Marketplace Technical Hackathon

Introduction

This document presents the technical framework for building an E-Commerce Platform, designed to support small businesses and entrepreneurs by offering an online marketplace for their products. The planning incorporates insights from Hackathon Day 1 discussions and recommendations derived from Day 2 guidelines.

Core Technologies

- **Frontend Framework:** Next.js
- **Content Management System:** Sanity CMS
- **Order & Shipping Management:** ShipEngine
- **Database Solution:** MongoDB (for authentication and data storage)
- **Hosting & Deployment:**
    - **Frontend:** Vercel
    - **Backend:** AWS
- **Payment Processing:** Stripe, JazzCash, EasyPaisa, KeenuPay

System Architecture

Overview

1. **Frontend (Next.js)**

Uses client-side rendering for responsiveness.

    - Server-side rendering (SSR) is leveraged for SEO and preloading product pages.
    - Integrates Sanity CMS for managing dynamic content efficiently.
2. **Backend Services**
    - Provides RESTful APIs for handling user authentication, product management, order processing, and shipping.
    - Ensures secure data validation and manages external API integrations.
3. **Content Management (Sanity)**
    - Manages dynamic content such as promotional banners, product highlights, and blog articles.
4. **Order & Delivery Tracking (ShipEngine)**
    - Enables real-time order tracking and shipment management.
5. **Authentication (Clerk)**
    - Secure user authentication and session management.
6. **Deployment Strategy**
    - **Frontend:** Hosted on Vercel for fast and reliable delivery.

System Components & Workflow

User Authentication

- **Inputs:** Email & password
- **API Endpoints:**
    - `POST /api/auth/register` - Registers a new user.

    - `POST /api/auth/login` - Authenticates user login.
    - `GET /api/auth/verify` - Confirms authentication status.
- **Outcome:** Issues JWT tokens for managing user sessions.

Content Management (Sanity CMS)

- **Admin Role:** Oversees product listings, homepage banners, and blog posts.
- **API Integration:** Utilizes GROQ queries for fetching data dynamically.
- **Outcome:** Ensures seamless content updates that reflect instantly on the frontend.

Product Browsing & Checkout

- **Frontend Implementation:** Utilizes Next.js with SSR for product pages.
- **Database Structure:** Stores product details such as name, price, stock, description, and images in MongoDB.
- **API Endpoints:**
    - `GET /api/products` - Retrieves product listings.
    - `GET /api/products/:id` - Fetches details of a single product.
    - `POST /api/products` - Allows sellers/admins to add new products.
- **Outcome:** Enables users to explore, add items to the cart, and complete purchases.

Order Processing

- **Database Schema:** Maintains order records with customer and product references.
- **API Endpoints:**
  - `POST /api/orders` - Creates an order entry.
  - `GET /api/orders` - Retrieves order history for users.
  - `GET /api/orders/:id` - Fetches specific order details.
- **Outcome:** Orders are processed securely and cannot be modified once confirmed.

Shipping & Tracking (ShipEngine)

- **Integration:** Connects with ShipEngine API for real-time tracking.
- **API Endpoint:** `GET /api/shipments/:orderId`
- **Outcome:** Customers receive live delivery updates.

Payment Processing (Stripe, JazzCash, EasyPaisa, KeenuPay)

- **Integration:** Supports multiple secure payment gateways.
- **API Endpoints:**
  - `POST /api/payments` - Initiates a transaction.
  - `GET /api/payments/status` - Retrieves payment confirmation.
- **Outcome:** Transactions are processed securely before finalizing orders.

Key System Components & Interactions

Frontend (Next.js)

- Manages user interactions and dynamically renders data fetched from APIs.
- Communicates with backend for authentication, product listings, and order processing.

Backend Services

- Exposes RESTful APIs for handling users, orders, products, and shipping.
- Integrates with external services such as ShipEngine and payment gateways.

Database (MongoDB)

- Stores structured data for users, products, and orders.
- Provides a flexible schema design that accommodates future enhancements.

Sanity CMS

- Maintains up-to-date marketing content and dynamic product listings.

Data Model

Users

- `userId` (Unique Identifier)

- `username`
- `email`
- `passwordHash`
- `role` (Admin, Seller, Buyer)
- `orderHistory` (List of past orders)
- `listedProducts` (For sellers only)

Products

- `productId` (Unique Identifier)
- `name`
- `price`
- `stockAvailability`
- `description`
- `imageUrls`
- `variations` (Optional)
- `sellerId`

Orders

- `orderId` (Unique Identifier)
- `customerId`
- `productList`
- `orderStatus` (Pending, Confirmed, Delivered)
- `orderTimestamp`

Delivery Zones

- `zoneId` (Unique Identifier)
- `zoneName`
- `coveredAreas`
- `assignedDrivers`

Sellers

- `sellerId` (Unique Identifier)
- `storeName`
- `contactEmail`
- `inventory`
- `shippingRegions`

Third-Party Integrations

- **Sanity CMS**
  - Manages promotional banners and category updates.
  - Uses GROQ for optimized content retrieval.
- **ShipEngine**
  - Generates shipment labels and tracks deliveries.
  - Provides real-time logistics updates.
- **Stripe Payment Gateway**
  - Ensures PCI-compliant payment processing.
  - Handles refunds and customer disputes.

Deployment & Hosting Strategy

- **Frontend:** Hosted on Vercel with GitHub-based CI/CD.

Security Measures

1. **Data Encryption:**
    - Implements HTTPS for secure data transmission.
    - Encrypts sensitive information such as passwords.
2. **Authentication & Access Control:**
    - Uses JWT-based authentication.
    - Implements role-based permissions for different user types.
3. **Transaction Security:**
    - Utilizes Stripe's PCI-compliant APIs.
    - Ensures fraud protection and dispute resolution mechanisms.
4. **API Security:**
    - Implements rate limiting to prevent abuse.
    - Validates incoming data to protect against XSS and SQL injections.

Final Thoughts

This plan sets the groundwork for a **scalable, secure, and efficient** E-Commerce platform, ensuring a seamless experience for both sellers and buyers.