|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * Test Case | * Purpose | * Input Data | * Expected Output |  |
| * Test Case 1 | * Check a new category can be added successfully. | * Category name -"shopping" | * New category with the name "shopping" is added. |  |
| * Test Case 2 | * Check duplicate category name | * Category name - "shopping" * ( existing category) | * Error message: category name already exists. |  |
| * Test Case 3 | * Check invalid characters in category name | * Category name - "13#$^&" | * Error message: category name contains invalid characters. |  |

**Category table**

### Customer table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * Test Case | * Purpose | * Input Data | * Expected Output |  |
| * Test Case 1 | * Check a new customer can be added successfully. | * Customer valid details | * New customer is added with given details. |  |
| * Test Case 2 | * Check duplicate email | * Email - "ok@gmail.com" (existing email) | * Error message: email is already registered. |  |
| * Test Case 3 | * Check valid role for customers | * Invalid role | * Error message: role is invalid. |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * Test Case | * Purpose | * Input Data | * Expected Output |  |
| * Test Case 1 | * Check a new product can be added successfully. | * Product details - Description, Image etc. | * New product is added with given details. |  |
| * Test Case 2 | * Check only valid category IDs are accepted for products. | * Invalid ID * (sw345) | * Error message: ID is invalid. |  |
| * Test Case 3 | * Check that products cannot have a negative price. | * Negative price * (-100) | * Error message: price cannot be negative. |  |
| * Test Case 4 | * Check that only valid image URLs are accepted for products. | * Invalid URL for image | * Error message: image URL is invalid. |  |

### Product table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * Test Case | * Purpose | * Input Data | * Expected Output |  |
| * Test Case 1 | * Check a product can be successfully added to the cart. | * New product to be added | * The product is added to the cart successfully. |  |
| * Test Case 2 | * Check a product can be successfully removed from the cart. | * Product to be removed from the cart | * The product is removed from the cart successfully. |  |
| * Test Case 3 | * Check a customer can be set for the cart. | * Customer to be set for the cart | * The customer for the cart is set successfully. |  |
| * Test Case 4 | * Check the cart ID can be retrieved. | * Cart ID to be retrieved | * The cart ID is successfully retrieved. |  |

### Cart class

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Add a valid product to an empty cart | * Valid product i | * Product is added to the cart successfully |
| * Test Case 2 | * Add a valid product to a non-empty cart | * Valid product | * Product is added to the existing products in the cart |
| * Test Case 3 | * Delete a product from a non-empty cart | * Valid product existing | * Product is removed from the cart successfully |
| * Test Case 4 | * Delete the last product from the cart | * Valid last product in list | * Last product is removed from the cart |
| * Test Case 5 | * Set a valid customer for the cart | * Valid customer | * Customer for the cart is set successfully |
| * Test Case 6 | * Get ID for a newly created cart | * --------- | * Cart ID is not null |
| * Test Case 7 | * Set a valid list of products for the cart | * Valid list of product | * List of products in the cart is set successfully |
| * Test Case 8 | * Set an empty list of products for the cart | * Empty list | * Cart has no products |

**Product class**

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Create a valid product | * Valid product with all attributes | * Product is created successfully |
| * Test Case 2 | * Update product name | * Valid product with a new name | * Product name is updated successfully |
| * Test Case 3 | * Update product category | * Valid product with a new category | * Product category is updated successfully |
| * Test Case 4 | * Update product quantity | * Valid product with a new quantity | * Product quantity is updated successfully |
| * Test Case 5 | * Update product price | * Valid product with a new price | * Product price is updated successfully |
| * Test Case 6 | * Update product weight | * Valid product with a new weight | * Product weight is updated successfully |
| * Test Case 7 | * Update product description | * Valid product with a new description | * Product description is updated successfully |
| * Test Case 8 | * Set null name for the product | * Valid product with a null name | * Error message: Attempt to set a null name for the product |
| * Test Case 9 | * Set null category for the product | * Valid product with a null category | * Error message: Attempt to set a null category for the product |
| * Test Case 10 | * Set negative quantity for the product | * Valid product with a negative quantity | * Error message: Attempt to set a negative quantity for the product |

**User class**

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Create a valid user | * User’s all valid attributes | * User is created successfully |
| * Test Case 2 | * Update username | * New valid username | * Username is updated successfully |
| * Test Case 3 | * Update email | * New valid email | * Email is updated successfully |
| * Test Case 4 | * Update password | * New valid password | * Password is updated successfully |
| * Test Case 5 | * Update role | * New valid role | * Role is updated successfully |
| * Test Case 6 | * Update address | * New valid address | * Address is updated successfully |
| * Test Case 7 | * Set null username for the user | * User’s valid null username | * Error message: Attempt to set a null username for the user |
| * Test Case 8 | * Set null email for the user | * User’s valid null email | * Error message: Attempt to set a null email for the user |
| * Test Case 9 | * Set null password for the user | * User’s valid null password | * Error message: Attempt to set a null password for the user |
| * Test Case 10 | * Set null role for the user | * User’s valid null role | * Error message: Attempt to set a null role for the user |
| * Test Case 11 | * Set null address for the user | * User’s valid null address | * Error message: Attempt to set a null address for the user |
| * Test Case 12 | * Set empty username for the user | * User’s valid empty username | * Username is set successfully with an empty value |
| * Test Case 13 | * Set empty email for the user | * User’s valid empty email | * Email is set successfully with an empty value |
| * Test Case 14 | * Set empty password for the user | * User’s valid empty password | * Password is set successfully with an empty value |
| * Test Case 15 | * Set empty role for the user | * User’s valid empty role | * Role is set successfully with an empty value |
| * Test Case 16 | * Set empty address for the user | * User’s valid empty address | * Address is set successfully with an empty value |
| * Test Case 17 | * Create a user with a null username | * User’s null username | * Error message: Attempt to create a user with a null username |
| * Test Case 18 | * Create a user with an empty username | * User’s empty username | * Error message: Attempt to create a user with an empty username |
| * Test Case 19 | * Create a user with a null email | * User’s null email | * Error message: Attempt to create a user with a null email |
| * Test Case 20 | * Create a user with an empty email | * User’s empty email | * Error message: Attempt to create a user with an empty email |

### AdminController

#### adminlogin(String username, String pass)

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check Valid admin login credentials | * Username="hello", Password="hello123" | * Successful login |
| * Test Case 2 | * Check Valid user login credentials | * Username="world3344", Password="helloworld11" | * Successful login |
| * Test Case 3 | * Check Invalid (empty) username | * Username="", Password="hello123" | * Unsuccessful login and an error message |
| * Test Case 4 | * Check Invalid (empty) password | * Username="world3344", Password="" | * Unsuccessful login and an error message |

#### getcategory()

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check Admin logged in | * ------- | * View name is "categories" with a list of categories |
| * Test Case 3 | * Check Admin not logged in | * ------- | * Redirected to "adminlogin" |

#### addCategory(String category\_name)

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check Valid category name | * Category name="shopping" | * Category is added successfully |
| * Test Case 2 | * Check Invalid (empty) category name | * Category name="" | * Category is not added and an error message |
|  |  |  |  |

#### removeCategoryDb(int id)

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check Existing category id to remove | * Category ID=1 | * Category with ID=1 is removed |
| * Test Case 2 | * Check Non-existing category id | * Category ID=1000 | * No change in the database and an error message |

#### updateCategory(int id, String categoryname)

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 2 | * Check Valid category id and name | * Category ID=1, Category name="UpdatedCategory" | * Category with ID=1 is updated |
| * Test Case 2 | * Check Invalid (empty) category name | * Category ID=1, Category name="" | * Category is not updated and error message |
| * Test Case 3 | * Check Non-existing category id | * Category ID=1000, Category name="UpdatedCategory" | * No change in the database and error message |

### User controller

#### userlogin(String username, String pass, Model model, HttpServletResponse res)

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check valid credentials | * Username="hello111", Password="check123" | * Successful login |
| * Test Case 2 | * Check Invalid (empty) username | * Username="", Password="check123" | * Unsuccessful login and error message |
| * Test Case 3 | * Check Invalid (empty) password | * Username="hello111", Password="" | * Unsuccessful login and error message |

#### getproduct()

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check products available | * ------- | * list of products |
| * Test Case 2 | * Check no products available | * ------- | * Message: "No products are available." |

#### newUseRegister(User user)

|  |  |  |  |
| --- | --- | --- | --- |
| * Test ID | * Purpose | * Input | * Expected Output |
| * Test Case 1 | * Check valid user details | * valid details | * User is registered successfully |
| * Test Case 2 | * Check empty email | * An empty email field | * User is not registered and an error message |