Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 12, No. 3, 2024

Design of the background of an e-commerce mall

Yunzheng Ding *

School of Information Engineering, Jingdezhen Ceramic university, Jingdezhen 333403, P.R. China
* **Corresponding author Email:** jciddyz@163.com

Abstract: Java technology is a platform independent programming language that has the characteristics of being written once and running anywhere, making it very suitable for distributed network programming. The backend management website of the online shopping system mainly implements the functions of the online shopping system through the application of Hibernate, Struts, Tomcat, Jsp, and MySQL databases. The backend of the entire online shopping system mainly consists of three modules, namely the product order management module, the website user management module, and the product classification management module. The product order management module includes four parts: adding new products, viewing and modifying products, managing product orders, and confirming orders; The main function of the management website user module is to provide an operation interface for user backend maintenance and management. Management personnel can perform user queries, deletions, and other operations; The main function of the product classification management module is to complete the processing operation of product categories.

Keywords: Hibernate; Struts; Jsp; MySQL.

1. Requirement analysis

The backend maintenance and management system for online shopping malls should generally provide functions such as product information management, product category management, customer information management, and order processing.

According to the basic requirements of online mall backend maintenance, the specific tasks that this system needs to complete are as follows.

Product classification management: Through this module, website administrators can add new product categories as needed, and can also modify, delete, and other operations on existing categories.

Basic Product Information Management: In order to ensure the effectiveness of product information in online shopping malls, management personnel can use this module to add new product information at any time, and can also modify and delete existing product information.

Order processing: Backend management personnel can use this module to query order information, so that website distribution personnel can process subsequent shipments and deliveries based on the order information. At the same time, for orders that have already been processed, historical records should also be kept for management personnel to query.

Member information management: Management personnel can query corresponding user information in this module and delete relevant information of specified users, which will be very helpful in ensuring the validity of user information.

2. Overall design

In the initial stage of software system development, it is generally necessary to select the appropriate development tools and software architecture based on the actual functional requirements of the system. The reliability and stability of online shopping systems have relatively high requirements. When designing this system, popular B/S designs include patterns based on JSP, ASP, PHP, CGI, and J2EE. Compared to others, PHP has relatively simple functions and is not suitable for making large programs; However, the efficiency of CGI is relatively low, so it is not considered either. Because the system does not have the original basic platform to expand and does not require too much interaction with other systems, the use of J2EE patterns cannot reflect the advantages of J2EE itself. JSP is one of the core technologies of J2EE, which can be upgraded to J2EE programs at any time.

The project ultimately believes that using JSP is a more suitable choice at this stage, and choosing the Struts architecture as the main framework for development and Hibernate as the data persistence processing layer takes into account its high-speed development efficiency, high code reusability, easy maintenance, and other advantages. The ultimate goal is to improve the reusability of the underlying business logic of the system, increase system scalability, and reduce system maintenance costs.

2.1. Operating Environment

In order to increase the throughput of the system and increase the number of concurrent customer requests, the system ultimately adopted a P4 server as the host. Considering that from a database perspective, there is no need to use stored procedures and server-side functions at the data layer to close too many business logic databases, therefore the database system adopts relatively sophisticated MySQL. If the shopping system server needs to be deployed on other hosts, the necessary conditions for that host are as follows.

Server side operating system: A cross platform system independent of the operating system, with client MicroSoft Windows 2000 and above.

Database: MySQL version 4.1.

Web server: Tomcat version 5.0.19 and above, compatible with Struts development architecture and Hibernate development architecture.

Client operating environment: capable of running operating systems with browsers above IE4 and Netscape 4.5.

Client running tool: The current system uses a browser as the client and can use any version of the browser above IE4.

This website system uses IE as the web platform, JSP+Tomcat+Struts+Hibernate as the website

implementation technology, establishes a core dynamic webpage based on MySQL database system, and implements functional modules such as front-end shopping and back-end maintenance management for e-commerce websites. The presentation layer of the system adopts page implementation techniques that fully comply with HTML4.0, DHTML, and Struts tag libraries, as well as optimized page code and image techniques.

2.2. Functional module division

The key to system software development lies in system design, and the results of user interface design directly determine the user's evaluation of the system. Therefore, a good user interface design is a necessary condition for the success of the system, especially for the design of pages in business systems. Fig.1 shows the hierarchical structure of the backend page determined by the online shopping system based on the required columns.

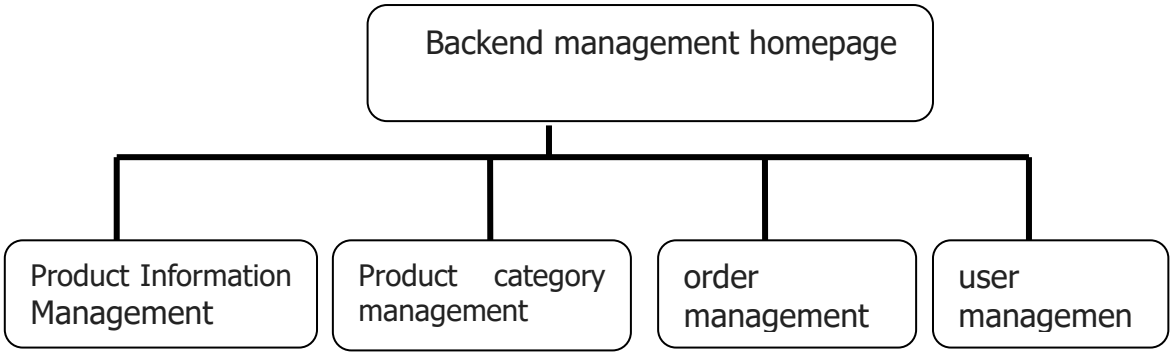

Fig.1 Hierarchical Structure of Backend Pages

Table 1. Website Column Settings in the Backend of Online Shopping System

column	explain
Backend product information management	Provide the addition, deletion, and modification of product information. And the functions of adding, deleting, and modifying product categories
Backend order management	Receive the latest customer orders and effectively allocate and process them
Backend user management	Provide maintenance and management functions for website administrator user information

Table 2. System Component

User presentation layer (view)	control processing layer (controller)	data persistence layer	business logic layer (model)
login.jsp index.jsp left.jsp main.jsp addProduct.jsp modifyOrder.jsp modifyProdut.jsp modifySort.jsp modifyUser.jsp processOrder.jsp	AddProductAction.java AddProductPreAction.java AddSortAction.java DelMemberAction.java DelProductAction.java DelSortAction.java LoginAction.java ModifyProductAction.java MyRequestProcessor.java MyResource_zh.properties ProcessOrderAction.java ViewMemberAction.java ViewOrderAction.java ViewProcessOrderAction.java ViewProductAction.java ViewSortAction.java struts-config.xml web.xml	hibernate.cfg.xml model.hbm.xml HibernateUtil.java DbOperate.java	AdminUser.java Constants.java Member.java Order.java Product.java Sort.java ViewList.java

This hierarchical diagram shows the link relationship between the backend homepage and each level of page. Table 1 shows the settings for each column of the business system website determined from customer requirements.

Table 2 shows the various components that make up the online mall backend maintenance and management system. Through this table, readers can gain a macro understanding of the functions of each component.

Based on the relevant settings of each column listed in the table above, the following functional modules of the online

shopping system can be determined.

2.2.1. Product classification management

This module implements classification management of basic product information. Management personnel can add new laptop brands at any time, and can also modify or delete existing product categories as needed.

2.2.2. Basic Product Information Management

This module implements the management of basic product information. Management personnel can add new product information at any time, and can also modify or delete existing product information as needed. In order to improve work efficiency, a product search function should be provided in this module. After searching for corresponding product information, management personnel can freely delete and modify product descriptions and images. The product information management module is a backend maintenance and management module provided to ensure the timeliness of product information in online shopping malls. With the help of this module, the backend maintenance and management personnel of the shopping website can add new products to the products sold online at any time, update product information including prices, and add and modify product categories. This functional module can provide the following sub functions:

Product classification management;

Product information management

2.2.3. Order processing

Management personnel can view customer orders through this module and effectively process orders placed by customers, such as assigning orders to salespeople. At the same time, for processed orders, management personnel can also query order history through the search interface. The order management module is used to receive the latest customer orders and effectively allocate and process them. Its main workflow management includes functions such as order management and order processing progress management. This functional module can provide the following sub functions:

Latest order information management, viewing customer orders;

Order processing, effectively handling orders placed by customers;

Order tracking and allocation, where the administrator assigns orders placed by customers to sales representatives.

2.2.4. Member Information Management

To ensure the validity of user information, management personnel can query corresponding user information in this module and delete relevant information of specified users. The user information management module is used to add, delete, and modify users for website backend maintenance and management. In order to ensure the security of information maintenance, when entering the corresponding backend maintenance management module, it is necessary to enter according to the administrator user identity, thereby strengthening the management of user permissions.

3. Database design

3.1. Analysis of Database

According to the processing requirements of the online mall backend maintenance center, the design and functions of the data table are as follows.

Product Basic Information Table: stores the basic information of products sold in online shopping malls.

Basic information table for product classification: stores the classification information of products provided by online shopping malls.

Membership form: stores basic information of online mall customers.

Order Information Table: stores basic order information related to customers.

System administrator basic information table: stores the username and password information of the backend administrator.

3.2. Detailed Design of Database

Product Basic Information Table

The data table used to store the basic information of each product in the online shopping mall is the basis for displaying, maintaining, and managing product data. Table 3 shows the fields and descriptive information contained in this table.

Table 3. Basic Product Information Table

field name	description	type	length	allowed to be empty	primary key
id	Product number	INTEGER		N	Y
sortid	Product classification number	INTEGER		N	N
name	Product Name	VARCHAR	50	N	N
price	commodity price	DOUBLE		N	N
saleprice	Sales price	DOUBLE	4	N	N
descript	Product Description	TEXT	500	N	N
contents	Detailed product introduction	TEXT	2000	Y	N
saledate	Shipment date	DATE		N	N
salecount	Sales quantity	INTEGER		Y	N
image	Storage path for product cover images	VARCHAR	50	Y	N

In order to display promotional images of the corresponding products on the page, an "image" field was designed in the data table to save the path of the corresponding product images, so that the corresponding images can be obtained and displayed according to this path in the future.

Basic information table for product classification (sort)

Used to record relevant information on various product classifications in online shopping malls, in order to facilitate the classification and display of product information. The

field settings are shown in Table 4.

In order to facilitate users in finding the required product information, the displayed products can be classified and managed by brand.

Member form

Used to store information about registered users in online shopping malls, including their names, contact information, and other related information. The field settings are shown in Table 5.

Order table (orders)

Used to store specific order information, its field settings are shown in Table 6.

Basic information table for system administrators (adminuser)

Used to store the basic information of the online mall backend maintenance administrator, including the username and password of the administrator user. The field settings are shown in Table 7.

Table 4. Product Category Table

field name	description	type	length	allowed to be empty	primary key
id	Product classification number	INTEGER		N	Y
name	Product classification name	VARCHAR	40	N	N

Table 5. Membership Table

field name	description	type	length	allowed to be empty	primary key
id	User ID	INTEGER		N	Y
username	user name	VARCHAR	20	N	N
password	password	VARCHAR	20	N	N
realname	User Name	VARCHAR	20	Y	N
tel	User phone number	VARCHAR	20	Y	N
address	User address	VARCHAR	100	Y	N
zip	zip code	VARCHAR	6	Y	N
email	User email address	VARCHAR	50	Y	N

Table 6. Order Table

field name	description	type	length	allowed to be empty	primary key
id	Order number	INTEGER		N	Y
orderno	Generate order number	VARCHAR	50	N	N
userid	User ID	INTEGER		N	N
realname	Consignee Name	INTEGER	20	N	N
Address	Receiving address	VARCHAR	100	N	N
zip	zip code	VARCHAR	6	Y	N
tel	Contact phone number	VARCHAR	20	Y	N
payment	Payment method	VARCHAR	20	Y	N
email	Email address	VARCHAR	50	Y	N
memo	Remarks Description	TEXT	2000	Y	N
price	price	DOUBLE		Y	N
time	Order generation time	VARCHAR	20	Y	N
tag	Flag whether the order has been processed	INTEGER		Y	N

Table 7. Basic Information of System Administrators

field name	description	type	length	allowed to be empty	primary key
id	User ID	INTEGER		N	Y
username	user name	VARCHAR	20	N	N
password	password	VARCHAR	20	N	N

3.3. Script for Creating Data Tables

After determining the basic structure of the data table, the work of the data table can be completed in MySQL. Below is the SQL script for creating the corresponding data table.

Basic Product Information Table

```
create table product
(
  id          integer        primary   key ,
  sortid      integer        not  null
              references  sort (id)  on delete  cascade ,
  name        varchar (50)     not  null ,
  price       double           not  null ,
  saleprice   double           not  null ,
  descript    text (500)       not  null ,
  contents    text(2000)       null ,
  saledate    double           not  null ,
  salecount   integer          null ,
image         varchar(50)     null  )
```

Basic information table for product classification

```
create  table  sort  (
  id      integer       primary  key ,
  name  varchar(40)   not  null  )
```

Membership form

```
create  table  member  (
  id            integer          primary  key ,
  username      varchar(20)      not  null ,
  password      varchar(20)      not  null ,
```

```
    realnaem      varchar(20)      null ,
    tel           varchar(20)      null ,
    address       varchar(100)     null ,
    zip           varchar(6)       null ,
    email         varchar(50)      null   )
  Order Information Table
  create   table   orders   (
    id            integer          primary  key ,
    orderno       varchar(50)      not  null ,
    userid        integer          not  null
    references    member (id)  on  delete  cascade ,
    realname      varchar(20)      not  null ,
    address       varchar(100)     not  null ,
    zip           varchar(6)       null ,
    tel           varchar(20)      null ,
    payment       varchar(20)      null ,
    email         varchar(50)      null ,
    memo          text(2000)       null ,
    price         double           null ,
    time          varchar(20)      null ,
  tag           integer          null   )
  Basic information table for system administrators
  create   table   adminuser (
     id           integer          primary   key ,
     username     varchar(20)  not  null ,
     passward     varchar(20)  not  null   )
```

4. Detailed design

4.1. System Page Design

In the user presentation layer, there are mainly some related JSP pages. The JSP pages in this layer should be placed in the root directory of the corresponding project. The following will introduce each webpage separately.

4.1.1. Online Mall Backend Maintenance Management System Login Page

The main function of this page (login. jsp) is to provide relevant redirects for subsequent pages. This includes links related to product information, category information, orders, and user management. This page is the authentication page for administrator users to enter the backend management system. On this page, when the administrator user enters user information such as username and password and clicks the "Login" button, a "login. do" request will be submitted for authentication processing.

4.1.2. Online Mall Backend Maintenance and Management Homepage

This page (index. jsp) is a framework page, which is divided into two parts. The specific corresponding page files and functions are as follows.

Left.jsp page: Provides hyperlinks for administrators to access different management interfaces.

Main. JSP page: A welcome page where relevant system usage instructions and other information can be added during actual use.

4.1.3. Adding and Modifying Product Information Pages

On this page (addProduct. jsp), backend maintenance personnel can enter the latest product related information. This page is redirected to when the controller responds to the request to add new product information submitted on the main menu page, that is, after issuing a "addProduct. do" request, and when editing and modifying product information, after submitting a "modifyProduct. so" request.

In this page, the form tag for file browsing in the Struts tag library was used to upload files, and a new attribute enctype="multipart/form data" was added to the form tag of the form itself to ensure successful upload of file information.

4.1.4. Modifying the Product Information Page

On this page (modifyProduct. jsp), you can search for corresponding products and perform maintenance operations such as modifying or deleting product information. This page is accessed by the administrator after selecting the "View and Modify Products" command, which means submitting a "viewProduct. so" request. On this page, the default pagination displays information for all products. After entering the keywords of the products to be queried, the administrator clicks the "Submit" button to submit a "viewProduct. do" request for product information query processing. Clicking on the "Delete" command for the corresponding product will submit a request for "delProduct. do? Id=<%=product. gegId()%>" to delete the corresponding product record. Clicking the "Modify" command will submit a "modifyProduct. do? Id=<%=product. gegId()%>" request for subsequent editing and modification of the corresponding product information.

4.1.5. Processing User Order Pages

The main function of this page (modifyOrder. jsp) is to provide an operation interface for order backend maintenance and management. Management personnel can query and process orders. This page is redirected to after the administrator clicks the "Manage Product Orders" command to submit a "viewOrder. do" request.

On this page, by default, all order information will be displayed in pages. After entering the keywords for the order number to be queried, the administrator will click the "Submit" button to submit a "viewOrder. do" request for order information query processing. Clicking the "Process" command for the corresponding order will submit a "processOrder. so? Id=<%=order. getId()%>" request for corresponding order records processing.

4.1.6. Query page displaying processed orders

This page (processOrder. jsp) is the page that the administrator clicks on the "Confirmed Order" command to submit a "viewProcessOrder. do" request, and redirects to. This page can achieve the effect of querying and displaying processed product orders.

4.1.7. Manage website user pages

The main function of this page (modifyUser. jsp) is to provide an operation interface for user backend maintenance and management. Management personnel can perform user queries, deletions, and other operations. This page is redirected to after the administrator clicks the "Manage Website Users" command to submit a "viewMember. do" request. On this page, you can achieve the function of querying and deleting website user information.

In this page, by default, all website user information will be displayed in separate pages. If the administrator enters the username keyword to be queried and clicks the "Submit" button, a "viewMember. do" request will be submitted to query user information. Clicking on the "Delete" command for the corresponding order will submit a "delMember. do? Id=<%=member. getId()%>" request for the deletion of user records.

4.1.8. Product Category Management Page

This page (cash. jsp) is the page that the administrator redirects to after clicking the "Product Category

Management" command to submit a "viewSort. do" request.

On this page, all product category information is displayed in default pagination. If the administrator enters the name of the new product category and clicks the "Submit" button, a "addSort. do" request will be submitted to process the addition of the new category. Clicking on the "Delete" command for the corresponding category will submit a "delSort. do? Id=<%=sort. getId()%>" request to delete the product category record.

4.2. Data persistence layer

This system adopts the Hibernate development architecture to implement the function of the data persistence layer. Therefore, the specific source code and functional explanation of Hibernate related configuration files and mapping files in the online mall backend maintenance management system will be provided.

The configuration files and mapping files corresponding to Hibernate are generally placed in the src directory of the project. The classes used to manage sessions that encapsulate methods related to database operations, like other classes, are placed in corresponding packages.

4.2.1. Hibernate configuration file (hibernate. cfg. XML)

In this file, the configuration should be based on the actual situation of the database system and corresponding JDBC driver used. Prior to this, the JDBC driver should have been deployed under the common/bb of the application server Tomcat.

```
<hibernate-configuration>
    <session-factory>
        <property
name="hibernate.connection.driver_class">org.gjt.mm.mysq
l.Driver</property>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost/test
</property>
        <property
name="hibernate.connection.username">root</property>
        <property
name="hibernate.connection.password">123</property>
        <property
name="hibernate.connection.pool_size">100</property>
        <property name="show_sql">false</property>
        <property
name="dialect">org.hibernate.dialect.MySQLDialect</prop
erty>
        <!-- Mapping files -->
        <mapping resource="model.hbm.xml"/>
    </session-factory>
</hibernate-configuration>
```

4.2.2. Hibernate mapping file (model. hbm. XML)

This mapping file is the most basic and important configuration file in the Hibernate architecture, used to declare the mapping relationship between the properties of entity classes implemented in Hibernate and the corresponding data table fields in the database. The corresponding file name is "model. hbm. XML". Note that the attribute names declared in this file must correspond to the field names in the data table, and must also be the same as the attribute names of the encapsulated information declared in the corresponding entity classes in the business logic layer. The code for establishing a mapping relationship with the administrator's basic information is as follows:

```
<class name="cn.com.shopadmin.AdminUser"
        table="adminuser">
    <id name="id">
    <generator class="increment"/>
    </id>
    <property name="username"/>
    <property name="password"/>
</class>
```

4.2.3. Encapsulated classes for database operations

This class encapsulates methods for database related operations using Hibernate. In the methods related to database operations declared in this class, traditional SQL statements are not used to complete database operations. Instead, Hibernate's HQL statements are used to add, delete, modify, and query data records. Some of the codes are as follows:

```
public class DbOperate {
    /**
     * Obtain administrator object based on username     */
    public AdminUser getAdminUser(String userId) throws
HibernateException {
        Session session = HibernateUtil.currentSession();
        AdminUser adminuser = null;
        Transaction tx = null;
        try {
            tx = session.beginTransaction();
//          Create query object
            Query query = session.createQuery("from
AdminUser where username=:userId");
            query.setParameter("userId", userId);
            List list = query.list();
            if (!list.isEmpty())
                adminuser = (AdminUser) list.get(0);
            tx.commit();
        } catch (HibernateException e) {
            if (tx != null)
                tx.rollback();
            throw e;
        }
        session.close();
        return adminuser;
    }
```

4.3. Business logic layer

In the business logic layer of this system, various types of objects such as product information, product classification information, shopping cart information, order information, and customer information have been saved and processed. Note that the class files corresponding to the business logic layer should be placed in the src directory of the project directory.

4.3.1. Bean (Product. java) that encapsulates basic product information

In the beans related to product information encapsulation processing, basic attributes related to product basic data are declared, and their names should be consistent with the attribute names and data table field names in the corresponding Hibernate mapping file. The specific code is as follows:

```
public class Product {
    private int id;
    private Sort sort;
    private String name;
    private double price;
    private double saleprice;
    private String descript;
```

```
private String contents;
private String saledate;
private int salecount;
private String image;
public String getContents() {
	return contents;
}
public void setContents(String contents) {
	this.contents = contents;
}
public String getDescript() {
	return descript;
}
public void setDescript(String descript) {
	this.descript = descript;
}
public int getId() {
	return id;
}
public void setId(int id) {
	this.id = id;
}
public String getImage() {
	return image;
}
public void setImage(String image) {
	this.image = image;
}
public double getPrice() {
	return price;
}
public void setPrice(double price) {
	this.price = price;
}
public int getSalecount() {
	return salecount;
}
public void setSalecount(int salecount) {
	this.salecount = salecount;
}
public String getSaledate() {
	return saledate;
}
public void setSaledate(String saledate) {
	this.saledate = saledate;
}
public double getSaleprice() {
	return saleprice;
}
public void setSaleprice(double saleprice) {
	this.saleprice = saleprice;
}
public String getName() {
	return name;
}
public void setName(String name) {
	this.name = name;
}
public Sort getSort() {
	return sort;
}
public void setSort(Sort sort) {
	this.sort = sort;
}
```

```
}
```

4.3.2. Encapsulated beans for processing product category information (Sort. java)

In the beans related to the encapsulation and processing of product classification information, basic properties and methods related to product classification data are declared.

```
package cn.com.shopadmin;
public class Sort {
   private int id;
   private String name;
   public int getId() {
        return id;
   }
   public void setId(int id) {
        this.id = id;
   }
   public String getName() {
        return name;
   }
   public void setName(String name) {
        this.name = name;
   }
}
```

4.3.3. Bean that encapsulates administrator basic information (AdminUser. java)

In the beans related to encapsulating and processing administrator basic information, basic properties and methods related to administrator basic information are declared. The specific code is as follows:

```
package cn.com.shopadmin;
public class AdminUser {
   private int id;
   private String username;
   private String password;
   public int getId() {
        return id;
   }
   public void setId(int id) {
        this.id = id;
   }
   public String getPassword() {
        return password;
   }
   public void setPassword(String password) {
        this.password = password;
   }
   public String getUsername() {
        return username;
   }
   public void setUsername(String username) {
        this.username = username;
   }
}
```

4.3.4. Encapsulates a bean (ViewList. java) that displays specific business logic in a paginated manner

Considering that data information pagination needs to be used multiple times in backend maintenance pages, in order to achieve code reuse, the specific business logic of pagination display control was separately encapsulated in a bean during design and implementation, and then specifically called in the control layer.

```
public class ViewList {
    public void display(List list,HttpSession session,Integer
pageId,String keyword){
        int iPageId = 0;
        if (pageId!=null) iPageId = pageId.intValue();
        if (iPageId<0) iPageId = 0;
       /*
        * Get total number of pages          */
        int pageCount = 0;
        if (list.size()%Constants.PRODUCT_PAGE_SIZE
==0){
            pageCount=list.size()                    /
Constants.PRODUCT_PAGE_SIZE;
        }
        else{
            pageCount=list.size()                    /
Constants.PRODUCT_PAGE_SIZE+1;
        }
        /*
         * Pagination display
         */
    If                ((list.size()>iPageId                *
Constants.PRODUCT_PAGE_SIZE )&&(iPageId>=0)){
            List   dispList=new ArrayList();
            for                                      (int
i=iPageId*Constants.PRODUCT_PAGE_SIZE;i<(iPageId+
1)*Constants.PRODUCT_PAGE_SIZE;i++){
                if (i<list.size()){
                    dispList.add(list.get(i));
                }
            }
            /*
             * Store paginated display information into
the session                  */

    session.setAttribute(Constants.SEARCH_LIST_KEY,d
ispList);

    session.setAttribute(Constants.CUR_PAGEID_KEY,ne
w Integer(iPageId));

    session.setAttribute(Constants.PAGE_COUNT_KEY,n
ew Integer(pageCount));

    session.setAttribute(Constants.CUR_KEYWORD_KE
Y,keyword);
        }
    }
}
```

4.3.5. Constant file (Constants. Java)

In Struts applications, it is advocated to define some attribute key constants in constant files, which will help improve the independence of Actions. For example, if the constant value of the attribute key changes in the program, only the constant file needs to be modified without modifying the corresponding Action. The specific code is as follows:

```
package cn.com.shopadmin;
public final class Constants {
   // Session keys
    public static final String LOGIN_USER_KEY =
"LOGIN_USER";
    public static final String SORT_LIST_KEY =
"SORT_LIST";
    public static final String SEARCH_LIST_KEY =
"SEARCH_LIST";
     public static final String CUR_PRODUCT_KEY =
"CUR_PRODUCT";
    public static final String CUR_SORTID_KEY =
"CUR_SORTID";
    public static final String CUR_KEYWORD_KEY =
"CUR_KEYWORD";
    public static final String CUR_PAGEID_KEY =
"CUR_PAGEID";
    public static final String PAGE_COUNT_KEY =
"PAGE_COUNT";
    public static final int PRODUCT_PAGE_SIZE =3;
}
```

4.4. Control processing layer

Table 8. Action Mapping Table

Action	entrance	ActionForm	Export
LoginAction	login.jsp	LoginForm	index.jsp
AddProductPreAction	left.jsp	bookIdForm	addProduct.jsp
AddProductAction	addProduct.jsp	ProductForm	wrong.jsp
ViewProductAction	left.jsp	viewForm	modifyProduct.jsp
ModifyProductActon	modifyProdut.jsp	IdForm	addRroduct.jsp
ViewOrderAction	left.jsp	viewForm	modifyOrder.jsp
ViewProcessOrderAction	left.jsp	viewForm	processOrder.jsp
ViewMembeAction	left.jsp	viewForm	modifyUser.jsp
ViewSortAction	left.jsp	viewForm	modifySort.jsp
ProcessOrderAction	modifyOrder.jsp	IdForm	wrong.jsp
DelProductAction	modifyProduct.jsp	IdForm	wrong.jsp
DelMemberAction	modifyUser.jsp	IdForm	wrong.jsp
AddSortAction	modifySort.jsp	SortForm	wrong.jsp
DelSortAction	modifySort.jsp	IdForm	wrong.jsp

Table 8 shows the Action mapping table in the online mall backend maintenance management system. This mapping determines the association between an Action and other web components. As a bridge between the front-end and back-end,

this table specifies the entry point (i.e. the component that calls the Action) and exit point (i.e. the target component for request forwarding) for each Action, as well as the ActionForm that passes the Action. Note that the class files corresponding to beans in the control processing layer should also be stored in the src directory under the project directory.

4.4.1. Web.xml configuration file

This configuration file is a universal configuration file for web applications. Mainly responsible for configuring the AciotnServlet and setting the default homepage. Some of the codes are as follows:

```
<!-- The Welcome File List  -->
  <welcome-file-list>
    <welcome-file>login.jsp</welcome-file>
  </welcome-file-list>
  <!-- Struts Tag Library Descriptors -->
  <taglib>
   <taglib-uri>/WEB-INF/struts-bean.tld</taglib-uri>
   <taglib-location>/WEB-INF/struts-bean.tld</taglib-
location>
</taglib>
  <taglib>
   <taglib-uri>/WEB-INF/struts-html.tld</taglib-uri>
   <taglib-location>/WEB-INF/struts-html.tld</taglib-
location>
 </taglib>
```

4.4.2. Struts config.xml configuration file

This file is the core of the Struts architecture, where developers use it to assemble various components and establish the overall framework of the application. It can be said that the function of this file is equivalent to a control and deployment center during the development and operation process of a large project. Some of the codes are as follows:

```
<global-forwards>
        <forward name="toIndex" path="/index.jsp"/>
        <forward                    name="toAddProduct"
path="/addProduct.jsp"/>
        <forward name="toWrong" path="/wrong.jsp"/>
        <forward                    name="toListProduct"
path="/modifyProduct.jsp"/>
        <forward                      name="toListOrder"
path="/modifyOrder.jsp"/>
        <forward              name="toListProcessOrder"
path="/processOrder.jsp"/>
        <forward                   name="toListMember"
path="/modifyUser.jsp"/>
        <forward                      name="toListSort"
path="/modifySort.jsp"/>
    </global-forwards
```

References

[1] Yang Hongbo ,Wang Zhishun, “J2SE evolution history”, Programmers, 2005(07), P50-52.

[2] LIU Xiao-zheng, “Analysis of Java GUI programming tool set”, SCIENCE & TECHNOLOGY INFORMATION, 2012(35), P596-597.

[3] Wang chun-li, “Network programming using Java”, CHINA SCIENCE AND TECHNOLOGY INFORMATION, 2006(04), P186-187.

[4] Li rui-ge,” Computer software development of Java programming language and applications”, Programmers, 2022(06), P15-16.