DAY 2:

PLANNING THE TECHNICAL FOUNDATION

1. Define Technical Requirements

a) Frontend Requirements:

i. Design and User Experience:

- **Simple and Clean Design:** The website should look modern and easy to navigate.
- **Mobile-Friendly:** The website must work perfectly on phones, tablets, and computers.
- **Smooth Navigation:** Easy-to-use menus and buttons for quick access to different pages.
- **Dark and Light Mode:** Offer theme customization so users can switch between dark and light modes for better accessibility and personalization.
- **Interactive Hero Section:**
    - Banners with animations showcasing sales and trending products.
    - Hover effects that reveal hidden deals or discount deals.
- **Hover Effects:** Use soft shadow and frosted glass efforts for buttons and products cards for a modern look.
- **Dashboard:** When user login, the dashboard adapts to show purchased history.
- **Animated Product Cards:** When hovering over products, cards, flip or expand to show quick details like price, stock, and “Add to Cart” button.

ii. Essential pages:

a) Home pages:

- Banners showing deals and new products.
- Recommended products based on what users like.
- Trending products and best sellers.

b) Products Listing Pages:

- Filters to search by price, brand, or category.
- Sort options like “Low to High” price or “Newest First”.
- Quick view to see product details without opening a new page.

c) Product Detail Page:

- Clear images.
- Detailed descriptions, reviews, and ratings.
- Show if the product is in stock or an sale.

d) Cart and WishList:

- View and edit cart items easily.
- Save favorite products in the wishlist.

e) CheckOut Page:

- Easy step-by-step checkout process.
- Options to checkout as an users (with an account).
- Multiple payments options like cards, wallets, or cash on delivery.

f) Order Tarcking and Confirmation:

- Track orders in real-Time.
- Email/SMS updates for order status.

g) Lognin and Signup:

- Easy account creation with social options (Google, Facebook).
- Forgot passwords and reset options.

- Secure login with Email.

h) <u>User Dashboard:</u>

- View past orders and saved items.
- Manage address and payment methods.
- Subscription to newsletters.

i) <u>Contact and Support:</u>

- Live chat for instant help.
- Contact form for common questions.
- FAQ page for common questions.

b) Backend (Sanity CMS):

a. For Product Management:

- Add and update products (name, prices, stock, image).
- Organize products into categories.

b. For Order Management:

- Keep track of all orders and their status (Pending, Shipped, Delivered).
- Automatically reduce stock when products are solid.

c. For Customer Data:

- Store customer details securely.
- Manage newsletter subscriptions.

d. **For Content Control:**

- Update banners, deals, and blog posts without coding.
- Manage flash sales with start and end seasons.

c) Third-Party API Integrations:

a. **<u>Payment Gateway:</u>**

- Connect with stripe, PayPal, and Cash on delivery for secure payments.

b. **<u>Shipping API:</u>**

- Show real-time delivery updates with shipping companies.
- Calculate delivery charges based on location

c. **<u>Email/SMS Notifications:</u>**

- Send automatic updates via email/SMS for orders and promotions.

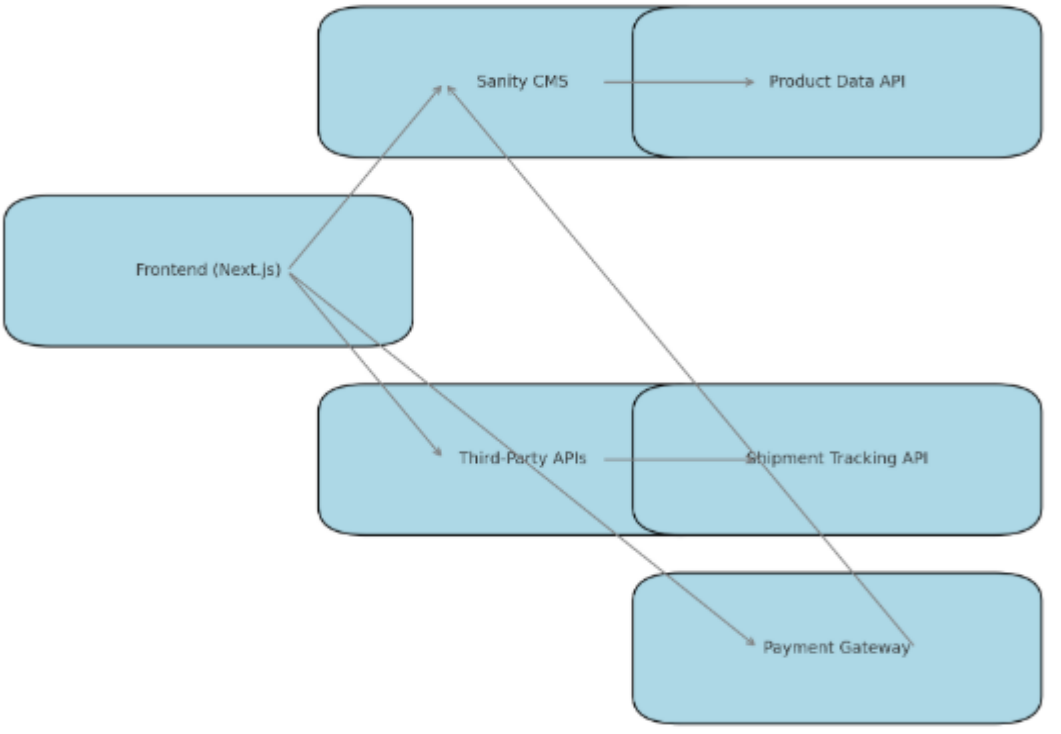
d. **<u>Smart Recommendations:</u>**

- Suggest products based on what customers like or view.

e. **<u>Google Maps API:</u>**

- Auto-fill address fields during checkout.
- Verify delivery zones.

2. Design System Architecture



Detailed workflow Explanation

Step 1: User Registration and Login:

- A user visits the marketplace and clicks **Sign Up**.
- The user fills in details (name, email, password).
- Data is stored in **Sanity CMS** under the user database.
- A confirmation email is sent via a third-party email API.

Step 2: Browsing Products:

- The user browses product categories.
- The frontend makes a request to the Product Data **API**.
- **Sanity CMS** sends product data (images, prices, descriptions) to the frontend.
- Products are displayed in a user-friendly layout.

Step 3: Adding to Cart and Placing an Order

- On clicking **Checkout**, the system collects shipping and payment details.
- Payment is processed securely through the **Payment Gateway API**.
- Once payment is successful, order details are saved in **Sanity CMS**.
- An **order confirmation email/SMS** is sent to the user.

Step 4: Shipment and Delivery Tracking

- The order is prepared for shipment.
- Shipment details are sent to the **Third-Party Shipment Tracking API**.
- Users can track their orders in real-time on the **Order Tracking Page**.
- Once delivered, the shipment status is updated in **Sanity CMS**.

Complete Data Flow Overvie

1. **Frontend (Next.js)** → Sends a request for product data.
2. **Sanity CMS** → Provides product information via API.
3. **Frontend (Next.js)** → Displays products.

4. **User** → Places an order.
5. **Payment Gateway API** → Processes payment.
6. **Sanity CMS** → Stores order details.
7. **Shipment Tracking API** → Updates delivery status.
8. **Frontend (Next.js)** → Shows live tracking to users.

3. Plan API Requirements

1. Product Management Endpoints:

a. Get All Products

- **Endpoint:** /products
- **Method:** GET
- **Description:** Fetch all available products from Sanity CMS.
- **Response Example:**

```json
[
  {
    "id": "prod_001",
    "name": "Wireless Headphones",
    "price": 150,
    "stock": 25,
    "image": "https://example.com/images/headphones.jpg",
    "category": "Electronics",
    "description": "High-quality wireless headphones with noise cancellation."
  }
]
```

b. Get Product by ID

- **Endpoint:** /products/{id}
- **Method:** GET
- **Description:** Fetch detailed information of a specific product.
- **Response Example:**

```json
{
  "id": "prod_001",
  "name": "Wireless Headphones",
  "price": 150,
  "stock": 25,
  "image": "https://example.com/images/headphones.jpg",
  "category": "Electronics",
  "description": "High-quality wireless headphones with noise cancellation."
}
```

c. Search Products

- **Endpoint:** /products/search
- **Method:** GET
- **Description:** Search for products by keyword or category.
- **Query Parameters:** ?query=electronics
- **Response Example:**

```json
[
  {
    "id": "prod_002",
    "name": "Smartphone",
    "price": 800,
    "category": "Electronics"
  }
]
```

2. User Authentication Endpoints

a. User Registration

- **Endpoint:** `/auth/register`
- **Method:** `POST`
- **Description:** Register a new user account.
- **Payload:**

```json
{
  "name": "John Doe",
  "email": "john@example.com",
  "password": "securePassword123"
}
```

- **Response Example:**

```json
{
  "userId": "user_001",
  "status": "Registration Successful"
}
```

b. User Login

- **Endpoint:** `/auth/login`
- **Method:** `POST`
- **Description:** Authenticate user and generate a login token.
- **Payload:**

```json
{
  "email": "john@example.com",
  "password": "securePassword123"
}
```

- **Response Example:**

```json
{
  "token": "jwt-token",
  "userId": "user_001",
  "status": "Login Successful"
}
```

3. Cart Management Endpoints

a. Add Item to Cart

- **Endpoint:** `/cart/add`
- **Method:** `POST`
- **Description:** Add a product to the user's cart.
- **Payload:**

```json
{
  "userId": "user_001",
  "productId": "prod_001",
  "quantity": 2
}
```

- **Response Example:**

```json
{
  "cartId": "cart_001",
  "status": "Product added to cart"
}
```

b. View Cart Items

- **Endpoint:** `/cart`
- **Method:** `GET`
- **Description:** View all items in the user's cart.
- **Response Example:**

```json
[
  {
    "productId": "prod_001",
    "name": "Wireless Headphones",
    "quantity": 2,
    "price": 150
  }
]
```

4. Order Management Endpoints

a. Create New Order

- **Endpoint:** `/orders`
- **Method:** `POST`
- **Description:** Place a new order and save it in **Sanity CMS**.
- **Payload:**

```json
{
  "userId": "user_001",
  "items": [
    {
      "productId": "prod_001",
      "quantity": 2
    }
  ],
  "totalPrice": 300,
  "paymentStatus": "Paid",
  "shippingAddress": "123 Main Street, City, Country"
}
```

- **Response Example:**

```json
{
  "orderId": "order_001",
  "status": "Order Placed Successfully"
}
```

b. Get User Orders

- **Endpoint:** `/orders/user/{userId}`
- **Method:** `GET`
- **Description:** Retrieve all past orders for a user.
- **Response Example:**

```json
[
  {
    "orderId": "order_001",
    "totalPrice": 300,
    "status": "Shipped"
  }
]
```

5. Shipment Tracking Endpoint

a. Track Shipment

- **Endpoint:** `/shipment/{orderId}`
- **Method:** `GET`
- **Description:** Get real-time shipment updates using a third-party API.
- **Response Example:**

```json
{
  "orderId": "order_001",
  "status": "Order Placed Successfully"
}
```

6. Payment Processing Endpoints

a. Initiate Payment

- **Endpoint:** `/payment/initiate`
- **Method:** `POST`
- **Description:** Start payment processing via the payment gateway.
- **Payload:**

```json
{
  "orderId": "order_001",
  "amount": 300,
  "paymentMethod": "Credit Card"
}
```

- **Response Example:**

```json
{
  "paymentId": "pay_123",
  "status": "Payment Initiated",
  "redirectUrl": "https://paymentgateway.com/checkout"
}
```

b. Confirm Payment

- **Endpoint:** `/payment/confirm`
- **Method:** `POST`
- **Description:** Confirm if the payment was successful.
- **Payload:**

```json
{
  "paymentId": "pay_123",
  "orderId": "order_001"
}
```

- **Response Example:**

```json
{
  "status": "Payment Successful"
}
```

Summary of API Endpoints

Functionality	Endpoint	Method	Description
Get All Products	`/products`	`GET`	Fetch all products from Sanity CMS.
User Registration	`/auth/register`	`POST`	Register a new user.
User Login	`/auth/login`	`POST`	Authenticate user and generate token.
Add to Cart	`/cart/add`	`POST`	Add item to the cart.
Place Order	`/orders`	`POST`	Create a new order.
Track Shipment	`/shipment/{orderId}`	`GET`	Get shipment status.
Payment Initiation	`/payment/initiate`	`POST`	Start payment process.
Payment Confirmation	`/payment/confirm`	`POST`	Confirm payment status.