PLAN Doc.

{assessment 2}

Doc is unfinished but should show some basic ideas to look into. Basically trying to create ideas to fuel a group discussion of how we envision the game turning out.

TLDR: **2**

Set-up 2

Combat 2

Shop 2

Initial state of project 25: **4**

Why 25? 4

Overview of 25: 4

Initial issues with 25: 5

Additional features: **6**

General **7**

Flow 7

Player **8**

Movement: 8

Attacking: 8

College **9**

Attacking: 9

Enemy ships **10**

Positioning: 10

Attacking: 10

Plunder **11**

Shop: 11

Layout: 12

Available in the shop: 13

Weapons: 13

Power-ups: **14**

Difficulty: **15**

Menus: **16**

Main menu: 16

Pause screen: 16

End screen: 16

Notes related to Assessment 2: **17**

TLDR:

This whole document is just an idea of how the game might function. We don't have to do things this way and it might open up a discussion in the event that anyone wants to do something differently which is useful.

Set-up

Colleges are distributed evenly around the map. Enemy ships orbit their parent college. The player approaches colleges in increasing order of difficulty and must therefore upgrade their ship to make defeating them more manageable.

The player starts in an empty part of the map and is pointed in the general direction of the next college. On the way to the college the player may encounter sea monsters that it must avoid or take heavy damage from. Upon reaching the college, the player initiates combat with it.

Combat

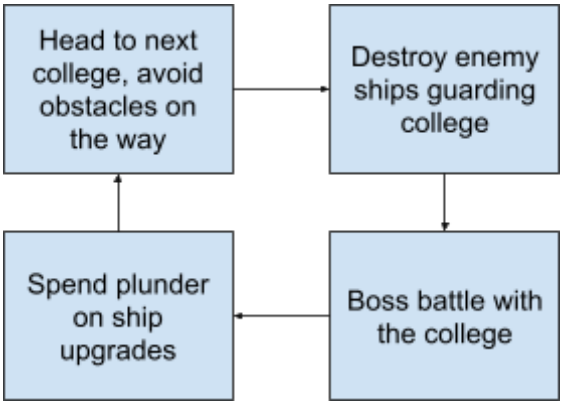
Both colleges and enemy ships fire in multiple directions, cycling through an array of predetermined patterns. The main focus for the player would be to dodge these attacks, returning fire when the chance comes around. While there are enemy ships guarding a college, the college would be more resistant to attacks. This would give the player an incentive to take out the enemy ships first. Once all the enemy ships have been removed, the college will start attacking with special attacks that are stronger and unique to that college (kind of like boss fight).

Shop

If the player defeats the college they receive plunder from it and are presented with the shop screen where they can upgrade their ship. The shop offers the chance to repair the ship, upgrade some of the ship's core stats or exchange their weapon to one with different mechanics. The

items in the shop would be randomly selected. For each selected item a random rarity would be assigned with the rarer items providing better value for the cost. This would provide variance and replayability while keeping the game simple at its core.

The cycle looks like this >>
Nice and simple.



Initial state of project 25:

Why 25?

We picked the project of group 25 for one main reason - It's really simple. This makes it much easier to add new features and minimises the time spent sorting out any issues left by the previous team. The retro nature of the game also means that any future features can be implemented in a lightweight fashion that adheres to the style of the game (again, saving time). The code should be simple to understand and once the initial bugs have been ironed out, everyone should be able to more comfortably contribute to the code.

Overview of 25:

Project 25 took the approach of making each college fire multiple projectiles in an array of predetermined patterns. It's up to the player to dodge the projectiles while also returning fire. This kind of gameplay is similar to something like 'Enter the Gungeon':

Initial issues with 25:

Project 25 contains a large amount of fairly noticeable bugs that would need to be fixed before we add any code to the project. Thankfully due to the simple nature of the project, these bugs should be super easy to fix. I'll check up on the assessment handouts to see if anything is mentioned about modifying their code. We asked the supervisor at the practicals and he said it's fine if you can give a good reason as to why it needs modifying. So the reason for every modification is 'there was a bug'.

A handful of issues have been added to a new Jira board which can be found here.

The performance of the game is really poor, mainly due to the inefficient use of batches to render the sprites. When each bullet is created it creates its own batch unique to itself. This means up to 24 batch objects are created every time the colleges shoot. This currently causes a stutter and results in poor frame rates. The fix is simple but requires replacing 90% of the code used for rendering.

We are also adding shaders as a quick but super effective way to play into the retro style. The shaders utilise the GPU so are super lightweight and shouldn't impact performance by any measurable amount.

The game is also missing a lake which we can potentially add ourselves as we implement box2D for the collision between entities in a similar way to assessment 1. Thanks to the retro style of the game, we can keep an abstract representation of the lake (like a square) which would allow really simple hit detection with the border (if player.posx > 10 and so on).

Additional features:

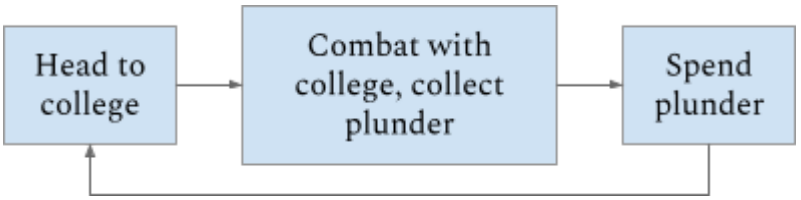
There are 6 additional features that we need to add. 1-3 are described in the initial assessment brief and 4-6 are mentioned on the vle:

1. There should be a way to "**spend**" **the plunder** (e.g., gold) acquired, e.g., to repair a ship, to acquire new provisions or weapons etc.
2. The player may engage in **combat with another ship** or College.
3. While sailing across the Lake of York **the player may encounter** other pirates/privateers, **obstacles** (e.g., Lake Monsters), or **bad weather.**
4. Implement **five special power ups** that the player's boat can obtain on the journey (e.g. granting temporary immunity, repairing damages).
5. Implement support for **different levels of difficulty** in the game (e.g. easy, normal, hard).
6. Implement facilities that allow players to **save the state of the game** at any point and resume a saved game later.

General

Flow

The player would start in an empty part of the map. The colleges are spaced out around the map. There is an arrow pointing the player towards the next college. Once within a certain distance of the college, the player enters combat and the enemy entities begin to fire. During combat the player would collect plunder and probably take damage. After combat has ended, the player is able to spend plunder in the shop to repair and upgrade their ship so they are ready to progress to the next college. The next college is spawned away from the player. The player is pointed towards the college where the cycle repeats.

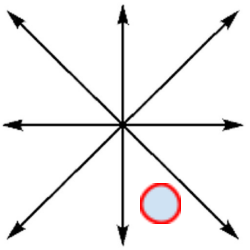


After each combat, the shop presented will provide the player with a different set of random items.

Player

Movement:

The player movement should be responsive, easily controllable and heavily damped to allow the player to dodge projectiles.



8-way movement would fit the theme of the game and is very simple to implement. The player would use WASD keys to control the player's movement.

Attacking:

The player would be able to attack in the same 8 directions using the arrow keys. This would prevent the player from positioning like the image above which would allow them to attack the enemy without the need to dodge attacks. (*Only thing is that the hitbox of the player would need to be larger than that of the player // or make college projectile hitbox larger than players projectile hitbox*)

College

Attacking:

Colleges would attack by shooting projectiles in a pattern of directions. They would randomly cycle through different attack patterns to make their attacks more unpredictable.

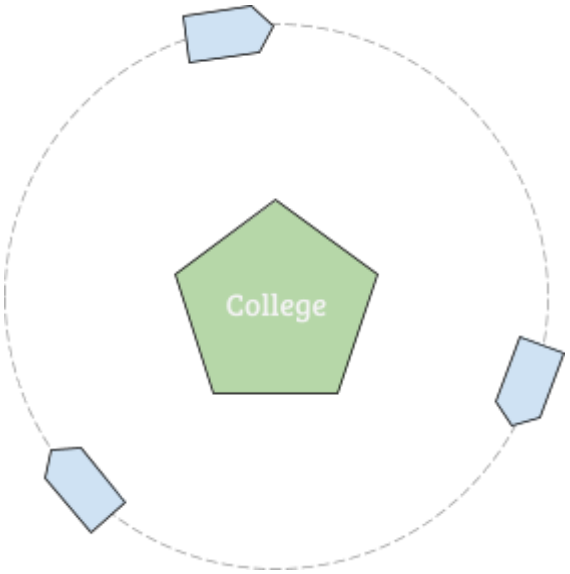
Could consider adding ‘special attacks’ (unique patterns) to each college that are more of a threat, and more rare than their standard attacks. This would be simple to implement but it’s not strictly necessary so maybe add towards the end if all else is fine.

Previously we had the colleges aim at the player when they attacked but the cannonball would always miss, going behind the player if they were travelling perpendicular to the segment connecting them. We could implement a system where the enemy takes into consideration the player's speed and heading then leads its target but this is all getting fairly complicated when we are focused on keeping stuff simple. For this reason I think keeping the attacks to patterns is the best route to take.

Enemy ships

Positioning:

One approach is for the colleges to take the form of an island and have the boats 'patrol' the island. To do this the boats would simply orbit in a circular pattern which would be very easy to implement. The distance from the island could be balanced (closer to the island would make it harder).



Attacking:

There are a few ways that the enemy ships could attack:

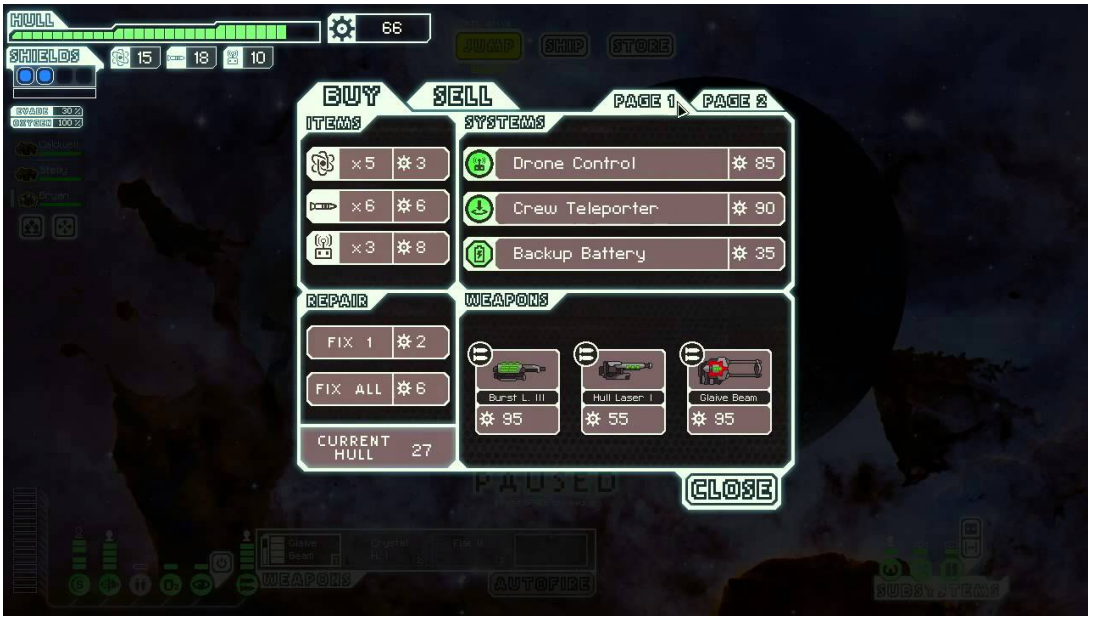
- Shoot towards the player. *More difficult, would be harder to dodge.*
- Shoot in a pattern. *More predictable. - Gets harder for the later colleges - e.g. more ships or different firing patterns*

Plunder

Plunder is the currency that the player can use to purchase **permanent** improvements to their ship. The player would spend the plunder through a shop screen that presents them with a range of items and upgrades to purchase. The player would receive plunder for defeating colleges and enemy ships as a way to improve their ship for the next fight. Spending plunder could also change the ship visually (like weapons being shown on deck, max hp changing material of hull to metal).

Shop:

One possible implementation of the shop takes inspiration from FTL:

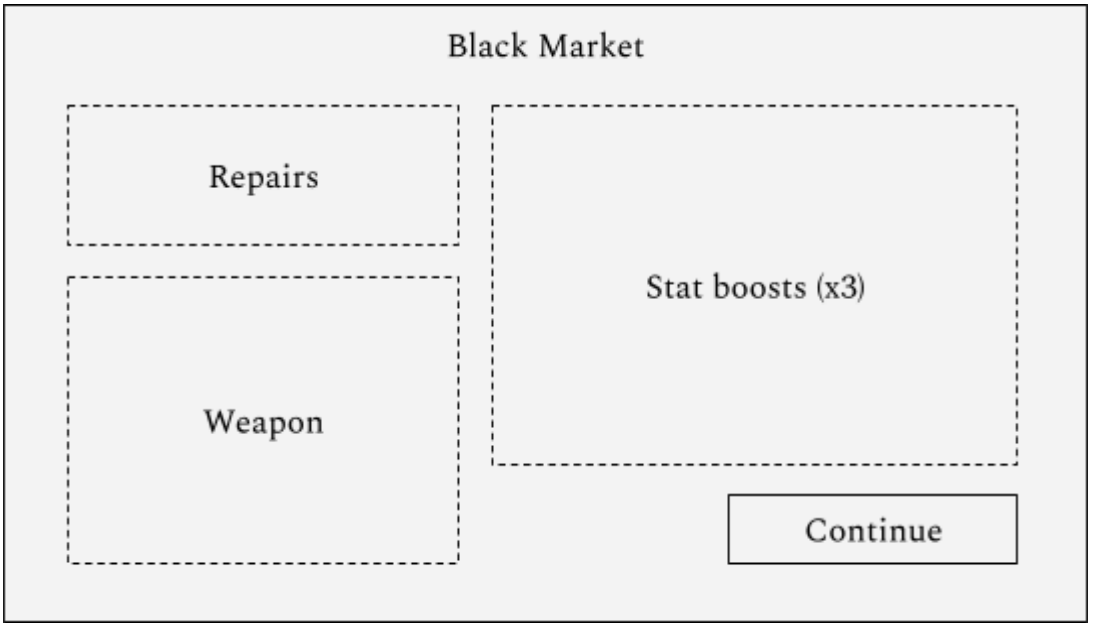


In FTL the player uses scrap to purchase a range of upgrades. Scrap can be spent on repairs, ammunition, permanent upgrades, and weapons. The items available to purchase are randomised, with better items being more rare and costing more scrap.

We could implement a simplified version of this shop, we would present the player with the ability to repair their ship as well as the option to purchase permanent improvements.

In our game, the contents of the shop would be different after each college is defeated. The random nature of the shop would result in more variation between each playthrough.

Layout:



An example layout. The key points are that it should be simple and easy to navigate using a keyboard or a mouse as input.

Each item in the shop available to purchase would take the form of a box, displaying an icon, item description and cost as shown below:

The item rarity could be indicated by changing its border colour.

Available in the shop:

The items available in the shop have different rarities for example:

Increase maximum hp:

Rarity	Increase by (hp)	Chance (%)	Price (coins)
Common (White)	15	60	15
Rare (Blue)	25	25	20
Legendary (Yellow)	40	15	25

The chance of each rarity could change with the selected difficulty.

Each stat boost could always appear in the shop but with randomised rarity. Alternatively, only a smaller selection of the items would appear, along with the randomised rarity.

Health and increase of stats can be bought with increasing price after every purchase. Stats bar like this then a plus sign to add a section of the bar. For damage, speed, ...

Weapons:

The player can replace the weapon on their ship. Different weapons with different pros and cons (e.g shotgun - three bullets at once but shorter distance, sniper - more damage slower fire rate)

Weapon Effects:

Rarer weapons have additional effects applied to their attacks:

- Lifesteal
- Chain damage (statikk shiv)
- Poison / Fire (Damage over time)
- *Crit chance?*
-

Power-ups:

Power-ups would be **temporary** boosts to the player that wear off after a certain amount of time. Depending on the difficulty selected the power-ups would have different strengths and drop rates. They could drop randomly or after defeating an enemy entity. The player picks up power-ups by colliding with them.

The player can upgrade their ship with plunder so power-ups should give me more of a simple boost, leaving the creativity for plunder items.

We need to include 5. Here are some suggestions:

- Heal player
- Increase player attack speed
- Increase player damage
- Temporary immunity to damage
- Increase player movement speed

Power-ups could be dropped when the player kills an enemy ship/ sea monster.

Difficulty:

The difficulty of the game would be selected before each play through in the menu. Playing on a higher difficulty could result in:

- Increased **score**
- Increased **enemy damage**
- Reduced **player health**
- Reduced **power-up drop chance**
- Reduced amount of **plunder dropped**

3 levels of difficulty is typical for most games (**Easy**/**Medium**/**Hard**). Sometimes a 4th level is added beyond hard, something along the lines of **Legendary** or **Extreme.** Once we have set up the code to handle difficulties, adding a 4th option would be easy so it's probably best to start with 3 and then add another if it seems to be needed.

The names of the difficulties could be changed to fit the theme, some possible suggestions:

- Easy / Medium / Hard
- Unknown / Notorious / Legendary
- Castaway / Deckhand / Captain

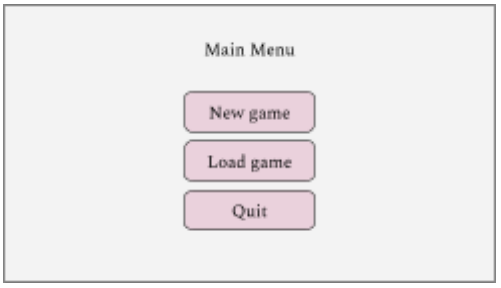
Menus:

Now that we have to implement difficulty and save files, it's worth implementing a set of menu screens. I've updated the start screen to use scene2d so adding stuff to the menu should be super simple now. We can use pretty much the same code to implement a pause, shop and after game screen.

Main menu:

Shown when the game loads, it would allow the player to:

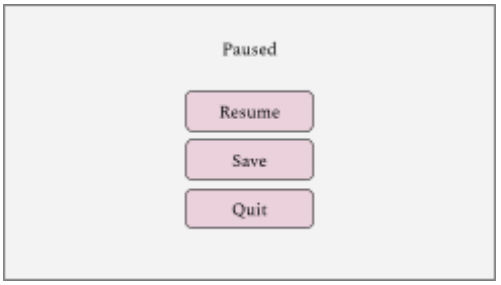
- Start a **new game**
- **Load** an existing save file
- **Quit**



Pause screen:

The player can pause the game at any point minus the start menu and end screens.

- **Continue** with the game.
- **Save** the current game state.
- **Quit**



End screen:

Shown after the player dies or completes a successful run. Displays a message about winning or losing, the players score and a button to return to the main menu.

Saving:

Implement a method to save the game, maybe by pressing a key to save the game to a specific state

https://libgdx.badlogicgames.com/ci/nightlies/docs/api/com/badlogic/gdx/Preferences.html

The user should be able to save to multiple different save state

Loading:

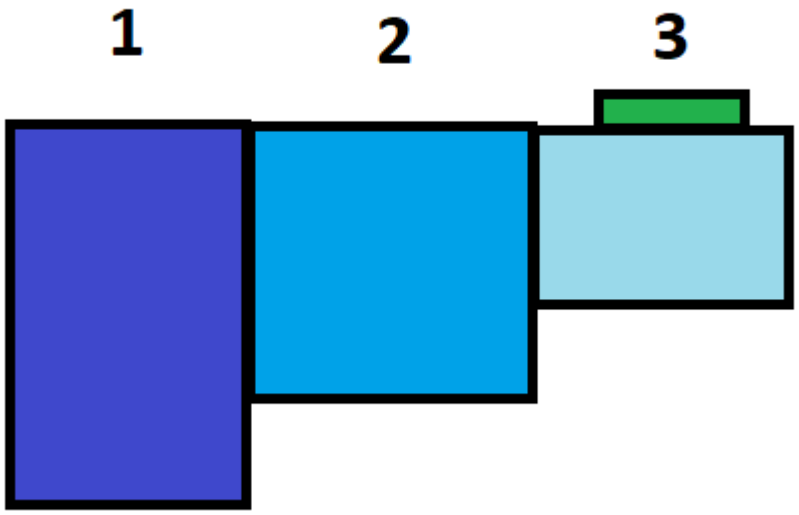
There should be some kind of screen at the start of the game in which the user can pick between which game state they would like to load from.

These could have data next to them stating what time/date they were saved.

Sea Monsters:

The map is made of 3 different depths of water:

- Shallow:
- Normal: Sea
- Deep: If the player spends too much time in this water they will die.



AI:

Implement some kind of simple AI for the enemy ships and sea monster.

- gdx-ai looks like a good AI framework to do this with
- https://github.com/libgdx/gdx-ai
- Can implement steering behaviours and collision avoidance

Notes related to Assessment 2:

- Look into using Finite State Machines (yeah, those ones) to make more complex entity behaviour more manageable. Looks a bit complex at first but it simplifies creating more intelligent enemies. info here
- They have implemented a box2D world but don't utilise it fully. It's worth looking into using it properly for hitboxes as we did before.
- Setting up the shop is probably the hardest part, just need to figure out the best way to have items effects be applied to the player. Using an entity component system is perfect for this stuff but that would change the whole architecture so instead maybe we have an item interface that allows us to create items. They have a function called onAttack, onPlayerHit, and so on. The player contains an array of its current items, when it attacks, it runs through the array and calls the onAttack function for every item. Worth talking about this as implementing in the wrong way could make the code pretty confusing.
- Look at how they do stuff here, using public classes for assets and so on. This should make the whole passing reference issue much more manageable.
- Camera moving with the player makes dodging a bit more difficult. Camera should lock onto college when combat with the college starts. If the player gets out of combat range, the camera returns to the player and combat ends.