

The Hong Kong Polytechnic University
Department of Electronic and Information Engineering

EIE3320 Lab 2: Online Bookstore

(Deadline for Submission: **Check the course information**)

Important Note: This is a group project. Two persons form a group. If you insist to do it on your own, one person per group is also fine.

Expected Outcomes

- Understand the principles of object oriented design.
- Apply Java in object oriented software development.
- Apply UML in object oriented software modelling.
- Apply object oriented approach to developing computer software.
- Learn independently and be able to search for the information required in solving problems.
- Present ideas and finding effectively.
- Work in a team and collaborate effectively with others.

Assessment Criteria

Your report should contain (but not limited to) the following:

1. Source code of your solutions (include the parts that you have use MVC techniques).
2. Explanation of how MVC was used in your implementation to simply the program structure.
3. The format of report should include the followings,
 - 3.1 **Introduction:** A detailed description of the objectives and requirements of the program, and a brief description of the methodology.
 - 3.2 **Methodology:** The methodology when implementing the program. It contains,
 - How your team divides the work among the team members?
 - The schedule of implementing the program
 - The MVC structure of the program developed, including
 - The specifications of the classes defined, and the public/private member functions/variables included
 - The flow of execution such as class diagram or flow chart.
 - 3.3 **Program Testing**
 - The validations of your program and confirmed that it is running correctly.
 - Include the execution results of your program captured from the screen.
 - 3.4 **Conclusion**
 - Summarize the experience gained in the program
 - 3.5 **Future Development**
 - Indicate how your program can be extended.
4. Basic features are **COMPULSORY** that included all features described in **Project Description**.
5. Additional features are **OPTIONAL**, it contains,
 - 5.1 **Image Retrieval:** You may provide an image for each record and the image can be displayed in *BookInfo.jsp*.
 - 5.2 **Database Management:** The user may add or remove record(s) in the database

"books".

5.3 **Input Validation:** Additional validations for input data.

5.4 Any other creative and significant features.

Note that additional features will **NOT** contribute to the final marks. You may implement these features if you want to learn more.

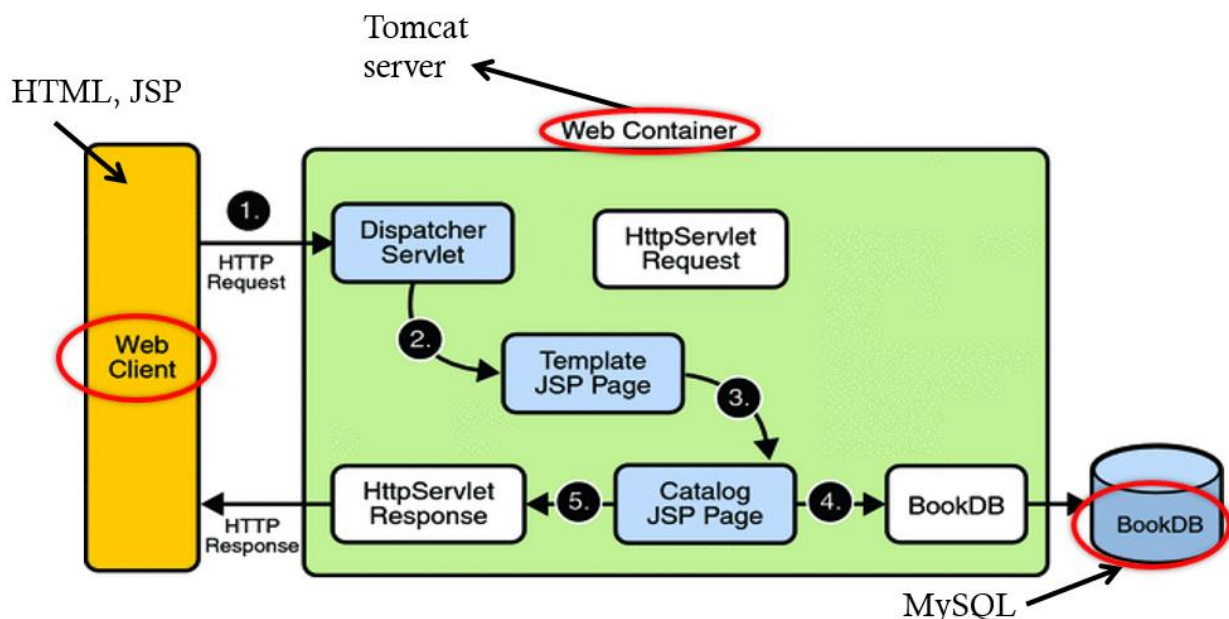
6. Mark distribution,
Basic Features: 60%
Coding style and correctness: 10%
Report: 30%

Submission

1. After finished the program, each team should compress (into zip format) and upload your source code and report to Blackboard and the screen captures should contain your name, student number, and data and time at which you run your program.
(Exception of Tomcat server, SQL server, SDK development kit and other NOT related programs, otherwise, your mark WILL BE DEDUCTED.)
2. Each team member should declare his/her responsibility in the report. Each member will be individually assessed based on the declared responsibility and the result obtained.
3. The report should be in PDF format. It is NOT required to include the complete source code in the report.
4. It is compulsory to use a word processing tool to write your report. The font size must not be bigger than 12 or smaller than 10. Use 1.5 lines spacing on both sides of a page.
(Including all diagrams/tables, the length of the report should not be shorter than 15 pages)

Introduction

In this project, you are required to develop an online bookstore by using HTML, JavaScript, Java Servlet and JSP. The bookstore is a web application through which a user can (1) view a list of books written by an author and (2) purchase book(s) from the bookstore. Through this project, you will come across the design, implementation, and testing phases of a typical web application. The following shows the overview of program structure.



Project Description

In this exercise, you need to create EIGHT files for the web app. Users shall follow the paths shown in Figure 1(f) when using the web app.

1. ***SearchBook.html***: This HTML file should provide two text boxes for a user to input an ISBN, an author name or both to search for book(s) that meet the criteria. A button shall be provided for the user to submit the form (see Figure 1(a)). Note that you are required to check whether the input is empty or not. If both text boxes are empty, you should display an error message asking the user to input again. If the input is valid, the web page will send a request to ***QueryServlet*** after pressing the button.

Replaced by the time
and date at which your
program is run.
(Google "HTML date
and time")

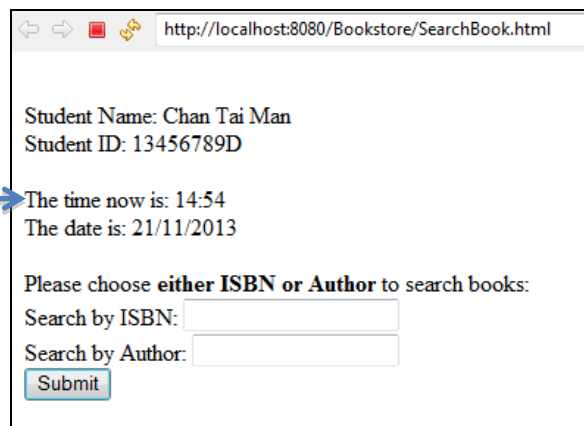


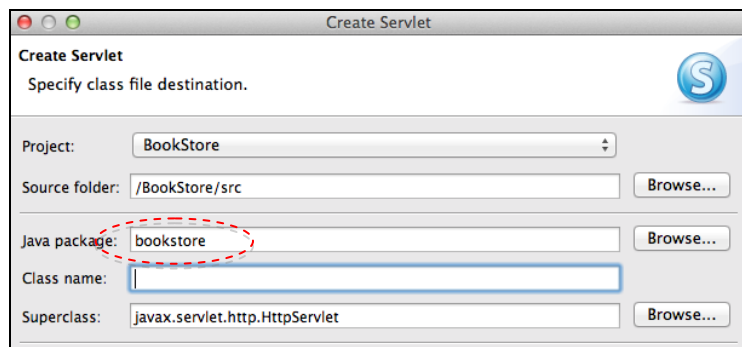
Figure 1(a) Output of *SearchBook.html*

2. ***QueryServlet.java***: This servlet shall receive requests from ***SearchBook.html*** and search a book database to retrieve the books that meet the search criteria specified by the user. It shall save the book information in an `ArrayList` object and associate the object with a session-based attribute. Then, the servlet shall forward its control to the ***BookInfo.jsp*** (see below). A database is provided for the online bookstore and you may download it from <http://www.eie.polyu.edu.hk/~enhylin/Servlet.html>

You should follow the instructions in the Servlet & JSP Tutorial to start a SQL server. The following actions can be used to validate whatever the SQL server has been started and the database has been properly installed.

- a. Use WordPad to read ***Books.sql***.
- b. Type `"show databases;"` in a MySQL client shell to show the content of the database (see Figure 2).
- c. Type `"use books;"` and then `"show tables;"` in the MySQL client shell to show the content of the database (see Figure 3).
- d. Type `"select * from bookinfo;"` to show all book information in the database "books" (see Figure 4).
- e. Type `"select * from bookinfo where isbn='0132404168';"` to show the book information of ISBN 0132404168.

You are required to design a class called `Book` that contains the information of a book and a class called `ShoppingCart` that contains an array of `Book` objects. In case this task is too complicated for you, you may download an example of *[QueryServlet.java](http://www.eie.polyu.edu.hk/~enhylin/Servlet.html)*, *[Book.java](http://www.eie.polyu.edu.hk/~enhylin/Servlet.html)*, and *[ShoppingCart.java](http://www.eie.polyu.edu.hk/~enhylin/Servlet.html)* from <http://www.eie.polyu.edu.hk/~enhylin/Servlet.html>. This page also contains the skeletons and hints of other files. You may read the hints to complete this lab exercise. All servlets in this web application should be under the package “bookstore”, i.e., when you create the servlet, remember to put “bookstore” in the package field as shown below.



3. ***BookInfo.jsp***: This JSP shall display the books that meet the search criteria as shown in Figure 1(b). This can be achieved by retrieving the `ArrayList` object created in *[QueryServlet.java](#)*. A hyperlink “Add to Cart” should also be displayed. The link shall allow users to send a request to *[OrderServlet.java](#)* (see below). A hyperlink “Home” that allows the user can go back to *[SearchBook.html](#)* shall also be implemented in this JSP.

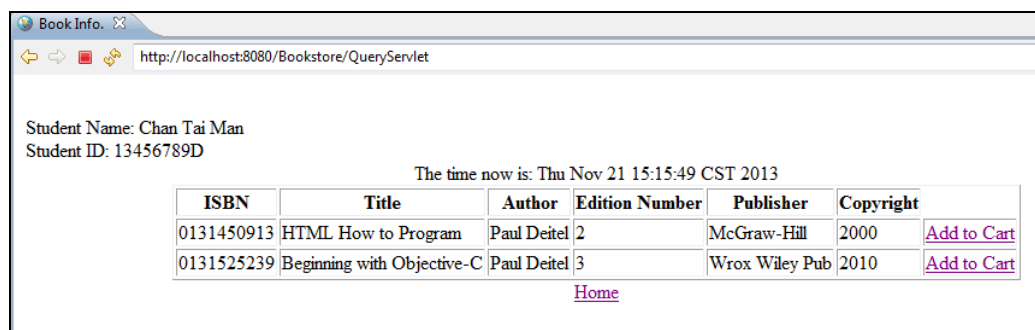


Figure 1(b) Output of *BookInfo.jsp*

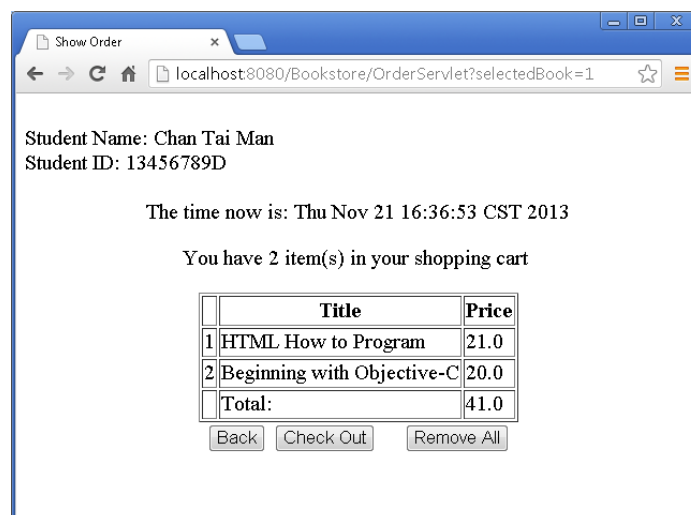
Hints: Your JSP may contain the following scriptnet.

```
<%@page import="java.util.ArrayList" %>
<%@page import="java.util.Date" %>
<%@page import="bookstore.*" %>
<%
    ArrayList<Book> books = (ArrayList<Book>)session.getAttribute("foundBooks");
    int numBooks = books.size();
%>
<center>The time now is: <%= new Date() %></center>
```

4. **OrderServlet.java** and **show-order.jsp**: This servlet and JSP work together to display the order information. The servlet shall receive requests from **BookInfo.jsp** and add the book(s) that the user has selected to his/her shopping cart. The JSP shall display the following information in the form of a table:
- The number of items in the user's shopping cart;
 - The title of each book in the user's shopping cart;
 - The price of each book in the user's shopping cart; and
 - The total price of all books in the user's shopping cart.

Finally, THREE buttons should be added at the bottom of the table:

- A "Back" button allowing the user to go back to the previous web page, i.e., **BookInfo.jsp**;
- A "Check Out" button forwarding the control to **check-out.jsp** (see below) so that the user can buy all books in his/her shopping cart; and
- A "Remove All" button allowing the user to remove all books in his/her shopping cart. To simplify the implementation, the user is not allowed to remove individual books; however, he/she can remove all of the books in the shopping cart in one operation. A snapshot of the output of **show-order.jsp** is shown in Figure 1(c).



Student Name: Chan Tai Man
Student ID: 13456789D

The time now is: Thu Nov 21 16:36:53 CST 2013

You have 2 item(s) in your shopping cart

	Title	Price
1	HTML How to Program	21.0
2	Beginning with Objective-C	20.0
Total:		41.0

Back Check Out Remove All

Figure 1(c) Output of **show-order.jsp**

Here is a code fragment in **OrderServlet.java**. It obtains the user's session and creates a **ShoppingCart** object if none exists.

```
public class OrderServlet extends HttpServlet {
    public void doGet (HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String url = "/show-order.jsp";
        HttpSession session = request.getSession(false);
        if (session == null) {
            System.out.println("Forward to SearchBook.html");
            RequestDispatcher dispatcher =
                getServletContext().getRequestDispatcher("/SearchBook.html");
            dispatcher.forward(request, response);
            return; // Need to return doGet() to prevent the following codes to run
        }
    }
}
```

```

ShoppingCart cart = (ShoppingCart)session.getAttribute("bookstore.cart");
if(cart == null){
    cart = new ShoppingCart();
    session.setAttribute("bookstore.cart", cart);
}
ArrayList<Book> books = (ArrayList<Book>)session.getAttribute("foundBooks");
int i = Integer.parseInt(request.getParameter("selectedBook"));
cart.addBook(books.get(i));
session.setAttribute("bookstore.cart", cart);
ServletContext context = getServletContext();
RequestDispatcher dispatcher = context.getRequestDispatcher(url);
dispatcher.forward(request, response);
}
}

```

5. **remove-all.jsp**: This JSP is invoked when the user presses the Remove All button in *show-order.jsp* (see Figure 1(c)). It should remove all selected books in the shopping cart.
Hints: You may need to use the `removeAll()` method of `ArrayList` class.
6. **check-out.jsp**: This JSP should provide two text boxes for the user to input his/her name and his/her credit card number. Moreover, it should provide two buttons:
 - a) Submit information: for submitting the information in the text boxes to the servlet **ReceiptServlet.java**. An error message should be displayed when any of the text boxes is empty.
 - b) Cancel: for forwarding the control back to *show-order.jsp*.

Fig. 1(d) shows the output of *check-out.jsp*.



Figure 1(d) Output of *check-out.jsp*

7. **ReceiptServlet.java**: This servlet should display an acknowledgement for the purchase order created in *check-out.jsp*. The servlet shall automatically forward to **SearchBook.html** after five seconds. You may search the internet using the keyword “HTML refresh code” to find some example implementation. Figure 1(e) shows the output of this servlet.

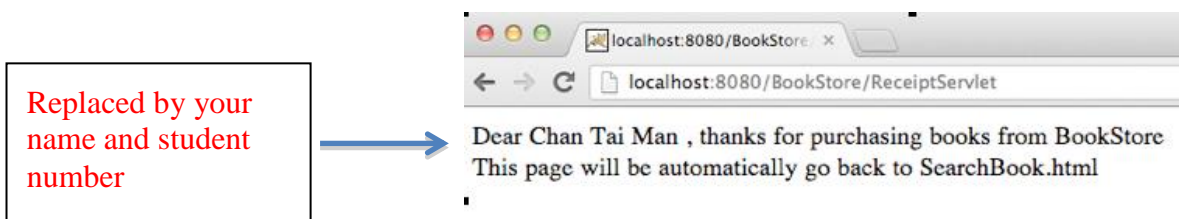


Figure 1(e) Output of *ReceiptServlet.java*

Figure 1(f) shows the interactions of the EIGHTS HTML/servlets/JSP files mentioned above.

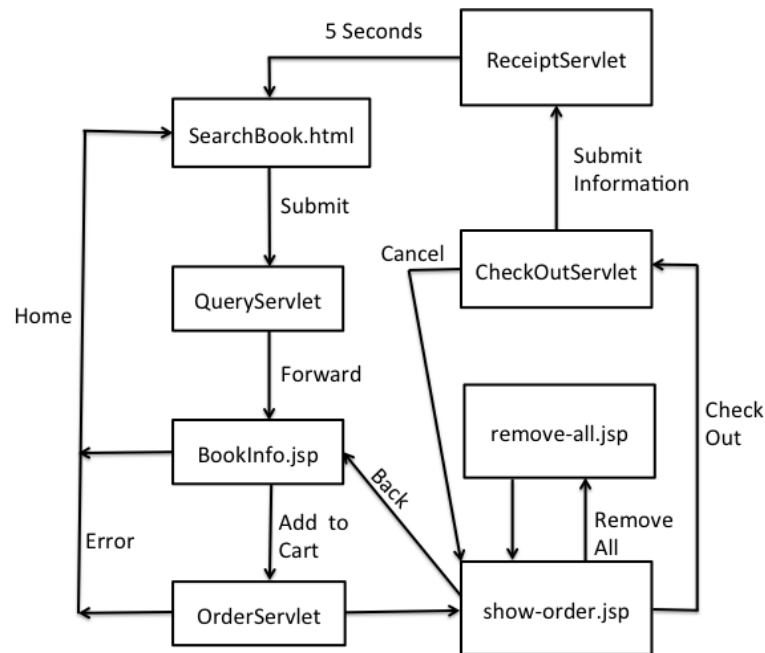


Figure: 1(f)

```

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| books        |
| mysql        |
| performance_schema |
| test         |
+-----+
5 rows in set (0.03 sec)

mysql>

```

Figure 2: Databases in MySQL server

```

mysql> show tables;
+-----+
| Tables_in_books |
+-----+
| authorisbn      |
| authors         |
| bookinfo        |
+-----+
3 rows in set (0.00 sec)

mysql> _

```

Figure 3: The content of the database "books"

```

mysql> use books;
Database changed
mysql> select ISBN, Title, EditionNumber, Publisher, Copyright from bookinfo;
+-----+-----+-----+-----+-----+
| ISBN      | Title                                     | EditionNumber | Publisher    | Copyright |
+-----+-----+-----+-----+-----+
| 0131450913 | HTML How to Program                     | 2            | McGraw-Hill | 2000      |
| 0131525239 | Beginning with Objective-C              | 3            | Wrox Wiley Pub | 2010      |
| 0131828274 | Internet & World Wide Web How to Program | 5            | McGraw-Hill | 2011      |
| 0131857576 | C++ How to Program                      | 5            | Peachpit    | 2005      |
| 0131869000 | Web Servers for Fun and Profit          | 3            | Peachpit Press | 2006      |
| 0132222205 | Java How to Program                     | 3            | Peachpit    | 2010      |
| 0132404168 | iPhone SDK 3                           | 2            | McGraw-Hill | 2010      |
+-----+-----+-----+-----+-----+
7 rows in set (0.03 sec)

mysql>

```

Figure 4: The contents of "bookinfo" in the database "books"

Lawrence Cheung
August 2017