MINISTRY OF EDUCATION & TRAINING

**HO CHI MINH CITY UNIVERSITY OF ECONOMICS AND FINANCE**



**FINAL PROJECT**

# Course: Web Application Development

# Course code: 232.ITE1265E

**Watch Shop**

**Student’s Name: Trương Hùng Dũng.**

**Student ID : 215051286**

**Student’s Name: Nguyễn Văn Mạnh.**

**Student ID : 215051286**

**Student’s Name: Nguyễn Hòa Hồng Đức.**

**Student ID : 215051286**

**Class: B04E**

**Ho Chi Minh city - 06-2024**

MINISTRY OF EDUCATION & TRAINING

**HO CHI MINH CITY UNIVERSITY OF ECONOMICS AND FINANCE**



**FINAL PROJECT**

# Course: Web Application Development

# Course code: 232.ITE1265E

**Watch Shop**

**Student’s Name: Trương Hùng Dũng.**

**Student ID : 215051882**

**Student’s Name: Nguyễn Văn Mạnh.**

**Student ID : 215051286**

**Student’s Name: Nguyễn Hòa Hồng Đức.**

**Student ID : 215051241**

**Class : B04E**

**Instructor:** Phạm Trọng Nghĩa

**Ho Chi Minh city - 06-2024**

|  |
| --- |
| Software Design Document, Testing, and Deployment and  Configuration Management |
| Watch Shop |

|  |
| --- |
| Prepared for : University Economics and Finance |
| Version 1.0  Submitted to: Pham Trong Nghia |
| By: Team9, Winter 2024 |

Members of Team 9:

|  |  |
| --- | --- |
| **Name** | **Student ID** |
| Nguyen Van Manh | 215051286 |
| Truong Hung Dung | [215051882](mailto:a_samso@encs.concordia.ca) |
| Nguyen Hoa Hong Duc | 215051241 |

## TABLE OF CONTENTS

[1. Introduction 1](#_Toc170142206)

[1.1. Purpose 1](#_Toc170142207)

[1.2. Scope 2](#_Toc170142208)

[1.3. References 2](#_Toc170142209)

[1.4. Overview 3](#_Toc170142210)

[2. Design considerations 3](#_Toc170142211)

[2.1. Assumptions 3](#_Toc170142212)

[2.2. Constraints 3](#_Toc170142213)

[2.3. System environment 3](#_Toc170142214)

[2.4. Design methodology 3](#_Toc170142215)

[3. Architecture 3](#_Toc170142216)

[3.1. System design 4](#_Toc170142217)

[3.2. System decomposition 5](#_Toc170142218)

[3.2.1. Functional decomposition tree 5](#_Toc170142219)

[3.3.User case 5](#_Toc170142220)

[3.3.1.Manager User case 5](#_Toc170142221)

[3.3.2.Manager Use case 6](#_Toc170142223)

[4. Data design 7](#_Toc170142225)

[4.1. Data description 7](#_Toc170142226)

[4.2. Data dictionary 7](#_Toc170142227)

[5. Component design 10](#_Toc170142228)

[5.1. Login 10](#_Toc170142229)

[5.2. Register Account 11](#_Toc170142230)

[5.3. Edit Profile 12](#_Toc170142231)

[5.4. Search Product 13](#_Toc170142232)

[5.5. See cart 14](#_Toc170142234)

[5.6. Change PassWord 14](#_Toc170142236)

[5.7. Add Staff 15](#_Toc170142238)

[5.8. Authorize 16](#_Toc170142240)

[5.9. Add Cart 17](#_Toc170142242)

[6. Software interface design 19](#_Toc170142243)

[6.1. User interface design 19](#_Toc170142244)

[6.1.1. Web pages in a tree 20](#_Toc170142245)

[6.1.1.1. Description 21](#_Toc170142246)

[6.1.1.2. Objects and Actions 21](#_Toc170142247)

[6.1.1.3. Screen image 24](#_Toc170142248)

[7. Class diagrams 32](#_Toc170142249)

[7.1. Basic folders for class diagram 32](#_Toc170142250)

[7.2. Class diagram 57](#_Toc170142251)

[7.3. Class Watch 59](#_Toc170142252)

[7.4. Class Brand 60](#_Toc170142253)

[7.5. Class User cart 62](#_Toc170142254)

[7.6. Class Watch shop user 62](#_Toc170142255)

[7.7. Class AspNetUsers 64](#_Toc170142256)

[7.8. Class Favorite 65](#_Toc170142258)

[8. Appendices 66](#_Toc170142259)

[A. Setup and Configuration 66](#_Toc170142260)

[Deployment of the UUIS on the server: On the server UUIS has following structure: 67](#_Toc170142261)

[B. Tool set and environment 68](#_Toc170142262)

[References 70](#_Toc170142263)

## 

## LIST OF FIGURES

[Figure 3.1.1. Block diagram for Watch Shop 4](#_Toc170143441)

[Figure 3.1.2. Context diagram for Watch Shop 4](#_Toc170143442)

[Figure 3.2.1. Decompositional Tree for Watch Shop App 5](#_Toc170143443)

[Figure 3.3.1. Manage user case diaram 6](#_Toc170143444)

[Figure 3.3.2. user use case diaram 7](#_Toc170143445)

[Figure 5.1.1 Sequence diagram for Login 10](#_Toc170143446)

[Figure 5.1.2 Activity diagram for Login 11](#_Toc170143447)

[Figure 5.1.3 Sequence diagram for Register Account 1 1](#_Toc170143448)

[Figure 5.1.4 Activity diagram for Register Account 12](#_Toc170143449)

[Figure 5.1.5 Sequence diagram for Edit Profie 12](#_Toc170143450)

[Figure 5.1.4.6 Activity diagram for Edit Profie 13](#_Toc170143451)

[Figure 5.1.7 Sequence diagram for search product 13](#_Toc170143452)

[Figure 5.1.8 Activity diagram for Edit Profie 13](#_Toc170143454)

[Figure 5.1.9 Sequence diagram for See cart 14](#_Toc170143455)

[Figure 5.1.10 Activity diagram for See cart 14](#_Toc170143457)

[Figure 5.1.11 Sequence diagram for change password 15](#_Toc170143458)

[Figure 5.1.12 Activity diagram for change password 15](#_Toc170143460)

[Figure 5.1.13 Sequence diagram for add staff 16](#_Toc170143461)

[Figure 5.1.14 Activity diagram for add staff 16](#_Toc170143463)

[Figure 5.1.15 Sequence diagram for Authorize 17](#_Toc170143464)

[Figure 19. Activity diagram for Authorize 17](#_Toc170143466)

[Figure 5.1.16 Sequence diagram for add cart 18](#_Toc170143467)

[Figure 5.1.17 Activity diagram for add cart 19](#_Toc170143468)

[Figure 6.1.1 A tree of web pages 20](#_Toc170143469)

[6.1.2 Home Page page 24](#_Toc170143470)

[Figure 6.1.3 Login page 25](#_Toc170143471)

[Figure 6.1.4 Page Register 25](#_Toc170143472)

[Figure 6.1.5 Your products Page 26](#_Toc170143473)

[Figure 6.1.6 profile Page 27](#_Toc170143474)

[Figure 6.1.7 Change Password page 28](#_Toc170143475)

[Figure 6.1.8 Manage Email 29](#_Toc170143476)

[Figure 6.1.9 Two authentication Page 30](#_Toc170143477)

[Figure 6.1.10 Personal data Page 31](#_Toc170143478)

[Figure 7.1.1 Basic folder for class diagram 32](#_Toc170143479)

[Figure 7.2.1 Class diagram 57](#_Toc170143480)

[Figure 7.2.2 Class Order 58](#_Toc170143481)

[Figure 7.2.3 Class Watch 59](#_Toc170143482)

[Figure 7.2.4 Class Brand 61](#_Toc170143483)

[Figure 7.2.5 Class User Cart-Description: 62](#_Toc170143484)

[Figure 7.2.6 Class Watch shop user 63](#_Toc170143485)

[Figure 7.2.7 Class AspNetUsers 65](#_Toc170143486)

[Figure 7.2.8 Class Favorite 66](#_Toc170143487)

[Figure 42. structure of Deployment 67](#_Toc170143488)

## 

## LIST OF TABLE

[Table 4.2.1 Data Dictionary 10](#_Toc170143489)

[Table 7.2.1 Class Order 58](#_Toc170143490)

[Table 7.2.2 Class watch 60](#_Toc170143491)

[Table 7.2.3 Class Brand 61](#_Toc170143492)

[Table 7.2.4 Class user cart 62](#_Toc170143493)

[Table 7.2.5 Class Watch shop user 64](#_Toc170143494)

[Table 7.2.6 Class AspNetUsers 65](#_Toc170143495)

[Table 7.2.7 Class Favorite 66](#_Toc170143496)

# Introduction

Today's society is constantly developing, along with the growing trend of e-commerce. Everything is so simple now, just having a computer or even just a smartphone connected to the internet, buying, selling and exchanging commerce becomes easier than ever for everyone with just a few minutes. mouse click. With electronic commerce, all barriers of geographical space or working time are eliminated. Products are clearly introduced to not only shoppers in that area but throughout Vietnam, and even people around the world. Sellers now no longer just sit around waiting for customers to come to them, but actively stand up and find customers. And when the number of customers increases, it is also proportional to the increase in revenue, which is what every business is aiming for. By e-competitioncommerce, all businesses from large, medium and small can unleash their creativity. , fair . Bold new business ideas, marketing strategies, promotions... can all be applied and directed directly to customers as quickly as possible without spending too much money because it's all still packaged in one package. e-commerce site (website). On those bases, my project is implementing the topic "Building a Watch website" to address the needs of product promotion and sales.

## Purpose

The main content of the project is to build an e-commerce website with necessary functions such as customers can create accounts and log in to the website, can search and view information about different types of watches. For customers who are logged in, they can manage accounts and make purchases. Administrators, through the website, can manage the store's situation such as: warehouse status, store revenue... thereby making future business strategies. An important requirement for the website E-commerce not only must process quickly and accurately, but it must also have an eye-catching interface to attract customers. Therefore, I decided to use ASP.NET – a C# framework as the logic processing language; HTML, CSS and JavaScript are the languages ​​used to design interfaces. Above are the system requirements within the framework of this project. Although the system only meets the functions of a small-sized e-commerce website - suitable for individual or household retail businesses, in the future I will continue to develop the system into a chain of stores. products, build mobile apps for the system and especially develop logistics services.

## Scope

This document gives a detailed description of the software architecture of the inventory system. It specifies the structure and design of some of the modules discussed in the SRS. It also displays some of the use cases that had transformed into sequential and activity diagrams. The class diagrams show how the programming team would implement the specific module.

## References

The user of this SDD may need the following documents for reference:

IEEE Standard 1016-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE Computer Society, 1998.

Team 9, 2024. Software Requirements Specification, Watch Shop. Last modified:.

Team 9, 2024. Software Test Documentation, Watch Shop. Last modified:

## Overview

This document is written according to the standards for Software Design Documentation explained in “IEEE Recommended Practice for Software Design Documentation”.

Sections 3 – 5 contain discussions of the designs for the project with diagrams, section 6 shows samples of UI from the system, and section 7 contains the class diagrams. The appendices contain the setup and configuration needed for the Watch Shop, a list of functions that are implemented in this version, and that are to be implemented in future version, and a list of tools and environment used in the entire project, along with the time contribution of team members. The appendices also include the test report and test cases.

# Design considerations

## Assumptions

The user of the inventory system is aware of basic operations of a computer and web pages. The user also understands the standard terms used for operation.

## Constraints

The built system is accessible only through the local of team. The system is implemented using C# and ASP.NET technology.

## System environment

The web based unified inventory system is designed to work on all operating systems.Can only be accessed via the device installed by the project of group 9, that is connected to the UI of a server. It is accessible at all times.

## Design methodology

The system is designed with flexibility for further development and/or modification. The system is divided into manageable processes that are grouped to sub-modules and modules that are built with abstraction.

# Architecture

## System design

The block diagram below shows the principal parts of the system and their interaction

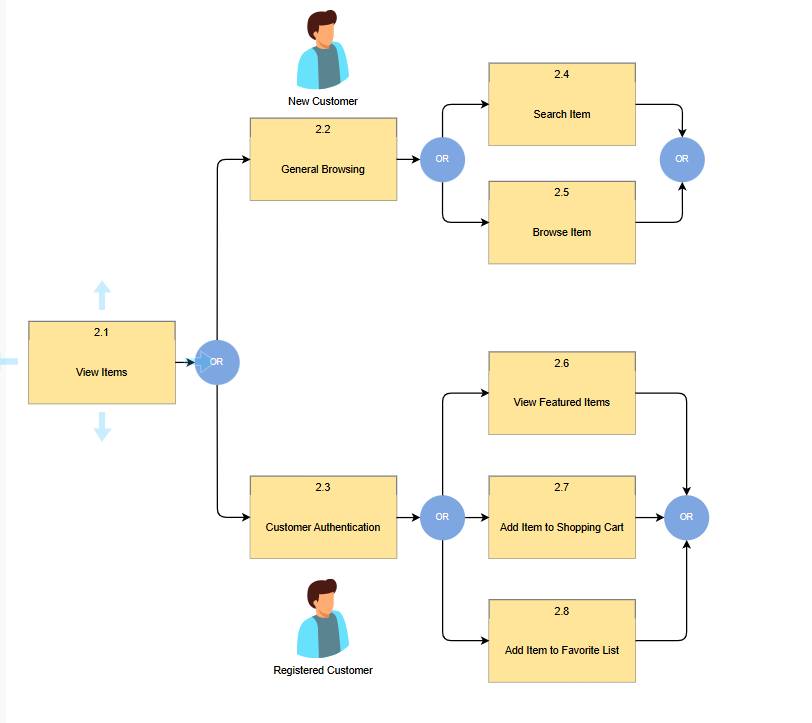


Figure 3.1.1. Block diagram for Watch Shop

The context diagram shows the main actors interacting with the system.

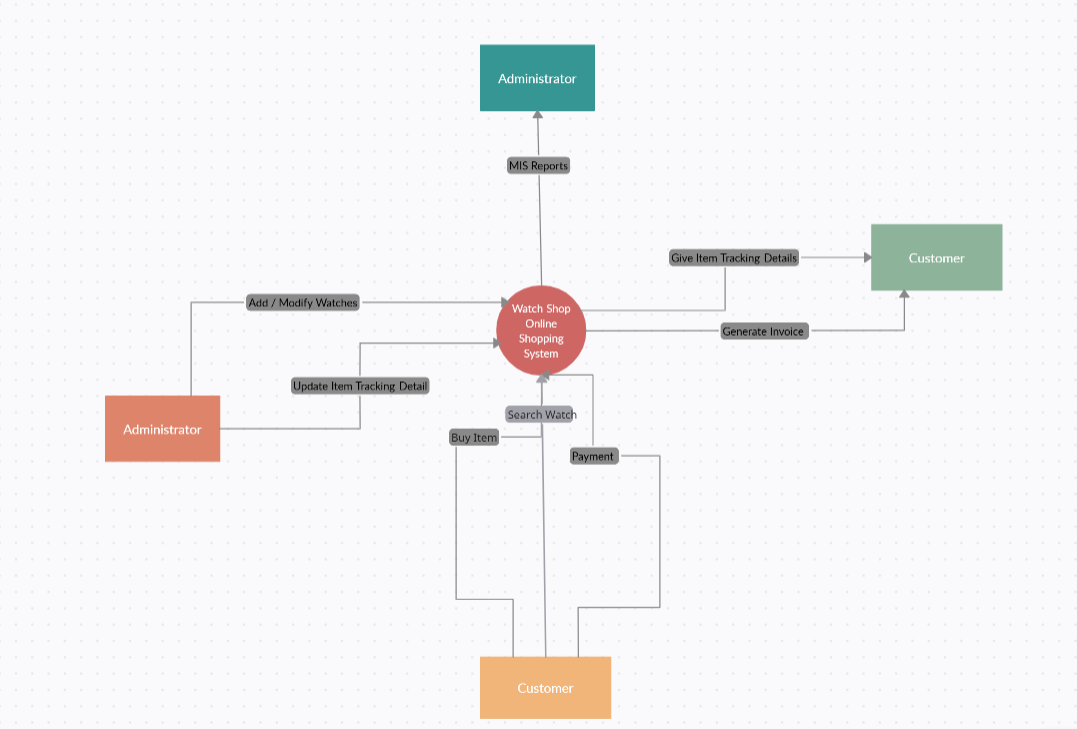


Figure 3.1.2. Context diagram for Watch Shop

## System decomposition

### Functional decomposition tree

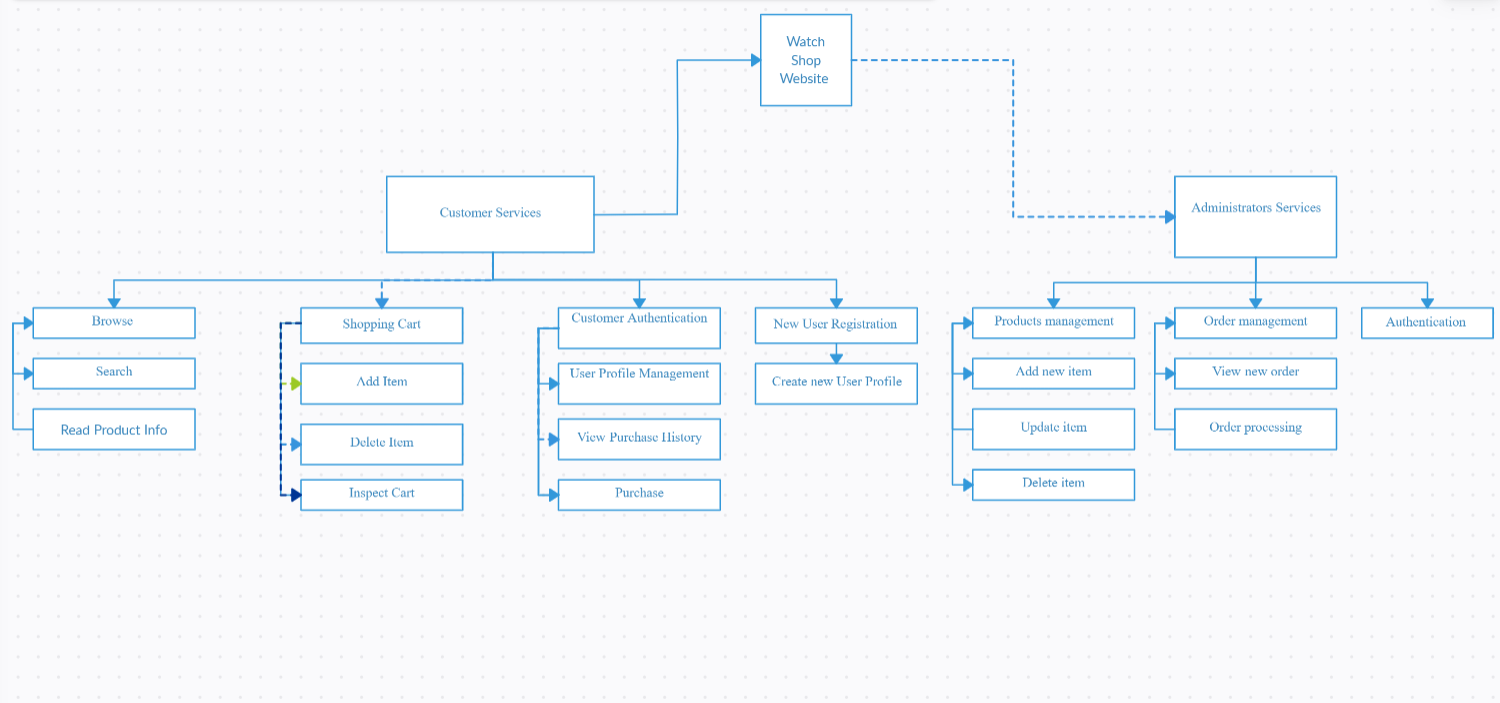


Figure 3.2.1. Decompositional Tree for Watch Shop App

## 3.3.User case

### 3.3.1.Manager User case

### 

Figure 3.3.1. Manage user case diaram

### 3.3.2.Manager Use case

### 

Figure 3.3.2. user use case diaram

# Data design

## Data description

SQL LocalDB to communicate with the database that is installed locally on the server.

## Data dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table Name** | **Field** | **Type** | **Null** | **Default** |
| Watch | ID | int | no | 0 |
|  | Name | String | yes | null |
|  | Price | int | yes | null |
|  | sale | int | yes | null |
|  | GlassType | String | yes | null |
|  | MachineType | String | yes | null |
|  | WireMaterial | String | yes | null |
|  | Pic1 | String | yes | null |
|  | Pic2 | String | yes | null |
|  | Pic3 | String | yes | null |
|  | Pic4 | String | yes | null |
|  | Stocking | int | yes | null |
|  | BrandID | int | yes | null |
|  | IsFavorite | int | yes | null |
| Brand | branID | int | No | 0 |
|  | Name | String | yes | null |
|  | Desciption | String | yes | null |
|  | LogoUrl | String | yes | null |
|  | Origin | String | yes | null |
| Oder | ID | int | No | 0 |
|  | quantity | int | yes | null |
|  | UserCartID | int | yes | null |
|  | WatchID | int | yes | null |
| HistoryPurchase | ID | int | No | 0 |
|  | HistoryPurchaseKey | String | yes | null |
|  | DatePurchase | Date | yes | null |
|  | UserID | int | yes | null |
|  | WatchID | int | yes | null |
| Favorite | Id | int | No | 0 |
|  | UserID | int | yes | null |
|  | WatchID | int | yes | null |
| AspNetUser | ID | int | No | 0 |
|  | Name | String | yes | null |
|  | UserName | String | yes | null |
|  | NomalizeedEMail | String | yes | null |
|  | EmailConfirmed | int | yes | null |
|  | PasswordHash | String | yes | null |
|  | SecurityStamp | String | yes | null |
|  | ConcurrencyStamp | String | yes | null |
|  | PhoneNumber | String | yes | null |
|  | PhoneNumberConfirmed | int | yes | null |
|  | TwoFactorEnabled | String | yes | null |
|  | LockoutEnd | String | yes | null |
|  | LockoutEnabled | String | yes | null |
|  | AccessFailedCount | int | yes | null |
| AspNetRole | ID | int | no | 0 |
|  | Name | String | yes | null |
|  | NormalizedName | String | yes | null |
|  | ConcurrencyStamp | String | yes | null |
| AspNetUserLogin | LoginProvider | int | No | 0 |
|  | ProviderKey | int | yes | null |
|  | ProviderDisplayName | String | yes | null |
|  | userID | int | yes | null |

Table 4.2.1 Data Dictionary

# Component design

“Users with appropriate permission” in the diagram refers to the users who are given exemptions or/and users of a particular level. Precise permissions are listed in Section 2.3 of SRS.

For diagrams with multiple functions, the design is the same for those functions, except for parameters/methods/classes.

## Login

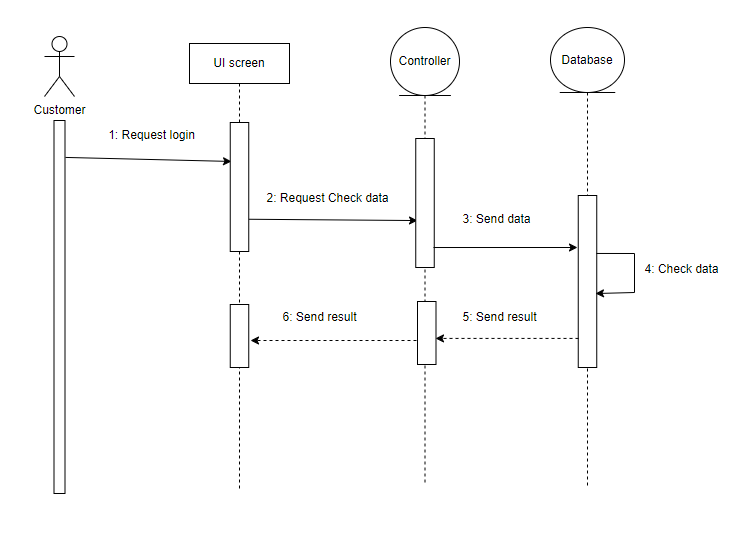
****

Figure 5.1.1 Sequence diagram for Login

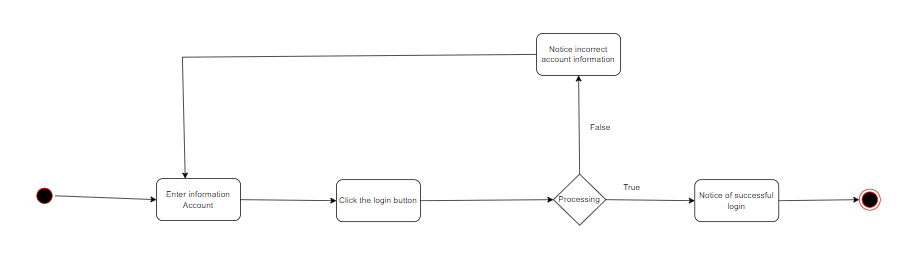
****

Figure 5.1.2 Activity diagram for Login

## Register Account

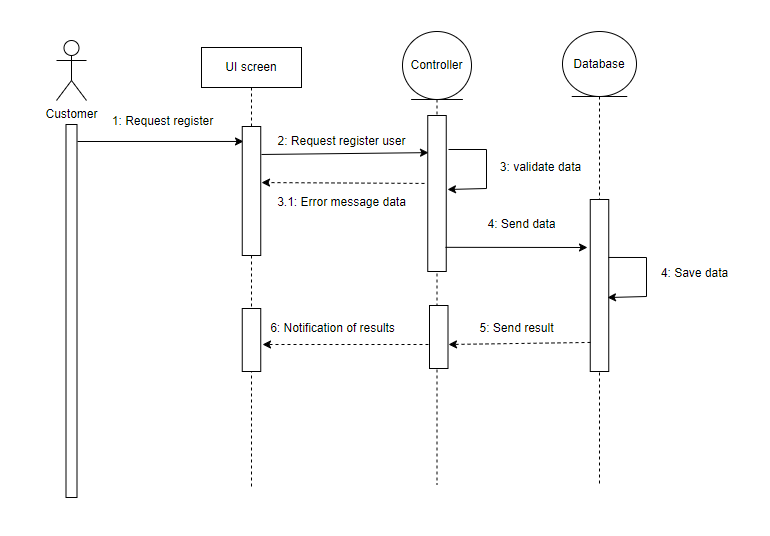
****

Figure 5.1.3 Sequence diagram for Register Account

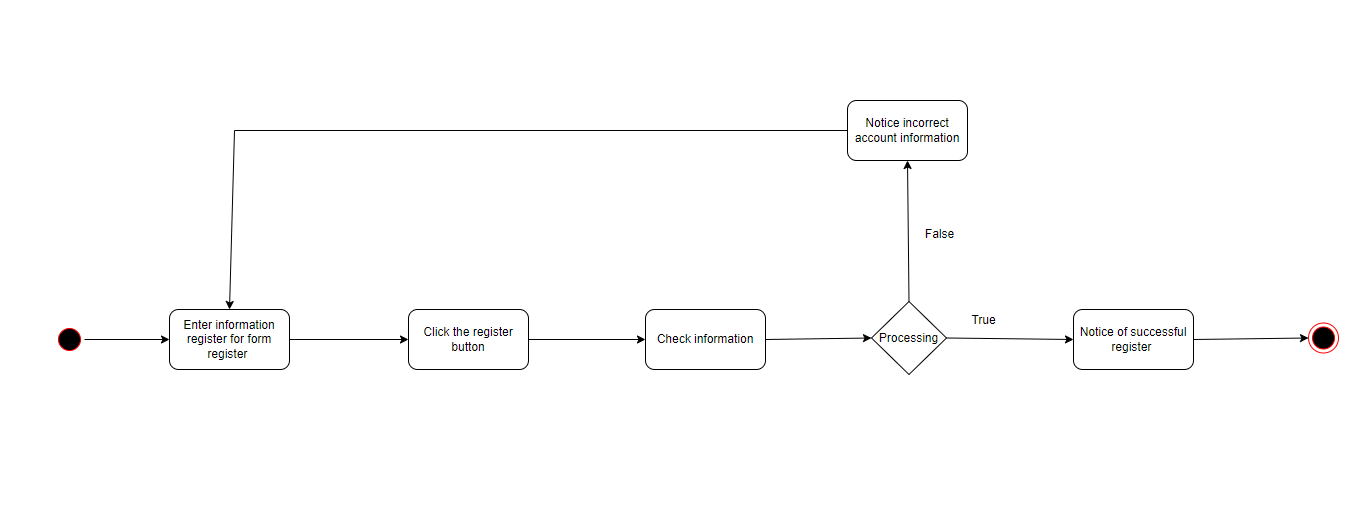
****

Figure 5.1.4 Activity diagram for Register Account

## Edit Profile

Figure 5.1.5 Sequence diagram for Edit Profie

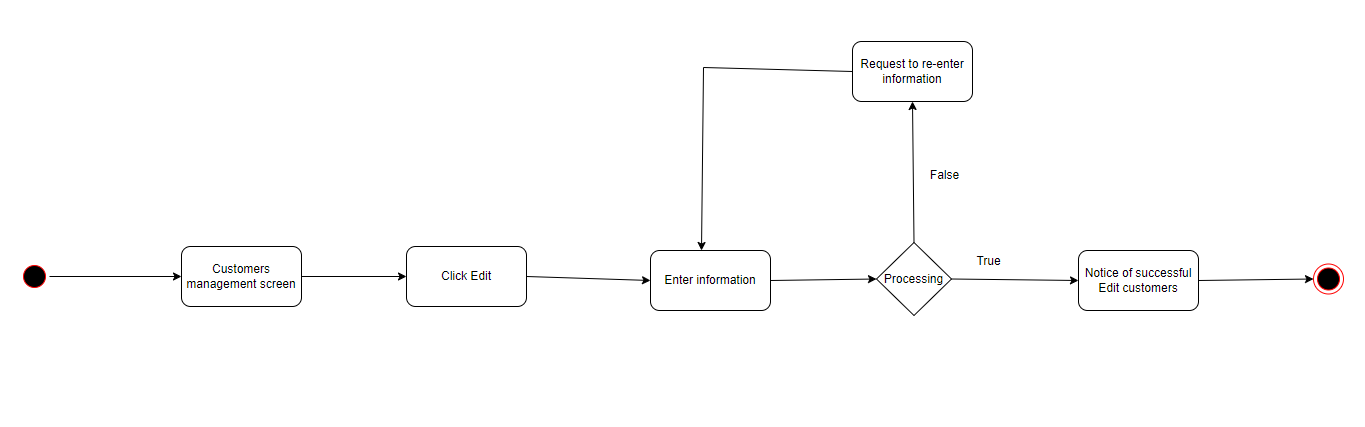
****

Figure 5.1.4.6 Activity diagram for Edit Profie

## Search Product

## 

Figure 5.1.7 Sequence diagram for search product

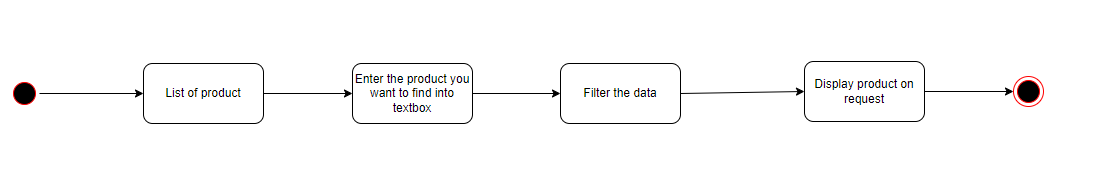


Figure 5.1.8 Activity diagram for Edit Profie

## See cart

## 

Figure 5.1.9 Sequence diagram for See cart

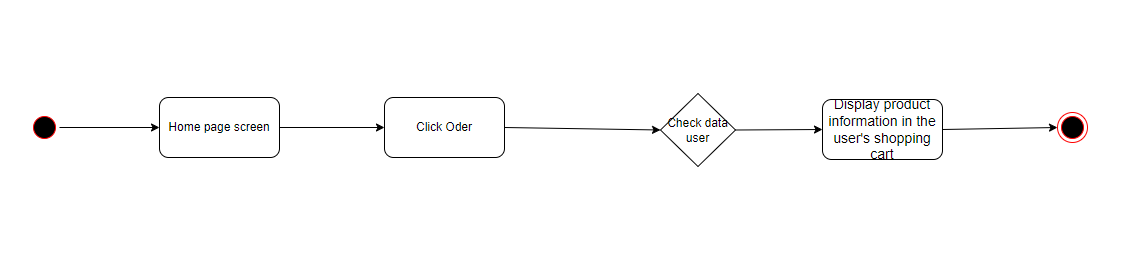


Figure 5.1.10 Activity diagram for See cart

## Change PassWord

## 

Figure 5.1.11 Sequence diagram for change password

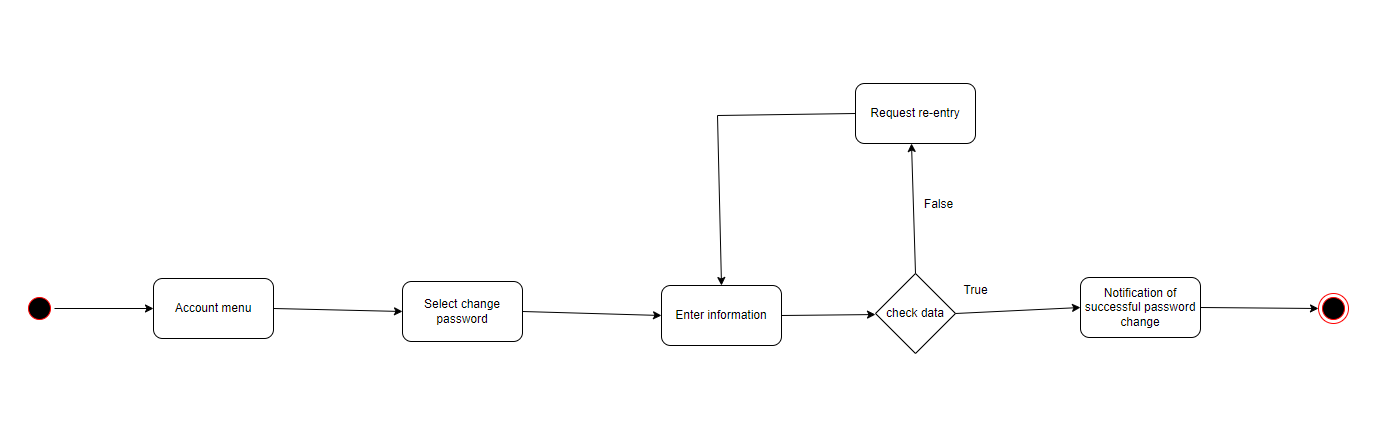


Figure 5.1.12 Activity diagram for change password

## Add Staff

## 

Figure 5.1.13 Sequence diagram for add staff

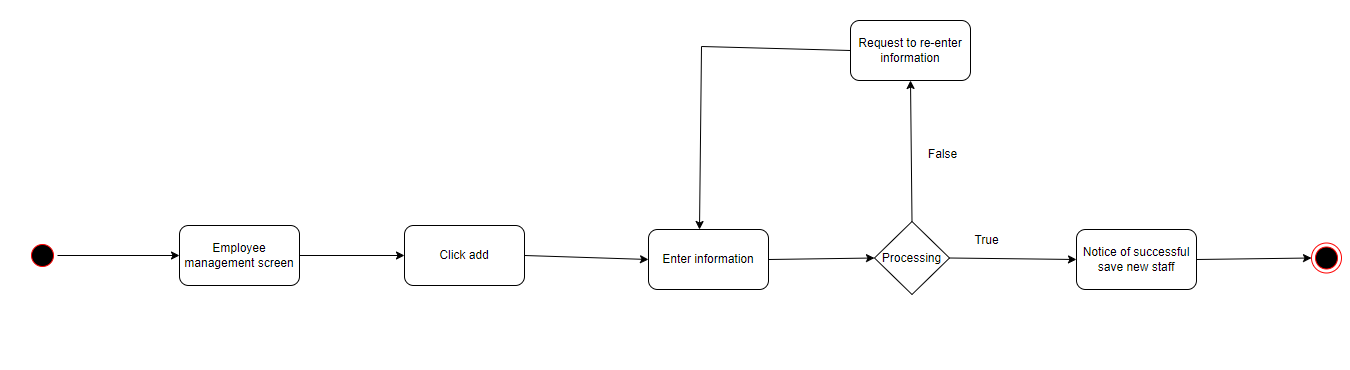


Figure 5.1.14 Activity diagram for add staff

## Authorize

## 

Figure 5.1.15 Sequence diagram for Authorize

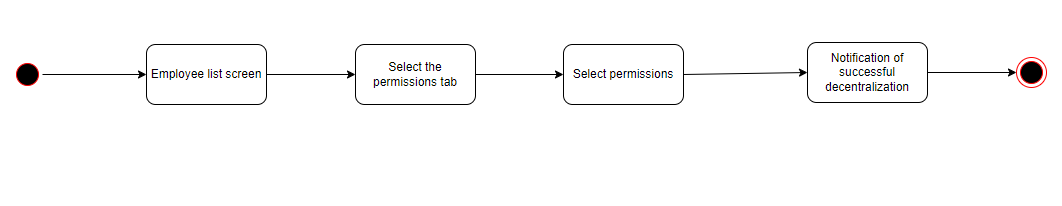


Figure 19. Activity diagram for Authorize

## Add Cart

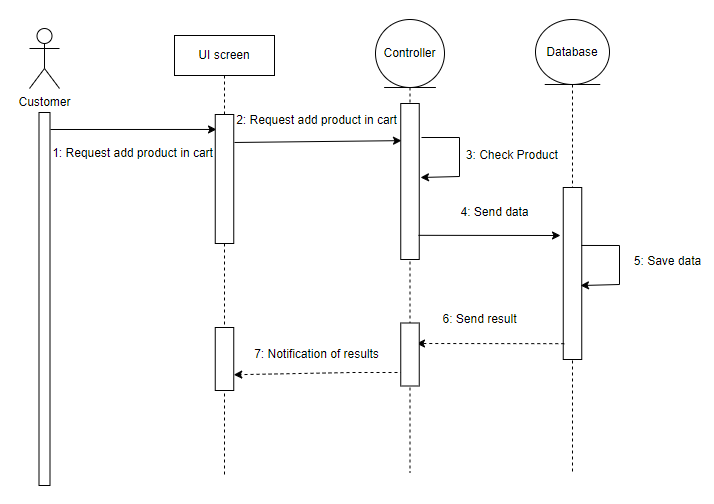
****

Figure 5.1.16 Sequence diagram for add cart

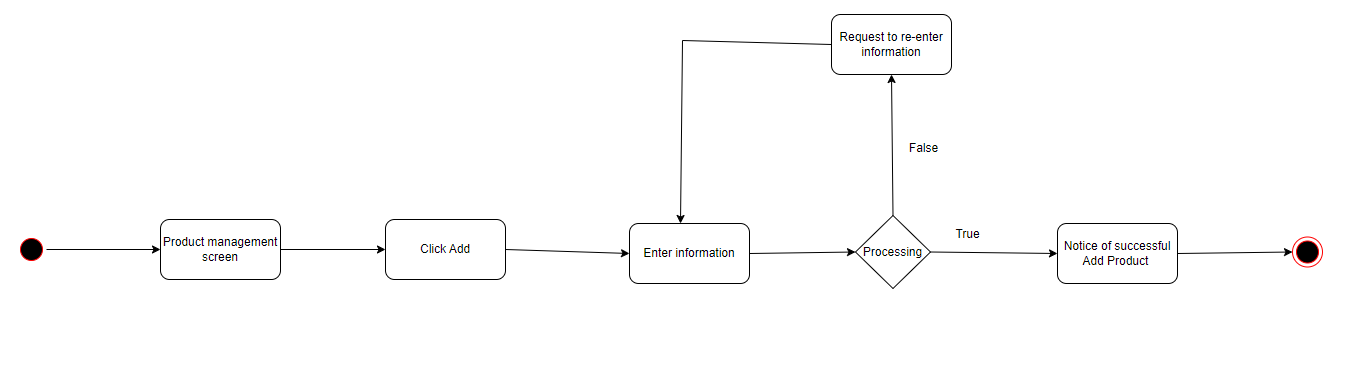


Figure 5.1.17 Activity diagram for add cart

# Software interface design

## User interface design

UI is designed according to UI design principles.

The structure principle: UI is organized in such a way that related things are combined together and

unrelated things are separated.

The simplicity principle: It is easy to follow the provided interface. In the case of mistake, system

displays error message

The visibility principle: All system‟s functions are available through UI. It does not overwhelm users with too many alternatives.

The feedback principle: Through the system of messages, the design keeps users informed of actions, errors, or exceptions.

The reuse principle: In design, same names were used to perform the same operations with different objects in order to reduce ambiguity.

### Web pages in a tree

Only when the user is logged in can access other functions according to permissions. "User" permission will access pages such as "Manage personal information", "favorites list", "Product list", "Shopping cart". As for "Admin" authority, you will be able to access pages such as: "Manage product list", "Manage user list", and all pages that user authority can access.

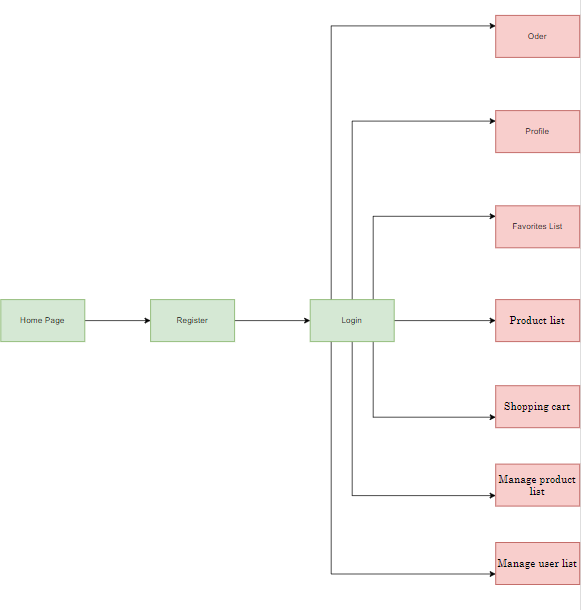


Figure 6.1.1 A tree of web pages

## Description

"Home Page" is the main interface of the website, it will display information about the year's featured items and best-selling items. In addition, it also displays buttons to access other pages including: login, register, profile, list of all products and shopping cart.

"Register" is a page used for users to register an account to access some pages that can only be accessed when logged in such as: Profile, shopping cart.

"Login" is a page used for users to log in to a shopping website to save their information on the system and be able to access some pages when requiring login.

"Order" is a page used to let users see all the products that Admin has added to the product management list.

"Profile" is a page that contains User information when the user registers for an account. In addition, the user can add some other information. In addition, this page also contains a list of products purchased by the User.

"Favorite product list" is a page containing products that have been added to the favorite list by the User by clicking the like icon on the product.

"Product List" is a page containing products that have been added by Admin to the product list for users to view.

"Shopping Cart" is a page containing products that have been added by the User through the product selection operation.

“Manage product list" is a page containing products that have been added by Adim through the new product add operation. In addition, Admin can add more products to this list.

"Manage User List" is a page for the system Admin to manage users including user account information and user personal information.

## Objects and Actions

In the "home page" after loading page, you will see the list of outstanding products of the year. Users will be able to view and access the page list of products for goods and details. However, if customers want to quickly products, they must be logged in if the user does not have an account, they will proceed to create a new VAT account to log in.

-The "Home Page" may link to other pages:

* Order
* Your Product
* profile
* Register
* Login

The "Register" page has several text boxes for users to enter their password and account information and press the submit button to proceed with the registration information.

- "Register" has the following functions:

* Register

The "Register" page has several text boxes for users to enter their password and account information and press the submit button to proceed with logging into the system.

- "Register" has the following functions:

* Login
* Register
* Forgot Password

"Order" is a page used to let users see all the products that Admin has added to the product management list.

-The "Order" may link to other pages:

* Product details.
* List Product.
* Add Product in cart

"Profile" is a page that contains User information when the user registers for an account. In addition, the user can add some other information. In addition, this page also contains a list of products purchased by the User.

* User details
* List Product bought

"Favorite product list" is a page containing products that have been added to the favorite list by the User by clicking the like icon on the product.

* Product details
* List Product Favorite.

"Product List" is a page containing products that have been added by Admin to the product list for users to view.

* Product details
* List Product

"Shopping Cart" is a page containing products that have been added by the User through the product selection operation.

* List of products added
* Product details
* Remove Product

“Manage product list" is a page containing products that have been added by Adim through the new product add operation. In addition, Admin can add more products to this list.

* Add Product
* Remove Product
* EditProduct

"Manage User List" is a page for the system Admin to manage users including user account information and user personal information.

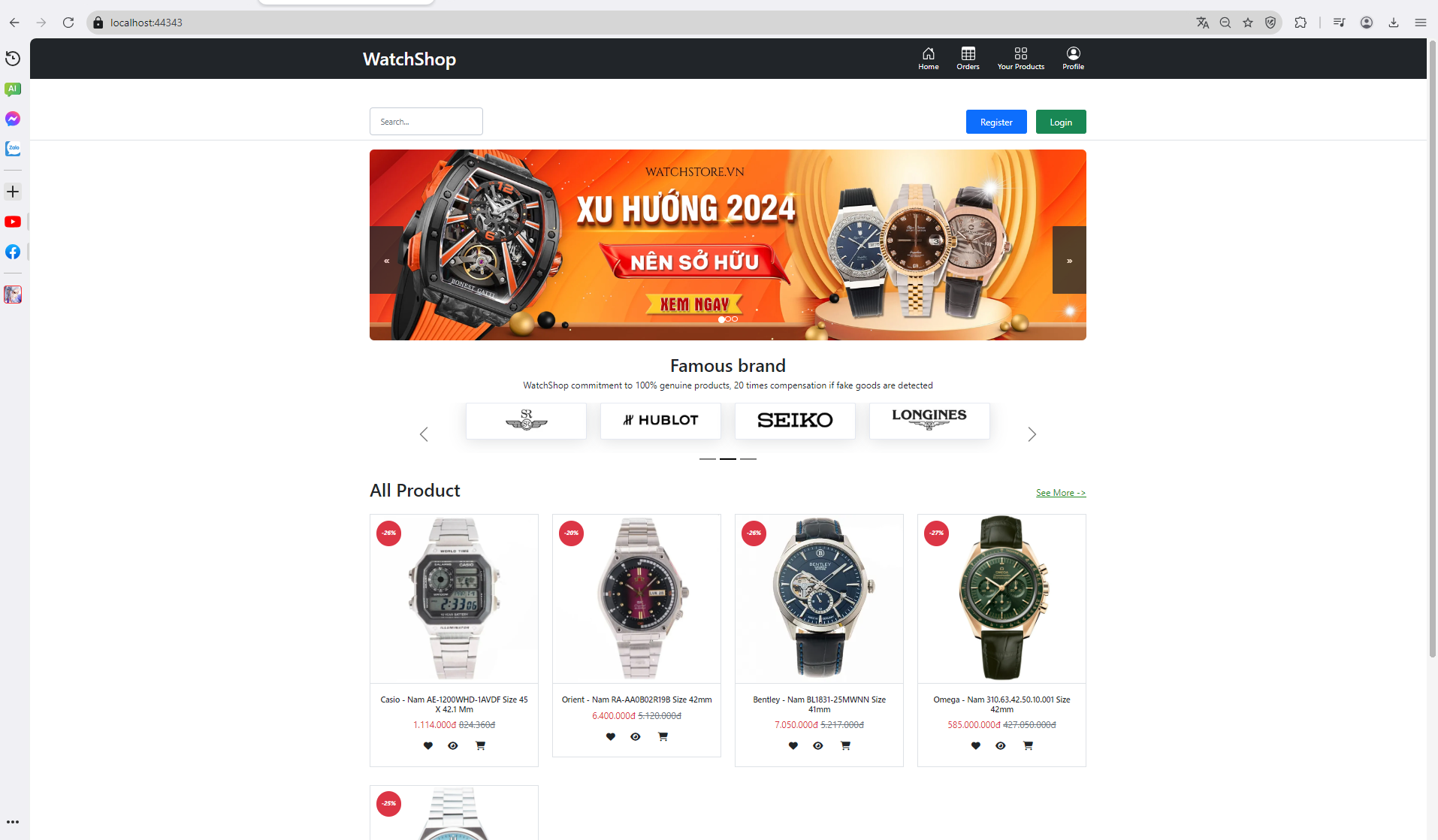
* Edit User.
* Add User

User interface

Used GUI components are menus, submenus, buttons, text boxes, check boxes, down drop lists, links, and tables. The only means of access to the entire database, by all users, is through this UI.

## Screen image

Some examples of UI are presented below:



6.1.2 Home Page page

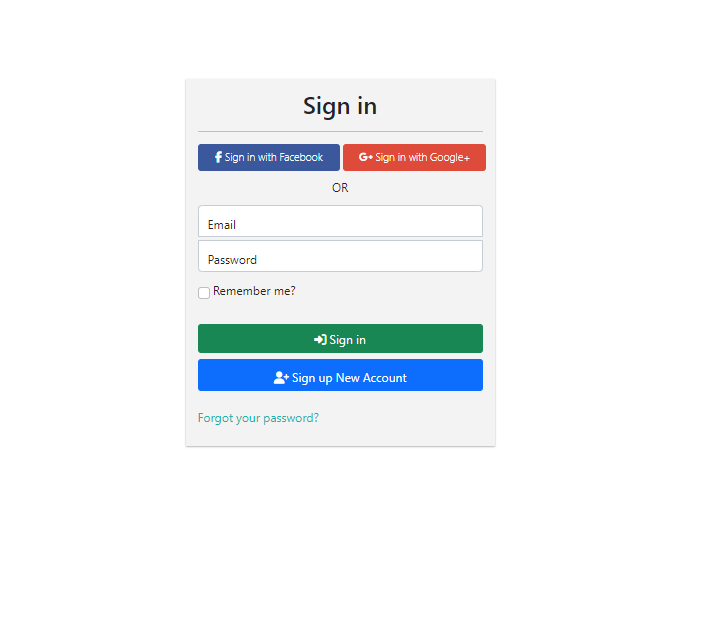


Figure 6.1.3 Login page

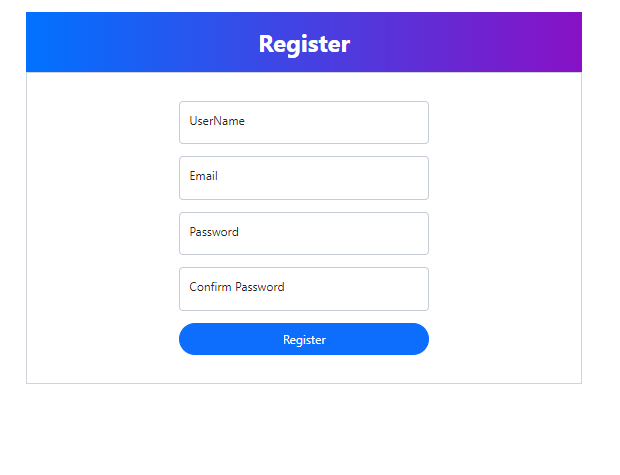
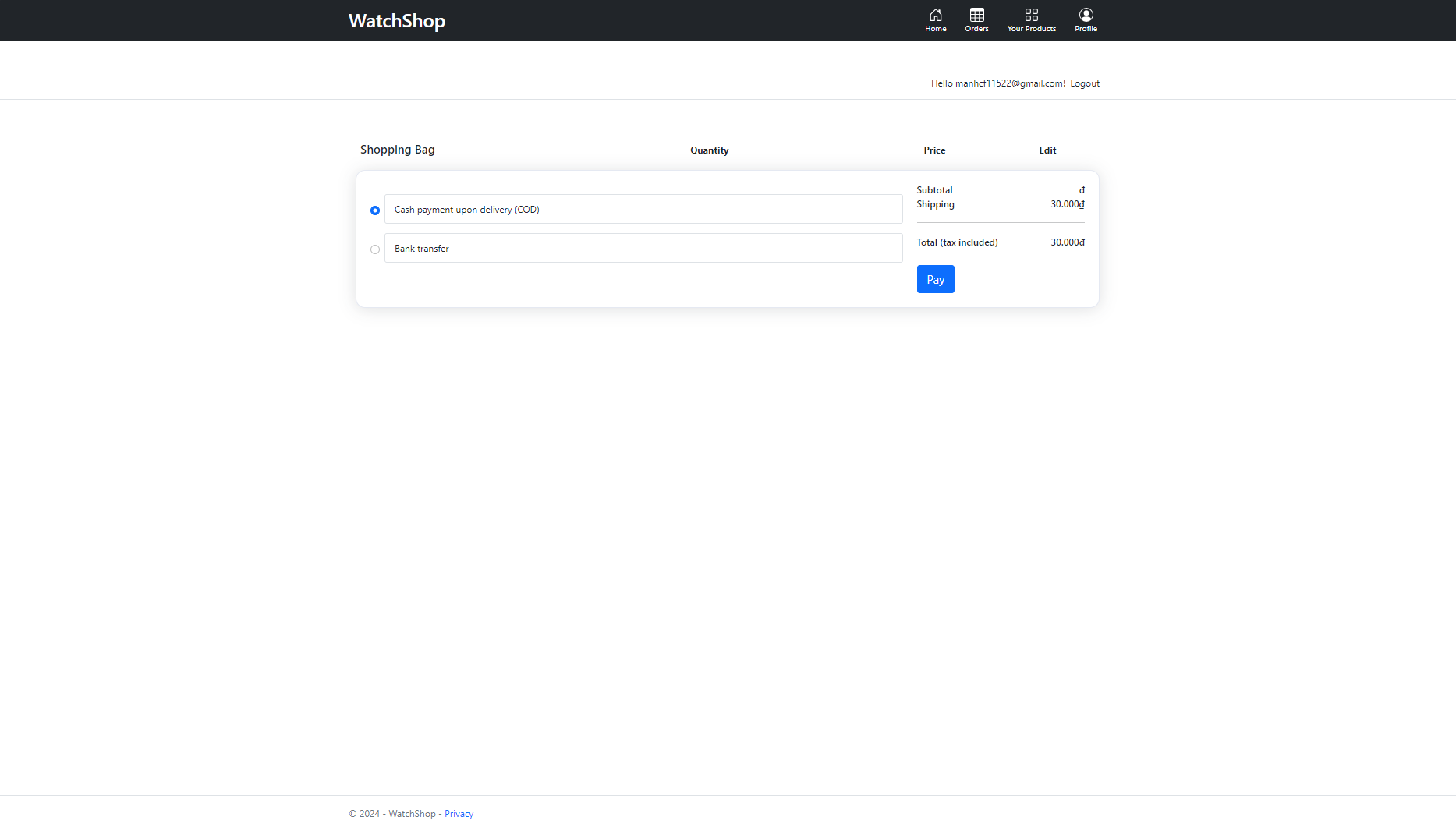


Figure 6.1.4 Page Register



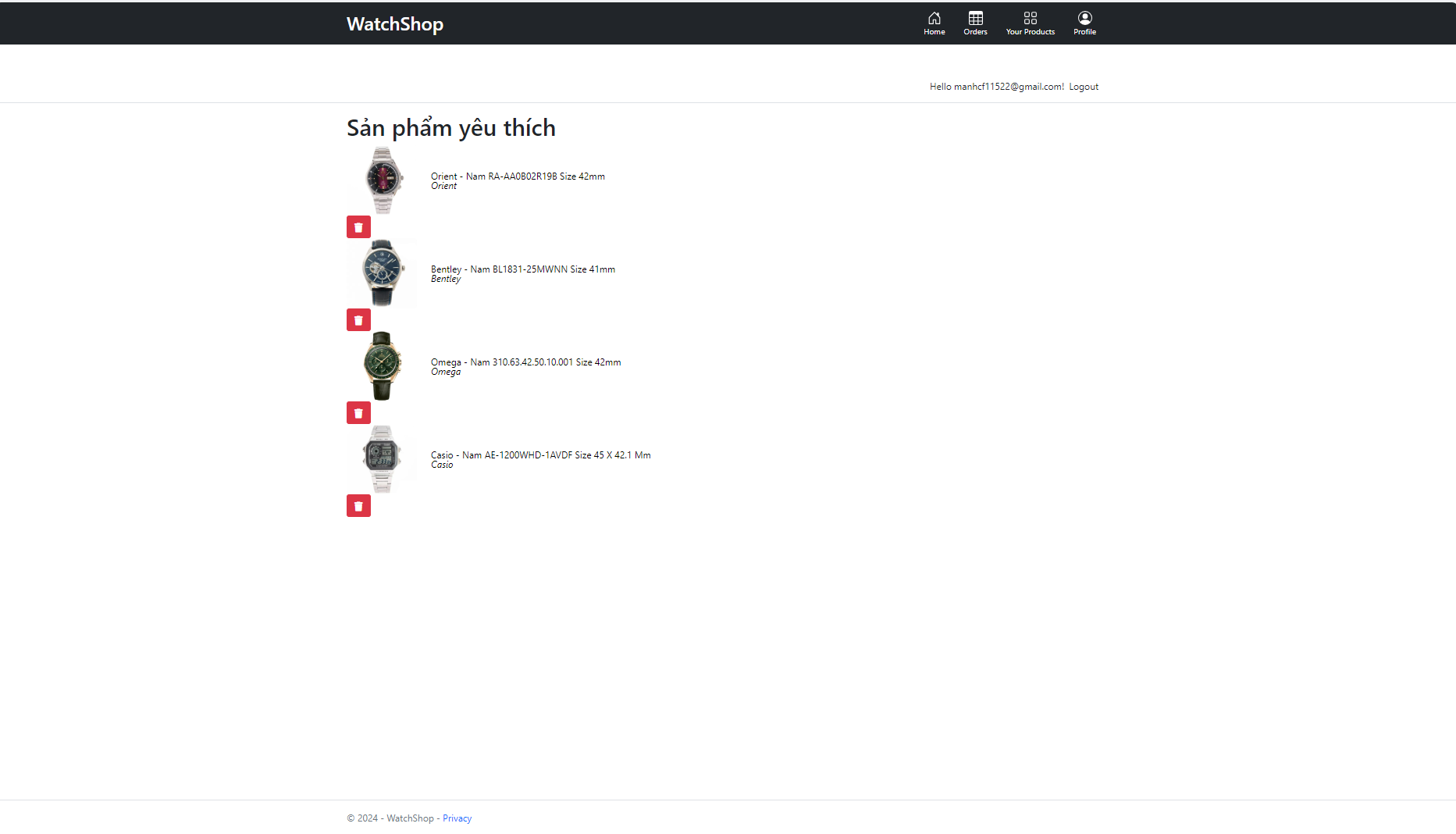
**Figure 26.** Order

Figure 6.1.5 Your products Page

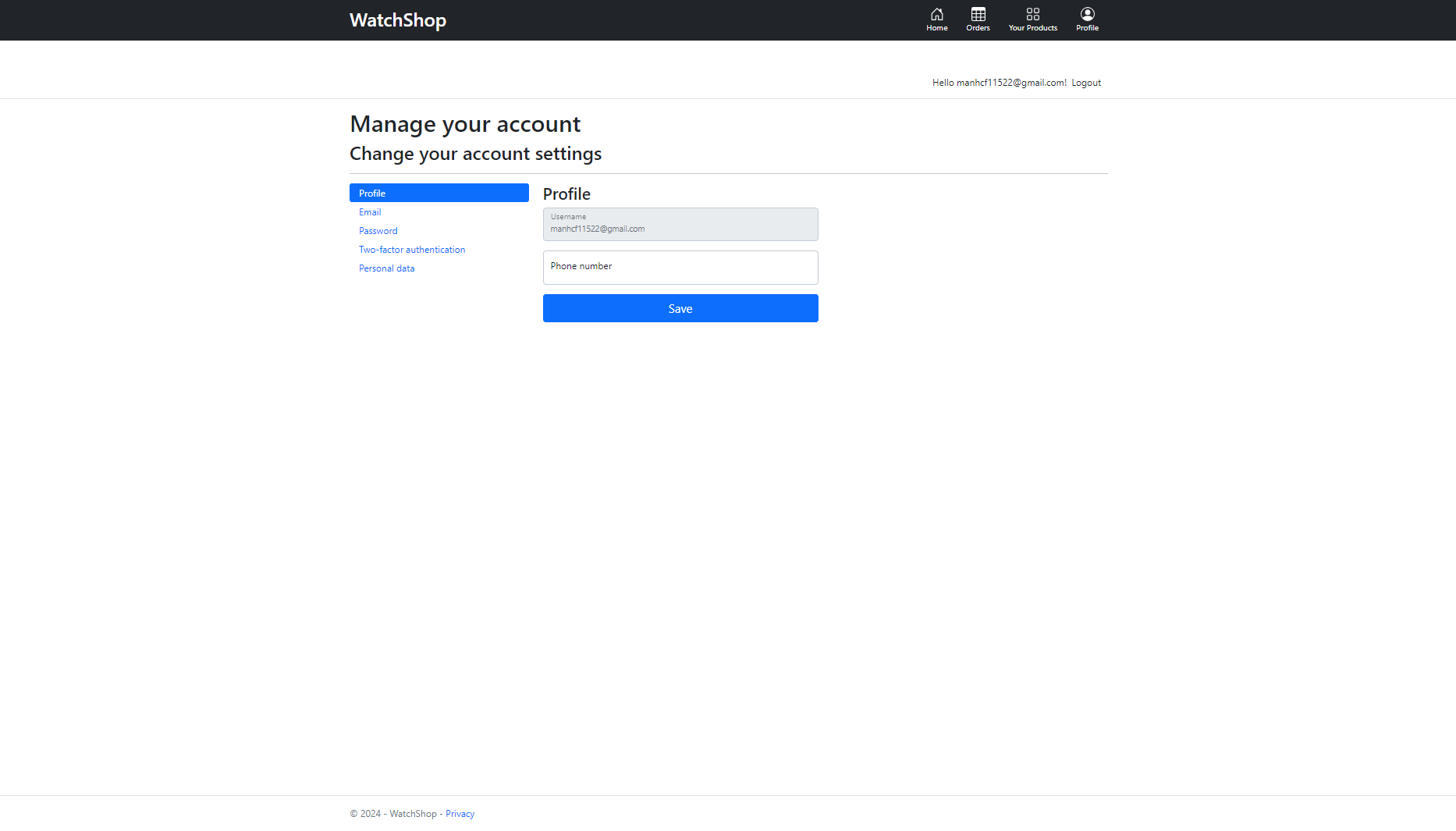


Figure 6.1.6 profile Page

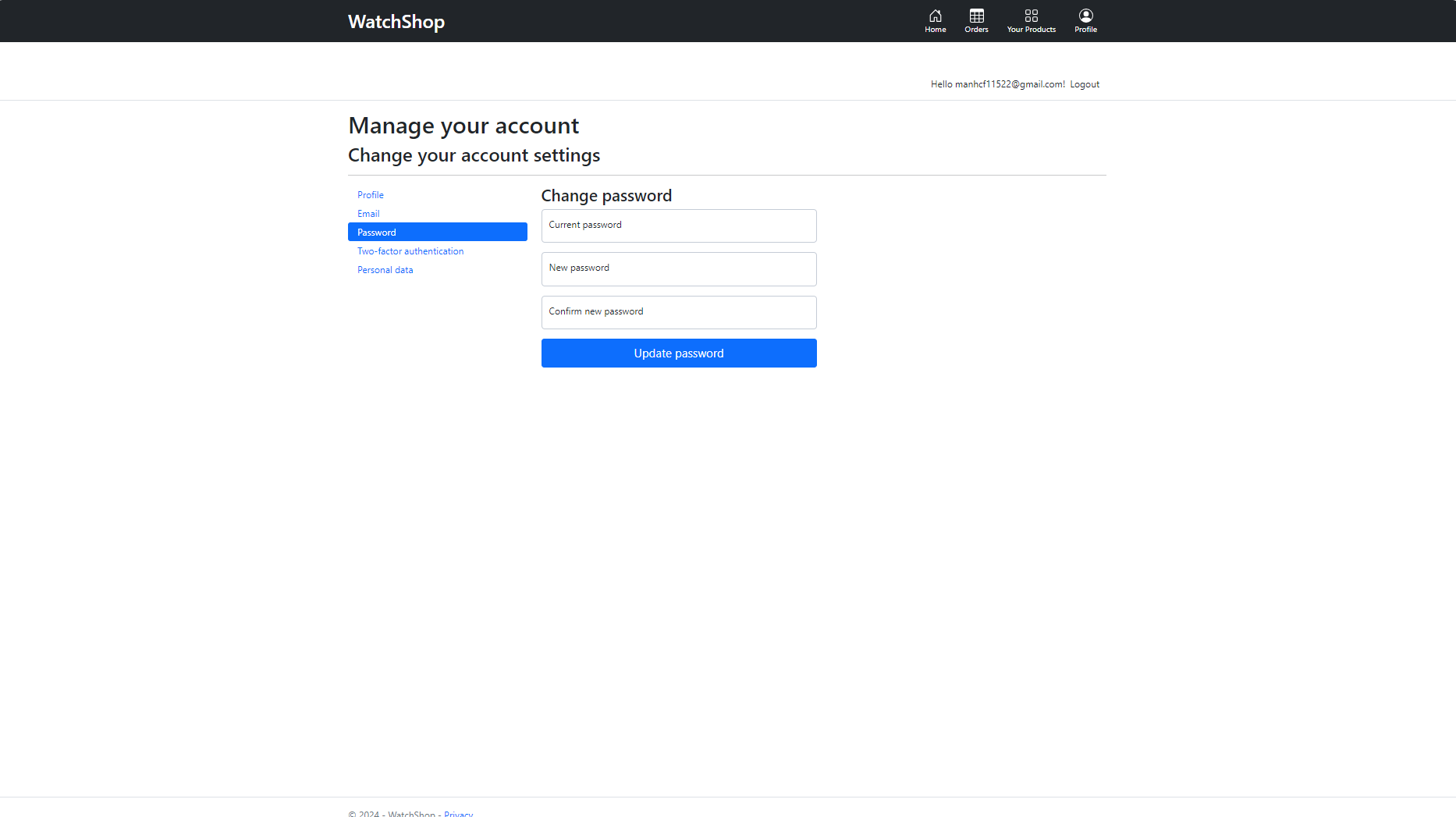


Figure 6.1.7 Change Password page

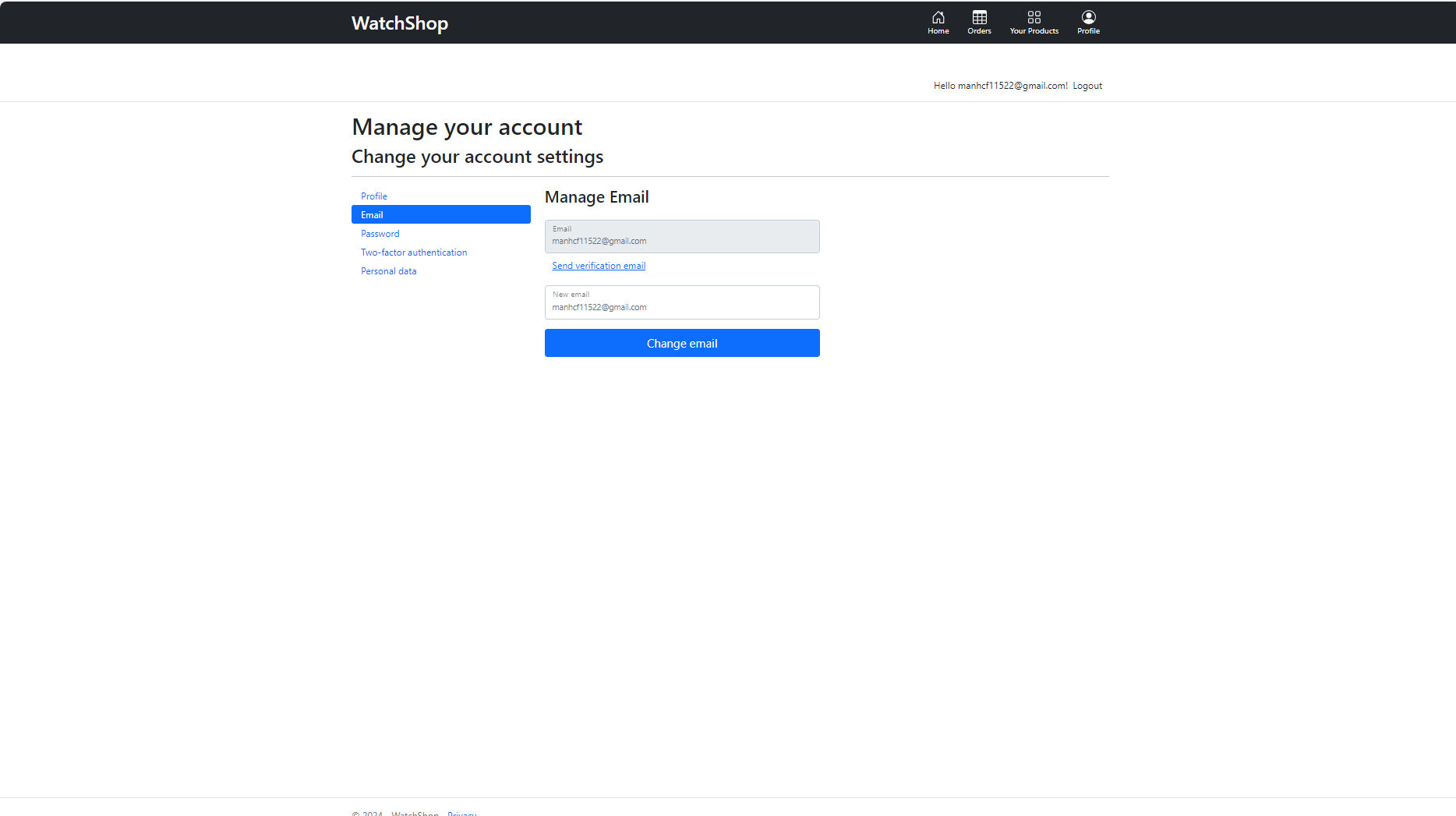


Figure 6.1.8 Manage Email

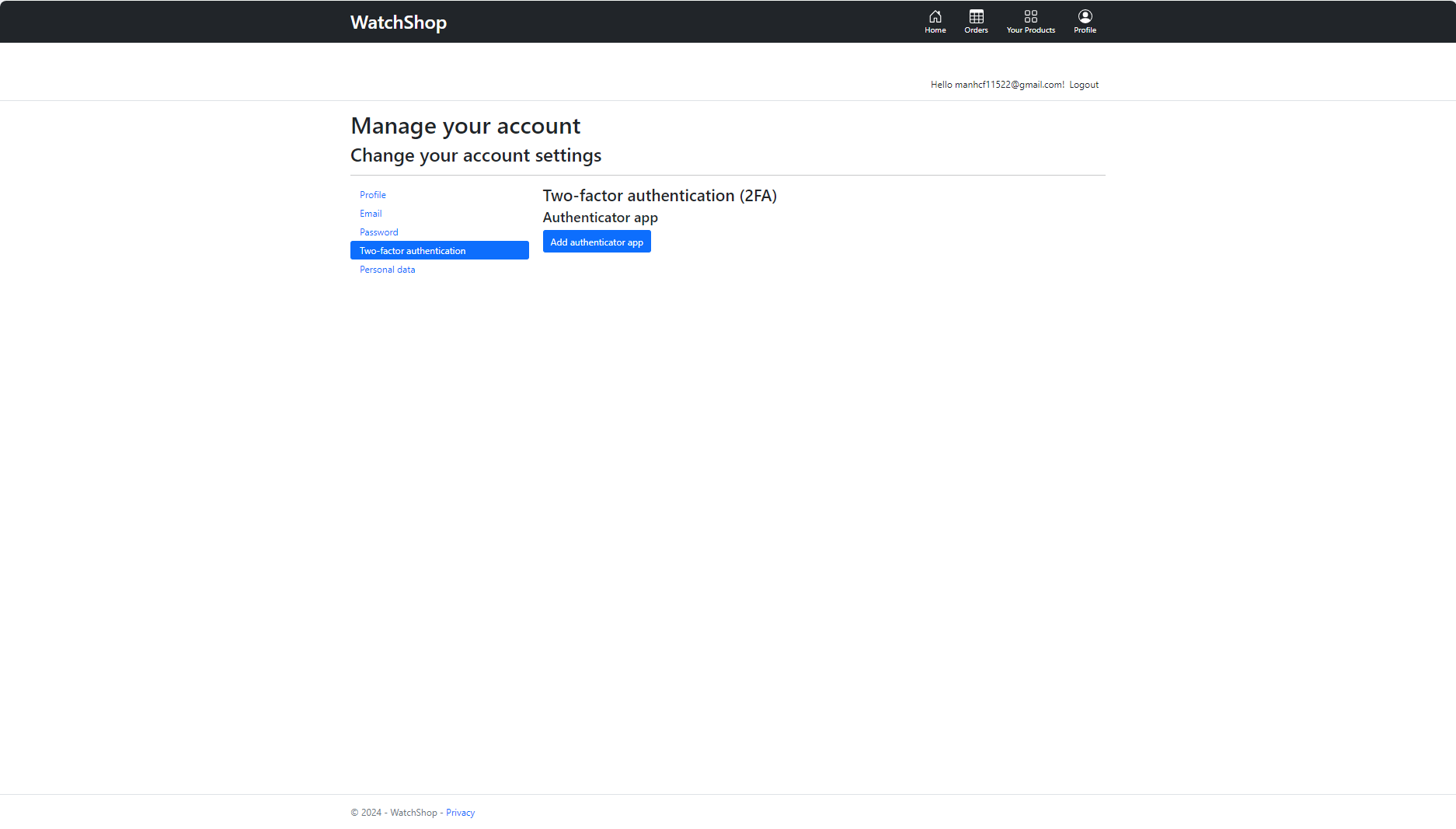


Figure 6.1.9 Two authentication Page

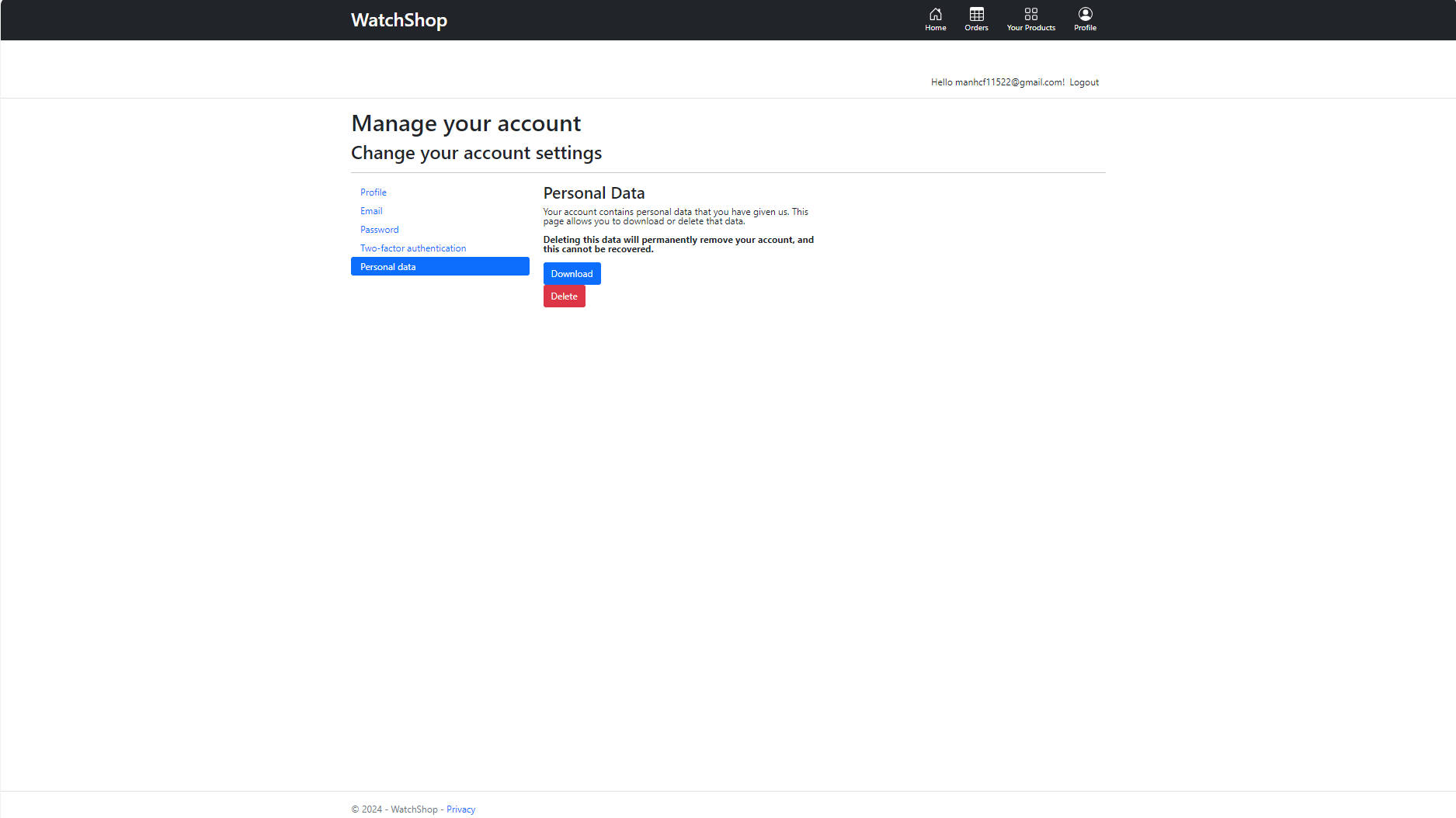


Figure 6.1.10 Personal data Page

# Class diagrams

## Basic folders for class diagram

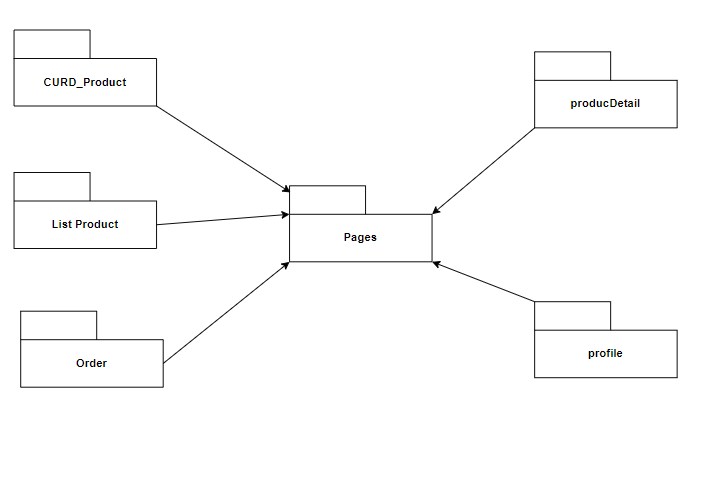


Figure 7.1.1 Basic folder for class diagram

## Class diagram

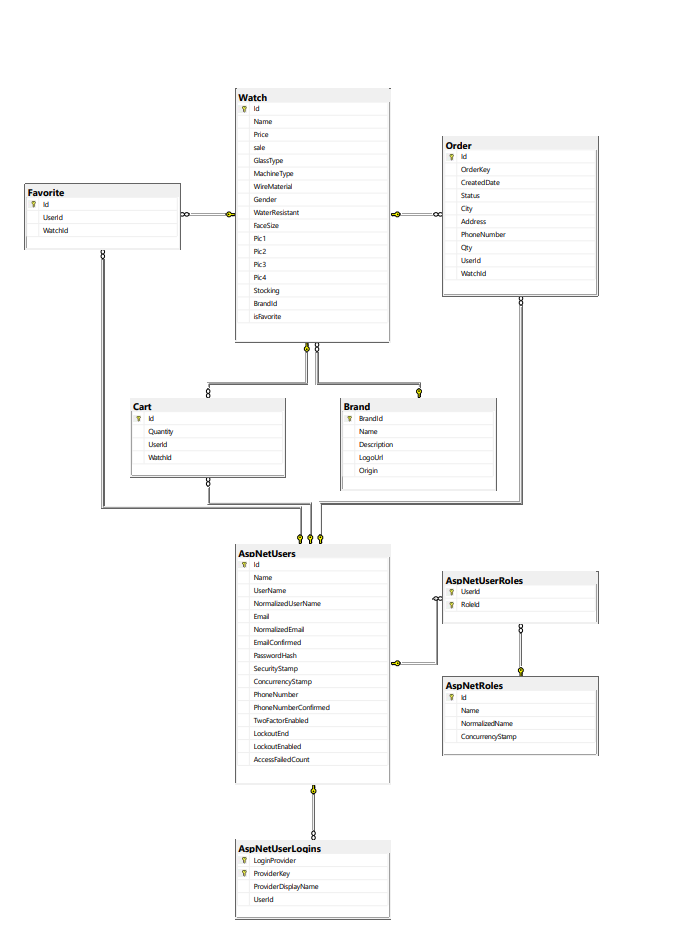
****

Figure 7.2.1 Class diagram

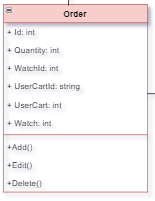
****

Figure 7.2.2 Class Order

-Description:

The Order is a class that contains product list information that the user has added to the cart.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| ID | int | Is the number of the shopping cart and is unique |
| Quantity | int | Quantity of products |
| WatchId | int | Is the serial number of the product is unique |
| UserCartId | String | As the user's name |
| UserCart | int | The number of the user who is visiting the website |
| Watch | int | Is the serial number of the Brand, may be the same |

Table 7.2.1 Class Order

## Class Watch

****

Figure 7.2.3 Class Watch

-Description:

The Class Watch is an uninformative class of a watch product.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| ID | int | Number of Product is unique |
| Name | String | Product Name |
| Price | float | Product Price |
| sale | int | Discount by item |
| GlassType | int | Number of glass type |
| MachineType | String | Name of Electromagnetic Mechanical Machine Type |
| WireMaterial | String | Material Name of Wire |
| Pic1 | String | Chain links to the position of image 1 of the product |
| Pic2 | String | Chain links to the position of image 2 of the product |
| Pic3 | String | Chain links to the position of image 3 of the product |
| Pic4 | String | Chain links to the position of image 4 of the product |
| Stocking | int | Inventory quantity of the product |
| BrandId | int | Is the serial number of the Brand, may be the same |

Table 7.2.2 Class watch

## Class Brand

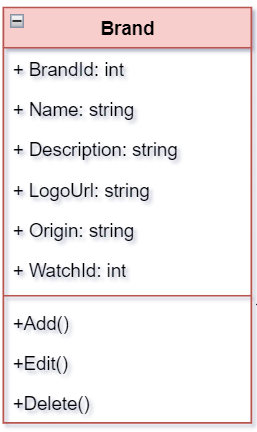
****

Figure 7.2.4 Class Brand

-Description:

The Class Brand is a class containing information about a watch brand that produces the corresponding watch.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| BrandId | int | Brand number and is unique |
| Name | String | Name of the brand |
| Description | String | Brief description of the brand |
| LogoUrl | String | Chain links to the position of image of the brand |
| Origin | String | Origin of the brand |
| WatchId | int | Brand number and is unique |

Table 7.2.3 Class Brand

## Class User cart

****

Figure 7.2.5 Class User Cart-Description:

The User Cart is an contains product information that the customer has planned to buy by adding to the cart before payment.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| UserId | int | Is the number of the shopping cart is unique |
| WatchShopUser | int | Is the number of the User is unique |
| Orders | int | Is the number of the Orders is unique |
| Watchs | String | Is the number of the Watchs is unique |

Table 7.2.4 Class user cart

## Class Watch shop user



Figure 7.2.6 Class Watch shop user

-Description:

The Watch shop user a class containing information about users of the online watch buying system. When the user registers, the information will be saved.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| Id | int | Is the number of the user is unique |
| Name | String | User's Name |
| UserCart | int | Is the number of the cart is unique |
| Favorites | int | Is the number of the Favorites is unique |
| Watchs | int | Is the number of the Watchs is unique |

Table 7.2.5 Class Watch shop user

## Class AspNetUsers

## 

Figure 7.2.7 Class AspNetUsers

-Description:

It is a table that does not have user information.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| Id | int | Is the number of the user is unique |
| Name | String | User's Name |
| UserName | String | Is the number of the cart is unique |
| Email | String | Is the number of the Favorites is unique |
| NomalizedEmail | String | Is the number of the Watchs is unique |
| EmailConfirmed | int | Email Authentication |
| PasswordHash | String | Contains encrypted passwords |
| Security Stamp | int | Security Code |
| PhoneNumber | String | User Phone Number |
| PhoneNumberConfirmed | int | Phone Number Verification |
| TwoFactorEnabled | int | Level two authentication |
| Lockoutend | int | Account Locking |
| LockoutEnabled | int | Deactivate an account |
| AccessFailedCount | int | Failed Visits |

Table 7.2.6 Class AspNetUsers

## Class Favorite

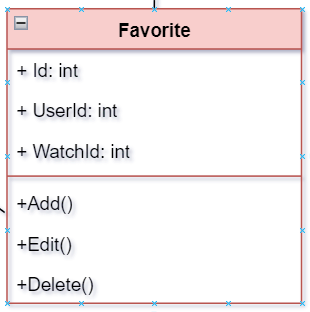


Figure 7.2.8 Class Favorite

-Description:

The class Favorite a class containing product information that users like by clicking the heart icon in the product list section.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **description** |
| Id | int | Is the number of the Favorite is unique |
| UserID | int | Is the number of the user is unique |
| watchID | int | Is the number of the Watchs is unique |

Table 7.2.7 Class Favorite

# Appendices

## Setup and Configuration

Installation:

1. Install Visual studio 2022. It is available on: https://visualstudio.microsoft.com/fr/vs
2. Include SQL Sever 2022 in installer. It is available on: https://www.microsoft.com/en-us/sql-server/sql-server-download.

## Deployment of the UUIS on the server: On the server UUIS has following structure:

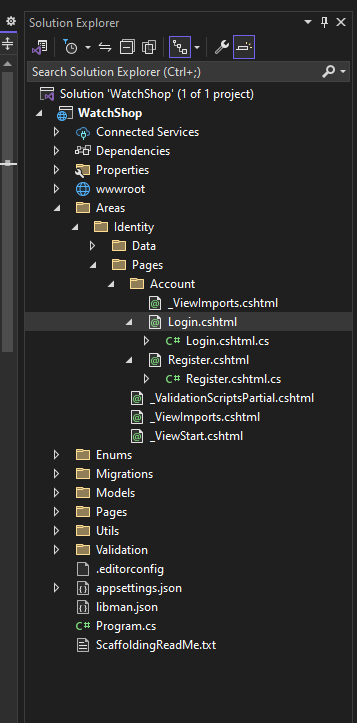


Figure 42. structure of Deployment

## Tool set and environment

**Table 3.** Tools and Environment

|  |  |
| --- | --- |
|  |  |
| Visual Studio 2022 | Programming, testing. |
| Draw.IO | UML diagrams: block diagram, DT, Sequence diagrams, Activity diagrams |
| Microsoft Office Word | Documentation |
| Microsoft Office Power Point | Presentation |
| Microsoft Office Excel | Log sheets, cost estimation |
| SQL Sever | For database management |
| e-mail | Gmail of google |
| ASP.NET CURD | Welcome pages |

Each team member contributed at least the hours indicated below, to finish the project.

**Table 5.** Logsheet

|  |  |
| --- | --- |
|  |  |
| Nguyễn Văn Mạnh | Document, class diagram, Use case, Login, Register, Forgot password, Change Password. |
| Trương Hùng Dũng | Document,Order,Mange User,Payment,Product detail, profile,Home Page, Card role base. |
| Nguyễn Hòa Hồng Đức | Document, Brand,Product,Favorited Product |

Page 82

# References

J2SE. In *Sun developer network*. Retrieved Mar. 26, 2010, from <http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Exception.html>

MCV for beginners. In chtivo. Retrieved Apr. 5, 2010, from <http://chtivo.webhost.ru/articles/mvc.php>

Samples. In *Concordia University*. Retrieved Feb. 3 and 13, 2010, from <http://users.encs.concordia.ca/~c55414/samples/>SDD. In *cmcrossroads*. Retrieved Mar. 25, 2010, from <http://www.cmcrossroads.com/bradapp/docs/sdd.html>

SDD. In *pdfqueen.* Retrieved Mar. 25, 2010, from

[http://www.pdfqueen.com/html/aHR0cDovL3dpbG1hLnZ1Yi5hYy5iZS9+c2UxXzA4M](http://www.pdfqueen.com/html/aHR0cDovL3dpbG1hLnZ1Yi5hYy5iZS9%2Bc2UxXzA4M) DkvZG9jdW1lbnRzL2ZpcnN0Y3ljbGVfY29weS9TREQucGRm

Skipfish. In *Google*. Retrieved on March 25, 2010, from <http://code.google.com/p/skipfish/>

Skipfish: An Active Web Application Security Reconnaissance Tool! In *PenTestIT*. Retrieved on March 27, 2010 from <http://pentestit.com/2010/03/20/skipfish-> active-web-application-security-reconnaissance-tool/

SkipfishDoc. In *Google*. Retrieved on March 25, 2010 from <http://code.google.com/p/skipfish/wiki/SkipfishDoc>

Skipfish Updates In *Google*. Retrieved on April 12th, 2010 from <http://code.google.com/p/skipfish/updates/list>

Std\_public. In *IEEE Standards Association*. Retrieved Mar. 25, 2010, from <http://standards.ieee.org/reading/ieee/std_public/description/se/1016-> 1998\_desc.html

User Interface Design Tips, Techniques, and Principles. In *ambysoft*. Retrieved Apr. 5, 2010, from <http://www.ambysoft.com/essays/userInterfaceDesign.html>

Visitor Pattern Versus Multimethods. In *Sourceforge.* Retrieved Mar. 26, 2010, from <http://nice.sourceforge.net/visitor.html>