香港城市大學
City University of Hong Kong

Textbook Subscription Management System

Group: Group

Member Name	Sid
BAO Huiwen	57737924
XU Jingzhe	58642146
YANG Hongkun	58953483
YANG Mengyuan	58979960
ZHAO Hongyang	58819522

Department of Data Science

MSc in Data Science

9th December 2024

CONTENT

1. System Overview

1.1 Current Situation Description

Nowadays, some universities still using textbook subscription systems when subscribing textbooks. The portal is accessed through the textbook subscription window on the teacher side of the academic affairs system. The subscription process is teacher application, department review, and academic affairs office review. The textbook subscription system has been designed with reference to the existing textbook subscription system process.

1.2 System Goals

The purpose of this course project design is to realize the information management system, using it to achieve the textbook subscription work of teachers and students, and realize the management of textbooks by the Academic Affairs Office to facilitate the procurement of textbooks.

1.3 System Development Methods

We adopt the system development method of structured life cycle, and the system development is divided into five steps, including system planning, system analysis, system design, system implementation, as well as system maintenance. Through comprehensive research and analysis, we understand the textbook subscription process and plan the tasks of each stage.

2. System Analysis

2.1 System Requirements Analysis

2.1.1 Registration and Login

System administrators, the Academic Affairs Office, teachers, and students can log in to the system and do related operations through their own accounts. Different users log in through different ports, and all personnel can modify the passwords of their accounts.

2.1.2 Administrator Information Management

Administrators can log in to the administrator port to manage the accounts of all users, manage the personal information of the Academic Affairs Office, teachers, classes, and students, and perform add, delete, modify, and query operations.

2.1.3 Textbook Subscription

Teachers or students can log in to the system to apply for textbooks. The completed applications can't be modified or deleted. Each teacher and student can only view their own application records. The Academic Affairs Office can review the applied textbooks and make decisions to reject or approve the application. The system summarizes the records of the Academic Affairs Office's approved applications.

2.1.4 Information View

After teachers and students log in to their accounts, they can view their personal identity information, check the types of existing teaching materials, and view course information.

2.1.5 Textbook Inventory Management

The Academic Affairs Office can manage suppliers, textbooks and corresponding inventory, increase inventory quantity by adding textbooks to the warehouse, and after registering textbooks, the inventory will add corresponding textbook information and set the current inventory to 0.

2.1.6 Teaching Materials Distribution

The Academic Affairs Office will choose whether to successfully ship the approved textbooks. If there is insufficient inventory, the textbooks will not be shipped. After receiving the textbooks, teachers and students will confirm receipt of the textbooks on the corresponding account side. The corresponding inventory quantity will be deducted from the successfully shipped textbooks.

2.1.7 Textbook Ordering

The system will summarize the textbook subscription applications that have passed the review. The Academic Affairs Office will set up a safety stock, and the

system will give textbook subscription suggestions based on the stock, making it easier for the Academic Affairs Office to supplement textbooks.

2.2 Business Process Analysis

This system includes three main businesses, textbook application review module, textbook distribution module, and textbook inventory management module. Teachers and students fill textbook application forms, and the Academic Affairs Office reviews the textbook application forms. If the application is approved, the records of the approved application will be summarized in the textbook application summary table. On the contrary the system will display the application rejection.

The Academic Affairs Office will choose whether to issue the records of the approved application according to the inventory. If the inventory is sufficient, the Academic Affairs Office can choose to issue the textbooks. Teachers and students can choose whether to confirm the receipt of the issued textbooks, and the inventory of the issued textbooks will be reduced accordingly.

The Academic Affairs Office can register the textbooks that have been put into the warehouse to increase the inventory quantity of the system. The textbook ordering suggestion table contains all the inventory and demand information of the undelivered textbooks. By setting the safety inventory, the system will give corresponding subscription suggestions to facilitate the development of textbook subscription.

Draw the business process according to system requirements as shown below:

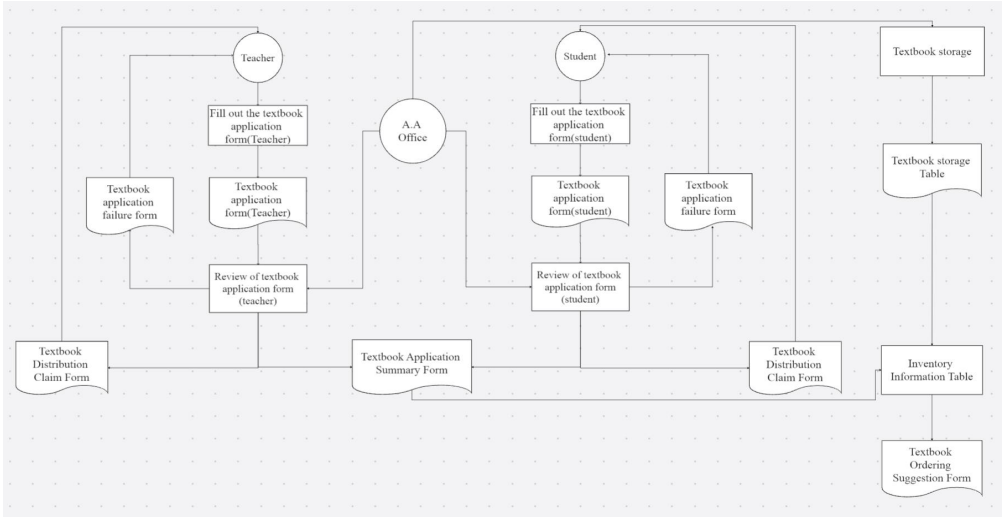


Figure 2.1 Business process diagram

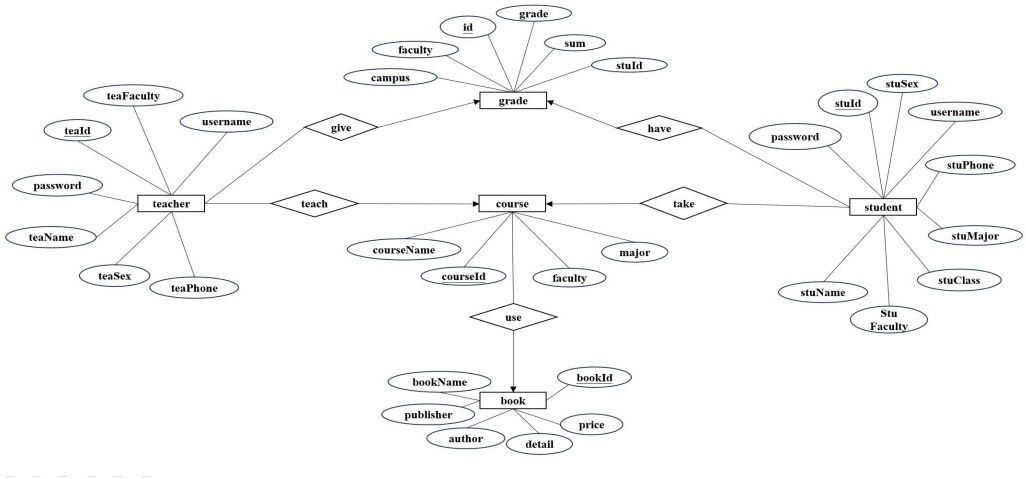


Figure 2.2 ER diagram for Teach model

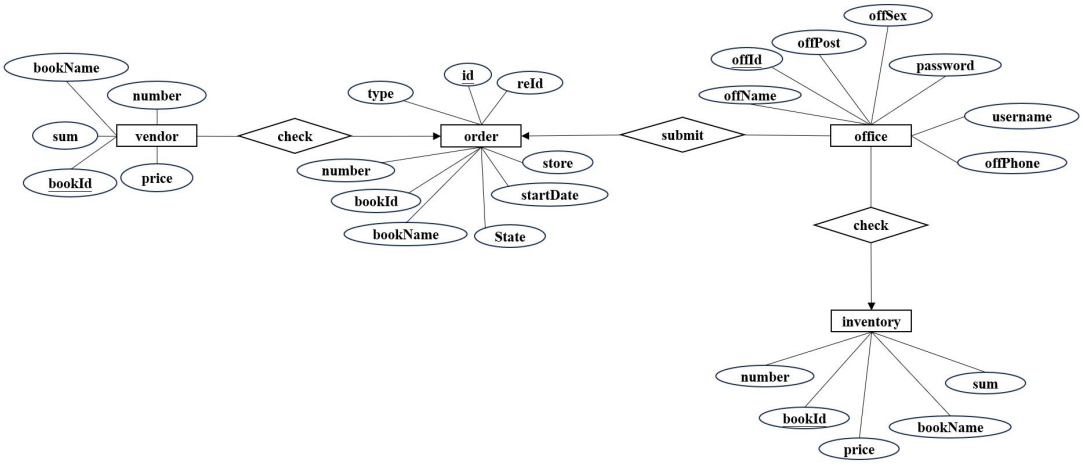


Figure 2.3 ER diagram for book delivery model

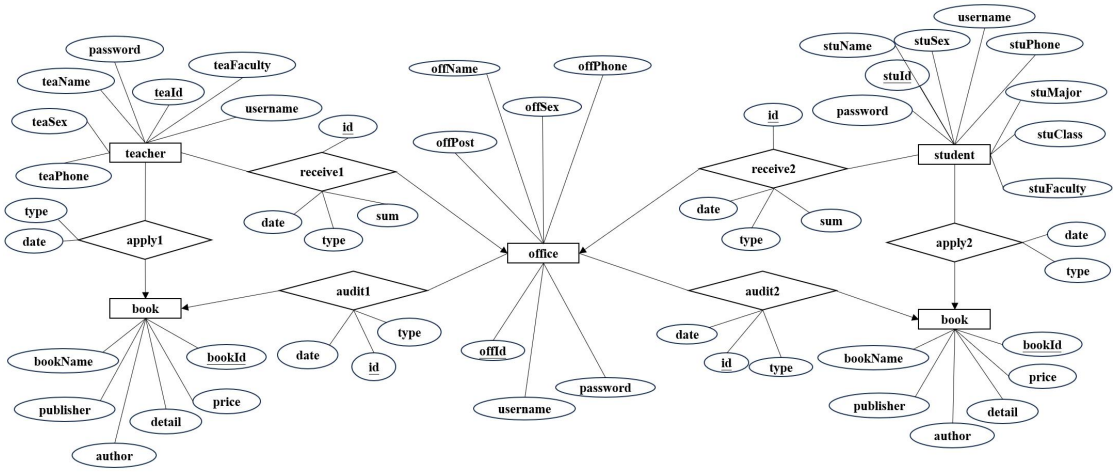


Figure 2.4 ER diagram for book applyment model

2.3 Data Flow Analysis

The following flowcharts are used to draw the data flow diagram:

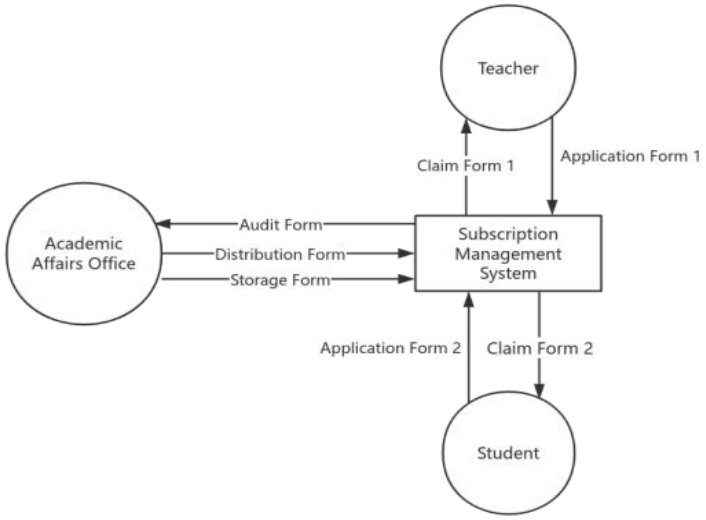


Figure 2.5 Top data flow diagram

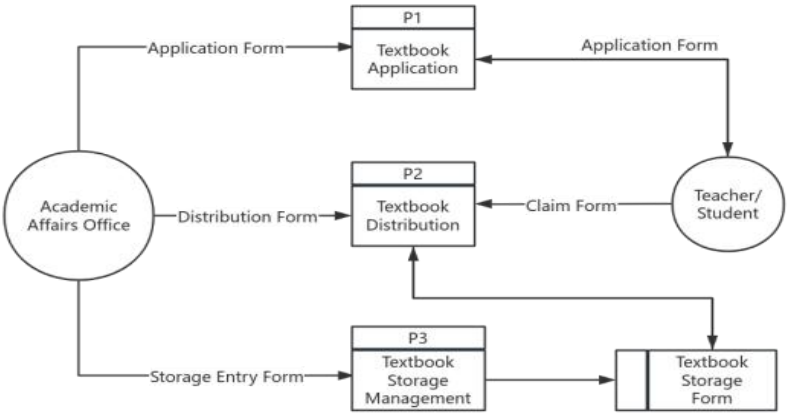


Figure 2.6 Second data flow diagram

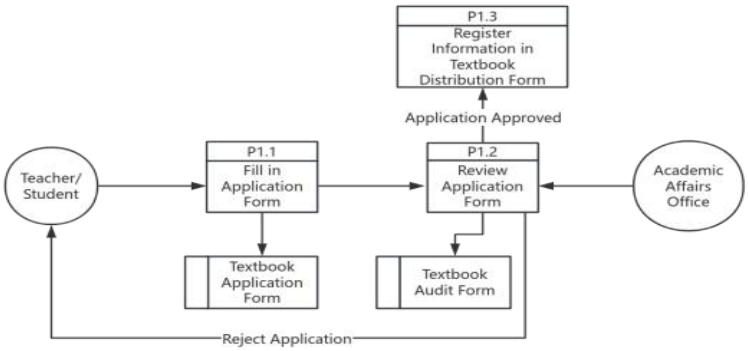


Figure 2.7 Bottom data flow diagram

2.4 Data Dictionary

Data Item Table4-1-1 Data Dictionary No：1-101

Number：101	Name：Application Number	Aliases：id
Brief Description：The number of the application record when teachers and students apply		
Continuous Values	Type：int Length：10	

Data Item Table4-1-2 Data Dictionary No：1-102

Number：102	Name：Textbook Number	Aliases：bookId
Brief Description：Textbook number registered with the Academic Affairs Office		
Continuous Values	Type：int Length：10	

Data Item Table 4-1-3 Data Dictionary No：1-103

Number：103	Name：Course Code	Aliases：courseId
Brief Description：Course number registered with the Academic Affairs Office		
Continuous Values	Type：int Length：10	

Data Item Table 4-1-4 Data Dictionary No：1-104

Number：104	Name：Supplier Number	Aliases：vendorId
Brief Description：Supplier number		
Continuous Values	Type：int Length：10	

Data Item Table 4-1-6 Data Dictionary No：1-106

Number：106	Name：Application Number	Aliases：number
Brief Description：Number of textbooks requested by students/teachers		
Continuous Values	Type：int Length：10	

Data Item Table 4-1-8 Data Dictionary No：1-108

Number： 108	Name：Review Status	Aliases：type1
Brief Description：Review status of student/teacher textbook requests		
Discrete Values	Value	Meaning
	0	Under review/Awaiting review
	1	Approved
	2	Rejected

Data Item Table 4-1-9 Data Dictionary No：1-109

Number： 109	Name：Teacher ID	Aliases：teaId
Brief Description：Teacher ID		
Continuous Values	Type：int Length：10	

Data Item Table 4-1-10 Data Dictionary No：1-110

Number： 110	Name：Student ID	Aliases：stuId
Brief Description：Student ID		
Continuous Values	Type：int Length：10	

Data Item Table 4-1-11 Data Dictionary No：1-111

Number： 111	Name：Shipping Status	Aliases：type2
Brief Description：The delivery status of the textbooks successfully requested by students/teachers		
Discrete Values	Value	Meaning
	0	Waiting for delivery
	1	Delivering
	2	Finished

Data Structure Table 4-2-1 Data Dictionary No：2-201

Number：201	Name：Teacher Textbook Request Form	Aliases：apply1
Brief description: Form for teachers to fill when applying for textbooks		
Composition: Number, textbook number, textbook name, course number, course name, application quantity, application time,Review status, teacher ID		

Data Structure Table 4-2-2 Data Dictionary No：2-202

Number：202	Name：Student Textbook Request Form	Aliases：apply2
Brief description: Form for students to fill when applying for textbooks		
Composition: Number, textbook number, textbook name, course number, course name, number of applications, application time, Review status, student ID		

Data Structure Table 4-2-3 Data Dictionary No：2-203

Number：203	Name：Teachers' Textbooks Review Form	Aliases：audit1
Brief description: Review form of the Academic Affairs Office for teachers' textbooks application		
Composition:Number, textbook number, textbook name, course number, course name, teacher ID, teacher name, Application quantity, textbook unit price, application total price, application time, review status		

Data Structure Table 4-2-4 Data Dictionary No：2-204

Number：204	Name：Student Textbooks Review Form	Aliases：audit2
Brief description: Review form of the Academic Affairs Office for student textbook applications		
Composition: Number, textbook number, textbook name, course number, course name, student class, student ID, student name, Application quantity, textbook unit price, application total price, application time, review status		

Data Structure Table 4-2-5 Data Dictionary No：2-205

Number：205	Name：Textbook Information Table	Aliases：book
Brief description: Form for registering textbooks in the Academic Affairs Office		
Composition: textbook number, textbook name, textbook author, publisher, textbook unit price, details		

Data Structure Table 4-2-6 Data Dictionary No：2-206

Number：206	Name：Course Information Table	Aliases：course
Brief description: Course registration form of the Academic Affairs Office		
Composition: Course number, course name, college offering the course, and course major		

Data Structure Table 4-2-7 Data Dictionary No：2-207

Number：207	Name：Class Information Table	Aliases：grade
Brief description: Table for administrators to manage classes		
Composition:Class number, campus, college, major, class, number of students, student number of the person in charge		

Data Structure Table 4-2-8 Data Dictionary No：2-208

Number：208	Name：Inventory Information Table	Aliases：inventory

Brief description: The Academic Affairs Office views the table for textbook inventory details
Composition:Textbook number, textbook name, inventory quantity, textbook unit price, total inventory price

Data Structure Table 4-2-9 Data Dictionary No：2-209

Number：209 Name：Academic Affairs Office Information Table Aliases：officer
Brief description: The Academic Affairs Office views the table for textbook inventory details
Composition: Work number, name of the academic affairs office staff, user name, password, gender, position, contact information

Data Structure Table 4-2-10 Data Dictionary No：2-210

Number：210 Name：Textbook Subscription Summary Information Table Aliases：order
Brief description: A table summarizing student and teacher textbook subscription information
Composition:Number, textbook number, textbook name, order type, person in charge number, Stock quantity, application quantity, order status, issue time

Data Structure Table 4-2-11 Data Dictionary No：2-211

Number：211 Name：Textbook Subscription Suggestion Information Form Aliases：prepare
Brief description: The system gives subscription suggestions based on inventory, orders, and safety stock.
Composition:Textbook number, textbook name, inventory quantity, order requirements, recommended order quantity

Data Structure Table 4-2-12 Data Dictionary No：2-212

Number：212 Name：Teachers' Textbooks Claim Form Aliases：receive1
Brief description: Teachers' claim form for approved textbooks

Composition:Number, textbook number, textbook name, course number, course name, application quantity, application time, order status, teacher ID

Data Structure Table 4-2-13 Data Dictionary No：2-213

Number：213	Name：Student Textbook Claim Form	Aliases：receive2
Brief description: Student claim form for approved textbooks		
Composition:Number, textbook number, textbook name, course number, course name, application quantity, application time, order status, student ID		

Data Structure Table 4-2-14 Data Dictionary No：2-214

Number：214	Name：Warehouse registration table	Aliases：storage
Brief description: The Academic Affairs Office records the textbooks registered in the library		
Composition:Number, textbook number, textbook name, quantity in stock, textbook unit price, textbook total price, supplier number, time of stock entry		

Data Structure Table 4-2-15 Data Dictionary No：2-215

Number：215	Name：Student Information Table	Aliases：student
Brief description: Student system personal information form		
Composition: student ID, user name, password, name, gender, department, major, class, contact information		

Data Structure Table 4-2-16 Data Dictionary No：2-216

Number：216	Name：Teacher Information Table	Aliases：teacher
Brief description: Teacher system personal information table		
Composition: Work ID, Username, Password, Name, Gender, Department, Contact Information		

Data Structure Table 4-2-17 Data Dictionary No：2-217

Number：217	Name：Supplier Information Table	Aliases：vendor
Brief description: Supplier information table registered by the Academic Affairs Office		
Composition: supplier number, supplier name, contact name, contact information		

Data Flow Table 4-3-1 Data Dictionary No：3-301

Number： 301	Name：Teacher/Student Textbooks Request Table	Aliases：apply
Brief description: Textbook application records are synchronized into the review table		
Composition: number, textbook number, textbook name, course number, course name, teacher ID, teacher name,application quantity, textbook unit price, application total price, application time, review status		
If Data Flow	Source: Teachers/Students	Destination: Academic Affairs Office

Data Flow Table 4-3-2 Data Dictionary No：3-302

Number： 302	Name：Textbook Subscription Summary	Aliases：order
Brief description: Textbook claim records are synchronized with the subscription and distribution textbook table		
Composition: number, textbook number, textbook name, course number, course name, teacher ID, teacher name, application quantity, textbook unit price, application total price, application time, review status		
If Data Flow	Source: Academic Affairs Office	Destination: Teacher/Student

Operation Table 4-4-1 Data Dictionary No：4-401

Number：401	Name：Fill the textbook request form
Input: Data content in the textbook application form	
Output: Textbook review form	

Brief description: Implement the textbook subscription review process
Processing logic: Fill out the textbook application form, and the Academic Affairs Office will review the content of the textbook application, with two options: approval and rejection.

Operation Table 4-4-2 Data Dictionary No：4-402

Number：402	Name：Modify the textbook subscription summary table
Input: Data content in the textbook subscription summary table	
Output: Textbook claim form	
Brief description: Implement the process of collecting and distributing textbooks	
Processing logic: If the inventory is sufficient, the teaching materials will be issued and the order status will change accordingly.	

2.5 Testing Plan and Method

2.5.1 Testing Methods and Strategies

2.5.1.1 Test Type

This system uses a black box testing method, which only inputs business data to observe whether it meets the expected output results.

(1) Test Case Design

Test cases will be designed based on actual business requirements and system functional modules, including normal scenarios, abnormal scenarios, edge scenarios, and load testing scenarios, so as to cover all possible business processes and functional points as much as possible.

(2) Test Data

The test data will be designed and constructed for different functional modules of the system, including student information, textbook list, subscription information, order information, etc.

(3) Test Environment

The test environment will be constructed according to actual conditions,including hardware, software and network environment to ensure the stability and accuracy of the test.

2.5.1.2 Test Items

(1) System functions

System functions include student information management, textbook subscription management, order management, inventory management, system settings, etc.

(2) System requirement

It will be classified and tested according to the project requirements document, including functional requirements, performance requirements, reliability requirements, security requirements, etc.

(3) Software architecture

The software architecture will be tested according to the actual development situation, including front-end interface, back-end business logic, database design, data interaction, etc.

(4) Integration test results

This system will involve overall testing of multiple modules and components, including system integration, performance testing, functional integrity testing, etc.

(5) Acceptance criteria

Acceptance criteria will be tested according to the project requirements document and contract requirements, including system performance, functional correctness, stability, security, etc.

2.5.1.3 Test Plan

Information Management Module	
Goals Description	Test methods
'ADD' Button	Click the ADD button, enter data, and add a new data
'DELETE' Button	Click the DELETE button to delete the corresponding record data.
'UPDATE' Button	Click the UPDATE button to modify the corresponding record content
'SEARCH' Button	Enter the corresponding information to see if fuzzy search can be achieved

Application Review Management Module	
Goals description	Test methods
Adding Audit Records	Click Add in the textbook and student interface and fill in the corresponding application record. The Academic Affairs Office will display the corresponding application record simultaneously. Check whether the applicant's ID and name are displayed in the application record.
'Pass' Button	Click the 'Pass' button, the application status changes to Pass, the teacher or student end will be synchronized to Pass, and the relevant records will be displayed in the summary table.
'Reject' Button	Click the 'Reject' button, and the application status on both the teacher side and the Academic Affairs Office side will be synchronized as Rejected.
Application record show	Each user's application record only displays his or her application information.

Textbook Inventory Management Module	
Goals description	Test methods
Textbook subscription recommendation Button	Enter the textbook subscription recommendation interface, enter the corresponding safety stock, and return to the subscription recommendation form
'Release' Button	Click the 'Release' button, the textbook order status will show shipped, the textbook inventory information will decrease accordingly, the inventory information on the summary interface will change, and the inventory information and demand information on the subscription recommendation interface will also change synchronously.
'Receive' Button	On the claim interface on the teacher and student sides, click the 'Receive' button, the status changes to Received, and the information on the interface on the Academic Affairs Office side changes to Completed.
Add to inventory	Click the 'Add to Inventory' button for inventory registration and fill in the inventory information. After completing the information, the corresponding textbook inventory will change.

3. System Design

3.1 Modular Structure Design

Module Design Specification:

System name: Textbook subscription management system

Module Number：A-101	Module Name: Textbook Application Review
The upper-level calling module number：A-100	Name of the upper-level calling module: main module
Input: Textbook application form data, textbook review form data	

Output: Textbook subscription summary table data
Processing: Teachers/students fill out the textbook application form, and synchronize the completed content to the regular review form of the Academic Affairs Office. The Academic Affairs Office reviews the application, and there are three review statuses (0 means reviewing, 1 means approved, and 2 means application rejected). If the review is passed, the data will be filled into the textbook subscription summary table.

Module Number：A-102	Module Name：Textbooks distribution
The upper-level calling module number：A-100	Name of the upper-level calling module: main module
Input: Textbook Subscription Summary Form, Textbook Claim Form	
Output: Textbook inventory details table	
Processing: The Academic Affairs Office will decide whether to ship the data that has been reviewed successfully. If the inventory is sufficient, the goods can be shipped and the inventory quantity will be reduced accordingly. If the inventory is insufficient, the goods cannot be shipped.	

Module Number：A-103	Module Name: Textbook Inventory Management
The upper-level calling module number：A-100	Name of the upper-level calling module: main module
Input: Textbook subscription summary table, textbook storage table	
Output: Textbook inventory details table	
Processing: The Academic Affairs Office will review the successfully reviewed data. If the inventory is insufficient, the Academic Affairs Office will check the textbook subscription recommendation table, set the safety inventory, and check the subscription suggestions given by the system. The textbooks that have arrived will be registered for entry, the textbook	

inventory will be changed, and the corresponding inventory will be subtracted from the textbooks that have been successfully shipped to achieve inventory management.

3.2 Database Design

(1) admin

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	10	-	Primary Key	
2	username	username	varchar	20	-		
3	password	password	varchar	20	-		
4	Type	adminType	int	20	-		0-Admin 1-Office 2-Teacher 3-Student

(2) officer

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	work no.	offId	int	20	-	Primary Key	
2	user name	username	varchar	20	-	Foreign Key	Admin-username
3	password	password	varchar	20	-	Foreign Key	Admin-password
4	name	offName	varchar	20	-		
5	sex	offSex	varchar	20	-		
6	position	offPost	varchar	20	-		
7	phonenum	offPhone	varchar	20	-		

(3) grade

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark

1	class id	id	int	20	-	Primary Key	
2	campus	campus	varchar	50	-		
3	faculty	faculty	varchar	50	-		
4	major	major	varchar	50	-		
5	grade	grade	varchar	50	-		
6	sum	sum	int	20	-		
7	student in charge id	stuId	int	20	-	Foreign Key	student-stuId

(4) student

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	student id	stuId	int	20	-	Primary Key	
2	username	username	varchar	20	-	Foreign Key	admin-username
3	password	password	varchar	20	-	Foreign Key	admin-password
4	name	stuName	varchar	50	-		
5	sex	stuSex	varchar	20	-		
6	faculty	stuFaculty	varchar	50	-		
7	major	stuMajor	varchar	50	-		
8	class	stuClass	varchar	50	-		
9	phonenum	stuPhone	varchar	50	-		

(5) teacher

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	teacher id	teaId	int	20	-	Primary Key	

2	username	username	varchar	20	-	Foreign Key	admin-username
3	password	password	varchar	20	-	Foreign Key	admin-password
4	name	teaName	varchar	20	-		
5	sex	teaSex	varchar	20	-		
6	faculty	teaFaculty	varchar	20	-		
7	phonenum	teaPhone	varchar	20	-		

(6) vendor

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	supplier id	vendorId	int	20	-	Primary Key	
2	supplier name	vendorName	varchar				
3	contact name	contact	varchar				
4	phonenum	phone	varchar				

(7) book

No	Name	Field Name	Type	Length	Decimal Places	Primary Key
1	textbook num	bookId	int	20	-	Primary Key
2	textbook name	bookName	varchar	50	-	
3	author	author	varchar	20	-	
4	publisher	publisher	varchar	50	-	
5	price	price	float	10	2	
6	detail	detailtrigger	varchar	100	-	

Trigger book1

```
CREATE TRIGGER `book1` AFTER INSERT ON `book` FOR EACH ROW
begin
```

```
insert into inventory(bookId,bookName,number,price,sum)
values(new.bookId,new.bookName,0,new.price,0);
insert into `prepare`(bookId,bookName,`storage`,needNum,suggestNum)
values(new.bookId,new.bookName,0,0,0);
end
Trigger book2
CREATE TRIGGER `book2` AFTER UPDATE ON `book` FOR EACH ROW
begin
update    inventory
set bookName=new.bookName,
price=new.price
where bookId=new.bookId;
update `prepare`
set bookName=new.bookName
where bookId=new.bookId;
end
```

(8) course

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key
1	course id	courseId	int	20	-	Primary Key
2	course name	courseName	varchar	50	-	
3	faculty	faculty	varchar	50	-	
4	major	major	varchar	50	-	

(9) apply1

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	

2	texkbook id	bookId	int	20	-	Foreign Key	Book -bookId
3	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	course id	courseId	int	20	-	Foreign Key	Course -courseId
5	course name	courseName	varchar	50	-	Foreign Key	Course -courseName
6	applicatio nnumber	number	int	20	-		
7	applicatio ndate	date	datetime	-	-		
8	type	type	int	10	-		
9	teacher id	teaId	int	20	-		Teacher -teaId

Trigger apply101

```
CREATE TRIGGER `apply101` AFTER INSERT ON `apply1` FOR EACH ROW INSERT INTO audit1
(id,bookId,bookName,courseId,courseName,teaId,teaName,number,price,sum,date,type)
SELECT new.id,new.bookId,new.bookName,new.courseId,new.courseName,
        new.teaId,teacher.teaName,
                new.number,book.price, new.number*book.price,
                new.date,new.type
FROM teacher,book
where    new.teaId=teacher.teaId and    new.bookId=book.bookId
```

Trigger apply102

```
CREATE TRIGGER `apply102` AFTER UPDATE ON `apply1` FOR EACH ROW
```

```
IF NEW.type = 1 THEN
  INSERT INTO `order`(id,bookId,bookName,type, reId,store,number,state)
Select new.id,NEW.bookId, NEW.bookName, 0 ,NEW.teaId,
inventory.number,new.number,0
from inventory
where inventory.bookId=NEW.bookId;
update `prepare`
set needNum=needNum+new.number
where new.bookId=bookId;
insert intoreceive1(id,bookId,bookName,courseId,courseName,
sum,date,type,teaId)
values( new.id,new.bookId,new.bookName,new.courseId,new.courseName,
new.number,now(),0,new.teaId);
END IF
```

Trigger apply103

```
CREATE TRIGGER `a12` AFTER UPDATE ON `apply1` FOR EACH ROW UPDATE
audit1 SET
        bookId = NEW.bookId,
        bookName = NEW.bookName,
        courseId = NEW.courseId,
        courseName = NEW.courseName,
        teaId = NEW.teaId,
        teaName = (SELECT teaName FROM teacher WHERE teaId = NEW.teaId),
        number = NEW.number,
        price = (SELECT price FROM book WHERE bookId = NEW.bookId),
        sum = NEW.number * (
SELECT price FROM book WHERE bookId = NEW.bookId),
        date = NEW.date,
```

type=NEW.type

WHERE id = NEW.id

Trigger apply104

CREATE TRIGGER `apply104` AFTER DELETE ON `apply1` FOR EACH ROW

DELETE FROM audit1 WHERE id = OLD.id

(10) apply2

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	textbook id	bookId	int	20	-	Foreign Key	Book -bookId
3	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	course id	courseId	int	20	-	Foreign Key	Course -courseId
5	course name	courseName	varchar	50	-	Foreign Key	Course -courseName
6	applicationnumber	number	int	20	-		
7	application date	date	datetime	-	-		
8	type	type	int	10	-		
9	student id	stuId	int	20	-		Student -st**uId**

Trigger apply201

CREATE TRIGGER `apply201` AFTER INSERT ON `apply2` FOR EACH ROW

INSERT INTO audit2 (id,bookId,bookName,courseId,courseName,

```
grade,stuId,stuName,number,price,sum,date,type)
SELECT new.id,new.bookId,new.bookName,new.courseId,new.courseName,
student.stuClass,new.stuId,student.stuName,
        new.number,book.price, new.number*book.price,
        new.date, new.type
FROM    student,book
where   new.stuId=student.stuId and   new.bookId=book.bookId
```

Trigger apply202

```
CREATE TRIGGER `apply202` AFTER UPDATE ON `apply2` FOR EACH ROW
UPDATE audit2 SET
        bookId = NEW.bookId,
        bookName = NEW.bookName,
        courseId = NEW.courseId,
        courseName = NEW.courseName,
         grade = (SELECT grade FROM student WHERE stuId = NEW.stuId),
        stuId = NEW.stuId,
        stuName = (SELECT stuName FROM student WHERE stuId = NEW.stuId),
        number = NEW.number,
        price = (SELECT price FROM book WHERE bookId = NEW.bookId),
        sum = NEW.number * (
SELECT price FROM book WHERE bookId = NEW.bookId),
        date = NEW.date,
        type = NEW.type
WHERE id = NEW.id
```

Trigger apply203

```
CREATE TRIGGER `apply203` AFTER UPDATE ON `apply2` FOR EACH ROW
IF NEW.type = 1 THEN
```

```
INSERT INTO    `order`(id,bookId, bookName, type,
reId,store,number,state)
        select new.id,NEW.bookId, NEW.bookName, 1 ,
NEW.stuId,inventory.number,new.number,0
        from inventory
        where inventory.bookId=NEW.bookId;
        update `prepare`
        set needNum=needNum+new.number
        where new.bookId=bookId;
       insert into    receive2(id,bookId,bookName,courseId,courseName,sum,date,type,stuId)
values( new.id,new.bookId,new.bookName,new.courseId,
new.courseName,new.number,now(),0,new.stuId);
```

Trigger apply204

```
CREATE TRIGGER `a23` AFTER DELETE ON `apply2` FOR EACH ROW
DELETE FROM audit2 WHERE id = OLD.id
```

(11) audit1

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	textbook id	bookId	int	20	-	Foreign Key	Book -bookId
3	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	course id	courseId	int	20	-	Foreign Key	Course -courseId
5	course name	courseName	varchar	50	-	Foreign Key	Course -courseName

6	teacher id	teaId	int	20	-	Foreign Key	Teacher -teaId
7	teacher name	teaName	varchar	20	-	Foreign Key	Teacher -teaName
8	applicationnumber	number	int	20	-		
9	price	price	float	20	2	Foreign Key	Book -price
10	sum price	sum	float	20	2		
11	applicationdate	date	datetime	-	-		
12	type	type	int	10	-		

(12) audit2

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	textbook id	bookId	int	20	-	Foreign Key	Book -bookId
3	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	course id	courseId	int	20	-	Foreign Key	Course -courseId
5	course name	courseName	varchar	50	-	Foreign Key	Course -courseName
6	grade	grade	varchar	50	-	Foreign Key	Student -grade
7	student id	stuId	int	20	-	Foreign Key	Student

							-stuId
8	student name	stuName	varchar	20	-	Foreign Key	Student -stuName
9	applicationnumber	number	int	20	-		
10	price	price	float	20	2	Foreign Key	Book -price
11	total price	sum	float	20	2		
12	applicationdate	date	datetime	-	-		
13	type	type	int	10	-		

(13) inventory

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	textbook id	bookId	int	20	-	Foreign Key	Book -bookId
2	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
3	inventory number	number	int	20			
4	price	price	float	20	2	Foreign Key	Book -price
5	inventory price	sum	float	20	2		

Trigger inventory1

```
CREATE TRIGGER `inventory1` AFTER UPDATE ON `inventory` FOR EACH ROW begin
update `order`
set store=new.number
```

where bookId=new.bookId;

update `prepare`

set `storage`=new.number

where bookId=NEW.bookId;

(14) receive1

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	textbook id	bookId	int	20	-	Foreign Key	Book -bookId
3	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	course id	courseId	int	20	-	Foreign Key	Course -courseId
5	course name	courseName	varchar	50	-	Foreign Key	Course -courseName
6	applicationnumber	sum	int	20	-		
7	approved date	date	datetime	-	-		
8	type	type	int	10	-		
9	teacher id	teaId	int	20	-		Teacher -teaId

(15) receive2

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	textboo	bookId	int	20	-	Foreign Key	Book

	k id						-bookId
3	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	course id	courseId	int	20	-	Foreign Key	Course -courseId
5	course name	courseName	varchar	50	-	Foreign Key	Course -courseName
6	applicationnumber	sum	int	20	-		
7	approved date	date	datetime	-	-		
8	type	type	int	10	-		
9	student id	stuId	int	20	-		Student -stuId

(16) Order

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	Textbook Number	bookId	int	20	-	Foreign Key	Book -bookId
3	Textbook Name	bookName	varchar	50	-	Foreign Key	Book -bookName
4	Application Type	type	int	20	-		
5	Person in charge	reId	int	20	-		

6	Textbook Inventory	store	int	20	-		
7	Number of applications	number	int	20	-		
8	Release Status	state	int	20	-		
9	Release Date	startDate	datetime	-	-		

(17) Prepare

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	textbook id	bookId	int	20	-	Foreign Key	Book -bookId
2	textbook name	bookName	varchar	50	-	Foreign Key	Book -bookName
3	inventory	storage	int	20	-		
4	Required quantity	needNum	int	20	-		
5	Suggest number	suggestNum	int	20	-		

(18) Storage

No.	Name	Field Name	Type	Length	Decimal Places	Primary Key	Remark
1	id	id	int	20	-	Primary Key	
2	Textbook Number	bookId	int	20	-	Foreign Key	Book -bookId

3	TextbookNumber	bookName	varchar	50	-	Foreign Key	Book -bookName
4	Quantity in stock	number	int	20	-		
5	Unit price	price	float	20	2		
6	Total price	sum	float	20	2		
7	Supplier Number	vendorId	int	20	-		Vendor -vendorId
8	Warehouse time	date	datetime	-	-		

Trigger storage1

```
CREATE TRIGGER `storage1` AFTER INSERT ON `storage` FOR EACH ROW begin
update inventory
set number=number+new.number
where bookId=new.bookId;
update inventory
set sum=number*price
where bookId=new.bookId;
End
```

Trigger storage2

```
CREATE TRIGGER `storage2` AFTER UPDATE ON `storage` FOR EACH ROW begin
update inventory
set number=number+new.number-old.number
where bookId=new.bookId;
update inventory
set sum=number*price
where bookId=new.bookId;
```

End

3.3 Module Test Plan

During the testing process, each module is tested through test cases, and the test results and defects found are recorded, and the defects are solved in time.

- Information management module: Verify whether the operations of adding, deleting, modifying and checking are carried out smoothly.
- Textbook application module: Verify whether the academic affairs office synchronizes data after the teacher and student submit applications. Whether only the application projects of the applicant are displayed
- Textbook review module: Verify whether the operation of the academic affairs office reviewing textbooks is synchronized to different tables.
- Textbook storage module: Verify whether the textbook inventory changes correctly.
- Textbook subscription module: Verify whether the correct textbook subscription quantity can be given according to the set safety inventory.
- Textbook distribution module: Verify whether the data synchronization and content change of the distribution table and the claim table are realized correctly
- Login module: Verify whether different identities can log in normally.

4.System Development

4.1 System Framework

This system is a distributed application designed to provide efficient and scalable management system development. The backend of the system is built with Spring Cloud framework to implement microservice architecture, while the frontend is based on LayUI framework, providing users with a friendly interface and good interactive experience. The following is a simple framework description.

- Back-end technology stack：

Spring Boot: An open source Java infrastructure for creating independent, production-level Spring-based applications. It simplifies Spring-based application development, and through the principle of "convention over configuration", it allows developers to easily start a new project and reduce the workload required for development, testing, and deployment.

Spring Cloud: A tool set for simplifying distributed system development, building some common patterns in distributed systems (including configuration management, service discovery, circuit breakers, etc.). The core is the Spring framework, and using Spring Boot's automatic configuration, the most simplified distributed application development can be achieved.

MySQL database: A widely used open source relational database management system (RDBMS) that manages databases based on SQL (Structured Query Language).

- Front-end technology stack:

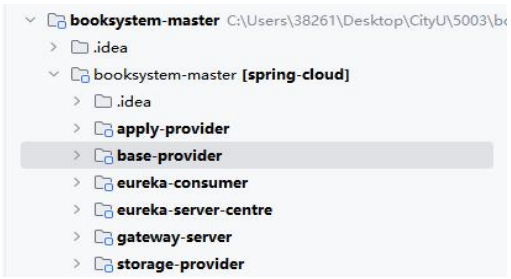
LayUI: A free open source Web UI component library based on the jQuery front-end framework, using its own lightweight modular specifications, following the original HTML/CSS/JavaScript development model, and is very easy to use. It can quickly build a web interface.

4.2 Architecture Description

- Microservice architecture: The system adopts microservice architecture, which divides different business functions into independent service units. Each service runs in an independent process and interacts through a lightweight communication mechanism.
- Service registration and discovery: Eureka is used as the service registration center to realize automatic registration and discovery of services.
- API Gateway: Spring Cloud Gateway is used as the API gateway to uniformly handle external requests and routing forwarding.
- Hystrix: Add Hystrix to the login interface. Hystrix provides a fuse mechanism to prevent the spread of faults and ensure the high availability of the system by isolating service calls, timeout control, fault recovery and other means.

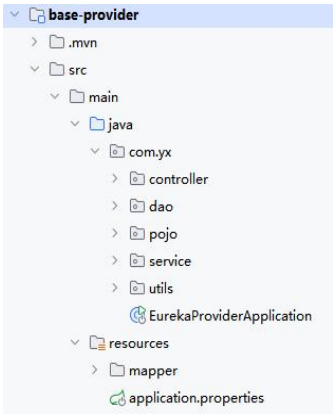
4.3 Code Structure Description

This system uses a distributed architecture design. The code directory is shown in the figure below:



- apply-provider: service provider, implements the code for filling out the textbook application form for teachers and students and the textbook review of the Academic Affairs Office.
- Base-provider: service provider, implements the basic addition, deletion, modification and query of textbooks, courses, teacher personal information, etc., and also includes user login and password modification.
- Eureka-comsumer: service consumer, including front-end development, and calling the interface of other service providers to realize the jump of js page.
- Eureka-server-centre: eureka registration center, used for service registration and discovery.
- Gateway-server: microservice gateway
- Storage-provider: implements the code related to textbook storage management, including textbook storage and textbook distribution

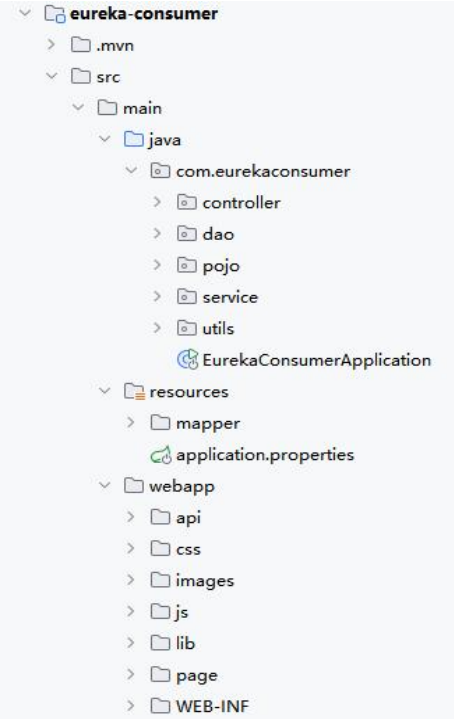
Each module uses springboot. The following is the directory structure of the service provider:



The java directory stores java files, the controller directory writes the control layer code, the dao layer is the data access layer, the pojo directory stores the entity class java files, and the service layer implements the business logic. The resources directory stores resource files, the configuration file is application.Properties, and the

mapper directory contains .xml files that encapsulate SQL statements. EurekaProviderApplication is the main startup class.

The following is the service consumer directory structure：



The front end is written in jsp and is under webapp/WEB-INF on the consumer side of the service. The service interface on the consumer side of the service remotely call the controller layer method of the service provider through openfeign, and the controller on the consumer side of the service implements the service layer method.

For detailed method descriptions of each module, the comments of the control layer will be useful.

4.4 Partial Code Analysis

The following code verifies the username and password, realizes user login, and uses request to obtain the entered username, password, and login type, there are four login types, namely administrator, academic affairs office, student, and teacher. Query the corresponding table to obtain the username and password.

```
@RequestMapping("/updatePwdSubmitStudent")
@ResponseBody
public DataInfo updatePwdSubmitStudent(@RequestBody StudentInfo studentInfo,
```

```
@RequestParam("oldPwd") String oldPwd, @RequestParam("newPwd") String newPwd){
StudentInfo studentInfo1 = studentInfoService.queryStudentInfoById(studentInfo.getStuId());
Admin admin13 = adminService.queryAdminByUsername(studentInfo1.getUsername());
    if (!oldPwd.equals(studentInfo1.getPassword())){
        return DataInfo.fail("The Old Password is not correct");
    }else{
        admin13.setPassword(newPwd);
        adminService.updateAdminSubmit(admin13);
        studentInfo1.setPassword(newPwd);
        studentInfoService.updateStudentInfoSubmit(studentInfo1);}
    return DataInfo.ok();}
```

After logging in, each user can click on the upper right corner to change the password, and the old password and new password need to be entered. The new password you enter not only changes the password in the admin table, but also changes the password in the corresponding user table to realize the synchronous update of the password.

```
HttpSession session = request.getSession();
if(session.getAttribute("type")=="admin"){
    //Administrator
    Admin admin = (Admin)session.getAttribute("user");
    Admin admin1 = adminService.queryAdminById(admin.getId());
    if (!oldPwd.equals(admin1.getPassword())){
        return DataInfo.fail("The Old Password is not correct");
    }else{
        admin1.setPassword(newPwd);
        adminService.updateAdminSubmit(admin1);}}
```

After the student or teacher logs in, the basic personal information is displayed,

the request operation is used to obtain the basic personal information of the user login, the map is used to return to the frontend, and the frontend uses requestScope to obtain the returned value.

```
@RequestMapping("/stuInfo")
public String findStu(HttpServletRequest request, Map<String, Object> map){
    HttpSession session = request.getSession();
    if(session.getAttribute("type")=="student"){
        StudentInfo studentInfo =(StudentInfo) session.getAttribute("user");
        map.put("stuId",studentInfo.getStuId());
        map.put("stuName",studentInfo.getStuName());
        map.put("stuSex",studentInfo.getStuSex());
        map.put("stuFaculty",studentInfo.getStuFaculty());
        map.put("stuMajor",studentInfo.getStuMajor());
        map.put("stuClass",studentInfo.getStuClass());
        map.put("stuPhone",studentInfo.getStuPhone());
    }
    return "student/stuInfo";}
```

Codes for teachers and students to apply for textbooks. Each person can only view their own application information, after the application information is submitted, the system will automatically fill in the applicant's work number into the apply form, only the application information under review can be modified and deleted, and the system will automatically set the type field of the review status to 0 after submitting the application information.

```
public DataInfo addApplyStuSubmit(@RequestBody ApplyStu applyStu,HttpServletRequest request){
    HttpSession session = request.getSession();
    StudentInfo studentInfo = (StudentInfo)  session.getAttribute("user");
    Integer id=studentInfo.getStuId();
    applyStu.setStuId(id);
```

```
    applyStu.setType(0);
    return applyService2.addApplyStuSubmit(applyStu);}
```

The following is the information code for the application of textbooks reviewed by the Academic Affairs Office. Due to the existence of the trigger on the apply table, use queryAuditStuAll to obtain the corresponding record in the corresponding apply table, and click the button to change the type value in the apply table, type=1 means the review is passed, and type=2 means the audit is rejected.

```
public DataInfo updateAuditStuSubmit1(@RequestBody AuditStu auditStu){
    ApplyStu applyStu =   applyService.queryApplyStuById(auditStu.getId());
    applyStu.setType(1);
    applyService.updateApplyStuSubmit(applyStu);
    return DataInfo.ok();
}
```

The following is the code for the textbook summary table. When the inventory is greater than the demand, the Registrar can choose to ship the goods. In this case, the type field of the Shipment Status in the Receive table is changed to 1, and the inventory quantity in the Shipment is displayed minus the corresponding Shipment Quantity, and the inventory quantity displayed in the current table is changed.

```
@RequestMapping("/updateOrderSubmit")
@ResponseBody
public DataInfo updateOrderSubmit(@RequestBody Order order){
    LocalDateTime date = LocalDateTime.now();
    Date now = Date.from(date.atZone(ZoneId.systemDefault()).toInstant());
    Integer id = order.getId();
    if(order.getType()==0){
        ReceiveTea receiveTea=receiveService.queryReceiveTeaById(id);
        receiveTea.setType(1);
        receiveService.updateReceiveTeaSubmit(receiveTea);
```

```
    }
    Inventory inventory = inventoryService.queryInventoryById(order.getBookId());
    Integer number1=order.getStore()-order.getNumber();
    inventory.setNumber(number1);
    inventory.setSum(number1*inventory.getPrice());
    inventoryService.updateInventorySubmit(inventory);
    order.setStore(order.getStore()-order.getNumber());
    order.setState(1);
    order.setStartDate(now);
    orderService.updateOrderSubmit(order);
    return DataInfo.ok();}
```

The following is the relevant code for the receipt form of the teacher and the student, which only shows the issuance status of all the approved textbooks of this user, and you can modify the status of the receipt form to confirm the receipt.

```
public DataInfo updateReceiveTeaSubmit(@RequestBody ReceiveTea receiveTea){
    Order order = orderService.queryOrder(receiveTea.getId(),0);
    order.setState(2);
    receiveTea.setType(2);
    orderService.updateOrderSubmit(order);
    receiveService.updateReceiveTeaSubmit(receiveTea);
    return DataInfo.ok();}
```

The following is the SQL statement for adding, deleting, modifying, and querying the information of the textbook, and other basic operations are similar, in which Query All uses the Select statement to query the relevant operations of the textbook according to the name and number of the textbook.

```
<select id="selectByPrimaryKey" resultMap="BaseResultMap"
parameterType="java.lang.Integer" >
  select *
```

```
    from book
    where bookId = #{bookId,jdbcType=INTEGER}
</select>
<delete id="deleteByPrimaryKey" parameterType="java.lang.Integer" >
    delete from book
    where bookId = #{bookId,jdbcType=INTEGER}
</delete>
<insert id="insert" parameterType="com.yx.pojo.Book" >
    insert into book (bookId, bookName,    author, publisher,
      price, datail)
    values          (#{bookId,jdbcType=INTEGER},          #{bookName,jdbcType=VARCHAR},
#{author,jdbcType=VARCHAR}, #{publisher,jdbcType=VARCHAR},
      #{price,jdbcType=FLOAT}, #{datail,jdbcType=VARCHAR})
</insert>
<update id="updateByPrimaryKey" parameterType="com.yx.pojo.Book" >
    update book
    set bookName = #{bookName,jdbcType=VARCHAR},
      author = #{author,jdbcType=VARCHAR},
      publisher = #{publisher,jdbcType=VARCHAR},
      price = #{price,jdbcType=FLOAT},
      datail = #{datail,jdbcType=VARCHAR}
    where bookId = #{bookId,jdbcType=INTEGER}
</update>
<select                id="queryBookAll"                parameterType="com.yx.pojo.Book"
resultType="com.yx.pojo.Book">
    SELECT * from book
    <where>
      <if test="bookName!=null">
        and bookName like '%${bookName}%'
      </if>
      <if test="bookId!=null">
```

```
        and bookId like '%${bookId}%'
    </if>
  </where>
</select>
```

The following is the main logical SQL statement that recommends ordering textbooks, and modifies the value of suggestNum in the prepare table by obtaining the input safety stock num.

```
<update id="updateNum" parameterType="java.lang.Integer" >
  update prepare
  set suggestNum =
      CASE WHEN storage >= needNum + #{num}
            THEN 0
            ELSE needNum +   #{num} - storage
        END
</update>
```

The following is the SQL statement displayed in the textbook application, so that only all the application information of the current logged-in user can be displayed, and it can be searched according to the bookID and type.

```
<select id="queryApplyTeaAll" parameterType="com.yx.pojo.ApplyTea"
resultType="com.yx.pojo.ApplyTea">
  SELECT * from apply1
  where   teaId = #{teaId,jdbcType=INTEGER}
    <if test="bookId!=null">
      and bookId like '%${bookId}%'
    </if>
    <if test="type!=null">
      and type like '%${type}%'
    </if>
</select>
```

The page is developed based on the LayUI framework, which implements different login groups with different login pages and different interfaces for the sidebar.

```
var options = {
    <c:choose>
        <c:when test="${sessionScope.type.equals('admin')}">
            iniUrl: "${pageContext.request.contextPath}/api/init0.json",
        </c:when>
        <c:when test="${sessionScope.type.equals('officer')}">
            iniUrl: "${pageContext.request.contextPath}/api/init1.json",
        </c:when>
        <c:when test="${sessionScope.type.equals('teacher')}">
            iniUrl: "${pageContext.request.contextPath}/api/init2.json",
        </c:when>
        <c:when test="${sessionScope.type.equals('student')}">
            iniUrl: "${pageContext.request.contextPath}/api/init3.json",
         </c:when>
    </c:choose>
```

Taking the textbook information management interface as an example, the following code implements the operation of deleting data by clicking the modify button, jumping to the modification interface, and clicking the delete button.

```
table.on('tool(currentTableFilter)', function (obj) {
    var data=obj.data;
    if (obj.event === 'edit') {
        var index = layer.open({
            title: 'Modify information',
            type: 2,
            shade: 0.2,
```

```
                maxmin : true,
                shadeClose: true,
                area: ['100%', '100%'],
 content: '${pageContext.request.contextPath}/queryBookById?bookId='+data.bookId,});
            $(window).on("resize", function () {
                layer.full(index); });
        } else if (obj.event === 'delete') {
            layer.confirm('Confirm deletion', function (index) {
                deleteInfoByIds(data.bookId,index);
                layer.close(index);});}});
```

The code for data deletion is as follows.

```
function deleteInfoByIds(ids ,index){
    $.ajax({
        url: "deleteBook",
        type: "POST",
        data: {bookIds: ids},
        success: function (result) {
            if (result.code == 0) {
                layer.msg('Deleted successfully', {
                    icon: 6,
                    time: 500
                }, function () {
                    parent.window.location.reload();
                    var iframeIndex = parent.layer.getFrameIndex(window.name);
                    parent.layer.close(iframeIndex);
                });
            } else {
                layer.msg("Deletion failed");}}})}
```

The following is to add the relevant code of the commit, by changing the

corresponding method of url to locate to the controller layer, the page is reloaded through windows.reload, and the add interface is launched, and the main interface is returned to the table display.

```
form.on('submit(saveBtn)', function (data) {
    var datas=data.field;
    $.ajax({
        url:"addBookSubmit",
        type:"POST",
        //data:datas,
        contentType:'application/json',
        dataType : 'json',
        data:JSON.stringify(datas),
        success:function(result){
            if(result.code==0){
                layer.msg('Added successfully',{
                    icon:6,
                    time:500
                },function(){
                    parent.window.location.reload();
                    var iframeIndex = parent.layer.getFrameIndex(window.name);
                    parent.layer.close(iframeIndex);
                })
            }else{
                layer.msg("Add failed");
            }
```

The following code implements the fuzzy search operation by entering the input box in the input box.

```
var $ = layui.$, active = {
    reload: function(){
```

```
var bookName = $('#bookName').val();
var bookId = $('#bookId').val();
console.log(name)
table.reload('testReload', {
    page: {
        curr: 1
    }
    ,where: {
        bookName: bookName,
        bookId:bookId
    }}, 'data');
```

The following is implemented with templet:function (res) to display different audit statuses based on the value of type.

```
{field: 'type', width: 150, title: 'Review Type', align: "center",templet:function (res) {
        if (res.type == '0'){return '<span class="layui-btn layui-btn-normal layui-btn-xs">Under review</span>';}
        else if (res.type == '1'){return '<span class="layui-btn layui-btn-normal layui-btn-xs" style="background-color: orangered;">Application passed</span>';}
        else if (res.type == '2'){return '<span class="layui-btn layui-btn-normal layui-btn-xs" style="background-color: purple;">Application rejected</span>';}}
```

The following implements the click Pass button 'Pass', and click the Reject button 'Back' to transfer to different URLs to realize the textbook review operation.

```
table.on('tool(currentTableFilter)', function (obj) {
    var data=obj.data;
    if (obj.event === 'pass') {
        $.ajax({
            url:"updateAuditTeaSubmit1",
            type:"POST",
            contentType:'application/json',
```

```
            data:JSON.stringify(data),
            success:function(result){
                if(result.code==0){
                    layer.msg('Modification successful',{
                        icon:6,
                        time:500
                    }, function () {
                        parent.window.location.reload();
                        var iframeIndex = parent.layer.getFrameIndex(window.name);
                        parent.layer.close(iframeIndex);
                    });
                }else{    layer.msg("Modification failed");
else if (obj.event === 'back') {
        $.ajax({
            url:"updateAuditTeaSubmit2",
            type:"POST",
            contentType:'application/json',
            data:JSON.stringify(data),
            success:function(result){
                if(result.code==0){
                    layer.msg('Modification successful',{
                        icon:6,
                        time:500
                    }, function () {
                        parent.window.location.reload();
                        var iframeIndex = parent.layer.getFrameIndex(window.name);
                        parent.layer.close(iframeIndex);});
                }else{ layer.msg("Modification failed");}}
```

5.Syetem Display

5.1 Development Tools

Programming Language：java jdk1.8

Development Tools：IDE

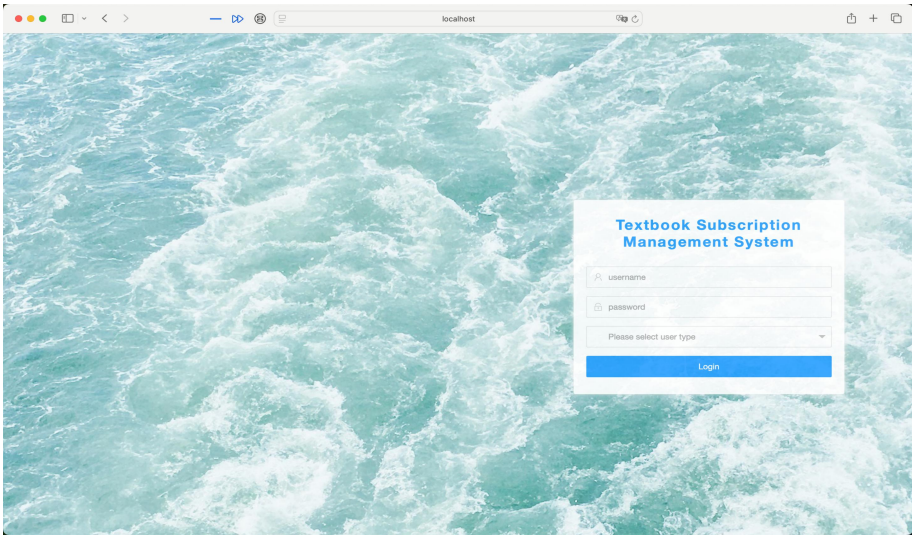
Database：mysql

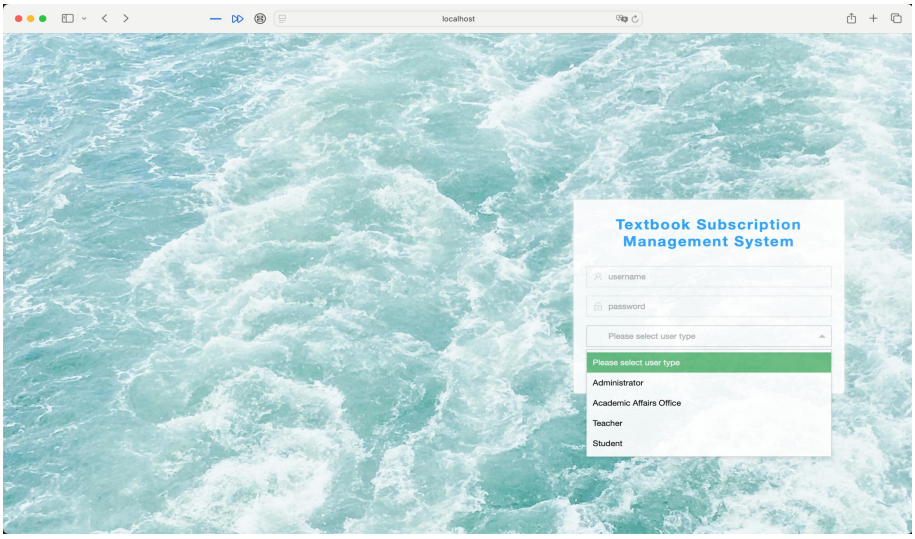
Framework: springcloud+springboot

Front End：layui

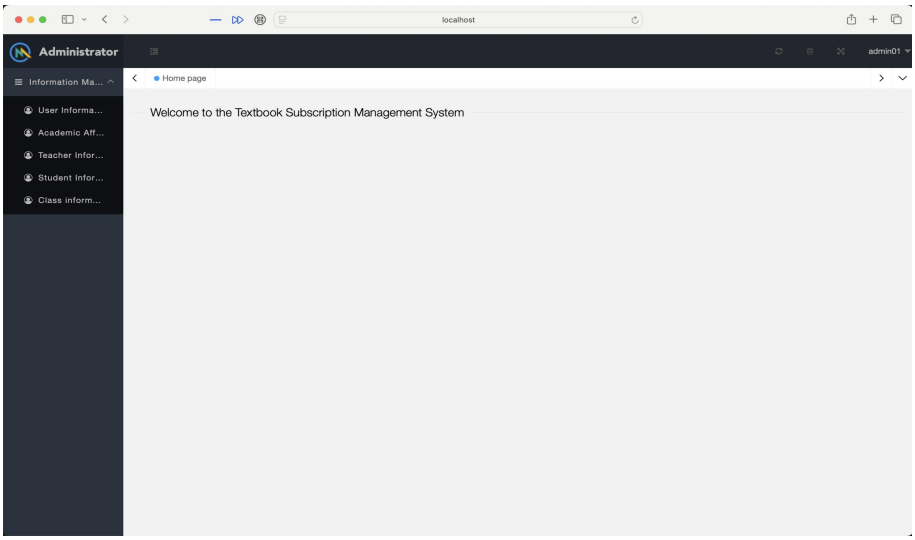
5.2 Software Interface Screenshot

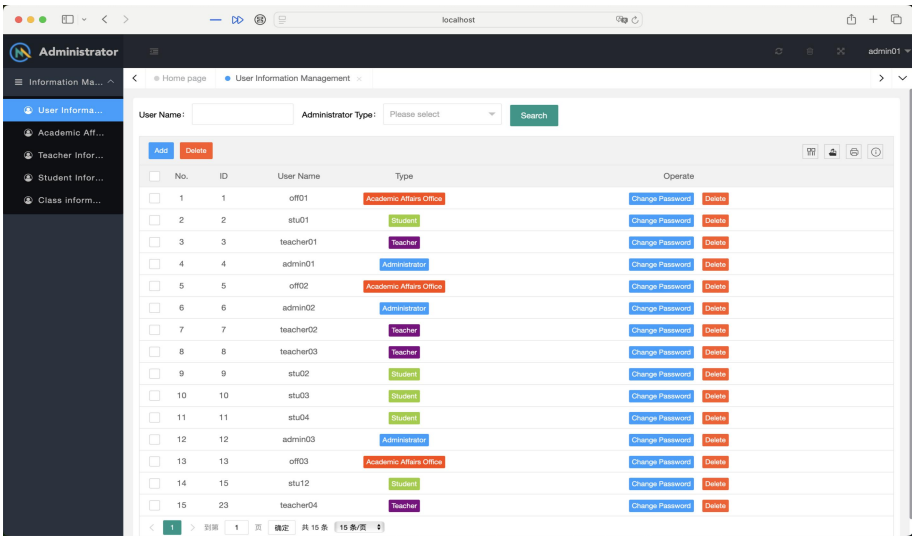
Login interface:

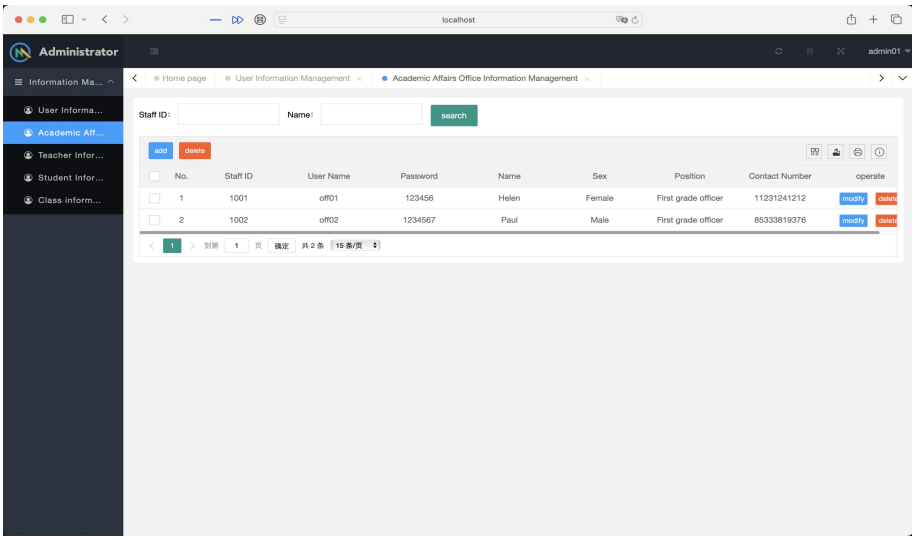


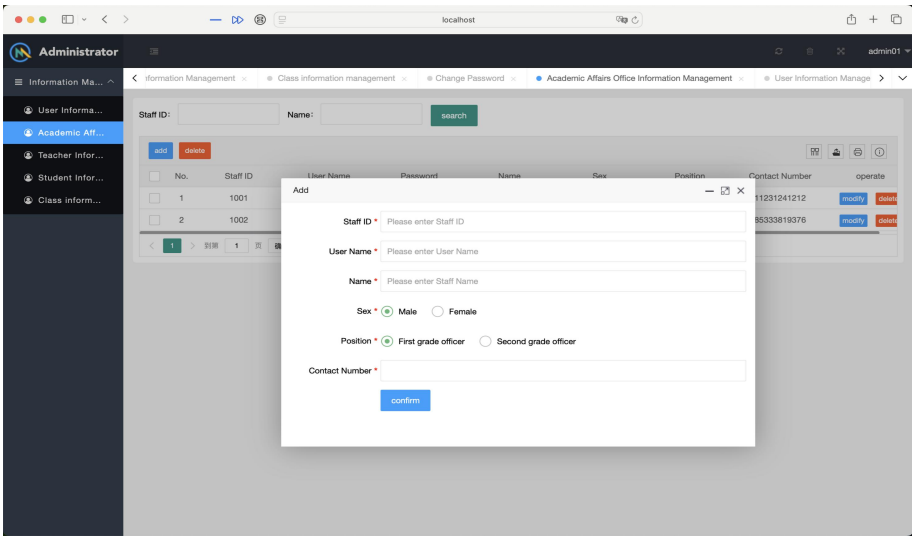


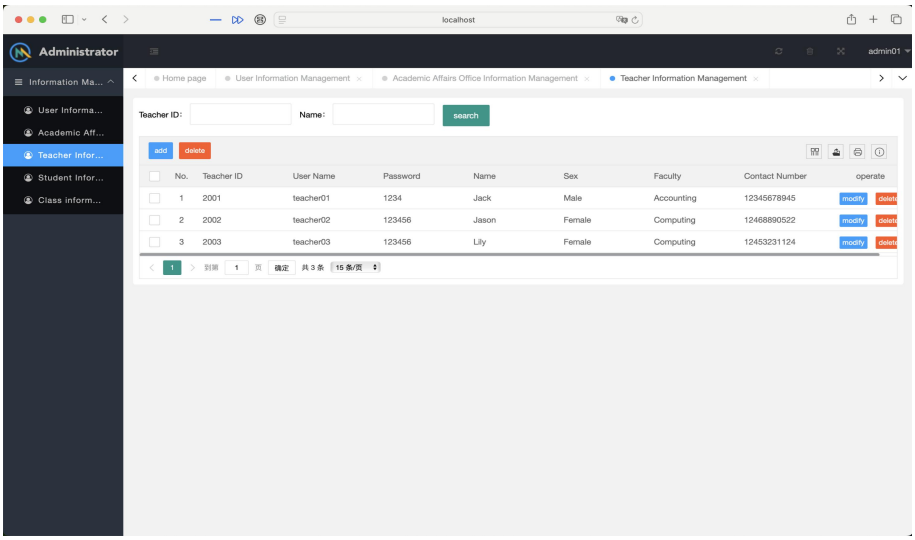
Administrator side：

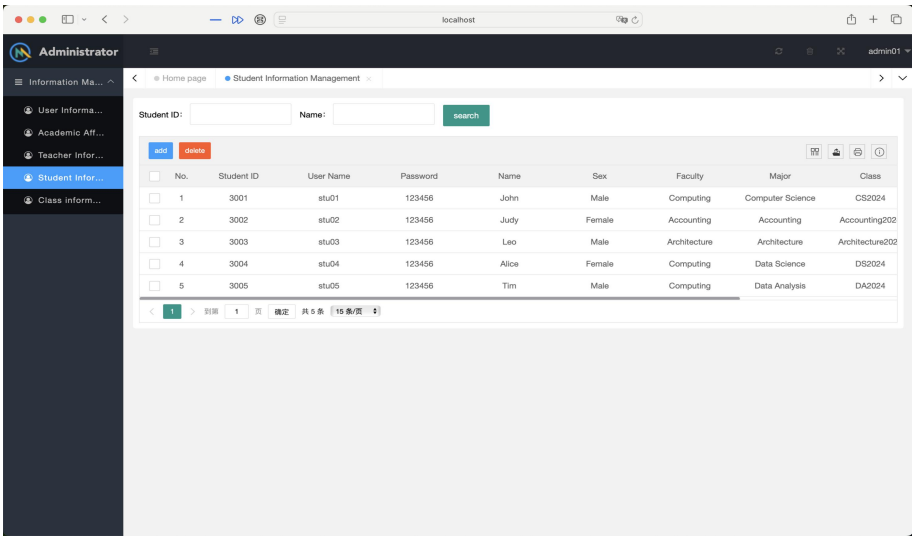

localhost
Administrator
admin01
Information Ma...
Home page
User Informa...
Academic Aff...
Teacher Infor...
Student Infor...
Class inform...
Welcome to the Textbook Subscription Management System

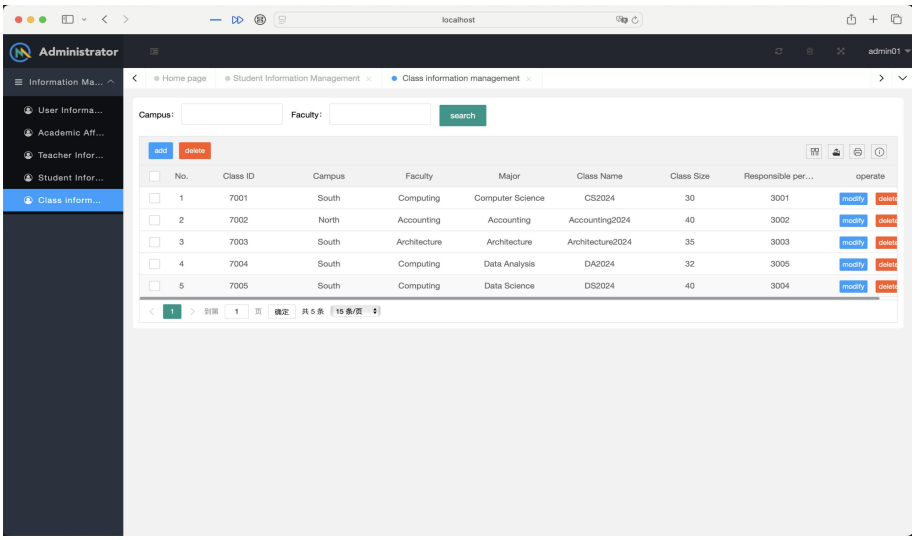

localhost
Administrator
admin01
Information Ma...
Home page
User Information Management
User Informa...
Academic Aff...
Teacher Infor...
Student Infor...
Class inform...
User Name:
Administrator Type:
Please select
Search
Add
Delete
No.
ID
User Name
Type
Operate
Change Password
Delete

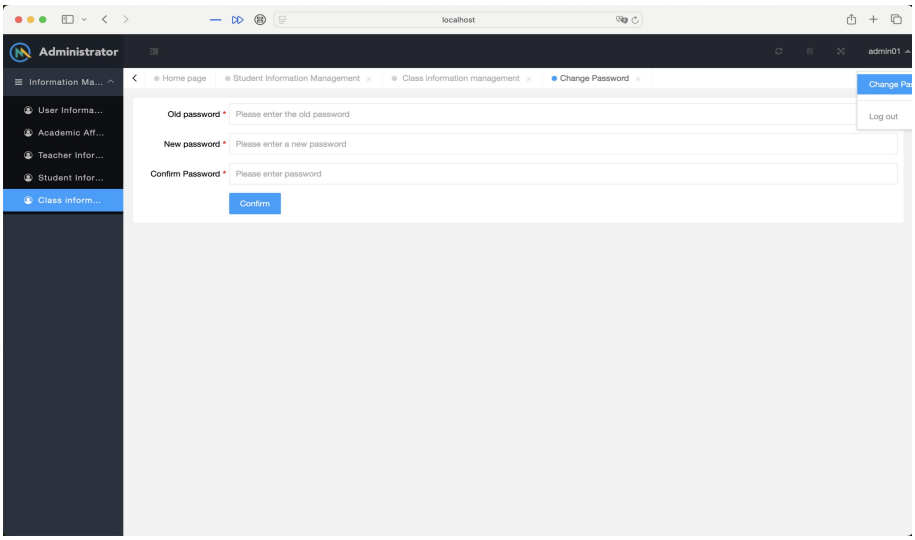

localhost
Administrator
admin01
Information Ma...
Home page
User Information Management
Academic Affairs Office Information Management
User Informa...
Academic Aff...
Teacher Infor...
Student Infor...
Class inform...
Staff ID:
Name:
search
add
delete
No.
Staff ID
User Name
Password
Name
Sex
Position
Contact Number
operate
modify
delete


localhost
Administrator
admin01
Information Ma...
User Informa...
Academic Aff...
Teacher Infor...
Student Infor...
Class inform...
Staff ID:
Name:
search
add
delete
No.
Staff ID
User Name
Password
Name
Sex
Position
Contact Number
operate
1
1001
2
1002
modify
delete
Add
Staff ID *
Please enter Staff ID
User Name *
Please enter User Name
Name *
Please enter Staff Name
Sex *
Male
Female
Position *
First grade officer
Second grade officer
Contact Number *
confirm

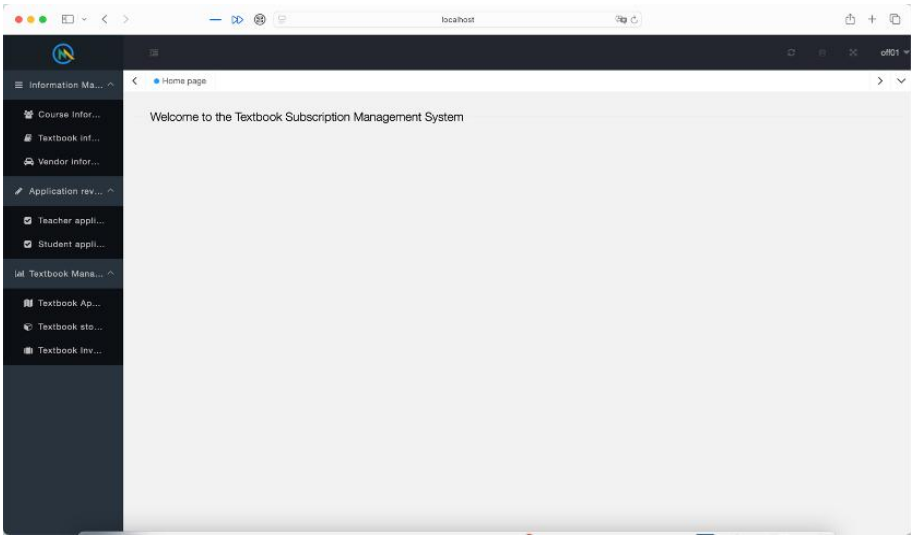

localhost
Administrator
admin01
Information Ma...
User Informa...
Academic Aff...
Teacher Infor...
Student Infor...
Class inform...
Home page
User Information Management
Academic Affairs Office Information Management
Teacher Information Management
Teacher ID:
Name:
search
add
delete
No.
Teacher ID
User Name
Password
Name
Sex
Faculty
Contact Number
operate
1
2001
teacher01
1234
Jack
Male
Accounting
12345678945
2
2002
teacher02
123456
Jason
Female
Computing
12468890522
3
2003
teacher03
123456
Lily
Female
Computing
12453231124
modify
delete
共 3 条
15 条/页

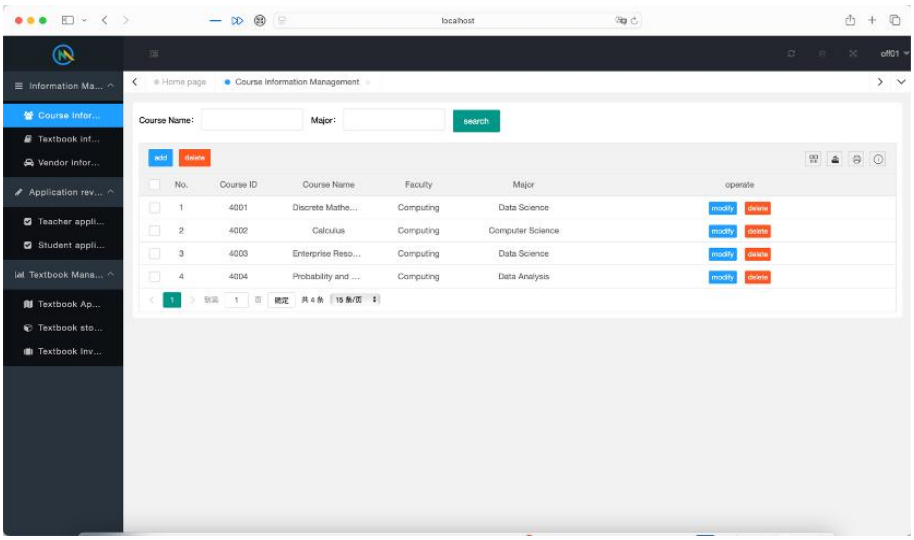

localhost
Administrator
admin01
Information Ma...
User Informa...
Academic Aff...
Teacher Infor...
Student Infor...
Class inform...
Home page
Student Information Management
Student ID:
Name:
search
add
delete
No.
Student ID
User Name
Password
Name
Sex
Faculty
Major
Class
1
3001
stu01
123456
John
Male
Computing
Computer Science
CS2024
2
3002
stu02
123456
Judy
Female
Accounting
Accounting
Accounting202
3
3003
stu03
123456
Leo
Male
Architecture
Architecture
Architecture202
4
3004
stu04
123456
Alice
Female
Computing
Data Science
DS2024
5
3005
stu05
123456
Tim
Male
Computing
Data Analysis
DA2024
共 5 条
15 条/页

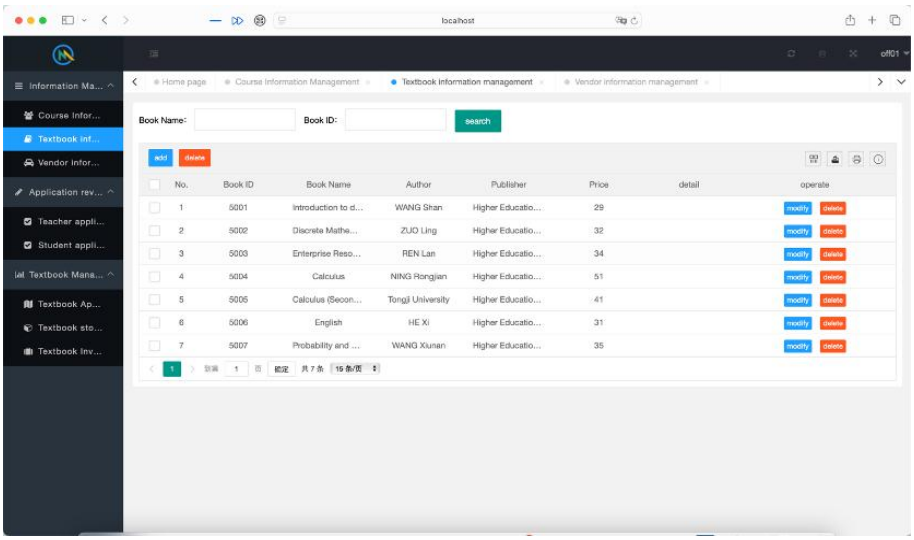


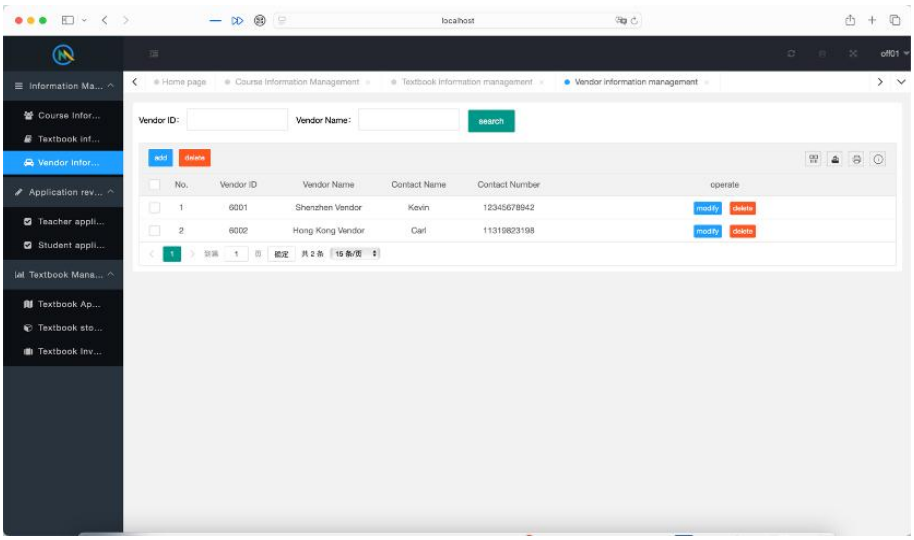


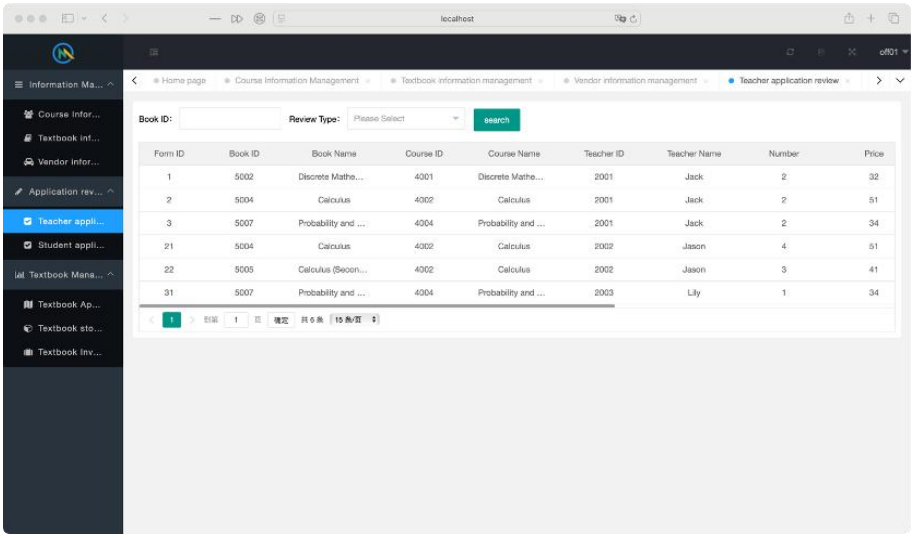
Academic Affairs Office side：

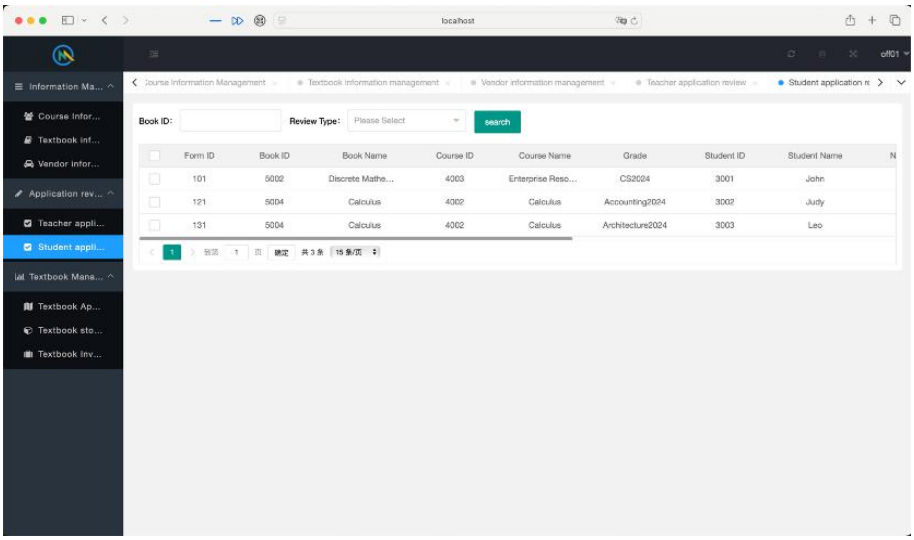


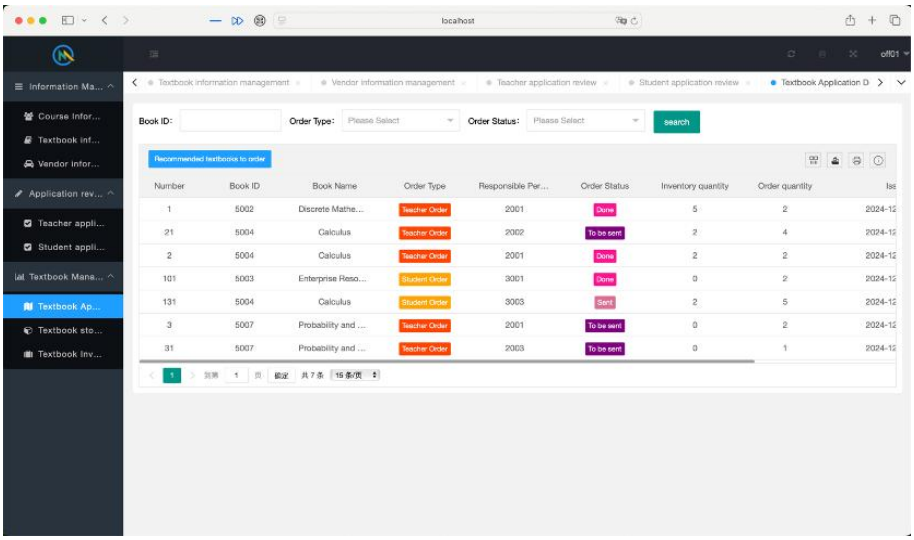

localhost
Information Ma...
Course Infor...
Textbook inf...
Vendor infor...
Application rev...
Teacher appli...
Student appli...
Textbook Mana...
Textbook Ap...
Textbook sto...
Textbook Inv...
Home page
Course Information Management
Course Name:
Major:
search
add
delete
No.
Course ID
Course Name
Faculty
Major
operate
1 4001 Discrete Mathe... Computing Data Science
2 4002 Calculus Computing Computer Science
3 4003 Enterprise Reso... Computing Data Science
4 4004 Probability and ... Computing Data Analysis
modify
delete

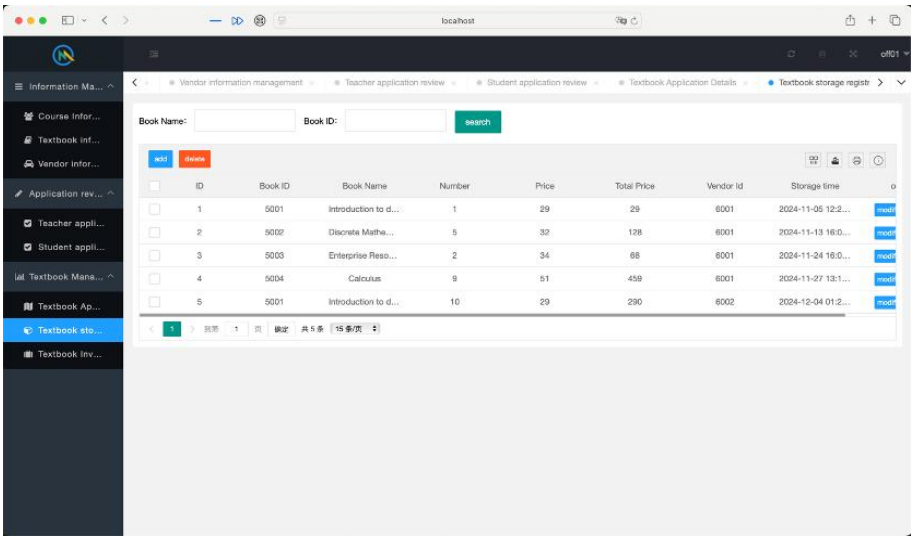

localhost
Information Ma...
Course Infor...
Textbook inf...
Vendor infor...
Application rev...
Teacher appli...
Student appli...
Textbook Mana...
Textbook Ap...
Textbook sto...
Textbook Inv...
Home page
Course Information Management
Textbook Information management
Vendor information management
Book Name:
Book ID:
search
add
delete
No.
Book ID
Book Name
Author
Publisher
Price
detail
operate
1 5001 Introduction to d... WANG Shan Higher Educatio... 29
2 5002 Discrete Mathe... ZUO Ling Higher Educatio... 32
3 5003 Enterprise Reso... REN Lan Higher Educatio... 34
4 5004 Calculus NING Rongjian Higher Educatio... 51
5 5005 Calculus (Secon... Tongji University Higher Educatio... 41
6 5006 English HE Xi Higher Educatio... 31
7 5007 Probability and ... WANG Xiunan Higher Educatio... 35
modify
delete

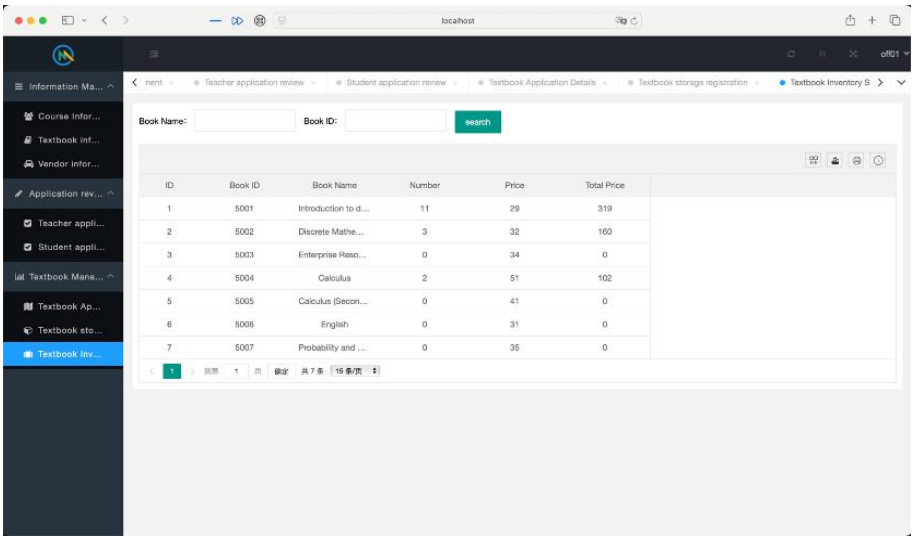

localhost
Information Ma...
Course Infor...
Textbook inf...
Vendor infor...
Application rev...
Teacher appli...
Student appli...
Textbook Mana...
Textbook Ap...
Textbook sto...
Textbook Inv...
Home page
Course Information Management
Textbook Information management
Vendor information management
Vendor ID:
Vendor Name:
search
add
delete
No.
Vendor ID
Vendor Name
Contact Name
Contact Number
operate
1 6001 Shenzhen Vendor Kevin 12345678942
2 6002 Hong Kong Vendor Carl 11319823198
modify
delete


localhost
Information Ma...
Course Infor...
Textbook Inf...
Vendor Infor...
Application rev...
Teacher appli...
Student appli...
Textbook Mana...
Textbook Ap...
Textbook sto...
Textbook Inv...
Home page
Course Information Management
Textbook information management
Vendor information management
Teacher application review
Book ID:
Review Type:
Please Select
search
Form ID
Book ID
Book Name
Course ID
Course Name
Teacher ID
Teacher Name
Number
Price
1 5002 Discrete Mathe... 4001 Discrete Mathe... 2001 Jack 2 32
2 5004 Calculus 4002 Calculus 2001 Jack 2 51
3 5007 Probability and ... 4004 Probability and ... 2001 Jack 2 34
21 5004 Calculus 4002 Calculus 2002 Jason 4 51
22 5005 Calculus (Secon... 4002 Calculus 2002 Jason 3 41
31 5007 Probability and ... 4004 Probability and ... 2003 Lily 1 34

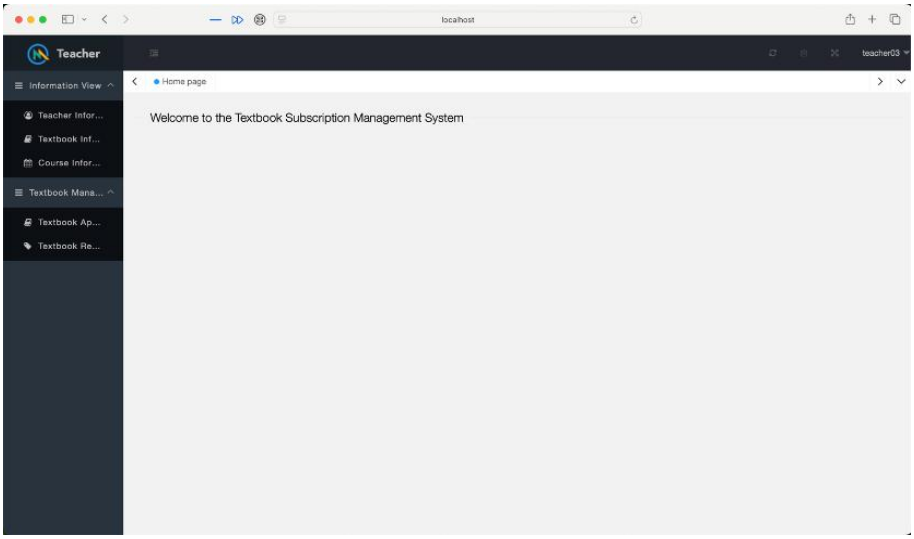

localhost
Information Ma...
Course Infor...
Textbook Inf...
Vendor Infor...
Application rev...
Teacher appli...
Student appli...
Textbook Mana...
Textbook Ap...
Textbook sto...
Textbook Inv...
Course Information Management
Textbook information management
Vendor information management
Teacher application review
Student application r
Book ID:
Review Type:
Please Select
search
Form ID
Book ID
Book Name
Course ID
Course Name
Grade
Student ID
Student Name
101 5002 Discrete Mathe... 4003 Enterprise Reso... CS2024 3001 John
121 5004 Calculus 4002 Calculus Accounting2024 3002 Judy
131 5004 Calculus 4002 Calculus Architecture2024 3003 Leo

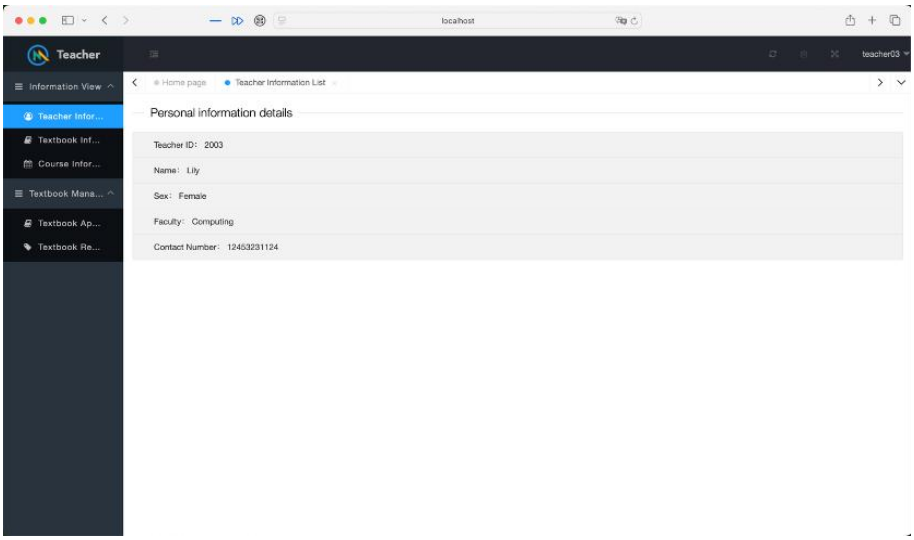

localhost
Information Ma...
Course Infor...
Textbook Inf...
Vendor Infor...
Application rev...
Teacher appli...
Student appli...
Textbook Mana...
Textbook Ap...
Textbook sto...
Textbook Inv...
Textbook information management
Vendor information management
Teacher application review
Student application review
Textbook Application D
Book ID:
Order Type:
Please Select
Order Status:
Please Select
search
Recommended textbooks to order
Number
Book ID
Book Name
Order Type
Responsible Per...
Order Status
Inventory quantity
Order quantity
1 5002 Discrete Mathe... Teacher Order 2001 Done 5 2 2024-12
21 5004 Calculus Teacher Order 2002 To be sent 2 4 2024-12
2 5004 Calculus Teacher Order 2001 Done 2 2 2024-12
101 5003 Enterprise Reso... Student Order 3001 Done 0 2 2024-12
131 5004 Calculus Student Order 3003 Sent 2 5 2024-12
3 5007 Probability and ... Teacher Order 2001 To be sent 0 2 2024-12
31 5007 Probability and ... Teacher Order 2003 To be sent 0 1 2024-12

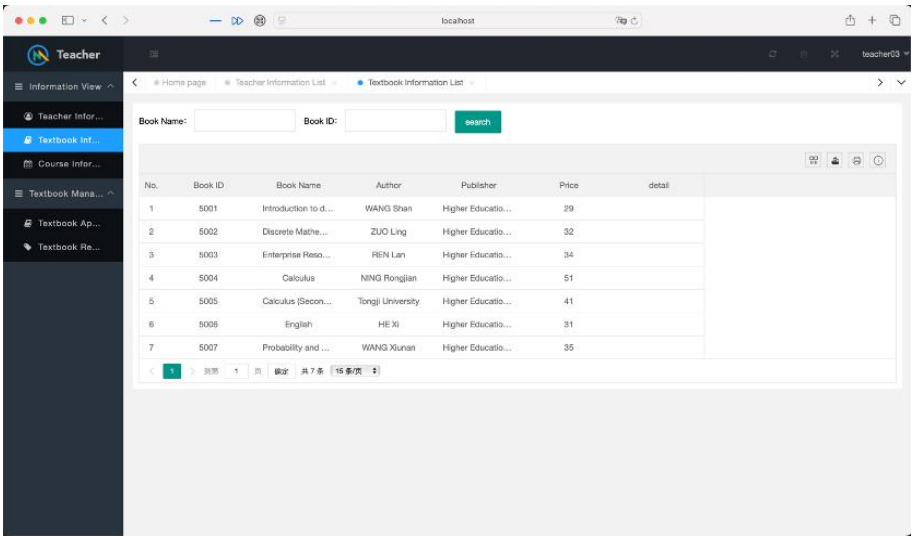


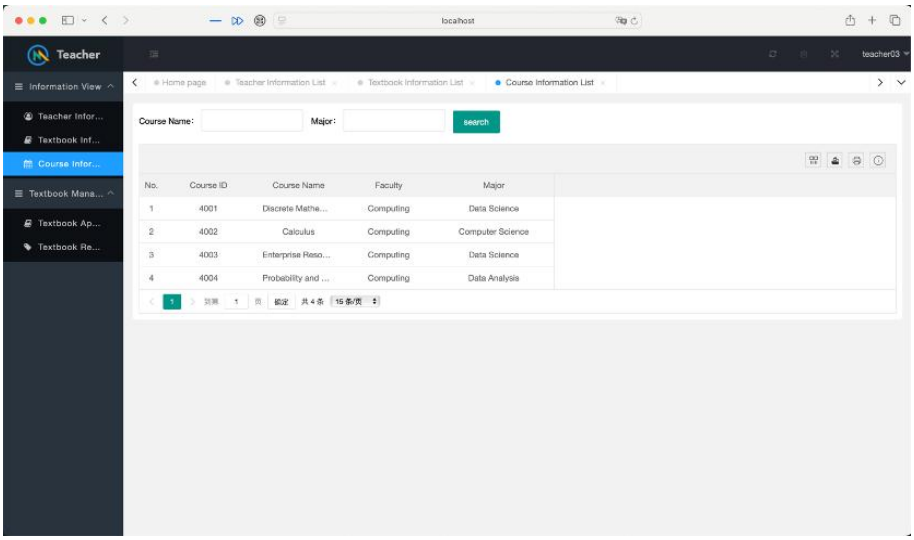


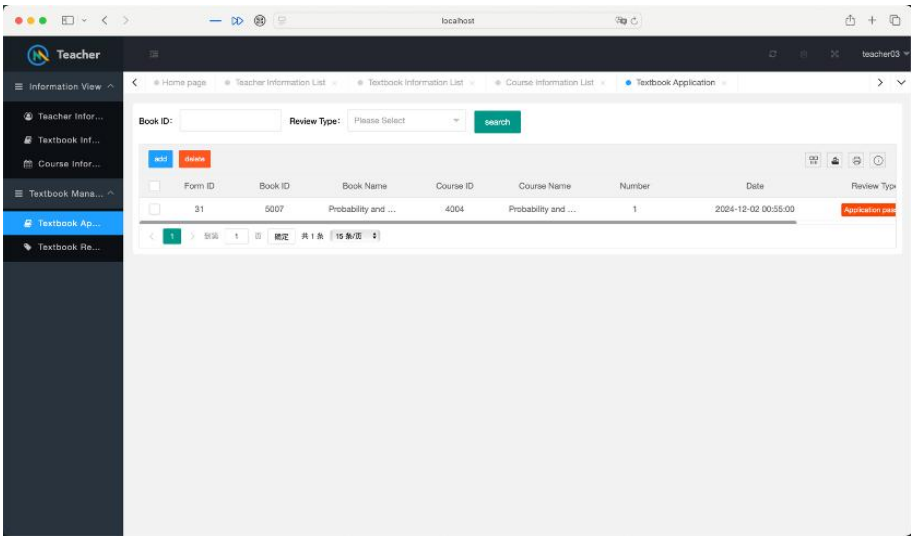
Teacher side:

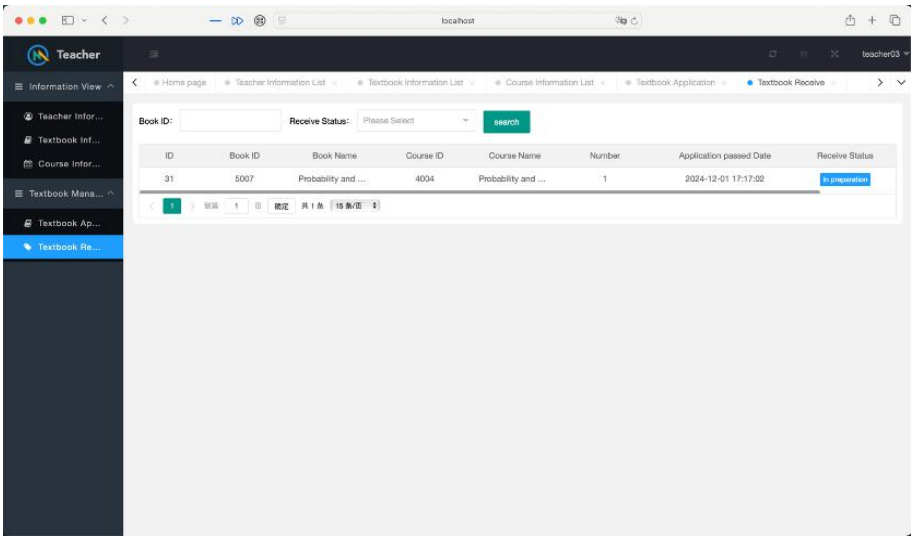


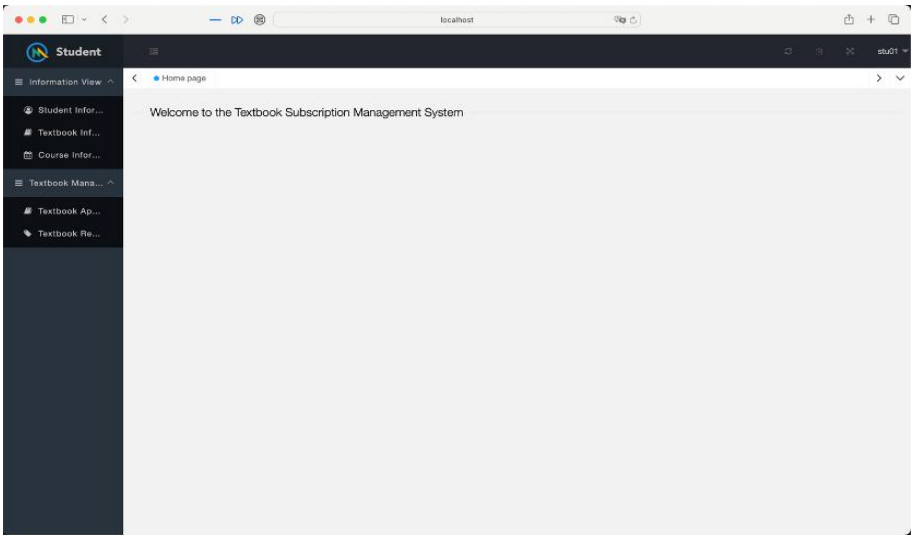

localhost
Teacher
teacher03
Information View
Teacher Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Teacher Information List
Personal information details
Teacher ID: 2003
Name: Lily
Sex: Female
Faculty: Computing
Contact Number: 12453231124


localhost
Teacher
teacher03
Information View
Teacher Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Teacher Information List
Textbook Information List
Book Name:
Book ID:
search
No.
Book ID
Book Name
Author
Publisher
Price
detail
1 5001 Introduction to d... WANG Shan Higher Educatio... 29
2 5002 Discrete Mathe... ZUO Ling Higher Educatio... 32
3 5003 Enterprise Reso... REN Lan Higher Educatio... 34
4 5004 Calculus NING Rongjian Higher Educatio... 51
5 5005 Calculus (Secon... Tongji University Higher Educatio... 41
6 5006 English HE Xi Higher Educatio... 31
7 5007 Probability and ... WANG Xiunan Higher Educatio... 35

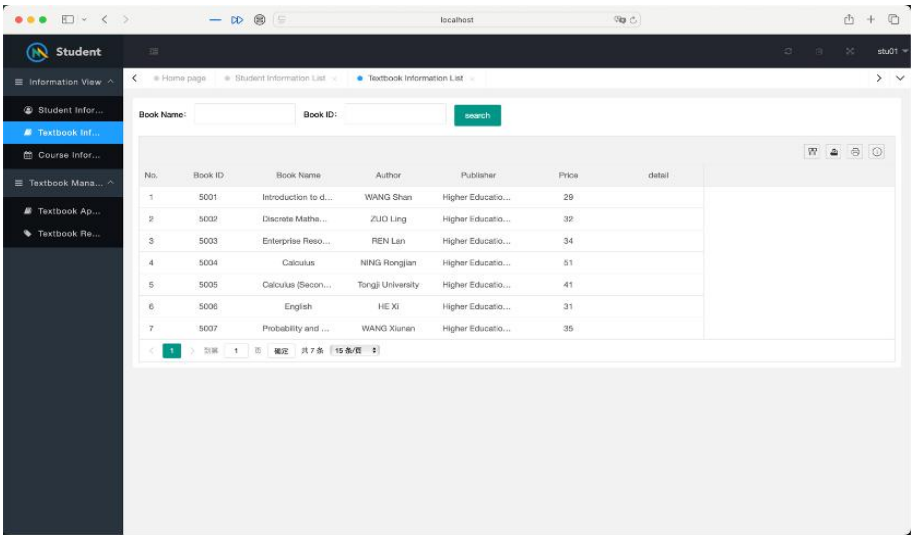

localhost
Teacher
teacher03
Information View
Teacher Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Teacher Information List
Textbook Information List
Course Information List
Course Name:
Major:
search
No.
Course ID
Course Name
Faculty
Major
1 4001 Discrete Mathe... Computing Data Science
2 4002 Calculus Computing Computer Science
3 4003 Enterprise Reso... Computing Data Science
4 4004 Probability and ... Computing Data Analysis

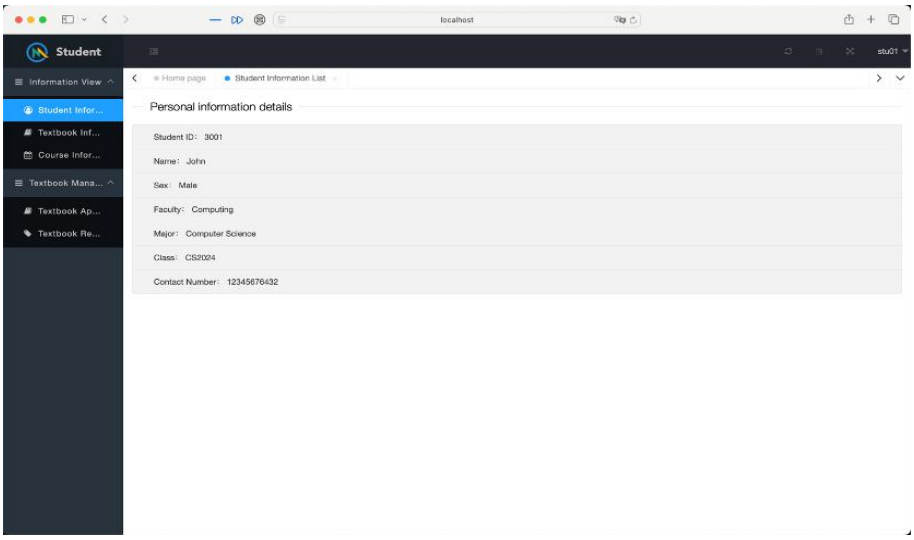


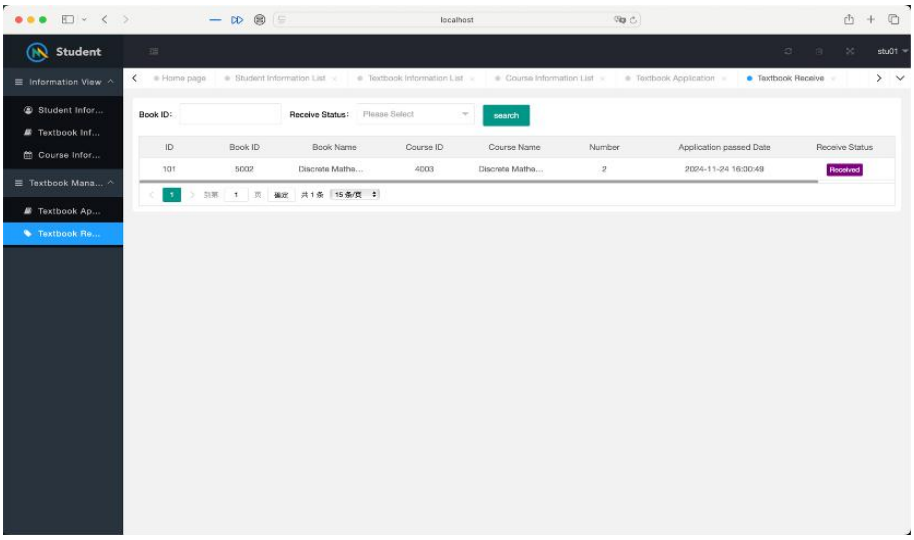


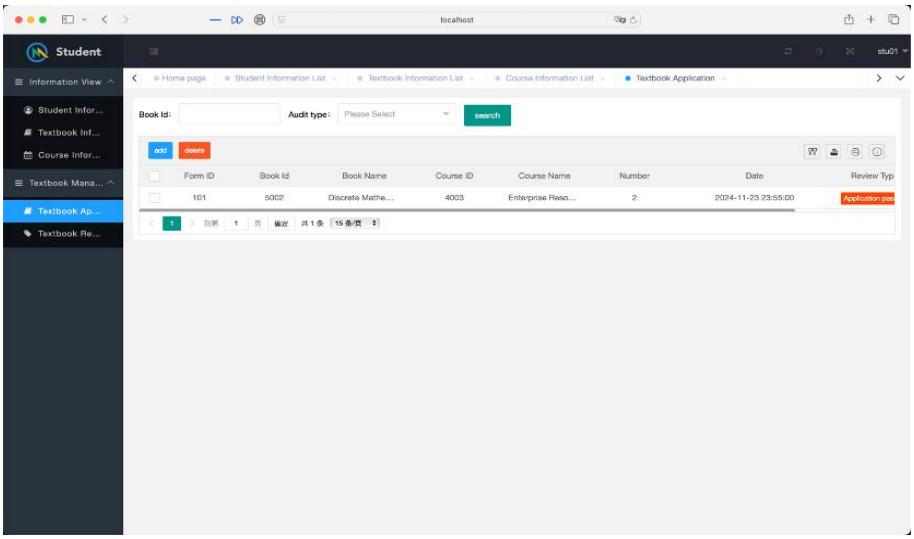


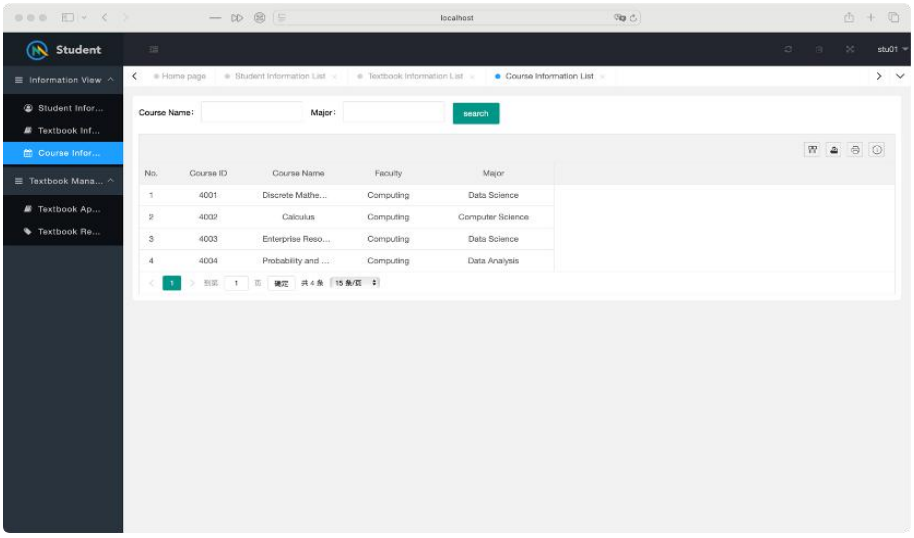
Student side：


localhost
Student
stu01
Information View
Student Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Student Information List
Textbook Information List
Book Name:
Book ID:
search
No.
Book ID
Book Name
Author
Publisher
Price
detail
1 5001 Introduction to d... WANG Shan Higher Educatio... 29
2 5002 Discrete Mathe... ZUO Ling Higher Educatio... 32
3 5003 Enterprise Reso... REN Lan Higher Educatio... 34
4 5004 Calculus NING Rongjian Higher Educatio... 51
5 5005 Calculus (Secon... Tongji University Higher Educatio... 41
6 5006 English HE Xi Higher Educatio... 31
7 5007 Probability and ... WANG Xiunan Higher Educatio... 35
共 7 条
15 条/页


localhost
Student
stu01
Information View
Student Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Student Information List
Personal information details
Student ID: 3001
Name: John
Sex: Male
Faculty: Computing
Major: Computer Science
Class: CS2024
Contact Number: 12345676432


localhost
Student
stu01
Information View
Student Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Student Information List
Textbook Information List
Course Information List
Textbook Application
Textbook Receive
Book ID:
Receive Status:
Please Select
search
ID
Book ID
Book Name
Course ID
Course Name
Number
Application passed Date
Receive Status
101 5002 Discrete Mathe... 4003 Discrete Mathe... 2 2024-11-24 16:00:49 Received
共 1 条
15 条/页


localhost
Student
stu01
Information View
Student Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Student Information List
Textbook Information List
Course Information List
Textbook Application
Book Id:
Audit type:
Please Select
search
add
delete
Form ID
Book Id
Book Name
Course ID
Course Name
Number
Date
Review Typ
101
5002
Discrete Mathe...
4003
Enterprise Reso...
2
2024-11-23 23:55:00
Application pas


localhost
Student
stu01
Information View
Student Infor...
Textbook Inf...
Course Infor...
Textbook Mana...
Textbook Ap...
Textbook Re...
Home page
Student Information List
Textbook Information List
Course Information List
Course Name:
Major:
search
No.
Course ID
Course Name
Faculty
Major
1
4001
Discrete Mathe...
Computing
Data Science
2
4002
Calculus
Computing
Computer Science
3
4003
Enterprise Reso...
Computing
Data Science
4
4004
Probability and ...
Computing
Data Analysis

Appendix

Source Code:

https://drive.google.com/file/d/1TQ4QgnxncUgZCWNyj1z4_w45ZO-xltiS/view?usp=drive_link

Deployment Documents:

https://drive.google.com/file/d/1f2-tM4VtZ1q8_b95ATltVXxb0k6Bjxz5/view?usp=drive_link

Video Demonstration:

https://drive.google.com/file/d/1FDyfCY-sFFt_tV8hr2dhFdkfEwnk3Fb2/view?usp=drive_link

Major

BAO Huiwen: Information Management and Information Systems

XU Jingzhe: Data Science

YANG Hongkun: Computer Science and Technology

YANG Mengyuan: Information Management and Information Systems

ZHAO Hongyang: Computer Science and Technology

Contribution

BAO Huiwen: Front-end Development and Interface Debugging &Demo Video

XU Jingzhe: Database Design and Implementation &Deployment Documents

YANG Hongkun: System Analysis and Requirement Analysis &Text Integration

YANG Mengyuan: General Design, System Framework and Background Development &Code Integration

ZHAO Hongyang: System Testing and Documentation &E-R Diagrams