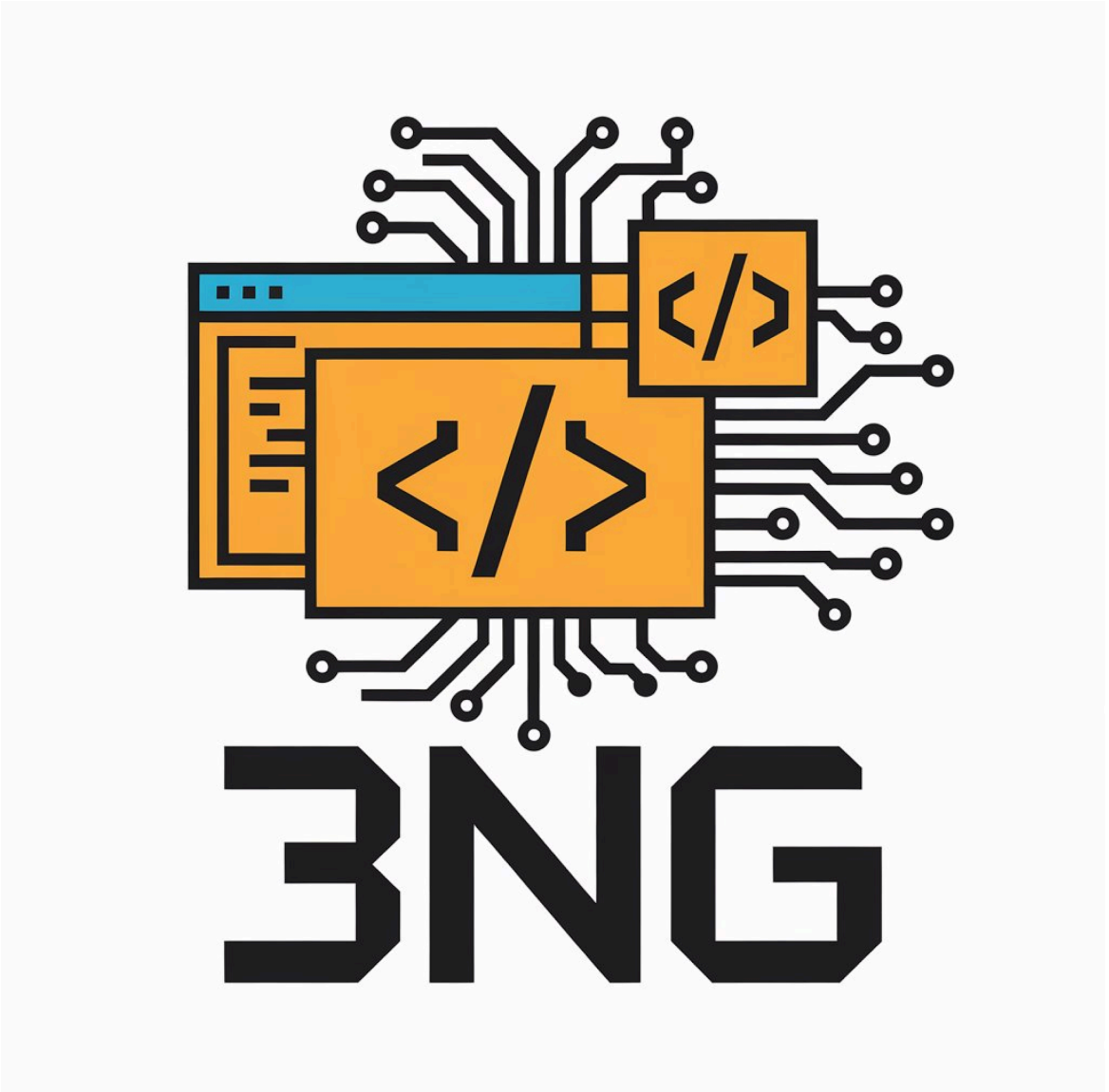
Method Selection and Planning

Cohort 4 Group 3

Team Members
Charles Fellows
Elif Gunes
Eve Davey
Hongrui Liu
Leo Owen-Burton
Tomi Olagunju
Will Bannerman



Development Methodology

We have chosen to use the Agile approach to software engineering, namely by using a Scrum-inspired model. This would be the most effective as it is also the most responsive to changing or developing ideas, which would allow us to easily add or remove certain features we may or may not need. The process of planning, developing, and then testing would allow for a more modular approach where we can easily develop and test individual objects and methods, which can then come together to form executable prototypes of the game. This would prevent the possibility of spending days developing code only for it to not work or become entirely redundant when we have any change in ideas, such as could happen using a waterfall approach or other similar ones. As well as active collaboration between design and development members, since the game is of an interactive and visual nature, frequent prototyping and feedback were inevitable to ensure that gameplay, user interface, and difficulty levels evolved seamlessly.

Chosen Method and justification: Agile-Scrum was selected over other traditional methods such as Waterfall because:

- It supports incremental development. Every sprint generates a playable version of the maze game, which could then be tested and improved upon.
- Agile's flexibility helped accommodate changes.

We also considered alternative methods such as Kanban and Spiral, but found that neither provided the sprint-based structure and frequent feedback loops that suited our small team and limited timeframe. Kanban's continuous flow approach lacked defined sprint boundaries, while the Spiral model required more formal planning phases than our project schedule allowed.

For collaboration, we initially considered Slack and Instagram, but later replaced them with Discord, Google Drive, and WhatsApp. Discord allowed for real-time voice and text communication during development sessions, making it more interactive and suited to our Agile-style sprint reviews. Google Drive provided a secure and organised space for sharing reports, diagrams, and documents, while WhatsApp worked best for quick daily updates and reminders.

As a team, the main tools we are using are the following development and collaboration tools:

- GitHub
- Google Drive
- WhatsApp
- Visual Studio Code / IntelliJ
- Discord

GitHub is the ideal tool to use for the codebase as it allows us all to work independently on local copies of the program with branches, which is suited for any methodology but especially Agile, as we have chosen. Following on from this, it helps the coder and code reviewer to look over the feature to be implemented before merging it into the main branch,

to mitigate the risk of any updated code breaking the system or simply not working to an ideal standard.

Approach to Team Organisation

We have split ourselves into several (not entirely strict, but recommended) roles for the course of the project. These are as follows:

- Meeting Lead (Taking charge of meetings, creating plans for the meetings and taking notes for any of these)
    - Charlie
- Librarian (Keeping documents clean and organized, and checking version control)
    - Tomi, Elif
- Report Editor (Taking charge and polishing up the documentation and reports, as well as adding in comments)
    - Will
- Code Reviewer (Looking over the code in the repository - reviewing. If possible, optimises and improves code or simply cleans it up)
    - Leo, Liu
- Creative Lead (Suggesting new ideas for the game UI and other more creative parts of the game - designs and other things of that sort)
    - Tomi, Elif, Eve

These roles were made and decided by the team to allow us to pick one that best fits our individual strengths, allowing us to focus on these areas of the project more while being flexible enough that we can all contribute to any part.

Splitting the project into these roles allows us to work more efficiently and in parallel on whatever we can do - for example a group may work on a certain feature or group of features, whilst another works on any relative designs, whilst another reviews each pushed feature and ensures it fits in with the code, and so on, concurrently.

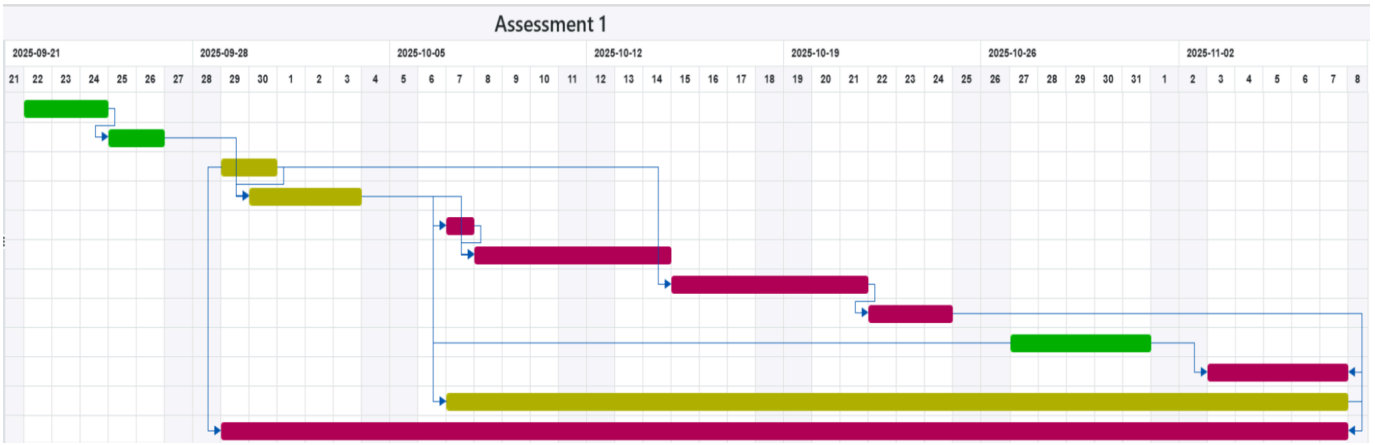
Project Plan

General plan:

- Reading Assessment document and Project Brief
- Designing customer questionnaire (relies on reading)
- Customer meeting (relies on above)
- Requirement write up (relies on answers from meeting)
- Create diagrams for Architecture (relies on requirements)
- Architecture write up (relies on diagrams)
- Method Selection and planning
- Risk assessment and mitigation write up

- Implementation Start (relies on everything being done)

In the image below, different colored columns represent different priorities:
(1) Red is the highest priority.
(2) Yellow is a high priority.
(3) Green is a medium priority.



To see more specific information of this Gantt Chart , you can enter this link :

https://www.onlinegantt.com/#/public/e34c5b59-8c2a-4e3d-a0a0-15498e7e45c1

In addition, the table below describes the tasks for each stage of the Gantt chart above :

Task ID	Task description
Assessment 1.1	Formed the team and made contact by using WhatsApp.
Assessment 1.2	A website with the corresponding group name was created (a project was created on GitHub and documentation was established on Google Drive).
Assessment 1.3	Previewed the project documentation, which included game requirements and project workflow.
Assessment 1.4	The project content was discussed, and tasks were scheduled. Additionally, a user requirements meeting was scheduled for the third week and customer questionnaire was designed.
Assessment 1.5	Participated in user meetings and collected user needs.
Assessment 1.6	Product and user requirements were summarized and their priorities were discussed, and a requirements document was created based on this.
Assessment 1.7	Different diagrams were created for Architecture (UML class diagrams, state diagrams, structure diagrams).
Assessment 1.8	Architecture was written up.
Assessment 1.9	Based on the tasks already planned and the future schedule, we begin risk assessment and analysis for the project.
Assessment 1.10	Risk assessment has been completed.
Assessment 1.11	The project plan is complete, and each stage has been marked.
Assessment 1.12	The implementation phase has been completed.