CySec-Game

DESIGN DOCUMENT

sdmay21-50

Client: Manimaran Govindarasu

Team Members: Harrison Majerus,

Nicholas Battani, Hayden Sellars,

Jonathan Greazel, Joseph Strobel, Stefan Peng

https://sdmay21-50.sd.ece.iastate.edu/

10/04/2020

Executive Summary

Development Standards & Practices Used

- Agile development
- CVSS
- NVD

Summary of Requirements

- The software shall be remotely accessible through a web application
- The software shall run use-cases of risk assessment scenarios using PowerCyber test bed
- The software shall be integrated with existing testbed and simulation tools
- The software shall utilize Game Theory algorithms for risk assessment of cyber physical systems like the smart grid and provide best strategies to mitigate the risk.
- The software's UI design shall encourage ease of use and work to minimize clicks per action
- Different environments (Windows, Linux, etc.) shall have no effect on software usability
- The software shall cost no more than $0
- A working proof of concept and completed documentation shall be delivered by the end of Spring 2021 semester
- Any sensitive user information shall be stored in a safe and secure manner
- All work shall be original for our development team with credit given to proper sources

Applicable Courses from Iowa State University Curriculum

- Com S 228
- Com S 309
- Com S 319
- Math 314
- CPRE 530
- SE 329
- SE 339

New Skills/Knowledge acquired that was not taught in courses

- Game theory
- Cybersecurity for critical infrastructure
- Network modeling

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

1 Introduction

1.1 Acknowledgement

We would like to thank our project advisor/client Dr. Manimaran Govindarasu and graduate students Burhan Hyder and Kush Khanna for assisting us in our understanding and design of this project.

1.2 Problem and Project Statement

Critical infrastructure like power and energy systems are often vulnerable to cyber attacks. Mitigating cyber risk to critical infrastructure is an important part of the design and maintenance of these systems. These power and energy systems commonly use legacy devices where complete upgrades are uneconomical, therefore, risk assessment plays an important role in selectively securing vulnerable or high-risk assets.

The goal of this project is to develop a software tool that helps the critical infrastructure industry to assess cyber risks to power and energy systems and to optimally allocate cybersecurity investments to mitigate the risks. This tool will use game theory models to identify high-risk targets and will directly interface with the PowerCyber testbed for cyber-physical risk assessment and mitigation.

1.3 Operational Environment

The operational environment for this product will be dependent on the user. The end user is targeted to be a system administrator who is responsible for network security. The operational environment will involve the user accessing a front-facing user interface and then using that user interface to run network security analysis on their network. There will be a back end that contains a game theory model as well as an optimization engine. They will be able to use this tool to optimize their cyber security investments. This tool will need to be able to work on all modern operating systems (Windows, macOS, Linux) and be easily accessible over the internet.

1.4 Requirements

Functional Requirements

- The software shall be remotely accessible through a web application
- The software shall run use-cases of risk assessment scenarios using PowerCyber test bed
- The software shall be integrated with existing testbed and simulation tools
- The software shall utilize Game Theory algorithms for risk assessment of cyber physical systems like the smart grid and provide best strategies to mitigate the risk.
- The software's UI design shall encourage ease of use and work to minimize clicks per action

Non-Functional Requirements

- Different environments (Windows, Linux, etc.) shall have no effect on software usability
- The software shall cost no more than $0
- A working proof of concept and completed documentation shall be delivered by the end of Spring 2021 semester
- Any sensitive user information shall be stored in a safe and secure manner

- All work shall be original for our development team with credit given to proper sources

1.5 Intended Users and Uses

Users and Uses

- Cyber Security Analyst - This tool will be used by a member of a company who analyzes risk of cyber attacks and what kinds of damage attackers can do to infrastructures (ie. Electrical Power Grids, advanced manufacturing environments, etc.)
- Cybersecurity Investment scenarios will also be produced by the tool

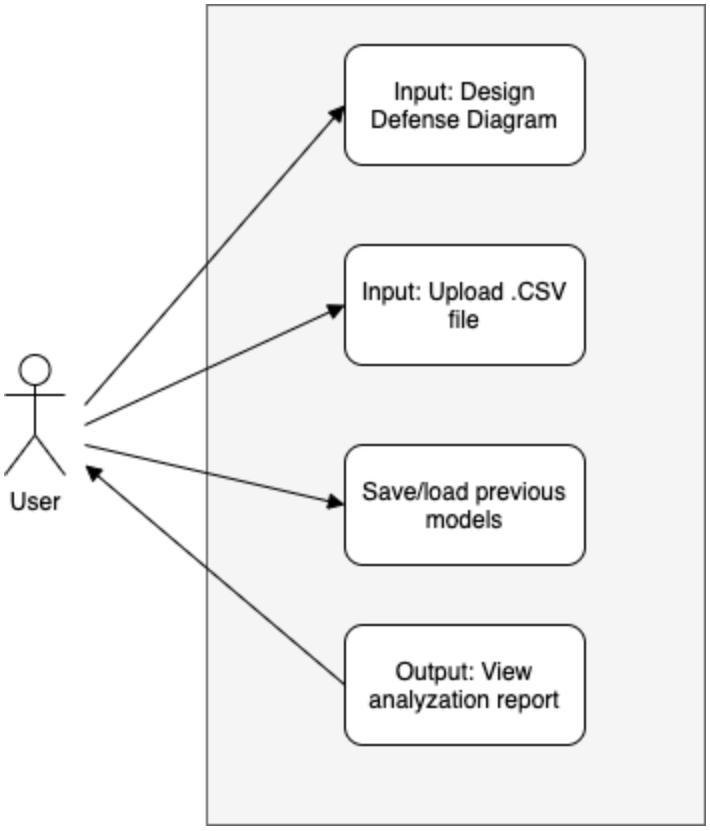

Figure 1.5: Use Case Diagram

1.6 Assumptions and Limitations

Assumptions

- Cyber vulnerability assessment for the possible cyber-attacks targets must be done offline by the user before submitting.
- User input will be in the form of .csv and selection of nodes later on.
- Algorithms will be provided by client and Graduate students.
- Server infrastructure will be provided by Iowa State University.
- The tool will be available via a web application.

Limitations

- Game Theory knowledge of the team is minimal.
- There will be no financial resources provided to the team.
- Project will be developed entirely remotely.

1.7 Expected End Product and Deliverables

User interface (3/19/2021)

- The user interface (UI) shall be in the form of a web application that can be accessed from any desktop web browser. Users will be able to enter parameters for attackers, defenders, and the network model. These parameters can be either created from scratch or edited from presets. The UI shall display the assessment results to the user.

Backend (4/1/2021)

- The backend will receive parameters and models from the UI and will analyze the given parameters and models using optimization algorithms. It shall support the existing cyber security simulation tool and shall utilize one or more game theory algorithms. The backend will send the results of the analysis back to the UI.

2. Project Plan

2.1 Task Decomposition

User Interface

- Network model
    - View network model
    - Create network model
    - Edit network model, possibly from presets
- Attack and defense parameters
    - View parameters
    - Create parameters
    - Edit parameters, possibly from presets
- API
    - Send data to backend
        - Data from CSV file with model and/or parameter data
    - Receive data from backend
    - Display data from backend
    - Standard user and admin user login

Backend

- API
    - Receive data from UI
        - Imported data from CSV file

    - Send data to UI
    - Integrate with simulator tool
    - Authorize user logins
- Optimization
    - Select appropriate optimization algorithm(s)
    - Implement optimization algorithm(s)
    - Analyze model and parameters using optimization algorithm
- Database
    - Construct required tables to store model and parameter data
    - Create required queries to load model and parameter data

See Section 2.4 for details on dependencies and timelines.

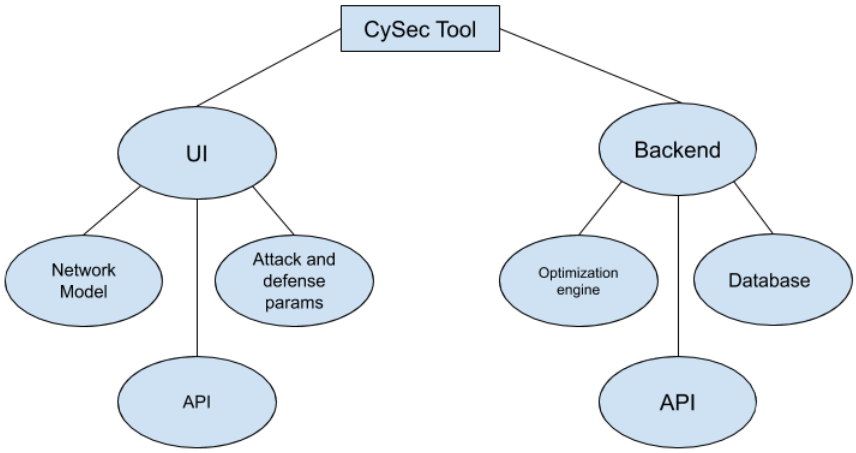

Figure 2.1.1 - Task Decomposition Diagram

2.2 Risks And Risk Management/Mitigation

Risks	Mitigation Strategy	Concern
Requirements change while development is already underway	○ A team meeting will be conducted to debate the best path forward. Considerations will be made in attempt to save already completed work and avoid backtracking ○ New requirements must be assessed for clarity and conflicts with our time budget	2
Project deliverables aren't complete upon due date	○ Must be avoided since delaying the due date isn't an option. The team will work diligently to meet intermittent deadlines, ensuring our work is completed on time.	8

	○ The team hopes to be well out of development and into testing by this time, so if a time constraint is met testing will face cuts rather than development	
User data is stolen and used to execute a cyber attack	○ We plan to store sensitive data separately and in a safe manner in order to mitigate this risk ○ In the case of a data breach we must first assess what was stolen, alert those who may be compromised, and patch the exploited vulnerability in a timely manner	10
The software's predictions are flawed/insufficient to prevent attack	○ Our application is meant to serve as a proof of concept, so further improvements may be made by other teams ○ The software will include data recording features to assess its failures/successes so our team may catch shortcomings quickly and act fast	6
Lack of team communication	○ This could result in any number of problems and must be avoided at all costs ○ Team members are actively engaged in channels of discourse such as Discord, Git, and semi-weekly meetings to mitigate this risk	8

2.3 Project Proposed Milestones

- Nov 15th - Design Document complete.
- Feb 5th - UI Design and UML Diagram complete.
- Feb 19th – Rough implementation of frontend and backend with simple requests between each side.
- Mar 5th - Testable backend game theory algorithm, front end can handle basic user input in the form of csv.
- Mar 19th - Frontend sends diagram to server's game theory algorithm, backend can communicate with simulator and respond with data in JSON format.
- April 2nd - Optimization of UI and better/more in-depth analysis of diagram and vulnerabilities. This includes full implementation of an algorithm provided by the client.
- April 16th - Full functionality: Frontend can receive diagrams and send them to the server via an API for analysis. Backend will run algorithms and in turn respond to the frontend with analysis.

- April 23rd - Presentation and visual displays regarding progress and/or completion of the project.

2.4 Project Timeline/Schedule

As mentioned above in the 'Project Proposed Milestones' the dates and planned objectives are scheduled into a Gantt chart, as documented here:

https://www.wrike.com/workspace.htm?acc=4089522#path=folder&id=573735000&c=timeline3&vid=8633423&a=4089522&s0=10&bs0=10&sd=0&st=space-573734921

This URL link will take the team to see the progress of our project throughout the 2 semester period of Senior Design.

Above Figure 2.4.1 shows Gantt Chart for months August - November

Above Figure 2.4.2 shows Gantt Chart for months January - April

2.5 Project Tracking Procedures

Our group will be using GitLab to track and manage our project. Within GitLab we will be using Issue Boards to create cards and track tasks that are being worked on and that need to be worked on. For non-technical communication we will be using a Discord server created specifically for our group. For all task specific communication, we will be writing our information within cards, commits, merge requests, and milestones within GitLab. All other technical information or designs will be within our shared folder on Google Drive.

2.6 Personnel Effort Requirements

Task	Hours
Design Document	50
UI Design	20
Front-end skeleton	10
Model editor	15
Back-end analysis algorithms	20
Game theory algorithm	30
Model feedback	10
Integration	25
Optimization	20

The tasks above are estimated based on the teams previous work experience. They are subject to adjustment as the team begins to meet checkpoints and begin tasks.

2.7 Other Resource Requirements

This project will require a server to host the service on. Our client and his research group will provide the server needed to host this service. This server will require a game theory model and a cyber optimization engine. The optimization engine will be obtained from our client.

3 Design

3.1 Previous Work And Literature

Previous work in cybersecurity modeling has focused on using attack trees and Petri nets to model networks and their associated attacks and countermeasures. Attack trees are the most basic model; implementing an attack tree model is straightforward, but the model fails to capture all the complexities present in real-life cyber-physical systems. An extension of attack trees are attack-defense trees. These models are able to model more properties of a system, but are still limited in modeling dynamic attacks and countermeasures. A master's student working with our client/advisor has previously developed a Petri net-based tool that allows users to draw a model diagram of a system and carry out security evaluations. This system more accurately models the dynamic nature of attacks and countermeasures.

There are a number of commercial and open-source products available to perform cyber security modeling. Each product has its own focus and is not suited for every application. For example, the CALDERA framework (The MITRE Corporation) allows users to run simulated breach-and-execution scenarios. This tool allows users to automate a simulated attack on their system to identify vulnerabilities. CyberX (CyberX) is designed to passively scan an IOT/ICS network to identify vulnerabilities specific to IOT/ICS systems. This is related to the focus of our project, which looks at power systems.

A key difference between our tool and the two commercial frameworks mentioned above is that like the PENET tool, our tool will allow users to model and analyze a network without a direct connection to the network. In mission-critical environments like power systems, this can be a benefit because vulnerabilities can be identified without risk of the assessment tool disrupting the network. Our tool will also implement a number of different analysis engines, including the attack tree model and other, more advanced game theory-based models.

3.2 Design Thinking

The largest factor of our design thinking was the end user. We assume the user to be computer savvy enough to navigate to our tool with little to no guidance, but our tool should be easy to use after that point. Because of this, we decided to choose a web-application for our front end to ensure that the user is able to easily navigate to the tool. This differs from our other idea of a desktop application. The ease of access is the biggest factor in why this is the best choice; the user doesn't need to install anything, they don't have to worry about not having the latest operating system, as long as they have a web browser installed they will be able to use our tool.

3.3 Proposed Design

Upon opening the web-application, the user will be prompted to login. From there the user will be able to view saved diagrams and be able to create a new model or upload a file to import a model. From the imported model or from scratch, the user will be able to interactively edit/create their model inside the tool. After finalizing their model, the application will convert the model into JSON format so the backend can process it.

In the backend, the model will be analyzed for risk and potential recommendations for improvement will be developed. The user will then get back a summary of risk with the

recommended changes to be made to their model. The user will also have the option save and export their results and model so they can use it again later.

The proposed design has full coverage of requirements.

3.4 Technology Considerations

Frontend

On the frontend, the team discussed possible options for our tool including Node.js, React, C# and Visual C#. Individuals on the team have experience with Node, React and C# while having no experience with Visual C#. We considered Visual C# because a similar tool was built with this framework, however our client Graduate advisors recommended against using Visual C# for its complexity, so we discarded that option. One member advocated for Node because of its ease of use and abundance of libraries. Members also were not excited about using C# because of our experience in the past and React seemed to have a bit of learning overhead to get started, so we have decided to use Node for the frontend.

Backend

On the backend, we know that we will be using a RESTful API to communicate with the frontend because this is an industry standard and we know we will not have issues interfacing between the two sides of the project. As far as backend languages are concerned, our client recommended using python because it has very good libraries for linear programming and Nash Equilibrium calculations which will be very useful when calculating risk assessment. Additionally, our backend members have experience writing in python. So, the team has determined that python will be the language of the backend. For a framework, the team will likely use Django or Flask as those are the two widely documented and maintained backend frameworks.

3.5 Design Analysis

The proposed design was agreed upon by the team and clients as a viable approach for the project. It is a reasonable development strategy because of the skill set and knowledge of each member in the group to complete their tasks. All of the technologies planned will fall into our financial capabilities as we are developing most software functionality on our own. As the development process continues the team has the flexibility to change technologies or design if a problem arises.

3.6 Development Process

The Agile framework will be leading the development process for our project. This is because its approach to achieve process by testing, developing, and reanalyzing the progress of every aspect of the project. This will be done in ‘Sprints’, which will help us create goals for each week of development for the project.

3.7 Design Plan

Describe a design plan with respect to use-cases within the context of requirements, modules in your design (dependency/concurrency of modules through a module diagram, interfaces, architectural overview), module constraints tied to requirements.

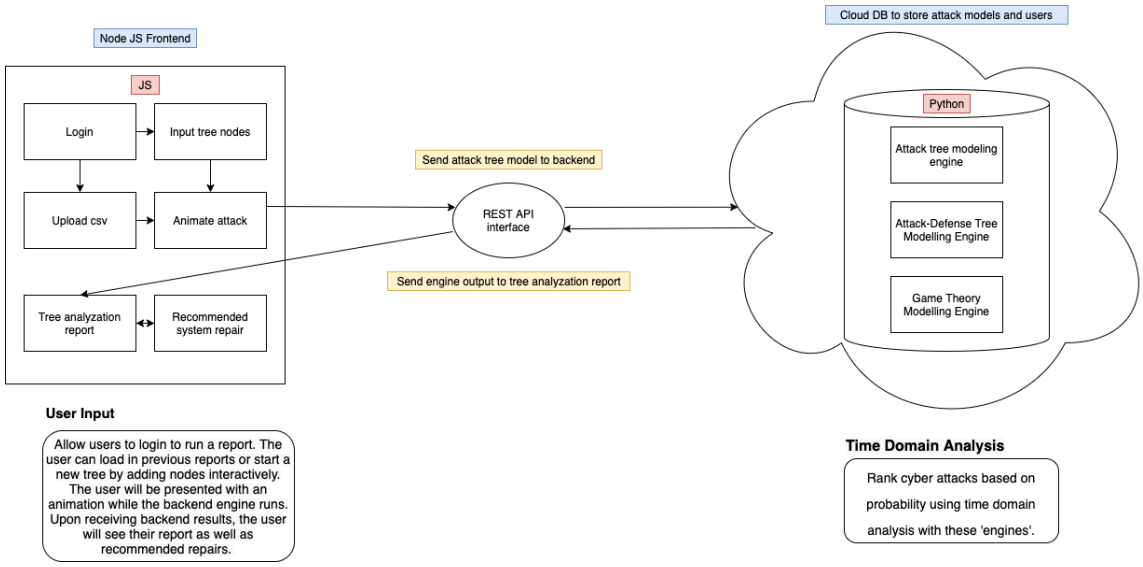

Figure 3.7.1: System Diagram

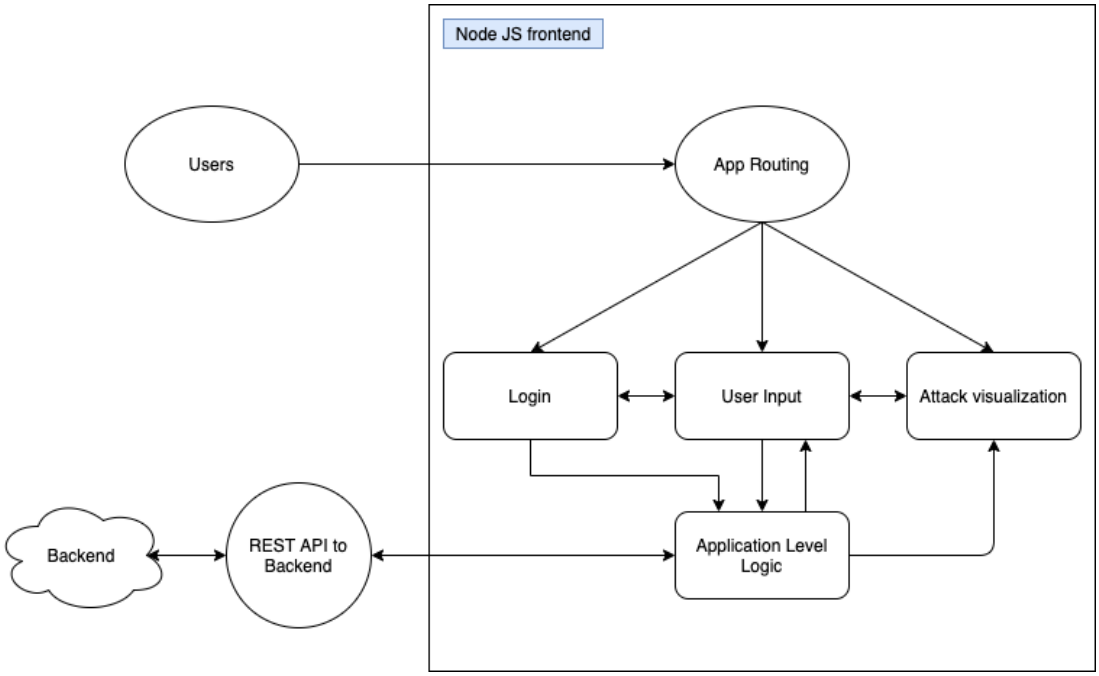

Figure 3.7.2: Frontend Diagram

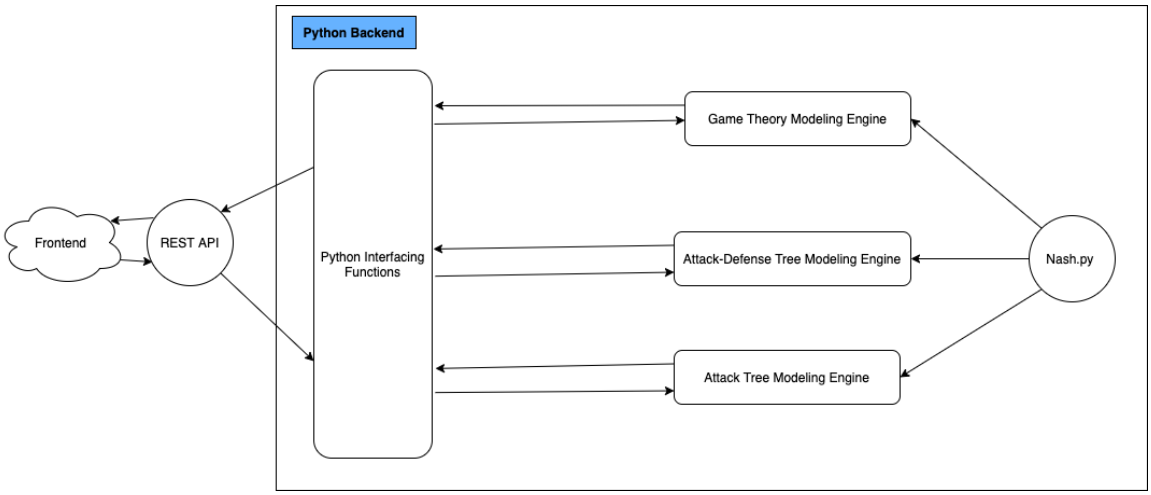

Figure 3.7.3: Backend Diagram

4 Testing

4.1 Unit Testing

The project will be unit tested on both the front-end and the backend. Front end testing will consist of React based-unit testing. This will be used to ensure that the user interface is functioning as expected and also ensure that the user interface is able to update with data sent and received to the API.

The backend will use python unittest framework to ensure that they are sending and receiving the correct information via the API. These unit tests will also be responsible for testing the following:

- Game theory algorithm to ensure certain inputs return a certain output.
- Database read, write, and delete functionality.
- Testing of all methods/functions related to the simulation tool, ensuring that specified input returns desired output.
- All methods/functions that are deemed critical.

Unit tests will be developed on an as needed basis, to be determined by the respective developers and the client. Unit tests will be used in conjunction with other forms of testing to ensure that the project is functioning properly.

4.2 Interface Testing

The main interfaces in our project are 1) between the front end application and the back end server, 2) between the back end server and the analysis engines, 3) between the back end server and the existing simulation tool, and 4) between the back end server and the database. To test these interfaces, we will use test cases representing normal use cases as well as scenarios that include edge cases or unexpected values. This will ensure that the interfaces correctly handle all cases that they are expected to handle and fail appropriately on unexpected or invalid inputs. Example test cases for each model are listed below; further test cases will be added as we progress in the project.

1) **Front end** - back end
    - **Valid network model**
    - **Invalid network model**
    - Valid network analysis
    - Invalid network analysis
2) **Back end** - analysis engine
    - **Valid network model**
    - **Invalid network model**
    - Valid network analysis
    - Invalid network analysis
3) **Back end** - simulation tool
    - **Valid network model**
    - **Invalid network model**
4) **Back end** - database
    - **Valid database information**
    - **Invalid database information**

4.3 Acceptance Testing

To test acceptability of the application we plan to use a Black Box testing approach, where the inner workings of the software are ignored (unit and integration testing will have already assured quality of inner modules) to focus on the functionality as a whole. A series of specific use cases designed with client input will be executed by the application. Expected output will have already been calculated by hand, and the app's output will be compared to its expected output. These use cases will be real-world scenarios designed in tandem by our team and our client's team; they will also be complex enough (include large data and many edge cases) that correct output from the app should guarantee functional performance. Additionally the software will undergo some beta testing performed by the client to assure generally usability is up to par.

4.4 Results

As our group is still in the design phase of our project, there is not much actual software for us to test yet. This section will be updated as this changes.