Implementation

Cohort 1 - Team 2

Team Members

Camran Qadeer

Kieron Jones

Will Burden

Tabitha Oldfield

Adam Nicholson

- For this section, we are going to go through each requirement that we elicited at the start of the project and explain the outcomes of them as well as linking to the relevant parts of our updated architecture
- The same colour key as used in the requirements deliverable is here for your convenience
    - **Blue**: Features that do not need to be implemented (the stakeholder did not explicitly ask for), so are **low priority**. The original team has also claimed that some of these were implemented, but we did not find evidence of such
    - **Yellow**: Assessment 2 requirements; **high priority**
    - **Red**: Either features that the original team have claimed *were* implemented (based on their implementation document), but aren't apparent in the game, or features from assessment 1 that they did not complete

USER REQUIREMENTS

UR_FRIENDLY_SHIP_ENCOUNTER

- This requirement **has not** been implemented
- The original team had already left some friendly ships on the map however they did not perform any actions, and we did not do anything further with them as they were a low priority requirement
- This was part of a folder called 'AI' which contained a lot of unused and broken code, so it was hard to clean this up as part of it still made the enemy ships move

UR_BULLET_DODGE

- This requirement **has** been implemented
- Enemy ships will fire at the player ship, and will damage the ship if the cannonballs make contact
- When a member of class NPCShip collides with another entity, it checks whether it is a cannonball and takes damage if the ship that fired the cannonball belongs to a different faction
    - NFR_SHIP_COLLISION

UR_FRIENDLY_BUILDING_INTERACT

- This requirement **has not** been implemented
- The original team said that this was implemented yet we could find no indication of such, so there was no need to do this feature

UR_EARN_POINTS

- This requirement **has** been implemented
- The user will earn points for time spent alive, as well as destroying enemy ships, colleges etc
- We have used 'points' as the only score-keeping feature in the game, as explained with the requirement directly below

UR_EARN_XP

- This requirement **has not** been implemented
- Having 'points' as well as 'xp' was needless, as both can be applied to the same activity in the game

UR_GAME_LOSE

- This requirement **has** been implemented
- No matter the difficulty level, if your ship's health reaches 0, you will die and the game over screen will be displayed

- The GameScreen class checks the player's health in its update method and switches to the EndScreen if health reaches 0, which ends the game and displays the player's stats
- Damage can be taken from colliding with obstacles or by being hit by enemy cannonballs

UR_SHIP_COMBAT

- This requirement **has** been implemented
- The user is able to fire cannonballs at enemy ships and colleges, and is able to damage and destroy them
- This is done by clicking an area with the mouse, or by pressing the spacebar to shoot in the direction of the ship
- These controls are also displayed on the screen
- Both player and NPC ships can detect collisions with cannonballs and take damage if they are hit by a cannonball fired by a member of a different faction
- If they collide with the player ship, NPCShips add the player to a list of targets, and will then follow and fire at them.
- Ships do not take damage from cannonballs fired by a member of their faction

UR_OBSTACLE_ENCOUNTER

- This requirement **has** been implemented
- There are obstacles on the tilemap in the form of driftwood (floating logs) that the user has to navigate around, otherwise they will take damage if they sail into it
    - The method that takes health from the player is a bit buggy, as it takes health away from the player twice instead of once, due to how the original team made their game update every second

UR_WEATHER_ENCOUNTER

- This requirement **has not** been implemented
- This is the one requirement we were unable to implement due to time constraints, and we felt that since we had done all the others it would be a better use of time to work on more tests

UR_SPEND_MONEY

- This requirement **has** been implemented
- The player is able to acquire plunder by defeating enemy pirate ships and collecting treasure chests
- By pressing 'Z', you can access the shop to spend money on powerups, such as increased speed or health

UR_POWERUPS

- This requirement **has** been implemented
- The PauseScreen menu allows the player to buy powerups when they have earned enough plunder.
- There are five powerups: increased health, increased ammo, increased speed, firing multiple cannonballs at once and reducing the damage taken from enemies.
- Five new methods were added to the Pirate class to check that the player has enough plunder and then apply the powerup

UR_DIFFICULTIES

- This requirement **has** been implemented
- The player may select the difficulty of the game by clicking the 'difficulty' button on the menu at the start of the game

- This opens the DifficultyScreen menu, which allows the player to choose easy, medium or hard difficulty, each of which changes the player's starting stats, with easy giving them more ammo and health and hard giving them less
- The game will play on medium difficulty by default if the player doesn't choose a difficulty option

FUNCTIONAL REQUIREMENTS

FR_MENU_KB_INPUT

- This requirement **has not** been implemented
- This was an unnecessary feature as you need the mouse to play the game anyway

FR_CROSS_PLATFORM_GNU_LINUX

- This requirement **has** been implemented
- The game will run on Linux (as well as Windows and MacOSs)

FR_GAME_RESET -> FR_SAVE_STATE

- This requirement **has** been implemented and blended into FR_SAVE_STATE
- This requirement is applicable to 'saving the state of a game and resuming at a different time', so due to the overlap they have essentially been merged
- This will save your position and stats, but not your quest progression

FR_FRIENDLY_AI

- This requirement **has not** been implemented
- There are friendly ships in the game however they are not able to move
- This was due to time constraints in the project

FR_FRIENDLY_INTERACT

- This requirement **has not** been implemented
- The user cannot interact with the friendly ships, other than by colliding with them
- This was due to time constraints in the project

FR_HOSTILE_AI

- This requirement **has** been implemented
- The game will control the actions of enemy ships, which is following the player and firing at them
- When an NPCShip collides with another entity, it checks whether it is a ship and if so whether it is a member of a different faction. If so, the NPCShip will follow the other ship and fire at it
- If the other ship leaves the NPCShip's attack range, the NPCShip will cease to target it and return to its idle behaviour
- They are also able to navigate around obstacles

FR_POINTS_TRACKING

- This requirement **has** been implemented
- The game will display your points in the top left corner, along with health, cannonballs, etc

FR_POINTS_UPDATE

- This requirement **has** been implemented
- The game will give you points for time survived, enemies killed, etc

FR_XP_TRACKING + FR_XP_UPDATE

- These requirements **have not** been implemented
- As explained in UR_EARN_XP, 'points' are used instead of having both

FR_BOSS_UNLOCK_TRACKING + FR_BOSS_SPAWN

- This requirement **has not** been implemented
- The implementation of a boss fight was not implemented due to time constraints

FR_GAME_WIN

- This **has** been implemented
- Upon completion of all quests, the player will win and be shown a screen that details their stats across the game
- The EndScreen class is shown when the player wins, but its text is changed to report a win instead of a loss, as loss is its default status.

FR_PLAYER_DEFEAT + FR_SCENARIO_FAIL

- This **has** been implemented
- If the player dies, they will lose and be shown a screen that details their stats up to that point
- These requirements have essentially been merged as they were described as the same thing by the original team

NON-FUNCTIONAL REQUIREMENTS

NFR_SHIP_COLLISIONS

- This requirement **has** been implemented
- Despite the original team saying it was not implemented, when we first interacted with the game we found that the player's ship would collide with enemy ones, and not pass through them
- This now also applies to cannonballs hitting either type of ship

NFR_GAME_DURATION

- This requirement **has** been implemented
- A typical playthrough of the game (to completion) is around 5 minutes

Other Changes To The Original Code

- The 'Text' class was removed completely as it was not being used at all
- The game code was refactored to run in headless mode in order for testing to be performed, as explained in the Software Testing Report