Drone Tracking Web Application SOP

Ashton Smith, Andrew Freeman, London Glenn | **Latest Revision:** SAT 3MAY2025

Overview:

This document has been prepared for users wanting to emulate this project from my GitHub. This will be a locally hosted application that shows you drone tracking in real time based on simulated data. The information is being sent via a Python script with hundreds of JSON packets. The application is using Python Flask as the back end, then using HTML, CSS, and JavaScript to make everything interactive and flashy.

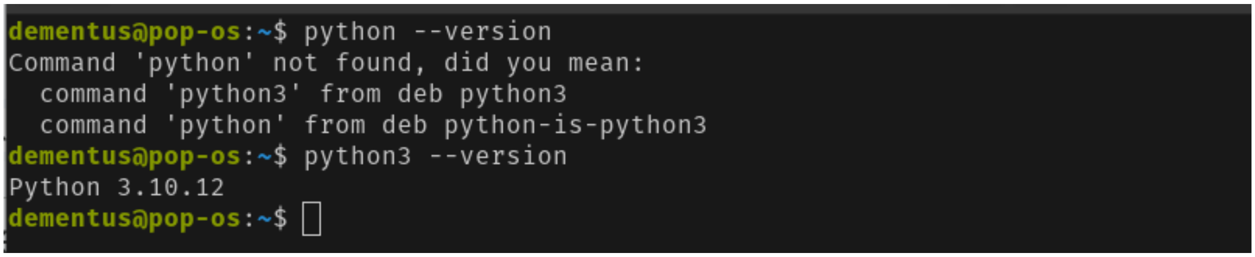
Hardware Setup:

**If it's the first time then start here, otherwise move on.*

1. Identify the system that will host the application. ***IT MUST BE LINUX or MAC***

Installing Python:

1. Open a bash terminal.
2. Check to see if you already have python installed by entering either of the following and pressing enter, you should see a number as output not an error: python –version or python3 –version



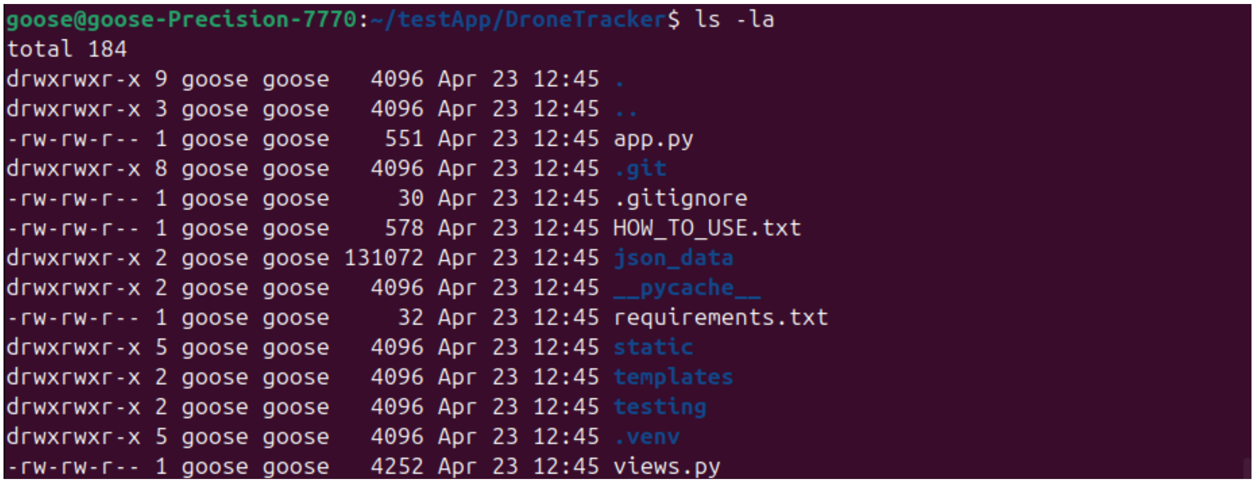
```
dementus@pop-os:~$ python --version
Command 'python' not found, did you mean:
  command 'python3' from deb python3
  command 'python' from deb python-is-python3
dementus@pop-os:~$ python3 --version
Python 3.10.12
dementus@pop-os:~$ 
```

3. This output shows that I have python version 3.10.12
4. If you don't have python installed then run this command: sudo apt update && sudo apt install python3-dev python3-venv build-essential
5. Run the checker commands again to ensure you have it downloaded.
6. Complete.

<u>Software Download:</u>

**If it's the first time then start here, otherwise move on.*

1. Within the Linux VM, or hosting device, open a bash terminal.
2. In this terminal, create a directory called 'apps' by typing: mkdir apps
3. Navigate to that directory: cd apps
4. Download the web application files from Github: git clone https://github.com/ashtontheanalyst/DroneTracker.git
5. Check to see if they're there: ls -la
   a. You should see a directory called 'DroneTracker'
6. Navigate into the directory: cd DroneTracker
7. Inside DroneTracker, check for files again: ls -la
   a. Should see an output similar to this:



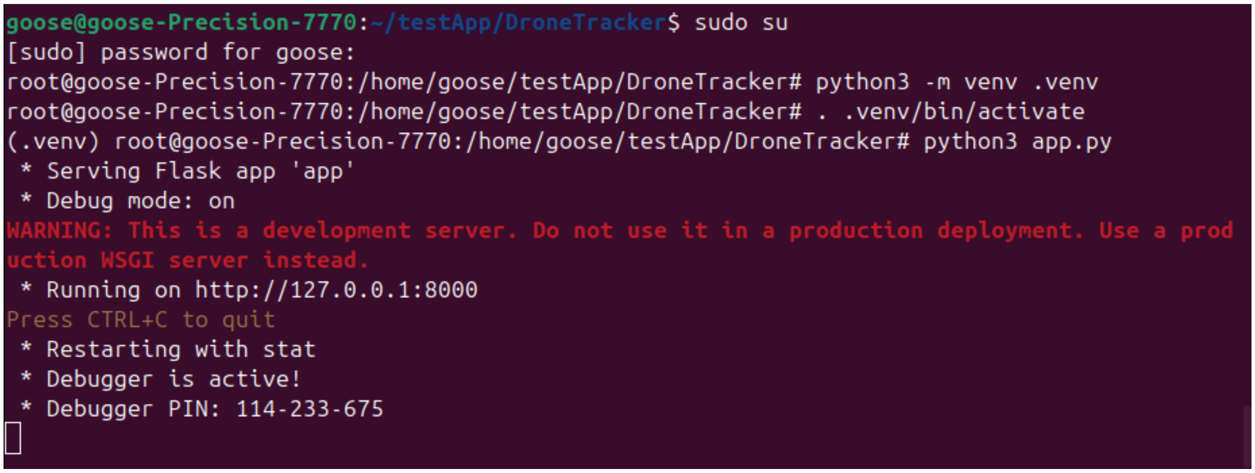
```
goose@goose-Precision-7770:~/testApp/DroneTracker$ ls -la
total 184
drwxrwxr-x 9 goose goose   4096 Apr 23 12:45 .
drwxrwxr-x 3 goose goose   4096 Apr 23 12:45 ..
-rw-rw-r-- 1 goose goose    551 Apr 23 12:45 app.py
drwxrwxr-x 8 goose goose   4096 Apr 23 12:45 .git
-rw-rw-r-- 1 goose goose     30 Apr 23 12:45 .gitignore
-rw-rw-r-- 1 goose goose    578 Apr 23 12:45 HOW_TO_USE.txt
drwxrwxr-x 2 goose goose 131072 Apr 23 12:45 json_data
drwxrwxr-x 2 goose goose   4096 Apr 23 12:45 __pycache__
-rw-rw-r-- 1 goose goose     32 Apr 23 12:45 requirements.txt
drwxrwxr-x 5 goose goose   4096 Apr 23 12:45 static
drwxrwxr-x 2 goose goose   4096 Apr 23 12:45 templates
drwxrwxr-x 2 goose goose   4096 Apr 23 12:45 testing
drwxrwxr-x 5 goose goose   4096 Apr 23 12:45 .venv
-rw-rw-r-- 1 goose goose   4252 Apr 23 12:45 views.py
```

8. Complete.

Starting the Application:

**You should already have the files downloaded, if not then see above.*

1. Make sure you are in the apps/DroneTracker directory.
2. Become super user: sudo su
    a. Enter password of the device if prompted
3. Initialize python virtual environment: python3 -m venv .venv
4. Start up the virtual environment and get into it: . .venv/bin/activate
5. Download necessary packages: pip install -r requirements.txt
6. Start the application: python3 app.py
    a. Should see the following output:



```
goose@goose-Precision-7770:~/testApp/DroneTracker$ sudo su
[sudo] password for goose:
root@goose-Precision-7770:/home/goose/testApp/DroneTracker# python3 -m venv .venv
root@goose-Precision-7770:/home/goose/testApp/DroneTracker# . .venv/bin/activate
(.venv) root@goose-Precision-7770:/home/goose/testApp/DroneTracker# python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a prod
uction WSGI server instead.
 * Running on http://127.0.0.1:8000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 114-233-675
```

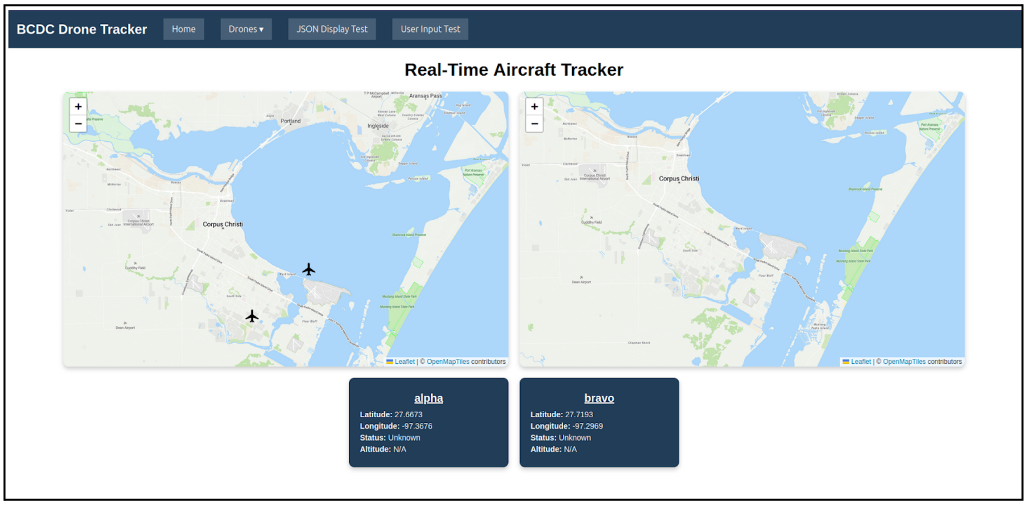
7. Complete.

Using the Application:

**Make sure you've completed the 'Start the Application' section*

1. With the application running, now open a browser of your choice like Chrome or Firefox.
2. In the browser, go to: http://127.0.0.1:8000
    a. Sometimes it can be tricky to load so just refresh the page a couple times if need be. Or, restart the application in the command line.
3. From here the application is open and you can navigate to the different pages.

Home Page:

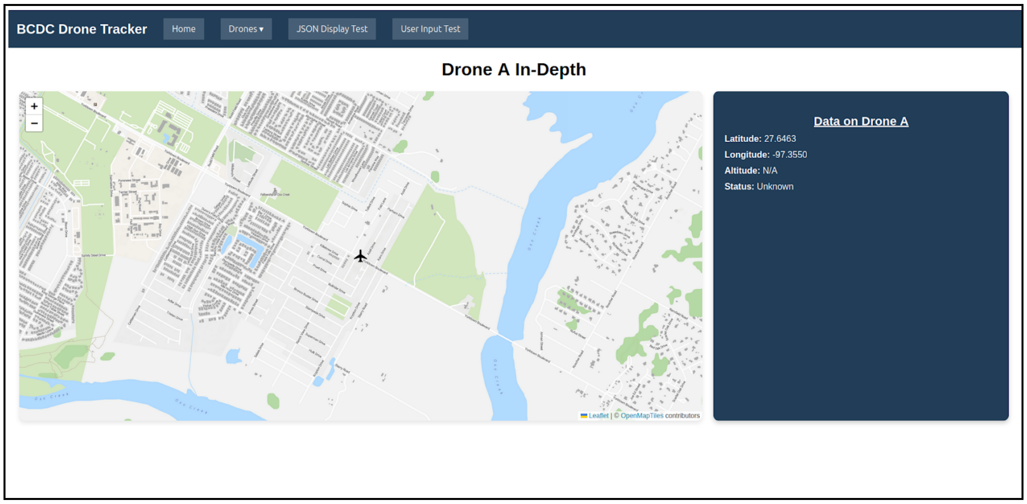


Real-Time Aircraft Tracker

alpha
Latitude: 27.6673
Longitude: -97.3676
Status: Unknown
Altitude: N/A

bravo
Latitude: 27.7193
Longitude: -97.2969
Status: Unknown
Altitude: N/A

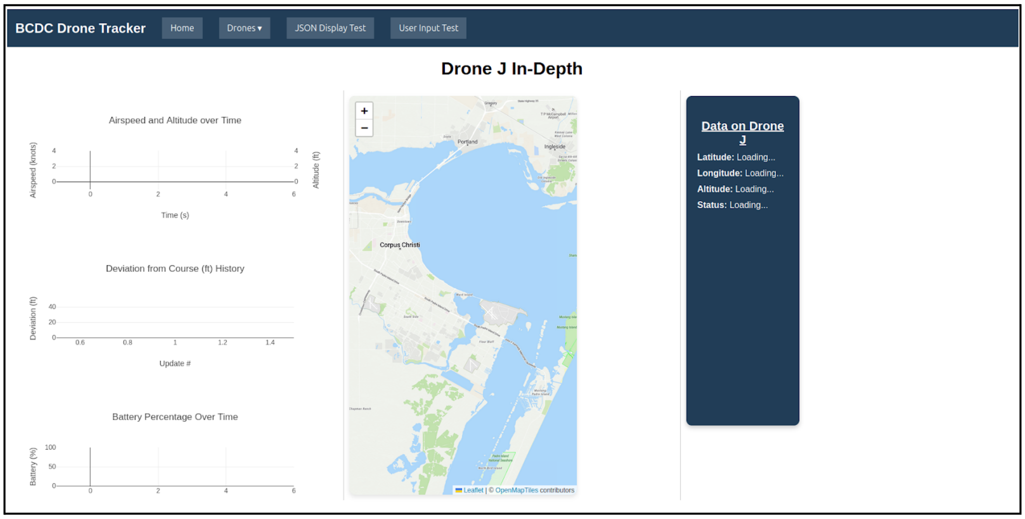
Drone A and B Page:



Drone A In-Depth

Data on Drone A
Latitude: 27.6463
Longitude: -97.3550
Altitude: N/A
Status: Unknown

Drone J Page:



Drone J In-Depth

Data on Drone J

Latitude: Loading...
Longitude: Loading...
Altitude: Loading...
Status: Loading...

*Should be blank for now, this is the site that ingests live data and then displays it. See below for how to use this page.

JSON Display Test:

Used for testing but otherwise not important or useful, could be removed.

User Input Test:

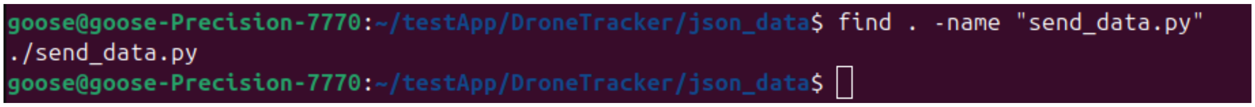
Used for testing but otherwise not important or useful, could be removed.

Drone J Page:

This page is able to ingest live JSON data from outside and then display it on screen. This page is most important for the actual test. In theory we would have four pages of this, one for each simulated drone on test day. Drone A and B pages are both simulated for us to show off the proof of concept.

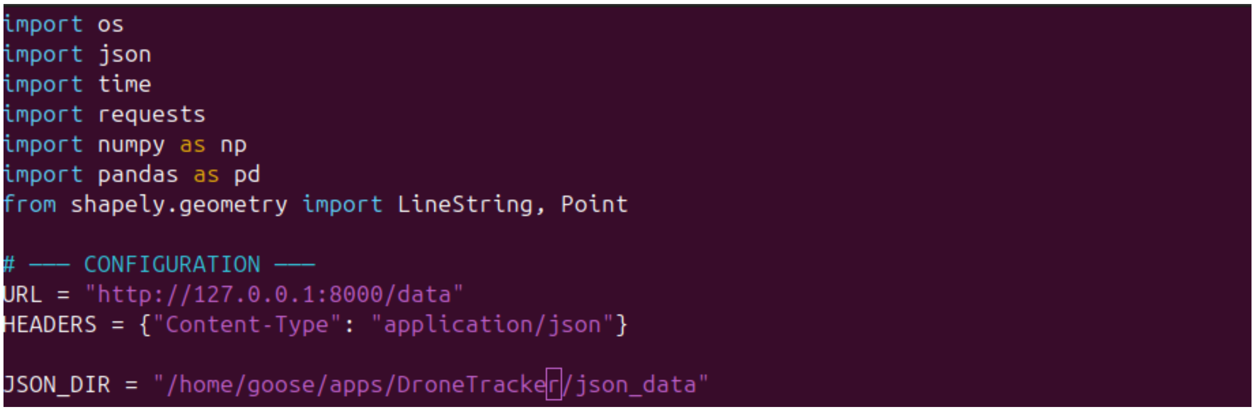
**Make sure all sections have been completed prior*

1. Ensure the site is live and you have it open in your browser.
2. Navigate to Drone J's page.
3. On the linux machine, open up a NEW terminal. Make sure not to interfere with the terminal that is running the web application.
4. Navigate to the directory holding the JSON script: cd ~/apps/DroneTracker/json_data
5. Ensure the script is there: find . -name "send_data.py"
   a. Should see an output like this confirming it's there:



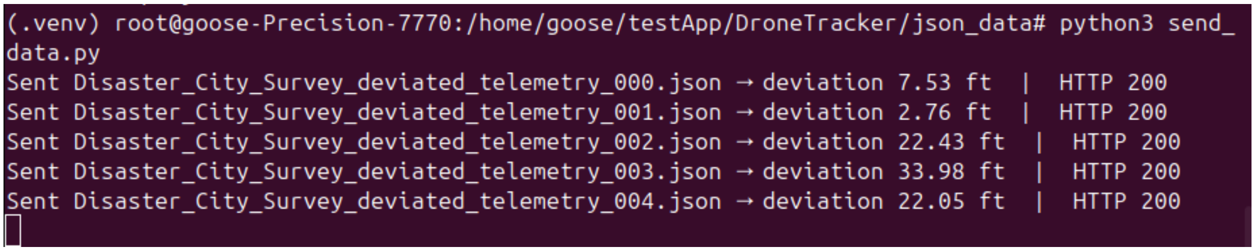
```
goose@goose-Precision-7770:~/testApp/DroneTracker/json_data$ find . -name "send_data.py"
./send_data.py
goose@goose-Precision-7770:~/testApp/DroneTracker/json_data$ 
```

6. Download vim (text editor): sudo apt install vim
7. Open the file in vim: vim send_data.py
8. Edit the file by first clicking the 'i' key, this will get you into insert or 'edit' mode.
9. **IMPORTANT:** Then with your arrow keys navigate down to the JSON_DIR = "/some/something" section. Change that directory string to be the one you're currently in, so most likely that will be "/home/user/apps/DroneTracker/json_data".
   a. Will looking something like this:



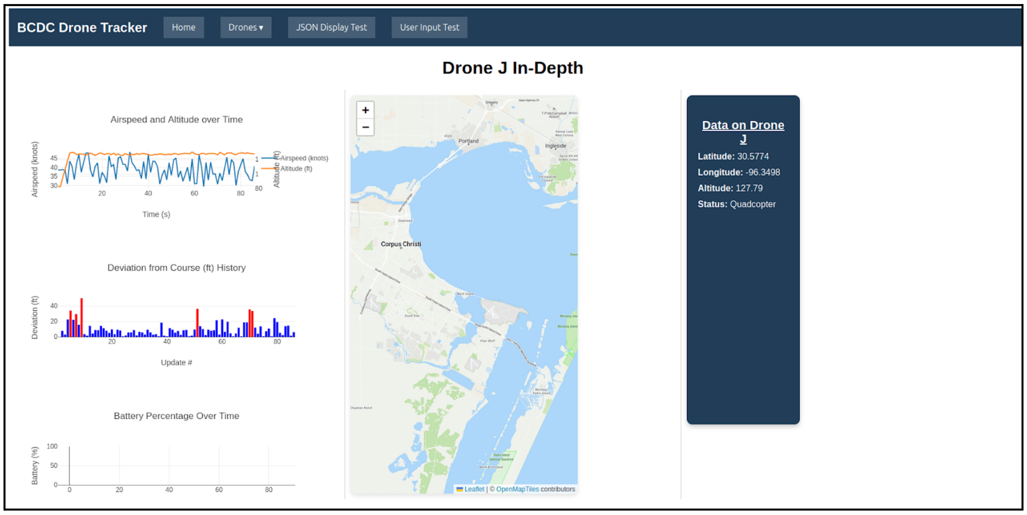
```
import os
import json
import time
import requests
import numpy as np
import pandas as pd
from shapely.geometry import LineString, Point

# ---- CONFIGURATION ---
URL = "http://127.0.0.1:8000/data"
HEADERS = {"Content-Type": "application/json"}

JSON_DIR = "/home/goose/apps/DroneTracker/json_data"
```

10. Hit the ‘esc’ key to leave edit mode, then type in “:wq” to save and exit.
11. Become super user: sudo su
    a. Enter password of the device if prompted
12. Initialize python virtual environment: python3 -m venv .venv
13. Start up the virtual environment and get into it: . .venv/bin/activate
14. Download necessary packages: pip install -r requirements.txt
15. Run the script: python3 send_data.py
    a. Should see an output like this:



```
(.venv) root@goose-Precision-7770:/home/goose/testApp/DroneTracker/json_data# python3 send_
data.py
Sent Disaster_City_Survey_deviated_telemetry_000.json → deviation 7.53 ft  |  HTTP 200
Sent Disaster_City_Survey_deviated_telemetry_001.json → deviation 2.76 ft  |  HTTP 200
Sent Disaster_City_Survey_deviated_telemetry_002.json → deviation 22.43 ft  |  HTTP 200
Sent Disaster_City_Survey_deviated_telemetry_003.json → deviation 33.98 ft  |  HTTP 200
Sent Disaster_City_Survey_deviated_telemetry_004.json → deviation 22.05 ft  |  HTTP 200
```

16. Check the webpage for live data updating:



Drone J In-Depth

Data on Drone J

Latitude: 30.5774
Longitude: -96.3498
Altitude: 127.79
Status: Quadcopter

17. Complete.