# Requirements

## Cohort 1 Team 9 - Team ‘ARRRGH’

Dominic Hall
Firas Marzouk
Ben Morrison
Harry Whittaker
Amelia Wigglesworth
Zehang Li

**2. (a)**

At the start of the project we were presented with a project brief, outlining what would be required from the finished game.

We read through this, taking note of any questions we felt were unanswered. For example, some of these questions were “How do we determine when the game ends?” and “What are the specifications of the machine the game will run on?”. These were really important to help us both design and implement the game.

We arranged a meeting with the customer to clarify what the stakeholders wanted, and confirm our understanding of the project brief. It gave us the opportunity to discuss what our interpretation of the initial brief entailed, and get feedback from the customer for what their expectations of the game were. We gained valuable insight from this meeting, such as learning that the customer would like an ability to mute audio if there is any, and that the game should be accessible to colourblind users.

We presented our requirements in a table format. This format makes it easy to split down requirements into small manageable chunks that we are able to work through and check off as we implement them [1]. It is also easy to read and understand, even allowing us to see how requirements interact - for example which system requirements support which user requirements. We made 3 tables, detailed below:

- User Requirements *(ID in the form UR_X, Description, Priority)*
- Functional Requirements *(ID in the form FR_X, Description, linked User Requirement)*
- Non-Functional Requirements (*ID in the form NFR_X, Description, linked User Requirement, the ‘fit criteria’ needed to meet this requirement)*

Each requirement has a unique ID, which is descriptive rather than numeric, so referencing the requirements is easy, and quick to understand.

The voice in which we present our requirements is important [2]. We used an active voice with clear language, such as “shall/should” for necessity, to avoid confusion. This meant there was little to no ambiguity within the requirements tables so the team were able to restrict what is implemented accurately.

[1] Lecture “Requirements Engineering” Aut/8 by Dimitris Kolovos. Last accessed 17th January 2022

[2] ISO/IEC/ IEEE 29148 “Systems and software engineering — Life cycle processes — Requirements engineering” Second edition. Last accessed 25th January 2022

**2. (b)**

User Requirements Table

| ID | Description | Priority (1-5 where 5=highest) |
|---|---|---|
| *UR_TEAM_SELECT* | The user shall select a college to play as from (a minimum of) 3 different teams in the game.<br>The enemy teams shall be made up of the other colleges. | 2 |
| *UR_SINGLE_PLAYER* | The game shall be single player with a single difficulty | 3 |
| *UR_MOVE_PLAYER* | The user shall to move their player on the water, but not on the land | 4 |
| *UR_GOALS* | The user shall view their progress towards their goals and sub goals, such as points collection. | 4 |
| *UR_END* | The game shall end: when the player ship sinks or time runs out the game is over. | 3 |
| *UR_REPLAY* | When the game has finished, the user should be able to click to replay from the start. | 2 |
| *UR_TIPS* | Tool tips and gameplay suggestions should pop up for players as they progress to guide their gameplay. It should only prompt a tip for the player once (i.e. not repeatedly telling the player what to do). As we are assuming all players are new players, due to the context of open day. | 2 |
| *UR_AUDIO** | There should be an ability to toggle mute the sound effects and music in the game. | 1 |
| *UR_VISUAL* | The player should be able to see and easily distinguish between different aspects of the game and understand what to interact with. | 3 |
| *UR_WIN* | The player must be able to win the game by meeting some requirements. | 5 |
| *UR_RESIZE* | The user should be able to scale and resize the game window. | 3 |

*UR_AUDIO Environment concern; a user does not have a way of listening to audio (they have no physical speakers) so the requirement becomes redundant.

Functional requirement Table

| ID | Description | User Requirements ID |
|---|---|---|
| *FR_END* | The game should end. | *UR_END* |
| *FR_STATS* | It should display and track game stats on screen. | *UR_GOALS* |

| | | |
|---|---|---|
| FR_REPAIR | It should allow game stats to be modified. | UR_REPAIR |
| FR_POINT | It should keep track of game stats. | UR_WIN, UR_GOALS |
| FR_AI | Non-player ships must display and move on the map, taking into account their surroundings. | UR_SINGLE_PLAYER |
| FR_INPUT | The player ship should move in response to keyboard inputs. | UR_MOVE_PLAYER |
| FR_SOUND* | There should be the ability to mute music / sound effects. | UR_AUDIO |
| FR_TIPS | Text boxes with tips should display at the start of the game. | UR_TIPS |
| FR_RESTART | Game shall have a button to restart at the end. | UR_REPLAY |
| FR_COLLIDE | Should prevent the player from moving off the map. | UR_MOVE_PLAYER |

*FR_SOUND Alternative: No audio is implemented in the game

Non-Functional Requirements Table

| ID | Description | User requirement | Fit criteria |
|---|---|---|---|
| NFR_CONTROLS | The player shall use a keyboard and mouse to control a single ship to navigate around the map. | UR_MOVE_PLAYER | All presses on the keyboard should invoke response, unless game is over |
| NFR_MAP | Whole map should be bordered with a shore line with enemy bases (harbour/dock) on opposite ends. There can be islands throughout the centre of the map with neutral/allied colleges. | UR_MOVE_PLAYER | There should be no way to 'escape' the map and there should be |
| NFR_GENRE | Shall be an arcade style game (one time play; highscore system etc.) | UR_SINGLE_PLAYER | The game should be quick to pick up and play with little instructions necessary |
| NFR_GAMEMODE | There shall be a single mode for the game, whether sailing or fighting (no cutscenes, or separate fighting screen). | UR_GOALS | The user should be able to always see their progress |
| NFR_COLOURBLIND | Unique identifiers shall not rely on colours. | UR_VISUAL | Any sprites that need to be distinguished from one another should use shapes or words instead of colours. |

| | | | |
|---|---|---|---|
| *NFR_DIFFICULTY* | There shall be only one set difficulty. | *UR_GOALS* | The game should not get harder/easier to play at any point |
| *NFR_PG* | The game shall be PG Rated. | *n/a* | There will be no blood, or swearing in the game. |
| *NFR_GUI* | The game should be bright and graphics need to be large enough for people physically walking past to be intrigued to play. Equally graphics should be appealing and not too excessive. | *UR_VISUAL* | Graphics should be visible on a 13 inch screen and legible. |
| *NFR_SECURITY** | The game shall not access the network, and should work completely offline. It shall also not collect any user information. | *n/a* | The game will not access any files or data from outside the game files themselves |
| *NFR_PLATFORM* | The game shall be a desktop game only. The game shall however be cross-platform in terms of desktop operating systems (i.e. MacOS, Linux, Windows) | *n/a* | The game will be able to be run on all desktop operating systems |
| *NFR_HARDWARE*** | A minimum of 4GB RAM, and all visuals easily distinguishable on a 13 inch screen | *n/a* | |

*NFR_SECURITY Risk; due to a lack of extensive testing there is no guarantee there is no bug that could result in other data on the computer being accessed. This links to risk R10 within the Risk Assessment.

**NFR_HARDWARE Risk; there is no guarantee that excess hardware usage may not be used, and possibly cause a computer fault. This links to risk R10 within the Risk Assessment.