Eleva Shop

```mermaid
graph TD
    Frontend(Next.js) --> BackendAPIs(Backend APIs)
    BackendAPIs --> CMS(Sanity CMS)
    BackendAPIs --> Database(Database)
    BackendAPIs --> ThirdPartyAPIs(Third-party APIs)
    Authentication(Authentication Service) --> Database
    Caching(Caching Layer) --> BackendAPIs
    Monitoring(Monitoring & Error Tracking) --> Frontend
    Monitoring --> BackendAPIs
    Monitoring --> Database
    Monitoring --> ThirdPartyAPIs
    CDN(CDN) --> Frontend
    CI_CD(CI/CD Pipeline) --> Frontend
    CI_CD --> BackendAPIs
```

Explanation of Components:

1. **Frontend (Next.js):**
   - Handles user interface and interactions.
   - Utilizes server-side and client-side rendering for performance and SEO.

2. **Backend APIs:**
   - Manages data processing and business logic.
   - Includes Product Data API, Shipment Tracking API, and Payment Gateway API.

3. **CMS (Sanity):**
   - Manages content and provides data to Backend APIs.

4. **Database:**
   - Stores persistent data such as user information and orders.
   - Could be relational (e.g., PostgreSQL) or NoSQL (e.g., MongoDB).

5. **Third-party APIs:**
   - External services like Shipment Tracking (e.g., UPS, FedEx) and Payment Gateway (e.g., Stripe, PayPal).

6. **Authentication Service:**
    - Handles user authentication and session management.
    - Integrates with providers like Google, GitHub (using NextAuth.js).

7. **Caching Layer:**
    - Improves performance by caching frequently accessed data.
    - Tools like Redis or Memcached can be used.

8. **Monitoring & Error Tracking:**
    - Monitors system health and tracks errors.
    - Tools like Sentry or New Relic can be employed.

9. **CDN:**
    - Serves static assets efficiently, improving load times.

10. **CI/CD Pipeline:**
    - Automates testing and deployment.
    - Tools like GitHub Actions or Jenkins are suitable.

Key Considerations:

- **Security:** Ensure data transmission security using HTTPS and input validation.
- **Scalability:** Design components to scale independently to handle increased load.
- **Redundancy:** Consider failover mechanisms for critical components.