VirusWatch

Requirements Specifications

Kevyn Sisante
Anas Albedaiwi
Colton Barboro
Bryan Stahman
Ziang Zhou

Project Sponsor: Dr. Crystal Hepp
Project Mentor: Volodymyr Saruta

November 1, 2020

Accepted as baseline requirements for the project:

Crystal M. Hepp
Client

Team Lead

TABLE OF CONTENTS

1. Introduction..3

2. Problem Statement...4

3. Solution Overview...5

4. Project Requirements...6

4.1 Functional Requirements...6

4.1.1 Data Analysis...6

4.1.2 Database System...8

4.1.3 User Authorization and Authentication.....................................9

4.1.4 Messaging and Alert System.....................................12

4.2 Non-functional Requirements..13

4.2.1 Analysis Runtime Performance.....................................13

4.2.2 Roles Based Security Performance................................13

4.2.3 Usability and Readability...13

4.2.4 Message Alert System Performance..............................13

4.3 Environmental Requirements...14

4.3.1 Browser Support...14

4.3.2 Data Backup and Storage..14

5. Potential Risks...15

6. Project Plan...17

7. Conclusion..18

1. Introduction

Since SARS-CoV-2 was first detected in the United States, more than 46 million people have been confirmed to be infected with the virus and 1.2 million people have died as of November 1st, 2020. In a high-density residential environment, the spread and speed of SARS-CoV-2 will increase significantly, leading to population outbreaks. Although the number of cases in Arizona has declined, compared with other states, Arizona has taken advantage of it. The terrain (located in a desert area with a small population) is densely populated and people have considerable advantages in maintaining social distance. The virus is not easy to break out in the crowd. It still cannot be guaranteed that the number of cases will not increase.

The task of our group is to design a website for our client Crystal Hepp to detect the virus content in wastewater and analyze the virus data. Crystal Hepp is an infectious disease expert and currently works at the Institute of Pathogenic Microbiology at Northern Arizona University.

Prior to this, Crystal and her team were tracking the coronavirus by collecting sewage samples. A lot of manual testing is currently underway, but from a long-term perspective, such testing cannot last forever. An application is currently being developed that will be a website that can automatically collect and analyze virus data and create a table for users to view the data. This will be a low-risk, high-yield project. Although the development process will not be particularly smooth, considering the need to collect and analyze data, once successful, it will potentially have an impact on the entire world. At present, the collection of wastewater is still primarily done near Flagstaff, but future collection areas will be expanded so that the data chain will be more complete, and therefore more accurate in predicting the location of the next viral outbreak.

2. Problem Statement

The detection of the infectious disease, COVID-19, has resulted in nearly 46 million cases and 1.2 million deaths globally as of November 1st, (World Health Organization). Consequently, the global community has been called upon to fight the pandemic by reducing its spread. Since the pandemic will probably stay for an unpredictable time, people must find ways of living with it while they continue performing normal activities while protecting themselves. For instance, learning institutions are in a dilemma in that they are to shift from holding in-person classes to online learning so that they reduce densely setting that limit personal contact and spread of the infection. Implementing in person learning is likely to increase the spread of the virus, especially when students fail to comply with wearing face masks. Currently, there are no resources which could implement continuous universal testing of the virus. Hence our client, Dr. Crystal Hepp, proposes the use of wastewater-based testing surveillance as a better solution to monitor the community and identify new cases of infections before outbreaks. Such surveillance can give important information to policy-makers and healthcare officials to take appropriate actions to mitigate the situation.

The current collection of data in communities is conducted manually by collecting samples in the potential areas with a high risk of infections such as outside dormitories. The primary issue at hand is that the overhead between the collection of data, the processing of data, and the delivery to health agencies is too high as researchers are currently forced to process and deliver this data manually. The resulting data collected is usually complex as it covers many large areas, consequently, dealing with large samples of data is difficult in terms of collection, storage, manipulation, and distribution. Even though technology can be implemented to help the current situation, no implementation has been proven and standardized. For instance, current sample reports are submitted in the form of charts and spreadsheets which are updated and submitted manually.

3. Solution Vision

The solution vision of this project focuses on having an efficient way of dealing with large and complex data. Since this project involves the collection of data over a long time, better mechanisms should be deployed to accommodate an ever-increasing data complexity. It is at this point where technology would be adopted to gain a maximum solution to data collection, manipulation, and storage. The development team proposes to develop a secure website and mobile application that can operate on smart devices. The idea will allow users with hand-held smart devices to simply log in and access various results and some analyses. This would enable users to compare data and analyses involving case counts in any location to take appropriate actions. This implies that this method would always allow users to access information remotely. Even though this project mainly focuses on COVID-19, data relating to how it is spread should be archived safely so that it would be used in related infectious viruses.

The development implemented also integrates new technologies and techniques to maximize the efficient monitoring of spread. It also includes informatics tools to make it easier to collect, manipulate, and retrieve data. The final product of the project will include the following main features:

- It provides a simple website with a clear GUI for main operations such as exporting data, searching data, and uploading data.
- It is safe and private in that users cannot access information in a different agency.
- It allows communications among users through text messages and emails to notify about the new results.
- It gives a secure web application with clear functional permission architecture based on different roles and user authentication.
- It gives a clear outline of summarized graphical data included in the system to enhance easier monitoring.
- It supports different types of data and saves on specific database architecture.

4. Project Requirements

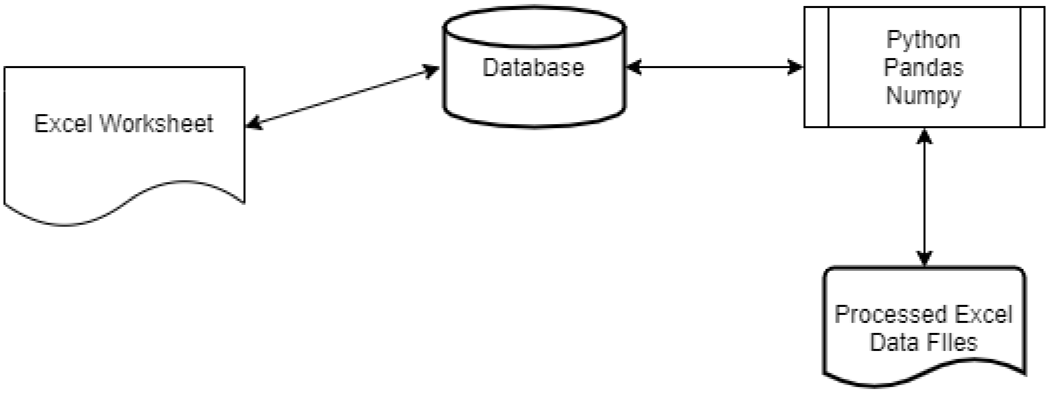
This section will provide a layout of the complete requirements for the secure web application. The requirements listed below will be split up into three categories; functional, performance, and environmental requirements. The functional requirements will define functions and specifications that the stakeholders will be utilizing on the web application. Next, the performance (non-functional) requirements will demonstrate the expected performance of each of the functional requirements. Finally, all constraints set by Dr. Hepp and NAU will be listed as the environmental requirements. The gathering of requirements for each category were done through extensive research by the team members and through weekly meetings with the client, Dr. Crystal Hepp.

4.1 Functional Requirements

4.1.1 Data Analysis

One of the primary functions of the application will be to take in data files uploaded to the web server and process the data by analyzing and calculating appropriately, after which it will have to graphically display the results to the user.

Figure 4.1.1.1: Simple diagram depicting the data analysis process in the application



1. Processing of Data

The web application must be capable of taking in viral wastewater data and processing it to display appropriate data to the stakeholders in question. The application will be expected to do the following:

- ❑ Take in data collected from wastewater collection, process it, and return the results to the stakeholders by displaying them to the front end GUI so that they may view the data.
- ❑ Compare any type of surveillance data (e.g. case counts, % positive, wastewater signal, viral copies per sample) so that it can be determined how far in advance wastewater surveillance can "alert" stakeholders of an upcoming outbreak.
- ❑ Take existing datasets and compare them to each other by epidemiological week.
- ❑ Once the data is processed, it must be displayed with a caption that explains what the data means for the most recent epidemiological week.
- ❑ Data will have to be able to output according to the stakeholder's given criteria in a swift manner. (i.e. ~5 seconds at most)

Overall, the application will be expected to process the data and display it to the stakeholders according to their specified search criteria.

2. Graph Representation of Data

Python's data graphing systems will be used to push graphed data to the user and will work with the front end to update data appropriately.

After the data has been processed, the web application will be expected to visualize the data in a variety of graphs to the stakeholders using a variety of graphical formats.

- ❑ Must be able to output the resulting data graphically according to the specifications of the stakeholder. Said specifications would be what kind of graph they wish to view, as well as the dataset they wish to see.

❏ Must be able to contextualize the data into different graphs as selected by the stakeholder. If the stakeholder wants to view the selected dataset in a variety of different graphs, they should be able to utilize a dropdown menu to select new graph formats to quickly reformat the data into a new graph.

❏ Graphs must be captioned with an explanation as to what the graphed data means in relation to the epidemiological week it represents.

❏ When selecting datasets to graph, it must be able to compare the data by “zooming in/out” scale of time and juxtapose time series. Python libraries such as matplotlib and plotly.

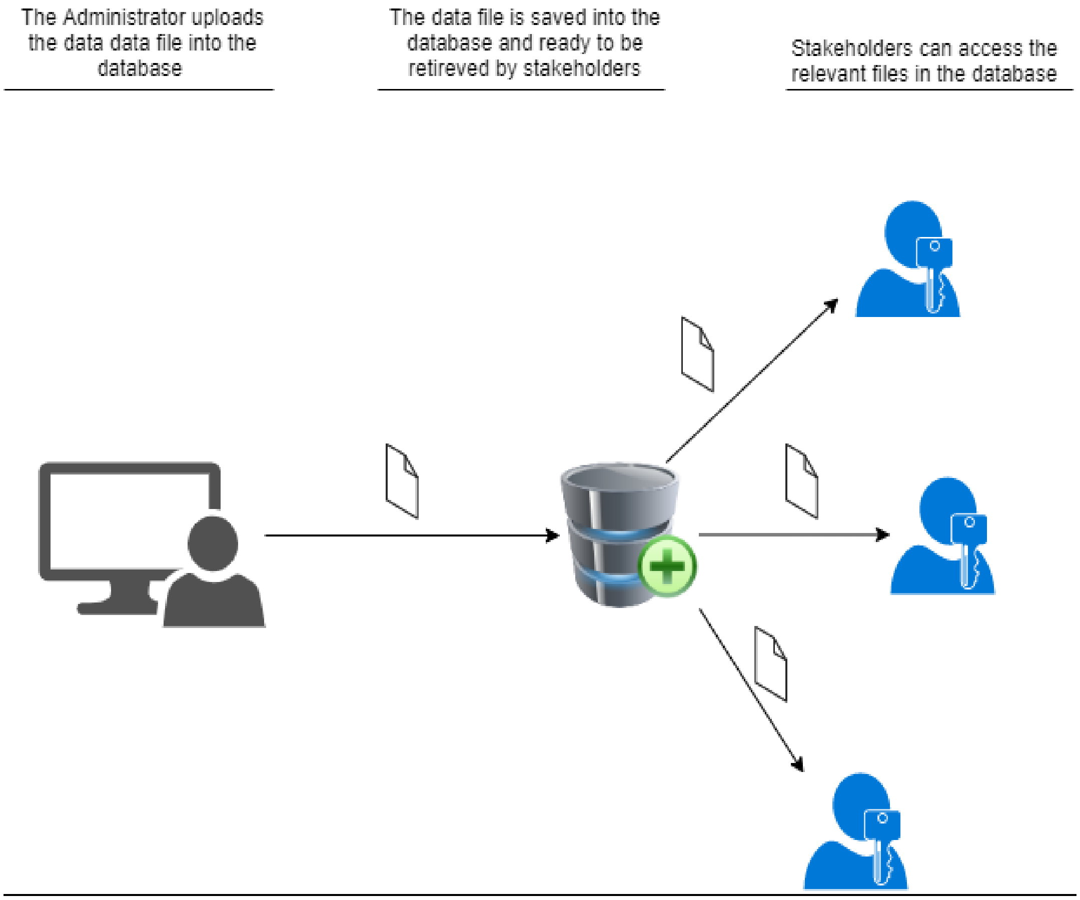
❏ Have the ability to compare datasets to give stakeholders the ability to see how data has changed over time. This may be in the form of graphing the two datasets and showing them side by side to give the stakeholder the ability to directly compare the two datasets. They would be able to choose this option on what would likely be a drop down menu in the GUI.

When implemented, the stakeholder is going to be able to view data graphically from a selection of graphing formats and will be able to compare data from different datasets.

4.1.2 Database System

The application will need a database system to save and pull datasets as needed. It will be responsible for saving all of the data that is recorded for wastewater analysis and using that data to calculate surveillance data and show the user the results of the calculations in a digestible format. This database will also be responsible for supporting the exporting of datasets to the appropriate users upon request. It is important that the application have an optimal database especially because the functions of this application are data heavy and performance of data transfer to and from the database needs to be secure and quick as it will be responsible for all of the data transfer and storage actions.

Figure 4.1.2.1: Simple diagram depicting the saving of data to the database by the administrator and stakeholders retrieving the data from the database.


The Administrator uploads the data data file into the database
The data file is saved into the database and ready to be retireved by stakeholders
Stakeholders can access the relevant files in the database

1. Save datasets to database

The web application will need to support the ability to allow users to upload supported formats of datasets into the application's database so that the data may be analyzed by the application and the results output to the user. This entails the following:

- ❑ Users must be able to save their wastewater datasets into the application using supported formats (currently .xlsx files) so that the application can process the data. This may be done by selecting the option to save or export files by choosing the corresponding option on a drop down menu in the GUI.
- ❑ The application must be able to store the data for long periods of time. This means that organization of the data needs to be kept in mind as it will be expected to be searchable by epidemiological weeks and pulled to analyze and compare datasets.
- ❑ Keeping security in mind, the application will also have to keep security in mind when storing data in the database. Users will have to be restricted to saving and accessing data only relevant to them and not other users or user agencies.

2. Export datasets from database

On top of saving data and storing it in the database, another requirement will be to support the option to export the datasets upon user request.

- ❑ When requested, the user should be able to have a copy of the dataset exported so that they may be able to use it as they see fit.
  - ■ The available datasets to be downloaded will by default be the processed data with the calculated surveillance data. Another option will be to export the raw datasets to the user.
- ❑ Export of data must also be protected by roles based security to prevent the exporting of data not related to the user. As mentioned in the data saving section, the operations must also be restricted to the data relevant to the user.

Overall, when implemented, the user should be able to retrieve and view data from the database as well as have data exported to view for themselves.

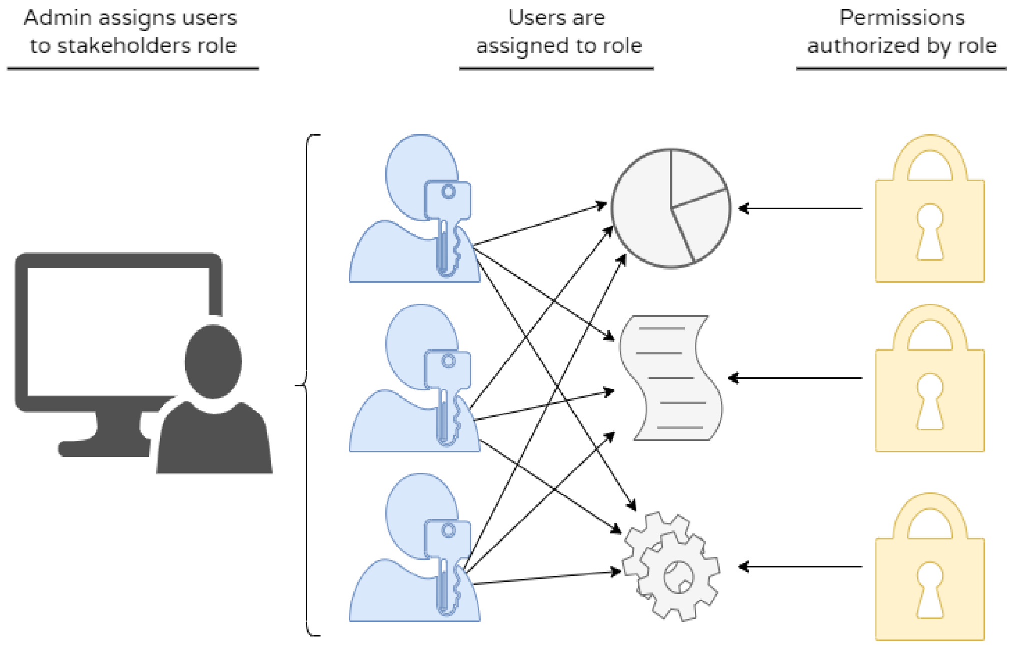
4.1.3 User Authorization and Authentication

The VirusWatch web application will be expected to handle copious amounts of data from different stakeholders. This data will not be available to the public and should only be accessed by users with the proper permissions. The data inputted by stakeholders should not be accessible by any other agency. The goal of user authentication and authorization is to allow efficient data entry with an emphasis on safety and privacy.

1. Roles Based Architecture

The VirusWatch web application will implement Roles Based Access Control (RBAC) to allow users to gain permissions that are dependent on the role given by the owner, in this case, Dr. Crystal Hepp. As seen below in Figure 4.1.3.1, the admin will be able to assign users to a role. Once a user has been assigned a role, they will be granted permissions based on the given role. Currently, the VirusWatch web application will be designed to allow two roles, the administrator and the stakeholder. As the application grows larger in scale, more roles and permissions can be added accordingly.

Figure 4.1.3.1: Simple diagram of role assignment of users by the administrator


Admin assigns users to stakeholders role
Users are assigned to role
Permissions authorized by role

2. User Permissions

User authorization, or permissions, will play an important role in ensuring the safety and privacy of one agency to another. As previously stated, the roles that are to be implemented are as follows:

- ❑ Admin - The admin will have access to all features provided by the web application as well as additional permissions to assign roles to the other agencies that will be using the web application. The admin will also be able to view all stakeholder data including all imported files and graphical analyses.
- ❑ Stakeholder - The stakeholder role will be able to upload data, use graphical analysis tools provided by the application, browse/search their previous analyses, and export data. Users with this role will not be able to access data other than their own.

3. User Authentication

Aside from granting permissions to specific users, the VirusWatch web application must use a specific user's credentials in order to be properly authenticated. To accomplish this feature, conditionals will be hardcoded into either the DBMS or front-end authentication API that will verify whether the user trying to access data has the correct permissions. Another method to properly authenticate users would be tags. Since data will be coming from different agencies, each user will receive a tag that associates the user with a location name. If a user tries to access data that does not match their specific tag, permissions will not be granted. This can help increase performance time as it does not need as many checks as hardcoded DBMS conditionals.

4.1.4 Messaging and Alert System

The VirusWatch web application must also support sending emails and text messages to stakeholders. These alerts can be sent either immediately after results are available or when an analysis displays extreme readings. Email addresses and phone numbers will be acquired through the database upon user sign up and fed into secure, automated email and SMS API's for quick and reliable delivery.

4.2 Non-functional Requirements

4.2.1 Analysis Runtime Performance

The analysis runtime in the application will depend on the size of the data files being analyzed, as well as the algorithms used to analyze the data files. It is expected that the runtime will be consistent with each run of the data analysis programs in the application as it will re-render the calculations according to the specification of the user each time a specified instruction is submitted on the application.

4.2.2 Roles Based Security Performance

Performance of user authorization and authentication will differ based on the methods currently being tested. User authorization methods, such as log-in and admin role assignment should execute in little to no time, barring an unstable internet connection. Log-in functions will be taken from premade MongoDB libraries and should be optimized out of the box. Role assignment will only be accessible to the administrator and will be as fast as a click of a button.

4.2.3 Usability and Readability

Because the application is going to be used by users from various agencies, it is important that the site have a clear design that makes the application friendly to navigate and instructions for how to engage in the site's functions. It is also important to keep documentation of the site detailed and up to date as there may be another team that may need to look at the source code for the site down the line, so keeping the codebase clean and documented should be a priority so that others as well as the development team can keep a clear image of the project.

4.2.4 Message Alert System Performance

The performance of our alert system is an important non-functional requirement in that alerts need to be sent out as soon as results are processed, with as little delay as possible. Barring any server issues on email servers and cell towers, the time-to-alert methods will be expected to deliver an email and/or SMS text messages to a specific user within 10 minutes of the current dataset being processed. The alert system feature is able to be toggled on or off, but will be turned on by default. Barring any server issues within email servers and cell towers, automated email and text messages should perform efficiently and quickly.

4.3 Environmental Requirements

Although there are not many constraints put in place by Dr. Crystal Hepp, there are a few limiting factors that the team must take into consideration. These factors include:

- ❑ Browser Support
- ❑ Data Backup and Storage

4.3.1 Browser Support

As a small team, the website cannot be guaranteed to be supported by the multitude of browsers that are available today. According to StatCounter, a free online stats tool, the two most commonly used web browsers are Google Chrome and Safari, totaling their market share at 66.7% and 17.24% respectively. The VirusWatch team will do extensive testing on these two browsers to ensure that usability is equally efficient on both browsers. By using these browsers as our benchmark, further development to support older or less used browsers will become an easier task.

4.3.2 Data Backup and Storage

Since VirusWatch must implement features that allow users to recall sensitive data or save multiple analyses, a measure must be put into place that will reliably store and/or backup data in the event of a server crash or outage. A physical server would be a potential fix to this problem due to VirusWatch storing data on a harddrive rather than using the web server's memory. A physical server is a fairly inexpensive solution that can be placed in a small office or work area and can be periodically upgraded as need be. With a physical server, data can constantly be backed up and is not reliant on the VirusWatch web server to be active in order to access data.

5. Potential Risks

Due to the critical and sensitive nature of our application's data, failure in any part of our system could lead to severe consequences for the Northern Arizona area. It is incredibly important that our application calculates and displays the data correctly, securely, and promptly in order for policymakers to make informed decisions about how they should proceed. Potential risks have been identified that our application could encounter and have weighed their probability of occurring and severity.

5.1 NAU Servers Go Down

Risk	**Severity**	**Probability**
NAU Servers Go Down	Moderate	Moderate

The application and database will be hosted using NAU's web server, meaning that it is reliant on the servers being online and functional. In the past few months, NAU's server has caused a few problems for our team website, causing it to become inaccessible for a short period. It was determined that the application will be vulnerable to similar problems. If the server was to go offline for maintenance, we will likely have advanced warning and will be able to notify stakeholders accordingly. It can be determined that this risk is moderately likely and has a moderate severity.

5.2 SQL Injection Attack

Risk	**Severity**	**Probability**
SQL Injection Attack	High	Low

Because user input will be used to construct SQL statements that communicate with our database, the system will be vulnerable to a SQL injection. A SQL injection is one of the most common hacking techniques that involves the placement of malicious code in SQL statements, via web page input. The rating of the probability of this attack is low, because with proper prevention techniques, the threat can be identified and halted.

5.3 User Account Security Breach

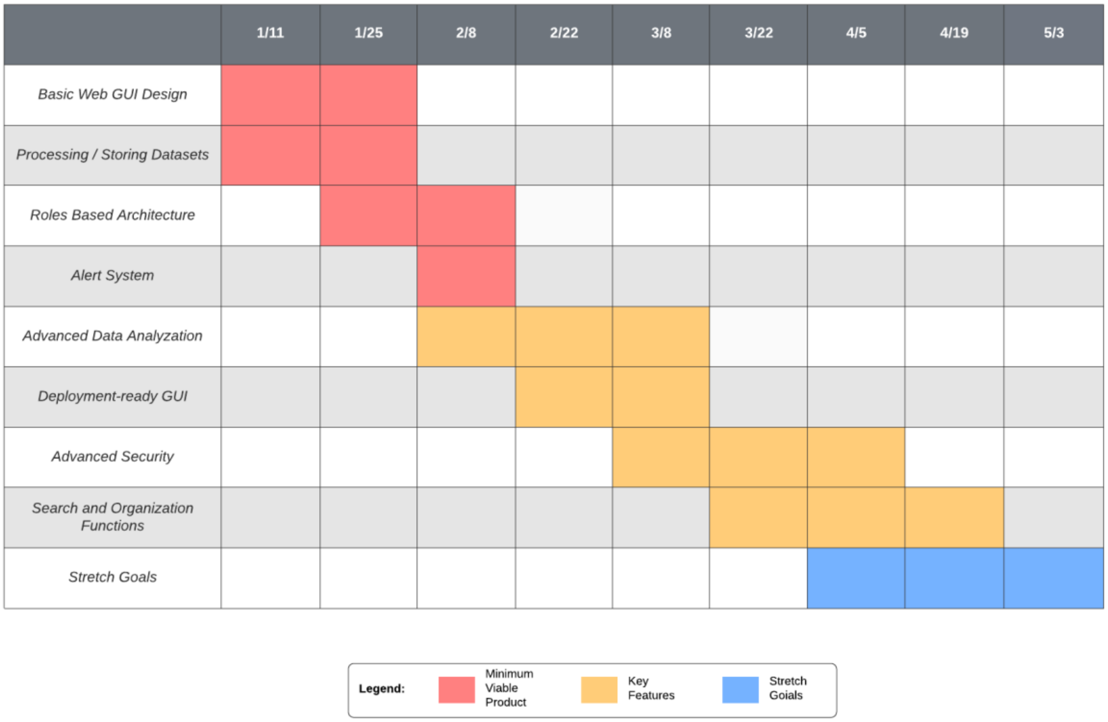
Risk	**Severity**	**Probability**
User Account Security Breach	High	Low

As is the case with all applications that have user accounts, there is the potential for an account to become compromised. A malicious actor could obtain the login information of one of our users in many different ways. This would likely lead to a breach of privacy which would be very ideal to avoid at all costs. With proper security measures, the risk can be mitigated. Strong password requirements will be included during user account creation and give the option for users to enable two-factor authentication.

6. Project Plan

After many weekly exchanges with our client, the client's needs have been determined. The Requirements Specification document is currently being completed based on information received from the client. At the same time, the Technical Prototype Demonstration is also being prepared. The demo is being constructed according to the requirements of the client. The demo will serve as a proof of concept for our minimum viable product.

Figure 6.1: A rough outline of the projected development schedule



	1/11	1/25	2/8	2/22	3/8	3/22	4/5	4/19	5/3
Basic Web GUI Design	🟥	🟥							
Processing / Storing Datasets	🟥	🟥							
Roles Based Architecture		🟥	🟥						
Alert System			🟥						
Advanced Data Analyzation			🟧	🟧	🟧				
Deployment-ready GUI				🟧	🟧				
Advanced Security					🟧	🟧	🟧		
Search and Organization Functions						🟧	🟧	🟧	
Stretch Goals							🟦	🟦	🟦

Legend: 🟥 Minimum Viable Product 🟧 Key Features 🟦 Stretch Goials

A rough schedule has been outlined by our team (Figure 6.1). In the beginning of the semester, the creation of a minimum viable product will be the focus. A barebones GUI will be designed and implemented for the website. At the same time, basic functions such as a Roles Based Architecture will be created.

7. Conclusion

Although COVID-19 is a very dangerous infectious disease, a way must be found to coexist with the coronavirus. In order to detect the virus content in different regions, Crystal Hepp suggested using wastewater-based testing methods to monitor the virus flow in the community and take action before the outbreak. Before taking over this project, all data collection was done manually, but the impact was that the collection of data was too complicated, the variety of data was not easy to handle, and the data storage was too difficult. The current sample reports are made into charts and tables for manual update and upload, which is very labor intensive.

Based on the above issues, the team had decided to develop a secure website application program that is long-term, large-scale, automatically collects and sorts the virus content data in wastewater, and uploads the results to the database. This will enable users to compare data and analysis of the number of cases involved in any location in order to take appropriate action.

The development team will lay out a secure web application according to three categories: function (what operations can be implemented on the web page), performance (ensure that the function meets the requirements) and environmental requirements (ensure that the web page is safe). One of the main functions of the application is to receive data files uploaded to the web server and process the data, such as case count, % positive, etc. After the data is analyzed, Python's data analyzation system will be used to push the drawn data to the user. Currently Virus Watch web application will be designed to allow two roles, the administrator and the stakeholder. As the application grows larger in scale, more roles and permissions can be added accordingly. When Virus Watch is completed, it would be ideal if the project makes a large contribution to society.