Ethics Number : 52892

React SEO Friendly E-Commerce Site - Handling SEO concerns with Single Page Applications

A REPORT SUBMITTED TO MANCHESTER METROPOLITAN UNIVERSITY
FOR THE DEGREE OF BACHELOR OF SCIENCE
IN THE FACULTY OF SCIENCE AND ENGINEERING

May 2023

By Danny Kettle
Department of Computing and Mathematics

Contents

List of Figures

Abstract

The abstract is a formal description of the reason for doing this work, the methods used and results found, and the conclusions drawn. It should not be more than one page in length.

Declaration

No part of this project has been submitted in support of an application for any other degree or qualification at this or any other institute of learning. Apart from those parts of the project containing citations to the work of others, this project is my own unaided work. This work has been carried out in accordance with the Manchester Metropolitan University research ethics procedures, and has received ethical approval number Your EthOS Number.

Signed: Danny Kettle

Date: 19/05/2023

Acknowledgements

- Ismail Adenira

Chapter 1

Introduction

As online shopping continues to dominate the retail market, businesses are recognizing the increasing importance of having a strong online presence. Search Engine Optimization (SEO) plays a crucial role in this, as it directly affects a website's visibility on search engines. Single Page Applications (SPAs) have become popular in recent years for their ability to provide a seamless user experience by dynamically updating content without requiring a page refresh. However, SPAs also present unique challenges for SEO, particularly for ecommerce sites where fast load times, real-time inventory updates, and personalized product recommendations are critical for user engagement and sales.

To address these challenges, this paper explores how React, a popular front-end JavaScript library for building SPAs, can be optimized for SEO. Specifically, will examine the use of Server-Side Rendering (SSR) in Next.js, a framework for building React-based web applications, to improve initial load times and boost SEO ranking. Finally, will explore the use of routing masks to make URLs more SEO friendly, ultimately providing readers with a comprehensive guide to creating an SEO-friendly React SPA.

1.1 Aims

The Aim of this project can be separated into these points:

- Understand the importance of SEO for online businesses and how it can impact website visibility on search engines.
- Explore the unique challenges that SPAs, particularly ecommerce sites, pose for SEO.
- Understand design choices for SEO friendliness
- Analyze the use of React, a popular front-end JavaScript library for building SPAs, in optimizing websites for SEO.
- Discuss the benefits of using Server-Side Rendering (SSR) in Next.js to improve initial load times and SEO ranking.
- Investigate the use of routing masks to create more SEO-friendly URLs.
- Provide readers with a comprehensive guide on creating an SEO-friendly React SPA that addresses these challenges and implements these solutions.

1.2 Objectives

The report objectives outlined below, are deemed a necessity for the successful completion of this project:

- Conduct research on various development techniques that optimize SEO for SPAs.
- Develop a visually appealing and user-friendly e-commerce SPA using React.
- Implement access levels within the website to enhance its sustainability and security.
- Integrate a payment gateway to enable seamless and secure transactions.
- Conduct a website crawl to identify the indexed pages and ensure SEO compatibility.
- Evaluate the developed application and analyze its performance against the defined metrics.
- Conclude the feasibility of building SPA-based e-commerce sites as a functional and marketable solution.

1.3 Report Structure

The report is structured into five chapters, each addressing specific aspects of the project. Here is a brief overview of each chapter:

Chapter 2 Literature Review

This chapter provides a comprehensive review of relevant literature, theories, and existing research related to optimizing React SPAs for SEO. It explores the current state of the field and identifies key strategies and best practices.

Chapter 3 Design

In this chapter, the design phase of the project is discussed. It outlines the architectural decisions, system requirements, and design principles used in developing the SEO-friendly SPA. It may also cover the selection of tools, frameworks, and technologies.

Chapter 4 Implementation

The implementation chapter focuses on the practical aspects of building the Ecommerce SPA. It covers the development process, coding techniques, and the integration of SEO optimization features.

Chapter 5 Evaluation

This chapter evaluates the performance and effectiveness of the SEO optimization measures implemented in the Ecommerce SPA. It discusses the methodology used for evaluation, analyzes key metrics, and presents the findings. It may also compare the results with initial goals and discuss any limitations or areas for improvement.

Chapter 6 Conclusion

The final chapter concludes the report by summarizing the key findings, lessons learned, and the overall success of the project. It reflects on the achievements, discusses the implications of the research, and provides recommendations for future work or enhancements.

Each chapter contributes to a comprehensive understanding of the project, from the initial research to the final evaluation and conclusion.

Chapter 2

Literature Review

2.1 Search Engine Optimization (SEO)

In this section, the origins, concept, and benefits of SEO is explored by introducing why SEO started and where it currently stands, why it is important for websites and in particular ecommerce sites and showcasing the main elements of SEO.

2.1.1 What is Search Engine Optimization (SEO)

SEO (Search Engine Optimization) is a strategy that involves optimizing a website's content and structure to improve its visibility on search engines. When a website is easily discoverable on search engines, it is more likely to attract visitors, which can potentially lead to a growing customer or client base. The ultimate objective of SEO is to bring in a steady stream of visitors that convert into loyal customers or followers.

The concept of SEO was first introduced in the mid-1990s, with the emergence of search engines such as Yahoo and Google. However, it was not until the early 2000s that SEO began to gain widespread popularity as businesses recognized its potential to drive traffic and revenue.

The origins of SEO can be traced back to the early days of the internet when website owners began to realize the importance of optimizing their sites for search engines. One of the earliest pioneers of SEO was Danny Sullivan, who founded Search Engine Watch, an online publication that covered the search engine industry.

(HubSpot. (n.d.). A Brief History of Search & SEO)

Over the years, modern SEO has undergone significant changes, and today it involves a wide array of strategies and techniques aimed at enhancing website visibility and traffic. One of the remarkable advancements in recent times is the capability of search engines to read and index JavaScript (JS) pages, which was once a significant challenge for SEO professionals. With the growth of single page applications (SPAs) constructed using JavaScript frameworks like React and Angular, search engines faced difficulties in crawling and indexing these pages, which led to poor search engine rankings and visibility. However, contemporary search engines, including Google, have made substantial progress in understanding and indexing JS pages, enabling SPAs to achieve good search result rankings. Consequently, this development has opened up new opportunities for businesses to create dynamic, complex websites using JS frameworks while still adhering to good SEO practices.

(Moz. (n.d.). Beginner's Guide to SEO)

2.1.2 Why is SEO important for ecommerce sites?

Search Engine Optimization (SEO) plays a crucial role in the success of all websites, particularly for ecommerce sites. As per a source, approximately 50% of all website traffic is driven by search engines such as Google and Bing. People often use search engines to find products or services they need, making SEO an essential aspect of website design. SEO aims to improve a website's visibility on search engines and drive more traffic to it.

(HubSpot. (n.d.). A Brief History of Search & SEO)

The key benefit of SEO is that it's a sustainable marketing channel - if it's implemented correctly, it can continue to drive traffic and engagement over time, unlike paid advertising campaigns which may decrease or stop entirely (Moz., n.d.).

(Moz. (n.d.). Why is SEO Important for Business?)

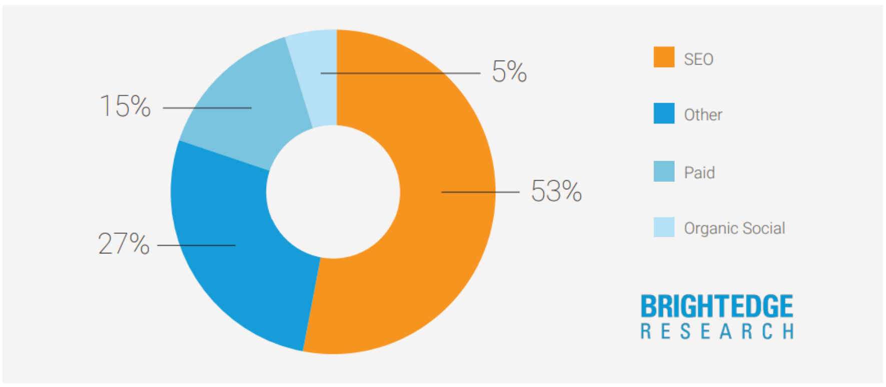


Figure 1

Figure 1. – Web Traffic Pie Graph (Brightedge Research, 2019)

2.1.3 Main dependents of SEO

Search Engine Optimization is a multifaceted process that involves various factors affecting a website's visibility and search engine ranking. Although search engines don't disclose the exact algorithm, they use to rank sites, SEO experts have identified essential elements that can significantly impact a website's performance. These factors include both on-page and off-page elements, such as website content, keywords, links, site structure, user experience, and social signals.

Here is a list of the main elements which effect SEO:

- Content quality and relevance to user search intent.
- Website structure, including internal linking and URL structure.
- Page speed and overall website performance.
- Mobile-friendliness and responsive design.
- Proper use of title tags, meta descriptions, and header tags.
- Keyword research and strategic use of keywords throughout the website.
- Backlink profile, including the number and quality of external links pointing to the site.
- Social media presence and engagement, including shares and likes of website content on social media platforms.
- User experience, including site navigation, layout, and accessibility.
- Technical SEO factors, such as site security, XML sitemap, and canonical tags.

("SEO Basics: A Beginner's Guide to SEO")

Please note that some of these factors are not directly linked to application development, and while I will acknowledge them in this paper, I will not provide detailed information about them.

2.2 Single Page Applications (SPAs)

This section explores the origins, benefits, and challenges of Single Page Applications (SPAs). It introduces what SPAs are and their advantages, particularly for ecommerce sites. The section also highlights the challenges of developing and implementing SPAs, along with potential solutions.

2.2.1 What is a Single Page Application (SPA)?

SPAs are web applications that load a single HTML page and update the page content dynamically from user actions and data received from remote APIs. They provide a smooth and seamless user experience that does not require page refreshing, allowing the user to interact with website whilst it updates its content. Overall, improving load times (after initial load), increasing user engagement, and satisfying the user with an uninterrupted experience.

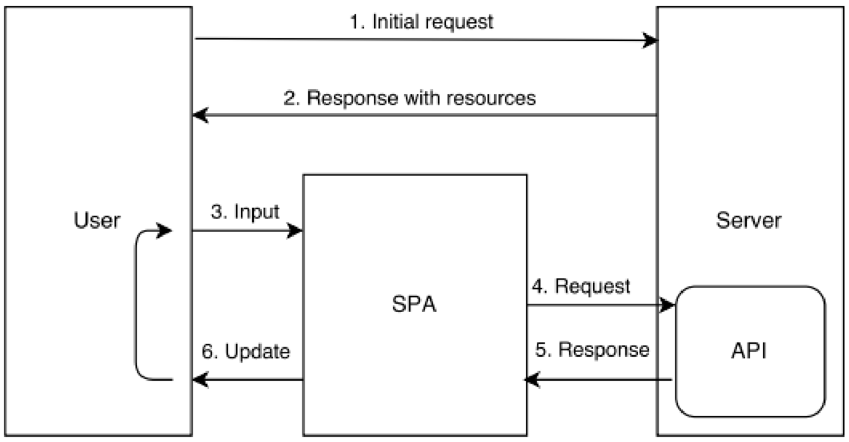


Figure 2

Figure 2 illustrates a typical communication of an SPA between a user and a web server (Molin, 2016).

2.2.2 Origins of Single Page Applications (SPAs)

Single Page Applications (SPAs) originated from the need to provide users with a seamless and interactive browsing experience without requiring the page to reload for every interaction. This concept emerged in the mid-2000s, with the advent of AJAX (Asynchronous JavaScript and XML) technology that allowed websites to retrieve data without requiring a page refresh. SPAs further evolved with the introduction of modern JavaScript frameworks, such as AngularJS and React, that enabled the creation of dynamic and highly responsive web applications.

(Mikowski, T., & Powell, J. (2013). Why single page web applications? In Single page applications: JavaScript end-to-end (pp. 1-14))

2.2.3 Advantages of SPAs in ecommerce sites

Single Page Applications (SPAs) provide significant benefits for ecommerce sites. By allowing new content to load without requiring page reloads, SPAs offer a seamless and fast user experience that improves site navigation, helps users find products easily, and enables quicker purchases. This enhanced user experience can lead to increased customer satisfaction and loyalty.

In addition, SPAs offer faster page load times compared to traditional multi-page applications, resulting in reduced wait times and improved page performance. This is particularly important for ecommerce sites where fast load times and quick access to product information and checkout are expected, allowing ecommerce sites to provide an overall better shopping experience for their customers.

SPAs can also lead to increased conversion rates for ecommerce sites. The improved user experience and faster page load times can help users find products easily and complete purchases quicker, as users are more likely to complete purchases when the checkout process is fast and responsive. Even a one-second delay in page load time can result in a 7% reduction in conversions for ecommerce sites (Akamai, 2017).

(Akamai. (n.d.). Akamai Releases Spring 2017 State of Online Retail Performance Report)

Finally, SPAs can be easily integrated with APIs, allowing ecommerce sites to access data from external sources such as product databases, payment gateways, and shipping providers. This makes it easier to manage inventory, process payments, and handle shipping and logistics. Easy integration with APIs can help ecommerce sites streamline their operations and provide a better overall shopping experience for their customers.

Overall, Single Page Applications (SPAs) offer a range of benefits for ecommerce sites, including an improved user experience with faster load times that help customers navigate the site quickly and complete purchases more efficiently. This can lead to increased customer satisfaction and loyalty, as well as higher conversion rates. Additionally, SPAs can be easily integrated with APIs, allowing for streamlined operations and better access to external data sources. With even a one-second delay in page load times resulting in a 7% reduction in conversions for ecommerce sites, adopting SPAs can be crucial for success in the competitive online retail market.

2.2.4 Challenges and Solutions of SEO in SPAs

Slow initial load time

Description Of Problem

The nature of SPAs relies on heavy API traffic, in other words the site requires data from persistent storage like a database to populate the page with content, whereas MPAs will have each page’s content within the root directory.

They also contain lots of files since the applications are modularized to improve scalability. Therefore, the site will have multiple CSS and JS files. Which will have to be downloaded to the client.

Both of these problems affect the SEO of the website as they increase the load time for the initial load. In turn worsening the SEO ranking, traffic, and performance of the ecommerce site.

There is however a solution called Server-Side Rendering, where it can combine all the API requests in one big request. The server can make rendering faster by saving the results of rendering specific parts or whole pages and using a serialized version of the parameters that affect the output. This is especially useful for parts or pages that will be viewed many times by the same or different users. If done correctly, this strategy can also help reduce load on the API server by setting a TTL for cached component renders.

(Konshin, 2018, pp 12-14)

Server-Side Rendering (SSR) is a technique that solves the issue of slow page loading times for Single Page Applications (SPAs). However, implementing SSR can also create a new problem of increasing server connections. When an application implements SSR, it opens a new connection to the database for each page instance, which can cause an increase in server costs.

To improve performance and reduce server costs, it is ideal for the application to use context functionality where data can be retrieved once and distributed among modules. However, contexts are client-side processes and are called after the server-side has loaded. This means that if content from the database needs to be crawled, a connection to the database must be established inside the server-side rendering.

In Summary, for the pages which contain database data which need to be called the app needs to establish a connection using Server-Side rendering. Then to improve performance and server costs it should use a context for the other pages and use the implementation of memos and callbacks to make sure these contexts are only re-rendered when the data changes.

Practical Solution

After conducting research, it has been determined that the "getServerSideProps" function in Next.js is the most effective technique for implementing server-side rendering within an ecommerce website. This is primarily due to the dynamic nature of ecommerce websites, which frequently require real-time updates to maintain accurate data such as prices and stock levels. The "getServerSideProps" approach executes on each server request, ensuring that data remains up-to-date.

(Next,js, "Get Server-side Props")

However, this approach comes with a potential drawback of increasing database connections. Since ecommerce websites often have multiple pages that require real-time content from the database, this can result in a significant increase in database connections, which can be costly.

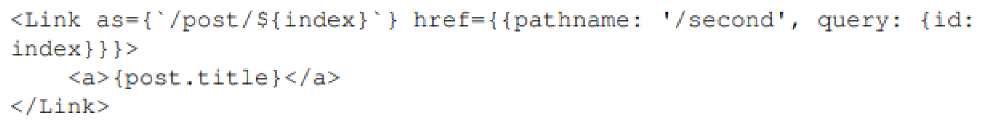
To mitigate this issue, a technique known as connection pooling can be employed. Connection pooling involves maintaining a pool of connections that can be passed between database operations, rather than opening and closing a new connection for each operation. This reduces the cost of opening and closing connections and can improve the performance and scalability of the ecommerce website.

In summary, the "getServerSideProps" function in Next.js is the most suitable technique for implementing server-side rendering in an ecommerce website due to the dynamic nature of the content. To address the issue of increasing database connections, connection pooling can be utilized to maintain a pool of connections that can be passed between database operations.

(Cockroach Labs, "What is Connection Pooling?").

Dynamic Routing

Another challenge associated with Single Page Applications (SPAs) is the unfriendly URLs which have a negative impact on search engine optimization (SEO). This issue arises due to the dynamic nature of the pages which require the inclusion of parameters such as IDs to determine the content displayed. Consequently, URLs for such pages appear in the form of ‘http://localhost:3000/products?id=0’ which does not provide good context for search engine crawlers to interpret the content of the page. To overcome this challenge, routing masks can be used. In Next.js, a special “as” prop of the Link component can be used to create routing masks.



```
<Link as={`/post/${index}`} href={{pathname: '/second', query: {id:
index}}}>
    <a>{post.title}</a>
</Link>
```

Figure 3

Figure 3 : The use of ‘as’ property for route masking (Konshin, 2018, pp 27-31)

To resolve this issue, it is necessary to create a custom server that listens for the masked URL and redirects the client to the actual page. This is because the URL masking technique only works on the client side, and refreshing the page would result in a 404 error. By creating a custom server, it can ensure that the server-side rendering is properly handled and the client is directed to the correct page even after refreshing the page. This can be achieved by using frameworks like Next.js, which allows for custom server-side logic to be implemented easily.

. (Konshin, 2018, pp 27-31)

Browser Compatibility

Single Page Applications (SPA’s) are prone to browser compatibility issues. Different web browsers may interpret JavaScript code differently, which can lead to inconsistencies in how the SPA is displayed or how certain features function.

To mitigate these compatibility issues, web developers may need to perform extensive testing across multiple browsers to ensure consistent functionality and display. This can be time-consuming and costly. One solution to this problem is to use a framework like Next.js. Next.js offers server-side rendering, which allows for improved browser compatibility and faster initial load times, as well as other benefits such as automatic code splitting and built-in optimizations. By using Next.js, web developers can improve the user experience and reduce the risk of browser compatibility issues.

2.3 Conclusion

The research shows that there are several ways to approach the development of a Next.js ecommerce site that can improve its performance, user experience, and security. One way is to use server-side rendering (SSR), which can significantly improve the loading speed and SEO of the site. Additionally, using React components and hooks can make the development process easier and ensure code reusability and maintenance. Integrating with a payment gateway API like Stripe can provide a secure and seamless checkout experience, while ensuring the site is mobile-responsive and adheres to accessibility best practices can improve usability for all users.

As search engines continue to improve in indexing and crawling JavaScript pages, using single-page applications (SPAs) for ecommerce sites is becoming more popular. SPAs can provide faster and more responsive browsing experiences, reduce server load, and enhance scalability, while also utilizing modern web technologies like React, Next.js, and GraphQL to offer rich user interfaces and interactive features. However, it's important to consider ethical concerns around data privacy and security when using SPAs, as they rely on client-side code execution and API calls that may expose sensitive user information to potential attacks. Implementing proper security measures like using HTTPS, encrypting sensitive data, and validating user input can mitigate these risks.

References

- Next.js. (n.d.). Get Server-side Props. Retrieved from https://nextjs.org/docs/basic-features/data-fetching/get-server-side-props

- Cockroach Labs. (n.d.). What is Connection Pooling? Retrieved from https://www.cockroachlabs.com/blog/what-is-connection-pooling/

- Akamai. (n.d.). Akamai Releases Spring 2017 State of Online Retail Performance Report. Retrieved from https://www.akamai.com/newsroom/press-release/akamai-releases-spring-2017-state-of-online-retail-performance-report

- Ahrefs. (n.d.). SEO Basics: A Beginner's Guide to SEO. Retrieved from https://ahrefs.com/blog/seo-basics/

- Moz. (n.d.). Why is SEO Important for Business? Retrieved from https://moz.com/learn/seo/why-seo-important

- Mikowski, T., & Powell, J. (2013). Why single page web applications? In Single page web applications: JavaScript end-to-end (pp. 1-14). Manning Publications.

- Konshin, K 2018, *Next. js Quick Start Guide : Server-Side Rendering Done Right*, Packt Publishing, Limited, Birmingham. Available from: ProQuest Ebook Central.

- Google Developers. 2021. JavaScript SEO basics. Accessed May 2, 2023. https://developers.google.com/search/docs/guides/javascript-seo-basics.

- Moz. (n.d.). Beginner's Guide to SEO. Retrieved May 2, 2023, from https://moz.com/beginners-guide-to-seo.

- Molin, E., 2016. Comparison of single-page application frameworks. A method of how to compare Single-Page Application frameworks written in JavaScript.

- HubSpot. (n.d.). A Brief History of Search & SEO. Retrieved from https://blog.hubspot.com/marketing/a-brief-history-of-search-seo

Chapter 3

Design

3.1 Introduction

In this chapter, the design process of an SEO ecommerce Single Page Application (SPA) for a business is discussed. The design approach focuses on creating a visually appealing interface with a simple and intuitive user experience that encourages visitors to browse and make purchases. The database design includes tables for products, customers, orders, and payments. Various use cases, such as the checkout process and search functionality, were considered to ensure a seamless user experience. Furthermore, SEO techniques were implemented to improve visibility and attract more organic traffic.

3.2 Design Approach

When designing an ecommerce platform, it is important to consider preferences and needs of the users. This section will showcase the design approach for the website, focusing on key design principles, user experience considerations, and the visual design elements used to create an engaging user interface.

3.2.1 Research and Analysis

For research and analysis of designs, the author examined two websites, namely Amazon.com and Petsmart.com. Both websites demonstrate a clean and modern design aesthetic with easy navigation, abundant product information, and the use of large images and clear fonts to create a visually appealing user interface.

In the case of Petsmart.com, however, users are required to navigate through multiple screens of content before reaching the product selection. Considering that users typically spend no more than 15 seconds on a site, it is crucial to showcase the catalogue early in the site's layout to capture their attention. A zoomed-out picture of Petsmart.com's website is provided below for reference.

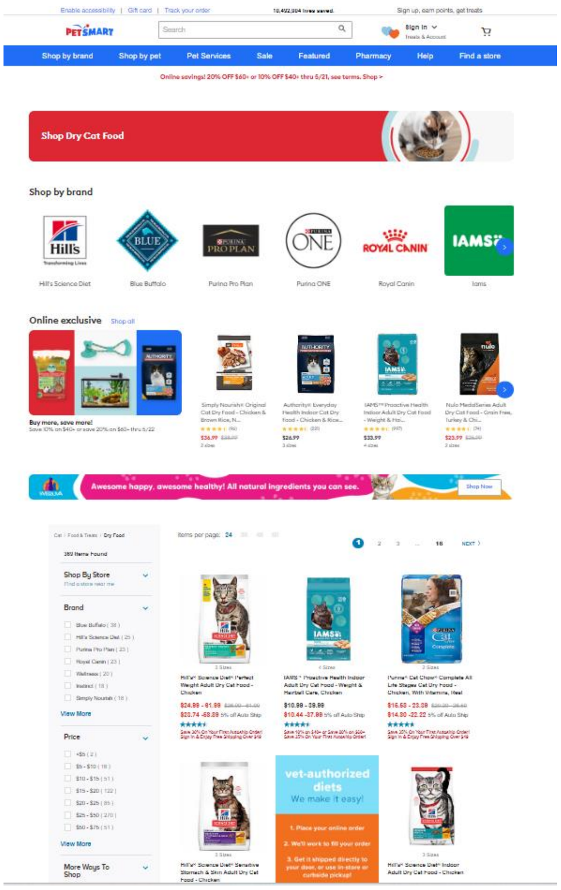
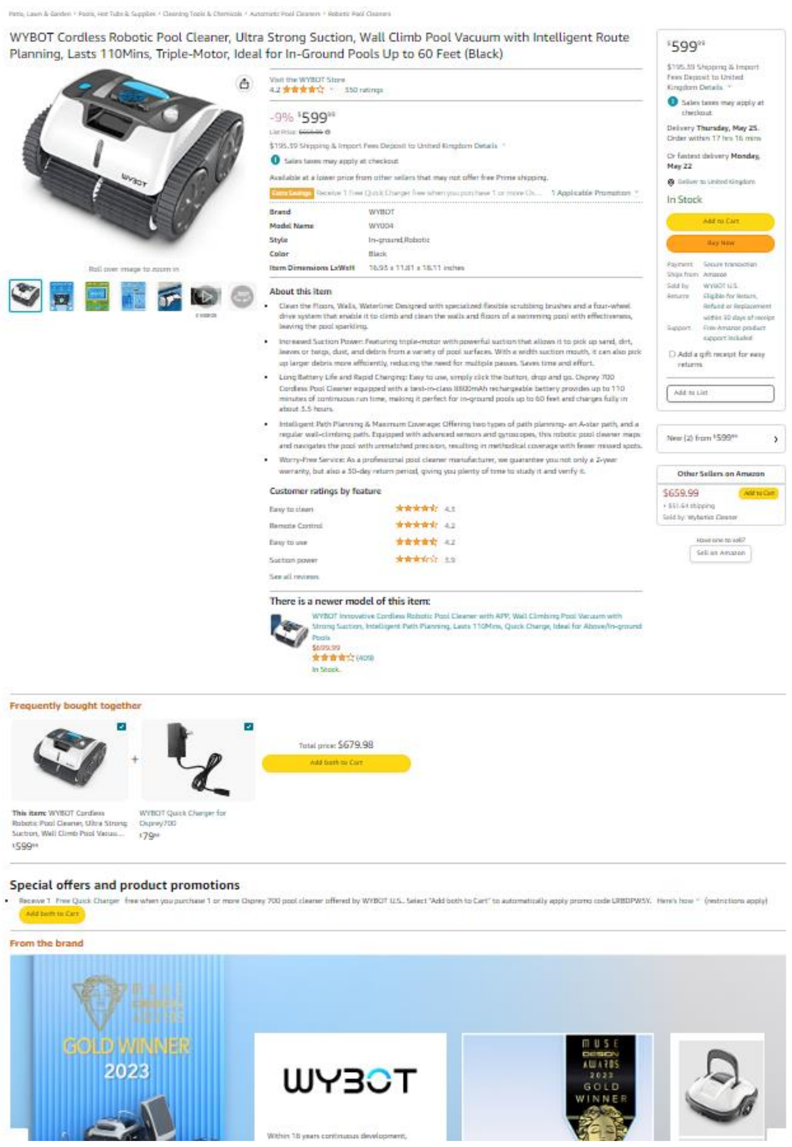


Figure 4

Figure 4 Screenshot of Petsmar homepage, Petsmart. (2021). Pet Supplies, Accessories and Products Online. Petsmart. https://www.petsmart.com/.

It was also concluded that amazon.com is very cluttered, the site has far too much content, which is great for a demographic which wants to know the intricacies of each product, but will overwhelm most users.



Patio, Lawn & Garden › Pools, Hot Tubs & Supplies › Cleaning Tools & Chemicals › Automatic Pool Cleaners › Robotic Pool Cleaners

WYBOT Cordless Robotic Pool Cleaner, Ultra Strong Suction, Wall Climb Pool Vacuum with Intelligent Route Planning, Lasts 110Mins, Triple-Motor, Ideal for In-Ground Pools Up to 60 Feet (Black)

Visit the WYBOT Store
4.2 ★★★★☆ 550 ratings

-9% $599.99
List Price: [illegible]
$195.39 Shipping & Import Fees Deposit to United Kingdom Details
Sales taxes may apply at checkout
Available at a lower price from other sellers that may not offer free Prime shipping.
Extra Savings Receive 1 Free Quick Charger free when you purchase 1 or more Os... 1 Applicable Promotion

Brand	WYBOT
Model Name	WY004
Style	In-ground,Robotic
Color	Black
Item Dimensions LxWxH	16.93 x 11.81 x 18.11 inches

Roll over image to zoom in

About this item

- Clean the Floors, Walls, Waterline: Designed with specialized flexible scrubbing brushes and a four-wheel drive system that enable it to climb and clean the walls and floors of a swimming pool with effectiveness, leaving the pool sparkling.
- Increased Suction Power: Featuring triple-motor with powerful suction that allows it to pick up sand, dirt, leaves or twigs, dust, and debris from a variety of pool surfaces. With a width suction mouth, it can also pick up larger debris more efficiently, reducing the need for multiple passes. Saves time and effort.
- Long Battery Life and Rapid Charging: Easy to use, simply click the button, drop and go. Osprey 700 Cordless Pool Cleaner equipped with a best-in-class 8800mAh rechargeable battery provides up to 110 minutes of continuous run time, making it perfect for in-ground pools up to 60 feet and charges fully in about 3.5 hours.
- Intelligent Path Planning & Maximum Coverage: Offering two types of path planning- an A-star path, and a regular wall-climbing path. Equipped with advanced sensors and gyroscopes, this robotic pool cleaner maps and navigates the pool with unmatched precision, resulting in methodical coverage with fewer missed spots.
- Worry-Free Service: As a professional pool cleaner manufacturer, we guarantee you not only a 2-year warranty, but also a 30-day return period, giving you plenty of time to study it and verify it.

Customer ratings by feature

Easy to clean	★★★★☆	4.3
Remote Control	★★★★☆	4.2
Easy to use	★★★★☆	4.2
Suction power	★★★☆☆	3.9

See all reviews

There is a newer model of this item:
WYBOT Innovative Cordless Robotic Pool Cleaner with APP, Wall Climbing Pool Vacuum with Strong Suction, Intelligent Path Planning, Lasts 110Mins, Quick Charge, Ideal for Above/In-ground Pools
$699.99
★★★★☆ (409)
In Stock.

$599.99
$195.39 Shipping & Import Fees Deposit to United Kingdom Details
Sales taxes may apply at checkout
Delivery **Thursday, May 25.** Order within 17 hrs 16 mins
Or fastest delivery **Monday, May 22**
Deliver to United Kingdom
In Stock
Add to Cart
Buy Now

Payment	Secure transaction
Ships from	Amazon
Sold by	WYBOT U.S.
Returns	Eligible for Return, Refund or Replacement within 30 days of receipt
Support	Free Amazon product support included

☐ Add a gift receipt for easy returns
Add to List

New (2) from $599.99

Other Sellers on Amazon
$659.99 Add to Cart
+ $51.64 shipping
Sold by: Wybotics Cleaner

Have one to sell?
Sell on Amazon

Frequently bought together

This item: WYBOT Cordless Robotic Pool Cleaner, Ultra Strong Suction, Wall Climb Pool Vacuu... $599.99
WYBOT Quick Charger for Osprey700 $79.99
Total price: $679.98
Add both to Cart

Special offers and product promotions

- Receive 1 Free Quick Charger free when you purchase 1 or more Osprey 700 pool cleaner offered by WYBOT U.S.. Select "Add both to Cart" to automatically apply promo code LRBDPW5Y. Here's how (restrictions apply)

Add both to Cart

From the brand

GOLD WINNER
2023
WYBOT
Within 10 years continuous development,
MUSE DESIGN AWARDS 2023 GOLD WINNER

Figure 5

Figure 5 : Amazon. (2021). Online Shopping for Electronics, Apparel, Computers, Books, DVDs & more. Amazon. https://www.amazon.com/

3.2.2 User personas

In order to develop a design approach, it was crucial to understand the demographic and their preferences. Therefore, the initial step involved creating a set of user personas to identify their specific design needs.

By creating detailed profiles of users representing the primary target audience, the author ensured that the design approach was customized to meet the specific needs and preferences of the users. The personas acted as a foundation for making well-informed design choices and prioritizing design elements and features that were particularly relevant to the target audience's requirements.

This process played a critical role in ensuring that the final design was intuitive, user-friendly, and engaging, which ultimately contributed to the success of the design.

Persona 1: The Tech Learner

This person is interested in learning about technology and wants to understand how the tech products work. They are looking for a website that can provide them with detailed information about each product.

Demographics: Age range: 18-35, curious about technology, interested in learning.

Goals: Understand how different tech products work.

Pain Points: Limited information or explanations on the website, concerns about the complexity of the information.

Persona 2: The Tech Seeker

This person is looking for specific information about a particular tech product or technology. They are looking for a website that can provide them with detailed information and specifications about different types of technology.

Goals: Get step-by-step instructions and guidance.

Pain Points: Limited guidance or instructions on the website, concerns about the complexity of the information.

3.3 User Interface

This section explores how UI design was approached. It will discuss the low fidelity design, high fidelity design and final implementation.

3.3.1 Wireframe (Low Fidelity)

The wireframe represents the structure of the website, with a focus on the placement of key elements on each page, including content sections, navigation, and products. The decision was made to prominently feature the product catalog on the homepage, catering to the user's goal of finding a product quickly. This approach reduces the number of clicks required for a customer to make a purchase.

Incorporating a hero section with a prominent shop button serves as a strong call-to-action, encouraging visitors to engage with the website's products or services. By capturing attention and directing users to the shopping experience, it enhances user engagement and conversion. Show in figure

Taking into account the personas outlined in section 3.2.2, it was recognized that providing detailed product information on the product page was essential. To meet this need, specific product specifications and expanded descriptions were included. Additionally, a dedicated section for product comparisons was designed to facilitate informed decision-making and enhance the shopping experience.

The structure of the other pages followed intuition and followed established ecommerce standards.

Overall, the wireframe provided a clear visual representation of the site's structure and functionality, contributing to a seamless development process. The included wireframes below demonstrate these design decisions and the overall website structure.

Homepage

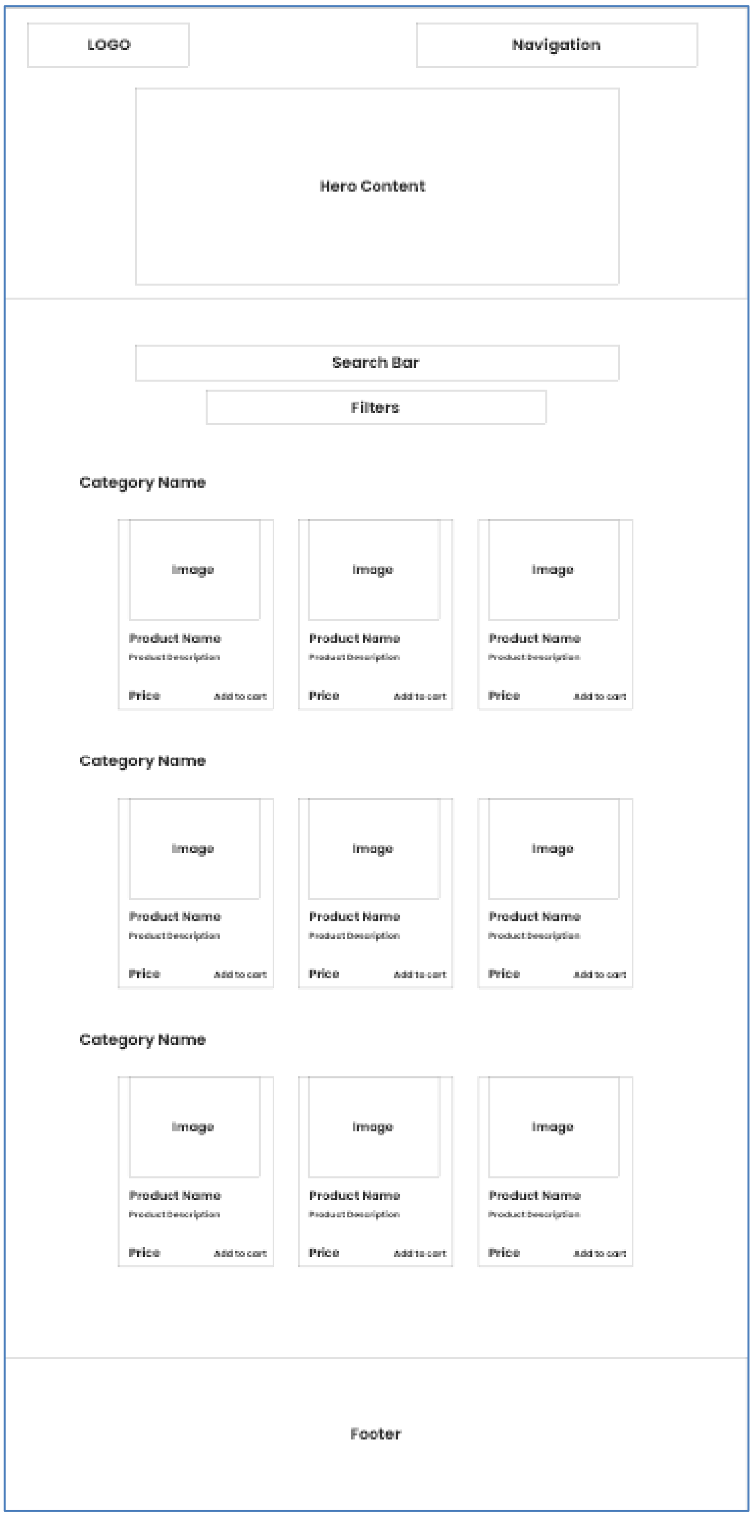


Figure 6

Figure 6 : Homepage wireframe

Checkout Page

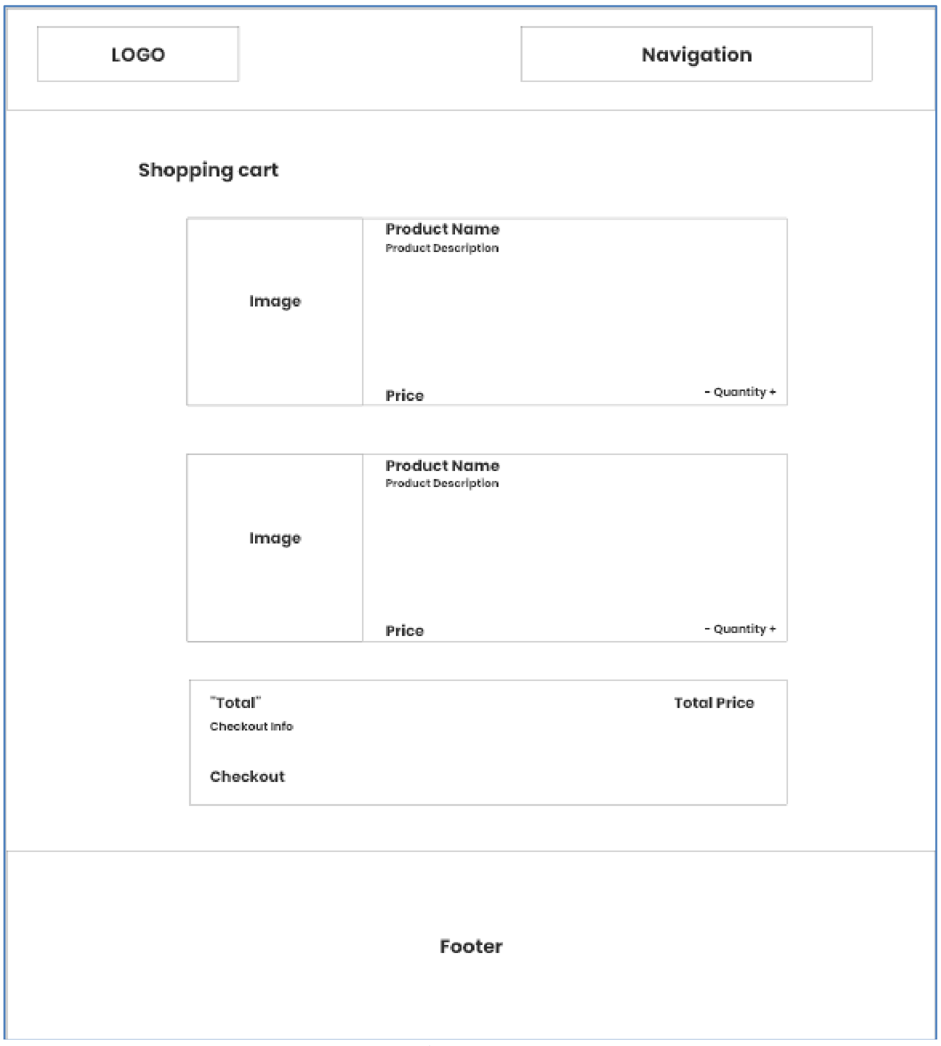


Figure 7

Figure 7 : Shopping Cart wireframe

Admin Stock Page

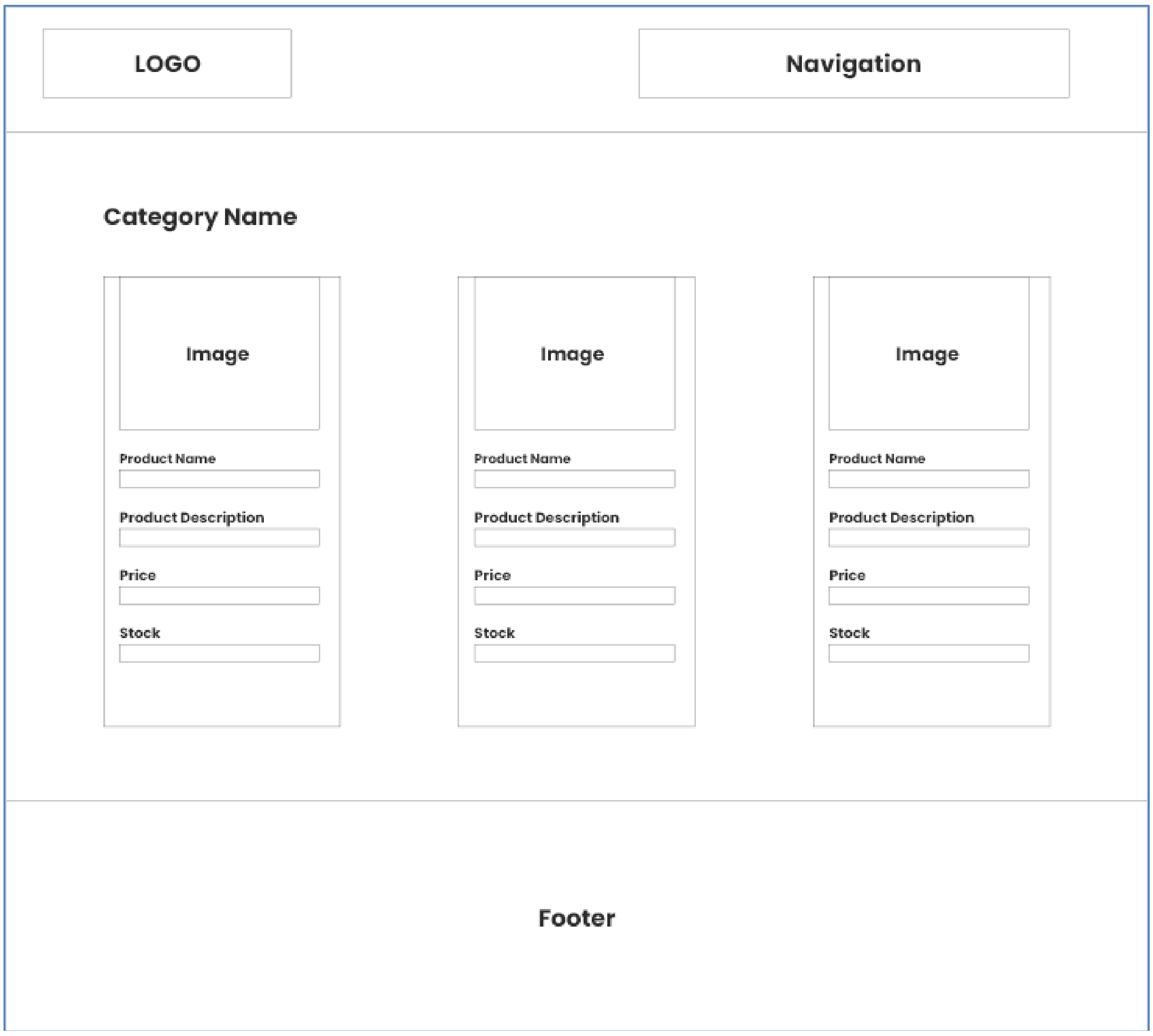


Figure 8

Figure 8 : Admin Stock wireframe

Login/Sign-up

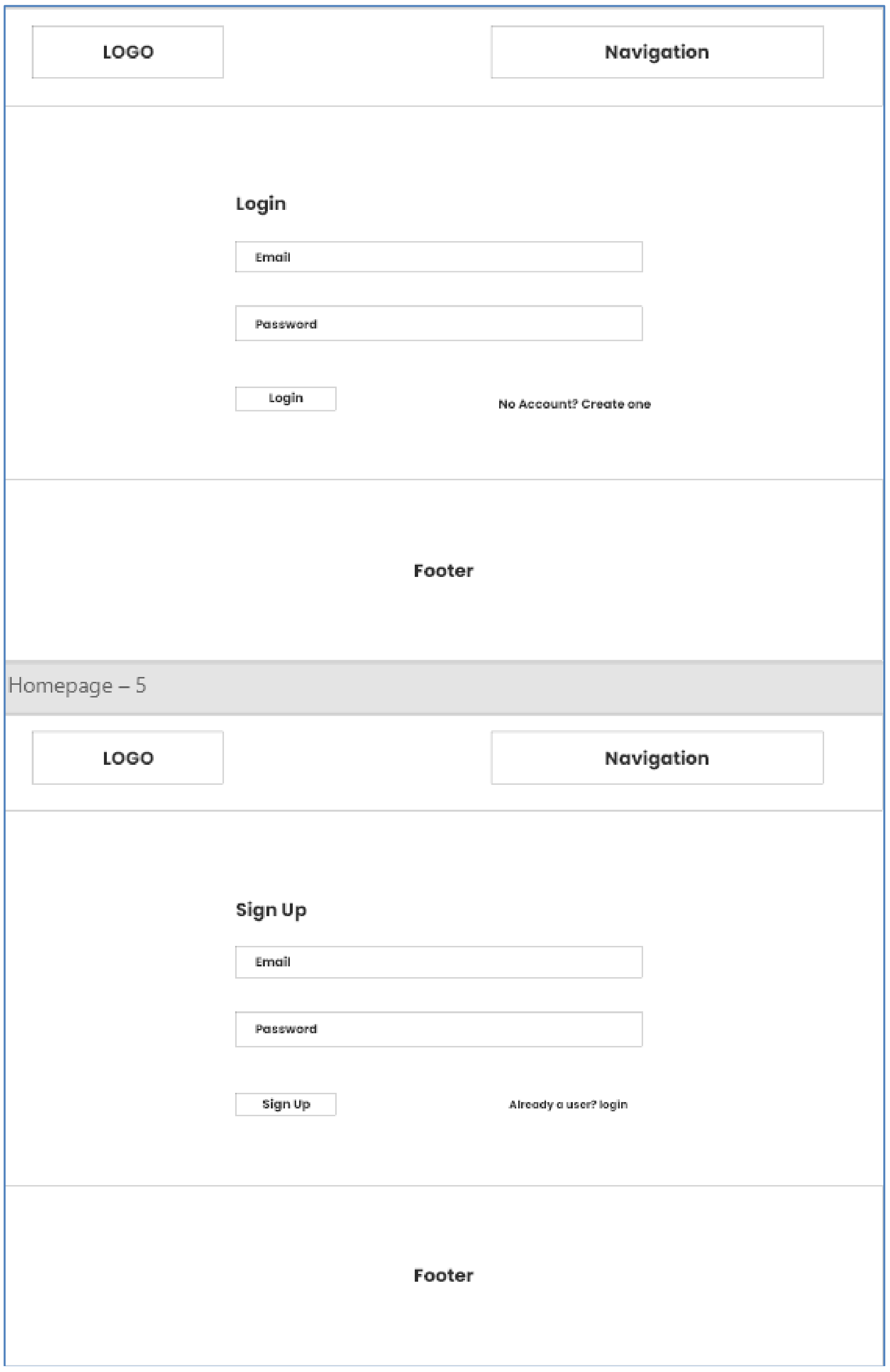
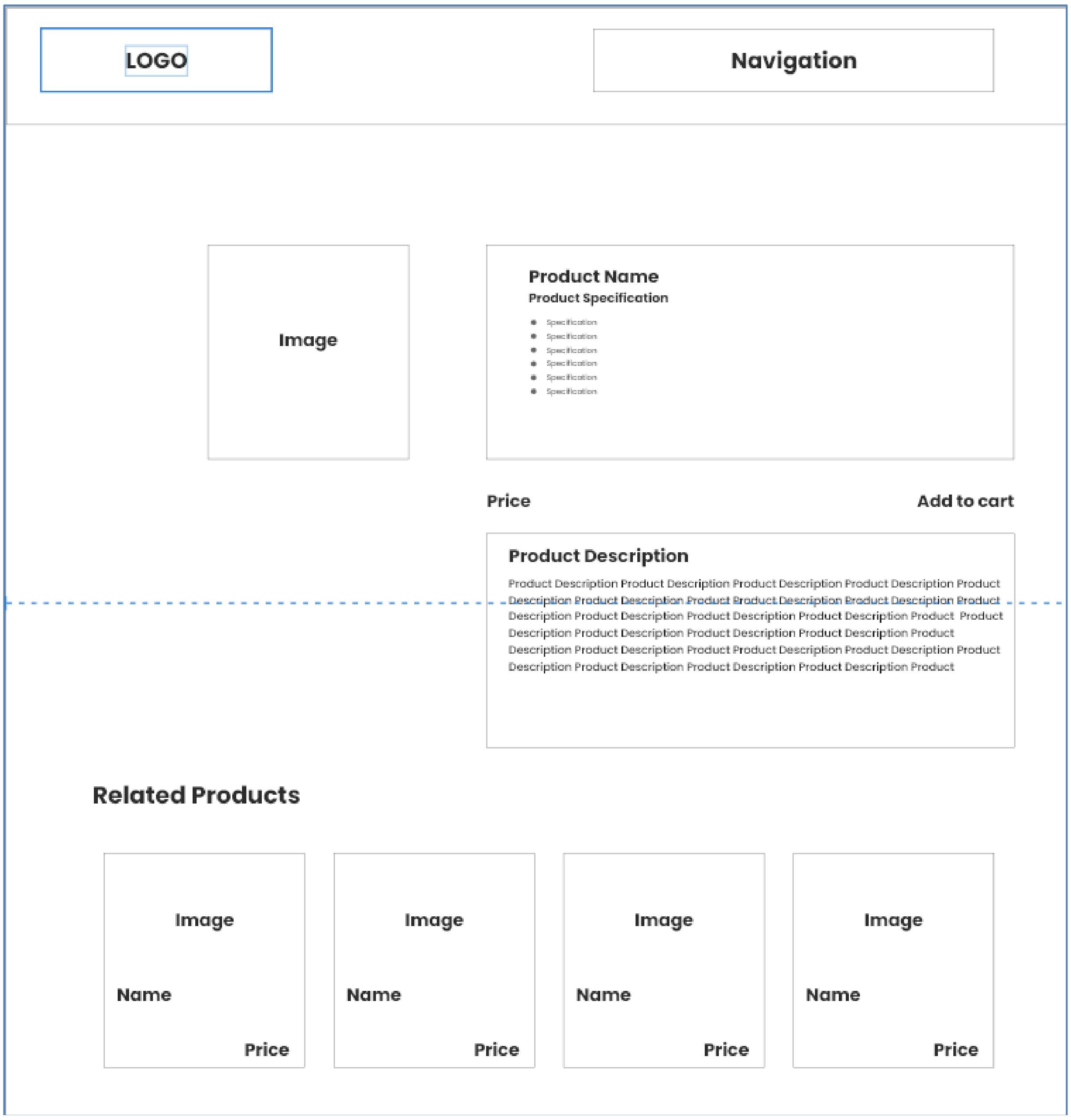


Figure 9

Figure 9 : Login/ Sign Up wireframe

Product Page



Product Description Product Description Product Description Product Description Product Description Product Description Product Product Description Product Description Product Description Product Description Product Description Product Description Product Product Description Product Description Product Description Product Description Product Description Product Description Product Description Product Product Description Product Description Product Description Product Description Product Description Product Description Product Description Product

Figure 10

Figure 10 : Product Page wireframe

3.3.2 Final Design (High Fidelity)

The high-fidelity design takes the wireframe design to the next level by incorporating a high level of detail and realism. The designer focuses on refining visual aspects, including typography, images, and colors, to closely resemble the final application. This phase of the design process aims to provide a more accurate representation of the visual experience that users will encounter when using the application. By paying attention to these finer details, the high-fidelity design enhances the overall aesthetics and user experience of the application.

Typography

In order to maintain a cohesive and consistent visual experience across the entire application, a deliberate decision to utilize a single font throughout was made. After careful consideration, Segoe UI was selected as the primary font for its modern, readable, and professional qualities. This font choice ensures that all users and websites interacting with the application benefit from a visually appealing and user-friendly typography. By employing Segoe UI, the application achieves a harmonious and polished look while effectively conveying information to the users.

Color Scheme

For the header and hero section of the site, a dark blue color was chosen with the hex code #28346F. This deep blue shade was chosen to create a visually appealing and impactful first impression. It adds a sense of elegance and professionalism to these sections, capturing the attention of users.

To ensure optimal visibility and readability of the products and information, a deliberate decision to use a light gray color for the body of the site was made. Unlike a stark white background, this shade of gray provides a softer and more comfortable visual experience, making the products stand out effectively.

Furthermore, in order to enhance readability, the application has a dark gray color for the text in the body of the content. This darker gray shade is known for its improved legibility compared to pure black. By using this color, the text becomes easier to read, reducing eye strain and ensuring that users can effortlessly consume the information presented.

Overall, the color choices for the header, hero section, body, and text were made with the intention of creating an aesthetically pleasing and user-friendly interface. The combination of dark blue, light gray, and dark gray colors enhances the overall visual experience and promotes readability for users navigating the site.

Images Selection

The application utilizes images with transparent backgrounds because it offers a lot of creative freedom and flexibility. With transparent backgrounds, these images can seamlessly blend into any background, regardless of its color or design.

This approach allowed application to be experimented with different backgrounds and layouts without worrying about the images clashing or looking out of place. Whether it's a vibrant and colorful background or a subtle and minimalistic one, the images with transparent backgrounds can effortlessly adapt and fit in perfectly.

By using images with transparent backgrounds, the focus remained solely on the subject of the image, without any distractions from the background. This enhances the visual impact and creates a more cohesive and professional look.

Overall, the use of images with transparent backgrounds creates visually stunning designs that can be easily customized and adapted to various contexts. It adds a layer of versatility and creative potential for the design process.

Summary

In conclusion, the high-fidelity design approach has helped create a realistic and detailed representation of the final application. By carefully selecting typography, colors, and images, a cohesive and visually appealing user interface has been achieved.

Using the Segoe UI font consistently throughout the application ensures a modern and professional look while maintaining readability. The color choices, like the deep blue for the header and hero section, light gray for the body, and dark gray for text, enhance visibility and legibility, making information easily accessible.

By using images with transparent backgrounds, they can be seamlessly integrate them into any background, creating a polished and visually pleasing design without distractions.

In summary, the high-fidelity design approach, with its thoughtful typography, color scheme, and image selection, has laid the foundation for an engaging and user-friendly interface, making the application visually appealing and easy to navigate.

Here are images from the high fidelity design.

Homepage

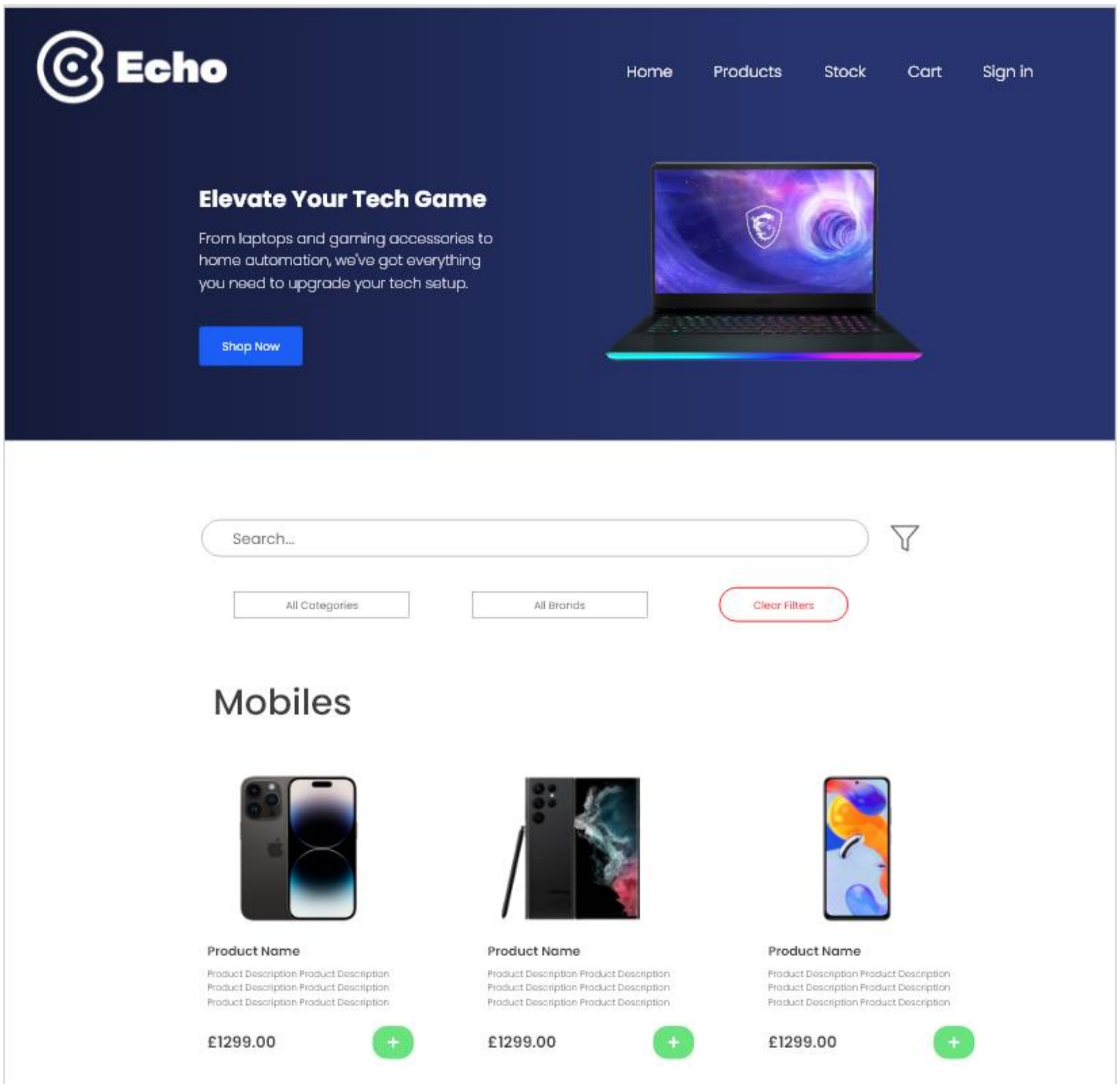

Echo
Home
Products
Stock
Cart
Sign in
Elevate Your Tech Game
From laptops and gaming accessories to home automation, we've got everything you need to upgrade your tech setup.
Shop Now
Search...
All Categories
All Brands
Clear Filters
Mobiles
Product Name
Product Description Product Description Product Description Product Description Product Description Product Description
£1299.00
+
Product Name
Product Description Product Description Product Description Product Description Product Description Product Description
£1299.00
+
Product Name
Product Description Product Description Product Description Product Description Product Description Product Description
£1299.00
+

Figure 11

Figure 11 : Homepage final design

Product Page

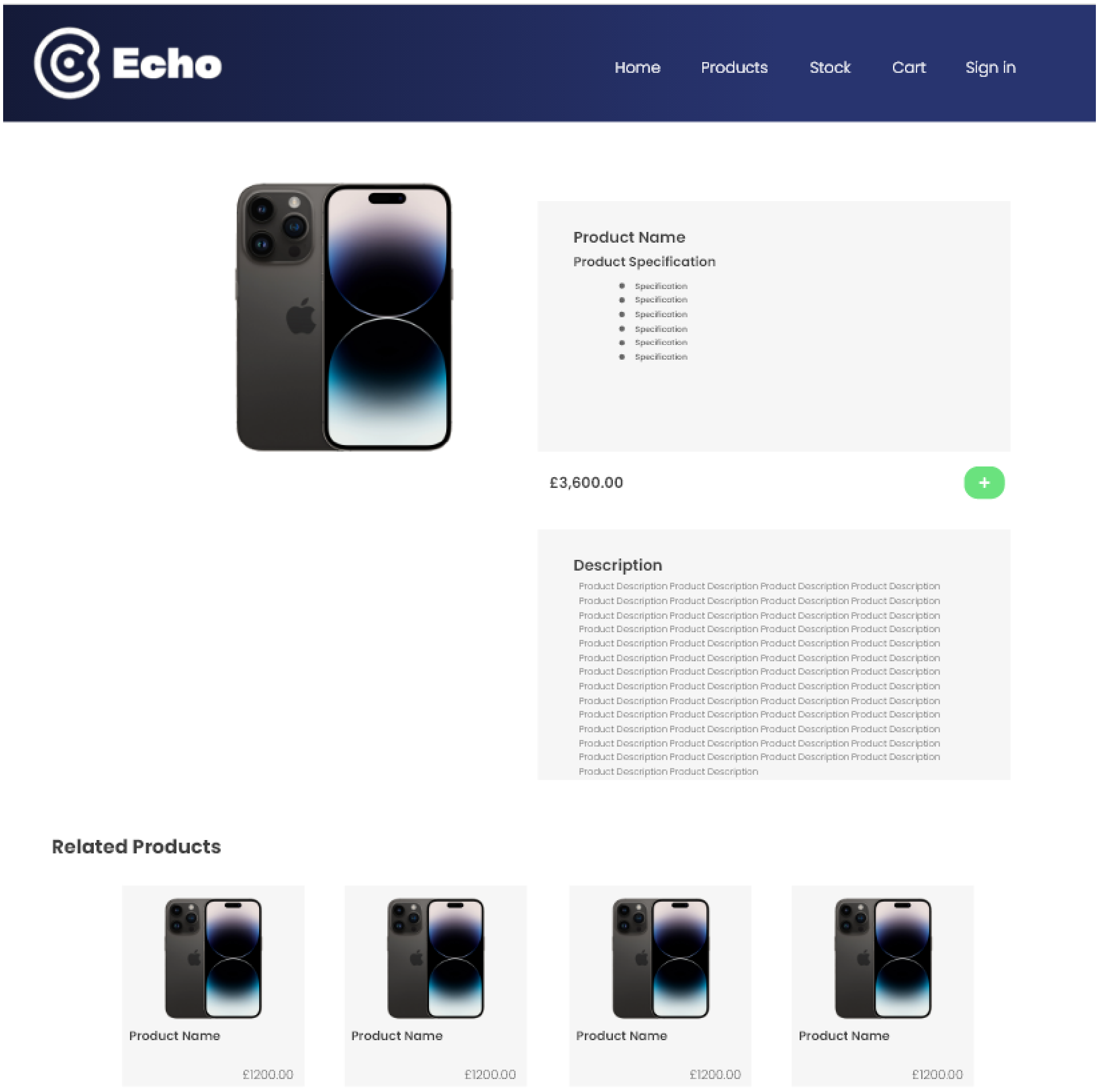
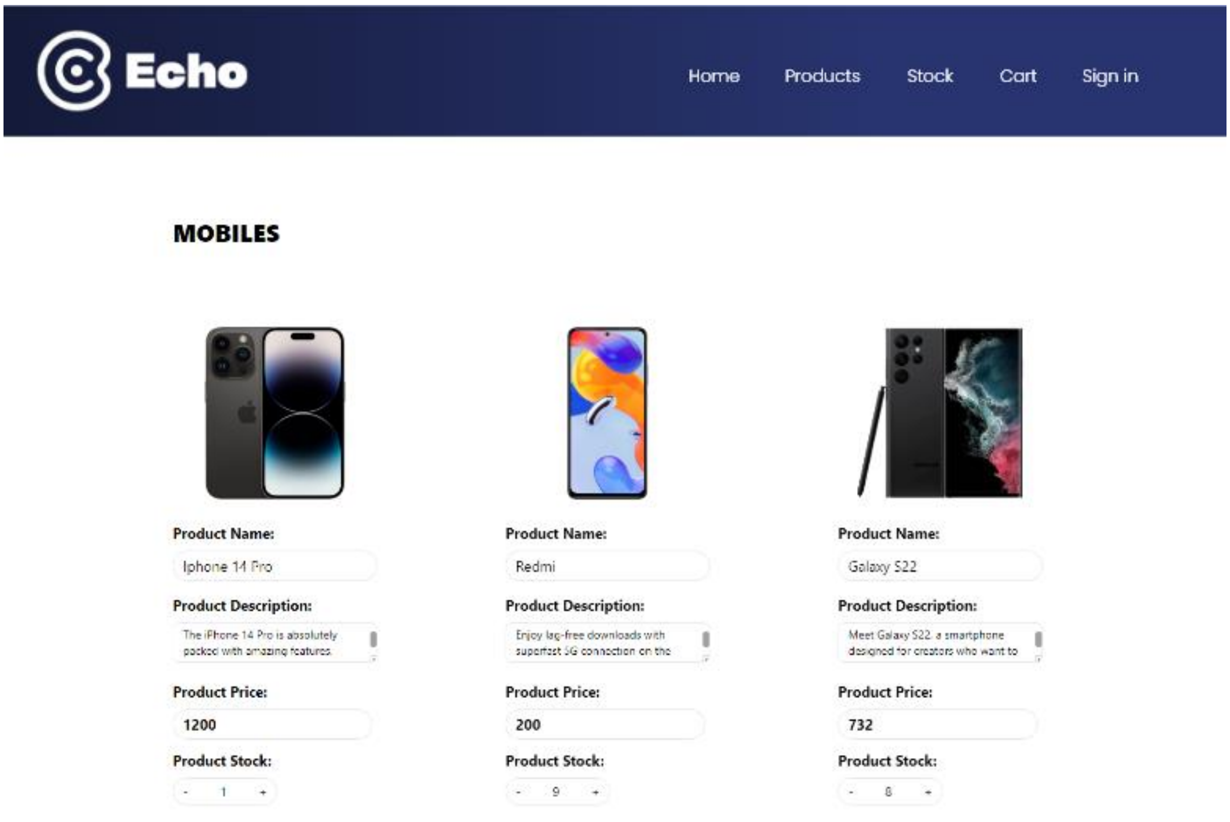

Echo
Home
Products
Stock
Cart
Sign in
Product Name
Product Specification
Specification
Specification
Specification
Specification
Specification
Specification
£3,600.00
+
Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description Product Description Product Description
Product Description Product Description
Related Products
Product Name
£1200.00
Product Name
£1200.00
Product Name
£1200.00
Product Name
£1200.00

Figure 12

Figure 12 : Product Page final design

Stock Page



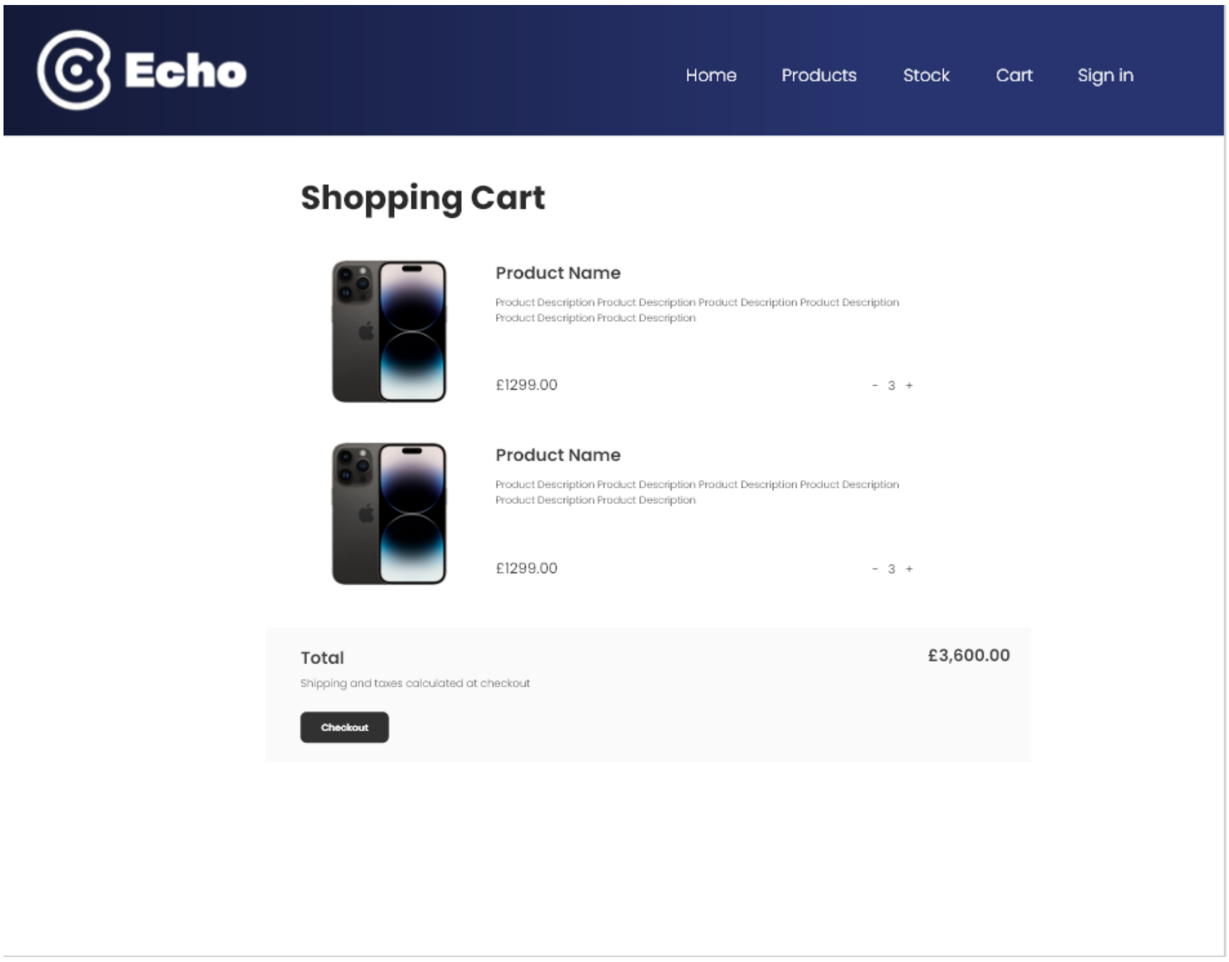
MOBILES

Product Name:	Product Name:	Product Name:
Iphone 14 Pro	Redmi	Galaxy S22
Product Description:	Product Description:	Product Description:
The iPhone 14 Pro is absolutely packed with amazing features.	Enjoy lag-free downloads with superfast 5G connection on the	Meet Galaxy S22, a smartphone designed for creators who want to
Product Price:	Product Price:	Product Price:
1200	200	732
Product Stock:	Product Stock:	Product Stock:
- 1 +	- 9 +	- 8 +

Figure 13

Figure 13 : Stock Page final design

Checkout Page



Shopping Cart

Product Name

Product Description Product Description Product Description Product Description Product Description Product Description

£1299.00 - 3 +

Product Name

Product Description Product Description Product Description Product Description Product Description Product Description

£1299.00 - 3 +

Total £3,600.00

Shipping and taxes calculated at checkout

Checkout

Figure 14

Figure 14 : Checkout Page final design

Login/ Sign-up

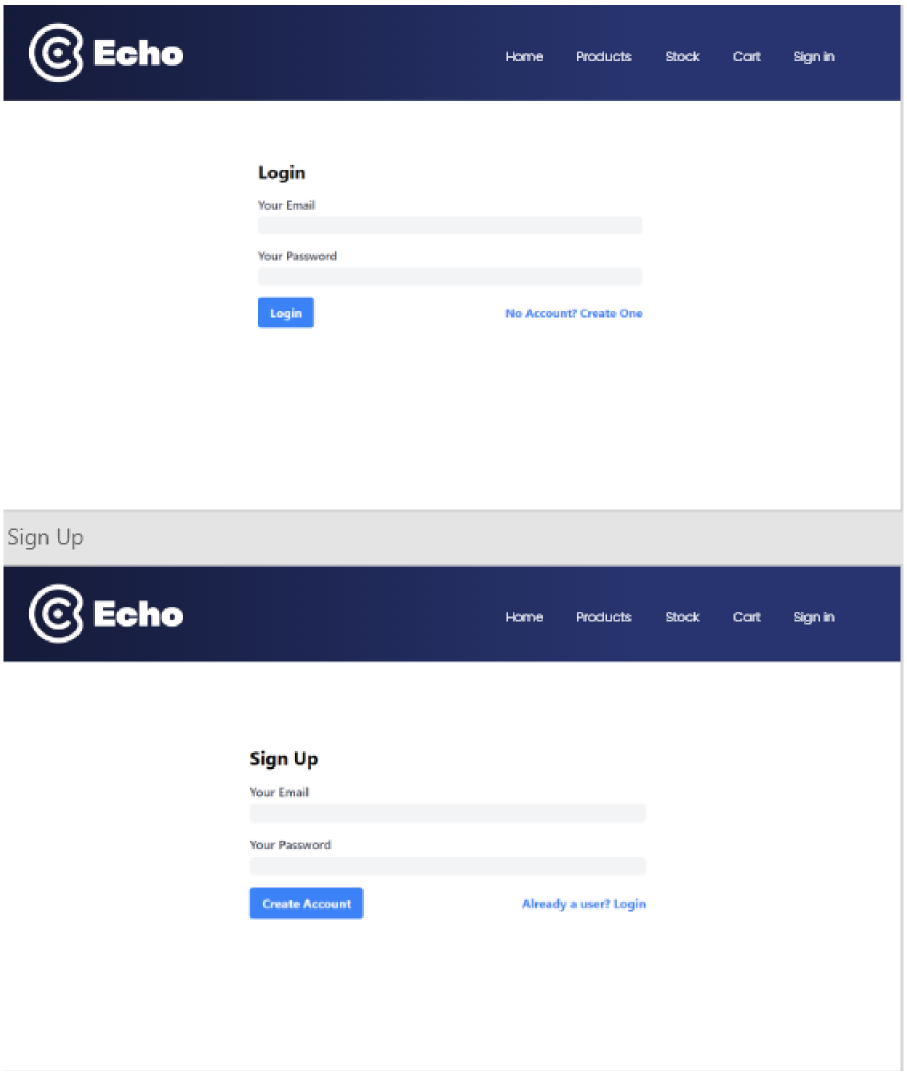


Figure 15

Figure 15 : Login/ Sign Up final design

3.3.3 Implementation Changes

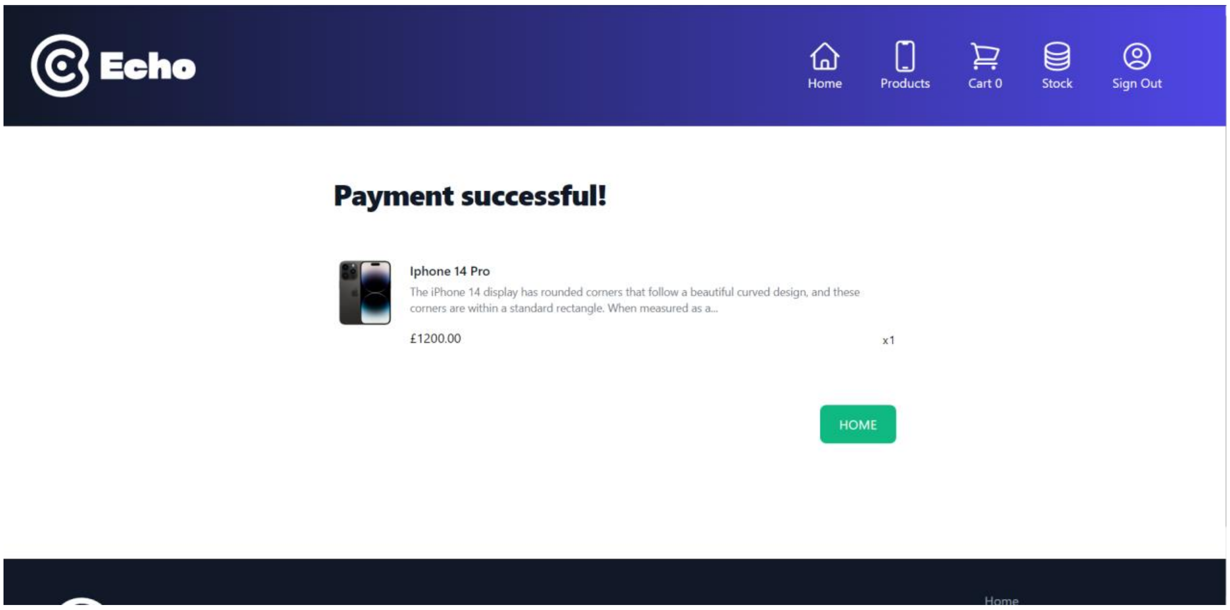
During the development phase, several valuable additions were made to enhance the design and user experience of the application. Firstly, the implementation of "Added to Cart" message to provide instant feedback to the user when they add an item to their cart. This notification serves as a confirmation and ensures that users are aware of their successful action.

Secondly, a dedicated success page that appears after the checkout process was introduced. This page displays the purchased items and includes a convenient "Return to Homepage" button. This step ensures that users receive clear confirmation of their successful transaction and provides a seamless transition back to the main page.

Lastly, responsive design was added, making the application adaptable across different devices and screen sizes. This involved strategic layout changes to optimize the user experience on various platforms. Additionally, a new navigation system in the form of a footer was implemented to ensure easy access to important sections of the application.

These additions contribute to an improved user journey by providing real-time feedback, clear confirmation, and seamless navigation. The responsive design further enhances accessibility and allows users to enjoy a consistent and optimized experience regardless of the device they are using.

Success Page



Payment successful!

Iphone 14 Pro

The iPhone 14 display has rounded corners that follow a beautiful curved design, and these corners are within a standard rectangle. When measured as a...

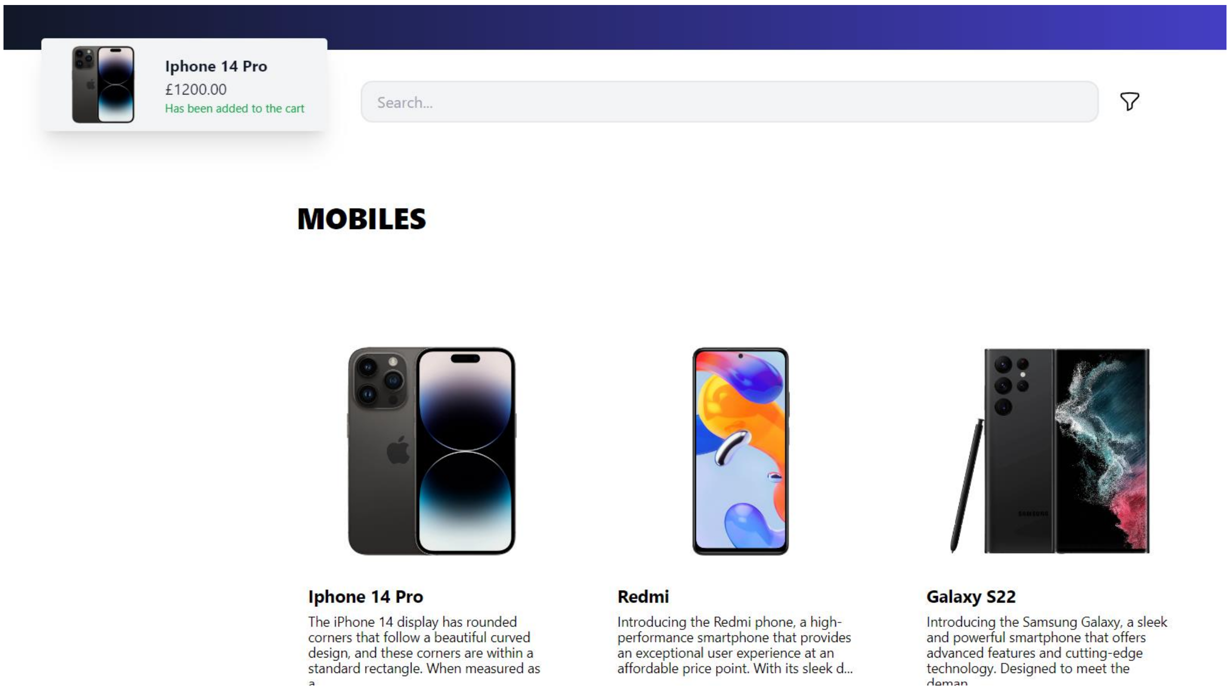
£1200.00

x1

Figure 16

Figure 16 : Success Page final design

Cart Message


Iphone 14 Pro
£1200.00
Has been added to the cart

MOBILES

Iphone 14 Pro
The iPhone 14 display has rounded corners that follow a beautiful curved design, and these corners are within a standard rectangle. When measured as a

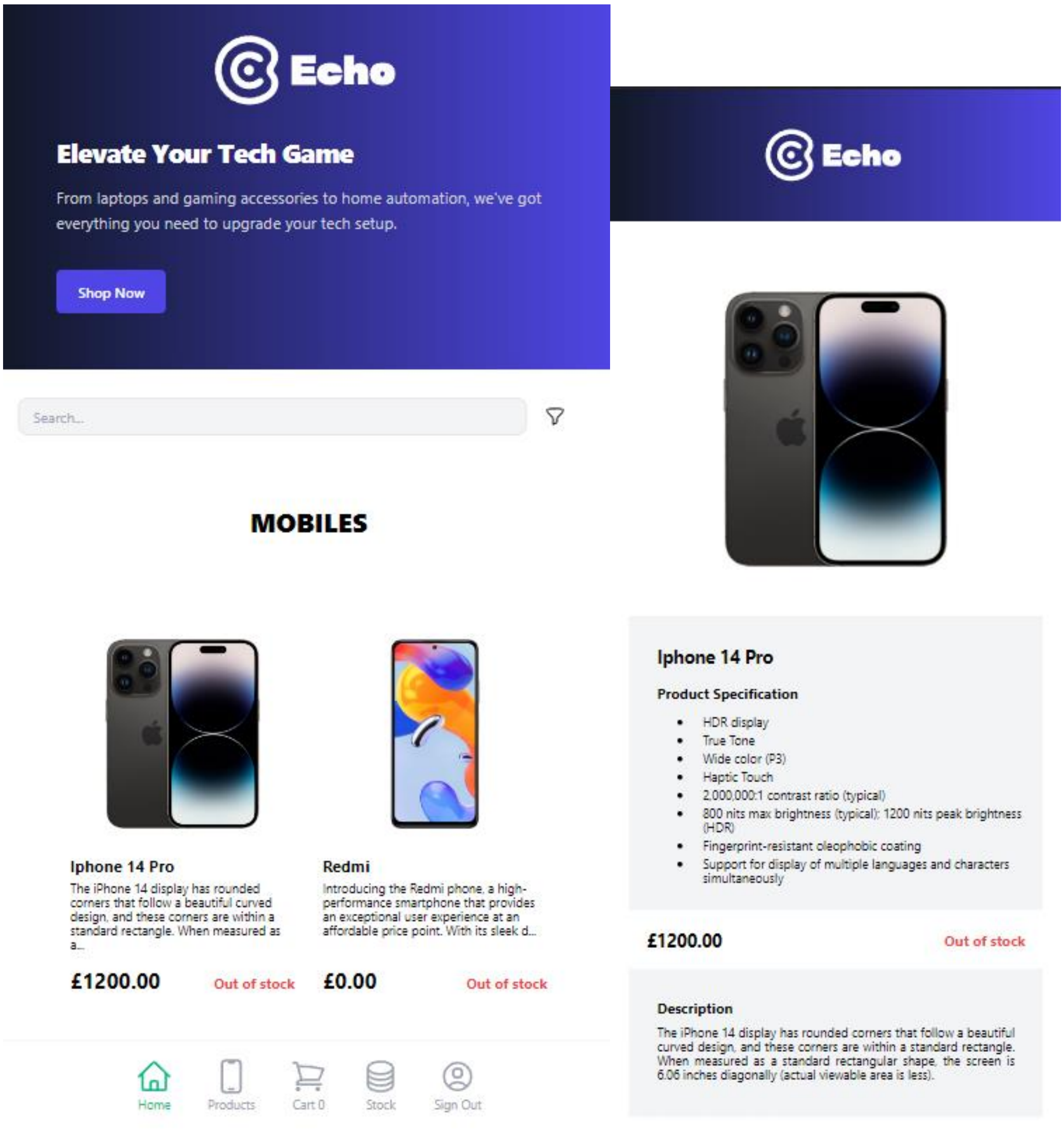
Redmi
Introducing the Redmi phone, a high-performance smartphone that provides an exceptional user experience at an affordable price point. With its sleek d...

Galaxy S22
Introducing the Samsung Galaxy, a sleek and powerful smartphone that offers advanced features and cutting-edge technology. Designed to meet the deman

Figure 17

Figure 17 : Cart Message final design

Responsive Design


Echo

Elevate Your Tech Game

From laptops and gaming accessories to home automation, we've got everything you need to upgrade your tech setup.

Shop Now

Search...

MOBILES

Iphone 14 Pro

The iPhone 14 display has rounded corners that follow a beautiful curved design, and these corners are within a standard rectangle. When measured as a...

£1200.00 Out of stock

Redmi

Introducing the Redmi phone, a high-performance smartphone that provides an exceptional user experience at an affordable price point. With its sleek d...

£0.00 Out of stock

Home Products Cart 0 Stock Sign Out

Echo

Iphone 14 Pro

Product Specification

- HDR display
- True Tone
- Wide color (P3)
- Haptic Touch
- 2,000,000:1 contrast ratio (typical)
- 800 nits max brightness (typical); 1200 nits peak brightness (HDR)
- Fingerprint-resistant oleophobic coating
- Support for display of multiple languages and characters simultaneously

£1200.00 Out of stock

Description

The iPhone 14 display has rounded corners that follow a beautiful curved design, and these corners are within a standard rectangle. When measured as a standard rectangular shape, the screen is 6.06 inches diagonally (actual viewable area is less).

Figure 18

Figure 18 : Mobile final design

3.4 Database Design

3.4.1 Database Solution

MongoDB was chosen as the database solution for its simplicity, flexibility, and performance benefits.

Firstly, MongoDB makes data parsing a breeze. Its flexible structure allows me to store and retrieve data in a way that fits the application's needs without rigid schemas.

Secondly, MongoDB scales effortlessly. As the application grows, MongoDB can handle increasing data volumes by distributing the load across multiple servers. This ensures that the application remains responsive and performs well under heavy traffic.

In terms of performance, MongoDB is designed to deliver speed and efficiency. Its indexing, query optimization, and caching features optimize data retrieval and manipulation, providing a seamless user experience.

Overall, MongoDB simplifies data management, adapts to the application's evolving needs, and ensures optimal performance. It's the perfect choice for my project, allowing me to focus on building a robust and user-friendly application.

3.4.2 Entity Relationship Diagram (ERD)

An ERD, or Entity-Relationship Diagram, serves as a visual representation of the database structure, illustrating the relationships between entities. In this case, it has three entities: Users, Carts, and Products, which correspond to real-world concepts.

The relationship between Users and Carts is one-to-one, meaning each user can have only one cart associated with them. On the other hand, the relationship between Carts and Products is one-to-many, indicating that a single cart can contain multiple products.

Each entity has a primary key, represented by a unique identifier. In the document-oriented approach, the primary key for each entity is a single id that uniquely identifies a document.

The Users entity includes attributes such as _id (the primary key), email, password, and role. These attributes help define security constraints and user roles within the system.

The Carts entity consists of attributes like _id (primary key) and user_id, which serves as a foreign key referencing the Users entity. Additionally, the Carts entity has an items array that allows for multiple products to be associated with a single cart. The items array utilizes the _id attribute from the Products entity as a foreign key within the array. Lastly, the Products entity contains attributes such as _id, Name, Description, Price, Category, Picture (representing a URL to the directory where the picture is stored), Stock, Specs, and Brand. These attributes provide details about the products available in the system.

Overall, this ERD provides a clear representation of the entities, their relationships, and the attributes associated with each entity. It helps in understanding the structure and connections within the database, enabling efficient data management and retrieval.

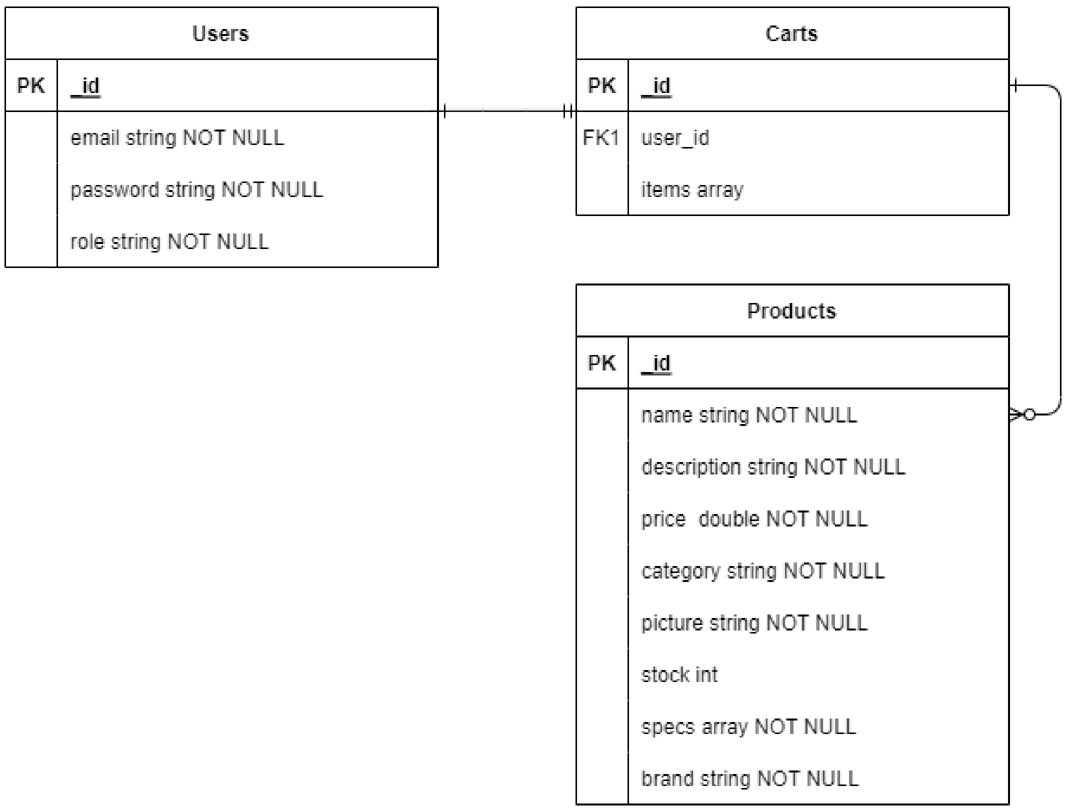


Figure 19

Figure 19 : Entity Relationship Diagram

3.5 Use Cases

A use case is a real-life story that helps the reader understand how users will interact with the system and how it will respond to their actions. It is seeing the system from the user perspective.

This section will provide example use cases and a use case diagram to offer a comprehensive understanding of the user's perspective and interactions with the system. By exploring these use cases and visual representation, that can gain insights into how the system caters to user needs and supports their desired actions.

3.5.1 Example use cases.

Checkout Process

- Scenario: A user wants to complete a purchase and proceed to checkout.
- Design Decision: The checkout process was designed to be intuitive and streamlined, minimizing steps and reducing friction.
- Implementation: The checkout page includes clear and concise form fields for capturing essential information, payment details, and order summary.
- Contribution to User Experience: By simplifying the checkout process, users can quickly complete their purchases, reducing the likelihood of cart abandonment and enhancing overall satisfaction.

Search Functionality

- Scenario: A user wants to find a specific product on the ecommerce site.
- Design Decision: The search functionality was prioritized to help users easily locate desired products.
- Implementation: The search bar is prominently placed.
- Contribution to User Experience: The efficient search functionality saves users' time and effort, enabling them to find products they are looking for with ease, enhancing their overall experience on the site.

Product Filtering Options

- Scenario: A user wants to refine product listings based on specific criteria.
- Design Decision: The ecommerce site incorporates robust product filtering options to cater to users' diverse preferences and requirements.
- Implementation: Users can filter products by various attributes such as brand and category.
- Contribution to User Experience: The availability of comprehensive product filtering options empowers users to narrow down their choices and find products that align with their specific needs, resulting in a more personalized and satisfying shopping experience.

3.5.2 Use Case Diagram

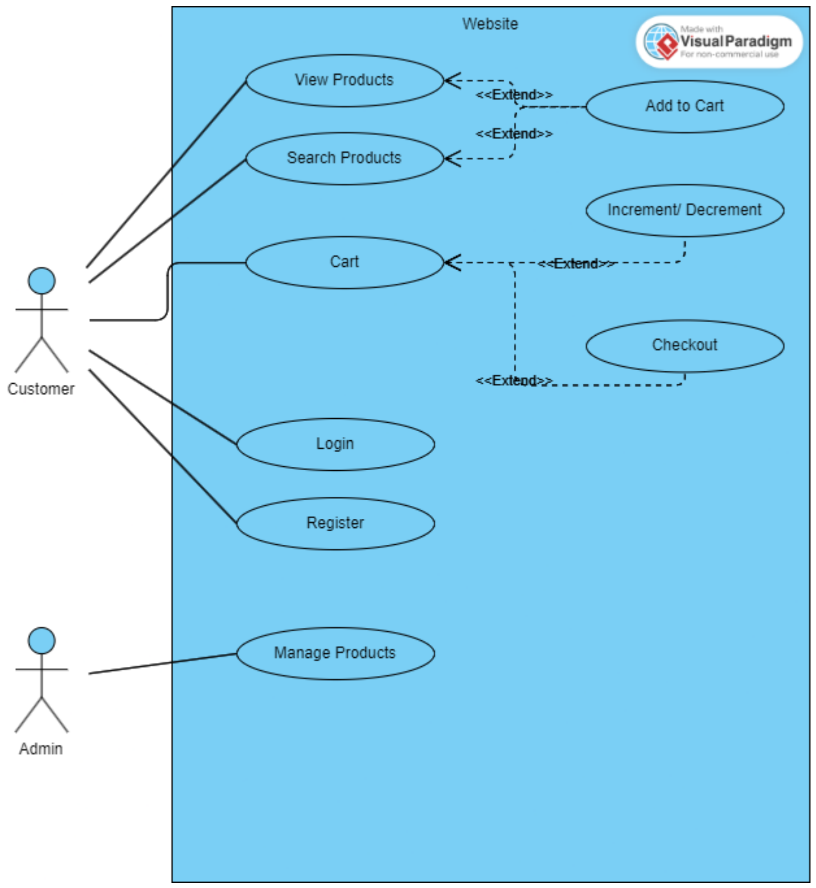


Figure 20

Figure 20 : Use Case Diagram

3.6 Conclusion

In conclusion, the design section of the SPA ecommerce site has prioritized creating an exceptional user experience while also recognizing its significant impact on improving SEO. The design decisions made align with search engine optimization principles, aiming to enhance website visibility, attract organic traffic, and facilitate business growth.

Throughout the design process, careful consideration was given to the needs and preferences of the target audience. By incorporating user personas, valuable insights into user behaviors and expectations were gained, enabling the tailoring of design decisions accordingly. This user-centric approach not only enhances the overall user experience but also contributes to improved SEO. Search engines place value on websites that provide relevant and engaging experiences to users, and by prioritizing their needs, the likelihood of higher search engine rankings and increased organic traffic is heightened.

Furthermore, the provision of comprehensive product information and descriptions not only empowers users to make well-informed purchasing decisions but also enhances the discoverability of the website. Search engines heavily rely on the relevance and quality of content to determine search rankings. By offering detailed product information, there is an increased likelihood of the website being indexed and ranked for relevant search queries.

While significant progress has been made in designing a user-friendly experience, the importance of incorporating customer reviews is recognized. User-generated content, such as reviews, adds credibility and trust to the website and plays a crucial role in improving SEO. Positive reviews and ratings can influence search engine rankings as they indicate the quality and reputation of the products and services offered. Therefore, the integration of a customer review feature would not only benefit the users by providing valuable insights but also contribute to improved SEO performance.

Additionally, streamlining the signup process and providing clear guidance and incentives for creating user accounts can have a positive impact on both the user experience and SEO. By encouraging users to create accounts and highlighting the personalized experiences and benefits that come with it, increased user engagement, loyalty, and potential generation of user-generated content can be achieved. These factors, in turn, contribute to improved SEO performance.

In summary, the design decisions made in the design section of the SPA ecommerce site have prioritized the creation of an exceptional user experience while strategically aligning with the goal of improving SEO. By focusing on user needs, providing comprehensive product information, incorporating customer reviews, and optimizing the signup process, the visibility, relevance, and credibility of the website in search engine rankings can be enhanced. This, in turn, drives organic traffic, improves user engagement, and ultimately contributes to the overall success of the ecommerce business.

Chapter 4

Implementation

4.1 Introduction

In this chapter, the specific tasks and activities performed to implement the proposed solution are explored. The coding techniques, algorithms, and technologies utilized to build the system are discussed, along with the considerations and decisions made during the implementation phase.

Overall, this chapter provides a comprehensive account of the practical work carried out, the results achieved, and the testing procedures followed. It offers a detailed insight into the implementation process and serves as a valuable resource for understanding the steps taken to bring the proposed solution to fruition.

4.2 Pre-implementation decisions

Prior to the implementation of the website, a critical decision was made to optimize the database performance by temporarily dropping the review system functionality. This decision aimed to reduce the number of read and write operations to the database, ensuring its efficient operation and mitigating potential scalability issues.

Although the review system (3.6) is an important feature for user engagement and credibility, a strategic prioritization of other aspects of the implementation process was undertaken to deliver a robust and seamless user experience. This approach enabled the effective allocation of development resources and focused efforts on optimizing the core functionality of the website.

By temporarily removing the review system, the overall load on the database was reduced, resulting in minimized read and write operations necessary for managing user reviews. This decision not only facilitated smoother and faster performance but also enabled the allocation of additional resources to implement and explore features that directly contribute to the SEO optimization of the website.

The importance of customer feedback and reviews is duly acknowledged, and there are plans to reintroduce the review system in future iterations of the website. However, the strategic decision to exclude this functionality in the initial implementation phase allowed for the prioritization of other crucial elements that significantly impact the overall success of the website, particularly its SEO performance..

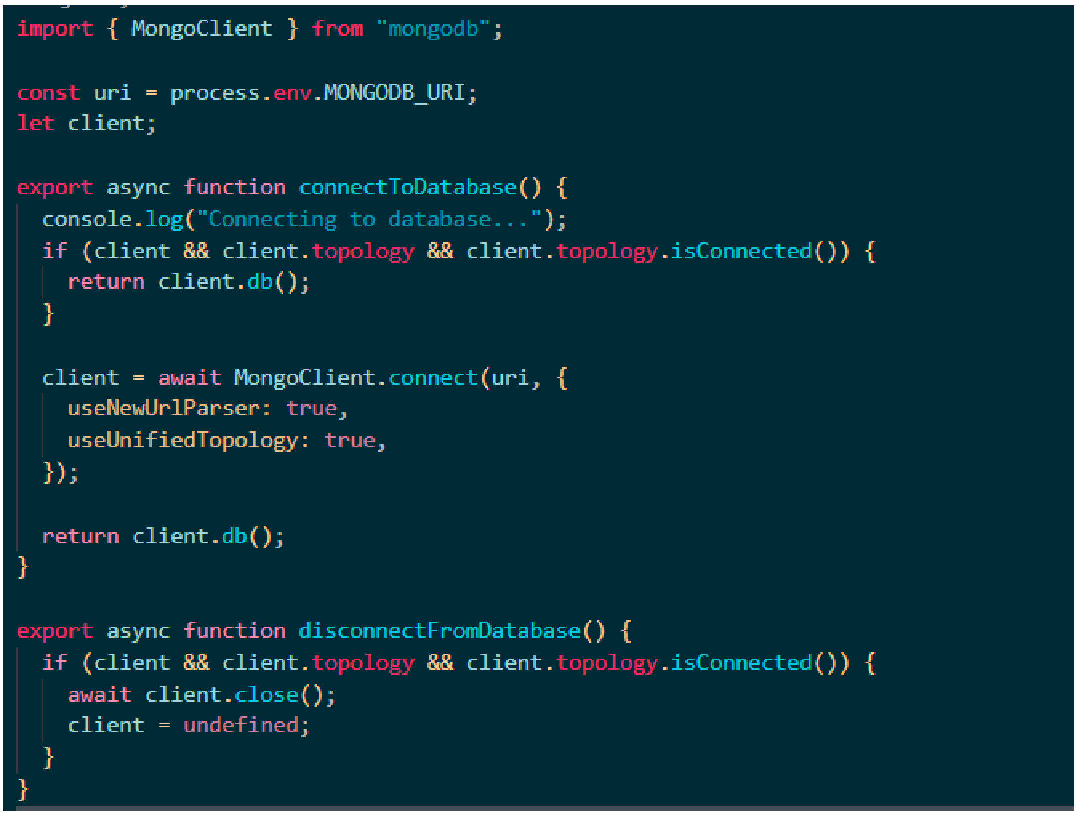
Through careful planning and consideration, the application aims to strike a balance between delivering a high-performing website and incorporating essential features such as the review system in the future development roadmap. This approach allows us to lay a solid foundation for the implementation process while setting the stage for future enhancements that will further elevate the user experience and boost SEO performance.

4.3 Implementation Of Database

During the implementation phase, a solid foundation for the database by setting up a project called 'Ecommerce' and creating a free cluster on MongoDB was established. To ensure the database structure aligns with the design specifications from Section 3.42, the collections were carefully mapped out : Carts, Products, and Users.

In order to integrate these collections seamlessly into the code, a few essential steps were followed. Firstly, the application retrieved the connection string from the MongoDB site and replaced the necessary credentials. To ensure easy accessibility throughout the project, a connection string in a .env file was stored, which enabled global access.

Next, the code was organized by creating a 'lib' folder, the mongoose.js file was placed. This file served as the foundation for working with the MongoDB database using the Mongoose library. It included the exportable functions 'connectToDatabase' and 'disconnectToDatabase,' which leveraged the capabilities of the MongoClient library.



```
import { MongoClient } from "mongodb";

const uri = process.env.MONGODB_URI;
let client;

export async function connectToDatabase() {
  console.log("Connecting to database...");
  if (client && client.topology && client.topology.isConnected()) {
    return client.db();
  }

  client = await MongoClient.connect(uri, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  });

  return client.db();
}

export async function disconnectFromDatabase() {
  if (client && client.topology && client.topology.isConnected()) {
    await client.close();
    client = undefined;
  }
}
```

The provided figure, illustrates the implementation of the connectToDatabase function.

This function plays a crucial role in establishing a connection to the database while ensuring the prevention of errors and unnecessary opening of new clients.

As shown in the figure, the connectToDatabase function begins by checking if there is an existing client. This check helps to avoid errors and ensures that multiple connections aren’t created unnecessarily.

Additionally, the figure highlights the disconnectFromDatabase function, which is responsible for closing the database connection. Similar to the connectToDatabase function, it includes a check to verify if there is an open connection before proceeding with the disconnection process. By implementing this check, the chances of encountering errors are reduced and the smooth closure of the database connection when it is no longer needed is ensured.

By implementing these steps, the application establishes a robust connection between itself and the MongoDB database. This allowed for seamless interactions with the collections and CRUD (Create, Read, Update, Delete) operations, which managed the user data, cart information, and product details.

Throughout the implementation process, a strong focus on data integrity, security, and scalability was maintained. By leveraging MongoDB and carefully structuring the collections based on the ERD, efficient data management and flexibility for future expansion was ensured.

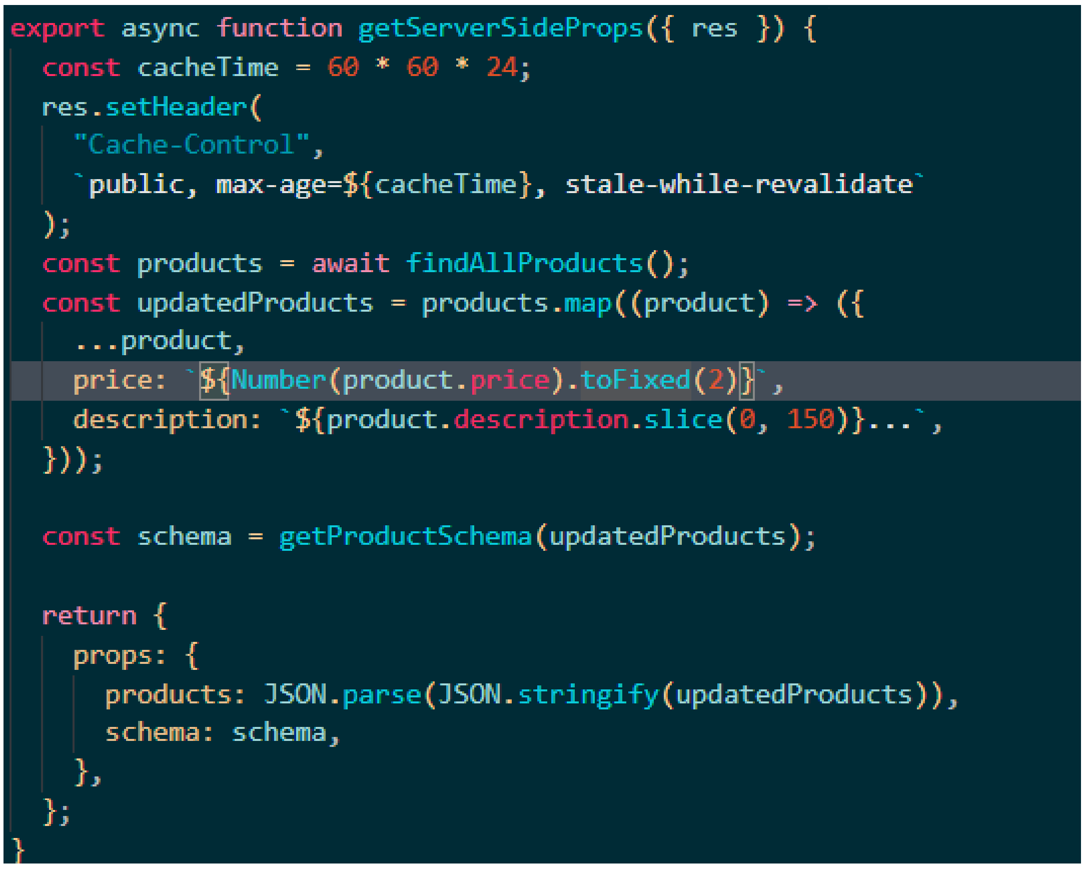
Overall, the successful implementation of the database for the Ecommerce project involved creating the necessary project structure, setting up a MongoDB cluster, mapping collections based on the ERD, configuring the connection string, and organizing the code in a way that promotes reusability and maintainability.

4.5 Implementation Of Server Side Rendering

Server-side rendering is a powerful technique that enables the application to generate dynamic content on the server before sending it to the client. This approach has significant advantages, particularly in terms of search engine optimization (SEO), as it allows search engine crawlers to access and understand the content more effectively.

The application leverages the benefits of server-side rendering by utilizing the getServerSideProps function that was researched in section 2.2.4. This function is unique to each page and is executed on the server side before the client instance. It provides a way to fetch and manipulate data from the database, ensuring that the rendered page contains the necessary dynamic content.

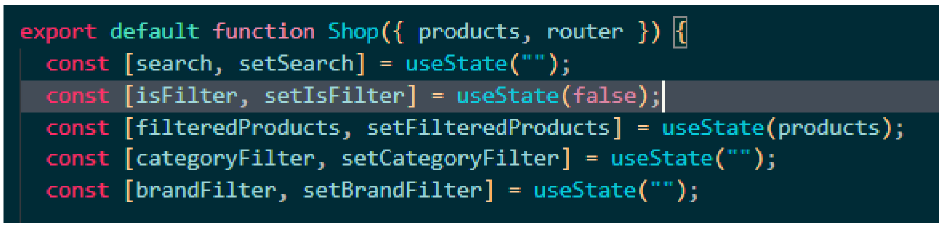
Specifically, the use of getServerSideProps function to retrieve product data from the database in JSON format. This data is then mapped into components, allowing for dynamically render elements on the page. By collecting the product data on the server side, the content is readily available and can be efficiently processed by search engine crawlers.



```
export async function getServerSideProps({ res }) {
  const cacheTime = 60 * 60 * 24;
  res.setHeader(
    "Cache-Control",
    `public, max-age=${cacheTime}, stale-while-revalidate`
  );
  const products = await findAllProducts();
  const updatedProducts = products.map((product) => ({
    ...product,
    price: `${Number(product.price).toFixed(2)}`,
    description: `${product.description.slice(0, 150)}...`,
  }));

  const schema = getProductSchema(updatedProducts);

  return {
    props: {
      products: JSON.parse(JSON.stringify(updatedProducts)),
      schema: schema,
    },
  };
}
```

In Figure, the getServerSideProps function retrieves product data from the API and prepares them as props for the components.

The getServerSideProps function acts as a bridge between the application and the server. It enables fetching data from the database via the API endpoint and processes it before rendering the page. This way, the necessary product data is readily available and can be seamlessly passed as props to the corresponding components.



```
export default function Shop({ products, router }) {
  const [search, setSearch] = useState("");
  const [isFilter, setIsFilter] = useState(false);
  const [filteredProducts, setFilteredProducts] = useState(products);
  const [categoryFilter, setCategoryFilter] = useState("");
  const [brandFilter, setBrandFilter] = useState("");
```

The figure shows how the props are retrieved and passed to child components or used within the component itself.

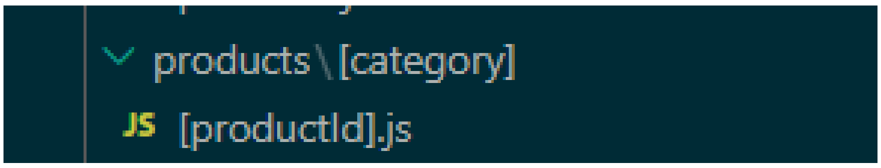
By incorporating server-side rendering and leveraging getServerSideProps, the overall performance is improved, SEO-friendliness, and user experience of the application. The dynamic content fetched from the database adds a layer of interactivity and customization, ensuring that each page is tailored to the specific needs and preferences of the users.

4.6 Implementation Of Route Masking

Route masking is a valuable technique that modifies the appearance of URLs, particularly when dealing with dynamic pages. By utilizing square brackets "[]" around specific sections of the filename, dynamic URLs that adapt to different content are created.

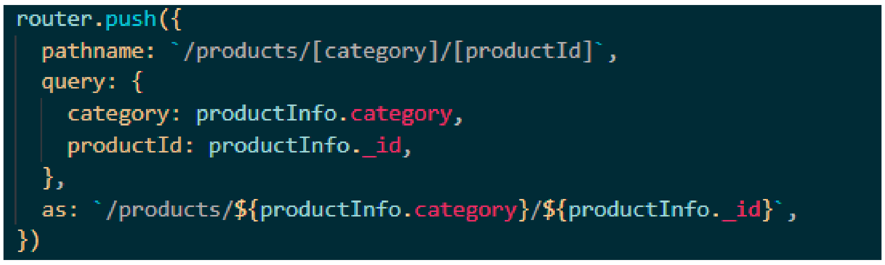
This approach offers several advantages, with one of the most significant benefits being its positive impact on search engine optimization (SEO). When search engines crawl websites, they pay attention to the structure and readability of URLs. By employing route masking and creating meaningful URLs, understandability and SEO performance is improved of the website.

Dynamic URLs generated through route masking provide valuable information about the content and context of a particular page. This makes it easier for search engine bots to interpret and index the website's pages accurately. Clear and descriptive URLs not only enhance SEO but also contribute to a more user-friendly browsing experience.



```
products\[category]
  [productId].js
```

The figure shows the implementation of dynamic pages in the application, specifically when navigating to individual product pages. It utilizes the power of dynamic routing by using [category].js to define the corresponding category and [productId].js to pass the product ID.



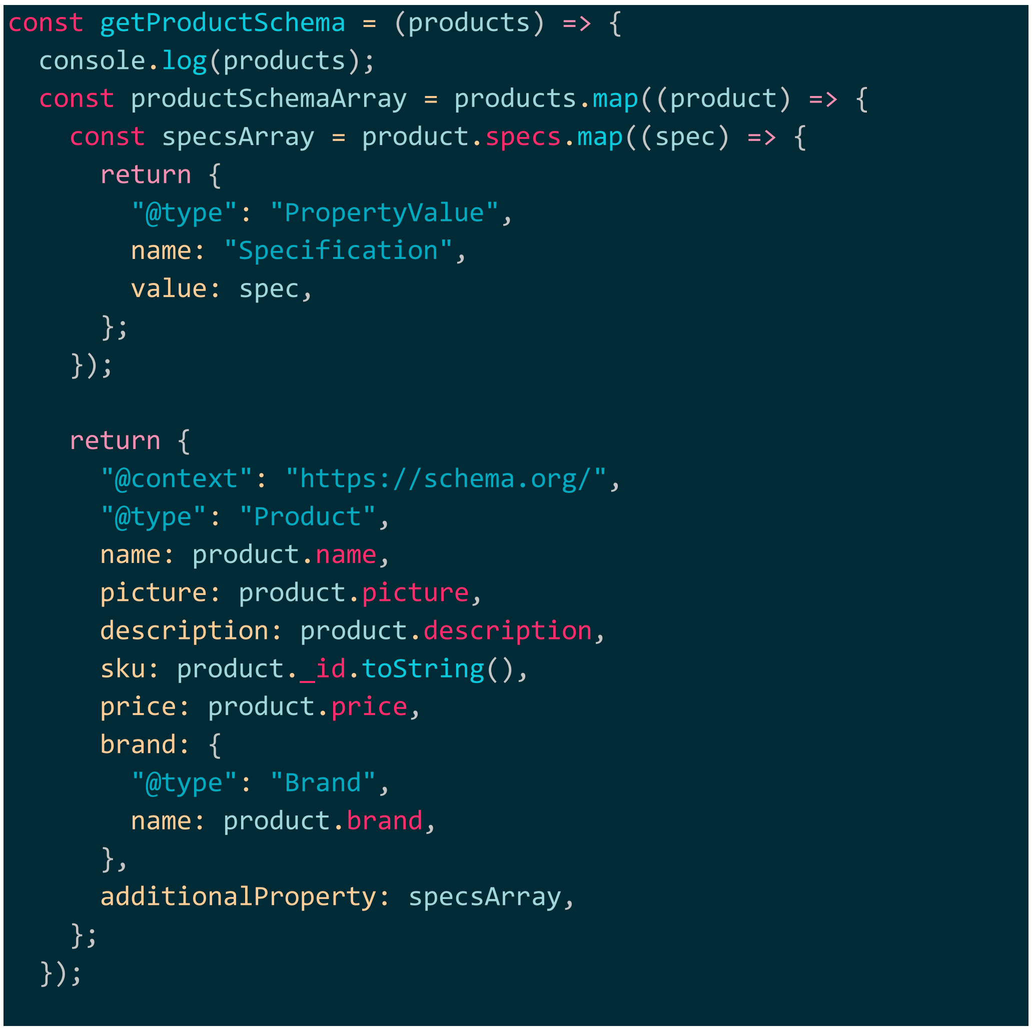
```
router.push({
  pathname: `/products/[category]/[productId]`,
  query: {
    category: productInfo.category,
    productId: productInfo._id,
  },
  as: `/products/${productInfo.category}/${productInfo._id}`,
})
```

The Figure shows next step in the process, which involves utilizing router.push to navigate to the appropriate file. By leveraging the ‘as’ property, from section 2.2.4, during this navigation, allows for the ability to modify the appearance of the URL. This technique allows the application to pass the relevant category and productId, along with any additional query parameters, to the next file that will be rendered. By passing this information, the subsequent page can be access the necessary data for proper functionality.

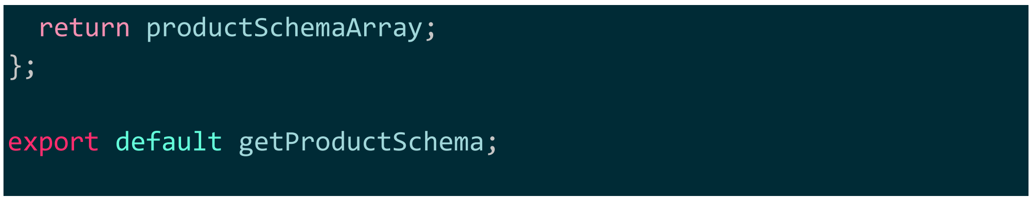
In summary, the implementation of route masking allows the application to create dynamic URLs that are search engine-friendly and human-readable. By adopting this technique, we improve the SEO performance of the website, enhance user experience, and contribute to better discoverability and accessibility of the content.

4.4 Implementation Of Schema

Implementing schemas enhances the search engine optimization (SEO) of the website. Schemas provide a structured format that communicates specific information about the content to search engines, making it easier for them to interpret and index the website accurately. This approach helps search engines better understand the context, meaning, and relevance of the content, resulting in improved visibility and ranking in search engine results pages. By leveraging schemas in headers, we optimize the website's discoverability and ensure that it effectively communicates its value to search engines, ultimately driving organic traffic and attracting relevant users.

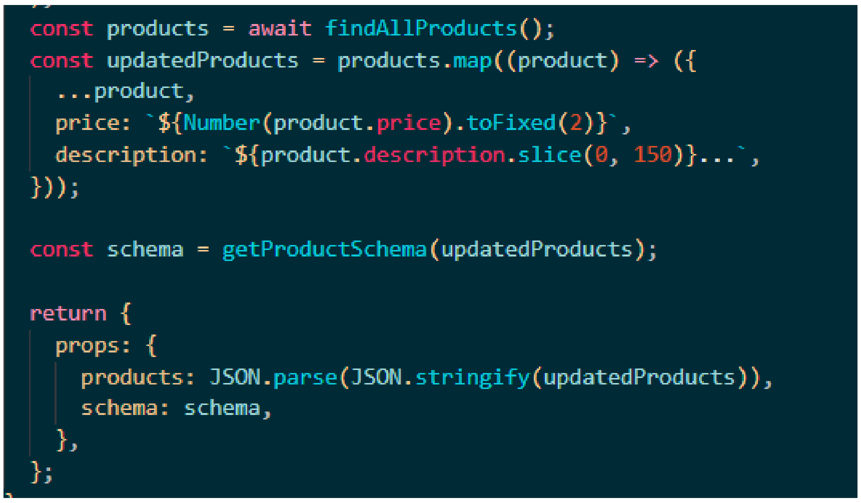


```
const getProductSchema = (products) => {
  console.log(products);
  const productSchemaArray = products.map((product) => {
    const specsArray = product.specs.map((spec) => {
      return {
        "@type": "PropertyValue",
        name: "Specification",
        value: spec,
      };
    });

    return {
      "@context": "https://schema.org/",
      "@type": "Product",
      name: product.name,
      picture: product.picture,
      description: product.description,
      sku: product._id.toString(),
      price: product.price,
      brand: {
        "@type": "Brand",
        name: product.brand,
      },
      additionalProperty: specsArray,
    };
  });
```



```
  return productSchemaArray;
};

export default getProductSchema;
```

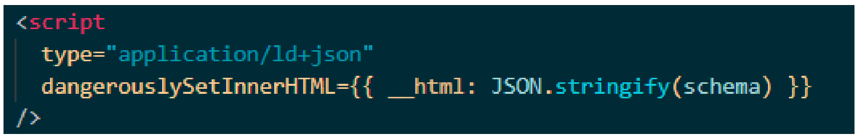
The Figure shows the code is creating a structured schema object for each product. This structured data includes properties such as the product's name, picture, description, SKU, price, brand, and additional specifications. Each property is assigned a specific key-value pair according to the schema.org specifications. The schema can now be used on the pages.



```
const products = await findAllProducts();
const updatedProducts = products.map((product) => ({
  ...product,
  price: `${Number(product.price).toFixed(2)}`,
  description: `${product.description.slice(0, 150)}...`,
}));

const schema = getProductSchema(updatedProducts);

return {
  props: {
    products: JSON.parse(JSON.stringify(updatedProducts)),
    schema: schema,
  },
};
```

In Figure, we can see the process of setting up the product schema. First, the application collects the products from the API. These products are then passed as a parameter into the getProductSchema function shown in the previous figure. This function iterates over each product, extracts the necessary information, and constructs a structured schema object using the schema.org vocabulary.

Once the schema objects are created, they are assigned to a variable. The schema is then passed as a prop to the relevant pages where it can be used to enrich the content. By incorporating the product schema, we provide search engines with valuable structured data that helps them understand and categorize the products more effectively. This can result in improved visibility and presentation of the products in search results.



```
<script
  type="application/ld+json"
  dangerouslySetInnerHTML={{ __html: JSON.stringify(schema) }}
/>
```

The Figure, shows the utilization of the schema within the head section of the HTML document. By including the type="application/ld+json" attribute in the script tag, it indicates that the content inside is JSON-LD data. JSON-LD is a structured data format that offers additional context to search engines.

4.4 Implementation Of Tailwind CSS

To implement Tailwind CSS, the command 'npm install tailwind autoprefixer postcss' was utilized. By incorporating Tailwind CSS into the project, a responsive and visually appealing interface for the website was created, thereby enhancing its SEO performance.

Tailwind CSS emerged as a valuable tool in improving the website's interface. Its extensive collection of utility classes provided the ability to style various elements quickly and efficiently, aligning with the design requirements. Leveraging Tailwind's responsive utility classes, adaptive designs were crafted, seamlessly adapting to different devices and screen sizes.

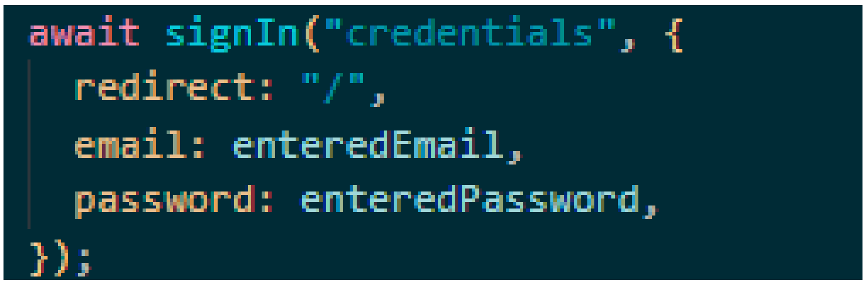
Implementing responsive design is crucial for SEO as search engines prioritize websites that offer a seamless user experience across multiple devices. Leveraging Tailwind CSS's responsive design capabilities ensures that the website provides optimal viewing experiences, potentially increasing user engagement and positively impacting SEO.

One of the key advantages of Tailwind CSS is its flexibility and customization options, it allows the styling to match the brand's identity. This cohesive and distinctive design language aimed to enhance user satisfaction, which can positively influence SEO metrics such as time-on-site and conversions.

In conclusion, the integration of Tailwind CSS created a visually stunning and responsive interface for the website. The utility-first approach, customization options, and performance optimizations provided by Tailwind CSS collectively contributed to an enhanced user experience and improved SEO. By aligning with search engine guidelines, the website gained increased visibility and discoverability, leading to its overall success.

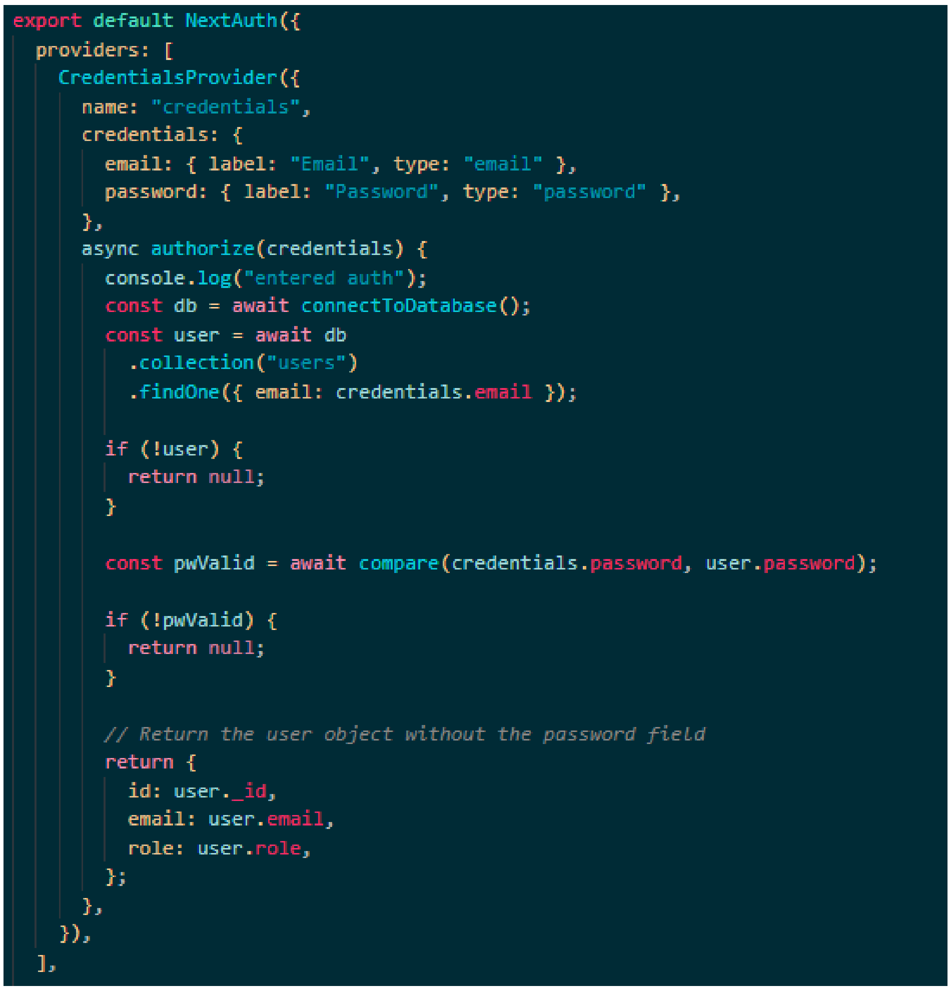
4.4 Implementation Of Next Auth

To implement Next-auth, the command ‘npm install next-auth bcrypt’ was used. Next-auth is a powerful serverless token-based authentication library that was integrated into the project. It allowed the application to create a secure and seamless authentication flow for the website.



```
await signIn("credentials", {
  redirect: "/",
  email: enteredEmail,
  password: enteredPassword,
});
```

The figure observes the utilization of next-auth's signIn function, which plays a crucial role in the authentication process. Invoking the signIn function and passing the necessary parameters, such as email and password, authenticates the user.



```
export default NextAuth({
  providers: [
    CredentialsProvider({
      name: "credentials",
      credentials: {
        email: { label: "Email", type: "email" },
        password: { label: "Password", type: "password" },
      },
      async authorize(credentials) {
        console.log("entered auth");
        const db = await connectToDatabase();
        const user = await db
          .collection("users")
          .findOne({ email: credentials.email });

        if (!user) {
          return null;
        }

        const pwValid = await compare(credentials.password, user.password);

        if (!pwValid) {
          return null;
        }

        // Return the user object without the password field
        return {
          id: user._id,
          email: user.email,
          role: user.role,
        };
      },
    }),
  ],
```

The Figure, observes the implementation of a provider called "credentials," which plays a crucial role in authorizing the user by establishing a connection to the database. By integrating MongoDB, the application can search for the user within the collection using their unique email as the identifier.

To ensure secure authentication, a comparison between the password provided by the user in the credentials and the encrypted password stored in the database is performed. This comparison is made possible by utilizing the "compare" function from the bcrypt library, which decrypts the encrypted password in the database and verifies its match with the provided password.

Once the user's credentials are successfully authenticated, we retrieve their details from the database. These details include relevant information such as their username, email, and any additional user-specific data. This user data is then returned, allowing us to generate an authentication token for further use within the application.

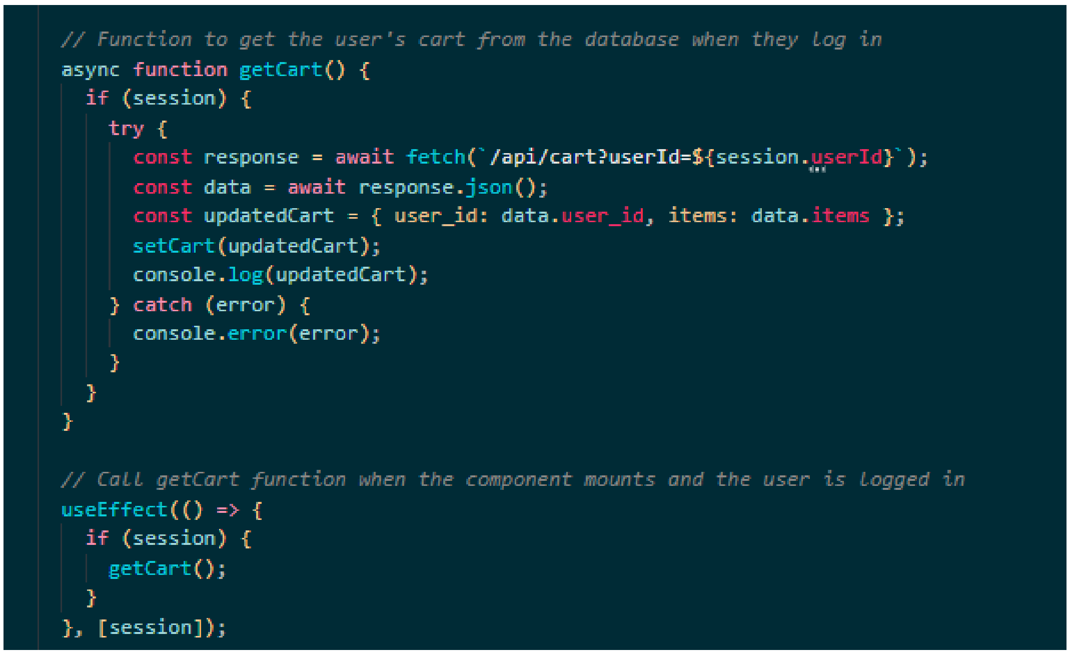
By integrating the credentials provider, a secure and reliable authentication process is established. By connecting to the database, retrieving user details, and verifying passwords using bcrypt, the confidentiality and integrity of user data is ensured. The tokenization of user details allows for seamless access to protected areas and personalized features within the application.

In summary, Figure illustrates the implementation of the credentials provider, which connects to the database and authorizes the user. By searching for the user using their email, comparing the provided password with the encrypted password, and returning user details for tokenization, a secure authentication flow is established. This integration ensures the protection of user data and facilitates a seamless and personalized user experience throughout the application.

4.4 Implementation Of RESTFul API

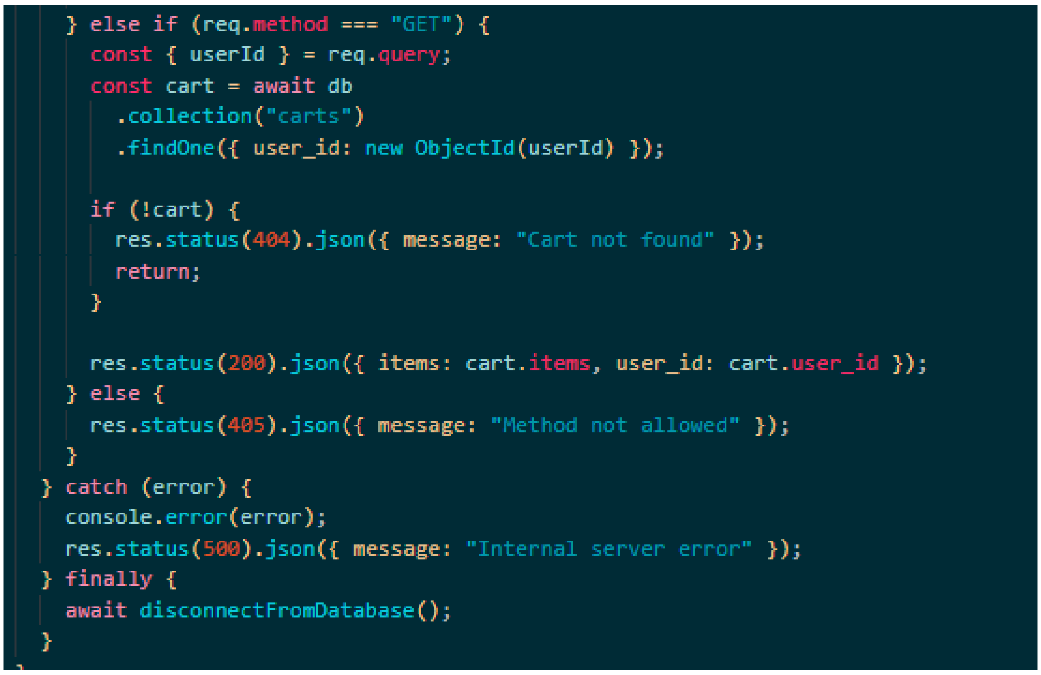
To incorporate CRUD (Create, Read, Update, Delete) functionality, a RESTful backend for the carts and products databases is implemented. This involved creating an API that would handle the necessary operations for managing cart and product data.

Additionally, a Cart React Context was used, which allowed efficient management and manipulation of local states related to the shopping carts. This context facilitated seamless communication with the Cart API, ensuring that the database remained up to date with any changes made.



```
// Function to get the user's cart from the database when they log in
async function getCart() {
  if (session) {
    try {
      const response = await fetch(`/api/cart?userId=${session.userId}`);
      const data = await response.json();
      const updatedCart = { user_id: data.user_id, items: data.items };
      setCart(updatedCart);
      console.log(updatedCart);
    } catch (error) {
      console.error(error);
    }
  }
}

// Call getCart function when the component mounts and the user is logged in
useEffect(() => {
  if (session) {
    getCart();
  }
}, [session]);
```

The Figure, shows the implementation of the getCart function within the Cart React Context. This function is responsible for retrieving the user's cart by making an API call.



```
    } else if (req.method === "GET") {
      const { userId } = req.query;
      const cart = await db
        .collection("carts")
        .findOne({ user_id: new ObjectId(userId) });

      if (!cart) {
        res.status(404).json({ message: "Cart not found" });
        return;
      }

      res.status(200).json({ items: cart.items, user_id: cart.user_id });
    } else {
      res.status(405).json({ message: "Method not allowed" });
    }
  } catch (error) {
    console.error(error);
    res.status(500).json({ message: "Internal server error" });
  } finally {
    await disconnectFromDatabase();
  }
```

In Figure, the actions were seamlessly synchronized with the backend database through the Cart API calls.

The RESTful backend, combined with the Cart React Context, provided a robust foundation for managing cart and product data. This approach allowed essential CRUD operations to be implemented while maintaining a synchronized and up-to-date database.

In summary, by implementing a RESTful backend and utilizing the Cart React Context, CRUD functionality was successfully added to the application. This enabled efficient management of the carts and products databases, ensuring that changes made locally were accurately reflected in the backend. The combination of these technologies facilitated a seamless and synchronized experience for users interacting with the shopping carts and product data.

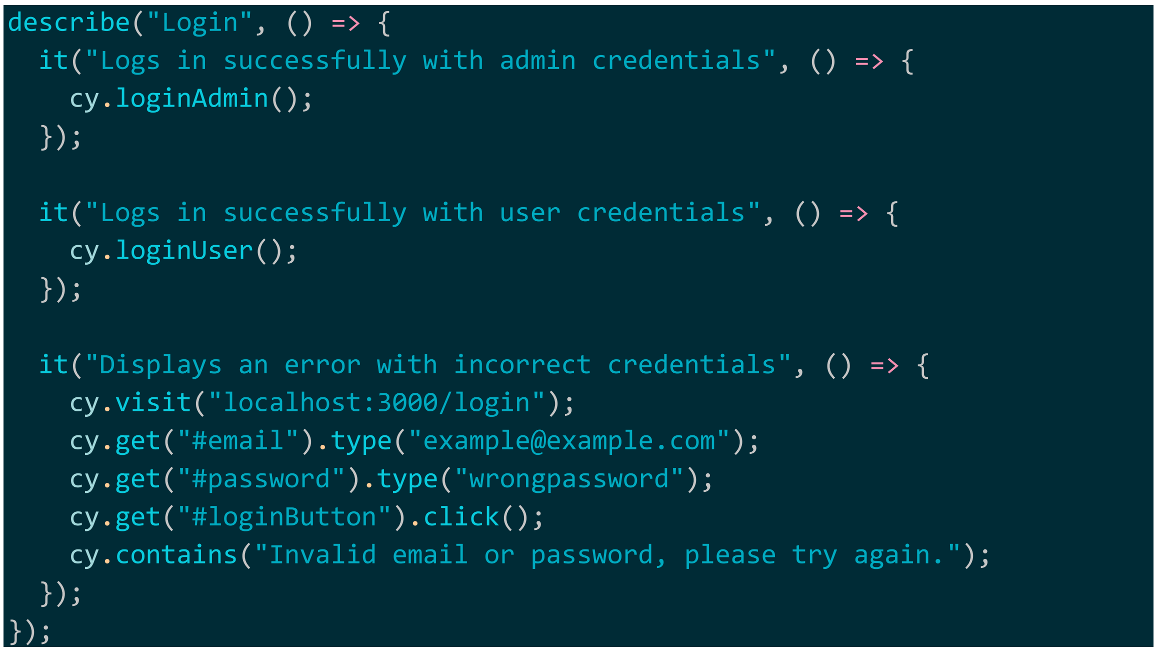
4.4 Implementation Of Testing

To incorporate testing into the application, Cypress was utilized, a powerful testing framework. The creation of test scripts ensured the main functionality of the application was working as expected. These unit tests allowed validation of various components and features, ensuring they functioned correctly and met the desired requirements.

In addition to Cypress, Cheerio was used, a library that helps with server-side rendering testing. Cheerio was utilized to determine whether the dynamic content was properly rendered on the server side, ensuring it was ready to be crawled by search engines. By validating the server-side rendering of the application, we can confirm that the necessary data and HTML structure were correctly generated, providing an optimized experience for search engine crawlers.

4.4.1 Implementation Of Cypress

To incorporate Cypress into the testing process, the tests were organized into a dedicated folder structure. Within the "cypress" folder, an "e2e" (end-to-end) folder was created which had scripts for various functionalities such as authentication, cart, login, search, and signup. This allowed for thorough tests in different aspects of the application and ensure they were functioning correctly.



```
describe("Login", () => {
  it("Logs in successfully with admin credentials", () => {
    cy.loginAdmin();
  });

  it("Logs in successfully with user credentials", () => {
    cy.loginUser();
  });

  it("Displays an error with incorrect credentials", () => {
    cy.visit("localhost:3000/login");
    cy.get("#email").type("example@example.com");
    cy.get("#password").type("wrongpassword");
    cy.get("#loginButton").click();
    cy.contains("Invalid email or password, please try again.");
  });
});
```

The Figure, observes the login testing process. We conducted tests to validate successful logins from both user and admin perspectives. Additionally, we performed tests to verify the handling of incorrect login attempts.

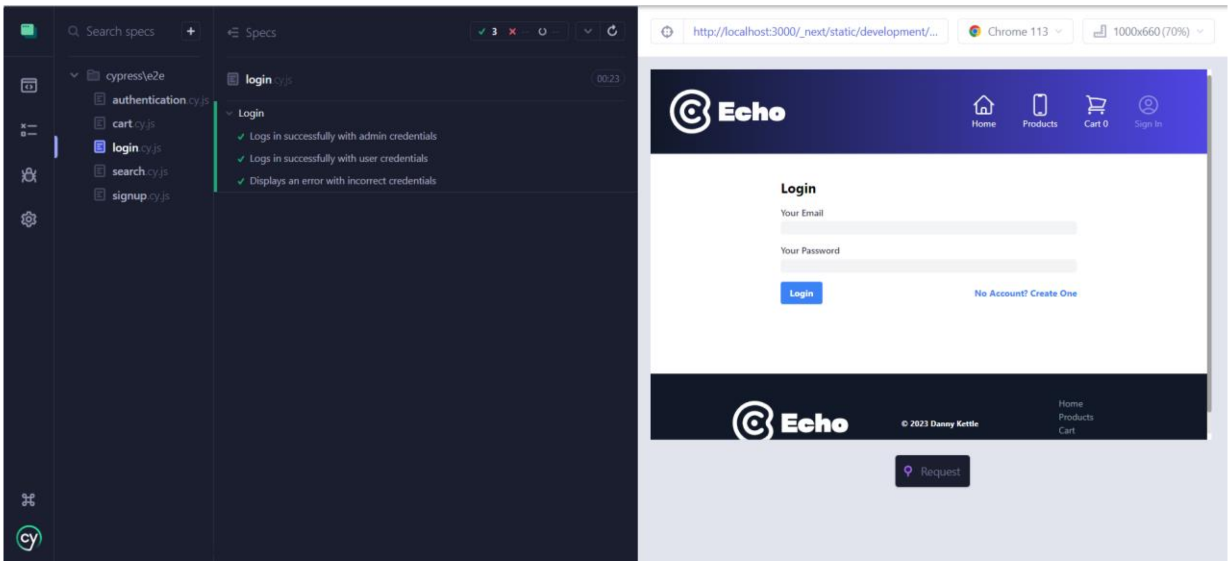
In addition, we utilized the "commands.js" file within the "support" folder. This file allowed us to create reusable and callable commands that could be utilized throughout the test scripts. By leveraging these custom commands, the efficiency and readability of the tests were improved.

```
Cypress.Commands.add("loginAdmin", () => {
  cy.visit("localhost:3000/login");
  cy.get("#email").type("admin@admin.com");
  cy.get("#password").type("root");
  cy.get("#loginButton").click();
  cy.wait(2000); // wait for the page to reload
  cy.url().should("include", "/");
});

Cypress.Commands.add("loginUser", () => {
  cy.visit("localhost:3000/login");
  cy.get("#email").type("test@test.com");
  cy.get("#password").type("test78");
  cy.get("#loginButton").click();
  cy.wait(2000); // wait for the page to reload
  cy.url().should("include", "/");
});
```

In the figure, can observe the commands used in the previous figure which logged the user in successfully.

During the testing, I carefully examined the content displayed on the page to assess the success of the login tests. I checked for specific elements and expected content to ensure that the login functionality behaved as intended.

By conducting these tests, I was able to validate the login feature's correctness and reliability. This comprehensive testing approach helped us identify any potential issues, improve the user experience, and enhance the overall quality of the application.



```
login.cy.js                                        00:23
Login
  ✓ Logs in successfully with admin credentials
  ✓ Logs in successfully with user credentials
  ✓ Displays an error with incorrect credentials
```

Figure 21 : Cypress Testing

The Figure, shows the success of the login tests.

Overall, Cypress testing proved to be highly beneficial for the project. It enabled us to identify and fix small errors, ensuring the final product met the quality standards. By thoroughly testing different functionalities, there is an increased confidence in the reliability and performance of the application. Cypress provided us with a robust testing framework that helped streamline the development process and deliver a polished product to the users.

4.4.2 Implementation Of Cheerio

```
const cheerio = require('cheerio');
const axios = require('axios');

async function checkVisibleContent() {
  try {
    const url = 'http://localhost:3000';
    const response = await axios.get(url);
    const $ = cheerio.load(response.data);

    $('*').each((i, el) => {
      const text = $(el).text().trim();
      const tag = el.tagName.toLowerCase();

      console.log(`${tag}: ${text} `);
    });
  } catch (error) {
    console.error(error);
  }
}

checkVisibleContent();
```

Cheerio is a powerful server-side HTML parsing library that played a crucial role in the web crawling process. Since Cheerio operates on the server side, it allowed us to retrieve and parse the HTML content of the local host. By using Axios, I obtained the response from the server and passed it to Cheerio for further analysis as shown in figure.

With Cheerio, I was able to separate the HTML response into individual elements, providing us with granular control over the content as shown in figure. This allowed us to specifically target and examine the dynamic content that I wanted to crawl. By checking if the dynamic content could be successfully accessed and parsed by Cheerio, I could ensure its visibility to search engine crawlers.

This combination of Axios and Cheerio proved to be effective in the SEO strategy, as it enabled us to verify that the dynamic content was readily available for search engine indexing. By ensuring that the dynamic content was accessible and crawlable, the chances of search engines recognizing and ranking the website's valuable information was increased.

Chapter 5

Evaluation

5.1 Introduction

This chapter aims to evaluate the application's effectiveness in achieving its goal of creating an SEO-friendly single-page application ecommerce site. It will analyze key metrics related to SEO performance, research findings, visual user-friendliness, the success of the payment gateway, and website crawling. This evaluation will provide insights into the application's overall performance and guide future improvements.

5.2 Quantitative Site metrics

5.2.1 Results

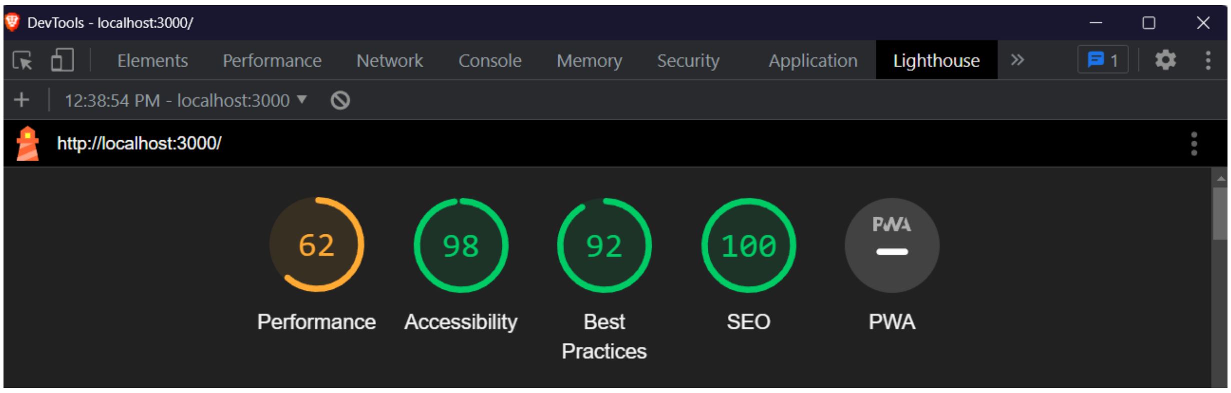


Figure 21

Figure 22 : In browser performance metrics

SEO

In Figure, the SEO performance of the application can be observed, as it achieved a perfect score of 100 in the Lighthouse metrics. This signifies that the application has effectively implemented SEO best practices and optimizations, resulting in a highly favorable performance in terms of search engine visibility and discoverability.

The perfect score in SEO metrics indicates that the application's content is well-structured, properly indexed, and optimized for search engines. This enhances the website's chances of appearing prominently in search engine results, attracting organic traffic, and improving its overall visibility to potential users.

The Figure also displays a high accessibility score of 98, indicating that the application has taken significant measures to ensure that it is accessible to users with diverse abilities and needs.

Accessibility

The accessibility score of 98 implies that the application adheres to best practices and guidelines for web accessibility, making it easier for individuals with disabilities to navigate, perceive, and interact with the website. This includes considerations such as providing alternative text for images, using proper heading structure, ensuring keyboard accessibility, and maintaining a clear and readable design.

By achieving a high accessibility score, the application demonstrates its commitment to inclusivity and providing an optimal user experience for all individuals, regardless of their abilities. It ensures that people with visual impairments, motor disabilities, other accessibility needs can access and engage with the website's content effectively.

Best Practices

The figure highlights an impressive score of 92 in the "Best Practices" metric, indicating that the application follows industry-standard coding practices and adheres to recommended guidelines for web development.

A score of 92 in "Best Practices" signifies that the application has been implemented with high-quality code, efficient resource usage, and optimal performance. It suggests that the development has taken measures to ensure that the application meets best practices in areas such as security, performance optimization, code readability, and maintainability.

By achieving a notable score in "Best Practices," the application demonstrates its commitment to delivering a robust and reliable user experience. It signifies that the development process has considered industry standards and guidelines, resulting in a well-structured and technically sound application.

Performance

The performance indicator was significantly affected due to image sizing issues. The application was resizing images using CSS, which resulted in suboptimal performance instead of rendering appropriately sized images.

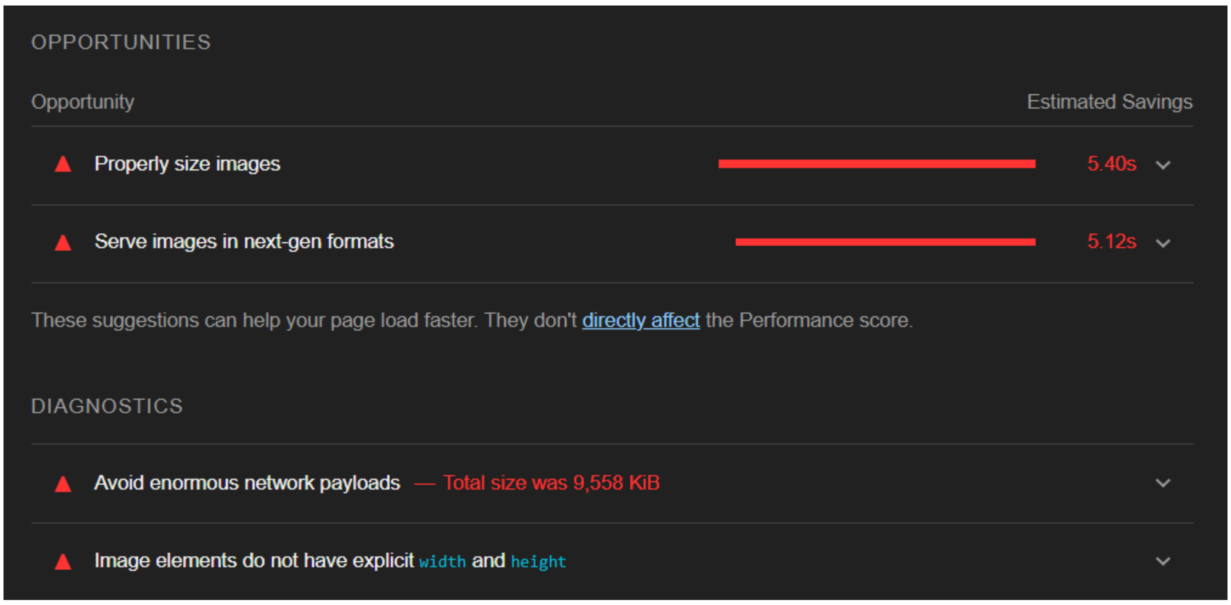
By relying on CSS for image resizing, the application incurred unnecessary overhead in terms of file size and loading times. This led to a decrease in overall performance and negatively impacted the user experience.

To improve performance, it is recommended to implement a more efficient approach for image sizing. This can be achieved by optimizing the images before uploading them to the application, ensuring that they are appropriately sized and optimized for web delivery. Additionally, implementing responsive image techniques, such as using the <picture> element or srcset attribute, can help deliver the correct-sized images to different devices and screen sizes, further enhancing performance.

Addressing the image sizing issue and optimizing the way images are handled can lead to significant improvements in performance, resulting in faster page load times and enhanced user satisfaction. It is crucial to prioritize image optimization as part of the application's performance optimization strategy to ensure an optimal user experience.

By optimizing image sizes and adopting best practices for image delivery, the application can achieve better performance indicators, improve loading times, and create a more efficient and seamless user experience

5.2.2 Opportunities for improvement



OPPORTUNITIES

Opportunity	Estimated Savings
Properly size images	5.40s
Serve images in next-gen formats	5.12s

These suggestions can help your page load faster. They don't directly affect the Performance score.

DIAGNOSTICS

Avoid enormous network payloads — Total size was 9,558 KiB

Image elements do not have explicit `width` and `height`

Figure 22

Figure 23 : In browser opportunities for improvement

The figure provides valuable insights into specific areas where the performance of the website can be improved. It highlights several opportunities to enhance performance, including:

Properly sizing images: Resizing images to their appropriate dimensions can significantly improve performance by reducing file size and optimizing loading times.

Serving images in next-gen formats: Utilizing modern image formats, such as WebP or AVIF, can further reduce file sizes without compromising visual quality, leading to faster image loading and improved overall performance.

Avoiding network payloads: Minimizing the amount of data transferred over the network, such as reducing unnecessary HTTP requests, utilizing caching mechanisms, and employing compression techniques, can help optimize performance.

Applying explicit width and height to images: Providing explicit dimensions (width and height) for images in HTML can prevent layout shifts and ensure that sufficient space is reserved for the images during page rendering, resulting in a smoother user experience.

By implementing these performance optimization techniques, the website can enhance its loading speed, reduce bandwidth consumption, and deliver a more seamless and efficient user experience. It is recommended to prioritize these improvements to achieve optimal performance and meet user expectations.

5.2.3 Conclusion

The quantitative site metrics revealed promising outcomes in terms of SEO performance. The application scored a perfect 100 in the Lighthouse metrics, indicating successful implementation of SEO best practices and optimizations. This high score ensures improved search engine visibility and discoverability, enhancing the website's chances of attracting organic traffic and reaching potential users.

Additionally, the application showcased a commendable accessibility score of 98. This signifies its adherence to web accessibility guidelines, ensuring that individuals with disabilities can navigate and interact with the website effectively. By prioritizing inclusivity, the application provides an optimal user experience for all users, regardless of their abilities.

Moreover, the application demonstrated adherence to industry-standard coding practices and guidelines, earning a noteworthy score of 92 in the "Best Practices" metric. This reflects the commitment to delivering a reliable and robust user experience, with a focus on security, performance optimization, code readability, and maintainability.

However, the performance indicator highlighted an opportunity for improvement. The suboptimal performance resulted from image sizing issues, where images were resized using CSS instead of rendering appropriately sized images. To enhance performance, it is recommended to optimize images before uploading them to the application and implement responsive image techniques to deliver correct-sized images to different devices and screen sizes.

Addressing these opportunities for improvement will lead to faster page load times, improved overall performance, and enhanced user satisfaction.

In conclusion, the evaluation chapter provided valuable insights into the application's success in achieving its objectives. It showcased impressive SEO performance, adherence to accessibility guidelines, and adherence to best practices in web development. By addressing the identified areas for improvement, the application can further enhance its performance, deliver an optimal user experience, and solidify its position as an SEO-friendly ecommerce site.

5.3 Payment Gateway

To implement the payment gateway, Stripe was chosen as the preferred solution due to its robust features, secure payment processing, and developer-friendly integration. Stripe provides a seamless payment experience for customers and offers a wide range of payment options, including credit cards, digital wallets, and local payment methods.

By utilizing Stripe as the payment gateway, the application benefits from its advanced fraud detection mechanisms, PCI compliance, and reliable infrastructure. Stripe's documentation and developer resources also make it easier to integrate and customize the payment flow according to the specific requirements of the application.

Overall, choosing Stripe as the payment gateway ensures a secure and efficient payment process, instilling confidence in customers and facilitating successful transactions.

5.4 Website Crawl

Section 4.4.2 demonstrated the effectiveness of the crawl using Cheerio in identifying the rendering of dynamic content on the server side, making it readily accessible to search engines. This successful implementation played a crucial role in achieving the objective of creating an SEO-friendly Single-Page Application (SPA).

By ensuring that dynamic content is properly rendered on the server side, I have improved the website's visibility and discoverability by search engines. This allows search engines to index and rank the content accurately, increasing the likelihood of attracting organic traffic and enhancing the overall SEO performance of the application.

The successful implementation of server-side rendering for dynamic content showcases the commitment to optimizing the SPA for search engine crawlers. By providing search engines with complete and relevant content during the crawl process, I have effectively improved the SEO-friendliness of the application.

Overall, the success in making the SPA SEO-friendly through the proper rendering of dynamic content on the server side sets a solid foundation for maximizing the website's visibility and attracting organic traffic.

Chapter 6

Conclusion

6.1 Introduction

This chapter provides an overview and summary of the various stages of the project. The first section presents a summary of each chapter, highlighting the key elements that were discussed.

The next section focuses on the aims and objectives outlined in the terms of reference (TOR). It evaluates whether these aims and objectives have been achieved and includes a discussion on how the objectives were met and any that were not achieved.

Additionally, this chapter explores the scope for future work, outlining potential improvements for the project and opportunities to enhance the system with new features and technologies.

The final section offers the author's reflection on the project, providing insights and personal perspectives. It concludes the chapter with an overall summary and conclusion of the project as a whole.

6.2 Chapter Overview

This section aims to evaluate the success of the chapters and provide a concluding perspective.

6.2.1 Introduction

This chapter serves as an introduction to the topic of optimizing React-based Single Page Applications (SPAs) for Search Engine Optimization (SEO). It begins by acknowledging the increasing significance of online shopping and the need for businesses to have a strong online presence. It highlights the challenges faced by SPAs in terms of SEO, especially in the context of ecommerce sites where factors like fast load times and real-time updates are crucial.

The chapter then focuses on exploring how React and Next.js can be leveraged to optimize SPAs for SEO. It introduces Server-Side Rendering (SSR) as a technique to enhance initial load times and improve SEO ranking. It also delves into the benefits of caching, React memos, and callbacks in boosting performance for SEO purposes. Moreover, the chapter discusses the use of routing masks to create more SEO-friendly URLs.

Furthermore, the chapter showcases the aims and objectives of the project along as well as a brief overview of the chapters.

Overall, this chapter provides a comprehensive overview of the main topics that will be covered in the subsequent sections. It sets the stage for further exploration and analysis of optimizing React SPAs for SEO, offering readers valuable insights and guidance in creating SEO-friendly applications.

6.2.2 Literature Review

The chapter serves as a literature review, presenting an overview of existing research on Search Engine Optimization (SEO) and Single Page Applications (SPAs). It focuses on examining the importance, benefits, and challenges associated with SEO and SPAs. Additionally, the chapter provides insights into various strategies and techniques that can be utilized to optimize websites and enhance user experiences. Overall, the chapter offers a comprehensive analysis of the literature, providing a foundation for the subsequent chapters and contributing to a deeper understanding of the topics.

6.2.3 Design

In this chapter, the design process of an SEO ecommerce Single Page Application (SPA) is discussed. The design approach focuses on creating a visually appealing interface with a simple and intuitive user experience. Research and analysis of websites like Amazon and Petsmart were conducted to understand design principles and user preferences. User personas were created to identify specific design needs. The user interface design includes low fidelity wireframes and high fidelity designs, emphasizing placement of key elements, detailed product information, and easy comparisons. Implementation changes, responsive design, and database design using MongoDB were also discussed. Use cases such as the checkout process, search functionality, and product filtering were considered to enhance the user experience. The chapter concludes by highlighting the importance of user-centric design and the integration of SEO techniques for improved visibility and organic traffic.

6.2.4 Implementation

This chapter focuses on the implementation of the proposed solution. It begins with an introduction to the specific tasks and activities carried out during the implementation phase. The chapter highlights the decision to temporarily drop the review system functionality in order to optimize the database performance and prioritize other key elements. The implementation of the database is discussed, including setting up the project, creating collections, and establishing a connection with MongoDB. The chapter also covers the implementation of server-side rendering, route masking, schema for search engine optimization, Tailwind CSS for responsive design, Next Auth for authentication, and a RESTful API for CRUD operations. Each implementation is explained with code snippets and figures to illustrate the process and the benefits it brings to the website's functionality, performance, and search engine optimization.

6.2.4 Evaluation

This chapter evaluates the application's performance in creating an SEO-friendly single-page application e-commerce site. The quantitative site metrics reveal a perfect score in SEO, indicating effective implementation of SEO best practices. The application also scores high in accessibility and adheres to best practices in web development. However, there are opportunities for improvement, particularly in image optimization and performance. The chapter recommends properly sizing images, utilizing next-gen image formats, minimizing network payloads, and providing explicit dimensions for images. The chosen payment gateway, Stripe, offers secure payment processing and seamless integration. The successful implementation of server-side rendering enhances the website's visibility to search engines. Future improvements include image optimization, temporary carts for non-users, customer reviews, and enhanced product navigation. Overall, addressing these areas will optimize performance, enhance user experience, and ensure long-term success.

6.3 Aims and Objectives

This subsection examines whether the aims and objectives, as previously discussed in the terms of reference and chapter one, have been achieved.

6.3.1 Aims

The overall aim for this project is to build an SEO friendly ecommerce single page application.

- The aim has been met by creating a tech ecommerce website using Next.js

6.3.2 Terms of Reference Aims and Objectives

- Understand the importance of SEO for online businesses and how it can impact website visibility on search engines.
    - This objective has been met by completing chapter 2 Literature Review
- Explore the unique challenges that SPAs, particularly ecommerce sites, pose for SEO.
    - This objective has been met by completing chapter 2 Literature Review, as well as implemented in chapter 4 Implementation.
- Analyze the use of React, a popular front-end JavaScript library for building SPAs, in optimizing websites for SEO.
    - This object is met in chapter 5 Evaluation.

- Discuss the benefits of using Server-Side Rendering (SSR) in Next.js to improve initial load times and SEO ranking.
    - This is has been met in chapter 2 Literature Review, and chapter 4 Implementation

- Investigate the use of routing masks to create more SEO-friendly URLs.
    - This is has been met in chapter 2 Literature Review, and chapter 4 Implementation

- Provide readers with a comprehensive guide on creating an SEO-friendly React SPA that addresses these challenges and implements these solutions.
    - This has been met as discussed in chapter 5 Evaluation.

- Understand design choices for SEO friendliness.
    - This has been met as discussed in chapter 3 Design.

6.4 What went well?

This section discusses what went well within the project, highlighting what was successful.

6.4.1 Literature Review

The process begins with conducting a thorough literature review on Search Engine Optimization (SEO) and Single Page Applications (SPAs), aiming to gain a comprehensive understanding of the existing research in these areas. The review focuses on highlighting the importance, benefits, and challenges associated with SEO and SPAs, providing valuable insights into their significance. By analyzing various strategies and techniques from the literature, the process identifies approaches that can be utilized to optimize websites and enhance user experiences. The literature review serves as a foundation for subsequent steps, shaping the direction of the project and guiding further exploration. Overall, this process contributes to a deeper understanding of SEO and SPAs, informing the subsequent phases of the project and facilitating the development of effective strategies and techniques.

6.4.1 Design

The design process involved a systematic approach to creating a visually appealing and user-friendly interface. Through extensive research and analysis of popular websites, such as Amazon and Petsmart, valuable insights were gained regarding design principles and user preferences. User personas were developed to better understand the target audience and their specific design needs. The design phase encompassed the creation of low fidelity wireframes and high fidelity designs, focusing on strategically placing key elements, providing detailed product information, and facilitating easy product comparisons. Implementation changes, responsive design techniques, and the utilization of MongoDB for database design were also discussed, ensuring adaptability to different devices and efficient data storage. Key use cases like the checkout process, search functionality, and product filtering were addressed to enhance the overall user experience. Throughout the chapter, the significance of a user-centric design approach and the integration of SEO techniques were emphasized as crucial elements for improving visibility and attracting organic traffic to the ecommerce site.

6.4.2 Implementation

The implementation phase of the proposed solution was executed successfully, with a focus on the specific tasks and activities involved. A strategic decision was made to prioritize database performance by temporarily excluding the review system functionality, showcasing optimization considerations. The chapter provides comprehensive insights into the implementation of the database, including project setup, collection creation, and establishing a connection with MongoDB. Additionally, key implementations such as server-side rendering, route masking, schema for search engine optimization, Tailwind CSS for responsive design, Next Auth for authentication, and a RESTful API for CRUD operations are thoroughly discussed. The explanations are supported by code snippets and figures, effectively showcasing the implementation process and highlighting the benefits it brings to the website in terms of functionality, performance, and search engine optimization.

6.4.3 Evaluation

Several aspects of the application's performance in creating an SEO-friendly single-page application e-commerce site went well. The quantitative site metrics highlight a perfect score in SEO, demonstrating the effective implementation of SEO best practices and optimizations. The application also excels in accessibility, with a high score indicating adherence to web accessibility guidelines. Furthermore, the chapter recognizes the application's adherence to best practices in web development, showcasing a commitment to code quality and optimal performance. The chosen payment gateway, Stripe, provides secure payment processing and developer-friendly integration. The successful implementation of server-side rendering contributes to improved visibility for search engines. Future improvements, including image optimization, temporary carts for non-users, customer reviews, and enhanced product navigation, show a proactive approach to enhancing user experience and ensuring the application's long-term success. By addressing these areas, the application can further optimize performance, deliver an exceptional user experience, and solidify its position as an SEO-friendly e-commerce site.

6.5 Problems and the solutions

During the project, several issues were encountered and addressed to improve the application's performance and user experience. One significant problem was the slowdown caused by multiple database connections. To resolve this, react contexts were implemented to ensure global usage of database connections, optimizing the application's performance.

Another challenge faced was creating a user-friendly design. Given the limited product catalog, implementing an extensive navigation menu seemed unnecessary and could potentially overwhelm users. To enhance the user experience, a more streamlined and intuitive design approach was adopted, focusing on providing key features and information without overwhelming the interface with unnecessary elements.

By addressing these problems and implementing suitable solutions, the project was able to improve database performance and enhance the user experience by simplifying the design to align with the specific needs of the application and its user base.

6.6 Future Work

Future improvements to the application involve addressing various aspects to enhance user experience, expand functionality, and optimize performance.

One area for improvement is image optimization. By fixing issues related to incorrect image sizing, adopting next-gen image formats, and incorporating inline width and height attributes, the application can significantly improve page load times and overall performance. Optimizing images ensures faster rendering and a smoother browsing experience for users.

Another enhancement is the implementation of temporary carts for non-users. This feature would allow non-registered users to add items and continue shopping until the checkout process. By reducing friction for potential customers, this improvement can enhance conversion rates and provide a seamless shopping experience.

Introducing customer reviews is another valuable addition to the application. By implementing an orders collection in the database, customers can submit reviews and share their experiences. This social proof element adds credibility, builds trust, and enhances the overall user experience.

As the product catalog expands, it becomes crucial to enhance the product navigation system. While the current inventory may not warrant advanced navigation features, such as brands and categories, it is important to plan for the future. Organizing products into brands and categories can improve discoverability and facilitate a more intuitive browsing experience as the application grows.

These future improvements aim to optimize the application's performance, enhance user experience, and provide a solid foundation for scalability. By addressing these areas, the application can continue to meet user needs, exceed expectations, and ensure its long-term success.

6.7 Personal Reflection

This project has been a valuable learning experience, providing insights into the complexities of developing a larger-scale application and enhancing my project management skills. It has also strengthened my ability to work effectively under pressure.

Throughout the development process, I encountered both successes and challenges. Having prior experience with React from Advanced Programming (6G5Z1101_2122_9Z6), I felt comfortable for the majority of the project. However, I quickly realized that creating an efficient and scalable application requires a deeper understanding of nuances specific to this project, which I learned along the way.

While I initially lacked structure, I was able to conclude the project with dedication and a more organized approach. Reflecting on this experience, I recognize the importance of planning and creating a clear structure from the beginning. If given the opportunity to redo the project, I would prioritize careful planning to ensure a smoother journey.

Despite the initial challenges, I successfully completed the project and delivered an effective solution to the given problem. This experience has provided me with valuable insights, enabling me to further refine my skills and approach in future projects.

Bibliography

Next.js. (n.d.). Get Server-side Props. Retrieved from https://nextjs.org/docs/basic-features/data-fetching/get-server-side-props

Cockroach Labs. (n.d.). What is Connection Pooling? Retrieved from https://www.cockroachlabs.com/blog/what-is-connection-pooling/

Akamai. (n.d.). Akamai Releases Spring 2017 State of Online Retail Performance Report. Retrieved from https://www.akamai.com/newsroom/press-release/akamai-releases-spring-2017-state-of-online-retail-performance-report

Ahrefs. (n.d.). SEO Basics: A Beginner's Guide to SEO. Retrieved from https://ahrefs.com/blog/seo-basics/

Mikowski, T., & Powell, J. (2013). Why single page web applications? In Single page web applications: JavaScript end-to-end (pp. 1-14). Manning Publications.

Konshin, K 2018, *Next. js Quick Start Guide : Server-Side Rendering Done Right*, Packt Publishing, Limited, Birmingham. Available from: ProQuest Ebook Central.
Google Developers. 2021. JavaScript SEO basics. Accessed May 2, 2023. https://developers.google.com/search/docs/guides/javascript-seo-basics.

Moz. (n.d.). Beginner's Guide to SEO. Retrieved May 2, 2023, from https://moz.com/beginners-guide-to-seo.

Molin, E., 2016. Comparison of single-page application frameworks. A method of how to compare Single-Page Application frameworks written in JavaScript.