DAY 2 PLANNING THE TECHNICAL FOUNDATION

Technical Requirements:

This step involves converting business goals into specific technical tasks or features that developers can implement. It ensures that your website has a clear structure and meets user needs.

Frontend Requirements:

These are the elements visible to users, focusing on design and usability:

User-Friendly Interface:

- The website should be intuitive, making it easy for users to browse products.

- Use clean layouts, clear navigation, and filters for seamless shopping.

Responsive Design:

The website should work perfectly on all devices (mobile, tablet, and desktop).

Essential Pages:

Home Page: First impression page with highlights of categories, offers, or popular products.

Product Listing Page: Displays all products in a category with sorting and filtering options.

Product Details Page: Detailed information about a specific product, including price, images, and descriptions.

Cart Page: Allows users to view selected products, modify quantities, or remove items.

Using Sanity CMS as a backend:

It stores and organizes all data for your e-commerce website, such as products, customers, and orders.

You can update or add products easily using Sanity's user-friendly interface without touching the code.

Sanity can track orders placed on your website, including product details, order dates, and payment status.

How it works:

Content Studio: Sanity provides a web-based interface where you can manage all your data.

API Integration: The frontend of your website connects with Sanity via its APIs to fetch or send data, such as displaying product listings or saving order information.

Third Party APIs:

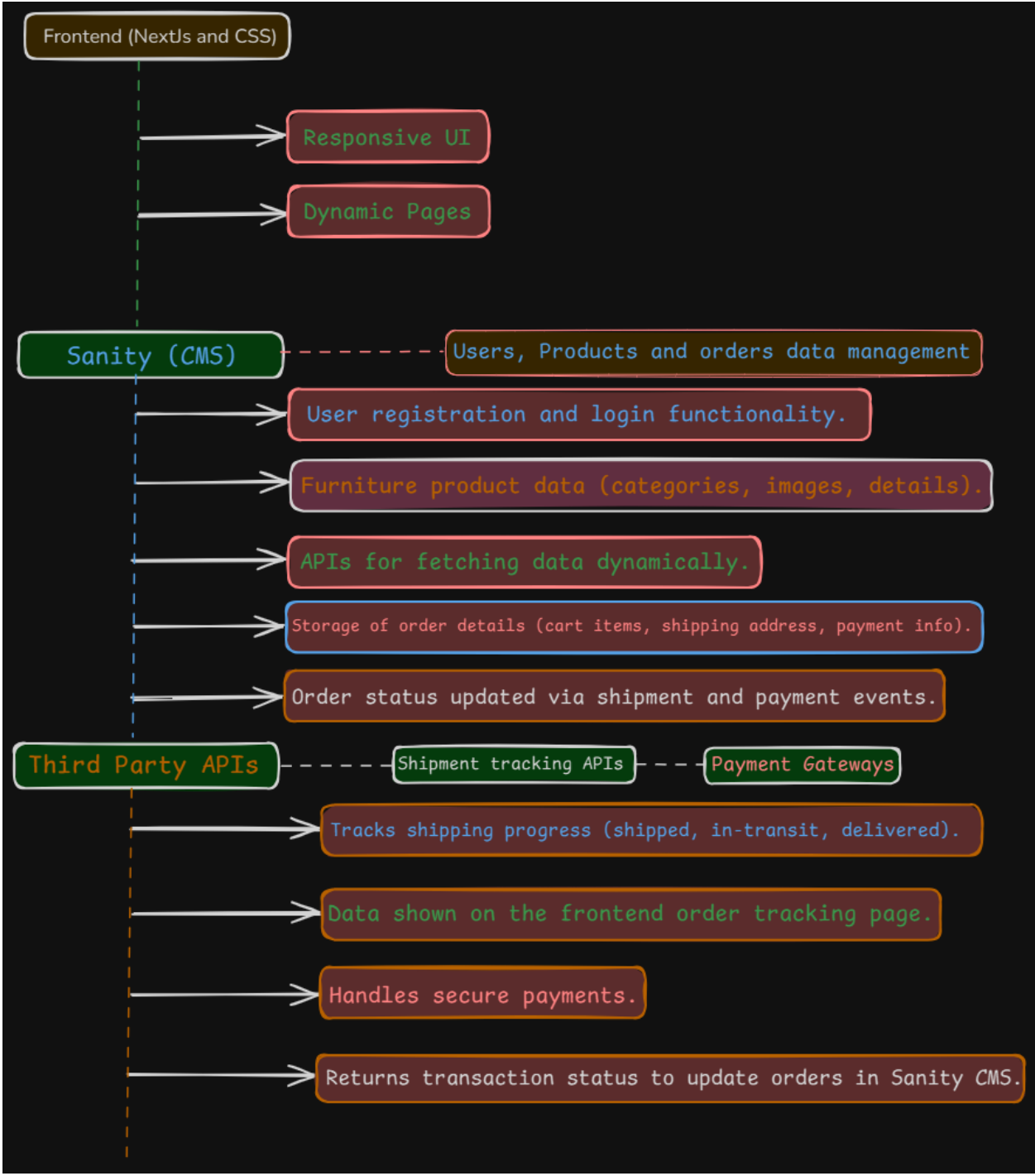
Third-Party APIs are essential for automating backend tasks like shipment tracking, payment processing, and other services. They connect external providers with your e-commerce platform, ensuring seamless functionality for both the backend and frontend.

1. Shipment Tracking APIs:

Integrate APIs from ShipEngine, Shippo , FedEx , EasyPost.

These APIs provide real-time tracking updates, delivery status, and shipment details for orders.

These APIs process payments, verify transactions, and return payment statuses to your system.


Frontend (NextJs and CSS)
Responsive UI
Dynamic Pages
Sanity (CMS)
Users, Products and orders data management
User registration and login functionality.
Furniture product data (categories, images, details).
APIs for fetching data dynamically.
Storage of order details (cart items, shipping address, payment info).
Order status updated via shipment and payment events.
Third Party APIs
Shipment tracking APIs
Payment Gateways
Tracks shipping progress (shipped, in-transit, delivered).
Data shown on the frontend order tracking page.
Handles secure payments.
Returns transaction status to update orders in Sanity CMS.

Endpoints	Method	Purpose	Response Example
/products	GET	Fetch all available products with details like price, stock, and image.	{ "id": 1, "name": "café chair", "price": 2500, "stock": 10, "image": "chair.jpg" }
/orders	POST	Create a new order with customer and product details, including payment.	{ "OrderID": 466, "status": "Success", "message": "Order created successfully!" }
/shipment	GET	Track the shipping status of an order via a third-party API.	{ "shipmentId": 788, "OrderID": 466, "status": "In Transit", "expectedDeliveryDate" "2025-01-20" }
/cart	POST	Add item to cart	{ “CartId”: 1, “quantity”: 2 , “status”: added }
/checkout	POST	Process checkout	{“OrderId”:466, “status”: “Confirmed” }
/login	POST	User login	{“token”: “abc_token”}
/Signup	POST	User Registration	{“status”: “Registered” }

1. Products Schema:

This schema represents the details of products available for sale.

```
export default {
name: 'product',
type: 'document',
fields: [
{ name: 'name', type: 'string', title: 'Product Name' },

{ name: 'description', type: 'string', title: 'Product Description' },
{ name: 'price', type: 'number', title: 'Price' },
{ name: 'stock', type: 'number', title: 'Stock Level' },
{ name: 'image', type: 'image', title: 'Product Image' }
]
};
```

2. Order Schema:

```
export default {
name: 'order',
type: 'document',
fields: [
{ name: 'customer', type: 'reference', to: [{ type: 'customer' }] },
```

]

};

3. Shipment Schema:

This schema tracks the shipment details for an order.

{

"shipmentId": "integer",

"orderId": "integer (Foreign Key)",

"status": "string (In Transit/Delivered)",

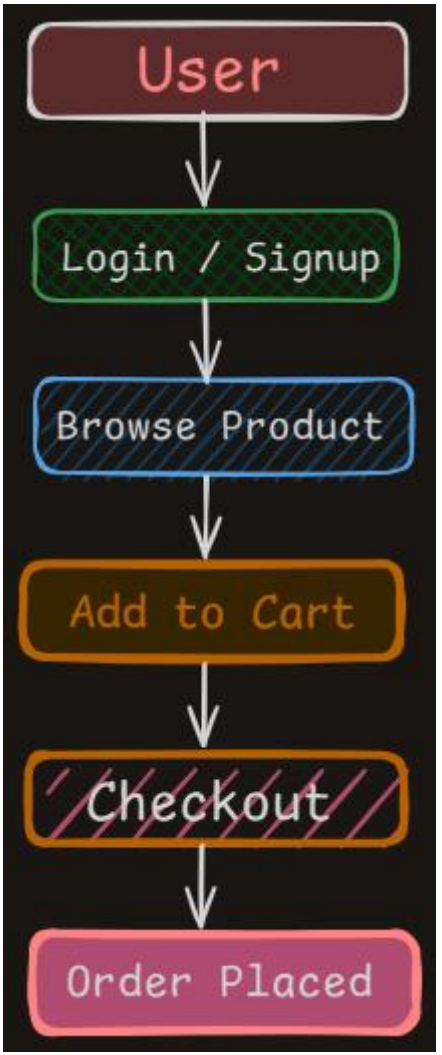
"expectedDeliveryDate": "date",

}

4. Technical Documentation:

Workflows:

- **Login and Signup:**
- User registers or logs in via the frontend.
- Details are authenticated and stored in the Sanity CMS.
- **Cart Management:**
- Items added to the cart are tracked using the /cart endpoint.
- **Order Confirmation:**
- User checks out, and the order is stored in Sanity CMS with payment status.


User
Login / Signup
Browse Product
Add to Cart
Checkout
Order Placed

1. System Architecture Diagram.
2. Detailed API Endpoints.
3. Workflows and Key Features.
4. Sanity CMS Schemas.

Thank You!
I hope you like my work
Day 2 Completed