CA400 Functional Specification

Project Title: We live in a society
Student 1: Sean McCann (16448004)
Student 2: Kacper Slowikowski (16446386)
Date: 22/11/2019

1. Introduction

1.1 Overview

The system we're developing is a simulation application with user inputs. It closely resembles a game in many ways. The simulation will consist of a number of societies competing against each other for resources. These societies will also have the ability to trade and work together in order to improve their current status.

The user will control their own society as a leader. Their goal is the same as the others. However if the user chooses to go against their societies best interests the society will rebel and the simulation will continue without the user. In other words the user loses.

We believe that our application will serve as a fun and entertaining piece of media for any who are interested in game theory, genetic algorithms and advanced decision making algorithms and tactical strategic analysis. Seeing as all of the above are at the core of this particular project.

Within our project we will be utilizing a database. This database will consist of all the information about each society at different time intervals. This information will be utilized during decision making and also used as a point of reference while creating our algorithms. It will be initialized at the start of the game and the data will be discarded when the user closes the application.

1.2 Glossary

Criticality: Ranges from 1 to 5. 1 being most critical and top priority and 5 very low priority

2. General Description

2.1 Product / System Functions

Map

The simulation will be initialized with 2d map of tiles. Each tile will be assigned a terrain. Each terrain will provide bonuses and hindrances for the society. A society can use its resources to claim new tiles and spread its borders each turn. The map will utilize Pixel art (also known as Retro art in games) to identify separate tiles and the societies which control them.

The User

Each society will have a leader at its head. This single individual will be responsible for the decisions made by the society. This is also a way for the user to interact with the system. As a user you will control this individual. Each society will consist of its members, each member will have certain traits and opinions. As a user, your goal is to control as much land as possible by the end of the time interval while maintaining a happy society. If a society dislikes the decisions its leader makes and those decisions have a negative impact on said society as a whole, that society will rebel against their leader. If that society belongs to a user they will lose.

We want to captivate user attention to the simulation while not barraging them with abundant amount of needless information. This allowed us to reduce the decisions a society wants to make to simple functions. These functions are

Expands borders. A society wants to expand its borders either by the means of conquering other societies. Buying land off other societies or claiming unclaimed land.

Trade. A society sees that other societies near it have better resources than them and wish to also have those said resources.

War. A society feels either threatened or dislikes another and wants to battle said society

Form Alliance. A society decides that merging with another one will be beneficial for them and wishes to combine strengths to form one larger society**.**

Each turn the user will choose one of these decisions and follow observe the results unfolding in front of them.

Society breakdown

In our simulation a society is broken down into two sections. One is the average statistics of the society and their leader. The other being an account of each member within the society.

The average statistics of a society will be used to show the user which decisions will most benefit their society and which decisions the members of their society want carried out. We hope that this allows the user to understand the system better while not directly forcing them to choose a decision based on our heuristics.

The other part of the society is its members. Each member consists of traits, opinions and skills. All of these contribute to the overall performance of a society.

This is the list of the traits a person will have within a society, Health, Age, Gender, Physical Prowess index, Fertility index and Pathfinding Index. These traits will be represented by

indexes and will change throughout the course of the simulation. They will also be used when determining which members of the society will reproduce.

The list of opinions a society member will have is of variable length. It depends on how many societies are in the simulation and how many societies they have made contact with. Each opinion instance will be a floating point number from the range -1 to 1. Starting at 0 for each society other than their own.

In addition to these traits and opinions we will also assign each member of society an index for 3 major skills. Technology, agriculture and medicine. These skills will have an impact on the traits of the society members as a whole. However a society can only utilize those skills and gain more in those skills if they own a tile which corresponds to the skill needed.

2.2 User Characteristics and Objectives

The intended users for this application will be, computer scientists and students who have an interest in both genetic algorithms as well as advanced decision making algorithms as well as game theory enthusiasts who have an interest in examining interactions between complex organisms. In both of these cases the expected user experience with software systems will be moderate to advanced. Users will only be required to have a basic understanding of user interface interactions in order to fully explore and utilise this application.

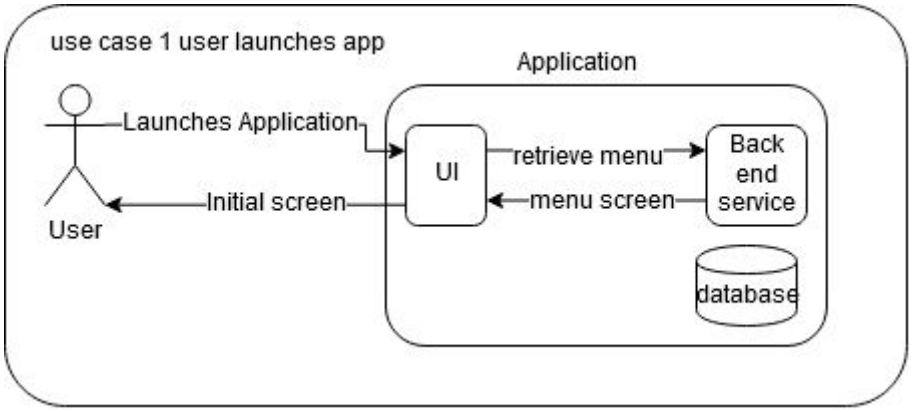
This project must achieve a variety of basic user needs such as an easy to understand and responsive GUI, an assortment of initialization options which will be provided to users in order to customise multiple factors in regards to their simulation and engaging real time feedback to the user which will provide all important updates and decisions as they occur.

Users will expect to also have a large amount of input to the system. Users should be allowed to make their own decisions in regards to their society in game. These decisions shall cause the simulation to adapt in realtime to the user’s actions. In turn this will provide more engaging feedback to the user. Users will also expect to be able to review important information and results at the end of each simulation.

2.3 Operational Scenarios

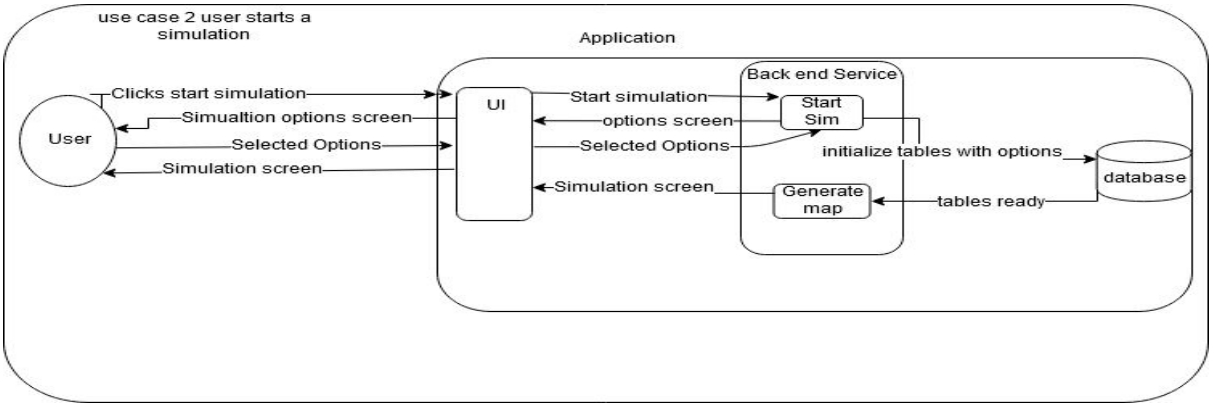
Use Case 1

As a user I want to start the application by using a shortcut on my desktop. This should launch all the necessary components of the application and present me with an easy to read main menu screen.



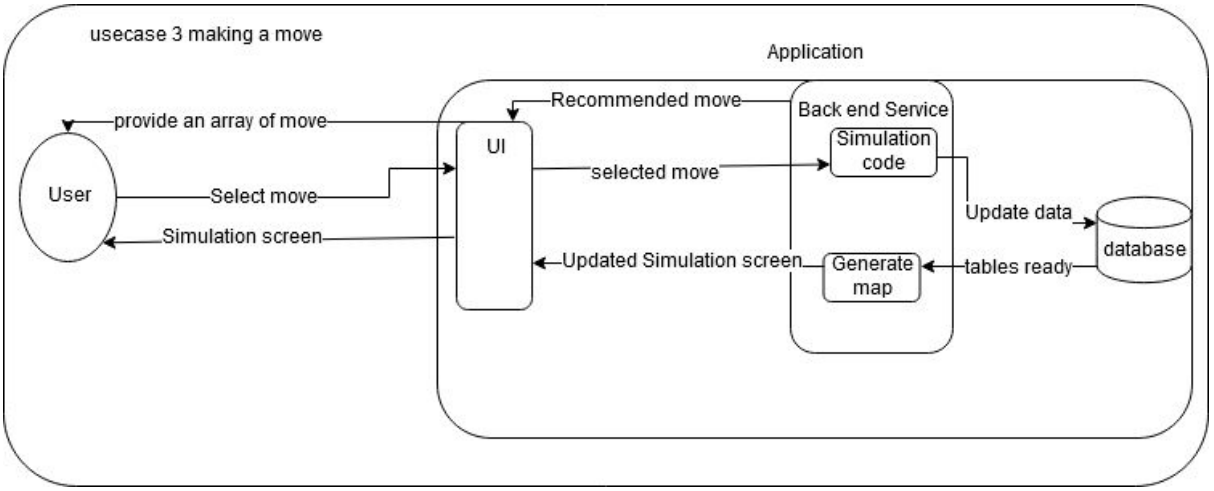
Use Case 2

As a user I want to start the simulation by clicking the start button on the main menu and selecting my societies name and starting population. I also want to select the size of the map and the average terrain quality along with the number of opponents i wish to go against. This should provide me with a generated map with my society clearly visible.



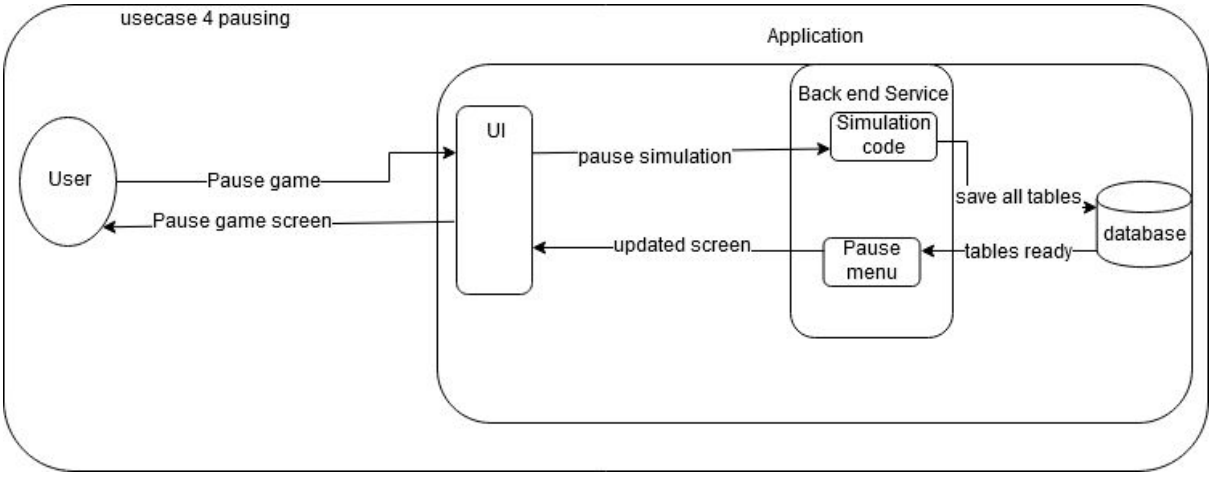
Use Case 3

As a user I want to be able to interact with the system by making choices for my society as a leader. I want these choices to be easy to understand and the results of those choices easy to see on the screen after I make them.



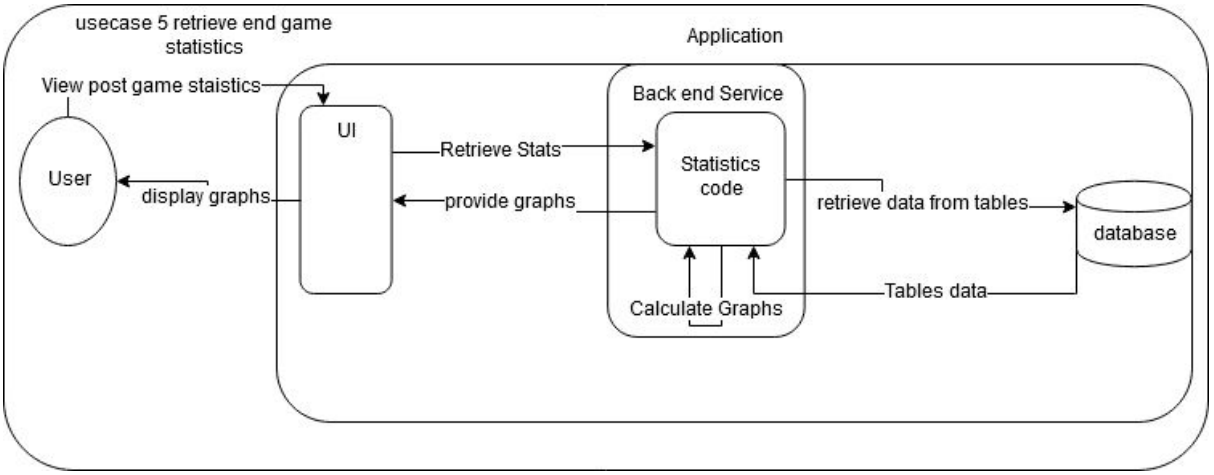
Use Case 4

As a user i wish to be able to pause the simulation at any time without causing a major system malfunction.



Use Case 5

As a user i wish to be able to review all of the societies statistics after a simulation ends. I wish to be able to save/export those statistics for future reference.



2.4 Constraints

Lists general constraints placed upon the design team, including speed requirements, industry protocols, hardware platforms, and so forth.

The application will have a multitude of constraints that apply to the front-end, database and back-end respectively.

Firstly constraints that apply to the front-end will be mainly speed and design based. The UI must provide enough information to the user in order to prove useful in terms of allowing the user to understand all relevant information at any given point of the simulation. Conversely. The UI should not overwhelm the user with an abundance of irrelevant information. All interactions that a user may take should be clearly visible and easy to understand and whenever a user makes a decision there should be a clear indication that this event has been recognised by the system. All decisions that both the user and system make should computationally efficient in terms of speed.

The database we will be using to store simulation data should be normalised properly as new data will be stored and removed regularly. Writing and reading data to and from the database should be efficient. As the simulation progresses more data will be written to the database and the scope of this data could potentially become quite large, it is crucial to handle this data thoughtfully.

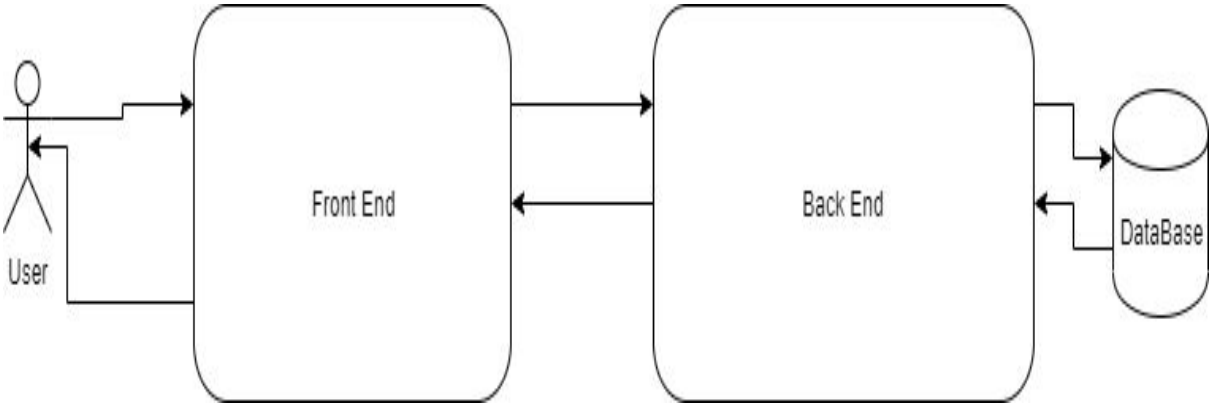
Finally the back-end for the system should be written efficiently so all genetic computations and decisions will impact the running speed of the application as little as possible

3. Functional Requirements

Description	**Criticality**	**Technical Issues**

The simulation should allow for multiple societies to exist at any one time within the simulation	**1**	This will heavily depend on the hardware used while developing and using the application. Memory is a finite resource within a pc and we assume that our application may require a lot of it.
An equal distribution of male to female characters has to exist within each society at startup to allow for reproduction	**1**	Not applicable
The map of the simulation should be as close to truly randomised as feasibly possible without creating automatic lose scenarios	**2**	Designing a map generation system which doesn't enclose societies and tiles within impassable water
User interaction should be available through both keyboard and mouse to allow people with limited hand mobility to still enjoy the simulation experience	**2**	Creating logical and easily understandable keybindings to each function in our system
Each turn within the simulation should give enough time for the user to observe all the changes occurred due to last decision	**2**	Must introduce some sort of a clock system for turn order making sure the other societies dont go ahead of the user
The user should be able to exit a simulation at any point without having to close the application.	**2**	Must assure that all currently active threads are closed properly and that the database is cleared and prepared for a new simulation.
The application should allow the user to save simulation results for future review.	**3**	Must be saved in an easily accessible directory and in a readable file format.
The application should display clear animation for each societal decision	**4**	Not applicable
This application should be launchable using a shortcut after installation	**5**	Converting the java files to an executable
The user should be able to launch a quick simulation without need of entering option parameters	**5**	Researching default statistics for a society using our own application
The simulation should work and look the same on most standard monitors and resolutions	**5**	This will require us to utilize scaling values while designing the UI so that everything will be readable.

4. System Architecture



The above diagram represents our system structure/architecture.

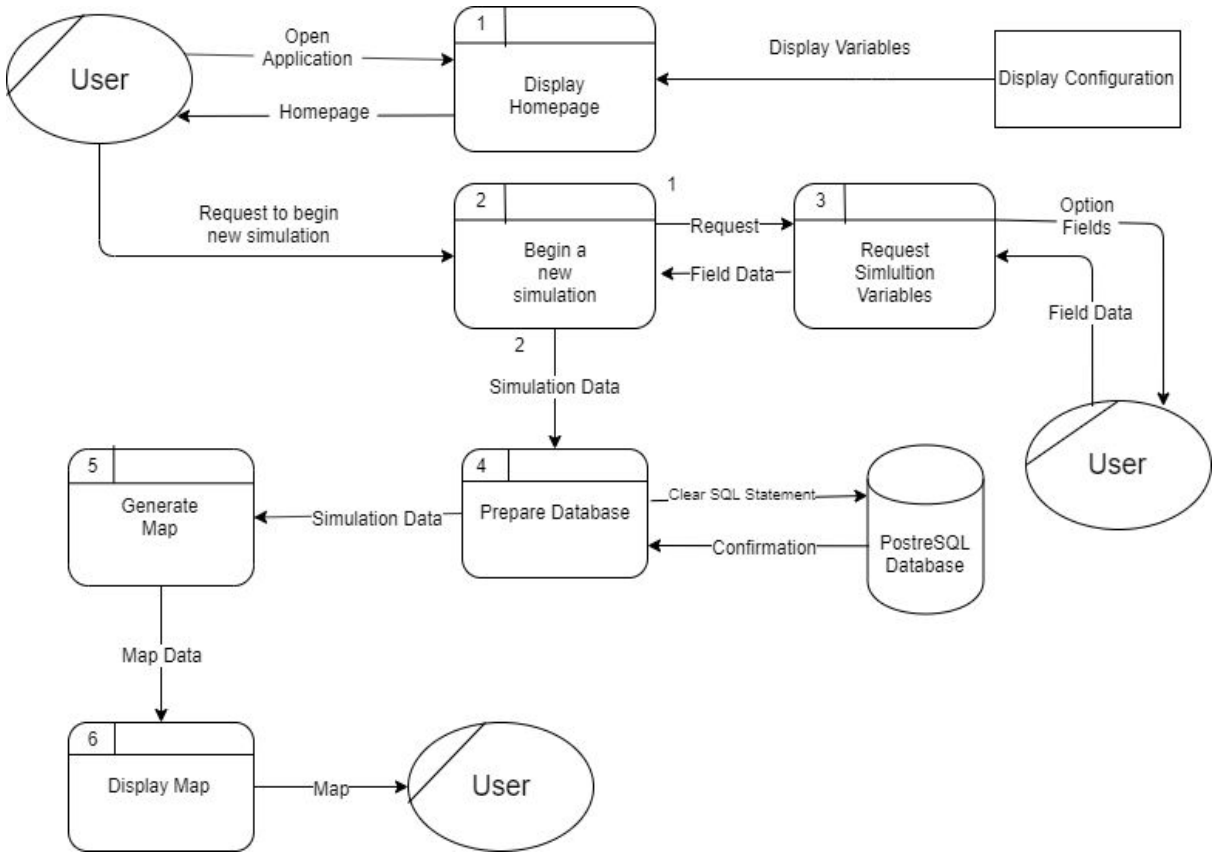
The **Front End** will consist of the UI and menu system that the user will be interacting with throughout the simulation. It will hold functions which retrieve and send data to the back end and the code for displaying the updated screen. This allows us to put all the heavy loaded computational work into our back end application.

Our **Back End** will hold our simulation code. This will include the decision making algorithm, genetic algorithm, map generation code and statistical analysis code. Separating these two conceptual systems will streamline our development and make the system easier to understand by both humans and machines alike.

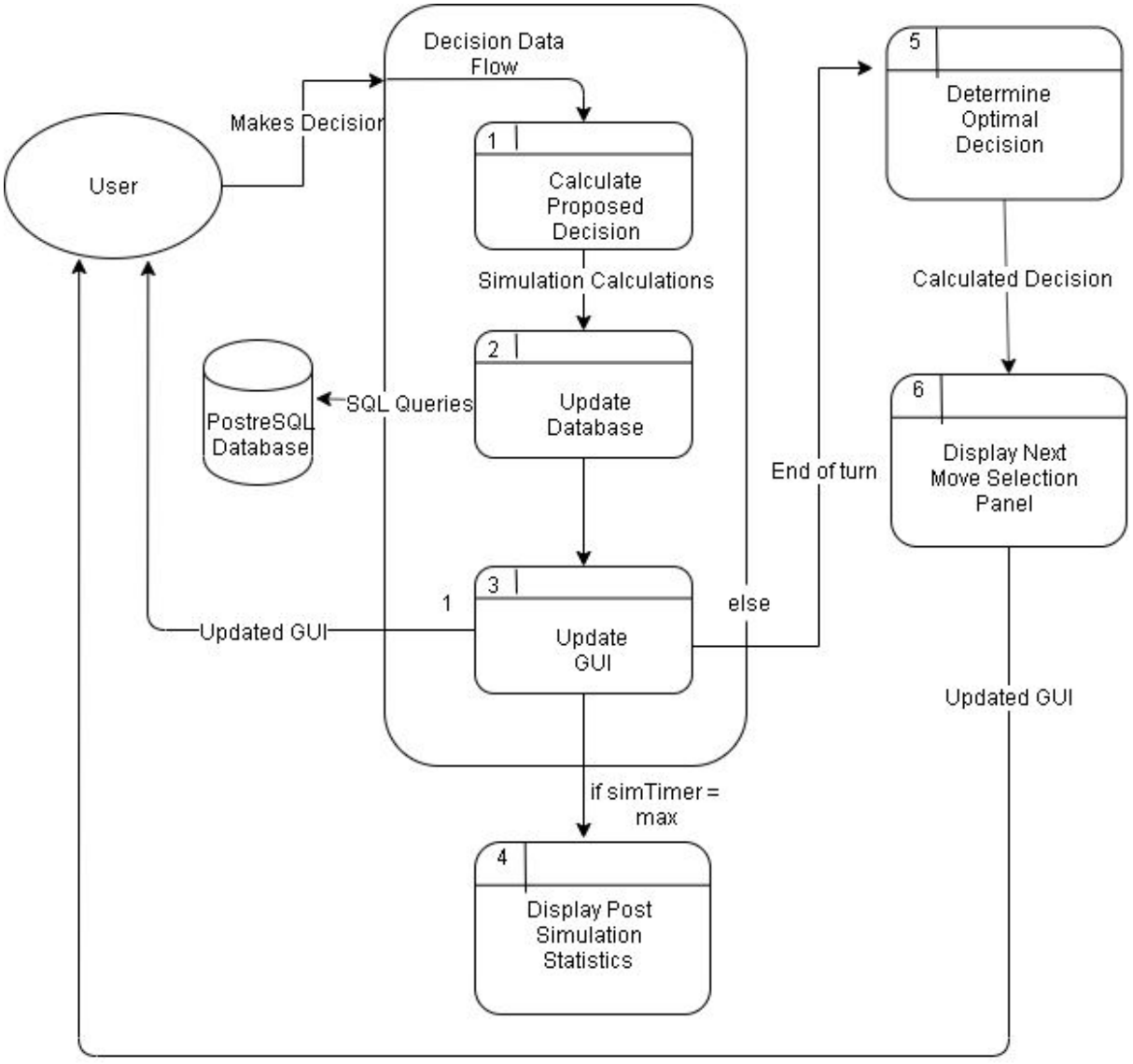
The **Database** part of our system will exist within the application itself but outside of the back and front end services. It will maintain data on the simulations which can be viewed by the user at the end of their simulation. It will be initialized upon startup of the simulation using the parameters provided by the user.

5. High-Level Design

Logical Data Flow of a User Launching the Application



High level decision cycle within the simulation



6. Preliminary Schedule

This section provides an initial version of the project plan, including the major tasks to be accomplished, their interdependencies, and their tentative start/stop dates. The plan also includes information on hardware, software, and wetware resource requirements. The project plan should be accompanied by one or more PERT or GANTT charts.

The initial version of our project plan will consist of a number of development phases. Each phase consists of a list of tasks which we will complete before moving onto the next phase. Initially we plan to try and implement the frontend(UI), database and map generational code before beginning to implement our algorithms. The reasoning for this is so we can achieve

full end to end communication for all future tasks. We will be following an SDLC development cycle for said tasks.

We will then follow an agile development process in which we create relevant tasks and assign to each member in order to achieve a goal. We will host daily stand-ups in order to assure that both members are on the same page. Once a week we will hold sprint retrospectives in order to review the previous week's sprint and address any issues within it. Once a week we will also hold a sprint review in which will address our current backlog and next upcoming sprint using the information from sprint retrospectives.

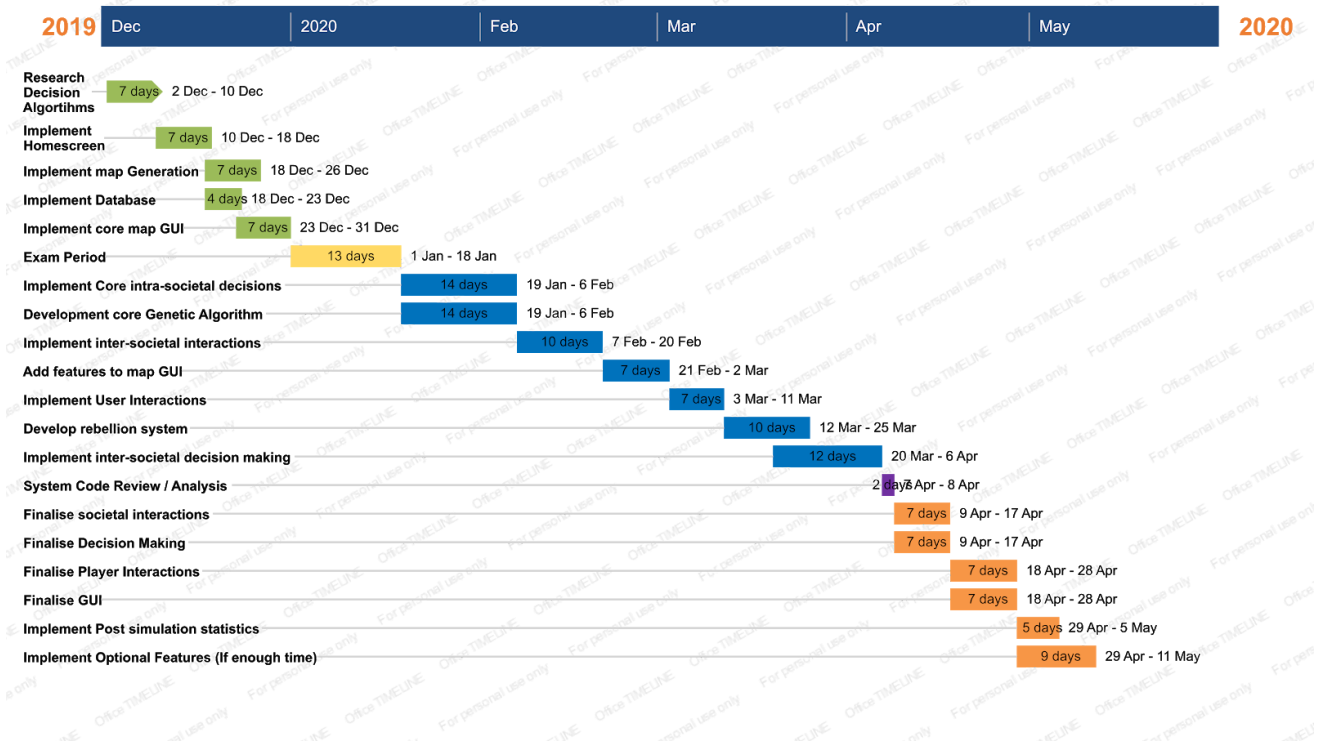
Our application will be written using Java and the Lightweight Java Game Library (LWGL). Our database will be written using PostgreSQL. Therefore the only hardware requirements is a functional desktop pc or laptop and the software requirements will be to have Java and PostgreSQL installed.

Unit testing will be utilised for each task in order to reduce any potential bugs or issues from appearing.

Halfway through our development process we plan on holding an application source code review and analysis on our entire application. The purpose of this is to ensure that all code is written to a high standard and to ensure that our simulation is performing to an acceptable standard. After this we will finalise all logical aspects of the application by both completing features and by fixing any concerns that may have arisen within the review.

Finally if we complete all initially proposed features we will discuss and plan future development for the application and try to implement new optional features if time allows

.



Appendices

[UI java library]
[Agile Diagrams]
[Sprint Board software]
[PostgreSQL Library]