Name: Muhammad Zahid Roll No: 00476916

HACKATHON DAY 02:

PLANNING THE TECHNICAL FOUNDATION:

Technical Planning for eCommerce Furniture Marketplace

Technical Foundation Overview

Objective:

The focus for Day 2 is transitioning from business goals to technical implementation. This involves defining the technical requirements for the marketplace, ensuring it aligns with the goals of the **Furniture eCommerce Marketplace**. Key areas include building a user-friendly frontend, setting up **Sanity CMS** as the backend, and integrating essential third-party APIs for seamless functionality.

Key Technologies

Frontend:

- **Next.js**: Framework for building fast, server-side rendered and static web applications.
- **TypeScript**: Ensures type safety and scalability in the codebase.
- **TailwindCSS**: For responsive and modern UI design.

Backend:

- **Sanity CMS**: Manages product data, customer details, and order records with customized schemas.
- **GROQ Queries**: Fetches structured content dynamically for the frontend.

Third-Party Integrations:

- **Payment Gateways**: Stripe, PayPal, or Razorpay for secure payments.
- **Shipment Tracking APIs**: Shippo or EasyPost for tracking deliveries.
- **Communication APIs**: Twilio or SendGrid for order confirmations and notifications.

Technical Architecture

System Overview

The architecture of the **Furniture eCommerce Marketplace** is designed to seamlessly integrate frontend user interactions with backend content management, third-party services, and secure payment processing. It ensures a robust, scalable, and efficient platform for purchasing furniture products.

System Components

Frontend (Next.js)

- **Description:**
    - The user-facing layer that offers an intuitive interface for product browsing, order placement, and tracking.
- **Key Features:**
    - Responsive design for mobile and desktop.
    - Pages: Home, Product Listing, Product Details, Cart, Checkout, and Order Confirmation.
    - Fetches and displays data dynamically via APIs.

Sanity CMS (Backend)

- **Description:**
    - A headless CMS acting as the database for managing product details, user records, and orders.
- **Key Features:**
    - Schemas for products, orders, customers, and payments.
    - Real-time updates for content.
    - Stores all critical marketplace data.

Third-Party APIs:

- **Shipment Tracking API:** Integrates with logistic partners to fetch real-time shipping updates.
- **Payment Gateway API (e.g., Stripe/PayPal):** Handles secure transactions and ensures compliance with financial regulations.
- **Authentication Services (e.g., Firebase/Auth0):** Manages user authentication and ensures secure login and registration.
- **Notification APIs (e.g., Twilio/SendGrid):** Sends real-time order confirmations and shipment updates to users.

System Workflow

1. User Browsing Products:

- User visits the frontend (Next.js).
- Frontend sends API requests to Sanity CMS to fetch product details.
- Products are dynamically displayed with filters and search options.

2. User Registration and Login:

- User enters details on the frontend.
- Data is sent to an authentication service for validation and storage in Sanity CMS.
- Success message or error handling displayed on the frontend.

3. Placing an Order:

- User selects items and adds them to the cart.
- At checkout, order details (products, customer info) are sent to Sanity CMS.
- Payment API processes payment and sends confirmation to both user and CMS.

4. Shipment Tracking:

- Once the order is shipped, the frontend requests status updates from the Shipment Tracking API.
- Real-time status is displayed on the user dashboard.

5. Payment Processing:

- User provides payment details via the frontend.
- Payment API handles the transaction securely.
- Confirmation is sent to the user, and data is stored in Sanity CMS.

API Endpoints

User Management

- **POST /api/auth/register** - Register a new user.
- **POST /api/auth/login** - User login.
- **GET /api/users/profile** - Fetch user profile (requires authentication).
- **PUT /api/users/update** - Update user details.

Product Management

- **GET /api/products** - List all available products.
- **GET /api/products/:id** - Fetch product details by ID.
- **POST /api/products** - Add a new product (requires seller role).
- **PUT /api/products/:id** - Update product details (requires seller role).
- **DELETE /api/products/:id** - Delete a product (requires seller role).

Order Management

- **POST /api/orders** - Create a new order.
- **GET /api/orders** - List all orders for the authenticated user.

- **GET /api/orders/:id** - Fetch details of a specific order.

Delivery Zone

- **POST /api/delivery-zone** - Add delivery zone details.

Payment Management

- **POST /api/payments** - Initiate a payment for an order.
- **GET /api/payments/status** - Fetch payment status.

Shipment Management

- **POST /api/shipments** - Create a new shipment.
- **GET /api/shipments/track** - Track shipment status.

Sanity Schemas for Key Data Entities

1. Users Table

Column Name	Data Type	Description
user_id	INT	Primary key, unique identifier for the user.
username	VARCHAR	Full name of the user.
email	VARCHAR	Email address of the user.
password_hash	VARCHAR	Encrypted password for security.
role	ENUM	Role of the user (admin, seller, customer).
order_ids	TEXT	List of order IDs associated with the user.

2. Products Table

Column Name	Data Type	Description
product_id	INT	Primary key, unique identifier for the product.
name	VARCHAR	Name of the product.
price	DECIMAL	Price of the product.
stock	INT	Availability of the product.
description	TEXT	Detailed description of the product.
image_url	VARCHAR	URL of the product image.

3. Orders Table

Column Name	Data Type	Description
order_id	INT	Primary key, unique identifier for the order.
user_id	INT	Foreign key referencing Users table.
product_ids	TEXT	List of product IDs in the order.
total_price	DECIMAL	Total cost of the order.
status	ENUM	Status of the order (pending, completed, canceled).

4. Payments Table

Column Name	Data Type	Description
payment_id	INT	Primary key, unique identifier for the payment.
order_id	INT	Foreign key referencing Orders table.
amount	DECIMAL	Payment amount.
status	ENUM	Payment status (successful, failed, pending).

This technical plan lays the foundation for developing a robust **Furniture eCommerce Marketplace**. The next steps involve implementing these structures into a working prototype.

5. Delivery Zones Table

This table stores information about the different delivery zones where furniture products can be shipped.

Column Name	Data Type	Description
zone_id	INT (PK)	Unique identifier for the delivery zone.
zone_name	VARCHAR	Name of the delivery zone (e.g., city, region, postal code).
zone_description	TEXT	Details about the zone (e.g., coverage area, delivery conditions).
shipping_cost	DECIMAL(10,2)	Base cost for shipping to this zone.
rental_fee	DECIMAL(10,2)	Additional fee for rented furniture in this zone.
delivery_time_estimate	VARCHAR	Estimated delivery time for this zone (e.g., "2-5 business days").

Column Name	Data Type	Description
created_at	TIMESTAMP	Date and time when the delivery zone was added.
updated_at	TIMESTAMP	Date and time when the delivery zone details were last updated.

---

6. Shipments Table

This table stores details about furniture shipments, including shipping status, carrier, and delivery dates.

Column Name	Data Type	Description
shipment_id	INT (PK)	Unique identifier for the shipment.
order_id	INT (FK)	References order_id in the **Orders** table.
delivery_zone_id	INT (FK)	References zone_id in the **Delivery Zones** table.
shipment_date	TIMESTAMP	Date and time when the shipment was initiated.
estimated_delivery_date	TIMESTAMP	Estimated date for the shipment to arrive.
actual_delivery_date	TIMESTAMP	Actual delivery date, if applicable.
carrier	VARCHAR	Name of the shipping carrier (e.g., FedEx, UPS, Local Furniture Movers).
tracking_number	VARCHAR	Tracking number assigned by the carrier.
status	ENUM('pending', 'in-transit', 'delivered', 'failed', 'returned')	Current shipment status.
shipping_cost	DECIMAL(10,2)	Cost of the shipment (if applicable).
handling_fee	DECIMAL(10,2)	Additional handling or setup fees for large furniture.
installation_required	BOOLEAN	Indicates if the shipment includes installation services.
delivery_notes	TEXT	Special instructions or notes for the delivery team.

ENTITY-RELATIONSHIP DIAGRAM : FURNITURE STORE