**Project Document - Assignment – Online Bookstore**

|  |  |
| --- | --- |
| **Version Number** | **Author** |
| **v1.0** | **Andre Alexandrov** |

**Part 1 - Fixing Existing Code Errors**

|  |  |  |
| --- | --- | --- |
| **Code Error** | **Description of Error** | **Fix Details** |
| **Null pointer when loading List<Book>** | **References are in lowercase when attributes listed in uppercase** | **Changed attributes to be lowercase to match naming convention** |
| **Import dispatchers.\* was used** | **The package “dispatchers” does not exist** | **Comment out the import statement.** |
| **List<books>** | **“Books” does not exist** | **Changed “Books” to “Book”** |

**Part 2 – Reasons for using IDE**

|  |
| --- |
| 1. **NetBeans is created specifically for java.** 2. **NetBeans has native Database integration** 3. **NetBeans provides comprehensive support for JSP development** |

**Part 3 – Java EE Best Practices/Patterns- Summary (Research)**

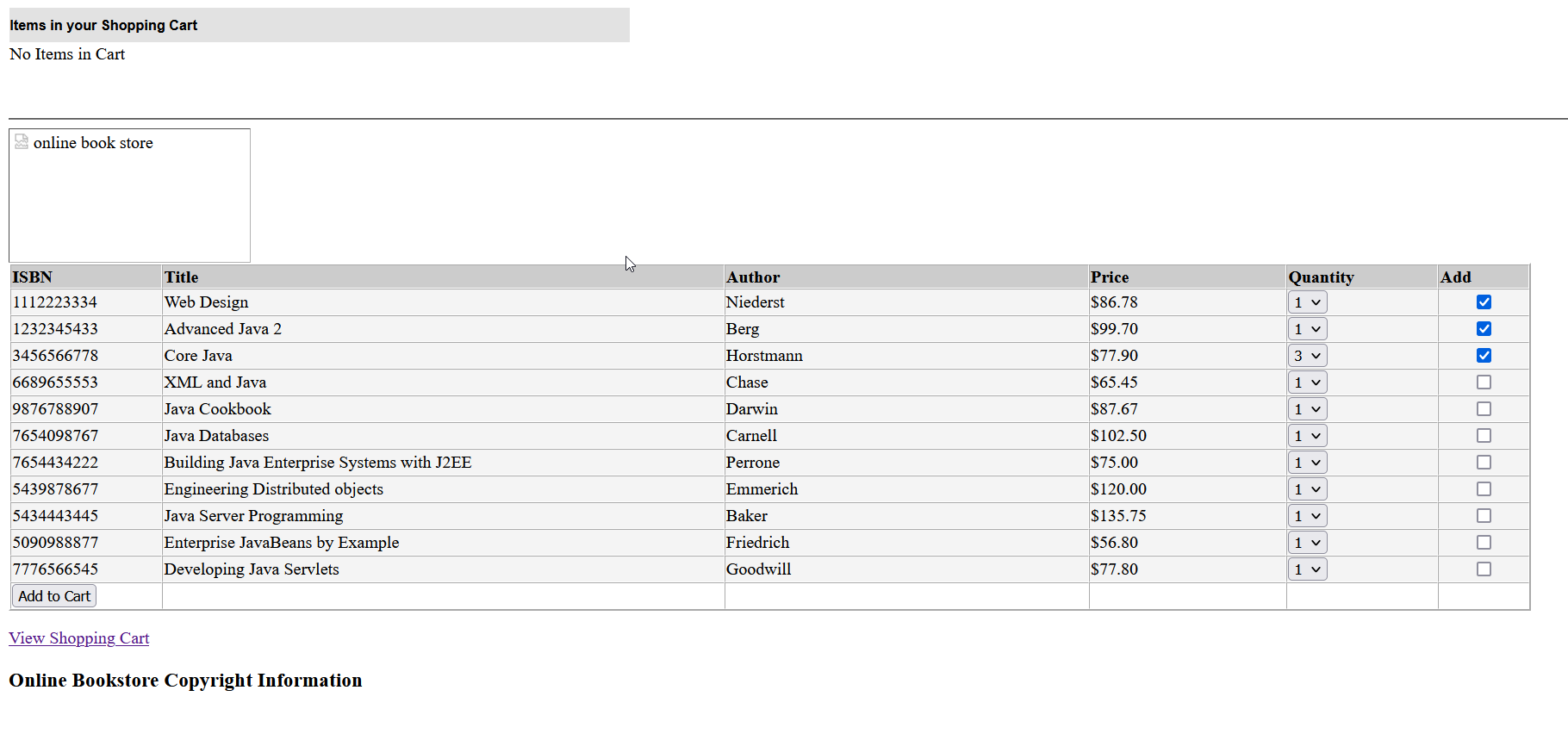
(Include the references to your sources)

|  |
| --- |
| ViewHelper pattern :  The view helper pattern in Java EE is a design pattern that separates the presentation logic from the business logic by using helper classes. It encapsulates data and behavior specific to the view, providing a clean and modular approach for building user interfaces.  https://www.oracle.com/java/technologies/viewhelper.html  Template pattern :  The Template pattern in Java EE is a behavioral design pattern that defines the skeleton of an algorithm in a superclass, allowing subclasses to implement specific steps. It promotes code reuse and encapsulates common behavior, enabling flexible customization and extension of the algorithm in different scenarios.  https://www.javatpoint.com/template-pattern  Command pattern:  The command pattern is a behavioural design pattern that encapsulates actions as objects, including method calls and required data. It promotes decoupling between command producers and consumers, facilitating flexibility and extensibility. It is often referred to as the producer-consumer pattern.  https://www.baeldung.com/java-command-pattern |

**Part 4 – Test Plans**

Document your test cases/test data /expected and actual outcomes using a desk-check format. Also include all screen shots of test run executions of all Use Cases here.

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case | Test Data | Expected result | Actual result |
| View books | Book database | All books listed | All books listed |

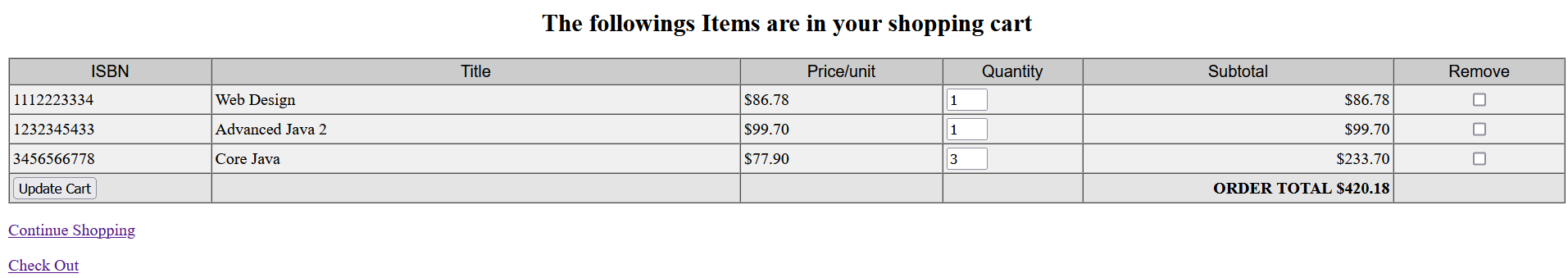
****

|  |  |  |  |
| --- | --- | --- | --- |
| Add books to cart | 1x Niederst, 1x Berg 3x Horstmann | Test data added to cart | Test data was added to cart |

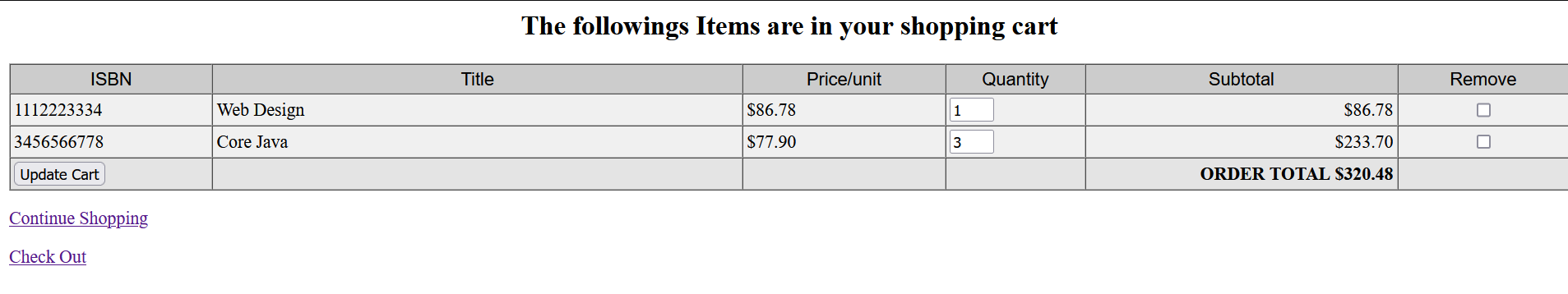
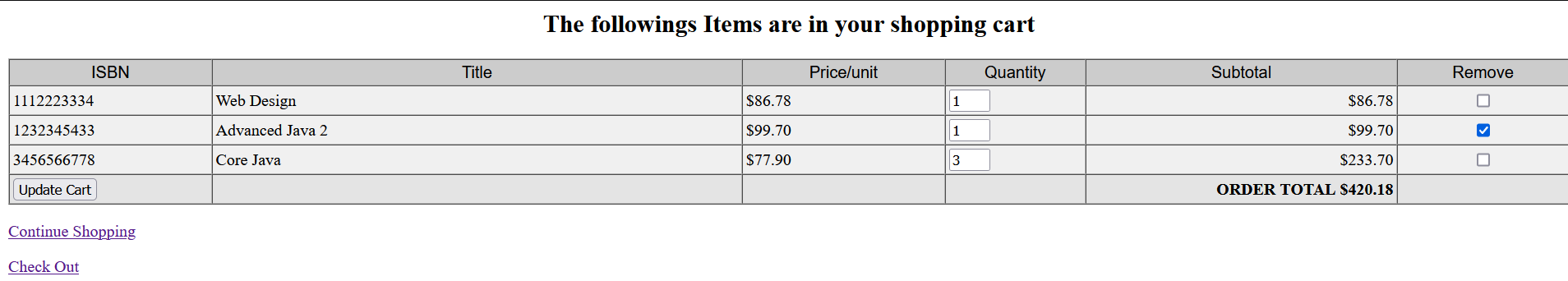
**A screenshot of a computer

Description automatically generated with medium confidence**

|  |  |  |  |
| --- | --- | --- | --- |
| View shopping cart shows books | Books in cart from previous test | 1x Niederst, 1x Berg 3x Horstmann listed in cart | 1x Niederst, 1x Berg 3x Horstmann listed in cart |

****

|  |  |  |  |
| --- | --- | --- | --- |
| remove book from cart/update cart button working | **Books in cart** | 1x Niederst, 3x Horstmann listed in cart/ 1xBerg removed | 1x Niederst, 3x Horstmann listed in cart/ 1xBerg removed |

****

|  |  |  |  |
| --- | --- | --- | --- |
| **Continue shopping button works** | **Cart items** | **All cart items stay in cart and all books are listed** | **All cart items stay in cart and all books are listed** |

**A screenshot of a computer

Description automatically generated with medium confidence**

|  |  |  |  |
| --- | --- | --- | --- |
| Check out works | Items in shopping cart | Price of order matches check out order amount and checkout items match cart items | Price of order matches check out order amount and checkout items match cart items |

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a shopping cart

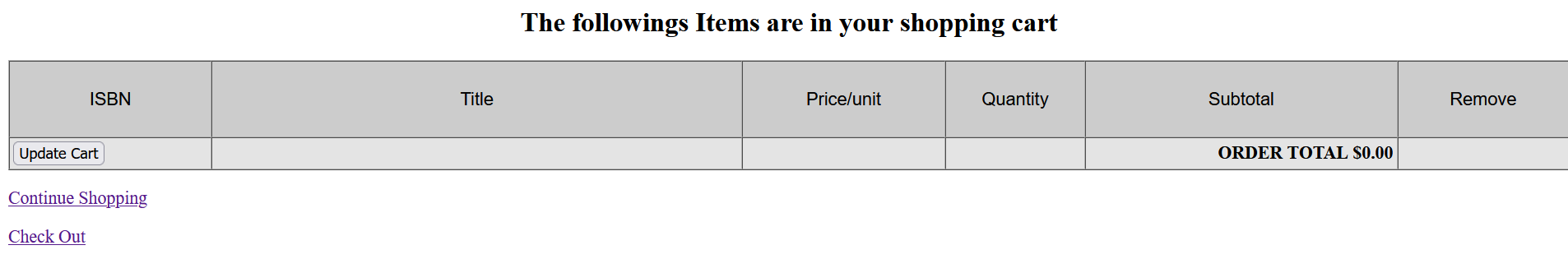
Description automatically generated with medium confidence**

|  |  |  |  |
| --- | --- | --- | --- |
| **Submit button shows thankyou jsp** | **“check out” button then “Submit button”** | **Thankyou view is shown** | **Thankyou view is shown** |

**A picture containing screenshot, text, font, line

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **Remove all items from cart and check out** | **No items in cart.** | **Error shown that no books are in the checkout** | **Error not shown, test Failed** |

****

**Part 5 – java doc System Documentation**

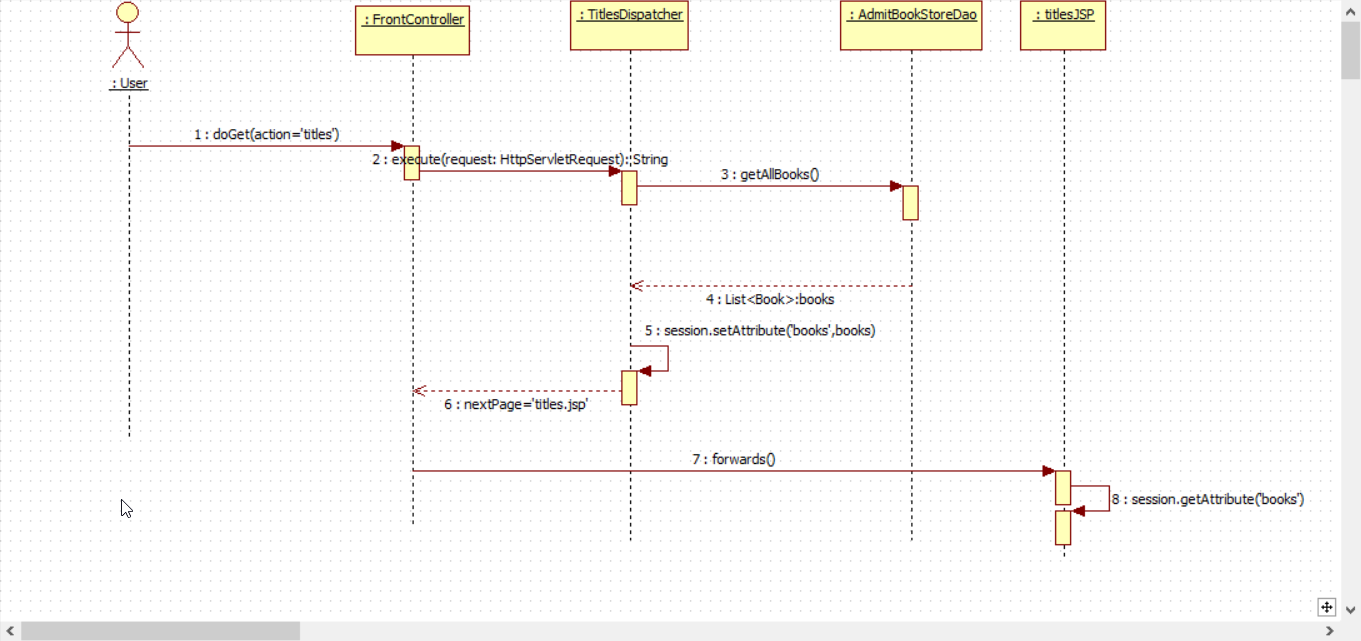
Include the URL to access the above documentation of the re-engineered application

|  |
| --- |
| file:///C:/Users/Andre/Desktop/BookShop/dist/javadoc/index.html |

**Part 6 – Final UML Mode**

Include separate sequence diagrams for each of the Use Cases here.

View Books



Add to cart

A picture containing text, handwriting, parallel, diagram

Description automatically generated

Delete Item from cart

A picture containing text, parallel, diagram, screenshot

Description automatically generated

View Items

A picture containing text, diagram, parallel, plan

Description automatically generated

Checkout Items

A picture containing text, diagram, line, screenshot

Description automatically generated