Software Analysis Model Documentation

System Analysis and Design

Spring 2021

@Tongji

A Comprehensive Campus Community Platform for Students

Team Members:

1853572 Qiao Liang

1854062 Zhibo Xu

1854117 Zhenyu Li

1953803 Yixi Zheng

1954106 Wenchao Chen

同济大学 TONGJI UNIVERSITY

Contents

1. Introduction

1.1. Main Goals

This project aims to provide a campus information exchange platform with the three core functions of on-campus second-hand trading, community information management, and evaluation of public elective courses for the majority of students of Tongji University. Among them, campus second-hand trading is the core function of this software platform.

1.2. Target Domains

Our project aims to serve the students of Tongji University, facilitates students' life, course selection and extracurricular activities.

1.3. Progress Has Been Made

In the last assignment, we conducted agile analysis, gave the empathy map, user journey map, and user story, and completed the preliminary use case model. From the perspective of use cases, the system is divided into 7 subsystems. Because of the use case granularity is not Even, the problem of redundant actor settings. In this assignment, we revised the use case model, reducing the number of subsystems from 7 to 6, and optimized the actor settings.

In this assignment, **firstly**, we analyzed the architecture from different angles, and proposed a four-layer architecture model and an MVC architecture model. **Then,** we gave a **complete use case realization**, including sequence diagrams, mu-class diagrams, VoPC diagrams, Communication diagrams, etc. **Also,** we provide our **revised general use case diagram** in Section 4. **Finally,** in order to improve the user experience, we **updated the UI interface** of the three major functional areas.

1.4. Reading Instructions

In order to provide an overview of this article, we recommend that all readers read Section 1, which is a brief introduction to this article. For the details of the system architecture decision, you can read Section 2, from which you will learn more about the high level structure and system-level analysis of the system. Section 3 focuses on the application and will show you the specific work of the target organization through the class diagram and the relationship between the classes. Section 4 presents the revised use case model. To get familiar with the user interface of this system, you can read Section 5 for more information. Section 6 shows the reference list of this document. Section 7 presents the contributions of the team members.

2. System Architecture Design

2.1. Four-Layers Logical Architecture Analysis

As a comprehensive service platform, it will inevitably involve data processing, information transmission, emergency response management and so on.

In order to ensure the efficiency, stability, and security of the system, we need distributed storage services, database services, and a series of servers with a higher level of computing power.

Therefore, our system-level architecture design adopts the application layer, business specific layer, middle layer and Foundation layer architecture, as shown in the following figure:

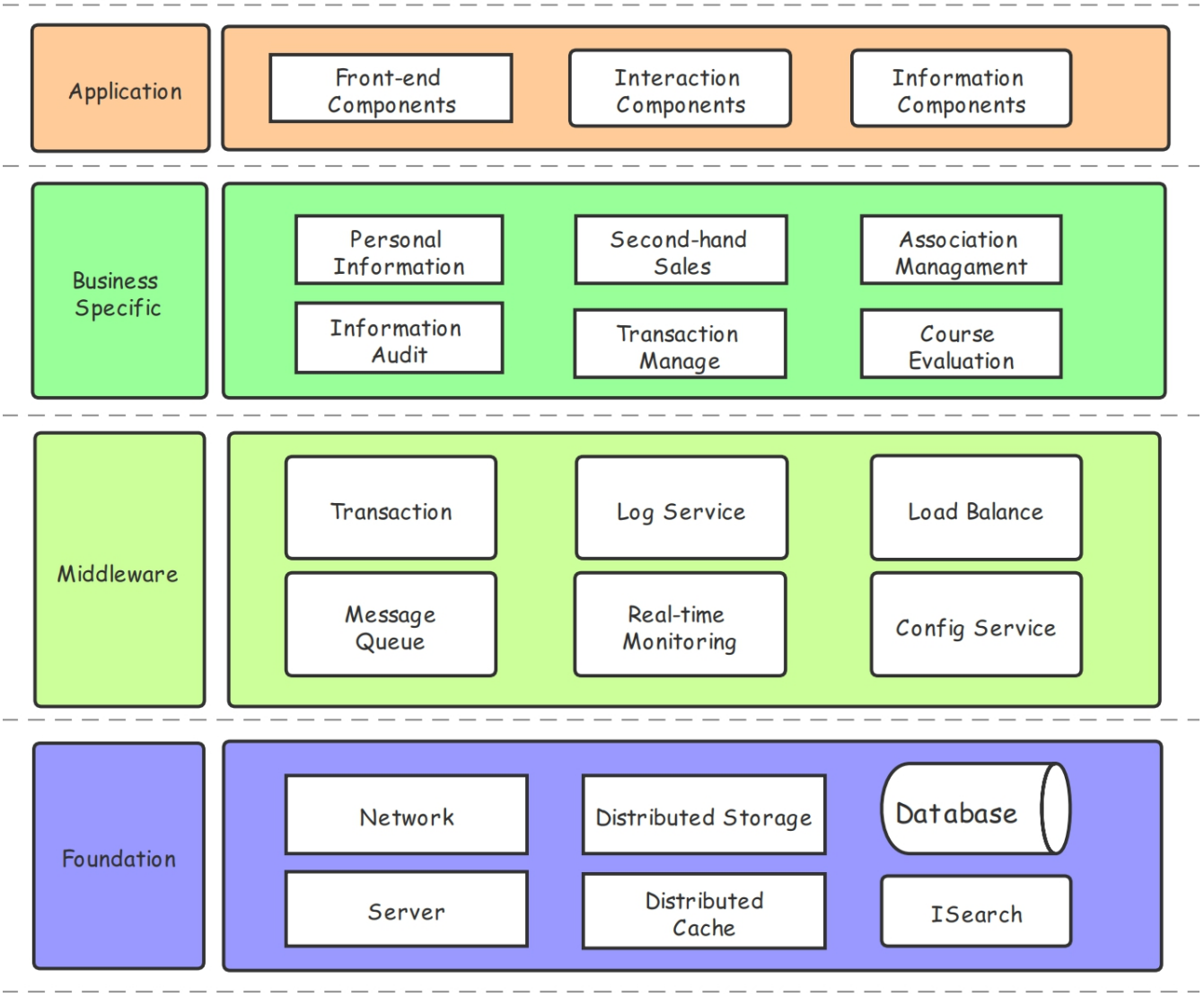


Fig.1 Logic Architecture

2.1.1. Application Layer

Front-end components

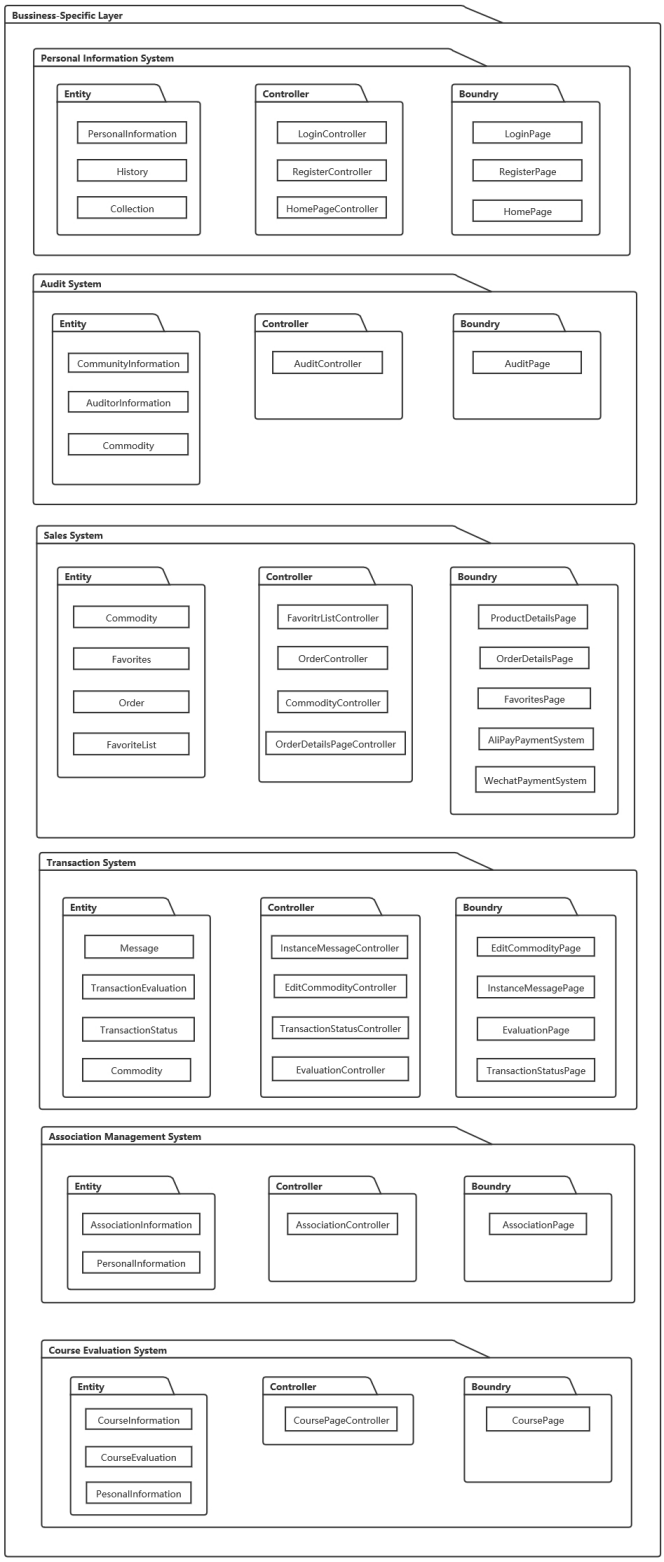
Responsible for UI and user input.

Interaction components

Manage interaction with users.

Information components

The process of handling information release and information transmission.



2.1.2. Business Specific Layer

The Business Specific Layer is composed by six subsystems. The following is the introduction of each subsystem and the description of the package diagram of the business layer.

Package Diagram Description

Business Specific Layer is divided into personal information system, audit system, trading system, management system, community management system, course evaluation system. In the class diagram, each subsystem contains entity, controller, and boundary. Here is our business layer architecture. The main functions of our system include secondary trading, community information management, course information evaluation, and basic login and registration and other functions related to personal information. In addition, there is a need for auditors to review the information, so there is also a need for a review system.

Personal Information Subsystem

Users can uniquely and effectively register an account through the system to realize "One person one ID", and can maintain all aspects of their personal information in real time, including browsing records, sold items, selected elective courses, and so on.

Information Audit Subsystem

The authority administrator has his own personal account and can manage all kinds of authority of various users in real time, including the authority to publish items, the authority to publish public elective course evaluation, and the authority to publish community information.

Second-hand Sales Subsystem

All student users can search, browse and filter second-hand items on this homepage. The homepage can also push the preferences of each user according to the internal.

Transaction Manage Subsystem

Buyer users and seller users can implement commodity transactions safely and quickly, and provide relatively complete after-sales service. All users can chat in real time after entering the second-hand trading platform, which is convenient for discussing the time and location of item transaction, and can also chat with the customer service administrator in real time.

Association Manage Subsystem

As one of the secondary functions of the platform, the system provides a platform for community users to edit and publish various information related to the community, and also allows student users to view information about clubs and student organizations that they are interested in.

Course Evaluation Subsystem

As one of the secondary functions of the platform, the system provides a platform for students to view the evaluation of public elective courses and publish the evaluation of public elective courses. In order to provide a harmonious evaluation environment, the evaluation system does not support the posting of text comments, and the function posts scoring comments on the courses that they have taken.

2.1.3. Middleware Layer

Transaction

Transaction components ensure atomicity, consistency, independence and durability of event processing.

Log Service

The log service component is responsible for recording system operation and user operation business logs during system operation.

Load Balance

We plan to use **Ribbon** to achieve load balancing on the **client side**. Ribbon will perform load balancing based on certain rules, polling, random, and so on. Ribbon can also implement load

balancing algorithms defined by ourselves. Relatively, we plan to use **Nigfix** to achieve **server side** load balancing.

Message Queue

The message queue component is used to implement asynchronous communication between services. It is used to achieve reliable, efficient and real-time cross-platform data transmission.

Real-time Monitoring

System real-time monitoring is used to collect the monitoring status of the host (network, memory, CPU, disk, kernel, etc.), including most sensitive indicators of databases and middleware.

Config Service

The configuration service component is used to configure the system settings of the software.

2.1.4. Foundation Layer

Distributed Storage

Distributed storage is a data storage technology that uses the disk space on each machine in the enterprise through the network, and forms a virtual storage device with these scattered storage resources, and the data is stored in all corners of the enterprise.

Distributed Cache

Distributed caching is to solve the bottleneck between the database server and the Web server. If a website has a lot of traffic, this bottleneck will be very obvious, and the time consumed by each database query will not be optimistic.

Database

Plan to use MySQL to implement a lightweight distributed database

ISearch

A kind of high efficiency search engine.

Network

Needed for communication between nodes.

Server

Act as a centralized transaction handler.

2.2. Deployment Architecture Analysis

The deployment architecture is shown in the figure below. The user terminal equipment is divided into mobile devices and PCs. The mobile terminal accesses the system through App, and the PC terminal accesses the system through a Web browser.

The interactive interface and interactive components are mainly deployed on the user side to realize the interaction between the user side and the distributed server.

Distributed servers use distributed cache and distributed storage. At the same time, six major subsystems are also deployed on distributed servers to process user requests on the servers. Middleware is used to process and merge request information from heterogeneous clients, and to record the running status of the server.

As an independent node, the search engine is connected to the server through the TCP/IP protocol. ISearch is a separately optimized and developed search engine for the characteristics of second-hand transactions, course evaluation, and community management. It is bound to the database through a wired connection.

The distributed server accesses the distributed database through JDBC, and interacts through the interface of the interactive component. The database is planned to be implemented in MySQL.

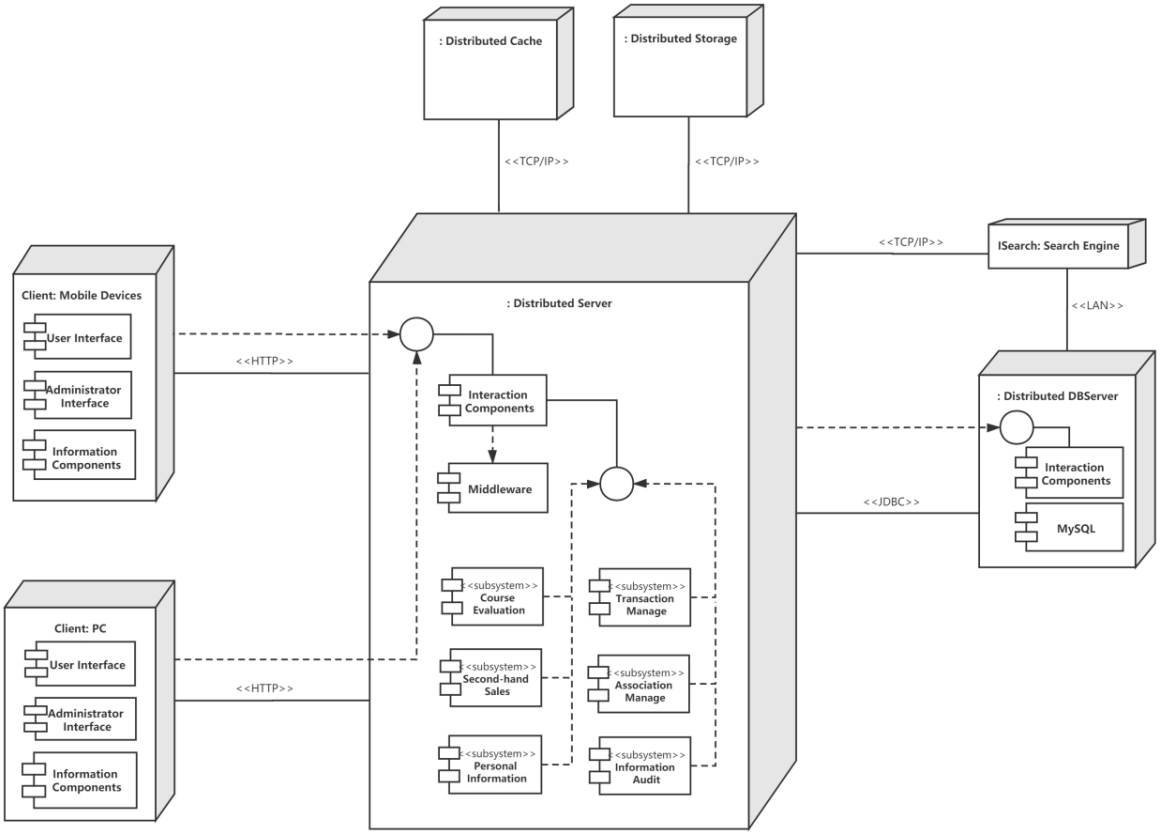


Fig.2 Deployment Architecture

2.3. MVC Physical Architecture Analysis

In the physical layer architecture, we use the MVC architecture. At the same time, we also refined the MVC architecture to 5 layers using the layered architecture. The architecture design mainly has the following advantages:

1. Our structure reduce the coupling of the code. In our MVC model, the three layers perform their duties appropriately.If the requirements of one layer change, we only need to change the code in the corresponding layer without affecting the code in other layers.

2. Our structure is conducive to division of labor and cooperation. Our MVC model separates the system by layer, so it better realizes the division of labor in development.For example, web designers can develop the JSP in the view layer, those who familiar with the business can develop the business layer, and other developers can develop the control layer.

3. Our architecture is conducive to the reuse of components. For example, the control layer can be regarded as an independent component, and the presentation layer can also be made into a general operation interface.At the same time, we can create and use multiple views for the model.

Each layer we design has its own purpose:

1. View layer: The view can organize and beautify the obtained data, and finally show it to user. At the same time, users cannot change and delete views easily, which can also ensure the data security. Each subsystem has a corresponding user view, which makes them more intuitive.

2. Control layer: Control layer is responsible for calling the model, calling the view, and passing the data generated by the model to the view. And make the relevant view to display the data. Each subsystem has a corresponding control module.

3. Business logic layer: Business logic layer is responsible for defining business logic (rules, workflow, data integrity, etc.), and receiving data requests from the controller. After processing the data request, the business logic layer submits the request to the data access layer and passes the data access result to the controller. Each subsystem has a corresponding business logic module.

4. Data access layer: The data access layer indirectly stores data in one or more data storages through the framework. We used Mybatis in the persistence layer. MyBatis is an excellent persistence layer framework that supports custom SQL, stored procedures, and advanced mapping. MyBatis integrates all the JDBC code, and it is very convenient to set the parameters and get the result set. It is more convenient that this can map the result set to many data structures implemented in java, such as list.

5. Data layer: The data layer directly manages the link and operation of the database.We use Mysql to implement the data layer.MySQL is a relational database management system, a product of

Oracle.MySQL is one of the most popular relational database management systems.MySQL is the best DBMS for web applications.MySQL saves data in different tables in relational databases, instead of putting all data in a large warehouse, which increases speed and flexibility.MySQL uses SQL to access the database, which is small, fast, and low in total cost of ownership, especially open source.These characteristics make the development of small and medium-sized websites and applications generally choose MySQL as the database.

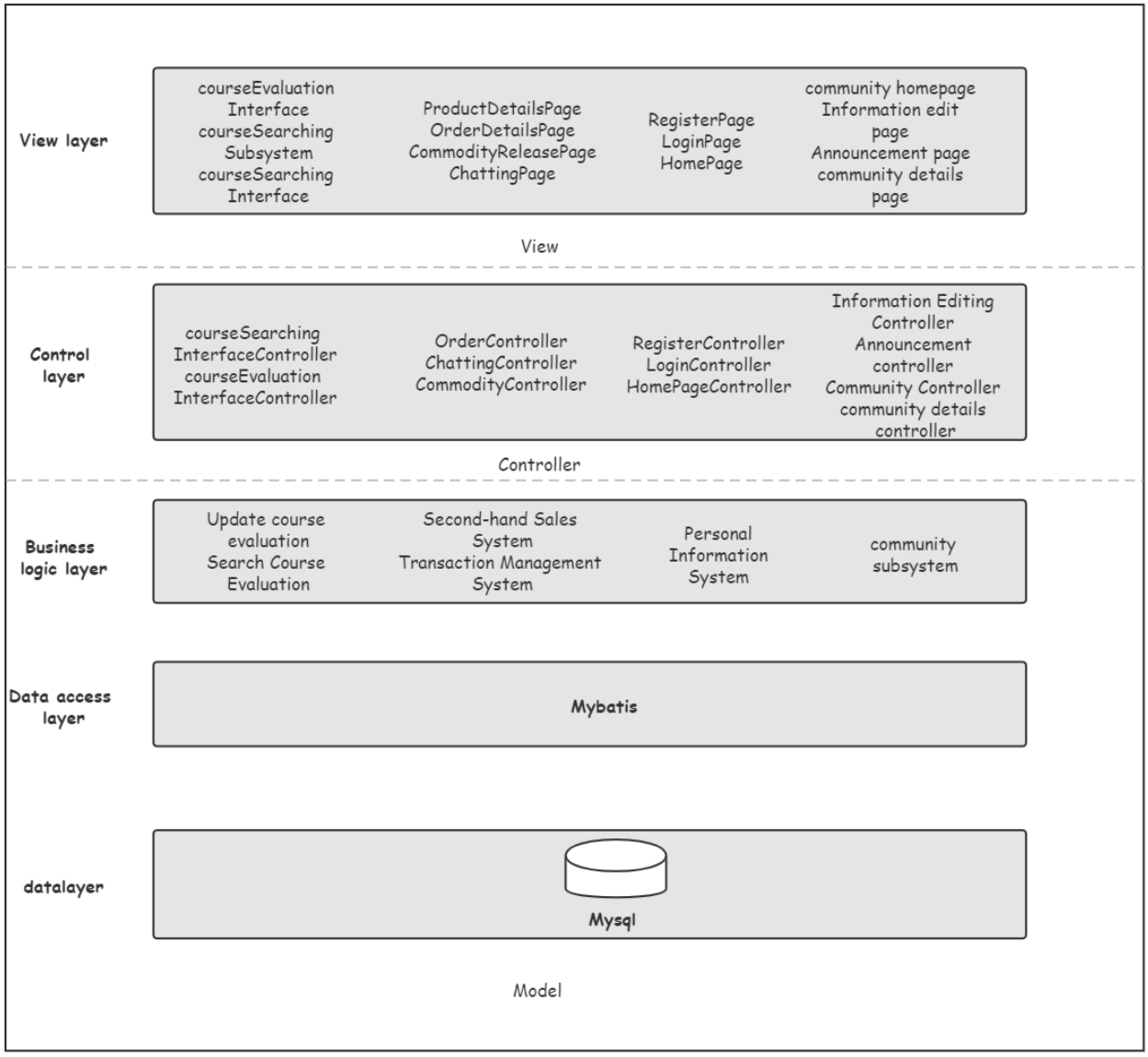


Fig.3 MVC Architecture

3. Analysis Model (Use Case Realization)

3.1. Class Diagram

This article draws the class diagrams of the **two subsystems in details** according to the sequence diagrams in each subsystem, which will be described in details below.

Regarding the design of the class diagram, because there are many controllers in each subsystem, it is not convenient to express the sequence diagram, which makes the system more bloated. In order to increase the clarity of the class diagram logic, we encapsulated the various controllers of the subsystem as a big controller named **Facade**, hiding the layers containing complex functions to a certain extent, making the interaction between the entity class, the boundary class and the control class **simpler and clearer** .

3.1.1. Class Diagram of Personal Information Subsystem

The personal information system mainly manages the user's personal information, including the basic registration and login as well as the function of modifying personal information. In addition, users can also browse and manage their own history and collections through the system. The following is a detailed list of the entity classes, control classes and boundary classes contained in the personal information system class diagram.

Stereotype	Class Name	Description
Entity	PesonalInformation	User's basic personal information, including ID, password,studentID, etc
	History	User's browsing history，stored as web link
	Collection	User's collections, stored as web link
Boundary	LoginPage	The interface the user enters when logging in
	RegisterPage	The interface that the user enters when registering
	HomePage	User's home page, including browsing history and collection
Controller	LoginController	Control the information interaction between the login interface and personal information
	RegisterController	Control the information interaction between the Register interface and personal information
	HomePageController	Control the information interaction between personal homepage and personal information, history , collection

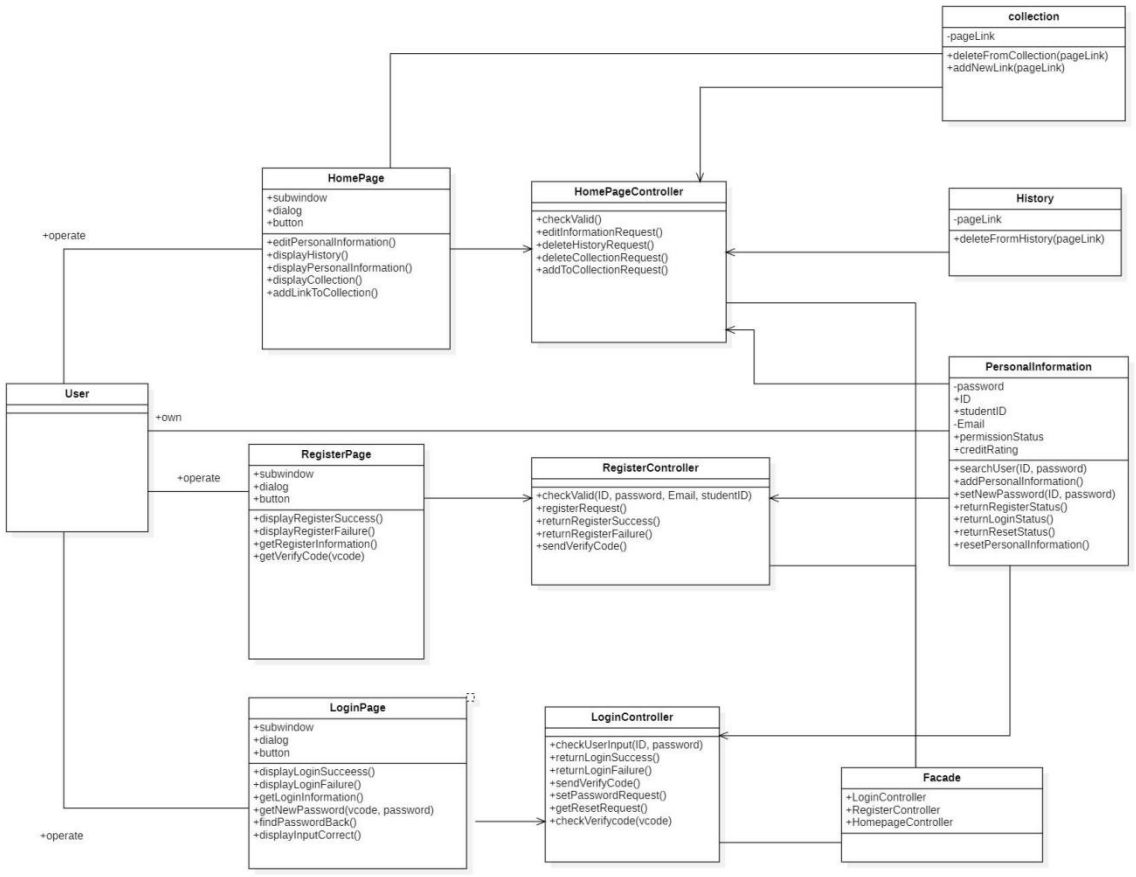


Fig. 4 Class Diagram of Personal Information Subsystem

3.1.2. Class Diagram of Second-hand Trading Subsystem

The Second-hand Trading Subsystem is mainly used to **process the buyer's browsing of goods and the payment part of the transaction with the buyer**. The main actors of the system are Student Buyer and Student Seller. In order to make this class diagram clearer, the following table details the entity classes, boundary classes and control classes of this class diagram.

Stereotype	Class Name	Description
Entity	StudentBuyer	Student users who browse purchased products .
	StudentSeller	Student users who sell products .
	Order	Record the information of each order for easy management .
	Favorites	Encapsulate the user’s favorite product information for easy management and editing .
	Commodity	Product information published by the seller .
	FavoriteList	Encapsulate different kinds of favorites of users.

Boundary	FavoritesPage	User interface for editing favorites .
	OrderDetailsPage	Interface for filling in order information and paying.
	ProductDetailsPage	Page used to browse product details .
	WechatPamentSystem	External WeChat payment system with external API that can be called .
	AliPayPaymentSystem	External AliPay payment system with external API that can be called
Controller	OrderController	control the message between the order class and the external payment system .
	FavoriteListController	Used to control the editing of different favorites between personal favorites .
	CommodityController	Used to control the information transmission and interaction between commodities and other entity classes .
	OrderDetailsPageController	Used to control the interaction between the order detail page and the entity set such as the order .

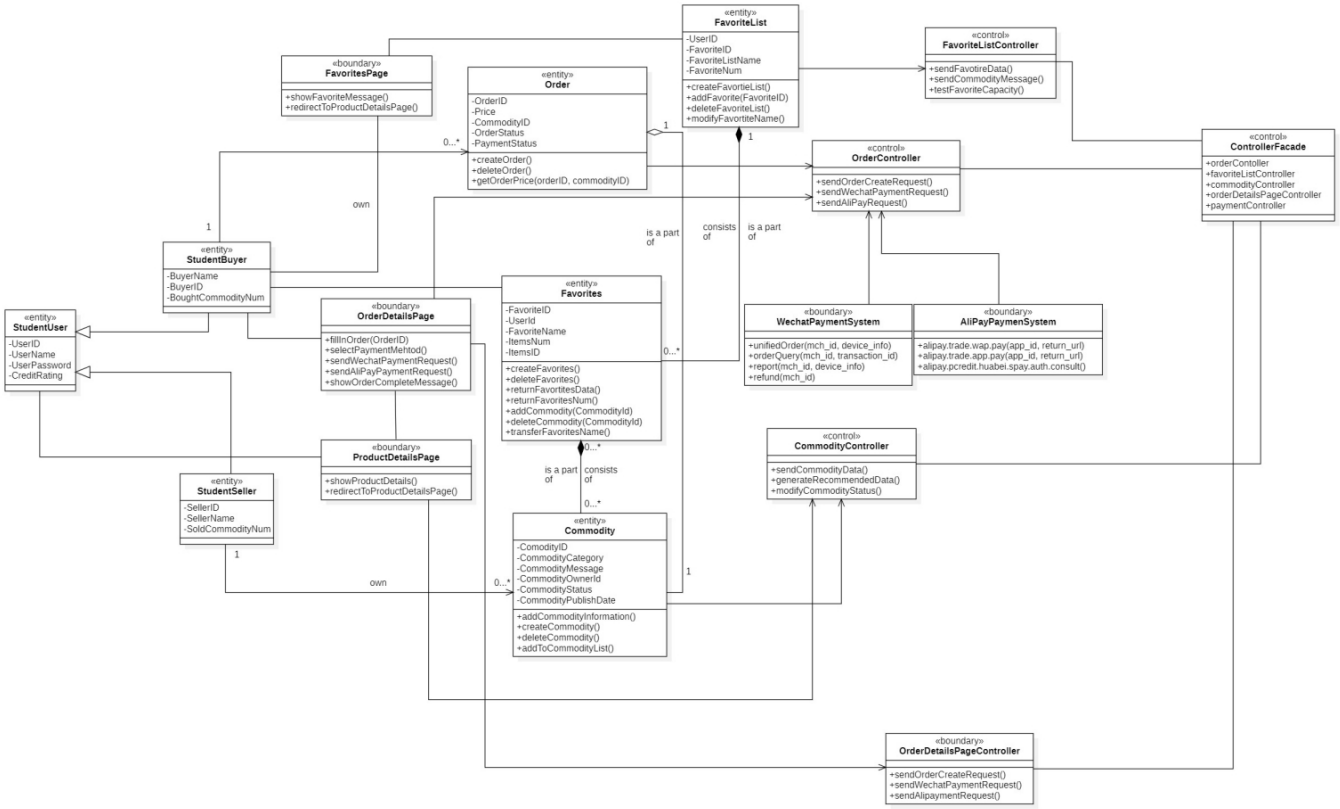


Fig. 5 Class Diagram of Second-hand Trading Subsystem

3.2. Interaction Diagram

3.2.1. Use Case of "Login"

According to the class diagram and demand analysis of the personal information system, draw the time sequence diagram of the personal information system. Since there are many use cases, login use case is selected here to draw a sequence diagram.

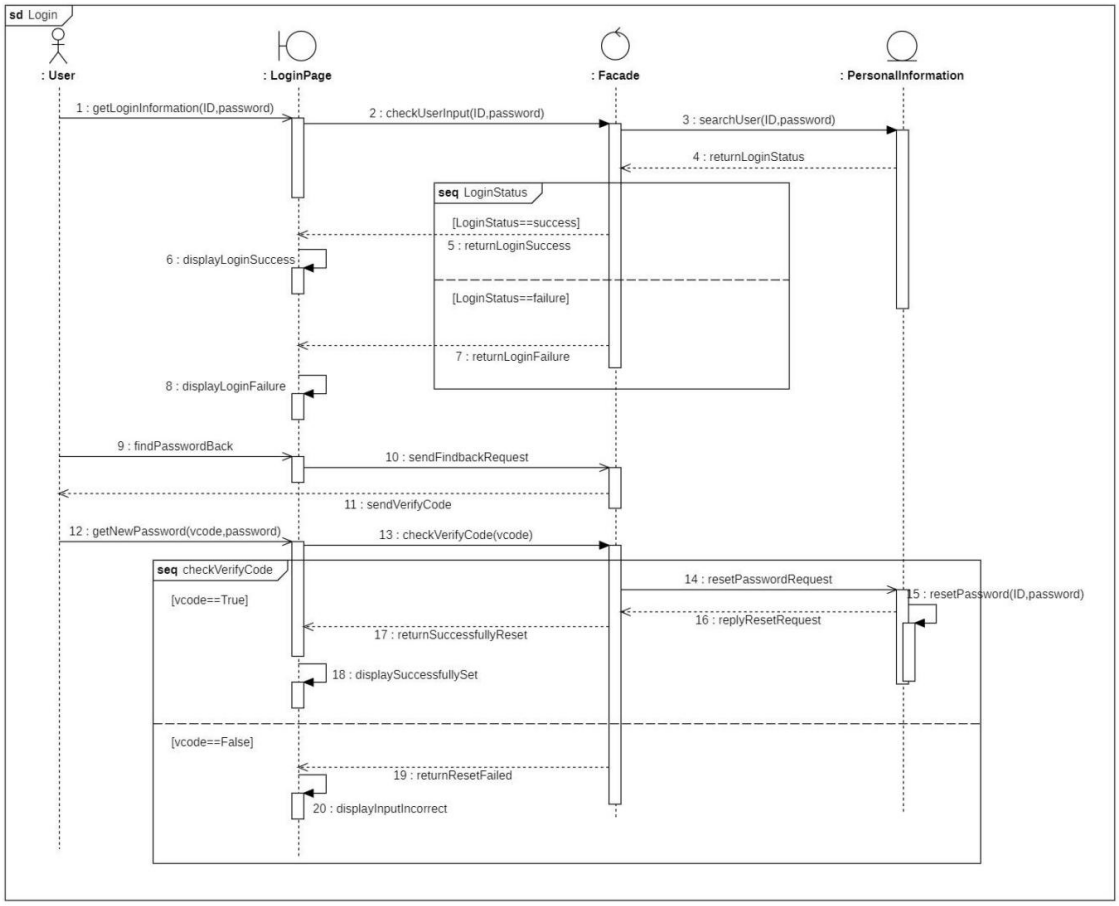


Fig.6 Sequence Diagram of "Login"

Login sequence diagram mainly includes the common login function and password retrieval function. The precondition for this sequence diagram is that the **user** has successfully registered an account. The classes involved in the sequence diagram mainly include the boundary **LoginPage,** the controller **Facade**, and the entity **PersonalInformation**. In the login use case, the user first enters the ID and password in the LoginPage, and then interacts with the PersonalInformation entity through the LoginController to judge whether the user has successfully registered. If it has successfully registered, the user can log in to the online system, otherwise, it cannot be logged in. Second, if the user forgets his password, he can reset it with the **Forgot Password** function. The user initiates the password retrieval application in the LoginPage, and then the LoginController sends the verification code to the user. The user then enters the verification code in the LoginPage. After

the verification is successful, the user can establish contact with the PersonalInformation entity through the boundary and the controller, and reset the password.Facade is an integration of a set of controllers, which we will cover in more detailes later.

VoPC diagrams can be used to more clearly and concisely discover the relationship between users and entity, boundary and controller.

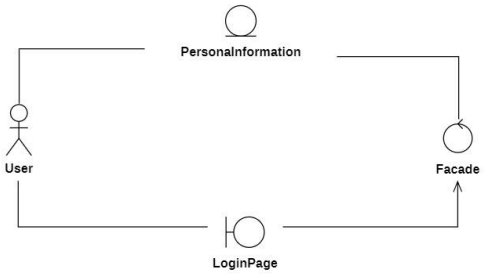


Fig.7 VoPC Diagram of “Login”

Communication diagrams show the communication of information between classes and users. Although similar in structure to sequence diagrams, communication diagrams emphasize the organization of objects.

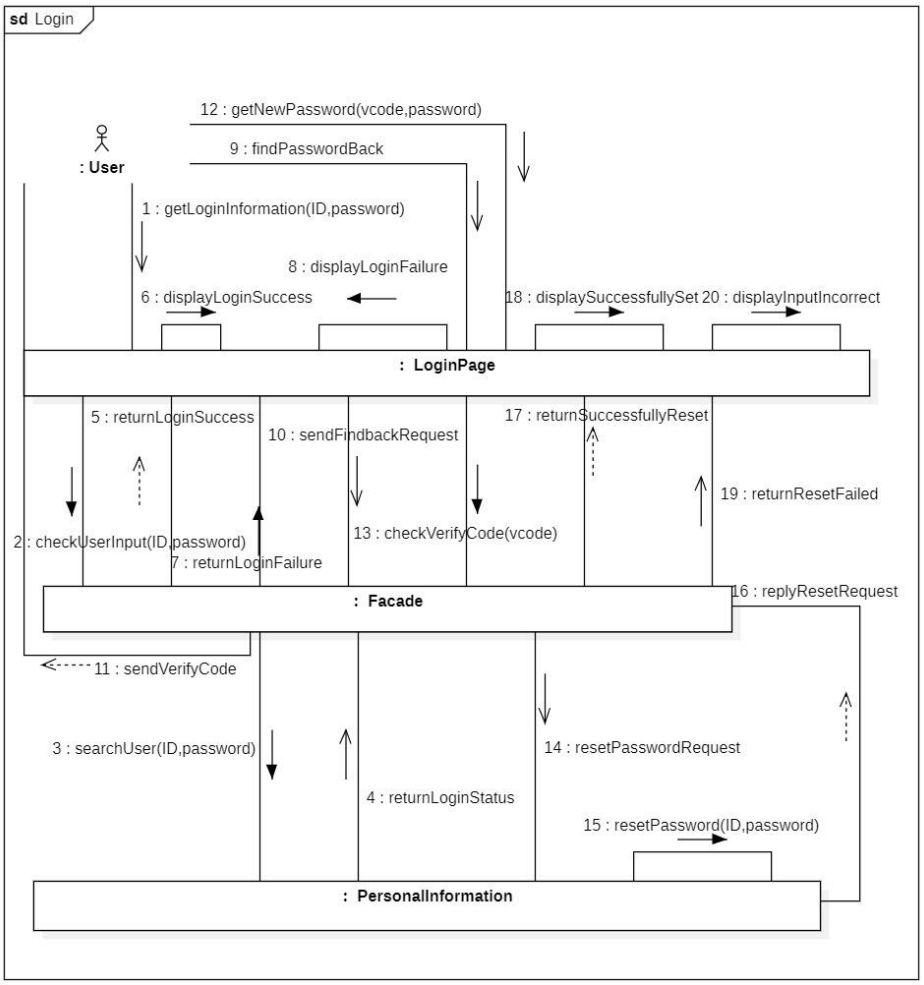


Fig.8 Communication Diagram of “Login”

3.2.2. Use Case of "Audit Community Information"

In the audit system, we select the audit community information use case to draw sequence diagram.

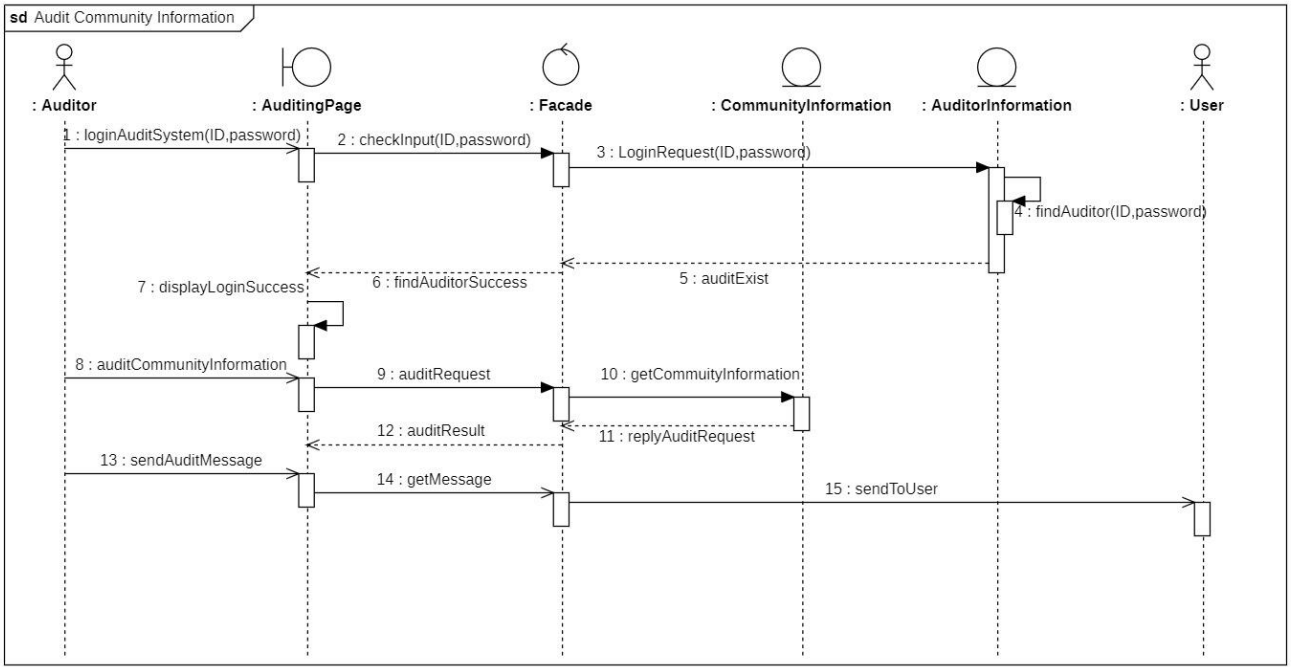


Fig.9 Sequence Diagram of "Audit Community Information"

The main purpose of the auditor to audit the community information is to check whether the community information conforms to the norms and whether it contains inactive information. Secondly, through the audit, the auditorr can also timely remind the person in charge of the community to update the community information. Because the events are in the community management system, here only involves the AuditPage as boundry, and the main body of the audit is information auditor, the auditor needs to log in the audit system to enter the AuditPage, and through the controller Facade to enter the AuditorInformation entity to check whether there is the auditor. After successfully logging into the audit system, the auditor can audit the community information, and then the auditor sends the audit letters to the community users through the boundary and the controller to complete the audit process.

The following figure is the communication diagram of the audit community information use case, which mainly shows the content and direction of information transmission between different types of auditor audit process.

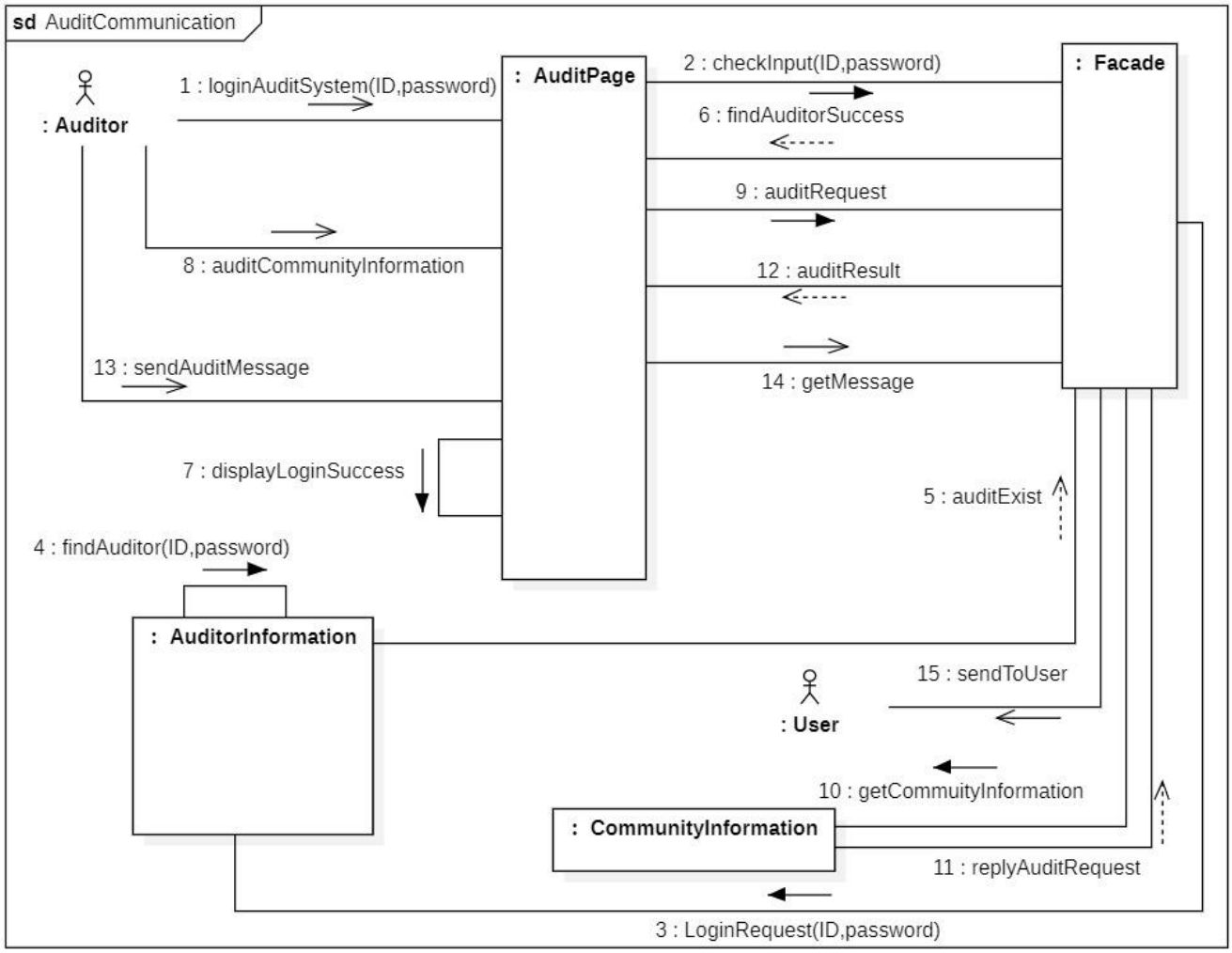


Fig.10 Communication Diagram of “Audit Community Information”

The following figure is a VoPC diagram for the use case of audit community information, which is more concise than the sequence diagram and communication diagram.

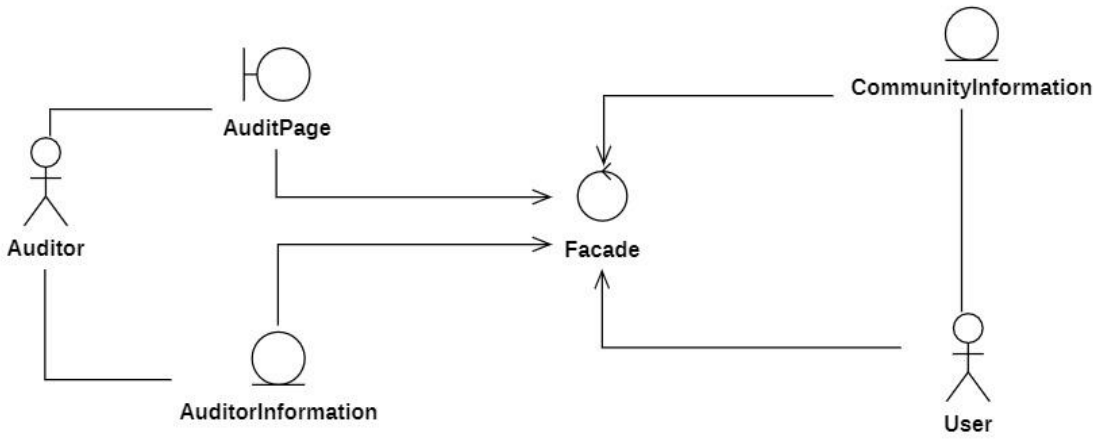


Fig.11 VoPC Diagram of “Audit Community Information”

3.2.3. Use Case of “Place an Order”

According to the use case specification and activity diagram of the "place an order" use case in the demand analysis, we extracted the corresponding candidate classes and drawn the corresponding sequence diagram, as shown in the following figure:

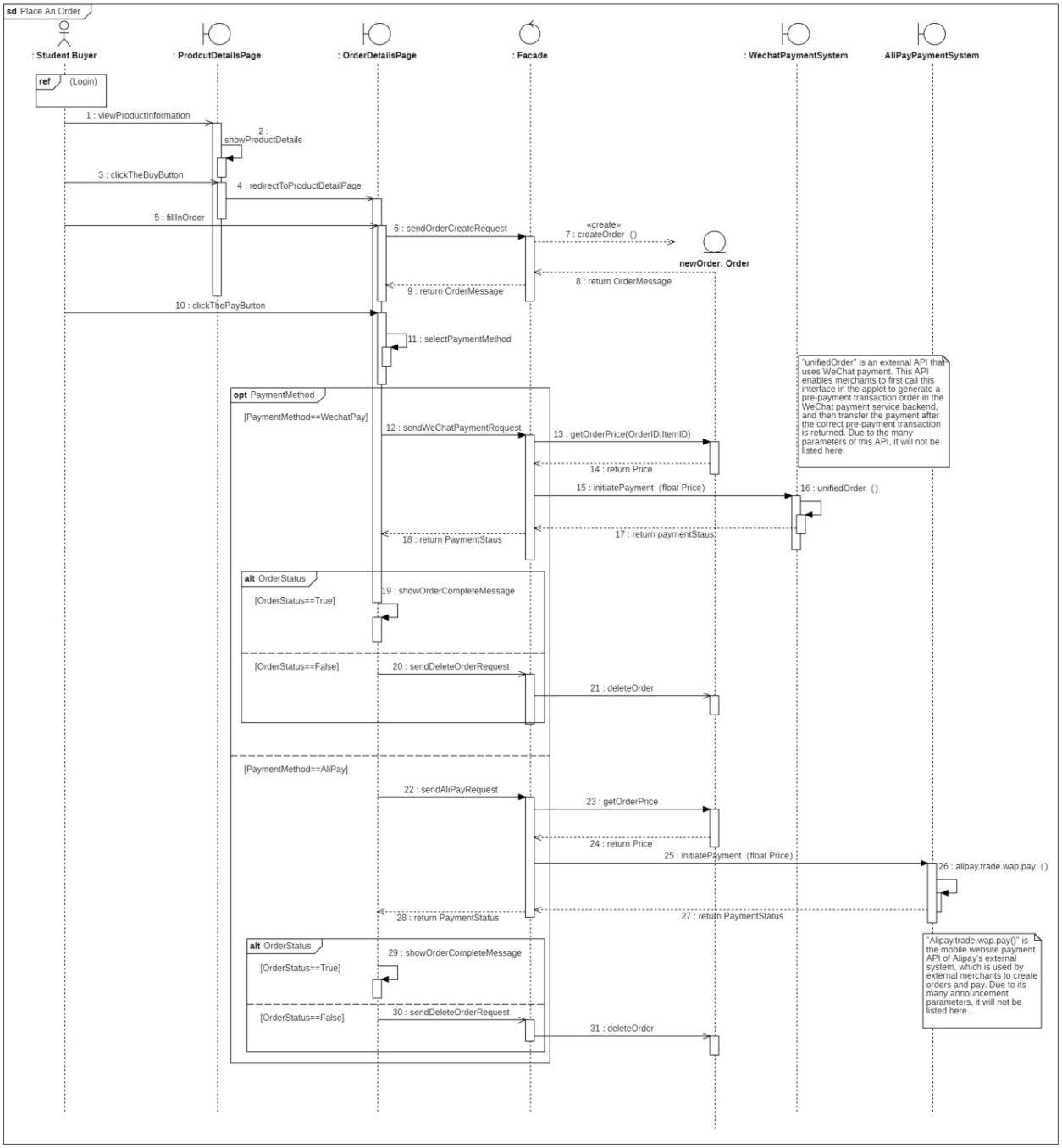


Fig.12 Sequence Diagram of “Place an Order”

The above sequence diagram describes in details of the complete internal interaction process of the system.When **Student Buyer** clicks the purchase button on the product details page to enter the order details page ,and then fill in the order form. The precondition for the beginning of the sequence diagram is that the user has already logged in. As the only Actor in the sequence diagram, the student buyer first checks the product details page and clicks the "Buy" button. The boundary class named **ProductDetailsPage** will be redirected to the boundary class named OrderDetailsPage, which is the page for order filling. After the user fills in the order, the **Facade contoller** receives **the order creation request** sent by the OrderDetailsPage, and completes the creation of an Order object.

It should be noted that the facade controller is an encapsulation of the controllers of the subsystem. In order to make the presentation of the sequence diagram more logical, we regard the inside of the controller as a "black box".

After completing the creation of the order object, the order filling details page will prompt the student buyer to **select the payment method**, and then send the corresponding payment request to the Facade controller. Taking the WeChat payment method as an example. The controller will call the corresponding **API** of WeChat payment to complete the initialization of order payment. Through the call of external API, the design of the system can be simplified and the stability of the completion of the payment function can be improved. Finally, **if the payment completes**, the corresponding payment success information will be displayed on the order details page; **if the payment fails**. The order details page will display the corresponding payment failure information.

Since the above sequence diagram fails to show the details of the call interaction inside the Facade controller , the following sequence diagram shows the detailed control process of each **sub-controller** inside the Facade control class in the order use case:

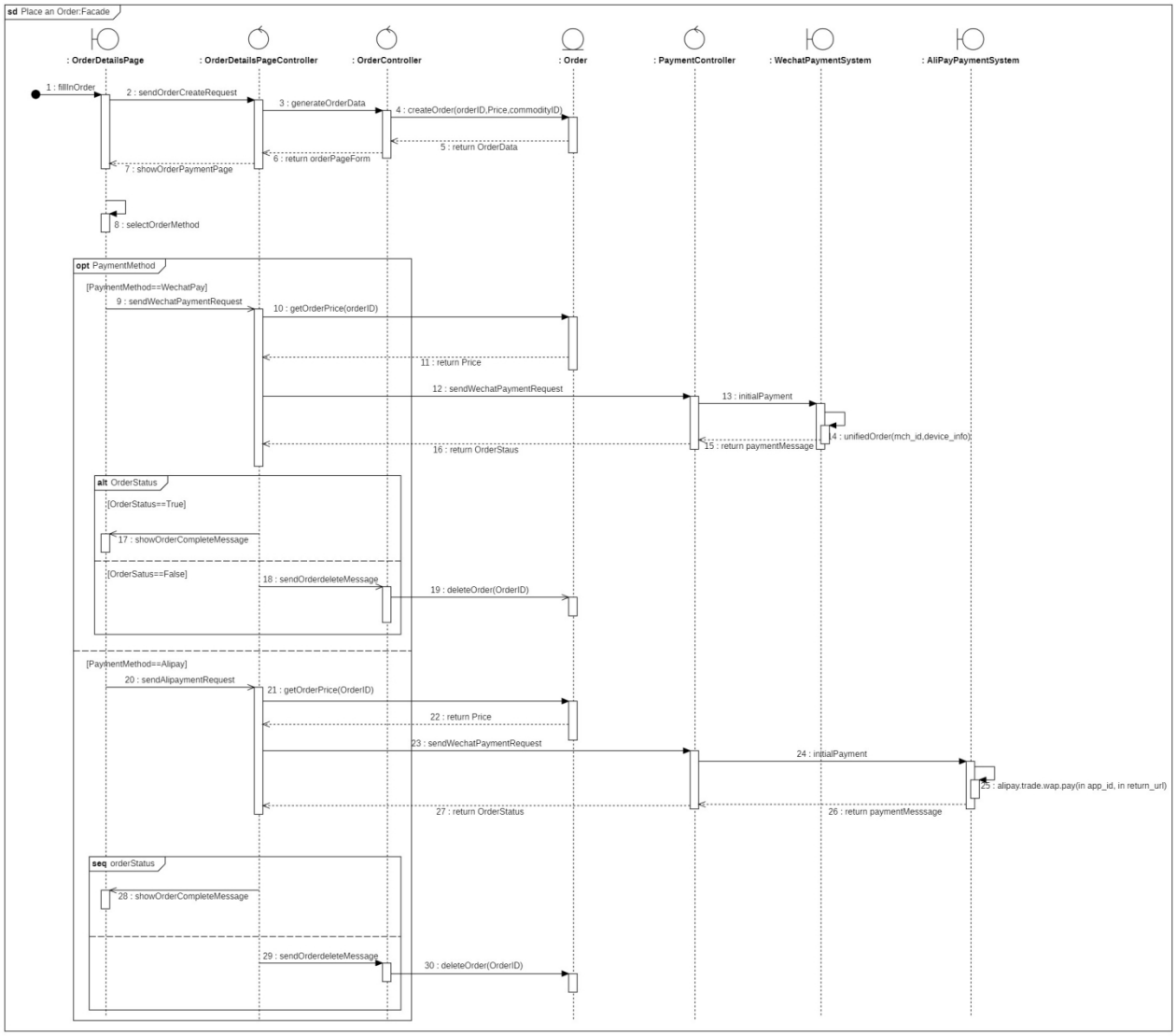


Fig.13 Sequence Diagram of the details of the interaction in Facade Controller

In addition, in order to clearly explain the classes that are related in the order use case perspective, this article also draws the corresponding VoPC diagram of the order use case (use case perspective enhancement class diagram), as shown in the following figure:

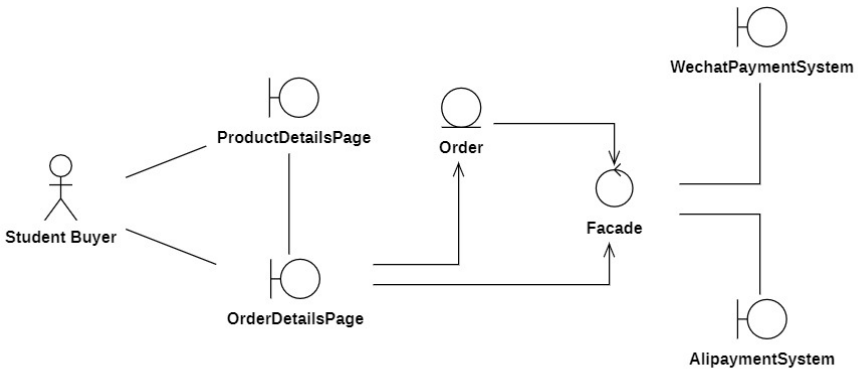


Fig.14 The VoPC Diagram of use case "Place an order"

In addition to sequence diagrams, communication diagrams emphasize the organization of objects, showing a path and how surface objects are connected to another object. The following figure shows the Communication diagram corresponding to "Place an order" use case:

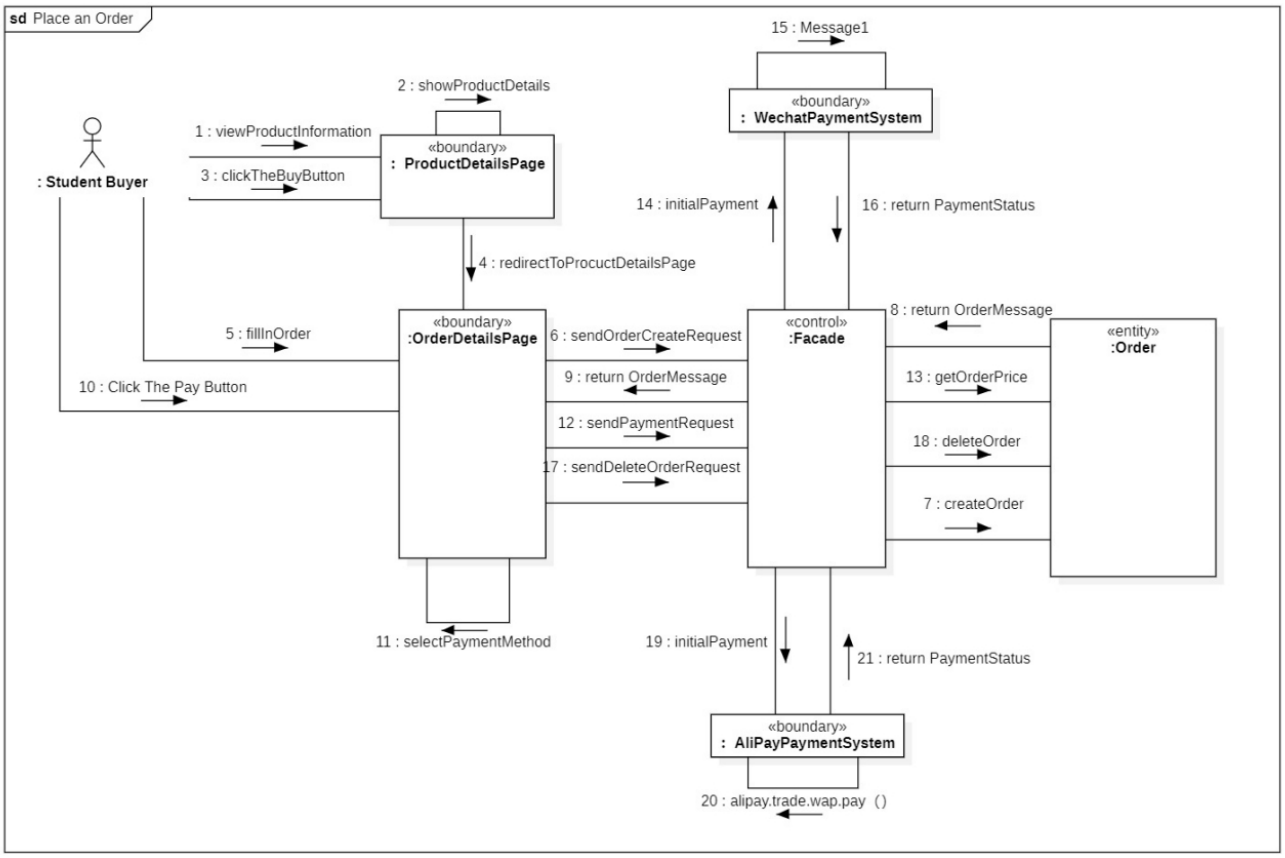


Fig.15 The Communication Diagram of use case "Place an order"

3.2.3. Use Case of “Search Course Evaluation”

According to the use case specification and activity diagram of the "**Search course evaluation**" use case in the requirement analysis, we extracted the corresponding candidate classes and drawn the corresponding sequence diagram, as shown in the following figure:

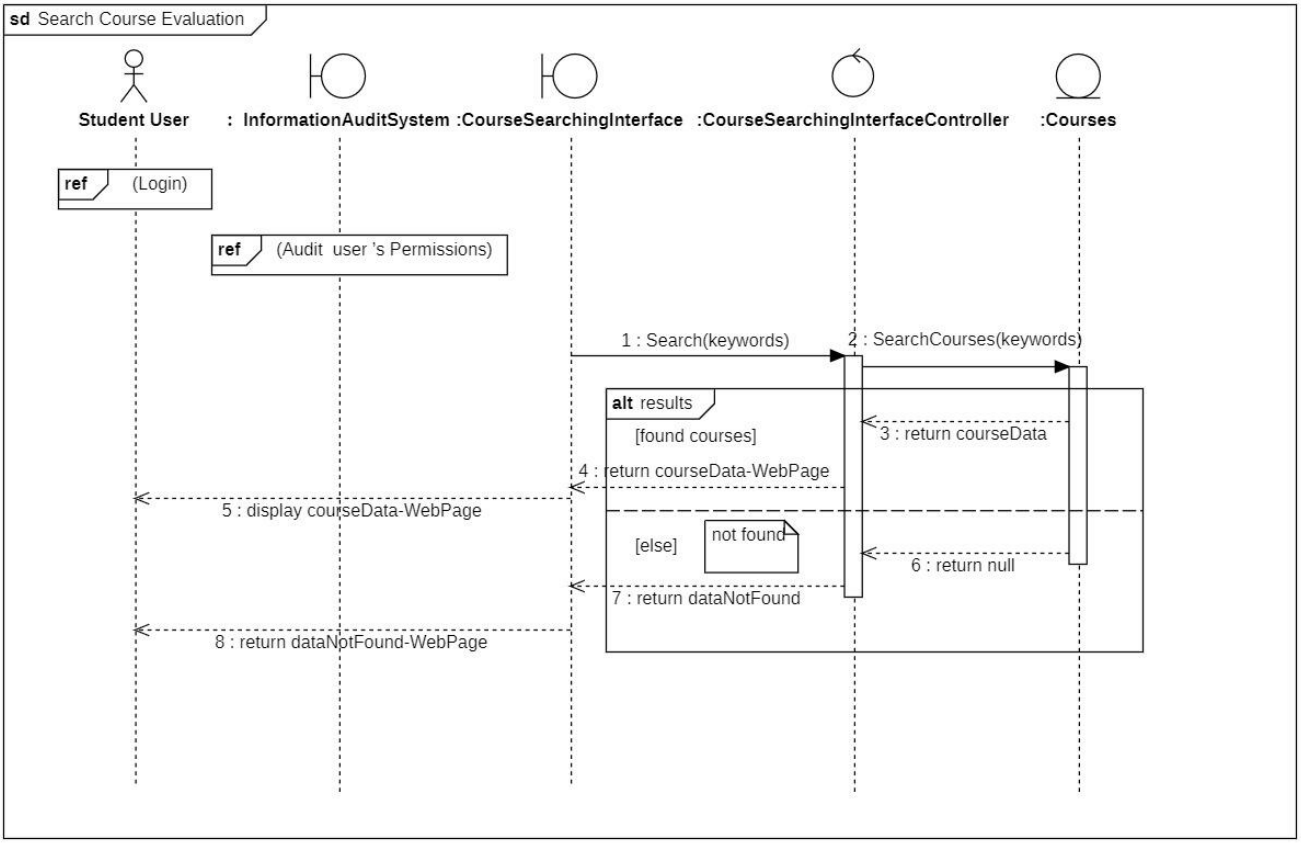


Fig.16 Sequence Diagram of Use Case "Search course evaluation“

The above figure is the sequence diagram of the "Search course evaluation" use case. First, draw the sequence diagram according to the use case specification drawn before and the detailed process shown in the activity diagram. We can see that the user is required to log in before entering the course search. Here we use a **ref** to introduce the login behavior of an external subsystem. After logging in, the user needs to be audited, and the same as passing to our information audit subsystem to operate. When the search is started, the student user enters keywords in the **course searching interface** to search, and then the "**CourseSearchInterfaceController**" searches for related courses in the course entity class, and then returns the search results to the interface, and the interface feeds back the results to the user. At this time, there may be two situations. In most cases, the system searches for relevant courses, so these courses are returned to the interface, and then displayed to the user by the interface; in special cases, it may be because the system course update is not perfect

or the user is critical Improper word search results in no search results and no ideal results for users, so users are provided with a feedback channel to remind system administrators to update courses.

In addition, in order to more clearly explain the classes that have relationships in the order use case perspective, this article also draws the corresponding VoPC diagram for the "Search course evaluation" use case, as shown in the following figure:

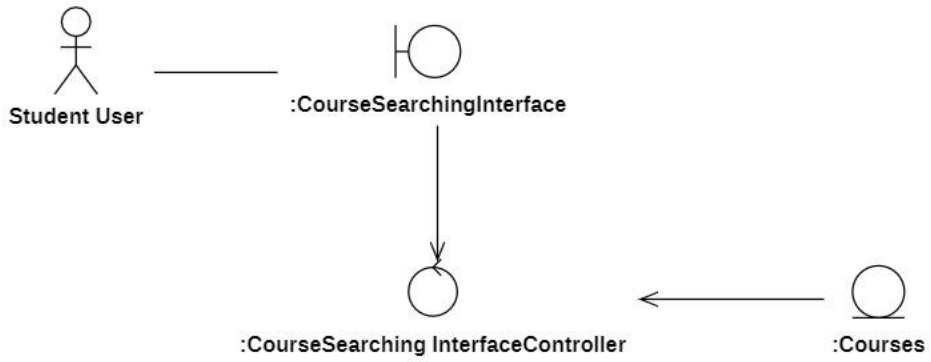


Fig.17 The VoPC Diagram of use case "Search course evaluation"

The following figure is the communication diagram corresponding to the "Search course evaluation" use case, which shows the communication relationship between classes.

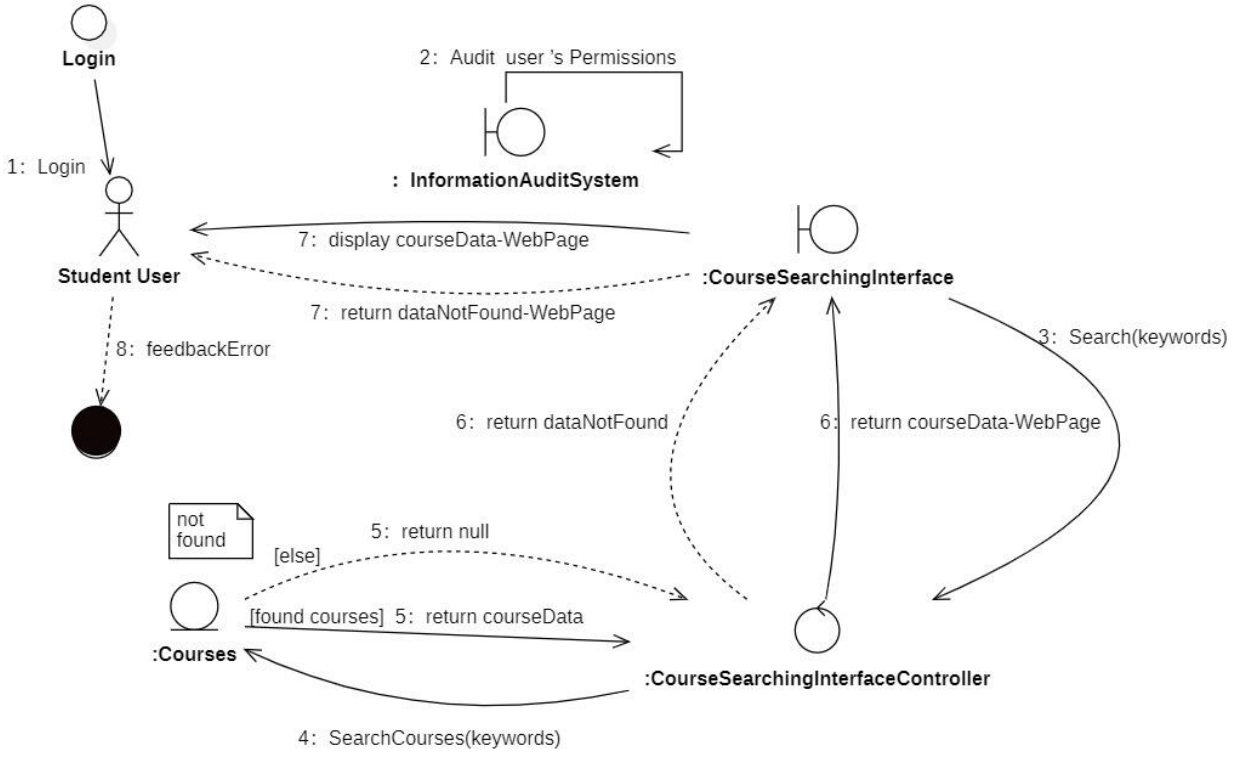


Fig.18 The Communication Diagram of use case "Search course evaluation"

3.2.4. Use Case of ”Update Course Evaluation”

According to the use case specification and activity diagram of the "**Update course evaluation**" use case in the requirement analysis, we extracted the corresponding candidate classes and drawn the corresponding sequence diagram, as shown in the following figure:

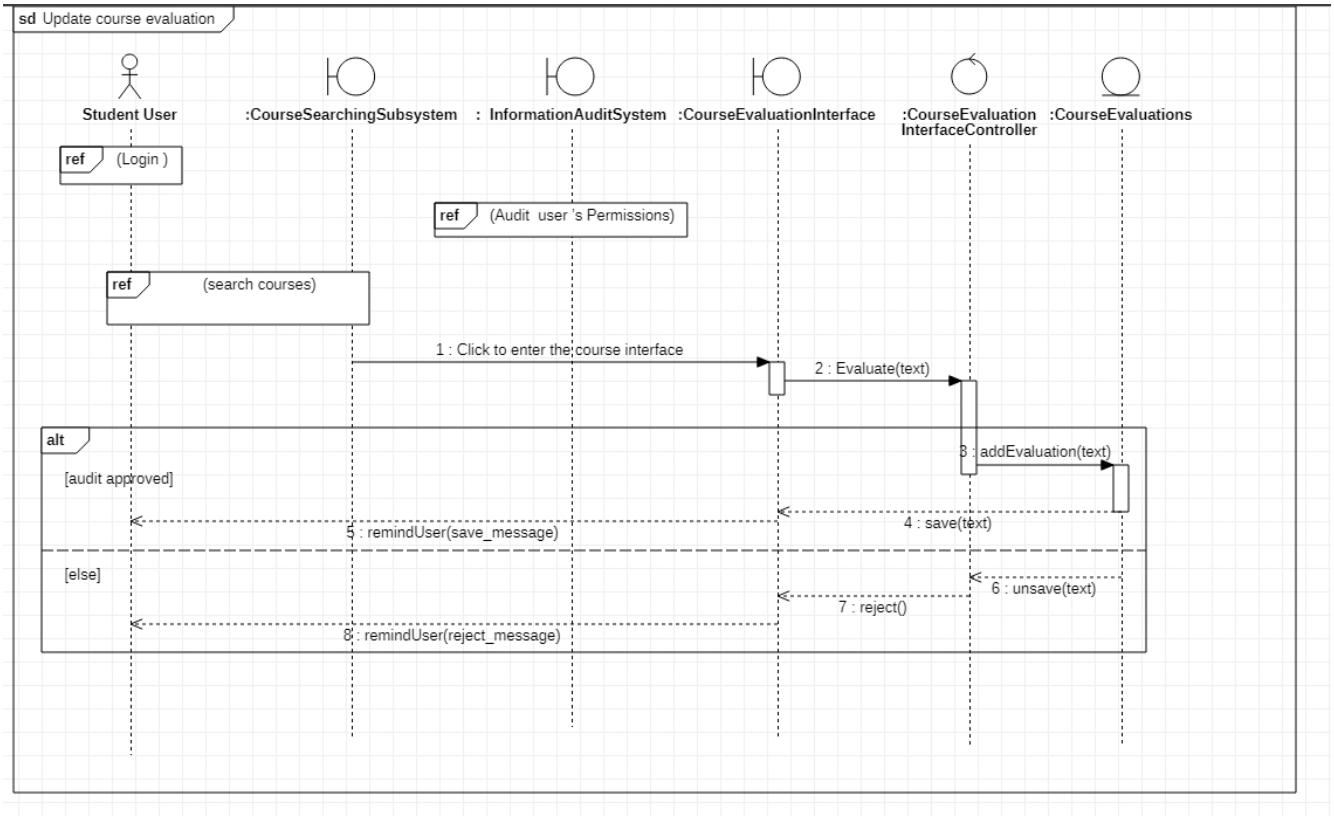


Fig.19 Sequence Diagram of Use Case "Update Course Evaluation“

This is the use case sequence diagram for updating the course evaluation. First of all, it is still necessary to log in and audit, and use ref to introduce login behavior, permission audit behavior, and search behavior of external systems. The search behavior is completed by the course search subsystem of the last use case sequence diagram, which is "Search course evaluation", which mainly reflects the process of evaluation update. Click the evaluation button to enter the course evaluation interface, enter the evaluation text, and "UpdateEvaluationController" will analyze and review the evaluation text. If the review is passed, the evaluation is saved in the "CourseEvaluations" entity class, and a message is sent to the interface to notify the user that the evaluation is successful; if the review is not passed, the Controller will call back and notify the user to inform the reason why it has not passed the review.

In addition, in order to more clearly explain the classes that have relationships in the order use case perspective, this article also draws the corresponding VoPC diagram (use case perspective

enhancement class diagram) for the "Update course evaluation" use case, as shown in the following figure:

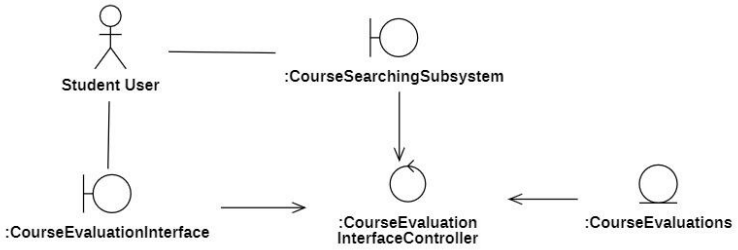


Fig.20 The VoPC Diagram of use case "Update course evaluation"

The following figure is the communication diagram corresponding to the "Update course evaluation" use case, which shows the communication relationship between classes.

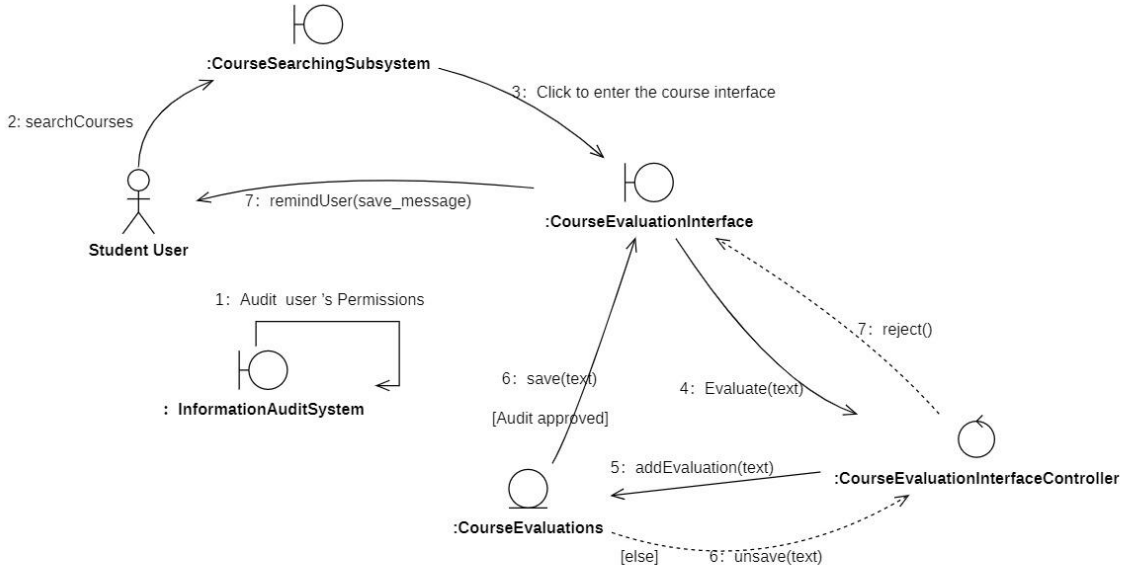


Fig.21 The Communication Diagram of use case "Update Course Evaluation"

4. Updated Use Case Model

4.1. Adjustment and update of the overall use case diagram

When moving from the system requirements analysis stage to the system design stage, we found that the general use case diagram drawn in the requirements analysis stage has the following problems:

- The Generation relationship between some actors is unreasonable.
- The **granularity** between different use cases is not uniform, some use cases' granularity is too small, which makes it difficult to realize some use cases.
- There are functional crossovers between some use cases.
- The boundary between subsystems is relatively **fuzzy.**

According to the above questions, we have merged and deleted some use cases in the general use case diagram, and re-divided the subsystems. The following figure shows the general use case diagram before and after the modification:

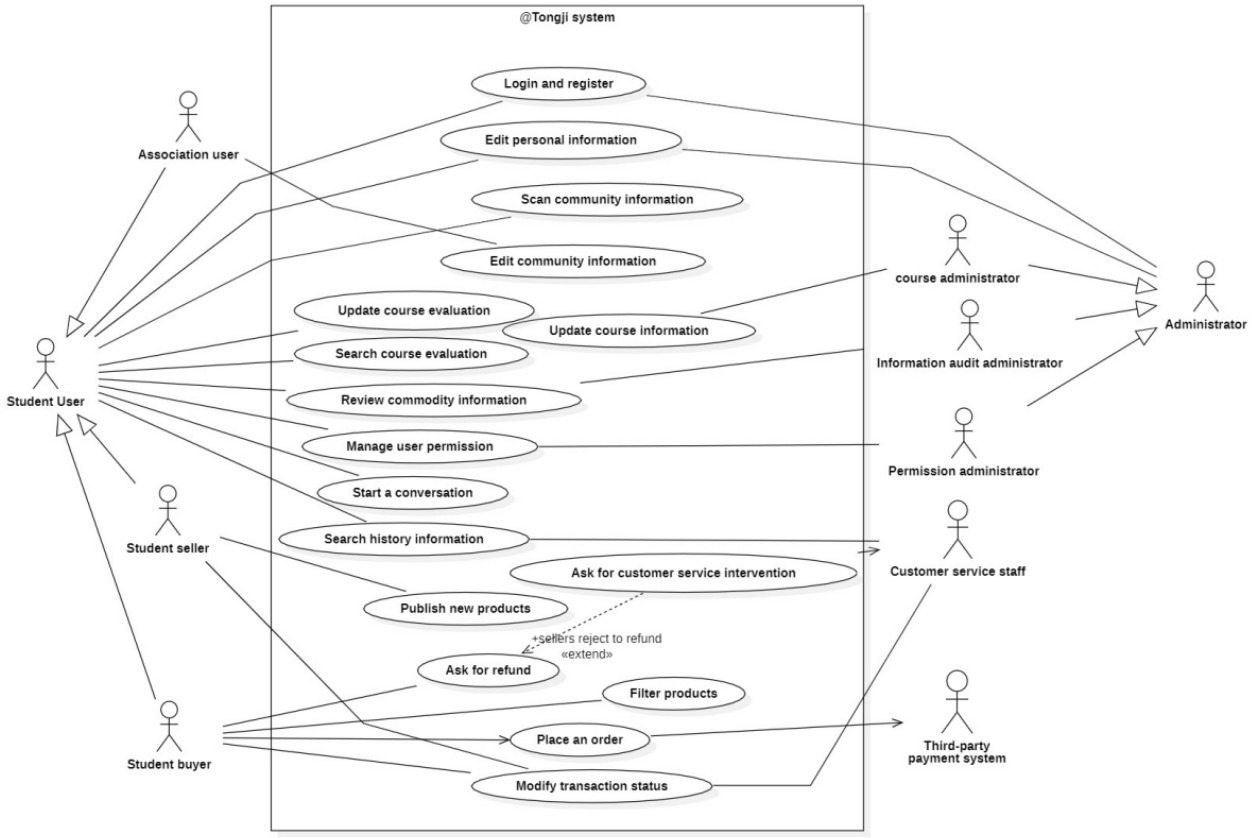


Fig.22 The Original Total Use Case Diagram

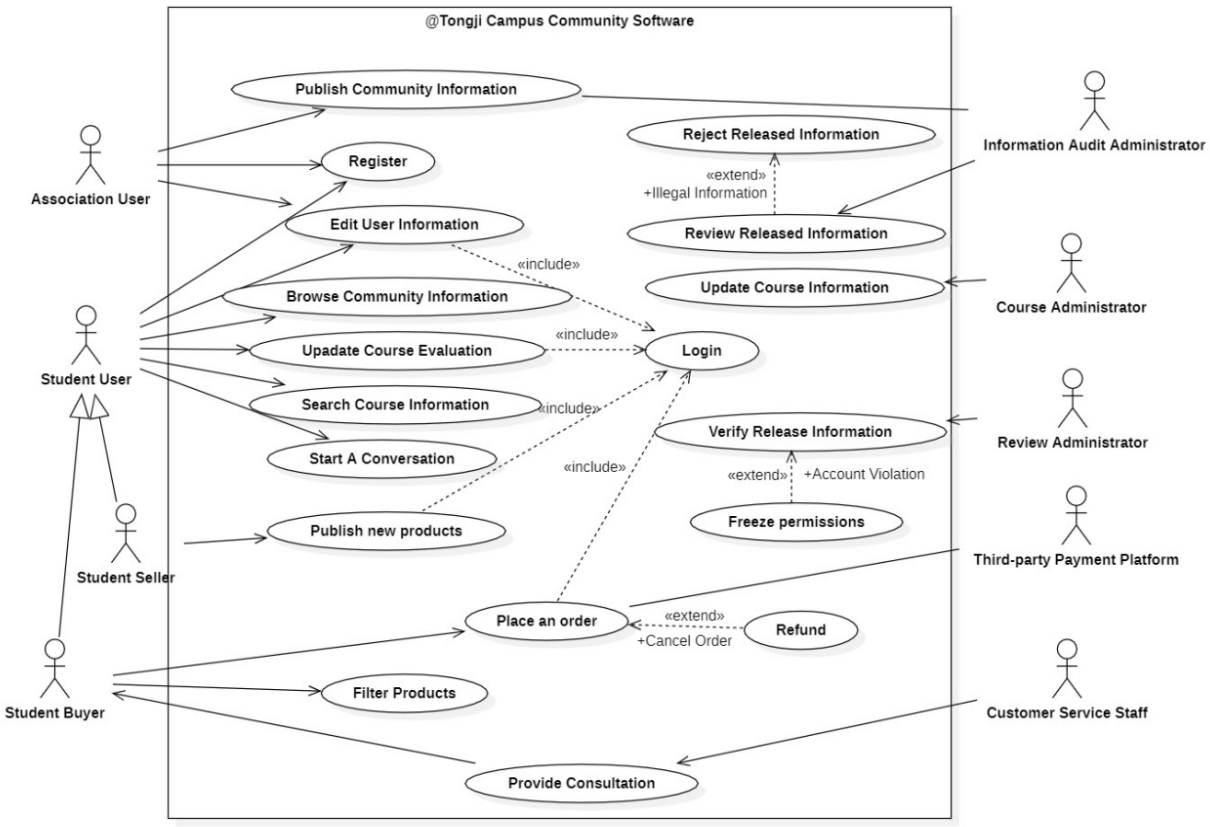


Fig.23 Modified general use case diagram

As shown in the figure above, we have made the following changes to the general use case diagram:

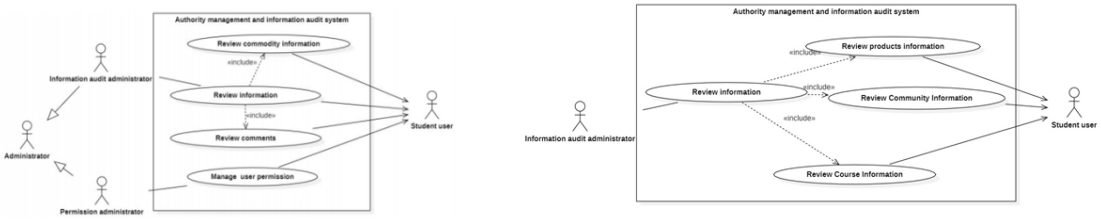
- ✓ Since the login and registration's functions are independent of each other, they are split into two independent use cases
- ✓ Delete some use cases with smaller granularity: Search Course Evaluation, Search History Information
- ✓ Some use cases with unclear requirements description were adjusted: For example, the **Ask for customer service intervention** was modified to **Provide Consultation.**
- ✓ Adjust part of the "extend" relationship to make it more reasonable.

4.2. Update of the use case diagram of the subsystem

4.2.1. Updated use case diagram for the audit system

In the last assignment, we established a permission management and information review system, but the following problems occurred. First of all, we think that permission management is very low-level and can be expanded and extended a lot. Second, we think that permission

management is not particularly suitable for drawing time sequence diagram, there are some difficulties. Therefore, in this operation, we changed the authority management and information audit system into an audit system, and further carried out the specific audit content.



before the update after the update

4.2.2. Updated use case diagram for the second-hand trading function area

In the demand analysis stage, this part mainly has three subsystems, namely: Second-hand Trading Homepage subsystem, Payment Transaction Subsystem, Chatting and After-sale Subsystem; there are two modified subsystems, namely **Second-hand Sales Subsystem, Transaction Management Subsystem**, the specific modification is shown in the figure below:

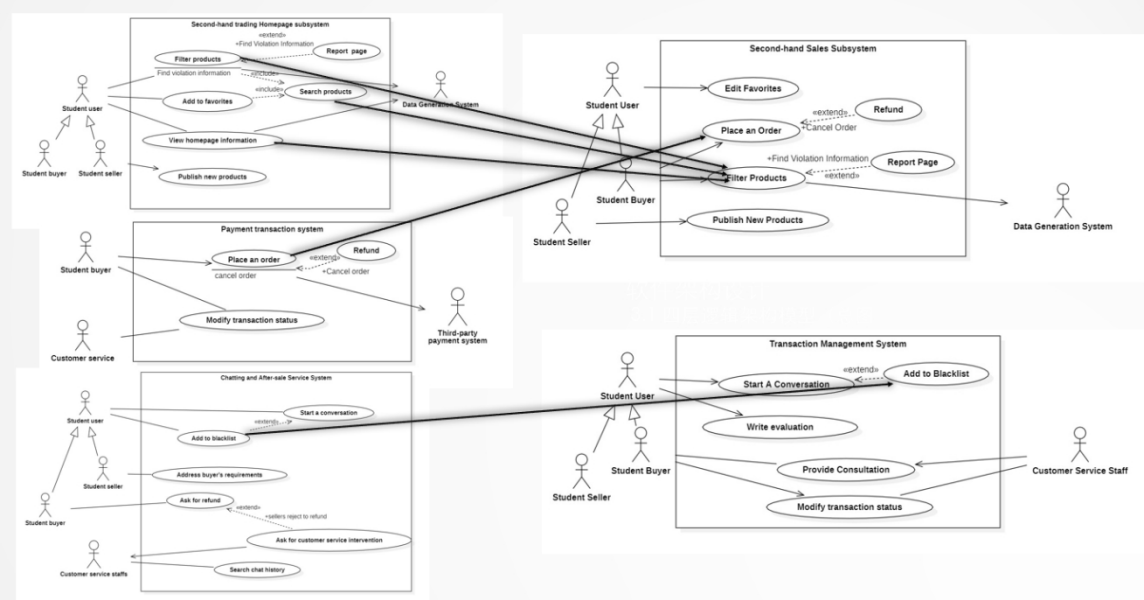


Fig.24 Updated use case diagram for the second-hand trading function area

For the use case diagram in this part, we mainly made the following changes:

First of all, for the use case Filter Products includes the completed functions of Search products and View homepage information, so we merge them into the **"Filter Products"** use case;

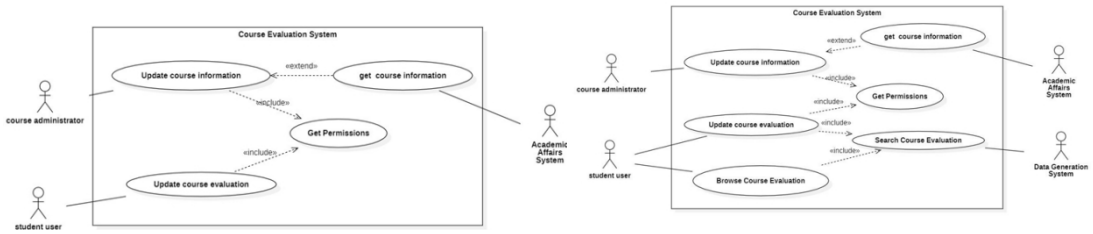
Secondly, because we noticed that the functional area can be roughly divided into **two functional modules** with clear boundaries during the use case realization process, one is the browsing payment product module, the other is the after-sales service module, and the Place an Order use case is related to the browsing relationship of the product. So we merge the above use

cases into a new subsystem: Second-hand Sales Subsytem; and merge the remaining use cases into another subsystem: Transaction Management Subsystem.

The above update of use cases and re-division of subsystems **makes the granularity of use cases more uniform and the boundaries of subsystems are clearer**.

4.2.3. Updated use case diagram for the course evaluation system

This is the use case realization of the course evaluation part. First of all, the update of the use case diagram is reflected in the addition of the search course use case. Since the updated course evaluation and browsing evaluation are based on the search course, the include relationship is added. This makes the use cases more complete, the granularity is more uniform, and the use case relationships are more logical.



before after

5. UI Interface Update

5.1. Updated personal homepage

This page shows the personal homepage of the student user. According to the sequence diagram of the "place an order" use case users need to interact with the order information frequently**.** Therefore, **an order and transaction report module** has been added to the user homepage interface most frequently used by users, so that users can view their transaction records and transaction reports at any time. Follow-up shopping plans and financial management plans have a clearer understanding.

5.2. Updated Second-hand Trading homepage

The updated UI displays the homepage of the second-hand trading function area. In the use case realization stage, we found that the use case of "Searching browsing record" is closely related to this subsystem, so we added browsing records and favorites modules on this page. In addition, the display method of the transaction ranking module has been changed. In summary, we mainly made the following changes:

- ✓ Cancel the "fuzzy search" function module with low demand.
- ✓ Add recently browsed function module.
- ✓ Add the favorite function module.
- ✓ Change the display method of transaction ranking.

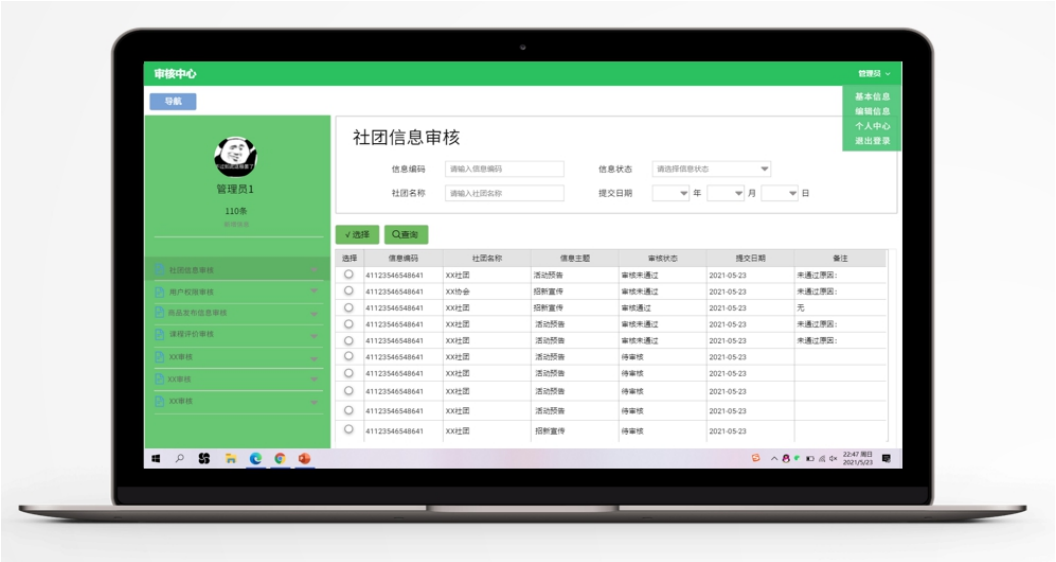
5.3. Updated Order Page

The updated interface is the order details page for filling in the order. In the above-mentioned use case realization process, we made a detailed analysis of the order use case, so we also found that the order function page corresponding to the use case needs to be updated. For this page, we have mainly updated the following:

- ✓ Add the function of choosing payment method.
- ✓ Adding the seller description section on the order submission page allows buyers to clarify the seller's requirements again before placing an order, which indirectly reduces the probability of refunds.
- ✓ Optimized the aesthetics of the interface.

5.4. Add information audit system interface

Our previous UI diagram did not draw the interface of the audit part, but the information audit system is an important part of our @Tongji, so we designed the UI diagram of the information audit part. This interface can only be accessed by the administrator and can only be performed on the computer side (due to the limitations of the mobile phone screen, it is inconvenient to display the audit information).



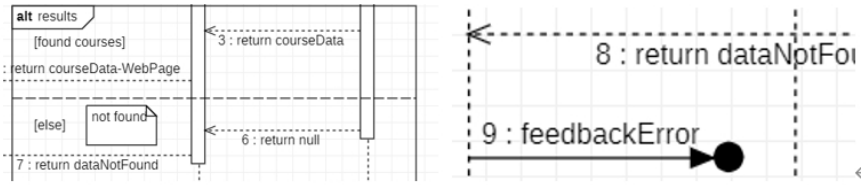
The administrator identity is displayed in the upper right corner of the screen. The administrator can enter the personal homepage, edit the administrator information, log out, and so on. The left part shows the various audits required by the information audit system for each subsystem of @Tongji, including community information audit, user authority audit, product release information audit, course evaluation audit, and so on. The review of whether the user can evaluate the course or not, which will be mentioned later, is also handled by the administrator here.

Take community information review as an example, click "Community Information Review" in the column on the left to enter the community information review interface. The small box in the upper middle part can filter the information in the lower audit list. The two green buttons, "Select" can select and batch process the circles in the lower table; "Query" can filter the information entered by the administrator Inquire.

The table contains some content related to community information, such as information code, community name, information subject, and so on. The review administrator approves the information, and the unreviewed display status is "unreviewed"; the reviewed one has two states: "review passed" and "review failed".The administrator will inform the failure reason in the "Remarks" column of the failed review.

5.5. Add searching results' feedback interface

Combined with the sequence diagram of the Search course evaluation use case in the course evaluation subsystem, we have added the interface that the user appears when there is no search result. We have added links for user feedback. The following two pictures are partial screenshots of the Search course evaluation sequence diagram corresponding to this interface.



As shown in the figure, taking a user search for "XXX 课" as an example, there is no corresponding search result. This may be caused by user input errors (missing words or multiple words, incomplete keywords, etc.) or the system courses are not updated in time. The interface prompts the user to check their input, and provides search links for similar keywords. If the user checks that the keywords are correct, click the "Feedback Error" button to report that there is no result when searching for "XXX 课" and remind the course administrator to update the course in time.

6. References

6.1. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. Craig Larman.

In chapter 13 of the book, the logical hierarchical architecture and the associated UML notation are introduced. The UML Package Diagram can be used as part of the design model to describe the logical architecture. The UML package diagram can also be used as a view in a software architecture document to describe and supplement the architectural constraints and key points documented in the document. The logical architecture is the macro organization of the classes in the software implementation. He organizes the classes in the software into Package, Subsystems and Layer. Layers are very coarse-grained groupings of classes, packages, or subsystems. At the same time, the "higher" layer can invoke the "lower" service and not the "lower" service. UML packages provide a way to organize elements. UML packages can organize any transaction: classes, other packages, use cases, and so on.

6.2. 面向对象技术 UML 教程 王少峰

The class diagram is introduced in the fifth chapter of the book. UML uses class diagrams to describe classes, interfaces, and their relationships. Classes have properties and methods. Classes can be associated, aggregated, generalized, and so on. Association is a kind of semantic connection among model elements. It is the description of links that share common structural characteristics, behavioral characteristics, relations and semantics. Aggregation is a special form of association that expresses the relationship between whole and part of classes. Generalization defines a taxonomic relationship between a general element and a special element. There are three main types of classes in UML: Boundary, Control, and Entity. Boundary classes are located at the boundary between the system and the outside world. Forms, dialogs, reports, and classes that represent communication protocols, classes that interact directly with external devices, and classes that interact directly with external systems are examples of boundary classes. Entity classes hold information to be put into the persistent store. The so-called persistent storage is a database, files and other media can store data permanently. Control classes are classes that do the work of other classes. Each use case usually has a control class, which can also be shared between multiple use cases.

6.3. Requirements Analysis and System Design:Developing InformationSystems with UML. Leszek A. Maciaszek

Requirements Analysis and System Design emphasizes object-oriented techniques with an excellent balance of practical explanations and theoretical insights, and uses a large number of real-world scenarios and the Unified Modeling Language (UML) to bring us closer to concepts such as subsystems, class diagrams, and architectural patterns. We learned the basic knowledge and concept of the system architecture from Chapter 6, in order to do an architecture analysis of our system. In the end, we chose a layered architecture. Chapter 7 focuses on the design of the graphical user interface (GUI). Through this section, we learned the principles of GUI design and applied them to our project to make the UI more user-friendly. You learned the Package Diagram and the use of subsystems from Chapters 3 and 10. As shown earlier, in Section 2 (Architecture Analysis), we divided the business-specific layer into six subsystems and presented their package diagrams. At the same time, these two chapters also provide some useful suggestions for our Domain Model.

6.4. Introduction to Systems Analysis and Design:An Agile,Iterative Approach. John W. Satzinger

Developing interaction diagrams is the core of object-oriented design. The implementation of the use case, which determines which classes cooperate with other class rows by sending messages, is done during the development of the interaction diagram. At design time, you develop two types of interaction diagrams: a cachine diagram and a collaboration diagram. Chapter 11, Section 3, describes how to design using sequence diagrams, and Chapter 11, Section 4, briefly describes collaborative graph adaptive projects for system design that use iterative and agile modeling techniques to simplify the design form. For these types of projects, the corresponding developers sometimes do the analysis, design, and programming work. Those sample designs are often drawn on whiteboards or notes and discarded after the programming is complete. On distributed teams, analysts and designers are in one place and programmers in another -- usually not together. The following sections of Chapter 11 explain in detail the phases and techniques required for use case realization.

7. Group Members' Contributions

Student ID	Name	Contribution
1853572	QiaoLiang	Draw the interaction diagram for "Placing an Order"
		Draw the Class diagram of the second-hand Sales subsystem
		Modify the general use case diagram and the use case diagram of the corresponding subsystem
		Update the snapshot of UI
1854062	ZhiboXu	Draw interaction diagrams for logging in, registering, and reviewing course information
		Draw the Class Diagram of the User Information System
		Modify the use case diagram of the subsystem
		Draw the package diagram of the business layer
1854117	ZhenyuLi	Draw a logical architecture diagram
		Draw a deployment architecture diagram
		Project goal restatement and document layout
1953803	YixiZheng	Draw an interaction diagram of the course evaluation system
		Update the snapshot of UI
1954106	WenChao	Draw MVC architecture diagram