Team Ceres

Requirements Specification

Sponsored By: David Trilling, Michael Gowanlock
Team Mentor: Fabio Santos
Team Members: Javier Quintana, Joseph Sirna, Miles Barrios, Zach Messenger

Accepted as baseline requirements for the project:

Client Signature:_____ Date:_____

Client Signature:_____ Date:_____

Team Lead Signature:_____ Date:_____

Version 1.0

November 19, 2020

Table of Contents

1. Introduction

The purpose of this document is to outline the expectations for our project and review the expected requirements in the final product that Team Ceres designs. The document will be broken down into multiple sections outlining the process of how these requirements were acquired and other potential challenges that the team may face in development. This document is subject to change and is intended to be an agreement between the clients and team members for the expected capstone project outcome.

Background

Every night astronomers across the globe participate in all-sky surveys, where the night sky is recorded in hopes to gain knowledge of the galactic entities that surround the Earth. These surveys can produce very large quantities of data and are useful for cataloging notable bodies, such as asteroids. The Zwicky Transient Facility (ZTF) in San Diego, California generates nearly 2 terabytes of data every night and that data alone is very difficult to examine due to the high rate that data is collected.

It is estimated that when the Vera C. Rubin Observatory is finished being built in 2021, 20 terabytes of data will be collected every night for the next 10 years. By the end of the Rubin Observatory's participation in these all-sky surveys, 73 petabytes of data will have been collected.

Our clients, Professor David Trilling and Professor Michael Gowanlock, are interested in using this data for their personal research but many of the interfaces that use data from the ZTF, are either outdated or not user friendly. The main problem that our clients face however, is that they have no interface available to them with functionality that is user friendly and easy to navigate. Because of this, Professor Trilling and Professor Gowanlock are interested in the development of a new interface that uses data from the ZTF and provides easy to access data that doesn't overwhelm its users on first view. Team Ceres goal is to create this graphical user interface and to create it with all of the requested accommodations to make it a valuable tool for our sponsors.

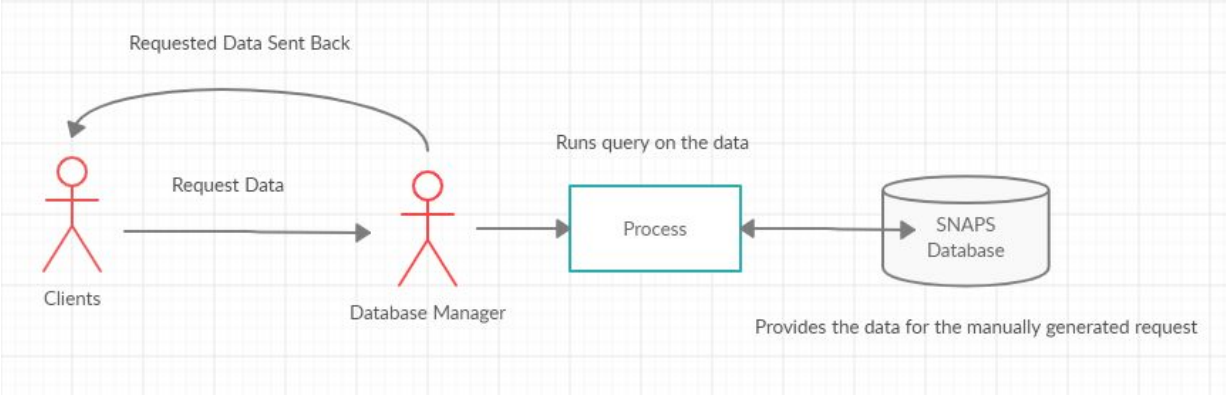
2. Problem Statement

2.1 Current Solution

Right now, our clients will choose specific sections of data to import from the Zwicky Transient Facility (ZTF) observation process, mostly focusing on small bodies and asteroids. In order to analyze this data, our clients must request that a series of scripts and processes are run in order to:

1. extract the relevant data from ZTF
2. store the data in separate database files
3. run analysis on hundreds of thousands of small bodies
4. export the results to another database file
5. chart/graph the requested information using these results

These processes are run by another colleague, who then makes the relevant charts and graphs available on NAU's RCDATA portal.



2.2 Drawbacks

The current solution in place does not provide the client with an easy way to query data and get results. This is due to the lack of user interface. The current solution is also extremely time consuming as the client needs to reach out to another colleague in charge of performing the queries and then compiling the data analysis into a usable

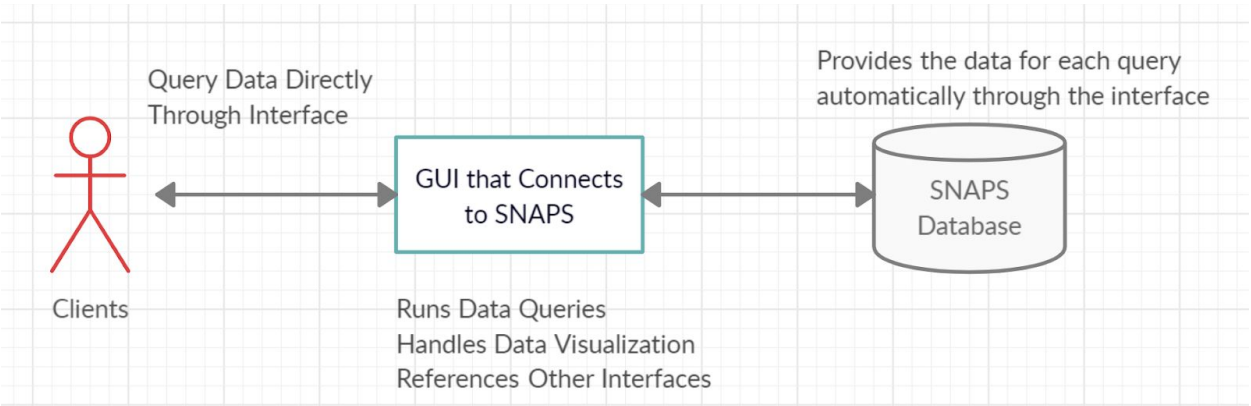
form suitable for their needs. Another issue faced with the current solution is the responsiveness of the database. Whenever a script is run to add new charts and data, the current site directories need to be manually cleared and repopulated. This causes the site to be down momentarily.

This process is workable for our client, but leaves a few things to be desired. Namely:

- Easy access to the charts and graphs
- An intuitive way to browse different aspects of analysis for a particular asteroid
- A compiled overview of the analysis as a whole
- On-demand generation of the above charts and graphs
- A way to share the results of this analysis
- A way to mark particular areas of study as important, unique to each individual user

3. Solution Vision

Team Ceres is proposing that a web interface, REST API, and hosted database solution be used to address the desired characteristics listed above. The web interface will be responsible for generating the graphs and being a useful frontend tool for analyzing the data. The REST API will manage interactions with the underlying database layer in order to deliver relevant data to the web interface. The database layer consists of two parts; the database itself and the method of importing the data from its current form into a hosted environment at regular intervals.



3.1 Solution Specifics

Elaborating upon what was mentioned previously, specific aspects of each part of the solution are outlined below:

3.1.1 Responsive GUI

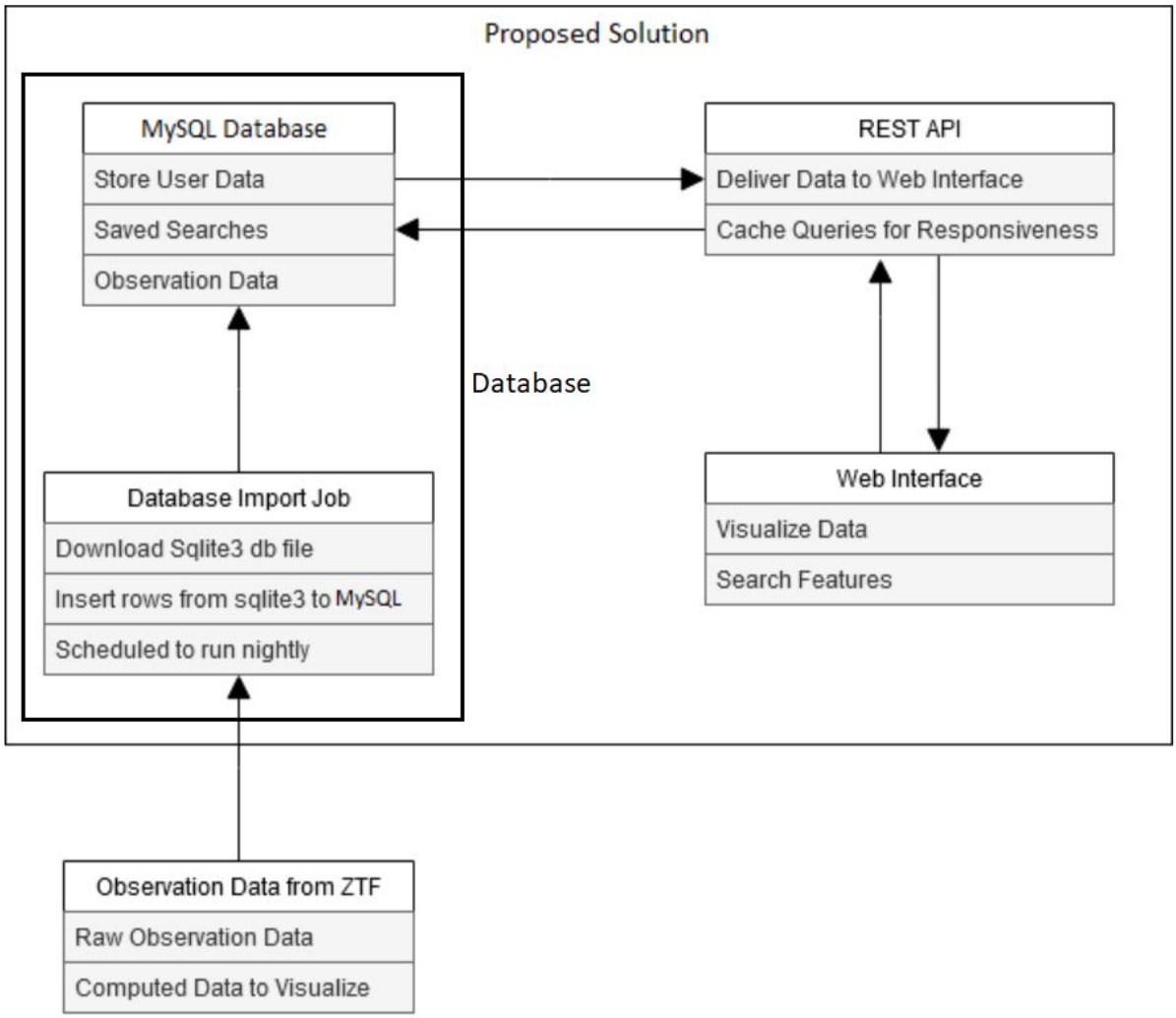
- Responsiveness when viewing graphs with large amounts of data points
- Intuitive structure and way to browse different areas of analysis
- Low latency when browsing/searching
- Utilize React to build a single page application that operates on the client side to increase performance
- Implement a hosted database and API to increase data querying and provide responsive filters
- Run import jobs to ensure that the database is filled with current information regarding asteroids

3.1.2 Interactive Graphs

- Create interactive graphs using Google's graph library using Javascript
- Use the REST API to perform quick searches for data and render the corresponding graph
- Create React components to provide the users a way to click on individual asteroids on the graphs
- Provide a way to allow users to download PNGs of the graphs for further analysis

3.1.3 Role-based permissions and User Authentication

- Utilize Google's Firebase Authentication software to handle user login, account creation, and permissions
- Use the API to store user settings based unique user id provided by Firebase
- Allow users to bookmark specific graphs and asteroids that are used frequently



4. Requirements

This section of the report details all of the specific technological requirements our software must include. These requirements have been gathered through a series of ongoing meetings with our clients and analyzing the project description provided to us at the start of the semester. This section of the report will detail the key components of the software we are planning to develop. Once all of these requirements have been met, we should have a working web application that satisfies all of the needs of our clients in order for them to further their research. Requirements fall into one of three categories:

- Functional Requirements
- Performance Requirements
- Environmental Requirements

4.1 Functional Requirements

Functional requirements represent the bulk of what our product needs to be able to do. These requirements listed will be included in the final product as the base functionalities for our web application. This section represents the core components we have agreed upon implementing throughout our meetings with our clients this first semester.

4.1.1 Filter/Display Data

Our clients emphasized the importance of not putting off anyone not well-versed in the field of astronomy when seeing the landing page. Other similar analysis visualization tools suffer from “information overload” when someone arrives at their front pages. Our solution aims to minimize this effect by concisely displaying interesting information in an easy to understand format. This feature can be broken down into smaller components that represent this functional requirement when used together, these smaller components include filtering, selecting filtered datasets to be displayed, rendering graphs that represent the filtered and selected data

Selecting filtered datasets to be displayed

After the user is returned the filtered data that they requested, the user should be able to choose specific sets of data or ranges that are going to be displayed in various formats (table, graph, etc). This select feature needs to be user friendly and allow for all types of data points. This feature will be a big help to various researchers and users that are creating a variety of displays.

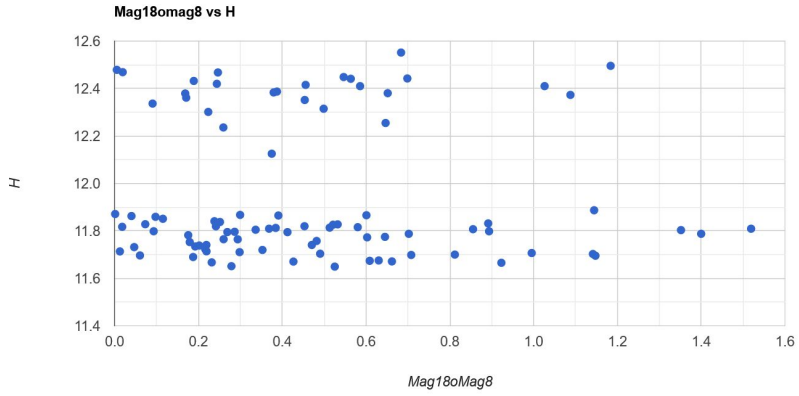
Rendering graphs that represent the filtered and selected data

After filtering and selecting data, the user will need to be able to render graphs based on the data provided. This will be done by providing Google Charts with the selected data and providing instructions on what type of graph and how it should display the

data. One way to suggest a large amount of data exists without displaying it all in an overwhelming way is to rank interesting and/or well-observed asteroids in a small table, while large visualizations about interesting characteristics expressed in layman's terms take center-stage on the front page. Clicking one of these asteroids on a graph will take the user to a similar detail page. This should create an interesting and interactive experience for the average user.

Asteroid Id	Oberservations
11004	137
105339	137
10032	135
10847	128
10505	128
1038	123
10555	123

Example: a small table, sorted by observation count.

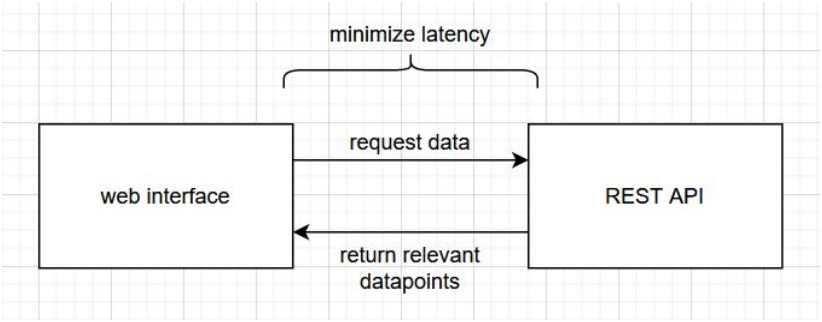


View this Asteroid on Antares
View this Asteroid on MARS

Example: clicking an asteroid Id will take the user to a page with more details.

Filtering

The ability to filter and search through large amounts of data is crucial to the core functionality of this solution. This filtering section will allow for the user to choose desired characteristics and properties of an asteroid that they require. The interactions between the web interface and the REST API must keep latency to a minimum. In addition, the REST API must be able to quickly and efficiently evaluate what data points match the constraints provided via the filtering component.



4.1.2 Export Data and Graphs

In meeting with our clients, it has been deemed necessary that users are able to easily save data and graphs to their own machines. This feature allows users to come back to data they have been analyzing previously, load datasets and figures into other softwares, or present their analysis alongside their research. On top of this, if an interesting phenomenon is observed, the astronomer/researcher will likely want to save the data/figures they found. Having the ability to download data and figures means the user does not need to recreate either the queries or figures previously generated.

Exporting

Users will have the ability to export/download files to their machines in the following formats:

- CSV
- PNG
- PDF
- FITS

Data

In our web application, the user will be able to search for asteroids using filters to view specific characteristics as described in the above requirement. This will present the user with a subset of the larger database, and the user will have the ability to download this data as described by the above requirement. The user should be able to download it in the above formats.

Graphs

Similar to data, users will have the ability to download figures such as graphs created before and after applying specific analysis tools. Any of the graphs that are able to be created as described in the previous section will be able to be downloaded to the users local machine in the above mentioned formats.

4.1.3 Create User Roles and Preferences

Our client has shared, both verbally and written that the ability to create user roles and set preferences is a key requirement for the web application. Having multiple user roles could allow for teams of researchers to have a person that manages the visualizations and queries performed. In addition, this would allow for any researchers using this software to modify their preferences which might relate to specific filtering or custom queries on the asteroid database. This functional requirement can be broken down into the following smaller requirements:

Account Creation/Login

The implementation of user roles and preferences relies on the creation of a create account/log in feature in the application. This login feature would be similar to that of any other web page where anyone could create an account. However, roles would be assigned by someone with the status of admin. These accounts that have roles lesser than admin would have the ability to perform certain tasks such as viewing/exporting graphs, data, and other visualizations.

User Settings/Preferences

Having a user settings feature will be essential to how users are able to interact with the web application. The people using our application are researchers attempting to pull different types of information regarding the asteroids, thus having a way for them to adjust their experience using the web application should be crucial. These settings could include preferences such as the maximum data points per graph, default export format (CSV, PNG, xslx), bookmarked asteroids/observations, and the ability to limit the number of datapoints returned from a query to name a few.

This settings feature could also provide administrators and other higher permission roles to create settings for users lower roles. This allows for admins to keep people from deleting essential information stored on the web application (data tables and visualizations).

Saved Work Section

As research can take longer than a single sitting to complete, the ability for a user to save their work becomes crucial to creating an application that meets the needs of the client. This saved work section could include various types of visualizations, data tables, and common queries. This section would use the share data analytics feature mentioned below because allowing users to save their work would make sharing results easier and faster.

The smaller requirements discussed above all prove to be a key part of having user roles and permissions. These roles and permissions provide ways to improve the user experience (personal settings) and protect information that admins and other high roles might want preserved (role-specific settings).

4.1.4 Share Data Analytics with Other Users

Another valuable feature that was requested from our clients was the ability to share data analytics with other users. The reason this is a functional requirement is because the intention our clients have is to share this data with other users across the country.

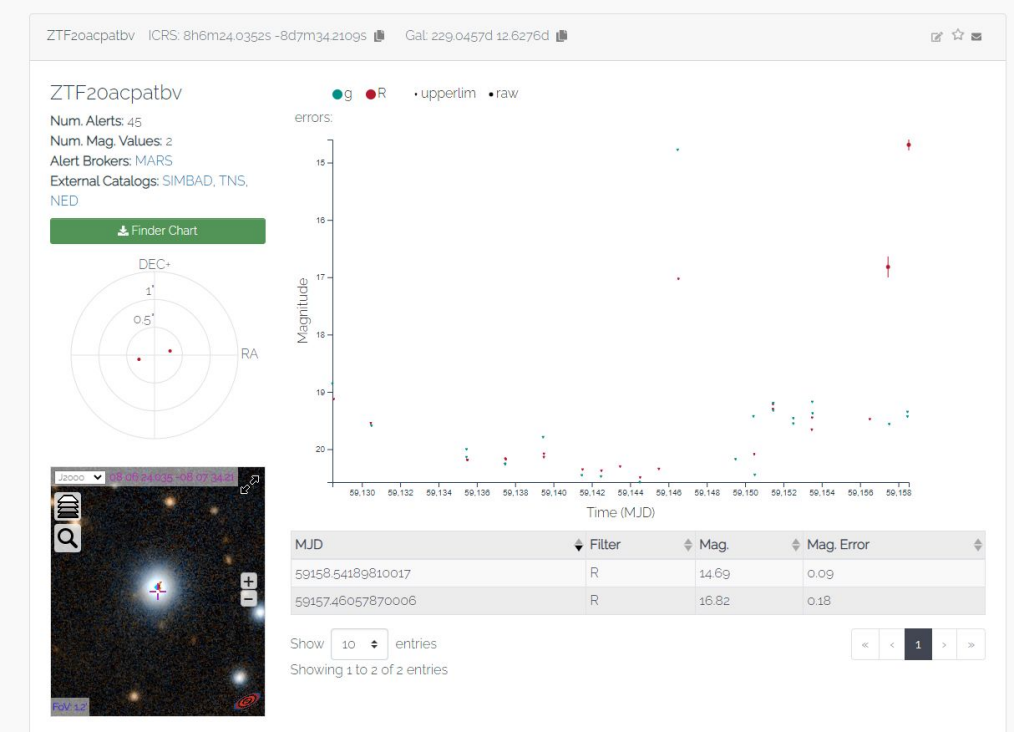
Accompanied with sharing, we want to also provide links to other existing databases for asteroids in order to provide more resources readily available.

Sharing

The graphical user interface we are creating will have the option to share each asteroid's metrics with other users via email or web URL in order to ensure that the end user has access to the live data in case of any changes or developments. This will allow easy collaboration amongst peers and researchers.

Linking

Along with the ability to share specific asteroid data with other users, our clients have requested that we also link to two other asteroid databases, the Antares interface and the Mars interface, both of which pull information from the ZTF data stream. The reason this is important is because these databases have access to resources that our team does not, such as maps displaying the asteroids position and images from previous all-sky surveys. Below is an example of some of the resources available on these other interfaces.



The implementation of making these resources available in our application will be done by using simple linking to the same asteroid in the other database. We were able to study the other databases to create a hyperlink that directly queries the needed asteroid.

4.1.5 Database Import Job

Another functional requirement that the team found as a major one was the database import job. In order to ensure that the application is using up to date data, the team will run database import jobs to bring in any new information introduced to the database. These import jobs need to be performed at ideal times when web application use is little to none. This will ensure that no users are interrupted in the middle of work. It is important that our application is able to receive import jobs for new data, or else the web application would require manual updates which would defeat the purpose of our application.

4.2 Performance Requirements

The performance requirements of our project represent how our product needs to perform as a whole. In our case we want to have a slight time constraint so ensure our software maintains responsiveness, and this ties in with the software being reliable as well. In order to verify both of the previous requirements, we will be using time measurements for how long our database takes to respond to both short and long queries.

On top of these, we want our software to protect our user's personal information and preferences, and be easily usable as a whole. In order to verify this requirement, we will be able to check if any of our user's personal information or keys are available to an unverified user.

4.2.1 Responsive

Due to the nature of our web application, responsiveness is essential to creating a successful piece of software. This application deals with a massive amount of data, which in turn could take a longer than necessary time to render when filtering for specific results. Users should be able to choose specific filters from a large number of options and instantly be returned the information they were looking for. Our team hopes

In addition to quick loading of data visualizations, our team feels it is necessary to optimize our login system as well as decrease the load time for any page in the web application. This is where single page applications shine due to the quick render times that occur on the client side. We will be able to measure and verify our responsiveness by checking the latency of our application. This will provide us with the ability to time some features of our application to make sure it is responsive.

4.2.2 Reliable

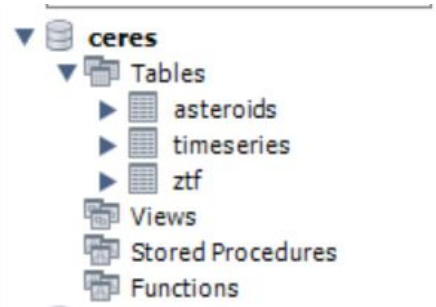
The ability to reliably visualize data is contingent on a variety of factors, including server load, database load, and specific database processes that might impact the speed at

which data may be delivered to the web interface. One such database process is the import of data both from ZTF and of the additional calculated data.

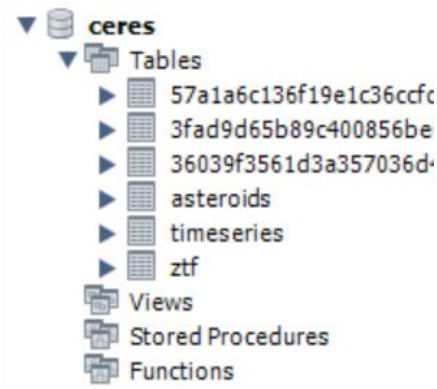
Generally speaking, the database layer should be relatively fast, in all circumstances except times where large amounts of write queries are being executed. An extreme example of this circumstance is during the database import process, where hundreds of thousands of rows are written to three tables in the database and are processed at the speed the database can handle them. In order to mitigate the impact this will invariably have on performance, temporary tables will be created and written to for the duration of the import process. When this is complete, the original tables will be dropped, and the temporary tables will be renamed to reflect the enforced schema. This approach benefits in the following ways:

- minimizes the probability of a lock on a table in use by the REST API
- reduces the impact writing and reading to and from a single table concurrently has on performance
- allows the (outdated) tables to be used for the duration of the import process, which may take up to an hour

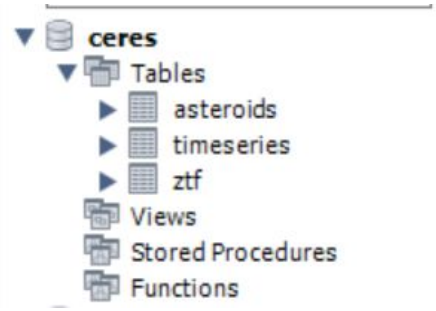
This process is visualized below:



Step 1: The database before an import process.
These tables are being read from normally, and the database load is low.



Step 2: The database during an import process.
The first 3 listed tables are being written to, while the last 3 are being read from using the API.
These tables will remain named as random MD5 hashes until the import job is complete.



Step 3: The database after an import process.
The first 3 original tables have been dropped and the temporary tables have been changed to reflect their correct names.

Since the process of dropping and renaming tables is orders of magnitude faster than the duration of the import job, the amount of time that the database is able to provide complete information about all of the observations is maximized.

4.2.3 Secure

Users being able to create accounts and share analytics with other users is an important feature of our application. Because of this, the security of our users' sensitive information is important. User’s account details, personal information, and personal analysis should be secure and hidden from other users. The amount of data being

protected is rather small, but this is still an important feature nonetheless as there is existing data needing protection.

Users will be able to sign into their profile, link their email, and save data without fear of having any of their information being stolen. This is a research friendly website as well, so we want to make sure that users feel comfortable with performing their research in our application. If data was not protected, researchers might be dissuaded by the fact their work could easily be stolen by others. These reasons are proficient for the existence of authentication and security measures.

To implement this, we are using an authentication api that is able to independently verify the login of our users and manage their login sessions. This api will also be used to supply secure keys that we will use to store user information in our database. Through the combination of our authentication api and securing the user information in our database, we will be able to easily protect any sensitive information. Additionally, we believe that our authentication api will allow us to easily implement role based permissions for other users. We would be able to link user keys with admin, base user, or any other desired role type.

4.3 Environmental Requirements

The environmental requirements of our project represent the constraints that our clients have given us for the implementation of our final product. The main concern we had in this category of requirements was how we were going to host the web application once it was finished. Through meeting with our clients, we found that there was a desire for the application to be hosted locally on an NAU owned machine. This means that the application will not be hosted by any paid service provider, but instead on either a machine owned by one of our clients or the university itself.

4.3.1 Compatibility with NAU Devices

The key component of our web application is that it needs to be able to run on an NAU owned machine. In meeting with our clients, we brought up hosting our web application

at some point in the year for testing. Our clients were adamant that a server at NAU would be more than capable of handling the volume of traffic our product is expecting, and therefore we would not need a hosting service. Therefore, we are building our product with the expectation that it can be run reliably with one server from NAU and not need any additional hosting services that would add to the cost of testing.

4.3.2 Import New Data at Night

One other environmental constraint that became apparent as we started testing imports to our database, was the need to run these jobs at times when the level of user interaction was at a minimum. This means that for us, the most ideal time to run these imports would be overnight as our clients would likely be at home and no longer working. This might seem like a small requirement to include, but it plays a vital role in maintaining the responsiveness and availability of our application.

5. Potential Risks

In this segment of the document we will be discussing the risks associated with our product implementation. Risks generally encompass anything that could go wrong with specific portions of our implementation or technologies, and the impacts of these faults. The topics mentioned in this section pertain to how they will affect our clients and overall functionality of our final product. Each potential risk will additionally have its own rating of severity for the negative outcomes it could cause.

There are two main risks that the team will be assessing when creating this project:

Risk	**Severity**	**Likelihood**	**Mitigation**
Implementation Change in ZTF Data Gather	Medium	Unlikely	Reformat data import job to gather data from the changed ZTF (reformat dependent on implementation change).
Large Data Imports	High	Very Likely	Have certain down time periods to

Blocking Access to Database			allow for the interface to spool any data changes into the GUI.

5.1 Implementation Change in ZTF Data Gathering

In the case that the Zwicky Transient Facility changes how the data is gathered in their database, it would directly affect the SNAPS database that Team Ceres' GUI will use. In order to accommodate these changes, the SNAPS database and the import job from the GUI will need to be reformatted to match the changes in the ZTF database. This is a *medium* severity issue as the data still in the GUI would be maintained but no new data would come in until these changes have been accounted for.

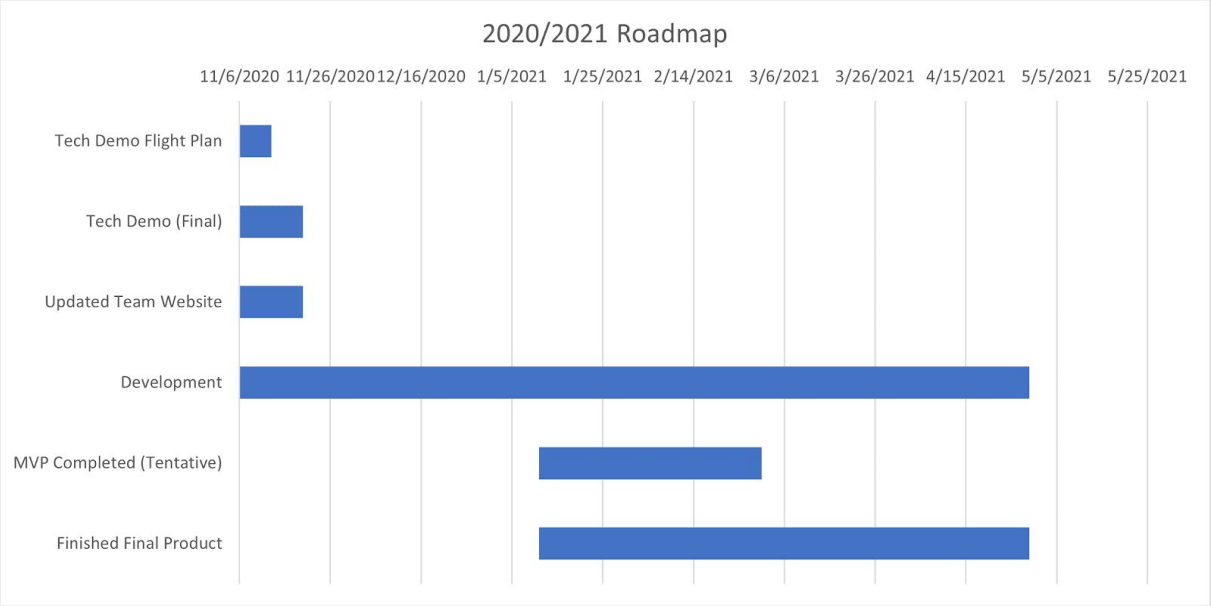
Along with this, the person who manages the SNAPS database is knowledgeable of the current ZTF schema and can quickly make adjustments to the SNAPS database as needed while the GUI's maintainers can mimic the changes within the import script.

5.2 Large Data Imports Blocking Access to Database

A very common issue that users may encounter in using the interface is the time period where the GUI is importing data from the SNAPS database and becomes temporarily unavailable. The reason this is a *high* severity issue is because the GUI will constantly need to update in order to display current data from SNAPS. These import jobs do take some time to add to the GUI but as it currently stands there is no way around this.

In order to reduce the number of users having issues with the GUI loading new data, the issue could be mitigated by having planned periods of time where the GUI loads new data from SNAPs. Ideally this time period would be chosen when the least amount of users would typically access the application such as very early morning or late at night. While new data may be unavailable during this time period, it is expected that the interface will use temporary tables holding the old data from before and while the Team Ceres GUI is updating, there are alternative interfaces available to users such as the Antares and Mars interfaces which interface with the data we are using.

6. Project Plan



As our team continues to refine our requirements and gain a better understanding of the web application we will be building, we have laid out a couple of key milestones to be hit in the coming months. Prior to the end of the year, Team Ceres would like to complete the tech demo flight plan, tech demo (final), update the team website, and continue development on the web application slowly over winter break. Following the start of the new year, the team will hopefully be moving along with general application development as well as preparing to meet the tentative deadlines for completing the Minimum Viable Product (MVP) and delivering a final product. Each milestone will be discussed in further detail below:

6.1 Tech Demo

In order to be prepared for the tech demo approaching on November 20, 2020, the team has begun to split up responsibilities for what needs to be developed as well as creating a plan for what we want the client to see in our demo. Due to our strategic choices regarding technologies, we see little to no issues with compatibility.

6.2 Updated Team Website

After completing several important documents related to the application this semester, the team will need to ensure that the team website is updated to match the completed assignments. This will allow the client to be able to refer to these documents whenever needed. This website has been modified throughout the semester and proves to be a small milestone to reach thanks to continuous development.

6.3 Development

As the tech demo approaches and the team continuously communicates with the client, development has begun and will continue until a final product is ready to be delivered. Leading into winter break, the team has discussed a plan to continue development on the application until we return to NAU in January for the Spring Semester. This will include training on different technologies, staying in contact with the client, and continuing communication amongst the team to ensure everyone is on track.

6.4 Completed MVP

During the first half of the Spring semester, the team hopes to be working on the MVP (minimum viable product), a version of the web application that meets the client's basic requirements. This MVP should be a representation of what the application is going to look like (design) as well as how it should function. The team expects for the MVP to be completed no later than March 1st, 2021 where we will show the client the state of the application to ensure that the current state of the product meets the client's expectations.

6.5 Finished Final Product

After completing the MVP, the team will have finished developing the basic requirements for this application. Our team would like to focus on the stretch goals provided in the initial document by the client. These features will be refined as needed through conversations with the client. The team hopes to have these stretch goals complete by the end of the semester (end of capstone) on April 29th, 2021.

7. Conclusion

The big data revolution is coming to astronomy, and there are no user-friendly computational methods for analyzing this data as it currently stands. The knowledge to be gained from this data could lead to many powerful discoveries, but most importantly it could help us to prepare for the inevitability of an asteroid impacting Earth. So Team Ceres plans to address this problem and build this interface to computational methods for analyzing data from bodies in space. We want to build a scalable, responsive web application in order to aid in this revolution. We plan to use a comprehensive data visualization framework, user authentication, and importing data to a web-accessible database system in order to reach our goals.

We spent a good deal of time researching the various technologies that we could use for this project, and we feel confident in the decisions that we made. Additionally, we have been able to verify our feasibility through multiple small scale demos being pieced together. Soon we will implement the above requirements on a small scale for our tech demo. This will again verify that we are able to meet the requirements and specifications provided in this document. Ultimately, we are confident that we can reach our goal and complete this product to aid our clients in the analysis of both small and large bodies in space.