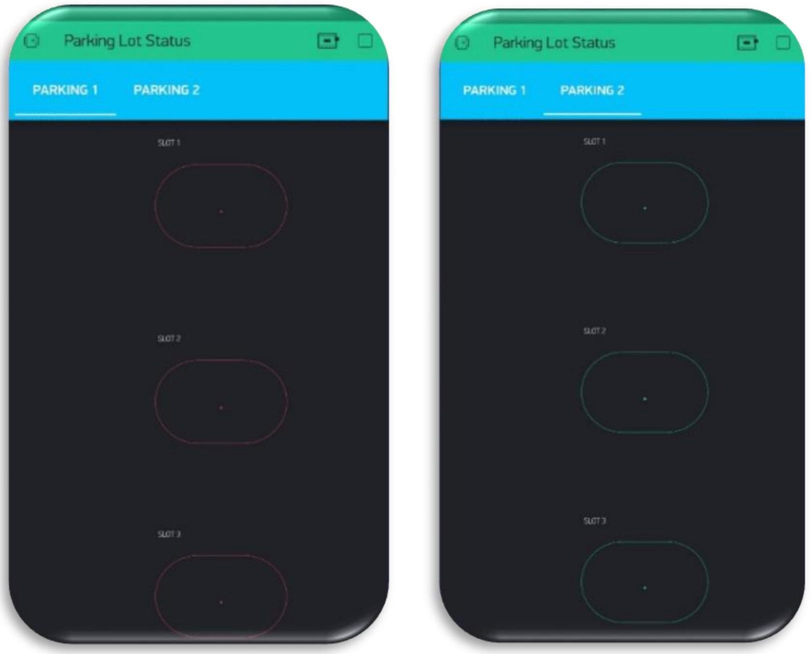
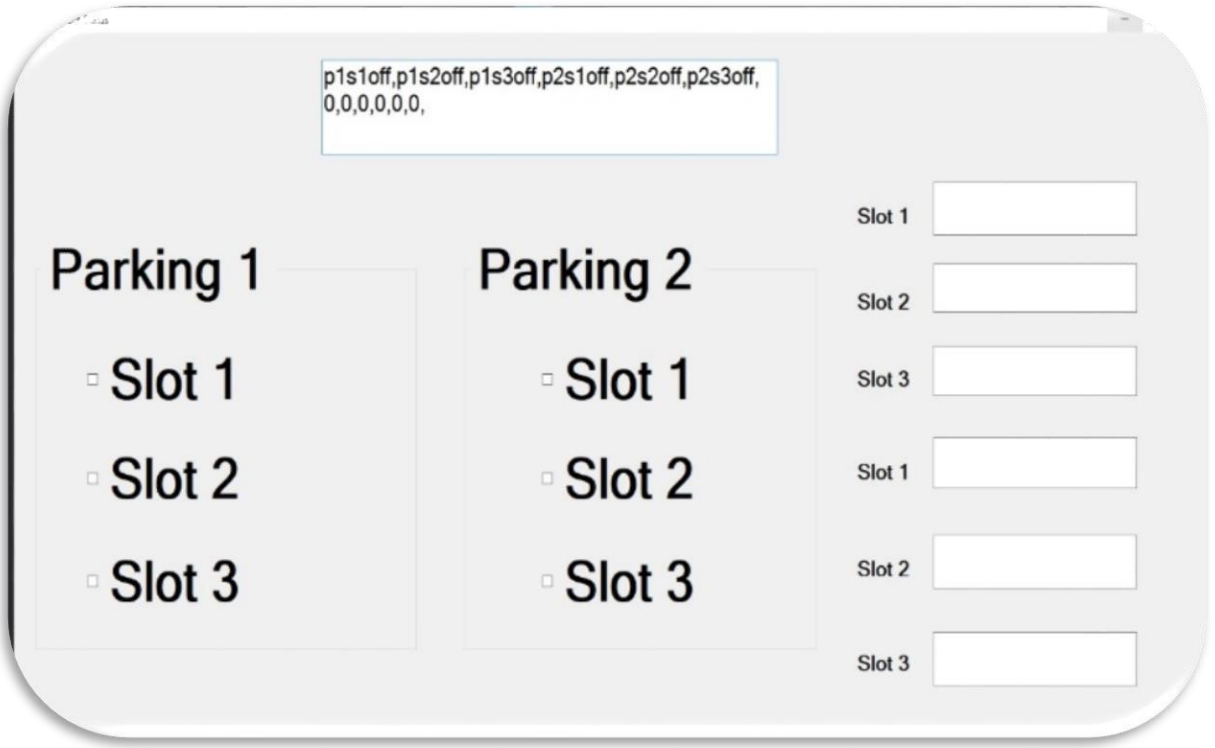


Figure 2: Mobile app; all the LEDs are at off state



p1s1off,p1s2off,p1s3off,p2s1off,p2s2off,p2s3off,
0,0,0,0,0,0,

Figure 3: Desktop app; all boxes are not ticked

Now, after filling up the slot 1 of parking lot 1 and slot 2 of parking lot 2 as shown in figure 4, we will get the indication in our desktop and mobile application as shown in figure 5 and 6 respectively.

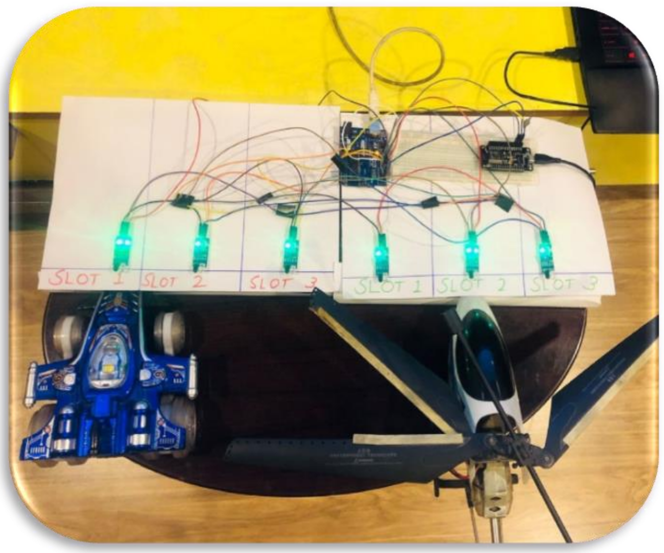
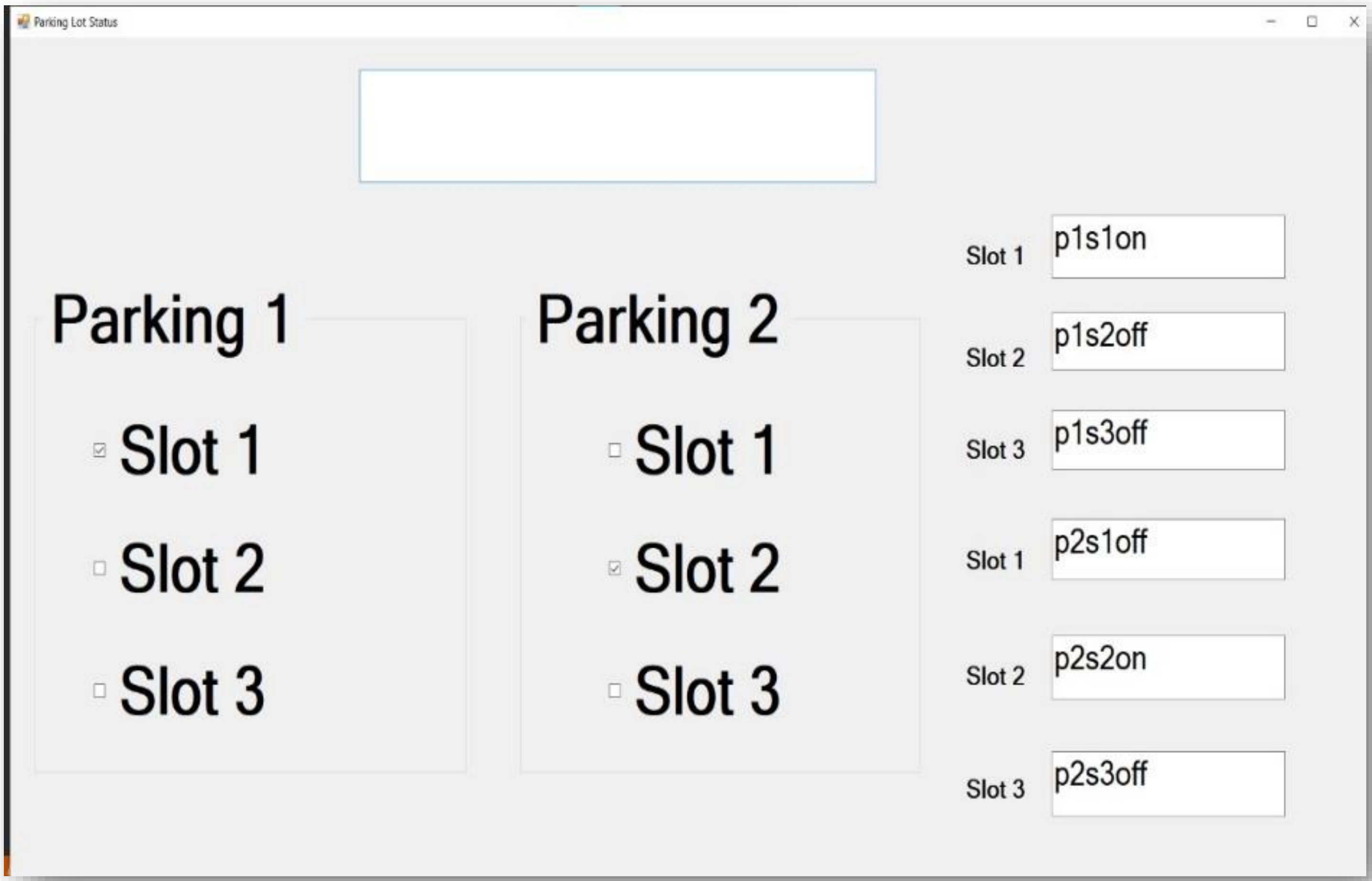


Figure 4: Vehicles added at p1s1 and p2s2



Parking 1

- ☑ Slot 1
- ☐ Slot 2
- ☐ Slot 3

Parking 2

- ☐ Slot 1
- ☑ Slot 2
- ☐ Slot 3

Slot 1	p1s1on
Slot 2	p1s2off
Slot 3	p1s3off
Slot 1	p2s1off
Slot 2	p2s2on
Slot 3	p2s3off

Figure 5: p1s1 and p2s2 are tick boxes are checked

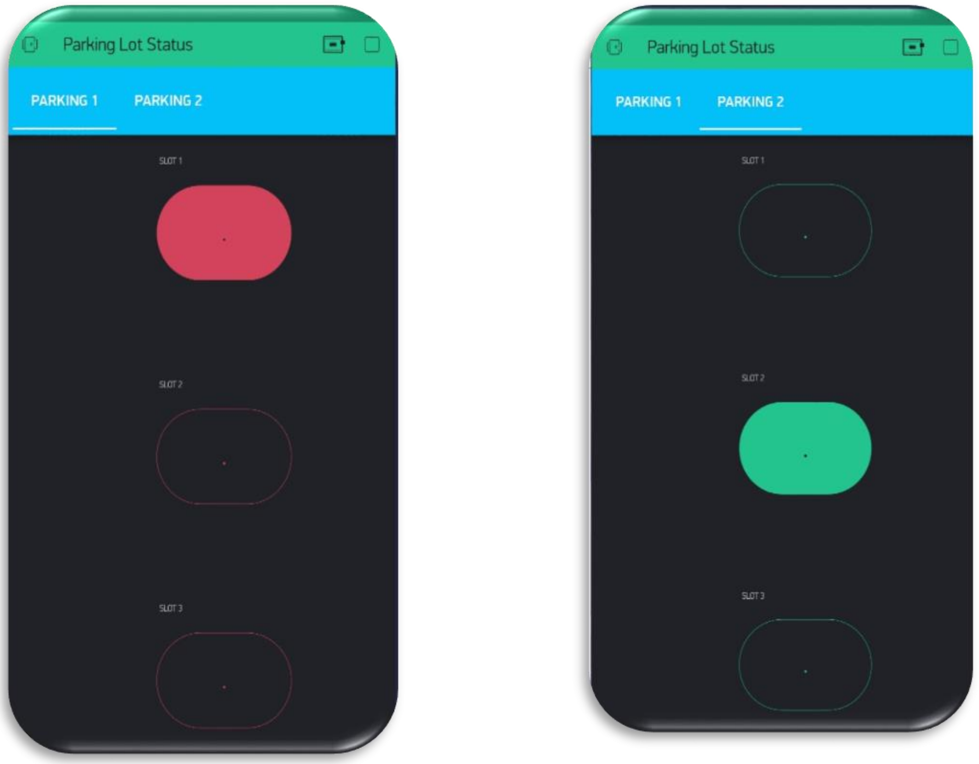


Fig 6: Blynk application showing p1s1 and p2s2 as occupied parking slots

Now, after filling up the all the parking slots, as shown in figure 7, we will get the indication in our desktop and mobile application (as per the parking lots) as shown in figure 8, 9 and 10 respectively.

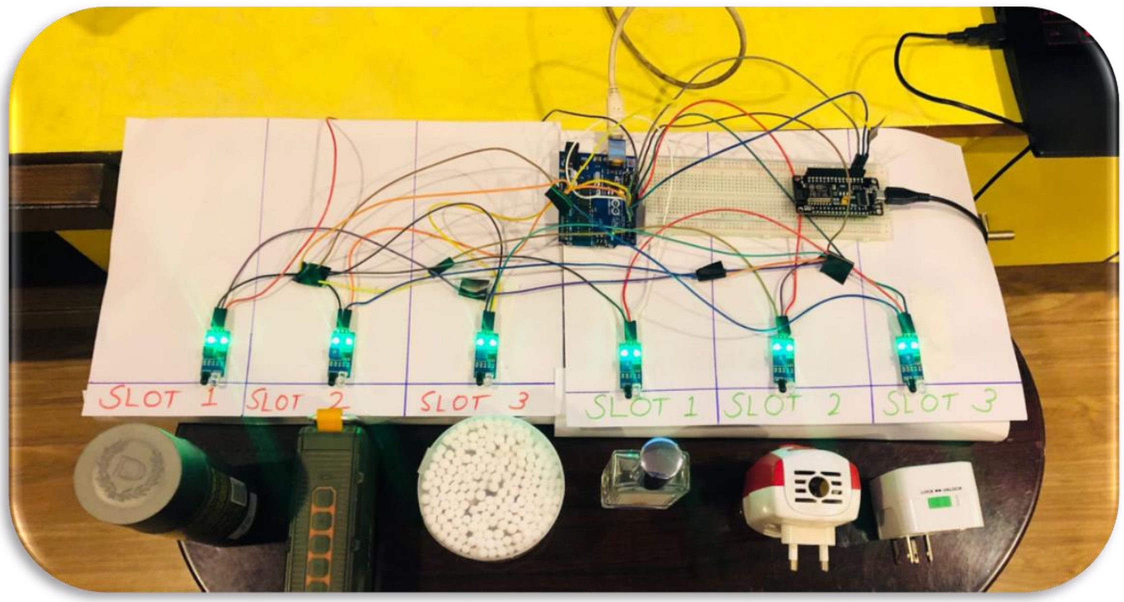


Fig 7: All the parking slots are filled

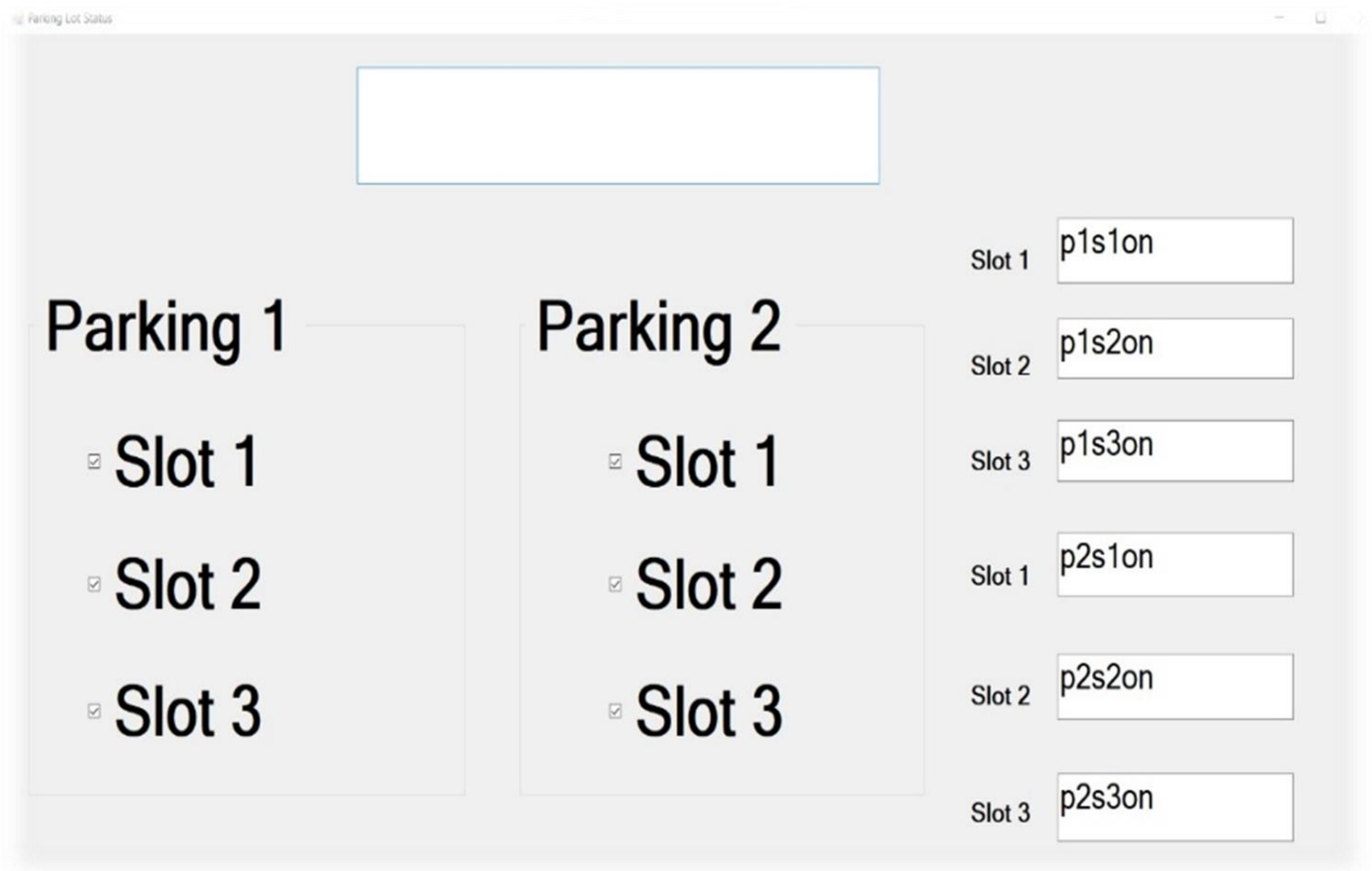


Figure 8: Every tick box is checked

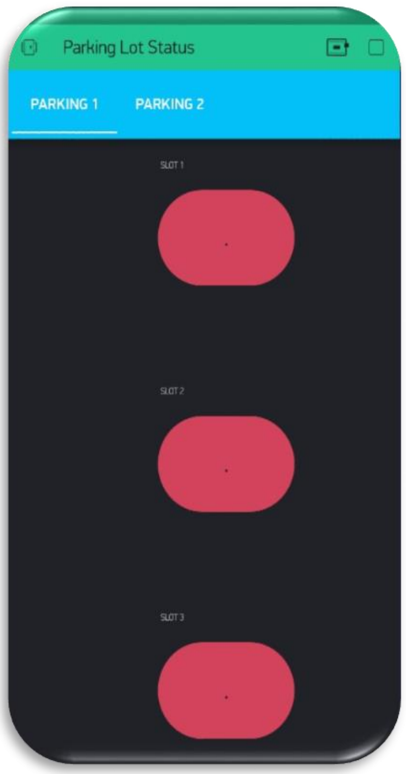


Figure 9: All the slots of 'parking 1' are filled

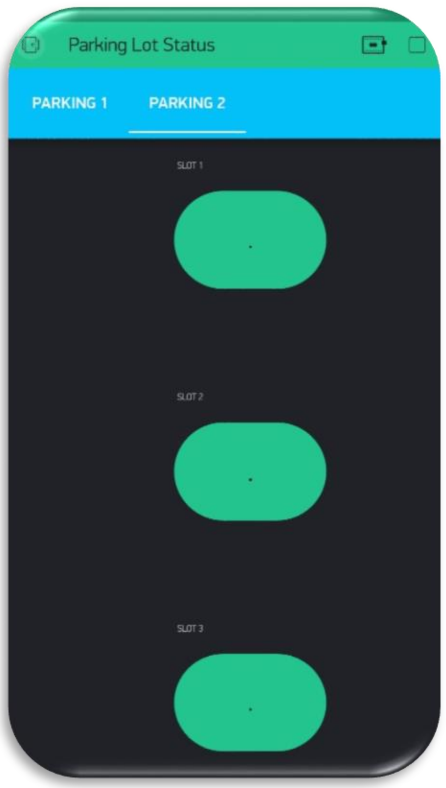


Figure 10: All the slots of 'parking 2' are filled

CONCLUSION

The services provided by Smart Parking system (vehicle monitoring) 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. An attractive and effective application was designed for iphone, android mobile phones (with the help of blynk application) and desktop application (windows and linux with the help of vb.net). 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, reducing pollution and fuel consumption.

Link of Project execution (along with basic description):

https://www.youtube.com/watch?v=g9auJxSghq4

Note: The project execution begins from the timestamp: [4:34]

References:

[A MULTI-AGENT ORGANIZATION FOR THE GOVERNANCE OF MACHINE-TOMACHINE
SYSTEMS, IEEE 2011]
https://ieeexplore.ieee.org/document/6040668
[SPARK: A NEW VANET-BASED SMART PARKING SCHEME FOR LARGE PARKING LOTS, IEEE 2009]
https://ieeexplore.ieee.org/document/5062057
A privacy-preserving smart parking system using an IoT elliptic curve based security platform
https://dl.acm.org/citation.cfm?id=2972908
[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[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
[Smart Parking 26516]
https://www.happiestminds.com/whitepapers/smart-parking.pdf
[PARKING MANAGEMENT SYSTEM AND FEE COLLECTION BASED ON NUMBER PLATE RECOGNITION]
https://www.researchgate.net/profile/Mohd-Suhaimi-Nur-Farahana/publication/281060377_Automatic_Parking_Management_System_and_Parking_Fee_Collection_Based_on_Number_Plate_Recognition/links/5662e9b208ae4931cd5ebd40/Automatic-Parking-Management-System-and-Parking-Fee-Collection-Based-on-Number-Plate-Recognition.pdf
[PARKING SYSTEM THAT SOLVES MILP]
https://www.sciencedirect.com/science/article/pii/S1877042812043042
[VISON BASED PARKING SYSTEM]
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.432.1223&rep=rep1&type=pdf
[SMART PARKING USING RFID TECHNOLOGY]
https://ieeexplore.ieee.org/abstract/document/4368108