Critical Path & Dependencies

- SCRUM-52 (Fix package items in orders) → SCRUM-56 (Fix inventory logic for packages)
- SCRUM-54 (Fix: Can't add products and packages to cart) → SCRUM-55 (Fix: products/packages should appear under correct categories)

These tasks were technically linked: proper package display and cart insertion needed to be fixed first before validating the visibility of items and calculating stock availability correctly.

What went wrong

- The inventory logic for packages was more challenging than anticipated. Calculating available stock based on the minimum of bundled items required careful query logic and multiple database updates.
- There was misalignment between frontend expectations and backend implementation, especially when integrating cart behavior and stock deductions.

What we did

- Held a mid-sprint debugging session to review database triggers and cart flow issues.
- Prioritized blockers like cart and order visibility bugs before moving on to cosmetic or minor UI tasks.
- Increased cross-functional communication (frontend/backend pairing) to reduce friction and resolve mismatches.

Lesson Learned

- It's important to finalize backend stock logic early when working with dynamic data like bundled packages.
- Dependencies across technical tasks should be clearly visualized during sprint planning to prevent redundant work.
- Mid-sprint triaging and reprioritization are crucial for keeping delivery on track under shifting priorities.