Requirements Specification

11 December 2019
Version 2

Sponsors

Dr. Kiona Ogle
Dr. Michael Fell

Mentor

Isaac Shaffer

TreeViz

Riley McWilliams
Qi Han
Haitian Tang
Daniel Rustrum
Alex Bentley

Accepted as baseline requirements for the project:

For the client: ______________________ __________
Signature *Date*

For the team (team lead): ______________________ __________
Signature *Date*

Contents

1. Introduction

1.1 Background

Climate change is a natural cycle of the Earth that impacts many ecosystems. However, due to the recent and rapid growth in industrialism, climate change is now being more affected by artificial causes than natural ones. A severe increase in the amount of carbon dioxide (CO2) emitted into the atmosphere by human-made technology is dangerously speeding up the effects of climate change on the planet.

Trees play an extremely important role in Earth's climate behavior. They cover around 50% of Earth's land area and contain upwards of 90% of the global vegetation carbon. This means that a lot of CO2 that humans emit is being taken in by trees and used for their growth. Observing tree growth patterns can lead to a better understanding of how trees and climate interact with each other. These observations can also be used to predict the effects of climate change in the future.

To better understand how certain factors affect trees, Dr. Kiona Ogle and Dr. Michael Fell of Ogle Labs developed a simulation that shows a tree's growth over time. The simulation is called the Allometrically Constrained Growth and Carbon Allocation (**ACGCA**) model. It uses over 30 input parameters to run the simulation, which calculates the state of the tree over time. The output of the model contains useful information such as the tree's height, trunk radius, the carbon in the leaves and trunk, etc.

1.2 Problem

The issue is that the **ACGCA** model currently has a very small user base. This means that very few people are using the model to learn about tree growth. The specific problems that the current iteration of the model has are as follows.

- Not available online

  The **ACGCA** model is not available online. This limits the use of it to only those acquainted with Dr. Ogle or Dr. Fell.

- Unfriendly user input

  The model's input is command line based, which requires the user to have programming experience to run it.

- Requires knowledge in biology

  The user needs to know what each of the 30+ input parameters are, which requires experience in tree biology.

- Unfriendly output

  The model only outputs raw numerical data, which can be useful for biologists, but no one else. It is not visually appealing nor informative to non-researchers.

The combination of these factors creates a user environment that is unfriendly and requires knowledge in specific fields. The fact that the model is not available online also contributes heavily to the low usage of it. With so few people using the model, there is not a lot of information coming from it that can be used to better people's understanding of tree growth. A problem that is not specifically related to the model's functionality is the lack of funding for the project. Our client's have to pay for everything with their own money, so we need to develop with low-cost solutions in mind.

1.3 Solution

Dr. Ogle and Dr. Fell want to expand their audience so that anyone can utilize and learn from the **ACGCA** model. More specifically, they envision the model being used in a classroom setting where students can experiment with the inputs. This way, students can learn how certain factors affect tree growth. While students are the primary demographic of focus, researchers will also be considered during development as the original demographic of the model.

Our clients also want a way to track how the model is being used. They are interested in knowing if more students versus researchers versus everyday people are using it. They are also interested in the general location of the users. While mandatory, this information will only be used for analytical purposes and will not infringe on anyone's privacy.

Team TreeViz is tasked with making the model more user-friendly. To do this, we will create a website that will be hosted online for anyone to use. This will be a good way to get it into the hands of students and educators, or anyone else that wants to use it. The website will run the model as usual and significantly increase the accessibility to it by overhauling the input and output. Some specific features that will solve the model's problems are as follows.

- Available online

  The website will be hosted online so that anyone with internet access can use it.

- User surveys

  Users will submit a survey so that our clients can see how the model is being used. This will help our clients develop the product further, allowing the prioritization of certain user demographics for future updates.

- User-friendly input

  The 30+ input parameters will be made up of text boxes and sliders instead of a command line. This will allow anyone to use it, rather than only researchers.

- Access to the model

  The model will be accessible by the website and ran through our user-friendly interface.

- Tree visualization

  The output will be a rendered tree visualization. This will be significantly more intuitive than the raw data and will allow many more users to learn from the model. The raw data will also be available to those who wish to use it.

- Local desktop development

  To avoid costly solutions to our clients, the biggest change we can make is developing for a local server. We chose this solution to avoid paying for an online hosting service.

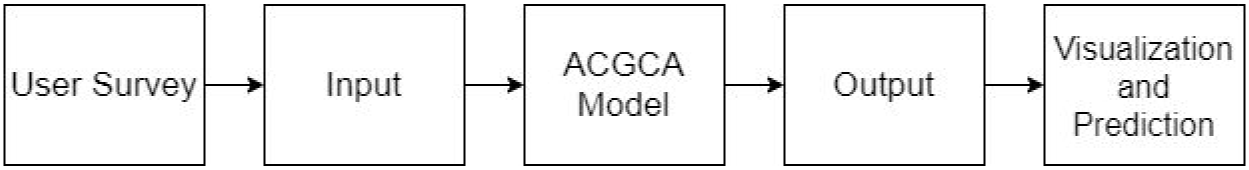

Figure 1: Overall Flow of the System

Figure 1 shows the overall flow of our system. It starts with a user survey, where new users will fill out information such as their affiliation, location, and title (student, researcher, teacher, etc.). Then they will enter the necessary inputs for the model. This will run the **ACGCA** model normally and produce an output. It will be displayed to the user as a visual representation of the tree. Then the user can observe the tree and learn about tree growth from it. Researchers can use this data to assist in climate change prediction.

While these features solve the main issues with the current iteration of the model, they are not all that our solution has to offer. The system will mainly be used for educating people about tree growth. This will be done by allowing users to experiment with the 30+ input parameters and observe the state of the tree over time. Each of the input parameters will come with a description and recommended range of values so that even non-biologist users can understand them. We will also group the input parameters in such a way that keeps the user from feeling overwhelmed.

A key decision that we made when researching solutions is to have quick feedback between the inputs and outputs of the model. We originally thought to include the inputs and outputs on their own pages. However, after consulting with a user experience professional, we decided it would be better to have everything on a single page. This way, users can modify input values, run the model, and get a visualization all in one place. This will allow users to more quickly experiment with the input and get a result. We predict that keeping everything in one place will lead to a more intuitive learning experience for users as opposed to splitting up the input and output into multiple pages.

With this solution, we hope to give our clients a way that will allow anyone to access their **ACGCA** model and learn about tree growth. The website may be used in labs for researchers to conduct experiments and study climate change. It could also be used in a classroom where students are learning together about how trees grow. Team TreeViz is confident in our ability to deliver the specified product and help our clients get their model into the hands of more people.

1.4 Purpose of this Document

The purpose of this document is to list the specific requirements necessary for a complete product. It will act as a contract, to ensure the requirements are commonly understood between the client and the developers. This document is signed by team lead, Riley McWilliams, as well as a client, Dr. Kiona Ogle. The requirements are organized into environmental, domain-level, functional, and performance requirements for ease of understanding.

2. Product Requirements

The product is compiled of individual requirements that can be organized into categories. These categories are domain-level (what the user needs), functional (*what* the product must do), and performance (*how* the product will do its functions). We also discuss the environmental requirements that our clients have set for us.

2.1 Environmental Requirements

Environmental requirements are the requirements that our client has set for us. These determine how we will be developing the product and what the broader aspects of the product are. We have 5 environmental requirements: the product must be low cost, internet accessible, developed on a local desktop, be a web application, and have cloud deployment.

- Low-Cost Solutions

  Our clients have to pay for the product themselves. Thus, we do not have much budget for development or for any costs that occur after release.

- Internet Accessible

  Our clients want their product to be accessible for school students or researchers from different places all over the world. Thus, our product needs to be available online.

- Local Desktop

  To meet with our clients' low-cost requirement, we must use a local desktop as our host environment during development. This way, we do not need to pay extra money for a host environment during development. However, considering further growth, we have researched online web services like Heroku and Firebase.

- Web Application

  Our clients want the website to be as user-friendly as possible and provide users with indirect access to the **ACGCA** model. This way, users can avoid operating with the command line and dealing with raw data output. However, our web application will still allow users to view the raw data output if they wish to.

- Cloud Deployment

  Due to our clients' need for the product to be online, we need to build a web application that requires cloud deployment.

2.2 Domain-level Requirements

Domain-level requirements are the business rules that determine the environments that we will be working in. The domain requirements were determined by abstracting the requirements that we had already gathered and determining what those requirements ultimately pertained to. Some of these functional and performance requirements overlap with other domains but have broken down the domains into 3 categories: web application, survey, and maintenance.

- Web Application

  The web application domain encapsulates all of the functional and performance requirements needed for our product to interact with the model and produce the desired results.

- Survey

  The user surveys are required by our client to gather user data for analysis. The survey domain encapsulates the administrative duties and user experience requirements.

- Maintenance

  Maintenance is required for the client to be able to maintain the product after the completion of the project. The maintenance domain encapsulates many administrative requirements, performance requirements, and functional requirements that help retain a functional product. The project will be thoroughly documented so that future developers can reference it for maintenance purposes.

2.3 Functional Requirements

Functional requirements describe what the product must do. These are split up into 3 categories: web application, survey, and maintenance. Below are two Venn diagrams that outline the functional requirements. Figure 2 shows the high-level requirements and Figure 3 shows some lower-level requirements.

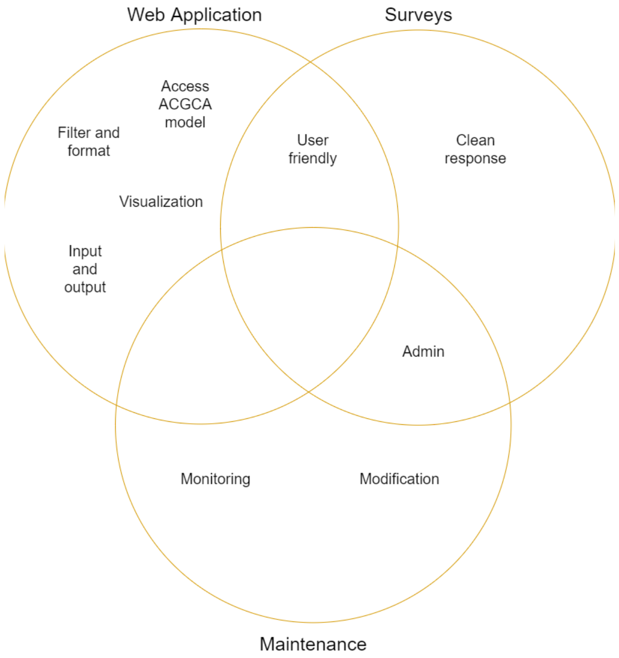

Figure 2: High-level Requirements

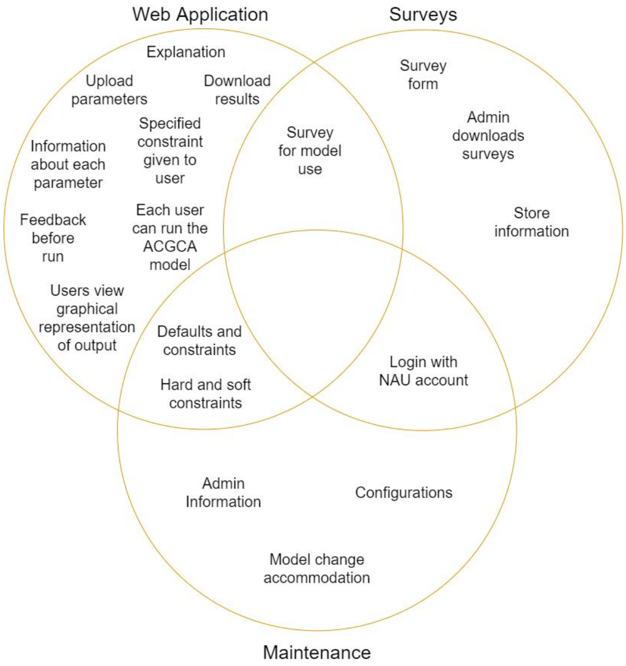

Figure 3: Low-level Requirements

2.3.1 Web Application Functional Requirements

The product must be a web application that provides different ways for users to input data to the **ACGCA** model and get the results. These requirements include the functions that will give users access to the functionality of the model. They also include functions that will give the user a better experience when interacting with the model.

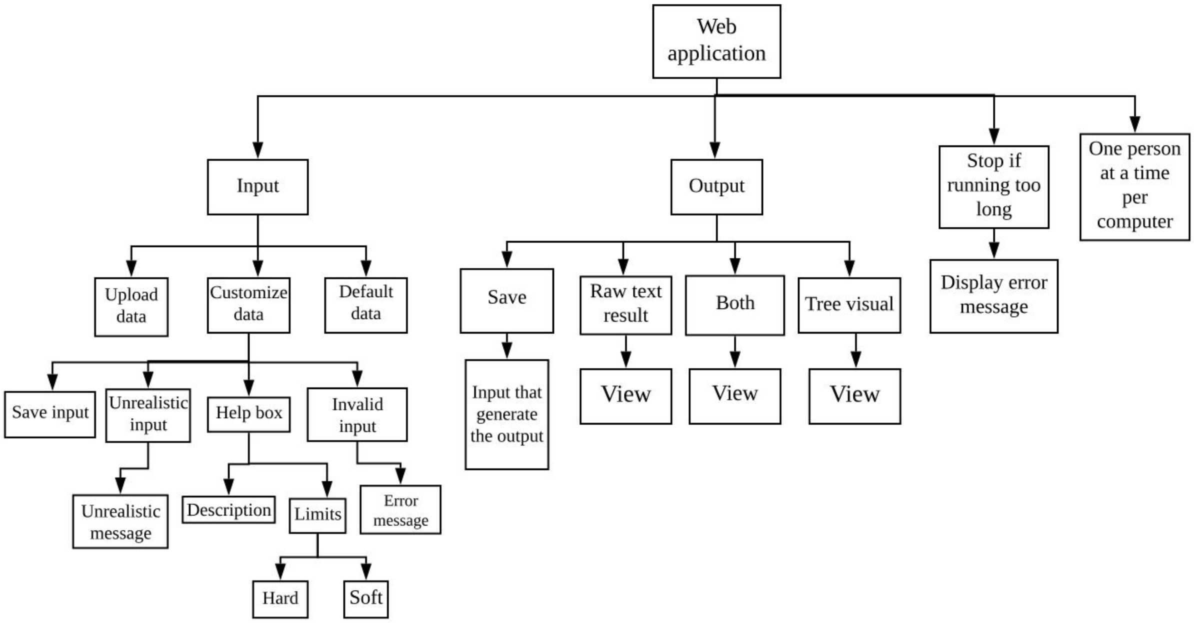

Figure 4: Web Application Functional Requirements

Figure 4 shows a tree graph of the web application functional requirement breakdown. We have split these requirements into the 2 main groups of input and output. Additionally, there are also a few requirements that do not fit into the two groups.

1. **ACGCA** Model Input

   The product will provide three ways to input data into the **ACGCA** model. These are by default scenarios, custom input, and uploading input.

1.1. Default Data

Users can use the default inputs which will be built into the program. These will take the form of scenarios created by administrators.

1.2. Customize Data

Users can change the inputs themselves and experiment with different values.

1.2.1. Help Box

We will provide a help box marker on the side of each parameter. When users do not know what the parameter is, they can click the help box marker. A help box will pop up to help the user understand it more.

1.2.1.1. Description

Help boxes will give a description of the parameter.

1.2.1.2. Limits

Help boxes will also give the limits of each parameter. Some inputs will have hard limits that the user cannot break. Most inputs will have soft limits that recommend values to the user.

1.2.2. Unrealistic Input Error

If a user enters an unrealistic value, a message will pop up that lets them know that the value may cause problems in the simulation.

1.2.3. Invalid Input Error

If a user enters an invalid value, a message will pop up that lets them know that they cannot run the model with that value.

1.2.4. Save input

Users can save their input values after entering them. This way, they can upload the input file to the application when they come back to it, rather than entering all of the values again.

1.3. Upload Data

Users will be able to upload an input parameter file as long as it contains the necessary data. This will allow users to use input sets that they previously saved.

2. **ACGCA** Model Output

The product will provide two ways of output that users can choose from it. One way is the tree visualization, and the other is the raw data results. If the model is running for too long, the program will stop and give an error message back. Only one user can use the **ACGCA** model at a time per computer.

2.1. Tree Visualization

This will display a rendered tree visualization. The tree will be generated based on the model's output variables. We will provide two ways of visualization. One way is displaying a 3D model that will be more realistic. The other way is displaying a geometrically shaped tree for more precise measurements. Tree Visualization will be a great way for the non-professional users to utilize the model.

2.2. Raw Data

This will display the model outputs in their raw data format. It will be useful for more experienced users and researchers.

2.3. Both Tree Visualization and Raw Data

Users will be able to view the output data both as a tree visualization and as the raw data itself.

2.4. Save Output

Users will be able to save the output to reference for their own use. When saving the output, users can choose to also save the input that generated said output, so they can see the connection between them.

2.3.2 Survey Functional Requirements

Survey requirements are the requirements that pertain to the mandatory surveys that users need to fill out to gain access to the **ACGCA** model and the visualization software. The surveys are necessary because our client would like to be able to gather data on who uses the model and why. They would also like to know how often people use the model and their affiliation, i.e., students, teachers, independent users, etc.

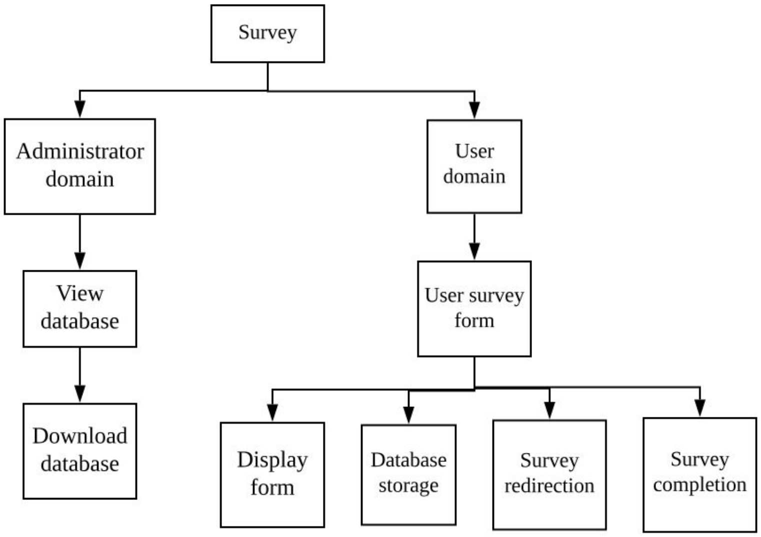

Figure 5 : Survey Functional Requirements

Figure 5 shows a tree graph of the breakdown of the survey requirements. We have split the survey requirements into 2 domains, the user side and the administrative side, each with their own sub-requirements.

Administrator Domain

1. Storage

The database will need to be stored on a server and the administrators will need access to it. There will also need to be another database that stores the administrators and their database authority.

1.1. View Database

The main interaction between the administrator and the user data is the ability to view the stored data.

1.1.1. Download Database

To simplify the method of viewing the database, the administrator will download the database as a .csv file. This will allow the administrator to easily read the data and be able to use the data with other programs.

User Domain

1. User Survey Form

The survey is how the user will input their information to populate the database.

1.1. Display Form

We will need to be able to display the form to the user and provide input boxes for them to enter their data.

1.2. Database Storage

The data that is collected will need to be transferred to the database for the administrators to be able to analyze.

1.3. Survey Redirection

If a user tries to access the model directly by entering the URL that corresponds to the visualization, they will be redirected to the survey page.

1.4 Survey Completion

The user needs to fill out every part of the user survey to access the model and if they try to leave a field blank, the page will display an error indicating to the user that all fields must be completed before continuing.

2.3.3 Maintenance Functional Requirements

Maintenance requirements are the functions the product needs in order to be easily maintainable in the future. Implementing these requirements will allow future developers to continue the project with minimal resistance.

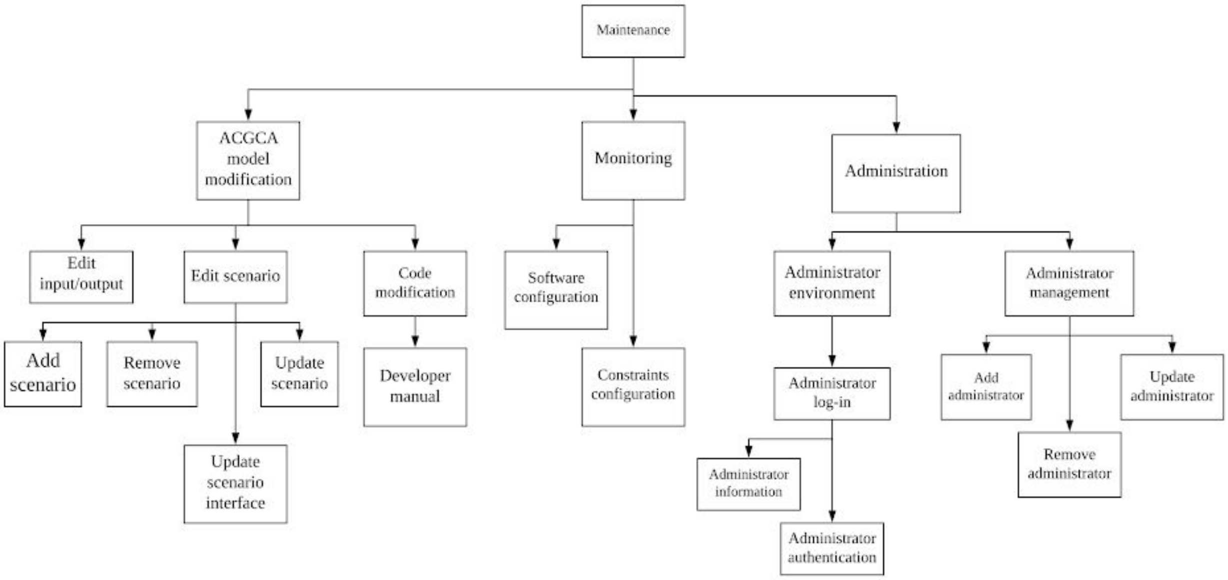

Figure 6 : Maintenance Functional Requirements

Figure 6 shows a tree graph of the breakdown of the maintenance requirements. We have outlined 3 requirements related to maintaining the system as well as their sub-requirements. The 3 high-level requirements are **ACGCA** model modification, monitoring, and administration.

1. **ACGCA** Model Modification

   The **ACGCA** model itself may change in the future. These changes must be made easy to implement without causing the whole system to fail.

1.1. Editing Input/Output

If input parameters or output variables change within the model, then these changes need to be reflected in the web page.

1.2. Editing Scenarios

Scenarios are a feature that adds user-friendliness to the product. They are default input parameter sets that will give users a way to run the model without prior knowledge.

1.2.1. Add scenario

Administrators can add to the list of scenarios.

1.2.2. Remove Scenario

Administrators can remove from the list of scenarios.

1.2.3. Update Scenario

Administrators must be able to update an existing scenario.

1.2.4. Update Scenario Interface

Whenever a scenario is added/removed/updated, it should be reflected in the user interface that displays them.

1.3. Code Modification

All of the product's code must be accessible by future developers. This will allow them to update the system as needed.

1.3.1. Developer Manual

A manual will list all of the functions of the system so that future developers can learn how the system works.

2. Monitoring

Monitoring is keeping track of how the system is running. This will be done through analytics, and will affect system configurations.

2.1. Software Configuration

There will need to be a configuration file that will determine important factors that may need to be updated in the future. For example, this configuration will include a value for the max amount of users that can access the model at once.

2.2. Constraint Configuration

Some input parameters must be constrained so that the user does not enter a value that will break the **ACGCA** model. These constraints may change, so they must be configurable.

3. Administration

Administration will be needed so that select individuals can access the application's back end and make updates where needed.

3.1. Administrator Environment

Administrators will need an environment that has access to the back end of the application. This will allow them to update it.

3.1.1. Administrator Log-In

The administrator will need to supply their credentials to gain access to this environment.

3.1.1.1. Administrator Information

An administrator's login information must be stored in a database to keep track of who has administrator privileges.

3.1.1.2. Administrator Authentication

Administrators must be authorized so that only trusted individuals can work on the product. This will be handled by Google's authorization feature and use the nau.edu domain.

3.2. Administrator Management

Administrators must be able to manage themselves and other administrators.

3.2.1. Add Administrator

Administrators should be able to add new administrators.

3.2.2. Remove Administrator

Administrators should be able to remove administrators.

3.2.3. Update Administrator

Administrators should be able to update their log-in information.

2.4 Performance Requirements

Performance requirements are how well the product performs its functions. These are decided based on product usability. They will be further considered during the user testing phase in Spring 2020.

- Parallelization

  The program is designed for primarily with educational use in mind which means that it might be used in a situation where more than 20 students need to use the program at the same time. We need the program to have the ability to perform the parallelization required so that students do not need to wait for too long before their input data is executed.

- Queue

  The number of people that are using the program might be higher than the number that the parallelization can take care of. In this case there will be a queue function. Users will be informed about where they are in the queue when they run the program and they can estimate how much time they need to wait before their input data will run.

- Time to receive output

  When a user reaches the front of the queue, they will need to wait until their input is executed by the program. Since the program will not run for a long time, users' wait time for the program execution should not be longer than 20 seconds.

- Time to interact with the website

  The website should not take a long time to show up on the user's screen once they navigate to the website. The time should be less than 3 seconds.

3. Potential Risks

As seen from our performance requirements, we have a few risks to consider. Some of the performance requirements are mitigations to these risks. However they do not cover all of the risks.

Risk	Likelihood of Happening	Impact
Steep Scalability Curve	30%	Moderate
Low User Retention	50%-60%	High
Lack of Funding	95%	Moderate/High
Complex Model Input	80%	Moderate

Table 1: Potential Risks

As seen in Table 1, there are 4 risks that we anticipate: steep scalability curve, low user retention, lack of funding, and complex model inputs. Each one of these has a likelihood of happening and a size of Impact. These metrics are the result of our analysis before our mitigations take place.

Steep Scalability Curve

A steep scalability curve occurs when the number of users exceeds the predicted growth of users in a period of time. This creates a steep growth curve in which we use to predict how much we need to scale for those users which are directly proportional to each other.

The likelihood of this happening is fairly low. This is because the main groups that our product is targeting are undergraduate and graduate students to be used within classes. We predict these groups will not grow too fast.

However, the impact that this can cause, if it does occur, is fairly moderate. It would be more serious than an inconvenience to our client, but not severe enough to stop the product from existing.

We devised a couple of mitigations that would help minimize the impact of this risk:

1. We would queue the user's inputs and process them as soon as possible. This helps because it moves the processing and calculations off the critical path which would slow down the application for everyone.

2. All the user-facing operations are all client-side operations, meaning that the user runs these operations on their own computer through the browsers.

Low User Retention

Low User Retention occurs when there is a low conversion rate from users that are checking out the product to users that stay and continually use the product. As time goes on the chance of Low User Retention would increase, due to this fact we are only concerned with a couple of years that follow the end of our involvement.

The likelihood of this risk occurring is around fifty to sixty percent. This is paired with the impact of this risk which is high, meaning that if this risk happens it could lead to our product from not being useful and taken down.

We devised a couple of mitigations to minimize the impact and likelihood:

1. We have consulted UX designers to improve the user's experience and plan to consult with more designers as our design matures. This will not ensure user retention, but this will help improve the likelihood of this happening.

2. We will try to keep the time to interaction, which is the time it takes a website to be interactable, under three seconds for a typical school computer. This reduces the likelihood that User Retention is lost to website performance.

Lack of Funding

Our clients have to pay for any costs that our product incurs with their own money. This limits our options in what we can use. However, there is not any way to avoid costs due to the platform the product is located on. This is where the risk happens. The risk is when the costs that incur surpass the expected amount.

The likelihood is 95% because, without planning and preparations, the chance of getting billed unexpectedly is almost certain. The impact of this is moderate/high because if funding runs out, there is a possibility that the application will be taken down. However, that is the worst possible scenario. The best possible scenario is that our clients would have to continue to pay for the application. This risk also assumes that our clients do not get any funding in the future.

The mitigations for this risk are:

1. We would run non-critical processes on a local desktop, which would move to cost of maintenance from of subscription cost for a server.

2. We will develop on our local machines to ensure that all costs are due to the released product and that nothing that we do will affect the billing

3. We will discuss options with our client for what would be the best option to host the product. This planning will let us set an expectation on what to expect with our client to prevent any unexpected costs.

Complex Model Inputs

The **ACGCA** model has 30+ parameters that need to be entered for it to run. The risk comes in when a user cannot understand what each parameter does or how it affects the inputs, therefore, not utilizing the **ACGCA** model fully.

The likelihood of this risk occurring is fairly high. However, given our target audience being undergrad and graduate students, they would have guidance from professors and lectures that would have more of an idea of what the parameters would be. The impact is moderate because users without guidance may be clueless, which can lead to them blindly putting in values, or worse leaving the application.

The mitigations for this risk are:

1. We would have default values for parameters so that users can explore an input or set of inputs and their effects.

2. We will provide a description, the units, and recommended range of values to input for each parameter.

There may be more risks; however, we are confident that our assessments of the risks encapsulate the majority of problems that will affect our product. We are also confident that the mitigation strategies will counter and lower the overall impact of these risks.

4. Project Plan

During the current semester, we met with our clients, Dr Kiona Ogle and Dr Michael Fell, once a week. To figure out our product requirements, we decided to divide it into several design decisions including: rest framework, tree visualization, website development service, and database. The ReST Framework requires us to create a ReST API to be able to mediate the information between the client-side and the server/algorithmic side. It should allow our business logic to work independently and mimic the structure of our architecture diagram. Tree visualization is a way to create graphical trees based on the input that a user enters. These trees should be dynamically created from the output of the **ACGCA** simulation. A website development service needs to exist within the program that is able to act as an interface for the user. A database is required to store and view the data of users when they complete a preliminary survey. Figure 7 outlines the progress of the current semester.

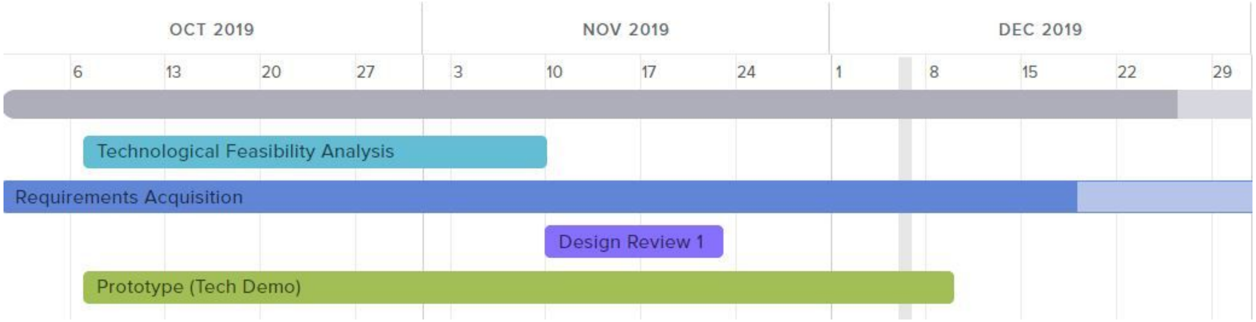

Figure 7: Schedule for Fall 2019

We have already completed our technological feasibility analysis from early October to early November. We met with our clients weekly and focused on building a prototype from the beginning of October to the end of the semester.

To better prepare for the months to come, we have set up a gantt chart (Figure 8) with the following milestones.

1. Set up Server
2. Set up Front-end Website
3. ACGCA model connection
4. Improve Website
5. Set up Back-end Database
6. Create Tree Visualization
7. User Testing & Improvement Based on Feedback
8. Final Product

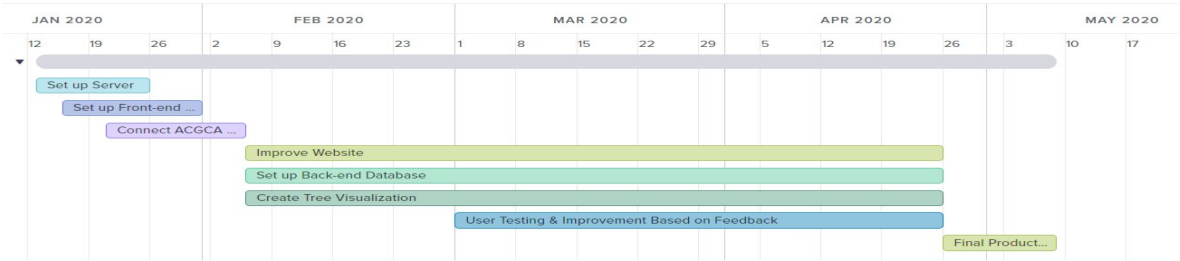

Figure 7: Schedule for Spring 2020

We will begin by setting up the server for the website. Then our front-end website will be created and we will focus on connecting the **ACGCA** model to it. For our front-end website we need to improve it based on the feedback from user testing. Meanwhile, we will develop both the back-end database and the tree visualization. Finally we will combine all our work together to achieve a functional product at the end of the semester that satisfies the needs of our client.

5. Conclusion

Various environmental factors affect the growth of trees and those factors are directly connected to climate change and our understanding of it. Being able to provide an easy-to-understand visualization that connects students and teachers to the **ACGCA** model can facilitate their understanding of tree growth and the effects that various factors have on it.

Our clients, Dr. Ogle and Dr. Fell, are biologists that have created a model that uses parameters and algorithms to simulate the growth of trees over time. The model is used to research the various factors that can affect our environment and how those factors are affected by climate change. This model has not been in too much use because it is written in R code and run through a command-line interface. This interface creates a divide between users and the model and our clients need a way to decrease this divide. They would like the model to be accessed by as many people as possible, especially students, so that anyone can see the effects of climate change on the world around us. The problem that our clients face is that they do not have the resources to build a user-friendly software that connects their **ACGCA** model to a visualization software and host it on the internet themselves. Dr. Ogle, in particular, is not a software developer and has spent most of her time building the model and the algorithms that simulates the growth of a tree over time. Our clients would like the model to be as accessible as possible and in a way that stimulates the learning process. They also want the model to be visualized so that the user will want to interact with it.

Our solution is to develop a website that hosts the connection between users and the **ACGCA** model. This will help our clients obtain the product that they envision. The user will be able to access the website from any computer and be able to play with various inputs that affect the visual output of the tree. Connecting the model to a visualization software will create an appeal for students to interact with the model and enjoy learning about the subject matter.

Gathering the requirements for our product was a crucial step that will lead to building a functional product that our clients envision. The most important requirements that built the foundation for the rest of our requirements were the client requirements, which specified some of the domains and technologies that needed to be used. The product being hosted online is a requirement that the clients have specified, which limits us to technology that would need to be hosted online. We then needed to choose that technology, such as servers, databases, web frameworks, and visualization software that will work in the domains that the clients want. After deciding on which technology we should use, we needed to decide on how each part would work, creating a whole new set of requirements. Some of these requirements include: the web framework needs to speak with the **ACGCA** model, the visualization software needs to be interactive, the database needs to be accessible to the administrator. Developing the requirements for each stage of the design process has created a clear set of guidelines that will lead us to the completion of the product.

Team TreeViz has made a lot of progress in the completion of this project. We have developed many sample programs of each technology that show their effectiveness as well as their functionality with the requirements that we have established. We have built web pages that are hosted on local machines for development and we have been able to access a sample visualization through that web page. A lot of planning has been done to provide a smooth transition into the implementation process of our product and we have already done minor development to test our technology. Dr. Kiona Ogle and Dr. Michael Fell of Ogle Labs have specified that they need this product to be hosted online, connect to the **ACGCA** model, provide a visualization for students and others to interact with, and have a way of storing user information for analytics. This has given us the foundation for where our design now rests. Every piece of technology that we have tested and every requirement that we have gathered pertains to our clients' main requirements: developing a database and how it should work with a website, how the website will fetch data from the model, how the visualization will work to be the most appealing, etc. We foresee the development being relatively easy because of the amount of designing and planning that we have done. Meeting with our clients frequently has given us a clear understanding of the product that they want and how to make that happen. The members of TreeViz are excited to build this product for our client and are confident in our abilities to do so.