Software Design Specification

for

MoonTrek Telescope

Version 1 approved

Prepared by

Kevin Aguilera
Pavit Chawla
Jacob Fausto
Alex Lamb
Nicolas Ojeda
Albert Ramirez
Gerard Rosario
Elvira Sakalenka
Dakota Townsend

NASA Jet Propulsion Laboratory (JPL)

05/14/2021

Table of Contents..... pg #1
Revision History.....pg #2
1. Introduction.....pg #3
1.1. Purpose.....pg #4
1.2. Document Conventions……..... pg #4
1.3. Intended Audience and Reading Suggestions..... pg #4
1.4. System Overview..... pg #5
2. Design Considerations..... pg #7
2.1. Assumptions and dependencies..... pg #7
2.2. General Constraints.....pg #7
2.3. Goals and Guidelines.....pg #9
2.4. Development Methods..... pg #9
3. Architectural Strategies.....pg #10
4. System Architecture..... pg #12
4.1. ..... pg #12
4.2. ..... pg #13
5. Policies and Tactics.....pg #14
5.1. Specific Products Used..... pg #14
5.2. Requirements traceability.....pg #14
5.3. Testing the software..... pg #14
5.4. Engineering trade-offs..... pg #14
6. Detailed System Design.....pg #15
6.1 User Interface..... pg #15
6.2 Communication..... pg #15
6.3 Registration ..... pg #16
7. Detailed Lower level Component Design…… pg #16
8. Database Design …… pg #19
9. User Interface..... pg #20
9.1. Overview of User Interface..... pg #20
9.2. Screen Frameworks or Images..... pg #20
9.3. User Interface Flow Model.....pg #21
10. Requirements Validation and Verification..... pg #23
11. Glossary..... pg #24
12. References..... pg #24

Revision History

Name	Date	Reason For Changes	Version
Draft 1	12/09/20		1.0
Final Draft	05/14/21		2.0

1. Introduction

1.1 Purpose

The main purpose of this document is to present a detailed description of the MoonTrek Telescope program version 2.0. It will cover all aspects of the software including the purpose and features, the interface of the application, what the system will do, and the constraints under which it will operate.

1.2 Document Conventions

Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.

- Main Paragraphs
    - As seen above bullet points and directly below numbered subtitles, indicate a description of bolded topics.
- Specific Components
    - As seen next to black bullet points, list specific parts of bolded topics
    - Specific components may also be presented using a (-)
- Specific Details
    - As seen next to hollow bullet points, describe specific components and their function/ role within the bolded topic.
- Outline Form
    - Some Main Paragraphs may be in the shape of an outline.
    - The form of this outline will usually be a numerical number, followed by lowercase letters, and then lowercase roman numerals.
- Acronyms
    - If used, will be written out fully once, and then followed by the first instance of the word's acronym.

1.3 Intended Audience and Reading Suggestions

While this software requirement specification document is written for a more general audience, this document may be directed towards individuals more involved in the development of MoonTrek Telescope. This includes software developers, project advisors, liaisons, and team managers. This document need not be read sequentially; users are encouraged to jump to any section they find relevant.

- Introduction
    - This section offers a summary of the MoonTrek Telescope project, including purpose, document conventions, intended audience and reading suggestions, and system overview.
- Design Constraints
    - This section describes many of the issues that need to be addressed or resolved before attempting to devise a complete design solution.

- Architectural Strategies
    - This section describes any design decisions and/or strategies that affect the overall organization of the system. It will offer clarity into the key abstractions and mechanisms used in the system architecture.
- System Architectures
    - This section offers a high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components.
- Policies and Tactics
    - This section describes any design policies and/or tactics that do not have sweeping architectural implications. These will affect the details of the interface and/or implementation of various aspects of the system.
- Detailed System Design
    - Each subsection of this section will refer to or contain a detailed description of a system software component.
- Detailed Lower level Component Design
    - This section will describe other lower-level Classes, components, subcomponents, and assorted support files.
- Database Design
    - This section will include details about any databases used by the software.
- User Interface
    - This section is all about the UI design from the user's perspective. It will talk about the restrictions, constraints, refinement of the user experience through the application.
- Requirements Validation and Verification
    - This section offers a table that lists each of the requirements that were specified in the SRS document for this software.
- Glossary
    - This section provides definitions for any relevant terms, acronyms, and abbreviations that are necessary to comprehend this document.
- References
    - This section lists any other documents or Web addresses to which this SDD refers.

1.4 System Overview

The Telescope MoonTrek project, TMT, is meant to be a user-friendly version of the current NASA MoonTrek web application, NMT. This project has two parts which work in conjunction to execute similar functions and produce results with similar accuracy to the NMT. The first part of this project in terms of visibility is the interface which is viewable on a web browser. This interface is meant to receive a user image from a telescope, after which data layers can be added or api calls can be made so as to give a user more information about the moon image submitted by the user, these api calls and layering will be done through interface buttons, which may be clicked by a user. The second part of the project involves image registration, this part is meant to take a user image and identify the parts of the image that are actually of the moon, this is meant

to ensure that the data layers and api calls are set on top of the moon's image and not all over the interface. The image registration is also meant to determine whether a submitted image can be used by the application interface, if the image is usable it will show up in the interface window, if not a message will be printed prompting a user to submit a different image to be registered. The final part of the project is a 3D model, implemented in the back end which is meant to clean up distortions from the user image to get the most accurate results possible. However, after building the final iteration of the project, the team leader decided to implement the 3D model as a way for users to test data layering on a visual medium, if an individual is not able to provide appropriate images for the application.

2. Design Considerations

This section describes many of the issues that need to be addressed or resolved before attempting to devise a complete design solution.

2.1 Assumptions and Dependencies

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

- Celestron Telescope (Unknown model)
    - Unknown models make compatibility difficult.
    - Whether the team will actually have access to a telescope, due to covid 19
- User Telescope (Unknown entity)
    - We will know a lot more about the celestron telescope than the user's telescope, whose compatibility will be more difficult to gauge.
- Cross-Browser Functionality
    - No cross-browser compatibility tests have been done so far, so applications running on different browsers is an assumption.
- NASA's MoonTrek API
    - Has existing calls that meet functional requirements.
    - Whether individual components have the ability to call JPL's APIs

2.2 General Constraints

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

- Hardware or Software Environment
    - Lack of django experience: May limit the functionality of the application to what can be executed properly instead of what is expected.
    - Lack of 3D modeling experience: May skew accuracy of image registration on sphere object
    - Lack of image registration experience: Image registration may not function properly on all images
    - Lack of telescope knowledge: May have trouble getting application to function alongside certain types of telescope.
    - Lack of software intersectionality awareness: We don't know how well the components will come together in django.

    - Lack of server experience: Don't know how well the application will behave once implemented in a host's server.
- End-User Environment
    - Cannot determine if the interface is clearly seen by the user on their computer, may have to make the interface with lighter colors to make it easier to navigate in the dark and create contrast so it isn't difficult to see in sunlight.
    - Cannot determine how the styling of the page looks in different computer screens. Some components may not look as great depending on the specifications of a user's display.
- Availability Or Volatility Of Resources
    - Cannot determine how the end-user's computer will handle the costs of running the application on their computer, may have to make the application more cost-effective once delivery-date comes closer.
- Interoperability Requirements
    - Must ensure that image data is being accepted in the application so data layers can be added.
    - Must ensure that connection to telescope can be verified so user can submit images to application
- Interface/Protocol Requirements
    - Must make the interface as user-friendly as possible, since there is no guarantee that users have already used the NASA MoonTrek application.
    - Must create alerts if certain inputs from a user are invalid or inoperable, so that the application doesn't crash.
- Data Repository And Distribution Requirements
    - Software aims to store locally as a means of privacy and prevents having to require various permissions that would otherwise be required.
- Security Requirements (Or Other Such Regulations)
    - Staff or people in charge of maintaining software must uphold integrity and discretion as noted in the Association for Computing Machinery (ACM), code of ethics.
- Memory And Other Capacity Limitations
    - May need to shrink the media file to increase speed of application, as multimedia files often use a lot of memory.
- Performance Requirements
    - Must be able to execute API calls.
    - Must be able to run Python scripts

- Network communications
    - Access to the application is as dependable as the internet connection. May need to make certain functions available offline
- Verification and validation requirements (testing)
    - Must ensure that inputs other than telescope images will not be accepted, and will not crash the application.
    - Must ensure that the user knows why certain inputs are not accepted.

2.3 Goals and Guidelines

- Designing a user friendly, efficient, and intuitive UI that encourages amatuer astronomers to cultivate their curiosity of the moon.
- Developing a UI that bridges the gap and serves as the connection between the backend development, telescope, and user.
- The application is an extension and user-friendly version of the existing NASA MoonTrek portal.
- The application is able to perform image registration to gather appropriate images for data layering and api annotations.
- The application has a 3D model to demonstrate data layering for user's with no usable images.
- The application has a mandatory delivery date that must be met. (by May 2021)

2.4 Development Methods

The development method that our MoonTrek Telescope team most closely resembled using was an Agile Development method. This approach values individuals and tends to accommodate for change. We reacted to change and developed our code in pieces typically building off the previous week's goals. We worked collaboration-heavy, meeting at least twice a week to review our progress and move forward focusing on team strengths and efficiencies along with building off feedback from our advisor and liaisons, weekly. The Agile Development method highlights the importance of client collaboration throughout the development process, responds to change instead of following a concrete plan, and it focuses on presenting working software with less emphasis on documentation.

3. Architectural Strategies

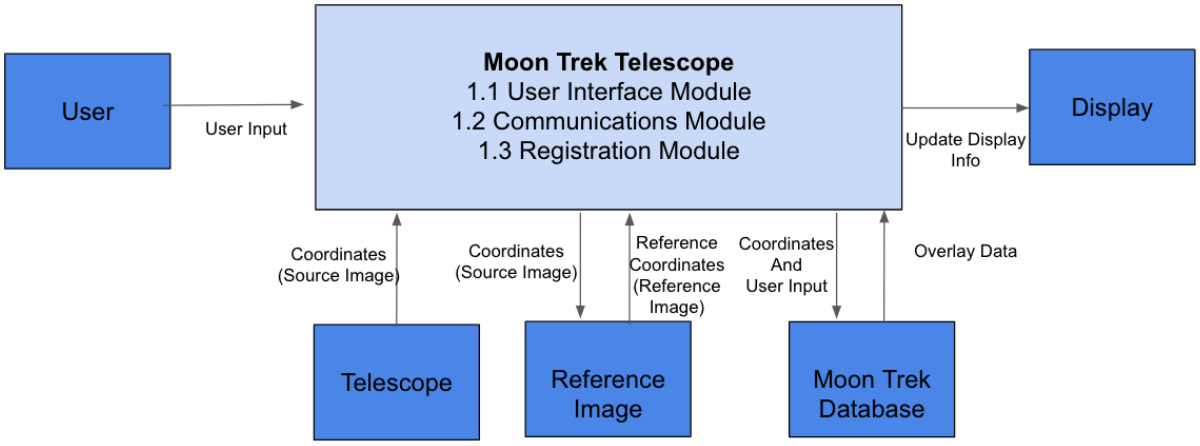
The application will take into account image registration of the image provided by the telescope . The design of the software needs to include an architecture that will register the image while fetching data once after the image is registered . The front end will just shoot the image of the moon routed from the telescope , while the back end will be executing several processes . The design will require us to implement these processes at rapid speed so the main feature of adding overlays can be executed as fast as possible . The processes in order will be such as ;

1. Route image into browser/application view
2. OpenCV circle detection on the moon
3. Image registration of source image and reference image
    a. Recreate an image of the moon that closely resembles the source image .
        i. Image recreated will contain
            1. LRO WAC Mosaic map of the moon from JPL
            2. Right shape of the moon that closely resembles the source image
            3. Right shadow and light from sun .
        ii. Image recreated will be a snapshot of a 3D model of the moon. The reason we create this reference image and the way we do it is to get rid of geometric distortion that otherwise would be present in just registering to the LRO WAC Mosaic of the moon from JPL's Moon Trek portal .
4. Once each pixel coordinate of the image from the telescope(source image) is located in the LRO WAC Mosaic 3D model snapshot (reference image) , the application will require the fetching of data of the moon regarding those specific coordinates of the moon in the source image .
    a. Data to fetch will be such as ;
        i. Latitude and longitude
        ii. Crater
        iii. Mare
        iv. Landing sites
        v. etc .
    b. The data will be displayed on the users image on the client side . The image is placed on a canvas and then the annotations are placed.

- The products to be used ;
    - Python
    - SQLite
    - Django
    - OpenCV

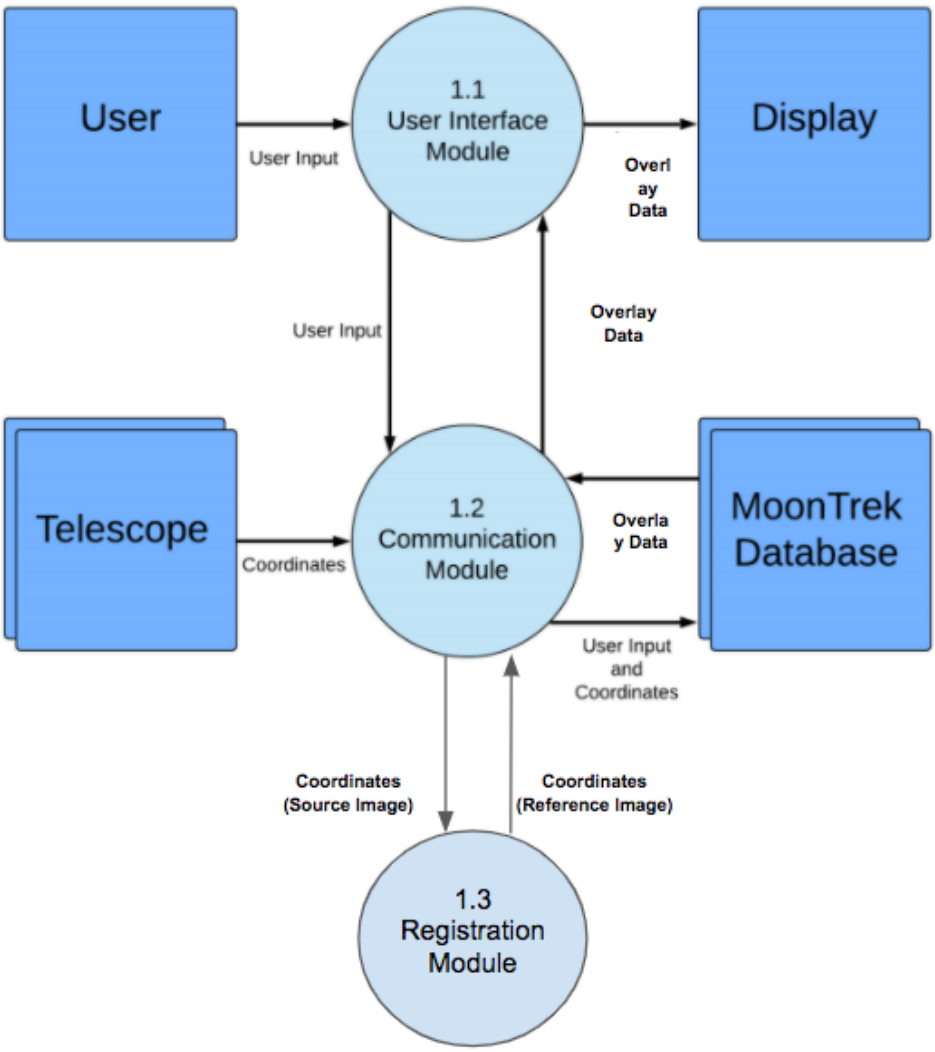
- Threejs
- JPL Moon Trek

4. System Architecture



DFD Level 0

The two main modules of the Moon Trek Telescope are the User interface module and the communications module . The user interface module is in charge of displaying the information gathered from JPL's database . This information can be annotations or overlays relative to the graphical coordinates of the moon present in the user's view . The registration module will be in charge of properly locating each coordinate in the source view of the moon to the reference image of the moon . The communications module would allow us to reach the JPLs datasets of the moon and fetch them to use them in our user interface module.



DFD Level 1

The system is composed of mainly three modules ; user interface , communications and registration module . The main modules doing the work are the communications and registration modules .The telescope image will be provided to these modules and execute important tasks . Image from the telescope will be passed to the registration module where all image registration procedures will take place . The right coordinates will be generated and then passed back to the communication module . The communication module will fetch overlay data with the correct produced coordinates implemented from the registration module. Then the data will be passed to the user interface module in order to display it to the users view of the moon.

5. Policies and Tactics

5.1 Choice of which specific products used

For development, the following were used -

- Django - “Python-based free and open-source web framework that follows the model-template-views architectural pattern.”
- Three.js - “JavaScript library and application programming interface used to create and display animated 3D computer graphics in a web browser using WebGL.”
- OpenCV - “A library of programming functions mainly aimed at real-time computer vision.”
- WebGL - “A JavaScript API for rendering interactive 2D and 3D graphics within any compatible web browser without the use of plug-ins.”
- SQLite - “a relational database management system contained in a C library.”

5.2 Ensuring requirements traceability

1) Requirements listed in the Software Requirements Specification (SRS) and Software Design Specification (SDS) served as roadmaps or checklists for what we set out to implement.
2) Our team Communications Lead took notes during meetings and distributed the notes among the team immediately after meeting.
3) Our Team Leader assigned tasks to individuals which we each took note of.
4) Our team communicated through Slack.com so any ideas of requirements discussed in chat rooms were available to look back at anytime.
5) We were guided by an advisor and liaisons who explained what our requirements were depending on the progress we had made.

5.3 Testing the software

Software testing occurred during the development of the software, as the team needed to constantly ensure that API’s used for calculations within the code were behaving as expected. Also, when compiling/running the code in a testing environment, to ensure desired images and results loaded on the screen.

The software which is deployed on JPL servers was also tested to ensure all functionality stayed the same as with the testing environment.

5.4 Link to software (how to compile, link, load, etc.)

Our software can be obtained by cloning the project directories somewhere in a local computer

Access to our repo : https://github.com/nicocoa10/MoonTrekTelescopeV2

The file “installation_instructions.txt” should direct any future development teams on how to install the project in a local machine .

The project can also be run through JPL servers but for this JPL will need to activate the server where the project is installed .

The project is installed in IP address : 54.157.167.17 and port : 8000

To run it from a local browser the applied address would have to be http://54.157.167.17:8000/ and the application should appear after JPL has activated the server.

6. Detailed System Design

6.1 User Interface Module (UIM)

6.1.1 Responsibilities

The UIM works to display an image of the moon and the statistical data based on geographical portions of the moon from the user's given position on Earth. The UIM is configured to connect either through smartphone and a telescope or by smartphone and user input. The UIM overlays informative annotations over the portion of the moon being displayed based on the user's geographical location based on longitude and latitude.

6.1.2 Constraints

The UIM is expected to work as long as the user has a valid and steady internet connection, and would otherwise be non-functional without an internet connection. Without the use of a connectable telescope, this application will be provided a set of hard coded data that may not accurately represent the moon from the user's location on Earth.

6.1.3 Composition

The UIM acts specifically to relay information to the user and connects with the Registration Module (RM) and the Communication Module (CM) in order to display the data to the user. The UIM receives the overlay information that the CM obtains from the RM. The UIM then displays the overlay data to the user via smartphone or monitor display.

6.1.4 Uses/Interactions

The user interactions with the UIM include a drop-list that can display various statistical data from JPL's database. Assuming the user is using a monitor with mouse and keyboard, the user will use the mouse to select the type of information of the moon that they select from the drop-list provided.

6.1.5 Resources

The UIM uses JPL's database to obtain data about the moon that relays through the CM and RM to display recorded data of the moon to the user.

6.1.6 Interface/Exports

The UIM displays information to the user with a drop list on the left side of the display window and the rest of the window displays the moon and the related data based on user input with annotations related to the user's selection from the drop-list.

6.2 Communication Module (CM)

6.2.1 Responsibilities

The CM works as a link between the UIM and the RM and allows the software to fetch necessary data from the RM and the JPL databases and then relay that information to the UIM. The primary use of the CM is as a bridgeway for data to be displayed to the user using the user's telescope geographical coordinates and the MoonTrek Database.

6.2.2 Constraints

Based on internet connectivity and speed of JPL's databases, the CM could take longer times to relay information with any disruptions to connectivity. The hardware and software of connectable telescopes can have a partial delay in receiving information.

6.2.3 Composition

The CM receives the coordinates of the user on Earth and then connects with the RM to obtain a reference image. The CM then takes the user's input, coordinates, and the reference image from the RM and runs these components through the Moontrek database to then send information from the database to the UIM to display to the user.

6.2.4 Uses/Interactions

The user does not directly interact with the CM. The CM interacts with the user's telescope and uses the coordinates to interpret necessary data and relay the information. When the user is accessing Moon Trek Telescope without a connectable telescope, the CM receives a hard coded location of the user instead.

6.2.5 Resources

The CM connects with the user's telescope to obtain location coordinates and connects with the MoonTrek Database to interpret necessary data to send to the UIM for display. The CM interacts with the RM to collect a reference image based on the user's coordinates on Earth.

6.2.6 Interface/Exports

The CM exports data received from the MoonTrek database to the UIM for display. The CM exports geographical coordinates, received from the user's Telescope or from hard coded values, to the RM to interpret a reference image.

6.3 Registration Module (RM)

6.3.1 Responsibilities

The RM works to interpret a reference image based on coordinates received from the user's telescope that is exported by the CM. The RM properly locates each coordinate of a source image and creates a reference image for display.

6.3.2 Constraints

The RM is reliant on the user's coordinates in order to relay information. The user's geographical coordinates are necessary for the RM to function.

6.3.3 Composition

The RM relies on the source image and is in charge of properly locating each coordinate in the source view of the moon to the reference image of the moon.

6.3.4 Uses/Interactions

The user does not directly interact with the RM. The RM interacts with the user's input of coordinates to produce a reference image for display.

6.3.5 Resources

The RM needs a source image and the geographical coordinates of the user to display necessary data correctly.

6.3.6 Interface/Exports

The RM exports a reference image determined by a source image based on coordinates of the user. The reference image is exported to the CM to be set for display by the UIM.

7. Detailed Lower level Component Design

Not currently available.

8. Database Design

The MoonTrek Telescop application shall use Nasa MoonTrek portal database and API to retrieve moon data.

To process the user moon images we use a basic SQLite database to store references to images .

9. User Interface

9.1 Overview of User Interface

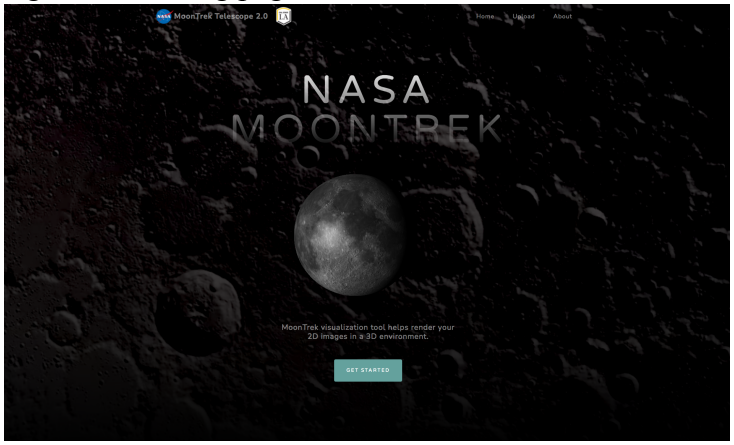
Describe the functionality of the system from the user's perspective.
Explain how the user will be able to use your system to complete all the expected features and the feedback Information that will be displayed for the user.
This is an overview of the UI and its use.
The user manual will contain extensive detail about the actual use of the software.

- Functionality of the system from the user's perspective
    - Users will interact with 2 buttons on the landing page: "Choose File", and "Upload".
    - "Choose File" will allow a user to select an image to upload.
    - "Upload" will allow the user to upload their selected image.
- Feedback information that will be displayed for the user
    - Rendered and tagged model corresponding to their photo
        - Any points of interest that our imaging algorithm finds will be displayed to the user.
- UI and its Use
    - The User Interface will display information and models back to the user.
    - It is interactable through buttons
    - Follows a linear approach that guides the user through the upload process

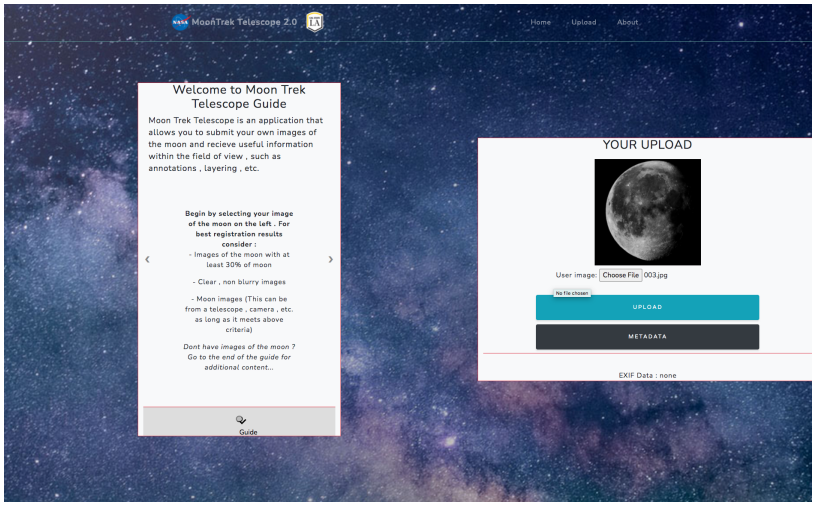
9.2 Screen Frameworks or Images

The attached figures are for demonstrative purposes, they may not be accurate representations of what our final product will be.

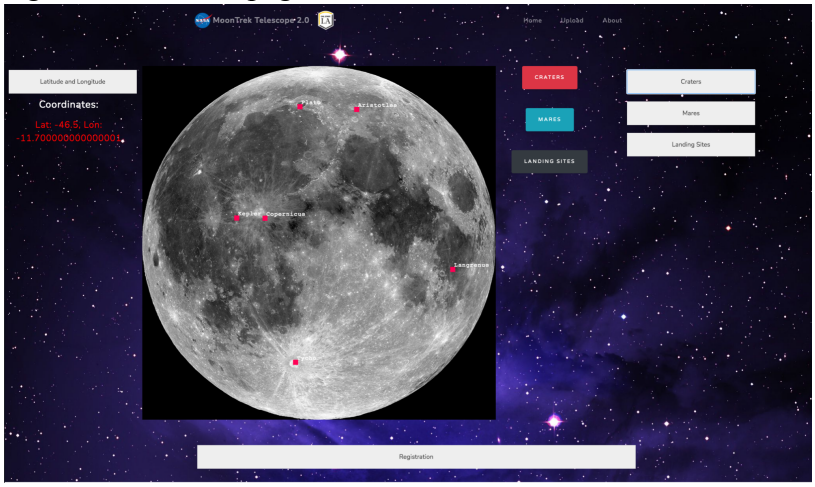
- Figure 1: Landing page for the Moontrek website.



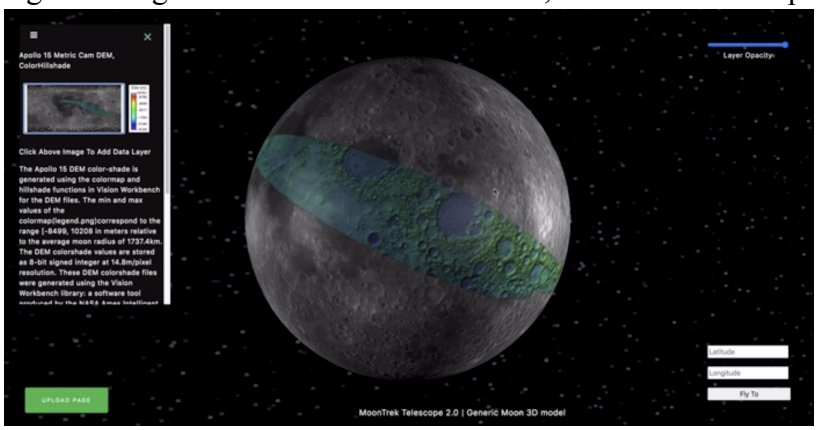
- Figure 2: The upload page



- Figure 3: The final page that contains annotations like craters, layers, mares, etc.



- Figure 4: A generic 3D model of the moon, for users without proper images.



9.3 User Interface Flow Model

A discussion of screen objects and actions associated with those objects. This should include a flow diagram of the navigation between different pages.

- Not currently available.

10. Requirements Validation and Verification

Requirements	Modules/UI/Components and Testing Methods
The application shall take in a visual input from the user	● Communication module and User Interface ● Running application
The application shall query and display different layers of data from the MoonTrek API and return specific details.	● User Interface ● Running the application trying out different layers of data
The application shall limit user input to available data layers.	● User Interface ● Run the application to see what layers user can see
The application shall be able to overlay layers and change opacity for visualization convenience.	● User Interface ● running application trying out different layers and opacity
The application should allow user search for specific data set	● User Interface ● Search for some data set
The application should be able to allow user to change the scale of the returning data layer	● User Interface ● Try to change the scale of the data layer

11. Glossary

SRS - Software Requirements Specification
SDD - Software Design Document
JPL - Jet Propulsion Laboratory
UI - User Interface
WAC – Wide Angle Camera: often used with lunar photography
LROC - Lunar Reconnaissance Orbiter Camera: often used with lunar photography
TMT - Telescope MoonTrek
NMT - Nasa MoonTrek
ACM - Association of Computing Machinery

12. References

Software Design Document Template: provided by CSULA on CSNS

ASCOM Standards https://ascom-standards.org/

MoonTrek API: https://trek.nasa.gov/tiles/apidoc/trekAPI.html?body=moon

Django Web Framework: https://docs.djangoproject.com/en/3.1/

OpenCV : https://docs.opencv.org/master/d6/d00/tutorial_py_root.html

ThreeJS : https://threejs.org/docs/