Requirements2

Cohort 1, Team 1:

Connor Blenkinsop
David Luncan
Emily Webb
Muhidin Muhidin(MO)
Sam Laljee
Sooyeon Lee

a) Introduction to requirements

Single Statement of Need:

ENG1 Product Brief 2021 - York Pirates! pdf available here, though fairly detailed, is a great SSON as it is written by the customer, detailing exactly what they are looking for in plain text.

We held a meeting within our group to discuss the brief and the initial ideas, and what questions we had. The questions were separated into different sections such as general ones about which platform the game should run on, or specific ones about the details of combat. This list was refined to:

- Remove duplicates
- Remove questions that were already answered in the brief
- Give certain questions priority over others
    - If, for example, we asked if combat should be real time or turn based, questions about specific implementations of an unnecessary system could be ignored

With this list of questions we booked a meeting with our client and presented them to him. We took notes of the answers, and followed up with any additional queries that were raised from the meeting.

Research and implementation of methodology

The primary source for our method in eliciting the requirements was the lecture and information provided by the university. The system presented gave us an effective way of distinguishing between the types and levels of requirements, and also an effective way of displaying the requirements as a set of tables. This gives us three different forms of requirements:

- User requirements
    - The broadest and highest level requirements. These define general requirements that the client has about the game without necessarily specifying functionality
- System Requirements
    - Contain more detail about how those user requirements are to be implemented.
- Non-functional requirements
    - Features that need to be added for non-functional reasons – the product will work without them, but they might improve the games accessibility for example.

This method also calls for a priority rating for the user requirements which was divided into:

- Shall
    - Deemed necessary by the client for the game to be considered complete
- Should
    - Not strictly necessary for the game to be considered complete, but still desirable to be included
- May
    - Ideas that are not necessary, introduce the potential for additional workload, and are likely not going to be included. But they have not been explicitly ruled out

b) The following requirements are a selection that we elicited. The full list is available on the website at the following link:https://eng1team1.github.io/assessment1/team1-deliverables/Req2-Full.pdf

This selection is based on a number of factors, of which including but not limited to relevance to Assessment 2, dependency to successfully completing the brief and particular risks that may be discussed in regards to the particular requirement.

User Requirements

ID	Description	Priority
UR_FIRST_TIME_PLAYER	The game shall be easily understood by a player who has not seen the game before	Shall
UR_USABILITY	The application must be playable on a department pc, even without network access	Shall
UR_GAME_LENGTH	The game shall be playable in a single session	Shall
UR_COLLEGE_FIGHT_BACK	Colleges should be able to fight back in some seemingly intelligent way	Shall
UR_GAME_CURRENCY	The game shall have gold which the player can spend, and XP to level up	Shall
UR_POWERUP	Powerups shall be available to collect during the game	Shall

Functional

ID	Description	User Requirements
FR_SHOPPING	There should be ways to spend accumulated plunder	UR_GAME_CURRENCY
FR_POINT_EARNING_WEATHER	More points should be awarded when navigating bad weather	UR_NEW_OBSTACLES
FR_GAME_DIFFICULTY	There will be options to change how challenging the game is - these will impact speed of movement of enemy ship AND/OR rate of fire of enemies	UR_FIRST_TIME_PLAYER

Non-Functional

ID	Description	User Requirements	Fit Criteria
NFR_SCREEN_SIZE	The game shall make good use of screen real estate on most screen sizes.	UR_SCALABLE	Adjust for screen sizes between 13 and 27 inches.
NFR_HARDWARE_REQUIREMENTS	The game shall not have high hardware requirements.	UR_SCALABLE	Runs well without issues on a department laptop.
NFR_CRASHES	The game should not crash.	UR_RELIABILITY	Played 10 times while engaging with all systems without it crashing.
NFR_EPILEPSY	The game shall not use any forms of flashing images or repeated patterns that are likely to cause seizures.	UR_EPILEPSY	Avoids common epilepsy triggers[1]
NFR_SAVING	The application will be able to save/load one saved state	UR_SAVE_STATE	State of the game in main screen can be paused at any time, from a sample of 15 sves

- UR_USABILITY
    - This requirement has an assumption that each of these platforms specified are up to date and work on the version of the engine we look to use. There is the risk that, for example, some versions of MacOS are incompatible while others are.
- UR_FIRST_TIME_PLAYER
    - This requires some assumptions about what would be unclear to a first time player, and also some assumptions about what a first time player would intuitively understand. Some might not even get the premise while others might not necessarily need to be told anything. From our perspective, if we have added enough tutorial and information to explain the concepts on each launch this requirement will be met.
- UR_CHILD_APPRORIATE
    - This has some environmental context, as what is considered child friendly might not be shared by everyone. As per NFR_CHILD_APPROPRIATE, we have chosen to design our game around the PEGI standard, and this requirement can be deemed complete if the game conforms to PEGI 7[2]
- UR_COLLEGE_FIGHT_BACK and UR_AI_PIRATE
    - This requirement uses the terminology “seems intelligent”, our group understood this to mean that the colleges and other AI powered entities have some potential to aim for the player, and are able to hit them enough to pose a threat.
- NFR_CRASHES
    - The fit criteria is relatively arbitrary here, and there is always the possibility that a crash is in the code but it remains uncaught by our group. Therefore we have to set a point that means we test the game enough to eliminate these issues, but also means we are not pointlessly testing.
- NFR_EPILEPSY
    - The clear risk with this requirement is that we fail, and there is an epilitic trigger in the game. We therefore have to be especially sure that this is met by avoiding rapid sprite changes for example.
- FR_GAME_DIFFICULTY
    - Risks making the game unenjoyable for some users. To avoid, default level of difficulty should be low, but it should be easy to increase the difficulty after
- NFR_SAVING
    - The fit criteria here is rather arbitrary also, and with time being a continuous measure - a harder one to test as such and will never be completely exhaustive
- UR_SHIP_COMBAT, UR_NEW_OBSTACLES, UR_POWERUP,UR_SAVE_STATE
    - These are all new and substantial additions, failing to implement these will result in a loss of marks and also knock-on effect on other deliverables such as testing.

References:

1. http://gameaccessibilityguidelines.com/avoid-flickering-images-and-repetitive-patterns/

2. https://pegi.info/index.php/what-do-the-labels-mean