Student name: Hoang Long Tran

Student ID: s223128143

SIT225: Data Capture Technologies

Activity 2.1: Working with sensor - DHT22

DHT22 is a temperature and humidity sensor.

Hardware Required

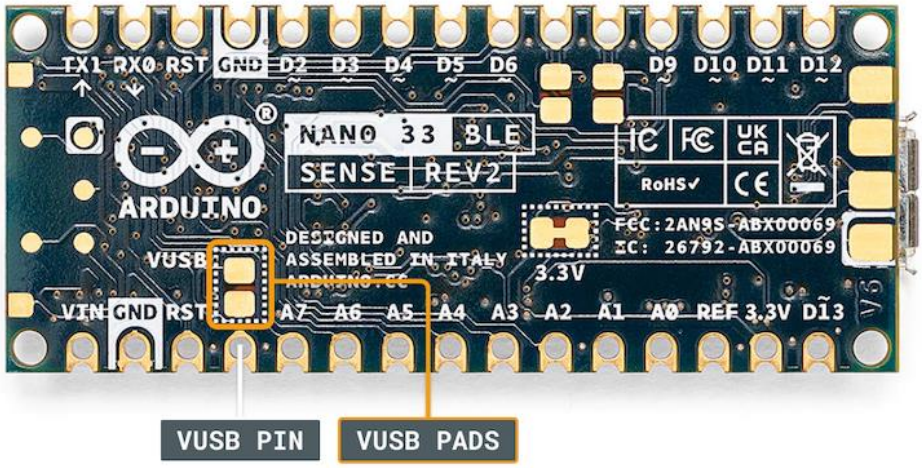
- Arduino Board
- DHT22 sensor
- USB cable

Software Required

- Arduino programming environment

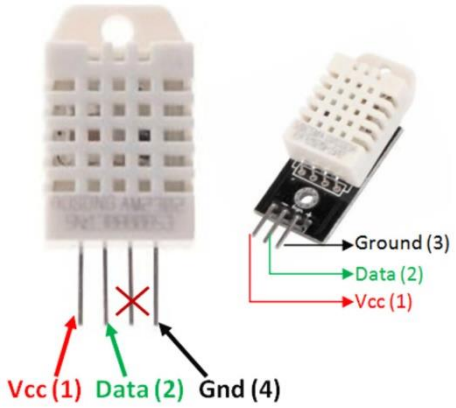
Known issue, action required

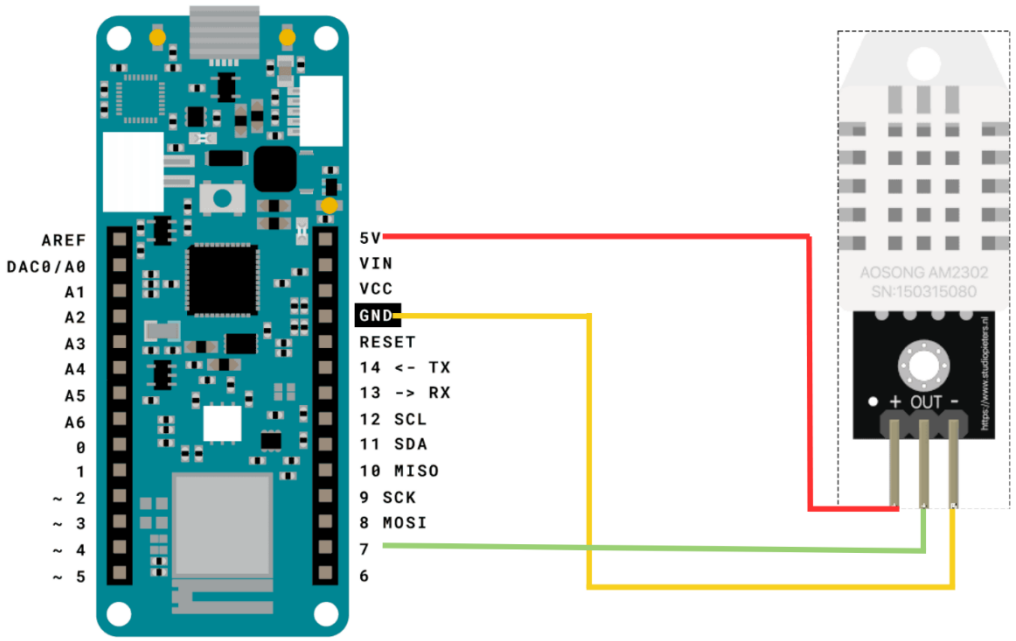
Arduino Nano 33 IoT board operates on 3.3 V, it needs to be arranged to make it 5 V. The Arduino board has a pin called VUSB or VBUS and there are 2 pads next to the pin. **These two pads must be shorted to enable the pin** (see detail here https://support.arduino.cc/hc/en-us/articles/360014779679-Enable-5-V-power-on-the-VUSB-or-VBUS-pin-on-Nano-boards).



To test, you can use a wire and connect these 2 pads manually by hand to see if data is coming through DHT22 sensor. For long data collection, you will need to solder the wire permanently to connect the pads. Seeking help from tutors there is an on-campus facility called Maker Space where you can do it.

Steps:

Step	Action
1	Connect your DHT22 Temperature and Humidity Sensor to the Arduino board. Note that the pin layout in the image below may look different than the board you may have. 

	 a. Pick a red male-female jumper wire and attach the female end to pin 1 (VCC pin) on the sensor. Plug the male end into the Arduino board's 5V power pin. b. Pick a blue male-female jumper wire and attach the female end to pin 2 (DATA pin) on the sensor. Plug the male end into the Arduino board's digital data pin 2. c. Pick a black male-female jumper wire and attach the female end to pin 4 (GND) on the sensor. Plug the male end into the Arduino board's GND pin. d. Sensor's pin 3 is not used.
2	Connect your Arduino board to your computer using the USB cable.
3	Write an Arduino sketch (or download it from https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/sketch_dht22.ino) which looks like below. Compile the code in Arduino IDE, deploy to the board and observe output in the Arduino IDE serial monitor.

```
// Add "DHT sensor library"
#include <DHT.h>

#define DHTPIN 2  // digital pin number
#define DHTTYPE DHT22  // DHT type 11 or 22
DHT dht(DHTPIN, DHTTYPE);

// variables to store data.
float hum, temp;

void setup() {
  // Set baud rate for serial communication
  Serial.begin(9600);

  // initialise DHT libarary
  dht.begin();
}

void loop() {
  // read data
  hum = dht.readHumidity();
  temp = dht.readTemperature();

  // Print data to serial port - a long way
  //
  // Serial.print("Humid: ");
  // Serial.print(hum);
  // Serial.print(" %, Temp: ");
  // Serial.print(temp);
  // Serial.println(" Celsius");

  // Print data to serial port - a compact way
  Serial.println(String(hum) + "," + String(temp));

  // wait a while
  delay(15*1000);
}
```

4

Question: A spec of the DHT22 sensor is given in the link below. It mentions that the sampling rate is 0.5 Hz.

https://lastminuteengineers.com/dht11-dht22-arduino-tutorial

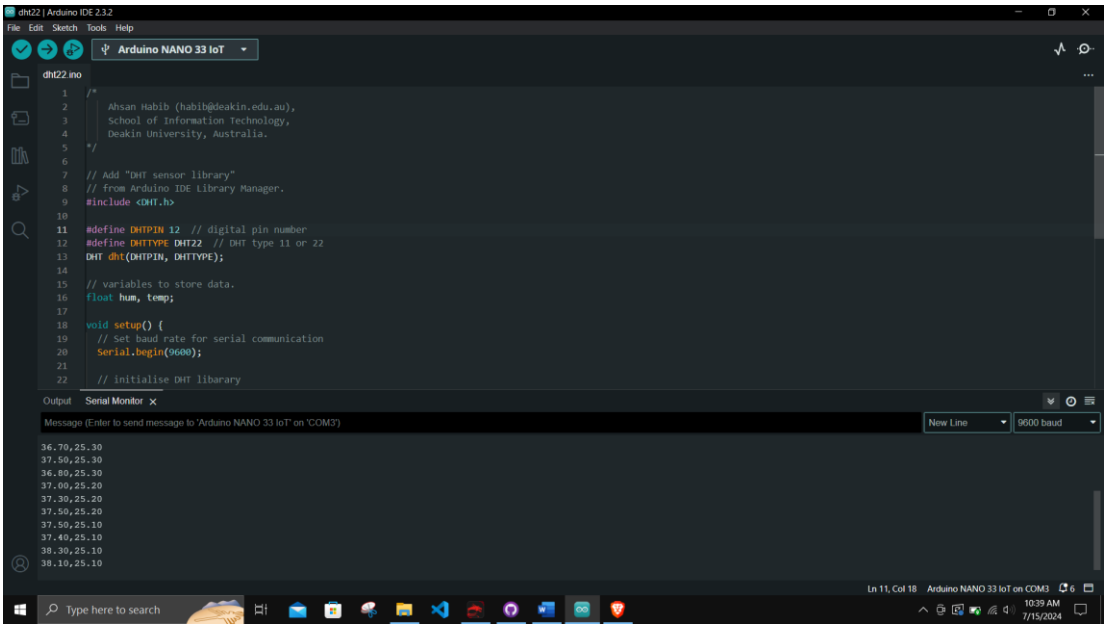
i) What does the sampling rate mean?

ii) Where is this used in the Arduino code?

Answer:

i) The sampling rate means the interval of time the sensor can take in data. The measurement is in Hz, but we can convert it to seconds by 1/0.5 = 2 seconds.

ii) The `delay` in the Arduino code pause the program for enough time for the dht sensor to capture the next interval of data.

5	Question: Take a screenshot of your Serial Monitor displaying temperature & humidity sensor data logs. Add the image here. Answer: 

Activity 2.2: Working with sensor - HC-SR04

HC-SR04 is an Ultrasonic sensor.

Hardware Required

Arduino Board

HC-SR04 Ultrasonic sensor

USB cable

Software Required

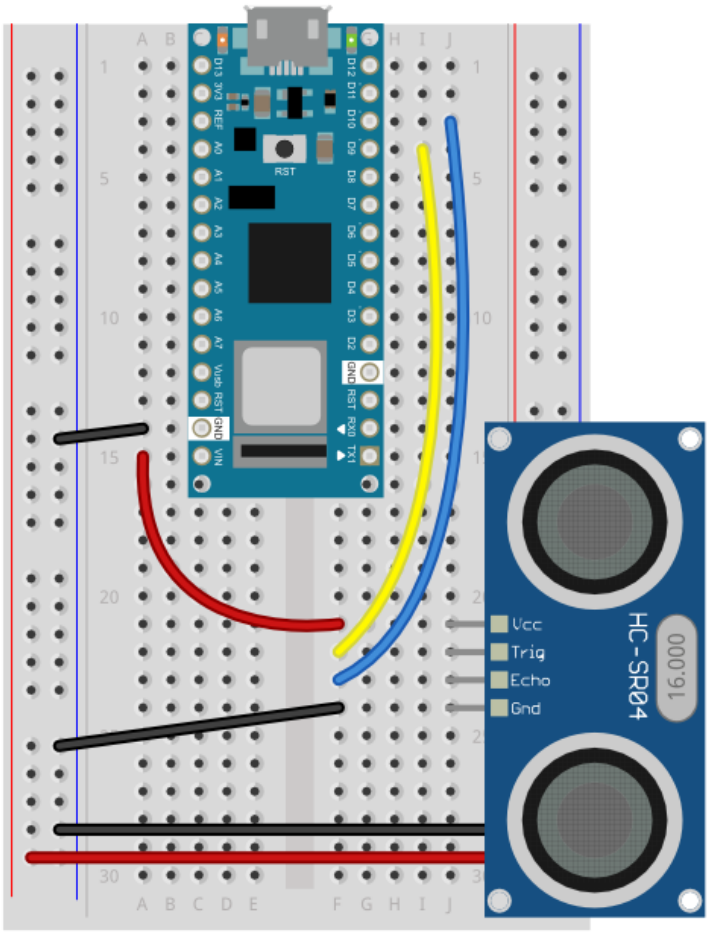
Arduino programming environment

Known issue, action required

The same known issue applies to SR04 which operates at 5 V source while the Arduino Nano 33 IoT supplies 3.3 V. It requires 2 VUSB (or VBUS) pads to be shorted to enable the pin.

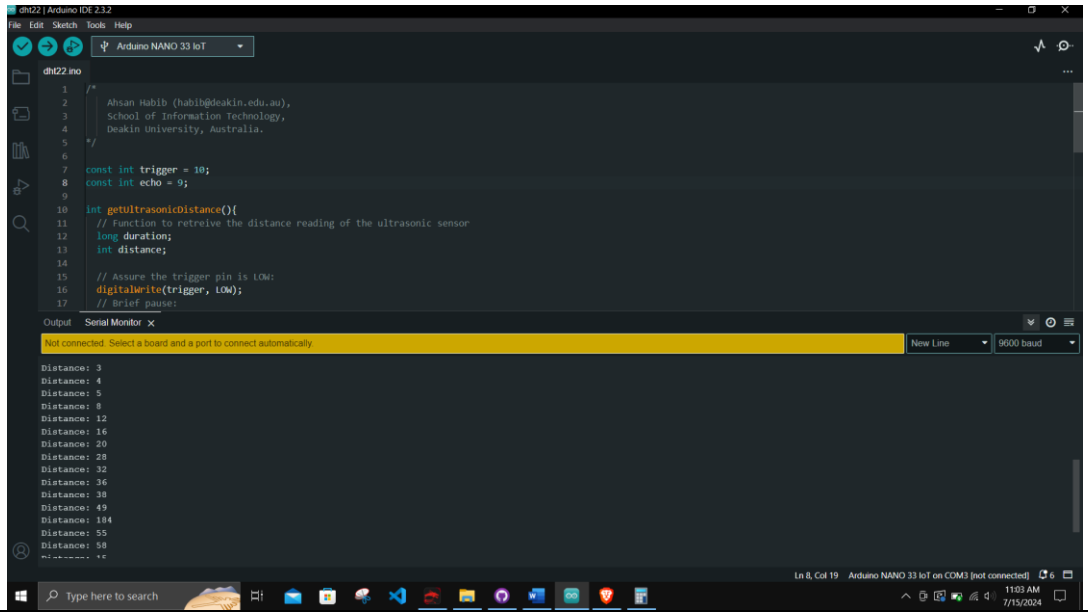
Steps:

Step	Action
1	Connect your HC-SR04 sensor to the Arduino board. Note that the pin layout in the image below may look different than the board you may have.

	
2	Connect your Arduino board to your computer using the USB cable.
3	Write an Arduino sketch (or download it from https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/sketch_hcsr04_distance.ino) which looks like below. Compile the code in Arduino IDE, deploy to the board and observe output in the Arduino IDE serial monitor.

```
const int trigger = 2;
const int echo = 3;

int getUltrasonicDistance(){
  // Function to retreive the distance reading of the ultrasonic
  long duration;
  int distance;

  // Assure the trigger pin is LOW:
  digitalWrite(trigger, LOW);
  // Brief pause:
  delayMicroseconds(5);

  // Trigger the sensor by setting the trigger to HIGH:
  digitalWrite(trigger, HIGH);
  // Wait a moment before turning off the trigger:
  delayMicroseconds(10);
  // Turn off the trigger:
  digitalWrite(trigger, LOW);

  // Read the echo pin:
  duration = pulseIn(echo, HIGH);
  // Calculate the distance in centimeter (CM):
  distance = duration * 0.034 / 2;

  // Uncomment this line to return value in IN instead of CM:
  //distance = distance * 0.3937008

  // Return the distance read from the sensor:
  return distance;
}

void setup() {
  // Define inputs and outputs:
  pinMode(trigger, OUTPUT);
  pinMode(echo, INPUT);

  // Start the serial monitor:
  Serial.begin(9600);
}

void loop() {
  // Print the distance to the serial monitor:
  Serial.print("Distance: ");
  Serial.println(getUltrasonicDistance());

  // Wait one second before continuing:
  delay(1000);
}
```

4	Question: Spec of SR04 is available here (https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf). Identify 2 critical aspects you should be careful about this sensor operation. Answer: The first aspect is to avoid connecting directly to a power supply and if the connection was made, the GND (ground) should be connected first, otherwise the module will malfunction. But since we are using an Arduino, not a raw power source like battery, then we don't need to worry about that. Also, we are establishing the ground and vcc connection before we plug to cable, whatever

	wires were connected first does not really mean anything, just need to keep in mind that the wires are properly connected it its corresponding pins. The second aspect is the surface area and the smoothness of the object we are detecting. The area needs to be no less than 0.5 square meters, and the object should be as smooth as possible, or it will effect to results of measuring.
5	Question: Take a screenshot of your Serial Monitor displaying distance values while you try to generate a periodic motion by moving your hand gradually back and forth towards the sensor. Add the image here. Answer: 

Activity 2.3: Working with sensor - Accelerometer

LSM6DS3 module on the Arduino Nano 33 IoT is an accelerometer and gyroscope sensor.

Hardware Required

Arduino Nano 33 IoT Board (has inbuilt LSM6DS3 module)

USB cable

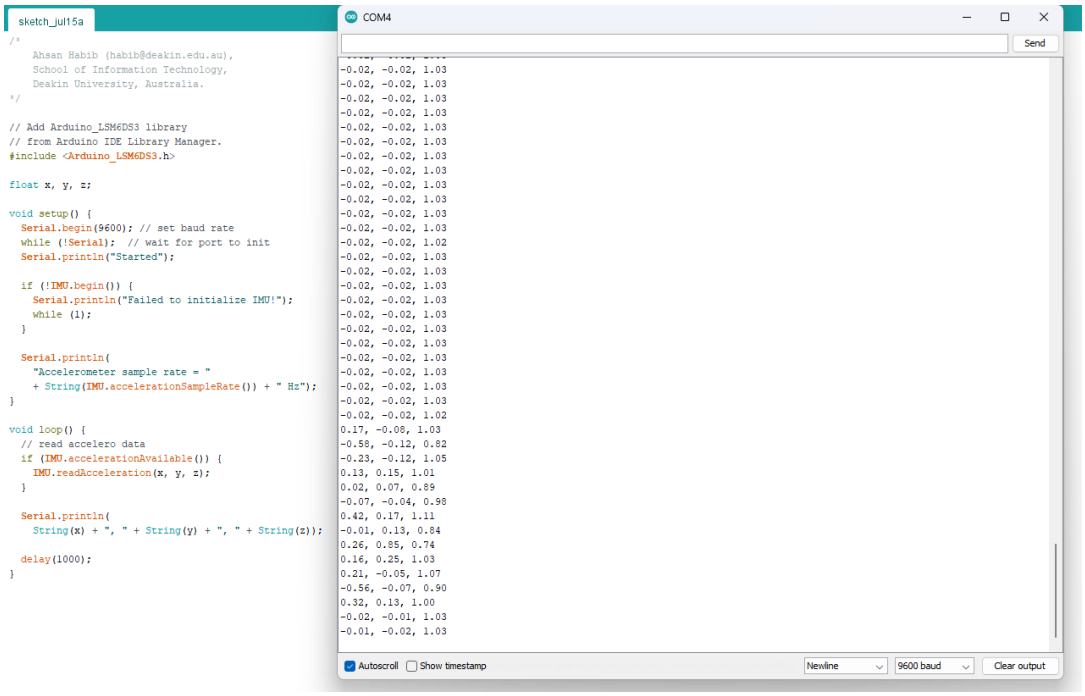
Software Required

Arduino programming environment

Steps:

Step	Action
1	Write Arduino sketch (or download code from https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/sketch_accelero.ino) which looks like below. Compile the code in Arduino IDE, deploy to the board and observe output in the Arduino IDE serial monitor.

	`` // Add Arduino_LSM6DS3 library // from Arduino IDE Library Manager. #include <Arduino_LSM6DS3.h> float x, y, z; void setup() { Serial.begin(9600); // set baud rate while (!Serial); // wait for port to init Serial.println("Started"); if (!IMU.begin()) { Serial.println("Failed to initialize IMU!"); while (1); } Serial.println("Accelerometer sample rate = " + String(IMU.accelerationSampleRate()) + " Hz"); } void loop() { // read accelero data if (IMU.accelerationAvailable()) { IMU.readAcceleration(x, y, z); } Serial.println(String(x) + ", " + String(y) + ", " + String(z)); delay(1000); } ```
2	Question: Spec of LSM6DS3 is available here (https://content.arduino.cc/assets/st_imu_lsm6ds3_datasheet.pdf). Identify at least 3 attributes of this sensor you think important to work with. Answer: The first attribute I found important is the make up of LSM3DS3 where it combines both accelerometer (linear motion) and gyroscope (angular motion, the turning velocity) to detection motion. The second attribute is flexible power consumption. It can perform in low or high-power consumption depending on the requirements of the task. The last attribute is the built-in embedded features. Things like event detection (free-fall, wakeup, tap), pedometer functions (step detector and counter), tilt detection, motion, sensors are all integrated directly into the hardware.
3	Question: Take a screenshot of your Serial Monitor displaying sensor readings. Add the image here.

	Answer: 
4	Question: Identify the max sampling rate and consider reducing the delay (line 37 in the sketch) to increase the number of samples. Summarise your findings here. Answer: The accelerometer data can be sampled at around 6.7 kilosamples per second. In the code, we can change the delay to 1 millisecond to increase the number of samples. The data are updated very fast. https://learn.sparkfun.com/tutorials/lsm6ds3-breakout-hookup-guide/all

Activity 2.4: Plot data using Python Notebook

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. You can find detail in official website (https://matplotlib.org).

Hardware Required

No hardware required

Software Required

Python Jupyter notebook

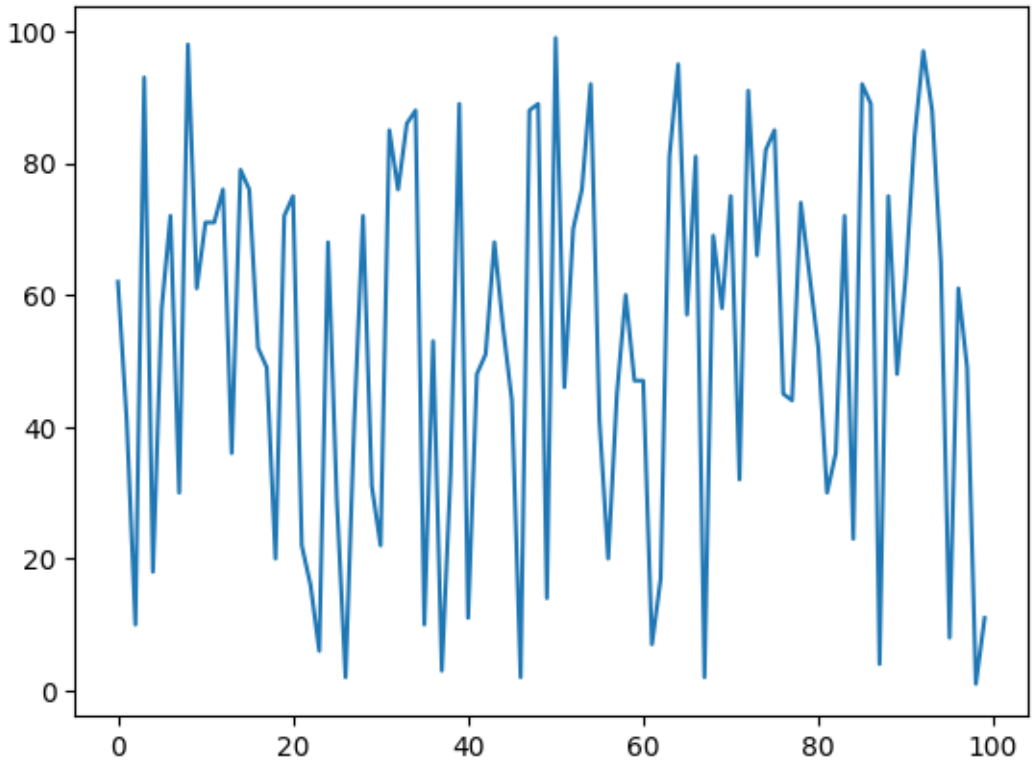
Steps:

Step	Action
1	Download the Jupyter notebook **week2_notebook.ipynb** from here (https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/week2_notebook.ipynb). Follow the instructions in the notebook to carry out instructions and finally convert the notebook to PDF so you can combine it with this activity sheet PDF.

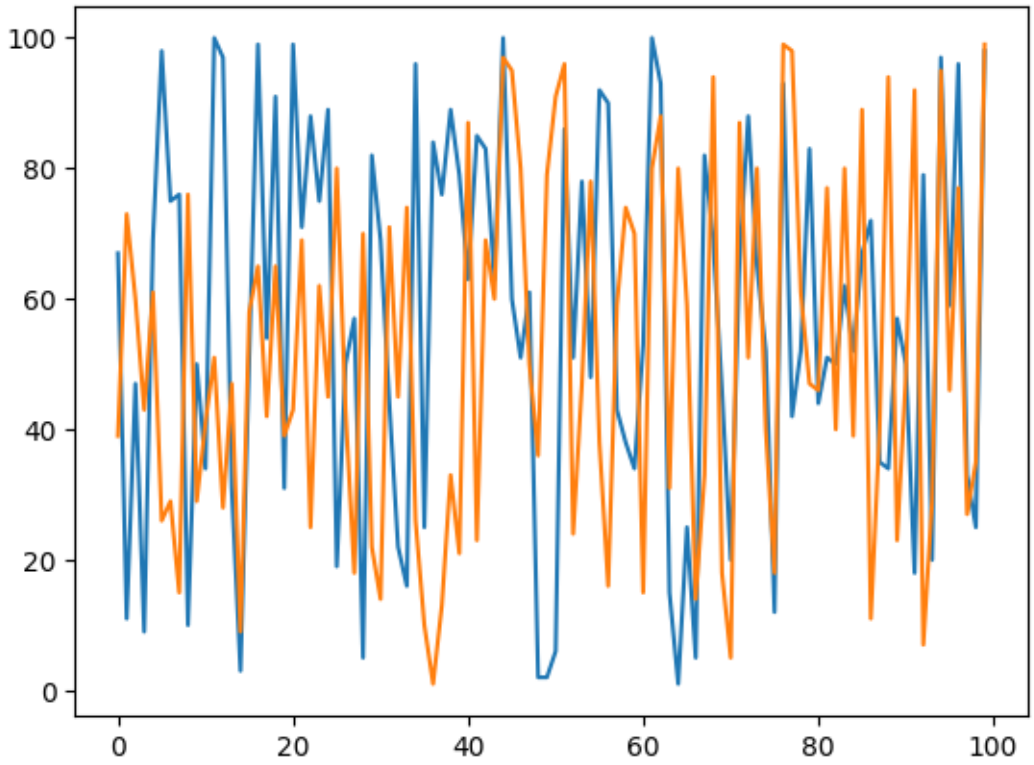
```python
student_name = "Hoang Long Tran"  # fill your name
student_id = "s223128143"  # fill your student ID
print("Student name: " + student_name)
print("Student ID: " + student_id)
```

```
Student name: Hoang Long Tran
Student ID: s223128143
```

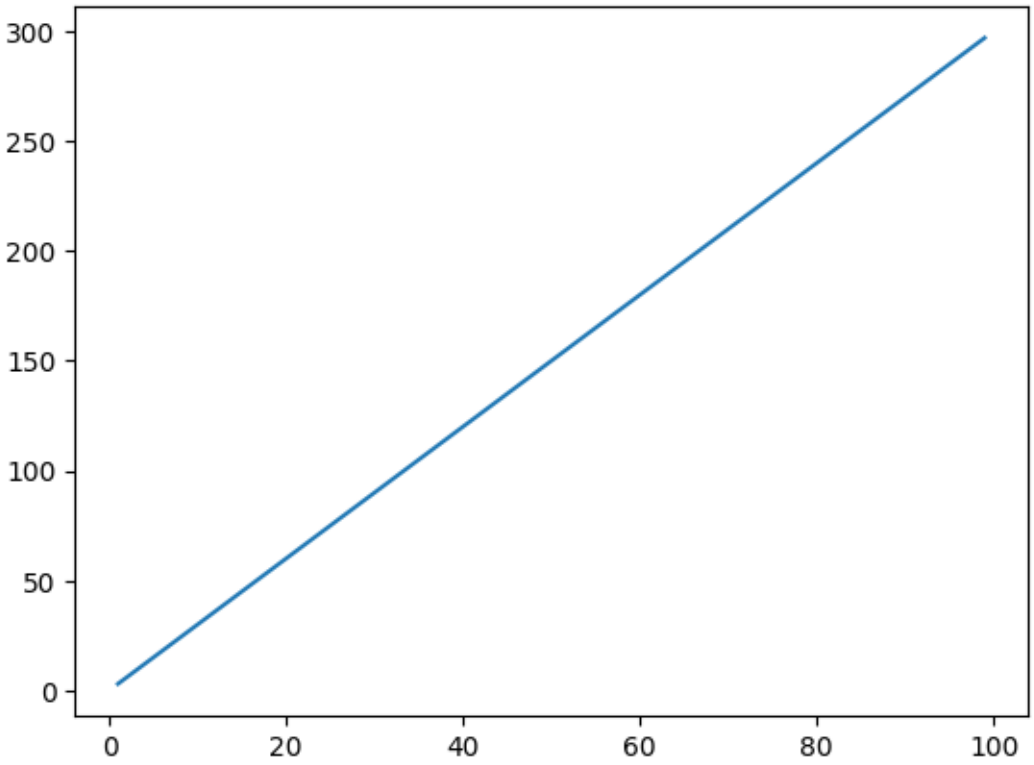
```python
import random
import matplotlib.pyplot as plt

n_values = 100
y_values = []

# Create data (y_values) randomly between 1 and 100.
for i in range(n_values):
    y_values.append(random.randint(1, 100))

x_values = range(n_values)  # X is sequence of values 0-99
plt.plot(x_values, y_values)
plt.show()
```



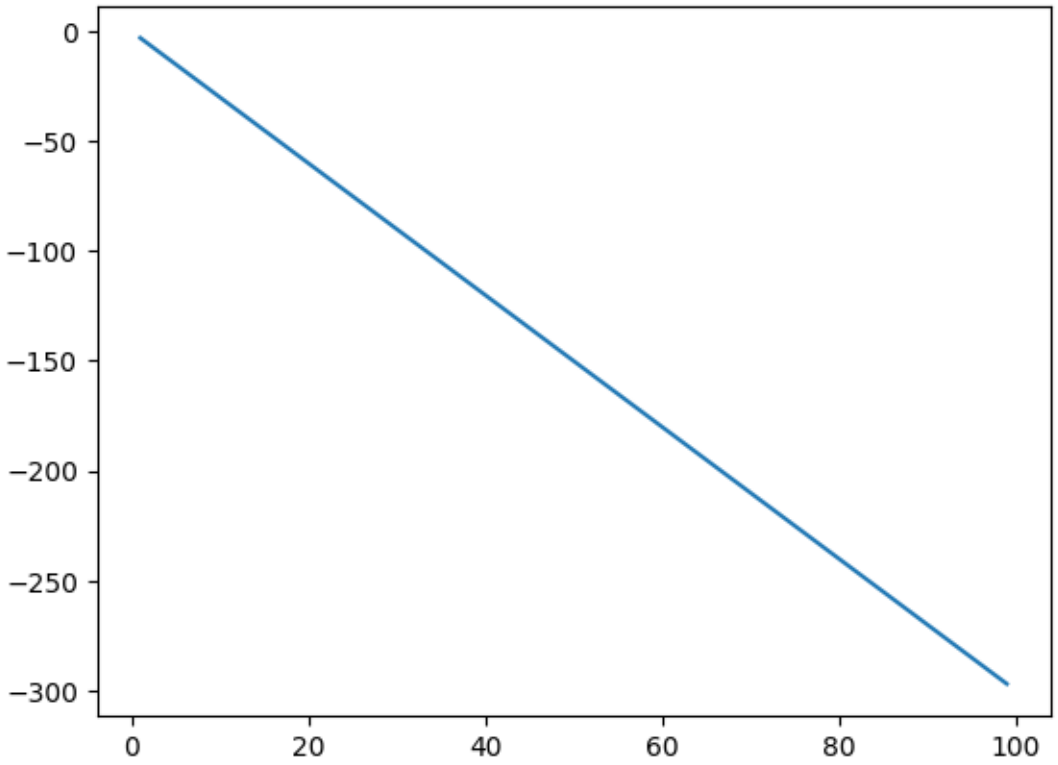

```
# Plot 2 variables
#

n_values = 100
y_values_1 = []
y_values_2 = []

# Create data (y_values) randomly between 1 and 100.
for i in range(n_values):
    y_values_1.append(random.randint(1, 100))
    y_values_2.append(random.randint(1, 100))

x_values = range(n_values)  # X is sequence of values 0-99
plt.plot(x_values, y_values_1)
plt.plot(x_values, y_values_2)  # call plot again draws in the same graph.
plt.show()
```



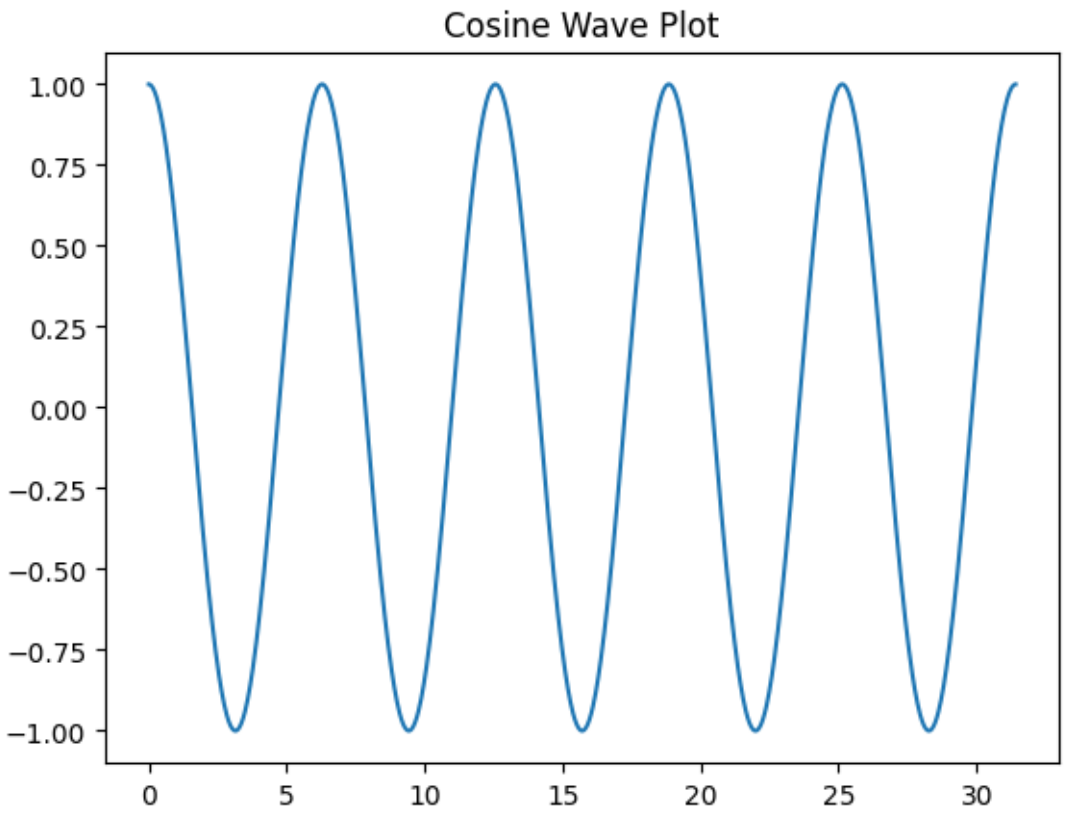

```
#
# Activity 1: Create data so that the plot draws an
# ascending line (y_values increase at any rate).
#

x_values = range(1,100)
y_values = [x*3 for x in x_values]

plt.plot(x_values, y_values)
plt.show()
```




```
#
# Activity 2: Create data so that the plot draws a
# descending line (y_values decrease at any rate).
#

x_values = range(1,100)
y_values = [x*(-3) for x in x_values]

plt.plot(x_values, y_values)
plt.show()
```



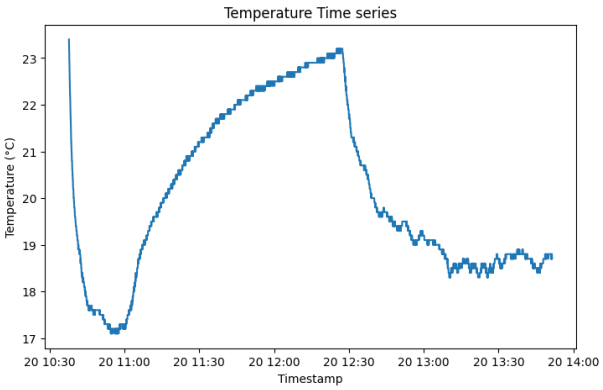

```
#
# Activity 3: Create data so that the plot draws a
# wave. You can consider using Python's math libarary, which has
# a sin function (detail https://www.w3schools.com/python/ref_math_sin.asp).
#

import numpy as np

x_values = np.linspace(start=0, stop=10 * np.pi, num=1000) # num is the number of samples to
y_values = np.cos(x_values)

plt.plot(x_values, y_values)
plt.title('Cosine Wave Plot')
plt.show()
```

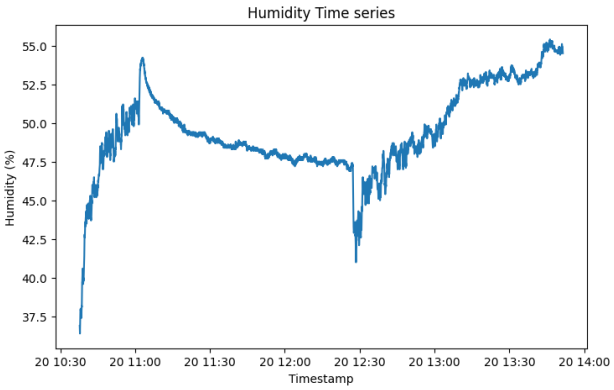

Cosine Wave Plot
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30

Weekly activity.

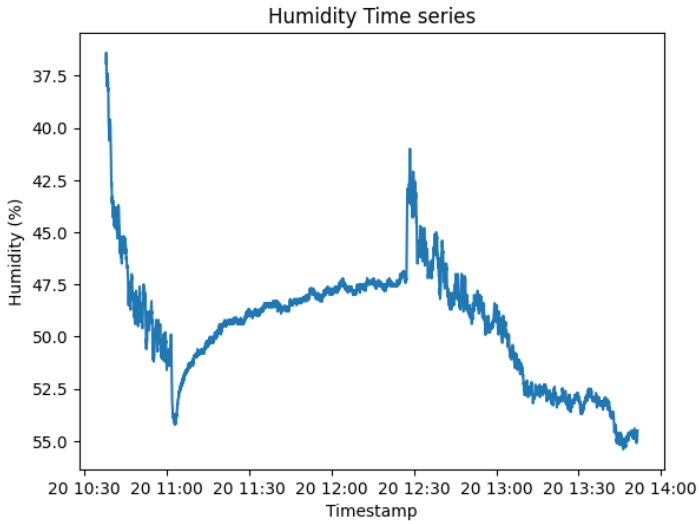
Q2.

I have used the DHT22 to record the temperature and humidity data in my room from around 10:30AM to 2PM. In the Temperature plot, initially, the temperature is quite high, since that was when I just turned off the heating and opened the window. Then the temperature started to drop from 23 to 17 degrees. After which, I felt cold, so I closed the window and turned on the heating. Then at around 12:30AM, I felt hot again, so I turned off the heating and opened the window, so the Temperature once again dropped. The humidity data seems to be reverse of the temperature data. So, when the temperature is high, the humidity is low, and vice versa when the temperature is low, the humidity is high.





Here is a plot of Humidity when I reverse the y-axis.



Q3.

In the Arduino code, I just record the temperature and humidity. I set the delay to 2 seconds, which is the minimum interval DHT22 can update the data.

```ino
#include <DHT.h>

#define DHTPIN 10  // digital pin number
#define DHTTYPE DHT22  // DHT type 11 or 22
DHT dht(DHTPIN, DHTTYPE);

// variables to store data.
float hum, temp;

void setup() {
  // Set baud rate for serial communication
  Serial.begin(9600);

  // initialise DHT libarary
  dht.begin();
}

void loop() {
  // read data
  hum = dht.readHumidity();
  temp = dht.readTemperature();

  // Print data to serial port - a compact way
  Serial.println(String(hum) + "," + String(temp));

  // wait 2 seconds before updating the data
  delay(2000);
}
```

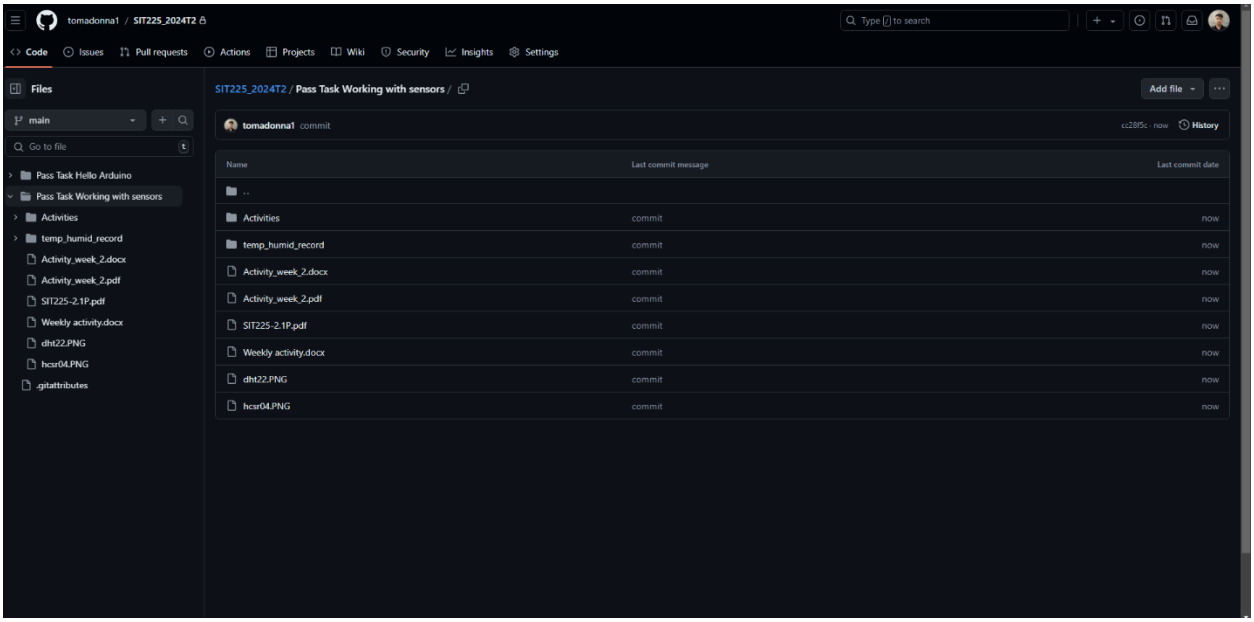
Python script is used to take the DHT22 data from Arduino, add a timestamp for each row of data, and write it in a csv file.

```
import serial
import time
from datetime import datetime
import os

# Function to get the current time
def timestamp():
    return datetime.now().strftime('%Y%m%d%H%M%S')

# Serial port and saving csv file in desire destination
ser = serial.Serial('COM4', 9600)
filename = os.path.join(r'C:\Users\tomde\OneDrive\Documents\Deakin\Deakin-Data-Science\T1Y2\SIT225 - Data Capture Technologies\Week 2 - Working with Sensors in Arduino\2.1P\temp_humid_record', 'dht22.csv')

try:
    while True:
        # Check if data is waiting in serial buffer
        if ser.in_waiting > 0:
            data = ser.readline().decode('utf-8').strip() # read data from serial port and decode it
            formatted_data = f"{timestamp()}, {data}"

            # Add data to csv file
            with open(filename, 'a') as file:
                file.write(formatted_data + '\n')

            print(f"{formatted_data}")

        time.sleep(1)

except KeyboardInterrupt:
    print("Forced stop by user.")

finally:
    ser.close()
    print("Serial port closed.")
```

Q4.

https://www.youtube.com/watch?v=TMYK64tUzOI

Q5.



https://github.com/tomadonna1/SIT225_2024T2/tree/main/Pass%20Task%20Working%20with%20sensors

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv("dht22 data.csv", header=None, names=['Timestamp', 'Humidity', 'Temperature
df.head()
```

	Timestamp	Humidity	Temperature
0	2024-07-20 10:37:47.756	36.9	23.4
1	2024-07-20 10:37:49.765	36.8	23.3
2	2024-07-20 10:37:51.771	36.6	23.2
3	2024-07-20 10:37:53.778	36.4	23.0
4	2024-07-20 10:37:55.784	36.6	23.0

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5791 entries, 0 to 5790
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Timestamp    5791 non-null   object
 1   Humidity     5791 non-null   float64
 2   Temperature  5791 non-null   float64
dtypes: float64(2), object(1)
memory usage: 135.9+ KB
```

```
# Convert 'Timestamp' feature to correct dtype
df.Timestamp = pd.to_datetime(df.Timestamp)
df.dtypes
```

```
Timestamp      datetime64[ns]
Humidity              float64
Temperature           float64
dtype: object
```

```
df
```

	Timestamp	Humidity	Temperature
0	2024-07-20 10:37:47.756	36.9	23.4
1	2024-07-20 10:37:49.765	36.8	23.3
2	2024-07-20 10:37:51.771	36.6	23.2
3	2024-07-20 10:37:53.778	36.4	23.0
4	2024-07-20 10:37:55.784	36.6	23.0
...	...	...	...
5786	2024-07-20 13:51:19.231	54.9	18.8
5787	2024-07-20 13:51:21.239	54.8	18.8
5788	2024-07-20 13:51:23.245	54.7	18.8
5789	2024-07-20 13:51:25.252	54.6	18.7
5790	2024-07-20 13:51:27.259	54.5	18.7

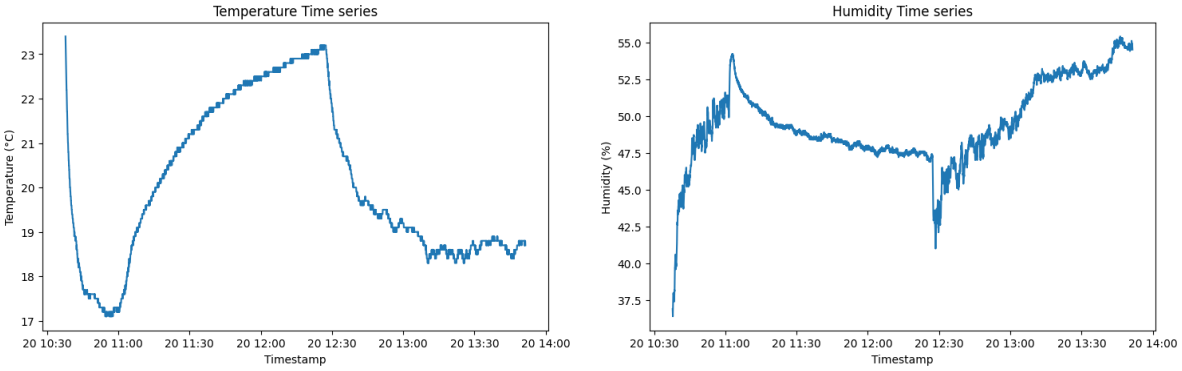
```
# set 'Timestamp' as index
df2 = df.copy()
df2.set_index('Timestamp', inplace=True)
df2.head(3)
```

Timestamp	Humidity	Temperature
2024-07-20 10:37:47.756	36.9	23.4
2024-07-20 10:37:49.765	36.8	23.3
2024-07-20 10:37:51.771	36.6	23.2

```
plt.figure(figsize=(18, 5))

# Plot the time series plot of Temp
plt.subplot(1, 2, 1)
plt.plot(df2.index, df2.Temperature)
plt.ylabel('Temperature (°C)')
plt.xlabel('Timestamp')
plt.title('Temperature Time series')

# Plot the time series plot of Humid
plt.subplot(1, 2, 2)
plt.plot(df2.index, df2.Humidity)
plt.ylabel('Humidity (%)')
```

```
plt.xlabel('Timestamp')
plt.title('Humidity Time series')
```

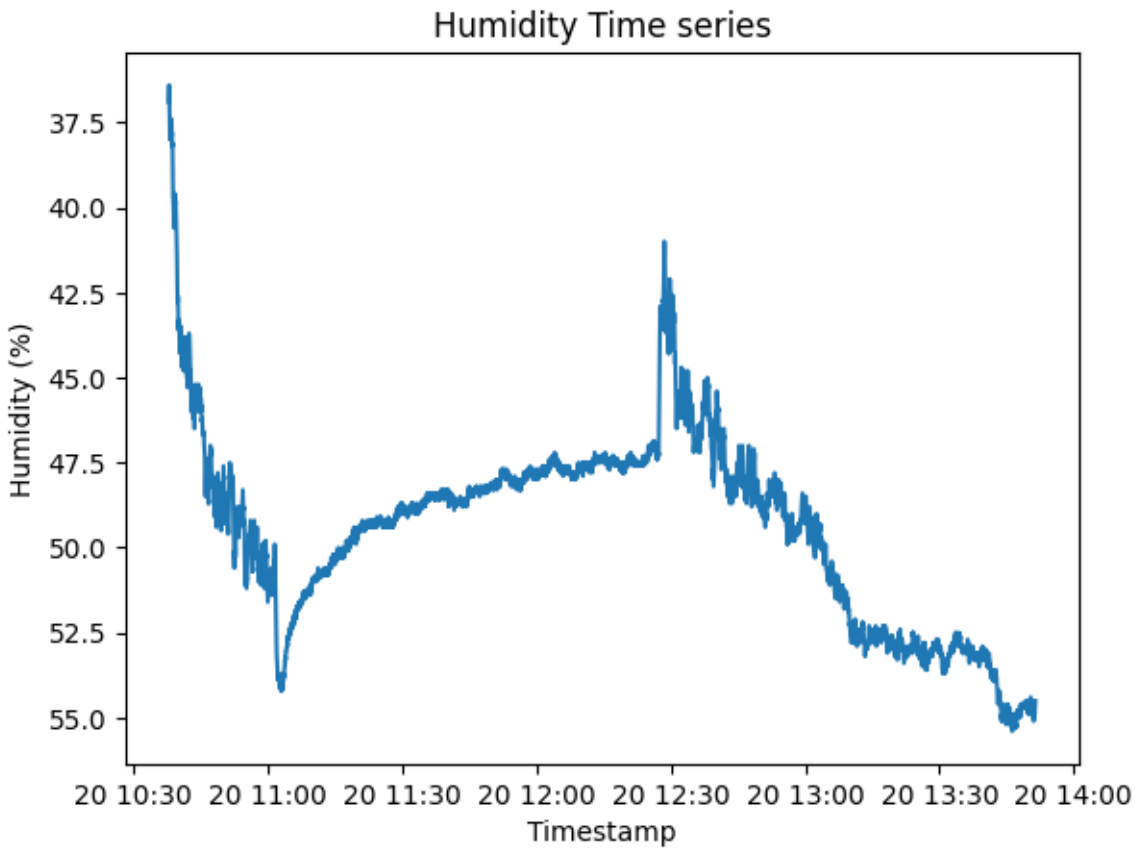
```
Text(0.5, 1.0, 'Humidity Time series')
```



I have used the DHT22 to record the temperature and humidity data in my room from around 10:30AM to 2PM. In the Temperature plot, initially, the temperature is quite high, since that was when I just turned off the heating and opened the window. Then the temperature started to drop from 23 to 17 degrees. After which, I felt cold, so I closed the window and turned on the heating. Then at around 12:30AM, I felt hot again, so I turned off the heating and opened the window, so the Temperature once again dropped. The humidity data seems to be reverse of the temperature data. So, when the temperature is high, the humidity is low, and vice versa when the temperature is low, the humidity is high.

```
plt.plot(df2.index, df2.Humidity)
plt.gca().invert_yaxis()
plt.ylabel('Humidity (%)')
plt.xlabel('Timestamp')
plt.title('Humidity Time series')
```

```
Text(0.5, 1.0, 'Humidity Time series')
```



This plot reverse on the y-axis, we can see that the humid is reverse of temperature.