Method Selection & Planning

Cohort 1, group 9:
Chris Sewell, Fedor Kurochkin, Matt Durham, Max Peterson, Oladapo Olaniran, Wojtek Tomaszewski, Yuqi Fu.

Method Selection

As a team, we have adopted the Agile Methodology due to its focus on the client, keeping them at the centre of development. The Agile Methodology allows us to immerse the client in the development process and enables us to respond dynamically to changes in their requirements through iterative development. Iterative development means we develop the software in small stages, with each iteration improving on the previous one. In contrast, a fixed approach involves a single, inflexible plan. As students, we are inexperienced; it is unrealistic for us to plan every stage of our project before we begin. Consequently, an agile approach benefits us, as we can correct our mistakes in real time without having to restructure an entire plan. Furthermore, using an agile approach also allows us to deliver software quickly to our clients, ensuring that we meet their requirements and enable them to see their product frequently. Moreover, using an agile approach allows us as a team to break the project down into smaller, more manageable tasks, helping us stay organised and structured.

Since Agile is our overall philosophy, we will use the Extreme Programming (XP) and Scrum frameworks during the development process. We chose to use XP as our main development framework because it provides practical engineering practices such as pair programming, test-driven development, and continuous integration, to ensure that our game is reliable and adaptable. Additionally, XP enables us to maintain a simple implementation, with a focus on frequent releases and iterations. It is much more efficient to work on one feature at a time and have another team member test these while others are being developed. XP promotes these practices and prevents us from developing the entire project before we discover issues with the implementation.

Using Scrum as our project management framework provides a flexible way to apply agile principles. We can organise our work into sprints, keeping the project manageable as team members can focus on one small feature at a time. Additionally, scrums enable better team collaboration, where everyone knows what they're working on and how their tasks fit into the sprint goal.

Tools

Documentation

For our documentation, we chose to use Google Docs because it is free to use and allows for real-time collaboration, enabling multiple team members to work on the same document simultaneously. Furthermore, with all files stored on the cloud, team members can access documentation anytime, anywhere, and from any device. Additionally, Google Docs automatically saves all changes, eliminating the risk of losing work. Likewise, the version history feature enables us to easily track edits and restore previous versions if needed.

Microsoft Word was considered as an alternative way of documentation due to its ease of use and familiarity with all team members. However, we chose Google Docs instead due to its real-time collaboration capabilities, which are essential for an Agile team, as we can see

updates instantly and leave comments for discussion. Also, the auto-save feature is crucial for preventing the risk of losing work, which would waste time and reduce productivity.

Communication

As a team, we chose WhatsApp as our primary communication method because all group members were already familiar with the app and had it installed. Furthermore, using a mobile app enabled us to communicate quickly, reducing delays compared to alternative methods such as email. The app also works on tablets and computers, allowing everyone to stay connected wherever they are.

Additionally, we also chose to use the Discord app due to its accessibility across multiple devices and familiarity. Its extensive feature list supports our agile approach. Features like voice channels are useful for pair programming sessions as they allow for quick discussions without the need to schedule formal meetings. Moreover, creating channels for different conversations allows us to keep them organised. Consequently, it is easier for us to find information later without having to scroll through a single large chat, as is often the case with WhatsApp. Discord's file-sharing feature also allows us to instantly share assets, such as sprites or code snippets, by simply dragging and dropping them into the chat.

Version Control

We chose GitHub as our version control system because it supports code collaboration through the use of branches. These Branches allow multiple people to work on the same project simultaneously, which is essential for our agile approach. Branches will enable us to do this without accidentally overwriting someone else's work. Changes can then be merged smoothly using pull requests. In addition, GitHub maintains a full history of changes to code and documentation, allowing us to review and restore previous versions if necessary.

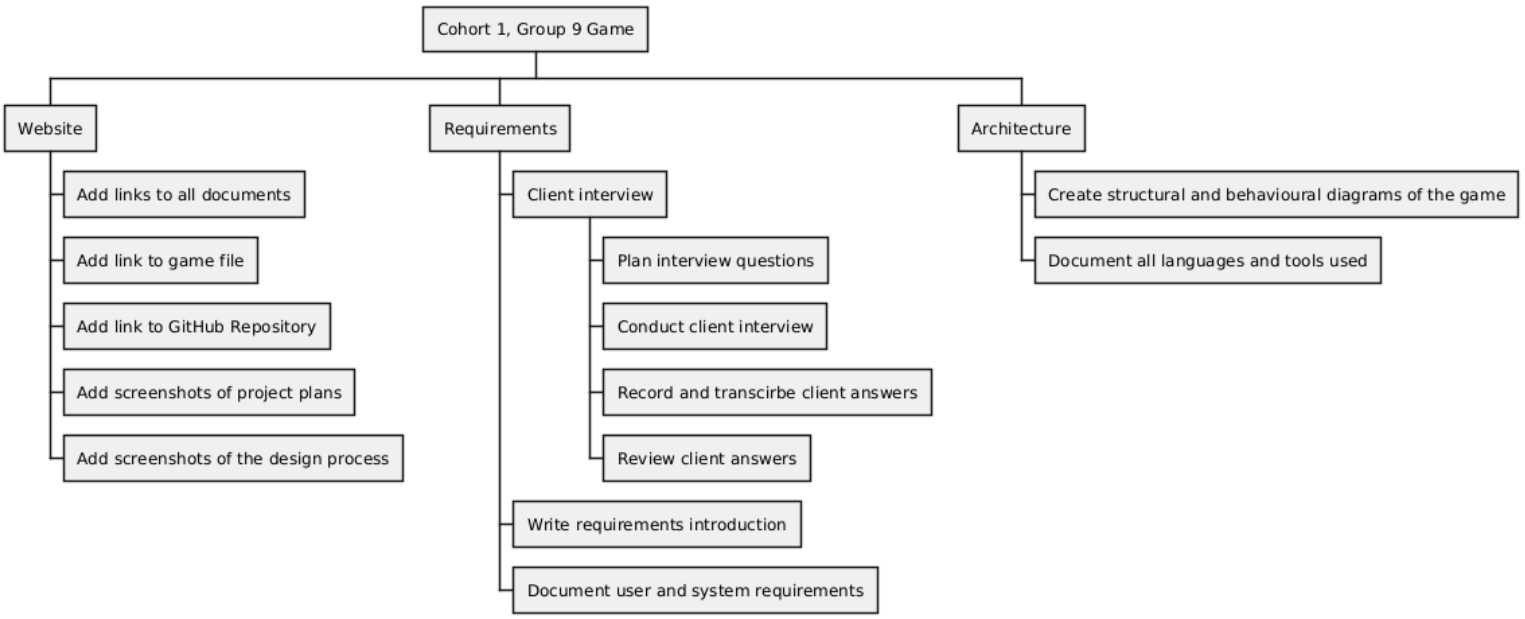
Another version control system considered was Apache Subversion since it is easy to understand and widely used. However, it encourages large commits, which contradicts our development framework of smaller iterative work. Furthermore, branching and merging are more error-prone compared to GitHub, where managing branches is easy. Because we are using an Agile methodology, frequent iterative development is essential; therefore, using Apache Subversion would not be the best approach.

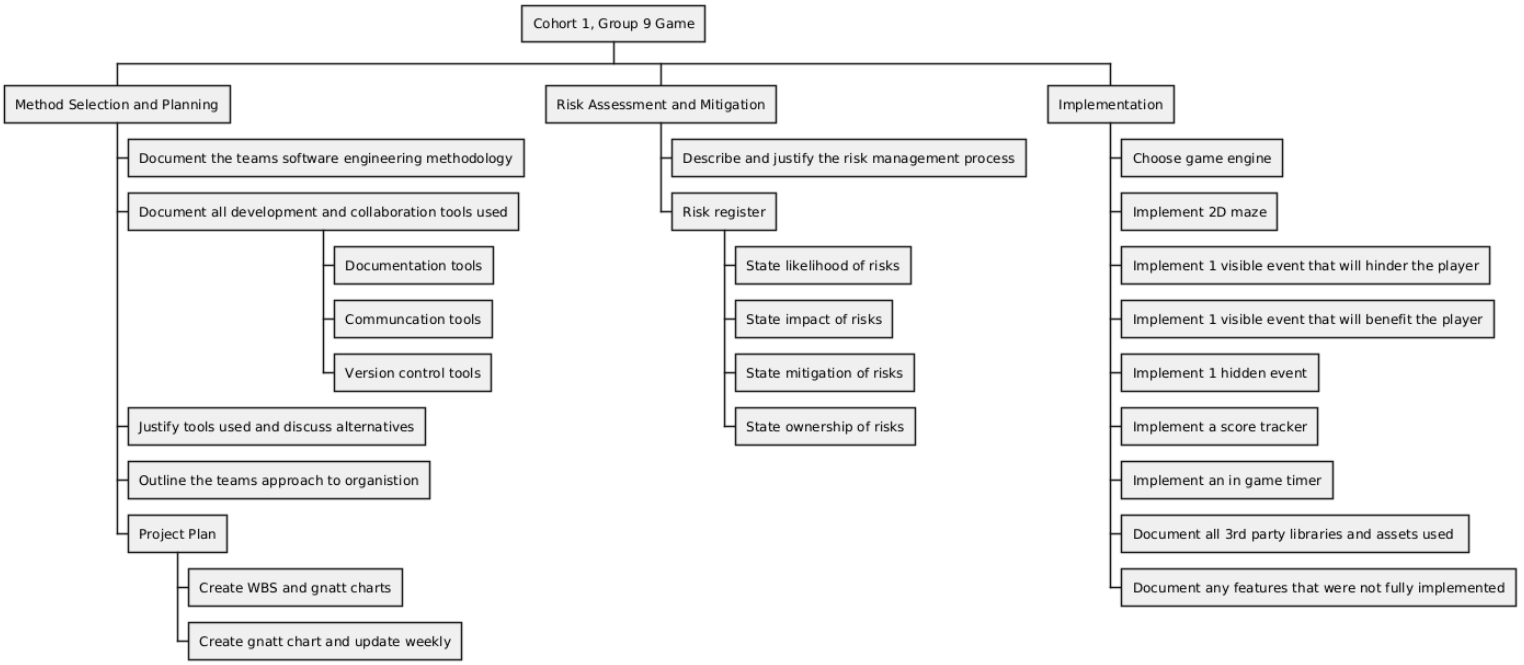
Organisational Approach

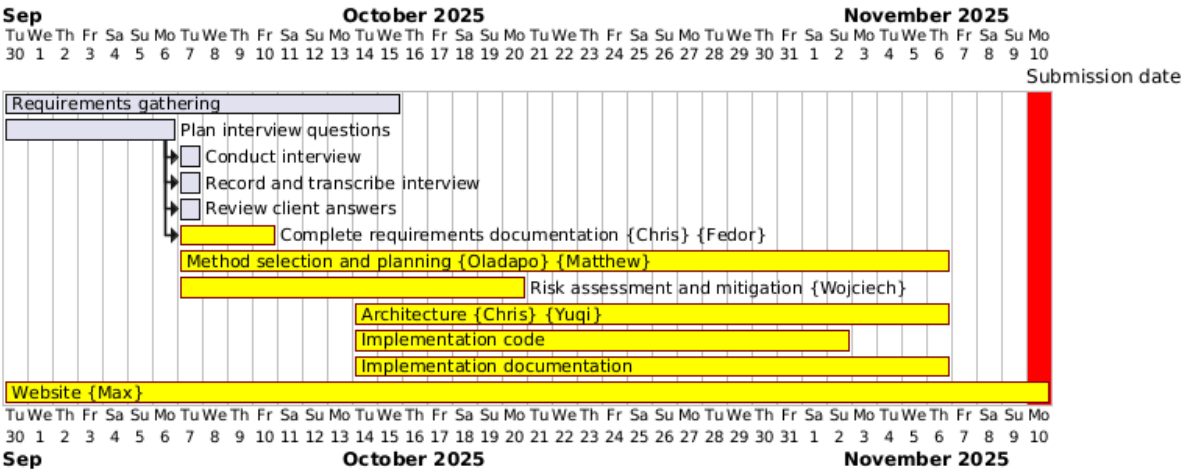
At the start of the project, we listed everyone's strengths so we knew who could do what best and could assign tasks accordingly. We decided this was the best approach because it builds teamwork and understanding, reduces confusion later, and helps the project flow more smoothly. Furthermore, it also highlighted gaps in our skills, which we can develop later on by taking on tasks that weren't our initial strengths.

Every week, we meet in person to review our progress on the project and set deadlines for the upcoming week. A Gantt chart is then created for that week, so we can visually see the tasks to be completed. This approach allows all team members to see what needs to be done and by when. Improving team coordination and making it easier to track our progress. Furthermore, it keeps the team accountable and helps us identify when a team member is struggling to meet deadlines, so we can take appropriate steps to address it. We assign multiple people to a task to increase the speed and efficiency with which we complete our set deadlines. Moreover, it improves accuracy and quality by having multiple people review the same work, ensuring our project is of the highest quality. During our meetings, we also use that time to make decisions about our project as a group. We chose this approach over a virtual one because we found it easier to communicate our ideas face-to-face than over a call. In addition, we can ask questions, clarify points, and reach agreements more quickly than during an online call.

Project Plan







During week 2, we first created an overall Gantt chart to gain a general idea of when we needed to complete each deliverable to submit the project before the deadline. Furthermore, we also created a work breakdown diagram to break each deliverable down into smaller, manageable pieces. Additionally, it helped each person understand what needed to be done for the section they were working on.

During week 3, we interviewed our client to gather requirements for the game. This allowed us to begin working on the requirements section of the documentation, which was essential for other parts of the project, such as architecture and implementation. Furthermore, we also realised that we needed to assign two people to implement the code instead of one, due to the project's scope and the limited time we had to fulfill the requirements. Because of this, one person was removed from working on method selection and planning and added to help implement the code as well. We also extended the time taken to complete the requirements documentation by a few days, as we initially underestimated the amount of work required to complete it.

During week 4, the requirements documentation was completed, allowing us to begin working on the architecture. Furthermore, it was at this point that we decided how we wanted the game to look, and coding for the game started.

During week 5, we found that the risk assessment and mitigation weren't completed by the time we initially stated, as the team member was ill. Therefore, we assigned a team member who had finished their tasks to help out with the documentation.

During week 6, we also found that the game implementation wasn't completed by the time we initially stated. However, we discovered that we still had some time before the submission deadline, so we extended it by a few days. All documentation was completed by the initial date we set, so team members without any work to do helped to complete the code to ensure that we met the submission deadline.