Sprint 4 Retrospective
Sprint Number: Sprint 4
Sprint Duration: December 9 - December 14, 2025
Retrospective Date: December 14, 2025
Facilitator: Omar El Hajj Chehade (Scrum Master / PO)
Attendees: Full Development Team

Sprint 4 Overview
Sprint Goal
Finalize Weather Watcher for submission by stabilizing the application, completing documentation, and improving monitoring a

Sprint Outcomes
Completed User Stories:
- Final Documentation & Submission (3 SP) ■ Closed
- Application Insights & Monitoring (3 SP) ■ Closed
- Weather Comparison Tool (3 SP) ■ Closed
- Temperature Unit Toggle (2 SP) ■ Closed

Total Story Points:
- Committed: 11
- Completed: 11
- In Progress: 0
- Not Started: 0

Velocity: 11 story points (100% completion rate)

Key Metrics:
- Pipeline Success Rate: 100% (all GitHub Actions runs green by end of sprint)
- Deployment Frequency: On every merge to main
- Application Uptime: 99%+ (during testing)
- Observability: Application Insights dashboard and alerts configured

Retrospective Format: Start-Stop-Continue

What Went Well (Continue)
- All planned Sprint 4 stories were completed with 100% completion.
- CI/CD pipeline was stable; no blocking failures by the end.
- Application Insights was fully wired with health checks, custom telemetry, and dashboards.
- Documentation was finalized: README, architecture diagram, CI/CD documentation, and sprint reports.
- Team collaboration was strong, with good coordination on final changes and reviews.
- Hotfixes and last■minute adjustments were deployed safely through the pipeline.

What Didn’t Go Well (Stop)
- Too many last■minute documentation edits created stress near the deadline.
- Some work (like custom telemetry) was started late in the sprint, compressing testing time.
- Local environment issues and branch mismatches slowed some teammates temporarily.

What To Start Doing (Start)
- Lock the scope earlier for final sprints so last■minute requests don’t overload the team.
- Enforce a “freeze window” before submission where only critical fixes are allowed.
- Keep a single source of truth for documentation to avoid duplicate or outdated files.
- Use smaller, more frequent PRs for configuration changes (e.g., monitoring, alerts).

Technical Achievements
- Fully configured GitHub Actions CI/CD workflow (test, build, deploy to Azure Web App).
- Integrated Application Insights with:
  - Availability tests
  - Request and failure metrics
  - Custom telemetry for weather searches and autocomplete
  - Alerts for failed requests
- Created monitoring dashboard (workbooks) to visualize requests, failures, and performance.
- Completed final features: unit toggle, comparison tool, and final documentation polish.

Process Improvements
- Sprint planning for Sprint 4 was realistic and matched the remaining capacity.
- Work items were clearly owned (e.g., monitoring, docs, comparison tool), reducing overlap.
- Code reviews stayed focused and fast, which helped keep the pipeline green.