NORTHERN ARIZONA UNIVERSITY

School of Informatics, Computing, and Cyber Systems

To: Dr. Robert Severinghaus, EE-486
From: Team 13 - NAU Research Greenhouse
Date: March 26, 2021
Subject: Testing Results Report

The NAU Research Greenhouses support university-wide research. Over the past 30 years, they have supported projects that have encompassed many disciplines and departments, including: Biology, Chemistry, Ecological Monitoring and Assessment, Ecological Restoration Institute, Engineering, Environmental Sciences, and Forestry. Each of the eight greenhouses supports a number of different research projects simultaneously from diverse clients such as students, NAU researchers, and external customers. The greenhouses have control systems installed for changing temperature, and there are old monitors with that system for temperature and humidity that are inconveniently located for real-time monitoring. Customers typically need a record of the temperature and humidity for their projects, but it cannot be provided real time. A few temperature control system cabinet Sensor boxes for existing system projects require a light level and we don't have that capability at all. Future projects will use high humidity enclosures inside the greenhouse, and there is no monitoring system in place for that.

We have done two inspection tests and one unit test per each of the sensors, we tested to make sure the sensors had low hysteresis, high accuracy, and low power usage while reading. Our sensors we have been testing since October when we started making decisions towards which sensors we were going to implement into our system. We then moved onto system testing, we had ⅗ of our most important requirements within the system section. The step by step test of waterproofing the system was something that took the most time. When it came to time, this test essentially took the entire month of March, because we had to complete the following: soldering circuits, hot gluing solder joints, printing the casings, applying nail polish to the circuits, and finally putting the circuits together. We performed an unconsolidated undrained test as a sub test of an integration test when receiving data and logging it with our DHCP server and then sending to the website to graph. We tested having no files and then updating and population the file system with CSV files with the sensor data. Once the sensor data was received we filtered through it to prepare for new update files to replace them so as to not exceed the storage capacity of the website. The software side of our project has been something we have been developing for the last two weeks, really working out the kinks and bugs.

Overall, we would say the project was a success. When we first met our client, our requirements were making a system that could give the client temperature, humidity, and if possible, light levels. What we have given her is a self-sustaining module that will essentially require no mantincence, and collects temp, humidity, light, pressure, and altitude. We have presented the collected data onto a webpage that can be viewed from a mobile device, computer, or anything that has a browser.

EE-486
Team 13
NAU Research Greenhouse
March 26, 2021

A. Introduction

Each module is essentially broken down into 2 parts: the power system and the microcontroller module. The microcontroller module is an ESP8266 D1 mini, which is wired to 2 sensors that collect data points of humidity, temperature, pressure, altitude, and light levels respectively. The DH1750 handles light levels, while the BME280 handles everything else. There will be 8 modules composed of the microcontroller and a power system, then there will be one Raspberry Pi that will act as a main hub for the modules. The ESP's are connected via WIFI to the Raspberry Pi, so that the data can be sent to it and stored as a CSV file. From there, the data is sent to a Mosquito database in order to store and save the data, as well as post it to a webpage for real time viewing. The power System aspect consists of 3 parts, a lithium ion battery, a TP4056 charge circuit, and a solar panel to continuously recharge the lithium battery.

B. System Architecture

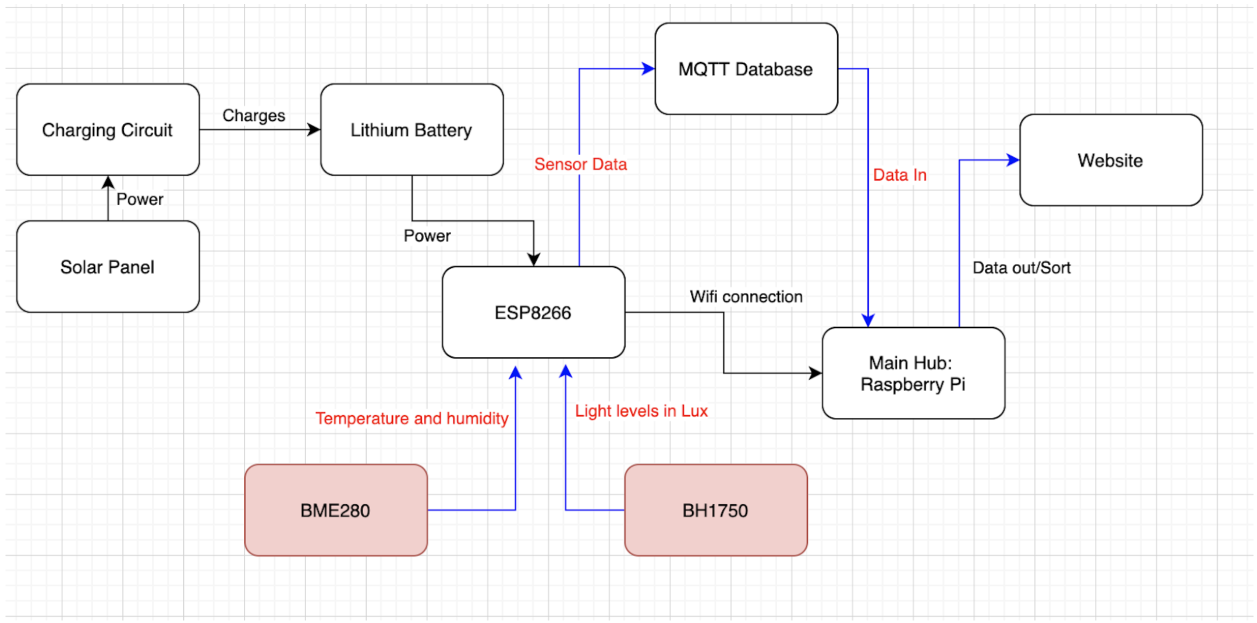


Figure 1: System Architecture

The systems architecture shows ¾ of the tests that were crucial to the project. The fourth test conducted was a Step by Step test of the waterproofed system. The 3 shown in the System Architecture are the Unit Tests of both sensors, as well as the integration tests of the modules datastream from the ESP's to the Raspberry Pi.

EE-486
Team 13
NAU Research Greenhouse
March 26, 2021

C. Requirements, Status, and Type of Test

Type of Test	Status	Req. #	Requirement
		1	**Humidity Sensor**
Inspect	🟩	1.1	Low hysteresis, ideally accurate for up to 5 years.
UTM	🟩	1.2	Humidity sensor accuracy within +-2.5%
Inspect	🟩	1.3	Low power usage when operating, 10-100 µA
		2	**Temperature Sensor**
Inspect	🟩	2.1	Operable at temperatures between 40 and 90 degrees fahrenheit.
UTM	🟩	2.2	An accuracy of +-1.5 degrees fahrenheit.
Inspect	🟩	2.3	Low power usage, when operating it should be between 10-100 µA.
		3	**Light Sensor**
Inspect	🟩	3.1	The light sensor will measure illuminance in lux.
Inspect	🟩	3.2	Low power usage, when reading light usage should be in the range of 100-200µA
UTM	🟩	3.3	Accuracy within +-2.5%
		4	**System**
Inspect	🟩 *	4.1	The system will need to be water resistant to withstand moisture and mist (IP56 rating, Ingress Protection).
UTS	🟩	4.2	The sensors will need to be able to be picked up and moved to different locations.
Integration	🟩	4.2.1	The sensors will need to be able to be picked up and moved to different locations.
Integration	🟩 *	4.3	The system will have remote access to data.
Integration	🟩	4.3.1	Able to access temperature, humidity, and light readings from the website.

Inspect	🟩	4.4	Withstand temperature between 40 degrees and 90 degrees fahrenheit.
Integration	🟩	4.5	Notification to the user if temperature is outside designated “safe” range, depending on each greenhouse the ranges are 70-85 degrees fahrenheit.
Integration	🟩	4.6	The system should operate with minimal user interaction.
Inspect	🟩 *	4.7	Each unit must be self sustaining and not require outside power.
Inspect	🟩	4.7.1	Unit must be able to charge and power itself to reduce down-time and interaction with the unit.
		5	**Data collection will ideally happen every 5-15 minutes.**
Integration	🟩 *	5.1	Data will be categorized by each greenhouse.
Integration	🟩	5.2	A daily collection of sensor data will sent and stored on the Raspberry Pi
Integration	🟩	5.2.1	If a USB is inserted, the Raspberry Pi will save the readings onto the storage device
Integration	🟩	5.2.2	Data readings will be stored in a CSV file on a Raspberry Pi
Integration	🟩 *	5.3	Must collect data at least every hour.
Integration	🟩	5.3.1	Data collection will ideally happen every 5-15 minutes.
Integration	🟩	5.4	Must be capable of sending the data to a website for mobile use.
Integration	🟩	5.4.1	The website should be able to view sensor data from all modules within the 8 greenhouses.

Table 1: Requirements Table

D. Most Important Requirements

The requirements that are crucial to the success of the project are indicated by the Asterisk (*) in the requirements chart. The requirements that were deemed crucial are: Waterproofing the system, having remote access to the sensor data, the unit's ability to sustain itself, categorization of the data, as well as multiple data readings every hour.
In all, the important parts of our requirements are meant to keep the system running while being able to withstand the harsh electrical environment that is the Research Greenhouse.

E. Types of Tests

The three types of tests we conducted on the requirements of our project were the integration tests, unit step by step tests, as well as inspection tests. The two tests that were mainly used were integration and inspection, mainly to ensure the individual systems behaved as we wanted them to and to ensure that the system will have as little maintenance as possible for the client.

F. Major Tests

The most important tests conducted was the Integration test of the modules into the Research Greenhouse. This test was conducted over the course of April, ensuring that the data stream was able to transfer data from the sensors, to the ESP, then sent over wifi to the Raspberry Pi to eventually be posted to our website for real time viewing. The unit test of the system to ensure that a high humidity and possible chance of getting wet would not ruin the system in any way, we covered solder joints in hot glue, while also coating the sensors and microcontrollers in nail polish to ensure the system could achieve an IP56 waterproof rating. Lastly, the unit tests on both the BME280 as well as the BH1750 ensured the accuracy of both of our sensors

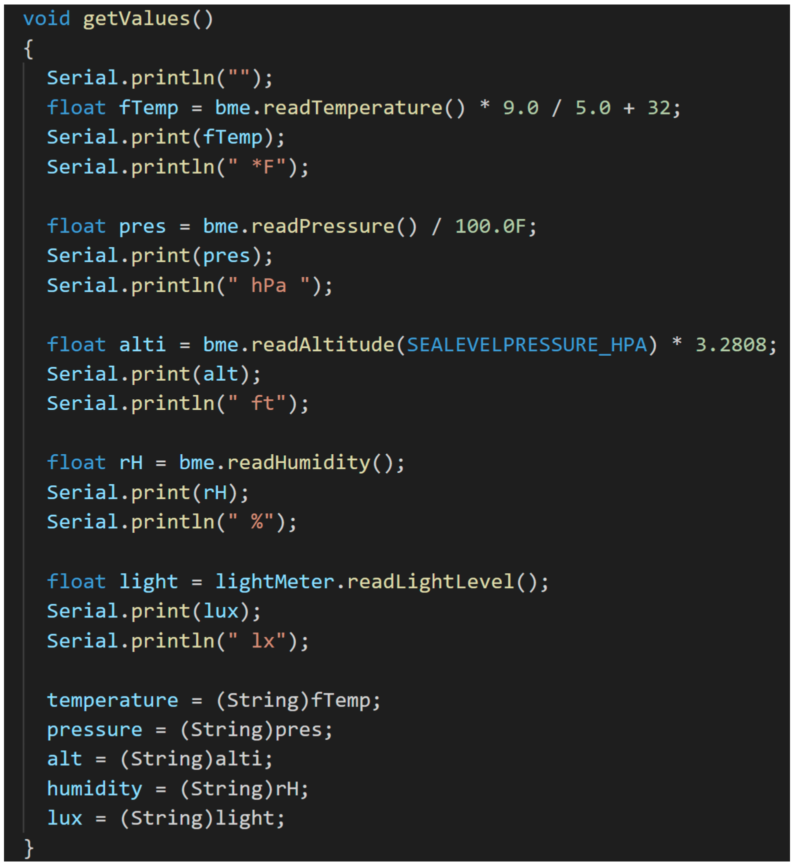
G. Analysis of Results

The 4 major tests we performed in the project to assure that the project have met our expectations regarding the waterproofing, power to the circuit, sensor operations and data streaming. The water proofing was accomplished by coats of nail polish and an abs/acetone mixture to prevent unwanted water short circuiting. Testing the water proofing involved drenching the whole project in mist and having it post back and continue to communicate. The sensors were tested by putting them into an Arduino R3 and running an example sketch to get sensor readings and testing their accuracy with higher end measurement tools. The power circuit was tested by charging the battery using the charge controller then powering a circuit for an extended amount of time. The final test was establishing a connection from the esp's to the Raspberry Pi as an access point and then the connection of the Raspberry Pi to the website graphing page.

H. Lessons Learned

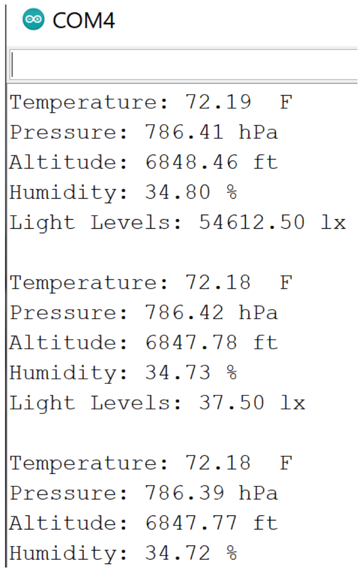
Overall, the team did not face that many difficulties throughout the project. Starting off after the first few weeks of Fall Semesters capstone class, our fourth partner dropped the class and we became a three man team. In the fall, we did not think much of this until it came time to write the Team Design Document, we realized that not having some of our missing partners documents, we had to fill in a few extra pages that we normally would not have. When it came time to the Spring iteration of capstone, the team never felt as if we were not going to complete the project because we divided up the work that the 4th partner would have done and eventually completed his section of the project without even thinking about it. Besides this, the another lesson we have learned throughout this project is how uncooperative school WIFI is, with the ESP's not even being able to connect to the WIFI thats the school hosts, so we had to make a work around to be able to get all of our modules connected so we can have a data stream from the sensors to the website that will host the data to display. Lastly, the problem that we encountered that delayed our entire project by 2-3 weeks was ITS flagging one of the microcontrollers that we ordered because it had the ability to connect to the schools WIFI. We had not had any problems with our Purchase Sheets thus far, it came to us as a shock when in the beginning of march for our final order, it was delayed for 2 weeks because it had to be approved by NAU ITS. Now, after completing the project, we know that anything that can go wrong will go wrong, so we should start things as early as possible.

I. Appendix



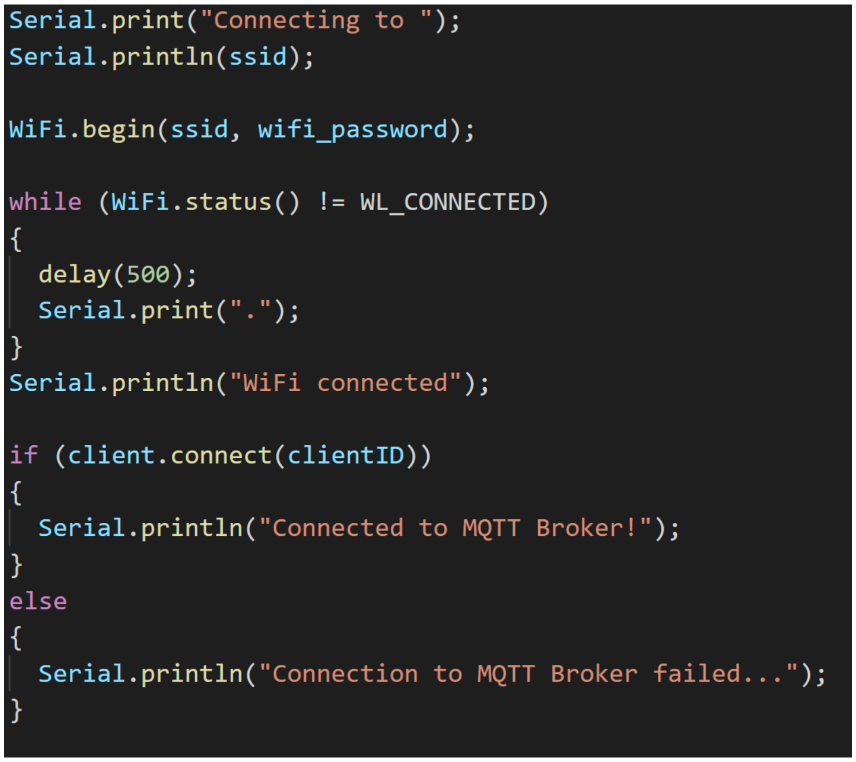
```
void getValues()
{
  Serial.println("");
  float fTemp = bme.readTemperature() * 9.0 / 5.0 + 32;
  Serial.print(fTemp);
  Serial.println(" *F");

  float pres = bme.readPressure() / 100.0F;
  Serial.print(pres);
  Serial.println(" hPa ");

  float alti = bme.readAltitude(SEALEVELPRESSURE_HPA) * 3.2808;
  Serial.print(alt);
  Serial.println(" ft");

  float rH = bme.readHumidity();
  Serial.print(rH);
  Serial.println(" %");

  float light = lightMeter.readLightLevel();
  Serial.print(lux);
  Serial.println(" lx");

  temperature = (String)fTemp;
  pressure = (String)pres;
  alt = (String)alti;
  humidity = (String)rH;
  lux = (String)light;
}
```

Figure 2: Unit Test - Arduino Code for Sensors



```
COM4

Temperature: 72.19  F
Pressure: 786.41 hPa
Altitude: 6848.46 ft
Humidity: 34.80 %
Light Levels: 54612.50 lx

Temperature: 72.18  F
Pressure: 786.42 hPa
Altitude: 6847.78 ft
Humidity: 34.73 %
Light Levels: 37.50 lx

Temperature: 72.18  F
Pressure: 786.39 hPa
Altitude: 6847.77 ft
Humidity: 34.72 %
```

Figure 3: Unit Test - Sensor Testing



```
Serial.print("Connecting to ");
Serial.println(ssid);

WiFi.begin(ssid, wifi_password);

while (WiFi.status() != WL_CONNECTED)
{
  delay(500);
  Serial.print(".");
}
Serial.println("WiFi connected");

if (client.connect(clientID))
{
  Serial.println("Connected to MQTT Broker!");
}
else
{
  Serial.println("Connection to MQTT Broker failed...");
}
```

Figure 4: Inspection Test - Initializing Wifi and MQTT Server

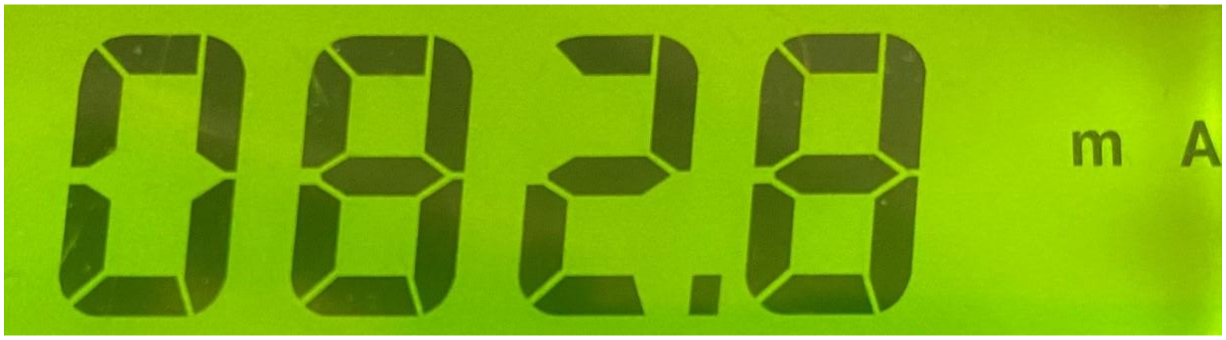


Figure 5: Inspection Test - Current Usage WiFi Modem

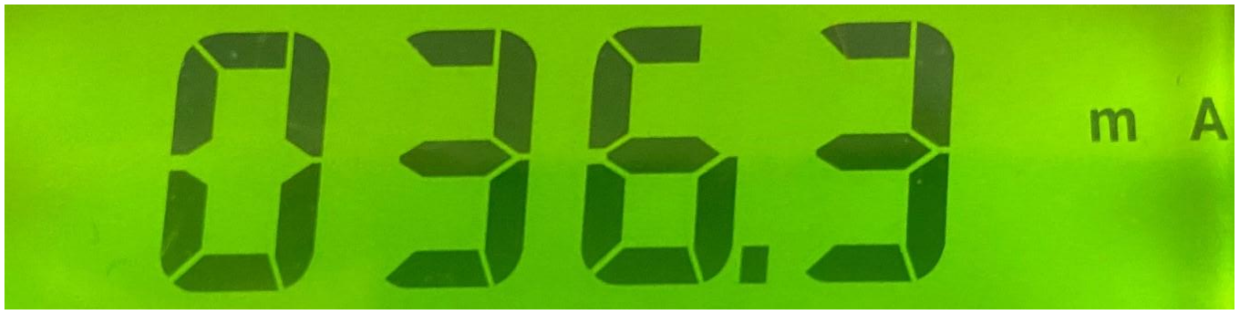


Figure 6: Inspection Test - Current Usage MQTT Data Push

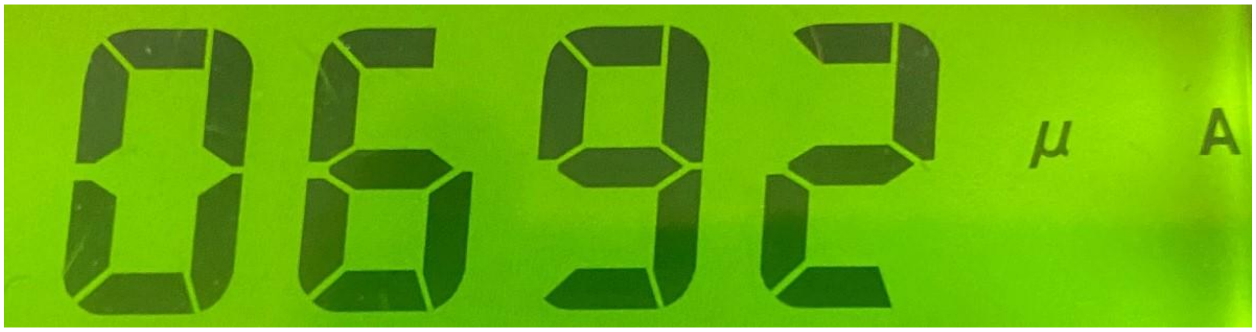
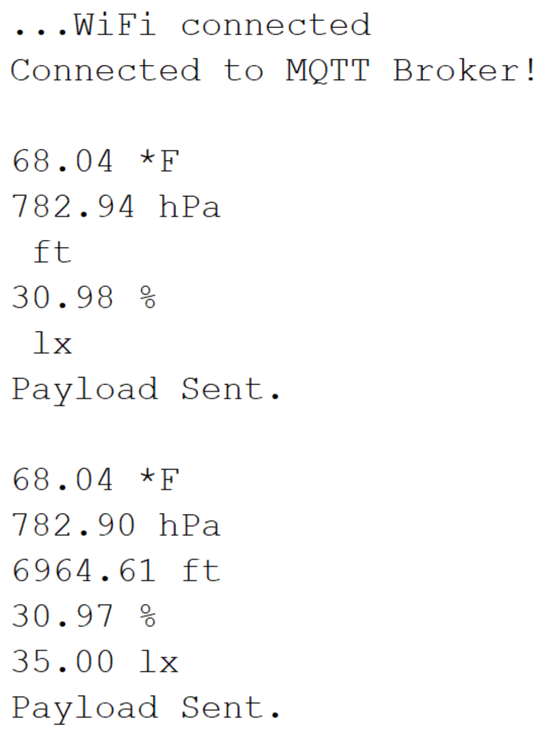


Figure 7: Inspection Test - ESP8266 Deep Sleep Current Usage



```
...WiFi connected
Connected to MQTT Broker!

68.04 *F
782.94 hPa
 ft
30.98 %
 lx
Payload Sent.

68.04 *F
782.90 hPa
6964.61 ft
30.97 %
35.00 lx
Payload Sent.
```

Figure 8: Integration Test - Data Stream Integration Test

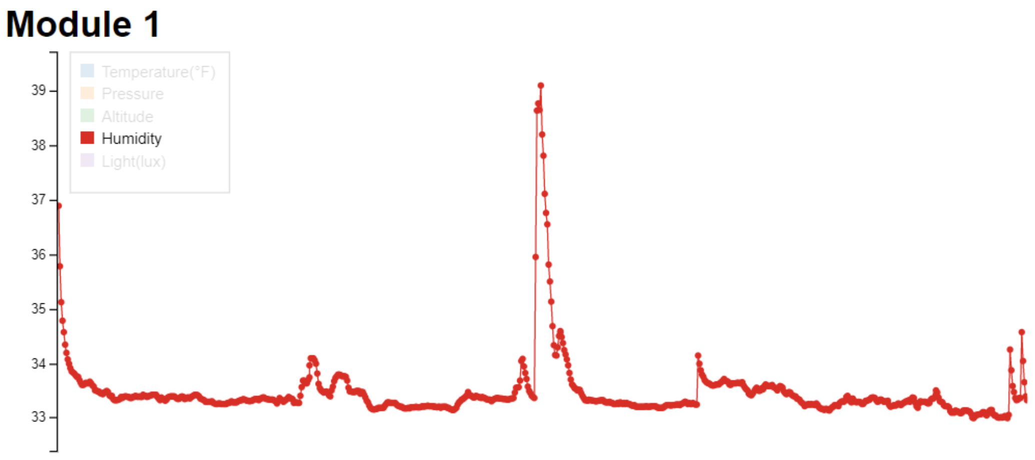


Figure 9: Integration Test - Data Stream Output on Website

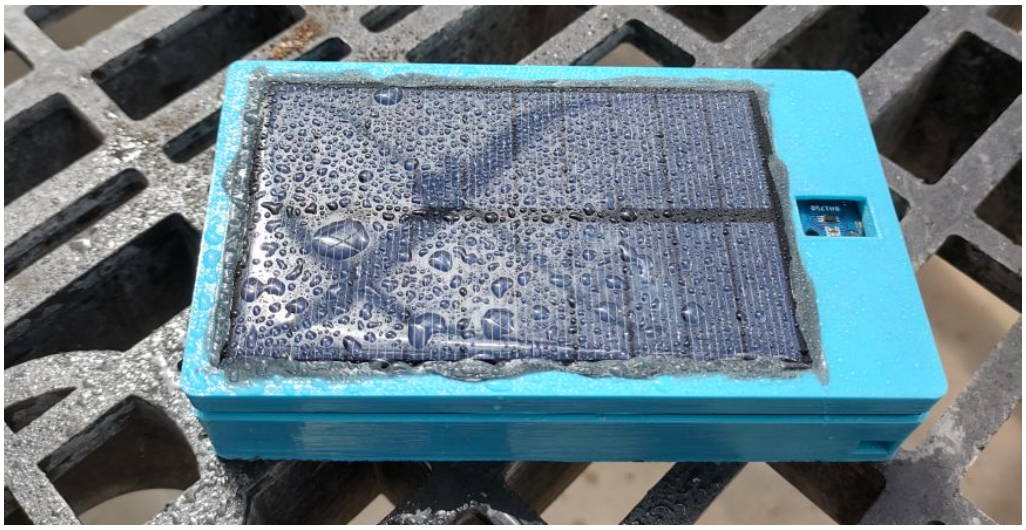


Figure 10: Unit Test - Waterproofing Testing

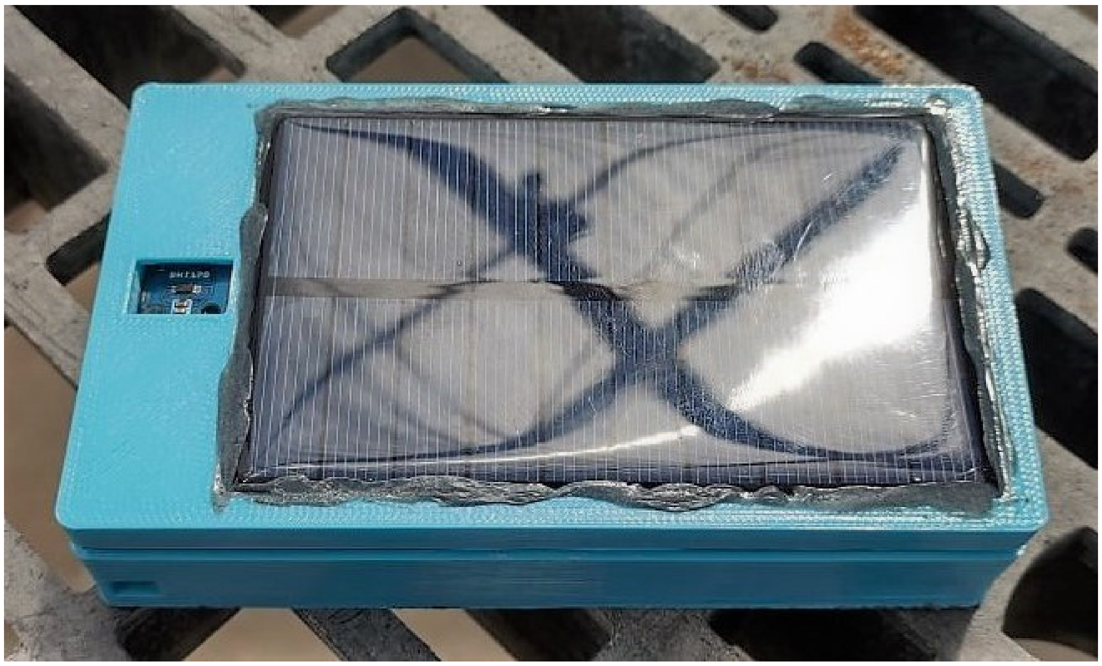


Figure 11: Unit Test - Module Days After Waterproofing Test