GONZAGA UNIVERSITY
School of Engineering and Applied Science
Center for Engineering Design and Entrepreneurship

FINAL PROJECT REPORT
May 3rd, 2024

Wildfire Resource Simulator

Prepared by:

Kevin Dang

Claire Yegian

Drew Bogdan

Vincent Do

Kole Davis

1 Project Overview

1.1 Overview

The current wildfire management strategy in the United States faces scrutiny due to inefficiencies in aerial firefighting resource allocation. Our sponsor Fire Armada aims to address this issue by raising awareness and advocating for change. Wildfires are becoming increasingly frequent and severe, catalyzed by climate change and improper forest management, and aerial retardant drops play a vital role in suppression efforts. Large airtankers (LATs) and very large airtankers (VLATs) are crucial assets deployed from over one hundred strategically located tanker bases—stations for holding planes, loading fire retardant, and refueling—across the country. Our project centers on analyzing flight trajectories and tanker base usage to identify operational inefficiencies that support Fire Armada's goal to drive change by advocating for improved resource allocation and management. This change may entail increased government support for private contractors, fleet expansion, enhanced detection methods, or better base and aircraft utilization. By promoting awareness and reform, we aim to enhance wildfire management for public safety and environmental conservation.

Utilizing data-driven simulations and visual analytics, we highlight deficiencies in existing methods, focusing on LATs and VLATs due to their reliable data and capacity. Our simulator integrates wildfire data, flight trajectories of LATs and VLATs, and tanker base locations, which allows the user to interactively explore the way the US aerial firefighting system functions. Statistical analyses aim to uncover trends and inefficiencies, such as suboptimal base utilization and response times. We investigate whether increased drops or earlier responses reduce fire impact, and assess plane usage efficiency. By offering intractability and visual statistical analysis of the aerial firefighting system's efficiency, we aim to support Fire Armada in their endeavors to instigate change in the system.

1.2 Project Requirements & Deliverables

1.2.1 Data Retrieval, Cleaning, & Storage

Firefighting aircraft (LATs and VLATs), fire, and tanker base data is collected, cleaned, and organized into a database.

- Fire data: Including at least location (latitude and longitude), size of fire, time started, time burned, and date for fires in the US during the given time range
- Plane data: Including at least location (latitude and longitude, updated at most every few minutes for trajectory tracking), time and date, altitude, and tail number for flights of the 27 LATs and VLATs in the US during the given time range
- Tanker base data: Including at least location (latitude and longitude) for all tanker bases in the US.

Collected data is limited to the years 2020-2022 primarily because of costs associated with obtaining the aircraft flight data.

1.2.2 Interactive Simulator: Mapping Fire & Aircraft Data

The simulator uses a mapping tool to plot data about flight paths, fires, and tanker bases over time. It includes a time-scaling feature that allows the user to restrict and change the time range for which data is displayed. The map features required are as follows:

- Icons: The map has icons representing fires, tanker bases, and planes
- Trajectories: Plane trajectories are tracked with a line following the trajectory that is the same color as the plane. All of a given plane's flights and its icon are the same color
- Scalability: The user is able to zoom in and out to view all parts of the US at varying scales
- Time range interface: The user is able to select a time range and view all fires and flights during that range. The user is prevented from entering an invalid time range
- Icon hiding: The user is able to hide or show icons for planes, fires, and/or tanker bases
- Information boxes: The user is able to view additional information for fires, tanker bases, and planes by clicking on the icon to open up an information box

1.2.3 Statistical Investigation: Calculations & Dashboard Visualizations

The application dynamically queries the database and calculates the necessary attributes for statistics designed to investigate potential inefficiencies in plane use and tanker base distribution. Calculations are primarily generated by static scripts and their results are then loaded into the database. Statistics are visualized using a graphing tool that pulls data directly from the database.

The statistics may be grouped into the following categories:

- Preliminary Statistics
  - Number of airtankers deployed per US contractor
  - Number/percentage of fires sorted by discretized size (acres burned)
- Suppression Statistics
  - Number of drops on a given fire vs. suppression results
  - Time for first airtanker response to fire vs. suppression results
- Tanker Base Statistics
  - Most visited tanker bases in the last three years
  - Proximity of fires to tanker bases based on average distance
- Plane Statistics
  - Average flight hours per airtanker
  - Time spent grounded vs. flying per airtanker during fire season

2 Work Accomplished

2.1 Data Retrieval & Cleaning

The datasets used in this project (fire, airtanker, flight, and airtanker base) were collected from many sources that required separate operations before they could be inserted into the database. These cleaning operations were performed statically and must be repeated for the project once new data is added.

The fire data was sourced from the National Interagency Fire Center (NIFC), an organization that provides public, authoritative, and updated datasets on their website. The Wildland Fire Incident Locations dataset met our project needs with unique fire identifiers; coordinate locations; date-timestamps for discovery, containment, and other events; and amount of acres burned. The operations used to clean the fire data were:

- Converting the date-timestamps into one consistent format
- Dropping entries with invalid dates or duplicate unique IDs (due to repeated reporting)
- Sorting the data by discovery date
- Filtering the data to fires from before 2023

The LAT and VLAT data was sourced from a Schedule of Items provided by the US Forest Service in 2022. Relevant attributes like tail number, plane type, and contractor were manually recorded in a spreadsheet.

The flight data for identified tankers was sourced from the FlightAware AeroAPI, which was queried through the FlightData library. The list of airtanker tail numbers was used by the modules to retrieve flight identifiers, date-timestamps, coordinates, and other attributes to be added to formatted datasets representing flight information (overall details about a flight) and flight points (details about a flight at a given time). A configuration file specified the date range desired and cost threshold. The cost threshold limits the script from reaching a certain cost, and terminates the entire program and stops with the data gathered up to that point should it reach the threshold. This measure was implemented to avoid extraneous costs (see Section 4.3 for additional technical details about the FlightData library).

The airtanker base data was sourced from the National Wildfire Coordinating Group's (NWCG) Airtanker Base Directory published in 2018. Since this document is in PDF format, attempts were made to programmatically collect the necessary information using Python parser libraries; however, the format of pages varied greatly and a script solution was set aside in favor of manually recording the attributes in a spreadsheet. Additional cleaning was performed on the derived dataset due to reporting errors and missing values.

2.2 Database

This project used a MySQL database hosted on a remote server in development, and the database scripts have also been tested to be compatible with a SQLite local instance. The database contains 10 tables, with 6 of the tables containing datasets and 4 containing derived statistical information, and 3 views with dynamically generated statistics.

All tables have a provided schema and insert statement SQL file to be run directly on the database (the execution order matters due to foreign key constraints, see Section 4.2 for the database design). The insert statements are created through Python generation scripts, which share a general design. These scripts accept a cleaned dataset, load them using the Pandas library, and then perform formatting operations based on the given dataset and the corresponding table's requirements. For example, some tables allow for null values, so the associated scripts are designed to handle empty attributes in the loaded data.

2.3 Statistical Calculations

2.3.1 Plane Visiting

Since elements of three of the chosen statistics (plane drops on a fire, response time to a fire, and airtanker base visits) require detecting when an airtanker visits a location, we used an algorithm based on rounding coordinates. The latitude and longitude coordinates for flights, fires, and bases are all stored in decimal degrees in the database,Rounding to the 2nd decimal produces a coordinate grid of 1.1 square kilometers, and rounding to the 1st decimal creates a grid of 11.1 square kilometers. If two rounded coordinates were equal, then we considered them in the same grid area, indicating a plane visit. Plane drops used the smaller area since they had to be more precise (this may not be as applicable for larger fires, however the margin of error is too large for smaller ones), and airtanker base visits used the larger area due to airport sizes.

Additional work was done to determine whether an airtanker performed a drop on a given fire. We initially filtered to potential flights that could have responded to a fire based on if the timeframe of the flight overlapped with the fire's. For each fire, the potential flight candidates were checked to see if their coordinate grids matched to determine if a drop occurred (which was determined if a visit was made to the fire in the flight path).

2.3.2 Response Time

Using the results of plane drops on fires which mapped unique flights to fires, response time was calculated by finding the difference between the fire's discovery date and the flight's take off timestamp.

2.3.3 Time Spent in Flight & Grounded

This statistic uses an inverse calculation to determine the amount of time a plane spent on the ground by inverting the time spent in the air. With this calculation, we proceeded to calculate the time a plane spent in the air over or under the average flight time of a plane over the duration of a year.

2.3.4 Suppression

We based fire suppression results on the acres and time (hours) burned for a given fire. We used a logarithmic transformation to reduce the right skew of the data resulting in the following equation to calculate suppression result as a percentage:

$$\left(1 - \frac{\left(\frac{\log(x+1)}{\log(\text{MaxHoursBurned}+1)} + \frac{2(\log(y+1))}{\log(\text{MaxAcresBurned}+1)}\right)}{(3)}\right) \times 100$$

where x = *HoursBurned* and y = *AcresBurned* of a given fire.

We gave twice the weight to acres burned as to time burned because of our judgment that the acres burned is a more comprehensive metric of damage than time burned.

2.3.5 Proximity

In order to determine which tanker bases were generally closer to the fires they deployed planes to, we calculated the distance between each fire and its closest tanker base. We then averaged all the distances by tanker base.

2.4 Grafana Visualizations

Visualizations were developed using Grafana, an open-source analytics application that allowed us to connect and graph data directly from our database. Grafana does have embedding capabilities which would have made our visualizations dynamic, but we were not able to achieve embedding with the time we had. Embedding is a potential improvement for further development.

2.4.1 Preliminary Statistics

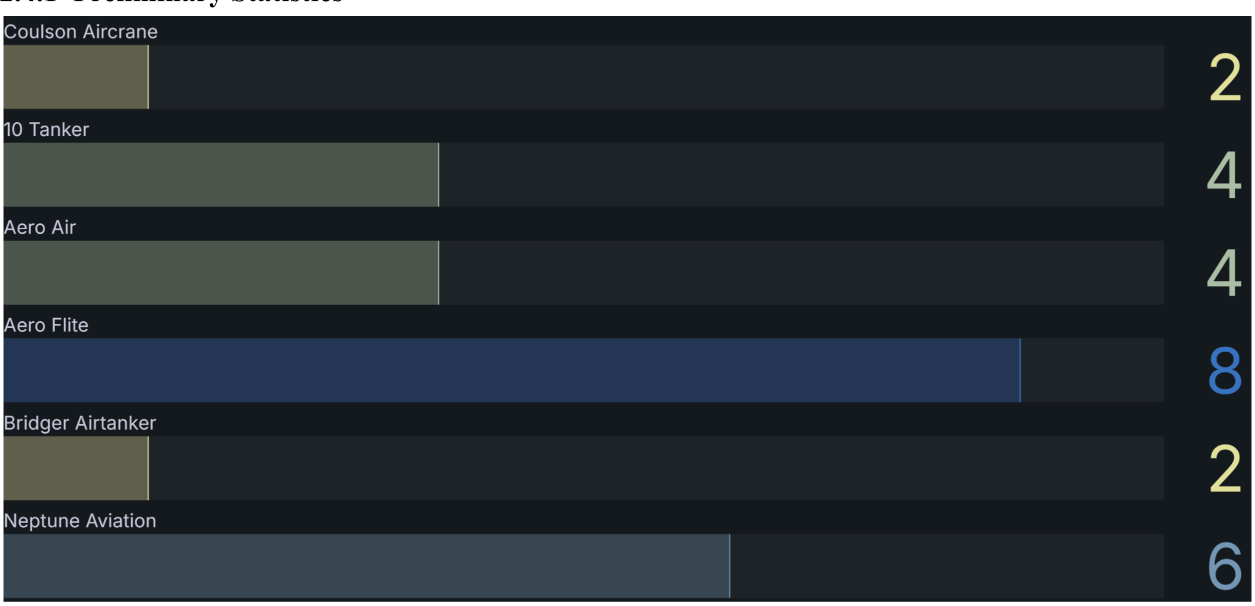

Figure 1: Number of Tracked Tankers by Contractor

The use of privately owned Large Airtankers (LATs) and Very Large Airtankers (VLATs) by the United States is primarily due to the unique dynamics of firefighting operations and the cost-effectiveness associated with utilizing private contractors. Private companies can rapidly mobilize their aircraft in response to wildfires. They can deploy their resources across different regions as needed, ensuring a quicker response time compared to waiting for government—owned assets, which might be limited in number and availability. The involvement of private companies fosters innovation and competition in the aerial firefighting industry. This graph shows the distribution of airtanker ownership among the different contractors in the US.

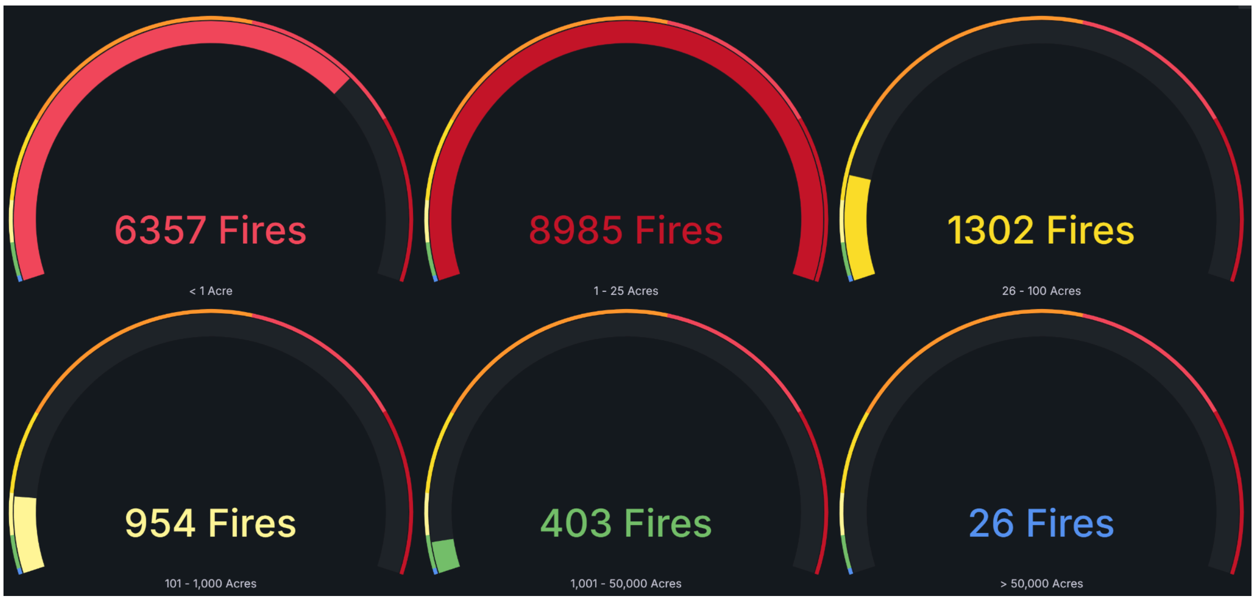

Figure 2: Total Fires by Fire Size (Acres Burned)

We collected data pertaining to wildfires that occurred in the United States from 2020 through 2022. In order to visualize the distribution of data, the chart shows all recorded fires during this time period and divides them into six groups distinguished by the amount of acres a given fire burned. A total of 26 fires burned over 50,000 acres, with one of the biggest being the Dixie Fire in California which burned over 400,000 acres in total. However, the vast majority of fires that took place from 2020 through 2022 only burned between 0 and 25 acres.

2.4.2 Suppression Statistics

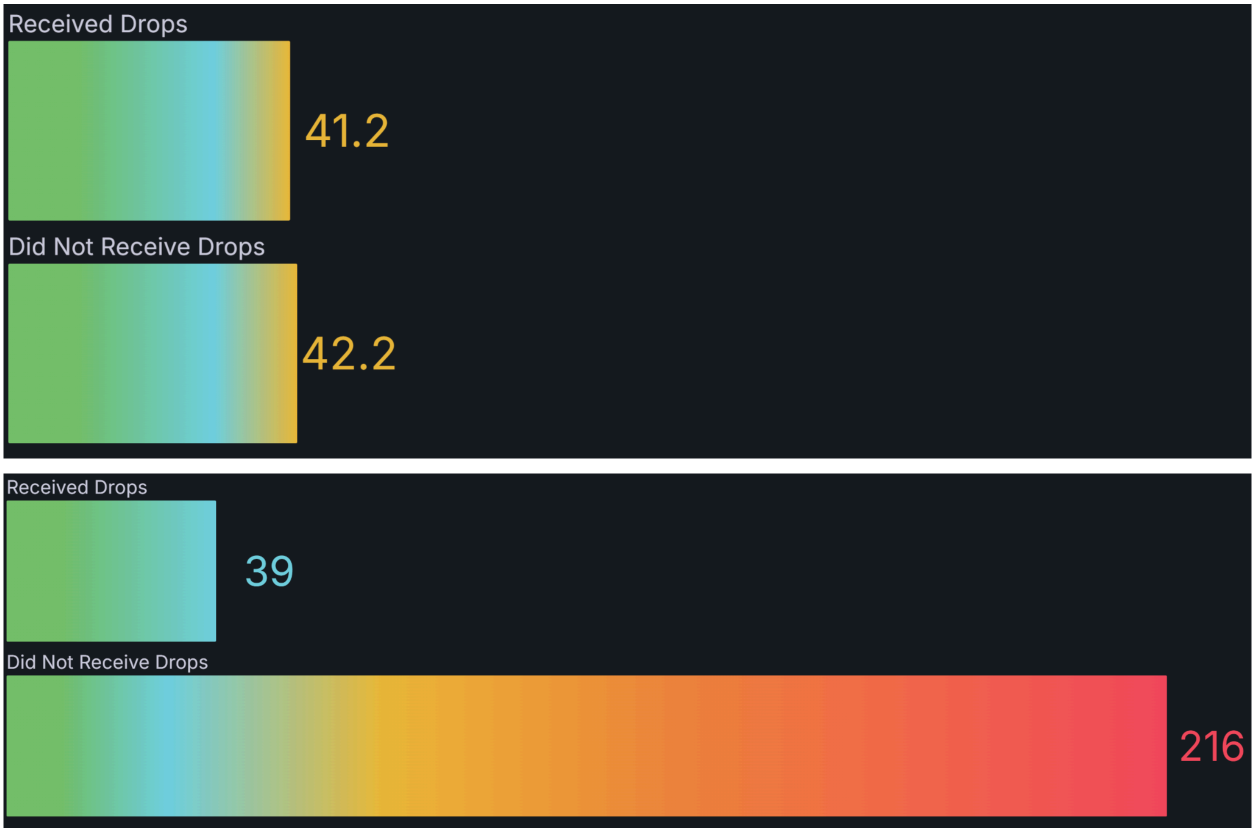

Figure 3: Wildfire Suppression Results

We calculated the number of larger fires that received retardant drops from LATs and VLATs during 2020 through 2022. Our data was limited due to missing information in the fire dataset, but of the instances with all necessary information, only 39 fires received drops, while 216 did not. The suppression results (based on time and acres burned) for both groups of fires—those that did and those that did not receive drops—was about 42%. While it is possible that the similarities in average suppression results indicate that aerial firefighting does not impact the outcome of a fire, we believe it may also be because the fires that received drops were worse and would have had a worse average suppression result had they not received drops. It is also possible that our results may be skewed by the portion of fires we were not able to calculate suppression results for due to missing data in our datasets.

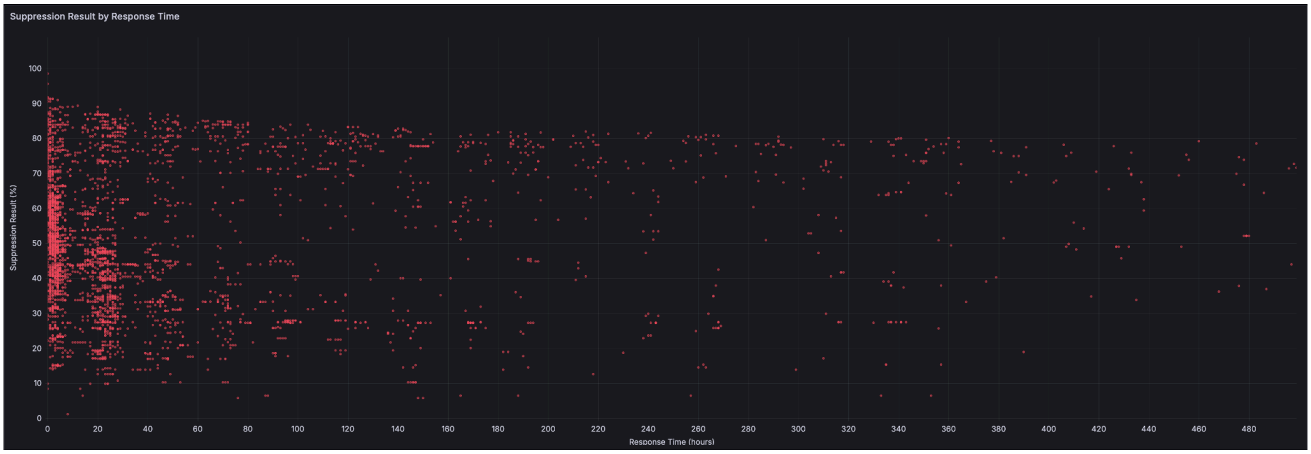

Figure 4: Wildfire Suppression Results vs. Response Time

Here we present the suppression result of each fire (as a percentage suppressed) plotted against the response time in hours. This statistic investigates a potential inefficiency in which quicker response times would result in better suppression of fires. We did not, however, find any strong pattern suggesting that this is the case; suppression results are fairly well distributed by response time. This visualization is cut off at 500 hours, but there are a few fires with response times exceeding that amount. The longest response time was 18,000 hours for a California fire that lasted 2 years.

2.4.3 Tanker Base Statistics

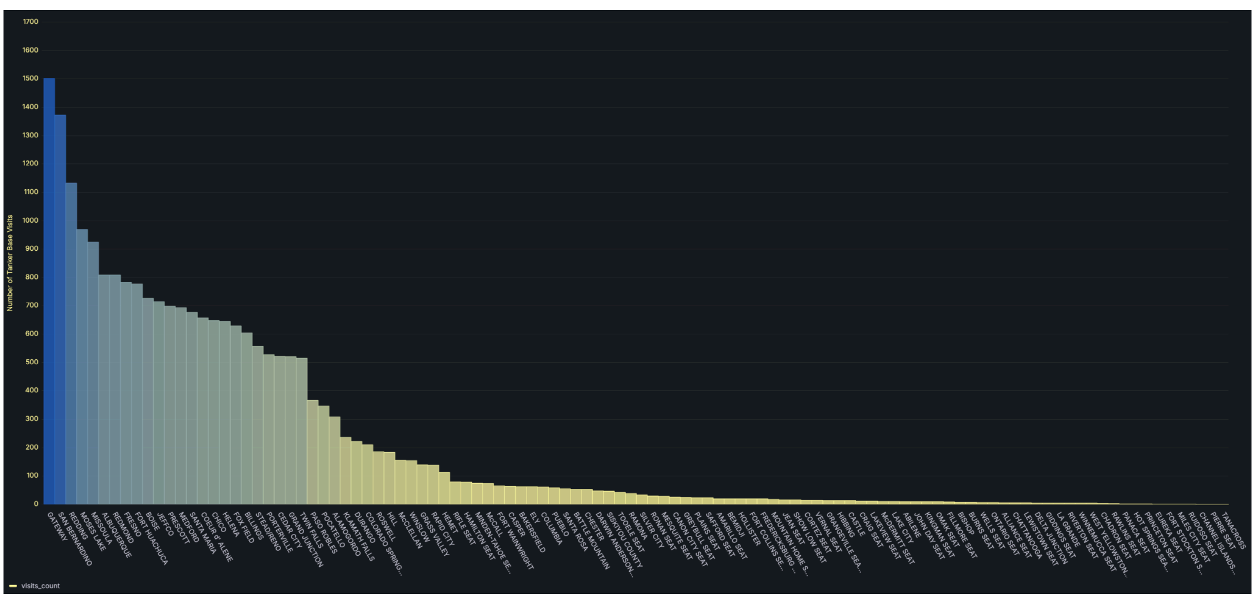

Figure 5: Visited Tanker Bases

Gateway is the most utilized tanker base, with over 1500 recorded visits from 2020 through 2022. This base plays a crucial role in firefighting efforts, given its high frequency of use. On the other end of the spectrum, Tanacross has seen only one recorded visit during the same period. This suggests that it is the least utilized base for tracked airtankers. Overall, the data indicates significant variation in utilization across different bases. This suggests that resources may be more effectively allocated to the bases that receive more visits, and it may point to places for new bases to be considered or for old ones to be retired if those resources are needed elsewhere.

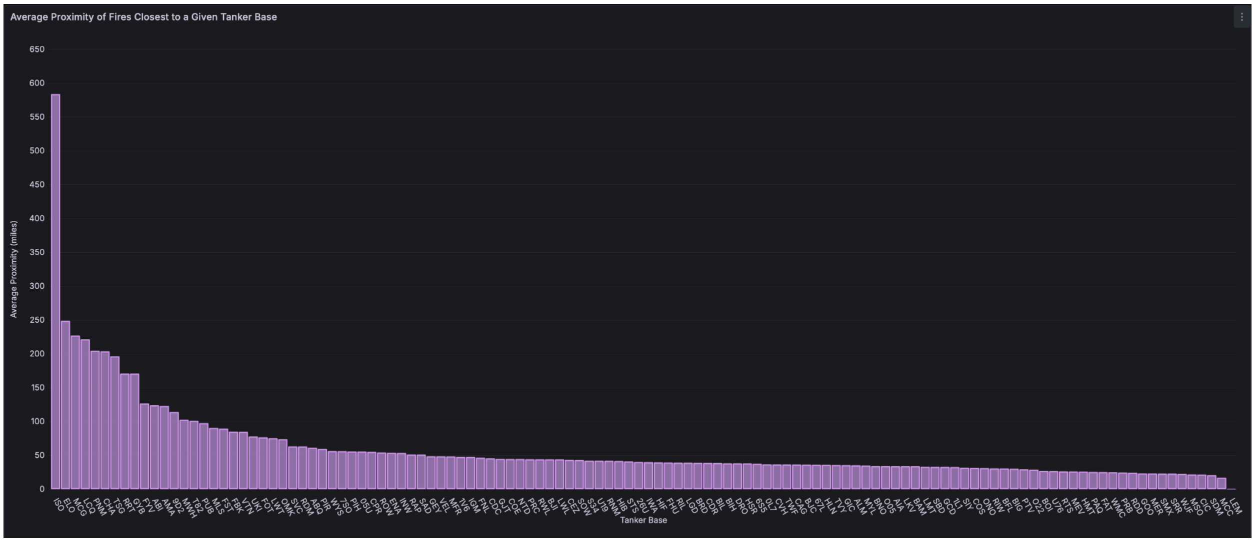

Figure 6: Proximity of Tanker Bases to Closest Fires

We wondered if perhaps some tanker bases tended to be better located, generally closer to fires. In order to explore this, we found the closest tanker base to each fire in our database. We then averaged the distances between the base and the fire of all fires closest to a given base. In this way, we can see that some tanker bases are generally much closer to fires—as close on average as 16.8 miles—while a few tend to be alarmingly far away—as far on average as 584 miles. This points to tanker bases that should be relocated or areas that might benefit from a higher density of bases.

2.4.4 Plane Statistics

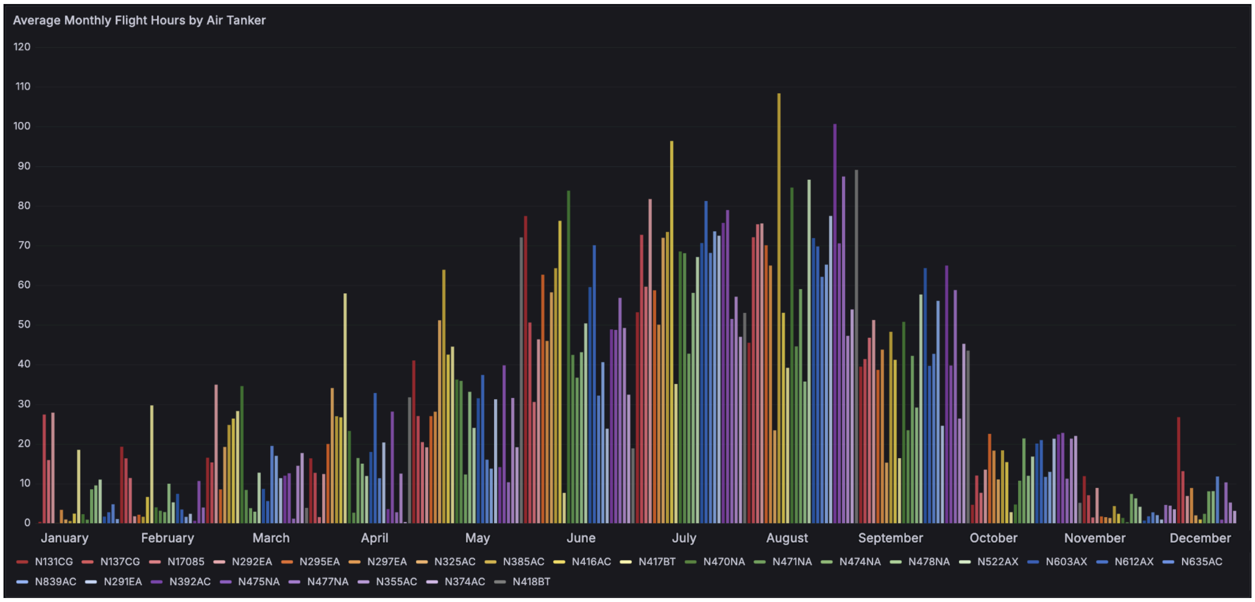

Figure 7: Average Monthly Flight Hours by Airtanker

The graph shows monthly average hours for LATs and VLATs for 2020-2022. Each LAT and VLAT is uniquely color coded, and the graph reveals the expected pattern of more usage during the summer months, especially July and August when fire season is at its peak. Some planes are evidently used more during some times of the year than others, which is an unexpected pattern that may warrant further investigation to see if all planes are being used efficiently.

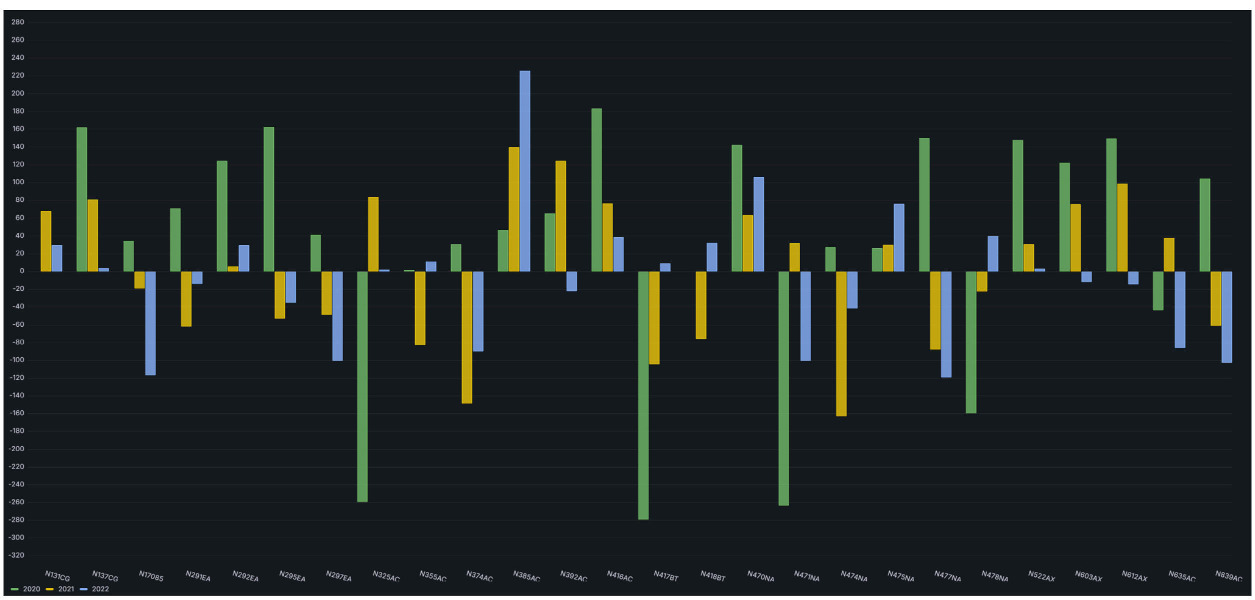

Figure 8: Tankers Airborne vs. Grounded Time

This graph measures the average difference between the amount of time each aircraft spent on the ground vs. in the air from 2020 through 2022. Each craft is categorized by their tail number. If a given value for a plane is positive on the Y axis, it means that aircraft spent more time in the air that year than usual. If it is negative, that craft was grounded more often than it was in the air.

2.5 Web Application

We utilized a React front-end as the foundation for building our web application. This would serve as the main framework for the project, where we added our various components ranging from our interactive Google Maps API to our statistics dashboard. Various sections of the web application are arranged into React components that are easily modified and implemented into the final product.

Figure 9: Web Application About Section

Figure 10: Web Application Info Section

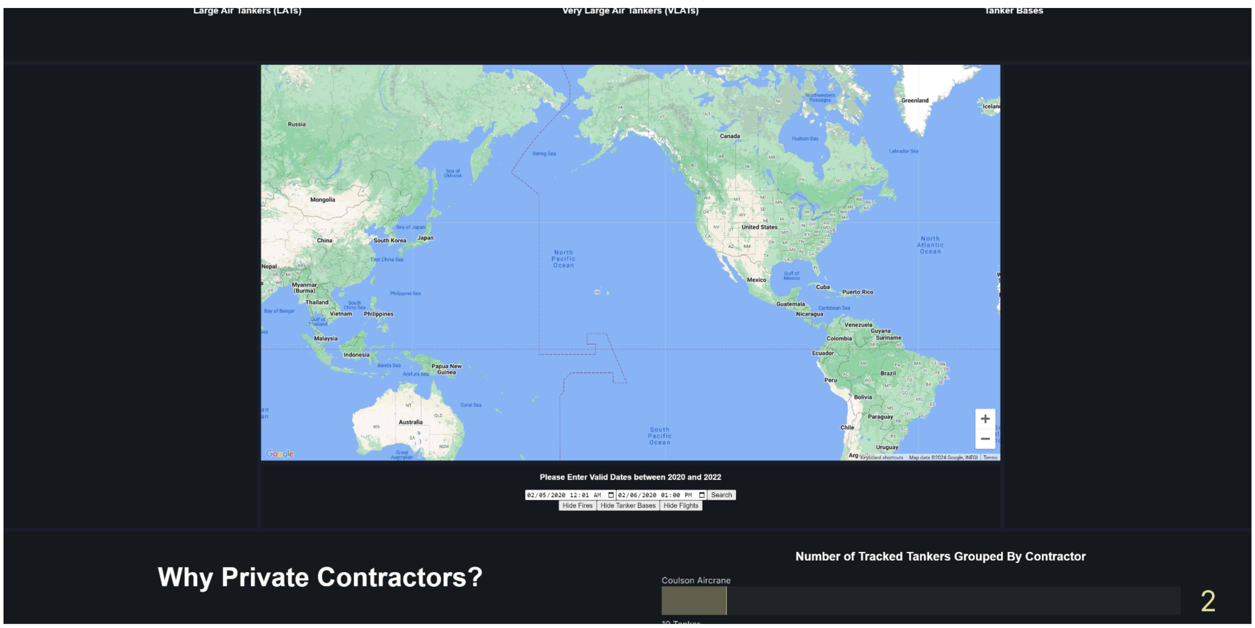

Figure 11: Web Application Map Section

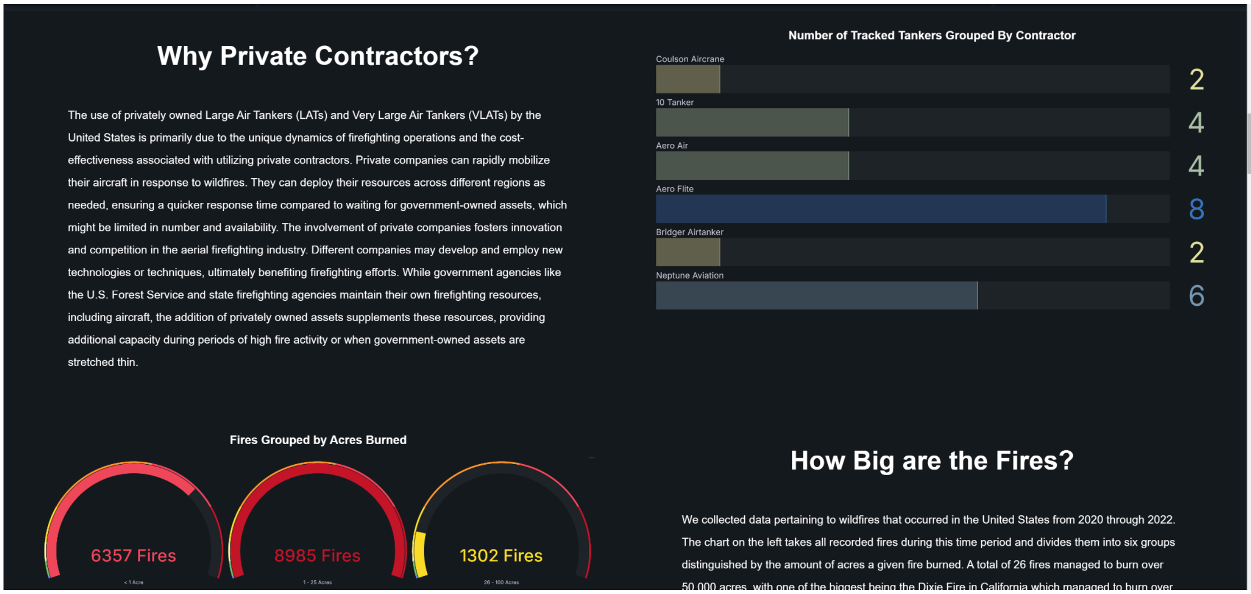

Figure 12: Web Application Dashboard Section

2.6 Map & Timeline

This project utilizes the Google Maps API to display all of the fire, flight, and tanker base data. Upon loading the dashboard, the application will populate lists of data based on the default date and render the data points onto Google Maps as “markers” (the icons you normally see on a Google map). Any icon marker can be clicked to display corresponding information about the fire, flight, or tanker base based on their location. Three hide buttons are included to allow icons to be hidden based on the data. The input dates are validated to only include dates from 2020 and 2022. It also ensures that dates are in order to ensure the database query works as intended.

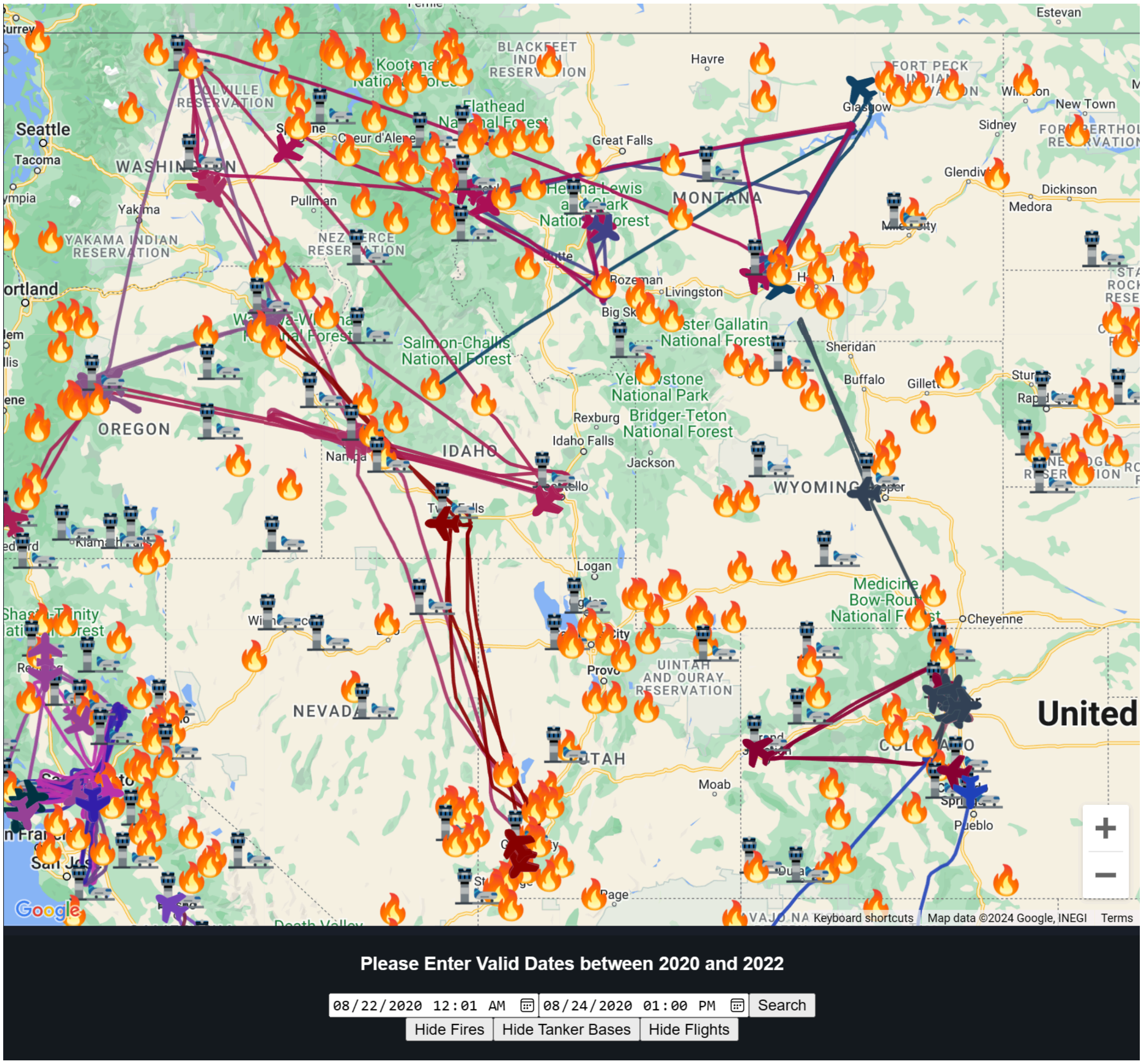

Figure 13: Google Maps with Fires, Flights, & Tanker Bases

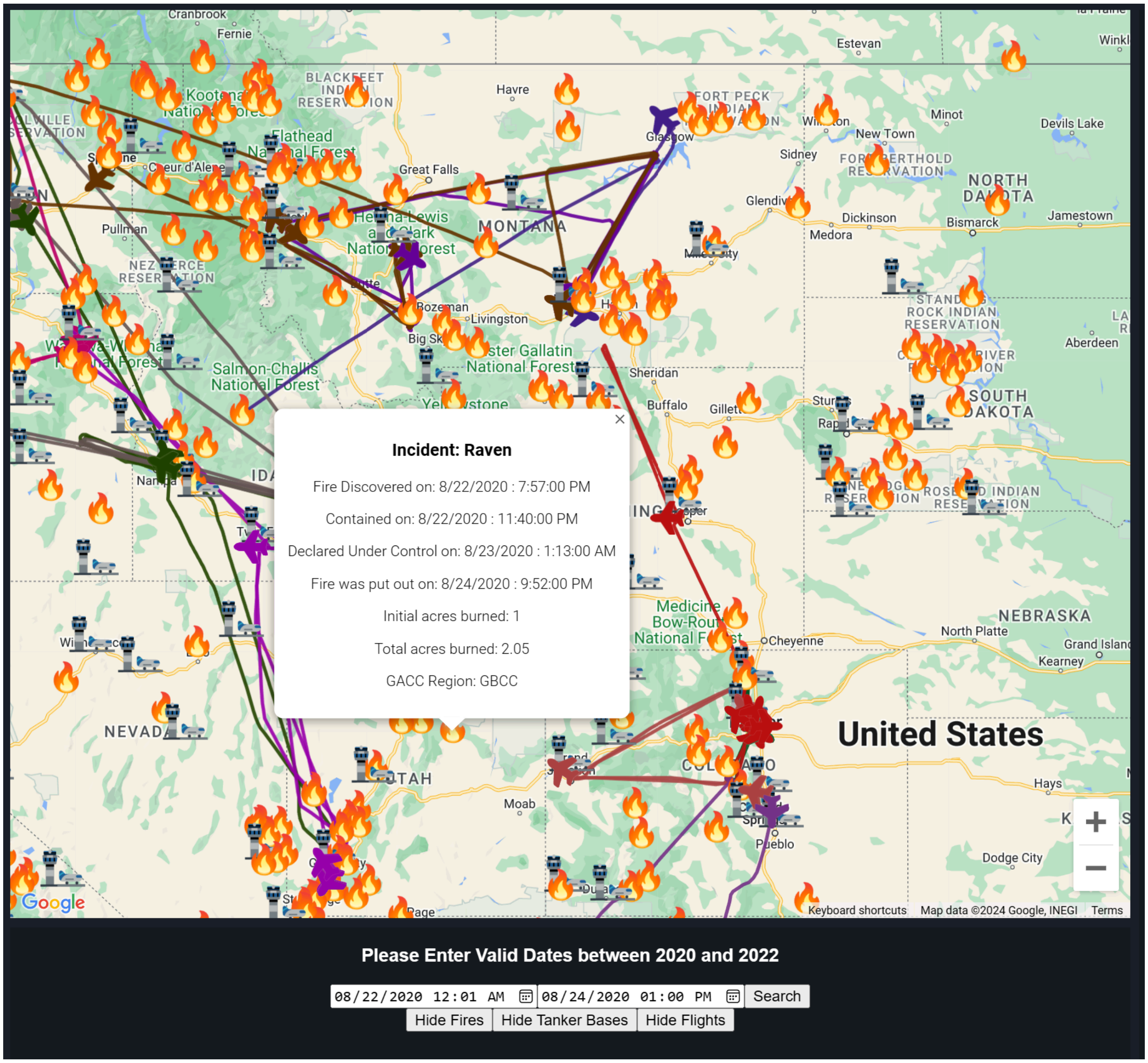

Figure 14: Infobox Displaying Fire Information

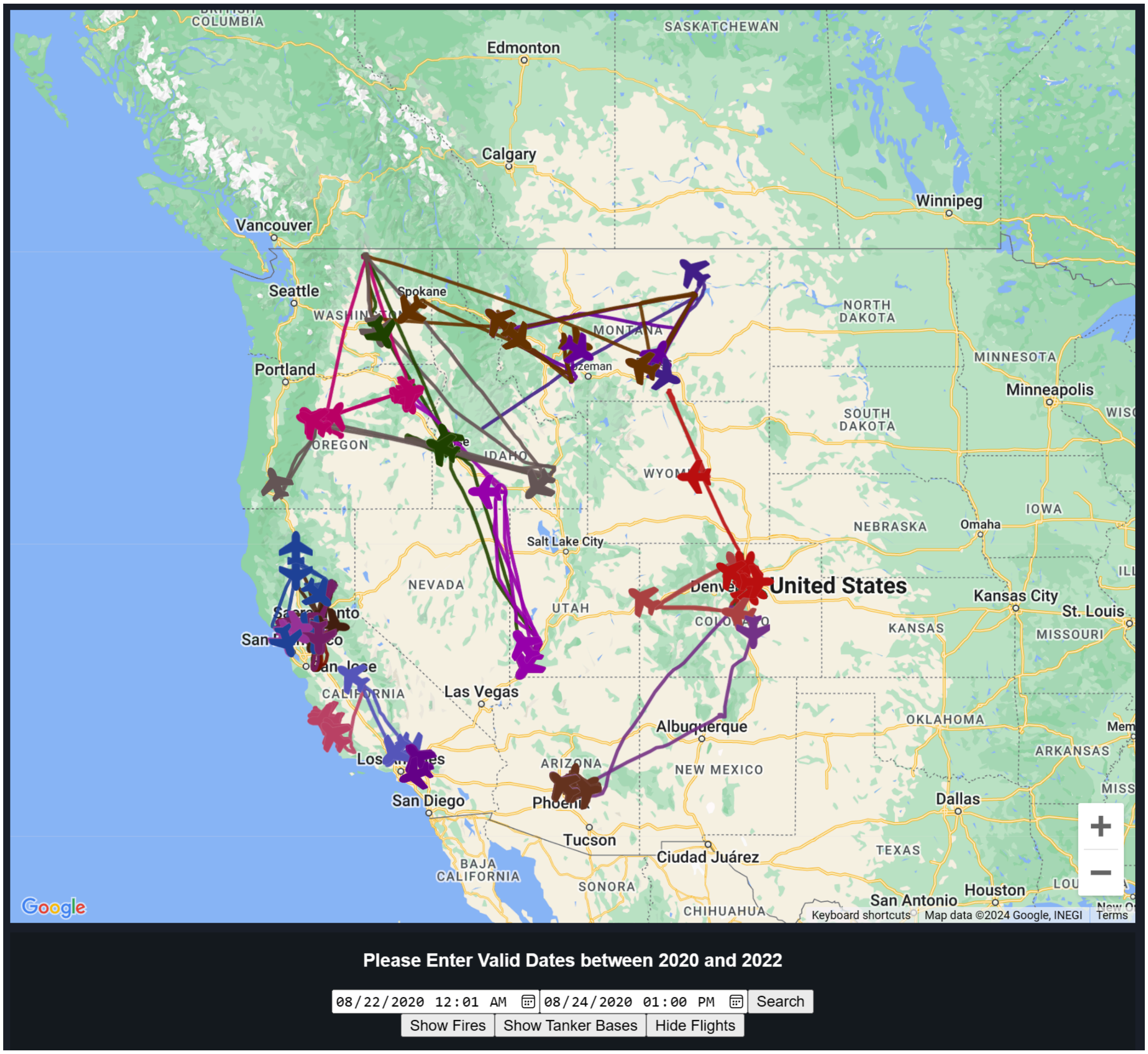

Figure 15: Google Maps with Fires and Tanker Bases Hidden

3 Work Remaining

All of the major features listed in the Scope Commitment checklist were completed. Additional work remaining includes stretch goal features that were planned out but not implemented and attribute trimming in the database, which would remove the unnecessary columns retained from earlier iterations of the project that do not impact the results. These goals were:

- Timeline Timelapse: This feature would allow the data points to be played back in increments of an hour between the two dates provided from the search query.
- Database Pruning: The removal of unnecessary and unused attributes from the fire table. This includes modifying the Python scripts to produce SQL scripts that do not include the removed fire attributes.

4 System Architecture and Design

4.1 System Architecture

The layout of the application (Figure 10) details the interactions between each part of the simulator.. Our system is built of 3 main structures and a custom library. The front-end web app handles the display of data information and plotting of retrieved points on the map. The data is retrieved with the back-end that handles the database connection, database queries and Google Map API requests. Our data is stored in a MySQL database running on an offsite server. Finally, our custom library was built to handle the retrieval of flight data from the FlightAware AeroAPI.

Implementation Languages

- Python
- React Webpage
- MySQL

Major Libraries

- Pandas
- Google Maps API
- FlightAware AeroAPI

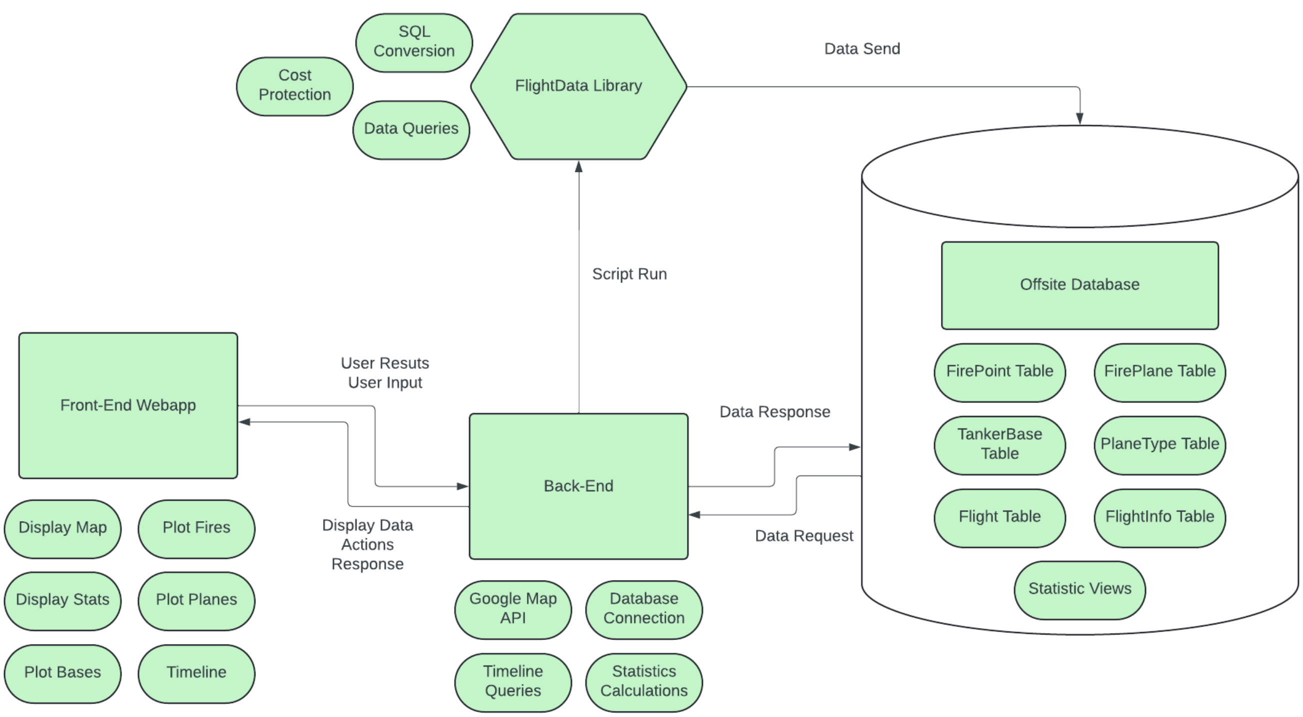

Figure 16: System Architecture

4.2 Database Design

4.2.1 Table Information

AirTanker: This table holds information about a type of AirTanker like plane model and capacity

tanker_type	tank_size
B737	4000
BAe-146	3000

FirePlane: Information about AirTankers containing tail number, contractor and type

tail_no	tanker_type	contractor
N131CG	C130Q	Coulson Aircrane
N137CG	B737	Coulson Aircrane

FlightInfo: Contains information about individual flights that a plane took. includes takeoff and landing times, and if the source and destination airports are known those are also provided

flight_id	source	destination	takeoff	landing
ADSB123-1642537493-adhoc-0	BZN	BZN	2022-01-18 20:35:11	2022-01-18 21:40:06
BGR282-1647707320-adhoc-0	AZA	AZA	2022-03-19 17:25:28	2022-03-19 20:21:40

Flight: Each row contains information about a plane's exact position and information at a single timestamp. This information includes latitude, longitude, heading, altitude, and groundspeed. Usually we have an entry for a single flight every 5 seconds, but sometimes logging occurrences have gaps and can be up to a few minutes apart.

flight_timestamp	latitude	longitude	altitude	altitude_change	ground_speed	heading
2021-04-11 15:19:08	34.0889300000000	-117.2565200000000	13.0000	C	175	250
2021-04-11 15:19:24	34.0856000000000	-117.2715600000000	15.0000	C	192	265

TankerBase: This contains location and base information about a tanker base. The location information provides us latitude and longitude values in the center of the base, and base information gives us values including the amount of planes they can stow, and the largest planes they can handle

base_code	base_name	airport	region	elevation	latitude	longitude
05U	EUREKA SEAT	Eureka Airport	GREAT BASIN	5958	39.6038333333333	-116.0036666666667
0L7	JEAN SEAT	Jean Airport	GREAT BASIN	2831	35.7728333333333	-115.3291666666667

FirePoint: This contains relevant data about the fires that we collected. Each row contains a large amount of information, but we mostly used the latitude and longitude, acres burned, and the date the fire was put out or controlled/contained.

incident_name	latitude	longitude	containment_date	fireout_date	final_acres
BAY CREEK	60.9271017090001	-149.5750228470000	2014-08-20 05:25:00	2014-08-28 18:48:00	NULL
Hope Rx	60.9111017050001	-149.6060228360000	2014-09-12 16:31:00	NULL	NULL

The decision to design multiple tables for the flight info was to ensure the fastest query throughput. FlightInfo keeps the most valuable information about a flight in a smaller dataset, to allow querying of that smaller table when looking for flights during certain times. With the information from the FlightInfo table we can access all the rows from the Flight table that share the flight_id. We can then use those specific entries from the FlightTable to display flight trajectories in our map and calculate statistics such as cross checking a flight path with fire locations to register a drop.

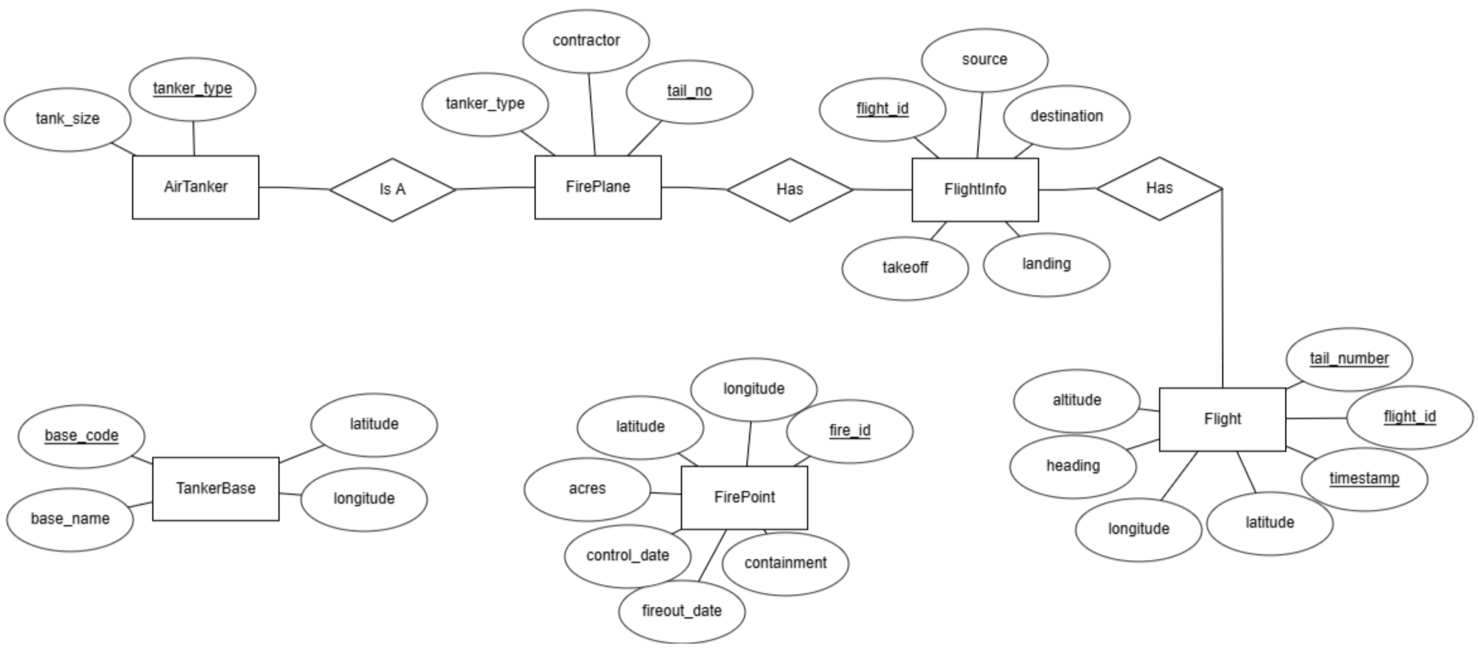

Figure 17: Database Entity Relationship Diagram

4.3 FlightData Library

The FlightData library is a custom built library to query the FlightAware AeroAPI in one continuous script. It starts by reading a list of all the Tail Numbers for the LAT and VLAT planes and querying every flight_id for those planes within our 3 year period from 2020 through 2022. Then using these flight_ids we query all the flight points and flight information from that flight. A flight point is information about a plane at an exact timestamp. Querying the flight_id for flight points gets us time stamped points of latitude, longitude, heading, ground speed and altitude at that exact time during the flight. Finally, all the retrieved information is sent to a script to convert it to SQL rows in our database. Due to the expensive nature of flight data, this script evolved through rigorous testing to ensure no extraneous costs were incurred. This library can easily be expanded to handle more planes or a larger date range by updating its configuration files. If in the future our sponsor wishes to expand on data, they have the ability to do so.

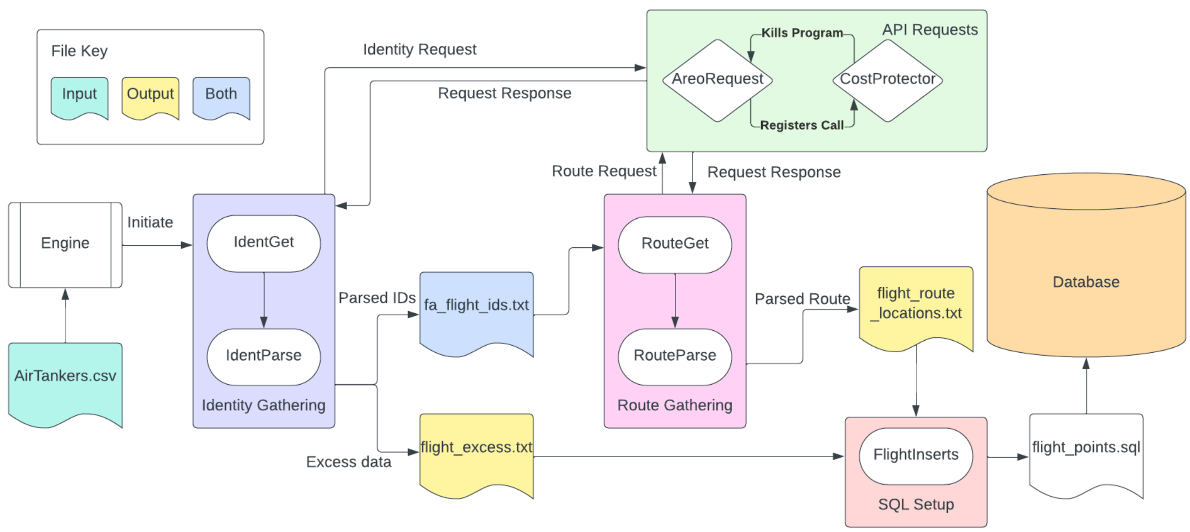

Figure 18: FlightData Library Modules

4.4 Web Application Design

The front-end web application utilizes the React Framework to configure and create the visuals of our project. The application renders Google Maps onto a webpage to display the fire, flight, and tanker base data. The static NodeJs server handles communication between the React application and the MySQL database when the application needs to populate new information and render it onto Google Maps. We created a timeline to allow the user to query for different data between two dates. Due to the sheer volume of data, it takes time to query, format, and render it onto Google Maps for the user to interact with them, however, we implemented the FlightInfo table to reduce the time to query flights. All data points on Google Maps support clicking to pop up info boxes to explain fire, flight, and tanker base logistics and important information.

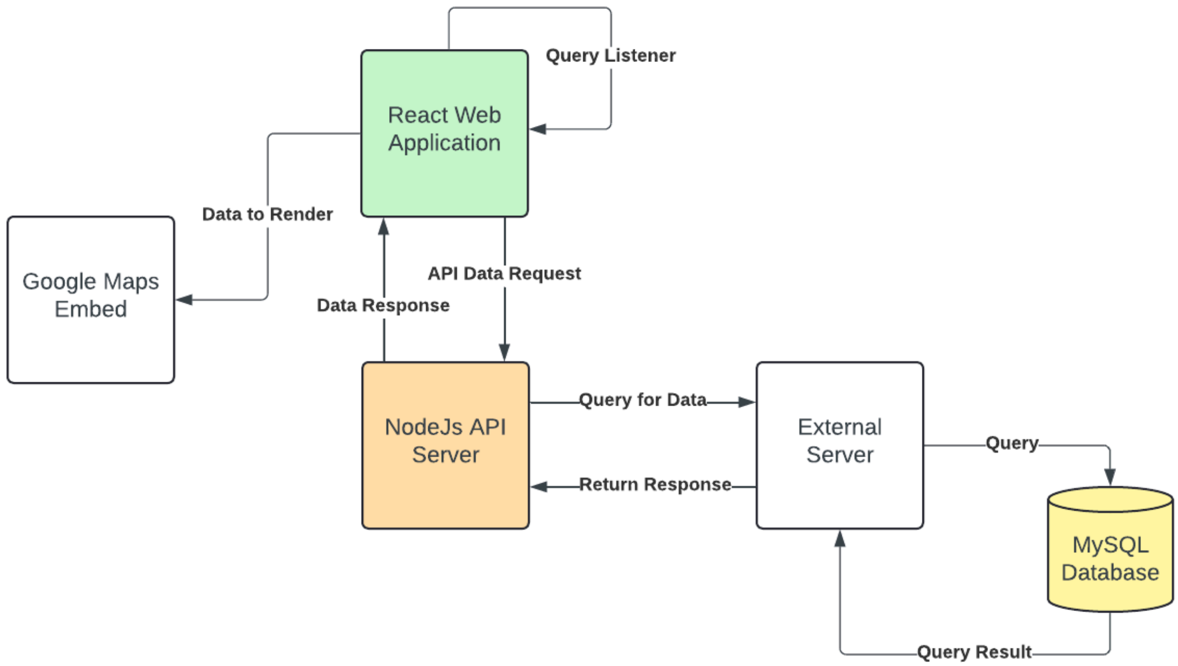

Figure 19: Web Application Architecture

5 System Evaluation

Our application underwent two separate usability test sessions during spring semester. These tests aimed to evaluate the usability of our front-end and the components we had implemented. We came prepared to these sessions with packets of user-oriented questions we wanted our test users to answer. These questions targeted the usability and effectiveness of numerous UI/UX components including our Grafana visualizations, accompanying descriptions, and the Google Maps API functionality. Over several hours across multiple days, we received valuable feedback about our application and its effectiveness at conveying relevant information to users. We received feedback regarding how our frontend effectively conveys the information users need to know in order to understand and navigate the various features of the application. More contextual information was added throughout the web page as well as the complete overhaul of several section layouts to make room for said information. We added an 'About' section, info sections regarding the tankers we are tracking, their tanker bases, and ensuring each and every visualization is accompanied by their own paragraph of text explaining our findings.

6 Project Delivery, Deployment, and Maintenance

The team will provide the sponsor with the following: The React web app source code, Python scripts that create SQL scripts to create the database, local SQLite database files to run the web app locally, and documentation that explains the setup for the web app, python scripts, and the queries used for the visualizations. There are no plans for deploying the web application. This will be handled by the sponsor.

For any maintenance for the application, most solutions will be provided through documentation, such as package deprecations and possible features that can break.

7 Project Management

Our team worked in two-week sprints throughout our project. We communicated daily with asynchronous Discord standups, and worked together in small groups as needed between our weekly team meetings. We met with our advisor weekly and our sponsor bi-weekly. The system we initially devised worked well for us. We ironed out some technical coordination on standards for coding and Trello cards, and initially struggled by doing the bulk of our work at the end of the sprints, putting more pressure on us to get it done. We also developed a guide for standards and improved on spreading work out more evenly.

7.1 Major Feature Estimates and Final Time

The following table includes our major features with our initial estimates versus actual time spent.

Feature Completion Table

Feature	*Description*	*Initial Estimate*	*Final Total Time*
Complete Database	Our database is complete with all of the data we intend to work with for this project. Data that is cleaned and ready for other project components to utilize.	68	52
Functional Dynamic Map	A map of the US overlaying tanker bases as well as fire location data and flight paths of aerial firefighting tankers as they change on a user-controlled time scale.	115	102
Machine Learning Model (Dispatch Optimizer)	Predicting fire outcomes based on the optimized distribution of aerial resources (involving the movements of large airtankers from tanker bases to fires). ● 1st stage: functioning predictive model ● 2nd stage: integration into map (best resource scenario side-by-side with the actual progression of the historical fire)	37	Moved to stretch goals through scope change
Statistical Analysis & Visualizations	A series of statistics and related visualizations about interactions between planes, fires, and tanker bases with the intent to identify inefficiencies in the system.	55	55

Our estimates were close to accurate with most of our major features taking a bit less time than initially expected. The numbers given are through our story point system which we agreed upon during the early stages of the project, with 1 point meaning “half of a school day's work”. Our statistical analysis was the most accurate because we decided on the story points half way through the project after we had a better feel for our estimates.

7.2 Ownership

Our project was split up through an ownership system. This system gave each team member a certain piece of the project to be the owner of. While the owner was not the only one working on that piece of the project, they were the expert to go too when questions and ideas arose.

Data:

- Fire Data: Kevin Dang
- Flight Data: Drew Bogdan
- Tanker Bases: Claire Yegian
- Database Design & Population: Vincent Do

Application:

- Back-end Work: (Connections, Routing, Stats) Kevin Dang
- Front-end Work: (Slider, Stats, Styling) Kole Davis
    - (Map) Kevin Dang

7.3 Major Contributions by Member

Kevin Dang

- Completed Dynamic Map, Timeline, API Endpoints for Database

Drew Bogdan

- Flight Data Library, Statistics Calculations

Vincent Do

- Fire Data Gathering, Database Design, Stats Algorithms, Statistics Calculations

Kole Davis

- Grafana Visualizations, Front-End Design & Architecture

Claire Yegian

- Tanker Base Data, Suppression Algorithm, Statistics Calculations