

Hackathon 3 Document Day 2

General E-commerce Marketplace Technical Foundation

Prepared Date: January 16, 2025 by: Azmat Ali

1. Technical Plan Aligned with Business Goals

Business Objectives

- Create a scalable e-commerce platform
- Provide seamless shopping experience
- Enable efficient product management
- Ensure secure payment processing
- Implement order tracking and management

Technical Requirements

Frontend Requirements

- Next.js-based responsive web application
- Server-side rendering for improved SEO
- Client-side state management using React Context/Redux
- Progressive Web App (PWA) capabilities
- Responsive design breakpoints: Mobile (< 768px), Tablet (768px - 1024px), Desktop (> 1024px)

Backend Requirements (Sanity CMS)

- Content management for products, categories, and orders
- Real-time inventory tracking
- Order management system
- Customer data management
- Role-based access control

Third-party Integrations

- Payment Gateway (Stripe)
- Email Service (SendGrid)
- Image CDN (Cloudinary)
- Analytics (Google Analytics)
- Search Implementation (Algolia)

E-Commerce Work Process By Image

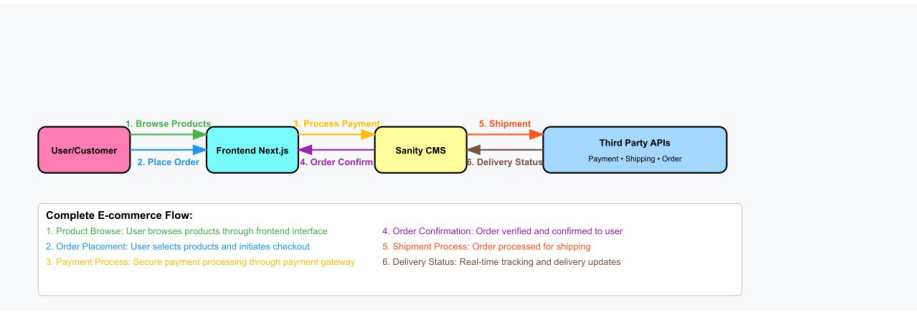
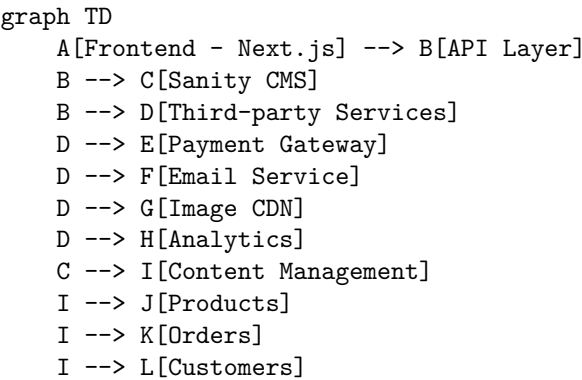


Figure 1: Image

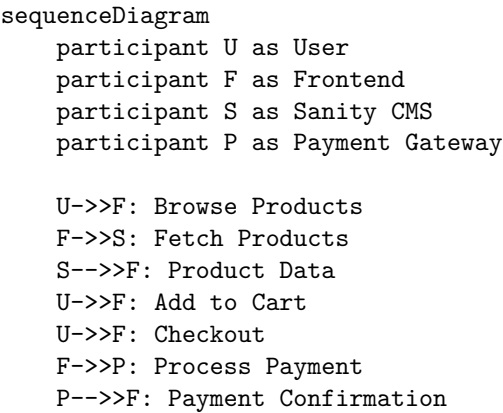
2. System Architecture

High-Level Architecture Diagram



Core Workflows

Product Browse & Purchase Flow



F->>S: Create Order
 S-->>F: Order Confirmation
 F->>U: Success Message

3. API Requirements

Product Management APIs

| Endpoint | Method | Description | Request/Response |
|--------------------|--------|--------------------|--|
| /api/products | GET | Fetch all products | Response: { products: [{id, name, price, stock}] } |
| /api/products/{id} | GET | Get single product | Response: { id, name, price, description, stock } |
| /api/categories | GET | Fetch categories | Response: { categories: [{id, name, products}] } |
| /api/orders | POST | Create order | Request: { items, customer, payment } |
| /api/orders/{id} | GET | Get order status | Response: { id, status, items, tracking } |

4. Sanity Schemas

Product Schema

```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    {
      name: 'name',
      title: 'Name',
      type: 'string',
      validation: Rule => Rule.required()
    },
    {
      name: 'slug',
      title: 'Slug',
      type: 'slug',
      options: {
        source: 'name'
      }
    },
    {
      name: 'price',
```

```

        title: 'Price',
        type: 'number',
        validation: Rule => Rule.required().positive()
      },
      {
        name: 'description',
        title: 'Description',
        type: 'text'
      },
      {
        name: 'images',
        title: 'Images',
        type: 'array',
        of: [{ type: 'image' }]
      },
      {
        name: 'category',
        title: 'Category',
        type: 'reference',
        to: [{ type: 'category' }]
      },
      {
        name: 'stock',
        title: 'Stock',
        type: 'number',
        validation: Rule => Rule.required().min(0)
      }
    ]
  }
}

```

Order Schema

```

export default {
  name: 'order',
  title: 'Order',
  type: 'document',
  fields: [
    {
      name: 'orderNumber',
      title: 'Order Number',
      type: 'string',
      validation: Rule => Rule.required()
    },
    {
      name: 'customer',
      title: 'Customer',

```

```

    type: 'reference',
    to: [{ type: 'customer' }]
  },
  {
    name: 'items',
    title: 'Items',
    type: 'array',
    of: [{
      type: 'object',
      fields: [
        {
          name: 'product',
          type: 'reference',
          to: [{ type: 'product' }]
        },
        {
          name: 'quantity',
          type: 'number'
        }
      ]
    }]
  },
  {
    name: 'status',
    title: 'Status',
    type: 'string',
    options: {
      list: [
        'pending',
        'processing',
        'shipped',
        'delivered',
        'cancelled'
      ]
    }
  },
  {
    name: 'totalAmount',
    title: 'Total Amount',
    type: 'number'
  },
  {
    name: 'paymentStatus',
    title: 'Payment Status',
    type: 'string',
    options: {

```

```

        list: [
          'pending',
          'completed',
          'failed',
          'refunded'
        ]
      }
    }
  ]
}

```

5. Implementation Roadmap

Phase 1: Setup & Basic Structure

- Initialize Next.js project
- Set up Sanity CMS
- Implement basic routing
- Create core components

Phase 2: Core Features

- Product listing and details
- Shopping cart functionality
- User authentication
- Basic checkout process

Phase 3: Integration & Testing

- Payment gateway integration
- Order management
- Email notifications
- Testing and bug fixes

6. Best Practices Implementation

Performance Optimization

- Image optimization using next/image
- Lazy loading for product listings
- API response caching
- Code splitting for route-based chunking

Security Measures

- Input validation
- XSS protection
- CSRF tokens

- Secure payment handling

SEO Considerations

- Meta tags management
- Structured data implementation
- Sitemap generation
- robots.txt configuration

7. Quality Assurance Checklist

- ☐ Mobile responsiveness
- ☐ Cross-browser compatibility
- ☐ Performance metrics
- ☐ Security compliance
- ☐ API error handling
- ☐ Loading states
- ☐ Form validation
- ☐ Payment flow testing

8. Monitoring & Analytics

- Implement error tracking (Sentry)
- Set up performance monitoring
- Track user behavior
- Monitor API performance
- Track conversion rates

This documentation serves as a comprehensive guide for implementing the e-commerce marketplace, focusing on maintainability, scalability, and user experience.

E-commerce Data Schema Relationships

```
erDiagram
    PRODUCT ||--o{ ORDER_ITEM : contains
    ORDER ||--|{ ORDER_ITEM : includes
    CUSTOMER ||--o{ ORDER : places
    PRODUCT ||--o{ PRODUCT_CATEGORY : belongs_to
    CATEGORY ||--|{ PRODUCT_CATEGORY : has
    ADDRESS ||--o{ CUSTOMER : has
    CUSTOMER ||--o{ REVIEW : writes
    PRODUCT ||--o{ REVIEW : receives
    ORDER ||--o{ ORDER_STATUS : tracks

    PRODUCT {
```

```

        string id PK
        string name
        float price
        int stock
        string description
        string[] images
        datetime created_at
        datetime updated_at
        boolean is_active
    }

    CATEGORY {
        string id PK
        string name
        string description
        string slug
        string parent_id FK
    }

    PRODUCT_CATEGORY {
        string product_id FK
        string category_id FK
    }

    CUSTOMER {
        string id PK
        string first_name
        string last_name
        string email
        string phone
        datetime created_at
        boolean is_active
    }

    ADDRESS {
        string id PK
        string customer_id FK
        string street
        string city
        string state
        string postal_code
        string country
        boolean is_default
        string type
    }

```



```

ORDER {
    string id PK
    string customer_id FK
    float total_amount
    string payment_status
    datetime order_date
    string shipping_address_id FK
    string billing_address_id FK
}

ORDER_ITEM {
    string id PK
    string order_id FK
    string product_id FK
    int quantity
    float unit_price
    float subtotal
}

ORDER_STATUS {
    string id PK
    string order_id FK
    string status
    string description
    datetime created_at
}

REVIEW {
    string id PK
    string product_id FK
    string customer_id FK
    int rating
    string comment
    datetime created_at
}

```

**** Day 1 ****

I Choose My Marketplace Type

General E-Commerce

1. **Solve a Problem:**
 - Provide a seamless shopping experience with easy navigation and fast checkout.
2. **Target Audience:**
 - Cater to a specific demographic (e.g., tech-savvy millennials, working professionals, or niche communities).
3. **Product Offering:**
 - Offer high-quality, diverse, or niche products that address customer needs.
4. **Competitive Edge:**
 - Focus on affordability, exclusive deals, fast delivery, or exceptional customer service.
5. **Scalability:**
 - Build a scalable platform to accommodate growth in products, customers, and regions.
6. **Brand Recognition:**
 - Establish a trusted brand with strong customer engagement and loyalty.
7. **Sustainability:**
 - Incorporate eco-friendly practices to attract environmentally conscious customers.

These goals will help establish a solid foundation for your e-commerce business.

Reasons for Choosing E-Commerce Marketplace:

1. **Wide Reach:** Access to a global customer base, expanding beyond geographical limitations.
2. **Growing Demand:** Increasing consumer preference for online shopping.
3. **Low Overheads:** Reduced operational costs compared to traditional retail.
4. **Scalability:** Easy to scale operations and add new products or services.
5. **24/7 Availability:** Customers can shop anytime, increasing sales opportunities.
6. **Diverse Revenue Streams:** Monetize through commissions, subscriptions, or advertisements.
7. **Data-Driven Insights:** Leverage customer data for personalized marketing and product recommendations.
8. **Convenience:** Simplifies the buying process for customers, enhancing satisfaction and loyalty.

Prepared By: Azmat Ali

Figure 2: General E-Commerce For Me

I Don't Choose My Marketplace Type

Q-Commerce: Not chosen due to the following risks:

- Potential loss of customer trust if goods are delayed.
- Local environmental factors (e.g., protests, road closures) impacting delivery times.
- Risk of running out of stock and failing to meet customer expectations.

Rental E-Commerce: Not chosen due to the following challenges:

- Risk of customers not returning, breaking, or losing rental items.
- Potential financial loss if customers disappear without settling dues.
- Difficulty in consistently finding reliable customers, leading to idle inventory.

Prepared By: Azmat Ali

Figure 3: Image

Summary and Conclusion

Summary and Conclusion

I outlined a **general e-commerce marketplace** powered by **Next.js 15** and **Sanity CMS**. The business goals center on offering a seamless online shopping experience with efficient management of **customers, orders, products, and delivery zones**.

Data Schema Overview

1. **Customers:** Captures user profiles, preferences, and contact details to personalize the shopping experience.
2. **Orders:** Manages order details, statuses, and payment information for efficient order tracking and fulfillment.
3. **Products:** Catalogs product details, categories, inventory levels, and pricing for a dynamic storefront.
4. **Delivery Zones:** Defines geographic regions, delivery costs, and availability to streamline logistics.

Conclusion

This schema design, integrated with **Next.js** and **Sanity**, ensures scalability, flexibility, and ease of content management for your e-commerce platform. It aligns with your goals of delivering a robust and user-friendly marketplace.

Prepared By: Azmat Ali

Figure 4: Image

Delivery Zones Schema

```
export default {
  name: 'deliveryZone',
  title: 'Delivery Zone',
  type: 'document',
  fields: [
    {
      name: 'zoneName',
      title: 'Zone Name',
      type: 'string',
    },
    {
      name: 'coverageArea',
      title: 'Coverage Area',
      type: 'text',
    },
    {
      name: 'assignedDrivers',
      title: 'Assigned Drivers',
      type: 'array',
      of: [{ type: 'string' }],
    },
  ],
};
```

Prepared By: Azmat Ali

Figure 5: Image

Customers Schema

```
export default {  
  name: 'customer',  
  title: 'Customer',  
  type: 'document',  
  fields: [  
    {  
      name: 'name',  
      title: 'Name',  
      type: 'string',  
    },  
    {  
      name: 'email',  
      title: 'Email',  
      type: 'string',  
    },  
    {  
      name: 'contactInfo',  
      title: 'Contact Info',  
      type: 'string',  
    },  
    {  
      name: 'address',  
      title: 'Address',  
      type: 'text',  
    },  
  ],  
};
```

Prepared By: Azmat Ali

Figure 6: Image

Products Schema

```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    {
      name: 'name',
      title: 'Name',
      type: 'string',
    },
    {
      name: 'description',
      title: 'Description',
      type: 'text',
    },
    {
      name: 'price',
      title: 'Price',
      type: 'number',
    },
    {
      name: 'stock',
      title: 'Stock',
      type: 'number',
    },
    {
      name: 'category',
      title: 'Category',
      type: 'string',
    },
    {
      name: 'image',
      title: 'Image',
      type: 'image',
    },
  ],
}
```

Prepared By: Aamir Ali

Figure 7: Image

Orders Schema

```
export default {
  name: 'order',
  title: 'Order',
  type: 'document',
  fields: [
    {
      name: 'customer',
      title: 'Customer',
      type: 'reference',
      to: [{ type: 'customer' }],
    },
    {
      name: 'orderDate',
      title: 'Order Date',
      type: 'datetime',
    },
    {
      name: 'products',
      title: 'Products',
      type: 'array',
      of: [{ type: 'reference', to: [{
        type: 'product' }] }],
    },
    {
      name: 'totalPrice',
      title: 'Total Price',
      type: 'number',
    },
    {
      name: 'status',
      title: 'Status',
      type: 'string',
      options: {
        list: [
          { title: 'Pending', value:
            'pending' },
          { title: 'Shipped', value:
            'shipped' },
          { title: 'Delivered', value:
            'delivered' },
        ],
      },
    },
  ],
}
```

Prepared By: Azmat Ali

Figure 8: Image
16

× Nextjs Sanity Schema

```
1 marketplace/ # Root folder for the Next.js project
2 |— public/    # Static assets (e.g., images, icons)
3 |— src/       # Source files
4 |   |— pages/ # Next.js pages
5 |   |   |— index.js    # Home page fetching products from Sanity
6 |   |   |— _app.js     # App configuration
7 |   |   |— _document.js # Custom document structure
8 |   |— utils/ # Utility functions
9 |   |   |— sanityClient.js # Sanity client configuration
10 |— sanity/   # Sanity CMS studio folder
11 |   |— schemas/ # Sanity schema definitions
12 |   |   |— customer.js # Customer schema
13 |   |   |— product.js  # Product schema
14 |   |   |— order.js    # Order schema
15 |   |   |— deliveryZone.js # Delivery Zone schema
16 |   |   |— schema.js   # Main schema linking all
17 |   |— .sanity.json    # Sanity project configuration
18 |   |— deskStructure.js # Optional custom desk structure
19 |— node_modules/ # Installed dependencies
20 |— package.json  # Project metadata and dependencies
21 |— sanity.json   # Sanity configuration
22 |— next.config.js # Next.js configuration
23 |— README.md     # Project documentation
24
```

Figure 9: Image