

**Website Phone Sales**

Double T

CE130438 - Tang Minh Tin

CE140079 - Bach Nguyen Phuc Thinh

Mentor: Luong Hoang Huong

Table of Contents

[I. Chapter 1: Introduction 2](#_Toc46520906)

[**1.** **Problem Definition** 2](#_Toc46520907)

[**2.** **Customer Requirement Specification** 4](#_Toc46520908)

[**a.** **System Requirements** 5](#_Toc46520909)

[**b.** **Required Software** 6](#_Toc46520910)

[II. Role, Schedule, Gantt diagram, Meeting and link Github 7](#_Toc46520911)

[**1.** **Role & Schedule** 7](#_Toc46520912)

[**2.** **Gantt diagram** 9](#_Toc46520913)

[**3.** **Meeting schedule** 10](#_Toc46520914)

[**4.** **Link Github** 11](#_Toc46520915)

[III. Chapter 2: Theory 12](#_Toc46520916)

[**1.** **Introduction to JSP** 12](#_Toc46520917)

[**a.** **What is JSP?** 12](#_Toc46520918)

[**b.** **Advantages of JSP over Servlet** 12](#_Toc46520919)

[**c.** **The Lifecycle of a JSP Page** 13](#_Toc46520920)

[**2.** **Introduction to MVC in JSP** 15](#_Toc46520921)

[**a.** **What is MVC?** 15](#_Toc46520922)

[**b.** **The advantages of MVC** 17](#_Toc46520923)

[IV. Chapter 3: Architecture System 18](#_Toc46520924)

[**1.** **Client-Server Architectures** 18](#_Toc46520925)

[**2.** **Use-case Diagram** 20](#_Toc46520926)

[**3.** **Class Diagram** 21](#_Toc46520927)

[**4.** **Entity relationship diagram** 22](#_Toc46520928)

[**5.** **Activity Diagram** 23](#_Toc46520929)

[**6.** **DFD** 26](#_Toc46520930)

[**a.** **Data flow diagram symbol** 26](#_Toc46520931)

[**b.** **Contextual Level 0** 27](#_Toc46520932)

[**c.** **Level 1** 28](#_Toc46520933)

[**d.** **Level 2** 30](#_Toc46520934)

[**e.** **Level 3** 32](#_Toc46520935)

[**f.** **Level 4** 34](#_Toc46520936)

[V. Chapter 4: Functions, User Interfaces and Flow chart 35](#_Toc46520937)

[**1.** **Functions** 35](#_Toc46520938)

[**a.** **User** 35](#_Toc46520939)

[**b.** **Admin** 35](#_Toc46520940)

[**2.** **User Interfaces** 36](#_Toc46520941)

[**3.** **Flow chart** 42](#_Toc46520942)

[VI. Conclusion 44](#_Toc46520943)

1. **Chapter 1: Introduction**
2. **Problem Definition**

With the rapid development of information technology today, these small and discrete mobile phone shops cannot help being affected by this development. If the shops continue to maintain the traditional form of business, it is force majeure in the present time.

The competitiveness of small shops is force majeure, which leads to inadequate ability to compete with rivals of mobile phone stores today. This situation leads to the risk of inventory, there are many old and outdated products and do not keep up with the trends of new technology and products on a daily basis.

In the traditional business form, we have a lot of difficulties in managing the items, categories, models, components, accessories, etc. Products that our customers need, we are not enough to sell, or import many products. Products that customers are not paying attention to.

Difficulties in choosing products and models. Customers can not be configured with this phone, how beautiful taking pictures, how many hours of battery life, how the chip, what product is suitable for myself .

The issue of where the seller is also a factor that pleases customers. Customers want the stores that have the products they need to buy nearest to experience, shop ... From the places where customers buy more or less is the best place to import and export the best goods for the store.

In order to solve the above problems, Double T decided to build Website Phones Sales with the goal of bringing the store's products to the national business market, solving customer needs. Website for users to easily access, search, buy and sell quickly and conveniently.

1. **Customer Requirement Specification**

A mobile phone shop in Can Tho wants to sell its products in the form of online business. They want to build a system that allows the deployment of product information on the website system with the following requirements and functions:

1. **Interface requirements**

* Nice user interface and user friendly.
* Actions on the Web should be familiar, easy to use with users, allowing manipulation with both mouse and keyboard.
* There are illustrative images, clear and detailed content, and a well-organized content structure.

1. **Functional requirements**

* Can add, edit, delete, search products quickly and easily.
* Have a function of order management, user management.
* Registration, login, and logout functions.
* Manipulate the cart, issue invoices, charge.
* Assign users, report statistics by month, by year.
* High security, maintainability, and easy system development

1. **Customer's requirements**

* Easy to find, fast, accurate search for equipment to buy.
* Can check the invoice easily, see the products in the cart.
* Payment is quick, easy and confidential

1. **Hardware and Software Requirement**
2. **System Requirements**

|  |
| --- |
| Minimum Hardware Configurations |
| Microsoft Windows Vista SP1/Windows 7 Professional:  Processor: 800MHz Intel Pentium III or equivalent  Memory: 512 MB  Disk space: 750 MB of free disk space  Ubuntu 9.10:  Processor: 800MHz Intel Pentium III or equivalent  Memory: 512 MB  Disk space: 650 MB of free disk space  Macintosh OS X 10.7 Intel:  Processor: Dual-Core Intel  Memory: 2 GB  Disk space: 650 MB of free disk space |

|  |
| --- |
| Recommended Hardware Configurations |
| Microsoft Windows 7 Professional/Windows 8/Windows 8.2:  Processor: Intel Core i5 or equivalent  Memory: 2 GB (32-bit), 4 GB (64-bit)  Disk space: 1.5 GB of free disk space  Ubuntu 15.04:  Processor: Intel Core i5 or equivalent  Memory: 2 GB (32-bit), 4 GB (64-bit)  Disk space: 1.5 GB of free disk space  OS X 10.10 Intel:  Processor: Dual-Core Intel  Memory: 4 GB  Disk space: 1.5 GB of free disk space |

1. **Required Software**

* NetBeans IDE runs on the Java SE Development Kit (JDK) which consists of the Java Runtime Environment and developer tools for compiling, debugging, and running applications written in the Java language.
* The tested JDK for this release is JDK 8u101 for Windows, Linux, and OS X. The 8.2 version of the IDE cannot be installed or run on the JDK older than JDK 8.
* Note:
* The PHP and C/C++ NetBeans bundles only require the Java Runtime Environment (JRE) 8 to be installed and run.
* Java features in the IDE and JavaFX 8 features require JDK 8.
* Download Tomcat for webserver, XAMPP for database.

1. **Role, Schedule, Gantt diagram, Meeting and link Github**
2. **Role & Schedule**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Description | Date made | Finish day | Number of hours completed | Person in charge | Status | Note |
| 1 | Choose a topic & design ideas | 26/05/2020 | 27/05/2020 | 3 hours | Tín, Thịnh | Done |  |
| 2 | Planning, meeting time group | 30/05/2020 | 03/05/2020 | 4 hours | Thịnh, Tín | Done |  |
| 3 | Draw a GANTT diagram | 06/06/2020 | 09/06/2020 | 5 hours | Thịnh, Tín | Done |  |
| 4 | Analysis, functional design | 11/06/2020 | 13/062020 | 8 hours | Tín, Thịnh | Done |  |
| 5 | Create a database diagram, create a database | 15/06/2020 | 18/06/2020 | 4 hours | Tín, Thịnh | Done |  |
| 6 | Analysis of front-end design, back-end, user / system requirement | 19/06/2020 | 21/06/2020 | 6 hours | Tín, Thịnh | Done |  |
| 7 | Complete document phase 1 | 23/06/2020 | 25/06/2020 | 4 hours | Tín, Thịnh | Done |  |
| 8 | Create project, add DAO classes | 26/06/2020 | 28/06/2020 | 5 hours | Tín, Thịnh | Done |  |
| 9 | Front-end interface code, testing, fixing bugs | 29/06/2020 | 03/07/2020 | 6 hours | Thịnh | Done |  |
| 10 | Simple module code | 05/07/2020 | 07/07/2020 | 9 hours | Thịnh, Tín | Done |  |
| 11 | Front-end testing, functions and simple module | 08/07/2020 | 10/07/2020 | 4 hours | Thịnh, Tín | Done |  |
| 12 | Back-end interface code | 11/07/2020 | 13/07/2020 | 5 hours | Thịnh, Tín | Done |  |
| 13 | Code for the main module, writing document phase 2 | 14/07/2020 | 17/07/2020 | 10 hours | Tín, Thịnh | Done |  |
| 14 | Back-end testing, main functions and modules | 18/07/2020 | 20/07/2020 | 5 hours | Thinh, Tín | Done |  |
| 15 | Write the final stage document | 21/07/2020 | 22/07/2020 | 4 hours | Tín, Thịnh | Done |  |
| 16 | General testing website, putting website on server, finishing document | 22/07/2020 | 24/07/2020 | 6 hours | Tín, Thịnh | Done |  |
| 17 | Presentation | 27/07/2020 | 27/07/2020 | 1 hours | Tín, Thịnh | Done |  |

Figure 1: Role & Schedule

1. **Gantt diagram**

Figure 2: Gantt diagram

1. **Meeting schedule**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Time | Participants | Meeting content | Note |
| 1 | 16/06/2020 – 3 hour | Thịnh, Tín | Prepare doccument phase 1 | Meeting online(link: <https://meet.google.com/thh-xhzh-znj>) |
| 2 | 20/06/2020 – 3 hour | Thịnh, Tín | Progress report for phase 1 | Meeting offline |
| 3 | 26/06/2020 – 3 hour | Thịnh, Tín | Edit doccument phase 2 | Meeting online(link: <https://meet.google.com/thh-xhzh-znj>) |
| 4 | 10/07/2020 – 3 hour | Thịnh, Tín | Progress report for phase 2 | Meeting offline  Address(library FU) |
| 5 | 20/07/2020 – 3 hour | Thịnh, Tín | Edit doccument phase 3 | Meeting offline  Address(library FU) |
| 6 | 26/07/2020 – 3 hour | Thịnh, Tín | Progress report for phase final | Meeting offline  Address(library FU) |

Figure 3: Meeting schedule schedule

1. **Link Github**

Link for project: <https://github.com/tangminhtin/PRJ321_SE1403_DoubleT_WebsitePhoneSales>

1. **Chapter 2: Theory**
2. **Introduction to JSP**
3. **What is JSP?**

* JSP technology is used to create web application just like Servlet technology. It can be thought of as an extension to Servlet because it provides more functionality than servlet such as expression language, JSTL, etc.
* A JSP page consists of HTML tags and JSP tags. The JSP pages are easier to maintain than Servlet because we can separate designing and development. It provides some additional features such as Expression Language, Custom Tags, etc.

1. **Advantages of JSP over Servlet**

* Extension to Servlet: JSP technology is the extension to Servlet technology. We can use all the features of the Servlet in JSP. In addition to, we can use implicit objects, predefined tags, expression language and Custom tags in JSP, that makes JSP development easy.
* Easy to maintain: JSP can be easily managed because we can easily separate our business logic with presentation logic. In Servlet technology, we mix our business logic with the presentation logic.
* Fast Development: No need to recompile and redeploy. If JSP page is modified, we don't need to recompile and redeploy the project. The Servlet code needs to be updated and recompiled if we have to change the look and feel of the application.
* Less code than Servlet: In JSP, we can use many tags such as action tags, JSTL, custom tags, etc. that reduces the code. Moreover, we can use EL, implicit objects, etc.

1. **The Lifecycle of a JSP Page**

* Translation of JSP Page
* Compilation of JSP Page
* Classloading (the classloader loads class file)
* Instantiation (Object of the Generated Servlet is created).
* Initialization (the container invokes jspInit() method).
* Request processing (the container invokes \_jspService() method).
* Destroy (the container invokes jspDestroy() method).

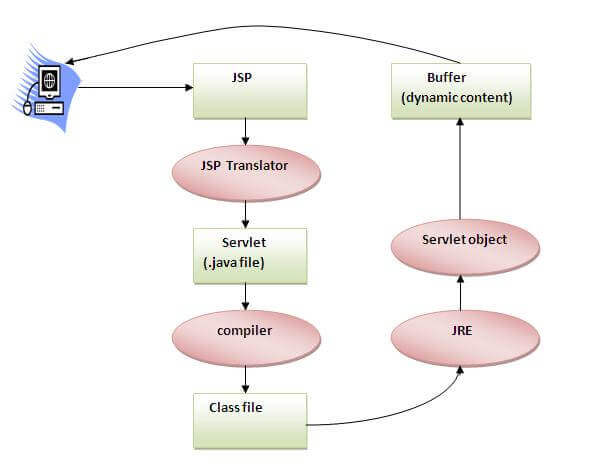


Figure 4: Introduction to JSP

* As depicted in the above diagram, JSP page is translated into Servlet by the help of JSP translator. The JSP translator is a part of the web server which is responsible for translating the JSP page into Servlet. After that, Servlet page is compiled by the compiler and gets converted into the class file. Moreover, all the processes that happen in Servlet are performed on JSP later like initialization, committing response to the browser and destroy.

1. **Introduction to MVC in JSP**
2. **What is MVC?**

* MVC is an architecture that separates business logic, presentation and data. In MVC, M stands for Model, V stands for View, C stands for controller.
* MVC is a systematic way to use the application where the flow starts from the view layer, where the request is raised and processed in controller layer and sent to model layer to insert data and get back the success or failure message.

**Model Layer:**

* This is the data layer which consists of the business logic of the system.
* It consists of all the data of the application
* It also represents the state of the application.
* It consists of classes which have the connection to the database.
* The controller connects with model and fetches the data and sends to the view layer.
* The model connects with the database as well and stores the data into a database which is connected to it.

**View Layer:**

* This is a presentation layer.
* It consists of HTML, JSP, etc. into it.
* It normally presents the UI of the application.
* It is used to display the data which is fetched from the controller which in turn fetching data from model layer classes.
* This view layer shows the data on UI of the application.

**Controller Layer:**

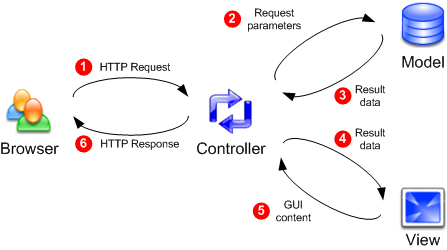
* It acts as an interface between View and Model.
* It intercepts all the requests which are coming from the view layer.
* It receives the requests from the view layer and processes the requests and does the necessary validation for the request.
* This requests is further sent to model layer for data processing, and once the request is processed, it sends back to the controller with required information and displayed accordingly by the view.

Figure 5: MVC Architecture

1. **The advantages of MVC**

* Easy to maintain
* Easy to extend
* Easy to test
* Navigation control is centralized

1. **Chapter 3: Architecture System**
2. **Client-Server Architectures**

A Client-Server Architecture consists of two types of components: clients and servers. A server component perpetually listens for requests from client components. When a request is received, the server processes the request, and then sends a response back to the client. Servers may be further classified as stateless or stateful. Clients of a stateful server may make composite requests that consist of multiple atomic requests. This enables more conversational or transactional interactions between client and server. To accomplish this, a stateful server keeps a record of the requests from each current client. This record is called a session.

In order to simultaneously process requests from multiple clients, a server often uses the Master-Slave Pattern. In this case, the Master perpetually listens for client requests. When a request is received, the master creates a slave to processes the request and then resumes listening. Meanwhile, the slave performs all subsequent communication with the client.

Internally, the client component may consist of a ClientUI that forwards user requests to a controller component. The controller component forwards the request across a process or machine boundary to a RequestListener inside the server. The listener, which acts like a master, creates a RequestHandler slave and forwards the request to it:

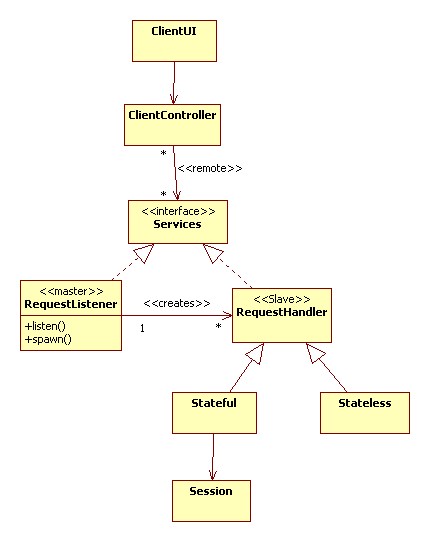


Figure 6: Client-Server Architectures

1. **Use-case Diagram**

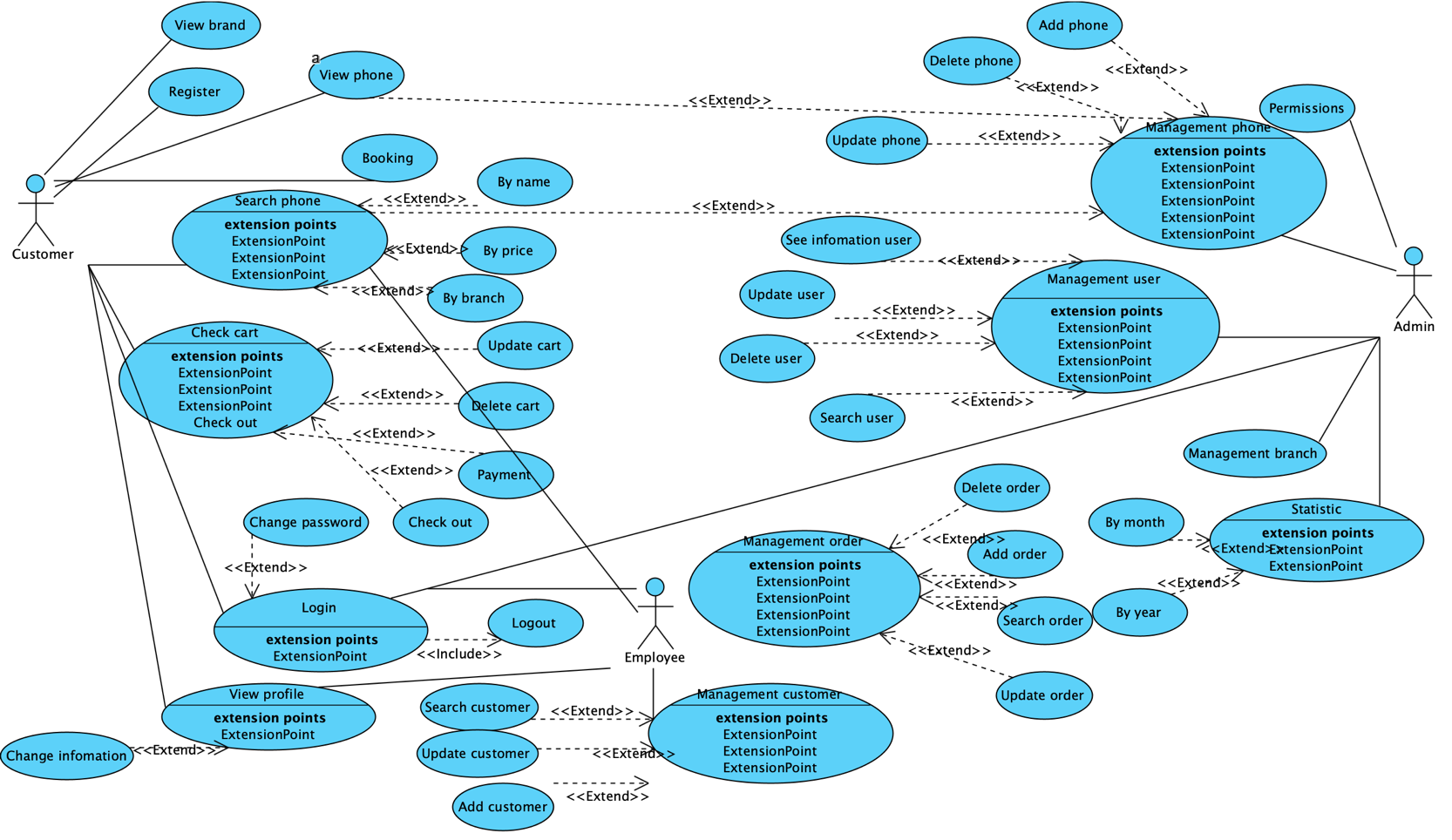


Figure 7: Use-case website phone sales

1. **Class Diagram**

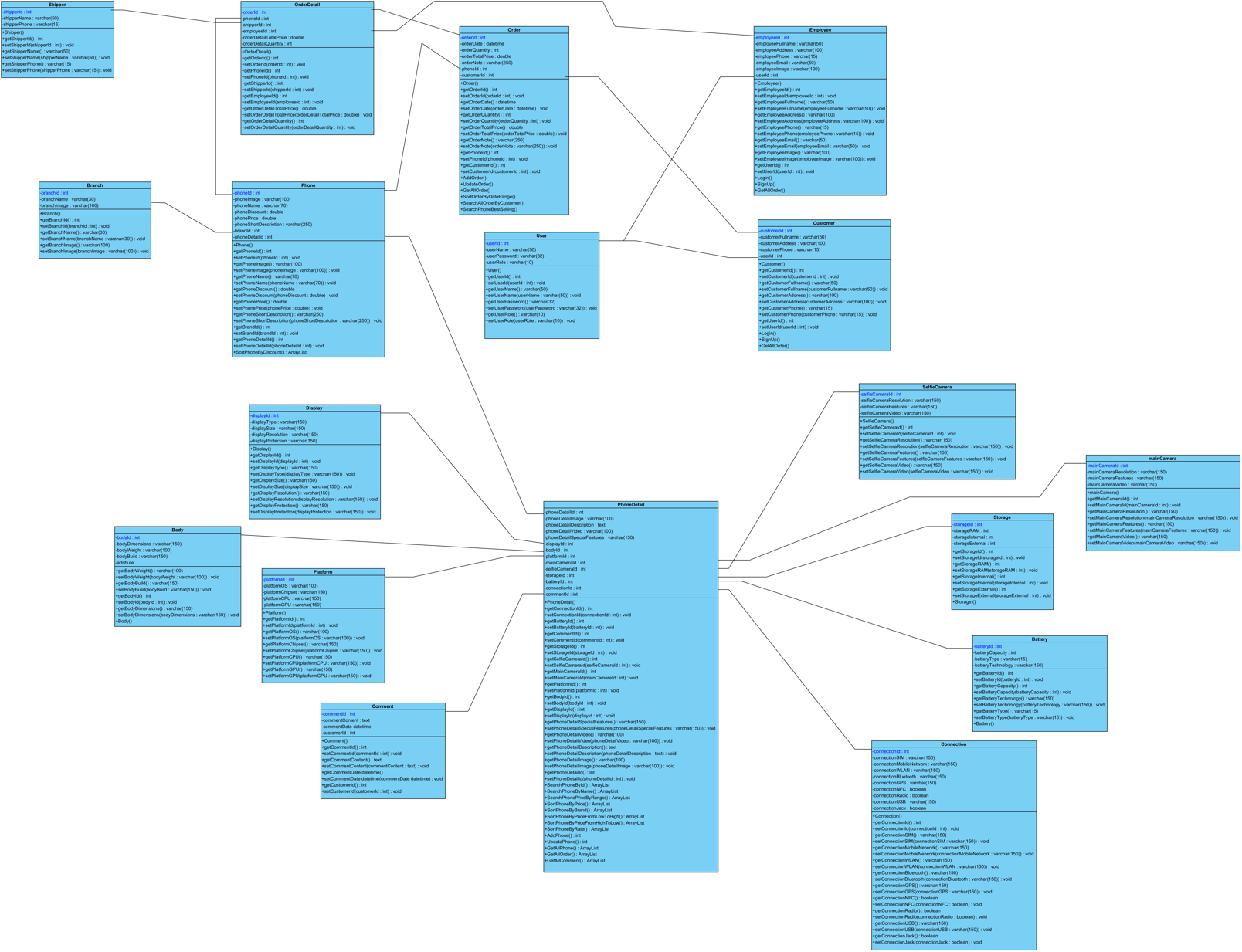


Figure 8: Class Diagram

1. **Entity relationship diagram**

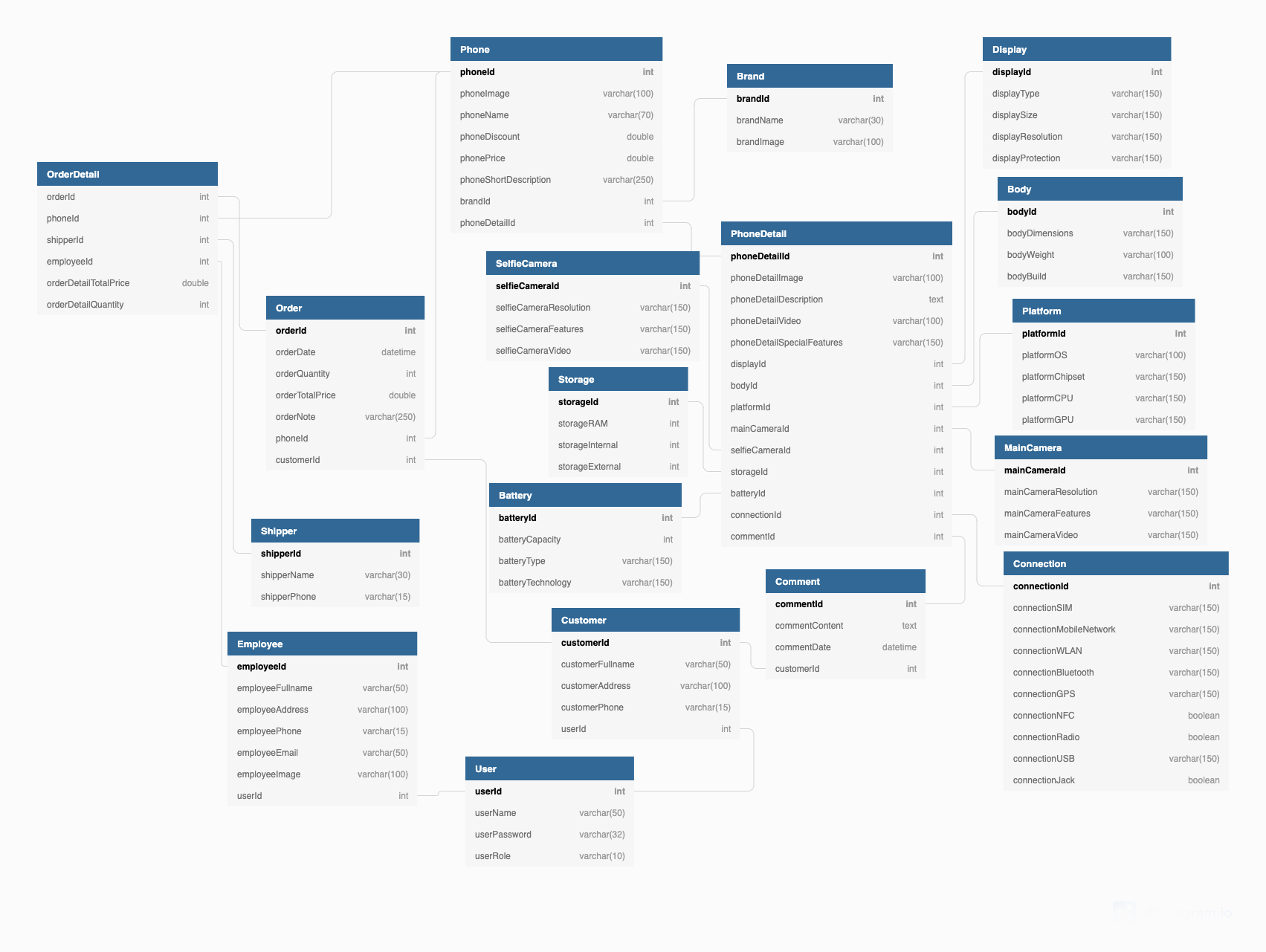


Figure 9: Database diagram of website phone sales

1. **Activity Diagram**

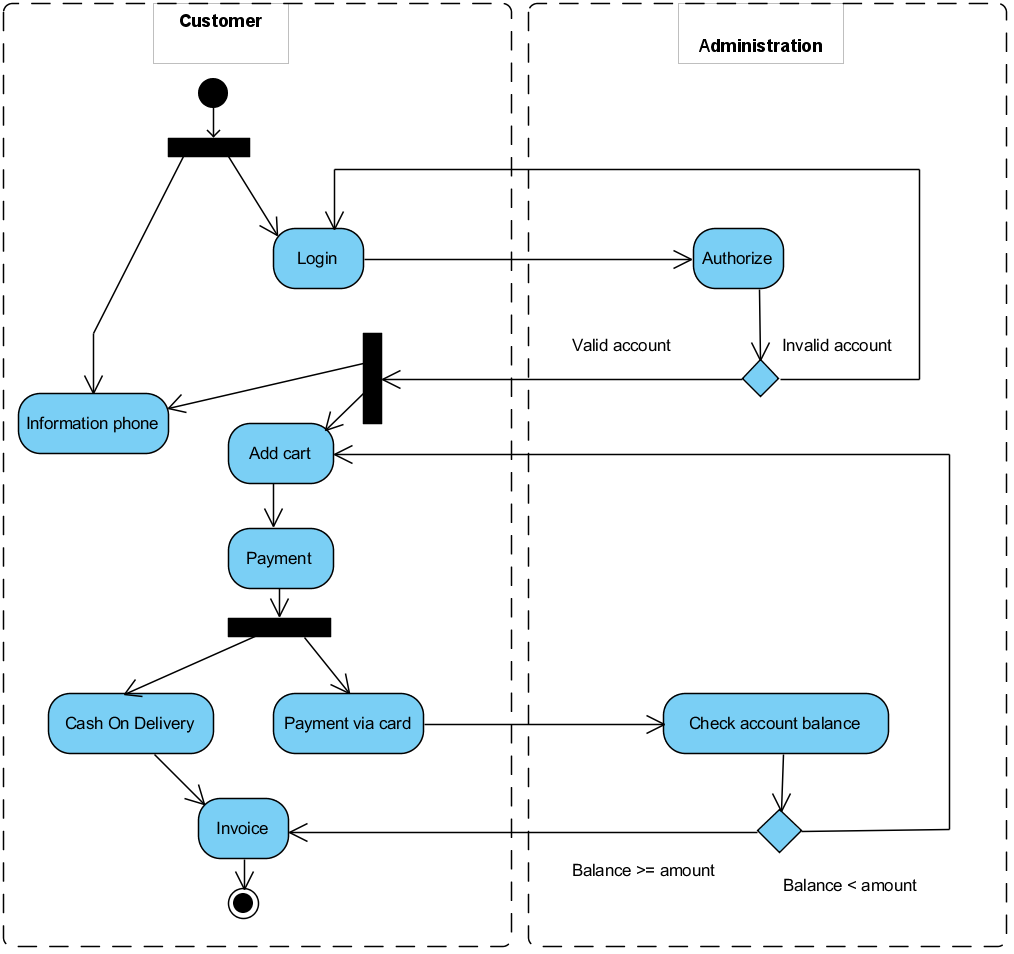


Figure 10: Activity diagram of Customer

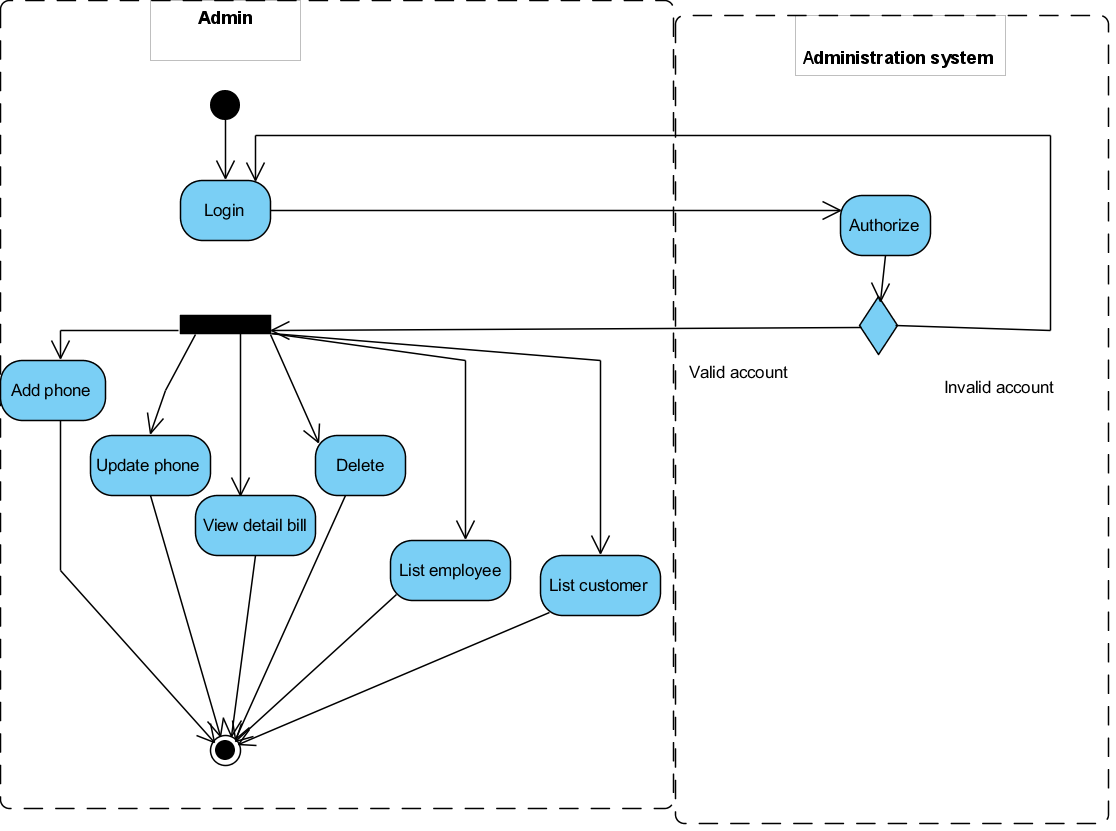


Figure 11: Activity diagram of Admin

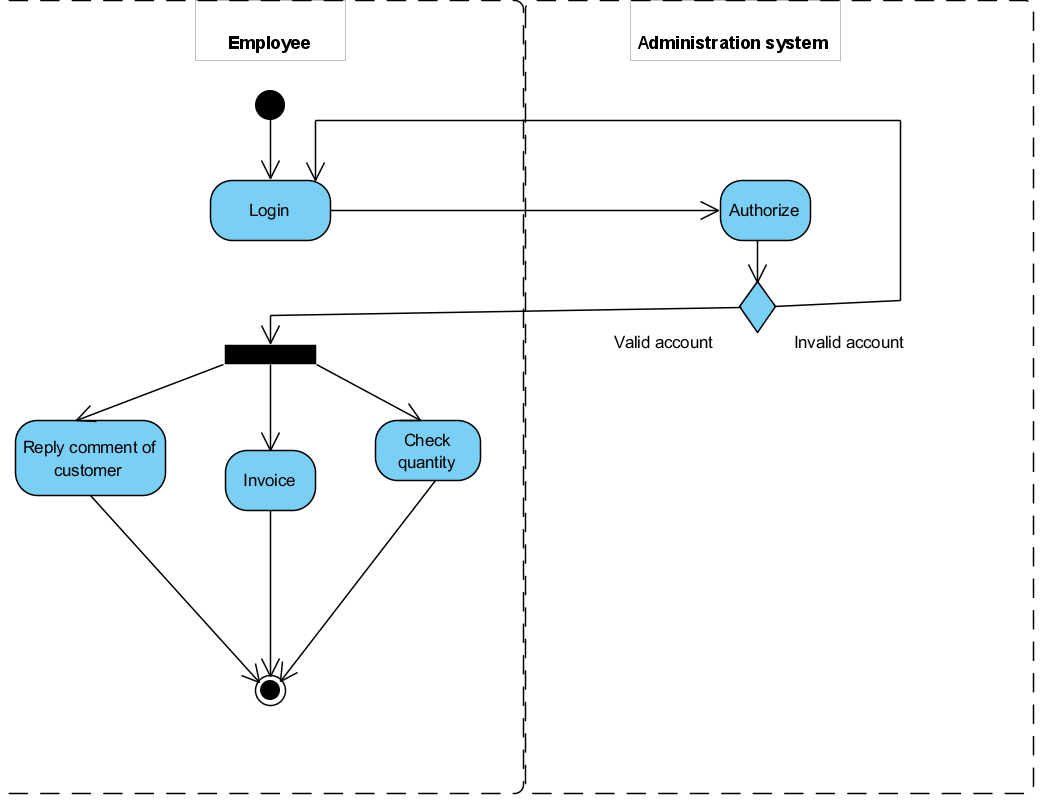


Figure 12: Activity diagram of Employee

1. **DFD**
2. **Data flow diagram symbol**

|  |  |
| --- | --- |
| Symbol | Description |
|  | **Data Flow:** Data flow are pipelines through the packets of information flow. |
|  | **Process:** A Process or task performed by the system. |
|  | **Entity:** Entity are object of the system. A source or destination data of a system. |
|  | **Data Store:** A place where data to be stored. |

Figure 13: Data flow diagram symbol

1. **Contextual Level 0**

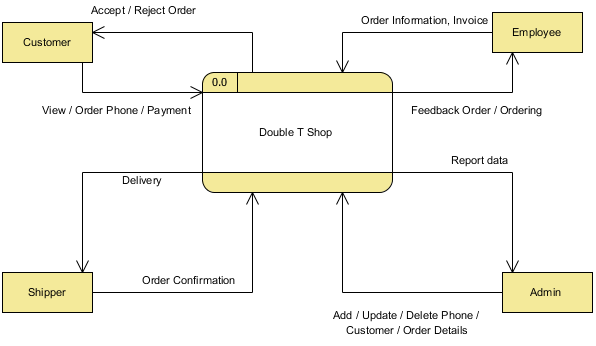


Figure 14: Contextual Level 0

1. **Level 1**

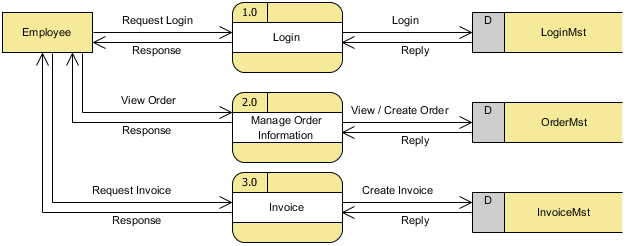
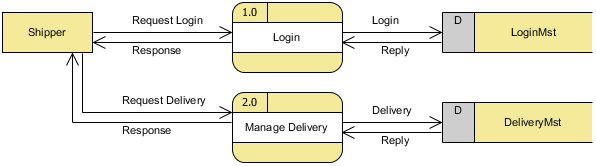
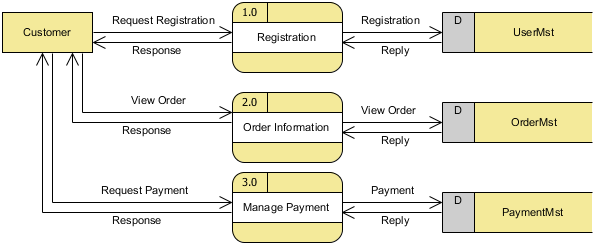
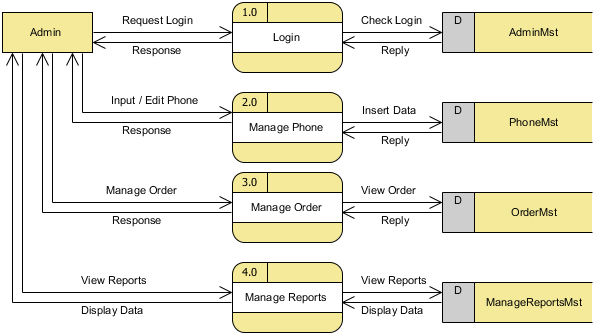


Figure 15: Admin Level 1

Figure 16: Customer Level 1

Figure 18: Shipper Level 1

Figure 17: Employee level 1

1. **Level 2**

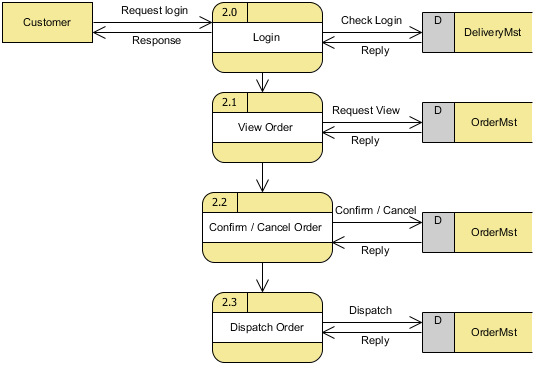
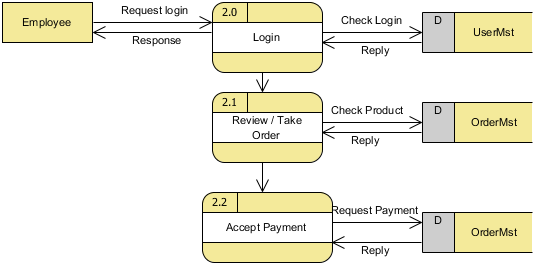


Figure 20: Employee Level 2

Figure 19: Customer level 2

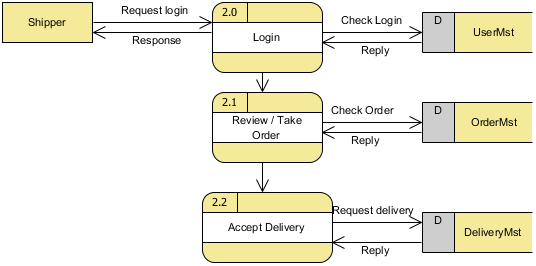
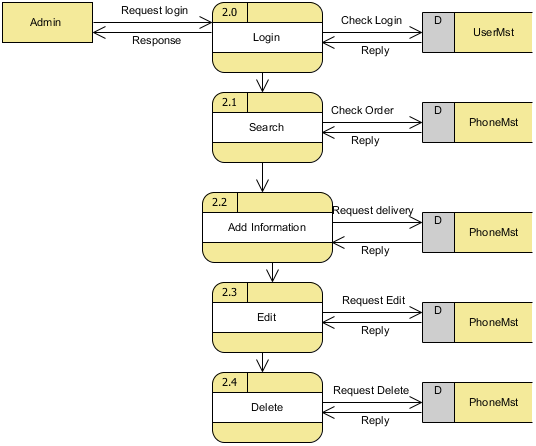


Figure 22: Admin Level 2

Figure 21: Shipper Level 2

1. **Level 3**

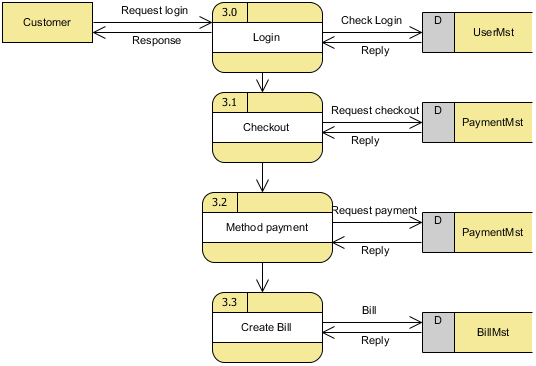
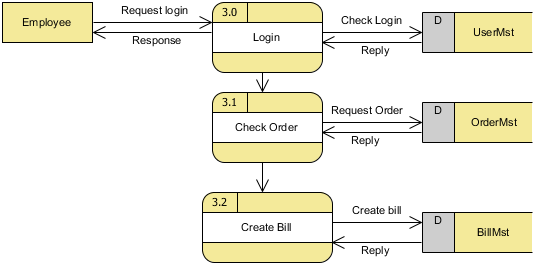


Figure 24: Employee Level 3

Figure 23: Customer Level 3

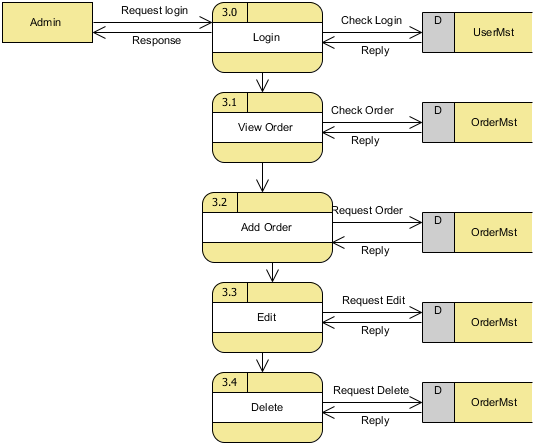


Figure 25: Admin Level 3

1. **Level 4**

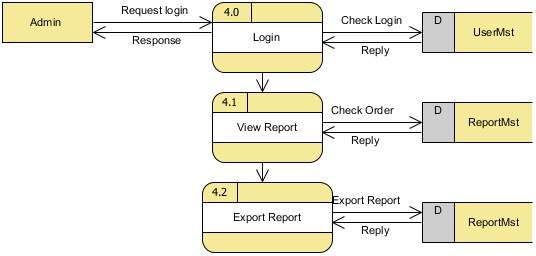


Figure 26: Admin Level 4

1. **Chapter 4: Functions, User Interfaces and Flow chart**
2. **Functions**
3. **User**

* Login
* Logout
* Sign up
* Reset password with email
* View product
* Add product to cart
* Delete product in cart
* Checkout
* Receive bill by email
* Show phone with the same brand
* Search phone by name
* Sort phone with increase price
* Sort phone with decrease price

1. **Admin**

* Login
* Logout
* Add phone
* Update phone
* Delete phone
* Search phone
* View order
* Manage User

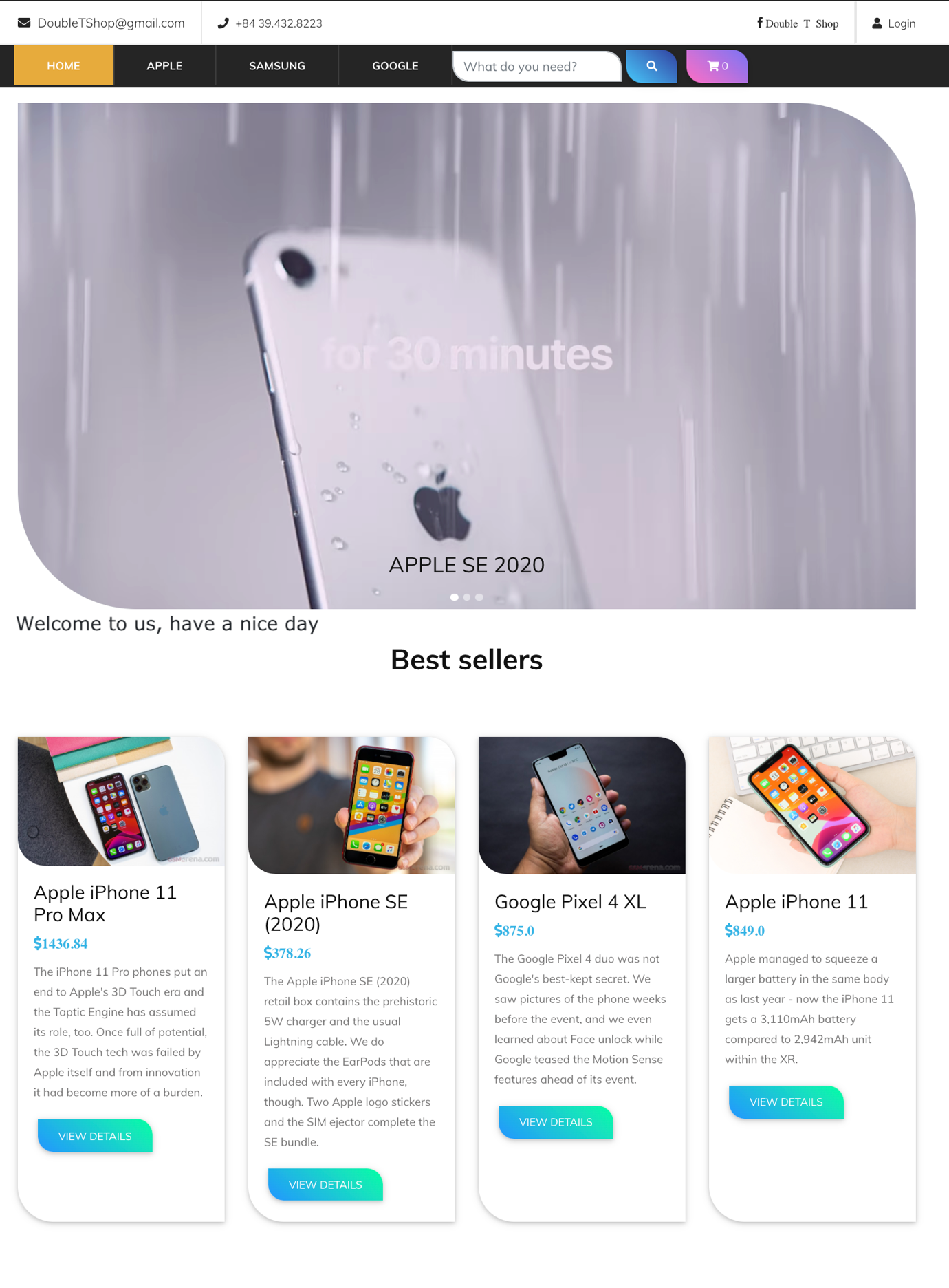
1. **User Interfaces**

Figure 4: Home Page

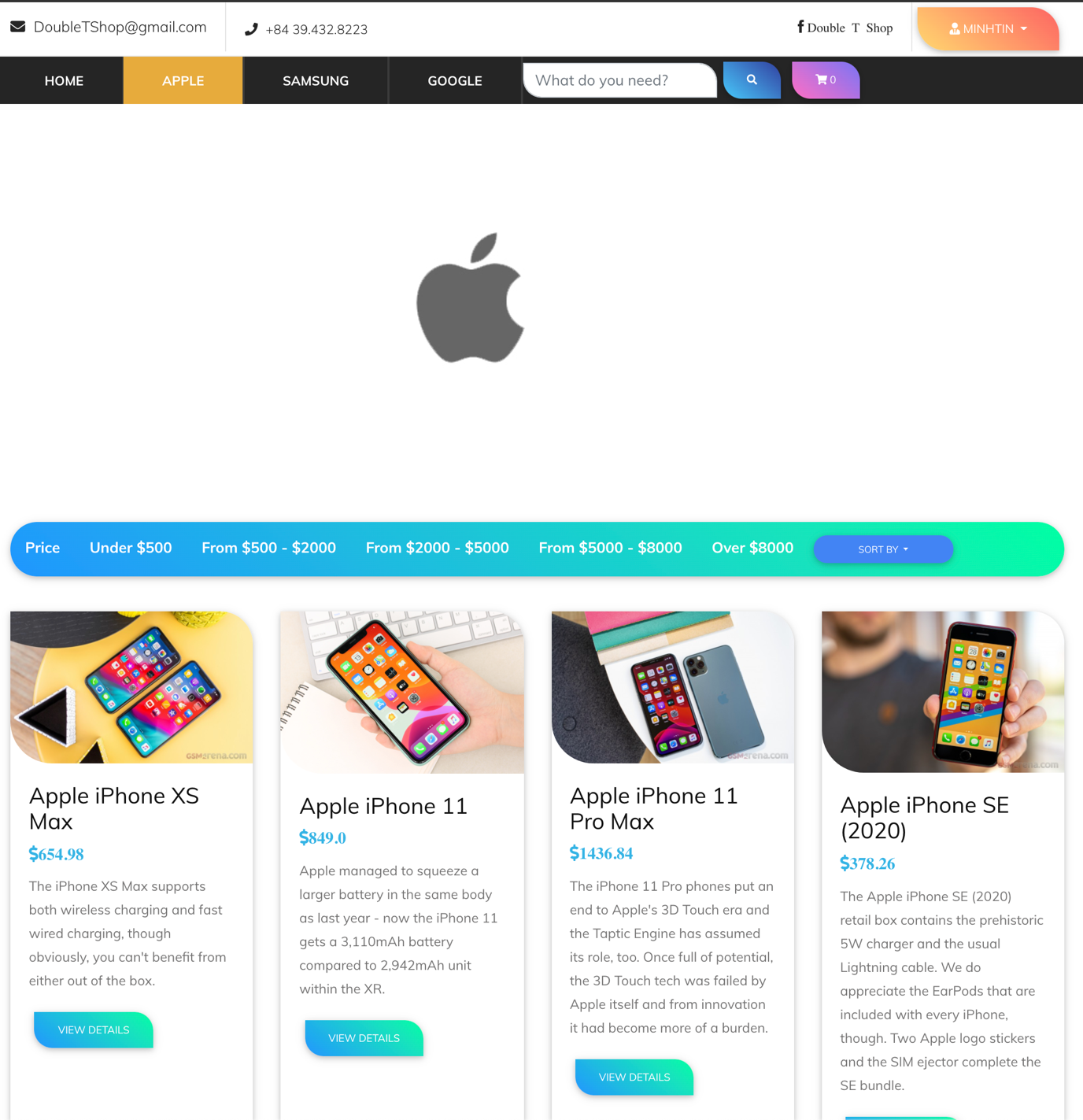


Figure 5: Apple Page

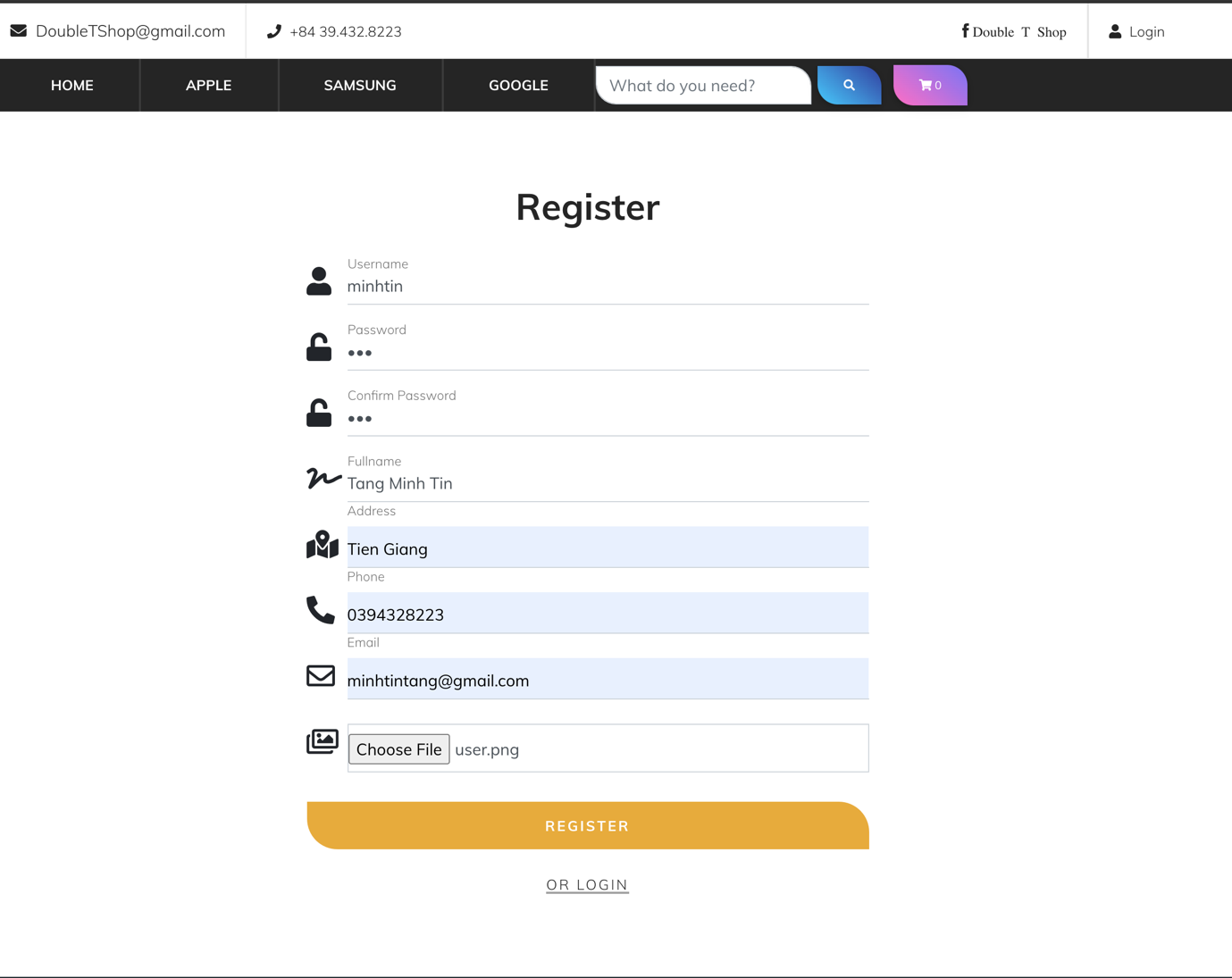
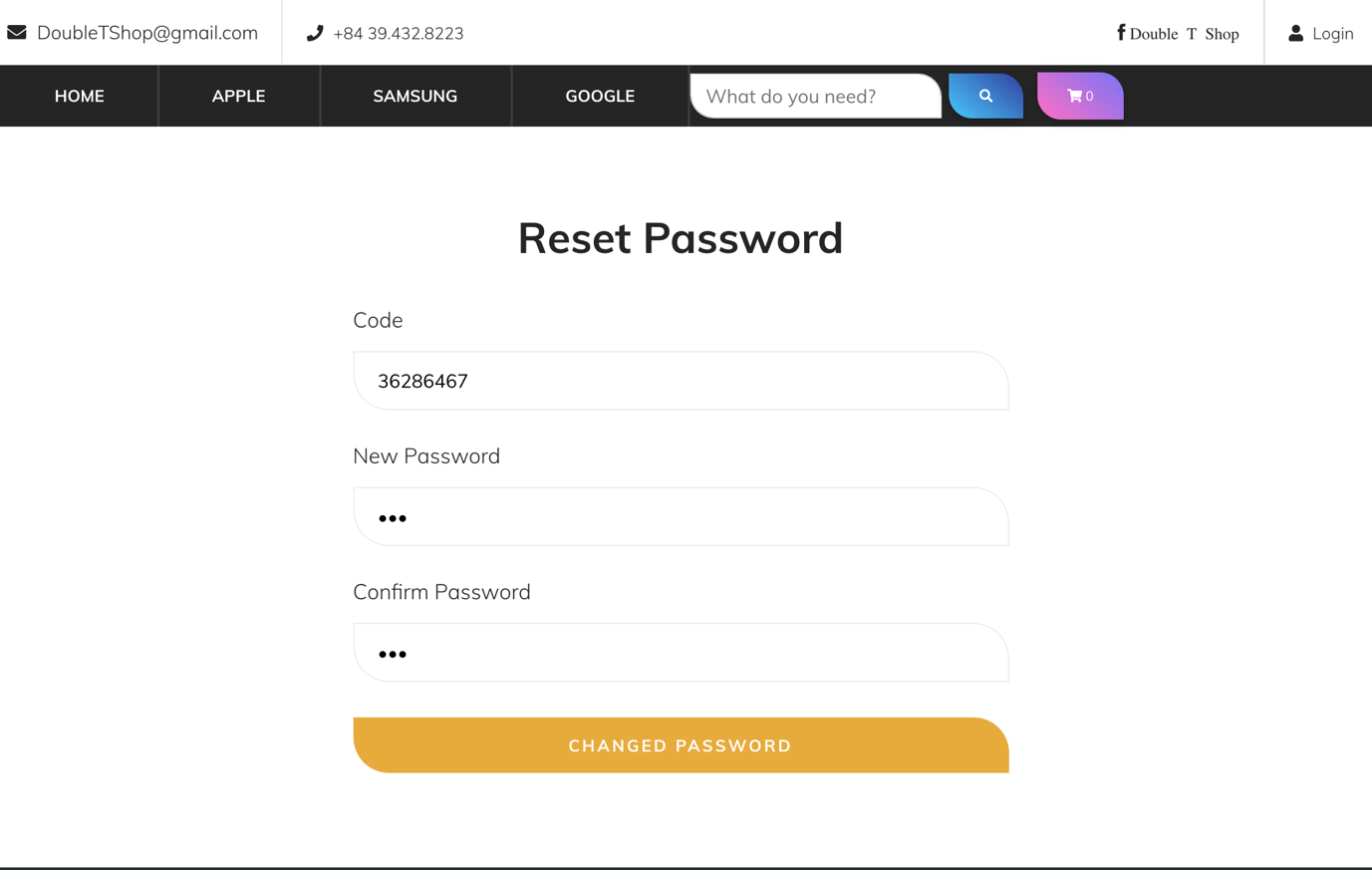


Figure 6: Reset password by code from email

Figure 7: Register Page

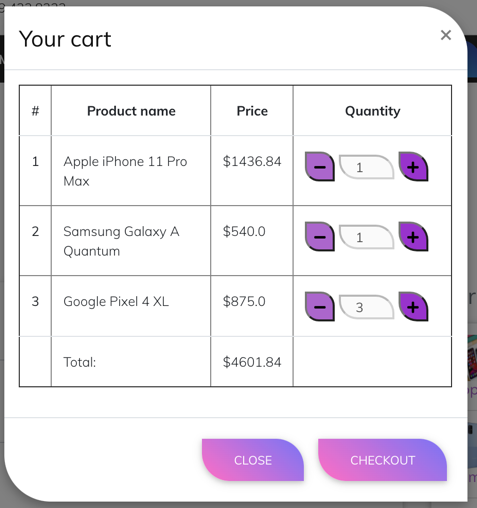
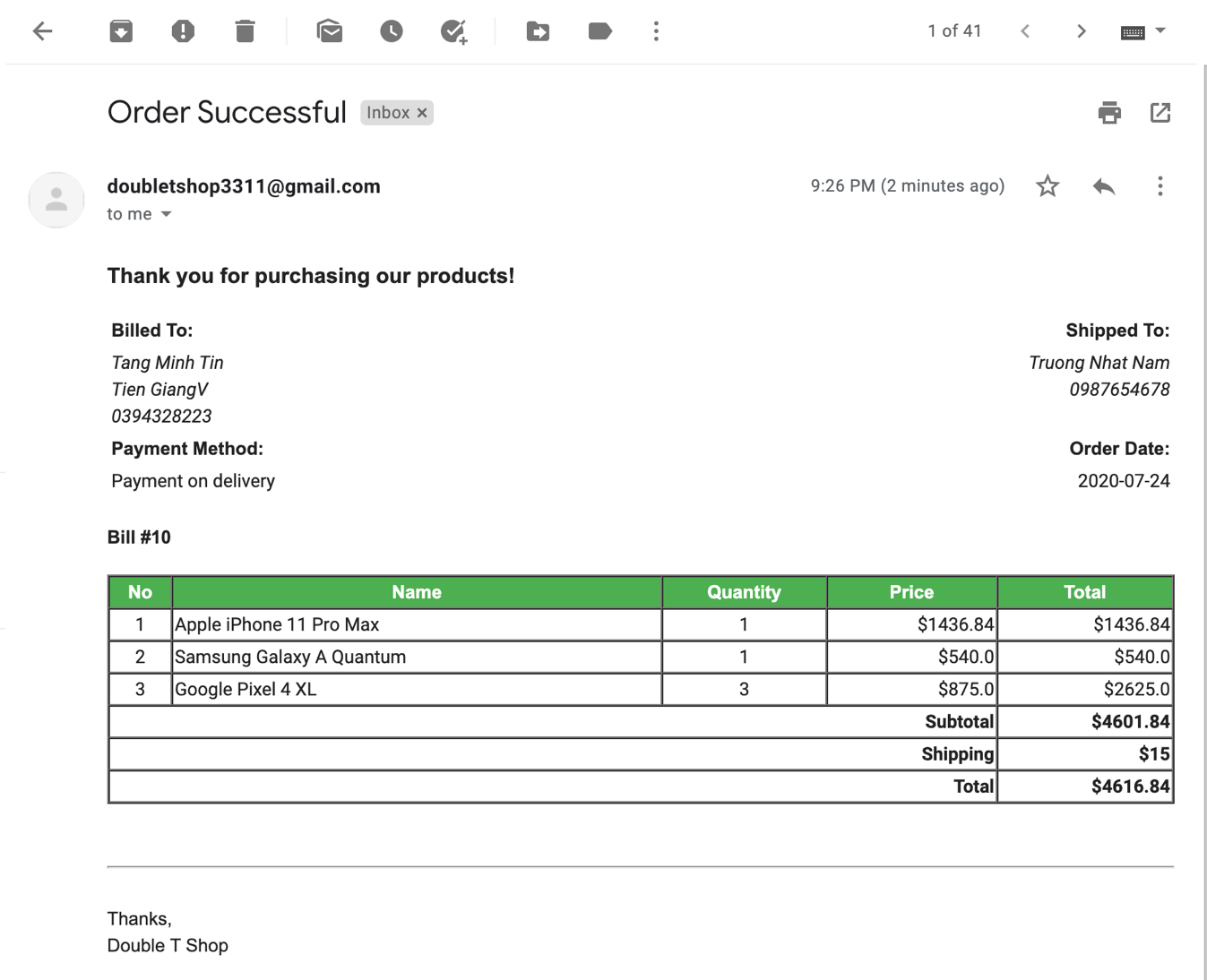


Figure 8: Bill of products

Figure 9: Checkout Information

Figure 10: Cart

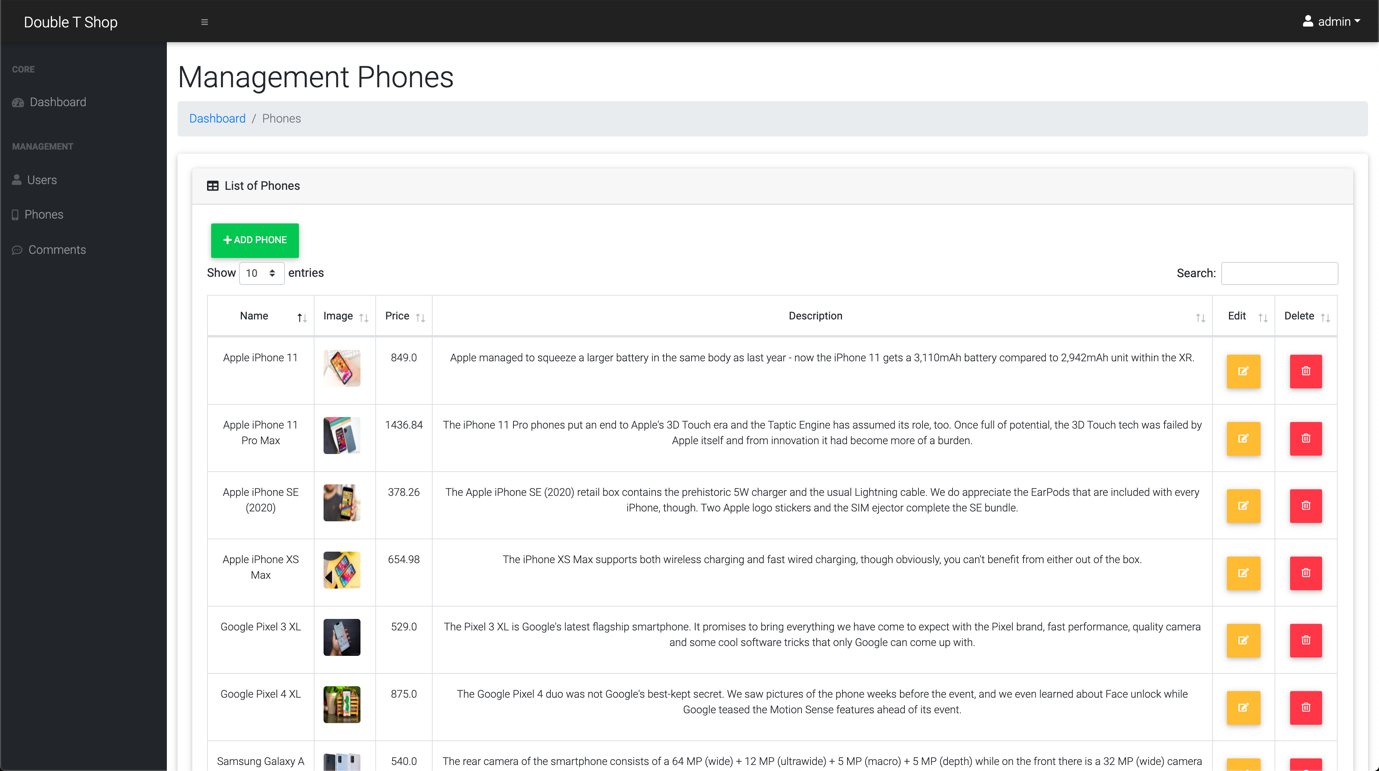


Figure 11: Dashboard

Figure 12: Manage Phones

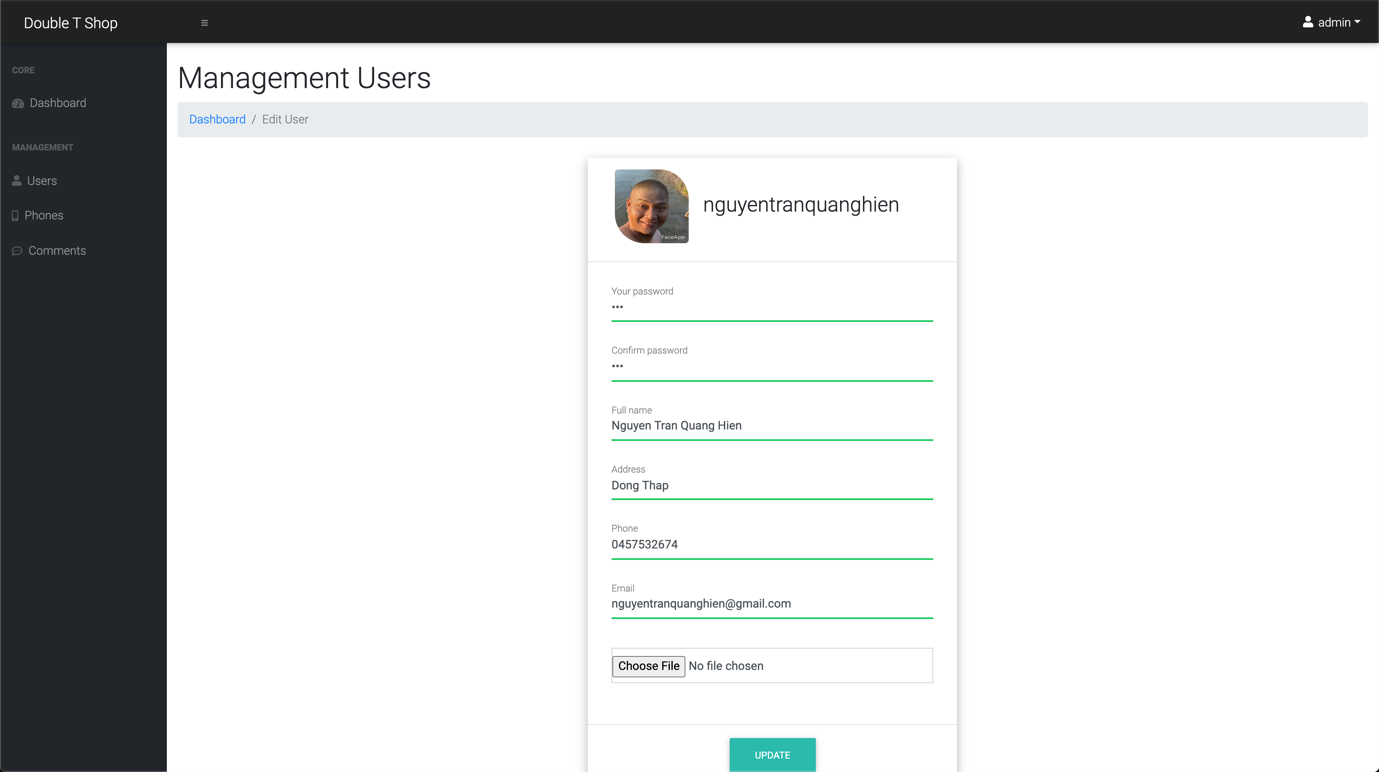
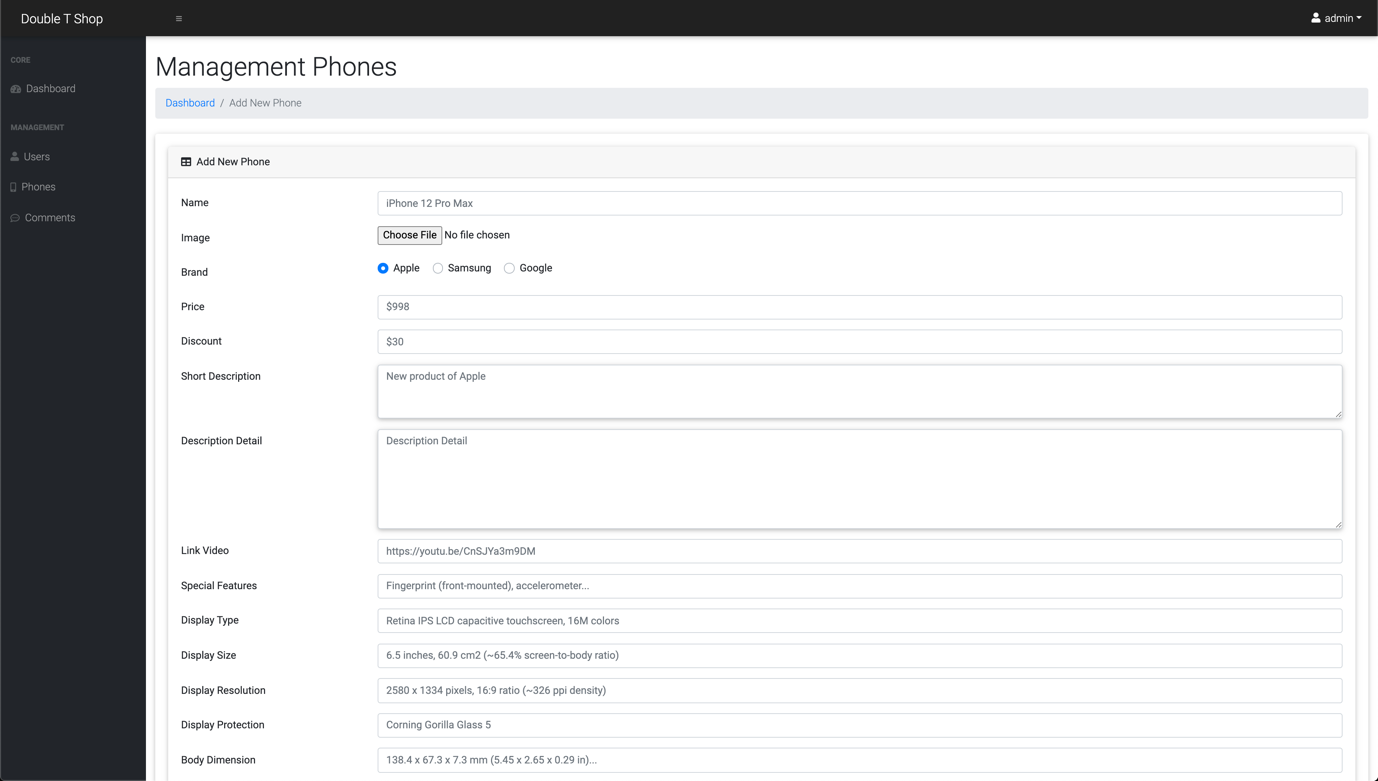


Figure 13: Insert New Phone

Figure 14: Update User

1. **Flow chart**

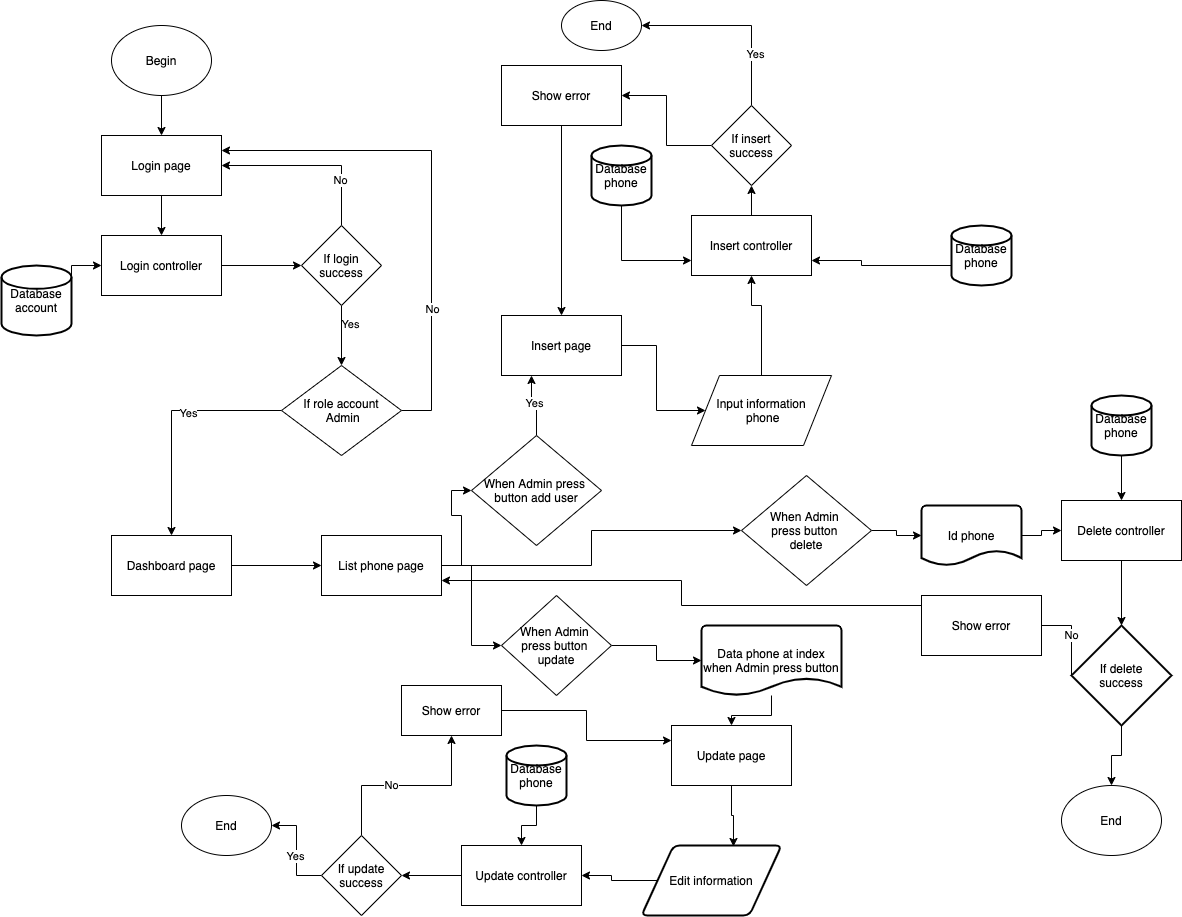


Figure 15: Insert new phone

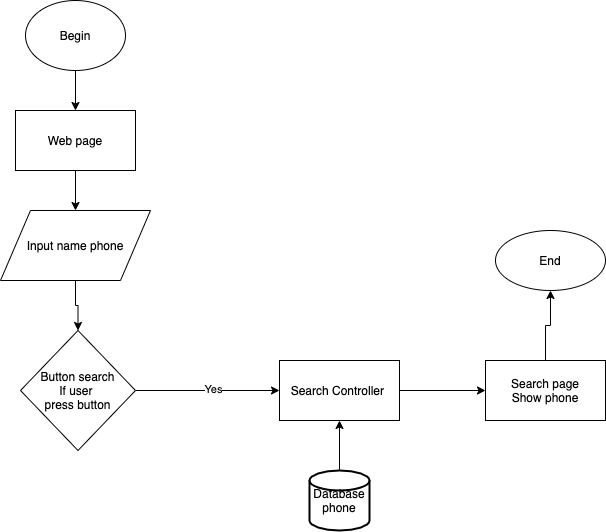
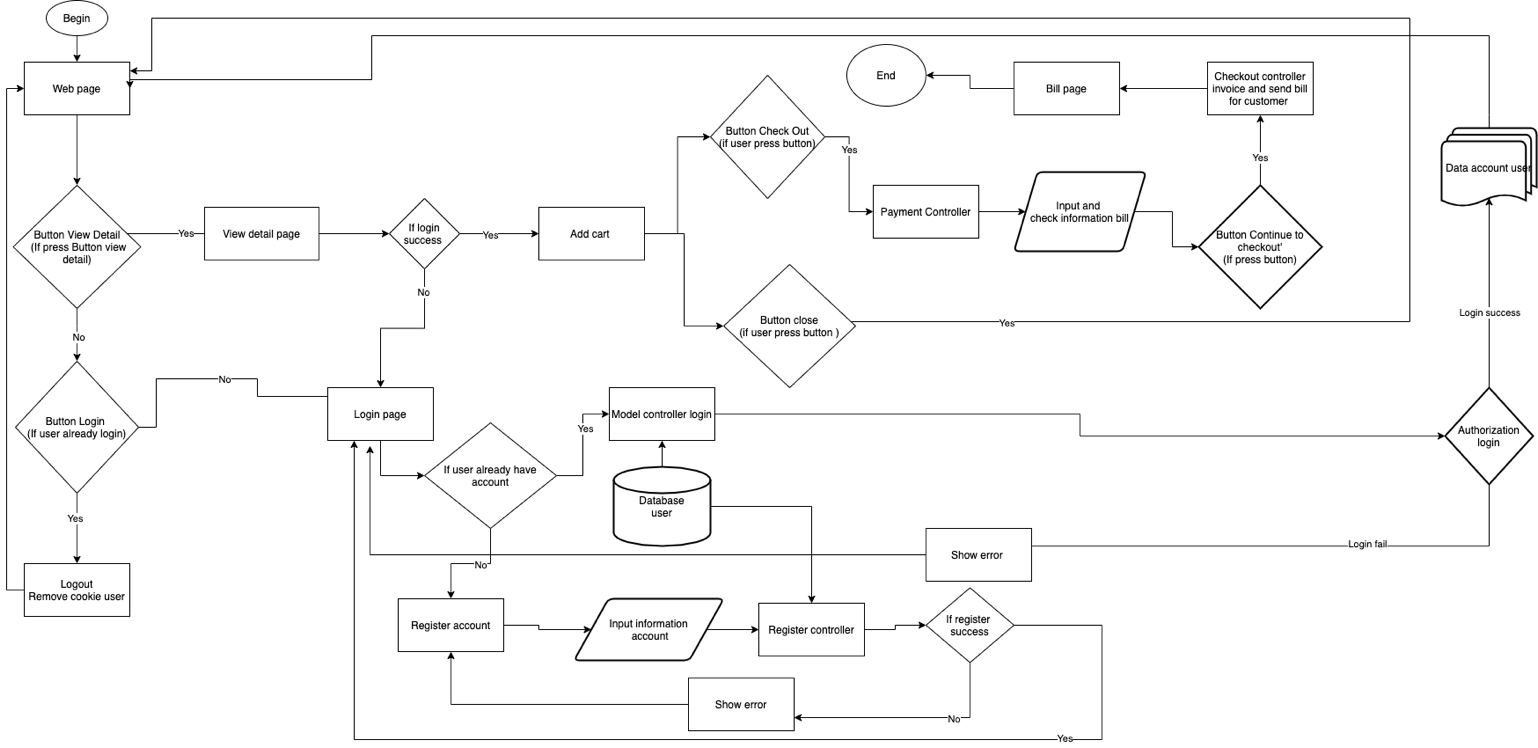
****

Figure 16: Checkout

Figure 17: Search

1. **Conclusion**

After completing the "Double T Shop" project, we have a lot more understanding about java web and client-server models, know how to handle server-side data, apply the MVC model to the project.

Thereby we can write our own websites, applications as well as for customers, knowing the stages from coding to deploying websites to hosting. Know how to handle requests from the client and from the server responding to data.