Method Selection and Planning

Team 15

Aous Abdulla
Byron Morris
Eddie Barun
Ethan Griffiths
Theo Toth
Weihan Zhu

4a)
We used a Scrum framework to plan our project, Scrum involves a few separate phases during the life cycle of the project. They are:

1. Planning: This phase involves defining the goals of the project and determining which tasks need to be completed to achieve those goals.
2. Development: During this phase, the team works on completing the tasks in the product backlog. They do this in short, focused work periods called "sprints." Each sprint typically lasts two to four weeks, and at the end of each sprint, the team should have a working product that has new features or functionality.
3. Review: At the end of each sprint, the team conducts a review meeting to demonstrate the work that has been completed and to get feedback from stakeholders.
4. Retrospective: After the review, the team conducts a retrospective meeting to reflect on their processes and identify areas for improvement.

Throughout the Scrum process, the team had regular meetings called "scrum meetings" to discuss progress, identify and resolve any issues, and plan for the next sprint. The team had a "scrum master" who was responsible for facilitating the Scrum process and helping the team stay focused and on track.

There were several benefits to using the Scrum framework to plan our project. Some of these are:

1. Improved focus and productivity: the framework is centred around short, efficient work periods(sprints). These allowed the team to identify areas that would take longer than anticipated and divide large difficult to face tasks into smaller easier to complete ones.
2. Greater flexibility: Scrum allowed us to change and adapt the time scale of certain areas of the project as time went on. Some tasks look less tiem than anticipated and other took more. The scrum meetings allowed us to change and adapt as the project evolved.
3. Enhanced collaboration and communication: The framework allowed the group to discuss and evaluate certain areas of the project that they are struggling on. It encourages a level of communication where issues that are presented in the sprint, can be addressed in the meetings. Meaning overall a better end result and less misunderstandings.

There are multiple sources of research into the affectiveness of scrum such as a study conducted by Klein et al. (2007) which found that Scrum encourages a higher level of productivity, quality, and satisfaction compared to other more traditional approaches. Another study by Westerlund et al. (2008) also came to the same conclusion.

We used a variety of tools for development and collaboration, these included Github, Google drive, Discord, Intellij, Gradle for java and OpenJDK 17. Below is a brief justification of the tools that we used and a more in depth analysis of the more important ones.

GitHub had a number of features that made it useful in this context of a group programming project. These include:

1. Version control: GitHub utilises Git, which is a version control system that tracks changes made to the code. In turn allowing separate members of our team to work on the same code at the same time without overwriting other's changes. It also makes reverting back to previous versions, if there is an overlooked error, easy.
2. Collaboration tools: GitHub provides a large range of collaboration tools, such as pull and push requests along with code reviews, which we utilised. These helped with dividing up different tasks, the pull requests allowed team members to put forward their code to be merged onto the main branch, and the code review allowed others to inspect the code before allowing the changes to be merged. This meant that there was an element of

security, all code was reviewed by 2 people who had to approve the changes, to make sure the merge wouldn't break or interfere with any other elements of the code.

Overall, GitHub provided a range of features that helped our team work effectively and efficiently on the project. Github was the best version control system and most of the team were familiar with it.

Google Drive is a cloud-based file management system that allows collaboration and other key features. We chose to use Google drive due to:

1. University subscription: we all have access to google drive, do to the university providing it to us and have access to a vast amount of storage that we will need to use in order to store all the files we will create in the development of our project. It is also a product that the whole team is familiar with.
2. Accessibility: Google Drive can be accessed from any device with an internet connection, which made it easy to access files, such as this one, from anywhere. This meant that we were able to continuously work on the project from home, at university and anywhere else.
3. Real-time collaboration: Google Drive allowed us multiple members of the team to edit the same document at the same time. This was particularly useful for certain areas of the project where multiple members were working during a meeting in the same document.

Overall, Google Drive was convenient and the fact that it was something everyone was familiar with, had access to and understood, is why we chose it as opposed to alternatives such as dropbox or OneDrive.

Discord can be used effectively for group programming projects. We chose Discord as a form of communication due to the fact that it had features such as voice and text chat, for online meetings. Group channels, for dividing information on separate sections of the project, for ease of access. Discord is a tool that the entire team was familiar with and it just made sense to use it, everyone does anyway.

Gradle was a easy choice for the group as a build tool this was due to a number of reasons:

1. Build Speed: Gradle uses incremental builds alongside parallel execution which drastically increased build speed, which makes it fast and efficient when compared with the alternatives,
2. Dependency management: Gradle provides a good number of dependency management features, this allows easy navigation and management of external dependencies and their transitive dependencies.
3. Good integration: Gradle has a good integration with other tools and systems that we utilised such as libGTX and GitHub. Because of this continuous integration at later stages will be easier to achieve.
4. Flexibility: Gradle makes it easy to create custom tasks and implement plugins which. In turn makes it faster and easier to fit the needs of our project.

IntelliJ was our choice of IDE for this project due to a number of reasons:

1. Integration with other tools: intelliJ has a well built integration with other tools, such as GitHub our chose version control system, this made it easy to interact with GitHub.
2. Easy to use: IntelliJ is an easy to use IDE with a clean and intuitive UI that makes it easy to find specific tools that might be needed.

References:

- Klein, G., Karlsson, J., & Mathiassen, L. (2007). Scrum as a framework for project management: An empirical investigation. Information and Software Technology, 49(10), 962-972.
- Westerlund, M., Nosek, J., & Lyytinen, K. (2008). Agile software development methods: Review and analysis. The Journal of Systems and Software, 81(8), 1275-1296
- https://www.scrum.org/resources/scrum-guide

b) The teams approach to organisation. Each member of the team had a different skill set that would be more appropriate for certain tasks that needed completing. We assessed through discussion what our own strengths and weaknesses were in the initial practicals. These became more obvious and other merits and shortcomings of the group became apparent as time went on. There were 6 separate sections that were required for the first assessment: The website, the requirements, architecture, method selection and planning, risk mitigation and implementation. We decided to split them in a way that made the most sense.

- Theo: Theo was identified as being the best programmer on the team and so took charge of the implementation section. Theo did most of the technical work due to his skills being more apt to the task than others.. Theo understood the architecture of the system, as he did most of the leg work in creating it, and so was heavily involved in the architecture section of the project as well, both 3a) and 3b).
- Byron: Byron had previous website design experience and so was tasked, alongside ethan, in creating the website, maintaining and managing it as the project progressed. Byron was also tasked in completing section 3b), Byron has previously completed similar tasks on other projects, his prior experience made him more adept and suitable than other members of the team .
- Ethan - Ethan was involved in a minor scope within the requirements section as another perspective that was helpful in identifying areas missed by other team members. Heavily involved in the website development due to previous experience making him a good fit for the role. He also developed the base libGTX project that Theo used, this meant that he was a good fit to develop the architecture diagrams as he understood the implementation.
- Aous: Aous was tasked with 2 sections, the requirements section and the risk assessment section. He undertook the requirements section with Eddie, being present and engaged in the interview he understood what was required and needed by the system. The risk assessment was also undertaken by Aous as he presented an affinity for understanding what was needed by the system, and so by proxy what could go wrong with it.
- Eddie - Eddie was tasked with the requirements section along with Aous, Eddie noted down the interview and condensed it, this meant that he had to understand the important aspects of the interview, what was needed by the system, which made him a good fit for the role. He also was tasked with the method selection and planning stage of the project. Eddie organised and booked meeting rooms and meeting times. As team secretary he was responsible for noting down what was discussed in each scrum meeting. He also generated a list of tasks for the following sprint, for each team member.
  Weihan: Weihan created the first draft of the requirements section 2a).

Scrum meetings: At the end of each sprint, in the scrum meeting, we would all discuss what we had done during the most recent sprint. Here there would be alot of collaboration. So even though the gantt chart shows that certain members of the team were assigned to certain sections. Most team members had a say in most elements of the project. Specifically, larger, more text based questions: sections 2a; 3b; 4(a,b and c) and 5a. During the scrum meetings the status quo was to scrutinise and evaluate all the elements of the questions that were not the best they could be. We all took the approach of ‘tear it to shreds’ this meant that we could voice opinions openly ensuring that we had, simply, the best version of the sections that we could make.

c) The team used Gantt charts to map and visualise the progress of the project. Included in this document is the first Gantt chart (diagram 1) and the final Gantt Chart (diagram 8), the rest of the Gantt charts, along with their timelines can be found on the team's website.

The priorities in our Gantt chart's were developed in a simple way. Throughout any time period in the Gantt charts, the higher priority goes to the tasks that are on the top, and the lower priority on the bottom. This wasn't very clear for future tasks in the beginning, as not all subtasks and details of different aspects were discovered. However, in the last diagram (Diagram 8), the picture of the group plan and action is complete.

The Gantt chart allowed us to have a structure for our plan, this allowed us to save time on planning and put more time into the project. This was a very efficient way to plan the project. It allowed the team to plan the whole project and account for changes in duration as well as the addition of subtasks to make it more detailed (Obviously the creation of the Gantt Chart was not included in the Gantt Chart!). Throughout the project, the Gantt chart was almost always adjusted in every meeting as certain tasks would be started and more details about the subtasks would be revealed.

Within this section we chose to utilise the capabilities of the Gantt chart due to a number of reasons. The main ones being:

- Visualisation: Using a Gantt chart is an easy way to represent the timeline and schedule of the project. It is a tool that makes it easy to communicate what tasks are currently being undertaken, and when they should be completed by.
- Tracking progress: Gantt charts are an easy way to track what has changed over the duration of the project. What sub-tasks were born over the course of the project and how timings differed from the initial plans.
- Planning: Using a Gantt chart also allowed us as a team to work out rough deadlines in the distant future and more concrete ones between sprints. This allowed the team to be more confident in the timings and development of the project, knowing it would be completed on time.

SEE BELOW.

Diagrams of the first (D1) and final (D8) Gantt charts.

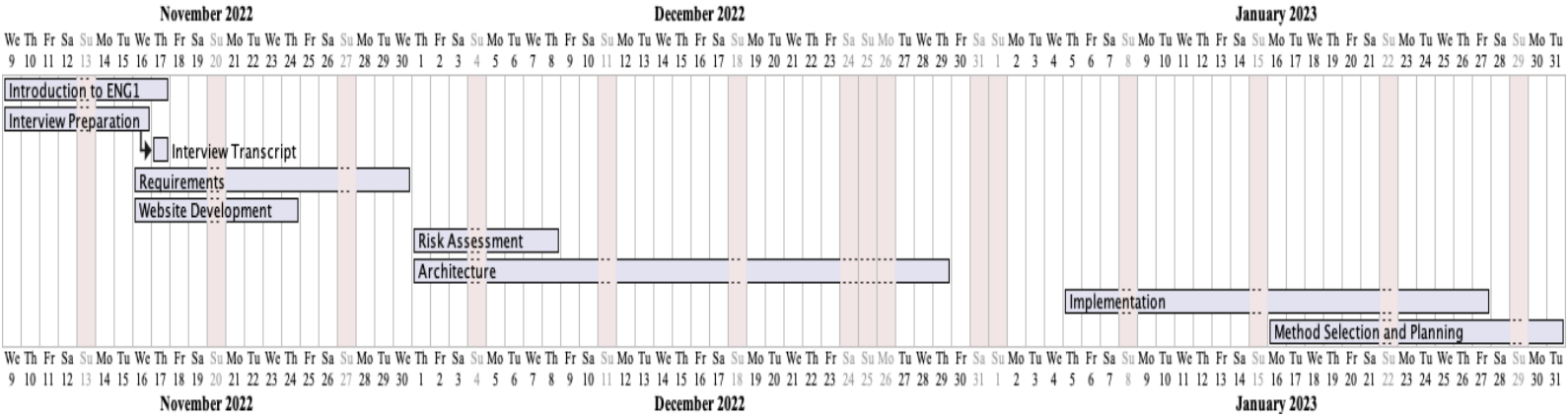

Diagram 1.

Diagram 1.png

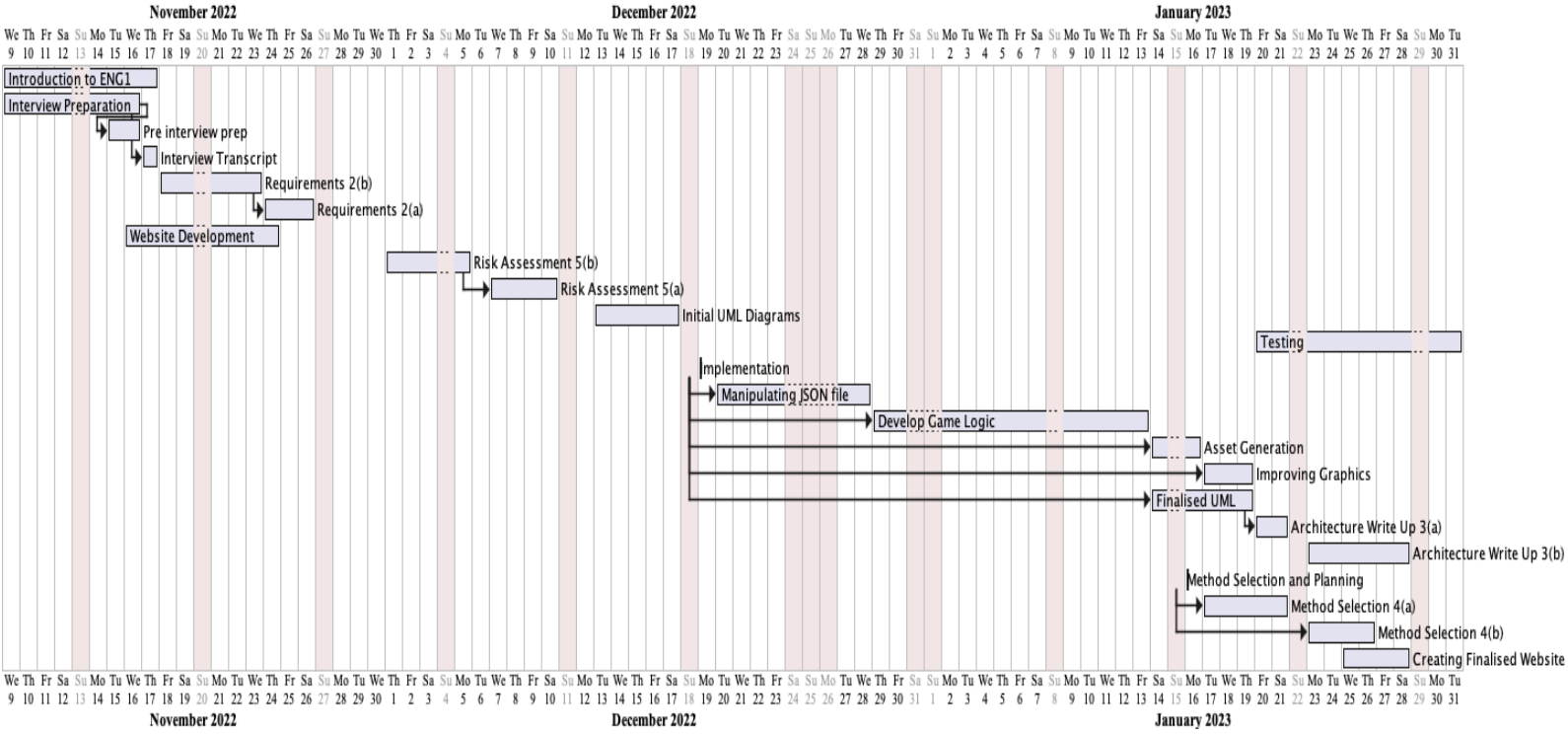

Diagram8

Diagram 8.png