1. Introduction:

To begin the requirement elicitation process, our group discussed the brief provided by the customer, to identify the main features of the product and the requirements associated with them. Once our team had identified the product's general aims and requirements, we met with the customer to clear up any ambiguity in the brief's requests and used the opportunity to ask questions we had about the requirements. Finally, our team had one more discussion amongst ourselves and created a paper prototype to finalise the requirements.

We presented the requirements in accordance with the IEEE Recommended Practice for Software Requirements Specifications (SRS) [1], which suggests that the document should clearly describe the functionality, design constraints, performance, external interfaces and other attributes of the product. The specification [1] also explains the characteristics of good requirements, for example, the requirements should be testable, unambiguous and traceable.
The format used for this SRS includes an overall description of the system, an outline of the external interface requirements, categorises the functional requirements by system feature and lists non-functional requirements.

The overall description identifies the main aims of the product, outlines the product users, explains conventions used within the requirements document and contains a description of the design and implementation constraints. The overall description also lists user documentation that will be provided with the product and the intended operating environment.

The sections about requirements are categorised into system features, external interface requirements and other requirements. The main section of the SRS outlines the system's functional requirements, which are separated by the system features they apply to. The requirements have been presented in this way because it allows the user to easily see an outline of the main system features, but also allows the development team to quickly identify which requirements relate to the features being implemented.

The external interface requirements are separated into User Interfaces and Hardware Interfaces, and the non-functional requirements of the system are listed towards to end of the document and are categorised by Performance, Safety and Security.

We used the IEEE Recommended Practice for Software Requirements Specifications, as it is a format that has been created and acknowledged by IEEE so it should be reliable. Also using a standard makes documents easier to read and understand.

2. Overall Description

2.1. **Purpose** - The goal of our project is to develop a turn-based strategy game based on the geography of the University of York campus.

2.2. **Document conventions** - University of York abbreviated to UoY.

2.3. **Intended audience** - This game is part of a university project and so it is being made for our team to enjoy, however, it is also being made with our customer in mind and may be used for promotion by The University of York Communications Office. This document will be used by our team as well as UoY professors. The game may be made open to the public in the future.

2.4. **Product scope** - The objective of our project is to provide an enjoyable turn-based strategy experience for our customer to use for marketing and selling. The game will feature local multiplayer and single-player to give players more options and also will be based off of the UoY campus to appeal to the relevant player base. The games will be fairly short (around 30 minutes) for convenience and the customer will need to be able to show around 2 minutes of the game to demonstrate its key features and gameplay.

2.5. **Product perspective** - The software being developed is a self-contained product and is not part of any larger system.

2.6. **User classes and characteristics** - Gamer: A typical user who will play the game and will have access to saving and loading.

2.7. **Operating environment** - The software will have to run on the University's computers on Windows 10 and it will not be played on any other system (unless released to the public).

2.8. **Design and Implementation Constraints** - Minimalistic design with appealing 3d isometric graphical style so that it runs smoothly on most hardware. The system has to be fully implemented, tested and evaluated by the 2nd May 2018.

2.9. **User Documentation** - All user documentation will be included within the interfaces of the game. These interfaces will be designed to be intuitive and easy to use.

3. External Interface Requirements

3.1. **User Interfaces**

3.1.1. **Tutorial** - A short tutorial will pop up automatically at the beginning of a game for players to understand the dynamics and mechanics of the game. After the tutorial window is closed, the "Help" button can be used to invoke the same tutorial anytime during the game.

3.1.2. **Error Messages** - All error messages will be displayed in a pop up in the middle of the screen. Most importantly, they must be user-friendly.

3.1.3. **Player-related Information** - Any information that would be helpful to the player during the gameplay will be displayed at the top of the screen. They may include the details such as the number of territories and units the player has in total etc.

3.2. **Hardware Interfaces**

3.2.1. **Game controls** - The entire game will be controlled using only a mouse and players will select their units, move them, and perform other actions, such as rotating and resizing the game view. *[Risk: ID. 4]*

4. System Features

4.1. **Spawn Units**

4.1.1. *Description*

This feature assigns units to the player based on the number of sectors they own. *[Priority - High]*

4.1.2. *Functional Requirements*

REQ-1.1: The user should be able to spawn in units equal to the sectors they own plus any bonus units for capturing all of sectors in a college.

REQ-1.2: The user should be able to choose how many of each unit is placed in each sector.

REQ-1.3: The user should not be able to move on until all units are placed in the user's sectors.

REQ-1.4: The system should provide special units to players that capture all the sectors of a college.

4.2. **Basic Unit Combat**

4.2.1. *Description*

This component allows users to attack adjacent sectors. Each sector can only attack once per turn and must have more than 1 unit. *[Priority - High]*

4.2.2. *Functional Requirements*

REQ-2.1: When a sector is selected the system should show the available sectors it can attack.

REQ-2.2: When a sector is selected to be attacked it should take the attack value for the attacking sector and the defence of the defending sector into account to create the end outcome.

REQ-2.3: The combat system shall not be entirely random, but will include a skill-based slider mini game that will alter the player's attack/defence scores.

4.3. **Slider Mini Game**

4.3.1. *Description*

This is an addition to the combat system that adds a slider mini game. Each player involved in the combat will play a slider game where they will try to click when a marker passes a target. The closer to the target the higher the damage multiplier. *[Priority - Medium]*

4.3.2. *Function Requirements*

REQ-3.1: The system will display a mini game when a player declares an attack, first for the attacker and then for the defender.

REQ-3.2: The mini game will obtain a multiplier value based how close the users are to the target; the closer to the target, the higher the multiplier.

4.4. **Unit Movement**

4.4.1. *Description*

This function allows user to move their units between sectors as long as they are adjacent and both owned by the player. *[Priority - High]*

4.4.2. *Functional Requirements*

REQ-4.1: The system shall show the user the available territories to move to.

REQ-4.2: The user should be able to split up their units and send some to separate sectors.

REQ-4.3: The user should be told which units they have not moved yet and also be able to end their turn without moving them.

4.5. **Map**

4.5.1. *Description*

The map is the play-area for the game, set on the UoY campus. The map will be split into sectors which can be captured by the game's users. *[Priority - High]*

4.5.2. *Functional Requirements*

REQ-5.1:The system shall display the UoY campus map, separated into a number of sectors.

REQ-5.2:The map should be separated so that at least 75% of the sectors contain University buildings.

4.6. **Graphical User Interface**

4.6.1. *Description*

The graphical user interface is the front end of the system. It will be used to display relevant information to the users and allow them to play the game. *[Priority - High]*

4.6.2. Functional Requirements

REQ-6.1: The system shall provide a graphical user interface. *[Risk: ID. 5]*

REQ-6.2: The graphical user interface shall display a UoY map, along with a representation of the sectors, sector ownership and number of gang members in each sector.

REQ-6.3: The graphical user interface should provide a mechanism to pause and save the current game state as well as load previously saved game states.

4.7. **Players**

4.7.1. *Description*

Players are predominantly controlled by the users, and each player can own sectors and gang members. Players can command their gang members and capture new sectors. *[Priority - High]*

4.7.2. *Functional Requirements*

REQ-7.1: The system shall contain at least three players, one of which is a neutral player that does not attack sectors.

REQ-7.2: The system shall allow players to take turns. The turns shall alternate players, and should do so in the same order each round.

REQ-7.3: The system shall allow players to move gang members between their owned sectors.

REQ-7.4: The system shall allow players to move gang members to opposition sectors if there is a clear route up to two player owned sectors away.

REQ-7.5: The system shall provide three types of player; human players, passive computer players and aggressive computer players. Passive computer players will only defend, and aggressive computer players attack and defend their sectors.

REQ-7.6: At the start of a game, the system shall assign five sectors to each player (human and aggressive computer) and assign the remainder to the passive computer player.

REQ-7.7: The system shall allow players the option to do nothing on their turn, passing their turn onto the next player.

4.8. Pro-Vice Chancellor (VC)

4.8.1. *Description*

The VC is a side objective that will spawn in one of the territories. When a player moves units into a territory containing the VC, it will start a minigame where winning it will give you the VC as a unit. *[Priority - Low]*

4.8.2. *Functional Requirements*

REQ-8.1: The VC needs to spawn in a random sector at the start of the game.

REQ-8.2: Once a player moves into the sector with the VC, the mini game needs to start.

REQ-8.3: If the VC is in a sector it needs to be able to attack twice.

5. Non-Functional Requirements

5.1. Performance Requirements

The game must run at a reasonable speed for the user, therefore, we require that the game must be able to run at a speed of 30fps on the university computers. *[Risk: ID. 15]*

5.2. Safety Requirements

There are no safety requirements for the software itself, the equipment required to play the game could potentially cause some safety issues. Users continuously playing the game without suitable breaks or in poor lighting, could potentially be straining their eyes. Therefore, the game will include a message to encourage users to take breaks between games. *[Risk: ID. 3]*

5.3. Security Requirements

There are not security requirements linked to this project as the user will not be logging into the game. This means that we will not be storing any personal details connected to the player.

6. Other Requirements

ORQ-1.1: A single game must last for at least 20 minutes. *[Risk: ID. 16]*

ORQ-1.2: Within two minutes, users should be able to move gang members, attack an opponent and have at least two turns each.

Bibliography:

[1] IEEE Computer Society Software Engineering Standards Committee, *"IEEE Recommended Practice for Software Requirements Specifications."* IEEE Std 830-1998, 1998.