Nebula Net Interactive Feed

Software Design Specification (SDS), Project Plan, Software Requirements Specification (SRS)

Jacob Burke, Isabella Cortez, Freddy Lopez, Daniel Willard, Simon Zhao

24FEB2024 v1.1

Overview:
This document is a combined project plan, software design specification, and system requirements Specification into a single document for clarity and efficient documentation and grading each document is summarized and broken up for readability.

Table of Contents:

1. Software Requirements Specification

1.1. Document Overview

- This document serves as Software Requirements Specification (SRS) for the Computer Science 422 Software Methodologies Project 2 assignment Nebula Net Interactive Feed (NNIF). This document functions as the comprehensive requested feature set that is set by the ideation of project stakeholders. This document will spell out the justification, optional feature set, user types, system performance standard, user interface requests, and performance standards. This is an initial draft of this document and will be submitted upon completion and will undergo revisions in response to feedback provided by the instructor.

1.2. SRS Revision History:

Date	Author	Description
14FEB2024	Simon Zhao	Creation of initial SRS document for the JWST interactive feed website.
15FEB2024	Simon Zhao	Finished rough draft of SRS document.
20FEB2024	Daniel Willard	Reworking Document layout
23FEB2024	Daniel Willard	Poshishing statements
25FEB2024	Daniel Willard	Rework Requirements
27FEB2024	Daniel Willard	Finalizing for first submission

1.3. Introduction:

- The Nebula Net Interactive Feed website aims to provide a comprehensive platform for the public to access and explore captivating images captured by the National Aeronautical and Space Administration's (NASA) James Webb Space Telescope (JWST) in an interactive image gallery. This System Requirements Specification document outlines the functional and non-functional requirements necessary for the development and deployment of the website.

1.4. The Concept of Operations (ConOps)

1.4.1. Current System or Situation:

- Currently, users access James Webb Space Telescope (JWST) observations through various scattered sources provided by NASA and third parties in a disjointed and decentralized way. There is no singular, interactive platform that shows the current mission of JWST consolidates and provides detailed information and user engagement through a centralized and interactive feed. Nor is there a website that then collects and shows the completed missions. This not only creates a system that allows people in the United States of America how their tax dollars are being spent but also allows for space exploration to be accessible to everyone within reach of a computer.

1.4.2. Justification for a New System:

- A unified platform is necessary to enhance the accessibility and educational value of JWST images. There is a clear need for a system that not only displays these images but also allows users to interact with the data, customize their viewing experience, and receive updates on discoveries.

1.4.3. Operational Features of the Proposed System:

1.4.3.1. Real-time connection to JWST feed:

- The proposed system shall establish a real-time connection to the James Webb Space Telescope (JWST) feed, ensuring that the website is continuously updated with the most current publicly released images and data. This real-time integration will enable users to access the latest images and information captured by the JWST, providing an immersive and up-to-date experience.

1.4.3.2. Historical Log Since Launch (12JUL2024):

- A comprehensive historical log shall be maintained for each day since the launch of the James Webb Space Telescope on 12th July 2024. This log will serve as an invaluable archive, allowing users to explore past missions and observations conducted by the JWST. By providing access to historical data, the system enhances the educational value of the website, enabling users to track the progress of the mission over time.

1.4.3.3. Detailed Annotations and Celestial Object Information:

- The system shall incorporate detailed annotations and information for each image captured by the JWST. These annotations will provide context and insight into the celestial objects depicted in the images, enhancing the educational and scientific value of the website. Users will have access to comprehensive data provided by NASA.

1.4.3.4. Historical Gallery Organized by Date:

- The system shall feature a historical gallery organized chronologically by the most recent unlocked photos provided by NASA, allowing users to navigate through past observations and missions conducted by the JWST. This organization scheme enhances usability and facilitates efficient exploration of the vast image database.

1.4.3.5. Interactive Features for Image Viewing:

- The proposed system shall offer interactive features for image viewing, including zooming and downloading capabilities. Users will have the ability to zoom in on images to explore details with enhanced clarity, facilitating closer examination of celestial objects and phenomena. Furthermore, users will be able to download images for personal use or further analysis, fostering engagement and scientific exploration.

1.4.4. User Classes:

- The proposed system will cater to a diverse range of user classes, each with unique needs and expectations. The following user classes have been identified:

1.4.4.1. Astronomy Enthusiasts:

- Astronomy enthusiasts represent a significant user class who are deeply interested in acquiring detailed celestial information. These users seek to delve into the intricacies of space exploration and astronomical phenomena. They desire access to comprehensive data, including high-resolution images, detailed annotations, and scientific insights provided by experts in the field. Astronomy enthusiasts engage with the system to satisfy their curiosity and passion for understanding the universe.

1.4.4.2. Educators and Students:

- Educators and students form another crucial user class who utilize the system's images for educational purposes. They rely on the platform to access visually captivating imagery that enhances learning experiences in classrooms, laboratories, and educational settings. Educators integrate the system into their curriculum to illustrate key concepts in astronomy and astrophysics, while students utilize the images for research projects, presentations, and academic exploration. The system plays a vital role in facilitating knowledge dissemination and fostering a deeper understanding of celestial phenomena among educators and students alike

1.4.4.3. Casual Browsers:

- Casual browsers comprise a diverse user class consisting of individuals with a general interest in space imagery. These users may not possess a specialized background in astronomy but are drawn to the beauty and wonder of the cosmos. They visit the system to explore stunning images captured by the James Webb Space Telescope, seeking inspiration, entertainment, and awe-inspiring experiences. Casual browsers appreciate the accessibility and user-friendly interface of the system, which allows them to effortlessly browse through captivating imagery and learn about celestial objects and phenomena at their own pace.

1.4.5. Modes of Operation:

- The proposed system will operate through various modes to accommodate the diverse needs and interactions of users. Each mode represents a distinct scenario or use case in which users engage with the system. The following mode of operation has been identified:

1.4.5.1. Use Case 1: Image Exploration and Personal Gallery Creation:

1.4.5.1.1. Description:

- In this mode of operation, a user engages with the system to explore high-resolution images of celestial objects, focusing on a galaxy of interest. The user utilizes interactive features to zoom in on specific areas of the image, examining details with precision. Additionally, the user has the option to save favorite images to a personal device for future reference and enjoyment.

1.4.5.1.2. Steps:

- **1) Accessing the System:** The user navigates to the system's homepage using a web browser or mobile device.
- **2) Exploring Images:** The user browses through the image gallery and selects a high-resolution image of a galaxy for exploration.
- **3) Zooming In:** Upon downloading an image, the user utilizes the zoom feature to magnify specific areas of interest within the galaxy image, such as star clusters, nebulae, or spiral arms.
- **4) Examining Details:** The user carefully examines the detailed features of the galaxy image, observing intricate structures and phenomena captured by the James Webb Space Telescope.

1.4.5.1.3. Outcomes:

- By engaging in this mode of operation, the user experiences an immersive journey through the cosmos, exploring high-resolution images of galaxies captured by the James Webb Space Telescope. The ability to zoom in on specific details and save favorite images to a personal gallery enhances the user's connection to the celestial realm, fostering a sense of wonder and appreciation for the beauty of the universe.

1.5. Specific Requirements:

1.5.1. External Interfaces:

1.5.1.1. Image Feed Interface:

- The system shall interface with the Mikulski Archive for Space Telescopes (MAST) database to pull real-time images and data from the James Webb Space Telescope (JWST). This interface will facilitate the continuous updating of the website with the latest imagery and scientific information captured by the JWST. Through this interface, the system will retrieve image files in the Portable Network Graphics (PNG) format, along with accompanying metadata provided by the National Aeronautics and Space Administration (NASA). The image feed interface will ensure the availability of detailed annotations and celestial object information, enhancing the educational and scientific value of the website.

1.5.1.2. User Interface:

- The system shall provide a user interface for interaction with the website, offering intuitive navigation and functionality for users to explore, and interact with JWST imagery and data. Through the user interface, users will be able to browse the image gallery and see specific images organized by the most recent unlock date. Interactive features such as downloading for further study will be available to enhance user engagement and exploration of celestial imagery and accessibility to JWST photos.

1.5.1.3. Hosting Environment:

- The system will be hosted on a Linux server environment, specifically designed to accommodate the requirements of the application. Additionally, the system may utilize Amazon Web Services (AWS) for cloud hosting services, ensuring scalability, reliability, and efficient resource utilization. The Linux server and/or AWS hosting environment will provide the necessary infrastructure to support the functionality of the website, including image storage, data processing, and web server capabilities. By leveraging these external interfaces, the system will deliver a seamless and robust platform for accessing and interacting with JWST imagery and data beyond the completion of this course.

1.5.2. Functional Requirements:

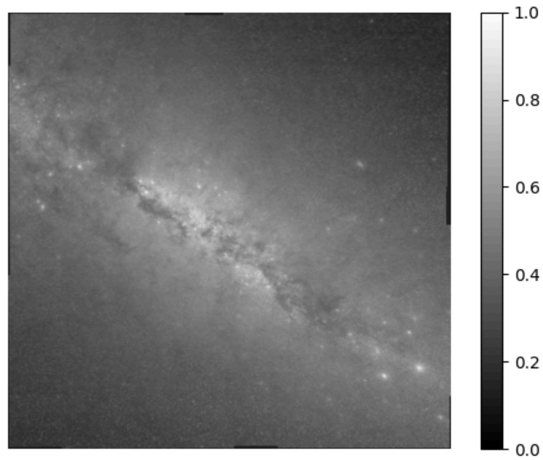
1.5.2.1. Validation:

- Unit tests shall be conducted to validate the system's functionality before deployment/updates.
- The MAST API shall be executed weekly to check for new photos and update the website accordingly.

1.5.2.2. Image display:

- The system shall provide a real-time feed of images and data from the James Webb Space Telescope (JWST) as they become publicly available.
- Each image displayed on the website shall be accompanied by relevant metadata, including the date of capture and any additional data provided by the National Aeronautics and Space Administration (NASA).
- The website shall support the display of images in the Portable Network Graphics (PNG) format due to its lossless compression bitmap characteristic and flexibility for storage and display.
- Users shall have the ability to download images directly from the website.
- Conversion of photos from .fits to .png format shall be performed to ensure viewability and accessibility of images.
- The most recent photos released to the public shall be accessible first to users.

1.5.2.2.1. **Image Display Example:**



(Scale on the side slow light emission by percentage representative of a single float value. {0.0 = no light, 1.0 = all spectrums of light} No data is shown here but information should be shown in a clean and understandable format below the photo.)

1.5.2.3. Image Metadata Management:

- Administrators shall have the capability to manage image metadata, including adding, editing, and deleting metadata entries.
- Changes to image metadata shall be reflected accurately and promptly on the website

1.5.3. Optional Functional Requirements:

1.5.3.1. Design Expansion:

- If time allows Expand the project to other Telespoces and observations from the MAST database.
- Add 3d Models
- Links to further educational resources
- Detailed annotations of celestial objects captured in the images shall be provided by NASA and displayed alongside the corresponding images.

1.5.3.2. **Responsive Design**:

- The website shall be responsive across various devices and screen sizes, including desktops, laptops, tablets, and smartphones.
- Content layout and navigation shall adapt dynamically to provide an optimal viewing experience on different devices.

1.5.3.3. Dynamically Hosted Website:

- The goal is to have the website Dynamically hosted on Amazon Web Services (AWS) to support seamless updates and modifications. Keeping the website as a static service will save on future costs.
- Content updates, including image additions and metadata changes, shall be reflected on the website without disruption to user access. This will be done through intermittent timed updates.

1.5.4. Nonfunctional Requirements:

1.5.4.1. Performance Requirements:

- SQL database updates shall occur within 24 hours of receiving new data.
- File conversion processes, specifically from .fits to .png format, shall be completed within one hour to ensure timely availability of images.
- The cumulative size of the photo directory shall not exceed 10 gigabytes at any given time.

1.5.4.2. System Environment:

- The website shall be capable of running in static mode to ensure reliability and stability.
- The system shall be compatible with Linux-based environments, specifically, the ix-dev Linux server hosted by the University of Oregon.

1.5.4.3. Accreditation and Compliance:

- All images displayed on the website shall be accredited appropriately, adhering to copyright and usage guidelines.
- Any libraries or third-party components utilized in the system must be approved by the designated authority, Professor Anthony Hornof overseeing the project.
- The total expenditure on AWS services shall remain under $50 to maintain cost-effectiveness.
- The corresponding data shall be presented alongside the photos to provide context, information, and accreditation.

1.5.5. Usability Requirements:

1.5.5.1. Intuitive Navigation:

- The user interface shall feature intuitive navigation controls, to enable users to easily browse through images and access desired content.
- The navigation menu shall be logically organized, with clear labels and hierarchical structure, facilitating efficient exploration of website content.

1.5.6. Software System Attributes:

1.5.6.1. Maintainability:

- The system shall be designed with modular and well-structured code, facilitating easy updates and maintenance of the image feed.
- Code documentation shall be comprehensive and up-to-date, providing guidance for developers and administrators during maintenance activities.
- Changes to the image feed, including updates to image sources or metadata structures, shall be implemented efficiently and with minimal disruption to system functionality.
- Version control systems, such as Git, shall be utilized to track changes to the codebase and facilitate collaboration among development team members.
- Automated testing suites shall be implemented to verify system integrity after updates or modifications to the image feed, ensuring that new changes do not introduce regressions or unexpected behavior.

1. Project Plan

1.1. Document Summary:

- This document presents the comprehensive plan for the Computer Science 422 Software Methodologies Project 2 Nebula Net Interactive Feed (NNIF). This Document delineates the composition of the team, delineates their designated roles and deliverables, establishes meeting schedules, outlines the project timeline and deliverables, and provides the rationale for these determinations. This is the initial draft of this document to be submitted and refined based on constructive feedback from the instructor.

1.2. Organizational Makeup:

1.2.1. **Jacob Burke:** Test Automation/ Hosting/ Cloud Engineer

1.2.2. **Isabella Cortez:** Front-End/ UI/ Web Developer

1.2.3. **Freddy Lopez:** Front-End/ UI/ Web Developer

1.2.4. **Daniel Willard:** Project Manager

1.2.5. **Simon Zhao:** Backend /Integration Developer

1.3. Project Roles / Division of Tasks:

1.3.1. **Test Automation Engineer:** Responsible for designing, implementing, and maintaining automated testing frameworks and scripts to validate software functionality. Collaborates with the development team to identify test cases, automate repetitive tasks, and ensure the reliability and efficiency of testing processes. Conducts test execution, analyzes results, and reports any issues or defects to the development team for resolution.

1.3.2. **Hosting/ Cloud Engineer:** Responsible for the configuration, deployment, and maintenance of web hosting and cloud infrastructure. Tasks include connection to existing servers and ensuring the availability and performance of hosted websites. Collaborates with development teams to troubleshoot hosting-related issues and optimize web server configurations for optimal performance and security. Implements best practices for web hosting management, including backup and presentation strategies.

1.3.3. **Front-End/ UI/ Web Developer:** Responsible for creating visually appealing and user-friendly interfaces for web applications. Tasks include designing and implementing front-end components using HTML, CSS, and JavaScript frameworks, ensuring compatibility across various browsers and devices. Collaborates with designers and backend developers to integrate UI elements with backend systems, ensuring smooth functionality and optimal user experience. Implements responsive design principles to enhance usability and accessibility of web applications.

1.3.4. **Project Manager:** Responsible for overseeing all aspects of the project, including planning, execution, and delivery. Tasks include defining project scope, defining roles, documentation creation, objectives, and timelines, and allocating resources. Coordinates with team members to assign tasks, monitor progress, and ensure adherence to project milestones and deadlines. Acts as the main point of contact for stakeholders, providing regular updates on project status and addressing any concerns or issues that arise. Facilitates communication and collaboration among team members, fostering a positive and productive working environment, and assignment of roles.

1.3.5. **Backend /Integration Developer:** Responsible for establishing connections to existing databases and developing back-end systems to format and package data efficiently. Collaborates with other developers to establish APIs and access to back-end services, as well as facilitating the shipping of data.

1.4. Decision Guidelines:

- The Project Manager referred to as Willard, is responsible for strategizing the project's overarching plan and delegating tasks among developers. Willard also assumes the responsibility of finalizing and submitting documentation. Git/GitHub serves as the designated version control system, with Simmon entrusted to oversee final merges from development branches to the main branch, subject to approval by the Project Manager.

1.5. Meeting Times / Location:

Day	**Location**	**Time**
Sunday	Discord (online communication platform)	12:00-14:00
Monday	Knight Library, Eugene, Oregon (in person)	15:30-17:30
Tuesday	Knight Library, Eugene, Oregon (in person)	15:30-17:30
Wednesday	Knight Library, Eugene, Oregon (in person)	15:30-17:30
Thursday	Allan Price Science Library, Eugene (in person)	12:00-14:00
Friday		
Saturday		

1.6. Project Timeline / Deliverables:

Task or Milestones	Status	Assignment	Due Date	Confirmed Completion Date
Week 0 (11FEB - 17FEB) Planning and Setup				
Project Ideation	Complete	Everyone	12FEB2024	12FEB2024
Setup Github repo	Complete	Freddy	18FEB2024	15FEB2024
Week 1 (18FEB - 24FEB) Prototyping				
Complete 3-page proposal	complete	Willard	24FEB2024	19FEB2024
Web scraper	Complete	Isabella	19FEB2024	19FEB2024
Website Prototype	Complete	Freddy	24FEB2024	19FEB2024
Connection test to Database	Complete	Simon	24FEB2024	19FEB2024
Set up AWS/IXDev Web hosting	Complete	Jacob	24FEB2024	18FEB2024
Week 2 (25FEB - 02MAR) Integration				
SRS/SDS/Project Plan	Inprocess	Willard	26FEB2024	
Instillation/Startup Scripts	Incomplete	Jacob	02MAR2024	
Data collection Unittests	Incomplete	Jacob	02MAR2024	
IMP Module Unittests	Incomplete	Jacob	02MAR2024	
Data Packaging from the database	Incomplete	Simon	02MAR2024	
Connection to Website	Incomplete	Simon	02MAR2024	
Web scraper data handoff to Web	Incomplete	Isabella	02MAR2024	
The web page layout is finished	Incomplete	Freddy	02MAR2024	25FEB2024
Website Connection to data	Incomplete	Freddy	02MAR2024	

Week 3 (03MAR - 09MAR) Testing/Debugging/Finalize				
Finalized and debug software	Incomplete	Everyone	8MAR2024	
Testing User interface	Incomplete	Jacob	8MAR2024	
Week 4 (10MAR - 16MAR) Submission and Presentation				
Finalized Documentation	Incomplete	Willard	10MAR2024	
Project Submission	Incomplete	Willard	11MAR2024	
Project Presentation	Incomplete	Everyone	12MAR2024	

1.7. Monitoring/ Reporting Guidelines:

- Reporting will be facilitated through a revision history log provided below, which will be consistently updated by team members during their project engagement. Each update will encompass a summary of completed tasks or encountered impediments. Project monitoring and progress assessment will be conducted during the Sunday meetings, where deliverables will be reviewed, and any outstanding issues will be promptly addressed after that.

Date	Author	Description
WEEK 0		
12FEB	Everyone	Everyone Completed project ideation and NNIF was chosen for Development
15FEB	Willard	Project documents set up and shared: SDS/ project plan, Programer note, meeting notes
15FEB	Simon	GitHub repository Established for version control
15FEB	Simon	Created rough Draft SRS
15FEB	Jacob	Created Web hosting instructions
15FEB	Freddy	Web Page Design Mockup on Fuma.
WEEK 1		
18FEB	Jacob	Set up Amazon Web Services and hosted the initial Static Website
19FEB	Isabella	JWST Data Web Scraper
19FEB	Willard	3-page proposal submitted after review
19FEB	Freddy	Web page Prototype built
19FEB	Simon	Connection to NASA database tested and connection confirmed (sign up for token)
WEEK 2		
25FEB	Freddy	Finished React Home and About Page
26FEB	Simon	Implemented FITS processing class
27Feb	Isabella	Web Scraper + Database Creator Implementation
WEEK 3		

WEEK 4		

1.8. Rationale Behind Timeline/ Project Taskings:

- In order to foster effective collaboration and ensure progress within the group, task assignments have been meticulously defined, and a timeline has been meticulously outlined. Responsibilities have been distributed among team members, each assigned with specific roles and titles tailored to their respective backgrounds and desired learning objectives for the capstone project. These roles were carefully allocated based on individual expertise and interests, with project taskings delineated according to the expected deliverables associated with each role's responsibilities.
- To facilitate seamless collaboration, various communication tools have been selected for team interaction. While team members are expected to engage in ongoing collaboration throughout the project duration to ensure the integration of all system components, multiple meeting times have been established during the week to address any obstacles or concerns. A Discord server has been established as the primary platform for project meetings and communication, enabling remote gatherings via conference calls and screen sharing. Additionally, Discord facilitates continuous collaboration through designated server rooms for text-based discussions, accessible at any time. While project documents can also be shared via Discord, the primary tool for document collaboration remains Google Docs, offering real-time editing and collaboration features.
- The project timeline has been meticulously structured to provide each team member with sufficient time to accomplish weekly objectives. Sunday meetings, in particular, are dedicated to reviewing deliverables and addressing any significant challenges, allowing for subsequent resolution of outstanding issues or adjustments to the project plan, if necessary. With the proposed project timeline outlining tasks, completion is anticipated by March 8, with an additional three-day buffer allocated for flexibility in resolving any lingering issues.

2. Software Design Specification

2.1. Document Summary:

- This document serves as the Software Design Specification (SDS) for the Computer Science 422 Software Methodologies Project 2 assignment Nebula Net Interactive Feed. It functions as the comprehensive design blueprint for the project, encapsulating program requirements, system architecture, detailed module specifications, and documentation revision history. Incorporated within are elaborate diagrams conforming to the Unified Modeling Language, illustrating architectural components and modes. This is an initial draft of this document and will be submitted upon completion and will undergo revisions in response to feedback provided by the instructor.

2.2. SDS Revision History:

Date	Author	Description
Week 0		
15FEB	Willard	SDS Google Doc Setup/Shared
17FEB	Entire Group	Project Plan Roughly Drafted
Week 1		
18FEB	Willard	SDS setup or self-reporting of tasks
20FEB	Willard	Timeline Set up and revised
23FEB	Entire group	Filled out respective modules and added models
24FEB	Willard	Working Initial Submission models added and modified
25FEB	Willard	Working on module edit description form peer input
Week 2		
26FEB	Willard	Finalize SDS for submission and submited
Week 3		
Week 4		

2.3. System Overview:

- The Nebula Net interactive Feed hosted website on local servers is a website and supporting software that allows users to view up-to-date JWST photos and the completed mission compiled photos. This is done with two web pages. The first is the home landing page that will display the last mission photo taken from JWST. This page will have a list of informative information and the current mission this photo is a part of. The second page is a mission timeline that will provide the mission information from the NASA JWST website. These user interactions as well as the user to download and view photos from the JWST are done through a limited user interface so all interactions are click-based and not text-based. The website will be run on a React software framework to allow flexibility in hosting on different devices. The set of Processing modules

2.4. Software Architecture:

- The Nebula Net Interactive Feed is a web-based platform hosted on local servers, facilitating the viewing of up-to-date James Webb Space Telescope (JWST) photos and compiled mission images. The system comprises two primary web pages: a landing page displaying the latest mission photo captured by the JWST and a mission timeline page providing comprehensive mission information sourced from the NASA JWST website. These user interactions, including photo viewing and download capabilities, are facilitated through a limited user interface optimized for click-based interactions.

2.4.1. NebulaNet Component Architecture

2.4.1.1. Space User Interface (SUI) (The displayed website):

- Provides and displays photographic, mission, and informative data from the official James Webb Space Telescope (JWST) Database and website hosted by the National Aeronautics and Space Administration (NASA). This allows users to view the mission photo of the day that is part of a larger mission that is used to create the large and compiled beautiful photo that most are familiar with. The website will display only the current mission photo set and the mission timeline with the compiled photos from each completed mission.

2.4.1.1.1. Photo and Metadata Coalescence - Fits→Notation+PNG (PMC-FNG)

- The PMC-FNG module plays a crucial role in enhancing user accessibility to astronomical imagery captured by the JWST. Its primary function is to convert Flexible Image Transport System (FITS) files, the standard format for astronomical data, into Portable Network Graphics (PNG) images, which are more readily viewable by users. Leveraging libraries such as astropy, numpy, and matplotlib, PMC-FNG processes FITS files to render detailed PNG images, enabling users to visualize celestial data in grayscale or color format. Additionally, PMC-FNG stores metadata for each PNG image within text files, ensuring that relevant contextual information accompanies the visual representation. This module interfaces with the Space User Interface (SUI) though a directory to encourage los coupling and replication for othe, providing PNG images and metadata derived from JWST observational data fetched from the MAST database.

2.4.1.1.2. Mission Information Gatherer (MIG)

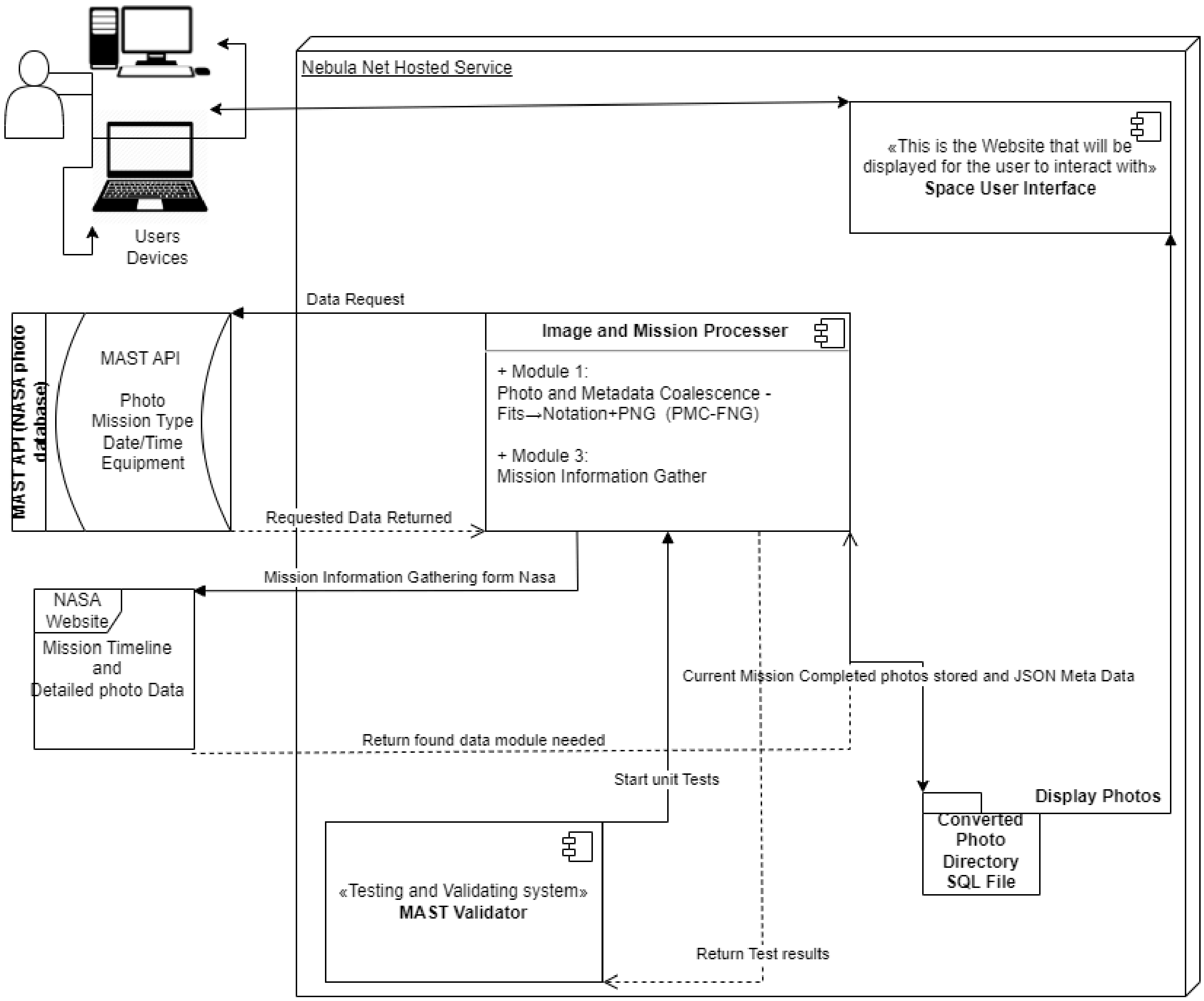
- The MIG module serves as the backbone for retrieving mission-critical data from the James Webb Space Telescope (JWST) website, ensuring users have access to up-to-date mission information. Its primary function is to gather observing schedules data and organize it into a structured format for seamless integration into the Nebula Net platform. By efficiently collecting and organizing mission data, MIG enables users, including researchers and enthusiasts, to stay informed about upcoming missions and observations. Leveraging web scraping techniques, MIG extracts data from the JWST website's observing schedules, converting it into JSON format for efficient storage and processing. This module interfaces directly with the PMC-FNG and in the static version SUI, supplying mission data for display on the website's landing page and mission timeline.

2.4.1.2. MAST Validator (MASTv):

- The MASTv module serves as a critical component in ensuring the integrity and functionality of the Nebula Net Interactive Feed system. Its primary role is to validate the functionality of key modules, including the Mission Information Gatherer (MIG) and Photo and Metadata Coalescence - Fits→Notation+PNG (PMC-FNG), by conducting comprehensive unit tests. MASTv verifies the proper functioning of the MAST API connection and the accessibility of the JWST observation website, detecting any potential issues or changes that may impact the system's operation. By running defined unit test cases, MASTv assesses the functionality of each module, identifying bugs, dependencies, or implementation changes that require attention. This module operates during the initial setup or reset of the website, ensuring that the back-end Python programs function as intended and providing early detection of any discrepancies or inconsistencies. Through its rigorous testing procedures, MASTv contributes to the reliability and stability of the Nebula Net platform, ensuring a seamless user experience and facilitating timely resolution of any detected issues.

Nebula Net Component Diagram

This is a UML Diagram Showing the components and their interaction



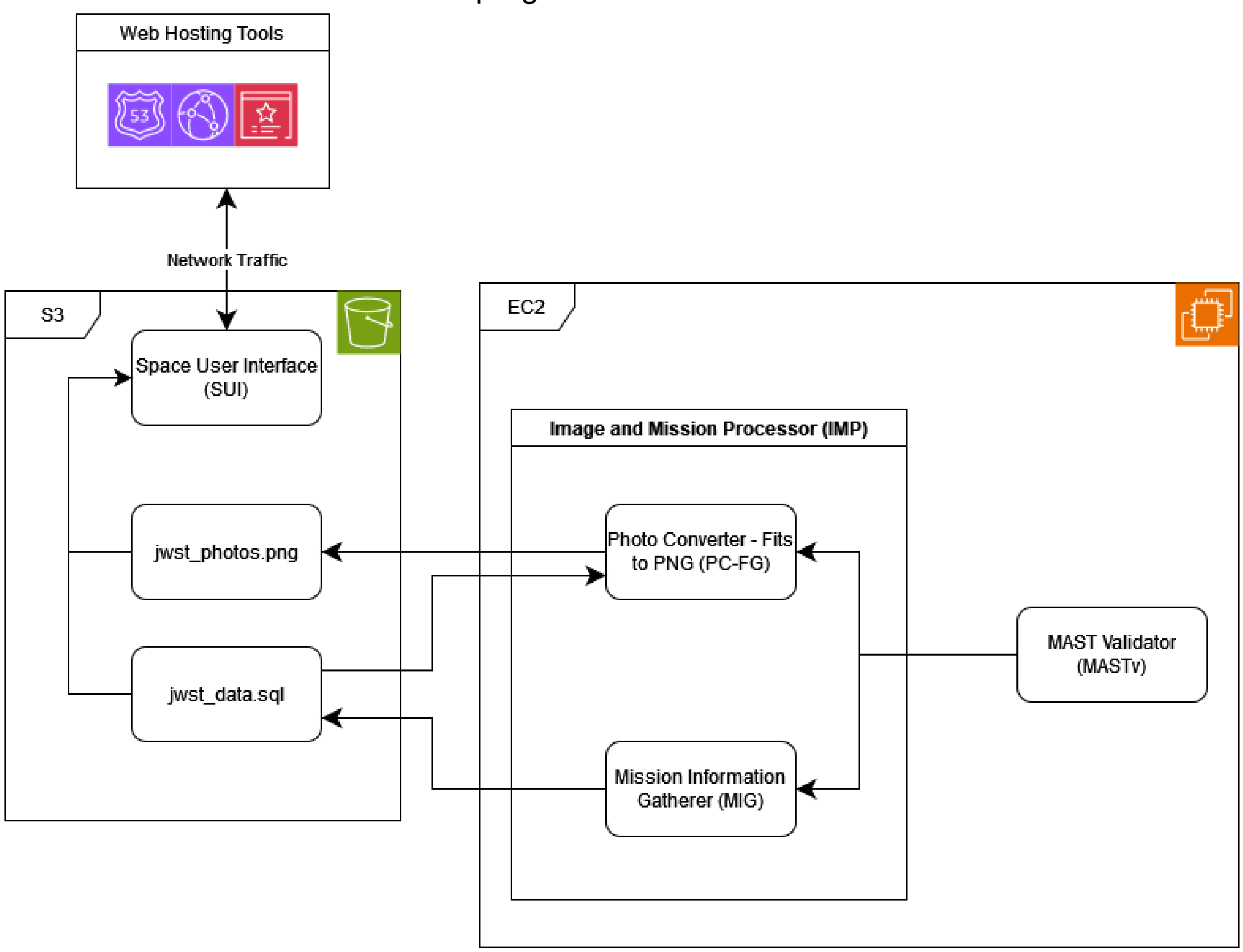
2.4.2. Web Hosting Architecture

2.4.2.1. AWS Components

- **Route53:** A Domain Name System (DSN) webservice offered by AWS. Used to register the domain name for the website, as well as connect incoming user requests to the respective Availability Zone the site is hosted at.
- **CloudFront:** Amazon Web Services Content Delivery Network. Essentially is a distributed network

of servers amongst the different availability zones. Optimizes delivery of website traffic and supports use of SSL certificates.

- **Certificate Manager:** Used to register and manage SSL/TLS certificates in other AWS services.
- **Simple Storage Service (S3):** The standard cloud storage service offered by AWS. Used to store website files and source code.
- **Elastic Compute Cloud (EC2):** A virtual private server hosted by AWS. Allows hosting of Linux Instances. Used to run web applications and python programs for the NebulaNet website.



Web Hosting Architecture Diagram

2.5. Software Modules:

2.5.1. Space User Interface (SUI) Module:

2.5.1.1. Role and primary function:

- **Role:**
  - The Space User Interface (SUI) Module plays a pivotal role in shaping the website's user experience by implementing the landing page and key functionalities. Its primary role is to design and structure the landing page, including the header section with essential links to other pages such as Calendar, About, and Sources. Additionally, it orchestrates the display of the daily James Webb telescope photo and previews of pictures taken within the previous three days, providing users with easy access to relevant content and navigation options.
- **Function:**
  - The Space User Interface (SUI) Module serves as the visual representation of the website's components and modules, ensuring an intuitive and seamless user experience. Leveraging REACT, it adopts a modular approach to website development, enabling the creation of subpages and a mobile version adaptable to various screen sizes. By incorporating responsive design principles, it enhances accessibility and usability across different devices, catering to the diverse needs of users.

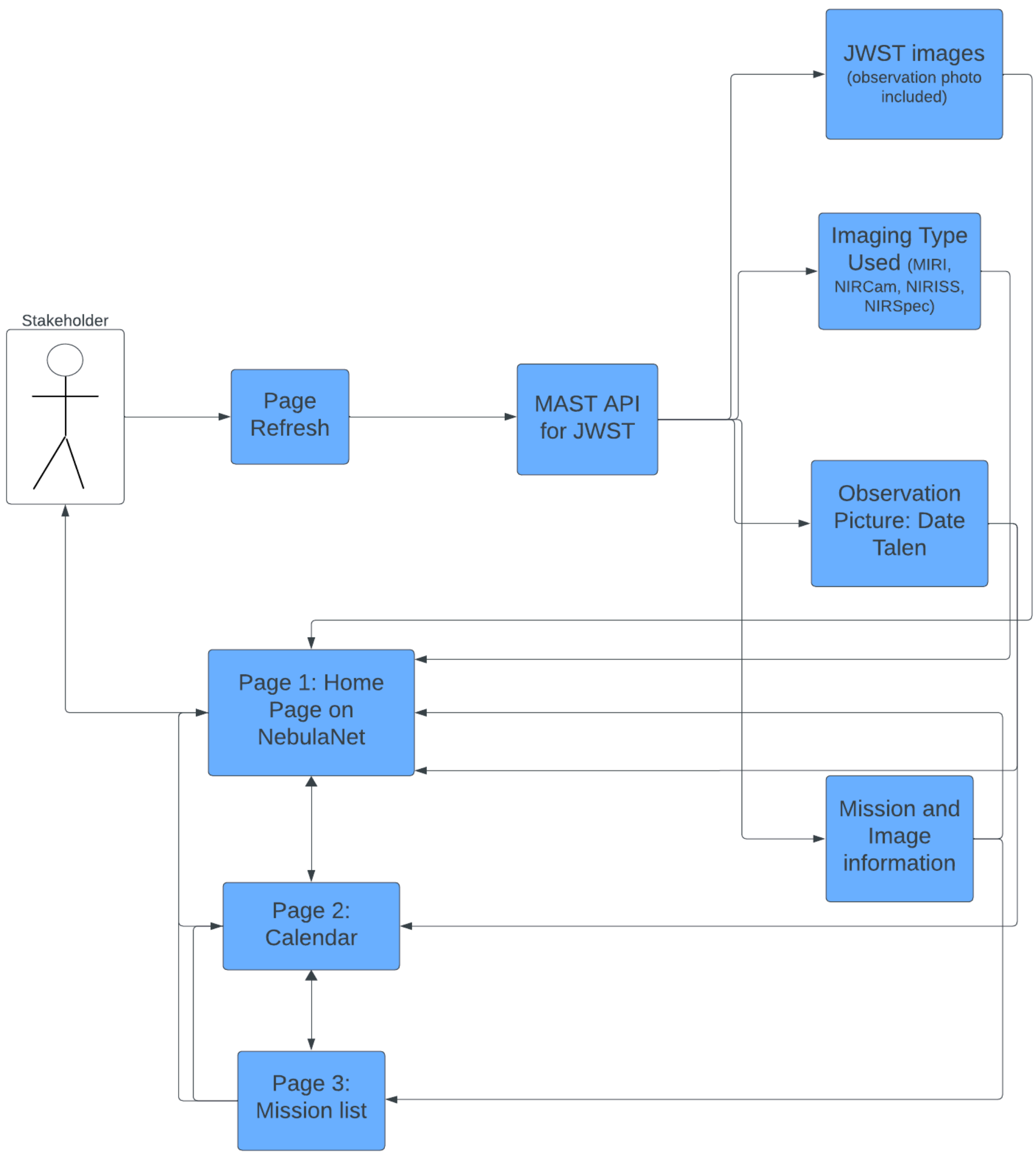
2.5.1.1.1. Components:

- **Role:** The components directory contains all the javascript and corresponding css files that create each object used within the website.
- **Function:** This allows for easy creation of additional pages throughout the website as predefined components can be called to be placed wherever they are needed. This promotes modularity and ensures strong cohesion within the entire SUI.

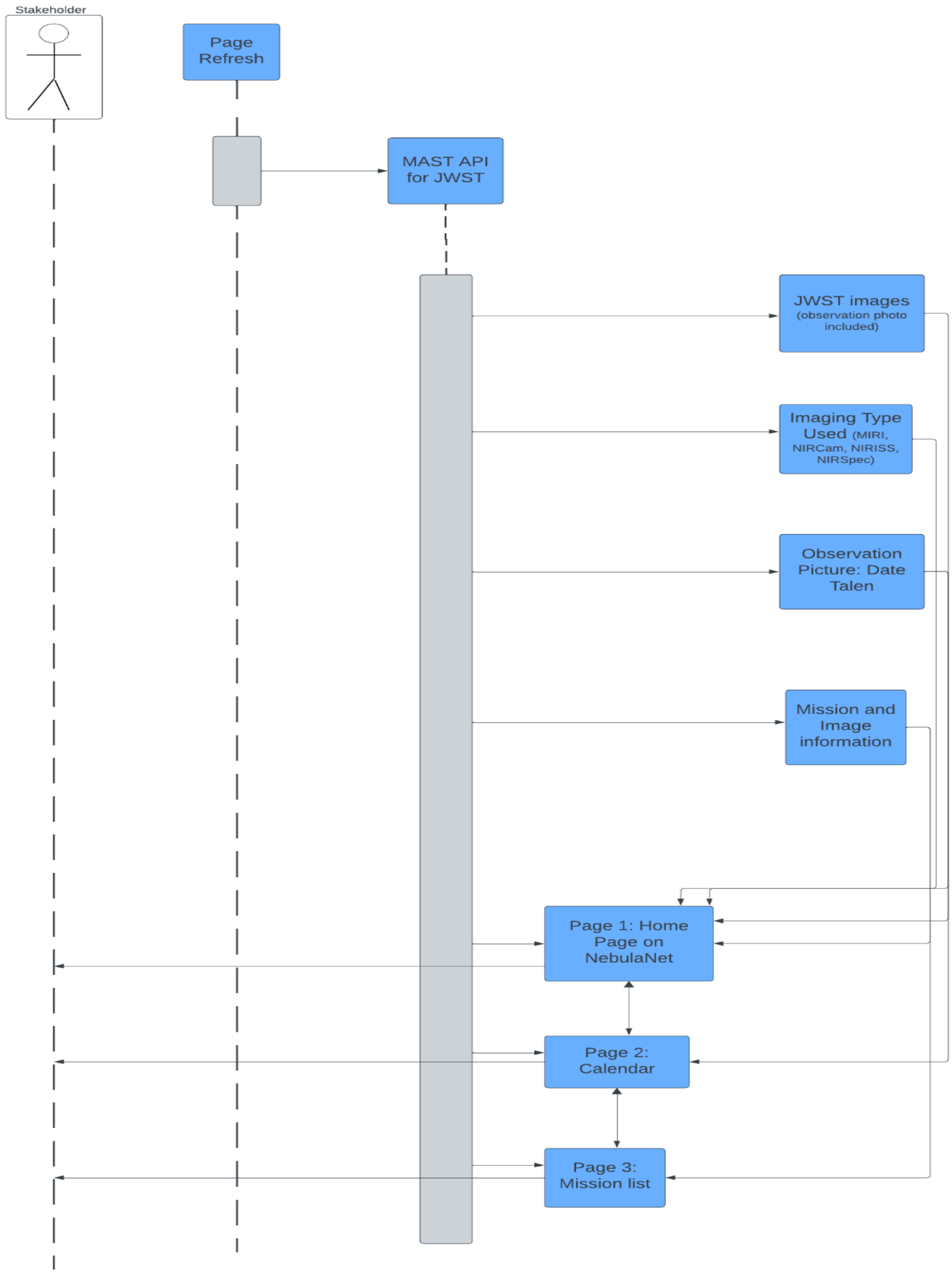
2.5.1.2. Interface to other modules:

- The app.js web page serves as the cornerstone for interconnecting all modules within the software system, acting as the primary gateway for user interaction. It orchestrates the retrieval and display of backend data in a visually appealing format, ensuring a seamless and aesthetically pleasing user experience. Additionally, embedded links to other pages within the website are integrated into the landing page, facilitating effortless navigation between modules and web pages.

2.5.1.3. Static model:



2.5.1.4. Dynamic model:



2.5.1.5. Design rationale:

- The design rationale for the landing page prioritizes the presentation of the daily observation from the James Webb telescope as the central focus upon user entry, aligning with the site's primary purpose. Following this, a descriptive section is situated below, featuring details such as the object's name, date of capture, and a concise narrative on the photographic process and associated mission. This sequential layout aims to facilitate user engagement by allowing them to appreciate the displayed photo before delving into its contextual background. As users navigate down the page, a condensed preview of the previous day's photos is provided alongside a button leading to the mission timeline webpage which will display a timeline of all missions that the JWST has completed since launch.

2.5.2. Photo and Metadata Coalescence - Fits→Notation+PNG (PMC-FNG):

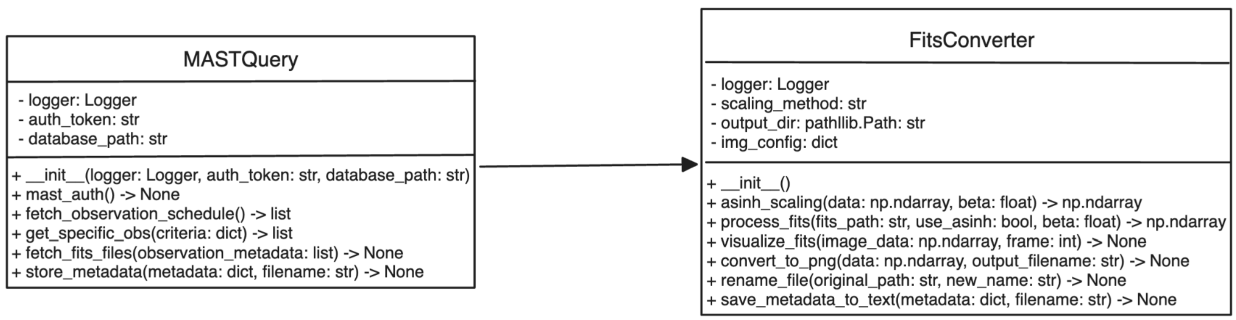
2.5.2.1. Role and primary function:

- **Role:**
  - The Photo and Metadata Coalescence - Fits→Notation+PNG (PMC-FNG) module is designed to facilitate the conversion of Flexible Image Transport System (FITS) files, the standard format for astronomical data, into PNG (Portable Network Graphics) image formats. It leverages the *astropy* library for FITS file processing, then uses *numpy* library to process the FITS file and return the scaled image data. The *matplotlib* library is used to translate the FITS data into 2D arrays representing pixel shading values. These values are then rendered into detailed images, enabling the visualization of celestial data in grayscale or color PNG format.
- **Function:**
  - Beyond converting FITS to PNG, the PMC-FNG module stores metadata for each corresponding PNG image within text files. This approach ensures that each visual representation is accompanied by relevant contextual information, which enhances the understanding and utility of the images. By parsing FITS field for both imagery and metadata.

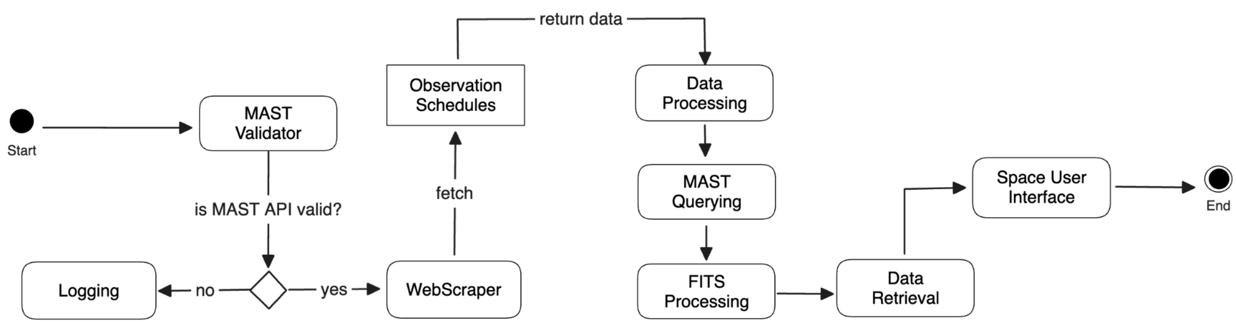
2.5.2.2. Interface to other modules:

- This module interacts with the Space User Interface (SUI) with PNG images and their metadata, derived from the current mission's observational data fetched from the MAST database. This interface supports the SUI's role in displaying mission-critical photos and providing user engagement with the data.

2.5.2.3. Static model (Class Diagram):



2.5.2.4. Dynamic model (Activity Diagram):



2.5.2.5. Design rationale:

- The rationale behind the design of the Photo Converter - FITS to PNG (PMC-FNG) module stems from the inherent challenge of viewing .fits files without specialized software, rendering them inaccessible to the average user. Thus, the module aims to address this limitation by facilitating the conversion of .fits images into viewable photo files accessible over a network for the Space User Interface (SUI). The design choices are constrained by the nature of the SUI being a website, necessitating adherence to the preexisting network and MAST API systems prescribed by NASA.

- Python was selected as the programming language due to its extensive support for scientific computer and data visualization. Specifically, the *astropy* library allows for robust handling of FITS files that enables precise reading and processing of astronomical data. This library is essential for extracting the detailed, multidimensional arrays that FITS files often contain, which provides raw data that’s necessary for image conversion.
- Furthermore, *numpy* enhances the modules capabilities by offering high-performance numerical computations. This is particularly crucial for processing datasets typical in astronomical observations, which allow for efficient manipulation of image data arrays. Numpy’s array operations are crucial for transforming pixel values into a format suitable for visualization.
- For the actual conversion to PNG format, *matplotlib* is utilized for its extensive plotting functionalities and support for various output formats. It translates the numerical data processed by *numpy* into gradations of shading, which renders detailed grayscale or color visual representations.
- Post conversion, to optimize storage, the original FITS files are discarded, acknowledging their large size and the premium on storage space. Concurrently, metadata for each PNG image is saved into a text file, ensuring that essential information is retained and easily accessible. This approach not only makes astronomical data more approachable but also manages storage efficiently, maintaining the system's responsiveness and relevance. The module’s strategic focus on the current mission's photo set, coupled with periodic purging of the photo directory, further ensures that only pertinent and timely data is available, enhancing the SUI's operational efficiency and user experience.

2.5.3. Mission Information Gatherer (MIG) Module:

2.5.3.1. Role and primary function:

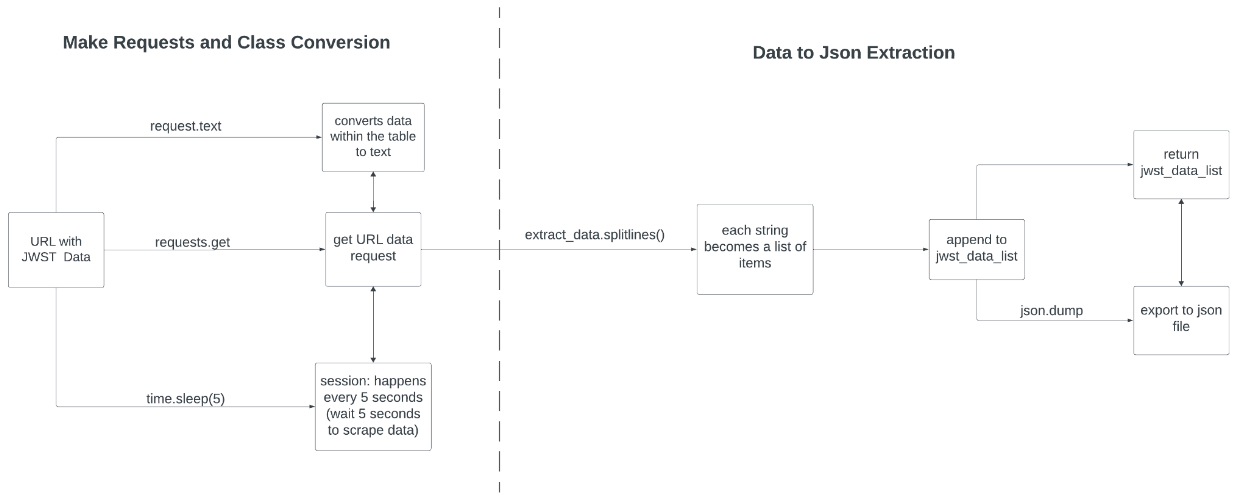
- **Role:**
    - The Mission Information Gatherer (MIG) Module is entrusted with the responsibility of managing and retrieving requests from the James Webb Space Telescope website, with a specific focus on gathering observing schedules data. Its primary role is to extract data from the website's observing schedules and organize it into a structured list format. Subsequently, this organized data is stored in a JSON file for efficient storage and further processing.
- **Function:**
    - The Mission Information Gatherer (MIG) Module assumes a pivotal role in facilitating access to mission-related data sourced from the James Webb Space Telescope website. By efficiently collecting and organizing observing schedules data, it enables researchers and enthusiasts to remain informed about upcoming missions and observations. Additionally, by providing the data in a structured JSON format, it facilitates seamless integration with other systems or applications for analysis and utilization.

2.5.3.2. Interface to other modules:

- The design rationale for the web scraper is centered on retrieving current data from the observing schedule of the James Webb Space Telescope website. This observing schedule data serves as valuable input for various modules, including the landing page and information about previous telescope images.

2.5.3.3. Static model:

2.5.3.4. Dynamic mode:



2.5.3.5. Design rationale:

The design rationale for the web scraper emphasizes the extraction of all of the data from the observing schedule of the James Webb Space Telescope website. Since each URL is needed, BeautifulSoup and the requests library are used to find the URLs with the ‘a’ tag and then limit the scope to finding any href link that ends in ‘txt’. Once those are retrieved, there is a second request that allows the retrieval of the data, converting it into a .text format for subsequent processing. Once retrieved, the data is appended to a list and returned. A secondary function is employed to write this data to a Txt file, while a tertiary function converts it into a Python list for further manipulation and analysis. Two more Python files are created, one for parsing the txt file and extracting it to JSON format and the other for reading the JSON file and extracting it to SQLite. This approach ensures efficient data extraction and processing, facilitating seamless integration with other modules within the software system.

2.5.4. MAST Validator (MASTv)Module:

2.5.4.1. Role and primary function:

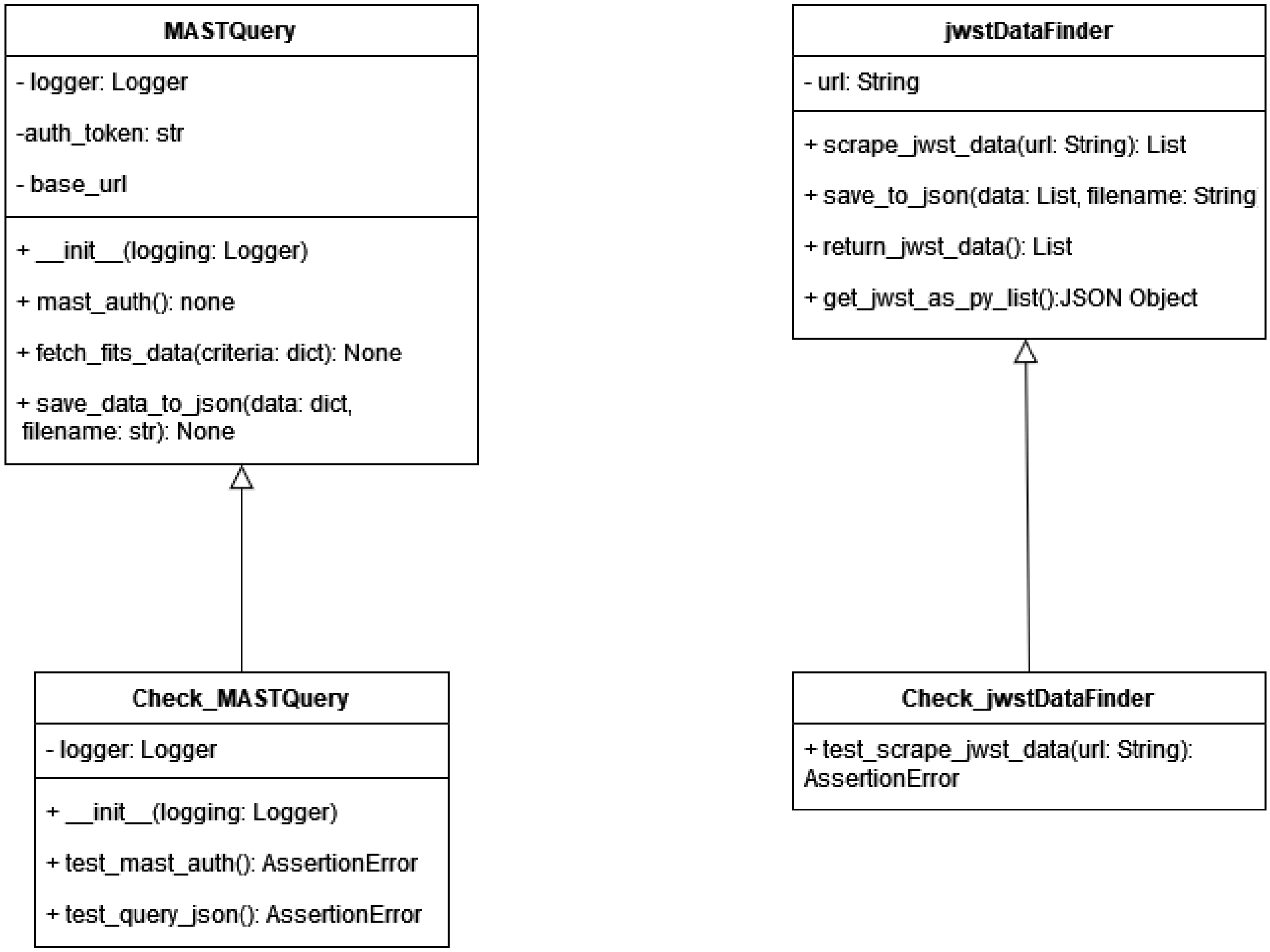
- **Role:**
    - The MAST Validator Module acts as a verification mechanism for both the Mission Information Gatherer (MIG) and Photo to Converter - First to PNG (FC-PG) modules to determine if the MAST API is functioning as intended for the current expected implementation and to verify that the JWST observation website is online. It accomplishes this by trying several different API requests by importing the FC-PG classes and running defined unit test cases. There is also a separate test to verify that the JWST observation website is online just before the MIG module runs.
- **Function:**
    - MASTv is a part of the system admin tools and runs during initial website setup or reset. It's intended to verify that the scheduled back-end Python programs work, but also to detect any changes that may have occurred resulting in bugs and issues, whether that be changes to the code implementation or the website/python library dependencies breaking or changing significantly in how they function.

2.5.4.2. Interface to other modules:

- The MAST Validator directly imports the PC-FG Module Python classes and runs several unit tests using the imported functions. There is no direct communication with the other modules, as the MAST Validator module only runs during the initial website setup or any reset processes.

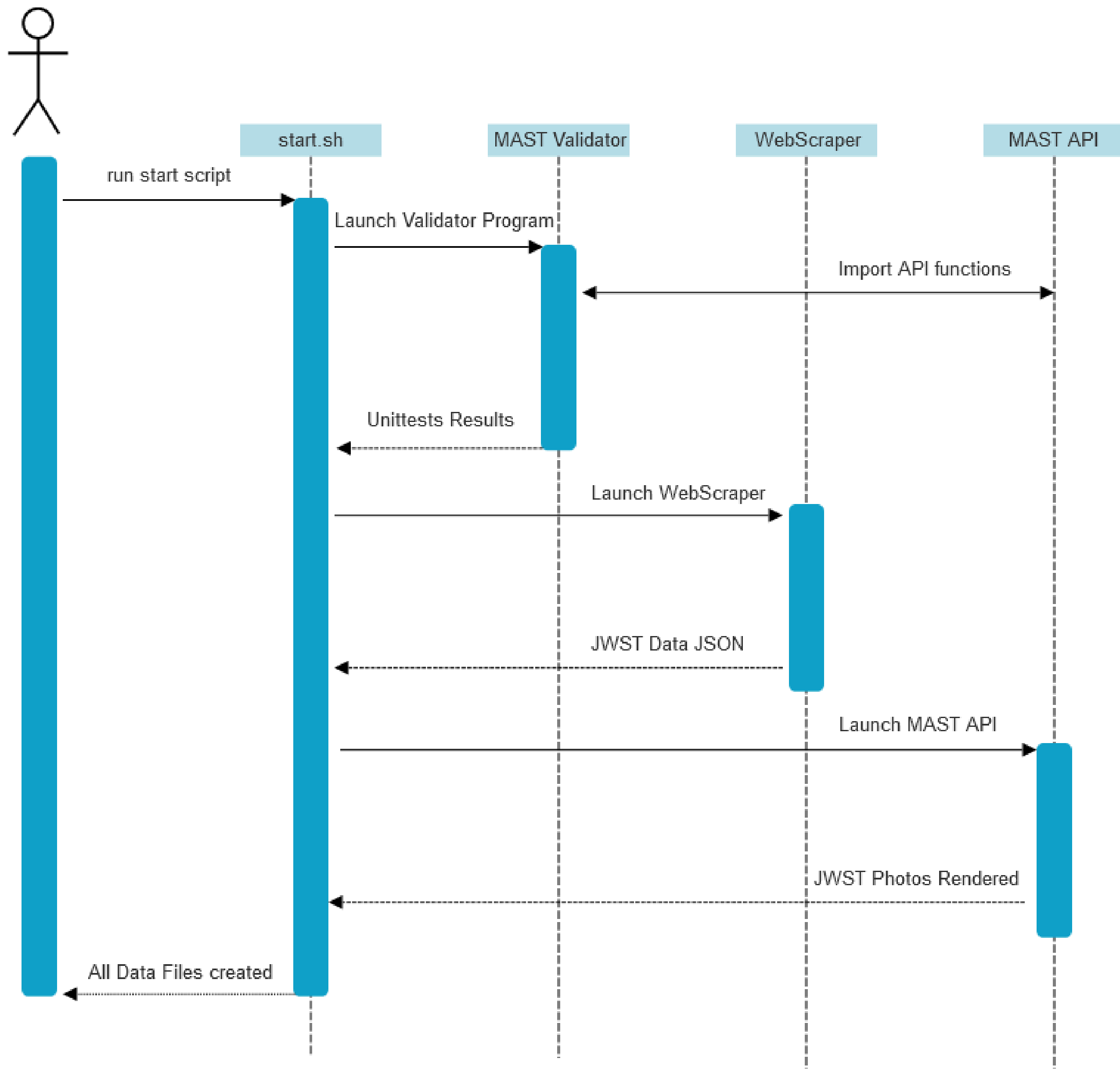
2.5.4.3. Static model:

UML Class Diagram for MAST Validator



2.5.4.4. Dynamic model (Sequence Diagram):

UML Sequence Diagram for Start Shell Script



2.5.4.5. Design rationale:

- Choosing Nose extends the default unit testing functions that Python provides. A majority of the group has prior experience with using the testing library from previous computer science courses. The Library also supports easy shell scripting and coordination if there are multiple different tests to run. Because the PC-FG & MIG modules are written in Python, MASTv should also be written in the same language to avoid programming language incompatibilities. The reason for the modules' existence is to verify that the functionality of the current module works as intended, help debug issues during the implementation step of the software design lifecycle process, and to detect any issues with dependencies that the project relies on such as libraries or websites/API's.

3. References:

- Faulk, Stuart. (2011-2024). CIS 422 SDS Template. Downloaded from https://classes.cs.uoregon.edu/24W/cs422/Handouts/Template-SDS.pdf
- Hornof, Anthony. (2024). CS 422 Project 2 Resources download form https://classes.cs.uoregon.edu/24W/cs422/P2/
- IEEE Std 1016-2009. (2009). IEEE Standard for Information Technology—Systems Design—
- Software Design Descriptions. https://ieeexplore.ieee.org/document/5167255
- Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. Commun. ACM, 15(12), 1053-1058.
- Sommerville, Ian. *Software Engineering*. Pearson Higher Ed, 2011.

4. Acknowledgments:

- The authors would like to express their gratitude to Stuart Faulk for his contribution to the initial template upon which this document is based, as well as to the authors of the publications referenced within, particularly IEEE Std 1016-2009. Additionally, the authors acknowledge Professor Hornof for providing the template used in the Software Methodologies course at the University of Oregon during the Winter 24 term, which served as the foundation for this document. Lastly, the JWST team for their contributions to science, space exploration, and our project.