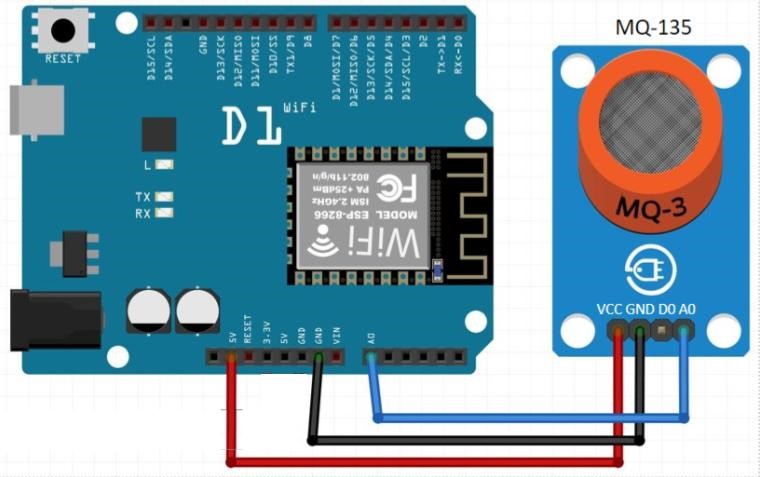
**Work Experience Week 2021 Air Quality Project**

**4. Wiring Up and coding MQ-135**

1. Unplug your Wemos from the USB cable.   
   Wire up **MQ-135 -> wemos**, with the jumper cables   
   (the connections on the MQ-135 are labelled on the back of it’s board)   
   *Note: the colour of the cables doesn’t matter. Your connection layout may be different just ensure you’re connecting to the correctly labelled pins.*   
    **MQ-135 -> Wemos**,
2. **A0 -> A0**
3. **Ignore D0.**
4. **GND -> GND**
5. **VCC -> 5V**



1. Download MQ-135 library: <https://github.com/ViliusKraujutis/MQ135> (click Code > Download ZIP). This library helps us read values from the sensor.
2. In Arduino click Sketch > Include Library > Add .ZIP Library and add the zip file from your downloads folder.
3. Start a new sketch in Arduino (File > New)
4. Add the following to the **top of your code**, to initiate the sensor library and tell the wemos you’ve connected the MQ135 to pin A0:

#include "MQ135.h"

MQ135 mq135\_sensor = MQ135(A0);

float temperature = 21.0;

float humidity = 50.0;

1. Guess a rough temperature and humidity for your room and fill those values in the above code. (I’ve put 21C and 50% humidity for my room) This is used to help calculate a ppm value for air quality from the MQ-135 sensor. We’ll improve this measurement with our temperature/humidity sensor later!
2. Add the following as the **setup()** to set up the communication over usb:

void setup(){

Serial.begin(115200);

}

1. Now lets make a loop where we read the MQ-135 and relay it back to our computer to see what values we’re getting :

void loop(){

float airquality = mq135\_sensor.getCorrectedPPM(temperature, humidity);

Serial.print("Air Quality: ");

Serial.println(airquality);

delay(5000);

}

**What this code does:**

* **Serial begin –** sets the data rate over the USB cable in bits per second
* **Float** “airquality” is a **variable** – we're storing the value of PPM we’re measuring from the MQ-135
* We’ll then **Serial** **print -** display that value on the computer over serial (USB)
* and then **delay** 5000 miliseconds (wait 5 seconds)
* and do another measurement over and over again as this bit of code is inside the loop { }, writing a new value to “a” and printing that value.

1. Click the  verify button to check your code for any errors. If it’s ok it should say ”Done Compiling” at the bottom. If not, double check your code against the code above for any mistakes, if you’re not sure, do ask for help.
2. Now we’ve created some code to read the MQ-135 air quality sensor lets test it! Plug the USB cable into the wemos and click the  upload button.
3. Once the code is fully uploaded click the  serial monitor button. Make sure at the bottom right of the serial monitor the baud is set to 115200 baud. Every 5 seconds you should see a value appear from the MQ-135 sensor.

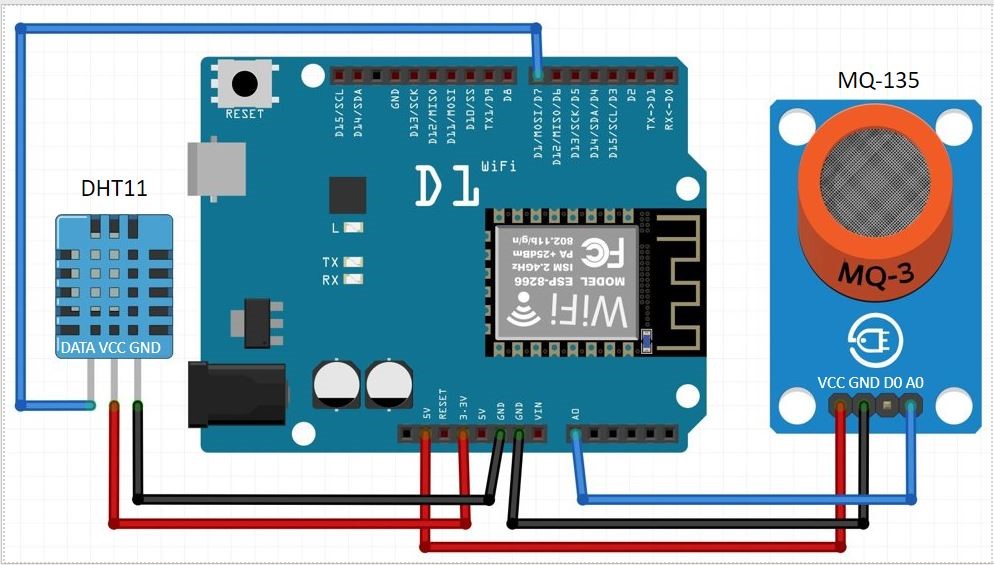
The sensor takes around 60 seconds to properly heat up and get roughly correct readings. If you breathe on the sensor you should see the value change as you’re increasing CO2 on it!

**5. Wiring Up and coding DHT11**

1. Unplug your Wemos from the USB cable. Wire up **DHT11 -> wemos**; (left to right on DHT11):

**DHT11 -> wemos**;

1. **+ -> 3V3**
2. **Out -> D7**
3. **- -> GND**



1. Download DHT11 library: <https://github.com/beegee-tokyo/DHTesp> (click Code > Download ZIP). This library helps us read values from the sensor.
2. In Arduino click Sketch > Include Library > Add .ZIP Library and add the zip file from your downloads folder.
3. Add the following to the **top of your code** to initiate the sensor library:

#include "DHTesp.h"

DHTesp dht;

1. Add the following code to the **end (just before } ) of setup** to initialise the DHT11 and tell the wemos we’ve wired it to pin D7:

dht.setup(D7, DHTesp::DHT11);

1. Add the following code to the **beginning of void loop**:

humidity = dht.getHumidity();

temperature = dht.getTemperature();

Serial.print("Humidity: ");

Serial.print(humidity);

Serial.print(", Temperature: ");

Serial.println(temperature);

**What this code does:**

It's now updating the variables “humidity” and “temperature” that you filled in earlier with readings from the DHT11 sensor and printing this to the serial monitor. These values are now fed in to the calculation for air quality ppm coming from the MQ-135 for a more accurate reading!

Verify , and Upload  the code, once uploaded check the temperature, humidity and air quality readings from your sensors using the serial monitor 

That’s all the sensor code done and hopefully working! Next session, we’ll hook this up to the IoT platform so we can store and visualise the data.

**6.Connecting the Wemos To Thingspeak**

**Sign in to Thingspeak:**

1. Go to Thingspeak.com. Create an account, verify your email.
2. Click Channels > New Channel
3. Set Channel name as your name
4. Click Save channel
5. Click Sharing > Share channel view with everyone
6. Click API Keys and copy your Write API Key. You’ll need this API key for the code below.

**Template Thingspeak / IoT code:**

1. In Arduino, File > New. Clear the default code and copy paste the code below:

#include <ESP8266WiFi.h>

// replace with your channel’s write API key (on the API Keys tab) and your wifi name and password

String writeApiKey = "write api key here";

const char\* ssid = "wifi name here";

const char\* password = "wifi password here";

const char\* server = "api.thingspeak.com";

int t = 1;

WiFiClient client;

void setup(){

Serial.begin(115200);

delay(10);

WiFi.begin(ssid, password);

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

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

delay(500);

Serial.print(".");

}

Serial.println("WiFi connected");

}

void loop()

{

if (client.connect(server,80)) {

String postStr = writeApiKey;

postStr +="&field1=";

postStr += String(t);

client.print("POST /update HTTP/1.1\n");

client.print("Host: api.thingspeak.com\n");

client.print("Connection: close\n");

client.print("X-THINGSPEAKAPIKEY: "+writeApiKey+"\n");

client.print("Content-Type: application/x-www-form-urlencoded\n");

client.print("Content-Length: ");

client.print(postStr.length());

client.print("\n\n");

client.print(postStr);

Serial.println("Sending data to Thingspeak");

Serial.print("Value: ");

Serial.println(t);

t = t + 1;

}

client.stop();

Serial.println("Waiting 20 secs");

// thingspeak needs at least a 15 sec delay between updates

// 20 seconds to be safe

delay(20000);

}

**Useful bits of code:**

**Char / string** – store zero or more characters

**Int t –** store a number – in this case “t” you’re sending to thingspeak

**t = t + 1** - adding one to integer t (within the loop)

1. This code sets up wifi on the wemos and then transmits basic data to the thingspeak site. For now it’s going to transmit the variable “t” every 20 seonds, but every time it’s going to add 1 to t.   
   In the code you’ll need to change the following:
2. Get your Write API key from thingspeak, paste this in the quotes to replace write api key here
3. Your wifi network name, paste this in the quotes to replace wifi name here
4. Your wifi password, paste this in the quotes to replace wifi password here
5. Verify  the code runs, plug in your wemos and then upload 
6. Once uploaded, click the serial monitor button 
7. Change the serial monitor baud to 115200
8. You should see the wifi connect and the value get sent to thingspeak
9. Every 20 seconds it will send a new value to team speak (currently integer t which is adding 1 every loop.
10. Have a look on the public view tab on thingspeak – you should see the value go up by 1 every 20 seconds!
11. Now you’ve created an IOT device, let’s add the sensors.

**7. Connect the sensors up to thingspeak**

We’ll now add back our sensor code from the last session into this thingspeak code, so here’s the changes to make to your code:

1. Add to the **top** of your code (the code that sets up the sensor libraries):

#include "DHTesp.h"

DHTesp dht;

#include "MQ135.h"

MQ135 mq135\_sensor = MQ135(A0);

float temperature = 21.0;

float humidity = 50.0;

1. Add to the end of **setup** within { } to link up the DHT11 to pin D7:

dht.setup(D7, DHTesp::DHT11);

1. At the top of **void loop** delete these 3 lines (you’re removing the test data we were sending to teamspeak):

String postStr = writeApiKey;

postStr +="&field1=";

postStr += String(t);

1. add these lines below **if (client.connect(server,80)) {**    
   for measuring the sensors and sending them to teamspeak:

humidity = dht.getHumidity();

temperature = dht.getTemperature();

Serial.print("Humidity: ");

Serial.print(humidity);

Serial.print(", Temperature: ");

Serial.println(temperature);

float airquality = mq135\_sensor.getCorrectedPPM(temperature, humidity);

Serial.print("Air Quality: ");

Serial.println(airquality);

String postStr = writeApiKey;

postStr +="&field1=";

postStr += String(airquality);

postStr +="&field2=";

postStr += String(humidity);

postStr +="&field3=";

postStr += String(temperature);

1. At the **bottom of void loop delete** these 3 lines (we’re no longer using this test data):

Serial.print("Value: ");

Serial.println(t);

t = t + 1;

1. Verify  the code runs. If there are errors have a look at the error black box and see what the issue is. If all is good plug in your wemos and then upload 
2. Once uploaded, click the serial monitor button  You should see it connecting to wifi and printing out the values from the sensors! Now we need to link this up on thingspeak.
3. On thingspeak go to your channel > Channel Settings and tick and add the fields:
4. Field 1: Air Quality (PPM)
5. Field 2: Humidity (%)
6. Field 3: Temperature (°C)
7. Now if you head back to the public view tab on thingspeak you should able to see all three readings coming through as 3 graphs!

If you get the chance leave your sensors running over night between session so you have more data to look at for the next session.