Technical Overview of Our Furniture E-Commerce Website

This document outlines the essential components and features of our furniture e-commerce website. Each section provides a detailed explanation of the goals and requirements specific to our project.

Frontend Overview

User-Friendly Interface

Our website will provide an intuitive and visually appealing interface that simplifies the shopping experience. Key aspects include:

- Easy navigation through categories and collections.
- Quick access to product details and customer reviews.

Responsive Design

To ensure accessibility across all devices, the website will feature:

- Adaptive layouts that adjust seamlessly to mobile, tablet, and desktop screens.
- Optimized loading times for a smooth user experience on any platform.

Essential Pages

- **Home Page:**
  - Display featured collections, promotions, and a welcoming introduction.
  - Highlight trending products and seasonal sales.
- **Product Listing Page:**
  - Provide filters for price, category, and availability.
  - Include sorting options such as "Newest First" or "Price: Low to High."
- **Product Details Page:**
  - Showcase high-quality images and detailed descriptions.
  - Include customer reviews and related product recommendations.
- **Cart Page:**
  - Allow users to view selected items, adjust quantities, or remove products.
  - Display the total price, including applicable taxes and discounts.
- **Checkout Page:**
  - Simplify the checkout process with clear steps for billing and shipping.
  - Integrate secure payment options.
- **Order Confirmation Page:**
  - Provide a summary of the completed order.
  - Include an estimated delivery date and tracking information.

- **Blog Page:**
  - Feature articles on furniture care, interior design tips, and seasonal trends.
  - Encourage user engagement with helpful and shareable content.
- **Blog Detail Page:**
  - Provide a detailed view of each blog post with high-quality images.
  - Include embedded links to related products for seamless shopping.

Backend Architecture with Sanity CMS

Product Data Management

Sanity CMS will act as the backbone of our website, handling:

- Storage of product information, including names, prices, descriptions, and images.
- Categorization of products for better organization and searchability.

Customer Information Management

The system will securely store customer profiles, including:

- Contact information and order history.
- Preferences for personalized recommendations.

Order Records and Tracking

Sanity CMS will maintain detailed logs of:

- Current and past orders.
- Order statuses, from processing to delivery.

Schema Design

Custom schemas will be developed to align data organization with the specific needs of our furniture e-commerce platform, ensuring scalability and flexibility.

Authentication System

Importance of Authentication

Implementing an authentication system will:

- Link cart items, order history, and saved preferences to user accounts.
- Prevent unauthorized access to cart and checkout functionalities.
- Provide personalized experiences and promotional offers through verified user accounts.

Features of the Authentication System

- **Login and Registration:**
  - Secure login via email, phone, or third-party OAuth (Google, Facebook).
  - Simple and intuitive registration process.
- **Email and Phone Verification:**
  - Validate user accounts with OTP (One-Time Password) verification.
  - Ensure authentic customer interactions.
- **Session Management:**
  - Remember user sessions for a seamless shopping experience.
  - Use tokens (like JWT) for secure user identification.
- **Add-to-Cart and Order Flow:**
  - Require login before users can add items to the cart or proceed to checkout.
  - Ensure cart items are saved in the database and linked to the user account.
- **User Roles and Permissions:**
  - Differentiate between regular customers and admin users.
  - Allow admins to manage inventory, orders, and user accounts.

Integration of Third-Party APIs

Payment Gateways

We will integrate secure payment solutions, such as:

- Stripe or PayPal for smooth and reliable transactions.
- Support for multiple payment methods, including credit cards and digital wallets.

Shipment Tracking

To enhance user trust and satisfaction, the website will offer:

- Real-time updates on shipping statuses.
- Automated notifications for delivery progress.

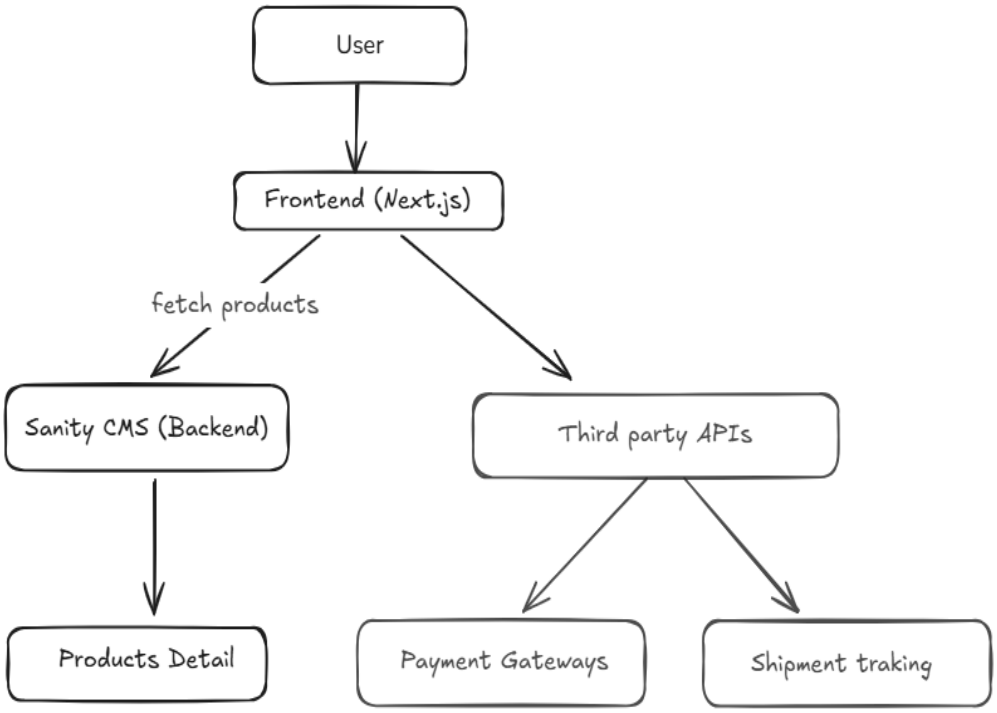
Additional Services

Other planned integrations include:

- Currency converters for international customers.
- Tax calculators to display accurate pricing.
- Email services for order confirmations and promotional campaigns.

System Architecture Overview

System Architecture Diagram



High-Level Architecture

The system architecture demonstrates the interaction between the frontend, backend, and third-party services. Below is the data flow:

- **Frontend (Next.js):**
  - Handles user interface and requests for data from the backend.
- **Sanity CMS:**
  - Acts as the database for product data, customer details, and order records.
- **Third-Party APIs:**
  - Provide functionalities like shipment tracking and payment processing.

Data Flow Example

1. A user visits the marketplace frontend to browse products.
2. The frontend makes a request to the Product Data API (powered by Sanity CMS) to fetch product listings and details, which are displayed dynamically on the site.
3. When the user places an order, the order details are sent to Sanity CMS via an API request, where the order is recorded.
4. Shipment tracking information is fetched through a third-party API and displayed to the user in real time.
5. Payment details are securely processed through the payment gateway, and a confirmation is sent back to the user and recorded in Sanity CMS.

API Requirements

1. Endpoint Name: /products

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

```
{
  "id": 1,
  "name": "Wooden Chair",
  "price": 3000,
  "stock": 50,
  "image": "url-to-image"
}
```

2. Endpoint Name: /orders

- **Method:** `POST`
- **Description:** Create a new order in Sanity CMS.
- **Payload:**

```
{
  "customerId": 123,
  "products": [
          { "productId": 1,      "quantity": 2 },
          { "productId": 2,      "quantity": 1 }
  ],
  "totalPrice": 9000,
  "paymentStatus": "Paid"
}
```

- **Response Example:**

```
{
        "orderId": 789,
        "status": "Order Confirmed",
         "estimatedDelivery": "2025-01-20"
}
```

3. Endpoint Name: /shipment

- **Method:** `GET`
- **Description:** Track order status via a third-party API.
- **Response Example:**

```
{
  "shipmentId": "SHIP123",
  "orderId": 789,
  "status": "In Transit",
  "expectedDelivery": "2025-01-20"
}
```

API Endpoints:

Endpoint	Method	Purpose	Response Example
`/products`	GET	Fetches all product details	`{ "id": 1, "name": "Product A", "price": 100 }`
`/products/:id`	GET	Fetches details of a product	`{ "id": 1, "name": "Product A", "price": 100, "description": "Details" }`
`/add-to-cart`	POST	Adds a product to the cart	`{ "success": true, "cart": [{ "id": 1, "quantity": 2 }] }`
`/cart`	GET	Fetches the current cart items	`{ "cart": [{ "id": 1, "name": "Product A", "quantity": 2 }] }`
`/checkout`	POST	Processes the order	`{ "status": "success", "orderId": 123 }`
`/blogs`	GET	Fetches all blog posts	`[ { "id": 1, "title": "Blog Post 1", "author": "Admin" } ]`
`/blogs/:id`	GET	Fetches details of a blog post	`{ "id": 1, "title": "Blog Post 1", "content": "Full content here" }`
`/order-status/:id`	GET	Fetches the status of an order	`{ "orderId": 123, "status": "Shipped" }`

Sanity Schema Example

Blog Schema:

```
export default {
  name: 'blog',
  title: “Blog
  type: 'document',
  fields: [
    { name: 'title', type: 'string', title: 'Blog Title' },
    { name: 'content', type: 'text', title: 'Blog Content' },
    { name: 'author', type: 'string', title: 'Author' },
    { name: 'publishedAt', type: 'datetime', title: 'Published At' },
  ],
};
```

Product Schema:

```
export default {
  name: 'product',   title: “Product”,     type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Product Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'stock', type: 'number', title: 'Stock Level' },
    { name: 'description', type: 'text', title: 'Description' },
    { name: 'imageUrl', type: 'url', title: 'Image URL' },
  ],
};
```

Conclusion

This documentation serves as a comprehensive guide for building a scalable and user-friendly eCommerce marketplace. By integrating modern technologies like React, Next.js, Sanity CMS and Third party APIs, the platform ensures efficient content management and seamless user interactions. The outlined workflows, API specifications, and schemas provide a solid foundation for streamlined development and deployment.