**Final Project! Sprint Review and Retrospective**

**Kyle Altemeyer**

**Date:10/17/2025**

**Applying Roles**

Throughout this course, Ive been able to look at how each Scrum agile role works together to make a project successful. The product owner keeps the project focused on customer needs, the developer builds and adjusts based on some feedback, and then the tester makes sure everything works as intended. From my perspective, the tester plays a key role in making sure the team is constantly improving through feedback. Even though we didnt run an actusl real software project, working through this Travel App and Vision Quest assignment has made it clear to me that every role needs to communicate often and stay aligned on the goals. When those roles work together properly then agile runs smoother and faster (Hoda et al., 2021).

**Completing User Stories**

The scrum agile approach made a lot more sense once I saw how user stories are broken down into the smaller, doable tasks. Its easier to estimate time and effort that way, also it helps keep motivation up as the team completes things gradually. Ive noticed this would be useful in the real world too, like my job in industrial or maintenance settings, because breaking big projects into smaller pieces makes it easier to stay on track and manage as a whole. Agile focuses on progress you can actually see, not just long term goals that take months to show. That visibility builds confidence within the team (Campanelli et al., 2023).

**Handling Interruptions**

In an agile setup, interruptions are part of the game as with project. The big difference is that instead of shutting down progress when something changes, agile has observed the nature of a process and encourages adjusting and finding a new route forward. I can see this being super useful in engineering or production work where unexpected problems pop up constantly. During my course work, the Vision Quest case study showed how switching from waterfall to scrum made it easier for teams to regroup when priorities changed. That adaptability is what is needed and it keeps momentum going and helps teams deliver results instead of getting stuck.

**Communication**

The biggest thing Ive taken away from all of this by far is that communication is everything. Agile forces you to talk, check in, and keep your teammates updated. Daily standups, retrospectives, and sprint reviews all keep the team accountable. Indeed we only simulated those practices more than lived them, but its clear how much they matter. If communication breaks down, the whole agile process falls to pieces. However when its done right, everyone knows whats happening, and whos doing what, and where help is needed. That level of openness is what separates agile from the gray head old,school waterfall way.

**Organizational Tools**

If I had to choose a tool to use going forward, Id have to go with something like Jira or AzureBoards. From what Ive learned in reading, they keep everything in one place and organized like the tasks, bugs, comments, and updates. They make progress visual, which helps the whole team stay on the same page. These tools ought to make tracking user stories and sprints easier, and they should help reduce confusion on who is working on what. Its basically like a digital version of the agile board, but more detailed and accessible for remote teams (Hroník et al., 2023).

**Evaluating Agile Process**

After working through the assignments and seeing how each part fits together, Id say the scrum agile approach is hands down more effective than waterfall for projects that are going to change often. Waterfall is just too rigid, it expects everything to go right the first time and it never will. Agile plans for things to go wrong AND gives you a system to handle it. That mindset shift is what makes it so useful for modern software and even industrial engineering work. You dont get paralyzed by change; you simply roll with it. I think the Travel project animations really drove that home with short cycles and open communication to beat long, drawn out planning every time.

**References**

Hoda, R., Salleh, N., Grundy, J., & Lowry, P. B. (2021). *The rise and evolution of Agile software development*. Journal of Systems and Software, 173, 110871. <https://doi.org/10.1016/j.jss.2020.110871>

Campanelli, A. S., Parreiras, F. S., & Rocha, A. R. C. (2023). *Adapting Agile practices to hybrid and industrial contexts: A practical view on modern software processes*. Information and Software Technology, 160, 107255. <https://doi.org/10.1016/j.infsof.2023.107255>

Hroník, F., Kuchta, D., & Blahušiaková, M. (2023). *Agile project management tools and their impact on team performance in software development*. Procedia Computer Science, 219, 375–384. <https://doi.org/10.1016/j.procs.2023.01.091>