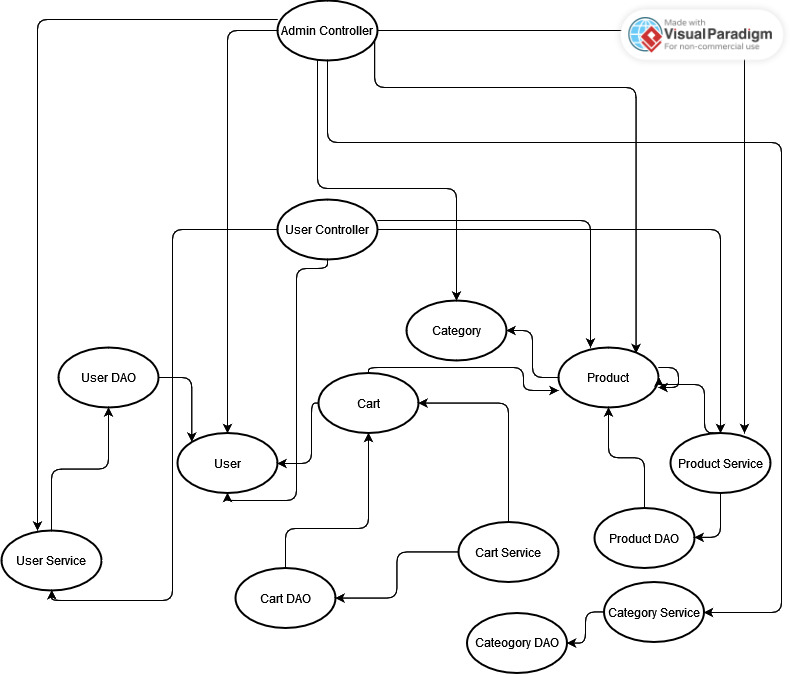
Call Graph Based Integration: ****

* **Adjacency Matrix:**

|  | **AC** | **UC** | **P** | **C** | **UDAO** | **US** | **CDAO** | **CS** | **PS** | **PDAO** | **CtS** | **U** | **Ct** | **CtDAO** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **AC** |  |  | X |  |  | X |  |  | X |  | X | X | X |  |
| **UC** |  |  | X |  |  | X |  |  | X |  |  | X |  |  |
| **P** | X | X |  | X |  |  |  |  | X | X |  |  | X |  |
| **C** |  |  | X |  |  |  | X | X |  |  |  | X |  |  |
| **UDAO** |  |  |  |  |  | X |  |  |  |  |  | X |  |  |
| **US** | X | X |  |  | X |  |  |  |  |  |  |  |  |  |
| **CDAO** |  |  |  | X |  |  |  | X |  |  |  |  |  |  |
| **CS** |  |  |  | X |  |  | X |  |  |  |  |  |  |  |
| **PS** | X | X | X |  |  |  |  |  |  | X |  |  |  |  |
| **PDAO** |  |  | X |  |  |  |  |  | X |  |  |  |  |  |
| **CtS** | X |  |  |  |  |  |  |  |  |  |  |  | X | X |
| **U** | X | X |  | X | X |  |  |  |  |  |  |  |  |  |
| **Ct** | X |  | X |  |  |  |  |  |  |  | X |  |  |  |
| **CtDAO** |  |  |  |  |  |  |  |  |  |  | X |  |  |  |

**Neighborhood Integration:**

|  |
| --- |
| Total Nodes: 14  Neighborhoods: 14  Sink Nodes: 3  Test Sessions: 14-3 = 11 |

* **Test Sessions:**

|  |  |
| --- | --- |
| T-01 | Admin Controller, User Service, User, Category, Category Service, Product, Product Service |
| T-02 | User Controller, User Service, User, Product Service, Product |
| T-03 | Product, User Controller, Admin Controller, Product Service, Product DAO, Category, Cart |
| T-04 | Cart, Cart Dao, Cart Service, User, Product |
| T-05 | User DAO, User, User Service |
| T-06 | User Service, User DAO, Admin Controller, User Controller |
| T-07 | Cart DAO, Cart, Cart Service |
| T-08 | Cart Service, Cart, Cart DAO |
| T-09 | Product Service, Product DAO, User Controller, Admin Controller, Product |
| T-10 | Product DAO, Product, Product Service |
| T-11 | Category Service, Category, Admin Controller, Category DAO |

**User Functionalities:**

1. **Use Case for Login User:**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Login User | |
| **Preconditions** | None | |
| **Typical Case of Events/ Main Success Scenario (U: User, S: System)** | **Step** | **Action** |
| 1 | U: Enters Username |
| 2 | U: Enters Password |
| 3 | U: Clicks “Login” Button |
| 4 | S: Check entered credentials are valid from the database. |
| 5 | S: Redirects to Welcome Page. |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 4a | Credentials not found. S: Reloads the login page. |
| 4b | Username is incorrect. S: Reloads the login page. |
| 4c | Password is incorrect. S: Reloads the login page. |

1. **Decision Table for Login User:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Conditions | - | Rule 1 | Rule 2 | Rule 3,5 | Rule 4 | Rule 6 | Rule 7,8 |
| Input Valid Username | T | T | - | F | F | - |
| Input Valid Password | T | T | F | T | T | F |
| Input Complete Credentials in Fields | T | F | T | T | F | F |
| Actions | Redirects to White label Error Page |  |  |  |  |  |  |
| Prompts to fill empty fields |  |  |  |  |  |  |
| Redirects to Welcome Page |  |  |  |  |  |  |
| Invalid Case |  |  |  |  |  |  |

1. **Test Case for Login User:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Login to Account | Valid Username & Password (username: admin, password:123) | Redirects to Homepage | Redirects to Homepage |
| 2 | Invalid | Test invalid username | Invalid Username (abc) & Valid Password (123) | Display Error Message | Redirects to Error Page; White label Error |
| 3 | Invalid | Test invalid password | Valid Username (admin) & Invalid Password (412) | Display Error Message | Redirects to Error Page; White label Error |
| 4 | Invalid | Test empty field (blank username) | No Username & Valid Password | Display Error Message & Prompt user to fill all fields | Prompts user to enter username, doesn’t display error message |
| 5 | Invalid | Test empty field (blank password) | Valid Username & No Password | Display Error Message & Prompt user to fill all fields | Prompts user to enter password, doesn’t display error message |

1. **Use Case for User Registration:**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Register User | |
| **Preconditions** | User clicks “Register Here” option on Login Page | |
| **Typical Case of Events/ Main Success Scenario (U: User, S: System)** | **Step** | **Action** |
| 1 | U: Enters Username |
| 2 | U: Enters Email |
| 3 | U: Enters Password |
| 4 | U: Enters Address |
| 5 | U: Clicks “Register” button |
| 6 | S: Validates credentials |
| 7 | S: Redirects to Login Page |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 1a | Empty Username Field S: Prompts user to enter username |
| 2a | Empty Email Field S: Prompts user to enter email |
| 3a | Empty Password Field S: Prompts user to enter password |
| 4a | Empty Address Field S: Registers other credentials in database and redirects user to login page |

1. **Decision Table for User Registration:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Conditions | - | Rule 1,2 | Rule 3,4,7,8 | Rule 5,6 | Rule 9,10,11,12,13,14,15,16 |
| Input Username | T | T | T | F |
| Input Email Address | T | - | F | - |
| Set Password | T | F | T | - |
| Input Address | - | - | - | - |
| Actions | Prompt to fill empty fields |  |  |  |  |
| Redirects to Login Page |  |  |  |  |

1. **Test Case for User Registration:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Register new user | Valid Username (1-32 characters long), Email, Password, & Address | New User Registered Message shown, Redirects to Welcome Page | Redirects to Login Page |
| 2 | Valid | Test empty field (blank address) | Valid Username (1-32 characters long), Email, Password, & No Address | New User Registered Message shown, Redirects to Welcome Page | Redirects to Login Page |
| 3 | Invalid | Test invalid username | Username longer than 32 characters, Valid Email, Password, & Address | Show Error Message, Prompt User to re-enter username | Redirects to Error Page: White label Error Page |
| 4 | Invalid | Test empty field (blank username) | Valid Email, Password, & Address, No username | Show Error Message, Prompt User to enter username | Prompts user to enter username |
| 5 | Invalid | Test empty field (blank password) | Valid Email, Username, & Address, No Password | Show Error Message, Prompt User to enter password | Prompts user to enter password |
| 6 | Invalid | Test Invalid Email | Email shorter than 6 characters, Valid Username, Password, & Address | Prompts User to edit email field | Prompts user to use at least 6 characters in email |
| 7 | Valid | Test Invalid Email | Email not following \*\*\*@\*\*\*.com format, Valid Username, Password, & Address | Display Error Message: Invalid Email, Prompt User to re-enter email address | Redirects to Login Page |
| 8 | Valid | Test invalid username | Username has multiple spaces/special characters, Valid Email, Password, & Address | Show Error Message, Prompt User to re-enter username | Redirects to Login Page |

**Admin Functionalities:**

1. **Use Case for Admin Login:**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Admin Login | |
| **Preconditions** | None | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Enters Username |
| 2 | U: Enters Password |
| 3 | U: Clicks “Login” Button |
| 4 | S: Looks up if entered credentials are valid from the database. |
| 5 | S: Redirects to Welcome Page. |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 4a | Credentials not found. S: Reloads the login page. (It should also show an error message that account doesn’t exist, but it doesn’t) |
| 4b | Username is incorrect. S: Reloads the login page. (It should also show an error message that login credentials are incorrect, but it doesn’t) |
| 4c | Password is incorrect. S: Reloads the login page.  (It should also show an error message that login credentials are incorrect, but it doesn’t) |

**2.Decision Table for Admin Login:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Conditions | - | Rule 1 | Rule 2 | Rule 3,5 | Rule 4 | Rule 6 | Rule 7,8 |
| Input Valid Username | T | T | - | F | F | - |
| Input Valid Password | T | T | F | T | T | F |
| Input Complete Credentials in Fields | T | F | T | T | F | F |
| Actions | Redirects to White label Error Page |  |  |  |  |  |  |
| Prompts to fill empty fields |  |  |  |  |  |  |
| Redirects to Admin Dashboard |  |  |  |  |  |  |
| Invalid Case |  |  |  |  |  |  |

1. **Test Case for Admin Login:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Login to Account | Valid Username & Password (username: admin, password:123) | Redirects to Homepage | Redirects to page |
| 2 | Invalid | Test invalid username | Invalid Username (sqe) & Valid Password (123) | Display Error Message | Redirects to Error Page; Whitelabel Error |
| 3 | Invalid | Test invalid password | Valid Username (admin) & Invalid Password (412) | Display Error Message | Redirects to Error Page; Whitelabel Error |
| 4 | Invalid | Test empty field (blank username) | No Username & Valid Password | Display Error Message & Prompt user to fill all fields | Prompts user to enter username, doesn’t display error message |
| 5 | Invalid | Test empty field (blank password) | Valid Username & No Password | Display Error Message & Prompt user to fill all fields | Prompts user to enter password, doesn’t display error message |

1. **Test Case for Admin Dashboard:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Test Manage Button Functionality under “Categories” | Click “Manage” button in Categories | Redirects to Categories Web Page | Redirects to Categories Web Page |
| 2 | Valid | Test Manage Button Functionality under “Products” | Click “Manage” button in Products | Redirects to Products Web Page | Redirects to Products Web Page |
| 3 | Valid | Test Manage Button Functionality under “Customers” | Click “Manage” button in Customers | Redirects to Customers Web Page | Redirects to Customers Web Page |
| 4 | Valid | Test Home Page Button Functionality | Click “Home Page” button | Redirects to Home Page | Redirects to Error Message Page: Whitelabel Error Page |
| 5 | Valid | Test Logout Button Functionality | Click “Logout” button | Redirects to Admin Login Page | Redirects to Login Page |

1. **Use Case for Admin Categories Functionality (Add Category):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Add Category | |
| **Preconditions** | Category page should be rendered | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Press Add Category button |
| 2 | S: Shows a pop-up window with input filed |
| 3 | U: Write the Category into the input field |
| 4 | U: Press Save Changes button |
| 5 | S: Redirects to Categories Page which shows the list of added categories |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 1a | S: Throw Error Message |
| 3a | Leave the input field empty.  S: Show a dialogue box message, Prompt user to fill this field |

1. **Test Case for Admin Categories Functionality (Add Category):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Add new category | Any Alphanumeric value, can include special characters | Displays Message of “New Category Successfully added”, Displays list of Categories with Newly Added category in the list | Displays list of Categories with Newly Added category in the list |
| 2 | Invalid | Test Invalid Input (empty category field) | Empty Category Field | Show Error Message, Prompt user to fill this field. | Show Error Message, Prompt user to fill this field. |

1. **Use Case for Admin Categories Functionality (Update Category):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Update Category | |
| **Preconditions** | Category page should be rendered | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Press Update button |
| 2 | S: Shows a pop-up window with two input fields one is the serial or id, but it is not editable the other is the category name |
| 3 | U: Edit the Category Name in the input field |
| 4 | U: Press Update Changes button |
| 5 | S: Redirects to Categories Page which shows the list of added categories |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 1a | S: Throw Error Message |
| 3a | Leave the input field empty.  S: Show a dialogue box message, Prompt user to fill this field |

1. **Test Case for Admin Categories Functionalities (Update Category):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Check update existing category | Alphanumeric input | Show updated list of categories | Show updated list of categories |
| 2 | Valid | Test invalid input (empty field) | No input/ Empty fields | Show an error message to fill the input field. | Show updated list of categories |
| 3 | Valid | Test invalid input (use special characters as field input) | Special characters | Show an error message to enter a valid input | Show updated list of categories |

1. **Use Case for Admin Categories Functionality (Delete Category):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Delete Category | |
| **Preconditions** | Category page should be rendered | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Press Delete button |
| 2 | S: Redirects to Categories Page which shows the list of added categories after deleting the category |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 2a | S: Display error message, cannot delete category |

1. **Test Case for Admin Categories Functionality (Delete Category):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Delete an existing category | Press Delete button | Delete the category, Display List of Categories without deleted category | Redirects to Error Message Page: Whitelabel Error Page |

1. **Use Case for Admin Categories Functionality (Add Product in Products):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Add Product | |
| **Preconditions** | Product page should be rendered | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Press Add Product button |
| 2 | S: Shows a pop-up window with input fields |
| 3 | U: Write the Name into the name input field |
| 4 | U: Select Product Category |
| 5 | U: Fill Price |
| 6 | U: Fill Weight in grams |
| 7 | U: Fill Available Quantity |
| 8 | U: Fill Product Description |
| 9 | U: Fill Image link |
| 10 | U: Press Submit button |
| 11 | S: Render the Product page which shows the list of added products |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 1a | S: Throw Error Message |
| 3a | The input field empty.  S: Show a dialogue box message, prompts the user to fill this field |
| 4a | Input field is empty. S: Redirects to Whitelabel Error Page |
| 5a | Input field is empty. S: Show a dialogue box message, prompts the user to fill this field |
| 6a | Input field is empty. S: Show a dialogue box message, prompts the user to fill this field |
| 7a | Input field is empty. S: Show a dialogue box message, prompts the user to fill this field |
| 8a | Input field is empty. S: Redirects to Whitelabel Error Page |
| 9a | Input field is empty. S: Show a dialogue box message, prompts the user to fill this field |

1. **Test Case for Admin Categories Functionality (Add Product in Products):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Add New Product | Valid Product Name, Category, Price, Weight in grams, Available Quantity, Product Description, & Product Image | Redirect to Products Page, displays updated list of products with newly added product shown. | Redirect to Products Page, displays updated list of products with newly added product shown. |
| 2 | Invalid | Test invalid input (empty field) | No Product Name, Valid Category, Price, Weight in grams, Available Quantity, Product Description, & Product Image | Display Error Message, Prompt User to enter product name | Prompt User to enter product name |
| 3 | Invalid | Test invalid input (empty field) | Valid Product Name, Price, Weight in grams, Available Quantity, Product Description, & Product Image, No Category | Display Error Message, Prompt User to select category | Redirects to Error Page: Whitelabel Error Page |
| 4 | Invalid | Test invalid input (empty field) | No Product Price, Valid Product Name, Category, Weight in grams, Available Quantity, Product Description, & Product Image | Display Error Message, Prompt User to enter product price | Prompt User to enter product price |
| 5 | Invalid | Test invalid input (empty field) | No Product Weight in grams, Valid Product Name, Category, Price, Available Quantity, Product Description, & Product Image | Display Error Message, Prompt User to enter product weight in grams | Prompt User to enter product weight in grams |
| 6 | Invalid | Test invalid input (empty field) | No Available Quantity, Valid Product Name, Category, Price, Weight in grams, Available Quantity, Product Description, & Product Image Link | Display Error Message, Prompt User to enter product’s available quantity | Prompt User to enter product’s available quantity |
| 7 | Invalid | Test invalid input (empty field) | No Product Description, Valid Product Name, Category, Price, Weight in grams, Available Quantity, & Product Image Link | Display Error Message, Prompt User to enter Product Description | Redirect to Products Page, displays updated list of products with newly added product (without description) shown. |
| 8 | Invalid | Test invalid input (empty field) | Valid Product Name, Category, Price, Weight in grams, Available Quantity, Product Description, & No Product Image Link | Display Error Message, Prompt User to enter product image link | Prompt User to enter product image link |
|  |  |  | Valid Product Name, Category, Price, Weight in grams, Available Quantity, Product Description, & Invalid Product Image Link | Display Error Message, Prompt User to enter valid product image link | Redirect to Products Page, displays updated list of products with newly added product shown (error shown on product image). |

1. **Use Case for Admin Categories Functionality (Update Product in Products):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Update Product | |
| **Preconditions** | Product page should be rendered | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Press Update button |
| 2 | S: Redirects to Update Product Webpage with input fields to alter details of product |
| 3 | U: Edit the input fields |
| 4 | U: Press Update Changes button |
| 5 | S: Redirects to Product Page which shows the list of added categories |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 1a | S: Throw Error Message: Whitelabel Error Page |
| 3a | Input field is empty.  S: show a dialogue box showing fill the empty fields and goes to step 3. |

1. **Test Case for Admin Categories Functionality (Update Product in Products):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Test Update Functionality | Clicks “Update” Button | Redirects to Update Product Page | Redirects to Error Page: Whitelabel Error Page |

1. **Use Case for Admin Categories Functionality (Delete Product in Products):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Delete Product | |
| **Preconditions** | Product page should be rendered | |
| **Typical Case of Events/ Main Success Scenario (U: Admin, S: System)** | **Step** | **Action** |
| 1 | U: Press Delete button |
| 2 | S: Redirects to Product Page which shows the list of added products after deleting the product |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| 2a | S: Shows error message and cannot delete |

1. **Test Case for Admin Categories Functionality (Delete Product in Products):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Test Delete Functionality | Clicks “Delete” button | Deletes Product, Displays Products List after deletion | Deletes Product, Displays Products List after deletion |

**Customers Category**

* Page not interactive.
* Lists down all the registered users.
* Delete button not available.
* Data not editable.
* CRUD operations cannot be performed.

1. **Use Case for Admin Categories Functionality (Customers Category):**

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | |
| **Use Case Name** | Customers Category | |
| **Preconditions** | Category page should be rendered | |
| **Typical Case of Events/ Main Success Scenario** | **Step** | **Action** |
| 1 | S: List of Customers Registered Shown |
| **Alternate Case of Events/ Extensions/ Alternate Courses** | **Step** | **Action** |
| - | - |

1. **Test Case for Admin Categories Functionality (Customers Category):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Case Type | Purpose | Input(s) | Expected Output | Actual Output |
| 1 | Valid | Test Category | - | - | - |

**MISSING FUNCTIONALITIES:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Where should it be?** | **Functionality/ Purpose** | **Reference Website Link & Image** |
| Drop Down Menu | Top of Page | Display all categories and options available on the website | <https://www.elo.shopping/> |
| Logo / Name of Brand (instead of Welcome on home page) | Top left corner of every page | Redirects back to home page | <https://www.elo.shopping/> |
| Search Bar | Top right corner/ top center of every page | Search any product on the website | <https://www.amazon.com/ref=nav_logo> |
| Home Page | Preview before login on website | View website before logging in | <https://www.elo.shopping/> |
| Shopping Cart | Opens when click cart icon on top right corner of page | View all products you have added to cart to purchase | <https://www.elo.shopping/>  à |
| Support/ Help Center | Opens when click button on bottom left corner of the page | To ask for any help or clarify any ambiguity or query regarding a product or service offered by website. | <https://www.elo.shopping/>    à |
| Account/ Profile | Opens when click profile option on top right corner of webpage | Displays details of account and order history. | <https://www.elo.shopping/account> |

**2-Pairwise Integration:**

|  |
| --- |
| Total Test Sessions = No. of edges  Test Sessions = 20 |

**Test Sessions:**

|  |  |
| --- | --- |
| 2T1 | Cart DAO,Cart |
| 2T2 | User Controller, User |
| 2T3 | Cart,User |
| 2T4 | Product Service ,Product DAO |
| 2T5 | User Controller,User Service |
| 2T6 | User Controller,Product Service |
| 2T7 | User Controller,Product |
| 2T8 | Cart Service,Cart |
| 2T9 | Cart Service,Cart DAO |
| 2T10 | User DAO,User |
| 2T11 | Product Service,Product DAO |
| 2T12 | User Service,User DAO |
| 2T13 | Product,Category |
| 2T14 | CategoryService,CategoryDAO |
| 2T15 | Admin Controller,Product Service |
| 2T16 | Admin Controller,User Service |
| 2T17 | Admin Controller,user |
| 2T18 | Admin Controller,Product |
| 2T19 | Admin Controller,Category Service |
| 2T20 | Admin Controller,Category |

**Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T20.1 | Checking the Category id is get | Checking the Category id is get | Get correctly | Get correctly |
| 2T20.2 | Checking the Category name is get | Checking the Category name is get | Get correctly | Get correctly |
| 2T20.3 | Checking the Category name is set | Checking the Category name is set | Set correctly | Set correctly |
| 2T20.4 | Checking the Category id is set | Checking the Category id is set | Set correctly | Set correctly |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T15.1 | Admin Accesses Product Management Page | ModelAndView with view name "products" and a list of products added to the model.. | ModelAndView with view name "products" and a list of products added to the model.. | Admin successfully access page for product management. |
| 2T15.2 | To verify that the admin can access the page for adding a new product | ModelAndView with view name "productsAdd" and a list of categories added to the model. | ModelAndView with view name "productsAdd" and a list of categories added to the model. | Admin successfully access page for adding new product. |
| 2T15.3 | To verify the addition of a new product by the admin. | Redirect to the product management page with the newly added product in the list. | Redirect to the product management page with the newly added product in the list. | Admin successfully add new product. |
| 2T15.4 | To verify that the admin can successfully update a product. | To verify that the admin can successfully update a product. | The admin can successfully update a product. | Admin is not able to update new product. |
| 2T15.5 | To verify that the admin can successfully remove a product. | To verify that the admin can successfully remove a product. | Redirect to the product management page with the product removed from the list. | Admin is not able to remove a product. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T19.1 | To verify that the admin can access the category management page. | ModelAndView with view name "categories" and a list of categories added to the model. | ModelAndView with view name "categories" and a list of categories added to the model. | Admin successfully access page for category management. |
| 2T19.2 | To verify the addition of a new category by the admin. | Redirect to the category management page with the newly added category in the list. | Redirect to the category management page with the newly added category in the list. | Admin is not able to add new category. |
| 2T19.3 | To verify that the admin can successfully delete a category. | Redirect to the category management page with the category deleted from the list. | Redirect to the category management page with the category deleted from the list. | Admin is not able to delete a category. |
| 2T19.4 | To verify that the admin can successfully update a category name. | Redirect to the category management page with the updated category name in the list. | Redirect to the category management page with the updated category name in the list. | Admin successfully update a category. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T18.1 | Checking the Product Id is set | Checking the Product Id is set | Set correctly | Set correctly |
| 2T18.2 | Checking the Product Name is set | Checking the Product Name is set | Set correctly | Set correctly |
| 2T18.3 | Checking the Product Category is set | Checking the Product Category is set | Set correctly | Set correctly |
| 2T18.4 | Checking the Product Description is set | Checking the Product Description is set | Set correctly | Set correctly |
| 2T18.5 | Checking the Product Price is set | Checking the Product Price is set | Set correctly | Set correctly |
| 2T18.6 | Checking the Product Image is set | Checking the Product Image is set | Set correctly | Set correctly |
| 2T18.7 | Checking the Product Weight is set | Checking the Product Weight is set | Set correctly | Set correctly |
| 2T18.8 | Checking the Product Quantity is set | Checking the Product Quantity is set | Set correctly | Set correctly |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T17.1 | Checking if Admin role is correct retrieve | Checking if Admin role is correct retrieve | Verify Admin and show Admin dashboard | Verify Admin and show Admin dashboard |
| 2T17.2 | Checking if System is robust on Other role [except Admin] | Checking if System is robust on Other role [except Admin] | System show message of invalid credentials | System does not show message. |
| 2T17.3 | Checking if Entered credentials are invalid | Checking if Entered credentials are invalid | System show proper message when user not found. | Show Exception. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T16.1 | To verify that the admin can log in successfully. | ModelAndView with view name "adminHome" and the admin user object in the model. | ModelAndView with view name "adminHome" and the admin user object in the model. | Admin login successfully |
| 2T16.2 | To verify that the admin can access the user management page. | ModelAndView with view name "displayCustomers" and a list of users in the model. | ModelAndView with view name "displayCustomers" and a list of users in the model. | Admin successfully access the user management page. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T14.1 | To verify that a new category can be successfully added. | Newly added category with the correct name. | Newly added category with the correct name. | Newly added category with the correct name as expected. |
| 2T14.2 | To verify that the list of categories can be retrieved. | List of existing categories. | List of existing categories. | List of existing categories as expected. |
| 2T14.3 | To verify that an existing category can be successfully deleted. | To verify that an existing category can be successfully deleted. | true indicating successful deletion. | true indicating successful deletion as expected. |
| 2T14.4 | To verify that an existing category can be successfully updated. | To verify that an existing category can be successfully updated. | Updated category with the correct name. | Updated category with the correct name as expected. |
| 2T14.5 | To verify that a category can be retrieved by its ID. | To verify that a category can be retrieved by its ID. | Category with the correct ID. | Category with the correct ID as expected.. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T13.1 | To verify that a category can be set for a product. | The product's category should be the assigned category. | The product's category should be the assigned category. | Category can be set for a product as expected. |
| 2T13.2 | To verify that a product can be set for a category. | The category's associated product should be the assigned product. | The category's associated product should be the assigned product. | Product can be set for a category as expected. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T12.1 | To verify that the getAllUsers method retrieves a list of users. | List of users retrieved from the userDao. | List of users retrieved from the userDao. | List of users retrieved from the userDao. |
| 2T12.2 | To verify that the SaveUser method save User | Save User successfully | Save User successfully | Save User successfully |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T11.1 | To verify that the getProducts method retrieves a list of products. | List of products should not be null, and it should contain the expected number of products. | List of products should not be null, and it should contain the expected number of products. | Contain List of Products as expected. |
| 2T11.2 | To verify that the addProduct method adds a product successfully. | The added product should be returned, and it should be present in the list of products retrieved by getProducts | The added product should be returned, and it should be present in the list of products retrieved by getProducts | The added product is returned, and it is present in the list of products retrieved by getProducts so works as expected. |
| 2T11.3 | To verify that the getProduct method retrieves a product by its ID. | The returned product should not be null, and it should have the expected ID. | The returned product should not be null, and it should have the expected ID. | The returned product is not null, and it have the expected ID as expected. |
| 2T11.4 | To verify that the updateProduct method updates a product successfully. | The updated product should be returned, and it should have the expected modifications. | The updated product should be returned, and it should have the expected modifications. | The updated product is returned, and it have the expected modification. |
| 2T11.5 | To verify that the deleteProduct method deletes a product successfully. | The method should return true, indicating that the product was successfully deleted. | The method should return true, indicating that the product was successfully deleted. | The method return true, indicating that the product was successfully deleted |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T10.1 | To verify that the getAllUser method retrieves a list of users. | List of users should not be null, and it should contain the expected number of users. | List of users should not be null, and it should contain the expected number of users. | List of users is not null, and it contain the expected number of users. |
| 2T10.2 | To verify that the saveUser method adds a user successfully. | The added user should be returned, and it should be present in the list of users retrieved by getAllUser | The added user should be returned, and it should be present in the list of users retrieved by getAllUser | The added user is returned, and it e present in the list of users retrieved by getAllUser |
| 2T10.3 | To verify that the getUser method retrieves a user by their username and password. | The returned user should not be null, and it should have the expected username and password. | The returned user should not be null, and it should have the expected username and password. | The returned user is not null, and it have the expected username and password. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T9.1 | To verify that the addCart method adds a cart successfully. | The added cart should be returned. | The added cart should be returned. | The added cart is returned. |
| 2T9.2 | To verify that the updateCart method updates a cart successfully. | The cart should be updated with the expected modifications. | The cart should be updated with the expected modifications. | The cart is updated with the expected modifications. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T8.1 | To verify getCarts method returns a list of Cart objects. | Method returns a list of Cart objects. | Method returns a list of Cart objects. | Work successfully. |
| 2T8.2 | The addCart method takes a Cart object as a parameter and adds it to the data store | takes a Cart object as a parameter and adds it to the data store | takes a Cart object as a parameter and adds it to the data store | Work successfully. |
| 2T8.3 | The updateCart method takes a Cart object as a parameter and updates its details. | akes a Cart object as a parameter and updates its details. | akes a Cart object as a parameter and updates its details. | Work successfully. |
| 2T8.4 | The deleteCart method takes a Cart object as a parameter and deletes it from the data store. | method takes a Cart object as a parameter and deletes it from the data store. | method takes a Cart object as a parameter and deletes it from the data store. | Work successfully. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T7.5 | To verify that the getproduct method retrieves a list of products. |  | method retrieves a list of products. | method retrieves a list of products. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T6.1 | To verify that a user can successfully log in, and the associated products are retrieved. | To verify that a user can successfully log in, and the associated products are retrieved. | The user should be successfully logged in, and the products associated with the user should be retrieved and displayed on the "index" page. | Functionality is incomplete for adding to Cart. |
| 2T6.2 | To verify the getProduct() method is working accurately | To verify the getProduct() method is working accurately | Retrieve list of products | Work successfully |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T5.1 | To verify that a user can successfully log in with correct credentials. | The user should be successfully logged in, and the ModelAndView should be directed to the "index" page. | The user should be successfully logged in, and the ModelAndView should be directed to the "index" page. | Work successfully. |
| 2T5.2 | To verify that a user cannot log in with incorrect credentials. | The user not able to login with incorrect credentials | The user not able to login with incorrect credentials | The user not able to login with incorrect credentials |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T4.1 | To verify productDao get all products | Get all products. | Get all products. | Works successfully. |
| 2T4.2 | To verify productDao add product. | Add product. | Add product. | Works successfully. |
| 2T4.3 | To verify get a Product by id. | Get product by id. | Get product by id. | Works successfully. |
| 2T4.4 | To verify delete Product | Delete Product | Delete Product | Unable to delete. |
| 2T4.5 | To verify update Product | Update Product | Update Product | Unable to update. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T3.1 | To verify that a new cart can be created and associated with a user. | The user should be successfully associated with the cart. | The user should be successfully associated with the cart. | Does not work. |
| 2T3.2 | To verify that a product can be removed from the cart associated with a user. | The product should be successfully removed from the user's cart | The product should be successfully removed from the user's cart | Does not work |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T2.1 | To verify that a user can successfully log out. | User logout his account. | User logout his account. | Unable to logout. |
| 2T2.2 | To verify User can view his profile. | User view his profile | User view his profile | Unable to view profile. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TID | Purpose | Description/Code | Expected Result | Actual Result |
| 2T1.1 | To verify that the list of carts can be retrieved from the database. | The list of carts should be successfully retrieved from the database, and the returned list should not be empty. | The list of carts should be successfully retrieved from the database, and the returned list should not be empty. | Successfully Work. |
| 2T1.2 | To verify that an existing cart can be successfully updated in the database. | The existing cart should be successfully updated in the database, and the returned Cart object should match the updated cart. | The existing cart should be successfully updated in the database, and the returned Cart object should match the updated cart. | Successfully Work. |
| 2T1.3 | To verify that an existing cart can be successfully deleted from the database | The existing cart should be successfully deleted from the database, and the returned Cart object from deleteCart should be null. | The existing cart should be successfully deleted from the database, and the returned Cart object from deleteCart should be null. | Successfully Work. |
| 2T1.4 | To verify that a new cart can be successfully added to the database. | he new cart should be successfully added to the database, and the returned Cart object should match the added cart. | he new cart should be successfully added to the database, and the returned Cart object should match the added cart. | Successfully Work |