iOS Capstone Project Proposal: Stock Counter App

Brand Mission Statement:

> **Rasuto** helps collectors and deal hunters stay ahead of the curve by alerting them when high-demand items are nearly gone. We empower our users with real-time stock insights, so they never miss out on that *last one*.

Brand Ethos:

“Rasuto” is derived from the Japanese phonetic spelling of the English word “Last” (ラスト). In Japanese pop culture and retail, **“Rasuto”** often refers to the **last available item**, the **final chance**, or the **ultimate edition** — creating urgency, significance, and emotional value.

It evokes a mindset of:

- **Exclusivity** — “The last one. Just for you.”
- **Timeliness** — “Now or never.”
- **Mindful collecting** — “Cherish what matters before it disappears.”

Rasuto isn’t just about counting stock.

It’s about helping **collectors, enthusiasts, and deal hunters** capture fleeting opportunities before they vanish — whether that’s the final size 10 in a grail sneaker drop or a rare colorway of tech gear.

Minimalist Brand Tone:

- Calm.
- Thoughtful.
- Forward-thinking.
- Inspired by Japanese minimalism and scarcity-driven design cues.

🎨 Logo Design Ideas

Style Direction:

- Minimalist and modern
- Inspired by **Japanese streetwear branding**, tech apps, and product scarcity aesthetics
- Great on light/dark backgrounds

Typography-based options:

- Custom stylized typeface for “Rasuto” (sans-serif, mono-style, or slightly rounded)
- Consider a **small katakana “ラ”** or “ラスト” icon variant for Japanese subcultural vibes

Symbolic logo (optional):

- “1” in a circle (like a limited prize marker)
- Clock icon stylized as a minimalist drop countdown
- A disappearing bar/stock bar concept

📱 App Icon Concepts

Icon Type: Flat and modern, optimized for iOS
Base Shape: Rounded square (iOS default)

**Ideas:

Concept	Visual	Notes
Lettermark	“R” or “ラ” in a sleek font	Clean, minimalist
Last One Symbol	Stylized number “1” fading out or glowing	Emphasizes scarcity
Drop Countdown	Vertical bar shrinking or low stock graph	Intuitive function
Shoe Tag / Barcode	Subtle sneaker culture reference	Very hype-friendly

Color Palette Suggestions:

- **Black + White** = sleek, premium, sneaker culture roots
- **Muted earth tones** = nod to techwear/fashion
- **Accent color** = Red (for urgency) or Neon Green (for alerts)

⏳ Splash Screen Design

Goal: Immediate emotional connection. Users should think: *“I’m not missing out this time.”*

Key Elements:

- **Logo centered** (animated fade-in or zoom)
- **Dark background or ultra-clean white**
- **Optional tagline:**

> “Never miss the last one.”
> OR
> “Track the drop. Stay ahead.”

- **Slight animation**: e.g., a loading bar shrinking, symbolizing diminishing stock

📁 Figma Setup Tips for Rasuto UI

Splash Screen	iPhone 15 Pro – 393x852pt	Logo fade-in, tagline, loading animation
Login/Onboarding (Optional)	Same frame	Optional for Checkpoint 1, helps with flow
Main Interface	iPhone 15 Pro	Tabs: Home, Watchlist, Favorites, History
Product Detail View	iPhone 15 Pro	Stock graph, alert button, wishlist toggle
Search + Filter UI	iPhone 15 Pro	Top search bar, filters by brand/category
Wishlist/Alerts View	iPhone 15 Pro	Focus on saved items, with stock alerts
Settings/About (Optional)	iPhone 15 Pro	Shows version, support, etc.

Tips:

- Use **Auto Layout** for scalable elements
- Set up a **Components Page** for reusable UI items (e.g., buttons, cards, alerts)
- Create a **Color Styles & Typography Tokens** file to stay consistent
- If needed, I can help build a **mini design system**

1. Project Overview

Stock Counter is an iOS app that helps shoppers track product availability in online stores and notifies them when a specific item is **low in stock** or **about to sell out**. Users can add products they’re interested in, and the app will **monitor inventory levels** using **web scraping or retailer APIs**, providing real-time alerts.

2. Core Functionalities

✅ **Product Tracking** – Users can add links to products they want to track from supported retailers.

✅ **Stock Monitoring** – The app periodically checks the stock availability of tracked products.

✅ **Low Stock Alerts** – Users receive notifications when an item reaches a predefined low stock threshold (e.g., “Only 1 left!”).

✅ **Wishlist & Favorites** – Users can save and organize products for later tracking.

✅ **Search & Filtering** – Users can quickly find tracked products and filter by category, retailer, or stock status.

✅ **Historical Stock Data (Optional)** – Users can see past stock fluctuations to predict trends.

3. Target Audience

- **Sneaker collectors** (Nike, Adidas, etc.)
- **Tech enthusiasts** (Limited edition gadgets, GPUs, iPhones)
- **Fashion shoppers** (Drops from Supreme, designer brands)
- **Deal hunters** (Amazon, Best Buy, Walmart shoppers looking for last-minute discounts)

4. Technology Stack

- **Language:** Swift, SwiftUI
- **Data Persistence:** SwiftData or UserDefaults for storing user preferences and tracked items
- **Networking:** URLSession for fetching stock data from retailer APIs
- **Concurrency:** Async/Await for handling stock checks in the background
- **Notifications:** Local notifications & push notifications for stock alerts
- **API Integration:** Attempt integration with known retailer APIs (Amazon, Best Buy, Walmart)
- **Web Scraping (if APIs unavailable):** Use SwiftSoup or a server-side function to extract stock data

5. Required Capstone Deliverables

✅ **Splash Screen & Custom App Icon** – Branding with a sleek, modern UI

✅ **Data Persistence** – Save tracked products locally

✅ **Proper Layout & Navigation** – Well-structured UI using SwiftUI

✅ **Concurrency** – Background stock checks using async/await

✅ **URLSession API Calls** – Fetch stock data from online sources

✅ **Error Handling** – Graceful handling of network failures

6. Challenges & Considerations

- Some retailers may **not provide stock APIs** (might require web scraping).
- Frequent stock checks could trigger rate limits—must **optimize API calls**.
- **Data accuracy** is crucial—if stock counts are unreliable, notifications might misfire.
- If scraping is needed, an **intermediate server** might be required.

📅 Project Milestones & Timeline: Stock Counter App

(2-3 months: **~10 weeks**)

🟢 Phase 1: Planning & Research (Week 1)

📌 **Deliverables:**

- Finalize **project proposal** & get mentor approval
- Research **retailer APIs** (Amazon, Best Buy, Walmart, etc.)
- Sketch **Figma wireframes** for core UI

🎯 **Goal:** Get API access & define UI/UX before development begins.

🟡 Phase 2: Core App Setup & UI Development (Weeks 2-3)

📌 **Deliverables:**

- Set up **Swift project** in Xcode with GitHub repo
- Build **Splash Screen & Custom App Icon**
- Implement **Main UI**:
  - Product tracking list
  - Add new product screen
  - Settings & notifications
- Implement **SwiftData/UserDefaults** for local storage

🎯 **Goal:** Get a working UI with basic data persistence.

🟠 Phase 3: Networking & Stock Tracking Logic (Weeks 4-5)

📌 **Deliverables:**

- Implement **URLSession** for stock tracking API calls
- Develop **background task** to check stock levels
- Set up **async/await** for concurrency
- Implement **error handling** for API failures
- Build **low stock alerts** (push/local notifications)

🎯 **Goal:** Enable real-time stock tracking with notifications.

🔵 Phase 4: Testing & Optimization (Weeks 6-7)

📌 **Deliverables:**

- Conduct **unit tests** for stock tracking logic
- Debug & optimize API requests
- UI/UX refinements based on testing
- Improve **performance & reliability**

🎯 **Goal:** Ensure the app runs smoothly & reliably.

🟣 Phase 5: Finalization & Submission (Weeks 8-10)

📌 **Deliverables:**

- **Prepare GitHub repository** for submission
- **Create a short demo video** showcasing app features
- **Submit final app & documentation** for evaluation

🎯 **Goal:** Deliver a polished capstone project within the 2-3 month timeframe.

🔥

Key Adjustments for a Shorter Timeline:

✅ Removed **historical stock tracking** (can be added later)

✅ Streamlined **testing phase** to focus on critical issues

✅ Minimized **extra UI/UX enhancements** to keep it lean

✅ Focused on **core stock tracking & notifications** as the MVP