C. ABDUL HAKEEM COLLEGE OF ENGINEERING AND TECHNOLOGY, MELVISHARAM

Hakeem Nagar, Melvisharam - 632 509, Ranipet District, Tamil Nadu, India.
(Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai)
(Regd. Under Sec 2(F) & 12(B) of the UGC Act 1956)

DEPARTMENT OF INFORMATIONTECHNOLOGY

CCS366-SOFTWARE TESTING AND AUTOMATION LABORATORY RECORD

(REGULATION - 2021)

Name of the Student :

Register Number :

Degree / Branch :

Year / Semester :

Academic Year :

C. ABDUL HAKEEM COLLEGE OF ENGINEERING AND TECHNOLOGY

Hakeem Nagar, Melvisharam - 632 509, Ranipet District, Tamil Nadu,India.
(Approved by AICTE, New Delhi and Affiliated to Anna University,Chennai)
Regd. Under Sec 2(F) & 12(B) of the UGC Act 1956)

Name of the Candidate:

Year: II **Semester:** III **Degree/Branch:** B.TECH/IT

Subject Name: SOFTWARE TESTING AND AUTOMATION LABORATORY **Subject Code:** CCS366

University Register Number:

CERTIFICATE

Certified that this is the bonafide record of work done by the above student in **CCS366– SOFTWARE TESTING AND AUTOMATION LABORATORY RECORD** during the year 2023 - 2024.

Signature of Head of the Department **Signature of Lab In-charge**

Submitted for the University Practical Examination held on ______________

EXAMINERS

Date:____________________ **Centre code:** 5106

Internal:____________________________ **External:**__________________________

CONTENTS

S. NO	DATE	EXPERIMENTS	PAGE.NO	SIGNATURE

EX NO 1 Develop the tests plan for testing an e-commerce web/mobile application (www.amazon.in).

Aim

Procedure

Test Plan for Ecommerce Website Testing

1 –Homepage
2 – Master page (Header &Footer)
3 –Login&Registration
4 – My Account Page
5 –SearchFeature
6 – Product Listing Page (PLP)
7– Product Details Page (PDP)
8– Shopping Cart
9 –CheckoutFlow
10 –OrderConfirmationPage

1 – Testing Homepage of an ecommerce Site

Obviously homepage of any website is the very first page that many users see. Homepages in ecommerce websites actually go far beyond than just a landing page. Besides a clickable hero image or product slideshow with auto-scroll features, homepage of an online store can actually serve as a powerful marketing tool. Here are some essential test cases that your ecommerce testing team should focus when testing the homepage:

– Does the hero carousel autoscroll? If so, then at what interval (image refresh time)?

– Does the hero carousel continue to autoscroll when the user hovers mouse over the image? Does thescrolling pause or continue to auto scroll to the next image slide?

– When clicked on one of the slider images or call to action (CTA) buttons, is it taking the customer tothe intended page?

– Is the hero slider scrolling too fast?

– Is the homepage loading speed acceptable?

Is the rest of the homepage content such as the newsletters, banners, social media links in the site footer etc easily viewable?

– Does the homepage load and appear the same way across supported browsers and screen resolutions?

– Can the shopping cart be easily located from homepage?

– Can the login/signup button be easily located from homepage?

– Can the contact information be easily located from homepage?

2 – Testing the Master page of an Online Shopping Website

Unlike most other pages, the master page of an ecommerce site is not a stand-alone page but consists of page components that are used in most of the other pages. Typically, the master page consists of the header, the navigation menu and the footer components. Here are some important test cases that your ecommerce QA team should focus when testing the master page components:

– Do the master page components appear on all page types as expected?

– Does the site logo appear as per the design?

– Is the website logo clickable and takes user to the site homepage?

– Is the navigation menu appearing as per the design?

– Does the nav menu constitute all the sub−menus and mega−menu (if implemented)?

– Does the nav menu turn to a burger menu when viewed on smaller screen breakpoints (e.g. mobileand tablet screens)?

– Is the footer appearing as per they design?

– Are all the footer components (e.g. footer menu, footer logos, legal links such as privacy policy,terms & conditions, the © notice and social icons etc) appearing as expected?

– Does the footer menu turn into an accordion to save space on smaller screen sizes (e.g. mobile andtablet screens)?

3 – Testing the Login & Registration flow of an eCommerce Site

While all of the ecommerce sites have user registration feature, some of them do allow their customers to purchase products as a guest, i.e. without forcing them to register and create an account, and then with an optional step to create an account in the order confirmation page after the order has been placed. Once an account has been created, the user can choose to log in at any stage during a checkout process. Here are some good test cases for your ecommerce testing team to focus when testing the login and registration process:

– Assuming that the site permits this, can the customer purchase items as a guest user?

– Can a guest user easily create an account from the order confirmation page after the order has beenplaced?

– Can a guest user register as a member from the registration page?
– Once logged in, are the products in the cart getting correctly associated with the logged in member's account?
– Once logged in, is the user still being prompted to sign in? This shouldn't happen.
– Are the login redirects working as expected and whether the user is being redirected to the correctpage after completing a particular user journey?
– Are the user sessions being maintained for the intended time period?
– Is the user's session timing out and expiring after the stipulated time?

4 – Testing the My Account of an Online Web Store

My Account area can differ from site to site but typically they contain pages such as My Account, Account Information, Address Book, My Orders, Newsletter Subscriptions and a link to Log Out. Here are some good test cases for your ecommerce QA team to focus when testing the My Account areas:

– Is the logged out user prevented from accessing My Account areas?
– Is a logged in user able to access My Account areas?
– Is the user able to add, edit, modify and access their account information (e.g. Name, Email,Password etc) in the Account Information area?
– Is the user able to add, edit, modify and access all the addresses in the Address Book?
– Are the addresses that the user is adding, editing, modifying in the checkout flow appearing asexpected in the Address Book?
– Is the customer able to view, track, cancel and reorder his past orders in My Orders area?
– Is the user able to enable and disable newsletter subscriptions in Newsletter Subscriptions area?
– Is the user able to log out of his session using the Log Out link?

5 – Testing the Search feature of an eCommerce Site

Believe it or not, the search feature is one of the most-used features in an ecommerce store. Even with ecommerce sites with sophisticated product listing (category) pages and easy to use navigation, customers often find it easier to use the website search feature to find exactly what they are looking for. Here are some important test cases for your ecommerce testing team to focus when testing the Search features:

– Is the search feature robust enough to allow searching by product names, brand names or product categories etc? e.g. iPhone X, Apple iPhone, latest iPhone models, cheap iPhone, iPhones in 2023 oreven just iPhones etc.
– Are the search results relevant for all variations of search keywords?
– Does the search result page have sorting options?
If sorting option is available on search results page, then can customers sort the results based on price (highto low, low to high), date ranges, brand, review / ratings etc?

– Is pagination option available on search results page when a long list of products are displayed?
– How many results are displayed on each page?
– Does the next page load automatically when user scrolls? If not, can the user easily navigate to thenext search result page using pagination?

6 – Testing the Categories or Product Listing Page (PLP) of an Online ShoppingStore

While it is neither possible nor prudent to test each and every category page of an ecommerce site, these pages list products that the customers are looking for and hence you must test at least few of the product listing pages randomly to ensure everything is in order. Here are some critical test cases for your ecommerce QA team to focus when testing the Category or Product Listing Page:

– Does this page display the correct products based on the selected category?
– Does this page display proper sorting options based on price (high to low, low to high), date ranges,brand, review / ratings etc?
– Does this page display proper filter options?
– Is pagination option available on this page when a long list of products are displayed?
– How many results are displayed on each page?
– Does the next page load automatically when user scrolls? If not, can the user easily navigate to thenext page using pagination?
– Does this page display the correct recommended products based on the selected category?
– Does this page display the correct related products based on the selected category?
– Does this page display the correct featured products based on the selected category?

7 – Testing the Product Details Page (PDP) of an eCommerce Site

Whether you realize this or not, but this is the page where every other page (home page, the navigation menu, category pages, search result pages etc) is trying to drive traffic in. Thus, the product details page is the page that ultimately begins the checkout flow journey that results in revenue for every ecommerce sites. Here are some crucial test cases for your ecommerce testing team to focus when testing the Product Details Page:

– Does this page display all the product related information (e.g. product name, SKU, price, discount ifany, product images, product specifications, user reviews, Add to Cart button, quantity changer etc) as expected?
– Does this precut display other similar products that were bought by customers who purchased thisitem?
– Can the customer customize the product if it is a customizable product and available with variations(size, colors etc)?
– Does this page show the stock availability accurately?
– Can the user select required number of items using the quantity selector and then add the product tocart?
– Does this product gets added to cart when added?
– Does the quantity in cart get updated when the product is added to cart?

8 – Testing the Shopping Cart of an Online Shopping Platform

Shopping cart is one of the crucial components of an ecommerce website for obvious reasons. Just like physical shopping baskets, the shopping carts in an ecommerce website allows the customers to select and store multiple different items in their cart and then purchase them together when ready. Here are some key test cases, which should be part of testing a shopping cart.
– Does the cart get updated to show updated quantity when an item is added to it?
– Does the mini−cart (that slides when you click on the cart icon) correctly reflect the items, theirquantity and price?
– Can you increase or decrease quantity of an item from the cart and mini−cart?
– Can you remove an item from the cart and mini−cart?
– Can you add the item several times to the cart?
– Can you add several variations (e.g. color, size etc) of the same item to the cart?
– Can you add several items of different types to the cart?
– Can you visit the product detail page by clicking on an item from the cart?
– Does the cart still remembers the items, their quantity and price correctly even if you accidentallyclose the browser and then reopen it?
– Does the cart still remembers the items, their quantity and price correctly even if you log out as themember and then logs back in?
– Can you apply valid coupon codes and discount vouchers in the cart?

– Does the discount get correctly applied when valid coupon codes and discount vouchers are appliedin the cart?

9 – Testing the Checkout Flow of an ecommerce Site

The Checkout flow is the user journey that takes place when the user decides to move the added product items from the shopping cart and to finally purchase them. In the checkout process, usually theuser is required to provide shipping details, billing information and payment details to complete the purchase. Timed out or unsuccessful transactions are one of the top reasons why most online shoppers abandon an ecommerce website without completing their purchase. Failed transactions are thus one of the main reasons of abandoned carts. Here are some of the most crucial test cases to test during the checkout flow.

– If the user is already logged in, does their shipping and billing information get auto– filled incheckout?

– If the user is a guest user, can they enter their shipping and billing information?

– Do all the types of supported payment methods work?

– Can customers check out as guests and make payments?

– For returning customers, does the checkout page prompt them to sign in?

– Are sensitive information such as customer's credit card details, PayPal / Amazon Pay / Google Pay /Apple Pay credentials, etc handled securely and not stored on the site's server?

– Is the user taken to Order Confirmation page upon successful checkout?

10 – Testing the Order Confirmation Page of an Online Shopping Portal

This is the final step of the checkout flow user journey and is displayed only when each steps of the checkout including shipping, billing and payment are successful. Here are some of the most importanttest cases to test on the order confirmation page.

– Does this page display the Order ID correctly?

– Does this page display the product details correctly?

– Does this page offer an easy way to create account, in case of guest users?

– Does the customer receive an order confirmation email / SMS text message along with the order

RESULT:

EX NO : 2 DEVELOP TEST CASES FOR TESTING E -COMMERCE APPLICIATION

AIM:

HOME PAGE

Test Case ID	Module Name	Test Scenario	Test Case
TC001	Home Page	Verify the details on Home page	Verify that home page is displayed after login or not.
TC002	Home Page	Verify the details on Home page	Verify that user name is displayed on home page or not
TC003	Home Page	Verify the details on Home page	Verify that featured products are displayed on home page
TC004	Home Page	Verify the details on Home page	Verify that Search Functionality is present on home page.
TC005	Home Page	Verify the details on Home page	Verify the home page of application on different browsers.
TC006	Home Page	Verify the details on Home page	Verify the alignment on the home page
TC007	Home Page	Verify the details on Home page	Verify that products displayed on home page are clickable.
TC008	Home Page	Verify the details on Home page	Verify that when user clicks on a product, user should be redirected to product specification page.

TC009	Home Page	Verify the details on Home page	Verify that user profile section is present on home page.
TC0010	Home Page	Verify the details on Home page	Verify that products displayed on home page are categorised.

SEARCH FUNTIONALTY

Test Case ID	Module Name	Test Scenario	Test Case
TC001	Product Search	Verify the product search functionality	Verify that search field accepts alphabets, numbers or symbols.
TC002	Product Search	Verify the product search functionality	Verify that after entering search text and clicking on search icon, the search should be performed.
TC003	Product Search	Verify the product search functionality	Verify that the search results should be as per the search query.
TC004	Product Search	Verify the product search functionality	Verify that user should be able to search based on product name,brand name or product specification.
TC005	Product Search	Verify the product search functionality	Verify that filter should be present for filtering the search results bases on Brand, Price, Reviews or Ratings.
TC006	Product Search	Verify the product search functionality	Verify that sorting options should be present on search results page.
TC007	Product Search	Verify the product search functionality	Verify that the number of search results displayed on one page.

TC008	Product Search	Verify the product search functionality	Verify that there should be navigation button(Next and Previous) for navigation to pages.
TC009	Product Search	Verify the product search functionality	Verify that user should be perform search in different categories for example, Movies, Books, Grocery etc.

PRODUCT DETAILS

Test Case ID	Module Name	Test Scenario	Test Case
TC001	Product Details	Verify the details on Product Specification page	Verify that images of product are displayed correctly.
TC002	Product Details	Verify the details on Product Specification page	Verify that price of product is displayed.
TC003	Product Details	Verify the details on Product Specification page	Verify that product reviews are mentioned.
TC004	Product Details	Verify the details on Product Specification page	Verify that product specifications are displayed.
TC005	Product Details	Verify the details on Product Specification page	Verify that information about IN-Stock/Out-Stock are displayed.
TC006	Product Details	Verify the details on Product Specification page	Verify that seller ratings should be displayed.

TC007	Product Details	Verify the details on Product Specification page	Verify that all the variations of product are displayed.
TC008	Product Details	Verify the details on Product Specification page	Verify that shipping information about product are displayed.
TC009	Product Details	Verify the details on Product Specification page	Verify that payment options are mentioned on product page.
TC0010	Product Details	Verify the details on Product Specification page	Verify that product suggestions related to product should be displayed.

CART PAGE

Test Case ID	Module Name	Test Scenario	Test Case
TC001	Cart Page	Verify the details on Cart Page	Verify that when user clicks on add to cart, the product should be added to cart.
TC002	Cart Page	Verify the details on Cart Page	Verify that user should be able to continue shopping after adding items to cart.
TC003	Cart Page	Verify the details on Cart Page	Verify that item quantity should be increased if user adds same item in cart again.
TC004	Cart Page	Verify the details on Cart Page	Verify that total amount of all items should be displayed to user.
TC005	Cart Page	Verify the details on Cart Page	Verity that the amount displayed to user

			should include the taxes and shipping charges as per location.
TC006	Cart Page	Verify the details on Cart Page	Verify that user should not be able to add items in cart beyond a certain limit.
TC007	Cart Page	Verify the details on Cart Page	Verify that when user clicks on remove from cart button the item should be removed from cart.
TC008	Cart Page	Verify the details on Cart Page	Verify that user should be able to apply coupons or vouchers at checkout from cart.
TC009	Cart Page	Verify the details on Cart Page	Verify that items in cart should be present if user logs out and logs in again.

CHECK OUT, PAYMENT PAGE

Test Case ID	Module Name	Test Scenario	Test Case
TC001	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that payments options applicable for the order should be displayed at checkout.
TC002	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that delivery details of items should be displayed at checkout.

TC003	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user should get order details by message or email.
TC004	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user should be directed to home page after checkout.
TC005	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user should be displayed payment options such as Credit Card, Debit Card, Net Banking etc.
TC006	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that if user is not registered then payment should be done as Guest user.
TC007	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user should be provided with an option to save the payment method.
TC008	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that session should be timed out if payment is not done for a certain time.
TC009	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user details should be displayed on payment page for

			registered customers.
TC001 0	Checkout,Pay ments Page	Verify the details on Checkout,Payments Page	Verify that on successful payment email or text message should be delivered to customer along with unique order number.

MY ORDER PAGE

Test Case ID	Module Name	Test Scenario	Test Case
TC001	My Orders Page	Verify the details on Orders Page	Verify that user should be able to track the order on My orders page.
TC002	My Orders Page	Verify the details on Orders Page	Verify that user should be able to change the delivery date and time.
TC003	My Orders Page	Verify the details on Orders Page	Verify that user should be able to cancel the order
TC004	My Orders Page	Verify the details on Orders Page	Verify that user should be able to return the order after delivery of order.
TC005	My Orders Page	Verify the details on Orders Page	Verify that user should be able to exchange the order from My orders page.

TC006	My Orders Page	Verify the details on Orders Page	Verify that user should be able to provide feed back and reviews about the item delivered.

CUSTOMER SERVICE PAGE

Test Case ID	Module Name	Test Scenario	Test Case
TC001	Customer Service Page	Verify the details on Customer Service Page	Verify that customer service options should be present on the website.
TC002	Customer Service Page	Verify the details on Customer Service Page	Verify that different modes of customer service such as Email, Chat or Call should be mentioned.
TC003	Customer Service Page	Verify the details on Customer Service Page	Verify that waiting time to connect to customer service shold be displayed to user.
TC004	Customer Service Page	Verify the details on Customer Service Page	Verify that customer service should be available in different languages.
TC005	Customer Service Page	Verify the details on Customer Service Page	Verify that the timings of customer service option should be displayed to user.

RESULT:

EX NO : 3 TEST THE E-COMMERCE APPLICATION AND REPORT THE DEFECTS IN IT

AIM:

HOME PAGE

Test Case ID	Module Name	Test Scenario	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Status	Defect ID
TC001	Home Page	Verify the details on Home page	Verify that home page is displayed after login or not.	Browser should be installed Internet connection should be present	Username - Username Password- password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
TC002	Home Page	Verify the details on Home page	Verify that user name is displayed on home page or not	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
TC003	Home Page	Verify the details on Home	Verify that featured	Browser should be installed Internet		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should			

		page	products are displayed on home page	connection should be present User should be logged in				be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
TC004	Home Page	Verify the details on Home page	Verify that Search Functionality is present on home page.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and Search functionality is present on home page.			
TC005	Home Page	Verify the details on Home page	Verify the home page of application on different browsers.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and home page should be same on different browsers.			
TC006	Home Page	Verify the details on Home page	Verify the alignment on the home page	Browser should be installed Internet connection should be present		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			

				User should be logged in		3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and alignment of products on home page should be proper.			
TC007	Home Page	Verify the details on Home page	Verify that products displayed on home page are clickable.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on any product displayed on home page	User should be redirected to product specification page.			
TC008	Home Page	Verify the details on Home page	Verify that when user clicks on a product, user should be redirected to product specification page.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			

						5	Click on any product displayed on home page	User should be redirected to product specification page.			
TC009	Home Page	Verify the details on Home page	Verify that user profile section is present on home page.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on user name displayed on home page.	User profile should be displayed.			
TC0010	Home Page	Verify the details on Home page	Verify that products displayed on home page are categorised.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Used should be logged in and home page should be displayed, products should categorised.			

SEARCH FUNTIONALTY

Test Case ID	Module Name	Test Scenario	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Status	Defect id
TC001	Product Search	Verify the product search functionality	Verify that search field accepts alphabets, numbers or symbols.	Browser should be installed Internet connection should be present User should be logged in	Username- Username Password- password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
						5	Click on search field.	Search field should accepts alphabets, numbers or symbols.			
TC002	Product Search	Verify the product search functionality	Verify that after entering search text and clicking on search icon, the search should be performed.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Enter search text in search field and click on search icon.	Search should be performed according to search text provided by user.			
TC003	Product Search	Verify the product search functionality	Verify that the search results should be as per the search query.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Enter search text in search field and click on search icon.	Search should be performed according to search text provided by user.			
TC004	Product Search	Verify the product	Verify that user should	Browser should be installed		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website			

			search functionality	be able to search based on product name,brand name or product specification.	Internet connection should be present User should be logged in				should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	User should be logged in and Search functionality is present on home page.			
							5	Enter search text in search field and click on search icon.	User should be able to perform search bases on product name, brand name or product specification.			
	TC005	Product Search	Verify the product search functionality	Verify that filter should be present for filtering the search results bases on Brand, Price, Reviews or Ratings.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	User should be logged in and home page should be same on different browsers.			
							5	Enter search text in search field and click on search icon.	Search should be performed according to search text provided by user and filters should be present.			
	TC006	Product Search	Verify the product search functionality	Verify that sorting options should be present on search results page.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	User should be logged in and alignment of products on home page should be proper.			
							5	Enter search text in search field and click on search icon.	Search should be performed according to search text provided by user and sorting options should be present on search results page.			
	TC007	Product Search	Verify the product search	Verify that the number of search	Browser should be installed Internet		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			

			functionality	results displayed on one page.	connection should be present User should be logged in		3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Enter search text in search field and click on search icon.	Search should be performed according to the search text provided and a specific number of results should be displayed on one page.			
	TC008	Product Search	Verify the product search functionality	Verify that there should be navigation button(Next and Previous) for navigation to pages.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Enter search text in search field and click on search icon.	Search should be performed according to text provided by user and navigation button should be present on search result page.			
	TC009	Product Search	Verify the product search functionality	Verify that user should be perform search in different categories for example, Movies, Books, Grocery etc.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Enter search text in search field and click on search icon.	Search should be performed in different categories such as movies, books, grocery etc.			

PRODUCT DETAILS

Test Case ID	Module Name	Test Scenario	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Status	Defect ID
TC001	Product Details	Verify the details on Product Specification page	Verify that images of product are displayed correctly.	Browser should be installed Internet connection should be present	Username - Username Password-password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
						5	Click on a product displayed on home page.	the images of product should be displayed correctly.			
TC002	Product Details	Verify the details on Product Specification page	Verify that price of product is displayed.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on a product displayed on home	the images of product should be displayed correctly.			

							page.				
TC003	Product Details	Verify the details on Product Specification page	Verify that product reviews are mentioned.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	The price of product should be displayed.			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on a product displayed on home page.	The reviews about the product should be displayed.			
TC004	Product Details	Verify the details on Product Specification page	Verify that product specifications are displayed.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and Search functionality is present on home page.			
						5	Click on a product displayed	Product specifications should be			

							on home page.	displayed.			
TC0 05	Product Details	Verify the details on Product Specific ation page	Verify that inform ation about IN-Stock/ Out-Stock are displa yed.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and home page should be same on different browsers.			
						5	Click on a product displayed on home page.	Stock information should be displayed.			
TC0 06	Product Details	Verify the details on Product Specific ation page	Verify that seller ratings should be displa yed.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and alignment of products on home page should be proper.			
						5	Click on a product displayed on home page.	The seller ratings should be displayed.			
TC0	Product	Verify the	Verify that all	Browser should be		1	Open	Browser should			

07	Details	details on Product Specification page	the variations of product are displayed.	installed Internet connection should be present User should be logged in			browser	be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on any product displayed on home page	User should be redirected to product specification page.			
						6	Click on a product displayed on home page.	All the variations of product should be displayed.			
TC008	Product Details	Verify the details on Product Specification page	Verify that shipping information about product are displayed.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on any product	User should be redirected to			

							displayed on home page	product specification page.			
						6	Click on a product displayed on home page.	the images of product should be displayed correctly.			
TC0 09	Product Details	Verify the details on Product Specification page	Verify that payment options are mentioned on product page.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	The shipping information should be displayed.			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on user name displayed on home page.	User profile should be displayed.			
						6	Click on a product displayed on home page.	The payment options should be displayed.			
TC0 010	Product Details	Verify the details on Product Specification page	Verify that product suggestions related to produc	Browser should be installed Internet connection should be present User should be		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and	user should be able to input username and			

			t should be displa yed.	logged in			password	password			
						4	Click on login button	Used should be logged in and home page should be displayed, products should be categorised.			
						5	Click on a product displayed on home page.	The product suggestions should be displayed.			

CART PAGE

Test Case ID	Module Name	Test Scenari o	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Statu s	Defec t ID
TC0 01	Cart Page	Verify the details on Cart Page	Verify that when user clicks on add to cart, the produc t should be added to cart.	Browser should be installed Internet connectio n should be present	Username-Username Password-password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
						5	Click on any product displayed on home page	The product page should be displayed.			
						6	Click on add to cart for the product.	The product should be added to cart.			
TC0 02	Cart Page	Verify the details on Cart Page	Verify that user should be able to	Browser should be installed Internet connectio n should		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username	user should be able to input username			

			continue shopping after adding items to cart.	be present User should be logged in			and password	and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on any product displayed on home page	The product page should be displayed.			
						6	Click on add to cart for the product.	The product should be added to cart and user can continue shopping.			
TC003	Cart Page	Verify the details on Cart Page	Verify that item quantity should be increased if user adds same item in cart again.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on any product displayed on home page	The product page should be displayed.			
						6	Click on add to cart for the product.	The product should be added to cart.			
						7	Click on add to cart again for the same product.	The item quantity should be increases.			
TC004	Cart Page	Verify the details on Cart Page	Verify that total amount of all items should	Browser should be installed Internet connection should be		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and	user should be able to input username and password			

				be displayed to user.	present User should be logged in			password				
							4	Click on login button	User should be logged in and Search functionality is present on home page.			
							5	Click on any product displayed on home page	The product page should be displayed.			
							6	Click on add to cart for the product.	The product should be added to cart.			
							7	Click on go to cart button.	The total amount of all items in cart should be displayed.			
	TC005	Cart Page	Verify the details on Cart Page	Verity that the amount displayed to user should include the taxes and shipping charges as per location.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	User should be logged in and home page should be same on different browsers.			
							5	Click on any product displayed on home page	The product page should be displayed.			
							6	Click on add to cart for the product.	The product should be added to cart.			
							7	Click on go to cart button.	The total amount should include taxes and shipping charges.			
	TC006	Cart Page	Verify the details on Cart	Verify that user should	Browser should be installed Internet		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			

		Page	not be able to add items in cart beyond a certain limit.	connection should be present User should be logged in		3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and alignment of products on home page should be proper.			
						5	Click on any product displayed on home page	The product page should be displayed.			
						6	Click on add to cart for the product for many times.	The user should not be allowed to add items to cart beyond a certain limit.			
TC007	Cart Page	Verify the details on Cart Page	Verify that when user clicks on remove from cart button the item should be removed from cart.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on any product displayed on home page	User should be redirected to product specification page.			
						6	Click on any product displayed on home page	The product page should be displayed.			
						7	Click on add to cart for the product.	The product should be added to cart.			

							8	Click on remove from cart button.	The item should be removed form cart.			
	TC0 08	Cart Page	Verify the details on Cart Page	Verify that user should be able to apply coupons or vouchers at checkout from cart.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Click on any product displayed on home page	User should be redirected to product specification page.			
							6	Click on any product displayed on home page	The product page should be displayed.			
							7	Click on add to cart for the product.	The product should be added to cart.			
							8	Click on go to cart button.	User should be able to apply cupons or vouchers.			
	TC0 09	Cart Page	Verify the details on Cart Page	Verify that items in cart should be present if user logs out and logs in again.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Click on user name	User profile should be displayed.			

							displayed on home page.				
						6	Click on any product displayed on home page	The product page should be displayed.			
						7	Click on add to cart for the product.	The product should be added to cart.			
						8	Click on Logut button	User is logged out.			
						9	Login again with same userid password	The user should be logged in and the items shold be present in cart.			

CHECK OUT ,PAYMENT

Test Case ID	Module Name	Test Scenario	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Status	Defect ID
TC001	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that payments options applicable for the order should be displayed at checkout.	Browser should be installed Internet connection should be present	Username - Username Password- password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
						5	Click on any product displayed on home page	The product page should be displayed.			

							6	Click on add to cart for the product.	The product should be added to cart.			
							7	Click on Checkout button.	The checkout page should be displayed with payments options.			
	TC002	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that delivery details of items should be displayed at checkout.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Click on any product displayed on home page	The product page should be displayed.			
							6	Click on add to cart for the product.	The product should be added to cart.			
							7	Click on Checkout button.	The checkout page should be displayed with delivery details.			
	TC003	Checkout,Payments Page	Verify the details on Checkout,Payments	Verify that user should get order details	Browser should be installed Internet connection should be		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			

			Page	by message or email.	present User should be logged in		3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Click on any product displayed on home page	The product page should be displayed.			
							6	Click on add to cart for the product.	The product should be added to cart.			
							7	Click on Checkout button.	The checkout page should be displayed with payments options.			
							8	Make payment for the order.	User should get order details by message or email.			
	TC004	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user should be directed to home page after checkout.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	User should be logged in and Search functionality is present on home page.			

						5	Click on any product displayed on home page	The product page should be displayed.			
						6	Click on add to cart for the product.	The product should be added to cart.			
						7	Click on Checkout button.	The checkout page should be displayed with payments options.			
						8	Make payment for the order.	Ther order should be confirmed and user should be directed to home page.			
TC0 05	Checko ut,Paym ents Page	Verify the details on Checko ut,Paym ents Page	Verify that user should be display ed payme nt options such as Credit Card, Debit Card, Net Bankin g etc.	Browser should be installed Internet connecti on should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and home page should be same on different browsers.			
						5	Click on any product displayed on home page	The product page should be displayed.			
						6	Click on add to cart for the product.	The product should be added to cart.			

							7	Click on Checkout button.	The checkout page should be displayed with payments options such as debit card, credit card, netbanking etc.			
	TC006	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that if user is not registered then payment should be done as Guest user.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Click on any product displayed on home page	The product page should be displayed.			
							4	Click on add to cart for the product.	The product should be added to cart.			
							5	Click on Checkout button.	The checkout page should be displayed with payments options.			
							6	Make payment for the order.	The payment should be done as guest user.			
							7					
	TC007	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user should be provided with an option to save	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			

				the payme nt metho d.			4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
							5	Click on any product displayed on home page	User should be redirected to product specification page.			
							6	Click on any product displayed on home page	The product page should be displayed.			
							7	Click on add to cart for the product.	The product should be added to cart.			
							8	Click on Checkout button.	The checkout page should be displayed with payments options.			
							9	Make payment for the order.	the user should be able to save payment method for future orders.			
	TC0 08	Checko ut,Paym ents Page	Verify the details on Checko ut,Paym ents Page	Verify that session should be timed out if payme nt is not done for a certain time.	Browser should be installed Internet connecti on should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			

							5	Click on any product displayed on home page	User should be redirected to product specification page.			
							6	Click on any product displayed on home page	The product page should be displayed.			
							7	Click on add to cart for the product.	The product should be added to cart.			
							8	Click on Checkout button.	The checkout page should be displayed with payments options.			
							9	Select payment method and click on pay.	Payment page should be displayed.			
							10	Stay on payment page for long time wtihout any activity.	The payment session should be timedout.			
	TC009	Checkout,Payments Page	Verify the details on Checkout,Payments Page	Verify that user details should be displayed on payment page for registered customers.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			

							5	Click on user name displayed on home page.	User profile should be displayed.			
							6	Click on any product displayed on home page	The product page should be displayed.			
							7	Click on add to cart for the product.	The product should be added to cart.			
							8	Click on Checkout button.	The checkout page should be displayed with user details.			
	TC0 010	Checko ut,Paym ents Page	Verify the details on Checko ut,Paym ents Page	Verify that on succes sful payme nt email or text messag e should be deliver ed to custom er along with unique order numbe r.	Browser should be installed Internet connecti on should be present User should be logged in		1	Open browser	Browser should be opened			
							2	Open url	Ecommerce website should be opened			
							3	Enter username and password	user should be able to input username and password			
							4	Click on login button	Used should be logged in and home page should be displayed, products should be categorised.			
							5	Click on any product displayed on home page	The product page should be displayed.			
							6	Click on add to cart for the product.	The product should be added to cart.			

						7	Click on Checkout button.	The checkout page should be displayed with payments options.			
						8	Make payment.	Order should be confirmed and email or text message should be delivered to customer.			

MY ORDER PAGE

Test Case ID	Module Name	Test Scenario	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Status	Defect ID
TC001	My Orders Page	Verify the details on Orders Page	Verify that user should be able to track the order on My orders page.	Browser should be installed Internet connection should be present	Username - Username Password- password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
						5	Click on My orders button	The orders by user should be displayed.			
						6	Click on Track order for a order.	Tracking information should be displayed for that order.			
TC002	My Orders Page	Verify the details on Orders Page	Verify that user should be able to change the deliver	Browser should be installed Internet connection should be present User		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			

			y date and time.	should be logged in		4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on My orders button	The orders by user should be displayed.			
						6	Click on change delivery date and time button.	The user should be provided details for selecting delivery date and time.			
TC0 03	My Orders Page	Verify the details on Orders Page	Verify that user should be able to cancel the order	Browser should be installed Internet connecti on should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on My orders button	The orders by user should be displayed.			
						6	Click on cancel order for an order.	User should be able to cancel the order.			
TC0 04	My Orders Page	Verify the details on Orders Page	Verify that user should be able to return the order after deliver y of order.	Browser should be installed Internet connecti on should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and Search functionality is present on home page.			

						5	Click on My orders button	The orders by user should be displayed.			
						6	Click on return item for an order.	User should be provided with pickup date and time for return.			
TC005	My Orders Page	Verify the details on Orders Page	Verify that user should be able to exchange the order from My orders page.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and home page should be same on different browsers.			
						5	Click on My orders button	The orders by user should be displayed.			
						6	Click on exchange item for an order.	User should be given details of exchange item .			
TC006	My Orders Page	Verify the details on Orders Page	Verify that user should be able to provide feed back and reviews about the item delivered.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and alignment of products on home page should be proper.			
						5	Click on My orders button	The orders by user should be displayed.			
						6	Click on Write review button for an item.	User can write review and provide feedback.			

CUSTOMER SERVICE PAGE

Test Case ID	Module Name	Test Scenario	Test Case	Pre-requisites	Test Data	Step No.	Test Step	Expected Result	Actual Result	Status	Defect ID
TC001	Customer Service Page	Verify the details on Customer Service Page	Verify that customer service options should be present on the website.	Browser should be installed Internet connection should be present	Username- Username Password - password	1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login			
						5	Click on contact customer care button.	Customer care page is displayed.			
TC002	Customer Service Page	Verify the details on Customer Service Page	Verify that different modes of customer service such as Email, Chat or Call should be mentioned.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on contact customer care button.	Customer care page is displayed with different modes of customer service such as Email,chat or call.			
TC003	Customer Service Page	Verify the details on Customer Service Page	Verify that waiting time to connect to customer	Browser should be installed Internet connection should be		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and	user should be able to input username and password			

			service shold be displayed to user.	present User should be logged in			password				
						4	Click on login button	Home page should be displayed after login and user name should be displayed on home page			
						5	Click on contact customer care button.	Customer care page is displayed with different modes of customer service such as Email,chat or call.			
						6	Select any mode of customer service.	Waiting time is displayed for that customer service.			
TC004	Customer Service Page	Verify the details on Customer Service Page	Verify that customer service should be available in different languages.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and Search functionality is present on home page.			
						5	Click on contact customer care button.	Customer care page is displayed with different available languages.			
TC005	Customer Service Page	Verify the details on Customer Service Page	Verify that the timings of customer service option should be displayed to user.	Browser should be installed Internet connection should be present User should be logged in		1	Open browser	Browser should be opened			
						2	Open url	Ecommerce website should be opened			
						3	Enter username and password	user should be able to input username and password			
						4	Click on login button	User should be logged in and home page should be same on different browsers.			

						5	Click on contact customer care button.	Customer care page is displayed with timings of different modes of customer care.			

RESULT:

EX NO : 4 DEVELOP THE TEST PLAN AND DESIGN THE TEST CASES FOR AN INVENTORY CONTROL SYSTEM

AIM:

Test Plan for an Inventory Control System

1. Introduction

The Inventory Control System is designed to efficiently manage inventory, track stock levels, and ensure accurate stock information. This test plan outlines the testing approach, scope, objectives, and methodologies to verify the functionality, performance, and usability of the system.

2. Objectives

The primary objectives of this test plan are to:

 Validate that the Inventory Control System meets the functional requirements.

 Verify the accuracy and consistency of inventory tracking.

 Assess system performance under varying loads.

 Ensure the system&#39;s security mechanisms are effective.

 Evaluate the system&#39;s usability and user-friendliness.

3. Test Scope

The testing will cover the following aspects of the Inventory Control System:

 User authentication and authorization.

 Adding, updating, and deleting products in the inventory.

 Monitoring stock levels and receiving alerts for low stock.

 Generating various inventory reports.

 User interfaces for different user roles (admin, manager, employee).

4. Test Environment

- Hardware: Servers, workstations, mobile devices for testing different platforms.
- Software: Inventory Control System build, web browsers, databases.
- Network: LAN/WAN with various network conditions (latency, bandwidth).
- Test Data: Realistic inventory data and user scenarios for testing.

5. Test Data

- Test data will include a range of products, quantities, prices, and suppliers.
- Both normal and boundary test cases will be created to cover various scenarios.
- Data will be structured to validate different calculations and reports.

6. Test Cases

A comprehensive list of test cases will be created for functional, performance, security, and usability testing.

Each test case will include:

- Test case ID
- Description of the scenario
- Preconditions
- Steps to execute the test
- Expected results
- Actual results
- Pass/fail status

7. Types of Testing

- Unit Testing: Testing individual modules and functions.
- Integration Testing: Testing interactions between different system components.

 Functional Testing: Validating functional requirements.

 Performance Testing: Evaluating system response times and resource usage.

 Security Testing: Checking for vulnerabilities and unauthorized access.

 Usability Testing: Assessing user-friendliness and navigation.

8. Test Schedule

Testing Phases and Estimated Timeline:

 Unit Testing: 1 week

 Integration Testing: 2 weeks

 Functional Testing: 2 weeks

 Performance Testing: 1 week

 Security Testing: 1 week

 Usability Testing: 1 week

9. Defect Management

 Defects will be logged using a defect tracking tool.

 Defects will be classified by severity (critical, major, minor) and priority.

 Defects will be retested after resolution.

10. Risk Assessment

 Identified risks include data loss, system downtime, security breaches.

 Mitigation strategies include regular data backups, redundancy, security protocols.

11. Test Deliverables

 Test cases document

 Test execution results and defect reports

 Performance testing results and analysis

 Usability testing observations and feedback

12. Resources

- Test Manager: Responsible for overall test coordination.
- Testers: Responsible for test execution, logging defects, and retesting.

13. Exit Criteria

- All high-priority defects are resolved.
- Key performance indicators meet predefined targets.
- All test cases are executed and passed.

14. Approval

This test plan requires approval from relevant stakeholders before testing commences.

15. Appendix

- Glossary of terms
- References used in the test plan

TEST CASES FOR INVENTORY MANAGEMENT SYSTEM

Sr.No	Testcases	Action	Steps	Input Data	Experienced result	Actual result	status
1	TC−1	Invoice no	Enter Invoice no	Input*1021*	It should accepted invoice no.		
2	TC−2	Bill date	Enter Bill date	Input*27/11/2021*	It should accepted bill date		
3	TC−3	Item name	select Item name	-	Item name should be automatically reflected		
4	TC−4	Available item stock	Click on text box	-	It should reflect automatically item stock		
5	TC−5	Quantity	Enter item quantity	Input*5000*	Item quantity should be accepted		
6	TC−6	Price	Click on text box	-	Price should be reflected automatically		
7	TC−7	Total	Click on text box	-	Total should be reflected automatically		
8	TC−8	Receive bill date	Enter receive bill date	Input*29/1/2021	Receive bill date should be accepting		
9	TC−9	Add item	Click on add item	-	It should be add item reflecting in database		

RESULT:

EX NO: 5 EXECUTE THE TEST CASES AGAINST A CLIENT SERVER OR DESKTOP APPLICATION AND IDENTIFY THE DEFECTS.

AIM:

Program:

1. **html:**

```
<html>
<head>
<title>JavascriptLoginFormValidation</title>
<!--IncludeCSSFileHere-->
<linkrel="stylesheet"href="form-style.css"/>
<!--IncludeJSFileHere -->
<scriptsrc="login.js"></script>
</head>
<body>
<divclass="container">
<divclass="main">
<h2>JavascriptLoginFormValidation</h2>
<formid="form_id"method="post"name="myform">
<label>UserName:</label>
<inputtype="text"name="username"id="username"/>
<label>Password:</label>
<inputtype="password"name="password"id="password"/>
```

```
<inputtype="button"value="Login"id="submit"onclick="validate()"/>

</form>

<span><bclass="note">Note:</b>For this demousefollowingusernameand password.
<br/><bclass="valid">UserName: Form<br/>Password:123</b></span>

</div>

</div>

</body>

</html>
```

login.js:

```
varattempt =3; //Variable tocount numberofattempts.

//BelowfunctionExecutesonclickofloginbutton.functio

nvalidate(){ varusername=document.getElementById("username")

.value;varp

assword=document.getElementById("password").value;

if(username=="Form"&&password=="123"){alert("

Login successfully");

window.location="success.html";//Redirectingtootherpage.returnf

alse;

}

else{

attempt --; // Decrementing by one.

Ialert("Youhaveleft"+attempt+"attempt;");

// Disabling fields after 3

attempts.if(attempt ==0){

document.getElementById("username").disabled=true;doc

ument.getElementById("password").disabled =

true;document.getElementById("submit").disabled =

true;returnfalse;
}
}
}
```

success.html:

```
<html>
<head>
<title>JavascriptLoginFormValidation</title>
</head>
<body>
<h2>SuccessfulLogin!</h2>
</body>
</html>
```

RESULT:

Ex.No:6 TEST THE PERFORMANCE OF THE E-COMMERCE APPLICATION

AIM:

E-Commerce applications are designed in such a way that customers can easily make purchases at any time of the day, irrespective of where they are located. If an eCommerce application malfunctions during peak hours, not only will it cause customer dissatisfaction, but it will also lead to revenue loss for the business.

To avoid such situations, companies can invest in performance testing of eCommerce applications that focuses on issues related to speed, stability, and scalability. Hence, in the wake of increasing demand for online shopping, it is necessary to do performance testing of the eCommerce application and toensure that the application is stable and scalable enough to serve all customer requirements.

Goals of E-Commerce Performance Testing

Performance testing is carried out on eCommerce software to ensure that the platform is functioning as expected. Let's explore why eCommerce application performance testing is important and how it affects the app's overall quality:

- Reduce Risks: Many times, making major and considerable changes to a site can cause notable strategic changes or even trigger significant losses. However, testing these changes in a planned manner can help eliminate the chances of these uncertain revenue losses .
- Increase Conversion Rates: By testing every aspect of an application and ensuring a smooth visitor experience through site optimization, the application conversion rate is bound to increase.
- Improve User Engagement: Testing tells which page element or process affects a user’s onsite journey and helps in rectifying the issues faster. The better the user experience, the greater the onsite engagement.

Benefits of E-Commerce Application Performance Testing

Performance testing, when done effectively and consistently, can greatly boost conversions while also enhancing the overall experience of site users. The following reasons explain the significance of testing and optimization:

- Coupon codes or gift vouchers are provided to increase product sales and performance tests ensure that coupon codes work well when applied to users in bulk.

- Multiple systems like email servers, social network sites, and enterprise content management systems are involved in the backend and the performance test ensures flawless functioning of various features like product images/videos, and social media in such an integrated system.

- eCommerce applications usually have more users than any other application and validation of users is a must to filter out genuine customers. Performance test helps in validating multiple sign-up and invoice email notifications in parallel.

- Augmented Reality (AR) is an advanced-level feature provided by modern eCommerce websites so that customers can feel better without seeing products in person. With augmented reality performance testing, we can make the feature more efficient and pleasing to the eye.

- A recommendation system is a program that generates online recommendations using several algorithms, artificial intelligence, and data analysis. A well-designed recommendation system helps in better customer acquisition and retention. It is developed for both new and existingcustomerswith different behavior for different customer types. Performance testing helps in making the recommendation system a faster system, to retain online shoppers.

- Chatbots interact with online shoppers to improve the user experience, and any delay in providing the required information may switch users to a competitor's application. Performance tests speed up chatbots to answer customer queries.

- Reviews are very important to increase the sale of a product. The majority of users prefer visiting the review listing page to get instant user feedback. Performance test helps in optimizing customer review listing load time.

- eCommerce applications attract the maximum number of customers during annual sales events. Backend servers should be set with the optimal settings, keeping future users' traffic in mind. Performance tests help in identifying the servers' optimum locations to manage a higher number of users.

Common Performance Testing Issues

Some common challenges faced in implementing performance testing are as follows:

- Slow DB server: Database size increases with the increase in product items of all ages, so ensure to optimize DB queries.

- Faster Integrated Systems Communication: Multiple systems like email servers, social media accounts, and payment channels are involved in the functioning of an eCommerce application to ensure that integrated systems work well under a heavy load .
- System Scaling: Brands cannot predict user load during peak hours, hence, the need of the hour is to be ready with tested scalable systems.

- Immoderate Scripts: Few eCommerce applications are associated with third-party scripts that are running in the background like Google Analytics and Ads, and these widgets can be a significant contributor to page load time.

- Slow Servers: Server resources and applications should be optimized to deliver content as quickly as possible after addressing any content-related performance issues.
- Massive Media: Oftentimes, images, videos, and other media are uploaded by non-technical employees that aren't familiar with the impact of loaded heavy files. It's important to ensure these assets are optimized to maximize performance

Main Features to be Tested in E-Commerce Applications

Your eCommerce website's success depends on choosing the features to test to ensure that it performs flawlessly. Some of the major features that require testing are as follows:

- Searching For a Product: eCommerce sites provide various search filters to make the search experience faster, which include price range, product type, country, and age. Hence, ensure that searching for a product among billions of records does not halt the system and return the expected records.
- Virtual Reality: Modern eCommerce sites provide advanced-level features to visualize products. Online shoppers can now see how they would look wearing an item virtually Hence, we need toensure the smooth movement of virtual images for a better experience.

- Testing Payment Gateway Integration: Failed transactions are one of the main reasons why most customers exit an application without completing the purchase. So it's important to ensure the proper functioning of payment gateways.

- Product Details Page: A fast product listing is important because every consumer will be attracted to the product by how it's presented and how it's visible to them.

Best Practices for Performance Testing

Mentioned below are some of the best practices for testing eCommerce applications:

- Test all CDNs (content delivery networks) by repeating time-bound hot transactions multiple times.
- Performance testing of big data plays an important role to ensure faster decompression and compression cycles to speed up the application’s speed.
- Online shopping has opened doors worldwide for businesses. Vendors shifted their servers to the cloud for better user coverage. Hence, tests from various geo-locations need to be performed to identify latency issues and to ensure that the geographical location of a user does not affect the speed of the application.
- Test individual servers before integration system testing to find bottlenecks like email servers, social accounts, DB servers, and enterprise content management systems.
- Volume testing plays a crucial role in seeing how storage systems function with or without a heavy load, hence including it in the test scope is a good practice.

Role of QASource

Testing experts at QASource have years of experience in running performance tests on a wide range of domains including eCommerce, media streaming, legal, human resources, eLearning, healthcare , enterprise content management, financial, and many more. QASource assists teams in setting up the performance test lab and identifying the best-fit performance test scenarios to ensure the best user experience. Following are our high-level performance testing services to meet the performance test requirement:

- Performance test plan
- Test data setup
- Simulation of test cases
- Generate performance test suite results
- Identify bottlenecks and follow up with the Dev team to fix them
- Generate daily status reports

Conclusion

Performance testing of eCommerce applications is a crucial part of any business's success.
It helps eCommerce applications generate better user experience, safeguard user data security, and ensure mobile responsiveness, security, and a quick load time. It is vital to conduct quick and effective application performance testing from time to time to ensure that the application is offering an exceptional customer experience to maximize revenue for businesses

RESULT:

EX NO: 7 AUTOMATE THE TESTING OF E-COMMERCE APPPLICATIONS USING SELEINIUM

AIM:

Positive Scenario

1. Test Case - Automate the User Registration process of an e-commerce website.

Steps to Automate:
1. Open this URL http://automationpractice.com/index.php
2. Click on the sign-in link.
3. Enter your email address in the 'Create and Account' section.
4. Click on Create an Account button.
5. Enter your Personal Information, Address, and Contact info.
6. Click on the Register button.
7. Validate that the user is created.

Selenium code for User Registration:

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;
import io.github.bonigarcia.wdm.WebDriverManager;
public class EcomSignUp {
 public static void main(String[] args)
  { WebDriverManager.chromedriver().setup
  ();WebDriver driver=new ChromeDriver();
  String URL="http://automationpractice.com/index.php";

  driver.get(URL);
  driver.manage().timeouts().implicitlyWait(2000, TimeUnit.MILLISECONDS);
  driver.manage().window().maximize();
  //Click on Sign in
  driver.findElement(By.linkText("Sign in")).click();
  //Enter email address
driver.findElement(By.cssSelector("[name='email_create']")).sendKeys("test1249@test.com");
  //Click on "Create an account"
  driver.findElement(By.xpath("//button[@name=\"SubmitCreate\"]")).click();
  //Select Title
  driver.findElement(By.xpath("//input[@id=\"id_gender1\"]")).click();
  driver.findElement(By.name("customer_firstname")).sendKeys("Test User");
  driver.findElement(By.name("customer_lastname")).sendKeys("Vsoft");
  driver.findElement(By.id("passwd")).sendKeys("PKR@PKR");
  // Enter your address
  driver.findElement(By.id("firstname")).sendKeys("Test User");
  driver.findElement(By.id("lastname")).sendKeys("Vsoft");
  driver.findElement(By.id("company")).sendKeys("Vsoft");
  driver.findElement(By.id("address1")).sendKeys("Test 81/1,2nd cross");
  driver.findElement(By.id("city")).sendKeys("XYZ");
  // Select State
  WebElement statedropdown=driver.findElement(By.name("id_state"));
  Select oSelect=new Select(statedropdown);
  oSelect.selectByValue("4");
  driver.findElement(By.name("postcode")).sendKeys("51838");
  // Select Country
  WebElement countrydropDown=driver.findElement(By.name("id_country"));
  Select oSelectC=new Select(countrydropDown);
  oSelectC.selectByVisibleText("United States");
  //Enter Mobile Number
  driver.findElement(By.id("phone_mobile")).sendKeys("234567890");
  driver.findElement(By.xpath("//input[@name=\"alias\"]")).clear();
  driver.findElement(By.xpath("//input[@name=\"alias\"]")).sendKeys("Office");
  driver.findElement(By.name("submitAccount")).click();
  String userText=driver.findElement(By.xpath("//*[@id=\"header\"]/div[2]/div/div/nav/div[1]/a")).getText();
  // Validate that user has created
  if(userText.contains("Vsoft"))
  { System.out.println("User Verified,Test case
  Passed");
  }
  else {
   System.out.println("User Verification Failed,Test case Failed");
  }
 }}
```

<u>**Negative Scenarios**</u>

2. Test Case - Verify invalid email address error.

Steps to Automate:
1. Open this URL http://automationpractice.com/index.php
2. Click on the sign-in link.
3. Enter an invalid email address in the email box and click enter.
4. Validate that an error message is displaying saying "Invalid email address."

3. Test Case - Verify error messages for mandatory fields.

Steps to Automate:
1. Open this URL http://automationpractice.com/index.php
2. Click on the sign-in link.
3. Enter your email address and click the Register button.
4. Leave the mandatory fields (marked with *) blank and click the Register button.
5. Verify that an error has been displayed for the mandatory fields.

4. Test Case - Verify error messages for entering incorrect values in fields.

Steps to Automate:
1. Open this URL http://automationpractice.com/index.php
2. Click on the sign-in link.
3. Enter your email address and click the Register button.
4. Enter incorrect values in fields like., enter numbers in first and last name, city field, etc., and enter alphabets in Mobile no, Zip postal code, etc., and click on the 'Register' button.
5. Verify that error messages for respective fields are displaying.

Try automating the above scenarios using Selenium commands, if you face any difficulty please refer to the Selenium Tutorial series.

5. Automate the 'Search Product' feature of Amazon like e-commerce website with Selenium

1. Test Case - Automate the 'Search Product' feature of the e-commerce website with Selenium.

Steps to Automate:
1. Open link http://automationpractice.com/index.php
2. Move your cursor over the Women's link.
3. Click on the sub-menu 'T-shirts'
4. Get the Name/Text of the first product displayed on the page.
5. Now enter the same product name in the search bar present at the top of the page and click the search button.
6. Validate that the same product is displayed on the searched page with the same details which were displayed on T-Shirt's page.

Automation Code for Product Search:

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
import io.github.bonigarcia.wdm.WebDriverManager;
public class EcomPractice2 {
 public static void main(String[] args) throws
  InterruptedException{WebDriverManager.chromedriver().setup();
  WebDriver driver=new ChromeDriver();
  String URL="http://automationpractice.com/index.php";
  driver.get(URL);
  driver.manage().window().maximize();
  // Initialise Actions class object
  Actions actions=new Actions(driver);
  driver.manage().timeouts().implicitlyWait(2000, TimeUnit.MILLISECONDS);
  WebElement womenTab=driver.findElement(By.linkText("WOMEN"));
  WebElement
TshirtTab=driver.findElement(By.xpath("//div[@id='block_top_menu']/ul/li[1]/ul/li[1]/ul//a[@title='T-
shirts']"));
  actions.moveToElement(womenTab).moveToElement(TshirtTab).click().perform();
  Thread.sleep(2000);
  // Get Product Name
  String
ProductName=driver.findElement(By.xpath("/html[1]/body[1]/div[1]/div[2]/div[1]/div[3]/div[2]/ul[1]/li[1]/di
v[1]/div[2]/h5[1]/a[1]")).getText();
  System.out.println(ProductName);
  driver.findElement(By.id("search_query_top")).sendKeys(ProductName);
  driver.findElement(By.name("submit_search")).click();
  // Get Name of Searched Product
  String
SearchResultProductname=driver.findElement(By.xpath("/html[1]/body[1]/div[1]/div[2]/div[1]/div[3]/div[2]/
ul[1]/li[1]/div[1]/div[2]/h5[1]/a[1]")).getText();
  // Verify that correct Product is displaying after search
  if(ProductName.equalsIgnoreCase(SearchResultProductname))
  {System.out.println("Results Matched;Test Case Passed");
  }else{
   System.out.println("Results NotMatched;Test Case Failed");
  }
  // Close the browser
  driver.close();
 }
}
```

6. Automate the 'Buy Product' feature of Amazon like an e-commerce website with Selenium

The most important function of an e-commerce website is buying a product, which includes various steps like selecting a product, selecting size/color, adding to the cart, checkout, etc. You will find every test scenario along with the automation code in the following section.

1. Test Case - Automate the end-to-end "Buy Product" feature of the e-commerce website.

Steps to Automate:
1. Open link http://automationpractice.com/index.php
2. log in to the website.
3. Move your cursor over the Women's link.
4. Click on the sub-menu 'T-shirts'.
5. Mouse hover on the second product displayed.
6. 'More' button will be displayed, click on the 'More' button.
7. Increase quantity to 2.
8. Select size 'L'
9. Select color.
10. Click the 'Add to Cart' button.
11. Click the 'Proceed to checkout' button.
12. Complete the buy order process till payment.
13. Make sure that the Product is ordered.

Automation code for Buy product

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.Select;
import io.github.bonigarcia.wdm.WebDriverManager;
public class EcomExpert {
 public static void main(String[]
  args){ WebDriverManager.chromedriver().
  setup();WebDriver driver=new
  ChromeDriver();
  String URL="http://automationpractice.com/index.php";

  // Open URL and Maximize browser window
  driver.get(URL);
  driver.manage().window().maximize();
  driver.manage().timeouts().implicitlyWait(3000, TimeUnit.MILLISECONDS);
  //Click on Sign in
  driver.findElement(By.linkText("Sign in")).click();
  //Login
  driver.findElement(By.id("email")).sendKeys("test1249@test.com");
  driver.findElement(By.id("passwd")).sendKeys("PKR@PKR");
  driver.findElement(By.id("SubmitLogin")).click();
  //Click on Women
  driver.findElement(By.linkText("WOMEN")).click();
  WebElement
SecondImg=driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[3]/div[2]/ul/li[2]/div/div[1]/div/a[1]/i
mg"));
  WebElement
MoreBtn=driver.findElement(By.xpath("/html/body[1]/div[1]/div[2]/div[1]/div[3]/div[2]/ul/li[2]/div[1]/div[2]
/div[2]/a[2]"));
  Actions actions=new Actions(driver);
  actions.moveToElement(SecondImg).moveToElement(MoreBtn).click().perform();
  //Change quantity by 2
  driver.findElement(By.id( "quantity_wanted" )).clear();
  driver.findElement(By.id( "quantity_wanted" )).sendKeys("2");
  //Select size as L
  WebElement Sizedrpdwn=driver.findElement(By.xpath("//*[@id='group_1']"));
  Select oSelect=new Select(Sizedrpdwn);
  oSelect.selectByVisibleText("M");
  //Select Color
  driver.findElement(By.id("color_11")).click();
  //Click on add to cart
  driver.findElement(By.xpath("//p[@id='add_to_cart']//span[.='Add to cart']")).click();
  //Click on proceed
  driver.findElement(By.xpath("/html//div[@id='layer_cart']//a[@title='Proceed to checkout']/span")).click();
  //Checkout page Proceed
  driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[3]/div/p[2]/a[1]/span")).click();
  driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[3]/div/form/p/button/span")).click();
  //Agree terms&Conditions
  driver.findElement(By.xpath("//*[@id=\"cgv\"]")).click();
  driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[3]/div/div/form/p/button/span")).click();
  //Click on Payby Check
  driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[3]/div/div/div[3]/div[2]/div/p/a")).click();
  //Confirm the order
  driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[3]/div/form/p/button/span")).click();
  //Get Text
  String ConfirmationText=driver.findElement(By.xpath("//div[@id='center_column']/p[@class='alert alert-
success']")).getText();
  // Verify that Product is ordered
  if(ConfirmationText.contains("complete"))
  { System.out.println("Order Completed: Test Case Passed");
  }
  else {
   System.out.println("Order Not Successfull: Test Case Failed");
  }

 }
}
```

2. Test Case - Verify that 'Add to Wishlist' only works after login.

Steps to Automate:

1. Open link http://automationpractice.com/index.php
2. Move your cursor over the Women's link.
3. Click on the sub-menu 'T-shirts'.
4. Mouse hover on the second product displayed.
5. 'Add to Wishlist' will appear on the bottom of that product, click on it.
6. Verify that the error message is displayed 'You must be logged in to manage your wish list.'

3. Test Case - Verify that Total Price is reflecting correctly if the user changes quantity on the 'Shopping Cart Summary' Page.

Steps to Automate:

1. Open link http://automationpractice.com/index.php
2. Log in to the website.
3. Move your cursor over the Women's link.
4. Click on the sub-menu 'T-shirts'.
5. Mouse hover on the second product displayed.
6. 'More' button will be displayed, click on the 'More' button.
7. Make sure the quantity is set to 1.
8. Select size 'M'
9. Select the color of your choice.

10. Click the 'Add to Cart' button.
11. Click the 'Proceed to checkout' button.
12. Change the quantity to 2.
13. Verify that the Total price is changing and reflecting the correct price.

RESULT:

EX.NO:8 INTEGRATE TestNG WITH THE ABOVE TEST AUTOMATION

AIM:

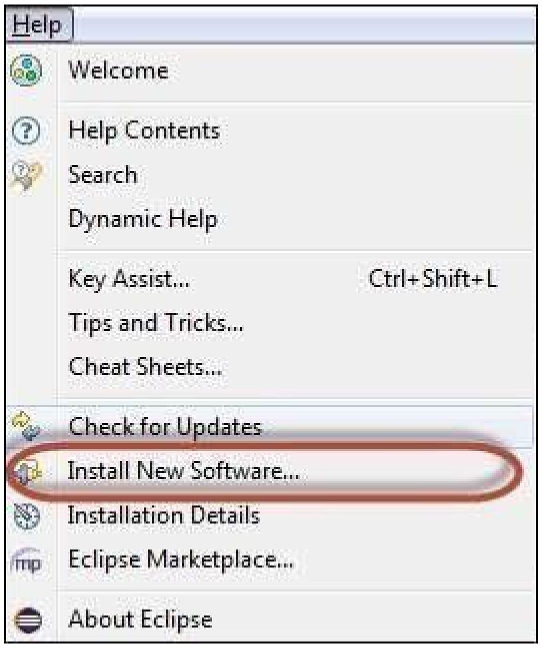
TestNG is a powerful testing framework, an enhanced version of JUnit which was in use for a long time before TestNG came into existence. NG stands for 'Next Generation'.

TestNG framework provides the following features −

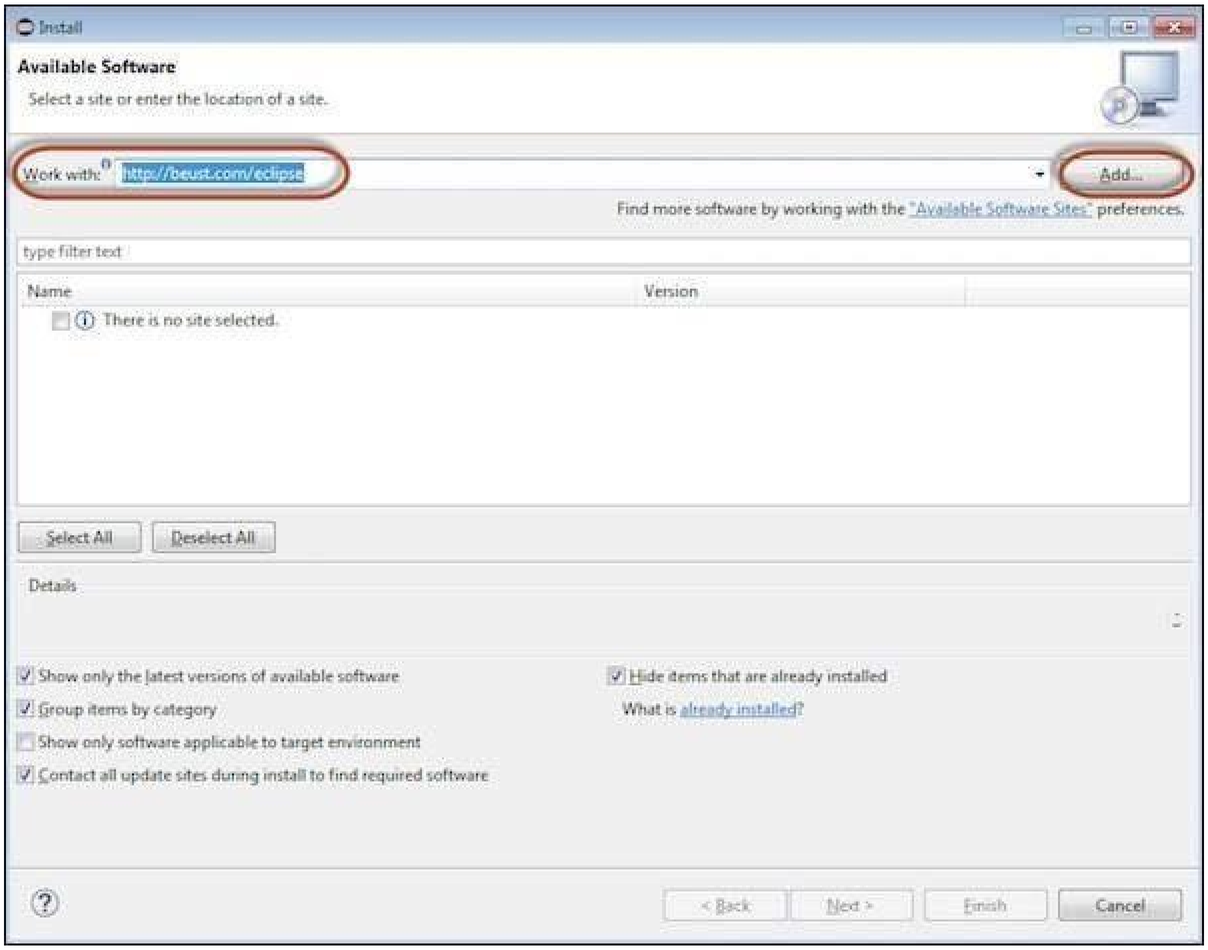
- Annotations help us organize the tests easily.
- Flexible test configuration.
- Test cases can be grouped more easily.
- Parallelization of tests can be achieved using TestNG.
- Support for data-driven testing.
- Inbuilt reporting.

Installing TestNG for Eclipse

Step 1 − Launch Eclipse and select 'Install New Software'.



Step 2 – Enter the URL as 'http://beust.com/eclipse' and click 'Add'.


Install

Available Software

Select a site or enter the location of a site.

Work with: http://beust.com/eclipse

Find more software by working with the "Available Software Sites" preferences.

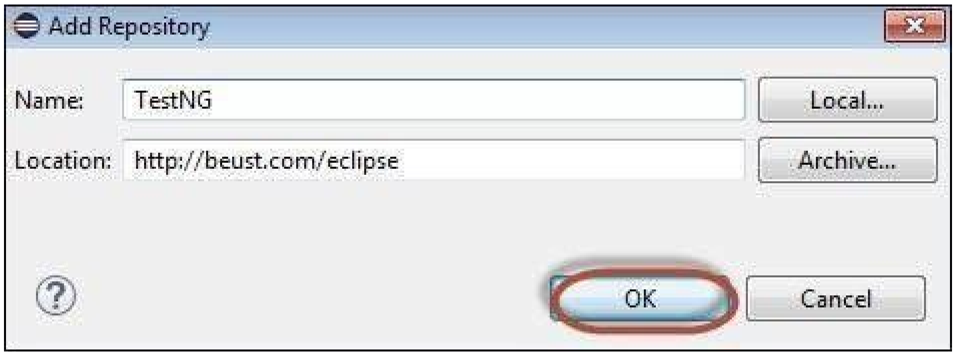
Name	Version
There is no site selected.	

Details

- [✓] Show only the latest versions of available software
- [✓] Group items by category
- [ ] Show only software applicable to target environment
- [✓] Contact all update sites during install to find required software
- [✓] Hide items that are already installed

What is already installed?

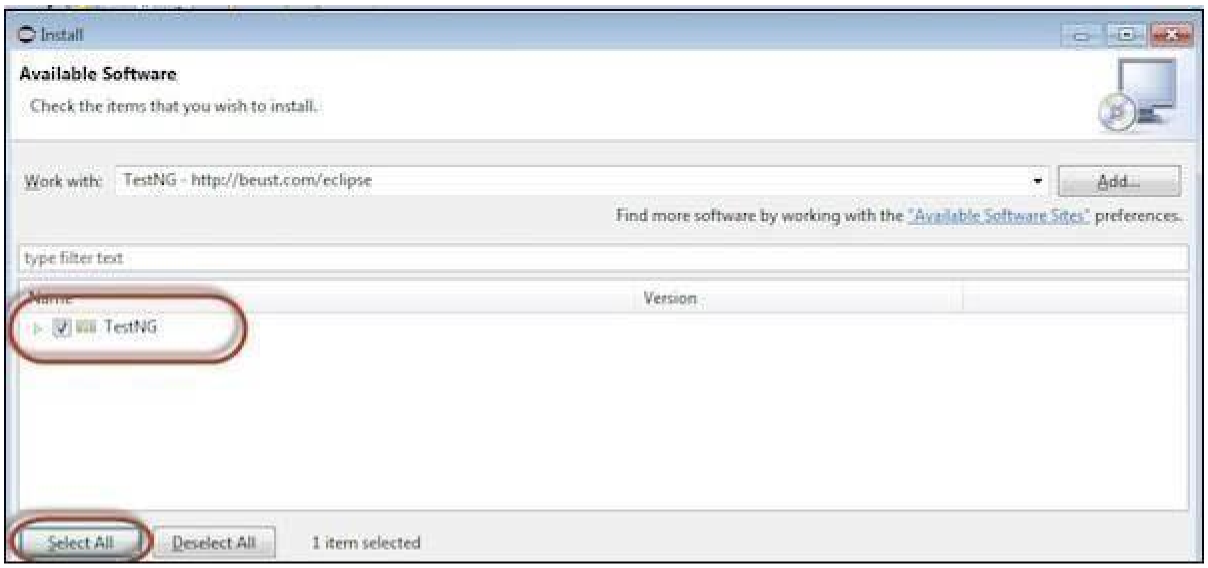
Step 3 – The dialog box 'Add Repository' opens. Enter the name as 'TestNG' and click 'OK'



Name: TestNG

Location: http://beust.com/eclipse

Step 4 – Click 'Select All' and 'TestNG' would be selected as shown in the figure.



Available Software

Check the items that you wish to install.

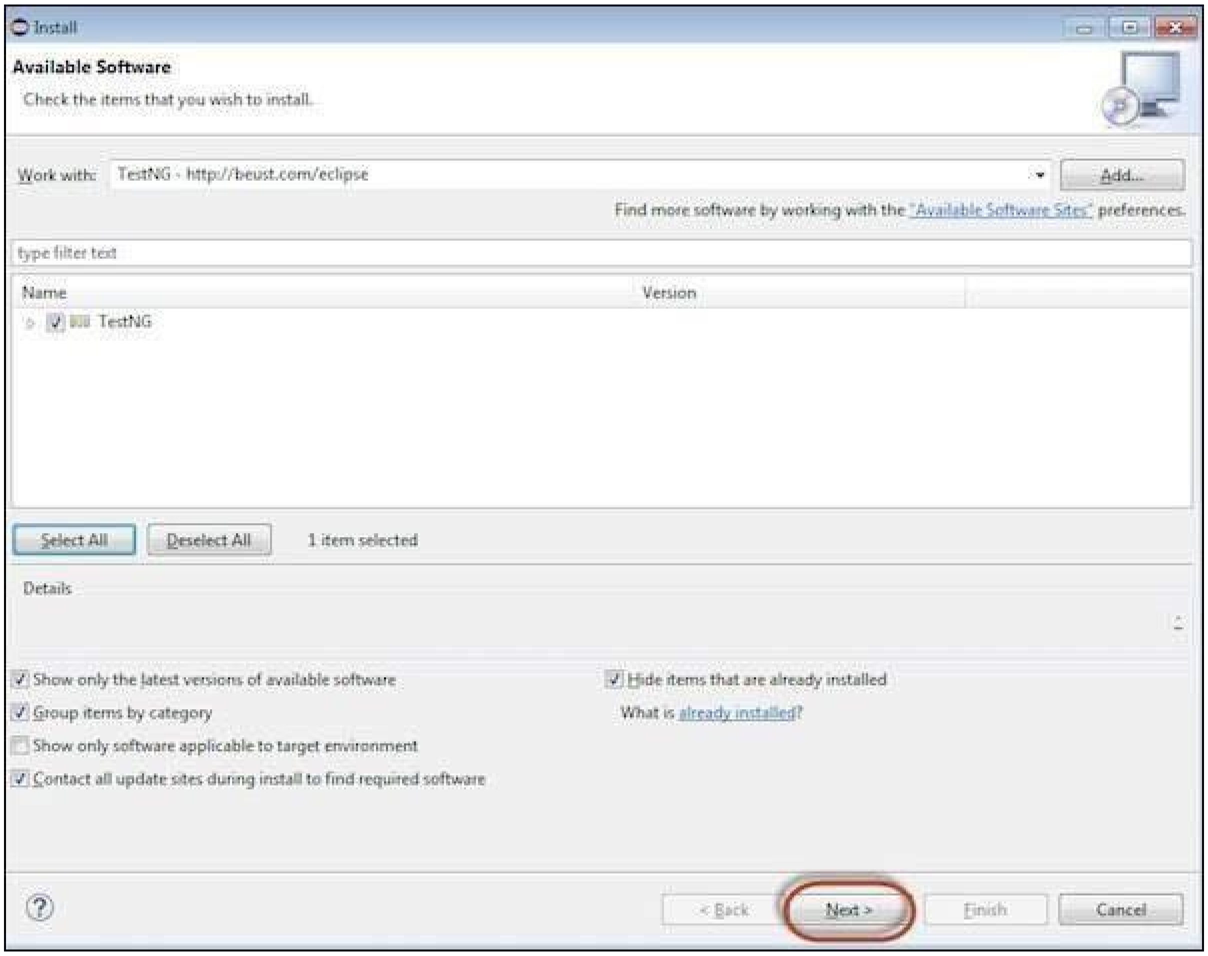
Work with: TestNG - http://beust.com/eclipse

Find more software by working with the "Available Software Sites" preferences.

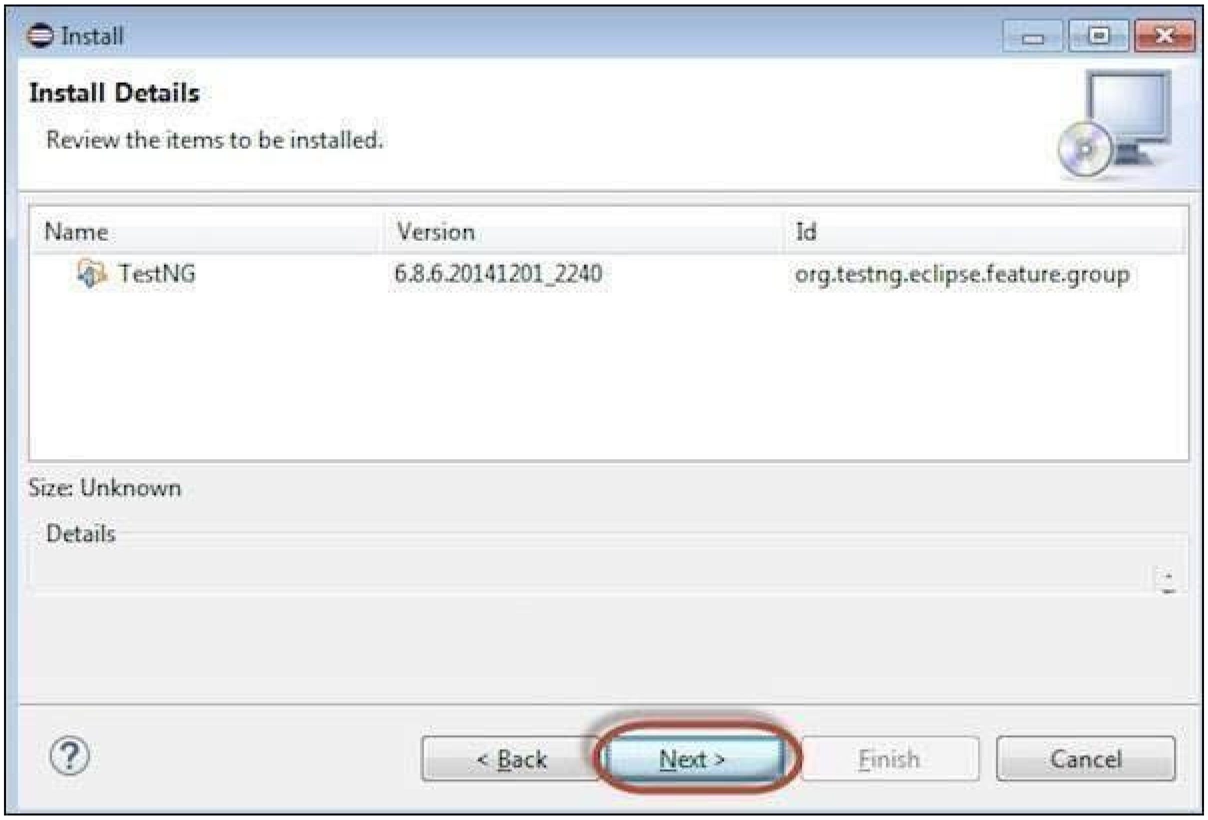
☑ TestNG

1 item selected

Step 5 – Click 'Next' to continue.


Install
Available Software
Check the items that you wish to install.
Work with: TestNG - http://beust.com/eclipse
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
TestNG
Select All
Deselect All
1 item selected
Details
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Show only software applicable to target environment
Contact all update sites during install to find required software
< Back
Next >
Finish
Cancel

Step 6 − Review the items that are selected and click 'Next'.



Install

Install Details

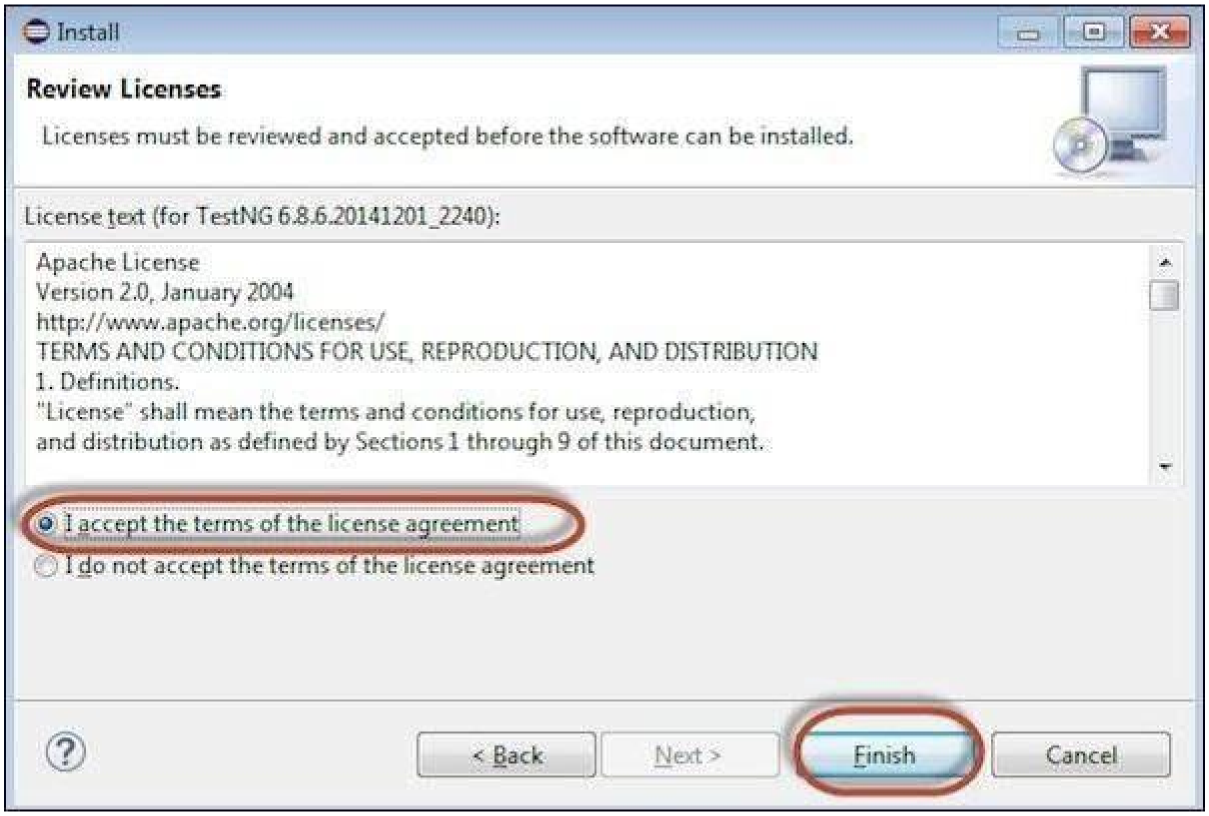
Review the items to be installed.

Name	Version	Id
TestNG	6.8.6.20141201_2240	org.testng.eclipse.feature.group

Size: Unknown

Details

Step 7 – "Accept the License Agreement" and click 'Finish'.



Review Licenses

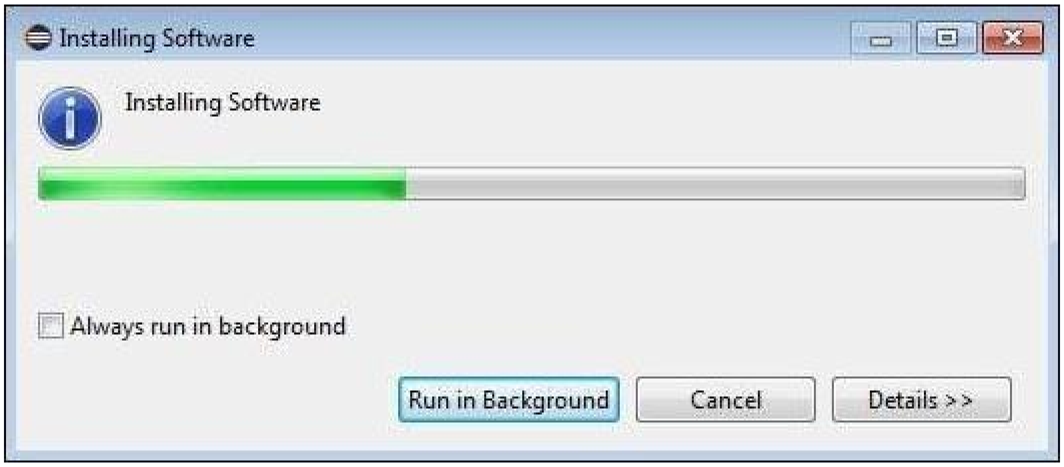
Licenses must be reviewed and accepted before the software can be installed.

License text (for TestNG 6.8.6.20141201_2240):

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/
TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
1. Definitions.
"License" shall mean the terms and conditions for use, reproduction,
and distribution as defined by Sections 1 through 9 of this document.

[✓ I accept the terms of the license agreement]
I do not accept the terms of the license agreement

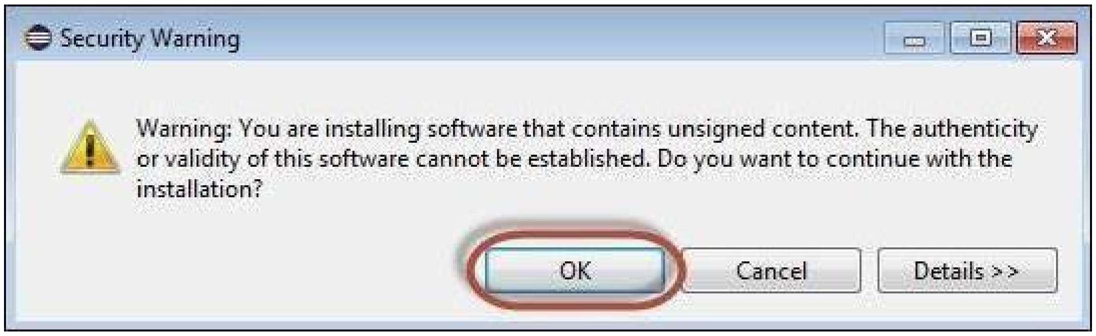
Step 8 − TestNG starts installing and the progress would be shown follows.



Installing Software

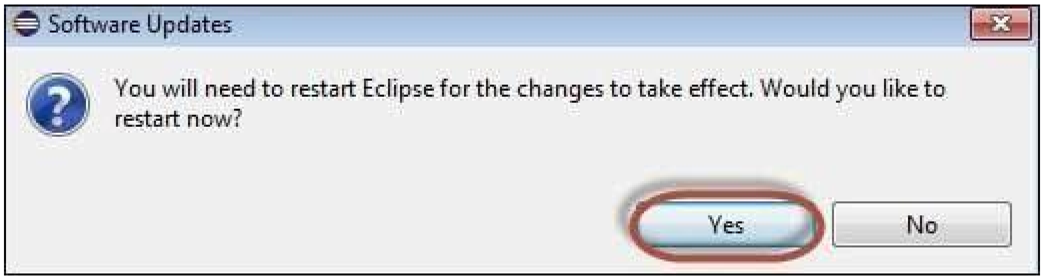
Always run in background

Step 9 − Security Warning pops up as the validity of the software cannot be established. Click 'Ok'.


Security Warning

Warning: You are installing software that contains unsigned content. The authenticity or validity of this software cannot be established. Do you want to continue with the installation?

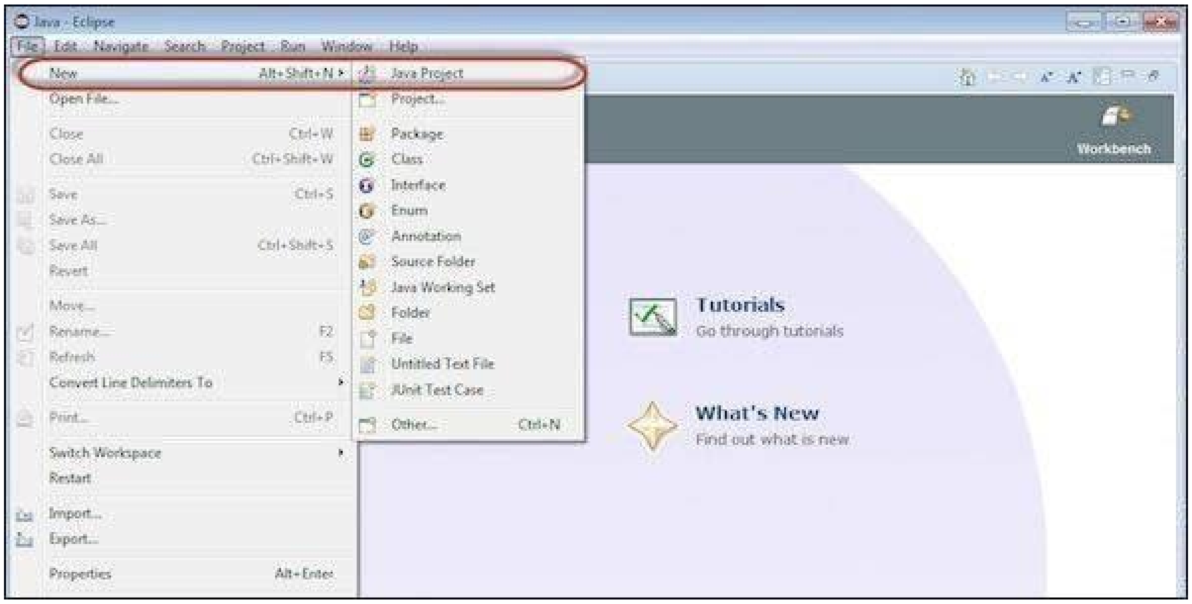
Step 10 – The Installer prompts to restart Eclipse for the changes to take effect. Click 'Yes'.


Software Updates

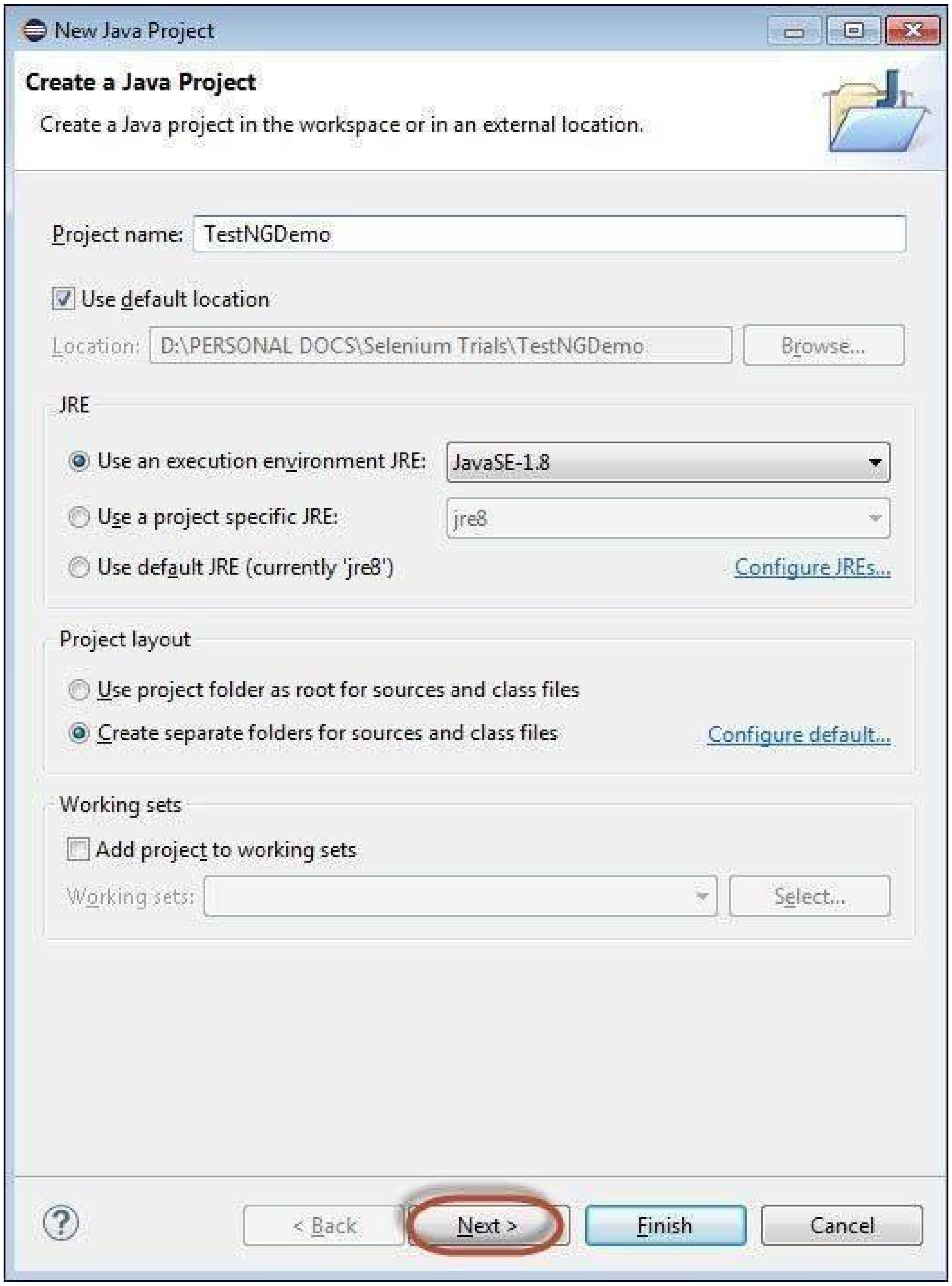
You will need to restart Eclipse for the changes to take effect. Would you like to restart now?

TestNG-Eclipse Setup

Step 1 – Launch Eclipse and create a 'New Java Project' as shown below.



Step 2 − Enter the project name and click 'Next'.



New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: TestNGDemo

[✓] Use default location

Location: D:\PERSONAL DOCS\Selenium Trials\TestNGDemo

JRE

(●) Use an execution environment JRE: JavaSE-1.8

( ) Use a project specific JRE: jre8

( ) Use default JRE (currently 'jre8')

Project layout

( ) Use project folder as root for sources and class files

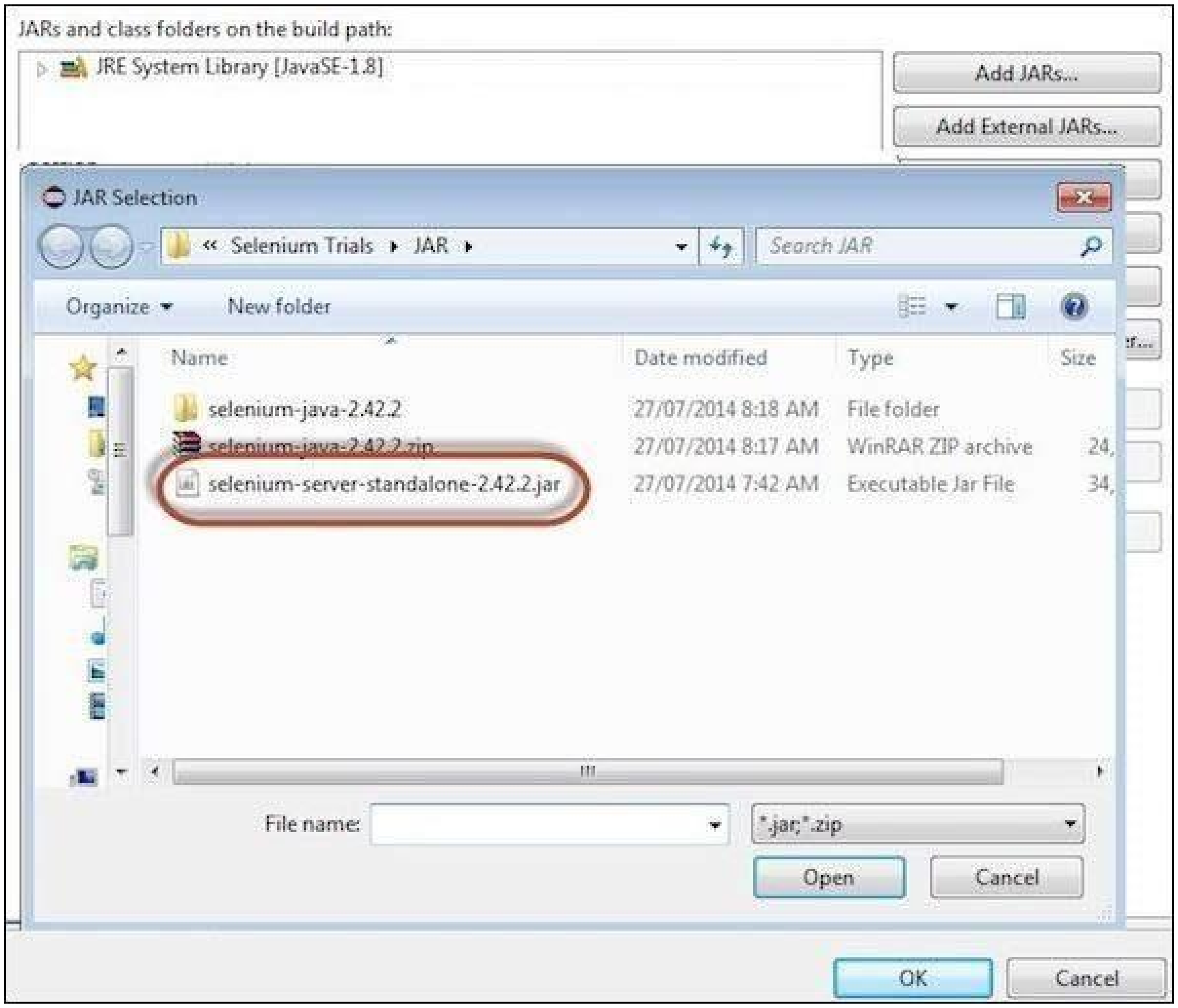
(●) Create separate folders for sources and class files

Working sets

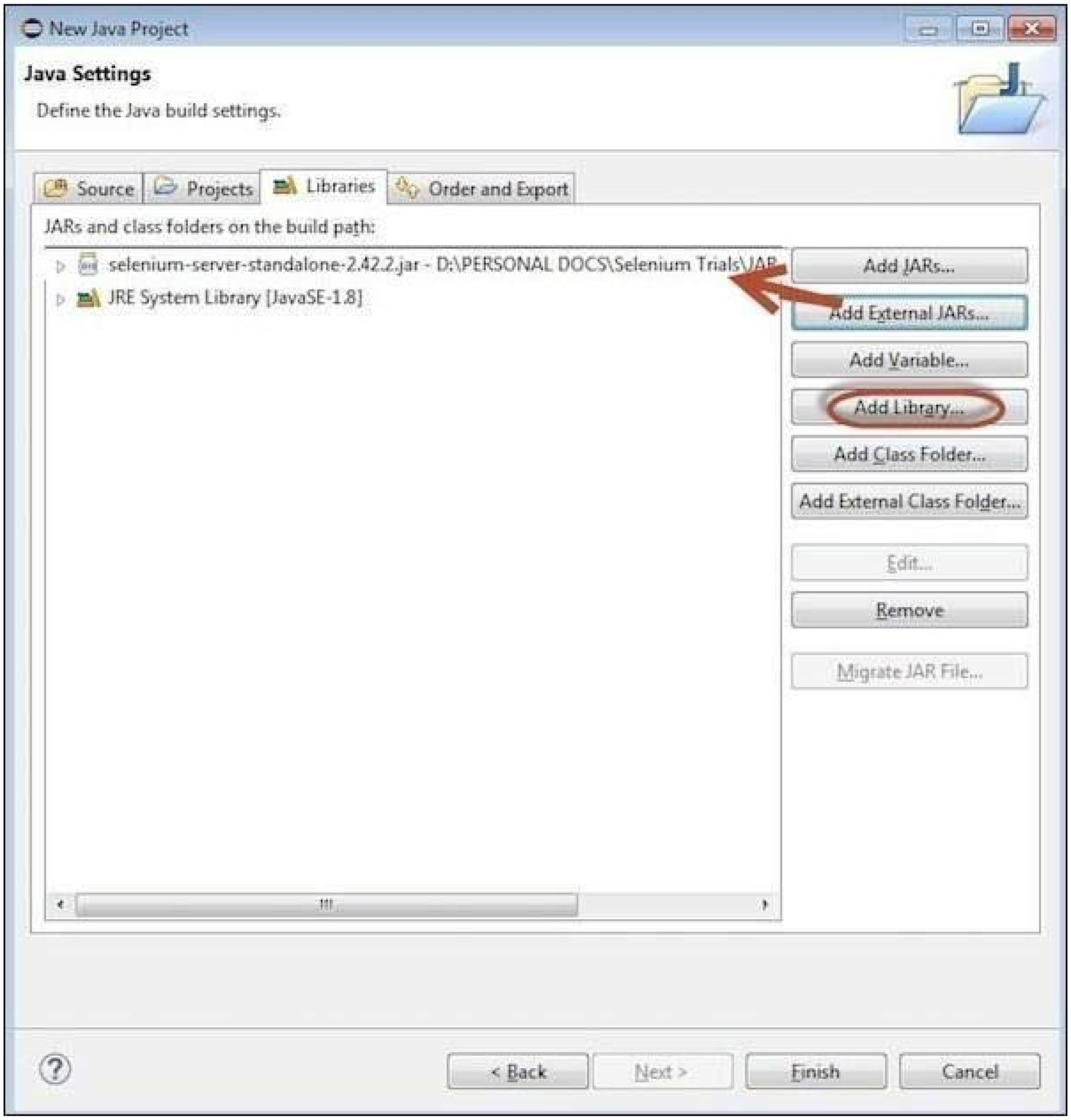
[ ] Add project to working sets

Working sets:

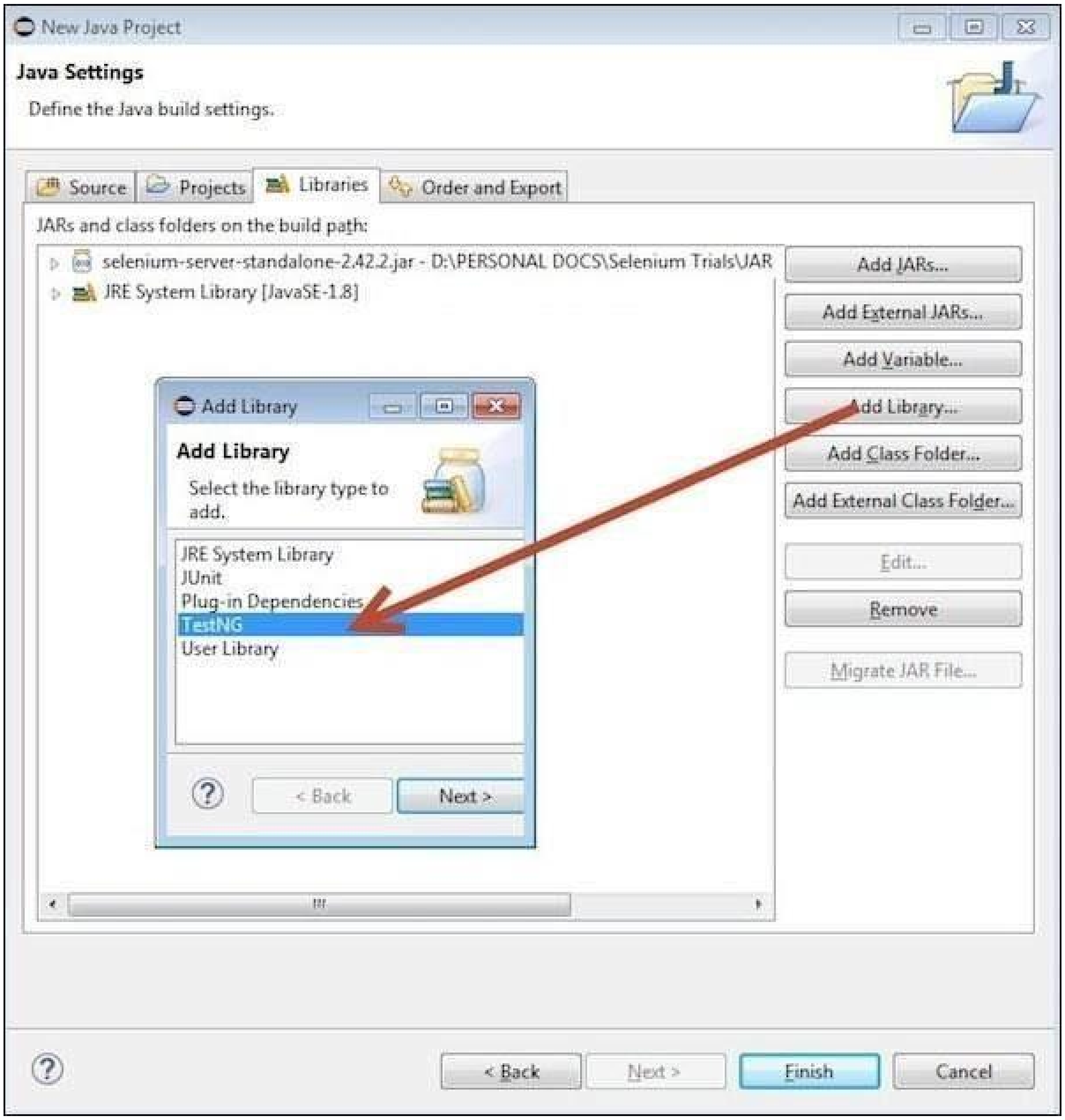
Step 3 − Navigate to "Libraries" Tab and Add the Selenium Remote Control Server JAR file by clicking on "Add External JAR's" as shown below.



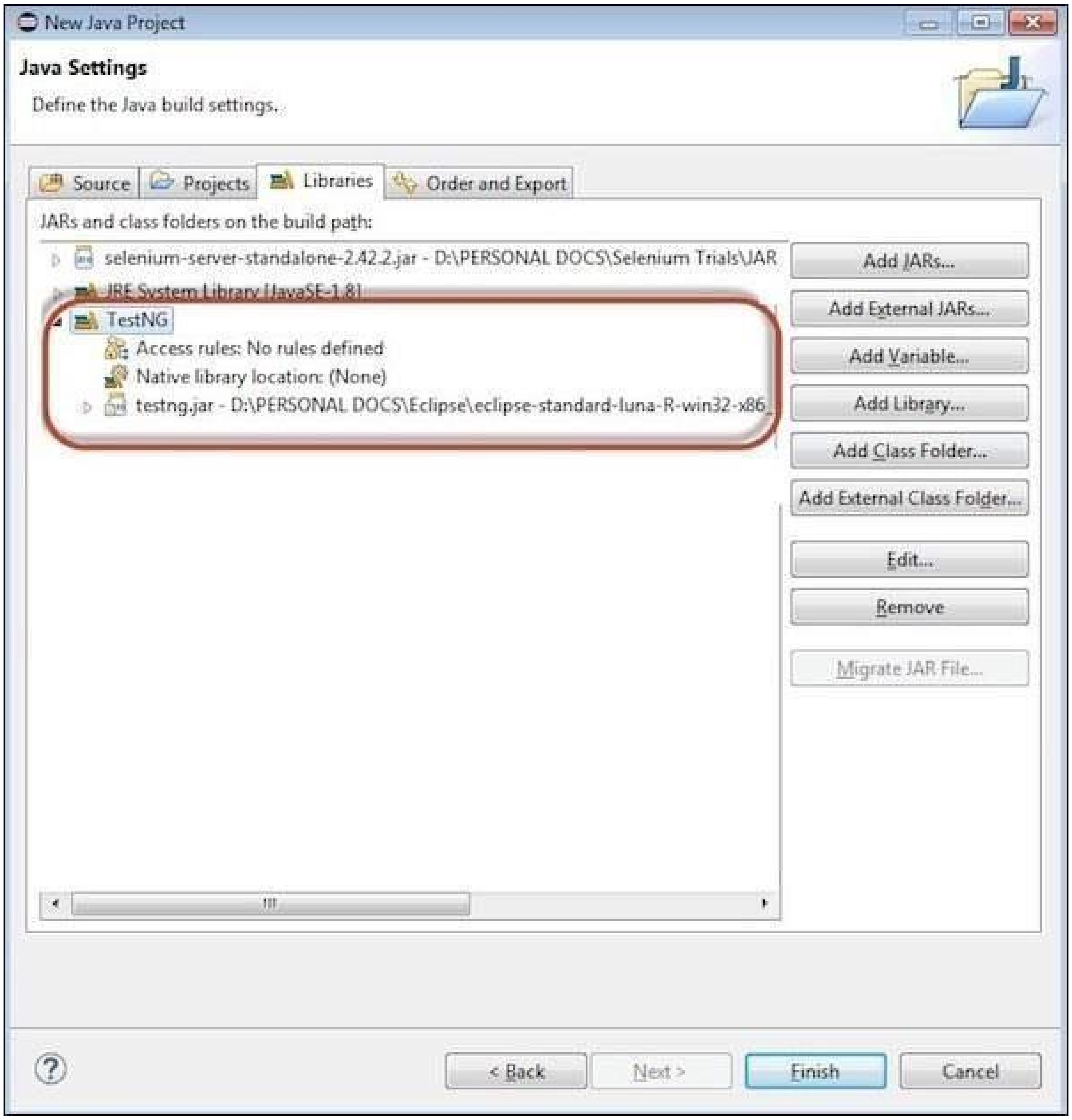
Step 4 − The added JAR file is shown here. Click 'Add Library'.


New Java Project
Java Settings
Define the Java build settings.
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
selenium-server-standalone-2.42.2.jar - D:\PERSONAL DOCS\Selenium Trials\JAR
JRE System Library [JavaSE-1.8]
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
Migrate JAR File...
< Back
Next >
Finish
Cancel

Step 5 − The 'Add Library' dialog opens. Select 'TestNG' and click 'Next' in the 'Add Library' dialog box.



Step 6 − The added 'TestNG' Library is added and it is displayed as shown below.


New Java Project

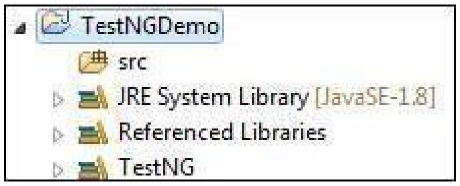
Java Settings

Define the Java build settings.

JARs and class folders on the build path:

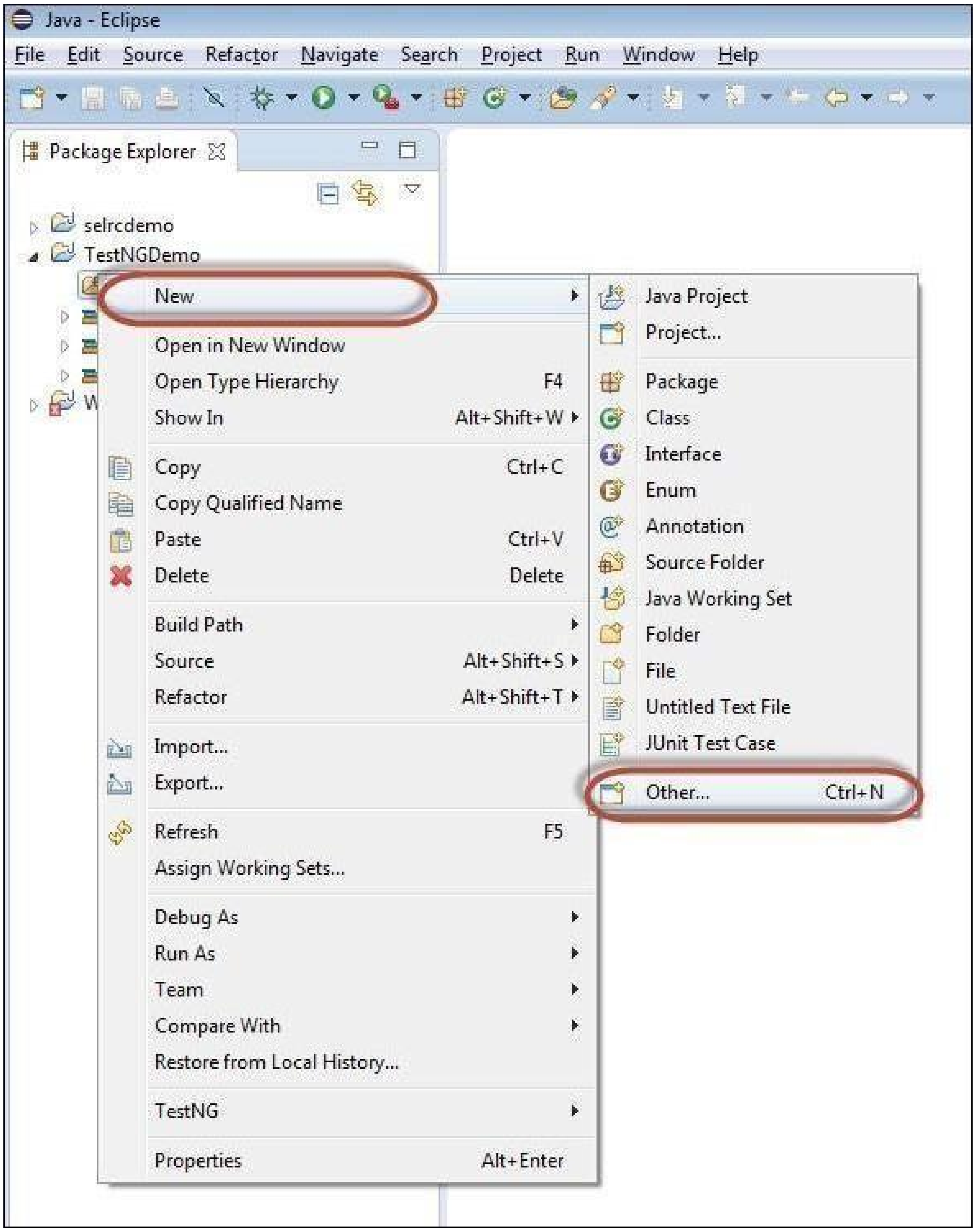
```
selenium-server-standalone-2.42.2.jar - D:\PERSONAL DOCS\Selenium Trials\JAR
JRE System Library [JavaSE-1.8]
TestNG
    Access rules: No rules defined
    Native library location: (None)
    testng.jar - D:\PERSONAL DOCS\Eclipse\eclipse-standard-luna-R-win32-x86_
```

Step 7 − Upon creating the project, the structure of the project would be as shown below.

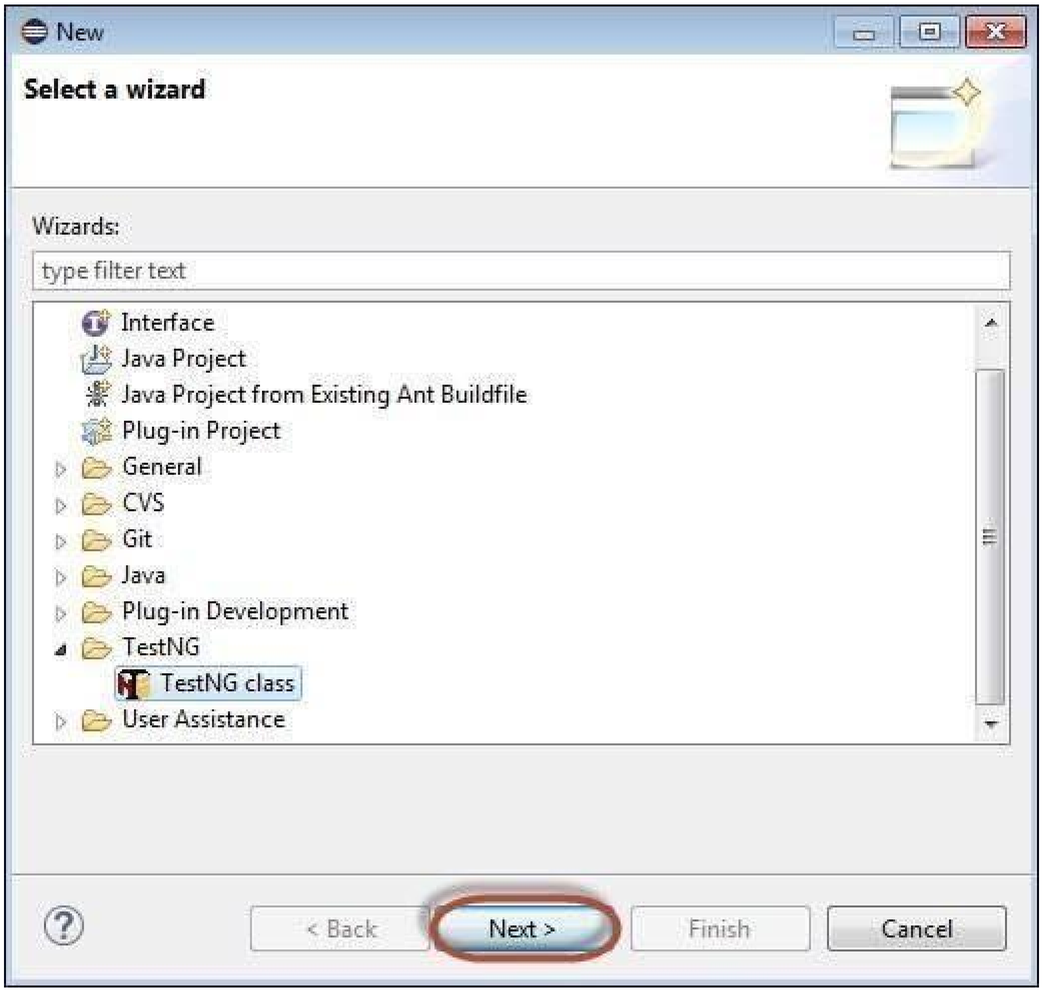


```
TestNGDemo
    src
    JRE System Library [JavaSE-1.8]
    Referenced Libraries
    TestNG
```

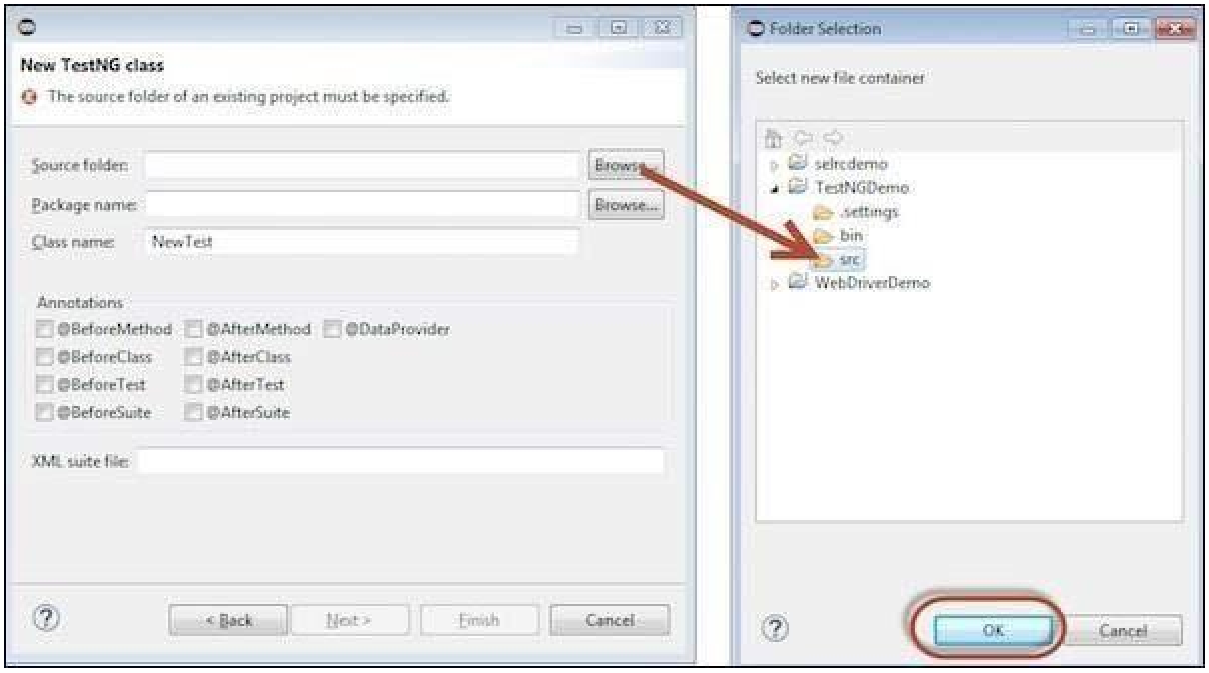
Step 8 – Right-click on 'src' folder and select New >> Other.



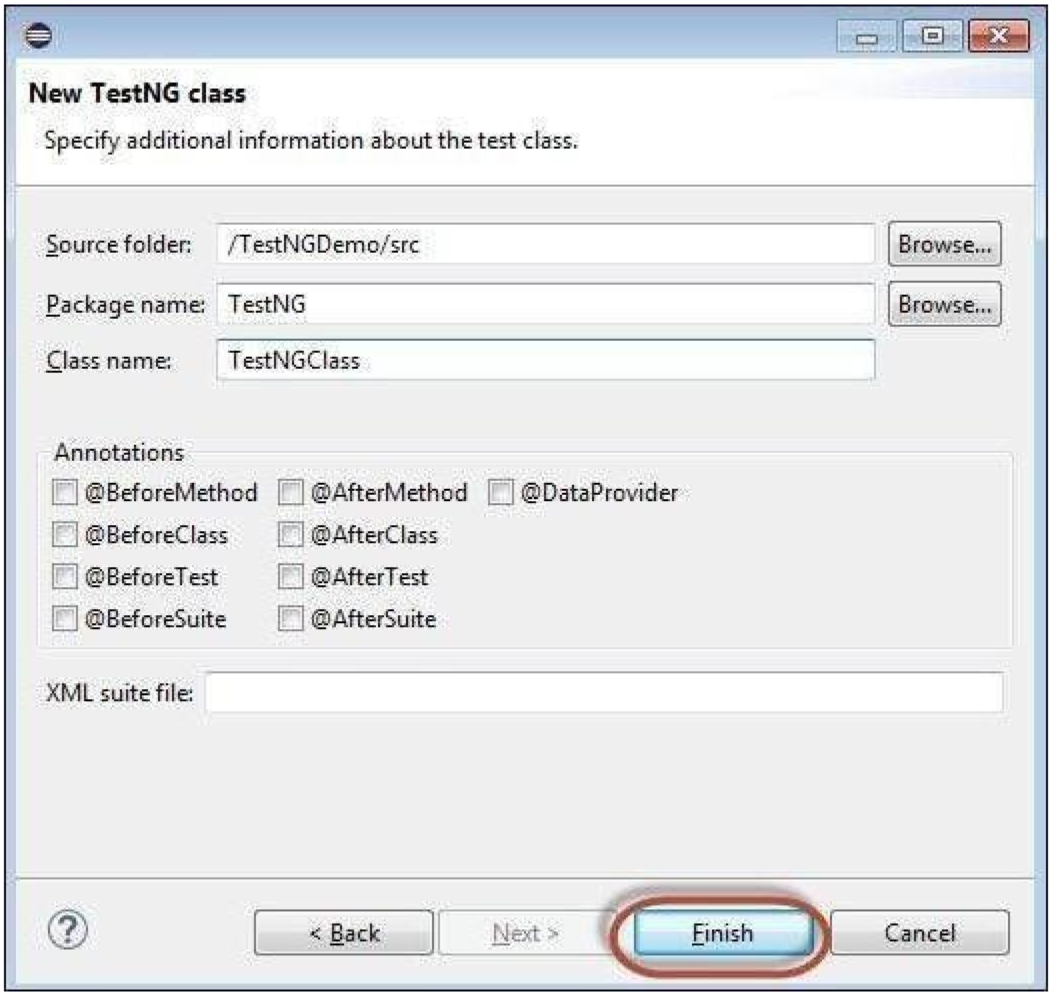
Step 9 − Select 'TestNG' and click 'Next'.



Step 10 − Select the 'Source Folder' name and click 'Ok'.



Step 11 − Select the 'Package name', the 'class name', and click 'Finish'.



Source folder: /TestNGDemo/src

Package name: TestNG

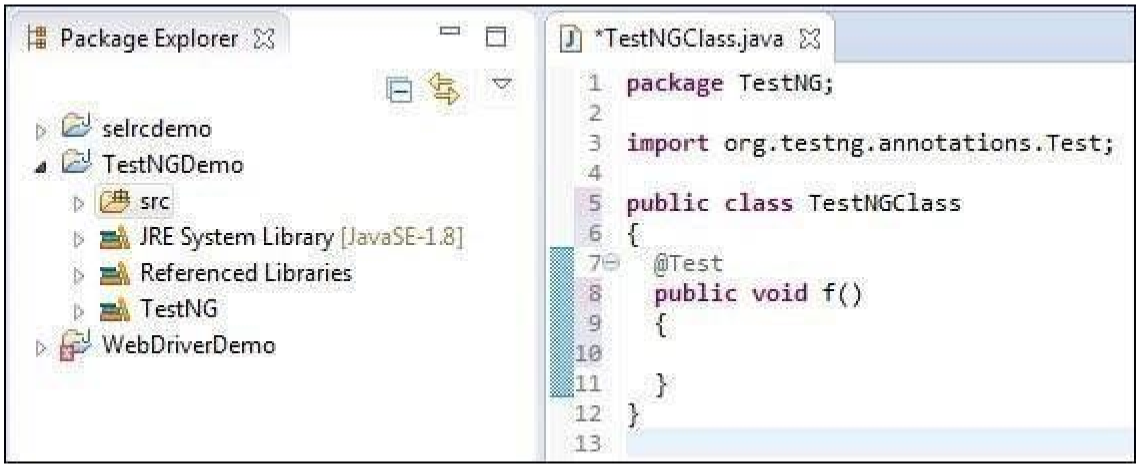
Class name: TestNGClass

Annotations

- [ ] @BeforeMethod
- [ ] @AfterMethod
- [ ] @DataProvider
- [ ] @BeforeClass
- [ ] @AfterClass
- [ ] @BeforeTest
- [ ] @AfterTest
- [ ] @BeforeSuite
- [ ] @AfterSuite

XML suite file:

Step 12 − The Package explorer and the created class would be displayed.



```
package TestNG;

import org.testng.annotations.Test;

public class TestNGClass
{
  @Test
  public void f()
  {

  }
}

```

RESULT:

EXNO : 9 DEVELOP A MINI PROJECT

a) **BUILD A DATA-DRIVEN FRAMEWORK USING SELENIUM and TestNG**
b) **BUILD PAGE OBJECT MODEL USING SELENIUM AND TestNG**
c) **BUILD BDD FRAMEWORK WITH SELENIUM,TestNG AND CUCUMBER**

AIM:

Page Object Model (POM):

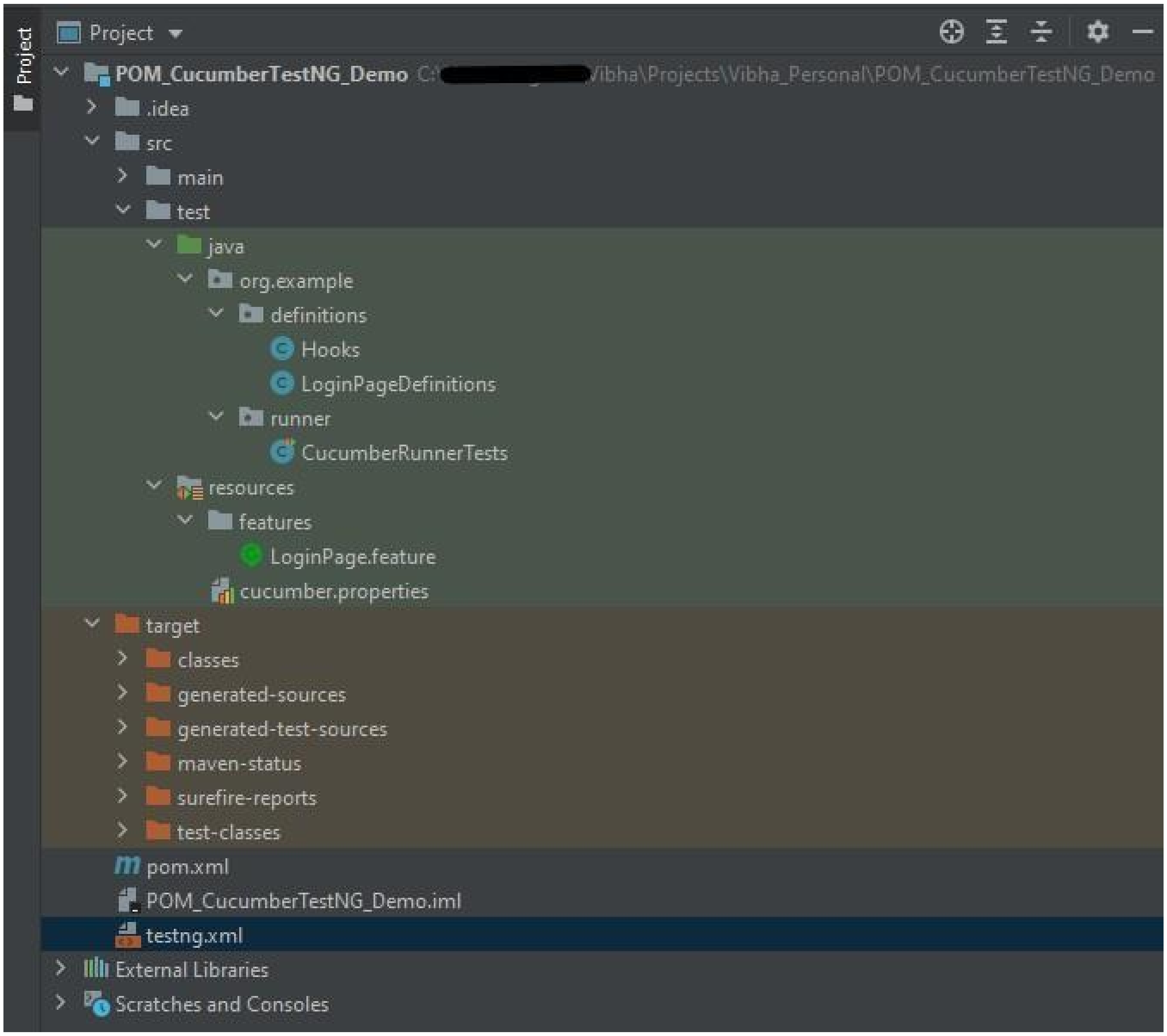
The Page Object model is an object design pattern in Selenium, where web pages are represented a classes, the various elements on the page are defined as variables in the class and all possible user interaction can then be implemented as methods in the class.

Cucumber:

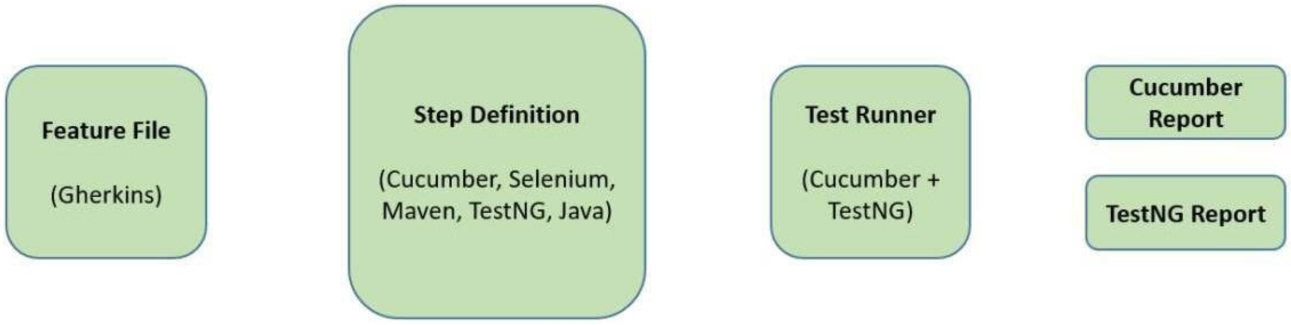
Cucumber is one such open-source tool, which supports Behavior Driven Development(BDD). In simple words, Cucumber can be defined as a testing framework, driven by plain English. It serves as documentation, automated tests, and development aid – all in one.Dependency List

1. Cucumber Java – 7.14.0
2. Cucumber TestNG – 7.14.0
3. Java 11
4. Maven – 3.8.6
5. Selenium – 4.14.0
6. TestNG – 7.8.0
7. WebDriverManager – 5.5.3

Project
Structure



```
POM_CucumberTestNG_Demo C:\[redacted]Vibha\Projects\Vibha_Personal\POM_CucumberTestNG_Demo
  .idea
  src
    main
    test
      java
        org.example
          definitions
            Hooks
            LoginPageDefinitions
          runner
            CucumberRunnerTests
      resources
        features
          LoginPage.feature
        cucumber.properties
  target
    classes
    generated-sources
    generated-test-sources
    maven-status
    surefire-reports
    test-classes
  pom.xml
  POM_CucumberTestNG_Demo.iml
  testng.xml
External Libraries
Scratches and Consoles
```



Page Object Model with Cucumber, Selenium, and TestNG

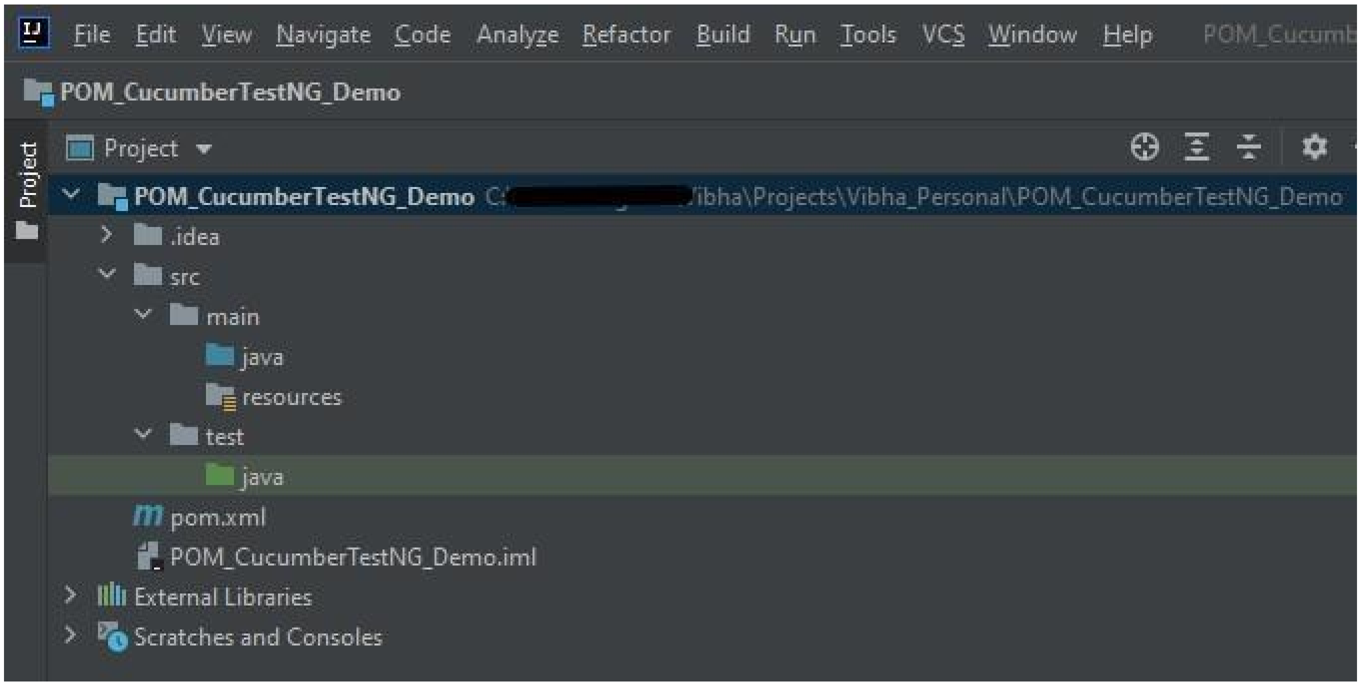
Implementation Steps

Step 1- Download and Install Java
Cucumber and Selenium need Java to be installed on the system to run the tests. Click here to learn How to install Java.

Step 2 – Setup Maven
To build a test framework, we need to add a number of dependencies to the project. Click here to learn How to install Maven.

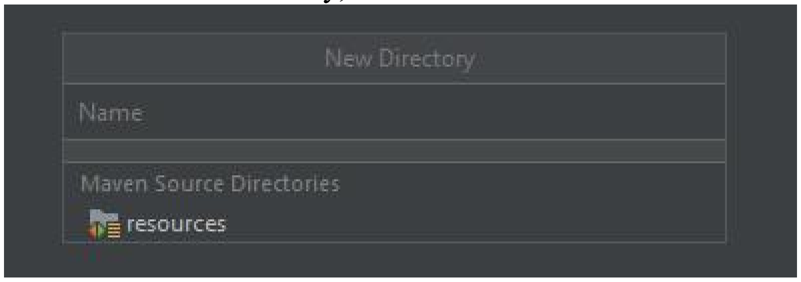
Step 3 – Install Cucumber Eclipse Plugin (Only for Eclipse)
The cucumber plugin is an Eclipse plugin that allows eclipse to understand the Gherkin syntax. When we are working with cucumber we will write the feature files that contain Feature, Scenario, Given, When, Then, And, But, Tags, Scenario Outline, and Examples. By default, eclipse doesn't understand these keywords so it doesn't show any syntax highlighter. Cucumber Eclipse Plugin highlights the keywords present in Feature File. Refer to this tutorial to get more detail – How to setup Cucumber with Eclipse.

Step 4 – Create a new Maven Project
To create a new Maven project, go to the File -> New Project-> Maven-> Maven project-> Next -> Enter Group ID & Artifact ID -> Finish.

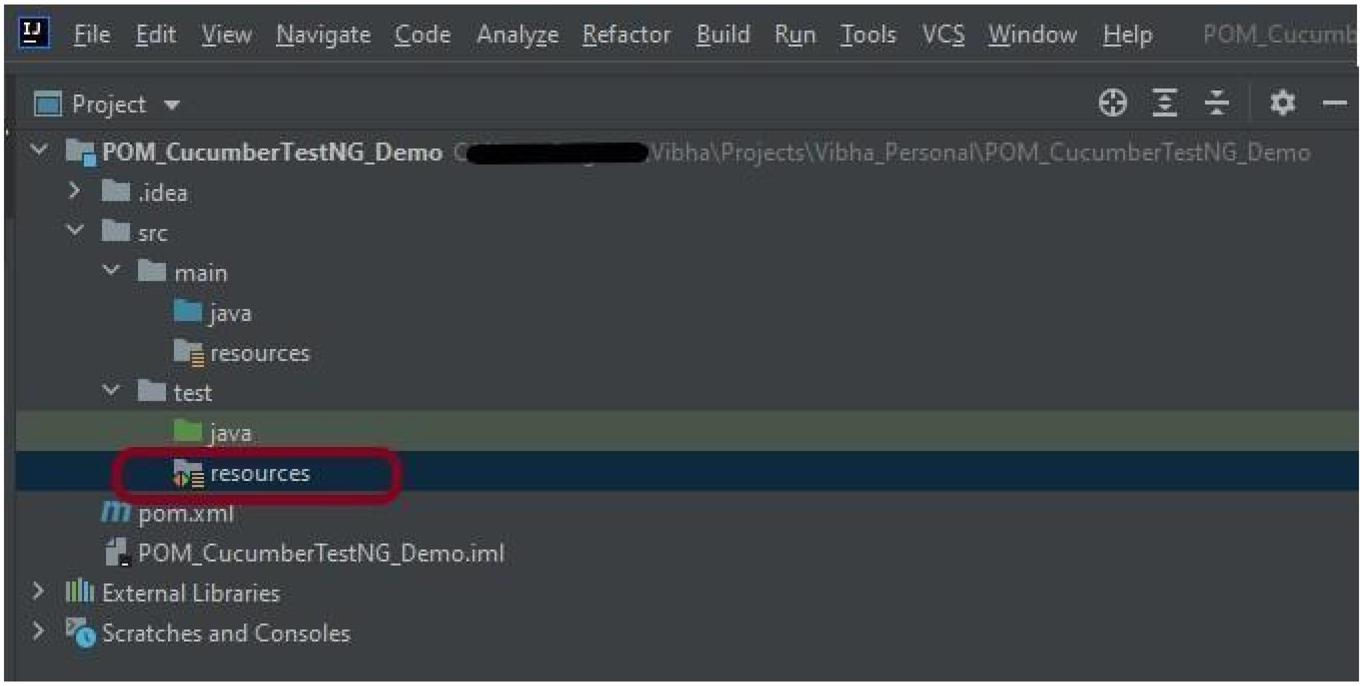

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
POM_CucumberTestNG_Demo
Project
POM_CucumberTestNG_Demo C:[redacted]ibha\Projects\Vibha_Personal\POM_CucumberTestNG_Demo
.idea
src
main
java
resources
test
java
pom.xml
POM_CucumberTestNG_Demo.iml
External Libraries
Scratches and Consoles

Click here to learn How to create a Maven project.

Step 5 – Create source folder src/test/resources to create test scenarios in the Feature file
A new Maven Project is created with 2 folders – src/main/java and src/test/java. To create test scenarios , we need a new source folder called – src/test/resources. To create this folder, right-click on test directory ->select New ->Directory, and then it shows Maven Source Directories as resources as shown below.



Double-click on the resources directory and a new source directory under your new Maven project is created as shown in the below image.


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help POM_Cucumb
Project
POM_CucumberTestNG_Demo C[redacted]Vibha\Projects\Vibha_Personal\POM_CucumberTestNG_Demo
.idea
src
main
java
resources
test
java
resources
pom.xml
POM_CucumberTestNG_Demo.iml
External Libraries
Scratches and Consoles

Step 6 – Add Selenium, TestNG, and Cucumber dependencies to the project

Add below mentioned Selenium, TestNG, and Cucumber dependencies to the project. I have added WebDriverManager dependency to the POM.xml to download the driver binaries automatically. To know more about this,
please *refer to this tutorial – How to manage driver executables using WebDriverManager.*

```
  <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <cucumber.version>7.14.0</cucumber.version>
        <selenium.version>4.14.0</selenium.version>
        <webdrivermanager.version>5.5.3</webdrivermanager.version>
        <testng.version>7.8.0</testng.version>
        <apache.common.version>2.4</apache.common.version>
        <maven.compiler.plugin.version>3.11.0</maven.compiler.plugin.version>
        <maven.surefire.plugin.version>3.1.2</maven.surefire.plugin.version>
        <maven.compiler.source.version>11</maven.compiler.source.version>
        <maven.compiler.target.version>11</maven.compiler.target.version>
    </properties>

    <dependencies>

        <dependency>
            <groupId>io.cucumber</groupId>
            <artifactId>cucumber-java</artifactId>
            <version>${cucumber.version}</version>
        </dependency>

        <dependency>
           <groupId>io.cucumber</groupId>
```

```
            <artifactId>cucumber-testng</artifactId>
            <version>${cucumber.version}</version>
            <scope>test</scope>
        </dependency>

        <!-- Selenium -->
        <dependency>
            <groupId>org.seleniumhq.selenium</groupId>
            <artifactId>selenium-java</artifactId>
            <version>${selenium.version}</version>
        </dependency>

        <!-- Web Driver Manager -->
        <dependency>
            <groupId>io.github.bonigarcia</groupId>
            <artifactId>webdrivermanager</artifactId>
            <version>${webdrivermanager.version}</version>
        </dependency>

        <!-- TestNG -->
        <dependency>
            <groupId>org.testng</groupId>
            <artifactId>testng</artifactId>
            <version>${testng.version}</version>
            <scope>test</scope>
        </dependency>

        <!-- Apache Common -->
        <dependency>
            <groupId>org.apache.directory.studio</groupId>
            <artifactId>org.apache.commons.io</artifactId>
            <version>${apache.common.version}</version>
        </dependency>

    </dependencies>
```

Step 7 – Add Maven Compiler Plugin and Surefire Plugin

The compiler plugin is used to compile the source code of a Maven project. This plugin has two goals,

which are already bound to specific phases of the default lifecycle:

- compile – compile main source files
- testCompile – compile test source files

```
<plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
```

```
                <artifactId>maven-compiler-plugin</artifactId>
                <version>${maven.compiler.plugin.version}</version>
                <configuration>
                    <source>${maven.compiler.source.version}</source>
                    <target>${maven.compiler.target.version}</target>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-surefire-plugin</artifactId>
                <version>${maven.surefire.plugin.version}</version>
                <configuration>
                <includes>
                    <include>**/*Tests.java</include>
                </includes>
                </configuration>
            </plugin>
        </plugins>
```

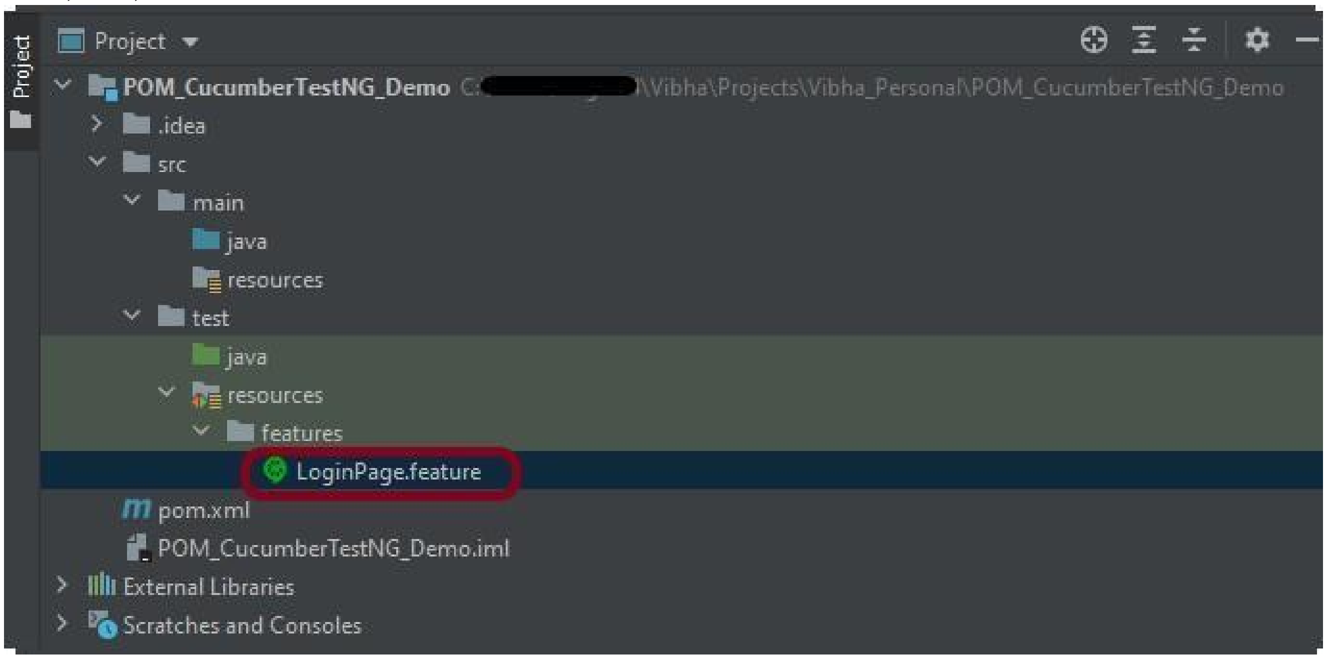
The complete POM.xml looks like as shown below

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation=
"http://maven.apache.org/POM/4.0.0https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.example</groupId>
  <artifactId>POM_CucumberTestNGDemo</artifactId>
  <version>0.0.1-SNAPSHOT</version>

<properties>
      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
      <cucumber.version>7.14.0</cucumber.version>
      <selenium.version>4.14.0</selenium.version>
      <webdrivermanager.version>5.5.3</webdrivermanager.version>
      <testng.version>7.8.0</testng.version>
      <apache.common.version>2.4</apache.common.version>
      <maven.compiler.plugin.version>3.11.0</maven.compiler.plugin.version>
      <maven.surefire.plugin.version>3.1.2</maven.surefire.plugin.version>
      <maven.compiler.source.version>11</maven.compiler.source.version>
      <maven.compiler.target.version>11</maven.compiler.target.version>
  </properties>

  <dependencies>

      <dependency>
          <groupId>io.cucumber</groupId>
          <artifactId>cucumber-java</artifactId>
```

```
        <version>${cucumber.version}</version>
    </dependency>

    <dependency>
       <groupId>io.cucumber</groupId>
       <artifactId>cucumber-testng</artifactId>
       <version>${cucumber.version}</version>
       <scope>test</scope>
   </dependency>

    <!-- Selenium -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>${selenium.version}</version>
    </dependency>

    <!-- Web Driver Manager -->
    <dependency>
        <groupId>io.github.bonigarcia</groupId>
        <artifactId>webdrivermanager</artifactId>
        <version>${webdrivermanager.version}</version>
    </dependency>

    <!-- TestNG -->
    <dependency>
        <groupId>org.testng</groupId>
        <artifactId>testng</artifactId>
        <version>${testng.version}</version>
        <scope>test</scope>
    </dependency>

    <!-- Apache Common -->
    <dependency>
        <groupId>org.apache.directory.studio</groupId>
        <artifactId>org.apache.commons.io</artifactId>
        <version>${apache.common.version}</version>
    </dependency>


</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>${maven.compiler.plugin.version}</version>
            <configuration>
              <source>${maven.compiler.source.version}</source>
              <target>${maven.compiler.target.version}</target>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>${maven.surefire.plugin.version}</version>
            <configuration>
              <suiteXmlFiles>
                <suiteXmlFile>testng.xml</suiteXmlFile>
              </suiteXmlFiles>
            </configuration>
        </plugin>
    </plugins>
</build>
</project>
```

Step 8 – Create a feature file in the src/test/resources

Create a folder with name *features*. Now, create the feature file in this folder. The feature file should be saved with the extension *.feature*. This feature file contains the test scenarios created to test the application. The Test Scenarios are written in Gherkins language in the format of *Given, When, Then, And, But*.

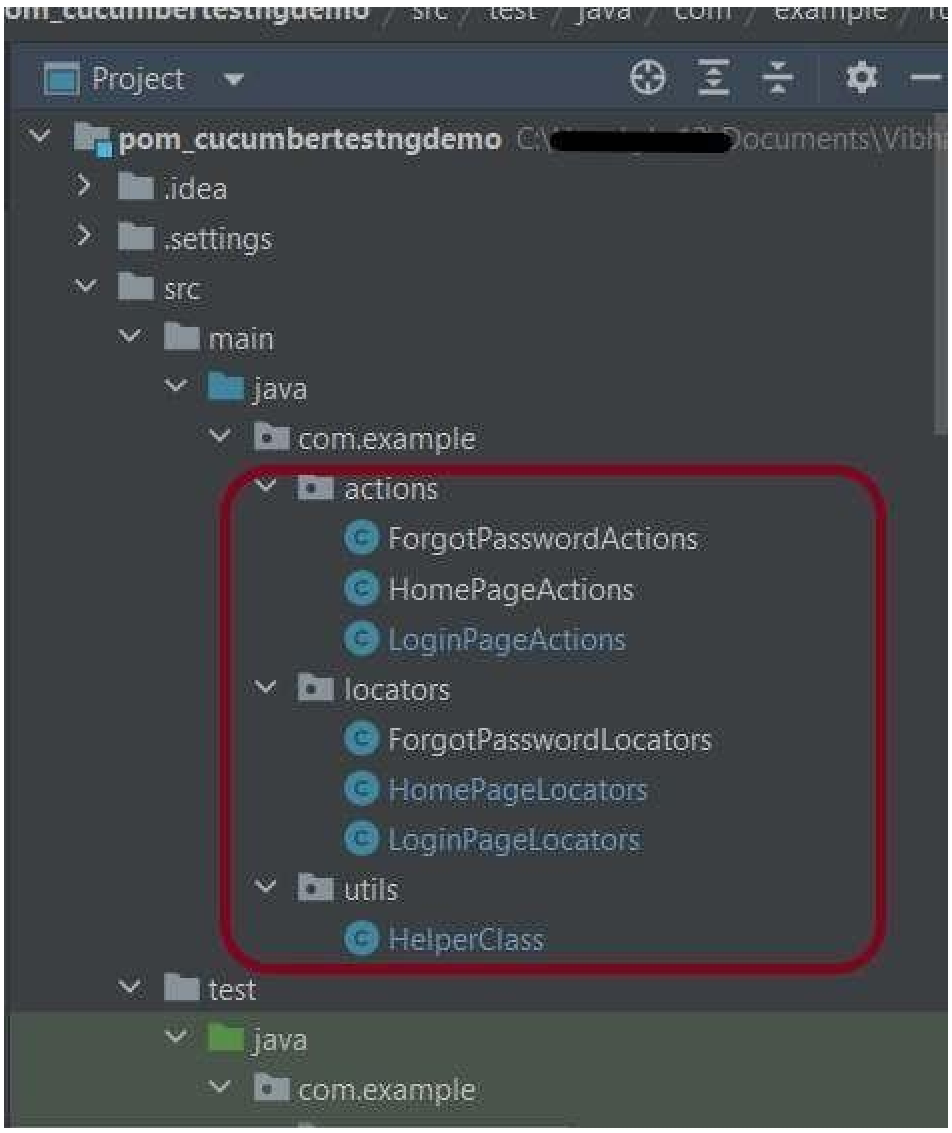


Below is an example of Test Scenarios in the feature file. I have failed one test scenario intentionally – *@MissingUsername*.

```
Feature: Login to HRM Application
```

```
Background:
  Given User is on HRMLogin page "https://opensource-demo.orangehrmlive.com/"

  @ValidCredentials
  Scenario: Login with valid credentials

   When User enters username as "Admin" and password as "admin123"
   Then User should be able to login successfully and new page open

  @InvalidCredentials
  Scenario Outline: Login with invalid credentials

   When User enters username as "<username>" and password as "<password>"
   Then User should be able to see error message "<errorMessage>"

 Examples:
 | username   | password  | errorMessage                |
 | Admin      | admin12$$ | Invalid credentials         |
 | admin$$    | admin123  | Invalid credentials         |
 | abc123     | xyz$$     | Invalid credentials         |


  @MissingUsername
  Scenario Outline: Login with blank username

   When User enters username as " " and password as "admin123"
   Then User should be able to see a message "Required1" below Username
```

Step 9 – Create the classes for locators, actions, and utilities in src/main/java
Create folders – *actions, locators, and utils* in *src/main/java.*



```
Project
pom_cucumbertestngdemo  C:\[redacted]Documents\Vibh
  .idea
  .settings
  src
    main
      java
        com.example
          actions
            ForgotPasswordActions
            HomePageActions
            LoginPageActions
          locators
            ForgotPasswordLocators
            HomePageLocators
            LoginPageLocators
          utils
            HelperClass
    test
      java
        com.example
```

Create a Java Class for each page where define WebElements as variables using Annotation *@FindBy*. Create another Java class that contains methods for actions performed on WebElements. Here, I'm going to create 2 classes for locators – *LoginPageLocators* and *HomePageLocators* as well as 2 classes for actions – *LoginPageActions* and *HomePageActions*

The *Locator* class contains WebElements which are identified by *@FindBy* annotation as shown below:-

```
@FindBy(name = "txtUsername")
WebElement userName;
```

Action class contains methods for the action to be performed on the web elements identified in the locator class.

The *initElements* is a static method of PageFactory class that is used to initialize all the web elements located by @FindBy annotation. Only after the WebElements are initialized, they can be used in the methods to perform actions.

```
public Login(WebDriver driver) {
        this.driver = driver;
        // This initElements method will create all WebElements
        PageFactory.initElements(driver, this);
```

```
    }
```

Below is the sample code of the LoginPageLocators.

```
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;

public class LoginPageLocators
{

    @FindBy(name = "username")
    public WebElement userName;

    @FindBy(name = "password")
    public WebElement password;

    @FindBy(xpath = "//*[@id='app']/div[1]/div/div[1]/div/div[2]/div[2]/form/div[1]/div/span")
    public WebElement missingUsernameErrorMessage;

    @FindBy(xpath = "//*[@id='app']/div[1]/div/div[1]/div/div[2]/div[2]/form/div[3]/button")
    public WebElement login;

    @FindBy(xpath = "//*[@id='app']/div[1]/div/div[1]/div/div[2]/div[2]/div/div[1]/div[1]/p")
    public  WebElement errorMessage;

}
```

Below is the sample code for the HomePageLocators.

```
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;

public class HomePageLocators
{

    @FindBy(xpath = "//span[@class='oxd-topbar-header-breadcrumb']/h6")
    public  WebElement homePageUserName;

}
```

Create the *action classes* for each web page. These action classes contain all the methods needed by the step definitions. In this case, I have created 2 action classes – *LoginPageActions*, *HomePageActions*

LoginPageActions

```
import org.example.locators.LoginPageLocators;
import org.example.utils.HelperClass;
import org.openqa.selenium.support.PageFactory;

public class LoginPageActions
{

    LoginPageLocators loginPageLocators = null;

    public LoginPageActions()
{
        this.loginPageLocators = new LoginPageLocators();

        PageFactory.initElements(HelperClass.getDriver(),loginPageLocators);
    }

    // Get the error message when username is blank
    public String getMissingUsernameText() {
    return loginPageLocators.missingUsernameErrorMessage.getText();
    }

    // Get the Error Message
    public String getErrorMessage() {
    return loginPageLocators.errorMessage.getText();
    }

    public void login(String strUserName, String strPassword)
{
        // Fill user name
        loginPageLocators.userName.sendKeys(strUserName);

        // Fill password
        loginPageLocators.password.sendKeys(strPassword);

        // Click Login button
        loginPageLocators.login.click();

    }
```

HomePageActions

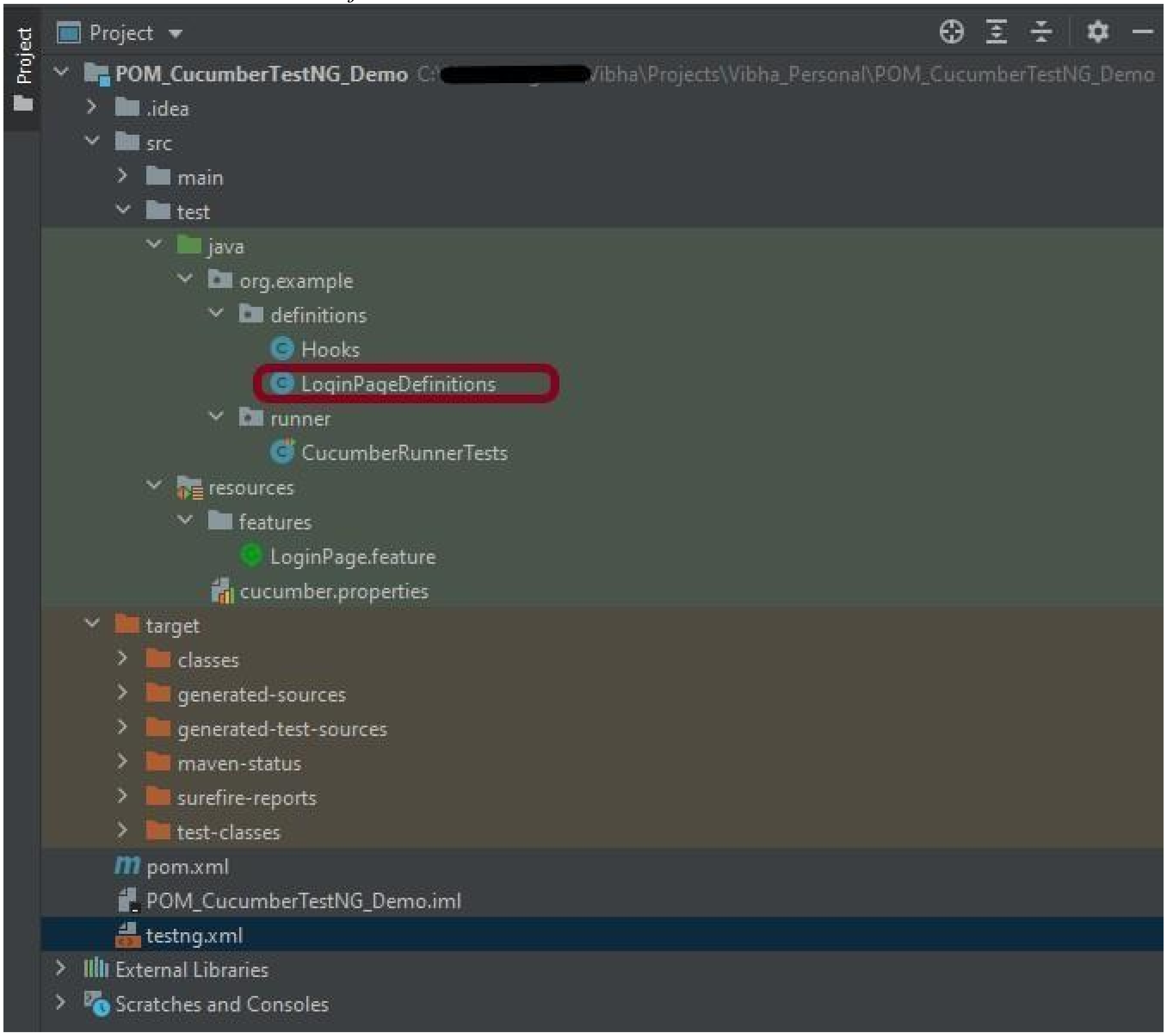
```
import org.example.locators.HomePageLocators;
import org.example.utils.HelperClass;
import org.openqa.selenium.support.PageFactory;

public class HomePageActions
{
    HomePageLocators homePageLocators = null;

    public HomePageActions()
{
        this.homePageLocators = new HomePageLocators();
    PageFactory.initElements(HelperClass.getDriver(),homePageLocators);
    }

    // Get the User name from Home Page
    public String getHomePageText() {
    return homePageLocators.homePageUserName.getText();
    }
}
```

Create a *Helper class* where we are initializing the web driver, initializing the web driver wait, defining the timeouts, and creating a private constructor of the class, it will declare the web driver, so whenever we create an object of this class, a new web browser is invoked.

```
package com.example.utils;

import java.time.Duration;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import io.github.bonigarcia.wdm.WebDriverManager;
import org.openqa.selenium.chrome.ChromeOptions;

public class HelperClass
{
    private static HelperClass helperClass;

      private static WebDriver driver;
      public final static int TIMEOUT = 5;

       private HelperClass()
{
         WebDriverManager.chromedriver().setup();
         ChromeOptions options = new ChromeOptions();
         options.addArguments("--start-maximized");
         driver = new ChromeDriver(options);
       driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(TIMEOUT));
       }

      public static void openPage(String url) {
      driver.get(url);
      }

      public static WebDriver getDriver() {
      return driver;
      }

      public static void setUpDriver()
{
         if (helperClass==null)
{
            helperClass = new HelperClass();
         }
      }

      public static void tearDown() {

            if(driver!=null)
                {driver.close()
                ;driver.quit();
            }

            helperClass = null;
        }

    }
```

Step 10 – Create a StepDefinition class in src/test/java

Create a Java Class called Definition where we will create the Test Code related to *the Given, When , Then* of Feature file in *src/test/java.*



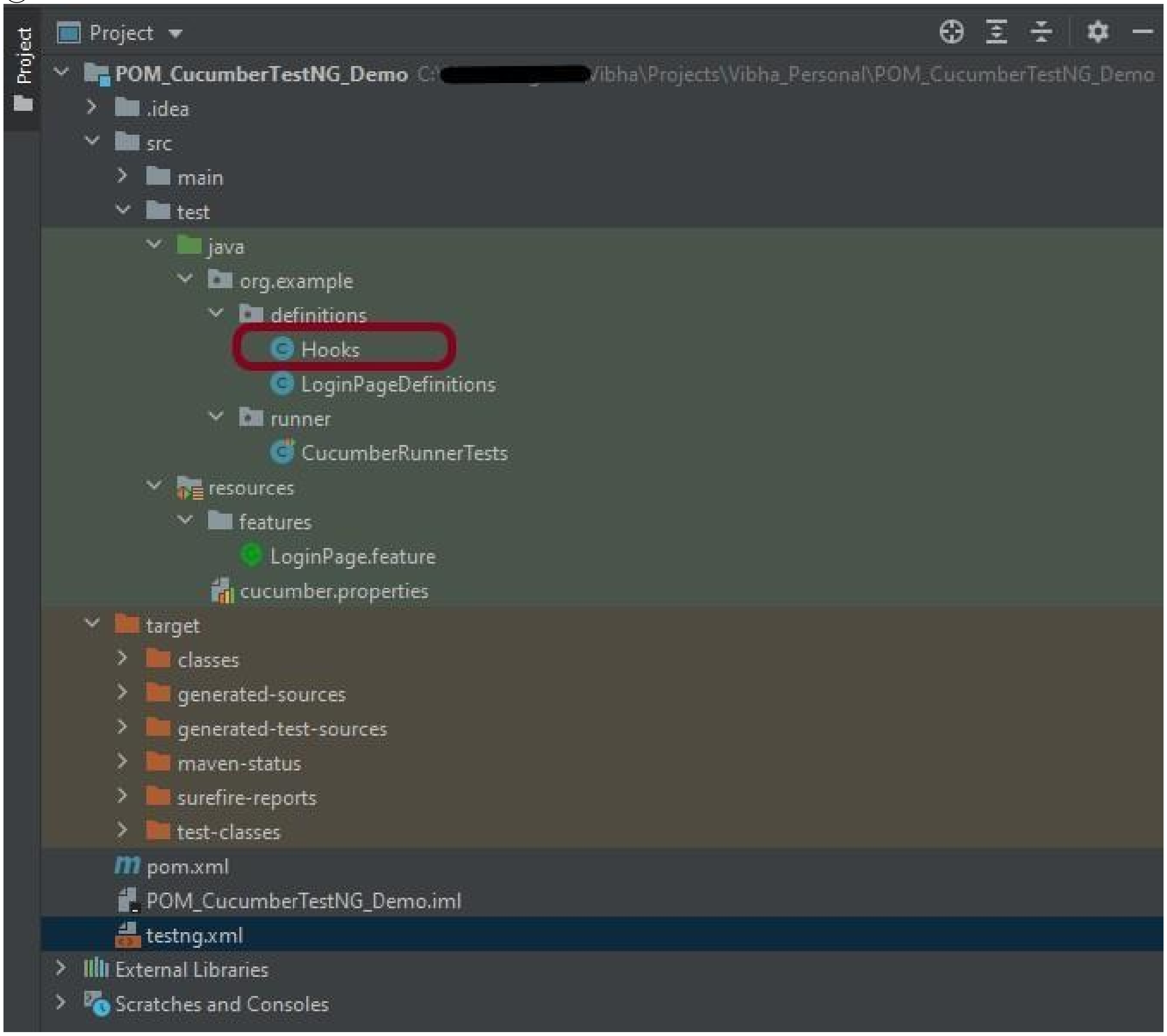
```
Project
POM_CucumberTestNG_Demo C:[redacted]Vibha\Projects\Vibha_Personal\POM_CucumberTestNG_Demo
  .idea
  src
    main
    test
      java
        org.example
          definitions
            Hooks
            LoginPageDefinitions
          runner
            CucumberRunnerTests
      resources
        features
          LoginPage.feature
        cucumber.properties
  target
    classes
    generated-sources
    generated-test-sources
    maven-status
    surefire-reports
    test-classes
  pom.xml
  POM_CucumberTestNG_Demo.iml
  testng.xml
External Libraries
Scratches and Consoles
```

Now, we need to create the Step Definition of the Feature File – *LoginPageDefinitions.java*.

```
import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
import org.example.actions.HomePageActions;
import org.example.actions.LoginPageActions;
import org.example.utils.HelperClass;
import org.testng.Assert;

public class LoginPageDefinitions
{
    LoginPageActions objLogin = new LoginPageActions();
    HomePageActions objHomePage = new HomePageActions();

    @Given("User is on HRMLogin page {string}")
    public void loginTest(String url)
    {
        HelperClass.openPage(url);

    }

    @When("User enters username as {string} and password as {string}")
    public void goToHomePage(String userName, String passWord)
    {
        // login to application
        objLogin.login(userName, passWord);

        // go the next page

    }

    @Then("User should be able to login successfully and new page open")
    public void verifyLogin()
    {
        // Verify home page
        Assert.assertTrue(objHomePage.getHomePageText().contains("Dashboard"));

    }

    @Then("User should be able to see error message {string}")
    public void verifyErrorMessage(String expectedErrorMessage)
    {
        // Verify error message
        Assert.assertEquals(objLogin.getErrorMessage(),expectedErrorMessage);

    }

    @Then("User should be able to see a message {string} below Username")
    public void verifyMissingUsernameMessage(String message) {

        Assert.assertEquals(objLogin.getMissingUsernameText(),message);
    }

}
```

Step 11 – Create a Hook class in src/test/java

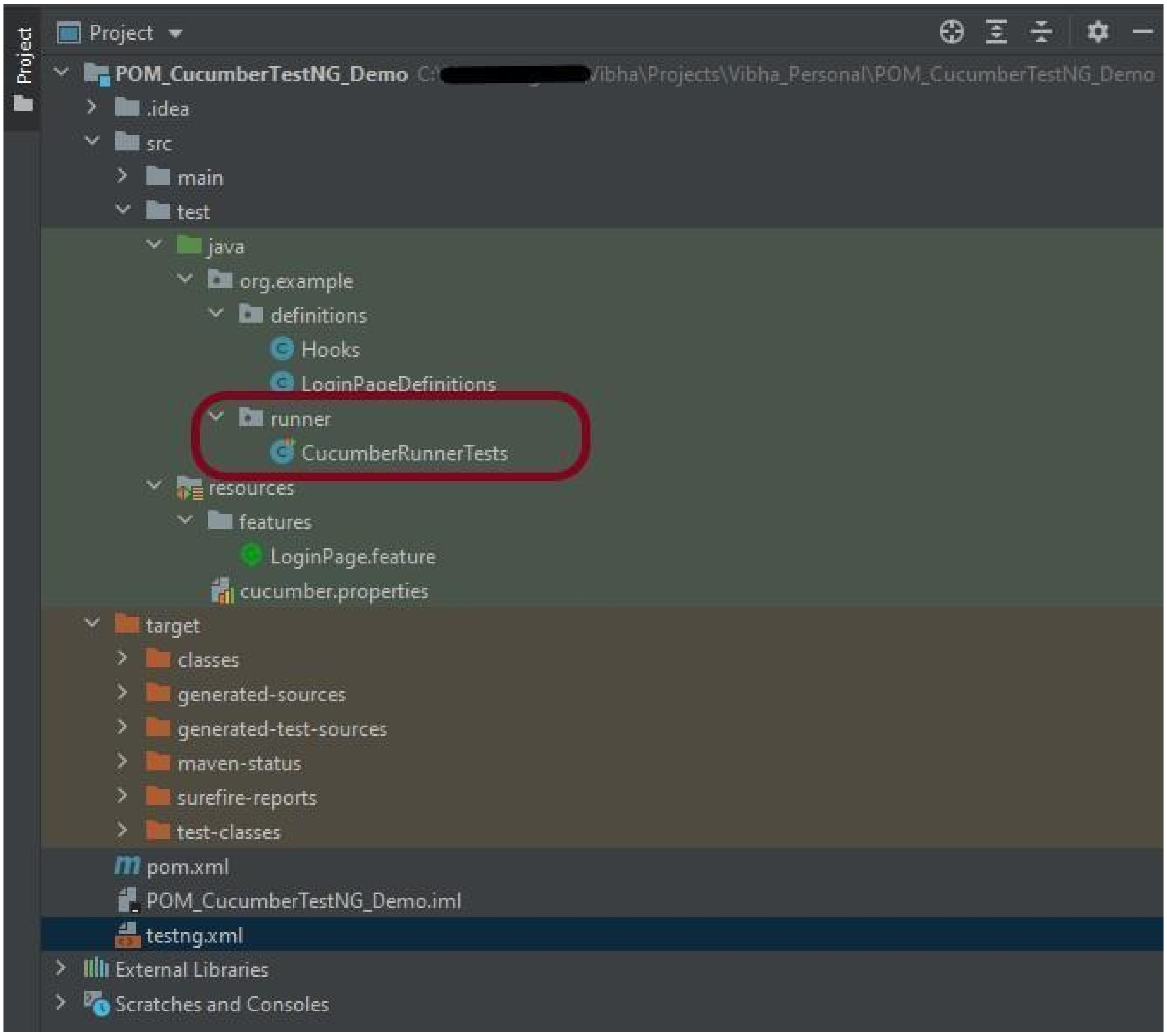
Create the *hook class* that contains the Before and After hook to initialize the web browser and close the web browser. I have added the code to take the screenshot of the failed scenario in @After Hook.



Below is the code for the Hooks class.

```
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import com.example.utils.HelperClass;
import io.cucumber.java.After;
import io.cucumber.java.Before;
import io.cucumber.java.Scenario;

public class Hooks
{

   @Before
   public static void setUp()
{

     HelperClass.setUpDriver();
   }

   @After
   public static void tearDown(Scenario scenario)
{

      //validate if scenario has failed
      if(scenario.isFailed()) {
         final byte[] screenshot = ((TakesScreenshot) HelperClass.getDriver())
.getScreenshotAs(OutputType.BYTES);
      scenario.attach(screenshot, "image/png", scenario.getName());
      }

      HelperClass.tearDown();
   }
}
```

Step 12 – Create a TestNG Cucumber Runner class in the src/test/java
Cucumber needs a *TestRunner* class to run the feature files. It is suggested to create a folder with the name of the *runner* in the *src/test/java* directory and create the Cucumber TestRunner class in this folder. Below is the code of the Cucumber TestRunner class.



Below is the code for CucumberRunnerTests class.

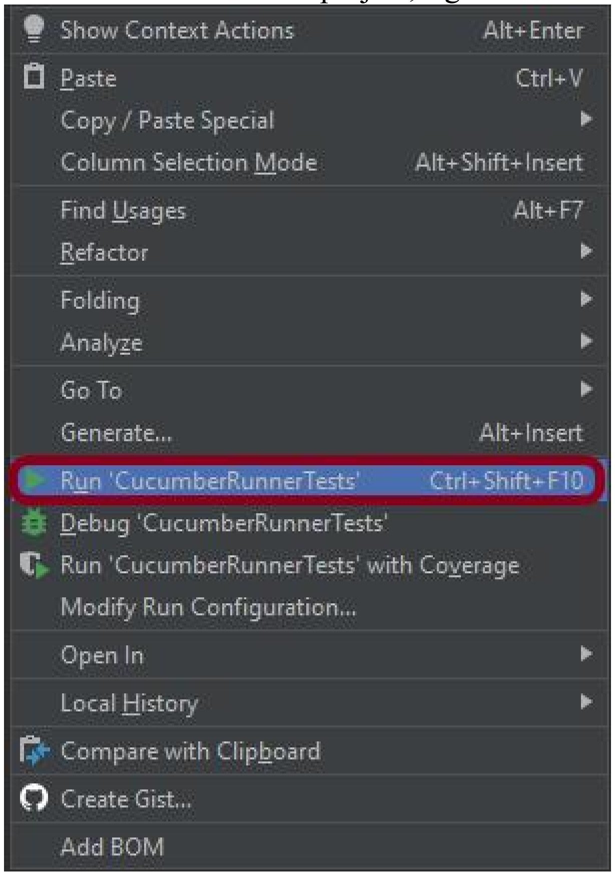
```
import io.cucumber.testng.AbstractTestNGCucumberTests;
import io.cucumber.testng.CucumberOptions;

@CucumberOptions(tags = "", features = "src/test/resources/features/LoginPage.feature"
, glue = "org.example.definitions",

        plugin = {})

public class CucumberRunnerTests extends AbstractTestNGCucumberTests {

}
```

Note:- The name of the Runner class should end with Test otherwise we can't run the tests using Command Line.

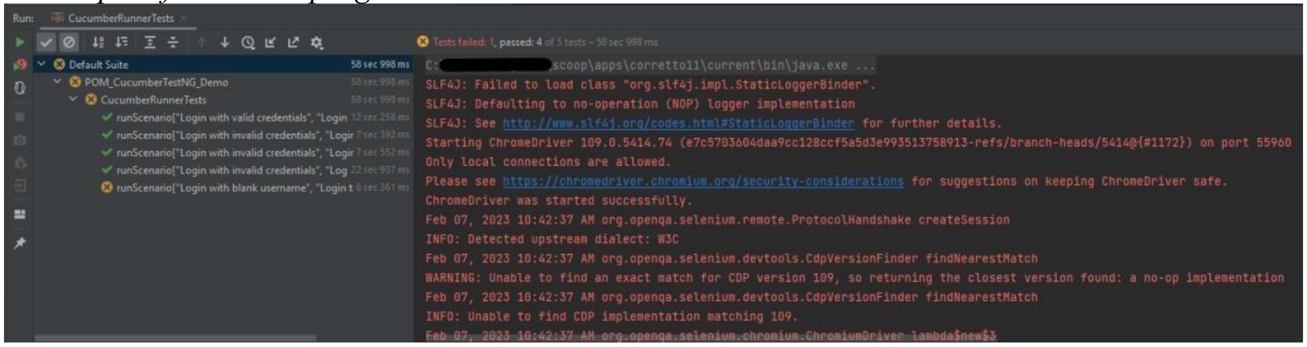
Step 13 – Run the tests from TestNG

You can execute the test script by right-clicking on *TestRunner class -> Run As TestNG*. (Eclipse)

In the case of the IntelliJ project, right-click on the runner class and select *Run 'CucumberRunnerTests'*.


Show Context Actions Alt+Enter
Paste Ctrl+V
Copy / Paste Special
Column Selection Mode Alt+Shift+Insert
Find Usages Alt+F7
Refactor
Folding
Analyze
Go To
Generate... Alt+Insert
Run 'CucumberRunnerTests' Ctrl+Shift+F10
Debug 'CucumberRunnerTests'
Run 'CucumberRunnerTests' with Coverage
Modify Run Configuration...
Open In
Local History
Compare with Clipboard
Create Gist...
Add BOM

The output of the above program is



```
Tests failed: 1, passed: 4 of 5 tests – 58 sec 998 ms

Default Suite                                              58 sec 998 ms
  POM_CucumberTestNG_Demo                                  58 sec 998 ms
    CucumberRunnerTests                                    58 sec 998 ms
      runScenario["Login with valid credentials", "Login   12 sec 258 ms
      runScenario["Login with invalid credentials", "Logir 7 sec 392 ms
      runScenario["Login with invalid credentials", "Logir 7 sec 552 ms
      runScenario["Login with invalid credentials", "Log   22 sec 937 ms
      runScenario["Login with blank username", "Login t    6 sec 361 ms
```

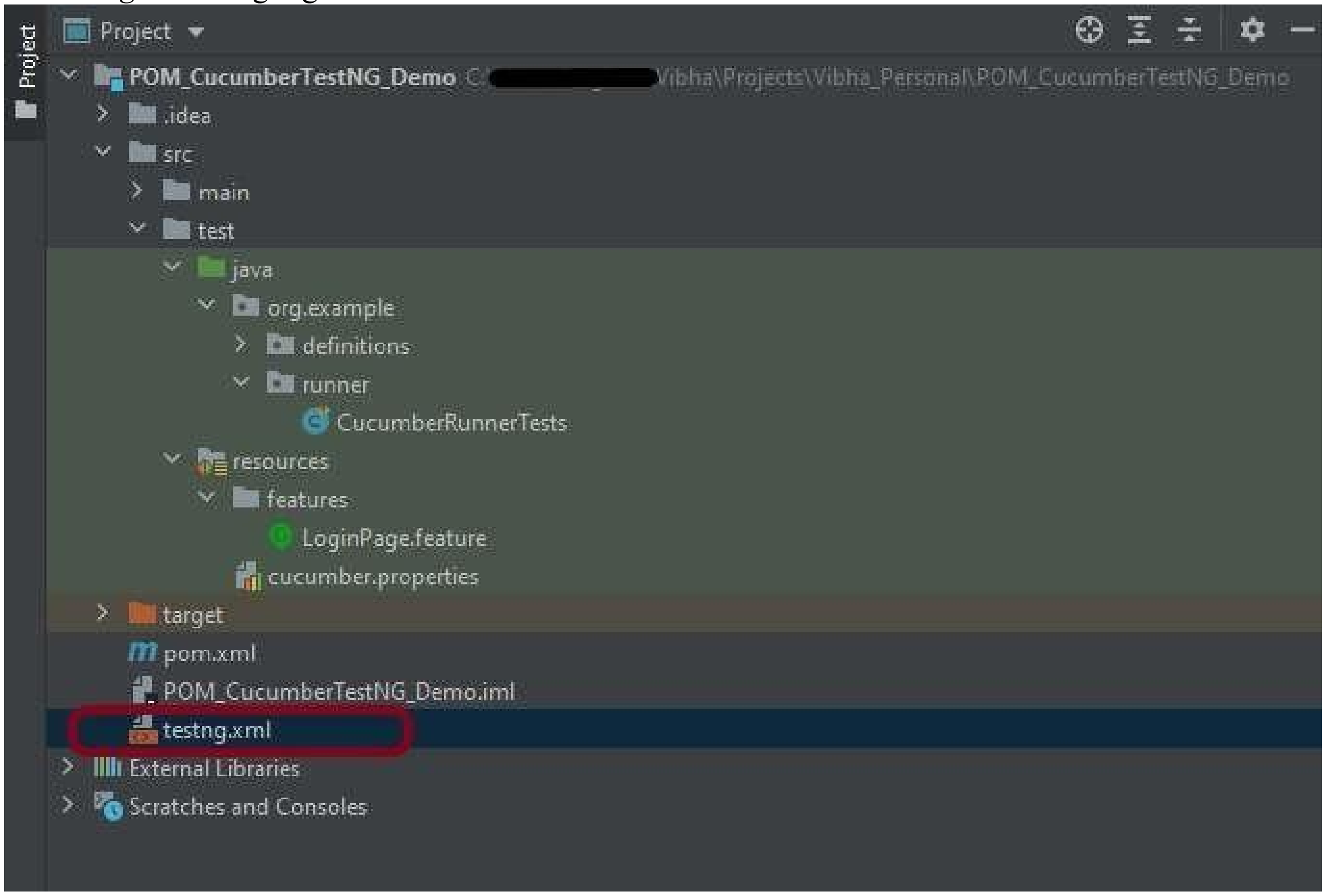
```
C:[redacted]scoop\apps\corretto11\current\bin\java.exe ...
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Starting ChromeDriver 109.0.5414.74 (e7c5703604daa9cc128ccf5a5d3e993513758913-refs/branch-heads/5414@{#1172}) on port 55960
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Feb 07, 2023 10:42:37 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected upstream dialect: W3C
Feb 07, 2023 10:42:37 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 109, so returning the closest version found: a no-op implementation
Feb 07, 2023 10:42:37 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Unable to find CDP implementation matching 109.
Feb 07, 2023 10:42:37 AM org.openqa.selenium.chromium.ChromiumDriver lambda$new$3
```

Step 14 – Run the tests from testng.xml

Create a *TestNG.xml* as shown below and run the tests as *TestNG*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
    <test  name="Cucumber with TestNG Test">
        <classes>
            <class name="org.example.runner.CucumberRunnerTests"/>
        </classes>
    </test><!-- Test -->
</suite><!-- Suite -->
```

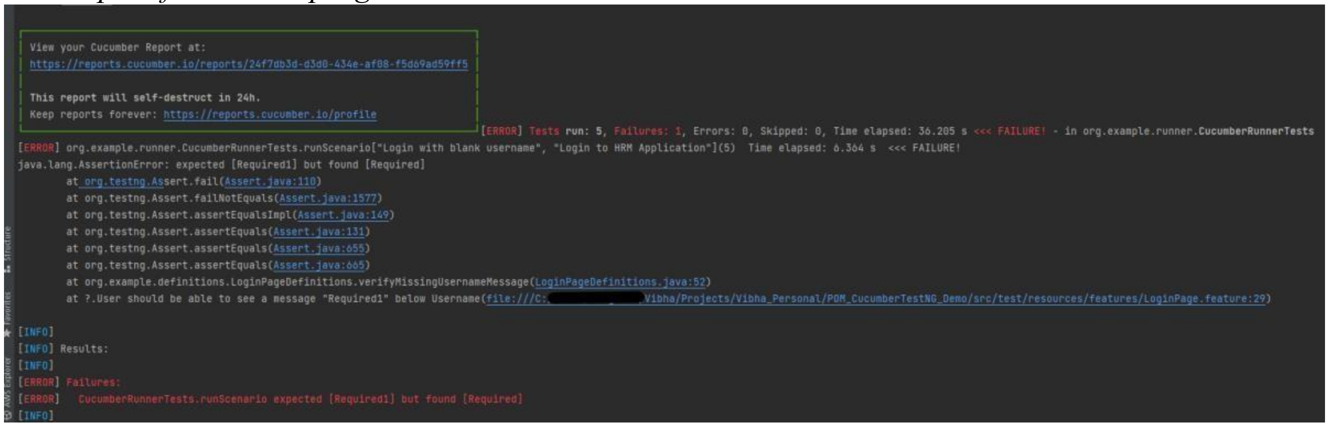
The *testng.xml* is highlighted below:



Step 15 – Run the tests from Command Line
Run the below command in the command prompt to run the tests and to get the test execution report.

```
1       mvn clean test
```

The output of the above program is

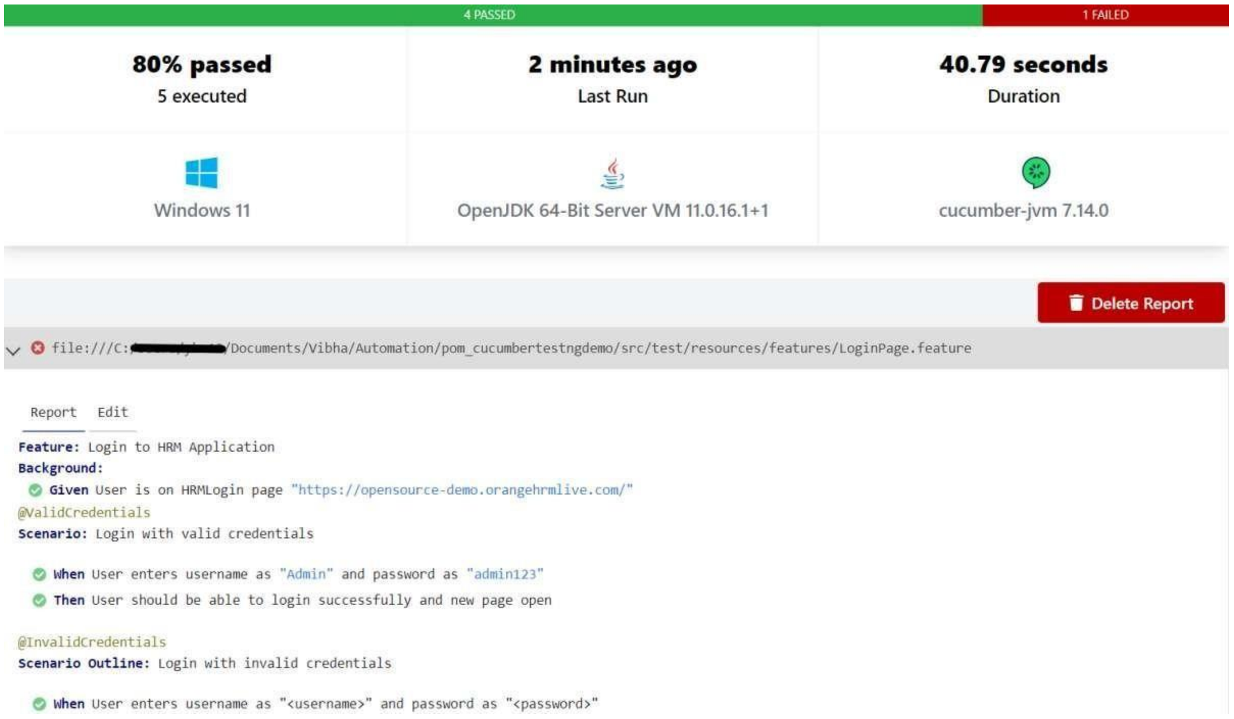


```
| View your Cucumber Report at:
| https://reports.cucumber.io/reports/24f7db3d-d3d0-434e-af08-f5d69ad59ff5
|
| This report will self-destruct in 24h.
| Keep reports forever: https://reports.cucumber.io/profile
                                                              [ERROR] Tests run: 5, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 36.205 s <<< FAILURE! - in org.example.runner.CucumberRunnerTests
[ERROR] org.example.runner.CucumberRunnerTests.runScenario["Login with blank username", "Login to HRM Application"](5)  Time elapsed: 6.364 s  <<< FAILURE!
java.lang.AssertionError: expected [Required1] but found [Required]
	at org.testng.Assert.fail(Assert.java:110)
	at org.testng.Assert.failNotEquals(Assert.java:1577)
	at org.testng.Assert.assertEqualsImpl(Assert.java:149)
	at org.testng.Assert.assertEquals(Assert.java:131)
	at org.testng.Assert.assertEquals(Assert.java:655)
	at org.testng.Assert.assertEquals(Assert.java:665)
	at org.example.definitions.LoginPageDefinitions.verifyMissingUsernameMessage(LoginPageDefinitions.java:52)
	at ?.User should be able to see a message "Required1" below Username(file:///C:[redacted]Vibha/Projects/Vibha_Personal/POM_CucumberTestNG_Demo/src/test/resources/features/LoginPage.feature:29)

[INFO]
[INFO] Results:
[INFO]
[ERROR] Failures:
[ERROR]   CucumberRunnerTests.runScenario expected [Required1] but found [Required]
[INFO]
```

Step 16 – Cucumber Report Generation
To get Cucumber Test Reports, add *cucumber.properties* under *src/test/resources* and add the below instruction in the file.

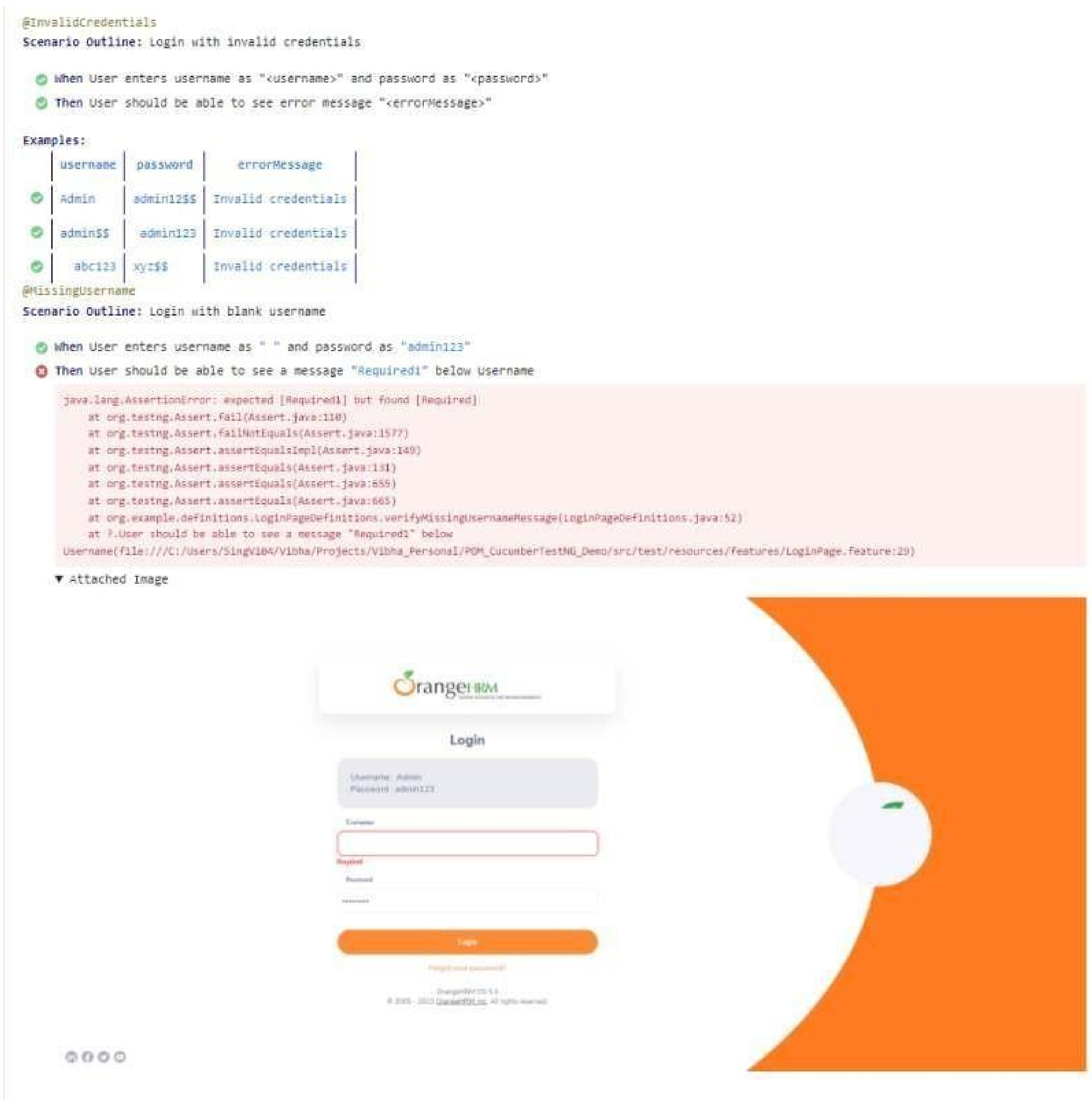
```
1       cucumber.publish.enabled=true
```

Below is the image of the Cucumber Report generated using the Cucumber Service.



```
file:///C:[redacted]/Documents/Vibha/Automation/pom_cucumbertestngdemo/src/test/resources/features/LoginPage.feature

Feature: Login to HRM Application
Background:
  Given User is on HRMLogin page "https://opensource-demo.orangehrmlive.com/"
@ValidCredentials
Scenario: Login with valid credentials

  When User enters username as "Admin" and password as "admin123"
  Then User should be able to login successfully and new page open

@InvalidCredentials
Scenario Outline: Login with invalid credentials

  When User enters username as "<username>" and password as "<password>"
```

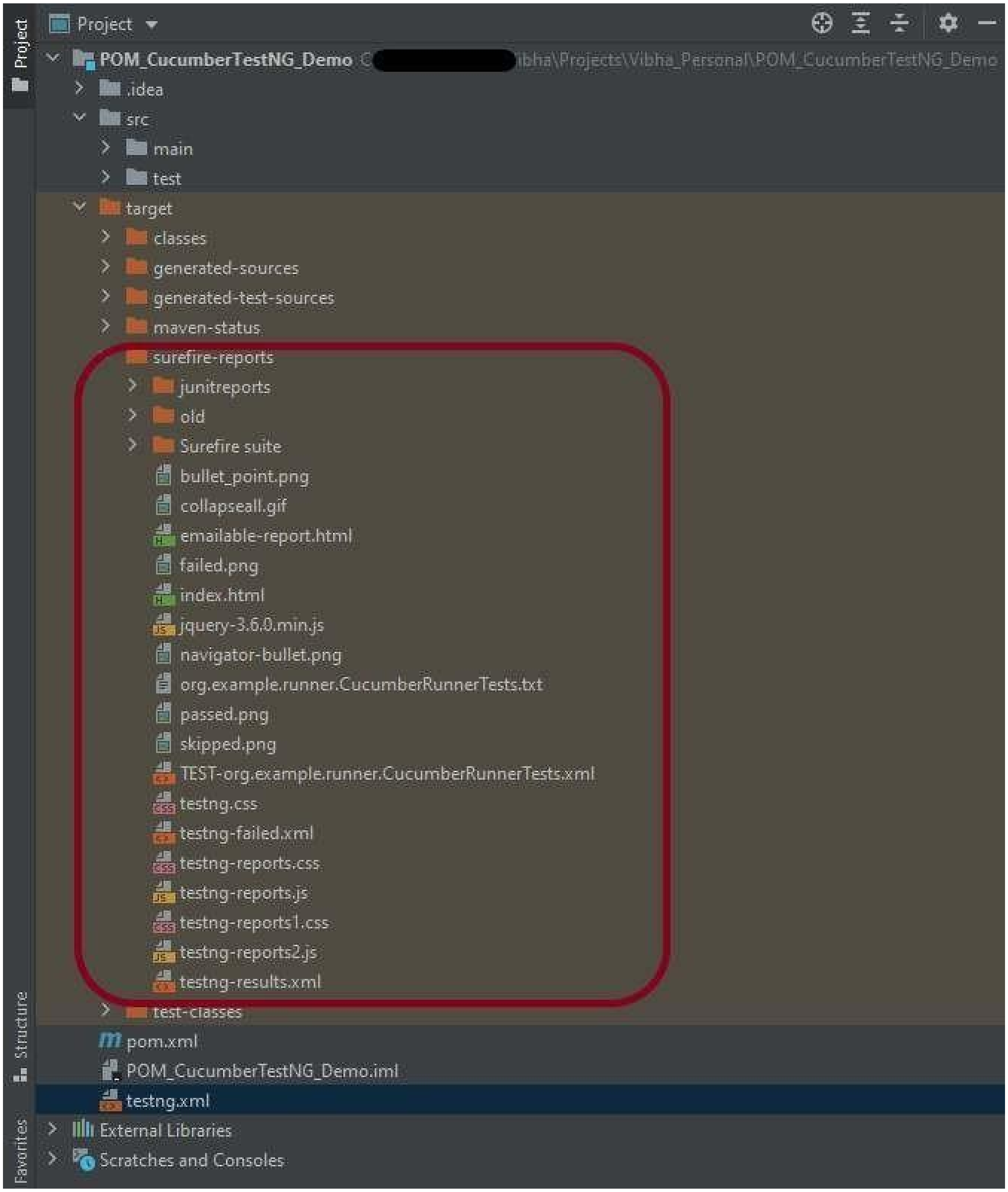
In the above example, as we can see, one of the tests has failed. So, when a test fails, we have written the code to take a screenshot of the failed step. The *Attached Image* shows the image of the failed test. You can click on that to see the screenshot.



```
@InvalidCredentials
Scenario Outline: Login with invalid credentials

  When User enters username as "<username>" and password as "<password>"
  Then User should be able to see error message "<errorMessage>"

Examples:
  | username | password   | errorMessage        |
  | Admin    | admin12$$  | Invalid credentials |
  | admin$$  | admin123   | Invalid credentials |
  | abc123   | xyz$$      | Invalid credentials |
@MissingUsername
Scenario Outline: Login with blank username

  When User enters username as " " and password as "admin123"
  Then User should be able to see a message "Required1" below Username
    java.lang.AssertionError: expected [Required1] but found [Required]
        at org.testng.Assert.fail(Assert.java:110)
        at org.testng.Assert.failNotEquals(Assert.java:1577)
        at org.testng.Assert.assertEqualsImpl(Assert.java:149)
        at org.testng.Assert.assertEquals(Assert.java:131)
        at org.testng.Assert.assertEquals(Assert.java:655)
        at org.testng.Assert.assertEquals(Assert.java:665)
        at org.example.definitions.LoginPageDefinitions.verifyMissingUsernameMessage(LoginPageDefinitions.java:52)
        at ?.User should be able to see a message "Required1" below
    Username(file:///C:/Users/SingV104/Vibha/Projects/Vibha_Personal/POM_CucumberTestNG_Demo/src/test/resources/features/LoginPage.feature:29)
  ▼ Attached Image
```

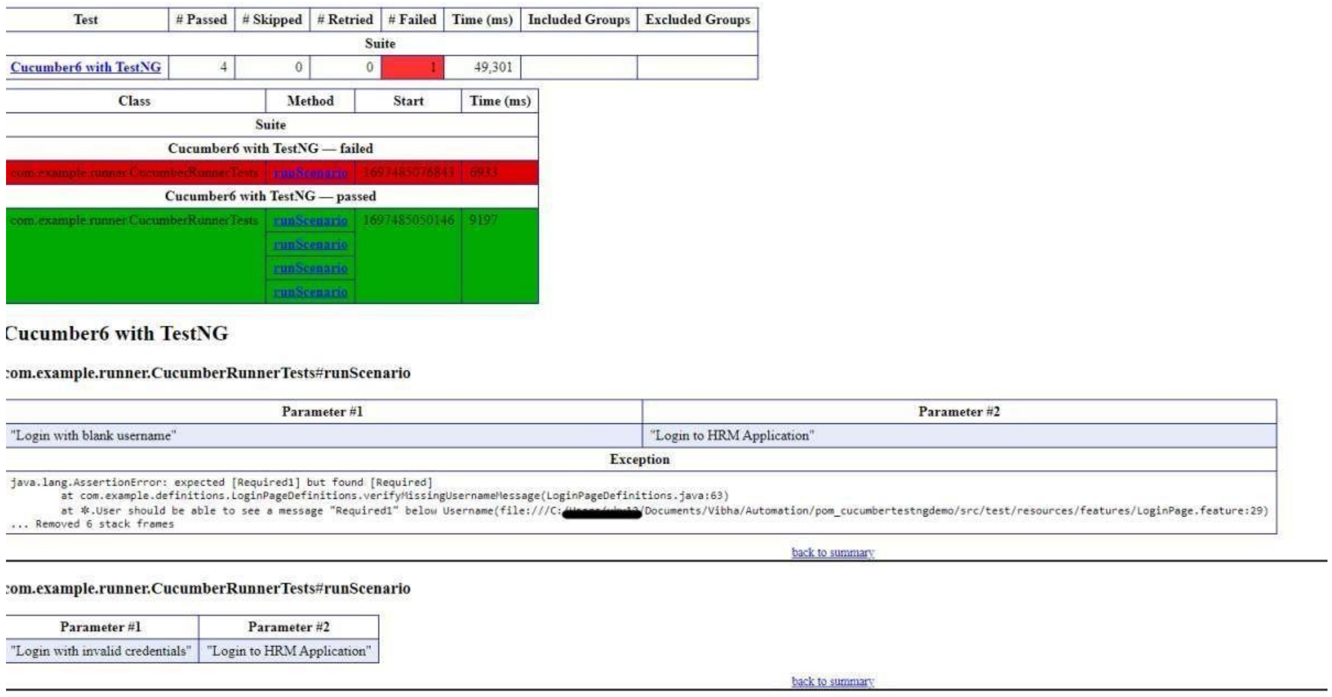
Step 17 – TestNG Report Generation

TestNG generates various types of reports under the *target->surefire-reports* folder like *emailable-report.html, index.html, testng-results.xml.*



```
Project ▾
⌄ POM_CucumberTestNG_Demo C[redacted]ibha\Projects\Vibha_Personal\POM_CucumberTestNG_Demo
  > .idea
  ⌄ src
    > main
    > test
  ⌄ target
    > classes
    > generated-sources
    > generated-test-sources
    > maven-status
      surefire-reports
      > junitreports
      > old
      > Surefire suite
        bullet_point.png
        collapseall.gif
        emailable-report.html
        failed.png
        index.html
        jquery-3.6.0.min.js
        navigator-bullet.png
        org.example.runner.CucumberRunnerTests.txt
        passed.png
        skipped.png
        TEST-org.example.runner.CucumberRunnerTests.xml
        testng.css
        testng-failed.xml
        testng-reports.css
        testng-reports.js
        testng-reports1.css
        testng-reports2.js
        testng-results.xml
    > test-classes
    pom.xml
    POM_CucumberTestNG_Demo.iml
    testng.xml
> External Libraries
> Scratches and Consoles
```

We are interested in the "*emailable-report.html*" report. Open "*emailable-report.html*", as this is an HTML report, and open it with the browser. The below image shows emailable-report.html.
emailable-report.html



Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Suite							
Cucumber6 with TestNG	4	0	0	1	49,301		

Class	Method	Start	Time (ms)
Suite			
Cucumber6 with TestNG — failed			
com.example.runner.CucumberRunnerTests	runScenario	1697485076843	6933
Cucumber6 with TestNG — passed			
com.example.runner.CucumberRunnerTests	runScenario	1697485050146	9197
	runScenario		
	runScenario		
	runScenario		

Cucumber6 with TestNG

com.example.runner.CucumberRunnerTests#runScenario

Parameter #1	Parameter #2
"Login with blank username"	"Login to HRM Application"
Exception	

```
java.lang.AssertionError: expected [Required1] but found [Required]
        at com.example.definitions.LoginPageDefinitions.verifyMissingUsernameMessage(LoginPageDefinitions.java:63)
        at ✻.User should be able to see a message "Required1" below Username(file:///C:/[redacted]/Documents/Vibha/Automation/pom_cucumbertestngdemo/src/test/resources/features/LoginPage.feature:29)
... Removed 6 stack frames
```

back to summary

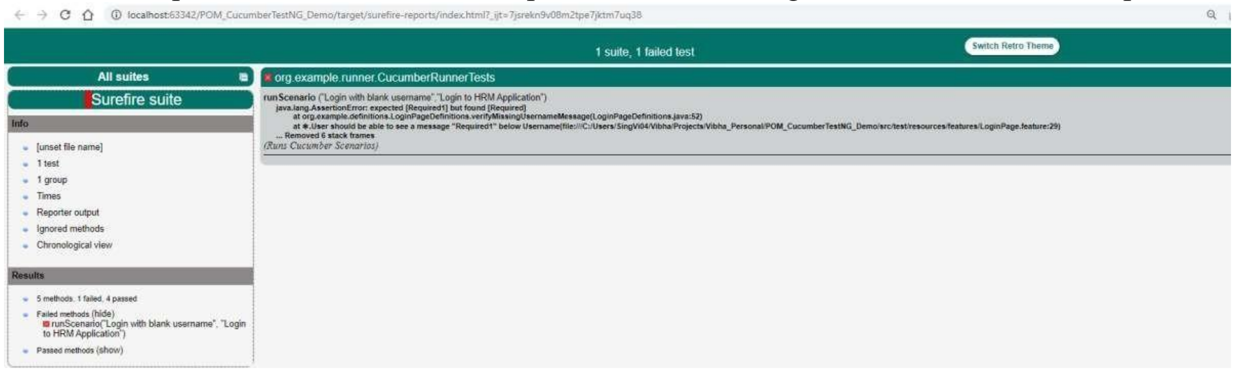
com.example.runner.CucumberRunnerTests#runScenario

Parameter #1	Parameter #2
"Login with invalid credentials"	"Login to HRM Application"

back to summary

Index.html

TestNG also produces an "*index.html*" report. The below image shows the *index.html* report.



localhost:63342/POM_CucumberTestNG_Demo/target/surefire-reports/index.html?_ijt=7jsrekn9v08m2tpe7jktm7uq38

1 suite, 1 failed test

- 5 methods, 1 failed, 4 passed
- Failed methods (hide)
  runScenario("Login with blank username", "Login to HRM Application")
- Passed methods (show)

org.example.runner.CucumberRunnerTests

```
runScenario ("Login with blank username","Login to HRM Application")
    java.lang.AssertionError: expected [Required1] but found [Required]
        at org.example.definitions.LoginPageDefinitions.verifyMissingUsernameMessage(LoginPageDefinitions.java:52)
        at ✻.User should be able to see a message "Required1" below Username(file:///C:/Users/SingVi04/Vibha/Projects/Vibha_Personal/POM_CucumberTestNG_Demo/src/test/resources/features/LoginPage.feature:29)
    ... Removed 6 stack frames
(Runs Cucumber Scenarios)
```

RESULT: