SYSC 4907 A

Proposal: A Web-based Platform for the Study and Analysis of COVID-19 Spread

Supervisor: Professor Gabriel Wainer

By: Tyler Mak (101108389) Andy Ngo (101132278)

Date: October 21, 2022

1. Project Requirements

This project will be about creating a web based application that is able to perform use-case specific simulations using an existing spatial-spread COVID-19 simulation model. We will mainly be focusing on optimizing the transmission of the simulation results as they are too large and take too much time to render the simulation on the frontend. Depending on time availability we may also work on implementing an interactive simulation.

1.1. Functional Requirements

Currently, the backend generates the simulation results and these results are transferred to the frontend as a whole. This would be fine if the result file is small but for larger sizes, we start to see performance deteriorate. In order to optimize these simulation results we plan to use web sockets to stream the results. Streaming the results allows the simulation to begin as soon as the first frame is received. It will also free up memory space on both the frontend and backend as the results no longer have to be stored.
The functional requirements for optimizing the transmission of simulation results will then be as follows:

1. Simulation results are parsed in the backend
2. The backend must be able to open web socket connection at the request of the front end
3. The frontend must be able to initiate the transfer of data through the socket connection
4. Backend must be able to stream simulation results to frontend through the socket connection
5. Frontend must be able to stop the socket transmission through the socket connection
6. Frontend must be able to receive and render simulation using streamed simulation results

In order to implement an interactive simulation, two way communication is necessary between the frontend user and the backend simulator. Currently, communication uses the REST architecture which in turn means the communication is one way. In order to implement interactive simulation, again we can utilize web sockets.
The functional requirements for interactive simulation will be as follows:

1. Frontend web platform and backend simulator establish a two way connection
2. Frontend user can pause and resume simulation at any moment during simulation
3. Frontend user can pause simulation to modify parameters whenever needed
4. Modified parameters are sent to backend and simulation results are recalculated
5. New simulation results are sent back to frontend and simulation is rerendered

1.2. Non Functional Requirements

The nonfunctional requirements for optimizing transmission of simulation results will be as follows:

1. Simulation results are sent in order to frontend
2. Frontend checks if data received makes sense (check for data corruption)
3. Connection is closed immediately after all data is sent through it
4. Web socket is compatible with Cadmium simulator
5. Simulation data transferred from backend must be compatible with Web Devs viewer
6. Entire system is able to handle a certain number of users

The nonfunctional requirements for interactive simulation will be as follows:

1. Both frontend and backend check for data corruption
2. Both frontend and backend sends acknowledgement back when properly received data
3. Both frontend and backend will retry sending data if they don't get acknowledgement from other side
4. Simulation resumes from exact position user is currently viewing, with newly modified parameters

1.3. Progress Measurements

To ensure progress is being made, certain unit and integration tests can be performed for each of the requirements and GitHub will be utilized to backlog all the work that has been done. Using the GitHub ticketing system we can track the progress of the work done as well as ensure that each update to the code is reviewed by the other person. Weekly debriefs will be held to have the opportunity to discuss the progress made each week to either Professor Wainer or Bruno. Further down the project timeline, we will also write progress reports and a final report.

2. Background

The current architecture of the system consists of a backend and a frontend. The backend uses a Cadmium simulator to compute all the simulations while utilizing information from data models. The frontend is a web-based application that uses the simulation results to render the simulation with the web devs viewer. Communication between the frontend and backend uses a REST architecture and is one way. When a model is chosen, such as the COVID-19 spatial-spread model, and the simulation is run in the backend, the simulation results are output in a log file. This log file contains all the necessary information for the frontend to visually display the simulation and can end up being a very large file in the gigabyte size. Currently this file is stored on the server and takes up memory resources. In the case of a large number of users all running a simulation, each user will have a new log file created on the server which in turn can pollute the server and cause a deterioration in server performance. In order to transfer the information from the log file to the frontend, the frontend downloads the entire log file and processes it only once the entire file has been downloaded. This causes a deterioration in the performance of the frontend and damages the user experience. Since the frontend is a web based application it may also cause memory issues as browsers are resource limited. Some attempts to fix this included minimizing the log file size and processing the file in chunks. These attempts were insufficient and a better solution is required.

The proposed solution is to implement web sockets such that the backend doesn't need to store the log file on the server and the frontend will no longer have to download the entire log file. With web sockets the simulation results from the backend can be parsed and streamed to the frontend, and as soon as the first frame is received the simulation can begin. This will also solve the memory issue as the full simulation results no longer have to be stored. With this solution we no longer have to worry about the file size of the simulation results.

Additionally, web sockets can be used to make the simulation interactive such that the user can pause and modify the simulation parameters at any moment. This problem may be difficult to implement as it will require more cases to handle during the simulation. The simulator is required to push the simulation results to the end user while it is simulating the model. To solve this, it requires two way communication through web sockets as currently one way communication is established using the REST architecture. With this solution we will be able to interact with the simulation.

3. Action Items

We will mainly be focusing on improving the already existing simulation through optimizing the method in which the results are transferred to the web application. Depending on time availability we may also work on changing the simulation to be interactive. To enable these the following will be done:

- Have backend parse the simulation results
- Build a web service that:
  - Opens a web socket on the frontend
  - Allows frontend to tell the backend to open a web socket, with a REST service
  - Connect frontend and backend web sockets
  - Streams parsed simulation results from backend to frontend
  - Allows frontend to stop transmission
- Have frontend understand and utilize the streamed simulation results
- Verification that project requirements are met through testing,
  - Testing socket connections (sending packets)
  - Testing frontend requests (opening backend web socket connection, closing web socket connection)
  - Checking for data corruption
  - Multiple user test
  - Stress test
  - Etc.

4. Relevance to Authors' Degree Programs

4.1. Tyler Mak - Computer Systems Engineering

In the Computer Systems Engineering program, information from courses such as SYSC3303 and SYSC4602 will be useful for this project. In SYSC3303 we used the Java programming language to create an elevator system that used datagram sockets, datagram packets and local ip address connections. In SYSC4602, we learned how a network is structured and the details of how data travels across the internet. Using this information, I can have a starting basis on how to use web sockets and use them to solve the problem at hand.

4.2. Andy Ngo - Computer Systems Engineering

The courses that were assigned for Computer Systems Engineering during previous years will be very useful to implement into this project. Many languages were taught during the years, for this project it will be focusing mainly on using Java and C++. The problem that will be focused on will be optimizing the simulation streaming, which will be done by using web sockets. Web sockets were taught in SYSC 3303, 4001 and 4602, giving a brief understanding of how to deal with the problem we have at hand.

5. Team Skill Set

Tyler and Andy have worked together as front-end developers, using the web-based application Webflow. The team also learned the basics for HTML, CSS, and Javascript to create simple websites. During this work period, another software that was used for the job was Miro, used to create wireframes for websites.

The team has experience with GitHub through the courses, SYSC 3010 and 3303, these were project courses that gave a better understanding of how to share work with a team, the GitHub ticketing system, and code version control.

Andy has previous experience working with Java for personal projects developing games and has done a little research on C++ for basic knowledge.

Tyler has C++ experience from his co-op, and learned how to properly produce software documentation. He had to analyze and comprehend code files from software in the Beta stage to add new features and fix bugs.

6. Methodologies

Agile development will be used for this project, and it can be branched off to more methods such as scrum and kanban. Scrum will be the main method that we will use.

The scrum methodology is when the progress is split into 2 week sprints. These sprints consist of assigning given tasks from the backlog to specific people and working on these tasks. After the sprint, the product owner, Bruno, will decide and direct which tasks are to be done or fixed for the next sprint. Scrum focuses more on completing increments of work and testing to ensure the work completed meets standards.

The kanban methodology is shown through visual progress. Solely through using a kanban board that displays the tasks, what is in progress, and what is completed. The tasks that are listed consist of most items from the backlog and this will be continuous work until the project is finished or more items are added to the backlog. Kanban focuses on reducing the time a project takes to complete and continually improving the flow of work.

Scrum will be applicable to the project as constant testing will be made to make sure that it is working properly. Since this project is being done while the team is registered in other courses, the shorter work cycles are more ideal. After implementation, the team will execute tests to identify any issues and decide what has to be changed and what else needs to be done for the next work cycle.

The team will be sharing progress using Github to collaborate and keep track of all code that each team member creates. Keeping the code updated on the project repository for every change is crucial, GitHub keeps a history of changes. This will allow the team to go back to old code if an issue occurs with the newest code. When the team has to give a progress update, releases can be prepared to organize the essentials when presenting. A scrum board will also be used to keep track of the team progress. This will be for the team to know what is due, in progress, and what has been completed.

7. Timeline

During the month of October, the team has meetings and gets familiar with the pre-existing simulation models, Cadmium, and web services. While learning about these, the Project Proposal is also in progress, due on October 21st, 2022. After the proposal is finished, the team will begin working on figuring out a solution to fix the optimization problem for the simulations through Java. This process will continue until the end of November.

By December if the tests were successful the team can begin debugging, and implement the Java websockets code into C++ to work with Cadmium. During this time, the team will begin practicing for the Oral Presentation and the Progress Report, due on December 9th, 2022.

For January the plan will be to practice for the Oral Presentation, make any necessary changes, and prepare for the Poster Fair. Ideally the simulation results are optimized at this point and will allow the team to begin implementing the interactive simulation through C++.

During February the deadline to request for software/hardware will be due. This month the focus will be to work on the interactive simulation and test the implementations. The Draft Final Report will be started in the middle of the month to have time to be properly revised.

The Draft Final Report will be due on March 14, 2023. While working on the Draft Final Report, the code will require debugging and more tests to make sure that it performs as it should.

The final month, April, the Final Report will be due April 12th, 2023. By this time the optimized results should be working properly, the interactive simulation is finished and works as well. It will be time to demonstrate the project and show that it is working as proposed.

Month	Deadlines / Tasks
October	- Proposal (Due October 21, 2022) - Get familiar with Cadmium and Simulation web services
November	- No specific deadlines - Begin optimizing the transfer of results - Test code
December	- Oral Presentation / Progress Report (Due December 9, 2022) - Debug code / implement Java code to C++
January	- Changes to Oral Presentation / Poster Fair - Present Oral Presentation - Finish implementing code on C++ - Begin implementing interactive simulation
February	- Deadline to request for software/hardware needs (Due February 14th, 2023) - Test implementations for interactive simulation
March	- Draft Final Report (Due March 14, 2023) - Debug code and perform more tests
April	- Final Report (Due April 12, 2023) - Finalize code and clean up repository

Table 1: Project Deadlines and Tasks

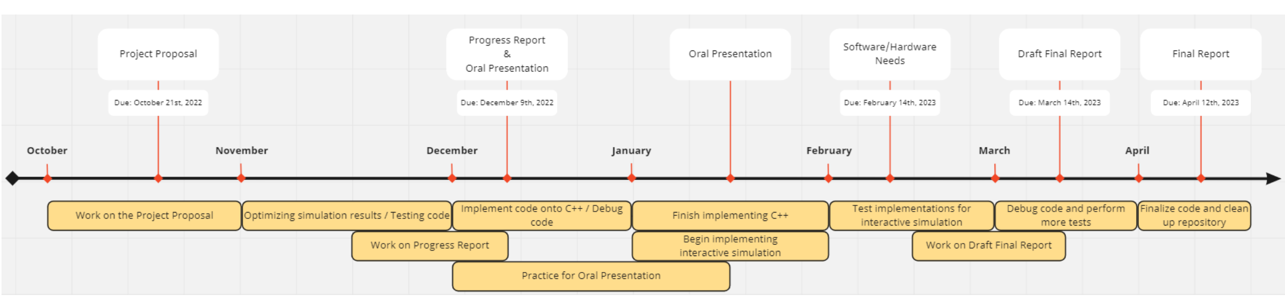

Figure 1: Project Timeline

8. Risks and Mitigation Strategies

Since this project is not a long term project, some of these risks and mitigations are hypothetical. The data that will be supplied for this project will not be confidential, making the risk relatively low. Few other possible risks that may arise during the project could be having the wrong web service to test our development and the implementation on Cadmium may be too difficult.

Probability	Impact				
	Trivial	Minor	Moderate	Major	Extreme
Rare	Low	Low	Low	Medium	Medium
Unlikely	Low	Low	Medium	Medium	Medium
Moderate	Low	Medium	Medium	Medium	High
Likely	Medium	Medium	Medium	High	High
Very Likely	Medium	Medium	High	High	High

Table 2: Risks and Mitigation Chart

8.1. Insecure Data

The data that will be used for this project will not be confidential, if given confidential data, it would create a much higher risk.

Risk Rating: Low (Moderate x Trivial) / High (Likely x Extreme)
Mitigation:

- Not using confidential data for the simulation models
- Make the data anonymous or ensure the data can be used publicly
- Improve security:
    - Ensure the web server information stays confidential only to those who were assigned
    - Encrypt data with high confidentiality that will only allow people with proper privileges to access the data

8.2. Wrong infrastructure (Web service)

When implementing the web service it can be done initially on our local machines. To properly test the development, a server provided by Compute Canada cloud will be used. But in the case that the infrastructure does not hold for the work period it may cause a few issues.

Risk Rating: Medium (Moderate x Minor)
Mitigation:

- Have backup infrastructure (AWS server) that will be accessible and can serve for the purpose of a web service. This will require the team to use the project budget to utilize the AWS backup server.

8.3. Cadmium cannot be modified easily

Cadmium DEVs tool will be used for the backend. A problem that might arise later on in the project could be that the solution derived may be difficult to apply through Cadmium.

Risk Rating: Medium (Moderate x Minor)
Mitigation:

- Create pseudocode of the interactive simulation and still try to apply the interactive simulation code into C++
- Learn more about DEVs and C++ to get a better understanding of the Cadmium simulator code
- Request for help from Bruno or the developer(s) of Cadmium if modifying is too complex

9. Required Components and Facilities

To optimize the simulation output this will be done through testing on any desired Java IDE on a web server. In the case where the web server is not working, the team might need to put money into an AWS server. After successful tests this will have to be implemented on the pre-existing web based architecture. The code will have to be applied onto the Cadmium simulator, the code will have to be implemented as C++ from Java.

After achieving optimized results, the next task will be to implement an interactive simulation. This will have to be implemented straight onto the Cadmium simulator, using any desired C++ IDE.

Authors’ Contribution to This Proposal

Author	Contributions to this Proposal
Tyler Mak	Project Requirements, Background, Action Items
Andy Ngo	Methodologies, Risks and Mitigation, Timeline, Required Components and Facilities

Table 3: Author Contributions