DAY: 02

Marketplace Technical Foundation:

[NIKE] an E-commerce website:

Goals:

The goals of this e-commerce website is to provide a easy, convenient, and enjoyable shopping experience for customers, helping them find and purchase the perfect pair of shoes, while offering a wide variety of styles, sizes and brands at competitive prices.

System Architecture overview:

The system architecture of an e-commerce shoe website typically involves multiple layers and components to ensure scalability, performance and security.

a) **User Interface layer (Frontend):**
   This layer provides and the interface for users to interact with the website.
   A responsive and user-friendly website for mobile and desktop users.

b) **Application layer (Backend):**
   This is the core of the system where business logics is implemented. It typically includes:
   APIs: API to connect the frontend with the backend.
   Product Catalogue Management: Handles the storage and retrieval of product information.
   Search Engine: Allows users to search for products using keywords, filters, or categories.
   User Authentications: Secure login, registration.
   Order Management system: Handles order processing and tracking.
   Payment Gateway Integrations: Secure processing of payments via gateways.

c) **Database Layers:**
   This layer manages data storage and retrieval.

d) **Payment Gateway:**
   This component handles secure payment transaction's support credit/debit cards, digital wallets and UPI.

e) **Third-Party Services:**
This is used to enhance functionality and streamline operations like Email and SMS for order confirmations, delivery updates, order fulfillment, real time tracking and marketing campaigns.

System Architecture of Nike website:

Workflow:

•**Engaging Frontend**: A sleek, intuitive interface where users explore, filter, and fall in love with the perfect pair of shoes.

•**Smart Backend Services**: The powerhouse that processes user actions, ensures smooth navigation, and handles business logic.

•**Dynamic Product Catalog**: A rich repository showcasing shoes with detailed descriptions, sizes, styles, and real-time availability.

•**Streamlined Order Management**: Efficiently tracks every step, from cart to checkout to delivery.

•**Secure Payment Gateway**: Ensures hassle-free, encrypted transactions with multiple payment options.

•**Real-Time Inventory Management**: Monitors stock levels and syncs seamlessly to avoid disappointments.

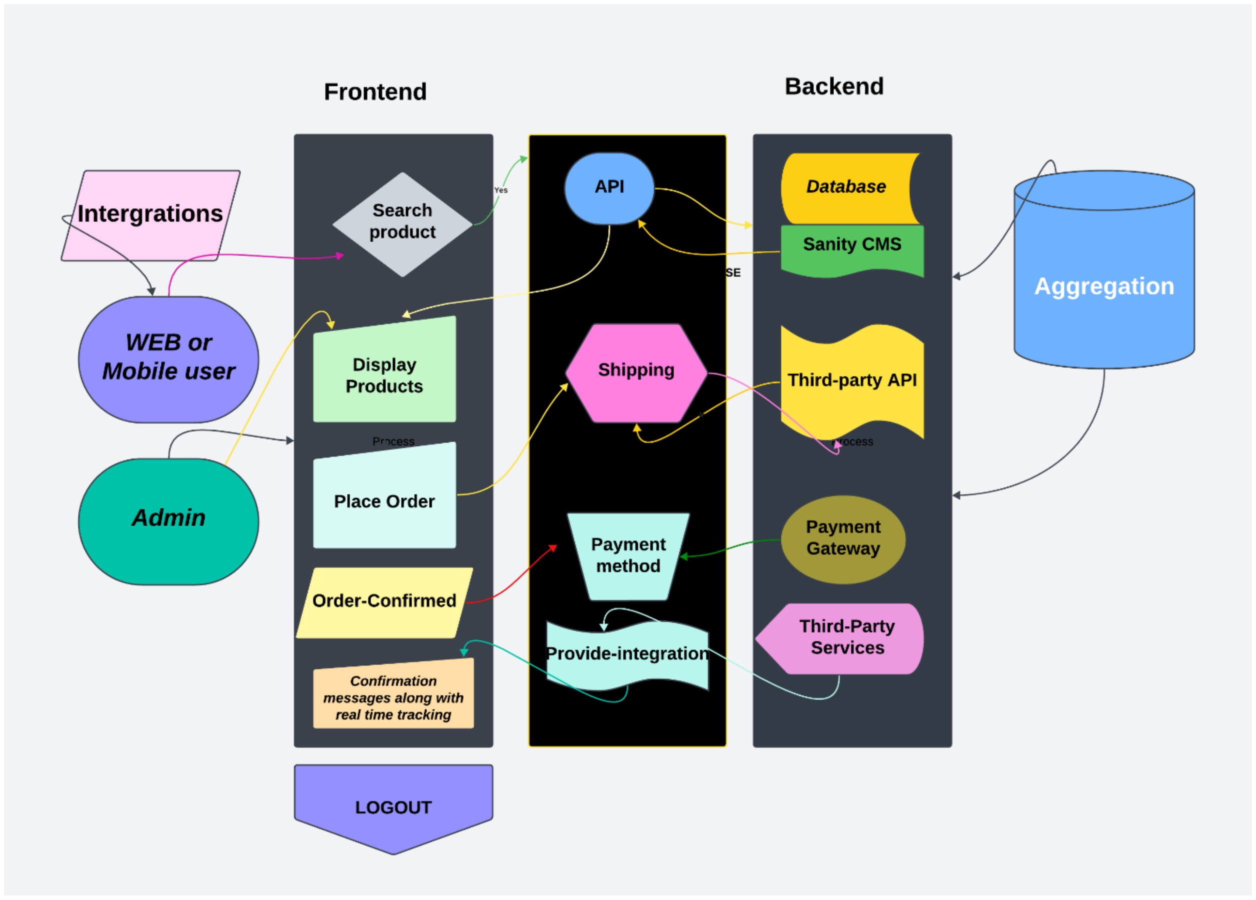
•**Integrated Shipping Module**: Connects with delivery partners to provide real-time tracking and swift deliveries.

•**Powerful Admin Panel**: Empowers management with tools for product updates, order insights, and business analytics.

•**Robust Database**: The heart of the system, securely storing user profiles, product data, and order history.

•**APIs as Connectors**: Bridges the frontend, backend, and third-party services for a cohesive, uninterrupted experience.

Diagram Showing System Architecture:



API Requirements:

API Category	Requirements	Method
Authentication	User registration, login and authentication	**POST**
Product Management	Retrieve product listings, filter, search, manage stock, add/update products.	**GET, POST**

API Category	Requirements	Method
Cart & Checkout	Add/remove items, view cart, checkout, apply discounts, payment, shipping calculation.	**GET, POST**
Order Management	Place orders, update status, order history, returns, refunds, shipment tracking.	**GET, POST**
Payment Processing	Handle payments, refunds, chargebacks via gateways (Stripe, PayPal).	**POST**
Inventory Management	Real-time stock updates, low stock alerts, sync with suppliers.	**GET, POST**
Shipping & Delivery	Calculate shipping rates, courier integrations, address validation.	**GET**
User Profile & Wish list	Manage profiles, wish list, password reset.	**GET, POST**
Review & Rating	Submit/retrieve product reviews and ratings.	**GET, POST**
Admin & Analytics	Admin management, sales data, user behavior reports.	**GET, POST**
Security	Secure communication	**GET**
Notifications	Push, email, and SMS notifications for order updates, confirmations.	**GET, POST**
Search & Recommendations	Full-text search, personalized recommendations.	**GET, POST**

Sanity Schema:

[Product sanity Schema]

```
export default {
name: ’product’,
type: ‘document’,
fields: [
{name: ‘name’, type: ‘string’, title: ‘Product Name’},
{name: ’price’, type: ‘string’, title: ‘Price’},

{name: ‘Image’, type: ‘image’, title: ‘Product Image’}}
```

```
{name: ‘category’, type: ‘reference’, title: ’Category’},
```

[Category Sanity Schema]:

```
export default {
name: ‘Category’,
type: ‘document’,
title: ‘category’,
fields: [
{name: “name”, type: “string”, title: ‘Category Name’},
{name: ‘slug’, type: ‘slug’, title: ‘slug’}
}
```

[Order Schema]:

```
export default {
name: ‘order’,
title: ‘order’,
type: ‘document’
fields: [
{name: ‘orderNumber’, type: ‘string’, title: ‘Order Number’},
{name: ‘customerName’, type: ‘string’, title: ‘Customer Name’},
{name: ‘customerEmail’, type: ‘string’, title: ‘Customer Email’},
{name: ‘totalPrice’, type: ‘string’, title: ‘Total Price’},
{name: ‘orderStatus’, type: ‘string’, title: ‘Order Status’,
options: {
list: [‘pending’, ‘Shipped’, ‘Delivered’, ‘Cancelled’],
}
```

[Customer Schema]:

```
export default {

name: ‘customer’,

type: ‘document’,

title: ‘Customer’,

fields: [

{name: ‘firstName’, type: ‘string’, title: ‘First Name’},
{name: ‘lastName’, type: ‘string’, title: ‘Last Name’},

{name: ‘email’, type: ‘string’, title: ‘Email’},

{name: ‘phoneNumber’, type: ‘string’, title: ‘Phone Number’},

{name: ‘address’, type: ‘string’, title: ‘Address’}
```

[Review Schema]:

```
export default {

name: ‘review’,

type: ‘document’,

title: ’Review’,

fields: [

{name: ‘product’, title: ‘Product’, type: ‘reference’,

To: [ {type: ‘product’}],},

{name: ‘rating’, title: ‘Rating’, type: ‘string’},

{name: ‘reviewText’, title: ‘Review Text’, type: ‘string’}
```