VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

WEB APPLICATION DEVELOPMENT PROJECT

Instructor: Mr. Nguyen Trung Nghia

FINAL REPORT

E-COMMERCE FLOWER WEBSITE

By Group Name "Narcissus" – Member List

Vũ Hồng Quang	ITCSIU21222 (Team Leader)
Hà Tấn Tài	ITCSIU21227 (Team Member)
Huỳnh Hữu Nghĩa	ITCSIU21207 (Team Member)

LIST OF CONTENTS

Chapter 1: INTRODUCTION....1
1.0. Introduction:....1
1.1. About Us:....1
1.2. About The Flower Website:....1
1.3. Work Breakdown Structure:....2
1.4. Development Process:....4
1.5. Development Environment:....5
Chapter 2: REQUIREMENT ANALYSIS AND DESIGN....7
2.0. Introduction:....7
2.1. Requirement Analysis:....8
2.2. Design:....26
Chapter 3: IMPLEMENTATION....33
3.1. UI Design and Functions:....33
Chapter 4: DISCUSSION AND CONCLUSION....51
Chapter 5: REFERENCES....52

Chapter 1: INTRODUCTION

1.0. Introduction:

This section briefly introduces the background information of our software development team – Narcissus. More importantly, the basic information about our project is also mentioned in this part.

1.1. About Us:

Building a flower shop website appears as a dynamic and sophisticated task in the rapidly advancing software technology and development This represents a confluence of disciplines, including web design, user experience, security policy and including effective data management. Our team, Narcissus, has undertaken this effort as part of the web application development theme. This project is an elective, designed to use the development cycles, time and functional dynamics of eCommerce platforms to refine our skills and apply the theoretical concepts explored in our web application development courses has been implemented in a practical manner.

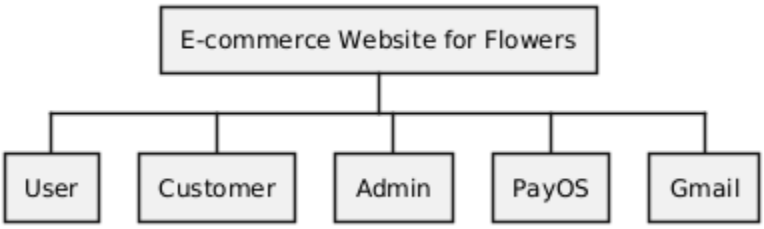
1.2. About The Flower Website:

Our website, Narcissus, provides users with an easy way to buy and sell flowers, while acting as a trusted intermediary to ensure the safety and confidence of both parties. The project invites users to a well-thought-out web-based experience, where they can search for vibrant flower offerings and communicate with confidence. The unique feature of our website is its role as a merchant, promoting security and reliability which sets it apart from traditional flower sales channels.

With a futuristic vision of optimizing the distribution system for cost efficiency, Narcissus not only achieves its immediate goals but also lays the groundwork for further expansion. This project challenges us to combine technical expertise with creative problem solving, incorporating the fundamentals of web application development into practical real-world applications.

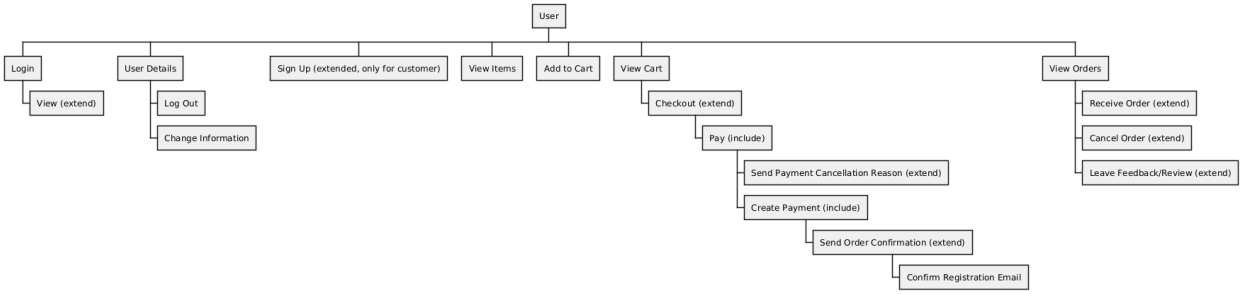
1.3. Work Breakdown Structure:

The structure of our project can be expressed:

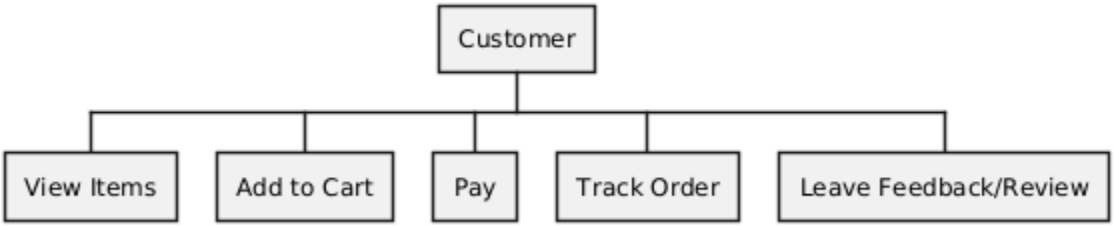


In each branch of this tree, we also have subtrees which describes the tasks needed to be accomplished of each team member.

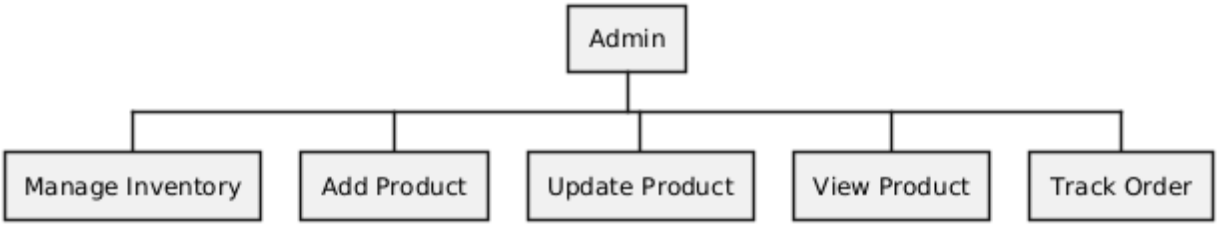
The task for the User:



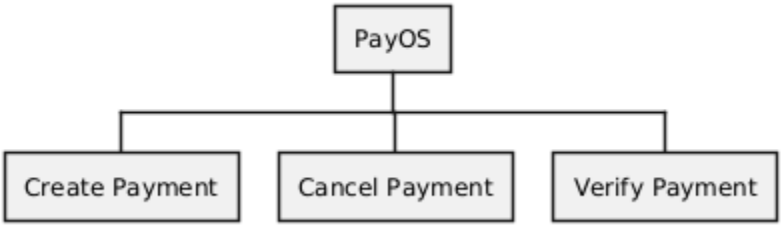
The task for the Customer:



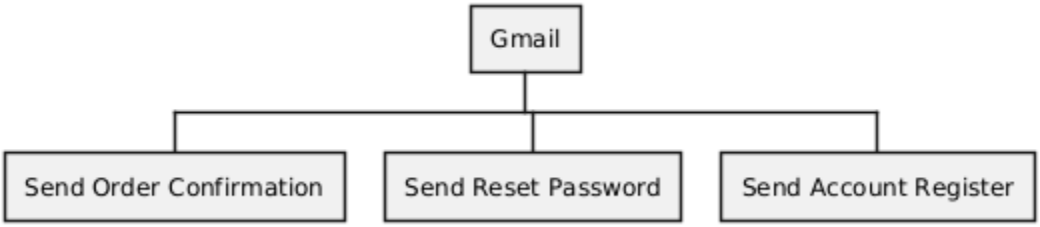
The task for the Admin:



The task for the PayOS:



The task for the Gmail:

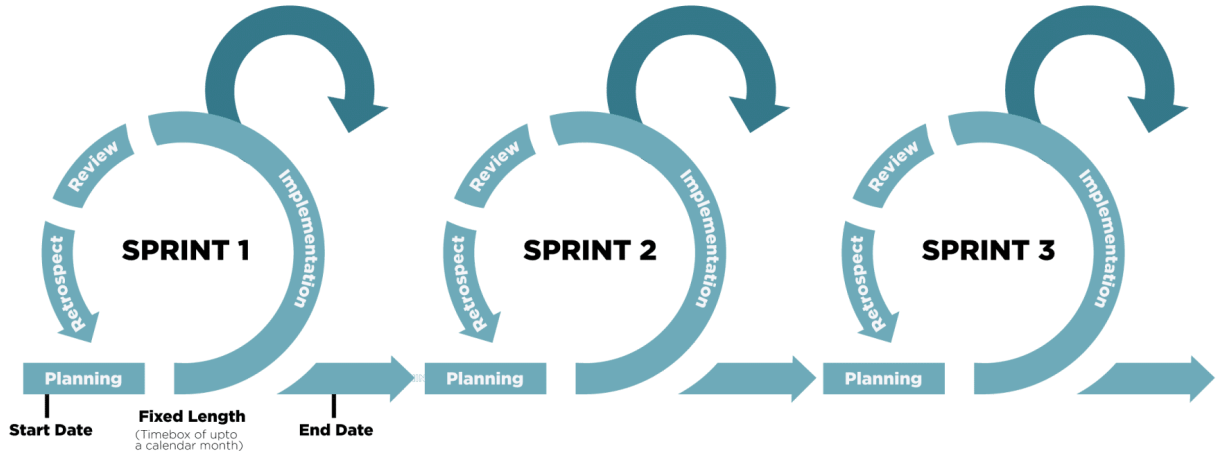


1.4. Development Process:

Our team has decided to adopt an Agile methodology, specifically the Scrum process, to guide the development of our project. Given the nature of our e-commerce flower shop, requirements are expected to change over time.

We use Trello to keep track of our running, allowing us to plan, track and organize tasks clearly and collaboratively. We aim to deliver the product incrementally, building each core function step by step based on customer needs.

After each sprint, new features and user-friendly interfaces are enhanced to ensure that users can seamlessly interact with the system to achieve their goals. That way this allows us to better adapt to change while continuing to provide value to our users.



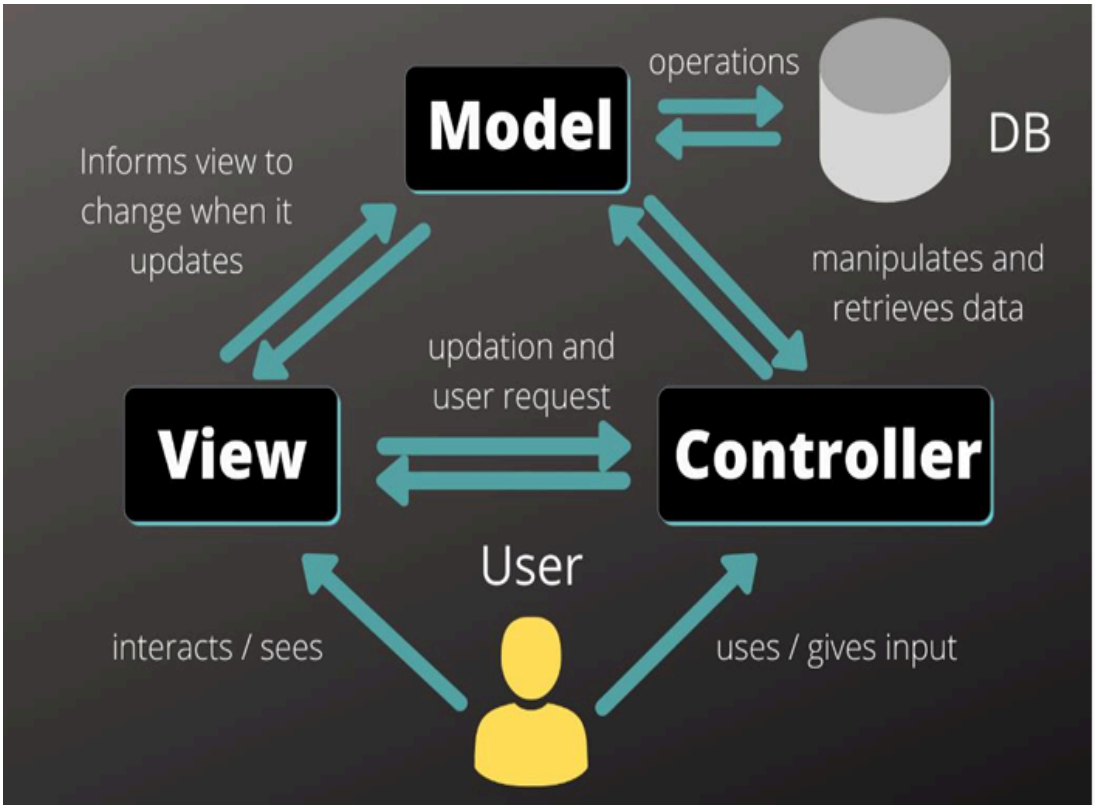
1.5. Development Environment:

Since this is a web-based product, our project is designed under the MVC (Model-View-Controller) architecture using a modern web design and programming language.

The following tools and technologies are used in our system.

- **Spring Boot**: To build a backend and manage server-side logic to properly handle requests and responses.
- **Tailwind CSS**: To design and create web pages with clean, responsive, and modern interfaces.
- **JavaScript**: Adding interactivity and dynamic actions to web pages.
- **MySQL**: To store and maintain the system's database, ensuring data integrity and security.
- **Figma**: To design the user interface prototype, ensuring a visually appealing and user-friendly layout.

These integrated tools ensure a scalable, manageable and user-centered eCommerce platform, which provides a smooth functionality and experience.



1. **Model**: Represents the data of the system, which is stored in the **MySQL** database and integrated with any external systems.

2. **View**: The user interface is designed using **Tailwind CSS** and interactive elements powered by **JavaScript**.
3. **Control**: The logic and algorithms are implemented using **Spring Boot** to develop a dynamic website with the required functionalities.

We use **Postman** to test and validate APIs, ensuring smooth communication between the frontend and backend.

We also use **Draw diagram tools** to help us create **Entity Relationship Diagrams (ERD)**, **Use Case Diagrams, and some other UML diagrams** essential for the project. Specifically, we utilize:

- **Draw.io**
- **Lucid.app**
- **ERDPlus.com**

Furthermore, we rely on project management tools like **Trello** to organize tasks, manage sprints, and track the overall progress, ensuring transparency in each team member's work performance.

Here is the link to our shared **GitHub repository** used for this project: https://github.com/VuH0ngQuang/Narcissus.

This repository helps us collaborate effectively, manage source code, track changes, and ensure version control throughout the development process.

Chapter 2: REQUIREMENT ANALYSIS AND DESIGN

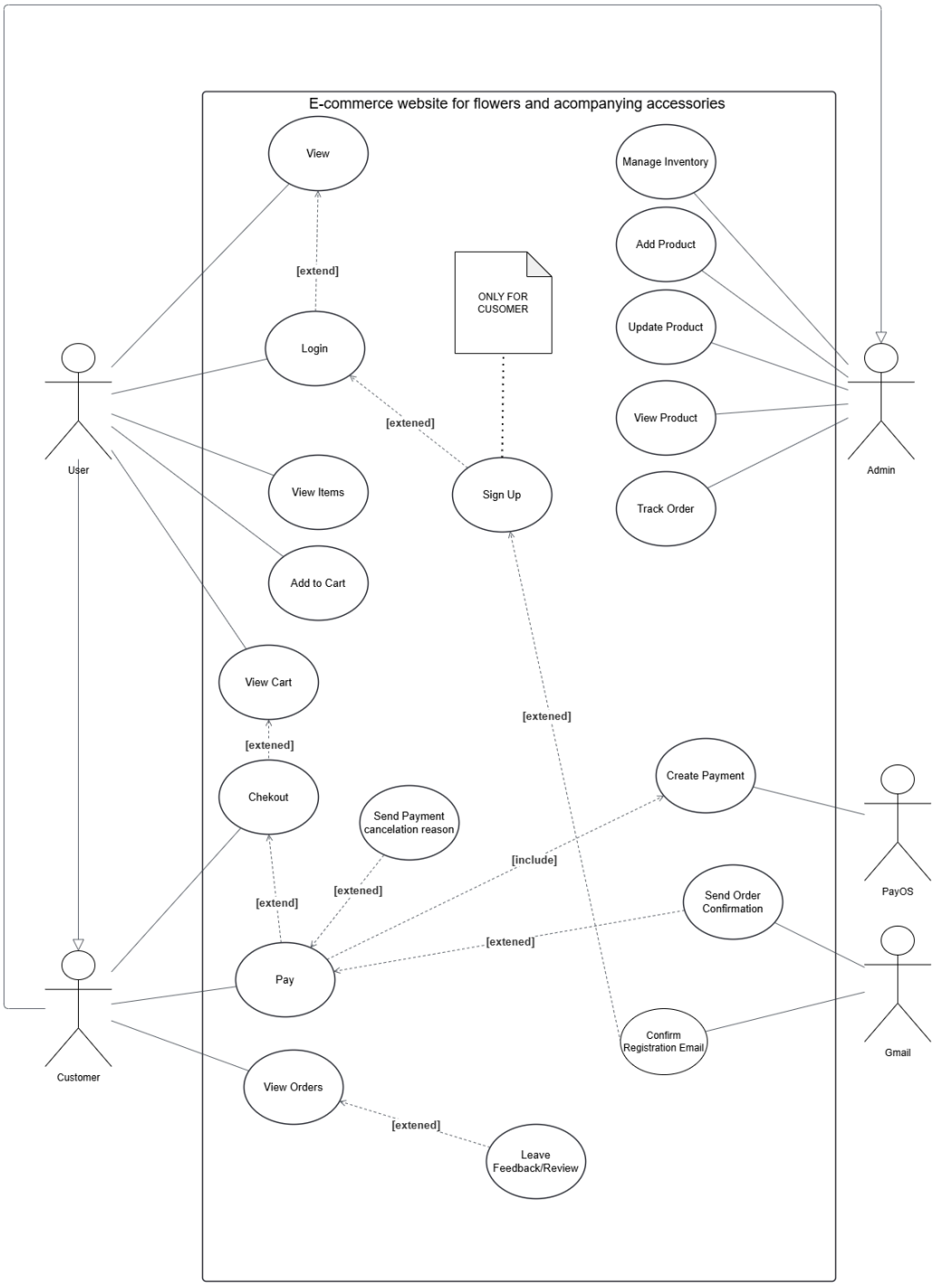
2.0. Introduction:

This section outlines the needs assessment and design process specifically designed for the development of our flower shop website. It provides a clear framework for future implementation, ensuring that all features, conditions, functional and non-functional requirements provided by users are adequately addressed.

While building the website, we take an iterative approach and will update and update the layout and functionality on a regular basis. This strategy ensures that the platform evolves seamlessly in order to meet user needs, while allowing us to better track progress and deliver high-quality, user-centric solutions.

2.1. Requirement Analysis:

USE CASE DIAGRAM



Use Case 1: Log In to the System

Identifier: UC1

Actors: User

Inputs:

1. Username
2. Password

Outputs:

1. Home page with user authorization (on success).
2. Login page with error message (on failure).

Basic Course:

1. The user opens the login page.
- The system displays the login form.
2. The user enters a username and password.
3. The user submits the form.
- The system verifies the credentials.
4. If credentials are valid:
- The system redirects to the home page.
5. If credentials are invalid:
- The system returns to the login page with an error message.

Preconditions:

The user must have a registered account.

Postconditions:

The user is logged into the system if successful.

User Story:

As a user, I want to log in to the system so that I can access available features based on my role (Customer or Admin).

Use Case 2: Sign Up

Identifier: UC2

Actors: User (Customer)

Inputs:

User information (user name, email, password, address, phone number)

Outputs:

1. Account confirmation email
2. Login page after successful registration

Basic Course:

1. The user selects the "Sign Up" option.
2. The user fills out the registration form.
3. The user submits the form.
- The system validates the information.
4. If the information is valid:
- The system creates an account.
- The system sends a confirmation email (via Gmail).
5. The system redirects to the login page.

Preconditions:

The user is not already registered.

Postconditions:

A new user account is created.

User Story:

As a user, I want to sign up so that I can access the e-commerce system's features.

Use Case 3: View/View Items

Identifier: UC3

Actors: User

Inputs:

None

Outputs:

A list of items/products available in the store.

Basic Course:

1. The user accesses the Homepage.
2. The user selects " Items" to view.
1. The system retrieves and displays the product list.

Preconditions:

Products exist in the system.

Postconditions:

None.

User Story:

As a customer, I want to view available products so that I can browse and decide what to purchase.

Use Case 4: Add to Cart

Identifier: UC4

Actors: User

Inputs:

Selected product

Outputs:

The item is added to the cart.

Basic Course:

1. The user selects an item to add to the cart.
2. The system adds the item to the cart.
3. The system confirms the addition to the cart.

Preconditions:

The product must be available.

Postconditions:

The cart is updated with the selected product.

User Story:

As a customer, I want to add products to my cart so that I can purchase them later.

Use Case 5: View Cart

Identifier: UC5

Actors: User

Inputs:

None

Outputs:

Cart details (products (product image, product name), quantities, product price, total price))

Basic Course:

The user selects "Cart."

The system displays the cart contents.

Preconditions:

The user has added products to the cart.

Postconditions:

User must login to the website.

User Story:

As a customer, I want to view my cart so that I can check what I plan to purchase.

Use Case 6: Checkout

Identifier: UC6

Actors: Customer (Admin)

Inputs:

Checkout details.

Outputs:

Payment confirmation.

Basic Course:

1. Checkout details will be selected by two options:
    - when user presses the “Buy Now” button
    - when the user selects a product and adds it to cart to make a purchase.
2. The system confirms the information and prints out the payment confirmation.

Preconditions:

1. The cart has products then the user presses the “make a purchase” button after selecting products in the cart.
2. User presses the “Buy Now” button.

Postconditions:

Payment confirmation displayed.

User Story:

As a customer, I want to complete the checkout process so that I can confirm my purchase.

Use Case 7: Pay

Identifier: UC7

Actors: Customer (Admin)

Inputs:

Click on “Pay” con the payment confirmation

Outputs:

Create payment

Basic Course:

1. The user selects "Pay".
2. The system verifies and processes the payment through **PayOS** or sending the payment cancellation reason.
3. The system sends an order confirmation email via **Gmail**.

Preconditions:

1. Payment information is successfully confirmed.
2. User clicks on “Pay” and scans the QR to purchase (QR is created by Create Payment use case).

Postconditions:

Payment is completed, and confirmation email is sent.

User Story:

As a customer, I want to complete the checkout process so that I can confirm my purchase.

Use Case 8: Manage Inventory

Identifier: UC8

Actors: Admin

Inputs:

Login with role admin

Outputs:

List of current product inventory

Basic Course:

1. The admin is logged in
2. The system displays the current inventory list.

Preconditions:

The admin is logged in.

Postconditions:
The current inventory list is displayed.

User Story:

As an admin, I want to manage my inventory so that I can ensure products are up-to-date.

Use Case 9: Add Product

Identifier: UC9

Actors: Admin

Inputs:

Product details (name, information, type, price, stock, image)

Outputs:

Product added to the inventory.

Basic Course:

1. The admin selects "Add Product."
2. The system prompts for product details.
3. The admin provides details and submits.
4. The system adds the product to the inventory.

Preconditions:

The admin is logged in and selects “Add product”.

Postconditions:

The inventory is updated with the new product.

User Story:

As an admin, I want to add new products to my inventory so that I can keep my store updated.

Use Case 10: Track Order

Identifier: UC10

Actors: Admin

Inputs:

None

Outputs:

Order status (Order ID, Order On, Payment Status, Price, “View Details” button)

Order detail (Product ID, Product name, quantity, price)

Basic Course:

1. The admin selects "Order."
2. The system retrieves and displays the order status.
3. Admin clicks on the “View Details” button for each order to see the details of order.

Preconditions:

Orders exist in the system.

Postconditions:

None.

User Story:

As an admin, I want to track customer orders so that I can ensure timely delivery.

Use Case 11: Update Product/ View Product

Identifier: UC11

Actors: Admin

Inputs:

1. For update (name, information, type, price, stock, image)
2. For delete (none)
3. For view (none)

Outputs:

1. For update (update product)
2. For view (display product)
3. For delete (delete product)

Basic Course:

1. The admin selects "Update"
2. The system retrieves and displays product detail button delete and update.
3. Admin adds the new information which replaces the old one and clicks on “Update” button for updating, “Delete” button for deleting product, and “Return” button to back to the previous page (which is view details).

Preconditions:

Product exists in the system.

Postconditions:

1. For update (update product)
2. For view (display product)
3. For delete (delete product)

User Story:

As an admin, I want to view, update, and delete products so that I can update the latest information and also remove the discontinued products.

Use Case 12: Send Payment cancellation reason

Identifier: UC12

Actors: Customer

Inputs:
Cancellation reason (optional)

Outputs:
Back to Purchased Product page

Basic Course:

1. In the payment process, the customer selects "Do not want to pay”.
2. The customer writes the cancellation reason (optional) and clicks “Send”.
3. The system will turn back to the previous page.

Preconditions:
Payment processing

Postconditions:
The cancellation reason will be sent.

User Story:
As the customer, I do not want to pay anymore and want to send the reason.

Use Case 13: Create Payment

Identifier: UC13

Actors: PayOS

Inputs:

1. Product name
2. Quantity
3. Price per product (not yet multiplied by quantity)

Outputs:

QR code embedded through PayOS, containing the necessary information (Checkout URL from PayOS).

Basic Course:

1. The customer clicks on "Pay".
2. The product name, quantity, and price per product are sent to the actor PayOS.
3. The actor PayOS responds with the Checkout URL.
4. The QR code is displayed and embedded on the website using the Checkout URL provided by PayOS.

Preconditions:

Product name, quantity, and price per product (not yet multiplied by quantity) are available.

Postconditions:

QR code is successfully embedded through PayOS using the provided Checkout URL.

User Story:

As a customer, I want a faster and more convenient payment method to quickly complete my transactions.

As an admin, I want a solution that allows users to pay me efficiently and provides clear transaction details.

Use Case 14: Send Order Confirmation

Identifier: UC14

Actors: Gmail

Inputs:

1. Username
2. Address
3. Total price
4. Set of products
5. Email
6. String (message content)

Outputs:

Order confirmation email sent to the customer

Basic Course:

1. The system sends the order information to Gmail.
2. The actor gmail processes the information and sends the confirmation email to the customer.
3. The customer receives the order confirmation email.

Preconditions:

Username, address, total price, and set of products are available.

Postconditions:

An order confirmation email is successfully sent to the customer.

User Story:

As a customer, I want to receive confirmation details about my order.

As an admin, I want the order information to be sent to customers for confirmation as part of good customer service.

Use Case 15: Confirm Registration Email

Identifier: UC15

Actors: Gmail

Inputs:

1. Username
2. Password
3. String (message content)
4. Email

Outputs:

Registration confirmation email sent to the customer

Basic Course:

1. The system sends the registration information to Gmail.
2. The actor Gmail processes the information and sends the registration confirmation email to the customer.
3. The customer receives the registration confirmation email.

Preconditions:

Username, password, string (message), and email are available.

Postconditions:

A registration confirmation email is successfully sent to the customer.

User Story:

As a customer, I want to receive confirmation details about my registration.

As an admin, I want the registration information to be sent to customers for confirmation as part of good customer service.

Use Case 16: View Orders

Identifier: UC16

Actors: Customer

Inputs:

None

Outputs:

Information about orders

Basic Course:

1. The customer clicks a button to navigate to the "Orders" page.
2. The system displays a list of the customer's orders.
3. The customer can view details about individual products within each specific order.

Preconditions:

The customer clicks the "View Orders" button.

Postconditions:

The system displays order information along with product details for each specific order.

User Story:

As a customer, I want to view my order history and check the details of individual products within each specific order.

Use Case 17: Leave Feedback/Review

Identifier: UC17

Actors: Customer

Inputs:

A comment (text string) and a star rating (numerical rating).

Outputs:

The submitted comment and star rating are displayed publicly for other users to view.

Basic Course:

1. The system verifies that the order status is "Delivered."
2. Once the delivery is confirmed, the customer gains access to the feedback/review feature.
3. The customer inputs their comment and star rating through the interface.
4. The customer submits the feedback by clicking the "Submit" button.
5. The system publishes the comment and star rating, making them visible to all other users.

Preconditions:

1. The order status must be confirmed as "Delivered."
2. The customer provides a valid comment and star rating.

Postconditions:

The feedback is publicly displayed and accessible for all users to view.

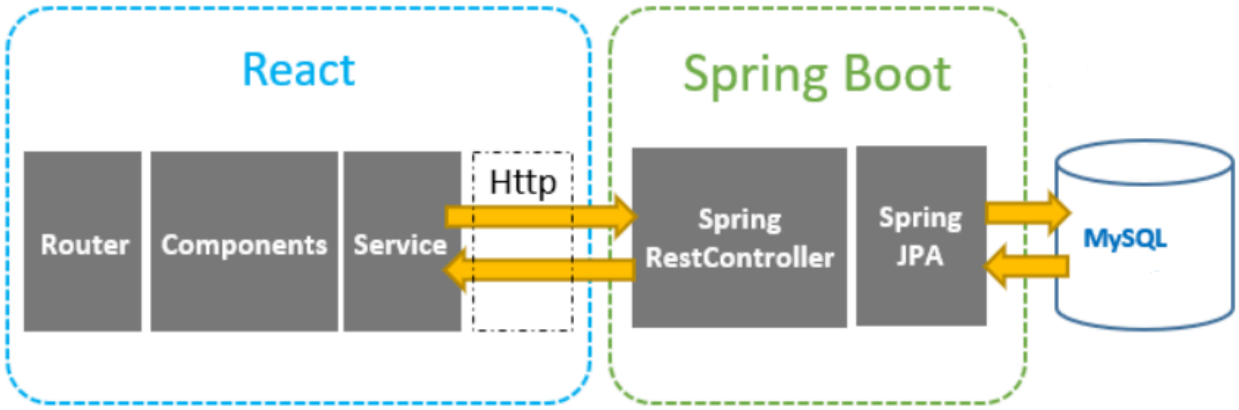
User Story:

As a customer, I want to read reviews from previous buyers to decide whether a product is worth purchasing.

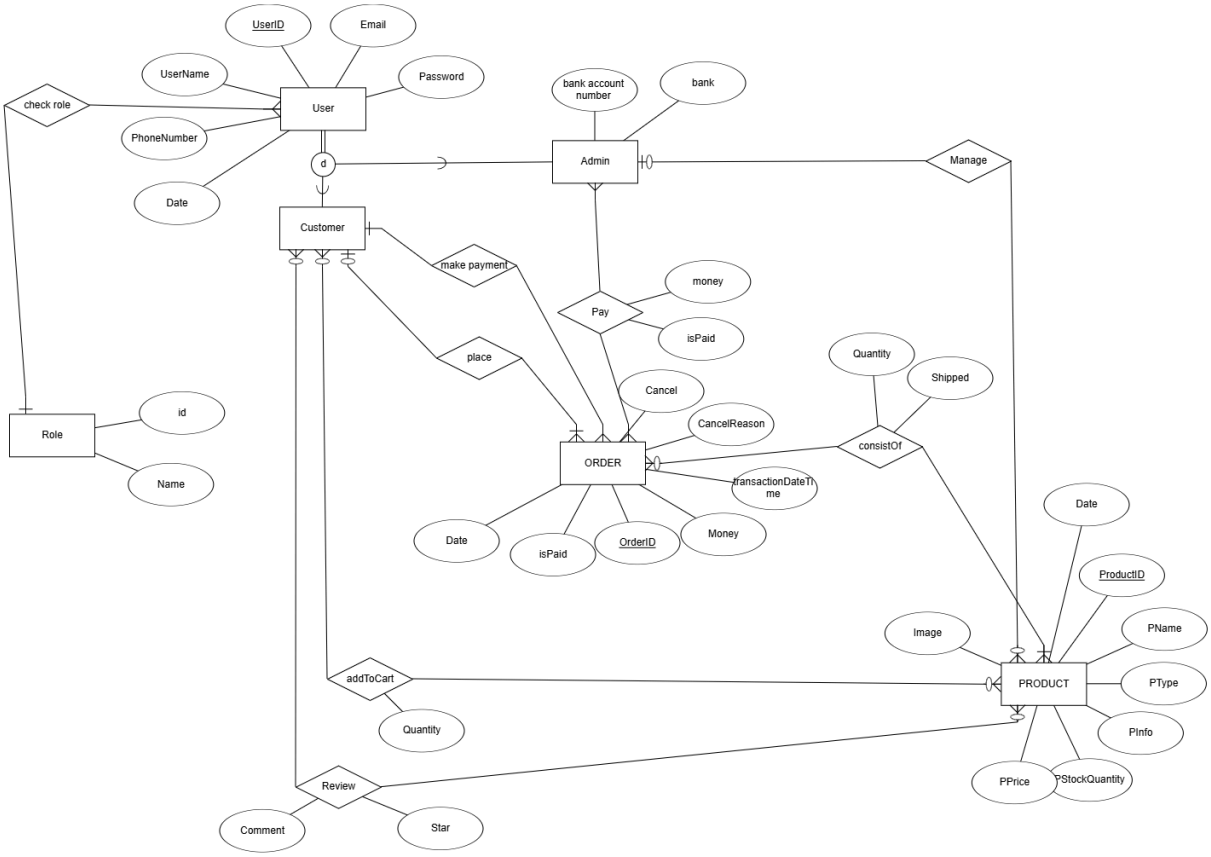
As a customer, I want to leave feedback and a rating for a product I have received to help other buyers make informed decisions.

2.2. Design:

System Architecture Model:



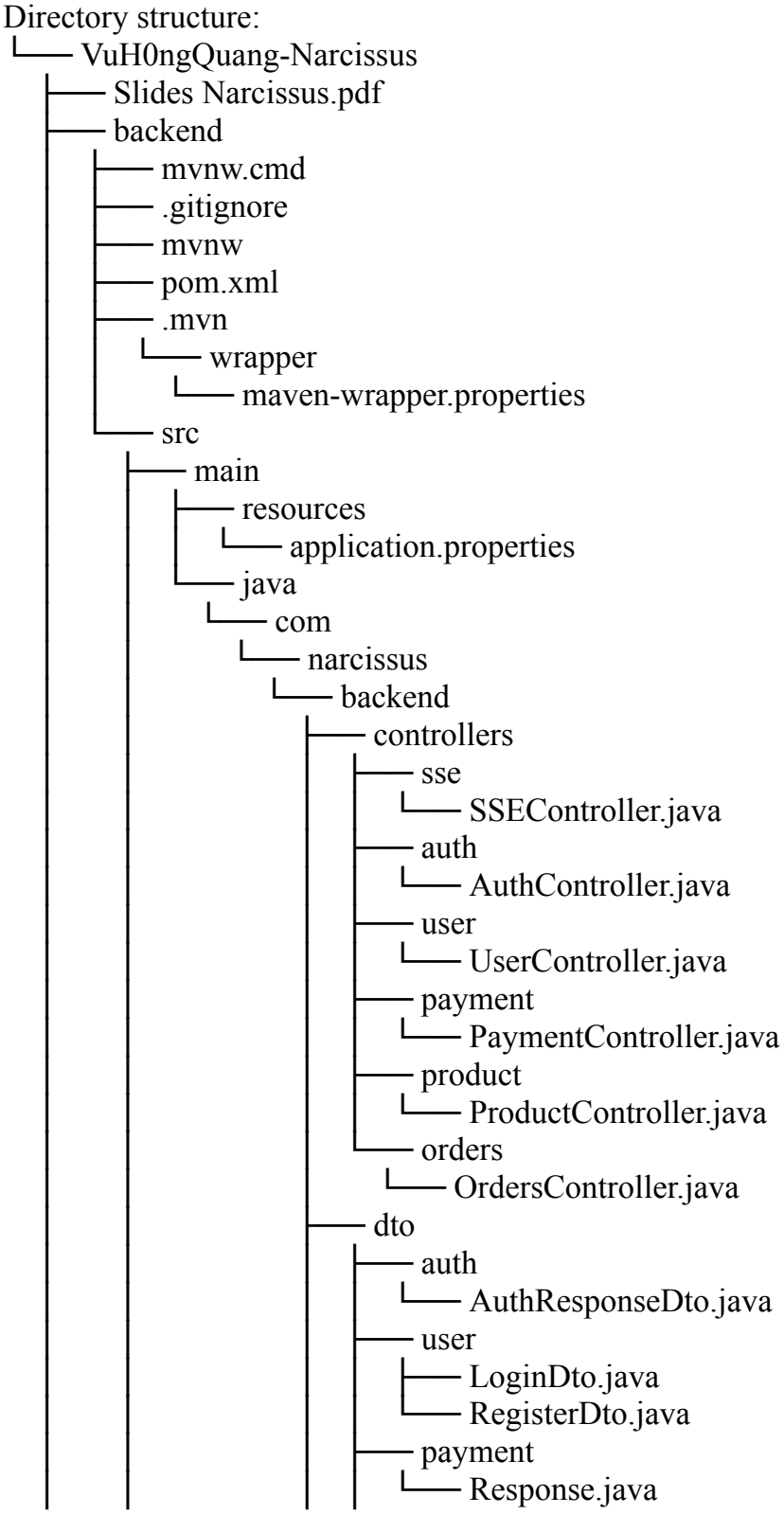
Entity-Relationship Diagram (ERD):

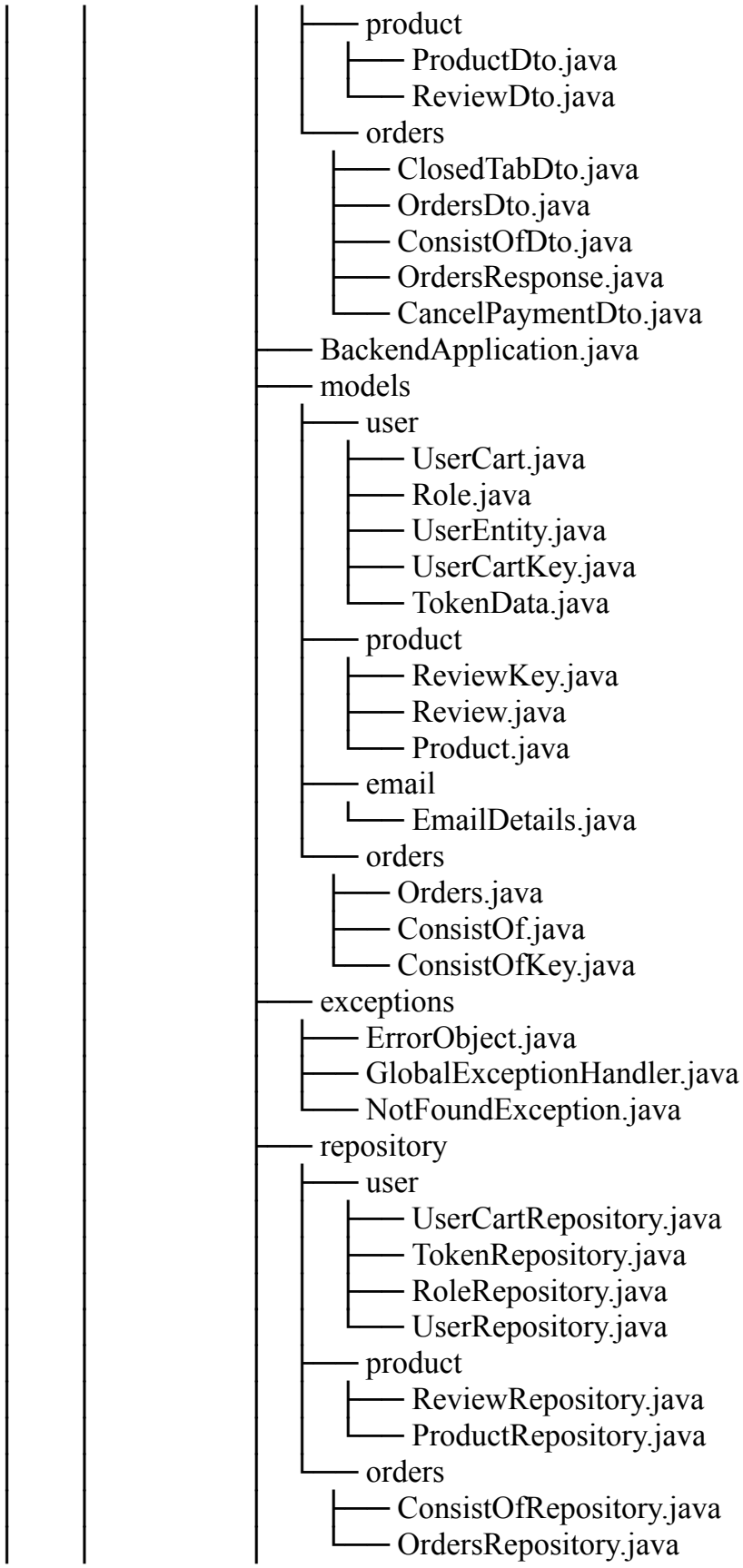


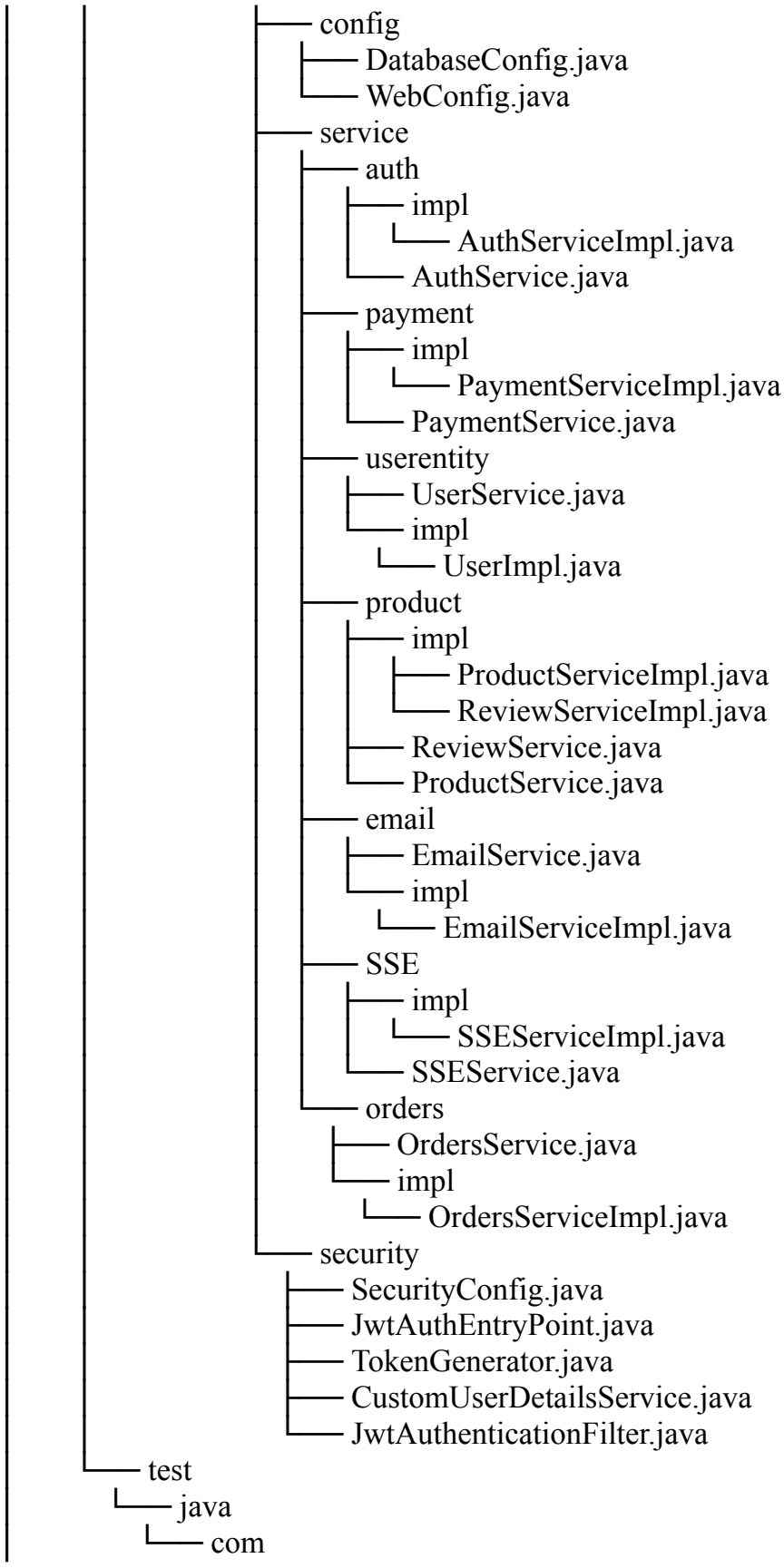
Class Diagram:

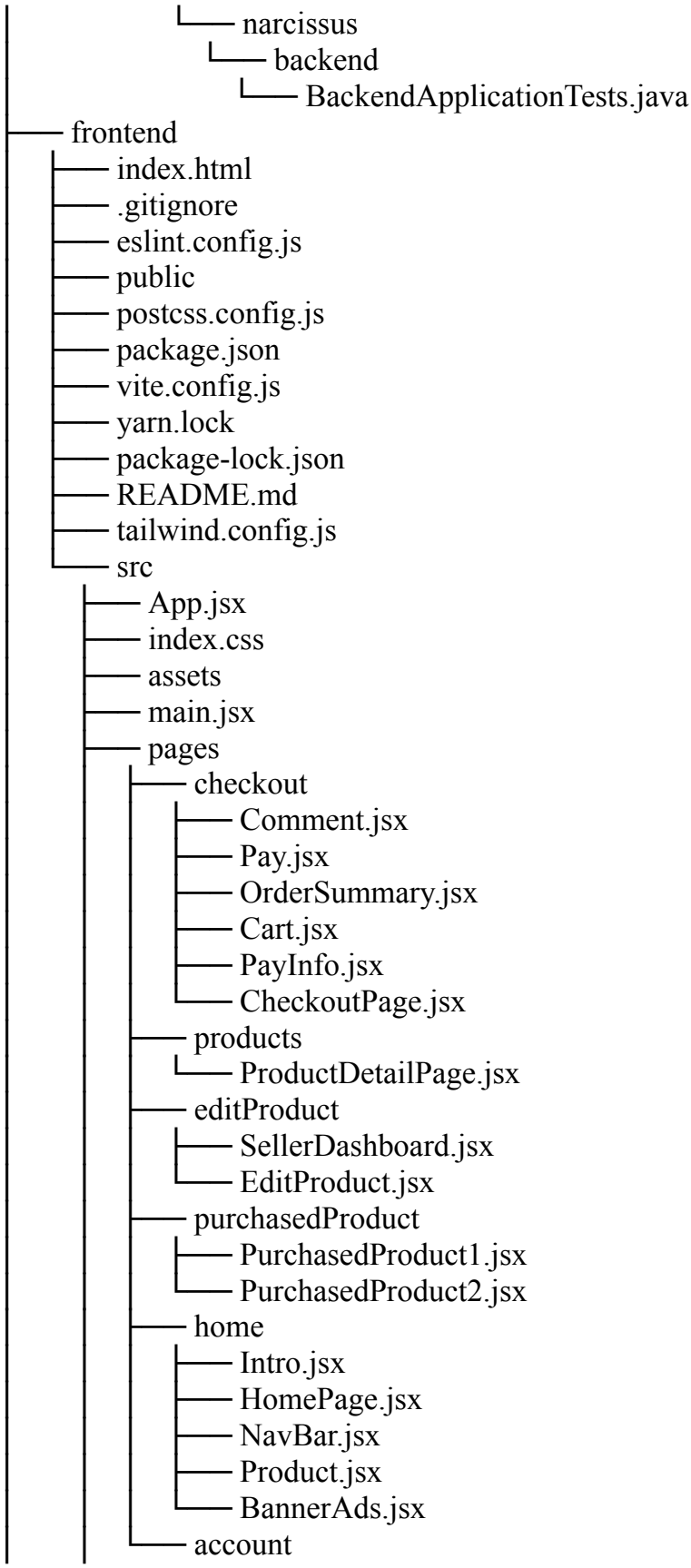
Find the Full HD image on our github

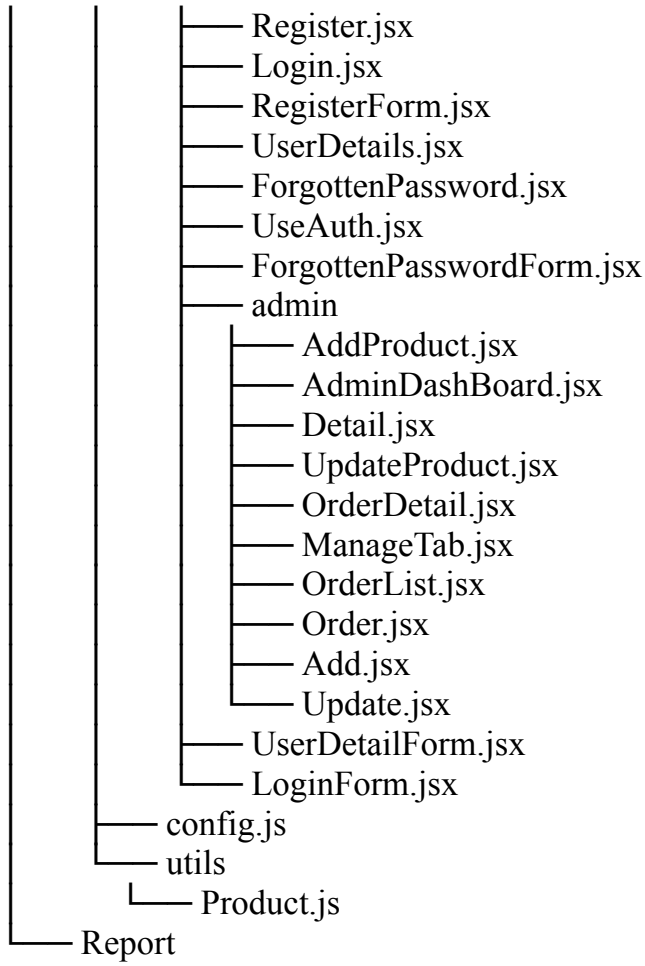
Directory structure:




product
ProductDto.java
ReviewDto.java
orders
ClosedTabDto.java
OrdersDto.java
ConsistOfDto.java
OrdersResponse.java
CancelPaymentDto.java
BackendApplication.java
models
user
UserCart.java
Role.java
UserEntity.java
UserCartKey.java
TokenData.java
product
ReviewKey.java
Review.java
Product.java
email
EmailDetails.java
orders
Orders.java
ConsistOf.java
ConsistOfKey.java
exceptions
ErrorObject.java
GlobalExceptionHandler.java
NotFoundException.java
repository
user
UserCartRepository.java
TokenRepository.java
RoleRepository.java
UserRepository.java
product
ReviewRepository.java
ProductRepository.java
orders
ConsistOfRepository.java
OrdersRepository.java


config
DatabaseConfig.java
WebConfig.java
service
auth
impl
AuthServiceImpl.java
AuthService.java
payment
impl
PaymentServiceImpl.java
PaymentService.java
userentity
UserService.java
impl
UserImpl.java
product
impl
ProductServiceImpl.java
ReviewServiceImpl.java
ReviewService.java
ProductService.java
email
EmailService.java
impl
EmailServiceImpl.java
SSE
impl
SSEServiceImpl.java
SSEService.java
orders
OrdersService.java
impl
OrdersServiceImpl.java
security
SecurityConfig.java
JwtAuthEntryPoint.java
TokenGenerator.java
CustomUserDetailsService.java
JwtAuthenticationFilter.java
test
java
com


narcissus
backend
BackendApplicationTests.java
frontend
index.html
.gitignore
eslint.config.js
public
postcss.config.js
package.json
vite.config.js
yarn.lock
package-lock.json
README.md
tailwind.config.js
src
App.jsx
index.css
assets
main.jsx
pages
checkout
Comment.jsx
Pay.jsx
OrderSummary.jsx
Cart.jsx
PayInfo.jsx
CheckoutPage.jsx
products
ProductDetailPage.jsx
editProduct
SellerDashboard.jsx
EditProduct.jsx
purchasedProduct
PurchasedProduct1.jsx
PurchasedProduct2.jsx
home
Intro.jsx
HomePage.jsx
NavBar.jsx
Product.jsx
BannerAds.jsx
account


Register.jsx
Login.jsx
RegisterForm.jsx
UserDetails.jsx
ForgottenPassword.jsx
UseAuth.jsx
ForgottenPasswordForm.jsx
admin
AddProduct.jsx
AdminDashBoard.jsx
Detail.jsx
UpdateProduct.jsx
OrderDetail.jsx
ManageTab.jsx
OrderList.jsx
Order.jsx
Add.jsx
Update.jsx
UserDetailForm.jsx
LoginForm.jsx
config.js
utils
Product.js
Report

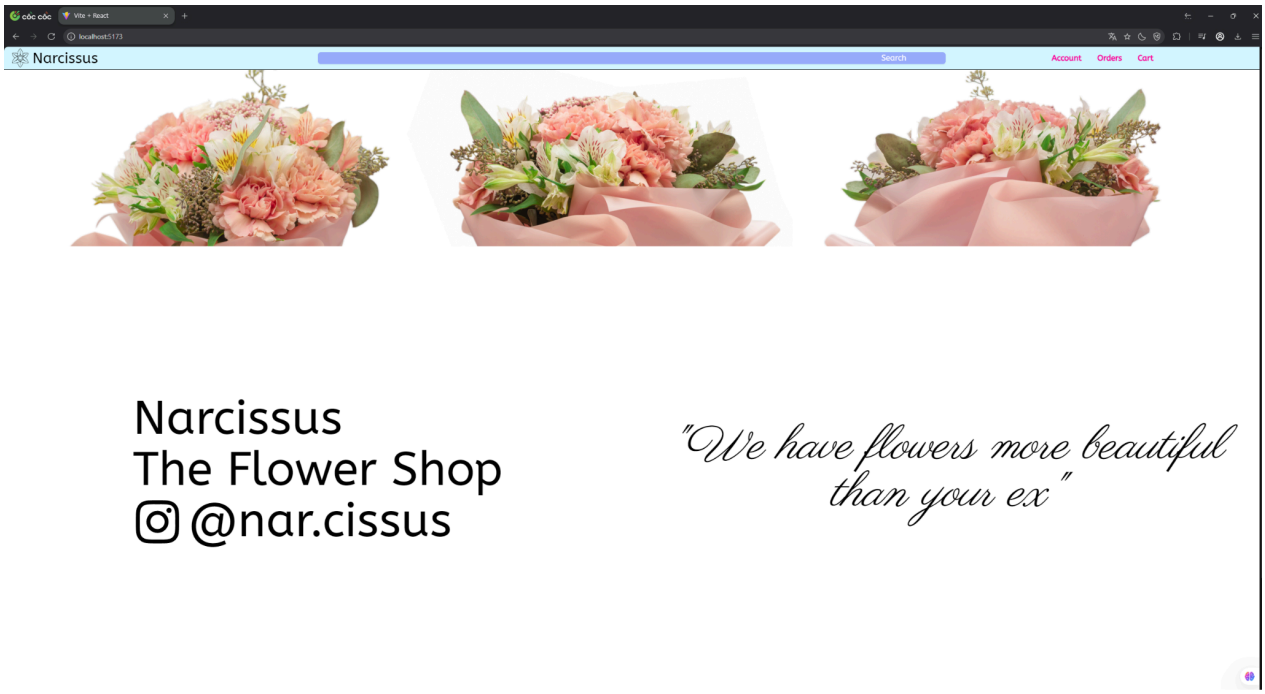
Chapter 3: IMPLEMENTATION

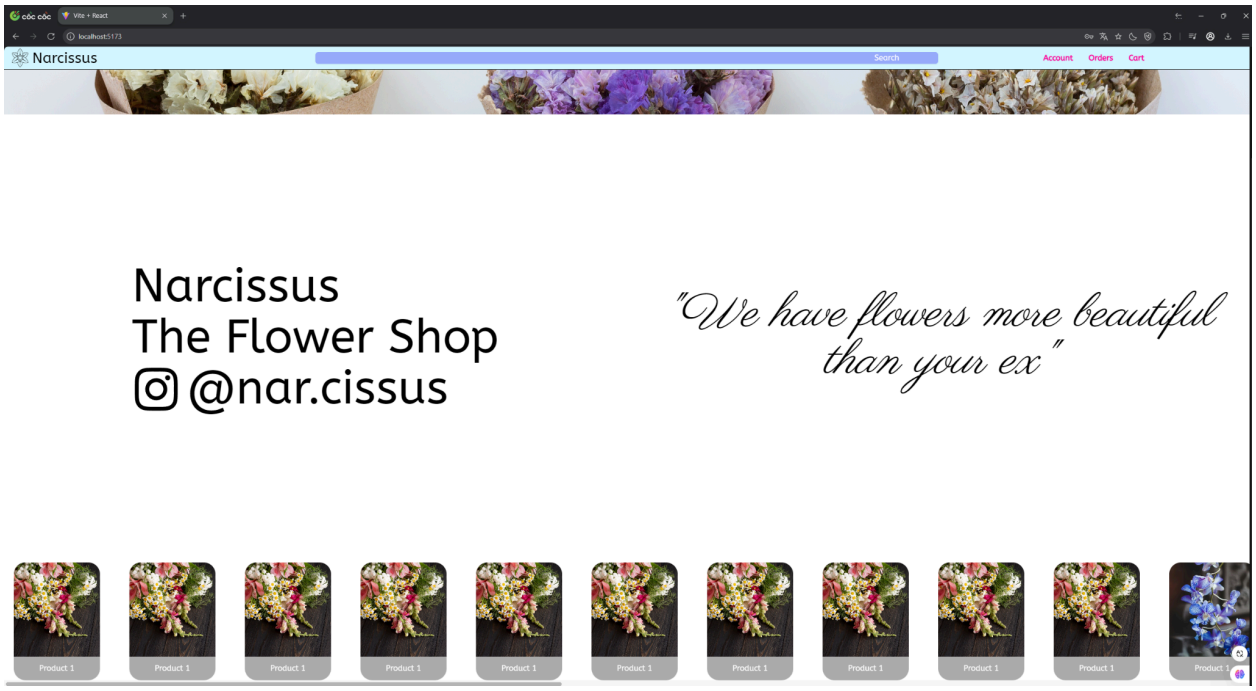
3.1. UI Design and Functions:

Add a taskbar on every page with the following features:

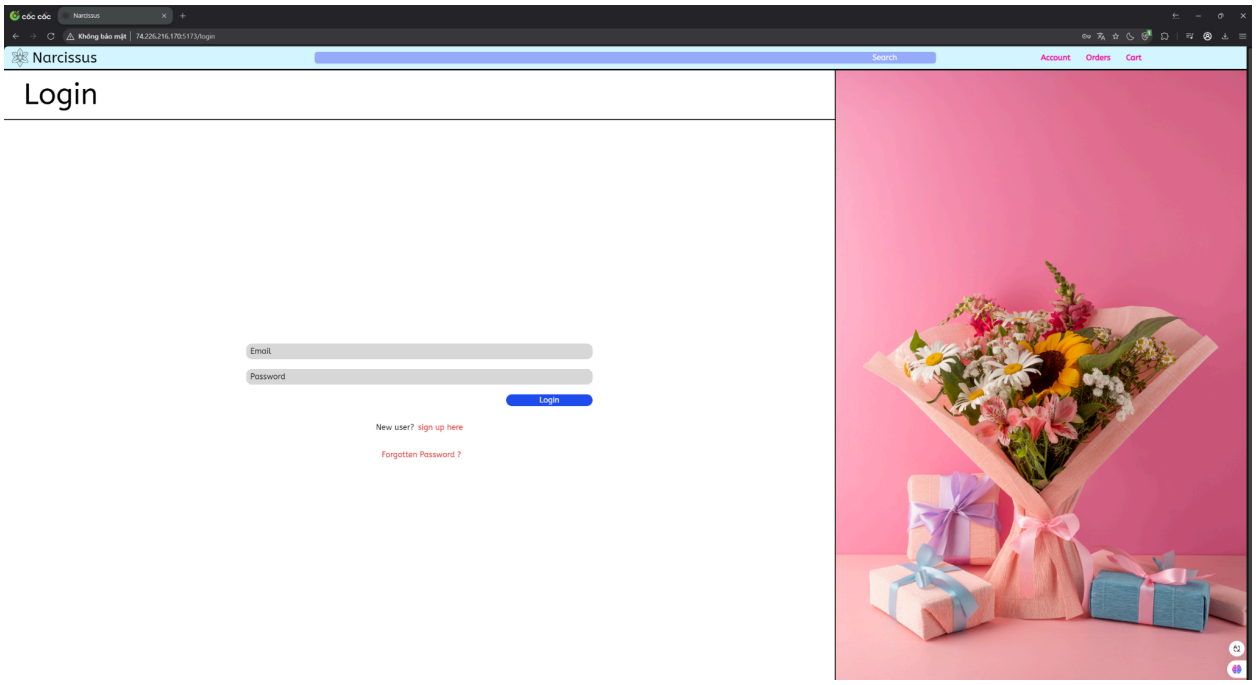
- **Narcissus**: Clicking this will redirect to the homepage.
- **Account**: To log in.
- **Order**: To view orders, but login is required.
- **Cart**: To view the cart, but login is required.

The **homepage** includes basic information and displays the products we offer. Users can scroll horizontally to view more products.

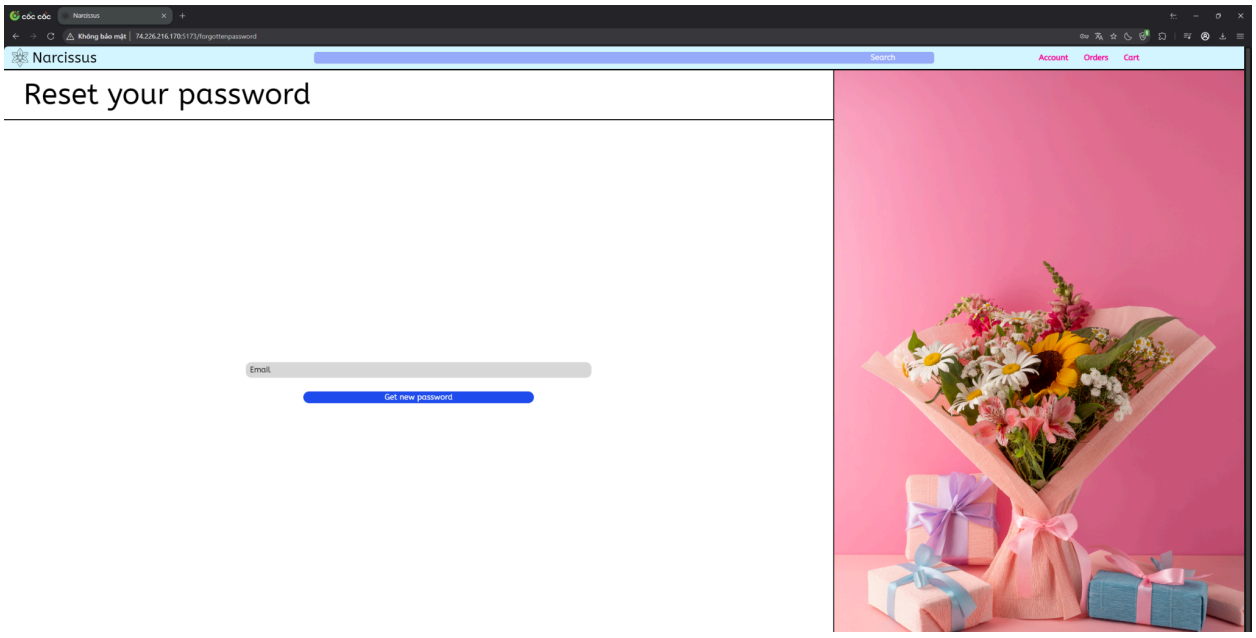



Narcissus
Search
Account
Orders
Cart
Narcissus
The Flower Shop
@nar.cissus
"We have flowers more beautiful than your ex"
Product 1

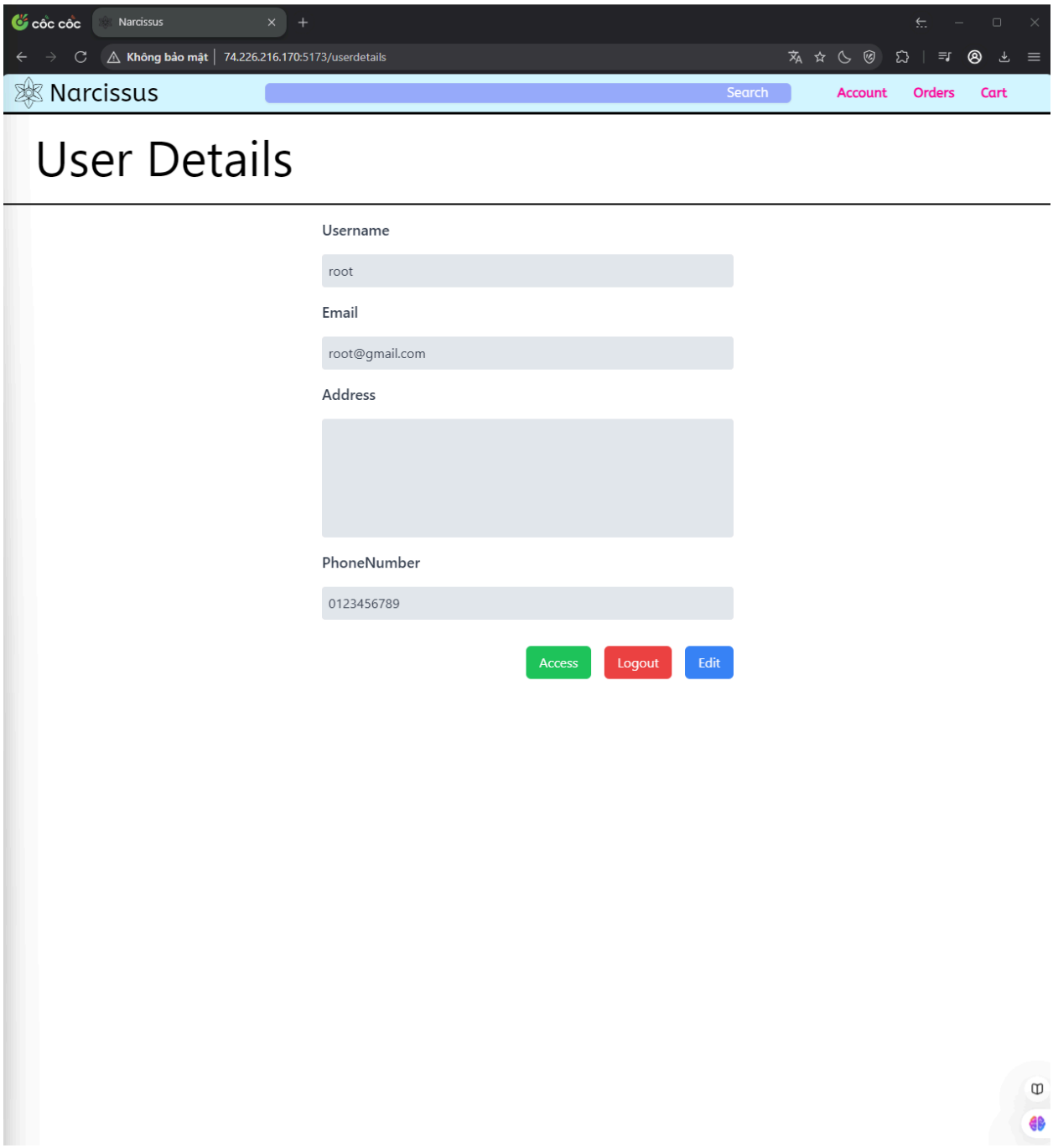
This is the **Login Page**, requiring an email and password to log in. Upon successful login, the system will redirect to the **Homepage** if the user has a customer role, or to the **AdminDashboard** if the user has an admin role.

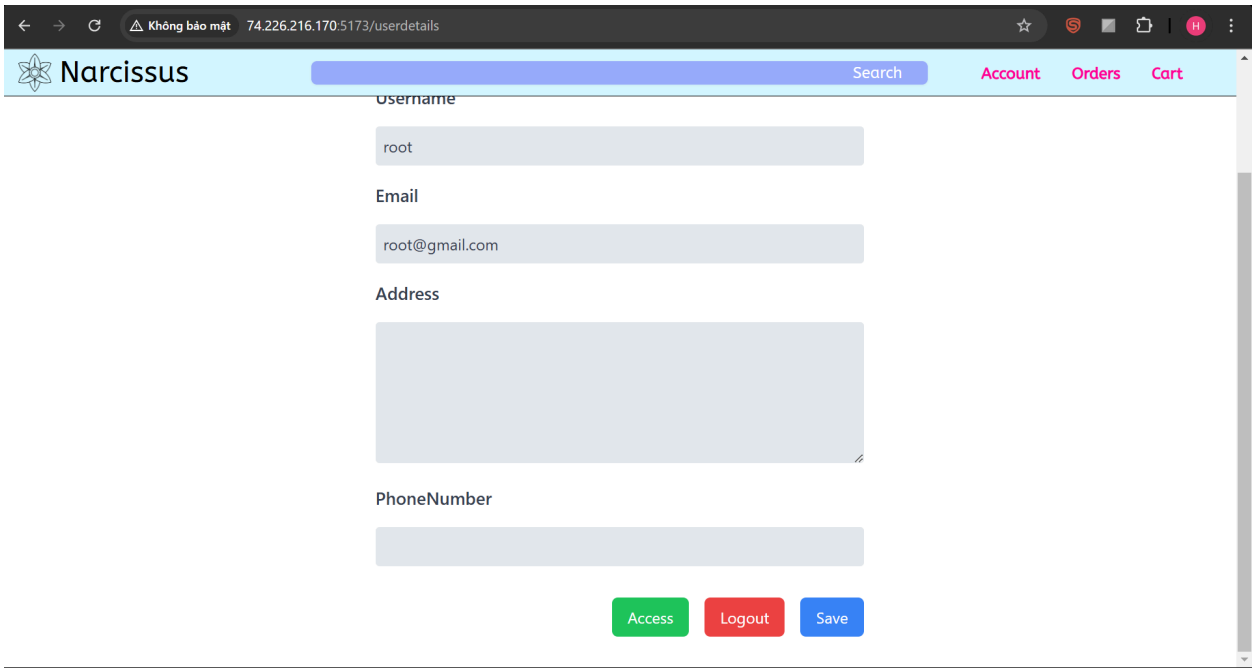


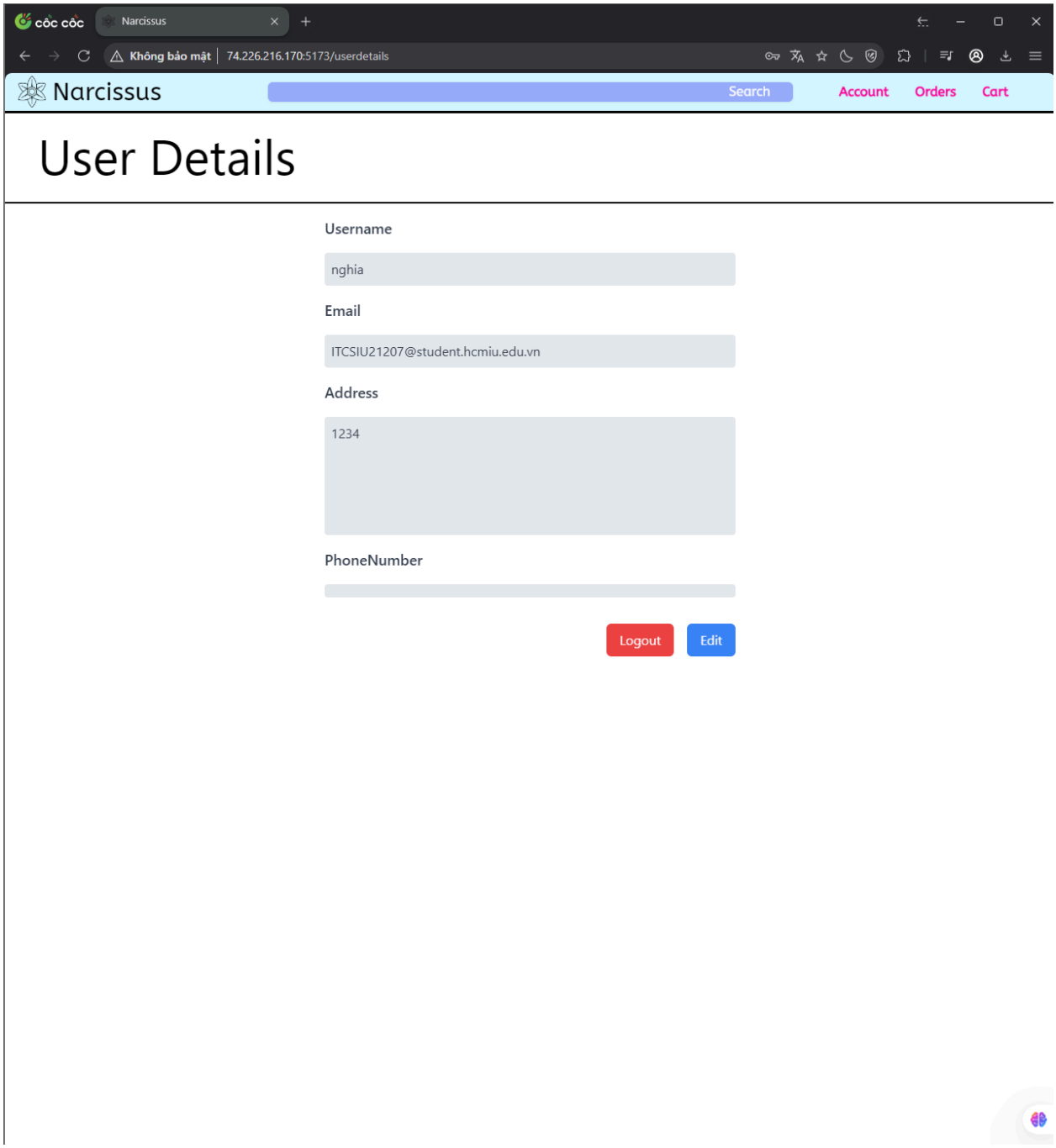
This is the **Reset Password Page**. It is used to recover a new password by entering your email and clicking submit, allowing you to get a new password if you've forgotten the old one.



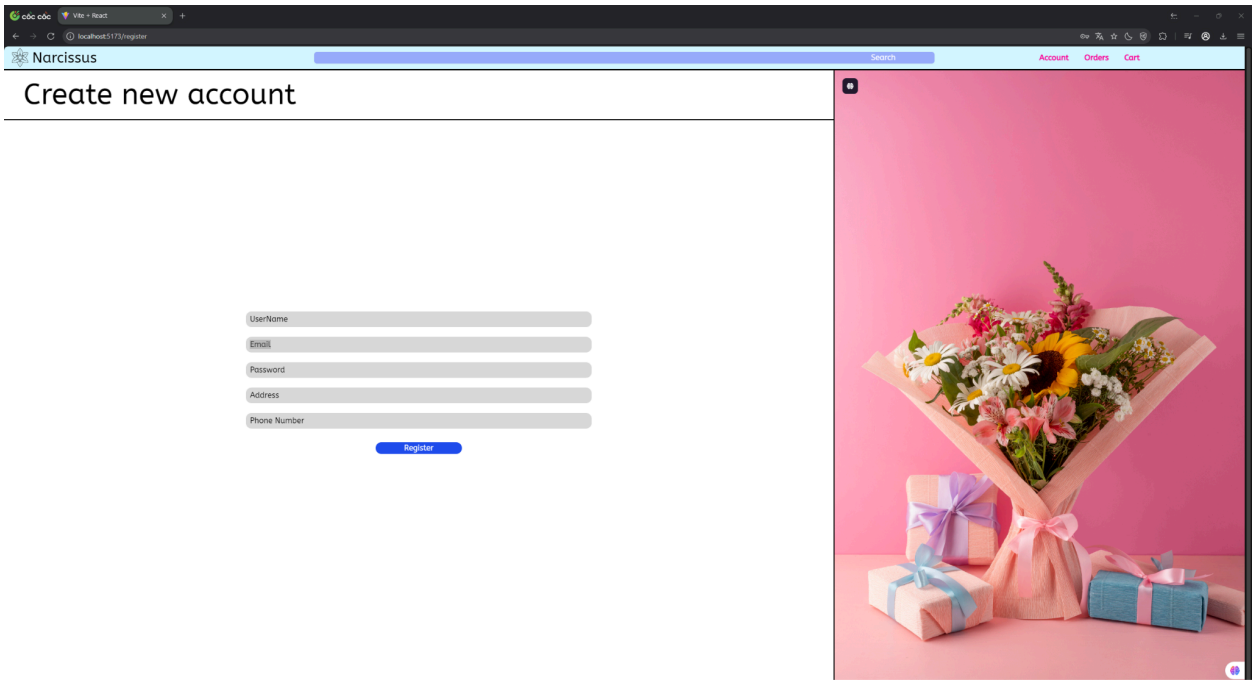
This is the **User Details Page**, which has two versions: one for admin and one for regular users. For admins, there is an additional "Access" button to navigate to the **Admin Dashboard Page**. The "Logout" button is used to log out of the account, and the "Edit" button is used to update information. Once edits are made, the "Save" button (which replaces the previous "Edit" button) is used to save the updated information.



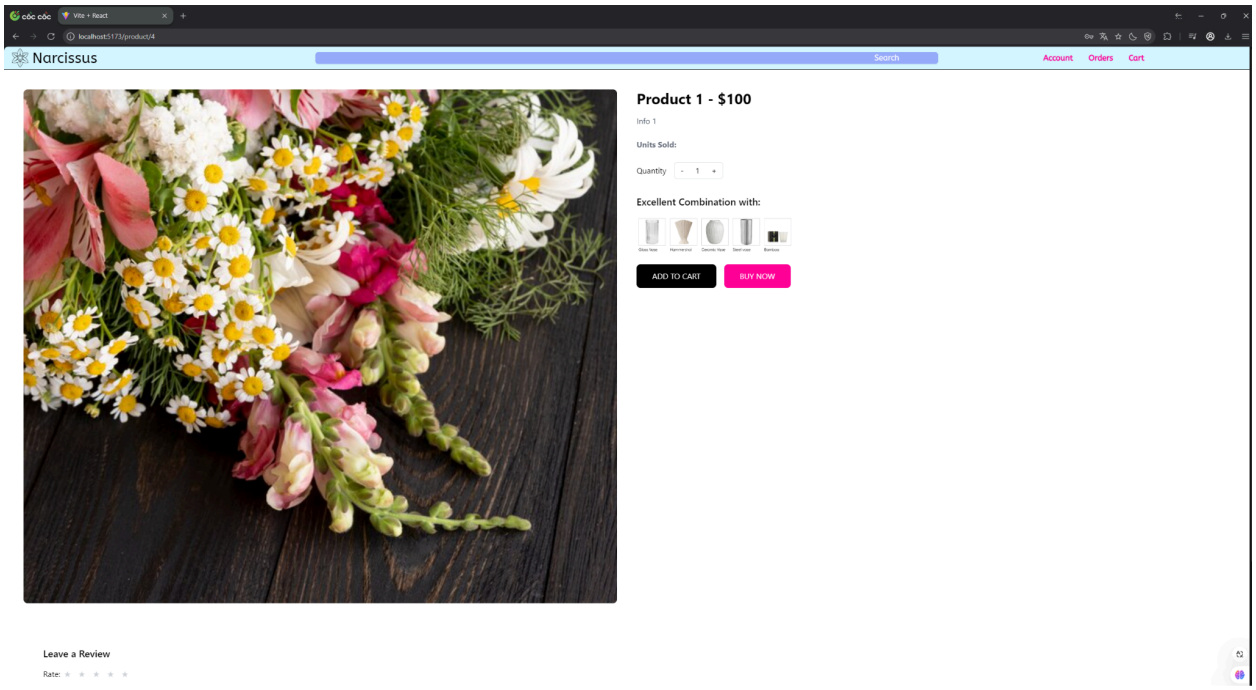

Không bảo mật
74.226.216.170:5173/userdetails
Narcissus
Search
Account
Orders
Cart
root
Email
root@gmail.com
Address
PhoneNumber
Access
Logout
Save

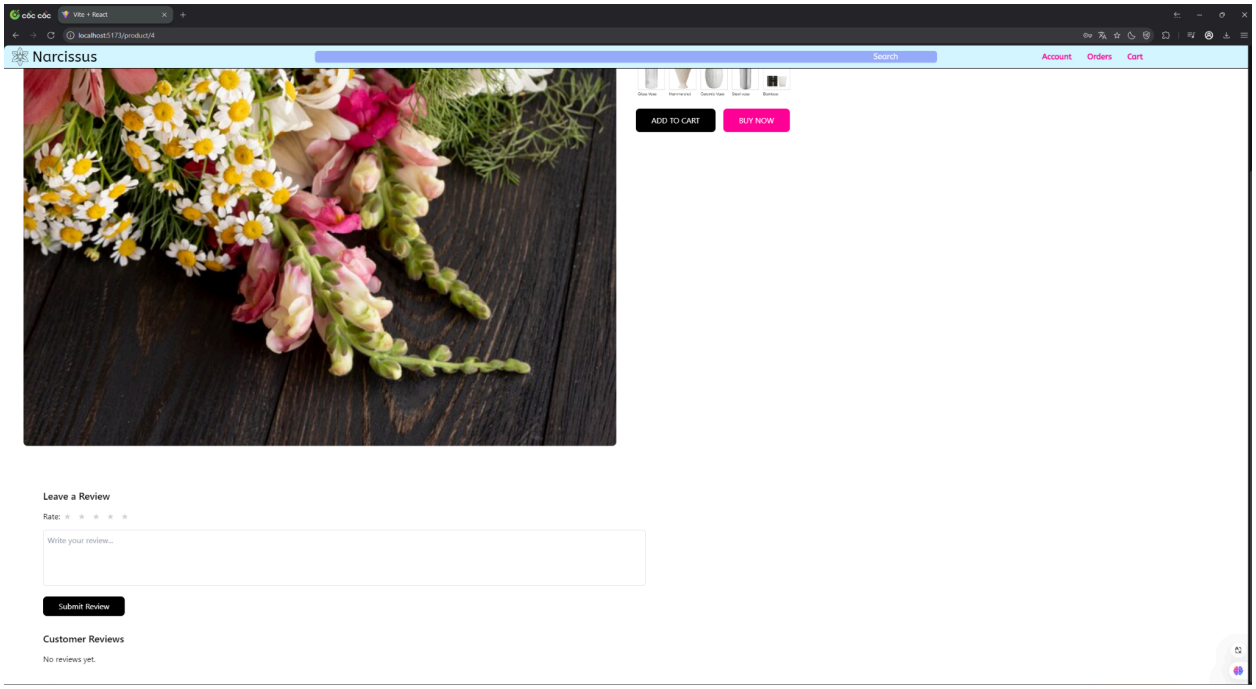

Narcissus
Không bảo mật | 74.226.216.170:5173/userdetails
Narcissus
Search
Account
Orders
Cart
User Details
Username
nghia
Email
ITCSIU21207@student.hcmiu.edu.vn
Address
1234
PhoneNumber
Logout
Edit

This is the **Account Creation Page**, where only customer roles can be created. Users must fill in all the required information and ensure that the email is not already in use. If the account creation fails, an error message will be displayed. If successful, a confirmation email will be sent, and the user will be redirected to the **Homepage**.

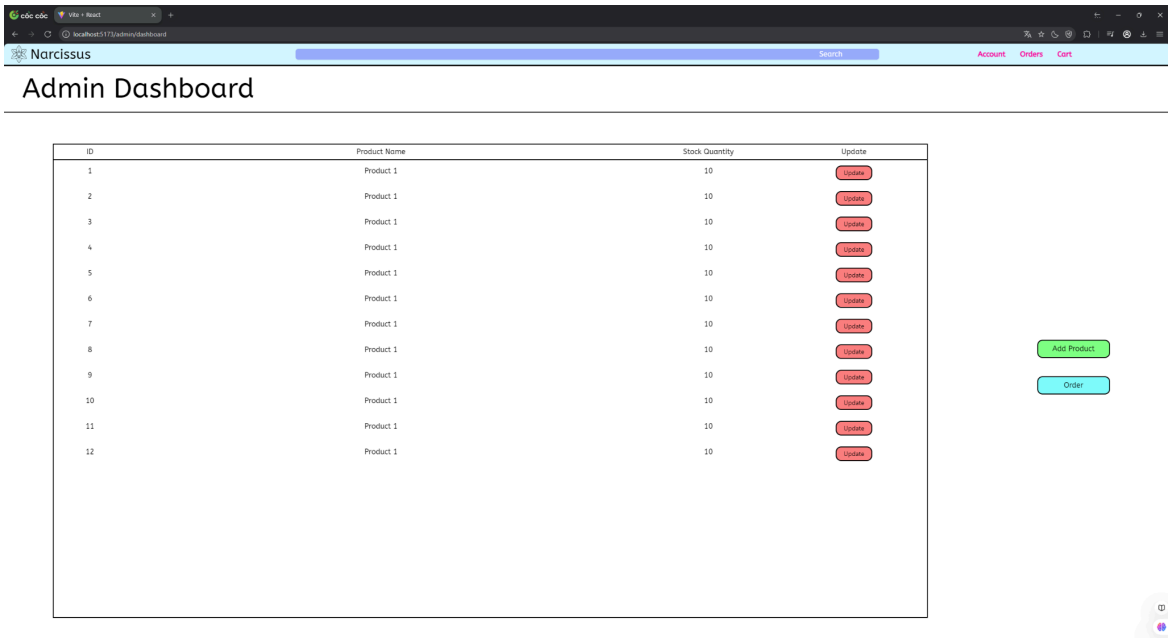


This is the **Product Detail Page**, which appears when clicking on a product. Here, users can adjust the quantity, **Add to Cart**, or **Buy Now**, as well as read and leave reviews. However, actions such as **Add to Cart**, **Buy Now**, and **Leave Review** require the user to be logged in.

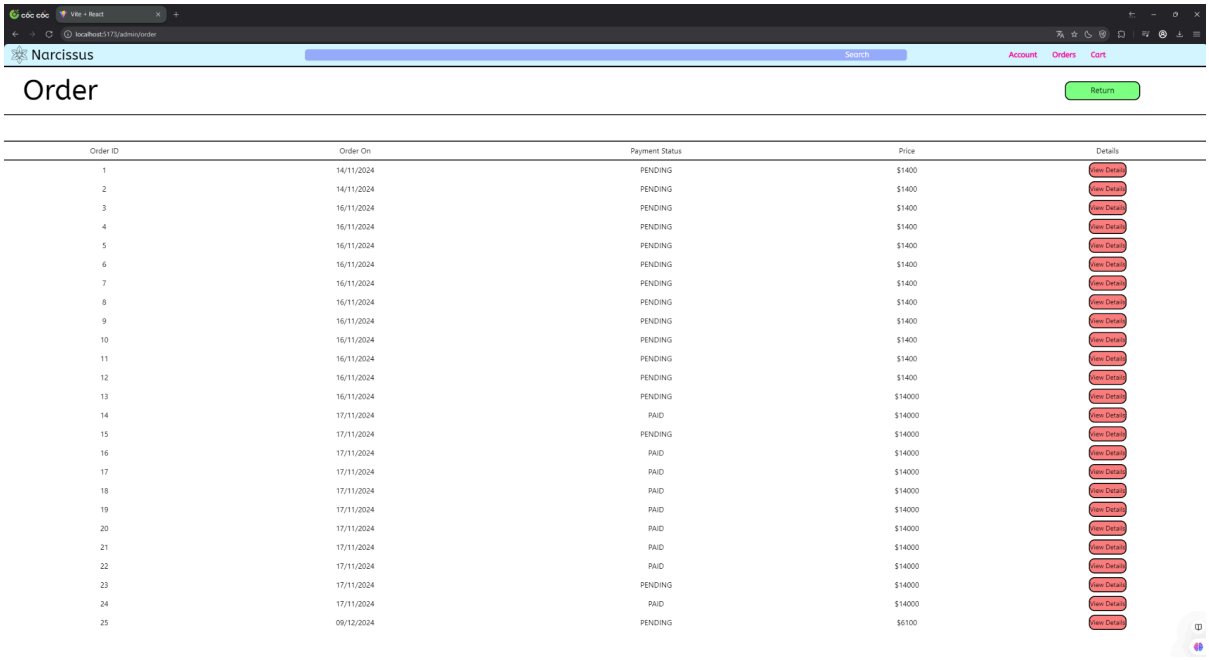




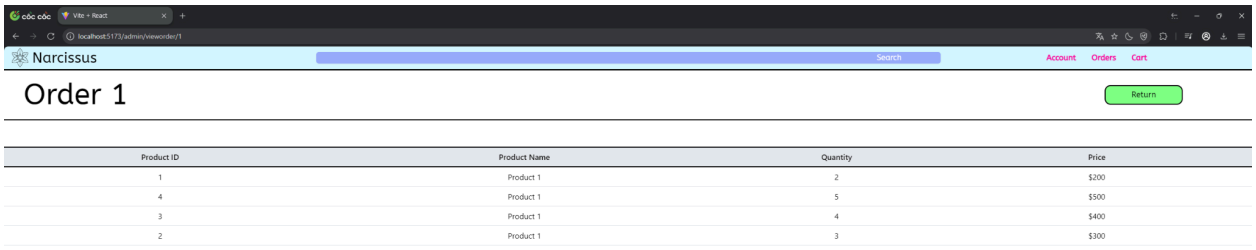
This is the **Admin Dashboard Page**, accessible only when logged in with an admin role. Users must have an admin role to enter this page. The dashboard provides an overview of product information and includes buttons for other actions such as **Update**, **Add Product**, and **Order**.



Clicking on **Order** in the Admin Dashboard will navigate to the **View Orders** page, which displays all completed orders. Access to this page requires an admin role. The page includes a **Return** button to go back to the previous page (**Admin Dashboard**) and a **View Details** button to see the details of each order.

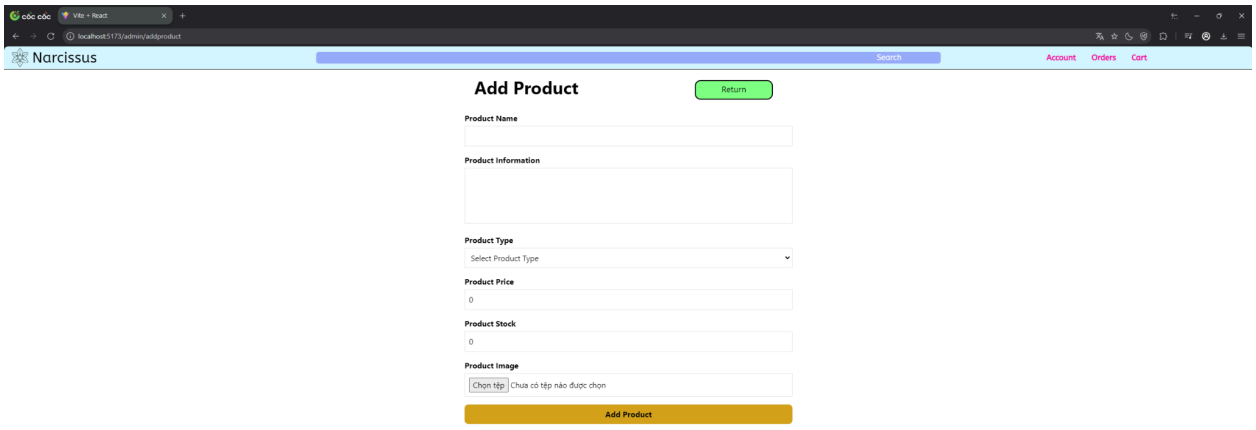


This is the **Order Detail Page**, which displays the specific products, quantities, and prices for each item in the order. Access to this page requires an admin role. It also includes a **Return** button to go back to the previous page (**Order**).



Product ID	Product Name	Quantity	Price
1	Product 1	2	$200
4	Product 1	5	$500
3	Product 1	4	$400
2	Product 1	3	$300

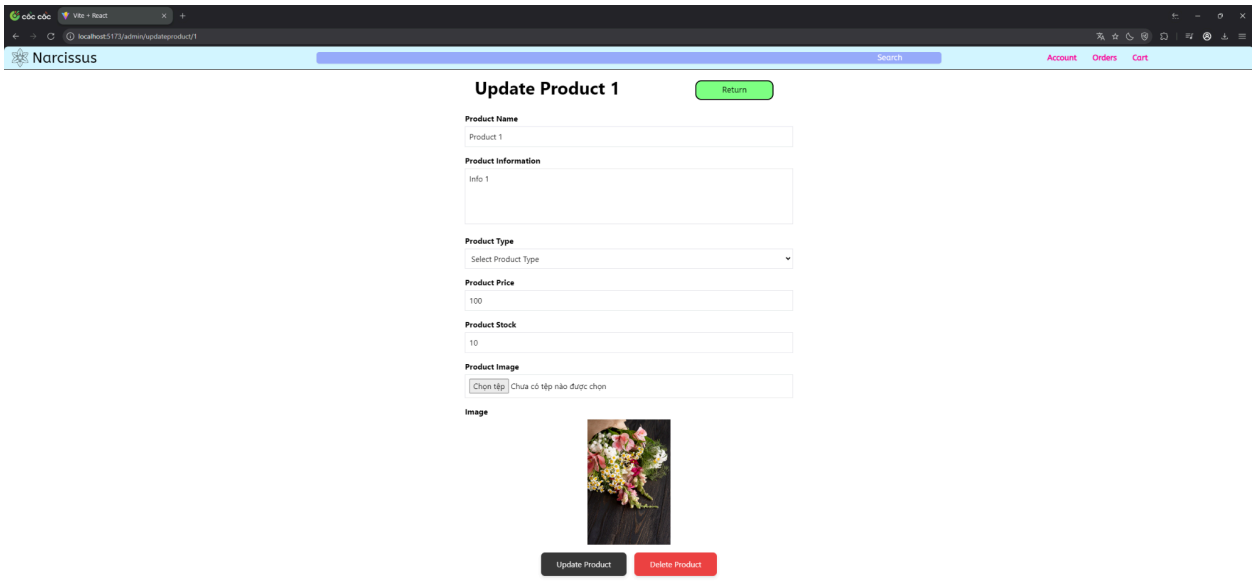
This is the **Add Product Page**, accessed by clicking the **Add Product** button on the **Admin Dashboard**. It allows the creation of a new product by filling in the required basic information. The **Add Product** button is used to add the product to the product list, after which the user is redirected back to the **Admin Dashboard**. Additionally, there is a **Return** button to navigate back to the previous page (**Admin Dashboard**).



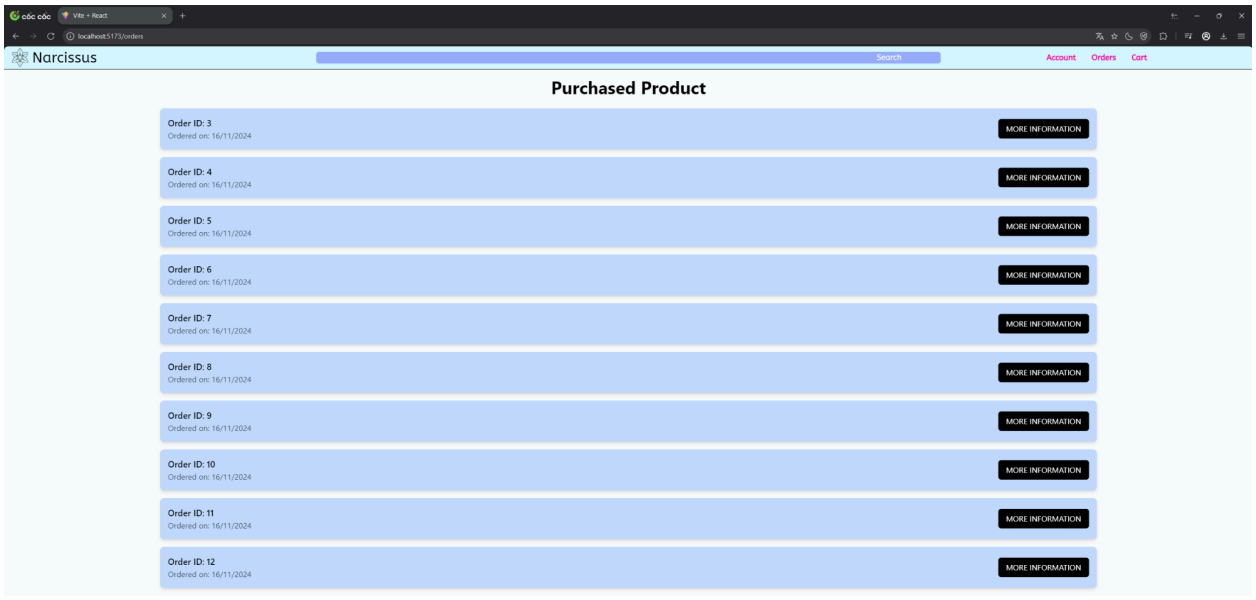
This is the **Update Product Page**, accessed by clicking the **Update Product** button on the **Admin Dashboard**. It allows users to:

- **Update Product**: Modify the product's details by filling in all the required fields (changes are mandatory and cannot be left blank).
- **Delete Product**: Remove the product from the list.
- **View Product Details**: Review the product's basic information.

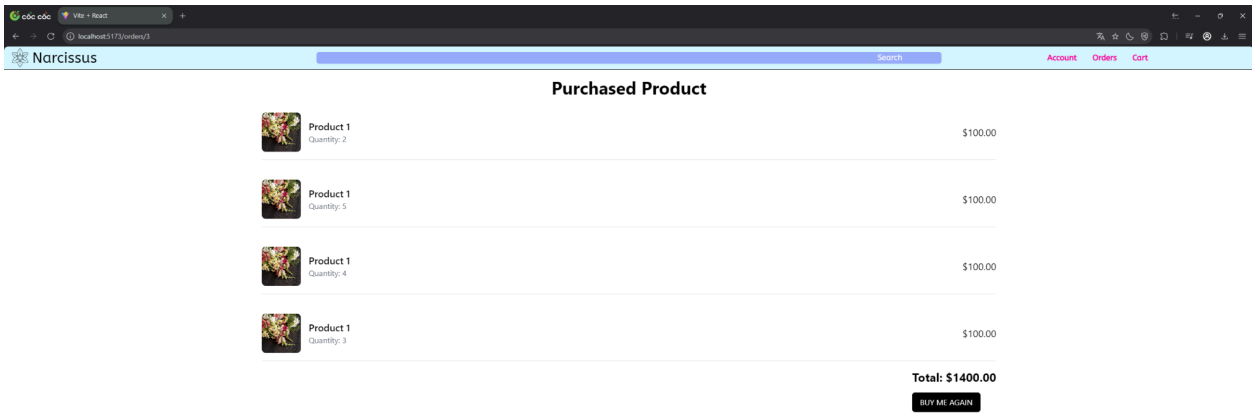
All three buttons—**Return**, **Delete Product**, and **Update Product**—redirect the user back to the **Admin Dashboard** after being clicked.



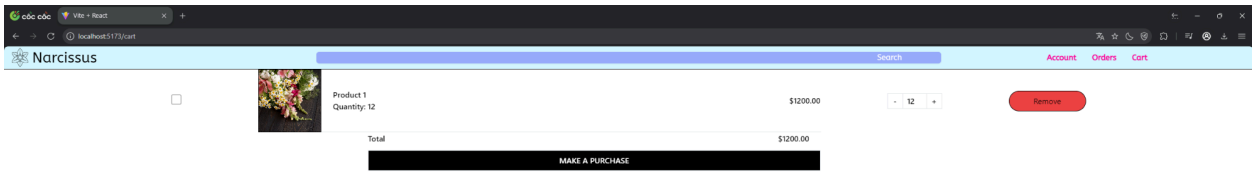
The **Purchased Products** page can be accessed by clicking on **Order** in the toolbar or after completing the payment. The page includes basic information about the order and a **More Information** button, allowing users to access and view detailed information about the order.



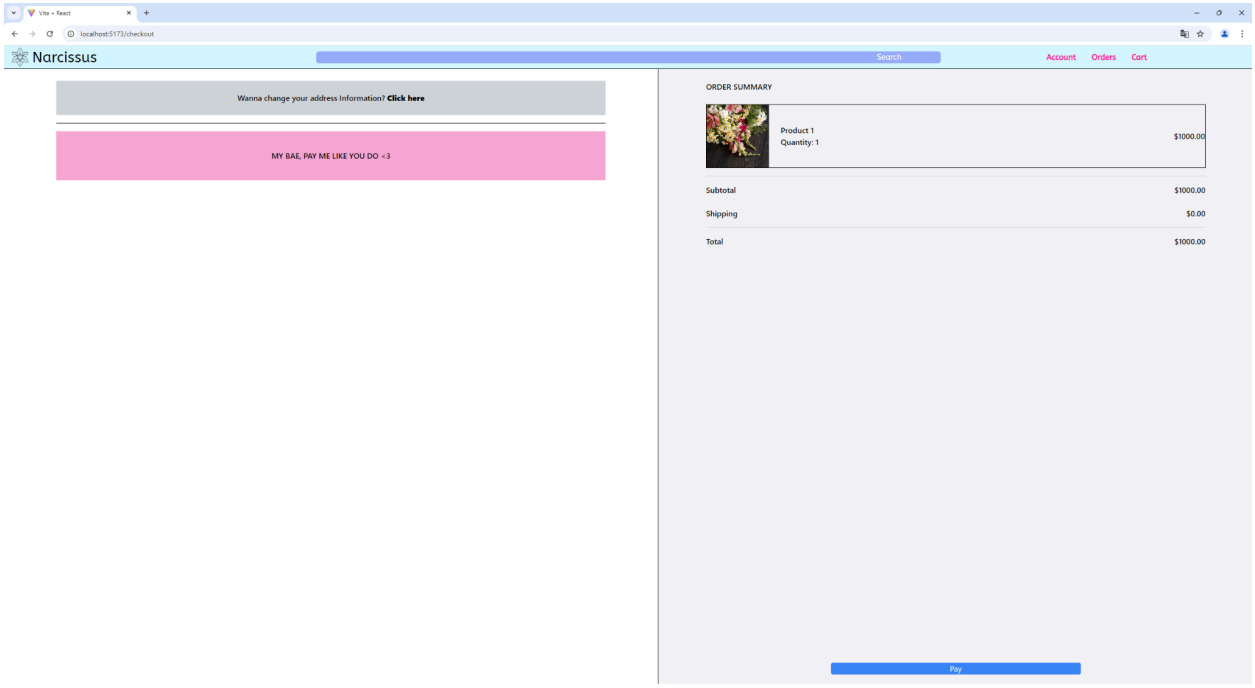
The **Purchased Product Details Page** is accessed by clicking **More Information** on the **Purchased Products** page. It displays detailed information about the products and includes a **Buy Me Again** button, allowing users to repurchase those products.



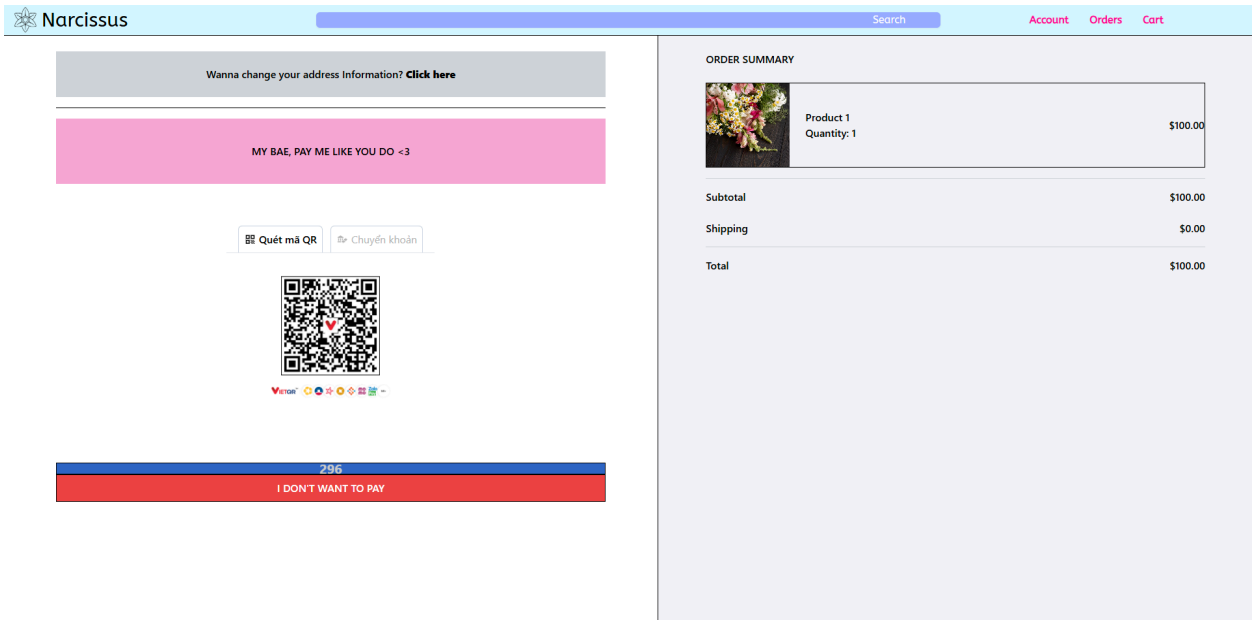
The **Cart Page** is accessed by clicking **Cart** on the toolbar. It displays information about the products along with their quantities, which were added to the cart on the **Product Page**. The page includes a **Remove** button primarily intended to remove items from the cart. To **Make a Purchase**, users must select at least one product. If no products are selected, a notification will appear reminding users to choose a product before proceeding to the **Check Out Page**.



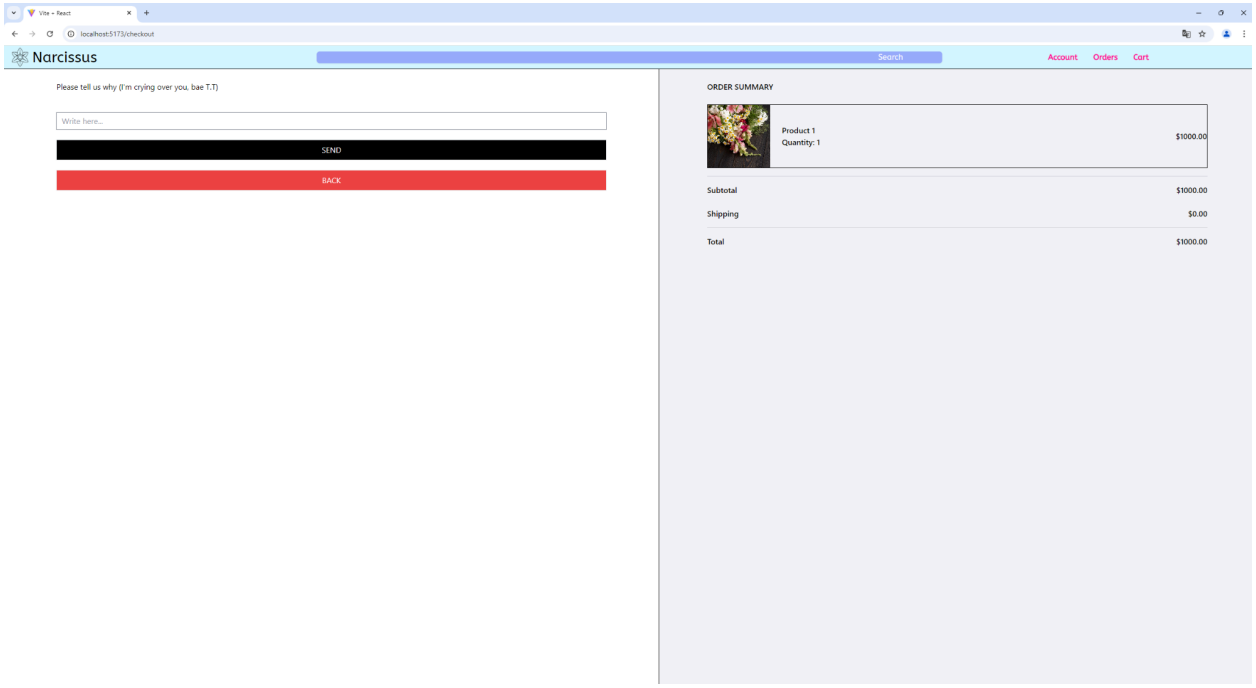
The **Checkout Page** is used to confirm order details. Once the user is ready to proceed with the purchase, they can click **Pay** to complete the payment.



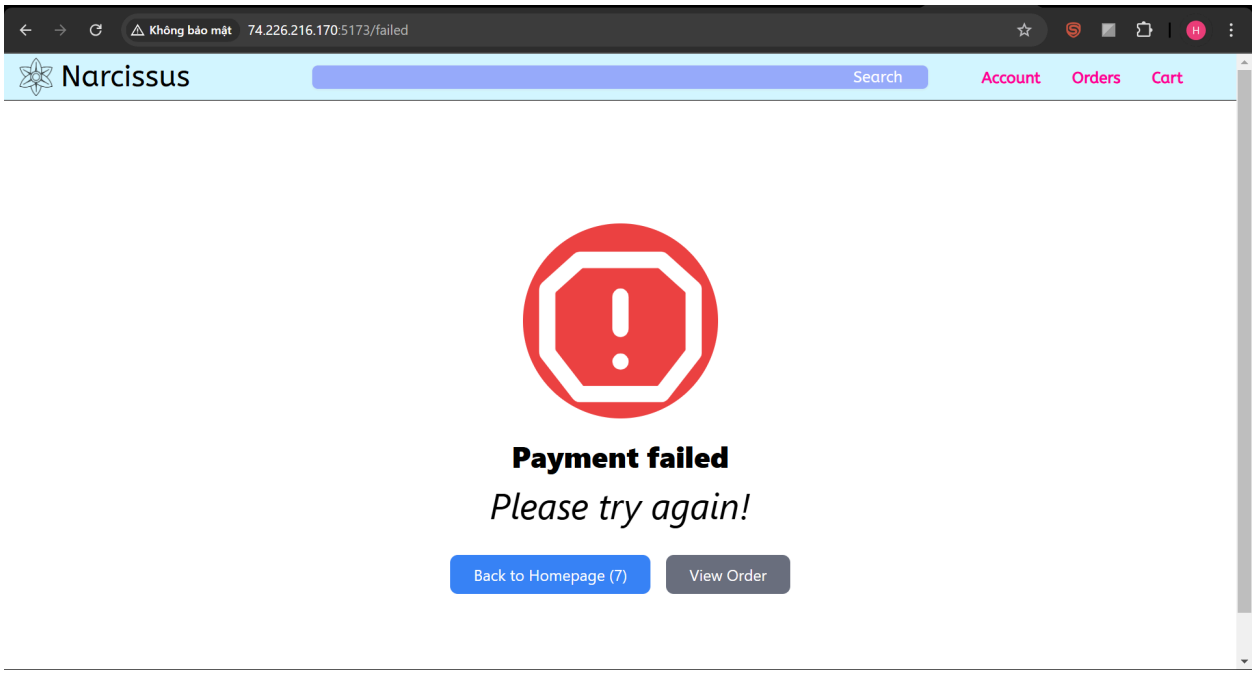
After clicking **Pay**, this page displays the **QR code** for making the payment along with a specific time limit to complete the transaction. If the user decides not to proceed with the purchase, there is an **I Don't Want to Pay** button to cancel the payment.

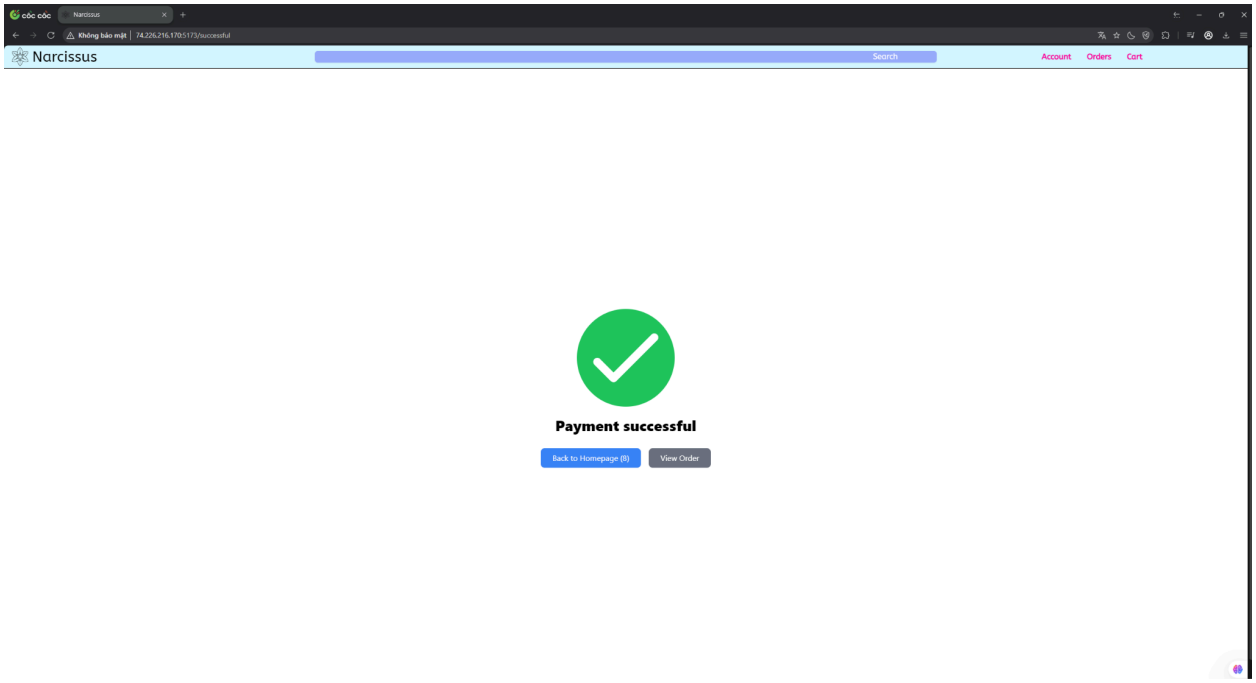


When clicking **I Don’t Want to Pay**, a page will appear that may or may not display additional information. From there, the user can click **Back** to continue the payment process or **Send** to submit the information and be redirected to the **Order Page**.



If payment is processed, there will be two possible statuses: successful or failed. After 10 seconds, you will be redirected to the **Homepage** automatically. Alternatively, you can click a button to either view your order or return to the homepage before the waiting time ends.





Chapter 4: DISCUSSION AND CONCLUSION

Developing our ecommerce flower website Narcissus has been an enriching journey that has allowed us to combine theoretical expertise with practical application. This project challenged us to integrate various aspects of web development, including user experience design, security measures, and effective data management, in a cohesive and responsive platform

By providing an easy-to-use and secure system that connects buyers and sellers, Narcissus sets the standard for trust in the online flower business. Its unique role as a trusted intermediary builds user trust and satisfaction, making it different from traditional sales channels. Furthermore, our focus on optimizing the supply chain for cost efficiency reflects a forward-looking vision, which enables flexibility and future growth opportunities for the platform

Through this role, our team has developed important skills such as coding, systems analysis, and project organization. We also strengthened our teamwork and communication capabilities, which proved essential in delivering a large-scale program like Narcisse. This hands-on experience gave us valuable insight into the software development lifecycle, preparing us to tackle more complex projects in the future

In conclusion, Narcissus is more than just an efficient e-commerce platform; It represents the culmination of our commitment, creativity and engineering. This project has been a cornerstone of our journey as software developers, providing the tools and confidence to excel in the evolving web application development landscape.

Chapter 5: REFERENCES

https://www.toolsqa.com/agile/scrum/sprint/
https://bap-software.net/en/knowledge/mvc-model/
https://www.bezkoder.com/react-spring-boot-crud/
https://payos.vn/docs
https://www.w3schools.com/