**HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**GRADUATION THESIS**

**Building ecommerce shoes website system**

**VU VAN HIEN**

[hien.vv148342@sis.hust.edu.vn](mailto:hien.vv148342@sis.hust.edu.vn)

**Information and Communication Technology**

| **Advisor:** | Dr. BUI QUOC TRUNG |
| --- | --- |
|  |  |
| **Institute:** | Information and Communication Technology |

| **HA NOI, June 2020** |
| --- |

**HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**GRADUATION THESIS**

**Building ecommerce shoes website system**

**VU VAN HIEN**

[hien.vv148342@sis.hust.edu.vn](mailto:hien.vv148342@sis.hust.edu.vn)

**Information and Communication Technology**

| **Advisor:** | Dr. BUI QUOC TRUNG |
| --- | --- |
|  |  |
| **Institute:** | Information and Communication Technology |

| **HA NOI, June 2021** |
| --- |

**GRADUATION THESIS DELEGATION SHEET**

**1.** **Student information**

Full name: **Vu Van Hien**

Mobile number: **0969708715** Email: **hien.vv148342@sis.hust.edu.vn**

Class: **LTU14** Type of tranning: **Engineer**

Graduation thesis is implemented at: Institute of Information Technology and Communications, Hanoi University of Science and Technology, Vietnam.

Time to do thesis: From February 26, 2021 to May 26, 2021

**2.** **Purpose of content of thesis**

Building ecommerce shoes website system.

**3.** **Specific tasks of the thesis**

Analyzing actual needs, necessary functions of the system. Then, analyze in detail the functions of the website system.

Produce design analysis materials and implement the project on the chosen technology.

**4.** **Student commitment**:

I *Vu Van Hien*  pledge that thesis is my own research under the guidance of *Dr. Bui Quoc Trung.*

The results stated in this thesis are honest, not a full text copy of any other work.

|  | *Ha Noi, 2021*  Thesis author  *Vu Van Hien* |
| --- | --- |

**5.** **Approval of Lecturer**:

|  | *Ha Noi, 2021*  Advisor  *Dr. Bui Quoc Trung* |
| --- | --- |

**Acknowledgement**

This project would not have been possible without the support of many people. First and unmost, I would like to express my sincere gratitude to **Dr. Bui Quoc Trung** lecturer of computer science subject for her patient guidance, critical feedback, encouragement, and constructive supervision throughout my research.

I would like to thanks to all of the teachers of Hanoi University of Science and Technology for their interesting lectures which provides me general background knowledge for this research. Lastly I am also grateful to my parents, my friends for encouraging, supporting and giving me many useful advices throughout my study.

**Abstract**

In the past, when computers were not widely used, buying and selling would usually be operated in small shops, markets, supermarkets, etc. used traditional methods which were time-consuming, labor-intensive as well as about finance. Today, with the development of science and technology and information technology, it is also very developed and widely used in all areas of life, making trading easier and more convenient.

Therefore, using technology in trading is essential to switch over from the traditional method of selling goods to electronic method of selling goods. With that in mind. I have built a project named “Building ecommerce shoes website system” to solve the problems in traditional shopping and to contribute a new way of life. I used some kinds of programming languages like HTML, CSS, JAVA, Javascript, MySQL and XAMPP..... to create an online shopping website. This website provides customers with accurate information about shoes products and they can order it easily with a few clicks. As the result, shopping is made quicker and convenient for customer.

However, the research is still facing many difficulties, so the subject has not been fully exploited, the topic still has many shortcomings, hope teachers understand.

**Table of contents**

[CHAPTER 1: INTRODUCTION 1](#_2et92p0)

[1.1 Problem 1](#_3dy6vkm)

[1.2 Objectives and scope of the topic 1](#_1t3h5sf)

[1.3 Solution Orientation 1](#_4d34og8)

[1.4 Thesis organization 2](#_2s8eyo1)

[CHAPTER 2: SURVEY AND ANALYSIS SYSTEM 3](#_3rdcrjn)

[2.1 Current status survey 3](#_26in1rg)

[2.1.1 Survey the user 3](#_lnxbz9)

[2.1.2 Surveying system available 3](#_35nkun2)

[2.1.3 Conclude 5](#_1ksv4uv)

[2.2 Functional overview 6](#_44sinio)

[2.2.1 Use-case Specification 6](#_2jxsxqh)

[2.2.2 Decomposition of Use-case Category Management 7](#_3j2qqm3)

[2.2.3 Decomposition of Use-case Account Management 8](#_4i7ojhp)

[2.2.4 Decomposition of Use-case Product Management 8](#_1ci93xb)

[2.2.5 Decomposition of Use-case Order Management 9](#_2bn6wsx)

[2.2.6 Decomposition of Use-case Brand Management 9](#_3as4poj)

[2.2.7 Decomposition of Use-case Blog Management 10](#_49x2ik5)

[2.2.8 Decomposition of Use-case Promotion Management 10](#_147n2zr)

[2.3 Usecase detail specification 11](#_3o7alnk)

[2.3.1 Login 11](#_23ckvvd)

[2.3.2 Register account 12](#_32hioqz)

[2.3.3 Product search 12](#_41mghml)

[2.3.4 Buy the product 13](#_vx1227)

[2.3.5 View product details 14](#_1v1yuxt)

[2.3.6 Add a product 14](#_2u6wntf)

[2.3.7 Order management 15](#_3tbugp1)

[2.3.8 Category management 16](#_nmf14n)

[2.3.9 Brand management 16](#_1mrcu09)

[2.3.10 Post management 17](#_2lwamvv)

[2.3.11 Promotion management 18](#_111kx3o)

[2.4 Sequence diagram 19](#_3l18frh)

[2.4.1 Login function 19](#_206ipza)

[2.4.2 Register function 19](#_4k668n3)

[2.4.3 Search function 20](#_2zbgiuw)

[2.4.4 View product function 20](#_1egqt2p)

[2.4.5 Comment function 20](#_3ygebqi)

[2.4.6 Buy product function 21](#_2dlolyb)

[2.4.7 Category management function 21](#_sqyw64)

[2.4.8 Brand management function 22](#_3cqmetx)

[2.4.9 Blog management function 22](#_1rvwp1q)

[2.4.10 Promotion management function 23](#_4bvk7pj)

[2.4.11 Order management function 23](#_2r0uhxc)

[2.4.12 Product management function 24](#_1664s55)

[CHAPTER 3: TECHNOLOGY USED 25](#_3q5sasy)

[3.1 Introduction to Spring boot 25](#_25b2l0r)

[3.1.1 Spring boot overview 25](#_kgcv8k)

[3.1.2 Advantages of Spring boot 26](#_1jlao46)

[3.1.3 Disadvantages of Spring boot 26](#_43ky6rz)

[3.2 Introduction to HTML 27](#_xvir7l)

[3.3 CSS 28](#_1x0gk37)

[3.4 JavaScript 29](#_2w5ecyt)

[3.5 MySQL 30](#_3vac5uf)

[3.6 XAMPP 31](#_pkwqa1)

[CHAPTER 4: APPLICATION DEVELOPMENT AND IMPLEMENTATION 32](#_1opuj5n)

[4.1 Architectural design 32](#_48pi1tg)

[4.2](#_1302m92) Detailed design 33

[4.2.1 Database design 33](#_3mzq4wv)

[4.3](#_184mhaj) Application construction 39

[4.3.1 Libraries and tools used 39](#_3s49zyc)

[4.3.2 Illustrate the main functions 39](#_279ka65)

[4.4](#_1d96cc0) Testing 44

[4.4.1](#_j8sehv) About the interface 44

[4.4.2](#_338fx5o) About the function 44

[4.4.3](#_1idq7dh) About performance 44

[4.5](#_42ddq1a) Deployment 44

[4.5.1](#_2hio093) Install on website server 44

[CHAPTER 5: CONCLUSION AND DEVELOPMENT DIRECTION 45](#_2pta16n)

[4.6](#_14ykbeg) Result 45

[4.7](#_3oy7u29) Restrictions exist 45

[4.8](#_243i4a2) Development 46

[References 47](#_243i4a2)

**List of Figures**

[Figure 2.2.1: Use-case Specification 6](#_z337ya)

[Figure 2.2.2: Decomposition of Use-case Category Management 7](#_1y810tw)

[Figure 2.2.3: Decomposition of Use-case Account Management 8](#_2xcytpi)

[Figure 2.2.4: Decomposition of Use-case Product Management 8](#_3whwml4)

[Figure 2.2.5: Decomposition of Use-case Order Management 9](#_qsh70q)

[Figure 2.2.6: Decomposition of Use-case Brand Management 9](#_1pxezwc)

[Figure 2.2.7: Decomposition of Use-case Blog Management 10](#_2p2csry)

[Figure 2.2.8: Decomposition of Use-case Promotion Management 10](#_2p2csry)

[Figure 2.4.1: Sequence diagram login function 19](#_wnyagw)

[Figure 2.4.2: Sequence diagram register function 19](#_wnyagw)

[Figure 2.4.3: Sequence diagram search function 20](#_wnyagw)

[Figure 2.4.4: Sequence diagram view product function 20](#_wnyagw)

[Figure 2.4.5: Sequence diagram comment function 20](#_wnyagw)

[Figure 2.4.6: Sequence diagram buy product function 21](#_wnyagw)

[Figure 2.4.7: Sequence diagram category management function 21](#_wnyagw)

[Figure 2.4.8: Sequence diagram brand management function 22](#_wnyagw)

[Figure 2.4.9: Sequence diagram blog management function 22](#_wnyagw)

[Figure 2.4.10: Sequence diagram promotion management function 23](#_wnyagw)

[Figure 2.4.11: Sequence diagram order management function 23](#_wnyagw)

[Figure 2.4.12: Sequence diagram product management function 24](#_wnyagw)

[Figure 3.1: Logo of Spring boot 25](#_34g0dwd)

[Figure 3.2: HTML 27](#_3hv69ve)

[Figure 3.3: CSS 28](#_4h042r0)

[Figure 3.4: JavaScript 29](#_1baon6m)

[Figure 3.5: MySQL 30](#_2afmg28)

[Figure 3.6: XAMPP 31](#_39kk8xu)

[Figure 4.1: MVC pattern illustration 32](#_2nusc19)

[Figure 4.2: Database 33](#_3gnlt4p)

[Figure 4.3: Home page 39](#_meukdy)

[Figure 4.4: Account registration page 40](#_36ei31r)

[Figure 4.5: Account login page 40](#_1ljsd9k)

[Figure 4.6: The page displays user information 40](#_45jfvxd)

[Figure 4.7: The page displays order history 41](#_2koq656)

[Figure 4.8: The page displays all product 41](#_zu0gcz)

[Figure 4.9: Product search page 42](#_3jtnz0s)

[Figure 4.10: The page displays the product details 42](#_1yyy98l)

[Figure 4.11: Payment 43](#_4iylrwe)

[Figure 4.12: Admin Dashboard 43](#_2y3w247)

**List of tables**

[Table 2.1: Login 11](#_ihv636)

[Table 2.2: Register account 12](#_1hmsyys)

[Table 2.3: Product search 12](#_2grqrue)

[Table 2.4: Buy the product 13](#_3fwokq0)

[Table 2.5: View product details 14](#_4f1mdlm)

[Table 2.6: Add a product 14](#_19c6y18)

[Table 2.7: Order management 15](#_28h4qwu)

[Table 2.8: Category management 16](#_37m2jsg)

[Table 2.9: Brand management 16](#_46r0co2)

[Table 2.10: Post management 17](#_46r0co2)

[Table 2.11: Promotion management 18](#_46r0co2)

[Table 4.1: Table “User” 34](#_haapch)

[Table 4.2: Table “Blog” 34](#_319y80a)

[Table 4.3: Table “Images” 35](#_1gf8i83)

[Table 4.4: Table “Brand” 35](#_40ew0vw)

[Table 4.5: Table “Category” 35](#_2fk6b3p)

[Table 4.6: Table “ Product” 35](#_1vsw3ci)

[Table 4.7: Table “Product\_Category” 36](#_upglbi)

[Table 4.8: Table “Product\_Size” 36](#_4fsjm0b)

[Table 4.9: Table “Promotion” 36](#_3ep43zb)

[Table 4.10: Table “Order” 37](#_1tuee74)

[Table 4.11: Table “Comment” 38](#_4du1wux)

[Table 4.12: Table “Statistic” 38](#_2szc72q)

# CHAPTER 1: INTRODUCTION

Chapter introduction

In this chapter, I will discuss about the trading with the growth of technology development and the application of information technology to selling with outstanding advantages. This is followed by an overview of the more general information about existing products and from which to draw comments and orientations in the software design and development process.

## **Problem**

Today, with the development and explosion of Information Technology and The Industrial Revolution 4, most informatics applications have been widely applied in life, from scientific research to various fields of social life. It has become a useful tool to help people manage, store and process information effectively in many industries such as medicine, transportation, education, etc., especially in the management in general and store management in particular.

With the diverse features of the website, it will eliminate obsolete and outdated management methods which are costly in many ways. At the same time, it will provide a simple, attractive interface so that users can easily manage and improve the system, helping store management be done quickly and avoid errors.

## Objectives and scope of the topic

The objective of the project is to make a system in java platform for manager from anywhere can also management their store. In order to build such a system complete web support need to be manager and staffs of store. A complete and efficient web system which can provide service manage product, category, brand and blog is the basic objective of the project. I hope to bring the best experience to users, like being able to can bring value to help the manager's job more easily. This system will provide in a simple and easy to operate user interface, which can be managed by any user without having prior in-depth knowledge of the computer system.

## Solution Orientation

To be able to solve the problems mentioned in the Summary and Problem statement section, I chose the topic for my graduate project to build an online selling application using web programming languages ​​and technology, HTML, CSS, JAVA, Javascript, MySQL and XAMPP.

First, the administrator will have more control over the site's data, can update order history, product statistics, open promotions for a period of specific time for each product, manage revenue and manage the payment of users, revenue management, user decentralization.

Employees can open their own accounts and manage order information, manage and split orders for shipper, check delivery history, switch order status, manage items, manage stock

Shipper can login and check to see order information assigned.

Users can open their accounts to be able to make purchases, change personal information, view purchase order history information.

## Thesis organization

The rest of the graduation thesis report is organized as follows:

In chapter 2, I will describe an overview of existing user and system surveys. Next, it is to analyze the functions and flow of operations based on UML diagrams, design class diagrams, briefly describe classes, design databases and briefly describe tables in the database.

In chapter 3, I will list and introduce the technologies and programming languages used in graduation thesis.

In chapter 4, I will include the conclusions, the results achieved after the development of the graduation thesis, the limitations that exist, the future direction of development, and finally the list of references have been used in the construction of the graduation thesis.

In chapter 5 is a summary of what you have done in your system, besides there are limitations and future developments to improve the system.

# CHAPTER 2: SURVEY AND ANALYSIS SYSTEM

In order to solve the problems in chapter 1 and develop an online selling system, chapter 2 will go to the survey and analysis of the requirements in an online selling system. In section 1, the target group I aimed for is those who have been using online selling sites today. In section 2, I choose an existing system and analyze the necessary functions of that system to draw clear goals for my system.

## 2.1 Current status survey

### 2.1.1 Survey the user

Surveying a few users who are using online selling offers some of the following requirements:

* The first is to register / login to manage that account.
* Basic user functions such as changing my information, password, avatar.
* Check order status
* Transaction history on the system
* Product variety and full of information such as item name, price, category, description,…
* Diverse product types
* Apply discount function to help attract new users and returning users
* Having full information, contact address, phone number, managing unit, …
* Having the function of customer care, direct and indirect chat
* There are functions to search, sort and filter

### 2.1.2 Surveying system available

#### 2.1.2.1 Survey Lazada.vn system

Lazada first entered the Vietnamese market in 2012, which is a member of Lazada Group - Southeast Asia's number one online shopping mall. Lazada offers customers an effective shopping experience. Just visit Lazada's website, your shopping needs are always met most effectively. By having chosen Lazada, I have drawn some important points in their system:

How to buy: search and choose the items - chatting with seller ( if you want) - check out - payment.

Purchase experience: You can buy at home, at the office, coffee shop or anywhere, anytime as long as your device (desktop, laptop, smartphone...) is connected to the internet.

Easy to handle and grasp: Products presented in detail, including images, prices, information, ... are posted on the website by the seller. It is divided into categories so that buyers easily find the product they are looking for.

Available functions on Lazada system:

* Register, log in, forget password via Email, social networks (Facebook, Google+)
* Personal account management, edit information, representative images
* Check order status, transaction history on the system
* Diversity of products: electronic device, beauty products, clothes, toy, car, motorcycle,….
* Product variety and full of information such as item name, price, category, description, rating, comment...
* Search, sort, filter product by topics, highlights, latest or by price...
* User can talk to the seller before payment to get the best choice
* Assess the satisfaction with the product evaluation function when a product has been purchased successfully
* Save to cart if you intend to buy or buy multiple items in one payment

There are 3 types of payment that Lazada currently uses:

* Pay online by bank card
* Pay online with MOMO e-wallet
* Payment on delivery

Sales, promotions and combos are pushed to the top by Lazada to help attract user

When buying a product, if a discount code is available, it will be directly discounted on the customer's purchase invoice

Based on the above analysis points, I draw a point to note with the Lazada system:

Products are divided into separate categories. Including household appliances, fashion accessories, clothes, toys .. It can be seen that the Lazada object focuses on all ages.

#### 2.1.2.2 Survey system Kingshoes.vn

Available functions on “Kingshoes.vn” system:

* Register, log in, forget password via Email, phone number.
* Personal account management, edit information.
* Check order status, transaction history on the system.
* Bonus points can be accumulated after purchase.
* Diversity of shoes product
* Product variety and full of information such as item name, price, category, description, ….
* Search, sort, filter products by topics, highlights, latest or by price...
* Save to cart if you intend to buy.

Unlike Lazada, Kingshoes only focuses on shoes items. Kingshoes meets the needs of many people with diverse products. The products are divided into different ages and genders so that it is easy for customers to find the one they need.

### 2.1.3 Conclude

Based on the analysis in the above 2 sections, I draw on some of the functions needed in an brand-shop system:

* Register, log in.
* Personal account management, edit information.
* Check order status, transaction history on the system
* Product variety and full of information such as item name, price, category, description, rating,...
* Payment

## 2.2 Functional overview

### 2.2.1 Use-case Specification

Diagram

Description automatically generated

*Figure 2.2.1: Use-case Specification*

Figure 2.1 shows the general Use Case diagram of the system. In the figure we can see that there are two main actors and functions in the system corresponding to the agents as follows:

User agent: The most basic type of agent, which can perform basic functions such as:

* Registration
* Login
* Account management
* Search product
* View product information
* Comment product
* View blog
* Comment blog
* Payment
* View Purchase Order History

Administrator agent (admin): The type of agent with the highest authority in the system, in addition to inheriting the functions of users, administrators can

* Manage members on the system
* Category management
* Brand management
* Blog management
* Product management
* Promotion management
* Statistics

### 2.2.2 Decomposition of Use-case Category Management

Diagram

Description automatically generated

*Figure 2.2.2: Decomposition of Use-case Category Management*

**View Category**: View information of category: name.

**Edit Category**: Create or delete information of category: name

**Delete Category**: Delete category

**Create new Category**: Create information of category

### 2.2.3 Decomposition of Use-case Account Management

Diagram

Description automatically generated

*Figure 2.2.3: Decomposition of Use-case Account Management*

**Edit information**: Create or delete information: name, email, telephone, birthday , address.

**Security**: Edit password

**View information**: View the current account information

### 2.2.4 Decomposition of Use-case Product Management

Diagram

Description automatically generated

*Figure 2.2.4: Decomposition of Use-case Product Management*

**View product**: View information of product: category, size, brand, image, price, quantity.

**Delete product**: Delete product

**Edit product**: Create, Delete information of product: category, brand, size, price, image, quantity.

**Create new product**: Create new content of product

### 2.2.5 Decomposition of Use-case Order Management

Diagram

Description automatically generated

*Figure 2.2.5: Decomposition of Use-case Order Management*

**View order:** View information of order: codes, prices, payments

**Edit order:** Check delivery history of order, create, delete information of order.

**Delete order:** Delete order.

**Create new order:** Create new content of order.

### 2.2.6 Decomposition of Use-case Brand Management

Diagram

Description automatically generated

*Figure 2.2.6: Decomposition of Use-case Brand Management*

**View brand:** View information of Brand: name, image.

**Edit brand:** Create, delete information of brand: name, image.

**Delete brand:** Delete brand.

**Create new brand:** Create new content brand.

### 2.2.7 Decomposition of Use-case Blog Management

Diagram

Description automatically generated

*Figure 2.2.7: Decomposition of Use-case Blog Management*

**View blog:** View information of blog: title, content, image, status.

**Edit blog:** Create, delete information of blog: title, content, image, status.

**Delete blog:** Delete blog

**Create new blog:** Create new content of blog

### 2.2.8 Decomposition of Use-case Promotion Management

Diagram

Description automatically generated

*Figure 2.2.8: Decomposition of Use-case Blog Management*

**View promotion:** View information of promotion: Name, code, expiry date, status.

**Edit promotion:** Create, delete information of blog: Name, expiry date, status.

**Delete promotion:** Delete promotion.

**Create new promotion:** Create new content of promotion.

## 2.3 Usecase detail specification

### 2.3.1 Login

*Table 2.1: Login*

| **Use case name** | Login |
| --- | --- |
| **Usecase ID** | UC-01 |
| **Brief Description** | The System allow user access. |
| **Trigger** | When user want to login in the system. |
| **Actors** | All users |
| **Preconditions** | Website URL is accessed and system interface is loaded |
| **Main Flows** | 1. User click button “Login” 2. The system displays the registered interface. 3. The user is expected to enter the email and password. 4. The system retrives the email and password on the database. 5. The system compares the email and password entered with the email and password from the database. 6. The system allows user access. |
| **Alternative Flow** | 1. Invalid Password and Email    1. The system informs the user that Email or password is invalid 2. User has entered invalid email or password    1. The alternative flow begins after    2. The system informs the user that his or her email or password entered is invalid 3. User forgets password    1. System displays message to connect system administrator |
| **Exceptions** | User access has been revoked |
| **Post-conditions** | The website is open and display user functions |

### 2.3.2 Register account

*Table 2.2: Register account*

| **Use case name** | Register account |
| --- | --- |
| **Usecase ID** | UC-02 |
| **Brief Description** | The System register new a account |
| **Trigger** | When user want to create a new account |
| **Actors** | Customer |
| **Preconditions** | Website URL is accessed and system interface is loaded |
| **Main Flows** | 1. User select register on the Main system menu. 2. The system provides the user with a form consisting of all required fields. 3. User enters all required fields and submits information. 4. The system display notification for successful registration |
| **Alternative Flow** | 1. The user enters usename already exists    1. Require enter again another username |
| **Exceptions** | User access has been revoked |
| **Post-conditions** | A new account |

### 2.3.3 Product search

*Table 2.3: Product search*

| **Use case name** | Product search |
| --- | --- |
| **Usecase ID** | UC-03 |
| **Brief Description** | The System finds some products based on User search criteria and displays them to the User. |
| **Trigger** | When user want to search a product |
| **Actors** | All Users |
| **Preconditions** | None |
| **Main Flows** | 1. The User selects ‘find product’ 2. The system displays page search 3. The User enters the requested criteria 4. The system searchs for products that match the User’s criteria 5. If the system finds some matching products then:    1. The system displays products. 6. Else    1. The system tells the User that no matching products could be found |
| **Alternative Flow** | None |
| **Exceptions** | None |
| **Post-conditions** | Display the list of products that match your search |

### 2.3.4 Buy the product

*Table 2.4: Buy the product*

| **Use case name** | Buy the product |
| --- | --- |
| **Usecase ID** | UC-04 |
| **Brief Description** | The system allows users to purchase the product they selected. |
| **Trigger** | When user choose to pay for a product of their choice |
| **Actors** | Customer |
| **Preconditions** | User login successfully and select desired product to pay. |
| **Main Flows** | 1. User selectable products. 2. The system displays information about the product. 3. User choose the "buy now" function 4. The system displays the form of payment and user information 5. User select form of payment and payment. 6. The system displays the notice of a successful product purchase. |
| **Alternative Flow** | None |
| **Exceptions** | 1. User has not selected the product to purchase.   * 1. The system notifies that no product have been added to the shopping cart yet |
| **Post-conditions** | The product is successfully purchased corresponding to the user who purchased it. |

### 2.3.5 View product details

*Table 2.5: View product details*

| **Use case name** | View product details |
| --- | --- |
| **Usecase ID** | UC-05 |
| **Brief Description** | The System allow users to view product details. |
| **Trigger** | When users select a product they need to buy. |
| **Actors** | Customer |
| **Preconditions** | None |
| **Main Flows** | 1. User select a product. 2. System display product details. 3. The user selects the product they want to watch to buy. 4. The system displays the corresponding product. |
| **Alternative Flow** | None |
| **Exceptions Flow** | None |
| **Post-conditions** | None |

### 2.3.6 Add a product

*Table 2.6: Add a product*

| **Use case name** | Add a product |
| --- | --- |
| **Usecase ID** | UC-08 |
| **Brief Description** | The system allows employee to add new product. |
| **Trigger** | When the employee wants to add a new product |
| **Actors** | Administrator |
| **Preconditions** | The admin has logged in as a admin. |
| **Main Flows** | 1. Admin choose additional product 2. The system displays the form interface to fill out basic information about the product 3. Admin complete the product information. 4. The system informs the product successfully. |
| **Alternative Flow** | None |
| **Exceptions Flow** | 1. Admin filled out missing product information.    1. The system notifies you that the product has failed. |
| **Post-conditions** | A new product is created. |

### 2.3.7 Order management

*Table 2.7: Order management*

| **Use case name** | Order management |
| --- | --- |
| **Usecase ID** | UC-9 |
| **Brief Description** | The system allows admin to manage orders. |
| **Trigger** | When the admin wants to manage order |
| **Actors** | Administrator |
| **Preconditions** | The admin has logged in as a admin. |
| **Main Flows** | 1. Admin logins to System. 2. Login success. 3. Admin uses function manage orders. 4. System shows list of orders. |
| **Alternative Flow** | None |
| **Exceptions Flow** | 1. The admin logins wrong username and password.    1. Notify to user that this information is not correct. |
| **Post-conditions** | View information of all orders. |

### 2.3.8 Category management

*Table 2.8: Category management*

| **Use case name** | Category management |
| --- | --- |
| **Usecase ID** | UC-10 |
| **Brief Description** | The system allows admin to manage their category. |
| **Trigger** | When the admin wants to manage category |
| **Actors** | Administrator |
| **Preconditions** | The admin has logged in as a admin. |
| **Main Flows** | 1. Admin logins to System. 2. Login success. 3. Admin uses function manage categories. 4. System shows list of categories. |
| **Alternative Flow** | None |
| **Exceptions Flow** | 1. The admin logins wrong username and password.    1. Notify to user that this information is not correct. |
| **Post-conditions** | View information of all category. |

### 2.3.9 Brand management

*Table 2.9: Brand management*

| **Use case name** | Brand management |
| --- | --- |
| **Usecase ID** | UC-11 |
| **Brief Description** | The system allows administrators to brand manage. |
| **Trigger** | When the admin wants to brand management. |
| **Actors** | Admin |
| **Preconditions** | The admin has logged in as a admin. |
| **Main Flows** | 1. Admin logins to System. 2. Login success. 3. Admin uses function brand management. 4. System shows list of brand management. |
| **Alternative Flow** | None |
| **Exceptions Flow** | 1. The admin logins wrong username and password.    1. Notify to user that this information is not correct. |
| **Post-conditions** | View information of all brands. |

### 2.3.10 Post management

*Table 2.10: Post management*

| **Use case name** | Post management |
| --- | --- |
| **Usecase ID** | UC-12 |
| **Brief Description** | The system allows administrators to blog manage. |
| **Trigger** | When the admin wants to blog manage. |
| **Actors** | Admin |
| **Preconditions** | The admin has logged in as a admin. |
| **Main Flows** | 1. Admin logins to System. 2. Login success. 3. Admin uses function blog management. 4. System shows list of blog management. |
| **Alternative Flow** | None |
| **Exceptions Flow** | 1. The admin logins wrong username and password.    1. Notify to user that this information is not correct. |
| **Post-conditions** | View information of all blogs. |

### 2.3.11 Promotion management

*Table 2.11: Promotion management*

| **Use case name** | Promotion management |
| --- | --- |
| **Usecase ID** | UC-13 |
| **Brief Description** | The system allows admin to manage their promotion. |
| **Trigger** | When the admin wants to manage promotion |
| **Actors** | Administrator |
| **Preconditions** | The admin has logged in as a admin. |
| **Main Flows** | 1. Admin logins to System. 2. Login success. 3. Admin uses function manage promotions. 4. System shows list of promotions. |
| **Alternative Flow** | None |
| **Exceptions Flow** | 1. The admin logins wrong username and password.    1. Notify to user that this information is not correct. |
| **Post-conditions** | View information of all promotion. |

## 2.4 Sequence diagram

### 2.4.1 Login function

Diagram

Description automatically generated

*Figure 2.4.1: Sequence diagram login function*

### 2.4.2 Register function

Diagram

Description automatically generated

Figure 2.4.2: Sequence diagram register function

### 2.4.3 Search function

Diagram

Description automatically generated

Figure 2.4.3: Sequence diagram search function

### 2.4.4 View product function

Diagram

Description automatically generated

Figure 2.4.4: Sequence diagram view product function

### 2.4.5 Comment function

Diagram

Description automatically generated

Figure 2.4.5: Sequence diagram comment function

### 2.4.6 Buy product function

Diagram

Description automatically generated

Figure 2.4.6: Sequence diagram register function

### 2.4.7 Category management function

A picture containing calendar

Description automatically generated

Figure 2.4.7: Sequence diagram category management function

### 2.4.8 Brand management function

A picture containing diagram

Description automatically generated

Figure 2.4.8: Sequence diagram brand management function

### 2.4.9 Blog management function

Diagram

Description automatically generated

Figure 2.4.9: Sequence diagram register function

### 2.4.10 Promotion management function

Diagram

Description automatically generated

Figure 2.4.10: Sequence diagram promotion management function

### 2.4.11 Order management function

Diagram

Description automatically generated

Figure 2.4.11: Sequence diagram order management function

### 2.4.12 Product management function

Diagram

Description automatically generated

Figure 2.4.12: Sequence diagram product management function

# CHAPTER 3: TECHNOLOGY USED

By examining and analyzing in detail the functions needed in your system as described in chapter 2. Chapter 3 is the selection of technologies and platforms to use for your system to solve problems on.

## 3.1 Introduction to Spring boot

### 3.1.1 Spring boot overview

Spring Boot helps you to create stand-alone, production-grade Spring-based Applications that you can run. We take an opinionated view of the Spring platform and third-party libraries, so that you can get started with minimum fuss. Most Spring Boot applications need very little Spring configuration.

You can use Spring Boot to create Java applications that can be started by using java -jar or more traditional war deployments. We also provide a command line tool that runs “spring scripts”.

Our primary goals are:

* Provide a radically faster and widely accessible getting-started experience for all Spring development.
* Be opinionated out of the box but get out of the way quickly as requirements start to diverge from the defaults.
* Provide a range of non-functional features that are common to large classes of projects (such as embedded servers, security, metrics, health checks, and externalized configuration).
* Absolutely no code generation and no requirement for XML configuration.

Logo, company name

Description automatically generated

*Figure 3.1: Logo of Spring boot*

### 3.1.2 Advantages of Spring boot

   Simple configuration

    - The spring boot embeds the servlet container, which reduces the environmental requirements. The machine has a java operating environment, and the project can be packaged into a jar package, and the java command is used to java -jar \*\*\*\*.jar to execute.

    - Quickly integrate third-party frameworks without configuration files

    - Solve the shortcomings of Spring

    - Less code, fewer configuration files, no need to worry about third-party frameworks, simplified projects, and greater cost savings for the development and maintenance of the entire team.

    - It is very easy to develop Spring-based applications using Java or Groovy.

    - It reduces a lot of development time and increases productivity.

    - It avoids writing a lot of boilerplate code, comments and XML Configuration.

    - Spring Boot applications and their Spring ecosystem (such as Spring JDBC, Spring ORM, Spring Data, Spring Security, etc.) The integration is very easy.

         It follows the "self-use default configuration" method to reduce development workload.

         It provides embedded HTTP server, such as Tomcat, Jetty, etc., to develop and test web applications very easily.

         It provides CLI (Command Line Interface) tools from the command prompt to develop and test Spring Boot (Java or Groovy) applications very easily and quickly.

         It provides many plugins to develop and test Spring Boot applications very easy to use build tools such as Maven and Gradle.

         It provides many plug-ins to make working with embedded and in-memory databases very easy.

.

### 3.1.3 Disadvantages of Spring boot

The biggest challenge many developers face when using Spring Boot is the lack of control. The opinionated style installs many additional dependencies (that often go unused) which increases the deployment file size.

The Spring Boot artifact may be run directly in Docker containers. This is useful to get when you need to quickly create microservices. Yet, some developers argue that since Spring Boot was designed to be lightweight and agile, it should therefore not be used for monolithic applications.

Though Spring Boot comes with some basic tools for logs and your app health monitoring, these aren’t sufficient. Tools like Retrace help teams to monitor Java apps with ease. This tool helps to detect slow SQL queries, provides performance and CPU usage reports and shows the most common errors by interpreting the logs.

On top of that, it can be quite challenging to update your legacy Spring code. You can overcome this problem by using tools such as the Spring Boot CLI (Command Line Interface) that will help you convert your legacy code.

Some other disadvantages are:

* If you have never worked with Spring before and want to learn about proxies, dependency injection, and AOP programming, it is not recommended to start with Spring Boot because it doesn’t cover most of these details.
* You really have to understand a lot of the underlying Spring systems (and a bit of Spring history too), along with some advanced topics in order to modify and troubleshoot it.
* Spring Boot works well with microservices. The Spring Boot artifacts can be deployed directly into Docker containers. However, some developers don’t recommend the framework for building large and monolithic apps.
* If you are not familiar with other projects of the Spring ecosystem like Spring Security, Spring AMQP, Spring Integration, etc), using them with Spring Boot will make you miss many concepts that you would grasp if you had started using them independently.

## 3.2 Introduction to HTML



*Figure 3.2: HTML*

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the » W3C, the organisation charged with designing and maintaining the language.

The definition of HTML is HyperText Markup Language.

HyperText is the method by which you move around on the web - by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear - i.e. you can go to any place on the Internet whenever you want by clicking on links - there is no set order to do things in.

Markup is what HTML tags do to the text inside them. They mark it as a certain type of text (italicised text, for example).

HTML is a Language, as it has code-words and syntax like any other language.

HTML is the language for describing the structure of Web pages. HTML gives authors the means to:

* Publish online documents with headings, text, tables, lists, photos, etc.
* Retrieve online information via hypertext links, at the click of a button.
* Design forms for conducting transactions with remote services, for use in searching for information, making reservations, ordering products, etc.
* Include spread-sheets, video clips, sound clips, and other applications directly in their documents.

With HTML, authors describe the structure of pages using markup. The elements of the language label pieces of content such as “paragraph,” “list,” “table,” and so on.

## 3.3 CSS



*Figure 3.3: CSS*

CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments. This is referred to as the separation of structure (or: content) from presentation.

## 3.4 JavaScript



*Figure 3.4: JavaScript*

JavaScript was initially created to “make web pages alive”.

The programs in this language are called scripts. They can be written right in a web page’s HTML and run automatically as the page loads. Scripts are provided and executed as plain text. They don’t need special preparation or compilation to run. In this aspect, JavaScript is very different from another language called Java.

Today, JavaScript can execute not only in the browser, but also on the server, or actually on any device that has a special program called the JavaScript engine.

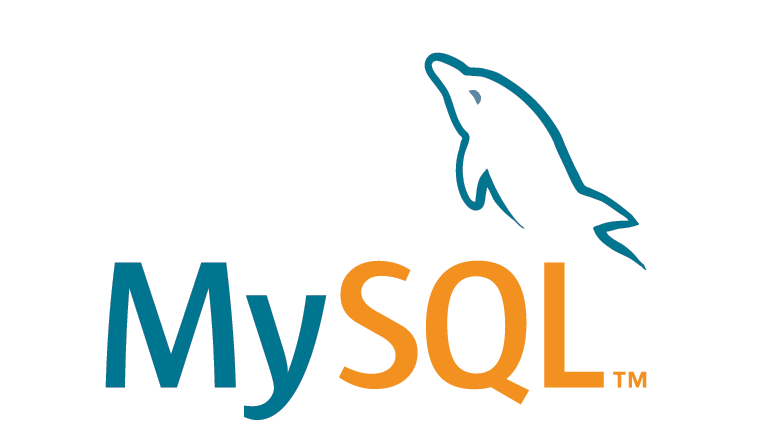
The browser has an embedded engine sometimes called a “JavaScript virtual machine”.

Different engines have different “codenames”. For example:

* V8 – in Chrome and Opera.
* SpiderMonkey – in Firefox.
* …There are other codenames like “Trident” and “Chakra” for different versions of IE, “ChakraCore” for Microsoft Edge, “Nitro” and “SquirrelFish” for Safari, etc.

The terms above are good to remember because they are used in developer articles on the internet. We’ll use them too. For instance, if “a feature X is supported by V8”, then it probably works in Chrome and Opera.

## 3.5 MySQL



*Figure 3.5: MySQL*

MySQL is a leading open source database management system. It is a multi-user, multithreaded database management system. MySQL is especially popular on the web. It is one of the parts of the very popular LAMP platform. Linux, Apache, MySQL and PHP. MySQL database is available on most important OS platforms. It runs on BSD Unix, Linux, Windows or Mac. Wikipedia, YouTube, Facebook use MySQL. These sites manage millions of queries each day. MySQL comes in two versions: MySQL server system and MySQL embedded system. The MySQL server software and the client libraries are dual-licensed: GPL version 2 and proprietary license.

The development of MySQL began in 1994 by a Swedish company MySQL AB. Sun Microsystems acquired MySQL AB in 2008. Sun was bought by Oracle in 2010.

MySQL, PostgreSQL, Firebird, SQLite, Derby, and HSQLDB are the most well known open source database systems.

MySQL is developed in C/C++. Except for C/C++, APIs exist for PHP, Python, Java, C#, Eiffel, Ruby, Tcl or Perl.

## 3.6 XAMPP



*Figure 3.6: XAMPP*

XAMPP is the title used for a compilation of free software. The name is an acronym, with each letter representing one of the five key components. The software packet contains the web server Apache, the relational database management system MySQL (or MariaDB), and the scripting languages Perl and PHP. The initial X stands for the operating systems that it works with: Linux, Windows, and Mac OS X.

Apache: ihe open source web server Apache is the most widely used server worldwide for delivery of web content. The server application is made available as a free software by the Apache Software Foundation.

MySQL/MariaDB: in MySQL, XAMPP contains one of the most popular relational database management systems in the world. In combination with the web server Apache and the scripting language PHP, MySQL offers data storage for web services. Current XAMPP versions have replaced MySQL with MariaDB (a community-developed fork of the MySQL project, made by the original developers).

PHP: the server-side programming language PHP enables users to create dynamic websites or applications. PHP can be installed on all platforms and supports a number of diverse database systems.

Perl: the scripting language Perl is used in system administration, web development, and network programming. Like PHP, Perl also enables users to program dynamic web applications.

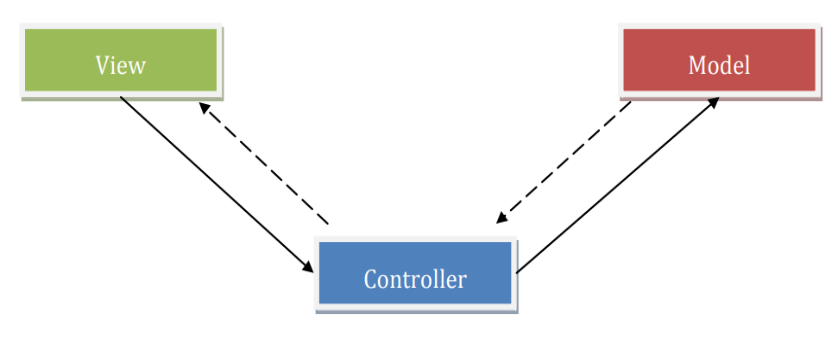
Alongside these core components, this free-to-use Apache distribution contains some other useful tools, which vary depending on your operating system. These tools include the mail server Mercury, the database administration tool phpMyAdmin, the web analytics software solutions Webalizer, OpenSSL, and Apache Tomcat, and the FTP servers FileZilla or ProFTPd.

# CHAPTER 4: APPLICATION DEVELOPMENT AND IMPLEMENTATION

Combined with the technology selection in chapter 3, the choice of software architecture for my system will be outlined in chapter 4. Chapter 4 will describe the architecture specific to my system.

## 4.1 Architectural design

Model-View-Control (MVC) is a software architecture, currently considered a design pattern in software engineering. The MVC model separates the data processing part from the interface, allowing the development, testing and maintenance of components independently.



*Figure 4.1: MVC pattern illustration*

Whereby:

* Model showing data structures Classes of the Model component often perform tasks such as querying, adding, deleting, and updating data. When the data in the Model changes, the View component will be updated.
* View is a component representing data in the Model into interfaces that interact with users. A model can have multiple views depending on different purposes.
* Controller acts as an intermediary between Model and View. The user information from the View is sent to the Controller for processing, then the Controller interacts with the Model to retrieve the requested data, and the Controller finally returns this data to the View.

The MVC pattern is often used in web applications, because the View component (HTML / XHTML code) is generated from web programming languages. Controller component will receive GET / POST data, process these data, then switch to Processing Model. The model will return data to the Controller, then the Controller generates HTML / XHTML code to be shown in the View.

## Detailed design

### 4.2.1 Database design

Diagram, schematic

Description automatically generated

*Figure 4.2: Database*

#### 4.2.1.1 Table “User”

*Table 4.1: Table “User”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | BIGINT | Used to contain a unique identifier for each data and as a primary key |
| 2 | email | VARCHAR(255) | Contains user email |
| 3 | password | VARCHAR(255) | Contains passwords when logged in by users |
| 4 | full\_name | VARCHAR(255) | Contains the first name and last name of the user |
| 5 | phone | VARCHAR(255) | Contains telephone user information |
| 6 | created\_at | TIMESTAMP | Account creation time |
| 7 | role | JSON | Permission array  Default: ["USER"] |
| 8 | address | VARCHAR(255) | Contains address of user |
| 9 | status | BOOLEAN | Account status: true - activate, false - disable |

#### 4.2.1.2 Table “Blog”

*Table 4.2: Table “Blog”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | BIGINT | Used to contain a unique identifier for each data and as a primary key |
| 2 | content | TEXT | Contains content |
| 3 | description | TEXT | Contains description |
| 4 | slug | VARCHAR(600) | Contains path |
| 5 | title | VARCHAR(300) | Contains title |
| 6 | thumbnail | VARCHAR(255) | Contains thumbnail |
| 7 | created\_at | TIMESTAMP | Contains post creation date |
| 8 | created\_by | BIGINT | Contains gender information |
| 9 | modified\_at | TIMESTAMP | Last modified time |
| 10 | modified\_by | BIGINT | Last modified person ID |
| 11 | published\_at | TIMESTAMP | Public time |
| 12 | status | INT | Post status: 0 - draft, 1 - public |

#### 4.2.1.3 Table “Images”

*Table 4.3: Table “Images”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | Id | VARCHAR(255) | Used to contain a unique identifier for each data and as a primary key |
| 2 | link | VARCHAR(255) | Image link |
| 3 | name | VARCHAR(255) | Contains file name of image |
| 4 | size | BIGINT | Image Size (Kb) |
| 5 | type | VARCHAR(255) | Picture type |
| 6 | uploaded\_at | TIMESTAMP | Upload time |
| 7 | uploaded\_by | BIGINT | Upload user ID |

#### 4.2.1.4 Table “Brand”

*Table 4.4: Table “Brand”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | INT | Used to contain a unique identifier for each data and as a primary key |
| 2 | name | VARCHAR(255) | Contains name of brand |
| 3 | thumbnail | VARCHAR(255) | Logo image link |

#### 4.2.1.5 Table “Category”

*Table 4.5: Table “Category”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | INT | Used to contain a unique identifier for each data and as a primary key |
| 2 | name | VARCHAR(255) | Contains name of category |

#### 4.2.1.6 Table “Product”

*Table 4.6: Table “Product”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | VARCHAR(255) | Used to contain a unique identifier for each data and as a primary key |
| 2 | brand\_id | BIGINT | Contains the binding key value of the “Brand” table |
| 3 | image | JSON | Contains a representative image of the product |
| 4 | Image\_feedback | JSON | Contains a feedback image of the product |
| 5 | name | VARCHAR(255) | Contains name of product |
| 6 | slug | VARCHAR(255) | Support seo |
| 7 | price | BIGINT | Contains price of product |
| 8 | Sale\_price | BIGINT | Contains discount of product |
| 9 | description | VARCHAR(255) | Contains description of product |
| 10 | status | INT | Open sale status |
| 11 | view | INT | Contains the number of views of the product |
| 12 | created\_at | TIMESTAMP | Contains product creation date |
| 13 | modified\_at | TIMESTAMP | Last modified time |

#### 4.2.1.7 Table “Product\_Category”

*Table 4.7: Table “Product\_Category”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | product\_id | VARCHAR(255) | Product ID |
| 2 | category\_id | BIGINT | Category ID |

#### 4.2.1.8 Table “Product\_size”

*Table 4.8: Table “Product\_size”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | product\_id | VARCHAR(255) | Product ID |
| 2 | size | INT | Contains size of product |
| 3 | quantity | INT | Contains quantity of product |

#### 4.2.1.9 Table “Promotion”

*Table 4.9: Table “Promotion”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | BIGINT | Used to contain a unique identifier for each data and as a primary key |
| 2 | coupon\_code | VARCHAR(255) | Coupon code |
| 3 | created\_at | TIMESTAMP | Creation time |
|  | discount\_type | INT | Discount type:  1 - percent (%)  2 - at a fixed rate |
|  | discount\_value | BIGINT | Discount value |
|  | expired\_at | TIMESTAMP | Expiration time |
|  | is\_active | boolean | Activation status |
|  | is\_public | boolean | Public status |
|  | maximum\_discount\_value | BIGINT | Maximum discount value |
|  | name | VARCHAR(300) | Name of promotion |

#### 4.2.1.10 Table “Order”

*Table 4.10: Table “Order”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | BIGINT | Used to contain a unique identifier for each data and as a primary key |
| 2 | created\_at | TIMESTAMP | Creation time |
| 3 | modified\_at | TIMESTAMP | Last modified time |
| 4 | note | VARCHAR(255) | Notes |
| 5 | product\_price | BIGINT | Product price |
| 6 | promotion | JSON | Promotions used:  {  "coupon\_code": varchar "discount\_type": int  "discount\_value": int "maximum\_discount\_value": int  } |
| 7 | receiver\_address | VARCHAR(255) | Recipient address |
| 8 | receiver\_name | VARCHAR(255) | Recipient name |
| 9 | receiver\_phone | VARCHAR(255) | Recipient phone number |
| 10 | size | INT | Product size |
| 11 | status | INT | Order status:  1 - Waiting for delivery  2 - on delivery  3 - delivered  4 - returns  5 - cancellation of application |
| 12 | total\_price | BIGINT | Total order value |
| 13 | buyer | BIGINT | Buyer ID |
| 14 | created\_by | BIGINT | Order creator |
| 15 | modified\_by | BIGINT | Last modified person ID |
| 16 | product\_id | VARCHAR(255) | Product ID |

#### 4.2.1.11 Table “Comment”

*Table 4.11: Table “Promotion”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | BIGINT | Used to contain a unique identifier for each data and as a primary key |
| 2 | content | TEXT | Contains content of comment |
| 3 | user\_id | INT | Contains the binding key value of the “User” table |
| 4 | product\_id | VARCHAR(255) | Contains the binding key value of the “Product” table |
| 5 | post\_id | BIGINT | Contains the binding key value of the “Post” table |
| 6 | created\_at | TIMESTAMP | Creation time |

#### 4.2.1.12 Table “Statistic”

*Table 4.12: Table “Statistic”*

|  | Name | Datatypes | Describe |
| --- | --- | --- | --- |
| 1 | id | BIGINT | Used to contain a unique identifier for each data and as a primary key |
| 2 | sales | BIGINT | Contains sales |
| 3 | profit | BIGINT | Contains profit |
| 4 | quantity | INT | Contains quantity |
| 5 | created\_at | TIMESTAMP | Creation time |
| 6 | order\_id | BIGINT | Contains the binding key value of the “Order” table |

## Application construction

### 4.3.1 Libraries and tools used

| Purpose | Tool | URL address |
| --- | --- | --- |
| IDE programming | Intellij idea | https://www.jetbrains.com/idea/ |
| Database design | MySQL Workbench | https://www.mysql.com/products/workbench/ |
| JAVA development environment | JDK | https://www.oracle.com/java/technologies/javase-downloads.html |

### Illustrate the main functions

#### 4.3.2.1 Home page

The homepage is the first place a user reaches the site. So the homepage is made so that users can understand through the system of the website, what stands out so that users feel more excited during use.

A screenshot of a video game

Description automatically generated with medium confidence

*Figure 4.3: Home page*

#### 4.3.2.2 Account registration page

At the member registration page, customers can register for an account as a member of the website by entering correctly and fully all the requirements stated in the form such as: Email, password, name, address, …

A picture containing text, indoor, screenshot

Description automatically generated

*Figure 4.4: Account registration page*

#### 4.3.2.3 Account login page

After clicking the Login link in the menu bar, customers will be redirected to the login page for customers who have previously registered an account.

A pair of shoes

Description automatically generated with low confidence

*Figure 4.5: Account login page*

#### 4.3.2.4 The page displays user information

On this page, users can update their own information for accuracy, change passwords to secure accounts.

A computer screen shot

Description automatically generated with low confidence

*Figure 4.6: The page displays user information*

#### 4.3.2.5 The page displays order history

On this page, users can check order status, transaction history on the system

A screenshot of a computer

Description automatically generated

*Figure 4.7: The page displays order history*

#### 4.3.2.6 The page displays all product

This page will display all the available product on the system, so customers can choose the product that interest them.

A screenshot of a computer

Description automatically generated with low confidence

*Figure 4.8: The page displays all product*

#### 4.3.2.7 Product search page

When users are bothered by too many products or bothered because the products are arranged in an unordered manner, it is difficult for users to find their favorite products. They can search by typing keywords into the "Search" box and the website will search for products related to that keyword.

A screenshot of a computer

Description automatically generated with medium confidence

*Figure 4.9: Product search page*

#### 4.3.2.8 The page displays the product details

This page will display the details of the product so that customers can know if this product is meeting their goals so that they can buy the product. After you are satisfied with the product, please click the "Add to Cart" button to proceed with payment

Graphical user interface, website

Description automatically generated

*Figure 4.10: The page displays the product details*

#### 4.3.2.9 Payment

Once they have chosen the product that are right for them, they must confirm that they already own the product by clicking the "Buy product" button and choose payment methods.

A screenshot of a computer

Description automatically generated

*Figure 4.11: Payment*

#### 4.3.1.10 Admin Dashboard

Dashboard helps administrators know the general statistics, the latest changes and especially it makes it easier to manipulate.

Graphical user interface, application

Description automatically generated

*Figure 4.12: Admin Dashboard*

## Testing

This test is performed on the basis of deploying the system to a server and then accessing the equipment table and testing each part of the system. Then, sum it up with the following result:

### 4.4.1 About the interface

The system interface is not changed from the design when used on computers with many different screen resolutions.

### 4.4.2 About the function

The functions operate as designed, do not generate errors during use. Requests from the user are handled correctly.

### 4.4.3 About performance

The system has fast response speed, in this section has not been tested with a large number of accesses.

## Deployment

### 4.5.1 Install on website server

* Requires server configuration

The minimum configuration for deploying a web server is as follows:

| Ingredient | Minimum requirements | Note |
| --- | --- | --- |
| CPU | 1.5 GHz |  |
| RAM | 1.0 GB |  |
| Hard drive is empty | 10 GB |  |

The recommended configuration for deploying a web server is as follows:

| Ingredient | Minimum requirements | Note |
| --- | --- | --- |
| CPU | 3 GHz |  |
| RAM | 2 GB |  |
| Hard drive is empty | 50 GB |  |

# CHAPTER 5: CONCLUSION AND DEVELOPMENT DIRECTION

## 5.1 Result

Within the framework of the time to implement the project with the effort of myself and the guidance of the teacher, I have achieved the following results:

* For customers:
  + - Customer can search for products, register to buy products, make suggestions and view news.
    - Customers review their registered information, review information on orders, and update the information that has been registered for purchase.
* For administrator:
  + - Admin can: add, edit, delete, update and statistics products.
    - Manage customers and update customer information register and manage customer lists.
    - View reports on sales situation.
    - In terms of processing speed (search, lookup, statistics...) fast, the results search results are clearly listed, convenient for customers to use.
    - The system uses a Web interface, a common interface that commonly used network applications today. The interface design is simple, easy to understand and clear, so it is suitable for users.

## 5.2 Restrictions exist

In the process of surveying and selecting solutions and limited time, the study also has the following limitations:

* Not mentioning the issue of security and data safety.
* Some accounting operations such as: sales invoice management ... not yet rigid.

## 5.3 Development

To overcome the above limitations, I will try to develop the topic well in the future and add the following functions:

* Manage configuration, landing-page, more interface to fit the promotions.
* Create customer care function, online support to add credibility to the website.
* Secure product as well as websites to prevent hackers from attacking.
* Apply many payment methods such as payment by bank card.
* The program will support more programming languages.
* Perform non-installed functions such as: send automatic emails to
* customers, completing and supporting the credit card payment process

**References**

| [1] | Aziz Ali " HTML, CSS, Bootstrap, Javascript and jQuery," [Online]. Available: https://www.quora.com/What-is-HTML-CSS-and-JavaScript |
| --- | --- |
| [2] | JAVA Programming Language (Spring Framework), [Online]. Available: https://spring.io/. |