Hackathon Day 2: Planning the Technical Foundation (Nike Website Edition)

Recap of Day 1: Business Focus (Nike Website Edition)

1. Business Goals Defined:

Create a dynamic e-commerce platform for Nike products.

- **Enable users to browse Nike product categories, view detailed product descriptions, and purchase items online.**
- **Provide an intuitive user experience with fast load times and a responsive design.**

2. Data Schema Drafted:

- Entities:
    - **Products:** Includes fields like name, price, description, images, sizes, and stock levels.
    - **Users:** User information for order tracking and account management.
    - **Orders:** Details like product IDs, quantity, total cost, and payment status.

3. Single Focus:

Concentrate on user-centric features such as browsing, adding items to the cart, and completing orders efficiently.

Day 2 Goal

To transition the Nike Website project from conceptualization to technical implementation by defining requirements, designing architecture, planning APIs, and documenting the technical aspects.

Day 2 Activities for the Nike Website

1. Define Technical Requirements

- Frontend Requirements:

Pages Required:

- Home Page: Showcase Nike's featured products.
- Product Listing Page: Display categories like shoes, clothing, and accessories.
- Product Details Page: Show individual product details (name, price, images, stock, etc.).
- Cart and Checkout Pages: Enable adding items to the cart and completing purchases.

User Experience:

- Mobile-first, responsive design.
- Integration of animations for transitions (e.g., hover effects on products).
- Seamless navigation between pages.

Sanity CMS as Backend:

- Product Management: Use Sanity to manage product data, categories, and images.
- Order Management: Store order records and user details in Sanity for tracking.

- **Third-Party APIs:**
  - Payment Gateway: For secure transactions.
  - Shipment Tracking: Display real-time order status.

2. Design System Architecture

Frontend:

- Built with **Next.js** and styled using **Tailwind CSS**.
- Fetch product data dynamically using Sanity APIs.

Backend:

- Sanity CMS for managing all content.
- Node.js server for API requests and integration with payment/shipping providers.

Third-Party APIs:

- **Payment Gateway API: Handle transactions.**
- **Shipment API: Provide order tracking information.**

System Workflow Diagram:

- **User Signup -> Product Browsing (via Sanity) -> Add to Cart -> Place Order (Sanity & Payment APIs) -> Shipment Tracking (Third-Party API).**

3. Plan API Requirements:

API Endpoints for the Nike Website:

Fetch Products

- **Endpoint:** `/products`
- **Method:** GET
- **Description:** Retrieve all Nike products.
- **Response Example:**

[{ "id": 1, "name": "Air Max 90", "price": 120, "image": "airmax90.jpg", "stock": 10 }]

Create Order

- **Endpoint:** `/orders`
- **Method:** POST
- **Description:** Save a user's order details.

Payload Example:

```
{
  "userId": 123,
  "products": [
    { "id": 1, "quantity": 2 },
    { "id": 2, "quantity": 1 }
  ],
  "total": 240
}
```

Response Example:

```
{ "orderId": 456, "status": "Success" }
```

Track Shipment

- **Endpoint:** `/shipment`
- **Method:** GET
- **Description:** Fetch shipment details for an order.
- **Response Example:**
  json
  CopyEdit
  ```
  { "orderId": 456, "status": "Shipped", "ETA": "3 days" }
  ```

Presented By:

Rimsha Mukhtar

Slot: Saturday 2 to 5