

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
School of Information and communications technology

Software Design Document

Version 1.3

AIMS PROJECT

Subject: ITSS Software Development

Group 01

Mai Đức An – 20210008

Vũ Đức An – 20215174

Đỗ Hoàng Anh – 20200012

Nguyễn Hải Anh – 20194725

Nguyễn Quốc Anh – 20194726

Hanoi, <June, 2024>

Table of Contents

1 Introduction.....	4
1.1 Objective.....	4
1.2 Scope.....	4
1.3 Glossary.....	4
1.4 References.....	5
2 Overall Description.....	6
2.1 General Overview.....	6
2.2 Assumptions/Constraints/Risks.....	6
2.2.1 Assumptions.....	6
2.2.2 Constraints.....	6
2.2.3 Risks.....	8
3 System Architecture and Architecture Design.....	10
3.1 Architectural Patterns.....	10
A separate frontend-backend architecture offers better scalability, flexibility, development efficiency, user experience, and promotes an API-first approach, making it a preferred choice over MVC for many modern applications.....	11
3.2 Interaction Diagrams.....	11
UseCase Diagram.....	11
Sequence Diagram for UC “Add product to cart”.....	12
Sequence Diagram for UC “Change Block Status”.....	13
Sequence Diagram for UC “Change Password”.....	13
Sequence Diagram for UC “Change User Role”.....	13
Sequence Diagram for UC “Create User”.....	14
Sequence Diagram for UC “CRUD Product”.....	14
Sequence Diagram for UC “Login”.....	14
Sequence Diagram for UC “Pay Order”.....	15
Sequence Diagram for UC “Place Order”.....	15
Sequence Diagram for UC “Reset Password”.....	16
Sequence Diagram for UC “Update Order Status”.....	17
Sequence Diagram for UC “Update Product Price”.....	17
3.3 Analysis Class Diagrams.....	18
Cart Analysis Class Diagram.....	18
Product Analysis Class Diagram.....	19
User Analysis Class Diagram.....	20
Order Analysis Class Diagram.....	21
3.4 Unified Analysis Class Diagram.....	21
4 Detailed Design.....	22
4.1 User Interface Design.....	22

4.1.1 Screen Transition Diagrams.....	22
4.1.2 Screen Specifications.....	23
4.2 Data Modeling.....	34
4.2.1 Conceptual Data Modeling.....	34
4.2.2 Database Design.....	34
4.3 Class Design.....	38
4.3.1 General Class Diagram.....	38
4.3.2 Class Diagrams.....	39
5 Design Considerations.....	42
5.1 Goals and Guidelines.....	42
5.2 Architectural Strategies.....	42
5.3 Coupling and Cohesion.....	43
Coupling.....	43
Cohesion.....	44
5.4 Design Principles.....	44
7.1 Design Patterns.....	45

1 Introduction

1.1 Objective

This document serves as a software design development (SDD) for the Internet Media Store System, which is designed to provide an online platform for customers to purchase digital media. The objective of this document is to give an outline description of the system. The document also provides a thorough understanding of the purpose of the system as well as the features and functionalities of the system. The intended audience for this document includes stakeholders and software developers.

1.2 Scope

The AIMS system is a system that manages a store and automates the management of products, orders, and users.

1.3 Glossary

<i>No</i>	<i>Term</i>	<i>Explanation</i>	<i>Example</i>	<i>Note</i>
1	invoice	An invoice for a customer of an online media store is a document that outlines the details of a transaction between the store and the customer. It typically includes the items or services purchased, the price of each item or service, any applicable taxes or fees, the total amount due, and payment instructions. The invoice serves as a record of the transaction and is used by the customer to make payment and by the store to track sales and inventory.		
2	CRUD operations	CRUD is an acronym that stands for Create, Read, Update and Delete		

<i>No</i>	<i>Term</i>	<i>Explanation</i>	<i>Example</i>	<i>Note</i>
3.	Transaction ID	A unique identifier assigned to each payment transaction		
4.	VNPay	A secure online service that authorizes payments between customers and businesses		
5.	Input Validation	The process of checking user input for errors or invalid data	AIMS will check the delivery information submitted by customers and ask them to update if there are any invalid entries	This prevents errors during order processing.

1.4 References

[1] *Kết nối Cổng thanh toán VNPAY · Cổng thanh toán VNPAY*,

<https://sandbox.vnpayment.vn/apis/docs/thanh-toan-pay/pay.html>.

2 Overall Description

2.1 General Overview

The software system includes four actors including Customer, Admin, Product Manager and VNPay.

- Customer is the person who visits the AIMS website to browse, search and purchase media products.*
- Administrator is the person responsible for managing the AIMS system, updating products information, managing orders and customers.*
- Product Manager is the person who uses AIMS software to manage and sell their products.*
- VNPay is the service provider that processes payment made by the customers through the AIMS system.*

2.2 Assumptions/Constraints/Risks

2.2.1 Assumptions

AIMS is a desktop software designed for online sales purposes. For AIMS to function properly, several requirements must be met. First, since AIMS is desktop software, it needs to be compatible with popular operating systems such as Windows, MacOS, and Linux. The system must have database management software to store and manage data related to sales activities. Furthermore, the system must always be connected to the internet to enable invoice connections and payments through VNPay. This software is designed for users who are already familiar with using online shopping software, as the workflow is designed similarly to other online sales systems. The desktop software is also assumed to have modest hardware requirements, allowing it to run on a wide range of consumer-grade desktop and laptop computers.

2.2.2 Constraints

The AIMS project has several global limitations and constraints that significantly impact the design of its hardware, software, and communication systems:

- Hardware or Software Environment: The AIMS software operates on desktop environments and must support 24/7 operation, handling up to 1,000*

concurrent users. It requires robust hardware to maintain performance without significant degradation.

- *End-User Environment: The software is designed for ease of use, allowing new users to familiarize themselves quickly. It supports product managers and customers, necessitating an intuitive user interface.*
- *Availability or Volatility of Resources: The software must maintain high availability, operating continuously for 300 hours and resuming normal operations within 1 hour after an incident.*
- *Standards Compliance: The software must comply with standards for e-commerce operations and data handling, including secure transaction processing and data privacy regulations.*
- *Interoperability Requirements: The AIMS system must interact seamlessly with external systems like VNPay for payment processing, adhering to specified APIs and protocols.*
- *Interface/Protocol Requirements: The software uses specific interfaces and protocols for payment processing, requiring integration with VNPay's payment API, query, and refund API specifications.*
- *Licensing Requirements: Licensing constraints for third-party software and libraries used within the project must be adhered to, ensuring all components are legally compliant.*
- *Data Repository and Distribution Requirements: The system requires a reliable database to store product, user, and transaction data. It must support efficient data distribution and retrieval to handle simultaneous operations by multiple users.*
- *Security Requirements: Robust security measures are essential, including user authentication, authorization, and secure handling of sensitive information like payment details. The system must prevent unauthorized access and ensure data integrity.*
- *Memory or Other Capacity Limitations: The software must manage memory efficiently to handle up to 1,000 simultaneous users without performance degradation. It should optimize resource usage to ensure smooth operation.*
- *Performance Requirements: The system must maintain a maximum response time of 2 seconds under normal conditions and 5 seconds during peak hours, ensuring a responsive user experience.*
- *Network Communications: Reliable network communication is vital for real-time operations, particularly for transactions and data synchronization between clients and the server.*
- *Verification and Validation Requirements (Testing): Comprehensive testing is required to ensure the system meets all functional and non-functional*

requirements. This includes performance testing, security testing, and integration testing with external systems.

- *Other Means of Addressing Quality Goals: Continuous monitoring and maintenance are necessary to address quality goals, including performance optimization and security updates.*

These constraints collectively shape the design and implementation of the AIMS system, ensuring it meets user needs while adhering to technical and regulatory standards.

2.2.3 Risks

Performance Degradation

- *Risk: The system may experience performance issues under high load, especially when handling up to 1,000 concurrent users.*
- *Mitigation Strategy: Implement load balancing and optimize database queries. Conduct thorough performance testing to identify and resolve bottlenecks. Use caching mechanisms to reduce database load.*

Data Security Breach

- *Risk: Unauthorized access to sensitive data, including user and payment information, could occur.*
- *Mitigation Strategy: Implement strong encryption for data storage and transmission. Use robust authentication and authorization mechanisms. Regularly update and patch software to protect against vulnerabilities.*

System Downtime

- *Risk: The system could fail to operate continuously for 300 hours or might not resume normal operation within 1 hour after an incident.*
- *Mitigation Strategy: Employ redundant systems and backup strategies. Implement automated failover processes. Conduct regular disaster recovery drills to ensure rapid restoration of services.*

Integration Issues

- *Risk: Integration with external systems, such as VNPay for payment processing, might face compatibility issues or downtime.*
- *Mitigation Strategy: Establish clear API specifications and test integrations thoroughly. Maintain communication with third-party providers for timely updates and support. Implement fallback procedures for payment processing failures.*

Compliance Violations

- *Risk: Failure to comply with relevant standards and regulations could lead to legal and financial penalties.*
- *Mitigation Strategy: Stay updated on regulatory requirements and ensure the system design adheres to these standards. Conduct regular compliance audits and seek legal consultation to ensure ongoing compliance.*

Scalability Limitations

- *Risk: The system might not scale effectively to accommodate future growth in user base or functionality.*
- *Mitigation Strategy: Design the system with scalability in mind, using microservices architecture and cloud-based solutions. Regularly review and adjust the architecture to meet growing demands.*

Data Loss

- *Risk: Critical data might be lost due to hardware failure, software bugs, or cyber-attacks.*
- *Mitigation Strategy: Implement robust backup and recovery procedures. Use redundant storage solutions and regularly test data restoration processes. Employ intrusion detection systems to identify and mitigate cyber threats.*

User Experience Issues

- *Risk: Users might find the software difficult to use, leading to dissatisfaction and reduced adoption.*
- *Mitigation Strategy: Conduct user testing and gather feedback to improve the user interface and experience. Provide comprehensive user training and support. Continuously iterate on the design based on user feedback.*

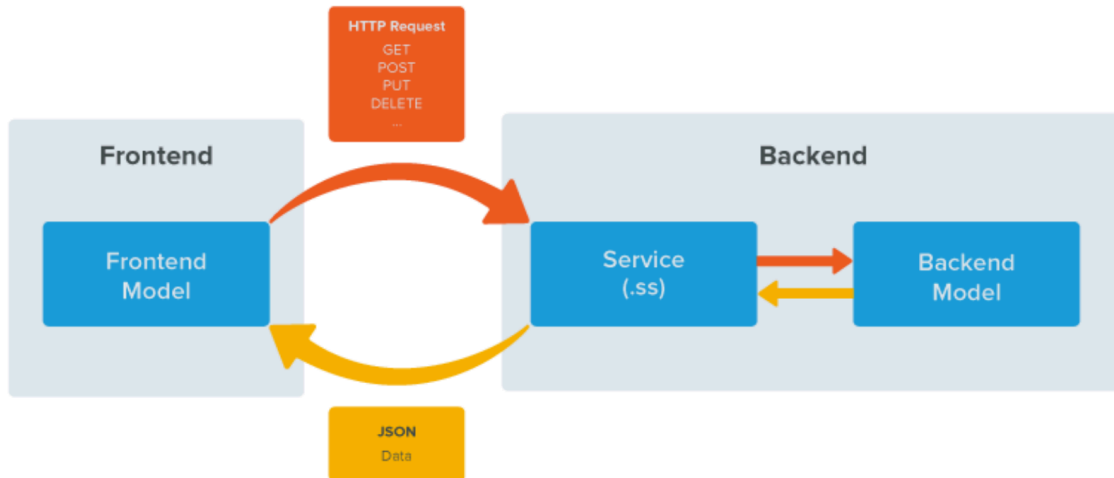
Financial Transaction Failures

- *Risk: Errors in the payment processing system could result in failed or incorrect transactions.*
- *Mitigation Strategy: Implement thorough testing for all payment processes. Use reliable and tested payment gateways. Provide clear instructions and support for users during the payment process.*

By addressing these risks with targeted mitigation strategies, the AIMS system can achieve higher reliability, security, and user satisfaction, ensuring its success in the competitive e-commerce landscape.

3 System Architecture and Architecture Design

3.1 Architectural Patterns



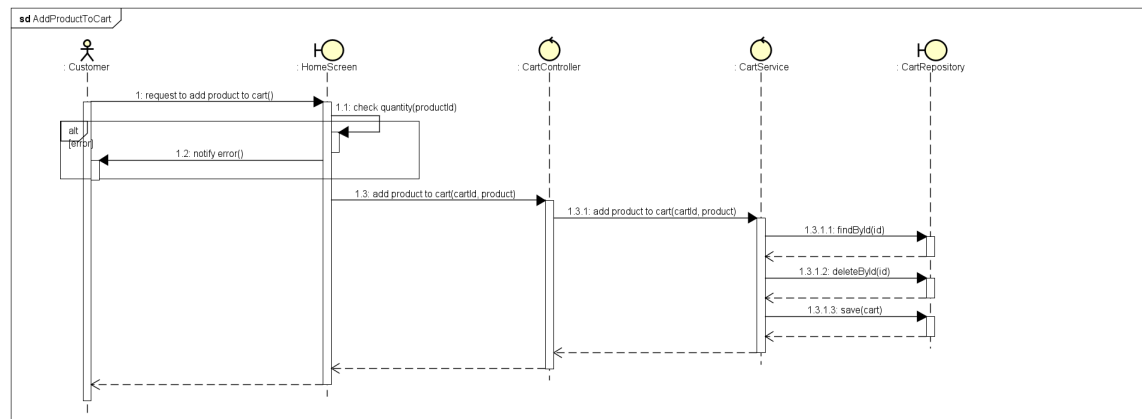
A separate frontend-backend architecture offers better scalability, flexibility, development efficiency, user experience, and promotes an API-first approach, making it a preferred choice over MVC for many modern applications.

3.2 Interaction Diagrams

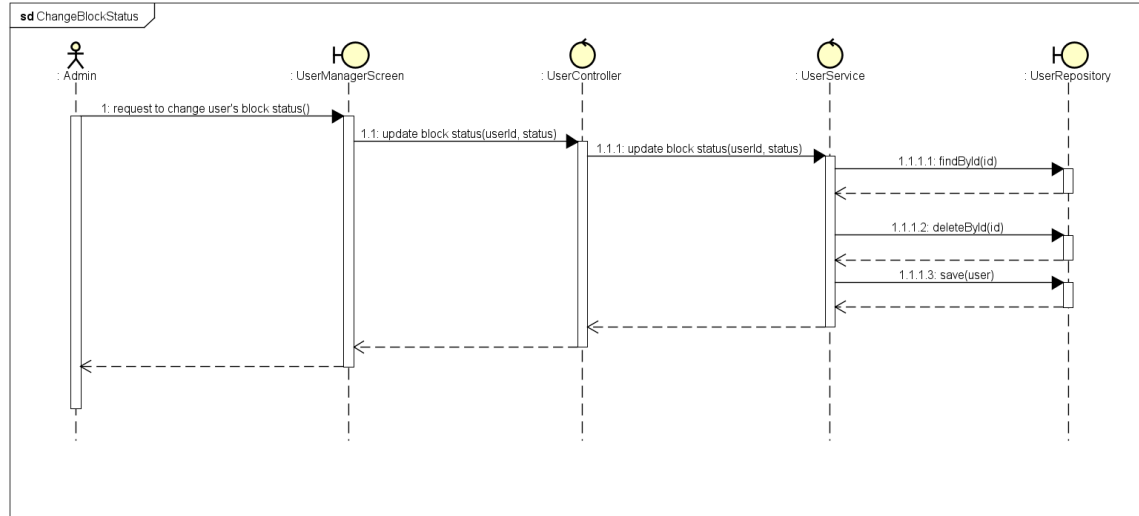
UseCase Diagram



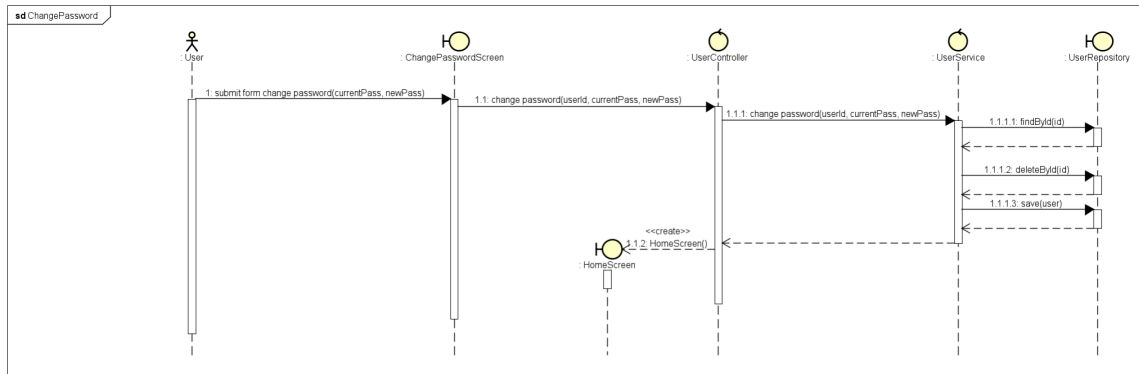
Sequence Diagram for UC “Add product to cart”



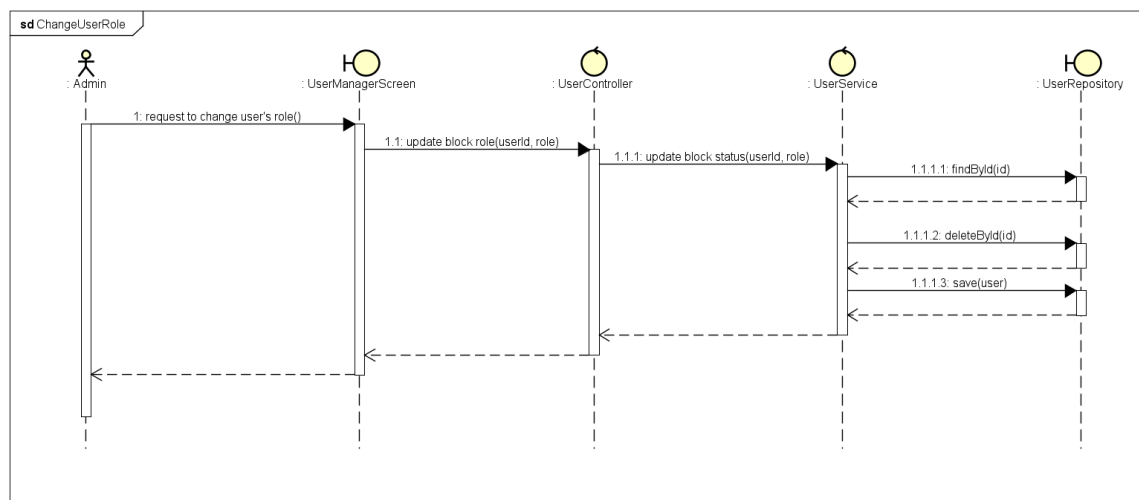
Sequence Diagram for UC “Change Block Status“



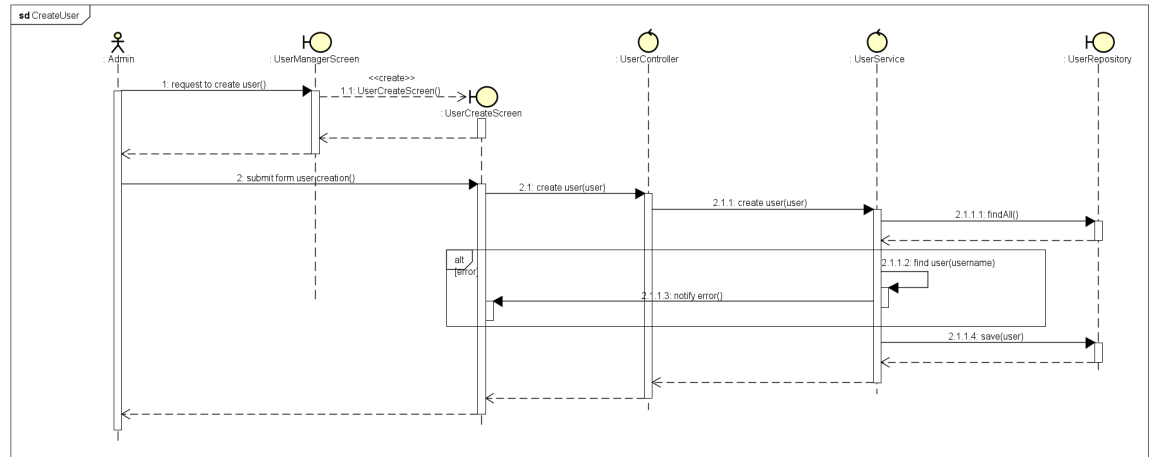
Sequence Diagram for UC “Change Password“



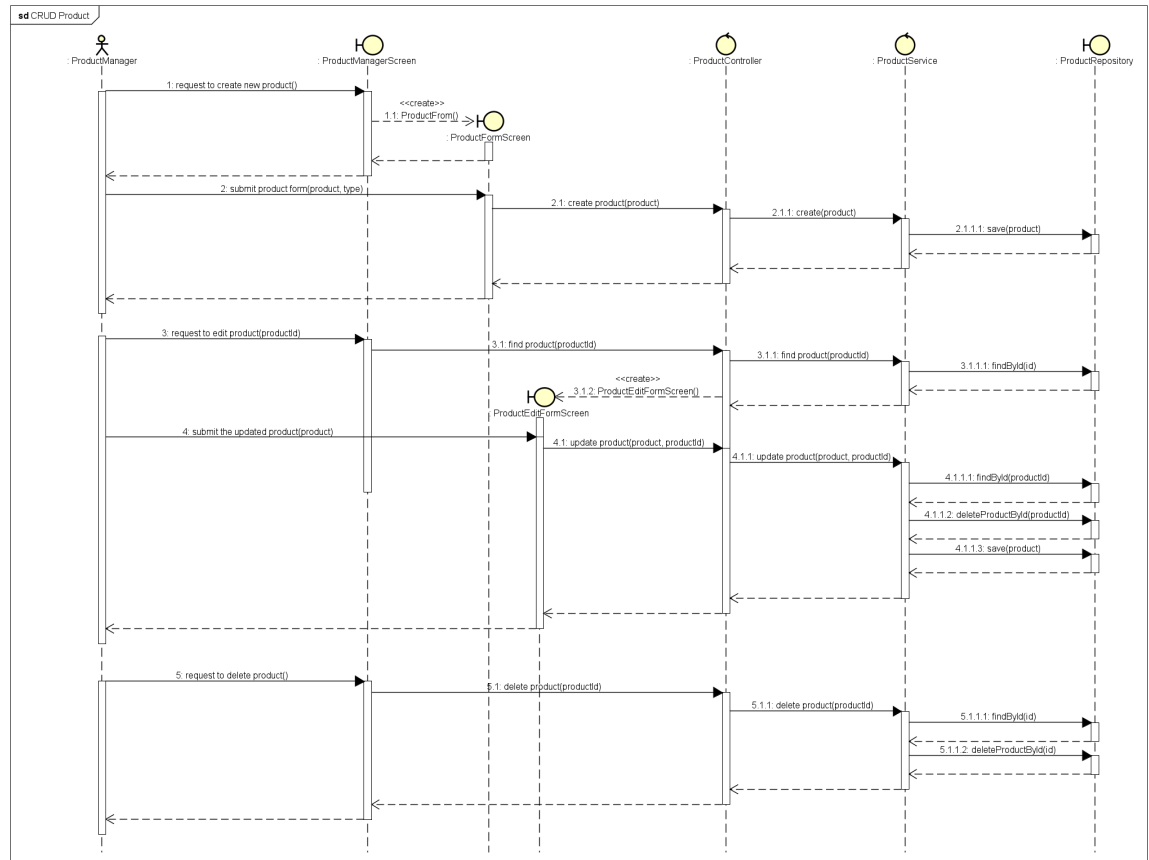
Sequence Diagram for UC “Change User Role“



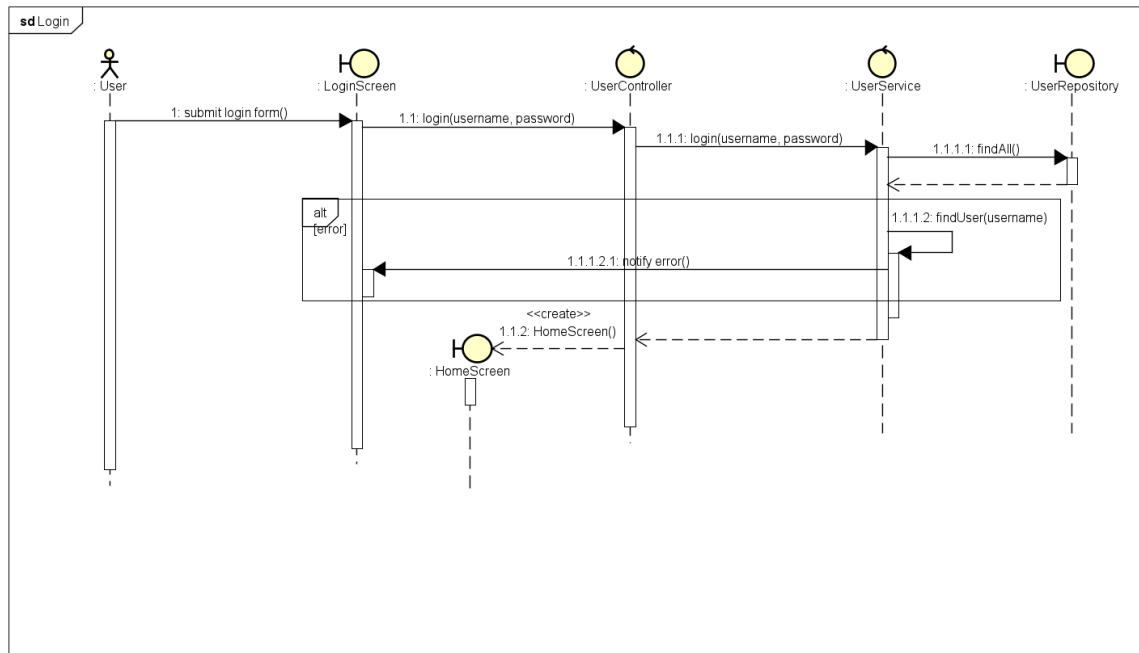
Sequence Diagram for UC “Create User”



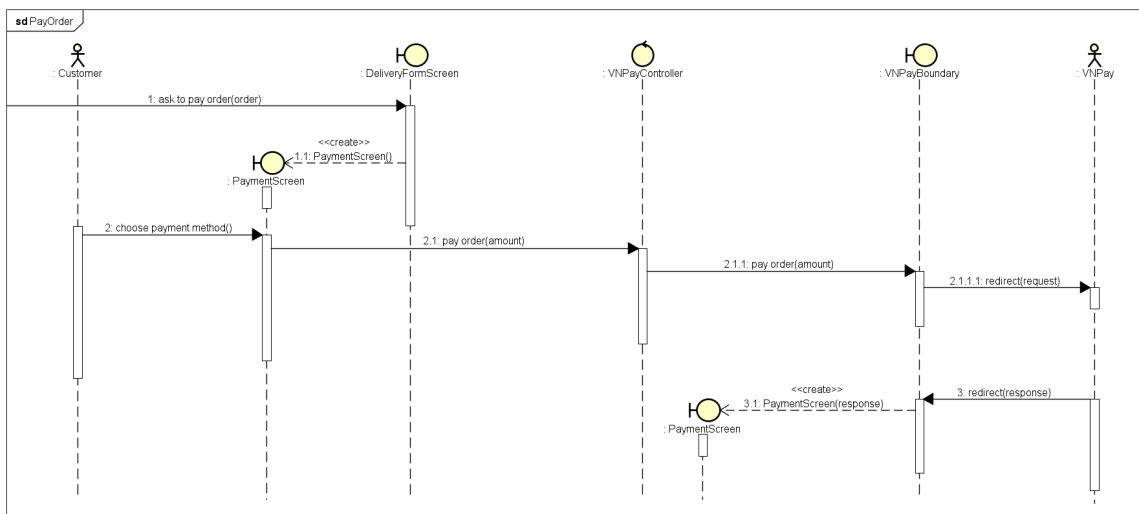
Sequence Diagram for UC “CRUD Product”



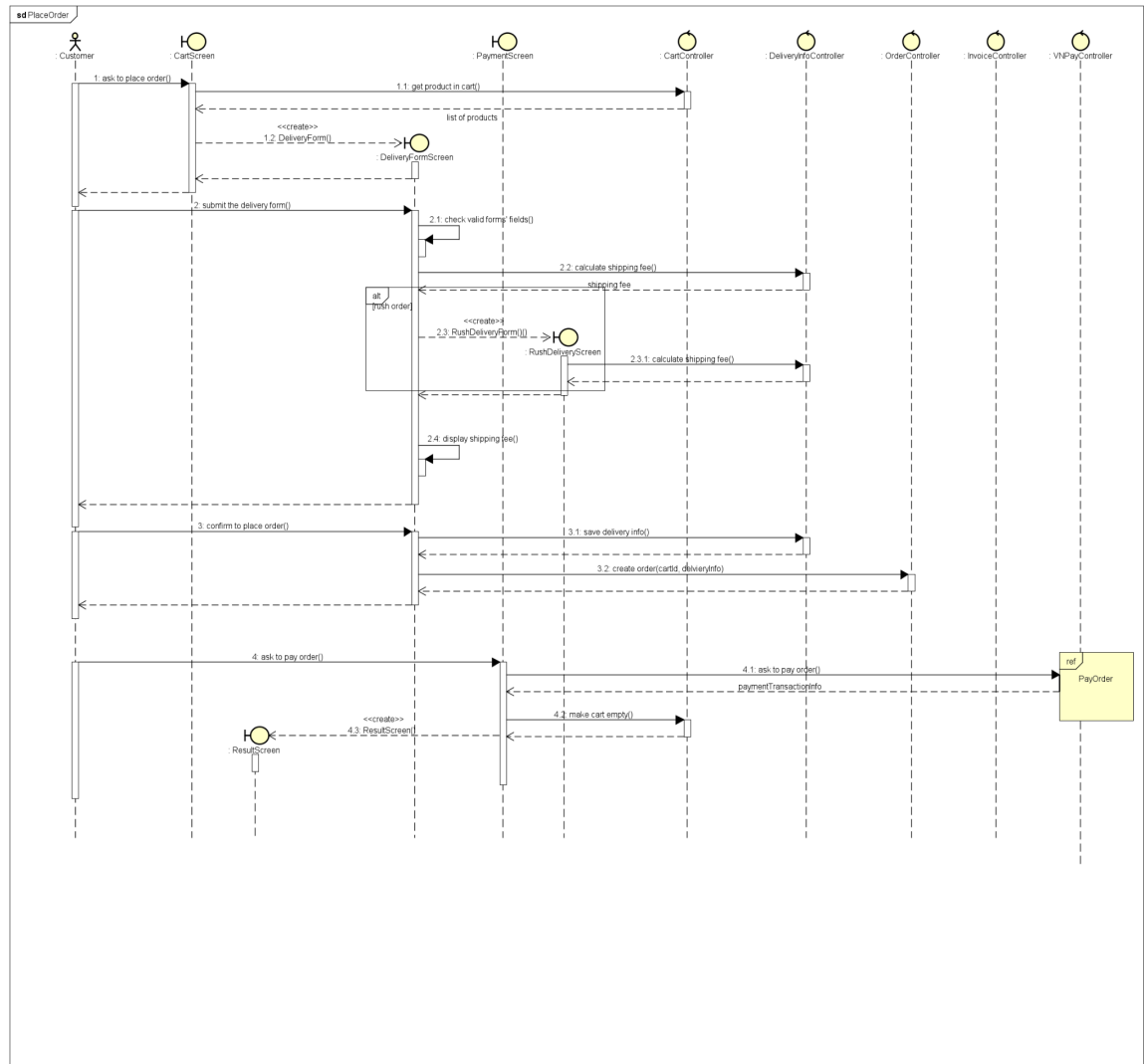
Sequence Diagram for UC “Login”



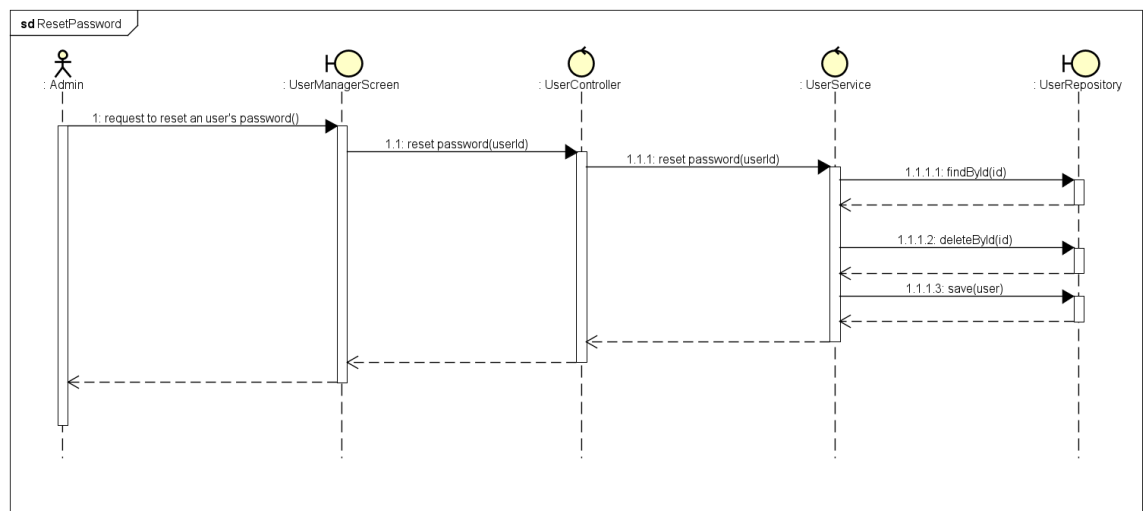
Sequence Diagram for UC “Pay Order“



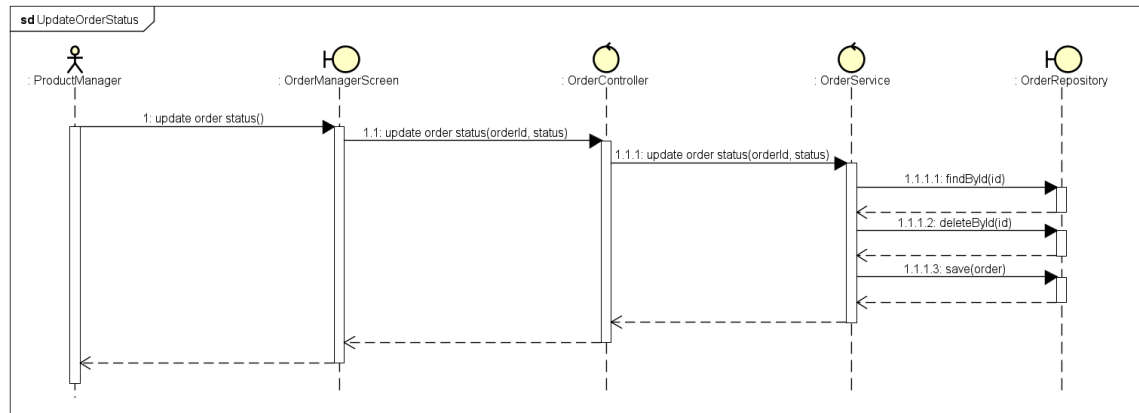
Sequence Diagram for UC “Place Order“



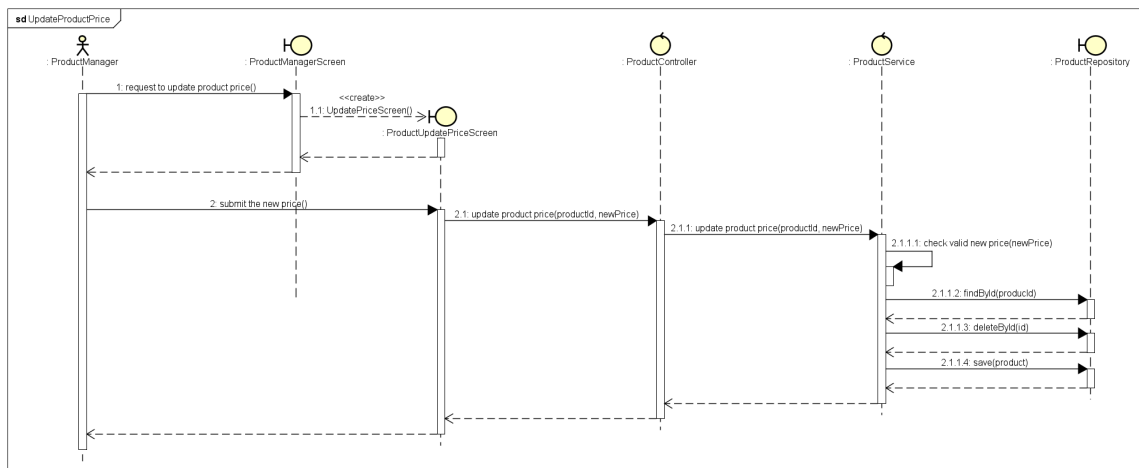
Sequence Diagram for UC “Reset Password“



Sequence Diagram for UC “Update Order Status”

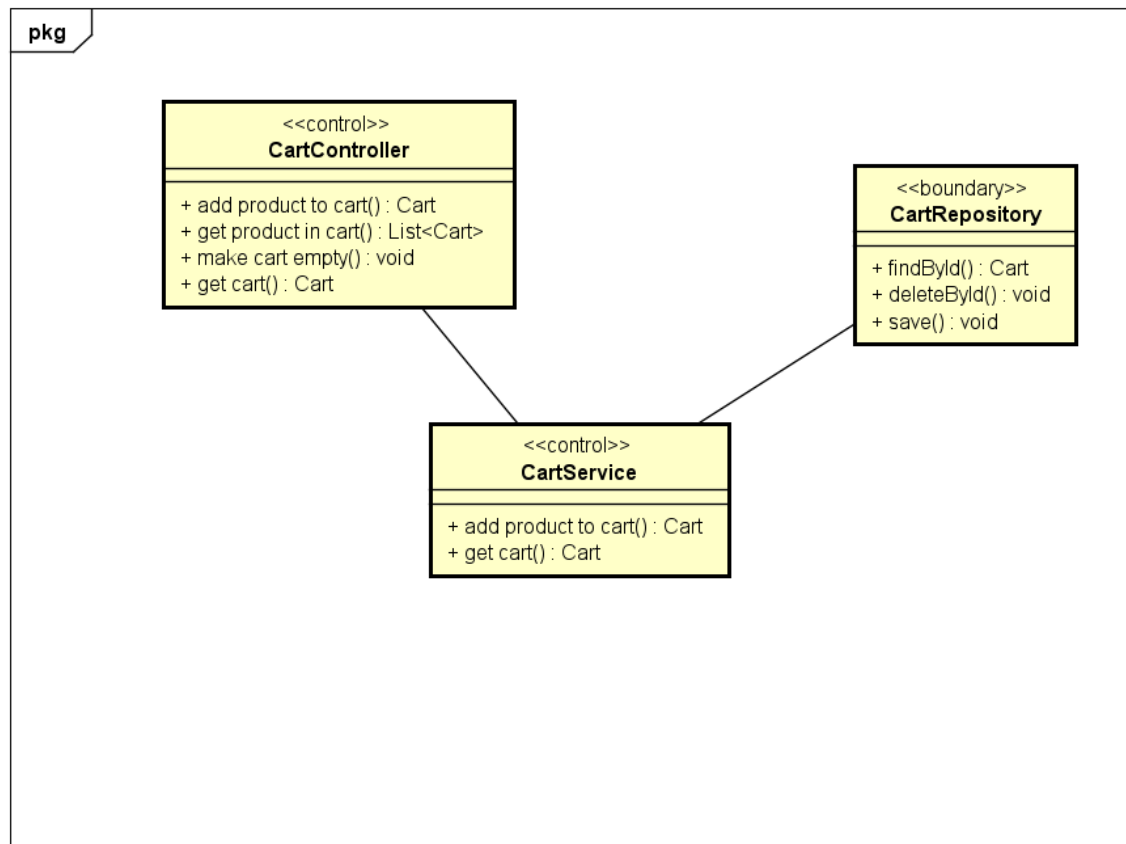


Sequence Diagram for UC “Update Product Price”

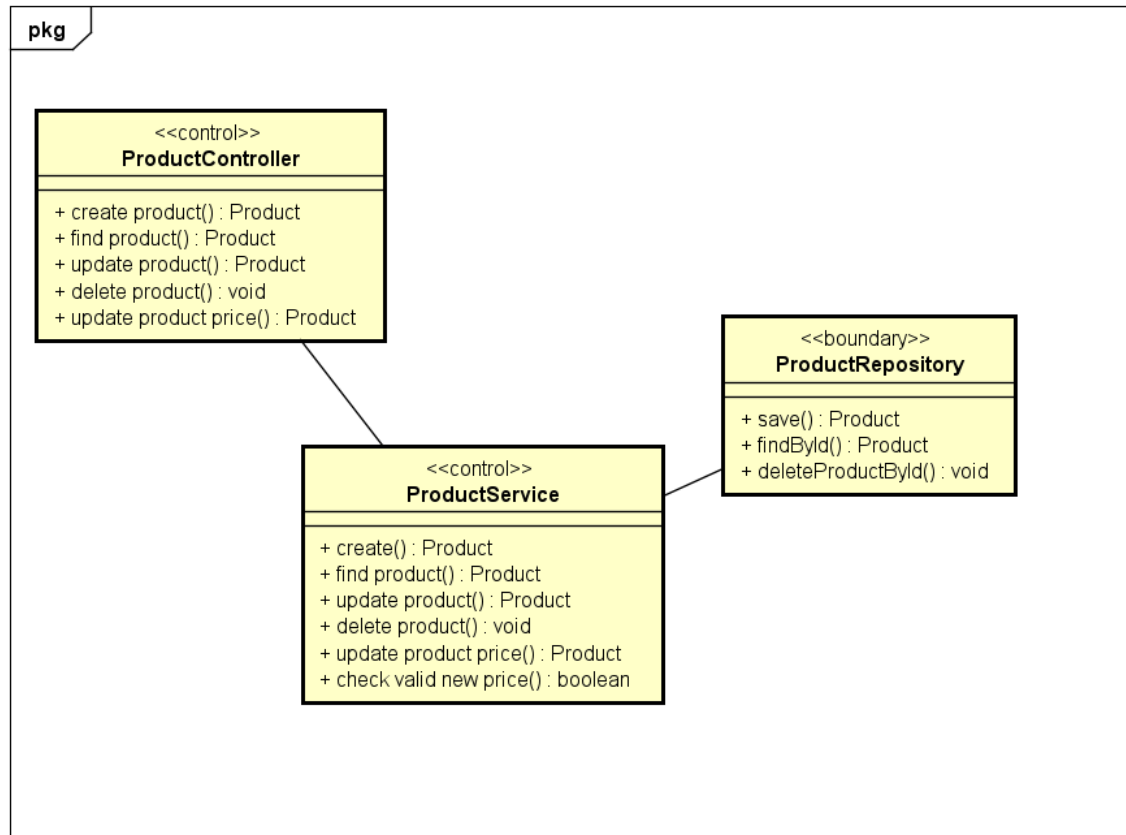


3.3 Analysis Class Diagrams

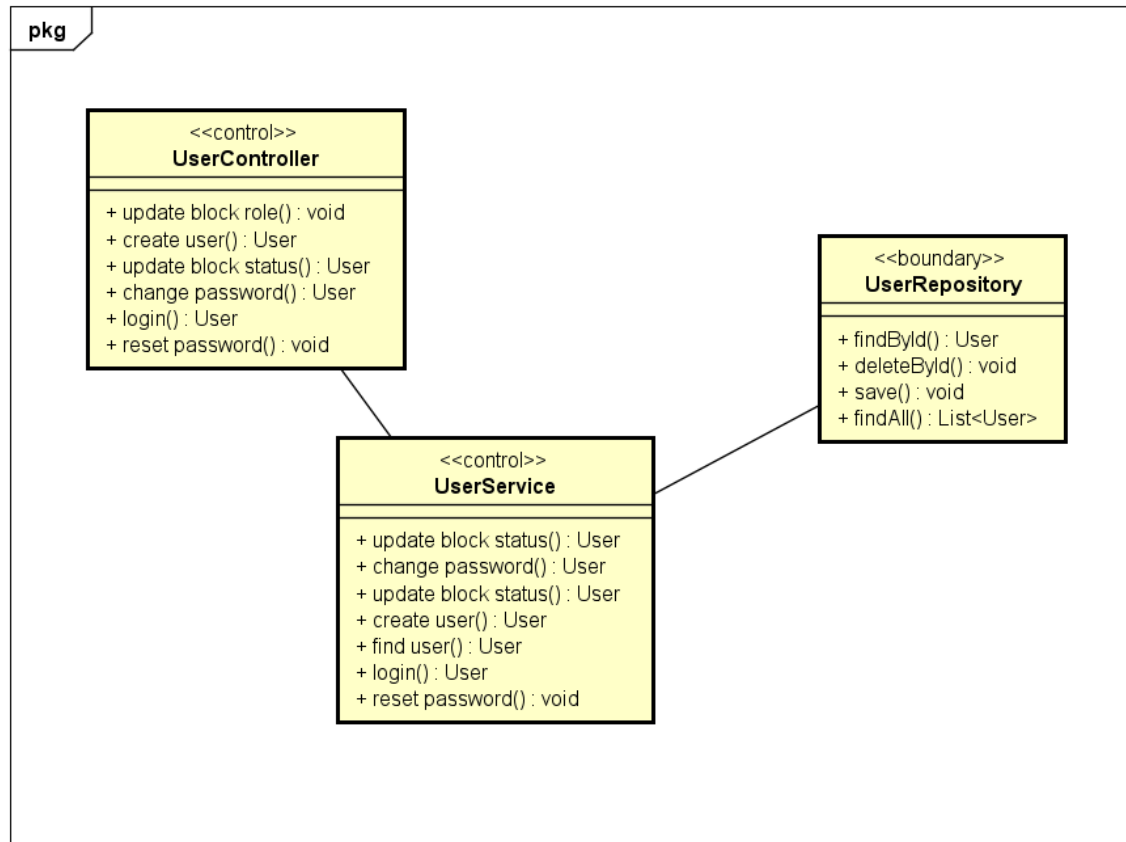
Cart Analysis Class Diagram



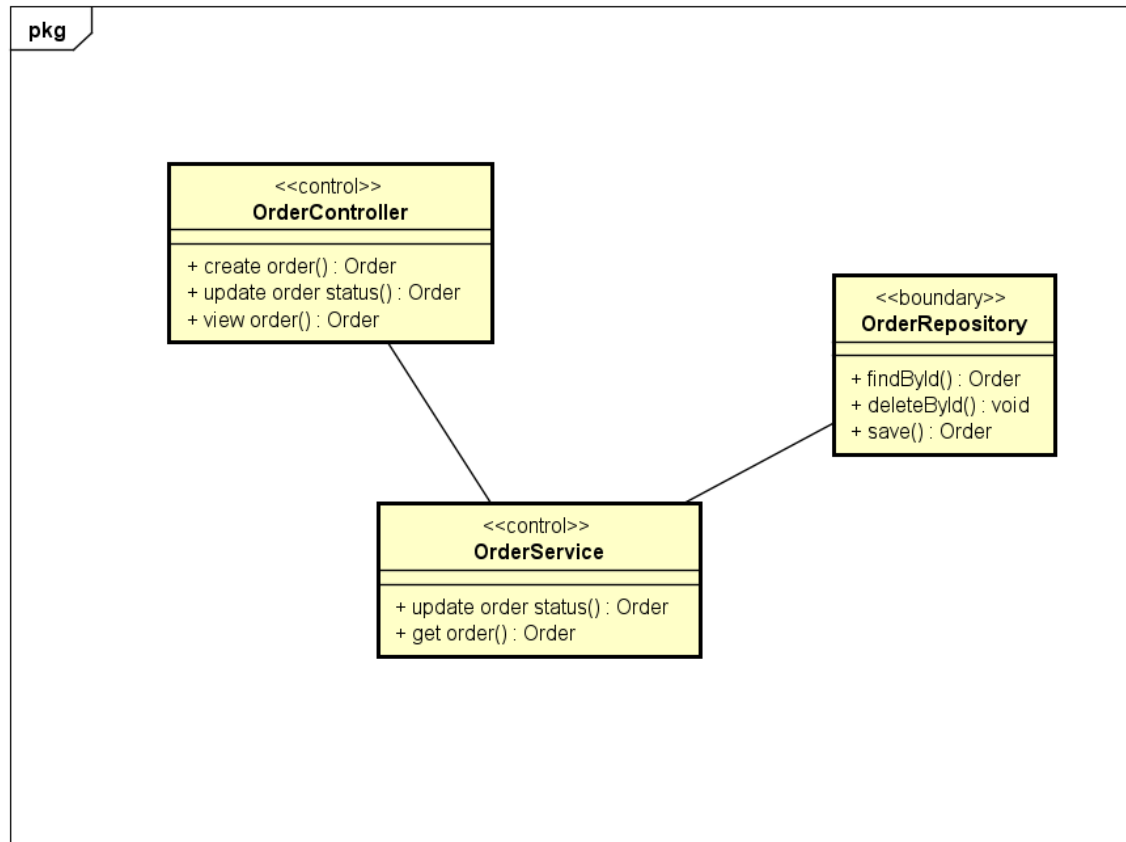
Product Analysis Class Diagram



User Analysis Class Diagram



Order Analysis Class Diagram



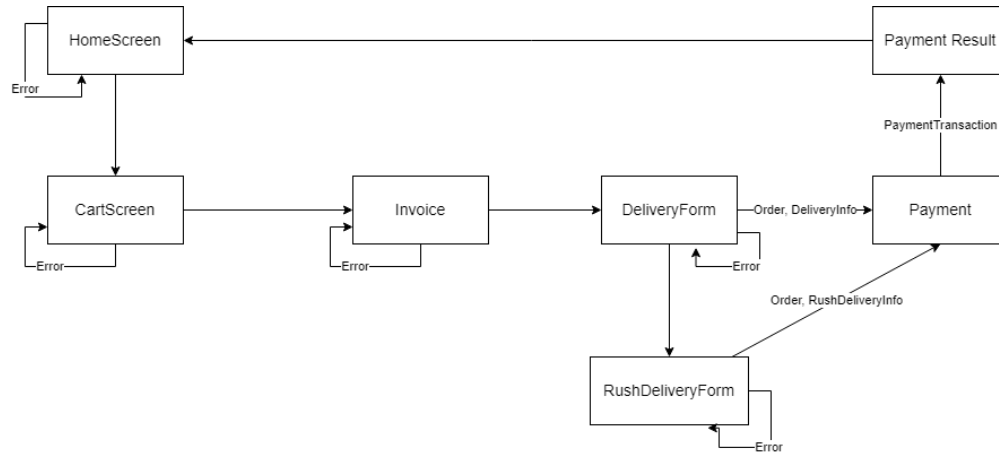
3.4 Unified Analysis Class Diagram

4 Detailed Design

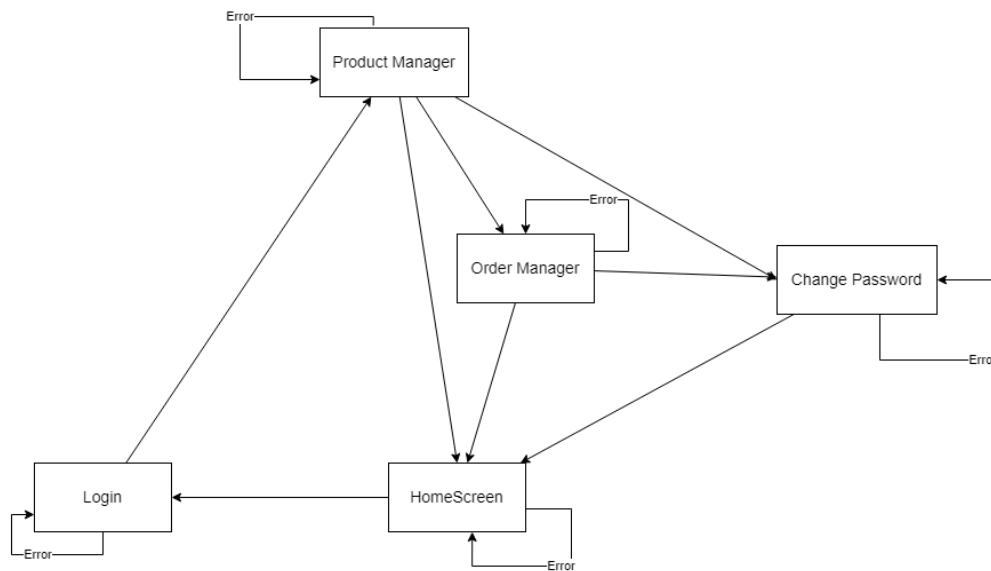
4.1 User Interface Design

4.1.1 Screen Transition Diagrams

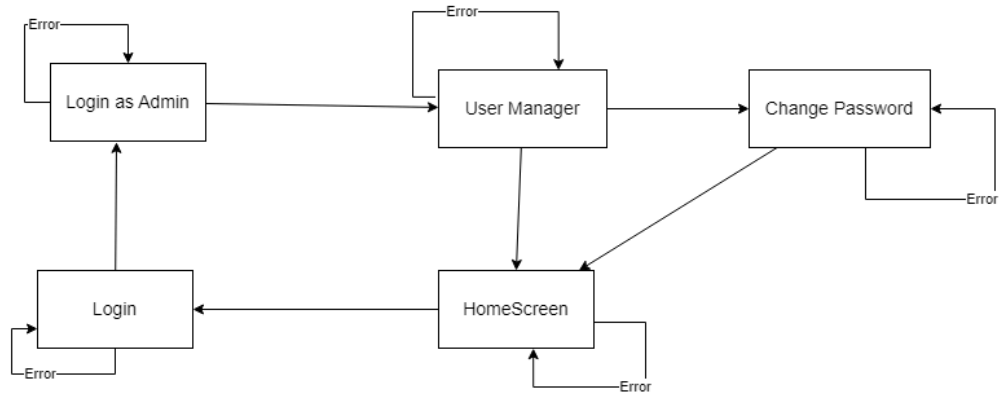
Screen Transition for Customer:



Screen Transition for Product Manager:

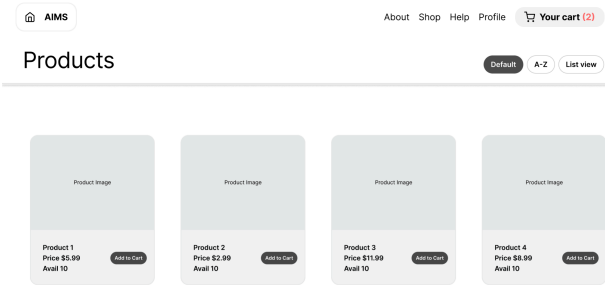


Screen Transition for Admin:



4.1.2 Screen Specifications

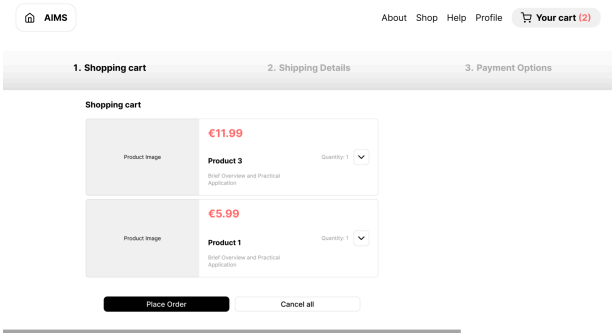
Home Screen: **Specification:**

AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Home screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, filter, item list	
		Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen	
		Switch Button	Click	Switch search filter	
		Button	Click	Add item to cart	

Field attributes:

Screen name	Home			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
Product title	50	Character	Black	Left justified
Price	50	Numeral	Black	Left justified

Cart Screen:
Specification:

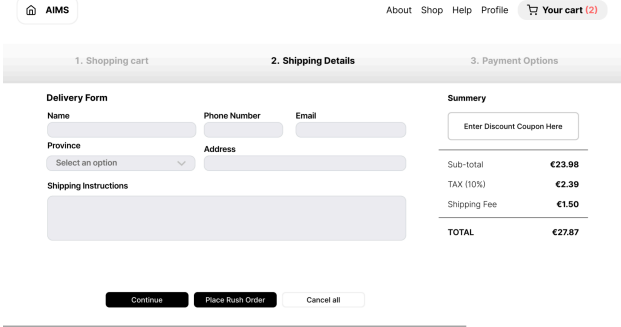
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Cart screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, layout, item cart list	
		Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen	
		Modify button	Click	Modify number of items	
		Button	Click	Confirm to place order	
		Button	Click	Cancel to place order	

Field attributes:

Screen name	Cart			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
Product title	50	Character	Black	Left justified
Product quantity	2	Numeral	Black	Larger than zero
Product price	5	Numeral	Black	Larger than zero

Delivery Form Popup:
Specification:

AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Delivery form popup	7/5/2024			Mai Đức An

	Control	Operation	Function
	Area	Initial	Display header, layout, delivery form
	Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen
	Button	Click	Choose place rush order
	Button	Click	Continue to place order
	Button	Click	Cancel to place order
	Input	Input Field	Input field for delivery form and sale coupon

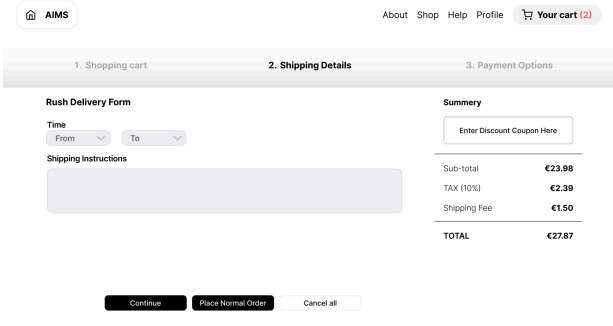
Field attributes:

Screen name	Delivery form			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
Name	20	Character	Gray	Left justified, blank can not be first, at least 3 characters, alert when error
Phone number	11	Digit	Gray	Left-justified, at least 10 number, alert when error
Email	30	Character	Gray	Left justified, has email form, alert when error
Address	50	Character	Gray	Left justified, blank can not be first, at least 3 characters, alert when error
Province	20	Character	Gray	Select option
Shipping instruction	100	Character	Gray	Left justified, optional for more detailed

				shipping instruction
Subtotal	10	Numeral	Black	Right justified, larger than zero
Shipping fee	5	Numeral	Black	Right justified, larger than zero
Tax	5	Numeral	Black	Right justified, larger than zero
Total price	11	Numeral	Black	Right justified, larger than zero

Rush Order Screen:

Specification:

AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Rush Order popup	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, layout, Rush Order form	
		Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen	
		Button	Click	Choose place normal order	
		Button	Click	Continue to place order	
		Button	Click	Cancel to place order	
		Input	Input field	Input field for rush order delivery form and sale coupon	

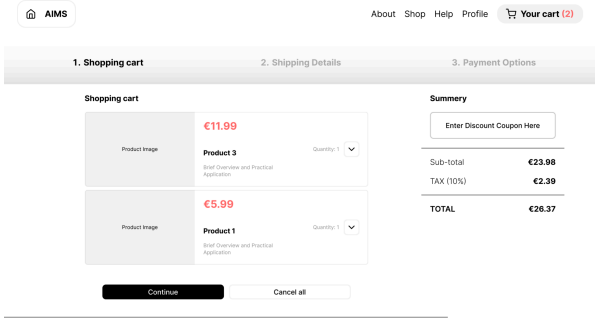
Field attributes:

Screen name	Rush Order			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks

Time From	5	Digits	Gray	Select option
Time to	5	Character	Gray	Select option
Shipping instruction	100	Character	Gray	Left justified, optional for more detailed shipping instruction
Subtotal	10	Numeral	Black	Right justified, larger than zero
Shipping fee	5	Numeral	Black	Right justified, larger than zero
Total price	11	Numeral	Black	Right justified, larger than zero

Invoice Screen

Specification:

AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Invoice screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, layout, item list, total price	
		Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen	
		Input	Input field	Enter Coupon	

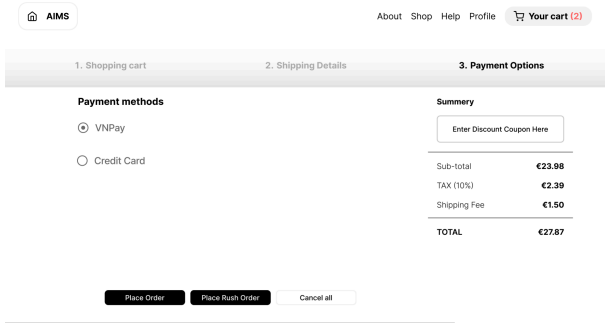
Field attributes:

Screen name	Invoice			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
Subtotal	10	Numeral	Black	Right justified, larger than zero

Tax	5	Numeral	Black	Right justified, larger than zero
Total price	11	Numeral	Black	Right justified, larger than zero

Payment Screen

Specification:

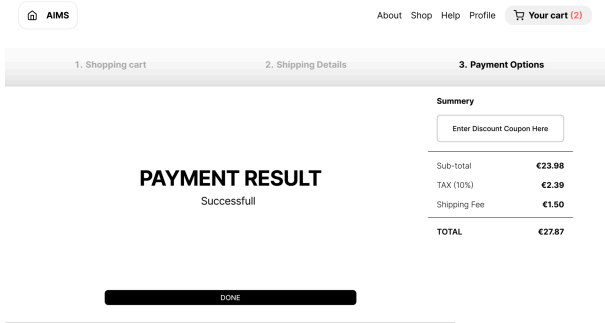
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Payment screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display payment form	
		Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen	
		Option	Click	Choose payment method	

Field attributes:

Screen name	Payment			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
Subtotal	10	Numeral	Black	Right justified, larger than zero
Shipping fee	5	Numeral	Black	Right justified, larger than zero
Tax	5	Numeral	Black	Right justified, larger than zero
Total price	11	Numeral	Black	Right justified, larger than zero

Result Screen

Specification:

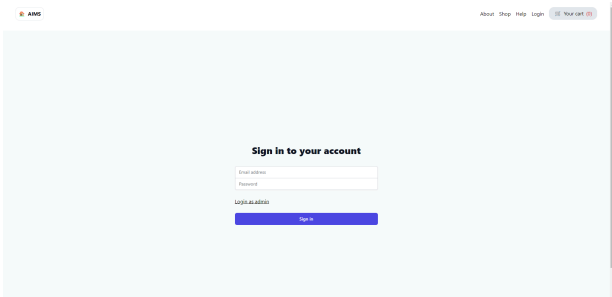
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Result screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, success part	
		Navigation Button	Click	Navigate page to home screen, cart screen, user screen, shop details screen	

Field attributes:

Screen name	Result			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks

Login Screen

Specification:

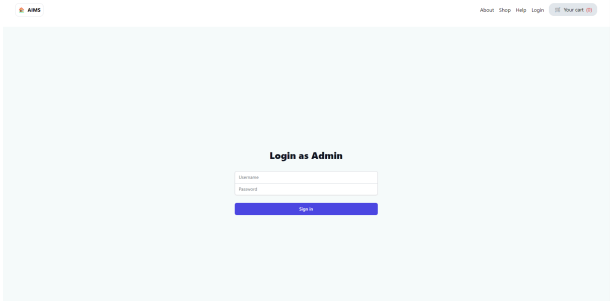
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Login screen				Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, success part	
		Navigation Button	Click	Navigate page to home screen, cart screen, login as admin screen	
		Button	Click	Login as Product Manager	

Field attributes:

Screen name	Result			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
username		character + numerical	white	
password		character + numerical	white	

Login as Admin Screen

Specification:

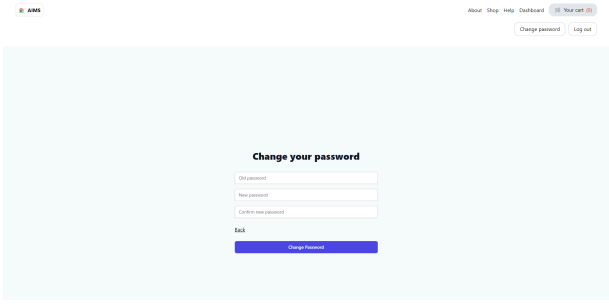
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Login screen				Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, success part	
		Navigation Button	Click	Navigate page to home screen, cart screen	
		Button	Click	Login as Admin	

Field attributes:

Screen name	Result			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
username		character + numerical	white	
password		character + numerical	white	

Change Password Screen

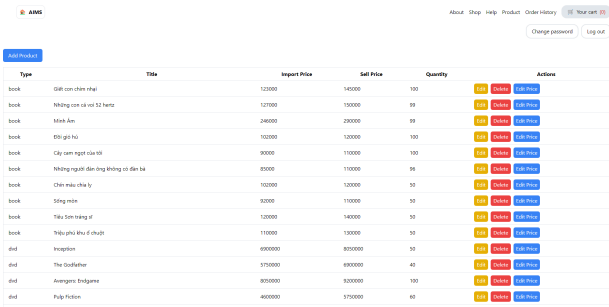
Specification:

AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Login screen				Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, success part	
		Navigation Button	Click	Navigate page to home screen, cart screen	
		Button	Click	Confirm change password	

Field attributes:

Screen name	Result			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks
username		Character	white	
old password		character + numerical	white	
new password		character + numerical	white	

Product Manager Screen: Specification:

AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Home screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, filter, item list	
		Navigation Button	Click	Navigate page to home screen, cart screen, change password screen, order manager screen	

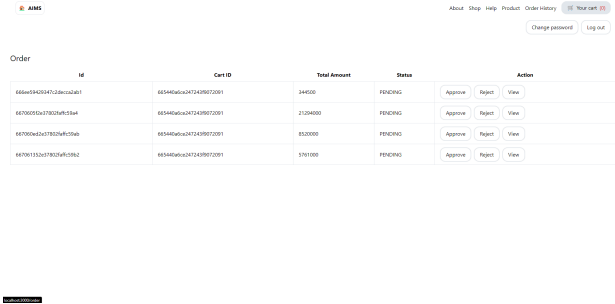
	Button	Click	Edit product
	Button	Click	Add new product
	Button	Click	Delete product
	Button	Click	Update product 's selling price

Field attributes:

Screen name	Home			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks

Order Manager Screen:

Specification:

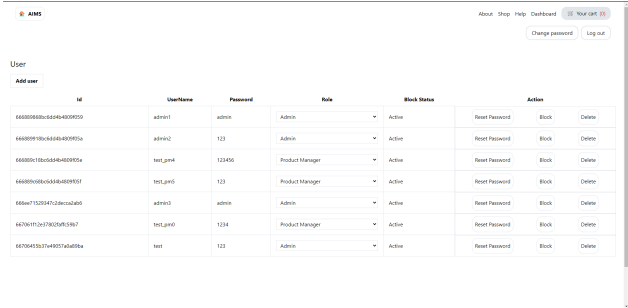
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Home screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, filter, item list	
		Navigation Button	Click	Navigate page to home screen, cart screen, change password screen, product manager screen	
		Button	Click	Approve order	
		Button	Click	Reject order	
		Button	Click	View order	

Field attributes:

Screen name	Home
-------------	------

Item name	Number of digit (bytes)	Type	Field attribute	Remarks
-----------	-------------------------	------	-----------------	---------

User Manager (Admin) Screen:
Specification:

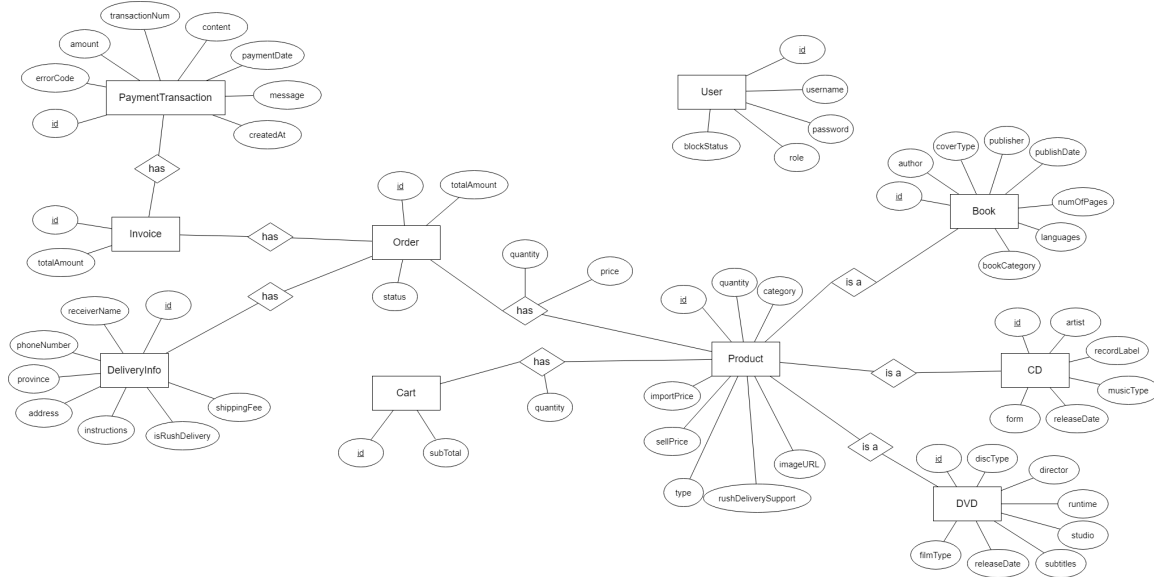
AIMS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Home screen	7/5/2024			Mai Đức An
		Control	Operation	Function	
		Area	Initial	Display header, filter, item list	
		Navigation Button	Click	Navigate page to home screen, cart screen, change password screen	
		Button	Click	Add user order	
		Button	Click	Block User	
		Button	Click	Delete User	
		Button	Click	Reset Password	
		Option	Click	Change role of user	

Field attributes:

Screen name	Home			
Item name	Number of digit (bytes)	Type	Field attribute	Remarks

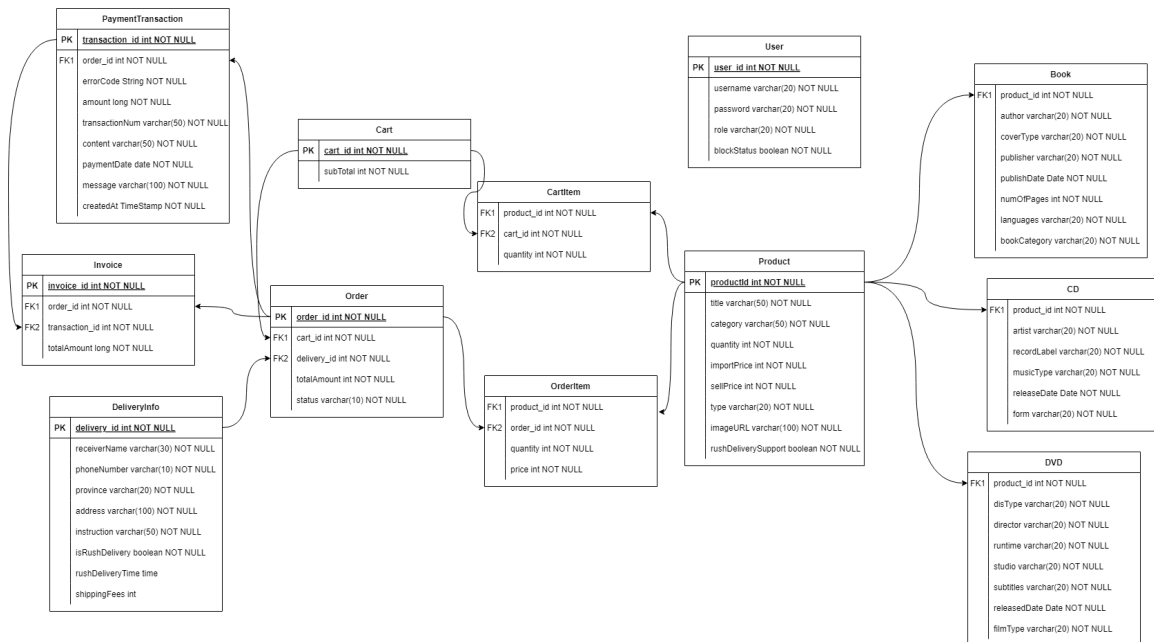
4.2 Data Modeling

4.2.1 Conceptual Data Modeling



4.2.2 Database Design

4.2.2.1 Logical Data Model



4.2.2.2 Database Detail Design

Table 1. Product

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		productId	int	yes	ID
2			title	varchar(50)	yes	product's name
3			category	varchar(50)	yes	product's category
4			quantity	int	yes	number of products
5			importPrice	int	yes	value of product
6			sellPrice	int	yes	current price
7			type	varchar(20)	yes	product type: cd, book ,...
8			imageUrl	varchar(100)	yes	product image path
9			rushDeliverySupport	boolean	yes	rush delivery availability

Table 2. CD

#	PK	FK	Column name	Data type	Mandatory	Description
1		x	productId	int	yes	ID
2			artist	varchar(20)	yes	artist's name
3			recordLabel	varchar(20)	yes	record label
4			musicType	varchar(20)	yes	music genres
5			releaseDate	Date	yes	release date
6			form	varchar(20)	yes	form

Table 3. Book

#	PK	FK	Column name	Data type	Mandatory	Description
1		x	productId	int	yes	ID
2			author	varchar(20)	yes	authors of the book
3			coverType	varchar(20)	yes	book cover
4			publisher	varchar(20)	yes	publishing house
5			publishDate	Date	yes	date of publishing
6			numOfPages	int	yes	number of pages
7			languages	varchar(20)	yes	language
8			bookCategory	varchar(20)	yes	book type

Table 4. DVD

#	PK	FK	Column name	Data type	Mandatory	Description
1		x	productId	int	yes	ID
2			discType	varchar(20)	yes	disc type
3			director	varchar(20)	yes	director
4			runtime	varchar(20)	yes	duration
5			studio	varchar(20)	yes	manufacturer
6			subtitles	varchar(20)	yes	subtitles
7			releasedDate	Date)	yes	release date
8			filmType	varchar(20)	yes	genres

Table 4. Cart

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		carttId	int	yes	ID
2			subtotal	int	yes	total value of cart without tax

Table 5. CartItem

#	PK	FK	Column name	Data type	Mandatory	Description
1		x	carttId	int	yes	ID of cart
2		x	productId	int	yes	ID of the product selected in cart
3			quantity	int	yes	number of product in cart

Table 6. DeliveryInfo

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		deliveryId	int	yes	ID
2			receiverName	varchar(20)	yes	receiver name
3			phoneNumber	varchar(10)	yes	receiver's phone number

4			province	varchar(20)	yes	province
5			address	varchar(100)	yes	delivery address
6			instruction	varchar(100)	yes	delivery instructions
7			isRushDelivery	boolean	yes	rush delivery or not
8			rushDeliveryTime	time	yes	rush delivery time
9			shippingFees	int	yes	shipping cost

Table 7. Order

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		orderId	int	yes	ID
2		x	cartId	int	yes	ID of cart
3		x	deliveryId	int	yes	ID of delivery info
4			totalAmount	int	yes	total amount including tax and shipping fee
5			status	varchar(10)	yes	status of order, eg: pending, processing, ...

Table 8. OrderItem

#	PK	FK	Column name	Data type	Mandatory	Description
1		x	productId	int	yes	ID of product
2		x	orderId	int	yes	ID of order
3			quantity	int	yes	number of product in order
4			price	int	yes	price of the product

Table 9. Invoice

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		invoiceId	int	yes	ID
2		x	orderId	int	yes	ID of order
3		x	transactionId	int	yes	ID of transaction

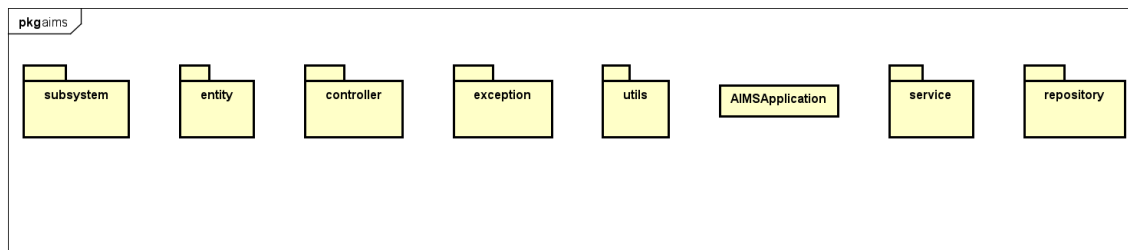
4			totalAmount	int	yes	total amount of order
---	--	--	-------------	-----	-----	-----------------------

Table 10. PaymentTransaction

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		transactionId	int	yes	ID
2		x	orderId	int	yes	ID of order
3			errorCode	varchar(2)	yes	response code from Payment System
4			amount	long	yes	amount of the payment
5			transactionNum	varchar(50)	yes	transaction number
6			content	varchar(50)	yes	transaction contents
7			paymentDate	date	yes	date of transaction
8			message	varchar(100)	yes	messages
9			createAt	timestamp	yes	timestamp of the payment request

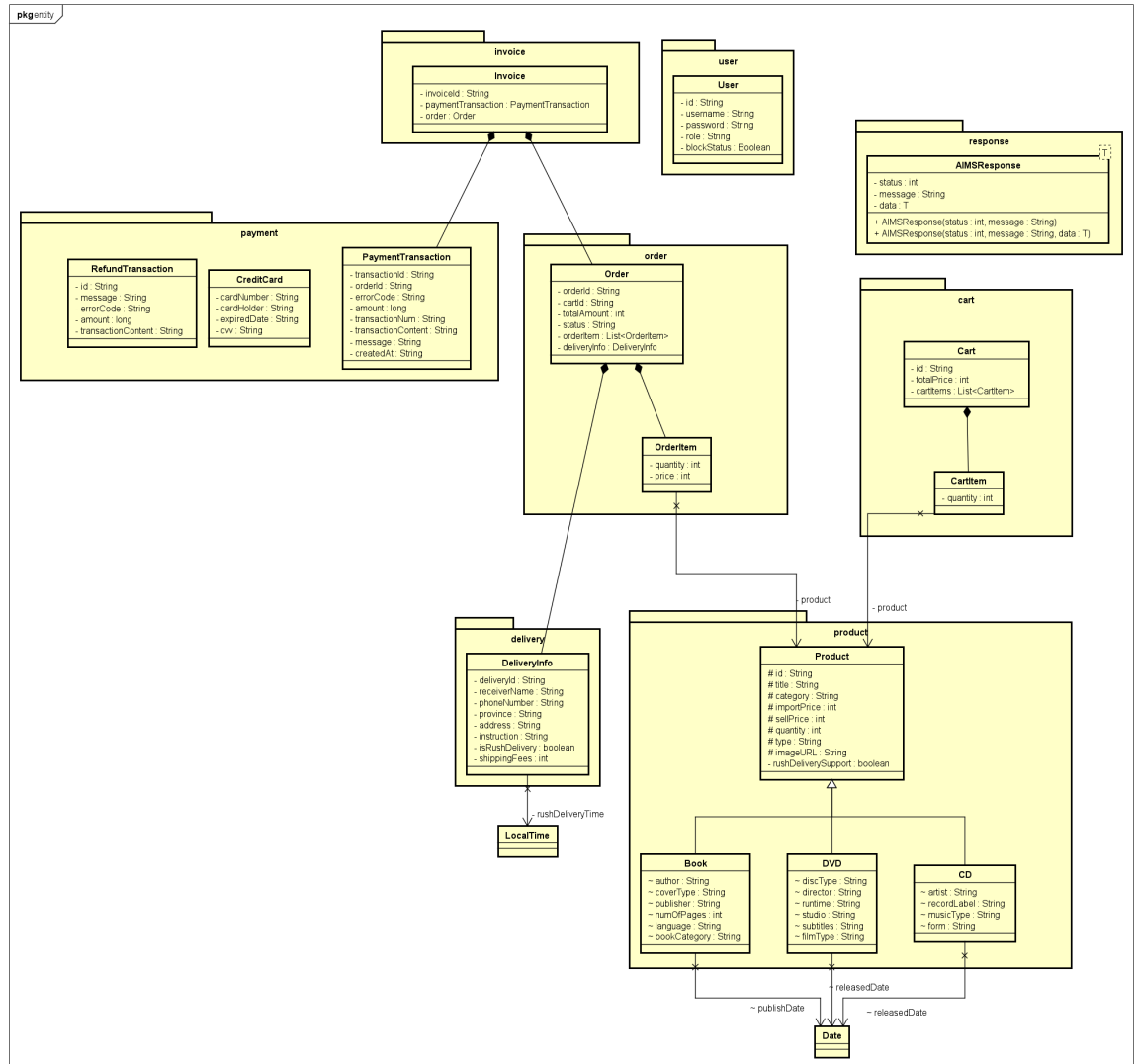
4.3 Class Design

4.3.1 General Class Diagram

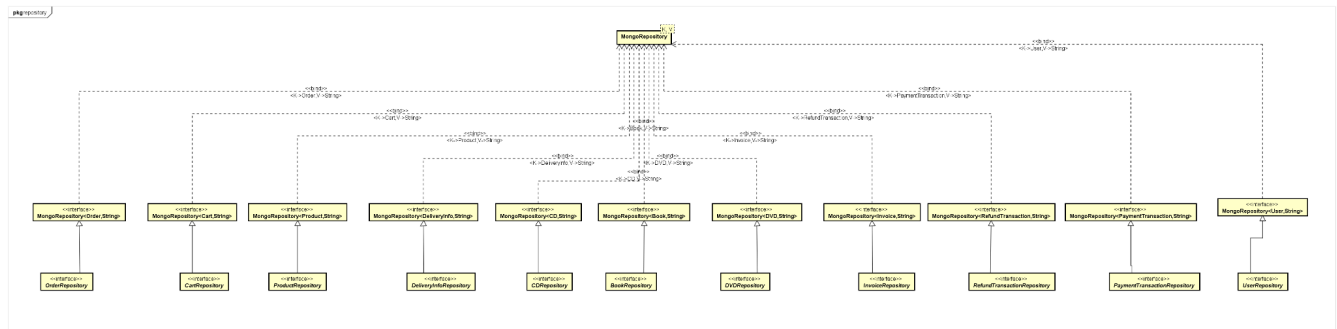


4.3.2 Class Diagrams

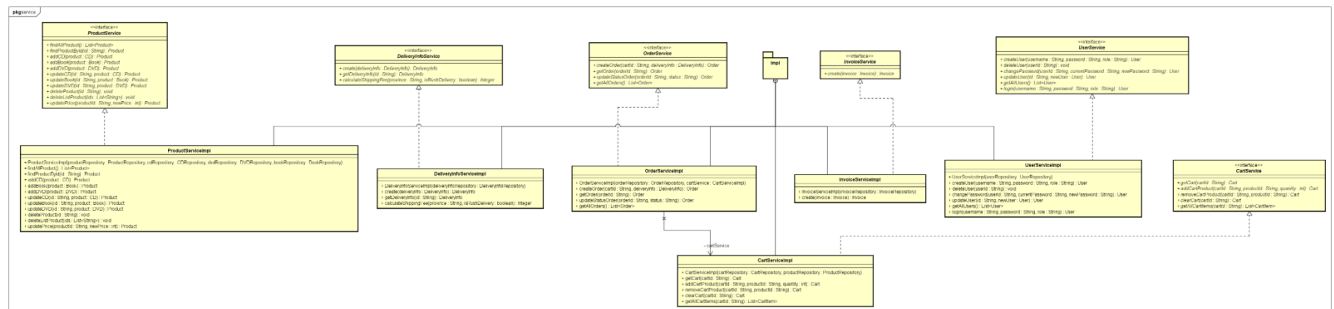
4.3.2.1 Class Diagram for Package Entity



4.3.2.2 Class Diagram for Package Repository

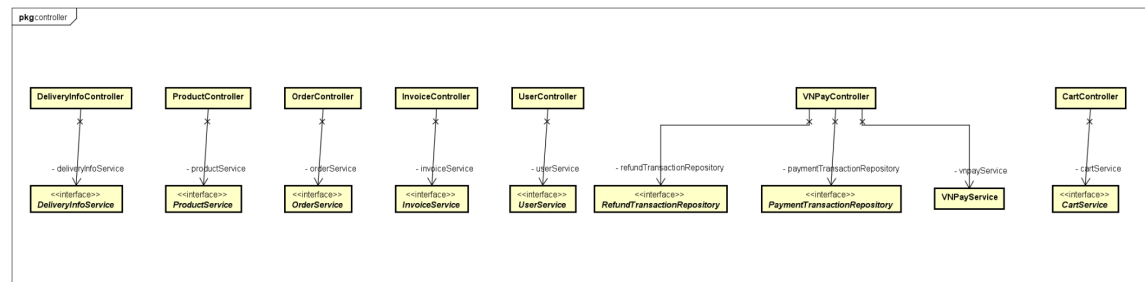


4.3.2.3 Class Diagram for Package Service

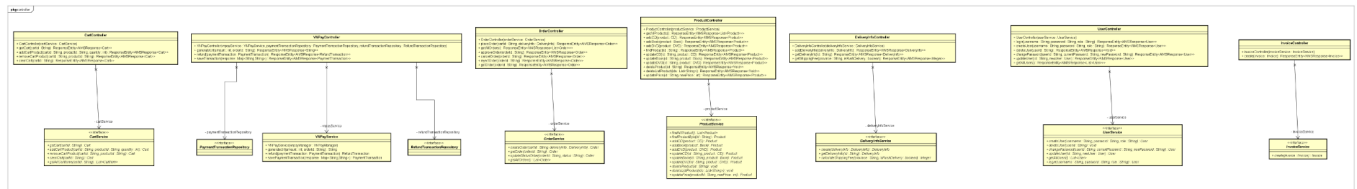


4.3.2.4 Class Diagram for Package Controller

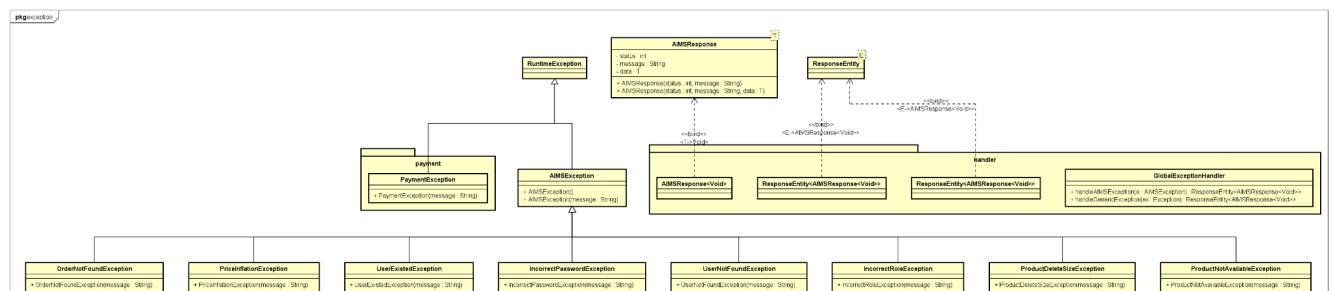
General Diagram:



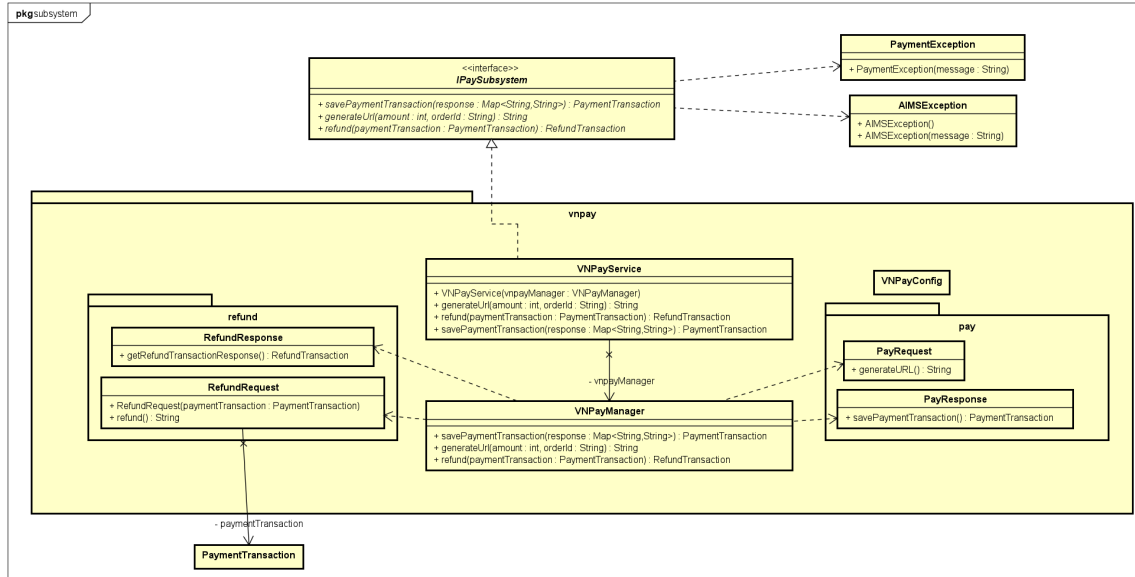
Detail Diagram:



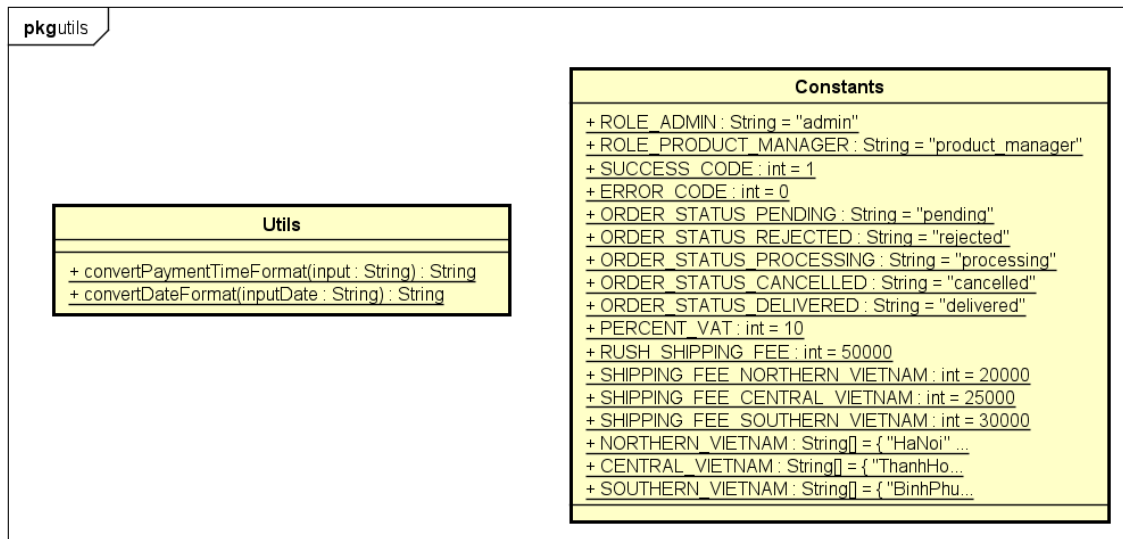
4.3.2.5 Class Diagram for Package Exception



4.3.2.6 Class Diagram for SubSystem



4.3.2.7 Class Diagram for Package Utils



5 Design Considerations

5.1 Goals and Guidelines

- *User-Friendly UI Design*
 - + *Ease of Use: The user interface should be intuitive, allowing users to complete tasks effortlessly. Each interaction, from browsing products to making a purchase, should be simple and guided by clear prompts and instructions.*
 - + *Error Handling: In the event of an error, users should receive immediate and clear notifications. Error messages should explain the issue and suggest corrective actions, helping users resolve problems without frustration.*
- *Backend Design*
 - + *Request Handling: The backend should be designed to be robust, with mechanisms to detect and handle bad requests early on. This ensures that invalid requests do not affect system performance or stability.*

5.2 Architectural Strategies

Backend: Spring Boot

- Spring Boot is a Java-based framework known for its ability to create stand-alone, production-grade Spring applications with minimal configuration. It promotes a microservices architecture that allows for easy scaling and modular development. With Spring Boot, we can build a loosely coupled system that is both extensible and maintainable, ensuring that individual services can evolve independently as requirements change.

Frontend: React.js

- React.js, a JavaScript library for building user interfaces, is chosen for the frontend. It allows developers to create dynamic and responsive user interfaces with ease. React's component-based architecture promotes code reuse and modular design, leading to a more maintainable and scalable codebase. By leveraging React.js, we can ensure a seamless and intuitive user experience, essential for any e-commerce platform.

Database: MongoDB

- For our database needs, we have selected MongoDB, a NoSQL database known for its flexibility and scalability. MongoDB's schema-less design allows for easy storage and retrieval of data in a JSON-like format, which is particularly useful for handling unstructured data. Its ability to scale horizontally ensures that our database can grow alongside our application, accommodating increasing amounts of data without compromising performance.
- By integrating these technologies—Spring Boot for the backend, React.js for the frontend, and MongoDB for the database—we aim to build a robust, scalable, and user-friendly Media Store online system that meets the demands of modern e-commerce.

5.3 Coupling and Cohesion

Coupling

The level of coupling in the project varies from average (control coupling) to low (data coupling).

Spring Boot is designed to promote a loosely coupled architecture by separating concerns into distinct modules such as controllers, services, repositories, and entities. This separation offers several advantages:

- **Error Isolation:** Decoupling different parts of the application ensures that errors in one module are less likely to impact others.
- **Extensibility:** Loose coupling facilitates easier extension and modification of the application as new requirements arise.

In a typical Spring Boot application, the request flow proceeds as follows:

- **Controller Layer:** Receives incoming requests and validates the input data.
- **Service Layer:** Handles business logic. After validation, the controller delegates the business logic to this layer.
- **Repository Layer:** Manages data access. Services interact with repositories to perform CRUD operations on entities.
- **Exception Handling:** Exceptions thrown in the service layer are directed to the exception controller class using annotations like `@ControllerAdvice` and `@ExceptionHandler`.

Cohesion

Cohesion refers to how closely related the elements (functions, classes, or modules) within a module are in software design. In our e-commerce project, we emphasize achieving Communication Cohesion and Sequential Cohesion.

Communication Cohesion:

Functions within the service layer are structured to perform a series of actions to achieve a specific business goal.

Functions often interact with each other, providing input and output, reusing code, and preventing duplication.

Each service function acts as an entry point for a controller and returns a result, ensuring that functions are cohesive and focused.

Sequential Cohesion:

Service functions are organized sequentially to accomplish a task.

This ensures that each function builds on the result of the previous one, maintaining a clear and logical flow of operations.

5.4 Design Principles

In our design, we follow several design principles, including the SOLID principles, to ensure a well-structured and maintainable e-commerce system.

Single Responsibility Principle (SRP):

- **Implementation:** We divide the project into multiple packages (controllers, services, exceptions), each handling specific roles.
- **Benefits:** This approach allows for effective class management, ensuring clear responsibilities and facilitating easier maintenance and modifications.

Open-Closed Principle (OCP):

- **Implementation:** Classes are designed to be easily extendable without needing to modify existing code.

- **Example:** Adding new payment methods or altering data storage mechanisms can be done with minimal changes to existing code.
- **Limitations:** Some parts may still require modification, especially when expanding the range of supported products.

Liskov Substitution Principle (LSP):

- **Implementation:** Child classes can replace parent classes without affecting system functionality.

Interface Segregation Principle (ISP):

- **Implementation:** Interfaces are designed to include only the necessary methods for specific functionalities.
- **Example:** The PaymentStrategy interface defines only the methods necessary for payment transactions, ensuring lean and focused interfaces.

Dependency Inversion Principle (DIP):

- **Implementation:** Dependencies are abstracted, and high-level modules depend on interfaces rather than concrete implementations.
- **Example:** The VNPay Service depends on the IPaymentSubsystem interface instead of concrete payment classes.

New Issues and Considerations:

- 6 **New Types of Media:** Adjustments may be needed to support new types of media appropriately in how they are returned to clients.
- 7 **Changes in Delivery Calculations:** Delivery methods and pricing, especially for rush delivery in specific areas, can be easily redefined within the DeliveryService and adjusted in a configuration class.

7.1 Design Patterns

The Strategy Pattern is a behavioral design pattern that defines a family of algorithms, encapsulates each one, and makes them interchangeable. This allows the algorithm to vary independently from the clients that use it.

In our e-commerce project, we need to support multiple payment methods, each with a complex implementation. To manage this complexity and maintain flexibility, we employ the Strategy Pattern.

Implementation:

- **Define Interface:** Create an interface called `PaymentStrategy` with the necessary methods that all payment methods must implement.
- **Payment Service:** Integrate the payment strategies within the payment service.
- **Strategy Factory:** Develop a `VNPayService` to retrieve the appropriate payment strategy by name.