INTERNATIONAL IT TRAINING SYSTEM

APTECH DA NANG

----🙠🕮🙢----



REPORT

COURSE PROJECTS

Topic:

ONLINE AUCTION



Instructor: Le Thien Nhat Quang

Students: Nguyen Ngoc Ky Phong (Leader)

Nguyen Van Quoc

To Anh Tu

Nguyen Trung Nghia

*Da Nang, March 11, 2024*

TABLE OF CONTENTS

[PREFACE 5](#_Toc160879715)

[Part 1: INTRODUCTION 6](#_Toc160879716)

[1.1 Introducing the project 6](#_Toc160879717)

[1.2 Project goals 6](#_Toc160879718)

[1.3 Object and scope of the study 7](#_Toc160879719)

[1.3.1 Subjects 7](#_Toc160879720)

[1.3.2 Scope of research 7](#_Toc160879721)

[1.4 Research Methods 8](#_Toc160879722)

[1.4.1 Collection method 8](#_Toc160879723)

[1.4.2 Information processing methods 8](#_Toc160879724)

[Part 2: CONTENTS 9](#_Toc160879725)

[CHAPTER I: SURVEY, DETERMINATION OF REQUIREMENTS AND IDEAS 9](#_Toc160879726)

[1. Survey 9](#_Toc160879727)

[1.1 Survey purpose 9](#_Toc160879728)

[1.2 Trends and overall market situation: 9](#_Toc160879729)

[1.3 Market survey 10](#_Toc160879730)

[1.4 User needs 11](#_Toc160879731)

[1.5 Survey scope 13](#_Toc160879732)

[1.6 Evaluate the infrastructure of current auction websites 14](#_Toc160879733)

[2 Determine requirements 15](#_Toc160879734)

[2.1 General system requirements 15](#_Toc160879735)

[2.2 Specific requirements for each function 16](#_Toc160879736)

[2.2.1 Content of the Home Page 16](#_Toc160879737)

[2.2.2 Functional Specification for Search 16](#_Toc160879738)

[2.2.3 Sell an item 16](#_Toc160879739)

[2.2.4 Browse categories 16](#_Toc160879740)

[2.2.5 Buy an item (Item home page) 17](#_Toc160879741)

[2.2.6 Edit an item 17](#_Toc160879742)

[2.2.7 Notification 17](#_Toc160879743)

[2.2.8 Review (Optional) 17](#_Toc160879744)

[2.2.9 Administrator function 17](#_Toc160879745)

[2.3 Technical requirements 18](#_Toc160879746)

[2.3.1 Hardware 18](#_Toc160879747)

[2.3.2 Software 18](#_Toc160879748)

[3 Idea 18](#_Toc160879749)

[3.1 System idea 18](#_Toc160879750)

[3.1.1 Backend 20](#_Toc160879751)

[3.1.2 Frontend 20](#_Toc160879752)

[3.1.3 API 20](#_Toc160879753)

[3.2 User interface ideas 22](#_Toc160879754)

[3.3 Ideas about technology and techniques used 24](#_Toc160879755)

[3.3.1 Overview of technology and techniques used 24](#_Toc160879756)

[3.3.2 Front end: ReactJS 24](#_Toc160879757)

[3.3.3 Back end: ASP.NET Core API 25](#_Toc160879758)

[3.3.4 Database: SQL Server 25](#_Toc160879759)

[3.3.5 Design Patterns 26](#_Toc160879760)

[a) Front-end 26](#_Toc160879761)

[b) Back-end 27](#_Toc160879762)

[3.3.6 Other Frontend and Backend Libraries 27](#_Toc160879763)

[a) Front-end 27](#_Toc160879764)

[b) Back-end 28](#_Toc160879765)

[CHAPTER II: SYSTEM ANALYSIS AND DESIGN 30](#_Toc160879766)

[1. Analyze topic requirements 30](#_Toc160879767)

[1.1 Summary of the operation of the system on which the project will be applied 30](#_Toc160879768)

[1.2 Subjects of use 30](#_Toc160879769)

[2. Sitemap 31](#_Toc160879770)

[3. Interface design 31](#_Toc160879771)

[3.1 Header interface 31](#_Toc160879772)

[3.2 Footer interface 32](#_Toc160879773)

[3.3 Login page interface 32](#_Toc160879774)

[3.4 Register page interface 33](#_Toc160879775)

[3.5 Forgot password page interface 33](#_Toc160879776)

[3.6 Category management page interface 34](#_Toc160879777)

[3.7 User management page interface 34](#_Toc160879778)

[3.8 User Leaderboard page interface 34](#_Toc160879779)

[3.8 Home page interface 35](#_Toc160879780)

[3.9 Category Browsing Page Interface 36](#_Toc160879781)

[3.10 Notification page interface 36](#_Toc160879782)

[3.11 Profile page interface 37](#_Toc160879783)

[3.12 My bid page interface 37](#_Toc160879784)

[3.13 Bid detail view page interface 38](#_Toc160879785)

[3.14 Bid detail create or edit page interface 38](#_Toc160879786)

[3.15 Report items page interface 39](#_Toc160879787)

[4. Analyze system functions 40](#_Toc160879788)

[4.1 Usecase overview 40](#_Toc160879789)

[4.1 Functions of the User Object 41](#_Toc160879790)

[4.1.1 Login function 42](#_Toc160879791)

[4.1.2 User remember function 43](#_Toc160879792)

[4.1.3 Registration function 44](#_Toc160879793)

[4.1.4 Forgot password function 45](#_Toc160879794)

[4.1.5 Logout function 47](#_Toc160879795)

[4.1.6 User information update function 48](#_Toc160879796)

[4.1.7 Function allows users to lock their accounts 50](#_Toc160879797)

[4.1.7 Function to view user rankings 52](#_Toc160879798)

[4.2 Functions of the Manager object (Admin) 53](#_Toc160879799)

[4.2.1 User management function 54](#_Toc160879800)

[4.2.2 User blocking function 55](#_Toc160879801)

[4.2.2 Category Management Function 56](#_Toc160879802)

[4.2.2.1. Add category function 56](#_Toc160879803)

[4.2.2.2 Edit category function 58](#_Toc160879804)

[4.3 Functions of Buyer and Seller Objects 60](#_Toc160879805)

[4.3.1 General functions 61](#_Toc160879806)

[4.3.1.1 Auction viewing function 61](#_Toc160879807)

[4.3.1.2 Evaluation function 62](#_Toc160879808)

[4.3.1.3 Function to view the list of auctions that buyers/sellers have participated in 63](#_Toc160879809)

[4.3.1.4 Notification viewing function 64](#_Toc160879810)

[4.3.1.5 Search function by product name 65](#_Toc160879811)

[4.3.2 Buyer's functions 66](#_Toc160879812)

[4.3.2.1 Pricing function 66](#_Toc160879813)

[4.3.3 Seller's functions 68](#_Toc160879814)

[4.3.3.1 Function to view auction history 68](#_Toc160879815)

[4.3.3.2 Auction creation function 70](#_Toc160879816)

[4.3.3.3 Auction editing function 71](#_Toc160879817)

[4.3.3.4 Function to view product list reports 73](#_Toc160879818)

[5. Database design 75](#_Toc160879819)

[5.1 Entity Relationship Diagram 75](#_Toc160879820)

[5.2 Specification of database tables and fields 76](#_Toc160879821)

[5.2.1 User table 76](#_Toc160879822)

[5.2.2 Bid Table 76](#_Toc160879823)

[5.2.3 Item table 76](#_Toc160879824)

[5.2.4 Category table 77](#_Toc160879825)

[5.2.5 Notification table 77](#_Toc160879826)

[5.2.6 Rating table 78](#_Toc160879827)

[CHAPTER III: DEPLOYMENT AND DEMO 79](#_Toc160879828)

[Part 3: CONCLUSION 84](#_Toc160879829)

[1. Conclude 84](#_Toc160879830)

[2. Development 84](#_Toc160879831)

[Part 4: REFERENCES 86](#_Toc160879832)

# PREFACE

In today's digital age, e-commerce is becoming an indispensable and rapidly growing part of the global economy. With the dramatic rise of the Internet and the popularity of connected devices, buying and selling goods online is more popular than ever. In the e-commerce industry, online auction systems play an important role, providing a fair and convenient platform for both buyers and sellers.

The project we worked on focused on building an online auction platform with basic features like item creation, bidding, and transaction management. Our goal is to better understand how the online auction system works and how it interacts with users, thereby creating the best experience for both buyers and sellers.

First of all, we would like to send our sincere thanks to instructor Le Thien Nhat Quang and the international information technology training system Softech Aptech. Your support and extensive knowledge helped us complete this project efficiently and professionally.

# Part 1: INTRODUCTION

* 1. Introducing the project

What is an online auction?

Online auctions are auction activities through information systems. Organizations and individuals can independently organize online auctions or use the online auction services of other organizations or individuals. Auctions can take place through the auction floor or the seller's sales website.

Advantages of online auctions?

* ***No time constraints***: Bidding can be done at any time. Products are listed for one or more days to give buyers time to search, decide and bid. The value of the item will be increased according to the number of people participating in the auction.
* ***No geographical constraints:*** Vendors and bidders can participate in auctions from anywhere with Internet access. This helps increase proximity and reduce the cost of attending an auction.
* ***The power of social interaction:*** Auctioneers often wait to win, like gambling. Many people participate in auctions to "play the game" and not necessarily to win the item.

## Project goals

* The project's goal is to develop a simple yet effective online auction system, focusing on basic features such as item creation, bidding, and transaction management. We emphasize understanding how the system works and how it interacts with users, to provide the best online auction experience for both buyers and sellers.
* The project is not aimed at developing complex technical features or advanced features. Instead, we focus on building a simple and easy-to-use system that makes it easy for users to create and manage auctions, and conduct transactions conveniently and securely.
* Our main goal is to create a fair and transparent online auction environment, where all parties can participate easily and without complications. We hope that the project will bring real value to the entire user community and contribute to the development of the online auction market.
  1. Object and scope of the study

1.3.1 Subjects

- In this project, we will focus on the online auction system. This object includes the following components:

***1. Users***: Includes buyers and sellers participating in the auction system. They will create items, place bids, and make transactions.

***2. Items***: These are products, services or assets posted to the system for auction.

***3. Transaction***: Includes the process of placing bids, accepting bids, and making payments.

1.3.2 Scope of research

- The scope of our research will include:

***1. System construction:*** Create an online auction platform with basic features like item creation, bidding, and transaction management.

***2. Discover how it works:*** Understand how the auction system interacts with users and manages transactions.

* 1. Research Methods

During the research process, we will apply methods to collect and process information. Here are some important methods:

### Collection method

We will use theoretical research methods to collect information from sources such as books, documents, and websites. This helps us learn the theoretical foundations related to the research problem.

### Information processing methods

During the information processing phase, we will apply the following methods:

* ***Analyze and evaluate***: Based on the theoretical basis we have found, we will - evaluate the problem in a practical context. From there, we can choose a perfect solution for the topic.
* ***Integrated approach:*** We will combine analytics with technology to make the best choices. Step by step go further to complete the topic.

# Part 2: CONTENTS

## CHAPTER I: SURVEY, DETERMINATION OF REQUIREMENTS AND IDEAS

* In this chapter, we will explore important aspects of online auction systems. We will learn about user requirements and define basic ideas for the project.

1. Survey

### 1.1 Survey purpose

* This phase focuses on collecting information and documents related to the system's structure and system operations to identify some issues before starting to build and develop a project.
* The content to be surveyed corresponds to the problems that need to be solved by user requirements and the scale of the information system.

### 1.2 Trends and overall market situation:

* The online auction market is going through a period of strong growth and is witnessing a significant change in the way consumers shop and do business online. Below are some trends and the general situation of this market:
* ***Significant Growth:*** The online auction market is witnessing significant growth, driven by the popularity of the Internet and the convenience of shopping and selling online. Consumers are increasingly turning to online auction platforms to find products and shopping opportunities.
* ***Product Diversification:*** The online auction market is not just limited to selling electronics or second-hand products, but has also expanded into many different fields, including real estate, art, sporting goods, and even services. service.
* ***Exciting Competition***: With the development of the market, competition in this field is also becoming increasingly fierce. Online auction platforms need to continuously improve user experience and provide true value to attract and retain customers.
* ***Diversity and Flexibility:*** One of the advantages of the online auction market is the diversity and flexibility in how auctions are organized. From live auctions to online auctions, from short-term auctions to extended auctions, there are many options for both buyers and sellers.
* ***Technological Innovation***: The online auction market is constantly innovating and applying new technologies such as artificial intelligence, machine learning and blockchain to improve the safety, transparency and efficiency of transactions.

In short, the online auction market is going through a period of development full of potential and competition. Understanding these trends and landscapes is key to success in this dynamic and exciting business environment.

* 1. Market survey
* ***Live auction websites in Vietnam and around the world***
* ***In Vietnam***

****

Figure 1: Logo of Lac Viet auction company

***Lac Viet Auction:***  This is one of the largest auction houses in Vietnam. Lac Viet has always been a pioneer in applying information technology to auction activities. They successfully organized the first official online auction in Vietnam on July 17, 2020.

****

Figure 2: Sohot Auction company logo

***Sohot.vn:*** Another online auction platform in Vietnam.

* *In the world*

****

Figure 3: Logo of online auction website eBay

***eBay:*** One of the most famous and popular online auction websites in the world. eBay allows users to auction buy and sell new and used goods.

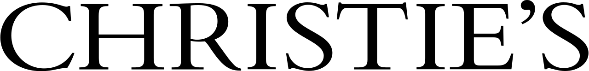
****

Figure 4: Christie's Auction Company logo

***Christie's:***  Is one of the most famous auction houses in the world, specializing in organizing auctions of works of art, antiques, and jewelry.

****

Figure 5: Sotheby's auction company logo

***Sotheby's:*** Also one of the world's leading auction houses, Sotheby's organizes auctions of works of art, antiques, and jewellery.

1.4 User needs

**-** Analyzing customers' preferences wants and needs in buying and selling through the online auction system is an important part of understanding and properly meeting their needs. Here are some basic factors to consider:

***Product Diversity:*** Consumers have diverse tastes, from electronics and fashion products to home appliances, art and antiques. They want a wide choice to find and buy products that reflect their personality and style.

***Value and Quality*:** Consumers care about the value and quality of the products they buy. They want to make sure they are buying quality and value for the money they are paying. Therefore, transparency and detailed product descriptions are very important to them.

***User experience*:** Consumers want a convenient and easy online shopping experience. They need a user-friendly and easy-to-use interface, along with effective product search and filtering features so they can easily find what they are looking for.

***Safety and Security*:** Security and confidentiality of personal information is an important factor for consumers when shopping online. They want to ensure that their personal and account information is securely protected when using the online auction system.

***Flexibility and Time:*** Consumers often desire flexibility in participating in auctions and completing transactions. They may wish to participate in short-term or long-term auctions depending on their schedules and needs.

***Interaction and Feedback:*** Consumers appreciate having the opportunity to interact and provide feedback on products and services. They want to hear and see their opinions cared for and responded to by the seller.

1.5 Survey scope

- Our course projects focus on applying the knowledge learned in practice and developing research, analytical and logical thinking skills. Make sure your scope reflects the specific goals and requirements of your course and instructor. This scope is limited to the information technology sector and focuses on developing and implementing an online auction website. Other aspects such as business, marketing and market analysis can be excluded from this scope.

**- *Basic Features of Online Auction Website****:* Focus on developing basic features such as account registration, login, item listing creation, bidding, and transaction management.

- ***User Interface:*** Friendly and easy-to-use user interface design. Take into account aspects such as user experience, interaction, aesthetics and compatibility across different devices.

- ***Safety and Security*:** Learn about safety and security measures to protect users' personal information and ensure the integrity of transaction data.

- ***Database and Data Management:*** Develop a database to store information about products, users and transactions. Determine how to manage data to ensure system performance and scalability.

- ***Optimize Performance and Flexibility:*** Integrate techniques and technologies to optimise performance and respond quickly to large numbers of users and transactions.

***- Testing and Inspection***: Perform tests and experiments to ensure the stability and quality of the website, including security testing, feature testing, and compatibility testing.

***- User Guide and Support Features***: Develop user documentation and provide support features such as an online help system or support centre.

* 1. Evaluate the infrastructure of current auction websites
* Below are some objective assessments of the infrastructure of current auction websites based on important criteria:

***1. Stability***: Some auction sites have achieved a high level of stability, ensuring that they can operate continuously without unwanted interruptions. However, some websites encounter stability problems, especially when a large number of users access them simultaneously.

***2. Performance***: Top auction sites typically have good performance, with quick response times and consistent page load speeds. However, some sites may experience performance issues when experiencing heavy traffic or when running multiple auctions at the same time.

***3. Security*:** Top auction sites often have strong security measures in place to protect users' personal information and ensure the integrity of transaction data. However, some small websites may not have strong enough security measures, causing concerns for users about information security.

***4. Scalability*:** Popular auction sites often have good scalability, allowing them to scale their infrastructure to meet increased user demand and scale their business.

***5. Integrating New Technology*:** Top auction sites often integrate the latest technologies to improve user experience and optimize system performance. However, some websites may have difficulty integrating the latest technologies due to limited resources or technical knowledge.

***6. Redundancy and Backup*:** Quality auction sites typically have adequate backup and redundancy measures in place, ensuring that user data is protected and can be restored when necessary.

1. Determine requirements
   1. General system requirements

- Below is an overview of the business process of the online auction system:

***1. Create Account and Login:***

* Users can create a new account on the system by providing necessary information such as name, email address and password.
* After creating an account, they can log in to access the system and start using features.

***2. Create Auction Items***:

* Sellers can create a new listing for the item they want to sell. They provide detailed product information such as name, description, images, and starting price.
* Once the item is created, it will be displayed on the website for buyers to view and bid on.

***3. Bid:***

* Buyers can browse the list of auction items available on the site and place a bid on the item they are interested in.
* The system automatically updates the highest bid and notifies when the auction ends.

***4. Transaction Management:***

* At the end of the auction period, the seller and buyer combine to complete the transaction. Contact information will be shared for payment and delivery purposes.
* The system provides features to track the status of transactions, including auction status and current bet amount.

***5. Rating:***

* Once the transaction is completed, both the buyer and the seller can rate each other. This helps build credibility and trust within the online auction community.

***6. User Interface:***

* The user interface is designed to ensure ease of use and a smooth online shopping experience for both buyers and sellers.
  1. Specific requirements for each function

Users need to register to the website and log in to access the Auction page

There are two types of users in the system

* Manager
* Normal users

As soon as the user logs into the system, the home page will be displayed. The home page should be a dashboard where users can access any other functions available on the website

2.2.1 Content of the Home Page

* Search the text box using the go button to search for individual items
* List of all Categories available in the system. It will also show in brackets the items sold in this category
* Sell ​​an affiliate item

### 2.2.2 Functional Specification for Search

The search must be a case-insensitive search for items containing the search keyword. The search results will display all items that match the search criteria. It will only return items that are currently on sale.

### 2.2.3 Sell an item

When a user clicks the Sell an item link, a page appears that allows the user to do the following:

* Provide a title for the item and a description of the Item.
* There will be an Upload Image button that users use to upload images of the item. There will be an Upload text button so the user can upload a Word document giving the description/specifications of the item.
* A text box to set the minimum bid
* A text box to provide the auction end date.
* A text box to provide the bid increase.

### 2.2.4 Browse categories

When a user clicks on any of the Categories on the home page, a new page opens showing all the items for sale in the Categories.

### 2.2.5 Buy an item (Item home page)

You can access the Section home page by clicking on a search result or by clicking on Section while performing a Category browse.

Home Section contains the following sections: -

* Title of the item
* Description of the item
* Uploaded image for the item (if available)
* Documents uploaded for the item (if applicable)
* Name of the Seller (This will be the link that opens a new page giving the Seller's review)
* Minimum bid price (if any)
* Current bid price.
* The link to view history of previous bids opens a new page providing the bid amount and the user who placed the bid
* Bidding increased
* Button with a text box next to it to place a new Bid (This feature will not be available for item sellers)

### 2.2.6 Edit an item

Sellers will be able to edit items. Merchants can do the following:

1. Upload a new image
2. Upload a document
3. Change bid increments

### 2.2.7 Notification

Whenever the auction end date ends, sellers and buyers will be notified by a new notification that will appear in the Notifications tab. (Seller and user should contact each other for shipping of goods).

### 2.2.8 Review (Optional)

* A rating mechanism could be developed for exchange rate buyers and sellers.
* For example, for each item sold, the seller will rate the buyer on a scale of -5 to +5. A rating of 0 will not be evaluated. For each rating, the buyer's score will be increased or decreased according to the rating. So if a buyer has a high cumulative rating it means he or she is a good buyer.

### 2.2.9 Administrator function

* Administrators must be able to block fraudulent users from using the system.
* Admins will be able to add more categories and merge categories
  1. Technical requirements
     1. Hardware
* A minimum computer system that will give you access to all the tools in the course is a Pentium 166 or higher
* 128 MB RAM or higher

### 2.3.2 Software

* Visual Studio .Net / ASP
* IIS server
* .Net framework
* Java virtual machine/J2EE server
* Notepad/Java editor
* j2sdk1.4.1\_02 (or later).
* EJB Development Kit
* The web server supports Java
* JSP/Servlet Developer Kits

1. Idea
   1. System idea

* *In the project system, there are three main parts: backend, frontend and API*

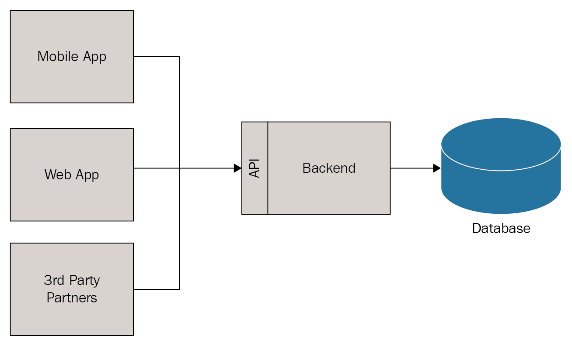


Figure 6: Three-element architecture

* In modern software architecture, frontend, backend, and API play an important role in creating a complete application.
* The front end, also known as the client side, is where users directly interact with the system. It is typically built using technologies such as HTML, CSS, and JavaScript, and is responsible for data display and user interface.
* The backend, also known as the server side, is where the system handles requests from users, interacts with databases, and performs complex tasks. Backends are often written in languages ​​like C#, Java, Python, Ruby, Node.js, etc.
* API (Application Programming Interface), acts as a bridge between the frontend and backend. When a user acts on the front end, the front end sends a request to the API. The API then passes that request to the backend for processing. When the backend completes processing, it sends the response back to the frontend through the API.
* Thus, the frontend, backend, and API together create a smooth operating system where users can interact with the system easily and conveniently.
* Benefits of using the frontend-backend-API architecture for website development

1. **Easier Application Maintenance:**

* By separating frontend requirements from backend concerns, this architecture simplifies maintenance.
* Changes in APIs won’t disrupt the front end, making applications more resilient.

1. **Improved Error Handling:**

* The backend-for-frontend pattern enhances error handling.
* Isolating frontend logic allows better handling of unexpected scenarios.

1. **Modularity and Scalability:**

* Decoupling frontend interfaces from backend services creates a modular system.
* Scalability becomes easier as each frontend can have its dedicated backend server.

1. **Simplified Frontends:**

* Focused on individual UIs, this pattern keeps frontends simple.
* Teams maintain a unified view of data through the backend.

1. **Streamlined Data Representation:**

* The backend-for-frontend approach provides a well-focused interface for the front end.
* Data representation is optimized, improving efficiency.

1. **Customization for Specific Frontends:**

* When specific frontend interfaces require tailored data or optimizations, this architecture allows dedicated backends.

3.1.1 Backend

* ***Data management:*** Build an effective data management system. Use databases (e.g. MySQL, PostgreSQL) to store information about users, products, transactions, etc.
* ***Authentication and authorization:*** Build an authentication and authorization system to ensure that only logged-in users can access critical functions.
* ***API for Frontend*:** Create APIs to provide data to the frontend. This includes building endpoints to retrieve product information, auction listings, etc

3.1.2 Frontend

* ***User interface:*** Attractive and friendly user interface design.
* ***Show auction list:*** Create an auction listing page with product information, starting price, time remaining, etc

3.1.3 API

* ***API Bid (Auction):***
  + - ***Create a new auction:*** Allows users to create new auctions for specific products.
    - ***Set price:*** Users can set prices for products in the auction.
    - ***Track price changes:*** Provides information about current prices and price changes during the auction.
* ***API Category:***
  + - ***Product catalogue management:*** Allows adding, modifying or deleting product categories.
    - ***List categories:*** Displays a list of product categories.
* ***API Item (Product):***
  + - ***Add new products:*** Users can add new products to the system.
    - ***Display product information:*** Provide detailed information about the product, including images, description, condition, etc
* ***API Notification:***
  + - ***Send notification:*** Allows notifications to be sent to users when important events occur (e.g. auction ends, new prices are set).
    - ***See announcement:*** Display a list of notifications to the user.
* ***API Rating:***
  + - ***User reviews:*** Allows users to rate products or sellers.
    - ***Show reviews:*** Displays information about user reviews.
* ***API User (User):***
  + - ***User authentication:*** When they log in or perform important operations.
    - ***Register and log in:*** Build API for account registration and login.
    - ***User information management:*** Allows users to update personal information, change passwords, etc
  1. User interface ideas
* ***Main page:***
* Displays a list of items being auctioned or featured items.
* Provide filtering and sorting options so users can easily search for products.
* Display a clear and easy-to-use search bar.
* ***Product Details:***
  + Display large and clear images of the product.
  + Provide detailed information about the product such as name, description, condition, starting price and remaining time of the auction.
  + Allow users to bid directly from the product detail page.
* ***Registration and Login page:***
* Simplify the registration and login process with concise forms.
* Provides quick login options via social media or Google accounts.
* ***Create/Edit Item Page:***
* Provide a simple form so sellers can create new items.
* Clear instructions and support for users in filling out product information.
* ***Transaction Page:***
* Shows information about the deal such as highest bid, time remaining, and winner (if applicable).
* Enables users to track and manage their transactions, including payments and shipping.
* ***Notification Page:***
* Show important notifications about transactions or account information.
* Allows users to customize their notification settings.
* ***Headers:***
* ***Logo and Website Name*:** Display the website logo and name that reflects the brand.
* ***Main Navigation Bar*:** Provides links to important sections of the site such as Home, Products, Login/Register, Support, etc.
* ***Search*:** Provide a search box so users can search for the desired product right from the main page.
* ***Notification*:** Display important notifications such as transaction information, product updates, etc.
* ***Log out:*** Allow users to log out of their accounts when necessary.
* ***Footer:***
* Provides quick links to important pages such as the main page, login page, registration page, and support information page.
* Displays information about the site's policies and terms.
  1. Ideas about technology and techniques used
     1. Overview of technology and techniques used
* **Front-end**

1. Tool: Visual Code
2. Platform: NodeJs
3. Language: Javascript
4. Support libraries:
   * + - ReactJs
         * React Hooks
         * React-bootstrap 2.8.0
       - Formik 2.4.5
       - Yup 1.3.3
       - Prettier 3.2.4
       - Scss
       - Eslint 8.39.0
5. Design Pattern: Atomic

* **Back-end**

1. Tools: Visual Studio .Net
2. Platform: .Net Core 7
3. Language: C#
4. Support libraries:

* Entity Framework Core
* Swagger

1. Design Pattern:

* Repository
* Dependency injection
* **Database**

1. Platform: SQL Server 2022
2. Tool: MSSQL Server Management 19
3. Language: SQL

3.3.2 Front end: ReactJS

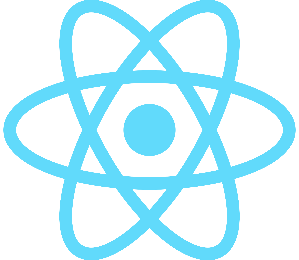


Figure 7: Logo of ReactJS

**ReactJS** is a popular JavaScript library for building user interfaces (UI). It allows you to create reusable components, manage application state, and optimize performance.

*Advantage:*

* High performance: ReactJS uses Virtual DOM to optimize interface updates, helping applications run faster.
* Large community: There is extensive documentation, libraries, and support from the ReactJS community.
* Integrates well with other libraries: ReactJS easily integrates with other libraries like Redux, Axios, etc.
  + 1. Back end: ASP.NET Core API

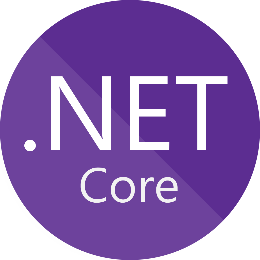


Figure 8: Logo of .NET core

* **ASP.NET Core is** a web application and API development framework developed by Microsoft. It supports building cross-platform and high-performance applications.

*Advantage:*

* Cross-platform: ASP.NET Core supports Windows, Linux, and macOS.
* Good performance: ASP.NET Core is optimized for high performance and has good concurrency capabilities.
* Security: Provides security features such as authentication, authorization, and data protection.
* **ASP.NET Code First** a method in Entity Framework that allows us to create a database from model classes in the source code. This is very useful when we want complete control over the source code and do not want to use a visual designer tool to create the database.
* With Code First, we can:
* Create a database from model classes.
* Use relationships between classes such as one-to-one, one-to-many, and many-to-many.
* Use Data Annotations and Fluent API to configure the model.
* Use Code First Migrations to update the database when the model changes.
* Leverage powerful ASP.NET features like Identity to handle user authentication and authorisation
* In our project, using Code First made it easy to manage the database and make model changes dynamically.
  + 1. Database: SQL Server

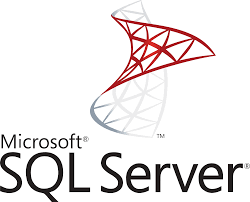


Figure 9: Logo of Microsoft SQL Server

**SQL Server is** a popular relational database management system from Microsoft. It supports structured data storage and data query management.

*Advantage:*

* Good integration with ASP.NET: SQL Server easily integrates with ASP.NET Core applications.
* Security and data management: SQL Server provides effective data management and security features.
  + 1. Design Patterns

1. Front-end
   * **Atom Design**

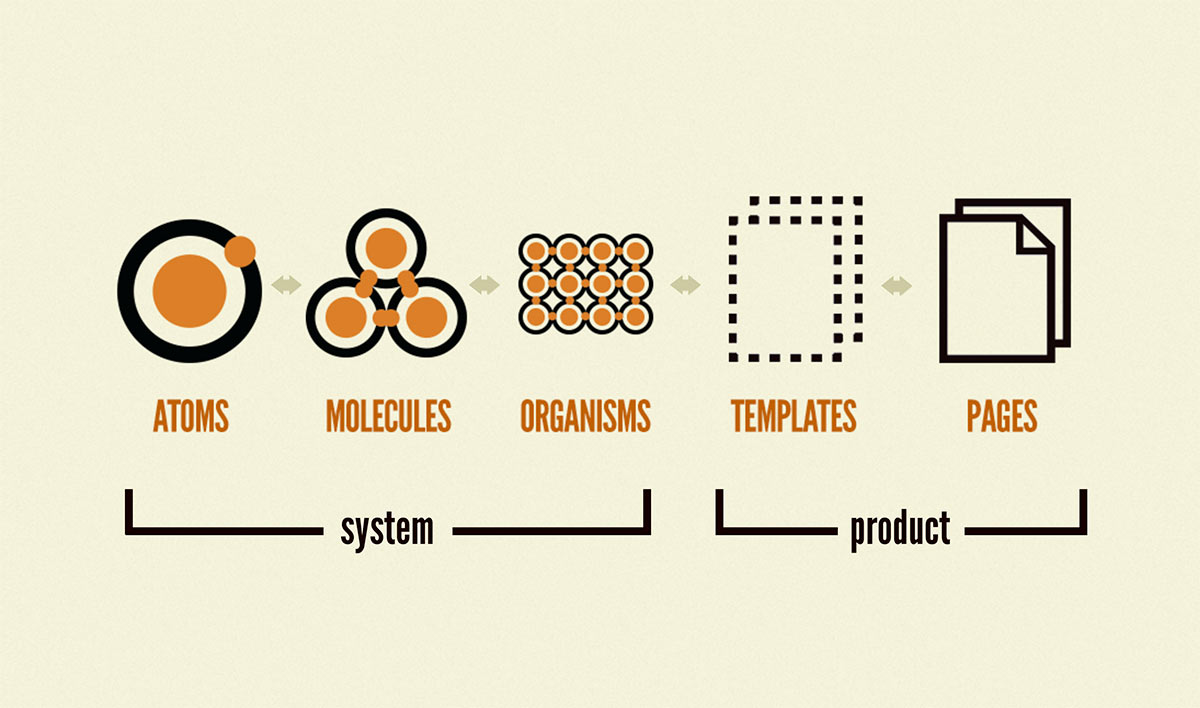


Figure 10: Atomic design

* Based on Brad Frost's Atomic Design methodology, this model helps you build user interfaces in an organized and manageable way. Here are some key points about the ATOM model:
* ***Atomic Design Principles:***
  + ***Atoms (Atoms):*** The most basic components, such as buttons, inputs, and icons.
  + ***Molecules (Molecules):*** Combine atoms, for example, a login form.
  + ***Organisms (Living Organs):*** Combine molecules to form larger components, such as headers and footers.
  + ***Templates:*** Build sample pages using living organisms.
  + ***Pages:*** Generate specific pages from templates.
* ***Benefits of ATOM:***
  + ***Reuse:*** Small components can be reused in multiple parts of the application.
  + ***Easy management:*** Separating the interface into small parts makes management and maintenance easier.
  + ***High performance:*** Using small components helps optimise performance.
* ***Apply ATOM to ReactJS:***
  + ***Components:*** Build ReactJS components corresponding to atoms, molecules, living organisms, templates, and pages.
  + ***Status management:*** Use Redux or Context API to manage the state of components.
  + ***Separation of logic and interface:*** Create container components (control logic) and presentational components (display interface).

1. Back-end
   * **Dependency Injection**

* Use Dependency Injection to manage dependencies between components.
* Ensure that services, repositories, and other components are injected into the controller dynamically.
  + **Repository Pattern**
* Create repositories to interact with the database.
* Repository helps separate the service layer and the data access layer.
  + 1. Other Frontend and Backend Libraries

1. Front-end

Using Formik and Yup is a smart decision to handle validation and checking the values ​​of form fields. Here is some information on how to use them:

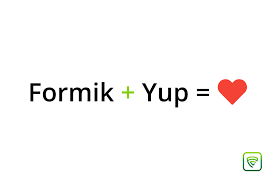


Figure 11: Build forms in ReactJS using Formik and Yup

* Formik
  + **Formik** is a ReactJS library that makes state management and form handling a breeze.
  + It supports building complex forms with validation.
  + You can use Formik to manage the values ​​of form fields and handle events when users submit the form.
* Yup
  + **Yup** is a library that helps build schemas to check the values ​​of fields.
  + You can define validation rules for fields (e.g. check email, password length, etc.) using Yup.
  + Formik and Yup work well together to create secure and easy-to-use forms.
* React-Bootstrap

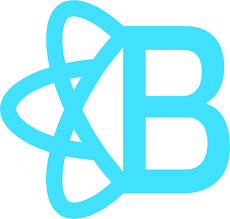


Figure 12: Logo of React-Bootstrap

**React-Bootstrap is** a useful library for building user interfaces in your ReactJS application. It provides pre-designed interface elements, helping you create attractive websites and applications quickly.

***Some benefits of using React-Bootstrap:***

* + Pre-designed: Components have been designed and optimised for the user interface. You can use them without writing custom CSS.
  + Compatible with React: React-Bootstrap is well integrated with ReactJS, making it easy for you to use components in your application.
  + Large community: There is extensive documentation and support from the ReactJS and Bootstrap communities.

1. Back-end

* Azure function



Figure 13: Logo of Azure Functions

**Azure Functions is** a serverless solution that allows you to write less code, maintain less infrastructure, and save costs. Instead of worrying about deploying and maintaining servers, cloud infrastructure provides all the resources needed to maintain your applications. You focus on the code that matters most to you, in the programming language that works best for you, and Azure Functions takes care of the rest.

***Common scenarios that Azure Functions supports include:***

* + Handle file uploads: Run code when a file is uploaded or changed in the blob storage.
  + Real-time data processing: Extract and transform data from IoT and event streams before storing.
  + Predictive processing on the data model: Extract text from the queue and feed it into artificial intelligence services for analysis and classification.
  + Run scheduled task: Execute data cleaning code at predetermined intervals.
  + Build scalable web APIs: Implement a set of REST endpoints for your web application using HTTP triggers.
  + Create serverless workflows: Create event-based workflows from a series of functions using Durable Functions.
  + Respond to changes in the database: Run custom code when documents are created or updated in Azure Cosmos DB.
  + Create a reliable messaging system: Process message queues using Queue Storage, Service Bus, or Event Hubs.

## CHAPTER II: SYSTEM ANALYSIS AND DESIGN

This chapter will focus on a more detailed analysis of how the system works and the design of key components. We will define how the user and the system interact.

1. Analyze topic requirements
   1. Summary of the operation of the system on which the project will be applied

* Is an online auction system where users can buy and sell goods. Here is a summary of how the system works:
* ***Register and Login*:** Users need to register and log in to use the system. They can also update their personal information.
* ***System Management*:** The system has three types of users: Manager, Buyer and Seller.
* **Manager:** Can manage categories and users.
* ***Home page*:** Once users log in, they will be taken to the Home page, where they can search for items and view a list of categories.
* ***Sell ​​Items*:** Users can post an item for sale. They need to provide a title, description, image, minimum bid, auction end date, and bid increment.
* ***Buy Items*:** Users can view items being auctioned, place bids, and view bid history.
* ***Rating*:** Buyers and sellers can rate each other after each transaction.
* ***Notification*:** When an auction ends, sellers and buyers will receive notifications.
* ***Automated system:*** The system automatically generates notifications, updates item status and bids, and generates auction ratings.
  1. Subjects of use

- There are three types of users in the system

* Manager
* Buyer
* Seller

## 2. Sitemap

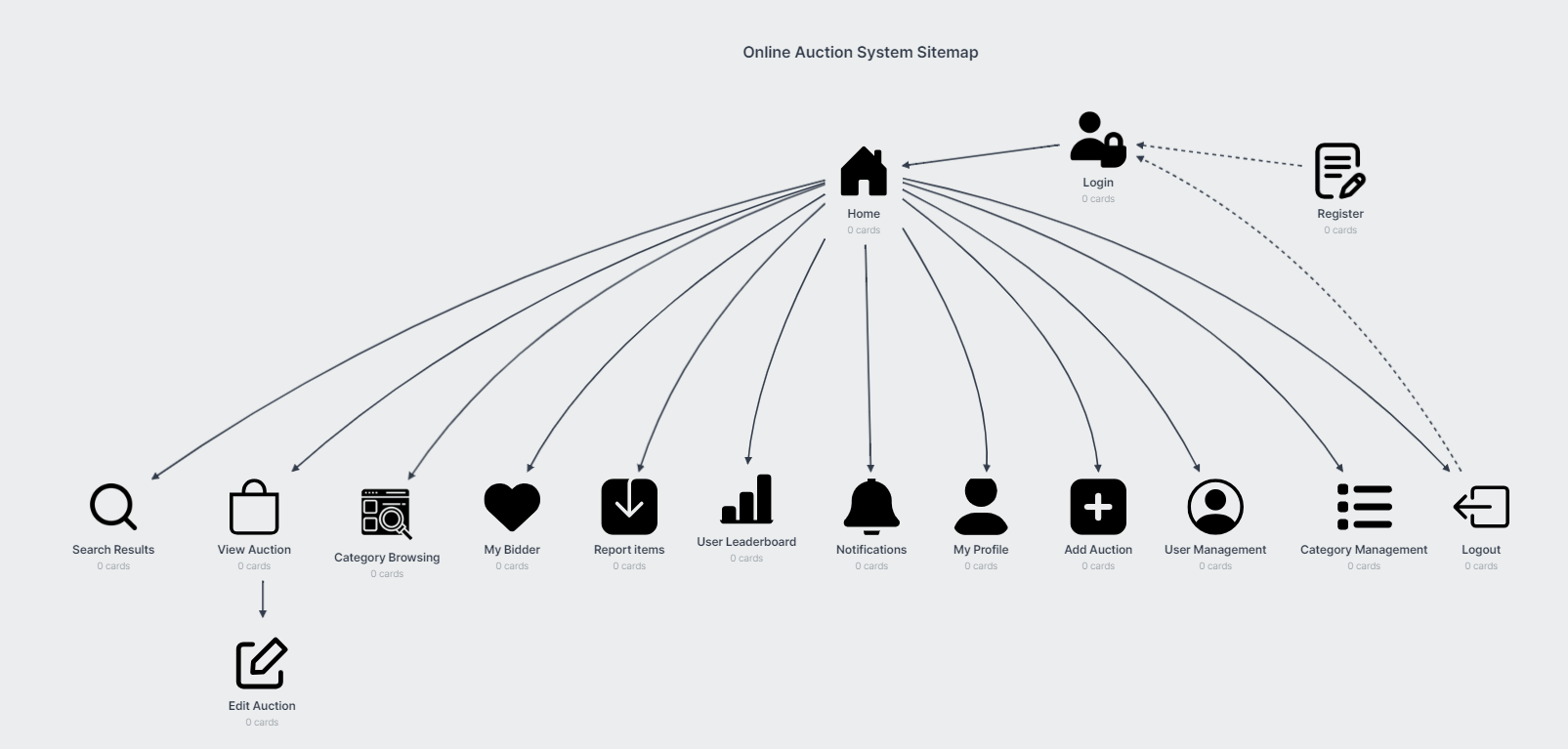


Figure 14: Sitemap of auction system website

## 3. Interface design

### 3.1 Header interface

a) Header interface for admin

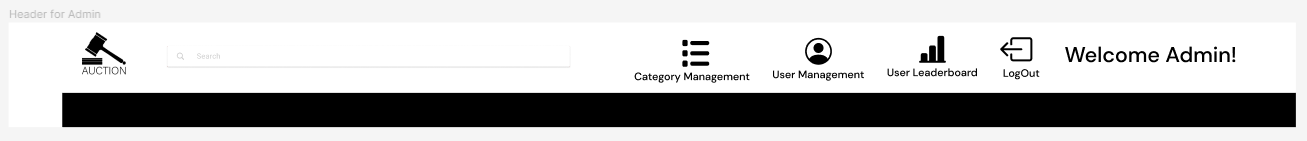


Figure 15: Header interface for admin

b) Header interface for buyer

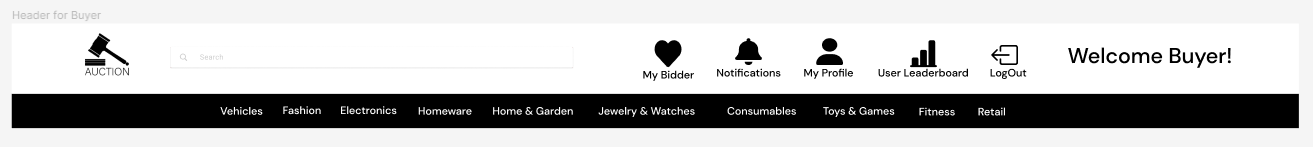


Figure 16: Header interface for Buyer

c) Header interface for seller

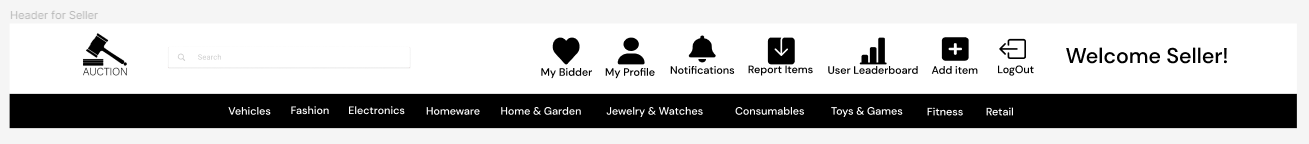


Figure 17: Header interface for Seller

### 3.2 Footer interface

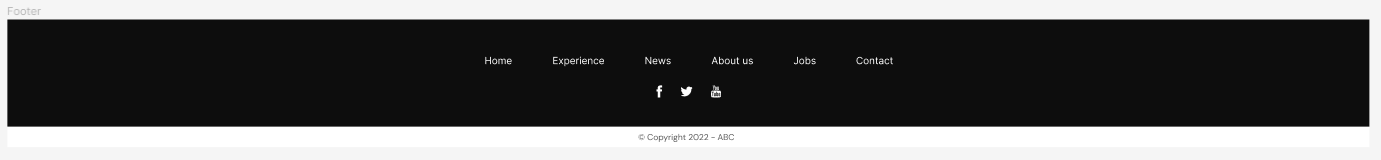


Figure 18: Footer interface

### 3.3 Login page interface

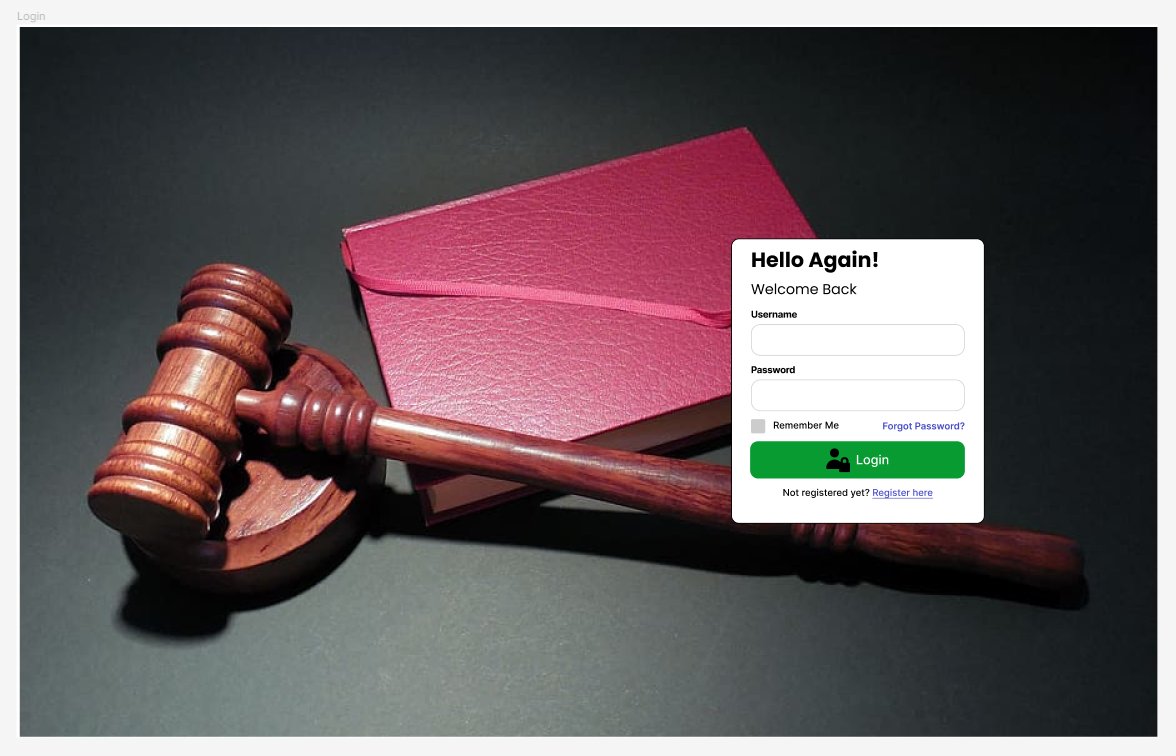


Figure 19: Login page interface

### 3.4 Register page interface

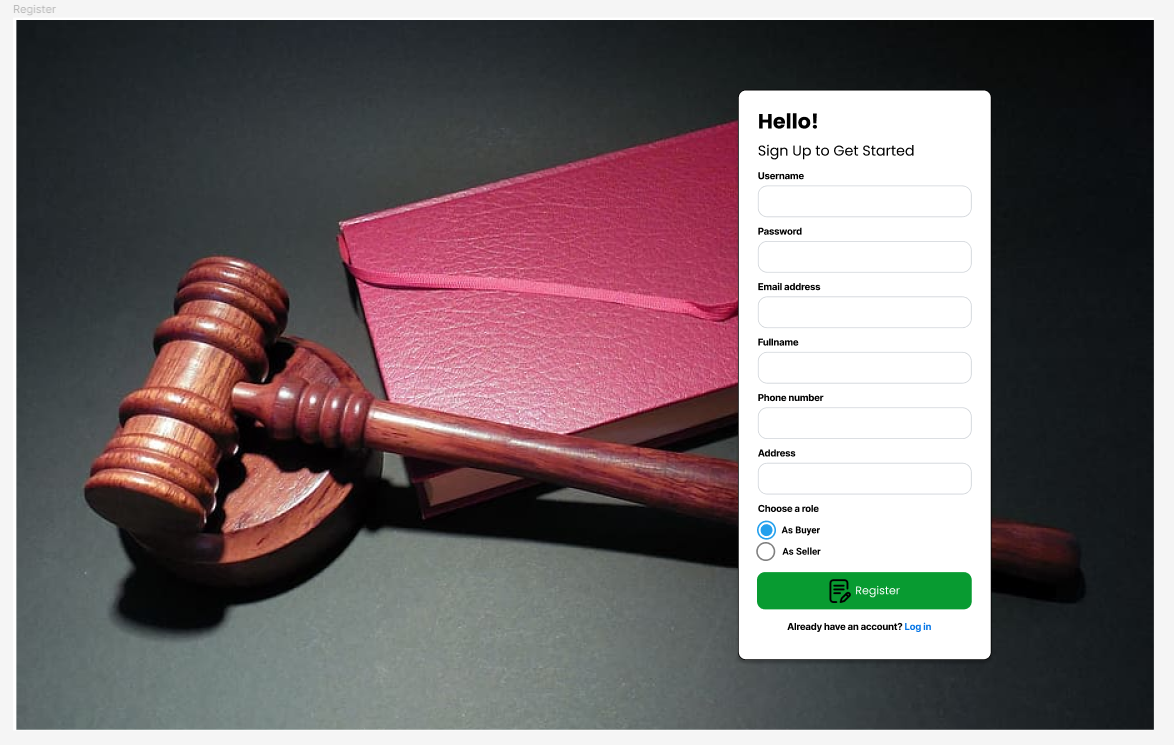


Figure 20: Register page interface

### 3.5 Forgot password page interface

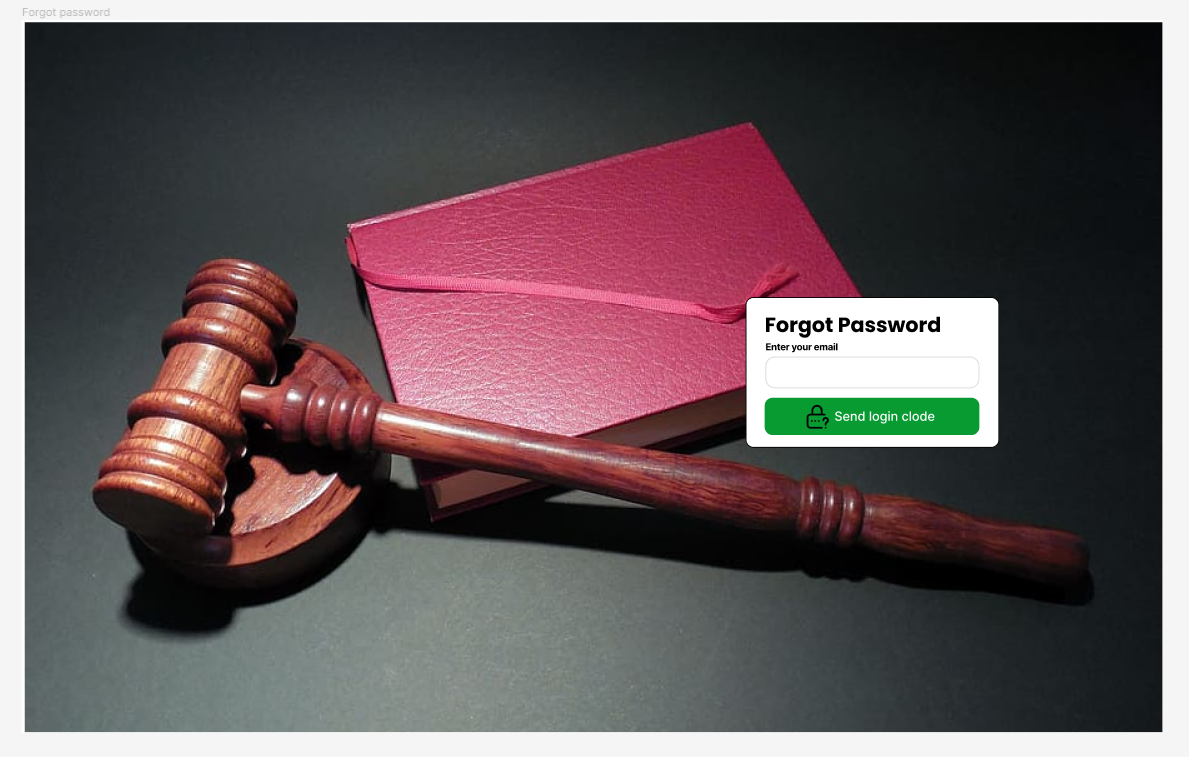


Figure 21: Forgot password page interface

### 3.6 Category management page interface

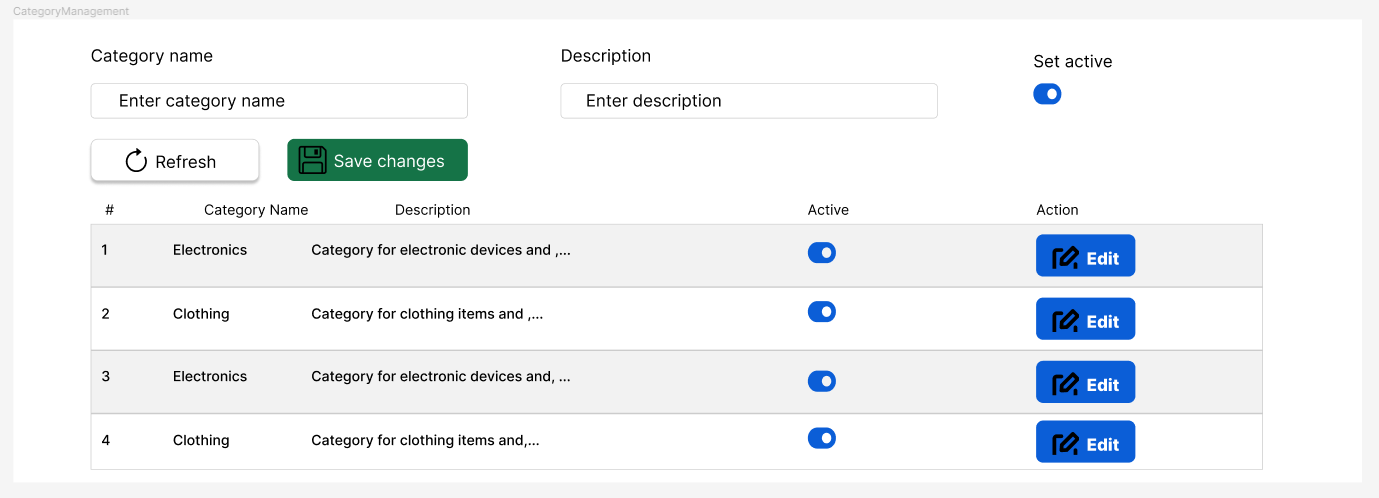


Figure 22: Category management page interface

### 3.7 User management page interface

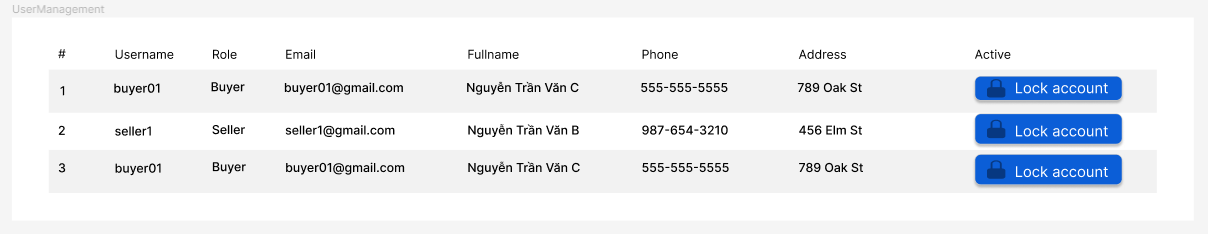


Figure 23: User management page interface

### 3.8 User Leaderboard page interface

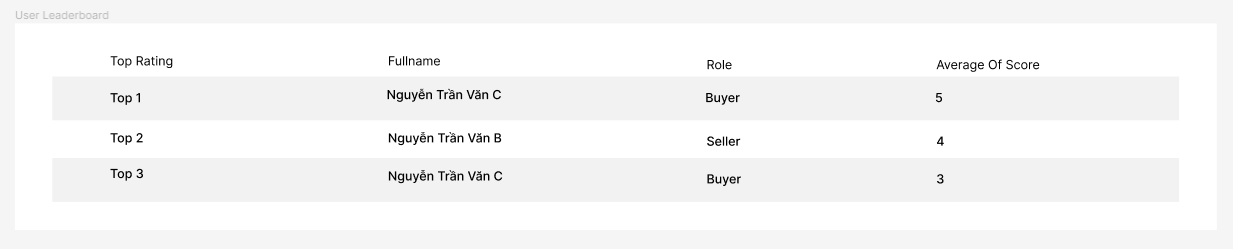


Figure 24: User leaderboard page interface

### 3.8 Home page interface

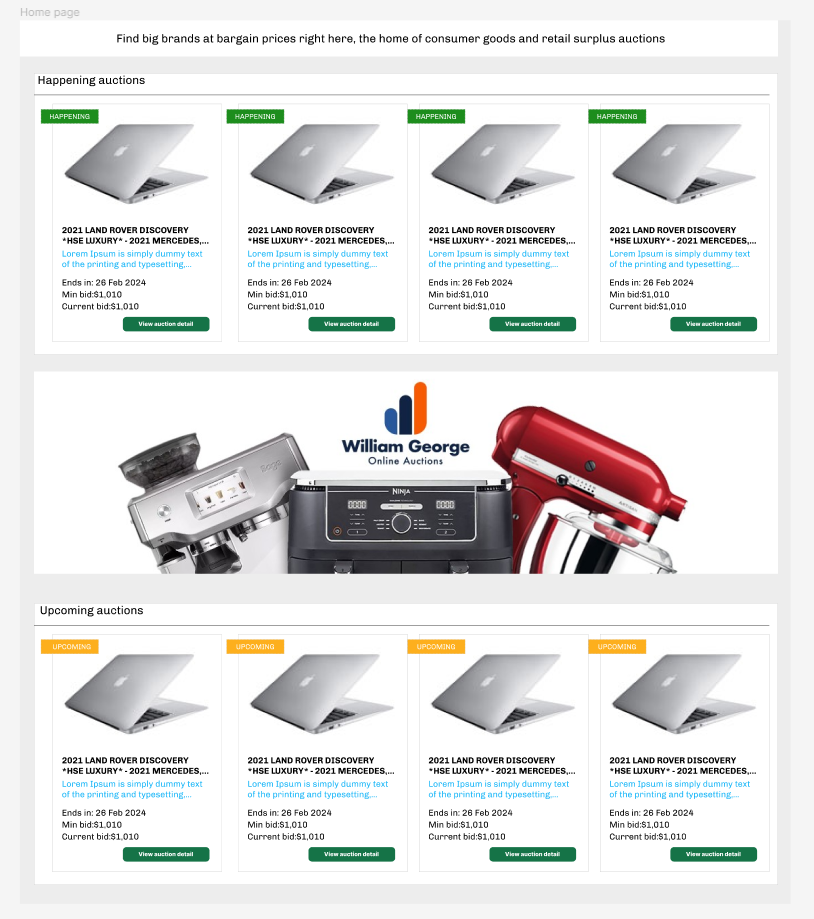


Figure 25: Homepage interface

### 3.9 Category Browsing Page Interface

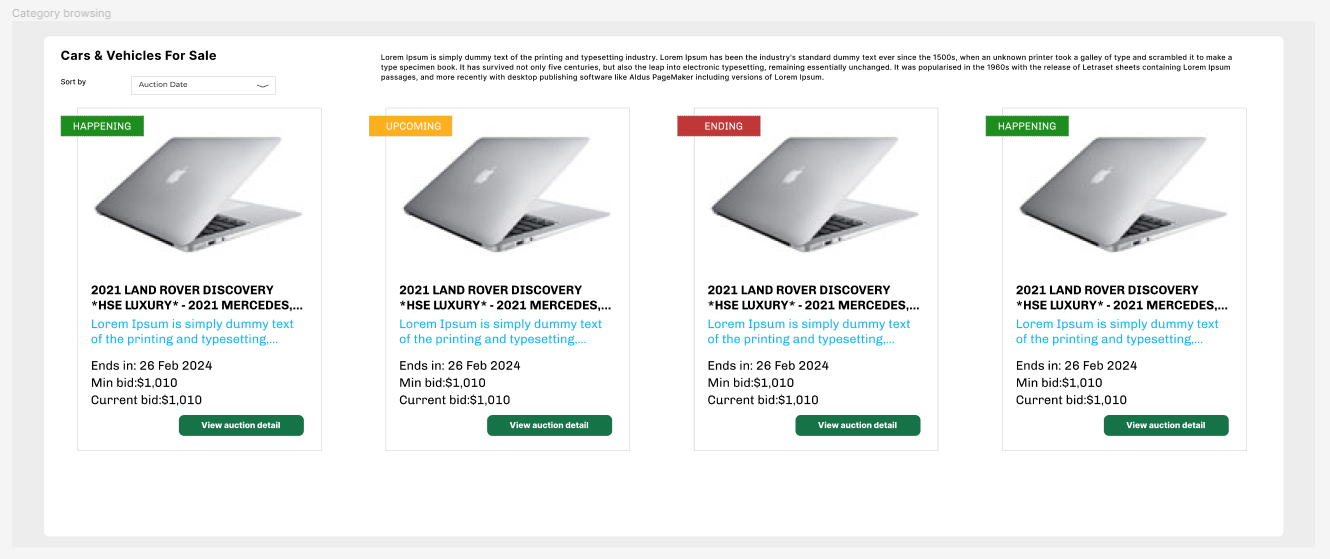


Figure 26: Category Browsing page interface

### 3.10 Notification page interface

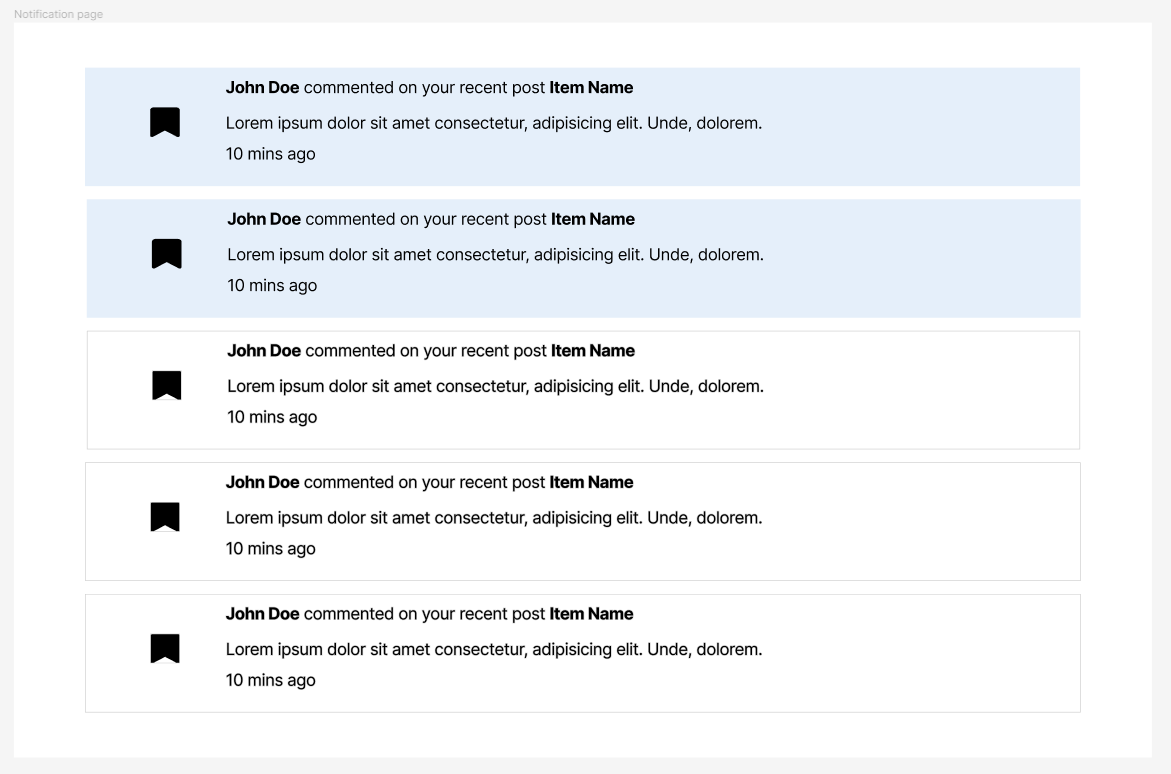


Figure 27: Notification page interface

### 3.11 Profile page interface

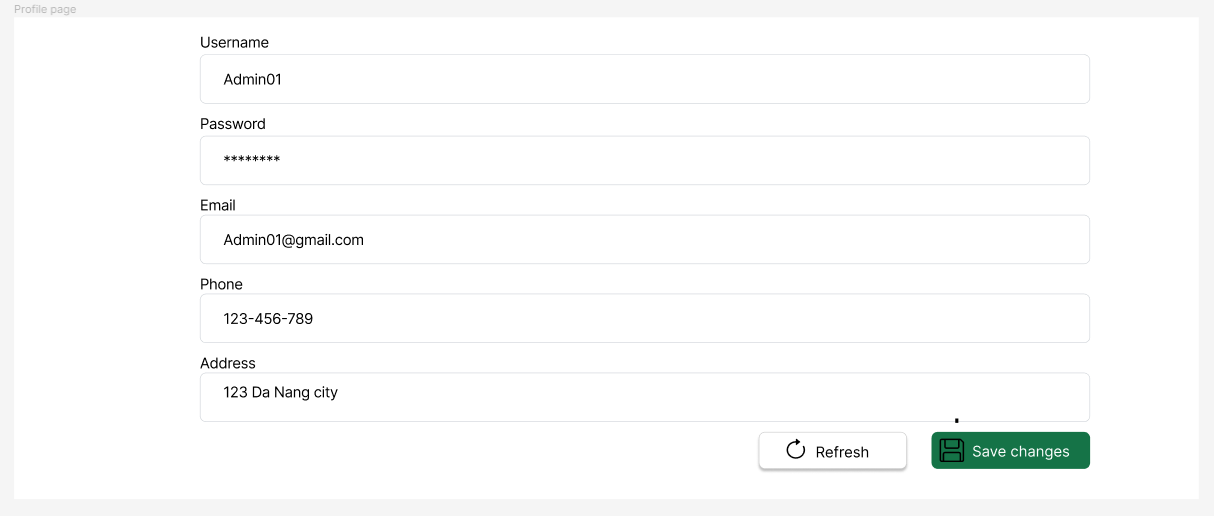


Figure 28: Profile page interface

### 3.12 My bid page interface



Figure 29: My bid page interface

### 3.13 Bid detail view page interface

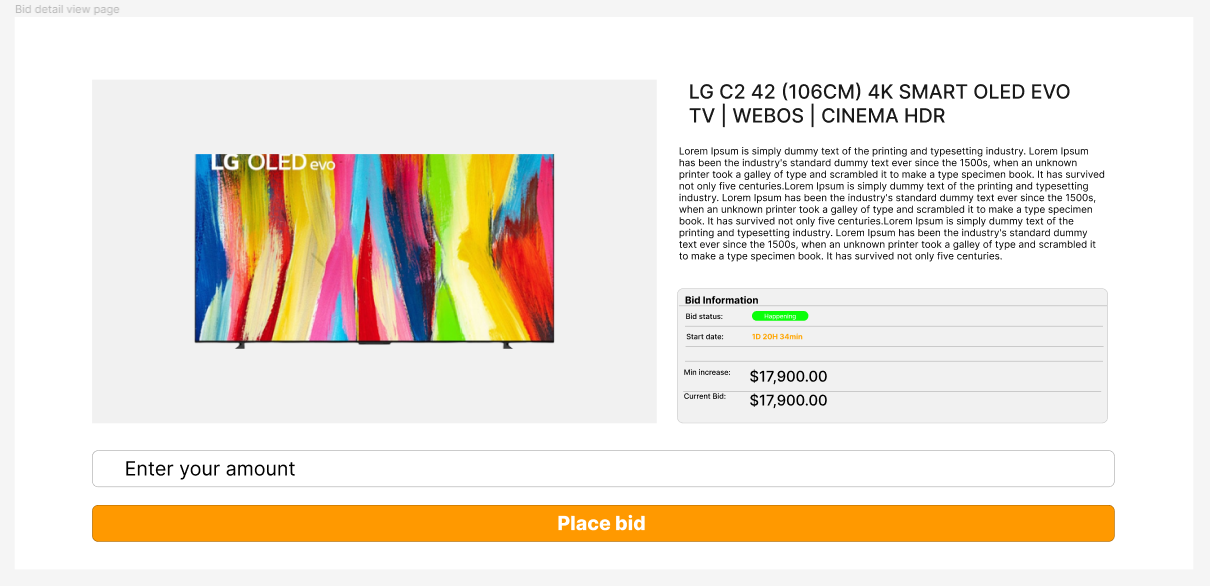


Figure 30: Bid detail view page interface

### 3.14 Bid detail create or edit page interface

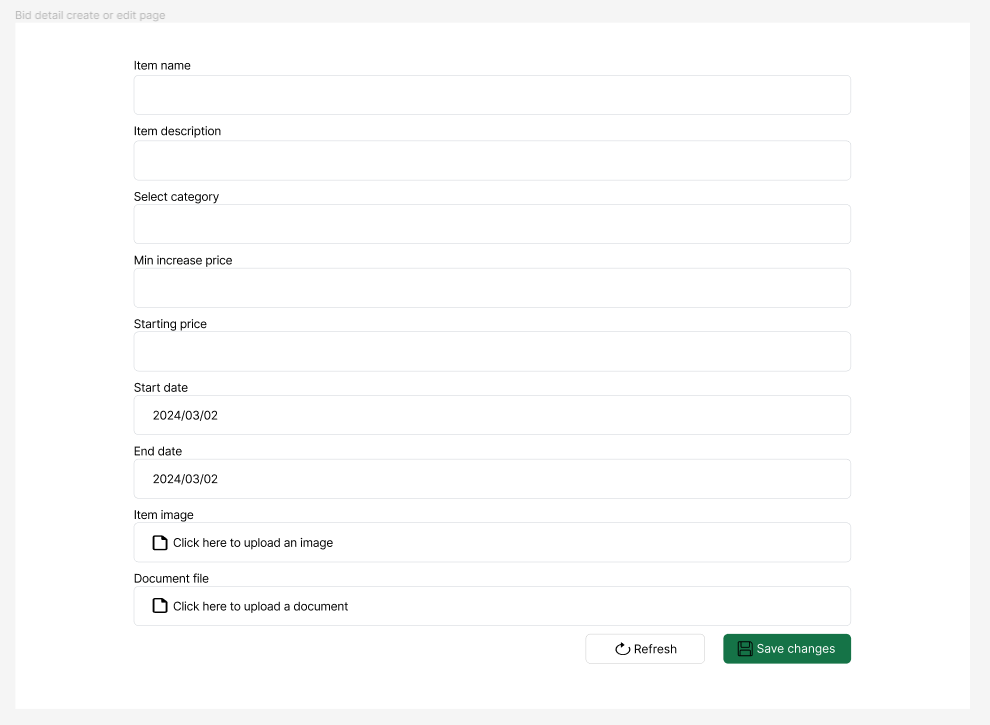


Figure 31: Bid detail create or edit page interface

### 3.15 Report items page interface

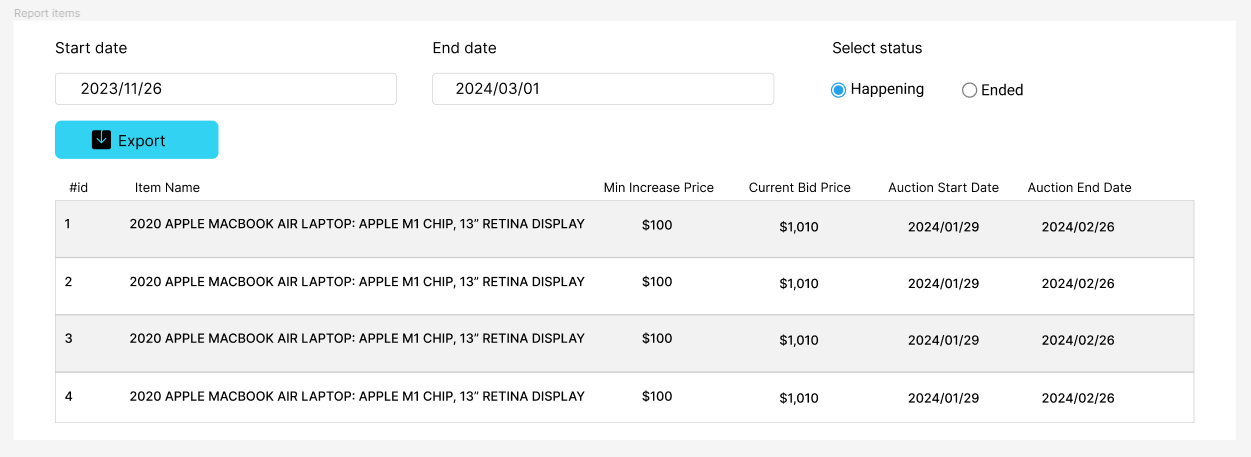


Figure 32: Report items page interface

## 4. Analyze system functions

- System actors include:

***1. User:*** This is the basic user of the system. They can register, log in, log out, update their profile and view user rankings.

***2. Manager (Admin):*** The manager has the right to manage categories and users in the system. They can add, edit, or delete categories and manage users.

***3. Buyer:*** Buyers can bid, view auctions, rate auctions, view auction listings, view announcements, and search for items.

***4. Seller:*** Sellers can view auctions, rate auctions, view auction listings, view announcements, search for items, and view auction history.

***5. Automation System:*** The automated system will generate notifications, update item status and bids, and generate auction ratings.

### 4.1 Usecase overview

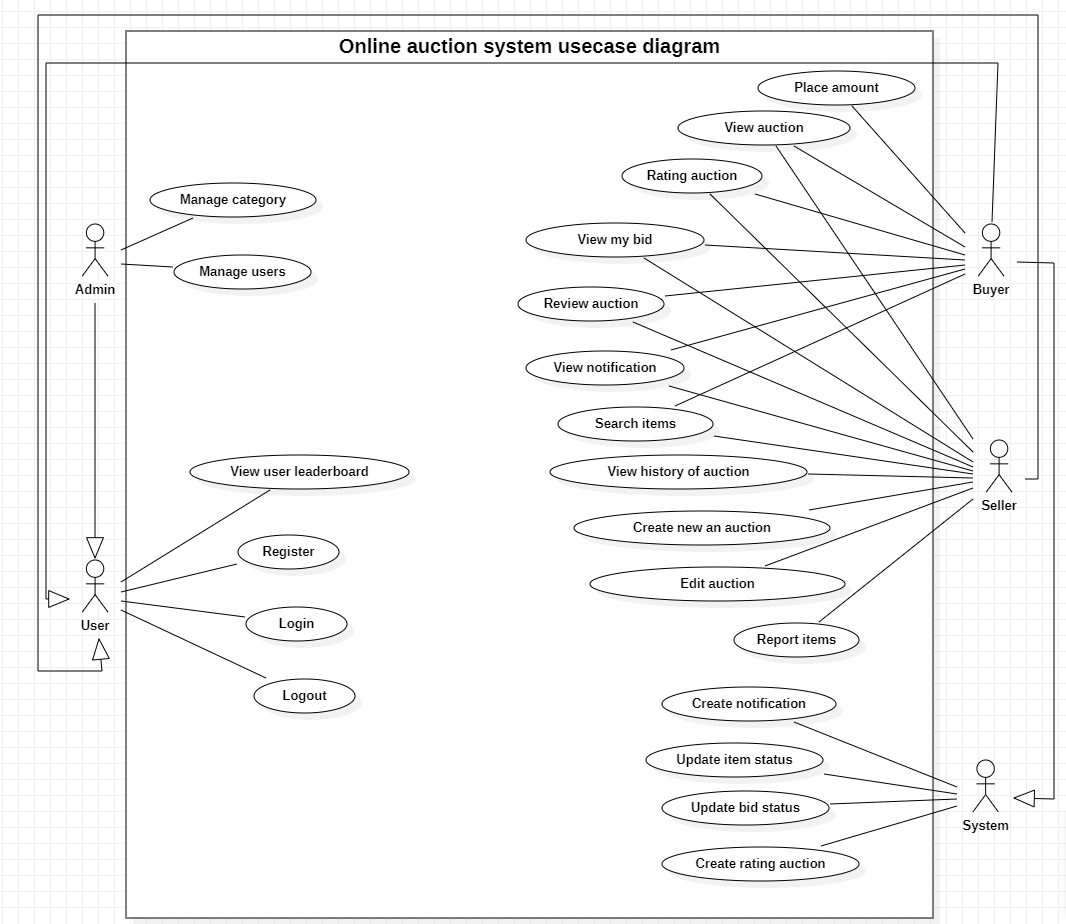


Figure 33: General use case of the system

### 4.1 Functions of the User Object

- User overview use case

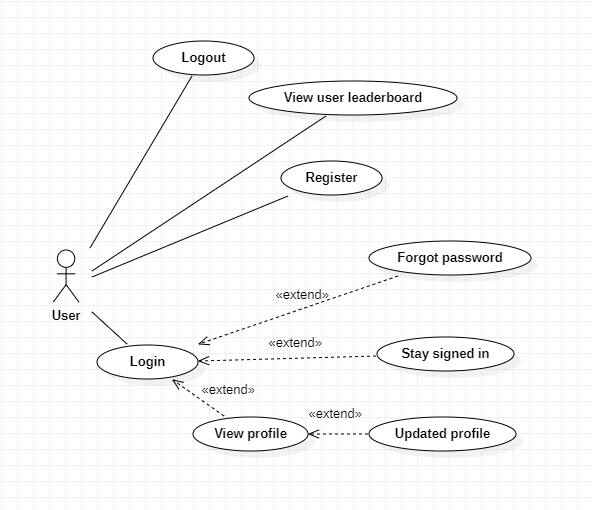


Figure 34: User overview use case

### 4.1.1 Login function

|  |  |
| --- | --- |
| Use case name | Log in |
| Usecase code | UC01 |
| Actor | User |
| Use case description | Allow users to log in to the system |
| Request implementation | The user already has an account in the system |
| Preconditions | The user is not logged in |
| Postconditions | The user is redirected to the home page |
| Main event stream | 1. The user goes to the login page 2. The user enters username and password 3. The user presses the login button 4. The system checks login information 5. If the login information is valid, the system redirects the user to the home page |
| Secondary event stream | 1. Login failed 2. If the login information is invalid, the system will display the error message "Invalid login information" 3. Return to step 2 of the main event flow |

- Activity diagram

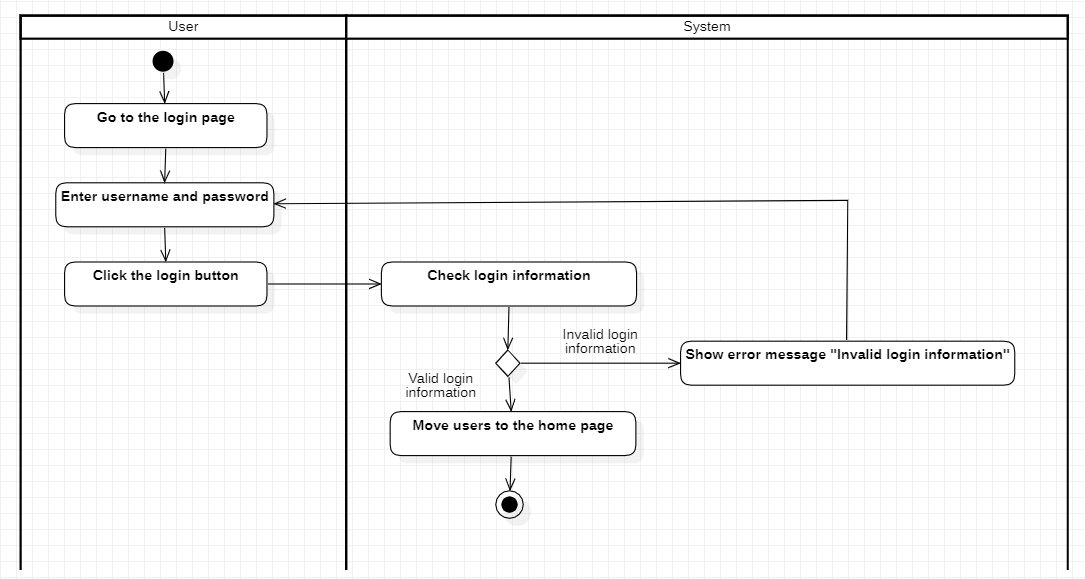


Figure 35: Activity diagram of the login function

### 4.1.2 User remember function

|  |  |
| --- | --- |
| Use case name | Remember the user |
| Usecase code | UC02 |
| Actor | User |
| Use case description | Allows users to register into the system |
| Request implementation | Do not have |
| Preconditions | Do not have |
| Postconditions | Users entering the system do not need to log in again |
| Main event stream | 1. The user goes to the login page 2. The user enters username and password 3. The user clicks on the Remember checkbox 4. The user presses the login button 5. The system checks login information 6. The system saves user information (including ID, username, full name, role, and token) into localStorage |

- Activity diagram

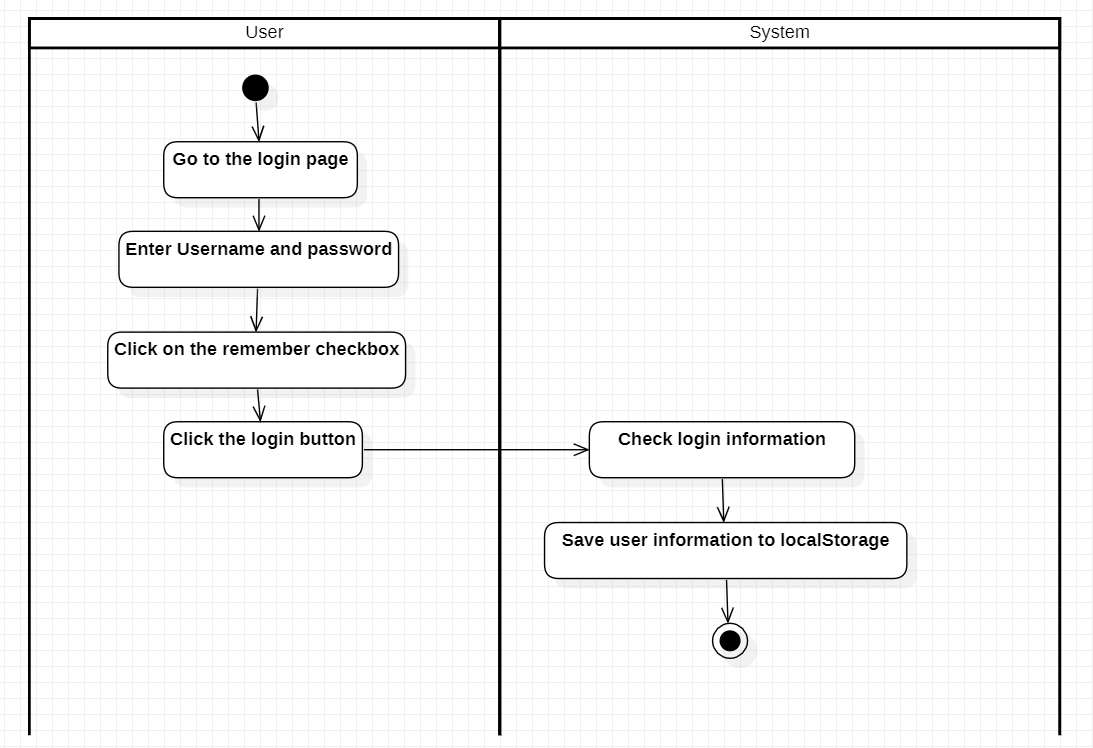


Figure 36: Activity diagram of the user remember function

### 4.1.3 Registration function

|  |  |
| --- | --- |
| Use case name | Register |
| Usecase code | UC03 |
| Actor | User |
| Use case description | Allows users to register into the system |
| Request implementation | Do not have |
| Preconditions | Do not have |
| Postconditions | User-created registration successfully |
| Main event stream | 1. The user goes to the registration page 2. The user enters personal information 3. The user presses the register button 4. The system checks registration information 5. If the registration information is valid, the system adds the user information to the database 6. The system redirects the user to the login page. |
| Secondary event stream | 1. registration failed 2. If the registration information is invalid, the system will display the error message "Invalid registration information" 3. Return to step 2 of the main event flow |

- Activity diagram

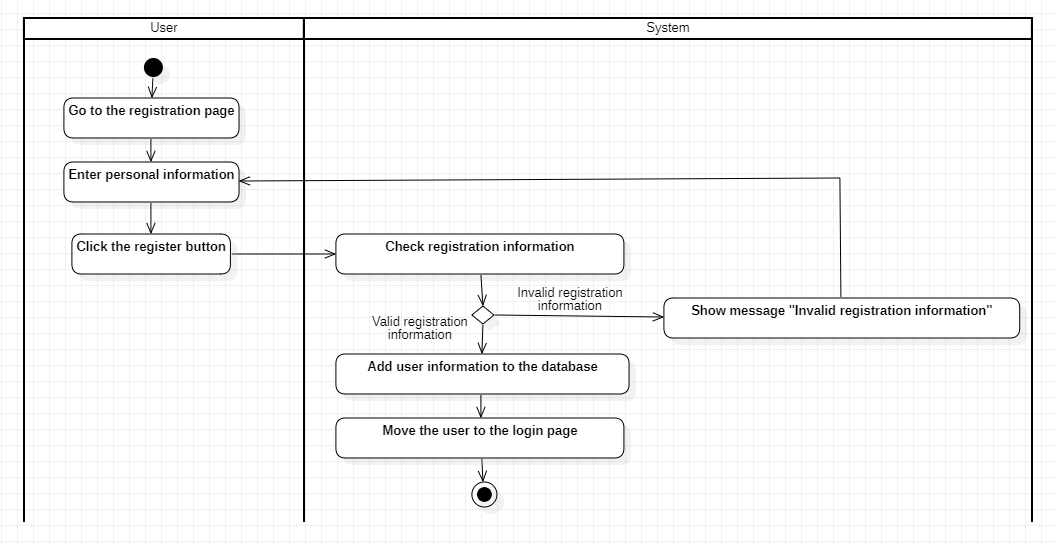


Figure 37: Activity diagram of the registration function

### 4.1.4 Forgot password function

|  |  |
| --- | --- |
| Use case name | Forgot password |
| Usecase code | UC04 |
| Actor | User |
| Use case description | Re-issue a new password to the user via email |
| Request implementation | The user already has an account in the system |
| Preconditions | The user is not logged in |
| Postconditions | Users are re-issued a new password via email |
| Main event stream | 1. The user goes to the login page 2. The user clicks on the forgotten password 3. The user enters the email registered with the system 4. The user presses send 5. Does the email checking system already exist in the system? 6. If it exists, the system sends the new password via the user's email 7. The system saves new passwords sent via email into the database 8. The system redirects the user to the login page |
| Secondary event stream | 1. Sending email failed 2. If the email does not exist, the system will notify 'Email is invalid' |

- Activity diagram

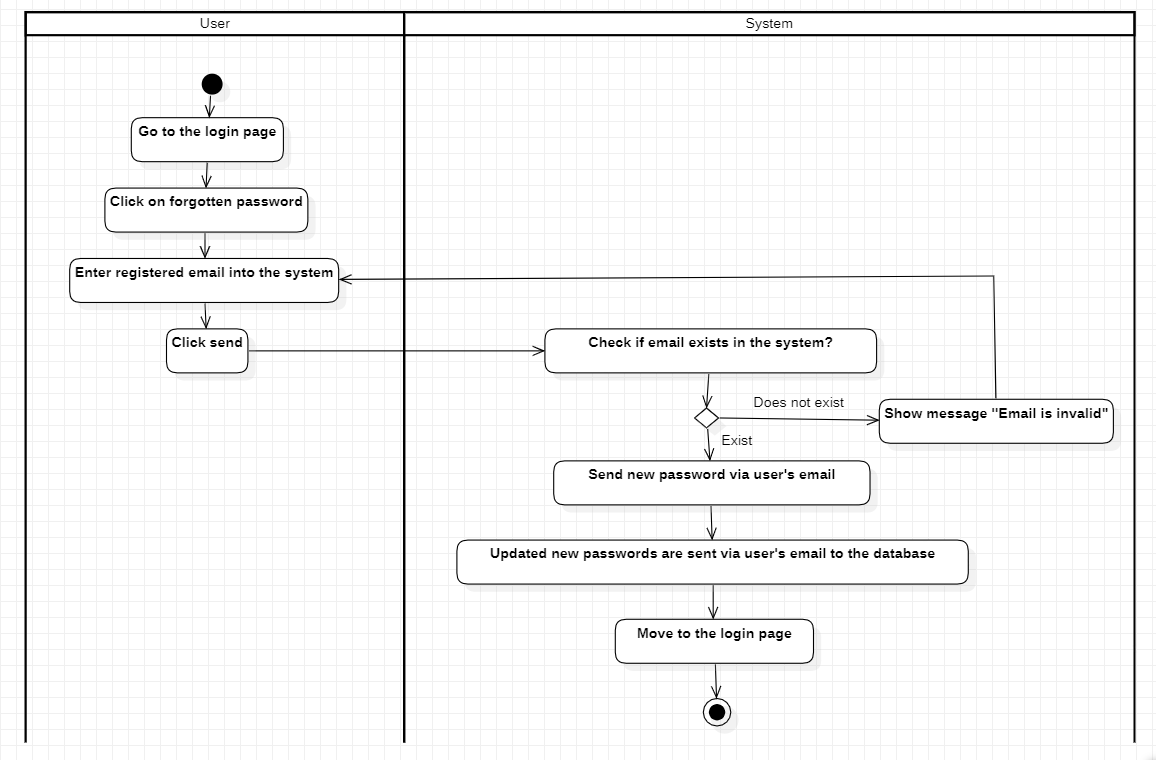


Figure 38: Activity diagram of forgotten password function

### 4.1.5 Logout function

|  |  |
| --- | --- |
| Use case name | Log out |
| Usecase code | UC05 |
| Actor | User |
| Use case description | Allow users to log out of the system |
| Request implementation | Do not have |
| Preconditions | The user is logged into the system |
| Postconditions | The user successfully logs out and returns to the login page |
| Main event stream | 1. The user is logged in to the system 2. The user clicks the logout button 3. The system displays the message "Confirm whether the user is sure to log out" 4. The user accepts logout 5. The system deletes the user's sessionStorage or localStorage 6. The system redirects the user to the login page. |
| Secondary event stream | 1. Logout failed 2. If the user does not accept log out 3. Return to step 1 of the main flow of events |

- Activity diagram

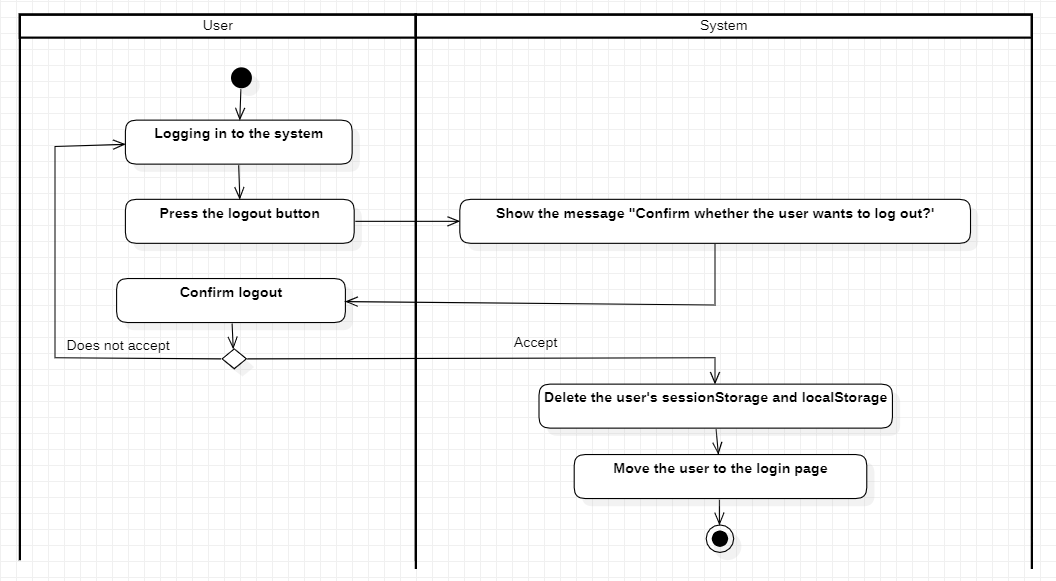


Figure 39: Activity diagram of logout function

### 4.1.6 User information update function

|  |  |
| --- | --- |
| Use case name | Update user information |
| Usecase code | UC06 |
| Actor | User |
| Use case description | Allows users to update personal information |
| Request implementation | The user must have role 0 or 1 |
| Preconditions | The user is logged into the system |
| Postconditions | The user successfully updated personal information |
| Main event stream | 1. The user is logged in to the system 2. The user clicks on the My Profile button in the header 3. The system switches to the My Profile page 4. The system checks the user ID and token 5. The system retrieves user information by id 6. The system returns data to the interface 7. The user enters all new personal information. 8. The user presses the save button 9. The system saves user information by userId into the database 10. The system displays a notification that the user's information has been successfully changed. 11. The system returns to the previous page before entering the My Profile page |
| Secondary event stream | 1. Change information failed 2. The user did not enter enough new information 3. The system requires the user to enter 4. Return to step 7, main event flow |

- Activity diagram

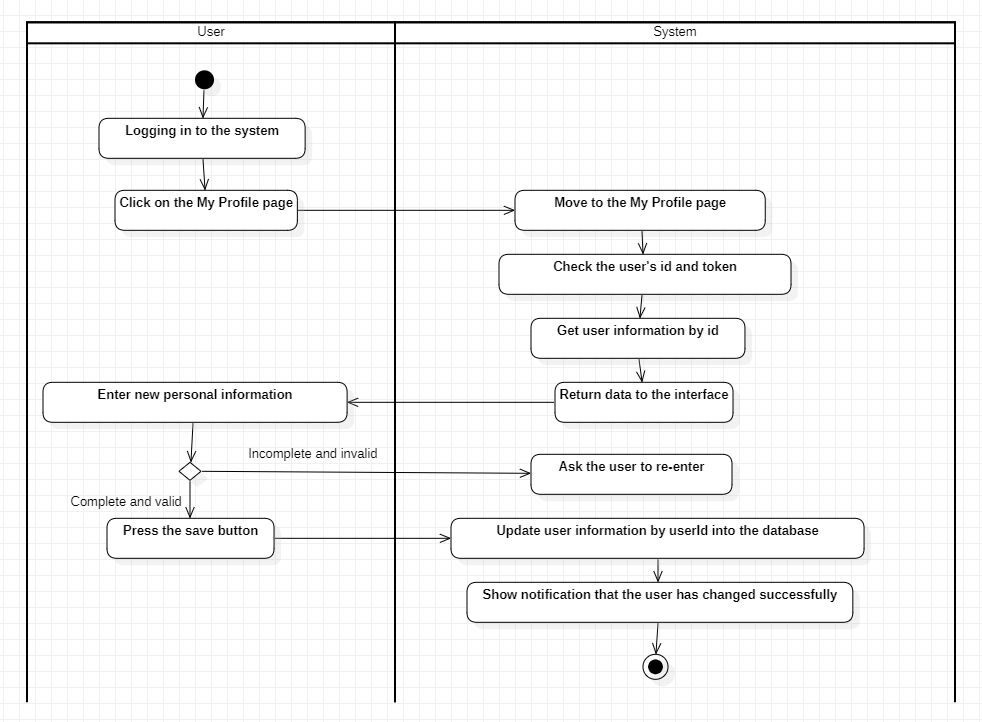


Figure 40: Activity diagram of the user information update function

### 4.1.7 Function allows users to lock their accounts

|  |  |
| --- | --- |
| Use case name | Lock personal accounts |
| Usecase code | UC07 |
| Actor | User |
| Use case description | Allow users to lock personal accounts |
| Request implementation | The user must have role 0 or 1 |
| Preconditions | The user is logged into the system |
| Postconditions | The user successfully locked their account |
| Main event stream | 1. The user is logged in to the system 2. The user clicks the My Profile button in the header 3. The system now switches to the My Profile page 4. The system checks the user ID and token 5. The system retrieves user information by id 6. The system returns data to the interface 7. The user presses the switch to lock the account. 8. The system changes the value of the user's isActive field to false 9. The user presses the save button 10. The system displays the message 'Confirm if the user is sure he wants to lock his account?' 11. The user confirms the account lock 12. The system updates the IsActive field in the user's database to false   13. The system deletes the user's sessionStorage and localStorage.   1. The system redirects the user to the login page. |
| Secondary event stream | 1. Account lock failed 2. The user did not confirm the account lock 3. The system will change the user's isActive field back to true 4. The system returns to step 4. |

- Activity diagram

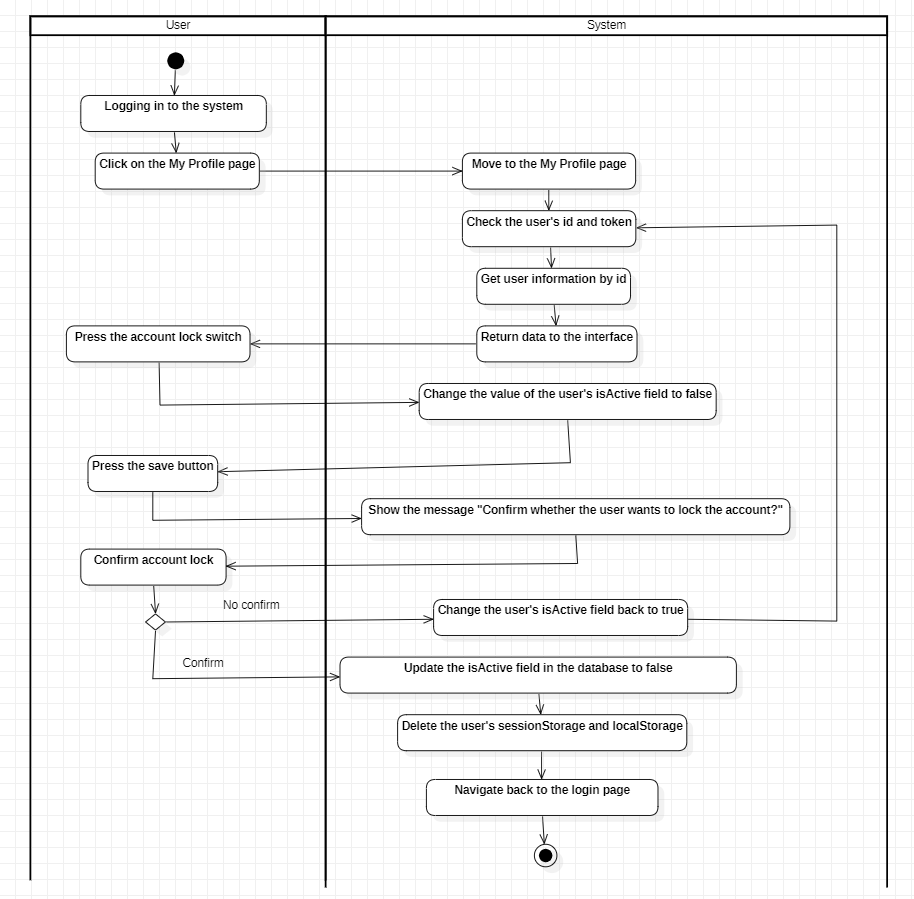


Figure 41: Activity diagram of the function that allows users to self-lock their accounts

### 4.1.7 Function to view user rankings

|  |  |
| --- | --- |
| Use case name | View user rankings |
| Usecase code | UC08 |
| Actor | User |
| Use case description | Allows users to view rankings |
| Request implementation | Do not have |
| Preconditions | The user is logged into the system |
| Postconditions | The user will successfully view the user rankings |
| Main event stream | 1. The user is logged in to the system  2. The user clicks the User Leaderboard button on the header  3. The system switches to the User Leaderboard page  4. The system returns the evaluation data table of all users in the system to the interface. |

- Activity diagram

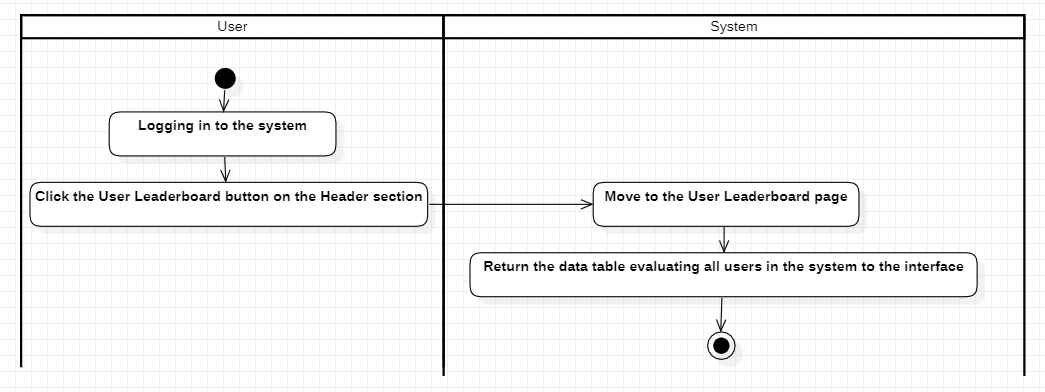


Figure 42: Activity diagram of viewing user rankings

### 4.2 Functions of the Manager object (Admin)

- Use case overview of the manager



Figure 43: Usecase overview of the manager

### 4.2.1 User management function

|  |  |
| --- | --- |
| Use case name | user management |
| Usecase code | UC09 |
| Actor | Manager |
| Use case description | Allows managers to view information of all users in the system |
| Request implementation | The user must have role 2 |
| Preconditions | The manager has logged into the system |
| Postconditions | The manager views the information of all users in the system successfully |
| Main event stream | 1. The manager is logged in to the system 2. The administrator clicks the User Management button in the header 3. The system switches to the User Management page 4. The system retrieves information of all users in the system in the database 5. The system returns a data table with information about all users in the system to the interface |

- Activity diagram

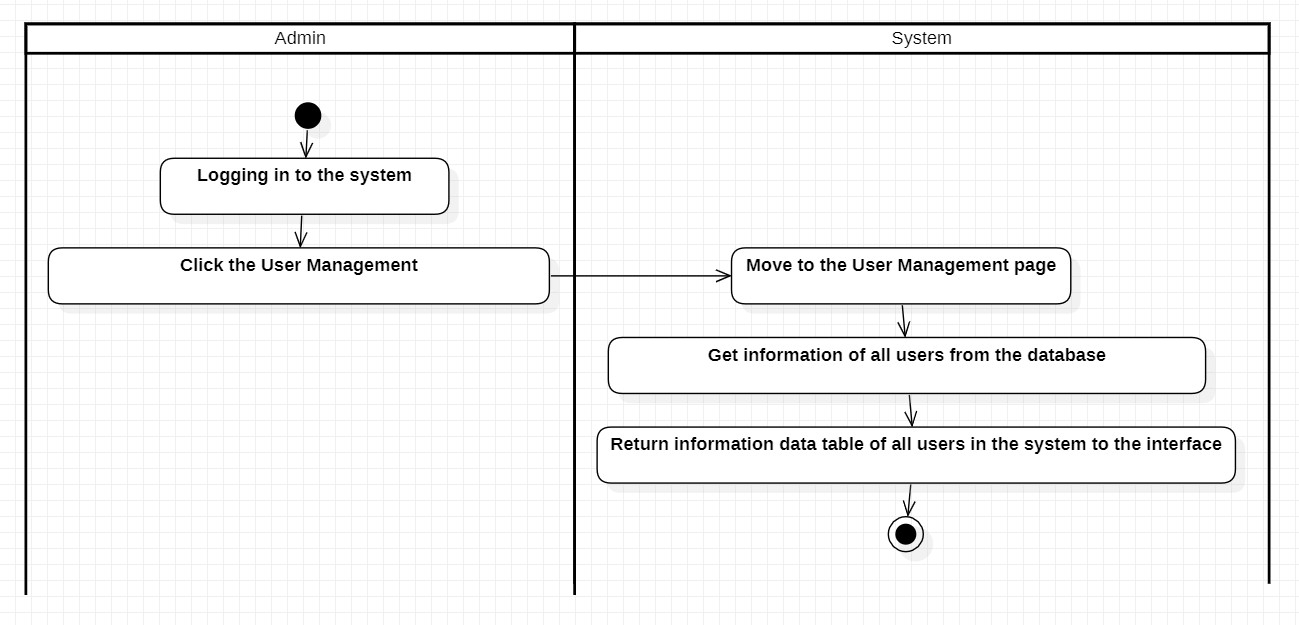
* 1. 

Figure 44: Activity diagram of user management

### 4.2.2 User blocking function

|  |  |
| --- | --- |
| Use case name | Block users |
| Usecase code | UC10 |
| Actor | Manager |
| Use case description | Allows administrators to block users |
| Request implementation | The user must have role 2 |
| Preconditions | The manager has logged into the system |
| Postconditions | The administrator successfully blocked the user |
| Main event stream | 1. The manager is logged in to the system 2. The manager clicks the User Management button in the header 3. The system switches to the User Management page 4. The system retrieves information about all user data in the database 5. The system returns a data table with information about all users in the system to the interface 6. The manager presses the Lock account button 7. The system will update the user's isActive data field to false 8. The system saves changed data to the database 9. Refresh system |

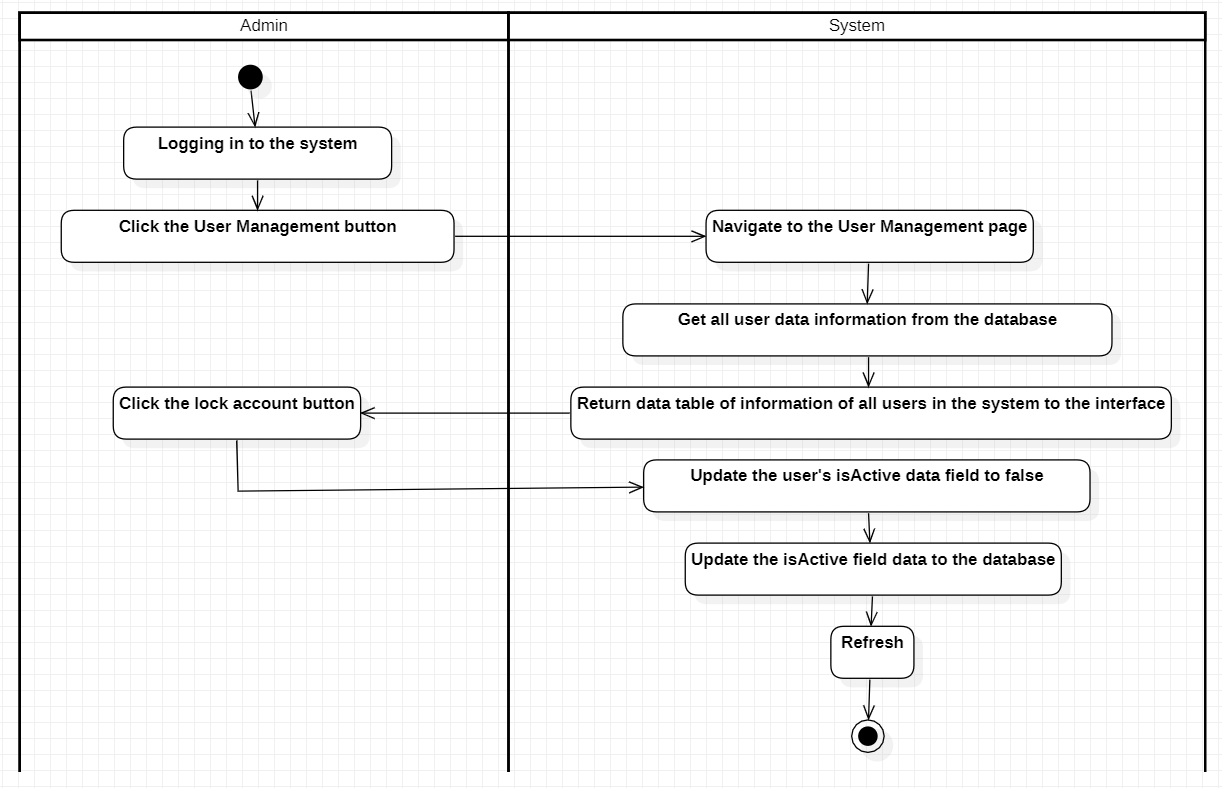
* 1. - Activity diagram
  2. 

Figure 45: Activity diagram of user blocking function

### 4.2.2 Category Management Function

### 4.2.2.1. Add category function

|  |  |
| --- | --- |
| Use case name | Add categories |
| Usecase code | UC11 |
| Actor | Manager |
| Use case description | Allows managers to add categories |
| Request implementation | The user must have role 2 |
| Preconditions | The manager has logged into the system |
| Postconditions | The manager will successfully add a category |
| Main event stream | 1. The manager is logged in to the system 2. The manager clicks the Category Management button in the header 3. The system now switches to the Category Management page 4. The system retrieves information from all categories in the database 5. The system returns data to the interface 6. The manager enters complete information about the new category 7. The manager presses the save button 8. The system checks whether the category ID already exists 9. If not, the system creates a new category. 10. The system saves newly created category information to the database 11. The system resets the form |
| Secondary event stream | 1. The manager did not fully enter the field 2. The system requires full input |

- Activity diagram

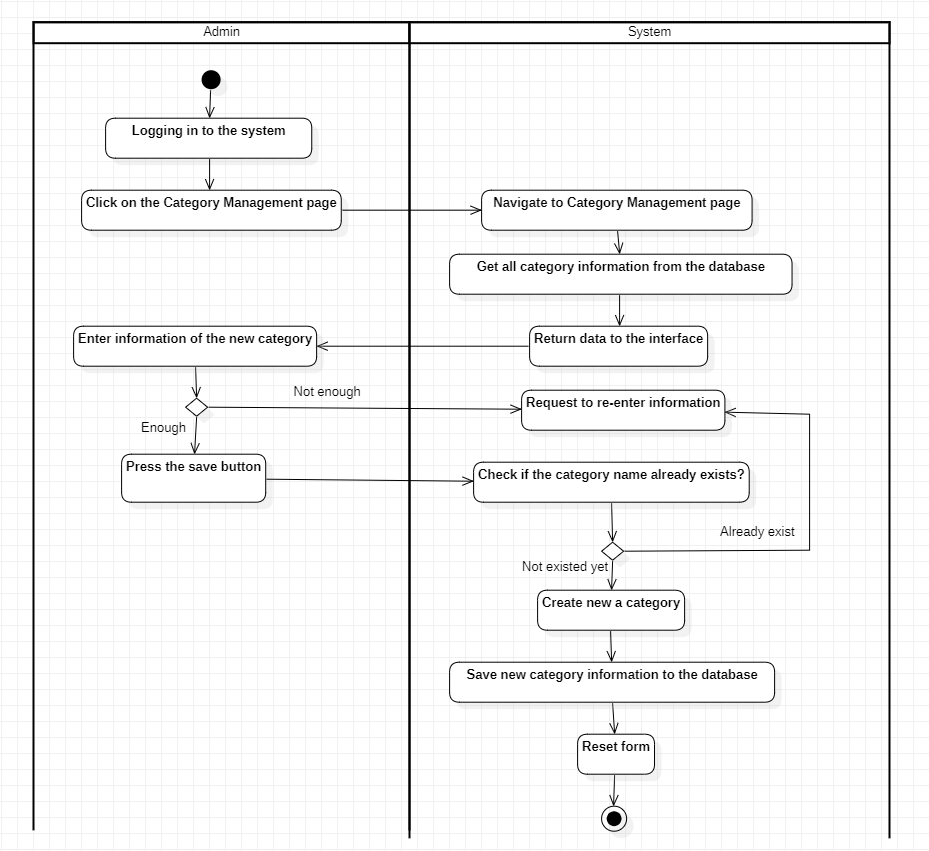


Figure 46: Activity diagram of the add category function

### 4.2.2.2 Edit category function

|  |  |
| --- | --- |
| Use case name | Edit Category |
| Usecase code | UC12 |
| Actor | Manager |
| Use case description | Allows administrators to edit categories |
| Request implementation | The user must have role 2 |
| Preconditions | The manager has logged into the system |
| Postconditions | The manager will successfully edit a category |
| Main event stream | 1. The manager is logged in to the system 2. The manager clicks the Category Management button in the header 3. The system now switches to the Category Management page 4. The system retrieves information from all categories in the database 5. The system returns data to the interface 6. The manager clicks the edit category button 7. The system will set the value of the category to be edited into the input 8. The manager fully enters the new information of the category 9. The manager presses the save button system checks whether the category ID already exists 10. If it exists, the system updates the category. 11. The system saves new category information to the database 12. The system resets the form |
| Secondary event stream | 1. The user does not enter the category name field 2. The system will notify you of the request to enter 3. The system returns to step 8 |

- Activity diagram

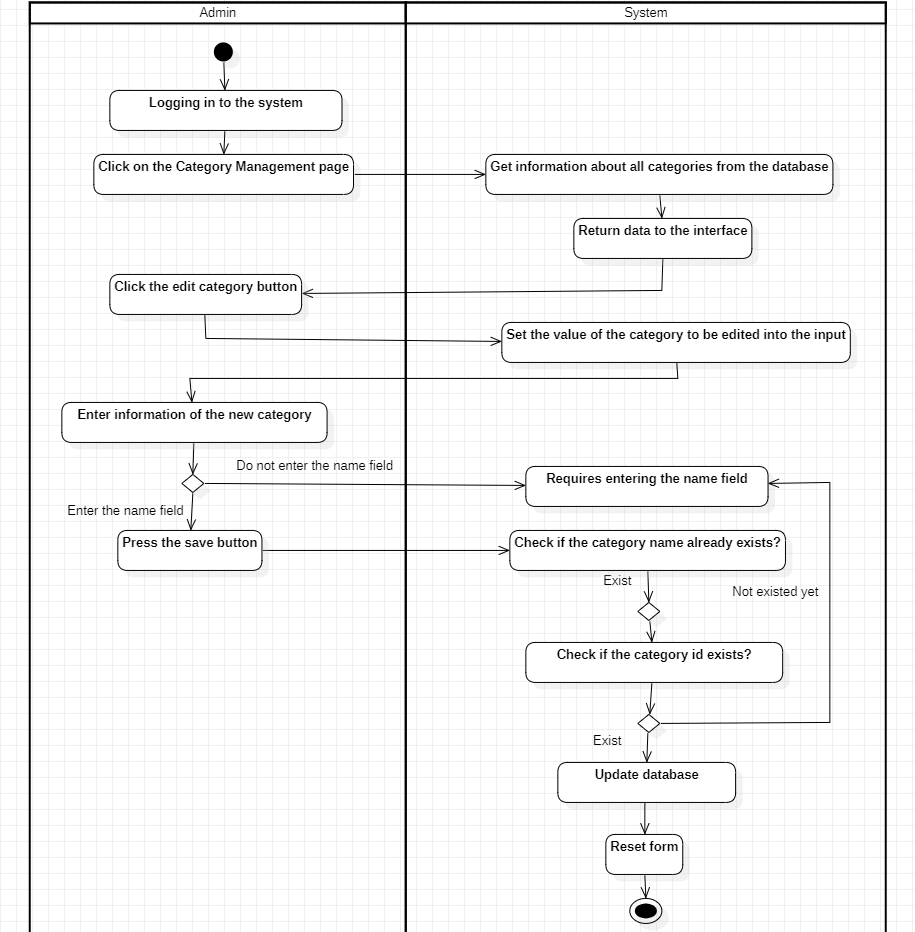


Figure 47: Activity diagram of the directory editing function

### 4.3 Functions of Buyer and Seller Objects

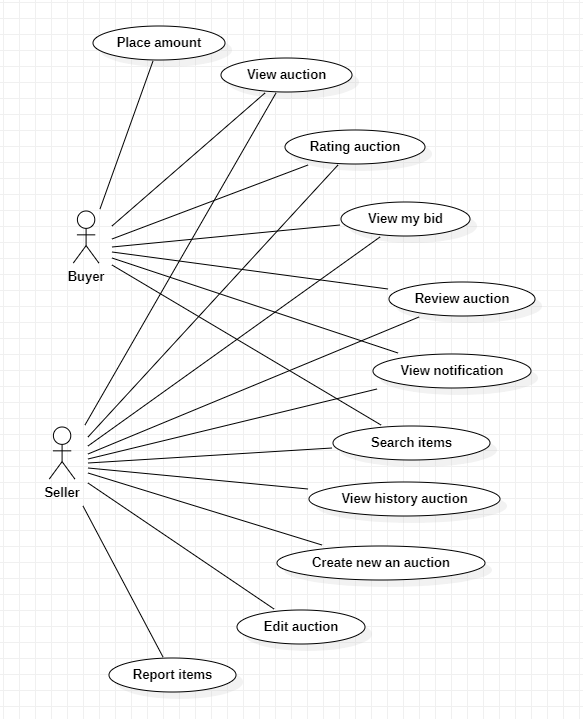


Figure 48: Usecase overview of seller/buyer

### 4.3.1 General functions

### 4.3.1.1 Auction viewing function

|  |  |
| --- | --- |
| Use case name | Watch the auction |
| Usecase code | UC13 |
| Actor | Buyer/seller |
| Use case description | Allows Buyers/Sellers to view the auction |
| Request implementation | The user must have role 0 or 1 |
| Preconditions | The buyer/seller has logged into the system |
| Postconditions | The buyer/seller will successfully view the auction |
| Main event stream | 1. Buyer/ seller are logged into the system  2. Buyer/ seller click the View Auction detail button  3. The system switches to the bid detail view page |

- Activity diagram

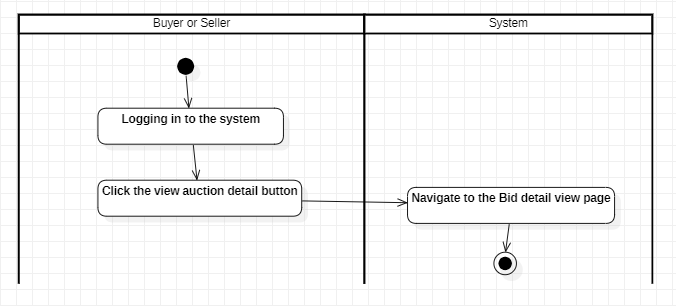


Figure 49: Activity diagram of the auction viewing function

### 4.3.1.2 Evaluation function

|  |  |
| --- | --- |
| Use case name | Buyer/seller reviews |
| Usecase code | UC14 |
| Actor | Buyer/seller |
| Use case description | Allows Buyers/Sellers to rate each other |
| Request implementation | + User must have role 0 or 1  + The auction must end  + Only the auctioneer and the auction winner are allowed to rate |
| Preconditions | The buyer/seller has logged into the system |
| Postconditions | The buyer/seller will evaluate the success |
| Main event stream | 1. Buyer/ seller is logging in to the system  2. Buyer/ seller clicks on the view auction detail button  3. The system switches to the bid detail view page  4. Buyer/ seller rate by choosing the number of stars (from 0-5)  5. Buyer/ seller clicks on submit form button  6. The system adds the evaluated score to the database |

- Activity diagram

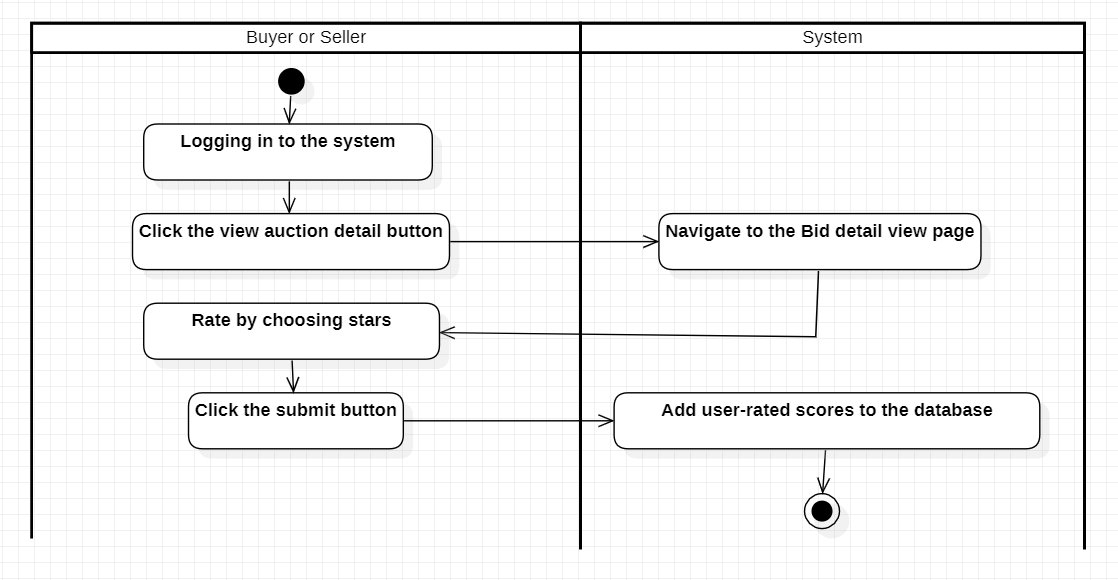


Figure 50: Activity diagram of the evaluation function

### 4.3.1.3 Function to view the list of auctions that buyers/sellers have participated in

|  |  |
| --- | --- |
| Use case name | See the auction buyers/sellers participated in |
| Usecase code | UC15 |
| Actor | Buyer/seller |
| Use case description | Allows Buyers/Sellers to see the auction, which participated in |
| Request implementation | The user must have role 0 or 1 |
| Preconditions | The buyer/seller has logged into the system |
| Postconditions | Buyers/sellers will be able to see a list of auctions in which they participate |
| Main event stream | 1. Buyer/seller is logging in to the system  2. Buyer/seller clicks on the My Bidder button in the header  3. The system switches to the My Bidder page  4. The system checks the user's id and token  5. The system retrieves product information according to the user's ID  6. The system returns data to the interface |

- Activity diagram

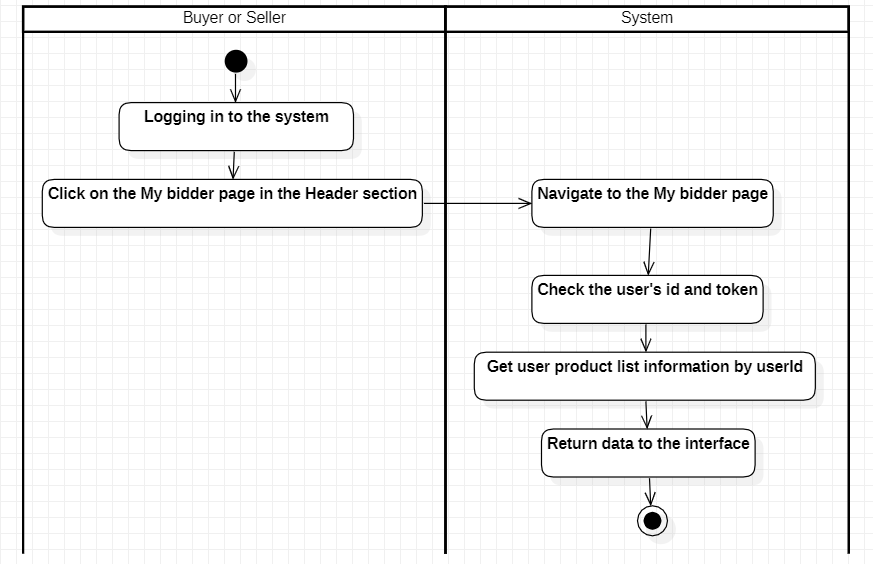


Figure 51: Activity diagram of the function to view the list of auctions of participating buyers/sellers

### 4.3.1.4 Notification viewing function

|  |  |
| --- | --- |
| Use case name | See announcement |
| Usecase code | UC16 |
| Actor | Buyer/seller |
| Use case description | Allow Buyers/Sellers to view notifications |
| Request implementation | The user must have role 0 or 1 |
| Preconditions | The buyer/seller has logged into the system |
| Postconditions | The buyer/seller will see a list of success notifications |
| Main event stream | 1. Buyer/ seller are logged into the system  2. Buyer/ seller click the Notifications button in the header  3. The system switches to the Notification page  4. The system checks the user's id and token  5. The system retrieves information on all notifications according to the user's ID  6. The system returns data to the interface |

- Activity diagram

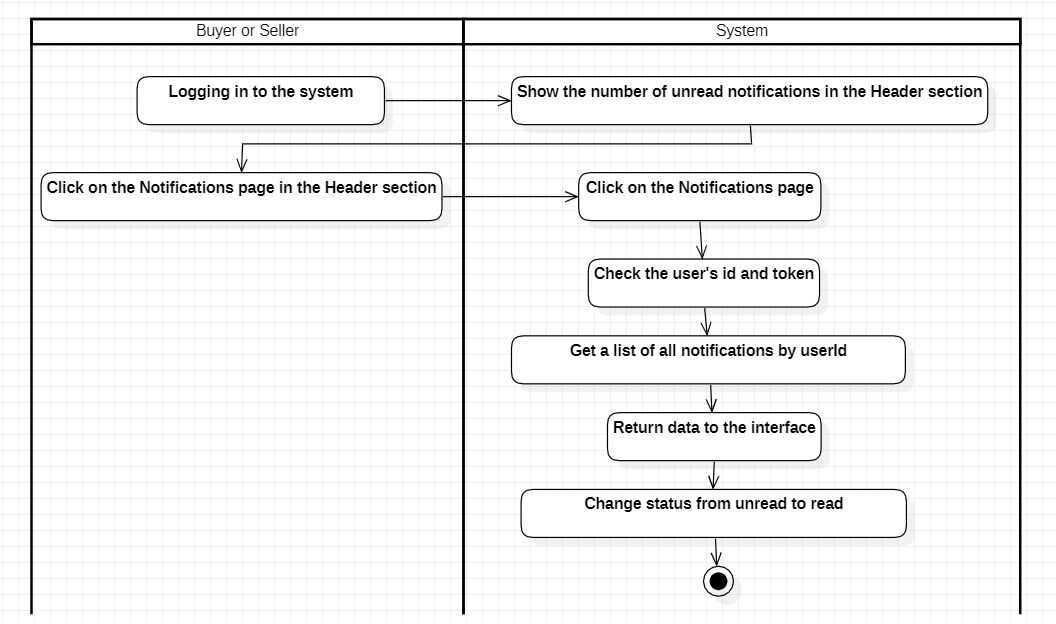


Figure 52: Activity diagram of notification viewing function

### 4.3.1.5 Search function by product name

|  |  |
| --- | --- |
| Use case name | Search by product name |
| Usecase code | UC17 |
| Actor | Buyer/seller |
| Use case description | Allow Buyers/Sellers to view notifications |
| Request implementation | + User must have role 0 or 1 |
| Preconditions | The buyer/seller has logged into the system |
| Postconditions | Buyers/sellers will see a list of product names they searched for |
| Main event stream | 1. Buyer/ seller is logged in to the system  2. Buyer/ seller enters the product name they want to search for in the input in the header section  3. Buyer/seller presses the search button  4. The system moves to the Search Result page  5. The system checks the entered ''keyword''  6. The system retrieves all items with names similar to ''keyword'' (absolute search)  7. The system returns data to the interface |

- Activity diagram

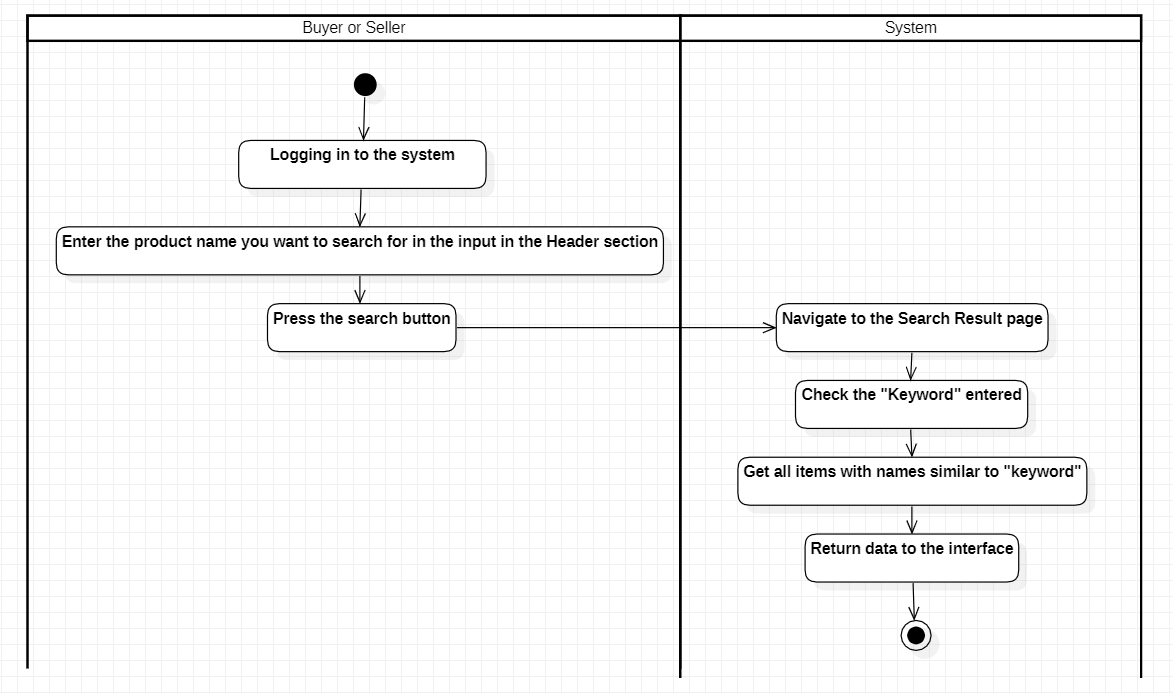


Figure 53: Activity diagram of search function by product name

### 4.3.2 Buyer's functions

### 4.3.2.1 Pricing function

|  |  |
| --- | --- |
| Use case name | Place bid |
| Usecase code | UC18 |
| Actor | Buyer |
| Use case description | Allow buyers to set prices |
| Request implementation | + The user must have role 0  + The auction is going on  + Each buyer can only bid once |
| Preconditions | The buyer has logged into the system |
| Postconditions | The buyer will bid successfully |
| Main event stream | 1. The buyer is logged in to the system  2. The buyer clicks the View Auction detail button  3. The system switches to the bid detail view page  4. The buyer enters the price  5. The buyer presses the place bid button  6. The system price check system  7. The system saves auction information to the database  8. The system updates the current price and bid date on the interface  9. The system hides the bidding form |

- Activity diagram

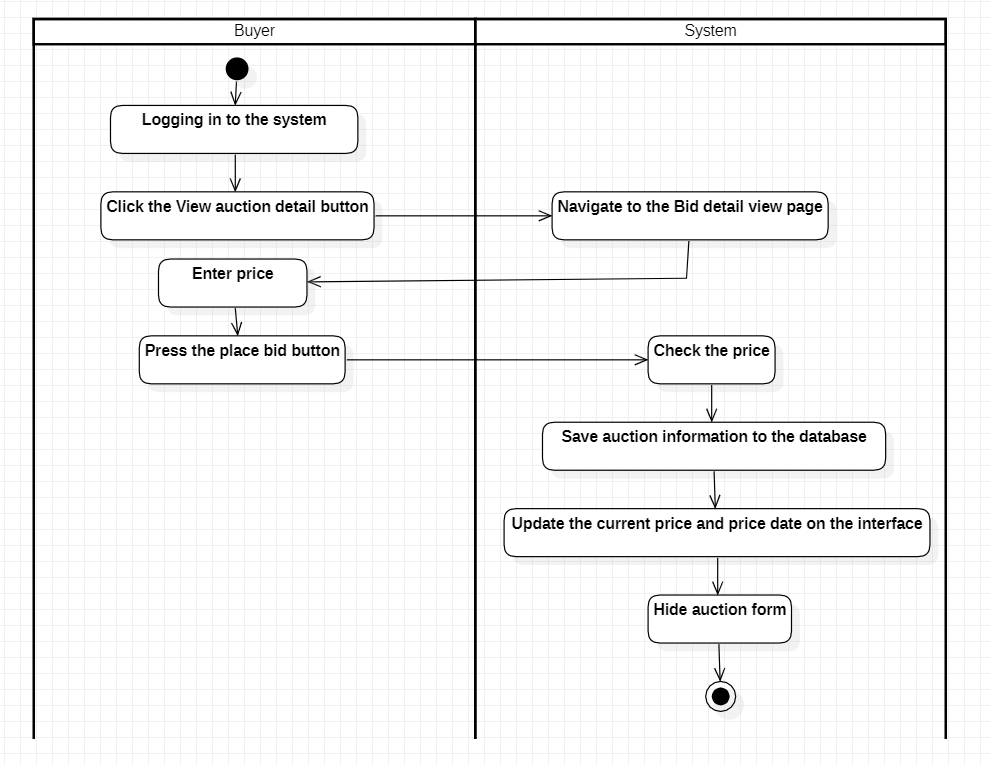


Figure 54: Activity diagram of the bidding function for buyers

### 4.3.3 Seller's functions

### 4.3.3.1 Function to view auction history

|  |  |
| --- | --- |
| Use case name | View auction history |
| Usecase code | UC19 |
| Actor | Seller |
| Use case description | Allows sellers to view auction history |
| Request implementation | + The user must have role 1  + There must be an auctioneer |
| Preconditions | The seller has logged into the system |
| Postconditions | Sellers can see the list of users (with hidden information) bidding on their products |
| Main event stream | 1. The seller is logged in to the system  2. The seller clicks the View Auction detail button  3. The system switches to the bid detail view page  4. The seller clicks the History Bid button  5. The system shows the overlay  6. The system checks itemId  7. The system retrieves all auctions of the product by item  8. The system returns data to the interface |

* 1. - Activity diagram
  2. 

Figure 55: Activity diagram of auction history viewing function for sellers

### 4.3.3.2 Auction creation function

|  |  |
| --- | --- |
| Use case name | Create an auction |
| Usecase code | UC20 |
| Actor | Seller |
| Use case description | Allow sellers to create auctions |
| Request implementation | + The user must have role 1 |
| Preconditions | The seller has logged into the system |
| Postconditions | The seller will successfully create the auction |
| Main event stream | 1. The seller is logged in to the system  2. The seller clicks the Add Item button in the header  3. The system switches to the bid detail create page  4. The seller enters complete information  5. The seller presses the save button  6. The system will save information on auction products into the database  7. The system will move to the bid detail view page |

- Activity diagram

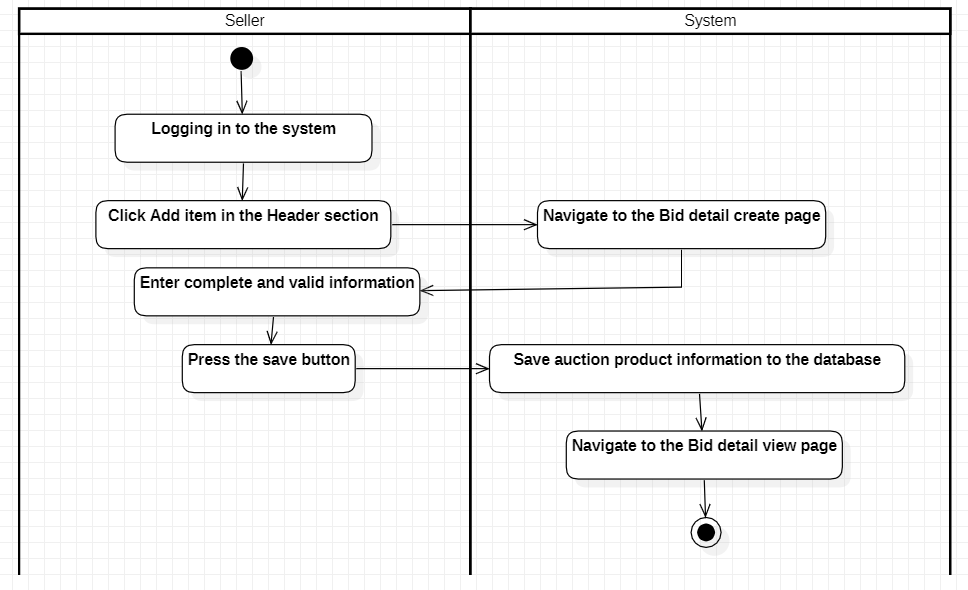


Figure 56: Activity diagram of the auction creation function for sellers

### 4.3.3.3 Auction editing function

|  |  |
| --- | --- |
| Use case name | Edit auction |
| Usecase code | UC21 |
| Actor | Seller |
| Use case description | Allow people to edit auctions |
| Request implementation | + The user must have role 1  + The auction has not yet taken place |
| Preconditions | The seller has logged into the system |
| Postconditions | The seller will successfully create the auction |
| Main event stream | 1. The seller is logged in to the system  2. The seller clicks the View Auction detail button  3. The system switches to the bid detail view page  4. The seller clicks on the edit auction icon  5. The system switches to the bid detail edit page  6. The system returns data to the interface according to the product ID  7. The seller edits product information  8. The seller presses the save button  9. The system will save product change information by ID into the database  10. The system returns to the bid detail view page |

- Activity diagram

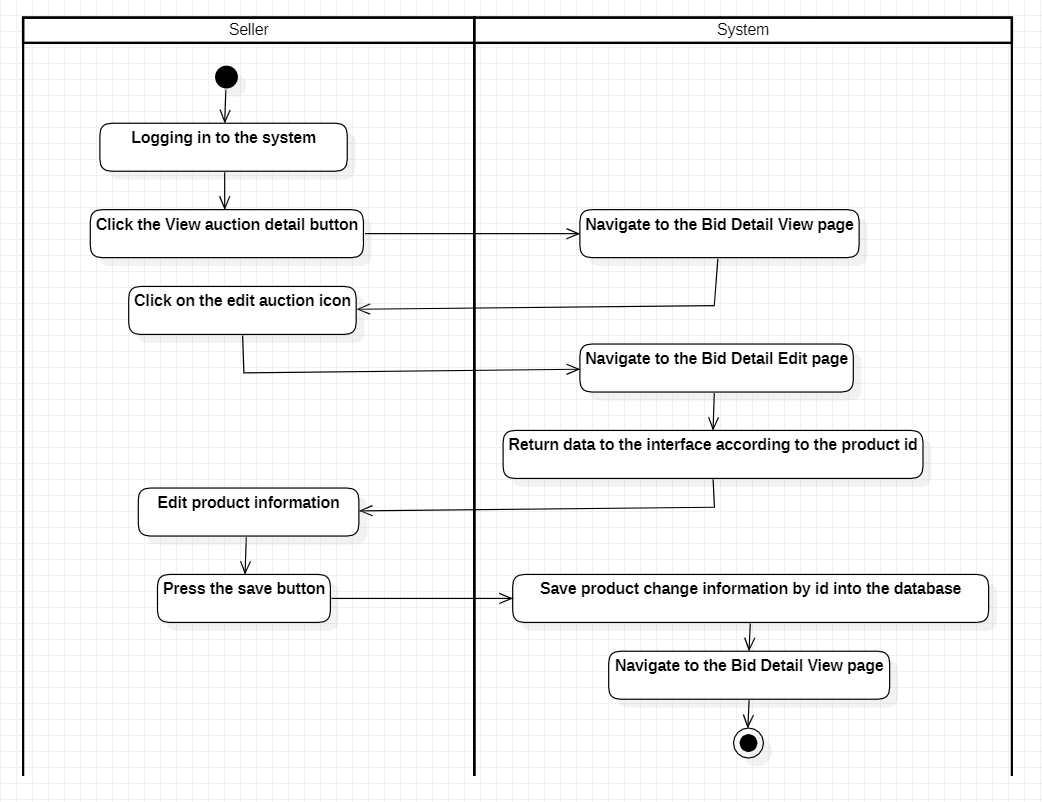


Figure 57: Activity diagram of the auction editing function for sellers

### 4.3.3.4 Function to view product list reports

|  |  |
| --- | --- |
| Use case name | View product report |
| Usecase code | UC22 |
| Actor | Seller |
| Use case description | Allows sellers to view statistics of their auctions |
| Request implementation | + The user must have role 1 |
| Preconditions | The seller has logged into the system |
| Postconditions | The seller will successfully view their auction listing |
| Main event stream | 1. The seller is logged in to the system  2. The seller clicks the Report items button in the header  3. The system switches to the Report Items page  4. The system retrieves all product list information from sellers  5. The system returns data to the interface |

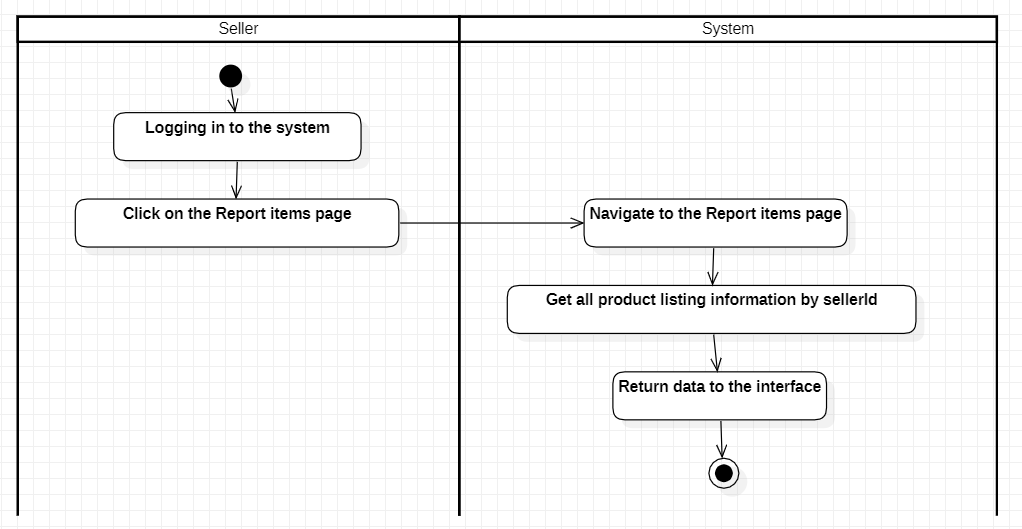
* 1. - Activity diagram
  2. 

Figure 58: Activity diagram of the product list report viewing function for sellers

5. Class diagram

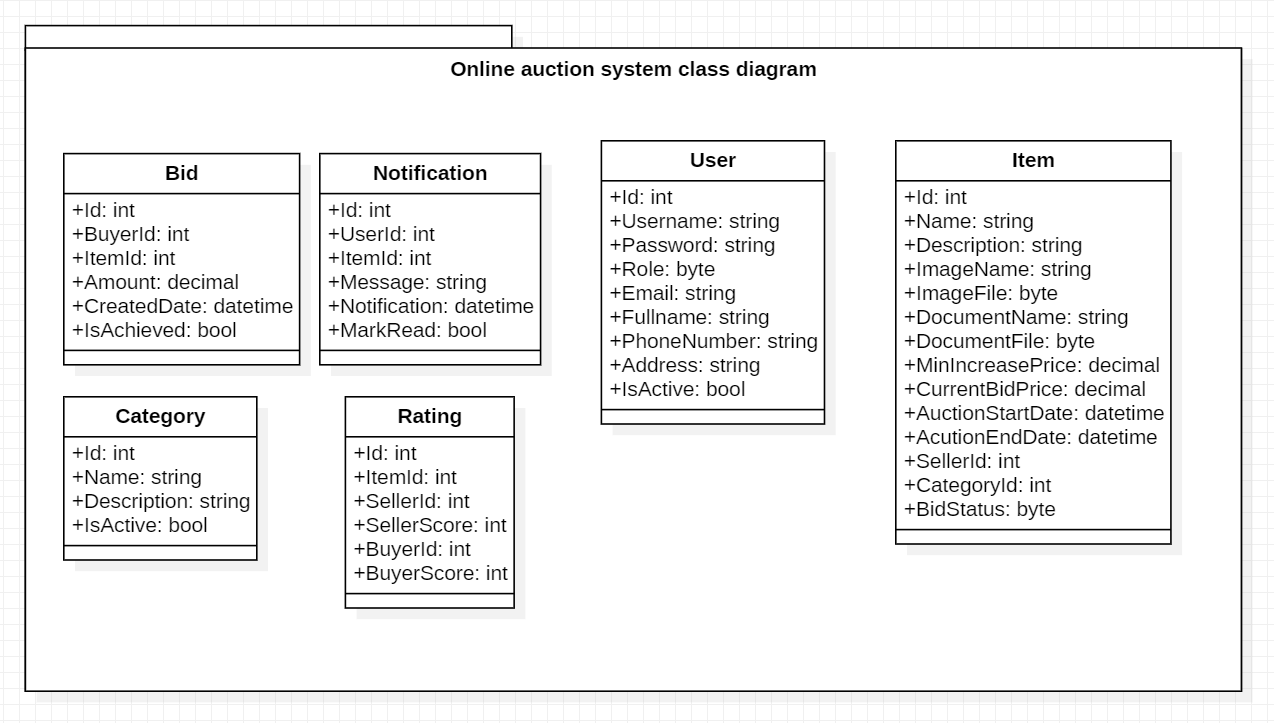


Figure 59: Class diagram of the online auction system

## 5. Database design

### 5.1 Entity Relationship Diagram

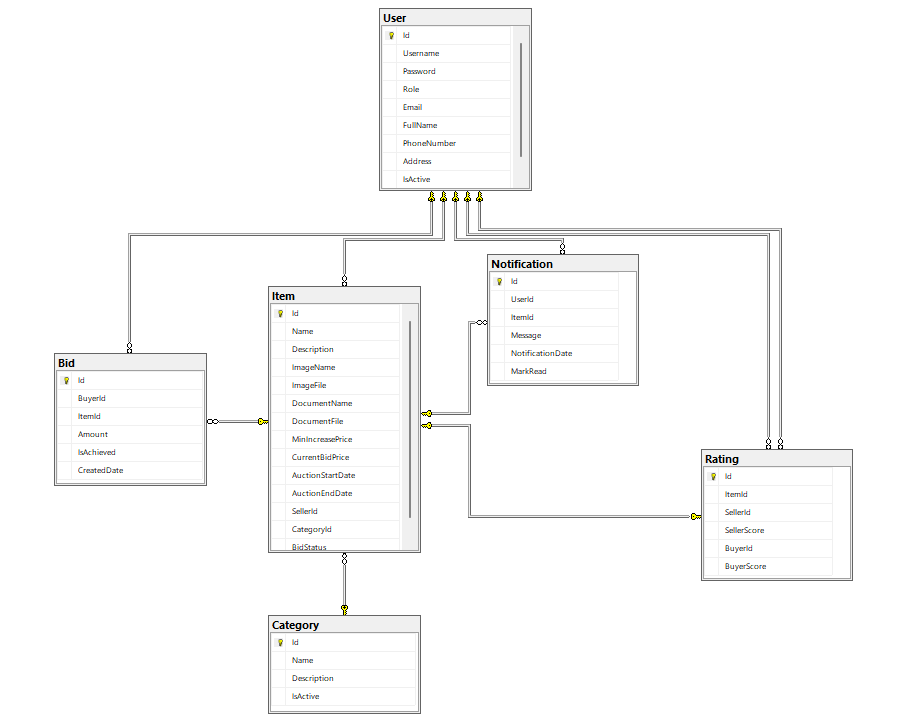


Figure 60: Entity relationship diagram of the online auction system

### 5.2 Specification of database tables and fields

### 5.2.1 User table

- Used to store user information

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Allow nulls | Note |
| Id (Primary Key) | Int | No | Unique identifier for each user |
| Username | Nvarchar(50) | No | User login name |
| Password | byte | No | User Password |
| Role | tinyint | No | User role (e.g. manager, seller, buyer) |
| Email | Nvarchar(100) | No | User's email address |
| Full name | Nvarchar(100) | No | User's full name |
| PhoneNumber | Varchar(50) | No | User's phone number |
| Address | Nvarchar(300) | No | User's address |
| IsActive | bits | No | User activity status |

### 5.2.2 Bid Table

- Used to store information about bids

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Allow nulls | Note |
| Id (Primary Key) | Int | No | Unique identifier for each bid |
| BuyerId (Foreign Key) | Int | No | Buyer's identification |
| ItemId (Foreign key) | Int | No | Identifier of the item |
| Amount | Money | No | Bid Amount |
| IsAchieved | Bit | No | Status of the bid (Successful or not) |
| CreatedDate | DateTime | No | The date of the buyer's auction |

### 5.2.3 Item table

- Used to store information about items

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Allow null | Note |
| Id (Primary Key) | Int | No | Unique identifier for each item |
| Name | Nvarchar(200) | No | Name of the item |
| description | Next | No | Description of the item |
| ImageName | Nvarchar(200) | No | The name of the image associated with the item |
| ImageFile | Varbinary(MAX) | No | Image data related to the item |
| DocumentName | Nvarchar(200) | Yes | The name of the document associated with the item |
| DocumentFile | Varbinary(MAX) | Yes | Document data related to the item |
| MinIncreasePrice | Money | No | The minimum bid for an item |
| CurrentBidPrice | Money | No | The current bid for the item |
| AuctionStartDate | DateTime | No | Auction start date |
| AuctionEndDate | DateTime | No | Auction end date |
| SellerId (Foreign Key) | Int | No | Identification of the seller |
| CategoryId (Foreign key) | Int | No | Identifier of the category to which the item belongs |
| BidStatus | tinyint | No | Status of the item auction |

### 5.2.4 Category table

- Used to store information about item categories

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Allow nulls | Note |
| Id (Primary Key) | Int | No | Unique identifier for each category |
| Name | Nvarchar(200) | No | Name of the category |
| description | Next | Yes | Category description |
| IsActive | bits | No | Category activity status |

### 5.2.5 Notification table

- Used to store information about notifications

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Allow nulls | Note |
| Id (Primary Key) | Int | No | Unique identifier for each notification |
| UserId (Foreign Key) | Int | No | Identifier of the user receiving the notification |
| ItemId (Foreign key) | Int | No | Identifier of the item associated with the notification |
| Message | Next | No | Content of notification |
| NotificationDate | DateTime | No | Date and time the notification was created |
| MarkRead | bits | No | Notification status (read or not) |

### 5.2.6 Rating table

- Used to store assessment information

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Allow nulls | Note |
| Id (Primary Key) | int | No | Unique identifier for each review |
| ItemId (Foreign key) | int | No | Identifier of the item associated with the review |
| SellerId (Foreign Key) | int | Yes | Identification of the seller |
| SellerScore | int | Yes | Seller rating |
| BuyerId (Foreign Key) | int | Yes | Buyer's identification |
| BuyerScore | int | Yes | Buyer ratings |

## CHAPTER III: DEMO

In the final chapter, we will begin building the actual system and performing testing to ensure completeness and reliability.

Prepared data: (Delete all data input by the web, do not input manually in the database)

category 1

name: Car SUVs and Pickup Trucks

description: These are versatile vehicles suitable for families or work purposes. SUVs offer spacious interiors and can accommodate multiple passengers, while pickup trucks are used for hauling cargo or towing trailers.

seller. seller

- username: Phong

- password: 12345

- email: phong0910195@gmail.com

- full name : Nguyen Ngoc Ky Phong

- phoneNumber : 0905080808

- address : 38 Yen Bai, Da Nang, Vietnam

- role: Seller

Buyer

- username: meaning

- password: 12345

- email: Nghia@gmail.com

- full name : Nguyen Trung Nghia

- phoneNumber : 0905060606

- address: 72 Nguyen Van Linh, Ho Chi Minh, Vietnam

- role: buyer

Backup data is available above. Once done, delete the above section in this document

Create later: when the Demo date is known

Auction 1:

- name: 2019 Range Rover Sport Range Car

- description: Range Rover, a symbol of luxury and off-road capability, has been in existence since 1970. Manufactured by Land Rover, a brand of Jaguar Land Rover, Range Rover has gone through five generations and continues to continue to dominate the premium SUV market. From smooth contours to powerful performance, Range Rover is the perfect combination of style and performance.

- category: Car SUVs and Pickup Trucks

- imageFile : <Binary data>

- imageName : Ranger rover.jpg

- documentFile : <Binary data>

- document name: Online Auction Project.docx

- minIncreasePrice: 500

- currentBidPrice : 39677

- auctionStartDate: Yesterday

- auctionEndDate: yesterday => (The purpose of today will be to run and complete this Auction. Prepare previous Bid data for Buyers 3 and 4 to generate Rating & Notification after running the tool)

Auction 2: create a demo date to change the bidding status on this Auction

Auction 3: created from today to 1 month later (avoid running Auto and not successfully changing the status to Happening, you can still show the Bid function)

**Data entered during Demo:**

1) Login with an admin account to create a category

name: Used Cars

description: These are pre-owned vehicles. They may have minor scratches or small defects but are still in good working condition. The value of used cars is generally lower than that of new cars.

2) Create a seller account

- username: Quoc

- password: 12345

- email: quoc@gmail.com

- full name: Nguyen Van Quoc

- phoneNumber : 09050707070

- address: 102 Tran Phu, Ho Chi Minh, Vietnam

- role: Seller

3) Create a Buyer account

- username: tu

- password: 12345

- email: trung@gmail.com

- fullname: To Anh Tu

- phoneNumber : 0905050505

- address: 05 Nguyen Thai Hoc, Da Nang, Vietnam

- role: buyer

4) Log in to your seller account to create an Upcoming auction 4

- name: 2023 Audi R8 GT Car

- description: The 2023 Audi R8 GT, is a limited-edition masterpiece that combines power, precision, and exclusivity. Powerful performance with the heart of the R8 GT is its 602-horsepower V-10 engine. This powerhouse roars to life, delivering spine-tingling acceleration and a symphony of mechanical music. The rear-wheel-drive configuration adds an element of excitement, allowing the driver to feel every nuance of the road. Wide-open-throttle upshifts at the 8700-rpm redline send shivers down your spine, reverberating through the carbon-fibre and aluminium monocoque structure.

- category: Used Cars

- imageFile : <Binary data>

- imageName : Audi A8.jpg

- documentFile : <Binary data>

- document name: Online Auction Project.docx

- minIncreasePrice: 2000

- auctionStartDate: Tomorrow

-End auction date: today + 2 days

5) Check the newly created Auction on the Home page

6) Edit Upcoming Auction:

- minIncreasePrice: 1000

-End auction date: today + 4 days

7) Azure Functions

Change status of Auction 1 upcoming => happening

Check the auction after running the Azure Function

Login seller room => home

"Auction 1" Changes to Ended

"Auction 2" Changes to Happening

"Auction 3" Does not change the Happening status

Editing data is not allowed in the above cases

"Auction 4" Does not change Upcoming status

Allows editing data

8) log in to your national seller account

Cannot edit any "Auction".

Unable to bid

9) Place Amount

Log in to your buyer Nghia account on the home page

Bidding can only be placed on "Auction 2" with the happening status

Place bid for Auction 2: 2000 => Save

10) My Bid & Bid History: buyer means "Auction 2"

11) Search:

Search the product name in the search box:

Enter "2023" => display auction "2023 Audi R8 GT Car"

Enter "2019" => display auction "2019 Range Rover Sport Range Car"

Enter "Car" => display both auctions

Leave blank => return to the home page

12) Category Browser

13) My Profile

address: “Le Duan”

password: delete the value in this field and leave the field blank

Logout then log back in with the old password and new address

14) Notification & Rating

Login to buyer 3.4 sellers 3.4 to check notifications then set the rating

15) Report Items

16) Report Rating

17) Forgot Password

Email: phong0910195@gmail.com

Password: Phong

18) Category Management

Create new Category => Check header

name: SmartPhone

description: All smartphones

Edit Category above => Check header

name: Car SUVs and Pickup Trucks => Vehicle SUVs and Pickup Trucks

Disable Category above => Check header

19) User Management

Lock account seller country => account does not exist

# Part 3: CONCLUSION

1. Conclude

The topic "Building an online auction website" also comes from today's reality to create an initial foundation to be able to further support those who want to design a website for a company or a personal website.

Although there have been many efforts to understand the knowledge learned, combined with searching specialized documents, due to limitations in time, ability and experience, certain shortcomings cannot be avoided and should be addressed. completed at the following levels:

* Learn Website programming using ASP.NET Core and Entity Framework with ReactJS, grasp knowledge of web programming, databases, and project management
* Applying to build experimental applications for today's online auction website
* This system not only provides basic functions such as registration, login, and auction, but also includes advanced functions such as evaluation, notification, and an automated system.

1. Development

- Based on experience from this project, we have some recommendations for similar projects in the future:

* **Performance optimization:** Optimizing system performance is important, especially when dealing with large amounts of data and requests from users.
* **Security:** Security is an important factor in any system. We recommend continuing to improve security measures, such as data encryption, user authentication and authorization.
* **User interface:** The user interface needs to be designed to be user-friendly and easy to use to attract and retain users.
* **Test:** Thorough testing of the system is important to ensure that all functions work as expected and without errors.
* Here are some recommendations for more complex functions that we might consider adding to our project in the future:
* **Live Auction System:** Instead of just allowing pre-bidding, you might consider adding live auction functionality where users can participate and bid in real time.
* **More Complex Rating System:** You might consider expanding your current review system to include more review criteria, such as product quality, delivery speed, and customer service.
* **Advanced Notification System:** You can enhance the current notification system to send notifications via email or text message, in addition to in-app notifications.
* **Advanced Search Features:** You might consider adding advanced search options, like searching by category, searching by price range, or searching by location.
* **Advanced Security Features:** You might consider adding advanced security features, like two-factor authentication, data encryption, or role-based access.

# Part 4: REFERENCES

<https://react.dev/>

<https://dotnet.microsoft.com/en-us/apps/aspnet/apis>

<https://www.microsoft.com/en-us/sql-server/sql-server-2022>

<https://learn.microsoft.com/en-us/dotnet/core/extensions/dependency-injection>

<https://learn.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/infrastructure-persistence-layer-design>

<https://atomicdesign.bradfrost.com/>

<https://formik.org/>

<https://www.npmjs.com/package/yup>

<https://react-bootstrap.netlify.app/>

<https://learn.microsoft.com/en-us/azure/azure-functions/>