Development Method

We have decided to use the Agile method Scrum to organise our work and track our progress as it allows each team member to work at their own pace while still having regular check-ins to ensure work is getting done.

Weekly Scrum sprints fit well to our schedule and so are appropriate for our project and allow for meetings to be infrequent but productive.

The method also involves regular ceremonies all with different objectives and discussion points.The first being “Sprint Planning” where at the start of each sprint, we hold a Sprint Planning meeting. Define the sprint goal, prioritise tasks, and break down larger work items into actionable steps. This ensures that each team member can go into the sprint knowing exactly what it is they need to be working on and how they can go about doing it.

At the end of the sprint there are then two ceremonies that are useful for keeping track of how work is progressing.The “Sprint review” and the “Retrospective” , these are similar in terms of what they are discussing but are different in the sense of how we are looking at the the work across the sprint. In our sprint review we are reviewing what was accomplished by the team and demonstrating it to keep everyone on track and motivated with the general progression of the project overall. The Retrospective is used after each sprint to discuss what went well, what could be improved upon, and then make a plan for the next sprint. It’s a chance for the team to analyse and improve our workflow and learn from any mistakes we made.

Tools

File sharing and organisation - Google Drive/Docs and GitHub Repository

We chose to use Google Drive and Google Docs as our main collaborative workspace to organise and manage all the project elements. Google Drive was the most logical choice for us, as we all have university Google accounts, making it really easy to set up. This platform offers simple tools for organising files and folders on the shared drive, allowing us to keep everything from design documents to code snippets in one location that’s accessible to everyone in the group.

One major benefit of Google Drive is that the documents are stored online and updated in real time, which means we can work collaboratively, even when we aren't together. This real time updating feature allows us to work simultaneously without worrying about any version control issues or having to wait for one person to finish working on a document. The ability to add comments and suggestions directly onto documents also makes it easy to give feedback, share ideas, and discuss tasks without needing to be active at the same time. This way, each team member can stay informed and up to date with the latest progress, even if they weren’t involved in a specific update to a document.

We decided to set up a GitHub repository to manage and streamline our code development. GitHub is a powerful tool for version control, which allows our team to work on code simultaneously while minimising conflicts and ensuring that we have a reliable history of every change made to the project.

GitHub's version control capabilities are particularly valuable because they allow us to maintain a record of every change, revert to previous versions when necessary, and debug more efficiently. If an update causes unexpected issues, we can easily go back to a previous commit to restore functionality, which gives us a safety net for experimenting and testing new features. This is crucial in collaborative coding environments, as it keeps the project moving forward without major setbacks from unintentional errors.

Additionally, GitHub serves as a centralised repository accessible to all the team members, meaning we can pull the latest updates and stay up to date with each other's progress. This is especially helpful in preventing "merge conflicts," which can arise when multiple people try to edit the same part of a project. GitHub's built-in conflict resolution tools make it easy to manage any conflicting code changes.

General chats and updates - Whatsapp Group Chat

For our more general informal communication we decided to simply exchange phone numbers and make a Whatsapp Group chat. This made the most sense as it is easy to use and everyone will have access to it all the time on their phones. Also as Whatsapp saves and backs up chat data, we can scroll back through old messages and discussions if we need to revisit anything.

Discussed alternative

Our team initially considered setting up a dedicated Discord server for our main communication for the project. Discord seemed like a good option due to its popularity as a platform for collaboration, with features like voice channels, video calls, and the ability to create multiple text channels for different topics. However, after some discussion, we ultimately decided against it. One key reason was the layout and functionality of Discord, which felt less intuitive and user-friendly on mobile devices. Since many of us would need to access the server on our phones throughout the day, ease of use on mobile was a priority, and Discord's interface seemed to lack the simplicity we were looking for.

Also, we were aiming for a more informal, unstructured form of communication. While Discord servers can be organised with specific channels for different project topics (e.g., one for design, one for coding issues, etc.), we felt that too many channels could make communication feel segmented and potentially even reduce our interactions. The added structure risked creating an environment where team members might feel unsure of where to post updates or questions, leading to missed information and possibly reduced engagement from the team. We wanted a setup that felt more like a natural conversation, where people could easily share updates, discuss ideas, and ask questions without navigating a complex system of channels.

Team Structure

Team Organiser - Harrison

Responsible for Building/Updating the website
Organising all files and folders in shared drives
Tracking and summarising discussions from each meeting
Laying out a project plan and deciding on recognised methods to ensure work is structured and evenly distributed.
Ensuring everyone is able to complete work assigned to them in a timely manner.

Lead Developers - Tom & Will

Responsible for the implementation of the requirements.
Ensuring robust version control
Involved in laying out the general architecture of the game.
Most Familiar with the game engine.

Requirements Supervisors - Lisandro & Nathan

Responsible for eliciting and negotiating requirements from the client
Ensuring that all requirements are met to an appropriate level
Involved in translating the requirements to a formal architecture
Give a systematic and appropriately-formatted statement of user and system requirements.

Risk Manager - Letam

Responsible for identifying and documenting all potential risks
Describing and justifying the risk management process followed by the team and the format of the team's risk register
Formally outlining each identified risk to then decide on a risk owner and a mitigation plan

We opted for a non-hierarchical team structure as we felt that it allowed for a better spread of responsibility across the team where everyone is equally as important and responsible for the project work. Specifically for our team this made the most sense as we felt that each group member having independent responsibility for their own section limits one group member from encroaching on all elements of the project and potentially leaving out important contributions from other team members.

This more fluid structure obviously requires more frequent and strong communication between members as everyone will be working at slightly different speeds and in different ways. Which is why it was still important to have a main team organiser to facilitate communication and keep track of the progress across the many elements of this particular project.

Project Plan

The main idea for our project plan is to work in smaller groups across a number of different tasks in parallel. By dividing the project into distinct tasks and assigning smaller teams to handle them, we can maximise efficiency and make progress on multiple fronts simultaneously. This structure also allows team members to specialise in areas where they feel most comfortable or skilled. These tasks and groups were decided collectively as a team to ensure everyone is not only satisfied with their responsibilities but also aware of each other's roles. This approach creates a sense of ownership and accountability, helping each team member feel invested in the project's overall success. Before assigning tasks, we thoroughly reviewed the exam document and identified its main deliverables, which allowed us to distinguish between sections that could be completed independently and those that would require collaborative effort.

One of the primary tasks is attaining and laying out the product's requirements. This was a deliverable suitable for a pair in our team, as it involves gathering specific information, analysing it, and structuring it comprehensively. By having two people dedicated to this, we ensure accuracy and completeness in documenting what the product needs to achieve. This pair will be responsible for validating and organising requirements, identifying key objectives, and making sure that each requirement is clearly understood by the rest of the team to guide further development.

Outlining risks and compiling the necessary information related to each risk is another key task in our project. A team member has been designated to handle this component, as it involves assessing potential problems that could arise during the project's lifecycle. This task requires attention to detail and forward-thinking to identify possible obstacles, whether technical, operational, or external. By having a dedicated person to evaluate and document these risks, we can proactively address and mitigate them, helping to keep our project on track.

Designing and developing the software architecture is one of the most collaborative aspects of our project. Unlike other tasks, this requires the input and coordination of the entire team, as it will serve as the foundation for the software's structure and functionality. Since the architecture outlines how different components will interact, it must incorporate information from surrounding tasks, such as requirements and risk analysis, to create an accurate representation of our development approach. This phase is crucial, as it directly informs the implementation process and ensures that the design aligns with both the project goals and the technical constraints.

The primary software implementation of the product has been assigned to two designated "Lead Developers," who will drive the hands-on coding and development. However, this task isn't isolated—these developers will require continuous feedback and input from the rest of the team to ensure that the implementation aligns with the outlined requirements and architectural plans. The development process will be informed by the outputs of all other tasks, making communication and integration essential for a cohesive product.

Finally, organising and overseeing all of these concurrent tasks is essential for smooth project execution. Effective communication is crucial when team members work semi-independently, as it helps prevent misunderstandings and keeps everyone aligned with the project's goals. A designated project coordinator will play a vital role in maintaining this communication flow, facilitating updates across the team, and ensuring that each task progresses as planned. This coordination not only enhances efficiency but also fosters a collaborative environment where team members feel supported and well-informed throughout the project's duration.

Key Milestones

- Read through the product brief and determine a number of questions and clarifications we want made in our meeting with our client in order to determine our requirement. *To be completed by 01/10/24*

- Create a functional website that allows for our various project documents to be displayed and accessed. *To be completed by 04/10/24*

- Have a systematic and appropriately formatted statement of the user and system requirements following the client meeting *To be completed by 14/10/24*

- Have a structured and systematic presentation of all identified risks with their relevant risk owner and mitigation strategies. *To be completed by 14/10/24*

- Create a number of diagrams (structural and behavioural) of the architecture of our product. Including justifications for choices made and relevant links to our obtained requirements. *To be completed by 17/10/24*

- Have a simple prototype of the software to allow for discussions on the functionalities we want/need to implement *To be completed by 21/10/24*

- Have a finished piece of software that meets all the requirements for assessment 1 and is well documented *To be completed by 04/11/24*

A summary of our weekly meetings that outline our progress and changes to our plans is accessible via the teams website. Weekly Summaries