POSSIBILITIES FOR UNIFYING E-COMMERCE PLATFORMS

A RESEARCH AND DEVELOPMENT OF E-COMMERCE WEBSITE TEMPLATE

Graduation Thesis

by Atanas Yonkov

Charged by Fontys Hogescholen, Eindhoven **Commissioned by Merkle, Bulgaria**

ABSTRACT

This thesis addresses the challenges encountered in the development of customized e-commerce websites, with a focus on optimizing the process through the creation of a reference architecture. The study centers around Merkle Inc., a specialized customer experience Management Company renowned for its e-commerce website development.

The research objectives encompass the identification of essential e-commerce reference architecture features, an exploration of the benefits these websites offer to customers, and an analysis of Merkle's backend systems. To achieve these aims, the study adopts the ICT DOT framework, utilizing literature studies and field interviews.

During the course of the investigation, two valuable tools were discovered to enhance backend development. The first tool involves leveraging TypeScript in conjunction with Redux and React to efficiently extract data from the backend. TypeScript proves instrumental not only on the frontend but also for handling API responses, saving time through precise typification of data and facilitating seamless integration with the Redux store.

Additionally, the study identifies Nodemon as an invaluable tool for Node.js applications. Nodemon automates server restarts upon detecting modifications to the source code, eliminating the need for manual intervention. This streamlines the development workflow, enhancing efficiency and reducing downtime during the development phase.

It is essential to acknowledge the study's limited generalizability, as it specifically focuses on the challenges faced by Merkle in e-commerce website development. Nonetheless, the detailed insights into the reference architecture and associated technologies contribute valuable knowledge to this domain.

The thesis follows a structured chronological approach, with individual chapters dedicated to significant milestones. Beginning with an overview of Merkle's challenges and the objective of creating the reference architecture, subsequent chapters delve into the company's background, analyze research questions, and propose potential solutions through technology exploration. The study concludes with a proof of concept, showcasing the practical implementation of identified solutions in an e-commerce website.

This research significantly contributes to the optimization of e-commerce website development, addressing specific challenges faced by Merkle. The incorporation of TypeScript, Redux, React, and Nodemon as powerful backend development tools further enhances the productivity and efficiency of developers in this competitive e-commerce market. The findings presented herein are expected to aid Merkle's success and hold the potential to benefit other organizations navigating similar challenges in e-commerce website development.

TABLE OF CONTENTS

List of Table

List of Figures

GLOSSARY

A API

API stands for Application Programming Interface. Through this interface, computer applications can communicate with each other.

C Client

In the context of my project, I would refer to Merkle as the client of this project.

Customers

In the context of my project, I would refer to customers as the customers who shop from the retailer brands.

R Retailers

In the context of my project, I would refer to retailers as the brands that happen to be clients of Merkle.

S SPA (Single Page Application)

A website or web application that dynamically rewrites a current web page with new data from the web server, instead of the default method of a web browser loading entire new pages.

1. INTRODUCTION

Merkle Inc., a distinguished customer experience management company, plays a pivotal role in the development of customized e-commerce websites for its clientele. As the e-commerce industry continues to witness unprecedented global growth over the past two decades, online retailers have leveraged this momentum to expand their customer base and achieve heightened business outcomes and profits (Masyhuri, 2022). Nevertheless, amid these opportunities, the realm of e-commerce development faces an array of challenges that warrant meticulous examination to optimize business results.

This thesis delves into a comprehensive exploration of the challenges intrinsic to e-commerce development, with a particular focus on Merkle's unique approach. By studying Merkle's process, this research endeavors to uncover invaluable strategies and solutions that can significantly enhance the efficiency and efficacy of e-commerce website development. As the e-commerce industry continues to evolve, this study holds the potential to drive meaningful advancements and contribute to the success of businesses operating in this dynamic and competitive landscape. Firstly, the thesis aims to understand the essential features of an e-commerce reference architecture, explore the benefits of e-commerce websites for customers, and analyses the backend systems used. By connecting the reference architecture with existing technologies. Secondly, the research study will figure out the technological domain that Merkle operates with by doing literature and field studies, the research will also investigate possible solutions how to connect the established reference architecture in the context of Merkle’s already used technologies.

A major limitation of this study is its limited generalizability. The study focuses specifically on Merkle's process of ecommerce website development and its associated challenges.
Therefore, the findings and conclusions may have limited generalizability to other companies or industries, as the research scope is narrow. The research method will use the best approach from the ICT DOT framework, each milestone in the research with chapters will be separated chronologically and set the relevant sub-questions. To answer some other questions, Literature study, Interview research method from the ICT DOT (Methods - ICT Research Methods, n.d.) framework was utilized and analyzed. The findings from this thesis will contribute to the comprehensive exploration of developing and implementing a reference architecture for optimized ecommerce website development, addressing the challenges faced by Merkle and contributing to their success in the competitive market.

READING GUIDE

The thesis begins with an introduction that outlines the challenges faced by Merkle in developing customized e-commerce websites and the goal of creating a reference architecture. It then explores the importance of e-commerce websites and their benefits for businesses and customers.
In the first chapter a background for the company will be provided, in order to understand what exactly the purpose of the Merkle is.
In the second chapter where I analyzed the given assignment, I figured out a research question, what is the cause of the assignment, what needs to be done in order to help Merkle resolve the problem. I also split the assignment in relevant sub-questions by topics in chronological order - from the orientation phase of the project - where I start to investigate Merkle and how it operates.

Coming through concepting possible solutions and later the process of implementing them into a proof of concept - example e-commerce website that uses the technologies, solutions and approaches that I found during the execution of the project.
Each main chapter is identified with a big point - 1. Then the relevant sub-chapters begin with one more indentation - 1.2. Their purpose is to give an overview of the sub-question that will be answered in them. Follow-up research questions and pieces of the research will be yet indicated with one more indentation 1.2.3 and 1.2.3.4.

2. BACKGROUND

This graduation project was commissioned by Merkle - a customer experience management company with offices all around the globe. This specific assignment was commissioned by the Bulgarian branch of Merkle.

The main goal of this project was to research a solution that would help the business department have a quick to-go solution when pitching a project for a client, and on the other hand assist developers by supplying them with a starting point for a new client's project.

This chapter will go into detail who are Merkle and their sphere of influence.

2.1. About the company

Merkle is a prominent company operating in the e-commerce industry. Their focus is providing a comprehensive one-stop solution for the online commerce presence of various retailers. With a team of over 350 skilled software engineers, architects, QA engineers, and support specialists based in a technology center in Sofia, Merkle has an impressive track record of creating and managing e-commerce platforms for big international brands and retailers. They have successfully launched over 400 e-commerce websites in 60 countries across 6 continents (Merkle Inc. 2023).

Online retail has been experiencing significant growth over the years and is expected to continue expanding in the future. E-commerce has emerged as a major player in the retail industry, revolutionizing the way we think about shopping and financial transactions (George Washington University, 2020b). Its

convenience and cost-effectiveness have contributed to its paradigm-shifting impact. As part of this dynamic e-commerce landscape, companies like Merkle play a crucial role in addressing various challenges and opportunities. Some of the common issues that online retail companies face include enhancing customer experience, optimizing e-commerce platforms for better performance and loading times, ensuring robust security features to protect sensitive customer data and prevent fraud, and focusing on scalability to handle a growing number of customers and transactions (Merkle Inc 2023).

Merkle Logo, source: (dentsu. (n.d.). Dentsu. https://www.dentsu.com/us/en/who-we-are/our-agencies/merkle)

3. THE ASSIGNMENT

3.1. Main Research Question

What strategies can be employed to develop and implement an optimized reference architecture website for unifying e-commerce platforms, to streamline the development process and enhance the pitching strategy?

3.2. Problem Statement

The company faces challenges in creating customized websites for brand clients, as the current approach involves duplicating existing websites and making adjustments for each new brand. This results in the accumulation of legacy code, making maintenance and upgrading difficult, and subsequently leading to slow delivery and potential loss of new clients. Therefore, there is a need to develop and implement an optimized reference architecture website that unifies e-commerce platforms, streamlines the development process, and enhances the pitching strategy.

3.3. Project Goals

The main objective of the project is to develop and launch a functional e-commerce website using a reference architecture approach.

The company, Merkle, aims to create a streamlined and basic website that serves as a foundation for future projects with new clients, focusing on essential features rather than specific styling or product catalogues.

The goal is to enable software engineers to efficiently customize

websites according to each customer's specific requirements, saving time and effort in the development process.

By implementing this reference architecture strategy, Merkle aims to enhance their ability to attract and retain customers effectively. The purpose is to streamline the website development process and offer tailored solutions promptly, ultimately accelerating the acquisition of new retailers and expanding their customer base in less time.

The successful implementation of the reference architecture approach is expected to contribute to increased profitability for the company.

3.4. Problem Analysis

The initial phase of the research involved conducting interviews with colleagues and the company mentor to identify the root cause of the problem within Merkle, exploring the interrelation between inefficient development and client acquisitions.

The next step focused on understanding the technological domain in which Merkle operates. This involved conducting literature and field studies to gain insights into the specific technologies used by Merkle and to determine how the proposed solutions could be adapted accordingly.

The subsequent phase investigated potential solutions, specifically focusing on integrating the reference architecture within the existing technological landscape of Merkle.

This phase aimed to identify strategies for effectively connecting the reference architecture with the technologies already in use.

The fourth and final phase of the project centered on the development of the reference architecture itself. Drawing on the

findings from the previous analysis, the focus was on meeting the specific needs identified during the research.

The development process aimed to address challenges faced by developers, optimizing code writing practices to minimize the need for rewriting old functionalities, reducing time spent on fixing bugs, and improving the maintenance of existing features.

3.5. Research Framework

The Framework I will be using organizes the research chapters chronologically, based on the milestones and sub-questions identified. Adopting the ICT DOT framework, this framework provides a structured approach for problem-solving and decision-making in my research.

I then conducted a thorough analysis of the results obtained, comparing and contrasting the different solutions based on their effectiveness, feasibility, and alignment with Merkle's specific context and provided conclusions on the best answer for each sub-question.

3.6. Sub-questions

I have adapted the research questions from the project plan **(Appendix A)** based on the problem analysis from point 3.4.

What strategies can be employed to develop and implement an optimized reference architecture website for unifying e-commerce platforms, to streamline the development process and enhance the pitching strategy? During the process the most employed methodologies were: literature review study, case study, and interviews with developers.

3.6.1. Essential features of a reference architecture for e-commerce website development

To study this the following sub-questions were asked:

3.6.1.1. What do e-commerce websites do and what do they solve?

3.6.1.2. What backend systems are used?

3.6.1.3. What are the most important features that should be included to meet users' experience expectations?

3.6.2. How to connect the reference architecture with existing technologies?

This will investigate the integration of the reference architecture with existing technologies and addresses the following sub-questions:

3.6.2.1. What is headless development and how does it work?

3.6.2.2. How to implement and utilize gathered findings in the context of Merkle?

3.6.3. Designing the reference architecture and development considerations?

This will explore the design aspects of the reference architecture and addresses the following sub-questions:

3.6.3.1. How to develop features in an effective way for Merkle?

3.6.3.2. What does it mean to design code for flexibility and maintainability in the long run?

3.6.4. How can the effectiveness of the project be evaluated?

3.6.4.1. How can the project be tested and validated?

4. STARTING PHASE

In this phase of the project, I began to answer the sub-question in chronological order. I call it a conception phase because this is the chapter where I try to understand what e-commerce solves as a whole and concept possible solutions in the context of Merkle. The objective of this phase was to identify a list of features that would need to be implemented and the platform necessary for adaptation.

4.1. Initial orientation

To begin, I analyzed the task and wanted to understand the business domain of e-commerce websites, so that I could adjust my technical knowledge and further research to be as relevant as possible. In the sub-points below I answered the question: What are the essential features of a reference architecture for e-commerce website development? Splitting this question into relevant sub-question and answering them methodologically should enable me to precisely find the requirements for the next important point - 4.2

4.1.1. Overview of ecommerce platforms

What platforms do e-commerce websites use and what do they solve, and most importantly – which platform does Merkle use?

Methodology:

To answer this question a mixed approach from the dot framework that involved conducting a literature study of various e-commerce platforms such as Amazon, eBay, Alibaba, Adobe Magento etc., expert interviews and community research. The participants in this study were 16 developers from different teams within Merkle. The data analysis process involved organizing the data into themes that were relevant to the research question and examining the data for patterns and trends. The result of this study indicated that the e-commerce field operates mostly few types of platforms. Conclusively, the information obtained from the study led to the question of which of these content management systems (platform) Merkle operate and how the finding could benefit the development of reference architecture.

Results:

Baidu, Alibaba, and Tencent (BAT) represent China's digital economy and are major international competitors to US-based digital technology giants. This literature review examines the business strategies and offerings of BAT and compares them with other e-commerce platforms, including Merkle. The BAT companies compete for domestic and international market share, user base, and possession of underlying data, profitability, and technological innovation. While they possess both similarities and differences, they represent a significant challenge to the global economic order and the power of nation-states (Su & Flew, 2021).

Rahman's (2021) paper identifies Alibaba as the world's fastest-growing e-commerce marketplace, providing opportunities for small and medium-sized enterprises to do business in local and global markets through its various platforms. This finding highlights the importance of Alibaba's business model and frameworks in the e-commerce industry. These features are compared to the offerings of Merkle's platform and it shows that both Alibaba and Merkle offer valuable opportunities for businesses in the e-commerce industry. Alibaba's widespread reach makes it an attractive choice for companies looking to tap into global markets, while Merkle's expertise in creating tailored e-commerce platforms makes it a compelling option for businesses seeking optimization and enhanced customer experiences.

Al Moaiad's (2023) research study compares eBay and Amazon's business models, marketing strategies, and revenue models. This study concludes that eBay has empowered supply and demand on the web with great outcomes, while Amazon has introduced several features to facilitate easy purchasing by customers. Amazon's features, including Amazon Web Services and Amazon Mechanical Turk, are also discussed.

These features are compared to the offerings of Merkle's platform. Merkle's platform helps businesses gain a single view of their customers and deliver the right experiences through media, CRM, and loyalty programs. The platform's benefits include a reduction in acquisition costs, more efficient spending, and the reward of long-term customers.

The platform is compared to the other e-commerce platforms discussed in the literature review. Targeted interviews within the company reveal that Merkle operates by implementing CXM with

the help of SalesForce Commerce Cloud. This implementation is discussed in comparison to the other platforms discussed in the literature review. **(Appendix B & Figure 1)**.

Conclusion:

This literature review aimed to answer the research question of which e-commerce platforms websites use and what problems they solve. The review discussed the business strategies and offerings of the BAT companies, Alibaba, eBay, and Amazon, and compared them to the offerings of Merkle's platform, the study proposes an efficient Development Process: By adopting proven strategies and frameworks from the literature study and case studies, the development process will be streamlined and more efficient.

In conclusion, this literature review highlights the importance of choosing the right e-commerce platform based on specific needs and the benefits it can bring in terms of business growth and customer satisfaction. The findings of the studies can inform the development of Merkle's platform and help it compete with other e-commerce platforms through its unique value proposition and key differentiators such as the Optimization for Performance and Loading Time. Further research could explore the effectiveness of CXM and Sales Force Commerce Cloud in different industries and business contexts.

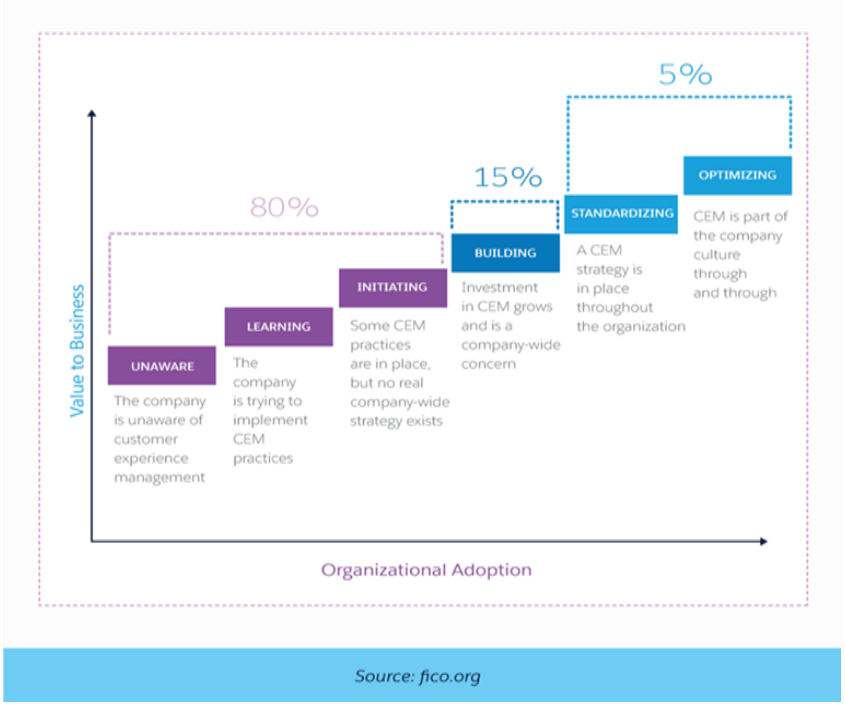


Figure 1 *Line Graph displaying the value of customer experience management for a business | Source: https://www.kyinbridges.com/the-top-companies-in-customer-experience-management/*

4.1.2. Identification of key features and functionalities

Cause of the question:

The study has identified the specific e-commerce domain and software that Merkle operates in. this research sub-question aimed to understand what makes e-commerce valuable to customers, by finding out which features end-users find useful. This information can inform the development of e-commerce websites with the clients of Merkle, who are retailers that have e-commerce websites in mind. Incorporating useful features can enhance the value of e-commerce websites to Merkle's clients, ultimately benefiting the company.

Question:

What are the most important features that should be included to meet users' experience expectations?

Methodology:

These research question findings would help to better understand the technical context of the assignment and set clear guidelines for features of the MoSCoW in order to implement later.

The research design involved a mixed-methods approach to identify valuable features in e-commerce websites. The study used case studies of well-known successful e-commerce brands including eBay, Alibaba and Amazon to identify features that overlap and are useful. The study also employed a literature research on general good features for e-commerce to validate the results. The data collection methods was from

case studies and literature review, I then analyzed the data collected from survey **(Appendix E)** to determine which features overlap and are useful. The survey was sent in the most popular channel in our ‘Slack’ and was targeted to the technical people in the company – developers, quality assurance specialists and dev leads. The themes discussed were about general opinions of developers on code maintenance in the long run and project specific questions related to which are the bottlenecks for the developers. The answers received counted 16 by the end of the survey, which lasted 5 working days.

Results:

The study found that end-users highly value e-commerce features that are user-friendly, efficient, and personalized. Specifically, participants rated features such as easy navigation, clear product descriptions, and personalized recommendations as highly useful. In addition, participants expressed a preference for e-commerce websites that offer multiple payment options, fast and reliable shipping, and easy returns.

Incorporating these features into e-commerce websites can enhance their usefulness and increase their value to Merkle's clients. (White, 2021 and Cassidy, 2020).

The study compared Merkle's platform with the e-commerce platforms of Alibaba, Amazon, and eBay. Alibaba's success can be attributed to its numerous platforms that provide opportunities for small and medium-sized enterprises to do business in local and global markets (Yogish & Nandha, 2021). Amazon's focus on customer satisfaction metrics and personalization can inform Merkle's approach to enhancing the user

experience (Round, 2004). eBay's robust feedback and rating system, easy navigation, high-quality product images and videos, and user-generated reviews and ratings can inspire Merkle's efforts to increase the value of its platform to clients (eBay.com USA 2023).

Conclusion:

The research question aimed to identify the most important features that should be included to meet users' experience expectations in e-commerce websites. The study's findings then suggest that to create a positive user experience, e-commerce websites within Merkle should prioritize mobile-friendliness, easy navigation, high-quality product images and videos, user-generated reviews and ratings, fast loading times, secure checkout processes, and personalization based on customer preferences and behaviors. These features enhance customer satisfaction and loyalty, leading to higher returns on investment for businesses.

To guide the development phase, the study enabled the creation of a MoSCoW table that prioritizes the identified functionalities. The MoSCoW table provides a clear guide for the development team, and it was developed by incorporating the most important features and identified functionalities into the platform first, leading to an enhanced user experience and higher value for Merkle's clients **(Appendix B)**.

Table 1: MoSCoW Table.

Must-Have	Should-Have	Could-Have	Won't-Have
User Registration and Authentication: Allow users to create accounts, log in, and securely authenticate their identities.	Responsive Design: Ensure the website is optimized for various devices and screen sizes to provide a seamless user experience.	Social Media Integration: Allow users to share products on social media platforms, increasing brand exposure and potential sales.	Multilingual Support: Exclude multilingual support in the initial development phase, focusing on a single language.
Product Catalog Management: Enable administrators to add, edit, and delete products, including details like images, descriptions, and prices.	Product Filtering and Sorting: Allow users to filter products based on attributes (e.g., price range, brand, color) and sort them by relevance, price, popularity, etc.	Personalized Recommendations: Implement recommendation algorithms to suggest related products based on user preferences and browsing history.	Advanced Analytics and Reporting: Omit comprehensive analytics and reporting features, such as sales reports or customer behavior analysis.
Shopping Cart Functionality: Allow users to add products to their carts, review and modify cart contents, and proceed to checkout.	Wishlist: Enable users to create and manage wish lists, save products for future purchases, and share them with others.	Multiple Payment Options: Offer various payment methods, such as credit cards, digital wallets, and alternative payment gateways, to accommodate different user preferences.	Social Login: Exclude the ability for users to log in or register using their social media accounts.
Payment Gateway Integration: Integrate with a secure payment gateway to process online payments from customers.	Product Reviews and Ratings: Allow users to leave reviews and ratings for products, enhancing credibility and helping others make informed decisions.	Order Tracking for Customers: Provide a tracking feature for customers to monitor the status and location of their orders.	

4.2. Connecting reference architecture with existing technologies

At this point, the main goal was to gather more information about the Salesforce Commerce Cloud (SFCC) field to ensure that the solution meets the requirements to run on the SFCC platform. The research aimed to identify the best techstack to be adapted for the SFCC domain, in order to optimize the product's performance and functionality. By understanding the SFCC platform's code wise operations and the most suitable techstack. **(Appendix C)**

4.2.1. How to connect the reference architecture with existing technologies?

Question:

As mentioned before, I wanted to get more information about the way SFCC works. That is the reasoning behind this exact question, what backend does it use and how does it work?

Methodology:

The research design involved a literature study and access to a repository for a project in a company with the most developers working on the SFCC platform. The data collection methods and analysis included a literature study of official SFCC documentation to find relevant information related to the techstack and from the repository access.

Results:

The results I got were happily straightforward, but more complicated than I imagined. It seemed that the salesforce

commerce cloud used a very specific structure for the codebase of a project (Tryzens, 2021). SFCC is run by utilizing different cartridges for different parts of the code. For simplicity, you can imagine these cartridges as different components in an automobile. For example, one cartridge with a few hundred file of code was dedicated solely for third party integration of a payment system. This was then connected with another cartridge which was only dedicated for the front-end functionalities of the code (Cloudinary, 2023). Like a vehicle, in order for just the engine to start, we need a motor, a starter, fuel pump, etc. They were interlinked with each other. The thing I noticed were the tons of legacy code, making it ever so difficult to understand what the flow of commands was. For the backend, salesforce provides separate functions that could be adapted for each component in an e-commerce website - basket page, product details page and so on. The functions were in of course another cartridge and utilized controller-based endpoints. Endpoints are used on the front-end to retrieve data from the backend (Vsupalov, 2022).

Another finding here was coming from one of the sales force's official blogs. It seems that in the past there were various approaches to build an application on Salesforce. What I found was that they are looking forward to implementing the headless development approach, this is something new - what is headless development is a question that emerged, which will require further study in the next phase.

Conclusion:

In conclusion, the research question aimed to explore how to connect the reference architecture used by SFCC with existing technologies. To answer this question, a literature study was conducted on SFCC, focusing on the official documentation to gain insights into the backend technology used by the platform.

Additionally, access to a repository for a project in a company with the most developers working on SFCC was requested to gain practical insights into the backend technology.

The research findings suggest that SFCC projects work via a complex set of cartridges and controller-based endpoints, which for Merkle were written in legacy code, making it difficult to maintain. This should be taken into account when thinking of a solution on the backend, and it is important to consider the best backend framework to use to optimize performance and maintainability.

4.2.2. What is headless development?

Cause of the question:

Having gained an understanding of SFCC and its functionality, I have developed new questions based on previous conclusions. This research question aims to comprehend the concept of headless development. That would help me to have a clear guide when choosing a framework for the front-end and back-end. That is the cause behind this question - understand headless and after that find the best framework.

Methodology:

The research methodology involves conducting a literature study to investigate the concepts of headless development, the problems it solves, and its implications for building the reference architecture, also Data is extracted from the selected sources and synthesized to identify common themes and patterns related to headless development and data is analyzed and interpreted to draw conclusions and make recommendations regarding the use of headless development when building the reference architecture.

Results:

In the context of Salesforce Commerce Cloud, headless development can be highly beneficial for businesses that want to provide engaging and personalized experiences to their customers across multiple touch points (Medium, 2021). By separating the front-end logic from the back-end data management, developers can focus on building custom front-end experiences using any technology they choose, without impacting the back-end system. This allows for quicker iteration and adaptation to changing customer needs and market trends.

One of the key benefits of headless development in Commerce Cloud is the ability to provide consistent experiences across different channels (Jain, 2023). By creating reusable components, businesses can ensure that customers have a seamless experience regardless of the channel they use to interact with the business. This not only saves development time, but also helps to improve customer satisfaction and loyalty.

Another important benefit of headless development in Commerce Cloud is the ability to leverage third-party services and integrations. (Medium, 2023). With APIs, businesses can easily integrate with other systems and services, such as payment gateways, social media platforms, and marketing automation tools. This allows for a more seamless experience for customers and also provides valuable insights into customer behavior and preferences.

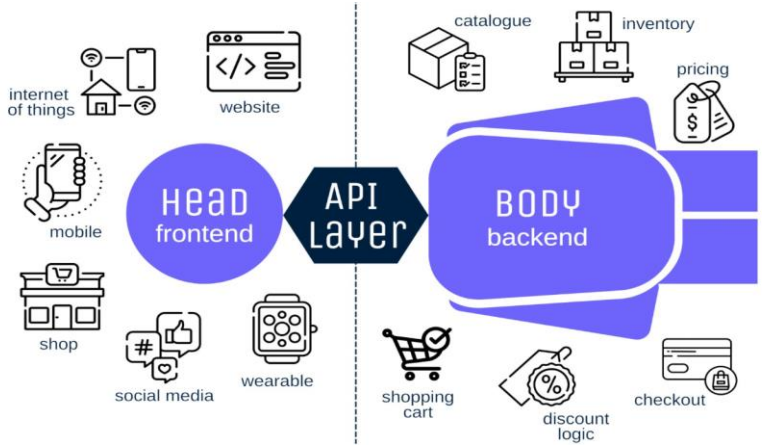


Figure 2. Visual separation between head and body in software development

Conclusion:

The research question aimed to understand the benefits of headless development in the context of Salesforce Commerce Cloud and find the best framework to achieve it. By leveraging the benefits of APIs and separating the front-end logic from the back-end data management, headless development can be highly useful for Merkle (Hossain, 2022). Next steps will include further investigation of which framework to use and why.

4.2.3. How to implement and utilize gathered findings in the context of Merkle?

Cause of the question:

The reason behind this question is to help find specific frameworks that make best use of OOP principles. What is the backend for them that also takes SFCC into account?

Methodology:

The research methodology for this study involved a literature review and consultation with developers who were participants in the study. Based on the previous research, it was determined that the most logical next step was to identify the frameworks on the front-end and back-end that would seamlessly integrate with the

headless development approach in the context of Salesforce Commerce Cloud.

To achieve this, three popular front-end frameworks, React, Angular, and Vue, were investigated as they are the latest trend in the market. The investigation involved comparing each framework to determine what problems they solve best and how they can be integrated with the headless development approach.

As this project assignment was mostly front-end oriented, the focus was primarily on the front-end framework. However, it was also important to ensure that the chosen front-end framework would work seamlessly with the back-end solution in SFCC.

Results:

React, Angular, and Vue are popular JavaScript frameworks/libraries commonly used for building user interfaces (UIs) in web applications. (Staff, CACM. 2016). React is a JavaScript library developed by Facebook. Developers can create reusable UI components that update effectively and adapt to data changes. React follows a component-based architecture, where the UI is broken down into reusable and independent components (Muldoon, 2023).

Angular is a comprehensive JavaScript framework developed by Google. It provides a complete solution for building web applications, including components, data binding, dependency injection, routing, and more. It employs a two-way data binding approach, where changes in the UI and data model are automatically synchronized (Uzayr et al., 2019).

Vue is a progressive JavaScript framework that aims to be approachable and versatile. It allows developers to build interactive UIs by composing reusable components. Vue takes a middle ground between React and Angular, offering simplicity and ease of use (Filipova, 2016).

It is known for its gentle learning curve, flexibility, and smooth integration with existing projects.

Finally, in order to summarize the results from these findings I've listed four important factors when choosing the adequate front-end framework:

- **Learning Curve**: React and Vue have a relatively easier learning curve compared to Angular, which has a steeper learning curve due to its complex concepts like dependency injection.
- **Size and Performance**: React and Vue have smaller bundle sizes and are generally considered lightweight. Angular has a larger bundle size due to its extensive features but provides excellent performance optimizations.
- **Ecosystem and Community**: React has a large and active community with a vast ecosystem of libraries and tools. Vue has a growing community and ecosystem, while Angular has a mature ecosystem backed by Google.
- **Integration and Flexibility**: Vue is highly flexible and can be easily integrated into existing projects. React is also flexible but requires making more decisions about libraries and tools.

Today's standards of using those architectures are by utilizing Node.js. For web-applications this is currently the most popular and best solution in terms of scalability, maintainability and complexity.

Conclusion:

Based on the findings research question, how can the findings from the investigation of React, Angular, and Vue be implemented and utilized in the context of Merkle? Ultimately, the choice between React, Angular, and Vue depends on the project requirements, team expertise, and personal preferences. Each framework has its strengths and is widely adopted in the

industry, so choosing the right one involves considering factors like project complexity, scalability, performance needs, and developer familiarity. Each of these frameworks could be used for this project, but could every other project in the company be adapted for each of these frameworks? Does each developer have time to study a brand-new framework?

Overall, the findings suggest that a careful consideration of all factors involved is necessary to implement and utilize the findings of the investigation of React, Angular, and Vue in the context of Merkle.

5. DEVELOPMENT PHASE

After answering all the questions in chapter 4, I gained a sufficient understanding of the concepts needed to build the project theoretically. This phase can be referred to as the conception phase, where I methodically gathered the required information, even if it meant starting with an understanding of the e-commerce business domain. This chapter will introduce the development process of the project - the coding of the web-application. **(Appendix D)**

5.1. DEVELOPING FEATURES FOR MERKLE

Chapter 4 provided me with valuable insights, including the fact that the company uses a CXM system through the SalesForce Commerce Cloud. This helped me understand the essential features needed for the project, which I prioritized using the MoSCoW method from table 1. I then delved deeper into SalesForce Commerce Cloud to understand its code base and learned that it will soon adopt the headless approach. This information helped me create clear criteria for the software architecture. However, since there were multiple valid frameworks to choose from, I needed to answer several sub-questions to understand the general technology stacks used in Merkle and best coding practices to keep in mind. Combining the outputs from Chapters 4 and 5, I was able to create a feature-proof web application that benefits Merkle.

5.1.1. How to develop features in an effective way for Merkle?

Cause of the question:

The cause of this question comes from the conclusion from point 4.2.3. There I found that I couldn't decide on a front-end framework just by doing general literature research. The results should be adapted in order to help developers at Merkle.

Methodology:

As part of the research methodology, to identify the most used front-end and back-end technologies among developers and the daily challenges they face, a survey was conducted and followed up with an interview **(Appendix E)**. The information gathered will be used to improve the chosen framework and address developers' daily blockers, resulting in a feature-rich web application that delivers value to Merkle.

Results:

As part of the research methodology, to identify the most commonly used front-end and back-end technologies among developers and the daily challenges they face, a survey was conducted and followed up with interviews. The information gathered will be used to improve the chosen framework and address developers' daily blockers, resulting in a feature-rich web application that delivers value to Merkle.

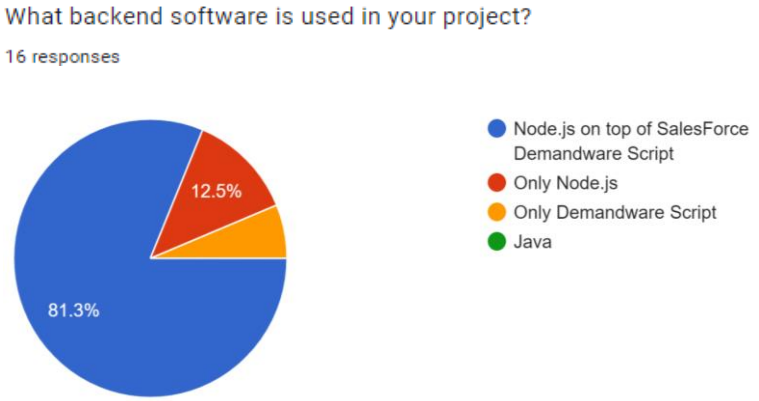


Figure 3: Survey responses – frameworks used on the back-end.

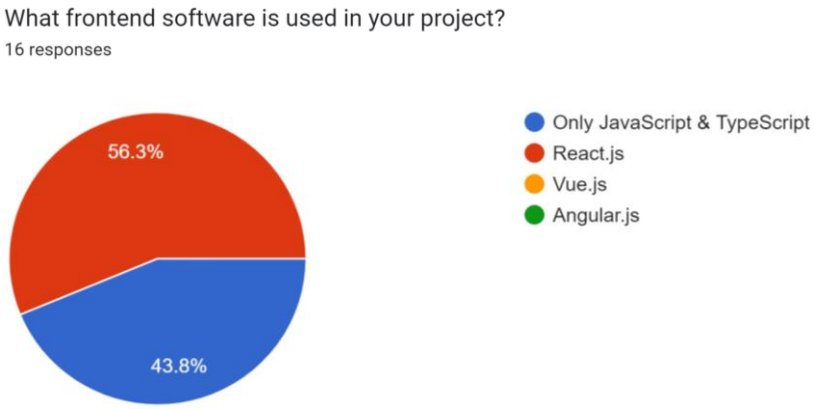


Figure 4: Survey responses – frameworks used on the front-end.

The most common issues encountered in maintaining e-commerce websites were slow page load times, difficulty making changes to the site, bugs and errors, and poor performance on mobile devices. Strategies found effective for maintaining code include regular code reviews, consistent documentation, frequent refactoring, and regular performance monitoring. Respondents reported that code is refactored on an irregular basis, with some doing it every quarter or every two sprints, while others do it every year or when a problem on production occurs.

Conclusion:

To effectively develop features for Merkle in an effective way, it is important to note that Developers in Merkle would mostly write code in React which now leads me to choose my framework of choice to be React. In the context of Merkle that would be the best fit as it would require less to be learned and adapted, since most of the developers already have the knowledge. Further I now know to look for tools on top of react that solve - minimize bugs, or at least simplify the debugging process and simplify the file structure. For the backend it was almost certain in advance, but here I confirmed that Node.js would be most suitable. I started writing the project as a SPA with React.js and Node.js with the architecture suited for SFCC - controller based endpoints and clear separation between the backend and the frontend.

5.1.2. What does it mean to design code for scalability and maintainability in the long run?

At this point, the research will discuss, analyze, and determine the best tools and technologies that could be implemented. Based on all the results obtained thus far, the focus will be on identifying new technologies and tools that can assist developers in their work.

5.1.2.1. Future-proofing the code on the front-end

Cause of the question:

After deciding to use React as the framework for development, the next step was to consider ways to future-proof the code. This involved identifying the best tools and technologies that would enable developers at Merkle to take advantage of a headless approach in the future. To achieve this, I researched the latest technologies and tools that could be used with React in the web development world. I also considered the limitations of React as a tool for complex e-commerce websites and looked for ways to address these limitations.

Methodology:

The methodological approach taken in this study was a literature review, focused on identifying solutions that could be implemented on top of React.

Results:

Through the literature review, two potential solutions were identified: Next.js and React Redux. Next.js is a framework for

building server-side rendered (SSR) and statically generated (SSG) React applications, providing features such as server-side rendering, automatic code splitting, and simplified routing. This makes it easy to build fast and scalable web applications.

Additionally, Next.js is built on top of React and adds server-side rendering capabilities, resulting in pre-rendered pages on the server that are delivered to the client as HTML (Thakkar, 2020). This improves initial page load times and search engine optimization (SEO).

React Redux, on the other hand, is a state management library for React applications. It provides a predictable state container that helps manage the application's state in a centralized manner.

React Redux follows the principles of Flux architecture and is commonly used with React to manage the state of large-scale applications (Teimur, 2022). This allows for better organization,

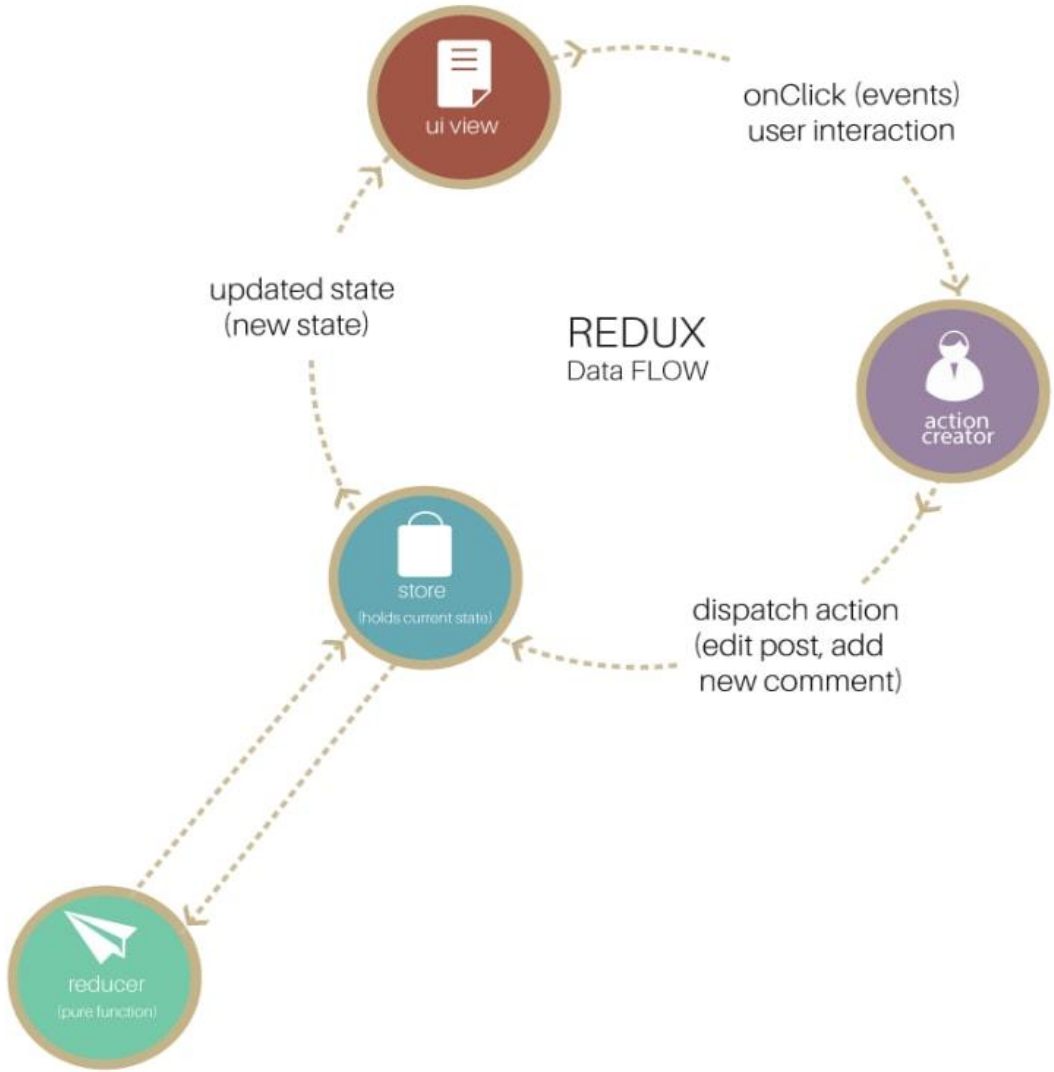


Figure 5: Data flow of states with Redux
Source: *https://dev.to/dhintz89/what-is-redux-anyway-part-2-1gf4*

separation of concerns, and data flow control in React applications.

Conclusion:

While Next.js and React Redux serve different purposes, they can be used together to build powerful web applications.

Today's standards prove that both could be implemented, but the time limitations of the project wouldn't allow me. I chose to implement Redux on top of React as its state management is far more important for the project, based on the previous results of the survey and expert interview. Devs have trouble writing new functionalities and maintaining code - Redux will help with that by a big margin, having a general 'store' for all the states in the Project and that could be accessed anywhere. (Hintz, 2020)

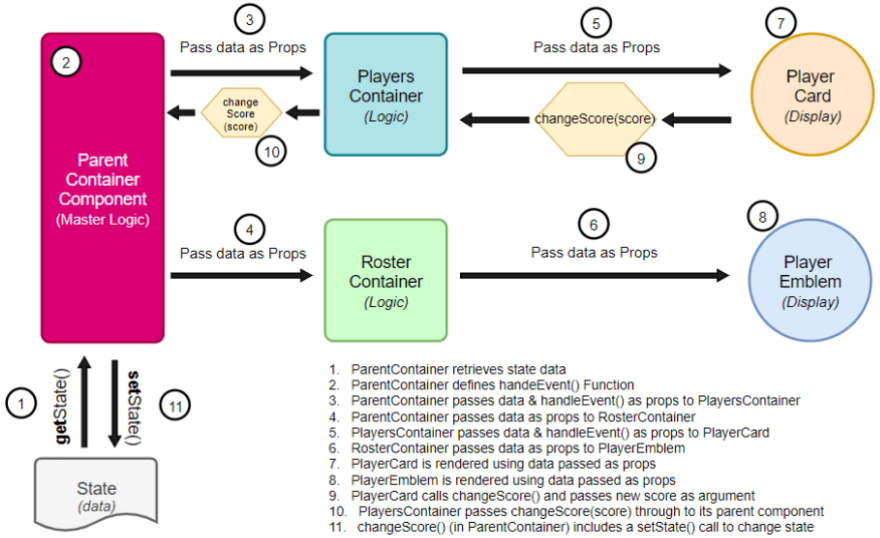


Figure 6. Data flow of states with Redux

Source: https://dev.to/dhintz89/what-is-redux-anyway-part-2-1gf4

5.1.2.2. Help developers be efficient on the back end

Question:

What tools or strategies can be implemented to improve the efficiency and productivity of developers on the backend?

Methodology:

The methodological approach taken in this study was to identify tools and strategies that could improve the efficiency and productivity of developers on the backend. The study was conducted as a case study, using my personal work routine as an example.

Results:

Through the case study, two tools were identified that could improve the backend development process. Firstly, it was found that using TypeScript with Redux and React could provide serious leverage when extracting data from the backend. . However, TS could be used also on the backend as well for the API part of the project, where I extract the response from the provided API and adapt it so it could be used on the front. That is why I decided to strictly use .ts and .tsx files in the whole project. Five minutes of typifying the response from an API request, could save tons of time navigating the project and populating the Redux store with that information.

While working on that part I noticed that backend was a hassle to work with despite being just a simple project so far. My experience was mostly saving and manually refreshing the front so that I could manually refresh the backend by starting in separately. Secondly, the study identified Nodemon as a tool for Node.js applications that automatically restarts the server whenever changes are made to the source code. It monitors the

files in the application directory and, upon detecting any modifications, it gracefully restarts the server. Nodemon saves developers from manually stopping and starting the server every time, a change is made, thus providing an efficient and streamlined development workflow.

Nodemon is particularly useful during the development phase as it helps in rapidly iterating and testing code changes without the need for manual server restarts. It eliminates the need to manually track file changes and relaunch the application, saving developers time and effort.

Conclusion:

Nodemon increases the complexity a little bit but that is only while it is being set up. After that it is just extra time that could be devoted to fixing the bugs and implementing features rather than waiting for the app to restart.

Nodemon, with further implementation of TypeScript on the backend proved to be an excellent solution. My company mentor also liked it a lot and wondered why no one came up with that before.

That’s why I refactored the code from initial JavaScript for the frontend & backend to only TypeScript and gave types and interfaces to all modules. Furthermore, I set up Nodemon for hot refreshes on the backend

5.2. IMPLEMENTATION

To implement the proposed solution for Merkle, I adopted a Model-View-Controller (MVC) architecture, (MVC) pattern plays a crucial role in this transformation from a monolith architecture to a headless architecture, which promotes the separation of concerns and divides the application into three components: Model (data layer), View (user interface), and Controller (logic layer). This separation enabled easier maintainability, modularity, and extensibility. I implemented the reference architecture website using this architecture (Sitecore, n.d.).

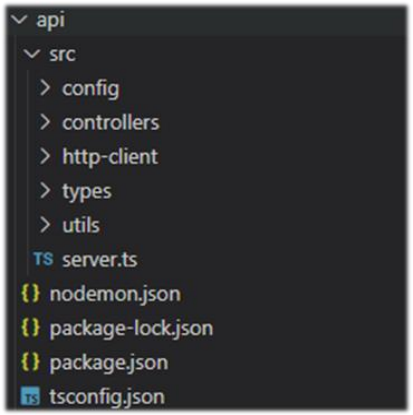


Figure 7. The API is the folder dedicated for the backend (business logic),

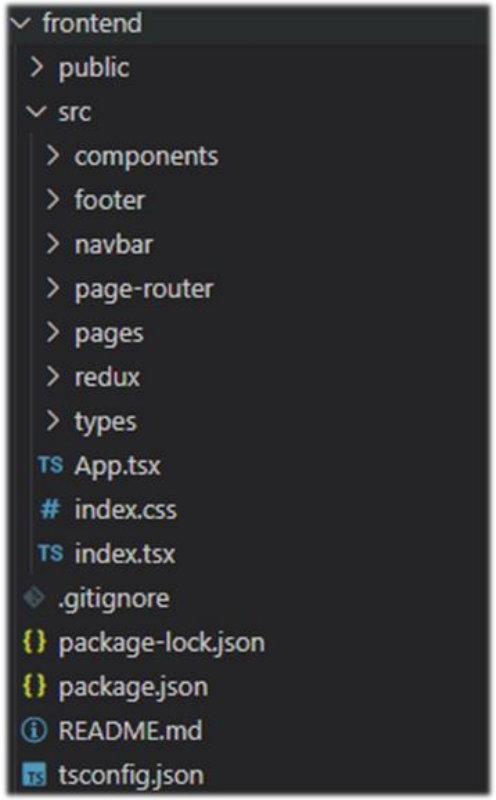


Figure 8. Representation of MVC in the folder structure of the repository the front-end for the react app components.

On the backend, I found that using Typescript with Redux and React provided serious leverage when extracting data from the backend. I also identified Nodemon as a tool for Node.js applications that automatically restarts the server whenever changes are made to the source code. This tool saves developers from manually stopping and starting the server every time, a change is made, thus providing an efficient and streamlined development workflow. (Cronj. 2023)

```
1  import { useSelector } from 'react-redux'
2  import { Link } from 'react-router-dom'
3  import { bagSelector } from '../redux/helper-functions'
4  import NavbarLink from './navbar-link'
5  import './navbar.css'
6
7  const Navbar = () => {
8    const bag = useSelector(bagSelector)
9    const { productsInBag } = bag
10
11   return (
12     <div className='navbar'>
13       <div className='navbar-top'>
14         <div>
15           <p>FREE RETURNS & EXCHANGE   </p>
16           <img
17             src='/downward-arrow.png'
18             alt='arrow'
19             className='arrow'
20           />
21         </div>
22   ...
```

Figure 9. Code view of the navigation bar with React.

On the frontend, transitioning from having no frontend framework to adopting React with Redux involved a fundamental shift in how the frontend code was structured and managed. This made it easy to build fast and scalable web applications. React Redux, on the other hand, is a state management library for React applications.

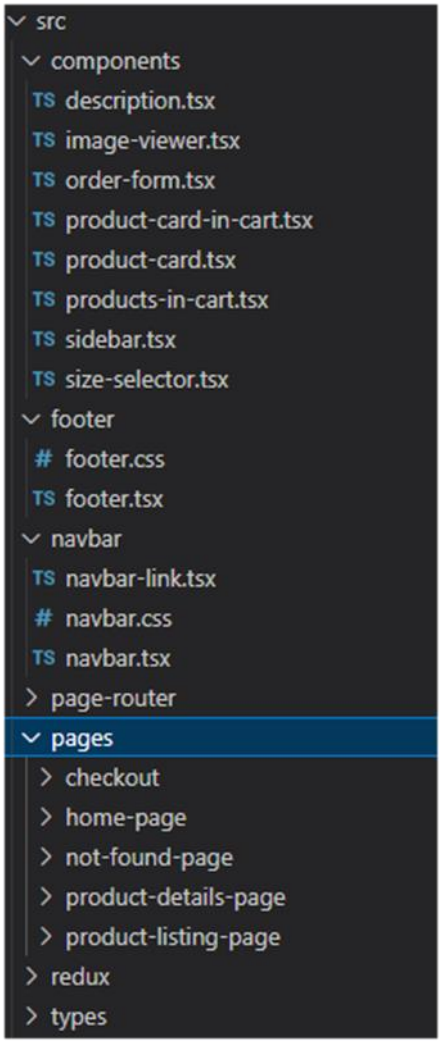


Figure 10. Front end folder.

It provides a predictable state container that helps manage the application's state in a centralized manner. React Redux follows the principles of Flux architecture and is commonly used with React to manage the state of large-scale applications. Redux enforces a unidirectional data flow, which means that data flows in a single direction within the application. Actions are dispatched to update the state in the Redux store, and components subscribe to changes in the store to reflect those changes in the user interface. (GeeksforGeeks, 2023). Reducers in Redux are pure functions responsible for handling state changes based on dispatched actions.

```
export const productDetailsReducer: Reducer<
  initialStateType['productDetails'],
  Action<void>
> = (state = initialState.productDetails, action: any) => {
  switch (action.type) {
    case PRODUCT_DETAILS_REQUEST:
      return { ...initialState.productDetails, loading: true, error: false }
    case PRODUCT_DETAILS_SUCCESS:
      return {
        ...initialState.productDetails,
        loading: false,
        error: false,
        product: action.payload,
      }
    case PRODUCT_SIZE_SELECT:
      return {
        ...initialState.productDetails,
        selectedQuantity: 0,
        availableQuantity: action.payload.availableQuantity,
        selectedSize: action.payload.selectedSize,
        loading: action.payload.loading,
        error: action.payload.error,
        product: action.payload.product,
      }
    case PRODUCT_QUANTITY_SELECT:
      return {
        ...initialState.productDetails,
        selectedQuantity: action.payload.selectedQuantity,
        availableQuantity: action.payload.availableQuantity,
        selectedSize: action.payload.selectedSize,
        loading: action.payload.loading,
        error: action.payload.error,
        product: action.payload.product,
      }
    case PRODUCT_DETAILS_FAIL:
      return {
        ...initialState.productDetails,
        loading: false,
        error: true,
        product: {},
      }
    default:
      return state
  }
}
```

Figure 11. Redux Reducer designated to retrieve content based on changes on the front-end.

Strategies that were found effective for maintaining code included regular code reviews, consistent documentation, frequent refactoring, and regular performance monitoring. Respondents reported that code was refactored on an irregular basis, with some doing it every quarter or every two sprints, while others did it every year or when a problem on production occurred. The frontend implemented using cutting edge web technologies such as React with Redux receives the response from the backend. The data within the response is parsed and processed on the frontend to extract the product information. Using the extracted product information, the frontend dynamically generates the necessary HTML elements and CSS styles to display the products on the webpage. This process ensures a responsive and user-friendly interface.

```
export type pdpStateType = {
  loading: boolean
  error: boolean
  product: {
    productId: string
    price: number
    color: string
    title: string
    text: string
    subtitle: string
    subTitle: string
    availability_status: string
    images: {
      src: string
    }[]
    variation_list: {
    }[]
  }
  selectedSize: string
      size: string
      availability_status: string
      availability: number
  availableQuantity: number
  selectedQuantity: number
}
```

Figure 12. Typification of the response returned from the API for the Product Description Page (PDP).

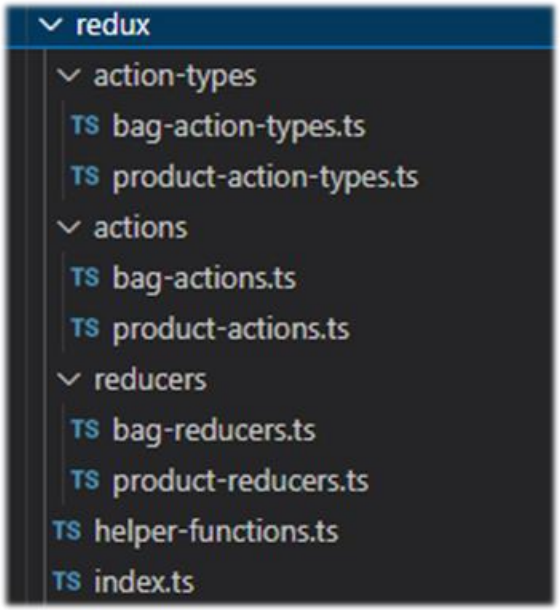


Figure 13. Folder structure of redux components.

5.2.1. Agile Approach for Merkle's E-commerce

To validate the practical benefits of the proposed solution for the proposed reference architecture for Merkle's e-commerce website development process, I implemented it in a small team that comprised the sales members within the company and tested it with the developers using guerrilla testing. The developer found the proposed implementation to be useful, and his feedback helped us refine the solution. The team then pitched the implementation to a client, who was pleased with the proposal and accepted the implementation of our proof of concept for their Asian market. The implementation involved several technical aspects, such as modifying the existing website code to allow for dynamic content loading, creating a new database to store the content, and developing a custom CMS to manage the contents.

To document the implementation, I created a user guide as a README.md file on github.com that provides a step-by-step guide on how to use the implemented solution, and a manual that includes screenshots and descriptions of the various features and functionalities of the solution was made available.

To deploy the solution from the study, I identified the infrastructure required, including the software, and networking components of Merkle. Together with other developers, we developed a detailed plan for deploying the solution, including the sequence of steps required to deploy the solution, any dependencies or prerequisites, integration with Jenkins CI/CD, and hosting.

To ensure that the proposed architecture remains functional and up to date, I plan to update it regularly to address any bugs, security vulnerabilities, or performance issues that may arise. I also plan to implement a monitoring system that regularly checks the health of the solution and alerts the team in case of any issues or errors. I will regularly back up the architecture to ensure that data is not lost in case of any

issues or errors. Additionally, I will provide technical support to users to help them troubleshoot any issues that may arise and ensure that the solution is functioning as intended.

Incorporating an Agile development methodology into the implementation process will further enhance the success of Merkle's e-commerce website project. Agile emphasizes collaboration, customer feedback, and continuous improvement. By adopting agile principles, we can foster a more flexible and iterative approach to software development, allowing us to adapt to changing requirements and deliver incremental updates. This approach will enable us to work closely with the client and stakeholders, gathering feedback early and ensuring that the final product aligns with their expectations. With frequent updates and regular communication, Agile will help us maintain a functional and customer-centric solution, providing a seamless user experience in the Asian market and beyond

6. CONCLUSION

The research question "What strategies can be employed to develop and implement an optimized reference architecture website for unifying e-commerce platforms, to streamline the development process and enhance the pitching strategy?" led to several key findings regarding the development of an e-commerce website for Merkle. Firstly, prioritizing easy navigation, high-quality product visuals, user-generated reviews, ratings, and personalized experiences based on customer preferences and behaviors greatly enhance the overall user experience. To implement these functionalities, a MoSCoW table was created to

prioritize and guide the development phase. Another significant finding was the complexity of SFCC projects, particularly the legacy code used in Merkle's cartridges and controller-based endpoints. To address this challenge, a suitable backend framework had to be identified, and the emerging trend of headless development was leveraged to design and adapt the reference architecture. **(Appendix F)**

Exploring headless development further revealed its alignment with Object-Oriented Programming (OOP) principles. This realization opened possibilities for utilizing front-end frameworks such as React, Angular, and Vue, which are known for their OOP concepts, alongside Node.js for the backend of Merkle. While this served as a general orientation, the next steps involved conducting qualitative research to determine the most suitable framework based on project requirements, team expertise, and personal preferences.

Taking into account the preferences and expertise of the developers at Merkle, React emerged as the most fitting choice for front-end and Node.js for the backend development. This decision was reinforced by the desire to minimize the learning curve and leverage the existing knowledge within the development team. Additionally, I recognized the importance of finding tools and libraries that simplify debugging processes and file structure organization, complementing the React ecosystem.

Demo

youtu.be/0TdmLu7DYlI

GitHub Repository

github.com/ayonkov1/graduation-project-reference-architecture

Since I was not allowed to export the internal APIs provided for fetching data, I mocked the requests. This question could be a point but relatively short as it is - How could I mock requests? Did a literature study and found mocky.io was the best tool. Anyway, I had to mock the requests which meant I had to manually select the data, in order not to expose any sensitive information of the client. For 48 pages of products, it was difficult to mock 2304 products, so I mocked only the first one. That is because all the scripts within the project are written to be reusable, so it doesn't matter if the products are many or one - the code will work the same. In the YouTube video I have shown what the website looked like with many products to give you a better perception.

7. RECOMMENDATION ABOUT INTEGRATION

Based on the conclusions drawn from this project, I would recommend implementing the proposed improvements to the projects at Merkle.

By prioritizing user experience and incorporating the identified functionalities, such as easy navigation, high-quality visuals, user-generated reviews, ratings, and personalization, Merkle can enhance customer satisfaction and loyalty, thus leading to higher client retention and acquisition.

Furthermore, adopting a headless development approach, utilizing React as the preferred front-end framework, and leveraging Node.js for the backend will contribute to improved development efficiency and maintainability.

Implementing Redux for state management and incorporating Nodemon for hot refreshes and TypeScript for codebase enhancement will further streamline the development process and reduce potential errors. By embracing these recommendations and taking into account the findings of this thesis, Merkle can create a robust and user-friendly e-commerce platform, empowering both developers and businesspeople.

REFERENCES

1. Masyhuri, M. (2022). Key Drivers of Customer Satisfaction on the E-Commerce Business. East Asian Journal of Multidisciplinary Research, 1(4), 657–670. https://doi.org/10.55927/eajmr.v1i4.405
2. Muñoz, M. (2019, July 26). 5 Challenges in ecommerce Development | Sales Layer. Sales Layer. https://blog.saleslayer.com/5-challenges-in-e-commerce-development
3. What We Do | Merkle. (n.d.). Merkle. https://www.merkle.com/what-we-do#:~:text=We%20help%20brands%20build%20dreams,call%20this%20customer%20experience%20transformation
4. Methods - ICT research methods. (n.d.). https://ictresearchmethods.nl/Methods
5. Valle, P. J., Garcés, L., Volpato, T., Martínez-Fernández, S., & Nakagawa, E. Y. (2021). Towards suitable description of reference architectures. PeerJ Computer Science, 7, e392. https://doi.org/10.7717/peerj-cs.392
6. Pääkkönen, P., & Pakkala, D. (2015). Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems. Big Data Research, 2(4), 166–186. https://doi.org/10.1016/j.bdr.2015.01.001
7. Van Geest, M., Tekinerdogan, B., & Rodriguez, D. (2021). Design of a reference architecture for developing smart warehouses in industry 4.0. Computers in Industry, 124, 103343. https://doi.org/10.1016/j.compind.2020.103343
8. Sulova, S. (2019). A SYSTEM FOR E-COMMERCE WEBSITE EVALUATION. In International Multidisciplinary Scientific GeoConference SGEM .. https://doi.org/10.5593/sgem2019/2.1/s07.004
9. Rahman, M. S. (2021). E-commerce & Alibaba's business strategy. 1. 20. ResearchGate. https://www.researchgate.net/publication/358199646_E-commerce_Alibaba's_Business_Strategy
10. Amazon. (2020). The SAGE International Encyclopedia of Mass Media and Society. https://doi.org/10.4135/9781483375519.n28

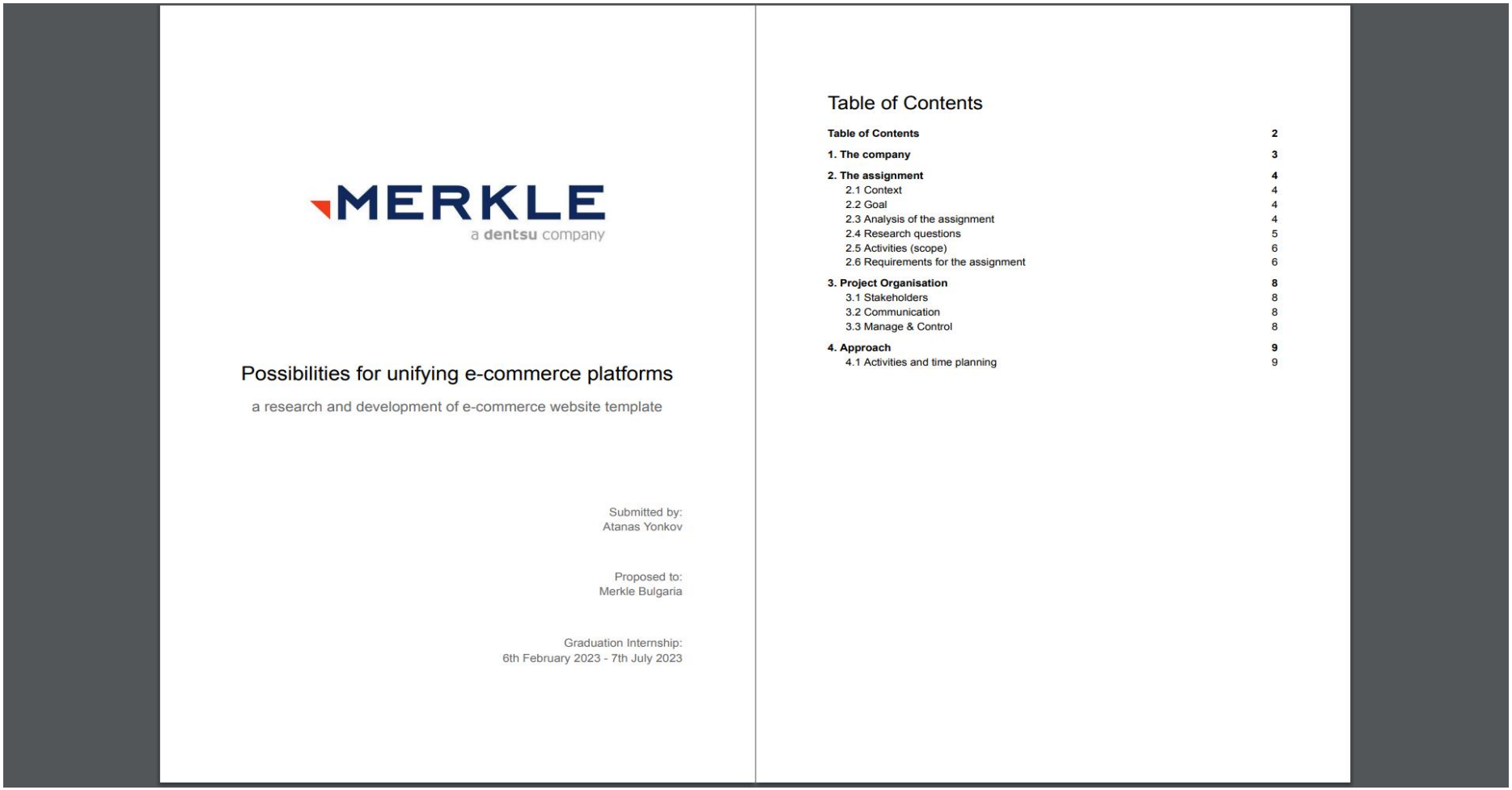
11. Su, C., & Flew, T. (2020). The rise of Baidu, Alibaba and Tencent (BAT) and their role in China's Belt and Road Initiative (BRI). Global Media and Communication, 17(1), 67–86. https://doi.org/10.1177/1742766520982324
12. Staff, C. (2016). React. Communications of the ACM, 59(12), 56–62. https://doi.org/10.1145/2980991
13. Al Moaiad, Yazeed & Ts,. (2023). A Comparative Study Of The E-Commerce Platforms Of Amazon and eBay. pnrjournal.com. https://doi.org/10.47750/pnr.2023.14.S02.152
14. Yogish Pai, U. & Nandha Kumar, K. G., (2021). E-Commerce to Multinational Conglomerate: Journey of Alibaba Group – A Case Study. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 5(1), 25-35. DOI: http://doi.org/10.5281/zenodo.4544394.
15. Nnn. (2023). EBay.com USA: the ultimate online marketplace. NNN FAQ. https://nnn.ng/faq/12035/
16. Round, M. (2004) Presentation to E-metrics, London, May 2005. www.emetrics.org. https://ai.stanford.edu/~ronnyk/emetricsAmazon.pdf
17. (Tryzens., 2021) Part 1: The path to headless on Salesforce Commerce Cloud (SFCC) https://tryzens.com/industry-trends/part-1-the-path-to-headless-on-salesforce-commerce-cloud-sfcc/
18. Salesforce Commerce Cloud B2C Commerce Cartridge installation and Configuration (Video tutorial) | CloudInary. (n.d.). https://cloudinary.com/documentation/salesforce_install_config_tutorial
19. How Does the Frontend Communicate with the Backend? (2019, March 19). vsupalov.com. https://vsupalov.com/how-backend-and-frontend-communicate/
20. Emadamerho-Atori, N. (n.d.). [GuIDE] Is Headless architecture right for you? Prismic Blog. https://prismic.io/blog/headless-architecture-guide
21. Cyntexa. (2022, January 5). Latest Salesforce Commerce Cloud Tools To Increase The Experience Of Developers. Medium. https://cyntexa.medium.com/latest-salesforce-commerce-cloud-tools-to-increase-the-experience-of-developers-946c65effb1e?source=user_profile---------14----------

22. Muldoon, C., Görgü, L., O'Sullivan, J. M., Meijer, W. G., Masterson, B., & O'Hare, G. M. P. (2021). Engineering testable and maintainable software with Spring Boot and React. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING. https://doi.org/10.36227/techrxiv.15147723.v1
23. Uzayr, S. B., Cloud, N., & Ambler, T. (2019). Angular. In Apress eBooks (pp. 209–223). https://doi.org/10.1007/978-1-4842-4995-6_7
24. O. Filipova, Learning Vue.js 2. Packt Publishing Ltd, 2016. https://sd.blackball.lv/library/learning_vue.js_2_(2016).pdf
25. Gasanov, T. (2022). The best React state management tools for enterprise applications. Retrieved from https://www.researchgate.net/publication/301959679_What_Is_React
26. Li, Z., & Xie, F. (2023). Concolic Testing of Front-end JavaScript. In F. Arbab, & M. Sirjani (Eds.), Formal Methods and Software Engineering (pp. 46-63). Springer International Publishing. https://doi.org/10.1007/978-3-031-30826-0_4
27. Thakkar, M. (2020). Adding server-side rendering to your React application. In M. Thakkar (Ed.), Full-stack React projects (pp. 79-96). Apress. https://doi.org/10.1007/978-1-4842-5869-9_4
28. Sayfan, G. (2019). Testing React.js Applications with Jest: A Complete Introduction to Fast, Easy Testing. Apress. https://doi.org/10.1007/978-1-4842-3980-3
29. Mostafa, S., & Wang, X. (2014). An Empirical Study on the Usage of Mocking Frameworks in Software Testing. In Proceedings - International Conference on Quality Software (pp. 127-132). https://doi.org/10.1109/QSIC.2014.19
30. Converting existing Sitecore MVC applications to the Jamstack architecture with Headless Rendering. (n.d.). https://doc.sitecore.com/xp/en/developers/hd/21/sitecore-headless-development/converting-existing-sitecore-mvc-applications-to-the-jamstack-architecture-with-headless-rendering.html
31. The Advantages Of Headless Content Management Systems - NP GROUP. (2016, June 27). https://www.npgroup.net/blog/the-advantages-of-headless-content-management-systems/
32. Singh, J. (2021, December 9). Headless architecture: the new normal | Accenture. Insights. https://www.accenture.com/nl-en/blogs/insights/headless-architecture-why-its-becoming-the-new-normal

33. Creating Reusable React Components for Faster and More Efficient Development. (n.d.). Cronj. Retrieved May 5, 2023, from https://www.cronj.com/blog/creating-reusable-react-components-for-faster-and-more-efficient-development/
34. GeeksforGeeks. (2023). ReactJS Unidirectional Data flow. GeeksforGeeks. https://www.geeksforgeeks.org/reactjs-unidirectional-data-flow/
35. Why use React Redux? | React Redux. (2021, June 19). https://react-redux.js.org/introduction/why-use-react-redux
36. Hintz, D. (2020). What is Redux Anyway? (Part 2). DEV Community. https://dev.to/dhintz89/what-is-redux-anyway-part-2-1gf4
37. Hossain, Mostaqim & Habib, Mubassir & Hassan, Mainuddin & Soroni, Faria & Khan, Mohammad. (2022). Research and Development of an E-commerce with Sales Chatbot. 557-564. 10.1109/AIIoT54504.2022.9817272.
38. Jain, Ankit. (2023). Impacts of cloud computing in India on E- commerce businesses. https://www.researchgate.net/publication/370560501_Impacts_of_cloud_computing_in_India_on_E-_commerce_businesses

APPENDICES

Appendix A - Project plan:



MERKLE
a dentsu company

Possibilities for unifying e-commerce platforms

a research and development of e-commerce website template

Submitted by:
Atanas Yonkov

Proposed to:
Merkle Bulgaria

Graduation Internship:
6th February 2023 - 7th July 2023

Table of Contents

1. The company

Online retail has become more and more important over the years and will probably do so in the future. This has led to e-commerce becoming a big part of the retail industry.

The company for which I will be carrying the assignment is called Merkle. They operate in the e-commerce industry. Merkle's task on the market is providing a one stop-solution for the online commerce presence of various retailers.

The goal of Merkle is to expand and provide better and modern customer experience, optimised ecommerce platforms - overall performance and loading time, security features that can help protect sensitive customer data and prevent fraud, and scalability - making it easier to handle an increasing number of customers and transactions.

The Bulgarian sub-branch is made up of a team of over 350 skilled software engineers, architects, QA engineers, and support specialists. That is where my assignment will take place. Located in a technology centre in Sofia, they have a history of creating and managing e-commerce platforms for big international brands and retailers. The team has launched over 400 e-commerce websites in 60 countries across 6 continents.

2. The assignment

2.1 Context

The company has many clients, some of them for as long as 12 years. Currently, creating a new customised website for a brand typically involves copying an existing website and making adjustments for the new brand. **This means that a lot of Merkle projects are written with the so-called legacy code which is difficult to maintain and upgrade with new features, thus slow to deliver and sometimes losing new clients.**

2.2 Goal

The project goal is to develop and possibly deploy a fully working e-commerce website. Merkle wants to invest time in a so-called 'reference architecture' – a modern website with just the essential features. There shouldn't be any particular styling or catalogue of products. This website should be the starting point for new projects for new clients. Such an architecture would allow software engineers to quickly develop a website tailored to the customer's needs. That way the company would be able to quickly 'hook' and keep potential clients. In the long term, that should allow for more clients in less time and thus gaining more money.

2.3 Analysis of the assignment

In general, the problem that the company is facing could be looked at from 2 sides. On one hand, it is losing potential new clients and on the other - increasingly difficult maintenance for existing clients. The reason for that issue seems to be simply 'time'. While Merkle has been scaling up, more time was devoted to building a working product, rather than an optimal product.

Furthermore, the Salesforce platform on which many clients' websites take place is being constantly updated. Recently there was a big release from their side, supporting many new features, most notable of which is headless commerce. Many old websites for clients couldn't benefit from that due to old code.

Thus without devoting time for improvements you end up building a website by old standards which is a formula for losing clients and money.

Creating a reference architecture website is the solution for both problems - it would allow for old websites to adopt this new structure and improve performance, as well as provide potential clients with a quick initial solution and hopefully seal a contract.

But creating such a website is a challenge on its own. The web industry is developing rapidly and in the time span of a few years new frameworks, methods and libraries come and go. My challenge would be picking the right tools and the right puzzle pieces in order to assemble a solution that actually helps or even improve Merkle's presence on the market.

2.4 Research questions

1. **What are the essential features of a reference architecture for e-commerce website development?**
    a. What do e-commerce websites do and what do they solve?
    b. What backend do they use?
    c. What are the most important features that should be included to meet users' experience expectations?
    d. What are the current technology trends?

    Research method: Library research, best good and bad practices, design pattern search, trend analysis

2. **How to connect the reference architecture with existing technologies?**
    a. What is headless development, how does it work, how is it used within the context of the company?
    b. How to implement and utilise optimal headless development in the context of Merkle?

    Research method: Competitor research of companies that have successfully developed and implemented reference architectures for e-commerce website development, context mapping

3. **How and what to design and develop?**
    a. How to develop features in an effective way for Merkle?
    b. What does it mean to design code for flexibility and maintainability in the long run?

    Research method: Expert interviews and surveys, fly on the wall type, A/B testings, thinking aloud, usability testing, to gather insights from software engineers and web developers with experience in e-commerce website development.

4. **How can the effectiveness of the project be evaluated?**

    Research method: Surveys and interviews with smaller team groups in the company like the business and sales teams to gather their perceptions and attitudes towards the company's website development process and the use of reference architectures.

2.5 Activities (scope)

Since I don't have contact with direct clients of the company (retailers), the guidance will come from my mentor, with whom I have discussed several important topics mentioned below. However, I would try to take the initiative and do an internal research within each of the smaller team groups in the company. The user groups would include developers who have built previous websites as well as project owners and team leads who would provide crucial insight on non-technical problems when dealing with clients. With that I would try to find the answer for problems any specific developer has found when working on a client project.

- Research answering the questions of the MoSCoW strategy below - functionalities with relevant content, then a deeper look at what is the best stack for those features- which database, which front-end framework?
- Develop a full-stack base template website – front-end and back-end. As guidance I would focus on the outputs from the 2 main important researches during this project:
    - The technical research looking for answers on how to technically build the website
    - The internal research looking for actual opinions of user groups mentioned above

2.6 Requirements for the assignment

The main deliverable which falls directly in the scope of the project is the actual website. My main task is to create a 'reference architecture' template website available for quick delivery to a new brand in order to maximise the turnover rate.

Must Have:

- **User registration and login functionality**
  How can user registration and login functionality be implemented securely and effectively?

- **Product listing and search functionality**
  What are the essential features of a product listing and search functionality for an e-commerce website?

- **Shopping cart and checkout process**
  How can the shopping cart and checkout process be designed to be user-friendly and efficient?

Should Have:

- **Product reviews and ratings system**
  What are the benefits and challenges of implementing a product reviews and ratings system for an e-commerce website?

- **Wish list and save for later functionality**
  How can a wishlist and save for later functionality be designed and implemented to improve the user experience?

- **Multiple payment options (e.g. credit/debit cards, PayPal, etc.)**
  How can an e-commerce website integrate multiple payment options effectively and securely?

Could Have:

- **User rewards and loyalty programs**
  What are the potential benefits and drawbacks of implementing user rewards and loyalty programs for an e-commerce website?

- **Social media integration for sharing and promoting products**
  How can social media integration be used to promote products and improve the user experience?

Won't Have:

- Support for outdated or unsupported web browsers
- Complex AI-powered features for personalised shopping experiences
- Integration with niche or obscure payment gateways
- Non-essential features that don't align with the core purpose of the website
- Advanced security features beyond industry-standard protocols

3. Project Organisation

3.1 Stakeholders

Name	Role
Lancee,Wouter W.	Teacher / University Mentor
Antonov, Emil	Company Mentor
Yonkov, Atanas A.	Intern

3.2 Communication

During the internship I will write a bi-weekly summary via Feedpulse, in which I describe the activities and products of the week in question. My company mentor shall be updated at the end of each sprint, validating the output and moving on with the next sprint tasks. Emil is also available to provide technical and non-technical guidance and assistance if needed.

3.3 Manage & Control

To track my progress and tasks I will be using Jira. This tool would allow for clean management and big-picture preview for all the tickets - their type (technical / bugfix) and their priority.

4. Approach

4.1 Activities and time planning

Phase 1 (February - March)

In phase 1, I will research and document e-commerce trends and principles. I will create a set of tasks and stories with acceptance criteria to be added to the project management tool JIRA.

1. What are the essential features of a reference architecture for e-commerce website development?
2. How can the reference architecture be designed to be flexible and adaptable to different client needs?
3. What are the impacts of using a reference architecture on client acquisition and retention?

Phase 2 (March – Late May)

In this phase, I will use the Agile method and Scrum to technically develop the project. I will work in sprints of about 1.5 weeks and plan and check progress with my mentor.

1. User registration and login functionality
2. Product listing and search functionality
3. Shopping cart and checkout process
4. Product reviews and ratings system
5. Wish list and save for later functionality
6. Multiple payment options (e.g. credit/debit cards, PayPal, etc.)

Phase 3 (June – July)

Wrap-up, final changes, document, technical time for administrative tasks, present and defend thesis

Appendix B:

Research Document ‘Essential features of reference architecture for e-commerce websites’:

MERKLE
a dentsu company

Possibilities for unifying e-commerce platforms

a research and development of e-commerce website template

Research:

Essential features of reference architecture for e-commerce websites

Atanas Yonkov

Table of contents

What do e-commerce websites do and what do they solve?

Brief history and overview of e-commerce

Electronic commerce, more commonly known as 'e-commerce' is the action of buying and selling of goods or services happening online. Typically occurring through a website, in simple terms that is the process of conducting a business electronically.

While commerce has existed for ages, it is said that e-commerce began in the 80s, when in 1982, the first online marketplace, Boston Computer Exchange, opened for business. That happened after the first use of a transaction-processing computer and a doctored television to create the very first secure data transmission, laying the groundwork for online shopping in 1979, by Michael Aldrich. (George Washington University's Online Healthcare MBA, 2020b)

E-commerce could be coming in several forms (*Types of E-commerce*, n.d.-b), most known of which are B2B - Business to Business, like a company selling a product, service or a subscription to another company - Microsoft 365 for example. However, the same product could also be sold to an individual, and that's referred to as B2C - Business to consumer. The other way around is C2B - Consumer to business, where an individual sells a service to a company, like business analytics or another valuable skill. If we totally exclude businesses from the conduction of businesses, we often end up with C2C - Consumer to consumer. That often involves transactions through marketplaces like eBay.

What problems do e-commerce websites solve for businesses?

One way e-commerce helps businesses is by giving the ability to increase the 'customer reach'. In the context of e-commerce, that refers to reaching a wider audience that wouldn't be possible with traditional brick-and-mortar stores. For one, online businesses can operate 24/7, which means customers can shop at any time. Removing opening and closing times from the equation, clients with busy schedules could reach the shop or service effortlessly. Second, e-commerce businesses do not necessarily need anything physical apart from a computer. The whole idea is that transactions happen online, which unlocks the ability to visit and conduct business from anywhere in the world. With a website, businesses could reach beyond their local areas, **expanding the potential customer base**.

Another benefit of e-commerce **eliminating physical stores** and running online is the reduction of overhead costs. Cutting on a physical site to run a business also reduces expenses such as rent, utilities and maintenance. Additionally, online stores could run with smaller staff, as they now do not need as many employees as in a conventional store.

The **reduction of overhead costs** in comparison to physical stores is one of the main advantages of e-commerce. It eliminates many of the costs associated with operating a physical store, such as rent, utilities, and maintenance. This allows businesses to reduce their overhead costs significantly, which can lead to higher profit margins. Additionally, online businesses can operate with smaller staff, as they do not need as many employees as a physical store would. For example, according to a report by the U.S. Small Business Administration, "According to the International Monetary Fund, 87% of the world's economic growth will take place outside of the US. For Cross border B2C e-commerce, that represents over US 1.2 trillion in Gross Merchandise Value (GMV) transactions by the year 2022 – with a 25% growth rate.."

Streamlining of business operations refers to the process of making a business more efficient by automating processes, eliminating redundancies, and reducing errors. E-commerce can help automate many business processes such as inventory management, order and payment processing, which can be handled automatically by an e-commerce platform.

What problems do e-commerce websites solve for consumers?

As I have mentioned in the previous paragraph, e-commerce allows for an expanded customer base and not necessarily physical stores. That could be not only beneficial for businesses, but for customers as well. With online shops, clients could shop from the convenience of their coach and have the products delivered next to their doorstep. For example, a survey by the National Retail Federation found that "72% of consumers say that the **ability to shop at any time** is one of the key benefits of online shopping."

Furthermore, consumers have the benefit to choose from a **wider variety of products and services**. The sole fact that customers could shop for products and services all around the globe, means that they have more choices and options to find the option that best meets their needs.

However, a wider range of products creates a problem on its own. With a few shops selling a similar product, how could the customers know which is the one with best quality, which is the one that meets the best price per buck? **Comparison shopping, product reviews and ratings** help with informed decision making. Those features in a way hook and engage customers since it reduces risk of purchasing a product that doesn't meet their expectations.

Examples of successful e-commerce websites and their impact

It wouldn't be correct to speak of the benefits of e-commerce without mentioning a few great examples of successful e-commerce platforms.

Amazon is on the largest e-commerce websites in the world. With a wide range of products and services available, their strategy includes offering competitive prices, fast-shipping and user friendly websites. Amazon's business model involves a combination of direct sales and third party sellers, thus engaging a global client base. Their impact on conventional stores is

huge, with many physical retailers struggling to compete with Amazon's low prices and convenience.

Alibaba is a Chinese e-commerce company that usually specialises in online marketplaces and payment systems. Their business strategy is providing a platform for businesses to sell products and services around that world in large stocks with great discounts, thus achieving highly competitive prices. The business model they have is charging fees for transactions and advertising after giving small businesses a platform to operate on. Their impact has been significant as they have disrupted industrial supply chains and enabled small businesses to operate on a global scale.

Those have been examples of platforms on B2B and B2C levels, but further there is a great example for a successful C2C platform - eBay. eBay is an online marketplace where individuals could buy and sell products to other individuals.

What backend do e-commerce use?

Content & Customer Management Systems

Whenever it comes to using different methods of reaching more customers for e-commerce, creating a web application is considered one of the top methods in this regard. Web applications more commonly consist of two sections, i.e., frontend and backend. The front end is the most often visible on the user's screen and is meant for users to interact with the application. However, in general terms, the backend consists of hardware resources, software technology, and application infrastructure that powers your website or application.

A **Content Management System** (CMS) is software that enables users to create, manage, and modify content on a website without necessarily needing technical knowledge. Users could manage and organise product catalogues, add or remove products, edit product description and images. Some examples of the most popular CMS platforms include - WordPress, Joomla and AWS.

Customer Relationship Management (CRM) is a strategy that companies use to manage interactions with customers and potential customers. It takes the CMS software a step further by also including the ability to manage customer interactions and improve customer satisfaction. It helps businesses to track consumer's behaviour, purchase history and preferences, and personalise the shopping experience.

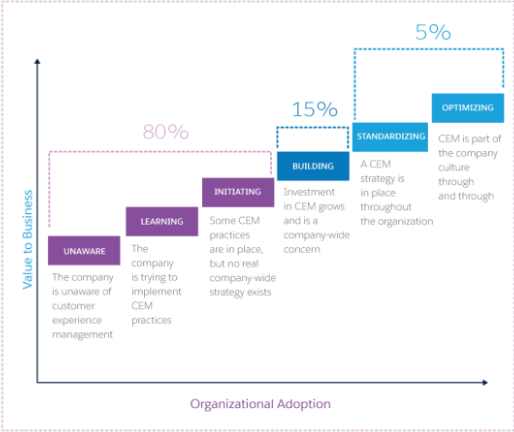
Customer Experience Management (CXM) is often confused with customer relationship management, or CRM, due to the similarity that they have in some basic functionality. While there is some overlap, CXM isn't a mere renaming of traditional CRM. CXM takes the next step in managing customer relationships. Where CRM mostly deals with optimising the internal processes of a business, CXM offers an even more customer-centric approach in several ways—with new technologies, as well as processes, strategies, and customer-centric design.

CXM differs from traditional CRM in the technology itself—with added benefits and capabilities for building customer relationships. While CRMs gather data through manual or batch entry, a true CXM will enable a real-time flow of data to offer deeper insights into customer preferences and behaviour.

(Kinsta, 2023)

SalesForce Commerce Cloud

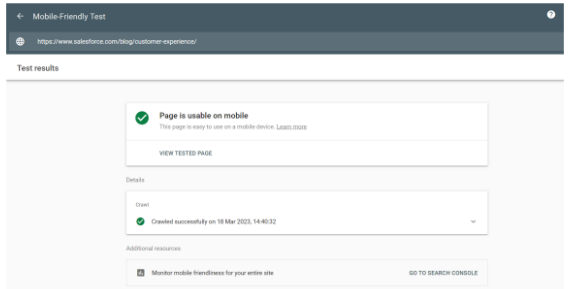
In the context of Merkle, the company where this graduation project is conducted, Customer Experience Management comes with the help of SalesForce's Commerce Cloud (SFCC). Previously known as Demandware, that is the backend software upon which many of the company's projects sit upon.



Source: fico.org

What are the most important features that should be included to meet users' experience expectations?

E-commerce websites should be designed to be **mobile-friendly, with a responsive design** that adjusts to different screen sizes. Many users now shop on mobile devices, and a website that is not optimised for mobile can be frustrating and difficult to use.



Picture 1. Google Developers. (n.d.). Mobile-friendly test. https://search.google.com/test/mobile-friendly

Easy navigation: E-commerce websites should have clear and intuitive navigation, making it easy for users to find what they are looking for. This includes a well-organised menu structure, breadcrumbs, and search functionality. (White, 2021)

High-quality product images and videos: Users expect to see high-quality images and videos of products they are considering purchasing. These images should be large, zoomable, and show the product from multiple angles. (*Perfect Product Photography: Craft a Perfect Setup in 2023*, 2022)

User-generated reviews and ratings: Including user-generated reviews and ratings on product pages can help users make informed decisions about what to buy. Users trust the opinions of other customers and appreciate the transparency.

Fast loading times: E-commerce websites should load quickly, as users have little patience for slow-loading pages. This can be achieved through optimising images, compressing files, and using a content delivery network.

Secure checkout: E-commerce websites must prioritise security and include a secure checkout process. This includes SSL encryption, secure payment gateways, and compliance with industry standards such as PCI-DSS.

Personalization: Users appreciate a personalised shopping experience, with recommendations based on their purchase history, browsing behaviour, and preferences. This can be achieved through the use of AI and machine learning algorithms. “Advanced personalization features have led to 70% of retailers having a return on investment of 200% or more, according to Kibo.”
(Cassidy, 2020)

What are the current technologies and trends?

Current technologies

By default it is agreed upon the fact that a classic e-commerce website should have 3 fundamental things - landing page, product listing page and product details page. Without those 3 it wouldn’t be relevant to refer to such a website as an e-commerce website. From there the features are from must-have, all the way to do-not-include spectrum. (Nedamo, 2023)

Challenges and future trends in e-commerce

As beneficial as it might be, currently the situation in e-commerce retailers might not be all roses. However, let’s begin with the positive outcome from bad situations.

The pandemic unlocked a new level of online commerce, no one has ever seen before. According to Sanjay Gupta, Country Head and VP, Google India, the silver lining to the whole situation is that businesses have squeezed four years of change into four months since the lockdown began, from a business transformation and skills perspective. Microsoft’s CEO Satya Nadella’s statement on digital transformation further fuels the argument that the e-commerce sector will thrive in the ‘new normal’: “We’ve seen two years’ worth of digital transformation in two months. From remote teamwork and learning to sales and customer service, to critical cloud infrastructure and security—we are working alongside customers every day to help them adapt and stay open for business in a world of remote everything.”

But the pandemic, as it seemed to be a huge boost for the e-commerce industry, left it faced with serious challenges on its own. One of the biggest threats for customers and businesses is related to cyber and data security. Being possibly fully online, as more consumers shop online there is increasing risk of fraud and identity theft. There is a lot of information and data involved in the process of shopping from an online retailer and a slight problem could go a long way in damaging the retailer's daily operations as well as brand image.

That could lead to another issue related to issues related to delivery and logistics. The pandemic proved a double-sided sword for the industry as it showed a huge increase in customers but on the other hand facing problems with supply lines and slow deliveries as there were no employees working in the factories due to the lockdown.

In the past few years, not only consumers moved to online shopping, but also retailers adapted their business to work on the web. With more and more incoming offers for clients, businesses could have problems with not possessing the best offer. On this train of thoughts, it is relevant to explain the importance of customer experience, loyalty and even in general converting shoppers to customers. Technological advancements could go a long way in hooking up potential customers. Artificial intelligence, augmented or virtual reality are all capable of transforming the e-commerce landscape. For example, AI can be used to personalise the shopping experience for customers, while AR and VR can be used to create immersive and interactive shopping experiences. Finally, the potential for new business models and the evolution of the e-commerce landscape is another trend to watch. For

example, the rise of social commerce, where customers can purchase products directly from social media platforms, is changing the way consumers shop online. In addition, the growth of mobile commerce and the increasing use of mobile devices for online shopping are also shaping the future of e-commerce.

Conclusion

In conclusion, e-commerce websites have revolutionised the way businesses operate and customers shop. These platforms have solved numerous challenges and provided significant benefits for both businesses and consumers.

For businesses, e-commerce websites have expanded their customer reach by enabling them to reach a wider audience beyond their local areas. Online stores operate 24/7, allowing customers to shop at their convenience, and businesses can operate with lower overhead costs compared to physical stores. The streamlining of business operations through automation and efficient processes further enhances the efficiency and profitability of e-commerce businesses.

On the consumer side, e-commerce websites offer convenience and accessibility. Customers can shop from the comfort of their homes and have products delivered to their doorstep. The availability of a wide variety of products and services from around the globe provides customers with more choices and options to find what best meets their needs. Comparison shopping, product reviews, and ratings enable informed decision-making, reducing the risk of purchasing unsatisfactory products.

To create a positive user experience, e-commerce websites should prioritise mobile-friendliness, easy navigation, high-quality product images and videos, user-generated reviews and ratings, fast loading times, secure checkout processes, and personalization based on customer preferences and behaviours. These features enhance customer satisfaction and loyalty, leading to higher returns on investment for businesses.

The current technologies and trends in e-commerce include the use of content and customer management systems like CMS and CRM, with advanced customer experience management (CXM) taking a more customer-centric approach. SalesForce Commerce Cloud (SFCC) is an example of a backend software that supports customer experience management.

Looking ahead, technological advancements such as artificial intelligence, augmented reality, and virtual reality have the potential to transform the e-commerce landscape, offering personalised and immersive shopping experiences. The evolution of new business models, such as social commerce and mobile commerce, further shapes the future of e-commerce.

Overall, e-commerce websites have provided solutions to traditional commerce challenges, driving economic growth, improving efficiency, and enhancing the shopping experience for businesses and consumers alike. As the industry continues to evolve, embracing new technologies and addressing emerging challenges will be crucial for sustained success in the e-commerce sector.

MoSCoW chart

Must-Have	Should-Have	Could-Have	Won't-Have
User Registration and Authentication: Allow users to create accounts, log in, and securely authenticate their identities.	**Responsive Design:** Ensure the website is optimised for various devices and screen sizes to provide a seamless user experience.	**Social Media Integration:** Allow users to share products on social media platforms, increasing brand exposure and potential sales.	**Multilingual Support:** Exclude multilingual support in the initial development phase, focusing on a single language.
Product Catalogue Management: Enable administrators to add, edit, and delete products, including details like images, descriptions, and prices.	**Product Filtering and Sorting:** Allow users to filter products based on attributes (e.g., price range, brand, colour) and sort them by relevance, price, popularity, etc.	**Personalised Recommendations:** Implement recommendation algorithms to suggest related products based on user preferences and browsing history.	**Advanced Analytics and Reporting:** Omit comprehensive analytics and reporting features, such as sales reports or customer behaviour analysis.
Shopping Cart Functionality: Allow users to add products to their carts, review and modify cart contents, and proceed to checkout.	**Wishlist:** Enable users to create and manage wishlists, save products for future purchases, and share them with others.	**Multiple Payment Options:** Offer various payment methods, such as credit cards, digital wallets, and alternative payment gateways, to accommodate different user preferences.	**Social Login:** Exclude the ability for users to log in or register using their social media accounts.
Payment Gateway Integration: Integrate with a secure payment gateway to process online payments from customers.	**Product Reviews and Ratings**: Allow users to leave reviews and ratings for products, enhancing credibility and helping others make informed decisions.	**Order Tracking for Customers:** Provide a tracking feature for customers to monitor the status and location of their orders.	

References

1. George Washington University's Online Healthcare MBA. (2020b, October 22). History of E-Commerce | George Washington University. George Washington University. https://healthcaremba.gwu.edu/history-of-ecommerce/#:~:text=1979%20%E2%80%94%20English%20inventor%20Michael%20Aldrich,Computer%20Exchange%2C%20opens%20for%20business.

2. Kinsta. (2023, March 20). What Is a Content Management System (CMS)? Kinsta®. https://kinsta.com/knowledgebase/content-management-system/

3. Types of e-commerce. (n.d.-b). Bloomidea. https://bloomidea.com/en/blog/types-e-commerce

4. Nedamo, P. E. (2023, May 25). 10 Must-Haves of an eCommerce Website in 2022. Envato. https://www.envato.com/blog/ecommerce-website-features/

5. Neusser, S. (2022, October 14). Top Technologies Used in E-Commerce Businesses (+10 Examples). Sana Commerce. https://www.sana-commerce.com/blog/technologies-used-in-ecommerce/

6. Cassidy, T. (2020). Personalization helps retailers earn a higher ROI. Digital Commerce 360. https://www.digitalcommerce360.com/2020/04/29/personalization-helps-retailers-earn-a-higher-roi/

7. Perfect Product Photography: Craft a Perfect Setup in 2023. (2022, October 2). Shopify. https://www.shopify.com/blog/12206313-the-ultimate-diy-guide-to-beautiful-product-photography

8. White, A. K. (2021, December 15). (Almost) Everything you wanted to know about designing website navigation. Medium. https://uxdesign.cc/almost-everything-you-wanted-to-know-about-designing-website-navigation-52bb96dce93d

Appendix C:

Research Document ‘Connecting Reference Architecture with existing technologies’:

MERKLE
a dentsu company

Possibilities for unifying e-commerce platforms

a research and development of e-commerce website template

Research:

Connecting reference architecture with existing technologies

Atanas Yonkov

Table of contents

Headless development - theory and principles

Overview

Headless development refers to the process of creating software without a graphical user interface (GUI). In headless development, the focus is on building software that runs on a server or backend system and interacts with other software and services through APIs (Application Programming Interfaces). (*What Is a Headless Website?*, n.d.)

Headless development is commonly used for web development, where the focus is on building server-side applications that interact with client-side applications through APIs. This approach enables developers to create software that can be used across different devices and platforms, without having to develop and maintain separate GUIs for each one.

Headless development is also used in the context of content management systems (CMS), where it allows developers to separate the content creation and management functions from the presentation layer. This enables them to create custom front-ends for different devices and channels, while maintaining a consistent back-end system for managing content. (Wikipedia contributors, 2023)

Overall, headless development allows developers to focus on building the underlying logic and functionality of software applications, without being constrained by the limitations of a GUI.

How does it work?

In headless development, the software is designed to interact with other software and services through APIs (Application Programming Interfaces), rather than through a graphical user interface (GUI).

This means that the software is built to receive requests and send responses in a structured format, usually using a standardised protocol like HTTP or JSON. The API provides a set of endpoints that can be used to perform specific actions or retrieve specific data from the system.

To use the headless application, a client-side application is required that can interact with the API to send requests and receive responses. This client-side application can be a mobile app, a web application, or any other type of software that is capable of communicating with the API.

For example, consider a headless CMS that is used to manage content for a website. The CMS might provide a set of API endpoints that allow developers to retrieve content from the system, create new content, and update existing content. A client-side web application can then be built that uses these API endpoints to display the content on a website.

The advantage of this approach is that the client-side application can be customised for different devices and platforms, without having to modify the back-end system. It also enables developers to use a wide range of programming languages and tools to build client-side applications, without having to learn the details of the back-end system.

Overall, headless development provides a flexible and scalable way to build software systems that can be used across different devices and platforms, and that can be customised to meet specific business requirements.

Implement and utilise optimal headless development in within Merkle

Headless in the context of the company

Salesforce can be connected to Node.js through the Salesforce REST API, which allows developers to interact with Salesforce data and metadata using HTTP requests. Node.js is a popular server-side JavaScript runtime that can be used to build backend applications and APIs, and it provides a number of features and libraries that can be helpful for integrating with Salesforce. In the context of Merkle the technology is used to:

- Building custom integrations: Developers can use Node.js to build custom integrations between Salesforce and other systems or services. For example, they could use Node.js to create a webhook that triggers an action in Salesforce when a new order is placed on an e-commerce website.
- Creating custom APIs: Developers can use Node.js to create custom APIs that interact with Salesforce data and metadata. For example, they could create an API endpoint that retrieves customer data from Salesforce and returns it to a front-end application.
- Building serverless functions: Developers can use Node.js to create serverless functions that interact with Salesforce data and metadata. For example, they could create a function that automatically updates a customer record in Salesforce when a user submits a form on an e-commerce website.

(*Headless Implementation Strategies for Salesforce B2C Commerce*, n.d.)

What is the best front-end framework for that backend?

There is no one "best" frontend technology for maintainability in e-commerce websites, as different technologies may be better suited for different projects or development teams. However, some frontend technologies that are commonly used for e-commerce websites and are known for their maintainability include:
(Joshi, 2023)

- React: React is a popular JavaScript library for building user interfaces, and is known for its modular, component-based architecture that makes it easy to maintain and update code over time. It also has a large and active developer community, which can be helpful for finding solutions to common problems or getting advice on best practices.
- Vue.js: Vue.js is another JavaScript framework for building user interfaces, and is known for its simplicity and ease of use. It has a smaller learning curve than some other frontend technologies, which can make it easier to maintain over time, especially for smaller teams or projects.
- Angular: Angular is a more comprehensive framework for building web applications, and is known for its scalability and robustness. It has a large and active developer community, and provides a number of tools and features that can help with maintaining code over time, such as dependency injection and built-in unit testing.

Ultimately, the choice of frontend technology will depend on a number of factors, such as the size and complexity of the project, the skill level of the development team, and the specific requirements of the e-commerce website. It's important to carefully evaluate different technologies and choose one that is well-suited to the project and the team's needs, as this can have a significant impact on the maintainability and scalability of the code over time.

Conclusion

In conclusion, headless development offers a powerful approach to software development by focusing on building server-side applications that interact with other software and services through APIs, without the constraints of a graphical user interface (GUI). This methodology is particularly beneficial for web development and content management systems, allowing developers to create software that can be used across different devices and platforms while maintaining a consistent back-end system.

Within the context of Merkle, the integration of Salesforce with Node.js through the Salesforce REST API allows developers to interact with Salesforce data and metadata using HTTP requests. Node.js serves as a powerful server-side JavaScript runtime for building backend applications and APIs, enabling custom integrations, creation of custom APIs, and development of serverless functions that interact with Salesforce.

When it comes to choosing the best front-end framework for the backend, there is no definitive answer as it depends on the specific project and development team. However, some frontend technologies known for their maintainability in e-commerce websites include React, Vue.js, and Angular.
React is lauded for its modular and component-based architecture, Vue.js for its simplicity, and Angular for its scalability and robustness. In order to get the answer to that question, I would have to follow up with more specific research of the tech stack used in the projects and what developers are used to.

By embracing headless development and making informed choices regarding frontend technologies, Merkle can leverage the benefits of flexibility, scalability, and customizability to build innovative software solutions that meet the evolving needs of its clients.

References

1. What is a headless website? (n.d.). Algolia Blog. https://www.algolia.com/blog/ux/what-is-a-headless-website-a-definition-and-examples/#:~:text=Headless%20web%20development%20frees%20front,unique%20that%20facilitates%20better%20interaction.

2. Wikipedia contributors. (2023). Headless content management system. Wikipedia. https://en.wikipedia.org/wiki/Headless_content_management_system

3. Headless Implementation Strategies for Salesforce B2C Commerce. (n.d.). https://trailhead.salesforce.com/content/learn/modules/b2c-headless-implementation-strategies/b2c-explore-headless-roles-architect

4. Joshi, M. (2023). Angular vs React vs Vue: Core Differences | BrowserStack. BrowserStack. https://www.browserstack.com/guide/angular-vs-react-vs-vue#:~:text=A%20simple%20difference%20between%20these,Vue%20with%2018.97%25%20of%20developers

Appendix D:

Research Document ‘Designing and Developing Features for Merkle’:

MERKLE
a dentsu company

Possibilities for unifying e-commerce platforms

a research and development of e-commerce website template

Research:
Designing and Developing Features for Merkle
Atanas Yonkov

Table of contents

Introduction

This research document explores the effective ways of developing features for Merkle, with an emphasis on designing code for flexibility and maintainability in the long run. The research method used was expert interviews, surveys, fly on the wall type, A/B testings, thinking aloud, and usability testing, which allowed us to gather insights from software engineers and web developers with experience in e-commerce website development. The document includes a discussion on the importance of designing and developing features with flexibility and maintainability in mind, as well as practical advice on how to achieve these goals.

Background and Literature Review

Current best coding practices and standards

Code quality is a crucial aspect of software development, and it is important to establish guidelines to ensure that the code is correct, complete, readable, and secure. In a recent research article, a set of code requirements and guidelines were proposed for commits to pass and be merged on production. The requirements were logically organised into categories and subcategories for ease of use. (Bellairs, 2019)

One of the major requirements is that the code should be logically correct. Additionally, the code should cover all required use cases and scenarios, and the solution should be as simple and elegant as possible. The code should also adhere to the agreed company and/or project specific coding conventions and standards, including the location of braces, variable and function styles and naming conventions, line length, indentations, formatting, and comments.

Furthermore, the code should have a well-organised folders/files/modules structure, and there should be no dead or unreachable code. The code should also be unit-testable and unit tests should be written whenever possible. In terms of performance, caching should be utilised whenever possible, and computational complexity should be as low as possible. Data inputs must be validated on the server side, and sensitive information should be stored and accessed securely. (*Coding Standards for Quality and Compliance | Perforce Software*, n.d.)

Finally, the code should comply with country-specific, legal, or other regulations. For instance, if the code is intended for use in the European Union, it should comply with GDPR, while if it is meant to be used in the United States, it should comply with PCI.

One such practice is to avoid content-heavy designs, such as pages with multiple carousels, as they can result in poor performance. Another example is to avoid embedding videos on the page and instead use a video player, such as the YouTube player API, to stream the video on demand. It is also recommended to keep separate JS and CSS files for each page to avoid large files that can block the browser while downloading and parsing.

To achieve this, the best practice is to keep separate CSS and JS files for each page, which can then be bundled automatically for each page using tools such as Webpack. However, if the files are already too large, one can separate the used code from the unused code for landing pages and load the separate CSS/JS file for the page in the head. The global combined CSS can be pre-loaded in a non-blocking way, and the execution of the global combined JS file can be deferred.

There are several tools available to track down unused code, such as the DustMe Firefox extension and the Chrome DevTools Coverage tab. YellowLabs is another tool that can be used to spot different issues with front-end code, such as unused selectors and duplicate ones.

It is also recommended to load the initial view content first, such as images and other content in the initial view, to improve user-perceived performance. Lazy-loading of secondary content, such as content below the fold line or content displayed on some interaction, can also improve performance. Images can be lazy-loaded by using placeholders in the initial HTML and replacing them with the actual image at onload or on clicking the button or scrolling.

Finally, it is recommended to not load desktop-only content on mobile devices to avoid downloading unnecessary resources. While having a single code base for each device can provide maintainability and scalability benefits, it should not hurt performance. One exception to this rule won't hurt these two qualities that much and can provide significant performance benefits.

These are all great tips for optimising JavaScript performance! I'll summarise them below:

- Use minification and uglification to reduce file sizes and improve load times.
- Defer loading and execution of non-essential scripts that don't update the page initial view.
- Use variables to store frequently accessed data.
- Avoid forced synchronous layouts (reflow) and layout thrashing by minimising changes to the DOM and applying styles using predefined CSS classes.
- Prioritise silky-smooth page updates for scrolling performance after the page has loaded.

(*Chrome DevTools - Chrome Developers*, n.d.)

Lazy loading is a technique that delays the loading of non-critical images until they are needed. This helps to reduce the initial load time of the page and improve the perceived performance. There are several ways to implement lazy loading of images:

Intersection Observer API: This is a modern browser API that allows you to track when an element becomes visible in the viewport. You can use this API to lazy load images as the user scrolls down the page.

Lazy Load plugin: There are several plugins available that implement lazy loading of images. One popular plugin is lazysizes, which is lightweight and easy to use.

Custom JavaScript: You can also implement lazy loading using custom JavaScript. This involves detecting when an image enters the viewport and then replacing the placeholder with the actual image.

Minimise image requests

The number of requests made to the server affects the load time of the page. To minimise image requests, you can use CSS sprites, which combine multiple images into a single file. This reduces the number of requests made to the server and improves the load time. Alternatively, you can use data URIs to embed small images directly into the HTML or CSS file. This eliminates the need for a separate HTTP request for the image. (Coyier, 2021)

Choose the right image format

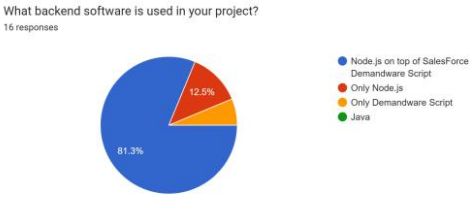
Choosing the right image format is important for performance. There are three main image formats used on the web: JPEG, PNG, and GIF. JPEG is a good choice for photographs and complex images with lots of colours. PNG is better for images with transparency and simple graphics with fewer colours. GIF is best for simple animations. WebP is a newer image format developed by Google that provides better compression than JPEG and PNG. However, it is not yet widely supported by all browsers. (Google TechTalks, 2009)

Field Research

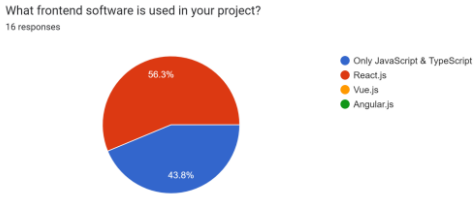
Survey sent to employees of Merkle (with results)

(*Code Maintainability Survey Merkle (Responses)*, n.d.)

This survey provides insights into the opinions and experiences of 16 developers with varying levels of experience working in e-commerce development. The survey asked about their job title, years of experience, the importance of designing code for flexibility and maintainability, maintaining legacy code, backend and frontend software used, the difficulty of writing new code when a new demand comes from the client, the importance of flexibility and maintainability as the size of an e-commerce website grows, testing, and ensuring code is maintainable, approaches to write code for flexibility and maintainability, common issues encountered in maintaining e-commerce websites, strategies found effective for maintaining code, and how often code is refactored.



Most respondents were software developers with 3-5 years of experience, and the majority worked with Node.js on top of SalesForce Demandware Script and React.js. All respondents considered designing code for flexibility and maintainability as very or somewhat important, and most had encountered issues with maintaining legacy code. Respondents agreed that the difficulty of writing new code increases as an e-commerce website grows, and that the importance of flexibility and maintainability becomes more important as well. All respondents rated testing and ensuring code is maintainable as very or somewhat important. The common approaches to write code for flexibility and maintainability are following industry best practices, consulting with team members, writing consistent documentation, and doing code reviews.



The most common issues encountered in maintaining e-commerce websites were slow page load times, difficulty making changes to the site, bugs and errors, and poor performance on mobile devices. Strategies found effective for maintaining code include regular code reviews, consistent documentation, frequent refactoring, and regular performance monitoring. Respondents reported that code is refactored on a regular basis, with some doing it every quarter or every two sprints, while others do it every year or when a problem on production occurs.

Expert Interview with a Software Developer in Merkle

***Interviewee**: Sasho Spayski, Software Engineer*

What are some common issues you've encountered when maintaining code for the e-commerce project you are working on?

- One of the most common issues is dealing with legacy code that was not designed with maintainability and flexibility in mind. This can make it difficult to make changes or add new features without introducing bugs or affecting the performance of the site.
- Another issue is ensuring that the code is scalable as the site grows. This requires designing the code in a way that allows for easy expansion and the ability to handle large amounts of traffic without causing the site to slow down or crash.

What approaches do you use to design code for flexibility and maintainability?

- One of the most important approaches is following industry best practices, such as adhering to coding standards and guidelines and using design patterns that are known to be effective.
- Consulting with team members is also important, as it can help identify potential issues early on and ensure that everyone is on the same page when it comes to design decisions.
- We also do regular code reviews in our project - a team of QA specialists has to verify the quality before launching the code to the production instance. That is another effective approach, as it allows for feedback and suggestions from other specialists who may spot issues that were missed during the initial development process.

How important is testing in ensuring code flexibility and maintainability?

- Testing is extremely important for ensuring that code is both flexible and maintainable. Without proper testing, it can be difficult to identify issues or potential performance problems that could arise later on.
- Automated testing is especially useful, as it allows for quick feedback and can catch issues before they become more serious. Unit tests, integration tests, and end-to-end tests can all be useful for ensuring that code is functioning as expected and that changes or updates do not introduce unexpected bugs or issues.

Conclusion

After conducting the background research, the best front-end framework for Merkle to use with their backend, which is connected to Salesforce through Node.js, would be React.js. This is because React.js has a modular, component-based architecture that makes it easy to maintain and update code over time. Furthermore, it seems that developers are used to TypeScript and abstracted code.

In terms of coding practices and standards, Merkle should adhere to guidelines that ensure code is logically correct, complete, readable, secure, and adheres to company and project-specific coding conventions and standards. The code should have a well-organised folder/file/module structure, be unit-testable, have no dead or unreachable code, and comply with legal or other regulations.

Best practices I could use to improve front-end performance include avoiding content-heavy designs, embedding videos, and large files that can block the browser while downloading and parsing. It is recommended to use separate CSS and JS files for each page, and to load the initial view content first to improve user-perceived performance.

Sources

1. Code Maintainability Survey Merkle (Responses). (n.d.). Google Docs. https://docs.google.com/spreadsheets/d/1fjfjCR5z59fiUUgtbKCsYNG1_N1CiZgNuYk1RnqNWIs/edit?usp=drive_link

2. Begishev, T. (2023). Quick guide to integrate react app with salesforce CRM. Nubessom Consulting. https://www.nubessom.com/blog-article/quick-guide-to-integrate-react-app-with-salesforce-crm/

3. Coyier, C. (2021, September 1). Data URIs | CSS-Tricks. CSS-Tricks. https://css-tricks.com/data-uris

4. Google TechTalks. (2009, June 5). Speed Up Your JavaScript [Video]. YouTube. https://www.youtube.com/watch?v=mHtdZgou0qU

5. Chrome DevTools - Chrome Developers. (n.d.). Chrome Developers. https://developer.chrome.com/docs/devtools/

6. Bellairs, R. (2019). What Is Code Quality? Overview + How to Improve Code Quality. Perforce Software. https://www.perforce.com/blog/sca/what-code-quality-overview-how-improve-code-quality

7. Coding Standards For Quality and Compliance | Perforce Software. (n.d.). Perforce Software. https://www.perforce.com/resources/qac/coding-standards

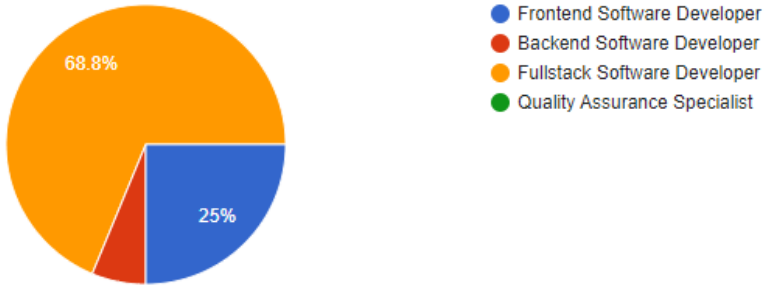
Appendix E:

Results from Survey in Appendix B

What is your current job title?

Copy

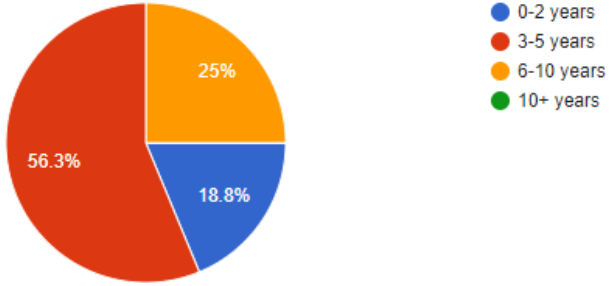
16 responses



How many of years of experience you have in e-commerce development?

Copy

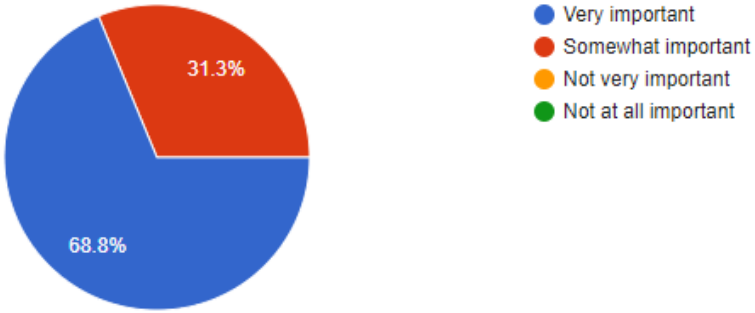
16 responses



How important do you think it is to design code for flexibility and maintainability in the long run?

Copy

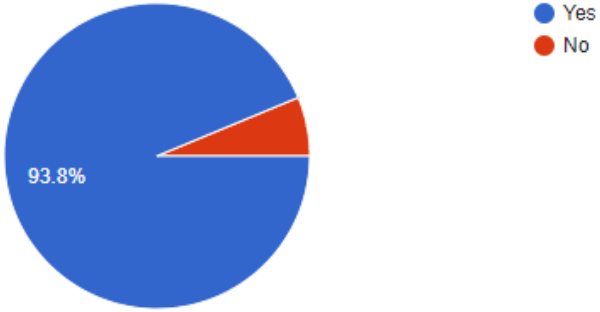
16 responses



Have you ever encountered issues with maintaining legacy code on an e-commerce website?

Copy

16 responses

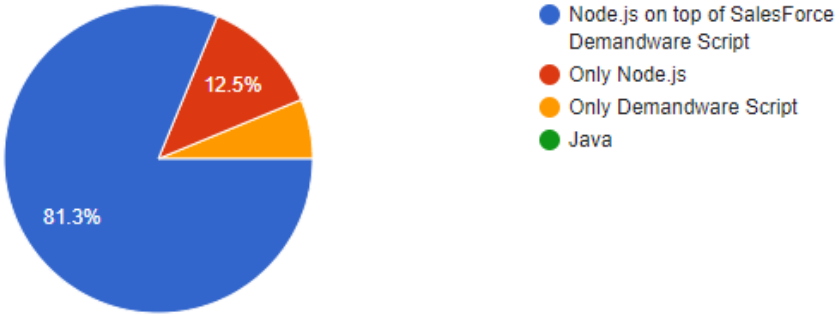


Project Specific Questions

What backend software is used in your project?

Copy

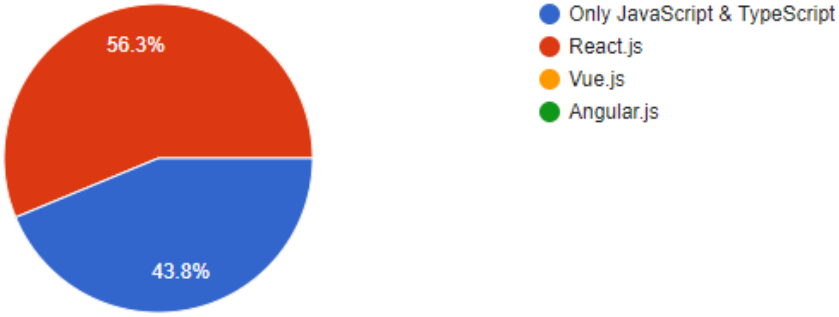
16 responses



What frontend software is used in your project?

Copy

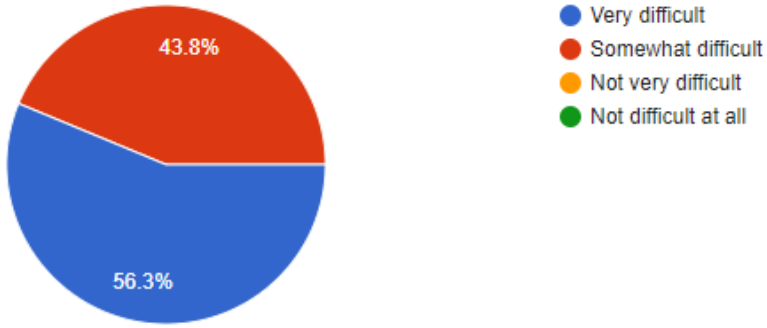
16 responses



How difficult you find writing new code when a new demand comes from the client?

Copy

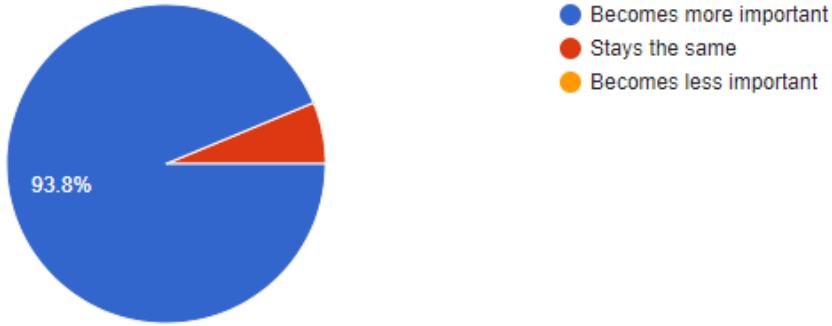
16 responses



In your experience, how important becomes the flexibility and maintainability as the size of a company's e-commerce website grows?

Copy

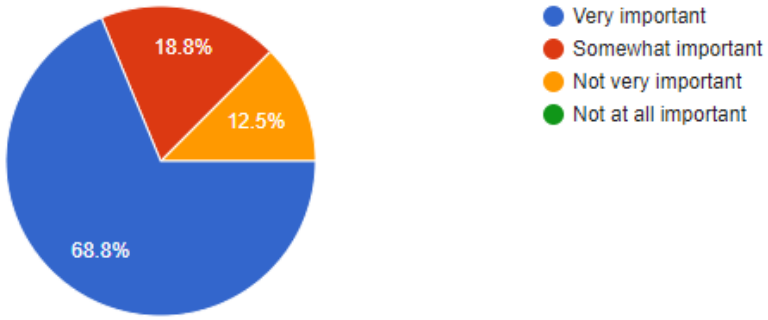
16 responses



How important is testing and ensuring code is maintanble?

Copy

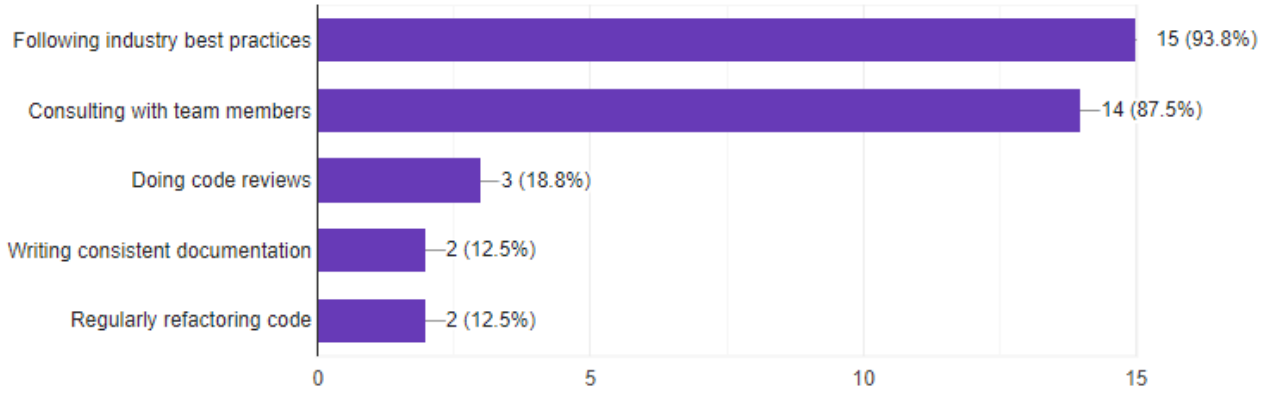
16 responses



What approaches do you typically use to write code for flexibility and maintainability?

Copy

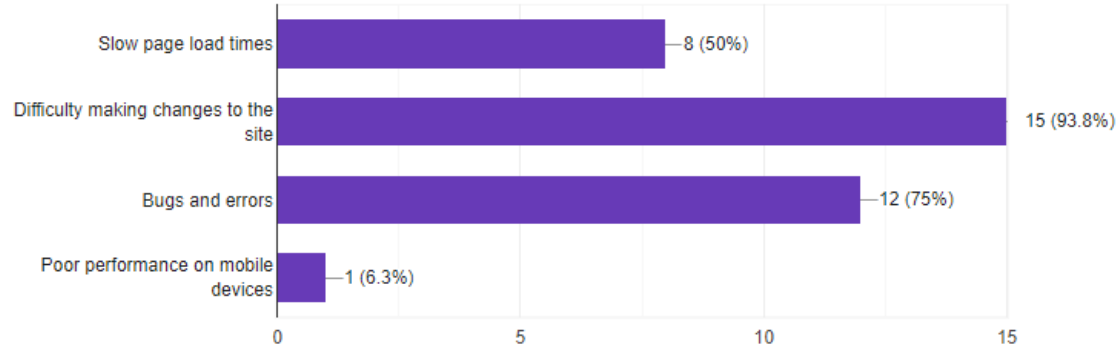
16 responses



What are some common issues you've encountered in maintaining an e-commerce website?

Copy

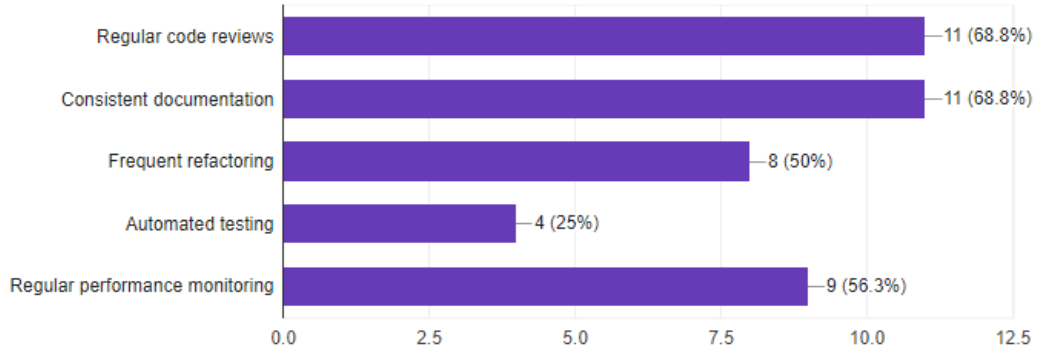
16 responses



What strategies have you found most effective for maintaining code on an e-commerce website?

Copy

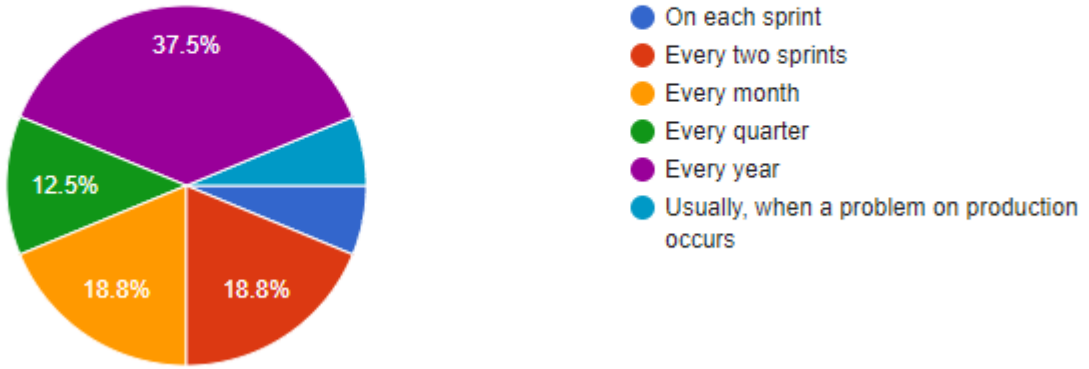
16 responses



How often do you refactor code to ensure its flexibility and maintainability?

Copy

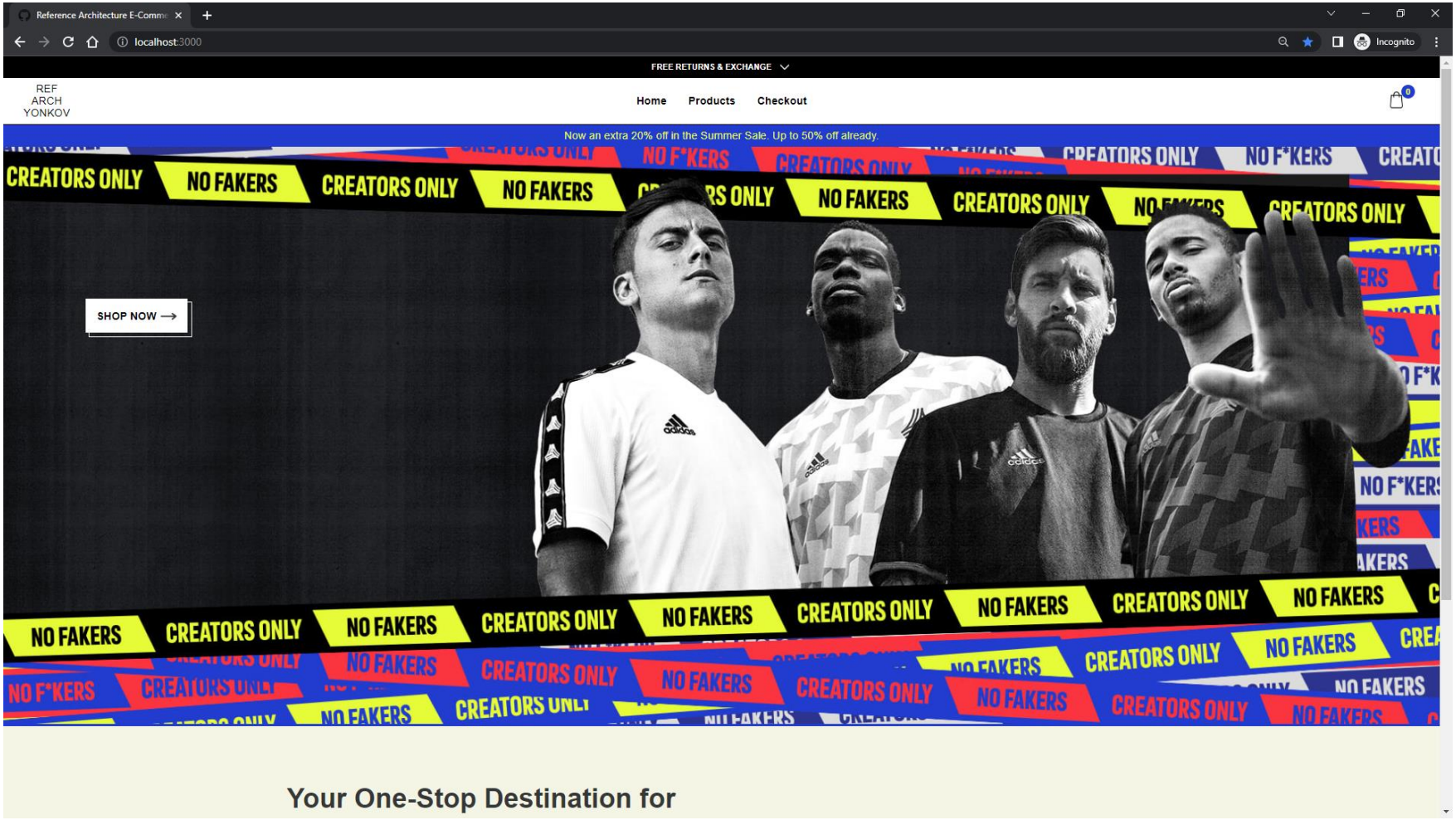
16 responses



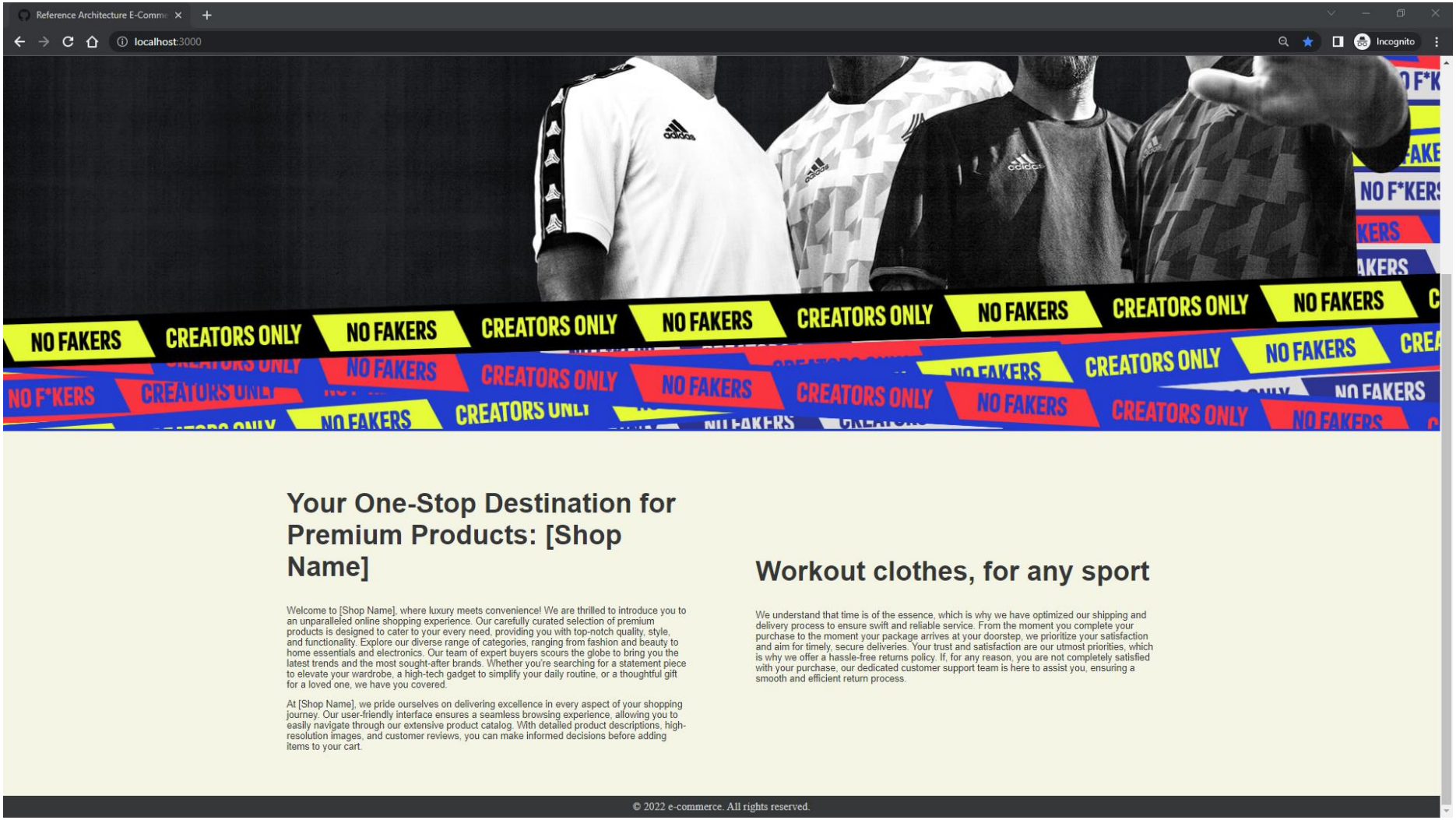
Appendix F Screenshots of website final version

Remark: The placeholder for a logo is temporarily the name of the project and the name of the developer

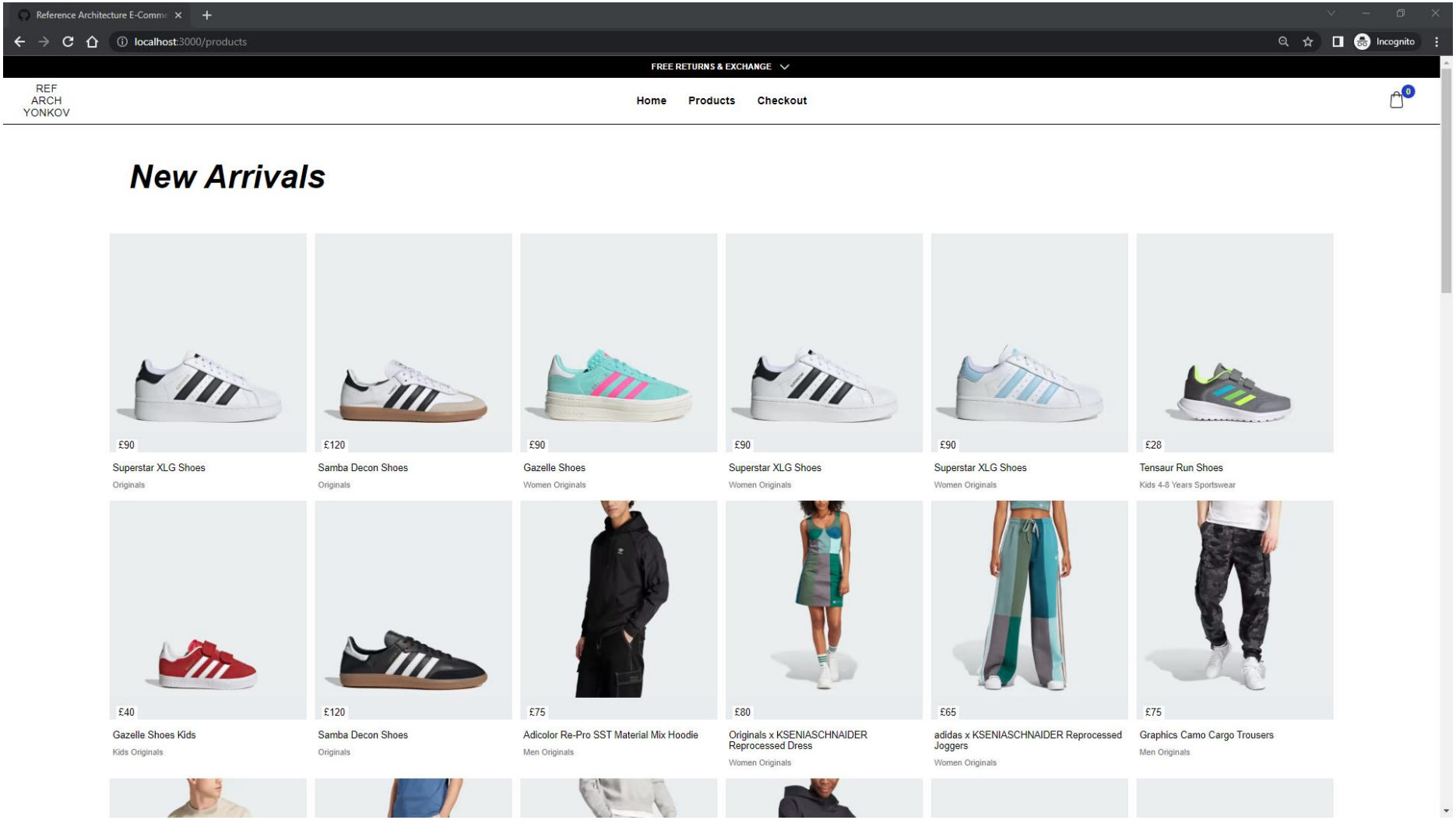
Screenshot 1 (Landing Page)



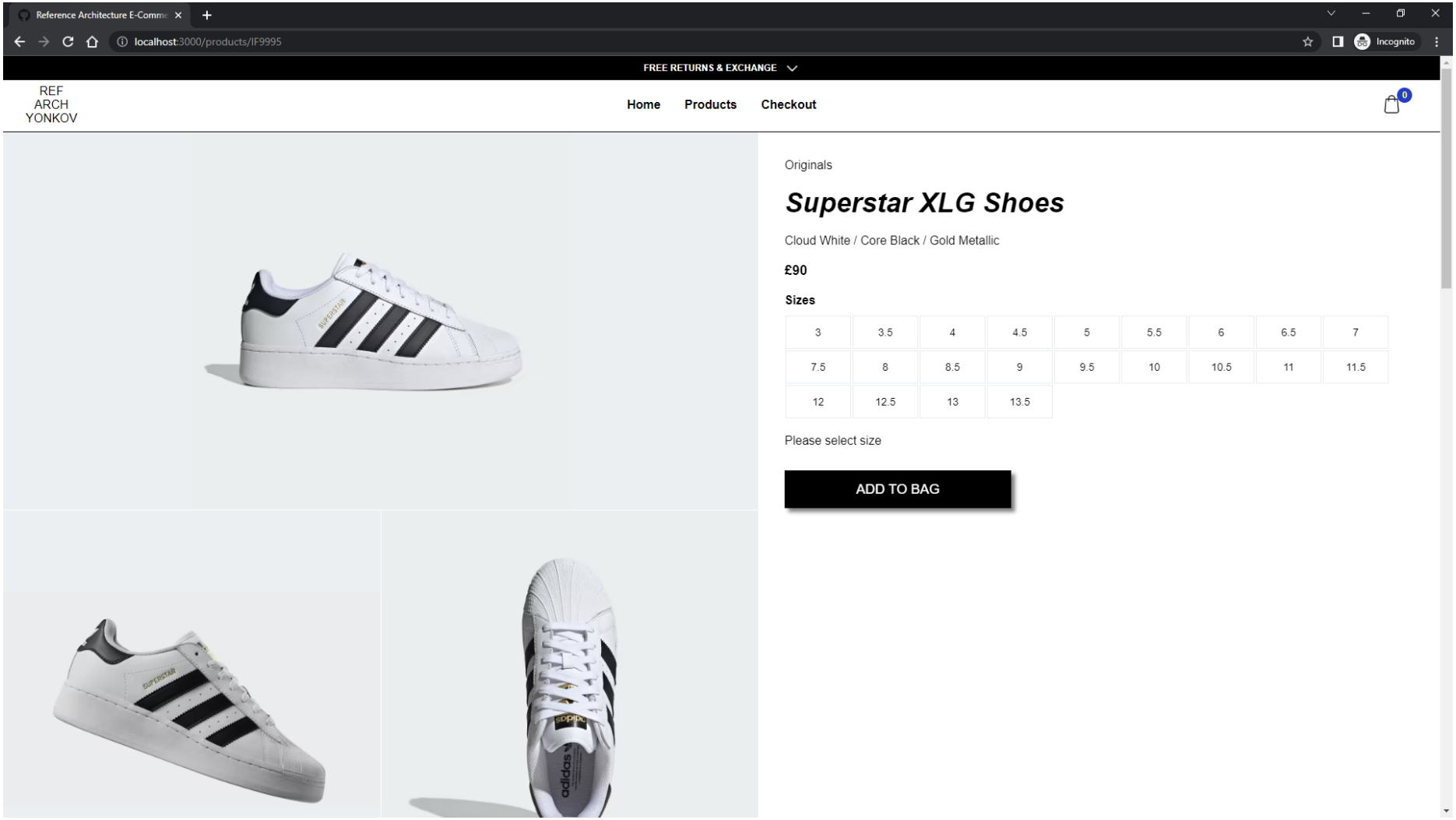
Screenshot 2 (Landing Page Scrolled)



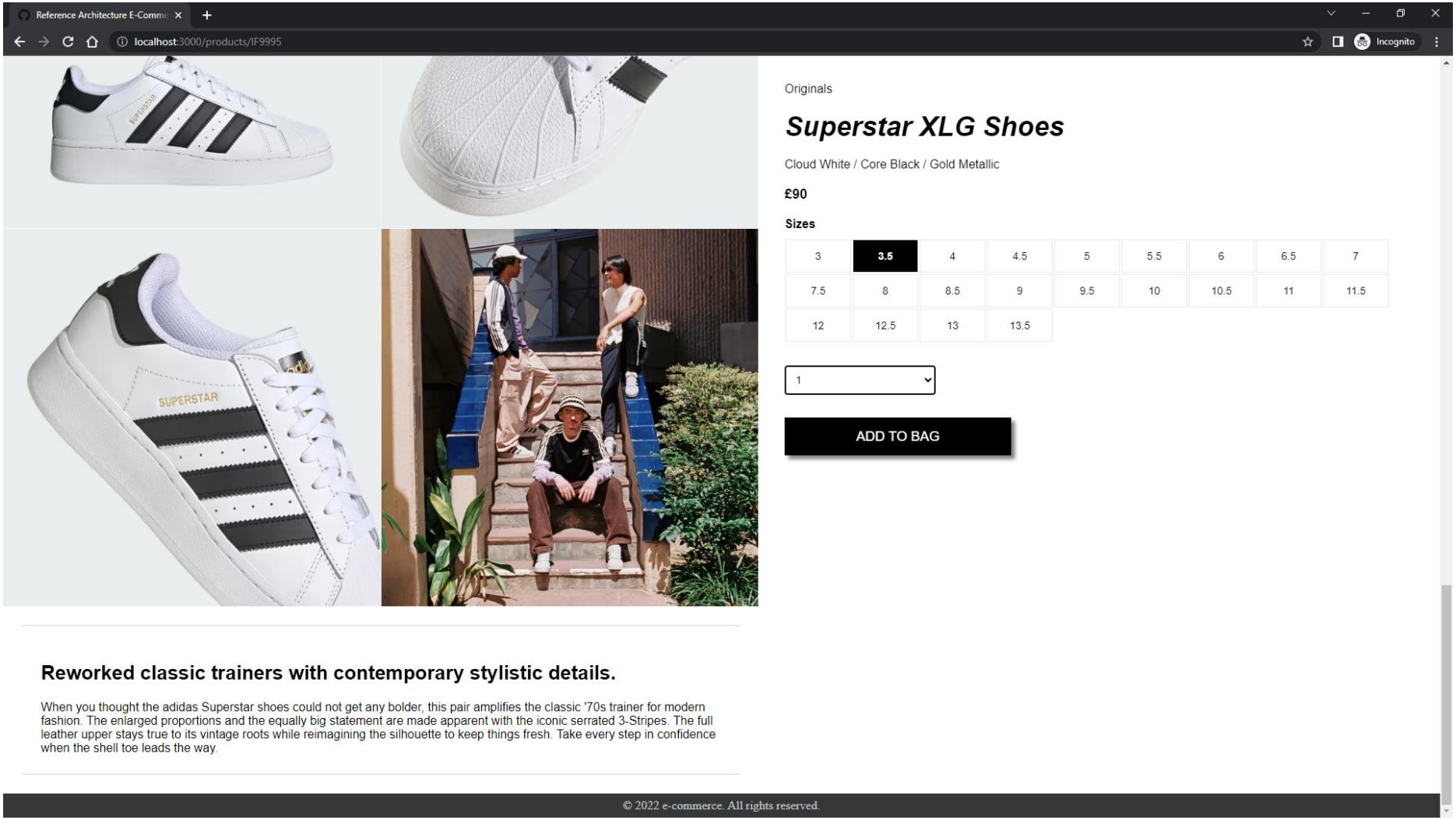
Screenshot 3 (Product Listing Page)



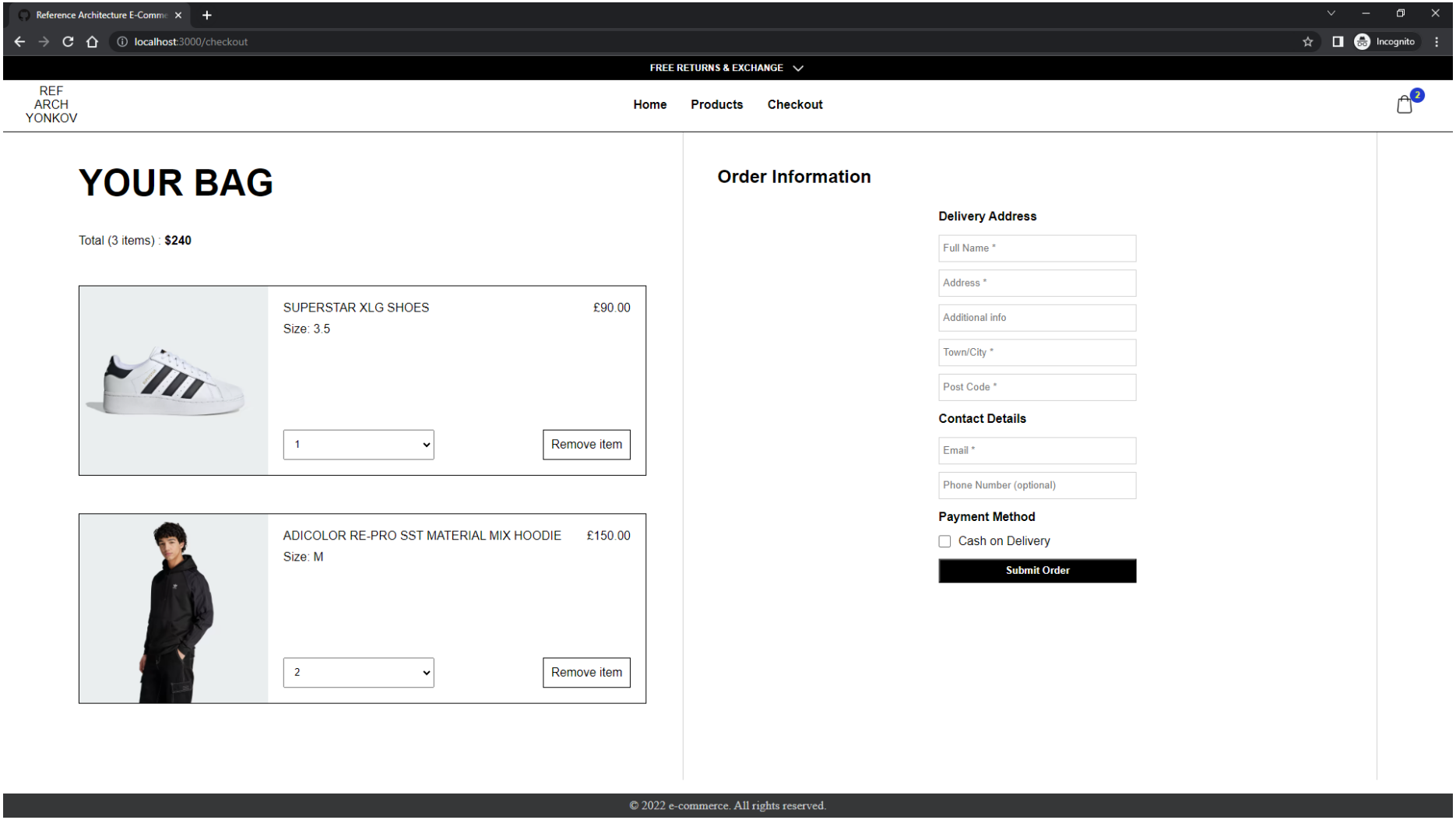
Screnshot 4 (Product Details Page)



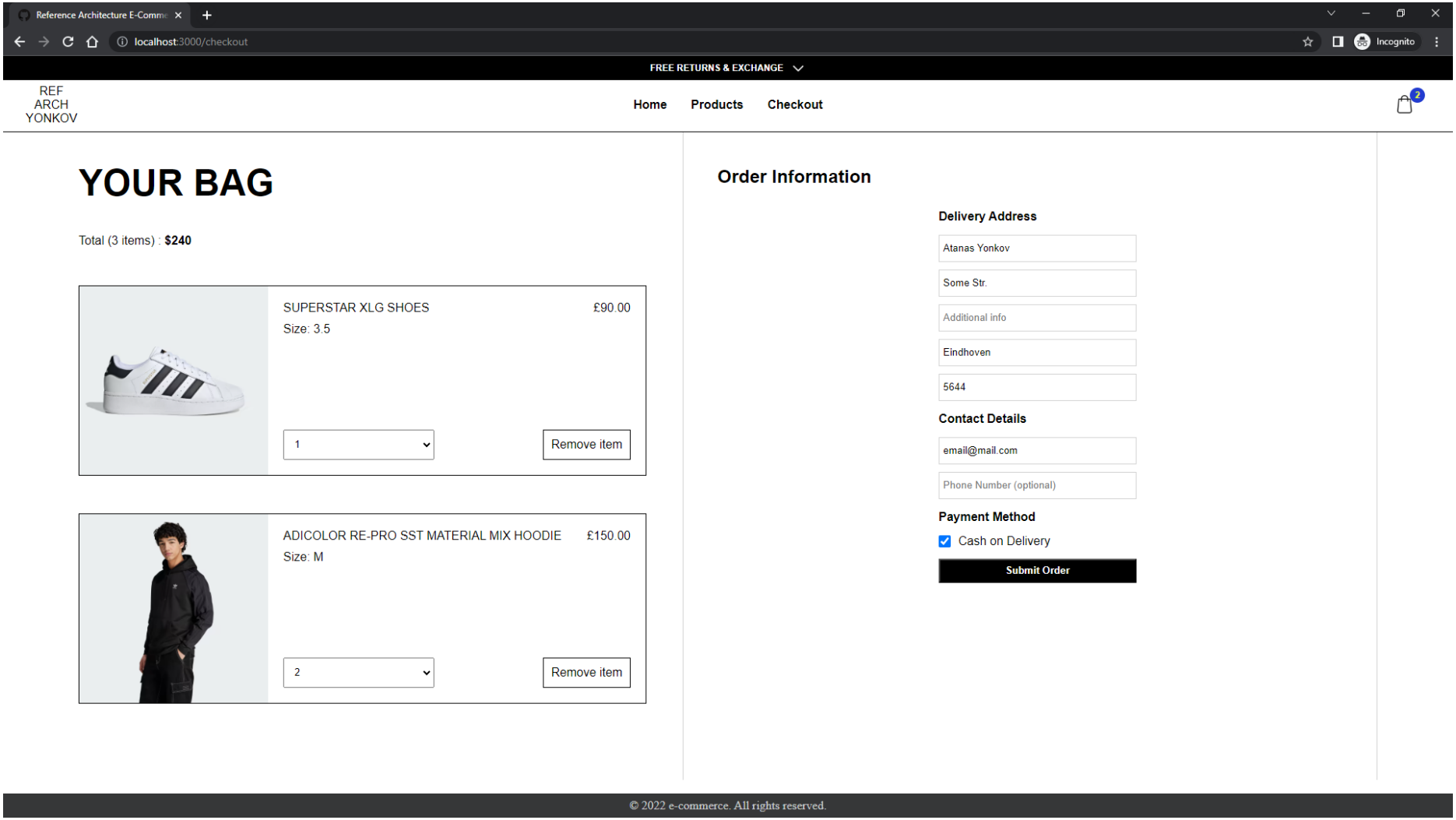
Screenshot 5 (Product Details Page with selected size and product added to bag)



Screenshot 5 (Checkout Page)



Screenshot 6 (Checkout page two products)



Screenshot 7 (Order submitted)