Final Retrospective

Project: Task Manager API
Date: February 19, 2026
Duration: 2 Sprints

Sprint 1 Improvements → Sprint 2 Results

Improvement Identified	How It Was Applied	Result
Add structured logging	Created logger utility, added to all controller methods	Full observability into API operations
Commit more frequently	Made 5 smaller commits in Sprint 2 vs 4 larger ones in Sprint 1	Better incremental delivery demonstration
Test edge cases first	Maintained test suite, all 21 tests passing	No regressions introduced

What Went Well Across Both Sprints

1. Planning Set Us Up for Success

- Sprint 0 planning with detailed user stories made implementation straightforward
- Story point estimates were accurate (completed 7 points as planned in Sprint 1)
- Definition of Done provided clear acceptance criteria

2. CI/CD Pipeline Worked Flawlessly

- GitHub Actions caught issues immediately on push
- Matrix testing (Node 18.x and 20.x) ensured compatibility
- Pipeline passed on every push throughout the project

3. Test-Driven Development Paid Off

- 21 comprehensive tests covering all endpoints
- 91%+ code coverage maintained throughout
- Tests served as documentation and caught regressions

4. Retrospective-Driven Improvement

- Sprint 1 retrospective identified 3 concrete improvements
- All 3 improvements were successfully applied in Sprint 2
- Visible improvement in commit granularity and observability

Challenges Faced

1. Git Authentication Configuration

- **Issue:** Credential conflicts when pushing to GitHub
- **Impact:** Lost time troubleshooting, had to use GitHub Desktop
- **Lesson:** Verify Git configuration before starting projects

2. Scope Creep in Sprint 1

- **Issue:** Implemented US-4 and US-5 ahead of schedule
- **Impact:** Less obvious incremental delivery across sprints
- **Lesson:** Stick strictly to sprint commitments for clearer demonstration

3. Logging Added Late

- **Issue:** Should have included logging from initial setup
- **Impact:** Had to retrofit logging in Sprint 2
- **Lesson:** Include observability as part of initial project scaffolding

Key Learnings

About Agile Practices

1. **User Stories Work**
   - Breaking work into user stories with clear acceptance criteria made it easy to know when "done" was done
   - Format "As a [user], I want [feature], so that [benefit]" forces thinking about value
2. **Sprint Planning Reduces Waste**
   - Prioritizing backlog by value and dependencies prevented building the wrong things first
   - Estimating in story points helped set realistic expectations
3. **Retrospectives Drive Improvement**
   - Specific, actionable improvements (not vague goals) led to real changes
   - Reviewing what went wrong without blame enables honest assessment

About DevOps Practices

1. **Automate Everything Possible**
   - CI pipeline running tests automatically = confidence in deployments
   - Manual testing doesn't scale and is error-prone
2. **Observability is Not Optional**
   - Logging should be built in from day one, not retrofitted
   - Structured logs (JSON) are easier to parse and search than plain text
3. **Small, Frequent Commits**
   - Smaller commits are easier to review and debug
   - Clear commit messages tell the story of development
4. **Health Endpoints Are Essential**
   - `/health` endpoint enables monitoring and load balancer integration
   - Including uptime and memory metrics helps diagnose issues

What I Would Do Differently

1. **Start with Logging Infrastructure**
   - Set up logger utility in Sprint 0 planning phase
   - Include logging in Definition of Done from the beginning
2. **Configure Git/GitHub First**

- Verify credentials and SSH setup before writing code
- Test push access early to avoid last-minute issues

3. **Stricter Sprint Boundaries**
   - Only work on committed stories for the sprint
   - Move ahead-of-schedule work to next sprint for clearer demonstration
4. **Add Integration Tests**
   - Current tests are good but could add end-to-end scenarios
   - Test the full request → response → logging flow

Final Project Statistics

Metric	Value
Total User Stories	7
Completed User Stories	7 (US-1 through US-5 + health + logging)
Total Commits	10
Total Tests	21
Code Coverage	91%+
CI Pipeline Runs	Multiple (all passing)
Sprint Velocity	100% (Sprint 1: 7/7 points)

Conclusion

This project successfully demonstrated the application of Agile principles and DevOps practices:

- **Agile:** Product backlog, user stories, sprint planning, reviews, and retrospectives
- **DevOps:** Version control, CI/CD pipeline, automated testing, monitoring/logging

The retrospective process proved particularly valuable — identifying specific improvements in Sprint 1 and applying them in Sprint 2 showed measurable progress in commit granularity and system observability.

Key takeaway: **Process discipline and iterative improvement matter as much as the final product.**

Repository

GitHub: https://github.com/illonaaddae/task-manager-api-assessment