Method Selection and Planning

Team 7: Bermuda Digital Entertainment

Prajwal Binnamangala
Joseph Sisson
Sebastian Sobczyk
Aymeric Goransson
David Ademola
CJ Donoghue

4.(a)

Methodology

- Originally we chose to use a SCRUM agile approach because:
    - We were familiar with this method from Assessment 1.
    - We have clear objectives for the project already set out and can further define requirements with the client.
    - We needed to prioritise creating working code as soon as possible over wasting too much time pre-planning.
- Later on in the project our method in practice became more similar to waterfall. [9]
    - We found that we were generally working towards broader goals rather than splitting things into small tasks, since the team was split into groups working on individual sections of the project.
    - We had a clear idea of the aim of the project and the documentation needed from the beginning and worked towards meeting these requirements.
    - Waterfall's methodology is a better solution for small projects that have well-defined requirements that will not change, while Agile is preferred when continuous delivery and feedback are important, requirements are not well defined and time to market is more important than releasing a full feature version. [11]

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.
- Since all members have access to all the files on GitHub and Google Drive, screen sharing was not necessary and instead we could simply instruct everyone to look at the relevant document.

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. This is also useful for the shadowing structure we discuss later, where one person monitors the work of another for mitigating risk. [6]
- 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 [6]. The version control also helps with change management.
- An alternative to this would be OneDrive or Dropbox, however Google Drive is the system that each team member already uses.

GitHub Issues will be used as a **group organisation tool** and to set tasks for each member. [13]

- We chose this instead of Trello [12] because this meant the tasks and code were in one central, easy to find location, because the functionality of Trello and GitHub

Issues was virtually identical for our use case and because this meant we did not have to make accounts for and learn how to use a new service.

- During each meeting, we decide on the tasks that should be done over the next sprint and who will be working on them. We create the task, write a brief description, assign it labels that describe which part of the project it is relevant to, such as ‘documentation’, ‘testing’ or ‘website’, and allocate it to the milestone for the relevant sprint. This is similar to the purpose of Kanban boards which are usually used for agile development, we used it for organising what team members should do in their own time between meetings.

During our weekly stand-up meeting, we reviewed the previous sprint and planned out the next one:

- The discussion leader goes over each open task, and the team member responsible for each task shows their work.
- The team discusses whether the task is complete, whether any possible changes may be required, and what next steps lead on from this.
- The Gantt chart and GitHub Issues board are updated to reflect what tasks were completed and what new ones should be added in for the next sprint.
- A record of what was discussed in the meeting is added to the Meeting Records document, a copy of which is on our website [17]

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.

For our **development environment** each of the members working on the codebase chose the tools they were most familiar with to facilitate efficient development, as there would be significant overhead due to the learning curve of getting everyone to use the same system, the costs outweighed the benefits. These included:

- Atom [15], an IDE which is similar to a text editor
- VS Code [16], an IDE which we were already familiar with from prior experience and use in Assessment 1 and was simple to set up for Java
- ReText [14], a lightweight MarkDown editor which was more efficient to use when making small changes to the website and building it locally

4.(b)

In terms of **team organisation**, the team's approach is:

- To make each member of the team responsible for one area of the project and relevant tasks, and split the team into two subgroups, one for implementation and the other for documentation.
    - We decided to delegate the implementation to a small group (Joseph and Sebastian) to prevent difficulties due to varying skill levels and understanding, and to avoid merge conflicts from too many features trying to be added at the same time.
    - However, the rest of the team was kept up to date on the state of the code during the weekly meetings, and the structure of the code was agreed with everyone.
    - We ensured everyone had equal input and understanding of how the implementation worked, so we could effectively write the documentation.
- To make sure all meetings are productive and efficient, one team member acts as the group leader who goes over all the open tasks and makes sure that everyone reports back in a way that the whole team is aware of what is happening, then updates the Gantt chart and writes a brief summary of the meeting so that everyone can go there for a quick overview of the status of the project.
- During this process we ensured that each member of the team would be up-to-date with the tasks being performed by other team members, which also ensured continuity between implementation and documentation.
    - This will help to keep the project on track and for example, if any of the members are ill/drop-out, then at least one of the members in the team will be able to continue from where they left off, hence it wouldn't be much of an issue as compared to if no one had any idea of what that person was working on. This helps to mitigate risk 4 and 7. [6]

We **planned** our work for each sprint in GitHub Issues and a Gantt chart during a group meeting where every member can talk about what they have done, what they would like to do, and what might be helpful for others to complete.

4.(c)

We planned our work in sprints that initially lasted one week, but later were extended to two weeks for practical reasons - it was difficult to organise team meetings with all the members so it was becoming unclear what was being worked on, so the sprints started to fall out of sync. Planning over a longer period gave us more flexibility.

When discussing the long-term plan for the project we took into account the circumstances of each team member such as family responsibilities and health conditions, as well as academic circumstances such as term breaks, other assignments due and assessments. Because of this, there were time periods where not much was getting done, and other times during which a lot of tasks were clumped together, but this was anticipated and we had given ourselves enough notice to make time to complete everything.

We also would sometimes underestimate the time a task would take, or wrongly predict when a task should take place, for example we marked down the task to write the summary for the testing report very early, but later realised that it would make more sense to do it after the actual tests were written. In this sort of situation we would simply move the task onto the next milestone and work on it in the next sprint, or mark it with the blocked label and keep it on hold until it was ready to be worked on again.

Our tasks were laid out in GitHub Issues and assigned to one or more people depending on the nature and scope. We also marked down our progress in a Gantt chart. Finally, we decided to start keeping records of each meeting so we were clear of what had been decided and added to the plan.

Screenshots of the GitHub Issues board and Gantt chart at various points in the project can be seen on our website [7].
The meeting records document [17] is also relevant to our planning process as it details the event at which the plans were formed.

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]JetBrains, “IntelliJ IDEA,” *JetBrains*, 2019. https://www.jetbrains.com/idea/.

[6]“York Pirates! Risks,” *York Pirates!* https://bermuda-digital-entertainment.github.io/assessment2/deliverables (accessed May 03 2022).

[7]“York Pirates! Plan,” *York Pirates! https://bermuda-digital-entertainment.github.io/assessment2/plan* (accessed May 03, 2022).

[8]“York Pirates! Use Case,” *York Pirates!* https://bermuda-digital-entertainment.github.io/assessment2/usecase

[9]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.

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

[11] 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.

[12] “Trello,” *@trello*, 2014. https://trello.com/en-GB.

[13] “Bermuda Digital Entertainment | York Pirates 2 Issues Board” https://github.com/Bermuda-Digital-Entertainment/York-Pirates-2/issues

[14] Retext-Project. “Retext-Project/Retext: Retext: Simple but Powerful Editor for Markdown and Restructuredtext.” GitHub, https://github.com/retext-project/retext

[15] “A Hackable Text Editor for the 21st Century.” *Atom*, https://atom.io/

[16] Microsoft. “Visual Studio Code - Code Editing. Redefined.” *RSS*, Microsoft, 3 Nov. 2021, https://code.visualstudio.com/

[17] “York Pirates! Meeting Records” *York Pirates!* https://bermuda-digital-entertainment.github.io/documents/meetingrecords.pdf