

SMART PARKING

With growing popularity of Smart Cities, there is always a demand for smart solutions for every domain. The IoT has enabled the possibility of Smart Cities with it's over the internet control feature.

A person can control the devices installed in his home or office from anywhere in the world by just using a smartphone or any internet connected devices.

There are multiple domains in a smart city and Smart Parking is one of the popular domain in the Smart City.

The smart parking involves finding innovative solution to optimize parking space and minimize the traffic congestion in urban areas then enhance the overall parking experience both drivers and parking operators.

This technology aims to make parking easier, more efficient and environment friendly

CODE:

For programming NodeMCU, just plug the NodeMCU to Computer with a Micro USB Cable and open Arduino IDE. The libraries are required for I2C Display and Servo Motor.

The LCD will display the availability of Parking Spaces and the Servo motors will be used to open and close the Entry and Exit gates. The Wire.h library will be used to interface LCD in i2c protocol.

The Pins for I2C in the ESP8266 NodeMCU are D1 (SCL) and D2 (SDA). The database here used will be Firebase so here we are also including the library (FirebaseArduino.h) for the same.

```
#include <ESP8266WiFi.h>
```



Edit with WPS Office

```
#include <Servo.h>
```

```
#include <LiquidCrystal_I2C.h>
```

```
#include <Wire.h>
```

```
#include <FirebaseArduino.h>
```

Then include the firebase credentials got from Google Firebase. These will include the Host name containing your project name and a secret key.

To find these values follow the previous tutorial on Firebase.

```
" #define FIREBASE_HOST "smart-parking-7f5b6.firebaseio.com"
```

```
#define FIREBASE_AUTH "suAkUQ4wXRPW7nA0zJQVsx3H2LmeBDPGmfTMBHCT"
```

Include the Wi-Fi Credentials such as WiFi SSID and Password.

```
#define WIFI_SSID "Smart Parking"
```

```
#define WIFI_PASSWORD "smartparking123"
```

Initialise I2C LCD with device address (Here it is 0x27) and type of LCD.

Also include the Servo Motors for entry and exit gate.

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
Servo myservo;
```

```
Servo myservo1;
```

Start the I2C communication for I2C LCD.

```
Wire.begin(D2, D1);
```

Connect the Entry and Exit Servo Motor to the D5, D6 Pins of the NodeMCU.

```
myservo.attach(D6);
```

```
myservos.attach(D5);
```

Select the Trigger Pin of Ultrasonic sensor as Output and Echo Pin as Input. The



Edit with WPS Office

ultrasonic sensor will be used to detect the parking spot availability.

If Car has occupied the space then it will glow else it will not glow.

```
pinMode(TRIG, OUTPUT);
```

```
pinMode(ECHO, INPUT);
```

The two pins D0 and D4 of the NodeMCU are used to take the IR sensor reading.

The IR sensor will act as Entry and Exit gate sensor. This will detect car presence.

```
pinMode(carExited, INPUT);
```

```
pinMode(carEnter, INPUT);
```

Connect to WiFi and wait for some time till it gets connected.

```
WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
```

```
Serial.print("Connecting to ");
```

```
Serial.print(WIFI_SSID);
```

```
while (WiFi.status() != WL_CONNECTED) {
```

```
    Serial.print(".");
```

```
    delay(500);
```

```
}
```

Begin connection with Firebase with Host and Secret Key as credentials.

```
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
```

Begin i2c 16x2 LCD and set cursor position at 0th row 0th column.

```
lcd.begin();
```

```
lcd.setCursor(0, 0);
```

Take the Distance from Ultrasonic sensor. This will be used to detect the vehicle presence in the particular spot. First send the 2 microsecond pulse and then read the



received pulse. Then convert it to the 'cm'.

```
digitalWrite(TRIG, LOW);  
delayMicroseconds(2);  
digitalWrite(TRIG, HIGH);  
delayMicroseconds(10);  
digitalWrite(TRIG, LOW);  
duration = pulseIn(ECHO, HIGH);  
distance = (duration / 2) / 29.1;
```

Digitally read the IR sensor pin as entry sensor and check if it is high.

If it is high then increment entry count and print it to 16x2 LCD display and also to serial monitor.

```
int carEntry = digitalRead(carEnter);  
if (carEntry == HIGH) {  
    countYes++;  
    Serial.print("Car Entered = "); Serial.println(countYes );  
    lcd.setCursor(0, 1);  
    lcd.print("Car Entered");  
}
```

Also move the servo motor angle to open entry gate.

Change the angle according your use case.

```
for (pos = 140; pos >= 45; pos -= 1) {  
    myservos.write(pos);  
    delay(5);  
}
```



```

delay(2000);

for (pos = 45; pos <= 140; pos += 1) {
    // in steps of 1 degree
    myservos.write(pos);
    delay(5);
}

```

And send the reading to firebase by using *pushString* function of Firebase library.

```
Firebase.pushString("/Parking Status/", fireAvailable );
```

Do similar steps as above for Exit IR sensor and Exit servo motor.

```

int carExit = digitalRead(carExited);
if (carExit == HIGH) {
    countYes--;
    Serial.print("Car Exited = "); Serial.println(countYes);
    lcd.setCursor(0, 1);
    lcd.print("Car Exited");
    for (pos1 = 140; pos1 >= 45; pos1 -= 1) {
        myservo.write(pos1);
        delay(5);
    }
    delay(2000);
}

```

```

for (pos1 = 45; pos1 <= 140; pos1 += 1) {
    // in steps of 1 degree
    myservo.write(pos1);
}

```



```

    delay(5);
}

Firebase.pushString("/Parking Status/", fireAvailable );

lcd.clear();
}

```

Check if the car has come to the parking spot and if it has arrived then glow led giving the signal that the spot is full.

```

if (distance < 6) {
    Serial.println("Occupied ");
    digitalWrite(led, HIGH);
}

```

Else show that the spot is available.

```

if (distance > 6) {
    Serial.println("Available ");
    digitalWrite(led, LOW);
}

```

Calculate the total empty space inside the parking lot and save it in the string to send the data to firebase.

```
Empty = allSpace - countYes;
```

```
Available = String("Available= ") + String(Empty) + String("/") + String(allSpace);
```

```
fireAvailable = String("Available=") + String(Empty) + String("/") + String(allSpace);
```

Also print the data to the i2C LCD.

```
lcd.setCursor(0, 0);
```

```
lcd.print(Available);
```



HARDWARE SETUP:

- 1.connect the ultrasonic sensor to the ESP8266 as described in previous response.
- 2.connect the LCD display to the ESP8266 using the 12c interface.make sure you have the correct 12c address for your specific display.

Alternate Python code:

```
import machine
import time

from machine import Pin, I2C

from esp8266_i2c_lcd import I2cLcd

# Define GPIO pins for the ultrasonic sensor
trigger = Pin(5, Pin.OUT)
echo = Pin(4, Pin.IN)

# Define GPIO pins for the I2C communication
i2c = I2C(-1, scl=Pin(14), sda=Pin(2), freq=400000)

# Initialize the LCD display
lcd = I2cLcd(i2c, 0x3F, 2, 16)

def measure_distance():
    # Trigger the ultrasonic sensor
    trigger.on()
    time.sleep_us(10)
```



```
trigger.off()

# Measure the time for the echo to return
pulse_time = machine.time_pulse_us(echo, 1, 30000)

# Calculate the distance based on the time
distance = (pulse_time / 2) / 29.1 # 29.1 microseconds per centimeter

return distance

while True:
    distance = measure_distance()
    if distance < 10: # Adjust this threshold for your specific setup
        status = "Occupied"
    else:
        status = "Vacant"

    # Display status on the LCD
    lcd.clear()
    lcd.putstr("Parking Spot:")
    lcd.move_to(0, 1)
    lcd.putstr(status)

    time.sleep(1)
```





Edit with WPS Office