Plan

Cohort 4 Group 6

Javengers

Braithwaite, Max
Faruque, Amber
Fu, Zhuoran
Kocaman, Melike
McDermott, John
Rissen, James
Scott, Charlotte

Unhighlighted = unchanged from part 1, blue highlight = added for part 2

Team 1's Software engineering methods (Assessment 1):

Our team adopted a Kanban Agile iterative development cycle by working in weekly cycles. Each week we have tasks created and tracked using GitHub projects, weekly meetings to discuss changes and review progress and make adjustments to priorities and internal deadlines. Following this approach allowed us to remain flexible and ensured efficiency when it came to the development process. We also integrated our Kanban board with Gantt charts to graph tasks over time, allowing for better time management of the project as a whole.
Agile development felt like the best for this project as it allows us to be flexible with our deadlines and team structure. We could change our plans as our risks become more prominent to allow for a prompt delivery of the software and documentation.

Our Adapted Software engineering methods (Assessment 2):

Team 1's software chosen engineering methodology overlaps nicely with our approach in the first half of the project, with an agile, Kanban iterative development approach, and so we will continue to adopt this structure in assessment 2. This will continue to provide us with clear visual tracking of tasks and workflow, and sets us up for continuous improvement and progress. As team 1 did, each week we will create tasks using github projects and place them under specific heading e.g. backlog, in progress, ready, reviewing. We will also integrate our Kanban board with Gantt charts to graph tasks over time, allowing for better time management of the project as a whole and a more accurate plan.

One additional methodology we would like to implement in which team 1 didn't is SCRUM. Our team utilised this approach in assessment 1 and we found it extremely effective in our development. SCRUM organises work into short-term, smaller scale iterations called sprints, which involve reviewing, planning, and development. Key roles include a Scrum Master, who facilitates meetings, ensures progress, and helps resolve blockers, and a Product Owner, who prioritises tasks and communicates requirements. By using this approach, we will continuously improve and deliver increments of the project regularly. As we did in assessment 1, we will perform our SCRUM sprints weekly, meeting at the start of the week to discuss that week's work. SCRUM tends to take place daily, however as we did in assessment 1, we will continue to meet weekly since it allows us to balance our academic workload while still maintaining regular progress and consistent communication. We will also resume daily contact through our WhatsApp group to discuss progress.

The use of SCRUM provides us with significant flexibility in prototyping, testing and evaluating new game features as we continue to expand upon team 1's game. Changes can be made swiftly, for example we may need to reassign roles / tasks due to changing task priorities and requirements. Each time we meet, we will continue to discuss these task priorities and everyone's progress on their assigned task from the previous week. We will also discuss risks and anticipate potential issues that may arise to prevent any setbacks in the project. The

flexibility allows us to reallocate roles within the team if any issues arise. Finally, we assign team members responsibilities for the next sprint for the upcoming week.

Scrum provides us with the ability to adapt quickly and make changes to our initial plans, unlike other non-Agile methodologies like plan-driven development, where plans aren't interchangeable and adaptable. This makes them less suitable for evolving projects like game development. The final product of our game development will likely not fully align with the initial plan we created at the start of Assessment 2. Changes will need to be made to the initial plan throughout the game's continued development of new, more efficient ideas and solutions are created.

Development and collaboration tools:

Tool	Purpose	Justification	Alternatives considered
Git	Version Control	Reliable and supports easy collaboration. It tracks changes, allows us to create and merge branches and gives all team members a copy of the project. Keeps detailed history to help with fixing mistakes and manage code efficiently. Several team members also already had experience with Git reducing time spent learning new tools.	BitBucket
PlantUML	Diagram Creation	Fast and easy tool to create detailed diagrams for task breakdown. Not too difficult to learn and integrate diagrams in Google Docs/GitHub. The text based nature of it makes version control and standardisation much easier than a tool like diagrams.net.	diagrams.net
libGDX	Game Development Framework	Compatible with 2D Java game development and can be deployed to various operating systems. Has built-in tools and documentation to help with quick game implementation. jMonkeyEngine's lack of documentation was a key factor that made us choose libGDX instead.	jMonkeyEngine No game engine
GSuite	Documentation	Multi-user editing and automatic saving helps with quick documentation and collaboration. Ease of use and familiarity took priority while considering other alternatives like Office and Git-Controlled Markdown.	Microsoft Office Git-Controlled Markdown

IntelliJ IDEA	IDE for Java Development	Provides support for Java development through its debugging tools making testing and writing code easier. Integrated frameworks like LibGDX, while using VS code would require extensions.	VS Code
Netlify	Website Hosting Provider	Quick and easy, no requirement for additional setup or complicated configuration. GitHub Pages or similar providers would work	GitHub Pages
Microsoft whiteboard	Brainstorming ideas	Collaborative space with an infinite canvas and real-time editing for visual brainstorming between group members. Easy way to display and break down individual ideas. Considered canva's whiteboard but we had previous experience with microsoft whiteboard.	Canva
VS Code	IDE for Web Development	VS Code is a familiar and user-friendly IDE that supports key web technologies like HTML, CSS and JavaScript, and provides built-in debugging tools to streamline the web development process. Unlike Notepad++, VS Code integrates Git and extensions, which is beneficial to us during development.	Notepad++
Discord	Communication	Discord is an efficient communication method as it's easy to use and provides organisation through the use of separate channels. WhatsApp and Slack, but Discord combined features of both.	WhatsApp Slack
Piskel	Character Design and Pixel Graphic Design	Allows us to create characters by using pixel art so we can make them suitable for our game. The website is intuitive to use and free to use so ideal for this project.	Asprite Pixel Art Make DinoPixel
Paint.net	Room Design and Bitmap Graphic Design	Free and incredibly simple bitmap image editor for creating a wide variety of assets. Particularly useful for room design.	Krita Photoshop MyPaint
Pixabay	Background music	Provides copyright free music that we can use in our game without a license.	Bensound

- **Rows highlighted in red indicate tools that we will not use them in Assessment 2 but were used by Team 1 in Assessment 1.**

Google Docs	Documentation	Multi-user editing and automatic saving helps with quick documentation and collaboration. Ease of use and familiarity took priority while considering other alternatives like Office and Git-Controlled Markdown.	Microsoft Office Git-Controlled Markdown
Whatsapp	Communication	Whatsapp is an efficient and easy communication method, and well suited since it is most commonly used..	Discord Slack
Photoshop	Room Design and Bitmap Graphic Design	Photoshop is the better choice because it has more tools, more editing options, and stronger features than Paint.NET and MyPaint. Furthermore, our team is more familiar with photoshop.	Paint.net MyPaint
Github Pages	Website Hosting Provider	GitHub Pages is ideal as it directly updates the GitHub repo, and is simple for static HTML/CSS/JS sites. No extra setup is needed.	Netlify

- **Rows highlighted in green indicate the updated tools that we have replaced with Team 1's versions for Assessment 2.**

Team 1's - Outline of Team Organisation

We organised our team by assigning primary and secondary leaders for each deliverable. This meant that each of us were part of multiple sub-teams. We ensured that each team member was assigned tasks based on their strengths to allow each sub-team to be as effective as possible. This approach ensured that each deliverable benefited from both individual and collaborative engagement to tackle each one to the highest standard.

The primary(s) is in charge of the main part of that deliverable and in delegating tasks to the secondaries to ensure internal deadlines can be met by multiple people working in parallel. The secondary members on each team allowed us to internally review each other's work, provide assistance and feedback easily when needed to reduce the chance of errors and ensure high quality work. This set up allows each member to clearly understand their role and responsibilities leading to seamless documentation and development.

As a team, to stay organised, we wanted to use a **Kanban**-style tool for task tracking, as well as **Gantt** charts for managing time constraints. This allows our sub-teams to communicate with each other and for us to know where the other teams are currently up to. We considered two main options for managing this:

1. Trello for Kanban Boards, PlantUML Gantt Charts
2. GitHub Projects, with GitHub Issues as our equivalent of "Tasks"

Trello Kanban Boards were a tool which we thought was well-suited for the project due to the versatility it would give us by assigning tasks, setting due dates, and creating automations.

PlantUML Gantt charts were also a consideration because of the standardisation of UML. They would facilitate version controlling our Gantt charts, whilst keeping track of tasks within the same *Google Docs* ecosystem used for writing up most deliverables.

We came to the conclusion that GitHub Projects, whilst more complicated to learn at first, integrates both Kanban and UML into one place. This would save time over option 1 since tasks are shared between charts automatically as Issues.

The benefits of using one tool instead of two made the decision to use GitHub Projects a quick one, particularly since there we would not be missing out on any of Trello's or PlantUMLs benefits by doing this. GitHub already tracks changes to issues on each individual task.
This decision also meant that we avoided the need to regularly produce new charts, and instead had a plan that adapted as we made changes.

Our updated / changed Outline of Team Organisation:

Similarly to team 1, we decided to assign primary leaders to each deliverable task, which they were primarily responsible for. However, Secondary members consisted of the rest of the members within the team, who could be assigned the role of assisting the primary leaders.

We discussed each other's main strengths and weaknesses and allowed each member to request their preferred role to take on for assessment 2. For example, we identified which team members would prefer to focus testing the game, or conducting the User Evaluation. However, as in assessment 1, we wanted to ensure that no team member was strictly limited to their role. Instead they were given primary responsibility for that role, and all team members (Secondary members) are able and encouraged to contribute to other tasks within the group provided their primary focus is on their initial role. This approach to organisation links nicely with SCRUM as it provides us with flexibility, allowing us to distribute team members to tasks of higher priority if needed. Finally, this approach ensured that each deliverable benefited from both individual and collaborative engagement to tackle each one to the highest standard.

The primary leader(s) is in charge of the main part of that deliverable and in delegating tasks to the other team members who may be assigned to the role if the need arises. Internal deadlines can be met by multiple people working in parallel on tasks. The secondary members on each team allowed us to internally review each other's work and provide assistance and feedback easily when needed in order to reduce the chance of errors and ensure high quality work. This set up allows each member to clearly understand their role and responsibilities leading to seamless documentation and development.

As a team, to stay organised, we wanted to use a **Kanban**-style tool for task tracking, as well as **Gantt** charts for structured planning and managing time constraints. This allows our group to communicate with each other and for us to know where and what each team member is currently up to.

We considered the following options for managing Kanban and Gantt charts:

Kanban Boards:

- Trello
- Github Projects

Gantt Charts:

- PlantUML
- GitHub Projects

We came to the conclusion to utilise GitHub Projects for our kanban board, in combination with PlantUML for Gantt charts.

Using **GitHub Projects'** Kanban boards is especially suitable because they live in the same environment where your code and collaboration takes place. This makes tasks quicker and easier to track.

Furthermore, we utilised **PlantUML** within assessment 1, and it was easy and simple to use, not being too overcomplicated, saving time whilst delivering the same product. It's also efficient keeping track of tasks within the same *Google Docs* ecosystem used for writing up most deliverables.

Team 1's Systematic Plan - Assessment 1:

Game Overview

QUACK? will be a top down, room-based game. The player is a student at the University of York who is being chased by the dean, who is accusing them of stealing the Long Boi statue. The player must find and retrieve the missing *Long Boi* statue somewhere on East Campus in order to get past the dean, who won't let you leave campus until the statue is returned.

To find the statue the player will have to search the University campus, triggering various events which may help or hinder them.

The player will move around the screen whilst the camera remains stationary, but they will be able to move orthogonally between rooms into other adjacent rooms (if they exist). East Campus will be approximated with an intertwining grid of rooms in a maze-like structure, whilst still vaguely approximating the University's campus.

The player will have energy which can be used to run faster, which benefits the player in achieving their goal of finding the statue before the timer runs out. Players may have their energy reduced by the University's geese attacking players who get too close to them. Energy will naturally regenerate, but if it ever hits 0 they will be slowed down until it has fully recovered.

The events which the player may trigger are described in internal deliverable d10.

Key Tasks

Each subdeliverable is a "key task" which has an individual responsible for it, as well as sub-tasks which break down the problem further. Some details of each of these subdeliverables

are included below, although full details are on the GitHub Project, available here. This includes dependencies and priorities for tasks.

We also have catalogued snapshots of the state of the GitHub Project (both Kanban and Gantt charts) on the website. These snapshots are available here. The plan evolved week by week in each of our meetings, these changes were reflected in the issues on GitHub and through informal discussion on Discord. For example, we initially planned on implementing the Map screen as our positive event for Assessment 1, but later decided that we would complete the Vending Machine event instead, as this would be easier to implement with the tight time constraints.

THIS TABLE AND ITS LINKS WERE UNCHANGED FROM PART ONE

Code	Name	Start Date	End Date	Assignee(s)
D1a	Links to Deliverables	Nov 1st	Nov 7th	Leo
D1b	Links to GitHub Repositories	Oct 17th	Oct 22nd	Sam
D1c	Link to Executable JAR	Nov 2nd	Nov 7th	Hamza
D1d	Interim UML Diagrams	Oct 31th	Oct 5th	Sam
D1e	Weekly Snapshots of Plan	Oct 8th	Nov 7th	Sam
D2a	Introduction	Oct 8th	Oct 22nd	Wasif Emma
D2b	Statement of Requirements	Oct 10th	Oct 22nd	Emma Leo
D3a	Statement on Tools & Languages	Nov 4th	Nov 7th	Wasif Leo
D3b	Structural UML Diagrams	Oct 15th	Nov 4th	Hamza
D3c	Behavioural UML Diagrams	Oct 15th	Nov 4th	Leo
D4a	Outline & Justification of Tools	Oct 3rd	Oct 22nd	Anexa Sam
D4b	Outline of Team Organisation	Oct 1st	Oct 22nd	Sam Anexa
D4c	Plan	Oct 6th	Nov 4th	Sam
D5a	Risk Management Process	Oct 5th	Oct 20th	Leo
D5b	Risk Register	Oct 3rd	Oct 20th	Leo

				Emma
D6a	Documented Code	Oct 17th	Nov 7th	All
D6b	3rd Party Libraries & Assets	Nov 3rd	Nov 6th	Anexa Emma

Our Updated, Systematic Plan For Assessment 2:

Game Changes Overview:

The changes we made to the game consist of the following:

Negative Events:
- There is a frog located on the map which can eat you if you get too close
- You can get caught by campus security, and they will take the found pieces of the goose.
- A power cut which is randomly triggered
- A coffee slip
- You can lose speed and a timeout will occur if you drink too many energy drinks.

Positive Events:
- Coins which can be collected around the map
- Energy drink via the vending machine which gives you a speed boost

Hidden Events:
- There is a rambling professor who will distract you from your task
- Strong gusts of wind which slow the player down
- The player can find a Top hat for the goose.

We broke the Assessment 2 into a series of **Work Packages** (WPs):

- **WP1**: InitialTeam Organisation
- **WP2**: Risk Assessment Change
- **WP3**: Requirements Change
- **WP4**: Architecture Change
- **WP5**: Methods / Planning Change
- **WP6**: Change Report
- **WP7**: Game Testing

- **WP8**: Game implementation
- **WP9**: Continuous Integration
- **WP10:** User Evaluation
- **WP11**: Website.

Each Work Package will contain a sequence of sub-tasks in the form: Task (per WP). These sub-tasks, along with their dependencies, priority, start and end dates, can be found on the project task table on the website.

Milestones: These mark important achievements and checkpoints in the project timeline, representing key points where significant progress or completion happens.

- **M1**: Establish and organise the team - Due Date: Monday 24th November
- **M2**: Finish implementing tests on the initial game - Due Date:
- **M3**: Finish implementing new Game features - Due Date:
- **M4**: Finish implement tests for new game features - Due Date:
- **M5**: Finalise the project's work packages before the peer review - Due Date:
- **M6:** Project completion and submission - Due Date:

Additionally, we have broken the project down into a series of deliverables, which are concrete outputs of the project. These can be either documents, like Risk Assessment / Architecture, or software, like prototypes / tests. Each deliverable is given an ID and can have multiple versions, such as interim drafts and final versions, each with unique IDs, e.g. (D1.1, D1.2).

Coe	Name	Start Date	End Date	Description	Visibility	Assignee(s)
D1a	Initial Team Organisation / Project Setup	21st Nov	24th Nov	Organise our team, assigning roles, ensure we are set up to continue team 1's project.	Internal	Everyone
D2a	Risk Management Process (Updated)	25th Nov	24th Dec	Any changes being made to the Risk Management Process	Internal / External	Amber
D2b	Risk Register (Updated)	25th Nov	24th Dec	Add any new additional risks to the risk register.	Internal / External	Amber

D3a	Requirements Introduction (Updated)	25th Nov	24th Dec	Any changes being made to requirements elicitation methods	Internal / External	John
D3b	Statement of Requirements (Updated)	25th Nov	24th Dec	Any new requirements added to the table	Internal / External	John
D4a	Statement on Tools and Languages (Updated)	25th Nov	24th Dec	Any new tools / languages used	Internal / External	Max
D4b	Structural UML Diagrams (Updated)	25th Nov	24th Dec	Any changes to the Structural UML diagrams	Internal / External	Max
D4c	Behavioural UML DIagrams (Updated)	25th Nov	24th Dec	Any changes to the Behavioural UML diagrams	Internal / External	Max
D5a	Outline & Justification of team tools (Updated)	25th Nov	24th Dec	Any changes to software methodologies or development and collaboration tools	Internal / External	James
D5b	Outline of team organisation (Updated)	25th Nov	24th Dec	Any changes made to team organisation approach	Internal / External	James
D5c	Plan (Updated)	25th Nov	24th Dec	The systematic plan for assessment 2.	Internal / External	James
D5d	Weekly Snapshots of plan for Assessment 2	25th Nov	24th Dec	Gantt charts showing our teams progress.	Internal / External	James
D6a	Final merged change report	6th Jan	12th Jan	Bringing all changed deliverables together	External	James, Amber, John, Max
D7a	Initial Testing for initial Game	25th Nov	29th Nov	Testing the game prior to implementation	Internal / External	Charlotte, Max, Zhouran
D7b	Final testing for final game with new implemented features	6th Jan	12th Jan	Testing the final, finalised game	Internal / External	Charlotte, Max, Zhouran
D8a	Game prototype - Interim Version	5th Dec	24th Dec	Game prototype after newly implemented	Internal / External	Charlotte, Max,

				events for assessment 2		Zhouran, Amber
D8b	Game prototype - Final Version	6th Jan	12th Jan	Game prototype after final testing and refactoring if necessary.	External	Charlotte, Max, Zhouran, Amber
D8c	Documented code	6th Jan	12th Jan	Final, complete, documented code.	External	Charlotte, Max, Zhouran, Amber
D9a	Summarise Continuous integration methods	5th Dec	24th Dec	Discuss CI methods / approaches our team will use	Internal / External	James
D9b	Continuous Integration infrastructure Report	5th Dec	24th Dec	Discuss the CI infrastructure our team has set up.	Internal / External	James
D10a	User Evaluation Method report	13th Dec	20th Dec	Discuss our approach to our user evaluation	Internal / External	John
D10b	User Evaluation Table	13th Dec	20th Dec	Collect / document results from user evaluation	Internal / External	John
D11a	Finalised Website	25th Nov	12th Jan	The finalised website containing all required information	External	Melike

The following write-up outlines the changes made to the project plan throughout the project timeline and references the weekly Gantt charts, which are available on our website.

Timeline: - Friday 21st November - Friday 12th December (overtime until January 12th

Initial Gantt Chart:This Gantt chart illustrates our initial project timeline, which we aimed to follow as closely as possible; however, we expected some adjustments, as minor deviations from the plan were unavoidable. At the start of the project, effective team organisation is essential to clearly define roles and responsibilities within the team. Once this is completed, the team can begin working on the change deliverables and building the website. Before beginning any architectural work, we planned to carry out a full and thorough testing phase of the game before making any changes or additions. This ensures the codebase is bug-free, clean, modular, and maintainable. During this stage, bugs may need to be fixed or code refactored to improve quality, which can directly affect the architectural structure, for example through the

addition or removal of classes. Once this process is completed, we can begin brainstorming ideas for additions to the game before starting their implementation. Alongside implementation, any new features introduced will be tested in parallel. Furthermore, we will begin implementing continuous integration to make development more efficient, for example by automating testing and build processes. We have planned to conduct our User Evaluation interviews on Friday the 12th of December, which is our final timetabled practical session before the Christmas holidays. This allows us maximum time to prepare and ensures that as many new features as possible are available for feedback. Afterwards, documentation of the User Evaluation can begin. Before our internal deadline on the 24th of December, we aim to have completed draft versions of all deliverable documents and implemented and tested the required new game features. This is because after the Christmas holiday period, we plan to finish with one week of peer reviewing and one week of refining and finalising our work before submission, ensuring quality assurance.

Week 1 Gantt Chart:As planned, during week 1 (Monday 24th November to Monday 1st December), including the weekend of the 21st–23rd, we began with team organisation and evenly distributed roles across the team. We also made a start on the change documentation, for example by adding any new requirements or identified risks. In addition, we began structuring and sectioning the website. Following this, we conducted testing on the current state of the game and carried out a review of the existing code. During this process, we identified several flaws and bugs within the game, as well as areas of code that required refactoring. One example was the “lights out” event, where the lights would remain permanently off if the player left the room, rather than turning back on as intended. Towards the end of the week, we began fixing these bugs and refactoring the affected code. Max, who was assigned responsibility for architectural changes, began updating the architecture section, including changes to the class diagrams. No changes were made to the Gantt chart for week 1, as progress remained in line with the initial plan.

Week 2 Gantt Chart:During week 2, we continued with bug fixing and code refactoring. This process took longer than expected, as the game required more work than initially anticipated, resulting in this task being extended by two days, until the 5th of December. Consequently, we had to delay the start of subsequent tasks that were dependent on the completion of this phase. Throughout this week, work on all change deliverables continued. We also began brainstorming new game features on the 5th of December and subsequently started implementing these features, with testing carried out in parallel. Finally, we began setting up the continuous integration (CI) infrastructure, including GitHub Actions and automated testing and build processes using Gradle.

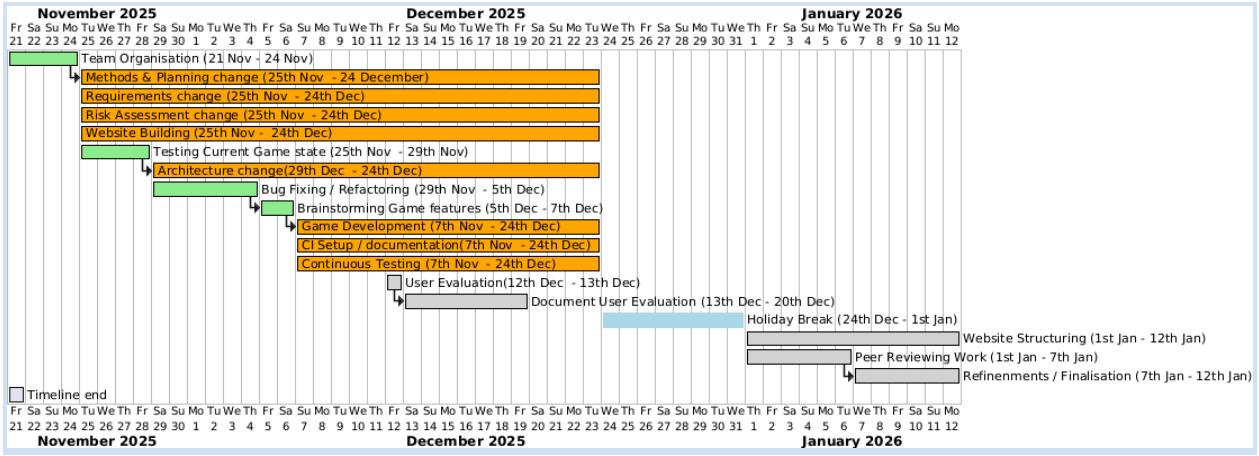
Week 3 Gantt Chart: In week 3, we conducted the user evaluation on the 12th of December during our timetabled practical session and began documenting the feedback obtained from the interviews. Alongside this, game development continued, for example through the implementation of positive and negative events. The team also began finalising draft versions of

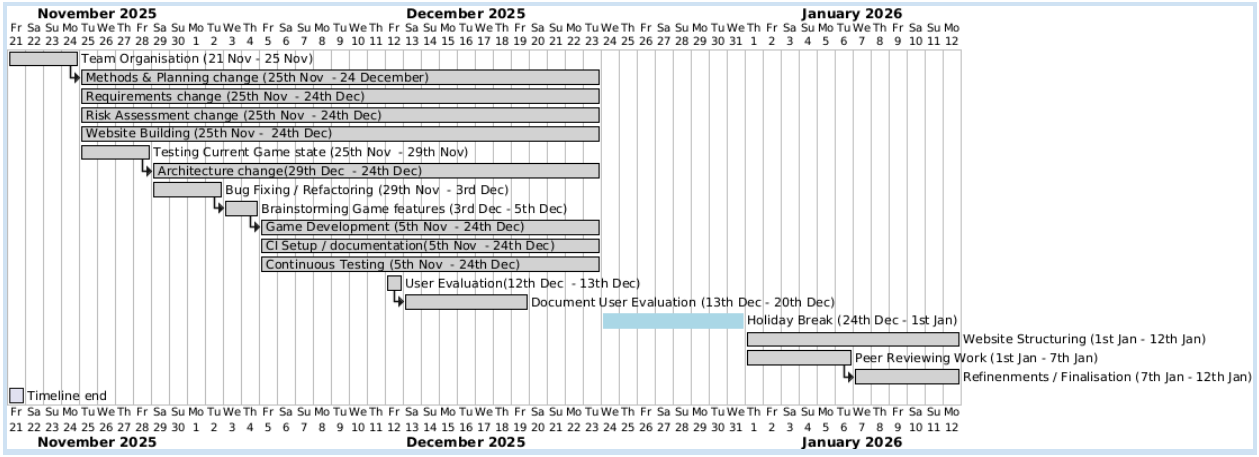
the change deliverables in preparation for the internal deadline on the 24th of December, ready for peer review after the Christmas holiday. No changes were made to the initial plan during week 3, as progress remained on track.

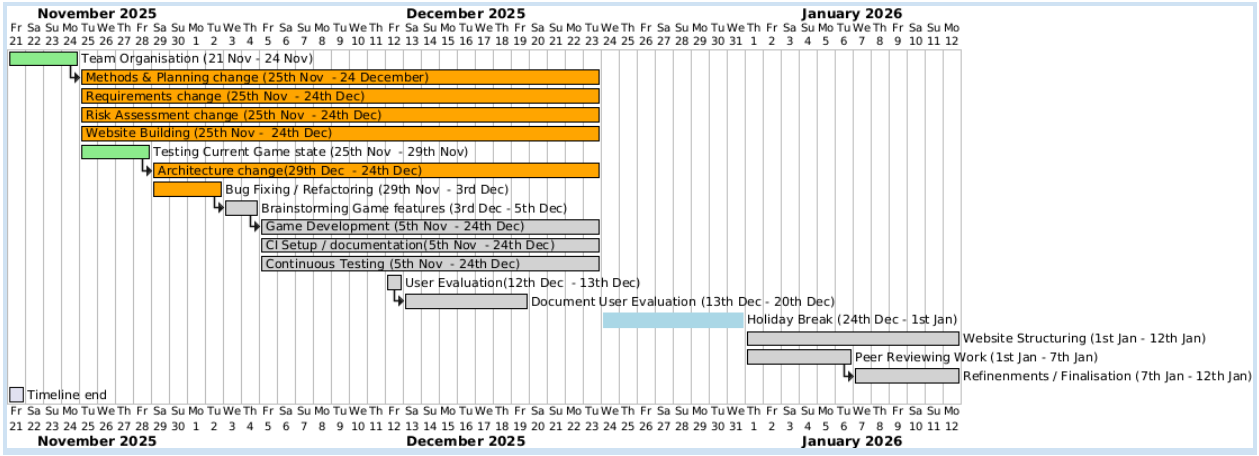
Week 4 Gantt chart:The week 4 Gantt chart shows that all tasks were completed in time for our internal deadline. During this week, we made two changes to the Gantt chart. Firstly, we were able to complete the user evaluation documentation two days earlier than planned. This allowed us to reassign team members from these tasks to assist with the remaining change documents, ensuring they were completed before the internal deadline. The second change involved extending website structuring tasks to continue after the holiday period and up until the final deadline. This decision was made because we anticipated that changes would be required during the two weeks allocated for peer review and refinement, meaning the website would need to be continuously updated until completion, for example to incorporate diagrams, documents, and other materials.

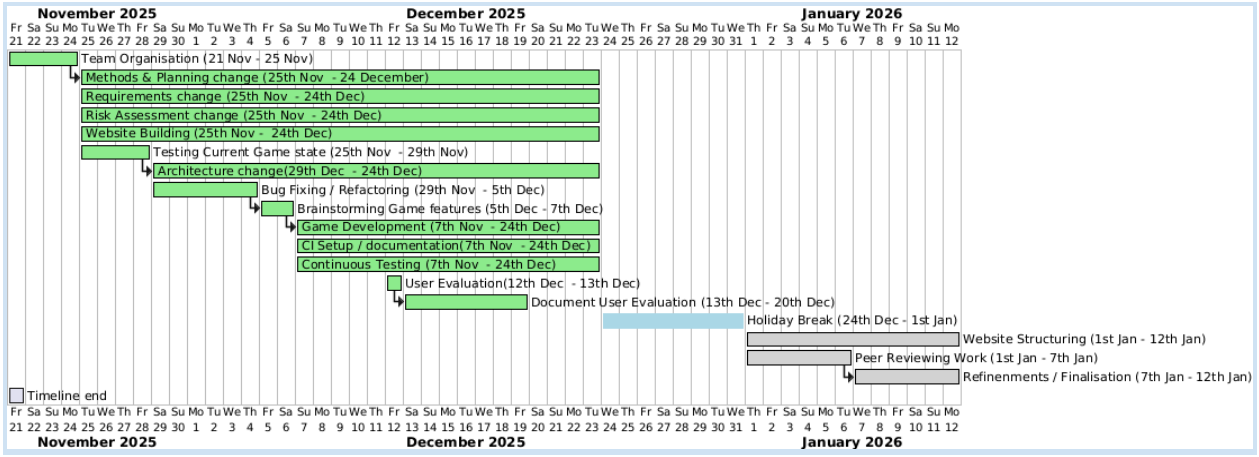
Week 5 Gantt Chart (Starting 1st January after christmas holidays: This week, we conducted our peer reviews. All team members working on the change deliverables reviewed a different change document and provided detailed feedback. Team members responsible for development also reviewed their code and carried out all necessary tests to ensure correct functionality. This process provided us with the feedback required to support refinement and finalisation during the final week.

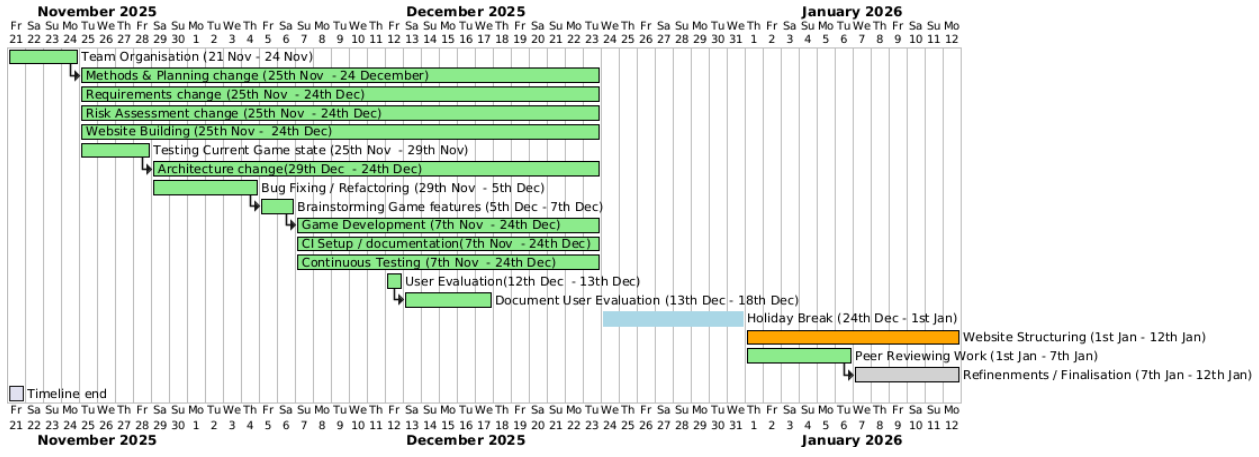
Final Chart: In the final week, we made final revisions to all deliverables based on the feedback received during the peer reviews, prior to submission. Additional team members were assigned to the website to ensure it was fully structured and complete. The final Gantt chart reflects the overall completion of the project.

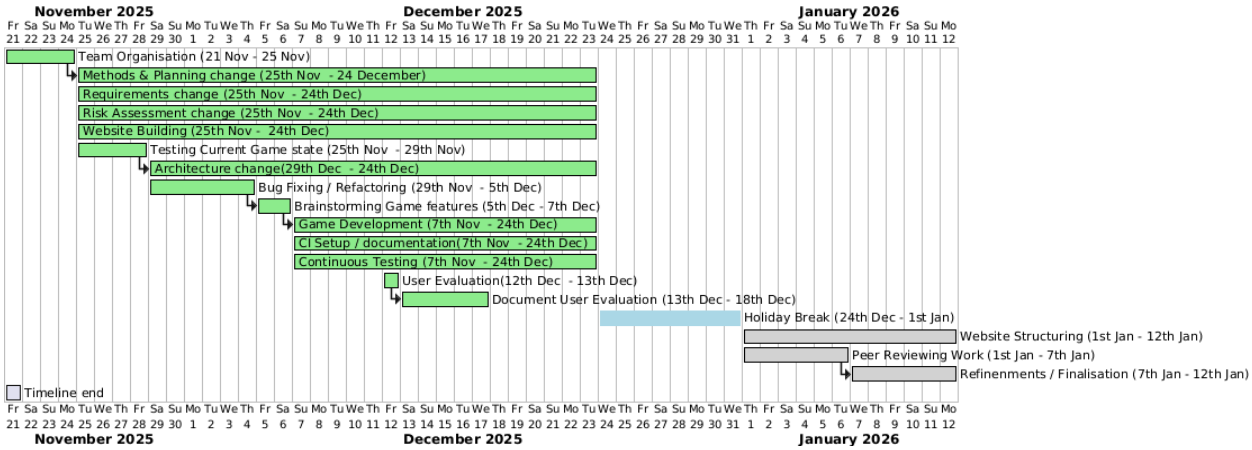

November 2025
December 2025
January 2026
Team Organisation (21 Nov - 24 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 5th Dec)
Brainstorming Game features (5th Dec - 7th Dec)
Game Development (7th Nov - 24th Dec)
CI Setup / documentation(7th Nov - 24th Dec)
Continuous Testing (7th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 20th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end

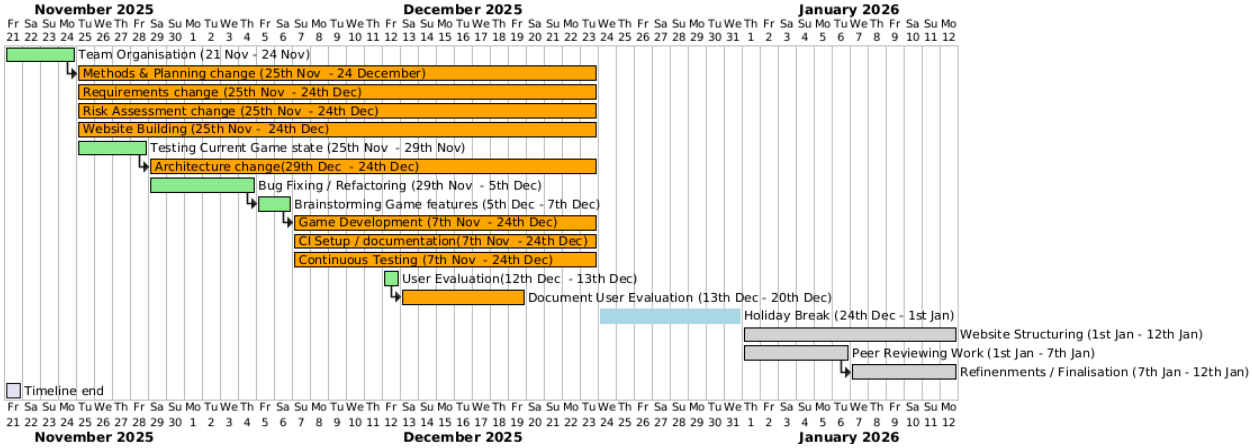

November 2025
December 2025
January 2026
Team Organisation (21 Nov - 25 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 3rd Dec)
Brainstorming Game features (3rd Dec - 5th Dec)
Game Development (5th Nov - 24th Dec)
CI Setup / documentation(5th Nov - 24th Dec)
Continuous Testing (5th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 20th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end

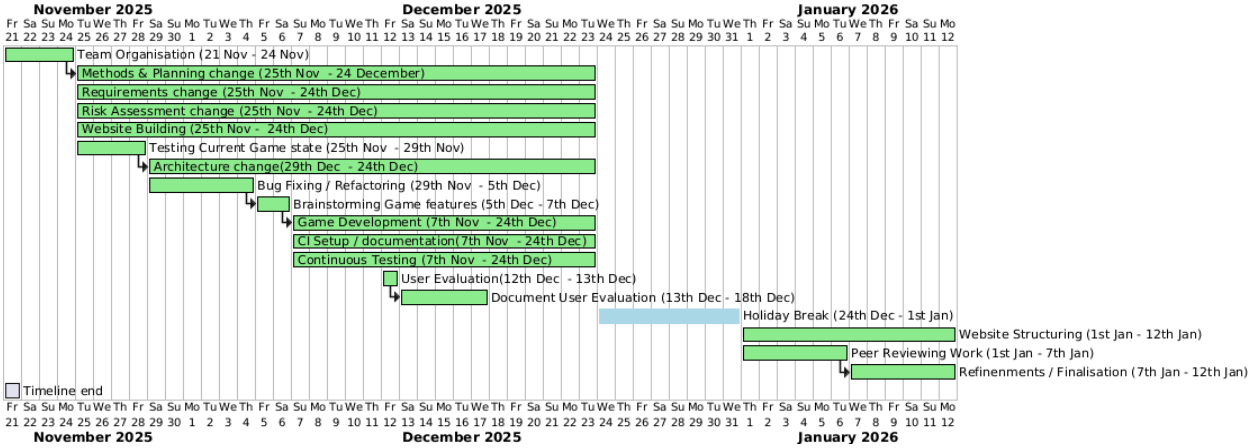

November 2025
December 2025
January 2026
Team Organisation (21 Nov - 24 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 3rd Dec)
Brainstorming Game features (3rd Dec - 5th Dec)
Game Development (5th Nov - 24th Dec)
CI Setup / documentation(5th Nov - 24th Dec)
Continuous Testing (5th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 20th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end


November 2025
December 2025
January 2026
Team Organisation (21 Nov - 25 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 5th Dec)
Brainstorming Game features (5th Dec - 7th Dec)
Game Development (7th Nov - 24th Dec)
CI Setup / documentation(7th Nov - 24th Dec)
Continuous Testing (7th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 20th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end


November 2025
December 2025
January 2026
Team Organisation (21 Nov - 24 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 5th Dec)
Brainstorming Game features (5th Dec - 7th Dec)
Game Development (7th Nov - 24th Dec)
CI Setup / documentation(7th Nov - 24th Dec)
Continuous Testing (7th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 18th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end


November 2025
December 2025
January 2026
Team Organisation (21 Nov - 25 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 5th Dec)
Brainstorming Game features (5th Dec - 7th Dec)
Game Development (7th Nov - 24th Dec)
CI Setup / documentation(7th Nov - 24th Dec)
Continuous Testing (7th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 18th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end


November 2025
December 2025
January 2026
Team Organisation (21 Nov - 24 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 5th Dec)
Brainstorming Game features (5th Dec - 7th Dec)
Game Development (7th Nov - 24th Dec)
CI Setup / documentation(7th Nov - 24th Dec)
Continuous Testing (7th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 20th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end


November 2025
December 2025
January 2026
Team Organisation (21 Nov - 24 Nov)
Methods & Planning change (25th Nov - 24 December)
Requirements change (25th Nov - 24th Dec)
Risk Assessment change (25th Nov - 24th Dec)
Website Building (25th Nov - 24th Dec)
Testing Current Game state (25th Nov - 29th Nov)
Architecture change(29th Dec - 24th Dec)
Bug Fixing / Refactoring (29th Nov - 5th Dec)
Brainstorming Game features (5th Dec - 7th Dec)
Game Development (7th Nov - 24th Dec)
CI Setup / documentation(7th Nov - 24th Dec)
Continuous Testing (7th Nov - 24th Dec)
User Evaluation(12th Dec - 13th Dec)
Document User Evaluation (13th Dec - 18th Dec)
Holiday Break (24th Dec - 1st Jan)
Website Structuring (1st Jan - 12th Jan)
Peer Reviewing Work (1st Jan - 7th Jan)
Refinenments / Finalisation (7th Jan - 12th Jan)
Timeline end