4/24/2017

Internet-based Bookstore

CS4280 Advanced Internet Applications Development
Assignment II: Group Project

Robert Cinca, Lok Hei Li
CITY UNIVERSITY OF HONG KONG

Table of Contents

Prototype Design: Internet-based Bookstore

Overview of Project and its Design Justification

For the group assignment of CS4280, we (Robert Cinca, Lok Hei Li) decided to implement task number 2, the Internet-based bookstore. We decided to base our design on the following criteria:

- Knowledge and Technologies we have learnt in class.
- The Use Cases given to us in the assessment guideline document.
- Existing internet-based bookstores, in particular Amazon.
- Our own knowledge of web development.

Course Knowledge and Technologies

We implemented a range of different technologies, which is detailed in the key features section. A quick overview of the most important concepts that we implemented from class:

- Basic concepts of Java OO programming.
- The use of NetBeans as an IDE for web application development.
- Tomcat Server integration for website hosting and deployment.
- Functioning of servlets, including HTTP requests and responses.
- JSP pages for HTML-heavy pages.
- Form based authentication.
- Database integration via the JDBC using Microsoft SQL.

Use Cases

At all stages of project development, we ensured that the use cases played a vital role in determining our design. The use cases were as follows:

Two actors: Customer and Book Manager.

1. **Browse books**: a *customer* can browse books at all times.
2. **Purchase books**: a *customer* may, at one time, purchase any number of books. A *customer* who is a registered member may use some or all accumulated loyalty points to pay for books purchased. Books purchased using loyalty points are not refundable.
3. **Request for refund of purchased books:** a *customer* who is a registered member may submit request for refund of purchased. The request for refund needs to be authorized by a *book manager*.
4. **Enquire loyalty points:** a *customer* who is a registered member may enquire the loyalty points accumulated.
5. **Authorize refund request:** a *book manager* needs to authorize each request for refund of purchased books.
6. **Manage books' quantity and price:** a *book manager* may add, remove, and change book details and available quantity and price.

These use cases will be referred to in the rest of the report, in order to highlight how we used them to design our project.

Existing Internet-based Bookstores
To implement our use cases, we based our key design choices on existing internet-based bookstores, in particular Amazon. Some of the key concepts that we implemented include:
- Implementing membership through a signup/login and loyalty points system.
- Ability to browse books at any time.
- The 'cart' concept: a place that stores the books the customer wants to buy, similar to an online concept of a shopping basket.
- Pay by card or by points, a key feature of any online marketplace.

Own Knowledge
We also used our own knowledge for the initial design and appearance of the website along with extending certain functionalities to improve the user experience and make the website more robust. Some of the concepts we implemented include:
- HTML for displaying the content on the web page.
- CSS to improve the user experience and user interface (design of web page).
- JavaScript for implementation of small handy functions, such as checking if there is an intervening space in the creation of the username.
- GitHub, a git repository that allows us to share and collaborate during the project. There were several reasons why we decided to use GitHub:
    - The Git repository is safe from file loss, as the project is stored in the cloud.
    - It is simple to revert back to previous versions of the code. There is an easy user interface that displays previous versions and who made modifications.
    - Conversely, it is also simple to keep up with the most up-to-date version.
    - It is easy to access the Git repository from other devices.
    - The use of the branch and fork features permits us to code different parts of the application separately and try out new ideas without damaging the working code.
    - GitHub is widely used so there is extensive support available.

Our GitHub repository can be found here: https://github.com/robertcinca/bookstore

Key Features and their Design Justification
The following section highlights the key features we implemented, along with a justification of their design. Particular attention is paid to the flexibility of the code, to make it easy to implement and change future code. We also link any key feature design to the use cases as a way of achieving the project requirements whilst also not deviating from the main goals of the website.

Form based authentication
We implemented a simple form-based authentication system as a way of registering members to our website. There is also the option to 'continue without logging in', which lets a guest user access the basic features of the website (browse and buy books, use cases 1 & 2) without needing an account.

The authentication system also makes it easy to control access to different parts of the website via web.xml. For example, a normal user can view account details whilst a guest user does not have access to this page. Additionally, pages such as the ‘refund’ and ‘add books’ page are reserved only for users granted ‘admin’ status. This is in line with the majority of internet-based marketplaces that give users different access rights based on their role.

Cart design
The ‘add to cart’ feature, implemented in many online shopping websites, helps the user keep track of the books they want to buy (use case 2), similar to a shopping basket in a real-life supermarket.

Whilst browsing for books, the user can easily add books to the cart using the ‘add to cart’ button. Moreover, the user is free to select the quantity of books desired and change it at a later time if they so desire. As an additional real-life constraint, we implemented the concept of ‘limited supply’ for a particular type of book: the user can only add the available quantity of that book to the cart. The manager can adjust the quota from the admin panel.

The cart is a simple table design that displays the total cost of each row (price x quantity) and the overall cost of all the orders. The user is able to delete entries or change the quantity of the orders.

Pay by card or points
Once the user chooses the ‘pay now’ option from the cart view, they are faced with two options: pay by card or with loyalty points (use case 2).

The loyalty points option only works if the user has enough accumulated points to buy the products. Additionally, the user is must not be a guest. Otherwise, the user is invited to pay by card. The user is warned that books purchased with loyalty points are not refundable.

Please note: no real credit card information is collected, the payment system is not real.

View account detail
Every user who signed up to our website will have their own account. They will be able to view their account details in the Account Detail page. Within the page, there are two main sections: Purchased Books and User details.

Purchased books contains a list of books that the user purchased. The user can view the name and quantity of the book they purchased, as well as the status. Status includes 'purchased', 'refund requested', 'refund accepted' and 'refund declined'.

When the user first purchased the book, the status will be 'purchased'. The user will have the option to request a refund if the book is paid by card. After the request, the status will change to 'refund requested'. The user can cancel the request, which changes the status back to 'purchased', or wait for the admin to authorize the refund. After the authorization from the admin, the status will change to 'refund accepted' or 'refund declined', depending on whether the admin accepted or declined the refund. The user cannot request the same refund again, if the first refund was declined (use case 3).

User Detail contains all the basic information of the user, including username, password and loyalty points (use case 4).

Manager/Admin panel
When admins sign in, instead of viewing the customer browsing page, they will have access to a separate panel of options: authorize refund request, add new books, change book information and delete books from the list.

Admins can authorize refund request on the admin refund page. Within the page, a list of refund request will be shown. Admins can accept or decline each request, which changes the request status to 'refund accepted' and 'refund declined' respectively. (use case 5).

Admins can also add a new book from the add book page, simply by filling in the book name, author, price, loyalty points, stock and image URL in the form. Admins can also change these book details from the browse page, as well as deleting book entries (use case 6).

Design Justification
Most of the features listed above are separated in different Java files. Each file has its own functionality which makes it easier to locate the code for future changes.

Instead of building separate pages for different roles, we use Java servlets to generate dynamic pages. Pages that serve similar functions, such as browse pages for the customer and manager, are designed in the same Java file.

Different actions in the same file are separated with individual processRequest functions. New features can be implemented easily without changing the code from other actions.

CSS is separated from the application logic. Interface design can be altered without changing the code of the application logic.

Addressing Assessment Guideline

We addressed the assessment guidelines in the following way:

- We designed the website based on the use cases presented in the overview document. Their implementation is highlighted in the 'key features' section of this report.
- We implemented the key technologies highlighted in the document, namely, XHTML (HTML 5), cascade style sheet, JavaScript, Java applet, Java Servlet, Java Server Pages, SQL query language. These are described in the 'overview of project' section of this report.

- We also placed the Web Archive File for our Bookstore in Tomcat Webapps as a way of deploying the website.

In order to implement the assessment guidelines, we organized regular meetings to discuss the progress of the project thus far and propose the next steps in order to achieve our goal. We set intermediary deadlines to keep a regular progress on our project and used online development tools such as GitHub for storing our project. We also used Whatsapp as a way of communicating and arranging meetings.

Strengths and Weaknesses of Design

Strengths of our Design

The greatest strength in our design is that it achieves all the use cases, along with a "reasonable set of business rules and the appropriate application logic" that ensures a decent user experience. For example, when the user places an order, they are limited to buying only the quantity available in stock (this is checked with the database entry for stock). In a real bookstore, this feature is important as otherwise users might be unhappy if they are told they cannot receive all the books ordered because they are out-of-stock.

Another strength is the straightforward user experience: by basing our design on pre-existing online bookstores that most users have seen before, they can easily learn to navigate and use our website. The most obvious example of this is the 'add to cart' feature that many users will recognize and know how to use.

We also implemented a responsive web design. The web page is easily accessible and navigable from any screen size, be it a computer, tablet or a smartphone. This was done using CSS media queries, used to define different style rules for different media types/devices.

Weaknesses in our Design

The user interface of our design can be improved further by implementing more CSS code (e.g. animating objects). Due to the limited time available in developing the project, we decided to focus mainly on the practicality and functionality of the website (achieving use cases, having appropriate application logic) rather than the UI. This was also due to the nature of the course CS4280: our intention was to learn how to integrate server-side processing rather than create an aesthetic appearance.

Another weakness is the lack of test cases: a large enterprise (e.g. Amazon) with a full-scale website would implement a range of tests to ensure the website does not 'break' on minor changes in the code. An example of a test would be an URL checker: a test that checks if any of the links point to files that have changed their location, thus warning the developer that they need to update the URL link. We did not implement any tests for two reasons: the first one being the limited time of the project development. The second reason is the fact our website is relatively simple so the need for tests was not very strong. For example, we employ only 5 database tables: a full-scale enterprise might have thousands of tables with millions of rows.

Individual Contributions

We had regular meetings where we agreed what each of us should contribute. This was based on giving each other a similar amount of work and also on what we were interested in.

Robert Cinca

I worked on the following areas:

- Authentication: login/logout/signup. I implemented further design logic to ensure the smooth process of authentication. Some examples of this include:
    - Signup: a user cannot sign up with an existing username. Additionally, the username has to be at least 4 characters and the password 3 characters.
    - Login: some simple validation, for example checking for a space in the username.
- View cart: this page displays a table with books the user has added to cart. This page also gives the user the option to change the quantity or delete entries.
- Payment page: this page allows the user to pay by points or by card.
- Confirmation page: this page displays the books that have been purchased and updates the relevant databases (for example, the new total for the loyalty points).
- JSP pages: I designed the bookstore footer that includes social media links and the disclaimer displayed on all pages.
- JavaScript contributions: I employed basic JS functions to make authentication validation easier.
- CSS: I used CSS for the UI and design of web pages.

Lok Hei Li

- Browse books: I designed and implemented the browsing page for both customer and manager. This page also included the add to cart function for customer view, change book details and delete book entries for manager view.
- Add to cart: customers can add books that they would buy to their cart by filling in the quantity.
- Change/Delete book: the manager can change the book details or delete book entries from the database to modify the book list shown in the browse page.
- Add book: this page allows the manager to add a new book to the database, which will be shown in the browse book page.
- Refund page: this page shows all the refund requests for admins to authorize the requests.
- View Detail: this page displays all the account information, including the purchased books list and user details.
- Purchased books: all purchased book will be shown and the user will have the option to request refunds.
- User detail: the user can view their account details including username, password and loyalty points.

Prototype URL

The prototype is accessible on both our personal CS environments.

URL for Robert Cinca: http://personal.cs.cityu.edu.hk/~robcinca2/cs4280/asgp2/index.html
URL for Lok Hei Li: http://personal.cs.cityu.edu.hk/~lokheili3/cs4280/asgp2/index.html

Please note: The Tomcat server needs to run for the website to work.

Reflection

Robert Cinca

This assignment taught me how to create a full-scale website that takes advantage of server-side processing as a way of achieving the requirements. It implemented concepts from all of the lectures along with the tutorials into one large assignment.

The most difficult part for me was to enable the web development from my MacBook. I ran into difficulties mainly relating to incompatibility between Windows and OS X operating systems. For example, Apple removed support for PPTP VPN connections from their latest OS X release. This type of connection was necessary in order to connect to the CS Labs and be able to access the MSSQL database (otherwise Tomcat would not connect to the database).

I have also learnt more about programming in Java and integrating web development within an IDE and within Java. I also studied more about server-side processing, database querying and implementation.

Lok Hei Li

Throughout this assignment, I learnt a lot from building the website step by step, from designing the UI, building the database, to implementing servlet functions. I made use of what I know and what I learnt from lectures and successfully created a functional website.

Instead of following instructions like in the tutorials, we had to create this website on our own. This allowed me to do so much more self-learning and learn from my mistakes. The most difficult part of the assignment was to implement the server side code since I do not have much experience in it. By going through the lecture notes and searching the internet, I have gained a better understanding of this and I am more comfortable in implementing it in the future.

References

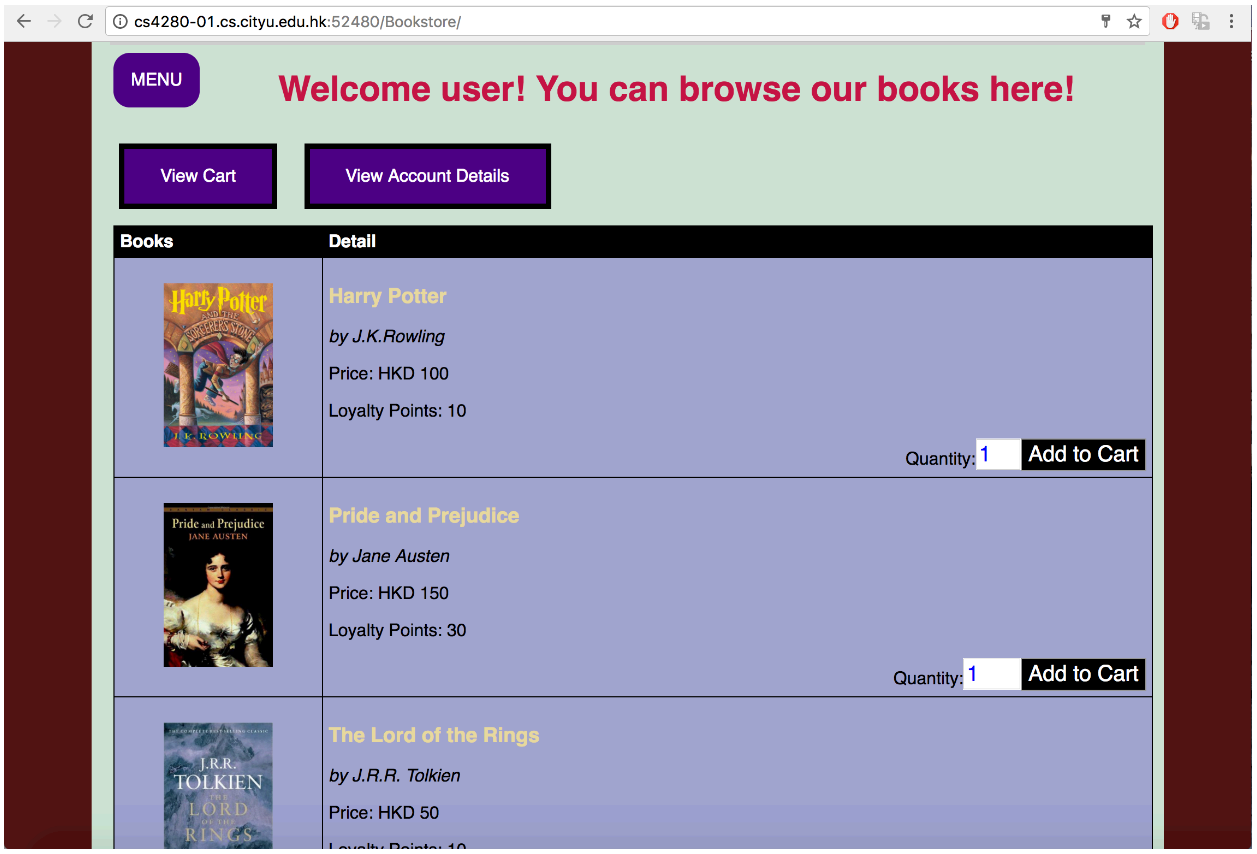
As with any project in computer science, reusing existing code is an important aspect because it saves time and makes it easier to change or integrate extra features in future developments. The idea is simple: there is no need to reinvent the wheel every time you implement something.

The following sources were used in developing the web site:

- Code from the CS4280 tutorials
- Code from the CS4280 lecture slides
- Stack Overflow: a useful forum for programmers to share solutions to programming implementations. Link: http://stackoverflow.com/
- Tomcat Documentation. Link: tomcat.apache.org
- MSSQL Documentation. Link: https://technet.microsoft.com/en-us/library/ms130214(v=sql.90). HYPERLINK "https://technet.microsoft.com/en-us/library/ms130214(v=sql.90).aspx"aspx
- Java Documentation. Link: https://docs.oracle.com/javase/7/docs/api/
- Stack Exchange: similar to stack overflow. Link: https://cs.stackexchange.com/
- w3schools: It provided the basic syntax for HTML, CSS, JS and SQL.
  Link: https://www.w3schools.com/

Appendix

For reference, here is a screenshot of how our browse page looks like:



cs4280-01.cs.cityu.edu.hk:52480/Bookstore/

Welcome user! You can browse our books here!

Books	Detail
	Harry Potter *by J.K.Rowling* Price: HKD 100 Loyalty Points: 10 Quantity: 1 Add to Cart
	Pride and Prejudice *by Jane Austen* Price: HKD 150 Loyalty Points: 30 Quantity: 1 Add to Cart
	The Lord of the Rings *by J.R.R. Tolkien* Price: HKD 50