Day 2 Activities: Technical Planning

System Architecture Overview:

The technologies used in designing this e-commerce website are as follows:

Frontend: Next.js

Dynamic Routing: pages like product details, categories, and user authentication (login/signup)

Server-Side Rendering (SSR) & Static Site Generation (SSG): Enhances speed and SEO

Component-Based Architecture: Reusable components include product cards, cart details, header/footer

ShadCN: Enhances User Interface:

Prebuilt Components: Faster UI development through adaptable, trendy components.

Responsive Design: Compatible with any device (desktop, tablet, mobile).

Content Management: Sanity CMS:

Headless CMS: Dynamically manages the content like products, descriptions, and images on Sanity.

Custom Schema: Defining the schema for content items like products, categories, and more.

API Integration with Sanity:

Data Fetching: From Sanity, it fetches dynamic content and makes API calls through the frontend for rendering in Next.js.

GROQ Queries: For retrieving product and other content data for rendering in Next.js from Sanity.

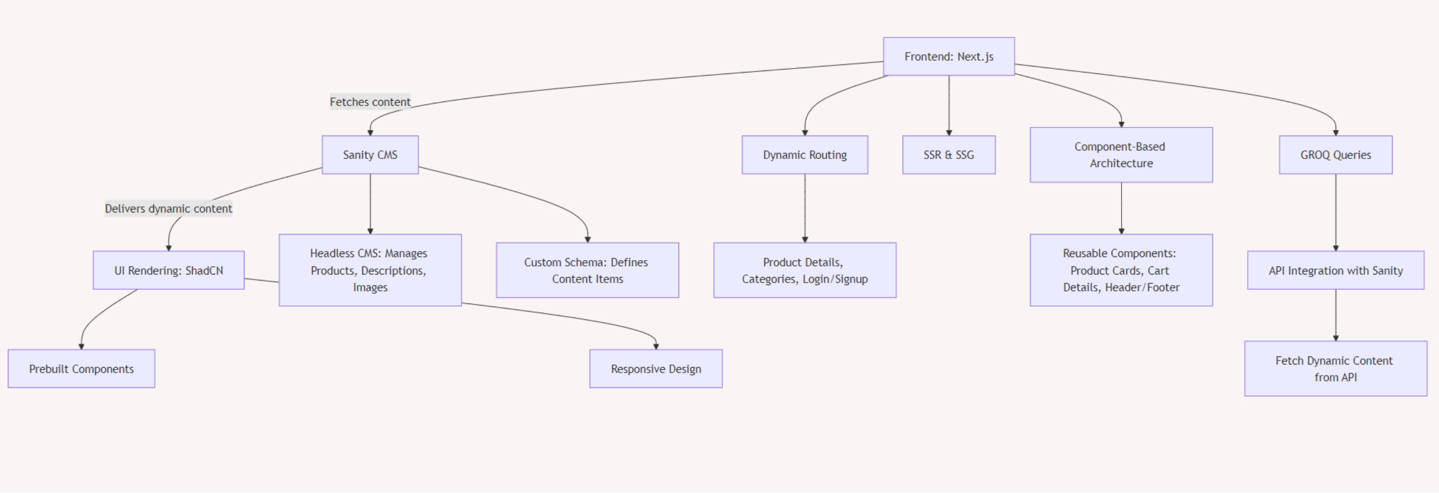
Data Flow:

Frontend (Next.js): Fetches the content from the Sanity API.

Sanity CMS: Delivers dynamic content through API endpoints.

UI Rendering ShadCN:

Renders the content using ShadCN components.


Frontend: Next.js
Fetches content
Sanity CMS
Dynamic Routing
SSR & SSG
Component-Based Architecture
GROQ Queries
Delivers dynamic content
UI Rendering: ShadCN
Headless CMS: Manages Products, Descriptions, Images
Custom Schema: Defines Content Items
Product Details, Categories, Login/Signup
Reusable Components: Product Cards, Cart Details, Header/Footer
API Integration with Sanity
Prebuilt Components
Responsive Design
Fetch Dynamic Content from API

API Integration Overview with Sanity:

User Engagement: Users call actions such as viewing products.

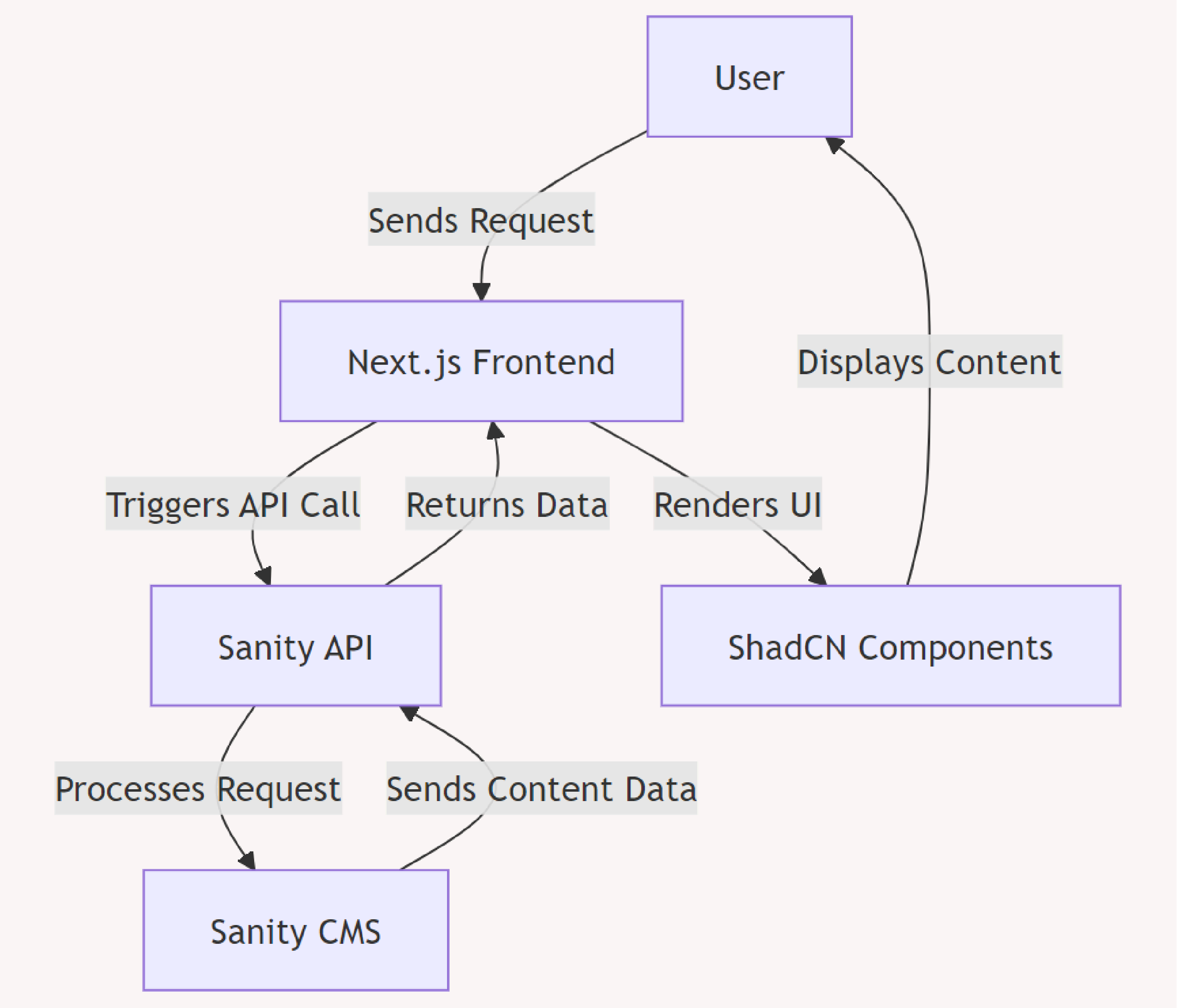
Next.js Frontend: Serves to query or update through the API and fetch data for Sanity.

Sanity API: Handles requests that reach the system and interacts with the CMS

Sanity CMS: Dynamically manages and keeps content

Data Fetching renders on the front-end through components from ShadCN

That is the sum of this, and the diagram below:



Order Placement Workflow:

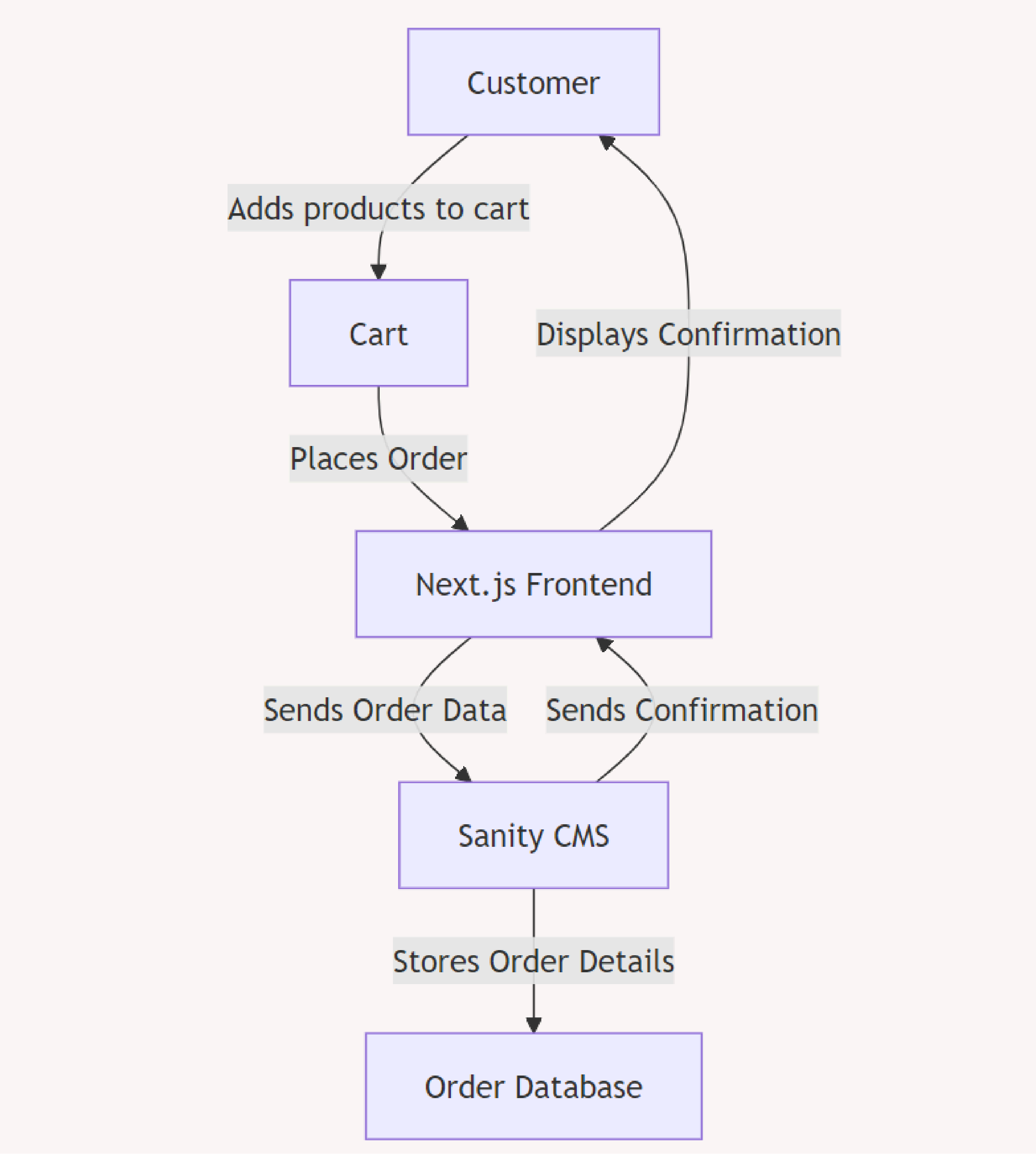
Product Selection: The customer browses products and adds them to the cart.

Order Placement: Once the customer confirms the order, the data is sent to **Sanity CMS** for storage.

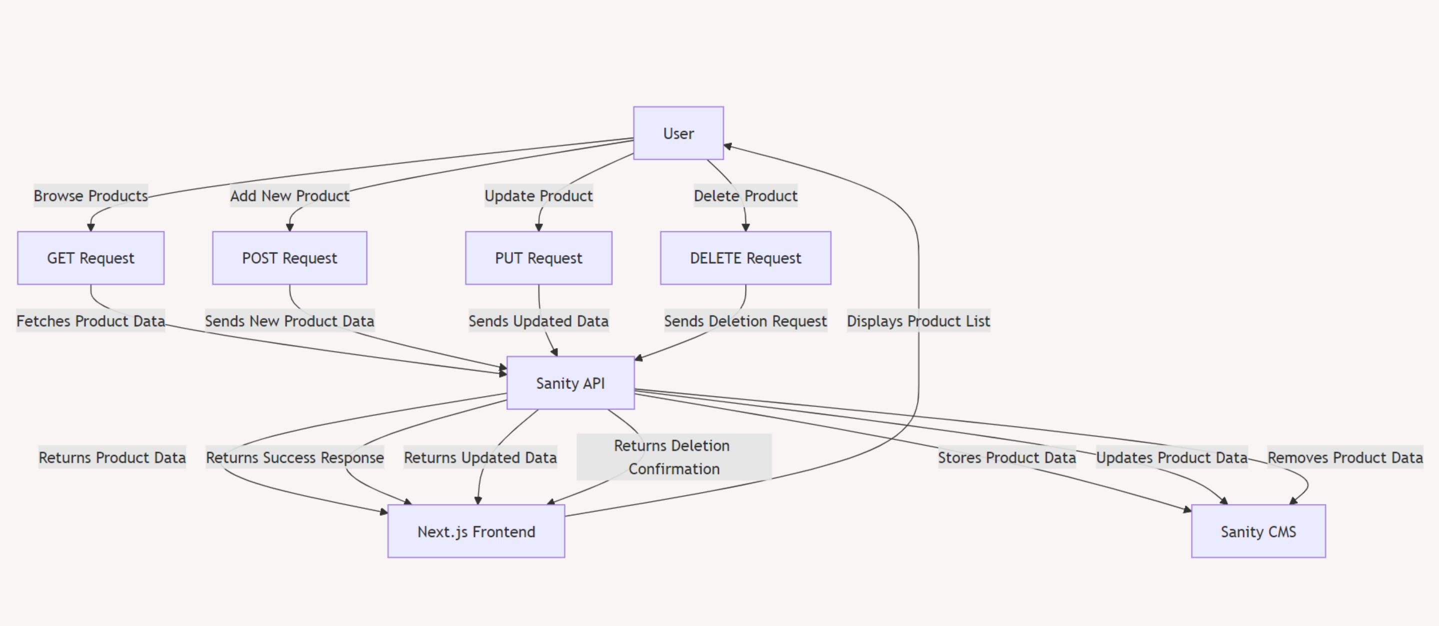
Sanity CMS: Stores order details such as customer information, product details, and order status.

Confirmation: The frontend retrieves a confirmation response from Sanity and displays it to the customer.

This flow ensures seamless order processing and secure data storage in Sanity CMS.



API Workflow for Furniture and Interior Design Website:
The API operations (GET, POST, PUT, DELETE) enable seamless communication between the frontend and Sanity CMS for managing product data, ensuring dynamic content updates and efficient data handling.


User
Browse Products
Add New Product
Update Product
Delete Product
GET Request
POST Request
PUT Request
DELETE Request
Fetches Product Data
Sends New Product Data
Sends Updated Data
Sends Deletion Request
Displays Product List
Sanity API
Returns Product Data
Returns Success Response
Returns Updated Data
Returns Deletion Confirmation
Stores Product Data
Updates Product Data
Removes Product Data
Next.js Frontend
Sanity CMS