Method Selection and Planning

Cohort 2 Team 1(Assessment1)

Ahmet Abdulhamit
Zoey Ahmed
Tomisin Bankole
Alanah Bell
Sasha Heer
Oscar Meadowcroft
Alric Thilak

Cohort 2 Team 2 (Assessment 2)

Bader Albeadeeni
Dan Hemsley
Jennifer Bryant
Mathilde Couturier-Dale
Oliver Elliott
Rosie-Mae Connolly
William Mutch

Chosen method and justification:

What development method are we using:

- We want to use an agile methodology. We believe that Scrum is the most suited for our task.

Justification for why it suits our project:

- An agile methodology means we can take an iterative approach to our project. It will allow us to produce software in increments that prioritises collaboration between team members and the customer and is continuously reviewed. This method will focus on developing the product through repeated iterations. We also included Continuous Integration and Continuous Delivery principles. This was to make sure there is consistent development. We think this is a good approach for our project which should evolve through iterations.

  Scrum is a framework based on Agile. It assigns roles and uses sprints to put development into iterations. Each sprint begins with a planning meeting to decide which tasks to complete. They end with a review to consider team progress and improvements. We believe this is essential in a team project to maintain accountability and also momentum.

  The 'Escape from Uni' game has evolving requirements and requires creativity. We believe a plan driven method such as Waterfall which requires finalising designs and requirements before implementation would be unrealistic for a game which depends on iterative improvement. Instead, an Agile approach with Scrum will let us refine builds by adapting to feedback and technical challenges. This will reduce the risk of late-stage failure and help with time constraints. Scrum's sprints produce small functional versions of the game, which can be regularly improved. Ultimately this matches the assessment requirements to build a working prototype and expand on it in assessment 2. Scrum practices will help our team with incremental integration of features and its focus on adaptability will allow the team to quickly change priorities based on feedback. Essentially, Scrums structure is practical for a small team to provide organisation while avoiding excessive documentation. This is a great fit for our project.

Methodology extension

We continued the same methodology approach as the previous team using a Scrum approach to the second part of the project. Continuing with the same methodology led to an easy transition with us

taking over the project. The Scrum methodology focuses on constant reviews and creating many iterations with feedback from the client. This made it easy for us to develop the project as we could understand the current state of the project and what tasks had to be developed. Our one-week work cycles act as Scrum sprints starting with Scrum planning where delayed tasks are identified and ending with a review to review the current progress on the project.

Alternatives considered - We considered continuing with the Waterfall-Hybrid approach that we used for assessment 1 where due to dependencies between the assessment deliverables and clear waterfall phases would mean we have clear milestones. However, we decided against this as due to assessment 2 having a larger focus on implementation we thought it would be better to be more flexible and not have any fixed phases like a waterfall approach. Also, it would be beneficial to change the architecture while we are developing the game as new challenges and design improvements may be needed during implementation. Therefore, a more iterative approach allows us to make better incremental decisions and reduce the risk of time consuming reworking later on in the development cycle.

Agiles 4 manifesto principles:

- The Agile Manifesto (2001) defines 4 key values that also guide Scrum. They shape how teams prioritise work, make decisions and work together during the development lifecycle.

1: Individuals and Interactions over processes and tools.
- Emphasises effective communication and collaboration between team members.
- Specifically for 'Escape from Uni' this means checking in often, team discussions and communication to address design/ coding issues.

2: Working software over significant documentation.
- The priority is delivering a functional product, not focusing on documentation. Documentation should be light, simply enough for traceability and understanding.
- For our project this means that the production of each sprint will be prioritised over producing detailed documentation/ reports.

3: Customer collaboration focus rather than contract negotiation.
- Teams engage the customer throughout the development. This is to refine requirements based on feedback.
- We should continuously revisit requirements.

4: Responding to change over following a plan.
- Agile believes that plans will evolve as new issues/ insights come to light. Teams should re-prioritise tasks and change goals instead of keeping to a schedule.
- This is essential for our creative, experimental game. Our project has features and mechanics that will require revision as testing progresses.

These principles mean we should ensure we frequently communicate (daily).
It also means that we should produce sprints which aim to produce a working prototype or new feature. Not significant reports.
The lecturers and testers are the ongoing stakeholders.

Tools used and justification:

We chose a combination of tools which would support our scrum workflow, focusing on making collaboration and iteration efficient. GitHub was chosen to manage all the code and documentation. This allows version control and continuous integration. GitHubs branching and pull-request features also help us merge updates and track progress across sprints. PlantUML is being used to create UML diagrams, making sure that diagrams evolve with the code. We also used LucidChart for UML component and sequence diagrams due to their ability to enable fast visual editing. This is transparent and helps the Scrum Master monitor the workload. We used Google Docs and WhatsApp for communication - this allowed quick updates and shared access to meeting notes and risk logs. Jekyll was used to develop and update the website.The tools we selected mean our work can align with Agile principles such as collaboration and adaptability while supporting iterations from continuous feedback. We do this while avoiding heavier platforms such as Jira which would make our process rigid.

Tools extension

We used similar tools to the previous team when we took over the project. For example, we used Git and Github to easily continue developing the project with commits and pull requests. We also used PlantUML to create Gantt charts and draw.io to create some of our structural and behaviour diagrams. LibGDX was used to develop the game, LibGDX is a free and open source game development framework that provides the core Java components for game making.

Team Organisation & Communication:

How do we divide the roles:

We tailored tasks to each individual's strengths and our structure adheres the principles of collaboration of Agile/ Scrum. We quantified workload by mark distribution and ensured each team member was given an equal amount. The tasks were matched to skill. Members who were more confident with coding were given implementation and UML architecture - others with strong communication and organisational skills are leading documentation, risk management and planning. We believe this is the most efficient and fair way. Roles are also not fixed, and responsibilities were reviewed and adapted as the project progressed. This meant team members could support each other in overlapping areas such as testing. Our structure shows our team's commitment to teamwork and improvement by iteration. It also shows our emphasis on accountability. This is better than rigid task boundaries.

Dividing roles extension

When we took over the project we also tailored tasks to individual strengths and split the workload evenly based on mark distribution. As this part of the project development had a heavy focus on implementation so we split the team in half, one half would focus on developing and the other half would focus on documentation and architecture. This allowed people who were better at developing to focus solely on implementation without needing to divert their attention to documentation and the same vice versa. However, to stay flexible in our approach this split was not rigid and some team members did tasks on developing and documentation.

Scrum roles:

Scrum Master - Alanah Bell

- Keeps the team organised. Led planning and progress meetings. Ensures communication stays clear and issues are quickly resolved.

Methodology and Risk Leads - Sasha Heer and Ahmet Abdulhamit - Managing the project plan and risk log. Tracks the team progress and guarantees we stay on track with the Agile process.

Architecture Leads - Alanah Bell, Tomisin Bankole and Sasha Heer - Designing the system structure and diagrams. Also responsible for refining these. They also ensure the code matches the architecture.

Developers - Alric Thilak, Alanah Bell and Oscar Meadowcroft

- Building and testing the game in LibGDX. Handling integration and fixing technical issues in each build.

Requirements and Website Coordinator (and general support) - Zoey Ahmed - Manages the requirements document and website. Keeps materials up to date and accessible.

Scrum roles extension

Documentation - Bader Albeadeeni, Oliver Elliott, Jennifer Bryant, William Mutch

Scrum Master and User Evaluation Lead - Bader Albeadeeni

- Ensures good communication between the team by planning meetings and leading discussions
- Manage and complete the user evaluation document
- Update the process and tools section of the change report

Testing Lead - Jennifer Bryant

- Update and manage the software testing report
- Complete the unit testing and integration testing and report on the results of these tests
- Complete the manual testing and report on any observation made after testing

Architect Lead and Risk management - William Mutch, Oliver Elliott

- Update and manage the architecture document
- Design system architecture diagrams (behavioral and structural) and refines them to make sure they match the code
- Update and manage the requirements document
- Update and manage risk assessment document
- Update and manage the change report document to reflect the changes made after taking over the project
- Present the final finished game to the clients

Developers - Rosie-Mae Connolly, Mathilde Couturier-Dale, Dan Hemsley:

Website and map developer - Rosie-Mae Connolly

- Manages and updates the website
- Builds and implements events into the game
- Develops the map and handles testing report for the map

Features developer - Mathilde Couturier-Dale

- Implements additional features into the game such as the leaderboard
- Fixes technical problems in each build
- Updates architecture and requirements change report

DevOps developer - Dan Hemsley

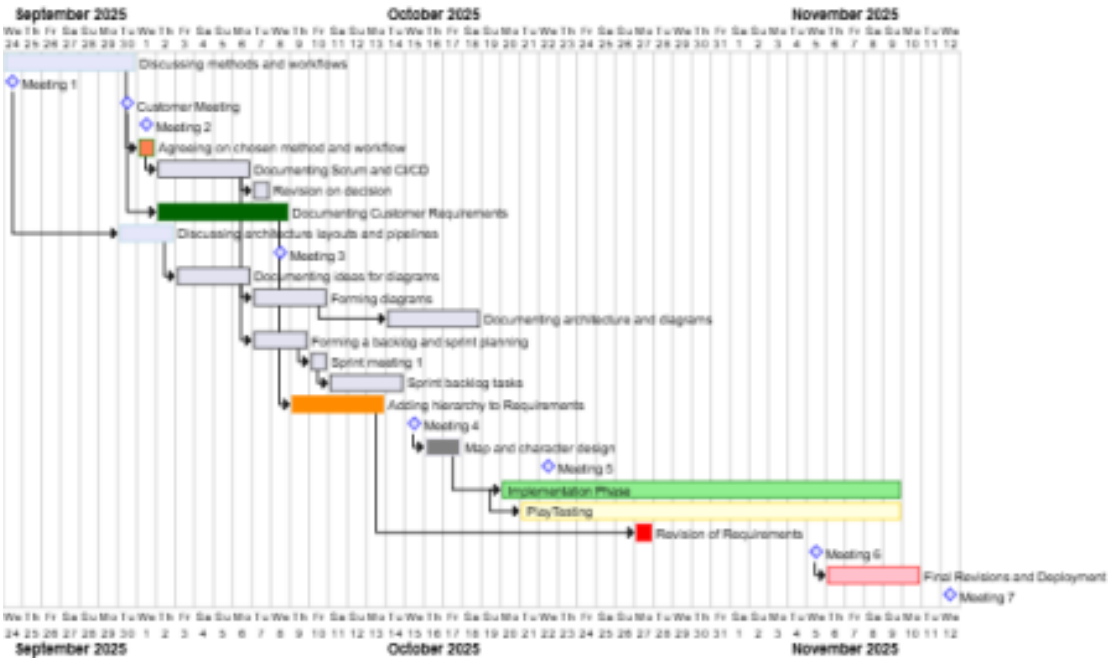
- Implements event logic into each build
- Manages and develops the continuous integration report
- Adds new events and classes and fixes bugs that arise from implementation

How often will we meet:

We have a timetabled weekly meeting between 9 and 11 on Wednesdays and for other matters or discussing other changes we will either call online or meet in person. We will aim for this alongside smaller task meetings which are focused on producing specific outputs.

Systematic Plan:

Timeline:



	Tasks	Date	Description	Priority	Dependency
1.	Discussing methods workflow	25/09/2025 - 01/10/2025	We prioritised this. This is how we all communicate and also figure out how to balance and manage our time.	High	No dependency
2.	Customer Meeting	30/09/2025	Important meeting. It told us what to implement.	High	Task 1
3.	Agreeing on chosen method and workflow	01/10/2025	We are familiar with SCRUM and CI/CD and they are a great fit for our project.	High	Tasks 1 and 2
4.	Discussing architecture layouts and pipelines	29/09/2025 - 02/10/2025	Discussing an effective implementation.	High	Task 3
5.	Documenting SCRUM and CI/CD	02/10/2025 - 06/10/2025	We discussed how we will use SCRUM. We broke our work down into manageable chunks which were revisited by both method leads to ensure a continuous improvement process.	Medium	Task 3
6.	Documenting Customer Requirements	02/10/2025 - 08/10/2025	We produced a clear document which explained and listed the requirements and constraints given by our customer.	High	Task 2
7.	Documenting ideas for diagrams	03/10/2025 - 06/10/2025	Brainstorming to form diagrams for implementation.	Medium	Task 4

8.	Forming diagrams	07/10/2025 - 10/10/2025	Forming ECS, OOP, etc. Diagrams to guide our implementation team in forming our game.	High	Task 7
9.	Forming a backlog and sprint planning	07/10/2025 - 09/10/2025	The backlog is our planning and methods, architecture and requirements documents.	High	Tasks 3,4 and 5
10.	Sprint Meeting	10/10/2025	This is one of our external meetings. We used it to check progress and manage risks.	High	Task 9
11.	Adding hierarchy to Requirements	09/10/2025 - 13/10/2025	Adding a hierarchy for our requirements to prioritise. This will ultimately aid us in implementation	Medium	Task 6
12.	Sprint backlog tasks	11/10/2025 - 14/10/2025	Our risk-management team checked team tasks and thought about risks that may disrupt planning. The team checked the progress to make sure we will finish by our deadline.	High	Task 9
13.	Documenting architecture and diagrams	14/10/2025 - 18/10/2025	Documenting architecture and diagrams.	Low	Task 8 and 4
14.	Map and Character design	17/10/2025 - 18/10/2025	Finishing the design of the map and characters. This meant our implementation team could start their work.	High	Task 13
15.	Implementation Phase and Playtesting	20/10/2025 - 09/11/2025	Continue implementing and playing/testing the game.	High	Task 14
16.	Revision of Requirements	27/10/2025	Check if we are working in accordance with our requirements.	Low	Task 15

17.	Final Revisions and Deployment	06/11/2025 - 10/11/2025	Finalise, revise and ensure our documents don't have any incorrect or missing information.	Low	Task 16
18.	Project takeover	19/11/2025 - 26/112025	Decided on which project to take over. Assigned people to the coding team and the documentation team.	High	No dependency
19.	Agreeing on roles	26/11/2025 - 3/12/2025	Decided on roles to split the team into coding and documentation. Began researching and developing the project.	Medium	Task 18
20.	Tutorial screen and diagonal movement	3/12/2025 - 10/12/2025	We added a tutorial screen and diagonal movement. The architectural diagrams began to be edited to add our additions.	Medium	Task 18 and 19
21.	Finish requirements and architecture, fix bugs	10/12/2025 - 17/12/2025	We added the additional requirements for the project and the extensions to the architecture. Fixed bugs in the game such as the leaderboard and the map (by creating a new map).	Low	Task 20
22.	Added new features and screens	17/12/2025 - 24/12/2025	Added 2 new screens, added new features (dean repellent), co-ordinated new events with new map. New event classes added to the requirements section.	High	Task 21
23.	Bug fixing and finishing documents	30/12/2025 - 10/01/2026	Finalise and complete our documentation for the project as well as finishing the game.	High	Task 22

This plan follows a logical, iterative Scrum workflow. This is evidenced by the dependencies in the table. Our early stages were focused on planning and later stages progressed into a focus on design and implementation (including testing). Our weekly sprint meetings meant our plan was continuously evolving, allowing the team to adapt their priorities.

Timeline extension

These additions to the timeline table reflect how we continued the iterative Scrum workflow after we took over the project. Differently to the previous team who worked on the project we planned and implemented simultaneously, as half the team was on documentation and the other half was on implementation. Similarly to the previous team we had sprint meetings weekly however closer to the end of the project these became more frequent. Further details on what was achieved each week and the main discussions of our meetings can be found in our weekly log: Weekly Log.