

Ground Control Station for Uncrewed Swarms and Teams



Nick Warren



Jacob Stock



Ricardo R-C



James Montgomery



Cameron Acree

Github: <https://github.com/rickysrc123/GUSTv2>
Website: <https://rickysrc123.github.io/GUSTv2/>

Project Overview



The end goal of GUST is to have a fully functioning ground control station that

- Can connect to and control multiple vehicles simultaneously
- Can plan and execute maneuvers for single or multiple vehicles
- Live telemetry and sensor readings from vehicles

Sprint Goals



- Finish Implementation of Telemetry Screen
- Design & Implement Planning Screen
- Integrate Telemetry Screen with Backend
- Integrate Planning Screen with Backend
- Design & Implement Vehicle Connection in Backend
- Test Multi-Vehicle Control in Simulation (SIL Testing)
- Test Vehicle Control on Hardware (HIL Testing)

Team Contributions

Jacob S.

- Mapping Widget
- Vehicle List
- Vehicle Telemetry Display
- Telemetry Screen Styling
- Integration

Ricardo RC.

- Planning Widget
- Maneuver Creation
- Planning Screen Styling
- Hardware Testing

James M.

- Planning Database Support
- Database refactor
- Integration

Nick W.

- FastAPI Endpoints
- FastAPI/DB Integration
- Hardware Test script
- Custom Docker Images.

Cameron A.

- Test code for single drone
- Test code for drone swarm
- Mavproxy testing
- Gazebo testing
- Hardware Testing



GET /test_data/get_generated_data Generate Data



GET /drones Get All Drones



POST /drones/create Create Drone



POST /drones/{drone_name}/delete Delete Drone



POST /drones/{drone_name}/update_position Update Drone Position



GET /maneuvers Get Maneuvers



POST /maneuvers/create Create Maneuver



POST /maneuvers/delete Delete Maneuver



POST /maneuvers/assign_to_drone Assign Path To Drone




POST /maneuvers/update_path Update Path



GET / Read Root





Sprint 2 Backlog

- Front end screen 2 drone list [done, ~2hrs]
- Front end screen 2 Path design component [done, ~8hrs]
- Front end screen 2 save/load paths for selected vehicle
[In Progress, ~3hrs]
- Front end screen 2 swarm paths component [done, ~4hrs]
- Backend support for screen 2 implemented [In Progress, ~6hrs]
- Backend integrated with screen 2 [In Progress, ~6hrs]
- Emergency land procedure [Done, ~6hrs]
- Teleop control of vehicles [In Progress, ~8hrs]

Sprint 1 Backlog

- Database setup and installed on docker container [done, ~2hrs]
- Database designed and implemented [done, ~6hrs]
- FastAPI installed and tested [done, ~3hrs]
- FastAPI backend for telemetry designed [done, ~3hrs]
- FastAPI backend for telemetry implemented [done, ~2hrs]
- Backend and db integrated [In Progress, ~3hrs]
- Front end design [done, ~2hrs]
- Telemetry screen fully implemented [In Progress, ~6hrs]
- Front end integrated with backend and DB [In Progress, ~6hrs]
- MavLink proof of concept and flight in sim [done, ~6hrs]
- MavLink test on hardware [Done, ~6hrs]

Sprint Reflection



- What went well
 - Front end screens look good and integration with backend is almost fully complete
 - FastAPI and database have their own container but run through the same compose
 - General API architecture upgraded to support thousands of requests per second

Sprint Reflection



- What didn't go well
 - Styling on both front-end screens
 - Hardware testing has been difficult because lab pilot has been busy
 - Setting up multiple connections in sim was more difficult than expected
 - Team communication especially around PR's

Sprint Reflection



- What could be improved
 - Better documentation
 - More in depth system level testing and unit testing
 - More descriptive commit messages as well as better PR practice

Cybersecurity Reflection

- Docker Passwords
 - Enable passwords on all containers
- Input and Request Validation
 - Ensure that all input is not dangerous
 - Ensure that all requests come from the app and not a malicious program



Demo Description

- Front End Screen #1
- Front End Screen #2
- Hardware Test (take off/land on S9000)

