Software Design Document

for

Displaying DOP Data

Version 2.0 approved

Prepared by John Chen, Phong Diep, Matthew Franco, Roaf Htun, Steven Lopez Carachure, Andrew Mendiola, Natalie Perales, Cindy Rodriguez, Steven Yuen

The Aerospace Corporation

5/10/24

Revision History

Name	Date	Reason For Changes	Version
Version 1	9/22/23	Created UI	1.1
Version 2	10/13/22	Connected BackEnd and Database	1.2
Version 3	11/03/22	Generated Heat Map	1.3
Version 4	11/17/22	General improvements all around	1.4
Version 5	12/1/23	Finishing touches for fall semester	1.5
Version 6	3/22/24	Connected FrontEnd Database and BackEnd	1.7
Version 7	4/5/24	FrontEnd touch ups	1.9
Version 8	5/1/24	Finishing touches for Spring semester	2.0

1. Introduction

1.1 Purpose

To develop a web application to visualize a GPS performance metric called Dilution of Precision (DOP) in a web application. DOP is a geometric measure of GPS accuracy and is commonly analyzed to gauge GPS performance. The resulting data is typically viewed on top of a world map. The ultimate goal of the web application is a world map view with a DOP overlay based on the selected date from user input.

1.2 Document Conventions

The standards and typographical conventions followed in this Software Requirements Specification (SRS) for this web application project dictate that priorities for higher-level requirements are assumed to be inherited by detailed requirements. This convention is maintained throughout the document to streamline the prioritization process. It ensures consistency by eliminating the need to assign priorities individually to each detailed requirement, as they are implicitly derived from their higher-level counterparts.

1.3 Intended Audience and Reading Suggestions

This Software Requirements Specification (SRS) is designed to cater to various stakeholders involved in the value web application project. The intended readers include developers, project managers, marketing staff, users, testers, and documentation writers. Each reader type has specific sections within the document that are most pertinent to their roles and responsibilities.

1.4 System Overview

The software system under consideration is a web application designed to display and present Differential Optical Pathlength (DOP) data. DOP data is critical in various scientific and industrial applications, particularly in atmospheric science, remote sensing, and environmental monitoring. The DOP data web application is a comprehensive solution for users requiring access to, analysis of, and collaboration on Differential Optical Pathlength data.

2. Design Considerations

2.1 Assumptions and Dependencies

Describe any assumptions or dependencies regarding the software and its use. These may concern such issues as:

- The web application assumes compatibility with commonly used web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge.
- The web application is designed to be platform-independent, assuming compatibility with major operating systems, including Windows, macOS, and Linux.
- Users are assumed to have basic proficiency in using web applications, including navigation, data interpretation, and interaction with visualization tools.
- The web application anticipates future updates and changes in functionality to accommodate evolving user needs, technological advancements, or changes in DOP data sources.

2.2 General Constraints

Describe any global limitations or constraints that significantly impact the design of the system's software (and describe the associated impact). Such constraints may be imposed by any of the following (the list is not exhaustive):

- The web application assumes compatibility with commonly used web browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge.
- The web application is designed to be platform-independent, assuming compatibility with major operating systems, including Windows, macOS, and Linux.
- Users are assumed to have basic proficiency in using web applications, including navigation, data interpretation, and interaction with visualization tools.
- The web application anticipates future updates and changes in functionality to accommodate evolving user needs, technological advancements, or changes in DOP data sources.

- The availability and accessibility of DOP data from external sources are assumed to be consistent during normal operation.

2.3 Goals and Guidelines

Describe any goals, guidelines, principles, or priorities that dominate or embody the design of the system's software. For each such goal or guideline, unless it is implicitly obvious, describe the reason for its desirability. Feel free to state and describe each goal in its sub-subsection if you wish. Such goals might be:

- Prioritize user experience by designing an intuitive, user-friendly interface.
- Implement advanced data visualization techniques for clear and meaningful presentation of DOP data.
- Design a scalable and modular architecture to accommodate future growth and updates.
- Emphasize speed and responsiveness in data retrieval, processing, and application performance.

2.4 Development Methods

Briefly describe the method or approach used for this software design. If one or more formal/published methods were adopted or adapted, then include a reference to a more detailed description of these methods. If several methods were seriously considered, then each such method should be mentioned, along with a brief explanation of why all or part of it was used or not used.

The software design for the DOP Data Web Application adopts an Agile Development methodology, specifically leveraging the principles of Agile Manifesto. Agile emphasizes iterative development, collaboration, and adaptability to changing requirements. The decision to use Agile was driven by the need for flexibility, frequent feedback loops, and the ability to accommodate evolving user needs.

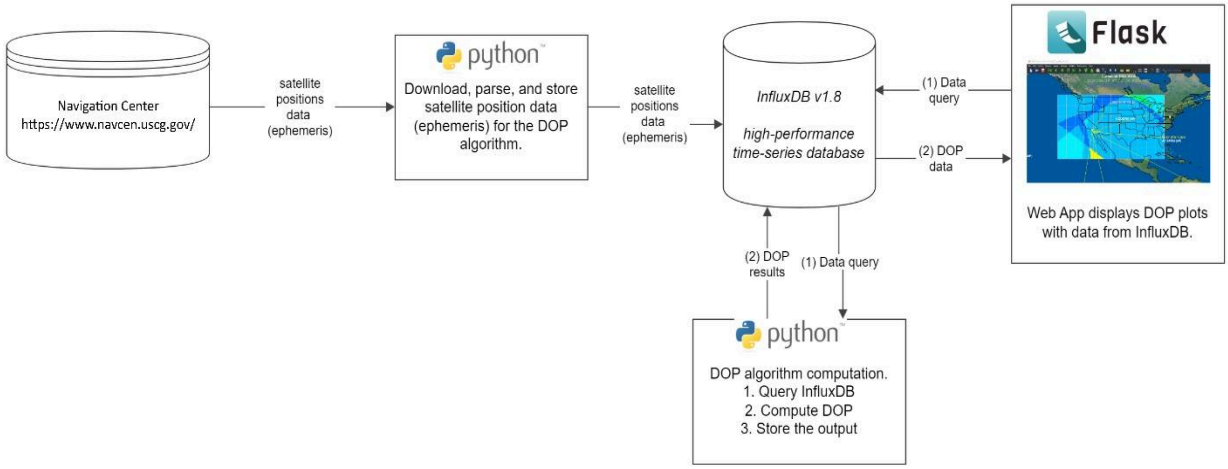
3. Architectural Strategies

Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (possibly referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off. Such decisions might concern (but are not limited to) things like the following:

- Utilizing a modern and widely supported programming language (e.g., Python/JavaScript) and a relevant technology stack (e.g., React for front-end, Express for back-end).
- Adopting a component-based architecture for the front-end.
- Implementing a responsive and intuitive user interface with an emphasis on data visualization.
- Adopting a distributed architecture to handle real-time updates of DOP data.
- Implementing resource management policies for efficient utilization of server resources.

These design decisions collectively shape the architecture of the DOP Data Web Application, aiming to fulfill user requirements, adhere to industry best practices, and support future extensibility and maintainability. Each decision is made considering the defined goals and principles, ensuring a coherent and effective system design.

4. System Architecture



- Develop a web application to display DOP data
- The web application should allow the user to select a date to display DOP data

5. Policies and Tactics

Describe any design policies and/or tactics that do not have sweeping architectural implications (meaning they would not significantly affect the overall organization of the system and its high-level structures), but which nonetheless affect the details of the interface and/or implementation of various aspects of the system. Make sure that when describing a design decision that you also discuss any other significant alternatives that were considered, and your reasons for rejecting them (as well as your reasons for accepting the alternative you finally chose). Such decisions might concern (but are not limited to) things like the following (Must include 5.1, 5.2, and 5.3. The rest of these categories or custom ones can be added as needed.):

5.1 Choice of which specific products used

Web browser, VSC, InfluxDB

5.2 Plans for ensuring requirements traceability

Ensure that each DOP data requirement is linked to downstream artifacts like UI components, database schemas, and test cases.

5.3 Plans for testing the software

Develop detailed test cases and test scenarios for each functional requirement, considering various input conditions, edge cases, and potential user interactions.

6. Detailed System Design

6.x Name of Component (Module)

6.x.1 Responsibilities

The primary responsibilities of this component in the DOP Data Web Application include:

- **Data Retrieval:** Retrieve DOP data from external sources based on user queries.
- **Data Processing:** Process and transform raw data into a format suitable for display.
- **User Interaction:** Facilitate user interactions for querying and filtering DOP data.
- **Real-time Updates:** Provide real-time updates of DOP data to users.
- **Error Handling:** Implement robust error handling mechanisms for data retrieval and processing issues.

6.x.2 Constraints

- Timing: Must ensure timely data retrieval and processing to meet user expectations.
- Storage: Manage data storage efficiently to prevent resource exhaustion.
- Component State: Maintain accurate internal state to support real-time updates.
- Data Formats: Adhere to specified data formats and standards.
- Concurrency: Address potential race conditions during data retrieval and processing.
- Error Handling: Enforce constraints on error reporting and resolution.

6.x.3 Composition

Responsible for fetching DOP data from external sources.

6.x.4 Uses/Interactions

- User Interface Module: Utilizes data for display and user interaction.
- Analytics Module: Provides data for statistical analysis.

6.x.5 Resources

- Memory: Manages memory for storing retrieved and processed data.
- External Databases: Resources for fetching DOP data.

6.x.6 Interface/Exports

- Fetches DOP data based on user queries.
- *Responsibilities:* Retrieve data from external sources and handle data retrieval errors.

- *Constraints:* Timing constraints for responsive data retrieval.

7. Detailed Lower level Component Design

7. x Name of Class or File

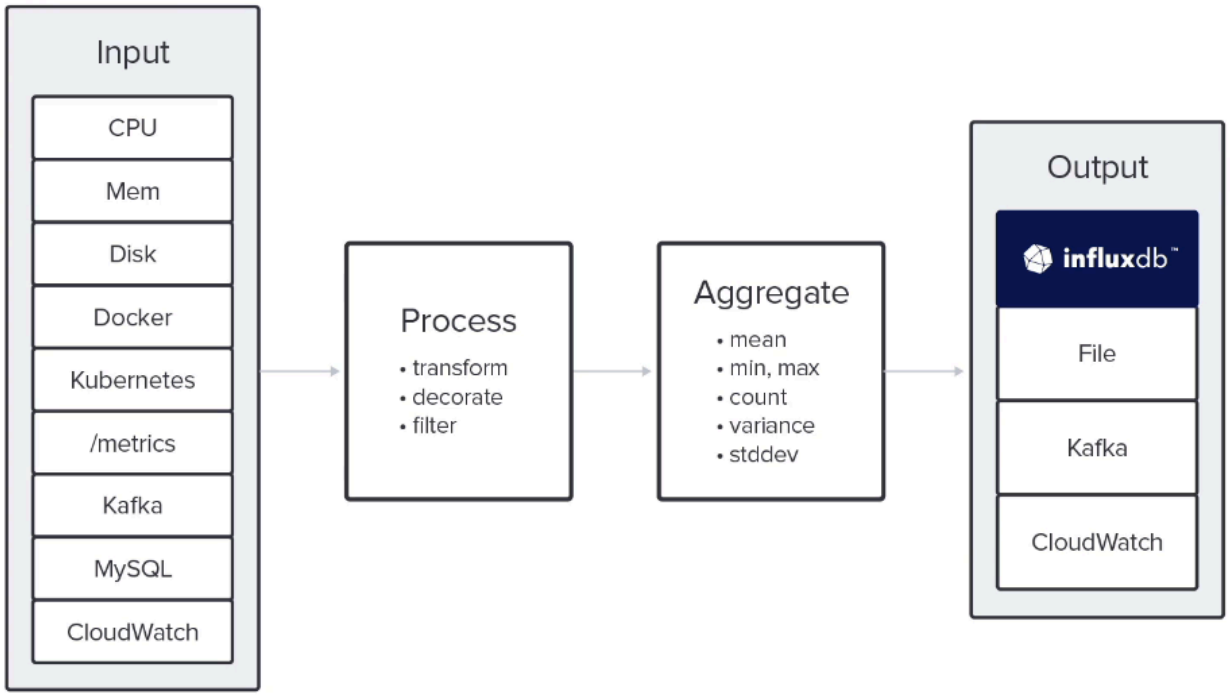
7.x.1 Classification

The component is classified as a **DataProcessor** class within the DOP Data Web Application. This class is responsible for processing raw DOP data retrieved from external sources.

7.x.2 Processing Narrative (PSPEC)

1. **Input:** Raw DOP data received from the DataRetrievalService.
2. **Processing Steps:**
    - Parse and validate the incoming data format.
    - Apply necessary transformations to structure the data for display.
    - Handle any data processing errors and log relevant information.
3. **Output:** Processed DOP data ready for presentation in the user interface.

8. Database Design

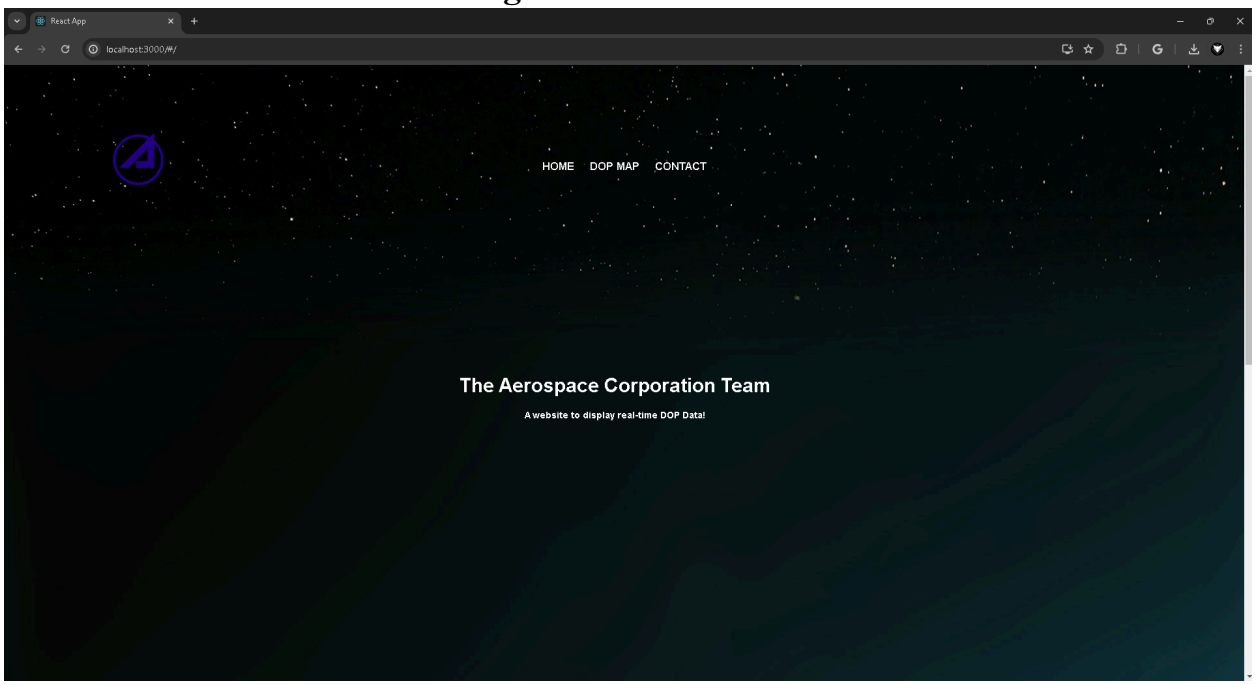


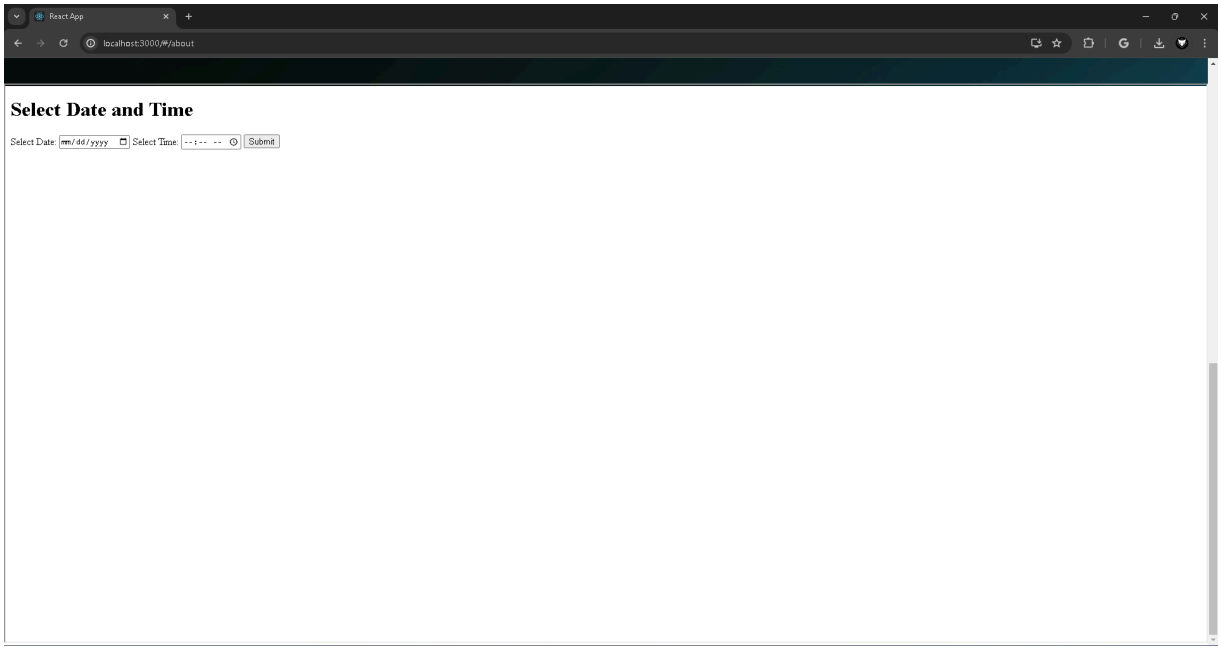
9. User Interface

9.1 Overview of User Interface

Once authenticated, users access a dynamic dashboard that provides an overview of DOP data through interactive charts and graphs. Users can explore trends and patterns by navigating through these visualizations, obtaining at-a-glance information. The system facilitates data exploration through custom queries and filters, allowing users to focus on specific subsets. Immediate feedback is provided on the success or failure of queries, guiding users in refining their search criteria.

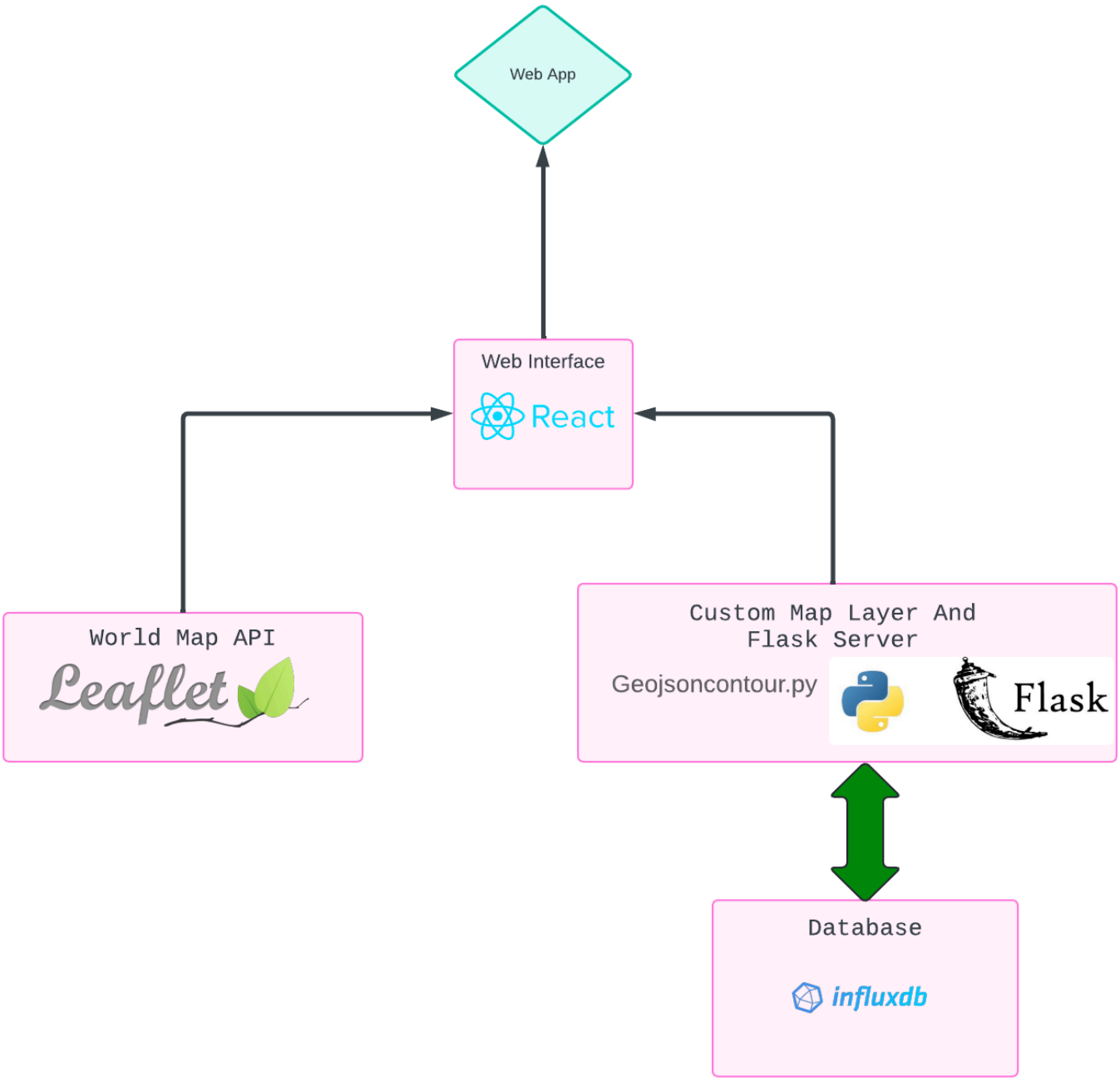
9.2 Screen Frameworks or Images




React App
localhost:3000/#/about
Select Date and Time
Select Date: mm/dd/yyyy
Select Time: --:-- --
Submit


React App
localhost:3000/#/about
1
3
5
6
8
10
POOP
ASIA
NORTH AMERICA
AFRICA
SOUTH AMERICA
OCEAN
ANTARCTICA

9.3 User Interface Flow Model



10. Requirements Validation and Verification

The web app can initially be a single-page application. It needs to display the following:

- The DOP data displayed on top of a world map like in the screenshots above, let's call it the **DOP map**,
- A date picker so that the user can select a single date and view the corresponding DOP map,
- A short list of predefined Points of Interest (PoI), such as "USA", "Russia", "China", etc. When the user clicks on a PoI, the DOP map updates,

Stretch goals (the above features should be finished before these are attempted):

- Zoom into the DOP map
- Export the DOP map to an image
- Zoom into the DOP map to view new, increased resolution DOP map
- DOP algorithm optimization
- Playback feature that plays back DOP data for a time window, like a movie.

11. Glossary

The following technologies will be used:

- Python data processing
- Flask web development
- InfluxDB Version 1
- Front-end web development with JavaScript and related technologies
- Leaflet.js for Map design
- (optional) DOP conceptual understanding
- Satellite ephemeris (satellite positions in space)
- DOP in GPS
- Satellite availability (Satellites' contact with ground stations)
- Common satellite and ground station terms such as:
    - Latitude Longitude Altitude (LLA)
    - Azimuth and Elevation (AZ/EL)
    - Two Line Elements (TLE).

12. References

Aerospace requires the following tech stack for this project:

- Leaflet.js Interactive Map API : https://leafletjs.com
- InfluxDB Version 1: https://docs.influxdata.com/influxdb/v1.8/guides/
- Almanac data that is publicly provided by the NAVCEN website at: https://www.navcen.uscg.gov/gps-nanus-almanacs-opsadvisories-sof .