VIETNAM-KOREA UNIVERSITY OF INFORMATION AND  
COMMUNICATION TECHNOLOGY

**Faculty of Computer Science**



SPECIALIZED PROJECT

**BUILDING AN E-COMMERCE SYSTEM APPLYING MACHINE LEARNING TO RECOMMEND ITEMS**

Student: **NGUYEN DUC HAI – 20IT518**

**DAO DUC TRUONG – 20IT499**

Supervisor: Assoc. Prof. Ph.D Nguyen Thanh Binh

***Da Nang, December – 2023***

VIETNAM-KOREA UNIVERSITY OF INFORMATION AND

COMMUNICATION TECHNOLOGY

**Faculty of Computer Science**

SPECIALIZED PROJECT

**BUILDING AN E-COMMERCE SYSTEM APPLYING MACHINE LEARNING TO RECOMMEND ITEMS**

Student: NGUYEN DUC HAI

DAO DUC TRUONG

Supervisor: Assoc. Prof. Ph.D Nguyen Thanh Binh

***Da Nang, December - 2023***

SUPERVIOR’S COMMENT

*DaNang, December 16th, 2023*

Assoc. prof. ph.d Nguyen Thanh Binh

ACKNOWLEDGEMENTS

I would like to profoundly acknowledge the people who have helped me during my studies:

First of all, I would like to sincerely thank my research supervisors Assoc. Prof. Ph.D Nguyen Thanh Binh. Without his assistance and dedicated involvement in every step throughout the process, this paper could have never been accomplished.

I would like to thank the friends who were by my side and always gave me solutions to solve the problem in a simpler way during the specialized project.

Most importantly, none of this could have happened without my family. My mother, who offered her encouragement through phone calls every day. Whenever I am depressed and do not want to do my homework, my mother is the one who encourages me and helps me regain the energy to complete this specialized project well.

Finally, I want to thank the homeroom teacher and the class staff for caring and helping me to successfully complete the activities of the class as well as the assigned school.

Thank you sincerely!

TABLE OF CONTENTS

[SUPERVIOR’S COMMENT](#_Toc123987300)

[ACKNOWLEDGEMENTS](#_Toc123987301)

[TABLE OF CONTENTS i](#_Toc123987303)

[ABBREVIATIONS v](#_Toc123987304)

[LIST OF FIGURES vi](#_Toc123987305)

[LIST OF TABLES viii](#_Toc123987306)

[INTRODUCTION 1](#_Toc123987307)

[Problem statement 1](#_Toc123987308)

[Aims and Objectives 1](#_Toc123987309)

[Structure of the thesis 2](#_Toc123987310)

[CHAPTER 1. RESEARCH OVERVIEW 3](#_Toc123987311)

[1.1 Overview of React: 3](#_Toc123987312)

[1.1.1 What is React? 3](#_Toc123987313)

[1.1.2 Virtual DOM: 3](#_Toc123987314)

[1.1.3 What is JSX? 4](#_Toc123987315)

[1.1.4 Introduction to Components: 4](#_Toc123987316)

[1.1.5 Props and State: 5](#_Toc123987317)

[1.2 ReactJs: 5](#_Toc123987318)

[1.2.1 What is ReactJs: 5](#_Toc123987319)

[1.2.2 Advantages: 5](#_Toc123987320)

[1.2.3 Defect: 6](#_Toc123987321)

[1.2.4 Things to know about React Js: 6](#_Toc123987322)

[1.3 NodeJs: 7](#_Toc123987323)

[1.3.1 What is NodeJs: 7](#_Toc123987324)

[1.3.2 Features of nodejs: 8](#_Toc123987325)

[1.3.3 Advantages: 8](#_Toc123987326)

[1.3.4 Defect: 8](#_Toc123987327)

[1.4 Python: 9](#_Toc123987328)

[1.4.1 What is Python? 9](#_Toc123987329)

[1.4.2 Features of Python: 9](#_Toc123987330)

[1.4.3 Advantages: 10](#_Toc123987331)

[1.4.4 Defect: 10](#_Toc123987332)

[1.5 Recommender system: 10](#_Toc123987333)

[1.5.1 What is Recommender system? 10](#_Toc123987334)

[1.5.2 Steps to recommender products: 11](#_Toc123987335)

[1.5.3 Main types of recommender: 11](#_Toc123987336)

[1.5.4 Pearson correlation coefficient: 12](#_Toc123987337)

[1.5.5 What does Pearson coefficient mean? 13](#_Toc123987338)

[1.6 Implementation tools: 14](#_Toc123987339)

[1.6.1 Visual studio code: 14](#_Toc123987340)

[1.6.2 Xampp: 14](#_Toc123987341)

[1.7 Summary 15](#_Toc123987342)

[CHAPTER 2. SYSTEM ANALYSIS AND DESIGN 16](#_Toc123987343)

[2.1 Collaborative filtering recommenders: 16](#_Toc123987344)

[2.1.1 Algorithm description: 16](#_Toc123987345)

[2.1.2 Describe the approach: 17](#_Toc123987346)

[2.2 List of use case: 17](#_Toc123987347)

[2.3 Use case diagram: 19](#_Toc123987348)

[2.3.1 General use case: 19](#_Toc123987349)

[2.3.2 Specific description use case: 20](#_Toc123987350)

[2.3.2.1 Use case register, login: 20](#_Toc123987351)

[2.3.2.2 Use case search products: 21](#_Toc123987352)

[2.3.2.3 Use case order tracking: 22](#_Toc123987353)

[2.3.2.4 Use case see product details: 23](#_Toc123987354)

[2.3.2.5 Use case product recommend: 24](#_Toc123987355)

[2.3.2.6 Use case add product to cart: 25](#_Toc123987356)

[2.3.2.7 Use case cart management: 26](#_Toc123987357)

[2.3.2.8 Use case enter promo code: 27](#_Toc123987358)

[2.3.2.9 Use case product reviews and ratings: 28](#_Toc123987359)

[2.3.2.10 Use case personal account management: 29](#_Toc123987360)

[2.3.2.11 Use case order through the system: 30](#_Toc123987361)

[2.3.2.12 Use case user accounts management: 31](#_Toc123987362)

[2.3.2.13 Use case product management: 32](#_Toc123987363)

[2.3.2.14 Use case order management: 33](#_Toc123987364)

[2.3.2.15 Use case product category management: 34](#_Toc123987365)

[2.3.2.16 Use case promotion management: 35](#_Toc123987366)

[2.3.2.17 Use case comments and star ratings management: 36](#_Toc123987367)

[2.4 Class diagram: 37](#_Toc123987368)

[2.5 Activity diagram: 38](#_Toc123987369)

[2.5.1 Register function activity diagram: 38](#_Toc123987370)

[2.5.2 Login function activity diagram: 39](#_Toc123987371)

[2.5.3 Order function activity diagram: 40](#_Toc123987372)

[2.6 Sequence diagram: 41](#_Toc123987373)

[2.6.1 Register function sequence diagram: 41](#_Toc123987374)

[2.6.2 Login function sequence diagram: 42](#_Toc123987375)

[2.6.3 View product details function sequence diagram: 43](#_Toc123987376)

[2.6.4 Order function sequence diagram: 44](#_Toc123987377)

[2.6.5 Add favorite products function sequence diagram: 45](#_Toc123987378)

[2.7 Summary 45](#_Toc123987379)

[CHAPTER 3. IMPLEMENTATION AND BUILDING SYSTEM 46](#_Toc123987380)

[3.1 Build website interface with React Js: 46](#_Toc123987381)

[3.2 Use collaborative filtering suggestions to recommend products: 46](#_Toc123987382)

[3.3 Building a server system using Node Js: 48](#_Toc123987383)

[3.4 Build a database: 49](#_Toc123987384)

[3.5 E-commerce system interface: 49](#_Toc123987385)

[3.5.1 Customer interface: 49](#_Toc123987386)

[3.5.2 Admin interface: 58](#_Toc123987387)

[3.6 Summary 62](#_Toc123987388)

[CONCLUSIONS AND SUGGESTIONS 63](#_Toc123987389)

[Conclusions 63](#_Toc123987390)

[Suggestions: 64](#_Toc123987391)

[REFERENCES 65](#_Toc123987392)

ABBREVIATIONS

|  |  |
| --- | --- |
| **ABBREVIATIONS** | **MEANING** |
| DOM | Document Object Model |
| NaN | Not a Number |
| JSX | JavaScript XML |

LIST OF FIGURES

[Figure 1.1 - Logo React 3](#_Toc123250858)

[Figure 1.2 - Tree DOM 4](#_Toc123250859)

[Figure 1.3 - Logo NodeJs 7](#_Toc123250860)

[Figure 1.4 - Main types of recommender 11](#_Toc123250861)

[Figure 1.5 - Pearson correlation coefficient 12](#_Toc123250862)

[Figure 1.6 - Strength of the pearson correlation coefficient 13](#_Toc123250863)

[Figure 1.7 - Visual Studio Code 14](#_Toc123250864)

[Figure 1.8 - Logo Xampp 14](#_Toc123250865)

[Figure 2.1 - Determine the similarity between users 16](#_Toc123250866)

[Figure 2.2 - General usecase 19](#_Toc123250867)

[Figure 2.3 - Use case login, register 20](#_Toc123250868)

[Figure 2.4 - Use case search products 21](#_Toc123250869)

[Figure 2.5 - Use case order tracking 22](#_Toc123250870)

[Figure 2.6 - Use case see product details 23](#_Toc123250871)

[Figure 2.7 - Use case product recommend 24](#_Toc123250872)

[Figure 2.8 - Use case add product to cart 25](#_Toc123250873)

[Figure 2.9 - Use case for cart management 26](#_Toc123250874)

[Figure 2.10 - Use case enter promo code 27](#_Toc123250875)

[Figure 2.11 - Use case product reviews and ratings 28](#_Toc123250876)

[Figure 2.12 - Use case for personal account management 29](#_Toc123250877)

[Figure 2.13 - Use case order through the system 30](#_Toc123250878)

[Figure 2.14 - Use case user accounts management 31](#_Toc123250879)

[Figure 2.15 - Use case product management 32](#_Toc123250880)

[Figure 2.16 - Use case order management 33](#_Toc123250881)

[Figure 2.17 - Use case product category management 34](#_Toc123250882)

[Figure 2.18 - Use case promotion management 35](#_Toc123250883)

[Figure 2.19 - Use case comments and star ratings management 36](#_Toc123250884)

[Figure 2.20 - Class diagram 37](#_Toc123250885)

[Figure 2.21 - Register function activity diagram 38](#_Toc123250886)

[Figure 2.22 - Login function activity diagram 39](#_Toc123250887)

[Figure 2.23 - Order function activity diagram 40](#_Toc123250888)

[Figure 2.24 - Register function sequence diagram 41](#_Toc123250889)

[Figure 2.25 - Login function sequence diagram 42](#_Toc123250890)

[Figure 2.26 - View product details function sequence diagram 43](#_Toc123250891)

[Figure 2.27 - Order function sequence diagram 44](#_Toc123250892)

[Figure 2.28 - Add favorite products function sequence diagram 45](#_Toc123250893)

[Figure 3.1 - Install the required libraries for ReactJS 46](#_Toc123250894)

[Figure 3.2 - Install the required libraries for Python 46](#_Toc123250895)

[Figure 3.3 - Use the library Flask 46](#_Toc123250896)

[Figure 3.4 - Configure database connection 46](#_Toc123250897)

[Figure 3.5 - Database query 47](#_Toc123250898)

[Figure 3.6 - Configure routes 47](#_Toc123250899)

[Figure 3.7 - Create a matrix and process the NaN values 47](#_Toc123250900)

[Figure 3.8 - Calculate the correlation coefficient using corr(method='pearson') 47](#_Toc123250901)

[Figure 3.9 - The formula for calculating the correlation coefficient Pearson 47](#_Toc123250902)

[Figure 3.10 - Use Json library to return data in Json format 48](#_Toc123250903)

[Figure 3.11 - Configure database connection 48](#_Toc123250904)

[Figure 3.12 - Install the necessary libraries for node JS 48](#_Toc123250905)

[Figure 3.13 - Tables in the database 49](#_Toc123250906)

[Figure 3.14 - Home page interface 49](#_Toc123250907)

[Figure 3.15 - Product type interface 50](#_Toc123250908)

[Figure 3.16 - Login page interface 50](#_Toc123250909)

[Figure 3.17 - Register page interface 51](#_Toc123250910)

[Figure 3.18 - Email verification page interface 51](#_Toc123250911)

[Figure 3.19 - Favourite page interface 52](#_Toc123250912)

[Figure 3.20 - Search page interface 52](#_Toc123250913)

[Figure 3.21 - Product detail page interface 53](#_Toc123250914)

[Figure 3.22 - Comment interface 53](#_Toc123250915)

[Figure 3.23 - Product suggestions interface 54](#_Toc123250916)

[Figure 3.24 - Shopping cart interface 54](#_Toc123250917)

[Figure 3.25 - Payment page interface 55](#_Toc123250918)

[Figure 3.26 - Paypal payment interface 55](#_Toc123250919)

[Figure 3.27 - Vnpay payment interface 56](#_Toc123250920)

[Figure 3.28 - Profile page interface 56](#_Toc123250921)

[Figure 3.29 - Orders invoice interface 57](#_Toc123250922)

[Figure 3.30 - Email verification interface 57](#_Toc123250923)

[Figure 3.31 - Overview interface 58](#_Toc123250924)

[Figure 3.32 - Account users manage interface 58](#_Toc123250925)

[Figure 3.33 - Edit user information interface 59](#_Toc123250926)

[Figure 3.34 - Account admin manage interface 59](#_Toc123250927)

[Figure 3.35 - Edit admin infomation interface 60](#_Toc123250928)

[Figure 3.36 - Add account admin interface 60](#_Toc123250929)

[Figure 3.37 - Catagory manage interface 61](#_Toc123250930)

[Figure 3.38 - Edit category information interface 61](#_Toc123250931)

[Figure 3.39 - Add category interface 62](#_Toc123250932)

LIST OF TABLES

[Table 2.1 - List of use case 17](#_Toc123249591)

[Table 2.2 - Specific description use case login, register 20](#_Toc123249592)

[Table 2.3 - Specific description use case search products 21](#_Toc123249593)

[Table 2.4 - Specific description use case order tracking 22](#_Toc123249594)

[Table 2.5 - Specific description use case see product detail 23](#_Toc123249595)

[Table 2.6 - Specific description use case product recommend 24](#_Toc123249596)

[Table 2.7 - Specific description use case add product to cart 25](#_Toc123249597)

[Table 2.8 - Specific description use case for cart management 26](#_Toc123249598)

[Table 2.9 - Specific description use case enter promo code 27](#_Toc123249599)

[Table 2.10 - Specific description use case product reviews and ratings 28](#_Toc123249600)

[Table 2.11 - Specific description use case personal account management 29](#_Toc123249601)

[Table 2.12 - Specific description use case order through the system 30](#_Toc123249602)

[Table 2.13 - Specific description use case user accounts management 31](#_Toc123249603)

[Table 2.14 - Specific description use case product management 32](#_Toc123249604)

[Table 2.15 - Specific description use case order management 33](#_Toc123249605)

[Table 2.16 - Specific description use case product category management 34](#_Toc123249606)

[Table 2.17 - Specific description use case promotion management 35](#_Toc123249607)

[Table 2.18 - Specific description use case comment and ratings management 36](#_Toc123249608)

INTRODUCTION

## Problem statement

E-commerce provides an easy way to sell products to a large customer base. However, there is a lot of competition among multiple e-commerce sites. When users land on an e-commerce site, they expect to find what they are looking for quickly and easily. Also, users are not sure about the brands or the actual products they want to purchase. They have a very broad idea about what they want to buy. Many customers nowadays search for their products on Google rather than visiting specific e-commerce sites. They believe that Google will take them to the e-commerce sites that have their product.

The purpose of any e-commerce website is to help customers narrow down their broad ideas and enable them to finalize the products they want to purchase.

## Aims and Objectives

Online business and sales have never been as exciting and powerful as today. When millions of people around the world use the Internet and consider it an ideal place to find information, purchase goods and services, the website is considered a highly effective online marketing strategy besides social networking site. This has pulled a series of e-commerce websites into existence to serve the shopping needs of a large number of users.

So I want to develop an HLE e-commerce system that can bring convenience to customers. The complete system is geared towards 2 types of users namely customers and admin:

* Customers: Allow customers to order, necessary products from the system's product catalog, with no quantity limit.
* Administrator: Manage the HLE e-commerce system.

## Structure of the thesis

After the *Introduction*, the thesis is structured in three chapters:

*Chapter 1,* *Research overview.* In this chapter, the thesis presents the concepts of language, tools and technology used in this e-commerce system.

*Chapter 2, System analysis and design.* This chapter includes the algorithm analysis of collaborative filtering recommenders and use case diagrams, class diagrams, activity diagrams, sequence diagrams of the main functions of the system.

*Chapter 3, Implementation and building system.* This chapter will be how to implement the database, how to use the commands and libraries to be able to run this e-commerce system.

Finally, there are *Conclusions*, *Suggestions,* *References* and *Appendices* related to the topic.

# RESEARCH OVERVIEW

## Overview of React:

### What is React?

React is a UI library developed at Facebook to support building highly interactive, stateful, and reusable UI components. One of the appeals of React is that this library not only works on the client side, but is also rendered on the server and can be interconnected. React compares the change between the values of this render with the previous render and updates the DOM with the least change.

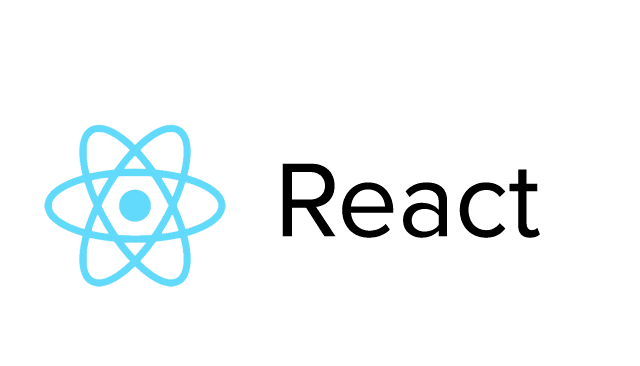


Figure 1.1 - Logo React

### Virtual DOM:

Virtual DOM technology enhances application performance. The fact that only the root node has state and when it changes, it will completely restructure, which means that the DOM tree will also have to change part of it, which will affect processing speed. React JS uses Virtual DOM (virtual DOM) to improve this problem. Virtual DOM is a Javascript object, each object contains all the information needed to create a DOM, when the data changes it will calculate the change between the object and the real tree, this will help optimize re- render the real DOM tree.

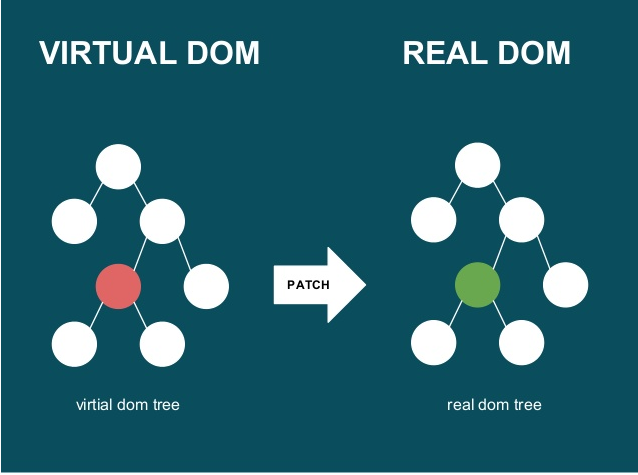


Figure 1.2 - Tree DOM

React uses one-way data binding mechanism – one-way data flow. Data is passed from parent to child via props. Simple data flow makes it easy to control as well as correct errors. With the above characteristics, React is used to build large applications whose data changes continuously over time. Data changes are mostly accompanied by changes in the interface.

### What is JSX?

JSX is a language that allows writing HTML code in Javascript. Features: Faster: Faster. JSX performs optimization while compiling to Javacscript code. These codes give a much faster execution time than an equivalent code written directly in Javascript. Safer: more secure. In contrast to Javascript, JSX is statically-typed, meaning it is compiled before runtime, just like Java and C++. So errors will be detected during compilation. In addition, it also provides very good debug when compiling.

### Introduction to Components:

React is built around components, not templates like other frameworks. In React, we build websites using small components. We can reuse a component in many places, with different states or properties, in a component that can contain another component. Each component in React has its own, mutable state, and React will perform component updates based on state changes. Everything React is a component. They help maintain code when working on large projects. A simple react component just needs a render method. There are many other methods available, but render is the dominant method.

### Props and State:

Props: helps components interact with each other, the component takes input called props, and returns properties that describe what the child component should render. Prop is immutable.

State: represents the state of the application, when the state changes, the component also re-renders to update the UI.

## ReactJs:

### What is ReactJs:

React Js is a library written in javascript, used to build user interface (UI) UI here is used mainly in two Web and Mobile platforms. React is widely used and has a rich and diverse ecosystem. Open source JavaScript and the father of ReactJS is a big guy with a name that everyone knows is Facebook. The purpose of creating ReactJS was to create attractive web applications with high speed and efficiency with minimal coding. And the main purpose of ReactJS is that every website when using ReactJS must run smoothly, quickly, and be highly scalable and simple to implement.

### Advantages:

ReactJS creates for itself a virtual DOM, which is where components live. Creating such DOM greatly improves performance, when there is a calculation that needs to be changed or needs to be updated on the DOM, ReactJS all calculates it first and the rest is just to do them on the DOM. this will help ReactJS avoid the necessary manipulations on a DOM without any additional cost.

ReactJS writing JS code will become easier because it uses a special syntax that is JSX syntax which means this syntax allows us to mix HTML and Javascript code. In addition, we can also include additional code in the render function without having to do string concatenation and this is considered one of the interesting features of ReactJS and converting HTML fragments into constructors. are all implemented from the main converter which is JSX.

SEO friendly, this is one of the special things and only ReactJS this is also a big problem for JS Frameworks because most JS Frameworks are not search engine friendly even though they have been improved. lots of. As for reactJS, it is quite proud because it is not in the group that is not SEO friendly because under the support of rendering and returning the browser as a webpage when you run ReactJS on the server and the Virtual DOM. It is for this reason that React can fully meet the SEO Friendly nature.

### Defect:

Reactjs only serves the View layer. React is just a View Library, it's not an MVC framework like other frameworks. This is just a Facebook library to help render the view. So React will not have a Model and Controller part, but must combine with other libraries. React also won't have 2-way binding or Ajax.

Integrating Reactjs into traditional MVC frameworks requires reconfiguration.

React is quite heavy compared to other frameworks React is similar in size to Angular (About 35kb compared to 39kb of Angular). Whereas Angular is a complete framework.

### Things to know about React Js:

ReactJs supports building UI components that are highly interactive, stateful, and reusable at any time. As a place to build around components, when active React works on both the client and renders on the server so that these two parts can connect. One more special thing is that react is not an MVC Framework so don't be surprised that react doesn't have a model and controller, and when you need to manipulate it, you need to combine it with other libraries.

Using less State components can say that the State is the place to keep the soul of the application, especially we should keep the State simple because react with applications is always expanding continuously. State will make the test more difficult and State is concerned with rendering the display has initialized the data for the State or not and has changed and whether or not the render is back. In addition, State only exists in a component with the exchange of data with the outside and using state is unnecessary, remember you can only use state when needed and reflect the correct state of the Component. .

Combined with Redux.js actually react is just a view so react should be combined with redux, flux or any other data flow is always essential and redux is one of the data sources used by many people and react's thinking is also quite good.

You can also use JSX, ES6, Babel, Webpack, and NPM. Where JSX is one of the best features of React, everything we write is what shows up and when you combine with babel compiles and takes advantage of the new features of ES6 as well. two factors NPM and webpack will play a role in helping the process of packaging and utilizing libraries more fully.

## NodeJs:

### What is NodeJs:

Nodejs is a platform independent development (Platform) built on top of Chrome's Javascript Runtime with which we can build network applications quickly and easily. Nodejs creates applications with fast processing speed, real-time real-time. Nodejs applies to products with high traffic, needs to scale quickly, need to innovate technology, or create Startup projects as quickly as possible.



Figure 1.3 - Logo NodeJs

NodeJS can be used to build different types of applications such as command line applications, web applications, real-time chat applications, REST API servers, etc. However, NodeJS is commonly used mainly. to build network programs as web servers, similar to PHP, Java, or ASP.NET.

### Features of nodejs:

NodeJS does not need to wait for the API to return data, so any APIs in the NodeJS library are not synchronized. This is a Platform, not a Framework. Therefore, NodeJS allows you to build websites independently and faster. NodeJS can run on multiple platforms including: Window, MacOS, Linux. NodeJS is considered a single-threaded server and cannot support multi-threading.

NodeJS is not considered a programming language, so newbies must have a solid grasp of basic programming knowledge such as protocols, Javascript, etc. to be able to use NodeJS. However, the NodeJS community is usually very large, ready to support you anytime, anywhere. The core part of NodeJS is usually known in C++ language, so it's performance and processing speed is relatively high. As a result, most NodeJS applications are capable of responding to real-time running on cross-platform, multi-device, etc.

### Advantages:

In terms of execution speed and scalability. Node.js is very fast. That's a pretty important requirement when you're a startup trying to create a large product and want to ensure it can scale quickly and accommodate a large number of users as your site grows.

Asynchronous event-driven IO helps handle multiple concurrent requests. Meet real-time requirements. Has extremely fast speed, meeting the usage needs of 'huge' visitors in a short time.

Use JavaScript, a programming language that is very easy to learn. Share the same piece of code with both server and client side.

Npm and its modules are very powerful and still evolving. There is a strong community, there is a lot of code shared via github. Compatible with many devices, many operating systems such as MacOS, Window, Linux, ...

### Defect:

Like most new technologies, implementing Node.js on the host is not an easy thing. If you have a shared web hosting, you cannot simply load up a Node.js application and expect it to work fine. VPS and dedicated server are a better choice - you can install Node.js on them.

Another major drawback of Node.js is that it is still in the early stages of development, which means some features will change in subsequent development. In fact, if you read the articles it includes a stability index, which shows how risky you are using existing features.

NodeJS offers no scalability and cannot take advantage of the many cores commonly found in today's server-grade hardware. Working with relational databases is a pain if you are using NodeJS.

Each use of the callback will end up with lots of nested callbacks. If you don't understand JavaScript well, you will have a hard time with NodeJS. NodeJS is not suitable for CPU-intensive tasks but only for I/O like web servers.

## Python:

### What is Python?

Python is a programming language widely used in web applications, software development, data science, and machine learning. Developers use Python because it's efficient, easy to learn, and can run on a variety of platforms.

### Features of Python:

Python is an interpreted language, which means that it runs code line by line directly. If there is an error in the program code, it will stop running. Therefore, the programmer can quickly find the error in the code.

Python uses the same word as in English. Unlike other programming languages, Python does not use braces. Instead, this language uses indentation.

Programmers do not need to declare variable types when writing code because Python will determine them at run time. So you can write Python programs more quickly.

Python is closer to the human language than other programming languages. Therefore, programmers do not need to worry about its basic functions such as architecture and memory management.

### Advantages:

Developers can easily read and understand a Python program because the language has the same basic syntax as English.

Python improves the productivity of developers because compared to other languages, they can use fewer lines of code to write a Python program.

Python has a large standard library that contains many lines of reusable code for almost any task. As a result, developers won't need to write code from scratch.

Developers can easily use Python with other popular programming languages like Java, C, and C++.

The active Python community includes millions of enthusiastic developers around the world. If you run into problems, you'll be able to get quick support from the community.

There are many useful resources on the Internet if you want to learn Python. For example, you can easily find developer videos, tutorials, documentation, and guides.

Python can be used on many different computer operating systems, such as Windows, macOS, Linux, and Unix.

### Defect:

Although it is a faster language than PHP, Python is quite slow compared to C/C++ and JavaScript.

Only use Python to create programs that work on the web platform, but cannot integrate program development on mobile devices.

Does not contain properties such as public, protected or private. In particular, do...while loops cannot be performed with switch...case.

## Recommender system:

### What is Recommender system?

The Recommender System is one of the most popular applications of data science today. They are used to predict the rating or preference a user will spend on an item. Almost every major tech company has adopted them in some form.

### Steps to recommender products:

Step 1: Find features that influence user ratings, through data exploration and analysis

Step 2: Analyze and apply suitable filtering algorithm

Step 3: Conduct model training

### Main types of recommender:

Simple Recommenders: Offers general recommendations for all users, based on popularity and/or movie genre. The basic idea behind this system is that movies that are more popular and critically acclaimed have a higher probability of being liked by the average audience. An example might be the IMDB Top 250.

Content-based Recommenders: Recommend similar items based on a specific item. This system uses item metadata, such as genre, director, description, actors, and more. for movies, to make these recommendations. The general idea behind these referral systems is that if a person likes a particular item, they will also like an item similar to it. And to recommend that, it will use the user's past item metadata. A good example might be YouTube, where based on your history it suggests you new videos you can watch.

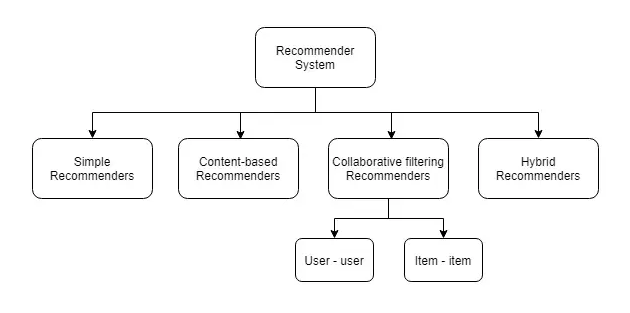


Figure 1.4 - Main types of recommender

Collaborative filtering Recommenders: These systems are widely used and they try to predict the “ratings” or “preference” that a user will give an item based on past ratings and preferences of other users. Collaborative filters do not require item metadata like their content-based filters do.

Hybrid Recommenders: Hybrid Filtering is a combination of two algorithms Content-based Filtering and Collabrative Filtering: Hybrid Fitering is used flexibly when the Collabrative Filtering system has no behaviors (ratings), then the system will use Content. -based Filtering and vice versa, when Content-based Filtering does not have the necessary features in the evaluation, the system will use Collaborative Fitering instead.

### Pearson correlation coefficient:

The correlation coefficient is a statistical indicator that measures the strength and weakness of the relationship between two variables.

The correlation coefficient has a value from -1.0 to 1.0. A calculated result greater than 1.0 or less than -1 means there is an error in the correlation measurement.

* A negative correlation coefficient shows that two variables have an inverse or negative relationship (absolutely inverse when the value is -1).
* A positive correlation coefficient indicates a positive or positive relationship (absolute covariance when the value is equal to 1).
* Correlation is zero for two independent variables.

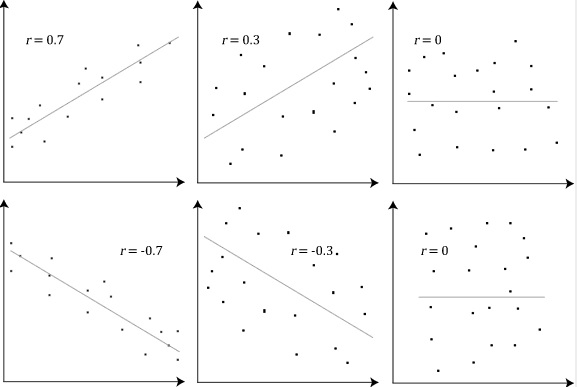


Figure 1.5 - Pearson correlation coefficient

### What does Pearson coefficient mean?

Pearson correlation coefficient (r) fluctuates in the continuous range from -1 to +1:

* r = 0: Two variables have no linear correlation
* r = 1; r = -1: Two variables have an absolute linear relationship.
* r < 0: Negative correlation coefficient. That is, the value of variable x increases, the value of variable y decreases and vice versa, the value of variable y increases, the value of variable x decreases.
* r > 0: Positive correlation coefficient. That is, the value of variable x increases, the value of variable y increases and vice versa, the value of variable y increases, the value of variable x also increases.
* If r is between 0.50 and ±1, it is said to be strongly correlated.
* If r is between 0.30 and ±0.49, then it is called the mean correlation.
* If r is below ±.29, then it is called a weak correlation.
* On a scatter plot, if r = -1 the data will be distributed on a straight line with a negative slope, r = 1 the data will be distributed on a straight line with a positive slope.

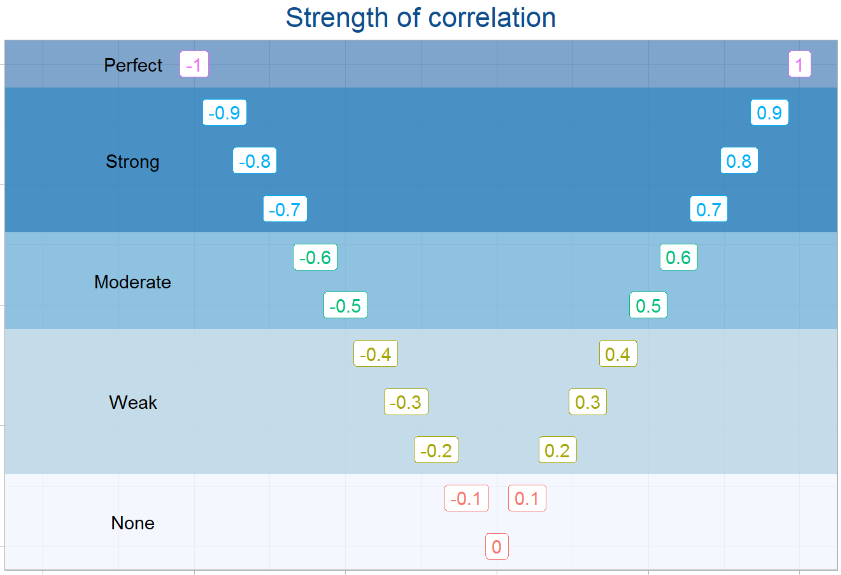


Figure 1.6 - Strength of the pearson correlation coefficient

## Implementation tools:

### Visual studio code:

Visual Studio Code is a code editor developed by Microsoft for Windows, Linux and macOS. It supports debugging, comes with Git, has syntax highlighting, smart code completion, snippets, and code enhancements.



Figure 1.7 - Visual Studio Code

### Xampp:

Xampp is a Web server creation program with built-in Apache, PHP, MySQL, FTP Server, Mail Server and tools like phpMyAdmin. Unlike Appserv, Xampp has a pretty convenient management program, allowing you to proactively turn on or off server services at any time.

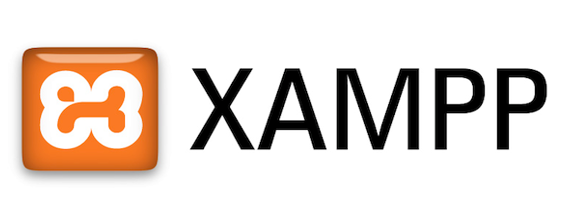


Figure 1.8 - Logo Xampp

MySQL is the world's most popular free and open source database management system and is very popular with developers in the application development process. Because MySQL is a high-speed, stable and easy-to-use, portable, multi-operating system that provides a large set of very powerful utility functions. With high speed and security, MySQL is well suited for applications that access databases on the internet. It has many versions for different operating systems: Win32 version for Windows series operating systems, Linux, Mac OS X, Unix, FreeBSD, NetBSD, Novell NetWare, SGI Irix, Solaris, SunOS, ...

MySQL is used to support PHP, Perl, and many other programming languages, as it stores the information of web pages written in PHP or Perl.

## Summary

*In this chapter, we have learned the concepts of React, ReactJs, NodeJs, Python. Know the features of the language so that you can understand more about them. Know their advantages and disadvantages so that we have the most general overview.*

*The focus is on the concept of recommender systems and recommender systems include four main categories: Simple Recommenders, Content-based Recommenders, Collaborative filtering Recommenders, and Hybrid Recommenders.*

*Learn about the tools needed to build this e-commerce system.*

# SYSTEM ANALYSIS AND DESIGN

## Collaborative filtering recommenders:

### Algorithm description:

The basic idea of this algorithm is to predict a user's preference for an item based on other users that are similar to the user in question. Determining the similarity between users can be based on the level of interest (rating) of these users with other items that the system has known in the past.

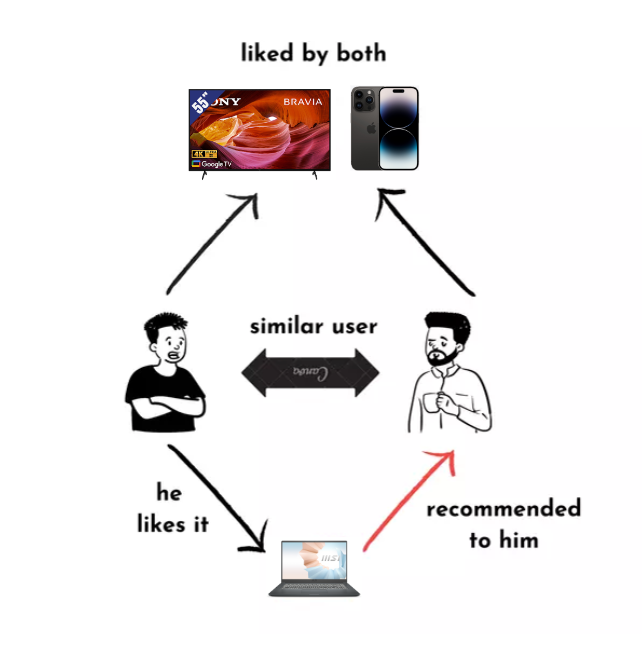


Figure 2.1 - Determine the similarity between users

For example: Two users A and B both like Acer laptop products (ie, they both rate these acer laptop products 4 -> 5 stars). Based on B's purchase history, we see that B bought an LG TV, so it is likely that A also likes this product, from which the system will recommend LG TV to A.

### Describe the approach:

Method one is to determine each user's interest in an item based on the interest of similar users in that item, also known as User-user collaborative filtering.

Method two is that instead of defining user similarities, the system determines item similarities. From there, the system suggests items that are close to the items that the user has a high level of interest in.

## List of use case:

Table 2.1 - List of use case

|  |  |  |
| --- | --- | --- |
| **STT** | **usecase** | **action** |
| 1 | Customer | * Register, account verification * Login * View products (Filter products by each attribute) * Search product * Manage your orders   + Processing   + Delivering   + Completed   + Canceled   + Return/Refund * Review products with stars and content * Chatbox * Add favorite product * Add product to cart * Enter promo code * Shopping cart management * Method payment   + Cash on delivery   + Vnpay   + Paypal |
| 2 | Admin | * Overview   + Order statistics for the day   + Order statistics for the month   + Total revenue for the month   + Order statistics by column chart   + Order statistics by pie chart * Account manage   + User manage     - Add user     - Edit status (active, lock, not verified)     - Edit information user     - Delete user   + Admin manage     - Add admin     - Edit status (active, lock)     - Edit information admin     - Delete admin * Categories manage   + Add category   + Edit information category   + Delete category * Product manage   + Add product   + Edit information product   + Delete product * Receipt manage   + Export receipt   + Edit information receipt   + Delete receipt * Warehouse manage   + Enter the product into the warehouse   + Edit the number of products in stock * Special event manage   + Create discount events   + Edit event time   + Delete event * Product reviews manage   + Delete review |

## Use case diagram:

### General use case:

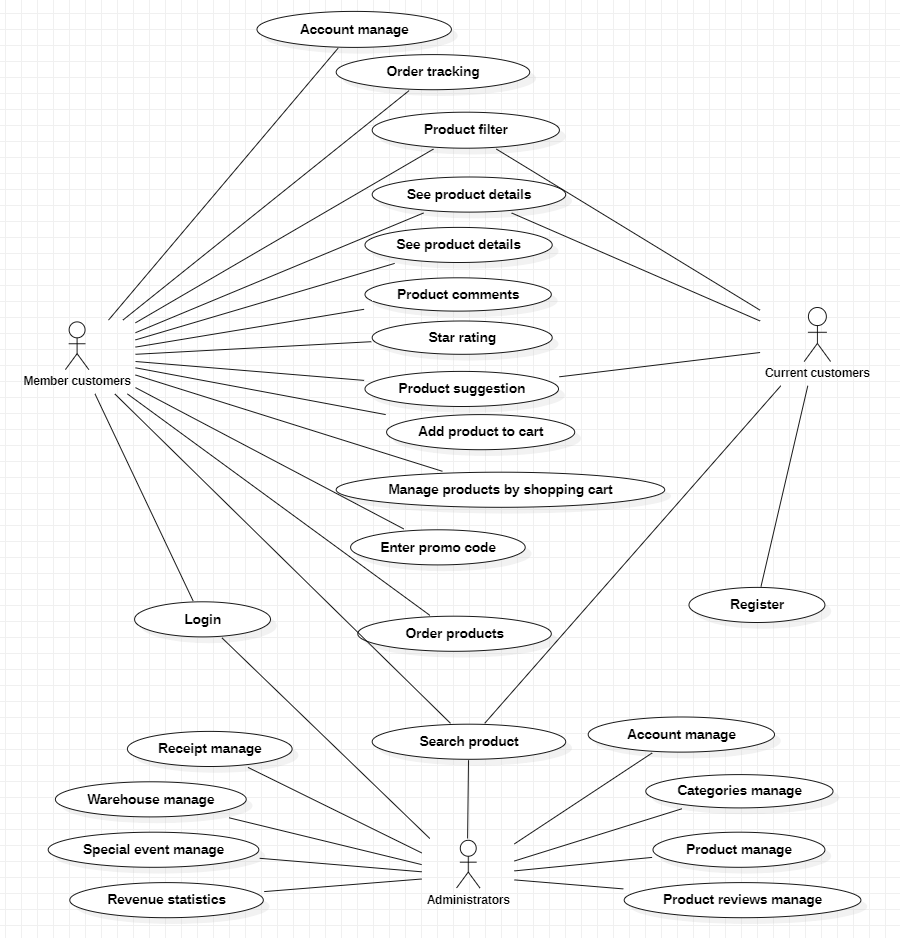


Figure 2.2 - General usecase

### Specific description use case:

#### Use case register, login:

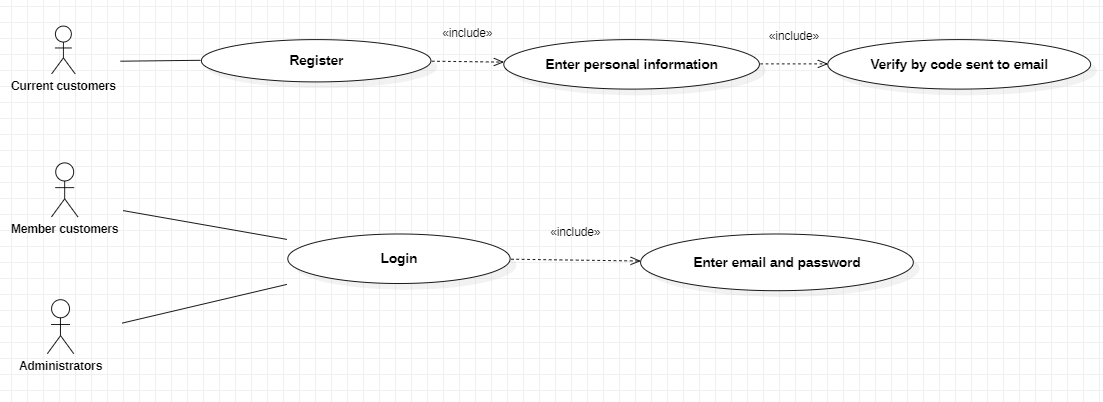


Figure 2.3 - Use case login, register

Table 2.2 - Specific description use case login, register

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Login | |
| Agent | Member customers | |
| Pre-conditions | You must register and verify your account before you can log in | |
| The following conditions |  | |
| Description | Users log into the system to use the application | |
| Action | Agent action | System action |
| (1) Enter email and password | (2) Get information for testing  (3) If successful, login to the system. If failed error message. |

#### Use case search products:

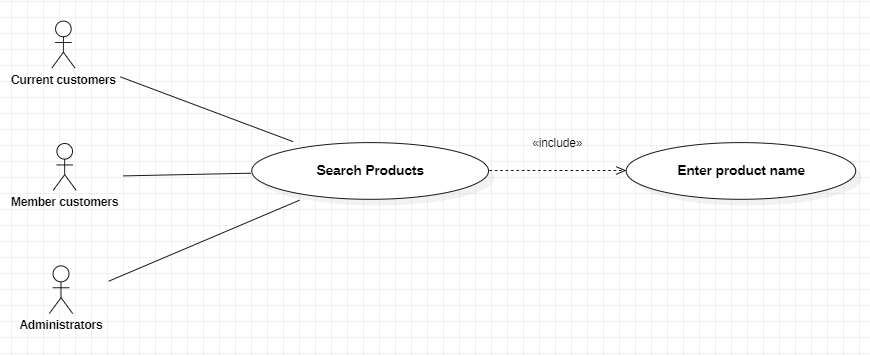


Figure 2.4 - Use case search products

Table 2.3 - Specific description use case search products

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Search products | |
| Agent | Member customers, curent customers, administrators | |
| Pre-conditions |  | |
| The following conditions | Return product information | |
| Description | Actors enter the product name to be searched, and the system displays the product information to the agents | |
| Action | Agent action | System action |
| (1) Agent enters product name | (2) Get information from input  (3) The system conducts a database query, If a product is found, then inform the agent. If not found message agent no product |

#### Use case order tracking:

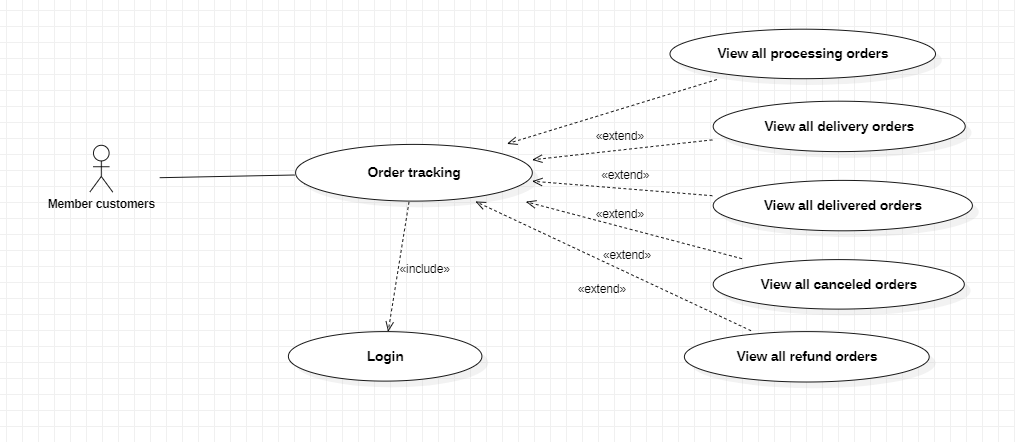


Figure 2.5 - Use case order tracking

Table 2.4 - Specific description use case order tracking

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Order tracking | |
| Agent | Member customers | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions | Show order information | |
| Description | Customers track order information such as viewing orders in progress, orders in progress, delivered orders, canceled orders, refund orders | |
| Action | Agent action | System action |
| (1) User selects account item | (2) Display the information from the database by customer id. If there are no orders, the system will tell you that you have no orders |

#### Use case see product details:

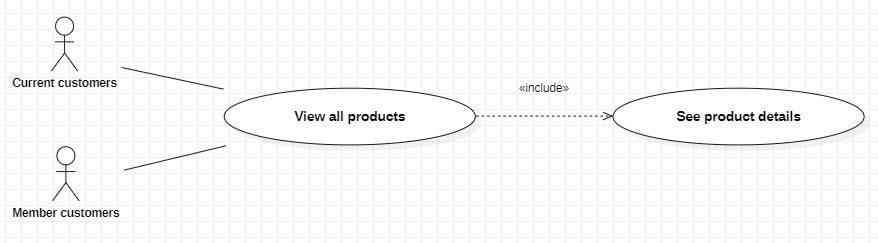


Figure 2.6 - Use case see product details

Table 2.5 - Specific description use case see product detail

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | See product details | |
| Agent | Member customers, Current customers | |
| Pre-conditions |  | |
| The following conditions | Show product information | |
| Description | Agent selects products, the system displays information about products | |
| Action | Agent action | System action |
| (1) The agent clicks on the product | (2) The system queries in the database by product id and displays product information to the user. |

#### Use case product recommend:

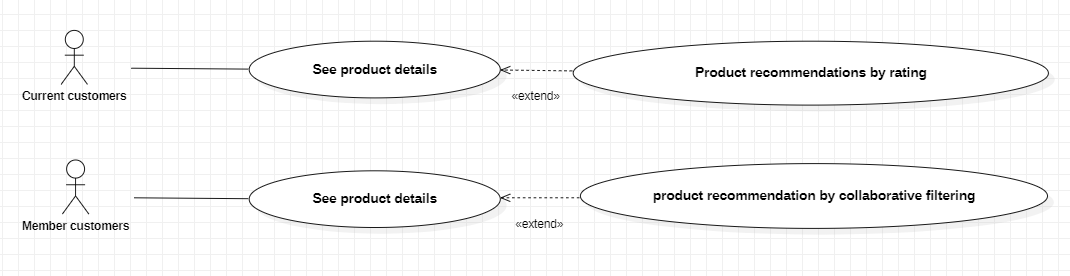


Figure 2.7 - Use case product recommend

Table 2.6 - Specific description use case product recommend

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | See product recommendation | |
| Agent | Member customers | |
| Pre-conditions | Successfully logged into the system and rated at least one product | |
| The following conditions | Show product information | |
| Description | The agent selects the product to enter the product details. Here the recommender system will suggest products that are close to your interests. | |
| Action | Agent action | System action |
| (1) The agent clicks on the product | (2) Recommender system will suggest products that are close to your interests. |

#### Use case add product to cart:

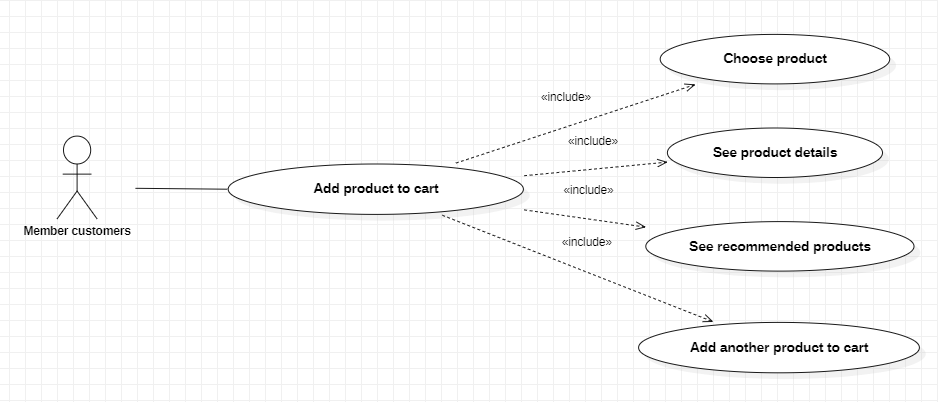


Figure 2.8 - Use case add product to cart

Table 2.7 - Specific description use case add product to cart

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Add product to cart | |
| Agent | Member customers | |
| Pre-conditions | Select a product, view product details, view product recommend | |
| The following conditions | Products displayed in cart | |
| Description | Customer selects products to add to cart | |
| Action | Agent action | System action |
| (1) On the product detail page, the agent chooses to add a shopping cart. | (2) The system understands that the cart has been successfully added and saved to localstorage. |

#### Use case cart management:

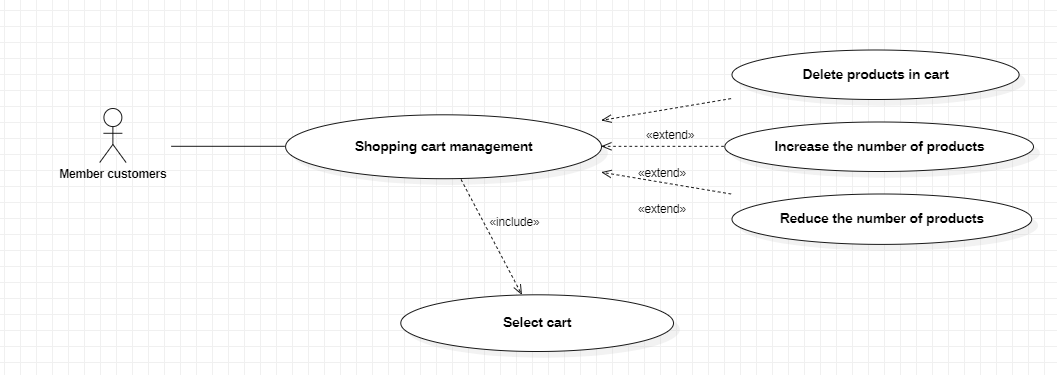


Figure 2.9 - Use case for cart management

Table 2.8 - Specific description use case for cart management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Add product to cart | |
| Agent | Member customers | |
| Pre-conditions | Select a product, view product details, view product recommend | |
| The following conditions | Products displayed in cart | |
| Description | Customer selects products to add to cart | |
| Action | Agent action | System action |
| (1) On the product detail page, the agent chooses to add a shopping cart. | (2) The system understands that the cart has been successfully added and saved to localstorage. |

#### Use case enter promo code:

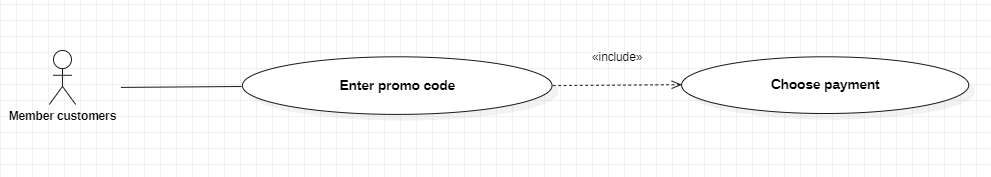


Figure 2.10 - Use case enter promo code

Table 2.9 - Specific description use case enter promo code

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Enter promo code | |
| Agent | Member customers | |
| Pre-conditions | Log in to your account, select cart products, enter all information and select checkout | |
| The following conditions | Discount for products according to events from the system | |
| Description | The customer enters all the information and chooses to pay. If the code is valid, the product will be discounted, if it is not, it will be notified and the discount will not be given. | |
| Action | Agent action | System action |
| (1) Select checkout on the cart page  (3) Enter the promo code input. | (2) The system displays a page to check personal information, product information, input promo code.  (4) The system checks the code and notifies the customer.  (5) If successful, notify the amount of the promotion |

#### Use case product reviews and ratings:



Figure 2.11 - Use case product reviews and ratings

Table 2.10 - Specific description use case product reviews and ratings

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Product reviews and ratings | |
| Agent | Member customers | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions | Discount for products according to events from the system | |
| Description | Customers evaluate the quality of the product and can comment on the product | |
| Action | Agent action | System action |
| (1) Select the product to review.  (2) Select the number of stars you want to rate.  (3) Enter a comment in the input box.  (4) Select comment button. | (5) The system says "thanks for the review". Record star and comment data into the database.  (6) Display comments and recently rated stars on the product's detail page |

#### Use case personal account management:

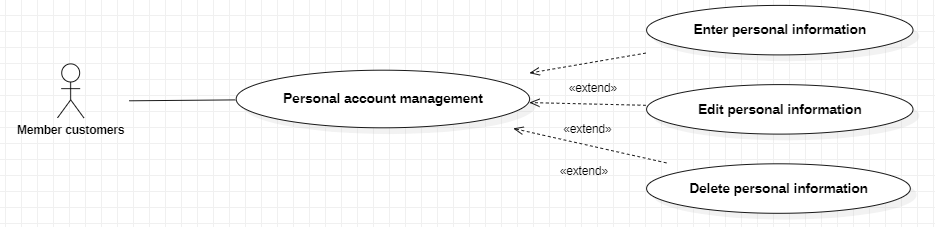


Figure 2.12 - Use case for personal account management

Table 2.11 - Specific description use case personal account management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Personal account management | |
| Agent | Member customers | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Users click on the profile item to view, edit and delete personal information | |
| Action | Agent action | System action |
| (1) Click on profile | (2) The system displays information for customers to add, edit and delete |

#### Use case order through the system:

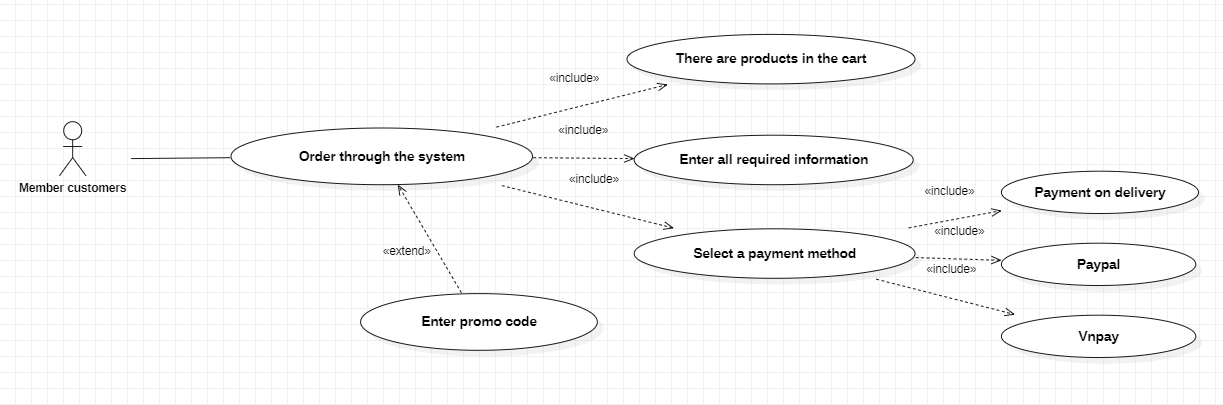


Figure 2.13 - Use case order through the system

Table 2.12 - Specific description use case order through the system

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Order through the system | |
| Agent | Member customers | |
| Pre-conditions | There are products in the cart, enter personal information according to the system's requirements, enter the promotion code (if any) | |
| The following conditions |  | |
| Description | Customers buy goods after choosing products on the system | |
| Action | Agent action | System action |
| (1) Customer selects Order. | (2) The system retrieves personal information, order information.  (3) If successful, the system will notify you of success, record it in the database, and send an order confirmation email.  (4) If it fails, the system will notify you that the order has failed. |

#### Use case user accounts management:

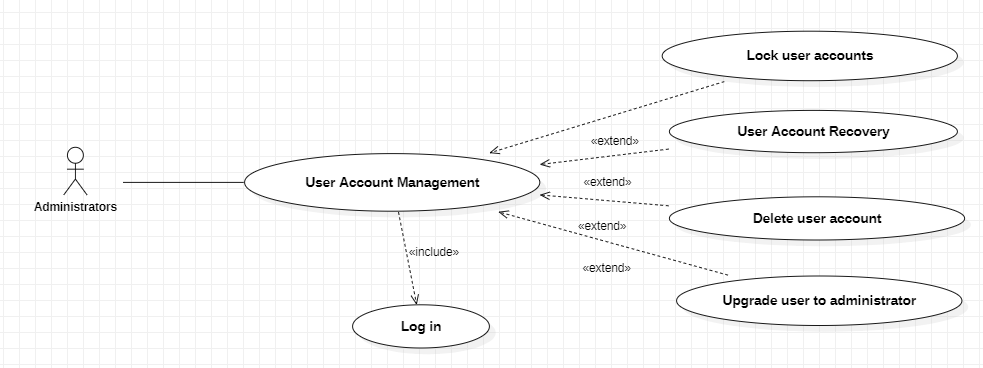


Figure 2.14 - Use case user accounts management

Table 2.13 - Specific description use case user accounts management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Management of user accounts | |
| Agent | Administrators | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Manage customer account information | |
| Action | Agent action | System action |
| (1) Select Manage customer accounts. | (2) The system displays all customer accounts. |

#### Use case product management:

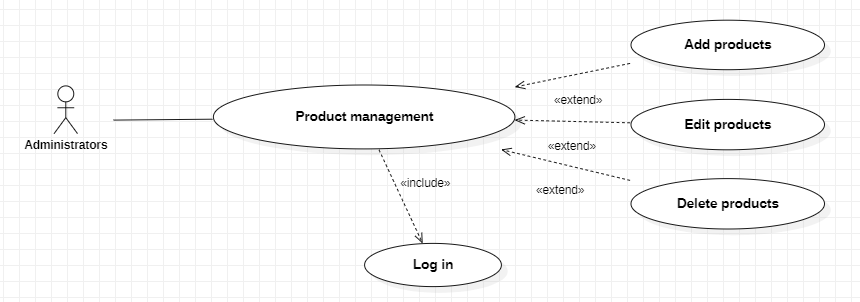


Figure 2.15 - Use case product management

Table 2.14 - Specific description use case product management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Product management | |
| Agent | Administrators | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Manage customer account information | |
| Action | Agent action | System action |
| (1) Select product management. | (2) The system displays product information, displays the functions of adding, editing and deleting products. |

#### Use case order management:

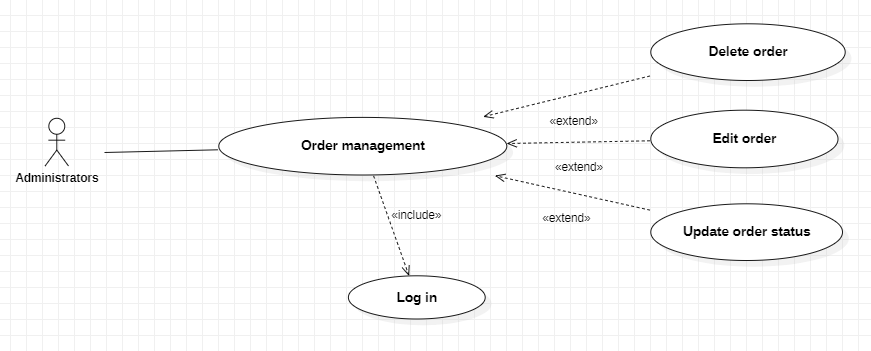


Figure 2.16 - Use case order management

Table 2.15 - Specific description use case order management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Order management | |
| Agent | Administrators | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Admin can manage all orders, and update order statuses. | |
| Action | Agent action | System action |
| (1) Select order management. | (2) The system displays orders for admin and functions to update order status. |

#### Use case product category management:

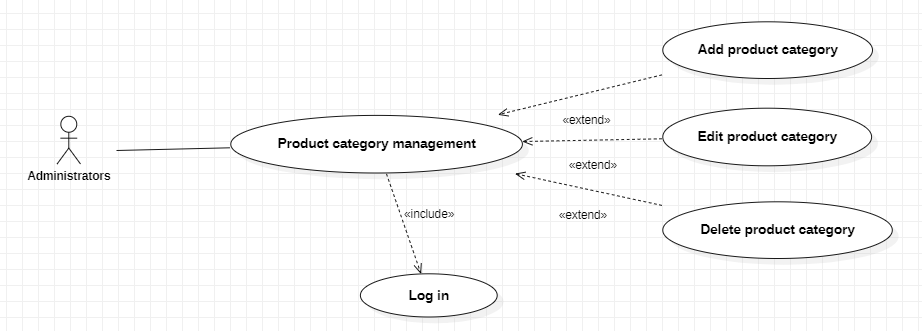


Figure 2.17 - Use case product category management

Table 2.16 - Specific description use case product category management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Product category management | |
| Agent | Administrators | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Manage adding, editing, deleting product categories | |
| Action | Agent action | System action |
| (1) Select product category management. | (2) The system displays all product categories and functions, add, edit, delete. |

#### Use case promotion management:

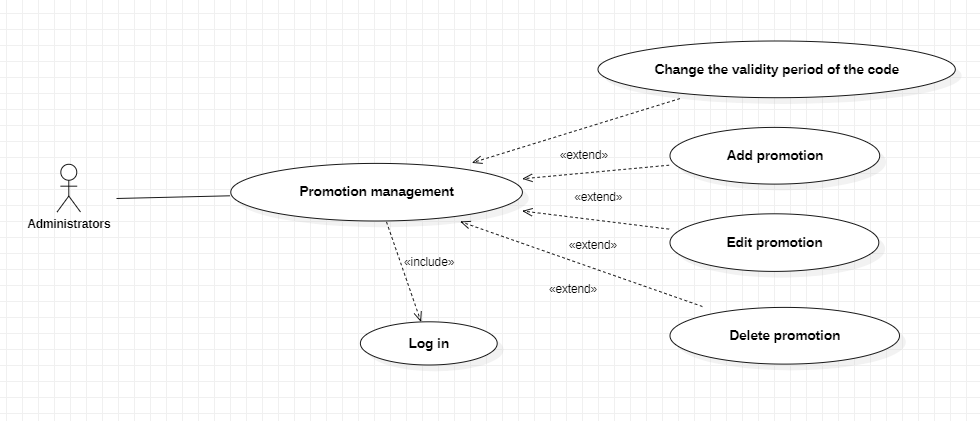


Figure 2.18 - Use case promotion management

Table 2.17 - Specific description use case promotion management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Promotion management | |
| Agent | Administrators | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Manage adding, editing, deleting promo codes. | |
| Action | Agent action | System action |
| (1) Select promotion management. | (2) The system displays promotional events and functions to add, edit, and delete promotional codes. |

#### Use case comments and star ratings management:

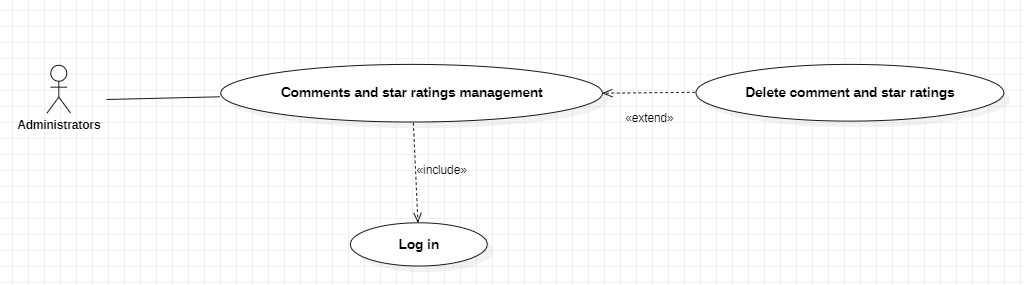


Figure 2.19 - Use case comments and star ratings management

Table 2.18 - Specific description use case comment and ratings management

|  |  |  |
| --- | --- | --- |
| **Atributive** | **Description** | |
| Function name | Comments and star ratings management | |
| Agent | Administrators | |
| Pre-conditions | Successfully logged into the system | |
| The following conditions |  | |
| Description | Manage deleting comments and star ratings. | |
| Action | Agent action | System action |
| (1) Select comments and star ratings management. | (2) The system displays comment and functions to delete comment and star ratings. |

## Class diagram:

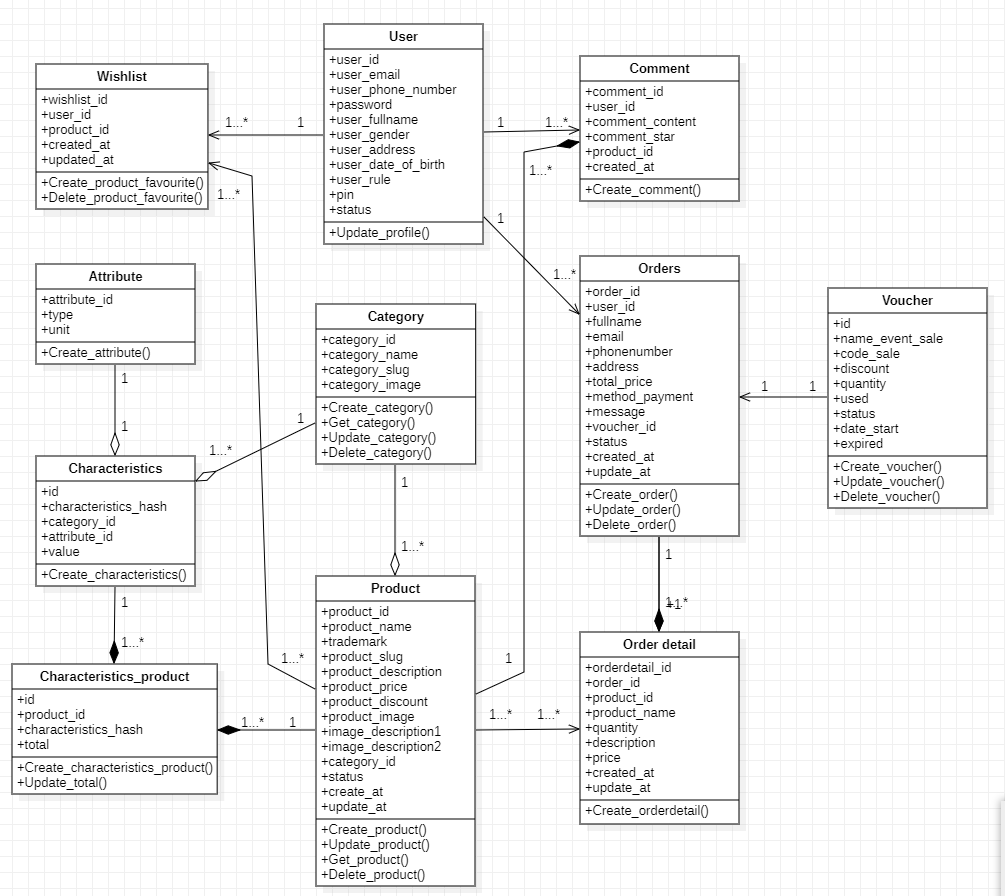


Figure 2.20 - Class diagram

## Activity diagram:

### Register function activity diagram:

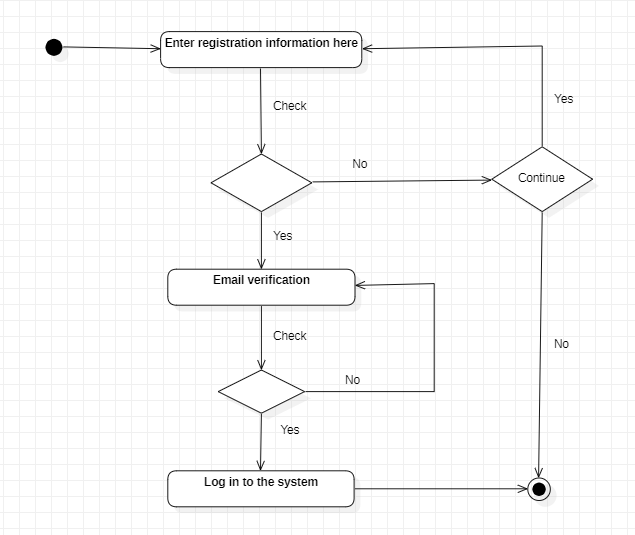


Figure 2.21 - Register function activity diagram

### Login function activity diagram:

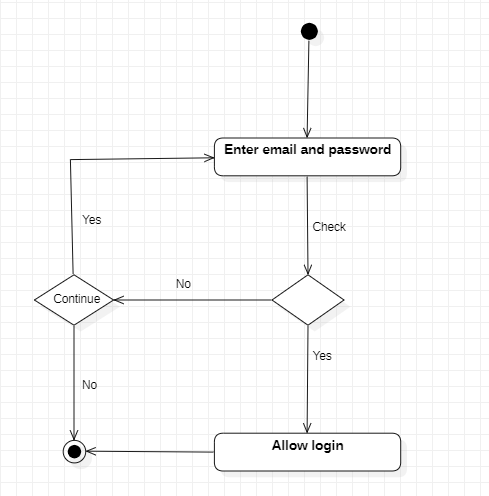


Figure 2.22 - Login function activity diagram

### Order function activity diagram:

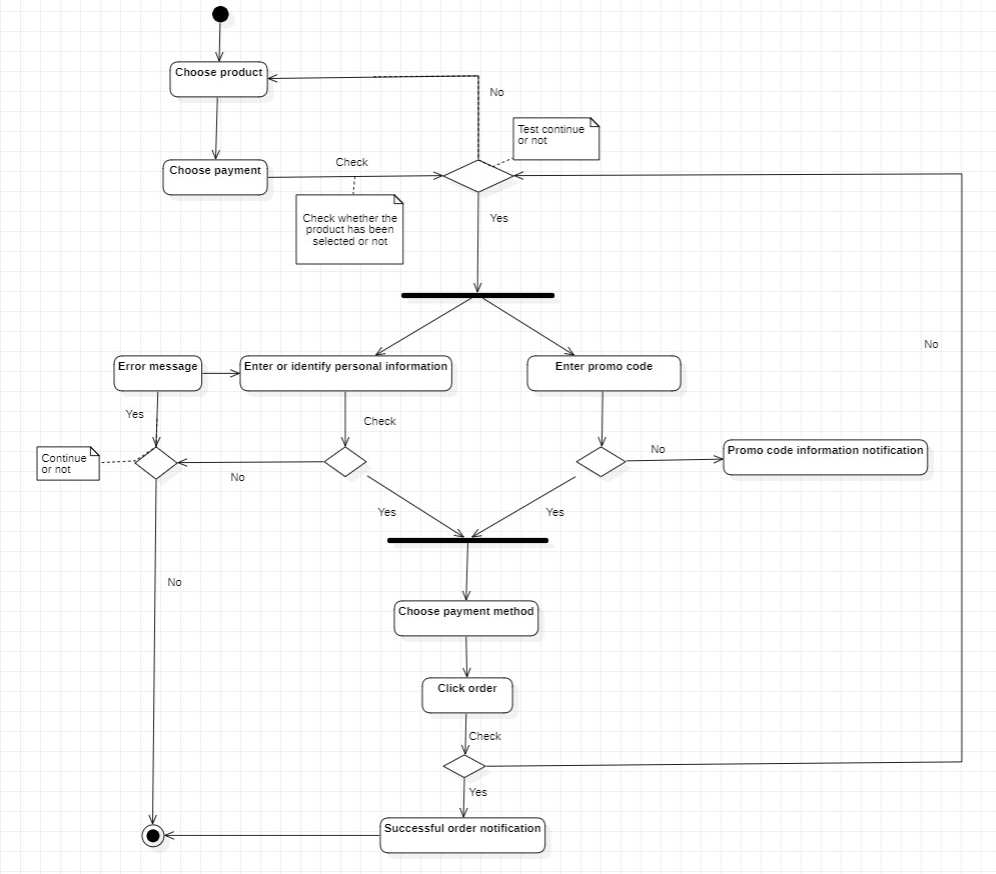


Figure 2.23 - Order function activity diagram

## Sequence diagram:

### Register function sequence diagram:

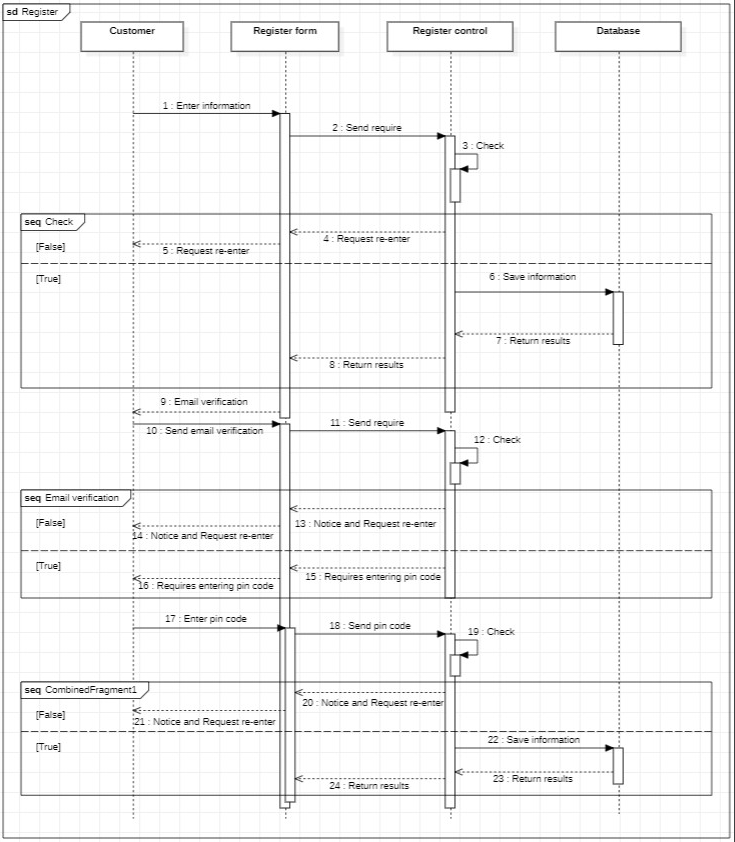


Figure 2.24 - Register function sequence diagram

Users fill in registration information in the registration form on the registration interface. Then this form will check the information about the entry rule according to the form. If it is wrong, it will send a message through the interface for the user to re-enter. If true, the server will send information to the Model to perform data entry, then return the results and switch to the email validation page to conduct the validation.

### Login function sequence diagram:

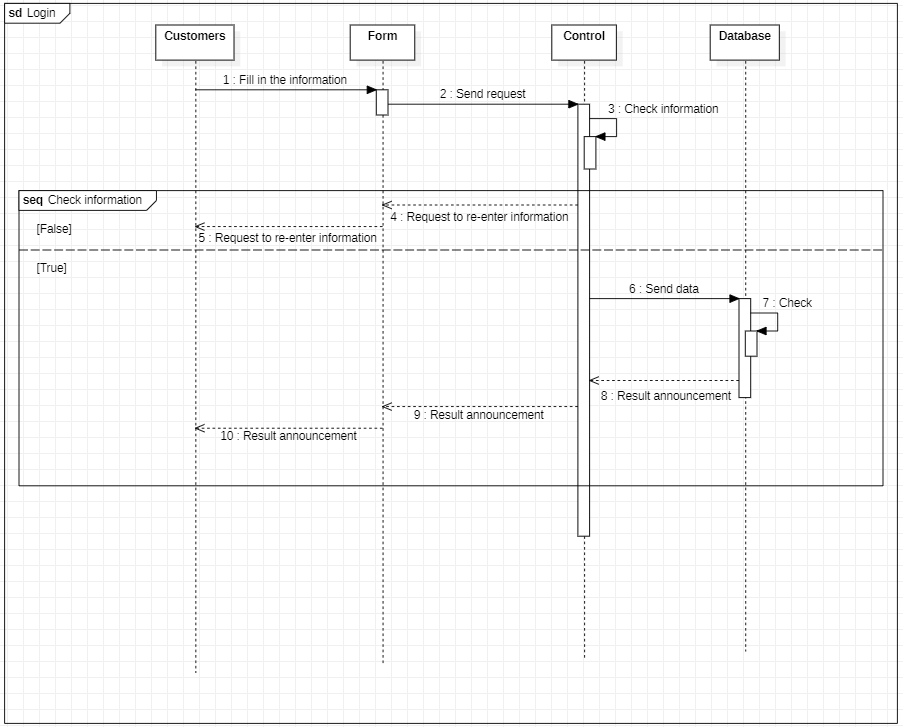


Figure 2.25 - Login function sequence diagram

Customers will fill in information on the login interface. The form will perform the check. If the information is in the wrong form of the username or the number of password characters, the server will return an error message about the interface for the agent to re-enter the information. If correct, the form is submitted to the server, the server will send a request to the Model to perform a database check and then return the results.

### View product details function sequence diagram:

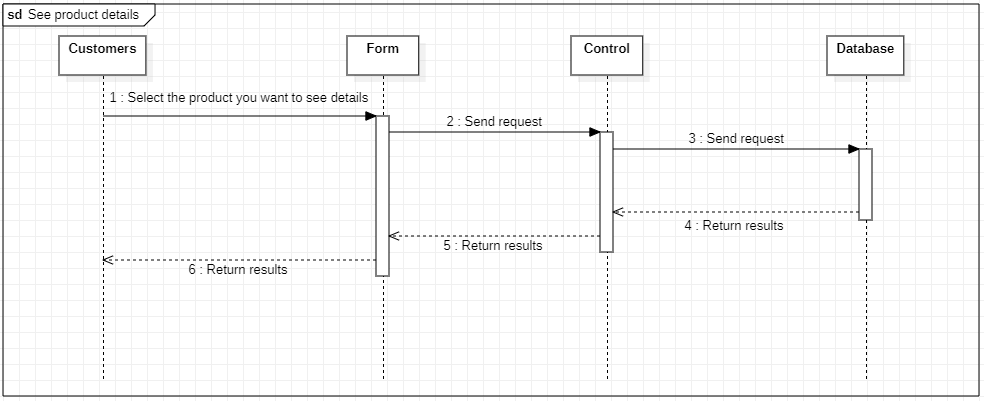


Figure 2.26 - View product details function sequence diagram

Users select products on the homepage interface or the search interface. The interface will then display a view request to the Controller (server). The processing server sends a request to the Model to get the necessary information from the database to return to the Controller. The controller will return the product details API and send the response back to the client to display to the user.

### Order function sequence diagram:

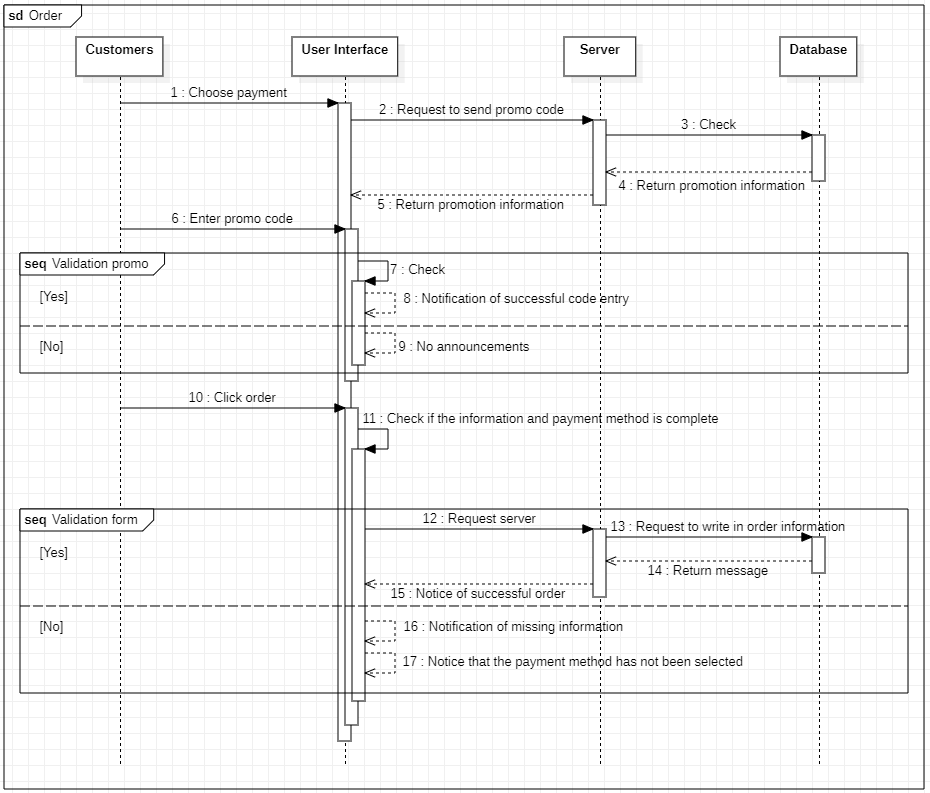


Figure 2.27 - Order function sequence diagram

### Add favorite products function sequence diagram:

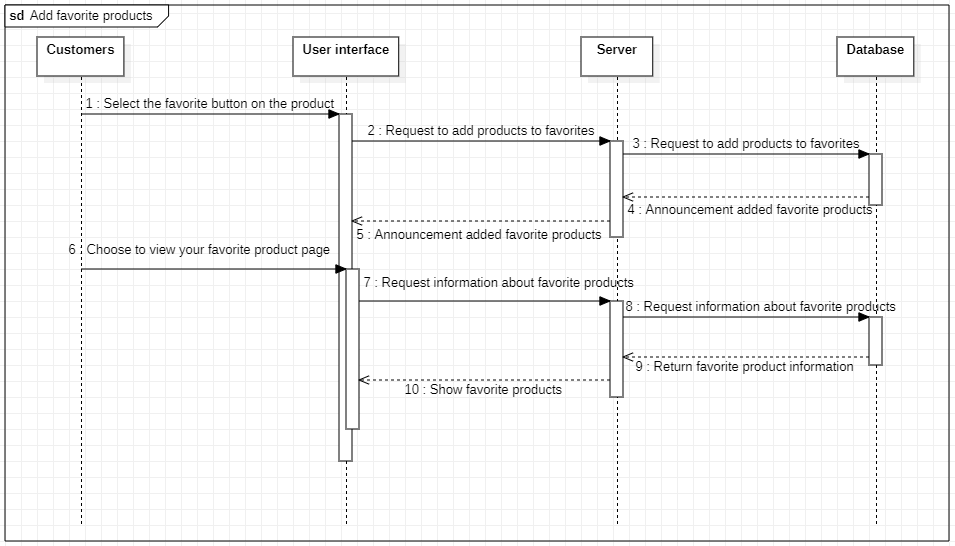


Figure 2.28 - Add favorite products function sequence diagram

## Summary

*In this chapter, we have deeply analyzed the working mechanism of the Collaborative filtering recommender system. Based on pictures and examples, we will better understand this recommendation system.*

*Analysis of general use case charts, use case specifications of each function in the system of two main actors, customers and admin.*

*Analyze class diagrams to get a better understanding of the objects and properties contained within that object.*

*Analyze the activity graph to understand the execution steps, actions, decision nodes and branching conditions to control the execution flow of the system.*

*Sequence diagram analysis to determine the sequence of events of a certain group of objects. It describes in detail the messages sent and received between objects and also focuses on the timing of sending and receiving those messages.*

# IMPLEMENTATION AND BUILDING SYSTEM

## Build website interface with React Js:

Install the required libraries for ReactJS:

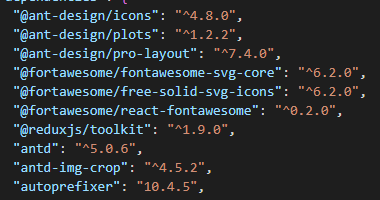


Figure 3.1 - Install the required libraries for ReactJS

## Use collaborative filtering suggestions to recommend products:

Install the required libraries for Python

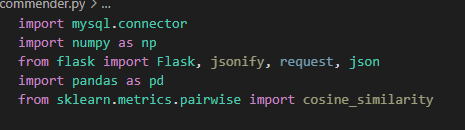


Figure 3.2 - Install the required libraries for Python

Use Flask library to write api that returns suggested product data



Figure 3.3 - Use the library Flask

Use the library mysql.connector for database connection

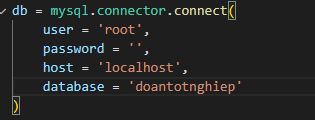


Figure 3.4 - Configure database connection

Use query statements to get or post data

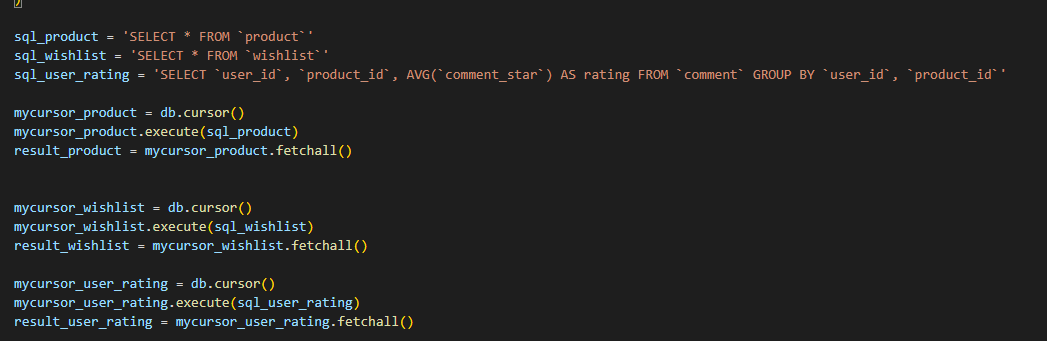


Figure 3.5 - Database query

Configure routes



Figure 3.6 - Configure routes

Before calculating the Pearson correlation coefficient, we need to create a matrix and process the NaN values



Figure 3.7 - Create a matrix and process the NaN values

Calculate the correlation coefficient using corr(method='pearson') and sort the coefficients in descending order

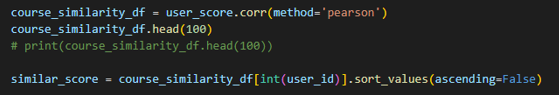


Figure 3.8 - Calculate the correlation coefficient using corr(method='pearson')

Use the Pearson correlation coefficient to compare the correlation between users using the formula:

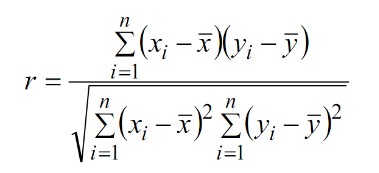


Figure 3.9 - The formula for calculating the correlation coefficient Pearson

Use Json library to return data in Json format



Figure 3.10 - Use Json library to return data in Json format

## Building a server system using Node Js:

Configure database connection using mysql library

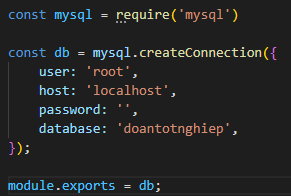
****

Figure 3.11 - Configure database connection

Install the necessary libraries for node JS:

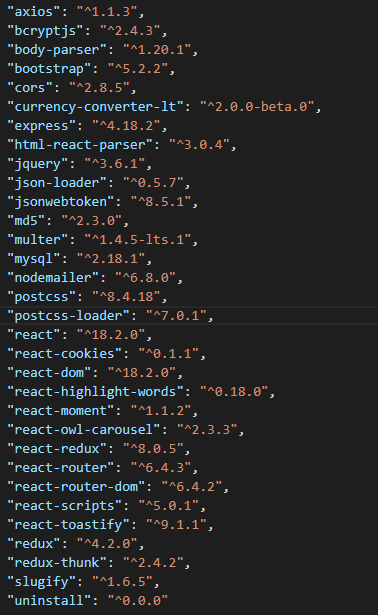


Figure 3.12 - Install the necessary libraries for node JS

## Build a database:

In this topic, I choose the database management system MySQL to work. The database to work with the system is called e-commerce. Includes 11 tables



Figure 3.13 - Tables in the database

## E-commerce system interface:

### Customer interface:

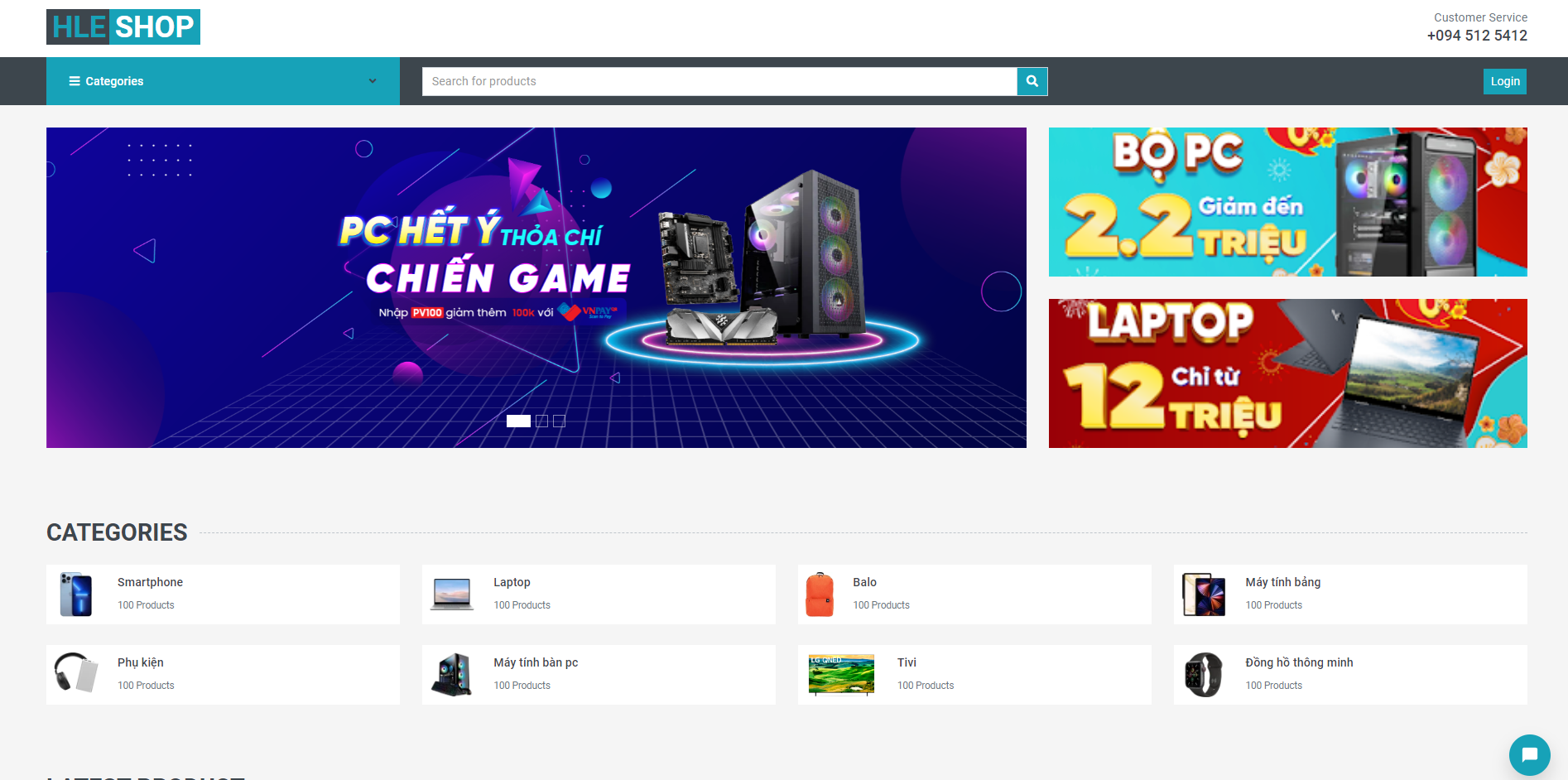


Figure 3.14 - Home page interface

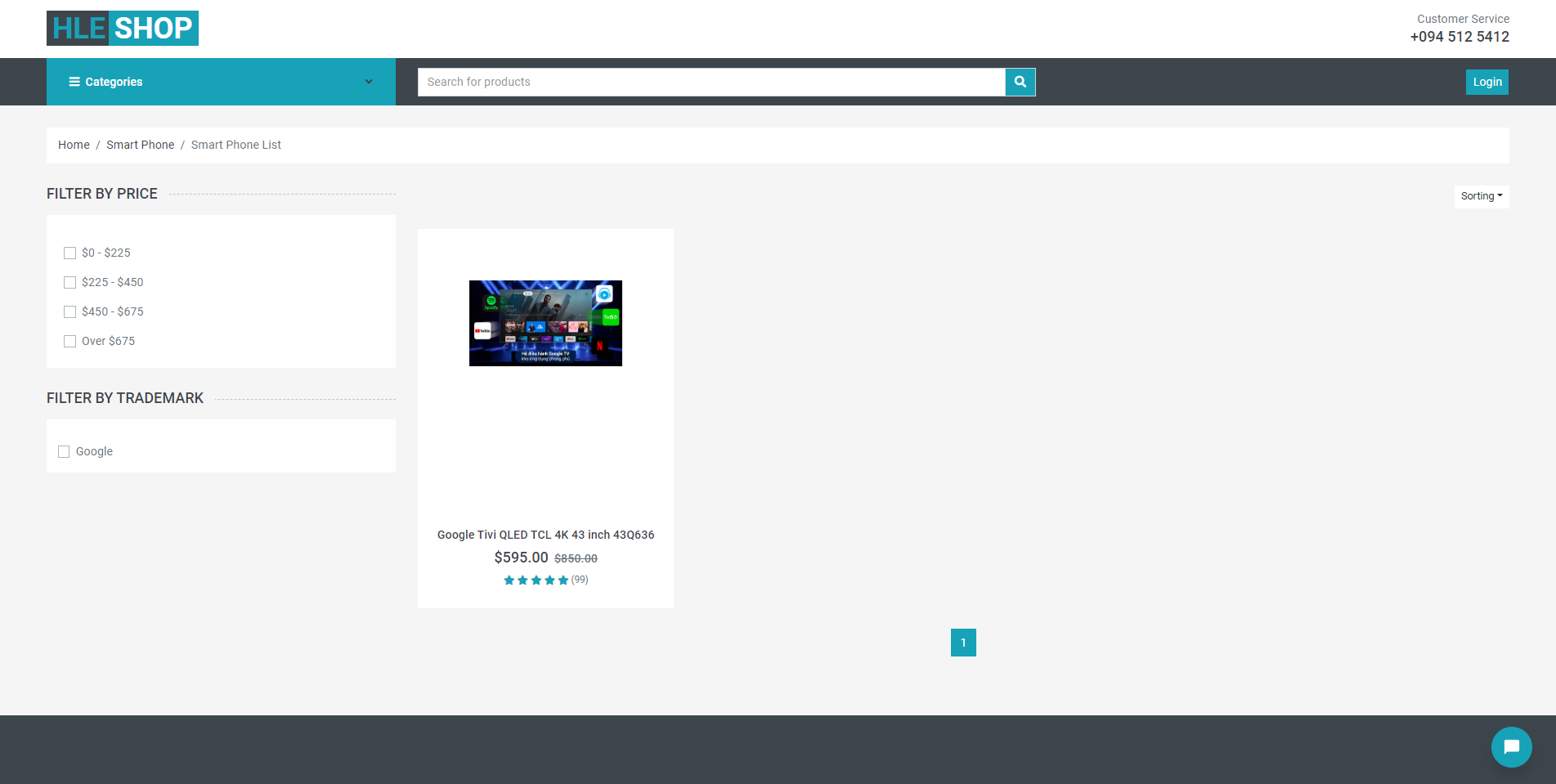


Figure 3.15 - Product type interface

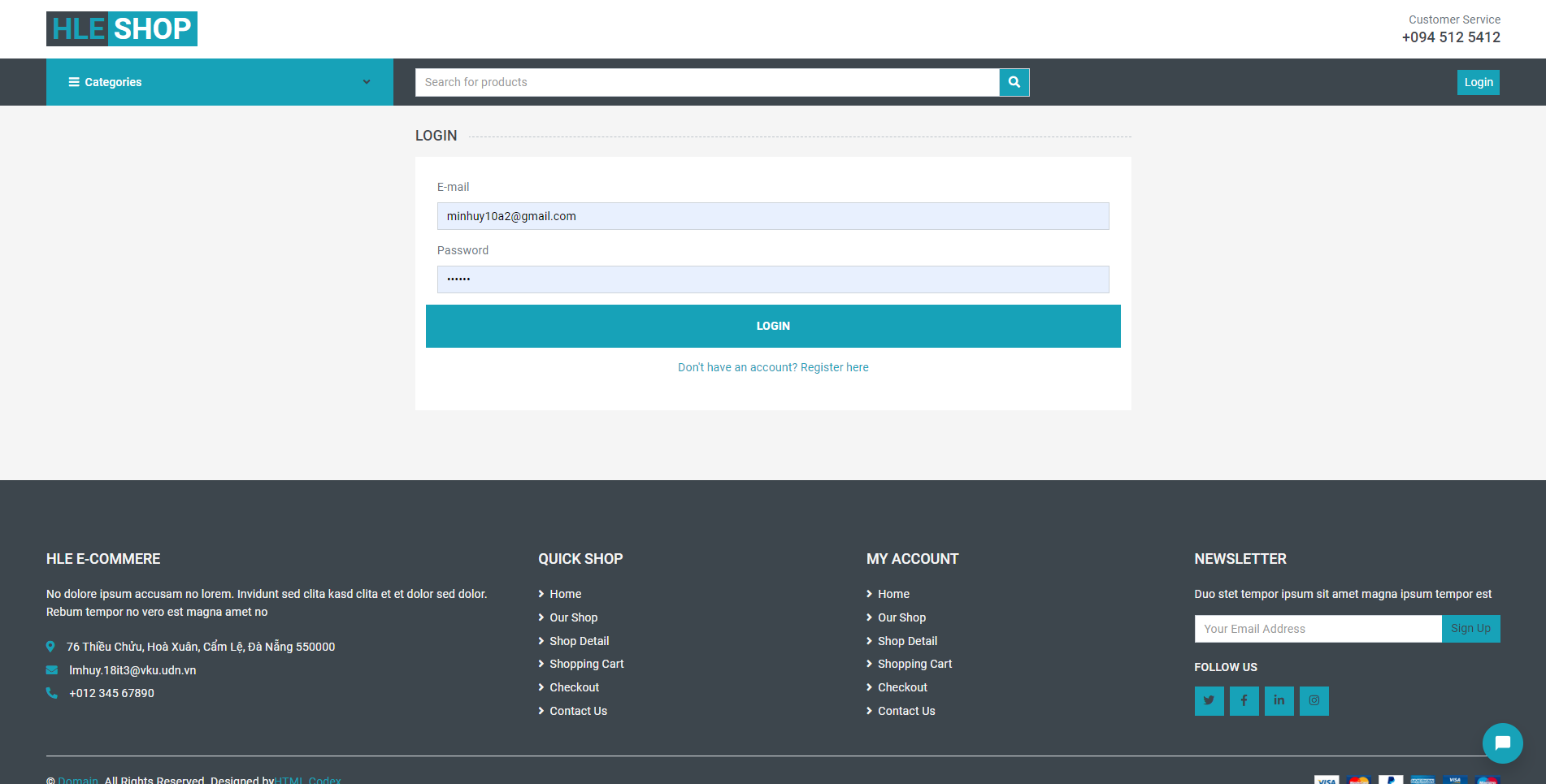


Figure 3.16 - Login page interface

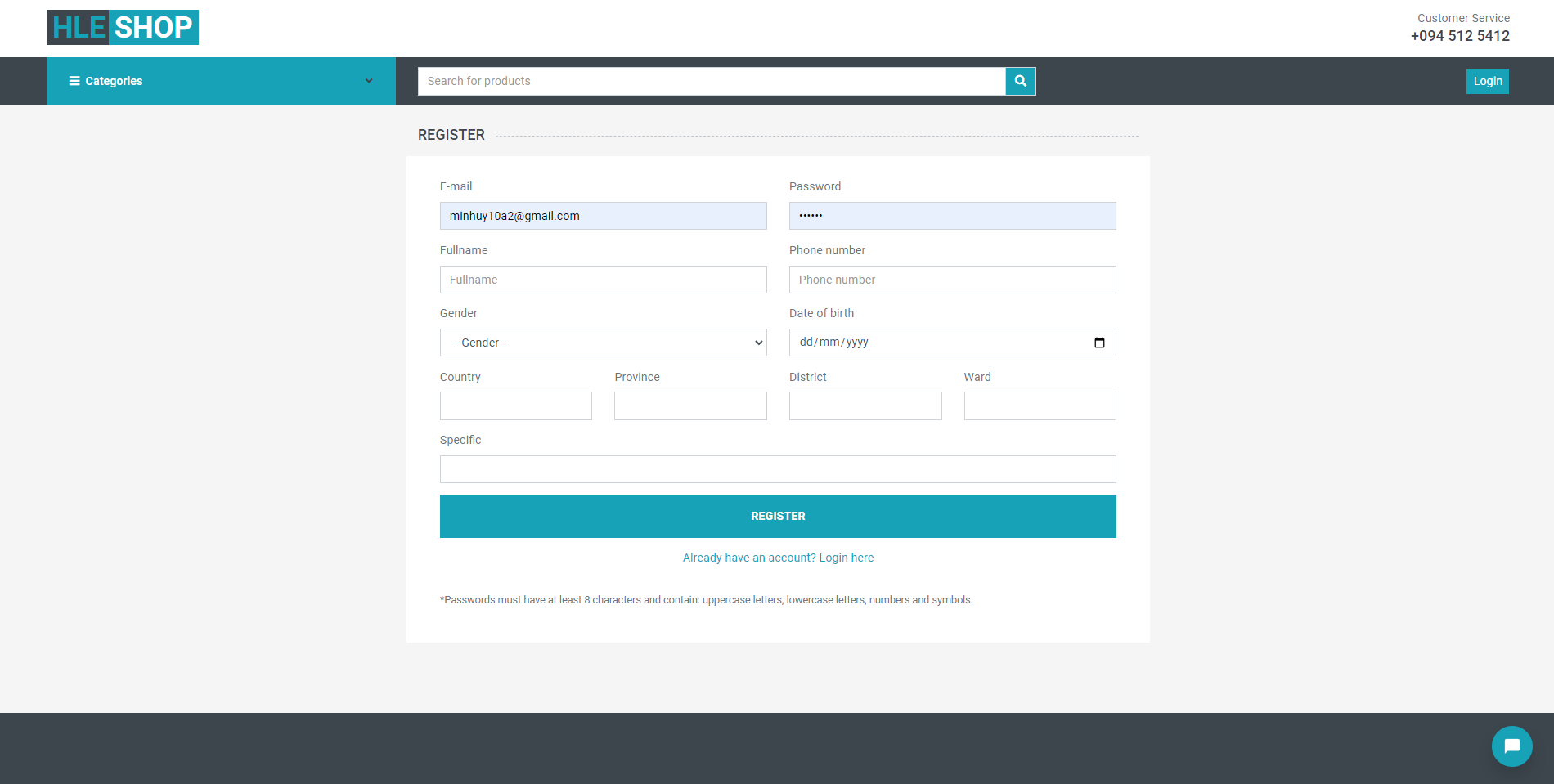


Figure 3.17 - Register page interface

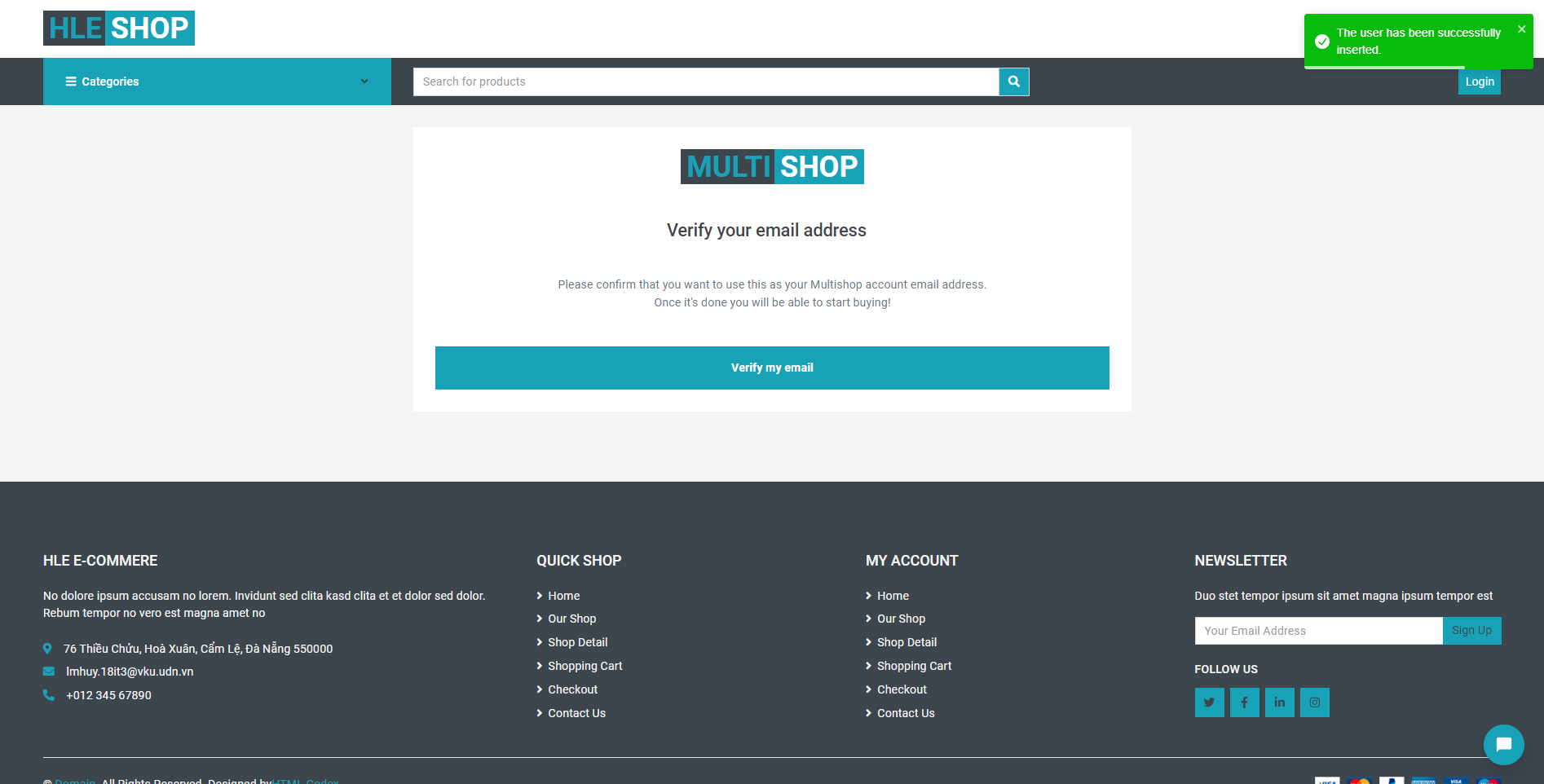


Figure 3.18 - Email verification page interface

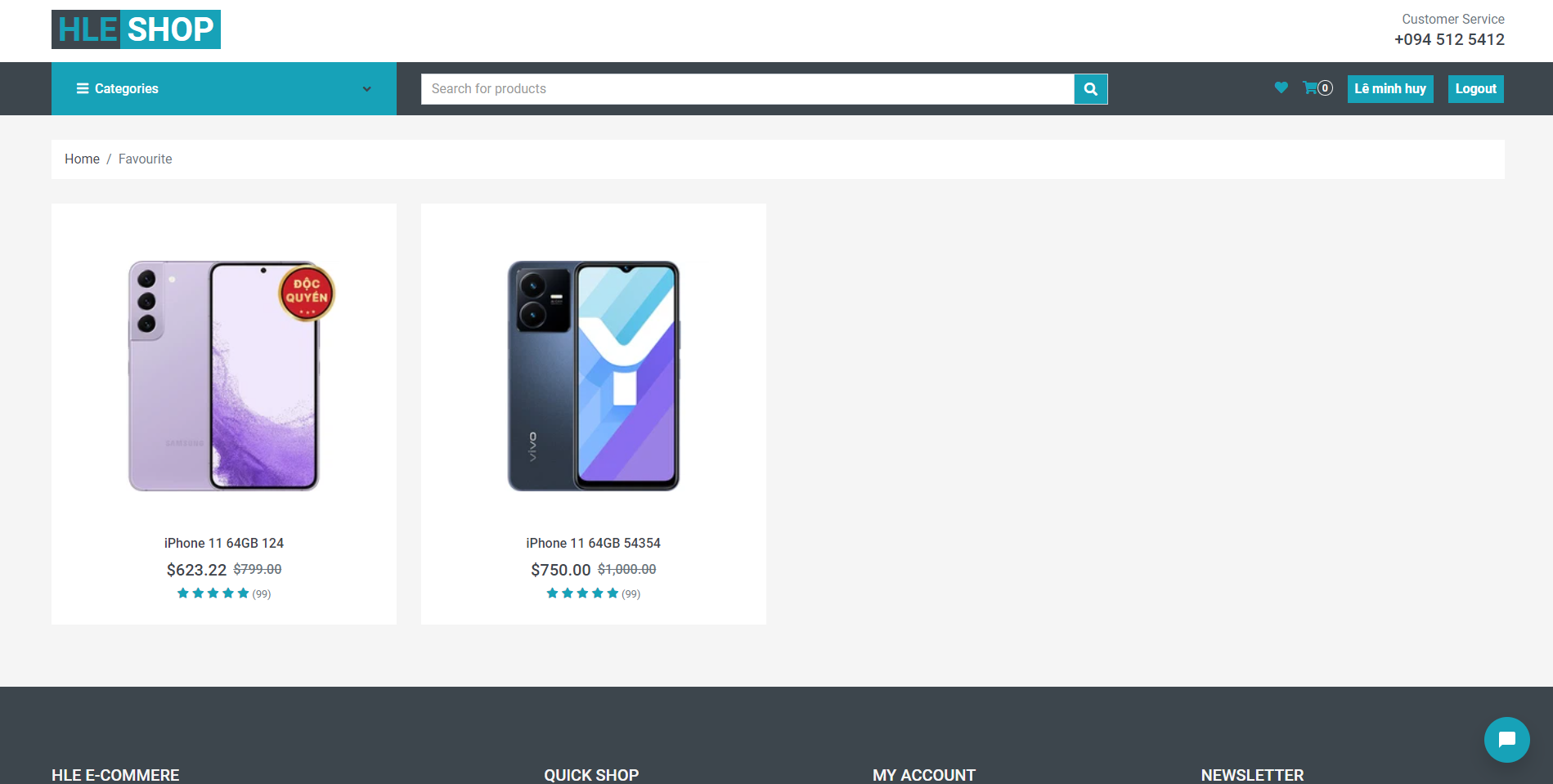


Figure 3.19 - Favourite page interface



Figure 3.20 - Search page interface

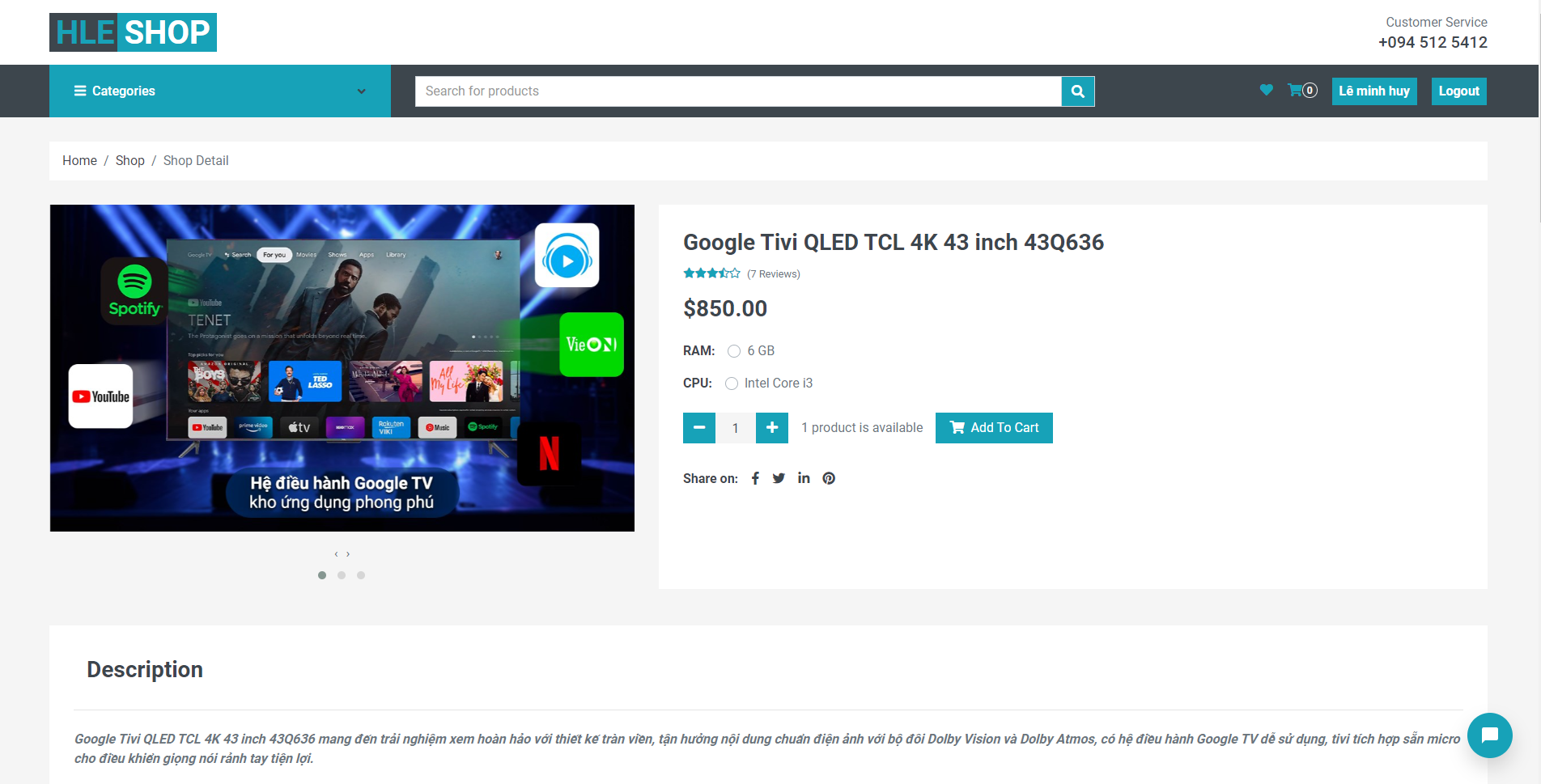


Figure 3.21 - Product detail page interface

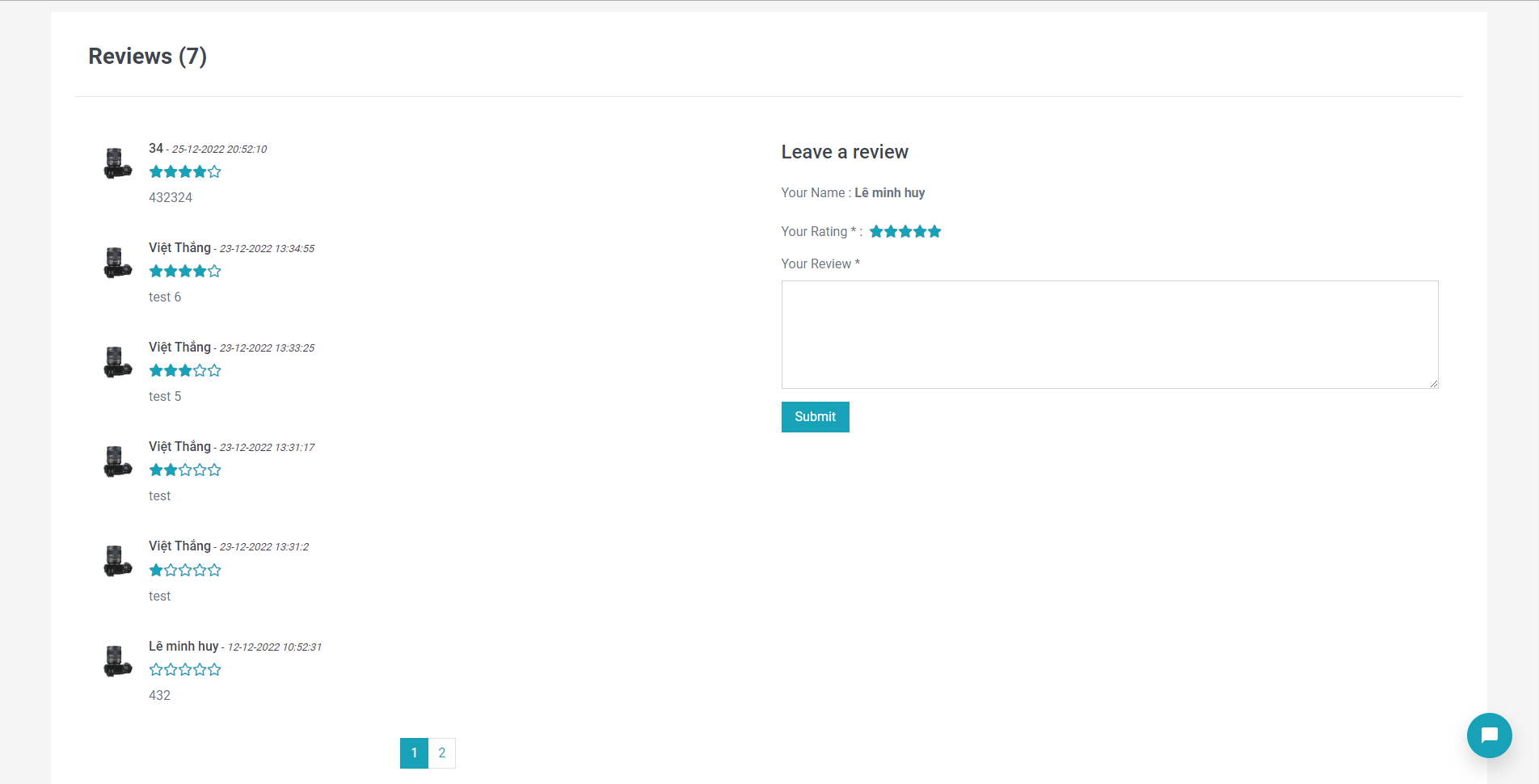


Figure 3.22 - Comment interface

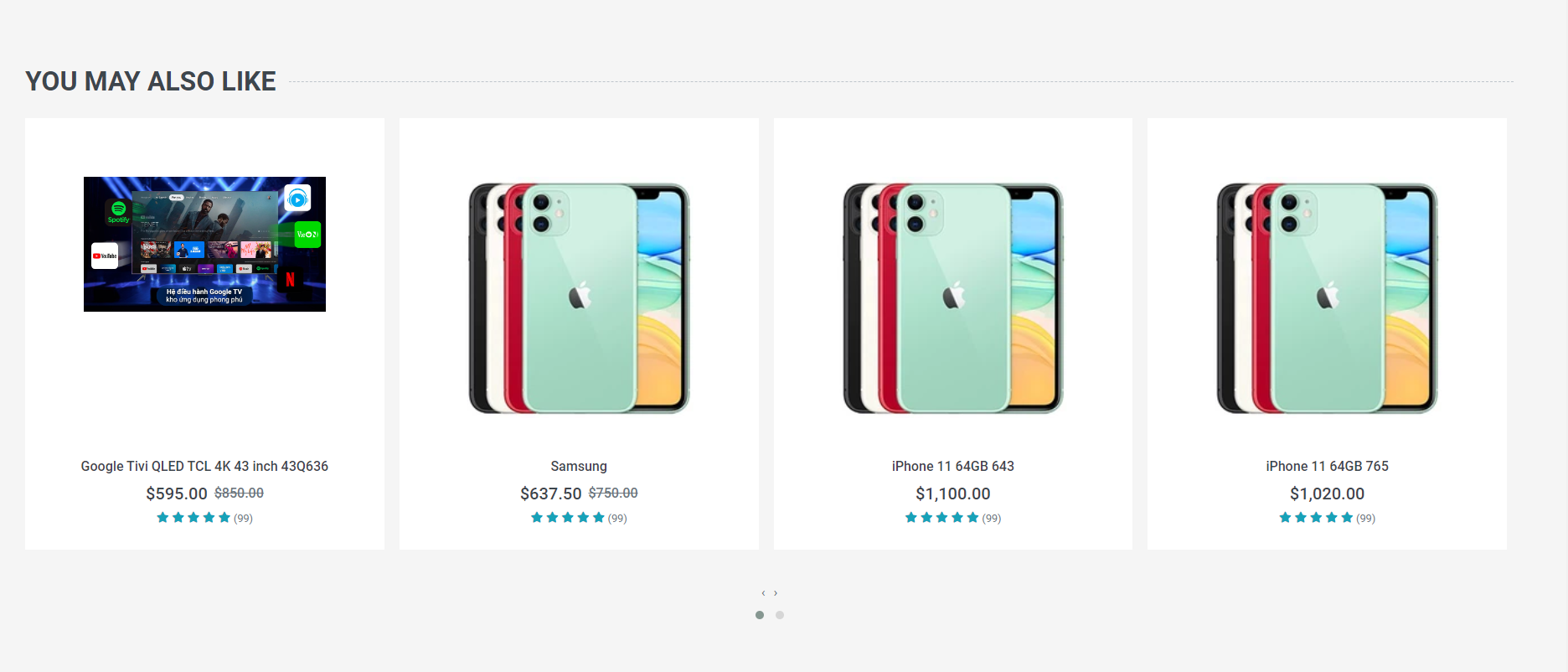


Figure 3.23 - Product suggestions interface

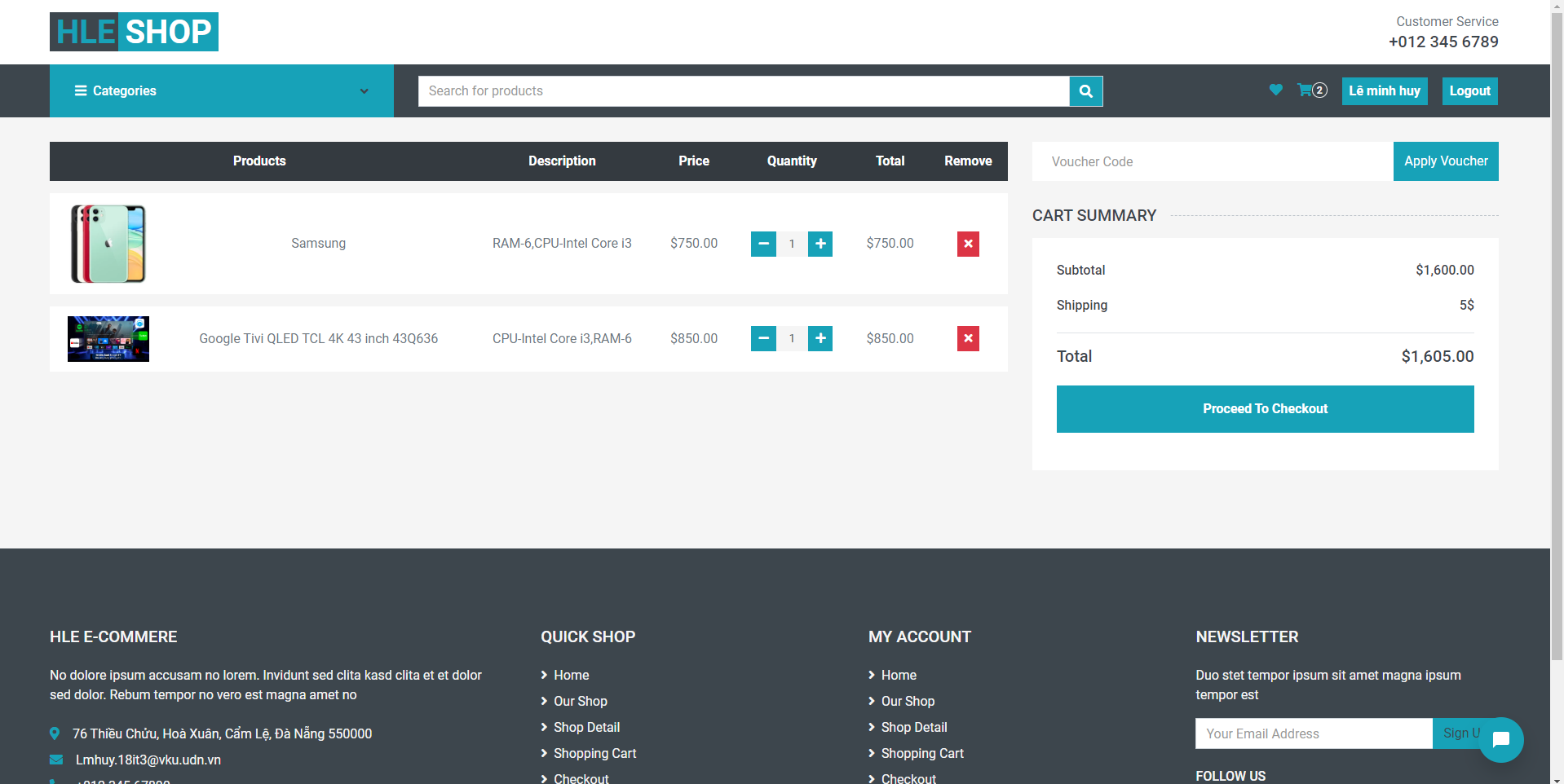


Figure 3.24 - Shopping cart interface

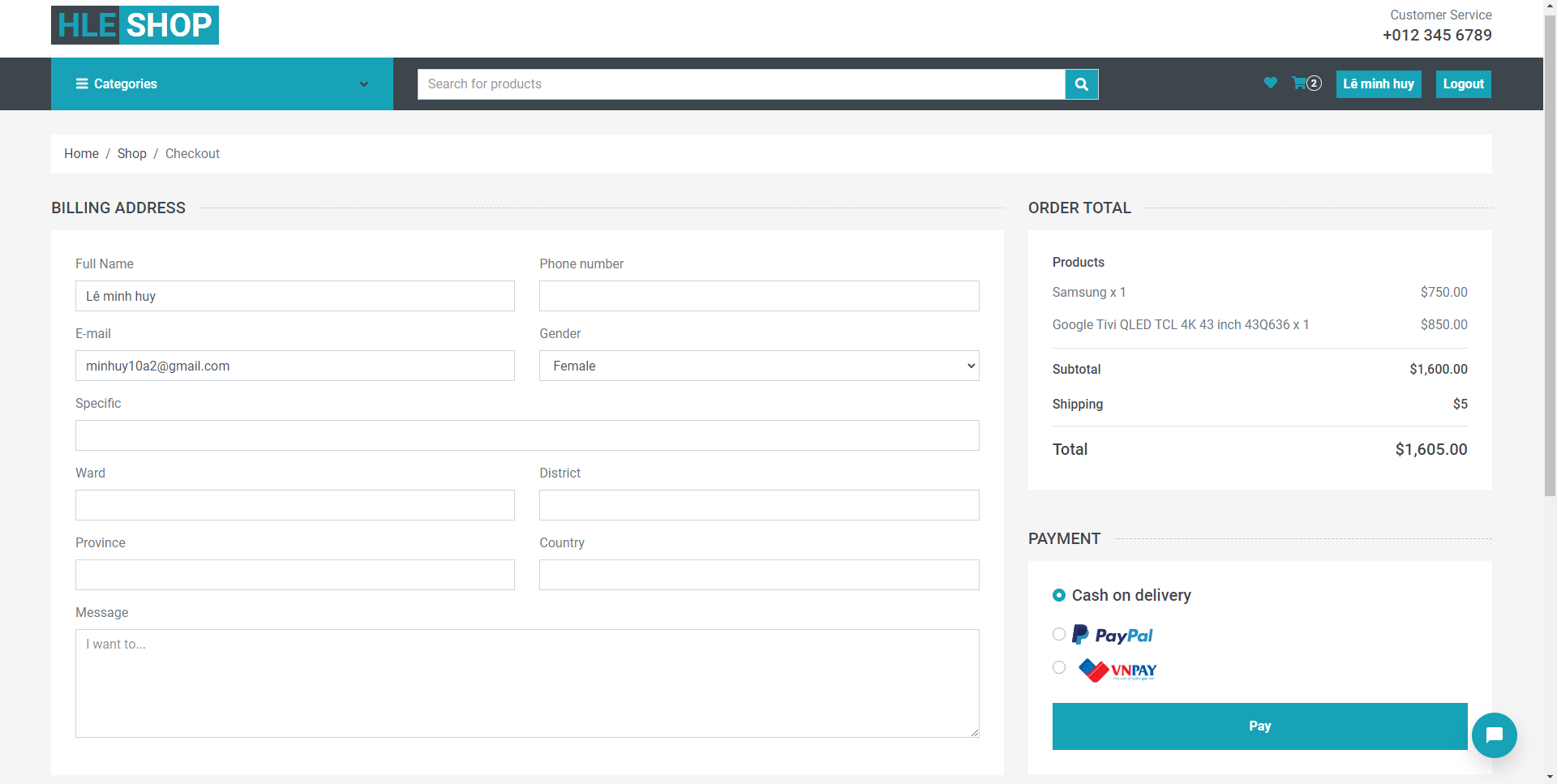


Figure 3.25 - Payment page interface

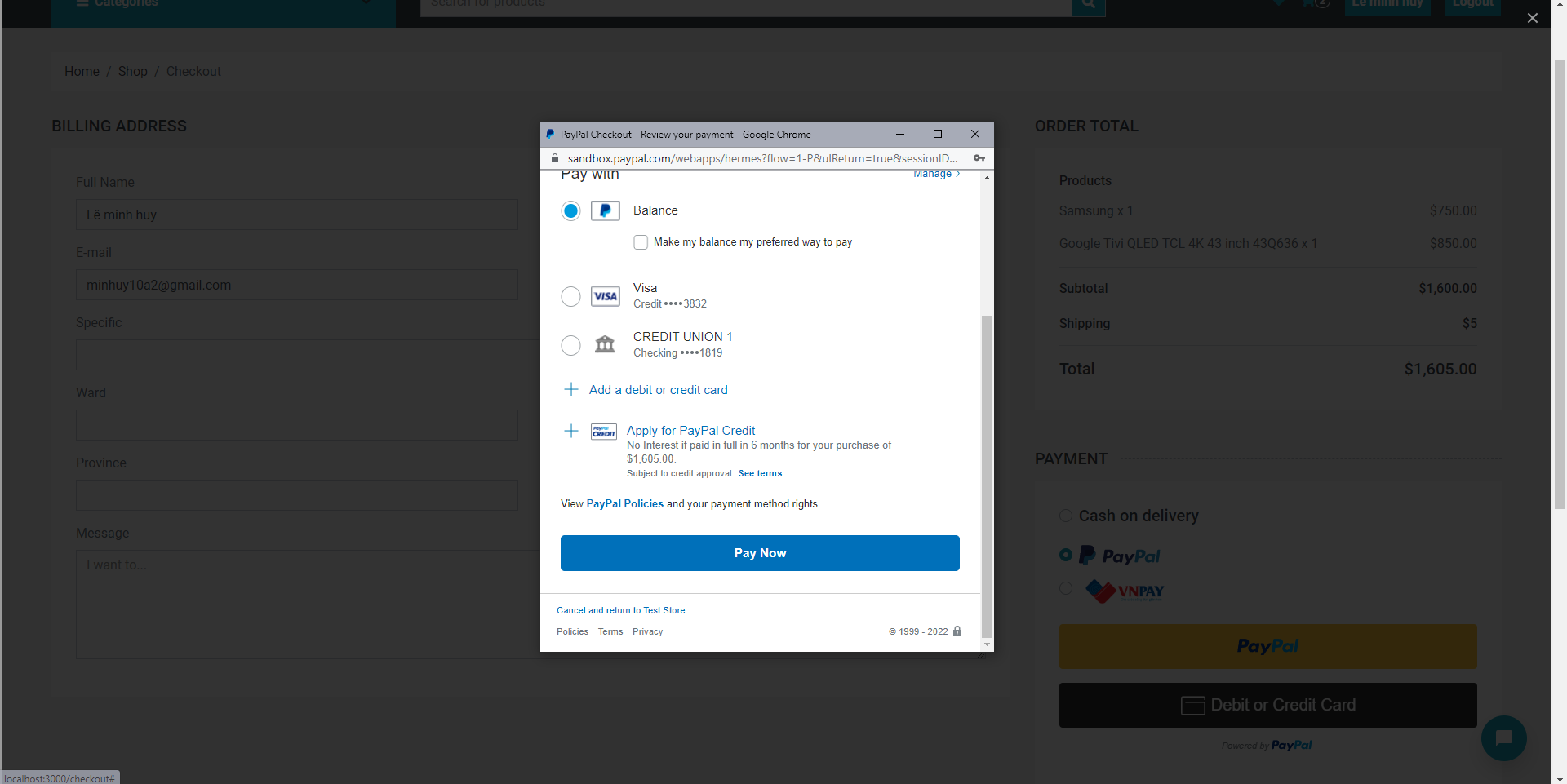


Figure 3.26 - Paypal payment interface

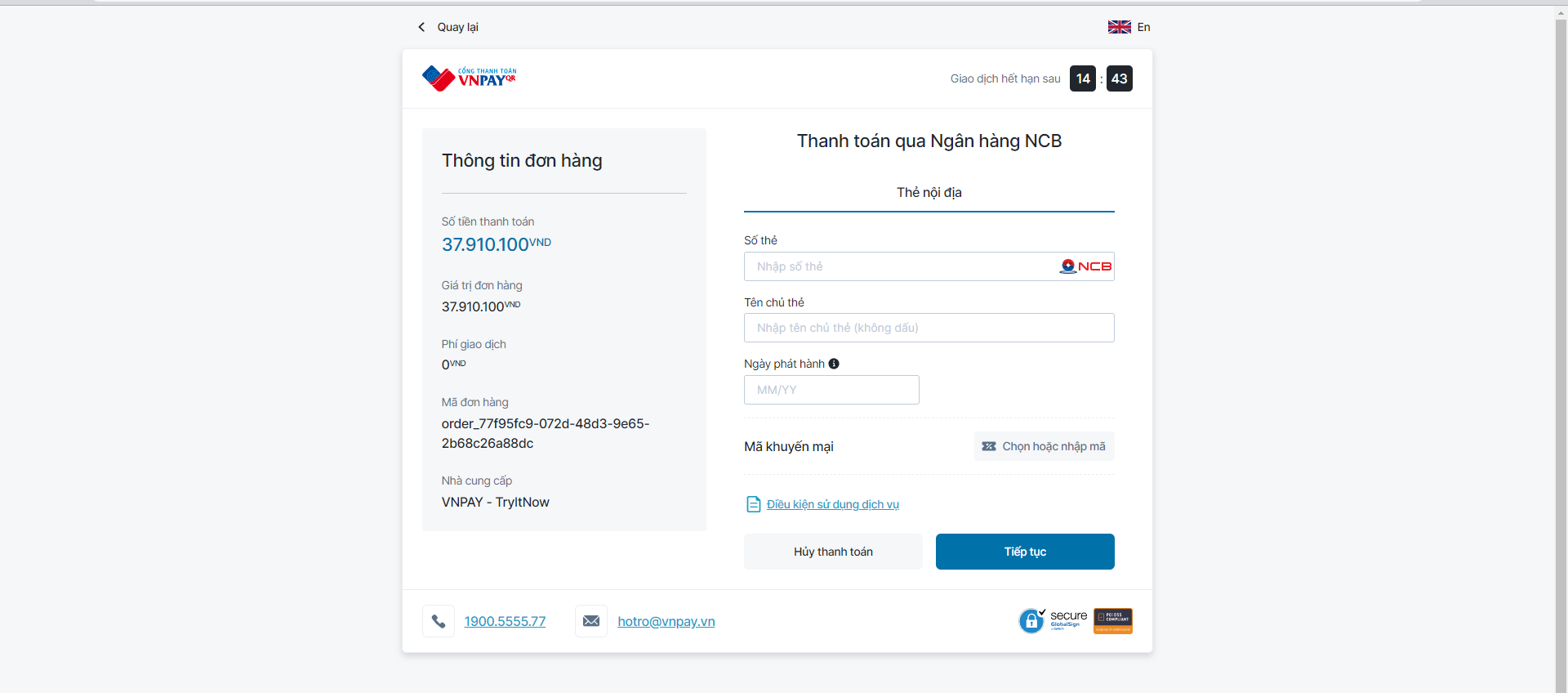


Figure 3.27 - Vnpay payment interface

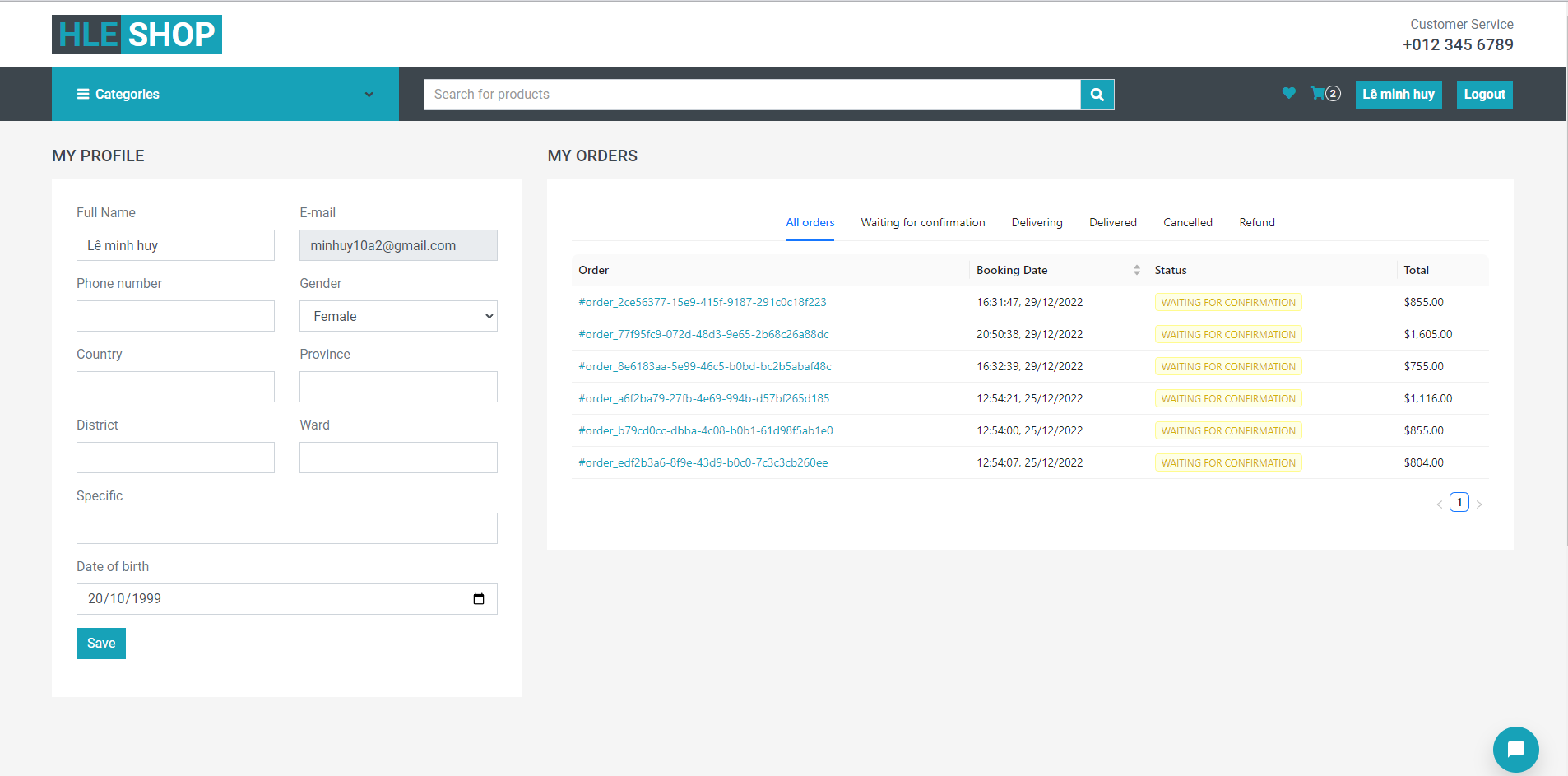


Figure 3.28 - Profile page interface

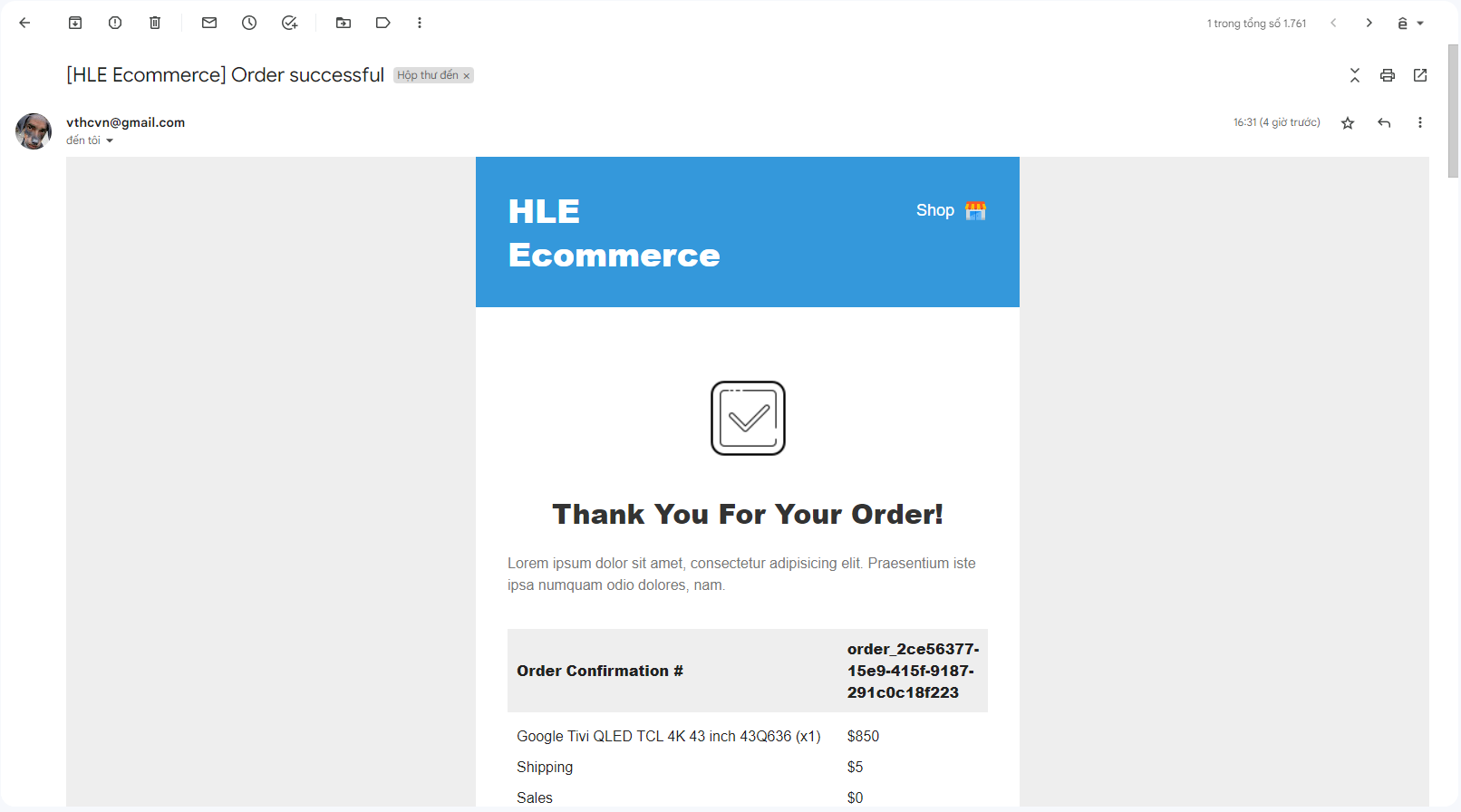


Figure 3.29 - Orders invoice interface

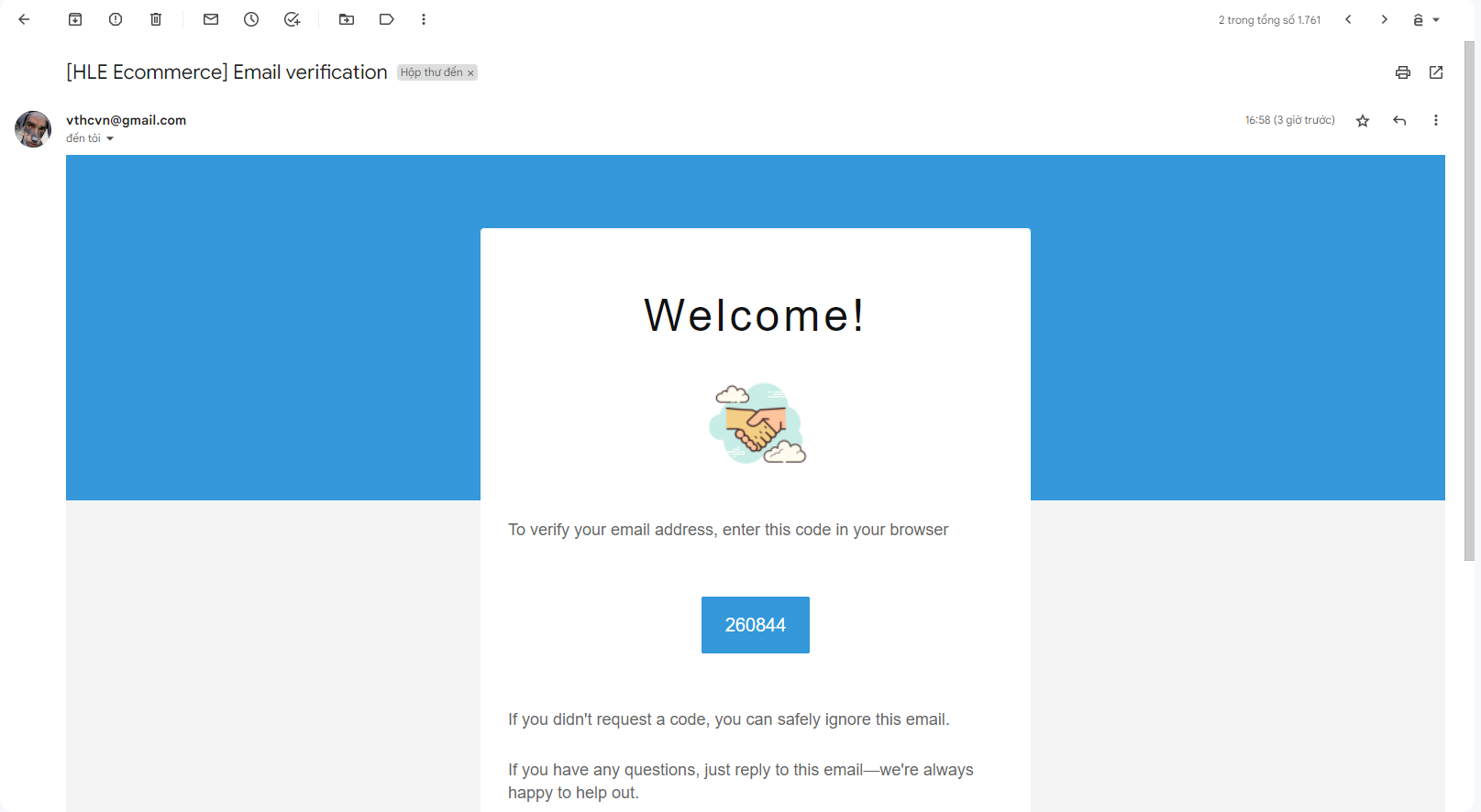


Figure 3.30 - Email verification interface

### Admin interface:

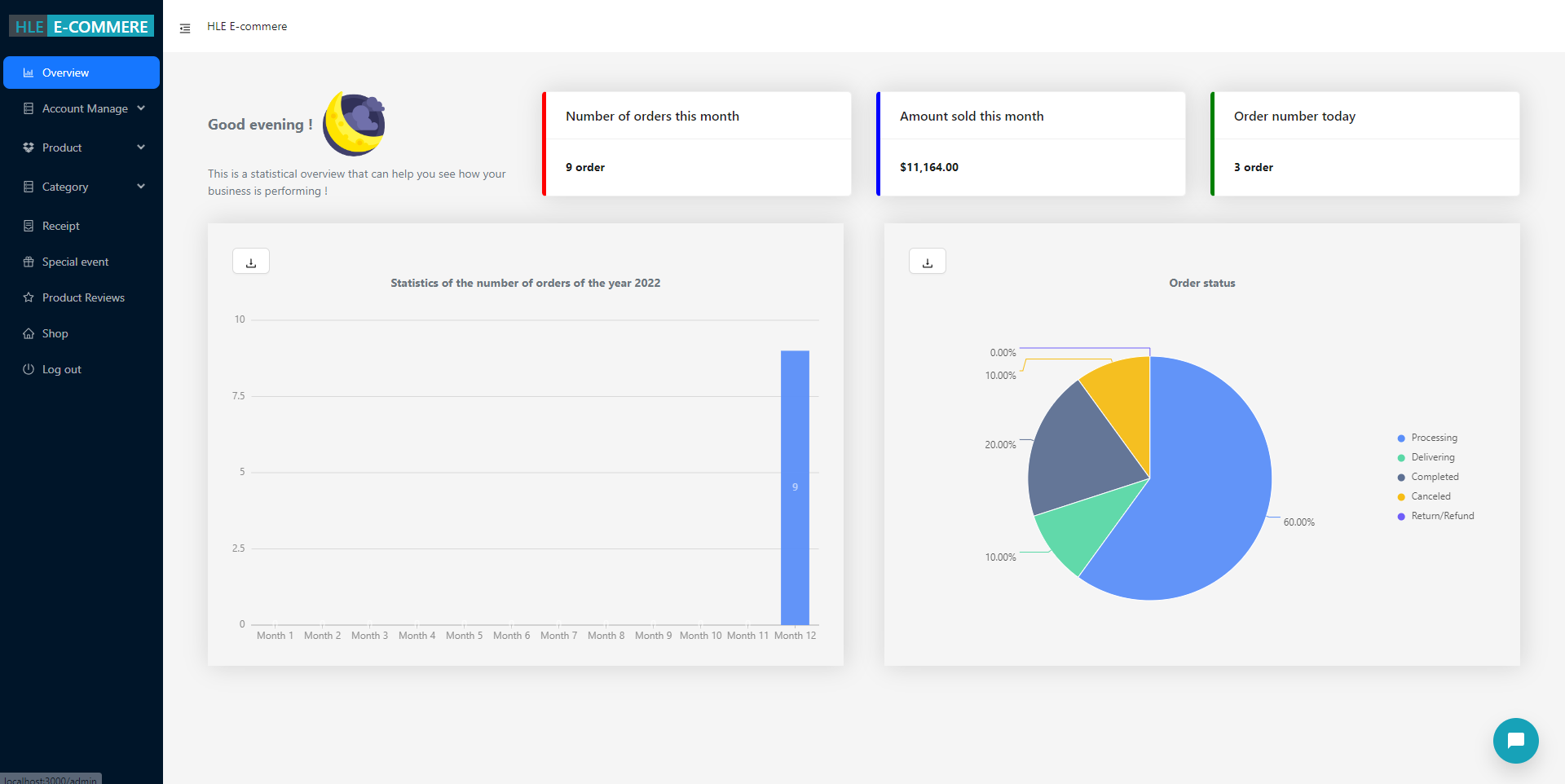


Figure 3.31 - Overview interface

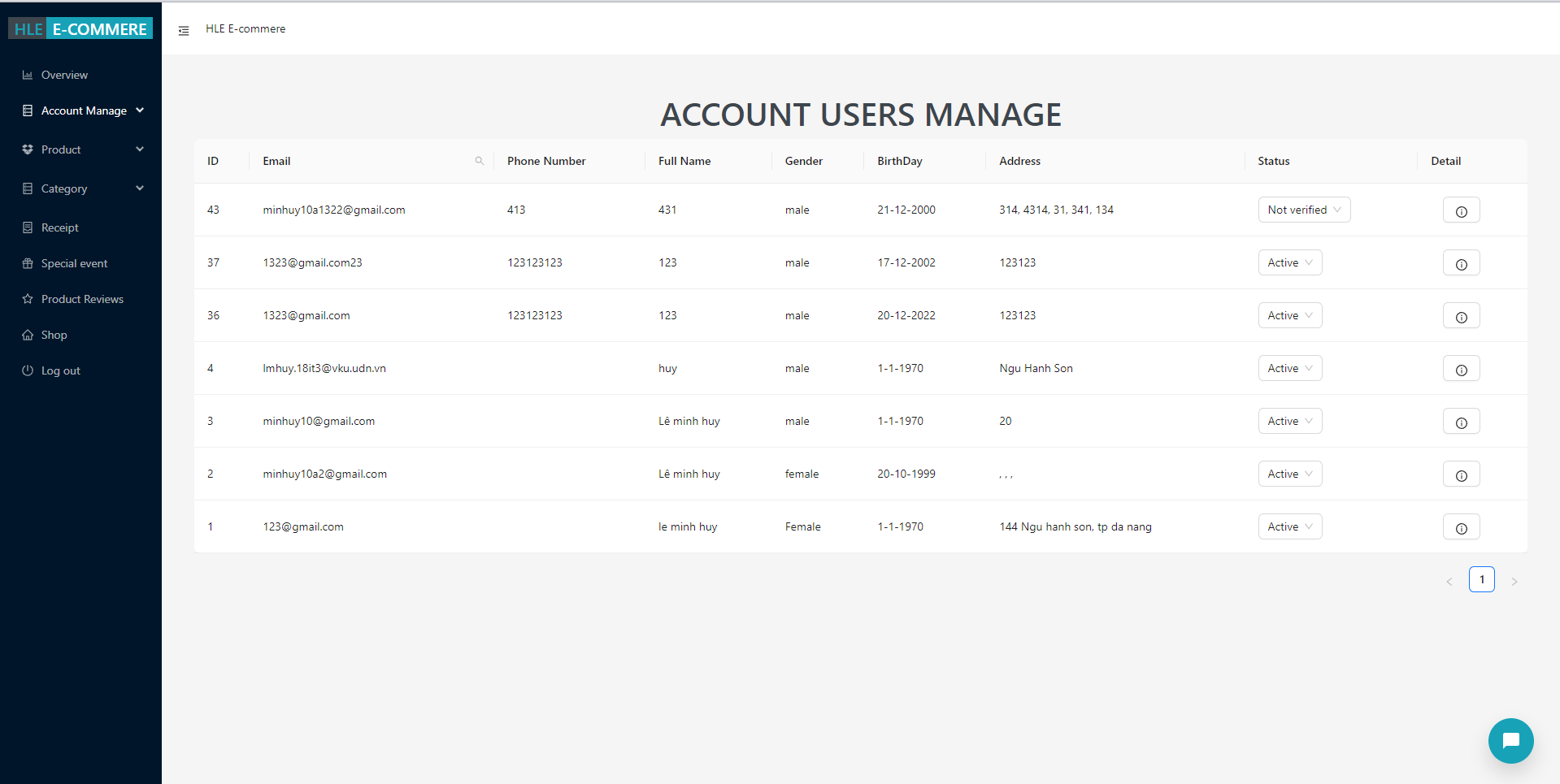


Figure 3.32 - Account users manage interface

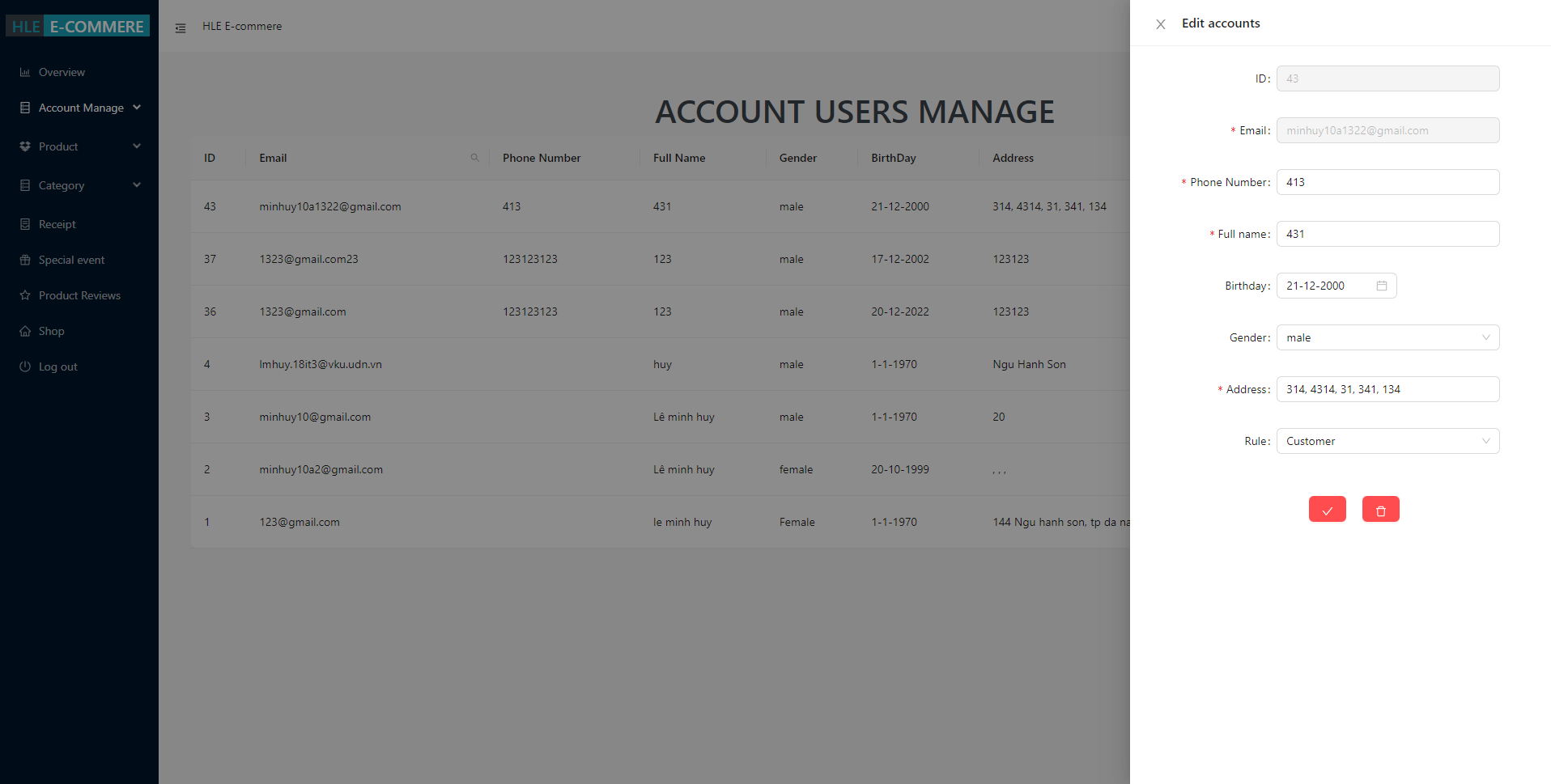


Figure 3.33 - Edit user information interface



Figure 3.34 - Account admin manage interface

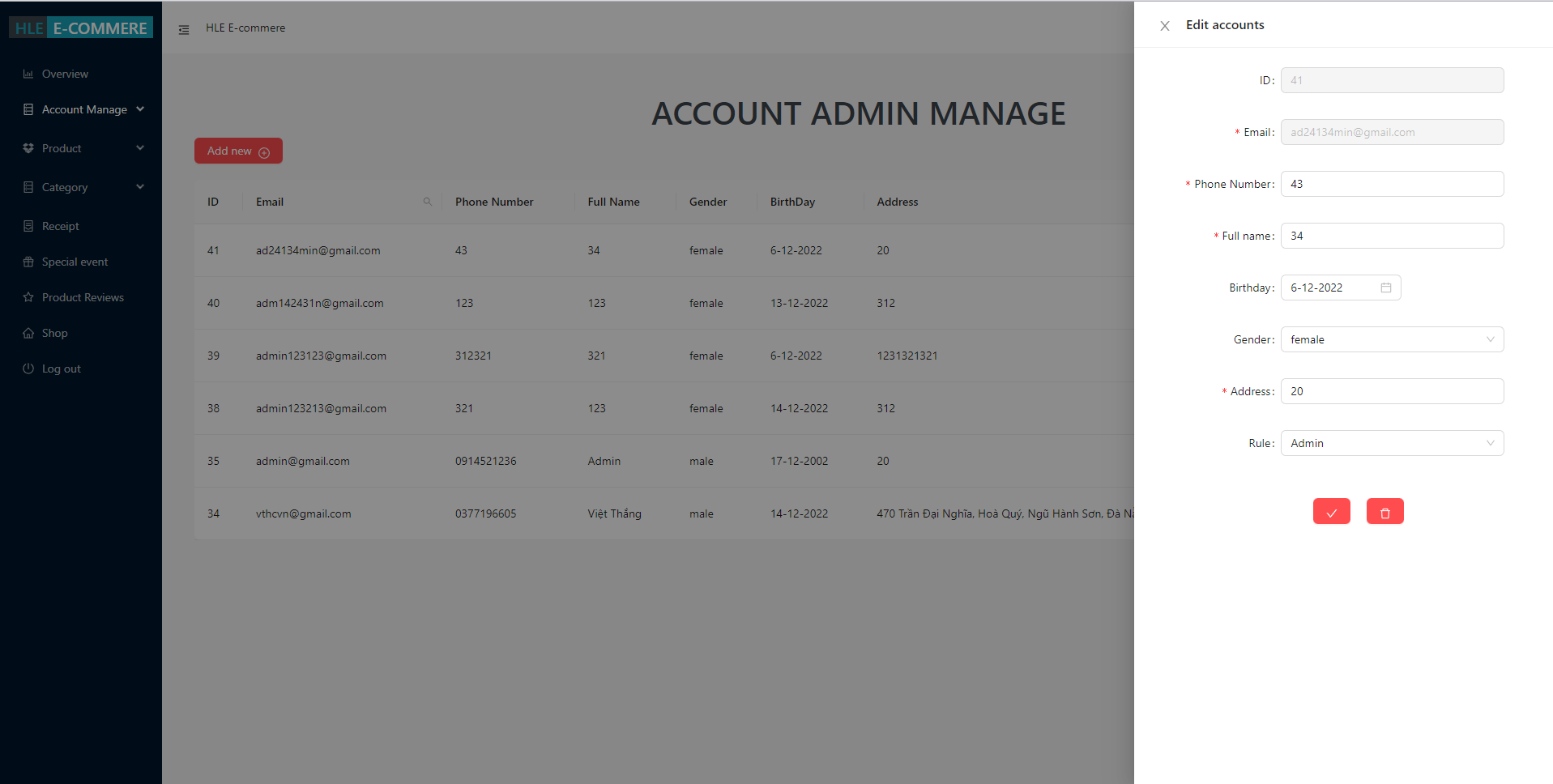


Figure 3.35 - Edit admin infomation interface

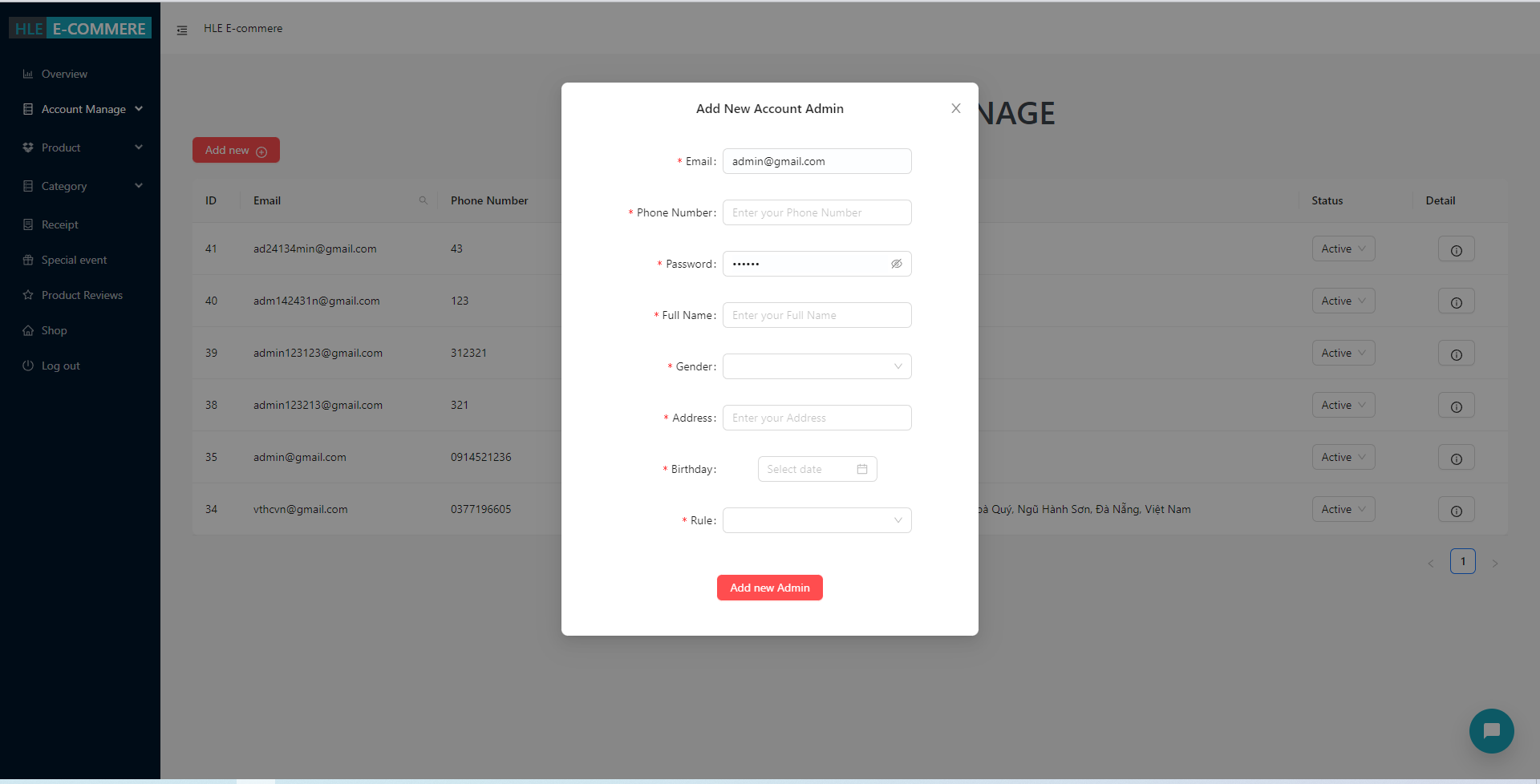


Figure 3.36 - Add account admin interface

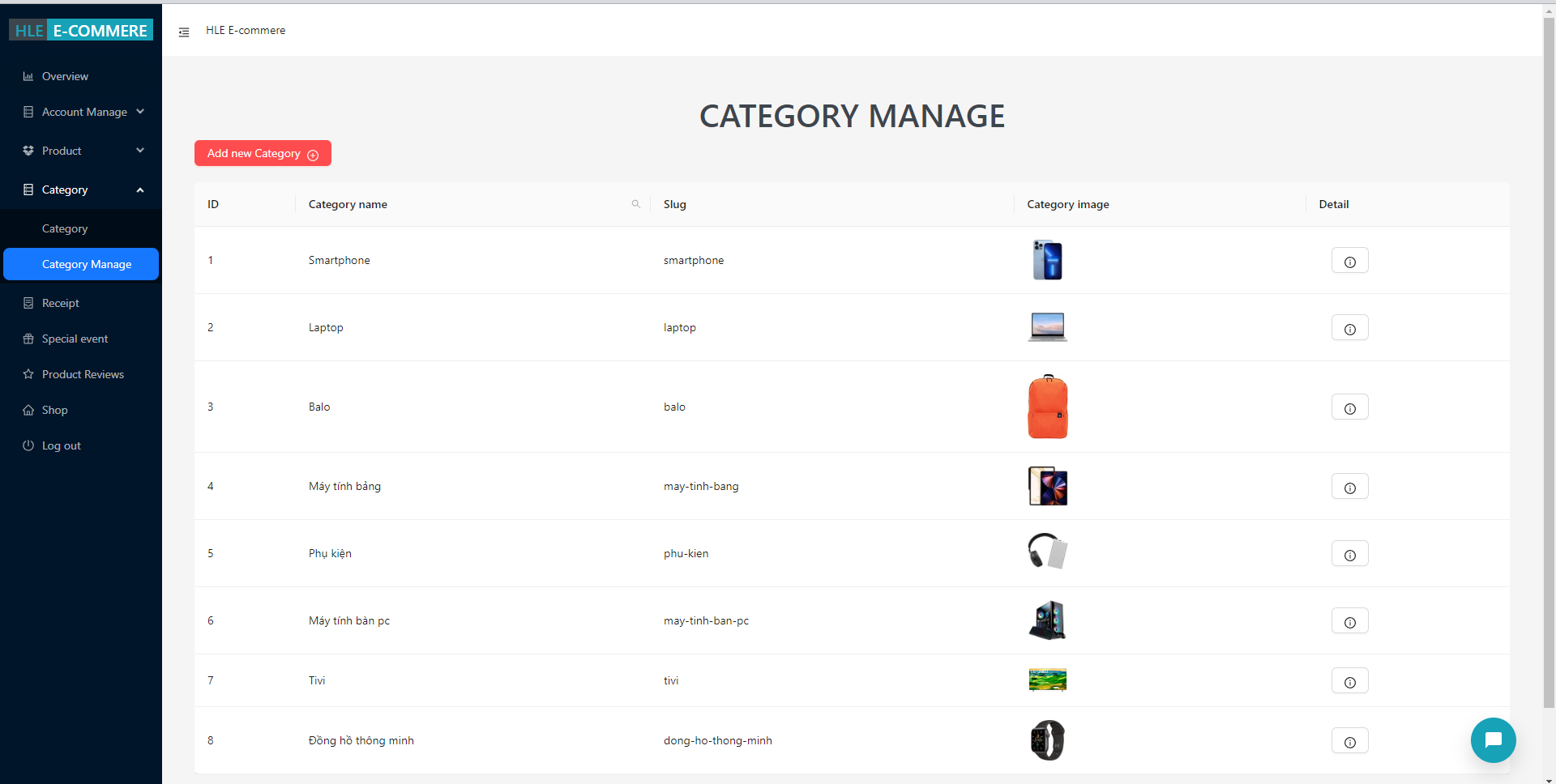


Figure 3.37 - Catagory manage interface

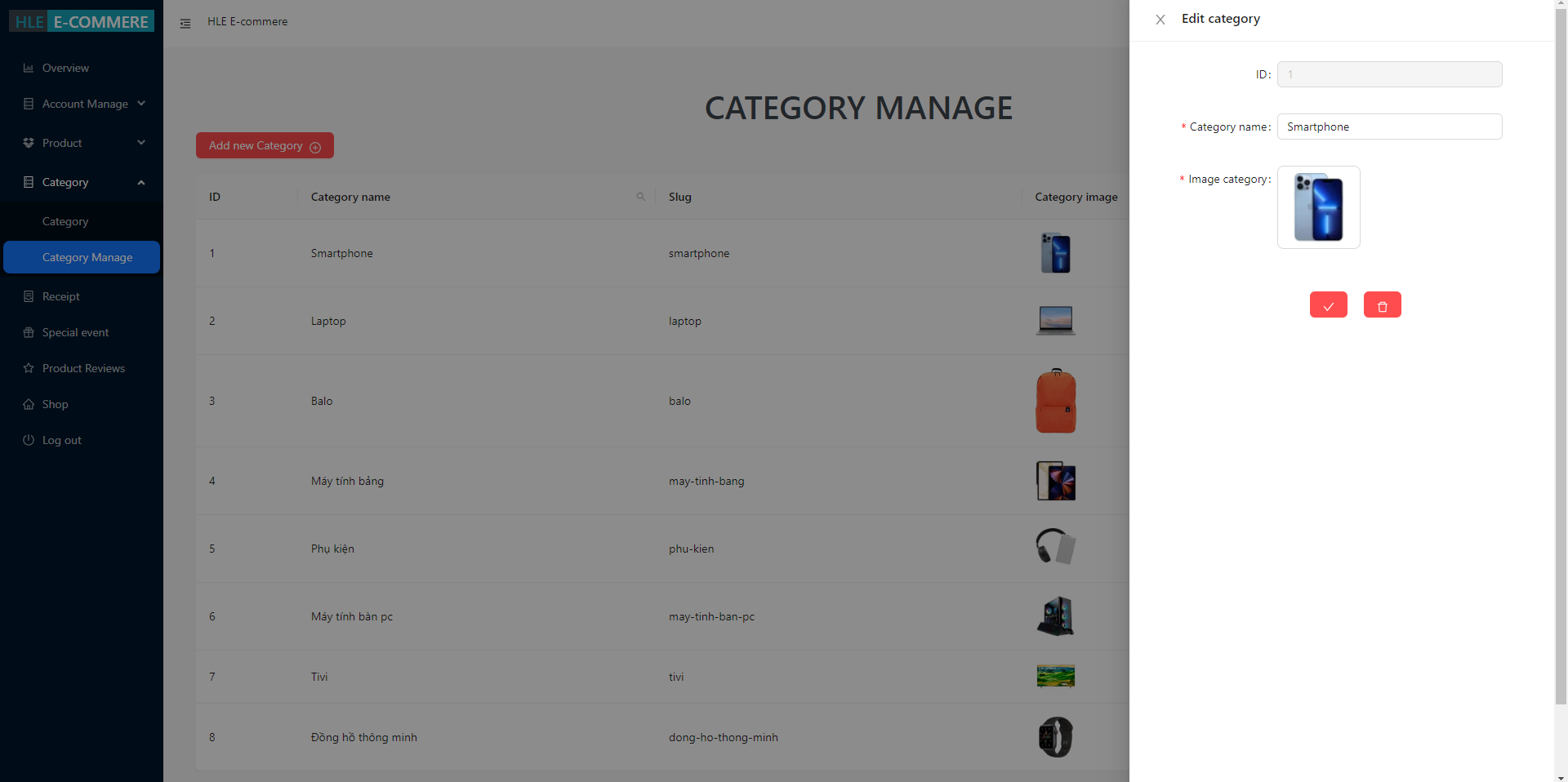


Figure 3.38 - Edit category information interface

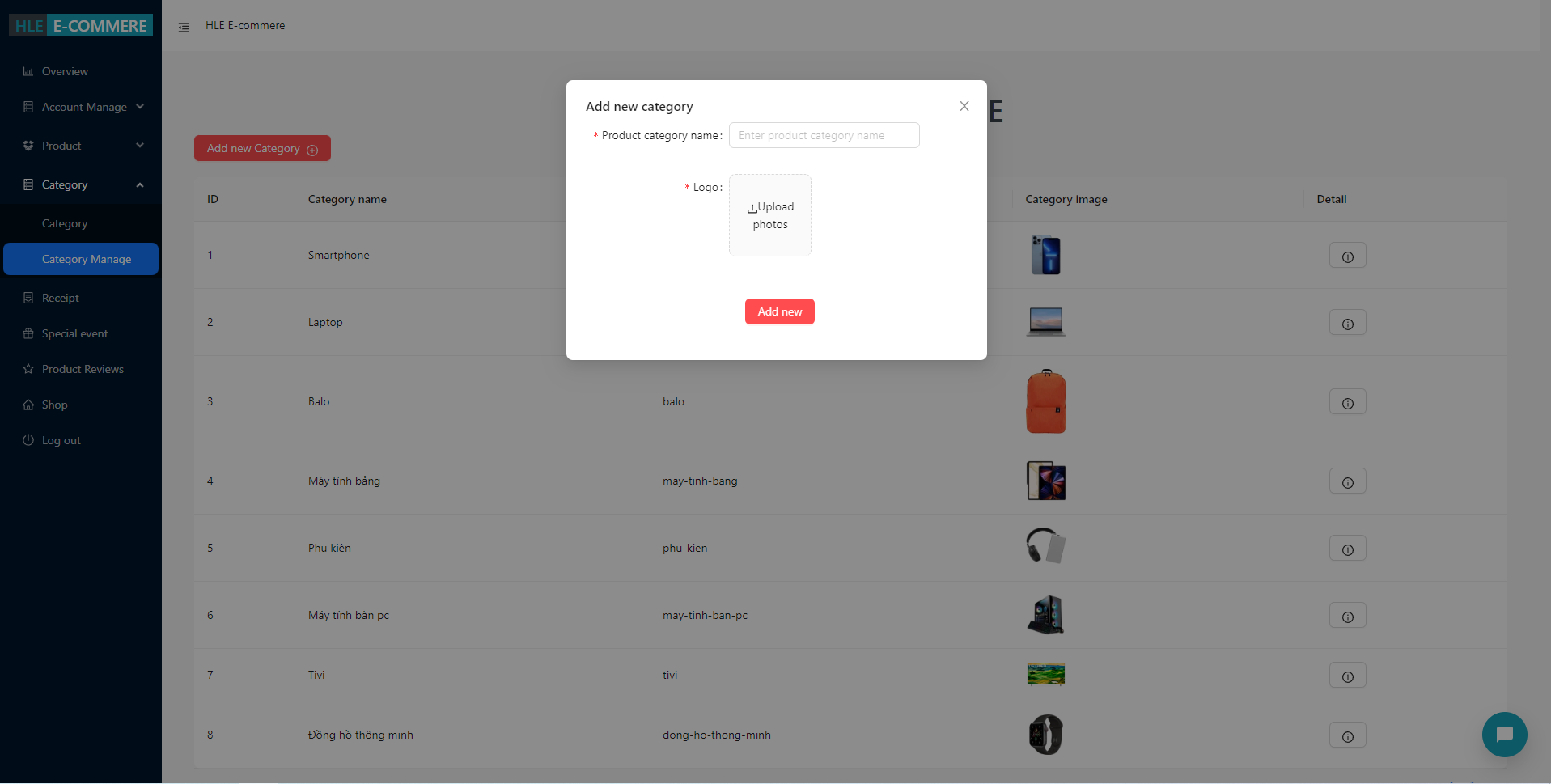


Figure 3.39 - Add category interface

## Summary

*In this chapter, we learned how to implement the commands and libraries needed to run the HLE e-commerce system.*

*Know how to configure a database connection to be able to query data from it.*

*Know how to use Flask library to be able to write API for user interface.*

*Know more about the mechanism and calculation formula to be able to calculate the correlation coefficient to suggest products that are similar to the user's preferences.*

*Get an overview of the database table, user interface and administrator interface in this system.*

CONCLUSIONS AND SUGGESTIONS

##### Conclusions

Parts achieved in this specialized project: building HLE e-commerce system.

* Build sales website for users to use.
* Build admin management system in management such as:
  + Products manage.
  + Category manage.
  + Manage discount codes.
  + Inventory manage.
* Buyer management: including managing user accounts, viewing account information,...

The existing advantages and disadvantages of the system::

* Advantage:
  + Simple admin interface, easy to understand, easy to use
  + Simple user interface, easy to use and grasp
* Defect:
  + The interface is not eye-catching.
  + Missing some functions

Through the process of completing my specialized project, I have learned a lot of new knowledge:

* How to build an admin page for good management, simple and easy to use but also full of functions.
* Build a product Recommend system for buyers.

##### Suggestions:

In the near future, The system will need to develop and upgrade some more parts like:

* Continue to upgrade the user interface to be friendly and easy to use.
* Develop cross-platform model, running on mobile app.
* Upgrading the website's security system.
* Add some other functions and features to the model.
* Expanded range of stores.

REFERENCES

1. <https://viblo.asia/p/tong-quan-ve-recommender-system-recommender-system-co-ban-phan-1-924IJGBb5PM>
2. <https://www.phamlocblog.com/2015/11/phan-tich-tuong-quan-pearson-trong-spss.html>
3. <https://sandbox.vnpayment.vn/apis/docs/huong-dan-tich-hop/>
4. <https://reactjs.org/docs/getting-started.html>