ENG1 - Assessment 2

Updated Method Selection and Planning

Plan2.pdf

Group 4

Mikaella Loppnow	ml2708
Tom Daly	td1026
Ethan Buss	eb2225
Dillon Pandya	dp1195
Ereife Odusi	ed781
Harriet Kirby	hk1114

In this project, we are choosing to use agile methods. There are many reasons agile methods fit this project best. Firstly, agile methods produce working software frequently while adapting to changing requirements [1]. The assessment 2 requirements will fit in much better with agile methods than plan-driven methods, as plan-driven methods do not adapt to change nearly as well. This project does not require the benefits of plan-driven methods, like their use in managing large teams and increased focus on documentation. Documentation is important for this project, especially when transferring projects between teams, but it is not worth the overhead to focus on it as much as plan-driven methods require. Our team is of a small size, so the benefit of a central plan is lost as communication is much simpler. Additionally, agile methods allow us to engage with the customer better, as we can produce a functional game for them to provide feedback on. This is particularly useful as we will get many opportunities to communicate with our customer in labs, or meetings when necessary.

We will be using scrum as our agile method of choice. The high-level nature of scrum allows us to tailor it to fit our specific needs, and therefore work most effectively [2]. This advantage over other methods such as extreme programming made selecting scrum the best choice for us.

We will have week-long scrum sprints with a meeting before and after each, where we will discuss plans for the sprint before, and progress after. Focus will be placed on changes which will need to be made for the next sprint if not enough tasks were completed. The daily scrum can be discussed between collaborators on specific tasks to denote how their work is progressing when necessary, but is not strictly required due to the collaboration tools we are using to support the project.

The two main tools utilised for collaboration on this project are Google Workspace for working on documents and GitHub as our frontend for git for version control.

Google Workspace includes Google Drive and Docs, which are both vital components. Drive hosts all of our files relating to documents, mainly Docs files for the assessment and other files including notes and links. It was chosen due to its tight integration with our already existing university emails, so everyone already could easily access and utilise it. Additionally, it provides convenient integration with Docs, which is important due to how much it is used for the project. We chose Docs for several reasons: it is available in the web browser which makes it accessible for everyone, it allows simultaneous editing of documents by multiple people, and it has a detailed version history showing all changes made to the document. This last feature is particularly useful in the case that evidence is required. We considered whether we should use git to version control our whole project, using LaTeX. However, we decided Google Docs was much simpler to use, especially due to higher experience with it in the group compared to LaTeX. For version control in the actual development of the software, git is the software we selected. Git has considerably more advantages when used for software development version control compared to if we used it for the documents; to name some: allowing pull requests, forks, cloning to the local machine to be edited with an IDE and executed, advanced merging, and continuous integration. Therefore, something like Google Workspace was not considered for this part of the project. We looked at other version control software briefly before settling on using git. Git's decentralised nature compared to other version control like SVN is beneficial when developing. This as well as more experience in the group meant using git was the logical choice for us. Git will be very useful during scrum sprints, as it allows for easy modification of the software by multiple developers.

To work with git, we are using GitHub for our central repository to allow for better management of continuous integration, documentation, tracking of issues, and pull requests. We selected GitHub, as it has many very useful features for collaboration, is simple to get started with, many group members have prior experience with it, and it provides the ability to easily configure a website using GitHub Pages. The tracking of changes on GitHub will be important during scrum sprints, as they allow people to view the daily changes in a much more concise and simple manner. Additionally, continuous integration will allow for the easy deployment of functioning software at the end of each sprint, letting the executable be accessed easily by everyone and shown to the customer.

Team Organisation

Our team made a collaborative decision early in the development process that we would avoid officially assigning specific roles to each team member; instead we chose to favour a more flexible approach wherein members all contributed equally. This does not mean that we did not favour unofficial positions in the team, which were filled naturally as the development proceeded.

For example, some members were more confident in speaking - particularly in the first few meetings - and so naturally rose to unofficial leadership roles within the team. Other members were more creative and had confidence in those skills, allowing them to participate in the team's branding (i.e., our name and logo): this gave us a clear identity as THEEMD from the early stages of our development. Other members were less outspoken but more than happy to pick up work on various aspects of the project, leading to an overall comfortable environment.

We also prioritised ice-breaking activities in our first few meetings, which allowed less outspoken members to feel comfortable voicing their opinions, enabling our team’s key ethic of equality between members to thrive. The team thus did not consist of a hierarchy, but instead consisted of a “round-table” approach.

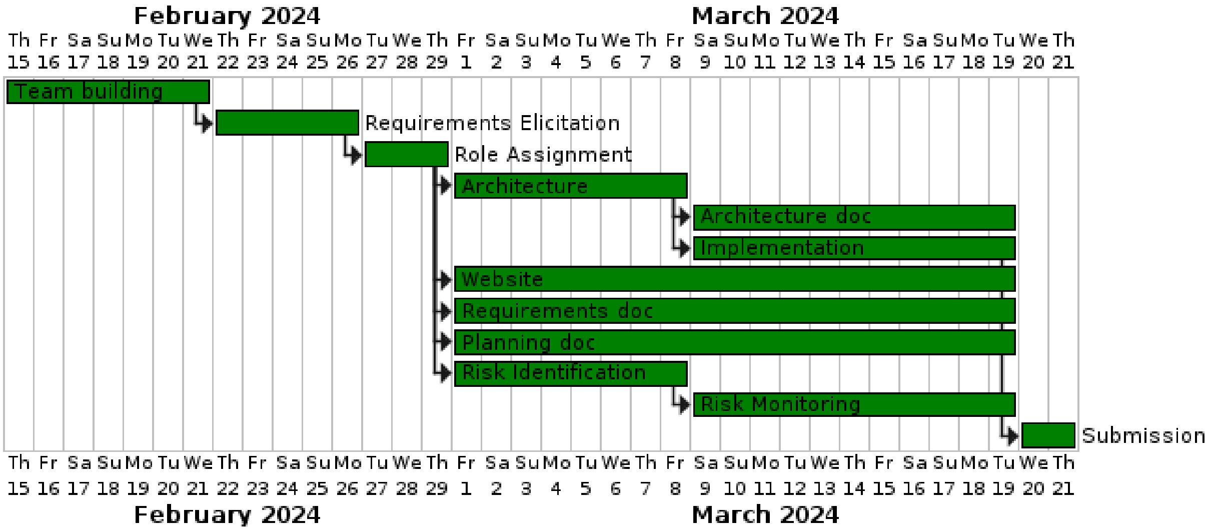
We also decided on a set time and location for weekly meetings outside of the practical session, which all group members made an effort to attend - these opportunities to check in on progress in person as well as the lower pressure environment outside of an allocated teaching environment lead to our team becoming comfortable with one another faster and also enabled issues / developments to be discussed in person. We chose to value in person communication where possible, as it is often more comfortable and less formal than communicating via either WhatsApp or Discord.

Overall, this was massively beneficial to our team’s motivation and created a comfortable atmosphere to work in. We ensured that all team members felt that their opinions could be voiced and would be valued just as much as anyone else. Our team ended up really feeling like a “team” instead of just a group of students set to work together to achieve a common goal.

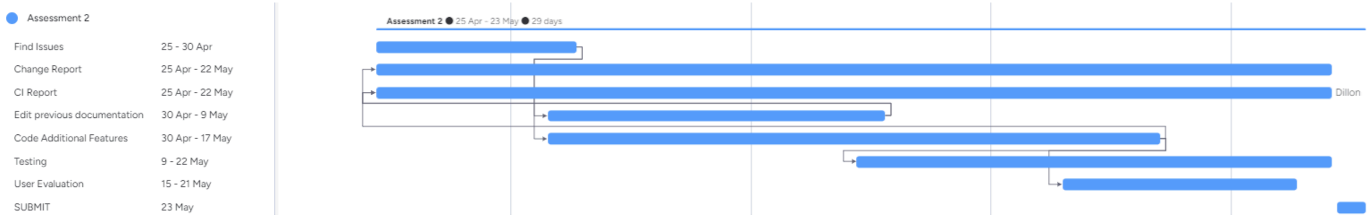
This overall approach enabled all group members to focus on areas most befitting their strengths and thus each member was able to utilise their full potential, as they felt comfortable to express an interest in what best appealed to them. This allowed people to develop aspects of the project that they genuinely cared about, leading to an overall higher quality of work as well as motivation amongst our team. As we made an effort to value each other’s strengths and weaknesses, and to work together when things didn't go as planned instead of attempting to blame each other, the team became a significantly more productive environment and far less stressful than it could have been.

Gantt Charts

The following Gantt chart shows the timeline that the previous group followed up until the deadline of Assessment 1.



The next Gantt chart shows the timeline that our group has followed until the deadline of Assessment 2.



Assessment 2	
Find Issues	25 - 30 Apr
Change Report	25 Apr - 22 May
CI Report	25 Apr - 22 May
Edit previous documentation	30 Apr - 9 May
Code Additional Features	30 Apr - 17 May
Testing	9 - 22 May
User Evaluation	15 - 21 May
SUBMIT	23 May

References

[1] I. Sommerville, “Agile planning” in *Software Engineering.* Harlow: Pearson, 2016, pp. 680-682.

[2] K. Schwaber and J. Sutherland (2020, Nov.). *Scrum Guide.* Scrum Guides [Online]. Available at: https://scrumguides.org/scrum-guide.html [Accessed: 10 March 2024].