Engineering 1

Group Assessment 1

Planning Document

Cohort 2 Team 12

Part A: Method and Tool Selection

For our project we decided to use the Scrum method as we are delivering a small piece of software with a small team, and it is essential for us to have a flexible working environment due to team members' different schedules. Scrum is an agile method, and we chose agile because there may be changing requirements, and we believe it is crucial to have constant team reflection and development.

Scrum involves dividing development into 'sprints', each of which are a week long and preceded by a planning meeting. At the end of each 'sprint' and often much more often, there is a team review of what everyone has done and any problems they came across. "Scrum employs an iterative, incremental approach to optimize predictability and control risk."[2]

We used 4 main tools for completing tasks on our project:

- Trello for immediate task management,
- Google for writing up documents,
- Discord for communication and scheduling of tasks, meetings and weekly focuses (announcements),
- Github for source control and code development/maintenance and as result feature management and review.

We used Github (alongside Github Pages) as the main hub for our project. It is where we uploaded the game files and relevant documentation. For our team's website, we decided to use Github Pages (although alternatives such as Google Pages were discussed).

Since our project was uploaded onto Github, we thought it would make sense to use Github Pages alongside it, as opposed to Google Pages.

Our team started off with emails and Google Chat as initial communication, but as time went on, we switched over to Discord because the majority of our team was very familiar with it, compared to the alternatives. Zoom was used during official supervised practical sessions.

Github is very suitable for agile software development as it allows multiple people to work on features of the code simultaneously with their own local repositories and then push and pull changes to/from the main/global repository on the Github server. All pull/merge requests can be reviewed and all changes are labelled so that it is clear what has been done at each stage of the work. Developers can also create branches with new code and then merge the branch with the main to include it, which makes changes easy to do and there is no risk of breaking the working code in the main program. Github makes it very easy and efficient to do this.

Google Drive is a great tool for aiding agile development as it allows all team members to see changes that have been made and access all files at the same time. We chose Trello for keeping track of our project as it is straightforward to use and encourages cooperation on tasks, and information can be adapted easily.

Initially, our team used Google Sheets instead of Trello to create and assign tasks for team members, however, after we discovered a more dedicated platform to do such things in, we switched over to Trello. It was both easier to use and easier for team members to see what needs to be done by them.

Part B: Team Organisation Approach

Team meetings happened at least twice a week (Mondays and Thursdays) on Discord and Zoom. At the start of the meeting, we would discuss what had changed since the last meeting and catch everyone up to speed.

In the case of meetings where not every member of the team was able to come, the team would communicate key and important ideas throughout the Discord channels. The Discord server we created had multiple channels for different parts of the project (See Appendix, Figure 1).

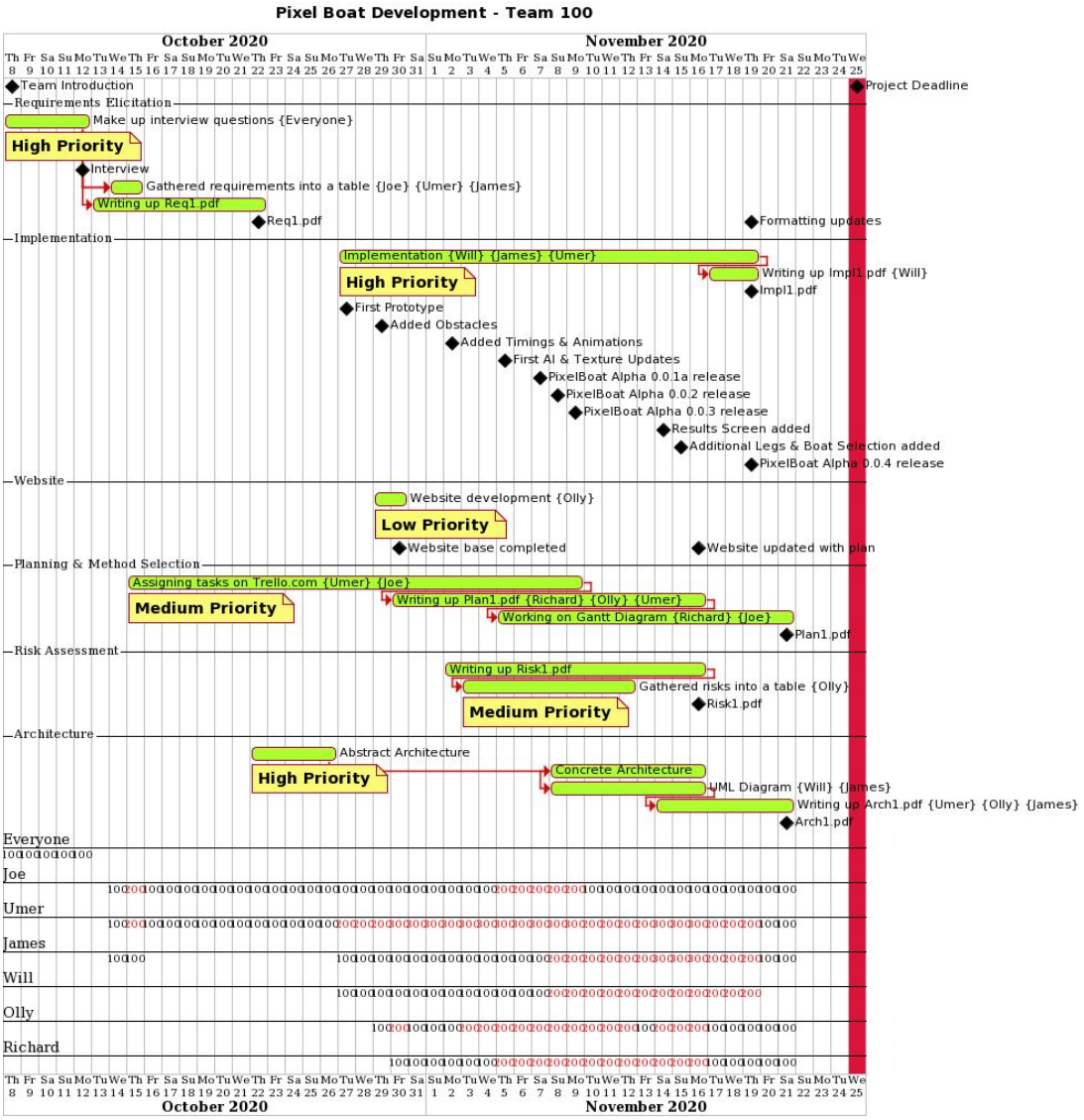
Our team was split into subteams for all the different parts of the assignment (coding, working on documentation, etc.) meaning that each subteam was debriefed separately by the team leader.

Each task had a leader as well as one or many collaborators to ensure good implementation. These subteams were very adaptive, changing people often, given the required skill set for the task. All tasks completed were peer-reviewed by other members of the team. Since we used Discord for most of our internal communication, feedback was received very quickly.

Our approach is appropriate for the project because it involves frequent and plentiful communication and feedback between team members which is essential for building software like this. The project is small overall, but involves lots of different parts, so this approach is perfect as it permits us to organise into sub teams and get things done efficiently.

The approach is suited to our team because we have different skill sets - some team members were more suited for building the foundations of the code base whilst others were tasked with feature programming. Some tasks required documentation or building websites and we worked well independently as well as in subteams which were tasked with tackling goals which required subtasks to be completed. These subtasks were discussed and reviewed in large team meetings when possible and in asynchronous Discord reviews.

Part C: Project Plan



Gantt Diagram of the project

Plan Strategy & Outcome Review

The plan evolved through ongoing meetings and “Scrums” where we discussed what tasks needed to be completed next and what each team member had been doing. Initially, we had a rough idea of what the plan was from our critical path, and after the first couple of weeks we elicited an actual plan using a Gantt chart (shown above). With our agile development methodology, we altered the plan frequently to adapt to any changing circumstances we came across such as absence of team members and technical issues, and some examples of weekly changes to the plan are shown on our project website. Furthermore, with the current pandemic, we had difficulties such as the presence and ability of team members being affected. However, we overcame these issues by keeping in contact daily with Discord and using Trello to keep track of tasks.

Key Events List

These key events originate from the Trello board and the Gantt diagram. Highest priority events were Implementation and Architecture as these sections held the most marks. Implementation was also dependent on Architecture.
The critical path of our project was Requirements Elicitation first followed by Planning, Architecture, Risk Assessment and finally, Implementation.

First Meeting

- Introduction to members & made a project notes document
- Scheduled Customer Meeting for Requirements Elicitation interview and prepared for it

Customer Meeting and Debrief

- Assigned half the group to taking notes, other half asking questions
- Review of how the interview went and what could have been improved
- Further team meetings scheduled

Requirements

- Requirements Elicitation (2a):
    - Introduction to how requirements were elicited
    - Description of customer inquiry approach taken and outcomes
- User Requirements Tables (2b):
    - Created list of preliminary requirements and added some initial MSCW priorities
    - Reviewed preliminary requirements and formatted them into tables with full details and priorities

Architecture

- Initial UML Charts produced by development team (Abstract)
- Ongoing explanatory text produced (Concrete)

Risk assessment and mitigation

- (5a) Wrote an introduction and justification for risk formatting and level of detail
- (5b) Devised tables for presentation of risks using Google Sheets, giving each risk a severity level and owners.

Method selection

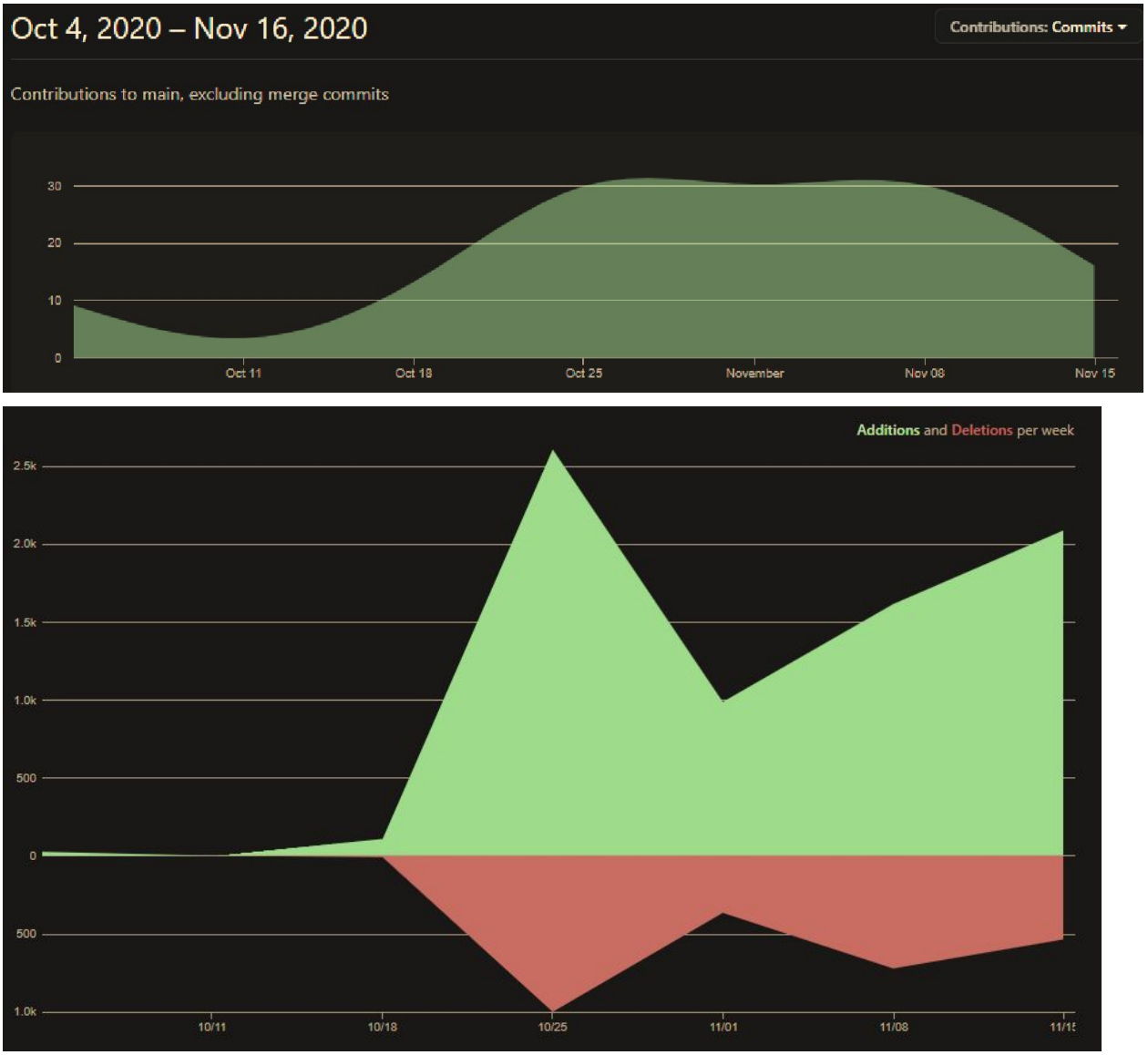
- (4a) Justification of engineering methods used
- (4b) Outlined approach to team organisation and set down why it was appropriate for the project and our team
- (4c) Presented the Gantt diagram(systematic plan) and came up with a Key Events List to aid it and show our critical path and workflow.

Implementation

- Events and planning are logged using github. We used development for grouping of specific features along with release notes and multi-branch issues which were raised through our Discord logs, Github issues & commit messages.

Appendix

Github Charts



Evidence of implementation progress

Communication

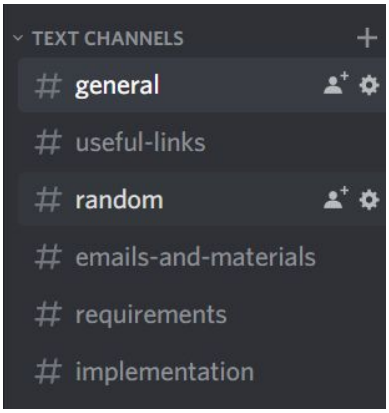


Figure 1 - Image of a sample of the Team Discord channels at the start of the project

Trello Board Screenshots

This is an example of our master trello board. We used kanban task board to organize tasks into phases.

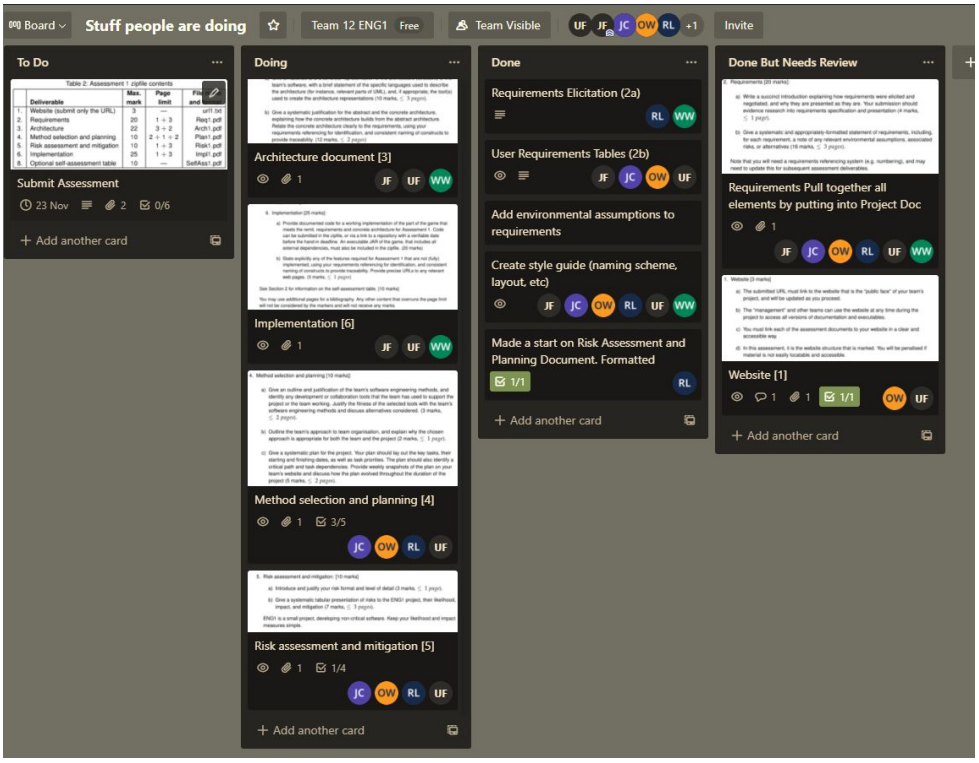


Figure 2: Trello board for documentation as of 09/11/2020

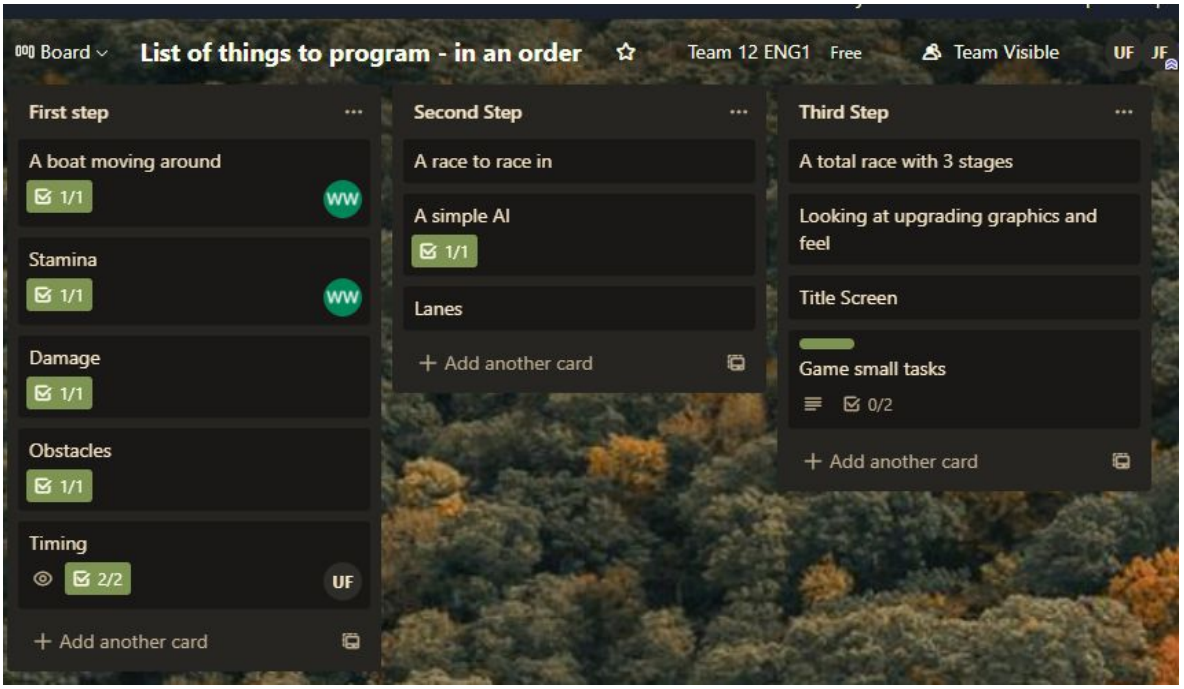


Figure 3: Programming Trello board as of 09/11/2020

Announcements

Discord announcements: this was a way to clarify what the focus of the week should be and meeting/schedules.

“@everyone sorry for the ping feel free to join” - Umer 14/10/2020

“@here group meeting if you're free” - Will 22/10/2020

“@everyone we're deciding tasks for the next few weeks” - Will 26/10/2020

“Hey guys sorry for the blip @everyone. Just checking for tomorrow if we are going to the Thursday eng1 at 13:30. We are going to look over progress and we will talk over what tasks we need to do over the next 5 weeks before submission. Will has made a start on the coding since Monday so we will also go over that.” - Umer 28/10/2020

“Olly has transferred the website to github pages and he'll put it on the github. @everyone You guys available to meet on monday 4pm?” - Umer 29/10/2020

Bibliography

[1] I. Sommerville, Software Engineering, Pearson Education, 2008, pp. 74-98.
[2] K. Schwaber and J. Sutherland, The Scrum Guide™, 2017 pp. 1-19