HACKATHON 3

DAY 2 PLANNING THE TECHNICAL

FOUNDATION

Marketplace Technical Foundation - [Hekto: Building the Future of Furniture Shopping]

Table of Contents

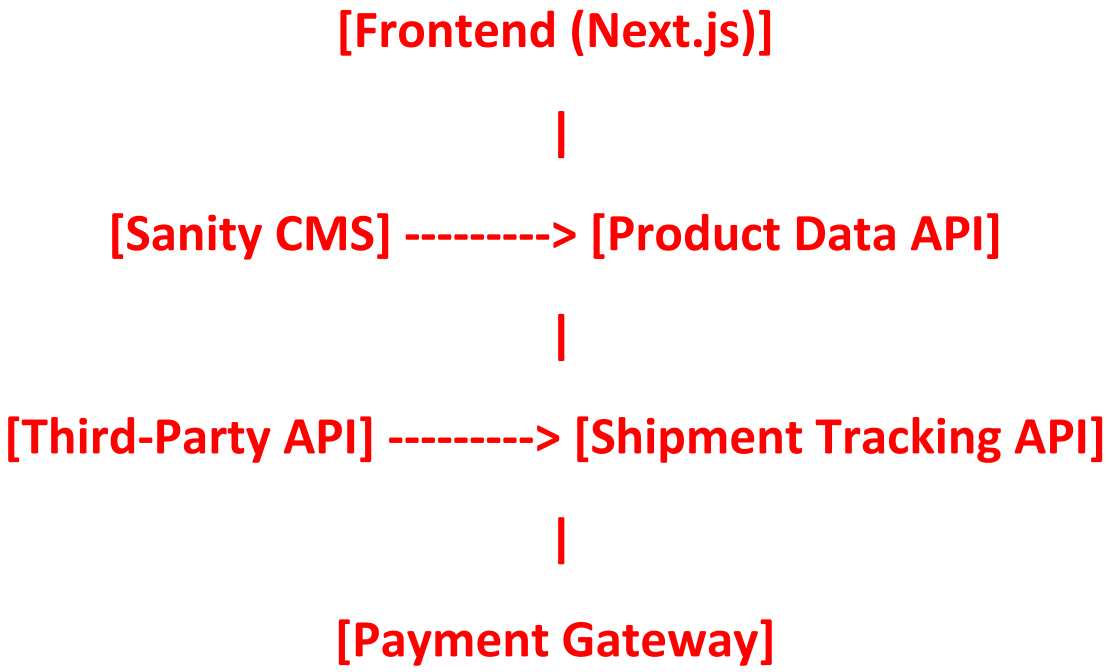
1. Overview

The Hekto eCommerce website is a modern platform designed to provide a seamless shopping experience for users. It focuses on delivering high performance, intuitive navigation, and scalable architecture. The platform leverages Next.js for frontend development, Tailwind CSS for styling, and Sanity CMS for backend content management.

2. System Architecture

- **The system architecture is based on a microservices approach to ensure scalability and modularity. The key components include:**
- **Frontend: Built with Next.js for server-side rendering (SSR) and static site generation (SSG).**
- **Backend: Sanity CMS for dynamic content management and product catalog.**
- **APIs: REST/GraphQL APIs to manage interactions between the frontend and backend.**
- **Database: Sanity's content lake and additional relational/non-relational databases as required.**
- **Third-Party Integrations: Payment gateways, shipping services, and analytics tools.**

High-Level Diagram



Component Interactions

1.Frontend (Next.js):

Provides a dynamic and responsive user interface for browsing products, managing the shopping cart, and completing purchases.

2.Sanity CMS:

Manages product data, customer information, order details, and content. It provides an API for accessing product data and storing order information.

3.Third-Party APIs:

Handle external functionalities such as secure payment processing and real-time shipment tracking.

4.Payment Gateway:

Responsible for processing user payments securely and providing transaction con rmation.

3- Technical Requirements:

Frontend Requirements

- User-Friendly Interface:
- Browsing and ltering categories: Sofas, chairs, tables, etc.

- Dynamic product listings with ltering options based on
- price, category, and availability.
- Pages to Implement:

1.Home Page:

- Overview of products, promotions, and featured items.
- Includes a banner for special o ers and seasonal promotions.
- Display of trending or best-selling products.

2. Product Listing Page:

- A catalog of products with ltering options such as categories, price ranges, and ratings.
- Sorting options like "Price: Low to High," "New Arrivals," "Best Sellers," etc.
- Users can view and browse products to select items for purchase.

3. Product Details Page:

- Detailed view of a single product, including:
- Price, description, and speci cations.
- High-quality images or videos of the product.
- Availability (in stock, out of stock).
- Option to select quantity and size (if applicable).
- Add to Cart button.

4. Cart Page:

- Allows users to view and manage their cart:
- List of selected products, including quantities and prices.
- Options to update quantities or remove products.
- Display of estimated total price and taxes.
- Option to proceed to checkout.

5. Checkout Page:

- User enters payment information and proceeds with the purchase:
- Address form (shipping address).
- Payment options (credit card, PayPal, etc.).
- Order summary with a nal price breakdown.

- Place Order button.

6. Order Con rmation Page:

- Conformation of the order with transaction details:
- Order number, con rmation message.
- Details of the purchased items and shipping address.
- Estimated delivery date.
- A thank-you note and options to continue shopping or view the order status.

7. Blog Page:

- Articles, product tips, and updates on promotions or new arrivals.
- Engaging content related to the products or industry.
- Option to share or comment on blog posts.

8. Wishlist Page:

- Allows users to save products they are interested in purchasing later.
- Option to move items to the cart or remove them from the wishlist.
- A helpful reminder for products that are on sale or about to run out of stock.
- Users can keep track of products they like but are not yet ready to purchase.

9. Shop Page:

- Displays all available products for immediate purchase.
- Products can be ltered or sorted by categories, price, popularity, etc.
- Users can browse through di erent products and add them directly to their cart.

10. About Page:

- Information about the store, company mission, and values.
- Brief history or background of the business.
- Contact information or links to social media profiles

11. Contact Page:

- Contact form for inquiries, returns, or feedback.
- Store address, phone number, and email.
- Embedded map for the store location (if applicable).
- Social media links for additional contact methods.

- Responsive Design: Ensures compatibility with mobile, tablet, and desktop devices.

Backend Requirements (Sanity CMS)

Data Management:

- **Products:** Store product information such as name, description, price, stock level, category, and images.
- **Orders:** Manage customer details, ordered products, status, and timestamps.
- **Categories**: Organize products into categories (e.g., sofas, chairs, tables).

Schema Design:

Product Schema: Includes elds for product attributes (name, description, price, category, stock, and images).

Order Schema: Manages customer and order details such as status, items, and order date.

Third-Party API Integrations:

1.Payment Gateway API:

- Secure payment processing for transactions.
- Sends con rmation to both users and the backend after payment is successful.

2.Shipment Tracking API:

- Provides real-time delivery updates.
- Integrates with the frontend to display shipment progress to the user.

4. Work ows:

Key Work ows

1.User Registration:

- User provides details (name, email, password).
- Data is stored in Sanity CMS.
- Conformation email is sent to the user.

2.Product Browsing:

- Users browse products on the frontend.
- Product data is fetched from the Sanity CMS API.

- Products are displayed dynamically with ltering options.

3.Order Placement:

- User adds items to the cart and proceeds to checkout.
- Order details are stored in the Sanity CMS.
- A payment gateway is used to process the transaction.

4.Shipment Tracking:

- After order placement, the shipment API provides real-time tracking updates.
- Shipment status (e.g., "Shipped," "Out for Delivery") is displayed to the user.

5.Payment Processing:

- Payment information is securely transmitted to the payment gateway.
- After payment is successful, a con rmation is sent to the user and recorded in the system.

5. Data Schema Design:

1. Schema for Products (Sanity CMS)

Documentation/SanitySchema.js

```
export default {
 name: "product",
 title: "Product",
 type: "document",
 .max(50)
 .warning("Tracking number should be between 5 to 50
characters."),
 },
 {
 name: "order",
 title: "Associated Order",
 type: "reference",
 to: [{ type: "order" }], // Link to the `order` schema
 validation: (Rule) => Rule.required().error("A shipment must be
associated with an order."),
 },
 {
 name: "carrier",
 title: "Carrier",
 type: "string",
 options: {
 list: [
 { title: "FedEx", value: "FedEx" },
 { title: "UPS", value: "UPS" },
 { title: "DHL", value: "DHL" },
 { title: "USPS", value: "USPS" },
 ],
```

```
layout: "dropdown",
},
validation: (Rule) => Rule.required().error("Carrier is required."),
},
{
name: "status",
title: "Shipment Status",
type: "string",
options: {
list: [
{ title: "In Transit", value: "In Transit" },
{ title: "Out for Delivery", value: "Out for Delivery" },
{ title: "Delivered", value: "Delivered" },
{ title: "Pending", value: "Pending" },
],
layout: "dropdown",
},
initialValue: "Pending",
validation: (Rule) => Rule.required(),
},
{
name: "estimatedDeliveryDate",
title: "Estimated Delivery Date",
type: "datetime",
options: {
dateFormat: "YYYY-MM-DD",
timeFormat: "HH:mm",
calendarTodayLabel: "Today",
},
validation: (Rule) => Rule.required().error("Estimated delivery
date is required."),
},
{
name: "actualDeliveryDate",
title: "Actual Delivery Date",
type: "datetime",
options: {
dateFormat: "YYYY-MM-DD",
timeFormat: "HH:mm",
calendarTodayLabel: "Today",
},
},
{
name: "shipmentNotes",
title: "Shipment Notes",
type: "text",
description: "Optional notes about the shipment.",
},
],
};
```

6. API Endpoints:

This document provides a detailed overview of the API endpoints required for the Furniture E-Commerce Platform. These endpoints are essential for interacting with products, orders, and shipment tracking within the platform.

1. Fetch All Available Products

Endpoint Name: /products

Method: GET

Description: Retrieves a list of all products available in the system. This data is sourced from Sanity CMS and includes product details such as ID, name, price, stock availability, and image URL.

Response Example:


```
[
 {
 "id": 1,
 "name": "Modern Sofa",
 "price": 599.99,
 "stock": 25,
 "image": "https://example.com/images/sofa.jpg"
 },
 {
 "id": 2,
 "name": "Wooden Dining Table",
 "price": 349.99,
 "stock": 15,
 "image": "https://example.com/images/dining-table.jpg"
 }
]
```


2. Create a New Order

Endpoint Name: /orders

Method: POST

Description: Creates a new order by collecting the customer's details, product selections, and payment status. The order is stored in Sanity CMS.

Payload Example:


```
{
 "customerName": "John Doe",
 "customerEmail": "johndoe@example.com",
 "items": [
 {
 "productId": 1,
 "quantity": 2,
```

"price": 599.99
},
{
"productId": 2,
"quantity": 1,
"price": 349.99
}
],
"paymentStatus": "Paid"
}

Response Example:

{
"orderId": "12345",
"customerName": "John Doe",
"totalPrice": 1549.97,
"status": "Pending"
}

3. Track Shipment Status

Endpoint Name: /shipment

Method: GET

Description: Fetches the shipment status for an order using the orderId. This data is sourced from a third-party shipment tracking API to provide customers with real-time delivery information.

Query Parameter Example:

/shipment?orderId=12345

Response Example:

{
"shipmentId": "67890",
"orderId": "12345",
"status": "Shipped",
"expectedDeliveryDate": "2025-02-01"
}

API Endpoint Summary

Endpoint Method Description Response

Fields

/products GET

Fetch all available

products from

Sanity CMS.

id, name, price,

stock, image

/product/:id GET

Fetch a speci c

product by ID to

view detailed info.

id, name,

description, price,

stock, images

/orders POST

Create a new order

in Sanity CMS.

orderId,

customerName,

totalPrice, status

/shipment GET

Track order

shipment status via

third-party API.

shipmentId,

orderId, status,

expectedDeliveryDate

4-Business Work ows:

1. Product Browsing and Selection:

- The /products endpoint allows customers to browse the catalog of products. The data fetched from this endpoint is displayed dynamically on the product listing page.
- The /product/:id endpoint is used when a user clicks on a speci c product to view more detailed information such as its description, price, and available stock.

2. Order Creation:

- Once customers have selected items, they can place an order by submitting the order details to the /orders endpoint. The backend captures customer information, order items, and payment status, and stores the order in Sanity CMS.

3. Shipment Tracking:

After placing an order, customers can use the /shipment endpoint to track their shipment status in real-time. The system fetches updates from the third-party shipment tracking API and displays the current status and expected delivery date to the user.

Conclusion

This API documentation highlights key endpoints for the Furniture E-Commerce Platform, supporting product browsing, order creation, and shipment tracking. The APIs are optimized for seamless frontend and backend integration and are easily extendable to meet future needs.

7. Technical Roadmap

Phase 1: Setup and Confuguration

- Set up Next.js for frontend development.
- Confegure Sanity CMS with required schemas for products, categories, and orders.

Phase 2: Frontend Development

- Design and implement essential pages: Home, Product Listing, Product Details, Cart, Checkout, Order Con rmation.
- Implement responsive design for a seamless user experience on all devices.

Phase 3: Backend Integration

- Integrate Sanity CMS for dynamic content management.
- Implement third-party API integrations for payment processing and shipment tracking.

Phase 4: Testing and Deployment

- Conduct comprehensive unit and integration testing for all components.
- Deploy the frontend to a platform like Vercel and the CMS to Sanity.
- Ensure scalability and reliability of both frontend and backend components.

Conclusion

This documentation details the design, technical specifications, and workflows for a scalable, high-performance Furniture E-Commerce Platform. Built with Next.js for the frontend, Sanity CMS for backend content management, and integrated third-party APIs, the platform ensures a seamless, user-friendly experience for customers and businesses. It is adaptable to evolving business and user needs throughout implementation.