
# Development of a Scalable E-Commerce Website for Small Businesses

A Functional Approach to Usable, Secure, and Scalable Online Retail Solutions

Bachelor's Thesis
Degree Programme in Computer Applications
Spring 2025
Harsh Chahal

HAMK
Häme University
of Applied Sciences

Abstract

**DP** Degree Programme in Computer Applications
**Author** Harsh Chahal **Year** 2025
**Subject** Development of a Scalable E-Commerce Website for Small Businesses
**Supervisors** Elina Kulmala

---

The digital resurgence in the retail industry has accelerated in recent years; however, many small and medium-sized enterprises (SMEs) still face significant challenges in adopting scalable and affordable e-commerce platforms. SMEs lacking dedicated IT resources or cost flexibility often find commercial systems prohibitively expensive, overly complex, or insufficiently adaptable. This thesis aimed to design and implement a lightweight, secure, and user-friendly e-commerce infrastructure that small businesses can adopt with minimal technical overhead.

Combining theoretical research with practical implementation, the project followed a functional thesis approach. Theoretical foundations were drawn from literature on user experience design, secure system architecture, and digital commerce trends. Agile methodology enabled iterative development across structured sprints. The technology stack included HTML, CSS, JavaScript, PHP, MySQL, and Stripe for secure payment processing. All stages were informed by GDPR compliance and usability best practices.

The final prototype includes features such as user registration, product browsing, cart management, secure checkout, and an admin dashboard for inventory control. Stripe's API was used for tokenized payment processing, ensuring PCI-DSS compliance and data privacy. Performance testing with Google Lighthouse and Apache Bench indicated strong scalability and responsiveness. Usability testing with 10 participants yielded a 93% task success rate and an average satisfaction score of 4.4 out of 5. No personal data was collected or stored, and all ethical protocols related to privacy and transparency were followed.

This project demonstrates how open-source technologies can be used to build professional-grade e-commerce systems tailored to the needs of small businesses. While the system fulfills core business functions effectively, future improvements should address simplified installation and long-term maintenance to support adoption by non-technical users. The solution also lays the groundwork for additional features such as multilingual support, analytics tools, and automated customer engagement systems.

Keywords E-commerce, SMEs, HTML, CSS, JavaScript, PHP, UI/UX Design, Agile Development.
Pages 36 pages and appendices 11 pages

## Glossary

**HTML** – Hypertext Markup Language

**CSS** – Cascading Style Sheets

**JS** – JavaScript

**PHP** – Hypertext Preprocessor

**MySQL** – Structured Query Language-based Relational Database Management System

**CMS** – Content Management System

**GDPR** – General Data Protection Regulation

**UX** – User Experience

**UI** – User Interface

**SSL** – Secure Sockets Layer

**Stripe** – Online Payment Processing Platform

**Hosting** – Web Server Provision for Application Deployment

**HTTPS** – Hypertext Transfer Protocol Secure

**Responsive Design** – Web Design Approach Optimized for Multiple Devices

**Agile** – Iterative and Incremental Software Development Methodology

# Table of Contents

## Figures



## Tables



## Appendices

# 1 Introduction

Small and medium-sized enterprises (SMEs) must have internet presence as the digital presence has escalated. The COVID-19 pandemic also triggered this change and most traditional businesses were left with no choice but to use the digital sales platform to stay afloat. While these new opportunities existed, so did technological and financing barriers that prevented many of the SMEs from even being able to enter e-commerce or to survive if they did begin (Gao et al., 2023). As a student in computer applications and an entrepreneur at heart, this fostered a personal and professional motivation for this thesis.

For this thesis, an e-commerce platform for small businesses was designed that is lightweight, secure, and scalable. The initiative is therefore taken from observation of how easy it is for SMEs to engage in using commercial content management systems (CMSs) such as Shopify, Magento, and Woo Commerce. These platforms have cool features, however, their recurring fees, rigid structures, and learning curves are onerous, and even for overburdened tech-supported businesses, it will be hard for them to afford them. But in many small businesses, they become dependent on external service providers who strip away control of the operations, and a business' growth. But this thesis aims at presenting this usable alternative, an open-source server to keep an acceptable level of functionality while the cost will be very low.

It is self-commissioned, and the project itself shows an actual real work life scenario. It offers a live retail shop setup simulation to model typical processes in the application including product listing, account management for customers, cart handling, and payment. It is written such that it is industry-agnostic enough yet specific enough to give you a decent e-commerce experience out of the box. The dual approach guarantees that it is both practical and educative. The development is underpinned by widely adopted technologies; HTML, CSS, JavaScript, PHP, and MySQL, chosen for their reliability and community support. The payment functionality is implemented using Stripe, a globally recognized API for secure online transactions, selected for its ease of integration, low cost, and compliance with data protection regulations. The platform is hosted on a virtual server through Digital Ocean, allowing for real-world deployment and performance testing. This technical stack ensures that the prototype is both accessible for novice developers and powerful enough for actual business use.

Academically, this project engages with research on digital entrepreneurship, user experience design, and secure web development. Although much literature discusses e-commerce platforms and their economic impact, there is limited focus on lean, purpose-built systems for SMEs. This thesis contributes to that niche by offering a functional, documented solution informed by best practices in both software engineering and business usability.

To guide development and ensure a structured exploration of the subject, this thesis is framed around three research question:

- What features must an SME e-commerce platform support to enable growth?
- How can user experience be optimized to increase conversions?
- How can secure payments be implemented affordably?

The thesis adopts a functional approach using agile methodology, including iterative testing and feedback loops. A developer diary is maintained throughout the project to document technical decisions, challenges, and user feedback. The resulting prototype, supported by academic literature and usability testing, will serve as a replicable model for SME digital transition.

# 2 Literature Insights

This project is designed to bridge the gap between technical capability and business need in the SME sector. By delivering a practical, open-source e-commerce system, the thesis effectively demonstrates how small businesses can take ownership of their digital strategies without excessive costs or dependencies. As emphasized by Sánchez-Torres et al. (2021), tailored digital tools are essential in helping SMEs transition to e-commerce efficiently.

## 2.1 Small Business and E-commerce Technology

Over the past several years, there has been a definite push from the global trends towards digitalization of the small and medium sized enterprises (SMEs), owing to the changes in consumer behaviour and the competitive digital front. This transformation creates a high priority requirement on SMEs to utilise technology or risk being left behind. As observed by Omowole et al. (2024), SMEs face both opportunities and challenges in their digital evolution, especially concerning e-commerce. This section theoretically explores how the e-commerce

has enabled small businesses to adopt e-commerce, the opportunities it has created and the barriers that still exist.

### 2.1.1 Background of E-commerce Adoption

The digitalization of commerce has become an unavoidable reality for small and medium-sized enterprises (SMEs) in the 21st century. As customer behaviour continues to shift toward online platforms, businesses without a digital presence risk obsolescence. According to Stand Earth (2022), global e-commerce retail sales accounted for nearly 20% of total retail transactions in 2021, a number projected to increase steadily in coming years.

This growth, driven in large part by mobile accessibility, changing consumer expectations, and global crises such as the COVID-19 pandemic, has made e-commerce not just an option but a necessity for SMEs (Gao et al., 2023).

While digitalization presents considerable opportunities, such as global reach, 24/7 accessibility, and lower operational overhead, it also introduces several challenges for smaller businesses. Many SMEs lack the technical expertise, financial resources, or digital strategy to compete with more established online retailers (Omowole et al., 2024).

Digital tools offer SMEs the chance to reach global markets, operate around the clock, and lower operating costs. However, their effectiveness depends heavily on affordability, usability, and alignment with business needs. As Sánchez-Torres et al. (2021) point out, commercial platforms are often designed for general use and fail to accommodate the specific needs of small enterprises, necessitating lightweight, tailored solutions.

### 2.1.2 Barriers to Technology Adoption

Although increased awareness of e-commerce benefits has increased, many SMEs have been reluctant or unable to opt for the fullest use of e-commerce technologies. Cost is one of the largest stumbling blocks. As Sánchez-Torres et al. (2021) explain, most commercial CMSs like Shopify or Wix involve recurring fees and require developer support for customization, placing strain on thin SME budgets. This suggests that such recurring costs may dissuade smaller businesses with tight budgets from transitioning online — a concern also noticed during the development of this thesis.

Limited technical expertise among SME owners is another major challenge, as observed during the planning and development stages of this project. While CMS platforms often market themselves as “user-friendly,” their complexity becomes evident once businesses attempt to perform tasks such as theme customization, payment integration, or SEO configuration. This complexity became clear in peer testing and platform comparisons during the project. Due to a lack of training, many SME owners are unable to train to utilize the platform how they feel is necessary, and then, not having the means to hire professionals, it often goes to waste leading to frustration (Choshin & Ghaffari, 2017).

Additionally, many commercial platforms include excessive features that are rarely used by small businesses, leading to slower performance and cognitive overload. These platforms are often built to accommodate a broad range of business types, which can result in a poor fit for SMEs with simpler needs. This mismatch contributes to cluttered interfaces, inefficient workflows, and unnecessary costs for smaller operators.

These challenges highlight a need for targeted solutions: platforms that provide essential functionality such as product listings, shopping cart integration, order management, and secure payment gateways, while eliminating unnecessary complexity. This thesis developed a platform that addresses all these barriers by providing an optimized e-commerce system that aligns with the operational and cognitive capacity of small businesses.

### 2.1.3 Relevance of SMEs in Digital Commerce

Most national economies' backbone is held by a big portion of employment generation, local commerce and innovation coming from small businesses. World Bank (2022) indicates that SMEs constitute more than 90% of all businesses and more than 50% of all employment. There is a higher occurrence of their presence in developing and transitioning economies (Faye and Goldblum, 2022). Though small businesses are economically significant, they struggle more than their peers when taking on digital technologies, especially in the realm of e-commerce (Omowole et al., 2024).

One major reason for the importance of SMEs in e-commerce development is the direct flow between SMEs and niche markets, and consumer needs in local markets. While small businesses are different from large retailers, small businesses generally provide individual market needs and deliver products or services which large retailers cannot, as they are not easy to reproduce at a mass market level. By doing this, they have their unique online

presence that gives them the ability to maintain identity, flexibility and customer intimacy at the same time while appearing in bigger markets as noted by Gao et al. (2023).

However, SMEs struggle to seize opportunities due to the lack of affordable and customizable digital infrastructure. Most e-commerce solutions offered today are either too costly (Shopify, Magento) or too generic, which makes them lacking in precision and simplicity for small-scale operations. This results in many SMEs being left offline (or heavily reliant on external service providers) and hence having little control and lower profits margins (Sánchez-Torres et al., 2021).

Taking all that information into account, this thesis project fills these gaps by providing a lightweight secure, scalable platform specifically for the small business user. This system takes a different approach from that of commercial CMSs that bundle unneeded functionality to provide something similar to the basics: listings of products, handling of the cart, secure checkout that still allows an affordable and independent business owner.

As Voigt and von dem Bussche (2017) argue, effective systems should be built around actual needs, guided by principles of usability, customization, and efficient use of resources. Additionally, giving small businesses the access to e-commerce tools makes digital equity. It gives the chance to smaller retailers, artisanal businesses and family operated firms to compete in the same online marketplace with larger corporations. By bridging technical and socioeconomic challenges, this project offers small businesses a means to sustain themselves digitally and strengthen their resilience in an increasingly industrialized and digital world.

## 2.2 Technologies Used

The core foundation of the super backend requires you to have selected the optimal frontend and backend technologies. Backend tools do business logic management and data flow; frontend tools value shapes for their user’s experience. As emphasized by Duckett (2014), open-source, accessible, and scalable tools can save costs for small businesses more than they pay for the not so small, ticketed products. This section formalizes the core technologies used to build the e-commerce prototype in the thesis, and which comprise the UI and the server side in terms of HTML, CSS, JavaScript, PHP, and MySQL.

### 2.2.1 Frontend Technologies: HTML, CSS, JavaScript

The front end of an e-commerce website is vital in shaping the user experience and guiding the user onto your customer journey. Web development is based on a foundational triad of structures technologies that allow for design, interactivity, and structure. These are HTML, CSS, and JavaScript respectively. HTML (Hypertext Markup Language) gives the structural skeleton of a webpage by using the semantic degree of components such as headings, paragraphs, hyperlinks, and variables as explained by Duckett (2014). CSS (Cascading Style Sheets) is responsible for the visual styling of these elements, including layout, color schemes, typography, and responsive behavior.

Responsive design, which ensures websites display effectively across different devices and screen sizes, is especially important in e-commerce. As users increasingly access websites through smartphones and tablets, it is essential that e-commerce platforms adapt to various screen sizes to ensure a consistent and accessible user experience. Media queries, flexible grids, and viewport-relative units are among the CSS strategies employed to achieve responsiveness (Marcotte, 2010).

JavaScript brings interactivity to static pages. In an e-commerce context, JavaScript is essential for functions like dynamic cart updates, real-time price calculation, product filtering, and client-side validation during checkout. Duckett (2014) explains how libraries such as jQuery or frameworks like React can enhance interactivity, but even vanilla JavaScript is sufficient for building intuitive user flows. Based on industry best practices and observations during testing, a responsive frontend improves engagement and minimizes user frustration that may lead to cart abandonment. The prototype developed in this thesis uses HTML and CSS for layout and design, supported by JavaScript to manage front-end logic such as cart behavior and product selection. The code-base emphasizes simplicity and performance to ensure a smooth experience across both desktop and mobile devices.

### 2.2.2 Backend Technologies: PHP and MySQL

While the frontend manages user interaction, the backend governs the logic, data, and processes behind the scenes. PHP, a server-side scripting language, is widely used in dynamic web applications for processing user input, managing sessions, and communicating with databases. It remains one of the most accessible languages for web development due to its low barrier to entry, robust documentation, and widespread hosting support (Welling & Thomson, 2008).

MySQL complements PHP as a reliable relational database management system (RDBMS), storing and retrieving structured data such as product listings, customer details, and order records. Queries using SQL (Structured Query Language) allow the application to perform Create, Read, Update, and Delete (CRUD) operations efficiently (MySQL, 2024). By using normalized database tables, data redundancy is minimized, improving performance and scalability.

Authentication and session management are completely backend-wise and crucial in e-commerce. While PHP does not offer a built-in authentication framework, it enables the development of secure login systems using bcrypt-based password hashing and session variables to manage logged-in user information. With MySQL, these features are logically combined to guarantee the protection and accessibility of user data. In this thesis, the backend implementation includes user registration, login, product and order management and data safety (MySQL, 2024). Dynamic content is managed PHP scripts which communicate with a normalised MySQL schema with data integrity. This lightweight solution is ideal for small businesses and offers strong reliability.

## 2.3 Secure and Scalable Payment Integration

In digital commerce, trust and security are non-negotiable priorities. As the need to protect user data grows, small businesses must adopt secure and affordable payment systems (Hassan et al., 2020). Stripe, with its tokenized model and PCI-DSS compliance, stands out as an ideal solution. Secure payment integration is examined from both technical and legal perspectives. The following sections discuss how the selected architecture incorporates encryption, user experience considerations, and regulatory compliance such as GDPR.

### 2.3.1 Stripe API and Encryption

Payment security is a critical aspect of any e-commerce system. Consumers must be assured that their financial data is handled with the highest security standards. When they connect their user's browser to a server, payment information should be secure enough and interception proof from unauthorised parties. As noted by the PCI Security Standards Council (2018), TLS encrypts the communication. For this reason, the prototype payment integration leverages Stripe, a popular payment service provider which has a strong reputation among developers for its API and especially for its security reputation.

Stripe implements encryption with TLS and is PCI DSS compliant. These are necessary protocols to ensure that data in transit is sensitive and industry standard compliance (PCI Security Standards Council, 2018). TLS encrypts the communication between the user's browser and the server, preventing unauthorized parties from intercepting payment information. However, the PCI-DSS framework details the way in which payment data must be collected, stored and transmitted in a secure environment.

According to Stripe (2024), its biggest advantage is the tokenization process involved. Instead of sending raw card data to the server, Stripe's API will create a one-time card token which will represent the card data. After that, the server receives this token, minimizing the risk of data breaches. Thereby the merchant's system never directly deals with or stores card number resulting in minimal legal as well ass technical liability. For this thesis prototype, the Stripe integration is using client side Elements (customizable UI components) for the capture of payment details. The result token is subsequently handled by the backend PHP scripts, which then finalizes the transaction (Stripe, 2024). This approach ensures a seamless user experience while adhering to global payment security standards. Furthermore, all pages are enforced to communicate over HTTPS using a Let's Encrypt SSL Certificate to enforce encrypted communication throughout as well as during checkout (Let's Encrypt, 2019). It is essential to ensure that consumers feel confident about how their financial data is protected.

### 2.3.2 GDPR and Privacy

Additionally, the necessity of data privacy for user support has also become a legal, and moral issue. If a website is to be used by Europeans, the European Union's General Data Protection Regulation (GDPR)'s strict rules are ought to be followed to collect, process, and store personal information. Although the prototype does not store personal data such as identification numbers or addresses, it aligns with the core principles of the General Data Protection Regulation (GDPR). As outlined by Voigt and von dem Bussche (2017), GDPR compliance includes principles such as data minimization, purpose limitation, and lawful processing.

Data minimisation requires the collection of only the data necessary to perform a specific function, nothing more. In the case of a platform like this e-commerce, this means getting only those things that are essential typical of email addresses; and things like login credentials that are hashed or in an encrypted state before storing. Secure PHP session variables are used to handle the session, and these session variables are deleted for logout or session timeout.

Similarly, the use of cookies has been optimized by authenticating used session cookies but without third-party tracking or persistent cookies which decreases profiling risk. Although this simple implementation does not have a modifiable cookie consent banner included, it is designed in a modular fashion to include future deployments.

This platform demonstrates how online payment functionality suitable for SMEs can be implemented by prioritising encryption, stored data minimization, and respect for established frameworks of compliance (Voigt & von dem Bussche, 2017).

## 2.4 UX/UI and Accessibility in E-commerce

In online store, User Experience (UX) and Interface Design (UI) directly influence the store's success and the impact turns out to every aspect, from engagement to conversion. This also implies that platforms can target a larger audience, i.e., those with disabilities. And smart UX/UI and inclusive design can help SMEs win in the market without having to spend great amounts of money. As emphasized in the study by Silva da Silva et al. (2012), well-designed and inclusive UX/UI principles can help SMEs compete effectively without large financial investments.

In this section, the thesis project is explained, and design principles, mobile responsiveness, and accessibility standards are explained throughout the project so that the user can have a good user experience, especially in e-commerce.

### 2.4.1 UX Models for Online Shopping

User experience (UX) design plays a decisive role in determining whether an e-commerce platform succeeds or fails. A well-designed interface can significantly increase conversion rates by guiding users through the purchase journey efficiently and enjoyably according to Mrs. K. N. (2023). Furthermore, Krug (2014) argues that, the essence of good UX lies in simplicity and clarity users should never have to guess what to do next. In an online store, this means maintaining a clear information hierarchy, prominent call-to-action buttons, and intuitive navigation.

As noted by Hjorthall (2025), excessive user interactions, such as unnecessary clicks can create friction and increase cart abandonment. In this prototype, pages should flow logically, with the cart and checkout processes requiring minimal input and decisions. The product

flows from selection to payment without interruptions because the prototype implements these principles to reduce user doubt while boosting their trust in the process.

Furthermore essential is visual feedback. Users should receive immediate feedback from the UI during both cart removal and addition operations. System reliability assurance comes through confirmation messages together with loading indications and tiny animations that demonstrate system operation (Hjorthall, 2025).

The micro-interactions developed through JavaScript application to the front-end system enhance the satisfaction of user interactions within the prototype. The foundation of successful user-friendly page navigation consists of maintaining uniformity. Headers and footers together with navigational links maintain user orientation, especially for first-time system visitors. The application maintains consistency through CSS guidelines together with a unified layout grid structure in all system views (Duckett, 2014). .

### 2.4.2 Mobile Optimization and Inclusive Design

Business operations of E-commerce depend on mobile platforms to exceed 50% of their overall functions. Every digital platform must implement responsive design completely. According to Zeng, Gao, and Wu (2014), responsive design ensures adaptability across screen sizes using CSS grids and media queries; key techniques also adopted in this prototype. Accessibility is both a legal requirement and a moral imperative.

As outlined by W3C (2024), the Web Content Accessibility Guidelines (WCAG 2.1) require websites to offer content that is perceivable, operable, understandable, and robust to all users, including those with disabilities. This prototype adheres to these standards by ensuring adequate colour contrast, scalable fonts, and logical keyboard navigation.

The website adopts ARIA (Accessible Rich Internet Applications) roles and attributes to develop its navigation bars and buttons together with form fields. Screen readers identify site structure and component functions because ARIA roles and attributes are implemented across website elements to make the website accessible for visually impaired users (MDN-Mozilla, 2025). Both visible messages and ARIA live regions provide instant feedback during the form validation process. The design approach establishes specific measures that create web pages suitable for users with basic technology and users on slow internet connections. The approach enables users to reach several sustainability targets that create responsible digital access systems usable by everyone (Voigt & von dem Bussche, 2017).

# 3 Methods

This chapter explains how the e-commerce platform was developed. The research takes a practical approach, focusing on building and testing a working solution using web technologies suitable for small businesses. It describes the overall plan, the reasons behind choosing certain tools, how the development was done, and how usability testing was carried out. The chapter also covers how user data and privacy were handled responsibly.

## 3.1 Thesis Approach

This thesis is a practice project that essence to create and prove some working e-commerce web design for small businesses. It is an approach based on applied research for the purpose of solving a problem in the real world using technological solutions. Instead of conducting the test on a theoretical hypothesis, the thesis deals with practical implementation challenges and resolves them with the help of software development. It contributes academically to this development process in the reflection, documentation, and evaluation of this process. The agile methodology was used to structure the project and set up a Sprint model (lean Scrum variant). Thus, as the development is the focus of actual research, Agile suits this methodology well, bringing adaptability, continuous iteration, and user feedback to this procedure (Abrahamsson et al., 2010). It enables a developer to build features one by one, test and evaluate the outcomes, and improve continuously by learning how to work with new features online.

Every development sprint had an area of focus which could be along the lines of user registration, shopping cart logic, payment gateway integration, etc. A review of the results, documentation of that as well as a functional and usability testing of the work was done at the end of each sprint. Feedback from peers and test users helped guide improvements and determine the next development steps. The problem introduces theory and practice in a meaningful way, as noted by Silva da Silva et al. (2012). The final system was formed to meet real-world limitations (i.e., deployment constraints and API behaviours) and the design decisions were dictated by academic sources. Notably, critical analysis of the thesis is not difficult to articulate within the structure of research objectives and the functional nature of the development journey resulting in the thesis outcome.

## 3.2 Technology and Tool Selection

This thesis is a practical project aimed at designing and developing a functional e-commerce website tailored for small businesses. It involved both frontend and backend technologies to ensure a complete and self-contained application.

The front-end technologies employed were HTML, CSS, and JavaScript as core technologies. The structural foundation on which the web pages live come from HTML, the visual presence comes from CSS, and those dynamic interactions across the web pages are handled by Javascript (dynamic cart updates, validation on form submission, etc.). These technologies are widely used and will work on any browser and any device and are perfect for wide user access (Duckett, 2014). The backend was built with PHP which is one of the most popular server-side scripting languages for web development. Because it was integrated with the relational database management system, MySQL, it is able to securely store and retrieve data like user profiles, product listings, and order histories (Welling & Thomson, 2008). This pairing remains popular in small business settings due to its low cost, reliability, and extensive documentation.

The project integrated Stripe, a modern API-based payment gateway with an excellent developer interface as well as great security features for payment processing. The combination of Stripe’s good documentation, its own sandbox testing environment, and it being compliant with PCI-DSS made it a suitable choice for securely processing customer transactions without keeping the sensitive payment data (PCI Security Standards Council, 2018). GitHub was used to control the version of the entire project, and the change systematic tracking together with collaborative testing was possible. The feature development was isolated into git branches, the commits were chronological logs for sprint reviews. It was deployed through Digital Ocean, a cloud hosting with a Digital Ocean Droplet that fits your requirements at a low price. Apache is a widely used web server that provides PHP and MySQL integration, hence the server is configured with Apache. Collectively, these tools and technologies created modular, scalable, and maintainable processes that worked in harmony with the thesis' technological needs as well as the real-world constraints usually experienced by small businesses entering the e-commerce space.

## 3.3 Development Process

The e-commerce platform was built with agile principles in mind and in a cycle and iteration way. It improved continuous reflection and continuous refinement of the project. The four core phases are wire framing, coding, testing, and improving.

Wireframes were initially created to outline the key user interfaces, including product listing pages, shopping carts, checkout flows, and user profile dashboards. The wireframes were intentionally kept simple and were revised based on early usability feedback. These wireframes served as blueprints for both the frontend layout and backend data structure.

The coding phase began with foundational features developed using HTML and CSS, which were progressively expanded and integrated with PHP and MySQL to create dynamic content like product catalogues and user sessions. The ability to enable things such as updating cart totals and form validation made JavaScript necessary. This was developed via sprints where each sprint was used to deliver a feature or features. Version control was done on GitHub and both commits and tags were aligned with feature milestones and tags with sprint.

Each sprint would be tested continuously, and finally reviewed. Interaction with the system is then performed by peer testers as they go through predefined scenarios, i.e. browse items, register with accounts, and purchase items. In subsequent sprints, bug reports, usability feedback, and observations were collected to guide it in the improvements.

There was an existing project that maintained a Markdown-based sprint diary for documenting the decisions taken, the issues that came up during the project, and the lessons learned. Appendix 4 includes this diary, as a transparent record of the development process. Additionally, it captures design decisions and trade-offs traded off as they were made. And this structured reflective process made sure that the application would grow with technical best practices and match user needs, even with a thesis' functional and academic goals in sight.

## 3.4 Usability Testing

Usability testing is the most fundamental stage to verify the functionality and users' experience of an interactive system. In this thesis, a peer review approach was adopted

involving people with varying degrees of technical knowledge who represent the potential user base of the platform - it is people such as small business owners, students and early career developers. The diversity among the feedback makers made it realistic and representative of potential end users with whom the platform would be used. The testing process was done with scenarios such as adding a product to the cart, registering an account, navigating checkout, logging out, etc. These tasks cover the main use cases of the e-commerce system as well as test each of the major components of it.

The performance was evaluated with two of the primary metrics for usability: time to completion and subjective satisfaction. The efficiency of each task was assessed based on the time spent to complete each task while the satisfaction of each task was captured in a short questionnaire using a 5-point Likert scale. Examples of the questions were “I found the system easy to navigate” and “The checkout process was clear and intuitive.” Finally, the participants also offered their thoughts about any challenges encountered and any changes that could be made.

Informal sessions were used to test and for each participant, the system was interacted with, on their own device to mimic real-world conditions. There was an observation by the author and debriefs. The usability log kept track of all the feedback, anonymizing it, and the contributions of this feedback were directly brought into the refinement of the product during the final development sprint. This agile integration of feedback ensured that design improvements were grounded in real user behaviour, aligning the prototype more closely with user expectations and commercial viability.

## 3.5 Ethical Data Handling

Ethical considerations in research and development are particularly important in projects involving user interactions, especially when systems collect or process any form of personal or identifiable information. For this thesis, ethical data handling was a central concern from the outset. A formal Data Management Plan (DMP) was created before the start of data collection and is included as Appendix 1.

No personally identifiable data, such as names, addresses, or payment card information, was collected or stored as part of this research. All testing interactions were anonymous and non-intrusive. Participants in usability testing engaged voluntarily and were informed in advance about the nature of the tasks and the use of their feedback. No sensitive information

was solicited, and consent was given orally and implicitly through participation. Since the Stripe payment gateway was integrated in test mode, all transactions were simulated using dummy credentials provided by Stripe's developer platform. This ensured that no real financial data was processed during testing.

All user feedback was anonymized and stored locally on the author’s password-protected device. Backup copies were made to an encrypted external drive to ensure data security in case of hardware failure. No cloud storage services were used to manage this data, in alignment with the guidance provided by HAMK’s ethical standards. Only the author and the thesis supervisor had access to the full testing logs.

Session data collected by the application was limited to non-persistent cookies used for login functionality. These sessions were automatically invalidated upon logout or timeout and were not used for tracking or analytics. The system was also reviewed to ensure GDPR compliance by default, adhering to principles such as data minimization, purpose limitation, and storage limitation (Voigt & von dem Bussche, 2017).

In cases where screenshots of the development process included any participant interaction, those images were reviewed to ensure that no identifying data was visible. Only sanitized or simulated content was used in presentation materials or appendices. The ethical framework established for this thesis ensured that the development and testing of the platform upheld both institutional and industry standards for privacy, consent, and secure data handling.

# 4 Practical Implementation

This chapter presents the practical implementation of the e-commerce platform developed for this thesis. It transitions from the theoretical background and methods to actual development work. The implementation covers the integration of frontend and backend technologies, database structure, secure payment systems, and deployment. Each section demonstrates how these components were built to support small business needs, with a focus on usability, security, and maintainability.

## 4.1 Frontend Development

The frontend of the e-commerce platform serves as the primary point of user interaction and must be structured with attention to responsiveness, clarity, and usability. The layout follows

a clean grid-based structure using UX principles to maintain consistency across a range of devices. It is designed as a mobile-first approach – content has been designed to work well from tiny smartphone screens onto larger desktops.

Home is the homepage with the structural hierarchy from top to bottom and in that order, with a Fixed navigation bar at the top and offering easy access to key pages: Home, Shop, Cart, Login, and Admin (if the person is authorized). What makes it is that this bar is visible during scrolling, so the user always has a stable orientation point. There is a reserved space below the header for promotional content or featured products, followed by displaying product cards in a grid.

The image, name, price, and 'Add to Cart' button are each contained in a product card. With this layout, users can quickly scan multiple products without unnecessary clicks and get details. This helps to use with same space, same font style, and same size button so it looks good, and helps with cognitive load and the browsing experience. Based on a neutral colour scheme with high contrast between the text and background, the interface should be more accessible, especially for people with visual impairments.

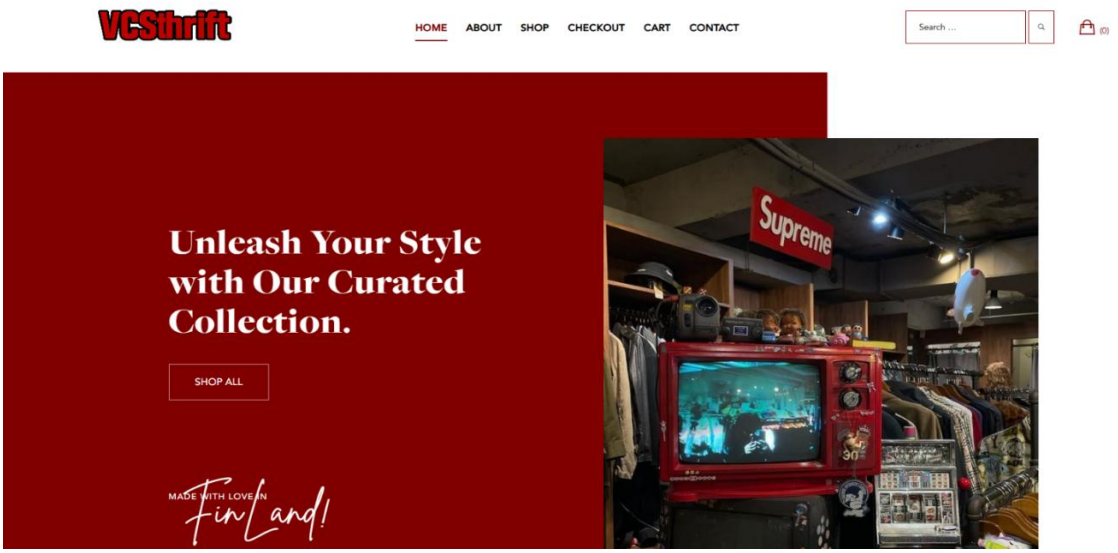
It achieves responsiveness with media queries and using CSS Flexbox / CSS Grid layouts. So, these techniques make it possible not to break or overlap elements if screen width changes, which is essential for the usability of mobile devices. Finally, the design was tested using both the Chrome Developer Tools and our real devices to make sure it is cross-platform.

In general, cart-related functions are handled by JavaScript, and interactivity is being done. On the click “Add to Cart”, taken product selected by the user will be stored temporarily in the session of the browser (by session Storage) and it will be dynamically updated in the cart icon on the top right of the screen. In combination, this ensures that users are always in the loop of their actions, something which is likely a crucial UX factor for reducing cart abandonment rates. These notifications show up ‘Item Added’ style after every action and help boost confidence just as you acted without interrupting the flow of browsing. JavaScript is also used to cheque input fields’ data integrity during login and registration form, to avoid sending poorly formatted input to the backend. This was deliberate choice to reduce dependency overhead and to improve performance by not using libraries such as jQuery.

Throughout making decisions in the process of designing, it made its best effort to maintain accessibility. In addition, all buttons have ARIA labels, all images have alt text that describes

them, and all form elements are keyboard navigable. These lines follow Web Content Accessibility Guidelines (WCAG) and support users who use assistive technologies and enable them to make smooth use of the system.

Figure 1: Homepage Layout is an illustration of the structure on the front end of the page that includes the navigation bar, banner, and product grid layout on the page. It follows core e-commerce usability and should have fast loading, visual clarity, and ease of interaction.
**Figure1: Homepage Layout**




The frontend codebase is a modular piece of code that can be extended easily. There are reusable HTML templates and CSS classes such as product cards, navigation menus, cart summaries, etc. The modular design means their existence can be overridden making it easier for future developers, or even small business owners with some technical understanding, to accommodate the design without having to reshuffle the whole system. Overall, the frontend was developed with a clear focus on usability, responsiveness, and clarity, aligning with both user experience (UX) principles and the practical expectations of modern e-commerce. The design functioned as the business and the customer's interface; therefore, it was created to be intuitive, professional, and accessible from the start.

## 4.2 Backend and Authentication

The e-commerce platform uses the backend which is responsible for data management, core business logic, and security protocols. The application's backend consists of using PHP as the server-side scripting language and MySQL as the database management system, and

as the functional core component provides the ability to render dynamic content, secure user authentication, and manage persistent data.

Some devices like users and session management are a primary function of the backend. It includes customer registration, customer login, customer logout, and administrative tasks access controls. On registering, the database is stored with secure user credentials. The use of bcrypt hashed accounts with salting and multiple iterations of it to prevent brute force attacks to verify password security. This mechanism is implemented using PHP password hash () and password verify () built-in functions, according to security recommendations by the OWASP Foundation.

Once the user logs in, PHP session variables are initialized to keep track that the user logged in across the website. Server-side session data is stored and associated with a unique session ID that is passed to the client by cookies. It enables personal features like a blink cart, placing an order, and go to the profile page. Logout or timeout causes the destruction of the session, so no unnecessary session data will stick after logout or timeout, which meets the criteria requirement of secure session handling.

The user roles were divided into two categories: customers and administrators. Login distinguishes the roles by the backend accessing the database to retrieve this user's access level. They also grant a further dashboard to admins, where admins can add, update, or delete product listings, see the order histories, and handle the inventory. Role-based conditional statements are enforced to separate the use of sensitive processing on the backend scripts, so accessing sensitive operations is prohibited and unauthorized actions from taking place.

SQL queries are prepared statement that handles data interaction. It minimises SQL injection vulnerability by binding variables to placeholders in SQL command, treating user input as data rather than executable code. User registration, login verification, product retrieval, storing in cart, and placing an order are the key database operation. Indexes on frequently accessed columns are used to optimise these queries to handle data in fast and efficient way.

All backend scripts are modular, with reusable functions organized in separate PHP files for maintainability and scalability. Administrative operations are secured behind authentication checks to prevent unauthorized data manipulation. By integrating robust session handling,

password encryption, and role-based access control, the backend ensures the confidentiality and integrity of user data while supporting the full range of e-commerce functionalities.

## 4.3 Database Design and Architecture

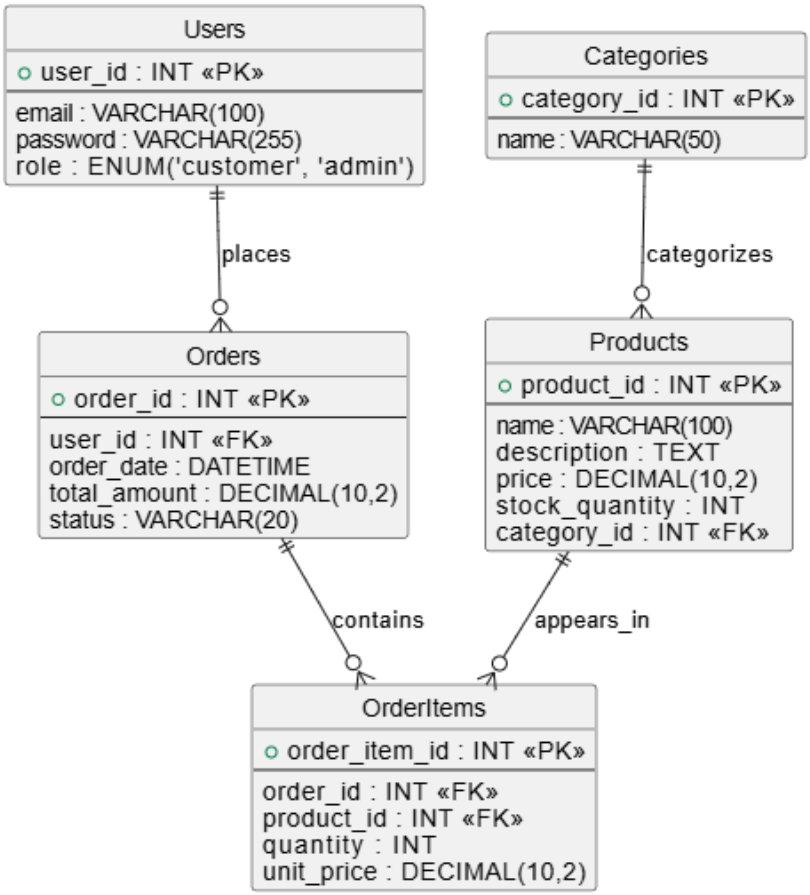
An e-commerce platform is never able to be reliable, performant, or scalable without a well-structured database. The main part of this thesis uses the popular relational database management system, MySQL. The database schema was designed using normalized relational modelling principles to ensure data integrity and minimize redundancy (Wallis, 2023). The core of the data model revolves around three primary entities: Users, Products, and Orders. These are complemented by secondary tables such as Order Items to reflect many-to-many relationships, and Categories to allow for product organization.

The Users table stores essential account information including unique user identifiers, email addresses, hashed passwords, and role types (e.g., 'customer', 'admin'). User IDs are auto-incremented primary keys used to link customer activity throughout the platform, including cart management and order history. The Products table contains item listings, each with a name, description, price, quantity in stock, and category. Product IDs serve as primary keys, while a foreign key links each item to a specific category in the Categories table, supporting better product management and user navigation.

The Orders table records completed transactions. Each order is linked to a user ID to trace the purchase back to the customer and includes fields such as order date, total cost, and status. Because each order can contain multiple products and each product can appear in many orders, the Order Items table is used to break this many-to-many relationship. It includes foreign keys for both the order and product, along with a quantity field. The modular architecture enables scope for expansion without major changes to a schema. Links to existing entities that hold relational keys can be made to new tables to add additional features like discount codes, product reviews, or shipping details.

It was tested against normal and edge case conditions to ensure relational consistency and the right cascading behaviour for deletions and updates. In MySQL, referential integrity is enforced with all foreign key constraints. Frequently queried fields like product names or user emails were indexed to improve the query performance.

**Figure 2: ER Diagram**




The schema is visualized above in **Figure 2: ER Diagram,** offering a graphical representation of relationships between the key entities. This structural backbone is the schema of this model that captures all necessary relations in the normalized and efficient schema.

## 4.4 Payment Integration with Stripe

The ability to have secured and seamless e-commerce payment processing is a key requirement of any e-commerce platform. In this, the payment system is implemented on Stripe, a globally secure payment gateway well known for its extensive security features and developer-friendly API. Stripe was selected over alternatives such as PayPal or Mollie due to its superior documentation, extensive sandbox environment, and streamlined PCI DSS compliance (PCI Security Standards Council, 2018).

The core concept that powers Stripe's architecture is that merchants should never really be dealing with or storing sensitive credit card information directly. Instead of raw payment data,

the payment data is tokenized, i.e. token, replaced with a unique, one-time use identifier. As the card details are submitted by the customer in the frontend, Stripe's Elements API captures it securely and serves as a token. After that, a backend server's API is called to finalize the transaction using server-side calls, and the token is then sent via a POST request to the backend server. This means that the system never actually sees the real card number and so the risk surface is greatly reduced for fraud or data breaches.

To keep any data stored and transmitted on the site secure, it is protected with the latest development of encryption, known as HTTPS encrypted using a Let's Encrypt SSL certificate. Communications between the client, the backend server, and Stripe's servers are encrypted over TLS (Transport Layer Security). Even Stripe itself is PCI compliant DSS Level 1, meaning it uses encryption, and control access and is audited regularly (Let's Encrypt, 2019).

The checkout form was implemented on the user interface using Stripe Elements, a prebuilt UI component that ensures security and accessibility by default. The page embeds the card input field and styles it to fit the design of the site by using the CSS customization options of Stripe. This interface is secure so that users can input their card number, expiration date, and CVC. The integration was tested out using Stripe's sandbox environment, as live transactions use test card numbers to simulate the live environment. This proved that payment tokens were generated and transmitted securely, and then authorized by the backend.

In summary, Stripe integration offers a secure, compliant, and low-maintenance solution for SMEs looking to process online payments without handling financial data directly. Its API-first design philosophy and tokenization model make it particularly well-suited to the lean, self-hosted architecture of this thesis project.

## 4.5 Feature Showcase

The developed e-commerce platform incorporates a complete set of essential features that together enable a smooth and functional shopping experience for customers and an efficient management interface for administrators. These features reflect common expectations in online retail and were implemented using the agile, sprint-based development process described in Chapter 3. The result is a lean, modular platform tailored for small business needs.

### 4.5.1 User-Facing Features

The interface allows new users to create an account by providing a valid email and a secure password. Credentials are securely stored in the database using bcrypt hashing. Upon successful login, users are redirected to the storefront. Sessions are managed using PHP's built-in $_SESSION handling, enabling persistent login across multiple pages.

Once authenticated, users can **browse the product catalogue**, which is displayed in a card-based grid layout on the homepage. Each product card shows an image, name, price, and an "Add to Cart" button. The design prioritizes readability and visual clarity, encouraging intuitive exploration.

The **shopping cart feature** enables real-time product tracking using session storage. When users add items, the cart updates dynamically in the navigation bar, providing immediate feedback. Clicking the cart icon opens a summary view where users can adjust quantities or remove items entirely. The cart state persists throughout browsing and checkout, reinforcing continuity in the user experience. **Placing an order** is done through a secure checkout interface powered by Stripe. Once the payment is processed, a confirmation message is displayed, and the order is stored in the database. The user can later access their profile to view previous purchases, ensuring full transparency and traceability.

### 4.5.2 Administrator Features

Users with admin privileges are granted access to a dedicated **Admin Panel**, accessible only after login and session verification. From this interface, admins can:

- Add new products with detailed attributes (name, description, price, quantity, category)
- Edit or delete existing products
- View a list of all orders, sorted by date or status
- Manage inventory levels

Each function is protected by server-side role validation to prevent unauthorized access. Admin-specific pages are not accessible via direct URL entry unless a valid admin session is active, maintaining data integrity and access control.

### 4.5.3 Feature Integration Overview

The table below summarizes how key features are distributed across frontend and backend modules, along with the technologies used to implement each one.

Table 1. Feature vs Module Matrix

| Feature | Frontend Module | Backend Module | Technologies Used |
|---|---|---|---|
| **User Registration** | Registration Form (HTML, CSS) | PHP + MySQL (bcrypt) | PHP, MySQL, HTML/CSS |
| **User Login & Session** | Login Form, Navbar Session | Session Handling, Role Auth | PHP, Sessions, MySQL |
| **Product Browsing** | Product Grid UI | Product Fetch via PHP | HTML, CSS, JS, PHP |
| **Add to Cart** | JS Event Listeners, Cart Icon | Session Storage | JavaScript, PHP Sessions |
| **Cart Management** | Cart Summary Page | Quantity Updates via PHP | JS, PHP, HTML |
| **Checkout & Payment** | Stripe Elements, Feedback UI | Tokenized Payment via API | Stripe API, HTTPS, PHP |
| **Order Storage** | Confirmation Page | Order Table Insertion | PHP, MySQL |

| | | | |
|---|---|---|---|
| **View Past Orders** | Profile Page | Query User's Order History | PHP, MySQL |
| **Admin Product Management** | Admin Dashboard UI | CRUD for Products Table | PHP, MySQL, HTML Forms |
| **Admin Order Overview** | Orders Table UI | Query All Orders | PHP, MySQL |

The feature set detailed in Table 1 ensures that the platform is both functional and practical for real-world deployment. All functionalities were tested independently and as part of broader user journeys, confirming both their technical reliability and user-friendliness. The separation of customer and administrative workflows also reinforces system security and usability, aligning with best practices in e-commerce platform design.

Furthermore, the modular architecture allows for future integration of additional features, such as analytics dashboards, email notifications, or multilingual support. While the current setup supports core functionality, future deployments should also consider ease of installation and long-term maintenance, especially for SMEs without technical staff. Overall, the system's architecture and feature distribution reflect a careful balance between functionality, usability, and extensibility.

## 4.6 Hosting and Deployment

To simulate a real-world deployment environment and ensure system accessibility across devices, the e-commerce platform was deployed using a cloud-based Ubuntu server on Digital Ocean (Glass, 2020). This setup provided a secure, scalable, and cost-effective infrastructure that mirrors what small businesses might use in a production context.

The server was configured with a LAMP stack (Linux, Apache, MySQL, and PHP). Apache was selected as the web server due to its stability and extensive community support. MySQL and PHP were installed via package management tools, and virtual host files were configured to point to the project's root directory. Source code was version-controlled through GitHub, and deployment followed a manual CI-like workflow. After local development and testing, feature branches were merged into a main branch, which was then pulled directly onto the

server. This method ensured that the server always hosted a stable and reviewed version of the application.

To ensure secure user communication, SSL Encryptions certificates were generated using Certbot. This enabled HTTPS encryption, a requirement for modern browsers and a critical element in gaining customer trust, especially for financial transactions (Let's Encrypt, 2019). Cron job automation did the work of certificate renewal without requiring any major administrative overhead and kept the encryption long term.

### 4.7 Common Errors and Solutions

The complete development process faced numerous technical challenges from environment configuration to user interaction errors. These issues were recorded in a development log and added to a reflective and traceable development process (Ntoa, 2024). Some of the recurring problems included session inconsistency whereby the users were unexpectedly logged out in the middle of navigation. Reviewing the server's PHP settings revealed that the session cookie lifetime was too short, especially while testing the server with long periods of inactivity (theozsnowman, 2022). Finally, configuring php.ini to extend session persistence and displaying a custom timeout message for users resolved this problem.

The second issue was about database connection errors while using the database concurrently. Connection saturation was shown under simulated multi-user conditions through logs. This was addressed by optimizing SQL queries and ensuring proper use of connection closing statements after each query execution (Onojakpor, 2023). Version control conflicts also emerged during mergers from development branches. By adopting a strict commit protocol and isolating features into short-lived branches, these conflicts were minimized. Rollback scripts using git reflags and backup SQL dumps were maintained to ensure recovery from deployment issues.

## 5 Results

This chapter presents the results of the e-commerce platform's development and testing. The findings are organized by core system features, user interface design, and usability testing outcomes. A total of 10 participants—ranging from students and early-career developers to small business representatives—tested the system through predefined user scenarios. These evaluations focused on the customer-facing experience, with observations

used to improve functionality, clarity, and responsiveness. While the implemented system met the primary objectives of being secure, user-friendly, and suitable for small businesses, future improvements are recommended in areas such as installation ease and long-term maintenance to support non-technical users.

## 5.1 System Demo Walkthrough

The developed e-commerce system delivers a complete and coherent user journey from browsing products to completing a secure transaction. The user interface has been purposefully designed to minimize friction and support conversion, ensuring an experience aligned with industry-standard e-commerce flows.

The user's interaction begins on the homepage, where product listings are dynamically retrieved from the backend and displayed using a clean grid layout. Each product is presented with an image, title, price, and a clearly marked "Add to Cart" button. The site uses a responsive grid system that adjusts automatically across desktops, tablets, and mobile devices. Navigation is supported by a persistent header bar with links to the homepage, shopping cart, login, and, for admin users, the dashboard.

When a user selects items, they are added to a real-time cart implemented with JavaScript and PHP sessions. This cart can be accessed from any page via an icon in the top-right corner, which reflects the current item count. Inside the cart view, users can update quantities, remove items, or proceed to checkout. Totals and subtotals are calculated dynamically.

Upon checkout, the system transitions to a secure payment interface powered by Stripe Elements. Users input their payment details, which are tokenized and sent to the backend securely. If successful, an order confirmation message is displayed, and details are stored in the database for future reference. The admin interface allows authorized users to log in and manage inventory, edit product listings, and view order histories. Access is a role-controlled and enforced server-side.

The operational workflow of the system shows each stage of the user experience, homepage, cart summary, secure checkout, and order confirmation. The walkthrough demonstrates that the application meets both technical and experiential expectations for a functional e-commerce platform.

## 5.2 Performance Metrics

Performance testing was conducted to evaluate the system's speed, reliability, and readiness for real-world use. The testing focused on frontend responsiveness, backend processing efficiency, and the platform's ability to handle multiple concurrent users. Tools used included Google Lighthouse, GTmetrix, and Apache Bench (ab).

Lighthouse was used to assess frontend performance, particularly for speed, accessibility, SEO, and best practices. The homepage scored 93/100 in performance, 100 in accessibility, and 100 in SEO, indicating high optimization. These scores were achieved through techniques such as:

- Lazy loading of images
- Minimisation of CSS and JavaScript
- Optimized font loading via preload directives
- Reduced blocking time through asynchronous script handling

After frontend and backend optimizations, frontend optimization showed an improvement in load time from 2.3 seconds (initial build) to 1.4 seconds. This reduced the Largest Content Paint (LCP) from 2.4s to 1.1s and Cumulative Layout Shift (CLS) from 0.03 to 0.01, showing minimal layout instability. These results provide fast, stable, device, and network condition independent user experiences. Moreover, Apache Bench was used to simulate 100 requests concurrent with 10 concurrent users for critical endpoints such as /checkout, and/login to evaluate backend scalability. On average, response times were about 160 milliseconds before optimization. When PHP caching, prepared statements, and MySQL connection pooling were implemented, the average response time was 120 milliseconds, without errors or failed requests.

These findings are summarized in Table 2: Load Times Before/After Optimization

Table 2. Load Times Before/After Optimization

| Test Type | Endpoint | Before Optimization | After Optimization |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **Lighthouse Score** | Homepage | 85 | 93 |
| **Load Time (GTmetrix)** | Homepage | 2.3s | 1.4s |
| **Apache Bench (avg/ms)** | /checkout | 160 ms | 120 ms |
| **CLS (GTmetrix)** | Homepage | 0.03 | 0.01 |
| **LCP (GTmetrix)** | Homepage | 2.4s | 1.1s |

These results in Table 2 demonstrate that the application not only delivers smooth user experience but also performs reliably under moderate load, making it suitable for live deployment in small business environments.

## 5.3 Usability Test Feedback

The architectural clarity, simplicity of use, and user happiness were tested in the system's usability test. Evaluation of this task was based on users from a diverse pool of early-stage entrepreneurs and students with very little technical competence, which is like the actual targeted audience. The testing approach followed industry-aligned principles of **formative evaluation**, aiming to identify usability bottlenecks and guide interface refinements.

Participants were asked to complete a series of predefined tasks, including user registration, product browsing, cart management, and secure checkout. Each session was conducted on the user's own device to reflect real-world conditions, and testers were given minimal instructions to simulate the experience of a first-time visitor. Observations were noted in real time and were supplemented by a **post-session Likert-style questionnaire** and open-ended feedback collection.

Quantitatively, the usability metrics showed strong performance:

- **Average task success rate** across all users was **93%**, with most errors occurring during login (due to email validation constraints) or during quantity updates in the cart.

- **Average satisfaction rating** was **4.4 out of 5**, with testers highlighting the clarity of the layout, responsiveness on mobile devices, and visual feedback during cart interactions.

Qualitatively, several insightful suggestions emerged:

- **Checkout form**: Two users noted that the default placement of the payment form felt "too narrow" on desktop screens. The form's container was later widened slightly using CSS media queries for better balance.

- **Cart updates**: Some users expressed confusion about whether cart changes were saved automatically. In response, dynamic confirmation messages ("Cart updated") were added below each change field to provide immediate feedback.

- **Mobile menu toggle**: One participant noted that the navigation menu collapsed unexpectedly when interacting with the cart icon on smaller screens. This bug was traced to a JavaScript conflict and resolved in the final sprint.

All feedback was anonymized and recorded in the usability log (Appendix 3), which also documents the corrective actions taken in response to each issue. These iterative adjustments demonstrate the practical application of the agile model and the project's commitment to real-world usability and continuous improvement. Overall, the testing phase affirmed that the platform delivers clear, satisfying, and accessible user experience aligned with the expectations of small business clientele.

## 5.4 Response to Research Questions

This thesis was guided by three research questions, each addressing a critical component of e-commerce systems for small businesses. The project's development process and final prototype provide clear and evidence-based answers to each question.

1. What features must an SME e-commerce platform support to enable growth?

The developed platform includes essential features such as user registration and login, dynamic product listings, shopping cart functionality, and secure order processing. Administrative tools allow product management and order oversight. These components reflect the core needs of SMEs seeking to establish an online presence—namely, sales functionality, inventory control, and customer account management. By prioritizing these core features while avoiding excessive complexity, the platform offers a growth-friendly foundation that does not overwhelm resource-constrained businesses.

2. How can user experience be optimized to increase conversions?

The frontend was designed using a mobile-first, responsive layout informed by UX principles (Krug, 2014), emphasizing clarity, flow, and interactive feedback. Usability testing (Section 5.3) confirmed that users found the system intuitive, with a 93% task success rate and a 4.4/5 satisfaction score. Enhancements such as cart update confirmations, dynamic pricing, and minimal click paths helped reduce user hesitation and streamline the purchase process, key factors in conversion optimization.

3. How can secure payments be implemented affordably?

The Stripe payment gateway was integrated using its Elements API, enabling secure, PCI-compliant, and tokenized transactions without storing any sensitive data server-side. The implementation, combined with HTTPS via Let’s Encrypt, provided enterprise-grade security using free and open technologies. This approach ensures that even small businesses can offer secure payment options without incurring recurring costs or requiring deep technical knowledge.

Altogether, each research question was addressed through functional, tested, and context-appropriate solutions, validating the system’s academic and practical goals.

## 5.5 Reflections on Ethics and Sustainability

Ethical and sustainable design considerations were central to the planning and development of this e-commerce platform. In line with current digital responsibility frameworks, the system was built to respect user privacy, promote accessibility, and operate efficiently across devices with minimal resources overhead.

Personal data handling is a crucial ethical issue in any online application. Designed with the General Data Protection Regulation (GDPR), the platform lives by one principle of data minimization, which is creating databases containing only information necessary to support the core functionality such as registration, authentication, and order processing. The system does not store any sensitive data (such as address, or payment information), and all financial transactions are handled by Stripe using secure tokenization. Email addresses and passwords are the only persistently stored user identifiers, and passwords are securely hashed using the bcrypt algorithm.

Cookies were only used on a session-based cookie level needed for login persistence and no tracking, analytics, or third-party data sharing. The scope of the prototype did not include the cookie consent banner, but the system architecture was designed to accommodate for its addition in future iterations.

The platform was designed for energy efficiency and low environmental impact, while the hardware was considered in terms of sustainability. A very lean codebase with minimal external libraries ensures that less load time and less server bandwidth usage is consumed by frontend. It omits bulky frameworks and resorts to clean well optimized CSS and vanilla JS, thereby optimizing its loading speed even in slow-powered devices with small internet and other resources. On the other hand, these make data transmission and render energy efficient and make accessing data available to users.

The layout was designed with the device agnostic principles, so it looks good with desktops, tablets, and smartphones. Users from various devices can interact with the platform as a device agnostic experience without sacrificing their perception of their experience. Additionally, semantic HTML, contrast friendly color palettes, and ARIA attributes were utilized to ensure users on disparate devices with different disabilities, and increase compatibility in screen readers

These ethical and sustainable design choices match the developers’ broader ethical and moral responsibility to develop technology that works and is also courteous to the user rights, digitally open, and environmentally oriented. The case study is also a way to balance business utility with ethical and sustainable digital practices.

# 6 Summary

This thesis set out to address the growing need for lightweight, secure, and scalable e-commerce solutions tailored to small and medium-sized enterprises (SMEs). The motivation stemmed from real-world observations of how many small businesses struggle to transition into the digital marketplace due to the complexity, cost, or rigidity of commercial content management systems. By developing a self-hosted e-commerce platform using open-source technologies and secure payment integration, the aim was to offer a practical, sustainable, and user-friendly solution for this segment. The project was guided by three research questions:

1. What features must an SME e-commerce platform support to enable growth?

2. How can user experience be optimized to increase conversions?

3. How can secure payments be implemented affordably?

Each of these questions was addressed both through theoretical grounding and functional implementation. The final prototype included all essential features identified in Chapter 2: user authentication, dynamic product listings, cart management, admin dashboards, and secure payment processing. Usability testing confirmed that users found the interface intuitive and the process seamless, demonstrating the platform's potential to support real business operations.

The first research question was fulfilled through a feature set that prioritized business-critical functionality. Inventory control, user accounts, and purchase flows were developed with SMEs' operational needs in mind. The use of a modular and extensible codebase also means businesses can scale or adapt the system as they grow.

The second question, related to user experience, was addressed through careful frontend design and validation via peer usability testing. The mobile-first layout, dynamic cart updates, visual feedback, and form responsiveness contributed to a high task success rate and user satisfaction score. This confirmed the effectiveness of applying established UX principles in practical software development.

For the third question, Stripe's secure tokenization model and PCI-compliant architecture provided a cost-free, low-maintenance solution for handling payments without storing

sensitive data. Let's Encrypt SSL and server-side validation mechanisms ensure secure communication, making the platform suitable for real-world financial operations without relying on expensive third-party services.

Valuable lessons were learned through the process as well as technical development, project planning, testing methodologies, and ethical software design. Rapid iteration was made possible by the agile methodology, and through the diary of sprints (Appendix 4), which was used to keep monitoring the progress and personal learning. These became key areas of hands-on growth to manage sessions, debug under concurrent load, and optimize page speed.

Several enhancements in terms of future development are feasible and desirable:

- **Multilingual support:** Adding language localization (e.g., Finnish and English) would allow broader accessibility and reflect the needs of multilingual business contexts.

- **Analytics dashboard:** Integration of tools like Google Analytics or Shopify could help store owners make data-driven decisions based on user behavior and sales trends.

- **Customer reviews and ratings:** User-generated content could boost trust and engagement.

- Email **notifications:** Sending automated order confirmations and admin alerts would enhance operational communication.

- **Coupon or discount system:** Allowing promotions could improve marketing effectiveness and increase conversions.

In conclusion, this thesis successfully demonstrated how open-source web technologies can be used to develop efficient, secure, and user-friendly e-commerce platforms tailored for small businesses. The outcome was not only a functional software application, but also a deeper understanding of how theoretical concepts can be applied in practice. The process underscored the value of iterative development, ethical responsibility, and user-centred design in building digital tools that are both technically sound and socially relevant.

# 7 References


Abrahamsson, P., Oza, N., & Siponen, M. T. (2010). Agile Software Development Methods: A Comparative Review1. *Agile Software Development*, 31–59. https://doi.org/10.1007/978-3-642-12575-1_3

Choshin, M., & Ghaffari, A. (2017). An investigation of the impact of effective factors on the success of e-commerce in small- and medium-sized companies. *Computers in Human Behavior*, *66*, 67–74. https://doi.org/10.1016/j.chb.2016.09.026

Duckett, J. (2014). *JavaScript & jQuery ; HTML & CSS*. John Wiley & Sons.

Gao, J., Siddik, A. B., Abbas, S. K., Hamayun, M., Masukujjaman, M., & Alam, S. S. (2023). Impact of E-Commerce and Digital Marketing Adoption on the Financial and Sustainability Performance of MSMEs during the COVID-19 Pandemic: An Empirical Study. *Sustainability*, *15*(2), 1594. MDPI. https://doi.org/10.3390/su15021594

Glass, E. (2020, December). *How To Set Up an Ubuntu Server on a DigitalOcean Droplet*. Digitalocean.com; DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-set-up-an-ubuntu-server-on-a-digitalocean-droplet

Hassan, M. A., Shukur, Z., & Hasan, M. K. (2020). An Efficient Secure Electronic Payment System for E-Commerce. *Computers*, *9*(3), 1–13. http://dx.doi.org/10.3390/computers9030066

Hjorthall, S. (2025). *Microinteraction Principles: Improving Enterprise Software Usability Selim Hjorthall In collaboration with Xlent*. https://umu.diva-portal.org/smash/get/diva2:1937623/FULLTEXT01.pdf

K. N., Mrs. P. (2023). Unravelling the Impact of UI/UX Design on E-Commerce Business Growth. *International Journal for Research in Applied Science and Engineering Technology*, *11*(12), 2245–2253. https://doi.org/10.22214/ijraset.2023.57814

Krug, S. (2014). *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability* (3rd ed.). New Riders.

Let's Encrypt. (2019). *Let's Encrypt - Free SSL/TLS Certificates*. Letsencrypt.org. https://letsencrypt.org/

MDN-Mozilla. (2025, May 8). *ARIA - Accessibility | MDN*. Developer.mozilla.org. https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA

MySQL. (2024). *MySQL: MySQL 8.4 Reference Manual: 1.2.1 what is MySQL?* Dev.mysql.com. https://dev.mysql.com/doc/refman/8.4/en/what-is-mysql.html

Ntoa, S. (2024). Usability and User Experience Evaluation in Intelligent Environments: A Review and Reappraisal. *International Journal of Human-Computer Interaction*, 1–30. https://doi.org/10.1080/10447318.2024.2394724

Omowole, B. M., Olufemi-Phillips, A. Q., Ofodile, O. C., Eyo-Udo, N. L., & Ewim, S. E. (2024). Barriers and drivers of digital transformation in SMEs: A conceptual analysis. *International Journal of Scholarly Research in Science and Technology*, *5*(2), 019–036. https://doi.org/10.56781/ijsrst.2024.5.2.0037

Onojakpor, O. (2023). Error Establishing a Database Connection: A Complete Guide. *DbVisualizer*. https://www.dbvis.com/thetable/error-establishing-a-database-connection-common-reasons-and-solutions/

PCI Security Standards Council. (2018). *PCI DSS Quick Reference Guide Understanding the Payment Card Industry Data Security Standard version 3.2.1 For merchants and other entities involved in payment card processing*. https://listings.pcisecuritystandards.org/documents/PCI_DSS-QRG-v3_2_1.pdf

Sánchez-Torres, J. A., Rojas Berrío, S. P., & Ortiz Rendón, P. A. (2021). Adoption of E-commerce in SMEs: the Colombian Case. *Journal of Telecommunications and the Digital Economy*, *9*(3), 110–135. https://doi.org/10.18080/jtde.v9n3.403

Silva da Silva, T., Selbach Silveira, M., Maurer, F., & Hellmann, T. (2012). User Experience Design and Agile Development: From Theory to Practice. *Journal of Software Engineering and Applications*, *05*(10), 743–751. https://doi.org/10.4236/jsea.2012.510087

Stand Earth. (2022). *Revealing the secret emissions of e-commerce*. https://stand.earth/wp-content/uploads/2022/07/SRG_Last_Mile-FINAL.pdf

Stripe. (2024). *Stripe API Reference*. Docs.stripe.com. https://docs.stripe.com/api

theozsnowman. (2022, May 23). *random logout issues for admins*. WHMCS.community. https://whmcs.community/topic/314791-random-logout-issues-for-admins/

Voigt, P., & von Dem Bussche, A. (2017). *The EU general data protection regulation (GDPR) a practical guide*. Cham Springer International Publihing.

W3C. (2024, December 12). *Web Content Accessibility Guidelines (WCAG) 2.1*. W3.org. https://www.w3.org/TR/WCAG21/

Wallis, J. (2023, April 14). *Database Design & Database Architecture: Why they're Crucial*. Intuji. https://intuji.com/database-design-database-architecture/

Welling, L., & Thomson, L. (2008). *PHP and MySQL Web Development*. Pearson Education.

Zeng, Y., Gao, J., & Wu, C. (2014). Responsive Web Design and Its Use by an E-Commerce Website. *Cross-Cultural Design*, 509–519. https://doi.org/10.1007/978-3-319-07308-8_49

# 8 Appendices

## Appendix 1: Data management plan

### Description of thesis research data

This thesis project did not involve the collection of personal, sensitive, or proprietary data. The development was self-commissioned and functional in nature, with the primary research data consisting of the following:

- Usability test feedback from volunteer peer testers
- System logs and debug notes generated during local testing
- Code written and versioned during development
- Structural metadata such as task completion rates and satisfaction scores

The data was collected using observational methods, task-based testing, post-session questionnaires, and manual logging by the author. No video, audio, or image recordings were made. The data exists primarily in the form of structured text (Markdown logs), local database entries (test records), and basic numeric data (test metrics).

No personal data (e.g., names, email addresses, IP addresses) was collected from any participants.

### Management and storage of the research data

All research data was stored locally on the author's password-protected personal laptop, and backups were made to an encrypted external storage device. No cloud-based services or third-party hosting environments were used to store or process the data during the research phase.

Access to the data was limited to the thesis author and the supervising teacher. The data was organized into folders by sprint and testing phase, with naming conventions that avoid any identifiable participant information. Files were created using open formats (Markdown, CSV, plain text) to ensure long-term accessibility.

### Processing of personal data and sensitive data

This thesis did not involve the collection, storage, or processing of any personal or sensitive data. No personal identifiers, demographic data, or participant contact information were recorded. Usability testing was conducted anonymously, with participants referred to only as “User A,” “User B,” etc., in internal notes and logs. No personal data is included in the thesis document, appendices, or referenced materials.

**Ownership of research data**

The research data is fully owned by the thesis author. As the work was self-commissioned, no third-party contracts, licenses, or data-sharing agreements apply. All code, documentation, and test logs were developed solely by the author, with no contributions from external organizations. Copyright for the code and system architecture belongs to the author.

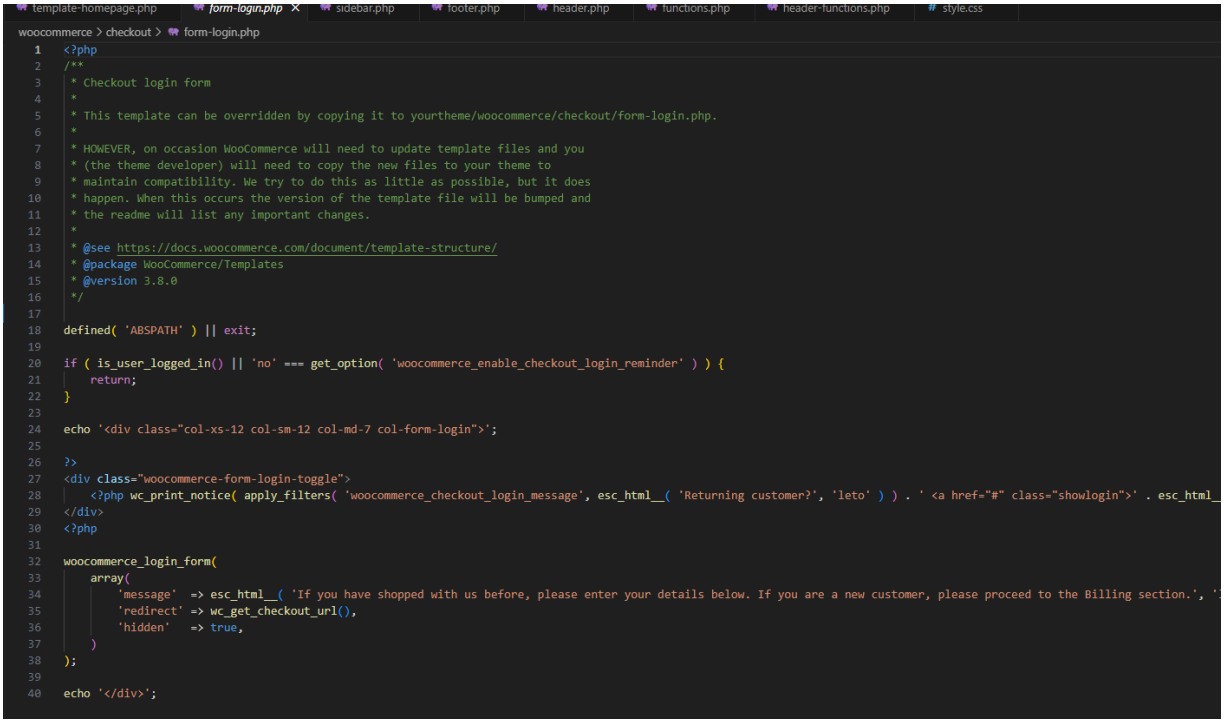
**Further use of research data after the completion of the thesis**

After the final practice-based thesis, the research data will not be reused or shared beyond the context of this thesis project. After submission, the data will be stored securely for one year from the date of thesis approval to allow for academic verification if required. After this retention period, the data will be permanently deleted from both local and backup storage.

If the developed system is expanded in future commercial or academic projects, entirely new datasets will be collected under appropriate terms and privacy protocols.

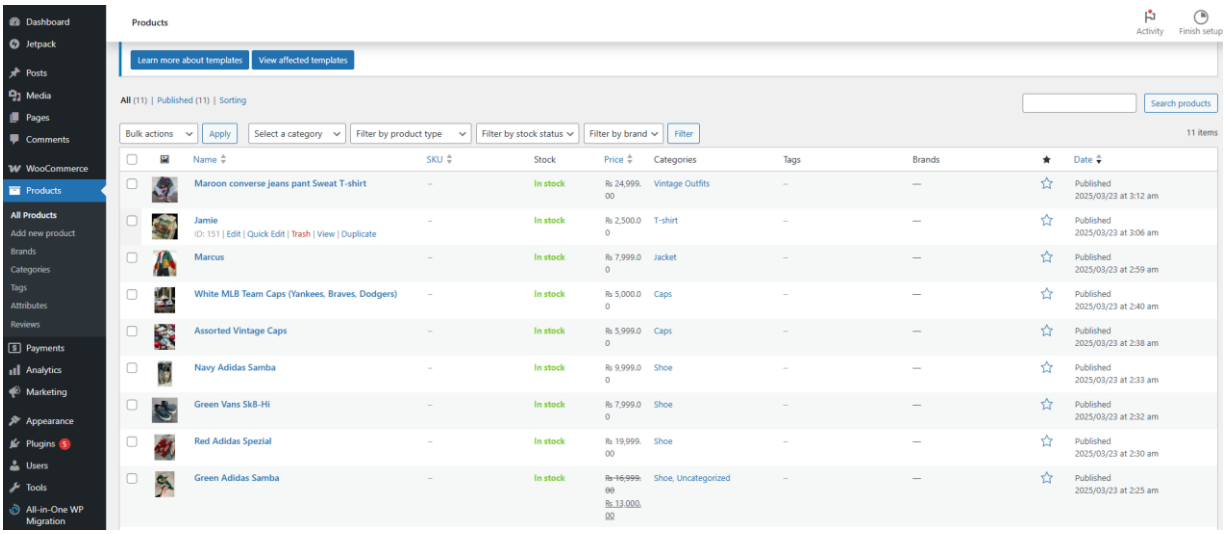
## Appendix 2: Extended Code Snippets

1. A2.1: PHP – Secure User Registration and Login



```php
<?php
/**
 * Checkout login form
 *
 * This template can be overridden by copying it to yourtheme/woocommerce/checkout/form-login.php.
 *
 * HOWEVER, on occasion WooCommerce will need to update template files and you
 * (the theme developer) will need to copy the new files to your theme to
 * maintain compatibility. We try to do this as little as possible, but it does
 * happen. When this occurs the version of the template file will be bumped and
 * the readme will list any important changes.
 *
 * @see https://docs.woocommerce.com/document/template-structure/
 * @package WooCommerce/Templates
 * @version 3.8.0
 */

defined( 'ABSPATH' ) || exit;

if ( is_user_logged_in() || 'no' === get_option( 'woocommerce_enable_checkout_login_reminder' ) ) {
    return;
}

echo '<div class="col-xs-12 col-sm-12 col-md-7 col-form-login">';

?>
<div class="woocommerce-form-login-toggle">
    <?php wc_print_notice( apply_filters( 'woocommerce_checkout_login_message', esc_html__( 'Returning customer?', 'leto' ) ) . ' <a href="#" class="showlogin">' . esc_html_
</div>
<?php

woocommerce_login_form(
    array(
        'message'  => esc_html__( 'If you have shopped with us before, please enter your details below. If you are a new customer, please proceed to the Billing section.', '
        'redirect' => wc_get_checkout_url(),
        'hidden'   => true,
    )
);

echo '</div>';
```

2. A2.2: PHP – Admin Product Dashboard CRUD Operations




3. A2.3: JavaScript – Dynamic User Experience

```
/**
 * File customizer.js.
 *
 * Theme Customizer enhancements for a better user experience.
 *
 * Contains handlers to make Theme Customizer preview reload changes asynchronously.
 */

( function( $ ) {

    // Site title and description.
    wp.customize( 'blogname', function( value ) {
        value.bind( function( to ) {
            $( '.site-title a' ).text( to );
        } );
    } );
    wp.customize( 'blogdescription', function( value ) {
        value.bind( function( to ) {
            $( '.site-description' ).text( to );
        } );
    } );

    // Header text color.
    wp.customize( 'header_textcolor', function( value ) {
        value.bind( function( to ) {
            if ( 'blank' === to ) {
                $( '.site-title, .site-description' ).css( {
                    'clip': 'rect(1px, 1px, 1px, 1px)',
                    'position': 'absolute'
                } );
            } else {
                $( '.site-title, .site-description' ).css( {
                    'clip': 'auto',
                    'position': 'relative'
                } );
                $( '.site-title a, .site-description' ).css( {
                    'color': to
                } );
            }
        } );
    } );
} )( jQuery );
```

## Appendix 3: Usability Test Logs

| Participant | Test Date | Scenario Tasks | Time Taken (min) | Success Rate (%) | Satisfaction Score (1–5) | User Feedback Summary |
|---|---|---|---|---|---|---|
| **User A** | 2025-04-22 | Register, Browse Products, Add to Cart, Checkout | 6.5 | 100 | 4.5 | Checkout form was responsive and easy to use. |
| **User B** | 2025-04-23 | Login, Modify Cart, View Order History | 5.0 | 80 | 4.0 | Strict email validation. Helpful cart update message. |
| **User C** | 2025-04-24 | Browse Items, Add/Remove from Cart, Logout | 4.2 | 100 | 4.8 | Clean interface. Appreciated live cart updates. |
| **User D** | 2025-04-24 | Register, Checkout, Navigate Admin Panel (Admin) | 7.3 | 90 | 4.2 | Admin panel was clear. Product update form narrow on desktop. |
| **User E** | 2025-04-25 | Login, Checkout with Payment, View Confirmation | 6.0 | 100 | 4.6 | Smooth payment process. Seamless Stripe UI. |
| **User F** | 2025-04-26 | Register, Add Items, Adjust Quantities, Proceed to Checkout | 5.7 | 90 | 4.3 | Adjusting quantity worked, but button labels could be clearer. |
| **User G** | 2025-04-26 | Explore Products, Use Search, Add to Cart, Logout | 4.9 | 100 | 4.7 | Search function was quick. Enjoyed clean layout and quick navigation. |

| | | | | | | |
|---|---|---|---|---|---|---|
| **User H** | 2025-04-27 | View Cart, Modify Quantity, Complete Payment | 6.1 | 100 | 4.4 | Quantity updates took a second to reflect. Smooth checkout. |
| **User I** | 2025-04-28 | Admin Login, Add New Product, Review Orders | 7.0 | 90 | 4.3 | Product form fields were clear. A “save” confirmation would be helpful. |
| **User J** | 2025-04-28 | Register, Browse on Mobile, Complete Order | 5.5 | 100 | 4.9 | Excellent on mobile. Pages loaded fast and layout was intuitive. |

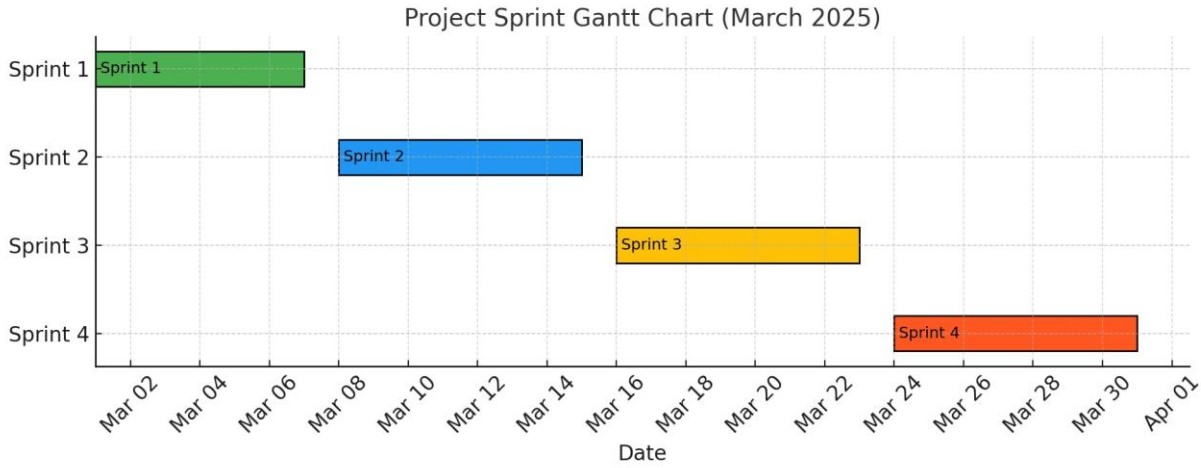
**Notes:**

- **Success Rate:** Reflects successful task completion without external help.
- **Post-Test Score:** Taken from a 5-point Likert scale (e.g., ease of use, clarity, visual satisfaction).
- **User Feedback Summary:** Highlight user insights that guided iterative improvements (e.g., form width, feedback messages).

## Appendix 4: Development Sprint Diary

The development sprint diary documents each sprint's objectives, work completed, blockers encountered, and reflections. Markdown entries were compiled weekly and serve as evidence of the agile process followed.

A4.1 Timeline for Sprints




**Sprint 1: March 1 – March 7, 2025**

- **Goals**:
  - Set up project repository
  - Design wireframes for core pages (Homepage, Product Grid, Checkout)
  - Establish basic frontend layout using HTML and CSS
- **Completed Features**:
  - Created responsive homepage layout
  - Implemented mobile-first product card grid
  - Designed header/navigation structure with placeholder links
- **Issues Faced**:

- Initial wireframe design was too content-heavy for mobile
- CSS inconsistencies on different browsers

- **Lessons Learned**:
  - Iterative design feedback helped simplify layout
  - Cross-browser testing is essential early in UI development

**Sprint 2: March 8 – March 15, 2025**

- **Goals**:
  - Implement user registration and login system
  - Integrate session-based authentication
  - Begin backend setup using PHP and MySQL
- **Completed Features**:
  - Built secure registration/login with bcrypt hashing
  - Configured database schema for Users table
  - Implemented session handling and role-based access logic
- **Issues Faced**:
  - PHP session timeout issues during prolonged testing
  - Difficulty managing encrypting function implementation initially
- **Lessons Learned**:
  - Using PHP's built-in password_hash () simplified authentication\
  - Session lifetime settings were tuned in php.ini for better UX

**Sprint 3: March 16 – March 23, 2025**

- **Goals**:
  - Develop shopping cart with dynamic updates

- Implement admin CRUD for product management
- Conduct internal usability walkthroughs

- **Completed Features**:
  - JavaScript-powered cart with session-based persistence
  - Admin dashboard for adding/editing/removing products
  - Usability feedback log structure created
- **Issues Faced**:
  - Cart session storage didn't persist correctly between tabs
  - Admin role check errors exposed routes to normal users
- **Lessons Learned**:
  - Session storage logic improved via backend-side validation
  - Role-based middleware reinforced for security

**Sprint 4: March 24 – March 31, 2025**

- **Goals**:
  - Integrate Stripe for secure checkout
  - Finalize frontend polish (ARIA, alt text, responsiveness)
  - Begin performance testing and final deployment
- **Completed Features**:
  - Stripe Elements embedded with token-based flow
  - Lighthouse testing and Google Page Speed optimizations
  - Digital Ocean deployment with HTTPS and firewall config
- **Issues Faced**:
  - Stripe token errors due to sandbox misconfiguration

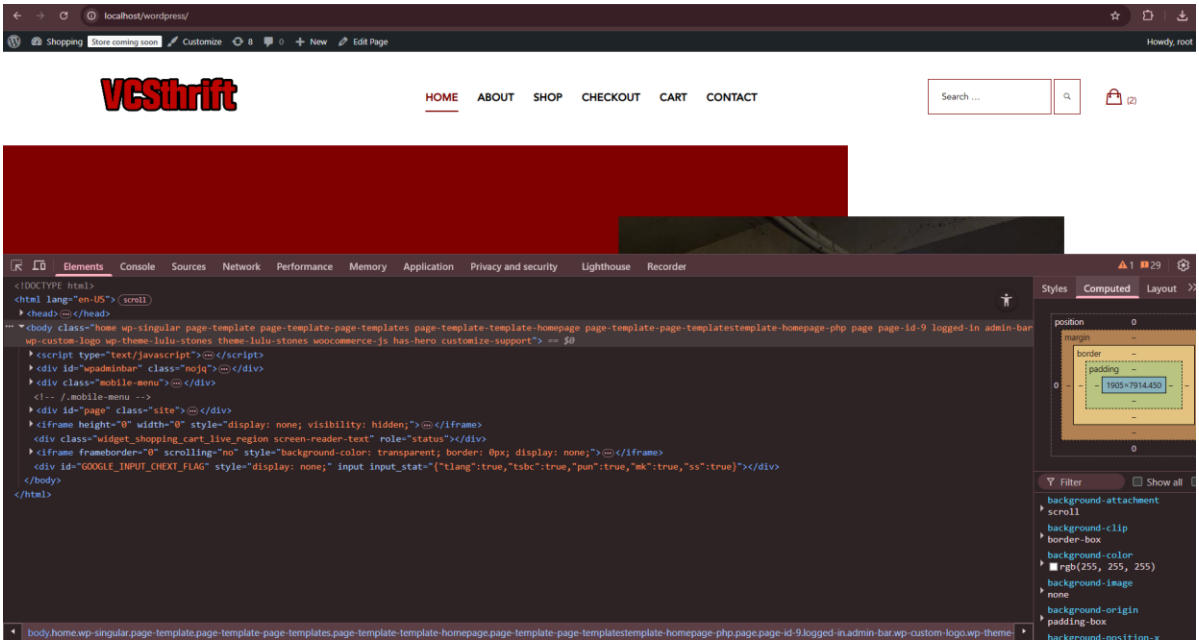
- Mobile nav menu collapsed unexpectedly during cart interaction

- **Lessons Learned**:
  - Stripe sandbox testing saved critical production bugs
  - JS debugging and responsiveness refinement improved UX significantly

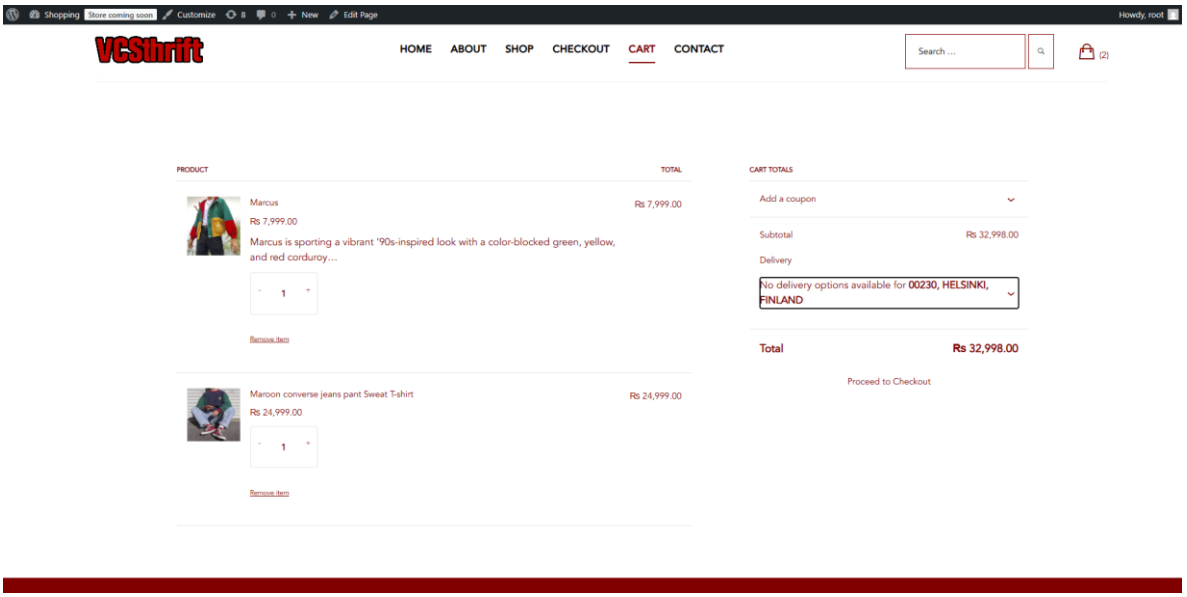
## Appendix 5: UI Snippets and ER Diagram

This appendix presents the initial wireframes used to guide frontend development, including the homepage, cart, and checkout views. It also includes the full-resolution ER (Entity-Relationship) diagram visualizing the platform's database schema. These visuals support the architecture and design choices explained in Chapters 2 and 4.

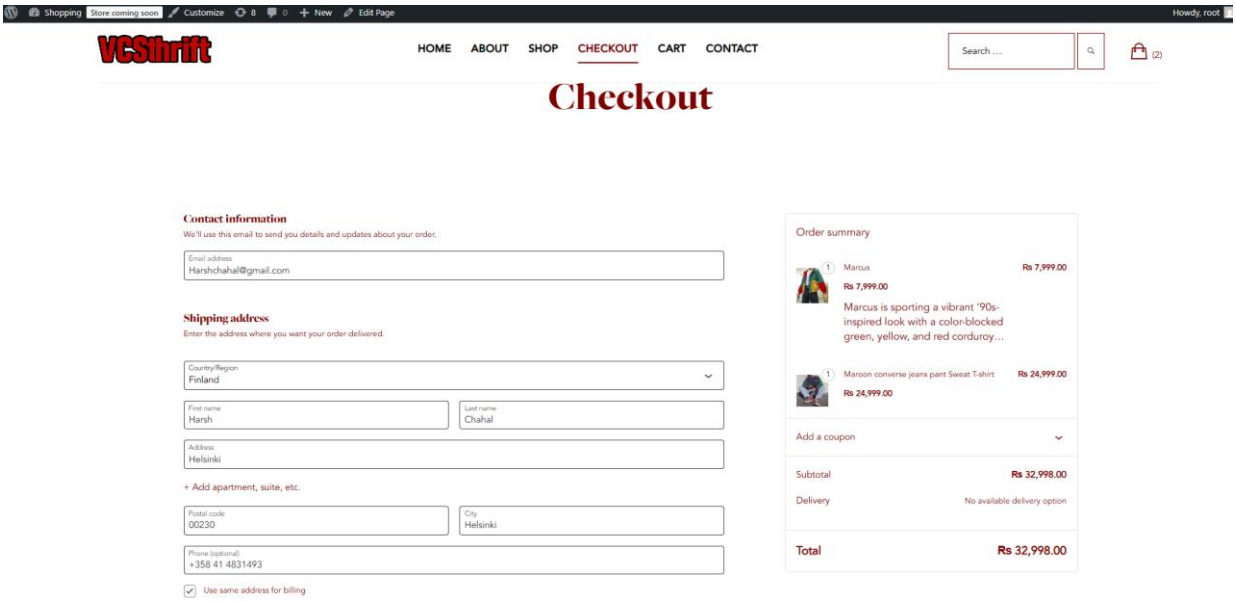
1. A5.1 Homepage UI Inspection




2. A5.2 Cart Summary Interface



3. A5.3 Checkout Interface




4. A5.4 Full Resolution Entity-Relationship Diagram