# Game Time Gear - Day 2: System Setup & API Integrations

**Prepared by: Syed Huzaifa Ahmed Hashmi**

---

## Introduction

Welcome to Day 2 of building **Game Time Gear**, a dedicated online marketplace for high-quality sportswear and fitness gear. Today, I focus on setting up the essential system components, including authentication, shipping, and payment integrations, as well as refining the frontend structure and database setup.

---

## Objective

By the end of today, I will have:

- Defined the core **frontend pages** required for the e-commerce platform.
- Integrated **Sanity CMS** to manage product data efficiently.
- Configured **Clerk** for authentication services.
- Set up **ShipEngine** for shipping calculations.
- Integrated **Stripe** for payment processing.
- Outlined the system's **data flow** and API usage.
- Documented the progress for future reference.

---

## Frontend Structure

### Key Pages Developed

- **Home Page:** Displays featured products and promotional banners.
- **Products Page:** Lists all available sportswear and fitness products.
- **Product Details Page:** Shows detailed information, images, and pricing.
- **Cart Page:** Allows users to review and manage selected products.
- **Checkout Page:** Facilitates the order placement and payment process.

- **Login/Signup Page:** Handles user authentication via Clerk.
- **Join Page:** Encourages new users to sign up and explore exclusive deals.

### Technologies Used

- **Frontend:** Next.js, Tailwind CSS, TypeScript
- **Backend:** API integrations with Sanity CMS, Clerk, Stripe, and ShipEngine
- **Data Management:** Sanity CMS (for products only)

---

## Backend & API Integrations

### Sanity CMS Integration

- **Purpose:** Managing product data dynamically.
- **Implementation:**
  - Connected Sanity CMS to Next.js for seamless product management.
  - Created a schema for products, including fields such as `ProductID`, `Name`, `Description`, `Price`, `Stock Quantity`, `Category`, `Images`, and `Tags`.
  - Set up Next.js API routes to fetch products from Sanity.

### Authentication with Clerk

- **Purpose:** Secure user authentication.
- **Implementation:**
  - Configured Clerk authentication services.
  - Enabled JWT-based authentication with digital signatures.
  - Integrated sign-in and sign-up functionality within the frontend.

### Shipping with ShipEngine

- **Purpose:** Handling shipping calculations.
- **Implementation:**
  - Integrated ShipEngine API to calculate shipping rates dynamically.
  - Set up API calls to fetch shipping costs based on user location.

### Payments with Stripe

- **Purpose:** Secure payment processing.
- **Implementation:**
  - Configured Stripe for handling transactions.
  - Set up API routes to process payments.

    - Ensured seamless redirection from checkout to Stripe’s secure payment page.

---

# System Architecture & Data Flow

## Data Flow Between Components

1. **User Interaction**
    - Users browse products (fetched from Sanity CMS).
    - Users log in or sign up via Clerk.
    - Users add items to their cart and proceed to checkout.
2. **Checkout Process**
    - Shipping costs retrieved from ShipEngine.
    - Payments processed via Stripe.
    - Order details stored in the database.
3. **Post-Purchase Actions**
    - Users receive confirmation emails.
    - Orders updated in the system.
    - Shipping details managed via ShipEngine.

---

# Challenges Faced & Solutions

## 1. Planning System Architecture

- **Challenge:** Lack of a high-level system diagram.
- **Solution:** Documented data flow and integration details for clarity.

## 2. Choosing the Right Authentication System

- **Challenge:** Deciding between Clerk and a custom authentication solution.
- **Solution:** Opted for Clerk due to its ease of integration and robust security features.

## 3. Understanding API Interactions

- **Challenge:** Managing multiple third-party integrations effectively.
- **Solution:** Structured API calls efficiently and documented each step.

## Next Steps & Future Enhancements

- **Day 3 Plans:**
  - Implement the cart and checkout functionality.
  - Finalize order placement logic.
  - Test API integrations thoroughly.
- **Future Enhancements:**
  - Add product reviews and ratings.
  - Implement real-time order tracking.
  - Enhance user experience with UI improvements.

## End Note

Day 2 was about setting up the core system integrations for **Game Time Gear**. With authentication, shipping, and payment solutions in place, the foundation for the e-commerce platform is now stronger. Moving forward, the focus will be on refining the checkout process and ensuring smooth order management.

**Prepared by Syed Huzaifa Ahmed Hashmi**