Day 2 implementation Report

 # Marketplace Builder Hackathon - Day 2 Report

Project Name: Furniro - A Modern Furniture Marketplace

1. Overview

Day 2 of the **Furniro Marketplace** hackathon focused on planning the technical foundation of the platform. This involved defining the data architecture, selecting APIs, structuring the backend, and planning integrations to ensure scalability and efficiency.

2. Objectives of Day 2

The key objectives were:

- **Defining Data Models:** Planning the structure for products, users, and orders.
- **Selecting APIs & Services:** Choosing the best APIs for authentication, product management, and payments.
- **Backend Architecture Planning:** Designing how the backend will handle requests, authentication, and data storage.
- **State Management Approach:** Deciding how to manage state across the application.

3. Technical Stack & Architectural Decisions

Frontend:

- **Next.js**: For server-side rendering and static site generation.
- **React.js**: Component-based UI development.
- **Tailwind CSS**: For responsive and modern styling.

Backend & APIs:

- **Node.js & Express**: Planned for handling API requests and authentication.
- **Firebase Firestore / MongoDB**: Considered for storing product and user data.
- **Stripe API**: Chosen for secure and seamless payment processing.
- **NextAuth.js**: For user authentication and session management.

State Management:

- **React Context API**: For lightweight state management.
- **Redux Toolkit** (optional): For handling complex global state scenarios.

4. Key Features Planned on Day 2

1. **Product Data Model Design:**
    - Product details: Name, description, price, category, stock status, images.
    - User roles: Admin, Seller, Customer.
    - Orders: Order status, payment details, user details.
2. **API Selection & Integration:**
    - Stripe API for payments.
    - REST API or GraphQL for fetching product data.
    - Firebase Authentication for secure user login.
3. **Backend System Planning:**
    - Database schema for structured data storage.
    - Routes for handling product fetching, user authentication, and orders.
    - Middleware for authentication and role-based access control.
4. **State Management Strategy:**
    - Implementing Context API for handling user authentication.
    - Using local storage to persist cart data.
    - Option to integrate Redux Toolkit if state complexity increases.

5. Challenges Faced & Solutions

Challenge	Solution
Choosing between REST and GraphQL for API	Opted for REST API for simplicity and ease of integration.
Ensuring secure authentication	Decided to use Firebase Authentication and NextAuth.js.
Selecting the best payment gateway	Chose Stripe for its security, ease of integration, and global support.
Structuring database schema for scalability	Planned a NoSQL approach with Firestore for flexibility.

7. Future Enhancements

- Implement GraphQL for optimized data fetching.
- Enhance security with OAuth and JWT-based authentication.
- Set up Webhooks for real-time order status updates.
- Optimize API response times using caching mechanisms.

8. Conclusion

Day 2 focused on laying the technical groundwork for the **Furniro Marketplace**. The project now has a well-defined data model, API structure, backend architecture, and state management approach, ensuring smooth development in the next phases.