Development Process for the Full-Stack AI Multi-Vendor E-Commerce Platform

The development of the full-stack AI multi-vendor e-commerce site followed a systematic approach, moving from initial setup and data preparation to feature implementation and final deployment.

---

1. Project Initialization & Setup

This phase involved setting up the foundation, including core dependencies and initial data.

1. **Initial Project Setup:** The project was set up using **Next.js** as the main framework. 2. **Environment Configuration:** Environment variables were added to configure access to various services. 3. **Dummy Data Creation:** Structural data for categories, ratings, stores, products, and addresses were created in the code to facilitate rapid feature development and testing. 4. **Authentication Setup:** The **Supabase Authentication** module was configured to manage user sign-up, login, and session persistence securely.

---

2. Database & Event Integration

The next steps focused on connecting the application to its persistence and event-handling layers.

1. **Database Connection (Supabase/PostgreSQL):** The project was configured to use **Supabase PostgreSQL**, which provides an integrated managed database, authentication, and real-time API. Prisma was optionally used as the ORM layer for schema management. 2. **Schema Migration:** The Prisma schema was defined and synchronized with Supabase using the command `npx prisma db push`, ensuring all tables were created automatically. 3. **Event Handling Setup:** Supabase's real-time and function triggers were utilized to handle background events such as order creation, status updates, and notification dispatch.

---

3. Core Feature and Service Implementation

This was the main development phase where the application logic and external services were integrated.

1. **Cloud Storage Integration (Supabase Storage):** Product images and store logos were uploaded and managed through Supabase Storage, replacing third-party services like ImageKit. 2. **Database Logic Implementation:** Backend logic was implemented using Prisma and Supabase APIs, enabling functions such as linking newly created stores to users, order management, and analytics. 3. **Developing User Interfaces and API Routes:** The team built all major features and routes, including: * Customer Checkout and Order Placement * Seller Dashboard (Order view, Product management) * Admin Dashboard (Store approval, Coupon creation) * Logistics Service Integration (Allow vendors and sellers to find logistics partners for delivery) 4. **AI Feature Integration:** The core AI feature was implemented using the **Google Gemini API** (via `OPEN_AI_API_KEY`). This AI model automatically generates product names and descriptions from uploaded images, which are then displayed dynamically in the "Add Product" form.

---

4. Deployment

The final step was to prepare the application for a live environment.

1. **Code Commit and Sync:** All finalized features were pushed to GitHub for version control. 2. **Cloud Configuration:** Environment variables (Supabase API keys, Google AI keys, etc.) were added securely in the **Vercel project settings**. 3. **Continuous Deployment:** The project was deployed seamlessly on **Vercel**, providing automatic build previews, HTTPS, and scaling. 4. **Monitoring & Optimization:** Supabase dashboards were used for database monitoring, and Vercel analytics tracked real-time performance and user traffic.

Outcome: The platform achieved full integration of e-commerce features, AI-driven automation, and logistics connectivity — all running efficiently with a scalable Next.js + Supabase + Vercel stack.