Requirements (Full)

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)

User Requirements

ID	Description	Priority
UR_USABILITY	The application must be playable on a department pc, even without network access	Shall
UR_FIRST_TIME_PLAYER	The game shall be easily understood by a player who has not seen the game before	Shall
UR_ACCESSIBLE_COLOUR	The game shall not convey key information through colour alone in order to accommodate colour blind players.	Should
UR_RELIABILITY	The game shall be reliable.	Shall
UR_EPILEPSY	The game shall be safe for players with epilepsy.	Shall
UR_CHILD_APPROPRIATE	The game shall be appropriate for children to play.	Shall
UR_EARN_GOLD_XP	The player shall be able to earn gold.	Shall
UR_EARN_POINTS	The player shall be able to earn points	Shall
UR_SINGLE_WORLD	The game shall exist only in a single connected world map	Shall
UR_COMBAT	The game’s combat shall occur in the main world and in real time	Shall
UR_COLLEGE_COMBAT	The player shall be able to fight opposing colleges	Shall
UR_SHIP_COMBAT	Player can engage in ship to ship combat	Shall
UR_CAM_LOCATION	The camera shall be in a top down perspective	Shall
UR_COLOUR_SCHEME	The game should include bright colours and large graphics to be visible at a distance	Should
UR_ALLY	You may be able to ally with different colleges	May
UR_TIMING	The game shall be playable in a single session	Shall
UR_WORLD_SCHEME	The world should avoid randomisation for world and objectives	Shall
UR_MOVE_SCHEME	The player shall have control over movement in the map	Shall
UR_COLLEGE_FIGHT_BACK	Colleges should be able to fight back in some seemingly intelligent way	Should
UR_AI_PIRATE	Pirates will have some level of AI to make actions	Shall
UR_COLLEGES_QUANTITY	There shall be at least 3 colleges featured in the game	Shall
UR_OBJECTIVES_STAGE	The game shall contain multiple stages of objectives	Shall
UR_NEW_OBSTACLES	Bad weather and other obstacles should be added	Shall

UR_OBJECTIVES_UI	The game shall clearly display the current objective on the UI	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
UR_SAVE_STATE	Player must be able to save the state of the game and then resume it later	Shall

Functional

ID	Description	User Requirements
FR_TEXT_LABELING	The game shall include text labels of important game objects (colleges, enemies, etc) in place of just colour	UR_ACCESSIBLE_COLOUR
FR_GOLD_COLLEGE_DEFEATED	Players shall earn gold for defeating a college.	UR_EARN_GOLD_XP
FR_SHIP_FIRE	Players should be able to fire from their ship and use these projectiles to damage opponents	UR_COMBAT
FR_NPC_COMBAT	NPC ships can now engage in combat and be destroyed/destroy	UR_SHIP_COMBAT, UR_COMBAT
FR_COLLEGE_AI	The college should have some simple AI to decide how to fight back when attacked	UR_COLLEGE_FIGHT_BACK
FR_CAM_POS	The camera should be located in some position above the game world, likely in an isometric/top down position	UR_CAM_LOCATION
FR_XP_GOLD_TRACKING	Gold shall be separately tracked from XP	UR_GAME_CURRENCY
FR_XP_GOLD_EARNING	Gold and XP should be earned by completing objectives (destroying collages)	UR_EARN_GOLD_XP
FR_POINT_EARNING	Points should be earned by completing objectives and moving	UR_EARN_POINTS
FR_POINT_EARNING_WEATHER	More points should be awarded when navigating bad weather	UR_NEW_OBSTACLES
FR_BAD_WEATHER	Areas of the maps should have different obstacles and the map should not be able to be navigated with no obstacles for its entirety	UR_NEW_OBSTACLES

FR_LOSE_GAME	The player shall lose the game if they’ve been defeated in combat by another college or lose the objective (run out of time)	UR_OBJECTIVES_STAGE
FR_WORLD_GENERATION	The world shall be a single map that remains constant across play sessions	UR_WORLD_SCHEME
FR_SHOPPING	There should be ways to spend accumulated plunder	UR_GAME_CURRENCY
FR_WORLD	The world should have a single map that one single player exist and move within	UR_SINGLE_WORLD
FR_CONTROL_CHOICE	Players are allowed to use the combination of keys and click to move	UR_MOVE_SCHEME
FR_COLLEGE_ATTACKS	Colleges will have simple AI to attack the player when near	UR_COLLEGE_FIGHT_BACK
FR_COLLEGE_DESIGN	The game will contain at least 3 distinct and recognisable colleges.	UR_COLLEGE_QUANTITY
FR_COLLEGE_BATTLE	Colleges will be able to be defeated in combat as an objective.	UR_COLLEGE_COMBAT
FR_OBJECTIVES_STRUCTURE	The game will provide the player with a sub objective and a goal objective. All sub objectives must be performed in order to complete the goal objective	UR_OBJECTIVES_STAGE
FR_OBJECTIVES_ORDER	The game will provide the player with a new sub objective after they complete a sub objective. After the player has completed all sub objectives the goal objective will be given.	UR_OBJECTIVES_STAGE
FR_OBJECTIVES_UI	The user interface will always display the current sub objective. The UI will show the goal objective at least at the start of the game and when the goal objective can be completed.	UR_OBJECTIVES_UI
FR_POWERUP	Five different types of power-ups will be available to collect from the map. These can allow the player, speed increase, temporary immunity, health repairs, higher rate of fire or higher damage per bullet	UR_POWERUP
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

FR_SAVING	The player must be able to pause the game by saving their current state and resume the game at a later time	UR_SAVE_STATE

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	100% of the game will perform from 15 inch laptop to 27 inch monitor
NFR_HARDWARE_REQUIREMENTS	The game shall not have high hardware requirements.	UR_SCALABLE	Any response should be within 0.5 of an input
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_CHILD_APPROPRIATE	The content in the game shall not contain content that isn't suitable for children.	UR_CHILD_APPROPRIATE	Follows guidelines for PEGI-7[2]
NFR_TIMING	The application will be completed within 5-10 minutes	UR_TIMING	90% of the users will be able to complete the game within 10 minutes
NFR_FILE_SIZE	The application will have a limited file size to accommodate the data portal	UR_USABILITY	The entire zipfile can be successfully uploaded to the portal
NFR_SAVING	The application will be able to save/load one saved state	UR_SAVE_STATE	The state of the game in the main screen can be paused at e

- 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_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

- UR_COLLEGE_FIGHT_BACK and UR_AI_PIRATE
    - This requirement uses the terminology “seems intelligent”, our group understood this to mean that the colleges have some potential to aim for the player, and are able to hit them enough to pose a threat
- NFR_SCREEN_SIZE
    - This requirement also features a deal of ambiguity with an almost infinite amount of combinations of aspect ratios and resolutions available for screens. This will be deemed complete if the game automatically adjusts for the different sizes without compromising the image itself.
- NFR_HARDWARE_REQUIREMENTS
    - This relies on the assumption that we can source a department laptop to test our game on. If this is not the case, we can make an assumption that within our group the range of computers at our disposal would be enough of a representation that if the game does not take too much resources, it will at least run on a department laptop.
- 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

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

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