**🚀 Day 2 - System Design & Integration: Marketplace Builder Hackathon 2025**

Welcome to Day 2 of my Marketplace Builder Hackathon 2025 journey! 🎉 Today, I focused on creating a robust system architecture, detailing how the frontend connects with the backend, and integrating third-party APIs to enhance functionality.

**🔧 System Design Overview**

The system is structured to ensure scalability and seamless interaction between components. This will deliver users a smooth and dynamic shopping experience. Below is the planned system design:

**🔗 Frontend-Backend Integration**

**Frontend Framework:**

* Built with **Next.js** and styled using **Tailwind CSS**, complemented by a component library such as ShadCn UI.
* **State Management:** Handled with **Redux** to manage complex application states effectively.

**Backend:**

* Developed using **Sanity.io** as the primary CMS and database, with custom API routes for communication.

**Communication:**

* The frontend interacts with the backend via RESTful API endpoints using standard HTTP methods (GET, POST, PUT, DELETE).

**🌐 Third-Party API Integration**

To enhance marketplace functionality, third-party APIs are integrated into the platform.

**Purpose:**

* Payment processing and live product availability updates.

**Integration Flow:**

1. The frontend sends requests to the backend.
2. The backend processes these requests, interacts with the third-party API, and retrieves data.
3. Responses are sent back to the frontend for display.

**Security Measures:**

* All communication uses HTTPS.
* API access is secured with key-based authentication.

**📊 API Routes Definition**

Here’s a breakdown of the API routes and their respective functions:

1. **GET /api/products:** Retrieves a list of available products.
2. **POST /api/order:** Creates a new order and processes payment through the third-party API.
3. **GET /api/order/:id:** Fetches details of a specific order by its ID.
4. **PUT /api/order/:id:** Updates the status of an order (e.g., "Shipped" or "Delivered").
5. **GET /api/payment/verify:** Confirms payment status via the third-party API.

**💡 Tech Stack**

* **Frontend:** Next.js, Tailwind CSS, Zustand for local state management.
* **Backend:** Sanity.io for database and API route handling.
* **Third-Party API:** Integration for payment processing and inventory updates.
* **Authentication:** Secured using **Clerk** for user login and transaction safety.

**📃 Documenting the Workflow**

The flow of data across components is outlined as follows:

1. The frontend sends requests to the backend via defined API routes.
2. The backend processes the requests, interacts with the third-party API, and sends the results back to the frontend.
3. The frontend updates the user interface based on the backend responses.

**📊 Visual Data Structure**

A detailed diagram illustrating the system’s data structure and interactions will be created to guide implementation. https://app.eraser.io/workspace/f8RoeSaW7UznSzcoHsup?origin=share

**🔒 Next Steps**

Moving forward, I’ll focus on:

* Developing the API routes.
* Integrating the third-party APIs for payment and product updates.
* Starting the frontend development to connect and visualize all components.

**🚀 Day 2 Summary**

Today’s efforts laid the foundation for a fully functional and integrated marketplace. With the system design and integration plan ready, the next phase will turn these designs into reality. Stay tuned for updates!