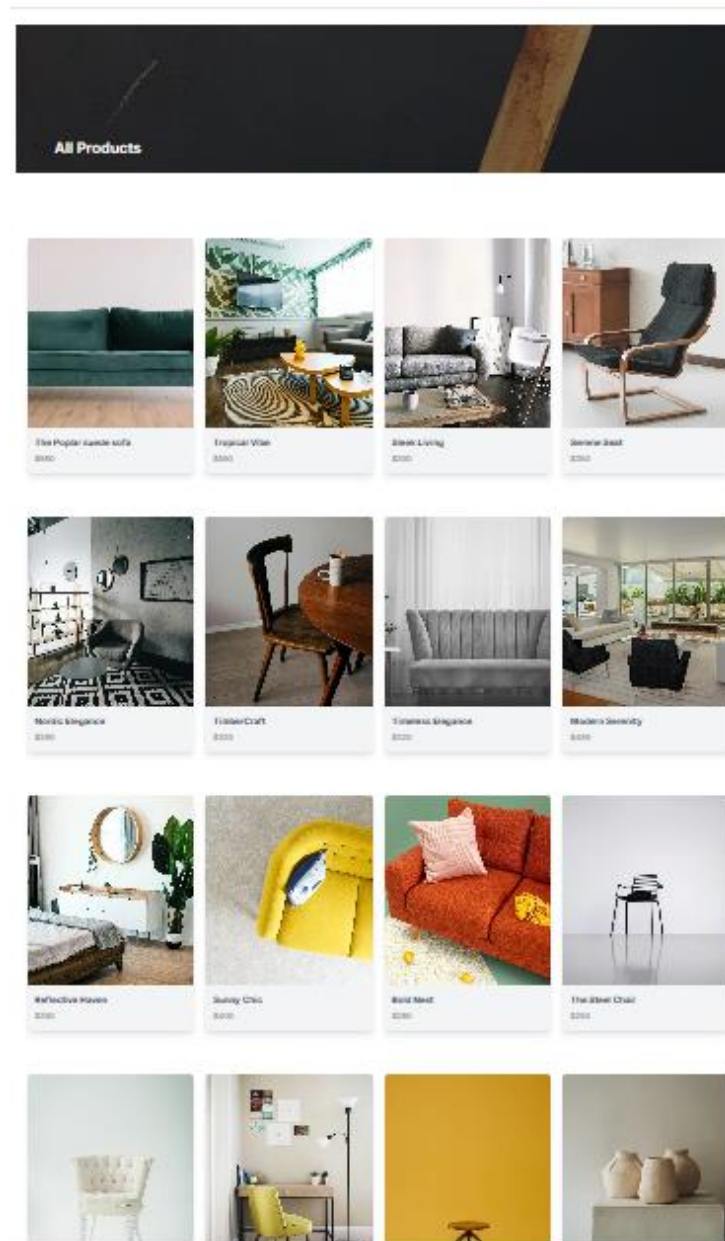
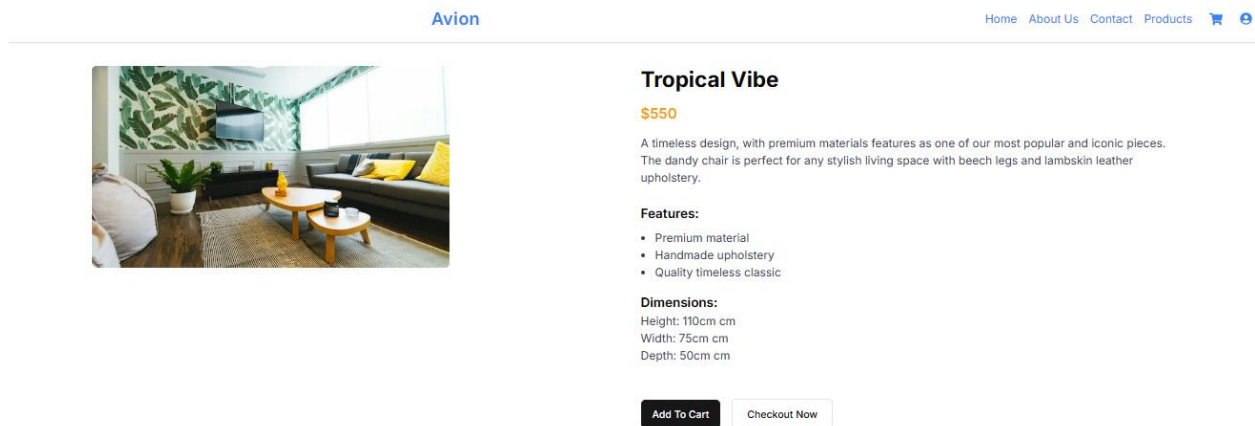


1. Functional Deliverables:

The product listing page with dynamic data Image.

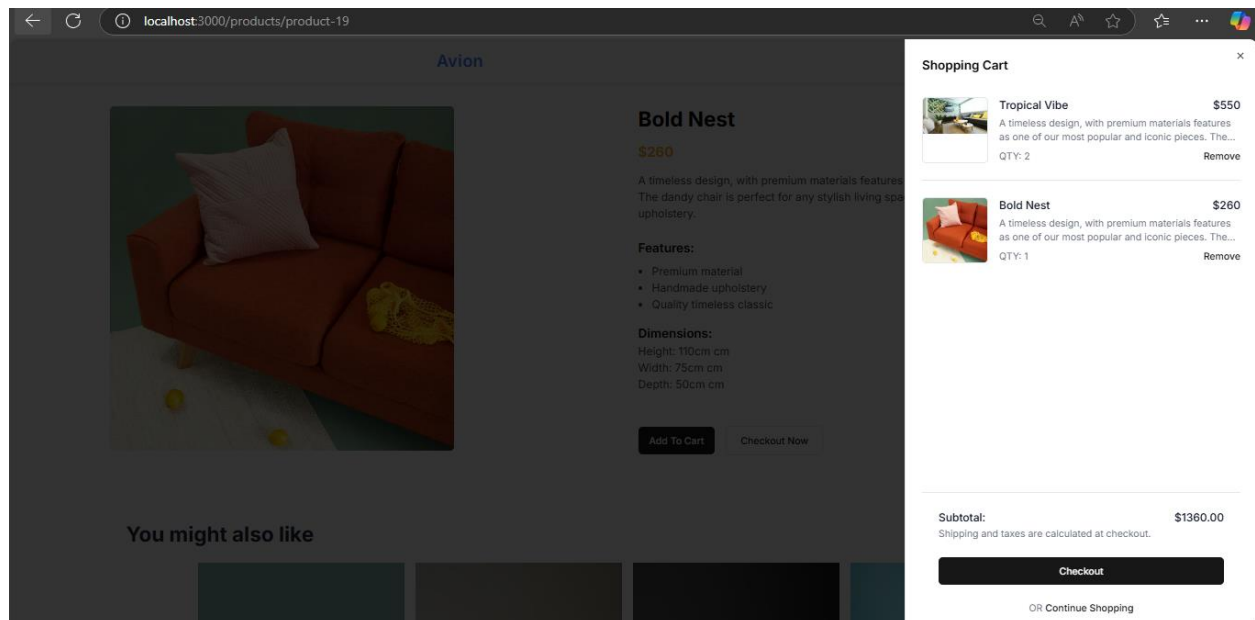


Individual product detail pages with accurate routing and data rendering.



You might also like

Additional features implemented, such as related products or user profile components is [Add to Cart](#), [ShipEngine](#), [Checkout](#), [Payment Method](#) Functionality.



←  TEST MODE

Pay

\$1,360.00



Tropical Vibe

Qty 2

\$1,100.00

\$550.00 each



Bold Nest

Qty 1

\$260.00

Pay with  link

Or pay with card

Email

Card information

1234 1234 1234 1234



MM / YY

CVC



Cardholder name

Full name on card

Billing address

Pakistan



Address line 1

Address line 2

Suburb

City

Postal code

☐

Securely save my information for 1-click checkout

Pay faster on this site and everywhere Link is accepted.

Pay

Avion

[Home](#) [About Us](#) [Contact](#) [Products](#)  



Payment Done!

Thank you for your purchase. We hope you enjoy it.

Have a great day!

Shipping

New business

Home

Balances

Transactions

Customers

Product catalog

Shortcuts

Terminal

Disputes

Radar

Billing overview

Payment Links

Products

Payments

Billing

Reporting

More

25/1/21

Search

Test mode

Payments

This customer doesn't have any chargeable payment sources on file. Add a source or payment method to create a new payment.

| Amount | | | Description | Date | |
|------------|-----|-----------|----------------------------------|------------------|-----|
| \$1,360.00 | USD | Succeeded | 2x Tropical Vibe 1x Bold Nest | Jan 21, 1:18 PM | ... |
| \$1,100.00 | USD | Succeeded | 2x Tropical Vibe | Jan 21, 11:44 AM | ... |
| \$550.00 | USD | Succeeded | 1x Tropical Vibe | Jan 21, 9:00 AM | ... |
| \$700.00 | USD | Succeeded | 2x serene-seat | Jan 21, 8:05 AM | ... |
| \$350.00 | USD | Succeeded | 1x serene-seat | Jan 21, 7:34 AM | ... |
| \$550.00 | USD | Succeeded | 1x Tropical Vibe | Jan 21, 5:32 AM | ... |
| \$260.00 | USD | Succeeded | 1x Bold Nest | Jan 21, 4:23 AM | ... |
| \$980.00 | USD | Succeeded | 1x The Poplar suede sofa | Jan 21, 12:15 AM | ... |
| \$1,100.00 | USD | Succeeded | 2x Tropical Vibe | Jan 21, 12:06 AM | ... |
| \$1,400.00 | USD | Succeeded | 2x Tropical Vibe 1x Sleek Living | Jan 20, 11:56 PM | ... |

10 of 32 results

Insights

Spent

\$26,380.67

Details

Guest details

mjj

mjj@gmail.com

Customer since

Jan 9

Alternative emails

ali@gmail.com

test@gmail.com

xyz@gmail.com

alia@gmail.com

Billing details

street123

street456

karachi-75640-PK

New business

Home

Balances

Transactions

Customers

Product catalog

Shortcuts

Terminal

Disputes

Radar

Billing overview

Payment Links

Products

Payments

Billing

Reporting

More

25/1/21

Search

Test mode

VIEW ALL

STATUS









ACTIVE

CREATE ITEMS

EXPORT PRICES

EXPORT PRODUCTS



EDIT COLUMNS

| Name | Pricing | Created | Updated | |
|---|--------------|---------|---------|-----|
|  The Poplar suede sofa | \$980.00 USD | Jan 20 | Jan 20 | ... |
|  The Steel Chair | \$250.00 USD | Jan 20 | Jan 20 | ... |
|  Rustic Vase Set | \$210.00 USD | Jan 20 | Jan 20 | ... |
|  Bed | \$250.00 USD | Jan 20 | Jan 20 | ... |
|  Wood Chair | \$100.00 USD | Jan 20 | Jan 20 | ... |
|  Retro Vibe | \$340.00 USD | Jan 20 | Jan 20 | ... |
|  Pure Aura | \$280.00 USD | Jan 20 | Jan 20 | ... |
|  The Luckv Lamp | \$200.00 USD | Jan 20 | Jan 20 | ... |

34 results

PreviousNext

Avion

[Home](#) [About Us](#) [Contact](#) [Products](#)  

Shipping Rates Calculator

Ship To Address

| | |
|--|--|
| <input type="text" value="John Doe"/> | <input type="text" value="+1 555-678-1234"/> |
| <input type="text" value="1600 Pennsylvania Avenue NW"/> | <input type="text" value="Address Line 2"/> |
| <input type="text" value="Washington"/> | <input type="text" value="DC"/> |
| <input type="text" value="20500"/> | <input type="text" value="US"/> |

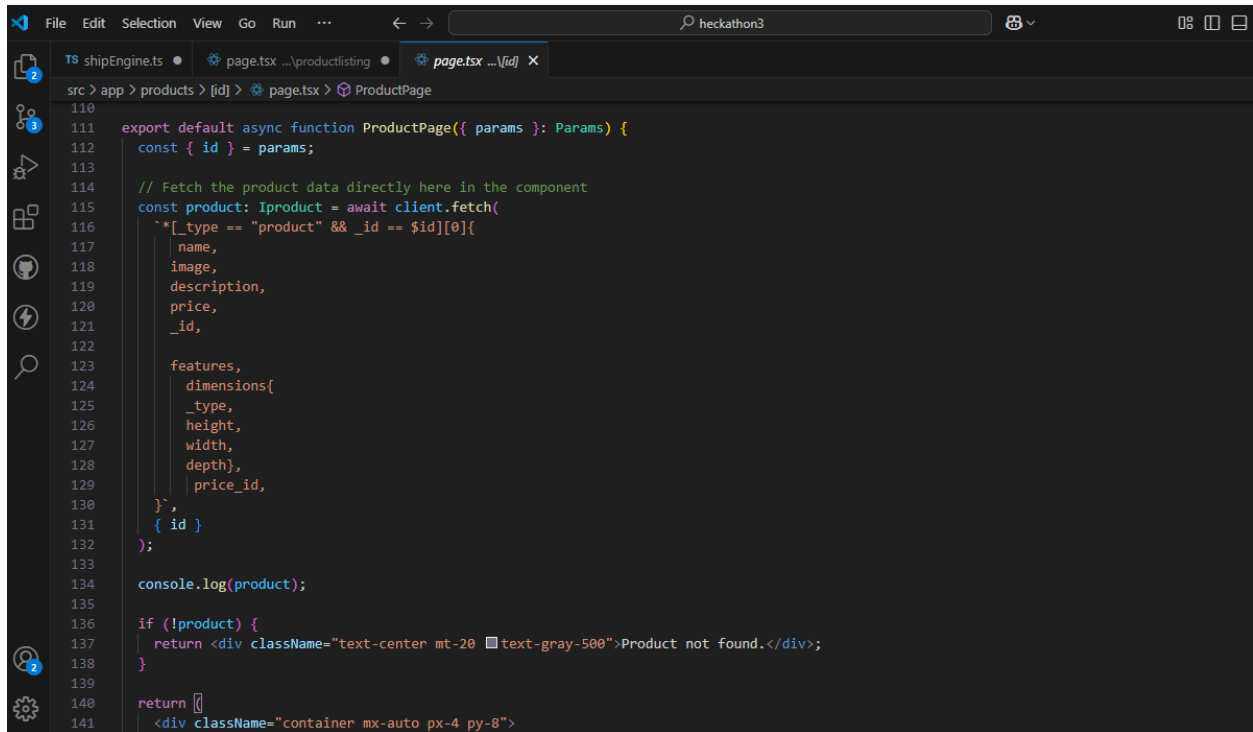
Get Shipping Rates

2. Code Deliverables:

Code snippets for key components (e.g., **Product Card**, **ProductList**).

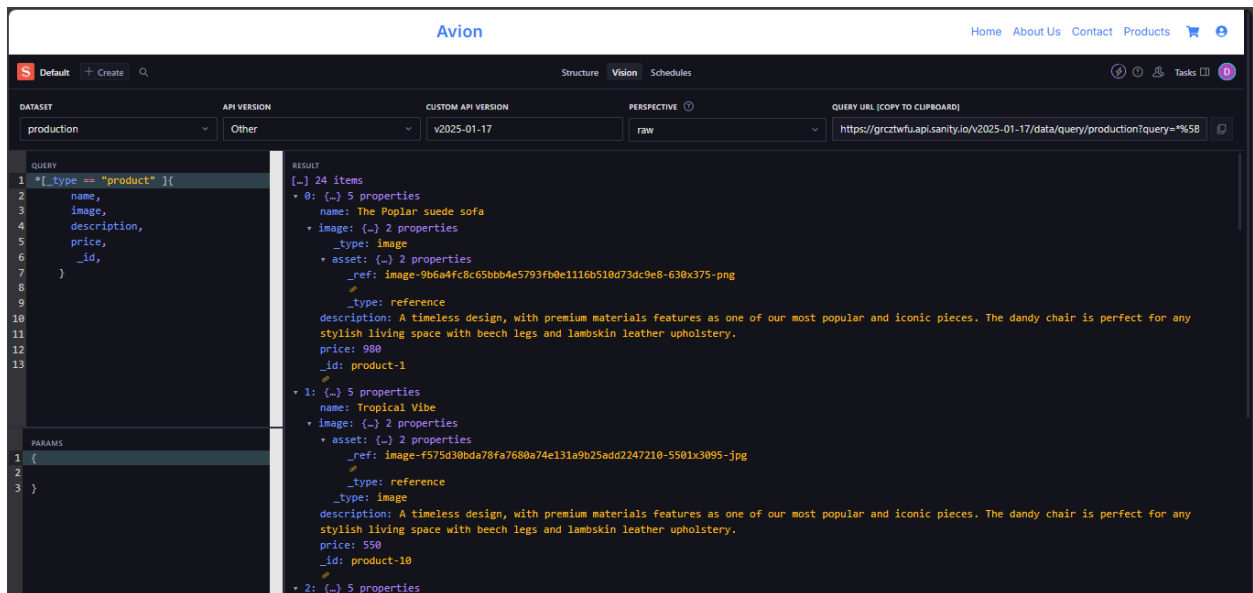
```
File Edit Selection View Go Run ... ← → heckathon3
TS shipEngine.ts • page.tsx •
src > app > productlisting > page.tsx > Product
1 import { client } from '@sanity/lib/client';
2 import { urlFor } from '@sanity/lib/image';
3 import Image from 'next/image';
4 import Link from 'next/link';
5
6 type Product = {
7   _id: string;
8   image: string;
9   name: string;
10  price: string;
11
12 };
13
14 export default async function Ourshop() {
15   const products: Product[] = await client.fetch(`
16
17   *[_type == "product"]{
18     name,
19     image,
20     price,
21     _id
22   }
23
24   `);
25   return (
26     <div className="container mx-auto px-4 sm:px-8 py-8">
27       { /* Hero Section with Image */ }
28       <div className="relative">
29         <div className="relative w-full h-[300px]">
30           <Image
31             src="/productlisting.png" // Replace with dynamic product image URL if needed
32             alt="Product image"

```



```
110
111 export default async function ProductPage({ params }: Params) {
112   const { id } = params;
113
114   // Fetch the product data directly here in the component
115   const product: Iproduct = await client.fetch(
116     `*_type == "product" && _id == ${id}[0]{
117     name,
118     image,
119     description,
120     price,
121     _id,
122
123     features,
124     dimensions{
125       _type,
126       height,
127       width,
128       depth,
129       price_id,
130     },
131     { id }
132   );
133
134   console.log(product);
135
136   if (!product) {
137     return <div className="text-center mt-20 text-gray-500">Product not found.</div>;
138   }
139
140   return (
141     <div className="container mx-auto px-4 py-8">
```

Scripts or logic for API integration and dynamic routing



Avion

Home About Us Contact Products

Default + Create

Structure Vision Schedules

Tasks

| DATASET | API VERSION | CUSTOM API VERSION | PERSPECTIVE | QUERY URL (COPY TO CLIPBOARD) |
|------------|-------------|--------------------|-------------|--|
| production | Other | v2025-01-17 | raw | https://grctwfuapi.sanity.io/v2025-01-17/data/query/production?query=*_type == "product" } |

QUERY

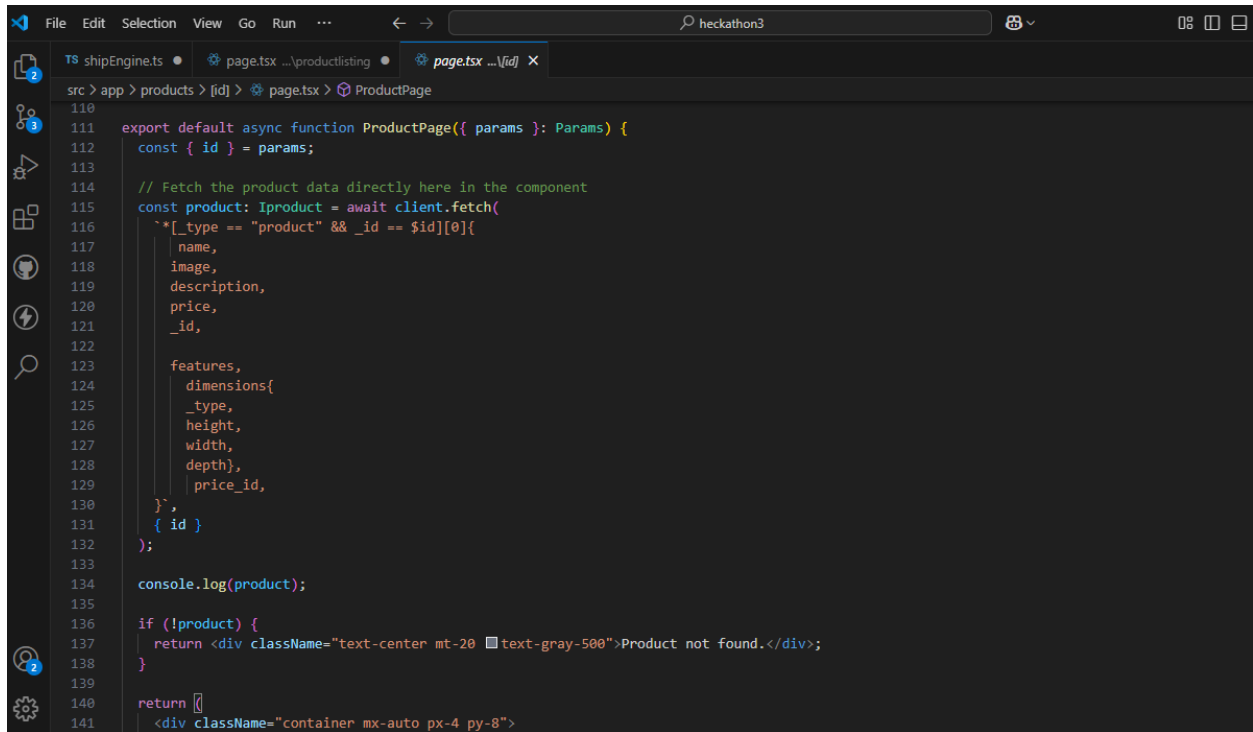
```
1 *_type == "product" ){
2   name,
3   image,
4   description,
5   price,
6   _id,
7 }
8
9
10
11
12
13
```

PARAMS

```
1 {
2
3 }
```

RESULT

```
[_] 24 items
+ 0: {~} 5 properties
  name: The Poplar suede sofa
  image: {~} 2 properties
    _type: image
    asset: {~} 2 properties
      _ref: image-9b6a4fc8c65bbb4e5793fb0e1116b510d73dc9e8-638x375-png
      _type: reference
  description: A timeless design, with premium materials features as one of our most popular and iconic pieces. The dandy chair is perfect for any stylish living space with beech legs and lambskin leather upholstery.
  price: 980
  _id: product-1
+ 1: {~} 5 properties
  name: Tropical Vibe
  image: {~} 2 properties
    _ref: image-f575d30bda78fa7680a74e131a9b25add2247210-5581x3895-jpg
    _type: reference
  _type: image
  description: A timeless design, with premium materials features as one of our most popular and iconic pieces. The dandy chair is perfect for any stylish living space with beech legs and lambskin leather upholstery.
  price: 550
  _id: product-10
+ 2: {~} 5 properties
```



```
110
111 export default async function ProductPage({ params }: Params) {
112   const { id } = params;
113
114   // Fetch the product data directly here in the component
115   const product: Iproduct = await client.fetch(
116     `*_type == "product" && _id == ${id}[0]{
117     name,
118     image,
119     description,
120     price,
121     _id,
122
123     features,
124     dimensions{
125       _type,
126       height,
127       width,
128       depth,
129       price_id,
130     },
131     { id }
132   );
133
134   console.log(product);
135
136   if (!product) {
137     return <div className="text-center mt-20 text-gray-500">Product not found.</div>;
138   }
139
140   return (
141     <div className="container mx-auto px-4 py-8">
```

Documentation:

A technical report summarizing: -

1. Introduction

This technical report outlines the development process for creating and integrating components of a fully functional e-commerce website. The website is designed to handle various tasks such as product listings, user authentication, order management, and payment processing, all with a seamless user experience. The development process focuses on scalability, performance, and security.

2. Steps taken to build and integrate components

1.1 Frontend Development:

Technology Stack:

- **Next.js:** The foundation of the frontend, providing server-side rendering (SSR), static site generation (SSG), and optimized routing to ensure fast performance and great SEO.
- **TypeScript:** Introduced for type safety and better code maintainability, ensuring early error detection and providing a more structured development process.
- **Tailwind CSS:** A utility-first CSS framework chosen for rapid and responsive design implementation, allowing us to style components quickly and efficiently with minimal custom CSS.
- **Sanity:** A flexible, headless CMS used for managing and retrieving dynamic content such as product data, categories, and more.

Integration:

- **API Calls:** RESTful APIs were integrated using Axios to fetch data such as product listings and user profiles.
- **Routing:** React Router was employed to handle different page navigation like product details, shopping cart, and checkout.

1.2 Backend Development:

Server Setup:

- **Framework:** Node.js were chosen for server-side development due to their flexibility and scalability.

- **API Design:** The backend APIs were designed to handle operations like user registration, login, product searches, and order management.
- **Authentication:** JSON Web Tokens (JWT) were used to authenticate users securely across sessions.

Integration:

- **Database Communication:** Mongoose was used for interacting with MongoDB to store user and product data.
- **Error Handling:** Global error handling was implemented in Express to catch and manage potential failures during API calls.

1.3 Integration of Payment Gateway:

- **Payment Processing:** Stripe was integrated for handling payments securely. The backend was configured to communicate with the Stripe API for payment verification and order processing.
- **Order Flow:** After payment success, an order confirmation API was triggered, and the order details were stored in the database.

1.4 Database Design and Integration:

- **Database Choice:** MongoDB was selected for its flexibility and scalability, especially in handling large amounts of user data and product information.
- **Schema Design:** Custom schemas were created for users, orders, and products. Relationships between products and orders were defined, and the database was optimized for read-heavy operations.

2. Challenges Faced and Solutions Implemented

2.1 Performance Optimization:

Challenge: As the number of users and product listings grew, website load times began to increase, negatively affecting user experience. **Solution**

- Implemented lazy loading for images and dynamic imports to reduce initial page load time.

- Utilized caching strategies with Redis to store frequently accessed data, reducing database load and improving speed.

2.2 Scalability Issues:

Challenge: Scaling the website to handle peak traffic times was a concern, especially during sales events. **Solution:**

- Adopted horizontal scaling by deploying multiple instances of the application and database across different servers.
- Used a load balancer to distribute traffic evenly, ensuring high availability.

2.3 Security Concerns:

Challenge: Securing sensitive user information like passwords and payment details was paramount. **Solution:**

- Passwords were hashed using bcrypt before being stored in the database.
- Integrated HTTPS across the website for secure data transmission.
- Used rate-limiting to prevent brute-force attacks on the login API.

2.4 User Authentication:

Challenge: Managing secure authentication across various user sessions and devices. **Solution:**

- Utilized JWT tokens to handle authentication, with refresh tokens for extended sessions.
- Implemented a secure "logout" feature to invalidate tokens upon user request.

3. Best Practices Followed During Development:

3.1 Code Modularization and Reusability:

- **Component-Based Architecture:** The frontend was designed with reusable components in mind, promoting modularity.

- **Backend Services:** Backend services were separated into modules, each responsible for a specific functionality (e.g., product management, user management).

3.2 Responsive Design:

- **Mobile-First Approach:** The website was designed with a mobile-first approach, ensuring that it provides a great user experience across all devices.
- **Flexbox and Grid Layouts:** These CSS tools were employed to create adaptive and flexible layouts that work across various screen sizes.

3.3 Performance Testing and Monitoring:

- **Continuous Testing:** Load testing was performed using tools like Apache JMeter to simulate high-traffic scenarios.
- **Real-Time Monitoring:** Integrated with tools like Google Analytics and New Relic to track user interactions and monitor the health of the application.

3.4 Regular Code Reviews

- **Collaborative Reviews:** Code reviews were conducted regularly to ensure that the code was of high quality, easy to maintain, and free from potential bugs.
- **Static Code Analysis:** We used ESLint and Prettier to maintain consistent coding styles across the team.