Technical Overview

1. Define Technical Requirements

Frontend Requirements:

- **User-friendly Interface**: The interface should be intuitive, allowing users to easily browse and filter products. It should have:
  - Search bar for products.
  - Filters for categories, price ranges, and other product attributes.
  - Clear product pages with images, descriptions, and prices.
  - Easy-to-use cart and checkout process.
- **Responsive Design**: The design must work on all screen sizes, from mobile phones to desktops. Ensure:
  - Fluid layout that adjusts to different screen sizes.
  - Navigation and buttons should be easy to tap on mobile devices.
- **Essential Pages**: The site must have key pages:
  - Home: Displays featured products, categories, and promotions.
  - Product Listing: Displays all products in a specific category or search result.
  - Product Details: Detailed information about each product.
  - Cart: Shows products added to the cart.
  - Checkout: Process to complete the order (enter shipping, payment info).
  - Order Confirmation: A page showing the order summary and confirmation.

Sanity CMS as Backend:

- **Sanity CMS** will manage the content and data for your marketplace:
  - Product Data: Product names, descriptions, prices, images, etc.
  - Customer Details: User information, order history, etc.
  - Order Records: Order IDs, products purchased, shipping status, etc.
- **Sanity Schema:**
  - Define product schema with fields like name, description, price, category, images, etc.
  - Define order schema with fields like order number, products ordered, customer details, and order status.
  - Define customer schema with fields like name, contact info, and order history.

Third-Party APIs:

- Shipment Tracking: Integrate with an API like ShipEngine to track the shipment of orders.
- Payment Gateways: Integrate with payment services like Stripe or PayPal for secure transactions.
- Other APIs: For additional functionality like user authentication, shipping rates, etc.

---

2. Design System Architecture

High-level System Architecture Diagram and Workflows:

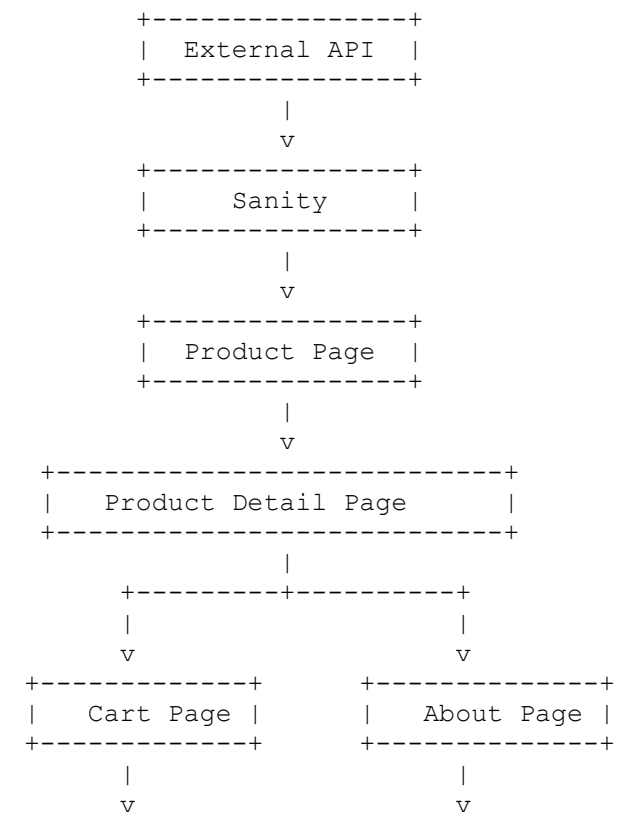
Architecture Diagram:

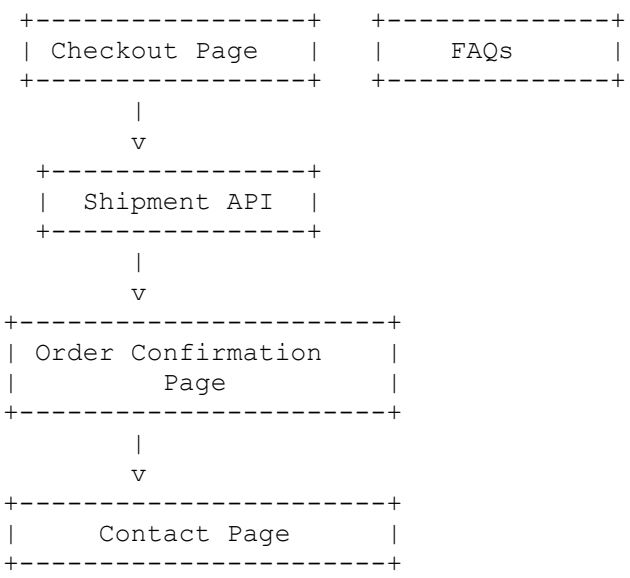
```
[Frontend (Next.js)] <--> [Sanity CMS] <--> [Product Data API]
                                |
                                v
                          [Shipment Tracking API] <--> [Payment Gateway]
```

Key Workflows:

- **User Registration**: Users sign up, and their details are saved in Sanity CMS.
- **Product Browsing**: Products are fetched from Sanity CMS and displayed on the frontend.
- **Order Placement**: Once a user places an order, order data is sent to Sanity CMS and processed by a payment API.
- **Shipment Tracking**: Once an order is confirmed, shipment details are fetched from a third-party shipment tracking API.

Detailed Workflow Block Diagram:





This diagram outlines the workflow and interactions between your website's pages and components. It visually supports the explanations provided in this section and aligns with the workflows described earlier.

---

3. Plan API Requirements

Define Key API Endpoints with Examples:

- **/products (GET)**: Fetch all product details.
  - Example: /api/products → returns a list of products with details like name, price, description, and images.
- **/orders (POST)**: Create a new order in Sanity CMS.
  - Example: /api/orders → accepts order data (products, customer details) and creates an order record in Sanity.
- **/shipment (GET)**: Track order status via a third-party shipment API.
  - Example: /api/shipment/:orderId → returns the current status of the order.
- **Additional Example Endpoints:**
  - **/express-delivery-status (GET)** → Fetch real-time delivery updates from a third-party shipment tracking API.
  - **/rental-duration (POST)** → Add rental details for a product (e.g., rental duration).

---

4. Write Technical Documentation

System Architecture Document:

- Provide a detailed explanation of the system components (Frontend, CMS, APIs).

- Explain how data flows between the frontend, Sanity CMS, and third-party APIs.
- Include any key decisions about tools, frameworks, or technologies used.

API Specification Document:

- Document each API endpoint:
  - Method (GET, POST, etc.)
  - Endpoint (e.g., /products, /orders)
  - Request Body (fields and data types)
  - Response Body (expected response)
  - Example Requests/Responses

Workflow Diagram:

- A diagram illustrating user interactions and data flow, from browsing products to placing an order and tracking shipment.

Technical Roadmap:

- Break down the development into milestones:
  - Phase 1: Set up Sanity CMS and define schemas.
  - Phase 2: Develop frontend pages (Home, Product Listing, etc.).
  - Phase 3: Integrate third-party APIs (payment gateway, shipment tracking).
  - Phase 4: Test and deploy the application.

---

5. Collaborate and Refine

Group Discussions:

- Engage with your peers to discuss the technical plan, challenges, and ideas.
- Brainstorm possible solutions and approaches to ensure the system works smoothly.

Peer Reviews:

- Share your documentation and code with peers to get feedback on clarity, completeness, and correctness.

Version Control:

- Use GitHub (or another version control tool) to track changes to your code, diagrams, and documentation.
- Create separate branches for different tasks and merge them once completed.

Key Outcome of Day 2:

- **Technical Plan Aligned with Business Goals**: The technical requirements and plan should reflect your marketplace type and business objectives.
- **System Architecture Visualized**: A clear diagram that illustrates the system components and their interactions.
- **Detailed API Requirements**: A well-defined list of API endpoints and their expected behavior.
- **Sanity Schemas Drafted**: A detailed schema for key data entities in Sanity.
- **Collaborative Feedback Incorporated**: Incorporate feedback from peers and mentors into the final plan.

Industry Best Practices:

1. **Plan Before You Code**: Creating a roadmap saves time and reduces rework.
2. **Use the Right Tools**: Leverage Sanity CMS and APIs to streamline development.
3. **Collaboration**: Always involve peers and mentors for valuable feedback.
4. **Focus on User Experience**: Ensure that the technical solution aligns with a seamless user experience.

Submission Guidelines:

1. **Repository Submission:**
    - Create a folder named "Documentation" in your repository and upload all technical documents, diagrams, and schemas.
2. **Document Structure:**
    - Follow the standard format provided in the task description.
    - If applicable, include collaboration notes or peer review comments.
3. **File Naming Convention:**
    - Use clear names for your documents (e.g., SystemArchitecture_Day2.pdf, APIEndpoints.xlsx, SanitySchema.js).
4. **Review and Quality Check:**
    - Double-check all diagrams, schemas, and written content for accuracy and clarity before submitting.

Collaborate with peers or mentors to refine your work