VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

DEVELOPMENT OF MODERN OF WEB APPLICATIONS

VÝVOJ MODERNÍCH WEBOVÝCH APLIKACÍ

THANH TUAN NGUYEN

Bachelor Thesis

Supervisor: Ing. Michal Radecký, Ph.D., MBA

Ostrava, 2023

Bachelor Thesis Assignment

Student: **Thanh Tuan Nguyen**

Study Programme: B2647 Information and Communication Technology

Study Branch: 2612R025 Computer Science and Technology

Title: Development of Modern Web Applications
Vývoj moderních webových aplikací

The thesis language: English

Description:

The aim of this thesis focused on nowadays platforms of web applications development. This thesis gives manual, how to select a proper technology framework and how to use it.

1. Analyze and describe nowadays web frameworks and connected technologies/tools.
2. Select one particular way of web app development and describe it in more detail.
3. Design case study of some application and describe development process including environment and related tools.
4. Implement designed application.
5. Evaluate selected framework and environment.

References:

[1] Jon Duckett: JavaScript and JQuery: Interactive Front-End Web Development, Wiley, 2014, ISBN: 978-1118531648
[2] Sasha Vodnik: HTML5 and CSS3, Illustrated Complete, Course Technology, 2015, ISBN: 978-1305394049
[3] Jason Beaird: The Principles of Beautiful Web Design, SitePoint, 2014, ISBN: 978-0992279448
[4] Erixc Elliot: Programming JavaScript Applications: Robust Web Architecture with Node, HTML5, and Modern JS Libraries, O'Reilly Media, 2014, ISBN: 978-1491950296

Extent and terms of a thesis are specified in directions for its elaboration that are opened to the public on the web sites of the faculty.

Supervisor: **Ing. Michal Radecký, Ph.D., MBA**

Date of issue: 01.09.2021

Date of submission: 30.04.2023

Fields of study guarantor: prof. RNDr. Václav Snášel, CSc.

In IS EDISON assigned: 07.11.2022 12:02:48

Abstract

This thesis focuses on the development of modern web applications, providing a guide on how to select a suitable technology framework and utilize it effectively. The thesis analyzes and describes current web frameworks and related technologies/tools, selects a particular framework for detailed description, and presents a case study for the development process, including environment and related tools. The designed application is implemented and evaluated, with the selected framework and environment being subject to analysis.

Keywords

HTML; CSS; JavaScript; ReactJS; NodeJS; MongoDB; Redux; framework; Web Development; npm

Abstraktní

Tato práce se zaměřuje na vývoj moderních webových aplikací, poskytuje návod, jak vybrat vhodný technologický rámec a efektivně jej využívat. Práce analyzuje a popisuje současné webové frameworky a související technologie/nástroje, vybírá konkrétní framework pro detailní popis a představuje případovou studii vývojového procesu, včetně prostředí a souvisejících nástrojů. Navržená aplikace je implementována a vyhodnocena, přičemž se analyzuje vybraný framework a prostředí.

Klíčové slovo

HTML; CSS; JavaScript; ReactJS; NodeJS; MongoDB; Redux; rámec; Vývoj webu; npm

I would like to thank Ing. Michal Radecký, Ph.D., MBA for his guidance, professional support for me to carry out my bachelor's thesis, and useful advice to carry out this thesis.

Contents

List of symbols and abbreviations

HTML	- Hyper Text Markup Language
CSS	- Cascading Style Sheet
JS	- JavaScript
JSX	- JavaScript XML
DOM	- Document Object Model
API	- Application Programming Interface
CDN	- Content Delivery Network
UI	- User Interface
MERN	- MongoDB-ExpressJS-ReactJS-NodeJS
NPM	- Node Package Manager
DB	- Database

List of Figure

List of Listing

1 INTRODUCTION

Technology is always changing over time, every day that passes can have a new technology born. Web development too, web technology also changes over time and there are always certain changes to bring more optimization, modernity, and convenience. Because of the development of technology, internet, and mobile devices, the demand for using modern web applications is increasing day by day. Therefore, developing modern web applications is an important topic in today's digital age and is attracting the interest and contributions of many researchers and businesses worldwide.

Web-related subjects and disciplines are increasing and growing, especially those related to Web Development, two popular disciplines related to Web Development are Graphic Design and Computer Science, and why is this subject so influential to so many people? Because people always use the Internet and Website every day, it is almost an irreplaceable necessity. So, the disciplines of Web Development gradually evolved, motivating more students, and developers to learn and build many websites with many different topics and fields. Besides, the development of the Web prompted them to create many Web Frameworks.

Today, many frameworks are born with more convenience in use and implementation such as React, Vue, Angular, and so on. Or languages that are very familiar to us like basic HTML5 (HTML, CSS, JavaScript). As a basic programming student, they can learn C because it is their first programming language, but nowadays everyone cans also choose C#, Java, C++, etc. as their first programming languages. For, web applications are ours today, also many choices of frameworks such as React, Vue, and Angular. With a complete analysis and understanding of frameworks, users will find and choose the right framework to build websites from a personal perspective.

Many frameworks appear and are popular, along with the support of the community to strengthen and develop the frameworks, we can also be more confident in the process of using those frameworks because when we have difficulties or if there is a problem during use, a strong community with many experienced programmers will help us fix and fix it easier.

In the web development framework, there are two types: Front-end framework and Back-end framework.

But in this thesis topic, the content will be focused more on the Front-end framework. Since the Front-end framework is the stub for users and developers to build the most basic things, the user interface is very important, they usually build the interface first and then develop the data structures to contain data or core parts that belong to the back-end management.

The main sources and content of this thesis are used on w3schools[1], with other sources on Google.

[1] https://www.w3schools.com/

2 FRONT-END DEVELOPMENT

The main focus of front-end development, also referred to as "client-side" development, is to provide users with an appealing visual experience when accessing a website or program. Along with ensuring a seamless and user-friendly experience, front-end developers are responsible for analyzing, designing, and debugging the application's code. Users oversee the website's appearance, feel, and final design as front-developer [1]. Users will pay closer attention to what people see when they access their app or website and ensure that it functions properly and is simple to use. New technologies and libraries like React, Vue.js, Angular, etc. also make front-end development easier and more convenient. Front-end development is an important part of building high-quality web applications and providing the best user experience.

Frameworks [2] [3] consist of pre-existing blocks of code that are grouped into a set of packages and programming libraries. Their platform offers pre-built functionalities such as mockups, APIs, and various components to streamline the process of creating sophisticated and interactive web applications. The frameworks are like having a pre-made house frame with a basic foundation, the developer just needs to build an interior as their like.

For example, to create a racing game, programmers will have to create (program) wheels, bodywork, people, roads, etc. and then start assembling the creation parts. But if we have the frameworks available, we just need to take them out of the framework and reassemble them. It will be fast and save a lot of money and time.

2.1 HTML

HTML5 [4] [5] is an upgrade of HTML (HyperText Markup Link) which is a quality HyperText Markup Language for web browsers, this is a language that allows creating web pages on many web browsers such as Google Chrome, Microsoft Edge, and so on.

When HTML was born, the difficulty that programmers faced was that some web browsers would be incompatible or not work well. So different browsers when displaying web pages will always have their way of working and compatibility, including mobile browsers, tablet browsers, etc. Programmers are always thinking and finding ways so that the HTML works fine on all web browsers without any difficulty. But when the HTML5 [1] version was released, those difficulties have been overcome and provided many features that bring many conveniences and benefits to developers.

Understanding HTML's basic features is crucial to starting web development, as it is one of the fundamental languages, along with CSS and JS. These three languages [6] are widely popular and used across multiple operating systems like Windows, Firefox, and Chrome, which have adopted open web standards for native app presentation. Furthermore, mobile devices such as iPhone and Android can also integrate JavaScript and HTML5 functionality into native applications through web views.

2.2 CSS

Having HTML without CSS is an extremely serious omission. So, what is CSS?

Cascading Style Sheets, commonly known as CSS [7] [4], is a language used for creating style sheets that determine the visual presentation of a web page. It enables the addition of various design elements, such as colors, effects, formatting, and fonts, to enhance the overall appearance of the web page. A CSS file is a type of file used by web browsers to display the content of a web page created

using HTML. Each website has its unique interface design, including the use of colors and effects on buttons and cover pages. CSS enables the creation of visually appealing web pages by allowing for greater flexibility and control over the presentation of the content. Additionally, CSS helps improve content accessibility by providing a separate file for formatting, reducing complexity and repetition within structured content. This also enables multiple sites to share formatting by using the relevant CSS in a separate .css file.

In one CSS file, we can adjust for all pages, like want to change the background color for all web pages, fonts, etc. Depending on how the developers want to build the website. But usually, we should split each file separately with each different className to be easier to manage, if there is an error. In case, there is an error that the CSS file we put in the same file, it will be very difficult to find the error and find out where the error is located.

The interesting thing about CSS is that it gives users the freedom to be creative when using CSS, when users use CSS, users are like "flying in the clouds", because users can think of countless signals application, color, and size for a web application. It is this that has motivated developers to create frameworks specifically for CSS with the desire to create functions and provide pre-made color data to users so that simple web applications can be created. Simple, and fast but still gives the project colorful user-generated creativity. CSS3 can be seen as the latest version of CSS.

And to better understand the CSS "Universe", users should understand how CSS works, its features and benefits. Once we understand how CSS and HTML work, we are free to build the most colorful and beautiful website depending on our creativity or based on the templates that the designers have drawn.

2.3 JavaScript

Besides HTML and CSS, JavaScript [8] [9] is a widely-used programming language in web development, used by the vast majority of websites and supported by modern web browsers on various devices. Its success has extended beyond the browser, thanks to NodeJS, making it the most commonly used language among developers. JavaScript [10] is now utilized to create full-stack applications, and the European Computer Manufacturers Association (ECMA) is the committee responsible for managing the development of JavaScript. If a user wants to build an event for his website, then JavaScript is a language that helps to bring that to the user. With the convenience of JS brought, many JS frameworks have been born giving many users, diversity in construction, comfort in development, and freedom in creativity. If user want to make a countdown timer or a carousel for the homepage, and so on. JavaScript will be the language that can help us do that the user will have to understand how it works.

JavaScript [8] [11] is a dynamic programming language for creating content for web pages and is one of the most used languages in the world. JavaScript is integrated into HTML files and created to increase interactivity and improve website performance. With JavaScript [11], users can enhance web page interactivity by manipulating the content and markup while it is being viewed in the browser. This results in various effects on the webpage, including dynamic background colors, animations, and more.

Besides, it helps users easily build websites with this language. Then many other individuals or developers such as Facebook, Google, etc. in turn launched famous frameworks such as ReactJS, and AngularJS, with the hope that users will be able to build websites faster and omit some complex structures and functions, helping users learn and access faster. However, although the framework was born to help users to access it faster, users also have to learn about the basic operating principles of JavaScript, so that they can easily approach the Framework without being surprised.

2.4 Front-end Framework

In simple terms, in web design, the front-end [12] is created by three basic languages, which are HTML, CSS, and JavaScript. Programmers are not just web designers but must make the designed web applications compatible with other types of devices and screens, possibly different operating systems.

Front-end frameworks are packages of pre-written code to apply. It's like a coding dictionary made to help users quickly perfect their application without having to code it themselves. In the front-end framework, two types of frameworks that can be used together to optimize the application, product and user interface. Those are the CSS framework and JavaScript framework.

When it comes to Front-end development, developers often think of the famous trio of HTML5, CSS, and JavaScript, and a lot of focus is given to Front-end frameworks. Utilizing a Front-end framework can help in speeding up the development process, reducing development time and costs, and ensuring consistency and ease of maintenance during web application development.

2.4.1 CSS Framework

CSS Framework [13] [14] is a set of CSS source code that has been pre-written with certain functions and declares each function in a separate class, programmers or designers will easily apply the functions to the product, or application by adding the class of the element they want to use on the element they need to apply it to, for example adding a style to a button or menu.

In CSS, there will be 2 types:

- CSS Framework: Bootstrap, Foundation, Tailwind, Pure, etc.
- CSS Preprocessors: SASS, LESS.

Numerous CSS frameworks, including Bootstrap and Tailwind CSS, are currently in high demand and widely utilized by many developers for various projects. Employing a framework can assist in streamlining the process of styling elements on a website, resulting in aesthetically pleasing and polished designs. Additionally, frameworks ensure that websites are responsive and display properly across a range of devices, from desktops to mobile phones.

However, the use of frameworks also has some limitations. One of those limitations is that websites built on frameworks can seem similar and not unique. Frameworks can also be large and make web pages load slower.

CSS Framework is the result of success in Web Development because it brings convenience, and simplicity to use. CSS is also the result of "creativity", developers have understood the needs and wants of users when using CSS. Because these frameworks help users to apply to websites faster and more optimally. However, they also need the user to know basic CSS, so the user won’t be surprised when the user encounters new methods.

In conclusion, CSS Framework can be a useful tool for web developers in speeding up development and helping to create responsive and beautiful websites. However, the use of frameworks should also be considered to ensure that the website is unique and has good performance.

2.4.2 JavaScript Framework

Like CSS Framework, JS Framework [15] [16]is a pre-written code, it contains compilers, interpreters, libraries, etc. Users only need to understand the operation rules and call out the functions for easy use. The benefit of using JS Framework is increased overall efficiency. JS Framework gives the project more detailed structure, ready to provide common programming solutions. JS Framework has many

different types of libraries and attracts a lot of users and developers like Angular, React, Vue, and so on.

JavaScript Framework is one of the most loved programming languages because it has simplified problem-solving around JavaScript, making it easy for users to handle difficult situations, concisely, easily and as fast as possible. With the combination with many other libraries, JavaScript Framework has brought other advantages and optimizations, making the JavaScript community gradually become more crowded and creating many other Frameworks over time. They will make JavaScript Framework become more and more developed.

However, the use of frameworks also has some limitations. One of those limitations is that applications built on frameworks can be difficult to customize and can have security issues. Frameworks can also be large and need to be downloaded before they can be used, which can make websites load slower.

In summary, JavaScript Framework can be a useful tool for web developers in speeding up development and helping to create complex and unique web applications.

Below are the functions, information about HTML, CSS, JS and some frameworks that come with it, because users have to understand what they are, and what they can do for Web Development. Because when users understand this basic information, users will be able to choose frameworks that are suitable for themselves, in accordance with their abilities.

2.4.3 ReactJS

React[2] [17] is a JavaScript library that allows for the development of dynamic web applications with quick response times to user input. It was created by Facebook as an open-source engine in 2013 and has become more popular than Angular and Bootstrap, which were previously the top-selling JS libraries. React [18] gives developers the freedom to structure their applications according to their own preferences without forcing specific tools or patterns for building applications. It is a tool for building UI components that makes it easier to develop large online applications that can modify data without requiring a page refresh.

React's primary focus is on speed, scalability, and user-friendliness. It works only with the application's user interfaces, similar to the view of the MVC template, and can be integrated with other JavaScript frameworks or libraries like AngularJS in MVC.

React is also one of the most used and downloaded JS libraries, as well as one of the programming libraries that are requested by many companies to recruit in some countries with high salaries and is also widely used because of the features it offers.

React [17] [15] consists of 2 prominent parts: ReactJS and React Native.

- ReactJS is used to build websites on computer environments: Windows, and MacOS.
- React Native is used to build websites on mobile environments on many operating systems: Android, and iOS.

Both technologies contribute to bringing users the latest functions and their convenience. And React is also rated as one of the most used because it always has a solid community that always supports users and React's usage will not cause difficulties for users. So, what is React? What features does it have? What are its benefits to users? These questions will be analyzed and answered below to help readers have a good view and can choose this framework to apply to their web projects.

[2] https://react.dev/

React's [19] features are many, but here are a few outstanding features of React those users can use or here are outstanding features in React.

- *JSX:* JSX is a language that enables writing HTML in JavaScript, providing faster execution and better security. When compiled to JavaScript, it undergoes optimization resulting in faster execution compared to writing code directly in JavaScript. It is also statically typed, like Java and C++, which detects errors during compilation, improving security. Additionally, it provides good debugging capabilities. However, web browsers cannot understand JSX directly, so it is necessary to convert JSX to JavaScript before it can be understood by the browser. And that's where Babel comes in. Babel lets you take the latest JavaScript features, including JSX, and convert them into classic JavaScript that browsers can understand.

Figure 1: JSX Code style

- *Components:* React is centered around building websites using small, reusable components, rather than templates like other frameworks. Each component has its own mutable state and React updates components based on state changes. This approach helps to maintain code when working on large projects. A simple React component requires only a render method, though there are other available methods.
- *Virtual DOM* [17]*:* React JS use Virtual DOM technology to improve application performance. The JavaScript object contains all the necessary information to create a DOM, and it calculates changes when data is modified. This optimizes the re-rendering of the real DOM tree, saving processing time and resources. It is especially useful for large and complex websites like e-commerce and food ordering, improving customer experience and performance.
- *Redux:* Redux is a tool for managing the state of JavaScript applications in a predictable manner. It is designed to make applications that run reliably across various environments and are easy to test. As a result, it is frequently paired with React.
- *Cross-Platform Support:* Frameworks such as React Native and Electron allow for the development of web, mobile, and desktop applications using ReactJS.
- *Community support:* ReactJS has a large and active developer community that helps support and contribute to the development and improvement of the library.
- *Testing:* ReactJS provides tools for unit and integration testing, which helps to ensure the stability and reliability of the application.

ReactJS and React Native are both frameworks developed by Facebook for building user interfaces, but they have some fundamental differences. There are the distinctions below:

ReactJS [17]:

- Original: ReactJS is a JavaScript library created by Facebook to solve performance and compatibility issues for web applications. ReactJS was born in 2011.
- Development Tools: In ReactJS, developers need to choose the best and most necessary modules before starting a project.
- HTML: ReactJS is usually rendered using mainly components that contain specific components or HTML tags
- Security: High

React Native:

- Original: React Native is a framework used to develop applications for mobile (specifically Android and IOS). Released 4 years after ReactJS.
- Development Tools: React Native provides everything needed. Users can instantly write an app in React Native with just one command line.
- HTML: React Native does not use HTML to display the application interface. Instead, React Native provides components that do the same thing, and most of them can be transformed into their HTML equivalents.
- Security: Lower than ReactJS.

ReactJS and React Native are two distinct technologies that share the same underlying React library and utilize JavaScript. ReactJS is used to build dynamic web applications, while React Native is used to build native mobile apps for iOS and Android platforms. However, the common source code base allows developers to use both technologies together for synchronized web and mobile app development.

As we all know, React is a JavaScript library that helps in UI design. But not everyone knows all the tools that will be introduced in this article, which will help users have a much more enjoyable React experience.

- React-proto: React-proto is a desktop prototyping tool for programmers and designers. Users can declare props and types, view components in a tree, add backgrounds, and define stateful/stateless components and parent components. It also allows zoom in/out and export to new or existing projects.
- React Developers Tools [10]: React developer tools are a utility that allows testing the React Component hierarchy in Chrome and Firefox developer tools.

React [20] is a highly favored option for front-end web app development due to its exceptional capabilities and advantages, making it a top choice among many developers Although there are some disadvantages, React's community is continuously working on overcoming these issues in upcoming updates. If developers are searching for the next big thing in the framework world, React is definitely worth considering.

One of the advantages of React [17] is its ability to reuse components. This feature enables developers to extensively use pre-optimized components and integrate them into other applications with the same functionality. Additionally, React has a large community that can provide maximum support for users who encounter difficulties in the development process.

React is recognized for its straightforwardness and user-friendliness. It works based on a combination of HTML and JavaScript, making it simple and easy to use for developers who have mastered JavaScript knowledge. JSX files used in React also make the application simpler and easier to understand.

Another advantage of React is that it is made by great developers from Facebook, ensuring that it is supported and built by many developers behind it. This means that it will be widely used by the community because of the support from back-end developers.

Facebook has released a Chrome extension for debugging during application development, which speeds up the product release process and coding process. In conclusion, React's numerous advantages make it a popular choice among developers for front-end web application development.

React, like any other framework, has its advantages and disadvantages. While React's superior features bring many benefits to users, it also has some drawbacks that developers should be aware of.

One of the challenges of using React is its rapid and continuous updates [20], which can be difficult for developers to keep up with. With each update, there are changes in programming that require developers to change their way of working, making it challenging to access and update continuously.

React's use of JSX syntax may pose a challenge for some developers, who may prefer basic JavaScript instead. It can also result in lengthy source code. Moreover, React is solely a view library, lacking a model or controller component that's found in other frameworks. It must be used with other libraries for these features. Additionally, React lacks 2-way binding and Ajax support.

Compared to other frameworks, React is quite hefty, which can impact application performance. Furthermore, developers should be aware that React only addresses the application's user interface layer, so they must still pick a framework to create other parts of the project.

In conclusion, while React has some disadvantages, developers can overcome them by being aware of these issues and staying up-to-date with the latest updates and best practices.

By analyzing React, users can grasp its features, benefits, and drawbacks. React has significantly contributed to the web development community and is expected to become the future of open-source JavaScript. With continuous improvement and development, Facebook is committed to making React more efficient and advanced. Users can look forward to a future version of React with new renders, unique JSX syntaxes, and better error-handling capabilities.

React is continuously improving and advancing with community and developer efforts to overcome its weaknesses. Users can access React's documentation page to stay updated with the latest patches and updates. ReactJS is a potent JavaScript library for building web applications with efficient component management through Virtual DOM and Lifecycle methods.

With its focus on component-based development, React has gained widespread popularity as a flexible framework for building entire applications through a hierarchy of components. Although the modular design of React may present challenges for beginners and sophisticated applications may require additional programs and familiarity with functional programming, React's [21] slim API, stable yet thriving ecosystem, and welcoming community have made it the number one UI building solution today. Choosing React requires basic knowledge of JavaScript and client-side programming but can provide numerous benefits for developing websites and apps.

2.4.4 Bootstrap

Bootstrap[3] [22] [23] is a framework for web development that is both free and open-source. It was designed by Twitter and is used for quicker and more straightforward web development. Bootstrap includes the design of HTML and CSS such as fonts, buttons, and images, including the designs of mobile devices. Bootstrap also gives users the ability to build functions, and classes suitable for individual needs.

Like any other Front-end Framework, Bootstrap is a front-end framework that contains HTML, CSS and JavaScript components. It follows web design standards and enables developers to create responsive

[3] https://getbootstrap.com/

websites that can display on multiple devices. Its advantage lies in its large community and documentation, allowing developers to find solutions to any issues they may face.

In the field of Web Development, Bootstrap is considered one of the most important tools for creating modern, professional and flexible websites. With Bootstrap, developers can create responsive websites with ease, while optimizing the user experience on a variety of devices. Bootstrap helps to reduce the time and effort required to create beautiful and efficient user interfaces, allowing developers to focus on other website features and functionality.

Bootstrap [24] is a highly popular and widely used CSS framework that simplifies website design and development, offering a wide range of UI components to ensure consistency in design. It supports customization features and integration with other tools like JavaScript, making websites more functional and flexible. Its many features have contributed to its fame in the field of CSS frameworks.

- *CSS Custom Properties:* CSS custom properties make CSS dynamic and programmable. CSS variables are prefixed with -bs to avoid conflicting CSS of the parties.
- *JavaScript Plugins:* Bootstrap provides several JavaScript plugins to help increase the interactivity and dynamics of the user interface. These plugins include a carousel, modal, dropdown, and tooltip.
- *Documentation and Community* [22]*:* The document is informatively updated with a community that has responded to these updates. The information on the user manual, how to add components, etc. has been edited and unloaded more clearly. When users encounter difficulties, the Bootstrap community is available to answer and support users and developers.
- *Typography:* Bootstrap provides a set of fonts and typefaces optimized for use on different devices. Users can easily customize the font and font size to fit their needs.
- *Color:* The palette in version 5 has been expanded. Developers will have more options; users will be able to style the color they need. The colors related to the contrast index are also updated in this version.
- *Icon:* Bootstrap has over 1300 icons with an open-source SVG library. Developers have a lot to choose from for their projects. Icons are very necessary for many projects; Bootstrap has responded very well to developers when gradually updating many of the most popular Icons for users.

With the above features, Bootstrap is a particularly useful tool to help users create responsive and interactive user interfaces in their web applications.

Bootstrap is a highly regarded front-end development framework, thanks to its many advantages. It provides an easy-to-use platform for those new to CSS, allowing users to apply pre-designed styles without having to write a CSS file from scratch.

Bootstrap also boasts cross-browser compatibility, making it accessible on a range of browsers, from Chrome to Firefox. Furthermore, Bootstrap has excellent documentation for both new developers and experienced users looking to expand their knowledge. This includes extensive style content that is relevant for most websites.

Bootstrap's responsive design is also a major benefit, as it can adjust and adapt to a range of device screen sizes. Additionally, the Bootstrap CDN makes it simple for users to get started, as they only need one link to access the framework. Bootstrap's grid system provides a range of layouts and components, allowing users to make quick and easy changes to suit their preferences.

As a cross-browser-compatible framework, Bootstrap ensures that websites perform well on different browsers. Bootstrap also prioritizes website security, with features such as Cross-Site Scripting (XSS)

and Cross-Site Request Forgery (CSRF). Finally, Bootstrap supports SASS, a CSS structural language that makes it easier and quicker for users to customize CSS.

Despite the various benefits that Bootstrap provides, developers must also take into account some limitations of the framework. One such limitation is the limited customization options that it offers. Although Bootstrap provides numerous predefined classes, UI elements, and features, developers may need to write custom CSS or create new classes to achieve a desired level of customization beyond these options.

Another drawback of Bootstrap is its large file size, which may pose a challenge for some users. With numerous built-in CSS, JavaScript, and HTML classes, the size of the Bootstrap file can exceed that of other options, slowing down the website and placing a strain on the server.

Additionally, overriding Bootstrap's parameters can be a challenging task, as developers must create separate parameters with the same name and override the Bootstrap parameter. This may result in duplicates in many other CSS files, which could affect the professional look of the site.

Furthermore, the complex structure of Bootstrap can be confusing for some developers, as some components may be nested inside other components, making it challenging to locate and modify elements within those components.

Another concern when using the entire Bootstrap framework is redundancy, as the framework provides numerous functions that may slow down the website and burden the server. As a result, users should only use the necessary layers to minimize file size and server load.

Finally, while Bootstrap is user-friendly, developers must have a basic understanding of CSS and invest time in learning about the components available in Bootstrap. Without prior knowledge of CSS, learning Bootstrap may be more challenging than other frameworks.

Bootstrap has become increasingly popular among developers in the field of web design [22] due to its convenient features and efficiency in saving time. By eliminating the need to manually write HTML, CSS, and JS code, developers can quickly create professional and responsive websites using the framework's versatile design templates. Bootstrap has become an essential tool in both Frontend Development and CSS framework and has streamlined the website development process for many developers.

2.4.5 NPM

Modern software development commonly utilizes third-party libraries [9] for significant programs, such as the Express framework for Node web servers and React, LitElement, or Angular for web browser UI design. Node Package Manager or NPM[4] [25] [26] [27]is a tool for managing dependencies such as modules, packages, and libraries for NodeJS applications. It comes as a part of NodeJS and is installed along with it. Developers can use NPM to manage the packages required for their NodeJS applications.

NPM provides a vast library of modules and packages developed by the NodeJS community, allowing users to easily reuse existing code. With NPM, users can install packages through the command line or the project's package.json file. NPM also offers other useful packages for ReactJS development, such as react-dom, redux, and react-router-dom. NPM simplifies package and dependency management in ReactJS application development, making it easy to install third-party libraries or packages that an application needs. Users can install local modules for their application or globally shared packages like React, Express, or Axios using the "npm install" command.

[4] https://www.npmjs.com/

The files package.json and package-lock.json are two important files related to NPM. Inside the package.json file, there is data about the application and its dependencies, while the package-lock.json file provides more comprehensive details about the application's dependencies, which includes their versions and any sub-dependencies. Both files need to be pushed when users share their apps with others.

In addition to installing packages, NPM also allows users to manage different versions of packages and resolve dependencies between them. It is one of the most popular package management tools in the NodeJS development community.

NPM [25]is package and module management tool in NodeJS. It offers many useful features, including:

- *Manage dependencies*: With NPM, managing dependencies for their application becomes possible, where installation of the required dependencies can be done easily by indicating the package name and desired version. Additionally, NPM enables simple updating of dependencies.
- *Install third-party packages:* NPM allows one to download and install third-party packages. It allows one to search and install these packages from the NPM repository (NPM registry) easily.
- *Version management:* NPM allows one to manage the versions of their packages and modules. They can easily update and manage different versions of their application.
- *Share code with the community:* NPM provides an NPM public registry that allows one to share their code with the NodeJS community. They can register and publish their packages on this NPM public registry.
- *Develop their own modules:* NPM allows one to develop their own modules. They can register these modules with the NPM public registry or use them in their application.
- *Manage scripts:* NPM provides a special function that allows one to define and manage scripts in the package.json file. This allows them to run commands easily without having to retype long and complicated commands.
- *Cross-platform compatibility:* NPM is able to function on a variety of platforms, such as Windows, macOS, and Linux. Furthermore, it is compatible with numerous versions of NodeJS.

NPM is a tool that simplifies dependency management for NodeJS projects. It provides an online repository of pre-written packages, libraries, and applications that saves developers time and effort. NPM supports version management, making it easy for developers to track and update dependencies. Developers can also create, publish, and share their packages with the community. Using NPM helps developers save time and effort in developing NodeJS applications.

Although NPM has many benefits when used in the development of NodeJS applications, there are also certain disadvantages.

Using NPM has some disadvantages. Complex dependencies and links can increase application complexity and cause errors. Third-party packages can create security holes, leaving the application vulnerable. Managing dependencies can become complicated for large projects. Errors during package installation can interrupt development and cause difficulties for developers.

As such, NPM is a very useful tool in the development of NodeJS applications. With NPM, users can easily manage dependencies, install third-party packages, and share their modules with the NodeJS community. However, like any other tool, NPM also has certain disadvantages, including dependency and security issues, complex dependency management, and package installation errors. However, with the right tools and processes, these disadvantages can be effectively addressed so that users can use NPM safely and effectively.

2.4.6 Other Front-end Framework and Library

These are some of the frameworks that have been analyzed and are popular among developers for building web applications. There are also many other versions of frameworks [28], such as Solid, Preact, Material UI, and Semantic UI, that have been released with their own unique features and benefits. Frameworks are designed to provide users with convenience, allowing them to build web applications quickly with minimal effort and limitations. When choosing a framework, users should consider factors such as benefits, weaknesses, features, and usage. These are the four basic factors to consider when deciding which framework to learn. Many frameworks are currently being studied, and it's possible that more versions will be released in the future with simpler features and advanced functionality. Ultimately, website performance is a critical aspect of web development.

With the relentless creativity of developers, frameworks and libraries are released continuously so that users can easily use methods and features without having to spend too much time designing themselves.

There are many statistical sites such as Google, Github and so on. All give ranking data on the number of users, efficiency, and speed. It is important that users and developers have contributed to the development of existing frameworks and create many more frameworks so that users can easily use and create more complete websites.

Furthermore, consumers can be confident that users can be proud of the effort the developers have put into creating and developing the framework after reviewing the statistics and usage data from the users in recent years.

Ratios Over Time is a form of statistics commonly used to analyze the changing trends of Frameworks over time. Typically, Ratio Over Time can be used to compare the change in market share of Frameworks over a given period. Thanks to Ratios Over Time, users can evaluate the popularity of Frameworks over time, thereby making decisions about choosing the most suitable Framework for their project. In addition, the Over Time Ratio can also help users predict the future development trend of Frameworks.

These are the Ratios Over Time statistics of the most used JavaScript and CSS frameworks as of 2022.

Figure 2: Ranking using frameworks of JS[5] and CSS[6]

2.5 Why HTML5 is important in Web Development?

Web development technology [29] is always developing. 60% of developers, on average, have already utilized HTML5 to create significant projects, according to data. Today, less than 20% of native mobile developers still rely on old-school technology. That HTML5 is evolving into the preferred platform is clear from this. When HTML5 development surpasses all other native development techniques, it will only be a matter of time.

Multimedia applications of all kinds are now possible thanks to the development of HTML5. It supports animations and plays audio and video without the need for any browser-specific proprietary methods. The new functionalities offered by HTML5 would add fresh value for site designers and developers.

Cross-platform support offered by HTML5 enables the display of webpages on devices such as Smart TVs, tablets, PCs, smartphones, etc. HTML5 elements are being used by a huge number of websites and browser developers. The key allure for web designers and browsers is that it's possible to make advanced forms, web-based applications, and rich web pages without learning or paying for a variety of proprietary approaches.

HTML5 enables quicker web page load speeds, enhanced caching, and lower server load, which can result in greater performance and a better user experience. Because HTML5 makes coding simpler, developers can build more complex web apps with fewer lines of code and less reliance on third-party libraries and plugins.

There will always be those who think more significant platforms require native mobile applications. HTML5 offers developers more flexibility, functionality, and performance, making it an essential tool for modern web development. Most developers, however, are now opting to join the "working smart and not hard" movement. The majority, if not all, mobile platforms will eventually demand the use of a web application. This is essentially the basis for why HTML5 is seen as a key tool in future web development.

[5] https://2022.stateofjs.com/en-US/libxsraries/front-end-frameworks/
[6] https://2022.stateofcss.com/en-US/css-frameworks/

3 CASE STUDY

The main concept is to build an e-commerce website to grow the business with the same pages and functions as other e-commerce sites like Footlocker, ZARA, Coolmate, etc. However, the focus is on the Frontend part, which is the user interface. The Backend is used only for the role of responding to some server-side requirements for convenience in construction.

ReactJS is chosen as the main framework for designing the website. ReactJS is one of the frameworks used and developed by many developers. Besides, ReactJS is also one of the languages preferred by companies and employers to hire and pay a higher salary than other frameworks. To make it easier to build concepts, dividing components is an important part of the functionality that ReactJS provides to users, so it becomes more convenient to come up with ideas, apply and build them.

The case study is based on popular or popular clothes management websites such as FootLocker, ZARA, UNIQLO, etc. So, to complete a complete project, some of the features below will be referenced and introduced for thesis implementation.

There is a multitude of e-commerce websites in existence, each with its own distinct features and designs. It is possible that some of these features have been borrowed from others. The forthcoming product may have some errors, but every attempt will be made to correct and enhance it.

Before delving into the Case Study segment, I would like to present the technologies and features that will be employed in the project. These technologies have a significant role in the project and are based on the case study for its completion.

The first thing we see on an e-commerce website we need at least the following pages:

- Home Page: For customers to have an overview of the e-commerce website, hot product models, new products, most purchased, and so on.
- Product Page: The product page will include the products of the shop, we can filter the products by name, rating and price.
- Product Details Page: The page will have product information, add to cart section.
- About Page: The page will talk about what our shop has, what to bring and the shop's offers.
- Contact Page: this page will have the shop's address and a contact section for users to contact the shop.
- Cart Page: The shopping cart page will have the products that the user has ordered, and the order section and the successful checkout page will appear.
- E-commerce, Size Guide, FAQ, Ship Policy, Privacy Policy, Return Policy Page: These are information pages about policies, and guidelines, providing customers with information about the store.
- Login and Register Page: For users to register and log in.

3.1 Features and Technologies

This section will analyze the tools, frameworks, and libraries that will be used in the thesis product.

The two main frameworks of CSS and JS, ReactJS is one of the famous libraries and is used by many developers with good patronage of Facebook. So, one can use it to learn and develop skills for themselves, besides CSS Framework is Bootstrap, and these two frameworks have been analyzed above. The above and below only analyze the content that has not been described during use.

3.1.1 Feature

The features that will be built into the project are the ones that have been adopted by many e-commerce sites into their sites.

- Sign in, sign up, sign out.
- Search product.
- Dark/light mode.
- Localization
- Filter
- Change product details in the cart (Add or remove product in the cart).
- Enter payment details, including customer's information, and other required information.
- Quick Buy
- Responsive

3.1.2 Technologies

- **ReactJS**
- **Bootstrap**
- **JQuery**[7]

jQuery [30] [11] is a JavaScript library that streamlines the client-side HTML scripting procedure. It offers a range of features, including simplifying HTML document manipulation, animations, event handling, and AJAX. With an intuitive API that is compatible with various browsers, jQuery makes these tasks easier to perform. In addition, jQuery is also used to add animations. jQuery can also be combined with other libraries to create complex and customized web applications according to the needs of each project. So, it is a useful tool for web developers to reduce development time, increase interactivity and improve user experience on their websites.

- **NodeJS**[8]

NodeJS is an open-source [31] [6] runtime environment for JavaScript that allows it to be used on multiple platforms. Its purpose is to run JavaScript code outside of web browsers. NodeJS [9] is essentially JavaScript with links to the operating system underneath, enabling JavaScript programs to read and write files, perform child process executions, and communicate via networks. NodeJS provides a server-side JavaScript execution environment that enables developers to quickly and efficiently build server-side web applications. Furthermore, it furnishes libraries and modules that facilitate various tasks, including file manipulation, data exchange with APIs, and database interaction. NodeJS is a versatile JavaScript runtime environment that is extensively employed for building web applications that require real-time functionality like chat apps, online games, and interactive applications. It facilitates web developers to use a single programming language for both the frontend and backend development of the application. NodeJS is an important technology in web application development and is widely used in both client-side and server-side web applications.

- **ExpressJS**[9]

ExpressJS [32] is a web framework designed to be used with NodeJS. Its primary use is on the back-end of websites and web applications. Express is adaptable and minimalistic, so it doesn't impose any rules on how one should create their application or have a large collection of pointless tools and packages. ExpressJS also allows integration with other tools and libraries such as template engines, WebSocket, OAuth and many more. This helps web developers build complex web applications easily

[7] https://jquery.com/
[8] https://nodejs.org/en
[9] https://expressjs.com/

and efficiently. ExpressJS is a powerful and reliable web application framework that helps web developers create complex web applications with great flexibility and minimal development time. When building a web application with NodeJS, the ExpressJS [33] framework can serve as a web framework for API routing and middleware. Express allows the inclusion of various middleware in the request-handling chain in any order, providing great flexibility. ExpressJS is a widely utilized web framework within the NodeJS community, and it is popularly employed for creating web applications and APIs.

- **MongoDB**[10]:

MongoDB [34] [35] is a NoSQL database management application. NoSQL databases provide an option to conventional relational databases that use SQL (Structured Query Language). MongoDB stores data as documents with a structure similar to JSON, rather than in tables as in a relational database. MongoDB allows for more flexible and easily extensible structured data storage and querying than traditional relational database management systems.

MongoDB supports many features, including full-text search, indexing, aggregation framework, and MapReduce. It is also scalable and handles big data with replica sets and sharding. MongoDB is also well integrated with popular technologies and languages such as NodeJS, Python, Ruby, Java, C++, C#, and many more. By using MongoDB as the database [33] for a Node and Express web application, developers can create a fully JavaScript-based, standalone server-side application. This allows for the integration of a client-side interface using a compatible frontend library like React to create a full-stack application. Data is represented and interacted with using documents in MongoDB, which uses a JSON-like format. With powerful features and good scalability, MongoDB is one of the widely used non-relational database management systems for modern web and mobile applications.

- **Redux**[11]:

Redux [30] [10] a library based on Flux, has become a popular choice among similar libraries. It was created to address the difficulty of comprehending how data changes travel through your application. The developers of Redux, Dan Abramov and Andrew Clark, have been recruited by Facebook to join the React team after the library's creation. Redux solves complex state management problems in large apps. It helps write consistent, flexible, and scalable apps that work across different environments and are easy to test. With features such as middleware and asynchronous actions, Redux is useful in modern web development, not just for React but also for other frameworks or libraries.

3.1.3 How to connect technologies

Depending on each person's construction, users can build NodeJS first to connect to Database and Frontend. Users have many choices other than MERN [36] (MongoDB-Express-React-Node), users can choose MEAN (MongoDB-Express-Angular-Node) to build websites. The project will primarily focus on the Front-end, with React interface being built first. The MERN [33] stack comprises MongoDB, Express, React, and Node, which are commonly used together to develop web applications. The web backend is connected by Node and Express, while MongoDB functions as the NoSQL database, and React is responsible for creating the user interface that users see and engage with. All four technologies are open source, cross-platform, and based on JavaScript, with wide support from the community and industry.

NodeJS [6] is a software system designed for writing scalable internet applications, especially web servers. The application is developed in JavaScript and makes use of event-driven, asynchronous input

[10] https://www.mongodb.com/
[11] https://redux.js.org/

and output to reduce overall resource consumption and increase scalability. NodeJS includes Google's V8 JavaScript engine, libUV, and several other libraries.

The JavaScript community has a vast library of pre-existing code available on NPM. This saves time by allowing new projects to reuse components, libraries, and frameworks without having to start from scratch. Simply by typing "npm install library/package/framework," users can access the vast range of available libraries, including instructions on how to use them. It's important to note that these libraries were created by other developers, so users need to read the instructions carefully to ensure they understand how to use them properly.

The server-side framework ExpressJS, which operates inside a NodeJS server, is the next step down. ExpressJS describes itself as a fast, minimalist web platform for NodeJS, and that is indeed what it is. ExpressJS does have powerful models for managing HTTP requests and replies as well as URL routing (mapping an incoming URL to a server function).

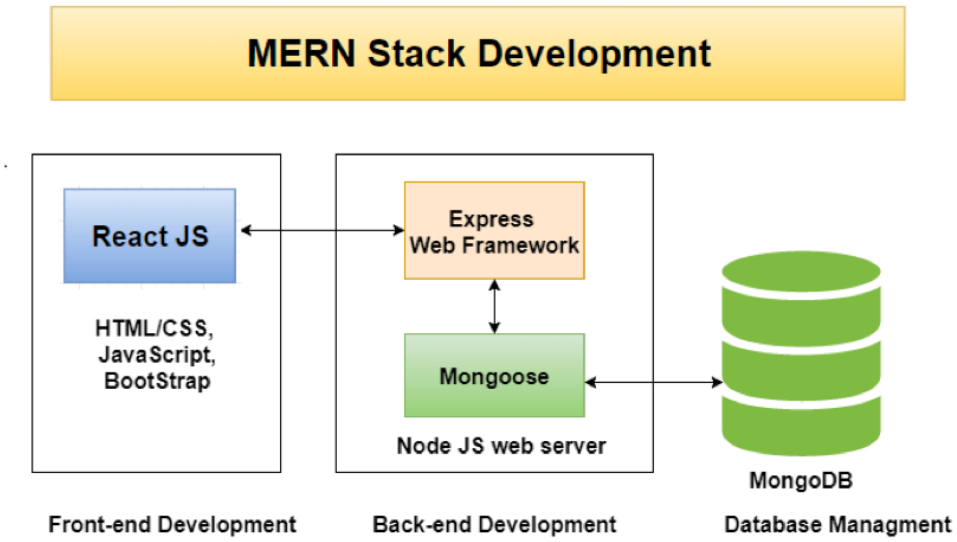

Figure 3: The MERN architecture

Normally, to use Back-end with React, users often use NodeJS to connect with each other. For NodeJS applications, a small but powerful framework that is often used is ExpressJS. A front-end, back-end, and database three-tier architecture may be easily created using the MERN architecture and can be done entirely in JavaScript and JSON.

The JavaScript framework ReactJS, which enables the development of dynamic client-side HTML5 apps, forms the top layer of the MERN stack. With the aid of a few straightforward components and React, developers can create complicated interfaces that can be rendered as HTML and connected to data on the back-end server.

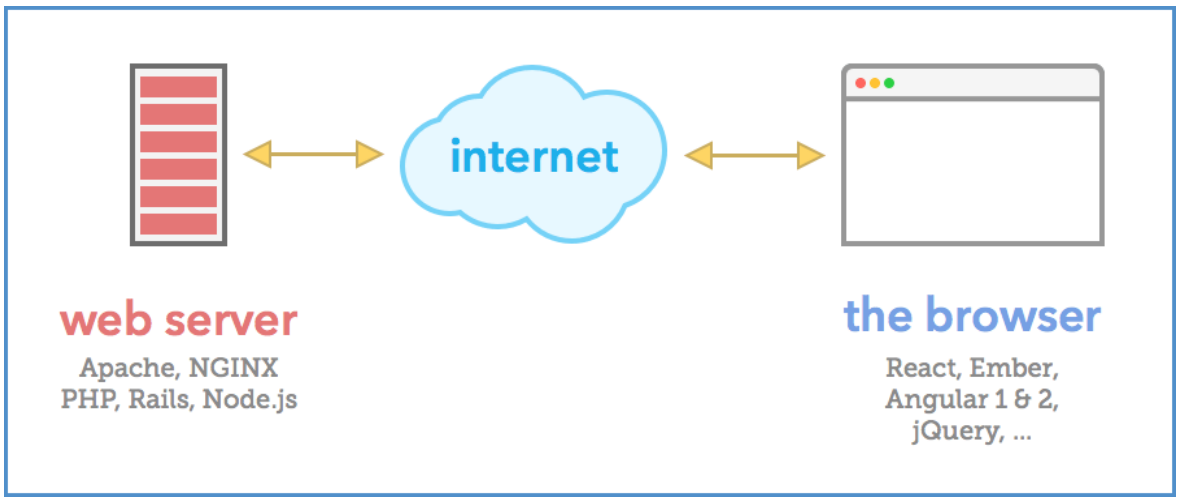

Figure 4: Connect Back-end by NodeJS

React is particularly adept at handling data-driven, stateful interfaces with minimal code and effort required. Additionally, it incorporates all the modern web framework features that one might anticipate.

The React library [37] is a browser-based frontend library. It is glad to be served by any old webserver – Apache – or any kind of backend – PHP, Rails, etc. just like any other frontend library or framework. Because React and Angular are simply client-side libraries made up of JavaScript files, any old HTTP server – PHP inside Apache, PHP inside Nginx, plain Apache/Nginx, Java Tomcat, Rails inside Passenger, and yes, NodeJS – can send them to users.

Sending XML HTTP Requests (XHR), GETs, or POSTs allows ReactJS to connect to the ExpressJS functions that drive a web application. The data in the MongoDB database is accessed and updated by those methods, which in turn use the NodeJS drivers for MongoDB.

If a web application saves any data (user information, comments, images, events, etc.), it will need a database that's just as simple to use as React, Express, and Node. MongoDB can help with this since it allows JSON documents written in ReactJS to be forwarded to the ExpressJS server for processing and, if they're valid, direct storage in MongoDB.

ReactJS offers various ways to integrate with different tools and frameworks. One of these options is JSX, which can replace JS and provide cleaner and more readable code. While JS is still compatible with React, using JSX can enhance the project's appearance and make the code easier to understand. However, if not organized properly, the JS source code can become confusing and more difficult to manage.

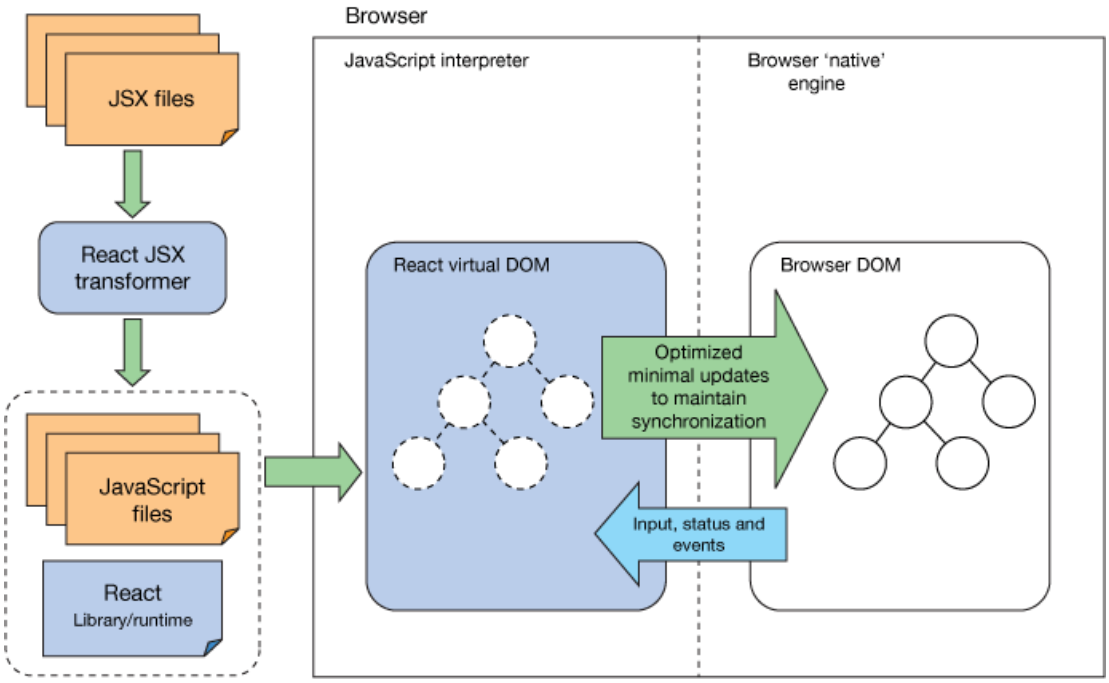

Figure 5: React operations

In the field of technology development, business owners and developers are always looking for the best methods to give their businesses a better competitive edge. And ReactJS is considered one of the top technologies that can provide an advantage to businesses in developing web applications and keeping ahead of the competition.

ReactJS allows businesses to create web applications with better UI to enhance user experience. This is also the technology that businesses need to get higher user engagement, click-through rates and conversions. Moreover, businesses using ReactJS are guaranteed to have a better interface than those using other frameworks because ReactJS helps prevent DOM updates for faster applications and better UX delivery.

3.2 General Features

Since this is a function located on the Header it will follow throughout the project and they do not have a standalone page.

3.2.1 Search

30% of e-commerce visitors use the search bar to find products, indicating high purchase intent. A quick and effortless search process is anticipated by website visitors, particularly on e-commerce sites. As the search bar significantly contributes to the satisfaction of customers and boosts business sales, it has become an almost essential feature for e-commerce websites.

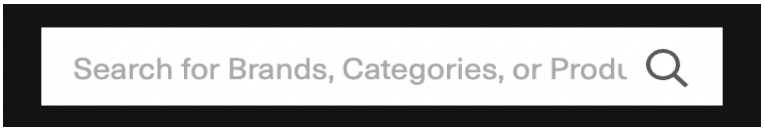

Figure 6: Search Function

3.2.2 Dark Mode

Dark Mode [2] is now prevalent in software, applications, and websites. However, it only became mainstream in web development from 2020 to 2021, with major companies like Google, Microsoft, Apple, and Android adopting the feature. Fashion brands like Nike and Adidas typically do not use it. Dark mode creates symmetry with other light colors and helps users avoid eye strain during prolonged use. It also provides a modern and accessible design for users. Fashion websites may avoid using the dark mode to ensure customers can see products clearly. However, the Dark Mode function is now a crucial feature for optimizing aesthetics and product interface. This function applies to the entire page except for images which won't change color.

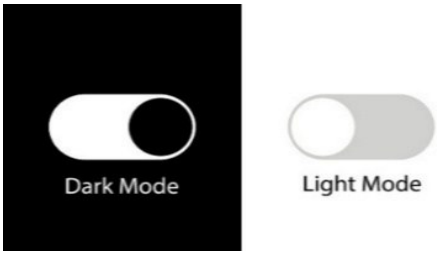

Figure 7: Dark Mode

3.3 Index Page

The index page, particularly the table of contents, is crucial in attracting viewers and customers. Featuring new items, introductions, and discounts can help retain customer interest despite the abundance of options on other websites. A colorful and informative interface can make a lasting impression on customers.

The homepage will feature key components such as Carousel, Latest Product, Best Collection, Banner and Policy Card, inspired by other e-commerce sites. These components will be adapted for dark mode, featuring multiple sections, star ratings, titles, and prices to entice customers to make immediate purchases. After viewing the carousel on the homepage, customers can scroll down to view more notable products for men, women, etc., as well as discounted items for events like Independence Day or Black Friday.

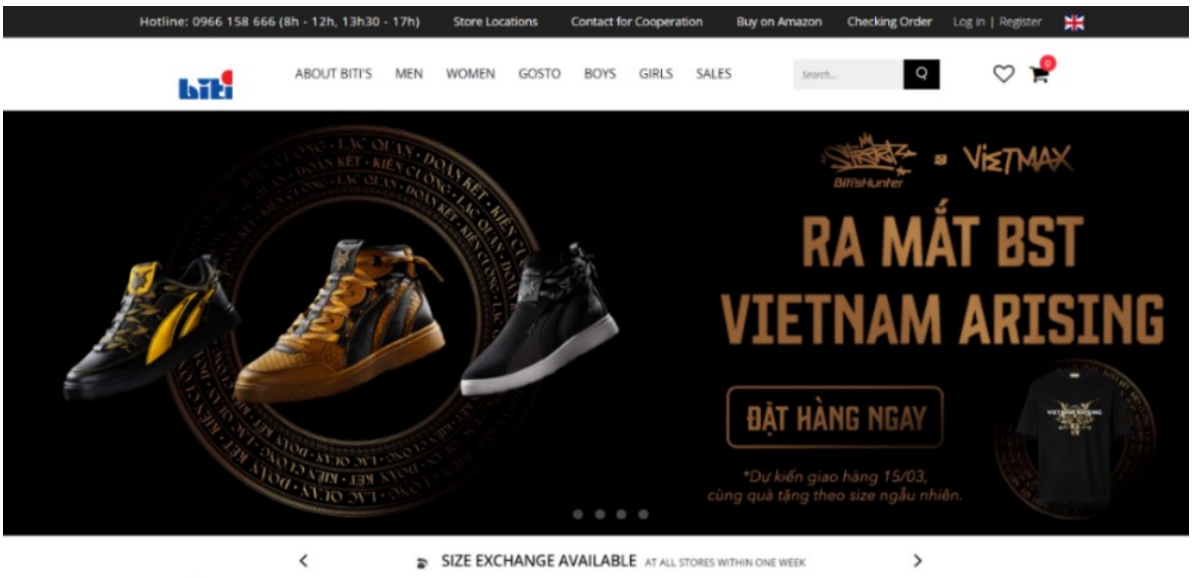

Figure 8: Home Page[12]

[12] https://bitis.com.vn/

In the Carousel section, many of the shop websites that were referred to, utilized designs such as banners, animations (GIFs), videos, or just images and buttons. Only a few sites did not use Carousel which automatically changed banners or images together.

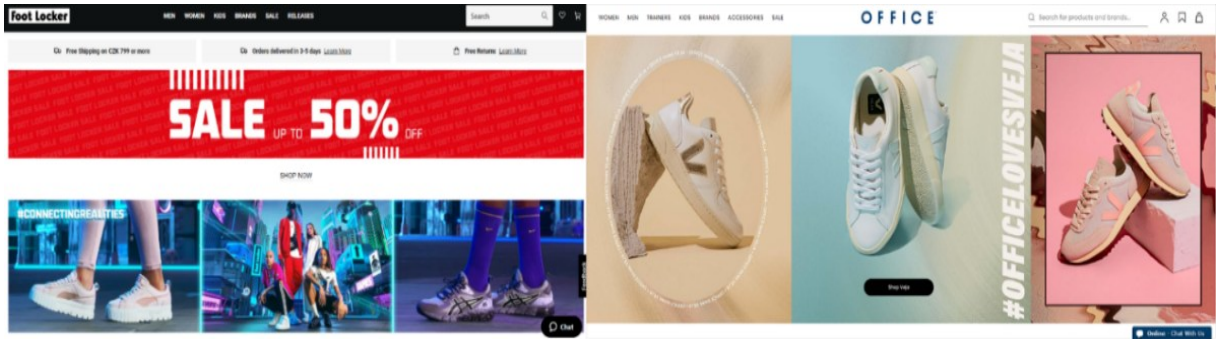

Figure 9: Carousel on Index page

Auto-motion images should be added to attract viewers and give customers more options when choosing the categories, they want. Add left and right buttons so customers may want to see what they will visit, images, when posted will fit into the frame, add effects to buttons to make them more interesting row.

The project involves designing a unique Carousel with animated images, background effects, and buttons that respond to mouse movement. It will move left and right and automatically switch to a new page after a set time, making the website more engaging.

3.4 Product Page

On the product page, customers can choose the items they want to buy. Besides selecting items, users can filter out options to choose products according to their wishes. With many filter types such as Size, Brand, Product Type, etc. there can be many choices to make.

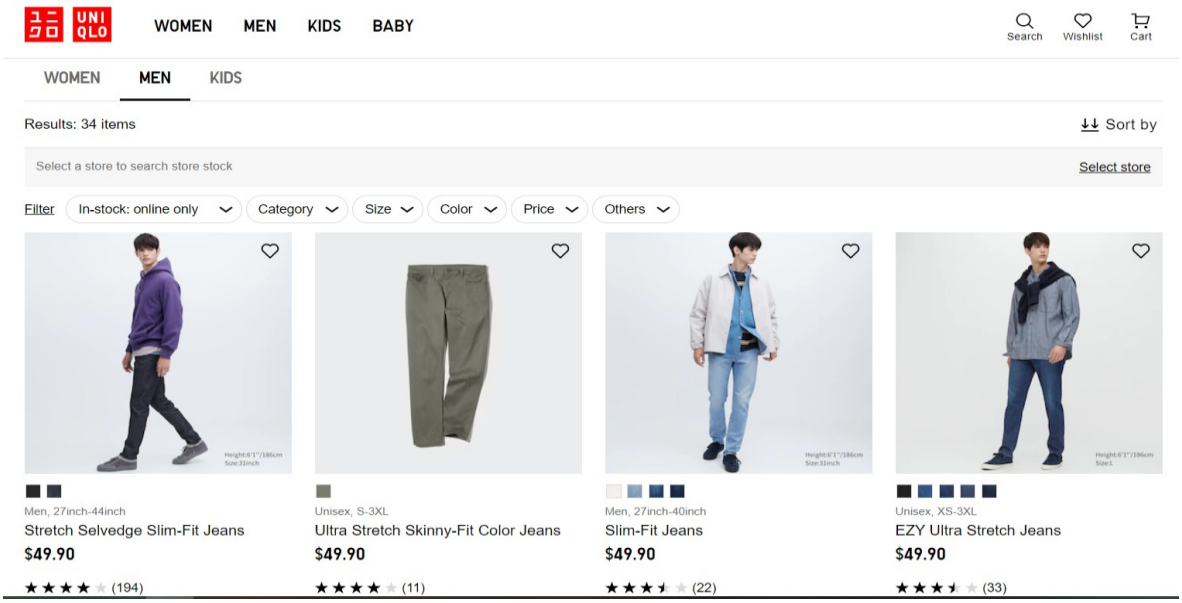

Figure 10: Product Page

The Product Page requires various elements to ensure full functionality and interface. Key components include Buy Now (Quick Buy), Filter, and Product Card. Additional features like Quick Buy and Filter can further enhance the website's convenience and versatility.

3.4.1 Product Card

Product Card is one of the required components when developing a sales website. Depending on the design of each website, each brand will have different Product Cards, such as being able to zoom, convert images, diversify images, etc. These are common effects of a Product Card. Users can click on an image to view product details or can click the Buy button to pay directly to the cart.

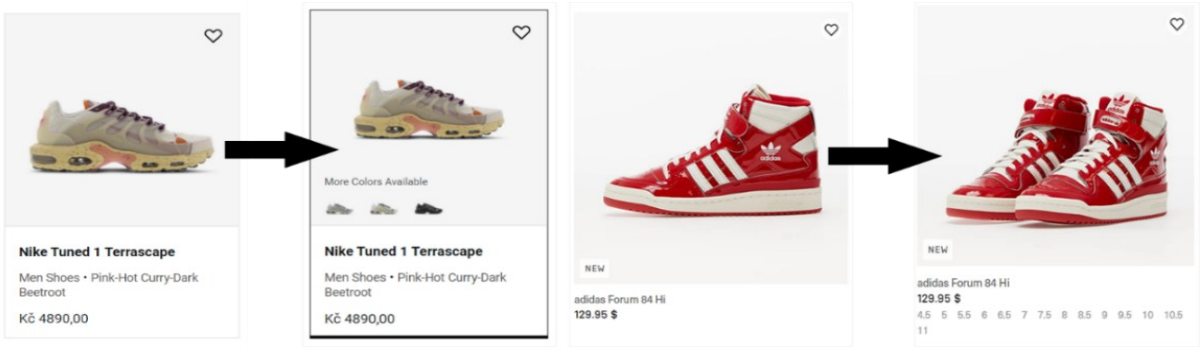

Figure 11: Product Card with effect

The Product Card for my project will include a feature where the first image changes to the second image when the mouse pointer hovers over it. This effect will be achieved through CSS functions. Along with the Button component, the Product Card will also display essential product information such as the name and price.

3.4.2 Quick Buy (Buy Now)

Quick Buy allows customers to purchase a product without visiting the Product Detail page or reviewing details. While it's not a popular feature, some websites offer alternative ways to view product information and add items directly to the cart without leaving the page.

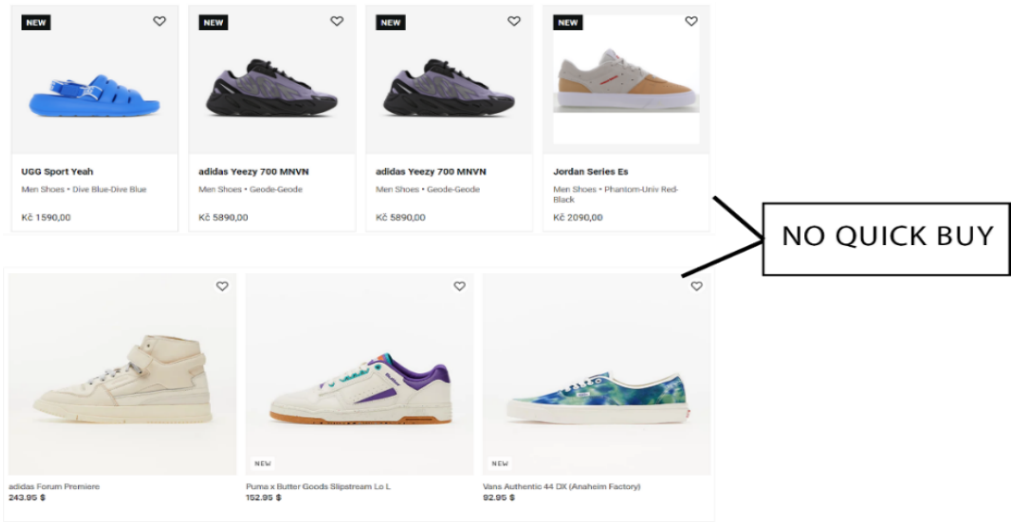

Figure 12: No Quick Buy function on Product Page

In the Quick Buy section, another page will be added without altering any links or impacting other pages. It will appear in front of the screen and blur the Product page, with the same functionality except for Related Products. The design will be simpler for easy and quick access to its features, giving users a full view without having to access the Product Detail page.

3.5 Product Detail

On product cards, users can point the mouse to see image changes to see details and have more options to consider product perspectives.

The Product Detail page offers a user-friendly design, providing a comprehensive view of the product and allowing customers to browse related items. It includes product information like size and color, and users can navigate through additional product images by clicking buttons. This page also offers add-to-cart features for easy purchasing, eliminating the need to visit a physical store.

3.6 Cart Page

After they add products to the cart, the basket will count the number of products that have been added to calculate the quantity that the customer has selected.

The Cart page is important as it relates to money. It shows customers the total cost of their selected items. Payment details, such as product name, size, color, and price are displayed in full for users to review and delete if necessary. Once confirmed, customers can proceed to checkout.

In the Cart page, the functions depend on the association of the Product section, because after clicking add to cart, the system must know how to add the items to the cart and display the Cart page. This forces the system to use Storage to store temporary data. With a good system, diverse, complete, long-term, and secure information can be stored. So, to design the function for the Cart Page, not only the interface but also the function must work smoothly to avoid mistakes.

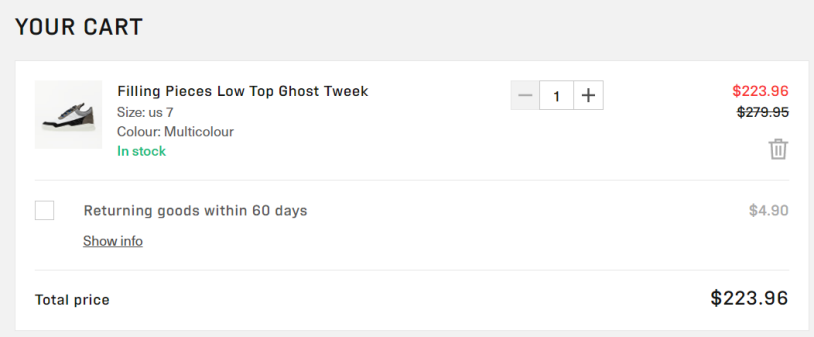

Figure 13: Cart Page

3.7 Login and Register Page

The login and registration pages are two pages that always go hand in hand. Once a user has registered, they will be able to log in. So, these pages do not require a too sophisticated interface, it is important that their function can work, requiring the creation and retrieval of data for these pages to work stably. This depends on the Data side to set up strict data management and the Client side to simply build the interface and connect to the Database to complete the functions.

3.8 About Pages

An About Page introduces the brand of an e-commerce site, including its mission and role. It doesn't have any functionality and serves solely to inform customers about the brand. However, it is essential for establishing trust and credibility with customers who may be hesitant to buy without knowing more about the site. Therefore, an About Page contributes to building customer trust.

4 IMPLEMENTATION

To start building web apps using ReactJS, users can use the **npx create-react-app** command to create a new React project. After successfully creating the project, the user needs to go to the my-app directory and run the command npm start to start a local server and open the application in their browser. Alternatively, the user can also modify the files in the src folder to start building their React app. It's up to the user who wants to use CSS or SASS (Syntactically Awesome Style Sheets). But they can use CSS with some framework.

The website will encompass numerous primary functions, including Header and Footer components that will be visible on all pages. ReactJS will be implemented to break down the pages into smaller components that can be utilized throughout the website, saving time and effort compared to generating new components for each page. React and Redux will be utilized in designing the Front-end interface, while Express and Node will be used for the Back-end. The database will be MongoDB.

To implement a project, one of the important tools of the entire project is indispensable and contribute greatly to the construction of the entire project, which is NPM.

`$ npm i bootstrap`, `$ npm i redux`, `$ npm i fortawesome`, etc.

To start the project, install node_module for the front-end using the syntax, **$ npm install** or **$ npm i**. After the installation is complete, continue to use the **$ npm start** syntax to start the website.

The interface will be built in the order of Index, Product, About, Contact, Login, Register, Cart, and other pages. Overlapping headers and footers will be included in some pages, and will be built as separate components. The header will contain links to other pages using the **<Link to = "target"/>** method. During the implementation, the focus will be on sharing how the problem was solved by explaining the steps taken and the research done.

The problem being addressed is improving the quality of e-commerce sites. While most websites have basic structures and issues such as security, responsive design, and SEO have been resolved, there are still missing functions such as Localize and Dark Mode. Improving the user experience (UX) by providing customers with interfaces and filters will be a priority, and efforts will be made to solve the problems that have been identified.

4.1 Routes

React components in App, will render the HTML structure of the application. It imports and includes various other components like Header, Footer, and Routes, each serving a specific purpose in the application. The Routes component is used to define the routing of the application. It includes multiple Route components, each with a specific path and element. The element prop of each Route component specifies the component that should be rendered when the path matches the current URL.

Other components included in the Routes are Home, Login, Cart, Register, Contact, About, ProductDetail, ProductList, SearchProduct, Success, Policy, FAQ, Return, Size, Ecommerce, Ship, and Error. The Footer component includes the footer section of the application, and the CallJavaScript component includes some JavaScript code for the application.

```
function App() {
    return (
        <div className="App">
            <Header />
            <Routes>
                <Route path="/" element={<Home />} />
```

```
                <Route path="/login" element={<Login />} />
                <Route path="/cart" element={<Cart />} />
                <Route path="/register" element={<Register />} />
                <Route path="/contact" element={<Contact />} />
                <Route path="/about" element={<About />} />
                <Route path="/product/:productId" element={<ProductDetail />} />
                <Route path="/productType/:productType" element={<ProductList/>}/>
                <Route path="/product/search/keyword=:searchValue"
element={<SearchProduct />}/>
                <Route path="/payment" element={<Success />} />
                <Route path="/policy" element={<Policy />} />
                <Route path="/faq" element={<FAQ />} />
                <Route path="/return" element={<Return />} />
                <Route path="/size" element={<Size />} />
                <Route path="/ecommerce" element={<Ecommerce/>} />
                <Route path="/ship" element={<Ship/>}/>
                <Route path="*" element={<Error />} />
            </Routes>
            <Footer />
            <CallJavaScript />
        </div>
```

Listing 1: Route

4.2 General Functions

Besides the main pages like Home, Products, etc. the website also needs to have important components to contribute to these main pages, these are the functions that help us appear on all pages, because it is in the header, and they help the interface to be lively and complete.

4.2.1 Header

The Header is an essential component that should appear on any web page. One of the main problems that need to be addressed is how to make it convenient for users. Since the header is located at the top of the page, users may not be able to access it when they need it. To resolve this matter, a lot of present-day websites implement a feature that enables the header to trail along with the user while they navigate through the webpage.

All the information gathered from case studies has been taken into account, and my concept for the header includes displaying functions such as a list, button, login, color, and scrolling as the user navigates through the screen. The design of the header focuses on simplicity, and my approach differs from other case studies in terms of color and appearance. The dark mode is one of the necessary functions that I have included in the header, providing diversity and comfort in color for users. The login and product menu are equally important items, and I have put in a lot of effort to create an interface for them. Creating the headers took me nearly 3-4 weeks to complete, as they are quite challenging to design and require a lot of visual thinking, color, programming, and photo elements.

The header is one of the important bars that almost every website must have. In this product, the head is simply designed to ensure features. For convenience and fast, Bootstrap has been used to make grid setup quicker and more convenient.

The Header includes buttons such as:

- Logo
- Menu
- Search bar
- Multi-language

- Dark mode
- Cart button
- Login

Of course, there will be other functions but with this project and my concept, the simpler they are, the better.

4.2.2 Localization

With the technological development of today's websites, multilingualism is one of the indispensable functions, so this function is gradually added to websites, especially e-commerce. So, this is not a big problem in designing and implementing as they are not too difficult to build them either.

To enable the use of Localization, the i18next and i18nexus[13] libraries were utilized. I18next [38] is a framework based on JavaScript that is used for internationalization purposes. Additionally, i18nexus [39] was used to automatically translate app strings into multiple languages and store them in the cloud. In order to utilize this functionality, one would need to access the i18nexus website to generate JSON data for the desired language and integrate it into the project. It should be noted that free users are restricted to a maximum of 500 strings for all languages combined. Therefore, I chose to implement only the English and Czech languages.

The i18n.js file was exported and integrated into the project as the first step.

```
import i18next from "i18next";
import HttpBackend from "i18next-http-backend";
import LanguageDetector from "i18next-browser-languagedetector";
import { initReactI18next } from "react-i18next";
const apiKey = "PcD_JRWJe-puPeD96Yk3-A";
const loadPath =
`https://api.i18nexus.com/project_resources/translations/{{lng}}/{{ns}}.json?api_k
ey=${apiKey}`;
i18next
  .use(HttpBackend)
  .use(LanguageDetector)
  .use(initReactI18next)
  .init({
    fallbackLng: "en",
    ns: ["default"],
    defaultNS: "default",
    supportedLngs: ["en","cs"],
    backend: {
      loadPath: loadPath
```

Listing 2: Export i18n.js using the i18nexus API

Then go to index.js and import i18n.js. Suspense from React is what we'll utilize because we're using an API; we'll cover the App component in Suspense and make loading the fallback because of this. On top of that, we are importing our i18n.js file.

```
import "./i18n.js";
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
    <React.StrictMode>
        <Provider store={store}>
            <Suspense fallback="loading">
                <Router>
```

[13] https://i18nexus.com/

```
                    <App />
                </Router>
            </Suspense>
        </Provider>
    </React.StrictMode>,);
```

Listing 3: import file i18n.js to index.js and use <Suspense>

For the rest, just create **const {t}** from **useTranslation**, and put in the places where the language needs to be changed. But it is important to name the string that needs to be translated.

```
import { useTranslation } from "react-i18next";
function FAQ() {
  const { t } = useTranslation();
  return (
    <div className="site-content page-faqs site-content--main">
      <div className="container" style={{ maxWidth: '900px' }}>
        <div className="page__content">
          <h1 dir="ltr" style={{ textAlign: 'center' }}>
            {t('faq.content1')}
```

Listing 4: import and create {t} from useTranslation()

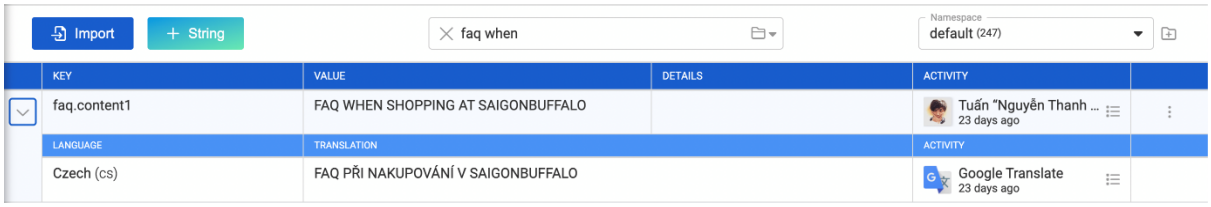

Figure 14: Create Key and Value for translation in i18nexus

The difficulty here when using Localization is that the amount of content is so large for a large project that one has to manually use and generate each value to translate. So, it will be very time-consuming for large projects.

4.2.3 Search

When creating a search bar, programming and interface design can pose difficulties, especially for beginners. It's important to ensure the search function is user-friendly and effective, as customers won't buy products they can't find. Search functionality is a critical aspect of creating a successful mobile app or website.

An issue arose, and extensive research was required on StackOverFlow to implement the product search functionality on the header. Eventually, a solution was found. In terms of the interface, the design was inspired by the previously mentioned case study, utilizing a bar interface with the word **"search"** displayed within it.

The search bar in the Header will make it easier for users to find the product they want, by typing in the Search bar. The results will navigate to the products that the user wants to find, on the website bar will show a domain name with the product they are looking for.

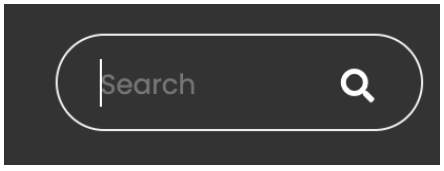

Figure 15: Search results

This is an event handler function when the user submits a form in the web application. When the form's submit button is clicked by the user, the browser will execute the handleSubmit function to manage the event. Inside this function, the first step is to utilize e.preventDefault() to stop the web page from refreshing upon form submission by the user.

Next, we check if the **searchValue** input field's value exists by using **.trim()** to remove unnecessary spaces at the beginning and end of the input string. If this value exists (non-null), then we use the **navigate()** function to redirect to the URL corresponding to the search keyword **searchValue**, for example: **/product/search/keyword=abc.** Then we reset the value of the **searchValue** input field to the empty string using **setSearchValue('')**. If the value of **searchValue** input field does not exist or is an empty string, then we use **navigate()** function to redirect to the web application home page **('/')**.

```
    const handleSubmit = (e) => {
        e.preventDefault();
        if (searchValue.trim()) {
            navigate(`/product/search/keyword=${searchValue}`);
            setSearchValue('');
        } else {
            navigate('/');
        }
};
```

Listing 5: Search Function

```
<form id="search-submit" className="form-box f-right" onSubmit={handleSubmit}>
<input
    id="search-id"
    type="text"
    name="Search"
    placeholder={t("header.search")}
    value={searchValue}
    onChange={(e) => setSearchValue(e.target.value)}/>
<button
    className="search-icon"
    type="submit"
    style={{ border: 'none', backgroundColor: 'white' }}>
    <i className="fas fa-search special-tag"></i>
</button>
</form>
```

Listing 6: Icon form

4.2.4 Dark Mode

Dark mode is one of those functions that few commercial websites use, this is the problem because these days dark mode technology has been applied so much that even Google has used it. And here is the implementation and solution I came up with. First, we use **useState** to initialize the theme variable to the value taken from **localStorage** or light theme ('light') if there is no value in **localStorage**. Also, we use **setTheme** to update the value of the theme variable when the user makes a light/dark theme switch.

Next, we use **useEffect** to save the value of the theme variable to **localStorage** and apply the corresponding theme to the elements in the web page. First, we set the **className** of the body element to the value of the theme variable. We then use the DOM methods to find and change the **className** of other elements on the web page to apply the corresponding theme. For example, we use **document.getElementById** to find the element with the id 'header-sticky' and set its **className** with the string **'row align-items-center'** and the value of the theme variable. Similarly, we use a for loop to change the **className** of elements whose class is **'name_product'** to apply the corresponding

theme. We use **getElementById** and other DOM methods to change other properties of the elements on the web page, such as the background color of the element with the id **'header1'**, the class of the element with the class **'slicknav_menu'** and **'shopping-card-color'**, **'favorit-items-color'** etc. to apply the respective theme.

The difficulty with setting dark modes is getting them to work as color transitions without overriding each other in the CSS. And each CSS code had to be edited separately and specified with other classes to avoid overlapping colors that could result in batch color changes and cause colors to change in unintended places.

```
.dark {
  background-color: #333;
  color: #fff !important;}
.light {
  color: #333;
}
```

Listing 7: Dark and Light color in CSS

```
    const [theme, setTheme] = useState(localStorage.getItem('theme') || 'light');
    const toggleTheme = () => {
        if (theme === 'light') {
            setTheme('dark');
        } else {
            setTheme('light');
        }
    };
    useEffect(() => {
        localStorage.setItem('theme', theme);
        document.body.className = theme;
        if (theme === 'dark') {
            toggleDarkMode(true);
        } else {
            toggleDarkMode(false);
        }
        document.getElementById('navigation').className = theme;
        const products = document.getElementsByClassName('name_product');
        for (let i = 0; i < products.length; i++) {
            products[i].className = 'name_product ' + theme;
        }
        document.getElementById('header-sticky').className = 'row align-items-
center ' + theme;
        document.getElementById('header1').style.backgroundColor = theme ===
'light' ? '#fff' : '#333';
        document.getElementsByClassName('slicknav_menu').className=theme;
        document.getElementById('search-submit').className = 'form-box f-right ' +
theme;
        document.getElementById('dark-mode-btn').className = theme;
        document.getElementById('shopping-card-color').className = 'shopping-card
' + theme;
        document.getElementById('favorit-items-color').className = 'favorit-items
' + theme;
```

Listing 8: Dark Mode function

4.2.5 Responsive

"Responsive" refers to the capability of a website or web application to adjust to varying screen sizes and device types on which it is accessed. In a responsive website, the interface elements, such as

text, images, icons, layouts, menus, etc. are automatically resized, positioned or laid out depending on the screen size.

Responsive design helps users have a better experience when accessing the website on different devices, such as smartphones, tablets, laptops or desktop computers. It also saves developers time and money by developing only one version of a website or web application instead of creating separate versions for each type of device.

This is the CSS code, used to apply CSS rules to an element on a web page when the screen size is less than or equal to 767px.

Specifically, this code sets the position of the element with the class "fix-card" as an absolute position and positions it 12px above the top of the parent element and 12px to the right of the element. father 85px. Often, it is used to adjust the position of elements on a web page so that it matches the screen and device used to access the web page.

```
@media (max-width: 767px) {
  .header-bottom .fix-card {
    position: absolute;
    top: 12px;
    right: 85px;}}
```

Listing 9: Responsive with @media in CSS

4.3 Index page

Index page, this is the starting page for the web interface, because this is the first page and also the page that decides whether to retain customers or not. Because the homepage must provide users with an eye-catching design and attract customers to retain them and refer to other pages.

The concept presented for this homepage is a simple design with black and white colors that depend on the dark mode function. Other elements on the page, such as the carousel, best collection, latest product, etc., will also have a simple yet eye-catching design to avoid being boring for users.

In which **LatestProducts** will be rendering the most popular products

In **useEffect**, the API is called by the GET method to the address **'http://localhost:8000/v1/product'** using the Axios library. Then use the **.then** method to update the state of PRODUCTS with the data returned from the API using **setPRODUCTS**. At the same time, also use **setLoading** to update the loading variable's status to false to notify that the data has been loaded. If any error occurs during the API call, use **.catch** to display the error message on the console.

Using an empty array [] as the second argument for useEffect guarantees that useEffect executes solely during the initial rendering of the component, preventing redundant API requests. In React, the useEffect hook is utilized together with Axios to retrieve data from a GET API located on the backend and then update the component's state with the retrieved data. How the product data is GET from the Backend is explained in the 4.9 Design Backend section.

```
useEffect(() => {
    axios.get('http://localhost:8000/v1/product')
        .then(res => {
            setPRODUCTS(res.data);
            setLoading(false);
        }).catch(err => console.log(err)); }, []);
```

Listing 10: Axios

The Index page will include sections:

- Carousel

- Latest Product
- Best Products
- Best Collection
- Latest Offers News
- Policy Card
- Banners
- List Image and Logo
- Marquee

Most of these items belong to the interface, so just designing and programming them is complete and they do not have special functions.

The next difficulty encountered was the **BestCollection** and **LatestOffers** sections. As there was no photo data available, hardcoding was necessary to ensure the proper functioning of the site.

4.4 Product page

Based on the idea described in the Case Study, the Product page will include the following Sections:

- Product Cards
- Filter

The Product page will include 3 functions:

- Filter
- More Products

View more function when there are too many products, and the products will be displayed up to 6, if more forces the user to click the see more button to avoid making the page too long. And the design will not be too complicated and fussy, making it difficult to program and not attract the attention of customers.

- Buy Now

On the Product page, customers can browse and select products by category. The challenge is to design an interface that entices customers to make a purchase. Designers and marketers face the difficult task of creating a website that not only showcases beautiful product images but also has the necessary features to drive sales.

To tackle this difficulty, three crucial functions were integrated into the Product page, and each of these functions will be elaborated upon thoroughly.

4.4.1 More Product

This function to view more products is used to load products gradually without having to display them all at once to avoid wasting time loading products on the server.

The initial challenge was that after rendering all the data, the "More" button remained active and did not disable itself. The solution is to add a **setLoadMore** variable to check when the more button will appear, and when pressing the more button, it will check if the data has been rendered or not.

```
    const setVisibleProduct = () => {
        console.log(visible);
        setVisible(visible + 6);
        if (visible + 6 >= PRODUCTS.length) {
            setLoadMore(false);}};
```

Listing 11: More Product

4.4.2 Filter

On the Product page, the Filter section plays a crucial role in e-commerce as it allows customers to refine their search results and find the products, they are looking for easily. Sort and filter options are often overlooked in online searches, but they make the process of finding products less daunting. By limiting the display of results to only relevant items, customers can avoid sifting through many "similar" products, which can be overwhelming. Filter and sort options are sub-elements of any search, as they enable users to organize and refine their search results without the need for extended pagination or scrolling on small screens. While filters offer numerous options, as a programmer, it is important to keep the options simple with a limited number of choices to avoid overwhelming customers with too many choices.

The concept of filters has already been described in the Case Study section. While filters can offer many options, it is important to keep them simple in terms of programming, with a limited number of choices and options to avoid overwhelming customers with too many choices.

To handle difficulty searching products without calling the server continuously, the front-end method **.filter()** was used. If **priceFilter** is 0, all products are returned. For 30, products priced less than or equal to 30 are returned. For 100, products priced between 30 and 100 (inclusive) are returned. For 200, products priced between 100 and 200 (inclusive) are returned. For 1000, products priced greater than or equal to 200 are returned. The function uses if-else statements to check each product's price against the **priceFilter** and returns the product if it meets the condition.

```
PRODUCTS.filter((item) => {
    if (priceFilter === 0) {
        return item;
    } else {
        if (priceFilter === 30) {
            if (item.price <= priceFilter) {
                return item;}
        } else if (priceFilter === 100) {
            if (item.price <= priceFilter && item.price >= 30) {
                return item;}
        } else if (priceFilter === 200) {
            if (item.price <= priceFilter && item.price >= 100) {
                return item;}
        } else if (priceFilter === 1000) {
            if (item.price >= 200) {
                return item;
            }}}}).filter((item) => {
    if (vodeFilter === 0) {
        return item;
      }
    else if (Math.round(item.ratting) === vodeFilter) {
        return item;}}
).filter((item) => {
    if (brandFilter !== '') {
        return item.category.includes(`BRAND-${brandFilter.toLowerCase()}`);
    }
    return true;
})
```

Listing 12: Filter function

4.4.3 Buy Now

The interface after clicking buy now will be like the Product Detail page but has been collapsed and displayed quickly, it is like a miniature Product Detail page to help users buy products more quickly and conveniently. To perform this function, the interface and functionality of the Product Detail page needed to be completed before implementing the Buy Now page. Since the interface of both pages is similar, the code is also similar. I will explain the code for the Buy Now page in the Product Detail section. One of the difficulties in designing the Buy Now page was to display the correct product information when the "Buy Now" button is clicked. This was resolved by passing the product object as a parameter.

```
function BuyNow({ open, setOpen, product}) {
```

Listing 13: BuyNow function

```
<BuyNow
      open={open}
      setOpen={setOpen}
      product={product} />
```

Listing 14: Call BuyNow function

4.4.4 Product Card

The Product Card component is responsible for displaying the product card. It includes various details such as the product name, image, price, and other relevant information. The purpose of the Product Card is to present the product information in a visually appealing and organized manner, making it easier for users to view and understand. Furthermore, the Product Card also ensures consistent display on different devices and screen sizes.

In terms of design, Product Card will be very difficult, I have also tried to refer to many websites, research and analyze each type but still ensure the concept given. So how to have a unique Product Card, impress customers, especially the function of the Product Card can help customers feel convenience when buying. This is my biggest obstacle when trying to research and analyze.

On the product page, the products will be displayed using the product tag element. The product tag is also very difficult to design because it has too much structure from HTML to CSS, so there are a lot of bugs such as framework bugs, structural bugs, etc., so it takes a lot of time to design and fix.

"**ProductItem**" takes in two props - "**data**" and "**type**". The component returns a JSX code that represents the visual appearance of a product item. It contains two main sections: the top section and the bottom section.

The top section includes an image of the product and a button to buy the product. When the user clicks on the product image or the "Buy Now" button, the "**useNavigate**" hook from the React Router library is used to navigate to a new page, passing the "data" object as state data.

The top section also includes a rating display for the product, showing the average rating and the number of ratings the product has received. The bottom section includes the name and price of the product. Overall, this component is designed to be used as a reusable UI element in an e-commerce website to display information about a product.

```
function ProductItem({ data, type, setOpen, setProduct }) {
    const { t } = useTranslation();
    const navigate = useNavigate();
    return (
        <div className="product_container">
            <Link className="content_up">
                <Link to={data.to}><img className="img_1" src={data.image}/></Link>
            <Link to={data.to}><img className="img_2"src={data.imageHover}/></Link>
```

```
                <div className="buy_btn">
                    <div style={{ color: 'black' }} className="buy_btn_inner"
                        onClick={(e) => {
                            setOpen(true);
                            setProduct(data);
                         }}>
                        {t('item.buy')}
                    </div>
                </div>
                <div className="ratting" style={{ fontSize: 14, color: 'black' }}>
                    {data.ratting}
                    <span><FontAwesomeIcon icon={faStar} /></span>
                    <span style={{ fontWeight: 600, fontSize: 14
}}>({data.countOfRatting})</span>
                </div>
            </Link>
            <div className="content_down">
                <Link className="name_product" to={data.to}>
            <div style={{ fontWeight: 600, fontSize: 14,}}>{data.nameProduct}</div>
                </Link>
                <div className="price">{data.price}$</div
            </div>
        </div>
    );}
```

Listing 15: Product Item component

```
<div className="shirt_container">
<div className="shirt_header">
    <h3 style={{ color: 'black', paddingTop: 50 }}>{ nameApp }</h3>
</div>
<div className="shoes_content">
<Fillter handleChoicePrice={handleChoicePrice} handleChoiceVote={handleChoiceVote}
handleClear={handleClear} handleBrand={handleBrand} />/*Filter UI */
    <div className="shirt_inner">
        { loading ? <div className="loading">Loading...</div> :
/*Filter function */}).slice(0, visible).map((item, index) => {
                return <ProductItem data={item} key={index} setOpen={setOpen}
                    setProduct={setProduct}/>;})}
        </div>
    </div>
<div className="button-show-more">
    {loadMore ? <ButtonMore onClick={ setVisibleProduct } /> : ''}</div>
            <BuyNow open={open} setOpen={setOpen} product={product} />
div>
```

Listing 16: Product Layout

4.5 Detail page

The Product page is the page where people will choose the product, after they like the product, and they want to learn more about the material and product information, they will click on the product and visit the Product Detail page to learn more details. Product Detail, when the customer has clicked on this page, we have reached the final step before the customer enters the checkout page, but the customer can also exit if they feel the website is not impressive enough about the product.

So, the image below is the Product Detail designed by me based on the case studies for reference and building them. With simple but complete information and not too complicated.

On the Product Detail page, the value is set to the slug, so when a user clicks on a product, the link will display the product name in the URL without the need to create numerous links. In order to fetch the product associated with the final URL, the initial stage is to acquire the product ID, followed by a server request to retrieve the product's particulars, which comprises:

```
useEffect (() => {
    axios.get('http://localhost:8000/v1/product/' + productId).then((res) => {
        setProduct(res.data);
        setRating(res.data.ratting.toFixed());}).catch((err) => {
        console.log(err);
    })}, [productId]);
```

Listing 17: Create pathname for product

The next challenge is to display related products on the Product Detail page. To achieve this, I need to search for products that have attributes related to the product being viewed. For this, I choose to search by category and display only 4 related products to avoid overwhelming the customer with too many options.

```
useEffect(() => {
    axios.get('http://localhost:8000/v1/product?Category='+ product.category)
        .then(res => {
            setPRODUCTS(res.data);
        }).catch(err => console.log(err));
}, [product]);
```

Listing 18: Related Product section

After that will process the add to cart, there will be 2 important parts: increase and decrease products and add products to the cart.

First will be the number of products before adding to the cart, will add a **useState** variable quantity and handle the increase and decrease so that the program does not run unreasonably (cart -1 or greater than 10).

```
const [quantity, setQuantity] = useState(1);
const handleAdd = () => {
    if (quantity < 10) {
        setQuantity((prev) => prev + 1);}};
const handleSub = () => {
    if (quantity > 1) {
        setQuantity((prev) => prev - 1);}};
```

Listing 19: Set Quantity of Product

This code uses Redux's dispatch function to dispatch an action to the reducer. Specifically, the action is created with the **addToCart** function, whose argument is an object containing the product's information such as **id, nameProduct, price, image** and **quantity**.

By dispatching this action, Redux will revise the application's state and include the product to the shopping cart. In the case where a product is already present in the cart, instead of appending a new product to the cart, Redux will modify the quantity of that existing product.

```
dispatch(
    addToCart({
              id: productId,
             nameProduct: product.nameProduct,
              price: product.price,
              image: product.image,
              quantity: quantity,
```

Listing 20: Add Product to Cart

That's all the functionality of the Product Detail page. From interface to function, building them also takes a lot of time to learn and develop them.

This code defines the layout and functionality of a product detail page, including displaying the product's image, name, rating, price, available colors, quantity selector, add-to-cart button, and description. The code also includes functions for rendering the product's rating and handling color selection.

```
<div className="detail_inner">
        <div className="detail_left">
            <img src={product.image} className="image_product"
alt={product.nameProduct} />
        </div>
        <div className="detal_right">
            <h1 className="name_product">{product.nameProduct}</h1>
            <div className="detal_ratting">
                {showRatting()}({product.countOfRatting})
            </div>
            <div className="price">{product.price}$</div>
            <div className="chose-color">
                <div className="color-header">{t('detail.color')}</div>
                <div className="color-list">
                    {colors && colors.map((color, index) => (
                            <span
                                key={index}
                                style={{ backgroundColor: color }}
                                className={color === choseColor ? 'color-item
choses-color' : 'color-item'}
                                onClick={() => handleColor(color)}></span>))}
                </div>
            </div>
            <div className="add-to-cart">
                <div className="quantity">
                    <div onClick={handleSub} className="sub-product-btn"> -
                    </div>
                    <span>{quantity}</span>
                    <div onClick={handleAdd} className="add-product-btn"> +
                    </div>
                </div>
                <div
                    className="action" onClick={() =>
                        dispatch(
                            addToCart({
                                id: productId, nameProduct: product.nameProduct,
                                price: product.price, image: product.image,
                                quantity: quantity,
                            }),
                        )}>
                    {t('detail.addtocart')}
                </div>
            </div>
            <div className="feature">
                <div className="feature_header">{t('detail.description')}</div>
                <div className="feature_content">
                    {product.features &&
                        product.features.map((feature, index) => (
                          <div key={index} className="feature_item">- {t(feature)}
                          </div>
```

Listing 21: Detail Layout

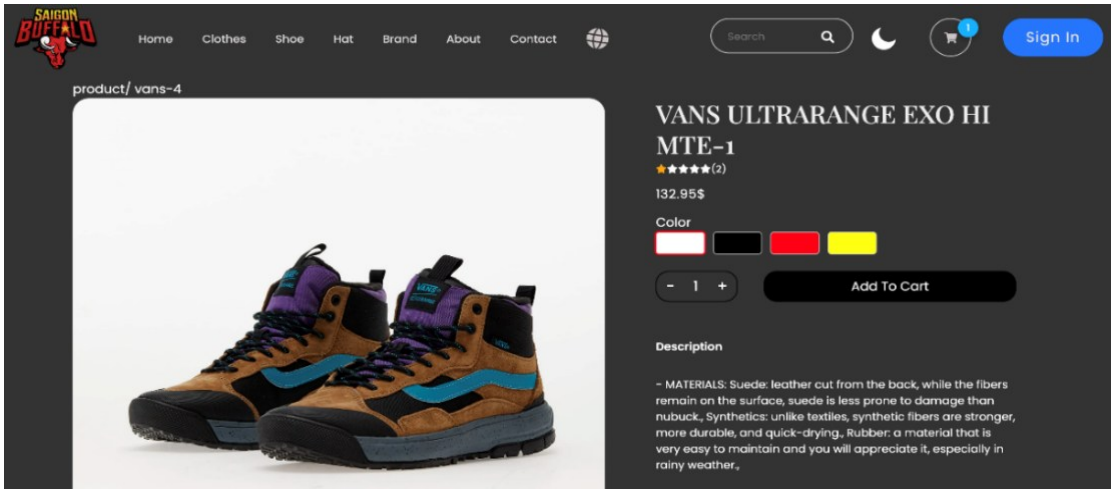

Figure 16: Product Detail page

4.6 Cart page

The cart page stores the products selected by customers, and there are various attractive designs available, such as Nike, Adidas, and more. However, a minimalist design was chosen as the main concept for the list design to avoid being too cluttered, even though the interface may not be aesthetically pleasing. Despite having several limitations in the design process and application to the project, visitors can still purchase goods or services using the cart. Customers can also read the specifics of the products and choose their payment methods before proceeding to checkout. During the coding process, confusion arose while selecting a programming language. Redux, a language supported by React, was ultimately chosen. However, the research on Redux revealed its complexity and several sources had to be consulted to improve the desired functionality.

Once the customer decides to purchase the product, it will be included in the shopping cart. But in order to be able to store it in the cart and be able to update it if they buy a lot or delete products, in React one should use **createSlice** in Redux. **createSlice** is a function that deals with everything needed for each slice, an object of reducer functions. This API is the approach to writing Redux logic. The above source code defines a slice of the Redux Toolkit, which is **cartSlice**. This **slice** contains several reducers to manipulate the cart state of the application.

cart is an array containing the products that have been added to the user's cart. This slice contains the following reducers: **addToCart**: This reducer is called when a product is added to the cart. It checks if the product has been added to the cart and if so, increases the quantity of that product by one, otherwise adds the product to the cart with the initial quantity of one unit.

incrementQuantity: This reducer is called when the user increases the quantity of a product in the cart. It searches for that product in the cart, increasing its quantity by one. **decrementQuantity**: This reducer is called when the user reduces the quantity of a product in the cart. It searches for that product in the cart, if the quantity of the product is 1, the quantity will be kept the same, otherwise, the quantity of the product will be reduced by one.

removeItem: This reducer is called when the user removes an item from the cart. It filters that product from the cart array.

```
const cartSlice = createSlice({
    name: 'cart',
    initialState: {
        cart: [], numberCart: 0,},
    reducers: {
        addToCart: (state, action) => {
```

```
            console.log(action.payload);
            const itemInCart = state.cart.find((item) => item.id ===
action.payload.id);
            if (itemInCart) {
                itemInCart.quantity += action.payload.quantity;
            } else {
                state.cart.push({ ...action.payload });
                state.numberCart++; }},
        incrementQuantity: (state, action) => {
            const item = state.cart.find((item) => item.id === action.payload);
            item.quantity++;},
        decrementQuantity: (state, action) => {
            const item = state.cart.find((item) => item.id === action.payload);
            if (item.quantity === 1) {
                item.quantity = 1;
            } else {
                item.quantity--;}},
        removeItem: (state, action) => {
            const removeItem = state.cart.filter((item) => item.id !==
action.payload);
            state.cart = removeItem;
            state.numberCart--;},
        payment: (state) => {
            state.cart = [];
            state.numberCart = 0;},
```

Listing 22: createSlice for adding, removing and updating in Cart page.

persistConfig object which contains the key for the persisted state in storage and the storage engine. Here, storage refers to the **localStorage** object in the browser. The **persistReducer** function is used to wrap the **cartReducer** from the **cartSlice** module with the **persistConfig** object. This creates a new reducer that can persist the state of the **cartReducer** in the browser's local storage.

The **configureStore** function from Redux Toolkit is used to create the Redux store. It takes an object as an argument with a reducer property set to the persisted reducer created using **persistReducer**. It also defines a middleware to handle serializable actions with the **getDefaultMiddleware** function, ignoring actions such as **FLUSH**, **REHYDRATE**, **PAUSE**, **PERSIST**, **PURGE**, and **REGISTER**.

Finally, **persistStore** function is used to create a persistor object, which enables the Redux store to be rehydrated from the persisted state. The **'store'** and **'persistor'** are exported so that they can be used in other parts of the application.

```
const persistConfig = {
    key: 'root', storage,
};
const persistedReducer = persistReducer(persistConfig, cartReducer);
export const store = configureStore({
    reducer: persistedReducer,
    middleware: (getDefaultMiddleware) => getDefaultMiddleware({
            serializableCheck: {
              ignoredActions: [FLUSH, REHYDRATE, PAUSE, PERSIST, PURGE, REGISTER],},
```

Listing 23: Store in Redux

First data must be got from the redux card by way:

```
const cart = useSelector((state) => state.cart);
{cart?.map((cart, index) => {
    return <CartProduct key={cart.id} props={cart}
setValueChecked={setValueChecked} checkAll={checkAll} />;
```

Listing 24: Get data from redux

Next, each value of the card will be printed on the screen. However, making a purchase requires the user to be logged in. Therefore, a check will be performed to determine if there is an active user session in Redux. If not, the user will be prompted to log in before being allowed to make a payment. The **useSelector** hook will be utilized to retrieve the **'user'** value from the store, and the value will be used in the conditional rendering process. If the 'user' is considered truthy, the "Process Checkout" button will become visible, and it will be assigned an onClick function that sets the openModal state to 'true'. Conversely, if the 'user' is determined to be falsy, a Link component will be presented in its place, and clicking on it will lead the user to the login page.

```
const user = useSelector((state) => state.user);
    {user ? (
        <div className="btn_1 checkout_btn_1"
            style={{ cursor: 'pointer' }}
            onClick={() => setOpenModal(true)}>
            Process check out
        </div>
    ): (
        <Link className="btn_1 checkout_btn_1" to="/login">
            Login to process check out
        </Link>
```

Listing 25: Login to pay

Processing registration: when pressing register will show 1 model, then a table will appear for the user to enter information, then when pressing confirm will proceed to payment and delete all products in the cart. At the same time, 1 more successful payment page will appear.

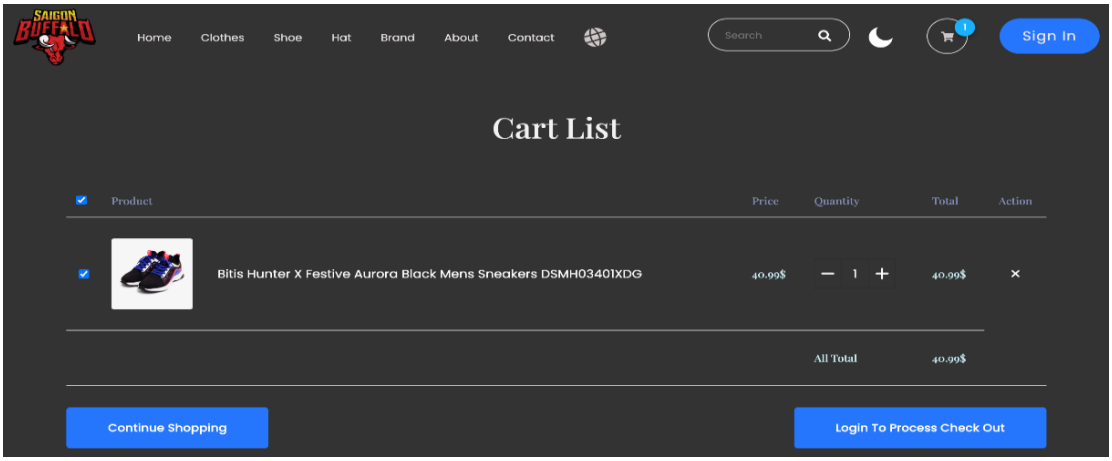

Figure 17: Cart page

After logging in to proceed with the payment, the user will see a modal to proceed with the payment, just enter the payment data, the admin will receive the information and proceed with the delivery. Because only focusing on the interface, there will be no backend for receiving purchase information from customers.

```
{openModal && cart.length > 0 && <Modal onOpen={setOpenModal} />}
```

Listing 26: OpenModal to checkout

```
<div className="dark-bg" onClick={() => onOpen(false)}>
            <div className="modal_container" onClick={(e) => {
                    e.stopPropagation();}} >
                <div className="modal_header">
                    <div className="modal_title">{t('modal.address')}</div>
```

```
                </div>
                <form className="modal_content" onSubmit={handlePay}>
                    <input placeholder={t("modal.fullname")} className="content_2"
required/>
                    <div className="choose">
                        <div className="content_1">
                        <select className="select_1" value={selectedCity.name}
onChange={handleCityChange}>
                            {cities.map(city => (
                 <option key={city.name} value={city.name}>{city.name}</option>))}
                        </select>
                        </div>
                        <select className="select_1" value={selectedDistrict}
onChange={handleDistrictChange}>
                          {selectedCity.districts.map(district => (
                          <option
key={district}value={district}>{district}</option>))}
                            </select>
                    </div>
        <input placeholder={t("modal.fulladdress")} className="content_4"
required/>
              <input placeholder={t("modal.note")} className="content_2"
required/>
              <input placeholder={t("modal.pay")} className="content_3" required/>
              <input placeholder={t("modal.code")} className="content_3"
required/>
                <button className="btn_1 checkout_btn_1 pay">{t('modal.checkout')}
                </button>
                </form>
            </div></div>
```

Listing 27: Open Modal for Checkout

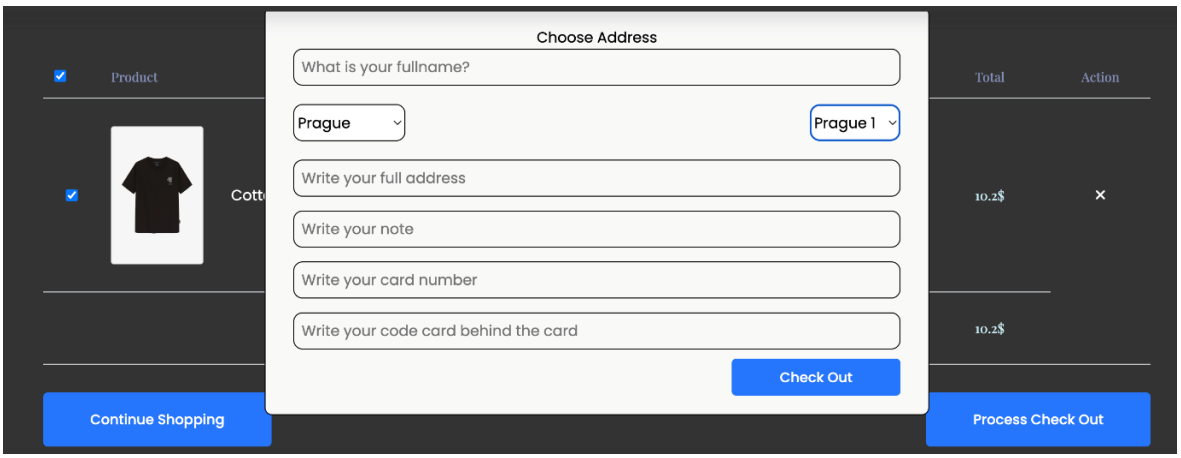

Figure 18: Check out

4.7 Login and Register Page

The login and register pages are essential components of any website that requires user authentication. The login page is used by registered users to log in to their accounts, while the register page is used by new users to create an account.

To build the login and register pages, web developers can use a variety of technologies, including HTML, CSS, JavaScript, and frameworks like React. The pages should be designed with usability in mind, with clear and concise instructions on how to log in or register, and any error messages should be displayed prominently to help users troubleshoot issues.

4.7.1 Register

This is a function that handles user registration. It first checks if all the necessary fields (name, phone, email, password, confirmPass) are filled in, and if not, it navigates the user back to the register page and displays an alert message to prompt the user to fill in all the required information.

If all fields are filled in, the function then checks if the password and confirms password fields match. If they do not match, it navigates the user back to the register page and displays an alert message to prompt the user to re-enter their confirm password.

If both checks pass, the function sends a POST request to the backend using **Axios** to register a new user. The POST request includes the user's name, phone number, email, and password. If the request is successful, the function displays an alert message to indicate successful registration, navigates the user to the login page and the data will be saved in the MongoDB cloud. If the request fails, the function displays an alert message to indicate that registration failed.

```
    const handleRegister = (e) => {
        e.preventDefault();
        if (name === '' || phone === '' || email === '' || password === '' ||
confirmPass === '') {
            navigate('/register');
            alert('Please enter enough information');
        } else if (confirmPass !== password) {
            navigate('/register');
            alert('Please re-enter your confirm password');
        } else {
            axios.defaults.headers.post['Content-Type'] = 'application/json';
            axios.post('http://localhost:8000/v1/user/register', {
                fullname: name, phone: phone, email: email, password: password, })
            .then((res) => {
                console.log(res);
                alert('Register successfully');
                navigate('/login');
            }).catch((err) => {
                console.log(err);
                alert('Register failed');
```

Listing 28: Register function

```
_id: ObjectId('640efc4cefaa36a276279a05')
fullname: "nguyen tuan"
phone: "093256437"
email: "tuan@gmail.com"
password: "123456"
createdAt: 2023-03-13T10:34:52.920+00:00
updatedAt: 2023-03-13T10:34:52.920+00:00
__v: 0
```

Figure 19: Save new user to MongoDB

A form component in React is used for user registration. The form contains several input fields for the user to enter their full name, phone number, email address, and password. There are also two password fields to confirm the user's password. The form has a submit button that triggers a function called **handleRegister** when clicked. This function is passed as a prop to the **onSubmit** attribute of the form tag.

When the user enters their name, phone number, email, and password, the **onChange** event is fired, which triggers a callback function to update the component's state with the user's input. This state is

used to pass data to the **handleRegister** function when the user submits the form. The form also contains a link to the login page and some text encouraging the user to register.

```
<form className="row contact_form" action="#" method="post"
noValidate="novalidate" onSubmit={handleRegister}>
<div className="col-md-12 form-group p_star">
    <input type="text" className="form-control" id="name" name="name"
        defaultValue="" placeholder={t('modal.fullname')}
        onChange={(e) => setName(e.target.value)} />
</div>
<div className="col-md-12 form-group p_star">
    <input type="text" className="form-control" id="password" name="phone"
        defaultValue="" placeholder={t('register.content3')}
        onChange={(e) => setPhone(e.target.value)} />
</div>
<div className="col-md-12 form-group p_star">
    <input type="email" className="form-control" name="email"
        defaultValue="" placeholder={t('about.email')}
        onChange={(e) => setEmail(e.target.value)} />
</div>
<div className="col-md-12 form-group p_star">
    <input type="password" className="form-control" id="password" name="password"
        defaultValue="" placeholder={t('login.content8')}
        onChange={(e) => setPassword(e.target.value)} />
</div>
<div className="col-md-12 form-group p_star">
    <input type="password" className="form-control" id="password" name="password"
        defaultValue="" placeholder={('register.content4')}
        onChange={(e) => setConfirmPass(e.target.value)}/>
</div>
<div className="col-md-12 form-group register_btn">
    <button type="submit" className="btn_3 btn">
        {t('register.content1')}
    </button>
    <div>
        {t('register.content5')}{' '}
        <Link to="/login" style={{ color: 'blue' }}>{t('login.content1')}</Link>
    </div>
</div></form>
```

Listing 29: Register form

4.7.2 Login

This function manages the process of user authentication. It employs **Axios** to send a POST request containing the user's email and password to an API endpoint located on the server. In the event that the request succeeds, the function will log the server response to the console, present an alert message informing the user of a successful login, and navigate them to the homepage. Furthermore, a login action containing the user's account data will be dispatched. Conversely, if the request is unsuccessful, the function will exhibit an alert message that indicates the login attempt has failed.

```
    const handleLogin = (e) => {
        e.preventDefault();
        axios.defaults.headers.post['Content-Type'] = 'application/json';
        axios.post('http://localhost:8000/v1/user/login', {
            email: email, password: password,
        }).then((res) => {
            console.log(res);
            alert('Login successfully');
```

```
            dispatch(login(res.data.account));
            navigate('/');
        }).catch((err) => {
            console.log(err);
            alert('Login failed');
```

Listing 30: Login function

Same with the Register page, this is a form component for a login page. It contains two input fields, one for the user's email and one for their password, and a checkbox for remembering the user's login credentials. The form is submitted by clicking the submit button, and a link to reset the password is provided using the React Router's Link component.

The **onSubmit** prop is passed a function called **handleLogin**, which is likely defined elsewhere in the code and handles the login functionality, such as validating the user's credentials and redirecting them to the appropriate page. The **onChange** event is used to capture the values entered in the email and password fields and update the state variables **setEmail** and **setPassword**, respectively. These state variables are likely defined elsewhere in the code and are used to store the user's input for submission to the server.

4.8 Other Page

A website has diverse pages serving distinct functions and carrying various content types. Examples include an About page for company info, a Contact page for communication, Shipping and Return Policy pages, an FAQ page for addressing common questions, a Privacy Policy page detailing user data handling, a Size Guide page to help select product sizes and an E-commerce page with relevant information.

Each of these pages is designed to serve a specific purpose, and the content on each page will vary accordingly. Depending on the purpose of the page, different technologies may be used to create it. For example, pages that primarily display information might be created using HTML and CSS, while pages that allow users to interact with the site might use JavaScript. However, there is a contact page I have designed for the web that must have enough data to be posted and if there is not enough data, an error message will be reported to the program.

4.9 Design Backend

After designing the Frontend, the Backend was created to connect data to it. Axios or fetch was used to send requests from the React front-end to the NodeJS server API. Express, NodeJS, and MongoDB were used to construct the server side. The server side did not have many features since the focus was on the Frontend. The Backend was used to manage the product and user data to ensure that the product met modern website standards. Whenever a user registered, the data was stored in Mongo's Atlas account. The product data was generated in the Backend and called by the Frontend during runtime.

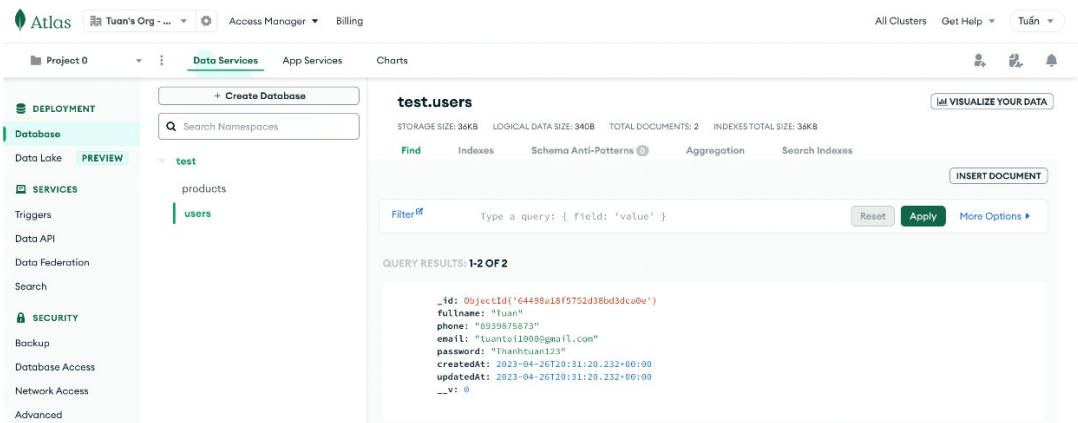

Figure 20: MongoDB cloud

4.9.1 Create Model

Two models are required for the design, namely user and product. The user model will contain basic information such as full name, phone number, email, and password. Since manual data entry would require admin access, fake data will be generated for the product instead. The product model will contain basic information such as id, name product, category, price, image, ratting, count of ratting, URL (to), and description of product (features).

ProductController is an object with methods that handle different types of requests related to products. For example, getAllProducts returns all products when the client makes a GET request to the route **/v1/product**, while getProductByCategory returns all products in a specific category when the client makes a GET request to the route **/v1/product/category/:id**, where **:id** is the category name.

The backend uses Express.js to handle HTTP requests, and the product routes are defined in index.js. For example, **app.use('/v1/product', productRoute)** specifies that any requests to **/v1/product** should be handled by productRoute, which is defined in product.js.

The frontend code uses **axios** to make an HTTP GET request to the backend server, specifically to the route **/v1/product** with a query parameter Category that specifies the type of products to retrieve. The response from the server is then used to update the state of the frontend component, which in turn updates the UI.

```
const userSchema = new mongoose.Schema({
    fullname: {
        type: String, required: true, minLength: 3, maxLength: 20, unique: false},
    phone: {
        type: String, required: true, minLength: 2, maxLength: 20},
    email: {
        type: String, required: true, minLength: 5, maxLength: 100},
    password: {
        type: String, require: true, minLength: 4},
},{timestamps: true});
```

Listing 31: User Model

```
const dataProduct = [
    {
        id: 'hat-2',
        nameProduct: 'CAP NEW ERA NEW YORK KNICKS TEAM ARCH 9FIFTY SNAPBACK CAP',
        category: 'HAT'+'BRAND-mlb',
        price: 33.95,
        image: `https://be-eshop.onrender.com/public/images/hat/product1.jpg `,
        ratting: 3,
        countOfRatting: 2,
        to: '/product/hat-2',
        features: [
            'feature.text'],
}]
```

Listing 32: Product Model

```
app.use('/v1/product', productRoute);
app.use('/v1/user', userRoute);
```

Listing 33: Index.js in Backend

```
const [loading, setLoading] = useState(true);
useEffect(() => {
```

```
    setNameApp(productType);
    document.title = `Product | ${productType}`;
    setLoading(true);
    axios.get('http://localhost:8000/v1/product?Category=' + productType)
        .then(res => {
            setPRODUCTS(res.data);
            if (res.data.length <= 6) {
                setLoadMore(false);
            } else {
                setLoadMore(true);}
            setLoading(false);
            setVisible(6);
        }).catch(err => console.log(err));
}, [productType]);
```

Listing 34: Make an HTTP GET request by axios in ProductList.js

4.9.2 Handling User for Login and Register page

The registration and login pages are two interconnected pages. In order to make purchases on the website, users are required to register and provide their information. The User object is initialized to enable the server to receive and store data in Mongo's Atlas cloud. The registration and login pages are two interconnected pages. In order to make purchases on the website, users are required to register and provide their information. The User object is initialized to enable the server to receive and store data in Mongo's Atlas cloud. Upon user registration, the data that they submit will be received and employed to generate a fresh User object. In the event of a successful registration, a message indicating success will be exhibited to the user, whereas an error message will be displayed if the registration process fails.

```
register: (req, res) => {
    try {
        const { fullname, phone, email, password } = req.body;
        const user = new User({
            fullname, phone, email, password
        });
        user.save().then(() => res.status(200).json({message: 'User created'}))
    } catch (error) {
        res.status(500).json({message: 'Error creating user'});
    }},
login: (req, res) => {
    const { email, password } = req.body;
    User.findOne({email: email}).then(user => {
        if(!user) {
            res.status(404).json({message: 'User not found'});
        }
        if(user.password === password) {
            user.password = "********";
            req.session.user = user;
            res.status(200).json({message: 'Login successful', account: user });
        } else {
            res.status(401).json({message: 'Incorrect password'});
        }}).catch(error => {
        res.status(500).json({message: 'Error logging in'});
    });
```

Listing 35: Handling User

For the login process, the email and password provided by the user are obtained and then checked to see if the email exists in the system. If the email exists, the password is checked for a match. If the

password is correct, the user is saved to the session and their information is returned to them. In case of an incorrect email or password, an error message will be returned to the user, typically a 401 or 500 error.

4.9.3 Handling Product

There are three primary methods used in the backend for product-related tasks: **GetAllProduct**, **GetProduct** and **GetProductByCategory**. First, **GetAllProduct** if there are GET methods on the pipeline, will filter the data by that field.

```
getAllProducts: async(req, res) => {
    try {
        const {Category, CategoryName, nameProduct} = req.query;
        Products.sort(() => Math.random() - 0.5);
        if (CategoryName) {
            const products = Products.filter(product =>
product.category.toLowerCase() == CategoryName.toLowerCase());
            return res.status(200).json(products);}
        if (Category) {
            const products = Products.filter(product =>
product.category.toLowerCase().includes(Category.toLowerCase()));
            return res.status(200).json(products);}
        if (nameProduct) {
            const products = Products.filter(product =>
product.nameProduct.toLowerCase().includes(nameProduct.toLowerCase()));
            return res.status(200).json(products);}
        return res.status(200).json(Products);
    }catch(err) {
        return res.status(500).json(err);}}
```

Listing 36: getAllProducts method

When the server receives a request, it invokes the getProduct function, which accepts two parameters: 'req' representing the request from the client and 'res' representing the response from the server. This function is defined with the async keyword, which allows the use of await commands to wait for asynchronous operations to complete.

In the try block, the code takes the value of the id parameter from the request and uses the find function to search for the product with the corresponding id in a list of products. The result is assigned to the variable product. The result is then returned to the client as JSON using the JSON method of the res. If an error takes place within the 'try' block, it is intercepted by the 'catch' block, and an error message is sent back to the client.

```
getProduct: async(req, res) => {
    try {
        const id = req.params.id;
        const product = Products.find(product => product.id == id);
        return res.status(200).json(product);
    }catch(err) {
        return res.status(500).json(err);
```

Listing 37: getProduct method

getProductByCategory works the same way as **getProduct**. But in **getProductByCategory**, the filter function is used to filter the products in the Products product list by the category with the corresponding name. The filtered list of products is assigned to the products variable.

```
getProductByCategory: async(req, res) => {
    try {
```

```
        const id = req.params.id;
        const CategoryName = id.toLowerCase();
        const products = Products.filter(product => product.category.toLowerCase()
== CategoryName);
        return res.status(200).json(products);
    }catch(err) {
        return res.status(500).json(err);
```

Listing 38: getProductByCategory method

4.9.4 Run the backend code

In the backend, a variable named **.env** is added to store essential data, such as configuration for the database. MongoDB online is used for convenient and easy access for many users.

- **tuantoi1000**: the username for the MongoDB Atlas cluster.
- **P6yFtqv8pYZcwARF**: the password for the MongoDB Atlas cluster.
- **cluster0.0nhdbyd.mongodb.net**: the hostname of the MongoDB Atlas cluster.
- **?retryWrites=true&w=majority**: connection options that specify retry behavior and the write concern.

```
MONGODB_URL =
mongodb+srv://tuantoi1000:P6yFtqv8pYZcwARF@cluster0.0nhdbyd.mongodb.net/?retryWrit
es=true&w=majority
```

Listing 39: MongoDB URL

The code uses the connect method of the Mongoose object to connect to the MongoDB database defined in the environment variable **process.env.MONGODB_URL**. This variable will contain the URL of the MongoDB, including the host address, port, database name, and login information if available.

After the connection is established, the code checks the connection status of the **mongoose.connection** object using the **readyState** property.

```
const mongoose = require('mongoose');
mongoose.connect(process.env.MONGODB_URL, () => {
    if (mongoose.connection.readyState === 1) {
        console.log("Connected to MongoDB");
} else {
        console.log("Error connecting to MongoDB " +
mongoose.connection.readyState);
```

Listing 40: Connect DB by Mongoose

Then register the routers for the web application and start the server to listen for incoming connections from the client.

```
app.use('/v1/product', productRoute);
app.use('/v1/user', userRoute);
let port = 8000;
app.listen(port, () => console.log('server is running in port ' + port));
```

Listing 41: Import routers and run code

5 EVALUATION

This is an e-commerce project that aims to develop a website with a modern interface and functionality, helping users have a new look at the interface, although each person has different views, with a personal perspective, this interface will offer a better complement of functionality and color. This is a website that specializes in selling fashion such as shoes and clothes, along with some famous and familiar brands. With a friendly and simple interface, users can access the website comfortably and without much difficulty in the selection and use process.

The screenshot below shows the main interface results of the website, along with two big and representative functions such as Dark Mode and Multilingual:

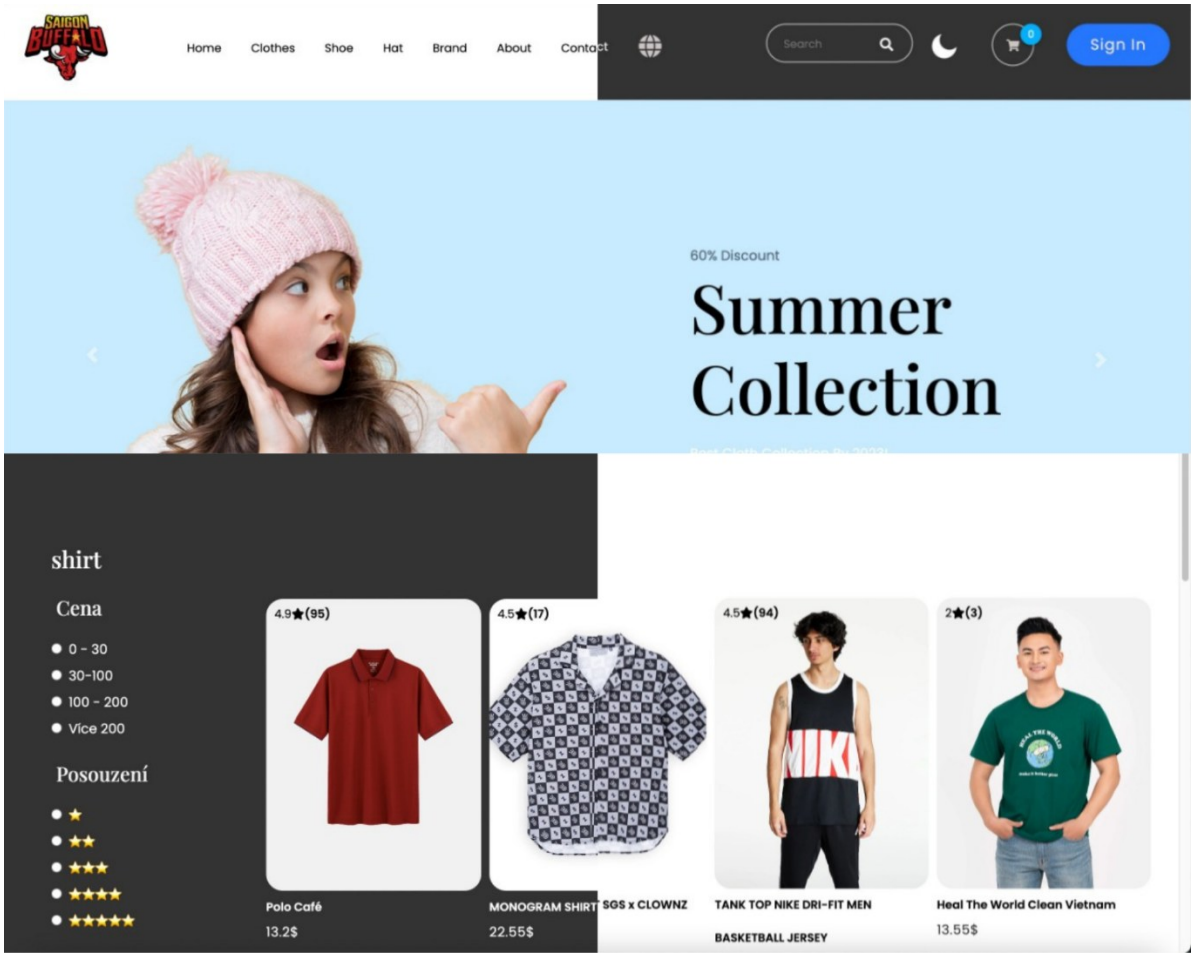

Figure 21: Saigon Buffalo e-commerce website with dark mode and multilanguage

After completing the e-commerce site, there are still many shortcomings and imperfections in this product. The speed of startup and display after activating **npm start** is still stable and guaranteed, users can access and use it, and the product page that loads data from the Back-end is still stable and without lag. Other features like Filter, More Product, Dark Mode, etc. also work well and can be appreciated.

However, some limitations have been identified on the website, such as an incomplete interface, some pages that are not yet perfect, and some minor bugs. As there has not been a user experience survey conducted yet, it cannot be confidently claimed that the e-commerce website is flawless.

During the project implementation, I encountered some problems such as adjusting the dark mode, some minor errors like Filter not showing results and using multiple languages, etc. But I tried to fix them as limited as possible and workable. However, in the shopping cart page, to enter the payment data, instead of having a separate page appear after checkout, I designed a shortened Modal component without jumping to another page, although simple, the interface is not beautiful and not fully required, along with some minor errors of some other pages, the Buy Now page is also a shortened modal of the Product Detail page that helps users to buy products quickly, but The interface is not perfect, the content is not different from the Product Detail page, so it is not completely outstanding. When making a purchase, it is not possible to put the product color in the shopping cart, this is also an error in the programming stage, currently the product color is only a form to provide the interface. Responsive design is still unstable and optimized on some devices, but the overview is still not fully functional, with errors appearing on some devices such as missing buttons, images being overwritten with text, missing menu when reloading, etc.

To address these limitations in the web, future work is to improve the site as it looks, improve the interface for mobile devices, because on mobile devices they are not optimized, add some other functionality, and edit the data in MongoDB. Since the knowledge of Backend like NodeJS, ExpressJS, and MongoDB was only learned after feeling the shortcomings and satisfying some functions, I could only improve a few of those missing functions in the Backend and connect them with ReactJS and I also want to move the product's data to the DB but haven't done it yet, but in the future I will improve better. Especially, there will be more focus on stable design and improved interface for mobile devices, because in the future users will use phones to work more instead of computers and laptops. In addition, if possible, I would also like to conduct a user experience survey with many people, and after collecting those comments, I can improve the website to become more diverse in appearance and function. Along with fixing minor bugs still outstanding in the project.

Overall, Saigon Buffalo's e-commerce website has been successful in construction, stability, and performance in all aspects. Our application provides a user-friendly interface, along with operational features that make it possible for users to use and experience like a real commercial website. Although some limitations and challenges have been identified in the project, in future work I hope to be able to address these issues and further improve giving users a commercial website more modern and complete.

6 CONCLUSION

Throughout this graduation thesis, I have analyzed, designed, and implemented the Saigon Buffalo e-commerce website. In the process, I have researched various technical and market factors and learned how to use React and other tools to design a basic e-commerce website with important features such as add to cart, calculate, and filter. Although my focus was on the front-end, I added a back-end to manage product and user data for a more realistic user experience.

The goal of this thesis was to provide users with a view of different frameworks so they can choose the right one for their needs. React was chosen as the primary framework due to its large community and diverse documentation, but other options such as Vue or Angular can also be combined with other technologies like Tailwind.

The topic of the Development of Modern Web Applications has been a fascinating area of study for me. Through this project, I have learned a great deal about various frameworks in JavaScript, despite the project having its shortcomings. However, I am confident that I will continue to improve my work in the future.

The main objective of this thesis is to equip users with a thorough understanding of various frameworks so that they can select the one that best suits their requirements. Although React, a JavaScript library (framework), has been personally selected for front-end design owing to its extensive community, diverse documentation, and ease of use and learning, there are other alternatives like Vue or Angular that can also be utilized alongside other technologies such as Tailwind to produce exceptional outcomes.

I hope that readers will gain an understanding of my design and implementation process and consider React as the primary framework to apply to their products. Although the product is not perfect, there are still some bugs in methods, functions, interfaces, etc. But what this project offers is an overview of interface design and how to use ReactJS. Web programming is a constantly evolving field, and many other frameworks will gradually emerge in the future, bringing users the most convenient tools for designing websites.

References

[1] C. Insight, "Front End vs. Back End: What's the Difference?," 17 August 2020. [Online]. Available: https://kenzie.snhu.edu/blog/front-end-vs-back-end-whats-the-difference/.

[2] S. Cherednichenko, "26 Hot Web Technologies and Web Design Trends to Watch in 2023," 03 September 2022. [Online]. Available: https://www.mobindustry.net/blog/26-hot-web-technologies-and-web-design-trends/.

[3] C. Team, "What Is a Framework?," 23 September 2021. [Online]. Available: https://www.codecademy.com/resources/blog/what-is-a-framework/.

[4] W3C, "HTML & CSS," [Online]. Available: https://www.w3.org/standards/webdesign/htmlcss#whatcss.

[5] D. G., "The Differences Between HTML and HTML5," [Online]. Available: https://www.hostinger.com/tutorials/difference-between-html-and-html5. [Accessed 14 April 2022].

[6] E. Elliott, Programming JavaScript Applications, First Edition ed., Sebastopol: O'Reilly Media, Inc.,, 2014, p. 253.

[7] A. B., "What is CSS," 31 March 2022. [Online]. Available: https://www.hostinger.com/tutorials/what-is-css.

[8] J. A., "What Is JavaScript? A Basic Introduction to JS for Beginners," 06 April 2022. [Online]. Available: https://www.hostinger.com/tutorials/what-is-javascript.

[9] D. Flanagan, JavaScript: The Definitive Guide, Master the World's Most-Used Programming, Seventh ed., Seventh, Ed., Sebastopol, CA: O'Reilly Media, Inc.,, 2020.

[10] Alex Banks, Eve Porcello, Learning React: Functional Web Development With React And Redux, First ed., Sebastopol, CA: O'Reilly Media, Inc.,, 2017.

[11] J. Duckett, JavaScript & JQuery: Interactive front-end web developmet, Indianapolis: John Wiley & Sons, Inc., 2014, p. 645.

[12] R. TIMBÓ, "Front End Frameworks: What They Are, and Best Options," 19 January 2023. [Online]. Available: https://www.revelo.com/blog/front-end-frameworks.

[13] R. Shankar, "Top 10 Best CSS Frameworks for Front-End Developers in 2023," 29 December 2022. [Online]. Available: https://hackr.io/blog/best-css-frameworks.

[14] M. Lawrence, "What is a CSS Framework?," 03 May 2019. [Online]. Available: https://medium.com/html-all-the-things/what-is-a-css-framework-f758ef0b1a11.

[15] S. K. Arora, "10 Best JavaScript Frameworks to Use in 2023," 29 December 2022. [Online]. Available: https://hackr.io/blog/best-javascript-frameworks.

[16] S. Khan, "WHAT IS A JAVASCRIPT FRAMEWORK?," [Online]. Available: https://generalassemb.ly/blog/what-is-a-javascript-framework/.

[17] J. A., "What Is React & How Does It Actually Work?," 07 December 2022. [Online]. Available: https://www.hostinger.com/tutorials/what-is-react.

[18] K. Chinnathambi, Learning React: A Hands-On Guide to Building Web Applications Using React and Redux PDF, 1 ed., M. Taber, C. Zahn and A. Manheim, Eds., United States of America: Addison-Wesley, 2016, p. 133.

[19] React, "Create a New React App," 10 January 2022. [Online]. Available: https://reactjs.org/docs/create-a-new-react-app.html.

[20] "Pros and Cons of ReactJS," [Online]. Available: https://www.javatpoint.com/pros-and-cons-of-react.

[21] R. Wieruch, The Road To React: Your journey to master plain yet pragmatic React, Berlin: Lean Publishing, 2020.

[22] J. A., "What is Bootstrap?," 18 March 2022. [Online]. Available: https://www.hostinger.com/tutorials/what-is-bootstrap/.

[23] A. P. Sofiya Merenych, "Web Development Trends and the Latest Web Technology Stacks in 2023," 10 January 2023. [Online]. Available: https://clockwise.software/blog/web-development-trends/.

[24] J. Davidson, "7 New Features to Look Out For in Bootstrap 5," 20 April 2021. [Online]. Available: https://www.makeuseof.com/new-features-to-look-out-for-in-bootstrap-5/.

[25] Kinsta, "What Is npm? An Introduction to Node's Package Manager," 26 October 2022. [Online]. Available: https://kinsta.com/knowledgebase/what-is-npm/.

[26] S. Singh, "What Is NPM? -A Simple English Guide to Truly Understanding the Node Package Manager," 14 October 2020. [Online]. Available: https://medium.com/swlh/what-is-npm-a-simple-english-guide-to-truly-understanding-the-node-package-manager-41e82f6c5515.

[27] Flavio Copes, Myles Borins, Onur Laru, Jean Gérard Bousiquot, Adam Miller, Ahmad Awais, "An introduction to the NPM package manager," [Online]. Available: https://nodejs.dev/en/learn/an-introduction-to-the-npm-package-manager/.

[28] D. P. Acharya, "The 40 Best JavaScript Libraries and Frameworks for 2023," 01 March 2023. [Online]. Available: https://kinsta.com/blog/javascript-libraries/.

[29] D. Robinson, "Reasons Why HTML5 Is The Future," 10 May 2020. [Online]. Available: https://www.topnotchdezigns.com/reasons-html5-future/.

[30] R. ElHousieny, "What Is Redux?," 30 January 2021. [Online]. Available: https://medium.com/swlh/what-is-redux-b16b42b33820.

[31] H. Johnson, "What is Node.js? A beginner's introduction to JavaScript runtime," [Online]. Available: https://dev.to/educative/what-is-nodejs-a-beginners-introduction-to-javascript-runtime-1iki#what-is.

[32] K. Kean, "What Is Express.js and Why Should You Use It?," 11 12 2021. [Online]. Available: https://www.makeuseof.com/what-is-express/.

[33] S. Hoque, Full-Stack React Projects: Learn MERN stack development by building modern web apps using MongoDB, Express, React, and Node.js, Second ed., BIRMINGHAM, MUMBAI: Packt, 2020.

[34] A. Christopher, "A Walkthrough of MongoDB," 12 October 2020. [Online]. Available: https://medium.com/analytics-vidhya/a-walkthrough-of-mongodb-aef402594144#:~:text=MongoDB%20is%20a%20JSON%20document,are%20like%20tables%20for%20MongoDB)..

[35] G. S. w. M. Atlas, "Colin Baird," 06 April 2020. [Online]. Available: https://medium.com/@colinbaird_51123/getting-started-with-mongodb-atlas-2b996d5be099.

[36] B. Simplified, "What is MERN Stack?," 15 January 2020. [Online]. Available: https://medium.com/@blockchain_simplified/what-is-mern-stack-9c867dbad302.

[37] D. Ceddia, "Do I need Node.js in the backend?," 08 June 2016. [Online]. Available: https://daveceddia.com/do-i-need-nodejs-backend-for-react-angular/.

[38] J. Koottala, "6 easy steps to localize your React Application — Internationalization with i18next," 20 November 2018. [Online]. Available: https://medium.com/@jishnu61/6-easy-steps-to-localize-your-react-application-internationalization-with-i18next-8de9cc3a66a1.

[39] D. Porter, "React Localization — The winner is i18next + i18nexus," 08 June 2020. [Online]. Available: https://javascript.plainenglish.io/react-localization-the-winner-is-i18next-i18nexus-b7cd9f14094e.