Top 3 Non-Functional Requirements Prioritized

Performance

Why chosen: Our platform involves real-time interactions like cart updates and order processing. Fast response times are essential to avoid frustrating the user during checkout or when browsing.

Status: Partially met. Most API responses are under 2 seconds, but order tracking and admin dashboard updates occasionally spike in response time.

Maintainability

Why chosen: With a growing backend and frontend codebase, it's important that future team members can understand and extend the codebase quickly. Maintainability was crucial for long-term viability.

Status: Improved. Sprint 4 introduced clearer route handling, comments, and bug-fix commit messages, but further refactoring (e.g., breaking up large server.js) remains.

Usability

Why chosen: Our primary users are students and parents who expect an intuitive, easy-to-navigate experience. Usability was prioritized to reduce friction when tracking orders or managing carts.

Status: Mostly met. All core pages are intuitive and tested with real flows; however, mobile usability remains an open issue.

Trade-Off Mades

Trade-off 1: Reduced Mobile Responsiveness to Focus on Stability

To prioritize core functionality, we decided to delay implementing full mobile responsiveness. Stability and a bug-free experience on desktop were more urgent to ensure we could deliver a reliable MVP. By doing so, we ensured that the checkout flow, admin panel, and package logic all worked smoothly across the primary use case without overextending our development resources.

Trade-off 2: Skipped Optimization for Package-Stock Queries

Instead of focusing on SQL query optimization for inventory logic, we chose to prioritize correctness and simplicity in our implementation. This made the logic easier for all team members to understand and test, which reduced the likelihood of critical bugs during checkout and package handling. It also improved team collaboration and helped us meet sprint deadlines more effectively.