

|  |  |  |  |
| --- | --- | --- | --- |
| **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  **Domain Name : Internet of things (IOT)**  **Project Title : smart car parking System** | | | |
| **1.** | **Name of the Student (s)** |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | **S.No** | **Name of the Student** | **E-Mail ID** | **Phone No.** | | **1** | Muthuannammalai.SV | Muthusivaraman04@gmail.com | 7904988251 | | **2** | Gowtham.S | Gowthamsel235@gmail.com | 7810045220 | | **3** | Thejesh bhagavanth.G | thejeshbhagavanth@gmail.com | 9626609830 | | 4  5 | Sasinath .D  Sibi.S | sasinathengineer@gmail.com  Sibilali17@gmail.com | 8825782192  9047774669 | | 6 | Hemasundar. | spideysundar2004@gmail.com | 6381543849 | | | |
| **2.** | **Name of the Guide** | : Ms.Suganya |
|  | **Department/ Designation** | : CSE/AP |
|  | **Institutional Address** | : Chettinad College of Engineering  and Tecnology  NH-67, Karur-Trichy Highway,  Puliyur CF, Karur |
|  | **Phone No** | : 6374527207 |

**PROBLEM DEFINITION :**

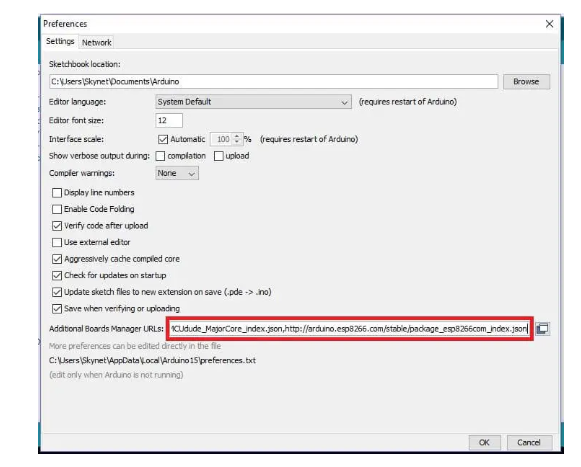
The project involves integrating IoT sensors into public transportation vehicles to monitor ridership, track locations, and predict arrival times. The goal is to provide real-time transit information to the public through a public platform, enhancing the efficiency and quality of public transportation services. This project includes defining objectives, designing the IoT sensor system, developing the real-time transit information platform, and integrating them using IoT technology and Python.

**upload code to ESP8266?**

To upload the given code to ESP8266 we need 2 things:

1. ESP8266 board package (Software).
2. An ESP8266 programmer (Hardware).

**Download ESP8266 board package?**

* Copy this link: <http://arduino.esp8266.com/stable/package_esp8266com_index.json>
* Now open Arduino IDE and click on **File > Preferences**.
* A window will open like this
* 
* Type “ESP8266” on the box as shown and you will get installation option, select the **latest version** and click install.
* Now the IDE will download the necessary packages and this could take more than 5 minute to complete.
* Now go to **Tools > Board > ESP8266 boards > select “Generic ESP8266”.**
* Now, copy the given ESP8266 program code and paste it on to Arduino IDE software.
* Now press compile button (Green tick button). The compilation of code may take more than couple of minutes and be patient. If the compilation failed please check whether have you selected the “Generic ESP8266 Module” in the board option or not.
* After successful compilation of code, now it’s time to upload the code to ESP8266.

**2)** **How to upload code to ESP8266:**

To program ESP8266 we need a programmer module

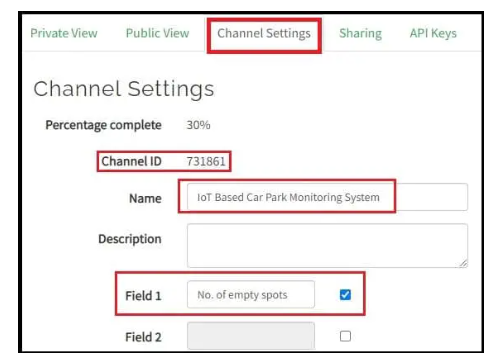
Insert the ESP8266 on the programmer module as shown below and insert it to your PC’s USB port and press upload.

Once the code is successfully uploaded you will see the following info, now you may insert the ESP8266 to the main circuit:

**3) setup your Thingspeak account?**

We are using a (free) cloud service called Thingspeak where we will send parking lot’s data to share it with public.

* First you need to sign up for Thingspeak:
* Enter the credentials it asks for and create a new channel and do the following to your new channel:



* Go to **channel settings** and enter the things as shown above and take note of your channel ID which we need to enter it in the program code.
* Scroll down and **click save** to save the changes.
* Now click on **API keys tab** and you will see your keys as illustrated below. API keys are responsible for writing and reading the data to your Thingspeak account.
* Go to **channel settings** and enter the things as shown above and take note of your channel ID which we need to enter it in the program code.
* Scroll down and **click save** to save the changes.

Now click on **API keys tab** and you will see your keys as illustrated below. API keys are responsible for writing and reading the data to your Thingspeak account.

* Take note of your “write API key” which needs to be entered in the program code and read API key is not used in this project.
* Now go to sharing tab and click on “share channel view with everyone”, this makes your channel visible to those who have the URL of “public view” page.

**Program code for Arduino:**

#include <LiquidCrystal\_I2C.h>

#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11);

LiquidCrystal\_I2C lcd(0x27, 16, 2); // set the LCD address to 0x27 for a 16 chars and 2 line display

const int trig\_1 = 2;

const int echo\_1 = 3;

const int trig\_2 = 4;

const int echo\_2 = 5;

const int trig\_3 = 6;

const int echo\_3 = 7;

float distanceCM\_1 = 0, resultCM\_1 = 0;

float distanceCM\_2 = 0, resultCM\_2 = 0;

float distanceCM\_3 = 0, resultCM\_3 = 0;

long Time\_1, Time\_2, Time\_3;

float car\_1, car\_2, car\_3;

float Dist\_1 = 8.0, Dist\_2 = 8.0, Dist\_3 = 8.0;

int total = 0, timer\_cnt = 0;

void setup()

{

mySerial.begin(115200);

pinMode(trig\_1, OUTPUT);

pinMode(trig\_2, OUTPUT);

pinMode(trig\_3, OUTPUT);

pinMode(echo\_1, INPUT);

pinMode(echo\_2, INPUT);

pinMode(echo\_3, INPUT);

digitalWrite(trig\_1, LOW);

digitalWrite(trig\_2, LOW);

digitalWrite(trig\_3, LOW);

lcd.init();

lcd.backlight();

lcd.setCursor(0, 0);

lcd.print(" IoT CAR PARK");

lcd.setCursor(0, 1);

lcd.print(" MONITOR SYSTEM");

delay(2000);

lcd.clear();

}

void loop()

{

total = 0;

car\_1 = sensor\_1();

car\_2 = sensor\_2();

car\_3 = sensor\_3();

lcd.setCursor(0, 0);

lcd.print("CAR1:");

if (car\_1 <= Dist\_1)

{

lcd.print("OK ");

}

else

{

total += 1;

}

if (car\_1 > Dist\_1) lcd.print("NO ");

lcd.print("CAR2:");

if (car\_2 <= Dist\_2)

{

lcd.print("OK ");

}

else

{

total += 1;

}

if (car\_2 > Dist\_2) lcd.print("NO ");

lcd.setCursor(0, 1);

lcd.print("CAR3:");

if (car\_3 <= Dist\_3)

{

lcd.print("OK ");

}

else

{

total += 1;

}

if (car\_3 > Dist\_3) lcd.print("NO ");

lcd.print("FREE:");

lcd.print(total);

if (timer\_cnt >= 50)

{

mySerial.print('\*');

mySerial.print(total);

mySerial .println('#');

timer\_cnt = 0;

}

timer\_cnt += 1;

delay(200);

}

float sensor\_1(void)

{

digitalWrite(trig\_1, HIGH);

delayMicroseconds(10);

digitalWrite(trig\_1, LOW);

Time\_1 = pulseIn(echo\_1, HIGH);

distanceCM\_1 = Time\_1 \* 0.034;

return resultCM\_1 = distanceCM\_1 / 2;

}

float sensor\_2(void)

{

digitalWrite(trig\_2, HIGH);

delayMicroseconds(10);

digitalWrite(trig\_2, LOW);

Time\_2 = pulseIn(echo\_2, HIGH);

distanceCM\_2 = Time\_2 \* 0.034;

return resultCM\_2 = distanceCM\_2 / 2;

}

float sensor\_3(void)

{

digitalWrite(trig\_3, HIGH);

delayMicroseconds(10);

digitalWrite(trig\_3, LOW);

Time\_3 = pulseIn(echo\_3, HIGH);

distanceCM\_3 = Time\_3 \* 0.034;

return resultCM\_3 = distanceCM\_3 / 2;

}

**Program code for ESP8266:**

#include "ThingSpeak.h"

#include <ESP8266WiFi.h>

//------- WI-FI details ----------//

char ssid[] = "SSID"; //SSID here

char pass[] = "PASSWORD"; // Password here

//--------------------------------//

//----------- Channel details ----------------//

unsigned long Channel\_ID =123456; // Your Channel ID

const char \* myWriteAPIKey = "ACBDE12345"; //Your write API key

//-------------------------------------------//

const int Field\_Number\_1 = 1;

String value = "";

int value\_1 = 0;

WiFiClient client;

void setup()

{

Serial.begin(115200);

WiFi.mode(WIFI\_STA);

ThingSpeak.begin(client);

internet();

}

void loop()

{

internet();

if (Serial.available() > 0)

{

delay(100);

while (Serial.available() > 0)

{

value = Serial.readString();

if (value[0] == '\*')

{

if (value[2] == '#')

{

value\_1 = value[1] - 0x30;

}

}

}

}

upload();

}

void internet()

{

if (WiFi.status() != WL\_CONNECTED)

{

while (WiFi.status() != WL\_CONNECTED)

{

WiFi.begin(ssid, pass);

delay(5000);

}

}

}

void upload()

{

ThingSpeak.writeField(Channel\_ID, Field\_Number\_1, value\_1, myWriteAPIKey);

delay(15000);

value = "";

}