**Sprint Review and Retrospective**

*Adam Gomez*  
*SNHU CS 250 – Software Development Lifecycle*  
*June 2025*

**Introduction**

Throughout the SNHU Travel project, I had the opportunity to work through the software development life cycle using the Scrum-Agile framework. Each week, I stepped into a different role—from developer to tester to Scrum Master—and gained a better understanding of how a real Agile team operates. This reflection covers what I experienced while building out the project, how my team adapted to unexpected changes, and what I learned about communication, tools, and Agile development overall.

**Roles I Took On and How They Helped the Project**

One of the most valuable parts of this project was getting to take on different Scrum roles. When I was in the **developer** role, I focused on writing the code to implement features based on user stories, like the trip search and booking process. That helped me think more about how user needs translate into technical functionality.

As a **tester**, I verified whether the features actually worked as expected. I specifically remember testing the booking confirmation and making sure the currency converter gave the correct results. That helped me catch small errors that could have impacted the user experience.

Later, I served as the **Scrum Master**, which gave me a broader view of the project. I led standups, helped the team stay on track, and made sure we kept to our sprint schedule—especially when things didn’t go as planned. That role gave me a new appreciation for how important communication and flexibility are to a successful Agile project.

**Finishing User Stories With an Agile Approach**

Using Agile made a big difference in how we completed user stories. At the start of each sprint, we held planning sessions to estimate how much work we could handle. That structure helped us break bigger stories down into smaller, more manageable pieces.

By Sprint 2, we had already delivered working versions of the trip search and booking features. Because we worked in short sprints, we could make changes and improvements based on feedback without having to rebuild everything. This approach really helped us focus on delivering value with each sprint instead of trying to do everything all at once.

**Handling Interruptions and Changing Directions**

The biggest challenge came during Sprint 3, when we had to pivot. Partway through the sprint, the stakeholders asked us to add a **“last-minute travel deals”** feature. That meant adjusting our backlog, shifting priorities, and figuring out how to fit in a brand-new request without throwing off everything we’d planned.

In my role as Scrum Master at the time, I helped lead a backlog refinement session. We identified which tasks could be moved to a future sprint to make room for the new feature. It wasn’t easy, but because of the way Scrum works, we didn’t have to start over or delay everything. We just adapted and kept going. That experience showed me how Agile helps teams stay flexible, even when things change unexpectedly.

**How We Communicated as a Team**

Strong communication made a huge difference on this project. We used **Slack** for fast updates and **Trello** to track our tasks. I also led daily standups during the sprint where we handled the pivot, which helped everyone stay focused and aware of what was changing.

One thing that worked really well was setting up a dedicated Slack thread just for the last-minute deals feature. That way, we didn’t clutter up the main channel, and everyone knew where to go for updates related to that specific change. Those small details made collaboration easier and kept us from losing momentum.

**Tools and Scrum Practices That Worked for Us**

We leaned heavily on a few key tools and Scrum events to stay organized. Trello helped us visualize and manage user stories, Slack kept communication moving, and we used retrospectives to figure out how to improve each sprint.

One improvement that came out of a retrospective was the decision to cut down on how much time we were spending in sprint planning. We realized we were getting bogged down in too many technical details, so we decided to save those deeper discussions for smaller breakout conversations. That small change helped make our planning sessions more efficient and focused.

**Looking Back at Agile’s Impact on This Project**

**What worked:**

* We made steady progress with each sprint.
* We were able to handle a major change (the last-minute deals request) without missing a beat.
* Regular standups and retrospectives helped us catch problems early.
* Everyone had a clear role and knew what they were responsible for.

**What was challenging:**

* Estimating how long tasks would take wasn’t always easy.
* Staying engaged in every role took extra effort, especially when priorities shifted.
* Making changes mid-sprint required strong coordination.

Overall, Agile was a great fit for this project. If we’d been using a Waterfall approach, that change in Sprint 3 would have been a huge problem. We would’ve had to go back and revise all our requirements and designs, and that would have delayed everything. With Agile, we stayed flexible and focused on delivering something useful every step of the way.

**Conclusion**

Working through this project using the Scrum-Agile approach taught me a lot about how software teams collaborate in the real world. By stepping into different roles, learning how to prioritize work, adapting to unexpected changes, and communicating effectively, I got a real sense of how Agile makes teams more resilient and productive. This experience showed me how valuable Scrum practices are—not just for managing work, but for building better software with better teamwork.