1- System Architecture

1. Frontend

Technology Stack:

Framework	Next.js (for SSR/CSR, SEO optimization, and dynamic routing)
Styling	Tailwind CSS/ShadCN/DaisyUI and others as per requirement.
State Management	Context API
Component Library	React Icons, Heroicons and others as per requirement.
Testing	Jest, React Testing Library

Key Features:

- Responsive and friendly UI
- Dynamic product pages
- User authentication pages (login, signup)
- Shopping cart and wishlist
- Search and filtering
- Checkout process

2. Backend

Technology Stack:

Framework	Sanity CMS will be integrated into the Node.js/Express API for serving content, specifically related to product data, categories, and any marketing-related content (like blog posts, static pages). You'll create endpoints in your Node.js API to fetch data from Sanity via their API.
Database	MongoDB will store dynamic and transactional data (like user profiles,

	orders, and cart data), while Sanity CMS will store structured content data (like products, categories, blog posts).
Authentication	JSON Web Tokens (JWT) for secure session handling
File Storage	Sanity CMS can store image metadata (e.g., URLs)
Payments	Stripe/PayPal integration
Cache	Redis for session and caching
Logging & Monitoring	Log updates to product data when changes are made in Sanity via backend API calls.

Key Features:

- Product management (CRUD operations)
- User management (authentication, roles)
- Order processing and management
- API for product search and filtering
- Integration with payment gateway APIs

3. Infrastructure

Technology Stack:

Hosting	Vercel for frontend, AWS EC2 for backend
CI/CD Pipeline	GitHub Actions
Domain & SSL	Cloudflare or AWS Route 53
Load Balancer	AWS ELB (Elastic Load Balancer)
Scaling	Auto-scaling with AWS ECS/Kubernetes