VIETNAM INTERNATIONAL UNIVERSITY – HO CHI MINH CITY

INTERNATIONAL UNIVERSITY

**WEB APPLICATION DEVELOPMENT PROJECT**

**FURNITURE SALES WEBSITE**

BY

Trịnh Quang Anh – Student ID: ITCSIU21164

Nguyễn Thành Khiêm – Student ID: ITCSIU21137

Advisor: Dr. Nguyen Van Sinh

A report submitted to the School of Computer Science and Engineering

in partial fulfillment of the requirements for the Final Project

in Web Application Development course - Fall 2024

Ho Chi Minh city, Vietnam, 2024

**TABLE OF CONTENTS**

1. **INTRODUCTION**

With a growing population and increasing population density, the demand for furniture also increases accordingly. Especially in big cities, the need to "move from house to apartment" is increasing. Besides, online shopping is gradually replacing buying directly from stores. From there, creating furniture buying and selling websites is necessary for the market. Furniture Sale Website was born to meet consumer tastes.

In this project, our team used HTML, CSS, JavaScript to create the interface, and used Django to manage the BackEnd

After implementing the project, the team achieved results such as: building a simple interface for the Furniture Sale Website, the function of adding products to the shopping cart, the function of registering and logging in for users, and the search function. search for products.

1. **SYSTEM DIAGRAM**

The HomeClick website system includes the functions listed in the image below:

A diagram of a product

Description automatically generated

Figure 1: System Diagram

1. **DETAIL DESCRIPTION OF FUNCTIONS**
2. Product Management

* Add New Product: administrator adds new product to the system.
* Delete Product: delete the product from the system.
* Update Product Information: update existing product information.
* Product Search: users search for products according to different criteria.

1. Order Management

* Add New Order: user creates new order when purchasing, new order is created when user clicks on the "Add to cart" button for each product.
* Update Order: admin updates order status.
* Delete order: administrator deletes order from the system.

1. User Management

* Registration: user creates a new account, user account information is saved in the database.
* Log in: if you already have an account, you must log in to make a purchase.
* Update address, update phone number: on the User Profile page, it allows updating the user's address and phone number to save delivery information.
* Delete customer: administrator deletes customer account in database.

1. Cart Management

* Add products, reduce products: users are allowed to increase or decrease products in the shopping cart. If the number of products decreases to 0, the product will be removed from the shopping cart.
* Count the number of products in the cart: based on the number of products in the cart and display the number of products.
* Calculate the total amount of the order: calculate the total amount of all remaining products in the shopping cart.

1. **DEVELOPMENT PROCESS**
2. Requirement Analysis

* Gather requirements from stakeholders, including product categories, target audience, and website functionality.
* Document functional and non-functional requirements.
* Define project scope, objectives, and deliverables.

1. Planning

* Create a project roadmap and timeline.
* Allocate resources, including team members and tools.
* Define milestones and key deliverables.
* Prepare a risk management plan.

1. Design Phase

* UI/UX Design:
* Develop wireframes and mockups for the website.
* Ensure responsive design for desktop, tablet, and mobile devices.
* Incorporate branding guidelines and user-friendly navigation.
* System Architecture Design:
* Design the database schema (e.g., products, users, orders).
* Define the architecture (e.g., client-server model, APIs).

1. Development Phase

* Frontend Development
* Implement responsive design using HTML, CSS, and JavaScript.
* Build components for product display, search filters, shopping cart, and checkout.
* Ensure seamless user interaction through dynamic features.
* Backend Development
* Set up the server and database
* Develop APIs for product management, user authentication, and order processing.
* Integrate a secure payment gateway

1. Testing Phase

* Unit Testing: Test individual components (e.g., product listing, checkout process).
* Integration Testing: Ensure seamless communication between frontend, backend, and external APIs.
* System Testing: Validate the complete system functionality.
* User Acceptance Testing (UAT): Collect feedback from end-users to ensure the website meets expectations.
* Performance Testing: Ensure the website can handle high traffic and large data loads.

1. Development Phase

* Prepare the production environment.
* Deploy the website to a live server
* Conduct a final round of testing on the live site.

1. **DATABASE**

The image below is the database used to build the website

A screenshot of a computer

Description automatically generated

Figure 2: Database of Furniture Sales Website

* Customer table contains personal information of customers. This information includes the customer's name, email, phone number and address.
* Product table stores information about available products on the website. Product information includes name, price, product code, type and image.
* Order table stores information about customer orders, such as order code, ordering customer, order date, completion status and transaction code.
* OrderItem table links between the Order table and the Product table, containing information about the products in each order, including quantity and date added to cart.
* ShippingAddress table stores information about the customer's shipping address, including detailed address, city, state, contact phone number, and date the address was added.
* Article table contains information about articles, including name, posting date, and image.

1. **REQUIREMENT ANALYSIS AND DESIGN**

This section briefly introduces the requirement analysis and design process. This version is created to provide the path for future implementation of the project. Based on this requirements specification, we will implement each function including all of the conditions as well as functional and non-functional requirements supplied by the customers. During our implementation, we will constantly revise and update the newer version so that we can always keep track of the progress of the project.

1. **REQUIREMENT ANALYSIS**

**Use Case Diagram**

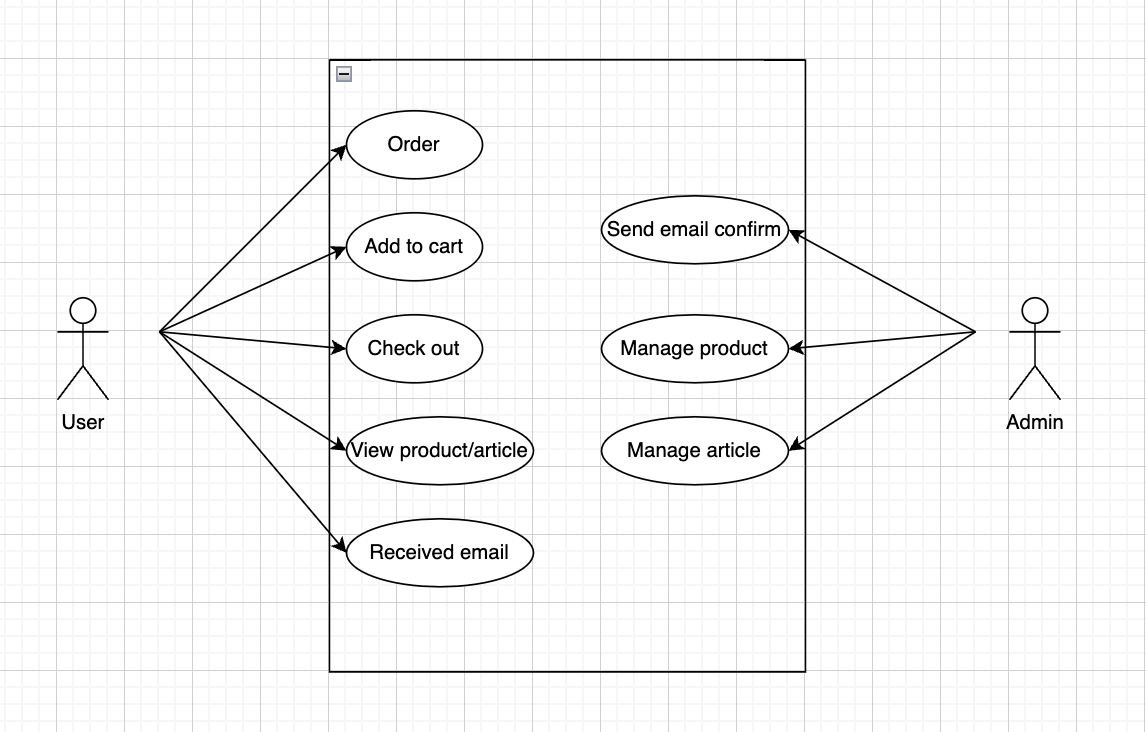


Figure 3: Use Case Diagram

**Use Case 1:**

Name: Order

Identifier UC1

Inputs:

* Product information
* User information
* Delivery preferences
* Payment method

Outputs:

* Order confirmation
* Payment receipt
* Notification email
* Updated stock levels in the database
* Order details saved in the database

Basic Course

|  |  |
| --- | --- |
| Actor: User | System |
| 1. Browses catalog and selects products | * 1. Displays product catalog with availability |
| 1. Adds products to the cart | * 1. Updates cart dynamically |
| 1. Views the cart | * 1. Displays selected items, quantities, and total price |
| 1. Proceeds to checkout | * 1. Validates delivery information and payment method |
| 1. Provides payment details | * 1. Processes payment via payment gateway and updates order status |
| 1. Confirms the order | * 1. Sends confirmation email, updates database, and displays order summary |

Pre-conditions:

* The user is logged in
* The selected products are available in stock.
* Valid payment details are provided.

Post-conditions:

* The order is saved in the database with the status "Pending" or "Confirmed."
* The user receives an order confirmation email or notification.
* The stock levels for the ordered products are updated in the database.
* Payment is processed and logged.

User story: As a customer, I want to be able to select furniture products and place an order easily, so that I can purchase the items I need and receive a confirmation of my transaction.

**Use Case 2:**

Name: Add to cart

Identifier UC2

Inputs:

* Product information
* User information

Outputs:

* Cart Update
* Confirmation Message
* Database Update
* Stock Validation

Basic Course:

|  |  |
| --- | --- |
| Actor: User | System |
| 1. Selects a product and clicks "Add to Cart." | * 1. Validates the product ID and retrieves its details (name, price, stock). |
| 1. Specifies product quantity and options Checks if the selected quantity is available in stock. | * 1. Checks if the selected quantity is available in stock. |
| 1. Confirms the addition | * 1. Updates the cart in the session or database with the new product details. |
| 1. Views the updated cart | * 1. Returns the updated cart information. |
| 1. Sees a success message | * 1. Displays a confirmation message to the actor: "Product added to cart successfully." |

Pre-conditions:

* The product exists in the database and is available for sale.
* The system is able to validate stock levels for the requested quantity.
* The user is browsing the website (logged in or as a guest).

Post-conditions:

* The cart is updated with the selected product and quantity.
* Stock levels are not deducted yet (deduction happens at checkout).
* The system reflects the updated cart for the user’s session.

User story: As a customer, I want to add products to my cart while browsing the website, so that I can review them later and proceed to checkout when I’m ready.

**Use Case 3:**

Name: Check out

Identifier UC3

Inputs:

* User information
* Cart Details
* Payment Information
* Delivery Preferences

Outpus:

* Order information
* Payment receipt
* Update database
* Notification

Basic Course:

|  |  |
| --- | --- |
| Actor: User | System |
| 1. Proceeds to checkout | * 1. Displays the checkout page with cart summary and total price. |
| 1. Enters delivery information | * 1. Validates the delivery details (name, address, phone number). |
| 1. Selects payment method | * 1. Displays payment options and validates the selection. |
| 1. Enters payment information | * 1. Processes the payment through the payment gateway |
| 1. Confirms the order | * 1. Validates stock availability and saves the order details in the database. |
| 1. Receives order confirmation | * 1. Sends a confirmation email and displays the order confirmation on the screen. |

Pre-conditions:

* The user has items in their cart.
* The selected products are available in stock.
* Payment details are valid and payment gateway is functional.

Post-conditions:

* The order is saved in the database with a unique order ID.
* The stock levels for the ordered products are updated.
* A confirmation email is sent to the user.
* The payment is logged.

User story: As a customer, I want to securely and efficiently complete my purchase through a checkout process, so that I can place my order and receive confirmation with delivery details.

**Use Case 4:**

Name: View product/article

Identifier UC4

Inputs:

* User Actions
* Query Parameters

Outputs:

* Product/Article Details
* Related Items
* Page Display

Basic Course:

|  |  |
| --- | --- |
| Actor: User | System |
| 1. Selects a product/article to view | * 1. Receives request and identifies the product/article ID. |
| 1. Waits for data to load | * 1. Queries the database for product/article details based on the ID. |
| 1. Views the displayed details | * 1. Returns the product/article details to the UI. |
| 1. Views related items | * 1. Optionally retrieves and displays related products/articles. |
| 1. Interacts with the displayed content | * 1. Logs user interaction for analytics. |

Pre-conditions:

* The product or article exists in the database.
* The user has access to view the product/article (e.g., not restricted or hidden content).
* The system has the ability to retrieve and display the requested details.

Post-conditions:

* The product/article details are displayed to the user.
* Any user interactions (e.g., clicks on related items) are logged for analytics (optional).
* Suggested items (products or articles) are shown based on relevance.

User story: As a customer, I want to view the details of a product or article, so that I can learn more about it and decide whether to purchase or read further.

**Use Case 5:**

Name: Received email

Identifier UC5

Inputs:

* Order Details
* System Information

Outputs:

* Email Sent
* Database Log
* User Notification

Basic Course:

|  |  |
| --- | --- |
| Actor: User | System |
| 1. Completes the order | * 1. Triggers the email sending process. |
| 1. Prepares email content | * 1. Retrieves email template and fills it with the order details. |
| 1. Sends email to the customer | * 1. Uses the configured email service (e.g., SMTP, SendGrid) to send the email. |
| 1. Logs the email status | * 1. Records the email delivery result (success/failure) in the database. |
| 1. Displays confirmation | * 1. Optionally displays a message to the customer that the email has been sent successfully. |

Pre-conditions:

* The order is successfully placed and recorded in the database.
* The customer’s email address is valid and present.
* The email service is configured and operational.

Post-conditions:

* The customer receives the email with order details.
* The email status (success or failure) is logged in the database.
* Any failure in sending the email triggers a retry mechanism or error handling process.

User story: As a customer, I want to receive a confirmation email after placing an order, so that I can have a record of my purchase and its details.

**Use Case 6:**

Name: Send email confirm

Identifier UC6

Inputs:

* Order Details
* System Information
* Trigger

Outputs:

* Email Confirmation
* Database Update
* User Notification

Basic Course:

|  |  |
| --- | --- |
| Actor: Admin | System |
| 1. Confirms the order | * 1. Detects the change in order status to "Confirmed". |
| 1. Prepares email content | * 1. Retrieves the email template and dynamically populates it with order details. |
| 1. Sends email to the customer | * 1. Sends the email using the configured email service. |
| 1. Logs the email status | * 1. Records the email delivery result (success or failure) in the database. |
| 1. Displays confirmation | * 1. Displays a success message to the admin or triggers a retry mechanism in case of failure. |

Pre-conditions:

* The order is successfully confirmed and stored in the database.
* The customer’s email address is valid and present.
* The email service is properly configured and operational.

Post-conditions:

* The confirmation email is sent to the customer.
* The email status (success or failure) is logged in the database.
* A retry or error handling mechanism is initiated if the email fails to send.

User story: As a customer, I want to receive an email confirming my order, so that I can review the details of my purchase and know what to expect for delivery.

**Use Case 7:**

Name: Manage product

Identifier UC7

Inputs:

* Admin inputs
* System inputs

Outputs:

* System Updates
* UI Feedback
* Database Update

Basic Course:

|  |  |
| --- | --- |
| Actor: Admin | System |
| 1. Admin selects an action (add/update/delete) | * 1. Displays the appropriate form or interface for the action |
| 1. Admin enters/edits product details | * 1. Validates the input data |
| 1. Admin confirms the action | * 1. Sends the request (add/update/delete) to the backend |
| 1. System processes the request | * 1. Interacts with the database to add, update, or delete the product |
| 1. System updates the catalog | * 1. Updates the product list and logs the changes |
| 1. Admin sees the result | * 1. Displays a success or error message to the admin |

Pre-conditions:

* Admin is authenticated and authorized to manage products.
* The system has a functional product database.
* The required input fields (e.g., product name, price) are available and correctly formatted.

Post-conditions:

* The product catalog is updated (added, edited, or deleted).
* A success or error message is displayed to the admin.
* The change is logged in the system for audit purposes.

User story: As an admin, I want to manage the product catalog by adding, updating, or deleting products, so that I can ensure the website displays accurate and up-to-date product information.

**Use Case 8:**

Name: Manage article

Identifier UC8

Inputs:

* Admin inputs
* System inputs

Outputs:

* System updates
* UI Feedback
* Database Update

Basic Course:

|  |  |
| --- | --- |
| Actor: Admin | System |
| 1. Admin selects an action (add/update/delete) | * 1. Displays the appropriate form or interface for managing articles. |
| 1. Admin enters/edits article details | * 1. Validates the input data (e.g., required fields, publication date format). |
| 1. Admin confirms the action | * 1. Sends the request (add/update/delete) to the backend. |
| 1. System processes the request | * 1. Interacts with the database to add, update, or delete the article. |
| 1. System updates the article list | * 1. Updates the article catalog and logs the changes. |
| 1. Admin sees the result | * 1. Displays a success or error message to the admin. |

Pre-conditions:

* Admin is authenticated and authorized to manage articles.
* The system has a functional article database.
* The required input fields (e.g., title, content) are available and correctly formatted.

Post-conditions:

* The article catalog is updated (added, edited, or deleted).
* A success or error message is displayed to the admin.
* The change is logged in the system for tracking purposes.

User story: As an admin, I want to manage articles by adding, updating, or deleting them, so that I can ensure the website displays relevant and up-to-date content.

1. **FUNCTIONAL REQUIREMENTS**

**Use Case 1: Order**

1. **The scope of the work:** The scope of the work involves implementing the functionality for customers to place orders on a furniture sales website. It covers the entire process from selecting products in the cart, entering delivery and payment details, and confirming the order, to saving the order information in the database and sending a confirmation email to the customer.
2. **The scope of the product:**

* Allowing users to browse and select furniture products.
* Managing the cart functionality to store selected products.
* Providing a secure and user-friendly checkout process.
* Handling payment processing through supported gateways.
* Sending confirmation emails with order details and delivery information.
* Maintaining accurate stock updates in the inventory.
* Storing all order information in a database for tracking and analytics.

1. **Functional and Data Requirements:**

* **Funtional Requirements:**
* Product Selection
* Cart Management
* Checkout Process
* Order Placement
* Order Confirmation
* Inventory Management
* Error Handling
* **Data Requirements**:
* Customer Data
* Product Data
* Cart Data
* Order Data
* Payment Data
* Inventory Data
* Notification Data

**Use Case 2: Add to cart**

1. **The scope of the work:**

* Frontend: Develop a user-friendly interface for adding products to the cart.
* Backend: Implement logic for cart management, inventory validation, and updating cart data in the database.
* Testing: Ensure the functionality works as intended on all supported devices and browsers.

1. **The scope of the product:**

* Select a product from the product listing or detail page.
* Specify the desired quantity.
* Add the selected product to their shopping cart.
* View an updated cart summary showing the product, quantity, and total price.
* Inventory validation (products cannot be added if out of stock or beyond available quantity).
* Cart data is persisted for logged-in users and stored temporarily for guest users.

1. **Functional and Data Requirements:**

* **Funtional Requirements:**
* Users can select a product and specify a quantity.
* Users can click an "Add to Cart" button to add the product to their cart.
* Validate that the selected quantity is available in stock.
* Update the user's cart with the selected product and quantity.
* Recalculate and display the cart's total price.
* Provide confirmation that the product has been added successfully.
* **Data Requirements:**
* Inputs:
* Product ID: Unique identifier for the product being added.
* Quantity: Desired quantity of the product.
* User ID: Identifier for the logged-in user
* Outputs:
* Cart Data
* Cart Summary: Updated total price of the cart.
* Confirmation Message: Message to notify the user that the product was added successfully.
* Validation Rules:
* Verify product availability in inventory.
* Ensure the selected quantity is valid (e.g., greater than 0 and does not exceed available stock).

**Use Case 3: : Check out**

1. **The scope of the work:** The scope of the work involves implementing the functionality required for customers to complete the purchase of their selected items. This includes gathering customer information, handling payment processing, validating order details, updating stock and order records, and providing order confirmation to the customer.
2. **The scope of the product:**

* Facilitating a user-friendly and secure checkout process.
* Validating customer-provided details (e.g., address, contact information).
* Offering various payment options for convenience.
* Confirming the order details and ensuring stock availability.
* Processing payments securely through a payment gateway.
* Saving the order in the database for record-keeping and fulfillment.
* Sending a confirmation email with order details and tracking information.

1. **Functional and Data Requirements:**

* **Functional Requirements:**
* Cart validation
* Customer Information
* Shipping Details
* Payment Handling
* Order Confirmation
* **Data Requirements:**
* Cart data
* Customer data
* Shipping data
* Order data
* Payment data

**Use Case 4: View product/article**

1. **The scope of the work:** The scope of the work involves implementing the functionality for customers or visitors to view detailed information about products or articles on the furniture sales website. This includes retrieving relevant details from the database, presenting them on the UI, and optionally suggesting related products or articles.
2. **The scope of the product:**

* Enabling users to browse and select products or articles.
* Displaying detailed information about a product (e.g., name, price, description) or an article (e.g., title, content, author).
* Allowing users to search and filter products or articles.
* Suggesting related products or articles based on relevance.
* Logging user interactions for analytics (optional).

1. **Functional and Data Requirements:**

* **Functional Requirements:**
* Product/Article Retrieval
* Detailed Display
* Search and Filter
* Related Items
* Error Handling
* **Data Requirements:**
* Product data
* Article data
* Search and filter data
* Related items data

**Use Case 5: Received email**

1. **The scope of the work:** The scope of the work involves implementing the functionality to send order-related confirmation emails to customers after they place an order. This includes triggering the email process upon successful order placement, generating the email content, and delivering it to the customer’s email address.
2. **The scope of the product:**

* Automatically sending a confirmation email after an order is placed.
* Including all relevant order details in the email, such as order ID, items purchased, total cost, and estimated delivery date.
* Ensuring email delivery using a reliable email
* Logging email delivery status (success or failure) for tracking and troubleshooting purposes.

1. **Functional and Data Requirements:**

* **Functional Requirements:**
* Trigger Email Sending
* Generate Email Content
* Send email
* Log email delivery status
* **Data Requirements:**
* Order data
* Email template data
* Delivery status data

**Use Case 6: Send email confirm**

1. **The scope of the work:** The scope of the work involves implementing a feature that automatically sends a confirmation email to the customer after a successful order placement. This includes generating the email content with order details, ensuring proper delivery to the customer's email address, and logging the delivery status for tracking and troubleshooting.
2. **The scope of the product:**

* Automatically triggering the email-sending process when an order is placed.
* Generating personalized email content with relevant order details.
* Ensuring the email is delivered securely and reliably using an email service provider.
* Logging the delivery status (success or failure) in the database for monitoring purposes.
* Allowing for error handling and retry mechanisms in case of failures.

1. **Functional and Data Requirements:**

* **Functional Requirements:**
* Triggering Email Process
* Generate Email Content
* Send Confirmation Email
* Log Email Status
* **Data Requirements:**
* Order data
* Email template data
* Delivery status data

**Use Case 7: Manage product**

1. **The scope of the work:** The scope of the work involves enabling admins to manage the product catalog on the furniture sales website. This includes adding, updating, and deleting product records through an admin interface and ensuring that the database reflects the changes accurately and securely.
2. **The scope of the product:**

* Providing an admin dashboard or API for managing products.
* Allowing admins to add new products with all relevant details.
* Enabling admins to update product information, such as price, description, and stock quantity.
* Facilitating the removal of products from the catalog.
* Ensuring all changes are logged for audit purposes.
* Maintaining data integrity and consistency in the product catalog.

1. **Functional and Data Requirements:**

* **Functional Requirments:**
* Add product
* Update product
* Delete product
* Product Validation
* Search and List Products
* Error Handling
* **Data Requirements**
* Product data
* Admin input data
* Database logs
* Validation data

**Use Case 8: Manage article**

1. **The scope of the work:** The scope of the work involves enabling admins to manage articles on the furniture sales website. This includes creating, updating, and deleting articles through an admin interface and ensuring all changes are reflected in the database and website accurately.
2. **The scope of the product:**

* Providing an admin dashboard or API for managing articles.
* Allowing admins to create new articles with all relevant details (title, content, author, images, etc.).
* Enabling admins to update existing articles, including modifying content, metadata, and attachments.
* Allowing admins to delete articles from the system.
* Ensuring all changes are logged for audit purposes.
* Maintaining data integrity and providing search and filtering options for article management.

1. **Functional and Data Requirements:**

* **Functional Requirements:**
* Add article
* Update article
* Delete article
* List and Search Articles
* Article Validation
* Error Handling
* **Data Requirements:**
* Article Data
* Admin Input Data
* Database Logs
* Validation Data

1. **NON-FUNCTIONAL REQUIREMENTS**
2. **Performance**

* The system must ensure fast response times when users perform order update actions.
* Page load times should provide a smooth user experience.
* The system must handle a large number of concurrent users without interruptions.

1. **Scalability**

* The system should be capable of scaling flexibly to meet increasing user and order demands in the future.
* The database must support storing and processing a large volume of data without impacting overall performance.

1. **Reliability**

* The system must operate stably and continuously, minimizing downtime.
* Data integrity must be ensured, and no data should be lost during order updates.
* The system should be able to recover data efficiently in case of failures.

1. **Usability**

* The interface must be user-friendly, allowing users to update orders easily and intuitively.
* The website must work well across different devices and browsers.
* Clear notifications and instructions must be displayed to guide users when errors or issues occur.

1. **Security**

* User and order data must be securely stored and transmitted.
* Only authenticated users should be allowed to update orders.
* Proper access controls must be in place to prevent unauthorized changes or misuse of data.

1. **Maintainability**

* The system must be designed to be flexible and allow for easy upgrades or feature expansions.
* The codebase must be clear, well-documented, and maintainable for future developers.
* The system must support error logging and monitoring for quick issue resolution.

1. **Compatibility**

* The website must function smoothly on various web browsers and devices.
* The system should be compatible with external tools and services, such as payment gateways or notification services.

1. **Design**

**Entiry-Relationship Diagram (ERD)**

**A diagram of a company

Description automatically generated**

**Figure 4: ERD**

**Use Case 1: Order and check out**

A diagram of a process

Description automatically generated

**Use Case 2: Add to cart**

A diagram of a diagram

Description automatically generated

**Use Case 3: View product/article**

A diagram of a diagram

Description automatically generated

**Use Case 4: Received email**

A diagram of a diagram

Description automatically generated

**Use Case 5: Send email confirm**

A diagram of a diagram

Description automatically generated

**Use Case 6: Manage product/article**

A diagram of a process

Description automatically generated

1. **IMPLEMENTATION**

A screenshot of a computer

Description automatically generated

**Figure 5: Login page**

A screenshot of a computer

Description automatically generated

**Figure 6: User Interface**

A screenshot of a website

Description automatically generated

**Figure 7: Search Page Interface**

A screenshot of a computer

Description automatically generated

**Figure 8: Cart Interface**

**A screenshot of a computer

Description automatically generated**

**Figure 9: Homepage**

1. **DISCUSSION AND CONCLUSION**

* During the implementation of the project, learning, understanding, and applying new technologies to the project was a challenge for the team.
* However, thanks to the efforts and hard work of each team member, we achieved several results, such as: building a basic interface for the Furniture Sales Website using HTML and CSS, optimizing website usability across multiple devices with the Bootstrap 5 framework, setting up a basic database for an e-commerce website, and using Django to implement website functionalities.
* The most satisfying feature is the shopping cart management, where users can easily add, update, and delete products. Additionally, displaying the product quantity and total price enhances user convenience.
* Despite these achievements, the team still has many shortcomings. For example, we were unable to apply Large Language Models (LLM) to the search functionality, did not allow users to apply discount codes, and some interface pages remain unfinished due to time constraints.
* The project has helped the team gain a deeper understanding of the process of developing an online sales website using Django and related technologies.

1. **REFERENCES**
2. How to Build a Large Language Model from Scratch Using Python. Link: <https://www.freecodecamp.org/news/how-to-build-a-large-language-model-from-scratch-using-python/>
3. BootStrap5. Link: <https://getbootstrap.com/docs/5.3/getting-started/introduction/>
4. What is the use of {% load static %}?. Link: <https://stackoverflow.com/questions/59308372/what-is-the-use-of-load-static>
5. What is a CSRF token? What is its importance and how does it work?. Link: <https://stackoverflow.com/questions/5207160/what-is-a-csrf-token-what-is-its-importance-and-how-does-it-work>