Throughout the SNHU Travel project, each role within the Scrum-Agile team played a critical part in ensuring the development process moved forward efficiently and effectively. Over the course of the semester, I took on various team roles to gain a full understanding of how each contributes to the final product.

As the Product Owner, I helped shape the project by identifying and prioritizing user stories that reflected the client’s needs. This included drafting three initial user stories focused on the “Top Destinations,” “Vacation Preferences,” and “Popular Vacation Spots” features. These stories guided the team’s efforts and acted as the foundation for sprint planning and testing.

As a Developer, I was responsible for implementing functionality based on the user stories and updating the application to meet evolving requirements. For instance, I revised the slide show to support individual destination cards, each with corresponding images and descriptions. I also incorporated feedback from the Product Owner, such as changing placeholder images and updating text formatting and layout for clarity and visual appeal.

Finally, as the Scrum Master, I guided the team through the development cycle, helping manage interruptions and reprioritizing work during the revised sprint when the wireframe changed. I also coordinated the creation of test cases and ensured they aligned with acceptance criteria. This helped us stay focused and adapt quickly to feedback without losing momentum.

Each role I filled gave me insight into how cross-functional teams collaborate in a real Agile environment. This role-switching also emphasized the importance of empathy and communication when working on a dynamic project.

Throughout the SNHU Travel project, I had the opportunity to take on various roles within the Scrum-Agile team structure, which gave me a clear understanding of how each role contributes to the success of a project. The roles of Product Owner, Scrum Master, and Developer all played a part in shaping the final product and ensuring we delivered value to the customer.

As the Product Owner during the early part of the project, my main responsibility was to define user stories that captured the needs and expectations of SNHU Travel. I focused on identifying key features such as displaying top travel destinations, customizing vacation preferences, and showcasing popular vacation spots. By prioritizing these stories in the Product Backlog and ensuring they were ready for development, I helped the team stay focused on delivering the most important features first.

Later, when I stepped into the Developer role, I used the user stories to guide my implementation of features in Eclipse. For example, when working on the Top 5 Destinations list, I designed and coded a GUI using Java Swing that displayed destination names, images, and descriptions. I also made sure to follow best practices like keeping the interface responsive and providing navigation through slides using Previous and Next buttons. The code I wrote reflected the details outlined in the user stories and acceptance criteria.

As the Scrum Master, I facilitated communication between roles and helped ensure that the project moved forward smoothly. This included organizing our test planning process and ensuring we adjusted our implementation based on feedback and changes in requirements. I guided the team through sprint milestones and helped resolve blockers, like when I had to troubleshoot image loading issues in Eclipse or re-import pictures into the resource folder.

Each role brought different strengths to the table and reinforced the importance of collaboration. Even though I was filling all of these positions myself, switching perspectives helped me better understand how these roles work together to produce working software that meets client needs.

Using a Scrum-Agile approach throughout the project helped ensure that user stories were completed in manageable increments and tied closely to the client’s needs. Instead of approaching the SNHU Travel project as one large task, breaking it down into user stories gave clear direction to each sprint and made tracking progress much easier.

Early on, I wrote user stories using the standard format: *As a [type of user], I want to [perform some task] so that I can [achieve some goal].* This helped me focus on the purpose behind each feature. One of the user stories, for example, involved allowing a user to view a list of the top five travel destinations. I used that story to design and build a Java-based slide show application that displayed each destination’s name, image, and short description. The story gave me a clear idea of what the feature should include and how it would bring value to the end user.

In a later user story about customizing vacation preferences, I had to think from the user’s perspective; what filters would a traveler want? What kind of details would help them choose a destination? Even though I didn’t build a full preferences system, the story helped me frame the UI around user decision-making, and it made me think ahead about how the system might expand in future sprints.

What really helped with completion was aligning test cases with the user stories. Each test directly reflected an acceptance criteria from one of the stories. For example, I created a test case to verify that clicking “Next” on the slide show would show the correct image and update the description. This kind of traceability ensured that the development stayed user-focused.

By the end of the project, all three user stories I wrote had been implemented and tested in some form. The iterative structure of Agile allowed me to revisit and improve the implementation of those stories without needing to scrap previous work. That flexibility, along with short feedback loops, made it easier to see stories through to completion.

One of the things I came to appreciate about working in an Agile environment was how naturally it handled interruptions or shifts in direction. Midway through the SNHU Travel project, the Product Owner announced a significant change in focus: the application needed to highlight wellness-themed travel rather than just general vacation destinations. This was a key moment where Agile principles, especially flexibility and responsiveness, really showed their value.

Instead of this change derailing the whole project or forcing a restart, the shift was treated like any other change to the Product Backlog. I reevaluated the existing work and adjusted the destination list, descriptions, and images to match the new wellness theme. Because each feature was built in short, iterative chunks, it wasn’t difficult to replace general travel images and descriptions with new wellness-based ones like Ubud, Blue Lagoon, Banff, Kyoto, and Tulum.

I also had to adapt how I approached the interface. The content of the slide show changed, but the core functionality didn’t need to be rebuilt. I simply updated the values in the appropriate methods and added clear comments to reflect the changes for the next sprint. In a waterfall model, a change like this would have meant going back to the beginning and redefining the entire scope. But because we were using Scrum, the change became a normal part of development instead of a crisis.

Additionally, there were small technical interruptions too, like image path issues, importing resources, or needing to shift development from one machine to another. Having experience using Eclipse, Git-style versioning, and modular code allowed me to adjust quickly and get things working again without throwing off the rest of the sprint.

Agile gave me the mindset and structure to treat interruptions as opportunities to improve, rather than obstacles. That mindset made it possible to keep moving forward even when plans changed. Effective communication was a big part of what kept the SNHU Travel project moving forward, even when I was juggling multiple roles. Since I wasn’t working with a real team, I treated my planning, notes, and documentation as the main way to communicate across roles. I approached every assignment as if I were handing it off to someone else, making sure my intentions were clear and the next steps were easy to follow.

One of the best examples of this was during the test planning phase. I created test cases that mapped directly to user stories and included clear pass and fail criteria. I also revised them when the project shifted to a slide show format instead of a basic list. In those revised test cases, I updated the user actions to reflect how someone would navigate through a slide-based interface and made sure each case was specific and easy to understand. If someone else were actually reading them, they would know exactly what was expected and how to test each feature.

Another example came when I ran into issues with image imports in Eclipse. I documented the process I followed to fix it and made sure the final file paths were correct and easy to update. This kind of communication helped me stay organized and would have helped any other developer picking up the project mid-sprint.

Throughout the project, I also made a habit of writing clear comments in my Java code. These comments explained what each section was doing and why it was necessary. For example, when I changed the slide content to wellness destinations, I added a short comment above each change to note that it reflected the new direction given by the Product Owner. Even though I was working solo, I treated those comments like they were for someone else reading my code for the first time. That mindset helped keep everything transparent and collaborative, even within a one-person Agile team.

Using Scrum-Agile principles alongside specific organizational tools helped keep the SNHU Travel project on track, even as goals evolved. Although I didn’t work with a live team, I still relied on Agile structure to guide my development process. The use of user stories, backlogs, sprints, and test cases gave me a framework to stay organized and focused.

The Product Backlog spreadsheet was one of the most helpful tools throughout the course. By sizing and prioritizing user stories early on, I could plan my work more effectively. I knew which features needed to be built first and which could be added or expanded on later. The backlog made it easier to shift priorities when the project moved from general travel destinations to a wellness-focused theme, and I didn’t have to rework everything from scratch.

Test case templates were also useful organizational tools. Each one connected directly to a user story, included clear steps, and defined what a pass or fail looked like. I revised them when the interface changed from a list view to a slide show, which made it easier to ensure the code still aligned with user needs. These templates helped simulate what a QA team would rely on to validate functionality.

Scrum events like the Sprint Review and Sprint Retrospective also served as checkpoints. Even though I wasn’t holding meetings, I used those moments to reflect on what was working, what wasn’t, and what needed to change in the next round. That mindset kept me focused on progress and improvement. If I had been working on a real team, these same tools and practices would have helped keep everyone aligned. Overall, combining Agile principles with structured templates and consistent code organization gave me a clear path to follow and made it easier to handle changes and additions throughout the project.