ENG1 Assessment 1: Method Selection and Planning

Greenfield Development

Group 5

Members:

Mark Faizi

Ben Faulkner

Christian Foister

Lewis Hanson

Ziyad Rashad Hassan

James Haworth

4a:

Team’s Methodology brief

The team adopted the Waterfall methodology for its structured, sequential approach, which suited the game’s fixed scope. Each development phase requirements, design, implementation, and testing was completed in order, allowing thorough documentation and timely delivery.

Discord facilitated real-time communication and task management, while LibGDX was chosen as the game engine due to its resources and support for Java. IntelliJ served as the IDE, optimised for LibGDX and Gradle, and GitHub managed the codebase with familiar tracking and collaboration tools. This combination enabled an organised and efficient development process, balancing structure with effective team coordination.

Justifications

Waterfall methodology

The team chose to use the Waterfall methodology for this project due to its linear and structured approach, which suited the defined nature of the game development tasks. Waterfall provided a clear framework, where each phase such as requirements gathering, design, implementation, and testing could be completed sequentially before moving on to the next. This method allowed the team to thoroughly define and document requirements upfront, which was ideal for a project with a relatively fixed scope. The lack of frequent iteration and changing requirements made Waterfall an appropriate choice, allowing the team to focus on completing each phase before progressing. The structured process ensured that each task was well-organised and delivered on schedule, providing a systematic way to handle development

Discord

Discord played a key role in communication and task management. Its "react" feature allowed team members to quickly assign themselves to tasks, and the flexibility of Scrum was well supported by Discord's messaging system. Changes to sprints could be communicated instantly via dedicated channels, keeping everyone aligned with deadlines and goals. Most team members were frequently active on Discord, making it an effective tool for quickly addressing urgent issues and fostering collaboration in real time.

LibGDX

The team chose LibGDX as the game engine due to its popularity in Java-based game development and the wealth of online resources available. Since the team had no prior experience with Java game engines, the support and tutorials for LibGDX helped them

quickly overcome initial learning hurdles. Its widespread use also made it easier for other teams familiar with the engine to pick up the project in the future. Additionally, LibGDX abstracts much of the complexity of OpenGL, which was particularly useful given the team's limited experience with graphics programming. Its integration with Gradle for build management further streamlined the development process.

IntelliJ

Although most team members were familiar with Visual Studio Code, the majority opted to use IntelliJ for this project, particularly for working with LibGDX and managing the build process through Gradle. The decision to use IntelliJ was influenced by the fact that most instructional materials for LibGDX and Gradle recommended it as the preferred IDE.

GitHub

GitHub was selected to manage the codebase due to its user-friendly interface and familiarity among some team members. It offered flexibility, allowing access through both the web interface and the command line, preventing workflow conflicts. GitHub's tracking of branches, forks, and code updates helped the team stay informed about project progress and understand parts of the code they weren't directly involved with. However, the team noted some challenges, such as difficulties in renaming or deleting files, which required additional coordination.

4b:

In the Waterfall methodology, each team member was assigned a specific role to ensure a clear and organised approach to development. Roles were distributed as follows: some team members focused on game development, working directly within the LibGDX engine to build core gameplay features, UI elements, and ensure compatibility with the designated operating system. Another group concentrated on architectural design and documentation, capturing and maintaining a record of the system's structure as it evolved through each phase.

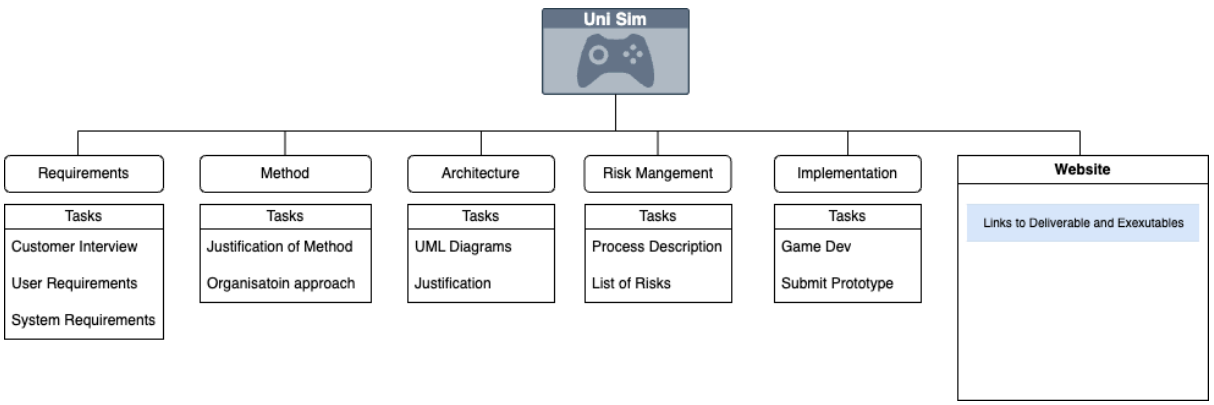
Additionally, a dedicated team was responsible for methodology and risk assessment, ensuring that each phase of the Waterfall process was followed accurately and evaluating potential risks associated with each stage. This group worked to identify and mitigate any issues that could impact the project timeline or quality, ensuring that the team remained aligned with project goals.

Throughout the development process, communication and task tracking were facilitated through Discord. Tasks were posted on the server, where members could indicate progress, and adjustments were made if additional support was required for certain areas. This structure allowed each team member to focus on their designated area, supporting a clear, linear workflow that aligned well with the Waterfall methodology's emphasis on completing one phase before progressing to the next.

4c:

Task#	Title	Start	End	Dependencies	Priority
Task1	Requirements	25/10/2024	05/11/2024	N/A	2
Task 2	Method Selection and Planning	01/11/2024	05/11/2024	N/A	3
Task 3	Architecture	22/10/2024	08/11/2024	Task 1	2
Task 4	Risk Assessment	25/10/2024	30/10/2024	N/A	3
Task 5	Implementation	20/10/2024	08/11/2024	Task 1, Task 3	2
Task 6	Website	06/11/2024	08/11/2024	Task 1-Task 5	3

Outline of the important tasks:



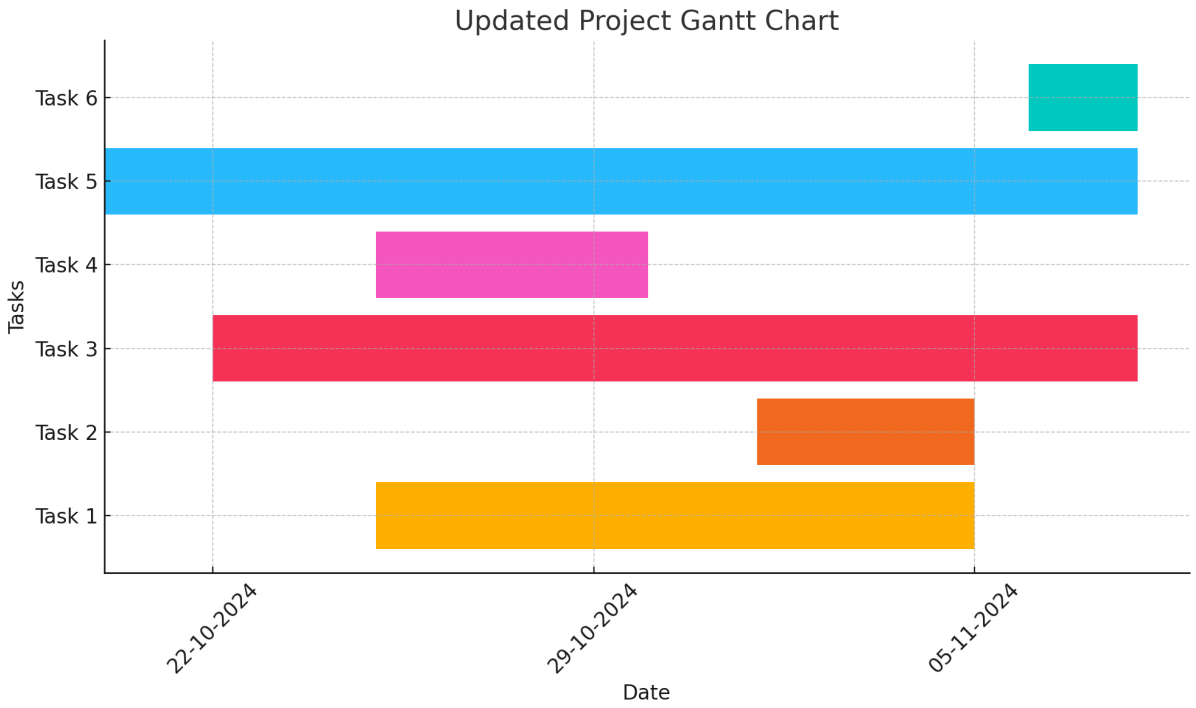
Deadlines for each deliverable:

	Title	Brief	Due Date
1	Website	The website will provide links to all deliverable documents, the executable, and updates on the progression of our project plan.	08/11/2024
2	Requirements	We will explain how our requirements were gathered and refined based on the project brief and an initial customer meeting. This section will cover both user and system requirements.	05/11/2024
3	Architecture	Diagrammatic representations will be included, featuring structural and behavioural diagrams to visually support the design and functionality of our system.	08/11/2024
4	Method selection and planning	We will outline and justify the software engineering methods applied, along with our organisational approach. This will include a systematic plan detailing all key tasks, their timelines, and expected completion dates..	05/11/2024
5	Risk Assessment	The risk management process will be described and justified, accompanied by a risk register documenting potential risks associated with the project.	30/11/2024
6	Implementation	A working implementation will be provided with documented code. Additionally, a list of all libraries and assets used will be included, with appropriate licences indicated for each.	08/11/2024

Milestones:

	Title	Brief	Due Date
1	Ready to program	Complete Deliverables Deliverables 2 to Deliverables 5	30/11/2024
2	Finish Implementation	Finalise the Implementation for Assessment 1 with a Functional Executable	09/11/2024
3	Finish Assessment 1	Prepare Assessment 1 for Full Accessibility and Information Sharing	11/11/2024

Initial Gantt chart showing when we need to complete tasks:


Updated Project Gantt Chart
Task 6
Task 5
Task 4
Task 3
Task 2
Task 1
Tasks
22-10-2024
29-10-2024
05-11-2024
Date