**What Worked?**

* **Business logic tests** were excellent - vote recording tests with comprehensive validation, edge cases, and error handling worked perfectly
* **Utility function tests** were well-structured and covered all scenarios (percentage calculations, vote formatting, sorting)
* **Validation tests** provided thorough coverage of input validation with realistic edge cases
* **Test organization** with clear describe blocks and meaningful test names was professional
* **Mock setup patterns** followed Jest best practices correctly

**What Didn't?**

* **Component tests failed completely** - AI generated tests for non-existent components (PollResultsChart, CreatePollForm)
* **Jest configuration issues** - Required manual setup of React imports, environment variables, and JSX parsing
* **Assumption-based testing** - AI assumed standard UI patterns and components that weren't actually implemented
* **File structure mismatch** - Generated tests for components that didn't exist in the project structure

**What Surprised You?**

* **Quality gap between logic and UI tests** - Business logic tests were nearly perfect while component tests were completely unusable
* **AI's inability to verify component existence** - Generated comprehensive tests for components it couldn't actually see
* **Excellent edge case coverage** - AI anticipated realistic failure scenarios (duplicate votes, inactive polls, anonymous restrictions)
* **The need for extensive Jest setup** - Simple tests required significant configuration work
* **Perfect mock structure** - Despite component failures, the mocking patterns and test structure were exemplary

**Key Takeaway:** AI excels at generating business logic tests with comprehensive scenarios but struggles significantly with UI component testing without actual component visibility. Always verify component existence before generating UI tests.