Gravity Waves Analysis Tool

Software Requirements Specification (SRS)

This document is an annotated outline intended for specifying software requirements and is adapted from IEEE 29148-2018.

Version 3.1
Prepared By: Thomas Marten, Duncan Zaug, Daniel Maslowski, Justyce Countryman, Timothy Dube
Prepared For: SUNY Oswego
05/01/24

Table of Contents

Change Log

Name	Date	Reason For Changes	Version
Thomas Marten	01/31/2024	Initial entry of data	1
Thomas Marten	02/01/2024	Entry of acronyms	1
Daniel Maslowski	02/01/2024	Entry of introduction and purpose	1
Duncan Zaug	02/01/2024	Entry of product functions, user characteristics, and limitations. Updated and added a couple of definitions and acronyms.	1
Thomas Marten	02/02/2024	Entry of references, editing spelling, grammar, semantics, updating references, moving info to correct tables.	1

Timothy Dube	02/02/2024	Reviewing IEEE document. Checking for spelling errors and missed abbreviations.	1
Justyce Countryman	02/02/2024	Second draft of sections 1.3.1 to 1.3.4 References that were used are in section 2. **TBD** definition is in section 1.4.	1
Justyce Countryman	02/09/2024	Added use case and sequence diagrams between appendix A and B.	1.5
Daniel Maslowski	02/12/2024	Added the updated Use Case and sequence diagrams between appendix A and B.	1.5
Daniel Maslowski	02/16/2024	Added the UML class diagram between appendix A and B.	2.0
Daniel Maslowski	02/23/2024	Added the Activity diagrams between appendix A and B.	2.0
Daniel Maslowski	02/26/2024	adjusted the “Graph Creator” Activity diagram between appendix A and B.	2.0
Thomas Marten	04/18/2024	Updated document	2.1
Daniel Maslowski	04/29/2024	Updated Activity diagrams	3.0
Daniel Maslowski	04/30/2024	Worked on section 3.2, 3.4, and 3.6	3.0
Thomas Marten	04/29/2024	Updated all diagrams Worked on sections 1.0, 3.0, 4.0	3.0
Justyce Countryman	04/29/2024	Updated all diagrams Worked on sections 1.0, 3.0, 4.0	3.0
Daniel Maslowski	05/01/2024	Worked on sections 4.1 and 4.4	3.0

Justyce Countryman	05/01/2024	Completed the remaining sections, proofread and edited all previously completed sections, and updated the table of contents page.	3.1

1. Introduction

[This section of the document is intended to give a wide overview of the project, not identify specific solutions. If a section does not apply, ***do not remove it from the document,*** *instead, write a brief note on why the section does not apply to the SUD.]*

The gravity waves analysis project is envisioned to be a web application that can handle radiosonde data text files and provide downloadable output PDF to detect the presence of gravity waves. The webpage will handle anywhere from one to thousands of users at a given moment with a “click-and-forget” graphical user interface.

1.1 Purpose

[Identify the primary reason for the SRS, the system, and the primary audience of the SRS and the system, if they are different. Keep it as short as possible, you will elaborate in future sections (Moore’s Vision Statement).]

For individuals with some collegiate level education or higher in atmospheric sciences or physics who would like to view the effects of gravity waves in radiosonde data. The gravity waves analysis project is a software application that will add a user-friendly interface to process radiosonde data and observe the presence of gravity waves. Unlike the current competitor IDL which is costly and requires extensive knowledge of IDL code, our product will be an open-source, free-to-use software that can efficiently parse through radiosonde data and provide downloadable results with no prior knowledge of IDL code and programming required.

1.2 Scope

[Identify the scope of the SRS document, what will be covered, what will be excluded? Additionally, identify the general scope of the system you are building. Keep these brief, use bullet points to help organize the information.]

- This document will organize and track the requirements of the gravity waves project.
- This document will not include anything about system architecture or detailed designs.
- This project will create a user-friendly way of interacting with IDL code.

- This project will allow for uploading, downloading, and analysis of radiosonde data in standard IO formats.
- This project will have a simple UI that is click and forget, with menu options for all commonly done tasks in the analysis of radiosonde data.

1.3 Product Overview

1.3.1 Product Perspective

[Identify the product's relationships with other products already in existence or currently in development. If this product is part of a larger system, identify that here and notate how the system will interact with external actors.]

This product will work in conjunction with *GDL - GNU Data Language*, which is an open-source version of IDL (*GDL - GNU Data Language*, 2023). This GDL code will be used to accept a given radiosonde data file as input, perform calculations, and display values through graphical means as output that will aid in describing gravity waves. The project will create and display sixteen 2D graphs (two of which are hodographs) for several x- and y-axis variables. These variables are temperature, altitude, wind speed and direction (meridional wind component and zonal wind component), and pressure. A graph for the weather balloon path is also included. The frontend and backend will communicate through a shared file space using Flask. This allows for easy file transfer and single handling and sending between the frontend and backend.

1.3.1.1 System Interfaces

[List the system interfaces and identify the functionality of that system. How will it be used in conjunction with the system you are developing?]

This system has one simple web-based interface, with four buttons, one for each action: The “Upload File” button (to upload a text file containing radiosonde data), the “Download PDF” button (to open a new tab where you can download the PDF), the “Preview PDF” button (to preview the PDF), and “Authors Page” (to provide contact information and summaries of the authors).

1.3.1.2 User Interfaces

[List the characteristics of each interface between the product and its users. Include the modes of operation in the user organization, data processing support functions, and any backup or data recovery operations present in the system.]

The primary user interface will include “easy-to-navigate” graphical user interface menus to allow the user to upload a text file, and then confirm the input text file with a submit button. The project will allow the user to read and download the PDF containing sixteen 2D graphs, a graph of the weather balloon path, and the output parameters of GDL based on the given data. Text files will be the only supported file format for uploading radiosonde files and PDF files will be the only permitted format for downloading the graphs and GDL parameters.

1.3.1.3 Hardware Interfaces

[Specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics. It also covers matters such as what devices are to be supported, how they are to be supported, and any relevant protocols.]

The user's computer is expected to have a CPU that can handle multiple sequential mathematical operations at a time and can download and display 2D graphs to the user’s operating system. The type and version of the operating system do not matter as long as it has access to the Internet and some modern search engines, particularly Google.

1.3.1.4 Software Interfaces

[List the name, mnemonic, specification number, version number, and source of each software product, be sure to include the purpose of each software product you are integrating and how it will be incorporated into the SUD.]

GDL - GNU Data Language, Version 1.0.4,
https://github.com/gnudatalanguage/gdl (1)

- Will be used when data input is inserted and submitted to perform computations that provide analysis regarding gravity wave characteristics. The calculated values will communicate with the graphing scripts. The Graphing Software is a bespoke program customized to make the 9 standard graphs provided by Dr. Jie Gong.

PyCharm’s Python Version 3

https://www.jetbrains.com/pycharm/ (2)

- For compatibility with Docker and GDL which was installed with PyCharm. Python was used to create the seventeen graphs that are displayed in the UI. It was also used to connect the frontend with the backend using Flask as the middleware.

Software applications mentioned in this document, but not included in this section, are not currently expected to be integrated into the system.

1.3.1.5 Communication Interfaces

[Identify the various networking protocols and other such communication interfaces. Unique and non-standard protocols should be documented here, standard protocols should be referenced and cited here and in the reference document section.]

- Flask 3.0.2
- Docker 25.0.3

1.3.1.6 Memory Constraints

[List any limitations on primary and secondary memory, as well as any applicable characteristics.]

- The user's machine requires enough memory to run a browser window.
- Internet connection

1.3.1.7 Site Adaptation

[Identify any situations, initializations, or protocols that are specific to a given site, mission, or operational model. If you are tailoring a product to meet specific client needs that differ from normal operation, include it in this section.]

- There are no bespoke site adaptations at this time.

1.3.1.8 Interfaces with Services

[List any interaction with any SAAS (Software as a Service) or cloud-based services.]

- The web application interacts with the Moxie server site. Access to this server was granted by Dr. Bastian Tenbergen.

1.3.2 Product Functions

This product will take a raw radiosonde data file in the form of a text file and then generate important parameters from the data. These parameters will be used for meteorological calculations and graphs of the data to allow for gravity wave analysis. This product assumes that users do proper quality control of the data before entry into the program (removing invalid data and providing radiosonde files that do not have chunks of missing data). That can be displayed or downloaded in a PDF format. The UI for this program is simple and easy to use.

1.3.3 User characteristics

The users of this product are professors and undergraduate or higher students in the field of meteorology and physics. These users are not expected to have expansive, in-depth experience with computers and programming. This could potentially force us to create a cloud/website-based program rather than an installable software. This also means it cannot be a command line-based program as they are not familiar with that environment. Some of our more experienced users, however, will usually have training on how to get radiosonde data out of some rather dated hardware.

1.3.4 Limitations

Our web program's only limitation is having an internet connection, and enough memory to launch a browser. The user must also have enough space on their computer to download the PDF document.

1.4 Definitions

[Include any terms and definitions needed to understand the SRS or the System Under Development (SUD), terms placed here should also be placed in the Appendix. Terms should use the full name and the general definition of that term, any abbreviations that will be used in the document and the source should be placed in Appendix 5.2]

Term	Definition
Example: *Context of use*	*Users, tasks, equipment (hardware, software, and materials), and the physical and social environments in which a product is used.*
System Under Development (SUD)	The system which is actively being developed.

Radiosonde	A sensor attached to weather balloons to collect a vertical profile of the atmosphere.
Click-and-forget	A style of design where the user does not need multiple inputs to get the needed output.
Gravity waves	Atmospheric phenomena where air is perturbed in a relatively stable environment due to a disturbance (i.e. convection, orographic lift).

2. References

[Include citations to external sources and resources in this section. References to other internal documents can be placed here but should also be referenced in the appendix.
Example: ISO/IEC/IEEE 29148.2018, Systems and software engineering
Life cycle processes — Requirements engineering]

The GDL Team. (2024, January 30). *GDL - GNU Data language*. GDL - GNU Data Language.

https://gnudatalanguage.github.io/

This source introduces the reader to the primary purposes, system capabilities, steps for installation based on specific operating systems and versions, optional and mandatory downloadable dependencies, and links to transfer the reader to the GDL GitHub repository and documentation concerning IDL. Since the GDL project and documentation are declared as open-source under the GNU General Public License, we have the permission to "run, study, share, and modify" the software ("GNU General Public License," n.d., para. 1).

Wikimedia Foundation. (2024, January 26). GNU General Public License.

https://en.wikipedia.org/wiki/GNU_General_Public_License

Products. NV5 Geospatial. (n.d.). https://www.nv5geospatialsoftware.com/Products/IDL

GDL - GNU Data Language. (n.d.). Gnudatalanguage.github.io. Retrieved February 2, 2024,

from https://gnudatalanguage.github.io/

Flask Version: https://flask.palletsprojects.com/en/3.0.x/

Docker Version: https://docs.docker.com/reference/cli/docker/version/

Python Cartopy Library: https://github.com/SciTools/cartopy

3. Requirements

[This section should contain all the software requirements at a level of detail sufficient enough to enable designers to design a system, and for testers to test that system, in a way that satisfies the requirements. Each requirement should be perceivable by users, operators, or other external systems. At minimum, the description should include the inputs and outputs of the system, and all functions performed by the system in response to an input or in support of an output. Specific requirements should include the following characteristics:

- *Correct*
- *Unambiguous*
- *Complete*
- *Consistent*
- *Verifiable*
- *Modifiable*
- *Traceable*
- *Ranked for importance and/or stability*
- *uniquely identifiable (usually by numbering)*
- *organized in a way that allows for maximum readability*

The purpose of the requirement is not to dictate design, but rather to guide designers to make the safest, most correct version of the system possible. Do not attempt to build solutions to your written requirements]

3.1 Functions

[Define the fundamental actions that the system must stake in order to accept inputs and generate outputs. It may make sense to organize or partition the functional requirements into sub-functions or sub-processes, do not expect development to mimic this organization. Functional requirements are typically identified using "shall" statements, and some functions that should be included in this section are:

- *Validity checks on the inputs*
    - "The system will check the input data file to see if it received a .txt file."
    - "The system will verify if the input data file contains entirely valid data."
- *Exact sequence of operations*
    - "Firstly, the system shall receive a valid radiosonde file."
    - "Then, the system shall create the required graphs based on the radiosonde file data."
    - "Then, the system must pass the radiosonde file to GDL and obtain GDL output parameters."
    - "Finally, the system shall generate the PDF containing the graphs and GDL output parameters."

- *Responses to abnormal situations including:*
    - *Error Handling and Recovery*
        - "The system will respond appropriately to invalid radiosonde files and it shall communicate this information effectively to the user."
            - "The system shall tell the user if the file is invalid."
            - "The system shall tell the user if the file could not be parsed."
            - "The system shall tell the user if no radiosonde file was attached when the user attempts to upload a file."
- *Relationships of outputs to inputs*
    - "The system shall generate a PDF containing graphs and GDL parameters with respect to the valid radiosonde input file."

3.2 Performance Requirements

[In measurable terms, specify the numerical requirements of the system. Include static performance requirements such as the number of terminals, simultaneous users, etc. As well as dynamic performance requirements such as the number of tasks able to be completed in a set period of time.]

The number of simultaneous users depends on the server hosting the software. In preliminary testing, less than 10 users were able to properly use the website out of roughly 15 users simultaneously attempting to. This led to further analysis concerning what is occurring with Docker on Moxie. A change in the amount of threads was implemented to attempt to solve this problem, but another round of user testing has not yet been conducted to see if this issue still persists since we will discuss with Dr. Gong continuing this project over the summer to iron it out. The same requirements as the ones in sections 1.3.1.3 and 1.3.1.6 apply here as well.

3.3 Usability Requirements

[Define usability and quality requirements that are measurable in effectiveness, efficiency, satisfaction, and in avoidance of harm that could arise from specific use cases.]

The project will only be fully functional when provided a valid radiosonde file with a format that meets one of the formats listed on the web application's "Files We Support" page. If one is provided, the application will run and display the desired PDF with all of the graphs. The final page of the PDF may or may not contain the calculated GDL output parameters. The parameters are only available when there is enough weather balloon data that covers enough area for GDL to determine reasonable parameter values, oftentimes based on the latitude, longitude, and altitude values of the radiosonde file.

3.4 Interface Requirements

[List all inputs and outputs from the system. It should mirror but not repeat the information found in sections 4.2- 4.6. For each defined interface, be sure to include:

- *the name of the item*
- *description of the purpose of the interface*
- *source of input OR output destination*
- *range, accuracy and/or tolerance*
- *units of measurement*
- *timing*
- *I/O relationships*
- *data formats*
- *command formats*

any information included within the I/O.]

website.html: The html serving as the homepage for the website. In Flask, the website is routed from the "templates" directory of the project folder. It uses scripts.js to receive text files and display PDFs. The images and icons in the html were pngs resized using px pixels as well as vh (viewport height).
authors.html: The html serving as the static Authors page. In Flask, the website is routed from the "templates" directory of the project folder. The information in this page is static. The images and icons in the html were pngs resized using px pixels as well as vh (viewport height).
styles.css: The css file used to format Website.html. It uses px (pixel) as the unit of measurement as well as vh (viewport height).
script.js: The javascript file used for functionality of the website (allows Flask to determine if an endpoint has been triggered).
terminal_interface.py : The backend scripts used for taking in a Radiosonde text file and outputting a formatted PDF file that is displayed on the website. The input comes from the script.js interface and the output is given to the script.js interface. This interface takes about 30-50 seconds to complete its run from the second the Radiosonde text file is passed in to returning of the PDF.

3.5 Logical Database Requirements

- This product has no databases, so there are no logical database requirements.

3.6 Design Constraints

[List any constraints on the system. These constraints should be from external sources such as regulatory standards, legal, or project limitations.]

- There are 3 file formats the project can handle, two radiosonde files provided by Dr. Gong and 1 provided by Dr. Barber. Snippets for each format are provided in section 3.8.

- File format 1 where the radiosonde contains the words "Profile Data:", has the variable names in the first line below it, the units in the second line below it, and the data in the third line and onwards. The order of and the amount of variables does not matter since the program will determine that dynamically. The variables that MUST be present however are Time "Time" (or time as "UTC" or "UTC Time"), Temperature "T", Wind Speed "Ws", Wind Direction "Wd", Longitude "Long.",Latitude "Lat.", and Altitude "Alt". If the file contains these variables but with a different abbreviation, it must be changed to the abbreviations stated in quotation marks in the previous sentence.
- File format 2 is where the radiosonde contains the variable names in the first line and the data in the second line and onwards. The order of and the amount of variables does not matter since the program will determine that dynamically. The variables that MUST be present however are Time "Time" (or time as "UTC"), Temperature "T", Wind Speed "Ws", Wind Direction "Wd", Longitude "Long.",Latitude "Lat.", and Altitude "Alt". If the file contains these variables but with a different abbreviation, it must be changed to the abbreviations stated in quotation marks in the previous sentence.

3.7 Software System Attributes

[For each of the attributes of the software system (Reliability, Availability, Security, Maintainability, Portability, etc.), list the factors that will establish functionality or stability. For example, when establishing requirements for the Security attribute, you may include one that restricts communication between one area of the program and another.]

- *Reliability*
    - This software will check the input radiosonde file format and attempt to match it to a previously known format.
    - This software is robust for accepted radiosonde inputs and resulting outputs (examples are in the GitHub repository for this product at this link).
    - If an error occurs the system shall give an error message back to the user.
- *Availability*
    - This software is a website allowing any user with an Internet connection access to the website.
    - This software will be available on the Moxie Server and NASA server.
- *Security*
    - The website only takes in a text file as input. The file is checked on the website side and the Flask side to determine if the file meets the criteria of one of the accepted file formats/layouts.
    - Flask checks that the output file format is a PDF and only a PDF.
    - The Docker Compose confirms the internal network between the two Docker containers is using the bridge driver.
- *Maintainability*

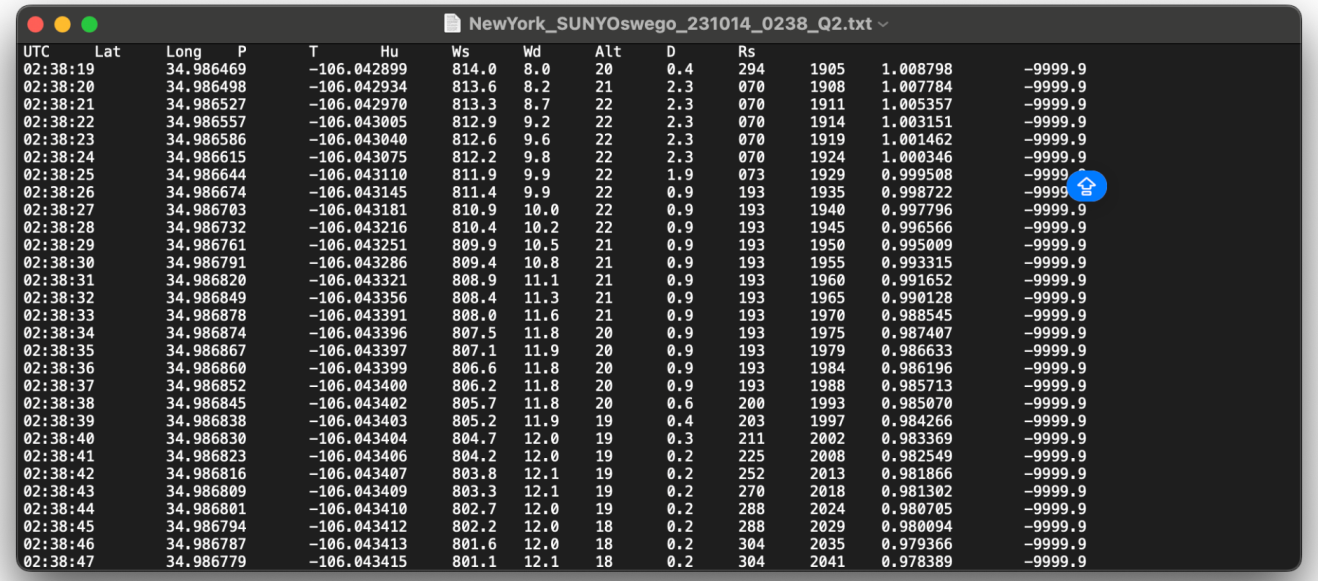
- The source code is public and efficiently documented online, allowing for easy and permissible changes.

- *Portability*
  - This software is deployed on Docker containers, which allows the software to be deployed on any device, regardless of operating system.

3.8 Supporting Information

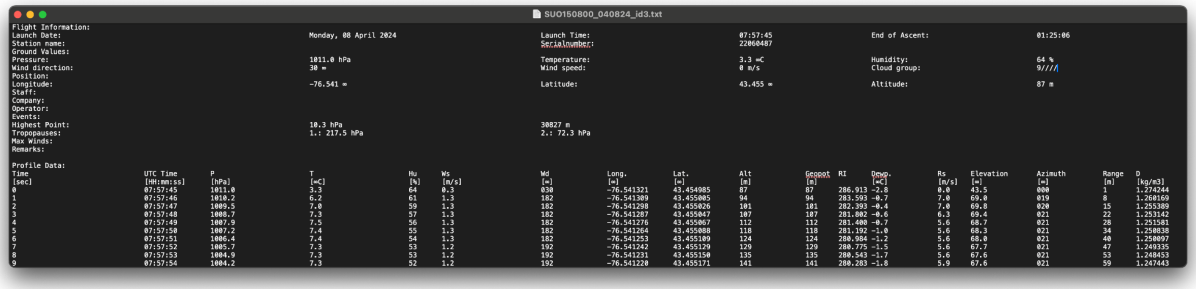
[Add any additional information needed to understand the SRS. Include things like background information, problem descriptions, packaging instructions for code, sample input/output formats, etc.]

Examples of Valid Radiosonde Input Files:



NewYork_SUNYOswego_231014_0238_Q2.txt

UTC	Lat	Long	P	T	Hu	Ws	Wd	Alt	D	Rs			
02:38:19		34.986469		-106.042899		814.0	8.0	20	0.4	294	1905	1.008798	-9999.9
02:38:20		34.986498		-106.042934		813.6	8.2	21	2.3	070	1908	1.007784	-9999.9
02:38:21		34.986527		-106.042970		813.3	8.7	22	2.3	070	1911	1.005357	-9999.9
02:38:22		34.986557		-106.043005		812.9	9.2	22	2.3	070	1914	1.003151	-9999.9
02:38:23		34.986586		-106.043040		812.6	9.6	22	2.3	070	1919	1.001462	-9999.9
02:38:24		34.986615		-106.043075		812.2	9.8	22	2.3	070	1924	1.000346	-9999.9
02:38:25		34.986644		-106.043110		811.9	9.9	22	1.9	073	1929	0.999508	-9999.9
02:38:26		34.986674		-106.043145		811.4	9.9	22	0.9	193	1935	0.998722	-9999.9
02:38:27		34.986703		-106.043181		810.9	10.0	22	0.9	193	1940	0.997796	-9999.9
02:38:28		34.986732		-106.043216		810.4	10.2	22	0.9	193	1945	0.996566	-9999.9
02:38:29		34.986761		-106.043251		809.9	10.5	21	0.9	193	1950	0.995009	-9999.9
02:38:30		34.986791		-106.043286		809.4	10.8	21	0.9	193	1955	0.993315	-9999.9
02:38:31		34.986820		-106.043321		808.9	11.1	21	0.9	193	1960	0.991652	-9999.9
02:38:32		34.986849		-106.043356		808.4	11.3	21	0.9	193	1965	0.990128	-9999.9
02:38:33		34.986878		-106.043391		808.0	11.6	21	0.9	193	1970	0.988545	-9999.9
02:38:34		34.986874		-106.043396		807.5	11.8	20	0.9	193	1975	0.987407	-9999.9
02:38:35		34.986867		-106.043397		807.1	11.9	20	0.9	193	1979	0.986633	-9999.9
02:38:36		34.986860		-106.043399		806.6	11.8	20	0.9	193	1984	0.986196	-9999.9
02:38:37		34.986852		-106.043400		806.2	11.8	20	0.9	193	1988	0.985713	-9999.9
02:38:38		34.986845		-106.043402		805.7	11.8	20	0.6	200	1993	0.985070	-9999.9
02:38:39		34.986838		-106.043403		805.2	11.9	19	0.4	203	1997	0.984266	-9999.9
02:38:40		34.986830		-106.043404		804.7	12.0	19	0.3	211	2002	0.983369	-9999.9
02:38:41		34.986823		-106.043406		804.2	12.0	19	0.2	225	2008	0.982549	-9999.9
02:38:42		34.986816		-106.043407		803.8	12.1	19	0.2	252	2013	0.981866	-9999.9
02:38:43		34.986809		-106.043409		803.3	12.1	19	0.2	270	2018	0.981302	-9999.9
02:38:44		34.986801		-106.043410		802.7	12.0	19	0.2	288	2024	0.980705	-9999.9
02:38:45		34.986794		-106.043412		802.2	12.0	18	0.2	288	2029	0.980094	-9999.9
02:38:46		34.986787		-106.043413		801.6	12.0	18	0.2	304	2035	0.979366	-9999.9
02:38:47		34.986779		-106.043415		801.1	12.1	18	0.2	304	2041	0.978389	-9999.9



SUO150800_040824_id3.txt

Flight Information:

Launch Date:	Monday, 08 April 2024	Launch Time:	07:57:45
Station name:		Serialnumber:	22060487
Ground Values:			
Pressure:	1011.0 hPa	Temperature:	3.3 ∞C
Wind direction:	30 ∞	Wind speed:	0 m/s
Position:			
Longitude:	-76.541 ∞	Latitude:	43.455 ∞
Staff:			
Company:			
Operator:			
Events:			
Highest Point:	10.3 hPa		30827 m
Tropopauses:	1.: 217.5 hPa		2.: 72.3 hPa
Max Winds:			
Remarks:			

End of Ascent:	01:25:06
Humidity:	64 %
Cloud group:	9////
Altitude:	87 m

Profile Data:

Time [sec]	UTC Time [HH:mm:ss]	P [hPa]	T [∞C]	Hu [%]	Ws [m/s]	Wd [∞]	Long. [∞]	Lat. [∞]	Alt [m]	Geopot [m]	RI	Dewp. [∞C]	Rs [m/s]	Elevation [∞]	Azimuth [∞]	Range [m]	D [kg/m3]
0	07:57:45	1011.0	3.3	64	0.3	030	-76.541321	43.454985	87	87	286.913	-2.8	0.0	43.5	000	1	1.274244
1	07:57:46	1010.2	6.2	61	1.3	182	-76.541389	43.455085	94	94	283.593	-0.7	7.0	69.0	019	8	1.260169
2	07:57:47	1009.5	7.0	59	1.3	182	-76.541298	43.455026	101	101	282.393	-0.4	7.0	69.0	020	15	1.255389
3	07:57:48	1008.7	7.3	57	1.3	182	-76.541287	43.455047	107	107	281.802	-0.6	6.3	69.4	021	22	1.253142
4	07:57:49	1007.9	7.5	56	1.3	182	-76.541276	43.455067	112	112	281.400	-0.7	5.6	68.7	021	28	1.251581
5	07:57:50	1007.2	7.4	55	1.3	182	-76.541264	43.455088	118	118	281.102	-1.0	5.6	68.3	021	34	1.250838
6	07:57:51	1006.4	7.4	54	1.3	182	-76.541253	43.455109	124	124	280.904	-1.2	5.6	68.0	021	40	1.250097
7	07:57:52	1005.7	7.3	53	1.2	192	-76.541242	43.455129	129	129	280.775	-1.5	5.6	67.7	021	47	1.249335
8	07:57:53	1004.9	7.3	53	1.2	192	-76.541231	43.455150	135	135	280.543	-1.7	5.6	67.6	021	53	1.248453
9	07:57:54	1004.2	7.3	52	1.2	192	-76.541220	43.455171	141	141	280.283	-1.8	5.9	67.6	021	59	1.247443

4. Verification

[List all inputs and outputs from the system. It should mirror but not repeat the information found in sections 3.1- 3.8. For each defined interface, be sure to include:

- *the name of the item*
- *description of the purpose of the interface*
- *source of input OR output destination*
- *range, accuracy and/or tolerance*
- *units of measurement*
- *timing*
- *I/O relationships*
- *data formats*
- *command formats*
- *any information included within the I/O.]*

- Input: Radiosonde file
    - Must be a format that the web application can handle and produce all common graphs for.
    - Provided by the user when the user presses “Upload Button” and chooses the file from a file dialog.
    - Input radiosonde file will be sent to a hidden temporary directory and removed at the completion and displaying of the PDF graph.
    - Tolerance will be that any radiosonde file that has the label “Profile Data:” on the first line, followed by the units on the next line, and the data on the third line and onwards. The order and quantity of the data does not have to be in order. However, the units of Time (“Time”, “UTC”, or “UTC Time”), Temperature (“T”), Wind speed (“Ws”), Wind direction (“Wd”), Longitude (“Long.”), Latitude (“Lat.”), and Altitude (“Alt”) need to be present. The unit abbreviations represented in the quotations are also required.
    - The web application server will begin processing the data from the radiosonde file immediately upon retrieval. However, completing the data processing may take approximately 30 seconds.
- Output: PDF file
    - Will contain sixteen 2D graphs for specific variables with respect to the radiosonde file data for both the stratosphere and troposphere, a graph that has the weather balloon path, and GDL output parameters. The current date at the time of processing the radiosonde file will be on the first page, a weather balloon path on the second page, the sixteen 2D graphs on pages 3-18, and formatted GDL output parameters on page 19, the final page.

- Will be displayed to the user as soon as the data from the radiosonde file is done processing. Once completed, the user will be automatically transferred to the "Preview PDF" page.

4.1 Functions

[See sections 4.0 and 3.1 for specific directions about what outputs should be included here.]

The system must be able to dynamically recognize endpoints and, based on those endpoints, trigger functions in the app.py using Flask. The first endpoint is for uploading a file where the user's action of pressing upload and providing a valid text file will trigger Flask to run a function that both deletes any previous textfiles, deletes previous pdfs, and runs a command that communicates to the backend that it must run the python program (terminal_interface.py) with the user provided text file to be processed. The system must also trigger functions at the endpoint for previewing and downloading pdfs so it will retrieve the processed pdf and display it to the UI.

4.2 Performance Requirements

[See sections 4.0 and 3.2 for specific directions about what outputs should be included here.]

The web application requires a system with a minimum of 8 CPU cores, along with a minimum of 128GB of available storage. However, as the number of users of the Gravity Wave Analysis Tool increases, so does the minimum storage requirement. The system must have at least Docker version 2.2.3 installed.

4.3 Usability Requirements

[See sections 4.0 and 3.3 for specific directions about what outputs should be included here.]

The same input and output requirement descriptions from sections 4.0 and 3.3 apply to this section as well. These usability requirements have been manually tested and are confirmed to be satisfied.

4.4 Interface Requirements

[See sections 4.0 and 3.4 for specific directions about what outputs should be included here.]

The interface requires HTML for website structure, CSS3 for formatting, and JavaScript for functionality (allowing users to upload files when pressing the "Upload File" button, download PDFs with the "Download PDF" button, etc.). Flask is required, acting as a middleware to connect the frontend to the backend by triggering endpoints when the user interacts with the UI

buttons. The files used in the project are website.html, authors.html, styles.css, script.css, and terminal_interface.py which are in section 3.4.

4.5 Logical Database Requirements

- This product has no databases, so there are no logical database requirements.

4.6 Design Constraints

[See sections 4.0 and 3.6 for specific directions about what outputs should be included here.]

Due to the interface using only default HTML5, CSS3, and JavaScript without additional libraries, design is limited to their respective capabilities.

4.7 Software System Attributes

[See sections 4.0 and 3.7 for specific directions about what outputs should be included here.]

The same input and output requirement descriptions from sections 4.0 and 3.7 apply to this section as well. All software system attributes have been manually tested and are confirmed to be satisfied.

4.8 Supporting Information

[See sections 4.0 and 3.8 for specific directions about what outputs should be included here.]

The same input and output requirement descriptions from sections 4.0 and 3.8 apply to this section as well. All provided radiosonde file format examples from section 3.8 are verified to be satisfied. However, be advised that the product may not process some radiosonde files if there is data that indicates unusual meteorological patterns, even if the format is valid. These patterns may include, but are not limited to, significant altitude drops in a short time interval, major gaps of missing data spanning multiple minutes indicating a radiosonde file may have been stopped abruptly, or a lack of given numerical data for a specific variable in contrast to other data variables.

5. Appendix A – Tailoring Policies

5.1 Assumptions and Dependencies

[Identify any and all factors that may impact the implementation and execution of the requirements written below. These factors do not add a constraint but may impact development if they are changed. Example: a major update to an operating system(OS) on which the SUD is

intended to run impacts the implementation of one of the core features. The version of the OS that the system had intended to run on should be listed in this section.]

Unless an operating system, software, or hardware update on a user's computer prevents the utilization of the previously mentioned software from sections 4.2 and 4.4, or the GDL/IDL open-source code, then there are no immediate factors that could hinder the usage of the application.

5.2 Acronyms and Abbreviations

Term	Definition	Abbreviation	Source (if applicable)
Example: Context of use	*Users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used.*	*COU*	*[SOURCE: ISO/IEC 25000:2014, 4.2]*
System Under Development	The system which is actively being developed.	SUD	
Input and Output	The entry and exiting of data in the gravity waves project.	IO	
To Be Determined	To be determined.	TBD	
User Interface	User interface.	UI	
Interactive Data Language	A private industry standard interactive data language used by many professionals in the meteorological and physics fields.	IDL	*Products*. NV5 Geospatial. (n.d.). https://www.nv5geospatialsoftware.com/Products/IDL
GNU Data Language	An open source version of the interactive programming language.	GDL	*GDL - GNU Data Language*. (n.d.). Gnudatalanguage.github.io. Retrieved February 2, 2024, from

			https://gnudatalanguage.github.io/
Central Processing Unit	Central processing unit	CPU	
National Aeronautics and Space Administration	National Aeronautics and Space Administration	NASA	Wikipedia Contributors. (2019, January 15). *NASA*. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/NASA
GNU Not Unix	GNU not Unix	GNU	*GNU Full Form*. (n.d.). Unacademy. Retrieved February 2, 2024, from https://unacademy.com/content/cbse-class-11/full-forms/gnu-full-form/
Two dimensions	Two-dimensional, contains an x- and y-axis.	2D	
Portable Document Format	Portable document format	PDF	

5.3 Tailoring Policies

Tailoring is not a requirement to bring the document into compliance with the standards set by IEEE 29148-2018. Tailoring should only occur when conformance to the standard is not possible or practical. The act of tailoring is the modification and/or removal of one of the content sections outlined in this document, adding additional information items for organization is not considered tailoring. Tailoring should only occur when factors or circumstances:

- surround an organization that is using the document
- influence a project using this document to meet an agreement
- reflect the needs of an organization.

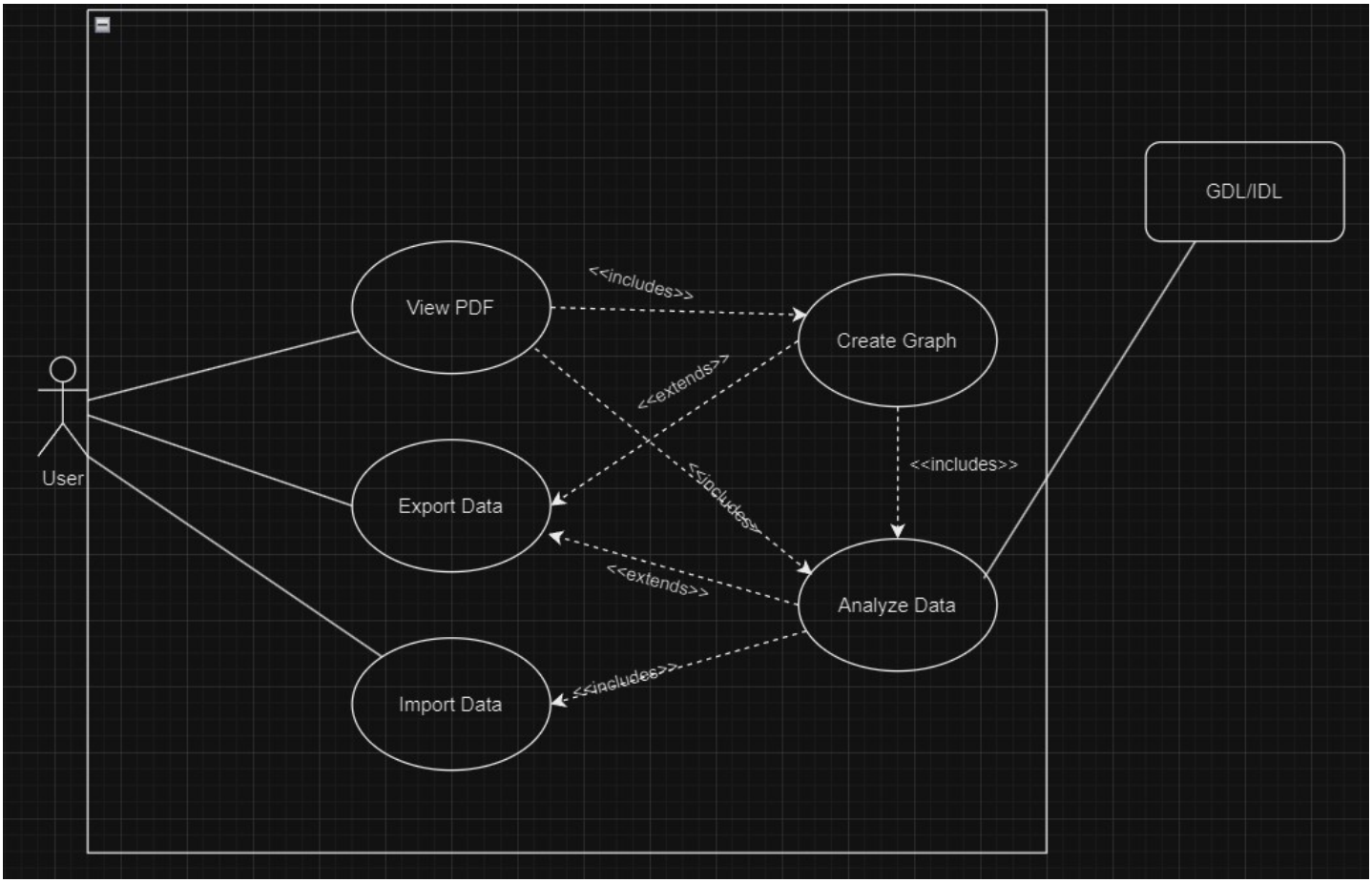
When tailoring the document, the following activities shall be implemented:

- Identify and document the circumstances that may influence tailoring.
    - novelty, size and complexity
    - stability of operating environments
    - variety in operating environments
    - starting date and duration

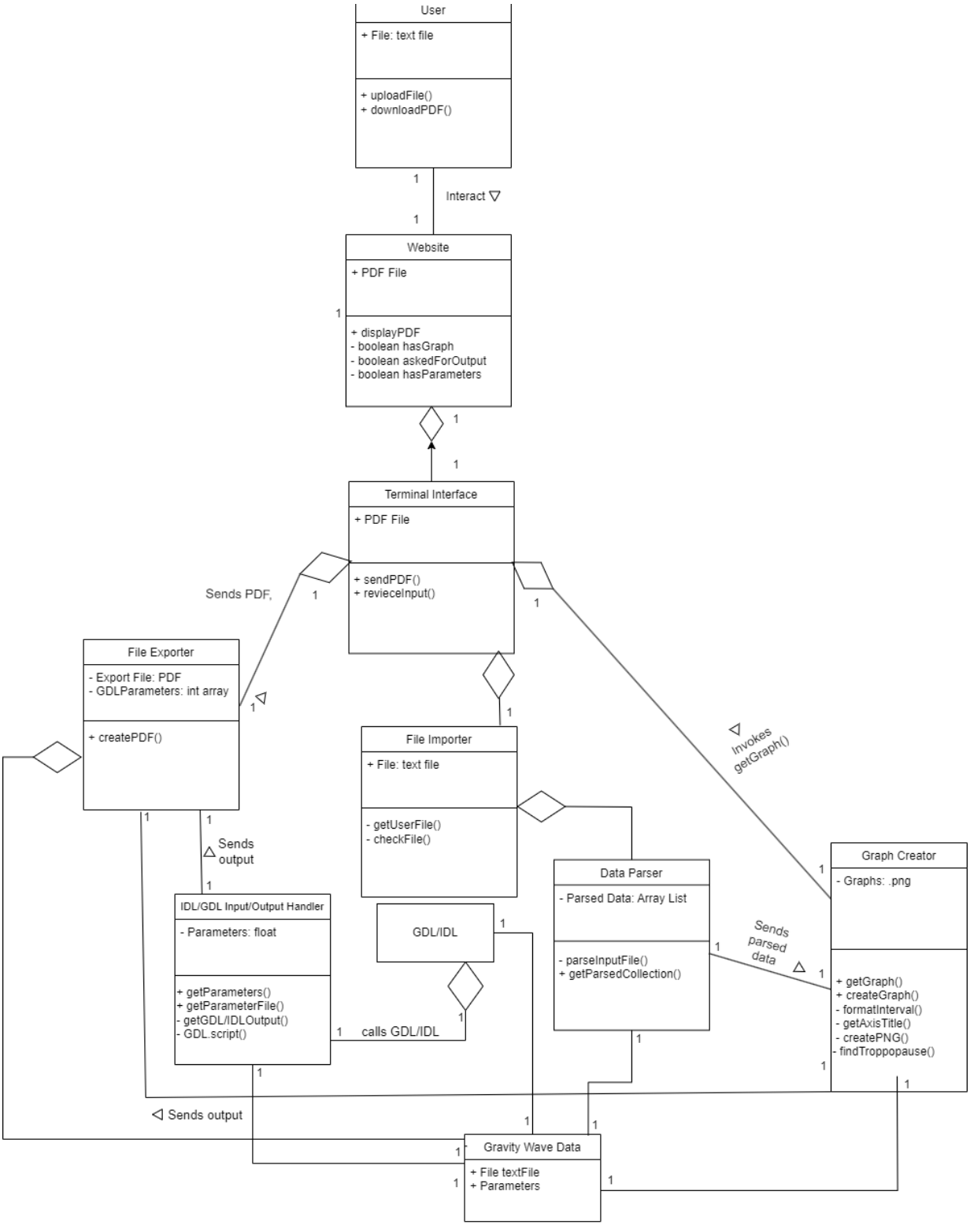
    - emerging technology
    - availability of services of enabling systems
    - other standards with which the document needs to conform.
- Identify and get input from all parties impacted by the tailoring process.
    - Such as stakeholders, contributors, and other interested parties
- Delete the information contents that require tailoring.

No tailoring policies have been required on this product at this time.

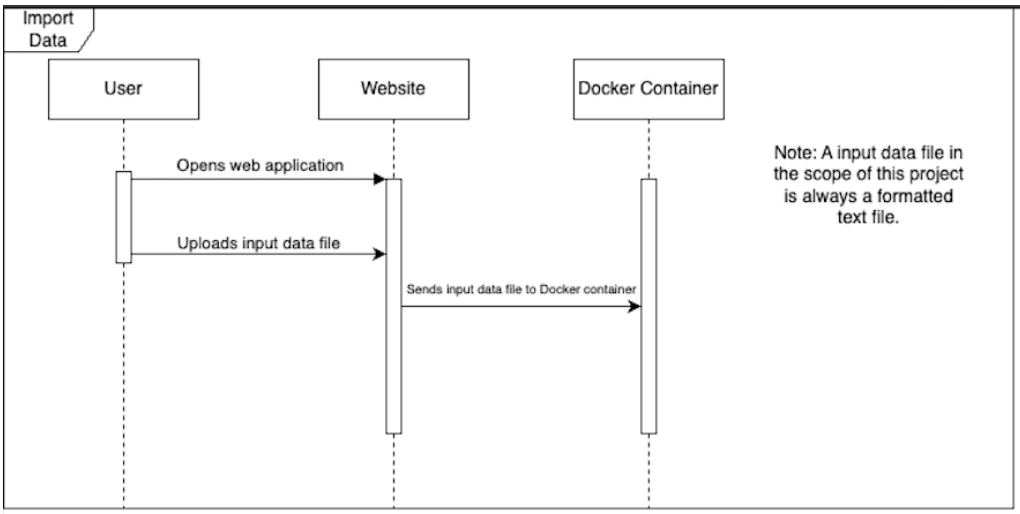
Use-Case Diagram:

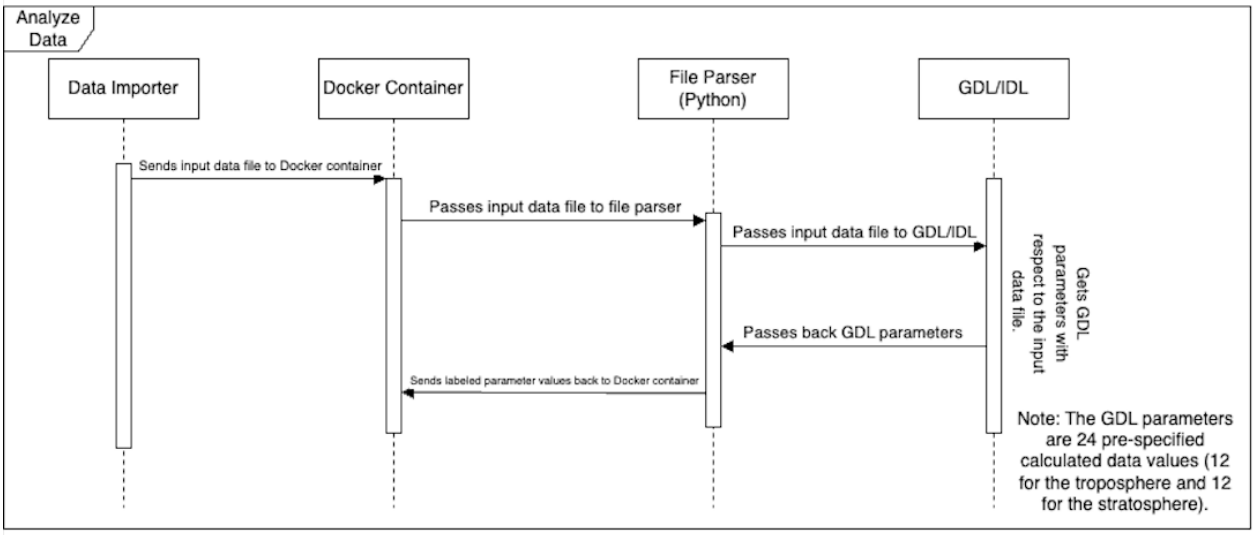


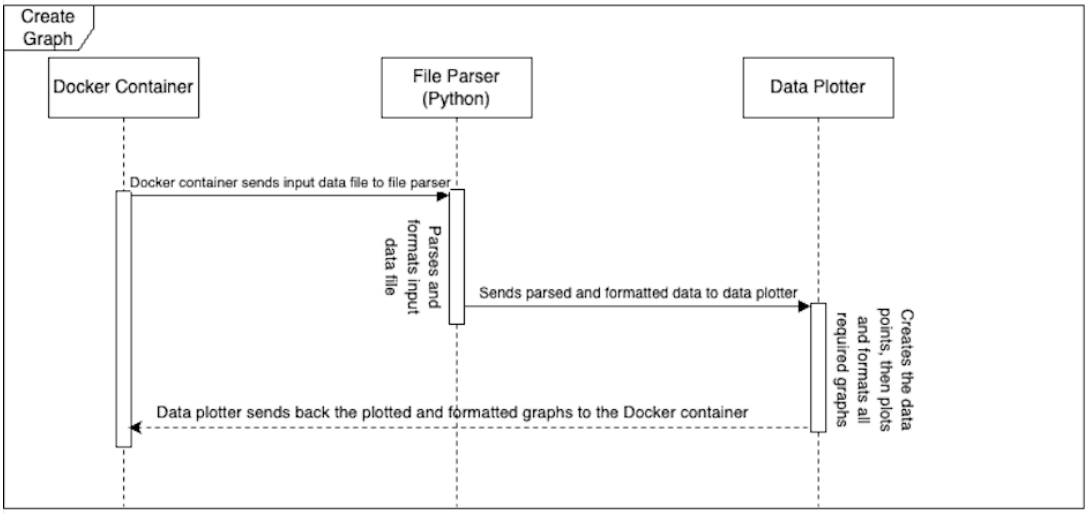
UML Class Diagram:

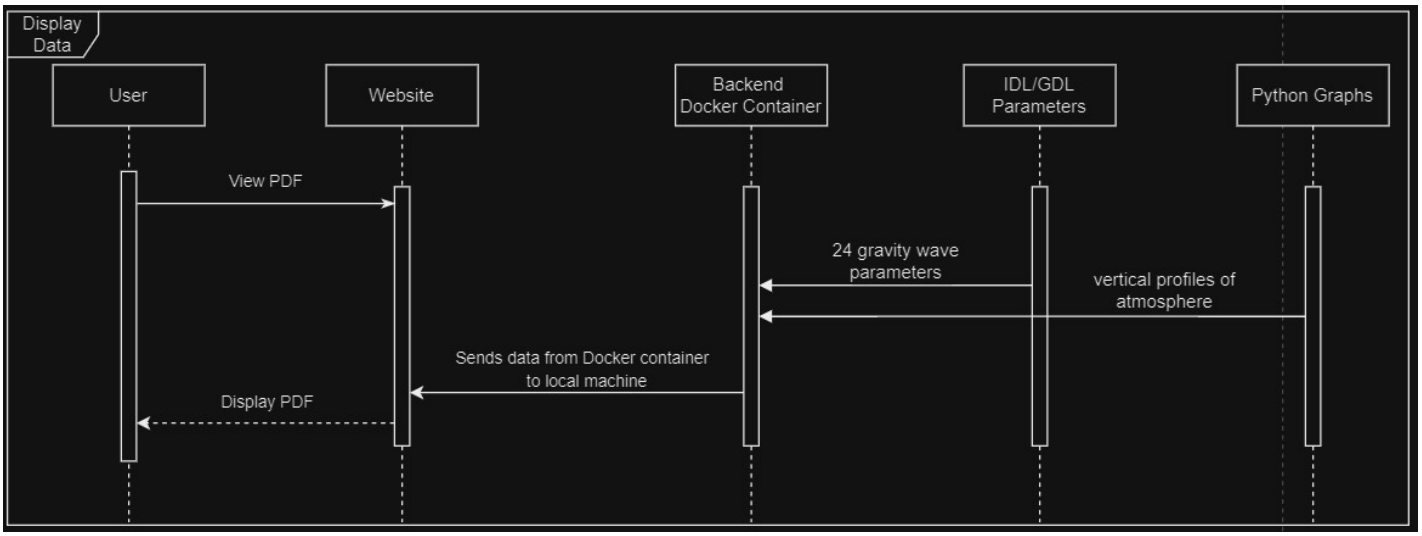


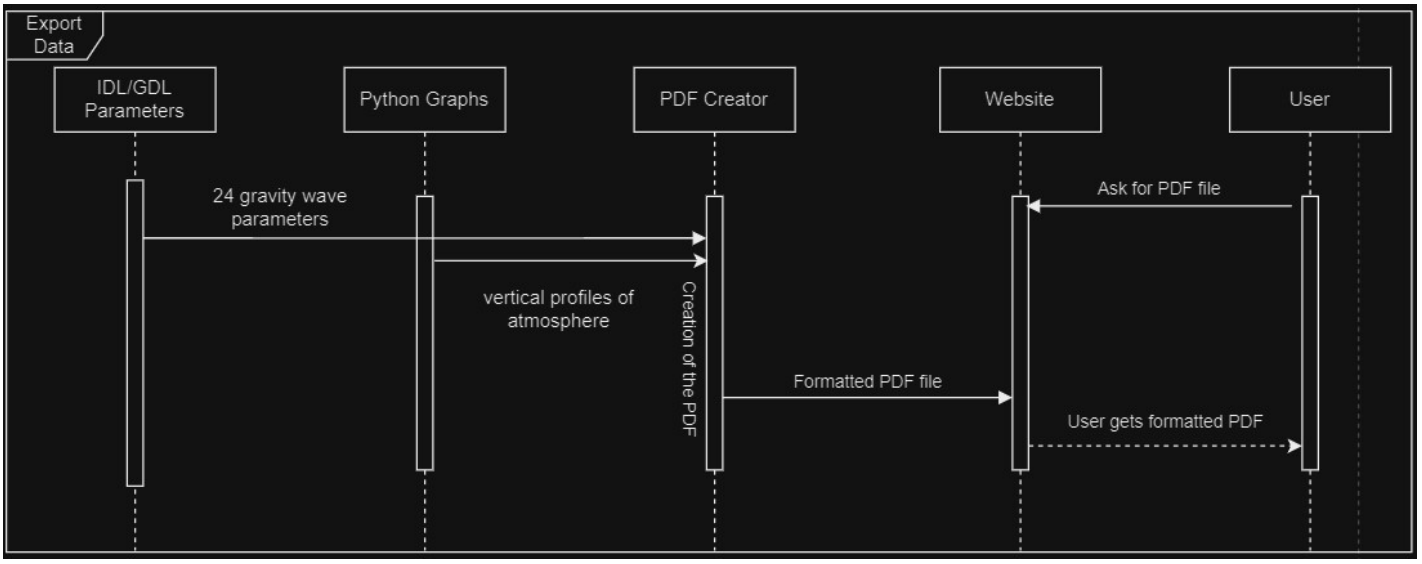
Sequence Diagrams:


Import Data
User
Website
Docker Container
Opens web application
Uploads input data file
Sends input data file to Docker container
Note: A input data file in the scope of this project is always a formatted text file.

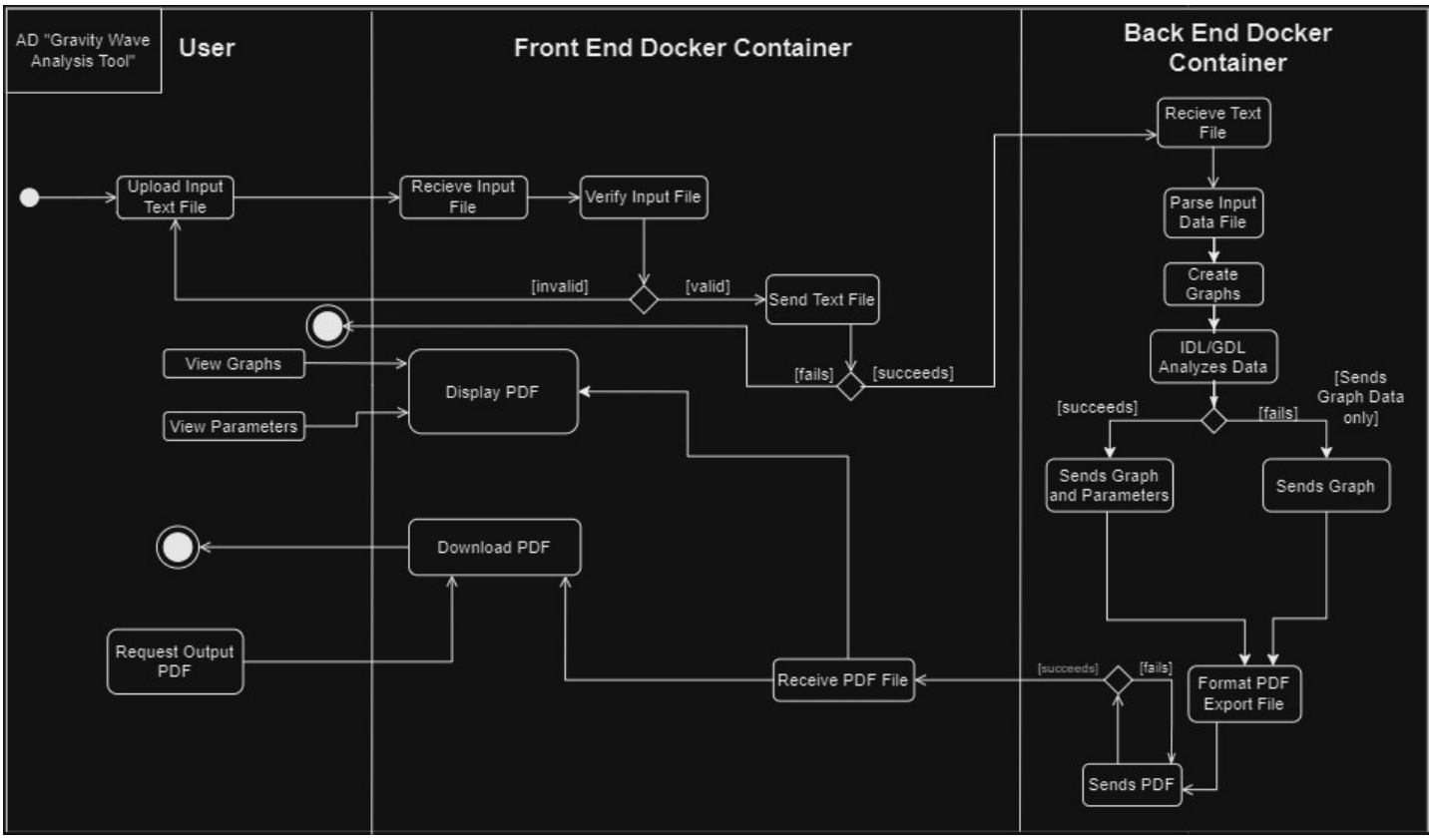

Analyze Data
Data Importer
Docker Container
File Parser (Python)
GDL/IDL
Sends input data file to Docker container
Passes input data file to file parser
Passes input data file to GDL/IDL
Gets GDL parameters with respect to the input data file.
Passes back GDL parameters
Sends labeled parameter values back to Docker container
Note: The GDL parameters are 24 pre-specified calculated data values (12 for the troposphere and 12 for the stratosphere).

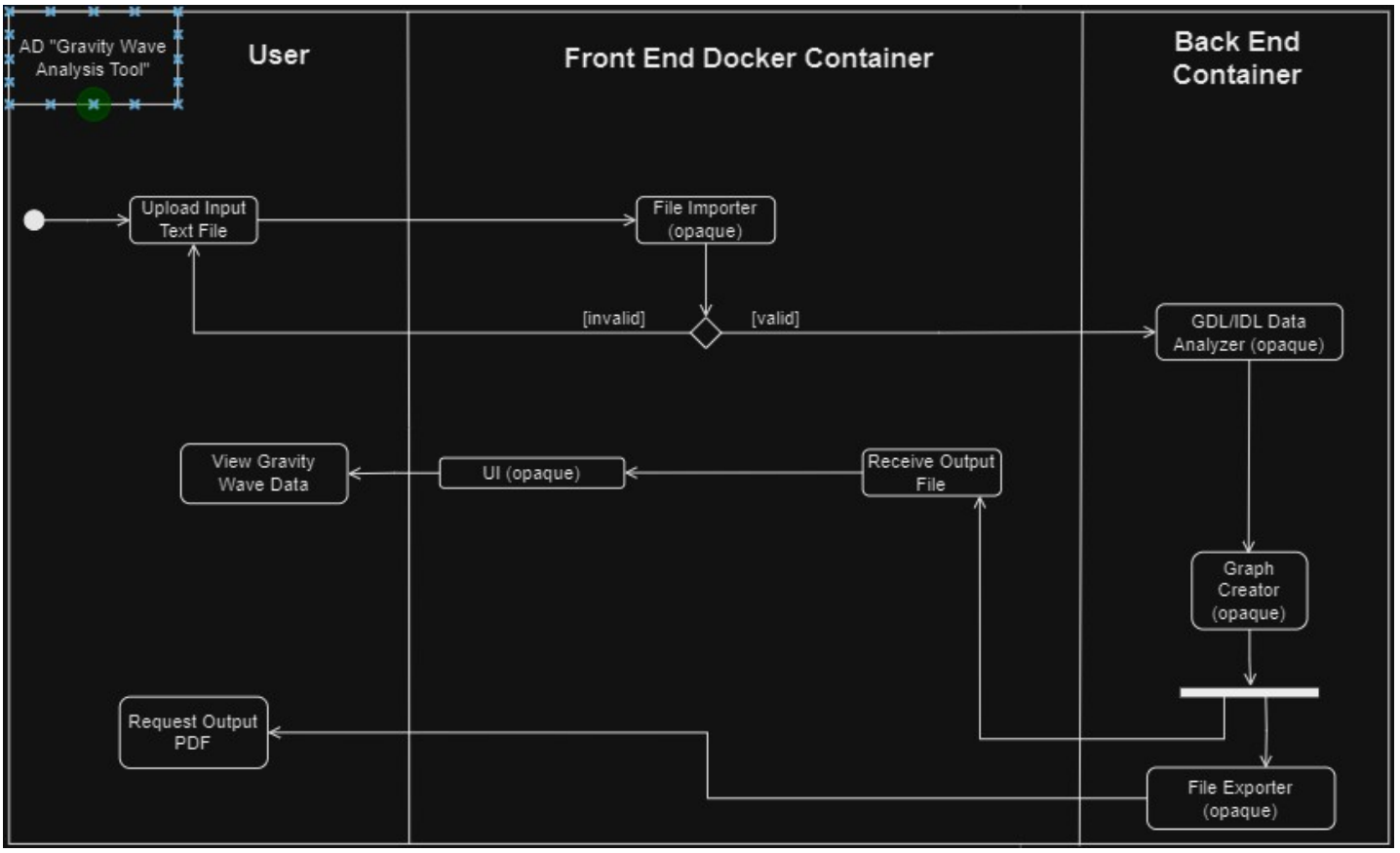

Create Graph
Docker Container
File Parser (Python)
Data Plotter
Docker container sends input data file to file parser
Parses and formats input data file
Sends parsed and formatted data to data plotter
Creates the data points, then plots and formats all required graphs
Data plotter sends back the plotted and formatted graphs to the Docker container

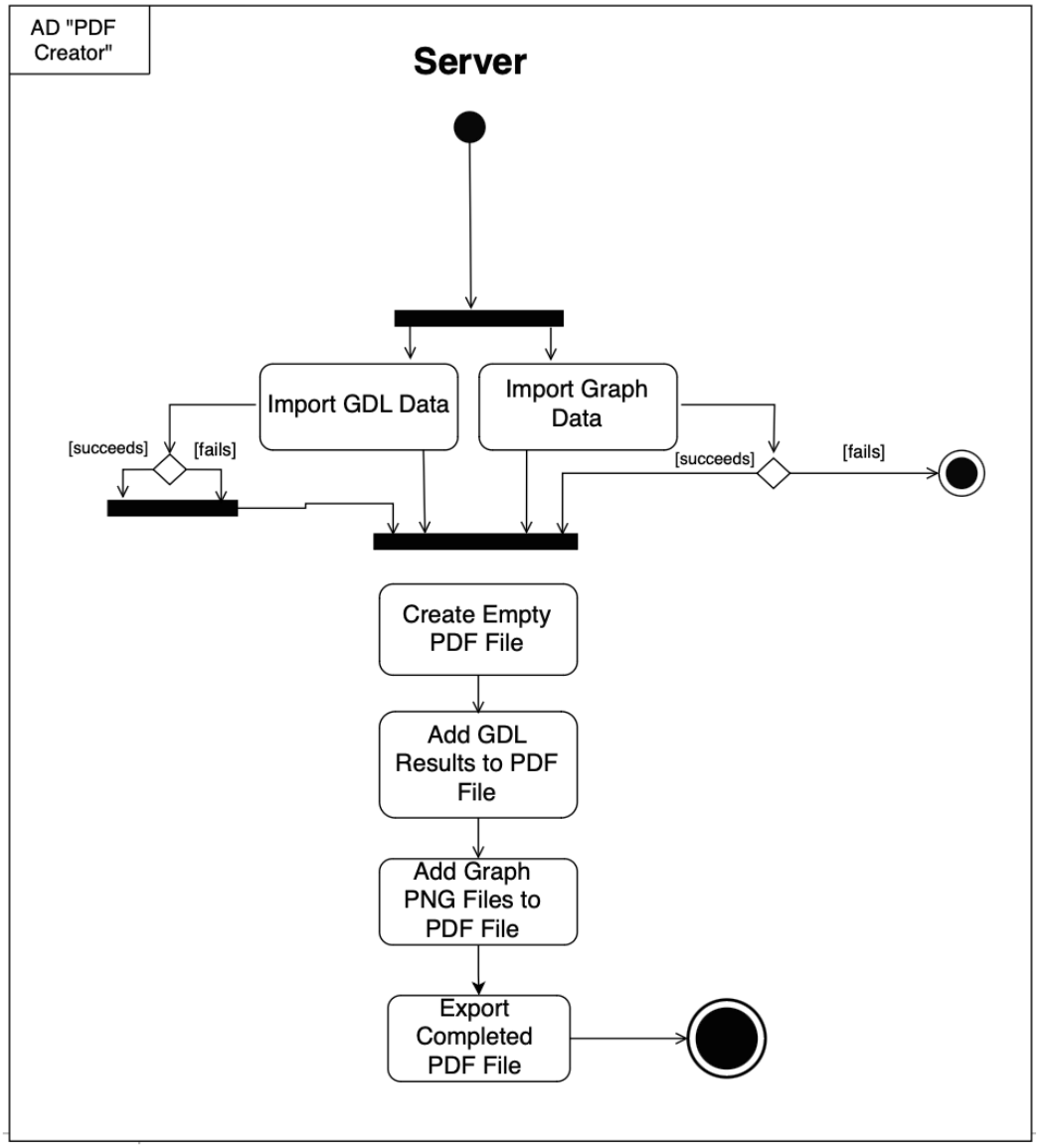


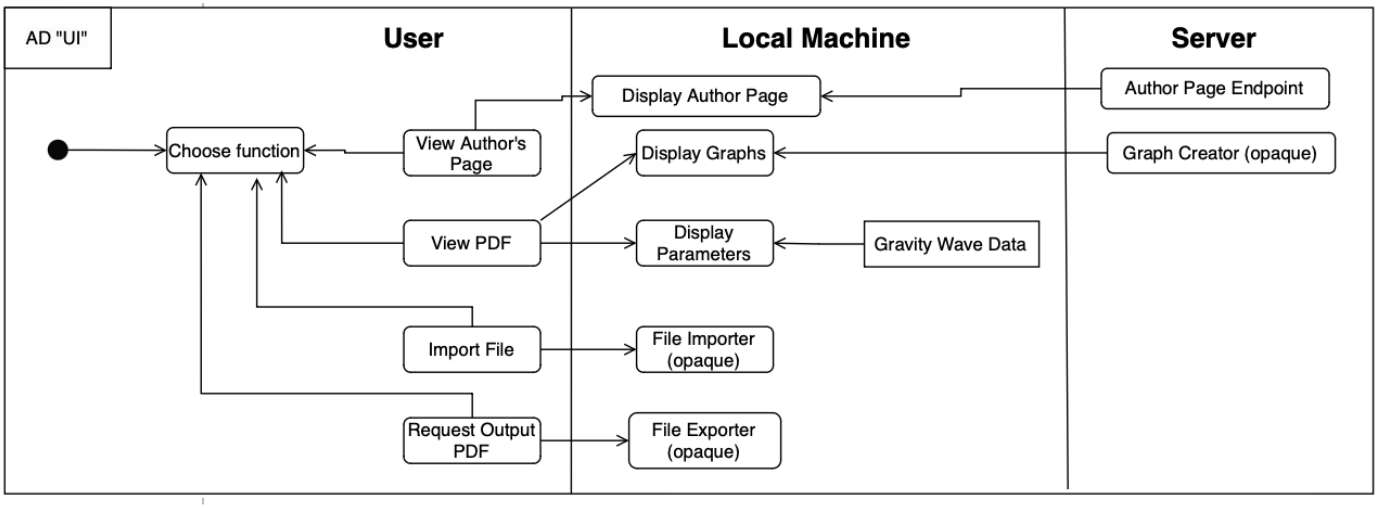


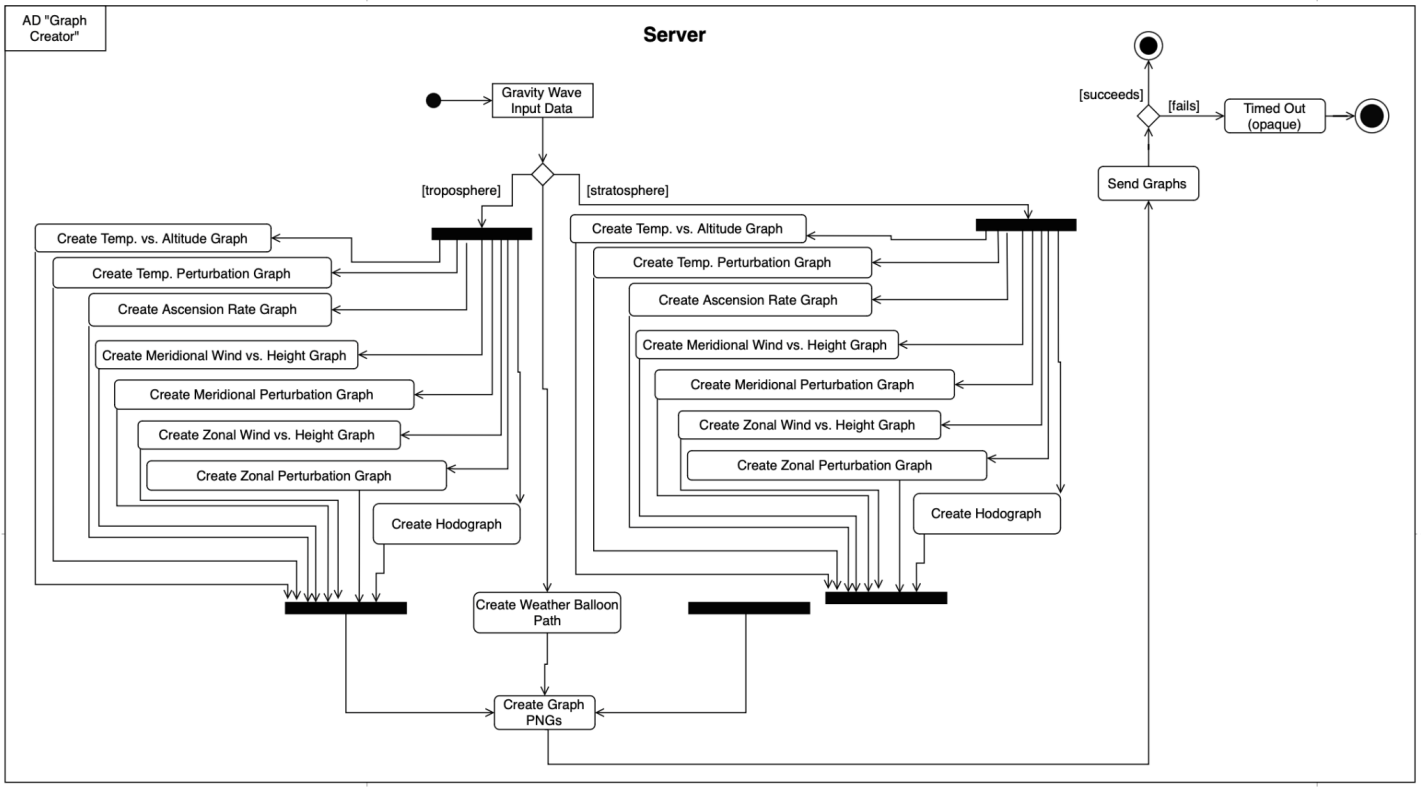
Activity Diagrams:

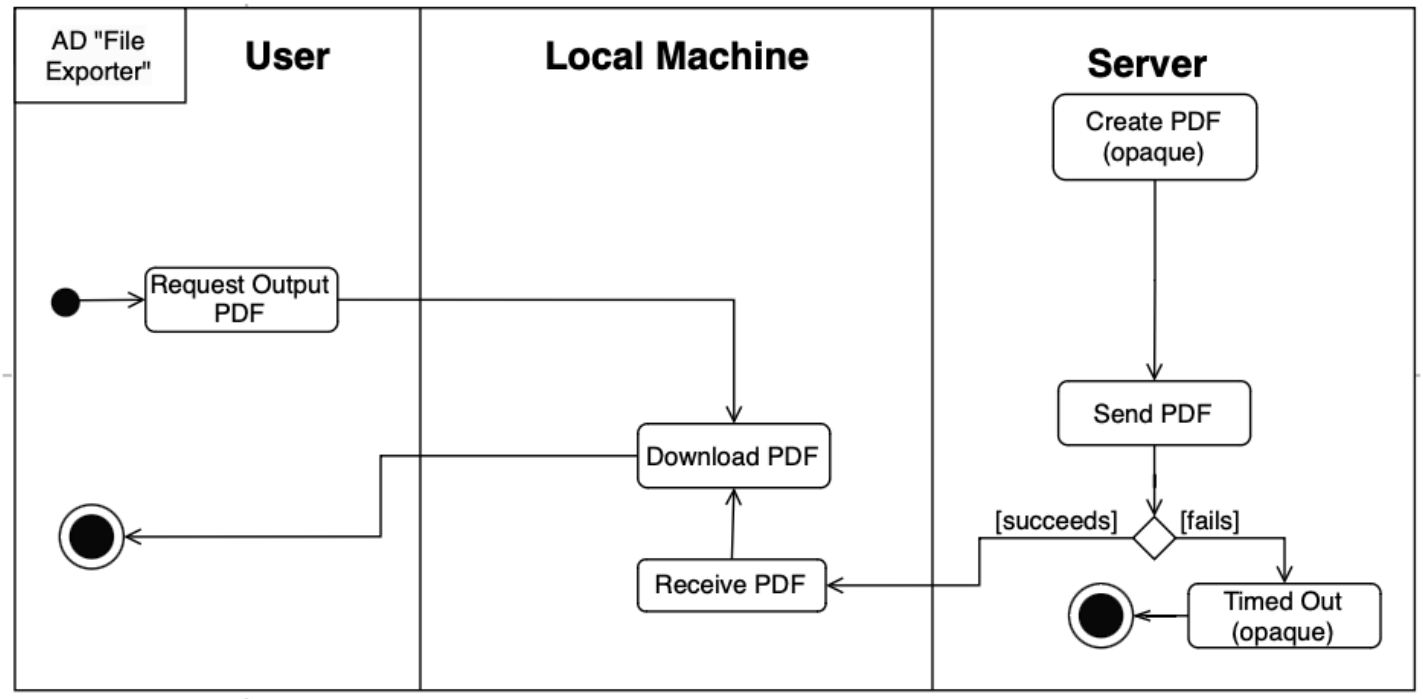

AD "Gravity Wave Analysis Tool"
User
Front End Docker Container
Back End Docker Container
Upload Input Text File
Recieve Input File
Verify Input File
[invalid]
[valid]
Send Text File
View Graphs
View Parameters
Display PDF
[fails]
[succeeds]
Download PDF
Request Output PDF
Receive PDF File
Recieve Text File
Parse Input Data File
Create Graphs
IDL/GDL Analyzes Data
[succeeds]
[fails]
[Sends Graph Data only]
Sends Graph and Parameters
Sends Graph
[succeeds]
[fails]
Format PDF Export File
Sends PDF


AD "Gravity Wave Analysis Tool"
User
Front End Docker Container
Back End Container
Upload Input Text File
File Importer (opaque)
[invalid]
[valid]
GDL/IDL Data Analyzer (opaque)
View Gravity Wave Data
UI (opaque)
Receive Output File
Graph Creator (opaque)
Request Output PDF
File Exporter (opaque)


AD "PDF Creator"
Server
Import GDL Data
Import Graph Data
[succeeds]
[fails]
[succeeds]
[fails]
Create Empty PDF File
Add GDL Results to PDF File
Add Graph PNG Files to PDF File
Export Completed PDF File


AD "UI"
User
Local Machine
Server
Display Author Page
Author Page Endpoint
Choose function
View Author's Page
Display Graphs
Graph Creator (opaque)
View PDF
Display Parameters
Gravity Wave Data
Import File
File Importer (opaque)
Request Output PDF
File Exporter (opaque)


AD "Graph Creator"
Server
Gravity Wave Input Data
[troposphere]
[stratosphere]
Create Temp. vs. Altitude Graph
Create Temp. Perturbation Graph
Create Ascension Rate Graph
Create Meridional Wind vs. Height Graph
Create Meridional Perturbation Graph
Create Zonal Wind vs. Height Graph
Create Zonal Perturbation Graph
Create Hodograph
Create Weather Balloon Path
Create Graph PNGs
Send Graphs
[succeeds]
[fails]
Timed Out (opaque)


AD "File Exporter"
User
Local Machine
Server
Request Output PDF
Download PDF
Receive PDF
Create PDF (opaque)
Send PDF
[succeeds]
[fails]
Timed Out (opaque)

6. Appendix B – Copyright

This document is based on a template meeting the ISO/IEC/IEEE 29148-2018 standard, available at https://www.iso.org/standard/72089.html. Template authors are:

Dr. rer. nat. Bastian Tenbergen,
Associate Professor of Software Engineering
bastian@tenbergen.org

Mikayla Conner-Spagnola, MA
Independent Consultant
mconner@oswego.edu

Department of Computer Science
State University of New York at Oswego
Oswego, NY 13126, United States