

Method Selection and Planning

Cohort 3 Team 4

Kiran Kang

Hannah Rooke

Ben Slater

Abualhassan Alrady

Cassie Dalrymple

Charles MacLeod

Dash Ratcliffe

Harley Donger

Software Engineering Method

Our team took an agile and scrum iterative development approach, as this is a flexible method which allows us to gain continuous feedback from the client to make sure our project is hitting targets throughout the course of its creation. This is done through sprints, where we provide small parts of the system at a time, allowing the customer to track its progress. Our main goal is to have a new playable piece of the game at each weekly meeting and then update our customer and get regular feedback to ensure we're building the right game that suits their needs. The reason why we chose an agile methodology is because it allows the project to adapt to the change, in the case of varying requirements from the client. It also helps to improve collaboration throughout the team, as we would need to regularly discuss the feedback we receive, as well as testing when we have reached certain levels of coding - this promotes communication with the team and the customer by using continuous feedback to produce a high quality game. We aim to deliver sprints weekly in order to verify we are on course. By creating a clear plan at the beginning of each sprint and collaborating to reach this, we will be able to create deliverables to present at the end of the sprints, which can be assessed by the client. The iterative repetition of this process is essential in creating appealing gameplay.

Development and Collaboration tools

Tool	Purpose	Justification
GitHub	Version control, collaboration and issue tracking	This is used to make sure all members will contribute and review code safely through branches and pull requests
GitHub projects	Organising tasks and managing each agile sprint	This is used to support the agile iterative approach where each week will be a sprint showing what issues need to be completed. This will be sorted in four sections: To do, In progress (game), In progress (documentation), Done.
IntelliJ	IDE for Java and supports LibGDX development	This is used to help write code more efficiently by using built in tools, debugging and launches LibGDX
plantUML	Designs UML diagrams to structure each class and how it processes also structures behaviour diagrams	This is used to create UML behaviour diagrams (such as timer, play movement, events and pause systems) and how they all relate to each other in a class diagram. PlantUML uses code to create these diagrams which can be integrated with IntelliJ so the diagrams can constantly evolve as the code updates.
draw.io	Designs the structural architecture design	This is used to show the main system architecture's relationships and its links between components (such as the physics, render and input system). This demonstrates how standard PCs can run on this structural architecture.
Google docs	Documentation and report writing	This is used to keep track of most recent edits, members can simultaneously work on the same document.

Why do these tools fit?

All of these tools support all stages of the agile and scrum development approach since we are able to plan and design architecture using PlantUML and draw.io, have version control and collaboration using IntelliJ and GitHub and be able to document all of this on Google Docs. It makes sure that the team can effectively plan, constantly refine the game and record all of the goals and progress that aligns with the scrum's iterative development.

Alternatives Considered

For the IDE, another tool that the team was about to use was **Eclipse** as it was also free and open source like IntelliJ. It also provides debugging, has built in tools and does support LibGDX. However, it does not contain any built in gradle support, which our game relies on for it to be launched. This makes setting up LibGDX more complex and time consuming. In addition to this, IntelliJ is more efficient when integrating to GitHub as it does not require any additional plugins whereas Eclipse does provide additional plugins.

Trello was another task management tool that was going to be considered instead of GitHub projects since it can easily be assigned and collaborated with the team. However, the reason why this tool wasn't chosen is because Trello works outside GitHub meaning that it would have to be updated manually and will no longer synchronise with the weekly sprint iterations. In addition to this, all members are unfamiliar with this tool hence we would waste more time trying to learn how to use the tool. Whereas for GitHub, team members familiar with this tool mean we'll save time as it automatically links all issues, branches and commits together which will give a clear view of how development has been progressing.

Team Organisation

Our team followed a set of roles and responsibilities that overlapped with each other. Each member had a main focus but also collaborated on other tasks.

Role	Responsibilities	Assigned to:
Project lead	Oversees all progress for each week. assigns tasks, sets deadlines and makes sure each agile sprint is completed in time.	Dash
Librarian	Make sure all GitHub repositories are maintained and are consistent. Supports organisation and documentation.	Cassie
Art Lead	Design the map art and any assets that can't be found online.	Harley
Secretary	Responsible for writing up the weekly meetings (MINUTES.md) by documenting discussions and tracked decisions, progress and new actions.	Hannah
Report editor	Make sure all documentation work is structured and organised so there will be consistency throughout the project.	Kiran
Meeting Chair	Organises team meetings and leads them by managing the decisions and agendas in the meeting.	Abualhasan
Requirements Lead	Manages the REQUIREMENTS.md file on GitHub, makes sure that all functional and non functional requirements are defined, traceable and updated.	Charles
Design Lead	Led the creation of the implementation, adding sprites, finding tile assets, and code. Assigns tasks to other members who are also doing implementation.	Ben

How does our agile development link with our roles?

The Meeting Chair Abualhassan will plan the sprint/review meetings. At the start of each meeting, the Requirements Lead Charles will present the high priority features from the REQUIREMENTS.md and translate this into actionable items. They will talk to the lecturer to get feedback from the req1.pdf and once implementation starts. Dash the Project Lead will assist the team in using the Scrum effectively by planning the sprints, coordinating communication between members and making sure the team will follow the schedule.

Members then decide how much work we can realistically complete within the weekly sprint. The Design Lead Ben will be responsible for implementation by delegating tasks to members and setting up the LibGDX game's repository in GitHub. In addition to this, Harley, our Art Lead, is in charge of the map's design. Harley will use the REQUIREMENTS.md to get an idea of the map's design. The Librarian, Cassie will make sure everything works smoothly by tracking/merging the branches on GitHub and ensure the product's backlog is up to date. During the meeting, all discussions will be documented by Secretary Hannah as a summary which will contain: meeting attendance, agenda and a description of the current progress. The Report Editor, Kiran will edit the documentation to make sure the sprint, implementation and progress reports are all up to date. At the end of each sprint, we should have an updated version of the working game and with new documentation.

Systematic Plan for Project

Documentation & Coding Tasks

Task	Start/End date	Priority & Why	Dependencies	Assignment
Project Setup and Methodology	01/10/2025 - 01/10/2025	High: Foundational setup defining roles & workflow before other work could begin.	None	Everyone
Requirements Definition and Risk Assessment	08/10/2025 - 15/10/2025	High: Defined project scope and goals (essential for all subsequent tasks).	Client interview, divide team into roles.	Dash,Hannah,Kiran
Create initial website.	09/10/2025 - 15/10/2025	Medium: Important for later, but didn't block development immediately either.	Divide the team into roles.	Abualhassan
Develop system design document, including architecture, classes, overflow.	09/10/2025 - 05/11/2025	High: Blueprint for all the coding work. We didn't explicitly assign implementation tasks until this was sufficiently complete.	Start work on requirements.md	Hannah
Write up research on a plan for how to run things.	15/10/2025 - 23/10/2025	Medium: Translated research (game engine selection) into an actionable plan.	Research suitable game engines.	Cassie
Document third-party tools , assets with licenses.	15/10/2025 - 05/11/2025	Medium: Important for legal compliance and our future reference.	Choose asset pack/tileset	Everyone
Write justifications for design and implementation decisions.	15/10/2025 - 05/11/2025	Medium: Provided rationale behind key choices which was crucial for the final report and handover to the next team.	Develop system design document, all implementation tasks.	Kiran
Formally write up the systematic plan for the project on a weekly basis.	09/10/2025 - 09/11/2025	Low: While important for tracking, this was an administrative task that documented progress.	All other tasks (that week)	Cassie/Hannah
Ensure website structure is clear and provides easy access to documentation.	15/10/2025 - 09/11/2025	Low: Usability and refinement task. Important for final presentation but didn't block other development work.	Create initial website	Abualhassan
Finish off final tasks for all documentation required for assessment 1	23/10/2025 - 09/11/2025	High: Final document summarising the entire project. Its completion depended on all other tasks being complete.	All other tasks.	Everyone
Researching and selecting a suitable game engine.	01/10/2025 - 15/10/2025	Critical: Foundational decision that all implementation depended on.	None	Everyone
Creating prototype using libGDX to structure the project.	15/10/2025 - 23/10/2025	High: Prototype was needed to familiarise everyone with libGDX and establish a foundational structure for all later tasks.	Researching and selecting game engines.	Ben
Player Character (Movement, Collision, Animation)	23/10/2025 - 30/10/2025	High: Core gameplay mechanic. The game is unplayable without a player.	Creating a map prototype.	Ben,Dash,Abualhassan
Map Design and Implementation	15/10/2025 - 30/10/2025	High: The game's main environment. The game is unplayable without it.	Client feedback from email (23/10/2025)	Dash, Ben
Theme and Asset Selection	08/10/2025 - 15/10/2025	High: Key decision before any visual implementation/map design could start.	Client interview.	Everyone
Event System (Design and	30/10/2025 -	High: A specific and complex core	Base game mechanics,	Ben,

Implementation)	09/11/2025	requirement from the client that defined much of the interactive gameplay.	client interview	Charles, Abualhassan
Implementing pause and resume functionality.	05/11/2025 - 05/11/2025	Medium: Key usability feature, but core gameplay was the priority.	Basic game and timer implementation.	Dash
Game Rules and UI (Timer, Win/Loss conditions, scoring).	15/10/2025 - 06/11/2025	High: Defined the game's objectives and provided essential player feedback (timer, scores). Core to making it a 'game'.	Client interview	Hannah, Dash, Charles
Integrating assets (textures, models, sounds).	15/10/2025 - 09/11/2025	High: Necessary to build visual and sound components of the game.	Choosing an asset pack& music/sounds.	Everyone
Designing a data structure to store event information and player state.	23/10/2025 - 30/10/2025	High: Critical task to ensure game data is managed efficiently and can be easily accessed.	Understanding client requirements for events and player stats/info.	Cassie, Ben
Testing and Quality Assurance	23/11/2025 - 09/11/2025	Medium: A quality assurance task performed after the feature was built.	Implementing the scoring system.	Everyone
Implementing a counter for tracking interactions with each event type.	23/10/2025 - 23/10/2025	High: Core requirement linked to the timer and scoring systems.	Implementing the event system.	Ben
Defining difficulty levels with at least an easy mode.	05/11/2025 - 06/11/2025	Medium: A design decision to improve accessibility but not the main priority. Game functionality was more important.	Client interview, functioning initial game.	Dash
Creating respawn points (e.g. restart game).	07/11/2025 - 07/11/2025	Low: A feature that depended on other game mechanics and wasn't a key requirement, more of a 'nice to have'.	Defining win/loss conditions, pause/resume menu.	Dash
Making sure all code is properly commented so other teams can pick up the game easily.	15/10/2025 - 09/11/2025	High: Project's goal was to have a successful handover to a future team, so this was key to ensure other teams would want to pick our game.	Client interview and requirements analysis.	Everyone (mainly Cassie)
Choosing and implementing music/sounds	07/11/2025 - 07/11/2025	Low: Game didn't require music/sound effects, but it made it feel more complete and immersive.	Choosing game theme, movement (for running sound).	Dash

Plan evolution throughout the duration of the project

Our plan changed and adapted as we progressed through the project. At the start, the foundations are laid out: sorting out communication, organising GitHub, and assigning roles. The first turning point was the client interview on October 8th, which allowed us to create requirements.md file and properly define the project's scope. After this, we established our adapted agile/scrum framework and made the key decision to plan out the system architecture before starting any implementation to avoid having to redo work later. This planning phase moved into active development after we chose our asset packs on October 15th, which is when we started assigning specific coding tasks like player movement and the initial map design. As development went on, our plan proved to be flexible. An example was when we tweaked the map design to be a simpler, single-page layout based on client feedback we got on October 23rd, as this would be easier for a future team to work with. In the final weeks, our focus shifted from building new features to completing, testing, and polishing what we already had. Our final meeting on November 5th identified what was left to do and reallocated team members to finish everything off. People who had finished their main tasks moved over to help with other things to ensure we delivered a complete and polished prototype by the deadline.