Marketplace Technical Foundation – FurnitureMart.pk

Technical Requirements:

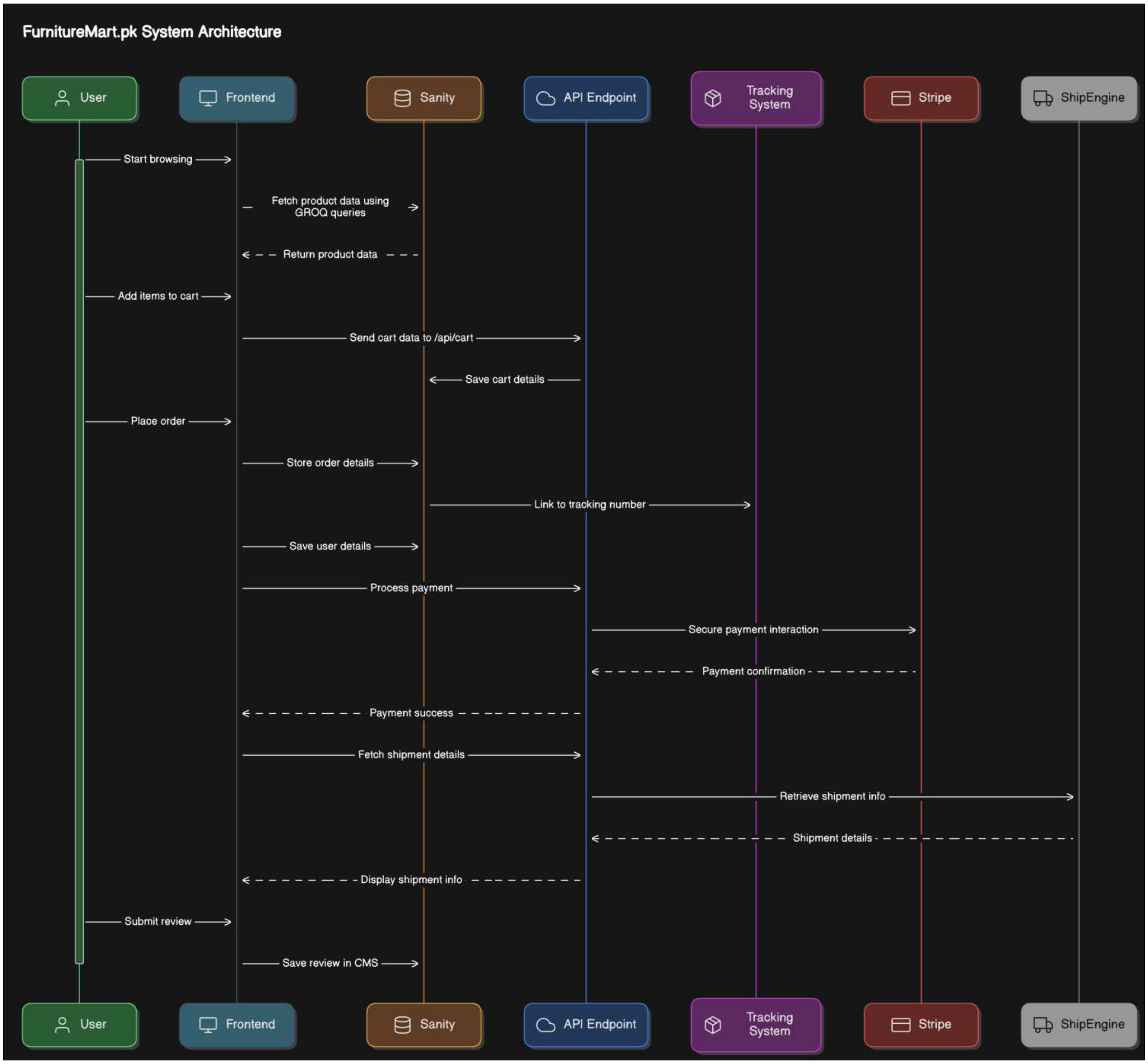
1. **Frontend Requirements:**
    - First, we need to focus on a **Responsive** and **user-friendly UI** for the Customers using Next.js, TypeScript, and Tailwind CSS with **easy navigation** and **clear CTA**.
    - **Pages We Need:**
        1. **Home Page**: To Showcase categories and featured furniture.
        2. **Product Listing Page**: To Display furniture items with filters.
        3. **Product Details Page**: For Detailed information with customization options.
        4. **Seller/Showroom Page**: To Display all products from a specific seller or showroom.
        5. **Cart Page**: To View selected items and proceed to checkout.
        6. **Checkout Page**: To Collect order and payment details.
        7. **Order Confirmation Page**: To Display order details After payment.
        8. **Customer Dashboard Page**: This will allow users to view past orders, tracking information, and other details.
        9. **Order Tracking Page**: User Can use this Page to track their orders with Provided Tracking ID.
        10. **Login/Signup Page**: User Can create new account or login to there account.

2. **Backend Requirements**

- Use **Sanity** for:
    - **Managing product data:** Name, price, stock, customization options, images, seller/showroom ID.
    - **Storing order details:** Customer information, product Details, Status, Order ID, Date/Time.
    - **Storing User Data:** User ID, Name, Contact Info, Address, Order history.
    - **Managing showroom/Seller data:** Seller ID, Name, Contact info, product listings.
    - **Storing and managing reviews for products:** Review ID, User ID, Product ID, Rating, Comment.
    - **Managing cart data:** Cart ID, User ID, product details, quantity.

- **NextAuth.js:**
    - For Authentication of user
- **External APIs:**
    - **ShipEngine API** for shipment tracking.
    - **Stripe API** for secure payment processing.
    - *In future we can add more payment methods, and AR Feature in this.*

Design System Architecture



System Workflow

1. Product Browsing:

- User visits the marketplace.
- Frontend fetches product data from Sanity.
- Products are displayed

2. Cart Management:

- User adds items to the cart.
- Frontend sends cart data to /api/cart.
- Cart details are saved in Sanity.

3. Order Placement:

- User adds items to the cart and proceeds to checkout.
- User’s Data is Saved in Sanity.

4. Payment Details:

- Frontend sends payment details to /api/payment.
- Upon payment success, order details are saved in Sanity.
- Tracking number is generated and linked to the user’s order.

5. Shipment Tracking:

- User views the order details page.
- Frontend calls /api/shipment with the order’s tracking number.
- Shipment status is fetched from ShipEngine and displayed to the user.

6. Review Submission:

- User submits a review for a purchased product.
- Frontend sends the review data to /api/review.
- Review is stored in Sanity and linked to the product with Customer ID.
- Reviews are displayed on the product details page.

7. User Authentication:

- **Signup**:
  - User creates an account on the signup page.
  - Frontend sends a POST request to /api/user to save the user's data in Sanity CMS.
  - A verification email is sent to the user with a unique token.
  - User clicks the link, and the token is verified via /api/auth/verify.
- **Login**:
  - User logs in using their email and password.
  - Frontend sends a POST request to /api/auth/login to authenticate the user and receive a session token.

API Endpoints

ApiEndpoints	Method	Description
/api/shipment	GET	Fetch shipment and tracking information from ShipEngine API.
/api/payment	POST	Process payment via Stripe API.
/api/cart	POST/GET/PATCH/DELETE	Store and manage cart data from frontend in sanity.
/api/user	POST	Add a new user to Sanity.
/api/order	POST	Submit a new order to Sanity
/api/review	POST/GET/PATCH/DELETE	Store and manage the reviews from frontend in sanity.
/api/auth/verify	GET	Verifies a user’s email address using a token.
/api/auth/reset	POST	Sends a password reset link to the user’s email.
/api/auth/login	POST	Authenticates a user and returns a session token.