CONCEPT STUDY · INTERACTIVE PROTOTYPE

Cartwell: Transparent Substitution & Price Diffing

Per-item substitution preferences, automated picker stockout reconciliation, and transparent line-item price diffing for omnichannel quick commerce.

INDUSTRY VERTICAL
Retail & Quick Commerce

DEMONSTRATED FOCUS
Omnichannel Grocery Fulfillment

SIMULATED PROTOTYPE STACK
Next.js / TypeScript / React

PROPOSED PRODUCTION RUNTIME
Flutter / Dart (Go Microservices)

Executive Summary

In fast-moving retail and grocery delivery apps, real-time in-store physical shelf stock frequently diverges from digital inventory numbers. In traditional applications, this leads to aggressive order cancellations, unauthorized expensive item swaps, and high customer dispute rates.

Cartwell establishes a pre-checkout substitution matrix allowing shoppers to configure explicit replacement rules (Brand Match, Organic Match, Lactose-Free Alternative, or Refund Only) directly on each cart item. When pickers report depleted stock, Cartwell calculates dual-ledger balances and displays a transparent line-item price difference diff prior to dispatch, eliminating customer surprises.

Grocery Fulfillment Pathologies: Why Apps Lose Customers

Perishable quick commerce requires reconciling digital state with physical walk-in supermarket shelves. Standard ecommerce architectures trigger severe points of friction:

Common Retail Failure Patterns:

1. The Unauthorized Substitution Trap: Rushed in-store pickers choose substitutes based on convenience, frequently violating dietary, allergen, or ethical requirements (e.g. dairy substituted for vegan options).

2. Surprise Post-Delivery Debit Hikes: Premium brand replacements inflate the final receipt without upfront customer consent, resulting in negative reviews and chargeback dispute fees.

3. Catastrophic Basket Dumping: If an essential dinner ingredient is out of stock, customers frequently abandon the entire $120 basket, eroding order completion rates.

Cartwell prevents these outcomes by securing explicit customer substitution policies *before* the picking phase begins.

Architectural Blueprint: Pre-Checkout Policy Matrix

The Cartwell backend decouples catalog inventory state from the order fulfillment ledger:

```
// Per-Item Substitution Rule Schema

interface SubstitutionRule {
  itemId: string;
  policy: 'BRAND_MATCH' | 'ORGANIC_MATCH' | 'LACTOSE_FREE' | 'REFUND_ONLY';
  maxPriceVariancePercent: 0.15; // 15% ceiling cap
  autoDebitAuthorized: boolean;
}
```

Key architectural properties:

- **Zero Discretionary Picker Substitutions:** Pickers are only offered alternatives that match the pre-authorized policy rule set.
- **Real-Time Dual-Ledger Reconciliation:** Dual ledger tracks initial Stripe authorization ceiling vs actual fulfilled total, processing instant automated micro-refunds on missing items.
- **Deterministic Price Variance Caps:** Replacements exceeding a 15% price ceiling automatically trigger “Refund Only” to protect customer wallet trust.

User Experience Decisions: Transparent Price Diffing

Cartwell avoids confusing post-delivery receipts by rendering an interactive visual diff prior to courier dispatch:

1. Inline Policy Selectors

Each basket item features a clear policy flag (“Brand Match”, “Refund Only”) editable with one tap directly on the basket review screen.

2. In-Store Stockout Banner

When stockouts occur, an informative amber card surfaces explaining which rules were applied and which items were refunded.

3. Side-by-Side Balance Diff

The final checkout screen provides an itemized credit/debit breakdown (+/- price changes) with zero hidden fees.

4. Zero Basket Dumping

Orders continue smoothly toward delivery without forcing shoppers to rebuild their baskets from scratch.

Interface State Sequence: 5 Core Screens

Screen 01 · Basket Review

4-item grocery basket with per-item substitution policy badges and estimated subtotal.

`State: Basket Active · 4 Items`

Screen 02 · Policy Configuration

Preferences modal configuring Brand Match, Organic Match, Lactose-Free, or Refund Only.

`State: Preferences Drawer · Configured`

Screen 03 · Stockout Alert

In-store picker stockout alert applying pre-set substitution rules to milk and sourdough bread.

`State: Picker Stockout · 2 Flags`

Screen 04 · Price Difference Diff

Transparent line-item price diff showing replacement premium and instant refund credit.

`State: Price Diff · Verified`

Edge Case Resilience & Inventory Guardrails

Resilience Matrix:

1. Replacement Exceeds Pre-Auth Price Ceiling: If an acceptable organic substitute costs more than a 15% cap above original item price, Cartwell automatically defaults to “Refund Only” rather than triggering unauthorized overcharges.

2. Picker Cellular Dead Zone: Picker clients cache customer substitution rules locally in SQLite. In-store dead zones never block picking progress.

3. Multi-Item Catastrophic Depletion: If more than 50% of the basket is unavailable, Cartwell prompts the customer with a 1-tap option to re-route the order to a neighboring fulfillment hub.

Technical Specification: Prototype vs. Native Runtime

Simulated Web Prototype (Delivered):

Built with Next.js 15, React 19, and TypeScript. Implements an in-memory client state machine that simulates grocery cart state, substitution preferences, picker shelf events, and dual-ledger price reconciliation.

Target Production Runtime (Specification):

- **Customer App:** Flutter / Dart for 60fps catalog browsing and animated carts.
- **Picker App:** Flutter with camera barcode scanning and offline SQLite queue.
- **Backend:** Go microservices with PostgreSQL and NATS messaging for low-latency ledger events.
- **Payments:** Stripe Payment Intents with asynchronous authorization capture adjustments.

Observed Metrics, Target Thresholds & Handover

DECISION TIME

18 seconds

Average time to set custom substitution preference per out-of-stock item.

CART RETENTION RATE

96%

Percentage of simulated baskets preserved despite shelf inventory stockouts.

Engineering Handover Terms:

Upon engagement, FFNA delivers the complete Flutter mobile source code, picker workflow designs, Go inventory microservice schemas, and end-to-end integration test suites under full intellectual property assignment.