Evaluate us!

Drone Tracker

Capstone Design Final Presentation

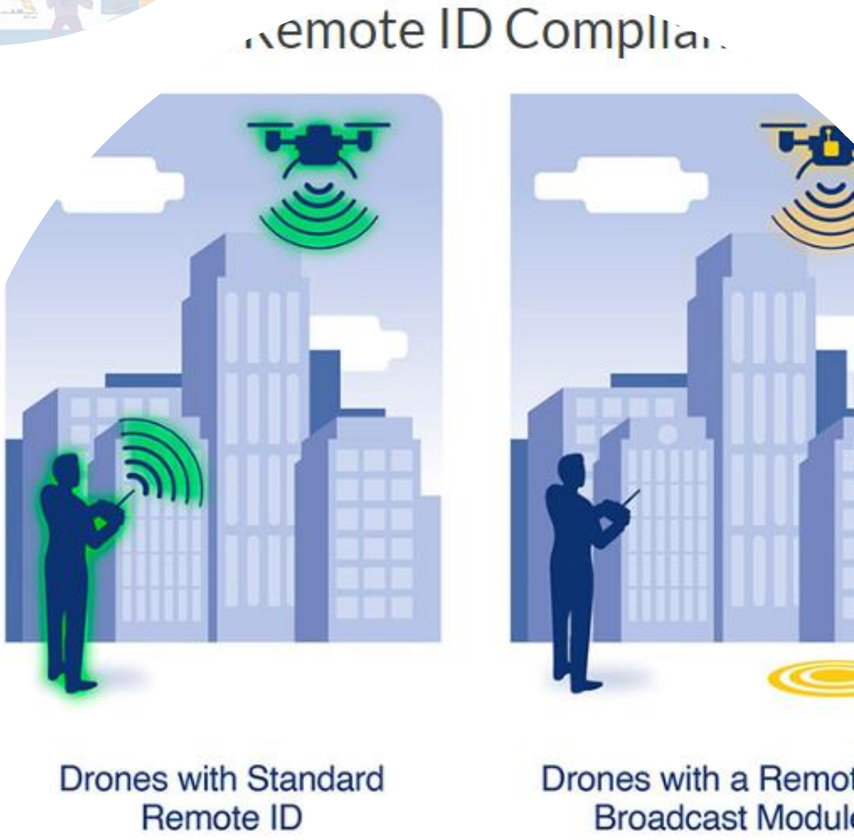
Meredith Nye
Aaron Stewart
Abe Perkins
Sterling Sloan
JieJun Stowell
Austin Williams

Project Origins

- Drone Tracking Project
  - Did not like the initial ideas offered and sought out other options.
  - Dr. Austen (secondary stakeholder and advisor) offered the drone tracking project idea utilizing a drone's remote ID signal.

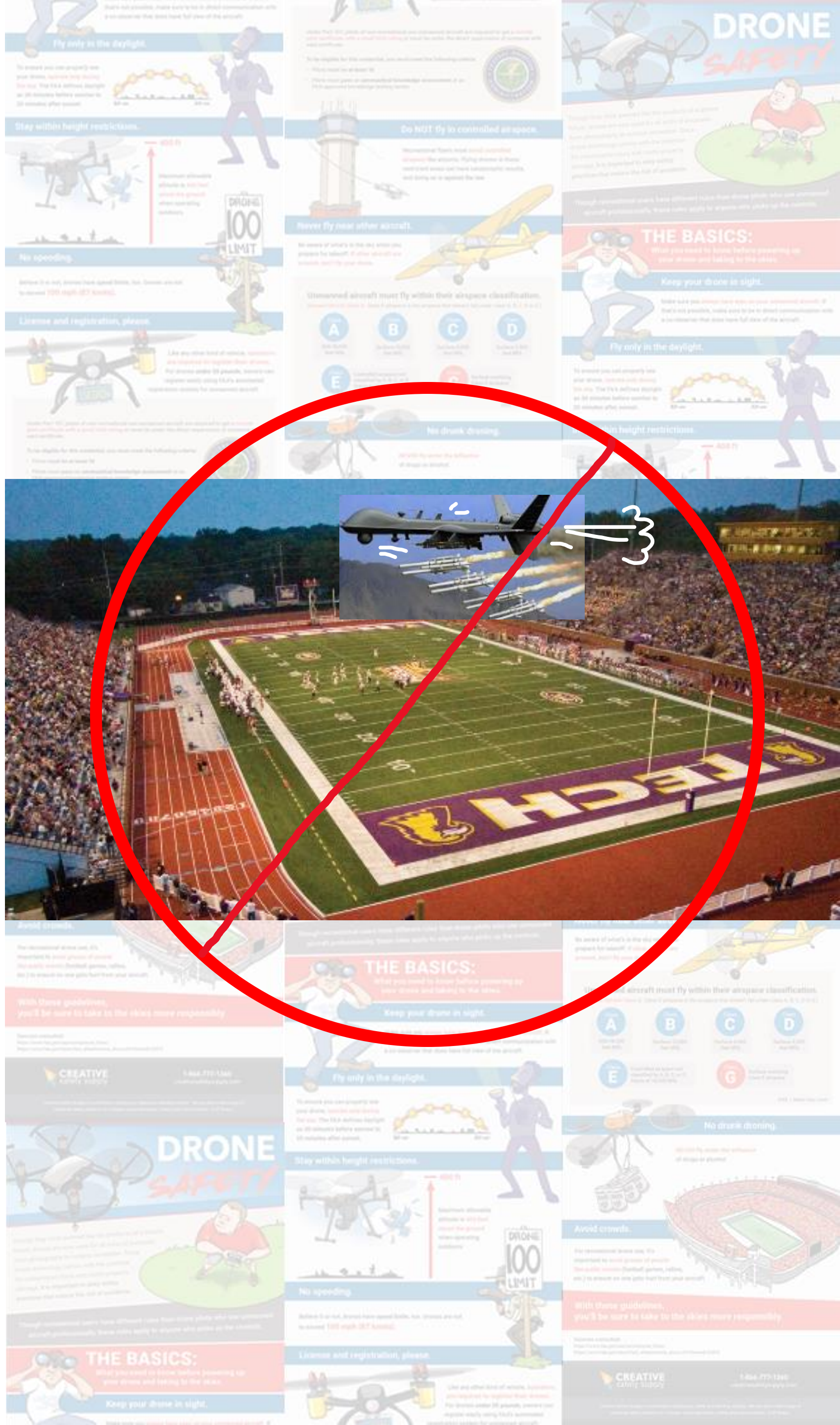
What is Remote ID?

- FAA regulations
  - In effect since Sep 16, 2023
  - Minimum drone weight of 0.55 pounds (250 grams)
  - Broadcast identification and location information over Wi-Fi or Bluetooth
- RID
  - Unique identifiers
  - GPS data
  - Operator information



Problem Formation

- How could drones be misused on campus?
  - Intrude privacy -Property damage -Personal injury
- Tech Police (primary stakeholder) shared insight on what drone misuse most often looks like at TTU
  - Flying over football stadium during competition



PROBLEM STATEMENT

Drones are being misused on campus and the Tech Police do not have a suitable method of enforcing drone regulations.

Broader Impacts

- Campus security and safety is improved.
- Research and development improvements in the drone tracking field
- Concerns about privacy and overregulation

Conceptual Design

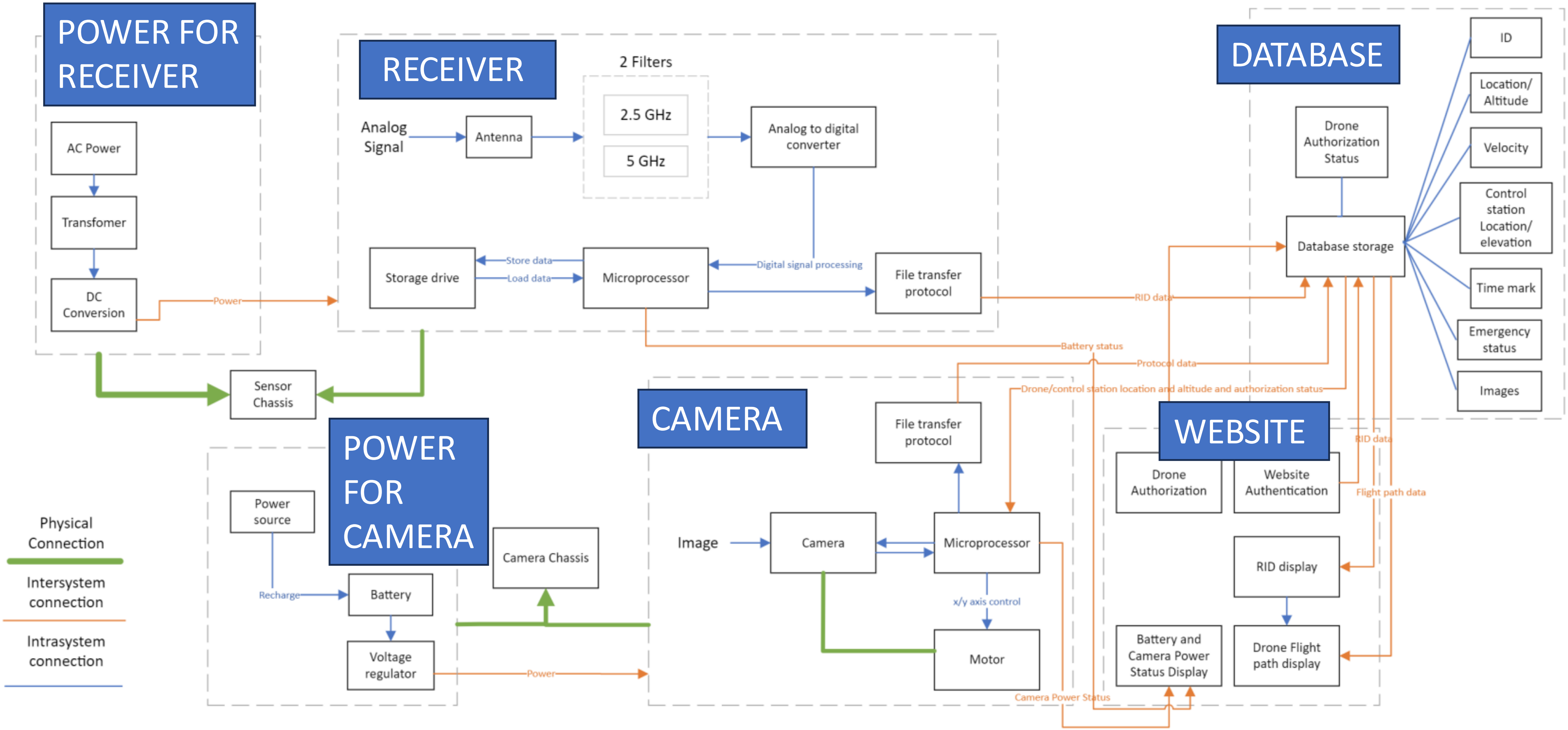
The main questions that outlined the complexity of the problem were:

- What kind of device would we need to pull the RID signal out of the air?
- How do we plan on storing the data that we pull out of the air?
- What are the constituent pieces of each subsystem? What is their purpose?
- What data is going to be transmitted between the systems – how is it encoded?

Critical Design Elements:

- Obtaining the full RID signal.
- Showing the data to the Tech Police in real time.

Block Diagram



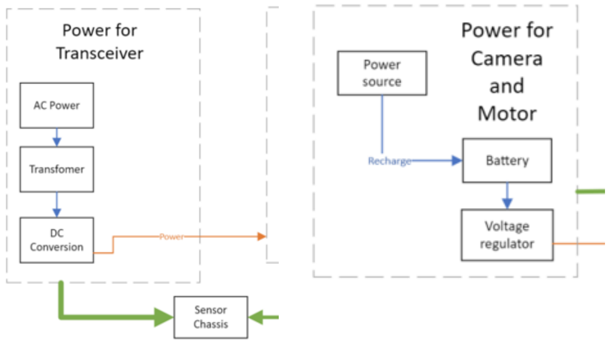
Constraints

Stakeholder constraints:

- Having an easy and secure way to see the drone's location.
- Covering the campus.
- Keeping cost low.
- Prioritizing tracking and capturing pictures of a drone when it is in a sensitive area.
- Allowing some drones to be authorized over certain time periods.

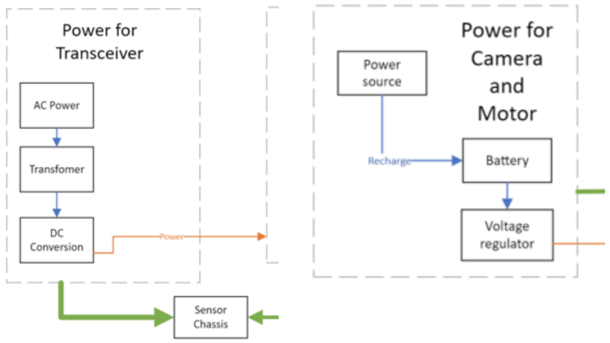
Regulations:

- RID transmission regulation ASTM F3411-22a
- TTU policy 403 which defines surveillance camera usage on campus.



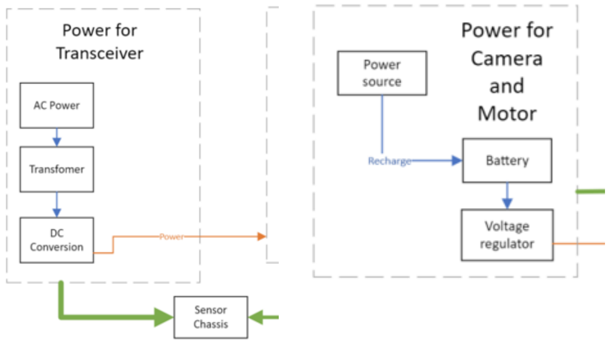
Power System

- Primary Constraints
  - Supply enough power to achieve full functionality
  - Safety features
  - Minimize future maintenance
- The power system was split into two subsystems to not constrain the separately located systems.



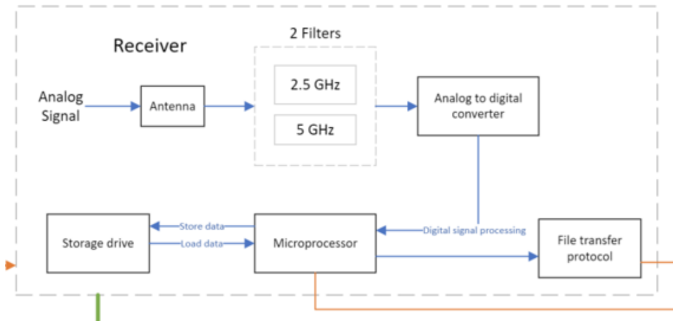
Power System - Camera

- LRS 50-5 Power Supply
  - 50 W 5 V Power supply for the Camera system's servo motors and camera.
  - Fuse, diode, and NEMA box were utilized for protections.
- Raspberry Pi Power Supply
  - Supplies 15 W 5 V to the Camera system's Raspberry Pi 4B



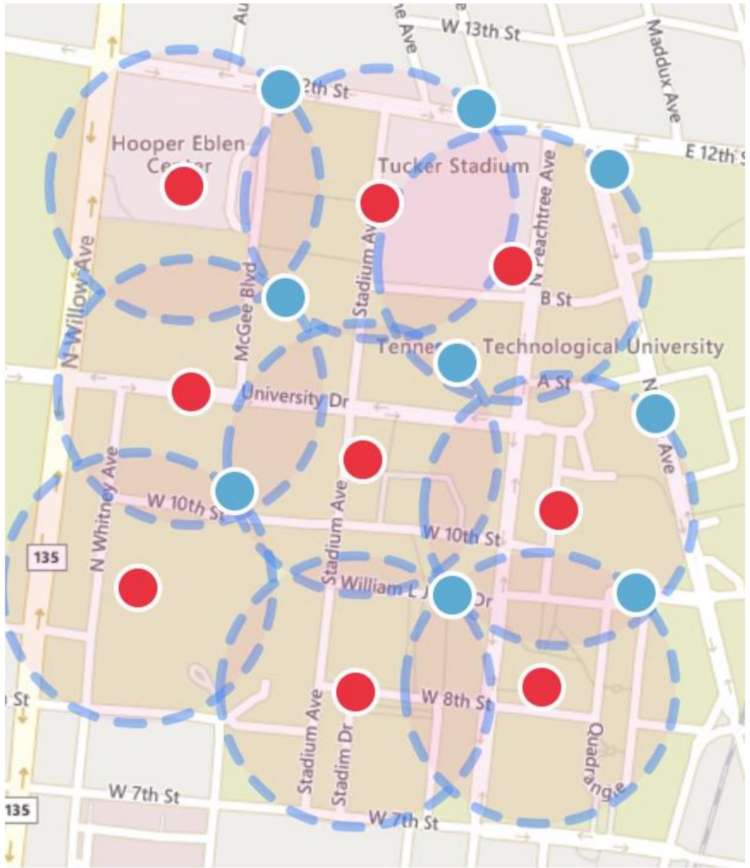
Power System - Receiver

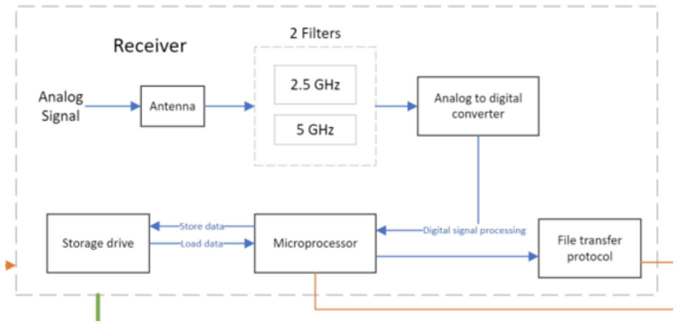
- Waveshare Solar Power Manager Module
  - Uses 14500 Lithium-Ion battery to supply 5 V to the receiver system.
- 5 W Solar Panel
  - Connects to the Waveshare module to charge the battery.



Receiver System

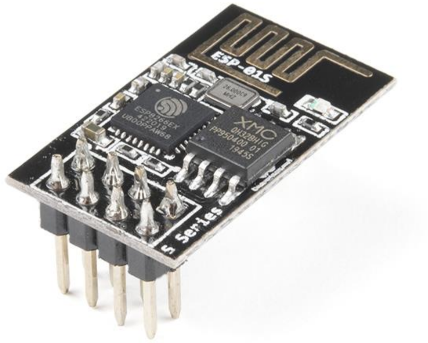
- Receive RID data in 2.4 GHz Wi-Fi and Bluetooth bands
  - 5.8 GHz is an optional add on
- Ignore other signals
- Contiguous TTU Campus

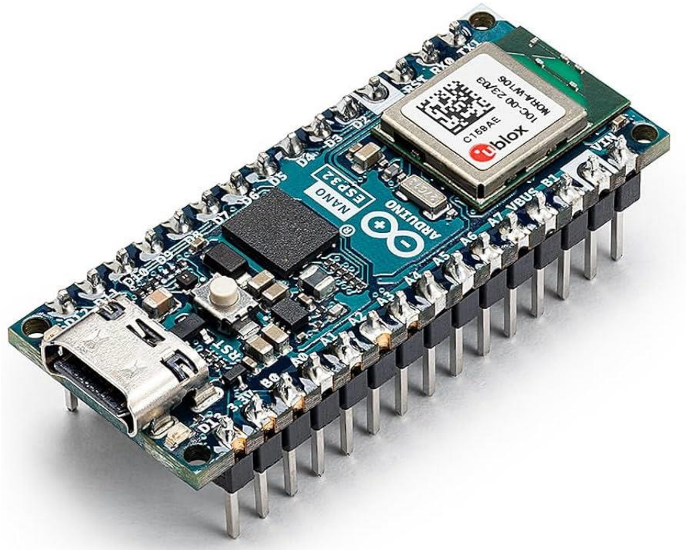


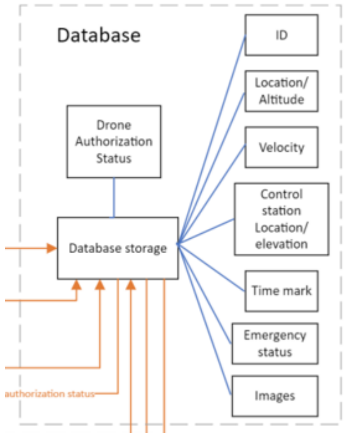


Receiver System

- Arduino Nano ESP32
  - 2.4 GHz Wi-Fi and Bluetooth
  - Receiver Sensitivity –98 dB
  - Programmable
- ESP8266
  - Connects to Wi-Fi

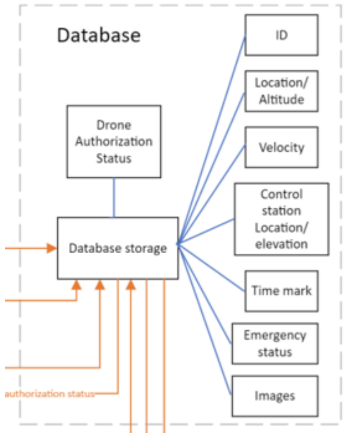






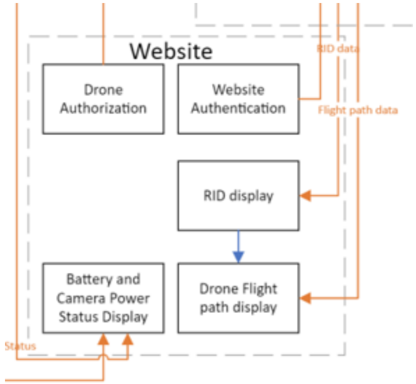
Database System

- Raspberry Pi 5
  - Built-in Wi-Fi
  - 2.4GHz Clock speed
  - 8GB RAM
  - External storage support
  - Host Database and website
- SQL
  - MariaDB



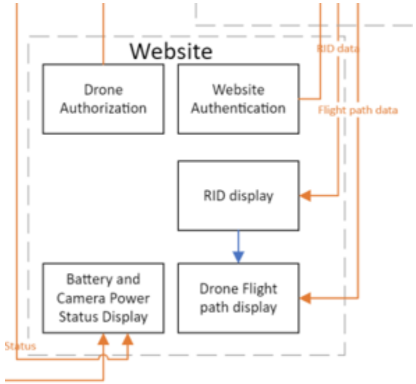
Database System

- Process RID data
  - Extract only the important information
- Store User input
  - Pre-authorized flight information
  - Pre-marked geographical zone
- Camera interaction
  - Drone and control station location
  - Image



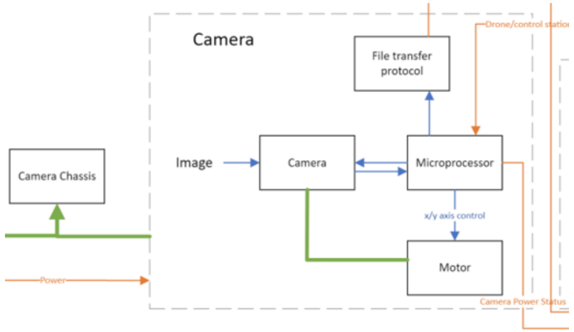
Website System

- Police wanted a website they could use! **NOT** a mobile application.
  - Real-time
  - Secure
  - Compliant
  - Concise
- The system also needed to be able to approve drones for flight – remove from map.
- Increase response urgency if malicious drone detected!



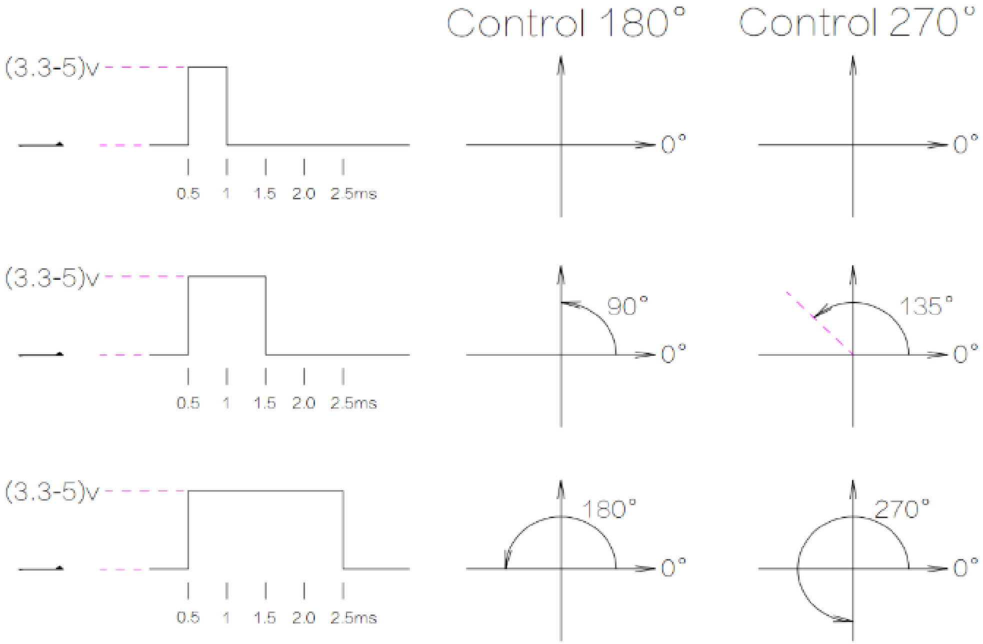
Website System

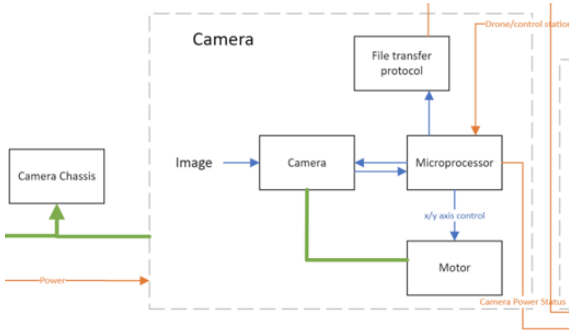
- Details!
  - HTML/CSS
    - Renders the webpage
    - Functionality emphasized over aesthetics.
  - JavaScript
    - Calls Google Maps API
    - Places markers on map
  - Node.js
    - Handles webpage/database communication



Camera Hardware System

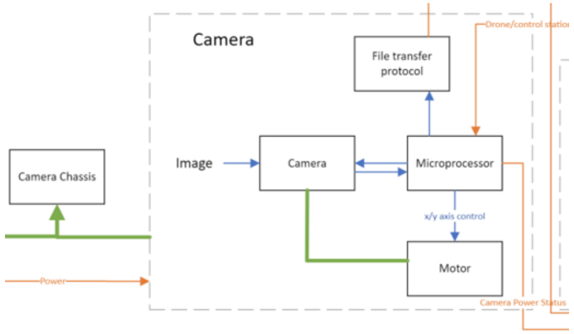
- Primary Constraints
  - Image capture of drone or operator
  - PNG or RAW image type
    - 1080p x 720p resolution
  - Servos pointing accuracy within ± 10°
    - Required angles reached within 500 ms
  - Draw less than 50 Watts total





Camera Hardware System

- Raspberry Pi 4B
  - High processing speed, built in Wi-Fi, GPIO pins
- DS3218 Servo Motors
  - High torque, lower power draw, fast, accurate
  - 270° and 180° rotation
- Generic Pan/Tilt Mounting Kit
  - Cheap, strong, reliable, simple
- IMX477 HQ Camera
  - 12.3 MP camera, native Pi control, varifocal lens

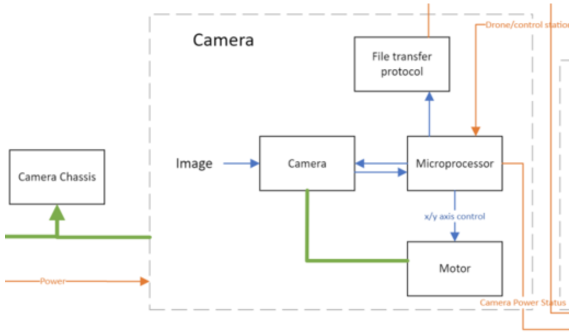


Camera Software System

How accurately and quickly can the system capture a picture of the target?

Main functions:

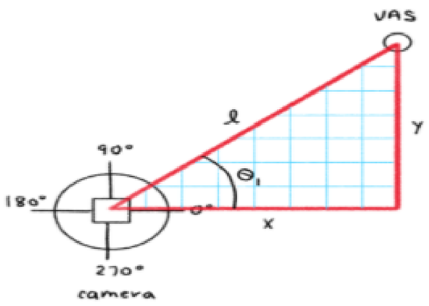
1. Communicates with the Database system.
2. Controls the servo motors and camera functions.
3. Determines what to take a picture of.



Camera Software System

- Focused on making the plan as simple as possible.
  - Performed the math required to (a) predict the location of the target and (b) point at the target
  - Reverse engineered the math.

Horizontal Pointing Angle



x = UAS longitude − Camera longitude

y = UAS latitude − Camera latitude

$\theta_1 = \tan^{-1}(\frac{y}{x})$

If x is a negative number: $\theta_1 = \theta_1 + 180°$

Determining Straight Line Motion

Given 3 points:

$p_1 = (x_1, y_1, z_1), p_2 = (x_2, y_2, z_2), p_3 = (x_3, y_3, z_3)$

The two directional vectors are:

$\overrightarrow{P_1P_2} = (x_1 - x_2, y_1 - y_2, z_1 - z_2)$

$\overrightarrow{P_2P_3} = (x_2 - x_3, y_2 - y_3, z_2 - z_3)$

And their magnitudes:

$|\overrightarrow{P_1P_2}| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$

$|\overrightarrow{P_2P_3}| = \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2}$

If equations 1, 2, and 3, below are all true, straight line motion exists within 5% error margines.

1.

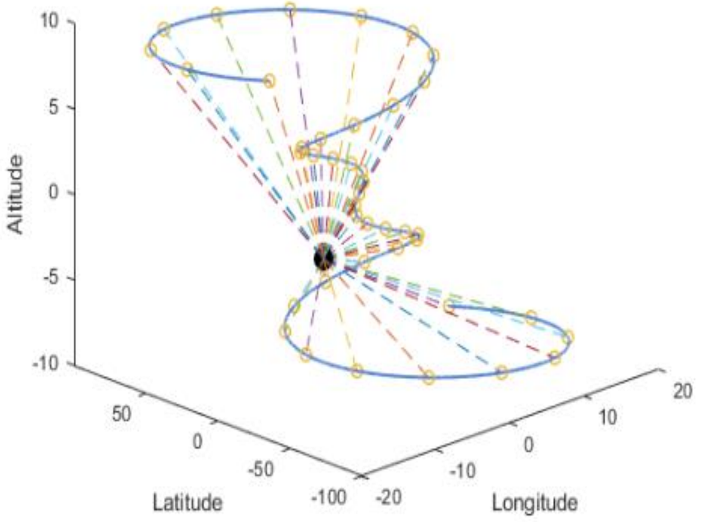
$(\frac{x_1 - x_2}{|\overrightarrow{P_1P_2}|} * .95) \leq \frac{x_2 - x_3}{|\overrightarrow{P_2P_3}|} \leq (\frac{x_1 - x_2}{|\overrightarrow{P_1P_2}|} * 1.05)$

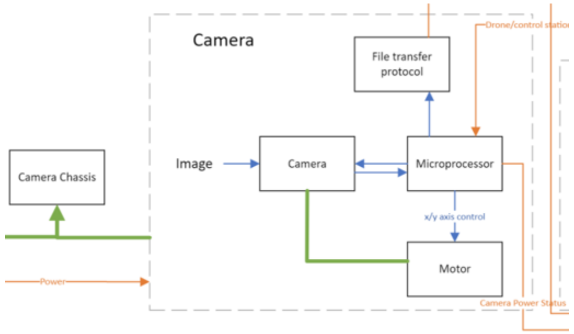
2.

$(\frac{y_1 - y_2}{|\overrightarrow{P_1P_2}|} * .95) \leq \frac{y_2 - y_3}{|\overrightarrow{P_2P_3}|} \leq (\frac{y_1 - y_2}{|\overrightarrow{P_1P_2}|} * 1.05)$

3.

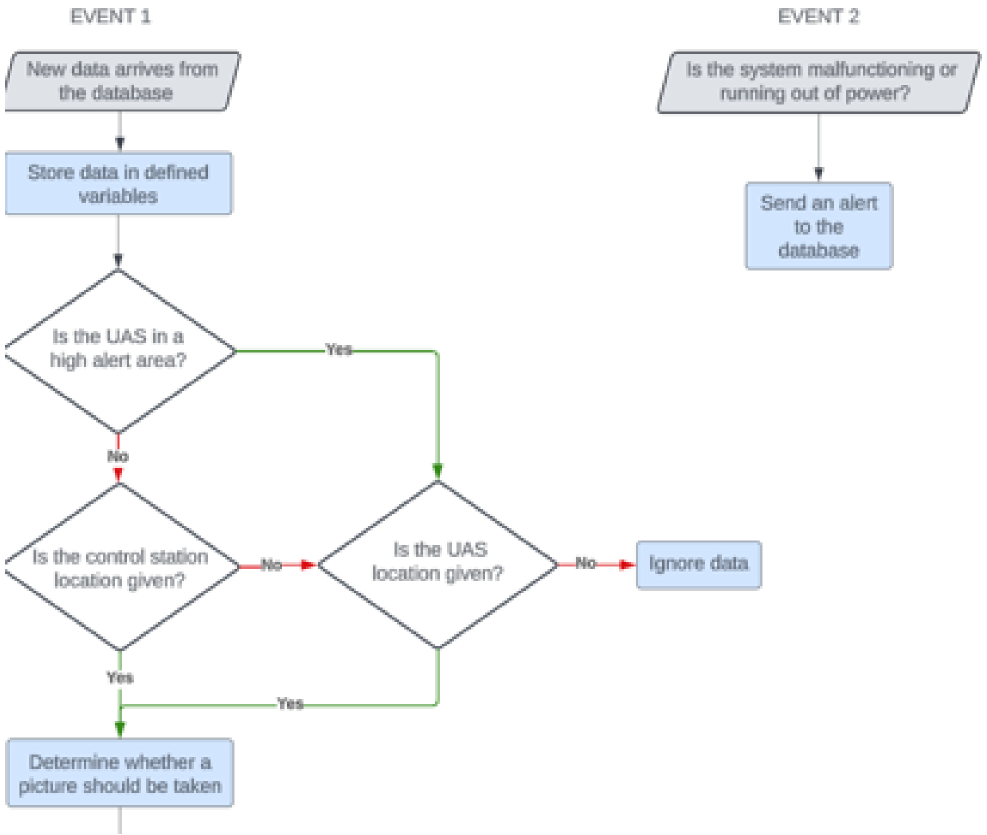
$(\frac{z_1 - z_2}{|\overrightarrow{P_1P_2}|} * .95) \leq \frac{z_2 - z_3}{|\overrightarrow{P_2P_3}|} \leq (\frac{z_1 - z_2}{|\overrightarrow{P_1P_2}|} * 1.05)$

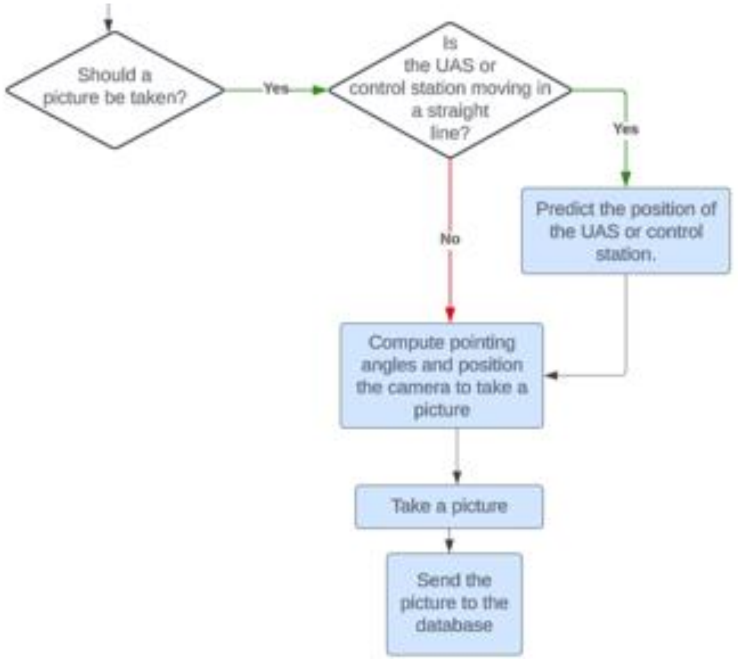




Camera Software System

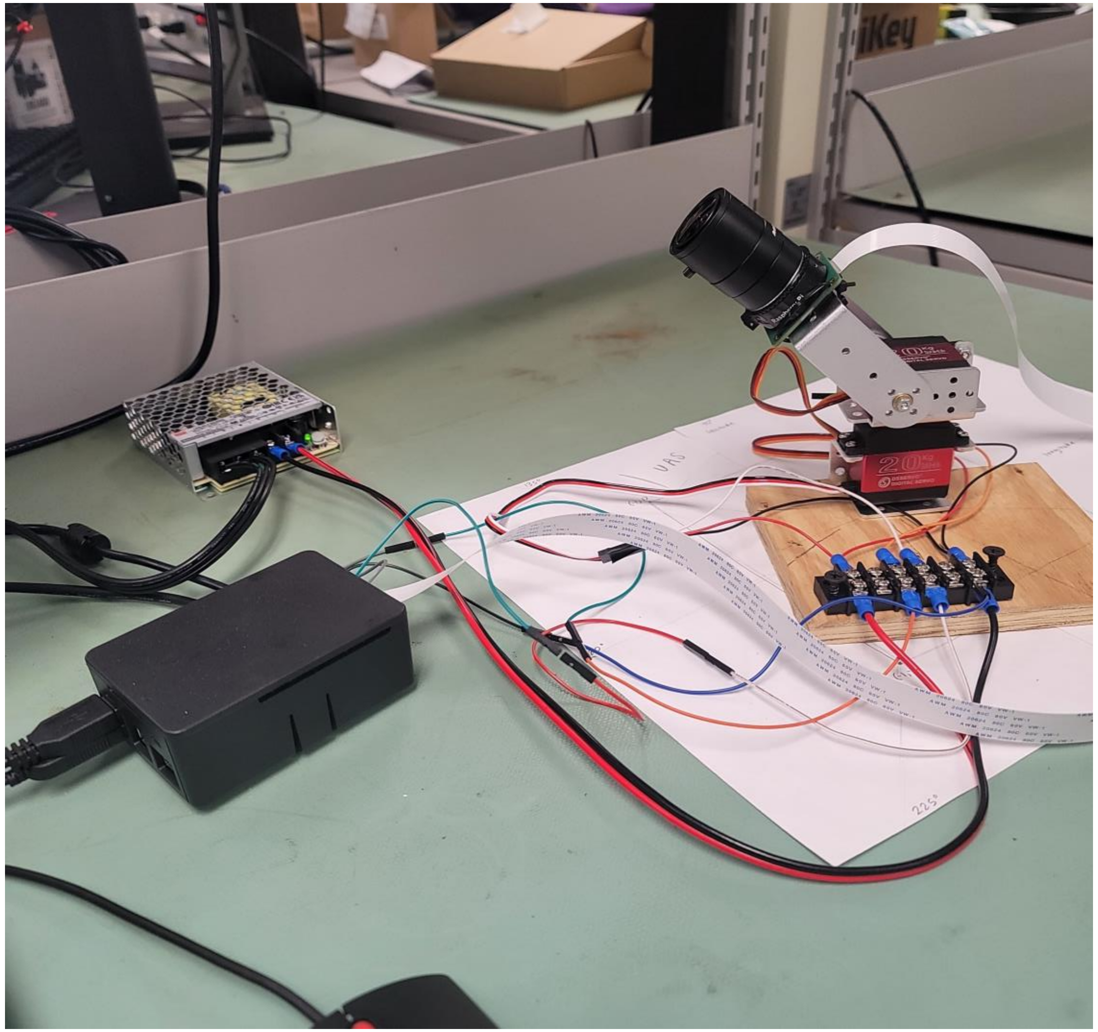
- Created a flowchart of system actions
- Outlined expected functions and variables





Physical Implementation
(Camera Hardware)

- Servo bracket kit installation
  - Servos set to 0° with function generator for proper adjustment
- Terminal block installation
  - +5V, Spare, Tilt PWM, Pan PWM, Spare, Ground
- PWM wires attached to GPIO pins
- Camera zip-tied to tilt servo arm
  - Temporary solution, no enclosure
  - Cable installed into CSI port



Code Implementation
(Database)

- List of Table
  - Drone Location
  - Operator Location
  - Priority Zone
  - Images

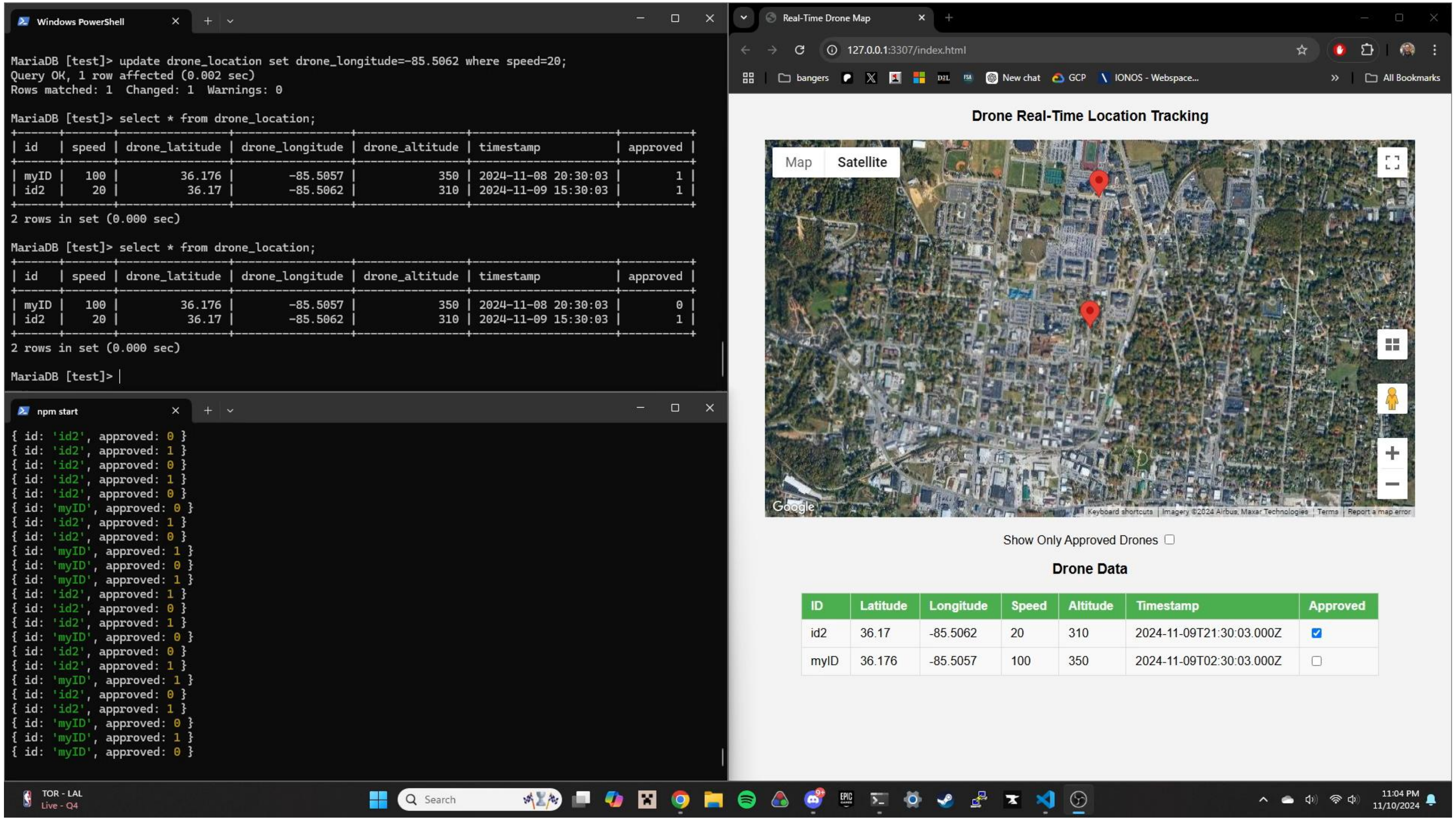
```
MariaDB [RID]> describe drone_location;
+-----------------+--------------+------+-----+---------+----------------
| Field           | Type         | Null | Key | Default | Extra          |
+-----------------+--------------+------+-----+---------+----------------
| list            | int(11)      | NO   | PRI | NULL    | auto_increment |
| id              | varchar(255) | YES  |     | NULL    |                |
| speed           | float        | YES  |     | NULL    |                |
| drone_latitude  | double       | YES  |     | NULL    |                |
| drone_longitude | double       | YES  |     | NULL    |                |
| drone_altitude  | float        | YES  |     | NULL    |                |
| timestamp       | varchar(20)  | YES  |     | NULL    |                |
| approved        | tinyint(1)   | YES  |     | 0       |                |
+-----------------+--------------+------+-----+---------+----------------
```

```
MariaDB [RID]> describe priority_zone;
+---------------+---------+------+-----+---------+----------------
| Field         | Type    | Null | Key | Default | Extra          |
+---------------+---------+------+-----+---------+----------------
| list          | int(11) | NO   | PRI | NULL    | auto_increment |
| max_latitude  | double  | YES  |     | NULL    |                |
| min_latitude  | double  | YES  |     | NULL    |                |
| max_longitude | double  | YES  |     | NULL    |                |
| min_longitude | double  | YES  |     | NULL    |                |
+---------------+---------+------+-----+---------+----------------
```

```
MariaDB [RID]> select * from images;
+------+----------+-------------------+-------------------------------------------------------------------+
| list | id       | image_name        | image_path                                                        |
+------+----------+-------------------+-------------------------------------------------------------------+
|    1 | ABC12345 | 24-12-05 01:36:36 | /home/tntech/Documents/capstone/ABC12345/24-12-05 01:36:36.jpg |
+------+----------+-------------------+-------------------------------------------------------------------+
```

Code Implementation (Website)

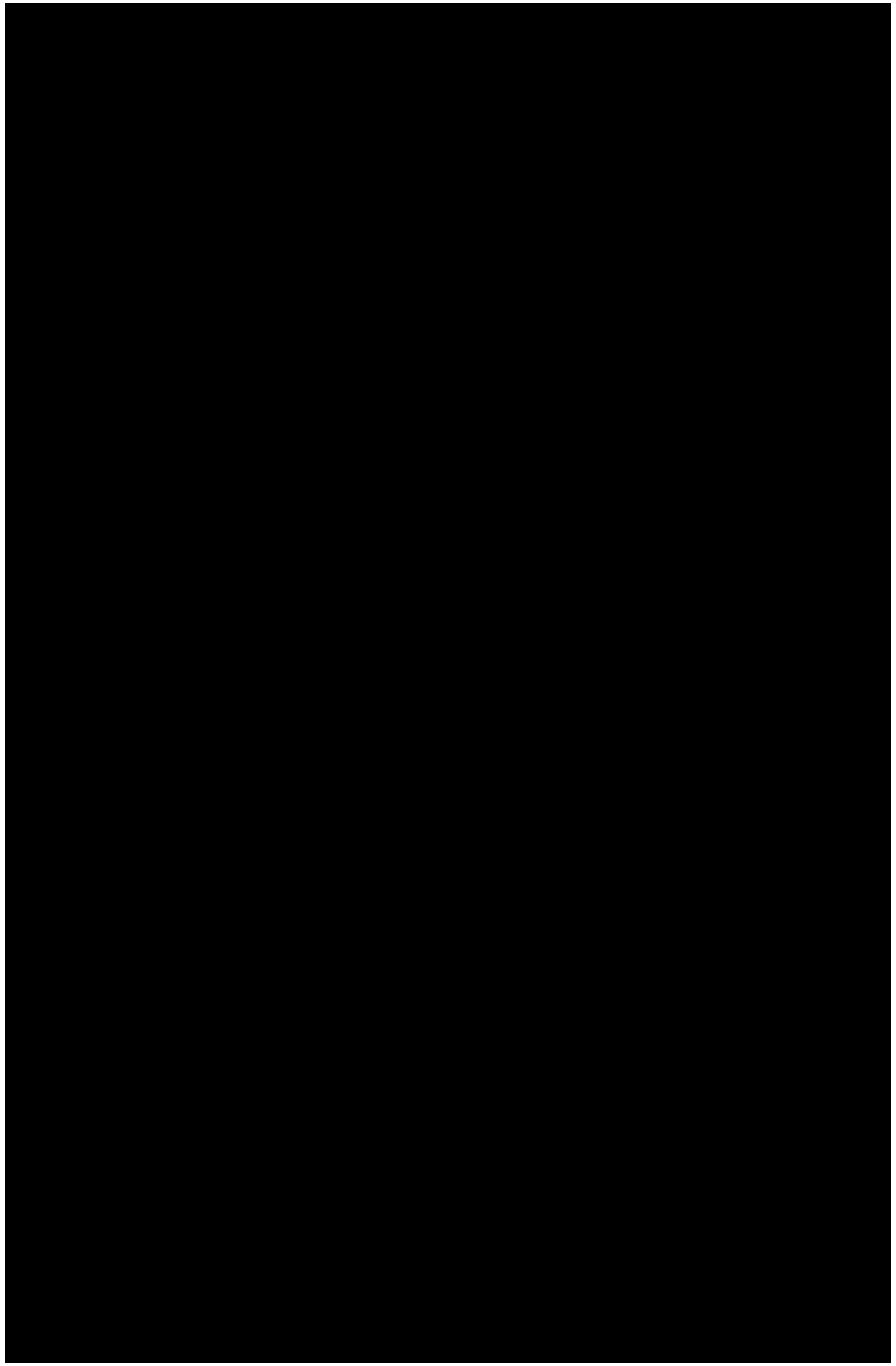
https://youtu.be/dnVWmRdjKF0



Code Implementation
(Camera Software)

https://youtube.com/shorts/AOBYHYJI4ZE?feature=share

```
int main() {
    int picNum = 0;
    check_and_launch_rpicam();

    while (1) {

            receive_data();

            store_data(ridData);

             cout << "\nAll Stored RID Data: " << endl;
                for (const auto& data : RID_data) {
                    cout << "Latitude: " << data.latitude << ", "
                      << "Longitude: " << data.longitude << ", "
                      << "Altitude: " << data.altitude << ", "
                      << "Speed: " << data.speed << ", "
                      << "CS Latitude: " << data.cs_latitude << ", "
                      << "CS Longitude: " << data.cs_longitude << ", "
                      << "High Priority Area: " << (data.highPriorityArea ? "Yes" : "No") << endl;
                }

            analyze_data();

            determine_slm();

            predict_location();

            determine_angles();
             cout << "\nPointing Angles: " << theta_h << " degrees" << ", " << theta_v << " degrees" << endl;

            move_camera();

            capture_image(picNum);

            send_image(picNum);

            picNum++;

    }
}
```

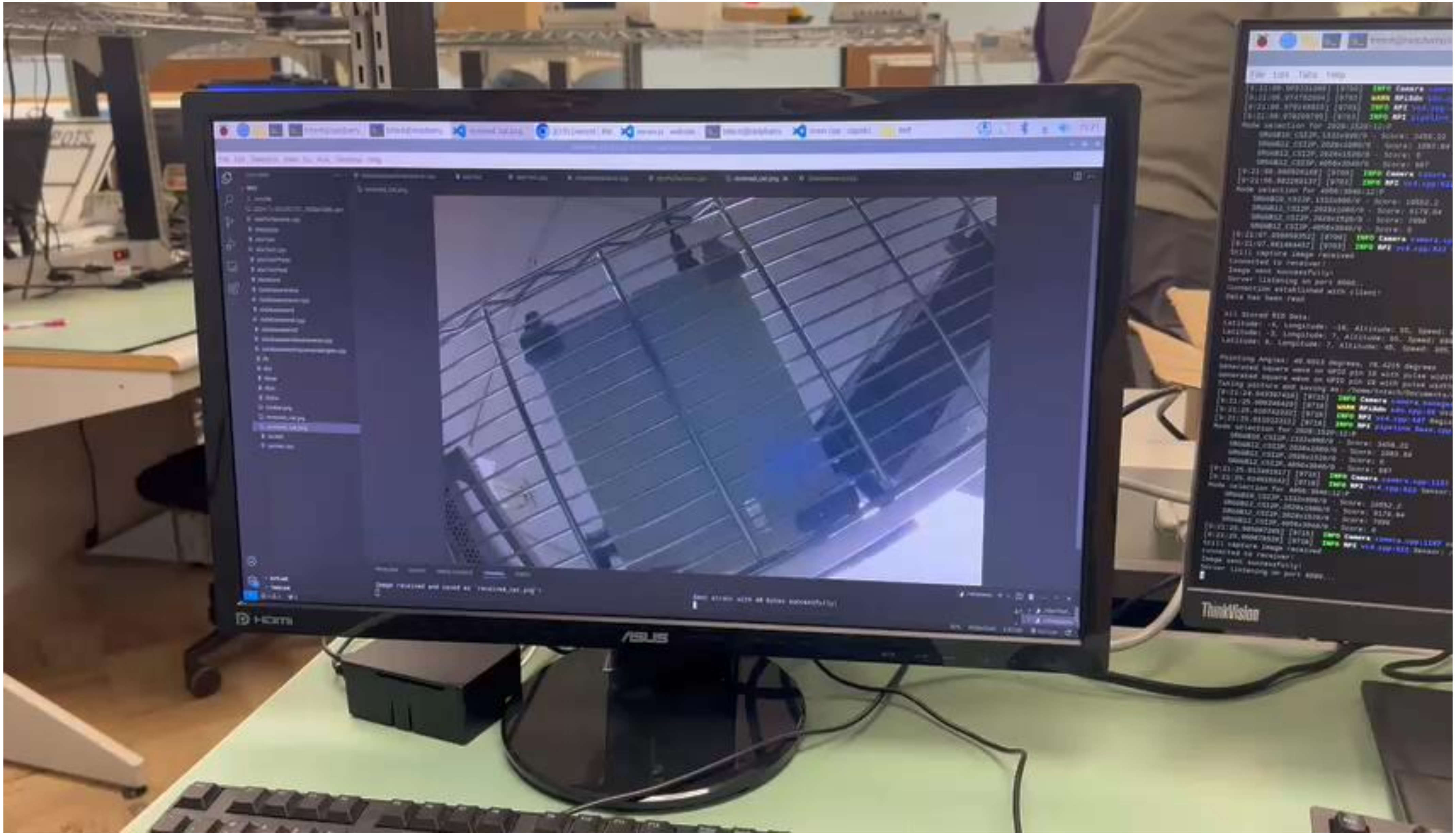


Experimentation

- For the experimentation phase of our project, the team was working with a version of the system that was parsed into different groups.
  - Database - Camera Software - Camera Hardware – Camera Power
  - Receiver Power – Receiver – Database
  - Database – Website
- Goal was to test a few constraints from each subsystem

Project Demonstration

https://youtu.be/FysjEsHIlvY



Experimentation Outcomes

Did we meet measure of success?

Ways we did:

- Power system allowed the receiver and camera systems to achieve full functionality.
- Camera system and database communicated and worked together well.
- Camera moves towards correct target depending on the priority area status.
- Database system is able process the received signal within 100 millisecond.
- Website system clearly displays information from the database.

Ways we didn't:

- Receiver system does not properly pick up or send RID packages.
- The camera module does not take a picture fast enough to keep up with a 1 second RID transmission frequency.
- The camera system does not handle sending and receiving data separately.
- The website encounters a bug when displaying an image.

Improvements

- Obtain an RID transmitter and/or a device that can pick up a Bluetooth package so that the full system can be tested.
    - Allow for full functionality.
    - Testing of real system range and processing time.
    - Testing of location prediction for camera software.
- Need to investigate the Raspberry Pi camera libraries as well as socket functions for transmitted data more effectively.

Budget

Proposed Budget	
Subsystem	**Cost**
Receiver	$429.48
Camera	$340.04
Power	$716.43
Database	$161.08
Website	$30.00
Total	$1,677.03

Final Budget	
Subsystem	**Cost**
Receiver	$31.99
Camera	$284.71
Power	$151.25
Database	$175.82
Website	$0.00
Total	$643.77

- Some components were not ordered to save costs
  - Receiver – 9 full systems, enclosures and exterior protections
  - Camera – Processor, Enclosures and Exterior protections
  - Power Systems – 9 receiver power systems, Enclosures and exterior protections

Highlighting Supplementary Roles*

- Sterling and Austin handled team communication
  - Meetings with the Tech police
  - Extra meetings with Dr. Austen
- Meredith was the team manager and organizer
  - Submitted reports and meeting minutes.
  - Created schedules, timelines, and outlines.
- Aaron and Austin were team scribes
  - Recorded meeting minutes

*Abe and JieJun were very helpful, they just did not have dedicated roles

Citations

References

[1] "Remote identification of drones," Federal Aviation Administration, Available: https://www.faa.gov/uas/getting_started/remote_id [Accessed Feb. 15, 2024].

[2] "190 unmanned aircraft," Tennessee Technological University, Available: https://tntech.navexone.com/content/dotNet/documents/ [Accessed Mar. 7, 2024].

[3] "Part 107- unmanned aircraft systems," Title 14- Aeronautics and Space, Code of Federal Regulations, Availiable: https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-107 [Accessed Feb. 15, 2024].

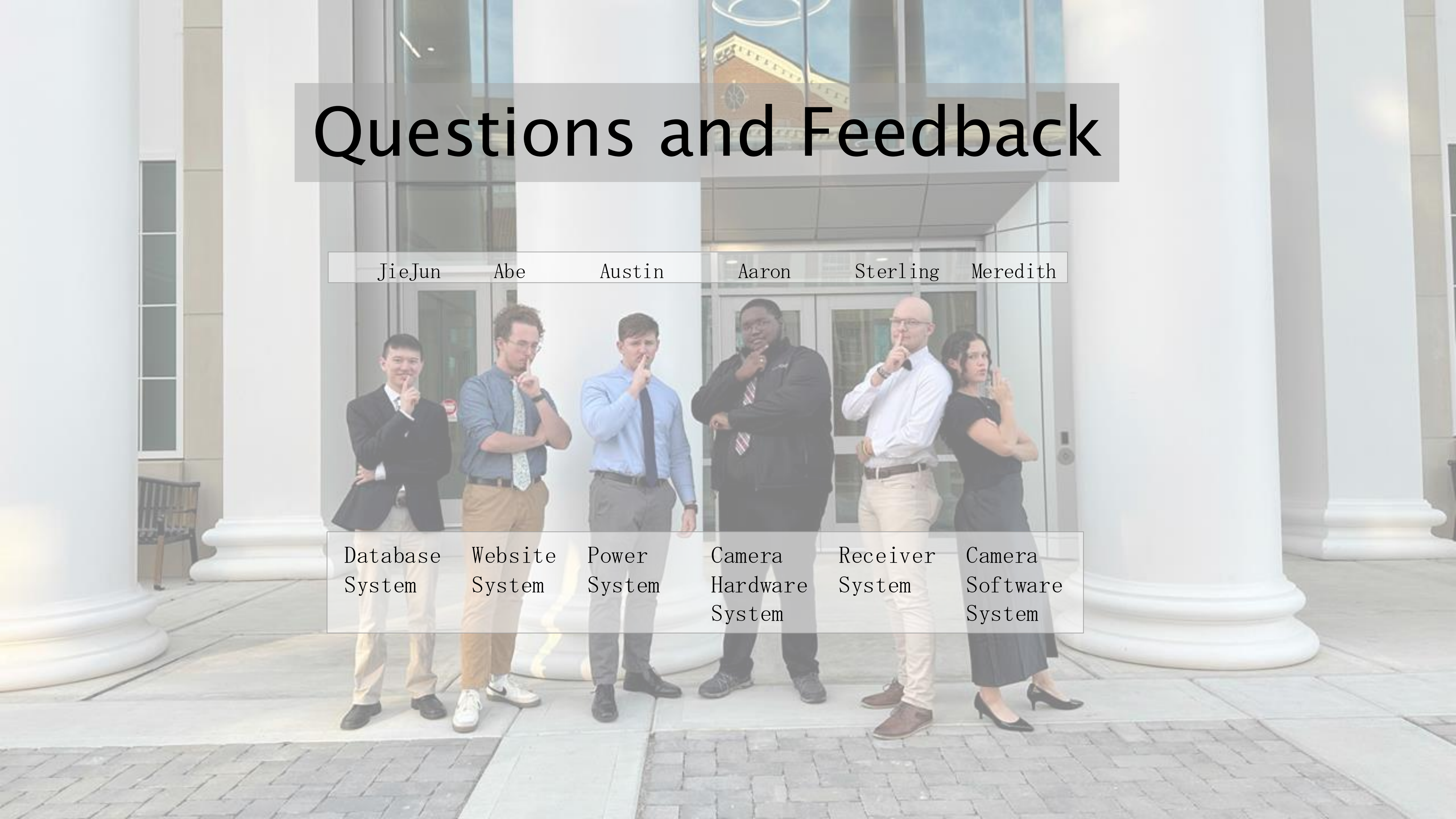
[4] "Part 15- radio frequency devices," Title 47- Telecommunication, Code of Federal Regulations, Available: https://www.ecfr.gov/current/title-47/chapter-I/subchapter-A/part-15 [Accessed Feb. 19, 2024].

[5] "Part 89- minimum message elements broadcast by standard remote identification unmanned aircraft," Title 14- Aeronautics and Space, Code of Federal Regulations, Available: https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-89/subpart-D/section-89.305 [Accessed Feb. 15, 2024].

[6] "Ieee 802.11-2020: Collision avoidance in wireless networks," American National Standards Institute, Available: https://blog.ansi.org/ieee-802-11-collision-avoidance-wireless-networks/ [Accessed Feb. 19, 2024].

[7] "403 safety and security camera acceptable use," Tennessee Technological University, Available: https://tntech.navexone.com/content/dotNet/documents/ [Accessed Feb. 19, 2024].

[8] "856 data security and handling policy," Tennessee Technological University, Available: https://tntech.navexone.com/content/dotNet/documents/ [Accessed Feb. 19, 2024].



Questions and Feedback

JieJun	Abe	Austin	Aaron	Sterling	Meredith
Database System	Website System	Power System	Camera Hardware System	Receiver System	Camera Software System

Thank you, Dr. Austen!