CPE 316 Final Project Report

Michael Albert and Quincy Collinsworth

Section 1: Project Description

This project is a low cost portable weather data logger (or weather station) for use by the Landscape Architecture program at Cal Poly SLO. The goal is to map climate patterns in microclimates so that landscape designs can cater designs towards fighting the effects of climate change and/or pollution. Our design takes Temperature, Humidity, Wind Speed, Irradiance, Time, and GPS Location, and writes incrementally to an SD card. The SD card allows the circuit to write data remotely for up to 4 years.

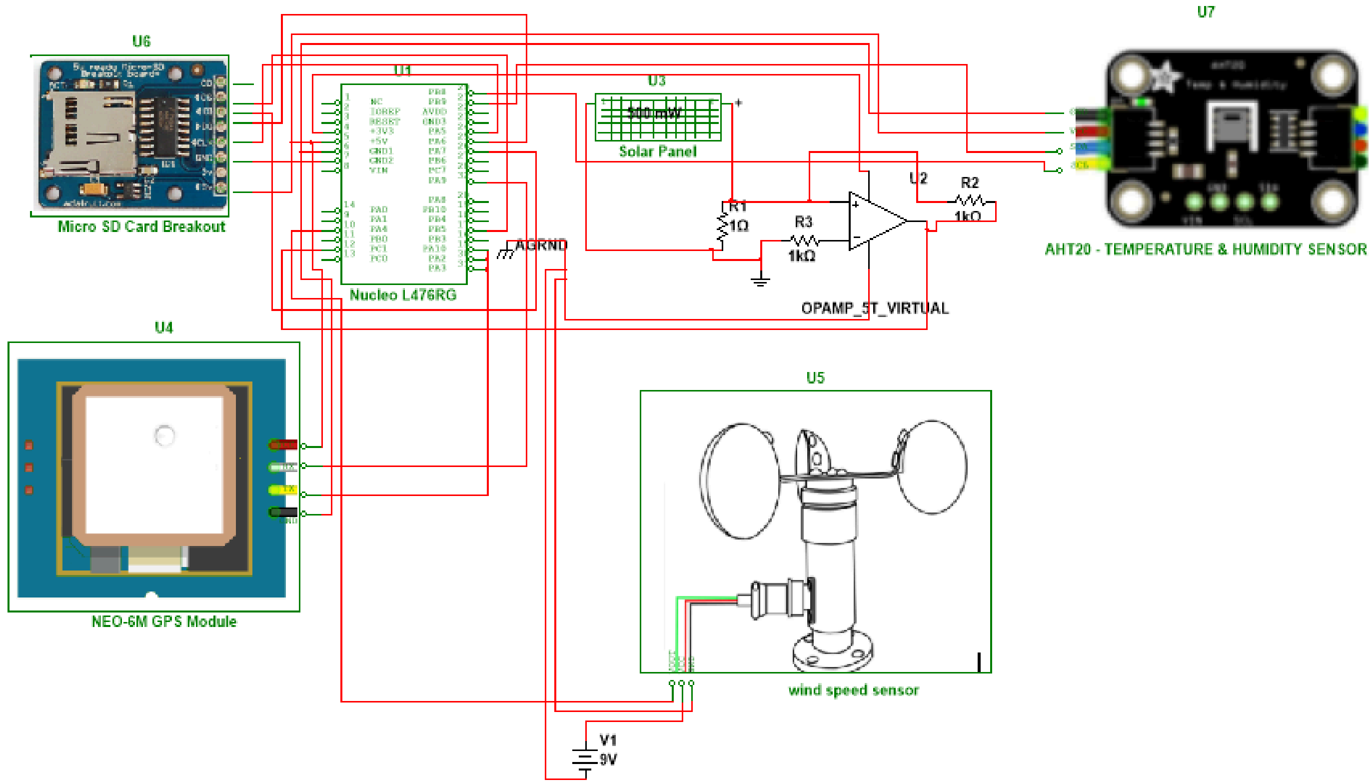
Section 2: Project Implementation

Hardware Design:

Bill of Materials:

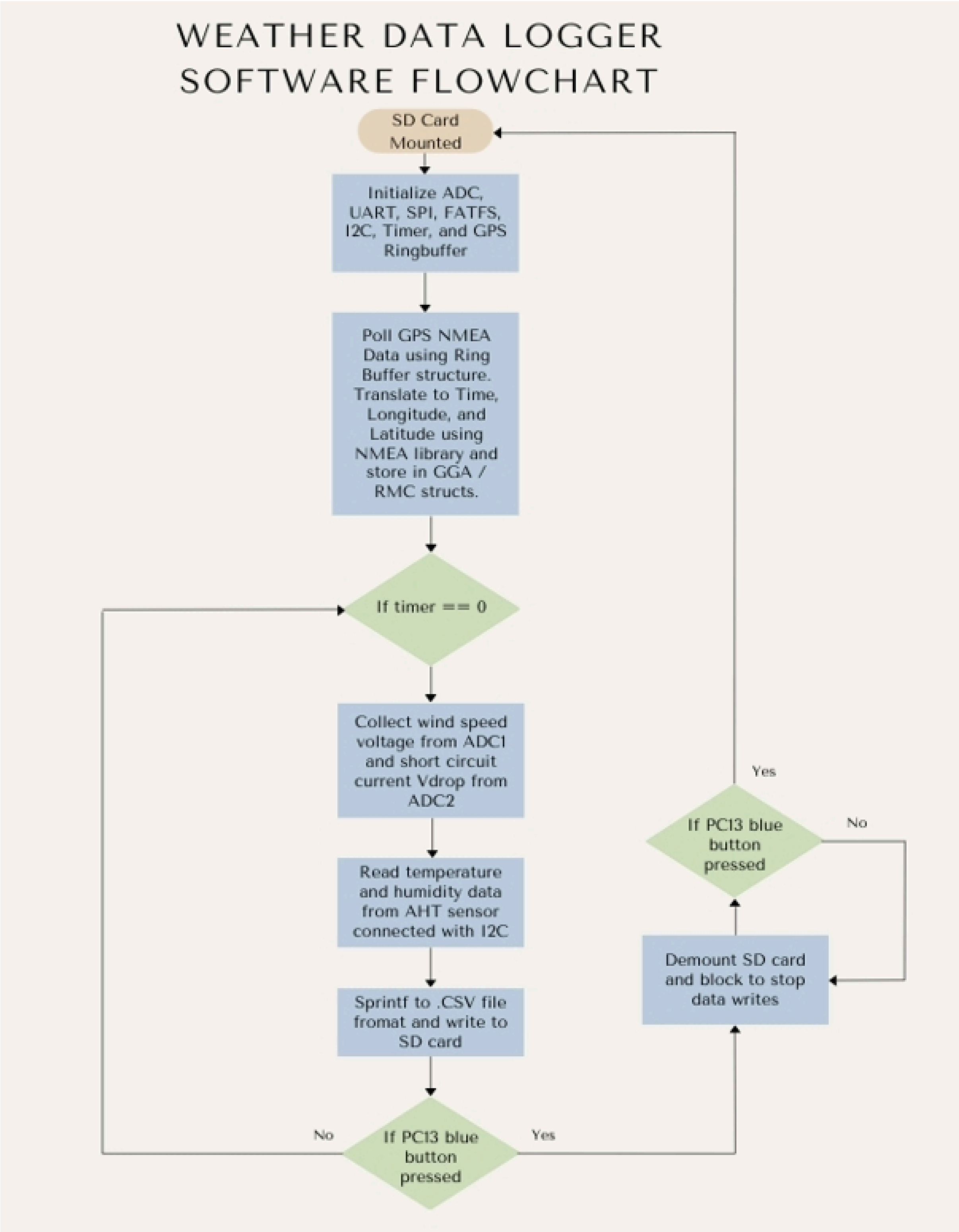
Part Name	Quantity	Unit Price	Est. Shipping Cost	Total Price	Purchasing Link
Solar Cell	1	$6.07	$6.99	$13.06	Link
Wind Speed Sensor	1	$44.95	$0.00	$44.95	Link
Humidity and Temperature Sensor	1	$4.50	$0.00	$4.50	Link
GPS	1	$12.99	$0.00	$12.99	Link
SD Card Board Attachment	1	$7.50	$0.00	$7.50	Link
SD card (memory)	1	$3.61	$0.00	$3.61	Link
Purchasing the parts together on digikey should save shipping costs			Estimated Total Parts Cost	$75.50	

Schematic Diagram:



Software Design:

Code Flowchart:

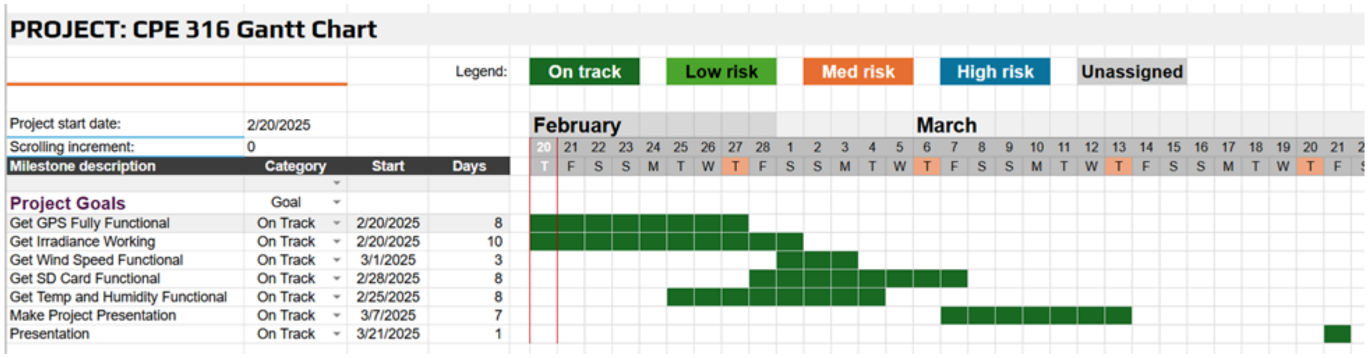

WEATHER DATA LOGGER
SOFTWARE FLOWCHART
SD Card Mounted
Initialize ADC, UART, SPI, FATFS, I2C, Timer, and GPS Ringbuffer
Poll GPS NMEA Data using Ring Buffer structure. Translate to Time, Longitude, and Latitude using NMEA library and store in GGA / RMC structs.
If timer == 0
Collect wind speed voltage from ADC1 and short circuit current Vdrop from ADC2
Read temperature and humidity data from AHT sensor connected with I2C
Sprintf to .CSV file fromat and write to SD card
If PC13 blue button pressed
No
Yes
Demount SD card and block to stop data writes
If PC13 blue button pressed
Yes
No

Section 3: Test and Demonstrate Your Project

Project Demo Video Link:
https://youtube.com/shorts/oSh-xWTJpss?feature=share

Section 4: Collaboration and Teamwork

We used a Gantt Chart to organize our work and set deadlines.



We also set aside time every week to meet together in person, mainly to debug software or hardware issues we came across in our independent work.

Section 5: Troubleshooting and Debugging

The main sources of debugging and troubleshooting arose from the GPS component and the SD card component.

For the SD, we used a driver that talks to an SD over SPI and utilizes the FatFS. This initially took a lot of effort to get working because the SD card was not receiving or responding to messages sent by the board. There were various hardware issues at first that likely would've caused an issue such as not setting up the CS for SPI to be active high and not giving the MISO and MOSI lines internal pull up resistors. After fixing that issue, the SD card was still not properly communicating with the MCU. After much debugging which included taking actual scope captures of the SPI messages (unfortunately we did not save them), we were able to determine that the SD card was not formatted for the correct FatFS. Once we set the SD card to be formatted to Fat32, it worked well. One of the most helpful tools during this process was the oscilloscope, which could be used to verify that information was being sent correctly from the chip to our STM32 board.

For the GPS, a standard buffer approach could not be used because the GPS was reading in an unknown amount of NMEA (GPS language) data depending on location and time. To work around this issue, we found a ring buffer library in github that was compatible with GPS applications. One of the main challenges with debugging the GPS was that it is only functional outside. When working inside, even if the GPS was able to connect to a satellite using radio, the GPS readings would come in as all 0's, filling the ring buffer with what looked like junk data. This made it impossible to debug unless the circuit was outside. Unfortunately, we did not have access to an oscilloscope outside, so all debugging had to be done in software, which made things more difficult.

Section 6: Lessons Learned

Through this project, we improved our ability to implement a large-scale project that utilizes multiple sensors and protocols parts. A major challenge was that many of the parts we were using had limited documentation and were designed for use with Arduino, not STM32 in C. For instance, there isn't any good official documentation detailing how to use an STM32 board to talk to an SD card over SPI. There also was no documentation on how to use the temperature and humidity sensor with an STM32 as it is a part with more architecture set up for using it with Arduino. As a result, we had to translate an existing c++ library. The same thing applies for the GPS.

Moreover, there was great value in this project because it wasn't just an interesting personal project. We have a deliverable to a customer, and there are specs that have to be met. A more personal project would have allowed us to do "whatever we want." We believe that this makes this project a very valuable experience because it is similar to what working a real job would be like.

If given the opportunity to do the same project again, we probably would've maintained cleaner code and done better version control. This might've been achieved by having a GitHub and multiple branches of code on the GitHub in order to test multiple parts of the code independently. There were a few moments where we changed a bunch of stuff, accidentally broke the project, and had to spend time just reverting all of our changes because we decided that starting over from scratch with the most recent working iteration of the project would be better. Another change would be spending more time researching parts and sensor libraries prior to ordering. In some ways, it was positive that we had to make our own sensor drivers and libraries, but we could have saved a lot of time by finding sensors that integrated better with STM32, or had generally more documentation.

Section 7: Future Possibility

This project has a lot of future possibilities already set up for it. The whole goal of this was to build a prototype weather station for the landscape architecture department. In order to get the prototype to a fully viable tool we need to work out prevailing bugs in the software. The main issue with software is that the solar panel and the wind speed sensor work properly independently, but when reading from two ADC's in one program, it stops working. This is because the ADC channels are not properly switching and reading from both sensors, causing faulty reads. A potential solution for the future is to use DMA to store values from each ADC.

Additionally, the irradiance sensor using the solar panel needs to be calibrated using a pyranometer. This is because we must know what both the short circuit current and solar irradiance is at the same time. Once we have these values, we can use the linear relationship between short circuit current of a solar panel and solar irradiance to calculate all other values in software. Lastly, this data logger has to be usable outdoors, meaning it must be packaged to be durable, waterproof, and easy to repair. The packaging must also allow the SD card to be easily removed, and for the STM's button to be pressed. We plan to build the landscape architecture department a PCB, and to 3D print an enclosure for the circuit.

Additionally, the landscape architecture has also requested that we build them a device that can measure water quality in a similar fashion, by writing to an SD card. Here is the Bill of Materials for that device, which we plan to build next quarter using an arduino.

Part Name	Quantity	Unit Price	Est. Shipping Cost	Total Price	Purchasing Link
Arduino Uno	1	$27.60	$6.99	$34.59	Link
USB type B to A	1	$7.60	$0.00	$7.60	Link
4 GB Micro SD Card	1	$8.49	$0.00	$8.49	Link
GPS	1	$12.99	$0.00	$12.99	Link
SD Card Reader	1	$6.99	$0.00	$6.99	Link
Power Bank	1	$21.84	$0.00	$21.84	Link
Waterproof Container	1	$45.49	$0.00	$45.49	Link

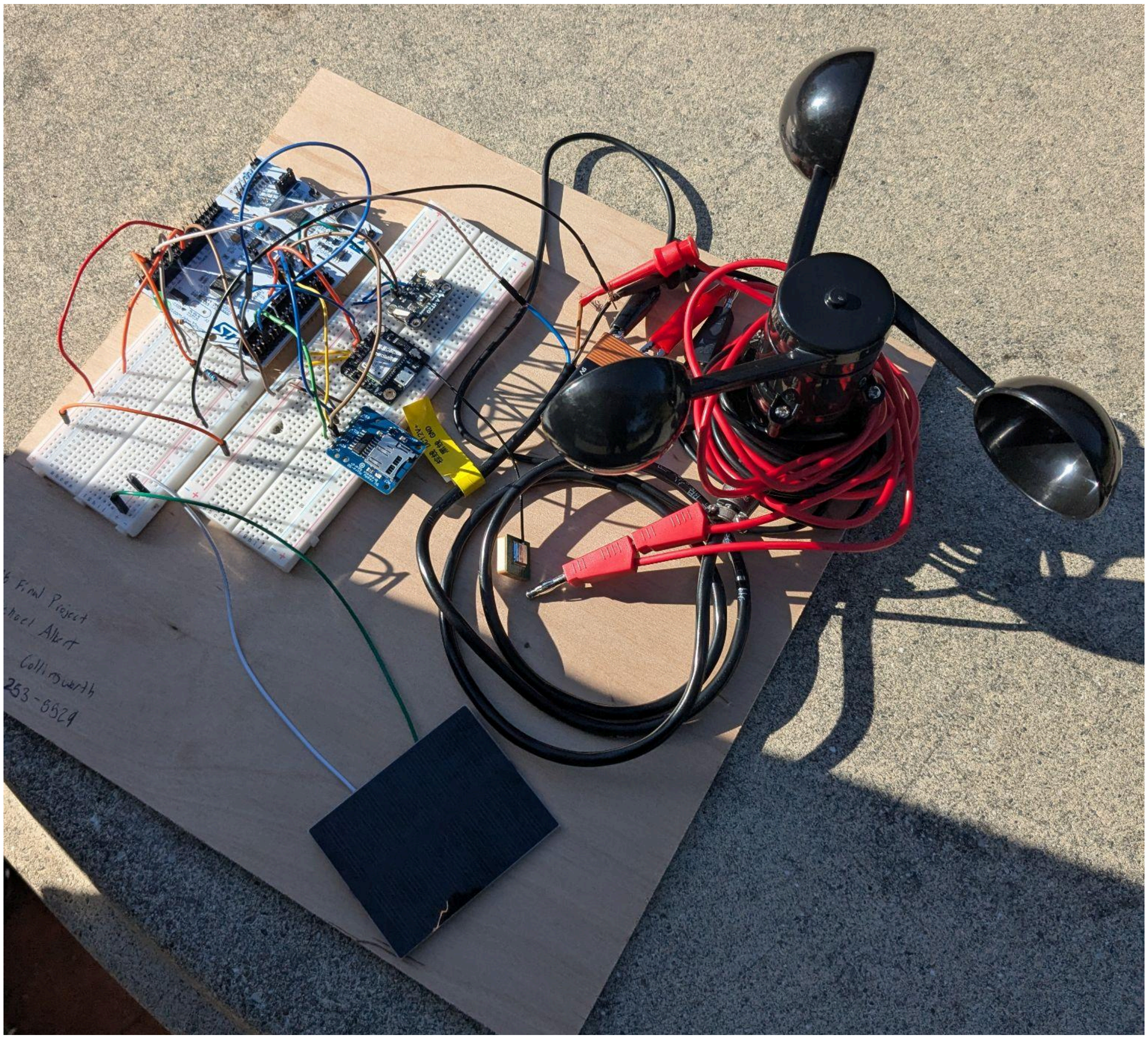
Turbidity Sensor	1	$9.90	$0.00	$9.90	https://www.dfrobot.com/product-1394.html
Analog TDS sensor	1	$11.80	$0.00	$11.80	https://www.dfrobot.com/product-1662.html
PH Electrode	1	$64.90	$0.00	$64.90	https://www.dfrobot.com/product-2069.html
Waterproof Temp Sensor	1	$7.50	$0.00	$7.50	https://www.dfrobot.com/product-1354.html
Parts we will get at a hardware store later: 1 ft of 1” PVC pipe -1” PVC cap -1-1/4-in x 1-in PVC bushing -6 ft of 16 AWG Speaker wire -Heat shrink tubing -PVC Cement -Silicone sealant -Duct tape -Flex tape -Zip ties		$40.00		$40.00	Hardware Store
Purchasing the parts together on digikey should save shipping		\	Estimated Total Parts Cost	$272.09	

costs					

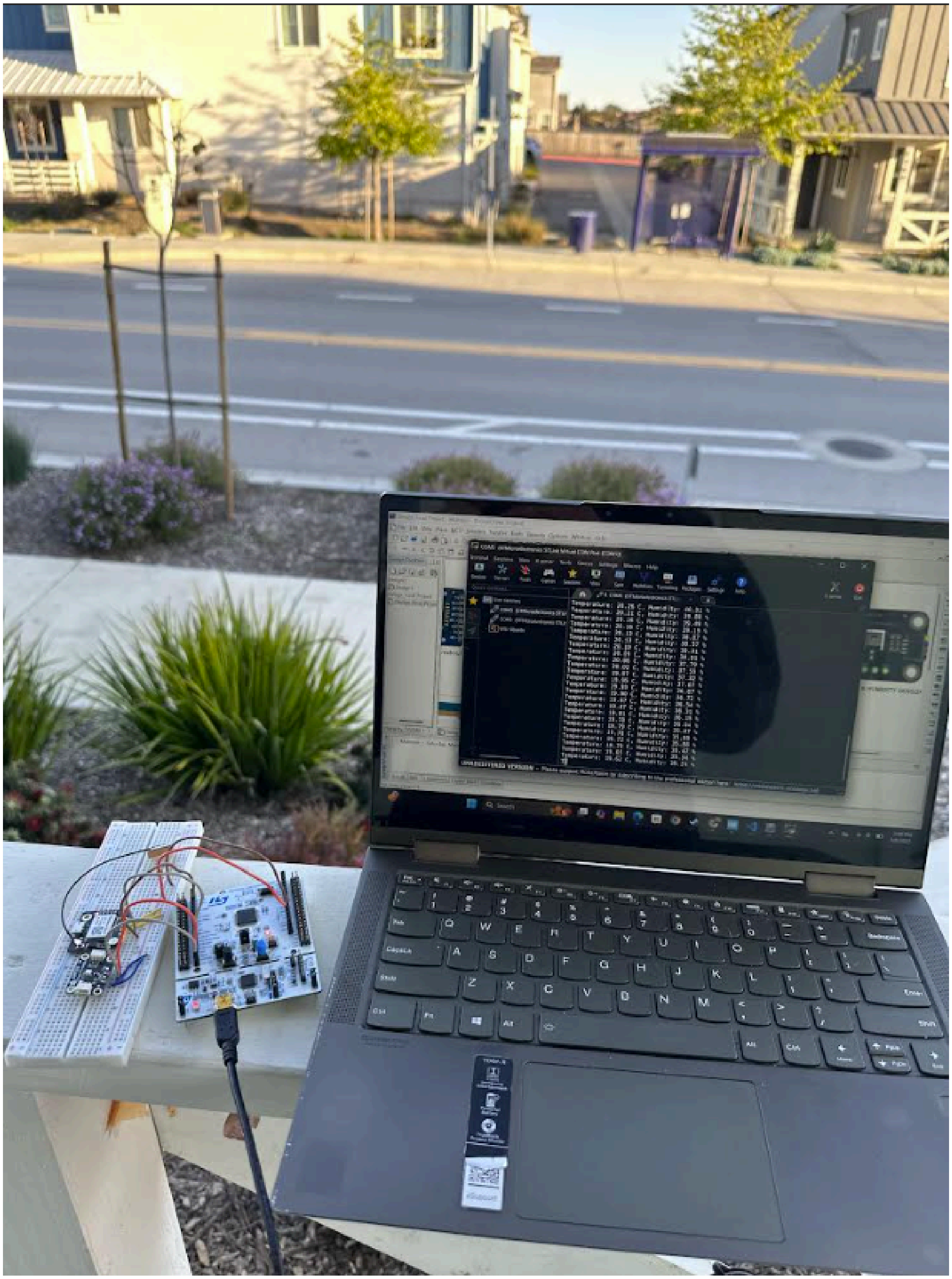
Section 8: Summary and Conclusion

We started this project in order to get some research experience and experience building a project that has requirements beyond "whatever" we feel like. Additionally, we took on the project because finishing it would feel like we contributed to important research for the landscape architecture department. We are very proud of what we accomplished this quarter and are looking forward to continuing the project going forward into next quarter. We are also excited to see how data collected using these data loggers will be used by the Landscape Architecture Department, and hope that our work has a positive impact.

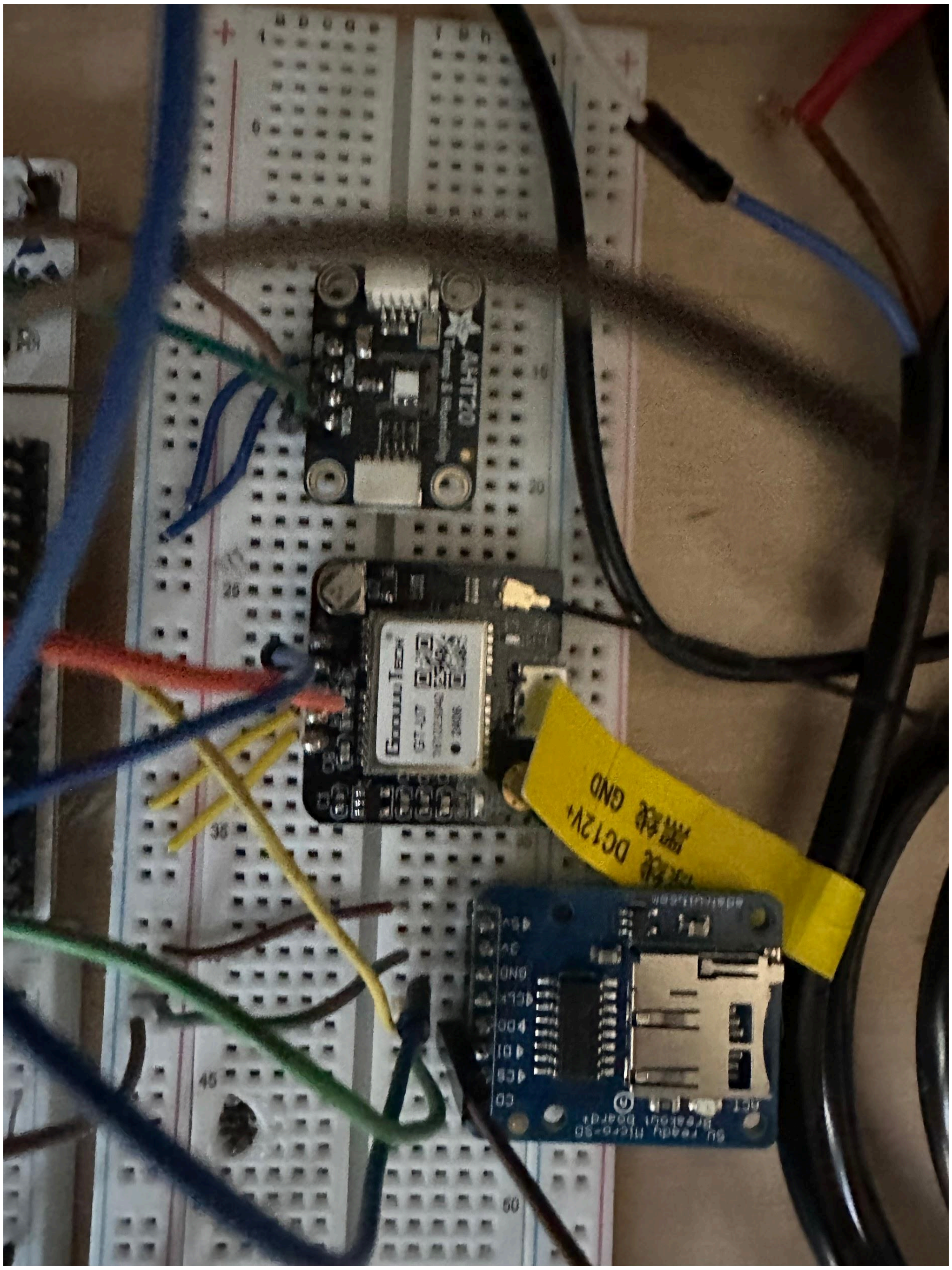
Section 9: Code and Images Appendix



A picture of the final prototype, being tested outside



Testing the temperature and humidity sensor and writing values to UART



A closer look at the SD card, GPS, and Temperature sensor

Here is a link to a google drive containing our zipped final project files:
https://drive.google.com/file/d/1Rq7gYp4kR8padHgLe-gYMVIRPskJJZuH/view?usp=sharing

Section 10: References and Bibliography

A how-to on using the SD card SPI driver:
https://01001000.xyz/2020-08-09-Tutorial-STM32CubeIDE-SD-card/

SD Card Driver GitHub Repo:
https://github.com/kiwih/cubeide-sd-card

GPS Ring Buffer Github Repo:
https://github.com/controllerstech/stm32-uart-ring-buffer

NEO-6M GPS Video Guide:
https://www.youtube.com/watch?v=tq_RoaPLahk

C++ AHT Temperature Sensor Github Repo:
https://github.com/adafruit/Adafruit_AHTX0/blob/master/Adafruit_AHTX0.cpp

Arduino Forum with some helpful insight into using a low power solar panel as an irradiance sensor:
https://forum.arduino.cc/t/using-small-solar-panel-to-measure-irradiance/550371/14