# CHAPTER 1 INTRODUCTION

**CHAPTER 1**

**INTRODUCTION**

Air pollution is the most effective pollution in the world. It is increasing every day. According to research in the year of 2021 India rank 5th place and more than 1.6 million deaths occurred due to air pollution. High level increase in air pollution caused risk of respiratory infections, dry throat, headache, chest pain, heart disease, asthma and lung cancer etc. We have implemented Smart Sensor System For Smog Monitoring that detects various poisonous gases such as Carbon Dioxide, Carbon Monoxide, Methane, LPG, Mercury, Sulphur Dioxide, Smoke, and Propane using Arduino UNO microchip ATmega328P with THINGSPEAK SERVER is used for Internet Of Things projects where data is been processed and analysed in real time , frequently used for prototyping and proof-of-concept systems. We have used sensors like MQ2, MQ135, MQ7 and DHT11. Air quality will show in parts per million. When the sensor value crossing more than the ppm value buzzer starts to beep.

* 1. **OBJECTIVE**
* To detect poisonous gases.
* To take precautions against smog pollution caused.
* To initiate green environment
* To display the accuracies of the harmful gases emitted.
  1. **SCOPE OF WORK**
* When there is emission of gases like carbon dioxide and smoke it is been detected by an sensor called mq135,mq2 and mq7
* Once the smoke increases an buzzer will start beeping
* By doing cultivation of plants and tress which will reduce the smog.
  1. **JUSTIFICATION**
* This is a real time IOT project which will detect the smog in the society.
* Using this IOT based on smart sensor system for smog monitoring, people can get the awareness or the alert of the smog emitted in the society and can take precautions .

**CHAPTER 2**

**LITERATURE SURVEY**

**CHAPTER 2**

**LITERATURE SURVEY**

Air Pollution caused in the areas of using fossil fuels for energy combustion releases a lot of sulphur dioxide into the atmosphere. Air pollution is caused by burning fuels, vehicle smoke emissions, which harm the environment. Agriculture is causing major effect while pesticides and fertilizers process. Factories are the primary source of producing carbon monoxide, chemicals and smoke. These are most important things which is lowering the quality of the air. A recent research, 45% of people lives will be shorten by 8years by air pollution. More than 500 million people would get affect living in India. Government of NCAP decided to cut-off pollution in 105 cities. There are many air pollution monitoring devices such as AQSync, TONGDY, Prana air pocket, Airatom, Smiledrive, Airveda and Temtop etc. Real time Smog monitoring machine which is used to measure the quality of the air is it “FRESH AIR” or “POLLUATED AIR”. Cost effective machine and easy for installation which can be installed or setup in various locations of the cities. IOT based open source microcontroller board ARDUINO UNO 8-bit microcontroller ATmega328P contains great features such as more instructions every cycle, more frequently, built-in control, flexibility & usability, adjustable pins, rapid start, additional flash memory, low voltage demand ,play and plug. USB connection and extra storage. Used 4 different sensors which is used to generate the data and send to the cloud server thingspeak. Thingspeak is a best platform which is used to store the sensors data with the interface of API key over an internet. After singed into the thingspeak account individual channels is been created for each different sensors , best user interface such as charts , graphs and collaborating app with web services. Once after the setup connections of channels are created and automatically data are visualised.

**CHAPTER 3**

**SYSTEM REQUIREMENTS SPECIFICATIONS**

**Chapter 3**

**SYSTEM REQUIREMENTS SPECIFICATIONS**

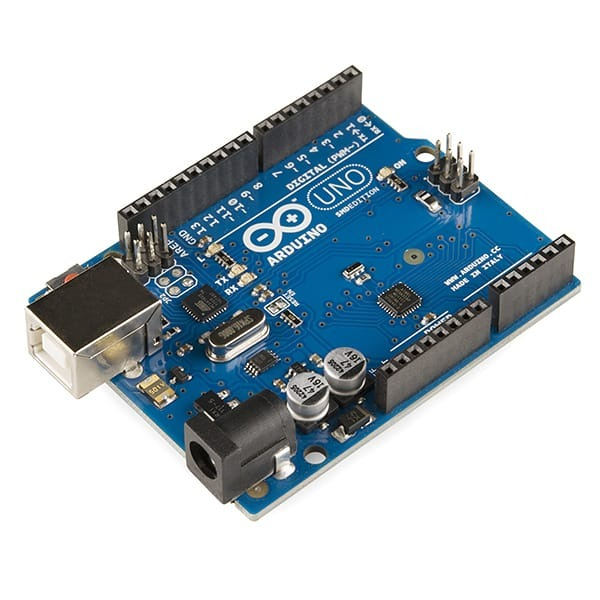
**3.1 HARDWARE REQUIREMENT**

1. Arduino uno
2. MQ135 gas sensor
3. MQ7 gas sensor
4. MQ2 gas sensor
5. DHT11 sensor
6. Wi-Fi module ESP8266
7. 16X2 LCD
8. Breadboard
9. 10K potentiometer
10. 220 ohm resistor
11. Buzzer
12. Header pins
13. USB cable
14. Jumperwire

**3.2 SOFTWARE REQUIREMENT**

1. Arduino IDE
2. THINGSPEAK

**1. ARDUINO UNO**



**Figure 3.1 Arduino uno**

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc and initially released in 2010. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[1] The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is similar to the Arduino Nano and Leonardo. Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, 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.

**2. MQ135 GAS SENSOR**

****

The gas sensor module consists of a steel exoskeleton under which a sensing element is housed. This sensing element is subjected to current through connecting leads. This current is known as heating current through it, the gases coming close to the sensing element get ionized and are absorbed by the sensing element. This changes the resistance of the sensing element which alters the value of the current going out of it.

**PIN CONFIGURATION**

* A0 Analog output
* D0 Digital output
* GND Ground
* Vcc Supply (5V)

**SPECIFICATIONS**

* Wide detecting scope
* Fast response and High sensitivity
* Stable and long-life Simple drive circuit
* Used in air quality control equipment for buildings/offices, is suitable for detecting
* of NH3, NOx, alcohol, Benzene, smoke, CO2, etc.
* Size: 35mm x 22mm x 23mm (length x width x height)
* Working voltage: DC 5 V
* Signal output instruction.
* Dual signal output (analog output, and high/low digital output)
* 0 ~ 4.2V analog output voltage, the higher the concentration the higher the voltage.

**3. MQ7 GAS SENSOR**

****

Sensitive material of MQ-7 gas sensor is SnO2, which with lower conductivity in clean air. It make detection by method of cycle high and low temperature, and detect CO when low temperature (heated by 1.5V). The sensor’s conductivity is more higher along with the gas concentration rising. When high temperature (heated by 5.0V), it cleans the other gases adsorbed under low temperature. Please use simple electrocircuit, Convert change of conductivity to correspond output signal of gas concentration. MQ-7 gas sensor has high sensitity to Carbon Monoxide. The sensor could be used to detect different gases contains CO, it is with low cost and suitable for different application.

**PIN CONFIGURATION**

* A0 Analog output
* D0 Digital output
* GND Ground
* Vcc Supply (5V)

**SPECIFICATIONS**

* The analog output voltage, the higher the concentration the higher the voltage.
* The carbon monoxide detection with better sensitivity.
* There are four screw holes for easy positioning.
* With a long service life and reliable stability.
* Rapid response and recovery characteristics.

**4. MQ2 GAS SENSOR**

****

MQ2 is one of the commonly used gas sensors in MQ sensor series. It is a Metal Oxide Semiconductor (MOS) type Gas Sensor also known as Chemiresistors as the detection is based upon change of resistance of the sensing material when the Gas comes in contact with the material. Using a simple voltage divider network, concentrations of gas can be detected.

MQ2 Gas sensor works on 5V DC and draws around 800mW. It can detect LPG, Smoke, Alcohol, Propane, Hydrogen, Methane and Carbon Monoxide concentrations anywhere from 200 to 10000ppm.

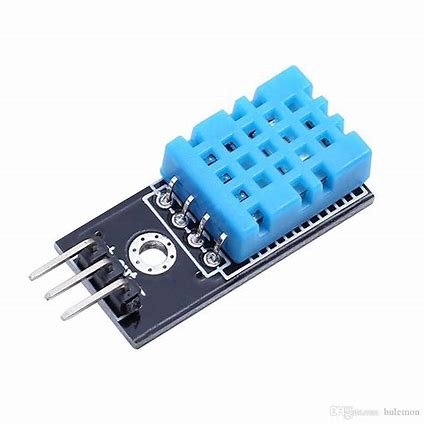
**PIN CONFIGURATION**

* A0 Analog output
* D0 Digital output
* GND Ground
* Vcc Supply (5V)

**SPECIFICATIONS**

* Preheat duration 20 seconds
* Can be used as a Digital or analog sensor
* The Sensitivity of Digital pin can be varied using the potentiometer

**5. DHT11 SENSOR**

****

DHT11 is a low-cost digital sensor for sensing temperature and humidity. This sensor can be easily interfaced with any micro-controller such as Arduino, Raspberry Pi etc… to measure humidity and temperature instantaneously. DHT11 humidity and temperature sensor is available as a sensor and as a module. The difference between this sensor and module is the pull-up resistor and a power-on LED. DHT11 is a relative humidity sensor. To measure the surrounding air this sensor uses a thermistor and a capacitive humidity sensor.

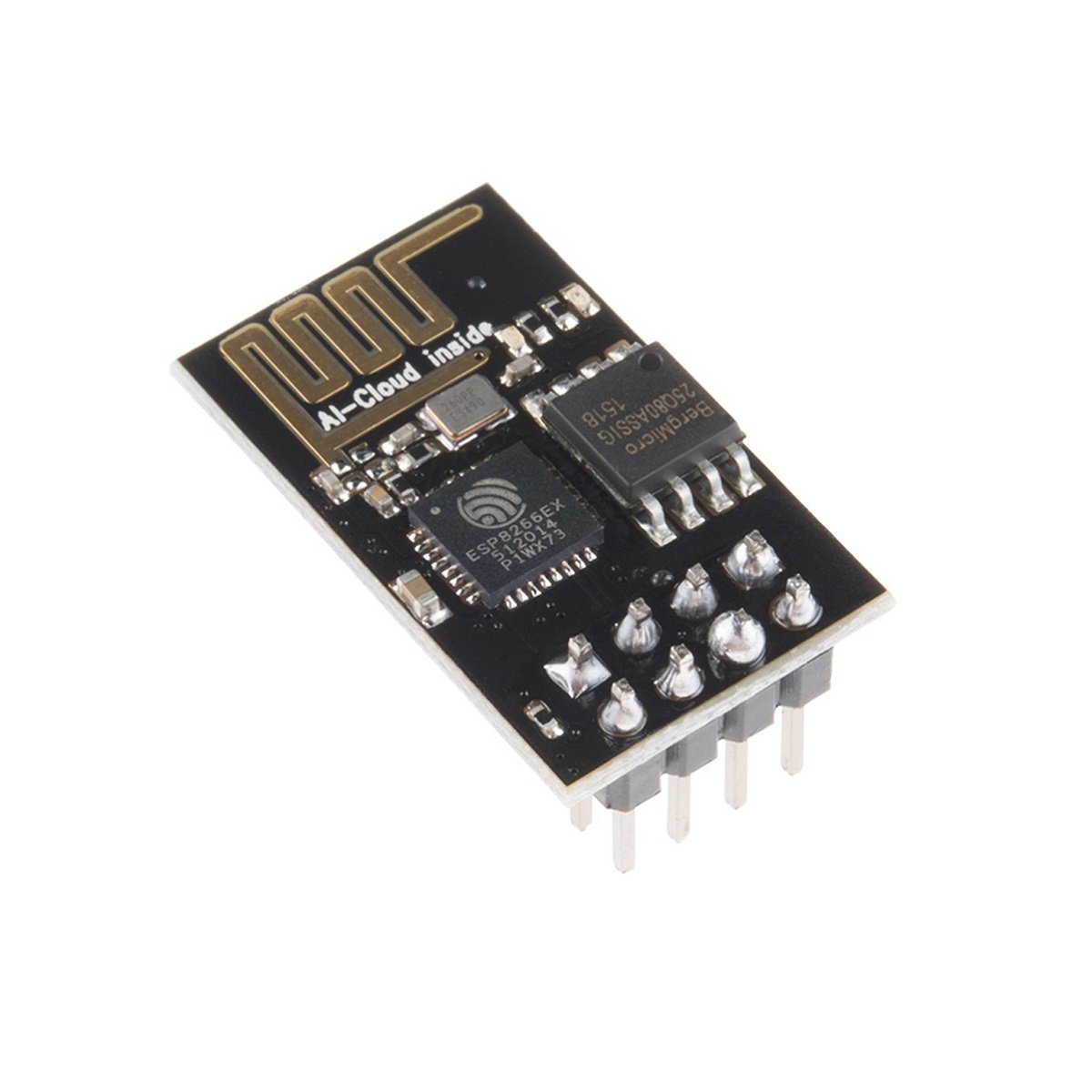
**PIN CONFIGURATION**

* Vcc - Power supply 3.5V to 5.5V
* Data - Outputs both Temperature and Humidity through serial Data
* NC - No Connection and hence not used
* Ground - Connected to the ground of the circuit

**SPECIFICATIONS**

* Ultra low cost
* to 5V power and I/O
* 2.5mA max current use during conversion (while requesting data)
* Good for 20-80% humidity readings with 5% accuracy
* Good for 0-50°C temperature readings ±2°C accuracy
* No more than 1 Hz sampling rate (once every second)
* Body size 15.5mm x 12mm x 5.5mm
* pins with 0.1" spacing

**6. WI-FI MODULE ESP8266**

****

The ESP8266 WiFi Module is a self-contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your Wi-Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Arduino device and get about as much WiFi-ability as a WiFi Shield offers (and that's just out of the box)!

**PIN CONFIGURATION**

* RX - Serial Receiver Pin
* Vcc - Power Pin (+3.3 V; can handle up to 3.6 V)
* GPIO 0 - General-purpose I/O No. 0
* RST - Reset
* CH\_PD - Chip power-down
* GPIO 1 - General-purpose I/O No. 2
* TX - Serial Transmitter Pin
* GND - Ground

**SPECIFICATIONS**

* 802.11 b/g/n
* Wi-Fi Direct (P2P), soft-AP
* Integrated TCP/IP protocol stack
* Integrated TR switch, balun, LNA, power amplifier and matching network
* Integrated PLLs, regulators, DCXO and power management units
* +19.5dBm output power in 802.11b mode
* Power down leakage current of <10uA
* 4MB Flash Memory
* Integrated low power 32-bit CPU could be used as application processor

**7. LCD 16x2**

****

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

**Working Principle**

* The basic working principle of LCD is passing the light from layer to layer through modules. These modules will vibrate & line up their position on 90o that permits the polarized sheet to allow the light to pass through it.
* These molecules are accountable for viewing the data on every pixel. Every pixel utilizes the method of absorbing light to illustrate the digit. To display the value, the position of molecules must be changed to the angle of light.
* So this light deflection will make the human eye notice the data that will be the ingredient wherever the light gets absorbed. Here, this data will supply to the molecules & will be there till they get changed
* At present, LCDs are used frequently in CD/DVD players, digital watches, computers, etc. In screen industries, LCDs have replaced the CRTs (Cathode Ray Tubes) because these displays use more power as compared to LCD, heavier & larger.
* The displays of LCDs are thinner as compared to CRTs. As compared to LED screens, LCD has less power consumption because it functions on the fundamental principle of blocking light instead of dissipating.

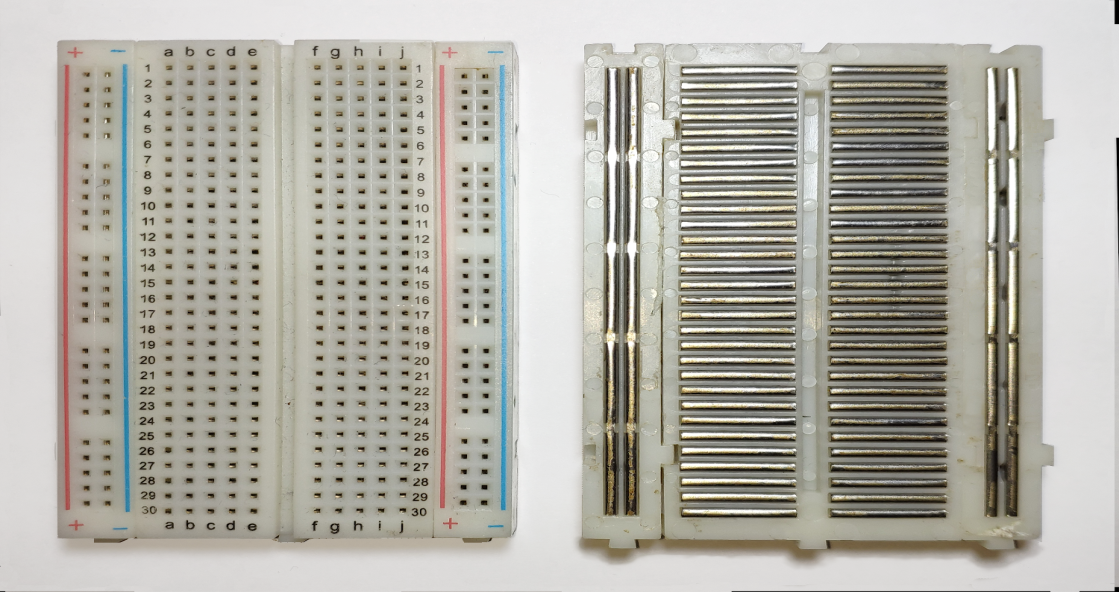
**PIN CONFIGURATION**

* Pin1 (Ground): This pin connects the ground terminal.
* Pin2 (+5 Volt): This pin provides a +5V supply to the LCD
* Pin3 (VE): This pin selects the contrast of the LCD.
* Pin4 (Register Select): This pin is used to connect a data pin of an MCU & gets either 1 or 0. Here, data mode = 0 and command mode =1.
* Pin5 (Read & Write): This pin is used to read/write data.
* Pin6 (Enable): This enables the pin must be high to perform the Read/Write procedure. This pin is connected to the data pin of the microcontroller to be held high constantly.
* Pin7 (Data Pin): The data pins are from 0-7 which are connected through the microcontroller for data transmission. The LCD module can also work on the 4-bit mode through working on pins 1, 2, 3 & other pins are free.
* Pin8 – Data Pin 1
* Pin9 – Data Pin 2
* Pin10 – Data Pin 3
* Pin11 – Data Pin 4
* Pin12 – Data Pin 5
* Pin13 – Data Pin 6
* Pin14 – Data Pin 7
* Pin15 (LED Positive): This is a +Ve terminal of the backlight LED of the display & it is connected to +5V to activate the LED backlight.
* Pin16 (LED Negative): This is a -Ve terminal of a backlight LED of the display & it is connected to the GND terminal to activate the LED backlight.

**SPECIFICATIONS**

* The operating voltage of this LCD is 4.7V-5.3V
* It includes two rows where each row can produce 16-characters.
* The utilization of current is 1mA with no backlight
* Every character can be built with a 5×8 pixel box
* The alphanumeric LCDs alphabets & numbers
* Is display can work on two modes like 4-bit & 8-bit
* These are obtainable in Blue & Green Backlight
* It displays a few custom generated characters

**8. BREADBOARD**

****

Breadboards are temporary work boards for electronic circuits. The general shape of breadboards is rectangle. Compatible with most breadboards, 24-gauge wire is used to connect circuits; solid wire, not stranded. Sometimes, kits may be available with various colours of fixed lengths to specifically fit breadboards. These are a nice convenience

A breadboard (sometimes called a plug block) is used for building temporary circuits. It is useful to designers because it allows components to be removed and replaced easily.

A breadboard consists of plastic block holding a matrix of electrical sockets of a size suitable for gripping thin connecting wire, component wires or the pins of transistors and integrated circuits (ICs). The sockets are connected inside the board, usually in rows of five sockets. A row of five connected sockets is filled in at the top right of the figure. The rows are 2.54 mm apart and the sockets spaced 2.54 mm apart in the rows, which is the correct spacing for the pins of ICs and many other components.

**TERMINAL STRIPS**

The main areas, to hold most of the electronic components. In the middle of a terminal strip of a breadboard, one typically finds a notch running in parallel to the long side. The notch is to mark the centerline of the terminal strip and provides limited airflow to DIP ICs straddling the centerline . The clips on the right and left of the notch are each connected in a radial way; typically five clips in a row on each side of the notch are electrically connected. The five columns on the left of the notch are often marked as A, B, C, D, and E.

**BUS STRIPS**

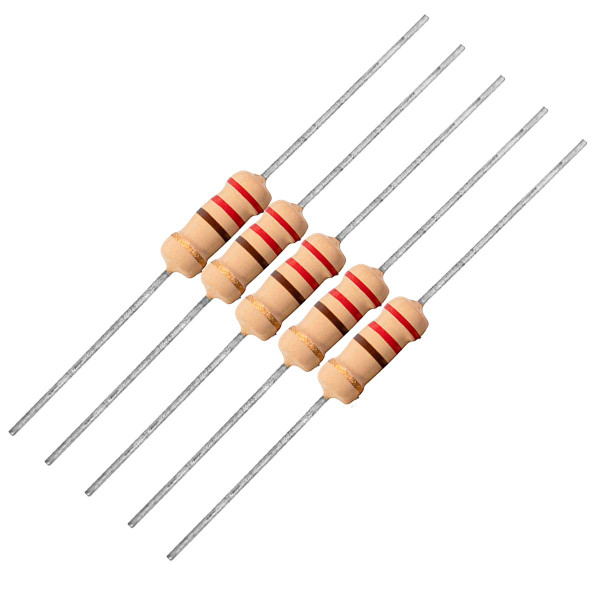
To provide power to the electronic components. A bus strip usually contains two columns: one for ground and one for a supply voltage. However, some breadboards only provide a single-column power distribution bus strip on each long side. Typically the row intended for a supply voltage is marked in red, while the row for ground is marked in blue or black. Some manufacturers connect all terminals in a column. Others just connect groups of, for example, 25 consecutive terminals in a column. The latter design provides a circuit designer with some more control over crosstalk (inductively coupled noise) on the power supply bus.

**9. 10K POTENTIOMETER**

****

A 10k potentiometer is an electronic component that can be used to control the flow of electricity through a circuit. A 10k pot has three terminals at one end - known as the wiper, outside leads, and ground

**10. 220 OM RESISTOR**

****

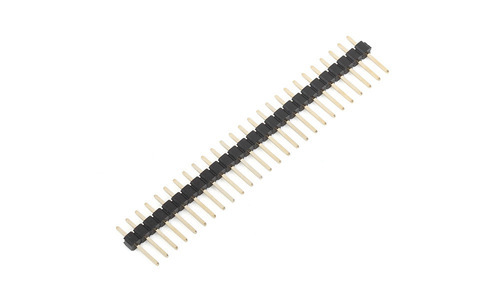
A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. Resistors act to reduce current flow, and, at the same time, act to lower voltage levels within circuits. In electronic circuits, resistors are used to limit current flow, to adjust signal levels, bias active elements, and terminate transmission lines.

**11. BUZZER**

****

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers, train and confirmation of user input such as a mouse click or keystroke.

**12. HEADER PINS**

****

A pin header (or simply header) is a form of electrical connector. A male pin header consists of one or more rows of metal pins molded into a plastic base, often 2.54 mm (0.1 in) apart, though available in many spacings. Male pin headers are cost-effective due to their simplicity. The female counterparts are sometimes known as female socket headers, though there are numerous naming variations of male and female connectors. Historically, headers have sometimes been called "Berg connectors”, but headers are manufactured by many companies.

**13. USB CABLE**

****

This cable is used to interface any of the Arduino board with your computer, you can also connect your USB printer, scanner, and more to your computer. These cables Transmits data at high speeds with the error-free, high-performance transmission.

**14. JUMPERWIRES**



A jump wire is an electrical wire, or group of them in a cable, with a connector or pin at each end which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.

**1. ARDUINO IDE**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

**EDIT**

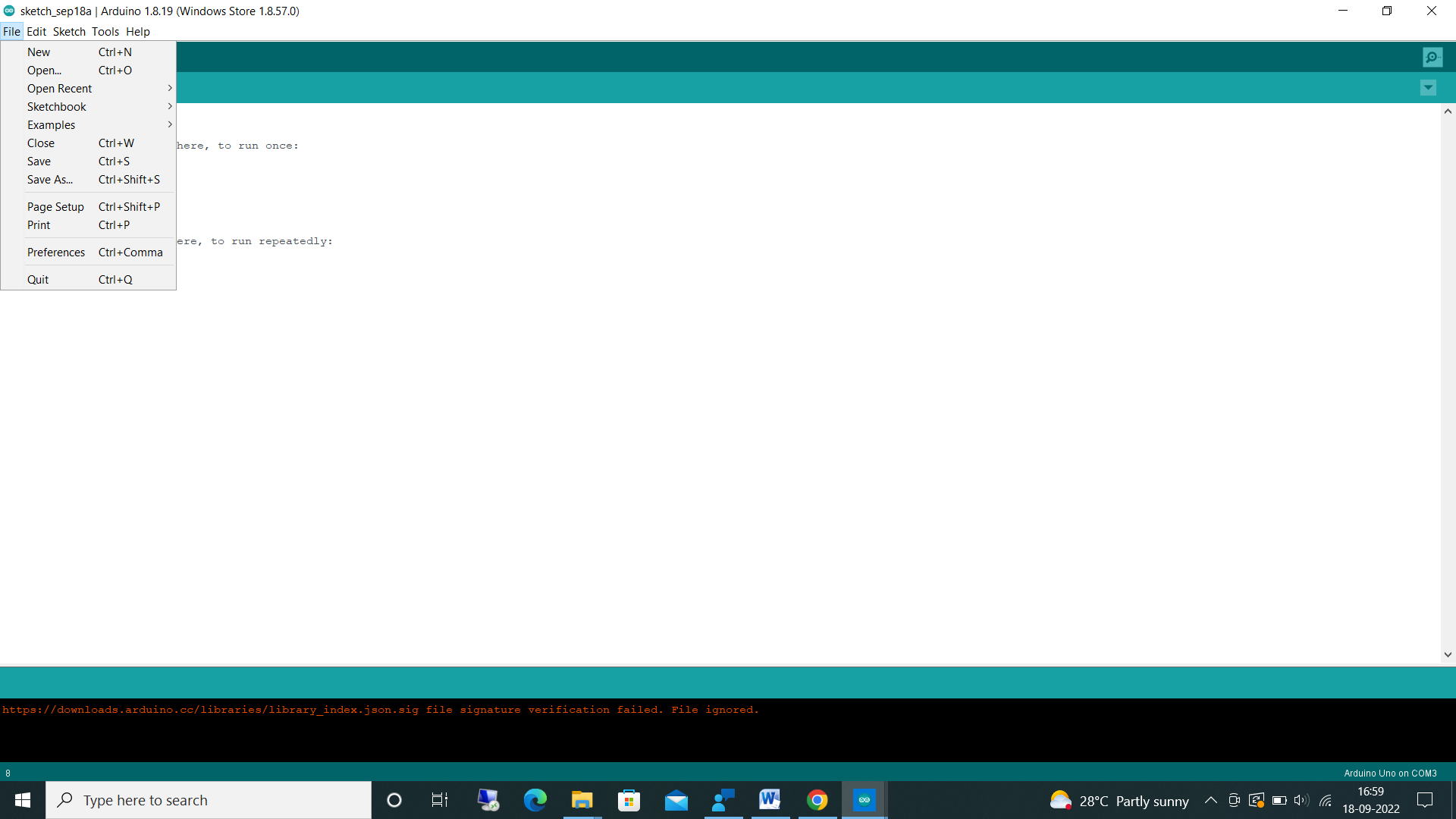
Undo/Redo Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo. Cut Removes the selected text from the editor and places it into the clipboard. Copy Duplicates the selected text in the editor and places it into the clipboard.

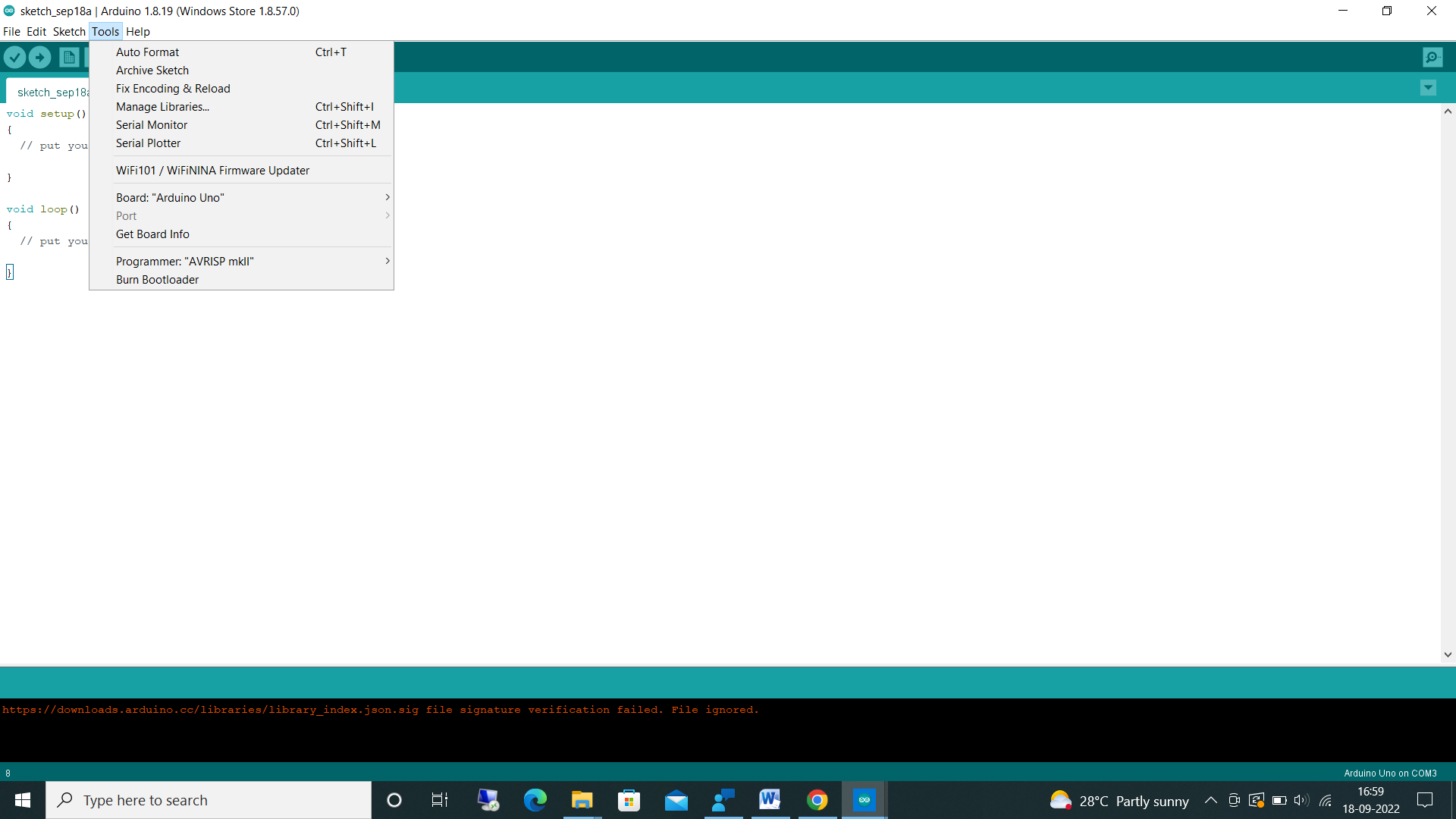
Copy for Forum Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax colouring .Copy as HTML Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages. Paste puts the contents of the clipboard at the cursor position, in the editor. Select All Selects and highlights the whole content of the editor. Comment/Uncomment Puts or removes the comment marker at the beginning of each selected line. Increase/Decrease Indent Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning. Find Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options. Find Next Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position. Find Previous Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

**SKETCH**

Verify/Compile Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area. Upload Compiles and loads the binary file onto the configured board through the configured Port. Upload Using Programmer This will overwrite the boot loader on the board; you will need to use Tools > Burn Boot loader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so Tools -> Burn Boot loader command must be executed. Export Compiled Binary Saves a .hex file that may be kept as archive or sent to the board using other tools. Show Sketch Folder Opens the current sketch folder. Include Library Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files. Adds a supplemental file to the sketch (it will be copied from its current location). The file is saved to the data subfolder of the sketch, which is intended for assets such as documentation. The contents of the data folder are not compiled, so they do not become part of the sketch program.



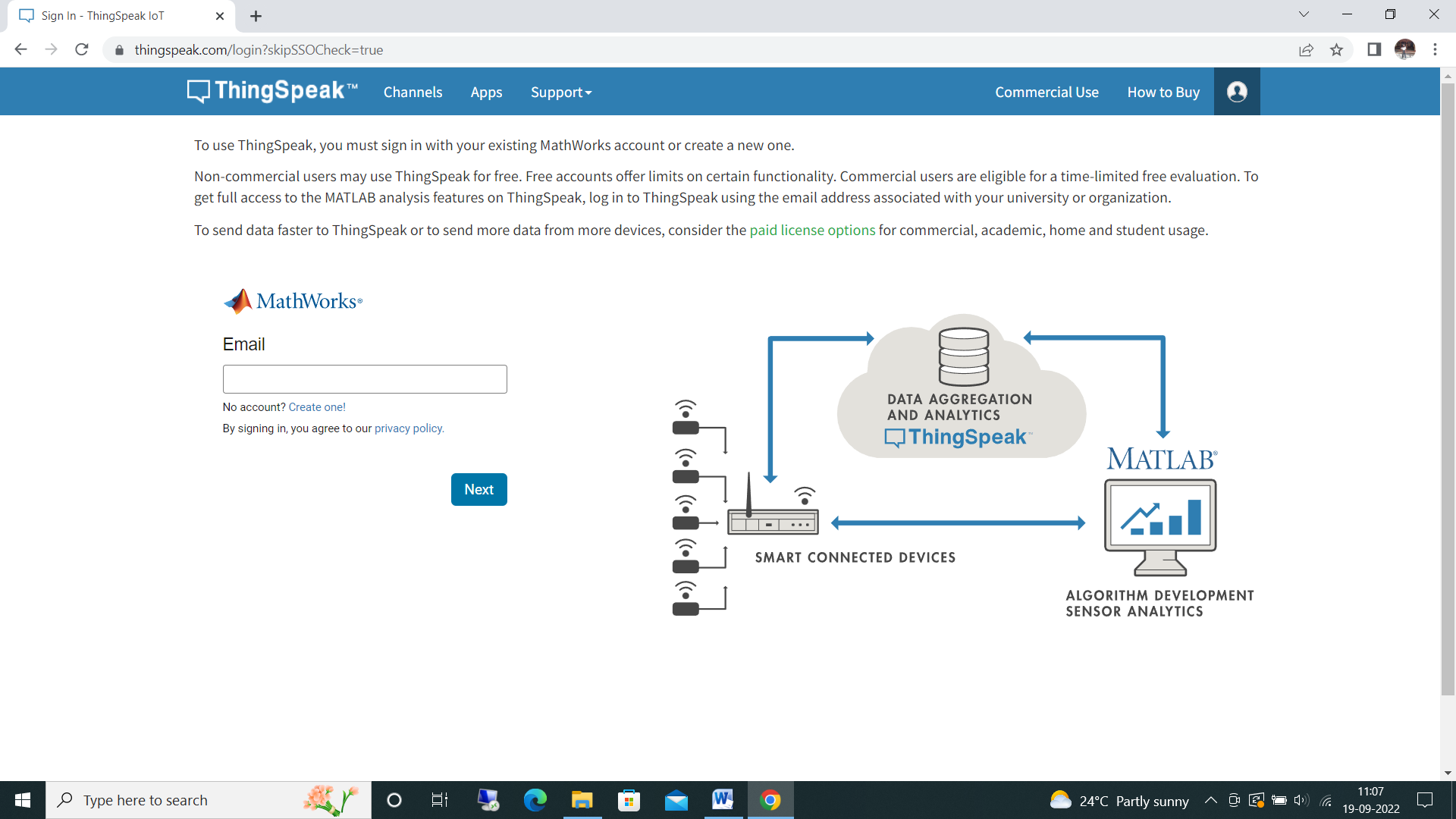
 

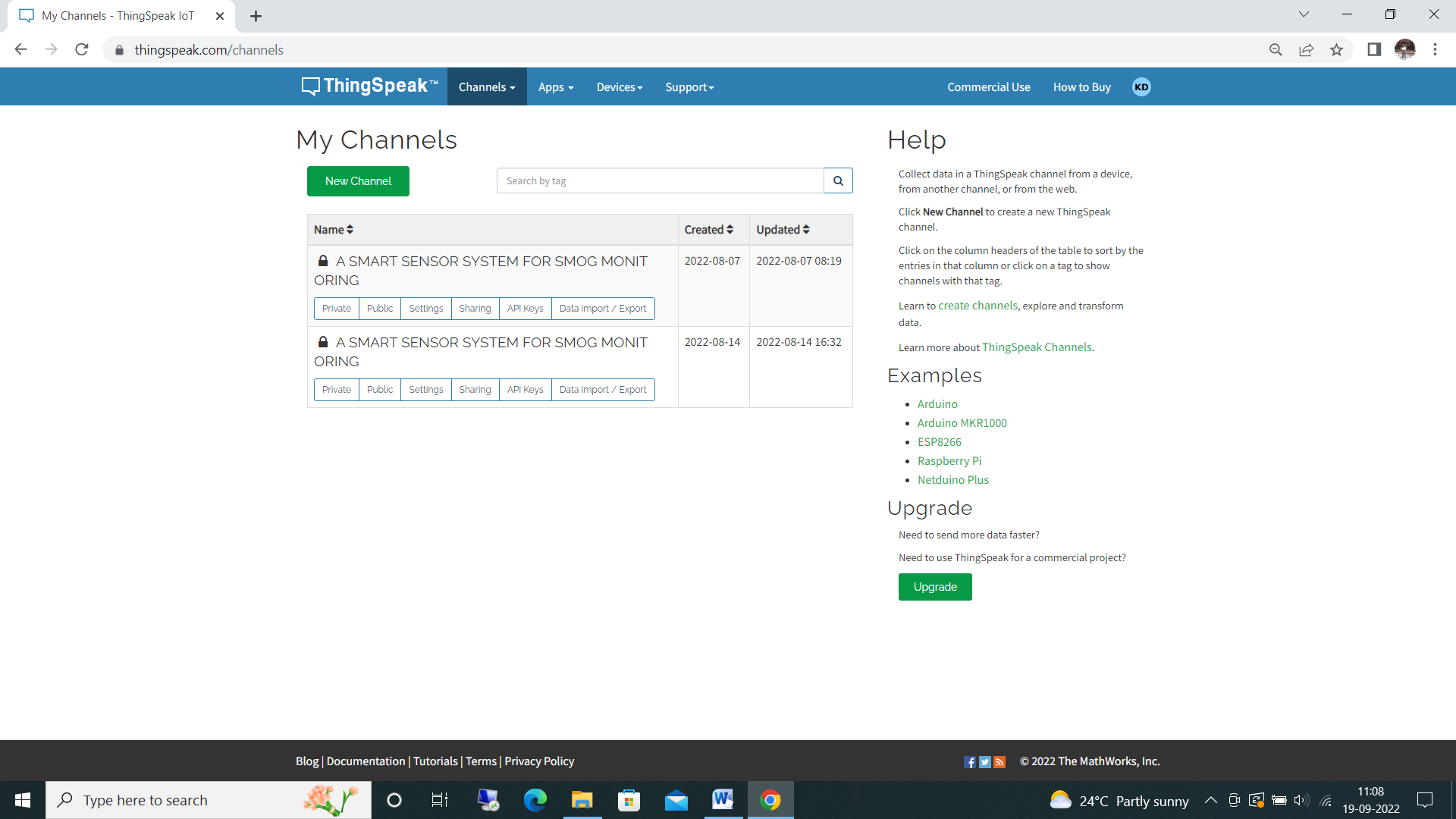
**2. THINGSPEAK**

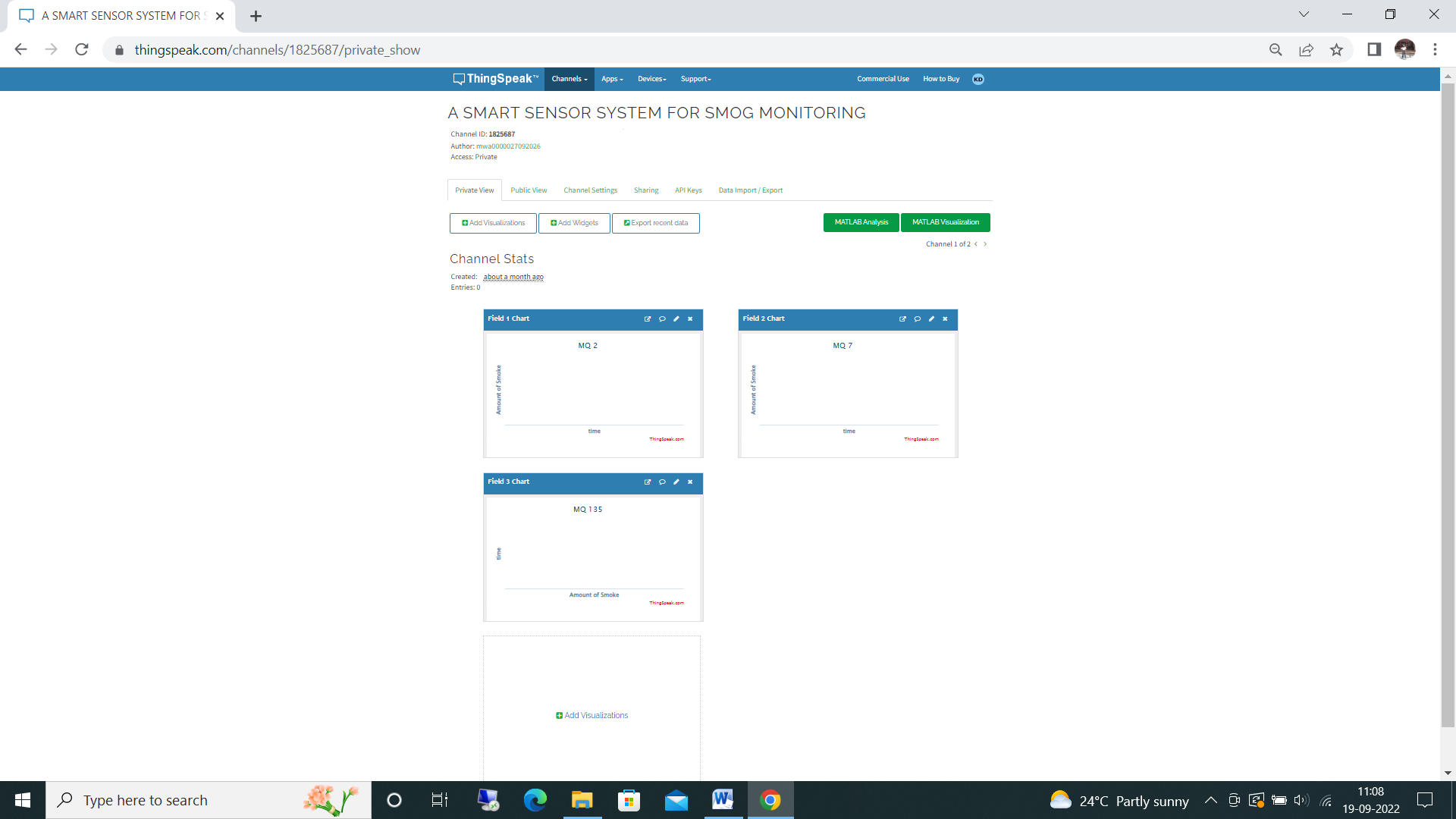
We can upload sensor data to the cloud using the IoT cloud platform called ThingSpeak. With MATLAB or other tools, we can also perform data analysis and data visualisation, as well as create our own apps. MathWorks runs the ThingSpeak platform. We must either log in to your current MathWorks Account or establish a new MathWorks Account in order to register for ThingSpeak. We can use THINGSPEAK for small iot projects, ThingSpeak is open source .We gather and store sensor data in the cloud with ThingSpeak's Web Service (REST API) and create Internet of Things apps. It is compatible with MATLAB, Raspberry Pi, and Arduino But that utilises a REST API and HTTP; we can compile all types of programming languages.

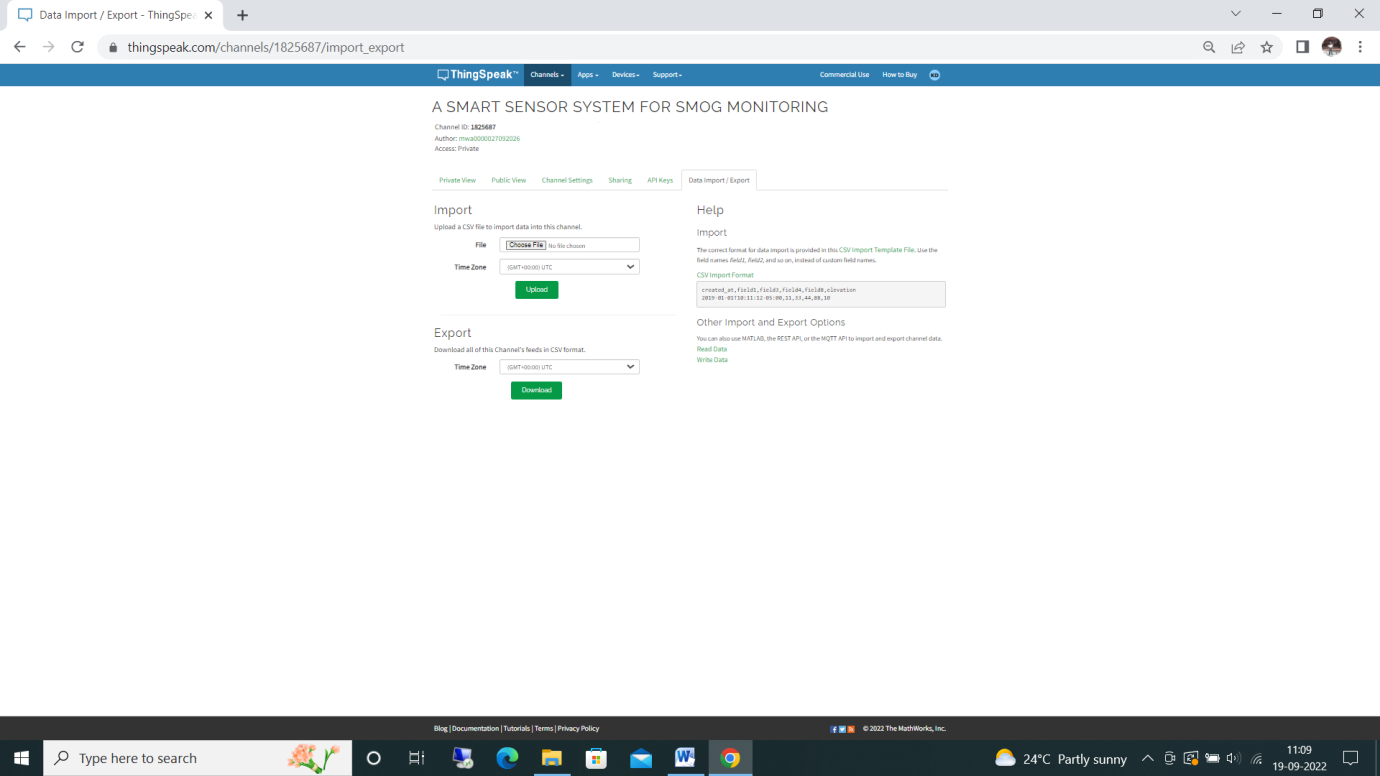
ThingSpeak is an IoT analytics service that allows you to aggregate, visualize, and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak. With the ability to execute MATLAB® code in ThingSpeak, you can perform online analysis and process data as it comes in. ThingSpeak is often used for prototyping and proof-of-concept IoT systems that require analytics. You can send data from any internet-connected device directly to ThingSpeak using a Rest API or MQTT. In addition, cloud-to-cloud integrations with The Things Network, Senet, the Libelium Meshlium gateway, and Particle.io enable sensor data to reach ThingSpeak over LoRaWAN and 4G/3G cellular connections. With ThingSpeak, you can store and analyse data in the cloud without configuring web servers, and you can create sophisticated event-based email alerts that trigger based on data coming in from your connected devices.

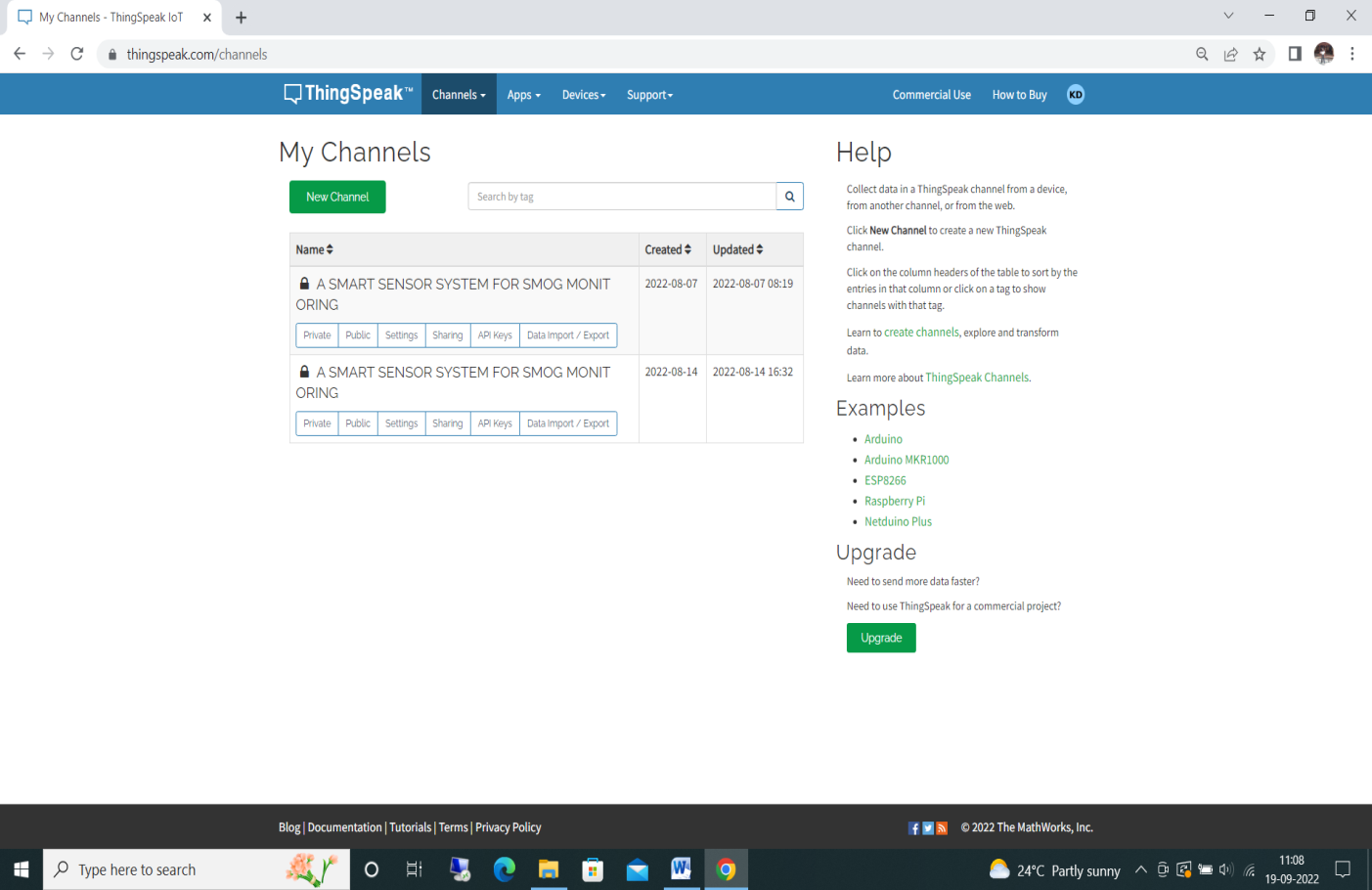












**CHAPTER 4**

**SYSTEM ANALYSIS**

**CHAPTER-4**

**SYSTEM ANAYSIS**

**4.1 PROPOSED SYSTEM**

Smog monitoring is an mini electronic monitoring device where it monitors all kind of poisonous gases such as carbon dioxide, carbon monoxide, methane, LPG, propane, sulphur, ammonia, butane, nitrogen dioxide. Cost effective and Portable device were we can setup in commercial and industrial purpose. Must stop burning of wastage, put layers of filters in the factory funnels, use public transport rather than using individual vehicles etc. Arduino Uno board is an best choice for iot projects. Could have taken one gas sensor for monitoring but we have used 4 different sensors because each sensor as different feature, ppm value and percentage varies. Data is stored and monitored through serial monitor in arduino ide and THINGSPEAK cloud. Coding in arduino ide was very simple were library files, example programs, compiling, uploading sketches is easier. By using this device can prior understand the changes in the air and take precautions according to it. Rather than cutting of the trees try to grow more plants and keep our health healthy

**CHAPTER 5**

**ALGORITHMS AND LIBRARIES**

**CHAPTER 5**

**ALGORITHM**

Step 1: Connect USB cable wire one end to Arduino uno board and another end to Laptop

Step 2. Compile and Upload the code to arduino ide

Step 3. Open Serial Monitor in arduino ide

Step 4. Can Monitor the values in LCD ,Serial monitor and Thingspeak

Step 5. When the smoke increasing and crossing threshold value buzzer starts to beep

Step 6. Once smoke decrease ,automatically buzzer stop beeping

Step 7. All the values will be recorded in the Thingspeak

**CHAPTER 6**

**IMPLEMENTATION**

**CHAPTER 6**

**IMPLEMENTATION**

Real time Smog monitoring machine which is used to measure the quality of the air is it “FRESH AIR” or “POLLUATED AIR”. Cost effective machine and easy for installation which can be installed or setup in various locations of the cities. IOT based open source microcontroller board ARDUINO UNO 8-bit microcontroller ATmega328P contains great features such as more instructions every cycle, more frequently, built-in control, flexibility & usability, adjustable pins, rapid start, additional flash memory, low voltage demand ,play and plug. USB connection and extra storage. Used 4 different sensors which is used to generate the data and send to the cloud server thingspeak. Thingspeak is an best platform which is used to store the sensors data with the interface of API key over an internet. After singed into the thingspeak account individual channels is been created for each different sensors , best user interface such as charts , graphs and collaborating app with web services. Once after the setup connections of channels are created and automatically data are visualised. Connect USB cable wire one end to Arduino uno board and another end to Laptop Compile and Upload the code to arduino ide Open Serial Monitor in arduino ide Can Monitor the values in LCD ,Serial monitor and Thingspeak When the smoke increasing and crossing threshold value buzzer starts to beep Once smoke decrease ,automatically buzzer stop beeping. All the values will be recorded in the Thingspeak

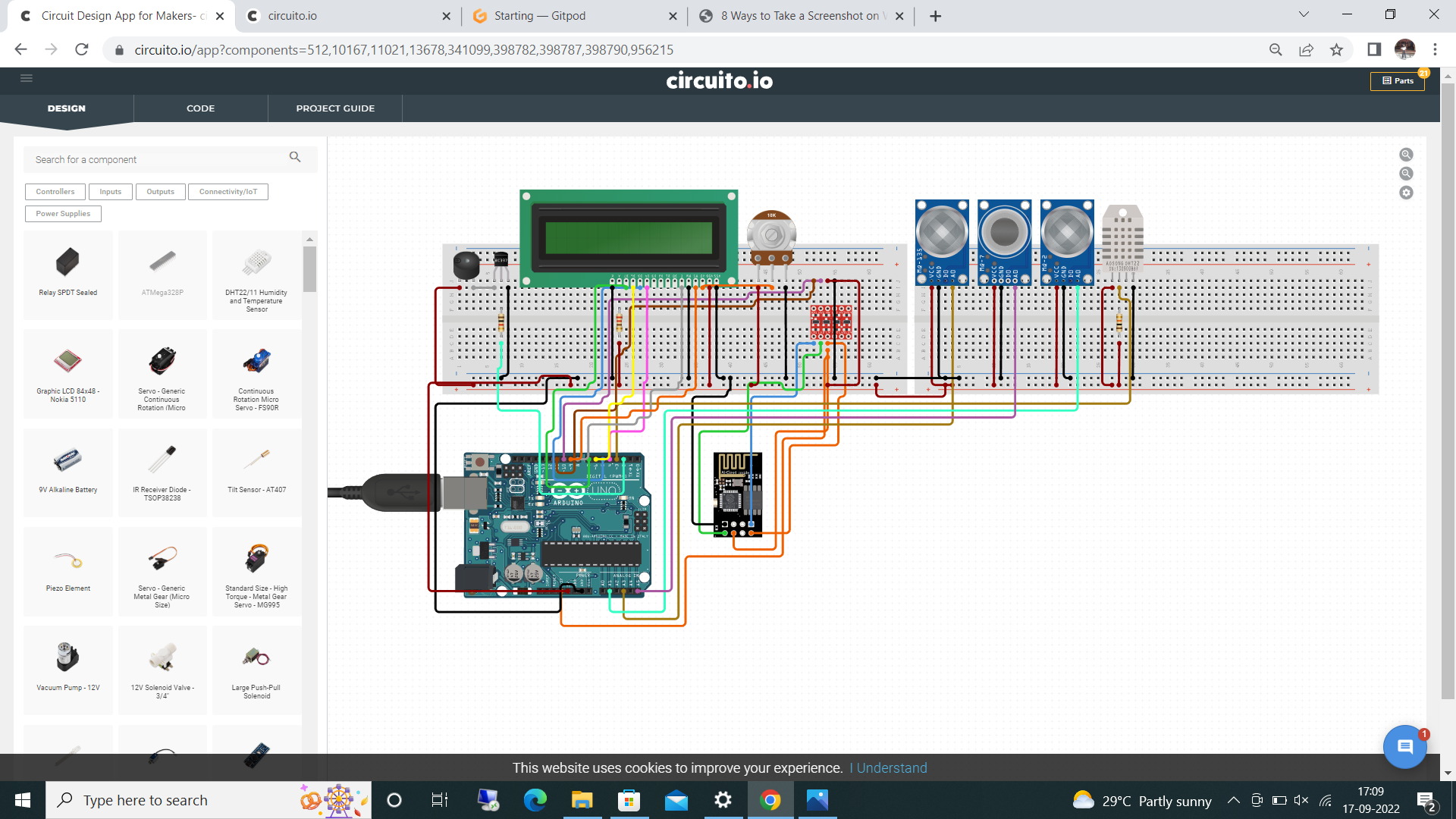
 

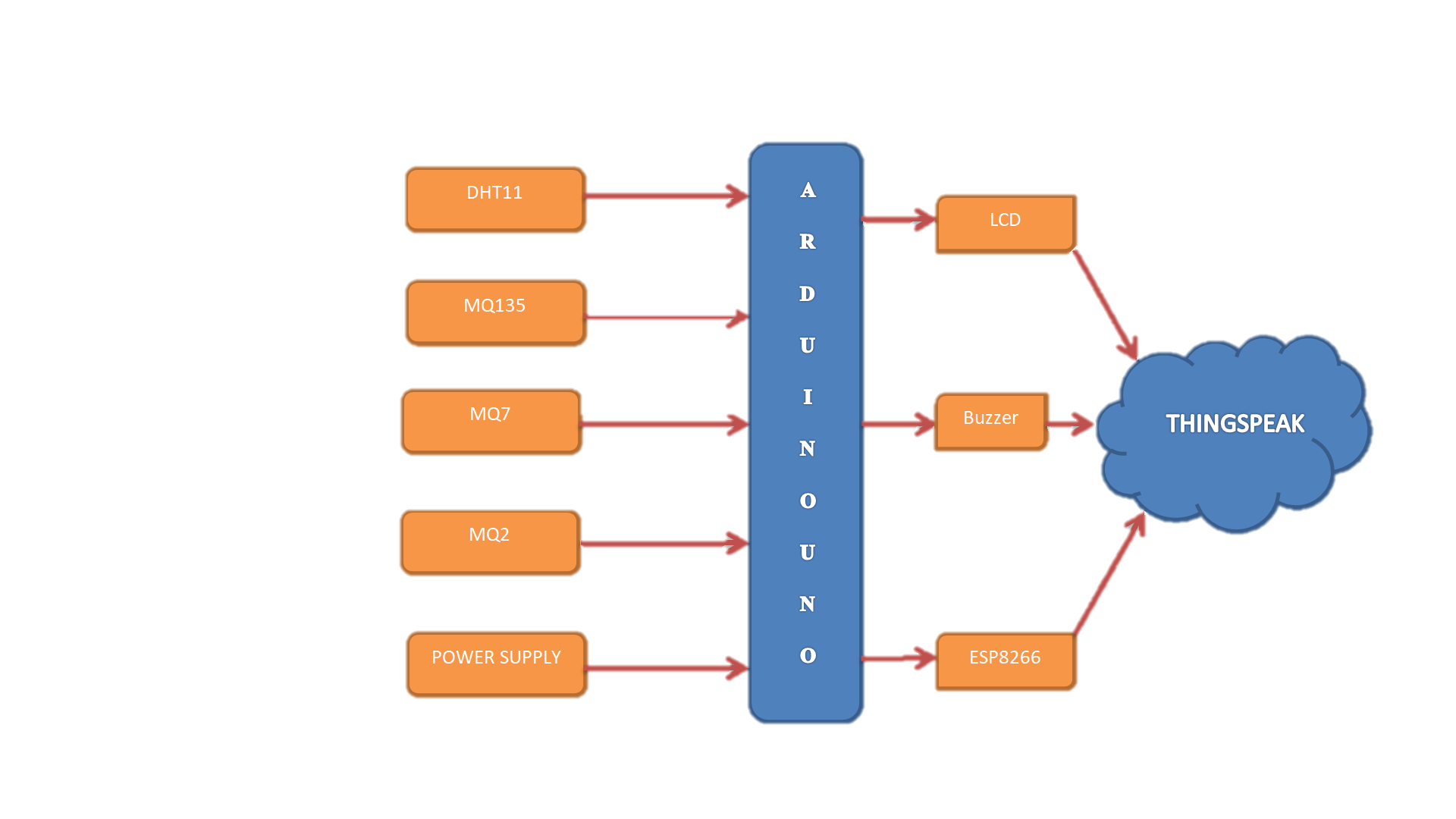
 



**CIRCUIT DIAGRAM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| COMPONENTS | PINS DESCRIPTION | | | |
| DHT11 | DHT11 VCC TO MQ135 VCC | DHT11 A0 TO ARDUINO UNO A3 PIN | | DHT11 GND TO MQ135 GND |
| MQ135 | MQ135 VCC TO MQ7 VCC | MQ135 A0 PIN TO ARDUINO UNO A2 PIN | | MQ135 GND TO MQ7 GND |
| MQ7 | MQ7 VCC TO MQ2 VCC | MQ7 A0 PIN TO ARDUINO UNO A1 PIN | | MQ7 GND TO MQ2 GND |
| MQ2 | MQ2 VCC TO ARDUINO UNO 5V.V | MQ2 A0 PIN TO ARDUINO UNO A0 PIN | | MQ2 GND TO ARDUINO UNO GND |
| BUZZER | POSITIVE(+) TO ARDUINO UNO GND | | NEGATIVE(-) TO ARDUINO UNO D8 PIN | |
| ESP8266 | ESP8266 GND TO ARDUINO UNO GND  ESP8266 TX TO ARDUINO UNO TX  ESP8266 VCC TO ARDUINO UNO 3.3V  ESP8266 RX TO ARDUINO UNO RX  ESP8266 RESET TO ARDUINO UNO GND  ESP8266 - CHEN 220oh TO VCC  ESP8266 – GPIO-0 TO GND | | | |
| LCD 16\*2 | LCD VSS TO BREADBOARD GROUND BUS(-)  LCD VDD TO BREADBOARD POWER BUS(+)  POTENTIOMETER 10K INPUT(Vout) TO LCD VE  LCD RS TO ARDUINO UNO D12 PIN  LCD RW TO BREADBOARD GROUND BUS (-)  LCD ENABLE(E) TO ARDUINO UNO D11 PIN  LCD BACKLIGHT ANODE TO BREADBOARD GROUND BUS(-)  LCD BACKLIGHT CATHODE 220oh RESISTOR TO BREADBOARD GROUND BUS(+)  LCD D7 TO ARDUINO UNO D2 PIN  LCD D6 TO ARDUINO UNO D3 PIN  LCD D5 TO ARDUINO UNO D4 PIN  LCD D4 TO ARDUINO UNO D5 PIN | | | |



****

**CHAPTER 7**

**CODING**

**CHAPTER 7**

**CODING**

**MQ2 GAS SENSOR**

#define MQ2pin (0) // connected to A0 of arduino

#include <LiquidCrystal.h>

float sensorValue; //variable to store sensor value

const int buzzer = 8 ;

int MQ2 = A0;

int sensorThres = 150;

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup()

{

pinMode(buzzer, OUTPUT);

Serial.begin(9600); // sets the serial port to 9600

Serial.println("Gas sensor warming up!");

delay(20000); // allow the MQ-2 to warm up

**}**

**void loop()**

**{**

int analogSensor = analogRead(MQ2);

Serial.print("Pin A2: ");

Serial.println(analogSensor);

if (analogSensor > sensorThres)

{

tone(buzzer, 200, 100);

}

else

{

noTone(buzzer);

}

delay(100);

sensorValue = analogRead(MQ2pin); // read analog input pin 0

Serial.print("Sensor Value: ");

Serial.print(sensorValue);

if(sensorValue > 200)

{

Serial.print(" |Warning! AIR POLLUTED HIGH!");

lcd.print(" AIR POLLUTED HIGH!");

}

else

{

Serial.print("AIR POLLUTED LESS");

lcd.print(" AIR POLLUTED HIGH!");

}

Serial.println("");

delay(2000); // wait 2s for next reading

}

**MQ135 GAS SENSOR**

#include <LiquidCrystal.h>

int sensorValue;

const int buzzer = 8 ;

int MQ135 = A2;

int sensorThres = 800;

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup()

{

lcd.begin(16, 2);

Serial.begin(9600);

pinMode(MQ135, INPUT);

pinMode(buzzer, OUTPUT);

}

void loop()

{

int sensorValue = analogRead(MQ135);

Serial.print("Pin A2: ");

Serial.println(sensorValue);

if (sensorValue > sensorThres)

{

tone(buzzer, 1000, 200);

}

else

{

noTone(buzzer);

}

delay(100);

sensorValue = analogRead(2); // read analog input pin 0

Serial.print("Air Polluted HIGH! =");

Serial.print(sensorValue, DEC); // prints the value read

Serial.println(" PPM");

lcd.setCursor(0,0);

lcd.print("MQ135=");

lcd.print(sensorValue,DEC);

lcd.print(" PPM");

lcd.println(" ");

lcd.print(" ");

delay(2000); // wait 100ms for next reading

}

**MQ7 GAS SENSOR**

#include <LiquidCrystal.h>

int sensorValue;

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup()

{

// initialize serial communication at 9600 bits per second:

lcd.begin(16, 2);

Serial.begin(9600);

}

void loop() {

int sensorValue = analogRead(A1); // connected to A1 of arduino

Serial.print("MQ7 READINGS : ");

Serial.println(sensorValue);

lcd.setCursor(0,1);

lcd.print("mq7 = ");

lcd.print(sensorValue,DEC);

lcd.println(" ");

lcd.print(" ");

delay(100);

}

**DHT11 SENSOR**

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#include <DHT\_U.h>

#define DHTTYPE DHT11 // DHT 11

#define DHTPIN 3

DHT\_Unified dht(DHTPIN, DHTTYPE);

uint32\_t delayMS;

void setup() {

Serial.begin(9600);

dht.begin();

sensor\_t sensor;

delayMS = sensor.min\_delay / 1000;

}

void loop() {

sensors\_event\_t event;

dht.temperature().getEvent(&event);

Serial.print(F("Temperature: "));

Serial.print(event.temperature);

Serial.println(F("°C"));

dht.humidity().getEvent(&event);

Serial.print(F("Humidity: "));

Serial.print(event.relative\_humidity);

Serial.println(F("%"));

delay(delayMS);

}

**ESP8266 WIFI MODULE**

#include <SoftwareSerial.h>

#define RX 7

#define TX 6

String AP = "Redmi 9A"; // AP NAME

String PASS = "1234567890"; // AP PASSWORD

String API = "XYMYJ9BEQHM5UDL4"; // Write API KEY

String HOST = "api.thingspeak.com";

String PORT = "30";

int countTrueCommand;

int countTimeCommand;

boolean found = false;

int valSensor = 1;

SoftwareSerial esp8266(RX,TX);

void setup()

{

Serial.begin(9600);

esp8266.begin(115200);

sendCommand("AT",5,"OK");

sendCommand("AT+CWMODE=1",5,"OK");

sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK");

}

void loop()

{

String getData = "GET /update?api\_key="+ API +"&field1="+gettemphumidValue()+"&field2="+getMQ135Value()+"&field3="+getMQ7Value()+"&field4="+getMQ2Value();

sendCommand("AT+CIPMUX=1",5,"OK");

sendCommand("AT+CIPSTART=0,\"TCP\",\""+ HOST +"\","+ PORT,15,"OK");

sendCommand("AT+CIPSEND=0," +String(getData.length()+4),4,">");

esp8266.println(getData);delay(1500);countTrueCommand++;

sendCommand("AT+CIPCLOSE=0",5,"OK");

}

String gettemphumidValue()

{

int temphumid;

temphumid=analogRead(A3);

return String(temphumid);

}

String getMQ135Value()

{

int MQ135;

MQ135=analogRead(A2);

return String(MQ135);

}

String getMQ7Value()

{

int MQ7;

MQ7=analogRead(A1);

return String(MQ7);

}

String getMQ2Value()

{

int MQ2;

MQ2=analogRead(A0);

return String(MQ2);

}

void sendCommand(String command, int maxTime, char readReplay[])

{

Serial.print(countTrueCommand);

Serial.print(". at command => ");

Serial.print(command);

Serial.print(" ");

while(countTimeCommand < (maxTime\*1))

{

esp8266.println(command);//at+cipsend

if(esp8266.find(readReplay))//ok

{

found = true;

break;

}

countTimeCommand++;

}

if(found == true)

{

Serial.println("OYI");

countTrueCommand++;

countTimeCommand = 0;

}

if(found == false)

{

Serial.println("Fail");

countTrueCommand = 0;

countTimeCommand = 0;

}

found = false;

}

**FULL CODE**

#include <Wire.h>

#include <DHT.h>

#include <LiquidCrystal.h>

#define DHTPIN A3

#define DHTTYPE DHT11

#define DELAY\_IN\_SEC 2000

DHT dht(DHTPIN, DHTTYPE);

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

const int mq2 = A0;

const int mq7 = A1;

const int mq135 = A2;

const int buzzer = 8;

float m = -0.6527;

float b = 1.30;

float R0 = 21.91;

double mq135Limit = 800;

double temperatureLimit = 99.5;

double humidityLimit = 50;

double mq7Limit = 400;

double mq2Limit = 10000;

double analogToPPM(int aValue)

{

float sensor\_volt;

float RS\_gas;

float ratio;

int sensorValue = aValue;

sensor\_volt = sensorValue \* (5.0 / 1023.0);

RS\_gas = ((5.0 \* 10.0) / sensor\_volt) - 10.0;

ratio = RS\_gas / R0;

double ppm\_log = (log10(ratio) - b) / m;

return ppm\_log;

}

void setup()

{

Serial.begin(9600);

lcd.begin(16, 2);

pinMode(mq135, INPUT);

pinMode(mq7, INPUT);

pinMode(mq2, INPUT);

pinMode(buzzer, OUTPUT);

dht.begin();

lcd.setCursor(0, 0);

lcd.print(" Atmosphere ");

lcd.setCursor(0, 3);

lcd.print("Gas Analyser");

delay (3000);

lcd.clear();

lcd.print("By");

lcd.setCursor(0, 3);

lcd.print("Kanaka");

delay (3000);

lcd.clear();

}

void loop()

{

//--------- DHT11 ---------//

float humidity = dht.readHumidity();

float temp = dht.readTemperature();

float f = dht.readTemperature(true);

if (isnan(humidity) || isnan(temp) || isnan(f)) {

Serial.println(F("Failed to read from DHT sensor!"));

return;

float hif = dht.computeHeatIndex(f, humidity);

float hic = dht.computeHeatIndex(temp, humidity, false);

}

//--------- DHT11 ---------//

double mq135\_value = analogToPPM(analogRead(mq135));

double mq7\_value = analogToPPM(analogRead(mq7));

double mq2\_value = analogToPPM(analogRead(mq2));

Serial.printl n("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Serial.print("MQ-135 Value: "); Serial.print(mq135\_value, DEC); Serial.println(" PPM");

Serial.print("MQ-7 Value: "); Serial.print(mq7\_value, DEC); Serial.println(" PPM");

Serial.print("MQ-2 Value: "); Serial.print(mq2\_value, DEC); Serial.println(" PPM");

Serial.print("DHT-Temp: "); Serial.print(temp); Serial.println(" c");

Serial.print("DHT-Humi: "); Serial.print(humidity); Serial.println(" %");

Serial.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

lcd.setCursor(0, 0);

lcd.print("mq135");

lcd.setCursor(0, 3);

lcd.println(mq135\_value);

lcd.setCursor(7, 7);

lcd.print("PPM");

if (mq135\_value > mq135Limit) {

digitalWrite(buzzer, HIGH);

}

delay(DELAY\_IN\_SEC);

digitalWrite(buzzer, LOW);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("mq7");

lcd.setCursor(0, 3);

lcd.println(mq7\_value);

lcd.setCursor(7, 7);

lcd.print("PPM");

if (mq7\_value > mq7Limit) {

digitalWrite(buzzer, HIGH);

}

delay(DELAY\_IN\_SEC);

digitalWrite(buzzer, LOW);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("mq2");

lcd.setCursor(0, 3);

lcd.println(mq2\_value);

lcd.setCursor(7, 7);

lcd.print("PPM");

if (mq135\_value > mq2Limit) {

digitalWrite(buzzer, HIGH);

}

delay(DELAY\_IN\_SEC);

digitalWrite(buzzer, LOW);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("DHT-11");

lcd.setCursor(0, 3);

lcd.print("Temp -");

lcd.setCursor(7, 4);

lcd.println(temp);

if (temp > temperatureLimit) {

digitalWrite(buzzer, HIGH);

}

delay(DELAY\_IN\_SEC);

digitalWrite(buzzer, LOW);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("DHT-11");

lcd.setCursor(0, 3);

lcd.print("Humi -");

lcd.setCursor(7, 4);

lcd.println(humidity);

if (temp > humidityLimit) {

digitalWrite(buzzer, HIGH);

}

delay(DELAY\_IN\_SEC);

digitalWrite(buzzer, LOW);

lcd.clear();

}

**CHAPTER 8**

**RESULTS**

**CHAPTER 8**

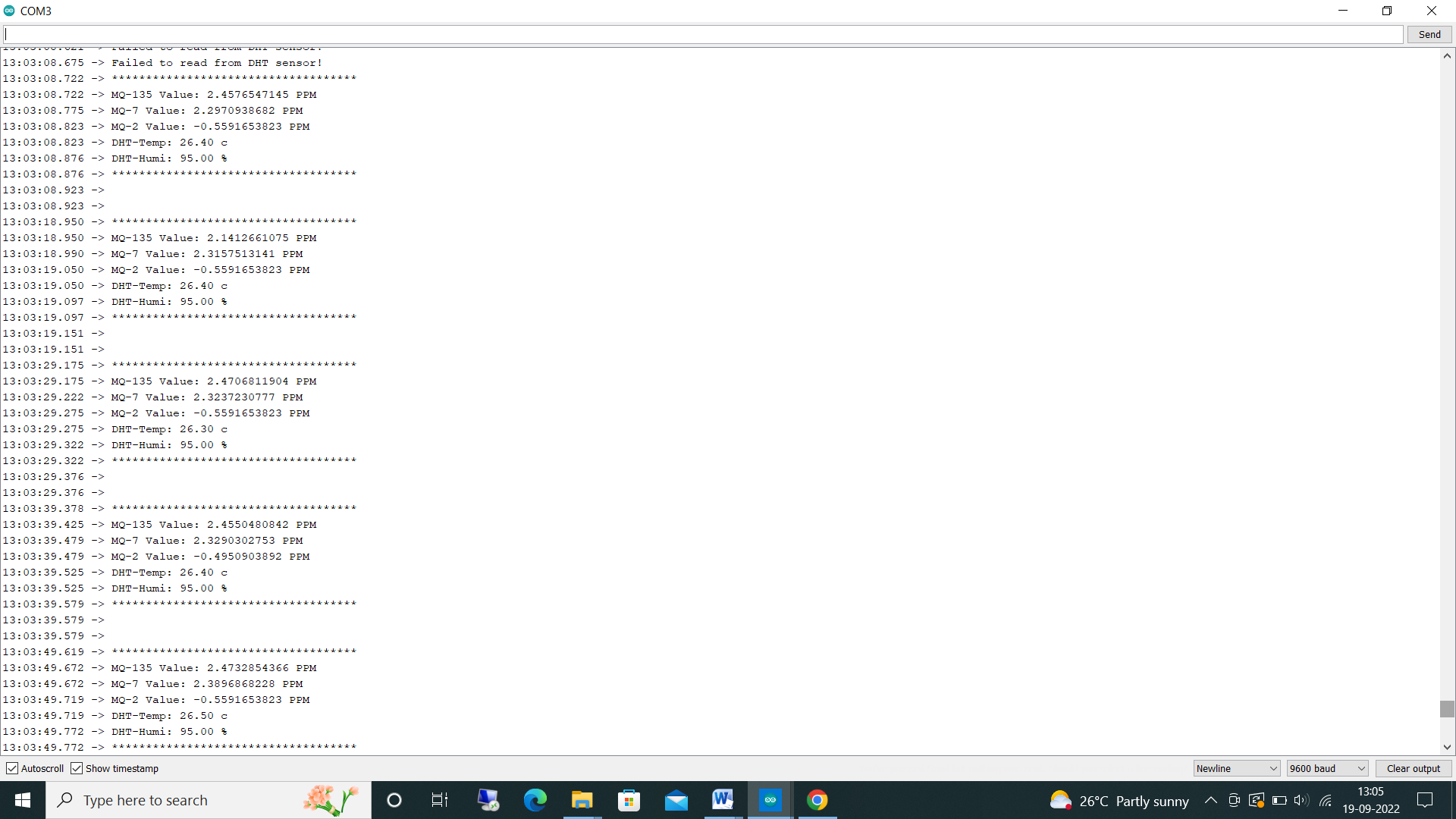
**RESULTS**

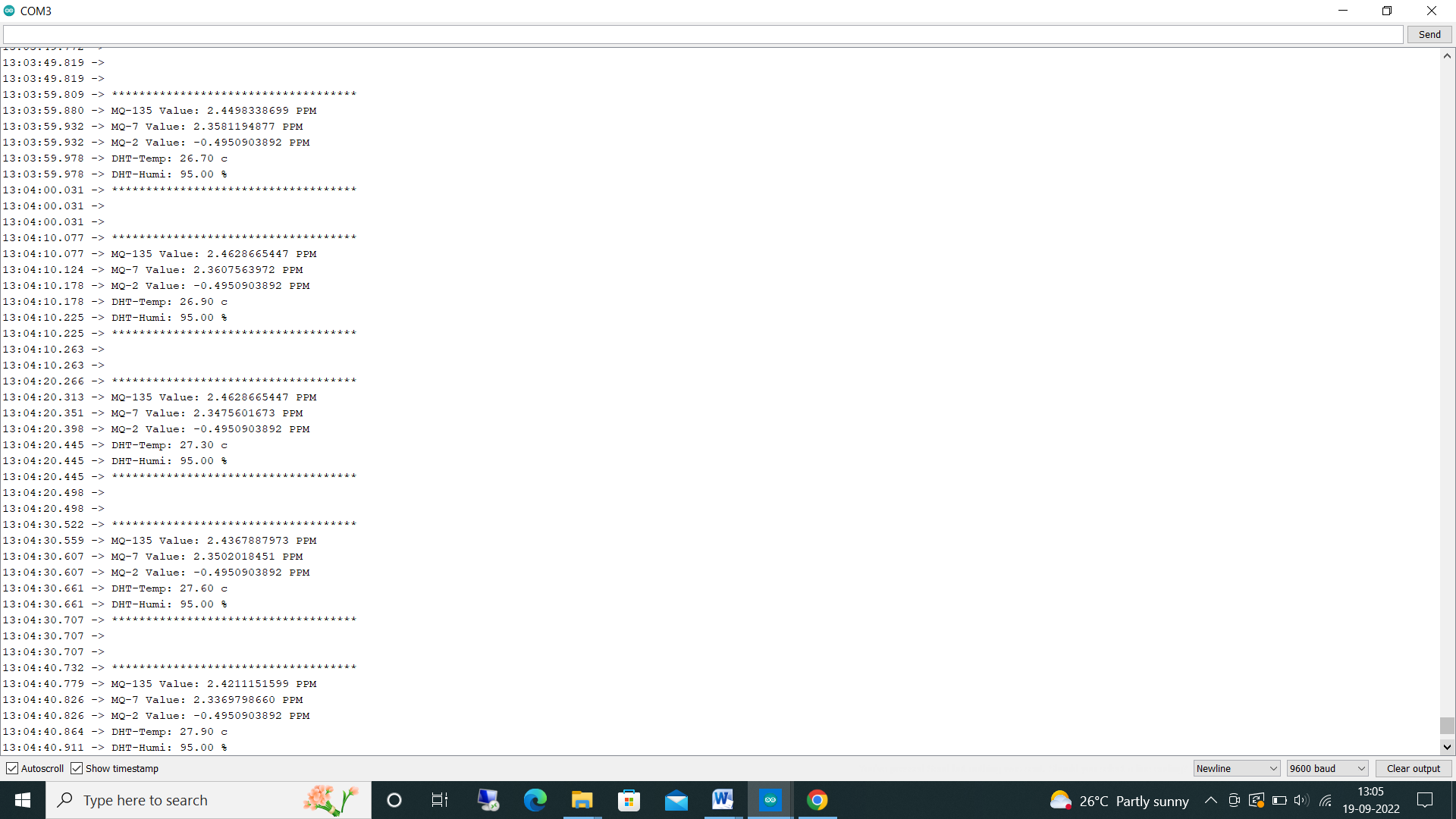


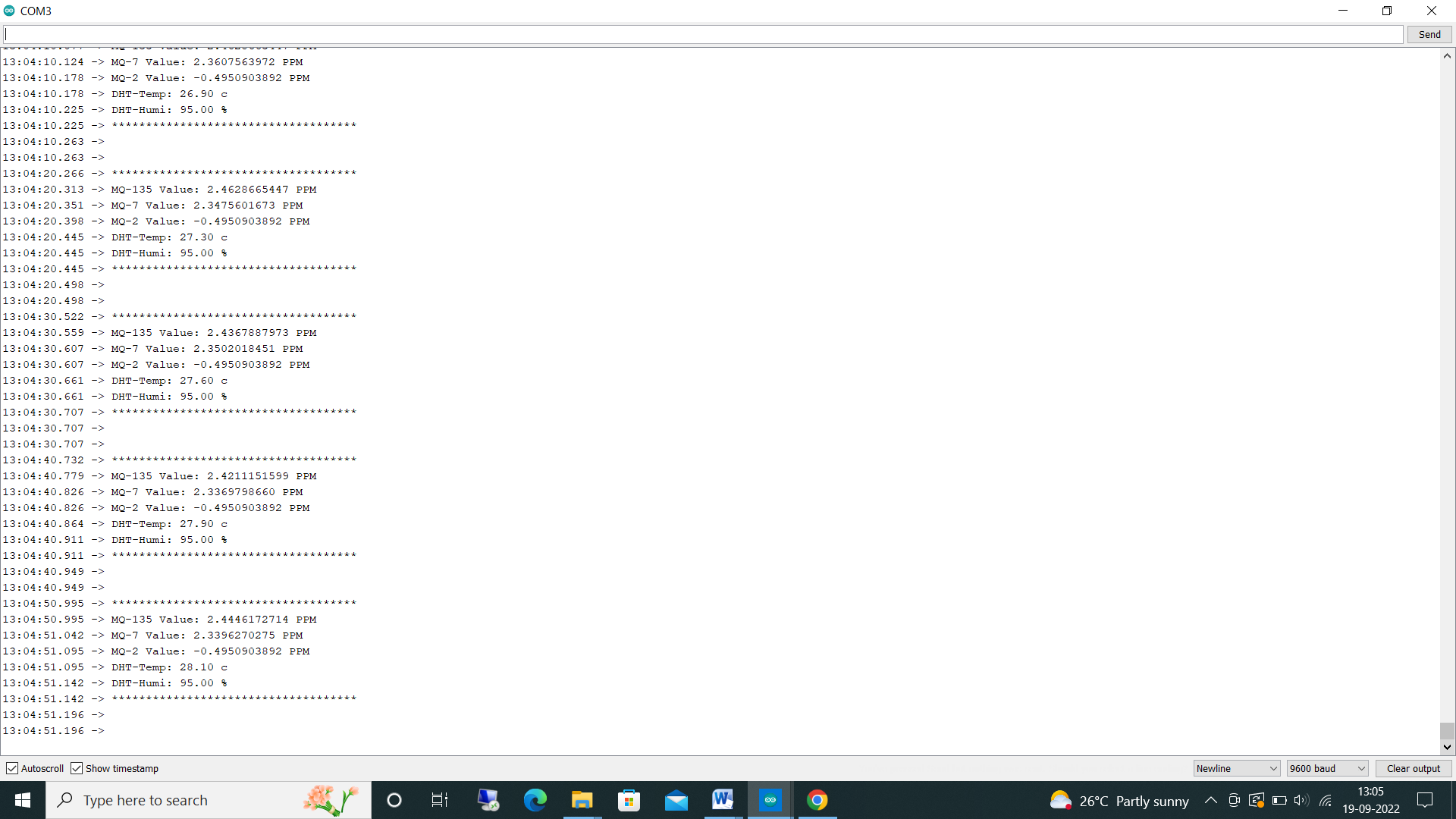


****



****

****

****

**CHAPTER 9**

**CONCLUSION**

**CHAPTER 9**

**CONLUSIONS**

Wanted to take initiative and bring the change in the society where this device will be very useful for human’s and setup green environment by growing more plants and trees. Smog monitoring is an mini electronic monitoring device where it monitors all kind of poisonous gases such as carbon dioxide, carbon monoxide, methane, LPG, propane, sulphur, ammonia, butane, nitrogen dioxide. Cost effective and Portable device were we can setup in commercial and industrial purpose. Must stop burning of wastage, put layers of filters in the factory funnels, use public transport rather than using individual vehicles etc. Arduino Uno board is an best choice for iot projects. Could have taken one gas sensor for monitoring but we have used 4 different sensors because each sensor as different feature, ppm value and percentage varies. Data is stored and monitored through serial monitor in arduino ide and THINGSPEAK cloud. Coding in arduino ide was very simple were library files, example programs, compiling, uploading sketches is easier. By using this device can prior understand the changes in the air and take precautions according to it. Rather than cutting of the trees try to grow more plants and keep our health healthy

**FUTURE SCOPE**

**FUTURE WORK**

Want to upgrade the device by using Raspberry Pi board because it has great features such as fast processor, support all different programming codes etc. Create on and off automatic sensor when air getting polluted device automatically gets ON and when air is getting fresh automatically gets OFF. Create a software application where users can easily monitor through an app.

**REFRENCE**

**REFERENCES**

[1] Tejas Naik, R Roopa Lakshmi, N. Divya Ravi Pawdhan Jain, B. H. Sowmya, Mani Chandra air monitiorning syste 2018 2nd International conference on Inventive communication conference

[2]"Priority based and secured smog monitorning ! Act Abdullahi Chowdhary. 2016 International conference

[3] "Shubhankar Vishwas Bhate, Prasad Vilas Kulkarni, Shubham Dhanaji lagad, Mahesh. Dhyaneshwar Shinde, shivaprasad Patil "107 based intelligent Traffic signal system for emergency Conference on inventive Computational Technologies. 2018 2nd International Communication and

[4] AM Amaresh, Kavya Shivanand Bhat, G.. Ashwin J. Bhagyashree, P. Aishwarya. "smog monitorning co2, co,butane,2019 Internal Intelligent Computing and control system (Iccs), International conference

[5] Anam Firdous, Indu, Vandana Niranjan Reliability air pollution equipment System " 2020 8th International conference on Infocom Technologies and optimization.

[6] W.A.C. J.K. Chandrasekara, R. M.KT Rathnayaka, L. L. G Chathuranga "A Real Time Smog Monitorning System" 2022 5th International conference on Information Technology Research