Software Design Document

for

MoonTrek: Telescope

Augmented Reality

Version 2 approved

Prepared by:

Nadir Abdusemed	Jackson Bentley
Jesus Cruz	Youssef Elzein
Derek Guevara	Joe Hineno
Rich Ho	Owen Ramirez
Salman Sheikh	Alex Sherzai

Sponsored by:

NASA Jet Propulsion Laboratory

05/12/2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
First Draft	12/05/2022	Template Creation	2.0
Update	12/11/2022	Updated document from previous semester	2.1
Fall Final Draft	12/12/2023	Completed documentation with updated information	2.2
Final Draft	5/12/2023	Updated with new information	2.3

1. Introduction

1.1 Purpose

The purpose of this document is to give informative documentation of the web application MoonTrek: Telescope Augmented Reality, hosted by Jet Propulsion Laboratory. This document will go over how to use the web application MoonTrek as well as the constraints, abilities, and features the application will operate on.

1.2 Document Conventions

Describe the standards or typographical conventions followed when writing this documentation, such as fonts, highlighting that have special significance, or listing conventions.

- Main Paragraphs
  - As seen above bullet points and directly below numbered subheadings, indicate a description of topics.
- Specific Components
  - As seen next to black bullet points, list specific parts of bolded topics.
- 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, it will be written out entirely once and then followed by the first instance of the word's acronym.
- Typefaces
    - The paper will primarily use one font.
    - Italics, denote titles and names of works or objects to allow titles or names to stand out and not be used overtly.
    - Bold, font is used to denote a heavier weight of the significance; Used in titles and to show importance.
- Margins
    - The document shall follow Modern Language Association (MLA) format and incorporate 1-inch margins on each section.

1.3 Intended Audience and Reading Suggestions

This software requirement specification document are written for the general audience however this document can also be directed to audiences that are involved in the development of MoonTrek: Telescope Augmented Reality development. The intended audience can be software developers, project advisors, liaisons, team managers, or whoever is involved with the development of MoonTrek for upcoming years.

- **Introduction**
    - This section summarizes the 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 devising 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 clarify the fundamental 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 Tactic**
    - This section describes 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.
- **Requirements Validation and Verification**
    - This section offers a list of each requirement specified in the Software Requirements Document (SRD) for this software.
- **Glossary**
    - This section provides definitions for relevant terms, acronyms, and abbreviations necessary to comprehend this document.
- **References**
    - This section lists any other documents or web addresses to which this Software Design Documents (SDD) refers.

1.4 System Overview

The System Overview provides a high-level user understanding of the non-technical functions of MoonTrek. Later in this document, there is more in-depth on the technical aspects of the application MoonTrek.

- It is built with Express and Vue.js. On the MoonTrek web application, the user shall capture an image of the Moon from their telescope or upload a corresponding image into the web application.
- The user shall give the image to the web application; then, in turn, the web application provides points of interest on the Moon such as craters, maria, and landing sites.
- The augmented reality portion of the project improves the data overlays, creating a 3D model of the Moon created by Jet Propulsion Lab's high-quality images and user-uploaded images of the Moon.
- The team's objective is also to complete the telescope for computer communication, improve the accuracy of the image registration, and create a 3D model of the Sun, Moon, and Earth based on the images uploaded.
- The same 3D model of Earth will also annotate where the picture was taken and its time.

2. Design Considerations

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

2.1 Assumptions and Dependencies

- Users Input

  - We are assuming that the users are uploading images that contain moons or that there is a live video feed with the moon in it and that those images are taken from somewhere on earth.
- Metadata
  - We are also dependent on them providing us information on when the image was taken and where it was taken. For this, we will be extracting their metadata
- Nasa APIs
  - There are several Nasa APIs we are dependent on that provide information regarding pixel data, overlay data, and central coordinates of the user's image
- 3D model
  - The 3D model we will create is dependent on the above-mentioned API calls. The context-aware image registration will, in turn, be dependent on this 3D model
- Coordinates
  - In order to map the overlays into the correct place on the user's image we will need to have coordinated with a one-to-one correspondence from the WAC composite image to the users image

2.2 General Constraints

- Users image quality
  - In the image registration process we are constrained by the quality of the user's image. The quality of the image can be degraded for a number of reasons including pixel count, the orientation of the camera, the quality of the telescope, telescope maintenance, lens type, atmospheric conditions, light pollution, right ascension and declination, and many others. There are future plans to supplant the user's image with higher quality images but we will not likely implement that this year
- JPL API calls
  - We are limited in the amount and accuracy of the overlays by the amount and accuracy of the data from the JPL Moontrek site. Additionally, the accuracy of the 3D model is constrained by the calculations made by JPL to determine that central point.
- Meta Data
  - Accuracy of the 3D generated image will be dependent on accurate metadata from the users, or accurate information on the upload form
- Time constraint
  - We are also constrained by the remainder of the time period we have to work in this semester, the remainder of the spring semester. We will set up dates using the agile work methodology in order to optimize our time usage and testing cycles.

2.3 Goals and Guidelines

- Designing a user-friendly, efficient, and intuitive UI that encourages amateur astronomers to cultivate their curiosity about 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 Moon Trek portal.
- Create a 3D model of the moon centered where the user's image will be centered using the location and timestamp of the uploaded image.
- Create a one-to-one mapping of coordinates from the WAC composite image to the user's image
- The application can perform image registration to gather appropriate images for data layering and API annotations.
- Use a 3D modeling JavaScript library to replicate the positioning of the Moon, Earth, and Sun systems.

2.4 Development Methods

To develop this web application we will mainly be using the Agile methodology, following a similar model from the previous group. We will be working in two-week sprints with weekly meetings with our advisor and biweekly meetings with our liaisons. A Schedule of deliverables will be made before the start of the semester and we will push to develop the most critical parts first in order to begin testing out implementations.

3. Architectural Strategies

3.1 Database

- Allow users to upload their moon image.
- Input the metadata attached to the image.
- Upload user lunar image and metadata to lunar image database.
- User image cropping for accurate moon detection.
- MySQL database for efficient data storage and retrieval.
- JavaScript for seamless image uploading and manipulation.

3.2 3D Model

- Create a 3D model based on the user's moon image.
- Create 3D models that represent the Moon, Earth, and Sun based on their positions of the user's location and time.
- This is done using a JavaScript library called Three.js, a library that allows us to create and display animated 3D computer graphics in a web browser.

- Three.js has the capability of performing texture mapping with a Lunar Reconnaissance Orbiter Wide Angle camera that is provided by JPL.
- Three.js also implements directional lighting onto a 3D model to cast shadows.

3.3 Image Registration

- Identify points of interest on the Moon.
- Discover points of interest using circle detection and an algorithm called SIFT, an algorithm that detects and matches locations on the Moon to plot points of interest.
- After performing context aware image registration, plot points of interest with information such as craters, landing sights, and Lunar maria and apply overlays in the correct positions

3.4 Coordinate Mapping

- Map from a flat surface to a sphere
- Account for transformations due to rotations about any axis

3.5 Team Leads

- Collaborate and communicate with all four teams to keep everyone on track for the objective of the project.
- Assist teams to navigate with the Vue.js web framework, and uploading to Git.
- Provide additional aid to each respective team.
- Provide leadership and guidance for the project team.

4. System Architecture

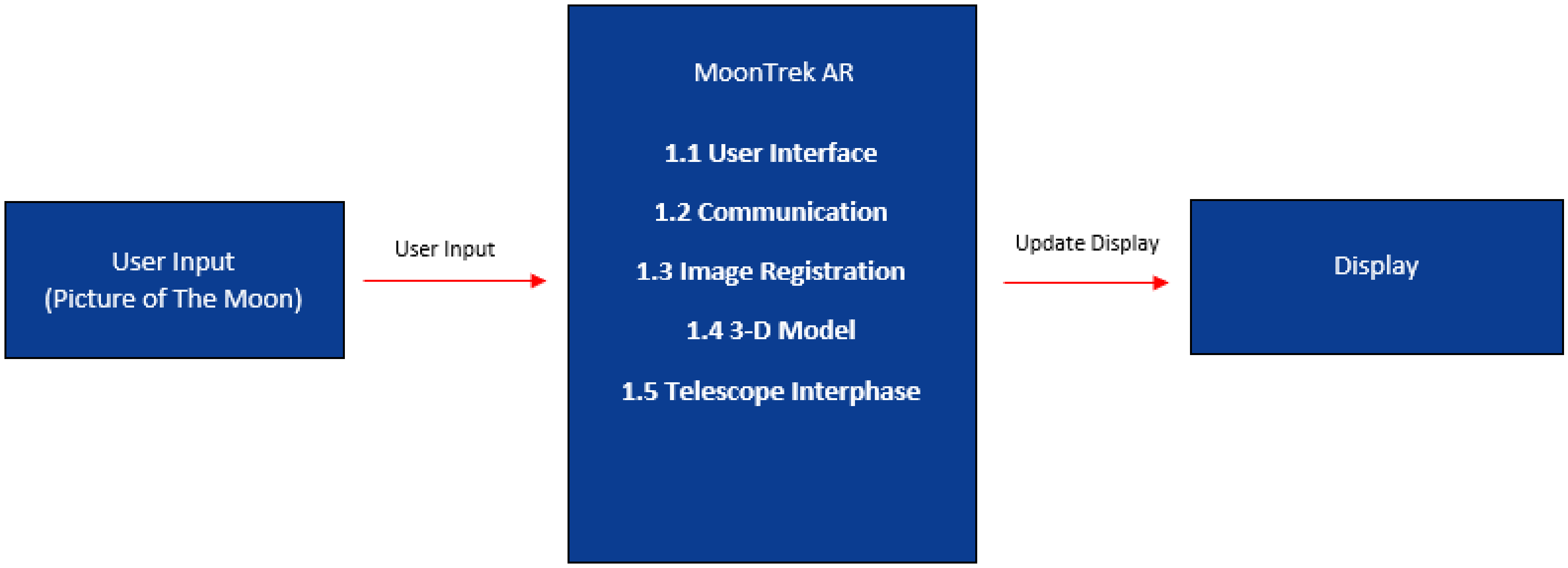


Figure 1: MoonTrek Data Flow Diagram 0 (DFD 0)

The broad view of this project is that we get a picture and we input it through our project and it will return an overlay using image registration, a 3-D model, and a live feed overlay using telescope integration. All of this would get displayed by the UI module. The communication module is there to communicate through the different modules and the JPL server that I will talk about in the next DFD.

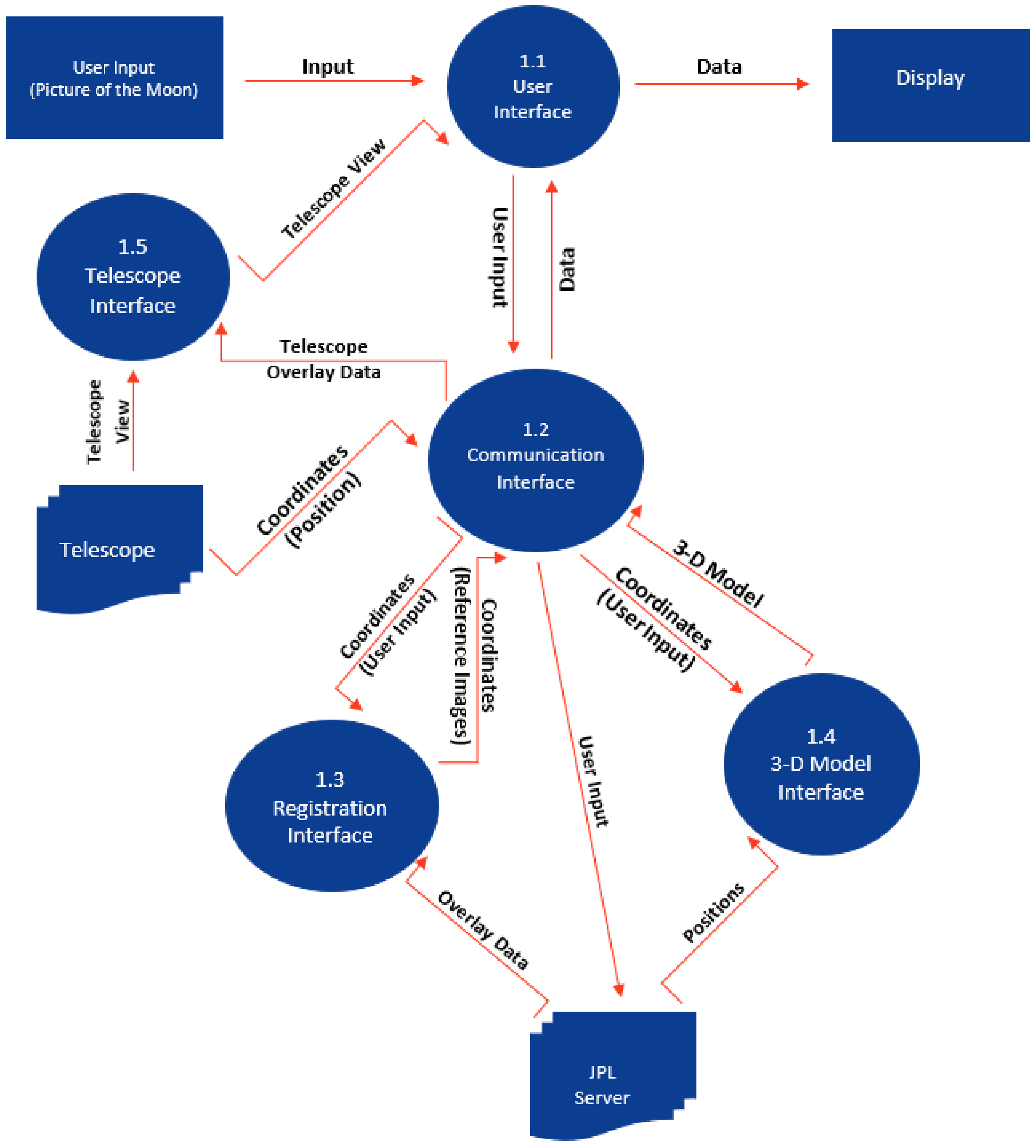


Figure 2: MoonTrek Data Flow Diagram 1 (DFD 1)

The figure above shows the layout of our project in a little more depth. The UI interface is the main module that will display the results that will be generated from the data gathered by the communications module. The communications module is the module where it will give and grab data that was received by the User and the 3 modules are Registration, 3-D model, and Telescope Interface. The Image Registration module will grab the user input and the overlay data given from the Server and, shift and flip accordingly to the user input. The 3-D model creates a 3-D model based on the coordinates and position given by the JPL server and the comms module to create a 3-D model of the earth, sun, and moon using the positions given. The telescope module will grab a live feed of the moon and apply overlay data received by the JPL server.

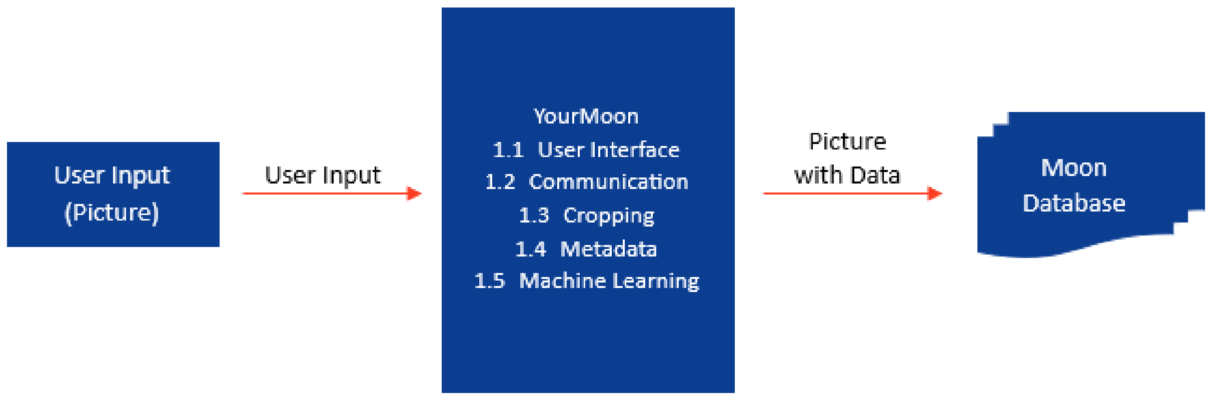


Figure 3: YourMoon Data Flow Diagram 0 (DFD 0)

The broad view of this project is that we get the user's cropped moon image and the metadata, then send it to the server. The communication module is there to retrieve the information from the metadata entry and the cropped image. It also communicates with the machine learning module for validation. At the end, it will be sent to the moon database module. More information will be discussed in the next DFD.

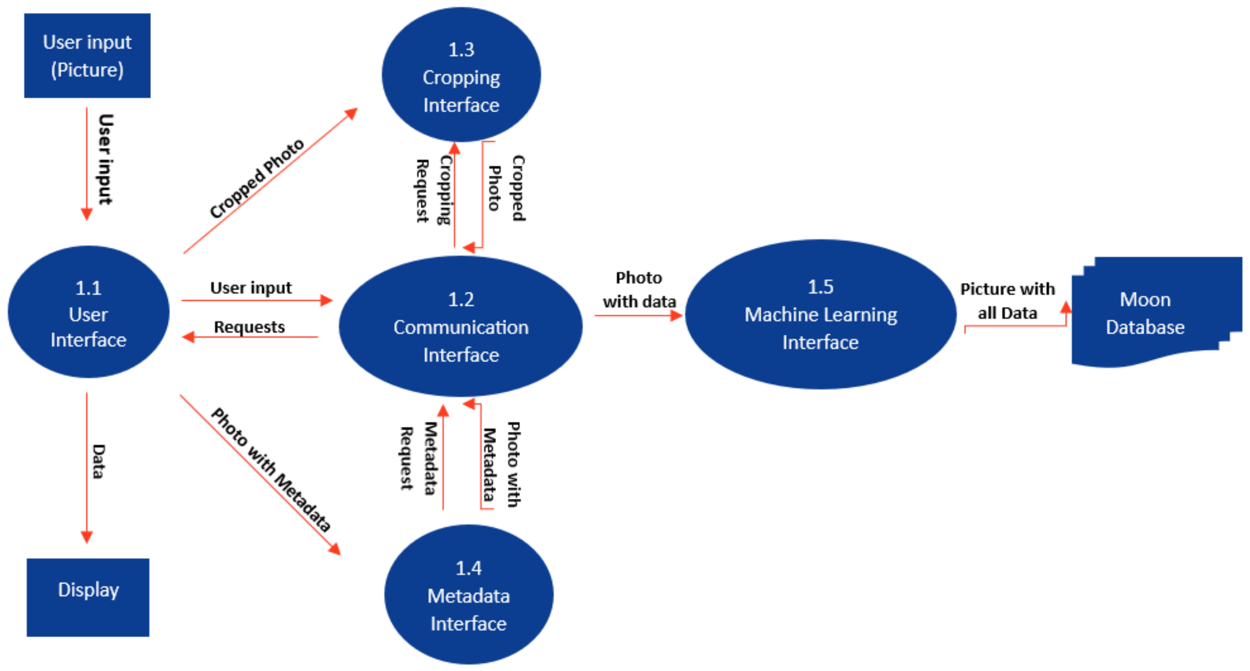


Figure 4: Your Moon Data Flow Diagram 1 (DFD 1)

The figure above shows the layout of *YourMoon* in depth. The UI interface is the main module that will display the file upload and metadata input form, which is also the metadata interface. When the user selects and image to upload, the cropping interface will pop up so that the image can be cropped to only the moon. The communications module is the module where it will grab the cropped image and metadata, then send it to the machine learning interface. The machine learning interface is where a model will determine whether the image is an image of the moon or not. Once the moon image is validated, the image will be sent to the moon database.

5. Policies and Tactics

5.1 Choice of which specific products are used

Our application is NodeJs based. We chose to use Express.Js for the back-end and VueJs for the font-end

5.2 Plans for ensuring requirements traceability

We will keep in regular contact with our JPL liaisons so that we are constantly aware of any changes in requirements or feedback on our attempts to fulfill them.

5.3 Plans for testing the software

The new web design will be a lot more modular which will allow us to test the code in sections and allow for more efficient debugging. We will be bug-testing the software using Visual Studio and GitHub. There are also plans to create a large database of images from different parts of the world at different times in order to find edge cases that might throw off the image registration. We will also create models that allow for greater lunar librations than actually occur to insure that our system works properly. We will be following standard best practices when it comes to programming conventions and guidelines.

6. Detailed System Design

6.1 User Interface Module (UIM)

6.1.1 Responsibilities

The UIM displays VueJs routes to allow the user to interact with the MoonTrek project. The UIM is configured to be viewed on a smartphone, tablet, or computer. The UIM accepts the user's input images and shows a form if the image metadata does not include the required time and date information.

6.1.2 Constraints

The UIM is expected to work as long as the user has a steady internet connection and would otherwise be non-functional.

6.1.3 Composition

The UIM involves multiple .vue files, including ConnectPage.vue, HomePage.vue, ModelPage.vue, and UploadPage.vue, which create the HTTP responses for each page that the user will access. These pages then reference component files such as AnimatedStars.vue, ImageCanvas.vue, ImageUploadForm.vue, and NavBar.vue.

6.1.4 Uses/Interactions

The UIM receives the overlay information about the image registration results from the RM. The UIM then displays the overlay data to the user via HTTP response.

6.1.5 Resources

The UIM utilizes the user's web browser, as well as the GPU in order to render images for the viewer to see.

6.1.6 Interface/Exports

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

6.2 Registration Module (RM)

6.2.1 Responsibilities

The RM uses the SIFT machine learning algorithm to perform image registration between the user's image and a reference image created by the 3DM. The output is a transformation matrix that can be applied to the MoonTrek points of interest which will place them in the correct locations on the moon in the input image.

6.2.2 Constraints

The RM needs to be very accurate so that it can correctly place the points of interest, and it also needs to be fast so that this can be rendered in real-time from a live telescope feed. It also needs adequate storage space to hold input images and processed copies.

6.2.3 Composition

The bulk of the RM processing takes place in process.py on the server side. Reference images used for registration are stored in the adjacent images folder.

6.2.4 Uses/Interactions

The user does not directly interact with the RM but instead all of the processing occurs on the server side.

6.2.5 Interface/Exports

The RM exports a transformation matrix to the UIM which is used in displaying the points of interest on the telescope image. Processed versions of the input images are also exported to the images folder, which are the results of the circle detection algorithm as a precursor to the image registration.

6.3 3D-Model Module (3DM)

6.3.1 Responsibilities

The 3DM is responsible for creating the context-aware reference image that the RM will use for image registration. The 3DM creates a dynamic photorealistic model of the earth, moon, and sun, and captures an image of the moon such that it appears very similar to the location and orientation of the input image. This will allow the image registration algorithm to be more accurate and dynamic since the images will be closer together to start with.

6.3.2 Constraints

The 3DM must have the adequate processing power to generate the 3D image in a timely manner and it must do so accurately.

6.3.3 Uses/Interactions

The current version of 3DM requires a longitude, latitude, time, and date as input in order to orient the planetary bodies and the camera in exactly the way that they were when the picture was taken. This data is stripped from the input image's metadata, and if the metadata is missing or invalid, the user is prompted with a form through which they can input the missing information directly.

6.3.4 Resources

The 3DM uses the browser as well as the GPU to run WebGL, the framework used for generating the 3D image.

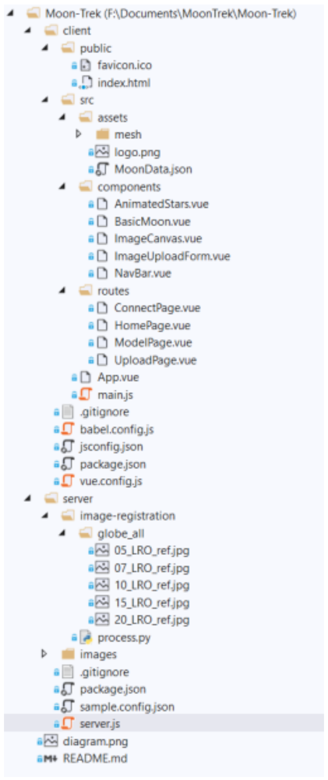
6.3.5 Interface/Exports

The 3DM exports the rendered image, and it currently does not allow for any user input beyond the required spatial and temporal information. In the future, a feature may be added which allows the user to interact with the 3D model for educational purposes.

7. Detailed Lower-level Component Design

7.1 Classification

The directory which contains all the source files for our web app is titled MoonTrek, and it contains two subdirectories: “client” and “server”. The client directory contains all the source files which make the web app functional for the user. This includes basic files for UI, such as the favicon, animated stars for the background, and the logo. The client directory also includes route files that manage the form upload process and the general structure of the backend. The server directory contains all the information necessary for image registration to take place. As of right now, this includes the reference images of the moon, which will be obsolete once our new method of image registration is implemented. Most importantly, it includes the process.py file which executes the SIFT algorithm which conducts image registration. The following image shows all the subdirectories and files in the MoonTrek directory:



```
Moon-Trek (F:\Documents\MoonTrek\Moon-Trek)
  client
    public
      favicon.ico
      index.html
    src
      assets
        mesh
        logo.png
        MoonData.json
      components
        AnimatedStars.vue
        BasicMoon.vue
        ImageCanvas.vue
        ImageUploadForm.vue
        NavBar.vue
      routes
        ConnectPage.vue
        HomePage.vue
        ModelPage.vue
        UploadPage.vue
      App.vue
      main.js
    .gitignore
    babel.config.js
    jsconfig.json
    package.json
    vue.config.js
  server
    image-registration
      globe_all
        05_LRO_ref.jpg
        07_LRO_ref.jpg
        10_LRO_ref.jpg
        15_LRO_ref.jpg
        20_LRO_ref.jpg
      process.py
    images
    .gitignore
    package.json
    sample.config.json
    server.js
  diagram.png
  README.md
```

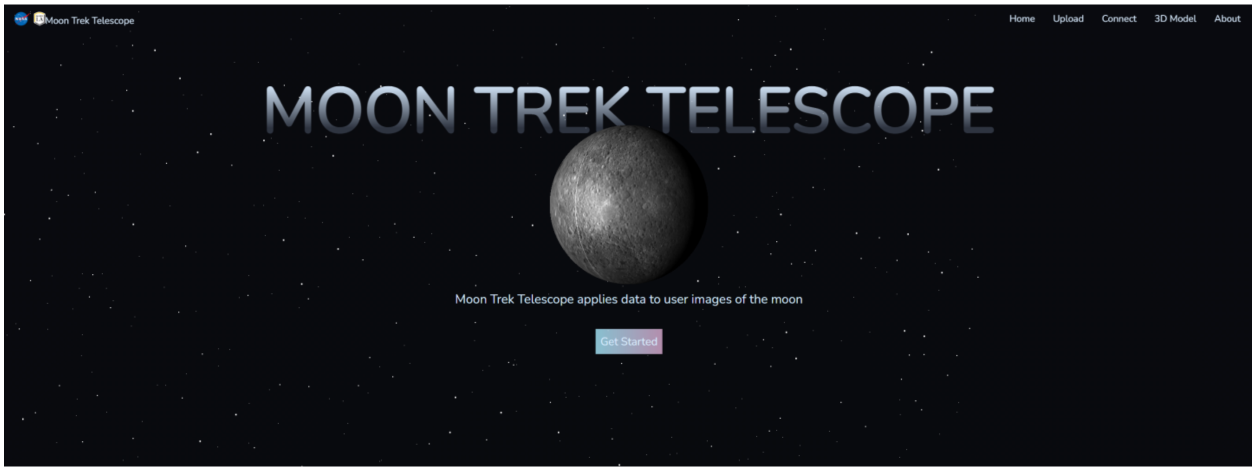
7.2 Processing Narrative (PSPEC)

The only instance of the user sharing information with the website is in the upload form. When the user is redirected to the upload page, the UploadPage.vue file displays a form that allows the user to share an image of the Moon. It checks whether or not the image contains metadata on the time, date, and location of the image taken, and if not, a form is displayed for the user to fill out that information manually.

We hoped to optimize this process by capturing the metadata from the image itself when possible. The fact that not all user images contain this metadata is a constraint that we had to deal with by displaying a form.

7.3 Interface Description

When the user loads the website, they are seeing the contents of the index.html file in the client/public directory. The index.html file doesn't contain any content besides an error message, but the HomePage.vue file displays the content we see in the following image



When the user chooses to upload a picture of the Moon they look at, they are redirected to the upload page whose content is displayed by UploadPage.vue. Once this page

7.4 Processing Detail

In this section, we'll go into some detail on the next steps and the limitations of our code.

7.4.1 Design Class Hierarchy

Our code does not utilize object-oriented programming, so this section does not apply.

7.4.2 Restrictions/Limitations

A major potential restriction is the processing speed of the application. We plan on implementing a live feed from the connected telescope, and in this situation, every part of the application must run in real-time. Because of this, every algorithm and procedure must be optimized.

Storage of data, particularly, keeping uploaded images of the moon and converting them to processed images throughout the circle detection process, is an issue that we will have to be aware of over the course of this project. Every time image registration occurs, we are doubling the amount of memory required to store images of the moon, and more, as we have to display lines between the two images for the sake of the SIFT algorithm.

8. Database Design

8.1 Overview

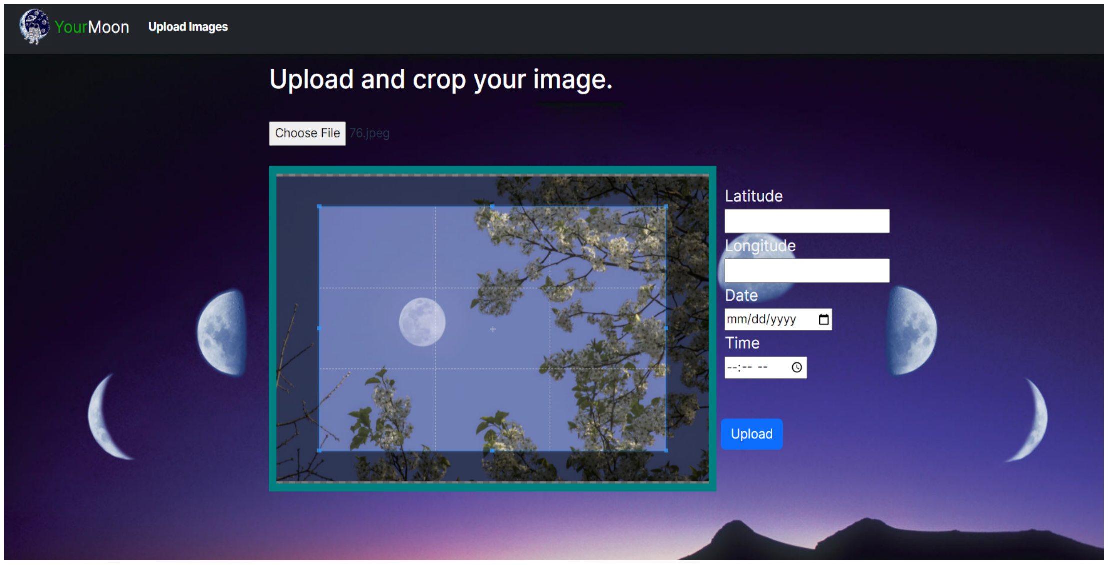
YourMoon is a supplement website that allows users to upload and store data into a MySQL database. The website is built on the Vue.js framework for the client side and Express.js for the server side. The purpose of this technical portion of the document is to describe the architecture and functionality of the YourMoon website, including the setup and configuration of the MySQL database and the implementation of JavaScript for handling upload features.

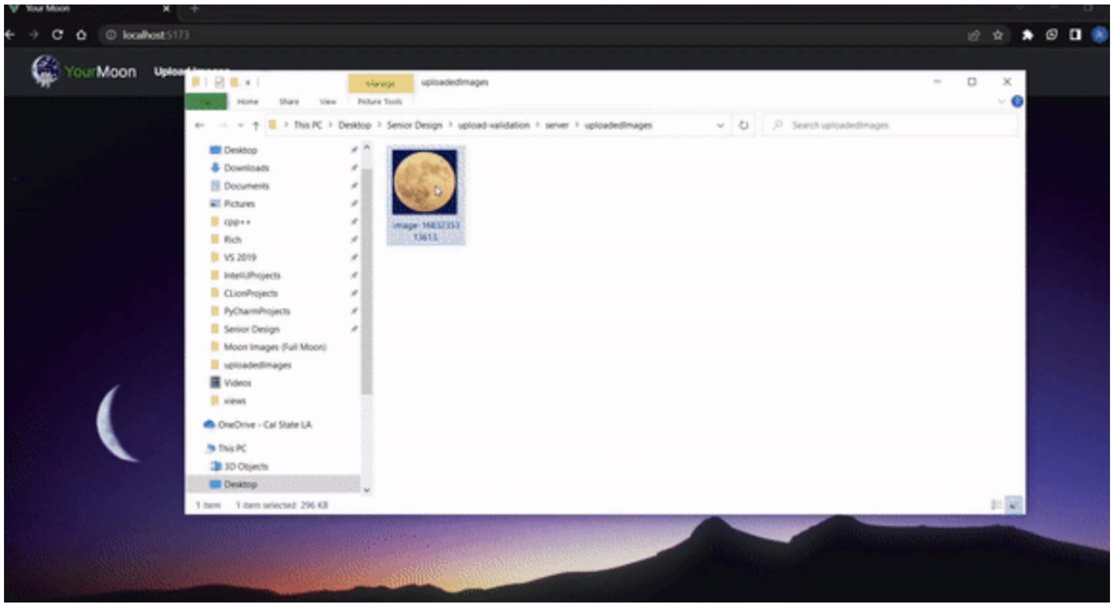
8.2 Database Setup and Implementation

To use YourMoon, our team set up their own MySQL server using MySQL Workbench and XAMPP control panel for the server. Once the server is set up, team members can create a database and tables to store data. The database is then connected to the website using Express.js. The connection is established through a configuration file that contains the database credentials. The database can then be accessed by the client-side Vue.js framework to upload and retrieve data.

YourMoon is a supplement website that allows users to upload their moon images to a MySQL database using Vue.js Framework for the client side and Express.js for the server side, while using JavaScript to handle upload features to the database. In addition, users can crop their images before uploading using a tool implemented with JavaScript, CSS, and HTML. This feature enhances the accuracy of our circle detection algorithm, allowing for better identification the moon in the images. To set up the MySQL server, users can use MySQL Workbench and XAMPP control panel for the server. With this implementation, YourMoon provides a user-friendly interface for image upload and processing, improving the accessibility of moon image data for research and educational purposes.

8.3 User Interface



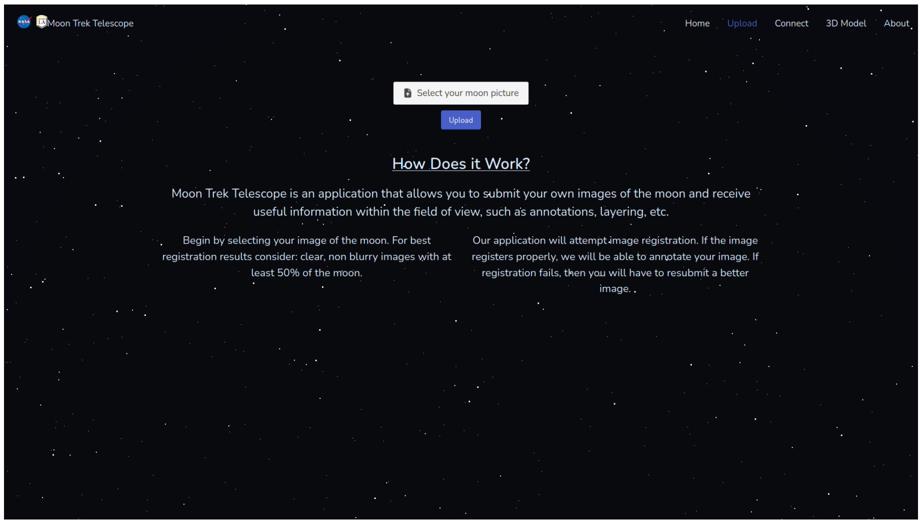

YourMoon
localhost:5173
uploadedimages
This PC > Desktop > Senior Design > upload-validation > server > uploadedimages
Desktop
Downloads
Documents
Pictures
Senior Design
uploadedimages
1 item 1 item selected 296 KB

9. User Interface

9.1 Overview of User Interface

The user will open the application and see the homepage. From the homepage, the user will be able to upload their own image of the moon. The user is able to see their moon next to a picture of the Moon with all the data. We will also implement a 3d model that shows the Moon, Earth, and Sun. The user will also be able to click the ‘About’ pages, which show information about the project and group members.

9.2 Screen Frameworks or Images

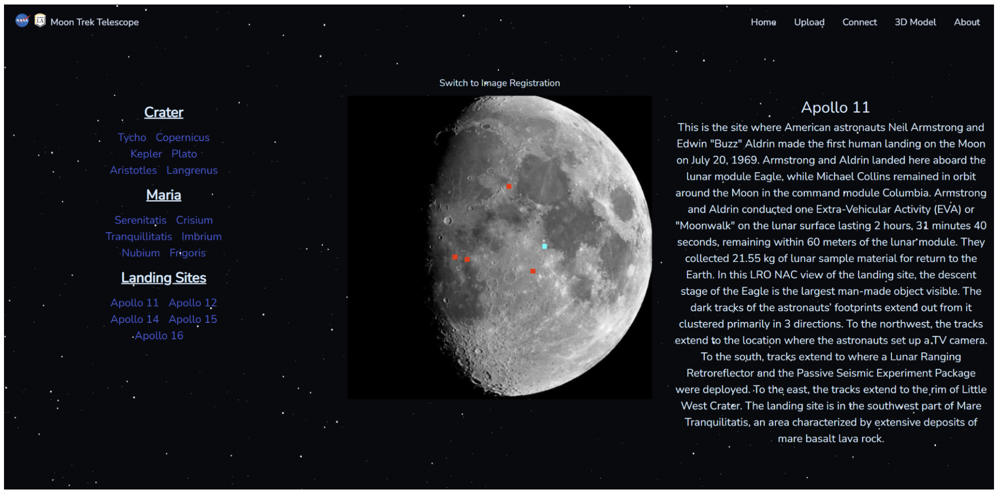


How Does it Work?

Moon Trek Telescope is an application that allows you to submit your own images of the moon and receive useful information within the field of view, such as annotations, layering, etc.

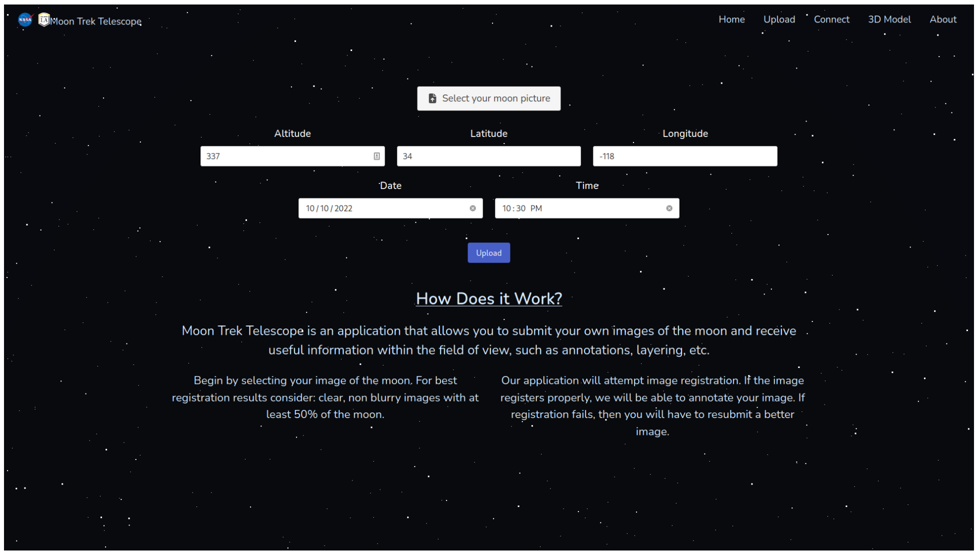
Begin by selecting your image of the moon. For best registration results consider: clear, non blurry images with at least 50% of the moon.

Our application will attempt image registration. If the image registers properly, we will be able to annotate your image. If registration fails, then you will have to resubmit a better image.



Apollo 11

This is the site where American astronauts Neil Armstrong and Edwin "Buzz" Aldrin made the first human landing on the Moon on July 20, 1969. Armstrong and Aldrin landed here aboard the lunar module Eagle, while Michael Collins remained in orbit around the Moon in the command module Columbia. Armstrong and Aldrin conducted one Extra-Vehicular Activity (EVA) or "Moonwalk" on the lunar surface lasting 2 hours, 31 minutes 40 seconds, remaining within 60 meters of the lunar module. They collected 21.55 kg of lunar sample material for return to the Earth. In this LRO NAC view of the landing site, the descent stage of the Eagle is the largest man-made object visible. The dark tracks of the astronauts' footprints extend out from it clustered primarily in 3 directions. To the northwest, the tracks extend to the location where the astronauts set up a TV camera. To the south, tracks extend to where a Lunar Ranging Retroreflector and the Passive Seismic Experiment Package were deployed. To the east, the tracks extend to the rim of Little West Crater. The landing site is in the southwest part of Mare Tranquilitatis, an area characterized by extensive deposits of mare basalt lava rock.



Altitude: 337
Latitude: 34
Longitude: -118
Date: 10 / 10 / 2022
Time: 10 : 30 PM

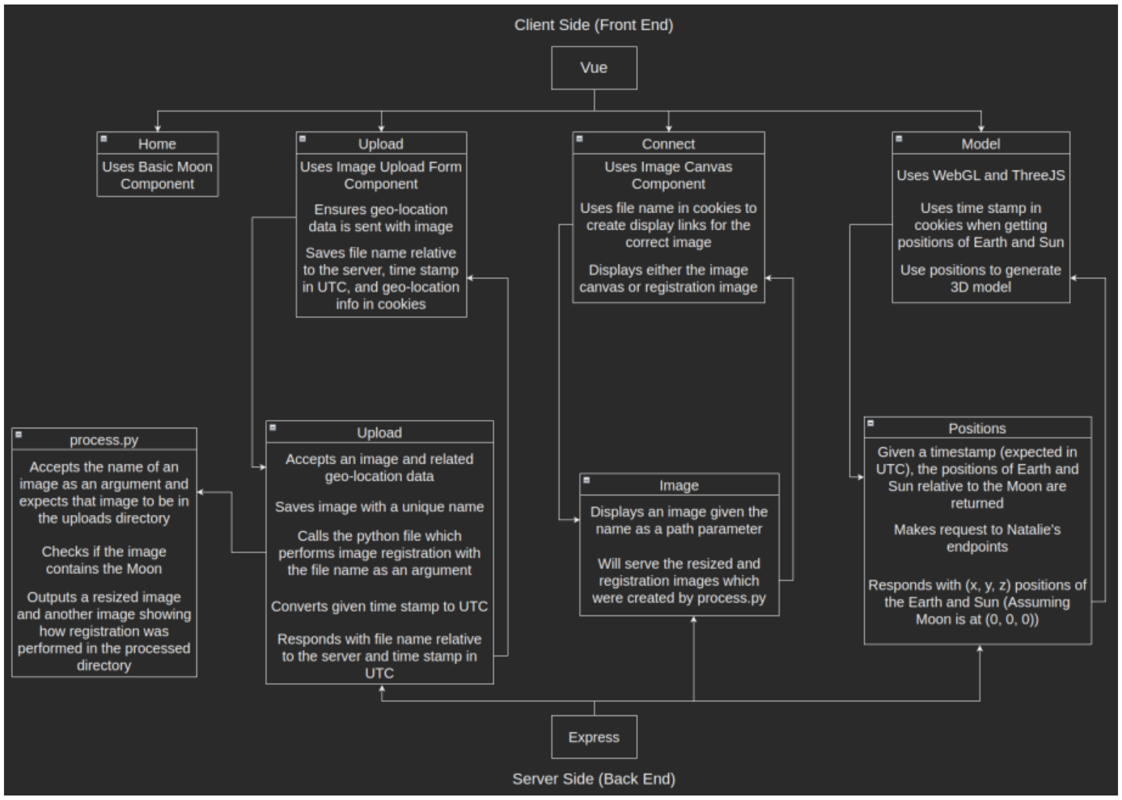
How Does it Work?

Moon Trek Telescope is an application that allows you to submit your own images of the moon and receive useful information within the field of view, such as annotations, layering, etc.

Begin by selecting your image of the moon. For best registration results consider: clear, non blurry images with at least 50% of the moon.

Our application will attempt image registration. If the image registers properly, we will be able to annotate your image. If registration fails, then you will have to resubmit a better image.

9.3 User Interface Flow Mode


Client Side (Front End)
Vue
Home
Uses Basic Moon Component
Upload
Uses Image Upload Form Component
Ensures geo-location data is sent with image
Saves file name relative to the server, time stamp in UTC, and geo-location info in cookies
Connect
Uses Image Canvas Component
Uses file name in cookies to create display links for the correct image
Displays either the image canvas or registration image
Model
Uses WebGL and ThreeJS
Uses time stamp in cookies when getting positions of Earth and Sun
Use positions to generate 3D model
process.py
Accepts the name of an image as an argument and expects that image to be in the uploads directory
Checks if the image contains the Moon
Outputs a resized image and another image showing how registration was performed in the processed directory
Upload
Accepts an image and related geo-location data
Saves image with a unique name
Calls the python file which performs image registration with the file name as an argument
Converts given time stamp to UTC
Responds with file name relative to the server and time stamp in UTC
Image
Displays an image given the name as a path parameter
Will serve the resized and registration images which were created by process.py
Positions
Given a timestamp (expected in UTC), the positions of Earth and Sun relative to the Moon are returned
Makes request to Natalie's endpoints
Responds with (x, y, z) positions of the Earth and Sun (Assuming Moon is at (0, 0, 0))
Express
Server Side (Back End)

10. Requirements Validation and Verification

Requirements	Modules/UI/Components and Testing Methods
The application shall upload a picture chosen by the user	● User Interface ● Running Application
The application shall display different data from the MoonTrek API	● User Interface ● Running Application
The application shall allow searching for data sets of their choice	● User Interface ● Database Search
The application shall be able to overlay layers and change opacity for visualization.	● User Interface ● Run the application to try different layers and opacity
The application shall limit user input to available data layers.	● User Interface ● Run the application to see layers available to the user
The application shall connect to the user's telescope	● User Interface ● Running Application
The application shall display a telescope view from the telescope or device controlling the telescope	● User Module ● Running Application

11. Glossary

Acronym	Long Version
SRD	Software Requirement Document
UI	User Interface
SDD	Software Design Document
MLA	Modern Language Association
JPL	Jet Propulsion Laboratory
MT	Moon Trek
UIM	User Interface Module
RM	Registration Module

12. References

Reference Name	Source
Software Design Document Software Requirement Document	CSULA
MoonTrek API	https://trek.nasa.gov/tiles/apidoc/trekAPI.html?bod
Django Web Framework	https://docs.djangoproject.com/en/3.1/
ASCOM Standards	https://ascom-standards.org/