Hackathon 03 (Day 2): Marketplace Technical Foundation

1. Frontend Requirements

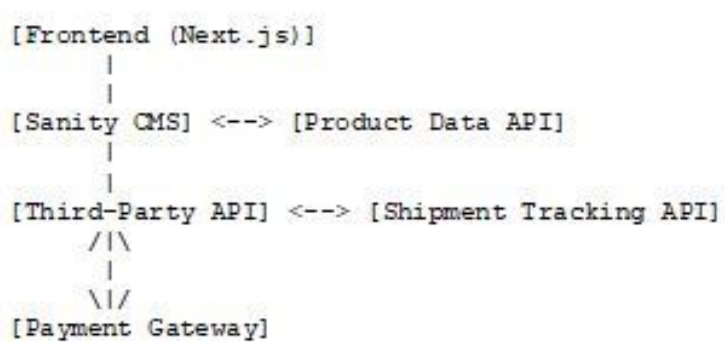
1. **User-Friendly Interface**:
   - A seamless and intuitive platform for browsing, selecting, and purchasing furniture.
2. **Responsive Design**:
   - Compatibility with mobile, tablet, and desktop devices to ensure an optimized experience for all users.

3. **Pages to include:**

- Home
- About
- Contact
- Products
- Dynamic Product Details Page
- Blogs
- Dynamic Single Blog Page
- Wishlist
- Login/Signup
- Cart
- Checkout
- Order Confirmation
- Separate Tracking

2. System Architecture Design

Here's an example of how components interact in my system:



Data Flow Example:

1. User browses the marketplace.
2. The frontend sends a request to the **Sanity CMS** to fetch product listings.
3. User places an order, and the order details are sent to **Sanity CMS** via an API.

4. Shipment updates are fetched from the **Shipment Tracking API** and displayed in real-time.
5. Payment details are processed securely through the **Payment Gateway**, and a confirmation is sent to the user and recorded in **Sanity CMS**.

3. Key Workflows

User Registration:

1. User signs up on the platform.
2. The user data is stored in **Sanity CMS**.
3. Confirmation is sent to the user.

Product Browsing:

1. User views categories of products.
2. **Sanity CMS** API fetches product data.
3. Products are dynamically displayed on the frontend.

Order Placement:

1. User adds items to the cart.
2. Proceeds to checkout, entering payment and shipping information.
3. Order details are saved in **Sanity CMS**.

Shipment Tracking:

1. Order status updates are fetched via the **Shipment Tracking API**.
2. Updates are shown to the user on the frontend.

4. API Requirements

Here are the API endpoints that will support my marketplace:

Endpoint Name	Method	Description	Response Example
`/products`	GET	Fetch all available products from Sanity CMS	`{ "id": 1, "name": "Product A", "price": 100 }`
`/orders`	POST	Create a new order in Sanity CMS	`{ "orderId": 123, "status": "Confirmed" }`
`/shipment`	GET	Track order status via third-party API	`{ "orderId": 123, "status": "In Transit", "ETA": "3 hours" }`
`/payment`	POST	Process payment via the payment gateway	`{ "paymentId": 456, "status": "Success" }`

5. Sanity Schema Example

Here's an example schema for the **Product** and **Order**:

Product Schema:

```
export default {
  name: 'product',
  type: 'document',
  fields: [
    { name: 'id', type: 'string', title: 'Product ID' },
    { name: 'images', type: 'array', of: [{ type: 'image'}] title: 'Produc
Images' },
    { 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: 'Product Description' }
  ]
};
```

Order Schema:

```
export default {
  name: 'order',
  type: 'document',
  fields: [
    { name: 'Orderid', type: 'string', title: 'Order ID' },
    { name: 'customerName', type: 'string', title: 'Customer Name' },
    { name: 'email', type: 'string', title: 'Email Address' },
    { name: 'productDetails', type: 'array', title: 'Product Details' },
    { name: 'totalAmount', type: 'number', title: 'Total Amount' },
    { name: 'orderStatus', type: 'string', title: 'Order Status' }
  ]
};
```

6. Technical Documentation

- **System Architecture Overview**: Diagram and description of how frontend, backend (Sanity CMS), and third-party APIs work together.
- **Key Workflows**: A detailed description of the user registration, product browsing, order placement, and shipment tracking.
- **API Specification Document**: Table format for all the required API endpoints.
- **Data Schema Design**: Define entities such as products, orders showing how data is structured.