Project Summary

Advanced WWW Software Development

CS 6522

Honglan Liu

NetId: sp6682

11/25/2017

1. Introduction

The purpose of this Web Application is to create an online store which sells lipsticks. User can view products, search products, add products to cart and update cart as guest, but must register and login to checkout and view history order by their fake card number.

2. Functionality

1. Cart: a guest can view his/her cart, add products into cart, update product's number in cart or remove product in cart;

2. Checkout: after a guest add product into his/her cart, he/she can checkout using a fake card; while the user checks out, his/her cart is cleared;

3. Login/Logout: before a guest checks out, the store requires he/she logins first; after the guest logins, he/she can choose to logout.

4. Register: if the guest doesn't have an account, the store requires he/she register first;

5. Search product: a guest can search products in store's main page, and the store will return searching result;

6. View history order: the user (must login) can view his/her history order by entering his/her card number;

7. Track session: use session attributes to remember cart and user information, so when user comes back this store, he/she can skip login and the cart keeps the same;

8. Error page: if a guest enters wrong URL or some exceptions throw, the store will show error page to guest;

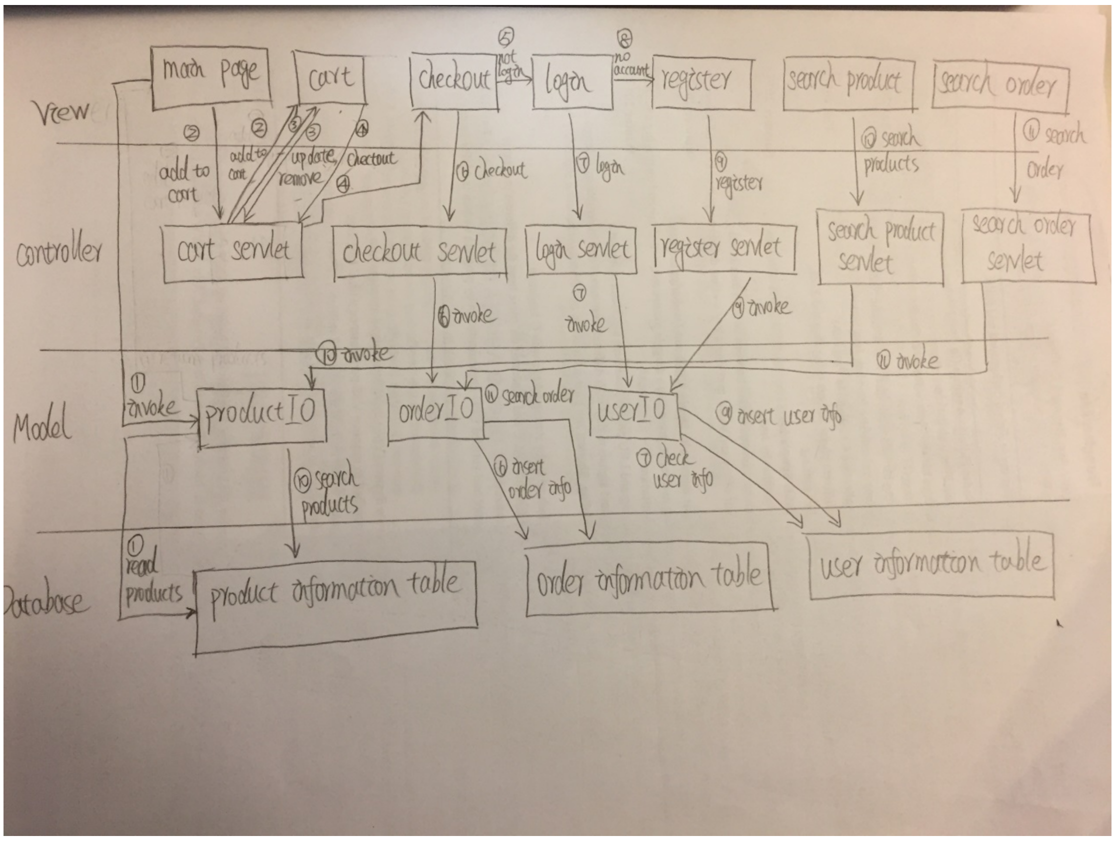
9. Header.html and footer.html: every jsp page has a header and footer page;

10. Customized information: initialize app's information in web.xml;

11. Database: use SQLite to store product, user and order information. Servlet use connection pool and prepared statement to access database.

3. Design and Implementation

3.1 Architecture of the whole app



In the view layer, main page shows all products by reading product information from product information table in database. Cart.jsp will show cart, and provide updating product function, removing product function and checkout function. Checkout.jsp shows the information user needs to provide to checkout, including user name, user email and user fake card. But before the user checks out, he/she needs to login or register. Login.jsp and register.jsp will provide those functions. SearchOrder.jsp is a page that user can enter their card number and search his/her history order. Search product is a function combined in main page which user can search some specific products.

In controller layer, CartServlet will deal with the request of updating product number, removing product and checking out, add cart into session attribute, and forward this request to corresponding URL. CheckoutServlet will deal with the request of checking out and clear cart. LoginServlet will deal with the request of user logging in.

RegisterServlet will deal with the request of user registering. SearchOrderServlet wil deal with the request of user viewing his/her history order. SearchProductServlet will deal with the request of user searching specific products.

In model layer, ProductIO helps main page read product information from product table in database. It also helps SearchProductServlet read specific products from database. OrderIO helps CheckoutServlet insert order information into order table in database. It also helps SearchOrderServlet read history order information from database. UserIO helps LoginServlet read user information from database. It also helps RegisterServlet insert user information into database.

There are some JavaBeans not showed in model layer: product, order, items, cart and user. Those JavaBeans can help new an object which stores data temporally.

In database, there three tables: Product, User_Info, Order_Info. They store product information, user information and order information. Product table can only be read. Others can be read and wrote by servlets.

3.2 Database Design:

This app chooses SQLite to be the embedded database. Across the app’s architecture, it indicates that the database will be read and wrote many times when user uses this online store. Opening a connection to access database every time is extremely expensive. So using connection pool is better. The database configuration is written into /META-INF/context.xml.

In database, there are three tables: Product, User_Info, Order_Info. Their architecture is as follows:

For Product table:

Product_ID	ProductName	ProductColor	ProductImg	ProductPrice

For User_Info table:

UserID	FirstName	LastName	Email	Password

For Order_Info table:

OrderID	Order_Date	Card_Number	Email	Product_Des	Amount	Total_Price

3.3 Design details:

1. Because SQLite is an embedded database, this app needs to initialize database online: create three tables, insert product information into Product table and save this database under application's /web direction.

2. And servlet will use connection pool to access database, this app needs to initialize connection pool configuration in /META-INF/context.xml.

3. To avoid SQL injection, this app will use prepared statement to execute when access database.

4. Database can't store image directly, so put products' image under /web/resources direction. And then insert images' path into database. Then the app can read the image's path from database and show that image.

4. Discussion

4.1 How to show all the products on main page?

For the main page, there is no specific servlet to handle the request that show all the products. So on the main page itself, there must be some Java code to read product information from database. And then use JSTL tag to show those information in a loop.

4.2 How to decide how many servlets the app will have?

For the app, there are 6 kinds of actions roughly: cart, checkout, login, register, search order and search product. So each action has a servlet to handle corresponding request.

4.3 How to track session?

In CartServlet, every time new a cart, put this cart into session as session attribute. And on cart.jsp, use EL to get cart object from session and use a JSTL tag to display items in cart.

It is the same for user. In LoginServlet, every a user logins, new a user object and put it into session as session attribute. And on some jsp pages, like main page and checkout page, use Java code to check if there is a not null user object in session. If yes, display the user's email, otherwise, provide links to login page and register page.

Then, even if the user exits the browser, next time he/she will see the same cart and skip login action while opens the browser again.

4.4 How to pass query result of database to jsp page?

Use JavaBean to store querying data from database temporally and put that JavaBean into session as session attribute. If a jsp needs to display query result of database, it uses EL to get session attribute and uses JSTL tag to display the order information.

5. Conclusion

This web app realizes some basic functions of an online store as cart, checkout, login, register, search order and search product. User can use this store buy some products with a fake card. Based on the MVC model, this app is easy to extend more functions without affecting already existed functions. This project is a good example to learn a whole web service system from front end to back end.