Distributing a Fleet of Drones over an Area with No-Fly Zones

DESIGN DOCUMENT

Team 21

Goce Trajcevsk - Client/Advisor

Nicholas Kokott - Team Organizer

Melani Hodge - Frontend Design/Implementation

Everett Duffy - Component/Module Design

Cole Stuedeman - Testing

Kenneth Schueman - Advisor Communication and Assistance

Samuel Russett - Research Discovery and Testing

sdmay25-21@iastate.edu

Revised: Version 1 | 12/6/2024

Executive Summary

This project is a web-based application designed to manage drone operations within specified areas, including no-fly zones. The goal of the project is to create an application where users can define their drone fleet size, the survey region, and any restricted zones within that area. Currently, no applications efficiently map no-fly zones and deploy multiple drones. This application will be one of the first to utilize partitioning and pathing algorithms to manage multiple drones while avoiding restricted areas. Upon starting the web application, the user can enter the region, the no-fly zones, and the number of drones. Once the user selects the inputs, the region will be split into partitions equal to the number of drones. After the map is partitioned, the user will have the ability to add events to the map therefore allowing drones to respond to these events based on which partition they are in. First, for the frontend of this application, we will use React and Vite which utilizes both JavaScript and TypeScript. Next, for our map display, we will use MapBox API to display the region, drones, and partitions to the user. Finally, for the backend, we will use Python (Django) to run the algorithms used for partitioning and drone management. Our backend code will allow us to display drone routes back to the UI for the user hopefully in real-time. So far, we have developed a frontend prototype of the application. Our backend is on hold due to waiting for an algorithm from the grad student we are working with. Our current design meets the requirements and addresses the users' needs. We have extensively discussed with our client/advisor about what is expected in the final design and what is working well in the current design. For our next steps, we will implement the application's backend, integrate Mapbox for coordinate handling and drone visualization, and develop a cohesive visual interface.

Learning Summary

Development Standards & Practices Used

- Agile Task Management Methodology
- Object Oriented Programming
- Data Structures
- IEEE Std 1012, Standard for Software Verification and Validation
- IEEE Std 1219, Standard for Software Maintenance
- IEEE/ISO/IEC 26512-2017, Requirements for acquirers and suppliers of information for users
- IEEE Std 982.1, Standard Dictionary of Measures to Produce Reliable Software
- IEEE/ISO/IEC 15288-2015, System life cycle processes

Summary of Requirements

- Design a web application that allows users to place events for drones to respond to
- User should be able to input number of drones, location, and no-fly zones
- User should be able to determine which pathing algorithm they would like to utilize for their drones
- UI should be clean and easy to use
- Application should respond quickly and be able to demonstrate drone flight correctly

Applicable Courses from Iowa State University Curriculum

- CS 227: Object Oriented Programming
- CS 228: Introduction to Data Structures
- CS 309: Software Development Practices
- CS 317: Introduction to Software Testing
- CS 311: Introduction to the Design and Analysis of Algorithms
- CS 319: Construction of User Interfaces
- CS 327: Advanced Programming Techniques
- CS 329: Software Project Management
- CS 352: Operating Systems Concepts

New Skills/Knowledge acquired that was not taught in courses

- React+Vite
- TypeScript
- Python
- MapBox API

Table of Contents

Tables and Figures

Tables

Figures

1 Introduction

1.1 Problem Statement

In the last several years, drones and automation have become incredibly popular tools for many scenarios. This mostly comes from the recent affordability for the average drone and automation consumers, but also due to the wide variety of applications drones can be utilized for. Many people may think drones are simply utilized for scanning environments and video work. Still, they also can be used for search and rescue operations, delivery services, as well as maintenance work. However, many of these drones cannot be used in an automated manner but rather have drawn out flight paths and event management. It would be significantly more straightforward to have all of this automated so that many important resources delegated to planning could be used elsewhere.

This project aims to develop a web application that allows users to input a given area or dataset of no-fly zones, as well as several drones, and then be able to visualize how these drones would respond to events within that given environment. Depending on the number of drones that are given, the area will be partitioned into smaller areas that avoid no-fly zone regions while optimizing response time to critical events. Each drone will be limited to responding to events within their partitioned region to ensure the fastest response times. Specifically, this project considers scenarios that users can utilize in the real world and can see drone responses to events in real-time.

1.2 Intended Users

The project aims to create a visualization for users to see an automated fleet of drones over areas the users consider important. Due to this, there are many different users, with several different use cases that can be considered.

1. Emergency Services
    a. Drones will respond to natural disaster events and evaluate damages
        i. Weather damage ii. Wildfires
        iii. Floods iv. Volcanic Activity
    b. Police usage
        i. Search large regions and rescue people in need
        ii. Track crimes that could be occurring
    c. Firefighters
        i. Track the shape of a large fire and find potential sources
        ii. Mark buildings that have fires
2. Delivery of Goods
    a. Grocery store deliveries
    b. Supply deliveries for disaster relief
    c. Pharmaceutical deliveries
3. Agriculture
    a. Water crops that are too dry
    b. Monitor different land regions
    c. Spread fertilizer in needed locations
4. Infrastructure
    a. Monitor electrical elements of cities (power lines, generators, transformers)
    b. Perform basic repairs on rooftops

2 Requirements, Constraints, And Standards

2.1 Requirements & Constraints

Our project has many requirements that can be divided and organized into several categories listed below:

UI Requirements

- Users are able to input the number of drones, the location to be used, as well as no-fly regions to be shown to them
- Users are able to see the partitioned regions with one drone per region
- Users are able to see the output of the algorithm in real-time
- Users are able to see the movement of the drones in real-time
- Users are able to input events for the drones to respond to
- Users are able to start/stop the simulation at any time
- Users are able to navigate to the home, contact, and About pages
- Users are able to select the pathing algorithm they desire most

Security Requirements

- Users can input real-world data without an attacker being able to compromise
- The server will stay active and not be broken into by outside actors
- Each session will be unique to each user and cannot be repeated unless the same information is provided

Backend requirements

- All data will be stored and used within each session
- The backend will be fresh with no stored data in each new session
- The backend will call out to external pathing API to path the drones and their respective flights
- The backend will take in data from the frontend as users provide it and use it to determine how drones respond to events
- The backend will partition the map based on the users initial input of no-fly zones
- The backend will do all algorithmic calculations

- The backend will perform the selected pathing algorithm from the frontend

Economic Requirements

- Will need to host a server for x amount of dollars

Resource Requirements

- Each drone used will be equipped with high-precision sensors, GPS, IMUs, cameras, and navigation systems that can call out to our API.
- The server used must be powerful enough to perform real-time calculations of flight paths, process large datasets, and support concurrent drone operations without delays.

Physical Requirements

- Drones must be able to operate effectively through the 2D space that the user gives.
- Drones must withstand weather conditions (heat, rain, cold).

2.2 Engineering Standards

Engineering standards are important as they will ensure safety, quality, and consistency across engineering projects. They help engineers use a common language and set of expectations that allow them to collaborate, regardless of their location or background efficiently. With these standards in place, engineers can avoid errors, streamline processes, and foster innovation through a shared understanding of practices.

These standards are very much relevant to our project. The first one discusses how to properly layout your architecture and connect your frontend and backend components to be of quality and look good to your users. Developing things isolationally will allow us to scale our product and its components properly without interfering with our other components or services. The second standard discusses how to securely manage our users' sessions and ensure that an outside source cannot access their data. This will be vital to the project as this is one of the main points. Regarding the third standard, we will need to test our algorithms regarding partitioning and drone flight paths. This will need to happen to ensure the proper outcomes for our users.

IEEE 1471 (Software Architecture Standard) is crucial because:

- The project involves complex real-time interactions between frontend and backend
- Multiple components need to interact (UI, backend algorithms, external API calls)
- The system needs to be scalable to handle multiple drone operations
- Architecture documentation will be essential for maintaining the system

ISO/IEC 27000 (Information Security) is relevant because:

- The requirements explicitly mention security concerns about attackers
- Each session needs to be unique and secure
- The system handles real-world location data that must be protected

3 Project Plan

3.1 Project Management/Tracking Procedures

We will adopt an Agile development methodology with 2-week sprints for this project. This choice is justified by:

- The need for frequent user feedback on UI/UX elements
- Complex requirements that may need refinement through iterations
- The ability to deliver incremental functionality
- The need to adapt to changing requirements as we better understand drone behavior

For this project, we will utilize an agile development methodology with 2 week sprints. We decided to do this because we will need frequent user feedback on UI/UX elements. We also have complex requirements that may need refinement through several feedback iterations. As well as this we will need the ability to deliver incremental functionality. With all the changing requirements we will be having as well, this will be necessary to utilize.

Project tracking will utilize the following tools:

- GitHub: Source code version control and project documentation
- Jira: Agile project management, sprint planning, and task tracking
- Slack: Team communication and integration with GitHub/Jira
- Discord: Daily standups and team meetings
- Git Flow: Branch management strategy for feature development

As for project tracking we will be using GitLab, Discord, and Google Docs for this project. GitLab will be the repository for our code and version control that we can see over time. Discord will be our primary communications tool to discuss changes and functionality in an easy to use manner. Google Docs is where we keep our documents, idea pools, and articles to read up on. It is a very simple place to keep and manage all these things.

3.2 Task Decomposition

Table 1 below shows our task decomposition for semester 1 and semester 2 of senior design.

Task #	Planned Completion Date	Task Description (Frontend)	Task Description (Backend)
1	10/9/24	Setup React project structure	Setup Python server
2	10/24/24	Implement map visualization component	Implement session management
3	11/01/24	Create drone control interface	Develop map partitioning algorithm

4	12/10/24	Develop no-fly zone input system	Create drone path calculation system
5	01/30/25	Build real-time event visualization	Implement pathfinding algorithms
6	02/20/25	Implement algorithm selection interface	Build event response prioritization
7	03/6/25	Create simulation controls (start/stop)	Build real-time event processing
8	03/21/25	Develop navigation components	Implement external API integration
9	04/10/25	Performance testing	Performance testing
10	04/24/25	User acceptance testing	User acceptance testing
11	05/01/25	API documentation	Deployment pipeline setup

Table 1: Task Decomposition

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Frontend Milestones

- Map visualization loads in < 2 seconds: Sprint 2
- UI responds to user input in < 100ms: Sprint 3
- Real-time updates achieve 60fps: Sprint 4
- 95% test coverage: Sprint 5

Backend Milestones

- The server handles 100 concurrent users: Sprint 3
- Path calculations complete in < 500ms: Sprint 4
- Area partitioning completes in < 1 second: Sprint 5
- API response time < 200ms: Sprint 6

Algorithm Milestones

- Partitioning algorithm optimality within 90%: Sprint 4
- Pathfinding completion in < 300ms: Sprint 5
- Collision avoidance accuracy 99.9%: Sprint 6

- Event response time < 1 second: Sprint 7

Testing Milestones

- Unit test coverage > 80%: Sprint 5
- Integration test coverage > 70%: Sprint 6
- Load testing supports 1000 requests/second: Sprint 7
- Security penetration testing passed: Sprint 8

3.4 Project Timeline/Schedule

Figure 1 shows our Gantt Chart depicting our timeline/schedule which includes sprints and goals for each sprint.

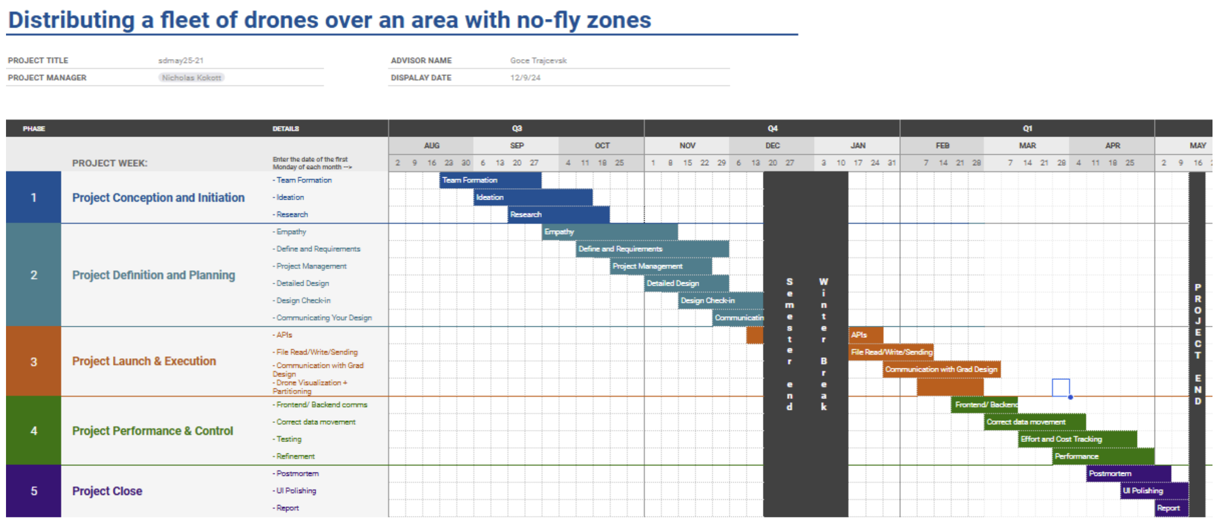


Figure 1: Gantt Chart

Sprint 1-2 (Weeks 1-4):

- Form team
- Understand the idea
- Research potential technologies

Sprint 3-4 (Weeks 5-8):

- Define requirements
- Figure out testing, goals, and technologies
- Begin prototype design

Sprint 5-6 (Weeks 9-12):

- Develop the UI
- Figure out how backend should look

Sprint 7-8 (Weeks 13-16):

- Continue UI development
- Begin on backend

Sprint 9-10 (Weeks 17-20):

- Make partitioning algorithm
- Display partitioning to frontend
- Add event management

Sprint 11-12 (Weeks 21-24):

- Add in pathing
- Test all units, interfaces, systems
- Finish out the reporting

3.5 Risks and Risk Management/Mitigation

A. High Probability Risks (P > 0.5)
   a. Real-time performance issues (P=0.7)
      i. Mitigation: Implement WebSocket for real-time updates
      ii. Fallback: Reduce update frequency
   b. Algorithm scalability problems (P=0.6)
      i. Mitigation: Implement caching and optimization
      ii. Fallback: Limit maximum area size
   c. Browser compatibility issues (P=0.6)
      i. Mitigation: Use polyfills and progressive enhancement
      ii. Fallback: Support the latest two versions of major browsers

B. High Severity Risks
   a. Security vulnerabilities
      i. Mitigation: Regular security audits
      ii. Implementation of security best practices
   b. Data Accuracy
      i. Mitigation: Implement validation layers
      ii. Regular calibration checks

3.6 Personnel Effort Requirements

Table 2 below refers to the individual contribution and total time spent for completing the design portion of our project.

Task #	Estimated Completion Time (hours)
Frontend Initialization	18 (3 hours * 3 people)
Backend Initialization	18 (3 hours * 3 people)
Design and User Interactivity design	24 (8 hours * 3 people)
Algorithm and communications	24 (8 hours * 3 people)
Pipeline and continued workflow	30 (5 hours * 6 people)

Completion of design document & presentation	558 (93 hours * 6 people)
Testing	12 (2 hours * 6 people)
Final Check	12 (2 hours * 6 people)
Finishing touches on presentation	18 (3 hours * 6 people)

Table 2: Individual Contribution

Total: 720 (120 hours* 6 people)

3.7 Other Resource Requirements

Develop Resources

- CI/CD pipeline tools
- Development workstations
- Testing environments

Software Resources

- IDE licenses
- Mapping service API credits
- Testing framework licenses
- Monitoring tool subscriptions

Development Tools

- Jira licenses
- GitHub Enterprise
- Code analysis tools
- Performance monitoring tools

4 Design

4.1 Design Context

4.1.1 Broader Context

The project primarily targets emergency response agencies, delivery services, and search-and-rescue organizations. These stakeholders need systems to optimize the deployment of drone fleets for efficient surveillance and rapid response over regions with no-fly zones. Indirectly affected communities include urban residents or government authorities responsible for managing no-fly zones. Additionally, industries reliant on drone technology for monitoring critical infrastructure or agricultural areas may benefit. This project addresses the societal need for efficient and timely emergency response, improved delivery services, or any form of drone deployment. It aims to minimize delays in drone operations for any of the listed groups. Table 3 refers to the public health, safety, welfare, global, cultural, social, environmental, and economic impacts or implications of our proposed design.

Area	**Description**	**Examples**
Public health, safety, and welfare	The project improves the response time of drones to emergencies, reducing risks to human life and property.	Reducing delays in search-and-rescue operations. Enhancing public safety in disaster-prone areas through rapid response. Decreasing safety risks by avoiding drone collisions in no-fly zones.
Global, cultural, and social	The design reflects respect for cultural practices by integrating with local regulations, such as avoiding flight paths over sensitive areas.	Adhering to airspace regulations in proximity to airports or military zones.
Environmental	The system optimizes drone flight paths to reduce energy consumption, minimizing the environmental impact.	Decreasing unnecessary flight distances for drones.
Economic	The project aims to deliver a cost-efficient system that benefits emergency agencies and industries without high overhead costs.	Reducing operational costs for drone fleets by optimizing deployments. Creating economic opportunities in drone technology for underserved regions.

Table 3: Broader Context

4.1.2 Prior Work/Solutions

Several systems exist for drone flight optimization, but few integrate no-fly zones and focus on minimizing average response time. Research on partitioning a map for drone management has been conducted, but these do not typically consider no-fly obstacles in their implementation. Some of the applications also do not allow consumers to map their own no-fly zones. This could lead to issues especially if a building or tree is not properly mapped in the flight path. Therefore, our application could add this feature for users being able to map their own no fly zones. The difference between our application and others is that our application will allow a user to input a number of drones that will then be used to partition an area in sections to reduce the response time of each drone in that area. Most applications similar to this project do not have this feature; therefore, we can take advantage of this opportunity gap to give users this ability.

Target Solution Comparison:

- **Pros:**
  - Direct integration of no-fly zones into partitioning algorithms.
  - Visualization and interactive user interface to aid decision-making.
  - Minimized response times for practical scenarios.
- **Cons:**
  - Dependence on high-quality obstacle data.
  - Computational overhead for real-time partitioning with large datasets

Some relevant products similar to our own include FlytBase [8] and DroneDeploy [9]. FlytBase offers drone management and automation features like no-fly zone visualization, planned flight paths, and scheduled missions. It integrates with docking stations for autonomous recharging and tracking, making it ideal for reconnaissance and mapping. Strengths include autonomy and visualization, but it lacks manual override options and has issues like low video resolution and slow customer support. DroneDeploy is an app for capturing site data manually or autonomously, supporting mapping, modeling, marketing, and inspections. It streamlines tasks like LAANC airspace authorization and uploading up to 10,000 images at once. While praised for ease of use and reporting compliance with surveying standards, it has drawbacks like high costs, slow processing, and the inability to add custom no-fly zones to flight paths.

4.1.3 Technical Complexity

Subsystems:

1. **Partitioning Algorithm Implementation:**
   - **Principle:** Computational geometry and graph theory.
   - **Complexity:** Handling irregular shapes of no-fly zones and ensuring optimal partitions.
2. **Drone Response Simulation:**
   - **Principle:** Planning flight paths.
   - **Complexity:** Simulating flight paths that respect partition boundaries and no-fly zones.
3. **UI/Visualization:**
   - **Principle:** Interactive graphics and geospatial data visualization.
   - **Complexity:** Rendering real-time feedback from backend solutions.
4. **Backend Integration:**
   - **Principle:** Database management with PostgreSQL/PostGIS.
   - **Complexity:** Efficiently handling geospatial queries and serving data to the frontend.

Challenging Requirements:

- Real-time response simulations for user-selected locations.
- Balancing computational efficiency with accuracy in obstacle-aware partitioning.
- Ensuring the system operates reliably under variable drone fleet sizes and geo-area complexities.

By addressing these aspects, the project meets and exceeds industry standards in obstacle-aware drone deployment systems.

4.2 Design Exploration

4.2.1 Design Decisions

Our design must consider technical constraints when designing the frontend and backend from scratch. We will need to determine the framework that can be utilized on the frontend so our users can have the best possible experience. We also need to decide what languages and technologies can be used on the backend to store and maintain state data for our users. This is crucial to be able to be sent back and forth between our computational geometry and pathfinding code. On top of this we need to consider the way we want to transfer data from the backend to the frontend. Different data types have different speeds, so finding the optimal data transfer method will be crucial to save time for the users.

4.2.2 Ideation

For at least one design decision, describe how you ideated or identified potential options (e.g., lotus blossom technique). Describe at least five options that you considered.

For our frontend design, there were several parameters that we considered that can be seen below:

- **Response time** - minimize communication time between the front and the backend
  - Framework has capability to use fast sockets
  - Framework has the ability to use multiple types of data transfer
- **Ease of use** - minimize time taken to develop other components that could just be easily implemented and reused later in the development
  - Pop-ups
  - Display Boxes
  - API displays
- **Quality of design** - we want the users to be able to see what is going on and understand the front facing UI that they can interact with
  - Easy to notice text boxes
  - Easy to use buttons
  - Easy to use mapping
- **Data transfer capabilities** - the design should be able to take in multiple data types that the user gives in and be able to push it correctly to the backend
  - Take in polygons via data sets
  - Take in polygons via plotting on the map
  - Take in numerical data
  - Be able to make this easily into JSON

- **Backend compatibility** - some frontend frameworks only work well with certain languages and backend frameworks
    - Works with many different languages and different frameworks
    - Easily communicate between them

4.2.3 Decision-Making and Trade-Off

When we decide on a frontend framework, our main criterion will be how easily data can be transferred and how well it interacts with our chosen backend and the languages used. We must consider how this data can be visualized to the front-facing user, as well as how well it can be transferred into the backend and be changed for the user to use easily and efficiently. Currently we have not chosen an option, but are leaning towards React+Vite as this has great ease in almost all of the areas described in 4.2.2.

4.3 Proposed Design

4.3.1 Overview

A website will be available to the general public for effective and efficient partitioning and flight path mapping of densely populated airspace. These *No-Fly Zones* cause a range of issues for standard operators of fleets of drones, i.e., delivery services, public safety officers, and geo-mapping services. This process will be completed by having the user provide initial information about the project scope for example, the number of drones in use and where the drones will be located geographically. This information is then passed from the website to our server, where a specially developed algorithm will partition the operation zone, minimizing the worst-case time scenario for each drone's possible target points, and then send this information back to the website to allow the user to visualize the operation. Finally, once a target destination is set, the server will run another algorithm to find the shortest possible route to the target location. This will all be combined into an animation on the website where the user can see in real-time how the drones are moving about the operation space.

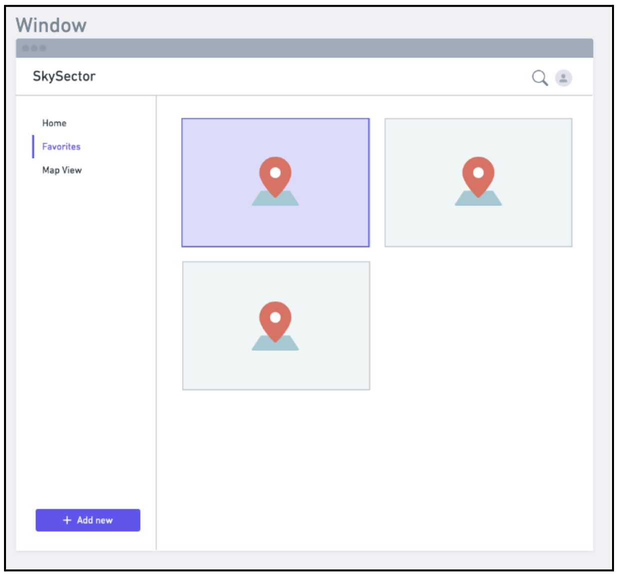
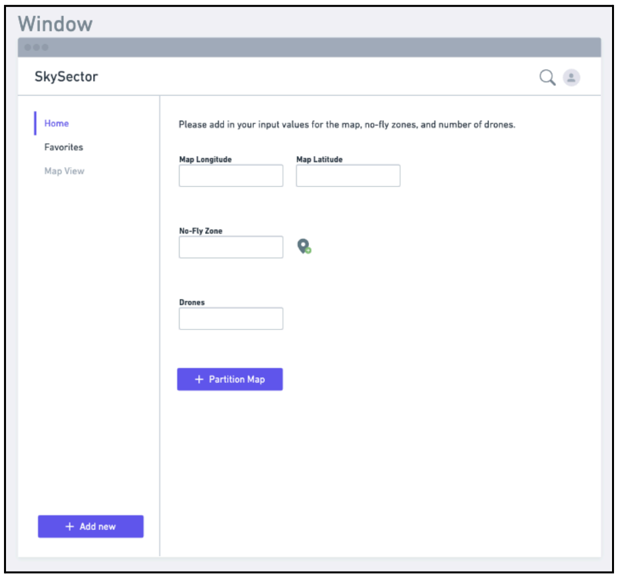


Figure 2: Frontend Design (Homepage)



Please add in your input values for the map, no-fly zones, and number of drones.

Map Longitude

Map Latitude

No-Fly Zone

Drones

Figure 3: Frontend Design (Input Selection)

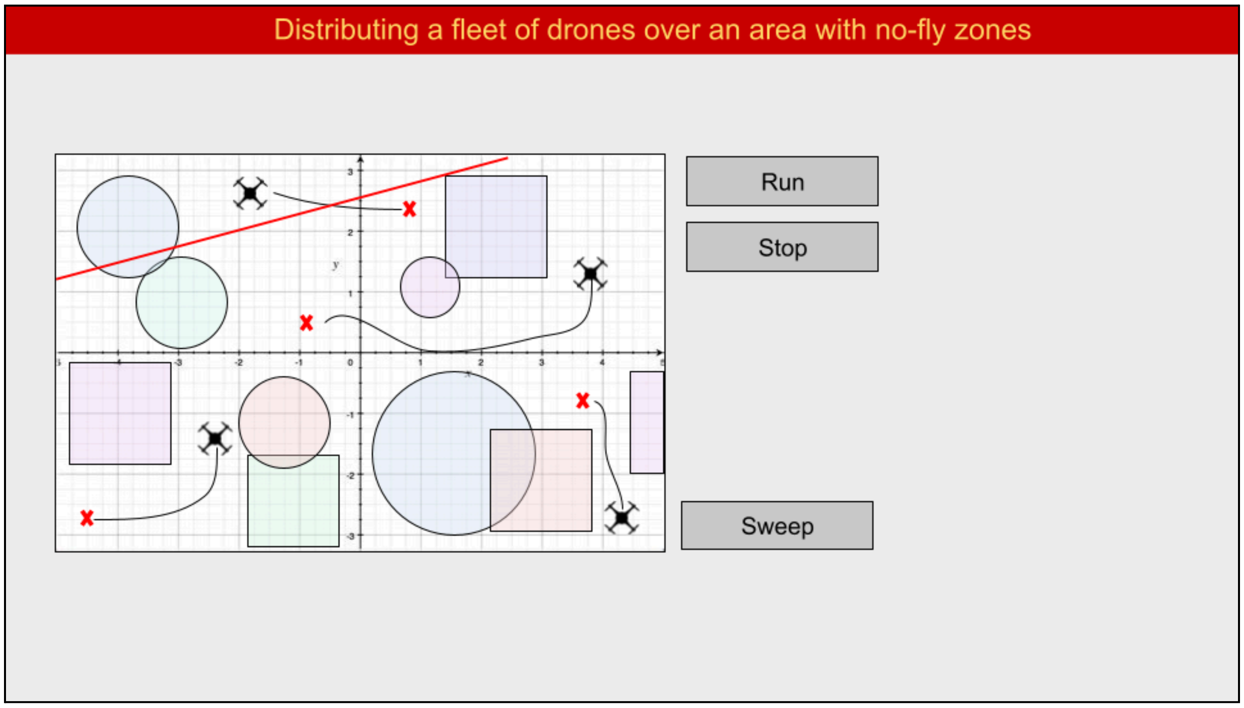

Distributing a fleet of drones over an area with no-fly zones
Run
Stop
Sweep

Figure 4: Frontend Design (Map View)

4.3.2 Detailed Design and Visual(s)

This project is a web application enabling users to input parameters such as area boundaries, no-fly zones, and event locations to observe an automated fleet of drones responding to events within a designated environment. Upon receiving user inputs, the application divides the area into smaller, partitioned regions, each assigned to a specific drone, to minimize response times to any critical event. The application comprises several interconnected components: the User Interface (UI), the Backend Server, a Database, an External Pathfinding API, a Partitioning Algorithm, and a Real-Time Data Processor. Each component has a distinct role in the system architecture.

The UI is the primary point of interaction, allowing users to enter relevant data, control simulation settings, and visualize real-time drone movement. It is built with technologies like HTML, CSS, and JavaScript, leveraging frameworks like React and Vue for dynamic rendering. The UI displays partitioned areas, event points, and drone positions in real time, providing users with a clear and interactive experience. Figures 2, 3, and 4 respectively illustrate the homepage, input selection, and map view. These mockups of our design were created to help design the frontend or UI of our application.

The Backend Server, developed using Python (with frameworks like FastAPI or Django) and PostgreSQL, handles data processing and storage. It receives data from the UI, manages user sessions, and controls algorithm execution. The backend also integrates with an External Pathfinding API, which calculates optimized routes for each drone, ensuring they avoid no-fly zones while covering their designated regions. The results from the API are then fed into the Real-Time Data Processor.

The Partitioning Algorithm, implemented in Python, divides the area into distinct regions based on the number of drones and user-defined no-fly zones. This algorithm utilizes computational geometry libraries to create efficient, non-overlapping regions, with minimal computation time to support real-time requirements. Once regions are determined, the Real-Time Data Processor continuously updates drone positions and routes as events are introduced or modified. The processor employs WebSocket technology to push updates to the UI, enabling seamless visualization of drone responses.

These components work together to ensure an efficient, responsive system that meets real-time visualization standards. In the final architecture as shown in Figure 5, the UI communicates with the Backend Server, which connects to the Database, Partitioning Algorithm, and Pathfinding API. The Real-Time Data Processor bridges the backend and frontend, maintaining continuous updates and event-driven responses, delivering a smooth, real-time experience for users observing automated drone fleet operations. This high-level overview and subsystem descriptions allow peer engineers to understand the project’s architecture and replication requirements clearly.

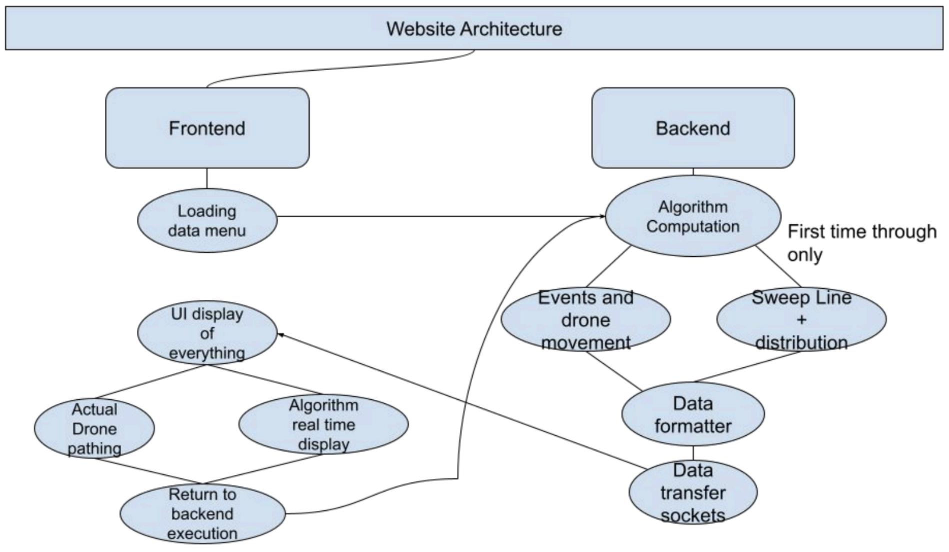


Figure 5: Global Architecture

4.3.3 Functionality

In the real world a user will firstly open our page to see a few different input prompts. As shown in Figure 6, the user would enter the number of drones, the map location, and the necessary no-fly zones. From here, the system would take all this data and send it to the backend to partition the map accordingly based on the no-fly regions so the drones could navigate in equal time. From here the drones will be placed, and partitions will be drawn onto the map. Now the user can both type coordinates for events to occur as well as place events on the map for the drones to respond to. Depending on the locations of the events, certain drones will fly to the events in real time for the user to see.

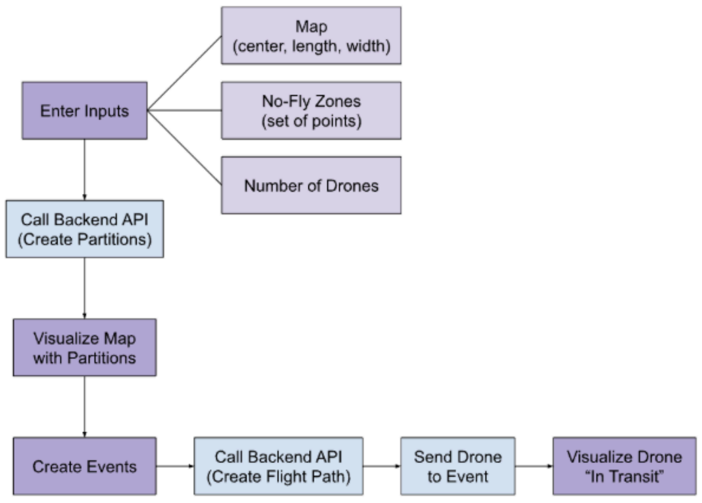

Map (center, length, width)
Enter Inputs
No-Fly Zones (set of points)
Number of Drones
Call Backend API (Create Partitions)
Visualize Map with Partitions
Create Events
Call Backend API (Create Flight Path)
Send Drone to Event
Visualize Drone "In Transit"

Figure 6: Functionality

4.3.4 Areas of Concern and Development

Our main concerns for the project's design focus on scalability, real-time processing, and security. First, we must ensure that the solution can scale effectively as more drones and complex parameters are added, such as diverse terrain, varying weather conditions, and larger no-fly zones. This requires a system architecture that can handle a range of drone counts and increasingly complex partitioning needs without significant loss in performance. Second, achieving reliable real-time processing is critical for effective visualization and rapid drone response. Real-time updates are essential for the application to support dynamic event handling, but they also introduce potential latency challenges as the dataset grows or pathfinding complexity increases. Finally, security is a priority as the system processes sensitive user-inputted geographic and event data. We need robust security measures to prevent unauthorized access and ensure each user's session is isolated and protected from data breaches.

To address these concerns, we will follow agile development in the next semester, implementing features iteratively and incorporating feedback from testing in various simulated scenarios. This approach will allow us to evaluate system performance under different scales, refine real-time data handling, and enforce security protocols effectively. By simulating conditions such as increased drone fleets, complex no-fly zones, and security breach attempts, we can identify and address any limitations early on, ensuring our system remains scalable, responsive, and secure.

4.4 Technology Considerations

Our project uses a mix of frontend, backend, database, and computational technologies to build a robust, real-time drone visualization system. Here's an overview of each technology choice, along with strengths, weaknesses, and trade-offs made to balance performance, scalability, and security.

1. Frontend: JavaScript with React (or Vue)

Technology Choice: We selected JavaScript, specifically frameworks like React or Vue, to build an interactive frontend that can handle complex, real-time data visualizations.

- **Strengths:** React and Vue are well-suited for real-time interactivity and state management, allowing us to update the UI as drone positions and events change efficiently. Both frameworks are widely used, well-documented, and supported by large communities, which helps troubleshoot and maintain code quality.
- **Weaknesses:** As with any frontend framework, the responsiveness may degrade if there are too many simultaneous updates or if the dataset grows large, potentially leading to slow rendering.
- **Trade-offs:** We chose JavaScript frameworks for their speed in development and interactivity. However, with increased data loads, we may face trade-offs in UI responsiveness.
- **Design Alternatives:** Alternatives like Angular could provide stronger structure and scalability but have a steeper learning curve. Another option could be WebGL for more efficient handling of complex animations, but it requires more specialized knowledge and would increase development time.

2. Backend: Python with FastAPI (or Django)

Technology Choice: Python, paired with FastAPI or Django, was chosen for its simplicity, extensive library support, and compatibility with computational tasks like partitioning and pathfinding.

- **Strengths:** Python's readability and extensive library ecosystem (NumPy, SciPy) support complex calculations required for partitioning algorithms and data processing. FastAPI offers high performance with async capabilities, critical for handling concurrent user sessions and requests, while Django provides more built-in features and a robust structure for larger applications.
- **Weaknesses:** Python could be faster in execution speed than languages like Go or C++. This could lead to delays in real-time operations, especially with larger datasets.
- **Trade-offs:** While Python may not be the fastest choice, its ease of use and large community support outweigh the need for optimizing real-time processing through faster languages.
- **Design Alternatives:** Node.js could be used for backend development with the advantage of a single language for both frontend and backend. However, it lacks Python's computational power, essential for our project's algorithmic needs.

3. Database: PostgreSQL

Technology Choice: We opted for PostgreSQL as the primary database for session data and user inputs.

- **Strengths:** PostgreSQL is reliable, ACID-compliant, and supports complex queries and indexing, making it ideal for managing and storing structured data securely. Its support for JSON is also helpful for handling flexible data types.
- **Weaknesses:** PostgreSQL may not handle high write operations or extremely large datasets as efficiently as NoSQL databases, which could become an issue if we scale to larger drone fleets and more extensive input data.
- **Trade-offs:** We selected PostgreSQL for its balance of robustness, relational data capabilities, and moderate scalability. Although it may not handle high-speed data ingestion and some NoSQL options, it provides the security and structure we need.
- **Design Alternatives:** A NoSQL database like MongoDB could allow for faster scaling and unstructured data storage, but it sacrifices the relational structure required for user sessions and partitioned areas. Redis or a time-series database could be introduced for high-speed data processing for high-traffic situations.

4. Partitioning Algorithm and Computational Libraries: Python with NumPy and SciPy

Technology Choice: We are using computational libraries in Python, including NumPy and SciPy, for partitioning regions and processing data to calculate efficient drone paths.

- **Strengths**: These libraries are optimized for numerical computations and offer powerful tools for mathematical operations, making them suitable for partitioning and pathfinding tasks.
- **Weaknesses:** While suitable for smaller datasets, Python's single-threaded nature and interpreted code may lead to slower computation times for extensive datasets, which could challenge real-time visualization requirements.
- **Trade-offs:** Python's library support and ease of use for computational tasks make it a strong candidate despite potential performance issues. We rely on efficient partitioning algorithms to mitigate this, optimizing code to keep processing times within real-time constraints.
- **Design Alternatives:** Implementing computational tasks in C++ or using specialized libraries like Boost or OpenCV could reduce processing time but would increase development complexity and may limit flexibility in algorithm testing and adjustments.

5. Real-Time Processing: WebSocket

Technology Choice: WebSocket manages real-time data transfer between the back and frontend, enabling continuous updates as drones move and respond to events.

- **Strengths:** WebSocket provides full-duplex communication, allowing for persistent connections essential for real-time applications. This ensures the application responds immediately to event updates and drone movements.
- **Weaknesses:** WebSocket connections can be resource-intensive, particularly under high loads. If many users are connected simultaneously, server strain could increase significantly.
- **Trade-offs:** WebSocket ensures low-latency updates, but robust server infrastructure is required to handle potential high loads.
- **Design Alternatives:** Alternatives like Server-Sent Events (SSE) could be simpler to implement for lightweight data streaming. However, they are not bi-directional and may not be as responsive for real-time interaction. Alternatively, using REST for periodic updates would simplify server load but lack the responsiveness needed for real-time visualization.

6. External Pathfinding API

Technology Choice: We rely on an external pathfinding API to calculate optimized flight paths that avoid no-fly zones and minimize response times.

- **Strengths:** The API reduces our internal workload for pathfinding and provides specialized algorithms optimized for handling geographic constraints.
- **Weaknesses:** Dependence on an external API could lead to latency, especially if the API needs to handle requests quickly enough for real-time application needs. It also introduces potential security concerns with third-party data transmission.
- **Trade-offs:** The API offloads complex calculations, but reliance on a third-party service risks response-time delays. We mitigate this by caching routes when possible.
- **Design Alternatives:** We could implement custom pathfinding algorithms internally to reduce reliance on third-party services. However, this would increase development complexity, and implementing high-performance algorithms may require additional expertise.

By balancing the strengths and limitations of each technology, we aim to create a scalable, secure, and responsive solution. Through agile development, we will test these components under varied scenarios to ensure they meet project requirements and deliver an optimized experience.

4.5 Design Analysis

As of right now, we have implemented and built a good portion of the frontend. We cannot make the backend at this time due to the fact that the PhD student making the pathfinding algorithms is not complete with his work, so we cannot process the data that users can give. It looks to be working great so far however, the user has a great way to input data as of right now, and should be easily parsable to be sent to the backend when we get to that point. For future implementation we want to develop the backend and get that data to be processed so our users can see how the drones should be flying within their respective partitions.

5 Testing

In this section we will discuss the testing methodology that will be applied for the development of this project. There will be tests for both the functional and non-functional requirements, as mentioned previously in section 1. This will be done to verify that the functional requirements are working as expected, and that the non-functional requirements are meeting the needs of our clients/advisor. Luckily with our product there are no cost related requirements as we are just developing a visualization system. When the components of our project are implemented, we will run them through unit, interface, integration, system, and regression tests. After these tests are completed we will discuss the results with our advisor to find areas of improvement for further development. Each of the subsections below will outline each of the particular tests to be performed throughout our development cycles.

5.1 Unit Testing

Unit testing will consist of testing the individual units that make up the project. When we refer to testing units, we are referring to the GUI components, classes, and methods that we develop for this project. On the front end of things, since we are utilizing React+Vite we are able to utilize the Jest framework in order to test individual components to ensure they are doing their expected job before integrating them to our overall project. As for the Python in the backend, we will need to test each individual function, as well as ensure that they work alongside our MapBox API and our Grad student's API.

5.2 Interface Testing

The interface testing will be tested through the combination of multiple units. These combinations will ensure that the interface of the application will be able to be successfully implemented. Our general interfaces are:

1. The grad students algorithm (GSA)
2. Our python backend
3. Our UI to display everything
4. The MapBox API

The backend will need to be able to discuss with the GSA in order to be able to display the proper drone data. This will in turn require us to write functions that take what comes back from the GSA and be able to feed it into the MapBox API to display drones moving to our users. Of course this will need to be able to be shown on our frontend in order to display the map, with the algorithm's working data, as well as the drones' movement and response to events.

5.3 Integration Testing

Integration testing will be completed by breaking the system into several different major functionalities and then testing each functionality separately from each other. This will ensure that all of these functionalities are performing as anticipated and will be able to send the correct data between one another. Some of the critical paths would be tracing through certain functions in the backend to ensure that data is being calculated correctly, ensuring that the data we receive through API operations is what is as anticipated, and ensuring that data being sent to the frontend is being used correctly and displaying what is needed for the user.

5.4 System Testing

System testing will be completed by running multiple integration tests together to verify that throughout an application run, we are seeing expected results at each step and that data is being properly displayed to the user. This is vital to the system, ensuring that if the user selects a location, this will go through the API and the response will go back to the UI and into the backend to compute data which will be computed and rerouted back to the UI. In order to perfect this, we will chain together the corresponding integration, interface, and unit tests. The critical requirements will be verified by utilizing all of these system tests.

5.5 Regression Testing

Regression testing will be done by running all the previously existing unit, interface, and system tests to ensure that nothing has changed from our previously expected results. On top of this we will be manually verifying the system functionality in order to determine that every part of the system is still functional for our users. Critical requirements will also be checked to ensure that there are no changes from the new implementations. This includes: UI display, path generation for the drones, partitioning algorithms, as well as API communications.

5.6 Acceptance Testing

Acceptance testing will be done by analyzing both the functional and non-functional requirements that we created and ensuring they are not being violated. For the functional requirements we will be creating a set of use cases for functions within the application. These use cases will cover all possible scenarios that a user could do when utilizing the application, and will be followed as test routes within to ensure anticipated performance. As for the non-functional testing, we will be mostly demonstrating the application to our advisor to ensure that the project requirements are being met. For each major development within the project we will demonstrate it to our advisor and listen for feedback to improve upon, come the following meeting. Once our advisor has acknowledged that the application is meeting the requirements, we will be satisfied with the development.

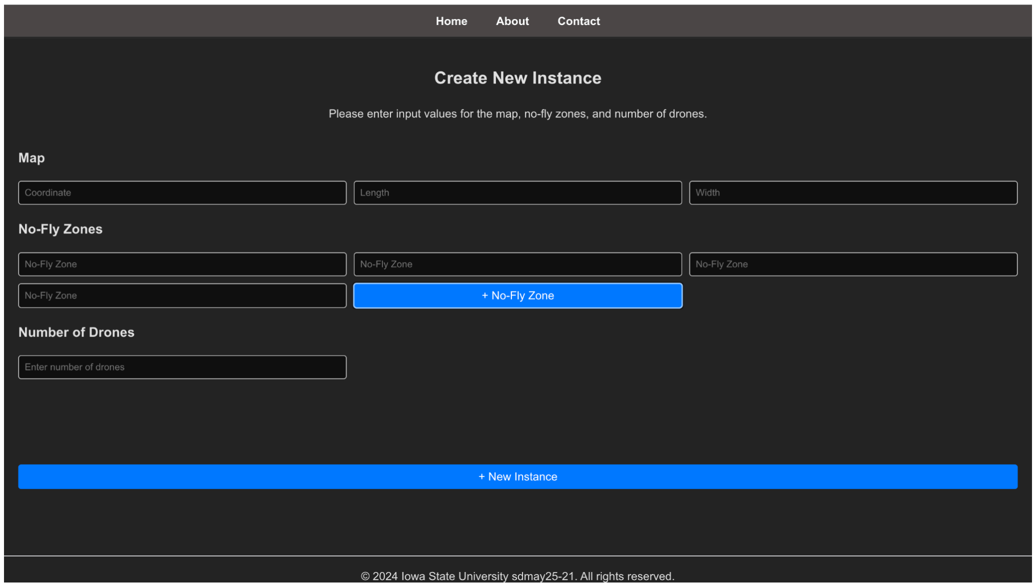
5.7 Results

As the project continues to be developed, it will be tested thoroughly utilizing the testing techniques described in the above sections. There will be many cases that are created to ensure that all written code is tested and produces the desired results. Based on the results that come back, performance will be evaluated and changes will be made as necessary. This will help motivate all goals that were created for this project.

6 Implementation

From our plans in section 3.3, the project is continuing along just as the plan describes. There is a great foundation of the frontend as shown in Figure 7 and 8 and a good idea of what the overall system architecture will look like. The team is still wrapping up the design phase of the process, and will dive heavier into the implementation in the second semester of class.

The implementation will consist of a code base and making visualizations with the map. Our team will also be developing data transfer between the frontend and the backend. Both of these processes will help get the implementation phase going smoothly for the second semester.



Create New Instance

Please enter input values for the map, no-fly zones, and number of drones.

Map

Coordinate | Length | Width

No-Fly Zones

No-Fly Zone | No-Fly Zone | No-Fly Zone

No-Fly Zone | + No-Fly Zone

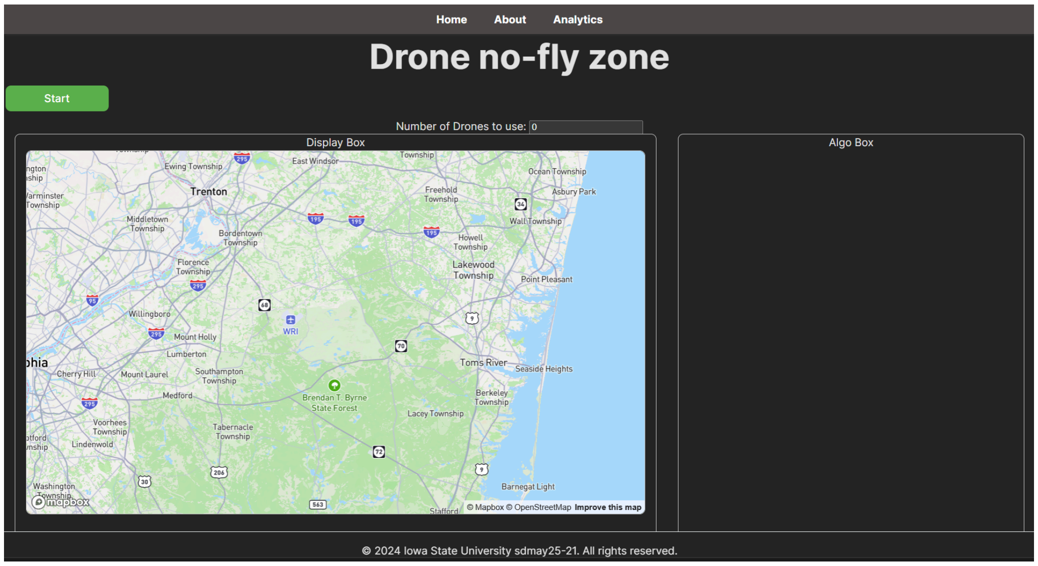
Number of Drones

Enter number of drones

+ New Instance

© 2024 Iowa State University sdmay25-21. All rights reserved.

Figure 7: Prototype (Input Page)



Drone no-fly zone

Number of Drones to use: 0

© 2024 Iowa State University sdmay25-21. All rights reserved.

Figure 8: Prototype (Map View)

7 Ethics and Professional Responsibility

This discussion is with respect to the paper by J. McCormack and colleagues titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416–424, 2012

7.1 Areas of Professional Responsibility/Codes of Ethics

We will be utilizing the SE code of ethics for this section of the document as shown in Table 4 below.

Area of Responsibility	Definition	NSPE Canon	Differences between NSPE and SE Code of Ethics
Work Competence	Perform work of high, quality, integrity, timelines, and professional competence.	Perform services only in areas of their competence. Avoid deceptive acts.	SE code says engineers need to make sure products are meeting the highest professional standards. NSPE does not define that engineers work only within their competences.
Financial Responsibility	Deliver products and services of realizable value at reasonable costs.	Act for each employer or client as faithful agents or trustees.	SE code says to act with the client and best public interest. NSPE says to act as the client or trustee.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	Issue public statements only in an object and truthful manner. Avoid deceptive acts.	SE code describes that engineers should act with integrity. The NSPE says instead that engineers should only speak objective truth.
Health, safety, and well-being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	SE code describes engineers to act with public interest. Whereas NSPE describes that engineers should hold health and safety of the public first and foremost.

Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.	SE code describes to use public interest with client and employees. NSPE says just to act as the client or trustee.
Sustainability	Protect the environment and natural resources locally and globally.		SE code says to act in the publics best interest, whereas the NSPE has nothing on sustainability.
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorable, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	SE code says engineers should advance society. NSPE says to do this with many with honor, responsibility, ethics, and to enhance the profession.

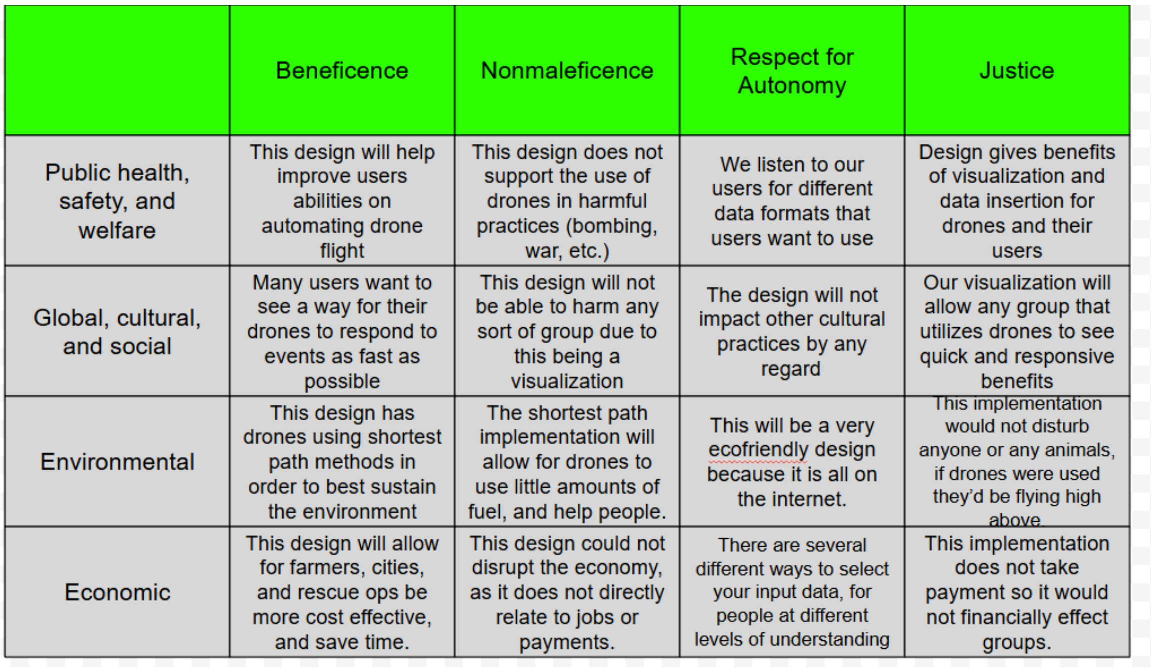
Table 4: Code of Ethics

Our team is doing very well in the communication honesty responsibility. All of us are very open with the way we communicate with each other, and are very honest about the ideas that we all come up with. There is no deception among us, and we are all able to work well together. Our advisor is very aware of where we are at, and how we plan to go about achieving our goals for this project.

We could likely improve the health, safety, and well-being responsibility. Currently we don’t really see how we are all doing mentally, just to see how we are all doing on our work.

Our team is having weekly meetings where we discuss many of these responsibilities working alongside the development of our project. This has been very beneficial to the advancement of our goals and sprints.

7.2 Four Principles



	Beneficence	Nonmaleficence	Respect for Autonomy	Justice
Public health, safety, and welfare	This design will help improve users abilities on automating drone flight	This design does not support the use of drones in harmful practices (bombing, war, etc.)	We listen to our users for different data formats that users want to use	Design gives benefits of visualization and data insertion for drones and their users
Global, cultural, and social	Many users want to see a way for their drones to respond to events as fast as possible	This design will not be able to harm any sort of group due to this being a visualization	The design will not impact other cultural practices by any regard	Our visualization will allow any group that utilizes drones to see quick and responsive benefits
Environmental	This design has drones using shortest path methods in order to best sustain the environment	The shortest path implementation will allow for drones to use little amounts of fuel, and help people.	This will be a very ecofriendly design because it is all on the internet.	This implementation would not disturb anyone or any animals, if drones were used they'd be flying high above
Economic	This design will allow for farmers, cities, and rescue ops be more cost effective, and save time.	This design could not disrupt the economy, as it does not directly relate to jobs or payments.	There are several different ways to select your input data, for people at different levels of understanding	This implementation does not take payment so it would not financially effect groups.

Table 5: Ethics Four Principles

Table 5 illustrates the four principles of ethics as it relates to our project. Economic beneficence is the most important broader context-principle pair within our project. This is due to the fact that we will help many different groups of people save time and money with our design. The way we will do this is by making our design as user friendly as possible. People will be able to see their potential drone fleet setups responding to events that they would have.

We are currently lacking in public health, safety, and welfare nonmaleficence. This is because we have not determined a way to prevent these drones from being used in a harmful manner. We may try to improve in this area by having people put in specific drone models they want to use for their fleets.

7.3 Virtues

1. Collaboration
   a. This is the ability to work well with others. In terms of sharing ideas, responsibilities, and resources to achieve common goals
   b. We support this value by organizing team meetings, listening to one another, and offering constructive feedback on design choices.
2. Respect
   a. This is the ability to value skills, contributions, and perspectives of all team members.

    b. We support this virtue by fostering an environment where all opinions are welcomed, allowing people to speak without interrupting, and resolving conflict in swift and efficient manners
3. Accountability
    a. This is the ability to take responsibility for your actions and decisions on the team.
    b. We support this virtue by setting clear expectations, checking in regularly, and focus on solutions rather than blaming people for things that may end up happening during development.

8 Closing Material

8.1 Conclusion

At this current stage of development, we have completed a frontend prototype that demonstrates how we want the website to look and feel for our users. The backend has very little work done on it as we are still awaiting the grad student's pathing and partitioning algorithms (if this takes too long we will begin working on this ourselves). Our goal is to have a fully functional backend and frontend communication that can allow the user to put in the required information they need, and then have the backend process that data to visualize what the user wants back on the frontend. We will do this by working closely as a team and communicating at a high level to ensure all of us are on the same page. For the upcoming semester we will be meeting twice a week to better coordinate and understand what all of us are working on to ensure a fulfilled deliverable.

8.2 References

[1] Bobby Sudekum. "Don't Fly Drones Here." *Medium*, Mapbox, 21 July 2014, blog.mapbox.com/dont-fly-drones-here-928dee4389e8. Accessed 4 Dec. 2024.

[2] G. Attenni, V. Arrigoni, N. Bartolini and G. Maselli, "Drone-Based Delivery Systems: A Survey on Route Planning," in IEEE Access, vol. 11, pp. 123476-123504, 2023, doi: 10.1109/ACCESS.2023.3329195. keywords: {Drones;Surveys;Job shop scheduling;Industries;Trajectory planning;Task analysis;Path planning;Product delivery;Urban areas;Drone delivery;drone route planning},

[3] Saeed H. Alsamhi, Ou Ma, Mohammad Samar Ansari, and Faris A. Almalki. 2019. Survey on collaborative smart drones and Internet of Things for improving smartness of smart cities. IEEE Access 7 (2019), 128125–128152. https: //doi.org/10.1109/ACCESS.2019.2934998

[4] Ambuj Kumar and Bilal Muhammad. 2018. On how Internet of Drones is going to revolutionise the technology application and business paradigms. In Proceedings of the 2018 21st International Symposium on Wireless Personal Multimedia Communications (WPMC'18). 405–410. https://doi.org/10.1109/WPMC.2018.8713052

[5] Jao Valente. 2014. Arial Coverage of Path Planning Applied to Mapping. Ph. D. dissertation. Polytechnic University of Madrid.

[6] Ouri Wolfson, Prabin Giri, Sushil Jajodia, and Goce Trajcevski. 2021. Geographic-region monitoring by drones in adversarial environments. In Proceedings of the 29th International Conference on Advances in Geographic Information Systems (SIGSPATIAL'21). ACM, New York, NY, 480–483. https://doi.org/10.1145/3474717.3484216

[7] Z. Qin, A. Li, C. Dong, H. Dai, and Z. Xu. 2019. Completion time minimization for multi-UAV information collection via trajectory planning. Sensors (Basel) 19, 18 (2019), 4032.

[8] "Enterprise Drone Autonomy Software Platform | FlytBase," *www.flytbase.com*. https://www.flytbase.com/

[9] "Drone & UAV Mapping Platform | DroneDeploy," *Dronedeploy.com*, 2019. https://www.dronedeploy.com/

8.3 Appendices

The algorithm is provided by our advisor, however it is not yet created. It will be linked here when it is completed.

9 Team

9.1 Team Members

1. Nicholas Kokott
2. Sam Russett
3. Everett Duffy
4. Melani Hodge
5. Kenneth Schueman
6. Cole Stuedeman

9.2 Required Skill Sets for Your Project

Frontend Development - TypeScript, JavaScript, React+Vite

Backend Development - Python, OOP development, Database Communication, Session Storing

Overlapping skills - Socket Communication, JSON

9.3 Skill Sets covered by the Team

For the frontend skills - Kenneth Schueman, Melani Hodge, Nicholas Kokott

For the backend skills - Kenneth Schueman, Nicholas Kokott, Everett Duffy, Samuel Russett

For the overlapping skills - Kenneth Schueman, Nicholas Kokott, Cole Stuedeman

Project Management Style Adopted by the team

For this project our team will be utilizing the agile development methodology, as we have all experienced it in the past, and feel that it would be beneficial to continue using it.

9.4 Project Management Style Adopted By The Team

For this project, we will be adopting an agile form of project management.

9.5 Initial Project Management Roles

Nicholas Kokott - Team Organizer

Melani Hodge - Frontend design/implementation

Cole Stuedeman - Testing

Everett Duffy - Component/Module Design

Ken Schueman - Advisor Communication and Frontend maintainer

Samuel Russett - Research Discovery and Testing

9.6 Team Contract

Team Name: Attack of the Drones (sdmay25-21)

Team Members:

1. Nicholas Kokott
2. Sam Russett
3. Everett Duffy
4. Melani Hodge
5. Kenneth Schueman
6. Cole Stuedeman

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
    a. Mondays at 6:10 at SICTR 0107
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    a. We will use the phone generally as more of us will see it on the fly rather than checking our emails, but if we need to involve our advisor, we will utilize either our face-to-face meeting or email if that is not close.
3. Decision-making policy (e.g., consensus, majority vote):
    a. We will be doing a majority vote for many of the decisions
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
    a. Nicholas Kokott will keep the meeting minutes; others will also take notes. However, these meeting minutes will be stored in our Senior Design Google Drive folder and will be accessible in the meeting minutes folder.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

    a. Everyone should arrive at least 5 minutes early to ensure we are on time for the professor. If anyone cannot make a meeting, they will email the professor and the team about what is happening. We will fill them in when we see them in class the following day.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
    a. We will equally contribute to the team assignments and complete them to the best of our abilities by the day before the deadline so that we can make modifications if needed. We will all communicate to ensure that this happens promptly and that we are all on the same page with our assignments.
3. Expected level of communication with other team members:
    a. We will communicate daily about what we are doing and contributing to the project. We will discuss what we have been researching and figuring out and what could be useful when designing the project.
4. Expected level of commitment to team decisions and tasks:
    a. As for team decisions, we will be all committed to discussing anything we are deciding on and ensuring that we come up with the best possible solution to any problems we face. When we start getting to individually assigned tasks, we will each be responsible for each task we are assigned and complete them by the given deadlines we set. Doing this will ensure we stay on task and can complete the project by the end of the year.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
    a. Nicholas Kokott will be responsible for the team organization
    b. Kenneth Schueman will be in charge of advisor communication
    c. Cole Stuedeman will be in charge of testing the design that we have been making.
    d. Everett Duffy will be responsible for the individual component design if needed.
    e. Samuel Russett will be in charge of research discovery and distribution.
    f. Melani Hodge will be in charge of algorithm design, ensuring they fit what we are given.
    g. We will all be responsible for movement towards development of the project.
2. Strategies for supporting and guiding the work of all team members:

    a. To support and guide the work of all team members, we will continually talk and contribute ideas to each other daily to keep our minds on the project. As well as this, if a member is struggling, we will have one or two members come in to assist and see what may be delaying the individual. If none of us can figure it out we will be asking the advisor what we can do to solve the problem.

3. Strategies for recognizing the contributions of all team members:
    a. This is very important, as all members should feel valued within the team. We will continually acknowledge each other's work and thank them for contributing to the team. We will also discuss our contributions with our advisor to demonstrate to him that we are all contributing.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
    a. Nicholas Kokott - brings embedded systems and cybersecurity experience to the team, virtual machines, and docker experience.
    b. Kenneth Schueman - brings lots of AI knowledge and profound programming experience to the team
    c. Everett Duffy - brings the computer engineering expertise on the team which will be vital when we start getting to the drones
    d. Cole Stuedeman - brings great coding experience and vast communication skills.
    e. Samuel Russett - brings fantastic app development skills and operating system understanding.
    f. Melani Hodge - brings great understanding of algorithms and design experience to the team
2. Strategies for encouraging and supporting contributions and ideas from all team members:
    a. In order to support and encourage contributions we will continuously discuss things that pop into our minds in order to see what might be the best solution. We are all aware that we don't know everything by any means and that we are all learning about computational geometry. So, we will push forward any new techniques we read and encourage others to read and learn about it as it might be incredibly beneficial for everyone to see.
3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)
    a. If a team member is having trouble contributing, they will just come forward and address the issue with the rest of the team. Everyone on

the team will be open to hearing what the issue is and be willing to change or help stop the obstruction that the individual is having at the time. If that does not work, we will have a team meeting to determine what we can do to fix the problem.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
    a. Our team goals for this semester are to research and plan for our project. We need to gain knowledge and understanding of background, formulas, and similar projects that have been done to develop a successful prototype. We plan to begin constructing our prototype towards the end of this semester.
2. Strategies for planning and assigning individual and teamwork:
    a. We will continuously analyze, see who does not have work to do or is close to completing their tasks, and give them something to work on. If there is a gap in the assignments or development, we will have them go look at research on the things we are curious about to understand better potential implementations for any problems that we are currently having.
3. Strategies for keeping on task:
    a. To stay on task, we have deadlines for the assignments already put in place. Outside of that we have already set up a Google calendar with due dates on our things or things our advisor has been giving us. This will gie us reminders on our laptops and other devices to better stay on track and ensure we do a great job.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    a. We will reach out to the individual who breaks the contract and encourage them to continue to push towards our collective goal. Also, we will let the advisor know that this person may begin to be a problem if they are not cooperating with us.
2. What will your team do if the infractions continue?
    a. If the infractions continue we will involve the instructors of the course and inform them of whatever this person is doing. This should be enough to stop them from breaking the contract continually and get the team back on track for the rest of the semester.

****

a) *I participated in formulating the standards, roles, and procedures as stated in this contract.*
b) *I understand that I am obligated to abide by these terms and conditions.*
c) *I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.*

1) Nicholas Kokott	DATE 9/12/24
2) Kenneth Schueman	DATE 9/12/24
3) Cole Stuedeman	DATE 9/12/24
4) Everett Duffy	DATE 9/12/24
5) Samuel Russett	DATE 9/12/24
6) Melani Hodge	DATE 9/12/24