Method and Planning

Group 4 - THADJAM

Abdul Fofanah
Matthew Holleran
Toby Watchorn
Arwen Minton
Daneena Roydean
Jessica McKerill
Henry Bambrough

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 prioritise 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 believe this is a suitable approach for our project, which will 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, Scrum's structure is practical for a small team to provide organisation while avoiding excessive documentation. This is a great fit for our project.

Agile 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. For our project, we will focus on the key 3 of these.

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: Customer collaboration over contract negotiation.

- Teams engage the customer throughout the development. This is to refine requirements based on feedback.
- We should continuously revisit requirements.

3: 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.

4: Working software over significant documentation.
- This value we will focus on less because we believe that the key to a good project is a balance between documentation and implementation.
- For example, if one of these areas is lacking, it makes it hard for us to progress in the other, as the implementation informs the documentation and the documentation makes sure we are working towards the goals we set out.

These principles mean we should ensure we frequently communicate (daily). It also means that we should produce sprints that 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. GitHub's 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 its 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.

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.

Scrum roles:
Change report - Abdul, Jess, Daneena, Arwen, Henry:

- For this part, we decided that it was best to split it into sections we are most familiar with we decided this by giving each person the report that they worked on in Assessment 1, as they are already familiar with that section.

Implementation - Matthew, Abdul, Toby

- Again, for this part, we kept the people who worked on our implementation in Assessment 1, as they are already familiar with those requirements

System Testing (ST) report - Toby, Daneena, Henry

- These 3 people felt comfortable testing the code and giving reviews on it, also with one person who is working on the implementation of the code, feedback would be easier to give.

User Evaluation (UE) report and Continuous Integration CI report - Jess and Arwen

- These 2 felt more comfortable and more efficient at working alone, so they were given separate tasks to complete to maximise our efficiency.

Website - Matthew

- Assigned to design the website and make sure it's up to date with all its documentation and code.

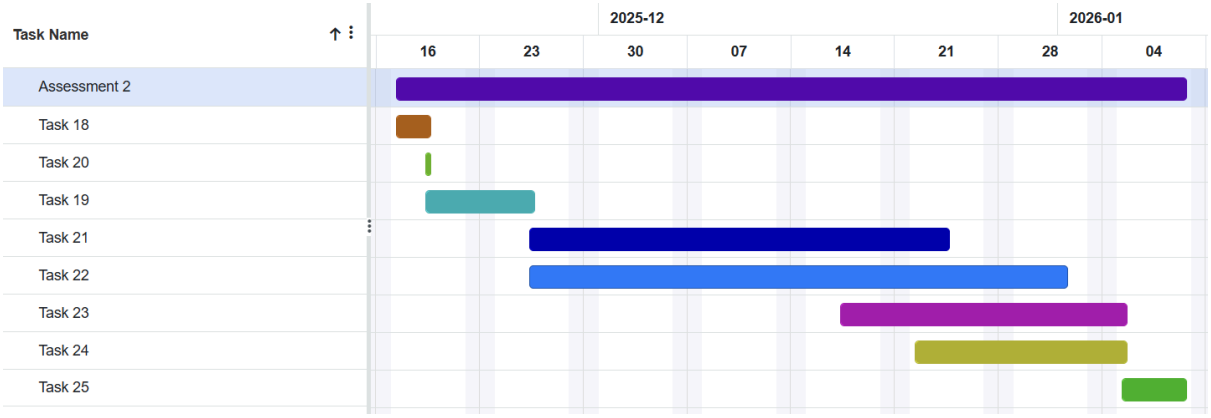
Scrum Master- Abdul

- Keeps the team organised and makes sure everyone is on task and on time with delivering their tasks.

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 the 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 the 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

	Tasks	Date	Description	Priority	Dependency
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	Tasks 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.	Receive Documents	17/11/2025- 19/11/2025	Getting documents to continue another group's project	High	Task 17
19.	Discussing requirements updates	19/11/2025- 26/11/2025	Deciding which requirements need updating to fit the new project scope.	High	Task 16 and 6
20.	Deciding team roles	19/11/2025	Choosing who is going to complete which tasks based on the project requirements.	High	Task 19
21.	Updating change reports	26/11/2025- 24/12/2025	Getting started on these early, allowing us to update them quickly, gives the people doing implementation a clearer goal.	Medium	Task 20
22.	Implementation and Testing	26/11/2025 01/01/2026	Started the same week to allow changes to previous documents to be made, to enable a clear goal to be formed.	High	Task 20 and 21
23.	User Evaluation	11/12/2025- 05/01/2026	Start receiving User evaluations and change requirements if needed to meet project scope.	High	Task 22

	Tasks	Date	Description	Priority	Dependency
24.	Continuous Integration	18/12/2025-05/01/2026	Start Continuous Integration to ensure code has no bugs and can give feedback to the development team.	High	Tasks 22 and 23
25.	Final check and Deployment	05/01/2026-09/01/2026	Finalise documents and code to ensure they meet the project requirements	Low	Tasks 22 and 23

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.