Requirements Elicitation and Negotiation

The project brief provided a high-level overview of some of the requirements that would need to be included in our research. In the first session as a group we read through this brief together to give everyone a solid understanding of what was being asked of us, while making notes of the high-level requirements that were mentioned - such as it being possible to capture another college via combat. During this process we developed a series of questions which needed to be clarified to ensure we were better informed about what was desired by the stakeholders. The primary stakeholders are the customer, University of York Communications Office and the players of the game (our SEPR cohort as well as prospective students). Building on this, we decided on methods which would allow us to further elicit the requirements for the game. Discussion as a group led to the decision of 3 more methods for elicitation:

- Meetings with the customer - This method of requirements elicitation would give the customer a chance to feedback on our ideas as a group meaning that if they had any disagreements with our proposal we could discuss and come to a conclusive idea. As well as this, we used the opportunity of meeting our customer to ask our questions about the brief in order to elicit more information of what was actually wanted by him. We are designing a game for the customer so it needs to be what they desire - rather than what we want in a game. Despite this we can suggest ideas to him and explain to him in meetings why we think certain gameplay choices should be made. In reflection, this method of requirements elicitation was an effective method - although we found more efficient results through emailing the customer due to his fast and detailed replies.

- User survey - One of the other main stakeholders of the game would be the players of the game. Therefore in order to gain an overview of what they would like in a game, we designed a survey which provided us with useful information regarding the requirements of the users. Using a survey is a fast way to acquire input from the players to see what they want in a game. Using this method allowed us to make better decisions as we had input from players - rather than just our group of 6 making a decision. Decisions for the game design were then easier to make due to the players influence in the survey meaning that we could design requirements based on what they wanted. For example, 82.4% of the responses agreed that the idea of resources in order to sail between places was a good idea. This is advantageous as realising what the players wanted in a later stage of the project, as explained by Dr Gary Morgan of ETAS Ltd, would increase the cost of fixing the problem 10 fold per stage.

- Prototyping - Prototyping is a good method to give a visual representation of what our vision is in order to show our ideas to the customer as well as to give us better understanding of how it would work and what extra things would be needed. By creating prototypes [1] of the combat system as well as the map and mechanics, we were able to more easily express our ideas to the customer during our meeting. Doing this meant that he could fully understand how our ideas would come together in the game so that he could discuss with us any disparities between his and our ideas. As well as this, the prototypes made the elicitation of requirements easier for us as we could see how the game would come together and what would be need for it to run smoothly.

Presentation of Requirements

In order to fully develop our list of requirements we took inspiration from the IEEE format for gathering requirements [2]. The functional requirements table provided, on page 18, was helpful in crafting our own table. We decided not to include the ‘Priority’ column as it is implied in the wording of the document - words such as ‘must’ and ‘could’ for example. The ‘Description’ section from the IEEE format was renamed ‘Requirements’. We added a ‘Notes’ column which included environmental assumptions, for clarity and detail; risks, as it is important to acknowledge times where our system may require more action and areas with a potential to stall our production; alternatives, where we attempted to offer solutions in case our original plan became unattainable, and fit criteria, which we felt was needed for some requirements to justify why they have been added. Perhaps our most important section was the ‘Extends’ column in system requirements which links the system requirement to the relevant user requirement, there is usually a many-to-one mapping with this.

User Requirements		
ID	**Requirement**	**Notes**
Functional - things a system must do		
Transformation: *Required response to a condition/event*		
A1.1	If the player defeats a ship, they should be rewarded	Risk: The player is rewarded disproportionately to the cost of battle
A1.2	The player should have encounters when they move in sailing mode	
A1.3	If the user travels, there should be a reasonable cost	Alternative: Encounters are difficult enough to be costly to the user
Invariant: *Property that must always hold*		
A2.1	The player should be able to spend resources	Assumption: The player has sufficient gold
A2.2	The player should have a main objective	Risk: Objective is not achievable
A2.3	The game must have a minigame	
A2.4	There must always be a way to lose the game	Assumption: The player does not want to lose the game
A2.5	The player should be able to attack land bound objects like colleges or departments	
A2.6	The game map must have at least 5 colleges, 3 departments	
A2.7	The player should earn points	Risk: Unbalanced point earning
A2.8	The game should include a sailing and combat mode	
A2.9	The game must be themed on the University of York campus	Fit Criteria: All names should be from the University of York
Failures: *Forbidden/permissible transformations*		
A3.1	The player should not have the final objective available to them immediately	Assumption: The player is starting a new game

Non-functional Qualities a system must have		
A4.1	The game must be enjoyable for the user	Fit Criteria: Create an enjoyment survey and give it to a number of new players
A4.2	A game session should last a reasonable amount of time	Assumption: The player is not very experienced with this game Fit Criteria: Time new player’s game time and confirm it does not exceed 30 minutes or fall under 5 minutes (assuming they don’t fail)
A4.3	Game must be intuitive to use	Assumption: The player does not play games frequently Fit Criteria: When observing, record any situations where the user accidentally does the wrong action or is confused
A4.4	The game must be themed on the University of York campus	Fit Criteria: All names should be from the University of York

System Requirements			
ID	**Extends**	**Requirement**	**Notes**
Functional Requirements			
Transformation *Required response to a condition/event*			
B1.1	A4.3	The game should have a user manual to help with play	Alternative: The game guides the player as they play
B1.2	A1.2 A2.8 A1.3 A2.5	Deciding to engage in combat must result in switching to combat mode	Alternative: Combat is optional (with some kind of cost) Risk: Encounters are too easy or hard
B1.3	A1.3 A2.1 A4.2	The player moving should use resources	Alternative: Encounters are challenging enough to be a cost
B1.4	A2.5 A2.6	Colleges and departments should be given a location on the map when the game starts	Risk: Locations are unbalanced or impractical
B1.5	A2.4	The system must end the game when the users health is 0	Assumption: No effects exist that prevent the player from dying

			Alternative: The player respawns and the game continues
B1.6	A2.1	The system must display contents of the shop and allow purchase of items when shopping	Assumption: The shop contains unowned items
B1.7	A1.1 A2.7	Successful encounters should give the player points and gold	Risk: Unbalanced point and gold income
B1.8	A2.4	The system must end the game when the user doesn't have enough resources to travel.	Assumption: The player cannot obtain resources without travelling
Invariant *Property that must always hold*			
B2.1	A1.1 A2.1	The game should include a virtual currency (gold)	Alternative: Some other currency or using points as currency
B2.2	A4.1 A2.2 A3.1 A4.2	The system must block access to a certain location which contains the final boss challenge	Risk: The area does not unlock once the criteria have been completed Alternative: Final objective is spread across the existing map
B2.3	A2.3	A minigame should exist that rewards the player	Risk: Unbalanced rewards Alternative: Minigame does not influence gameplay and is just for entertainment
B2.4	A2.8 A4.3	The game should always display ship statistics such as health.	Assumption: The player is in the game and not at the main menu
Failures *Forbidden/permissible transformations*			
B3.1	A2.1 A4.3	The player should not be able to spend more money/resources than they have	Alternative: Debt system is put in place
Non-functional requirements			
B4.1	A4.1	Responsive controls	Fit Criteria: No more than 0.5 seconds between an input and something starting to happen
B4.2	A4.1 A4.3	Intuitive user interface	Fit Criteria: User is able to play the game without accidentally making the wrong move
B4.3	A4.4	The game must be appropriate for students to play on open days, so not containing any offensive material	Fit Criteria: Contains no visible violence and no negative reference to staff or the university

Constraint Requirements - Global issues that shape the reqs (split into project/process/design)

ID	**Requirement**	**Notes**
C1	The game must be completed and delivered by 1/5/2019 and has milestones at 21/1/2019 and 18/2/2019	
C2	The game must not crash during any more than 1 in every 15 instances.	
C3	The game must run on Windows	Alternative: Also runs on Mac or Linux Assumption: University computers run Linux and Windows and most students have a Mac or Windows laptop
C4	The game must not contain copyrighted material	