Collaboration Notes for Peerzada Store - Day 2

Overview:

The collaboration process for planning the technical foundation of Peerzada Store involved detailed discussions with peers and mentors. The primary goal was to refine the technical plan to ensure scalability, performance, and alignment with the marketplace's business goals. This document captures the key discussions, feedback, and collaborative strategies used during Day 2 of the hackathon.

---

1. Discussion Topics:

1.1 System Architecture:

- **Initial Proposal:**
  - Frontend: Built with Next.js for dynamic and responsive user interfaces.
  - Backend: Sanity CMS for content management and data storage.
  - Third-Party APIs: Integration for payment processing and shipment tracking.
- **Feedback:**
  - Ensure separation of concerns between frontend and backend to simplify debugging.
  - Use environment variables for API keys and sensitive configurations.

1.2 Data Schema:

- **Products:**
  - Attributes: Name, Price, Stock, Category, and Customisation Options.
  - Suggestion: Include a "Discount" field for promotional campaigns.
- **Orders:**
  - Attributes: OrderID, CustomerID, ProductDetails, TotalAmount, Status.
  - Suggestion: Add "Timestamp" to track the time of order placement.
- **Customers:**
  - Attributes: Name, ContactInfo, City, OrderHistory.
  - Suggestion: Include a "Loyalty Points" system to enhance customer retention.

1.3 API Design:

- **Proposed Endpoints:**
  - `/products` (GET): Fetch all available products.
  - `/orders` (POST): Create a new order.
  - `/shipment` (GET): Track shipment details.
- **Feedback:**
  - Use pagination for `/products` to improve performance for large datasets.
  - Include error handling and status codes in API responses.

1.4 Workflow Optimization:

- **Product Browsing:**
  - Proposal: Fetch product data dynamically from Sanity CMS.
  - Suggestion: Implement caching for frequently accessed data to reduce latency.

- **Order Placement:**
  - Proposal: Send order data to Sanity CMS via API.
  - Suggestion: Validate customer inputs at both frontend and backend.
- **Shipment Tracking:**
  - Proposal: Use a third-party API for real-time updates.
  - Suggestion: Display shipment updates with a progress bar for better user experience.

---

2. Tools Used:

2.1 Communication:

- **Slack:**
  - Facilitated real-time discussions on API design and workflow diagrams.
  - Channels created for specific tasks, such as "API Feedback" and "Schema Design."

2.2 Collaboration:

- **Google Meet:**
  - Conducted video calls to brainstorm and finalize the technical foundation.
- **Lucidchart:**
  - Used for creating system architecture diagrams.

2.3 Version Control:

- **GitHub:**
  - Repository created to track changes to schemas, workflows, and API designs.
  - Branches established for individual contributions, such as "product-schema" and "workflow-diagram."

---

3. Peer Feedback:

3.1 Scalability:

- **Issue Identified:**
  - Potential bottleneck with Sanity CMS when handling a high volume of orders.
- **Solution Proposed:**
  - Implement database indexing and optimize API queries.

3.2 User Experience:

- **Issue Identified:**
  - Limited customisation options for users during checkout.
- **Solution Proposed:**
  - Add live previews of customised products and implement conditional logic for pricing.

3.3 API Design:

- **Issue Identified:**
  - Lack of authentication for sensitive endpoints.
- **Solution Proposed:**
  - Use JWT-based authentication to secure endpoints.

---

4. Key Outcomes:

1. **Enhanced System Architecture:**
   - Finalized the integration between frontend, Sanity CMS, and third-party APIs.
2. **Refined Data Schema:**
   - Included additional fields for promotional campaigns and loyalty programs.
3. **Improved API Design:**
   - Added pagination, error handling, and authentication mechanisms.
4. **Optimized Workflows:**
   - Introduced caching and real-time validation to improve performance and usability.
5. **Documentation:**
   - Prepared detailed technical documents, including system diagrams and API specifications.

---

5. Next Steps:

1. **Implementation:**
   - Set up Sanity CMS with the finalised schemas.
   - Develop API endpoints with secure and efficient logic.
2. **Testing:**
   - Perform unit and integration testing for workflows.
3. **Launch Preparation:**