🔗 Marketplace Technical Foundation - 👟ShoeTrend Store🛒

Welcome to my Marketplace Builder Hackathon 2025 project! 🎉
This doc showcases my journey in building a Shoes E-commerce Marketplace. Over the next 7 days, I will document and develop features step-by-step to create a robust and user-friendly platform.

1. Technical Requirements

Frontend Requirements 🔤

- **User-Friendly Interface:**
  1. 🎮 Easy-to-navigate website for shoes.
  2. 🔍 Showcase professional photos of shoes with zoom-in functionality.
- **Responsive Design:** 📱💻
  1. Optimized for both mobile and desktop users.
- **Essential Pages:**
  1. 🏠 **Home Page**: Highlights featured shoe collections.
  2. □ **Product Listings Page**: Displays categories (e.g., men's, women's, and kids' shoes).
  3. 📄 **Product Details Page**: Shows a shoe's description, sizes, price, and stock availability.
  4. 🛒 **Cart Page**: Lists items added for purchase.
  5. 💳 **Checkout Page**: Includes a form for delivery information and payment processing.
  6. ✅ **Order Confirmation Page**: Displays a summary of the placed order.

Backend Requirements (Sanity CMS) ⚙️

Use **Sanity CMS** for managing:

👟**Products**: Names, descriptions, prices, stock, sizes, and images.

📦 **Orders**: Customer information, product details, payment status, and order history.

👤 **Customer Details**: Store and retrieve customer names, emails, and addresses.

Third-Party APIs 🔌

- 💰 **Payment Gateway**: Use Stripe for secure and reliable payment processing.

- 🚚 **Shipment Tracking API**: Integrate a third-party API to update customers on order delivery status in real-time.

---

2. System Architecture 🌐

System Overview 🛠

Here’s how the components of the marketplace interact:

1. **Frontend (Next.js)**:
    - The user interacts with a modern and responsive interface.
    - Fetches product and order data dynamically via APIs.
2. **Sanity CMS**:
    - Acts as the database for products, orders, and customer information.
    - Provides APIs to interact with frontend components.
3. **Third-Party APIs**:
    - **Stripe**: Processes payments securely.
    - **Shipment Tracking API**: Tracks and displays the status of deliveries.

System Architecture Diagram 🎮

```
[Frontend (Next.js)]
     |
[Sanity CMS] <----> [Products API]
     |
[Third-Party APIs]
     |---> [Payment Gateway (Stripe)]
     |---> [Shipment Tracking API]
```

---

3. API Requirements 📡

API Endpoints 🚀

Endpoint	Method	Purpose	Request/Response
`/products`	GET	Fetch all products	🔄 { "id": 1, "name": "Shoe", "price": 150 }
`/product/:id`	GET	Fetch one product detail	🔄 { "id": 1, "name": "Running Shoe", ... }
`/cart`	POST	Add item to cart	📦 { "productId": 123, "quantity": 1 }
`/checkout`	POST	Place an order	🛒 { "customerInfo": {}, "cart": [] }
`/order/:id`	GET	Fetch order details	✅ { "orderId": 1, "status": "Shipped" }
`/shipment/:id`	GET	Track shipment	🚚 { "shipmentId": 123, "ETA": "2 Days" }

4. Technical Documentation 📝

Sanity Schema

1. Product Schema □

```
export default {
name: 'product',
type: 'document',
fields: [
   { name: 'name', type: 'string', title: 'Product Name' },
   { name: 'price', type: 'number', title: 'Price' },
   { name: 'stock', type: 'number', title: 'Stock Level' },
   { name: 'image', type: 'image', title: 'Product Image' },
   { name: 'description', type: 'text', title: 'Description' },
 ],
};
```

2. Order Schema

```
📦 export default {
name: 'order',   type:
'document',   fields: [
    { name: 'customer', type: 'reference', to: [{ type: 'customer' }], title: 'Customer' },
    { name: 'products', type: 'array', of: [{ type: 'reference', to: [{ type: 'product' }] }] },
{ name: 'paymentStatus', type: 'string', title: 'Payment Status' },
  ],
};
```

Workflows 🔄

1. Product Browsing ▯

- User visits the homepage or product listing page. ● Frontend fetches data via `/products` API.
- Products are displayed dynamically.

2. Order Placement 🛒

- User adds items to the cart (`/cart` endpoint).
- User proceeds to checkout and places an order (`/checkout` endpoint).
- Order details are saved in **Sanity CMS**.

3. Shipment Tracking 🚚

- Frontend sends a GET request to `/shipment/:id` endpoint.
- Shipment status is displayed on the frontend.