Method selection and planning

Group Name: Team 6
Group Number: 6

Ryan Bulman
Frederick Clarke
Jack Ellis
Yuhao Hu
Thomas Nicholson
James Pursglove

a)

Engineering Approach

We believe in using the agile approach for our software engineering project, this allows us to develop features rapidly, see if they work, test them and tweak them if they are deemed to be insufficient. The agile method allows multiple members of the team to work independently on their own tasks, then have their work reviewed and changed accordingly. This makes it significantly quicker and more efficient to create our codebase and have all features that need to be developed, developed to their highest quality.

Once all of the prototypes have been developed individually, we then compiled them into the main code base where everything was combined together to create the finishing project. Initially we implemented a waterfall approach to the development, where each feature was developed one after the other to create the product. We found that this became inefficient and the features weren't being developed quickly or accurately enough for what we wanted to do, this forced us to move on to a more agile approach, where we developed features almost cyclically.

Tools Used

We used Google drive and docs in order to sort out different roles in the group and keep track of everything we needed to do. This allowed each team member to remain on task, allowing us to complete the project before the given deadline. The Google drive contained shared files such as the deliverable outlines and the project brief, which were used to complete the project accurately and match the project guidelines. We also used this to write our questions to ask during the stakeholder interview, which made the whole process a lot more time efficient and a lot more worthwhile.

GitHub was one of the main resources we used in order for our team to keep up to date with everything and to share code written by multiple people. This resource really allowed us to create the project as a team, using things such as branches and merges to simplify and greatly improve the efficiency of the development cycle. GitHub lets each developer commit and push changes to the project to be reviewed and possibly added to the final product. This allows us to keep track of development, seeing who did what and at what time, revert any changes if we need to or discard them completely. Each person in the team was given a different role to play in the development of the game, for instance, one person was assigned the Player Movement task and another person was in charge of creating the shooting aspect of the player and colleges. This was really easy to do using GitHub, each task was assigned to a branch, developing individually on that branch so that no merge errors would occur until we were completely ready to merge them. Once each of the tasks were completed, we merged each branch to a different branch called 'combining', this allowed us to check that what we had done was valid and allowed us to make any changes if necessary. We then added any new changes or features that were required to make the project finalised, and pushed all of these changes to the 'master' branch once we were all completely satisfied.

Discord was essential for the development of our game; allowing us to discuss anything we needed to add, remove or change within our project. This tool is absolutely essential for communication as it allows us to talk not only in text form but in audio form too, giving us an alternative to in person meetings should anyone find it difficult to make it to any of those. Discord also allows for file sharing, we used this feature to send any images/artwork that we created which could then be easily downloaded and uploaded to our game assets.

The fitness of each collaboration method within our project shows us that we chose the right tools for the job as each of the ones we selected were perfect for all of our circumstances

and allowed us all to finish the project accurately and efficiently. For instance, GitHub was essential in the development cycle, giving the capability of sharing and modifying code written by multiple people synchronously. It also allowed us to view the history of the code base, giving us the opportunity to modify any of our work whenever we wanted to. However, we could have used programs such as GitLab or even tortoise git. The reason we didn't use GitLab was because most of our team had heard of GitHub and already had an account so it was seamless in our integration. We didn't use tortoise git because most of the developers liked the convenience of having a GUI to guide the process of the version control system, we weren't too keen on the idea of having to use the command line to do everything, even though this might have been the preference of some of the developers.

Discord was also essential in the development process, being our primary method of communication and idea generation. We used in person meetings to create ideas and talk over things, then we switched to discord later on in order to finalise ideas at a time more suitable for all of the developers. For this project, I would say that Discord was perfectly fit for our circumstances and for our needs. Again, we could have used different software tools such as Slack in order to keep in touch and talk over problems we may have had, but we decided that Discord would be better as everyone had an account and it was easily accessible and easy to use.

Google Docs was definitely the easiest thing to use to help us collaborate on documents that we needed to create. This is because if you have access to the internet and have been given permission to access certain documents, then you can use Google Docs. It's ubiquitous and easily accessible anywhere you are at any time. This made it the perfect fit for our project.

b)

Team Organisation

Our approach to team organisation was to have a group leader who had the most knowledge of software development going into the project, they would help the rest of the group with any problems or any issues they had with any of the code. We delegated tasks as a group depending on who felt comfortable doing what, if somebody felt more comfortable writing a lot of code, then they would be given a more challenging problem to solve whereas if they weren't as confident, they would be given a less challenging problem. We were all kept on track by each other making sure each person got what information they needed and any relevant advice was given to them accordingly. We all had to communicate well in order to get our views across, this was done using a conjunction of in person meetings and online meetings and messaging via discord. The approach of having a leader yet a flexible and communicative team, allows us to freely share information, ideas and accurately and efficiently complete tasks. This is because the team leader makes sure everyone is on track, on task and everyone else has the freedom to speak up if something isn't right or they have an idea that they want to share.

This approach to team organisation worked well because each member of the team knew exactly what they needed to do at any given moment as they had a lot of support around them. If it was just the team leader who was supporting, it would have been a lot more difficult to complete the tasks as we would have only had one point of contact. So, we had every member in the group willing to offer help and support if needed. Our group really got to know each other during the forming stage of the team working process. We talked to each other, communicated our strengths and what we would be willing to do during the project. This was the time to decide who would be doing the most programming and helping the other members start to generate ideas and really start thinking about the project.

During the storming stage, we created mind-maps and diagrams to help visualise the ideas we had in mind and how exactly we might implement them later down the line. Throughout the whole process, communication was key, especially during the initial stage of the first generation of ideas. Once we felt like we were all part of the team it was the norming stage, this allowed each individual to know each other's strengths and what they were going to contribute to the project and how that was going to affect everybody else and in what way. Lastly, we transitioned to the performing stage of the process, we were confident in each other and that each member was going to create work to their highest quality and on time for the given deadline. Criticism was handled well in the group, we didn't do it maliciously, it was done in a way such that all parties understood all points of view, then we came to a healthy compromise. This made the project run a lot smoother, giving each individual the confidence to speak up about any ideas they might have, this proved necessary as all developers gave some great ideas and points that we implemented into the overall system.

c)

Plan and Description

The plan changed drastically during the holidays. Our initial plan had us making progress with the code for the game during the Christmas break, but circumstances occurred which pushed our schedule backwards and put a lot of time strain on the project. These are the things that project managers should expect to happen and should proactively carry out measures to help prevent any lasting damage. In terms of our project, we knew that there were going to be exams straight after the break, this meant we had to do a lot of planning beforehand in order to make sure everyone knew what they could be doing after the break.

We used a gantt chart to make our plan:

- The bright yellow blocks represent work packages that correspond to deliverables
- The dull yellow blocks represent the task that make up each work package
- Red arrows signify dependency between tasks

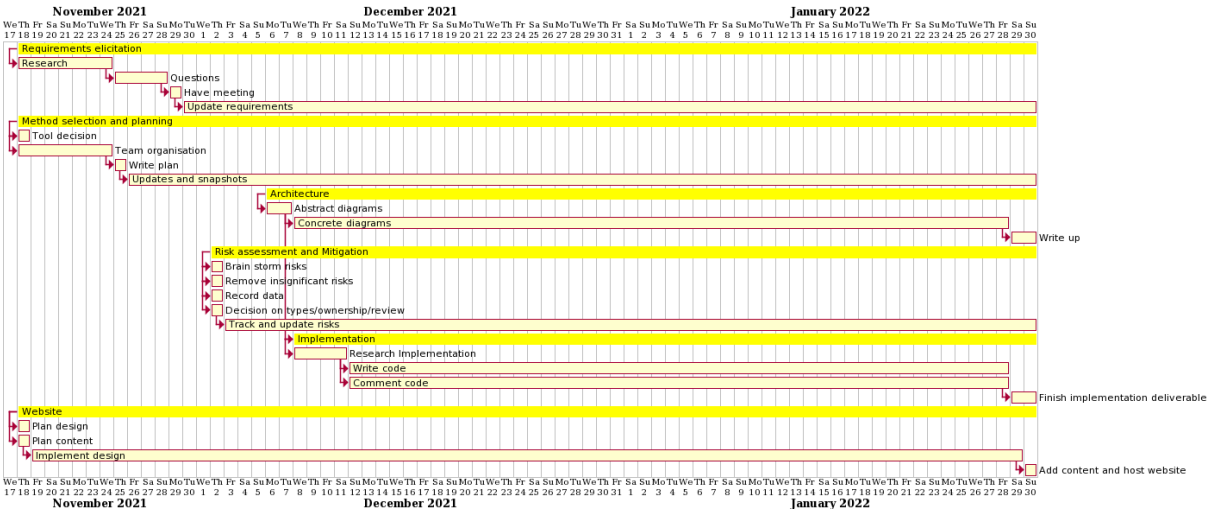

Figure 4.1: Plan before the christmas break

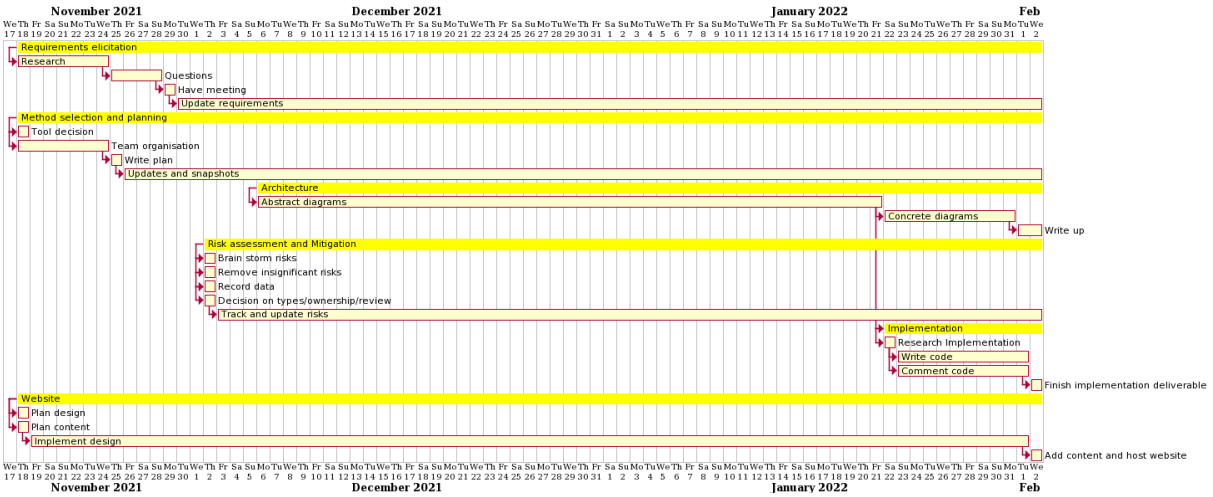

Figure 4.2: Plan after the christmas break