

# ***Shoppeasy (SRS) Document***

**[Shop Management Web App]**

**[Shoppeasy]**

**[9/24/23]**

**[Version]**

**By: [Dragon, Samuel, Jeong]**

**[Honor Code]**

# Table of Contents

1.	Introduction	3
1.1.	Purpose	3
1.2.	Document Conventions	3
1.3.	Definitions, Acronyms, and Abbreviations	3
1.4.	Intended Audience	4
1.5.	Project Scope	4
1.6.	Technology Challenges	4
1.7.	References	4
2.	General Description	4
2.1.	Product Perspective	4
2.2.	Product Features	4
2.3.	User Class and Characteristics	5
2.4.	Operating Environment	5
2.5.	Constraints	5
2.6.	Assumptions and Dependencies	5
3.	Functional Requirements	5
3.1.	Primary	5
3.2.	Secondary	5
4.	Technical Requirements	6
4.1.	Operating System and Compatibility	6
4.2.	Interface Requirements	6
4.2.1.	User Interfaces	6
4.2.2.	Hardware Interfaces	6
4.2.3.	Communications Interfaces	6
4.2.4.	Software Interfaces	6
5.	Non-Functional Requirements	6
5.1.	Performance Requirements	6
5.2.	Safety Requirements	7
5.3.	Security Requirements	7
5.4.	Software Quality Attributes	7
5.4.1.	Availability	7
5.4.2.	Correctness	7
5.4.3.	Maintainability	7
5.4.4.	Reusability	7

5.4.5. Portability	7
5.5. Process Requirements	7
5.5.1. Development Process Used	7
5.5.2. Time Constraints	7
5.5.3. Cost and Delivery Date	7
5.6. Other Requirements	7
5.7. Use-Case Model Diagram	8
5.8. Use-Case Model Descriptions	8
5.8.1. Actor: Admin (Dragon)	8
5.8.2. Actor: Buyer (Jeong Kim)	8
5.8.3. Actor: Seller (Samuel)	8
5.9. Use-Case Model Scenarios	8
5.9.1. Actor: Admin (Dragon)	8
5.9.2. Actor: Buyer (Jeong Kim)	9
5.9.3. Actor: Seller (Samuel)	9
6. Design Documents	9
6.1. Software Architecture	9
6.2. High-Level Database Schema	9
6.3. Software Design	9
6.3.1. State Machine Diagram: Admin (Dragon)	9
6.3.2. State Machine Diagram: Seller (Samuel)	9
6.3.3. State Machine Diagram: Buyer (Jeong Kim)	9
6.4. UML Class Diagram	9
7. Scenario	10
7.1. Brief Written Scenario with Screenshots	10

# 1. Introduction (by Dragon)

## 1.1. Purpose

The Shop Management Web App is a shopping tool web app and listing aggregator designed to be an easy and simple all-around shop-and-sell tool for newcomers to online shopping and selling. The goal of this web app is to be used both by buyers and sellers to be introduced to the ability to be able to easily shop, browse, and configure items online.

## 1.2. Document Conventions

The purpose of this Shop Management Web App (SMWA) / Shoppeasy is to provide an easy web application for buyers and sellers to shop and sell. In it, we will include a way to search for products, create listings of products for its users (for both buyers and sellers), configure product specifications such as pricing, description, etc. Providing a straightforward interface that gives a clear view of the web application's features.

## 1.3. Definitions, Acronyms, and Abbreviations

Java	A programming language We will be using this language to build the shopping web app.
MySQL	The open-source relation database our program uses..
HTML	Used to design and structure our content/data for display.
SpringBoot	SpringBoot is an open-source Java-based framework used to run our application and its contents.
Incremental Development Model	The incremental development model is an architecture pattern of software engineering we are using to follow and build our web application.
Spring Web	Spring Web is included in our dependencies for our application in association with SpringBoot and web services.
Thymeleaf	Thymeleaf is used on the server side of our Java system to emphasize HTML templates.
NetBeans	NetBeans is the IDE used to develop our application/system with Java.
API	Application Programming Interface. This will be used to implement a function within the software that gets a delivery origin.

## 1.4. Intended Audience

The intended audiences of this SRS document are buyers, individual sellers, and businesses (the users), as well as ourselves (the developers). For all relatively new users, sections 1 and 2 are intended for them. For the former, the first and fifth sections of the document are most suitable for use. After section 3, is for individuals who have "advanced" knowledge in the fields of computers.

## 1.5. Project Scope

The goal of the web application is to provide an easy-to-use interface for all consumers, sellers, and other users of businesses to buy, sell, or configure products to fit their needs. This aligns with the business goals of online shopping, as a shopping web app requires an organized, fast and clear interface that can lead users to easily find whatever they want to fulfill their goals.

The benefits of the project to business include:

- Relieves stress of finding products and price limits of products and ensures they have the tools to navigate through the web application.
- Increasing pleasure to customers as they are able to communicate with sellers and admins to guarantee the best possible service.
- Increases pleasure among consumers and sellers to be able to create listings to keep products in a user's shopping cart organized and to be able to configure their listings.

## 1.6. Technology Challenges

A low-end computer should be enough to run the web application (over 300 mb of RAM should suffice). But most if not all modern computers will do just fine.

## 1.7. References

Phillip Webb, D. S. (n.d.). Spring Boot Reference Documentation.

<https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>

<https://github.com/CSC340-f23/jpa-crud-demo>

<https://github.com/CSC340-f23/security-jpa-demo>

# 2. General Description (by Samuel)

## 2.1. Product Perspective

Shopping Management Web Application (SMWA) found its origin through a person that simply doesn't have the time to go out shopping and has the desire to shop from the comforts of their home, delivered right at their doorstep. The idea was originated by a person with not much free time.

## 2.2. Product Features

The product features include the ability for users to buy and sell listings, with identification being handled through individual account creation, and also the ability for administrators to manage user accounts and listings for moderation purposes. Users can also search and sort listings, as well as see listings posted by specific users.

## 2.3. User Class and Characteristics

Our website application does not require consumers and sellers to have any prior knowledge of a computer, as we are making our features as clear and visible as possible. Although, basic knowledge of using a web browser and one's own knowledge of shopping is advised. The website application disables the need to have prior knowledge of shopping online, as the visible features will guide them easily.

## **2.4. Operating Environment**

The web application is designed to operate on all web browsers across many computers (desktop, laptops, etc.).

## **2.5. Constraints**

Full use of the API implementation proved to be difficult during the development of our software, but the default settings of the API were able to be implemented fortunately.

## **2.6. Assumptions and Dependencies**

The web application will be dependent on Spring Web, SpringBoot and the OSRM API in order to create and run the application that will be developed within NetBeans. The application will use source code in Java and HTML to build our web application.

# **3. Functional Requirements (by Samuel)**

## **3.1. Primary**

- FR0: The system will allow the user to generate a listing from entered data, including name, price point, method of exchange, display image, and tags.
- FR1: The system will allow the user to search through listings generated by other users, categorizing them either chronologically, by tag, or by user.
- FR2: The system will allow the user to send a purchase request to another user who has generated a listing.
- FR3: The system will allow users who have interacted through a purchase request to contact each other through the use of a messaging interface.
- FR4: The system will allow administrators to manage the information stored on user accounts.

## **3.2. Secondary**

- Account authorization setup so that users cannot alter each others' listings and orders. As well as to not allow access to certain pages an account does not have access to.
- Logged pricing history to ensure that a user is charged only the price of an item at the time of purchase
- User Id is different and not changeable to avoid duplicate user accounts.

# **4. Technical Requirements (by Dragon, Sam, Jeong)**

## **4.1. Operating System and Compatibility**

The application will be compatible with any operating system that is able to view and to interact with traditional web pages.

## **4.2. Interface Requirements**

### **4.2.1. User Interfaces**

The application will have buttons to interact with between the users and our software include:

- Home Page (Title at top of screen with a name):
  - A navigation bar with respective actors and use-cases that results from functions and buttons from “Buyer”, “Seller”, and “Admin”.
  - “Manage listing”/”Manage shopping cart” button
  - “login” function that displays the message is username/password is incorrect.
  - “Logout” function/button that logs users out.
  - A cart button to see items added to cart.
  - “Add product” button under products.
- Manage Listing Page: Seller
  - “Add a product” button that allows sellers to enter their product information.
  - A “edit” button to edit the product (price, name, etc.) that was already submitted by the seller.
  - A navbar displaying tabs such as home, admin, seller, contact, about us, cart, and logout.
- Listing Page: Buyer
  - A “Add to wishlist” button to add items to a list that can be referred to later.
  - A “Add to cart” button that will add items selected to the cart page.
  - “Remove item” button that will remove the product from the cart.
  - A navbar displaying tabs such as home, admin, seller, contact, about us, cart, and logout.
- Admin Page (Title displayed at top of screen):
  - View user button, will display user information.
  - Under the “view user” button, there will be a “delete” user button, giving an alert before performing the function.
  - “Change roles” button to change a user’s roles for permissions of another role.
  - A navbar displaying tabs such as home, admin, seller, contact, about us, cart, and logout.

#### **4.2.2. Hardware Interfaces**

The application will be able to run on any device that has internet access such as smartphones, desktop computers, laptops and tablets.

#### **4.2.3. Communications Interfaces**

The application will utilize the HTTP protocol to connect to a web API and fetch info.

#### **4.2.4. Software Interfaces**

HTML, CSS to build the front end of our web application and using Java to support our back end code. Also using list libraries/APIs to hold and retrieve data for products/items and users, as well as maps for login credentials.

## **5. Non-Functional Requirements (by Samuel, Dragon, Jeong)**

### **5.1. Performance Requirements**

- NFR0(R): Once the user has inputted all necessary data, creating and managing a listing will take no longer than thirty seconds to complete.
- NFR1(R): The listing search algorithm will return the current results within ten seconds, regardless of how much content has been sorted
- NFR2(R): A given listing's history will never be greater than twenty entries. Any entries older than this will be culled.

### **5.2. Safety Requirements**

- NFR3(R): Create a listing history so that any price changes to a listing after an order is made don't change the order
- NFR4(R): User information corresponds to what pages a user has access to.

### **5.3. Security Requirements**

- NFR5(R): No user will be able to access the purchasing history or details of another user
- NFR6(R): Users' passwords are not displayed to a other users including administrators, unless a user explicitly reports to an administrator

### **5.4. Software Quality Attributes**

#### **5.4.1. Availability**

The software should be available to anyone with a personal computer and an internet connection, and the ability to purchase listings should be available to anyone with an account.

#### **5.4.2. Correctness**

When given two identical search queries, unless a new listing has been added, the software should provide precisely the same results to the search.

#### **5.4.3. Maintainability**

The software should be able to run without intervention, even when errors occur, and should be built in such a way that new features can be added to modular elements (such as available listing fields, the searching algorithm, etc.) without altering the existing features.

#### **5.4.4. Reusability**

Features should be built in a modular fashion, so that if a new view is added that requires them it may merely incorporate the functionality to access them that has already been constructed.

#### **5.4.5. Portability**

The software should be shipped as Java jars, so it can be run on any system with Java installed.

### **5.5. Process Requirements**

#### **5.5.1. Development Process Used**

Incremental Development Model

#### **5.5.2. Time Constraints**

Workload of multiple classes in folders, availability of group members.

#### **5.5.3. Cost and Delivery Date**

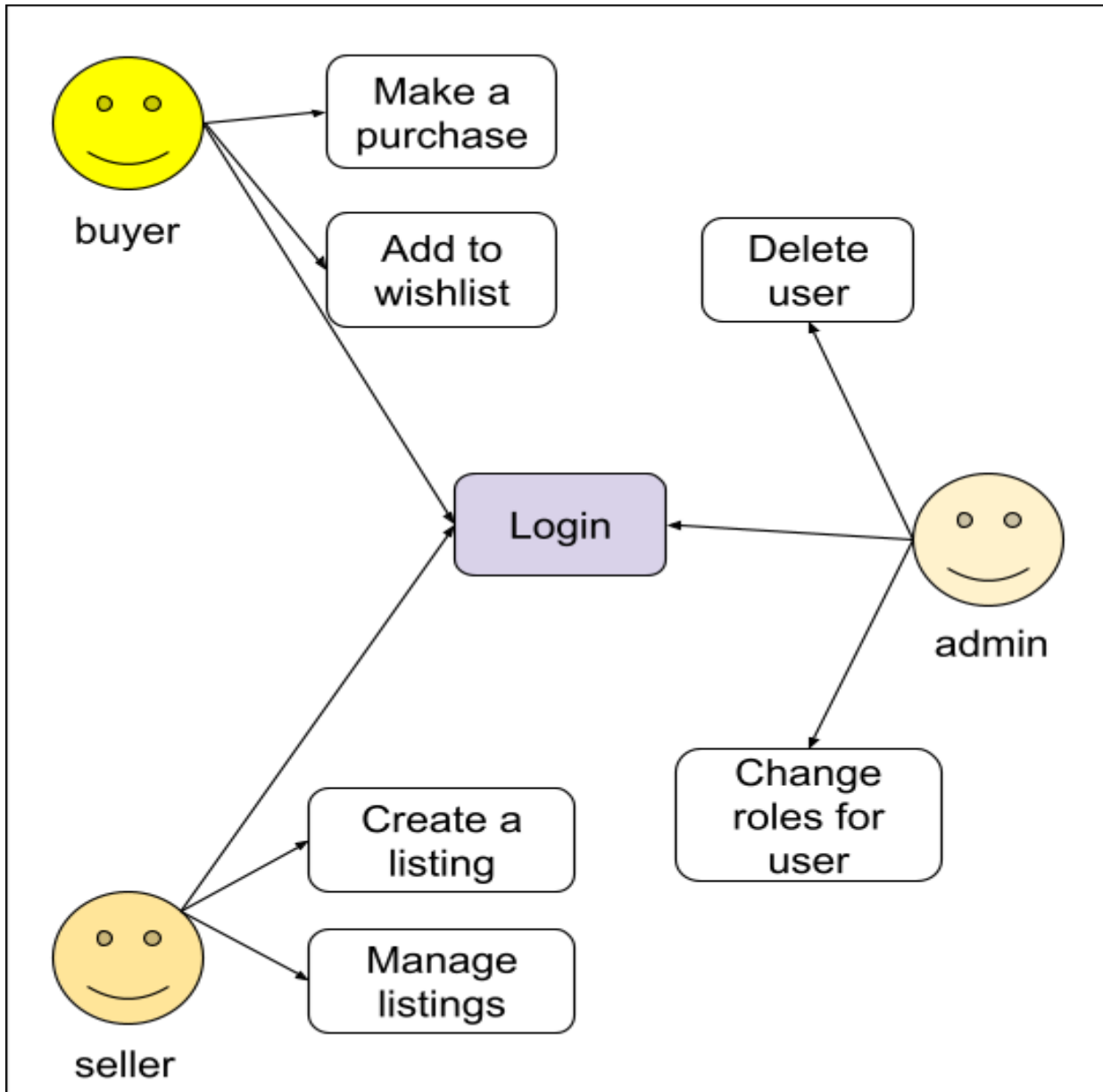
Cost of 0\$ and Delivery date is on December 5th, 2023



## 5.6. Other Requirements

XAMPP Control Panel  
MySQL

## 5.7. Use-Case Model Diagram



## 5.8. Use-Case Model Descriptions

### 5.8.1. Actor: Admin (Dragon Phiansin)

- **Login:** Username and password are inputted to identify a specific admin.
- **Delete User(s):** The admin may delete/ban users accordingly, based on user permissions or based on user behavior.
- **Change Role(s):** The admin can change the role of users (buyer, seller, admin) for certain permissions only that specific user has.

### 5.8.2. Actor: Buyer (Jeong Kim)

- **Login:** Username and password are inputted to identify the user as a buyer
- **Make a purchase:** The buyer may purchase an item according to their needs with a chosen payment method.
- **Rate the purchase:** The buyer can give a rating of an item that they purchased on a scale of 1 to 10.

### 5.8.3. Actor: Seller (Samuel)

- **Login:** Username and password are inputted to identify the current user as a specific seller.
- **Create Listing:** By inputting name, price point, exchange method, and optional tags and images, a seller can create a listing for a product that is visible to all buyers.
- **Manage Listing:** Previously generated listings can have fields edited, with earlier forms of the listing being stored for posterity.
- **Discount Listing:** The seller may apply time- or buyer-specific discounts to the price point of a listing, with automated creation, management, and removal after the time or purchase has elapsed.

## 5.9. Use-Case Model Scenarios

### 5.9.1. Actor: Admin (Dragon Phiansin)

- **User-case Name: Delete User**
  - **Initial Assumption:** A user asks for login credentials (user account) to be deleted or the user does not act according to code of conduct.
  - **Normal:** Admin will delete a user's login credentials (user account) corresponding to the user's behalf
  - **What Can Go Wrong:** The user's login credentials are not deleted or are temporarily deleted.
  - **Other Activities:**
  - **System State on Completion:** The user's account is deleted.
- **User-case Name: Change Roles**
  - **Initial Assumption:** The admin has changed a user's role from buyer to admin. The role key in the database has changed accordingly and the user that was a buyer now has the role of admin and its permissions.
  - **Normal:** A user's role is changed to another role.
  - **What Can Go Wrong:** A user's role is changed but the user cannot access certain permissions that were meant for that role.
  - **Other Activities:** The admin can change roles to all users as well as themselves.
  - **System State on Completion:** A user's role is changed and they now have the permissions of the changed role.
- **Use-Case Name: Login**
  - **Initial Assumption:** The admin has a registered account to login. That account is stored in the application database. The admin has access to the username and password to access their account.
  - **Normal:** The admin will enter the username and password and be granted access to their account.
  - **What Can Go Wrong:** The username or password may fail to match the stored username and password hash, preventing the admin from accessing their account.

- **Other Activities:** The admin can reset their account password with a forgot-password page.
- **System State on Completion:** The admin is logged in and can view their and other user's information page, as well as their listings aggregators.

### 5.9.2. Actor: Buyer (Jeong Won Kim)

- **Use-Case Name:** Login
  - **Initial Assumption:** The system asks for the username and password of the buyer and the buyer logs in with the username and the password they signed up with.
  - **Normal:** The user will enter the username and password and have access to their account.
  - **What Can Go Wrong:** The username or password may fail to match the stored username and password.
  - **Other Activities:** The buyer can reset their account password with a forgot-password page.
  - **System State on Completion:** The buyer is logged in and can view their user profile, items that are currently listed to be sold and their purchase history.
- **Use-Case Name:** Make a purchase
  - **Initial Assumption:** The buyer is logged in to their account and has the listing of their interest opened.
  - **Normal:** The buyer is able to make a purchase by pressing the purchase button and choosing a method of payment.
  - **What Can Go Wrong:** A buyer can make an accidental purchase.
  - **Other Activities:** Items can be added to the cart before making a purchase.
  - **System State on Completion:** Item is purchased and the confirmation message gets sent to both buyer and seller .
- **Use-Case Name:** Add item to wishlist
  - **Initial Assumption:** The buyer has an item they like but not purchasing yet.
  - **Normal:** The buyer saves item(s) to the wishlist.
  - **What Can Go Wrong:** An item does not get added to the wishlist.
  - **Other Activities:** Remove items from the wishlist they no longer want.
  - **System State on Completion:** The buyer can view their wishlist with items that they saved.

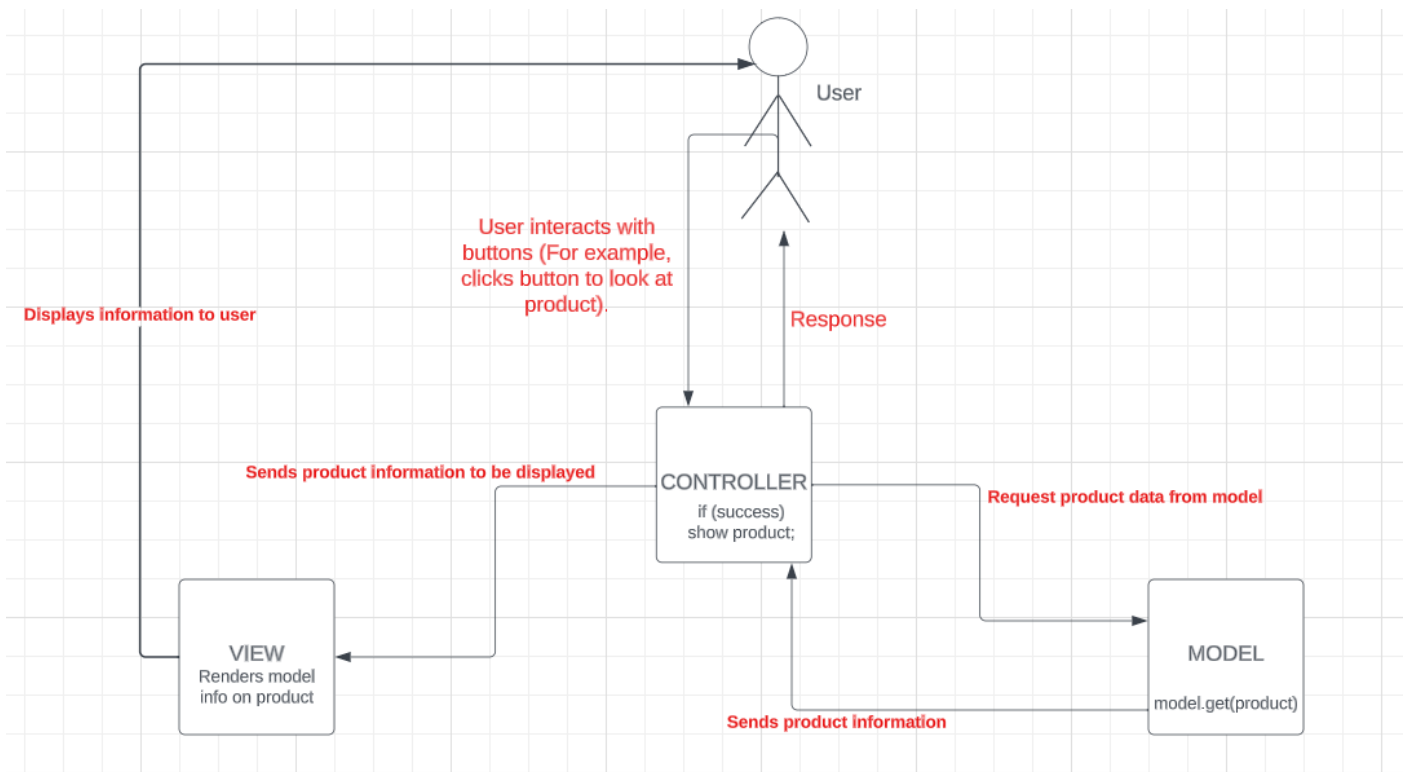
### 5.9.3. Actor: Seller (Samuel)

- **Use-Case Name:** Login
  - **Initial Assumption:** The seller has a registered account to login. That account is stored in the application database. The seller has access to the requisite username and password to access the account.
  - **Normal:** The seller will enter the username and password and be granted access to their account.
  - **What Can Go Wrong:** The username or password may fail to match the stored username and password hash, preventing the seller from accessing their account.
  - **Other Activities:** The seller can reset their account password with a forgot-password page.
  - **System State on Completion:** The seller is logged in and can view their user information page and listings aggregator.
- **Use-Case Name:** Create Listing

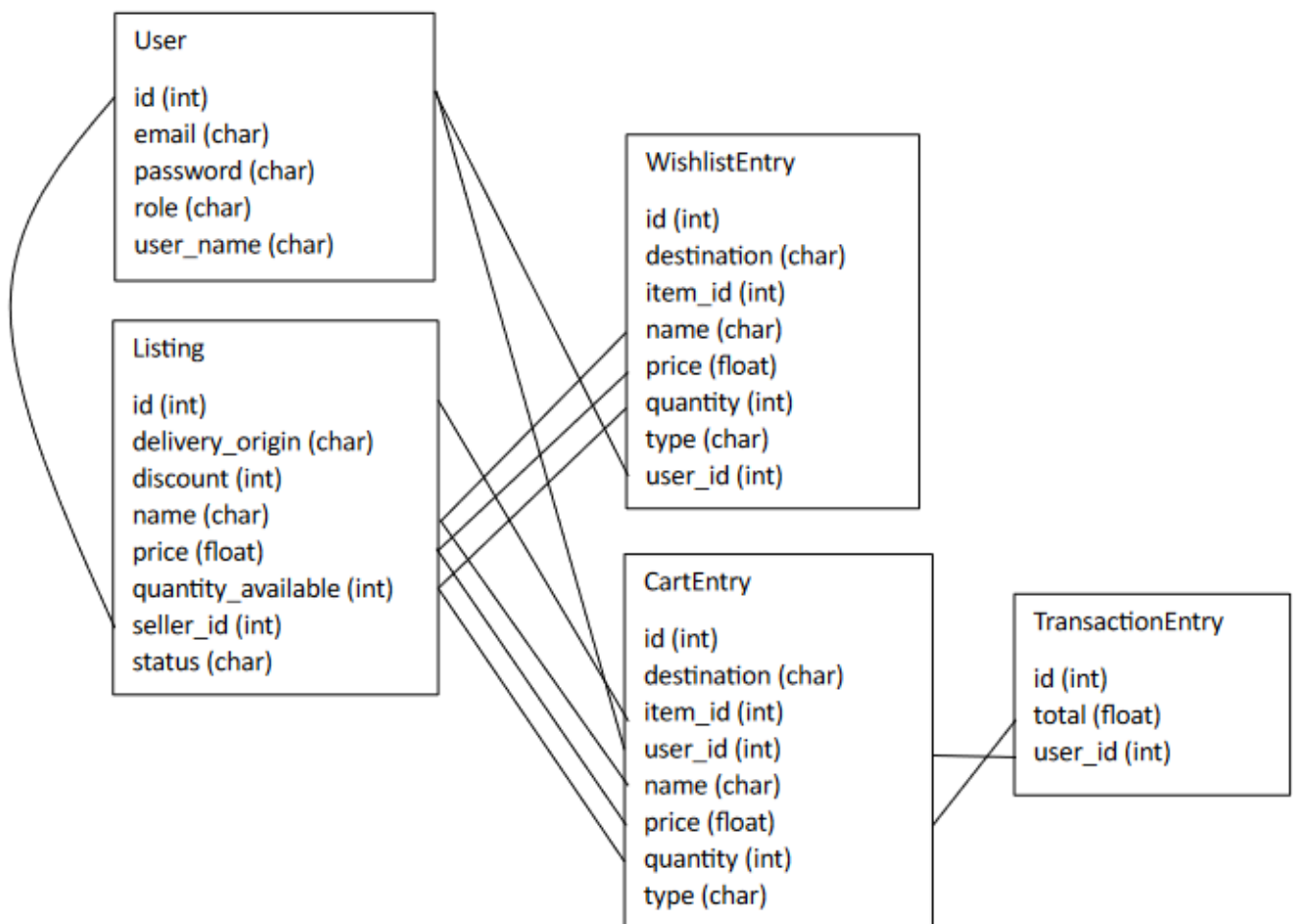
- **Initial Assumption:** The seller is logged into their account.
  - **Normal:** The seller inputs the necessary information to generate a listing, which is then added to the database.
  - **What Can Go Wrong:** The listing can fail to be created, in which case we should remain on the current page for the seller to try again.
  - **Other Activities:** A time for a listing to be made visible to all users can be set.
  - **System State on Completion:** The listing is now visible through both the search interface and the seller's listings page.
- **Use-Case Name:** Manage Listing
    - **Initial Assumption:** The seller is logged into their account, which has an active listing.
    - **Normal:** The seller can alter single elements of the listing without recreating it or altering the other elements.
    - **What Can Go Wrong:** The listing can fail to update.
    - **Other Activities:** The listing can be deleted altogether through a batch edit button.
    - **System State on Completion:** The listing now shows the changes on all pages, as well as a link to show previous versions of the same listing.
  - **Use-Case Name:** Apply Discounts on Listing
    - **Initial Assumption:** The seller is logged into their account, which has an active listing. In the case that the discount is buyer-specific, the seller must also know the username of the buyer account.
    - **Normal:** The seller can set a percentage or exact discount to a listing.
    - **What Can Go Wrong:** The discount can fail to be set, or the time can be set incorrectly.
    - **Other Activities:** Batch listing discount; i.e., a discount can be applied to a group or all listings created by the seller.
    - **System State on Completion:** A discount is visible on the listing for either all users for the duration set, or for one specific user until the order is completed.

## 6. Design Documents

### 6.1. Software Architecture (by Dragon)

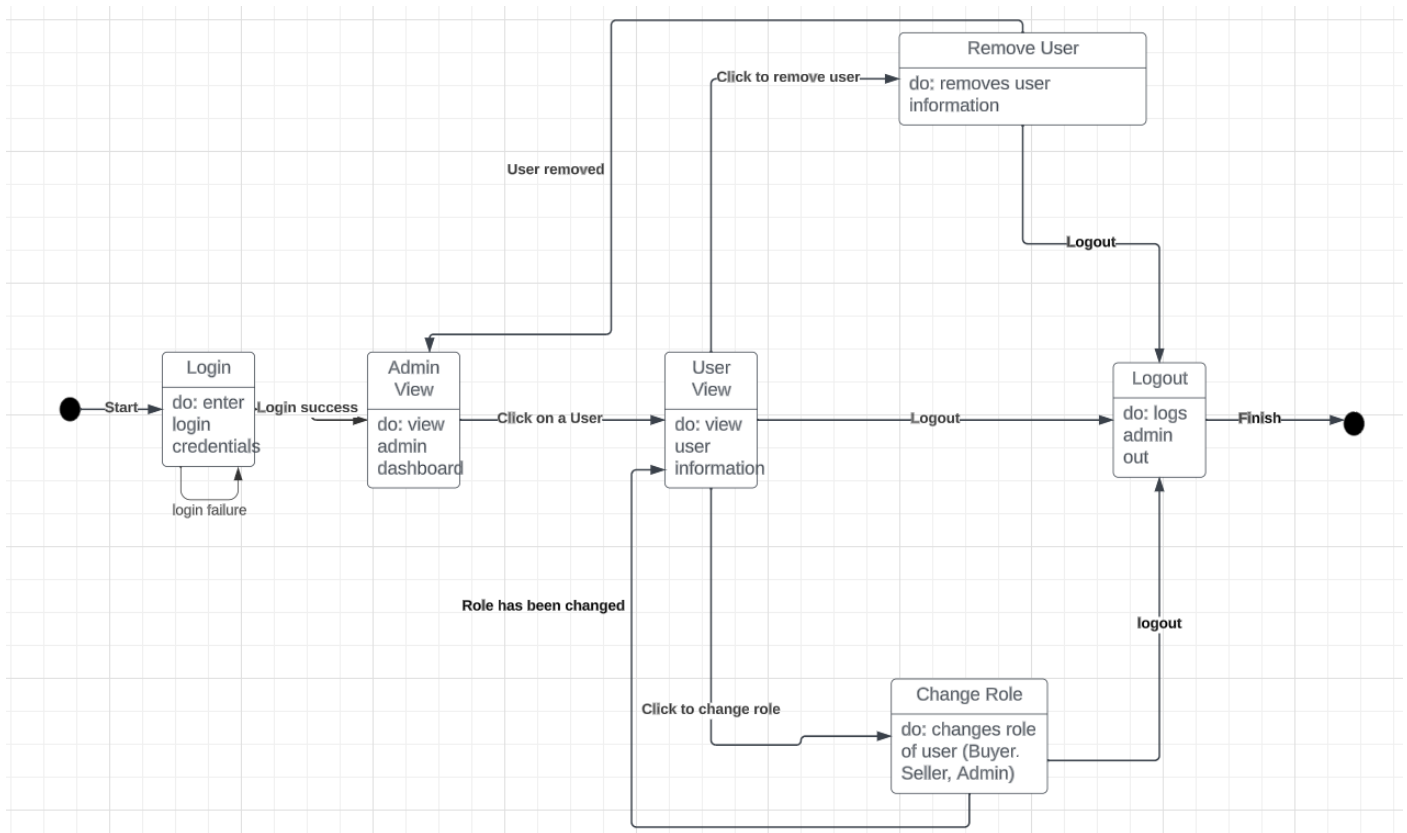


## 6.2. High-Level Database Schema (by Samuel)

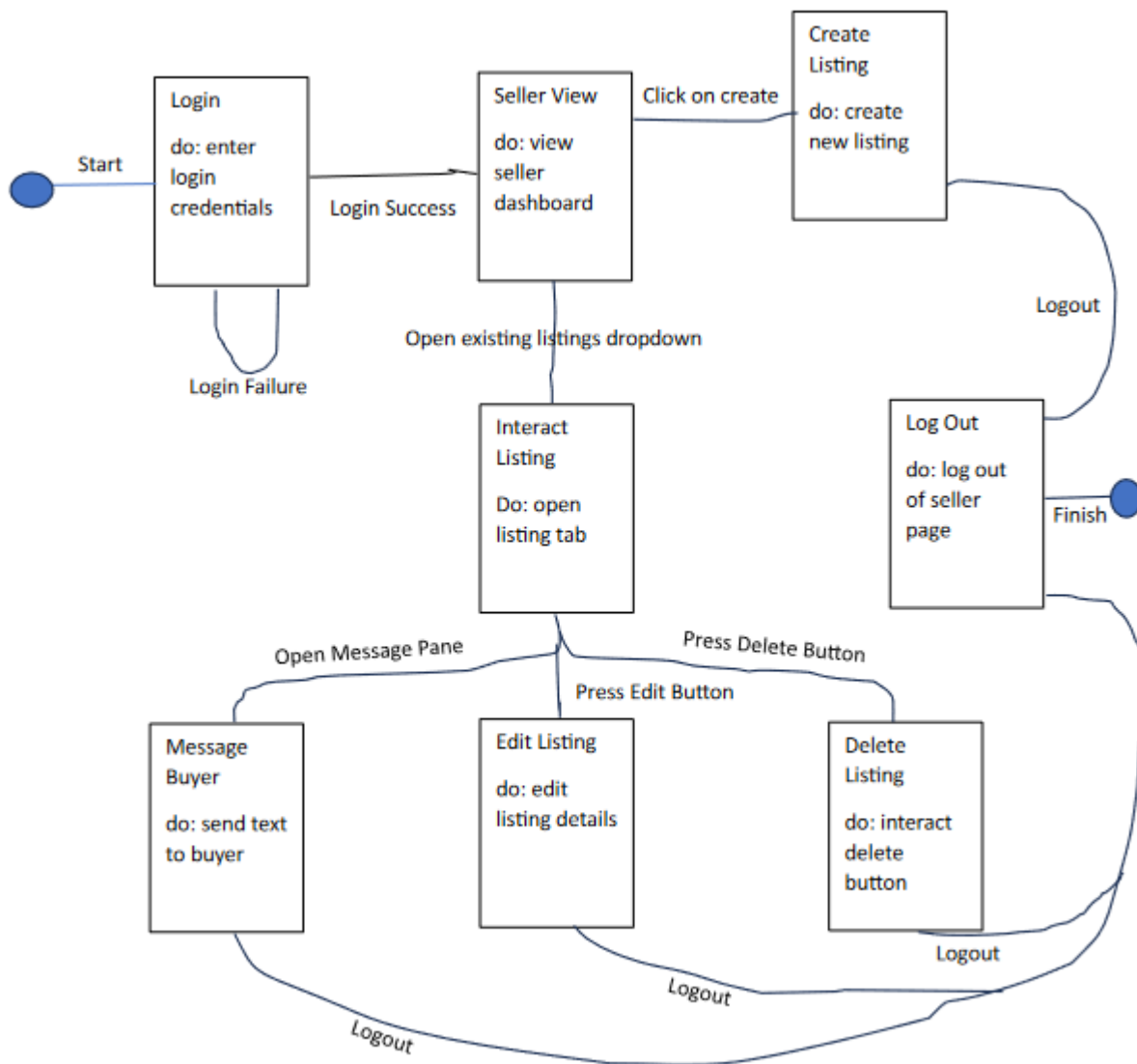


## 6.3. Software Design

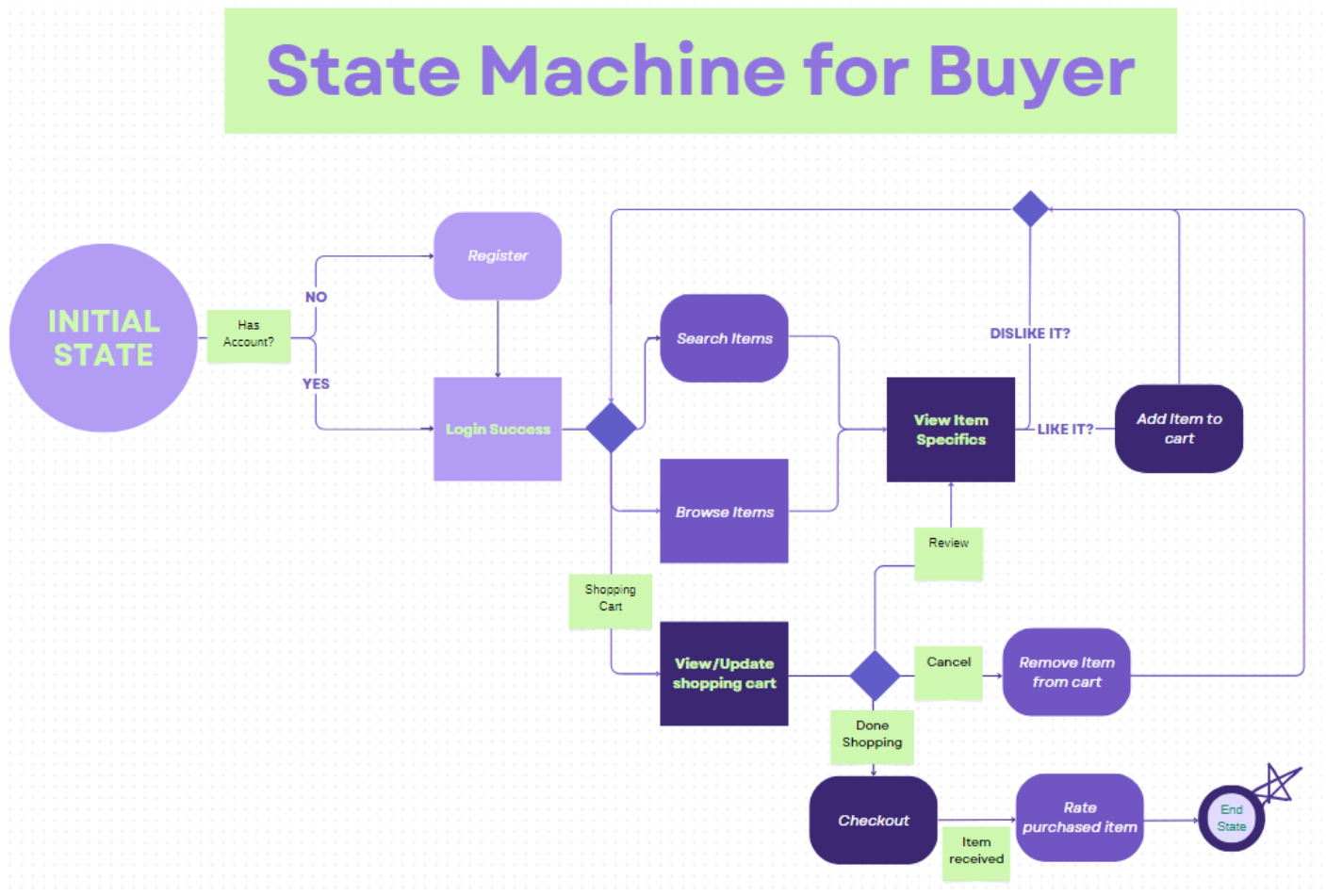
### 6.3.1. State Machine Diagram: Admin (Dragon Phiansin)



### 6.3.2. State Machine Diagram: Seller (Samuel Johnson)

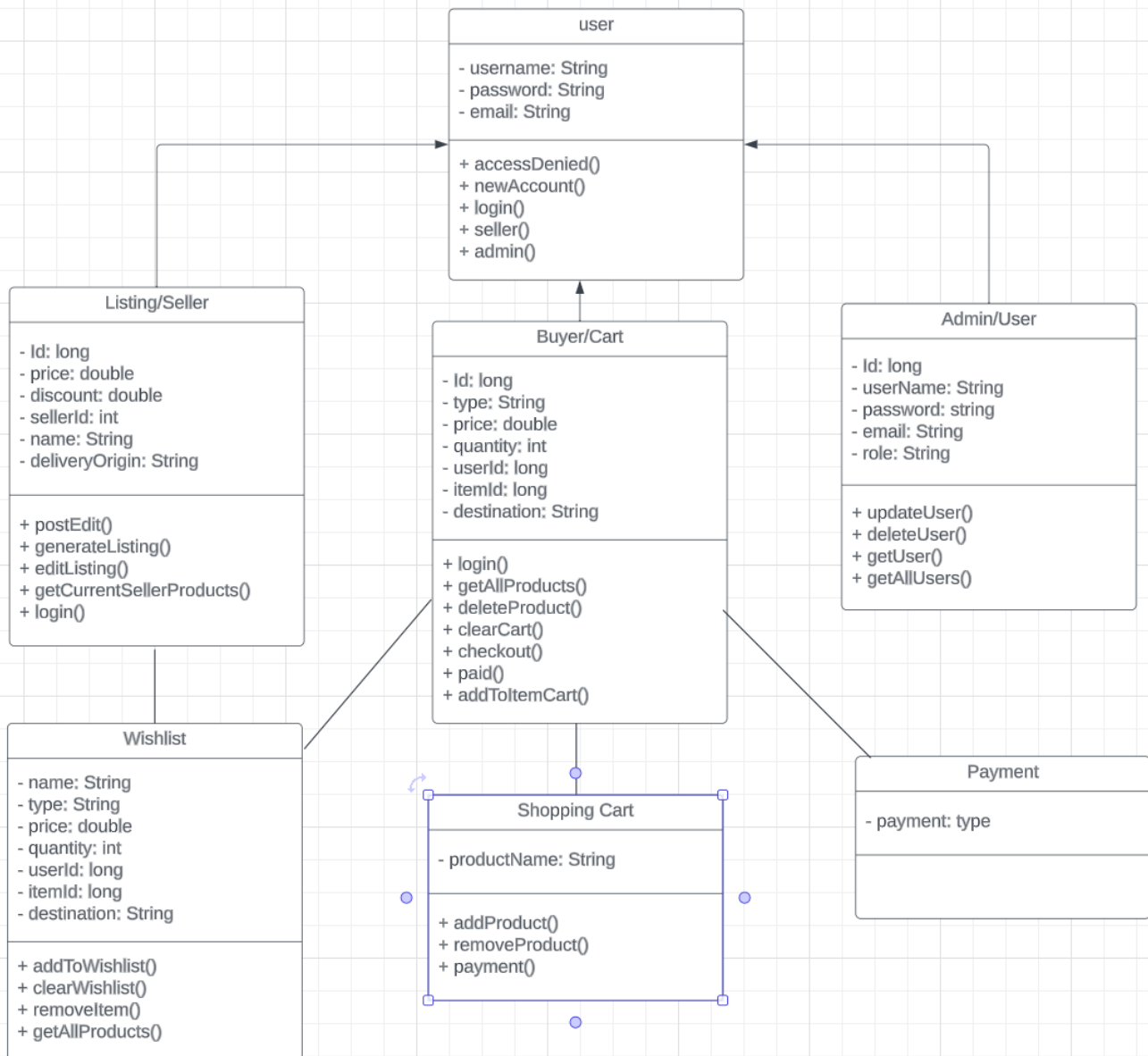


### 6.3.3. State Machine Diagram: Buyer (Jeong Won Kim)





## 6.4. UML Class Diagram (by Dragon)



## 7. Scenario

### 7.1. Brief Written Scenario with Screenshots

**7.1.1.** Buyer: I want to add items to my shopping cart so that I can review and proceed to checkout. I also want to be able to add items that I may purchase in the future to my wishlist.

1. Preconditions:

- The user is logged in as a registered customer.
- The user has navigated to the product listing page.

- Adding items:
  - The user should be able to view a list of products on the listings page.
  - Each product should display its name, price, “Add to Cart”, and a “Wishlist” button.
  - The user can click the “Add to Cart” button to add the selected item to their shopping cart, and can click the “Wishlist” button to save the selected item to their shopping cart.

[Home](#)

[Logout](#)
[Admin](#)
[Seller](#)
[About](#)
[Contact](#)
[Cart](#)

Shoppeasy

Premium brand mall

Est. 2020

Your Cart

My Wishlist

Name	Type	Quantity	Price	Action
Laptop		1	\$498.89	<div>Remove</div> <div>Move to Wishlist</div>

Clear All

Go to checkout

- Cart interaction:
  - The user can view the contents of their shopping cart by clicking the cart nav.
  - The cart page should display a list of added items with details such as product name, price, and quantity.
  - The user can remove items they no longer want from the shopping cart.
- Checkout process:
  - The user can proceed to checkout directly from the cart page.
  - The system should prompt a user to enter their address, cart info, and review the order summary.
  - The user should be able to submit the order for processing.

## Checkout

[Home](#) [Cart](#)

### Billing address

First name

Last name

Email

Address

### Your cart

Total (USD)

Promo code

Redeem

### Payment

☒ VISA

☐ MASTER CARD

☐ AMEX

Name on card

Credit card number

Expiration

CVV

Checkout


## Order Confirmation

[Home](#) [Order Confirmation](#)

# Thank you for your order!

### 7.1.2. Seller: I want to create and edit existing listings.

1. Seller2 logs in for the first time and creates a profile with values (email, username, password, role). They go to their main page and create a listing with values (name, price). Seller2 exits.


[Home](#)   [Logout](#) [Admin](#) [Seller](#) [About](#) [Contact](#) [Cart](#)

## Seller

Name:

Price:

Delivered From:

[Home](#)   [Logout](#) [Admin](#) [Seller](#) [About](#) [Contact](#) [Cart](#)

## Seller

Macbook Air

1199.99

Name:

Price:


Delivered From:

2. Seller1 logs in, views their listings from the profile, edits it with a new price, and then goes to their profile and search bar to ensure they have changed. Seller1 exits.

### 7.1.3. Admin: I want to delete users and change user roles.

1. Admin1 logs in, views all users, selects “buyer2” by ID and their information (ID, username, email, role).

[Home](#)


 [Logout](#) [Admin](#) [Seller](#) [About](#) [Contact](#) [Cart](#)

# Administration

Premium brand mall  
Est. 2020

ID #	Username	Email	Role
1	admin1	admin@domain.com	ADMIN
2	seller1	sdf@sdf	SELLER
3	buyer1	buyer@place.com	BUYER
5	buyer2	buyer2@place.com	BUYER

[Home](#)

 [Logout](#) [Admin](#) [Seller](#) [About](#) [Contact](#) [Cart](#)

# Administration

Premium brand mall  
Est. 2020

## User Configuration

### User Information

ID: 5

Username: buyer2

Email: buyer2@place.com

Role: BUYER

[Edit Role](#) [Delete](#) [Go Back](#)

2. “Admin1” changes “buyer2” role to seller in according to “buyer2”.

# Administration

Premium brand mall

Est. 2020

## Update User Role

ID:

Username:

Email:

Role:

3. “Buyer2” logs in and has access to the seller page and it’s permissions.

# Shopeasy Login

Premium brand mall

Est. 2020

Username

Password

.....

You have been logged out.

[Create New Account](#)

[Home](#)



[Logout](#)

[Admin](#)

[Seller](#)

[About](#)

[Contact](#)

[Cart](#)

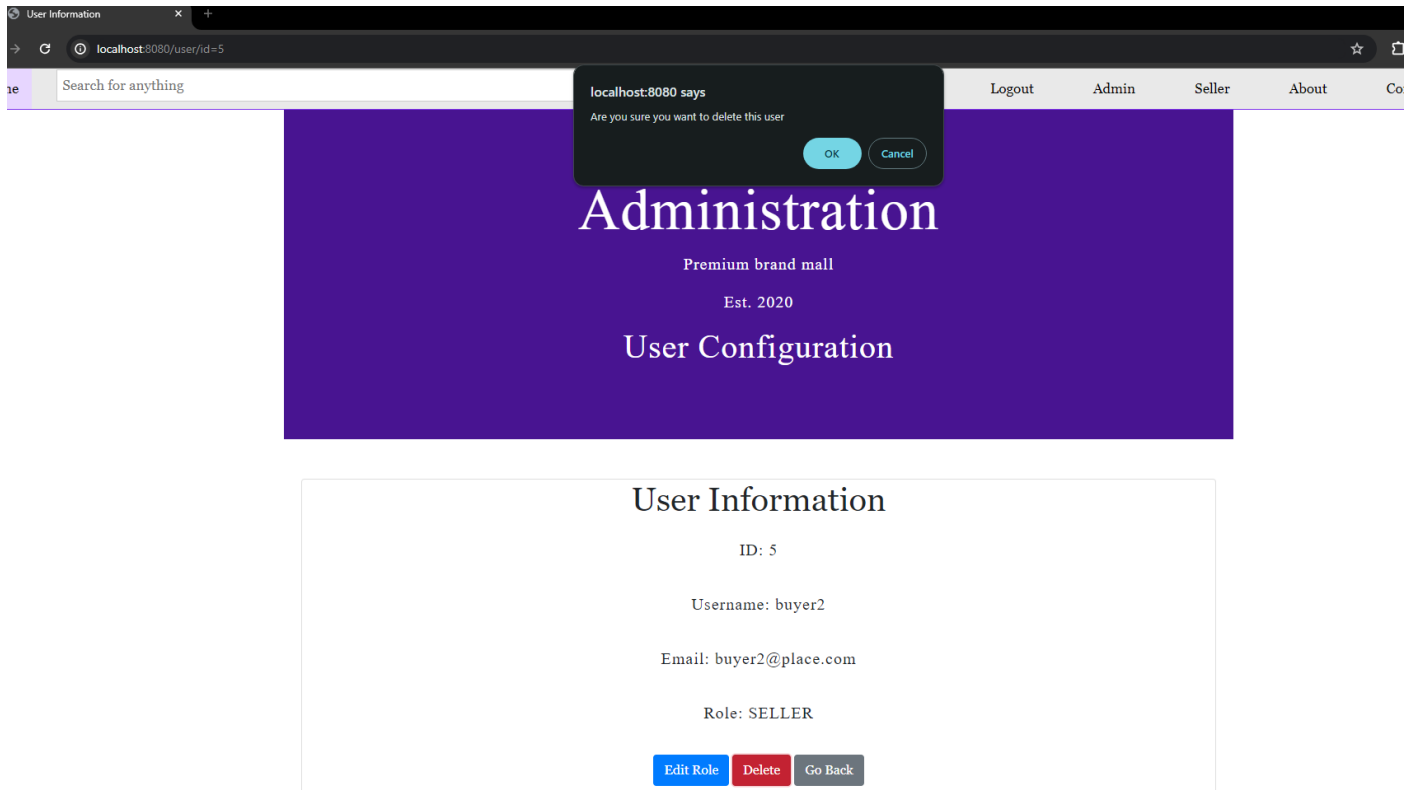
**Seller**

Name:

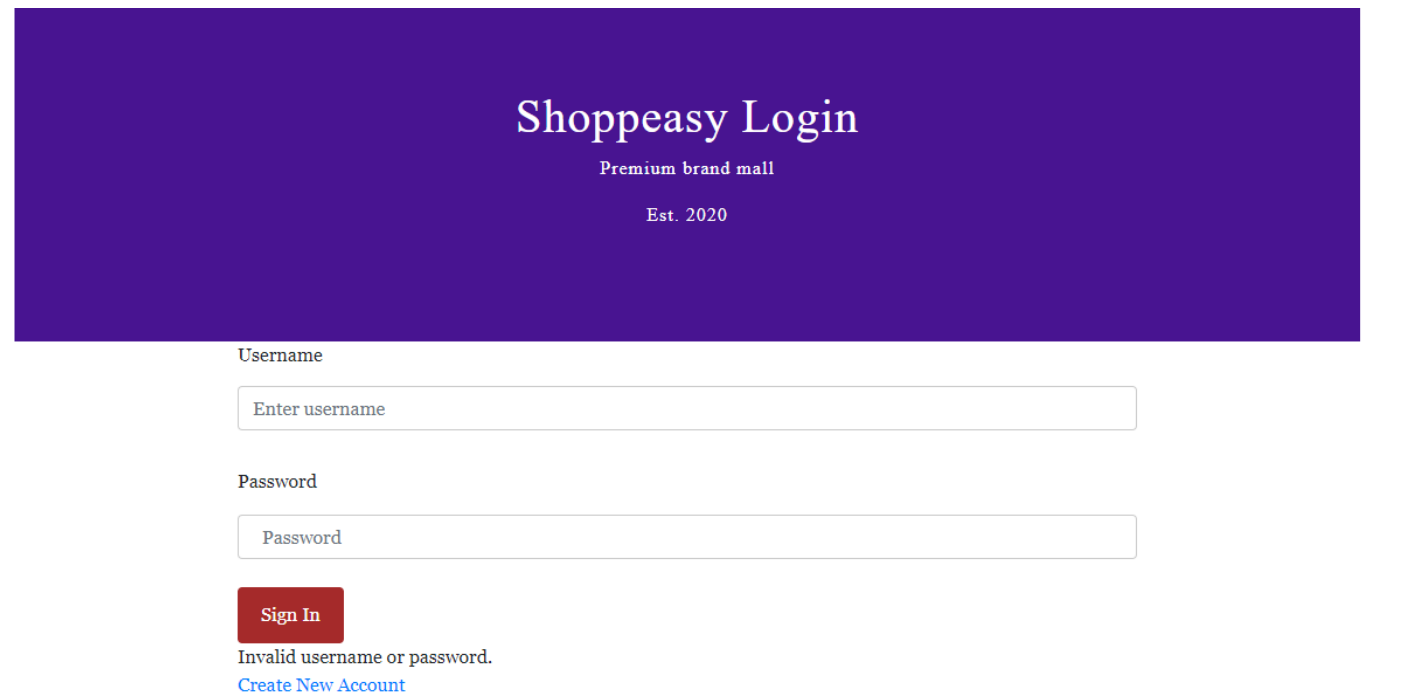
Price:

Delivered From:

4. Admin1 logs in, views all users, detects inappropriate activity of “buyer2” and selects “buyer2” by ID, and deletes “buyer2” from the database.



5. “Buyer2” tries to login but cannot log in.



Preconditions for Admin:

- The user is logged in as admin.

- The user has navigated to the user listings page.

“userListings” page interaction:

- The user listings page should display a list of all users registered with their name and email.

- Each user listing should show user ID, username, email, and their role.

- The user can click on user ID and change their role by clicking “Edit Role” or delete a user by “Delete” button.