**A blue and white logo

Description automatically generatedCOMSATS University Islamabad,**

**Abbottabad, Pakistan**

**FurnishARt (An AR-Based Furniture Store)**

(Version 1.0)

***By***

**Muhammad Umer CIIT/SP21-BCS-016/ATD**

**Muhammad Noman CIIT/SP21-BCS-014/ATD**

**Hafiz Talha Nazir CIIT/SP21-BCS-007/ATD**

***Supervisor***

**Ma’am Bushra Mushtaq**

***Bachelor of Science in Computer Science (2021-2025)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

**A blue and white logo

Description automatically generated COMSATS University Islamabad, Pakistan**

**FurnishARt (An AR-Based Furniture Store)**

**A project presented to**

**COMSATS University Islamabad, Abbottabad**

**In partial fulfillment**

**of the requirement for the degree of**

***Bachelor of Science in Computer Science (2021-2025)***

***By***

**Muhammad Umer CIIT/SP21-BCS-016/ATD**

**Muhammad Noman CIIT/SP21-BCS-014/ATD**

**Hafiz Talha Nazir CIIT/SP21-BCS-007/ATD**

**DECLARATION**

We hereby declare that no part of or all this software has been taken from any source. It is also stated that we created this software and associated report fully through our own efforts. If any component of this project turns out to have been copied from another source or to be an exact copy of others. We will not back down from the consequences. No part of the work presented has been submitted in support of any other degree or accreditation available at this or any other university or institute of research.

Muhammad Umer Muhammad Noman Hafiz Talha Nazir

------------------------ ------------------------ ----------------------

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (CS (Computer Science)) “FurnishARt (An AR-Based Furniture Store)” was Developed by **Muhammad Umer (CIIT/SP21-BCS-016), Muhammad Noman (****CIIT/SP21-BCS-014)** and **Hafiz Talha Nazir (CIIT/SP21-BCS-007)** under the supervision of “**Ma’am Bushra Mushtaq**” and that in their opinion; it is fully sufficient, in the scope and quality for the degree of Bachelors of Science in Computer Science.

---------------------------------------

**Ma’am Bushra Mushtaq**

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Dr. Kashif Bilal**

**Head of Department**

**(Department of Computer Science)**

**EXECUTIVE SUMMARY**

The shift toward digital shopping has created demand for innovative solutions that seamlessly blend online and offline experiences, particularly in the furniture sector, where customers benefit significantly from visualizing products in their real-world spaces before making a purchase. Traditional e-commerce platforms often fall short in providing this immersive experience, leading to uncertainty in purchasing decisions and increased return rates. To address these challenges, furnishARt introduces an Augmented Reality (AR)-based platform that enables customers to view and interact with 3D furniture models both on the web and through an AR-capable mobile application.

furnishARt provides users with the ability to preview interactive 3D models of furniture on a web application. For an immersive AR experience, users can select the "AR preview" option on the web, which will redirect them to the mobile app, allowing them to place and view models in their own environment through their device’s camera. This integrated flow offers customers a clear sense of furniture dimensions, style, and fit within their space, helping them make informed purchasing decisions without needing an in-store visit.

The furnishARt system features modules such as 3D model management, AR preview capabilities, user registration and authentication, furniture catalog management, customer reviews, and secure payment processing. An admin panel allows for efficient management of furniture listings and reviews, while users benefit from a secure and user-friendly shopping experience. The project is being developed using an Agile methodology to support iterative progress and rapid response to feedback, and object-oriented principles for modularity and maintainability.

With furnishARt, furniture retailers can offer a unique and engaging shopping experience, giving them a competitive edge while enabling customers to make confident, personalized choices directly from their own homes.

**ACKNOWLEDGEMENT**

All praise belongs to Almighty Allah, who enabled us to complete this difficult activity by giving us a tiny fraction of His limitless wisdom.

We are grateful to our project supervisor, "**Ma’am Bushra Mushtaq**". The accomplishment of this project would have been uncertain without their personal supervision, advice, and important guidance. We are grateful to them for their support and encouragement throughout this process.

We are also grateful to our parents and loved ones for their constant encouragement and for teaching us the values of honesty and respect.

Muhammad Umer Muhammad Noman Hafiz Talha Nazir

------------------------ ------------------------ ----------------------

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **AR** | Augmented Reality |
|  |  |

Table of Contents

[1 Introduction 11](#_Toc185906579)

[1.1 Brief 12](#_Toc185906580)

[1.2 Relevance to Course Module 12](#_Toc185906581)

[1.3 Project Background 13](#_Toc185906582)

[1.4 Literature Review 13](#_Toc185906583)

[1.5 Analysis from Literature Review 14](#_Toc185906584)

[1.6 Methodology and Software Life Cycle 15](#_Toc185906585)

[1.6.1 Rational behind Selected Methodology 15](#_Toc185906586)

[2 Problem Definition 15](#_Toc185906587)

[2.1 Problem Statement 15](#_Toc185906588)

[2.2 Deliverables and Development Requirements 16](#_Toc185906589)

[2.2.1 Deliverables: 16](#_Toc185906590)

[2.2.2 Development Requirements: 16](#_Toc185906591)

[3 Requirement Analysis 18](#_Toc185906592)

[3.1 Use Cases Diagram(s) 18](#_Toc185906593)

[3.2 Detailed Use Case 19](#_Toc185906594)

[3.2.1 Use Case: Register 19](#_Toc185906595)

[3.2.2 Use Case: Login 19](#_Toc185906596)

[3.2.3 Use Case: Forget Details 20](#_Toc185906597)

[3.2.4 Use Case: View Furniture 20](#_Toc185906598)

[3.2.5 Use Case: View AR Preview of Furniture 20](#_Toc185906599)

[3.2.6 Use Case: View Similar Furniture 21](#_Toc185906600)

[3.2.7 Use Case: Augment Multiple Furniture Items from List 21](#_Toc185906601)

[3.2.8 Use Case: View Customer Reviews 22](#_Toc185906602)

[3.2.9 Use Case: View 3D Model 22](#_Toc185906603)

[3.2.10 Use Case: Search By Filters 23](#_Toc185906604)

[3.2.11 Use Case: View Listing Details 23](#_Toc185906605)

[3.2.12 Use Case: Write Review 24](#_Toc185906606)

[3.2.13 Use Case: Add to Cart 24](#_Toc185906607)

[3.2.14 Use Case: Proceed to Checkout 25](#_Toc185906608)

[3.2.15 Use Case: View Cart 25](#_Toc185906609)

[3.2.16 Use Case: Update From Cart 26](#_Toc185906610)

[3.2.17 Use Case: Delete From Cart 26](#_Toc185906611)

[3.2.18 Use Case: View Orders 26](#_Toc185906612)

[3.2.19 Use Case: View Order Status 27](#_Toc185906613)

[3.2.20 Use Case: View Orders History 27](#_Toc185906614)

[3.2.21 Use Case: Enter Payment Information 28](#_Toc185906615)

[3.2.22 Use Case: Select Shipping Method 28](#_Toc185906616)

[3.2.23 Use Case: Place Order 29](#_Toc185906617)

[3.2.24 Use Case: Add Furniture Listings 29](#_Toc185906618)

[3.2.25 Use Case: Update Furniture Listings 30](#_Toc185906619)

[3.2.26 Use Case: Delete Furniture Listings 30](#_Toc185906620)

[3.2.27 Use Case: Add 3D Models 31](#_Toc185906621)

[3.2.28 Use Case: Update 3D Models 31](#_Toc185906622)

[3.2.29 Use Case: Delete 3D Models 32](#_Toc185906623)

[3.2.30 Use Case: Generate Inventory Reports 32](#_Toc185906624)

[3.2.31 Use Case: Generate Inventory Reports 33](#_Toc185906625)

[3.2.32 Use Case: Update Order Status 33](#_Toc185906626)

[3.2.33 Use Case: View Furniture Listings on Admin Panel 34](#_Toc185906627)

[3.2.34 Use Case: Shows Sales & Stock Dashboard 35](#_Toc185906628)

[3.2.35 Use Case: Secures Sales and Updates Stocks 35](#_Toc185906629)

[3.2.36 Use Case: Logout 35](#_Toc185906630)

[3.3 Functional Requirements 36](#_Toc185906631)

[3.3.1 Functional Requirement for UC-1: Register 36](#_Toc185906632)

[3.3.2 Functional Requirement for UC-2: Login 36](#_Toc185906633)

[3.3.3 Functional Requirement for UC-3: Forgot Details 37](#_Toc185906634)

[3.3.4 Functional Requirement for UC-4: View Furniture 37](#_Toc185906635)

[3.3.5 Functional Requirement for UC-5: View AR Preview of Furniture 37](#_Toc185906636)

[3.3.6 Functional Requirement for UC-6: View Similar Furniture 37](#_Toc185906637)

[3.3.7 Functional Requirement for UC-7: View Customer Reviews 37](#_Toc185906638)

[3.3.8 Functional Requirement for UC-8: View 3D Model 38](#_Toc185906639)

[3.3.9 Functional Requirement for UC-9: Search by Filters 38](#_Toc185906640)

[3.3.10 Functional Requirement for UC-10: View Listing Details 38](#_Toc185906641)

[3.3.11 Functional Requirement for UC-11: Augment Multiple Furniture Items from List 38](#_Toc185906642)

[3.3.12 Functional Requirement for UC-12: Write Review 39](#_Toc185906643)

[3.3.13 Functional Requirement for UC-13: Add to Cart 39](#_Toc185906644)

[3.3.14 Functional Requirement for UC-14: View Cart 39](#_Toc185906645)

[3.3.15 Functional Requirement for UC-15: Update from Cart 39](#_Toc185906646)

[3.3.16 Functional Requirement for UC-16: Proceed to Checkout 40](#_Toc185906647)

[3.3.17 Functional Requirement for UC-17: Enter Payment Information 40](#_Toc185906648)

[3.3.18 Functional Requirement for UC-18: Select Shipping Method 40](#_Toc185906649)

[3.3.19 Functional Requirement for UC-19: Place Order 40](#_Toc185906650)

[3.3.20 Functional Requirement for UC-20: View Orders 41](#_Toc185906651)

[3.3.21 Functional Requirement for UC-21: View Order Status 41](#_Toc185906652)

[3.3.22 Functional Requirement for UC-22: View Orders History 41](#_Toc185906653)

[3.3.23 Functional Requirement for UC-23: Delete from Cart 41](#_Toc185906654)

[3.3.24 Functional Requirement for UC-24: Generate Inventory Report 42](#_Toc185906655)

[3.3.25 Functional Requirement for UC-25: Secure Sales and Update Stocks 42](#_Toc185906656)

[3.3.26 Functional Requirement for UC-26: Show Sales and Stock Dashboard 42](#_Toc185906657)

[3.3.27 Functional Requirement for UC-27: Add Furniture Listings 42](#_Toc185906658)

[3.3.28 Functional Requirement for UC-28: Update Furniture Listings 43](#_Toc185906659)

[3.3.29 Functional Requirement for UC-29: Delete Furniture Listings 43](#_Toc185906660)

[3.3.30 Functional Requirement for UC-30: Add 3D Models 43](#_Toc185906661)

[3.3.31 Functional Requirement for UC-31: Update 3D Models 43](#_Toc185906662)

[3.3.32 Functional Requirement for UC-32: Delete 3D Models 44](#_Toc185906663)

[3.3.33 Functional Requirement for UC-33: Update Order Status 44](#_Toc185906664)

[3.3.34 Functional Requirement for UC-34: View Furniture Listings on Admin Panel 44](#_Toc185906665)

[3.3.35 Functional Requirement for UC-35: Logout 44](#_Toc185906666)

[3.4 Non-Functional Requirements 45](#_Toc185906667)

[3.4.1 Usability 45](#_Toc185906668)

[3.4.2 Performance 45](#_Toc185906669)

[4 Design and Architecture 46](#_Toc185906670)

[4.1 System Architecture 47](#_Toc185906671)

[4.2 Data Representation [Class Diagram] 49](#_Toc185906672)

[4.2.1 Website Application Class Diagram 49](#_Toc185906673)

[4.2.2 AR App Class Diagram: 50](#_Toc185906674)

[4.3 Process Flow/Representation 50](#_Toc185906675)

[4.3.1 User Activity Diagram: 51](#_Toc185906676)

[4.3.2 Admin Activity Diagram: 51](#_Toc185906677)

[4.3.3 ERD diagram 53](#_Toc185906678)

[4.4 Design Model 55](#_Toc185906679)

[4.4.1 Sequence Diagram 55](#_Toc185906680)

[5 Implementation 57](#_Toc185906681)

[5.1 Algorithm 57](#_Toc185906682)

[5.1.1 User Authentication (Sign Up and Login) 57](#_Toc185906683)

[5.1.2 Cart Management (Add, Get, Delete Cart Items) 57](#_Toc185906684)

[5.1.3 Creating Payment Intent 59](#_Toc185906685)

[5.1.4 Product Placement: 59](#_Toc185906686)

[5.2 External APIs 60](#_Toc185906687)

[5.3 User Interface 61](#_Toc185906688)

[6 Testing and Evaluation 68](#_Toc185906689)

[6.1 Manual Testing 68](#_Toc185906690)

[6.1.1 System Testing 68](#_Toc185906691)

[6.1.2 Unit Testing 68](#_Toc185906692)

[6.1.3 Functional Testing 70](#_Toc185906693)

[6.1.4 Integration Testing 71](#_Toc185906694)

[6.2 Automated Testing 72](#_Toc185906695)

[6.2.1 Tools Used 72](#_Toc185906696)

[6.2.2 Automated Test Scenarios 72](#_Toc185906697)

[7 Conclusion and Future Work 73](#_Toc185906698)

[7.1 Conclusion 73](#_Toc185906699)

[7.2 Future Work 73](#_Toc185906700)

[8 References 74](#_Toc185906701)

# Introduction

With the rapid advancement of augmented reality (AR) technology, the retail industry, especially e-commerce, has gained a powerful tool to elevate the online shopping experience. For furniture retail, AR brings a unique solution to a longstanding challenge—how to bridge the gap between online shopping and the in-person experience of seeing furniture in context.

furnishARt is designed to address this challenge by providing customers with an immersive AR shopping experience. Through the furnishARt web application, users can view interactive 3D models of furniture, exploring details and design from various angles. When ready for an AR preview, users can select the option to be redirected to the furnishARt mobile app, where they can view these models in their real environment, adjusting placement and angles to ensure the perfect fit.

This approach allows customers to make well-informed purchasing decisions, blending online convenience with a tactile sense of the product's presence in their homes. By incorporating an intuitive AR flow and easy-to-navigate 3D previews, furnishARt aims to revolutionize online furniture shopping, giving retailers a competitive edge and creating a more satisfying, user-centered shopping experience.

## Brief

furnishARt is an innovative AR-based online furniture store that allows customers to visualize furniture pieces in their physical space. Through an interactive web application and an AR-enabled mobile app, furnishARt empowers users to view 3D furniture models online and seamlessly transition to augmented reality for a true-to-life preview. By combining the convenience of online shopping with the immersive potential of AR, furnishARt enhances customer confidence in purchasing furniture online, reducing uncertainties and improving satisfaction.

## Relevance to Course Module

This system is built upon the foundation set up by many of the course modules we studied throughout the four-year program of BSCS. The courses modules that served as the building blocks for the project are:

**Software Engineering Concept**: We kept in mind the concepts we learned in software engineering Concept about how the project lifecycle progresses, which process model is used for what kind of project etc.

**Human Computer Interaction:** We kept in mind the major principles of HCI so that the interface is more user friendly and understandable by the end user.

**Software Project Management:** in this course we studied how to manage time, resources effectively and efficiently. How to make schedule for your project. Planning is also important we studied how to plan the development process and make the best use of available resources. Through this course we learn how to use the available tools for the planning like making Gantt chart on project plan.

**Web Technologies:** This project integrates essential web development principles by leveraging React.js, Next.js, and Three.js for creating interactive and responsive user interfaces that display 3D models online. RESTful APIs facilitate seamless data flow between the frontend and backend, aligning with the Web Technologies course focus on building dynamic, user-centric applications.

**Database Systems I:** With SQL database integration, this project applies core concepts from Database Systems I, including relational database design, data integrity, and transaction security. SQL stores and manages essential data such as user profiles, furniture details, and transactions, emphasizing efficient data retrieval and reliable storage. This supports high-volume operations typical in e-commerce platforms, directly tying into course topics on data management and security.

## Project Background

As online shopping grows, the need for enhanced customer experience is critical, especially in sectors like furniture retail where visualizing products in physical spaces plays a significant role in purchase decisions. Traditional e-commerce methods often fall short, offering only static images and leaving customers unsure of product fit and style within their own homes.

To address these limitations, augmented reality (AR) has emerged as a solution, allowing customers to view virtual furniture models in real-world environments. Leveraging AR, this project combines interactive 3D model viewing on a web platform with the option to preview furniture in AR via a mobile app, providing customers a highly immersive and realistic shopping experience. By enhancing the online shopping process with AR technology, this approach aims to bridge the gap between online and in-person shopping, ultimately leading to more informed purchasing decisions and customer satisfaction.

## Literature Review

**Table 1. 1: Literature Review**

|  |  |  |
| --- | --- | --- |
| **Application Name** | **Weakness** | **Proposed Project Solution** |
| VSurface | It lacks AR previews for furniture items, restricting its functionality to rugs only. | This system provides AR previews for a broader range of furniture items. |
| IKEA Place | The app's interface is complex or difficult to the end user – and IKEA is not available in Pakistan. | Our system will provide user friendly interface and this system will be launching in Pakistan. |
| Wayfair Spaces | Their Website is overwhelming and cluttered at times, with vast variety of products, categories, and subcategories to choose from. The website can be slow and laggy at times and Wayfair is not available in Pakistan. | We propose simplifying product categorization, optimizing the website for faster loading times, and refining subcategory structures. |

## Analysis from Literature Review

The literature review highlights the limitations and gaps in existing AR-based furniture and decor applications, particularly in usability, accessibility, and product range:

1. **Limited AR Functionality**

Applications like VSurface offer AR previews only for specific products, such as rugs, limiting their overall utility. Our platform will address this by providing a full range of AR-enabled furniture items, expanding the shopping experience.

1. **User Interface Complexity**

The IKEA Place app has a complex interface that may overwhelm users, and the service is not available in Pakistan. By focusing on a user-friendly design and accessibility in Pakistan, our system will offer a streamlined experience for a broader audience.

1. **Website Performance and Navigation**

Wayfair Spaces, while comprehensive, faces challenges with cluttered categories, subcategories, and loading speeds. We aim to overcome these issues by optimizing the website for faster performance, simplifying product categories, and ensuring efficient navigation for a smoother user experience.

## Methodology and Software Life Cycle

For the development of the project, an Agile software development methodology has been selected due to its emphasis on iterative progress and flexibility. Agile allows teams to adapt to changes quickly, making it particularly suitable for projects involving emerging technologies like augmented reality (AR). The Agile approach promotes collaboration among cross-functional teams and enables continuous feedback from stakeholders, ensuring that the final product aligns with user expectations and business goals.

The software life cycle in this project will consist of several iterative cycles, including planning, designing, developing, testing, and deploying the application. Each cycle will focus on delivering incremental improvements and features, allowing for regular updates and adjustments based on user feedback and testing results. This iterative nature supports rapid prototyping and encourages experimentation, which is vital for integrating AR functionality effectively into the online furniture store.

### Rational behind Selected Methodology

The Agile methodology was chosen for several reasons:

1. **Flexibility:** Agile allows for adjustments throughout the development process, accommodating changes in user requirements or market conditions. This adaptability is crucial in the fast-evolving field of AR technology.
2. **Customer-Centric Focus:** By prioritizing customer feedback and involving users in the development process, Agile ensures that the final product delivers maximum value and meets the actual needs of users. This focus on customer satisfaction is vital in the competitive e-commerce landscape.
3. **Iterative Development:** The Agile framework promotes incremental progress through sprints, enabling the team to release functional components of the application regularly. This approach allows for early detection of issues and continuous improvement based on real user interactions.
4. **Enhanced Collaboration:** Agile encourages collaboration among developers, designers, and stakeholders, fostering a culture of communication and shared ownership of the project. This collaborative environment enhances creativity and innovation, essential for integrating AR features.

# Problem Definition

## Problem Statement

Traditional online furniture shopping presents several challenges for customers, including:

**1. Visualization and Confidence:**

* Difficulty visualizing how furniture will look and fit in their living spaces.
* Lack of confidence in purchasing decisions due to uncertainty.

**2. Increased Returns:**

* Surprises upon delivery due to the lack of context in online product displays.
* Higher return rates leading to increased costs for businesses and inconvenience for customers.

**3. Engagement Limitations:**

* Limited engagement with static online product displays.
* Inability to replicate the immersive experience of physical showroom exploration.

By embarking on this project, we aim to transform the online shopping experience, boost customer satisfaction, and gain a competitive edge in e-commerce. Despite existing systems, re-implementing one offers hands-on learning in AR development, e-commerce integration, and UI design, essential for skill acquisition and innovation.

## Deliverables and Development Requirements

### Deliverables:

* **3D Furniture Models:** High-quality, accurate 3D models of all furniture items in the catalog.
* **AR Functionality:** Full AR integration to allow users to view furniture in real-world environments.
* **Product Catalog Management System:** A backend system for managing product listings, including 3D models, descriptions, and pricing.
* **User Interface (UI):** Intuitive, user-friendly interface for seamless navigation and interaction.
* **Secure Payment System:** Integration of a secure payment gateway with multiple payment options.
* **Admin Panel:** Dashboard for administrators to manage products, 3D models, and user data.

### Development Requirements:

* **Device Compatibility:** Ensure compatibility with common smartphones and tablets for AR functionality.
* **Database Management:** Efficient data handling for product catalog, user information, and order processing.
* **Security Protocols:** SSL encryption, secure login, and data protection measures.
* **Testing:** Comprehensive testing for device compatibility, AR performance, and payment processing reliability.

Table 2.1: Development Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Tools and Technologies | Technology | Version | Rationale |
| JavaScript | ES6 | Programming language |
| React.js /Next.js | 18 | Frontend Web Development |
| Node.js / Express | 20 / 4.1 | Backend Web Development |
| Report/Presentation | Microsoft365 | Documentation |
| PostgreSQL | 16.2 | DBMS |
| Visual Studio Code | 1.87 | Code Editor |
| Three.js/AR.js/Unity/Blender | Latest | 3D modeling and AR Preview |
| SQL | 2013 | Query Language |

# Requirement Analysis

## Use Cases Diagram(s)

## Detailed Use Case

### Use Case: Register

|  |  |
| --- | --- |
| **Use Case ID:** | UC-001 |
| **Use Case Name:** | Register |
| **Actors:** | Visitor |
| **Description:** | Allows a visitor to create a new account by providing necessary details. |
| **Trigger:** | Visitor chooses to sign up for a new account. |
| **Preconditions:** | The visitor is not logged in. |
| **Postconditions:** | A new user account is created, and the visitor is logged in. |
| **Normal Flow:** | 1. Visitor navigates to the registration page. 2. Visitor enters required details (e.g., name, email, password). 3. Visitor submits the registration form. 4. System validates the details. 5. System creates a new user account. 6. System logs the user in and redirects to the home page. |
| **Alternative Flows:** | If the email is already in use, the system prompts the visitor to use a different email or recover the existing account. |
| **Exceptions:** | 1. Network failure during registration.   Validation errors (e.g., invalid email format, weak password). |
| **Business Rules** | Password must meet security criteria (e.g., minimum length, contains letters and numbers). |
| **Assumptions:** | Users will have access to a valid email address for registration. |

### Use Case: Login

|  |  |
| --- | --- |
| **Use Case ID:** | UC-002 |
| **Use Case Name:** | Login |
| **Actors:** | Registered User |
| **Description:** | Allows a registered user to access their account by entering valid credentials. |
| **Trigger:** | User chooses to log in. |
| **Preconditions:** | User has a registered account. |
| **Postconditions:** | User is logged into their account and redirected to the home page. |
| **Normal Flow:** | 1. User navigates to the login page. 2. User enters email and password. 3. User submits the login form. 4. System validates the credentials. 5. System logs the user in and redirects to the home page. |
| **Alternative Flows:** | If credentials are incorrect, the system prompts the user to retry. |
| **Exceptions:** | Account locked due to multiple failed login attempts. |
| **Business Rules** | Password must meet security criteria (e.g., minimum length, contains letters and numbers). |
| **Assumptions:** | Users will have access to a valid email address for registration. |

### Use Case: Forget Details

|  |  |
| --- | --- |
| **Use Case ID:** | UC-003 |
| **Use Case Name:** | Forget Details |
| **Actors:** | Registered User |
| **Description:** | Enables users to recover or reset their forgotten login details. |
| **Trigger:** | User chooses the "Forget Details" option. |
| **Preconditions:** | User is unable to remember their login credentials. |
| **Postconditions:** | User receives instructions to recover or reset their details. |
| **Normal Flow:** | 1. User clicks on the "Forget Details" link. 2. User enters their registered email. 3. System sends a recovery email with instructions. 4. User follows the instructions to reset their password. |
| **Alternative Flows:** | If the email is not registered, the system prompts the user to check their email or register a new account. |
| **Exceptions:** | Email delivery failure. |
| **Business Rules** | Recovery email must contain a secure link with a limited validity period. |
| **Assumptions:** | User has access to their registered email. |

### Use Case: View Furniture

|  |  |
| --- | --- |
| **Use Case ID:** | UC-004 |
| **Use Case Name:** | View Furniture |
| **Actors:** | Visitor, Registered User |
| **Description:** | Allows users to browse and view different furniture items available in the store. |
| **Trigger:** | User navigates to the furniture listing page. |
| **Preconditions:** | System has furniture items listed in the database. |
| **Postconditions:** | User can see a list of furniture items with basic details. |
| **Normal Flow:** | 1. User navigates to the furniture page. 2. System displays a list of furniture items with images, names, and prices. 3. User selects an item to view detailed information. |
| **Alternative Flows:** | If no items are available, the system displays an appropriate message. |
| **Exceptions:** | Database connection failure. |
| **Business Rules** | Items must be displayed in a user-friendly manner, sorted by relevance or preference. |
| **Assumptions:** | Users will explore various furniture items before making a purchase decision. |

### Use Case: View AR Preview of Furniture

|  |  |
| --- | --- |
| **Use Case ID:** | UC-005 |
| **Use Case Name:** | View AR Preview of Furniture |
| **Actors:** | Registered User, Visitor |
| **Description:** | Enables users to visualize how the furniture would look in their physical space using AR technology. |
| **Trigger:** | User selects the AR preview option for a furniture item. |
| **Preconditions:** | User has a compatible mobile device with a camera. |
| **Postconditions:** | User can see a virtual representation of the furniture item in their physical environment. |
| **Normal Flow:** | 1. User navigates to a furniture item's details page. 2. User selects the AR preview option. 3. System accesses the mobile device's camera. 4. System overlays the 3D model of the furniture item onto the live camera feed. |
| **Alternative Flows:** | If the device is not compatible, the system displays an appropriate message. |
| **Exceptions:** | AR functionality fails due to software or hardware issues. |
| **Business Rules** | AR preview must be accurate and responsive to user movements. |
| **Assumptions:** | Users will use the AR feature to better visualize the furniture in their space. |

### Use Case: View Similar Furniture

|  |  |
| --- | --- |
| **Use Case ID:** | UC-006 |
| **Use Case Name:** | View Similar Furniture |
| **Actors:** | Visitor, Registered User |
| **Description:** | Shows products of same category like the one user is currently viewing. |
| **Trigger:** | User is viewing a specific furniture item. |
| **Preconditions:** | System has related products in the database. |
| **Postconditions:** | User can see a list of similar categories of furniture items. |
| **Normal Flow:** | 1. User views the details of a furniture item. 2. System displays a list of similar items below the main product details. |
| **Alternative Flows:** | If no similar furniture items are found, the system displays a relevant message or alternative suggestions. |
| **Exceptions:** | Database query failure. |
| **Business Rules** | Similar furniture items should be based on category, style, and price range. |
| **Assumptions:** | Users will be interested in exploring similar products for comparison. |

### Use Case: Augment Multiple Furniture Items from List

|  |  |
| --- | --- |
| **Use Case ID:** | UC-007 |
| **Use Case Name:** | Augment Multiple Furniture Items from List |
| **Actors:** | Registered User, Visitor |
| **Description:** | Enables users to visualize multiple furniture items simultaneously within their physical space. |
| **Trigger:** | User selects multiple furniture items to view in AR mode. |
| **Preconditions:** | User has initiated AR preview mode and has a compatible mobile device with a camera. |
| **Postconditions:** | User can see multiple virtual furniture items in their physical environment. |
| **Normal Flow:** | 1. User navigates to the furniture listing. 2. User selects multiple items for AR preview. 3. System accesses the mobile device's camera. 4. System shows similar category furniture items list in AR preview mode. 5. System overlays the 3D models of the selected furniture items onto the live camera feed. |
| **Alternative Flows:** | If the device is not compatible, the system displays an appropriate message. |
| **Exceptions:** | AR functionality fails due to software or hardware issues. |
| **Business Rules** | Multiple items must be displayed accurately and maintain spatial relationships. |
| **Assumptions:** | Users will use this feature to visualize how multiple pieces of furniture work together in their space. |

### Use Case: View Customer Reviews

|  |  |
| --- | --- |
| **Use Case ID:** | UC-008 |
| **Use Case Name:** | View Customer Reviews |
| **Actors:** | Visitor, Registered User, Admin |
| **Description:** | Displays reviews and ratings left by other customers for a particular furniture item. |
| **Trigger:** | User is viewing a furniture item’s details page. |
| **Preconditions:** | System has reviews stored for the furniture item. |
| **Postconditions:** | User can read and assess customer reviews. |
| **Normal Flow:** | 1. User navigates to a furniture item's details page. 2. System displays a list of reviews and ratings for the item. |
| **Alternative Flows:** | If no reviews are available, the system displays a message indicating there are no reviews yet. |
| **Exceptions:** | Database query failure. |
| **Business Rules** | Reviews must be displayed in chronological order with the most recent first. |
| **Assumptions:** | Users will read reviews to inform their purchasing decisions. |

### Use Case: View 3D Model

|  |  |
| --- | --- |
| **Use Case ID:** | UC-009 |
| **Use Case Name:** | View 3D Model |
| **Actors:** | Visitor, Registered User |
| **Description:** | Allows users to interact with a 3D model of the furniture item, including rotation and zoom functions. |
| **Trigger:** | User selects the 3D view option for a furniture item. |
| **Preconditions:** | System has a 3D model available for the furniture item. |
| **Postconditions:** | User can view and interact with the 3D model. |
| **Normal Flow:** | 1. User navigates to a furniture item's details page. 2. User selects the 3D view option. 3. System loads the 3D model. 4. User interacts with the model using rotation and zoom controls. |
| **Alternative Flows:** | If the 3D model fails to load, the system displays an error message and provides the option to view standard images. |
| **Exceptions:** | System failure in loading or rendering the 3D model. |
| **Business Rules** | 3D models must be accurate and high-quality representations of the actual products. |
| **Assumptions:** | Users will use the 3D model to get a better understanding of the product. |

### Use Case: Search By Filters

|  |  |
| --- | --- |
| **Use Case ID:** | UC-010 |
| **Use Case Name:** | Search By Filters |
| **Actors:** | Visitor, Registered User, Admin |
| **Description:** | Enables users to filter and search for furniture items based on various criteria like category, price, etc. |
| **Trigger:** | User accesses the search functionality. |
| **Preconditions:** | System has a diverse range of furniture items listed. |
| **Postconditions:** | User receives a refined list of furniture items matching the selected filters. |
| **Normal Flow:** | 1. User navigates to the search or filter section. 2. User selects desired filters (e.g., category, price range). 3. User submits the filter options. 4. System displays a list of items that match the criteria. |
| **Alternative Flows:** | If no items match the filters, the system displays a message and suggests clearing filters or adjusting criteria. |
| **Exceptions:** | System failure in applying filters or displaying results. |
| **Business Rules** | Filters must be comprehensive and user-friendly. |
| **Assumptions:** | Users will utilize filters to narrow down their search and find suitable furniture items. |

### Use Case: View Listing Details

|  |  |
| --- | --- |
| **Use Case ID:** | UC-011 |
| **Use Case Name:** | View Listing Details |
| **Actors:** | Visitor, Registered User, Admin |
| **Description:** | Provides detailed information about a specific furniture item, including dimensions, materials, and other specifications. |
| **Trigger:** | User selects a furniture item from the list. |
| **Preconditions:** | System has detailed information about the furniture item. |
| **Postconditions:** | User can see detailed information about the selected furniture item. |
| **Normal Flow:** | 1. User selects a furniture item. 2. System displays detailed information about the item, including dimensions, materials, and specifications. |
| **Alternative Flows:** | If detailed information is not available, the system displays a basic overview with a note about missing details. |
| **Exceptions:** | System failure in retrieving or displaying item details. |
| **Business Rules** | Information must be accurate and comprehensive to aid the user in making an informed decision. |
| **Assumptions:** | Users will review detailed listings to understand the product better before purchasing. |

### Use Case: Write Review

|  |  |
| --- | --- |
| **Use Case ID:** | UC-012 |
| **Use Case Name:** | Write Review |
| **Actors:** | Registered User |
| **Description:** | A registered user writes a review for a purchased furniture item. |
| **Trigger:** | The user is logged in and has purchased the furniture item. |
| **Preconditions:** | The user is logged in and has purchased the furniture item. |
| **Postconditions:** | The user has submitted a review for the furniture item. |
| **Normal Flow:** | 1. The user selects the option to write a review for a purchased furniture item. 2. The application displays a review form. 3. The user enters their review and rating. 4. The user submits the review. 5. The system saves the review and displays it in the customer reviews section. |
| **Alternative Flows:** | If the user cancels the review process, the application returns to the previous screen. |
| **Exceptions:** | System failure |
| **Business Rules** | Only users who have purchased the furniture item can write a review. |
| **Assumptions:** | The user has purchased the furniture item and is logged in. |

### Use Case: Add to Cart

|  |  |
| --- | --- |
| **Use Case ID:** | UC-13 |
| **Use Case Name:** | Add to Cart |
| **Actors:** | Registered User |
| **Description:** | A registered user adds a selected furniture item to their shopping cart. |
| **Trigger:** | The user selects the option to add a furniture item to the cart. |
| **Preconditions:** | The user is logged in and viewing a furniture item. |
| **Postconditions:** | The selected furniture item is added to the user's shopping cart. |
| **Normal Flow:** | 1. The user selects the option to add a furniture item to the cart. 2. The system confirms the addition of the item to the cart. 3. The system updates the cart with the selected item. |
| **Alternative Flows:** | If the user cancels the process, the system returns to the previous screen. |
| **Exceptions:** | System failure |
| **Business Rules** | Only logged-in users can add items to their cart. |
| **Assumptions:** | The user is logged in and viewing a furniture item. |

### Use Case: Proceed to Checkout

|  |  |
| --- | --- |
| **Use Case ID:** | UC-14 |
| **Use Case Name:** | Proceed to Checkout |
| **Actors:** | Registered User |
| **Description:** | A registered user proceeds to checkout to purchase the items in their cart. |
| **Trigger:** | The user selects the option to proceed to checkout. |
| **Preconditions:** | The user is logged in and has items in their cart. |
| **Postconditions:** | The user has initiated the checkout process. |
| **Normal Flow:** | 1. The user selects the option to proceed to checkout. 2. The system displays the checkout form with shipping and payment options. 3. The user enters the required information and submits the form. |
| **Alternative Flows:** | If the user cancels the checkout process, the system returns to the cart view. |
| **Exceptions:** | System failure. |
| **Business Rules** | Only logged-in users can proceed to checkout. |
| **Assumptions:** | The user is logged in and has items in their cart. |

### Use Case: View Cart

|  |  |
| --- | --- |
| **Use Case ID:** | UC-15 |
| **Use Case Name:** | View Cart |
| **Actors:** | Registered User |
| **Description:** | A registered user views the items in their shopping cart. |
| **Trigger:** | The user selects the option to view their cart. |
| **Preconditions:** | The user is logged in and has added items to the cart. |
| **Postconditions:** | The user has viewed the items in their cart. |
| **Normal Flow:** | 1. The user selects the option to view their cart. 2. The system displays the items in the user's cart. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Business Rules** | Only logged-in users can view their cart. |
| **Assumptions:** | The user is logged in and has added items to the cart. |

### Use Case: Update From Cart

|  |  |
| --- | --- |
| **Use Case ID:** | UC-016 |
| **Use Case Name:** | Update From Cart |
| **Actors:** | Registered User |
| **Description:** | A registered user updates the quantity or details of items in their shopping cart. |
| **Trigger:** | The user selects the option to update their cart. |
| **Preconditions:** | The user is logged in and has items in its cart. |
| **Postconditions:** | The user's cart is updated with the new item quantities or details. |
| **Normal Flow:** | 1. The user selects the option to update their cart. 2. The system displays the current items in the cart with editable fields for quantity and other details. 3. The user updates the desired fields and submits the changes. 4. The system updates the cart with the new information. |
| **Alternative Flows:** | If the user cancels the update process, the system returns to the cart view without making changes. |
| **Exceptions:** | None. |
| **Business Rules** | Only logged-in users can update their cart. |
| **Assumptions:** | The user is logged in and viewing their cart. |

### Use Case: Delete From Cart

|  |  |
| --- | --- |
| **Use Case ID:** | UC-017 |
| **Use Case Name:** | Delete From Cart |
| **Actors:** | Registered User |
| **Description:** | A registered user deletes items from their shopping cart. |
| **Trigger:** | The user selects the option to delete items from their cart. |
| **Preconditions:** | The user is logged in and has items in its cart. |
| **Postconditions:** | The selected items are removed from the user's cart. |
| **Normal Flow:** | 1. The user selects the option to delete items from their cart. 2. The system displays the current items in the cart with delete options. 3. The user selects the items to delete and confirms the deletion. 4. The system removes the selected items from the cart. |
| **Alternative Flows:** | If the user cancels the deletion process, the system returns to the cart view without making changes. |
| **Exceptions:** | None. |
| **Business Rules** | Only logged-in users can delete items from their cart. |
| **Assumptions:** | The user is logged in and viewing their cart. |

### Use Case: View Orders

|  |  |
| --- | --- |
| **Use Case ID:** | UC-018 |
| **Use Case Name:** | View Orders |
| **Actors:** | Registered User |
| **Description:** | A registered user views the list of their past and current orders. |
| **Trigger:** | The user selects the option to view their orders. |
| **Preconditions:** | The user is logged in. |
| **Postconditions:** | The user has viewed their order history. |
| **Normal Flow:** | 1. The user selects the option to view their orders. 2. The system displays a list of the user's past and current orders. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Business Rules** | Only logged-in users can view their order history. |
| **Assumptions:** | The user has placed orders in the past. |

### Use Case: View Order Status

|  |  |
| --- | --- |
| **Use Case ID:** | UC-019 |
| **Use Case Name:** | View Order Status |
| **Actors:** | Registered User |
| **Description:** | A registered user views the status of their current orders. |
| **Trigger:** | The user selects the option to view the status of their orders. |
| **Preconditions:** | The user is logged in and has placed orders. |
| **Postconditions:** | The user has viewed the status of their current orders. |
| **Normal Flow:** | 1. The user selects the option to view the status of their orders. 2. The system displays the status of the user's current orders. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Business Rules** | Only logged-in users can view their order status. |
| **Assumptions:** | The user has placed current orders. |

### Use Case: View Orders History

|  |  |
| --- | --- |
| **Use Case ID:** | UC-020 |
| **Use Case Name:** | View Orders History |
| **Actors:** | Registered User |
| **Description:** | A registered user views the history of their past orders, including details and status of each order. |
| **Trigger:** | The user selects the option to view their order history. |
| **Preconditions:** | The user is logged in. |
| **Postconditions:** | The user has viewed the details of their past orders. |
| **Normal Flow:** | 1. The user selects the option to view their order history. 2. The system retrieves the list of past orders for the user. 3. The system displays the order history, including order details and status. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Business Rules** | Only logged-in users can view their order history. |
| **Assumptions:** | The user has placed orders in the past and is logged in. |

### Use Case: Enter Payment Information

|  |  |
| --- | --- |
| **Use Case ID:** | UC-021 |
| **Use Case Name:** | Enter Payment Information |
| **Actors:** | Registered User |
| **Description:** | A registered user enters their payment information to complete the purchase of items in their cart. |
| **Trigger:** | The user proceeds to checkout and is prompted to enter payment information. |
| **Preconditions:** | The payment information is entered and validated. |
| **Postconditions:** | The user has viewed the details of their past orders. |
| **Normal Flow:** | 1. The user proceeds to checkout. 2. The system prompts the user to enter payment information. 3. The user enters the required payment details. 4. The system validates the payment information. |
| **Alternative Flows:** | If the user cancels the process, the system returns to the cart without saving the payment information.  If the payment information is invalid, the system displays an error message and prompts the user to re-enter the details. |
| **Exceptions:** | None. |
| **Business Rules** | Payment information must be valid and conform to standard formats. |
| **Assumptions:** | The user has valid payment information ready to enter. |

### Use Case: Select Shipping Method

|  |  |
| --- | --- |
| **Use Case ID:** | UC-022 |
| **Use Case Name:** | Select Shipping Method |
| **Actors:** | Registered User |
| **Description:** | A registered user selects a shipping method for the delivery of their purchased items. |
| **Trigger:** | The user proceeds to the shipping method selection step during checkout. |
| **Preconditions:** | The user is logged in, has items in their cart, and has entered valid payment information. |
| **Postconditions:** | The shipping method is selected and saved. |
| **Normal Flow:** | 1. The user proceeds to the shipping method selection step during checkout. 2. The system displays available shipping options. 3. The user selects their preferred shipping method. 4. The system saves the selected shipping method. |
| **Alternative Flows:** | If the user cancels the process, the system returns to the cart without saving the shipping method. |
| **Exceptions:** | None. |
| **Business Rules** | Only available shipping methods are presented to the user. |
| **Assumptions:** | The user has valid shipping information and options available. |

### Use Case: Place Order

|  |  |
| --- | --- |
| **Use Case ID:** | UC-023 |
| **Use Case Name:** | Place Order |
| **Actors:** | Registered User |
| **Description:** | A registered user places an order for items in their cart, completing the purchase process. |
| **Trigger:** | The user confirms the purchase at the end of the checkout process. |
| **Preconditions:** | The user is logged in, has items in their cart, and has entered valid payment and shipping information. |
| **Postconditions:** | The order is placed and confirmed. |
| **Normal Flow:** | 1. The user reviews the order summary and confirms the purchase. 2. The system processes the payment. 3. The system confirms the order and generates an order number. 4. The system sends an order confirmation to the user. |
| **Alternative Flows:** | If the payment fails, the system displays an error message and prompts the user to retry or enter new payment information.  If the user cancels the process, the system returns to the cart without placing the order. |
| **Exceptions:** | None. |
| **Business Rules** | The order can only be placed if all required information is valid and complete. |
| **Assumptions:** | The user has valid payment and shipping information, and the system is able to process the order. |

### Use Case: Add Furniture Listings

|  |  |
| --- | --- |
| **Use Case ID:** | UC-024 |
| **Use Case Name:** | Add Furniture Listings |
| **Actors:** | Admin |
| **Description:** | The admin adds new furniture listings to the system, making them available for users to view and purchase. |
| **Trigger:** | The admin chooses to add a new furniture listing through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The new furniture listing is saved and displayed in the inventory and available for users. |
| **Normal Flow:** | 1. The admin navigates to the add furniture listings page. 2. The admin enters the necessary details (name, description, price, etc.). 3. The system validates the entered details. 4. The system saves the new furniture listing. 5. The system confirms the addition of the new listing to the admin. |
| **Alternative Flows:** | If validation fails, the system prompts the admin to correct the errors |
| **Exceptions:** | None |
| **Business Rules** | The listing must include all mandatory fields. |
| **Assumptions:** | The admin has accurate and complete information to add the listing. |

### Use Case: Update Furniture Listings

|  |  |
| --- | --- |
| **Use Case ID:** | UC-025 |
| **Use Case Name:** | Update Furniture Listings |
| **Actors:** | Admin |
| **Description:** | The admin updates details of existing furniture listings. |
| **Trigger:** | The admin selects a furniture listing to update through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The updated furniture listing is saved and displayed in the inventory. |
| **Normal Flow:** | 1. The admin navigates to the list of furniture listings. 2. The admin selects a listing to update. 3. The admin modifies the necessary details. 4. The system validates the updated details. 5. The system saves the updated listing. 6. The system confirms the update to the admin. |
| **Alternative Flows:** | If validation fails, the system prompts the admin to correct the errors |
| **Exceptions:** | None |
| **Business Rules** | The listing must include all mandatory fields. |
| **Assumptions:** | The admin has accurate and complete information to update the listing. |

### Use Case: Delete Furniture Listings

|  |  |
| --- | --- |
| **Use Case ID:** | UC-026 |
| **Use Case Name:** | Delete Furniture Listings |
| **Actors:** | Admin |
| **Description:** | The admin removes furniture listings from the system. |
| **Trigger:** | The admin chooses to delete a furniture listing through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The furniture listing is removed from the inventory. |
| **Normal Flow:** | 1. The admin navigates to the list of furniture listings. 2. The admin selects a listing to delete. 3. The system prompts for confirmation. 4. The admin confirms the deletion. 5. The system removes the listing. 6. The system confirms the deletion to the admin. |
| **Alternative Flows:** | If the admin cancels the confirmation, the listing is not deleted. |
| **Exceptions:** | None |
| **Business Rules** | The listing can only be deleted if it is not associated with any active orders. |
| **Assumptions:** | The admin has the authority to delete listings. |

### Use Case: Add 3D Models

|  |  |
| --- | --- |
| **Use Case ID:** | UC-027 |
| **Use Case Name:** | Add 3D Models |
| **Actors:** | Admin |
| **Description:** | The admin adds 3D models to furniture listings. |
| **Trigger:** | The admin chooses to add a 3D model through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The 3D model is saved and associated with the selected furniture listing. |
| **Normal Flow:** | 1. The admin navigates to the 3D models section. 2. The admin selects a furniture listing to associate the 3D model with. 3. The admin uploads the 3D model. 4. The system validates the 3D model file. 5. The system saves the 3D model. 6. The system confirms the addition of the 3D model to the admin |
| **Alternative Flows:** | If validation fails, the system prompts the admin to correct the errors. |
| **Exceptions:** | None |
| **Business Rules** | The 3D model must meet specified format and size requirements. |
| **Assumptions:** | The admin has the necessary 3D model file available for upload. |

### Use Case: Update 3D Models

|  |  |
| --- | --- |
| **Use Case ID:** | UC-028 |
| **Use Case Name:** | Update 3D Models |
| **Actors:** | Admin |
| **Description:** | The admin updates 3D models associated with furniture listings. |
| **Trigger:** | The admin selects a 3D model to update through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The updated 3D model is saved and associated with the selected furniture listing. |
| **Normal Flow:** | 1. The admin navigates to the 3D models section. 2. The admin selects a 3D model to update. 3. The admin uploads the updated 3D model. 4. The system validates the 3D model file. 5. The system saves the updated 3D model. 6. The system confirms the update of the 3D model to the admin. |
| **Alternative Flows:** | If validation fails, the system prompts the admin to correct the errors. |
| **Exceptions:** | None |
| **Business Rules** | The 3D model must meet specified format and size requirements. |
| **Assumptions:** | The admin has the necessary 3D model file available for upload. |

### Use Case: Delete 3D Models

|  |  |
| --- | --- |
| **Use Case ID:** | UC-029 |
| **Use Case Name:** | Delete 3D Models |
| **Actors:** | Admin |
| **Description:** | The admin removes 3D models from furniture listings. |
| **Trigger:** | The admin chooses to delete a 3D model through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The 3D model is removed from the selected furniture listing. |
| **Normal Flow:** | 1. The admin selects a 3D model to delete. 2. The system prompts for confirmation. 3. The admin confirms the deletion. 4. The system removes the 3D model. 5. The system confirms the deletion to the admin. |
| **Alternative Flows:** | If the admin cancels the confirmation, the 3D model is not deleted. |
| **Exceptions:** | None |
| **Business Rules** | The 3D model can only be deleted if it is not associated with any active listings. |
| **Assumptions:** | The admin has the authority to delete 3D models. |

### Use Case: Generate Inventory Reports

|  |  |
| --- | --- |
| **Use Case ID:** | UC-030 |
| **Use Case Name:** | Generate Inventory Reports |
| **Actors:** | Admin |
| **Description:** | The admin generates reports on the inventory, including stock levels, stock movements, and reorder levels. |
| **Trigger:** | The admin chooses to generate an inventory report through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The inventory report is generated and available for viewing or download. |
| **Normal Flow:** | 1. The admin navigates to the reports section. 2. The admin selects the type of inventory report to generate. 3. The system compiles the relevant data. 4. The system generates the report. 5. The system presents the report to the admin. |
| **Alternative Flows:** | None. |
| **Exceptions:** | If there is insufficient data to generate the report, the system notifies the admin. |
| **Business Rules** | The report can only include data the admin has permission to view. |
| **Assumptions:** | The system has accurate and up-to-date inventory data. |

### Use Case: Generate Inventory Reports

|  |  |
| --- | --- |
| **Use Case ID:** | UC-031 |
| **Use Case Name:** | Generate Inventory Reports |
| **Actors:** | Admin |
| **Description:** | The admin generates reports on the inventory, including stock levels, stock movements, and reorder levels. |
| **Trigger:** | The admin chooses to generate an inventory report through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The inventory report is generated and available for viewing or download. |
| **Normal Flow:** | 1. The admin navigates to the reports section. 2. The admin selects the type of inventory report to generate. 3. The system compiles the relevant data. 4. The system generates the report. 5. The system presents the report to the admin. |
| **Alternative Flows:** | None. |
| **Exceptions:** | If there is insufficient data to generate the report, the system notifies the admin. |
| **Business Rules** | The report can only include data the admin has permission to view. |
| **Assumptions:** | The system has accurate and up-to-date inventory data. |

### Use Case: Update Order Status

|  |  |
| --- | --- |
| **Use Case ID:** | UC-032 |
| **Use Case Name:** | Update Order Status |
| **Actors:** | Admin |
| **Description:** | This use case allows the admin to update the status of a customer's order in the system. |
| **Trigger:** | The admin decides to update the status of an order based on its current state in the fulfillment process. |
| **Preconditions:** | The admin is logged into the system.  There are existing orders in the system. |
| **Postconditions:** | The order status is updated in the system.  The customer is notified of the status change. |
| **Normal Flow:** | 1. The admin selects the order management option from the admin dashboard. 2. The admin views the list of existing orders. 3. The admin selects a specific order to update. 4. The admin changes the order status to the appropriate state (e.g., 'Processing', 'Shipped', 'Delivered', 'Cancelled'). 5. The admin confirms the status update. 6. The system saves the updated status and sends a notification to the customer. |
| **Alternative Flows:** | If the selected order does not exist, an error message is displayed.  If the status update fails due to a system error, the admin is prompted to try again later. |
| **Exceptions:** | System error occurs during the status update process. |
| **Business Rules** | Only authorized Admin users can update order statuses.  The system must log all status changes for audit purposes. |
| **Assumptions:** | The admin has the necessary permissions to update order statuses.  The system is operational and accessible when the admin attempts to update an order status. |

### Use Case: View Furniture Listings on Admin Panel

|  |  |
| --- | --- |
| **Use Case ID:** | UC-033 |
| **Use Case Name:** | View Furniture Listings on Admin Panel |
| **Actors:** | Admin |
| **Description:** | This use case allows an admin to view the available furniture listings in the store for administrative purposes. |
| **Trigger:** | The admin decides to review the furniture listings for inventory management or other administrative tasks. |
| **Preconditions:** | The system contains a catalog of furniture item. |
| **Postconditions:** | The admin can access detailed information about furniture items for administrative purposes. |
| **Normal Flow:** | 1. The admin navigates to the furniture listings section. 2. The system retrieves the list of furniture listings. 3. The system displays the list to the admin. |
| **Alternative Flows:** | If no furniture items are available, a message indicating no items are available is displayed. |
| **Exceptions:** | System error occurs while retrieving the furniture listings. |
| **Business Rules** | The admin has full access to view all furniture listings and associated details. The system must update the furniture listings in real-time to reflect current availability. |
| **Assumptions:** | The system is operational and can retrieve furniture listings without delays. |

### Use Case: Shows Sales & Stock Dashboard

|  |  |
| --- | --- |
| **Use Case ID:** | UC-034 |
| **Use Case Name:** | Shows Sales & Stock Dashboard |
| **Actors:** | Admin |
| **Description:** | The admin views a dashboard that displays sales performance and stock levels. |
| **Trigger:** | The admin chooses to view the sales and stock dashboard through the admin panel. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | The sales and stock dashboard is displayed. |
| **Normal Flow:** | 1. The admin navigates to the dashboard section. 2. The system retrieves sales and stock data. 3. The system compiles the data into a dashboard format. 4. The system displays the dashboard to the admin. |
| **Alternative Flows:** | None. |
| **Exceptions:** | If there is insufficient data, the system notifies the admin. |
| **Business Rules** | The dashboard can only include data the admin has permission to view. |
| **Assumptions:** | The system has accurate and up-to-date sales and stock data. |

### Use Case: Secures Sales and Updates Stocks

|  |  |
| --- | --- |
| **Use Case ID:** | UC-035 |
| **Use Case Name:** | Secures Sales and Updates Stocks |
| **Actors:** | Admin |
| **Description:** | The admin secures sales transactions and updates stock levels accordingly. |
| **Trigger:** | Sales transactions occur and stock levels need updating. |
| **Preconditions:** | The admin is logged in and has the necessary permissions. |
| **Postconditions:** | Sales transactions are secured and stock levels are updated. |
| **Normal Flow:** | 1. Sales transactions are recorded in the system. 2. The system updates stock levels based on the transactions. 3. The system secures the sales data. 4. The system confirms the stock update to the admin. |
| **Alternative Flows:** | If stock levels cannot be updated, the system notifies the admin. |
| **Exceptions:** | None. |
| **Business Rules** | Stock levels must be accurately updated based on sales. |
| **Assumptions:** | Sales transactions are accurately recorded. |

### Use Case: Logout

|  |  |
| --- | --- |
| **Use Case ID:** | UC-036 |
| **Use Case Name:** | Logout |
| **Actors:** | Registered User, Admin |
| **Description:** | A registered user and admin log out of their account. |
| **Trigger:** | The user selects the option to log out. |
| **Preconditions:** | The user is logged in. |
| **Postconditions:** | The user is logged out, and their session is terminated. |
| **Normal Flow:** | 1. The user selects the option to log out. 2. The system terminates the user's session and returns to the home page. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Business Rules** | The system must securely terminate the user's session. |
| **Assumptions:** | The user is logged in. |

## Functional Requirements

### Functional Requirement for UC-1: Register

|  |  |
| --- | --- |
| **Identifier** | FR-01 |
| **Title** | Register |
| **Requirement** | The system shall allow visitors to create a new account by providing necessary details such as name, email, and password. |
| **Source** | Visitor |
| **Rationale** | To enable visitors to become registered users and access additional functionalities. |
| **Business Rule (if required)** | Password must meet security criteria (e.g., minimum length, contains letters and numbers). |
| **Dependencies** | None |

### Functional Requirement for UC-2: Login

|  |  |
| --- | --- |
| **Identifier** | FR-02 |
| **Title** | Login |
| **Requirement** | The system shall allow registered users to log in using their email and password. |
| **Source** | Registered User |
| **Rationale** | To provide access to personalized features and functionalities. |
| **Business Rule (if required)** | Valid email and password combination required. |
| **Dependencies** | User must be registered. |

### Functional Requirement for UC-3: Forgot Details

|  |  |
| --- | --- |
| **Identifier** | FR-03 |
| **Title** | Forgot Details |
| **Requirement** | The system shall allow users to recover their account details if they forget them. |
| **Source** | Registered User |
| **Rationale** | To ensure users can regain access to their account. |
| **Business Rule (if required)** | Verification through email or security questions. |
| **Dependencies** | User must have a registered account. |

### Functional Requirement for UC-4: View Furniture

|  |  |
| --- | --- |
| **Identifier** | FR-04 |
| **Title** | View Furniture |
| **Requirement** | The system shall allow visitors and registered users to browse available furniture. |
| **Source** | Visitor/Registered User/Admin |
| **Rationale** | To provide users with access to the product catalog. |
| **Business Rule (if required)** | None |
| **Dependencies** | None |

### Functional Requirement for UC-5: View AR Preview of Furniture

|  |  |
| --- | --- |
| **Identifier** | FR-05 |
| **Title** | View AR Preview of Furniture |
| **Requirement** | The system shall allow users to view an augmented reality preview of furniture. |
| **Source** | Visitor/Registered User |
| **Rationale** | To help users visualize furniture in their space. |
| **Business Rule (if required)** | Requires device with AR capabilities. |
| **Dependencies** | None |

### Functional Requirement for UC-6: View Similar Furniture

|  |  |
| --- | --- |
| **Identifier** | FR-06 |
| **Title** | View Similar Furniture |
| **Requirement** | The system shall suggest similar furniture based on the user's selection. |
| **Source** | Registered User |
| **Rationale** | To assist users in finding alternatives. |
| **Business Rule (if required)** | Suggestions based on category and attributes. |
| **Dependencies** | User must be viewing a furniture item. |

### Functional Requirement for UC-7: View Customer Reviews

|  |  |
| --- | --- |
| **Identifier** | FR-07 |
| **Title** | View Customer Reviews |
| **Requirement** | The system shall allow users to view reviews from other customers. |
| **Source** | Registered User, Admin, visitor |
| **Rationale** | To help users make informed decisions. |
| **Business Rule (if required)** | Display all reviews and ratings. |
| **Dependencies** | None. |

### Functional Requirement for UC-8: View 3D Model

|  |  |
| --- | --- |
| **Identifier** | FR-08 |
| **Title** | View 3D Model |
| **Requirement** | The system shall allow users to view 3D models of the furniture. |
| **Source** | Registered User |
| **Rationale** | To provide a detailed visual representation. |
| **Business Rule (if required)** | Requires device with 3D model viewing capabilities. |
| **Dependencies** | User Registration and Authentication |

### Functional Requirement for UC-9: Search by Filters

|  |  |
| --- | --- |
| **Identifier** | FR-9 |
| **Title** | Search by Filters |
| **Requirement** | The system shall allow users to filter furniture based on various criteria (e.g., price, category). |
| **Source** | Admin/User/Visitor |
| **Rationale** | To help users find furniture that meets their preferences. |
| **Business Rule (if required)** | Filter options based on product attributes. |
| **Dependencies** | None. |

### Functional Requirement for UC-10: View Listing Details

|  |  |
| --- | --- |
| **Identifier** | FR-10 |
| **Title** | View Listing Details |
| **Requirement** | The system shall display detailed information about a furniture listing. |
| **Source** | Admin/User/Visitor |
| **Rationale** | To provide users with all necessary information about the product. |
| **Business Rule (if required)** | Must include price, description, dimensions, and available stock. |
| **Dependencies** | None. |

### Functional Requirement for UC-11: Augment Multiple Furniture Items from List

|  |  |
| --- | --- |
| **Identifier** | FR-11 |
| **Title** | Augment Multiple Furniture Items from List |
| **Requirement** | It enables users to visualize multiple furniture items simultaneously within their physical space. |
| **Source** | Registered User, Visitor |
| **Rationale** | To enable users to visualize how multiple pieces of furniture will look together in their space. |
| **Business Rule (if required)** | Multiple items must be displayed accurately and maintain spatial relationships. |
| **Dependencies** | User must be logged in and have a compatible device for AR. |

### Functional Requirement for UC-12: Write Review

|  |  |
| --- | --- |
| **Identifier** | FR-12 |
| **Title** | Write Review |
| **Requirement** | The system shall allow registered users to write reviews for purchased furniture. |
| **Source** | Registered User |
| **Rationale** | To enable user feedback and ratings. |
| **Business Rule (if required)** | Review linked to verified purchase. |
| **Dependencies** | User must be logged in and have purchased the item. |

### Functional Requirement for UC-13: Add to Cart

|  |  |
| --- | --- |
| **Identifier** | FR-13 |
| **Title** | Add to Cart |
| **Requirement** | The system shall allow registered users to add furniture items to their shopping cart. |
| **Source** | Registered User |
| **Rationale** | To enable users to prepare for purchase. |
| **Business Rule (if required)** | Item must be in stock. |
| **Dependencies** | User must be logged in. |

### Functional Requirement for UC-14: View Cart

|  |  |
| --- | --- |
| **Identifier** | FR-14 |
| **Title** | View Cart |
| **Requirement** | The system shall allow users to view items in their cart. |
| **Source** | Registered User |
| **Rationale** | To review selected items before purchase. |
| **Business Rule (if required)** | Display all items, quantities, and total price. |
| **Dependencies** | User must have items in the cart. |

### Functional Requirement for UC-15: Update from Cart

|  |  |
| --- | --- |
| **Identifier** | FR-15 |
| **Title** | Update From Cart |
| **Requirement** | The system shall allow users to update quantities or remove items from the cart. |
| **Source** | Registered User |
| **Rationale** | To allow users to modify their purchase selection. |
| **Business Rule (if required)** | Must update total price accordingly. |
| **Dependencies** | User must be logged in and have items in the cart. |

### Functional Requirement for UC-16: Proceed to Checkout

|  |  |
| --- | --- |
| **Identifier** | FR-16 |
| **Title** | Proceed to Checkout |
| **Requirement** | The system shall allow users to proceed to checkout with items in their cart. |
| **Source** | Registered User |
| **Rationale** | To initiate the purchase process. |
| **Business Rule (if required)** | Must have valid items in the cart. |
| **Dependencies** | User must be logged in. |

### Functional Requirement for UC-17: Enter Payment Information

|  |  |
| --- | --- |
| **Identifier** | FR-17 |
| **Title** | Enter Payment Information |
| **Requirement** | The system shall allow users to enter their payment details. |
| **Source** | Registered User |
| **Rationale** | To complete the purchase transaction. |
| **Business Rule (if required)** | Secure payment gateway required. |
| **Dependencies** | User must be proceeding to checkout. |

### Functional Requirement for UC-18: Select Shipping Method

|  |  |
| --- | --- |
| **Identifier** | FR-18 |
| **Title** | Select Shipping Method |
| **Requirement** | The system shall allow users to choose a shipping method. |
| **Source** | Registered User |
| **Rationale** | To provide options for delivery. |
| **Business Rule (if required)** | Display available shipping methods and costs. |
| **Dependencies** | User must be proceeding to checkout. |

### Functional Requirement for UC-19: Place Order

|  |  |
| --- | --- |
| **Identifier** | FR-19 |
| **Title** | Place Order |
| **Requirement** | The system shall allow users to place an order after entering payment and shipping information. |
| **Source** | Registered User |
| **Rationale** | To complete the purchase process. |
| **Business Rule (if required)** | Confirm order details and deduct from stock. |
| **Dependencies** | User must have valid payment and shipping information. |

### Functional Requirement for UC-20: View Orders

|  |  |
| --- | --- |
| **Identifier** | FR-20 |
| **Title** | View Orders |
| **Requirement** | The system shall allow users to view their current and past orders. |
| **Source** | Registered User |
| **Rationale** | To track purchase history and order status. |
| **Business Rule (if required)** | Display order details, status, and history. |
| **Dependencies** | User must be logged in. |

### Functional Requirement for UC-21: View Order Status

|  |  |
| --- | --- |
| **Identifier** | FR-21 |
| **Title** | View Order Status |
| **Requirement** | The system shall allow users to view the status of their orders. |
| **Source** | Registered User |
| **Rationale** | To provide updates on the progress of their orders. |
| **Business Rule (if required)** | Display real-time status updates. |
| **Dependencies** | User must have orders placed. |

### Functional Requirement for UC-22: View Orders History

|  |  |
| --- | --- |
| **Identifier** | FR-22 |
| **Title** | View Orders History |
| **Requirement** | The system shall allow users to view the history of all their past orders. |
| **Source** | Registered User |
| **Rationale** | To provide a record of all previous transactions. |
| **Business Rule (if required)** | Show complete order history with details. |
| **Dependencies** | User must be logged in. |

### Functional Requirement for UC-23: Delete from Cart

|  |  |
| --- | --- |
| **Identifier** | FR-23 |
| **Title** | View Orders History |
| **Requirement** | The system shall allow users to remove items from their shopping cart. |
| **Source** | Registered User |
| **Rationale** | To enable users to manage their cart contents before checkout. |
| **Business Rule (if required)** | Item should be removed from the cart immediately upon user confirmation. |
| **Dependencies** | User must be logged in and have items in the cart. |

### Functional Requirement for UC-24: Generate Inventory Report

|  |  |
| --- | --- |
| **Identifier** | FR-24 |
| **Title** | Generate Inventory Report |
| **Requirement** | The system shall allow admins to generate reports on inventory status. |
| **Source** | Admin |
| **Rationale** | To monitor and manage stock levels. |
| **Business Rule (if required)** | Reports must include current stock levels and sales data. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-25: Secure Sales and Update Stocks

|  |  |
| --- | --- |
| **Identifier** | FR-25 |
| **Title** | Secure Sales and Update Stocks |
| **Requirement** | The system shall securely process sales transactions and update stock levels accordingly. |
| **Source** | Admin |
| **Rationale** | To ensure accurate inventory management and secure transactions. |
| **Business Rule (if required)** | Must integrate with payment gateway and inventory system. |
| **Dependencies** | Sale transaction initiated. |

### Functional Requirement for UC-26: Show Sales and Stock Dashboard

|  |  |
| --- | --- |
| **Identifier** | FR-26 |
| **Title** | Show Sales and Stock Dashboard |
| **Requirement** | The system shall provide a dashboard for admins to view sales and stock levels. |
| **Source** | Admin |
| **Rationale** | To provide a comprehensive overview of business metrics. |
| **Business Rule (if required)** | Dashboard must be updated in real-time. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-27: Add Furniture Listings

|  |  |
| --- | --- |
| **Identifier** | FR-27 |
| **Title** | Add Furniture Listings |
| **Requirement** | The system shall allow admins to add new furniture listings. |
| **Source** | Admin |
| **Rationale** | To expand the product catalog. |
| **Business Rule (if required)** | Must include all required details (e.g., name, price, description). |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-28: Update Furniture Listings

|  |  |
| --- | --- |
| **Identifier** | FR-28 |
| **Title** | Update Furniture Listings |
| **Requirement** | The system shall allow admins to update existing furniture listings. |
| **Source** | Admin |
| **Rationale** | To keep product information current. |
| **Business Rule (if required)** | Must allow updates to all listing details. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-29: Delete Furniture Listings

|  |  |
| --- | --- |
| **Identifier** | FR-29 |
| **Title** | Delete Furniture Listings |
| **Requirement** | The system shall allow admins to delete furniture listings. |
| **Source** | Admin |
| **Rationale** | To manage the product catalog by removing outdated or unavailable items. |
| **Business Rule (if required)** | Must ensure confirmation before deletion. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-30: Add 3D Models

|  |  |
| --- | --- |
| **Identifier** | FR-30 |
| **Title** | Add 3D Models |
| **Requirement** | The system shall allow admins to add 3D models for furniture listings. |
| **Source** | Admin |
| **Rationale** | To provide enhanced visualization of products. |
| **Business Rule (if required)** | Models must meet specified format requirements. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-31: Update 3D Models

|  |  |
| --- | --- |
| **Identifier** | FR-31 |
| **Title** | Update 3D Models |
| **Requirement** | The system shall allow admins to update existing 3D models for furniture listings. |
| **Source** | Admin |
| **Rationale** | To ensure accurate representation of products. |
| **Business Rule (if required)** | Models must meet specified format requirements. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-32: Delete 3D Models

|  |  |
| --- | --- |
| **Identifier** | FR-32 |
| **Title** | Delete 3D Models |
| **Requirement** | The system shall allow admins to delete 3D models for furniture listings. |
| **Source** | Admin |
| **Rationale** | To manage the 3D model repository. |
| **Business Rule (if required)** | Must ensure confirmation before deletion. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-33: Update Order Status

|  |  |
| --- | --- |
| **Identifier** | FR-33 |
| **Title** | Update Order Status |
| **Requirement** | The system shall allow admins to update the status of customer orders. |
| **Source** | Admin |
| **Rationale** | To provide accurate order tracking. |
| **Business Rule (if required)** | Status updates should reflect real-time changes. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-34: View Furniture Listings on Admin Panel

|  |  |
| --- | --- |
| **Identifier** | FR-34 |
| **Title** | View Furniture Listings on Admin Panel |
| **Requirement** | The system shall allow admins to view all furniture listings. |
| **Source** | Admin |
| **Rationale** | To manage the product catalog. |
| **Business Rule (if required)** | Display all listings with details. |
| **Dependencies** | Admin must be logged in. |

### Functional Requirement for UC-35: Logout

|  |  |
| --- | --- |
| **Identifier** | FR-35 |
| **Title** | Logout |
| **Requirement** | The system shall allow users to log out from their account. |
| **Source** | Registered User |
| **Rationale** | To secure the account when not in use. |
| **Business Rule (if required)** | None |
| **Dependencies** | User must be logged in. |

## Non-Functional Requirements

### Usability

**Usability-1: User-Friendly Interface**

* **Description:** The system must have a user-friendly interface that is easy to navigate and understand.
* **Rationale:** To ensure a positive user experience and easy adoption.
* **Verification:** User testing and feedback.

**Usability-2: Clear Instructions**

* **Description:** The system must provide clear and concise instructions for using each feature.
* **Rationale:** To facilitate user understanding and reduce the learning curve.
* **Verification:** Documentation and user feedback.

**Usability-3: Intuitiveness**

* **Description:** The system must be intuitive and allow users to easily perform tasks and access information.
* **Rationale:** To enhance user productivity and satisfaction.
* **Verification:** User feedback and usability testing.

**Usability-4: Accessibility**

* **Description:** The system must be accessible to users with disabilities, complying with WCAG 2.1 Level AA guidelines.
* **Rationale:** To ensure inclusivity and usability for all users.
* **Verification:** Accessibility testing and compliance checks.

**Usability-5: Error Handling**

* **Description:** The system must provide clear error messages with guidance for resolving issues.
* **Rationale:** To help users recover from errors quickly and easily.
* **Verification:** User testing and feedback on error recovery processes.

### Performance

**PER-1: Fast Loading and Execution**

* **Description:** The system must be optimized for fast loading and execution.
* **Rationale:** To provide a responsive user experience.
* **Verification:** Measured system response times.

**PER-2: Scalability**

* **Description:** The system must be able to handle a high volume of requests without experiencing significant delays.
* **Rationale:** To accommodate potential growth in usage.
* **Verification:** Load testing and response time measurements.

**PER-3: Large Data Handling**

* **Description:** The system must be able to handle large data sets efficiently.
* **Rationale:** To ensure efficient processing of substantial amounts of data.
* **Verification:** Performance testing with large datasets.

**PER-4: Concurrent Users**

* **Description:** The system must support at least 1000 concurrent users without performance degradation.
* **Rationale:** To ensure the website can handle peak usage times.
* **Verification:** Stress testing and load testing.

**PER-5: Transaction Processing**

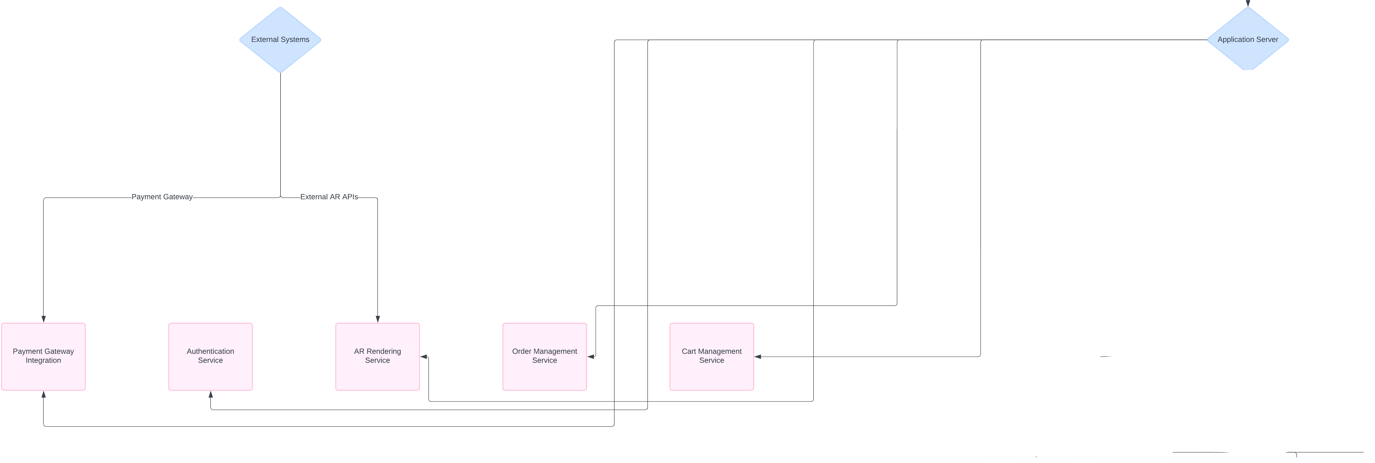
* **Description:** The system must handle up to 200 transactions per minute during peak shopping times without issues.
* **Rationale:** To ensure reliable performance during high-traffic periods.
* **Verification:** Transaction load testing.

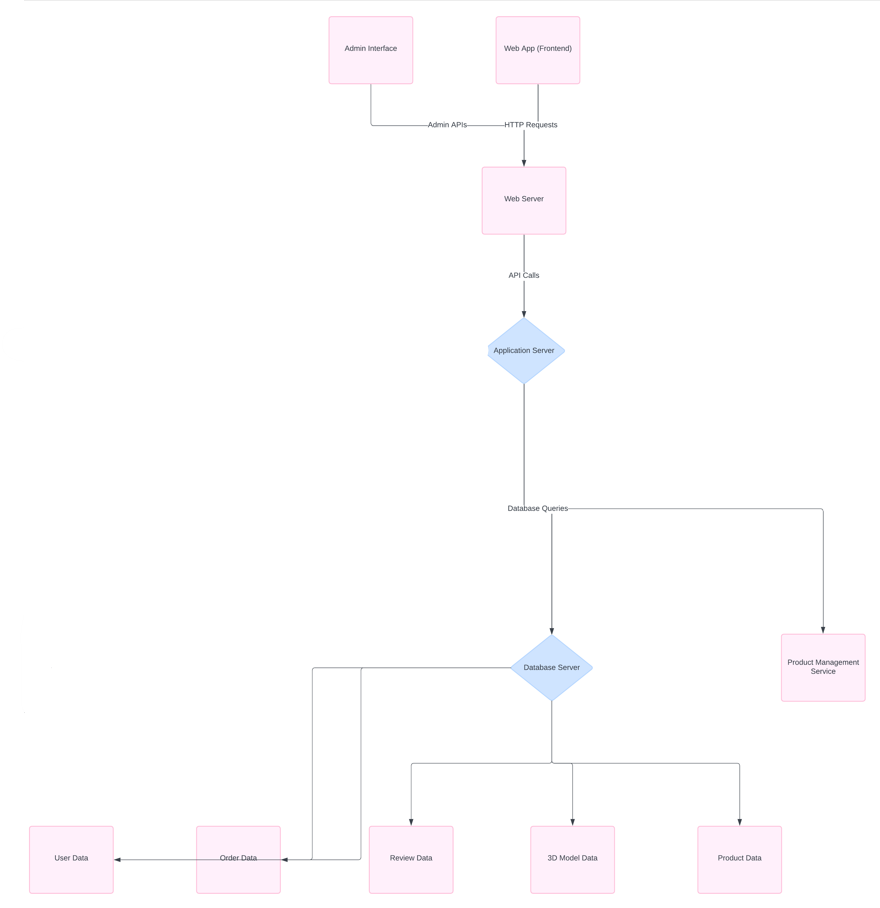
By adhering to these non-functional requirements, FurnishARt will ensure a robust, user-friendly, and high-performing AR-based online furniture store.

# Design and Architecture

In this chapter, Design and Architecture of our system furnishARt (An AR-Based Furniture Store) is explained.

## System Architecture

****



*Fig 4. 1: Architecture Diagram*

## Data Representation [Class Diagram]

### Website Application Class Diagram

*Fig 4. 2: Website Application Class Diagram*

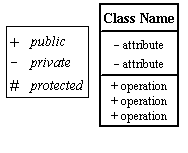
### AR App Class Diagram:

*Fig 4. 3: AR App Class Diagram*

*Fig 4.2: AR App Class Diagram*

**Legends**

A white background with black text

Description automatically generatedA black line on a white background

Description automatically generated

Composition

Association

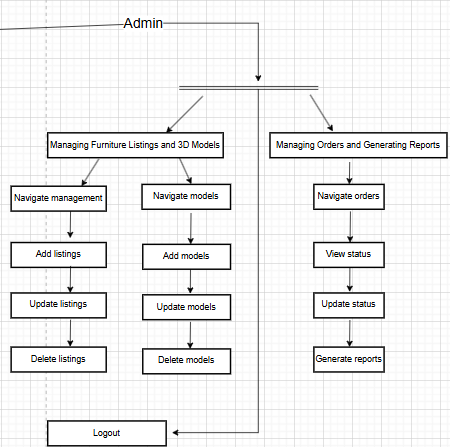
## Process Flow/Representation

Process Flow are given as follow:

### **User Activity Diagram**:

*Fig 4.3: User Activity Diagram*

### Admin Activity Diagram:



*Fig 4.3: Admin Activity Diagram*

### ERD diagram

A diagram of a computer program

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

*Fig 4.3: ER Diagram*

## Design Model

Design model of our system is given in this section

### Sequence Diagram

A screenshot of a computer screen

Description automatically generated

A screenshot of a program

Description automatically generated

A screenshot of a computer program

Description automatically generated

*Fig 4.4: Sequence Diagram*

# Implementation

Implementation includes the algorithms that used to build the project and user interface through which the user interacts with the system

## Algorithm

An algorithm is a step-by-step procedure or set of instructions designed to solve a specific problem or perform a particular task. Following are the algorithms that used in furnishARt.

### User Authentication (Sign Up and Login)

// Pseudocode for Signup

function createCustomerAndSendVerification(first\_name, last\_name, email, username, password, address):

check if email is unique

create a new customer record with the provided details

send verification email to the customer

return success message with customer details and verification message

// Pseudocode for Login

function authenticateUser(email, password):

find the user by email

if user does not exist or password is incorrect:

throw error "Invalid email or password"

if user is not verified:

send verification email and return message that user is not verified

generate an access token for the user

return access token and customer details (excluding password)

### Cart Management (Add, Get, Delete Cart Items)

// Pseudocode for addCartItem:

function addCartItem(userId, furnitureId, quantity):

item = getFurnitureItemPrice(furnitureId)

if item does not exist:

throw error "Item not found"

cart = findCustomerCart(userId)

if cart does not exist:

create a new cart for the user

add the item to the cart

else:

cartItem = findCartItem(cart.id, furnitureId)

if cartItem exists:

update the quantity of the existing cart item

else:

add the item to the cart

return updated cart

// Pseudocode for getCartItems

function getCartItems(userId):

cart = getCartItems(userId)

if cart does not exist:

throw error "Cart not found"

mappedCart = {

cart\_id: cart.id,

cart\_total\_price: cart.price,

cartItems: map cart items to a structured response

}

return mappedCart

// Pseudocode for deleteCartItem:

function deleteCartItem(userId, furniture\_item\_id):

cart = findCustomerCart(userId)

if cart does not exist:

throw error "Cart not found"

cartItem = findCartItem(cart.id, furniture\_item\_id)

if cartItem does not exist:

throw error "Item not found in cart"

delete the cart item from the database

update the cart total price by removing the item price

return updated cart

### Creating Payment Intent

if is\_pm\_save is true:

save the payment method to the customer

fetch cart details for the user

if cart price is invalid or empty:

throw an error

create a payment intent with cart price and payment method

if payment fails:

throw an error

else:

move cart items to order and delete the cart

return the payment details

### Product Placement:

// Pseudocode for Product Placement in AR Environment

function placeProductInAR(productId, userPosition, surfaceData) {

// Fetch product model and details

productModel = database.getProductModel(productId);

// Validate if surface is suitable for placement

if (isValidSurface(surfaceData)) {

// Calculate optimal placement coordinates

placementCoords = calculatePlacement(userPosition, surfaceData);

// Place product model in AR space

AR.renderModel(productModel, placementCoords);

return true; // Placement successful

}

return false; // Placement failed due to invalid surface

}

// Helper function: Validate surface for placement

function isValidSurface(surfaceData) {

return surfaceData.isFlat && surfaceData.isLargeEnough;

}

// Helper function: Calculate placement coordinates

function calculatePlacement(userPosition, surfaceData) {

// Compute offset to ensure visibility and alignment

offset = computeOffset(surfaceData);

return userPosition + offset;

}

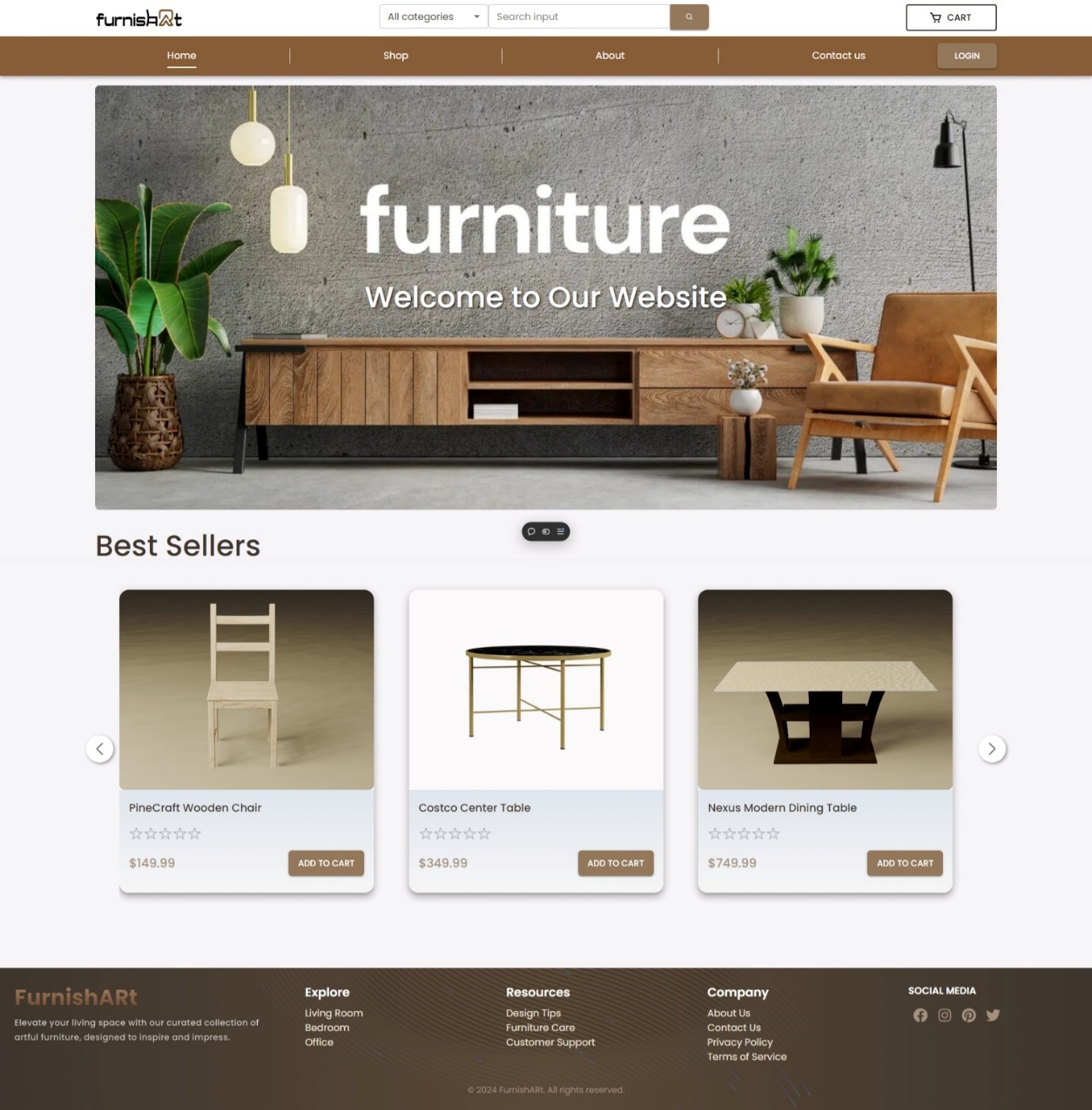
## External APIs

*Table 5. 1 External Api used*

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of API** | **Description** | **Purpose of usage** | **Function/Class Name** |
| NodeMailer | A library for sending emails in Node.js | Sending emails for notifications, confirmations, etc. | sendEmail |
| Stripe | A payment processing platform for online payments and customer management | Handling payments, creating customers, saving payment methods, creating payment intents | createStripeCustomer, savePaymentMethod, createPaymentIntent |
| Cloudflare | A platform for web performance, security, and content delivery | Storing assets (images, models), improving site performance and security | uploadAsset  fetchAsset |

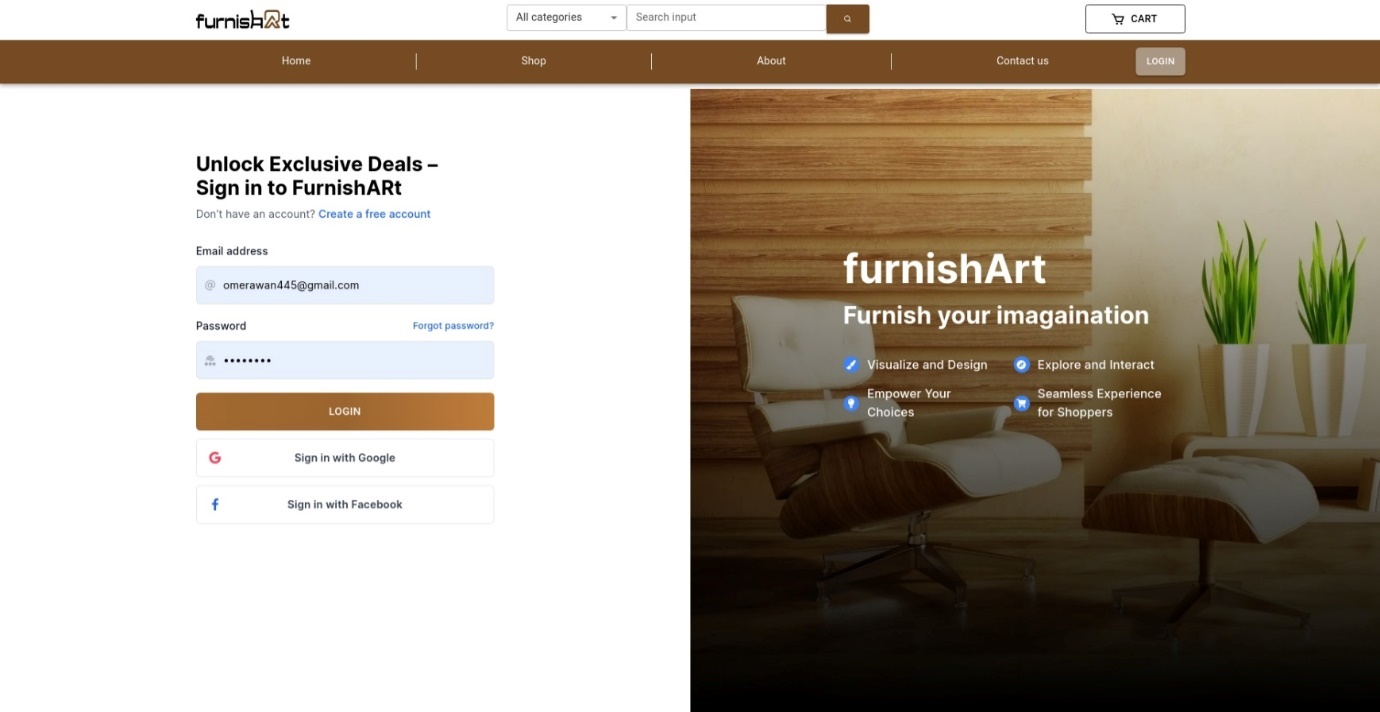
## User Interface

**Home Page**



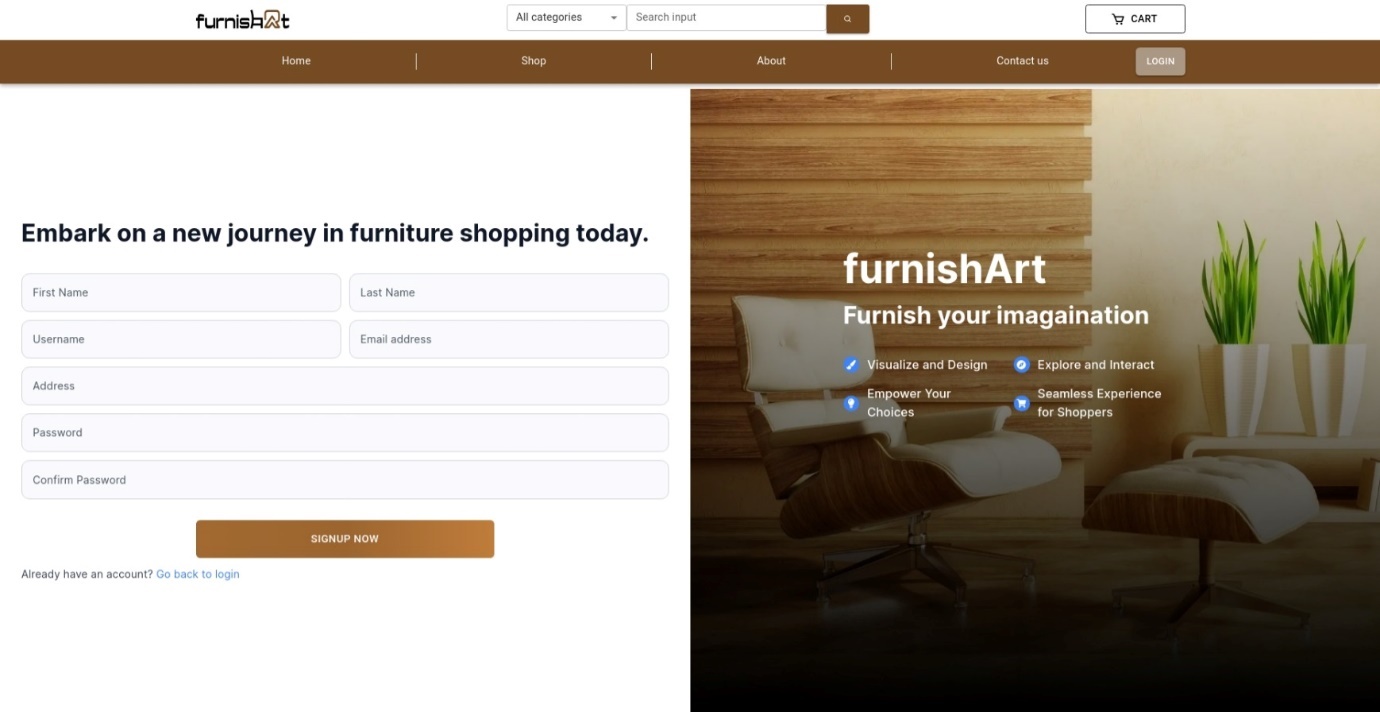
*Fig 5.1 Home Page*

**Login Page**



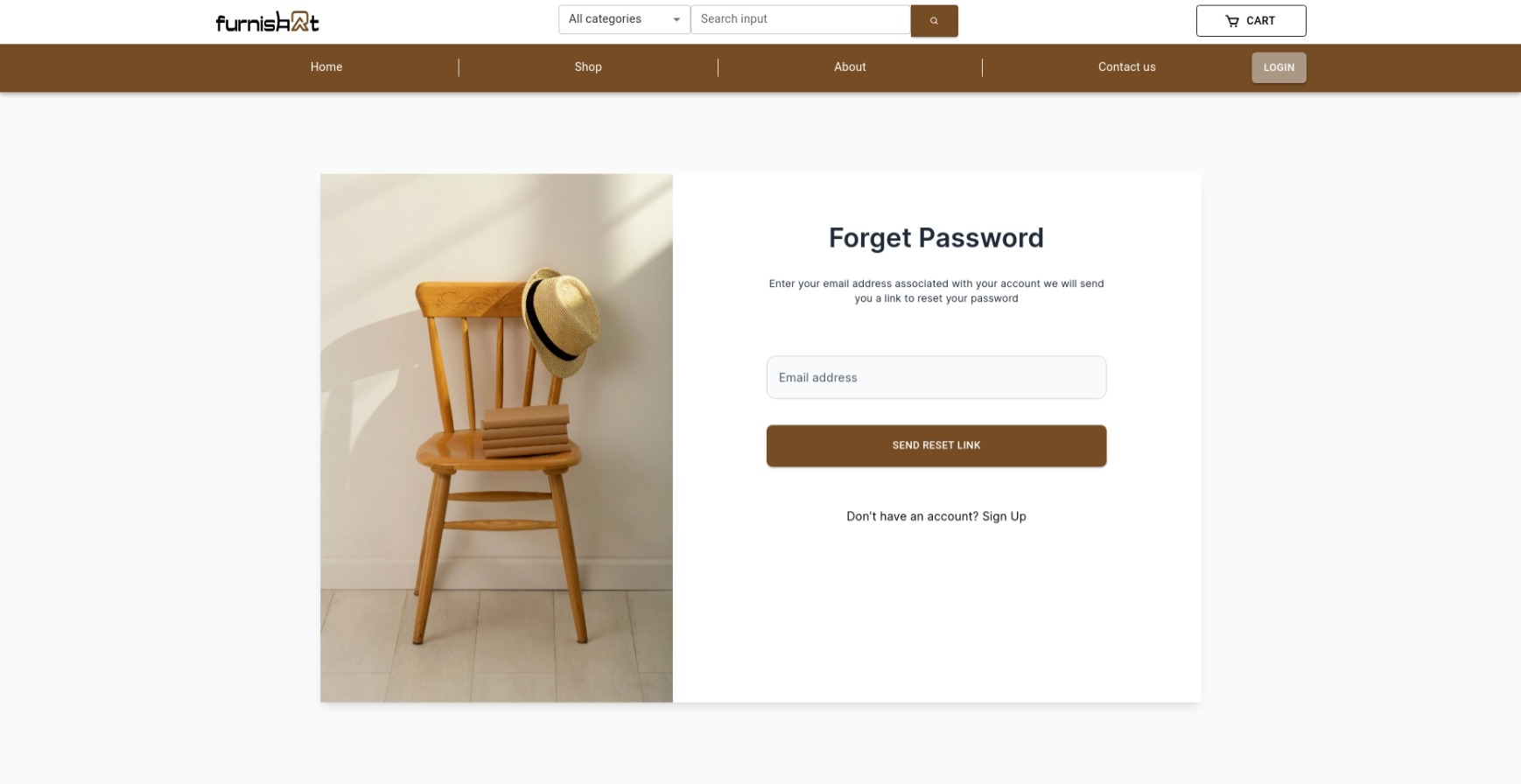
*Fig 5.2 Login Page*

**Signup Page**



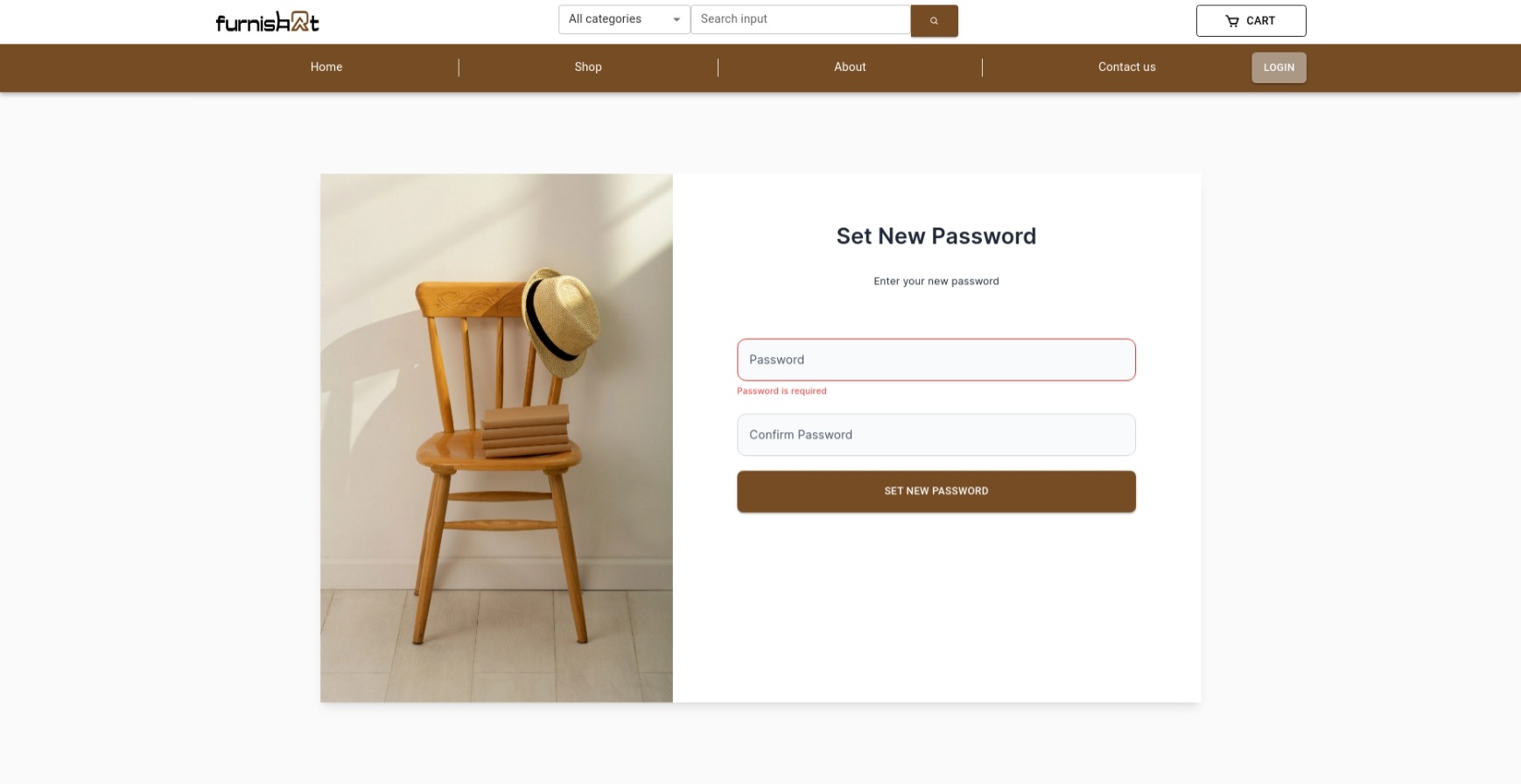
*Fig 5.3 Signup Page*

**Forget Password**



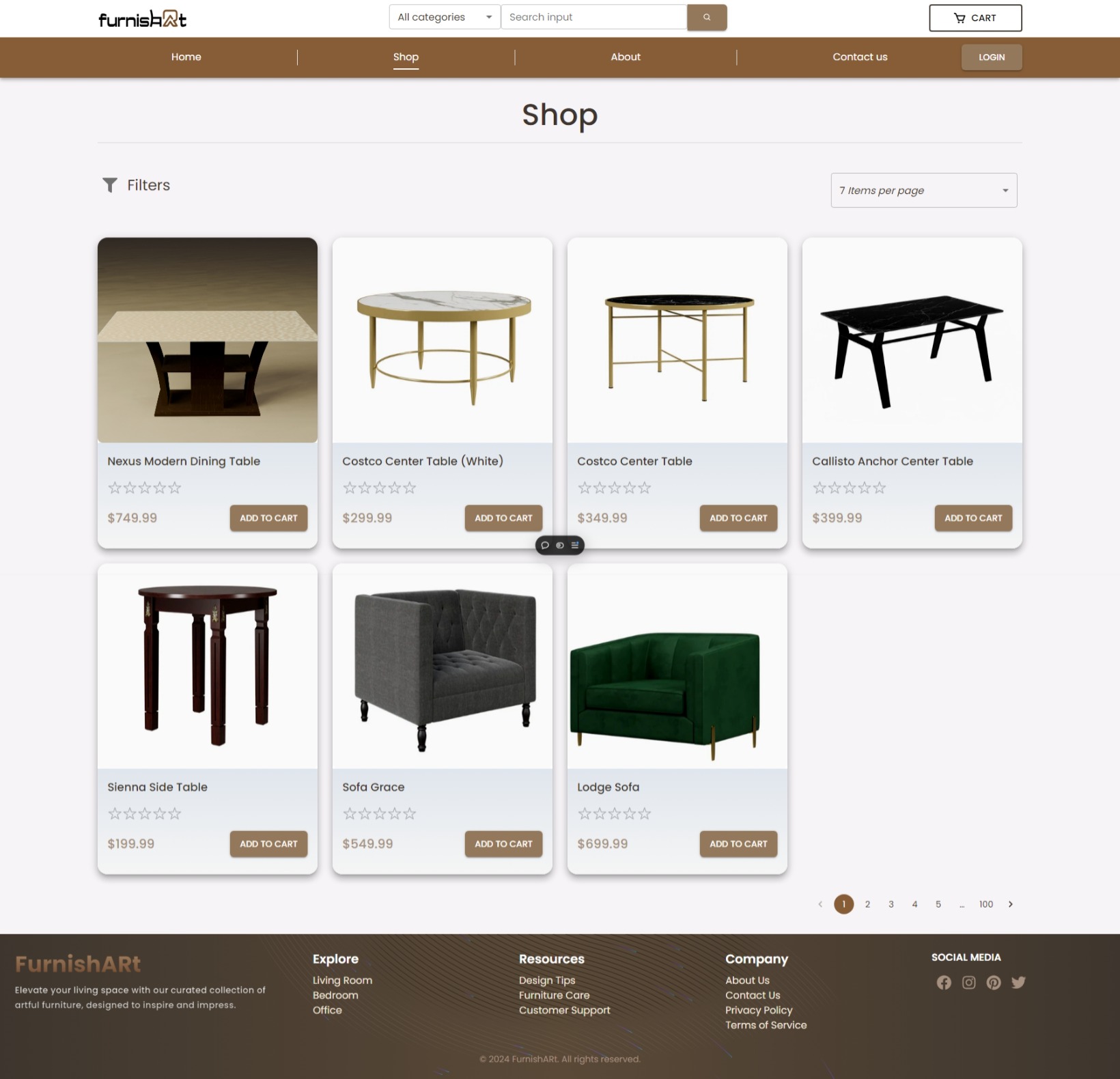
*Fig 5.4 Forget Password*

**Reset Password**



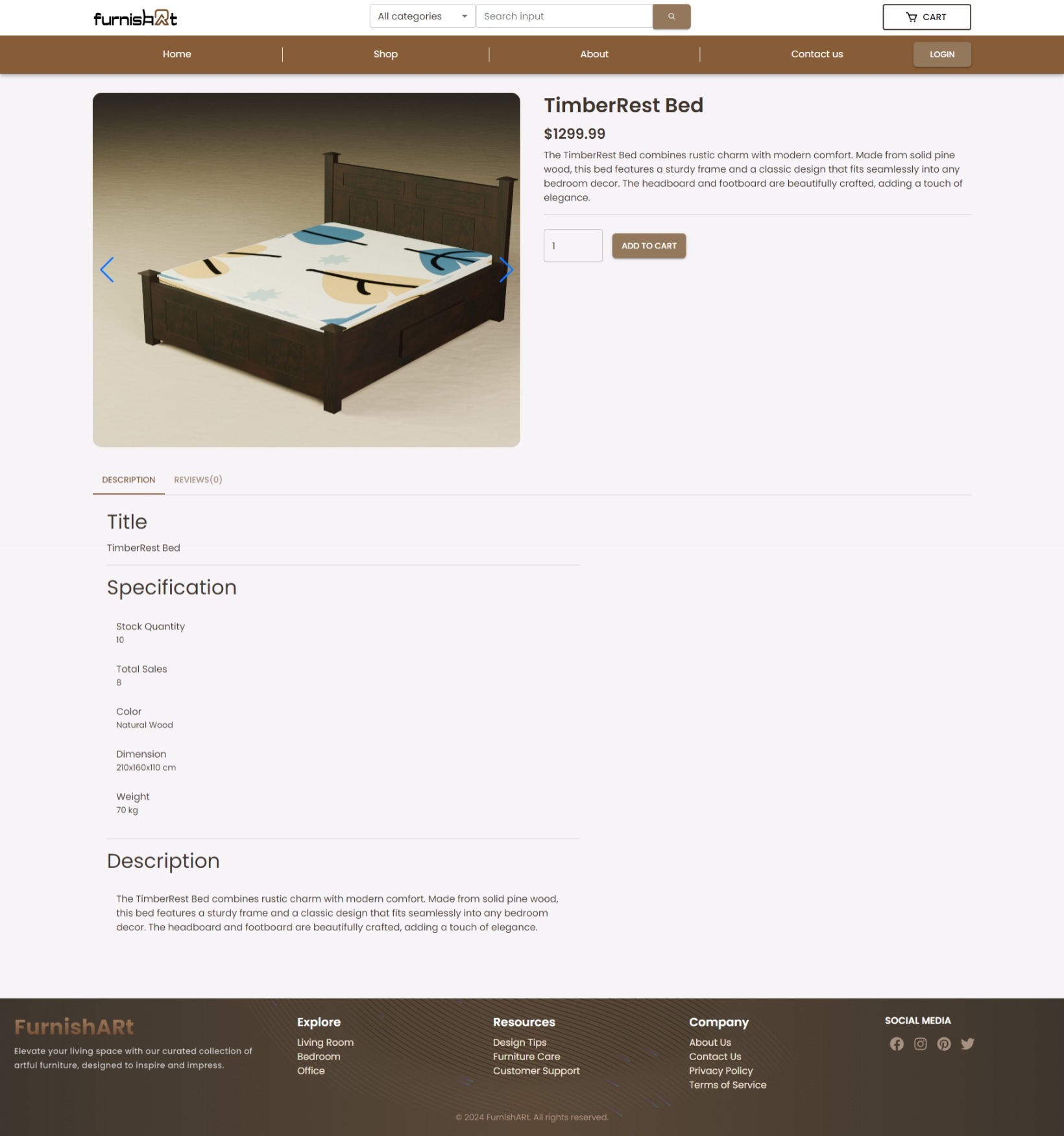
*Fig 5.5 Reset Password*

**Shop**



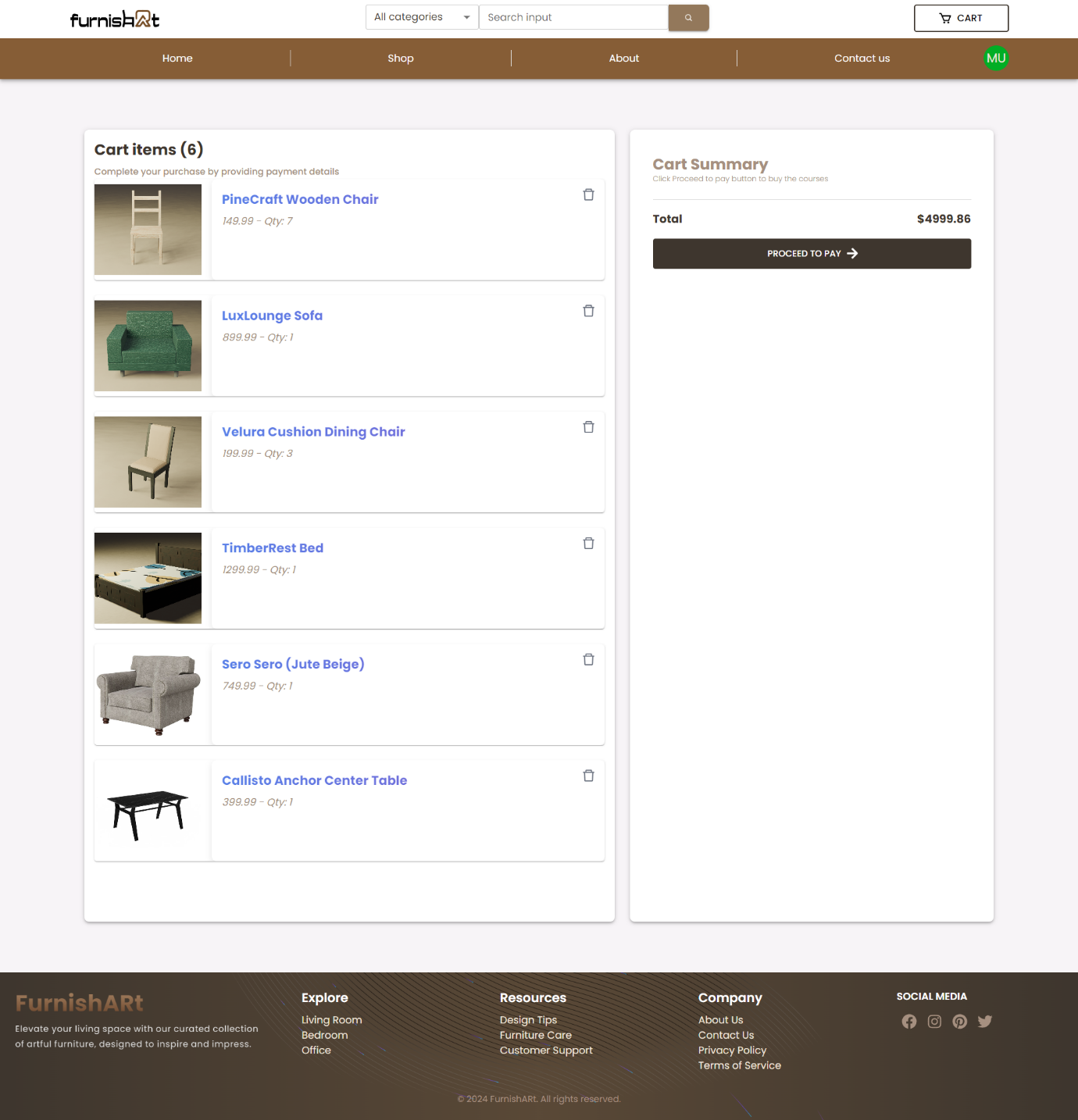
*Fig 5.6 Shop*

**Furniture Item Details Page**



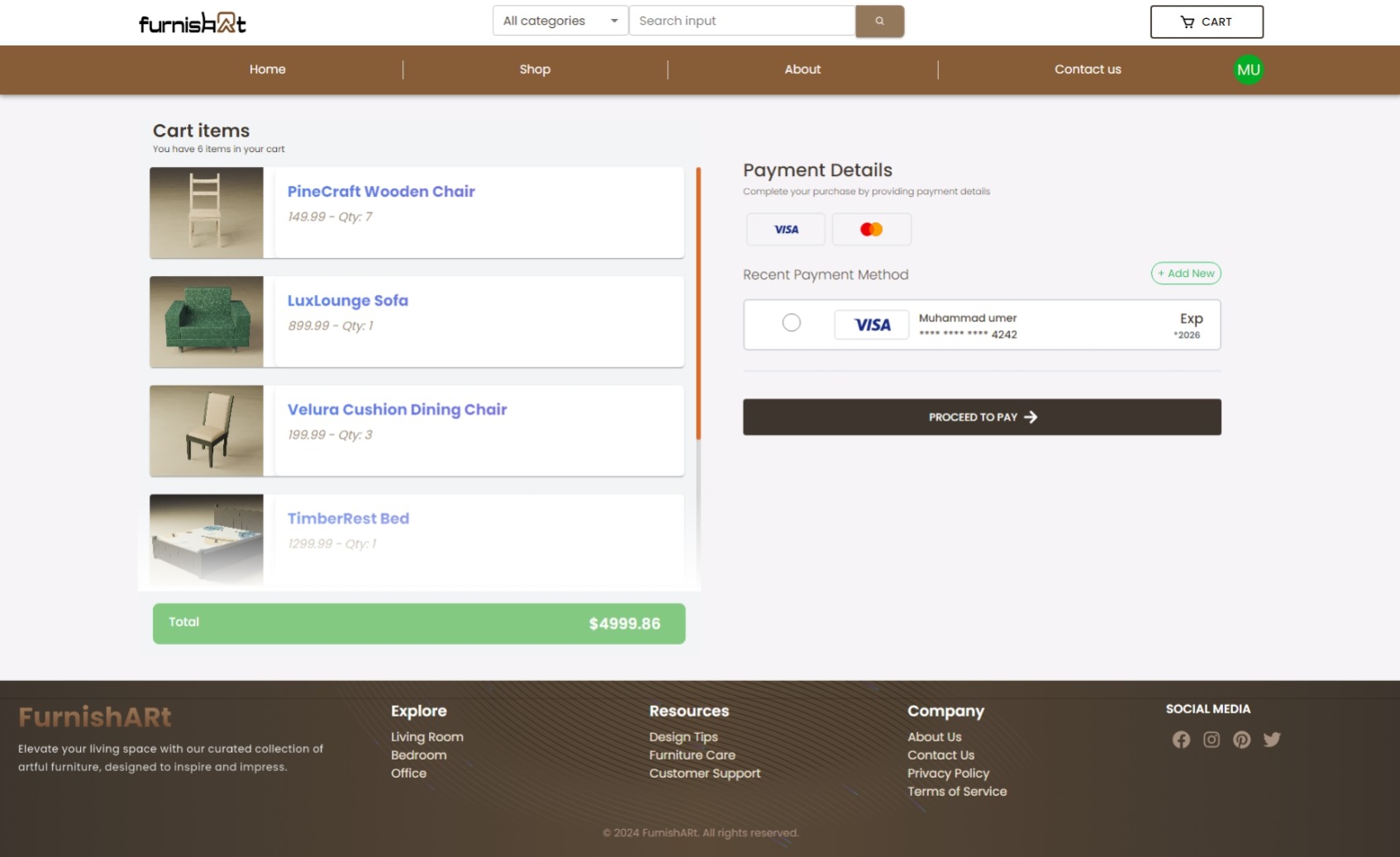
*Fig 5.7 Furniture Item Details Page*

**Cart Page**



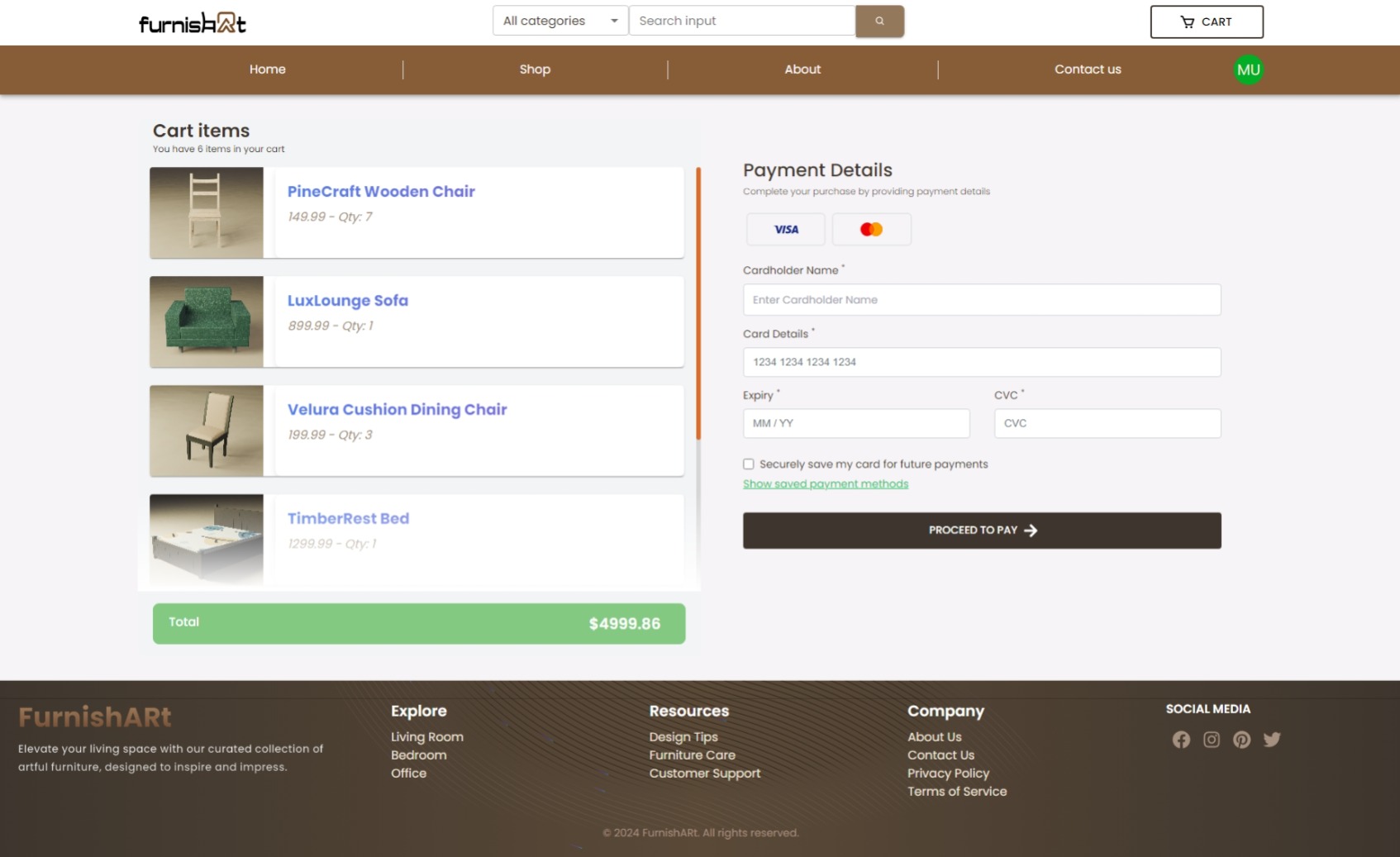
*Fig 5.8 Cart Page*

**Checkout Page (Saved Credit Cards)**



*Fig 5.9 Checkout Page (Saved Credit Cards)*

**Checkout Page (Stripe Form)**

****

*Fig 5.10 Checkout Page (Stripe Form)*

# Testing and Evaluation

Testing methodologies applied to ensure the "FurnishARt: An AR-Based Furniture Store" meets all specified functional and non-functional requirements. Both manual and automated testing approaches were employed to validate the system’s robustness, functionality, and user experience.

## Manual Testing

Manual testing was performed to verify the correct functioning of individual modules and the integration of components.

### System Testing

System testing was conducted after the development phase to ensure the application functions as intended. It included unit testing, functional testing, and integration testing. Errors identified during testing were resolved before deployment.

### Unit Testing

Unit testing verified the accuracy and behaviour of individual components of the system. Additional tests were included for modules such as Admin Management, Viewing Orders, and AR-based functionalities to ensure comprehensive coverage.

**Unit Testing 1:** **User Registration**

**Objective**: To ensure the user registration module functions correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Verify successful registration | Name: Umer  Email: omerawan445@gmail.com Password: Test1234 | User registered successfully | Pass |
| 2 | Verify error for duplicate email | Email: omerawan445@gmail.com | Error message: "Email already exists" | Pass |

*Table 6.1: User Registration Unit Testcase*

**Unit Testing 2:** **Login Functionality**

**Objective**: To ensure the login module operates as expected.

*Table 6.2: Login Functionality Unit Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Verify successful login | Email: omerawan445@gmail.com Password: Test1234 | Redirect to dashboard | Pass |
| 2 | Verify error for invalid credentials | Email: omerawan445@gmail.com Password: Test | Error message: "Invalid credentials" | Pass |

**Unit Testing 3: AR Preview**

**Objective**: To validate AR preview functionality.

*Table 6.3 AR preview Unit Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Launch AR preview | Item: Sofa | AR model displayed | Pass |
| 2 | Verify interaction | Action: Rotate, Zoom | Model responds to interaction | Pass |

**Unit Testing 4:** **Admin Management**

**Objective**: To ensure admin functionalities operate correctly.

*Table 6.4 Admin Management Unit Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Add a furniture item | Item Name: Chair  Category: Living Room | Item added successfully | Pass |
| 2 | Update a furniture item | Item Name: Chair  New Category: Office | Item updated successfully | Pass |
| 3 | Delete a furniture item | Item Name: Table | Item deleted successfully | Pass |

**Unit Testing 5:** **Viewing Orders**

**Objective**: To ensure users can view their orders.

*Table 6.5 Viewing Orders Unit Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | View order details | Order ID: 1234 | Order details displayed | Pass |
| 2 | Verify order status | Order ID: 1234 | Status: "Processing" | Pass |

### Functional Testing

Functional testing verified the functionality of individual modules against the system’s specifications.

**Functional Testing 1: Add to Cart**

**Objective**: To ensure the cart module functions properly.

*Table 6.6: Add to Cart Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Add item to cart | Item: Table, Quantity: 1 | Item added successfully | Pass |
| 2 | Verify cart total | Items: Table, Chair | Total price updated correctly | Pass |

**Functional Testing 2: Checkout Process**

**Objective**: To validate the checkout functionality, including error handling for invalid payment or missing shipping details.

*Table 6.7: Checkout Process Functional Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Enter valid shipping details | Address: 123 Main St, City: New York | Details saved successfully | Pass |
| 2 | Enter valid payment details | Card: 4111 1111 1111 1111, Expiry: 12/25 | Payment processed successfully | Pass |
| 3 | Attempt checkout with missing shipping details | Address: Empty | Display error: "Shipping details required" | Pass |
| 4 | Attempt checkout with invalid payment details | Card: 1234 5678 9012 3456 | Display error: "Invalid payment details" | Pass |

### Integration Testing

Integration testing ensured the seamless interaction between modules.

*Table 6.8: Integration Testcase*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Attribute and Value** | **Expected Result** | **Result** |
| 1 | Login and view cart | Email: user@example.com | Cart items displayed | Pass |
| 2 | Add item and proceed to checkout | Item: Chair, Address: 456 Elm St | Checkout page displayed | Pass |
| 3 | Place order and view status | Order ID: 12345 | Status: "Processing" | Pass |
| 4 | Admin adds a 3D model | Model Name: Sofa Model, File: sofa.glb | 3D model added successfully | Pass |
| 5 | Admin updates a 3D model | Model Name: Chair Model, New File: chair\_updated.glb | 3D model updated successfully | Pass |
| 6 | Verify AR preview after model upload | Model Name: Sofa Model | AR preview works with new model | Pass |

## Automated Testing

Automated testing was conducted to validate repetitive and critical functionalities.

### Tools Used

*Table 6.9: Tools used*

|  |  |  |  |
| --- | --- | --- | --- |
| **Tool Name** | **Description** | **Applied On** | **Results** |
| Selenium | Automated browser testing | Login, Cart, Checkout | Pass |
| Jest | JavaScript testing framework | Unit tests for Login, AR Preview | Pass |

### Automated Test Scenarios

*Table 6.10: Automated Test Scenarios*

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Scenario** | **Expected Result** | **Result** |
| 1 | Verify login with Selenium | Login successful | Pass |
| 2 | Verify cart updates with Jest | Cart updated correctly | Pass |
| 3 | Verify AR preview loads | AR preview displayed | Pass |
| 4 | Admin adds a furniture listing | Listing added successfully | Pass |
| 5 | Admin updates a furniture listing | Listing updated successfully | Pass |

# Conclusion and Future Work

## Conclusion

FurnishARt, an Augmented Reality-based furniture store, has successfully demonstrated the potential of AR technology in enhancing the online shopping experience. By addressing the limitations of traditional e-commerce platforms, such as inadequate product visualization and the lack of spatial context, FurnishARt has introduced innovative features like 3D model interactions and real-time AR previews. The system's user-friendly design, robust backend, and seamless integration of AR functionality have proven effective in bridging the gap between digital and physical shopping experiences.

Through the iterative development process, multiple challenges were tackled, including optimizing AR performance, ensuring cross-platform compatibility, and designing an intuitive interface. These efforts resulted in a platform that not only meets user expectations but also sets a benchmark for future AR-based e-commerce solutions. The project validates the significance of combining emerging technologies with a user-centric approach to revolutionize industries like furniture retail.

## Future Work

While FurnishARt has achieved its core objectives, there are several areas for future development to enhance its scope and impact:

* **Advanced AR Features:** Integration of features like multi-item AR visualization, dynamic lighting adjustments, and customizable furniture options to enrich the AR experience further.
* **AI-Powered Recommendations:** Implementation of machine learning algorithms to provide personalized furniture suggestions based on user preferences and browsing history.
* **Global Reach:** Expanding the platform to support multiple languages and currencies to cater to a broader, international audience.
* **Virtual Reality (VR) Integration:** Introducing VR functionalities to allow users to virtually explore entire room setups or furniture showrooms.
* **Sustainability Metrics:** Adding features that highlight eco-friendly products and materials, helping users make informed and sustainable purchasing decisions.
* **Enhanced Analytics:** Deploying advanced analytics tools for real-time insights into user behaviour, aiding in decision-making for both users and administrators.

These advancements will ensure FurnishARt remains at the forefront of technological innovation, continuously improving user satisfaction and business outcomes.

# References

1. **Glenford J. Myers, Corey Sandler, and Tom Badgett.** (2011). The Art of Software Testing. Wiley. Focused on software testing principles and strategies, essential for the testing chapter.
2. **Kent Beck and Cynthia Andres.** (2004). Extreme Programming Explained: Embrace Change. Addison-Wesley. Explains Agile methodology, which aligns with the iterative development approach used in this project.
3. **Alan Dennis, Barbara Haley Wixom, and Roberta M. Roth.** (2018). Systems Analysis and Design. Wiley. Covered key design principles applicable to project architecture and system integration.
4. **Ali Behforooz and Frederick J. Hudson.** (1996). Software Engineering Fundamentals. Oxford University Press. Chapter 8, pp255-235. Provided foundational insights into software engineering practices applied to this project.
5. **Nielsen, Jakob.** (1993). Usability Engineering. Academic Press. Insights into user-centric design principles relevant to the project’s UI/UX components.
6. **Reitmayr, G., & Drummond, T. (2006).** Going Out: Robust Model-Based Tracking for Outdoor Augmented Reality. In Proceedings of the 2006 IEEE/ACM International Symposium on Mixed and Augmented Reality.
7. **Dünser, A., Grasset, R., & Schmalstieg, D. (2011).** Augmented Reality and Game Development. CRC Press.