Marketplace Technical Foundation - Hekto

Day 2 Activities: Transitioning to Technical Planning

1. Define Technical Requirements

The e-commerce furniture marketplace "Hekto" will include the following technical requirements:

- **Frontend:** Built with React.js for a dynamic user interface.
- **Backend:** Node.js for server-side processing.
- **Database:** Sanity as the content management system and database for managing product data and content efficiently.
- **Hosting:** Deployed using Vercel for frontend and backend hosting.
- **Payment Gateway:** Integration with Stripe for secure payment processing.
- **Authentication:** User authentication and authorization with Firebase.
- **Responsiveness:** Fully responsive design compatible with desktop, tablet, and mobile devices.

Why Use Sanity?

Sanity was chosen as the CMS and database solution for Hekto due to its unique features:

- **Real-time Collaboration:** Sanity allows multiple users to work on content simultaneously with real-time updates.
- **Flexibility:** Highly customizable schemas that adapt to the project requirements.
- **API-first Approach:** Provides robust and developer-friendly APIs for fetching and managing data.
- **Scalability:** Scales efficiently as the project grows, ensuring stable performance with increased content.
- **Content Studio:** An intuitive user interface for managing data and content, improving efficiency.
- **Live Previews:** Enables live previews of changes, ensuring content accuracy before deployment.

These features make Sanity a powerful and flexible choice for managing the content and data of the e-commerce furniture platform.

2. Design System Architecture

The system architecture for Hekto includes the following layers:

Presentation Layer:

- **Technology:** React.js
- **Responsibility:** Handles the user interface, including product pages, shopping cart, and checkout screens.

Application Layer:

- **Technology:** Node.js and Express.js
- **Responsibility:** Manages business logic, API requests, and integration with the database and third-party services.

Data Layer:

- **Technology:** Sanity CMS
- **Responsibility:** Stores product data, customer information, orders, and delivery details.

Integration Layer:

- **Technologies:**
  - Stripe for payment processing.
  - Firebase for authentication.
- **Responsibility:** Ensures seamless communication between the application and external services.

System Architecture Diagram:

```
Frontend (React.js)

   |

[Sanity CMS] ---------> [Product Data API]

   |

[Third-Party API] -----> [Shipment Tracking API]

   |

[Payment Gateway]
```

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. Users add products to their cart and proceed to checkout.
4. Payments are processed through the Payment Gateway (Stripe).
5. After successful payment, the order is updated in the database, and the shipment details are sent to the Shipment Tracking API.
6. Delivery zones are managed to calculate delivery time and charges.

Sanity Schema Example

Detailed Sanity Schema for Furniture Products

```
export default {
  name: 'product',
  type: 'document',
  title: 'Product',
  fields: [
    {
      name: 'name',
      type: 'string',
      title: 'Product Name',
    },
    {
      name: 'slug',
      type: 'slug',
      title: 'Slug',
      options: {
        source: 'name',
        maxLength: 200,
      },
    },
    {
      name: 'price',
      type: 'number',
      title: 'Price',
    },
    {
      name: 'stock',
      type: 'number',
      title: 'Stock Level',
    },
    {
      name: 'category',
      type: 'string',
      title: 'Category',
      options: {
        list: [
          { title: 'Sofa', value: 'sofa' },
          { title: 'Table', value: 'table' },
          { title: 'Chair', value: 'chair' },
          { title: 'Bed', value: 'bed' },
          { title: 'Cabinet', value: 'cabinet' },
        ],
      },
    },
    {
      name: 'tags',
      type: 'array',
      title: 'Tags',
      of: [{ type: 'string' }],
    },
    {
      name: 'description',
      type: 'text',
      title: 'Description',
    },
    {
      name: 'image',
      type: 'image',
       title: 'Product Image',
       options: {
        hotspot: true,
       },
       fields: [
        {
         name: 'alt',
         type: 'string',
         title: 'Alternative Text',
        },
       ],
      },
     ],
    };
```

3. Plan API Requirements

Hekto will require the following APIs for its functionality:

Authentication API:

- **Endpoints:**
  - POST /register - Register a new user.
  - POST /login - Authenticate a user.
  - GET /logout - Log out a user.

Product Management API:

- **Endpoints:**
  - GET /products - Retrieve all products.
  - GET /products/:id - Retrieve a specific product by ID.
  - POST /products - Add a new product (admin only).
  - PUT /products/:id - Update product details (admin only).
  - DELETE /products/:id - Delete a product (admin only).

Order Management API:

- **Endpoints:**
  - POST /orders - Create a new order.
  - GET /orders/:id - Retrieve order details.
  - PUT /orders/:id - Update order status (admin only).

Payment API:

- **Endpoints:**
  - POST /payments - Process a payment.
  - GET /payments/:id - Retrieve payment details.

Delivery Zone API:

- **Endpoints:**
  - GET /zones - Retrieve available delivery zones.
  - POST /zones - Add a new delivery zone (admin only).
  - DELETE /zones/:id - Remove a delivery zone (admin only).

4. Write Technical Documentation

Overview

This documentation provides an outline of the technical specifications and architecture for "Hekto," an e-commerce furniture marketplace.

Features

- Product browsing and search
- Secure user authentication
- Flexible content management using Sanity
- Efficient order processing and payment
- Real-time shipment tracking
- Delivery zone management

Technologies Used

Component	Technology
Frontend	React.js
Backend	Node.js

Component	**Technology**
Database/Content	Sanity CMS
Hosting	Vercel
Payment Gateway	Stripe
Authentication	Firebase

Deployment

Hekto will be deployed using Vercel for the frontend and backend, with Sanity providing a cloud-hosted CMS solution.

Scalability

The platform will leverage Sanity’s scalability and real-time capabilities to ensure a seamless experience as the platform grows.

This technical documentation outlines the foundational planning for the Hekto e-commerce furniture marketplace, ensuring clarity and direction for the development team.