Part A

Software Engineering Methods

Our chosen Software Engineering method was Agile, where we decided to use the Scrum Framework. The team was divided into two smaller teams, namely, Implementation and Documentation. We then came up with a project “road-map” early in the development process as illustrated in Fig.1. As we decided to use the Scrum framework, our road-map shows a sprint of 2-weeks for each task followed by a review stage where we would discuss our progress and work on solutions to any problems the team is having. Once we had an idea of the main tasks that needed to be done, they were then broken down into sub-tasks based on their marks and each member was assigned ~15 marks worth of tasks based on their skillset and our experience from the first assessment, so that every member had an equal amount of contribution towards the project. Having all the requirements for the final game from the Customer, our road-map helped us efficiently plan the execution and completion of this project, with the aim of delivering a high-quality game and all its supporting documents on time.

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, such as game design and documentation.
- Allowing each team member to communicate their progress on their assigned tasks, and for new tasks to be assigned when necessary.
  The frequent team meetings alongside the bi-weekly review meetings, in which we discussed task progression, allowed us to keep each other accountable for our productivity and ensured that we stay on track.

Tools Used

Communication and Collaboration

- For our team meetings we either organised a discord call or met in-person. Using discord was an intuitive choice as it is a platform that we all have lots of experience with and therefore avoided an unnecessary learning curve. On the other hand, in-person meetings helped everyone get in the zone to put in work as working together in a group helped avoid any distractions.
- 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. Moreover, we created a "To-do list" channel in our server where after every meeting, notes were posted regarding what was discussed in the meeting and what needed to be done.
- We used GitHub Issues 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 numerical representation of tasks that were completed (closed issues) and tasks that were in progress or yet to begin (open issues).

Quality Control Process

To ensure that the work produced was of high-quality, our team decided that it was quite important to implement a Quality Control Process, whereby the work of one member is reviewed by at least one another member. To do this, we used GitHub Issues where once a member completed a task assigned to them, they would not be able to commit those changes to the main branch until they request any other team member for a review. Once the reviewer reviewed the changes and approved them, only then will a merge to the main branch be allowed. This ensured that the final work on the main branch was without any errors. Moreover, this also enabled us to implement “work-shadowing” as reviewing one another’s work from time to time kept us all up to date with the progress of tasks and would prove quite helpful in an emergency (e.g., if one of the team members were to fall ill, at least one member from the team would know what they were doing and where to pick-up from).

Website

- We used GitHub pages to develop our website considering the teams previous experience with using the tool and to avoid the lengthy and time-consuming process of making our own website.

Architecture

- We used PlantUML to create the Gantt Charts as well as Abstract and Concrete Architecture diagrams as the team had prior experience in using this tool.

Testing

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.

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

During this discussion we decided to combine the two roles as they seemed like largely interdependent tasks, hence forming the main role of Group Leader. We then added two additional roles, Implementation leader and Documentation leader, and discussed who would be most appropriate for each role and agreed on the following assignments:

- Group Leader – Katie
- Implementation Leader – Jacob
- Documentation Leader – Saud

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 once we had gathered all the requirements for the game as mentioned above. We went through all the deliverables required in Assessment 2 and discussed each major task that had to be done. We then calculated how much each task was worth and assigned each member ~15 marks worth of tasks with the help of GitHub Issues, based on their skill set and experience from Assessment 1, with the aim of ensuring efficiency and timely completion of tasks. We also made sure that each member’s work was reviewed by at least one other team member so that appropriate feedback could be given before the approval of a merge request and the member working on it could make any necessary changes. We then added sub-tasks to these major tasks and such sub-tasks were added till our last review session in order to ensure that each part of the deliverable was completed, and nothing was forgotten.

Part C

Intro

Our first step in planning this assessment was to create a "road-map" for the entire project in the form of a Gantt Chart, as shown below.

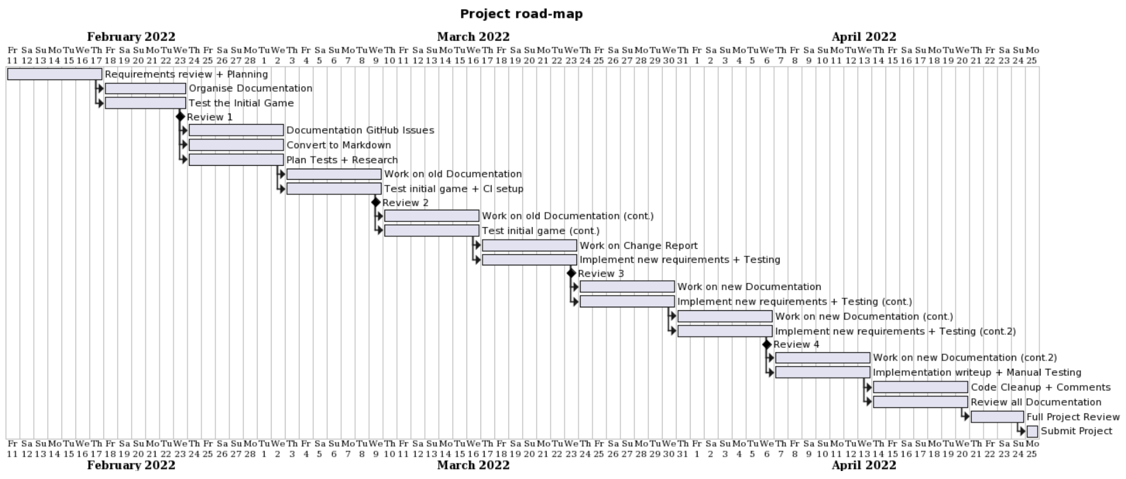
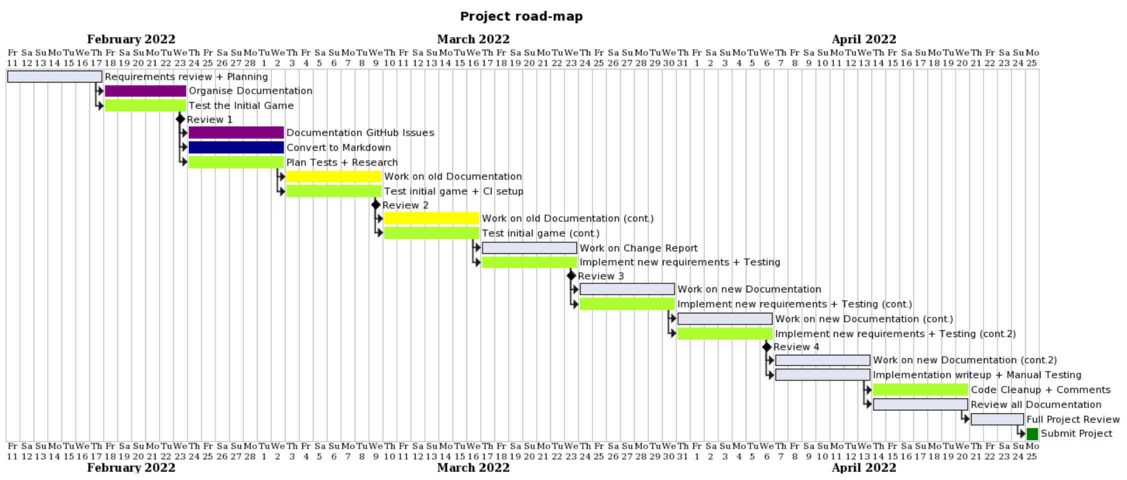


Fig.1

We made this road-map by referring to both, the Product Brief and the Team Assessment documents, highlighting all the main tasks which had to be completed in order to successfully complete and submit this project on time. Although the submission deadline was the 4th of May, we decided to set an early deadline for ourselves so that we could maximize efficiency and will have a few "buffer" days if in case things go wrong (e.g., a member falling ill and unable to do their assigned tasks so someone else taking over).

Once we had a Project Plan, we then assigned tasks to individuals/teams, using a colour-coding scheme, as shown below. The members for each of our two previously mentioned teams are as follows:

- Implementation Team: Jacob, Katie, Cody
- Documentation Team: Saud, Felix, Joachim



Task assignments using a colour-coding scheme

The colour-coding scheme is as follows:

- Everyone = Grey
- Implementation team = Green Yellow
- Documentation team = Yellow
- Saud = Dark Blue
- Katie = Purple
- Jacob = Green

At each team meeting our Group Leader, Katie, would create meeting notes which she would then upload to our discord server. Based on these notes, we would work on our assigned tasks/new tasks until the next meeting/review session where we would compare our progress with the road-map to see where we stand in terms of the submission deadline and the work to-be-done.