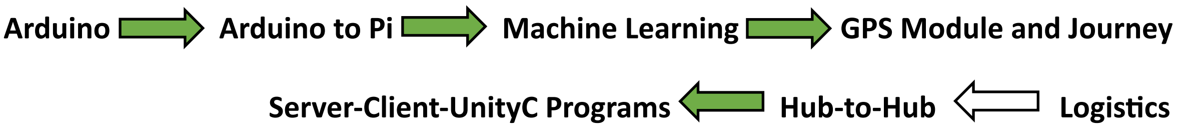
- **Arduino Program:** Integrated sensors code gets uploaded to the Nano.
- **Arduino to Pi Program:** Reads the sensor data from the Arduino and saves the data to a new CSV file every 200 sets of data.
- **Machine Learning Program:** Takes the read sensor data every 180 seconds and finds the Status and Condition based on the training set and that gets appended to the original sensor data.
- **GPS Module and Journey Logistics Program:** For now, it takes the current location and finds the distance in km and miles and how long it will take to get there using the GPS speed and using the assumption of walking speed od 3 miles per hour. It does this for a given latitude and longitude. We will be finding it for these buildings Wheatly, McCormick, Campus Center, University Hall, etc. and find how much consumables you need to get there, Oxygen, Water, and battery. This is get appended to the full Hub data file
- **Hub-to-Hub Program:** Sends the full Hub data file to the other hub and receives the hub's full data file with all the astronauts information. It sends the hub data every 180 seconds in chunks of 200 bytes. And receives the full appended data as saves it to a file.
- **Server-Client-UnityC Programs:** The python Client sends hub 1 and hub 2 CSV data files every 180s to the server and the Server saves them to two different folders; Hub 1 and Hub2 and the Unity Client receives the data as it is sent and saves it to two folders in its Assets to use to display.
- **Hub1/2 bash script:** A Bash script that executes a sequence of all the Python scripts in the background, with specified time intervals between them. As some of the background processes are an infinite loop, the script will not terminate until all the programs are manually stopped or the computer is rebooted, therefore the off button still doesn't terminate the script.
- **On/Off Program:** This Python script is designed to run on a Raspberry Pi computer to start the Hub1/2 bash script with a push button to toggle the state of the system on/off and turns on/off LEDs to indicate the state of the system. It listens for button press events on a specific GPIO pin, and when the button is pressed, it toggles the state of the system on/off and turns on/off LEDs to indicate the state of the system. The script also starts and stops a bash script in a subprocess, depending on whether the system is on or off. The script continuously checks for button press events using a while loop and uses the RPi.GPIO library to interact with the GPIO pins on the Raspberry Pi.


Arduino
Arduino to Pi
Machine Learning
GPS Module and Journey
Logistics
Hub-to-Hub
Server-Client-UnityC Programs