Test Cases- Online Shopping Store

Black-Box Test Cases

These test cases are defined based on the available documentation and execution of the program. The code was not inspected.

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation (“pass” or failure description)
1	Successful Registration	CSV file is initially empty	Launch the application. Fill in unique username, valid password, phone number, and email. Click "Register".	User details are written to CSV. Redirects to home page.	Pass
2	Empty Fields	CSV file is initially empty	Launch the application. Leave any field (username, password, phone number, email) empty. Click "Register".	A message "Please fill in all fields." should appear.	Pass
3	Invalid Email Format	CSV file is initially empty	Launch the application. Enter an invalid email format. Click "Register".	A message " Please Enter Valid Email " should appear.	Pass
4	Email Already Registered	CSV file contains registered email	Launch the application. Enter an email that is already in the CSV file. Click "Register".	Alert shows "E-Mail is already registered." message.	Pass
5	Phone Number Already Registered	CSV file contains registered phone number	Launch the application. Enter a phone number that is already in the CSV file. Click "Register".	Alert shows "Phone number is already registered." message.	Pass
6	Both Email and Phone Number Registered	CSV file contains both registered email and phone number	Launch the application Enter both email and phone number that are already in the CSV file.	Alert shows "E-Mail and Phone number are already	Pass

			Click "Register".	registered." message.	
7	Back Button Functionality	Application launched	Click on the "Back" button.	The application should redirect to the home page.	Pass
8	CSV File Writing	None	Register a user through the UI with valid data.	Data is correctly written to the CSV file.	Pass
9	Successful Password Update	CSV file contains user data	Launch the application. Enter a valid email, matching new and confirm passwords, then click "Submit".	Password for the corresponding email should be updated successfully. A success message appears.	Pass
10	Passwords Do Not Match	CSV file contains user data	Launch the application. Enter a valid email, mismatching new and confirm passwords, then click "Submit".	A message "Passwords do not match!" should appear.	Pass
11	Invalid Email	CSV file contains user data	Launch the application. Enter an email does not present in the CSV file, valid new and confirm passwords, then click "Submit".	A message " Please Enter Valid Email " should appear.	Fail Its displaying “Email not Found!”
12	Back Button Functionality	Application launched	Click on the "Back" button.	The application should redirect to the home page.	Pass
13	Empty Email Field	Application launched	Launch the application. Leave the email field empty, enter valid new and confirm passwords, then click "Submit".	A message "Please Enter Email" should appear.	Fail Its displaying “Email not Found!”

14	Empty New Password Field	Application launched	Launch the application. Enter a valid email, leave the new password field empty, enter the same value in confirm password, then click "Submit".	A message “Please Enter New Password" should appear.	Fail Its displaying “Passwords do not match!"
15	Empty Confirm Password Field	Application launched	Launch the application. Enter a valid email, enter a new password, leave the confirm password field empty, then click "Submit".	A message “Please Re-Enter Password” should appear.	Fail Its displaying “Passwords do not match!"
16	Valid Login Credentials	CSV file exists with valid user credentials	Launch the application. Enter a valid mail ID and password. Click "Login".	Redirects to the shopping cart page.	Pass
17	Invalid Login Credentials	CSV file exists with valid user credentials	Launch the application. Enter an invalid mail ID or password. Click "Login".	Displays "User Not Found, Enter valid credentials" message.	Pass
18	Register Button Navigation	Application launched	Launch the application. Click on "Register" button.	Redirects to the registration page.	Pass
19	Forgot Password Button Navigation	Application launched	Launch the application. Click on "Forgot Password" button.	Redirects to the forgot password page.	Pass
20	Navigate to Shopping Cart	Valid login credentials	Successfully logged in with valid credentials.	Redirects to the shopping cart page.	Pass
21	Empty Email Id Field	CSV file exists with valid user credentials	Launch the application. Leave email id field empty and enter valid password. Click "Login".	A message " Please Enter Valid Email " should appear.	Fail A message “User Not Found, enter valid credentials" appears
22	Empty Password Field	CSV file exists with valid user credentials	Launch the application. Enter valid email id and leave password field empty. Click "Login".	A message "Please Enter Password " should appear.	Fail A message “User Not Found, enter valid credentials" appears
23	Product Search	Application launched	Launch and Login to the application. Enter Product name which is available	Console should display Herbal Related products.	Pass

			Example: Enter "Herbal" in the search field. Click "Search".		
24	Empty Search Field	Application launched	Launch and Login to the application. Leave the search field empty. Click "Search".	No product should be displayed in the console and A message “Enter Product Name” appear	Fail Its displaying “All Products”
25	Search with Non-existing Product	Application launched	Launch and Login to the application. Enter the product name which is not available in the search field. Click "Search".	No product should be displayed in the console and A message “No Product Available” appear	Fail No message appears
26	Filter by Medicine Types	Application launched	Launch and Login to the application. Select "Ayurveda" checkbox.	Only products categorized under "Ayurveda" are displayed.	Pass
27	Filter by Multiple Medicine Types	Application launched	Launch and Login to the application. Select "Ayurveda" and "Homeopathy" checkboxes.	Products categorized under "Ayurveda" and "Homeopathy" are displayed.	Pass
28	Select and Deselect Medicine Types	Application launched	Launch and Login to the application. Select "Ayurveda" checkbox, then deselect it.	Initially, only "Ayurveda" products are shown; after deselecting, all products are displayed.	Pass
29	Logout Button Functionality	Application launched	Launch and Login to the application. Click on the "Logout" button.	Application redirects to the login page.	Pass
30	Product details	Application launched	Launch and Login to the application. Select the product available in product section and check for product details	Product details like “Name, Description and Price” should appear	Pass

31	Adding product to Shopping cart	Application launched	Launch and Login to the application. Select the desired product and add to shopping cart	Product should be added to shopping car	Fail Feature not implemented
32	Payment Option and orders page	Application launched	Launch and Login to the application. Select the desired product and add to shopping cart and select payment and checkout	Payment done and checkout to orders page	Fail Feature not implemented

White-Box Test Cases

These additional test cases were defined during inspection of the code.

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation (“pass” or failure description)
1	UI Components Existence	Application running	1. Launch the application 2. Initialize UI components for testing.	All UI components (usernameInput, passwordInput, loginButton, registerButton, forgotPasswordButton, promptLabel) should not be null.	Pass
2	Successful Login	CSV file	1. Load data tester123@example.com with password 12345 into CSV file using writeCSV method. 2. Enter tester123@example.com in usernameInput. 3. Enter 12345 in passwordInput. 4. Click on loginButton.	The promptLabel should be empty, indicating successful login.	Pass
3	Empty Username and Password	Application running	1. Enter an empty string in usernameInput. 2. Enter an empty string in passwordInput. 3. Click on loginButton.	The promptLabel should display "User Not Found, Enter valid credentials".	Pass

4	Login Failure (Invalid Credentials)	Application running	1. Enter invalid@example.com in usernameInput. 2. Enter wrong password in passwordInput. 3. Click on loginButton.	The promptLabel should display "User Not Found, Enter valid credentials".	Pass
5	Redirect to Register Page	Application running	1. Click on registerButton.	Application should redirect to the register page. (Verify by console output)	Pass
6	Redirect to Forgot Password Page	Application running	1. Click on forgotPasswordButton.	Application should redirect to the forgot password page. (Verify by console output)	Pass
7	Update Password Successfully	CSV file	1. Register tester@example.com with a password, username and mobile number using writeCSV method. 2. Call updatePassword method with tester@example.com and newPassword12. 3. Create a separate method isPasswordUpdated to check if the password is updated in the csv file. 4. Test case passes if it's true	Password should be updated successfully in the CSV file.	Fail
8	Email Input Field	Application running	1. Lookup emailInput TextField. 2. Write test@example.com into the emailInput field. 3. Verify the input.	The text in emailInput should be test@example.com.	Pass
9	New Password Input Field	Application running	1. Lookup newPasswordInput PasswordField. Write newPassword123 into the newPasswordInput field. 3. Verify the input.	The text in newPasswordInput should be newPassword123.	Pass
10	Confirm Password Input Field	Application running	1. Lookup confirmPasswordInput PasswordField. 2. Write newPassword123 into the	The text in confirmPasswordInput should be newPassword123.	Pass

			confirmPasswordInput field. 3. Verify the input.		
11	Load Products Successfully	Product list is pre-defined	1. Call ProductData.loadProducts(). 2. Verify the products list is not null. 3. Verify the list contains 10 products. 4. Verify attributes of each product.	Products list should be loaded successfully with correct attributes.	Pass
12	Products List Not Null	Product list is pre-defined	1. Call ProductData.loadProducts(). 2. Verify the products list is not null.	Products list should not be null.	Pass
13	Products List Size	Product list is pre-defined	1. Call ProductData.loadProducts(). 2. Verify the products list contains 10 products.	Products list should contain exactly 10 products.	Pass
14	Product Category Counts	Product list is pre-defined	1. Call ProductData.loadProducts(). 2. Count the number of products in each category. 3. Verify the counts.	Each category should contain 2 products.	Pass
15	No duplicates	Product list is pre-defined	1. Call ProductData.loadProducts(). 2. Verify that each product in the list is unique.	There should be no duplicates	Pass
16	Verify UI Components Existence	The JavaFX application is running	1. Check if searchField exists. 2. Check if searchButton exists. 3. Check if consoleLabel exists.	All the UI components (searchField, searchButton, consoleLabel) should exist.	Pass
17	Search Button Action	The JavaFX application is running	1. Enter "Test Query" in the search field. 2. Click the search button.	The console label should display "Search Query: Test Query".	Pass
18	Empty Search Input	The JavaFX application is running	1. Click the search button without entering anything in the search field.	The console label should remain empty or unchanged.	Fail
19	Special Characters in Search Input	The JavaFX application is running	1. Enter "Special #@$ Characters" in the search field. 2. Click the search button.	The console label should display "Search Query: Special #@$ Characters".	Pass
20	Multiple Searches	The JavaFX application is running	1. Enter "herbal" in the search field. 2. Click the search button. 3. Enter "remedy" in the search field. 4. Click the search button.	The console label should display "Search Query: herbal" after the first search	Pass

				and "Search Query: remedy" after the second search.	
21	Verify searching for products	The JavaFX application is running	1. Enter "herbal" in the search field. 2. Click the search button.	The product accordion should display 4 product panes.	Pass
22	Filter products by a single category	The JavaFX application is running	1. Click on the Ayurveda checkbox. 2. Click the Ayurveda checkbox again. 3. Click on the Naturopathy checkbox. 4. Click the Naturopathy checkbox again.	The product accordion should display 2 panes for Ayurveda, 2 panes for Naturopathy, and no panes when both are unchecked.	Pass
23	Filter products by multiple categories	The JavaFX application is running	1. Click on the Ayurveda checkbox. 2. Click on the Homeopathy checkbox.	The product accordion should display 4 product panes.	Pass
24	Clear search input	The JavaFX application is running	1. Enter "herbal" in the search field. 2. Click the search button. 3. Erase the search input. 4. Click the search button again.	The product accordion should display 4 panes after the first search and 0 panes after clearing the search input.	Fail
25	Logout functionality	The JavaFX application is running	Click on the logout button.	The login fields (username and password) should be present after logging out.	Pass
26	Verify email validation	JavaFX application is running	Invoke isValidEmail with various email inputs.	Correct Boolean results for valid and invalid emails.	Pass
27	Verify email registration	JavaFX application is running	Invoke isEmailRegistered with various email inputs.	Correct Boolean results based on existing emails in the CSV file.	Pass
28	Verify phone number registration logic	JavaFX application is running	Invoke isPhoneNumberRegistered with various phone number inputs.	Correct Boolean results based on existing phone numbers in the CSV file.	Pass
29	Verify CSV writing logic	JavaFX application is running	Invoke writeCSV to add a new user and check CSV file content.	CSV file contains the new user data.	Pass
30	Register a user with valid input	JavaFX application is running	Simulate user input for valid registration details and click register.	User data is written to the CSV file.	Pass
31	Register a user with an existing email	JavaFX application is running	Creates a user and then simulates a user input with an email that was created and click register.	Prompt label shows "E-Mail is already registered.".	Fail

32	Register a user with an existing phone number	JavaFX application is running	Creates a user and simulates user input with a phone number that was created and click register.	Prompt label shows "Phonenumber is already registered.".	Fail
33	Register a user with both existing number and email	JavaFX application is running	Creates a user and simulates user input with both email and phone number that was created and click register.	Prompt label shows "E-Mail and Phonenumber are already registered.".	Fail
34	Register a user with empty fields	JavaFX application is running	Simulate user input with empty fields and click register.	Prompt label shows "Please fill in all fields."	Pass
35	Register a user with invalid email	JavaFX application is running	Simulate user input with an invalid email and click register.	Prompt label shows "Please enter a valid email address.".	Pass
36	Constructor Initialization	A product object is created	1. Create a product object with specified parameters. 2. Verify the initialization of attributes.	The product attributes should be initialized correctly.	Pass
37	Get Name	A product object is created	Call getName() method.	The name should be "Herbal Tea".	Pass
38	Set Name	A product object is created	1. Call setName("Green Tea"). 2. Call getName().	The name should be "Green Tea".	Pass
39	Get Category	A product object is created	Call getCategory() method.	The category should be "Ayurveda".	Pass
40	Set Category	A product object is created	1. Call setCategory("Tea"). 2. Call getCategory().	The category should be "Tea".	Pass
41	Get Image Path	A product object is created	Call getImagePath() method.	The image path should be "/Images/herbal_tea.jpg".	Pass
42	Set Image Path	A product object is created	1. Call setImagePath("/Images/green_tea.jpg"). 2. Call getImagePath().	The image path should be "/Images/green_tea.jpg".	Pass
43	Get Description	A product object is created	Call getDescription() method.	The description should be "A soothing herbal tea blend.".	Pass
44	Set Description	A product object is created	1. Call setDescription("Refreshing green tea."). 2. Call getDescription().	The description should be "Refreshing green tea.".	Pass
45	Get Price	A product object is created	Call getPrice() method.	The price should be 5.99.	Pass

46	Set Price	A product object is created	1. Call setPrice(4.99). 2. Call getPrice().	The price should be 4.99.	Pass
47	Set Negative Price	A product object is created	1. Call setPrice(-5.0). 2. Call getPrice().	The price should be -5.0.	Pass
48	Set Empty Name	A product object is created	1. Call setName(""). 2. Call getName().	The name should be an empty string.	Pass
49	Set Empty Category	A product object is created	1. Call setCategory(""). 2. Call getCategory().	The category should be an empty string.	Pass
50	Set Empty Description	A product object is created	1. Call setDescription(""). 2. Call getDescription().	The description should be an empty string.	Pass
51	Set Null Image Path	A product object is created	1. Call setImagePath(null). 2. Call getImagePath().	The image path should be null.	Pass
52	Set Minimum Price	A product object is created	1. Call setPrice(Double.MIN_VALUE). 2. Call getPrice().	The price should be Double.MIN_VALUE.	Pass
53	Set Maximum Price	A product object is created	1. Call setPrice(Double.MAX_VALUE). 2. Call getPrice().	The price should be Double.MAX_VALUE.	Pass
54	Test getters return null initially	User object is initialized	Call getters (getName, getPassword, getPhoneNumber, getMail).	All getters should return null.	Pass
55	Test setters and getters	User object is initialized	1.Set fields (setName, setPassword, setPhoneNumber, setMail). 2.Call corresponding getters.	Getters should return the values set by the setters.	Pass
56	Test toString method	User object is initialized and fields are set	1.Set fields (setName, setPassword, setPhoneNumber, setMail). 2.Call toString.	toString should return name: John Doe, password:password123, phone:123-456-7890, mail:john.doe@example.com.	Pass
57	Test setters with null values	User object is initialized	1.Set fields to null (setName, setPassword, setPhoneNumber, setMail). 2.Call getters.	Getters should return null for all fields.	Pass
58	Test toString with null values	User object is initialized and fields are set to null	1.Set fields to null (setName, setPassword, setPhoneNumber, setMail). 2.Call toString.	toString should return name: null, password:null, phone:null, mail:null.	Pass

59	Test setters with max length values	User object is initialized	1.Set fields to max length values (setName, setPassword, setPhoneNumber, setMail). 2.Call getters.	Getters should return the max length values set by the setters.	Pass
60	Test equality	Two user objects are initialized with same values and one with different value	1.Use assertEquals to compare the equal ones. 2.Use assertNotEquals to compare the unequal ones.	User objects with the same field values should be equal. User objects with different field values should not be equal.	Fail
61	Test equality with null	User object is initialized	1.Create a user object. 2.Compare it with null using assertNotEquals.	User object should not be equal to null.	Pass
62	Test equality with different types	User object is initialized	1.Create a user object. 2.Compare it with a String object using assertNotEquals.	User object should not be equal to an object of a different type.	Pass
63	Test equality with same reference	User object is initialized	1.Create a user object. 2.Compare it with itself using assertEquals.	User object should be equal to itself.	Pass
64	Test equality with different emails	User objects are initialized with different emails	1.Create two User objects with different emails. 2.Use assertNotEquals to compare.	User objects with different emails should not be equal.	Pass
65	Test equality with different phone numbers	User objects are initialized with different phone numbers	1.Create two User objects with different phone numbers. 2.Use assertNotEquals to compare.	User objects with different phone numbers should not be equal.	Pass