Development Status & Technical Report: CCC eCommerce Command Center

> **NOTE** This document provides a deep technical scan and status assessment of the CCC eCommerce Command Center. It details the current architecture, implemented features, integration status, and development roadmap.

1. Executive Summary

Project Identity: A merged application combining the robust backend of "CCC" with the modern "Commerce-Canvas" UI. **Current Stage**: **Late Alpha / Early Beta (Foundation Complete)**. **Core Value**: An enterprise-grade operating system for e-commerce, integrating multi-platform ad management, AI-driven content generation, and centralized brand control.

The application has a fully established shell, navigation, and design system. The backend infrastructure (Supabase, Workers) is in place. Key integrations (Google, Meta, OpenAI) are scaffolded. The primary focus is now on **wiring the UI to the Backend services** and **polishing complex AI workflows**.

2. Technical Architecture

Stack Overview

- **Framework**: Next.js 14 (App Router)
- **Language**: TypeScript (Strict Mode)

- **Styling**: Tailwind CSS + Shadcn UI (Glassmorphism Theme)
- **Database**: Supabase (PostgreSQL)
- **Auth**: Supabase Auth
- **Background Jobs**: pg-boss
- **AI/ML**: Multi-provider (Google Vertex, OpenAI, Replicate, ElevenLabs)

Directory Structure (`src/`)

- **`app/`**: Next.js App Router handling 4 main verticals:
- - `studio/`: Creative workspace (Ad creation, Campaigns)
- - `brand/`: Brand identity, Voice, Assets
- - `commerce/`: Product catalog, Inventory features
- - `design/`: Design system reference
- - `api/`: Serverless functions for integrations
- **`components/`**: Modular UI architecture
- - `ui/`: 55+ reusable primitive components (shadcn compatible)
- - `ad-creation/`, `marketing/`, `copilot/`: Feature-specific logic
- **`lib/providers/`**: The "Brain" of the integrations layer.

3. Deep Scan: Feature Implementation Status

A. Core Platform (✅ Stable)

- **App Shell**: Glassmorphism layout with 3-tier navigation is fully implemented.
- **Authentication**: Supabase Auth integration is set up.
- **Database**: Migrations and schema (`migrations-combined.sql`) are comprehensive.

B. "Studio" & AI Generation (🚧 In Progress)

- **Product Persona Generator**:
- - **Status**: Logic exists in `product-persona-generator.ts`.

- - **Critical Issue Identified**: The Vertex AI implementation currently uses the `:predict` endpoint but fails to pass reference images in the payload. This effectively breaks "product preservation" in generated images.
- - **Fix Required**: Switch to `generateContent` endpoint or fix payload structure.
- **Video Generation**: `image-to-video` routes exist, using providers like Runway/Replicate (implied).
- **Ad Creation**: UI components exist in `src/components/ad-creation`.

C. Integrations Layer (✅ Scaffolded)

The `src/lib/providers` directory reveals an impressive array of connectors already written:

Category	Providers	Status
Generative AI	OpenAI, Gemini, Google Imagen, Flux, Replicate	**High** (Code exists)
Media AI	ElevenLabs (Voice), Synclabs (Lip sync), Rembg (Bg removal)	**High**
Ad Platforms	Google Ads, Meta (Facebook/Insta), TikTok	**Medium** (Scaffolded)
E-Commerce	Lazada, WooCommerce, Firecrawl (Scraping)	**Medium**

D. Data & Commerce (🚧 In Progress)

- **Catalog Sync**: Routes exist (`/catalog/sync`), but deep bi-directional sync (e.g. real-time Lazada inventory) likely needs robustness testing.
- **Brand Identity**: The `brand` section is structured to hold logos, colors, and "Brand Voice", essential for consistent AI generation.

4. Current Development Stage: "Integration & Refinement"

The project has passed the **"Greenfield"** phase. You are not building from scratch. You are in the **"Integration"** phase.

What is the software doing RIGHT NOW? 1. **Serving a UI**: It successfully renders a complex, modern dashboard. 2. **Managing Data**: It connects to Supabase for user profiles and settings. 3. **Ready for Jobs**: The `pg-boss` worker system is ready to handle long-running tasks (like video generation or data sync) asynchronously. 4. **Waiting for Wiring**: Many UI buttons likely look good but need their `onClick` handlers connected to the `lib/actions` or API routes.

5. Critical Next Steps & Recommendations

Priority 1: Fix AI Generation Pipeline

> **IMPORTANT** The Product-Persona Generator is the "Killer Feature". It is currently buggy.

- **Action**: Modify `vertex-imagen-provider.ts` to correctly handle reference images (Product + Persona) when calling Google's API.

Priority 2: Connect "Studio" UI to Backend

- **Action**: Ensure the "Create Campaign" forms in `/studio` actually trigger the `pg-boss` jobs to dispatch ads to Meta/Google.
- **Action**: Verify the "Audience Generation" flows are persisting results to Supabase.

Priority 3: Hardening Integrations

- **Action**: Test the `Lazada` and `TikTok` connectors with real credentials to ensure token refresh logic is robust.

6. Conclusion

The CCC Command Center is a sophisticated integration platform. It is architecturally sound but requires focused effort on **debugging specific AI pipelines** and **completing the "Last Mile" wiring** between the frontend components and the backend providers.

--- *Report generated based on codebase analysis at `2026-01-19`.*