CS-250 Final Project – Sprint Review & Retrospective

Role: Scum Master

Date: 10/19/25

As Scrum Master for the SNHU Travel pilot, I led the team through a focused Scrum-based sprint cycle targeting a Minimum Viable Product (MVP) for a destination search and booking prototype. This retrospective summarizes role contributions, user-story completion, interruption handling, communications, organization tools used, and an evidence-based evaluation of Scrum as applied to this project.

Each Scrum role made measurable contributions to the project. The Product Owner prioritized backlog items by business value and negotiated tradeoffs that kept the Sprint focused on core booking and search capabilities. The Developers—including database specialists—designed a schema to support rapid iteration: normalized core tables (users, trips, bookings), created derived views for analytics, and established a migration strategy so feature branches could safely evolve the database without blocking feature development. As Scrum Master I removed impediments by securing dev/staging DB credentials, coordinating schema review sessions, and resolving deployment bottlenecks. These activities align with Scum accountabilities and helped maintain team focus and predictable delivery.

Adopting Scrum helped user stories reach Drone through short iterations and clear acceptance criteria. Example user story: “As a product analyst, I need search results to return within 300ms so customers get a responsive experience.” Acceptance criteria included query plan verification, an index strategy, and an automated performance test. Because stories were small, testable, and prioritized, Developers and the Product Owner could refine scope during

Sprint Planning and the Daily Scrum, allowing stories to be completed end-to-end (code, DB migration, tests, and docs) each Sprint.

During the pilot, a mid-sprint stakeholder request changed a non-critical UI flow to support mobile-first search. Scrum's inspect-and-adapt cadence let us re-negotiate scope with the Product Owner, split one large backlog item into two smaller stories (mobile UI + backend adaption), and preserve the Sprint Goal. The Sprint Goal acted as a north start that helped the team decide what to protect vs. what to re-scope. This approach kept value delivery steady despite change. The 2024 State of Agile report also notes that Agile teams report improved alignment with business needs during change—a benefit we observed.

I used three communication mechanisms that improved collaboration:

- Daily Standup (Slack message):

  "Standup – 2025 – 10 -

  - Yesterday: Completed DB migration v1 and integration tests.
  - Today: Add index tuning and performance test; pair with QA at 2pm.
  - Blocker: Needing staging DB credentials (Ticket #JIRA-123)."

  Short, action-oriented, and surfaced impediments.

- Sprint Review demo message to stakeholders:

  "Subject: SNHU Travel Sprint 3 Review – Demo & Outcomes

  Body: Short summary of what was completed, link to demo, acceptance criteria met, new backlog items and requests, ask for feedback within 3 business days."

  Provided a brief summary and link to the working prototype with release notes (highlighting acceptance criteria met). This built confidence and invited early feedback.

- Confluence backlog notes:

"Title: US-105 – Fast destination search for users

As a user, I want search results returned in <300ms so I can choose options quickly.

Acceptance Criteria: a) Query returns top 20 results <300ms; b) index added and documented; c) migration script included + rollback; d) automated performance test passes."

Documented decisions about the schema design and the Definition of Done so future developers aren't blocked. Transparent, written history helped the team move faster.

We used lightweight toolchain: Jira for backlog & sprint tracking, GitHub (feature branches + PRs) for code review, Flyway for database migrations, and GitHub Actions for CI pipelines. The Scrum events (Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective) were tightly coupled to these tools: Sprint Planning produced a committed Sprint Backlog in Jira; Daily Scrum references were updated automatically by GitHub issue links; Review demos were deployed by CI to a staging URL for stakeholders. The Scrum Guide emphasizes artifacts and commitments (Product Goal, Sprint Goal, Definition of Done) that we used to increase transparency and focus.

Pros: Faster stakeholder feedback, clearer prioritization, incremental delivery of a usable product, and better risk mitigation (changes early). Small, testable backlog items reduced hand-offs and increased ownership. These advantages are consistent with case studies showing Agile's benefits in dynamic requirement environments.

Cons: Scaling across multiple teams created coordination overhead; some regulatory or data-integrity steps required more formal gating (where Waterfall-like checkpoints still had

value). Larger organizations may face barriers to scaling Agile across legacy processes, a trend reflected in recent industry surveys.

Scrum-Agile is the best approach for SNHU travel. For this pilot and product goal (rapidly validating travel search and booking flows), Scrum provided the necessary feedback cadence and adaptability. For extremely regulated, hardware-dependent, or long lead-time initiatives, a hybrid approach (Agile within Waterfall constraints) could be more appropriate. Applying Scum from a DBA / DevOps lens requires explicit investment in migration automation, test coverage for DB changes, and clear acceptance criteria that include data integrity checks. The product benefited from shorter cycles, clearer priorities, and better stakeholder alignment—all results that support recommending further Agile adoption with a planned, supported scaling strategy.