Introduction **1**
What went wrong **1**
1– Transition of Front-End Tools 1
2 – React 2
3 – C# Experience 2
4 – 80% Code Coverage 3
5 – Pressing Deadline 3
What went right **4**
1– Good pace 4
2 – Continuous Monitoring 4
3 – Agile Methodology Implementation 5
4 - Jira 5
5 - Collaborative Problem Solving 6
Conclusion **6**

Condospective

Introduction

Firstly, coming into this sprint, our team knew that multiple elements needed an update. Our product vision statement, requirements and user stories needed minor changes to make them more specific and clearer to what we wanted for our application. Secondly, our team took a collective decision to change the environment we had set up for our application. Indeed, for our front-end, we decided to switch from the Uno Platform to using React.

What went wrong

1- Transition of Front-End Tools

One of the more challenging parts of this sprint was the change of tools done for our front-end part. Our team encountered multiple issues and limitations with the Uno Platform, initially chosen, such as compatibility issues, performance concerns, or lack of community support. Therefore, our team decided to choose a different route and utilize React instead for the continuation of the project.

However, the shift in tools necessitated the rewriting of code, and the adjustment of development processes, resulting in a temporary slowdown of project progress. To address this challenge, our team recalibrated project milestones, revised timelines and allocated additional resources to facilitate the transition to React.

This transition could have been avoided by choosing the right tool for all team members at the beginning of the project. To do so, our team could have conducted a more thorough evaluation of the potential tools' considering factors such as platform compatibility, learning curves, long-term maintenance requirements, integration capabilities, etc.

2- React

An aspect that went wrong for some, but not all team members, was the utilization of React for our front-end. Since React introduces new concepts such as JSX syntax, virtual DOM, and component-based architecture, it was challenging for some team members that haven't had hands-on experience with it.

The complexity of React concepts led to confusion, frustration, and slower adoption among team members, affecting productivity and overall motivation. To address the issue, the team provided comprehensive documentation and tutorials to familiarize team members with React concepts gradually.

Implementing a structured pair programming where experienced React team members are paired with those new to React could have facilitated this issue for the whole team.

3- C# experience

A challenge that the team continued to deal with during this sprint is the lack of experience of some team members with C#, used for this project. This occurred due to factors such as the limited exposure to C# in previous projects and the limited time assigned to achieving the deliverables.

This aspect has led to slower adoption and learning curves throughout sprint #2. To fix this issue, the team continues to leverage online resources and documentation to provide additional support. Furthermore, the team has assigned tasks based on team members' strengths and areas for growth, allowing for a gradual hands-on experience.

Improvements could have been made by conducting a skills assessment at the outset of the project to identify the gaps within the team. Based on that assessment, the choice of language used for this project could have differed.

4- 80% code coverage

The most challenging aspect of this sprint was the implementation of necessary tests to achieve of 80% code coverage. The failure of doing so may have been caused due to our team facing time constraints that led to deprioritizing testing efforts.

Without sufficient test coverage, the project is susceptible to undetected bugs and regressions, potentially leading to degraded product quality and increased maintenance efforts. To address this issue, our team has conducted a retrospective analysis to identify the root causes of the testing gaps. Furthermore, we've reevaluated project priorities and allocated dedicated time and resources for test implementation.

An approach that may have been implemented from the start is a test-driven development (TDD). This approach could have ingrained testing as a fundamental aspect of the development process, ensuring comprehensive coverage from the outset.

5- Pressing Deadline

An aspect that our team struggled with during this second print, but not the first, was completing the requirements for this sprint before the deadline initially imposed (before the extension was given). This was caused by high-pressure of trying to finish everything on time while dealing with other academic work and personal life.

Team members felt compelled to rush certain tasks or skip thorough testing and validation processes to meet the deadlines. To address this issue, the team prioritized tasks based on criticality and potential impact.

Setting realistic and achievable deadlines based on thorough project planning and estimation could have prevented excessive pressure on team members and allowed for a more balanced allocation of resources and efforts.

What went right

1- Good pace

One aspect that continued to work well during the development was the pace at which our team worked during the sprint. This good pace was reached by the effective sprint planning and task breakdown our team maintained from the first print. This time around, the prioritizing and the assignment of tasks was done more quickly.

The good pace in executing tasks resulted in increased productivity, allowing our team to deliver more value within the sprint timeframe. This positively impacts overall project progress and contributes to meeting project milestones and deadlines. To address this aspect, our team prioritized tasks based on their importance and urgency, focusing on high-priority items first to ensure maximum value delivery within the sprint timeframe.

While the good pace in executing tasks is commendable, there is always room for improvement. Indeed, when it comes to the implementation of code, identifying and mitigating risks concerning the implementation could've avoided the large amount of time that was spent on resolving coding issues on the back-end part.

2- Continuous Monitoring

An aspect that continued to work well during the development is the continuous monitoring and adjustment. Regular meetings and updates on task progress helped keep everyone aligned and focused on their respective responsibilities once again. This included reallocating resources, re-prioritizing tasks, or providing additional support to team members as needed.

By doing so, our team was able to adapt quickly to evolving requirements, priorities, and challenges. It enhanced our ability to stay aligned with project goals and respond to the bigger changes that were made in the project environment. To address this aspect, daily stand-up meetings were conducted. These meetings have provided an opportunity for team members to share updates, discuss progress, and raise any issues or concerns.

However, the integration of monitoring tools with automated incident response systems could have helped us more. By automating the response to common alerts and predefined scenarios, our team could have reduced manual intervention, minimized downtime, and improved overall system reliability.

3- Agile Methodology Implementation

The implementation of Agile methodologies remained something that worked well during the development. This aspect occurred due to clear understanding and commitment from the team. Agile principles such as iterative development, frequent feedback, and adaptive planning were embraced and integrated into the team's workflow.

The impact of effective Agile methodology implementation included increased productivity within the team, as well as improved collaboration among all team members. To ensure this implementation, the team established daily stand-ups and sprint planning meetings. Furthermore, regular retrospectives were held to keep up with the team's processes and identify areas that needed more work.

However, this implementation could have been better by putting more effort in strengthening communication channels between the different parts of the sprints. For example, if some members finished earlier than anticipated, and other members needed help with their parts, teammates should be able to move from part to part to help each other. This aspect is something our team still must work on.

4- Jira

The use of the Jira Platform was improved during this sprint and was a result of something working well for our team. The effort in using a user-friendly platform like Jira originated from our team's need to efficiently organize tasks, assign them to each other, and track our progress throughout the development cycle.

This team effort significantly improved team coordination and helped in meeting project deadlines effectively. To ensure every member of the team put in this effort, regular stand-up meetings were held where team members discussed their task's status, identified any blockers, and collaborated on solutions.

While the task management aspect was efficient, providing more detailed task descriptions and acceptance criteria upfront could have improved clarity and reduced the need for frequent clarifications.

5- Collaborative Problem Solving

An aspect that did work well and was maintained during the development of sprint #2 is our team's collaborative problem solving. This was reached, because each member of the team brought unique perspectives and problem-solving approaches to the table.

This aspect increased creativity, faster problem resolution, and a stronger sense of ownership and accountability among team members. To address collaborative problem solving, the team organized brainstorming sessions and workshops to generate ideas collaboratively.

However, for future sprints, our team could organize more opportunities to share knowledge and expertise such as in-person meetings to further enhance this aspect.

Conclusion

In conclusion, the team has encountered both achievements and challenges in equal measure during this second sprint. One of the main takeaways from this sprint is the importance to remain consistent in communicating and collaborating with one another. Despite facing obstacles such as technical complexities in the back end and time constraints, the team's ability to work together, share knowledge, and support one another has been instrumental in overcoming obstacles. Additionally, the team has gained a deeper understanding of the technologies that were retained from sprint #1. However, when it comes to the new technologies that were installed, there is room for improvement and growth for all team members as this project continues. Finally, this sprint has highlighted the significance of adaptability when encountering hurdles. As the team navigated through unfamiliar territories implemented in this sprint, all team members demonstrated their ability to adapt quickly and remain focused on the project's requirements and deadlines to respect.