Group 25: IoT Environmental Sensors
Senior Design – Final Presentation

CSULA Team Members:
Juan Avila, Anthony Huynh, Jose Rodriguez, David Hernandez,
Jeffrey Espinoza, Vincent Oviedo, Akbar Rizvi

Advisor: Airs Lin

Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

Background

- Network of physical devices embedded with sensors or software that help interexchange information over the internet
- Each device supports easy connectivity to our sensor network without much consumer configuration



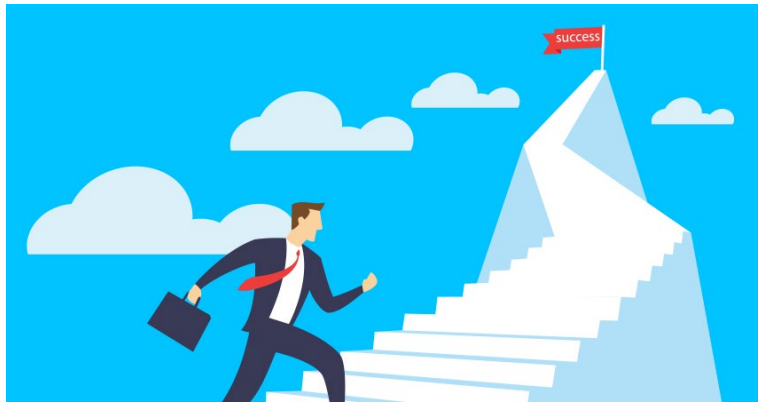
Problem

- Using IoT to detect CO2 in any environment
- Reduce human dependency
- Design sensors to collect environmental data and upload to our servers



Objective

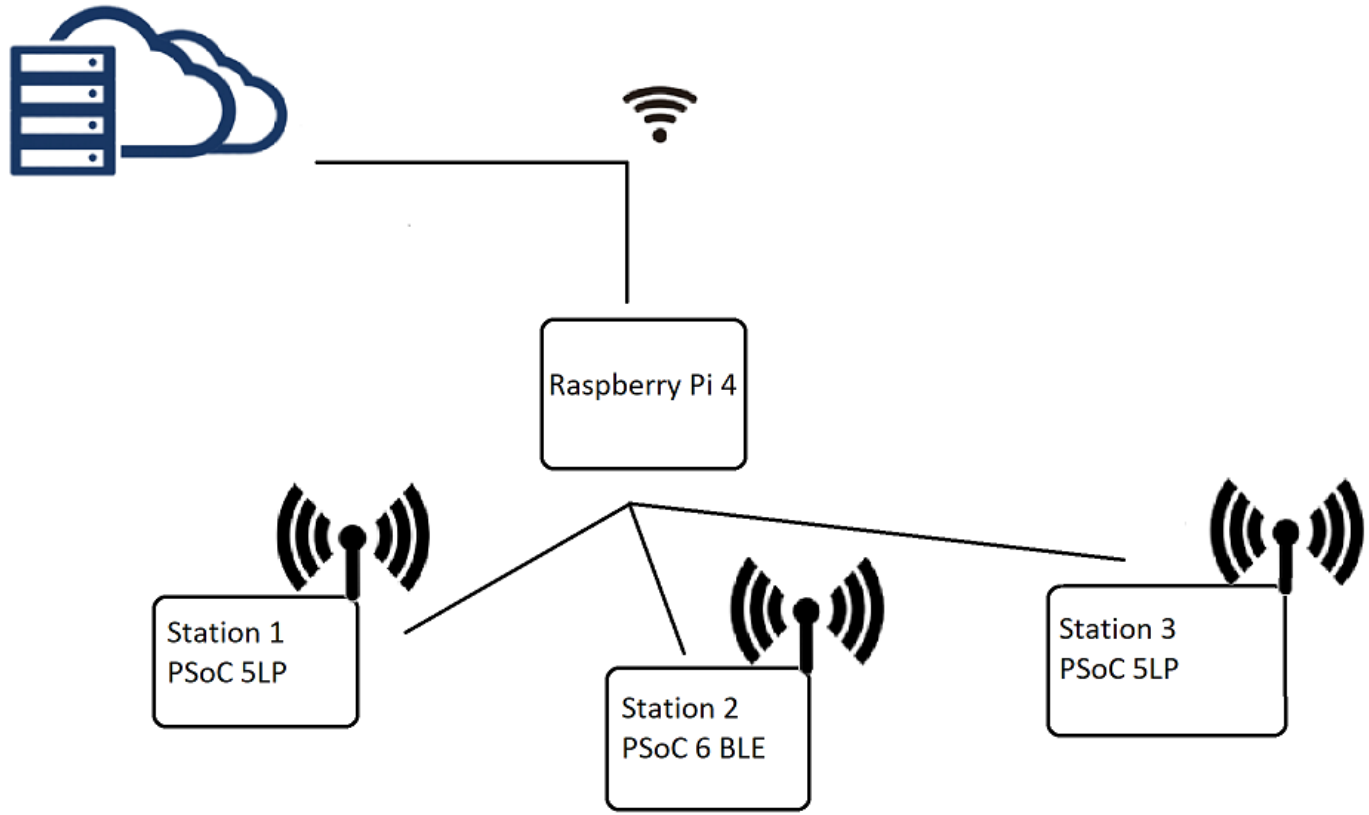
- Build a 4 stationed system that can:
    1. Detect both indoor and outdoor environment changes
    2. Gather data from stations and upload it to AWS server analyze
    3. Easy user deployment and scalability



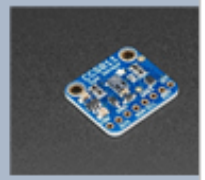
Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

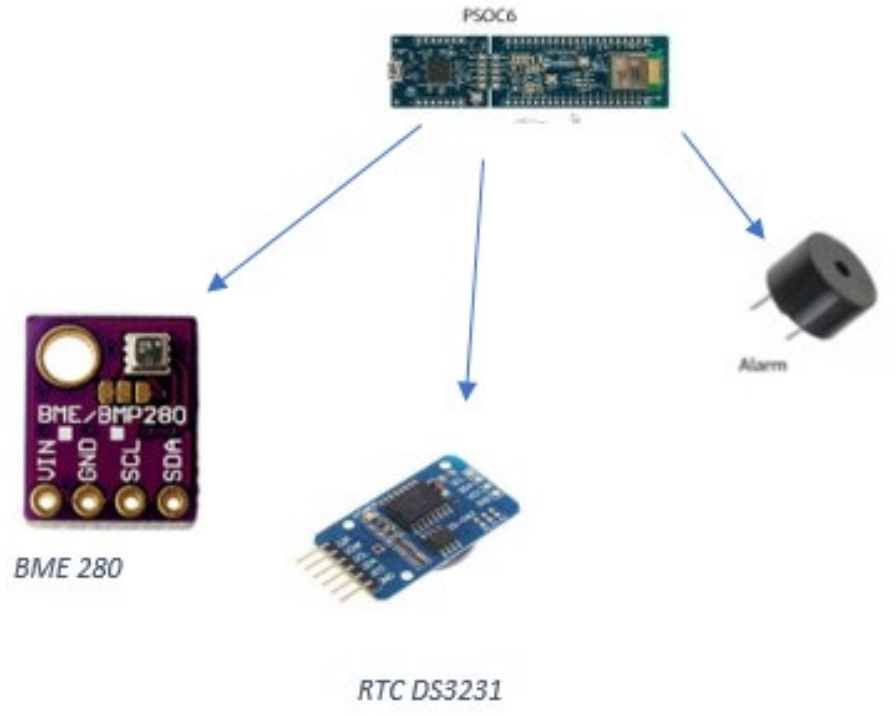
Concept Design

- Core station
  - Internet connectivity
    and RF connectivity
- Sensor stations
  - Scalable
  - Small footprint
- Server
  - Storage
  - Remote accessible
  - Affordable



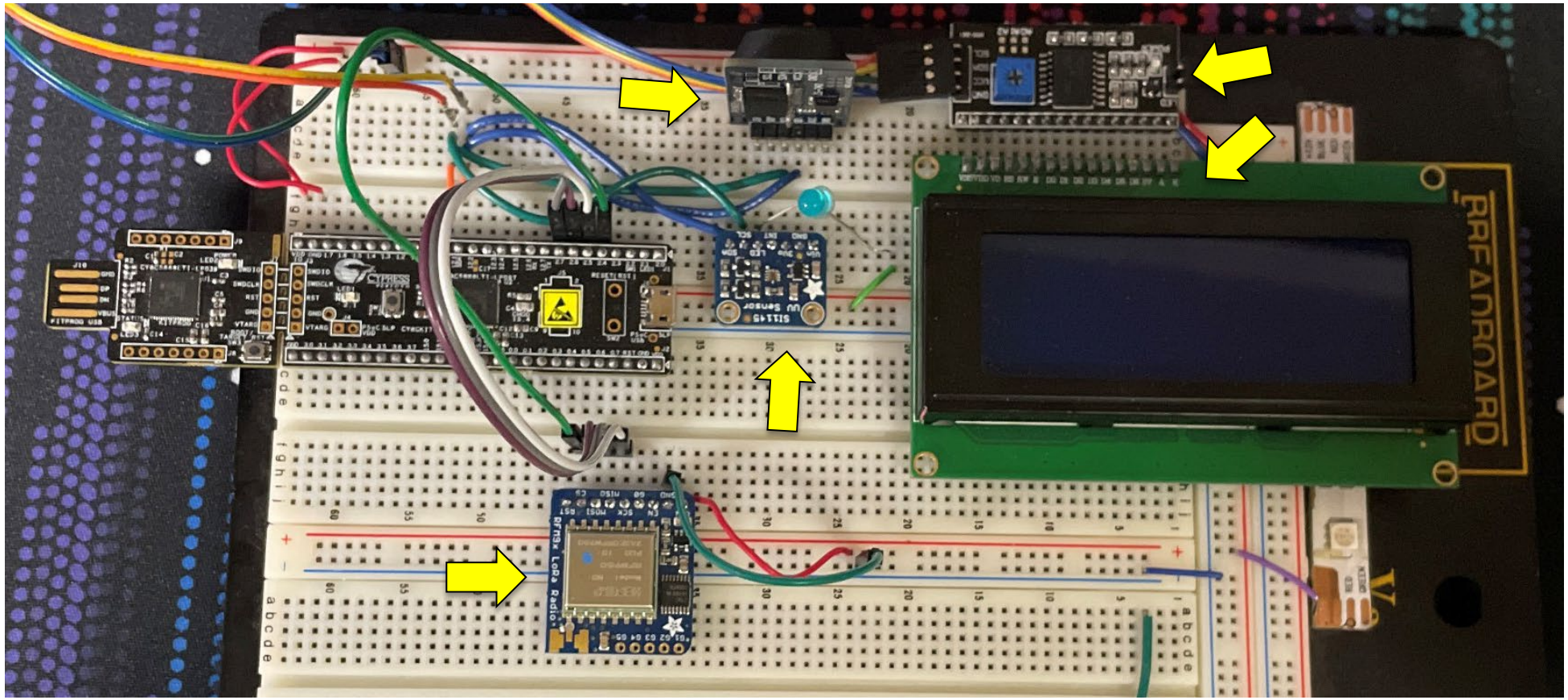
Environmental Sensors

Station 1	Weather sensors	Station 2	Indoor Temp/Humidity	Station 3	Indoor Gases/outdoor Gases
BME 280 Temp Humidity Pressure		BME 280 Temp Humidity Pressure		PPD42NJ Particle sensor unit	
SI1145 Digital UV light/index Infrared light				MQ-4 Methane Gas sensor	
SEN-15901 Weather meter kit 3 Parts:				CJMCU-811-CCS811 Gas sensor CO2 VOC MOX	
Rain Gauge Rain intensity					
Anemometer Wind speed					
Wind Vane Wind direction					

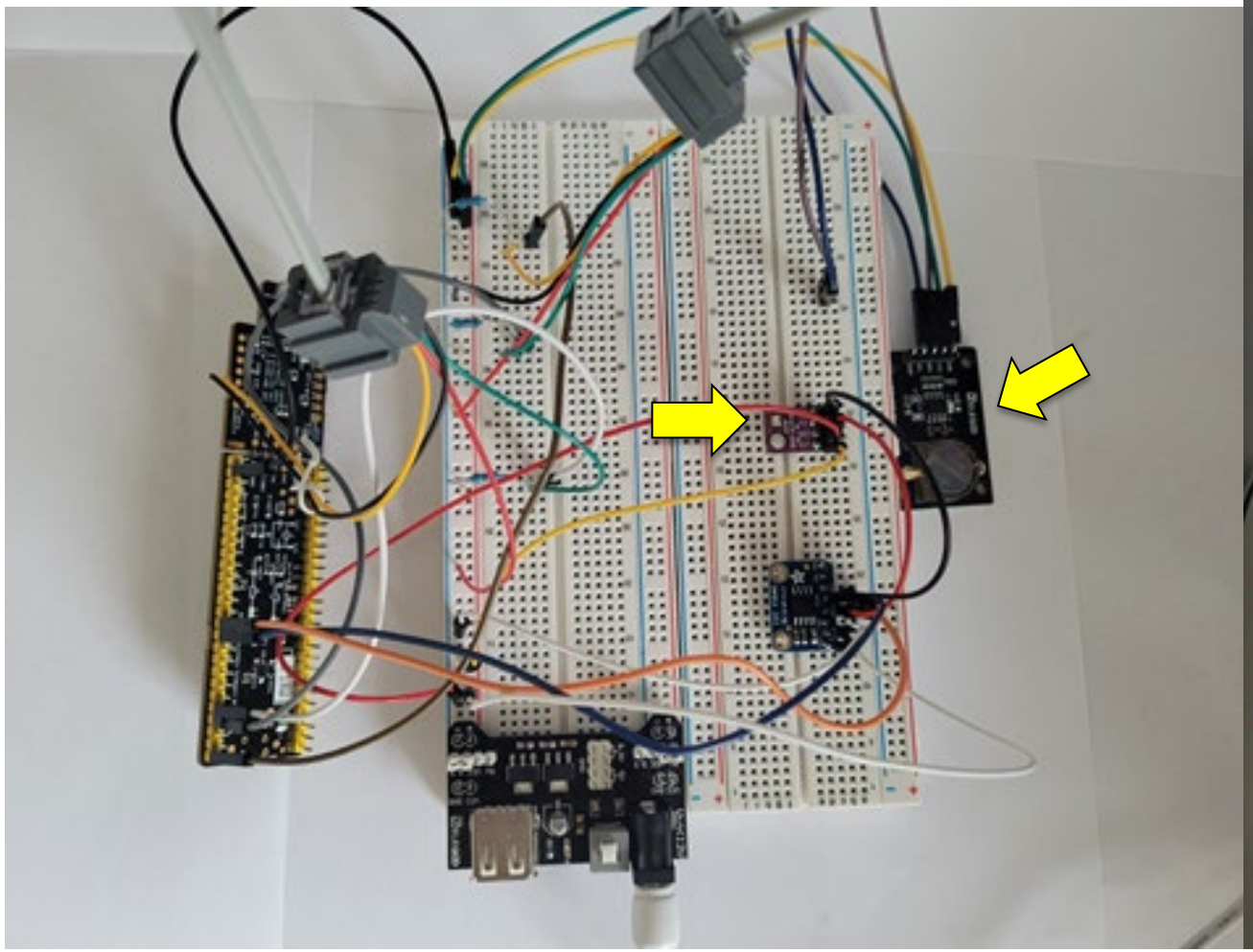


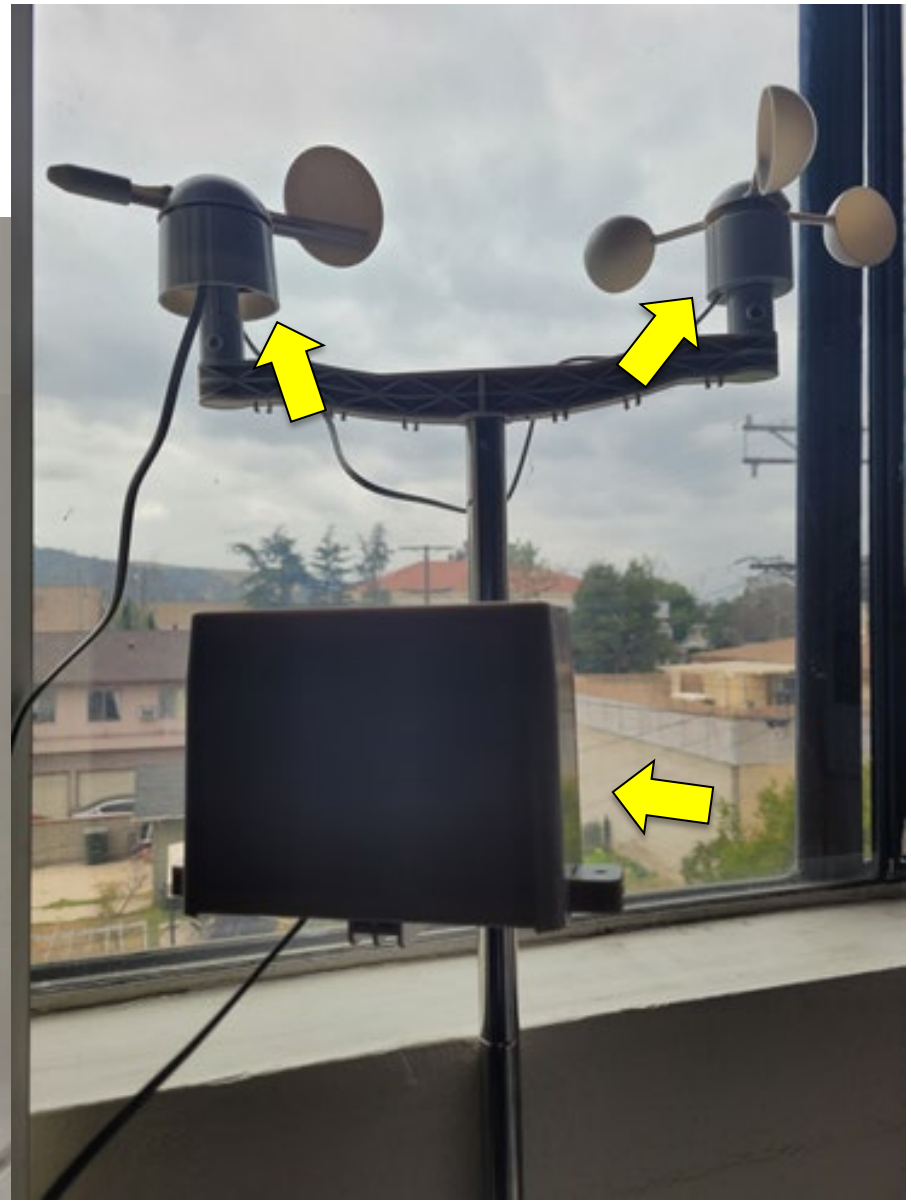
Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

Station 1 – Wired up



Station 1 – Wired up

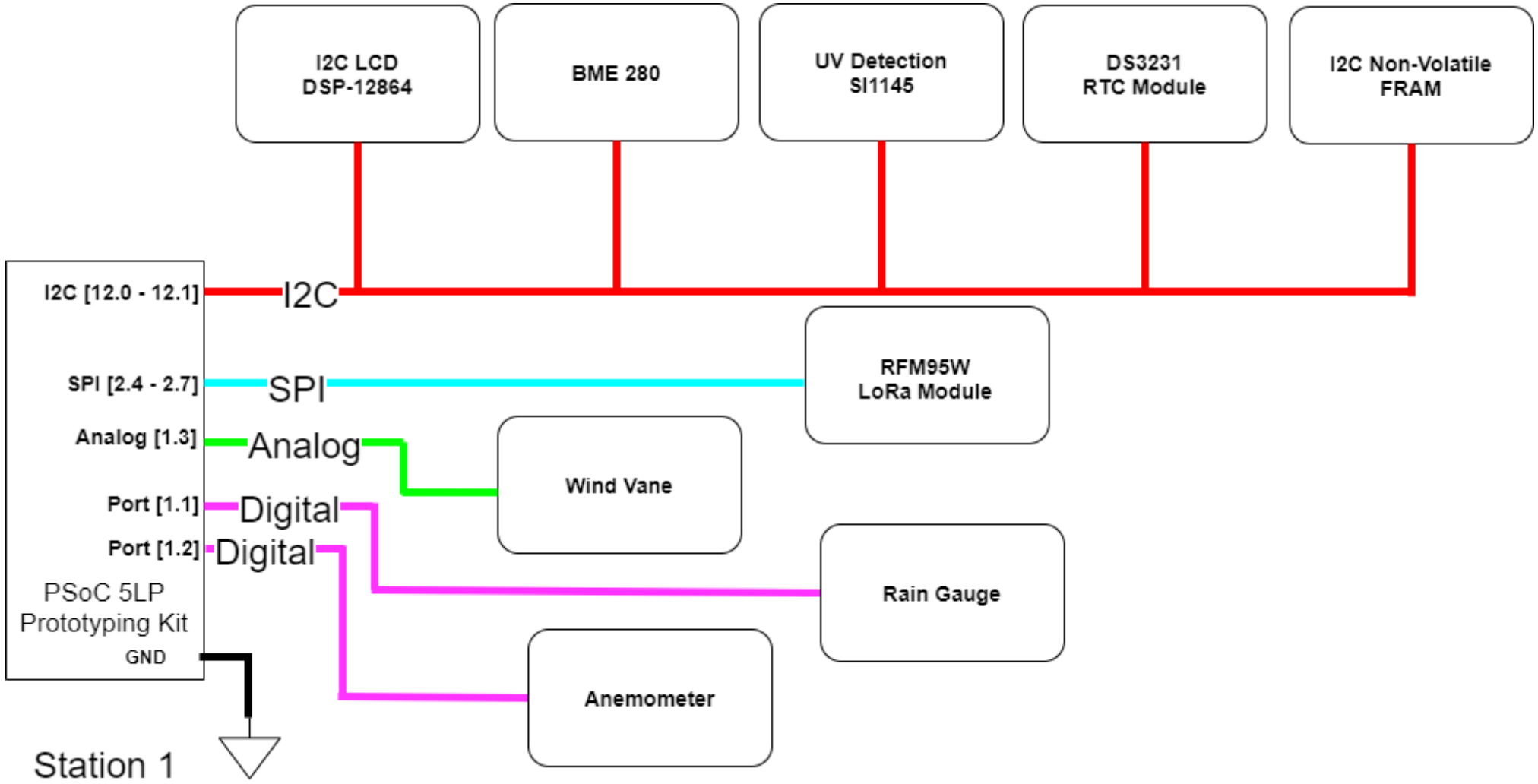




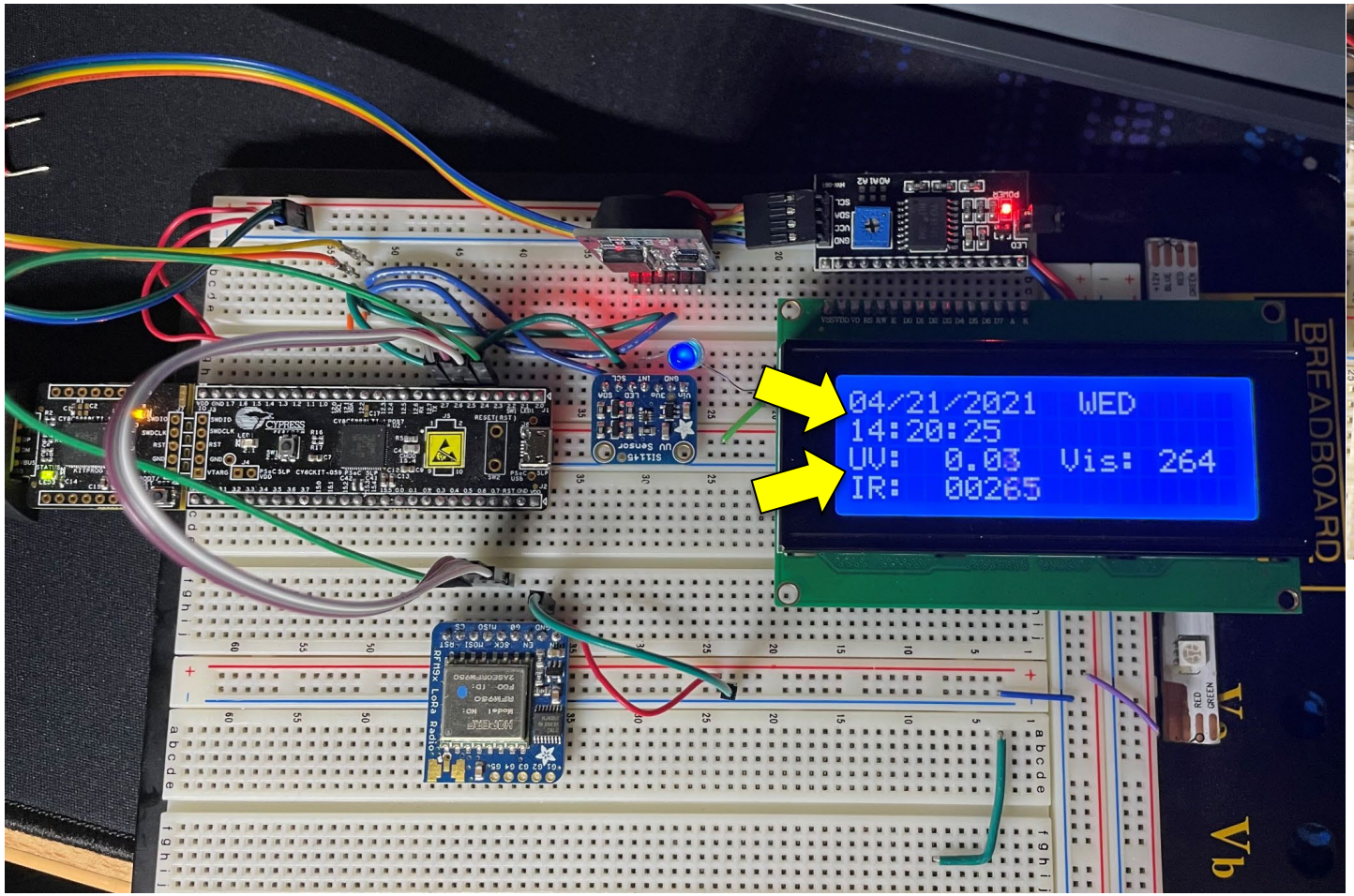
Station 1 – Sensor Information

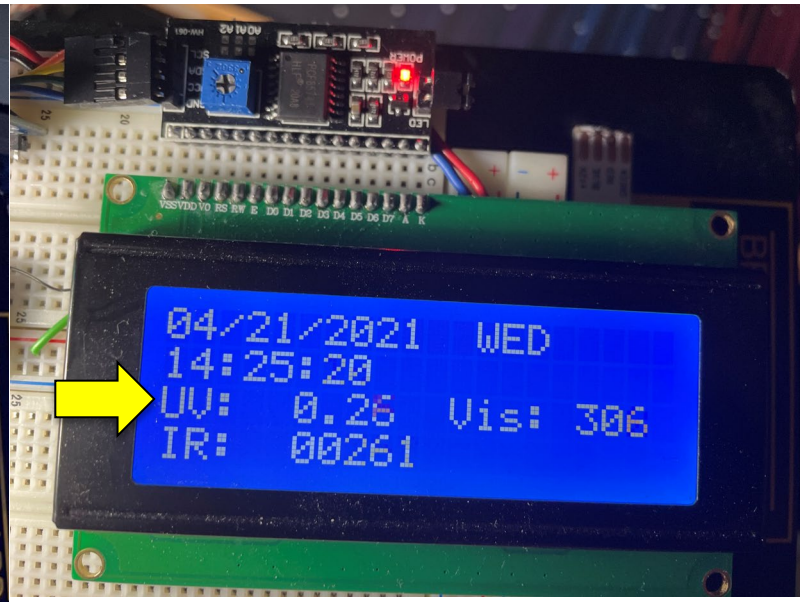
Microcontroller				
PSoC 5LP Prototyping Kit (CY8CKIT-059)				
Sensors	**Model**	**Voltage**	**Max Current**	**Interface**
LCD screen	2004A LCD Display Module & I2C Graphic LCD Adapter	3.3 – 5 V	66 mA	I2C
Temperature, Pressure & Humidity	BME 280	3.3 – 5 V	3.6 µA	I2C
Real-Time Clock	DS3231 RTC	2.3 – 5.5 V	200 µA	I2C
Ferroelectric RAM	I2C Non-Volatile FRAM	2.7 – 5.5 V	200 µA	I2C
LoRa Radio Transceiver	RFM95W LoRa Module	3.3 – 5 V	130 mA	SPI
Rain amount	Rain Gauge	5 V	0.5 mA	GPIO
Wind speed	Anemometer	5 V	0.5 mA	GPIO
Wind direction	Wind vane	5 V	0.5 mA	Analog(ADC)

Station 1 - Connection Diagram

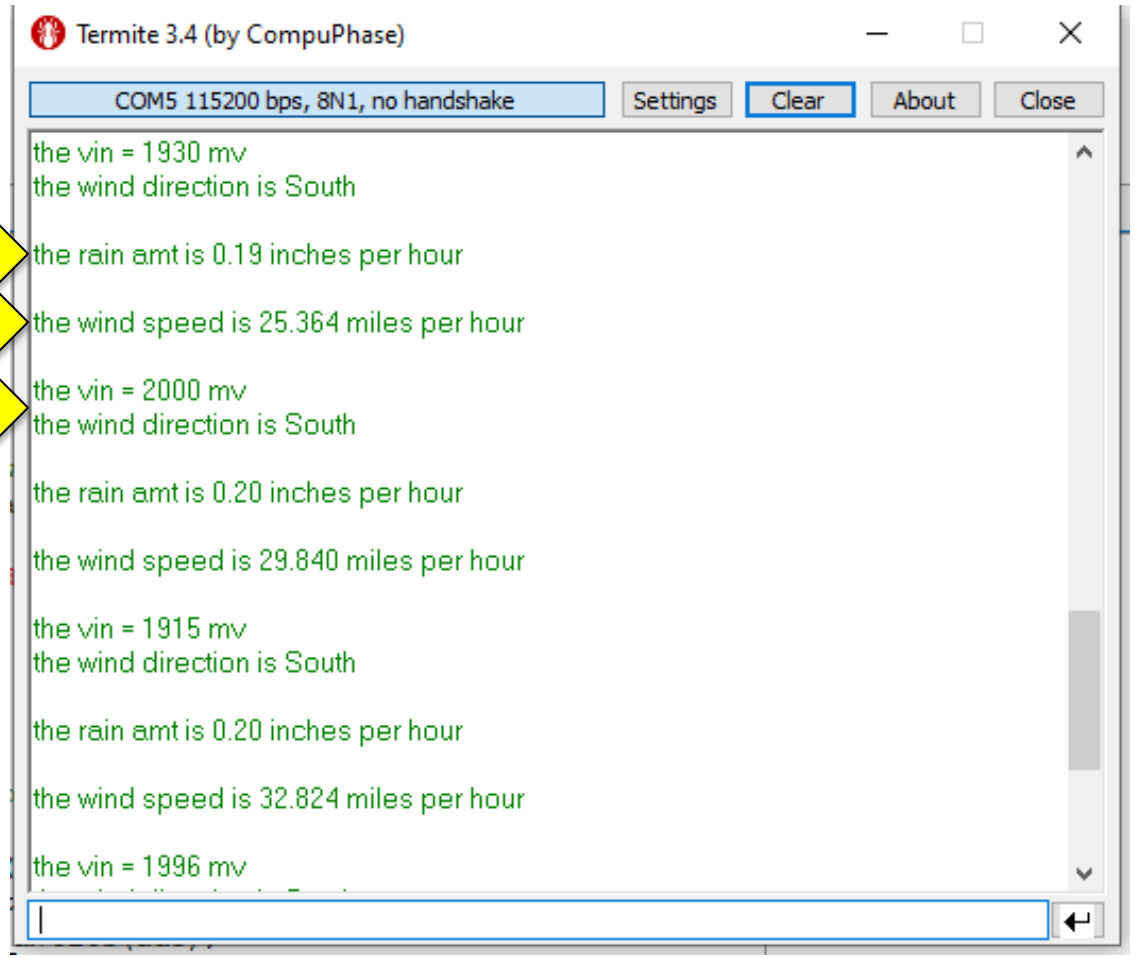


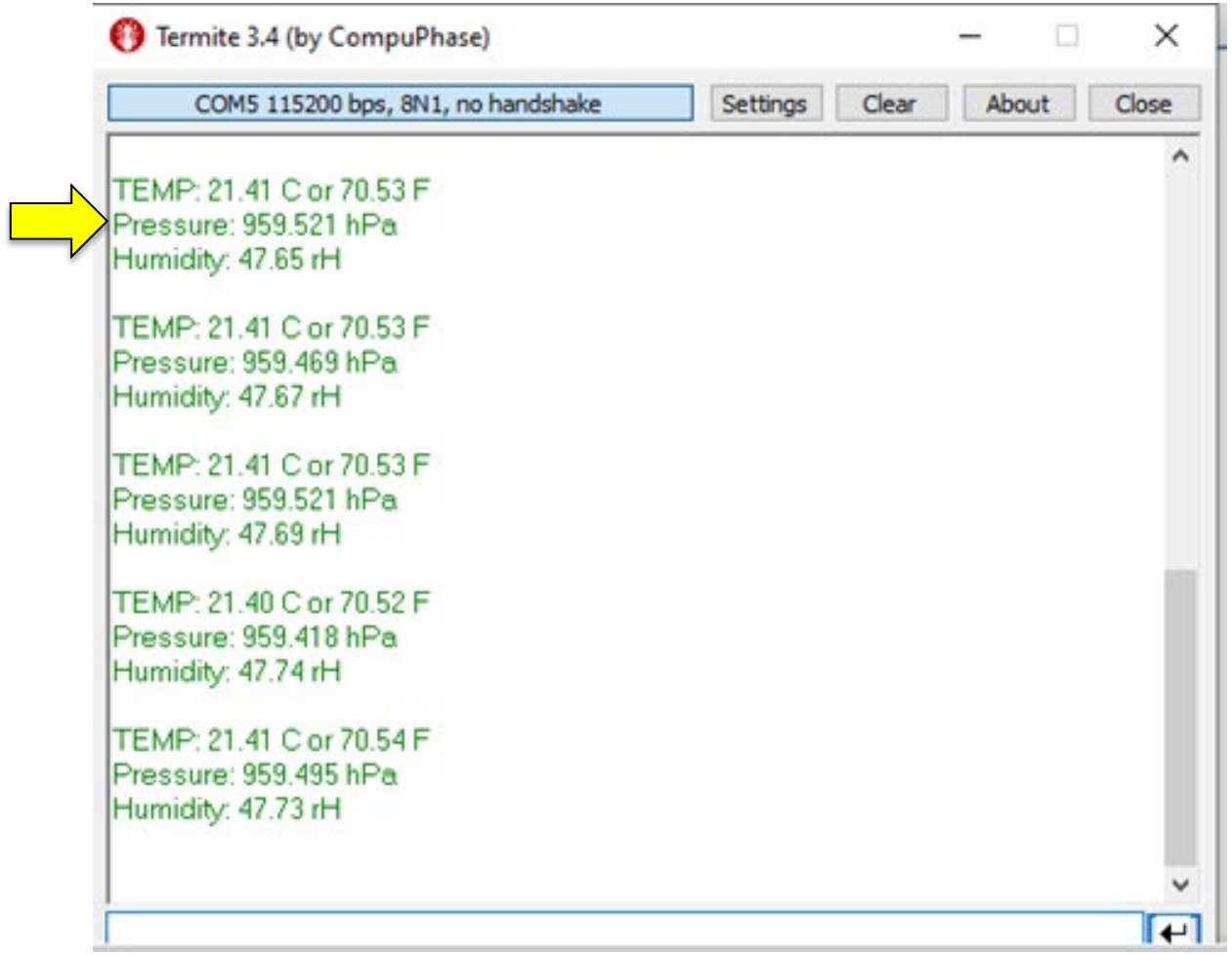
Station 1 – Testing results





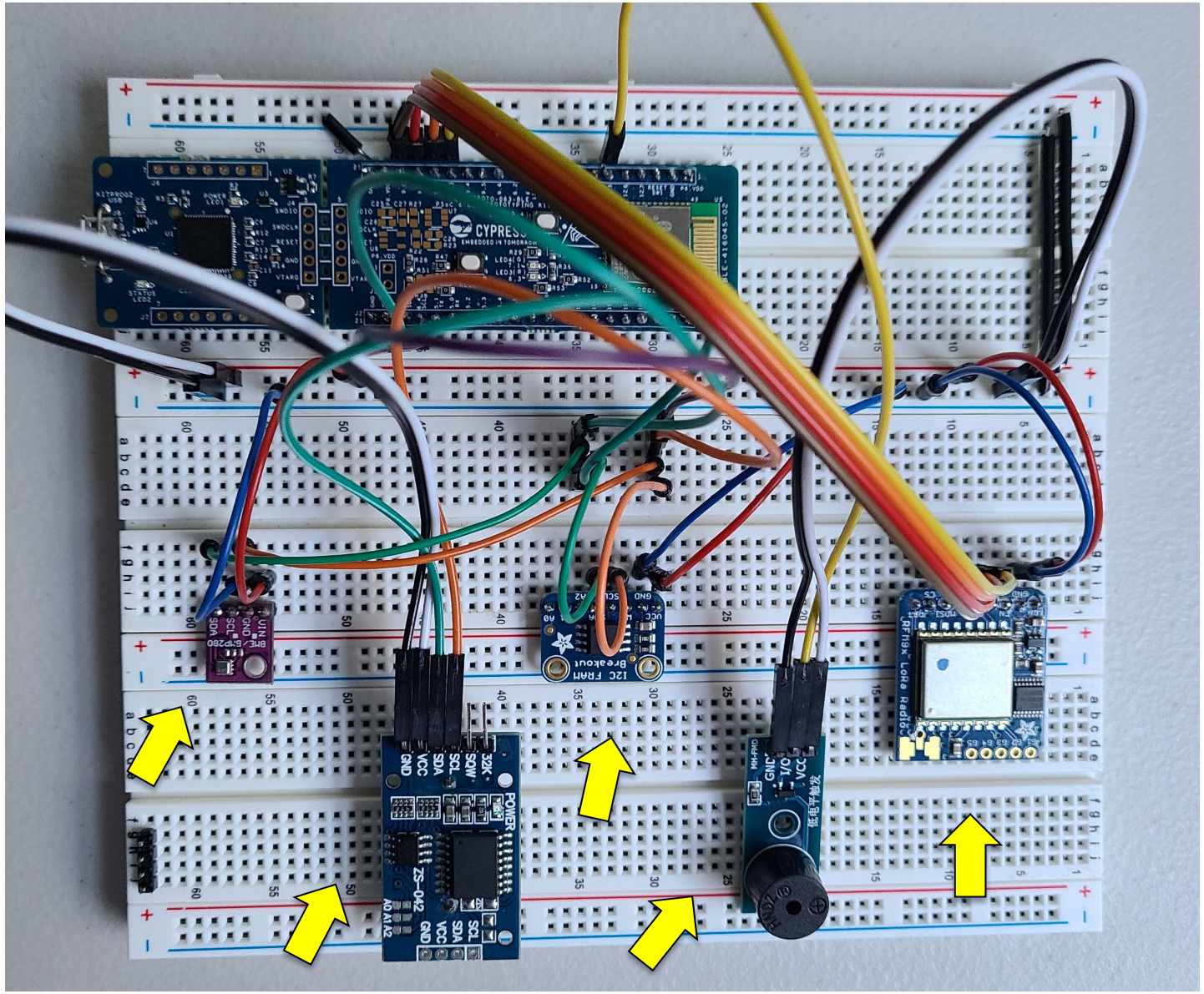
Station 1 – Testing Results cont.





Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

Station 2


CYPRESS
I2C FRAM Breakout
RFM9x LoRa Radio
POWER
ZS-042
VCC
GND
SCL
SDA

Station 2 – Details

- Location:
  - Indoors Environments

- Data Displayed by LCD Screen

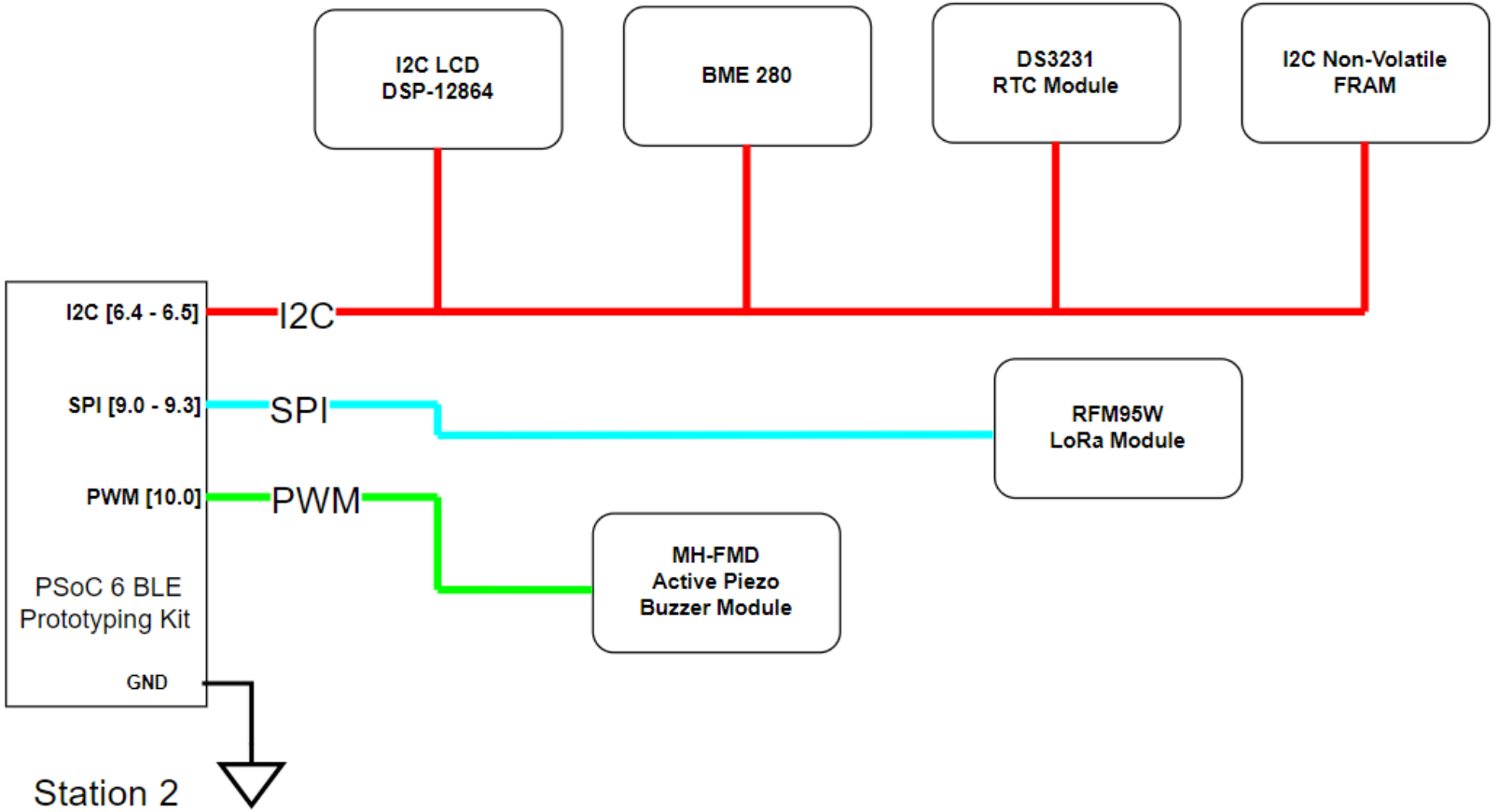
  - Set One:
    - Data from station 2 sensors

  - Set Two:
    - Data from Station 1 and 3
    - Sent from Central Station (Raspberry Pi) via LoRa Module

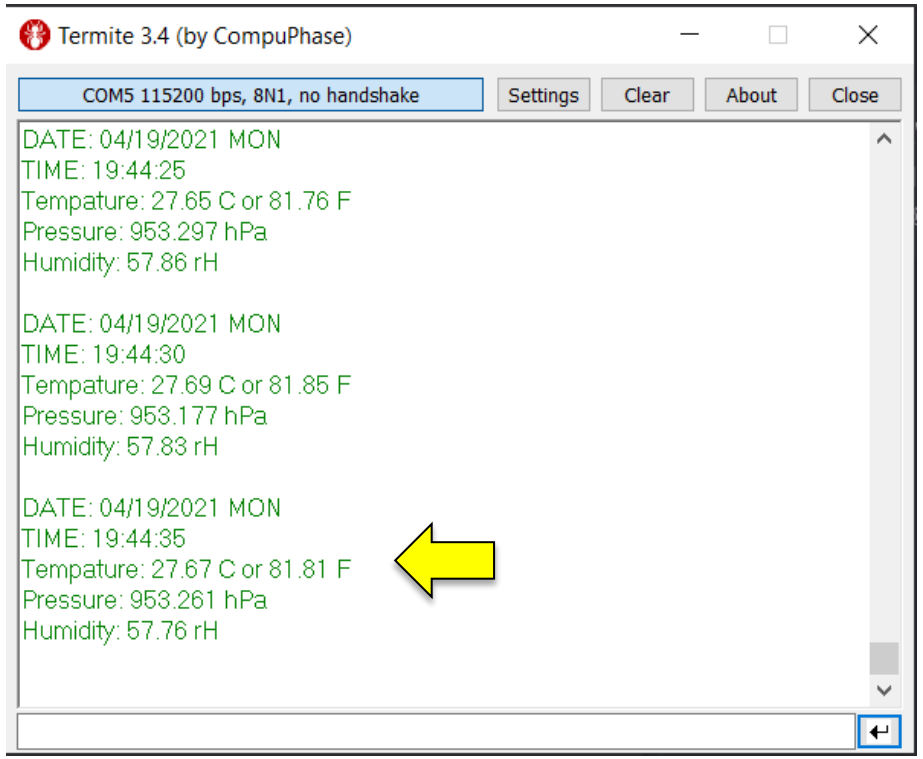
Station 2 – Sensor Information

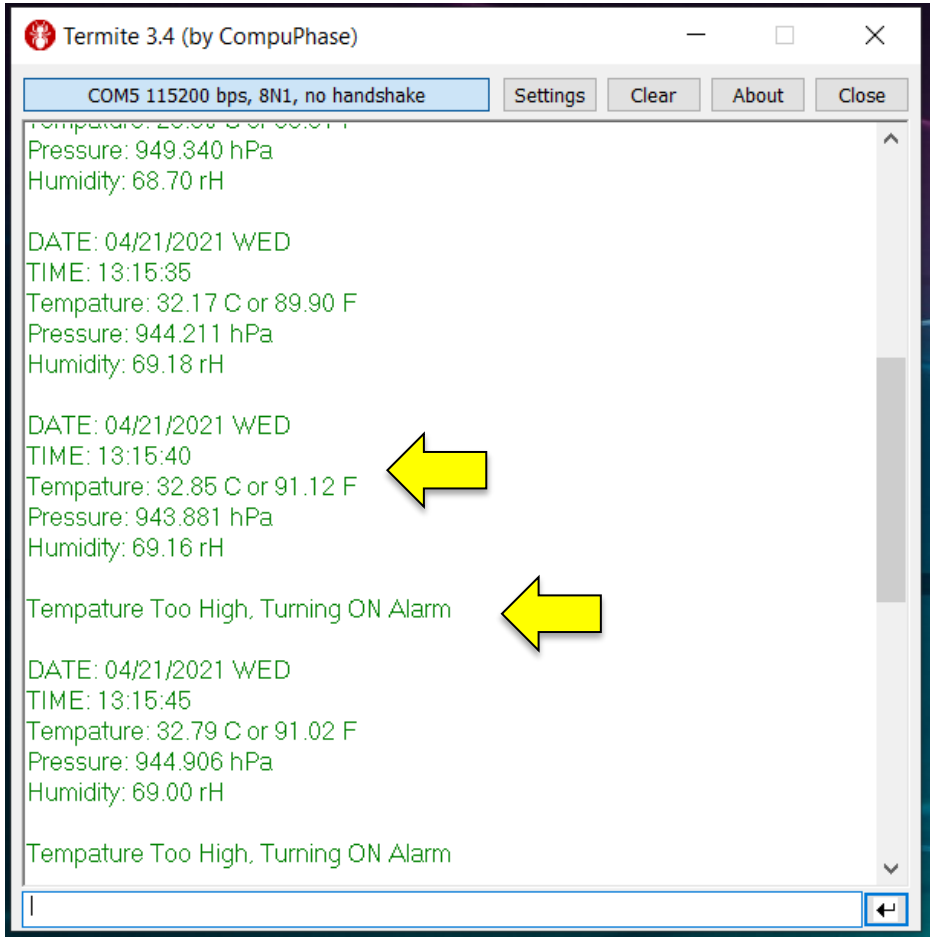
Microcontroller				
PSoC 6 BLE Prototyping Kit (CY8CPROTO-063-BLE)				
Sensor	**Model**	**Voltage**	**Max Current**	**Interface**
LCD Display	12864 LCD Display Module	3.3 – 5 V	66 mA	I2C
Temperature, Pressure & Humidity	BME 280	3.3 – 5 V	3.6 µA	I2C
Buzzer Alarm Module	MH-FMD	3.3 – 5 V	30 mA	PWM
Real-time Clock	DS3231 RTC	2.3 – 5.5 V	200 µA	I2C
Ferroelectric RAM	MB85RC256V FRAM Chip	2.7 – 5.5 V	200 µA	I2C
LoRa Radio Transceiver	RFM95W LoRa Module	3.3 – 5 V	130 mA	SPI

Station 2 – Connection Diagram



Station 2 – Results





Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

Station 3

Micro controller	PSoC 5LP Prototyping Kit (Cy8CKIT-059)			
Component	Voltage	Current	Interface Type	Sensor Features
CJMCU-811 CCS811	1.8 - 3.6V	30mA	I2C	CO2 Sensor
MQ-4	5.0V	0.25mA	Analog	Gas, CNG Sensor
PPD42NJ	5V	2.5µA	ADC	Dust Particle Sensor
DSP-1182 LCD Screen	3.3 - 5V	66 mA	I2C	20x4 LCD Display
MB85RC256V	2.7 - 5.5V	200 µA	I2C	256K-Bit FRAM
RFM95W Lora Module	3.3-5V	130mA	SPI	RF long range transceiver

Figure 1: Station 3 system specifications

Station 3 - Connection Diagram

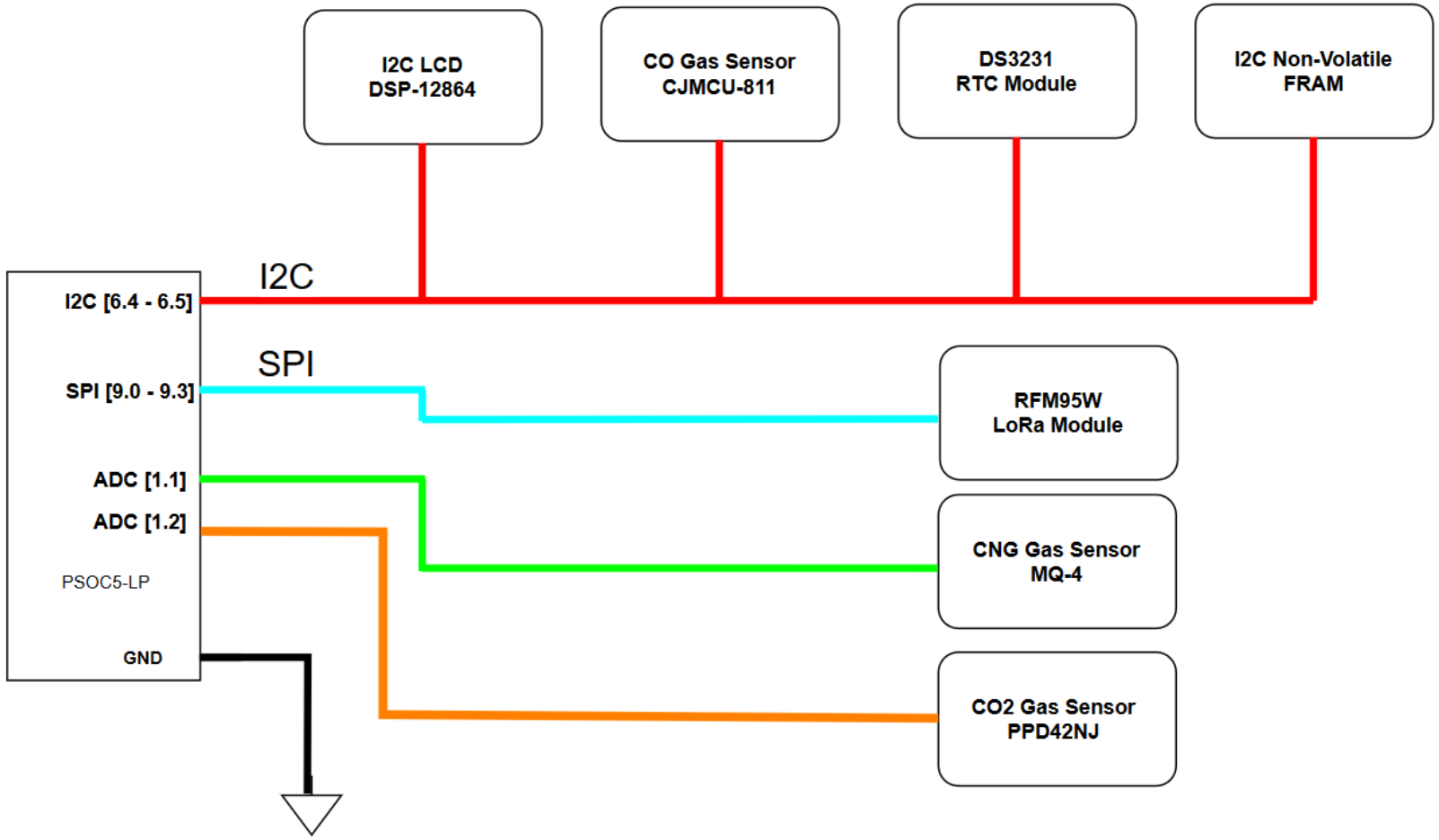


Figure 1: Station 3 Schematics

Station 3 – Test Bench

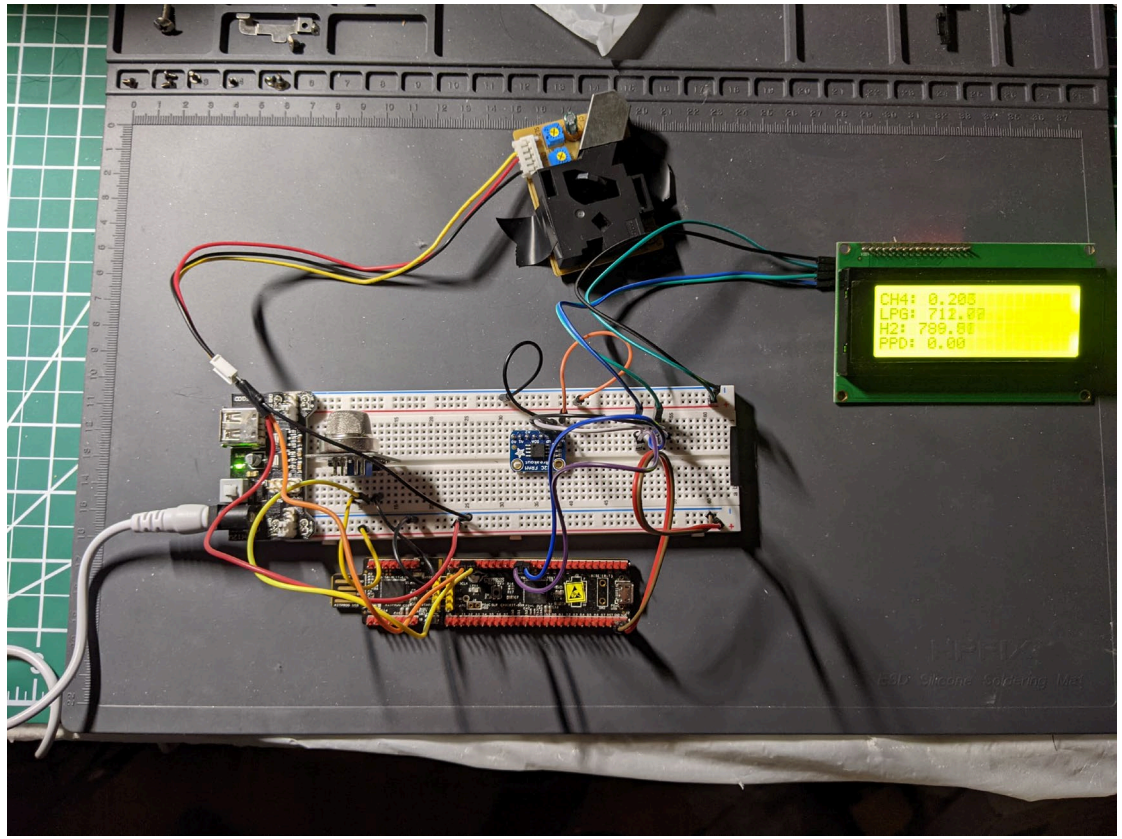


Figure 1: Station 3 test station

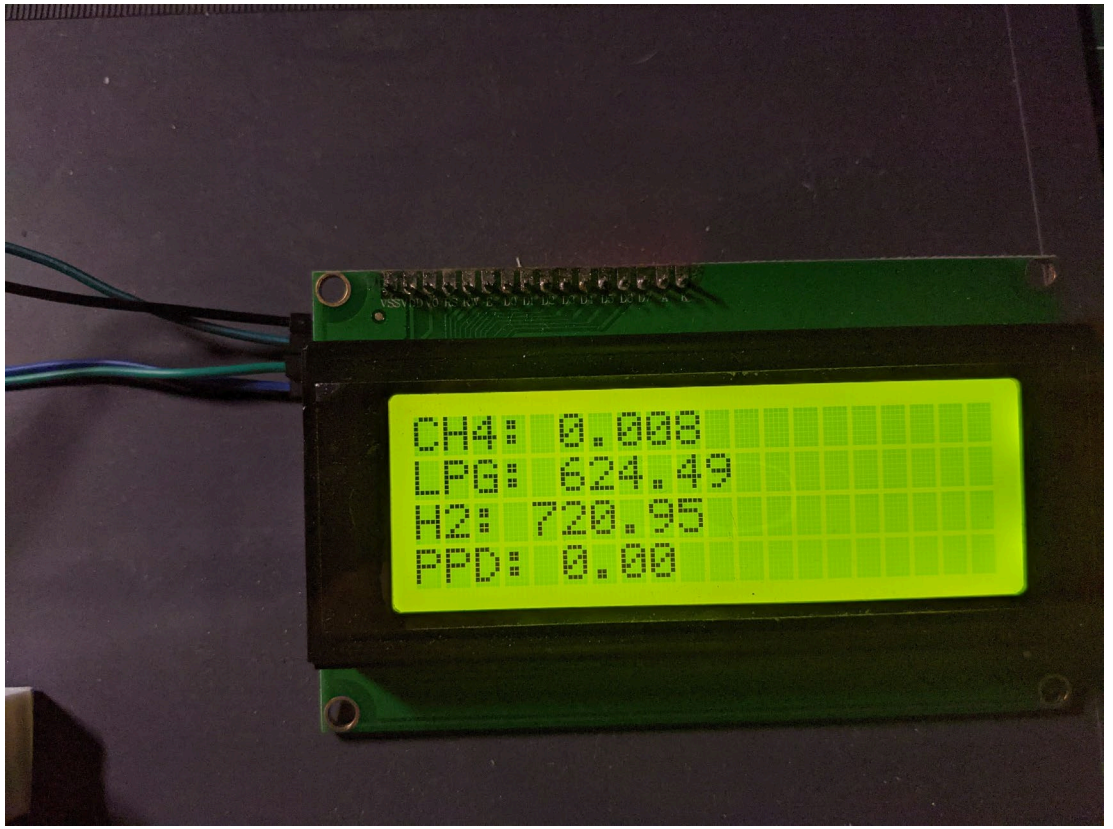


Figure 2: Close up of LCD output from station 3

Station 3 – Test Bench

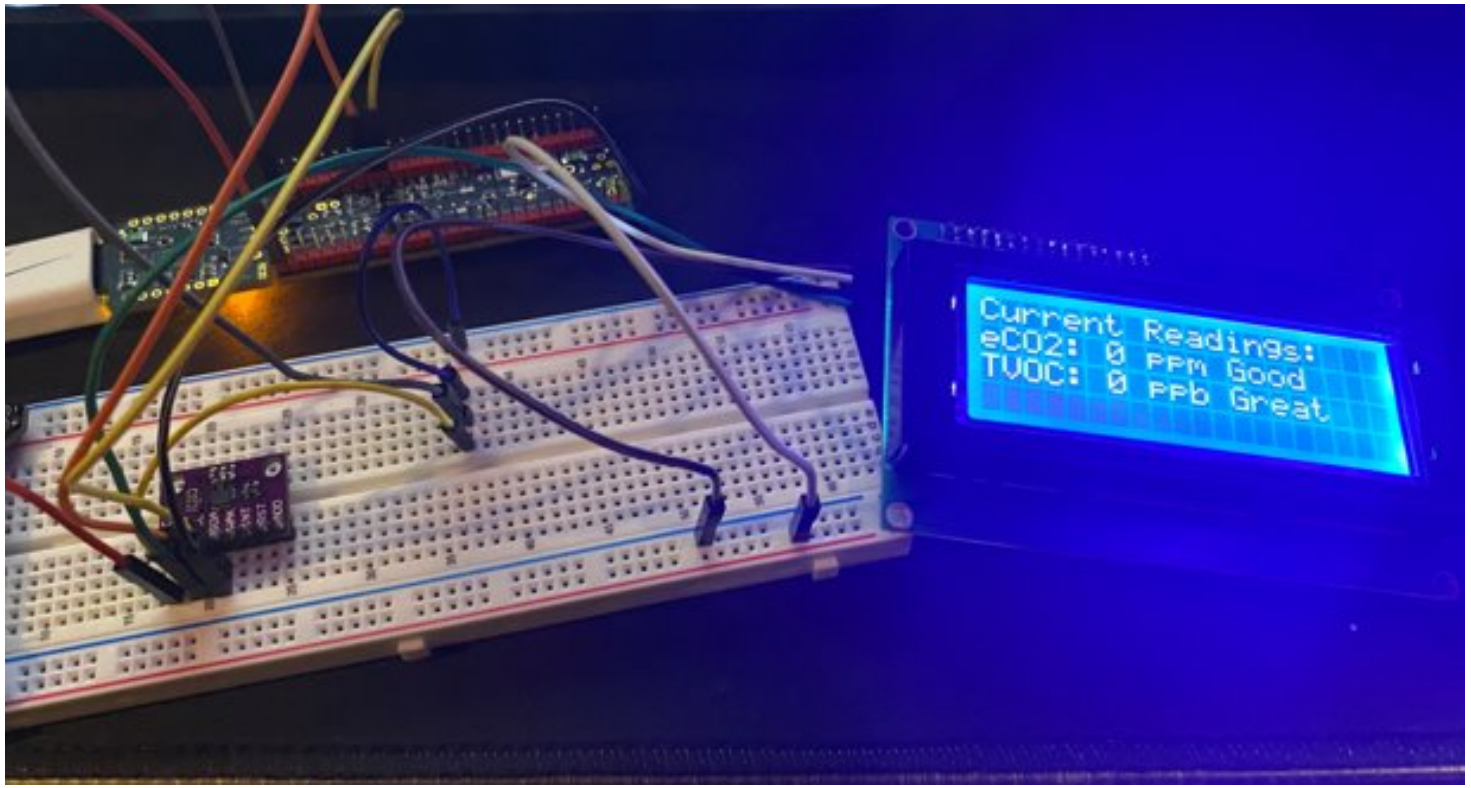


Figure 1: CJMCU-811 sensor running with output

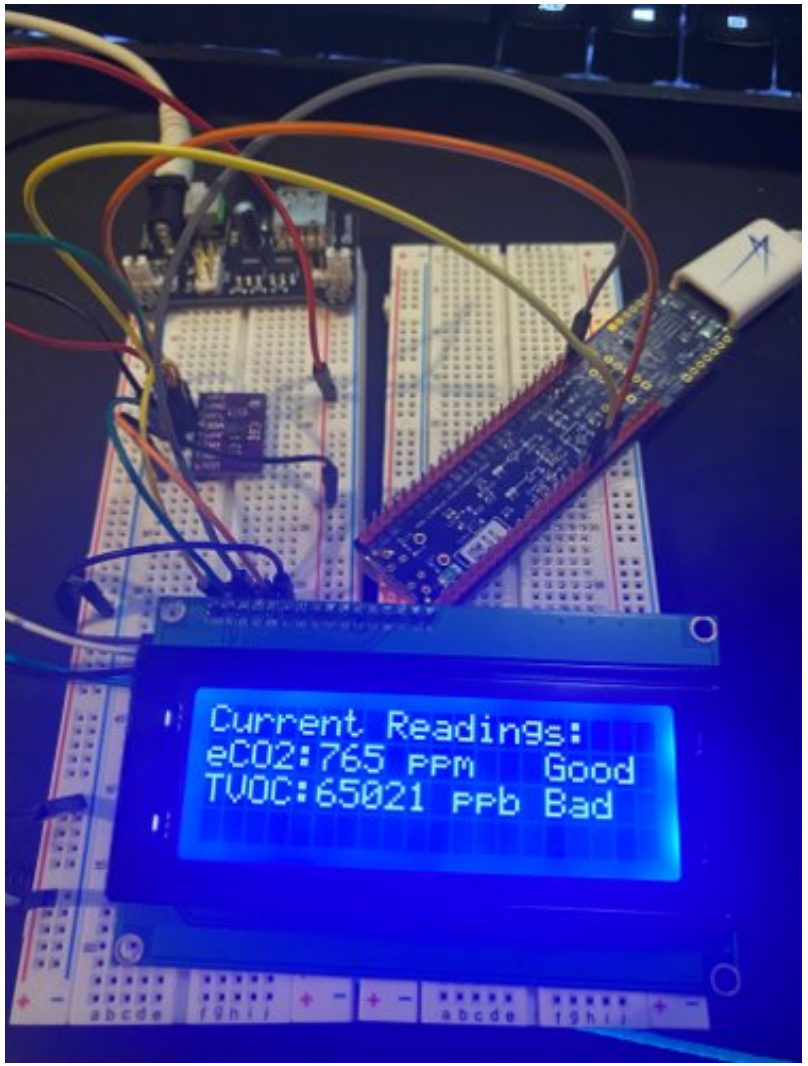


Figure 2: CJMCU-811 test bench

Station 3 - Results

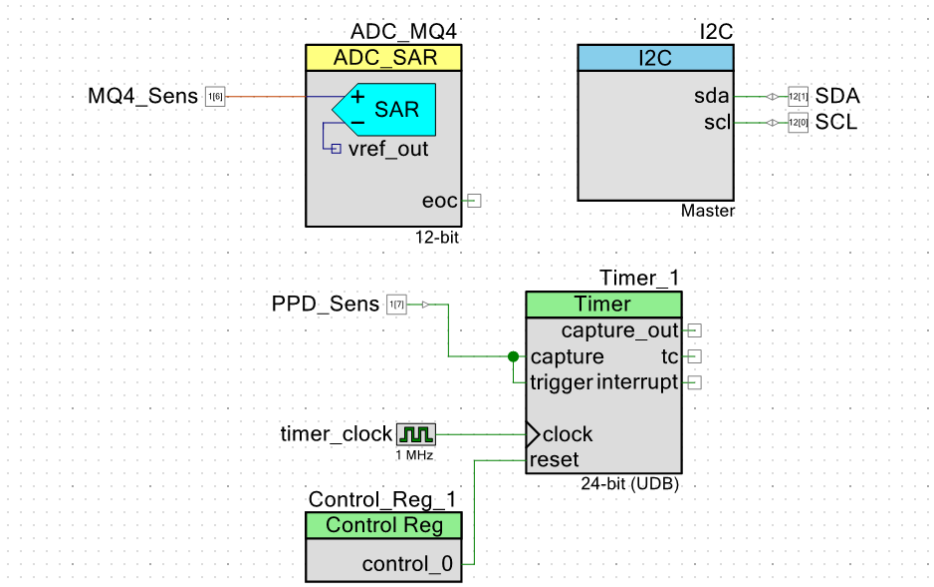


Figure 1: Top Design of Station 3

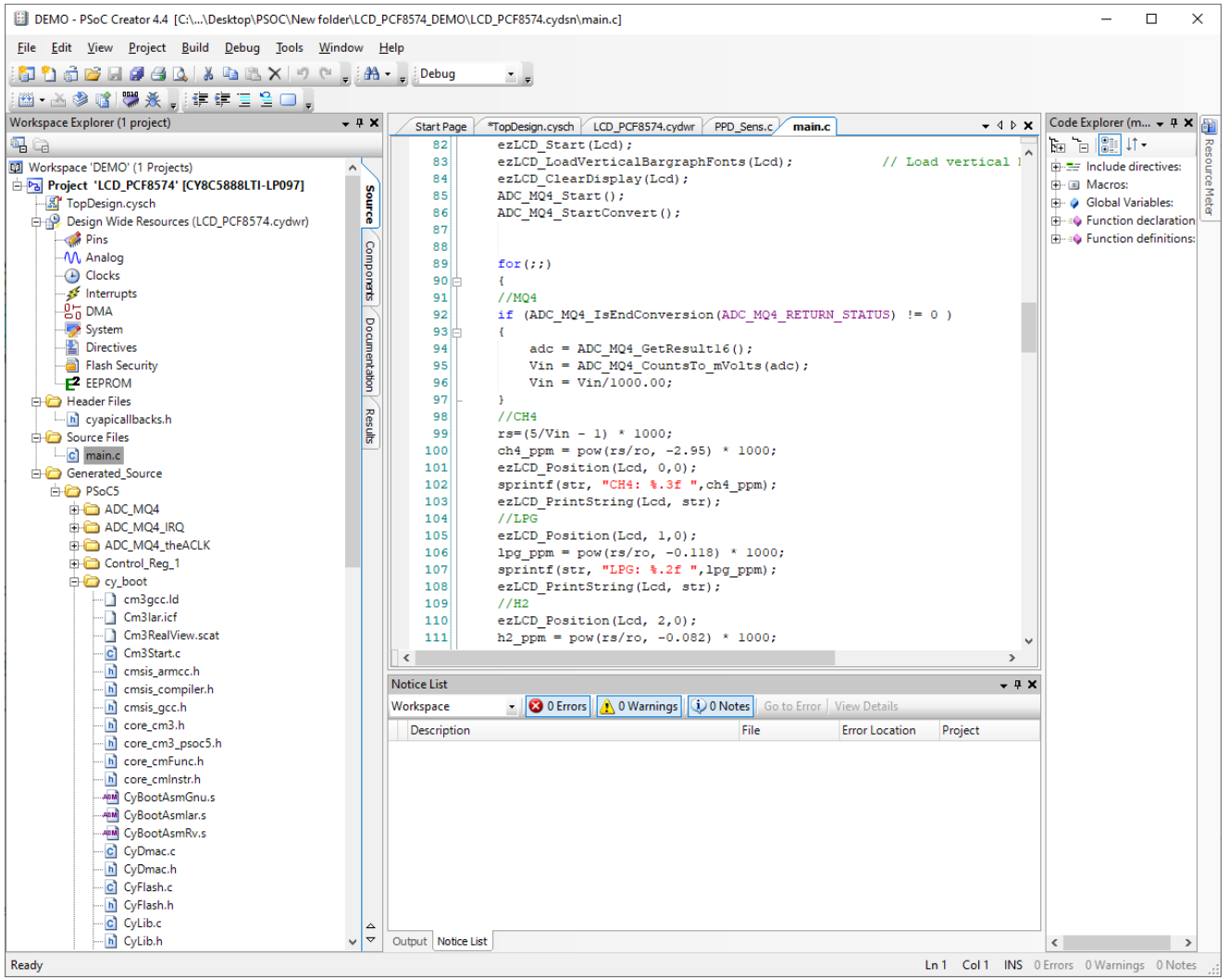


Figure 2: Runtime code of station 3

Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

AWS Amazon Web Services (A.W.S)

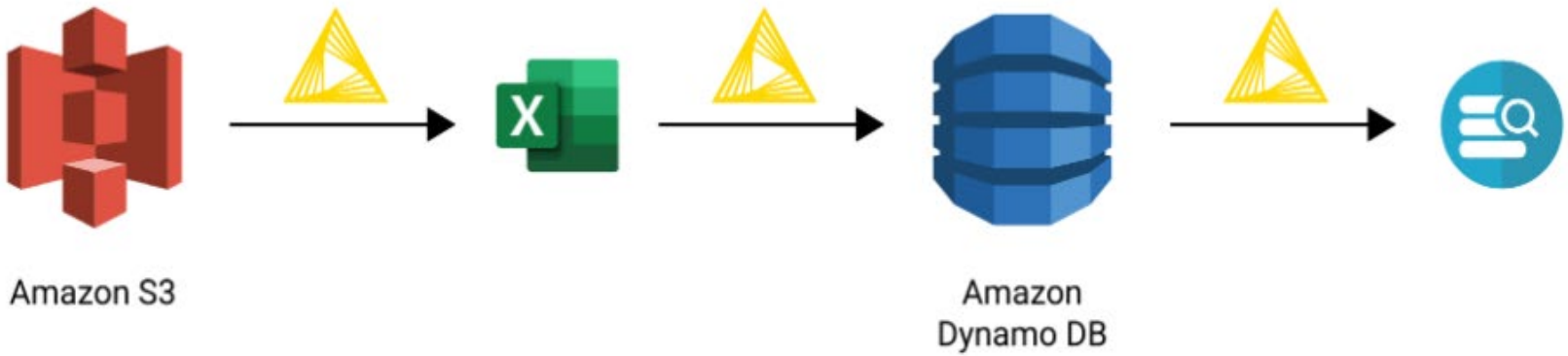
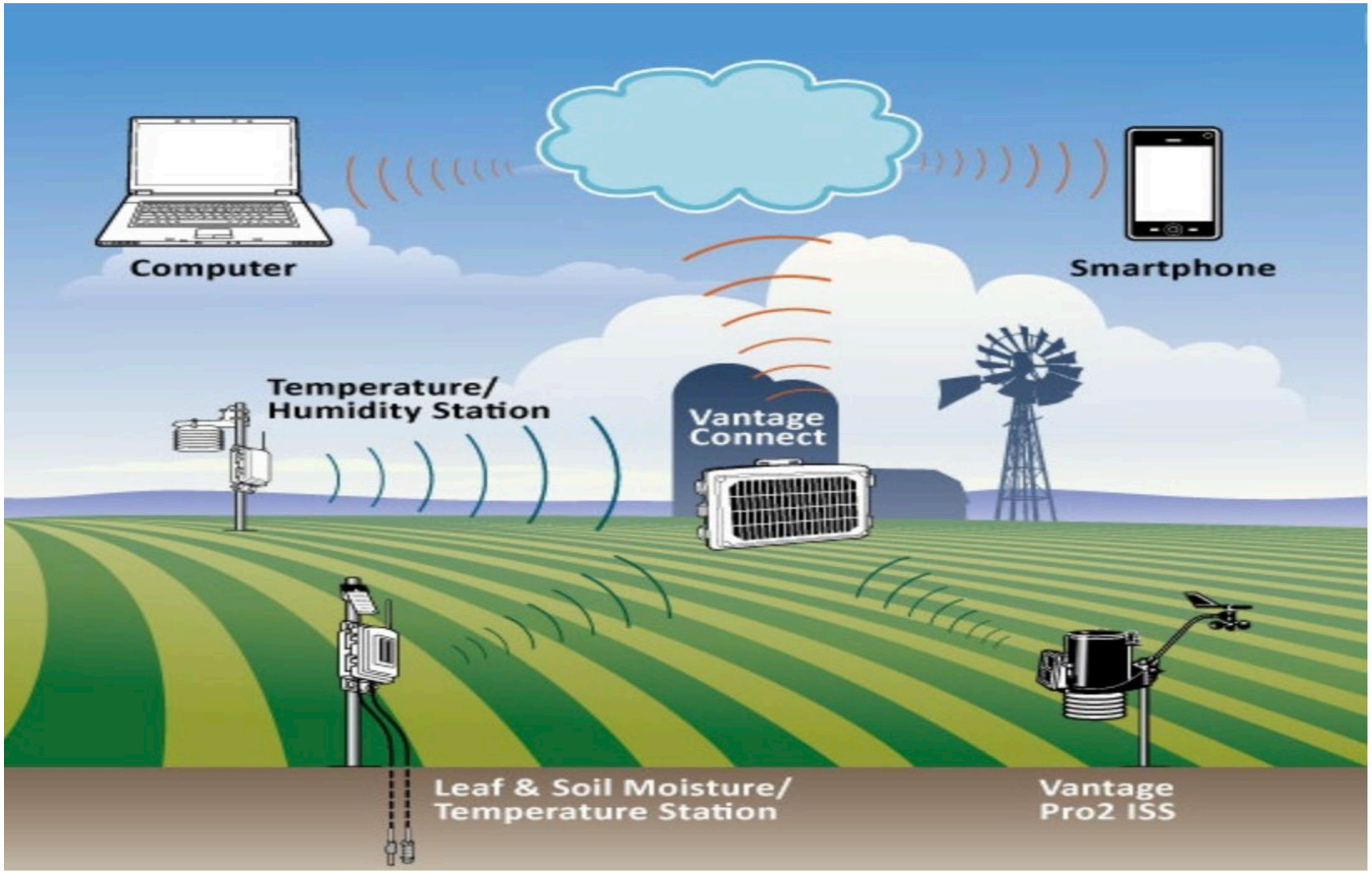


Figure 1

The image below *(figure1)* explains how data can be retrieved in real- time and can be stored in the database like Amazon Dynamo DB to help notify the humans about disastrous weather situations.

INTERNET OF THINGS IOT



- The **Internet of Things** (**IoT**) refers to a system of interrelated, internet-connected objects that are able to collect and transfer data over a wireless network without human intervention.

Storing data to Database

```python
import boto3

dynamodb = boto3.resource('dynamodb')
dynamodbTable = dynamodb.Table('test_table')

dynamodbTable.put_item(
#boto3.session('dynamodb', region_name='us-west-2', endpoint_url='https://18.191.139.28/2021/02/22/hello-world/')
    # storing sensor data in the form of dictionary
    Item={
    'WeatherStation': 'Cal State Los Angeles',
    'Station number': 1,
    'Location': 'Los Angeles CA'
```

Text ▾ DynamoDB JSON

```
{
  "weather Station": "CalsSateLa",
  "temperature": "60 degrees",
  "station number": "station 1"
}
```

- Storing data to AWS by using python SDK.
- Data storage format are structured in strings.

Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	---

Raspberry Pi 4

- In order to process the data set packets we used Python with Visual Studio Code.

- In order to connect to AWS IoT Core, we download necessary libraries from Raspberry Pi command terminal.

- Create certificates public and private keys.

```
# For TLS mutual authentication
myMQTTClient = AWSIoTMQTTClient("DesignID")
myMQTTClient.configureEndpoint("a5e1cu78wbs20-ats.iot.us-west-1.amazonaws.com", 8883) #Provide your AWS IoT Core endpoint (Example: "abcdef1
myMQTTClient.configureCredentials("/home/pi/certs/Amazon-root-CA-1.pem", "/home/pi/certs/private.pem.key", "/home/pi/certs/device.pem.crt")
myMQTTClient.configureOfflinePublishQueueing(-1)
myMQTTClient.configureDrainingFrequency(2)
myMQTTClient.configureConnectDisconnectTimeout(10)
myMQTTClient.configureMQTTOperationTimeout(5)
```

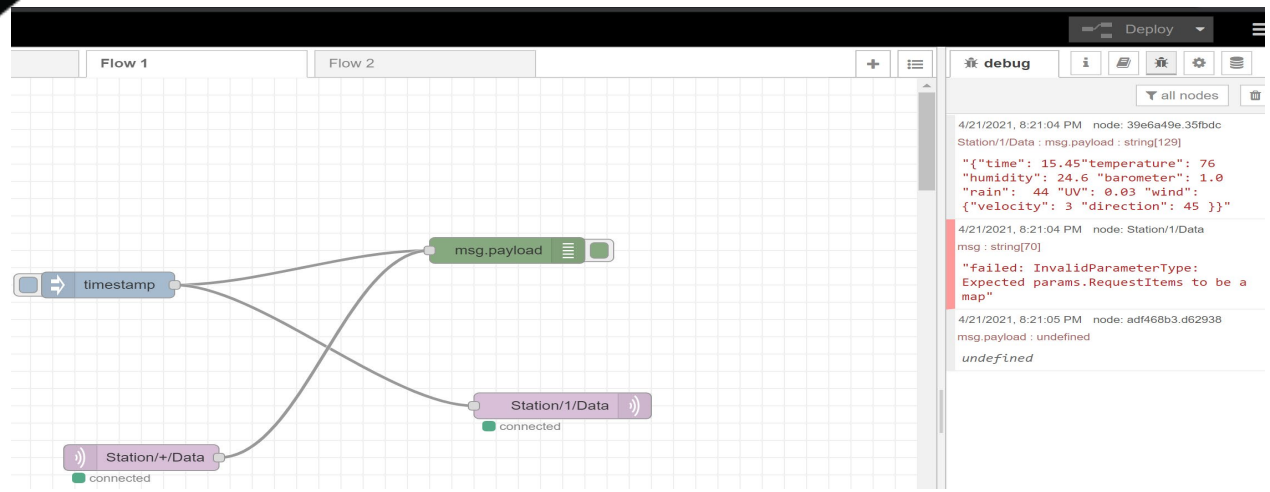
Data Set Notation

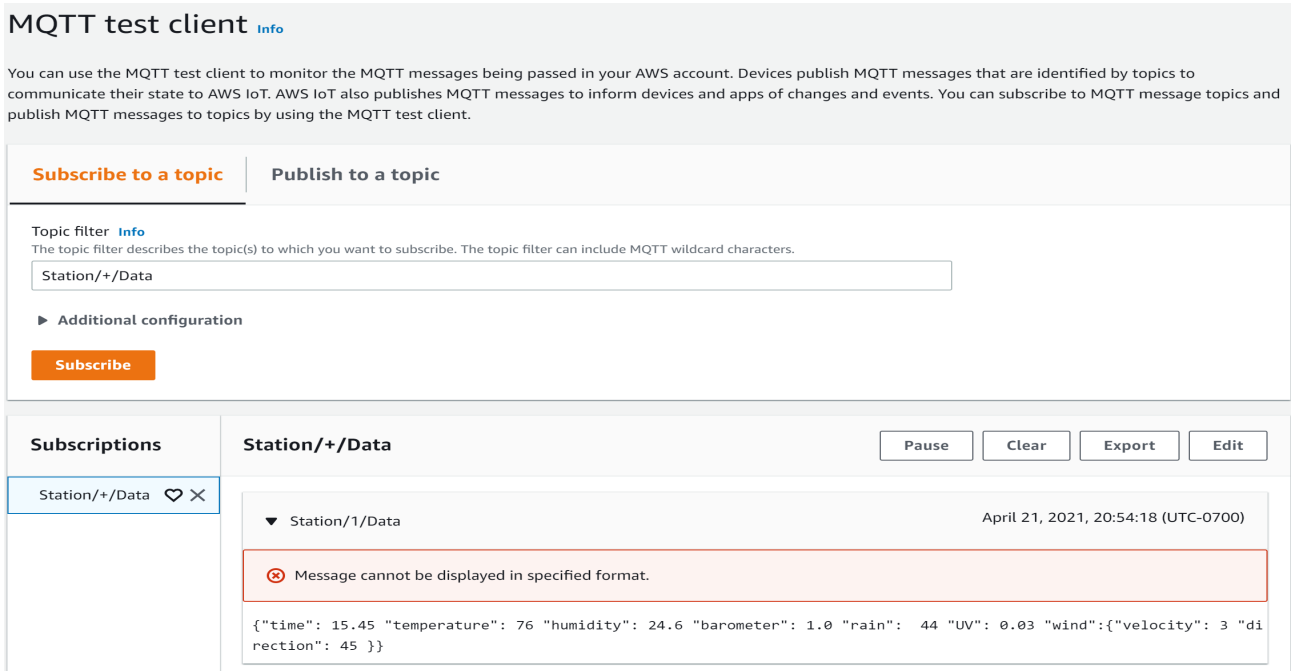
```
test_1 = 'Station_1| "time": 15.45, "velocity": 3 , "direction": 45 , "rain":  44 , "UV": ' \
    '0.03 , "temperature": 76 , "humidity": 24.6 , "barometer": 1.0 '
test_2 = 'Station_2| "time": 15.46 , "temperature": 76 , "humidity": 27.6 , "barometer":  2.0 '
test_3 = 'Station_3| "time": 15.47 , "methane": 23 , "particle": 66 , "cs811": -127 '
```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
pi@raspberrypi:~/aws-iot-device-sdk-python-v2 $  cd /home/pi/aws-iot-device-sdk-python-v2 ; /usr/bin/env /
b/python/debugpy/launcher 33191 -- /home/pi/aws-iot-device-sdk-python-v2/topics2.py
Initiating IoT Core Topic ...
Station_1
"time": 15.45
"velocity": 3 km/hr
"direction": 45 degrees
"rain":  44 mm
"UV": 0.03
"temperature": 76 C
"humidity": 24.6 %
"barometer": 1.0 psi
Publishing Message from Raspberry PI
pi@raspberrypi:~/aws-iot-device-sdk-python-v2 $
```

- The string must be formatted in a particular structure.
- Quotation marks on the environmental variable.
- String will be sent without variable units.

Sending Data Sets

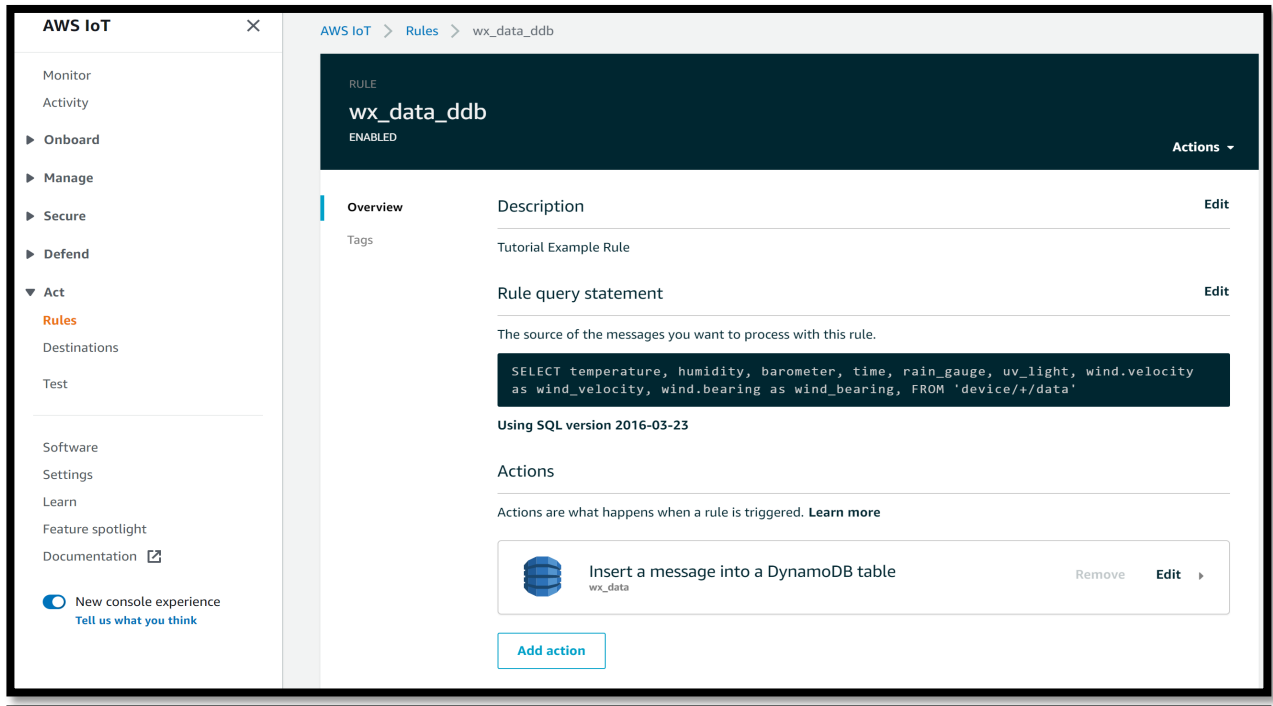




- To successful identify if a data set was sent, we used Node Red.
- In order to receive the data sets, we created topics for the stations to subscribe to and publish data.
- Within these topics there are rules that detect key words or variables.

IoT Core

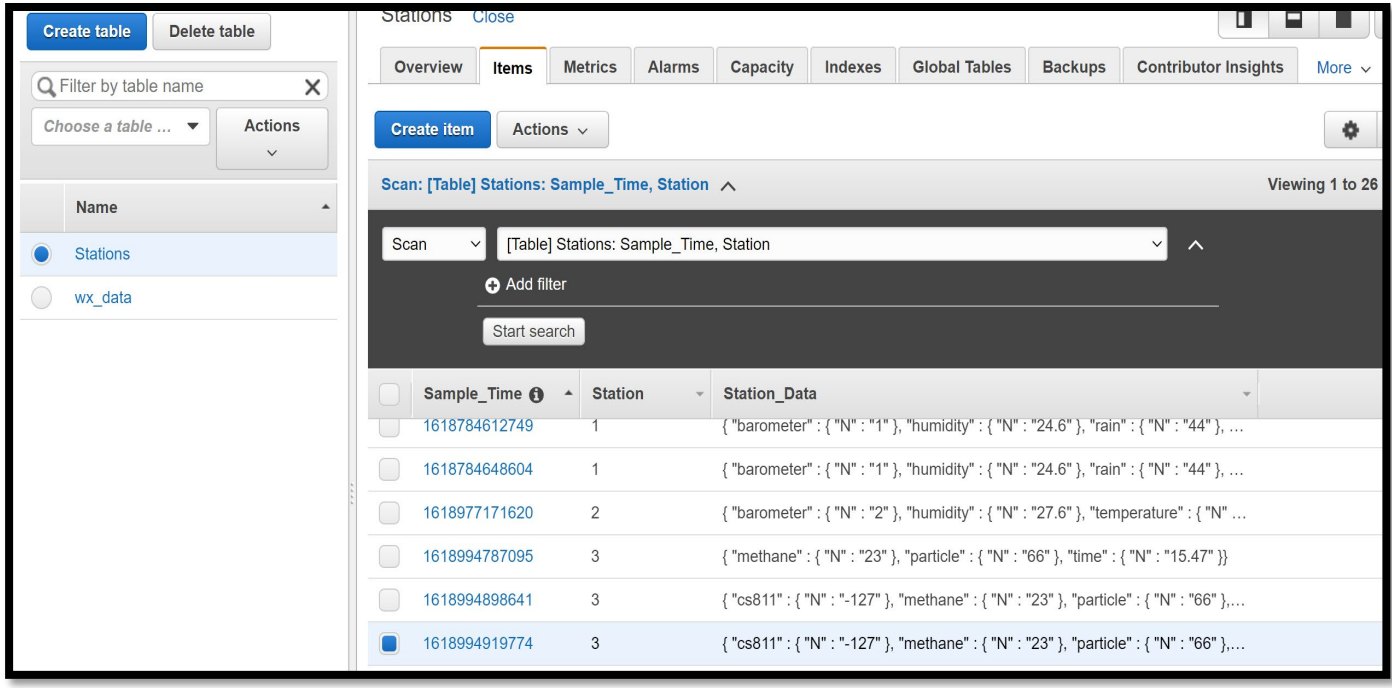
- IoT Core allows our team to rules which can identify environmental variables from a particular topic.
- Once variables are identified from MQTT client the rules in place will store the data sets into DynamoDB.
- In order for the Raspberry Pi to subscribe to IoT Core a policy must be created.

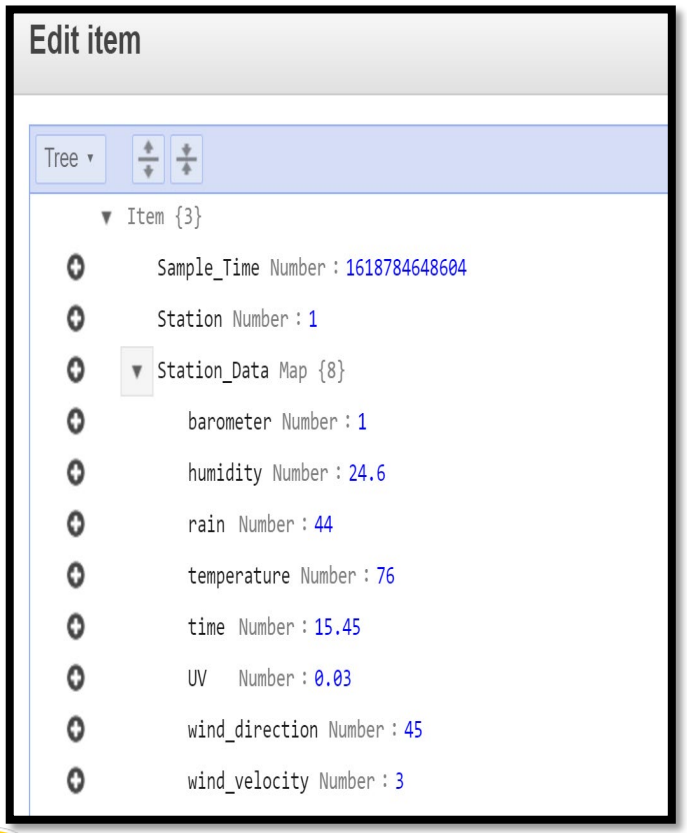


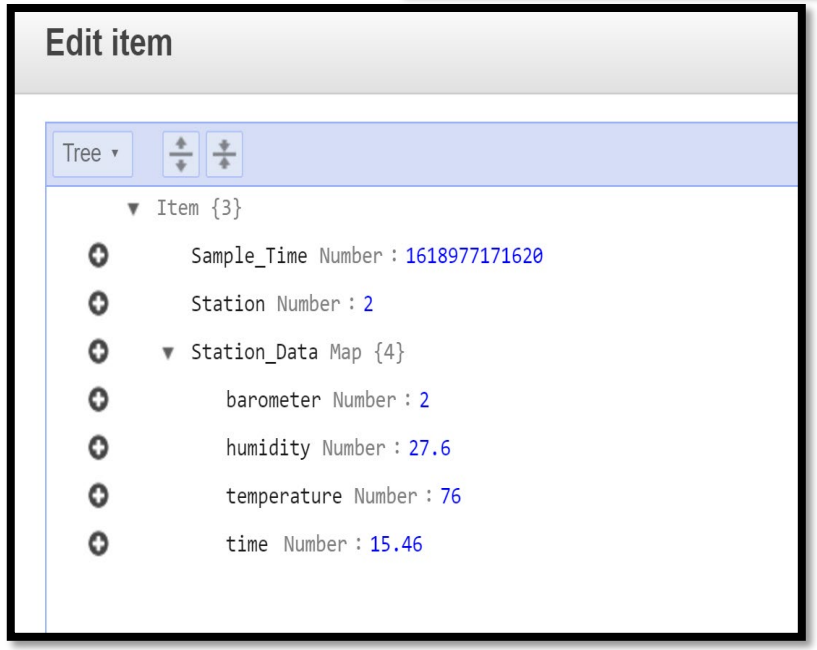


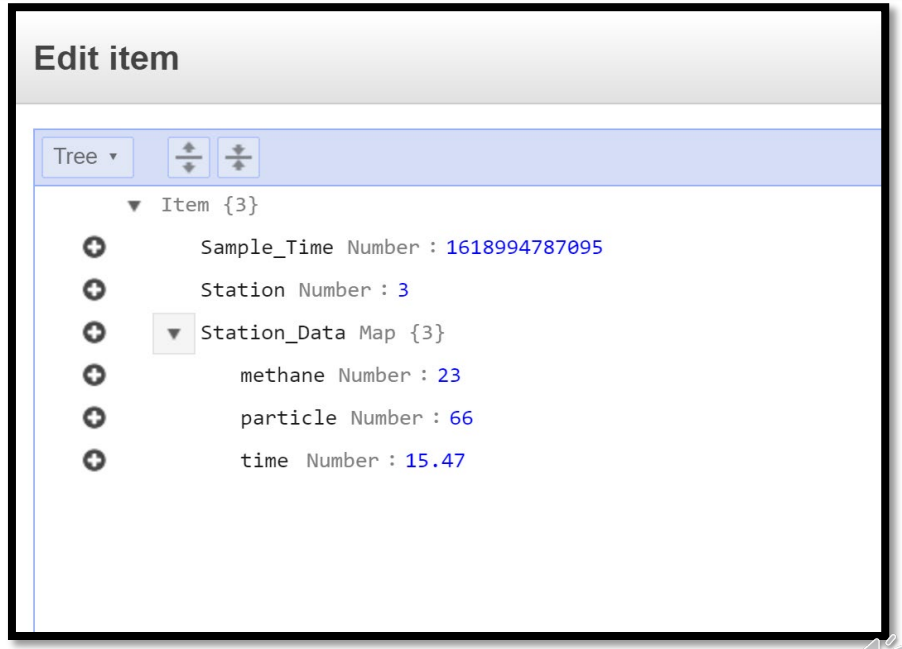
Data Base

- DynamoDB is used to store the sets according to the station number and environmental variables.









Agenda	Team Member
Backgrounds, Problem, and Objectives	Juan Avila
System Overview	David Hernandez
Implementation – Station 1	Anthony Huynh
Implementation – Station 2	Jeffrey Espinoza
Implementation – Station 3	Vincent Oviedo
Implementation – AWS database/Raspberry Pi Part 1	Akbar Rizvi
Implementation – AWS database/Raspberry Pi Part 2	Jose Rodriguez
Conclusion	Juan Avila

Conclusion – Accomplishments

- Built 4 stations and a database
- Build a wireless network of sensors
- Collect indoor and outdoor environmental data



Conclusion – Issues

- Limited due to Covid19 and remote learning, we had some limitations on what we could implement



Conclusion – Going Foward

- With the way we designed it we can add more different sensor types or more stations for scalability



Thank you

Any Questions?