METHOD SELECTION AND PLANNING

Group 3

Liam Martin

Aaliya Williams

Lucy Crabtree

Kai Nichol

Sammy Hori

Tim Gorst

Zac Ribbins

4A.

SOFTWARE ENGINEERING METHODOLOGY:

The Agile methodology involves iterative mini-increments to add functionality. It provides flexibility and adaptability by encouraging continuous collaboration between different teams such as designers, developers, testers, analysts, etc. Keeping customer satisfaction a priority also requires frequent customer interaction and feedback**.**

To ensure thoroughness, we looked at having an iterative approach to phases, which was the strength of the Agile methodology. For example, phases like 'Risk Assessment' were updated regularly after assessing the risks in each stage.

Requirements → Risk Assessment → Method Selection → Risk Assessment → Architecture → Risk Assessment → Implementation → Risk Assessment

DEVELOPMENT TOOLS:

1) LibGDX -In our quest to develop a 2D game with Java-based technology, LibGDX emerged as the optimal choice among game engines. Its versatility, performance, and extensive feature set make it a standout option for game development projects like ours. LibGDX provides comprehensive support for 2D graphics rendering, input handling, and more, empowering us to create immersive gaming experiences across the web. [1]

2) Tiled - Tiled is our go-to tool for managing tile-based maps in our game development process. With its user-friendly interface and powerful features, it simplifies the creation and editing of tilesets, tile layers, and object layers, allowing us to design intricate game environments with ease. Its support for various map formats, including JSON and XML, ensures compatibility with our game engine, LibGDX. Also, Tiled offers extensive customisation options, enabling us to define properties and behaviours for tiles and objects, such as collision detection and animation triggers. This flexibility is crucial for implementing complex gameplay mechanics and enhancing player interaction within our game world. [2]

3) IntelliJ- For our development environment, we opted for IntelliJ IDEA, primarily because of its robust features tailored for Java development. It offers seamless integration with various tools and frameworks commonly used in Java development which streamlines our workflow and enhances productivity. Additionally, its intuitive user interface and extensive plugin ecosystem provide us with flexibility and customisation options to suit our project's specific requirements.

COLLABORATION TOOLS:

1) Communication:

   Our main methods of communication are regular in-person meetings and WhatsApp, as well as a discord server, where more of the technical discussions took place, and we received automated messages when something changed on GitHub or Trello.

2) Organisation:

   Trello - In order to keep track of what needed to be done, as well as progress, we used Trello, a web-based Kanban-style list-making application, that helped us to keep track of where in the project we were, what needed to be done and in what order, and who was responsible for what. [3]

   Plant UML - To organise subtasks and visualise our long-term plan, we proposed the use of Gantt charts. We chose Plant UML to execute this as this software is open source and has a text-based syntax, which makes it easy to learn and update. It is platform-independent and is easy to integrate into documents, which further strengthens our ability to collaborate using it. [4]

   Google Drive - For rough diagramming, general planning, documentation and organisation, we relied on the most convenient tool - a shared Google Drive. Being students, we already had access to this resource and were fairly familiar with the system. Further, it accommodates various types of files, which worked well with our requirements.

3) Version Control:

   For continuous integration and collaboration, we chose to use the most popular tool, GitHub. GitHub is one of the most widely used tools, thus having a well-established base to learn from and to acquire help if required. It has a user-friendly interface making it easy for developers of different levels to use. Even though the team was divided on their familiarity with GitHub, seeing its utility and features, we decided on it being the best choice for the project.

4B.

We first created the base for the website, whilst waiting on client meetings and requirements.

We then organised meeting times. The team is set to have regular weekly meetings every Thursday between 9:30 and 11:30 a.m. Following the methodology, the team discusses progress made in each subfield, and any challenges faced and shares critical feedback on each other's work. Additional meetings take place on Tuesdays with varied timings, depending on the tasks assigned for each week.

At the start of the project, we decided to split up every deliverable task amongst the 7 of us, such that everybody gets the opportunity to learn and contribute to every field. However, we

soon realised that it was time-consuming and unproductive to do so. Each task was then divided amongst the members playing to their strengths and choices.

We used Trello to break down big deliverables into smaller subtasks. These sub-tasks are assigned to individual members, who can then place them in either ‘to do’, ‘in progress’ or ‘completed’. This helps us track the team’s progress and efficiently organise our utilities. We made Gantt charts to organise subtasks and visualise our long-term plan.

In terms of implementation, we used GitHub for collaboration. Each member of the development team created a branch for each feature they were working on, which once they were tested and polished, were merged into the main branch.

In this manner, each of us got a fair share of the workload and responsibility of the project, which played to their own individual strengths.

4C.

The project commenced with the Method Selection and Planning phase during its first week. This initial planning involved assigning deliverables to team members based on their interests and strengths. Website and Risk Management tasks were also initiated.

1. Method Selection and Planning (High Priority)
    - Start: 2024-02-19, End: 2024-03-15
    - Includes Research, Planning, Gantt Chart Creation
    - Dependencies: None
2. Website Creation (Medium Priority)
    - Start: 2024-02-24, End: 2024-03-01
    - Includes Domain creation on Github, Website Design
    - Dependencies: None
3. Risk Management (Medium Priority)
    - Start: 2024-02-25, End: 2024-03-15
    - Includes Introduction, Writing Risks, Documentation
    - Dependencies: Product Brief

In the second week, the Requirement phase began. Simultaneously, Risk Management progressed from the previous week and Website Creation was completed with the exception of links which would be added later on. After gathering the requirements, the team introduced the software Jira, leading to the addition of crucial sub-tasks.

1. Requirement Collection (High Priority)
    - Start: 2024-03-06, End: 2024-03-19
    - Includes Customer Interviews, Writing User/System Requirements,
    - Dependencies: None
2. Implementation (Highest Priority)
    - Start: 2024-02-28, End: 2024-04-11

- Includes Menu Creation, Map/Navigation Design, Character Design/Movement, Collision Detection, Interior Design, Map Transitions
- Dependencies: Requirement Collection

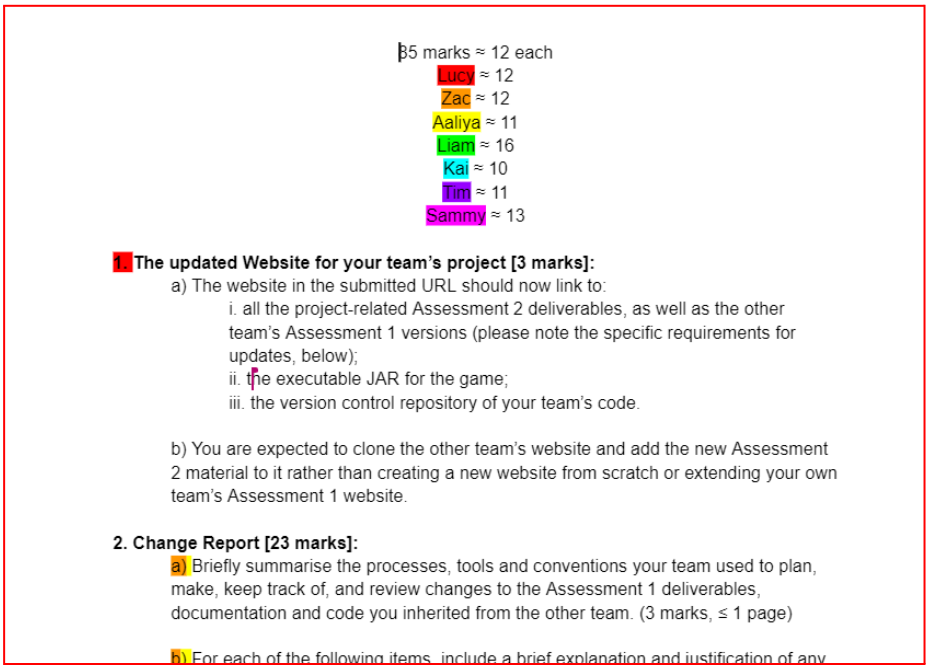
During the third week, the requirement phase was completed. The Implementation phase progressed. However, the "Architecture" phase, initially planned to commence earlier, was slightly delayed due to the priorities of other project components.

During the fourth week, the Implementation phase continued with tasks like integration. The Architecture phase finally commenced.

1. Architecture (High Priority)
    - Start: 2024-03-10, End: 2024-03-21
    - Includes System Design, Database Design
    - Dependencies: None

The fifth and final week was dedicated to addressing any remaining tasks or loose ends. While the core tasks and their order remained largely unchanged, task durations, start and end dates and resource allocations were adapted based on the project's evolution.

Following on from this, as the new team we wanted to hit the ground running, so once we got hold of the project in week 7, we immediately split all the new tasks into chunks for each person to do.



35 marks ≈ 12 each
Lucy ≈ 12
Zac ≈ 12
Aaliya ≈ 11
Liam ≈ 16
Kai ≈ 10
Tim ≈ 11
Sammy ≈ 13

1. The updated Website for your team's project [3 marks]:

a) The website in the submitted URL should now link to:

i. all the project-related Assessment 2 deliverables, as well as the other team's Assessment 1 versions (please note the specific requirements for updates, below);
ii. the executable JAR for the game;
iii. the version control repository of your team's code.

b) You are expected to clone the other team's website and add the new Assessment 2 material to it rather than creating a new website from scratch or extending your own team's Assessment 1 website.

2. Change Report [23 marks]:

a) Briefly summarise the processes, tools and conventions your team used to plan, make, keep track of, and review changes to the Assessment 1 deliverables, documentation and code you inherited from the other team. (3 marks, ≤ 1 page)

b) For each of the following items, include a brief explanation and justification of any

We used a combination of Trello, as well as Google Docs so that we could copy the new exam word for word, and break down who was responsible for doing what (screenshot below).

Once this was done, we quickly got working on our individual tasks simultaneously, and each week in the in-person session we checked in with one another to make sure everyone was on track. Below is our initial task plan:

1. Implementation (High Priority)
    - Start: 2024-04-19, End: 2024-05-15
2. Software Testing Report (Medium Priority)
    - Start: 2024-05-13, End: 2024-05-17
3. Change Report (Medium Priority)
    - Start: 2024-04-24, End: 2024-05-19
4. User Evaluation Report (Low Priority)
    - Start: 2024-05-10, End: 2024-05-20
5. Continuous Integration Report
    - Start: 2024-04-25, End: 2024-05-01
6. Website (Low Priority)
    - Start: 2024-04-25, End: 2024-05-20

Weeks 8-11 continued working on our tasks in the session, and once timetabled sessions ended from Week 12, we started regular check-in sessions using Discord.
The top priority was updating the code to meet the new requirements, which was finished in week 12, and from there we began user testing and were able to start finalising the updated deliverables documents and start work on the change report.

REFERENCES:

[1] libGDX, “LibGDX,” *LibGDX*, 2024. https://libgdx.com/

[2] Tiled, “Tiled,” *Tiled*, 2023. https://www.mapeditor.org/

[3] Trello, “Trello,” *trello.com*, 2023. https://trello.com/home

[4] PlantUML, “PlantUML at a Glance,” *PlantUML.com*, 2024. https://plantuml.com/