Final Project CS-250

The SNHU Travel project was a great learning experience for understanding how Scrum-Agile works in a real-world setting. Even though I wasn’t actually part of a team, I went through projects that helped me see how Agile teams function and how different Scrum events and roles contribute to a project’s success. This retrospective looks back at what I learned about Agile roles, completing user stories, handling interruptions, communication, and the tools that make Agile effective. I’ll also discuss the pros and cons of using Agile and whether it was the best fit for this kind of project.

Even though I didn’t work with an actual team, studying Agile projects gave me insight into how different roles contribute to success. The Scrum Master is responsible for keeping things organized, running meetings, and helping remove obstacles. The Product Owner gathers information from stakeholders and prioritizes what features should be developed. The Development Team is responsible for actually building the product, testing it, and making adjustments based on feedback. When looking at how travel recommendations were built in an Agile project, I saw that the Product Owner used customer feedback to guide what features should be included. The developers then worked on those features in small, manageable chunks while testers made sure everything functioned correctly. Even without actually being in a team, it was clear how important each role is in making a project successful.

One of the biggest takeaways from reviewing Agile projects was seeing how user stories help guide development. Instead of just jumping into coding, user stories keep the focus on what the customer actually needs. For example, a story like, *“As a user, I want to see travel recommendations based on my past trips so I can find vacations I’d enjoy,”* helps define what the feature should do. Agile projects break down these stories into smaller tasks, like setting up a database for user travel history, writing the recommendation algorithm, and designing the user interface. By reviewing case studies of Agile teams, I saw how working in short sprints helped developers stay on track and make improvements based on early feedback instead of waiting until the end.

One of the biggest strengths of Agile is how it handles changes and interruptions. Traditional project methods often plan everything from the start, which can make it hard to adapt when new ideas or problems come up. In Agile, interruptions are expected and handled through backlog refinement and sprint planning. One example I studied involved a travel booking system where stakeholders requested a new filter for eco-friendly travel options. Instead of derailing the entire project, the team discussed the change, adjusted their priorities, and fit it into future sprints. This kind of flexibility is what makes Agile so effective, especially in software development where customer needs often change.

Communication is a huge part of Agile, and studying different Agile projects showed me how teams stay in sync. Daily stand-ups help everyone share updates and ask for help if they’re stuck. Emails and instant messaging keep discussions going outside of meetings. Here’s an example of a message that would be sent in an Agile team:

Subject: Need Clarifications & Test Details for New Feature  
To: Product Owner & Tester

Hey [Product Owner Name] and [Tester Name],  
I’m reviewing the sign-up feature and had a few questions. Could you confirm which fields are required and how strict the validation should be? Also, [Tester Name], do you have test cases ready for this? Are there any specific browsers or devices we should focus on? Let me know when you get a chance. Thanks!  
Best,  
Blake Boyle

This type of message is direct and keeps communication clear, which is key for Agile teams to work efficiently. It was interesting to see how regular check-ins and updates help prevent confusion and keep the project moving forward.

Another thing I learned from reviewing Agile projects is how important organizational tools are. Jira and Trello help teams track tasks and see what’s in progress. Slack or similar messaging platforms help with quick updates. Agile events like Sprint Planning help teams set goals, while Daily Stand-ups keep everyone aligned. Sprint Reviews give teams a chance to show their progress, and Sprint Retrospectives help them reflect on what went well and what needs improvement. These tools and meetings work together to keep Agile projects running smoothly and make sure everyone knows what they should be working on.

One of the biggest advantages of Agile is its flexibility. Changes can be made without derailing the project, which is especially useful when working with evolving user needs. Another key benefit is continuous feedback, as teams get regular input from stakeholders and users, allowing them to make adjustments along the way. Agile also promotes strong team collaboration through regular check-ins, ensuring that everyone stays on the same page. Additionally, problems are addressed quickly before they become bigger issues, making it easier to maintain a steady workflow.

However, Agile also has some challenges. One downside is that it involves a lot of meetings, including stand-ups, reviews, and retrospectives, which can take up significant time. Another issue is the risk of scope creep, where constant changes and additions can lead to an ever-expanding workload. Additionally, Agile depends on quick feedback from stakeholders, and if responses are delayed, it can slow down the project’s progress.

Even though Agile has some challenges, I think it was the best choice for the SNHU Travel project. Because user’s needs in travel booking can change quickly, Agile’s flexibility made sure the project stayed aligned with what customers wanted. A traditional development approach would have made it harder to adjust along the way.

Looking back at what I learned from Agile projects, it’s clear that the Scrum-Agile approach is a strong way to manage software development. Even though I wasn’t in a real team, reviewing different case studies helped me understand how Agile works in action. The roles of the Scrum Master, Product Owner, and Development Team all play a big part in keeping things organized. Breaking work into user stories made it easier to focus on what really matters. The ability to adapt to changes kept the project relevant, and communication played a huge role in making sure everything ran smoothly. If I were working on a real Agile project in the future, I’d definitely use what I’ve learned to help keep things structured and efficient.