Method Selection and Planning

Group 11 - 11 Musketeers

Osama Azaz
Adam Dawtry
Tom Jackson
Brendan Liew
Holly Reed
Harry Ryan

4.(a)

Methodology we have chosen is an agile approach. [6]

- Agile is preferred when continuous delivery and feedback are important. [8]
- Agile development allowed the team to focus more on the planning aspects to ensure the project ran smoothly with a high level of communication amongst the group which integrated especially well with the scrum approach.
- As part of this agile technique we explored the Scrum technique which splits the tasks of the project into sprints which can then be followed up by a meeting to discuss the progress in the sprint and any concerns or problems found during the time period.
- The sprint size we used was a 2 week time frame, which we would begin and end with a meeting discussing the sprint that had passed and then the sprint we would begin.
- This approach method was chosen due to the short time frame involved and the smaller team size of 6 people meaning that work would have to be done in short bursts in order to hit the project deadline.

For **Communication** we have chosen Discord [1] because:

- It is a resource that each team member has experience with already.
- It has multiple ways to communicate so all can be kept on the same platform. For example; instant messaging for when we are doing work outside of meetings. It is also easy to go into calls if we want to do a quick meeting if we decide it is needed without having to send a link like with Zoom for example.
- The multiple chat rooms allowed meetings to be scheduled and all team members to be notified when they were scheduled and links to important references or information could be stored in another. All of our online team meetings were held in discord.
- Alternatives discussed for communication were zoom meetings and a whatsapp group chat however discord was chosen for its integration of both chat rooms and voice channels where screen share was available.

For **storing our documentation**, we have decided to use Google Drive and Google Docs [4]

- This is because of the ability to work on the same document concurrently. This is important as we will all be working one each section one after another so the whole team could be writing in the same piece of documentation.
- ■ Furthermore, the fact that Google Drive stores all files on the cloud and has a version control system mitigates the risk of losing files due to corruption or otherwise in Risk 12.
- It was useful when it came to the various meetings as it meant all team members could access and share feedback on the same files and any edits could be made and viewed by everyone working on that particular task.

Trello will be used as a **group organisation tool** and to set tasks for each member. [9]

- We have set it up so that we can decide what we want to do during that meeting and then at the end of each meeting, we assign cards to each member of the team and set a deadline which is usually the next meeting.
- With the columns on the board being:

    - ‘To do’ - where we add cards at the beginning of the meetings of tasks we plan on doing during the meeting and tasks left at the end of our meeting in this section are tasks for team members to do before the next meeting.
    - ‘Doing’ - where we list tasks that need to be done before its due date.
    - ‘Done’ - where all completed cards go.
- Different boards were set up for different aspects of the assessment, including: General Plan, Art Work, Change Report, and Implementation.
- For each task in the Implementation board, tags were assigned depending on if the feature was an assessment 1 or assessment 2 requirement, in which sprint the task was assigned, and whether the change was a bugfix.

For our **version control system,** we have chosen GitHub as it allows us to collaborate with one another efficiently and as it is the industry standard, we decided that using GitHub would enable us to gain relevant experience from this project which we can then use in the future for our personal projects or when we work within the industry. Its integration across the IDE and the website meant updating our website with the latest snapshots of the development was done efficiently.

For our **development environment** we chose Eclipse [5] as it works especially well with our collaborative tool GitHub by allowing integration with our GitHub and allowing the pushing and pulling of the project to and from the repository. With Eclipse integrating with GitHub this meant the rest of the team were able to feedback/integrate the code within their own tasks without the need of gathering the code from the people working on the task, therefore speeding up the process in the sprints as constant communication at all times of day were not required. When it came to implementing testing, we found that IntelliJ [11] worked much better than Eclipse, so we switched environments for testing purposes towards the end of the project.

4.(b)

Approach to team organisation

- **Role Allocation**
  - Team organisation was initially approached by identifying the strengths and weaknesses of each team member as brought up by Ian Sommerville [10], noting areas to exploit their strengths and avoid any weak areas.
  - By exploring these strengths team members could then be allocated to certain tasks where they had significant experience in the role and an eagerness to complete that section.
  - Each task was made to have a minimum of two people allocated to it. One would be the lead on that section whilst the other was to be there as a backup or “shadow” team member and enable the lead to offload some work load should it be necessary.
- **Regular Scheduled Meetings & Sprints**
  - Meetings were scheduled on a weekly basis with an extra meeting held on a bi-weekly to end and initiate a new sprint process.
  - Sprints were set into stages of two weeks in which we could allocate work to be done from the previous sprint should it not be completed and add to the work already completed.

How this approach is helpful to the team and project

- By establishing roles amongst team members a group structure was created allowing a heightened focus on each task and making the project less daunting by distributing the workload fairly and equally.
- Furthermore the roles utilised peoples prior experience in other subjects and hobbies meaning the most effective people for the job were chosen and thus each task would be done to the best standard possible in the short time frame available.
- Having a “shadow” team member for each task ensured that the ‘Bus Factor’ was never one for any task which was especially important given the current circumstances of the coronavirus pandemic.
- Having this extra person per task also ensured that knowledge could be shared out so that should a member of the group be unable to carry out their task for some reason the task can be picked up by another team member and continued with minimal disruption as once again recommended in Software Engineering 9th edition [7].
- The regular meetings were also an essential part of the organisation process as they provided structure to the teams time management applying pressure on each task to have progressed since the last meeting.
- Meetings allowed team members to express any concerns and update the team on their progress so far outlining their estimates of completion time and if they would meet their deadline in the sprint.
- Meetings also allowed feedback and constructive criticism where team members on other tasks could express their opinions and views on how aspects of our project can be improved or add ideas on components to add / edit.
- Each sprint was helpful to the project progression especially in the implementation area as it allowed for us to build upon the already existing game by adding new features to reach the requirements and enhancing old features as suggested by feedback amongst team members to improve on the quality of the project.

4.(c)

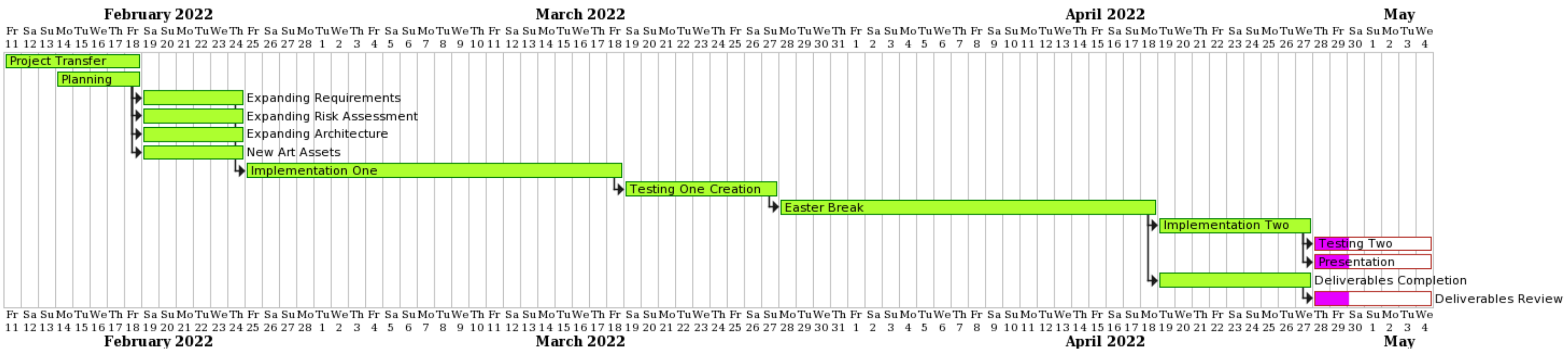
The systematic plan

- Project Transfer
    - Start Date - 11/2/2022
    - End Date - 18/2/2022
    - Dependencies - None.
    - Priorities - Come to a unanimous decision on which project we should take over by the end of this period at the latest so we can begin work on the new project as soon as possible.
- Planning
    - Start Date - 14/2/2022
    - End Date - 24/2/2022
    - Dependencies - New project to take over has been picked.
    - Priorities - Main focus on expanding the requirements, risk assessment, and architecture, as well as determining which art assets we need to create. It is a priority that these tasks are evenly distributed throughout the group as they are key aspects of the project that need to be started before a new implementation begins.
- Implementation One
    - Start Date - 25/2/2022
    - End Date - 18/3/2022
    - Dependencies - The requirements aspect of the planning stage has been completed. Requirements have been expanded to include the new features required.
    - Priorities - By the end of this period, the game should have the majority of the requirements implemented. Specific focus on any requirements missed by the previous team for Assessment 1. Further focus on documenting which changes have been made. Record any bugs found for fixing later.
- Testing Implementation One
    - Start Date - 19/3/2022
    - End Date - 27/3/2022
    - Dependencies - Implementation one should be started. Doesn’t matter if it is not completed as long as there is something to test. Some research has been done into the Software Testing Report and Continuous Integration.
    - Priorities - Priority on familiarising the testing team with the methods of testing and the structure of the tests.
- Implementation Two
    - Start Date - 19/4/2022
    - End Date - 27/4/2022
    - Dependencies - Implementation One and its testing has been completed.
    - Priorities - Identify any problems with gameplay or difficulty and adjust the code as necessary, make note of any requirements that have not been met yet. Focus on fixing any remaining gameplay bugs found earlier.
- Testing Implementation Two
    - Start Date - 28/4/2022
    - End Date - 4/5/2022
    - Dependencies - Implementation two has been completed.
    - Priorities - Aim to increase the code coverage and ensure that our continuous integration methods are fully implemented and functioning.
- Deliverable Review
    - Start Date - 28/4/2022

    - End Date - 4/5/2022
    - Dependencies - Deliverables should be at 80% completion.
    - Priorities - Focus on the Software Testing Report and Implementation as the other deliverables should mostly be done by this point.
- Presentation
    - Start Date - 28/4/2022
    - End Date - 4/5/2022
    - Dependencies - The project should be completed.
    - Priorities - Ensure the presentation is around 5 minutes long.

Evolution of the plan

- We found that we underestimated how long our implementations would take. For example, in Gantt Chart 5, Implementation One was extended. In Gantt Chart 6, Implementation One was further. We did this because we wanted to ensure that our first implementation was more fleshed out before we began testing it.
- Originally, we intended to go into Easter Break whilst on track with our original plan to ensure that we did not have to complete any work over the break. However, this changed due to the extension of Implementation One which meant that testing had to be pushed forward. Due to how large of a section testing is, we decided to move it into the Easter Break rather than leaving it for after the break to ensure that we did not fall too far behind.
- The final Gantt Chart is below:



Bibliography

[1] "Discord | Your Place to Talk and Hang Out," *Discord*. https://discord.com/. (accessed Feb. 01, 2022).

[2] "Why use Google Docs vs. Microsoft Word," *PaperStreet*, Aug. 28, 2019. https://www.paperstreet.com/blog/why-use-google-docs-vs-microsoft-word/. (accessed Feb. 01, 2022).

[3] GitHub, "GitHub," *GitHub*, 2018. https://github.com/.

[4] "Google Docs," *Google.com*, 2019. https://docs.google.com.

[5] Eclipse Foundation, "Eclipse," *Eclipse*. https://www.eclipse.org/.

[6] W. W. Royce, 'Managing the development of large software systems: concepts and techniques', *Proc. IEEE WESTCON, Los Angeles*, pp. 1--9, 1970[Online]. Available http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf.

[7] Braude, E. J., & Bernstein, M. E. (2016). Software engineering: modern approaches. Wave-land Press

[8] B.-A. Andrei, "A STUDY ON USING WATERFALL AND AGILE METHODS IN SOFTWARE PROJECT MANAGEMENT," 2019. [Online]. Available: http://www.rebe.rau.ro/RePEc/rau/jisomg/SU19/JISOM-SU19-A12.pdf.

[9] "Trello," @*trello*, 2014. https://trello.com/en-GB.

[10] I.S. Sommerville. *Software Engineering*. 9th ed. Pearson Education Limited 2011

[11] JetBrains, "IntelliJ IDEA," JetBrains, 2019. https://www.jetbrains.com/idea/.