

POWR Rocket

Pressure Observation Weather Research Rocket

Project Team:

Thomas Knepshield

Tyler Roux

Dymond Deans

Sage Bonfield

Table of Contents

1. Project Definition

Tornadoes are still a relatively unknown phenomenon in meteorology. Through years of research, scientists have been able to learn how they develop, but questions are still unanswered. Why do they form? Why do some storms produce tornadoes and others don't? Since the 1970s and an extra push in the late 90s and early 2000s, research teams have developed probes to gather data from the insides of tornadoes. Barometric pressure, humidity, temperature and wind speed are all data points needed from inside a tornado. Sensors for all of these will be included in this project.

The dynamics of a tornado make the design of an object to make it into a tornado very difficult. Due to the steep pressure gradients between the area outside of a tornado and the area inside the tornado, a drone, balloon and similar devices will not be an effective way to penetrate the wind and pressure fields. A rocket, slender and moving at a high speed, will have the best chance at entering the tornado. Also, a rocket with a parachute is perfect since the parachute will act as an air parcel moving through the atmosphere. This assumption will add more validity to future research using this device.

Scientific probes in this field, currently, are expensive, hard to build and too complicated for a common person. Many more opportunities for weather research are lost due to this problem. This project aims to help bring down the cost of production of tornado sensors while making it easier for other storm chasers and scientists with minimal computer engineering experience to add to this line of research. Arduino microcontrollers and sensors along with them are very inexpensive to buy and easy to use. This project will allow anyone to build the sensor, attach it to a rocket and then launch into a tornado and receive data (the goal for this project specifically is to have this data uploaded to our own database and website for independent data collection).

This project will use accessible and inexpensive arduino microcontrollers along with sensors made to work with these devices to create a reusable tornado probe capable of transmitting data over long distances to study the upper air dynamics of a tornadic storm. The data from the

rocket-sensing device will be transmitted to a receiver made from a similar microcontroller using LoRa radio. The data received from the receiver will upload the data to a database and then posted on a website made for this project with graphs mapping the changes in temperature, humidity and pressure along with a 3D mapping of gps coordinates. Data will be stored on the rocket device in case recovery is possible, on the receiver itself in case a connection to the database is unable to occur and in the database itself. If the rocket is able to be recovered, the website will reflect this data once it is manually uploaded. Anyone will be able to access or request the data if intended use is for research.

2. Project Requirements

- Functional Requirements
 - Website
 - Display data using graphs and tables.
 - Probe
 - Collect data and transmit it to a receiver.
 - SQL Database
 - Store collected data in a table to be used to download and display on the website.
 - Upload Portal
 - Acts as a middle-ground between the weather sensor and the database.
- Usability
 - User interface
 1. Users can view and interact with the weather data collected with the probe through the webpage. Graphs will show changes in the atmosphere using data collected from the sensors.
 2. A desktop application will facilitate the connection between the rocket and the website. The user is able to upload the captured data to onboard storage and to the database.
 - Performance
 1. Data is captured, then directed to the host computer via transceivers.
 2. Host computer saves data and uploads to Azure SQL Database.
 3. Webpage retrieves data from the database.

- System Requirements
 - Hardware:
 1. 2x Arduino Nano Microcontrollers
 2. 2x Reyax RYLR896 Lora Module
 3. MPL3115A2 - I2C Barometric Pressure/Altitude/Temperature Sensor
 4. Adafruit Ultimate GPS Breakout
 5. Adafruit SHT31 Temperature and Humidity Sensor
 6. Adafruit MicroSD card Breakout
 7. Wires and other miscellaneous items
 - Software
 1. C# Windows Forms App developed using Visual Studio 2022
 2. Glitch to develop the webpage.
 3. Using infinityfree and cyberduck to upload the files to create the website.
 4. Node.js for javascript to connect to the database through the website.
 5. Microcontrollers library: Adafruit MPL3115A2, Arduino SD, Arduino TinyGPS++, Software Serial, Arduino, Wire, SPI, Adafruit SHT31
 - Database: Azure SQL Database
- Security:
 - Onboard data encryption through LoRa transmitter/receiver.

3. Project Specification – Group responsibility

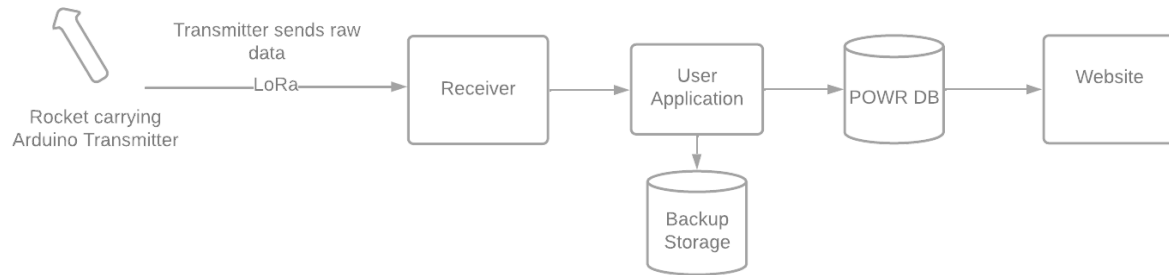
- This project focuses on a way to use aspects of computer science to help with real worth problems. Included in this project are:
 1. Website development
 2. Windows Forms Application Development
 3. Microcontroller programming and engineering
 4. Use of radio connections to transfer data
 5. Database configuration for remote connections
 6. Node.js to connect the website to the database
- Frameworks / Development Environment
 1. Arduino IDE
 2. Adafruit sensor libraries
 3. C/C++
 4. ASP .NET Windows Forms Development Visual Studio 2022

5. Microsoft Azure SQL Database
 6. Glitch and Cyberduck for JavaScript, HTML, CSS, and Node.js
- Platform
 1. Website
 2. Desktop Application
 3. Probe
 - Genre
 1. Scientific research
 2. Web application

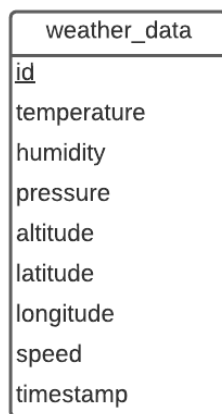
4. System – Design Perspective – *Group responsibility*

- Subsystems for this project:
 - [POWR Upload Portal](#): Desktop Application to open on data-receiving computer; will be used for uploading captured sensor data to storage and to the project database.
 - [POWR Rocket Website](#): Website used for displaying captured data in a meaningful way, using multiple types of graphs, and allowing users to select certain types of data to display.
 - [Arduino Transmitter Probe and Receiver](#): Versatile computer chips that will be able to collect data and communicate with each other over long distances (LoRa communication). Transmitting board will be equipped with weather sensor modules to capture data and a LoRa module to transmit data to the receiving board. Receiver module will listen for the data sent out by the transmitter, read and store sensor data to an on-board MicroSD card. Receiving board may also be connected to a computer to be used alongside the Upload Portal described above.

- Sub-System Communication
 - DataFlow Diagram begins with Rocket carrying Arduino transmitter:



1. Arduino sensors on board capture weather data
 2. Weather data is transmitted to the receiving Arduino via LoRa connection
 3. Receiver, connected via USB to host computer, is synced to the desktop User Application
 4. User Application is able to upload to backup storage or to the project database
 5. Database will hold timestamps for weather data. Website makes connections to the database to display data on the website in a presentable form.
- Entity Relationship Model (E-R Model)



- Developments on this model are being made.
- Overall operation - System Model
 - Seamless transfer of data from the rocket to the website is the goal. Upon launching the rocket into the atmosphere, the transmitters and receivers will work on their own, and the host computer will record the data. As of

now, it is unsure whether or not the data will be automatically or manually uploaded to the database - the project team sees the automatic option as a prominent feature for the future. The website will grab the data from the database and present the data neatly and intuitively, allowing the users to select specific details from the data.

5. System – Analysis Perspective – *Group responsibility*

- Subsystems and Analysis
 - POWR Upload Portal
 - Operates on ASP .NET Windows Forms Framework. The ability for the program to run fully-functionally is dependent on the connection to the Arduino (as well as the connection of the Arduino to the MicroSD Reader), and the connection to the database.
 - POWR Rocket Website
 - HTML and CSS for the frontend of the website. Javascript and PHP for the backend. We are considering using PHP for the connection to the database and for displaying the data, but Javascript may still be needed for now. The use of Javascript may be used with the help of React, but we are trying a new way to show the data on the website without using a command to run every time. But as of now we have connected to the database through javascript.
 - Arduino Sensors and Transceivers
 - Arduino sensors will collect data and store them onboard the device, on the receiver and on the database. Code on the probe, and receiver will be written in C. Using these devices is great due to their low cost, but the transmission speeds will be slow. To compensate, the device will store more data onboard and transmit segments of data rather than all of it. Recovery would be optimal for this, but not necessary. There will be data transmission from the receiver to the database. If there is no internet capabilities or that gets cut off for some reason, the uploading to the database will be ceased and uploaded manually once wireless capabilities are restored.
- System (Tables and Description)
 - Data analysis
 - Data Dictionary of project database

Name	Description	Data Type	Possible Values	Primary Key?
timestamp	The moment in time that data was collected	Int values converted to string values	11:27:16 03:56:29 09:53:16	Yes
latitude	Measure of the distance North-South	Float, left_digits=2, right_digits=6, min=-90.000000, max=90.000000	-87.432547 -2.555558 75.757554	No
longitude	Measure of the distance East-West	Float, left_digits=3, right_digits=6, min=-180.000000, max=180.000000	-100.585858 135.477894 97.229533	No
altitude	Measure of height above sea-level	Int	52800 356 12012	No
pressure	Barometric reading in in Hg	Float	35.65, 22.05, 1.01	No
temperature	Measure of the temperature in Fahrenheit	Float	55.23, 94.98, 61.12	No
humidity	Measure of air saturation	Float [0,1]	0.05, 0.89, 0.60	No

- Algorithm Analysis
 - Big - O Analysis of Subsystems: Assuming that the desktop app runs on a loop $O(n)$,
 - Data Probe: $O(n)$ due to a continuous loop run to collect data.
 - Receiver: $O(n)$ due to the continuous loop uploading data to the database.

6. Project Scrum Report - Group Responsibility

- Product Backlog:

Product Backlog

April 28, 2022

User story	Priority
(Website) As a site visitor, I want to know how I can download the data on my own computer for research.	1
(Website) As a site visitor, I have the choice to view both the tables and the graphs simultaneously.	2
(Website) As a site visitor, I should have the choice to view a certain amount of data points from the database.	3
(Portal) As a user, I want the ability to capture and view my data synchronously with the receiver.	4
(Website) As a site visitor, I want to be able to see past data captures.	5

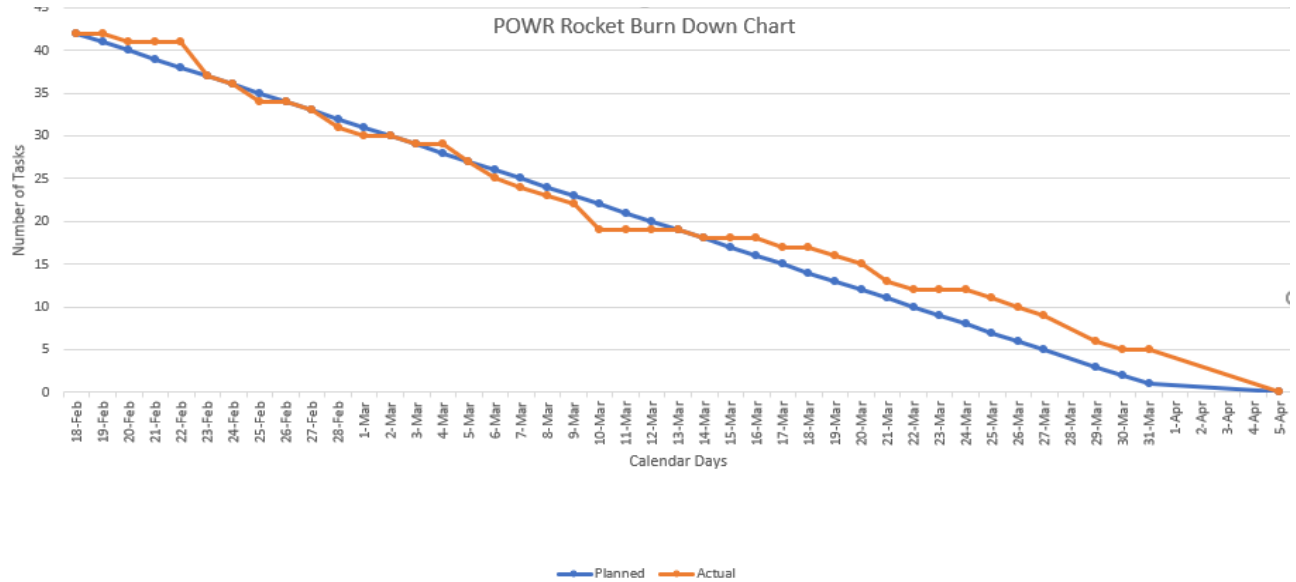
- Sprint Backlog:

Sprint Backlog

April 28, 2022

Sprint Item	Priority
Successfully operate the upload portal alongside arduino boards and Azure database	1
Configure receiver to work with upload portal	2
Ensure that upload portal works properly with receiver	3
Add user-input table fields for transaction into database (portal)	4
Engineer a functioning rocket prototype	5
Test in a real storm	6
Make website compatible with mobile/tablet browsing	7
Create logins for users on the website	8

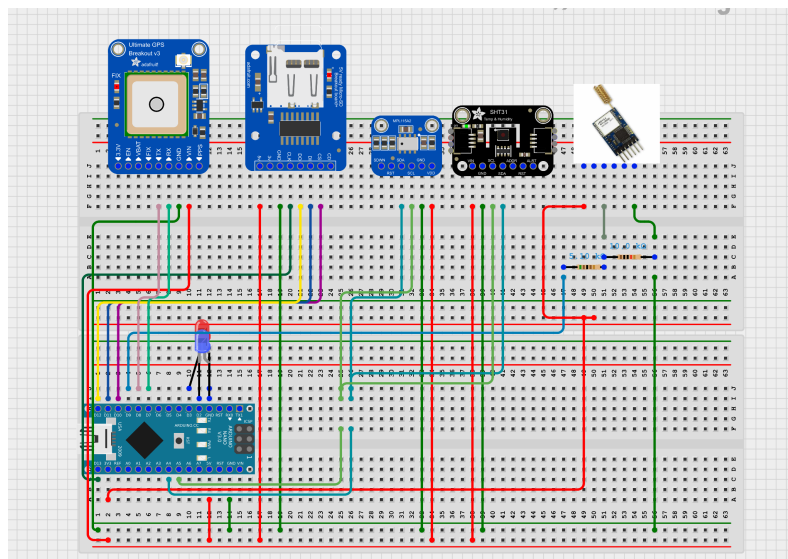
- Burndown Chart:



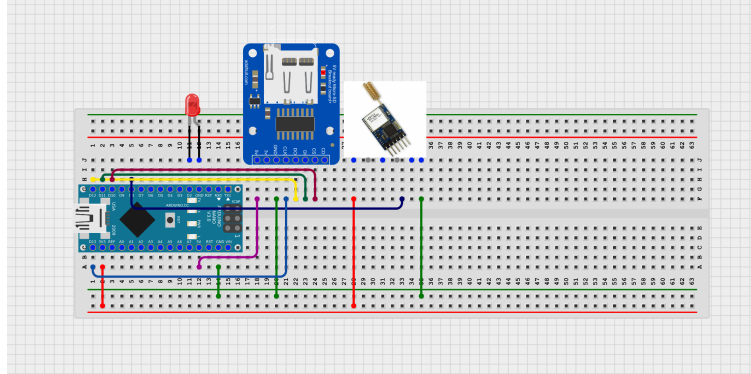
7. Subsystems

7.1 Hardware System – Thomas Kneppshield

- Transmitter/Data Collector
 - In this part, the sensors and main microcontroller will be placed in a capsule of a rocket for data collection. After data is collected, the information will be saved to an onboard SD card and transmitted to the receiver. A diagram of the wiring is done below.



- Receiver
 - The receiver collects the data and saves it to an onboard SD card. This will also have functions with the exporter created in another subsystem.
- Data dictionary



- GPS module: Latitude, Longitude, altitude, speed and time
 - Will be transmitted to the receiver as a String
 - Latitude and longitude will be six trailing zeros until a fix is made. Units: degrees
 - Speed and altitude will be 0 until a fix is made.
 - Speed units: kilometers per hour
 - Altitude units: meters
 - Timestamp will be 00:00:00 until a fix is made.
- SHT31 module: Humidity
 - Will be transmitted to the receiver as a String
 - Humidity units: percent relative humidity
- MPL3115A2 module: barometric pressure, temperature
 - Will be transmitted to the receiver as a String
 - Temperature units: degrees celsius
 - Barometric Pressure units: kilopascals
- Refinements
 - Code
 - Initial code framework called for the data to be saved to the SD card and every 10th transmission was sent to the receiver. Instead, every data point will be transmitted to the receiver until connection is lost.
 - Initially the hardware serial ports were going to be used to transmit and receive data, but they did not function correctly when the Arduino Nano boards were switched out with the Arduino Nano Every boards. Software Serial had to be used which consequently

lowered the Reyax Modules' baud rate to 38,400 instead of 115,200. This will not affect usage or the planned transmission rate.

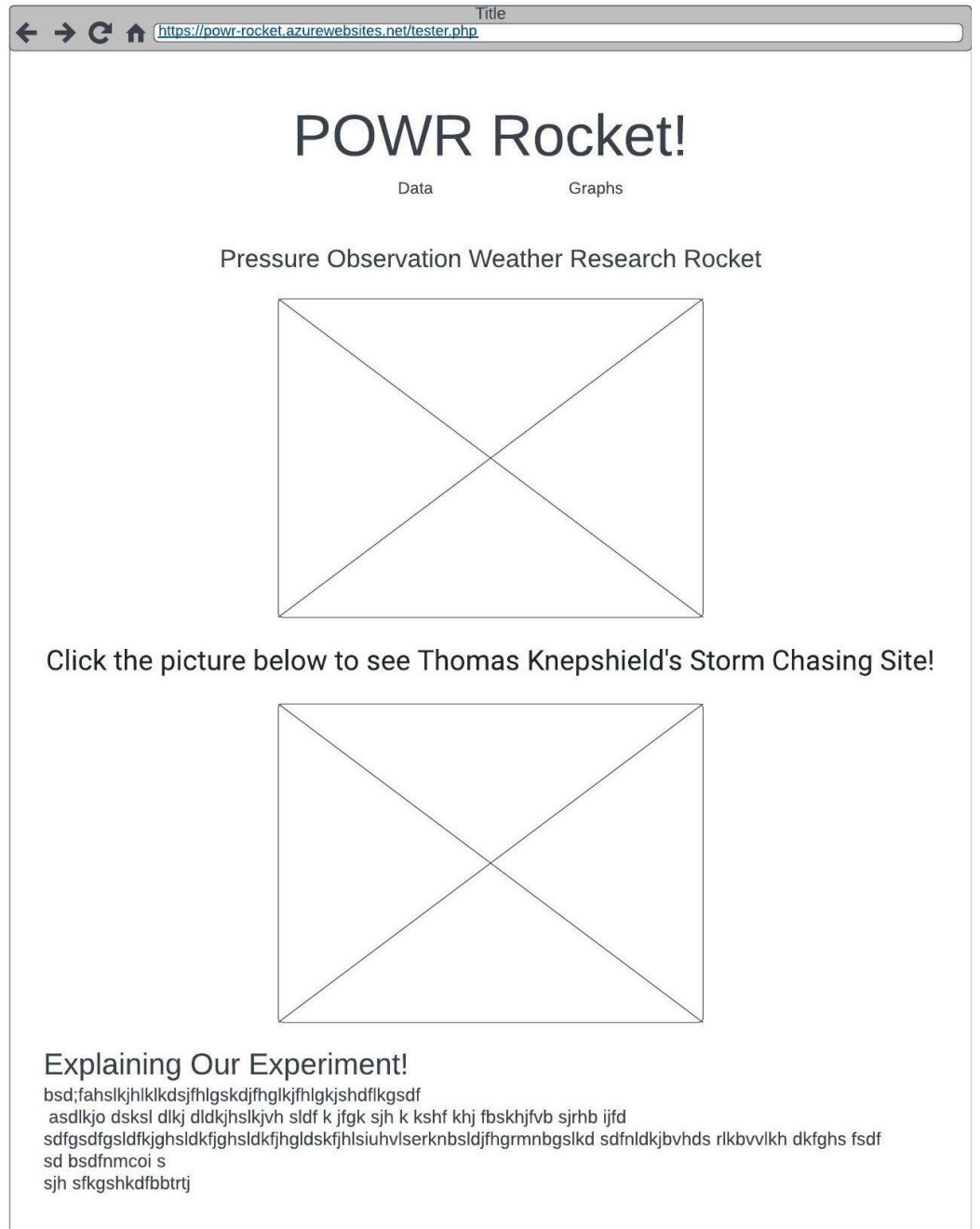
- Initially the code was going to continuously save data to the SD card and then check if there is a LoRa connection, but that is not necessary due to the nature of a transmission. A transmission only uses a print statement and can be done regardless of connection.
- There will be no cap to the amount of data points collected. It will cease when the battery is no longer able to support all of the sensors.
- Hardware
 - Initially the project used Arduino Nano boards, but those were not powerful enough to handle the code for this project. Over 100% of the flash memory was used up. The new microcontrollers have nearly double the onboard memory and flash memory, which is perfect for this project.
 - Too much power was reaching the Reyax LoRa module and led it to not work. The need of a 10k ohm and 5k ohm was used to bring the voltage on the module.
- Rocket
 - Initially a standard rocket was going to be used, but the discovery of a payload style rocket changed the idea of how it will be designed. The rocket will split in 2 after the engine is finished burning leaving the sensor package behind with a parachute.
- Refined Design
 - The above wire diagrams show how simple the wiring is. Each sensor needs power and ground and are connected to the power rails. After that each sensor is connected to the board in the correct pins.
- Scrum Backlog
 - Diagram in Section 6
- Coding
 - Each of the sensor's code was combined to collect the data after wiring was completed. Upon boot the code checks each sensor and module to verify they are wired and functioning correctly. After this the data from each is collected, transmitted to the receiver and then saved to the SD card onboard.
 - C/C++
- User training
 - Training / User manual will be on the arduino project hub for the transmitter and receiver.

- Testing
 - Once rocket is assembled, real testing will commence
 - Currently the breadboard prototype sends and receives data from each other, but testing has not been done over a distance due to the recent milestone.

7.2 Website – Dymond Deans and Tyler Roux

- Initial design and model
 - Design choices:
 - Website contains a homepage to allow the user to navigate through different links containing different ways to display the data.
 - One link being the data in list form will contain multiple buttons to allow the user to choose which parts of the data to view with filters. There will also be an option for the user to pick how many data values they will be able to view at a time.
 - The other link will contain different graphs that contain different parameters using filters that can be selected through buttons.

- Wireframe for website



•

•

POWR Data

Home

Graphs

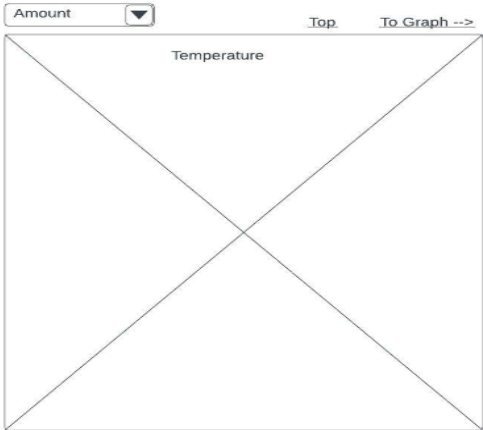
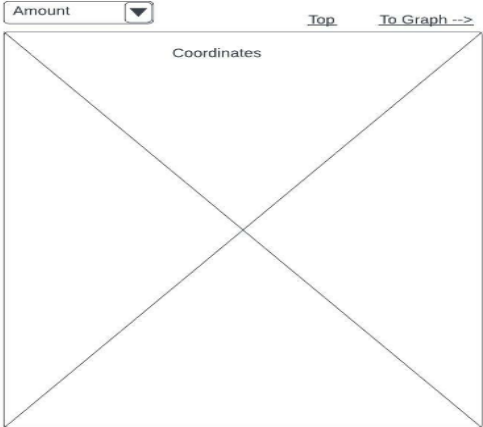
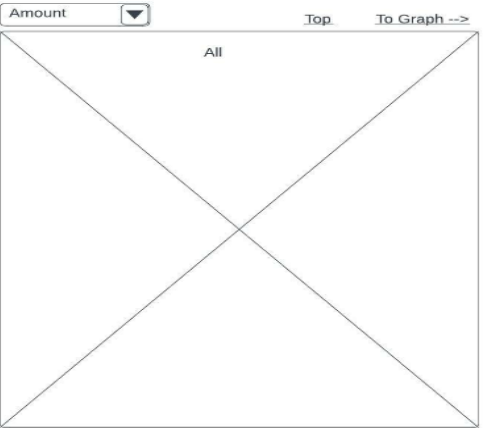
Show all

Coordinates

Temperature

Pressure

Humidity



...

...

POWR Graphs

Home

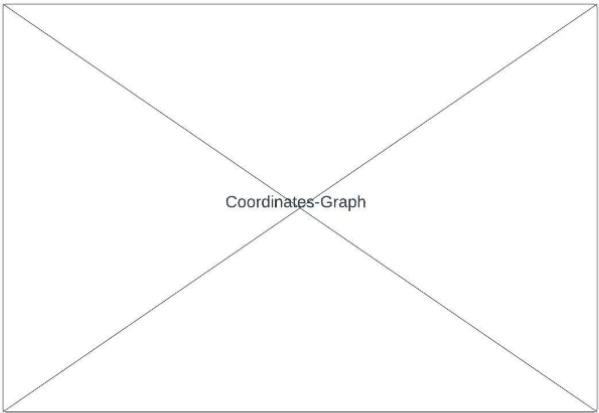
Data

Coordinates

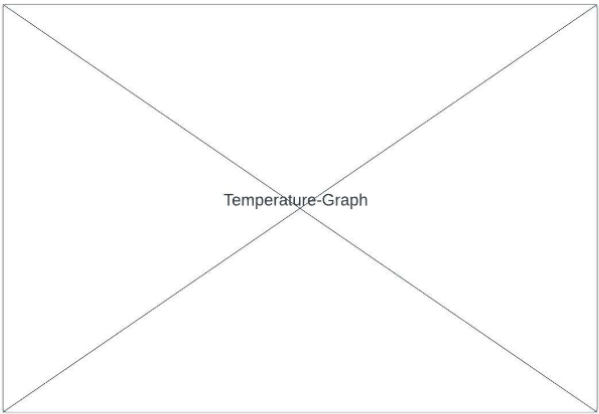
Temperature

Pressure

Humidity



[Top](#) [To Data -->](#)



[Top](#) [To Data -->](#)

...

...

- Data dictionary:
 - JSCharts is used to display the graphs.
 - Data tables are pulled from the database, converted into a csv, then displayed using javascript.
- If refined (changed over the course of project)
 - Reason for refinement (Pro versus Con):
 - Made it easier for the user to navigate through the webpage as before everything was on one page making it very cluttered.
 - Changes from initial model:
 - The data tables and graphs are now on their own separate pages.
 - There are buttons now that allow the user to pick which type of data they want to see, whether it be in list form or displayed on graphs. Different views are also shown for the data making it viewable with all of the data at once, or just the latitude, longitude, and altitude only, etc.
 - Refined model analysis
 - Refined design (Diagram and Description)
- Scrum Backlog:
 - Refer to Section 6
- Coding
 - Approach (Functional, OOP):
 - Object oriented.
 - Server side.
 - Client side.
 - Language
 - Javascript
 - PHP
 - HTML
 - CSS
- User training

User Manual for the [website](#) :

At the homepage of the website one is able to view links to the other webpages, as well as a brief explanation of the experiment, and a video explaining the prototype of the transmitter and receiver. By clicking on the picture of the tornado you will be redirected to [Storm Chasing Mysite](#), a website designed by Thomas Knepshield.

Clicking on the [Data Tables](#) header will lead to a page for viewing multiple tables with data provided by the project's database. Click on the

buttons below to view individual properties or all of the properties in one table.

Clicking on the Graphs header will send you to the page for viewing the data visually. Here you will be able to see the data individually by type or all of the data together. The data for viewing the coordinates (altitude, latitude and longitude) will display a Plotly graph. At the moment, other data points will be viewable as a line chart. Hovering your mouse over data points allows you to see that data point's precise data.

- Testing:
 - Using Visual Studio Code for writing and filezilla for updating and testing the code.

7.3 Upload Portal – Sage Bonfield

- Initial design and model

Wireframe / Illustrated Demo:

■ Design 1 (original design)

Design 1

POWR Upload Portal

[Add a new Arduino](#)

Nickname:
BN :
VID :
PID :
SN :

Submit

Arduino_1

Arduino_2

Arduino_3

ID: XX-XXXXXXX
Last Upload to DB: 2/8/2022 21:01:10
Last Upload to Storage: 2/7/2022 09:34:59

Upload to DB


Upload to Storage

■ Design 2 (refined design)

Design 2

POWR Upload Portal

Serial Connection



Connection status will appear in a label here

Arduino Info

AT:

Last Upload to Storage:

Last Upload to Database:

This button will be hidden unless a database connection has been established

Database Connection Status

Server Name:

DB Name:

Username:



Password:

A label will display telling if the connection was established

■ Final design

POWR Upload Portal

Serial Connection

Not connected

ID:

Last storage upload:

Last database upload:

C:\Users\sageb\OneDrive\Desktop\School Folder

No upload

Database Connection

Server Name:

DB Name:

Username:

Password:

Table:

Not connected

- The content is broken down into 3 distinct sections: Serial Connection, Arduino Info, and Database Connection. For the upload portal to operate correctly, a secure serial connection must be established with the Arduino to allow for uploading to database and storage, and a database connection must be made to allow for uploading to database only. This is to allow the user to upload to safe storage without an internet connection, since the user's computer will likely be without it when in an outdoor environment.
- Reason for refinement
 - Pros:
 - Allows user to enter custom information about database
 - Allows user to select correct COM port connection
 - Cons:
 - Cannot handle multiple arduino connections
- Refer to section 6 for Product Backlog
- Coding
 - Object-Oriented Approach
 - Language: C#
- User training
 - Training / User manual
 - [Upload portal manual](#)

8. Complete System

- Final software/hardware product
- Source code and user manual – screenshots as needed - Technical report
 - [POWR Rocket Docs](#) includes progress report and upload portal user manual
 - [POWR Rocket Source](#)
 - [Tornado Research Probe Rocket - Arduino Project Hub](#)