

ENG1 - Assessment 2

Method selection and planning

Plan2.pdf

Group 6

Freya Goodger	sg1967
Mikolaj Wyrzykowski	mw2179
Barnaby Matthews	bm1287
Cooper Love	cl2702
Oliver Cassey	oc854
Anna Hrynyshyn	ah2886
Oliver Thompson	ot699

Software Engineering Methods and Tools

In selecting our software engineering methods, our team meticulously evaluated various methodologies to ensure optimal project management and team collaboration. After careful consideration, we chose Agile Scrum as our framework due to its renowned flexibility and adaptability, which align well with the dynamic nature of our project. Agile Scrum's iterative approach allows us to respond swiftly to changing requirements and effectively prioritise tasks, ensuring incremental value delivery throughout the development process.

We opted for Agile Scrum over other methodologies such as eXtreme Programming (XP) and Waterfall due to its suitability for our project's scope and requirements. While XP emphasises coding and testing in short iterations, Agile Scrum provides a more comprehensive framework for managing the entire development lifecycle. Waterfall's sequential approach was deemed unsuitable for our project's evolving requirements and dynamic nature.

To support our Agile Scrum methodology, we carefully researched and selected a range of development and collaboration tools. GitHub serves as our primary version control and code collaboration platform, offering seamless collaboration and robust version tracking capabilities. Trello, with its Kanban board feature, facilitates task management and progress tracking, enabling us to visualise our workflow and prioritise tasks effectively. Discord serves as our communication channel for virtual collaboration, supplemented with bots to streamline project status updates and synchronise team efforts.

Although we considered alternative project management tools such as Kanbanchi Task and Project Management in Google Drive, we ultimately decided against their implementation. This decision was influenced by the seamless integration of Trello with Discord, which facilitates efficient communication and task management within our team.

In terms of project planning and visualisation, we utilise PlantUML for generating Gantt charts and Lucidchart for creating architecture diagrams. These tools enable collaborative visualisation and documentation, essential for effective communication and decision-making within the team. The use of PlantUML ensures that version history can be saved in GitHub, allowing easy access for all team members.

For game development, we employ LibGDX as our framework due to its robust tools and libraries for cross-platform game development. We utilise IntelliJ IDEA as our integrated development environment (IDE) for its comprehensive set of features and

tools, complemented by Liam's prior experience with it. While other game development libraries such as LWJGL were considered, LibGDX stood out for its versatility and suitability for our project's requirements. Similarly, IntelliJ IDEA's features and tools align well with our development needs, further justifying our choice.

Overall, our selection of software engineering methods and tools reflects a careful consideration of our project's requirements, team dynamics, and desired outcomes, ensuring efficient project management and successful collaboration.

Team Organisation

Our team operates with defined roles inspired by the Scrum framework [1], promoting effective collaboration and accountability throughout the project:

- **Development Team:** Sammy and Liam form the core development team, engaging in pair programming to ensure continuous code review and knowledge sharing. This approach enhances code quality and team cohesion, both in-person and virtually.
- **Product Owner (Sammy):** Sammy leads as the Product Owner, aligning the team's work with customer requirements and making final decisions on project direction.
- **Scrum Master (Lucy):** Lucy serves as the Scrum Master, enforcing Scrum principles and supporting the team in overcoming obstacles. With Trello's Kanban board, Lucy facilitates task prioritisation and deadline adherence, ensuring smooth project execution. Lucy also coordinates other project components, such as music and art.

While we allocated tasks on a weekly basis, we recognised the importance of remaining responsive to changing project needs and individual team dynamics. Our aim was to strike a balance between structured task allocation and the agility required to accommodate evolving requirements and team member capacities.

Within the Agile Scrum framework, task assignment is typically guided by the backlog of tasks prioritised by the Product Owner [2]. However, we adapted this approach to suit the specific needs of our project. Rather than strictly adhering to predefined task assignments for each sprint, we embraced adaptability in our task allocation process.

During our sprint planning meetings, we collaboratively identified and prioritised tasks based on the project's current objectives and priorities. While we aimed to allocate tasks for the upcoming week, we remained open to adjusting assignments based on emerging needs or insights gained during the sprint.

Our approach to task assignment was underpinned by the principles of transparency, inspection, and adaptation inherent in Agile Scrum [3]. We maintained clear communication channels within the team to ensure that everyone was aware of their roles and responsibilities. Regular sprint reviews and retrospectives allowed us to reflect on our progress, identify areas for improvement, and adapt our task allocation process accordingly.

Project Plan

Given that the project is being undertaken by a small team, located on site, with a product that is both small and novel to the developers, extensive upfront planning is counter productive, instead the team aim to produce smaller plans for each iteration of the software as the team learns more about the tasks they will face as they progress. As such the tasks and deliverables may have multiple revisions which will be reflected in their IDs [[project breakdown.png](#)]:

W3 Tasks

ID	Description	Dependencies	Due Date
T1	Setup environment		29/02
T2	Draw Sprites		07/03
T3	Display map and locations	T2,T1	09/03
T4	Import Bitmap of sprites	T2,T1	10/03
T5	Add music	T1	14/03
T6	Implement movement	T3,T4	14/03
T7	Implement bounding boxes	T3,T4	16/03
T8	Implement interactions with locations	T6,T7	17/03
T9	Implement score counter	T8	17/03
T10	Implement day counter		16/03
T11	Implement Ending	T9,T10	18/03

Deliverables:

ID	Name	Tasks	Description
D1	Codebase	<ul style="list-style-type: none"> - Setup environment - Display map and locations on screen - Import Bitmap of sprites - Bounding Boxes functionality - Implement movement for character - Implement interactions between character and locations - Implement score counter - Implement Day counter - Implement Music - Implement Ending 	The game itself
D2	Architecture	<ul style="list-style-type: none"> - Create Use case Diagram - CRC cards - Create Sequence Diagrams - UML class diagrams 	Document containing the overview of the structure of the code
D3	Requirements	<ul style="list-style-type: none"> - Create Requirements elicitation system - User Requirements - System Requirements - Requirements referencing system 	Document containing the requirements
D4	Risk Assessment	<ul style="list-style-type: none"> - Compilation of risks - Adding mitigation and ownership - Outline risk management process 	Document containing anticipated risks.
D5	Method Selection & Planning	<ul style="list-style-type: none"> - Ganitt Chart - Work Breakdown structure - Weekly snapshots - Roles and responsibilities - Resources to be used 	Document containing plans for the project.

Milestones:

ID	Description	Due Date
M1	Game Finished	21/03
M2	Art work finished	14/03
M3	All deliverables finished	21/03

Work Packages:

ID	Description
W1	Requirements elicitation from stakeholder
W2	Design
W3	Implementation
W4	Testing

W2 Tasks

ID	Description	Dependencies	Due Date
T1	Create Website		21/03
T2	Create Use Case diagrams		07/03
T3	Create CRC cards	T2	12/03
T4	Create Sequence Diagrams	T2	12/03
T5	Create UML class diagrams	T3	14/03

W1 Tasks

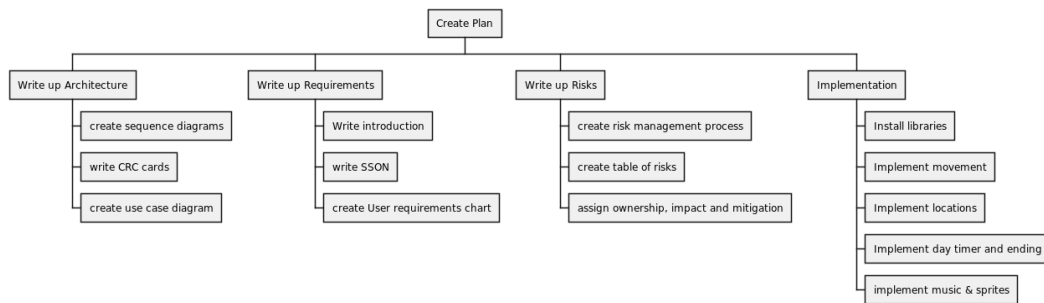
ID	Description	Dependencies	Due Date
T1	Draft Interview questions		27/02
T2	Customer meeting	T1	28/02
T3	Write up user requirements	T2,T4	07/03
T4	Write up user requirements method		21/03

W4 Tasks

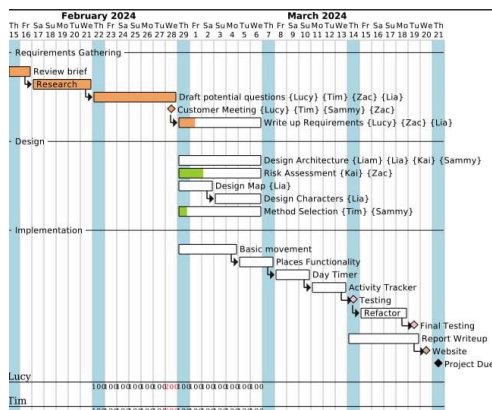
ID	Description	Dependencies	Due Date
T1	Decide which features require testing		14/03
T2	Design and carry out tests	T1	16/03
T3	Evaluate results and design fixes	T3	18/03

(Enlarged images linked & on website)

Work Breakdown Structure [[work structure.png](#)]:



We'll use the issue tracker on github to organise rolling tasks and issues that will then be assigned to team members on the trello board. In this way the project can be micro-planned through the various stages of the agile process.



In order to mitigate the risk of a time overrun, all the critical parts of the project, the deliverables, have a float of at least a week. Namely two weeks for architecture, risk assessment, method selection and one week for implementation. For every task shown on the Gantt chart the first team member listed is the team member assigned responsibility for that task. The team members primarily responsible for each deliverable are shown below:

Deliverable	Responsible team member	Additional team members
Website	Sammy	Lucy
Requirements	Zac	Lucy
Architecture	Lia	Zac
Method Selection & Planning	Tim	Lucy
Risk Assessment	Kai	Zac
Implementation	Liam	Sammy, Lucy

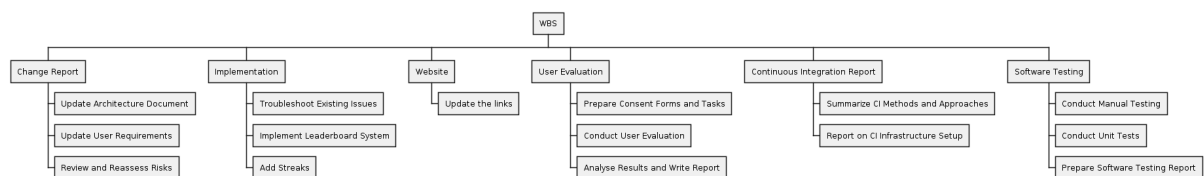
The plan remained very fluid throughout the execution of the project as team responsibilities, due dates and design ideas all shifted. Notably the team shifted towards having daily stand-ups as opposed to weekly meetings, this methodology closely mirrored the daily scrums described in Somerville [i]. Many tasks were pushed back as delays had cascading effects. Nonetheless the project was still completed on time thanks to the lengthy float associated with the implementation stage.

Assessment 2

For the second part of the project, our team organisation remained stable, with all members focused on achieving deployment milestones. We continued using the Scrum methodology, which had proven effective during the initial development phase due to its flexibility and adaptability.

Deliverable	Responsible team member	Additional team members
Website	Oliver Cassey	
Requirements	Mikolaj	Barnaby
Architecture	Oliver Cassey	Freya
Method Selection & Planning	Anna	
Risk Assessment	Anna	Cooper
Implementation	Oliver Thompson	Cooper, Oliver
User Evaluation	Mikolaj	Anna
Software Testing	Oliver Thompson	Mikolaj, Barnaby, Cooper
Continuous Integration Report	Anna	Cooper, Oliver Thompson

Work Breakdown Structure: [WBS.png](#)



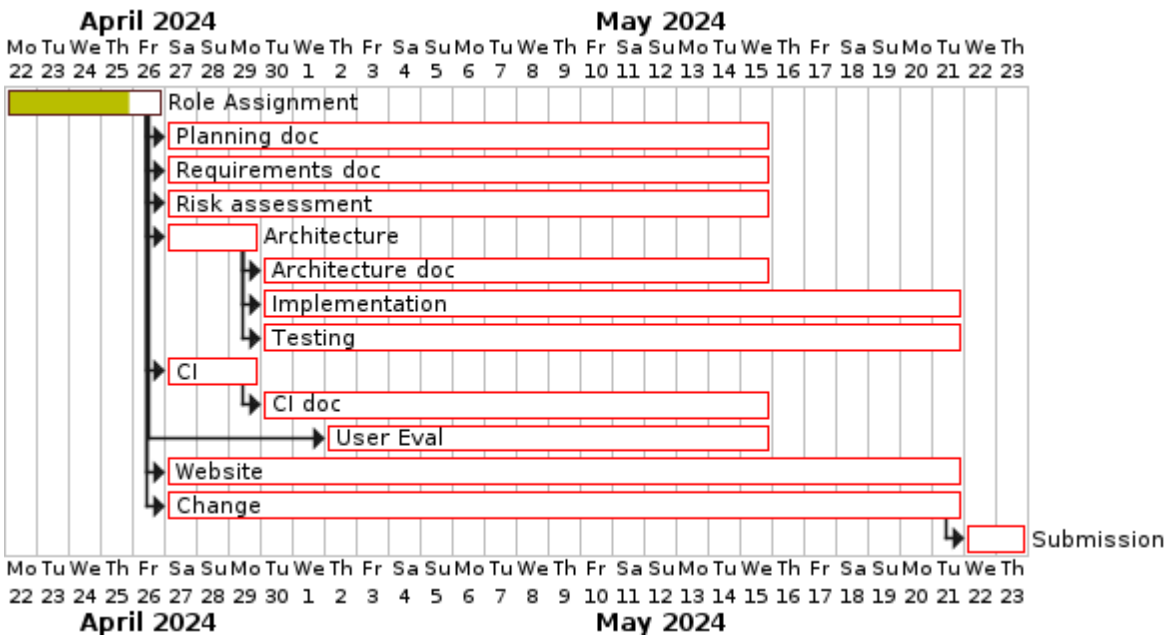
Throughout the second phase of our project , we maintained regular communication and adhered to core Scrum principles, including iterative development, weekly meetings, sprint planning, and retrospectives. This approach allowed us to swiftly address emerging issues and keep tasks aligned with project goals.

Approach and Methodology

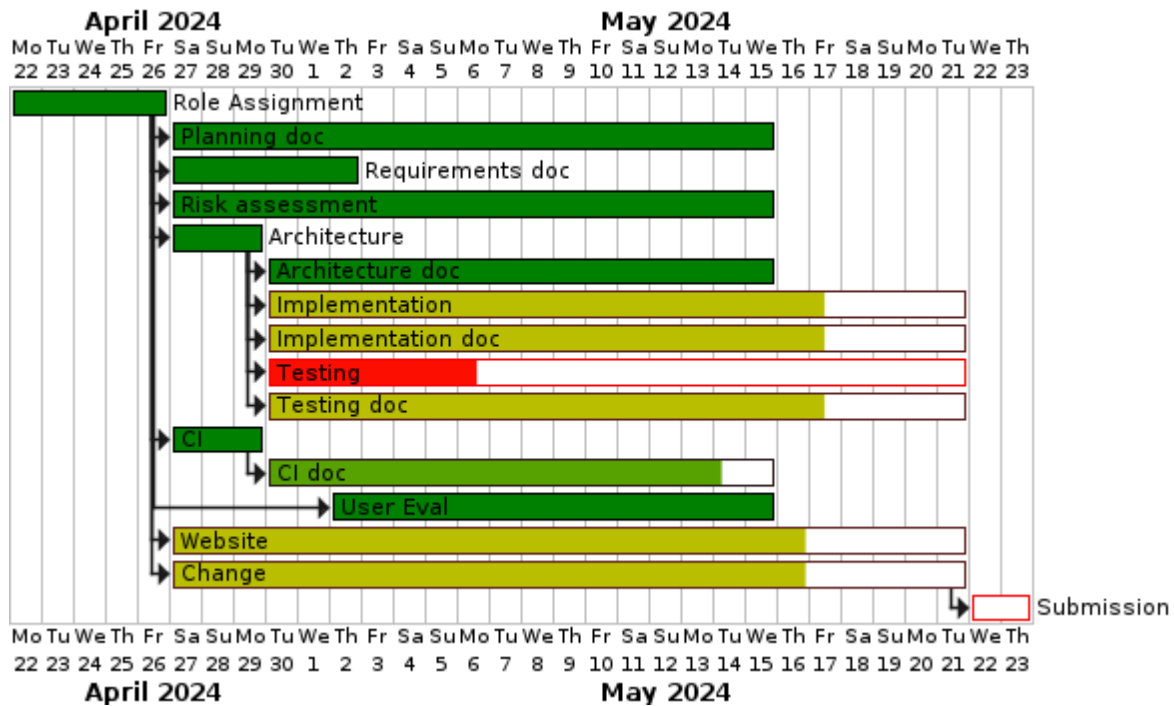
The flexibility of the Scrum methodology enabled us to adapt to minor adjustments and meet project objectives without significant disruptions. While we estimated initial

workloads, we did not impose strict deadlines for each task. Instead, each member was assigned tasks with the expectation of timely completion. As our team cultivated a collaborative environment, members were always open to offering assistance and providing feedback when necessary. In case of any issues with task completion, support was always available, making sure that tasks were completed on time.

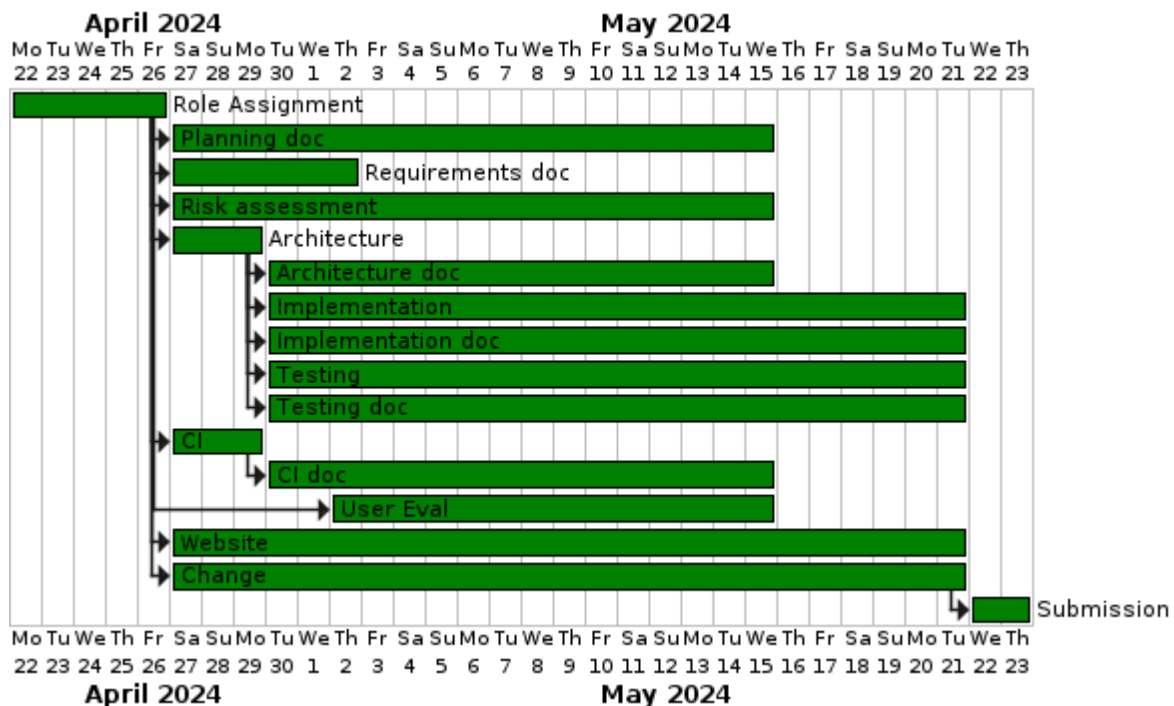
The use of a Gantt Chart was instrumental in tracking our progress, providing clear visibility into the development stages and remaining tasks. This tool, combined with clearly defined responsibilities for each team member, ensured smooth progression toward our goals and timely completion of all deliverables. Overall, our structured yet flexible approach facilitated effective project management and high-quality results.



Initial Gantt Chart



Interim Gantt Chart



Final Gantt Chart

Our plan was dynamically updated during our weekly meeting when we identified key priority tasks and assigned team members for them. We operated without a rigid plan ahead, allowing our strategy to evolve as our project progressed. This approach was highly effective during the first phase of development and remained unchanged in the second phase due to its continued success.

Project Plan:

Week 1 (April 25 - May 1)

- Update the website content.
- Conduct a comprehensive risk review and reassessment.
- Begin troubleshooting identified issues in the existing code.
- Update the User Requirements document to reflect the new defined requirements.
- Prepare for the upcoming User Evaluation.

Week 2 (May 2 - May 8)

- Conduct the User Evaluation to gather user feedback.
- Complete Change Report for the Risks Assessment and User Requirements part to reflect the latest project status and insights.
- Finish resolving coding issues, particularly those related to collisions
- Update the diagrams in Architecture to be up to date with changes in code

Week 3 (May 9 - May 16)

- Compile and document the Continuous Implementation Report.
- Develop and integrate the leaderboard system into the existing software.
- Create new structural and behavioural diagrams
- Perform detailed User Evaluation and document findings.
- Conduct manual testing of the software and prepare a report.

Week 4 (May 17 - May 23)

- Complete all pending project deliverables:
 - Finish CI Report
 - Finish Software testing Report
 - Update Architecture document
 - Finalise Change Report

- Finalise software implementation, ensuring all features are polished and functional.
- Prepare for the final presentation of the product.

References:

[1] D. West, "Agile Scrum Roles," *Atlassian*, 2019.

<https://www.atlassian.com/agile/scrum/roles>

[2] C. Harris, "Agile Scrum Artifacts," *Atlassian*.

<https://www.atlassian.com/agile/scrum/artifacts>

[3] Atlassian, "Three Pillars of Scrum: Understanding Scrum's Core Principles," *Atlassian*.

Bibliography:

i. Sommerville, "SOFTWARE ENGINEERING Ninth Edition," 2011. [Online]. Available:

<https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/Presentations/index.html>