Part A

Software Engineering Methods

Our chosen software engineering method was plan-based rather than agile, with a project "road-map" being created early in the development process as illustrated below. This seemed most appropriate as we anticipated no requirement adjustments to manage during the first assessment.

We organised frequent team meetings, on average twice a week. These meetings served two key purposes:

- Providing a platform to discuss and agree upon key development decisions, most often regarding game design
- Allowing each team member to communicate their progress on their assigned tasks, and for new tasks to be assigned when necessary

When creating the initial "road-map", we chose not to pre-emptively assign any tasks to particular people, instead we assigned tasks throughout the project with a view to utilising each member's skill set whilst keeping the workload even. Assigning tasks throughout the project was particularly appropriate for a team without large amounts of software development experience as our estimations for the time and effort needed for each task weren't perfect, therefore this system avoided any particular team member being assigned tasks at the start of the project which transpired to be an unfair portion of the workload. The frequent team meetings in which we discussed task progression, however, allowed us to keep each other accountable for our productivity.

Tools Used

Communication and Collaboration

- For our team meetings we used Zoom. This was an intuitive choice as it is a platform that we all have lots of experience with and therefore avoided an unnecessary learning curve.
- We created a Discord server for general communications throughout the project. This was crucial in allowing team members to ask clarifying questions without having to wait for the next meeting, avoiding any unnecessary obstructions to task progression.
- We created a Trello board in order to organise and keep track of our tasks throughout the project. Alongside the team meetings, this was crucial in keeping us organised by providing a visual representation of tasks that were completed, in progress and yet to begin.

Website

- We used GitHub pages to develop our website. This meant that our resulting site would be simple to replicate for any team which chose to take over our game for assessment 2.

Architecture

- Draw.io was used in order to create the abstract architecture diagram, as this relatively easy to use tool seemed more appropriate for this simpler diagram
- PlantUML in addition to adobe photoshop was used in order to create the concrete architecture diagram;
    - o PlantUML was used at first in order to create representations of classes, categories of classes and the relationships within each category
    - o Adobe photoshop was later used in order to add the inter-category relationships

Implementation

- We chose to use IntelliJ as our IDE. This was due to both the ease of use offered by the tool, along with the fact that several team members had previous experience with the tool: it felt like the ideal choice to avoid unnecessary and time-consuming learning curves.
- We utilised the libGDX game development framework during the implementation of our game. Similarly, to our choice of IDE, this was largely influenced by the previous experience of the team. We were also aware that LibGDX was a popular choice amongst other teams and therefore our use of this framework may make it easier for another team to take over and expand our code.

Alternatives considered

- We considered using other IDEs such as Eclipse for software implementation, however as mentioned previously we chose to use IntelliJ due to team members having previous experience with this IDE
- We also considered game engines other than libGDX in order to enable software development:
    - We considered using Unity, however we were discouraged from this choice by factors such as Unity’s reputation for largely outdated/incomplete documentation and the fact that many useful features are behind a paywall.
    - We also considered using the Unreal game engine, but quickly decided against this as it seemed inappropriate for developing what is a relatively small game and would cause the resulting game to be unnecessarily bloated.

Part B

Team Roles

During our initial team meeting we discussed assigning the following team roles.

- Meeting Chair: Ensuring organised and efficient team meetings that covered all necessary updates and decisions
- Secretary: Recording team decisions and keeping notes of the content discussed in each team meeting
- Librarian: Keeping track of documents and other resources, particularly ensuring that in-progress documents were regularly uploaded to the team shared google drive
- Report Editor: Overseeing document production, in particular ensuring that documentation progress was largely in line with the initial working plan

During this discussion we decided to combine the roles of librarian and report editor as they seemed like largely interdependent tasks. We then discussed who would be most appropriate for each role and agreed on the following assignments:

- Meeting Chair – Stan
- Secretary – Jarred
- Librarian/Report Editor – Alex

Assigning these roles enabled a smooth and efficient team working process and helped to keep track of team progress.

Task Assignments

Task assignment took place throughout the assessment process as detailed in part a of this document. When the time came to assign new tasks, we tried to keep these assignments in line with each person's particular skill set in order to ensure efficiency as such throughout the project members mainly focussed on the particular aspects which they felt most comfortable with, usually due to previous experience, for example Joe took responsibility for website development and Alex produced the bulk of our game's code.

Part C

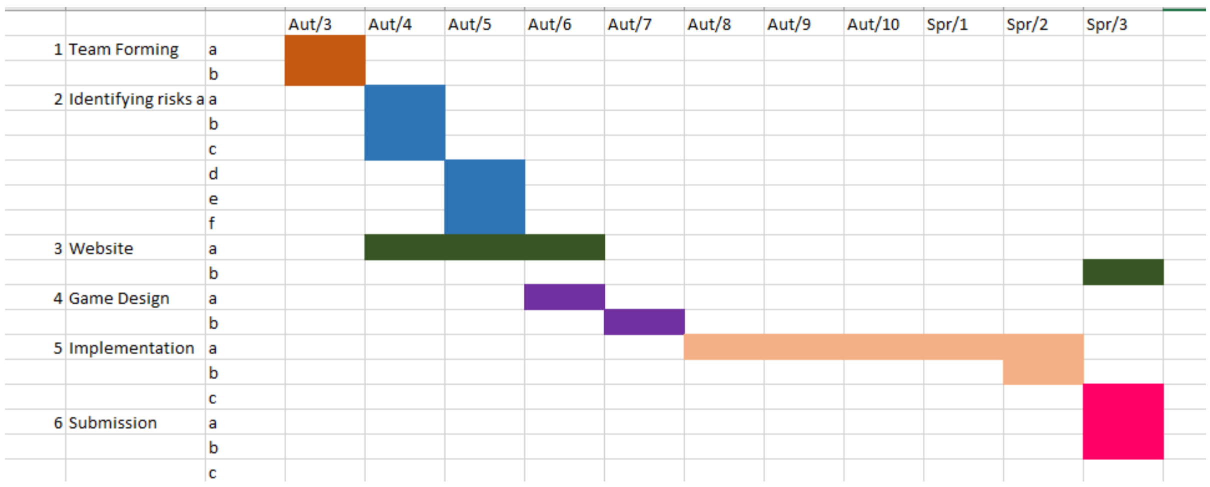
Intro

Our first step in planning this assessment was to create a task breakdown table, as shown below (note that time estimates were rounded up to the nearest week e.g., 1.a. which we believed would only need one team meeting are 1 week)

Main Task No.	**Main Task**	**Subtask letter.**	**Subtask**	**Dependencies**	**Estimated time (weeks)**	**Estimated Start and End Time (in Term Weeks)**
1	Team Forming	a	Team introductions and familiarisation	-	1	Aut/3
		b	Assigning team roles	-	1	Aut/3
2	Identifying risks and requirements	a	Meeting to discuss initial ideas about requirements and risks, and compile a list of client questions	-	1	Aut/4
		b	Client meeting to discuss requirements and ask formulated questions	2a	1	Aut/4
		c	Team meeting to agree upon necessary requirements and relevant risks	2b	1	Aut/4
		d	Creation of formal requirements representation (req1.b)	2c	1	Aut/5
		e	Creation of formal risk representation (risk1.b)	2c	1	Aut/5
		f	Written explanations of our requirement and risk process (req1.and risk1.a)	2d	1	Aut/5
3	Website	a	Create initial website (not yet populated with necessary documents)	-	3	Aut/6
		b	Add all necessary documents to website	5b, 3a	1	Spr/3
4	Game Design	a	Team meeting to discuss and agree upon game design ideas	2c	1	Aut/6
		b	Creation of abstract architecture diagrams	4a	2	Aut/7

5	Implementation	a	Creation of functional, commented code	4b	5	Spr/2
		b	Explanation of non-implemented features (impl1)	4a	1	Spr/2
		c	Creation of reflective concrete architecture diagrams	5a	1	Spr/3
6	Submission	a	Completing outstanding reflective documentation	5b	1	Spr/3
		b	Compiling documentation into PDFs, and combining this with code and website into a submittable zip file	6a	1	Spr/3
		c	Completing self and peer assessment	-	1	Spr/3

We then used this table in order to create an initial Gantt chart to show a theoretical schedule for when each task would be completed for assessment 1(as shown below;)



At each team meeting our assigned secretary, Jarred, would create meeting notes which he would then upload to our shared google drive. At the end of each week, we then used these notes in order to create a ‘snapshot’ of our team progress (shown on website snapshot page).

These snap-shots would be created by taking a condensed version of the task breakdown table and colour coding the tasks which were due to be done at this point in the project. With the following colour connotations:

- Red – Not started
- Orange – In progress
- Green – Completed