**Project Document - Assignment – Online Bookstore**

|  |  |
| --- | --- |
| **Version Number** | **Author** |
| **2.0** | **Jesse Hamilton-Young** |

**Part 1 - Fixing Existing Code Errors**

|  |  |  |
| --- | --- | --- |
| **Code Error** | **Description of Error** | **Fix Details** |
| **Null pointer when loading a list of books** | **Books session attribute is listed as uppercase when all references are in lower case** | **Changed the uppercase set attribute function to match its lower-case get attribute counter parts** |
| **Null pointer when clicking the add to cart button without any books selected** | **Null pointer tripped as the cart had nothing to populate itself with** | **Added an error page to catch the issue before crashing the site, advising the user there were no books selected and to try again** |
| **Book object typecast as book in the (requestedAction.equals("home")) section of the front controller** | **Book object typecast as Books causing an error** | **Changed the typcast to be the correct spelling i.e. Book** |
| **Session invalidates on the error page** | **Session invalidates causing issues to the flow of events** | **Commented out the invalidate session command to allow a user to hit an error, press back and continue browsing** |

**Part 2 – Reasons for using IDE**

|  |
| --- |
| **NetBeans supports java – meaning we can design our project with an object orientated approach**  **NetBeans allows us to use the standard tag library in our jsp meaning front end devs can do more and ease the workload across the backend**  **NetBeans allows for server accessibility – meaning it has built in functionality for testing how the product will work** |

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

(Include the references to your sources)

|  |
| --- |
| **View Helper Design Pattern – the view helper pattern separates the view I.E. jsp from the processing of the model data for example the Java Standard Tag Library allows us to build a view without embedding java code directly into the JSP.**  **(**[Core J2EE Patterns - View Helper (oracle.com)](https://www.oracle.com/java/technologies/viewhelper.html)**)**  **Template pattern- allows a super class to hold the bare bones of an object but allow a subclasses to override parts of the algorithms without changing the structure i.e. our IDispatcher class and proper dispatcher subclasses.**  **(**[Template Pattern - Javatpoint](https://www.javatpoint.com/template-pattern)**)**  **Command Pattern – makes a request a stand-alone object that contains the information about the request. Allows request to be passed as method arguments introducing more freedom and options with our code.**  **(**[Command (refactoring.guru)](https://refactoring.guru/design-patterns/command)**)** |

**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 Case | Test Data | Expected result | Actual result | Pass/Fail |
| View Books | Startup of application | Initial display of database items in the web form | All books are displayed | All books are displayed | Pass |
|  | | | | | |
|  | Returning to the displayed books from the cart | Clicking on the continue shopping link from the cart | Return to the title page with all books displayed | Returned to title page with all books displayed correctly | Pass |
|  | | | | | |
| Add books to cart | Adding nothing to the cart | Adding nothing before clicking add to cart | Error page is displayed advising the user that nothing is selected | Error page is displayed advising the user that nothing is selected | Pass |
|  | | | | | |
|  | Adding all books to the cart | Adding all books to the cart before clicking add to cart | All books displayed in the top left cart table | All books displayed in the top left of the screen in the cart items table | Pass |
|  | | | | | |
| Delete items from cart | Deleting no items from the cart | Selecting no items to remove before updating cart | No items are deleted; page remains the same | No items deleted; page remains the same | Pass |
|  | | | | | |
|  | Deleting all items | Selecting all items to be removed before clicking add to cart | All items are removed from the cart | All items removed from cart | Pass |
|  | | | | | |
| View Items in cart | Viewing an empty shopping cart | Clicking view cart while the cart is empty | System displays an empty cart | System displayed an empty cart | Pass |
|  | | | | | |
|  | Viewing the shopping cart with items | Clicking view while cart is filled with items | Systems displays the cart full of items | System displayed the cart full of items | Pass |
|  | | | | | |
| Checkout items in cart | Checking out with no items in the cart | Clicking checkout whilst the cart is empty | Application does not allow the user to continue | Application proceeded with the checkout process | Fail |
|  | | | | | |
|  | Checking out with items in the cart | Clicking checkout whilst there are items in the cart | Application proceeds with the checkout process | Application proceeded with the checkout process | Pass |
|  | | | | | |

**Part 5 – java doc System Documentation**

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

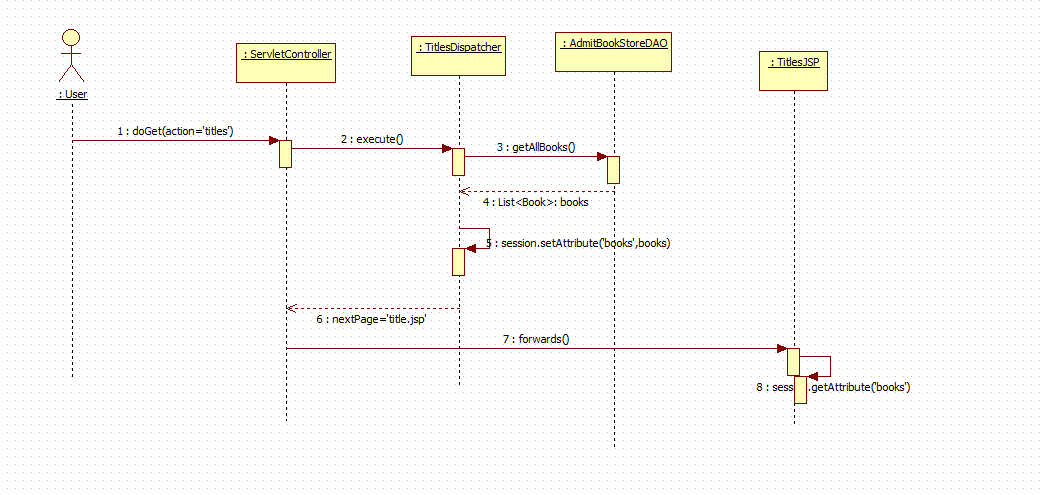
|  |
| --- |
| [Generated Documentation (BookShop)](file:///C:\Users\001175001\OneDrive%20-%20TAFE%20SA\Documents\Java%20Enterprise\Assignment\BookShop\dist\javadoc\index.html) |

**Part 6 – Final UML Mode**

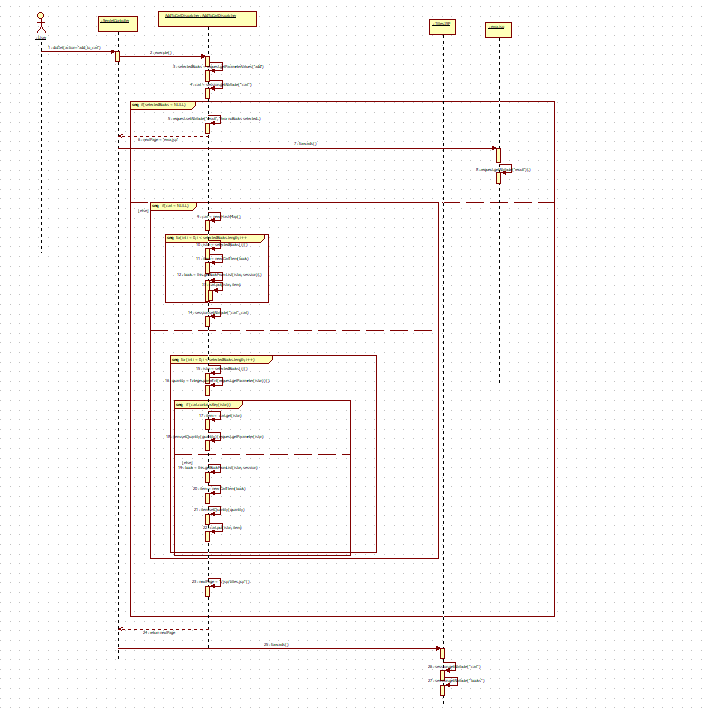
Include separate sequence diagrams for each of the Use Cases here.

As these are barely readable, I’ve included a copy of the UML file with my submission!

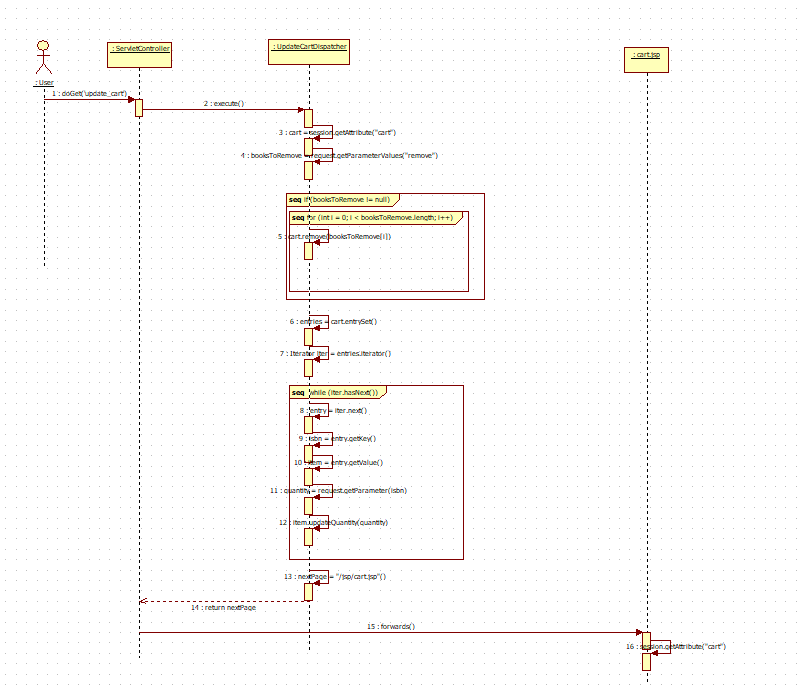
View books



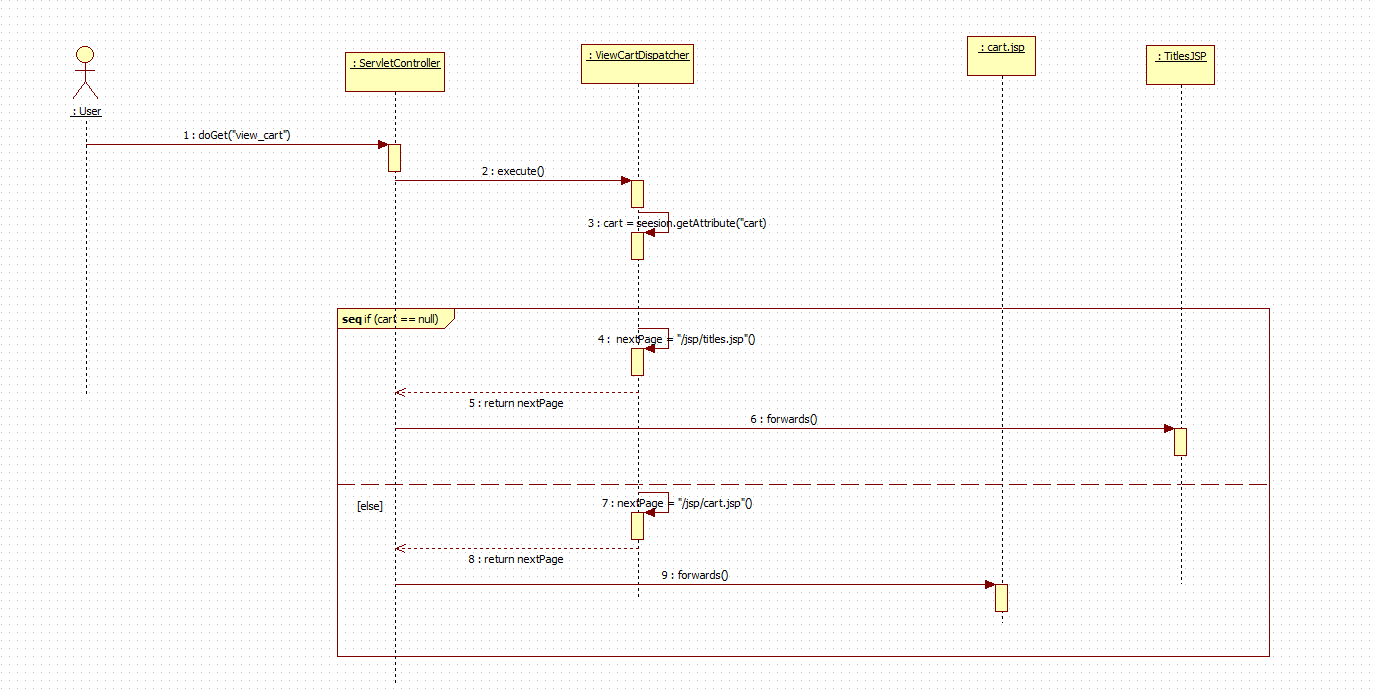
Add books to cart



Delete from cart



View items in cart



Checkout

