PROJECT WORK
INDOOR WEATHER STATION

STUDENT: ILIASOV USONBEK
SUBJECT COORDINATOR: KRISTÓF LAJBER

CONTENT

COLLECTING THE PARTS

The parts required to build this project are:

- An Arduino Uno
- A ProtoShield Mini Breadboard
- A DHT22 Temperature and Humidity Sensor
- A DS1307 RTC (real time clock)
- A Coin Cell Battery (CR1220)
- An SD Card Module
- An SD Card
- M-M Jumper wires
- M-F Jumper wires
- A USB cord
- A Housing to put it in (model attached)

https://www.hestore.hu/index.php

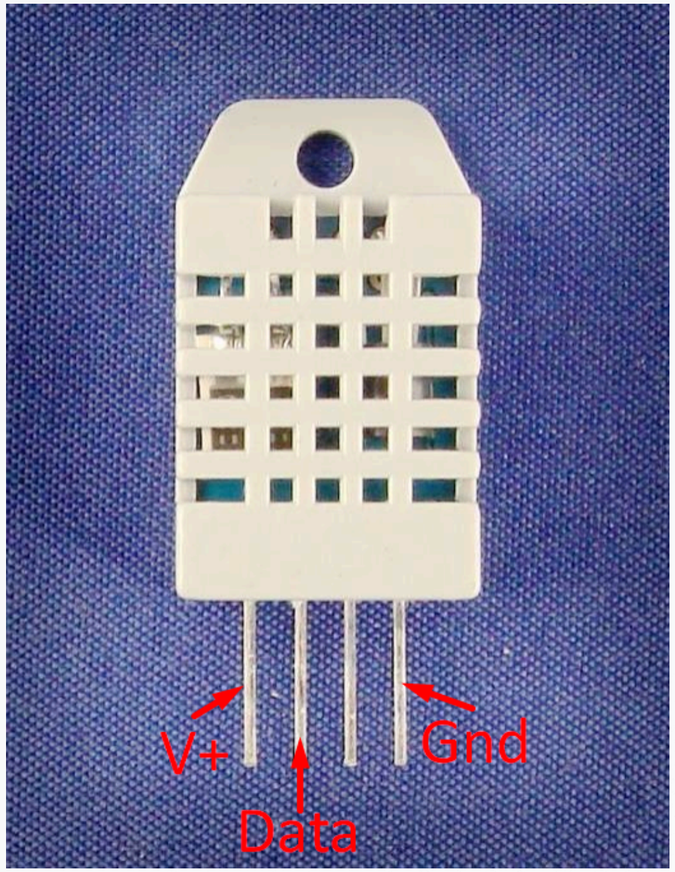
HOW TO CONNECT MODULES

Wiring it up

1. There are a few different components in this project. We found that the best way to assemble it is to first put the ProtoShield on top of the Arduino, and then stick the mini breadboard to the top of the ProtoShield, then wire up the different parts one at a time. We will break down the wiring of the different parts separately for clarity.

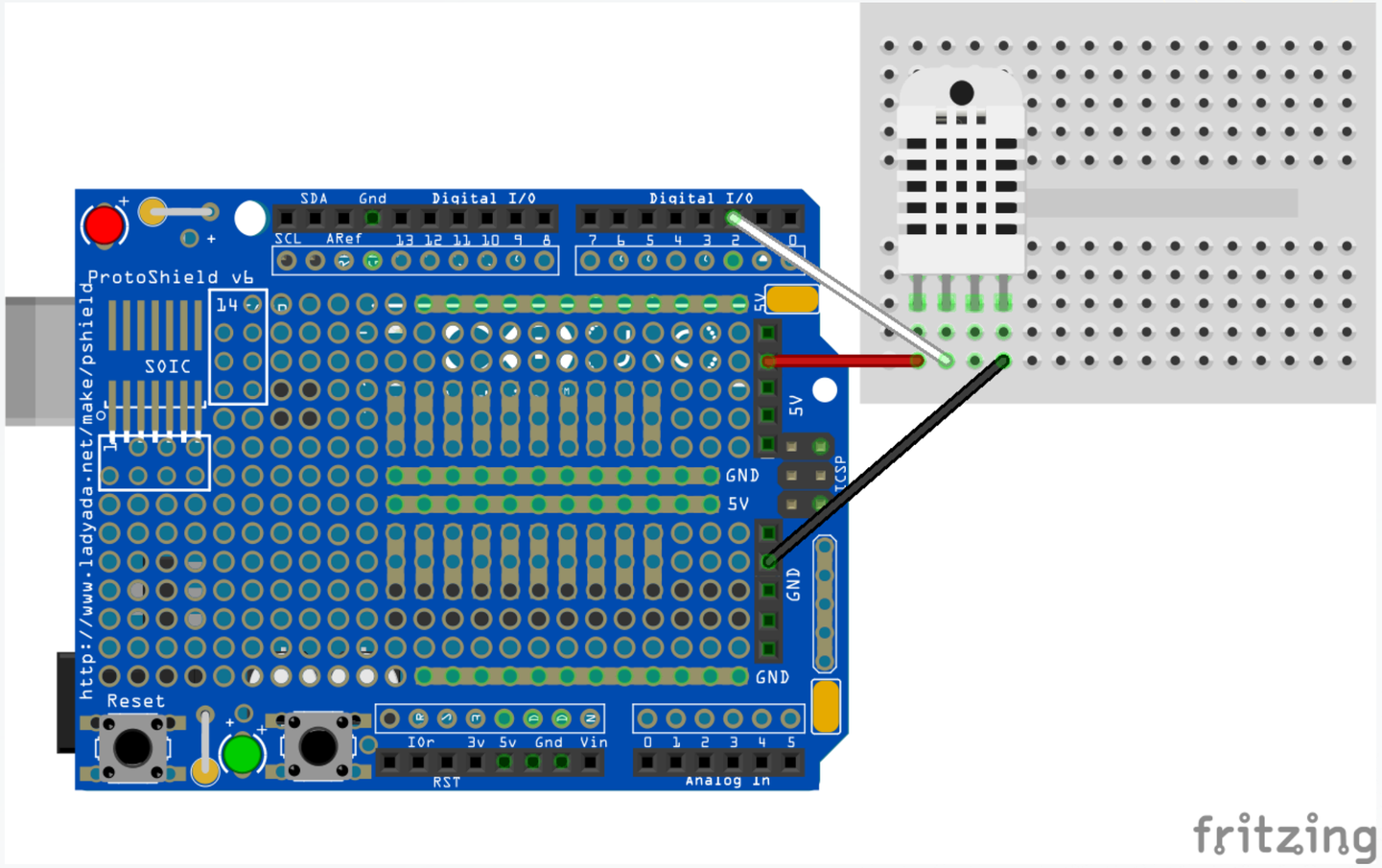
Connect the DHT22

1. This is the temperature sensor and humidity sensor of the weather station. The pins on it aren't marked, see the below photo for the pinout.



HOW TO CONNECT MODULES

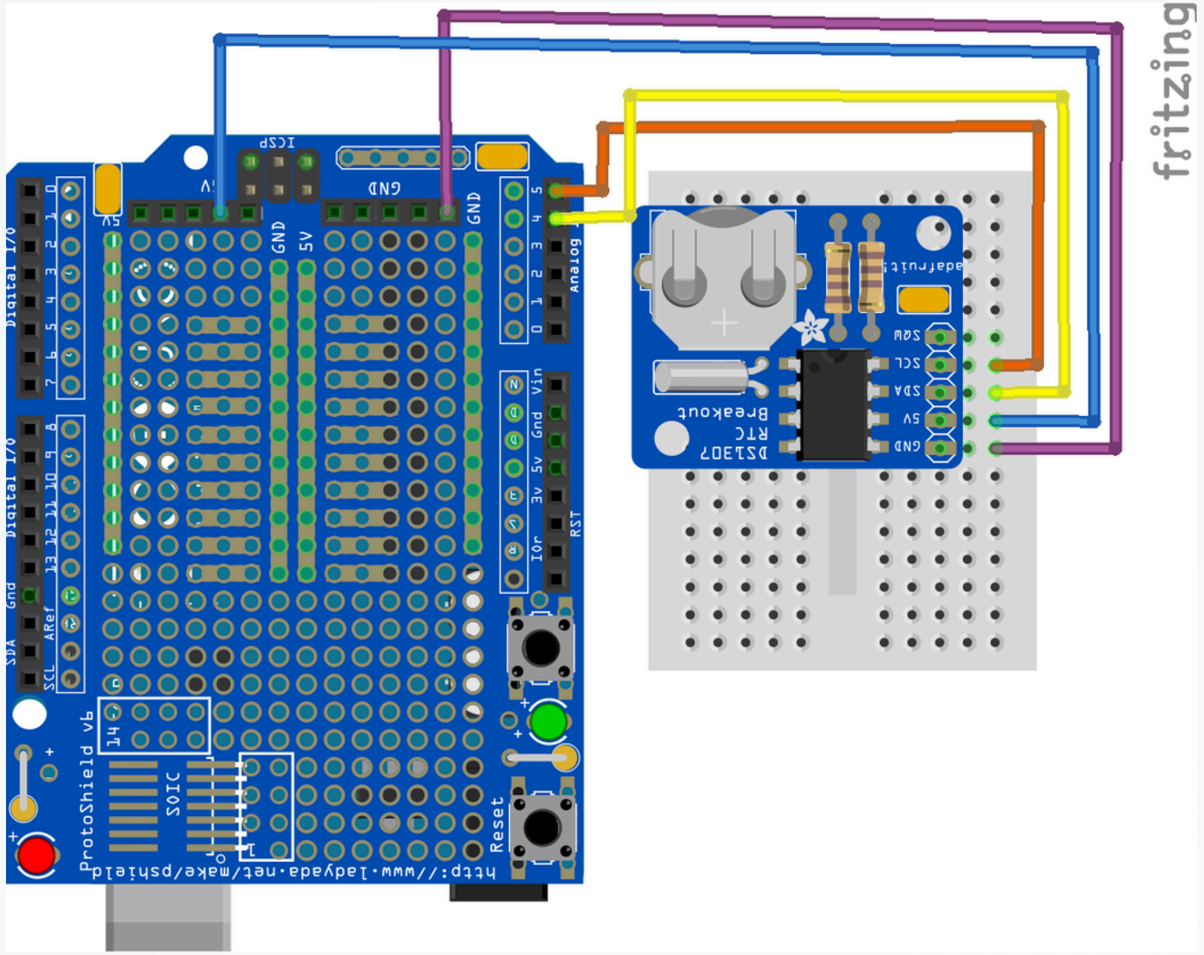
Connect the power and ground to the 5V and GND rails on the ProtoShield. The data line on the DHT22 will connect to Digital Pin 2 on the shield.



HOW TO CONNECT MODULES

Connect the RTC

Connect the 5V and GND to the respective rails on the ProtoShield, connect the SDA pin to Analog pin 4, SCL to Analog Pin 5.

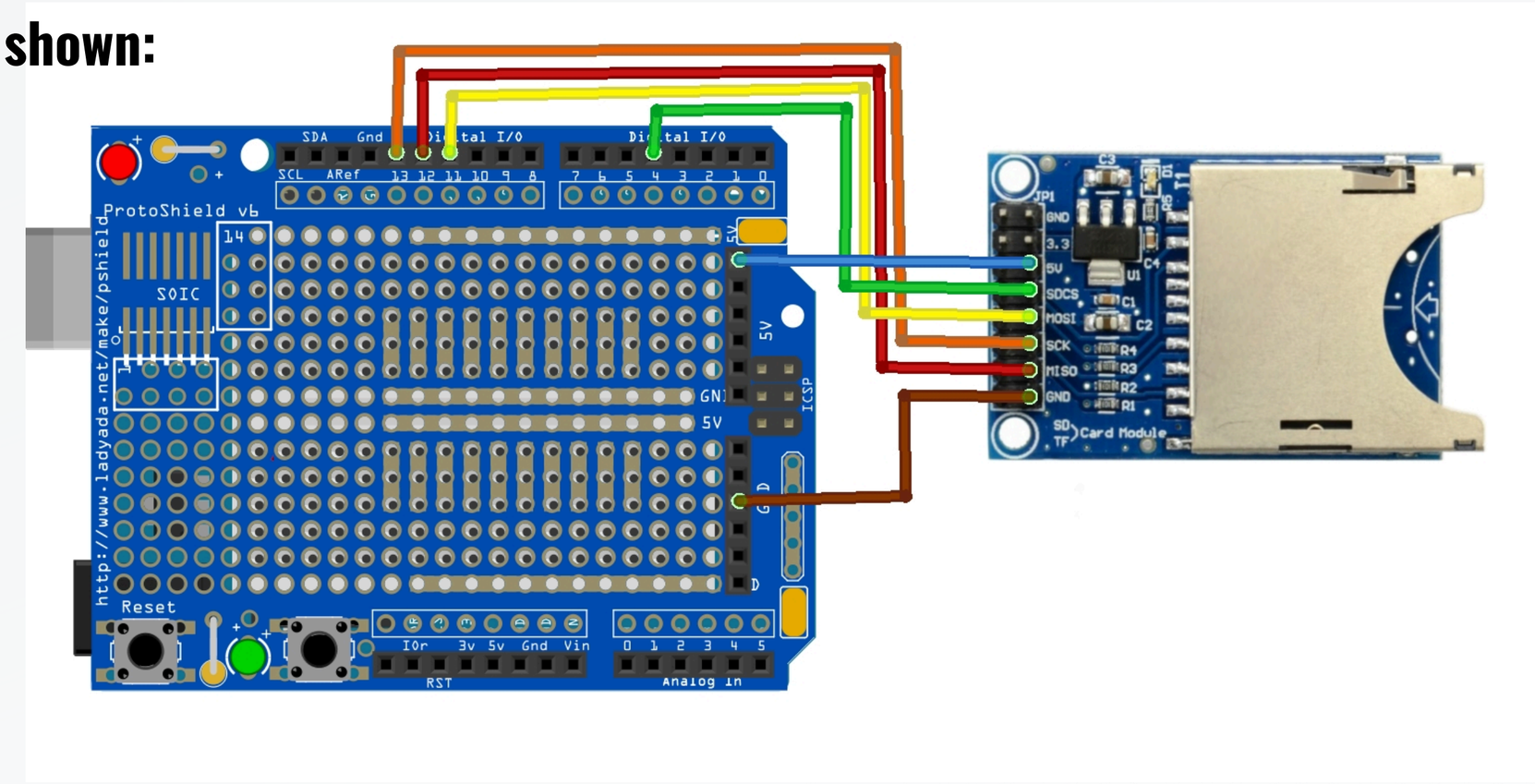


HOW TO CONNECT MODULES

Connect the SD Card Module

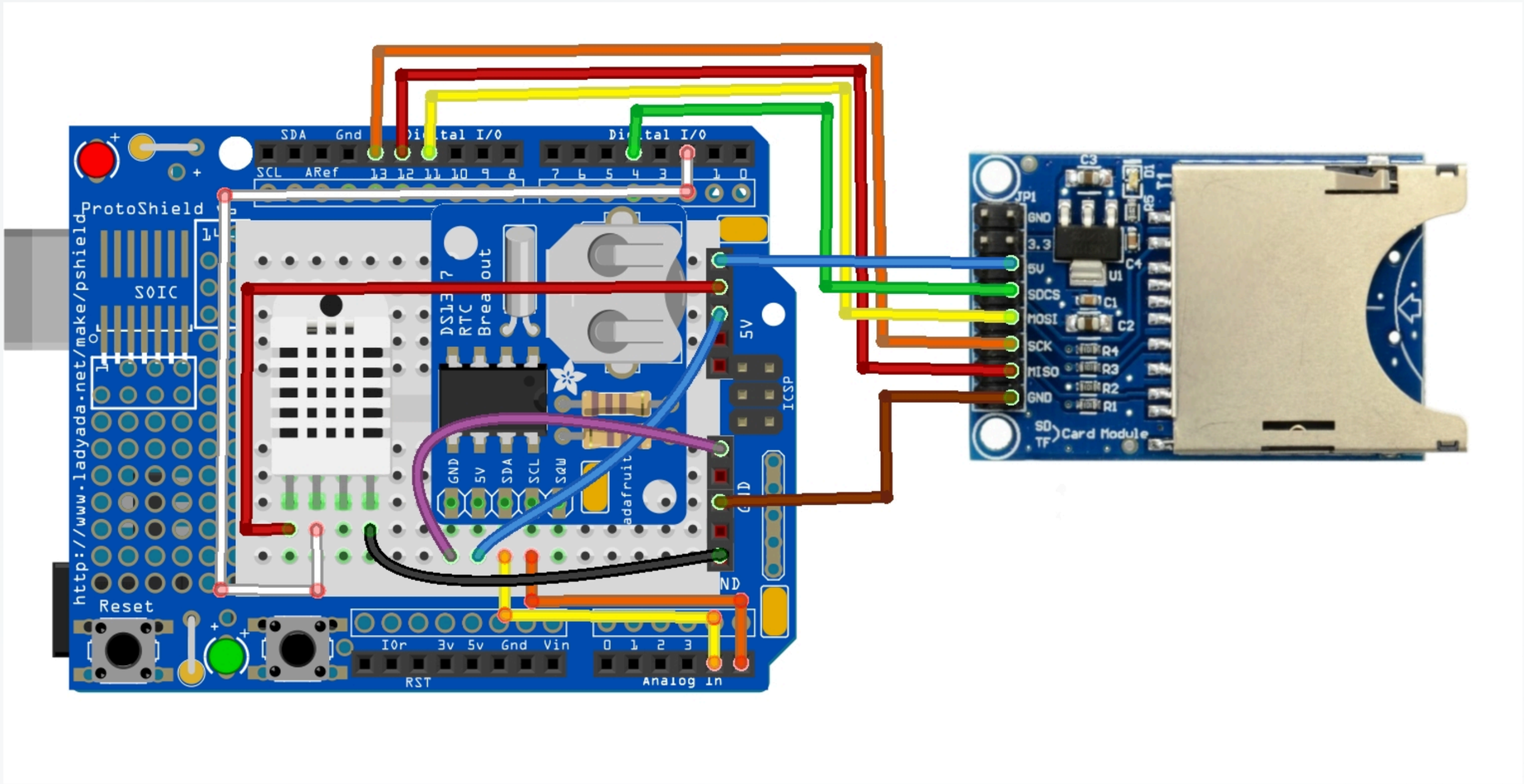
Connect the SD Card Module as shown:

- **Arduino 13 > SCK**
- **Arduino 12 > MISO**
- **Arduino 11 > MOSI**
- **Arduino 4 > SDCS**
- **Arduino 5V > 5V**
- **Arduino GND > GND**



HOW TO CONNECT MODULES

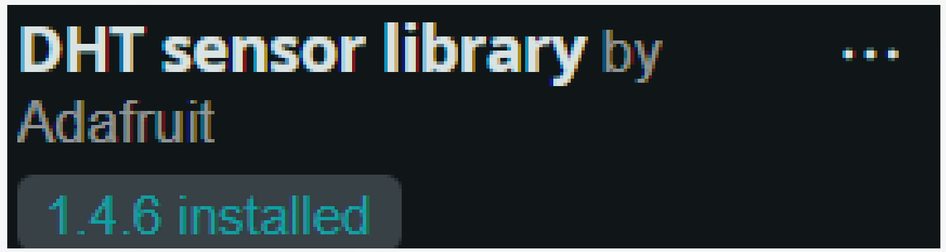
All Together

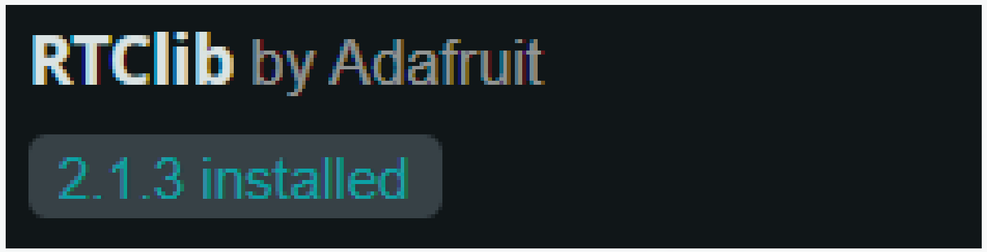


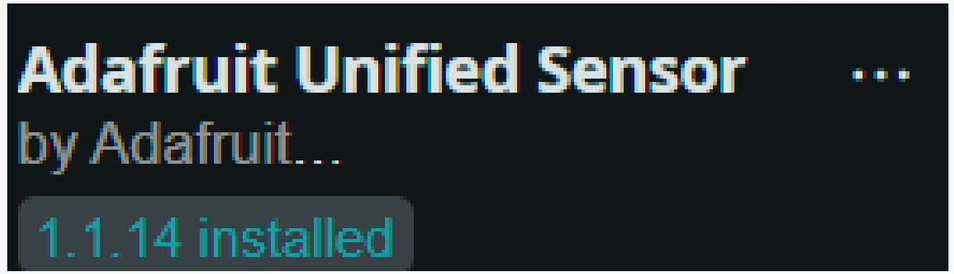
SETUP THE IDE

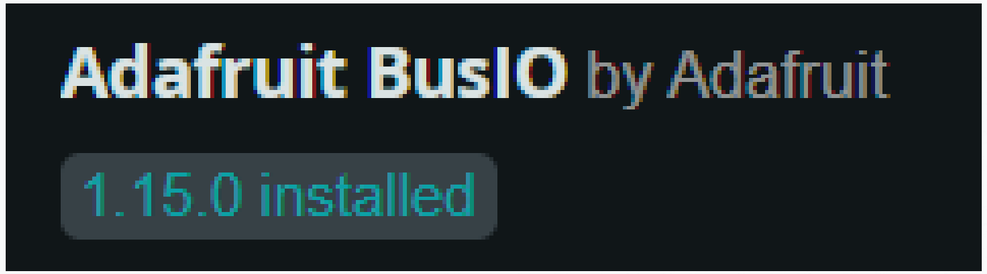
Now its time to load some code onto the newly built weather station! We will be programming with the Arduino IDE (see our Arduino IDE Tutorial). For this code to work we will need to install a few libraries. Here is a full explanation of how to add libraries to Arduino.

The libraries that we need to install are the DHT library, RTC library, and Adafruit Unified Sensor library. Using the 'manage libraries' option in the IDE search for "DHT" to find the DHT sensor library shown below, then select install.









SETUP THE IDE

Here is the code that we will be using for the weather station. We'll take a look at the whole thing, then break it down into parts and explain how it all works.

In this first section of code we include the libraries that we are using, and tell the program what sensors we are using.

WeatherStation.ino | test.ino

```
1   // include relevant libraries for the Arduino, sensor, and Real-time clock (RTC)
2   #include <DHT.h>
3   #include <DHT_U.h>
4   #include <SPI.h>
5   #include <SD.h>
6   #include <Wire.h>
7   #include <RTClib.h>
8
9   // Define DHT pin
10  #define DHTPIN 2        // Pin the sensor is connected to on Arduino (can be any number from 2 - 13)
11  #define DHTTYPE DHT22   // DHT22 is the type of temperature sensor we are using
12
13  File myFile;
14
15  // Initialize DHT sensor for normal 16MHz Arduino
16  DHT dht(DHTPIN, DHTTYPE);
17  RTC_DS1307 RTC;
```

SETUP THE IDE

First of all, we create a new data file if none exists. We want to store all our data in a text file on the SD card so we can easily read it later. We also check to see that the RTC is running. If it's not yet running then we set the time on the RTC to whenever the code was compiled. This only works the first time that the code is loaded. The RTC will stay running for about 5 years on a single battery (wow!).

```
23  void setup() {   // setup function runs once
24    Serial.begin(9600); // Opens serial connection, sets the data rate to 9600 bps (bits pe
25    while (!Serial) {
26      ; // Wait for serial port to connect. Needed for native USB port only
27    }
28    Serial.print("Initializing SD card...");
29    if (!SD.begin(4)) {
30      Serial.println("Initialization failed!");
31      while (1);
32    }
33    Serial.println("Initialization done.");
34    // Create a new data file if one does not already exist
35    if (!SD.exists("DATA.txt")) {
36      myFile = SD.open("DATA.txt", FILE_WRITE);
37      myFile.print("DATE, TIME, TEMP *C, HUMIDITY");
38      myFile.println();
39      myFile.close();
40    }
41    // Initializing the DHT and RTC
42    dht.begin();
43    Wire.begin();
44    RTC.begin();
45    // Check to see if the RTC is keeping time
46    if (!RTC.begin()) {
47      Serial.println("RTC is NOT running!");
48      // Sets the RTC to the time that this sketch was compiled
49      RTC.adjust(DateTime(F(__DATE__), F(__TIME__)));
50    }
51  }
52
```

SETUP THE IDE

The next part of the code is the main loop. This part of the code runs continuously whenever the Arduino is powered. We read the RTC, the temperature and humidity and save them as variables to be used later.

```
53  void loop() {
54    DateTime now = RTC.now(); // Query the RTC for the current time
55    // Read and store the temperature and humidity
56    float t = dht.readTemperature();
57    float h = dht.readHumidity();
58    if (isnan(t)) {
59      Serial.println("Failed to read from DHT sensor!");
60      return;
61    }
62    // Check if temperature or humidity exceeds the threshold
63    if (t > TEMPERATURE_THRESHOLD || h > HUMIDITY_THRESHOLD) {
64      // Alert mechanism - Blink LED, send notification, etc.
65      Serial.println("ALERT: Temperature or humidity exceeded threshold!");
66      // Add your alert mechanism here
67    }
68
69    // Write the current date, time, temperature, and humidity to SD card
70    myFile = SD.open("DATA.txt", FILE_WRITE);
71    myFile.print(now.day(), DEC);
72    myFile.print('/');
73    myFile.print(now.month(), DEC);
74    myFile.print('/');
75    myFile.print(now.year(), DEC);
76    myFile.print(',');
77    myFile.print(' ');
78    myFile.print(now.hour(), DEC);
79    myFile.print(':');
```

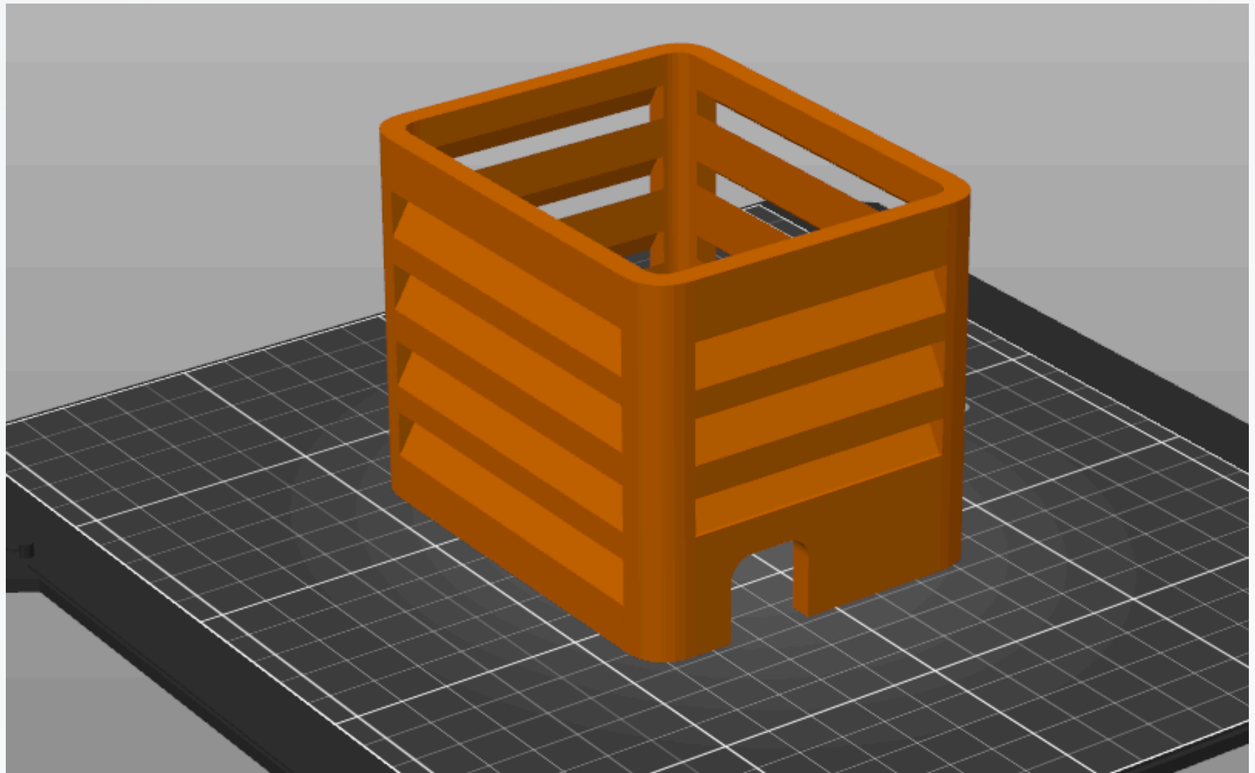
SETUP THE IDE

At the very end of the loop, we have a delay. This pauses the program for a bit so we don't have constant readings. An Arduino runs very fast and can take many readings every second. This slows the program down to save data and to make the data more meaningful. The temperature does not change every minute, let alone every second, there is no need to take superfast measurements.

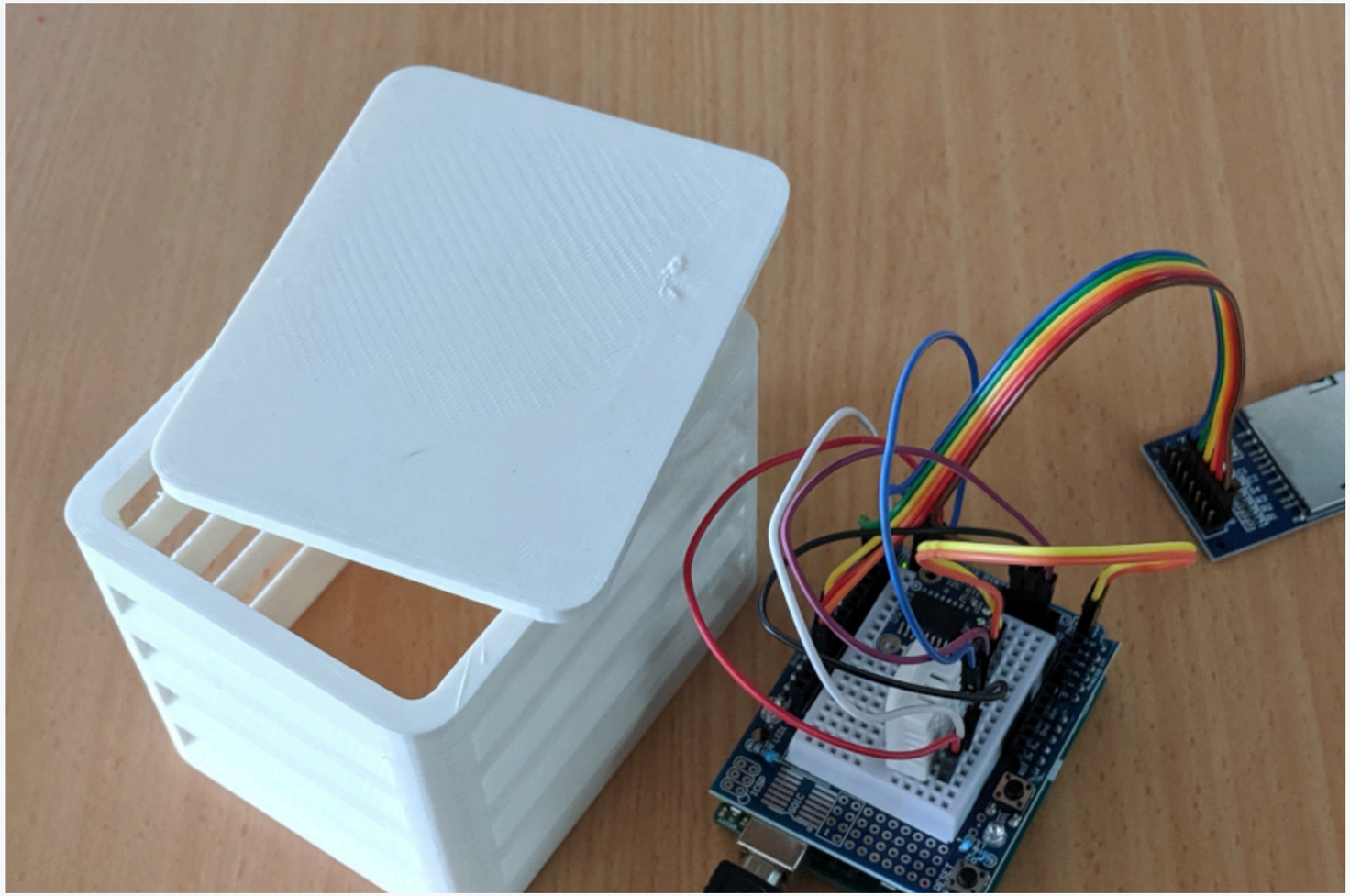
```
  // Write the current date, time, temperature, and humidity to the serial monitor
  Serial.print(now.day(), DEC);
  Serial.print('/');
  Serial.print(now.month(), DEC);
  Serial.print('/');
  Serial.print(now.year(), DEC);
  Serial.print(',');
  Serial.print(' ');
  Serial.print(now.hour(), DEC);
  Serial.print(':');
  if (now.minute() < 10) {
    Serial.print('0');
    Serial.print(now.minute(), DEC);
  }
  else {
    Serial.print(now.minute(), DEC);
  }
  Serial.print(',');
  Serial.print(' ');
  Serial.print("Temp: ");
  Serial.print(t);
  Serial.print(" *C");
  Serial.print(',');
  Serial.print(' ');
  Serial.print("Hum: ");
  Serial.print(h);
  Serial.println();

  delay(10000); // Wait time until the device reads the temperature again
}
```

MAKE HOUSING

In order for the weather station to take readings, it needs to be out in the elements! We obviously need to keep it dry somehow though, so that's where this 3D printed housing comes in handy. If you don't have access to a 3D printer, a plastic or Tupperware container with vents cut in the sides will work just fine.



MAKE HOUSING



CHALLENGES AND ACHIEVEMENTS

Challenges Faced

- Component Availability: Ensuring all required components were obtained.
- Sensor Integration: Connecting and calibrating multiple sensors.
- Software Development: Developing data

Achievements

- Precision Monitoring: Accurate readings with DHT22 sensor.
- Real-Time Accessibility: Immediate adjustments based on data.
- Data Logging: Successful storage and trend analysis.
- Customizable Alerts: Enhanced safety with predefined thresholds.

THANK'S FOR WATCHING