Team 5 Section 2, Project Post-Mortem

PREPARED FOR

Professor: Stacey Scott

CIS*4250 – Software Design 5

PREPARED BY

Emily Kozatchiner (1149665)

Jennifer Lithgow (1134108)

Jeremy Critoph (1140798)

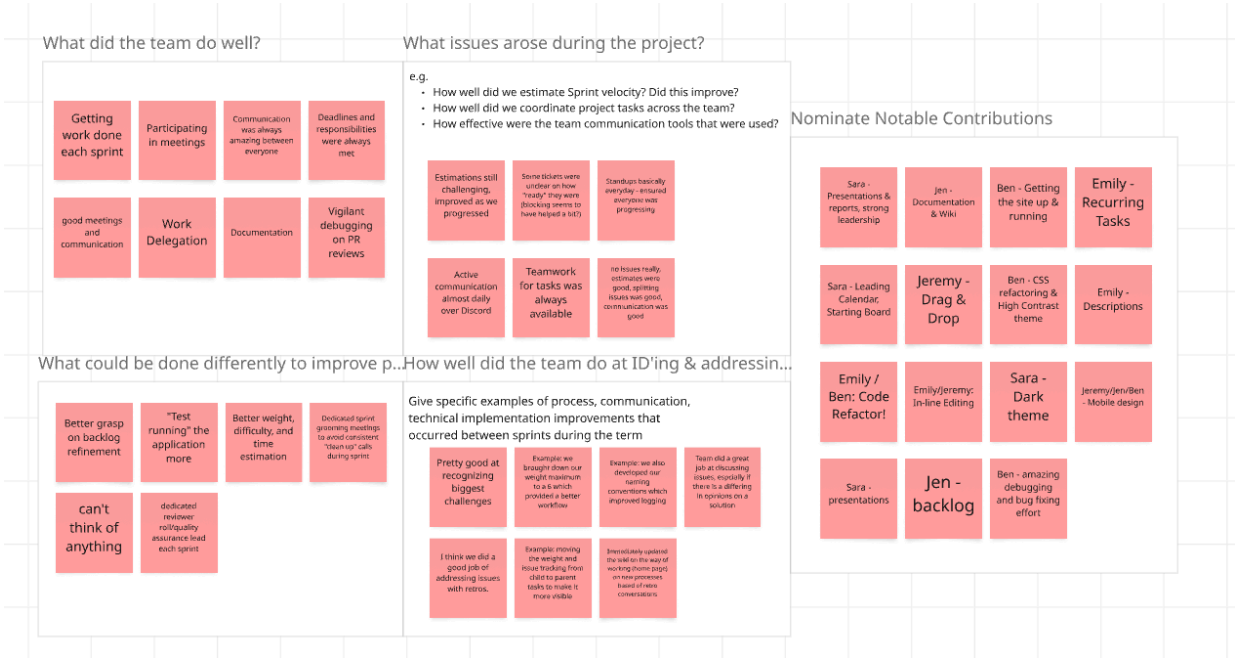
Ben Turner-Theijsmeijer (1152536)

Sara Adi (1129361)

Table of Contents

1.0 Documentation

Post-Mortem Miro Board Link: https://miro.com/app/board/uXjVIK3Aedw=/
Miro Board Screen Capture:



2.0 What Went Well

Throughout the four-sprint project, the team demonstrated strong collaboration and efficiency. Here are the key highlights:

Getting Work Done:

- Jen ensured the backlog and sprint boards were detailed and well-organized, enabling team members to easily pick up and work on tasks.
- All members demonstrated initiative by taking ownership of tickets, completing them efficiently, and keeping the team updated on progress.
- Timely and consistent review of Merge Requests (MRs) by all members significantly accelerated task completion.
- Work was distributed equally and we ensured that members were working on tickets interesting to them for increased effort and more efficient results.

Communication:

- A well-structured Discord server, with channels for MR links, new story ideas, and other topics, streamlined communication and task management.

- Weekly lab times were used effectively for team meetings, complemented by remote standups that ensured everyone stayed informed and engaged.
- Team members effectively communicated blockers and sought assistance when needed, often resolving issues through pair programming sessions over Discord.
- Daily standups provided regular updates and ensured steady progress.

Documentation:

- Comprehensive documentation of tests and user acceptance criteria, including linked tickets, testing videos, detailed descriptions, and screenshots, ensured clarity and accountability.
- Team members updated the wiki with expected goals and results for each sprint. Conventions were in writing for members to refer to.

Other Strengths:

- The team consistently met deadlines, demonstrating strong time management and focus.
- Vigilant merge request reviews facilitated swift feedback and task turnaround time, improving work quality.
- Tasks were delegated efficiently, ensuring an even distribution of workload.
- High participation in meetings fostered teamwork, productive discussions and effective decision-making.
- UI designs and back-end implementations involved high amounts of collaboration and feedback, yielding good end results.

3.0 What Issues Arose During the Project?

Despite the team's strong collaboration and commitment, some challenges emerged during the project:

Estimations:

- Sprint velocity estimation proved to be challenging, making it difficult to accurately predict the amount of work that could be completed within each sprint.

Ticket Readiness:

- It was occasionally unclear whether a task was fully fleshed out and ready to be worked on. This impacted task prioritization and created hurdles in work allocation and reporting.

Task Coordination:

- Coordinating project tasks across the team required continuous effort to maintain alignment and ensure everyone was working effectively on their respective responsibilities.

Communication Tools:

- Nothing to note here. Communication was very strong throughout the project lifespan. The communication tools were effective as mentioned in section 2.

4.0 What Could Be Done Differently to Improve Project Planning & Management?

For future projects, the team could implement the following improvements:

- Develop a stronger approach to backlog refinement, ensuring tickets are well-defined and ready for work.
- Conduct more frequent test runs of the application to identify issues earlier and refine functionality.
- Improve task weight estimations. While we considered estimated versus actual time from prior sprints, we still found it challenging to make accurate estimates.
- Schedule dedicated backlog grooming meetings to avoid ad-hoc "clean-up" discussions during active sprints.

5.0 Retrospective Effectiveness

General Performance:

- We did well at recognising our biggest challenges in each sprint, and made plans to resolve or minimize these challenges for the upcoming sprint. (included in our wiki)
- Despite disagreements, we would do our best to communicate our thoughts and reasoning with each other to pick the best option or come up with a compromise.

Specific Examples:

- After sprint 2, we noticed that we had multiple large tickets with an estimate of 8 or more. These ones took a while, but their time increased due to tedious amounts of merge conflicts that had to be resolved prior to merging them. We limited ourselves to a weight of 6 at most in the following sprint to try and avoid this outcome repeating itself.
- Immediately after our retros, we would update any relevant sections in our Gitlab wiki. For example, we wanted to change how/where we logged our times after our sprint 2 retro. We recorded the changes to our process outlined on the wiki’s home page, and implemented the changes in sprint 3 and onwards.
- After sprint 1, we realized developers all had different commit and branch naming conventions, leading to a disjointed workflow. Adding associated ticket numbers to commits and branch names in a specific convention were implemented once we noticed the divide during our retro.