WEATHERCREW

YRKESHÖGSKOLAN NOVIA

IoT Weather station

By Antonin, Bryan, Chantal and Job

Index

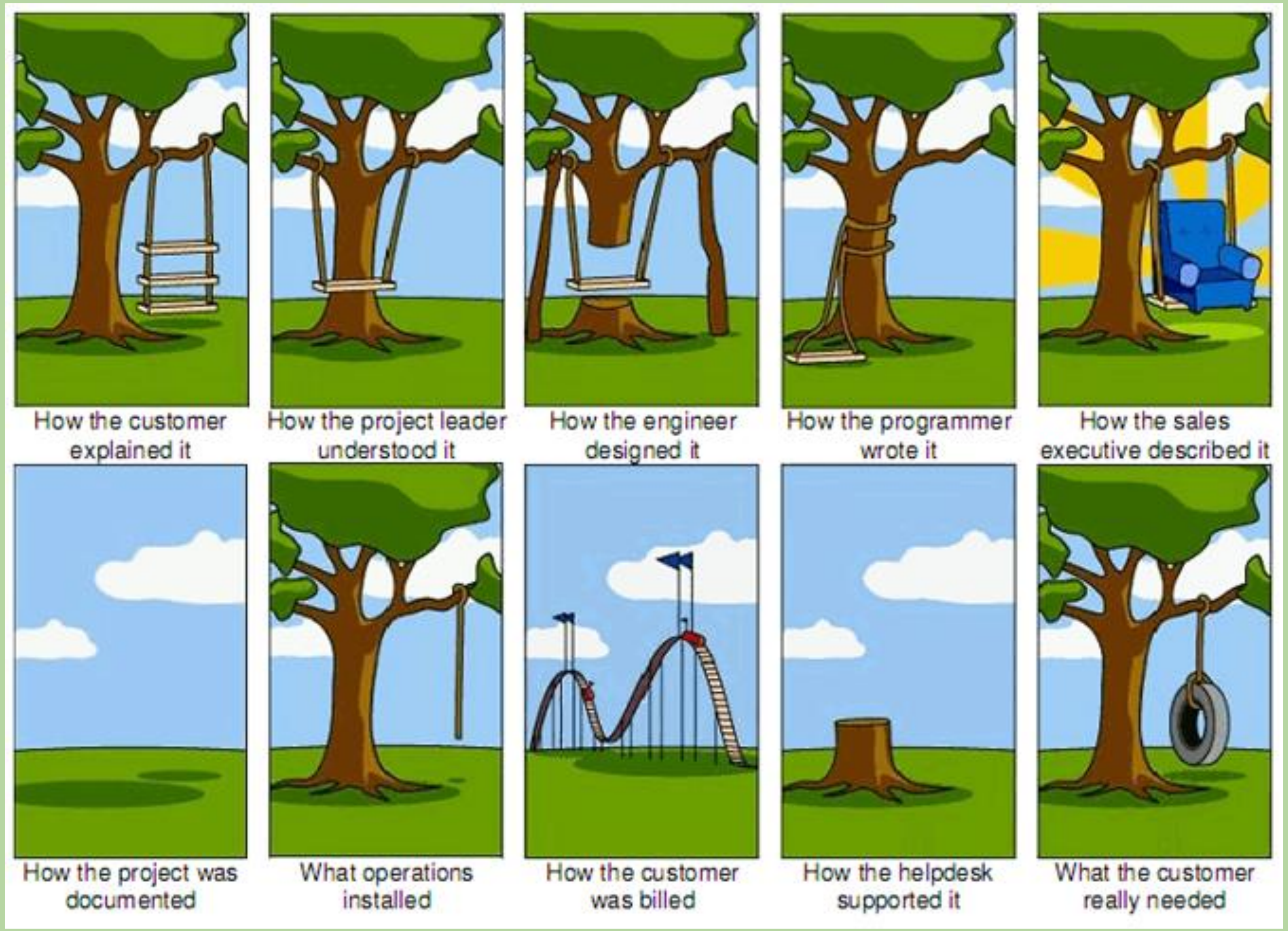
1. Analyse phase
2. Idea phase
3. Concept phase
4. Marketing
5. Evaluation
6. Questions

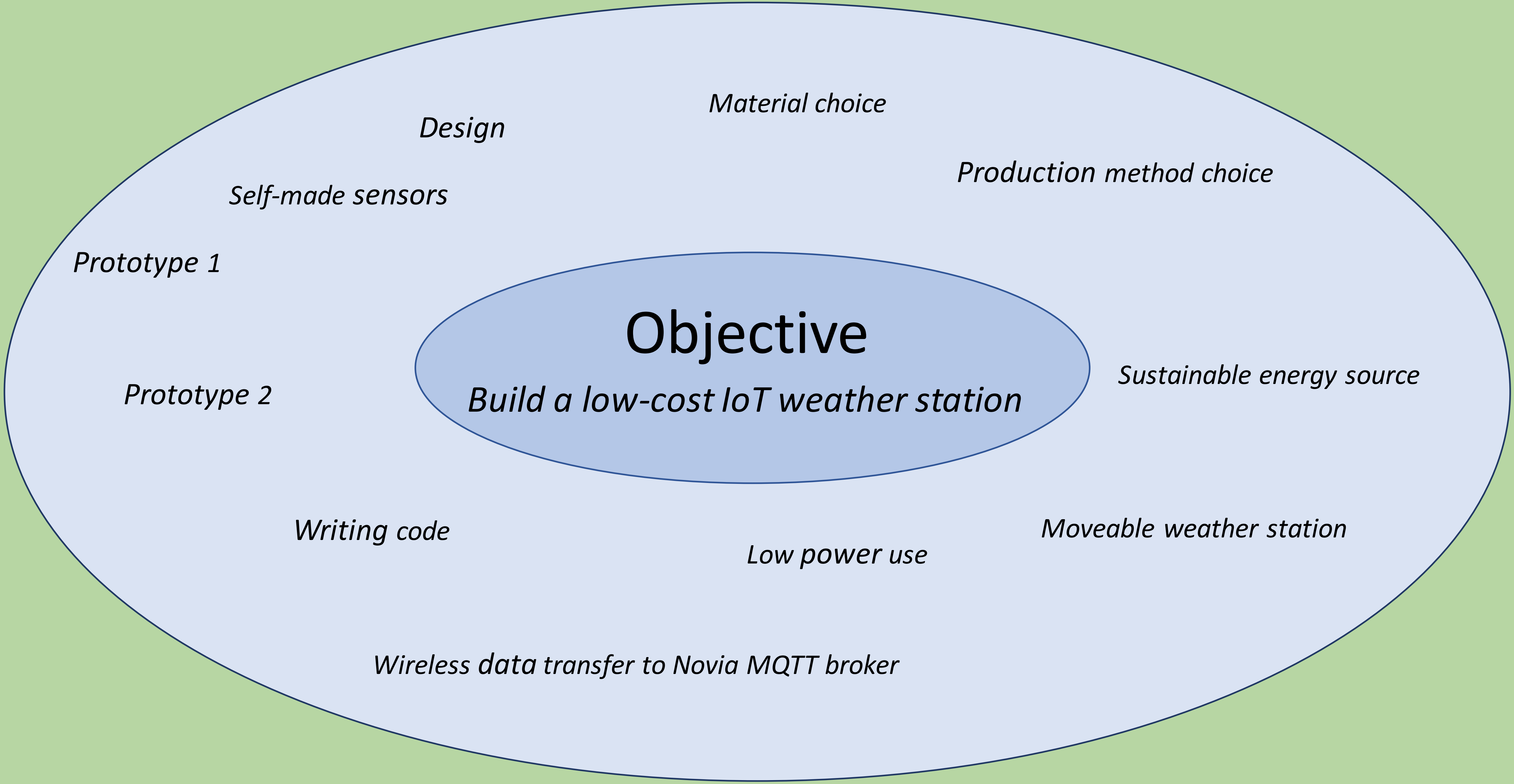
Analyse phase

Analyse the given information for all the problems that need to be solved

<u>Questions that needed to be answered</u>

- Objectives?
- What sensors to use?
- What materials to use?
- Forces on the weather station?
- What programming library’s to use?


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the helpdesk supported it
What the customer really needed



Objective

Build a low-cost IoT weather station

Design

Material choice

Production method choice

Self-made sensors

Prototype 1

Prototype 2

Sustainable energy source

Writing code

Low power use

Moveable weather station

Wireless data transfer to Novia MQTT broker

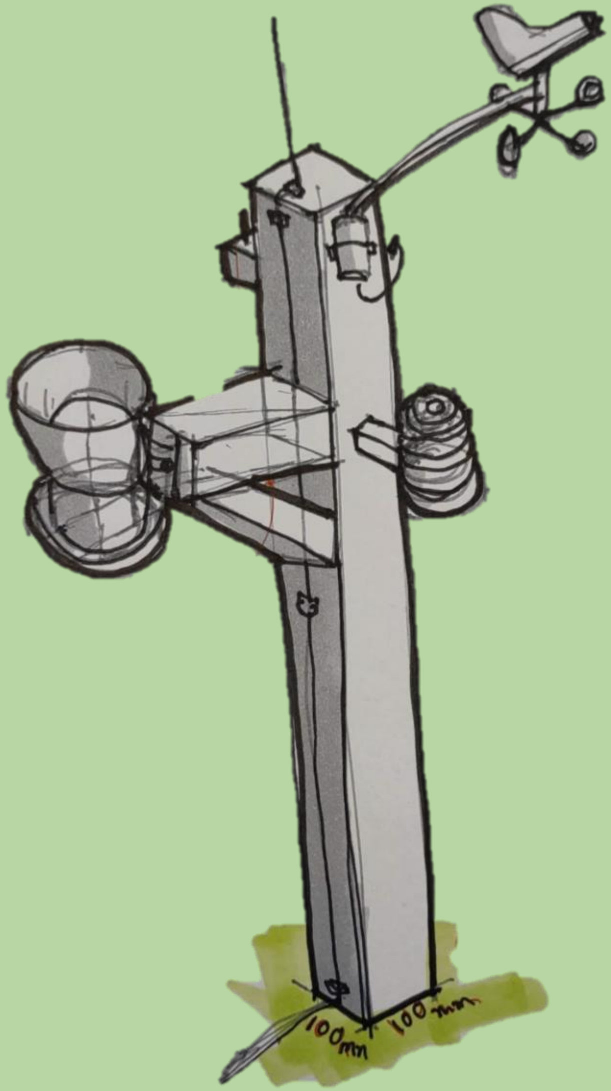
SENSORS

Weather station V1

- Wind speed
- Wind direction
- Temperature
- Relative humidity
- Rainfall

Weather station V2

- Wind speed
- Wind direction
- Temperature
- Relative humidity
- Rainfall
- Snow depth (Bonus)
- Light (Bonus)



Materials (Mounting station)

Mounting station material	Wood	Steel	Aluminium	Stone	Sandbags
Weather resistant	High	Medium	High	High	Medium
Corrosive resistant	High	Low	High	High	High
Weight	Medium	High	Medium	High	High
Re-usable	High	Medium	Medium	High	Medium
Price	Low	Medium	High	Low	Low
Complexity to work with	Low	High	Medium	Low	Low
Strength	Medium	High	High	Low	Low

Materials (Mounting station)

Mounting station material	Wood	Steel	Aluminium	Stone	Sandbags
Weather resistant	High	Medium	High	High	Medium
Corrosive resistant	High	Low	High	High	High
Weight	Medium	High	Medium	High	High
Re-usable	High	Medium	Medium	High	Medium
Price	Low	Medium	High	Low	Low
Complexity to work with	Low	High	Medium	Low	Low
Strength	Medium	High	High	Low	Low

Materials (Filament)

3D-print filament	PLA	ABS	PETG	ASA
Waterproof	Low	Low	High	High
UV-proof	Low	Low	High	High
Low temperature proof	Medium	Low	High	Medium
High temperature proof	Low	High	Medium	High
Price	Low	Low	Medium	High
Printable	High	Medium	Medium	Low
Strength	Medium	High	Medium	Low

Materials (Filament)

3D-print filament	PLA	ABS	PETG	ASA
Waterproof	Low	Low	High	High
UV-proof	Low	Low	High	High
Low temperature proof	Medium	Low	High	Medium
High temperature proof	Low	High	Medium	High
Price	Low	Low	Medium	High
Printable	High	Medium	Medium	Low
Strength	Medium	High	Medium	Low

Forces

Wind

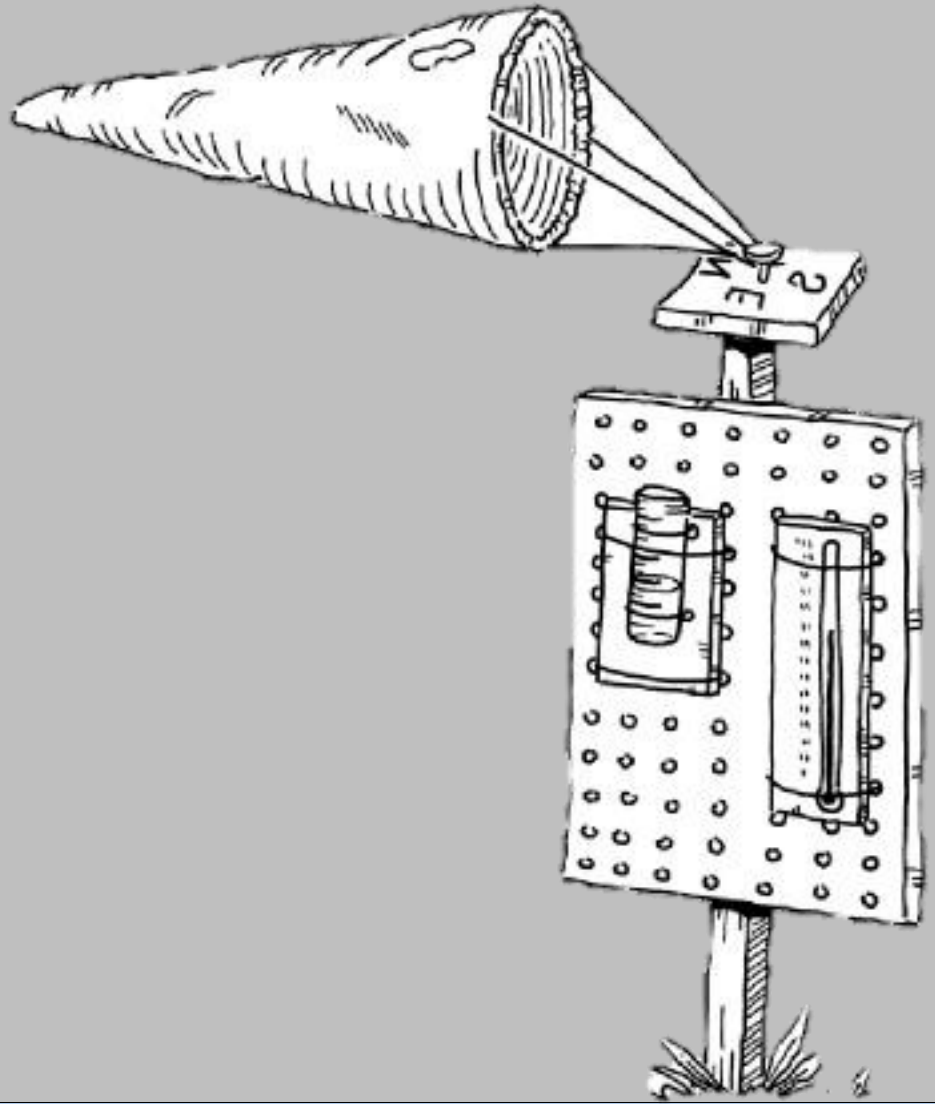
Weight of the sensors (Moment)

Birds

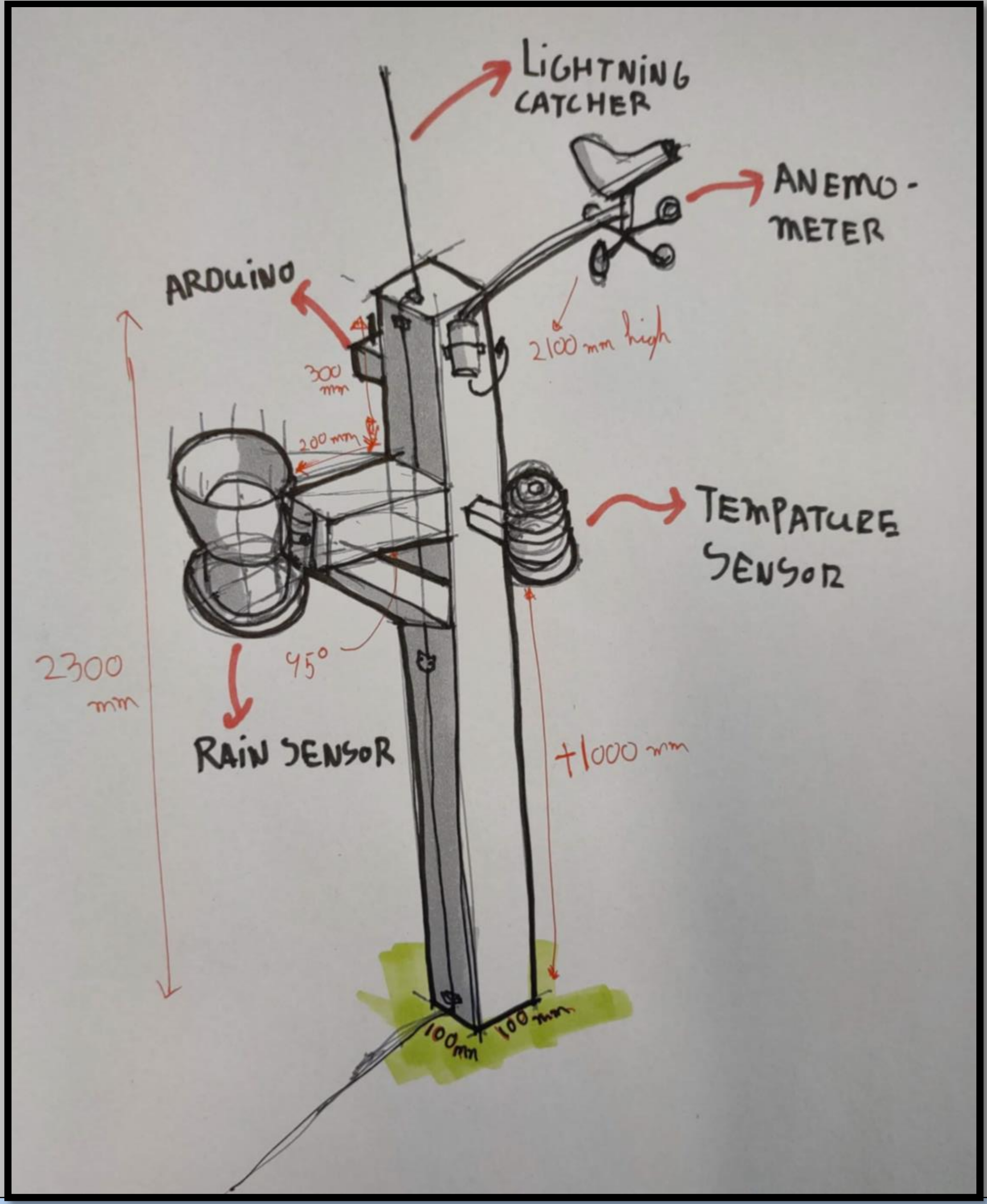
Idea phase

Generating sketch ideas for the design problems

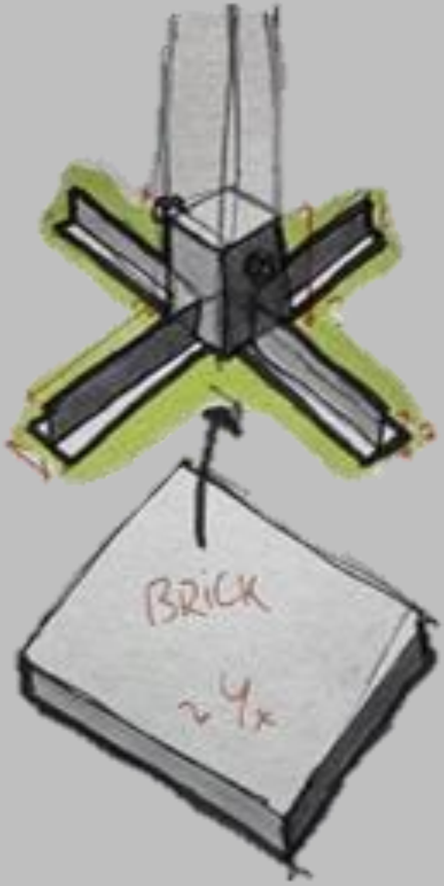
- General Layout
- Mounting station
- Arduino housing

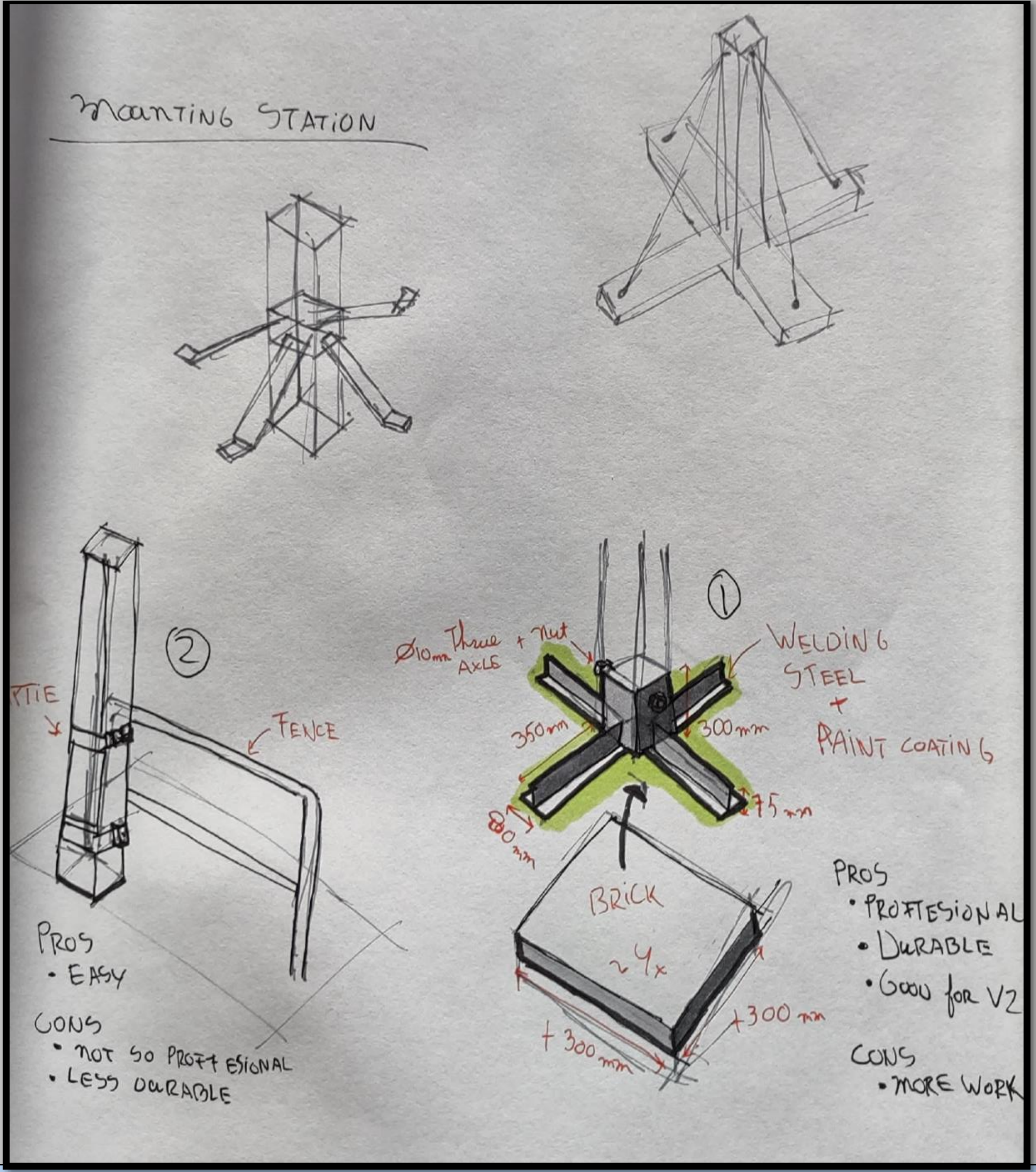


General layout

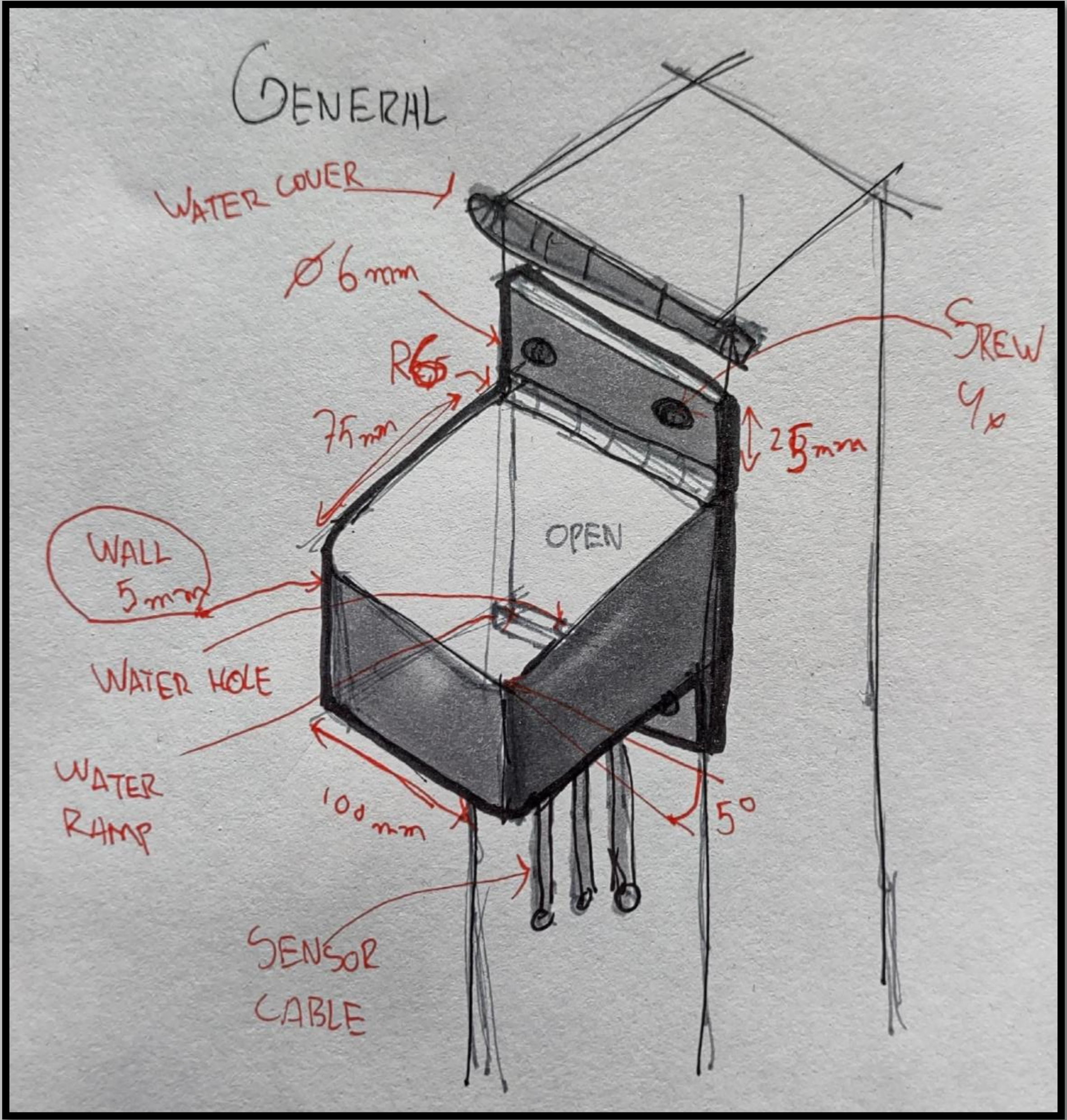


Mounting station (Base)





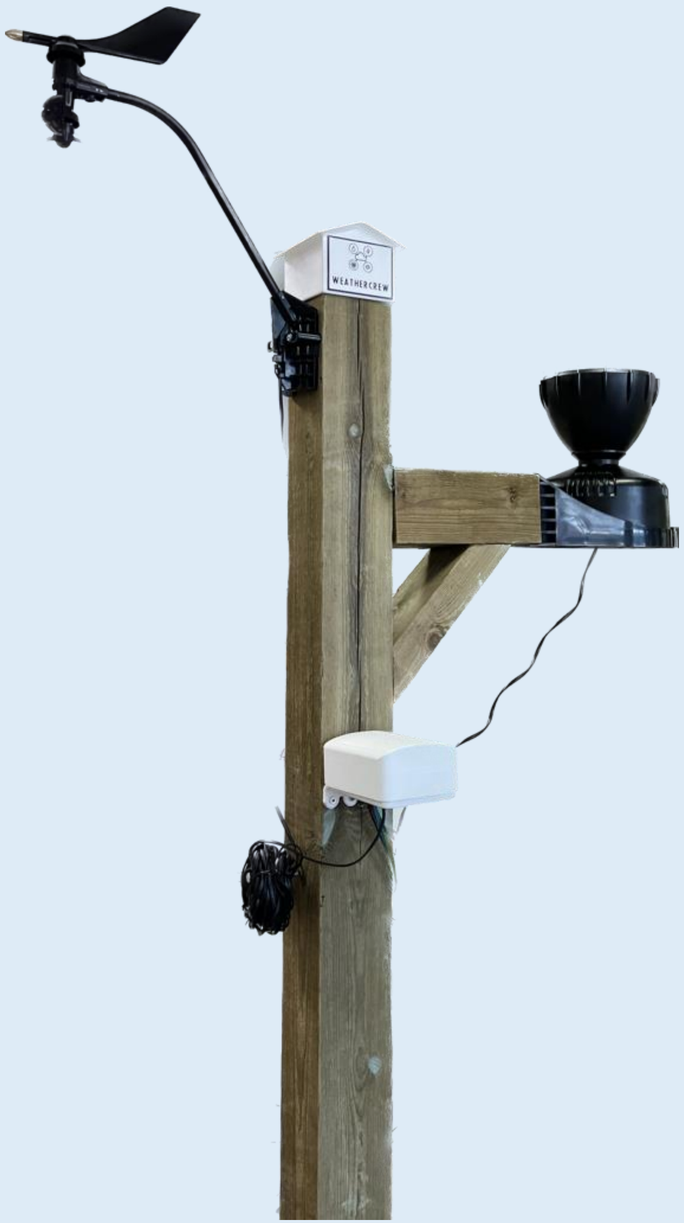
Arduino housing



Concept phase

Putting everything together into a prototype

- Sensors
- CAD design
- Code
- Prototype V1



Sensor: Anemometer

- Davis Vantage Pro2 Anemometer 6410
- Accurately measures wind direction and speed
- Wind speed range: 0 to 322 km/h
- Includes 12m cable
- Weight 1,5 kg
- Price 185 €

Sensor: Rainfall

- Davis rain collector 6466
- Aerodynamic and self-emptying tipping spoon provides accurate rain data
- Tapping spoon with magnetic switch
- Each tip indicates 0.2 mm of rain
- Includes 12m cable
- Price: 125 €

Sensor: Temperature and humidity

- AM2320 digital temperature and humidity sensor
- 3% humidity and 0.5°C temperature accuracy
- Temperature range: -40°C to 80°C
- Price: 3,95 €

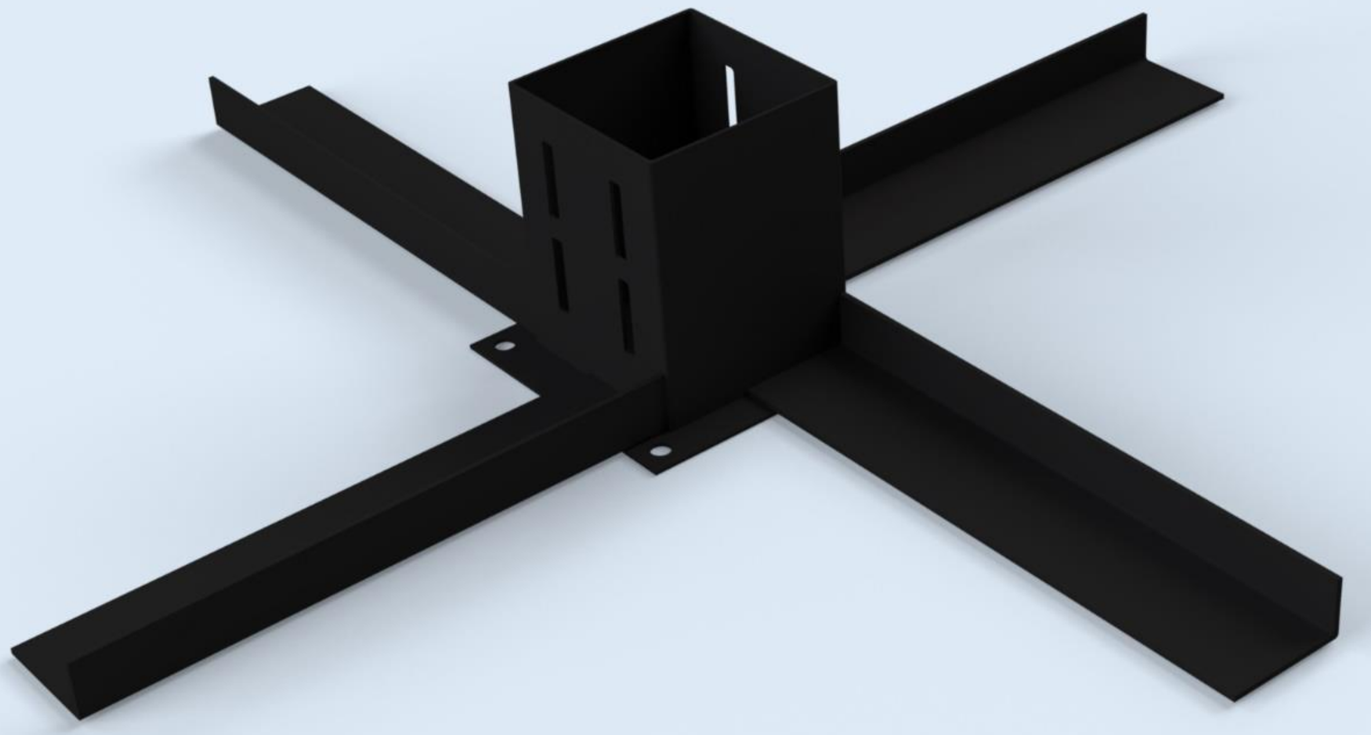
CAD: Marketing

CAD: Mounting base

- Garden pole mount
- L-beam 300mm (balance)

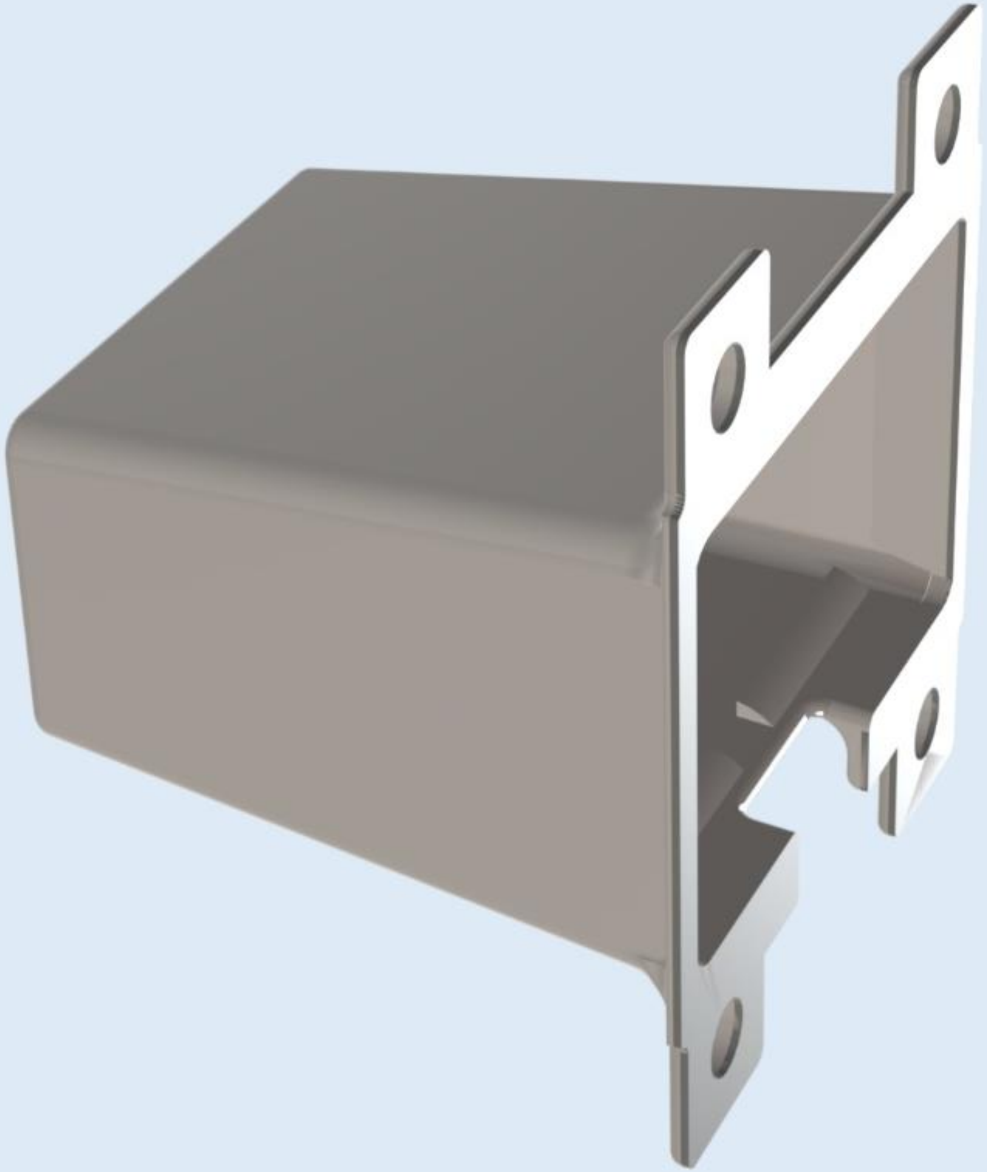
<u>Production</u>

- Sawing
- Grinding
- Cleaning
- Welding
- Sandblasting
- Painting



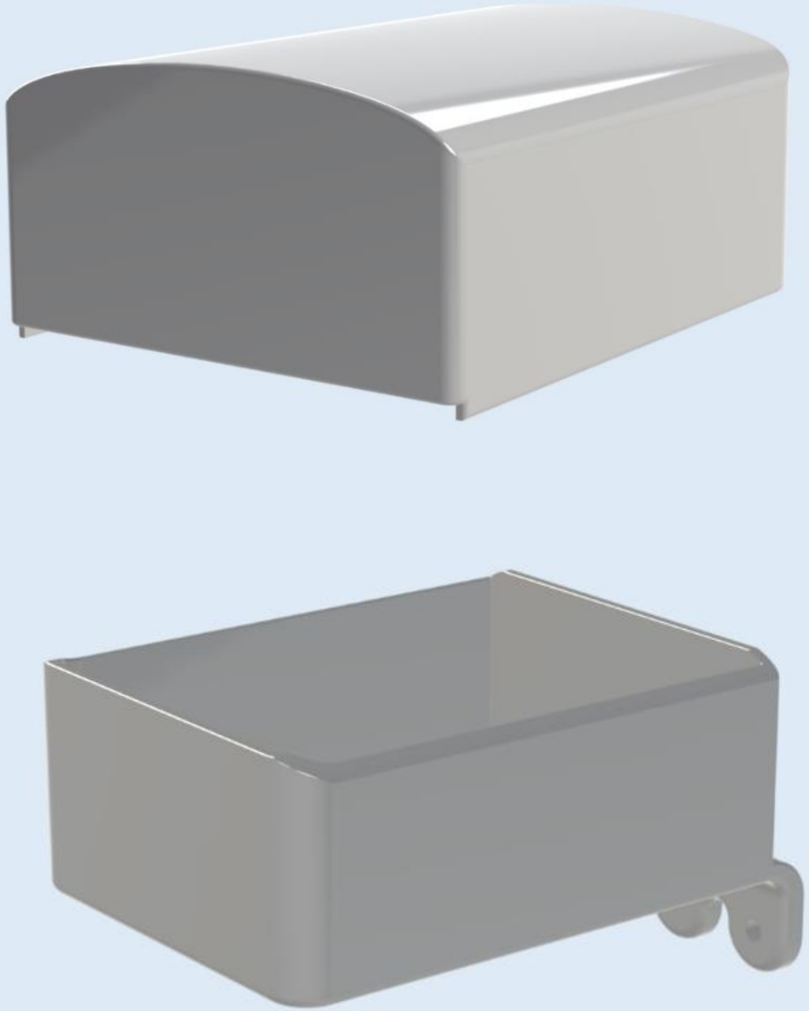
CAD: Arduino housing

- V1 Arduino housing
- Fits:
  - Arduino
  - Connectors
  - Antenna
  - Battery



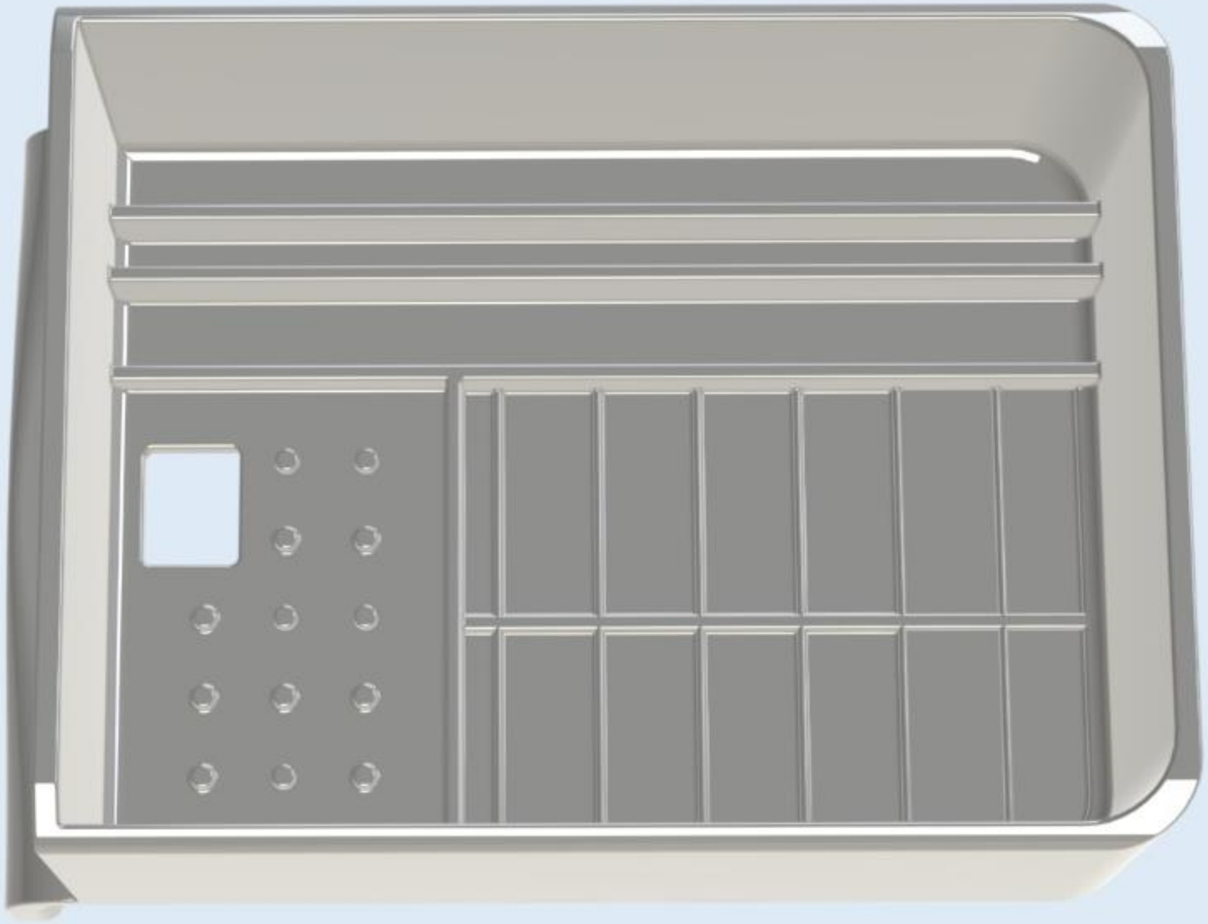
CAD: Arduino housing

- V2 Arduino housing
- Removable top
- Round top



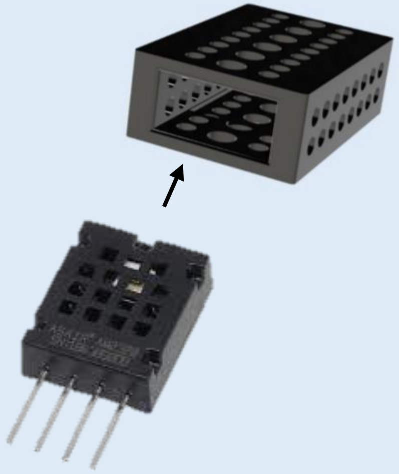
CAD: Arduino housing

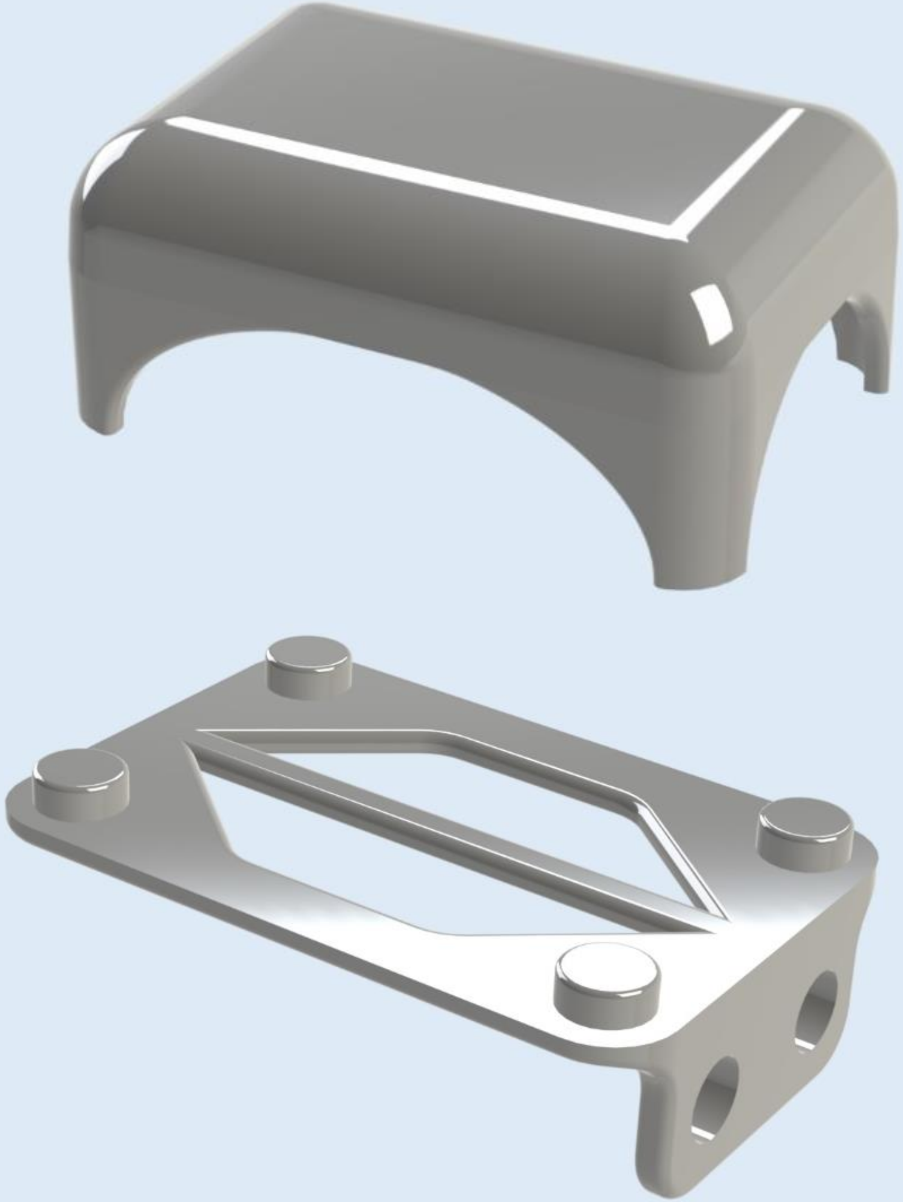
- Arduino
- Connectors
- Antenna
- Battery



CAD: Temperature sensor housing

- Waterproof
- Steady airflow



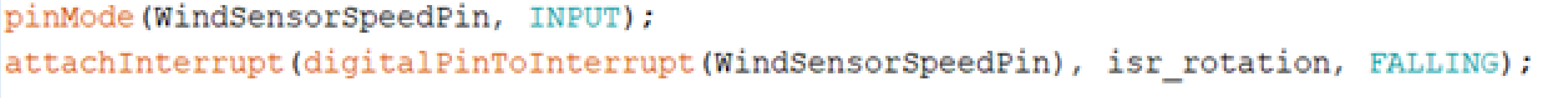


Library’s

Arduino library’s

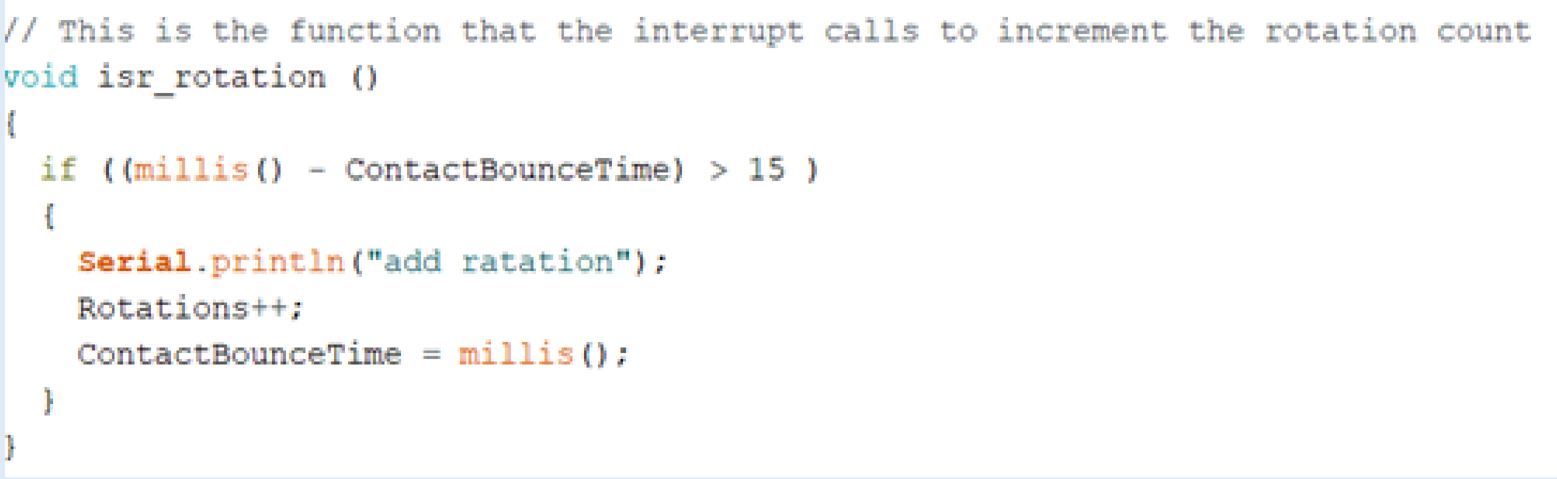
- math.h
- Adafruit_Sensor.h
- Adafruit_AM2320.h
- MKRNB.h
- RTCZero.h
- ArduinoMqttClient.h

Wind speed



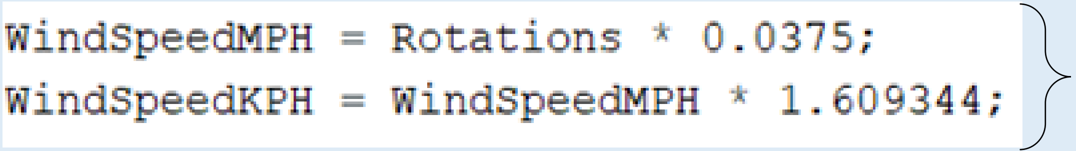
```
pinMode(WindSensorSpeedPin, INPUT);
attachInterrupt(digitalPinToInterrupt(WindSensorSpeedPin), isr_rotation, FALLING);
```

Interrupt



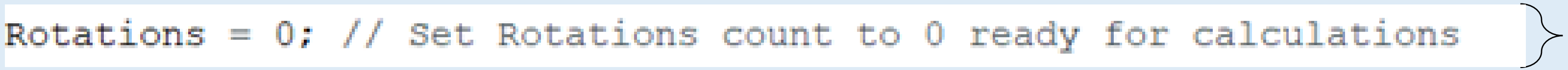
```
// This is the function that the interrupt calls to increment the rotation count
void isr_rotation ()
{
  if ((millis() - ContactBounceTime) > 15 )
  {
    Serial.println("add ratation");
    Rotations++;
    ContactBounceTime = millis();
  }
}
```

Conditions add 1 rotation, when the sensor rotates completely



```
WindSpeedMPH = Rotations * 0.0375;
WindSpeedKPH = WindSpeedMPH * 1.609344;
```

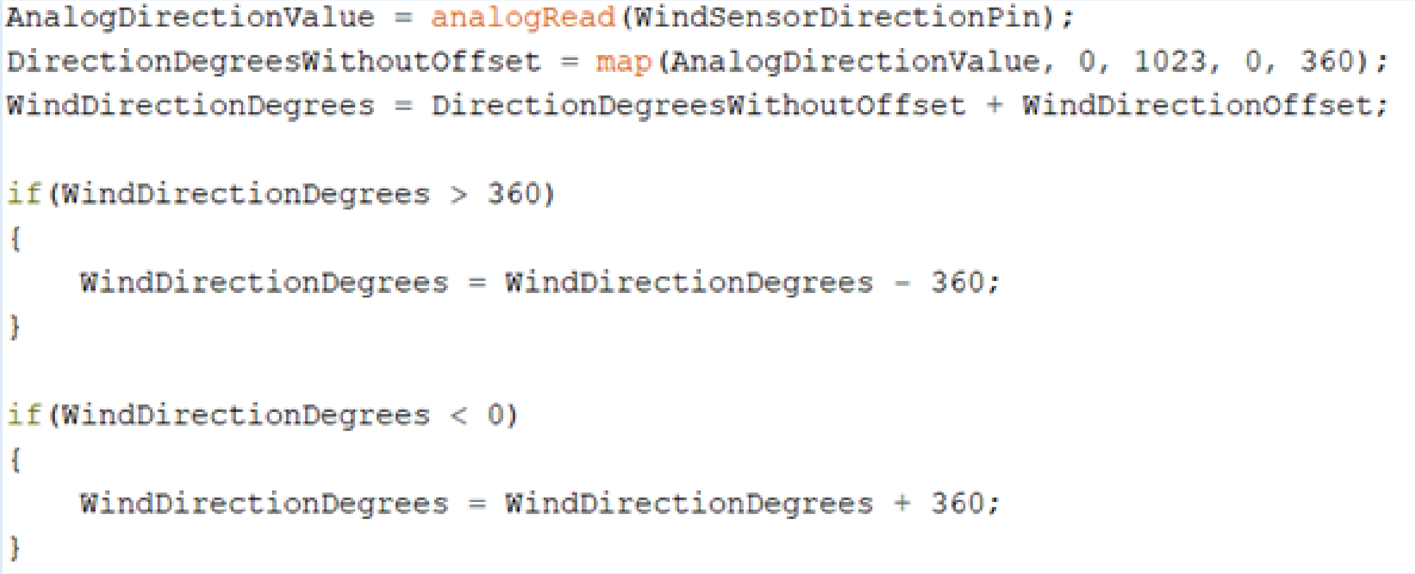
Calculation wind speed in mp/h and km/h



```
Rotations = 0; // Set Rotations count to 0 ready for calculations
```

Reset

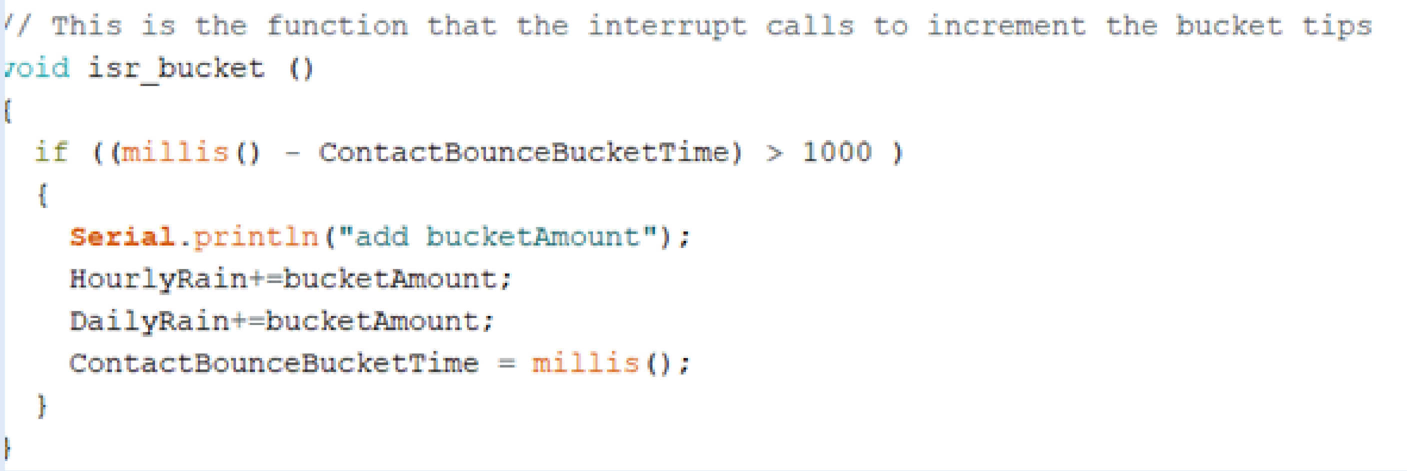
Wind direction



```
AnalogDirectionValue = analogRead(WindSensorDirectionPin);
DirectionDegreesWithoutOffset = map(AnalogDirectionValue, 0, 1023, 0, 360);
WindDirectionDegrees = DirectionDegreesWithoutOffset + WindDirectionOffset;

if(WindDirectionDegrees > 360)
{
    WindDirectionDegrees = WindDirectionDegrees - 360;
}

if(WindDirectionDegrees < 0)
{
    WindDirectionDegrees = WindDirectionDegrees + 360;
}
```

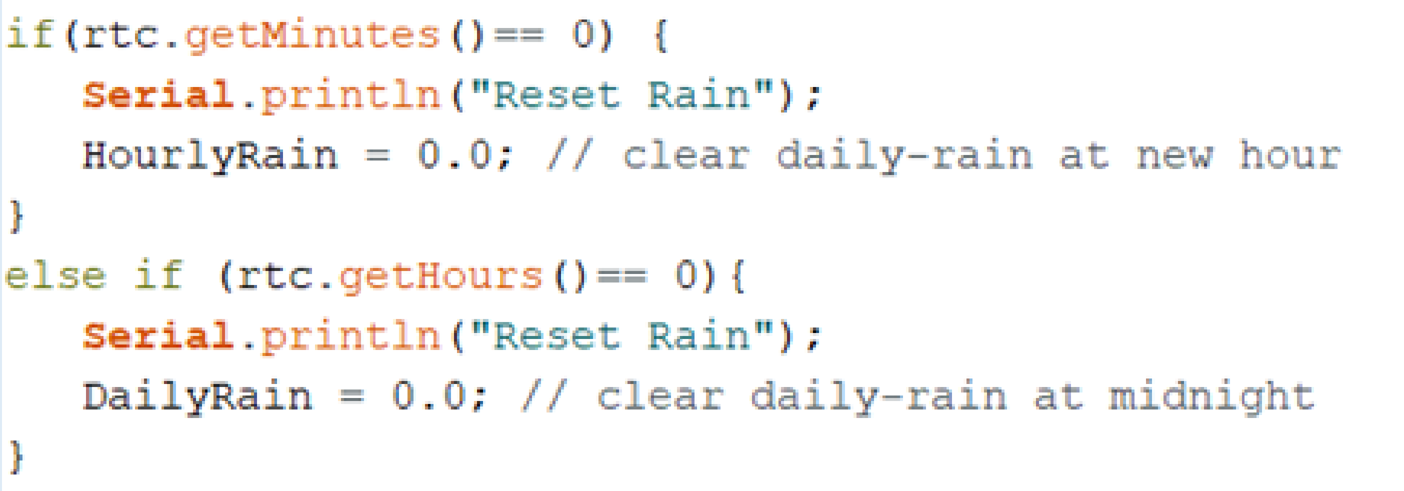
Wind direction degrees

Rainfall



```
// This is the function that the interrupt calls to increment the bucket tips
void isr_bucket ()
{
  if ((millis() - ContactBounceBucketTime) > 1000 )
  {
    Serial.println("add bucketAmount");
    HourlyRain+=bucketAmount;
    DailyRain+=bucketAmount;
    ContactBounceBucketTime = millis();
  }
}
```

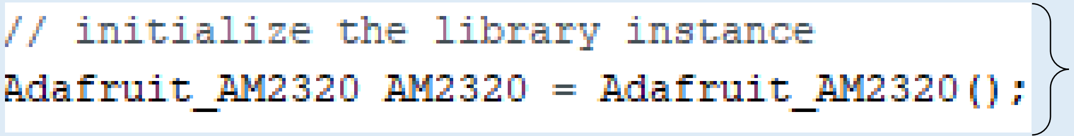
Bucket tips meter



```
if(rtc.getMinutes()== 0)  {
   Serial.println("Reset Rain");
   HourlyRain = 0.0; // clear daily-rain at new hour
}
else if (rtc.getHours()== 0){
   Serial.println("Reset Rain");
   DailyRain = 0.0; // clear daily-rain at midnight
}
```

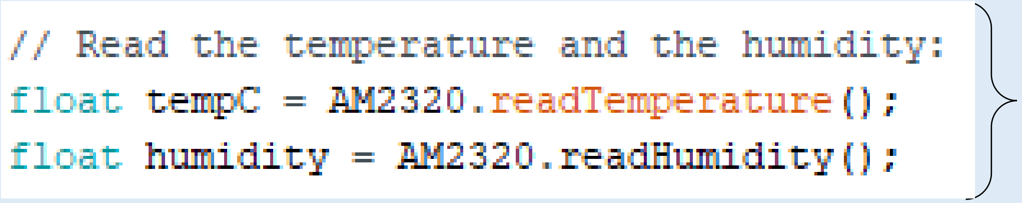
Reset the hourly and daily rain value

Temperature and humidity



```
// initialize the library instance
Adafruit_AM2320 AM2320 = Adafruit_AM2320();
```

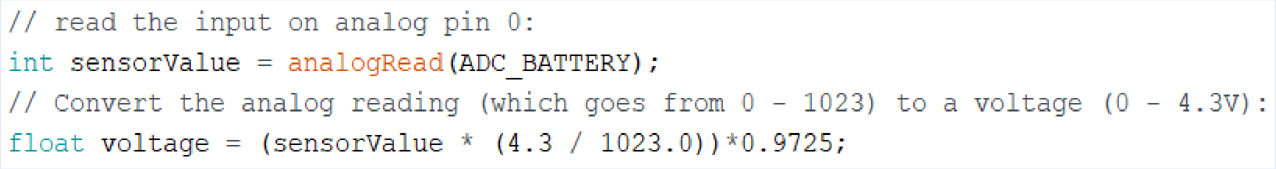
Library use



```
// Read the temperature and the humidity:
float tempC = AM2320.readTemperature();
float humidity = AM2320.readHumidity();
```

Execution

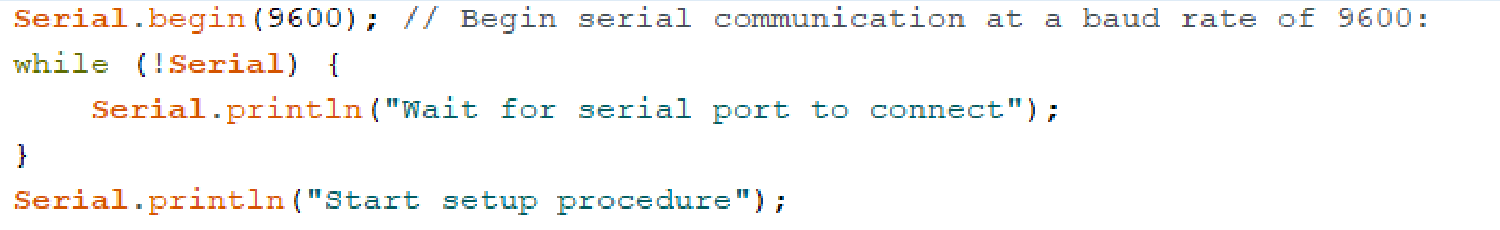
Battery



```
// read the input on analog pin 0:
int sensorValue = analogRead(ADC_BATTERY);
// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 4.3V):
float voltage = (sensorValue * (4.3 / 1023.0))*0.9725;
```

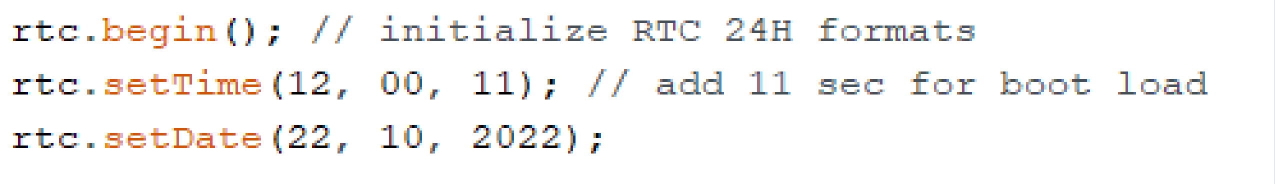
Battery voltage

Setup



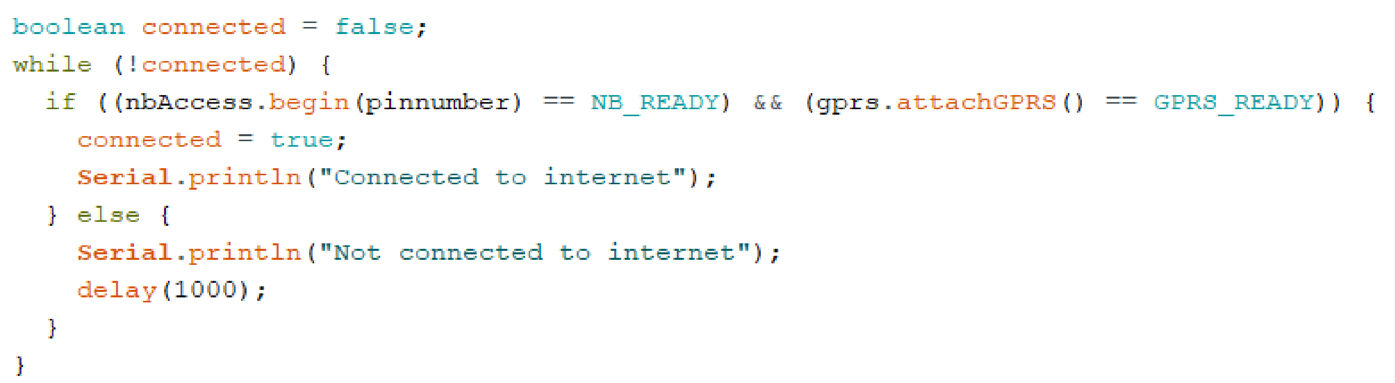
```
Serial.begin(9600); // Begin serial communication at a baud rate of 9600:
while (!Serial) {
    Serial.println("Wait for serial port to connect");
}
Serial.println("Start setup procedure");
```

Connect the serial port



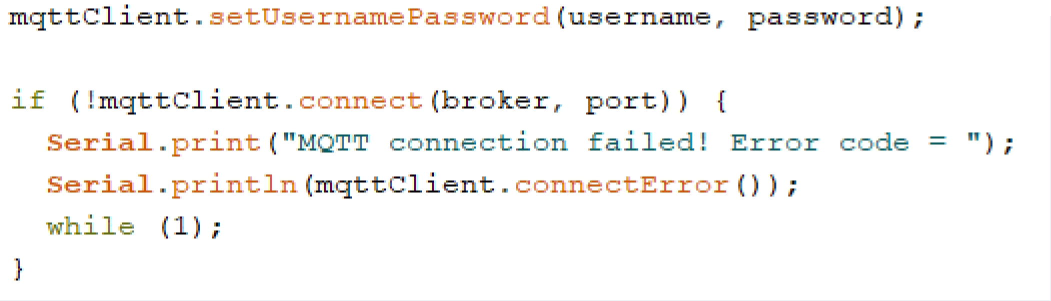
```
rtc.begin(); // initialize RTC 24H formats
rtc.setTime(12, 00, 11); // add 11 sec for boot load
rtc.setDate(22, 10, 2022);
```

Set time with offset



```
boolean connected = false;
while (!connected) {
  if ((nbAccess.begin(pinnumber) == NB_READY) && (gprs.attachGPRS() == GPRS_READY)) {
    connected = true;
    Serial.println("Connected to internet");
  } else {
    Serial.println("Not connected to internet");
    delay(1000);
  }
}
```

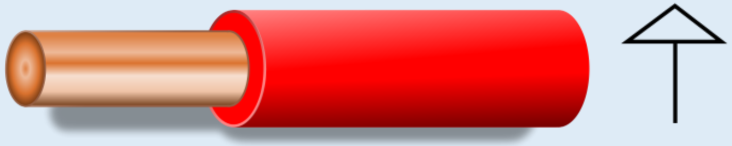
Internet connection



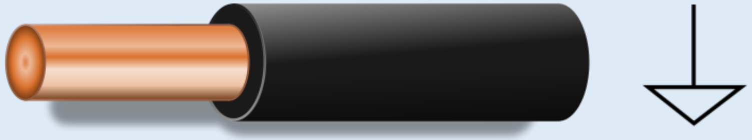
```
mqttClient.setUsernamePassword(username, password);

if (!mqttClient.connect(broker, port)) {
  Serial.print("MQTT connection failed! Error code = ");
  Serial.println(mqttClient.connectError());
  while (1);
}
```

MQTT connection

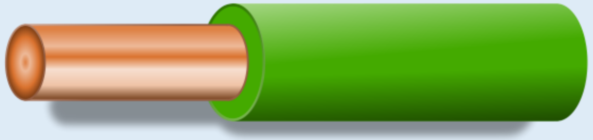
Wire Colors



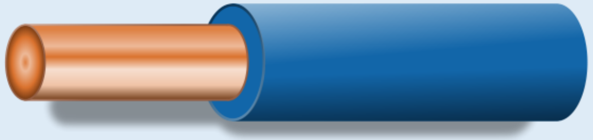
VCC 3.3 Volt



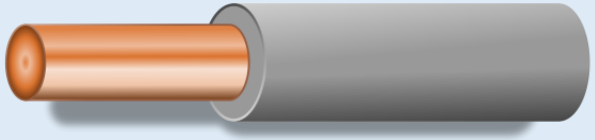
Ground



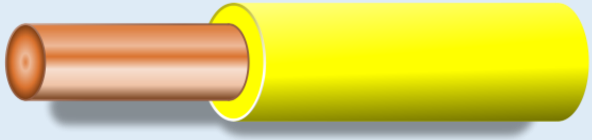
Serial clock line



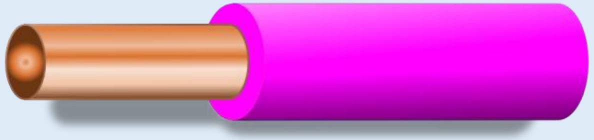
Serial data line



Wind Speed Sensor

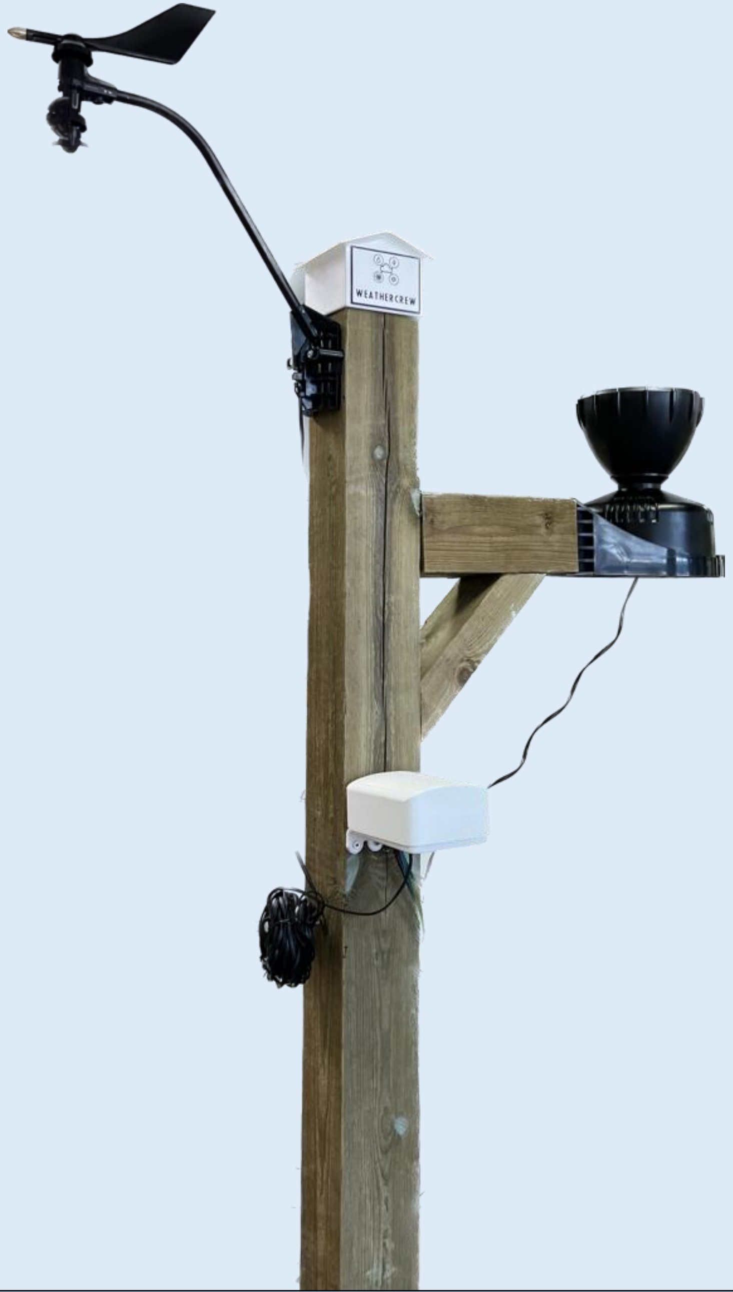


Wind Direction Sensor

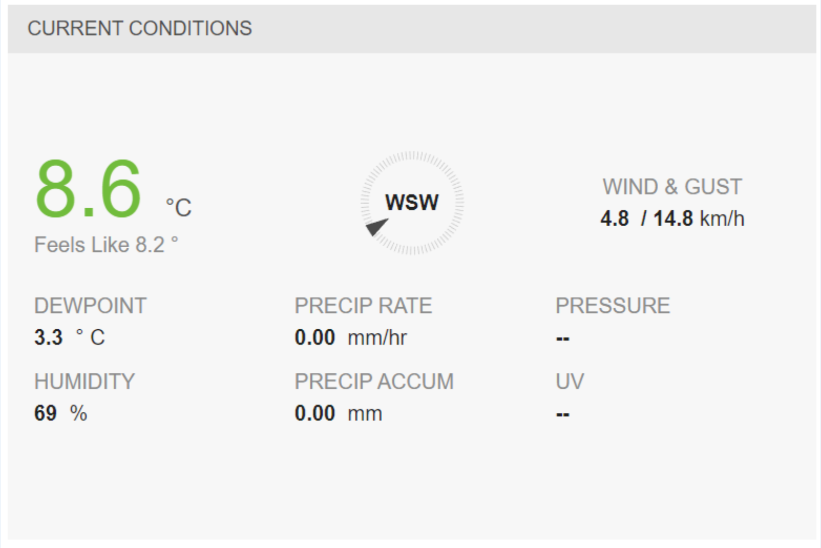


Rain sensor

Prototype V1



Prototype V1 measurements



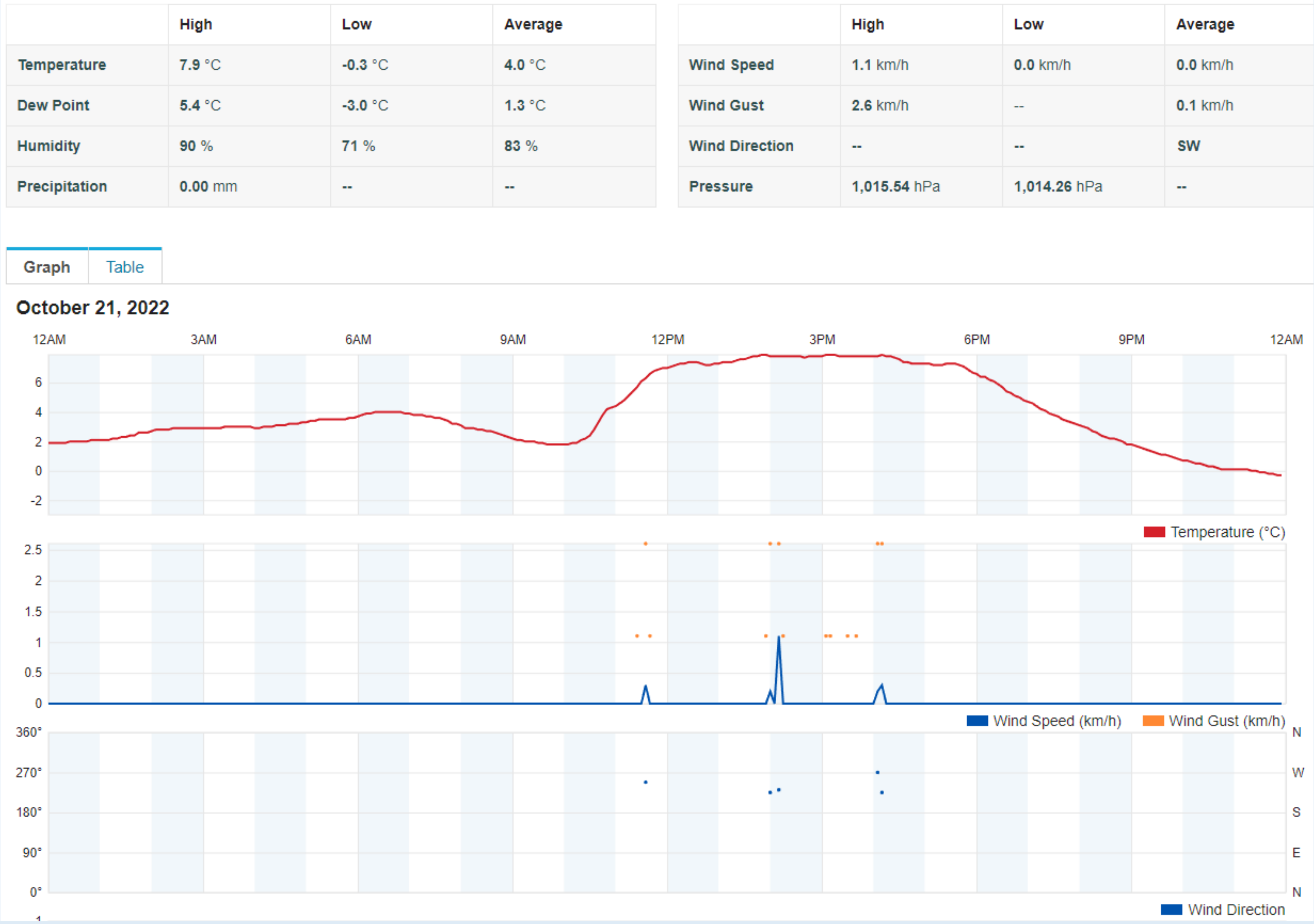
CURRENT CONDITIONS

8.6 °C

Feels Like 8.2 °

WIND & GUST
4.8 / 14.8 km/h

DEWPOINT	PRECIP RATE	PRESSURE
3.3 ° C	0.00 mm/hr	--
HUMIDITY	**PRECIP ACCUM**	**UV**
69 %	0.00 mm	--



	High	Low	Average
Temperature	7.9 °C	-0.3 °C	4.0 °C
Dew Point	5.4 °C	-3.0 °C	1.3 °C
Humidity	90 %	71 %	83 %
Precipitation	0.00 mm	--	--

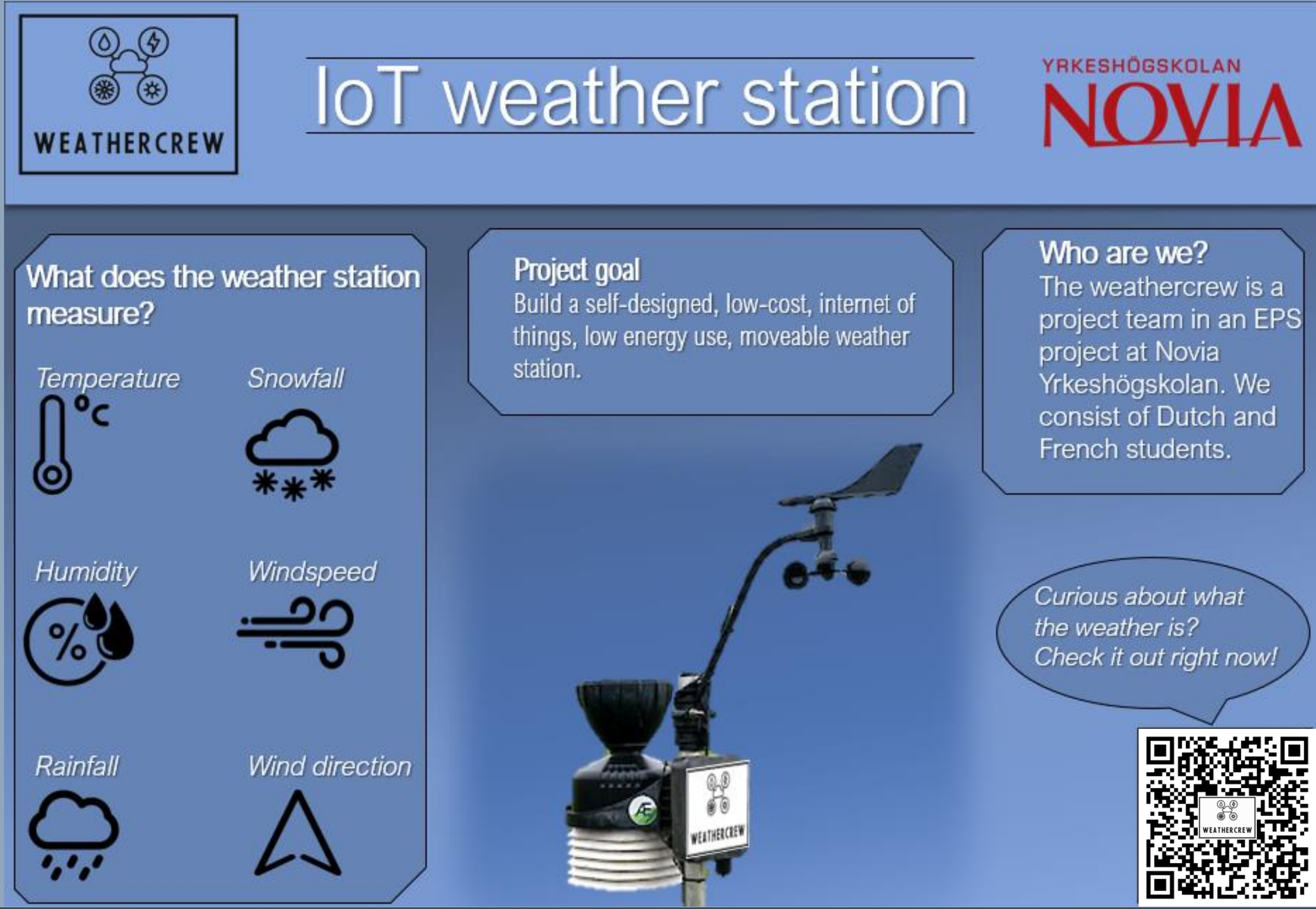
	High	Low	Average
Wind Speed	1.1 km/h	0.0 km/h	0.0 km/h
Wind Gust	2.6 km/h	--	0.1 km/h
Wind Direction	--	--	SW
Pressure	1,015.54 hPa	1,014.26 hPa	--

Graph Table

October 21, 2022

Concept phase

Poster



WEATHERCREW

IoT weather station

YRKESHÖGSKOLAN NOVIA

What does the weather station measure?

Temperature

°C

Snowfall

Humidity

%

Windspeed

Rainfall

Wind direction

Project goal

Build a self-designed, low-cost, internet of things, low energy use, moveable weather station.

Who are we?

The weathercrew is a project team in an EPS project at Novia Yrkeshögskolan. We consist of Dutch and French students.

Curious about what the weather is? Check it out right now!

Improvements for version 2

Design

- Strengthen the temperature sensor housing+ Arduino housing
- Design rainfall and wind sensor
- PCB design

Add-ons

- Snow depth sensor
- Light sensor

Code

- Make the code use less energy
- Add new code for the new sensors
- Switch from LTE-M to NB-IoT

Questions?