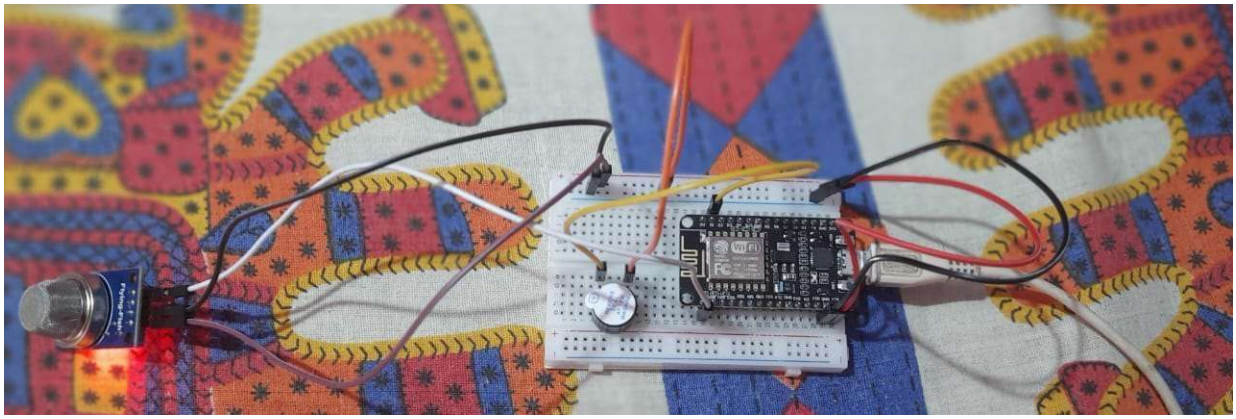
GURU NANAK INSTITUTE OF TECHNOLOGY

TOPIC: AIR QUALITY CHECKING WITH ANDROID APPLICATION



COMPUTER SCIENCE & ENGINEERING

2ND YEAR, 3RD SEMESTER,

PRIYANSHU BANERJEE

ACKNOWLEDGEMENT

It gives me a great sense of pleasure to present the report of the Project Work undertaken on the topic **"Air Quality checking with Android Application"**.

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful. We take this opportunity to express our deep gratitude towards our project mentor, **[SK SUMAN SIR]** for giving such valuable suggestions, guidance and encouragement during the development of this project work. A deep thanks to my team members because without their constant support I cannot make such a wonderful project.

Last but not the least we are grateful to all the faculty members of **Ardent Computech Pvt. Ltd.** for their support.

.

ABSTRACT

The objective of this project is to check the air quality efficiently and accurately using an Nodemcu ESP8266 board that will be monitored by an android OS smartphone.

As the world is getting back to normal, the familiar problem regarding air pollution affecting our health and daily life is faced by us again. It is not convenient to always check the national air quality monitoring network. But making sure that the quality of air is not degrading beyond a certain level is also important.

Our project solves this problem. The IOT based air quality checker monitors the quality of the air with the help of a smartphone.

CONTENTS:

INTRODUCTION:

In today's day and age with the rapid growth of automobiles on the road, increasing development of various industries and low afforestation rates , the degrading quality of air we breathe in has become a concerning factor for all. Average person spends an estimated 90% of their time indoors so that poor indoor air quality (IAQ) poses a substantial risk to public health. Poor air quality may cause increased short-term health problems such as fatigue and nausea as well as chronic respiratory diseases, heart disease, and lung cancer. It has become especially important that we check the quality of the air we breathe in. However, it is not always possible to check the national air monitoring network.

Our project brings a solution to this problem. We have put together a very accessible and cost friendly IOT based air quality checker. We are using android smartphone to monitor the quality of the air which makes it user friendly.

COMPONENTS USED:

1. NODEMCU ESP8266(ESP-12E)

2. MQ-135 AIR QUALITY SENSOR

3. HALF SIZE BREADBOARD

4. JUMPER WIRES

5. MINI 5V BUZZER

6. ANDROID DEVICE

DESCRIPTION:

A. ABOUT NODEMCU ESP8266

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE".[18] This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

B. FEATURES AND SPECIFICATIONS OF NODEMCU ESP8266

a) Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106

b) Operating Voltage: 3.3V

c) Input Voltage: 7-12V

d) Digital I/O Pins (DIO): 16

e) Analog Input Pins (ADC): 1

f) UARTs: 1

g) SPIs: 1

h) I2Cs: 1

i) Flash Memory: 4 MB

j) SRAM: 64 KB

k) Clock Speed: 80 MHz

l) USB-TTL based on CP2102 is included onboard, Enabling Plug n Play

m) PCB Antenna

n) Small Sized module to fit smartly inside your IoT projects

C. NODEMCU HARDWARE PART:

The NodeMCU ESP8266 development board comes with the ESP-12E module containing the ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. NodeMCU has 128 KB RAM and 4MB of Flash memory to store data and programs. Its high processing power with in-built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IoT projects.

NodeMCU can be powered using a Micro USB jack and VIN pin (External Supply Pin). It su Power Requirement

As the operating voltage range of ESP8266 is 3V to 3.6V, the board comes with a LDO voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA, which should be more than enough when ESP8266 pulls as much as 80mA during RF transmissions. The output of the regulator is also broken out to one of the sides of the board and labeled as 3V3. This pin can be used to supply power to external components

Switches & Indicators

- RST - Reset the ESP8266 chip
- FLASH - Download new programs
- Blue LED - User Programmable

The board also has a LED indicator which is user programmable and is connected to the D0 pin of the board.

Serial Communication

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serial and allows your computer to program and communicate with the ESP8266 chip.

- CP2102 USB-to-UART converter
- 4.5 Mbps communication speed

Flow Control support

If you have an older version of CP2102 driver installed on your PC, we recommend upgrading now.

D. PINS IN NODEMCU ESP8266

Power Pins: There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components.

GND: is a ground pin of ESP8266 NodeMCU development board.

I2C Pins: are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins: ESP8266 NodeMCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button

programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel: The NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins: ESP8266 NodeMCU has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signals, so it is usually used for printing logs.

SPI Pins: ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

4 timing modes of the SPI format transfer

Up to 80 MHz and the divided clocks of 80 MHz

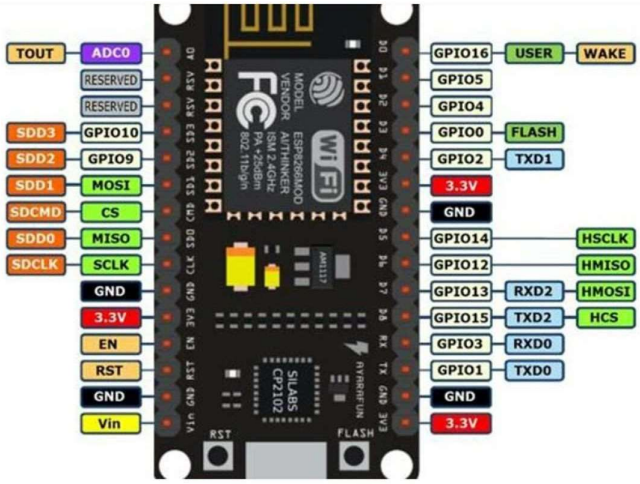
Up to 64-Byte FIFO

SDIO Pins: ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

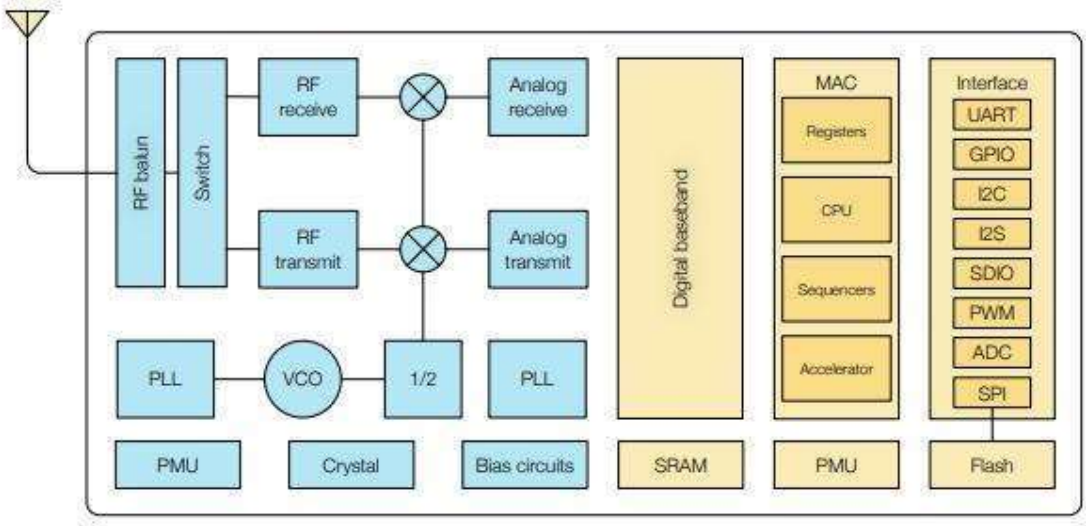
PWM Pins: The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 µs to 10000 µs, i.e., between 100 Hz and 1 kHz.

Control Pins: are used to control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

E. NODEMCU PINOUT



F. ESP8266 architecture



On the picture above, can you identify:

- The CPU (Central Processing Unit)
- The memory SRAM (Static Random Access Memory)

The ESP8266 uses a 32bit processor with 16 bit instructions. It is Harvard architecture which mostly means that instruction memory and data memory are completely separate.

The ESP8266 has on die program Read-Only Memory (ROM) which includes some library code and a first stage boot loader. All the rest of the code must be stored in external Serial flash memory (provides only serial access to the data - rather than addressing individual bytes, the user reads or writes large contiguous groups of bytes in the address space serially).Depending on your ESP8266, the amount of available flash memory can vary.

As any other microcontroller, ESP8266 has a set of GPIO pins (General Purpose Input(Output pins) that we can use to "control" external sensors.Our ESP8266 has 17 GPIO pins but only 11 can be used (among 17 pins, 6 are used for communication with the on-board flash memory chip). It also has an analog input (to convert a voltage level into a digital value that can be stored and processed in the ESP8266).It also has a WIFI communication to connect your ESP8266 to your WIFI network, connect to the internet, host a web server, let your smartphone connect to it, etc. Another advantage of an ESP8266 is that it can be programmed as any other microcontroller and especially any Arduino

G. SOFTWARE USED:

ARDUINO IDE: The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java/C++. It originated from the IDE for the languages *Processing* and *Wiring*. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple *one-click* mechanisms to compile and upload programs to a compatible board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic

functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the board by a loader program in the board's firmware.

BLYNK

Blynk is **an IoT platform for iOS or Android smartphones that is used to control Arduino, Raspberry Pi and NodeMCU via the Internet**. This application is used to create a graphical interface or human machine interface (HMI) by compiling and providing the appropriate address on the available widgets.

Blynk App - **allows to you create amazing interfaces for your projects using various widgets we provide**. Blynk Server - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally.
Now let's talking about how **BLYNK** is working in our project:

STEP 1: Go to https://blynk.io/ from browser and create an account in blynk.

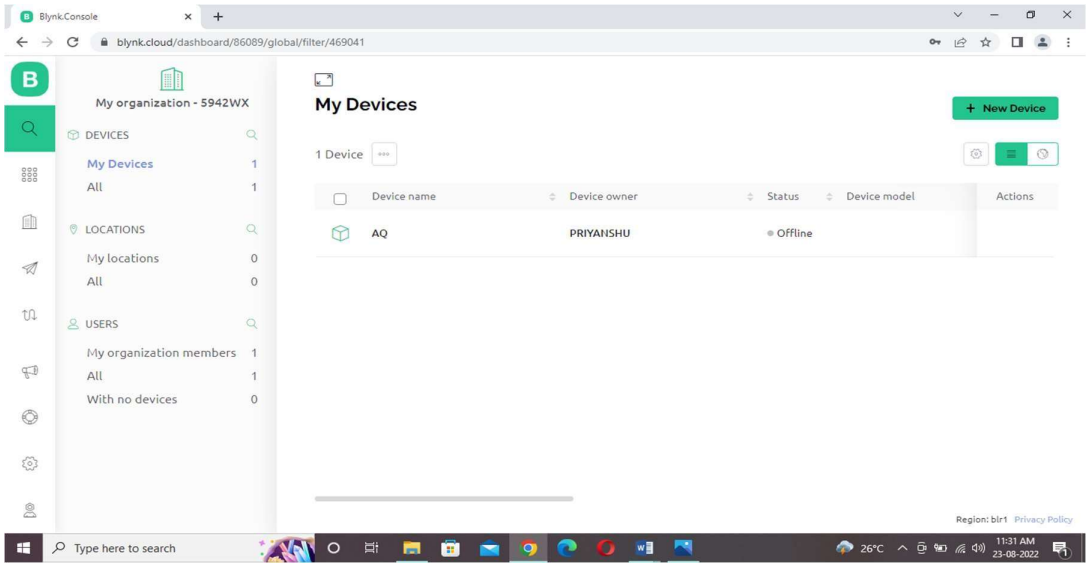
STEP 2: Then create a template in blynk with a suitable name.

STEP 3: After did the appropriate template setting and create a device(i.e do the needfull changes in datastream and in web dasboard) and saved it an api key will generate for your device.

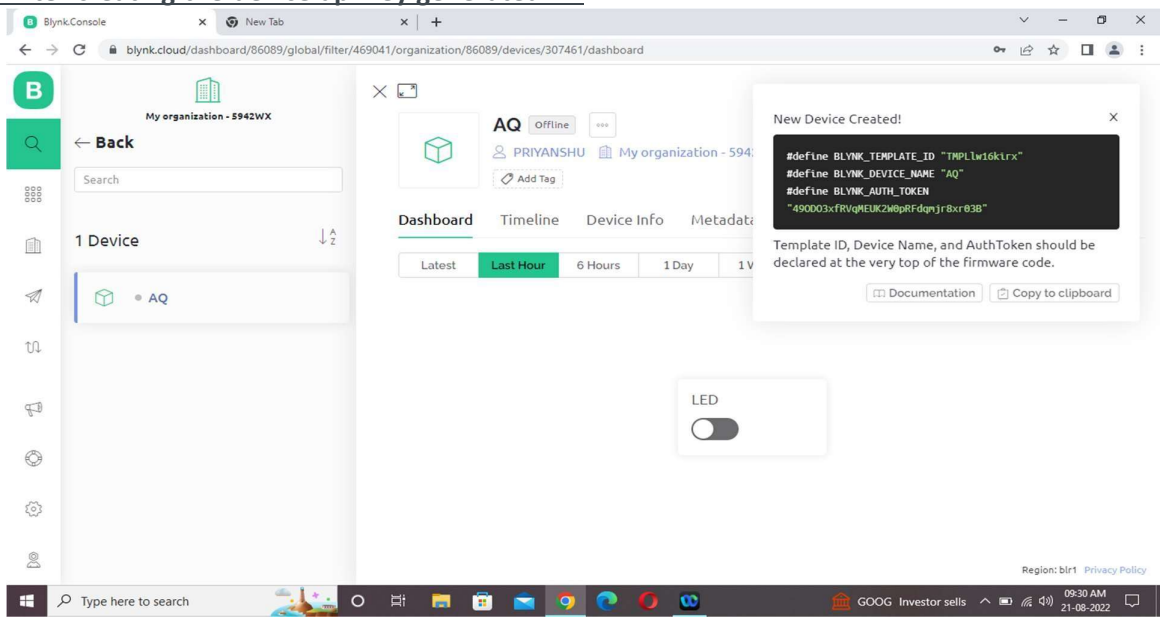
STEP 4: Now copy the api key and paste it in the code in Arduino IDE.

STEP 5: Now if we run the code the respective data measured by sensor will upload to the blynk again and again with short duration and you can check the data from anywhere, anytime(from android app also).

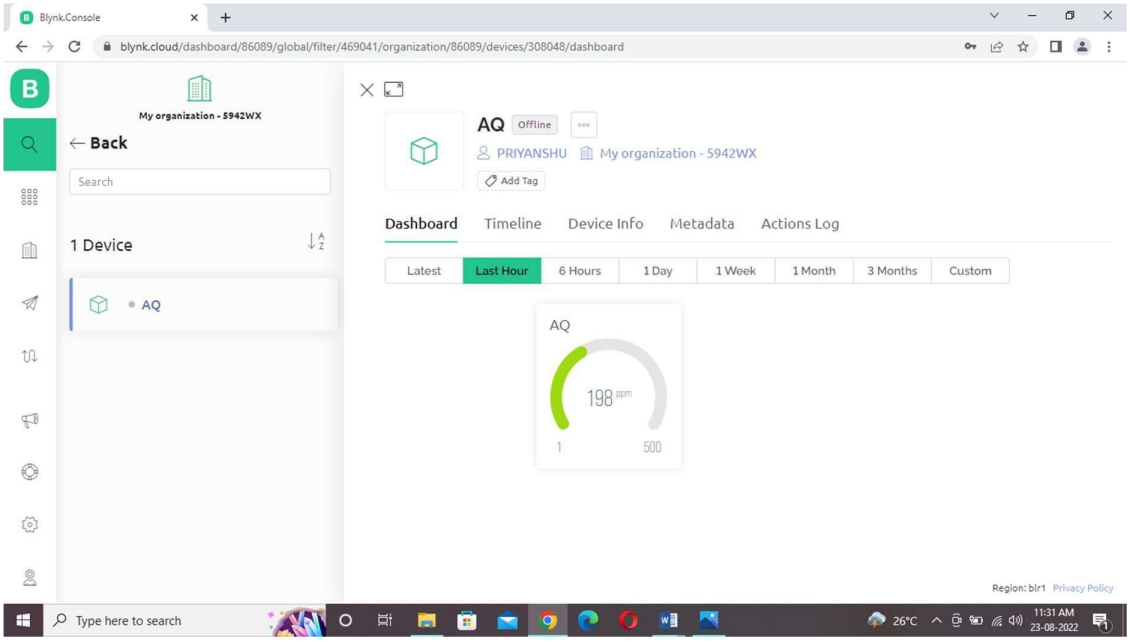
Device is created....



After creating the device api key generated……



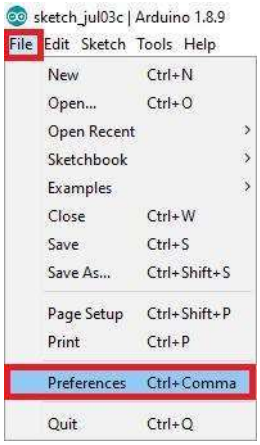
Data showing in our Air Gauge…..



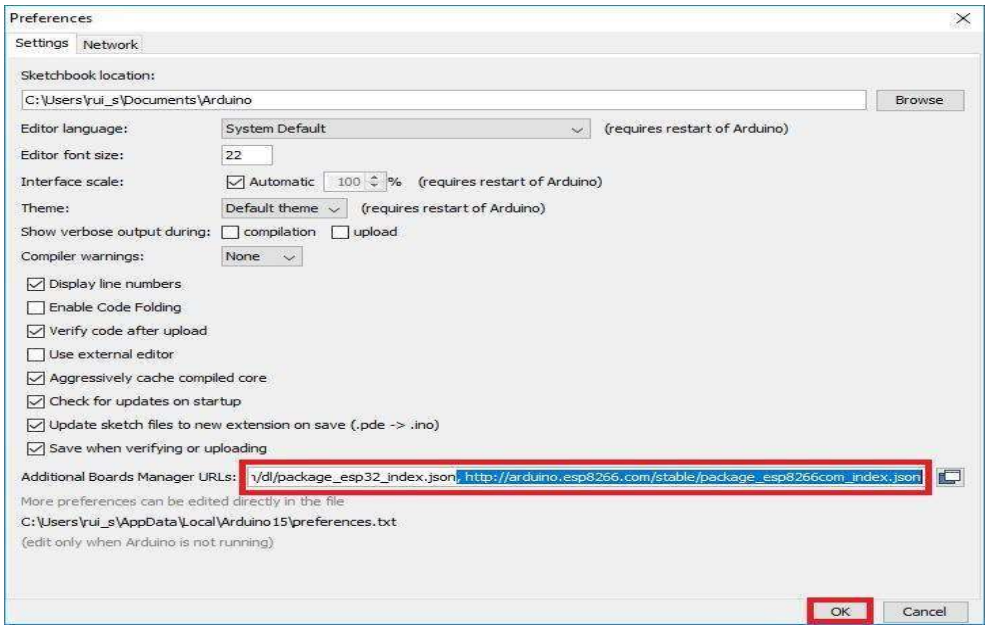
Install ESP8266 Add-on in Arduino IDE

To install the ESP8266 board in your Arduino IDE, follow these next instructions:

- In your Arduino IDE, go to **File**> **Preferences**



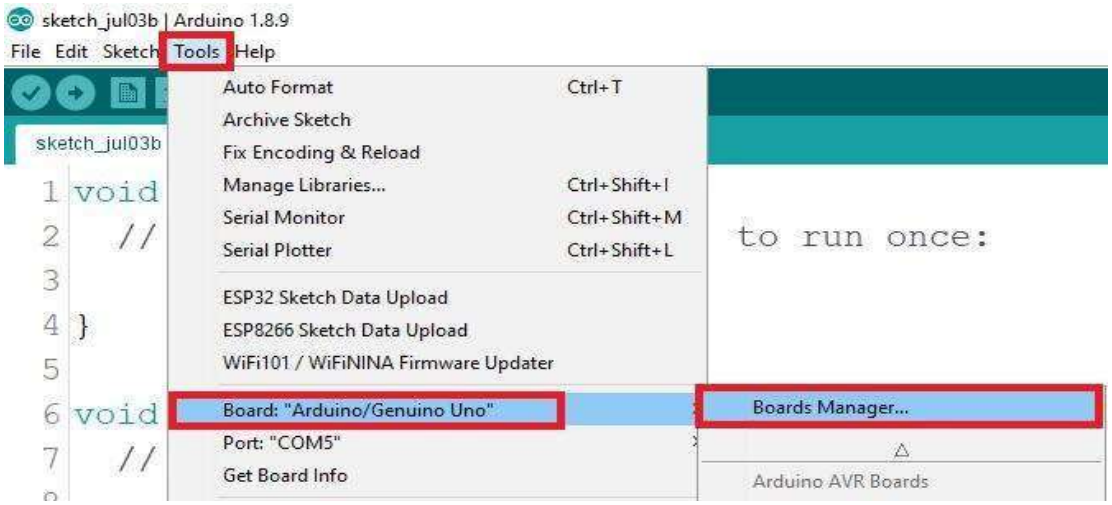
- Enter http://arduino.esp8266.com/stable/package_esp8266com_index.json into the "Additional Boards Manager URLs" field as shown in the figure below. Then, click the "OK" button:



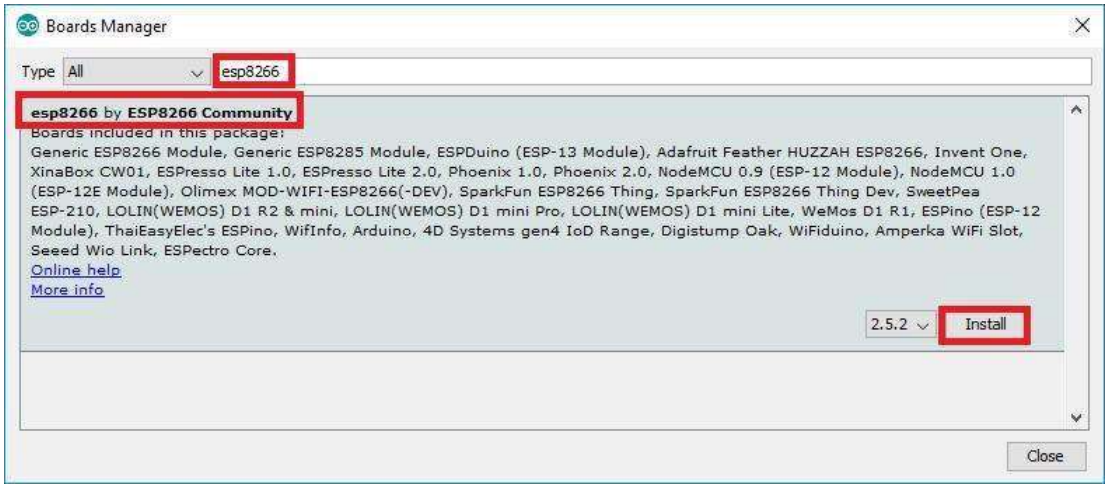
Note: if you already have the ESP32 boards URL, you can separate the URLs with a comma as follows:

https://dl.espressif.com/dl/package_esp32_index.json,
http://arduino.esp8266.com/stable/package_esp8266com_index.json

- Open the Boards Manager. Go to Tools > Board > Boards Manager…



Search for ESP8266 and press install button for the **"ESP8266 by ESP8266 Community":**



- **That's it. It should be installed after a few seconds.**

SKETCH: A program written with the Arduino IDE is called a *sketch*. ***Sketches*** are saved on the development computer as text files with the file extension *.ino*. Arduino Software (IDE) pre-1.0 saved sketches with the extension *.pde*.

A minimal Arduino C/C++ program consist of only two functions:

- ***setup()*:** This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- ***loop()*:** After *setup()* has been called, function *loop()* is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

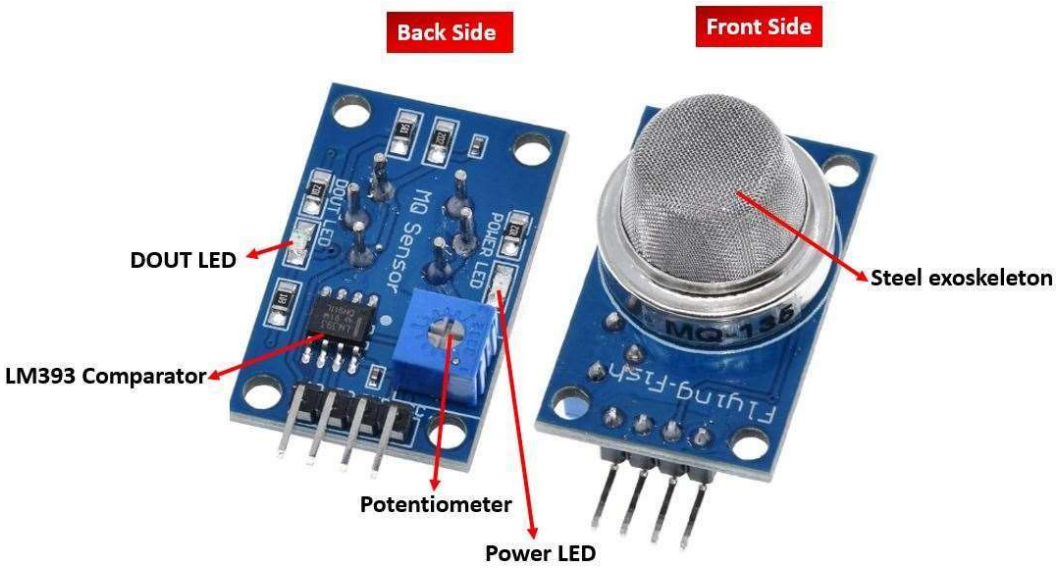
Mq135 AIR QUALITY SENSOR: In this project we use **MQ135 AIR QUALITY** sensor. Here is a brief discussion about it-

ABOUT MQ-135 AIR QUALITY SENSOR

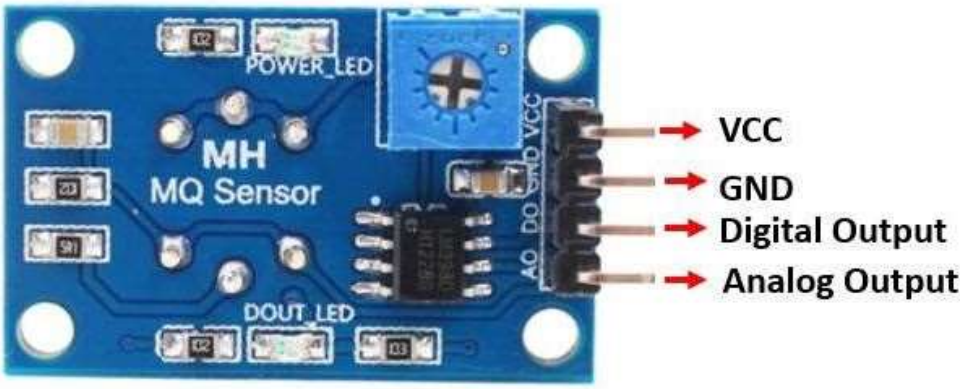
MQ135 Gas Sensor is an air quality sensor for detecting a wide range of gases, including NH3, NOx, alcohol, benzene, smoke and CO2. Ideal for use in office or factory. MQ135 gas sensor has high sensitivity to Ammonia, Sulfide and Benzene steam, also sensitive to smoke and other harmful gases. It is with low cost & particularly suitable for Air quality monitoring application.

FEATURES OF MQ-135 AIR QUALITY SENSOR

- High sensitivity to Ammonia, Sulfide and Benzene.
- Stable and Long Life.
- Detection Range: 10 - 300 ppm NH3, 10 - 1000 ppm Benzene, 10 - 300 ppm Alcohol.
- Heater Voltage: 5.0V.
- Dimensions: 18mm Diameter, 17mm High excluding pins, Pins - 6mm High.



Pinout



The MQ-135 sensor module consists of four pins namely VCC, GND, DO, and DO. The table below gives a brief description of them.

Pin	Description
VCC	Positive power supply pin that powers up the sensor module.
GND	Reference potential pin.
AO	Analog output pin. It generates a signal proportional to the concentration of gas vapors coming in contact with the sensor.
DO	Digital Output pin. It also produces a digital signal whose limit can be set using the in-built potentiometer.

How does it work?

Through connecting leads, the sensing element is exposed to current. The gases that come close to the sensing element are ionized and absorbed by the sensing element as a result of this current, which is known as heating current. This affects the resistance of the sensing element, hence changing the value of the current leaving it.

When no gas is present, digital output is 1 and analog output gives 1023 max value. When gas is present, digital output is 0 and analog output is much less than 1023.

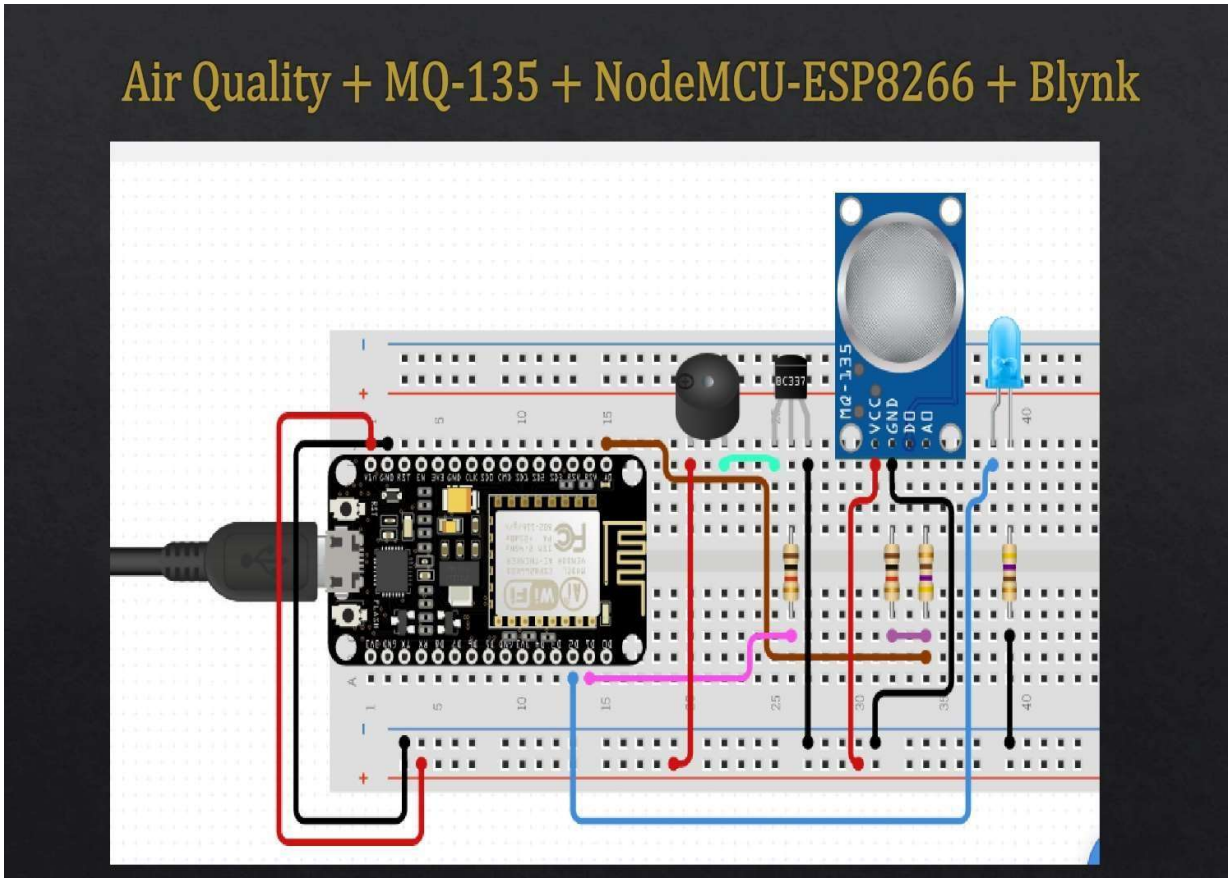
Using potentiometer on chip we can control the turning OFF point of digital pin at some value of analog pin. The sensor needs a load-resistor at the output to ground. Its value could be from 2kOhm to 47kOhm. The lower the value, the less sensitive is the sensor. The higher the value, the less accurate is sensor for

higher concentrations of gas. If only one specific gas is measured, the load-resistor can be calibrated by applying a known concentration of that gas. If the sensor is used to measure any gas (like in a air quality detector) the load-resistor could be set for a value of about 1V output with clean air. Choosing a good value for the load-resistor is only valid after the burn-in time

The following steps have done to **calibrate and implement** it in this project:

- The VCC pin of the MQ135 Gas Sensor is connected with the Nodemcu Module 3.3 volts pin, the AOUT pin is connected with the A0 pin of the Nodemcu Module and the ground of the Gas Sensor is connected with the ground of the Nodemcu ESP8266 Wifi Module
- Then upload the proper code to the nodemcu and run it .
- Then we start getting the value in blynk app .
- When the threshold value is crossed acc to condition the buzzer start ringing.
- Again when the value drops below the threshold value the buzzer stops

CIRCUIT DIAGRAM:



CODE:

```
#define BLYNK_TEMPLATE_ID "TMPLlw16kirx"
#define BLYNK_DEVICE_NAME "priyanshu"
#define BLYNK_AUTH_TOKEN "ZVB78goF70eK00F-NRK-4PMigm-6mto5"
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "SSID OF WIFI";
char pass[] = "PASSWORD OF WIFI";

const int BUZ = 5;
int mq135 = A0;
int data = 0;

BlynkTimer timer;

void sensorData()
{
  data = analogRead(mq135);
  Blynk.virtualWrite(V0, data);
```

```
  if (data > 200)
  {
     tone(BUZ,1000);
     delay(1000);
}
 else
  {
   digitalWrite(BUZ,LOW);
  }
}


void setup()
{
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(1000L, sensorData);
  pinMode(mq135, INPUT); // Set BUZ
  pinMode(BUZ, OUTPUT); // Set BUZ
}
void loop()
{
  Blynk.run();
  timer.run();
}
```

OTHER APPLICATIONS OF IOT

1. Smart Homes

One of the best and the most practical applications of IoT, smart homes really take both, convenience and home security, to the next level. Though there are different levels at which IoT is applied for smart homes, the best is the one that blends intelligent utility systems and entertainment together. For instance, your electricity meter with an IoT device giving you insights into your everyday water usage, your set-top box that allows you to record shows from remote, Automatic Illumination Systems, Advanced Locking Systems, and Connected Surveillance Systems all fit into this concept of smart homes. As IoT evolves, we can be sure that most of the devices will become smarter, enabling enhanced home security.

2. Self-driven Cars

We've seen a lot about self-driven cars. Google tried it out, Tesla tested it, and even Uber came up with a version of self-driven cars that it later shelved. Since it's human lives on the roads that we're dealing with, we need to ensure the technology has all that it takes to ensure better safety for the passenger and those on the roads.

The cars use several sensors and embedded systems connected to the Cloud and the internet to keep generating data and sending them to the Cloud for informed decision-making through Machine Learning. Though it will take a few more years for the technology to evolve completely and for countries to amend laws and policies, what we're witnessing right now is one of the best applications of IoT.

3. Farming

Farming is one sector that will benefit the most from the Internet of Things. With so many developments happening on tools farmers can use for agriculture, the future is sure promising. Tools are being developed for Drip Irrigation, understanding crop patterns, Water Distribution, drones for Farm Surveillance, and more. These will allow farmers to come up with a more productive yield and take care of the concerns better.

4. Wearables

Wearables remain a hot topic in the market, even today. These devices serve a wide range of purposes ranging from medical, and wellness to fitness. Of all the IoT startups, Jawbone, a wearables maker, is second to none in terms of funding.

5. Smart Grids

One of the many useful IoT examples, a smart grid, is a holistic solution that applies an extensive range of Information Technology resources that enable existing and new gridlines to reduce electricity waste and cost. A future smart grid improves the efficiency, reliability, and economics of electricity.

6. Traffic Management

Car traffic management in large cities can be greatly improved with the help of the Internet of Things (IoT). The Internet of Things helps us stay informed and improves traffic monitoring by allowing us to use our mobile phones as sensors to collect and share data from our vehicles through apps like Waze or Google Maps. This feeds and improves the data on the various routes to the same destination, distance, and estimated arrival time.

Analysis of traffic patterns over a long period is another **IoT application**. It provides an idea of what might happen during peak hours. Commuters will be better prepared to avoid traffic and delays by being made aware of possible alternatives

7. Water/ Waste Management

Many cities are adopting water recycling using water treatment units. Using an **IoT application**, you can see how much wastewater is being produced, how much is being consumed in a specific area, and how waste production is changing over time.

8.Telehealth

Telehealth, or Telemedicine, hasn't completely flourished yet. Nonetheless, it has great future potential. IoT Examples of Telemedicine include the digital communication of Medical Imaging, Remote Medical Diagnosis & Evaluations, Video Consultations with Specialists, etc.

CONCLUSION

"Air Quality Checking with Android Application " is the topic of our project and while doing this project we gain a lot of knowledge about IOT and how to fetch data from Nord mcu to android application. We use ESP8266, MQ135 sensor, breadboard ,buzzer and a android device to complete this project. WIFI is used as the communication channel between android phone and the microcontroller. This paper proposes a low cost, secure way to check air quality of your surroundings. . The approach discussed in the paper is novel and has achieved the target of checking air quality using wifi technology to form a connection between microcontroller and android device.The system design and architecture were discussed, and the prototype presents the basic level of air quality checking using android app in home.Finally we want to say that we enjoy doing this project and we came to know about new things about IOT which were unknown to us.We know that IOT is going to be game changer in field of technology in future and we feel this project has helped us to increase our curiosity to know more about IOT.

REFERENCE

Websites:

1. Wikipedia
2 https://microcontrollerslab.com/interfacing-mq-135-gas-sensor-arduino/
3 https://blynk.cloud/
4 www.nordmcu.com
5 https://www.electroniclinic.com/iot-smoke-detector-using-mq135-gas-sensor