RAMAIAH
Institute of Technology

Department of Master of Computer Applications

Software Testing (MCAE53)
Laboratory Exercises

Submitted by:

PANKAJ TERON

1MS23MC068

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU)

Accredited by National Board of Accreditation & NAAC with ‘A+’ Grade

MSR Nagar, MSRIT Post, Bangalore-560054

www.msrit.edu

2025

1. Problem Description for the Application Date: 06/11/2024

Problem Statement

Online shopping platforms often face issues like poor user experience, inefficient search and filtering, and complex checkout processes leading to cart abandonment. Many systems lack seamless integration with secure payment gateways and struggle with scalability. This project aims to resolve these challenges by offering a **user-friendly, scalable, and secure** e-commerce solution.

Need for the Project

- **Improved User Experience:** Traditional platforms have complex navigation and inefficient product discovery mechanisms.
- **Optimized Checkout Process:** Many users abandon their carts due to long and complicated checkout procedures.
- **Enhanced Security:** Secure handling of transactions and user data is essential.
- **Scalability:** The system should be able to handle large traffic volumes and product catalogs.
- **Mobile Responsiveness:** Ensuring a seamless shopping experience on both mobile and desktop devices.

2. Requirements Specification Date: 07/11/2024

System Requirements

1. Software Requirements

- **Operating System**: Windows, Linux, or macOS (based on deployment)
- **Database Management System**: MySQL or Oracle Database
- **Programming Languages**: Java, PHP,
- **Web Technologies**: HTML, CSS, JavaScript for front-end development
- **Frameworks**: Spring Framework (for Java), Laravel (for PHP)
- **Software Tools**: IDEs like Eclipse, Visual Studio; Database Management tools like MySQL Workbench
- **Web Server**: Apache Tomcat, Wamp, or Xamp.
- **Cloud Services**: (Optional) AWS, Google Cloud, or Azure for cloud hosting and scalability.

2. Hardware Requirements

- **Server**:
  - Processor: Minimum Intel i5 or AMD Ryzen 5
  - RAM: 8 GB or more
  - Storage: Minimum 500 GB SSD (based on the number of rooms and data volume)
  - Network: High-speed internet connection for data transfer and cloud access
- **Client Computers**: Desktop or laptop computers with basic specifications (e.g., Intel i3 or equivalent, 4 GB RAM) for the staff to access the system.

3. User Requirements

- **Customers:** Easy product search, secure checkout.
- **Admins:** Order and inventory management.
- **Developers:** Scalable and maintainable architecture.

4. Functional Requirements

- **User Authentication:** Registration, Login, Forgot Password.
- **Product Search & Filters:** Category-wise filtering and keyword-based search.
- **Shopping Cart Management:** Add/remove items, apply coupons, update quantities.
- **Checkout and Payment Gateway Integration:** Multiple secure payment options.
- **Order Tracking:** Live order status updates and estimated delivery time.

5. Non-Functional Requirements

- **Performance:** Page response time under 2 seconds.
- **Security:** Encrypted transactions using SSL for secure data transmission.
- **Scalability:** Supports increasing user traffic and large product catalogs.
- **Reliability:** Available 24/7 with minimal downtime.

3. Preparation of Test Scenarios — Date: 13/11/2024

Test Scenario 1: User Authentication

- **Can users register successfully with valid credentials?**
  – Yes, users can successfully register with valid credentials.
- **What happens when users enter incorrect login details?**
  – No, users entering incorrect login details will see an error notification.
- **Does "Forgot Password" functionality work?**

   – Yes, the "Forgot Password" functionality works, and users receive a password reset link.

Test Scenario 2: Product Search & Filtering

- **Can users find products using the search bar?**
  – Yes, users can find products using the search bar.
- **Are category filters working correctly?**
  – Yes, category filters are working correctly.

Test Scenario 3: Shopping Cart & Checkout

1. **Can users add/remove items from the cart?**
   – Yes, users can add and remove items from the cart.
2. **Does the system update the cart total dynamically?**
   – Yes, the system updates the cart total dynamically.
3. **Can users apply discount coupons?**
   – Yes, users can apply discount coupons.
4. **Can users complete the checkout process successfully?**
   – Yes, users can complete the checkout process successfully.

Test Scenario 4: Order Management & Tracking

- **Can users view past and ongoing orders?**
  – Yes, users can view past and ongoing orders.
- **Are order tracking details updated in real-time?**
  – Yes, order tracking details are updated in real-time.

Test Scenario 5: Mobile Responsiveness

- **Does the website function correctly on mobile devices?**
  – Yes, the website functions correctly on mobile devices.
- **Does the layout adjust properly on smaller screens?**
  – Yes, the layout adjusts properly on smaller screens.

4. Preparation of Test Cases

Date: 14/11/2024

4.1 User Sign-Up Module

Test Case ID	Test Steps	Test Data	Expected Results	Actual Results	Status
TC1	Enter valid username, email, and password	Username: John, Email: johndoe@gmail.com, Password: Test@123	Redirect to login page	Redirected successfully	Pass
TC2	Enter invalid email format	Username: John, Email: john#gmail.com, Password: John@123	Error message: "Invalid email format"	Displayed correctly	Pass
TC3	Leave password field empty	Username: John, Email: john@gmail.com, Password:	Alert message: "Please fill out this field"	Displayed correctly	Pass

4. Preparation of Test Cases

Date: 14/11/2024

4.2 User Login Module

Test Case ID	Test Steps	Test Data	Expected Results	Actual Results	Status
TC4	Enter valid email and password	Email: johndeo@gmail.com, Password: Test@123	Redirect to homepage	Redirected successfully	Pass
TC5	Enter incorrect password	Email: john@gmail.com, Password: wrongPass	Error message: "Invalid login credentials"	Displayed correctly	Pass
TC6	Leave both fields empty	Email: , Password:	Alert: "Please fill out all fields"	Displayed correctly	Pass

4.3 Shopping Cart Module

Test Case ID	Test Steps	Test Data	Expected Results	Actual Results	Status
TC7	Add product to cart	Product: Laptop	Cart updated successfully	Updated correctly	Pass

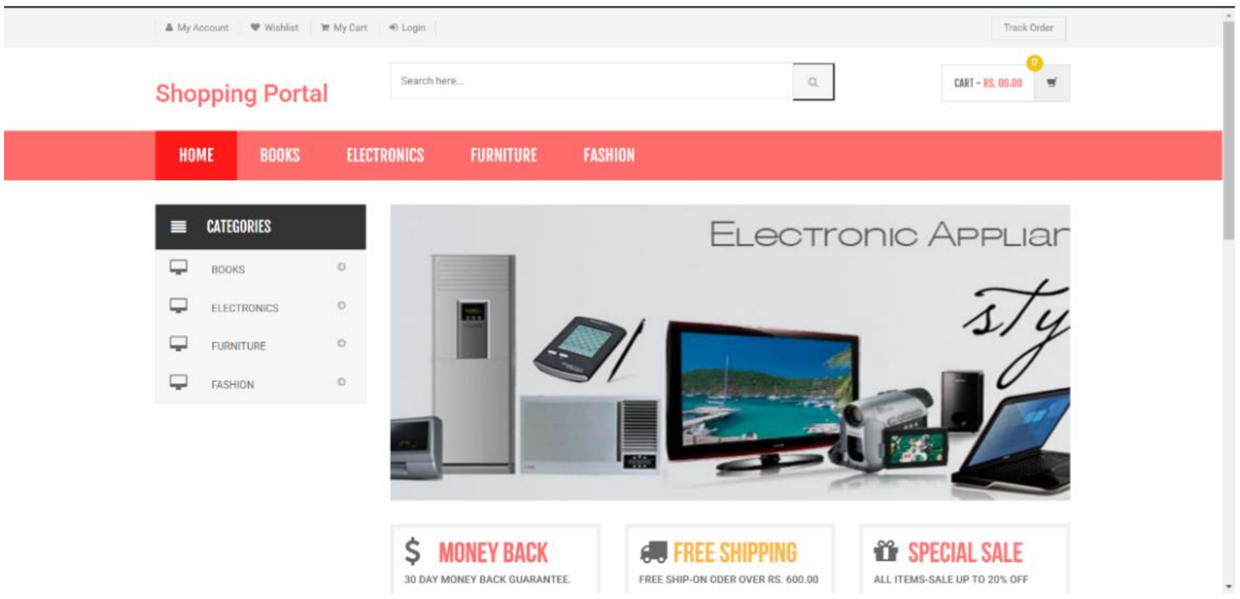
Test Case ID	Test Steps	Test Data	Expected Results	Actual Results	Status
TC8	Remove product from cart	Product: Laptop	Product removed successfully	Removed correctly	Pass
TC9	Checkout without adding product	No product in cart	Error: "Cart is empty"	Displayed correctly	Pass

4.4 Order Placement Module

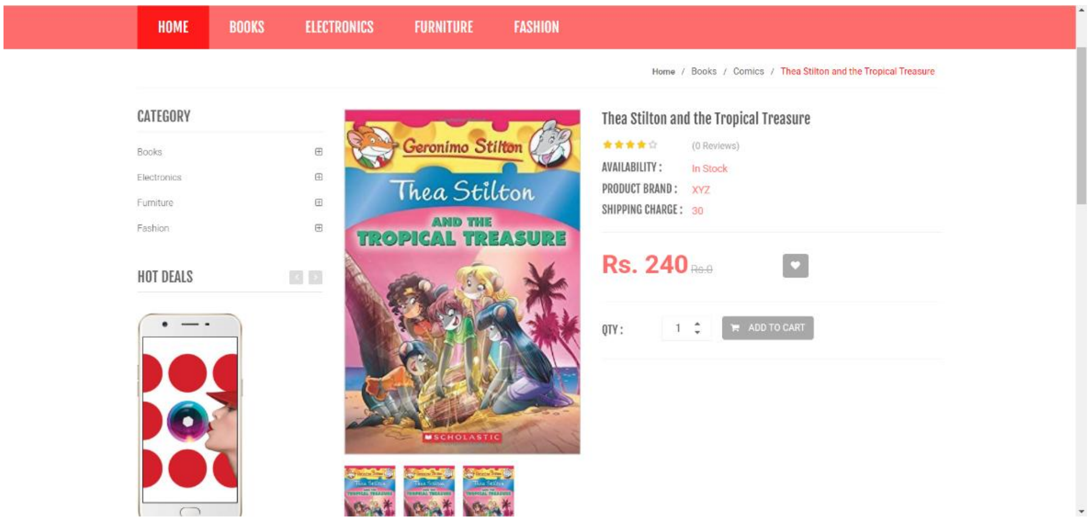
Test Case ID	Test Steps	Test Data	Expected Results	Actual Results	Status
TC10	Select COD as payment method	Click on COD and proceed	Product added to order list successfully	Product added	Pass
TC11	Select Card Payment	Click on Card and proceed	Product added to order list successfully	Product added	Pass
TC12	Select UPI Payment	Click on UPI and proceed	Product added to order list successfully	Product added	Pass

6. Development of the Application   Date: 27/11/2024

Home Page



Product



Home / Books / Comics / Thea Stilton and the Tropical Treasure

CATEGORY

Books
Electronics
Furniture
Fashion

HOT DEALS

Thea Stilton and the Tropical Treasure

(0 Reviews)

AVAILABILITY : In Stock
PRODUCT BRAND : XYZ
SHIPPING CHARGE : 30

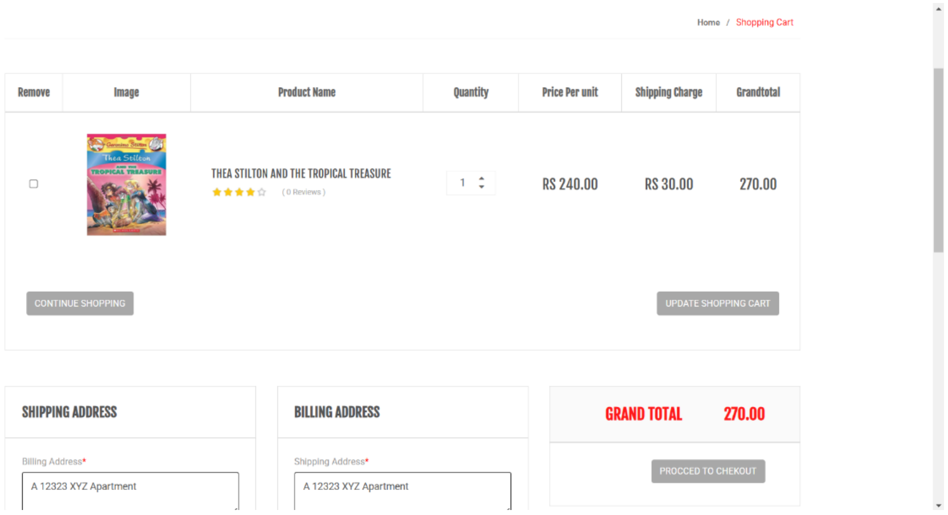
Rs. 240 ~~Rs.0~~

QTY : 1

7. Development of the Application

Date: 28/11/2024

Add to Cart



Home / Shopping Cart

Remove	Image	Product Name	Quantity	Price Per unit	Shipping Charge	Grandtotal
☐		THEA STILTON AND THE TROPICAL TREASURE (0 Reviews)	1	RS 240.00	RS 30.00	270.00

SHIPPING ADDRESS

Billing Address*

A 12323 XYZ Apartment

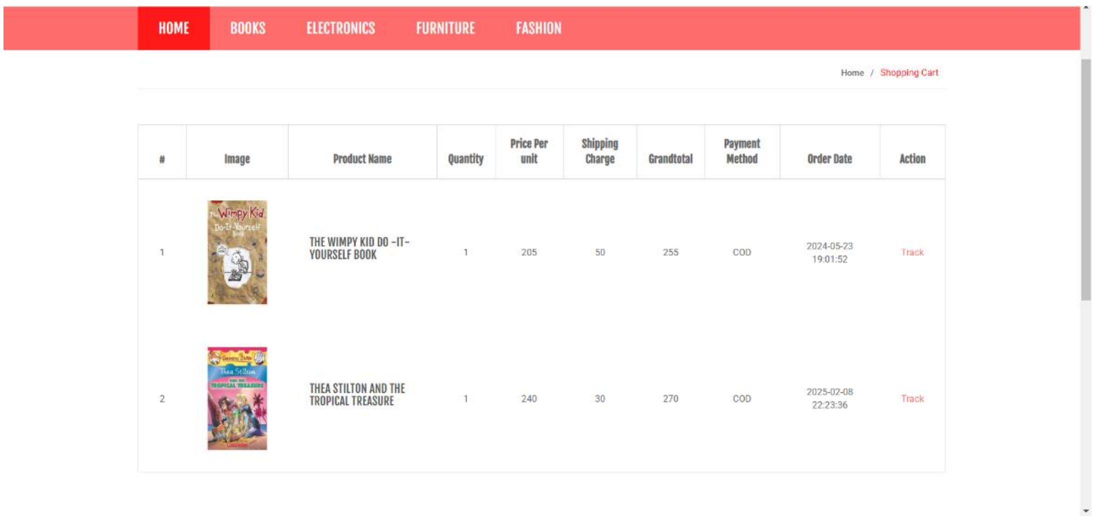
BILLING ADDRESS

Shipping Address*

A 12323 XYZ Apartment

GRAND TOTAL 270.00

Order



Home / Shopping Cart

#	Image	Product Name	Quantity	Price Per unit	Shipping Charge	Grandtotal	Payment Method	Order Date	Action
1		THE WIMPY KID DO -IT-YOURSELF BOOK	1	205	50	255	COD	2024-05-23 19:01:52	Track
2		THEA STILTON AND THE TROPICAL TREASURE	1	240	30	270	COD	2025-02-08 22:23:36	Track

8. Problem description and requirements for the given webpage Date: 04/12/2024

Problem Description:

Active individuals often struggle to find high-quality, stylish, and affordable gym apparel in one convenient location. Existing online retailers may have limited selections, confusing navigation, or lack detailed product information. This project aims to develop an e-commerce platform specializing in gym clothes, offering a wide variety of brands, styles, and sizes, along with a user-friendly shopping experience.

Requirements:

1. **Hardware Requirements**
   - **Server Requirements (for hosting the website):**
     - **Processor**: Multi-core processor (e.g., Intel Xeon, AMD Ryzen).
     - **RAM**: Minimum 8 GB (16 GB or more for better performance).
     - **Storage**: SSD with at least 100 GB of free space (expandable for large-scale applications).
     - **Network**: High-speed internet with low latency.
   - **Client-Side (for testing): o Processor: Dual-core or higher.**
     - **RAM**: 4 GB or higher.
     - **Storage**: 20 GB free space.
     - **Screen Resolution**: Minimum 1366x768.
2. **Software Requirements**
   - **For Hosting:**
     - **Web Server**: Apache, Nginx, or IIS.
     - **Database**: MySQL, PostgreSQL, or MongoDB.
     - **Backend Language Support**: Python.
     - **Operating System**: Windows Server.
   - **For Testing:**
     - **Browser Support**: Latest versions of Chrome, Firefox, Safari, and Edge.
     - **OS Compatibility**: Windows, macOS, and Linux.
     - **Device Testing**: Android and iOS devices for mobile responsiveness.
3. **Tools for Automation Testing**
   - **Web Testing Tool:**
     - **Selenium WebDriver** for browser automation.

4. **Manual Testing Requirements**
   - **Test Plan Documentation:** A document outlining test cases, scenarios, and expected outcomes.
5. **Dependencies**
   - **Frameworks and Libraries:**
     - **React.js, Angular, or Vue.js for frontend testing.**
     - **Bootstrap or Material UI for UI testing.**
   - **Plugins:**
     - **Extensions for browser automation, such as Chrome Developer Tools.**
6. **Network Requirements**
   - **Bandwidth:** Minimum 10 Mbps for smooth access and testing.
   - **Connectivity:** Stable connection to avoid disruptions during testing.

9. Manual Testing - Preparation of Test Cases and Manual Testing for the Application Date: 05/12/2024

A. User Authentication (Registration & Login)

Test Case ID	Test Steps	Test Data	Expected Results	Actual Results	Status
TC1	Enter valid username, email, and password	Username: John, Email: johndoe@gmail.com, Password: Test@123	Redirect to login page	Redirected successfully	Pass
TC2	Enter invalid email format	Username: John, Email: john#gmail.com, Password: John@123	Error message: "Invalid email format"	Displayed correctly	Pass
TC3	Leave password field empty	Username: John, Email: john@gmail.com, Password:	Alert message: "Please fill out this field"	Displayed correctly	Pass

B. Product Management

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC05	Add a new product	1. Login as Admin 2. Navigate to "Add Product" 3. Enter product details and save	Name: Laptop, Price: ₹50,000	Product added successfully	Product added successfully	Pass
TC06	Edit product details	1. Login as Admin 2. Edit a product 3. Save changes	Name: Gaming Laptop, Price: ₹55,000	Product updated successfully	Product updated successfully	Pass
TC07	Delete a product	1. Login as Admin 2. Delete a product	Product ID: 123	Product removed successfully	Product removed successfully	Pass

C. Shopping Cart & Checkout

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC08	Add product to cart	1. Open product page 2. Click "Add to Cart"	Product: Laptop	Product added successfully	Product added successfully	Pass
TC09	Remove product from cart	1. Open cart 2. Click "Remove"	Product: Laptop	Product removed successfully	Product removed successfully	Pass
TC10	Increase quantity of product in cart	1. Change quantity of an item in the cart	Quantity: 2	Quantity updated successfully	Quantity updated successfully	Pass

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC011	Checkout with valid details	1. Click "Checkout" 2. Enter address & payment info 3. Confirm order	Address: XYZ, Payment: Card	Order placed successfully	Order placed successfully	Pass
TC012	Checkout without payment details	1. Click "Checkout" without entering payment details	-	Error message "Payment details required"	-	Fail

D. Order Management

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC013	View order history	1. Login 2. Navigate to "My Orders"	-	Display user's past orders	Display user's past orders	Pass
TC014	Cancel an order	1. Open order history 2. Click "Cancel Order"	Order ID: 456	Order cancelled successfully	-	Fail

10. Manual Testing - Preparation of Test Cases and Manual Testing for the Web Page Date:11/12/2024

A. User Authentication (Registration & Login)

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC001	Register with valid details	1. Open registration page 2. Enter	Name: Alex, Email: alex@gmail.com, Password: Fit@123	Account created successfully	Account created successfully	Pass

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
		valid details 3. Click "Register"				
TC002	Register with an existing email	1. Enter an existing email 2. Click "Register"	Email: alex@gmail.com	Error message "Email already exists"	Error message "Email already exists"	Pass
TC003	Login with correct credentials	1. Enter valid email and password 2. Click "Login"	Email: alex@gmail.com, Password: Fit@123	Redirect to user dashboard	Redirect to user dashboard	Pass
TC004	Login with incorrect credentials	1. Enter correct email and wrong password 2. Click "Login"	Email: alex@gmail.com, Password: WrongPass	Error message "Invalid credentials"	Error message "Invalid credentials"	Pass

B. Product Management

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC005	View product details	1. Click on a product	Product: Gym T-shirt	Display product details	Display product details	Pass
TC006	Search for a product	1. Use search bar to find "leggings"	Search: "leggings"	Display matching results	Display matching results	Pass
TC007	Filter products by category	1. Select "T-shirts" from filters	Category: "T-shirts"	Display filtered results	Display filtered results	Pass

C. Shopping Cart & Checkout

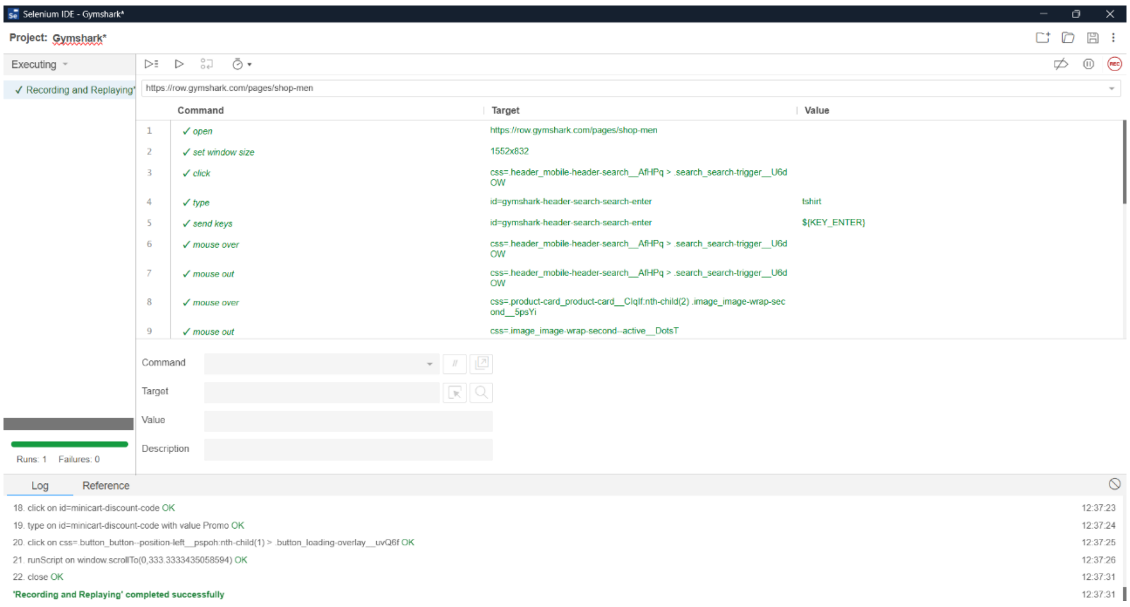
Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC008	Add product to cart	1. Open product page 2. Click "Add to Cart"	Product: Gym Shorts	Product added successfully	Product added successfully	Pass
TC009	Remove product from cart	1. Open cart 2. Click "Remove"	Product: Gym Shorts	Product removed successfully	Product removed successfully	Pass
TC010	Update product quantity	1. Change quantity of an item in the cart	Quantity: 2	Quantity updated successfully	Quantity updated successfully	Pass
TC011	Checkout with valid details	1. Click "Checkout" 2. Enter address & payment info 3. Confirm order	Address: XYZ, Payment: Card	Order placed successfully	Order placed successfully	Pass
TC012	Checkout without payment details	1. Click "Checkout" without entering payment details	-	Error message "Payment details required"	Error message "Payment details required"	Pass

D. Payment Processing

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC013	Successful payment	1. Select payment method 2. Enter valid card details 3. Confirm payment	Card: Valid	Payment successful, order confirmed	Payment successful, order confirmed	Pass

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC014	Payment failure	1. Enter incorrect card details 2. Try processing payment	Card: Invalid	Payment failure message displayed	Payment failure message displayed	Pass

11. Selenium IDE: Recording and Replay – For a given web page:
Date: 12/12/2024



Command	Target	Value
✓ open	https://row.gymshark.com/pages/shop-men	
✓ set window size	1552x832	
✓ click	css=.header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW	
✓ type	id=gymshark-header-search-search-enter	tshirt
✓ send keys	id=gymshark-header-search-search-enter	${KEY_ENTER}
✓ mouse over	css=.header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW	
✓ mouse out	css=.header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW	
✓ mouse over	css=.product-card_product-card__CIqIf:nth-child(2) .image_image-wrap-second__5psYi	
✓ mouse out	css=.image_image-wrap-second--active__DotsT	

```
18. click on id=minicart-discount-code OK                                                          12:37:23
19. type on id=minicart-discount-code with value Promo OK                                          12:37:24
20. click on css=.button_button--position-left__pspoh:nth-child(1) > .button_loading-overlay__uvQ6f OK   12:37:25
21. runScript on window.scrollTo(0,333.3333435058594) OK                                           12:37:26
22. close OK                                                                                       12:37:31
'Recording and Replaying' completed successfully                                                   12:37:31
```

Command	Target	Value
open	https://row.gymshark.com/pages/shop-men	
set window size	1552x832	
click	css=.header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW	
type	id=gymshark-header-search-search-enter	
send keys	id=gymshark-header-search-search-enter	${KEY_ENTER}
mouse over	css=.header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW	
mouse out	css=.header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW	
mouse over	css=.product-card_product-card__CIqIf:nth-child(2) .image_image-wrap-second__5psYi	

mouse out	css=.image_image-wrap-second--active__DotsT	
mouse over	css=.product-card_product-card__CIqIf:nth-child(1) .image_image-wrap-second__5psYi	
click	css=.image_image-wrap-second--active__DotsT	
mouse over	css=.image-gallery_gallery--item__OW3UF:nth-child(2) > img	
click	css=.add-to-cart_section__GygMh:nth-child(5) .size_size__saJiQ:nth-child(5)	
mouse over	css=div > .product-card_product-card__CIqIf:nth-child(1) .product-card_product-title-link__jDI6f	
mouse out	css=div > .product-card_product-card__CIqIf:nth-child(1) .product-card_product-title-link__jDI6f	
click	id=minicart-discount-code	
type	id=minicart-discount-code	Promo
click	css=.button_button--position-left__pspoh:nth-child(1) > .button_loading-overlay__uvQ6f	
run script	window.scrollTo(0,333.3333435058594)	
close		

Python Pytest:

```
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities
class TestRecordingandReplaying():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}
  def teardown_method(self, method):
    self.driver.quit()
  def test_recordingandReplaying(self):
    self.driver.get("https://row.gymshark.com/pages/shop-men")
    self.driver.set_window_size(1552, 832)
    self.driver.find_element(By.CSS_SELECTOR, ".header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW").click()
    self.driver.find_element(By.ID, "gymshark-header-search-search-enter").send_keys("tshirt")
    self.driver.find_element(By.ID, "gymshark-header-search-search-enter").send_keys(Keys.ENTER)
    element = self.driver.find_element(By.CSS_SELECTOR, ".header_mobile-header-search__AfHPq > .search_search-trigger__U6dOW")
    actions = ActionChains(self.driver)
    actions.move_to_element(element).perform()
    element = self.driver.find_element(By.CSS_SELECTOR, "body")
    actions = ActionChains(self.driver)
    actions.move_to_element(element, 0, 0).perform()
    element = self.driver.find_element(By.CSS_SELECTOR, ".product-card_product-card__CIqIf:nth-child(2) .image_image-wrap-second__5psYi")
    actions = ActionChains(self.driver)
    actions.move_to_element(element).perform()
    element = self.driver.find_element(By.CSS_SELECTOR, "body")
    actions = ActionChains(self.driver)
    actions.move_to_element(element, 0, 0).perform()
    element = self.driver.find_element(By.CSS_SELECTOR, ".product-card_product-card__CIqIf:nth-child(1) .image_image-wrap-second__5psYi")
    actions = ActionChains(self.driver)
    actions.move_to_element(element).perform()
    self.driver.find_element(By.CSS_SELECTOR, ".image_image-wrap-second--active__DotsT").click()
    element = self.driver.find_element(By.CSS_SELECTOR, ".image-gallery_gallery--item__OW3UF:nth-child(2) > img")
    actions = ActionChains(self.driver)
    actions.move_to_element(element).perform()
    element = self.driver.find_element(By.CSS_SELECTOR, "body")
    actions = ActionChains(self.driver)
    actions.move_to_element(element, 0, 0).perform()
    self.driver.find_element(By.CSS_SELECTOR, ".add-to-cart_section__GygMh:nth-child(5) .size_size__saJiQ:nth-child(5)").click()
    self.driver.find_element(By.CSS_SELECTOR, ".add-to-cart_button-container__aKAUo:nth-child(6) .button_success-overlay__aqLb7").click()
    element = self.driver.find_element(By.CSS_SELECTOR, "div > .product-card_product-card__CIqIf:nth-child(1) .product-card_product-title-link__jDI6f")
    actions = ActionChains(self.driver)
    actions.move_to_element(element).perform()
    element = self.driver.find_element(By.CSS_SELECTOR, "body")
    actions = ActionChains(self.driver)
    actions.move_to_element(element, 0, 0).perform()
    self.driver.find_element(By.ID, "minicart-discount-code").click()
    self.driver.find_element(By.ID, "minicart-discount-code").send_keys("Promo")
    self.driver.find_element(By.CSS_SELECTOR, ".button_button--position-left__pspoh:nth-
child(1) > .button_loading-overlay__uvQ6f").click()
    self.driver.execute_script("window.scrollTo(0,333.3333435058594)")
    self.driver.close()
```

12. Selenium IDE: Commands: Date: 18/12/2024

1. Navigation Commands

Command	**Description**
open	Opens a specified URL.
goBack	Navigates to the previous page in browser history.
goForward	Moves forward in browser history.
refresh	Reloads the current page.

2. Interaction Commands

Command	**Description**
Click	Clicks on an element.
clickAt	Clicks on an element at a specific coordinate.
doubleClick	Performs a double-click on an element.
Type	Enters text into an input field.
sendKeys	Simulates keyboard input.
Check	Selects a checkbox.
Uncheck	Deselects a checkbox.
Select	Selects an option from a dropdown.
chooseOkOnNextConfirmation	Confirms an alert pop-up.
chooseCancelOnNextConfirmation	Cancels an alert pop-up.

3. Assertion & Verification Commands

Command	**Description**
assertTitle	Verifies the page title (fails test if not matched).

Command	Description
assertText	Verifies if an element contains the expected text.
assertElementPresent	Ensures an element exists.
assertElementNotPresent	Ensures an element does not exist.
verifyText	Similar to assertText but does not stop execution on failure.
verifyElementPresent	Similar to assertElementPresent but does not stop execution on failure.
verifyChecked	Checks if a checkbox is selected.

4. Store Commands (Variable Handling)

Command	Description
store	Stores a value in a variable.
storeTitle	Stores the page title in a variable.
storeText	Stores text from an element.
storeValue	Stores the value of an input field.
storeAttribute	Stores an attribute of an element.

5. Wait Commands

Command	Description
waitForElementPresent	Waits for an element to be available on the page.
waitForElementNotPresent	Waits until an element is removed.
waitForText	Waits until an element contains specific text.
waitForValue	Waits until an input field has a specific value.

6. Control Flow Commands

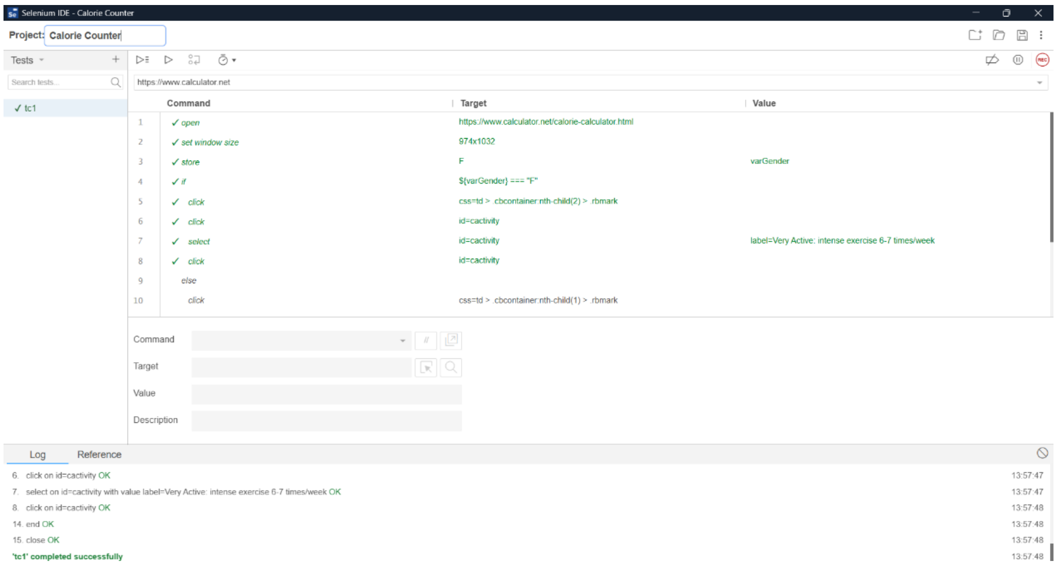
Command	Description
if...else	Executes a block of commands conditionally.
while	Repeats a block of commands while a condition is true.

Command	Description
do...repeatIf	Repeats a set of commands until a condition is met.
forEach	Iterates over an array of values.
break	Exits from a loop.
end	Ends an if, while, or forEach block.

7. Debugging & Execution Control Commands

Command	Description
pause	Stops execution for a specified time (in milliseconds).
echo	Logs a message in the execution log.
setSpeed	Sets the execution speed of Selenium IDE.
runScript	Executes JavaScript code on the page.

13. Selenium IDE: Conditional Statements: Date: 26/12/2024


Selenium IDE - Calorie Counter

Project: Calorie Counter

https://www.calculator.net

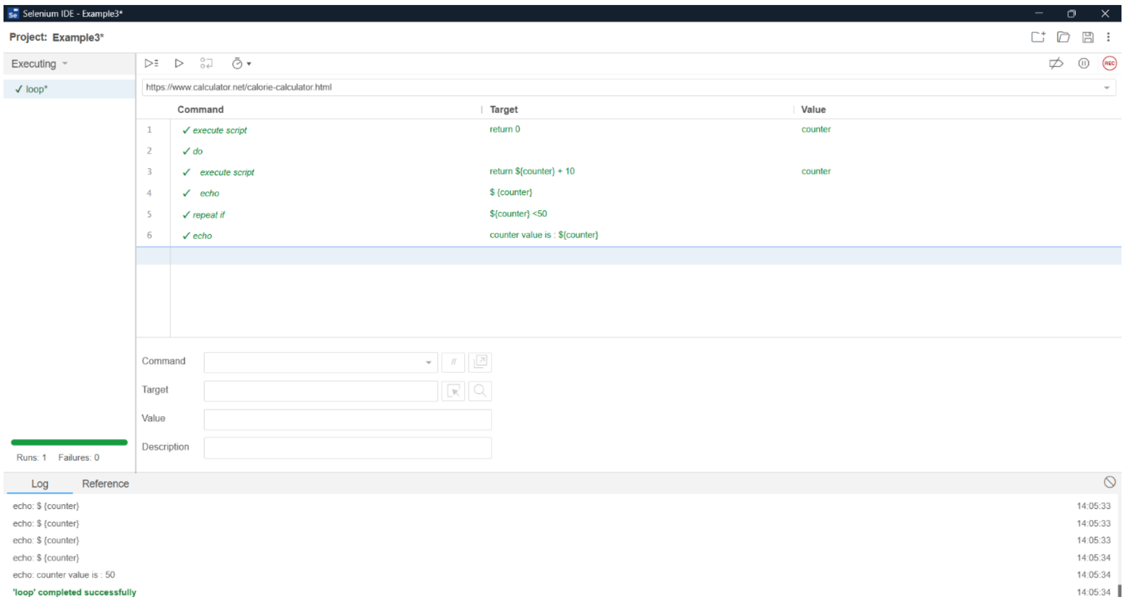
	Command	Target	Value
1	✓ open	https://www.calculator.net/calorie-calculator.html	
2	✓ set window size	974x1032	
3	✓ store	F	varGender
4	✓ if	${varGender} === "F"	
5	✓ click	css=td > .cbcontainer:nth-child(2) > .rbmark	
6	✓ click	id=cactivity	
7	✓ select	id=cactivity	label=Very Active: intense exercise 6-7 times/week
8	✓ click	id=cactivity	
9	else		
10	click	css=td > .cbcontainer:nth-child(1) > .rbmark	

```
6.  click on id=cactivity OK                                                          13:57:47
7.  select on id=cactivity with value label=Very Active: intense exercise 6-7 times/week OK   13:57:47
8.  click on id=cactivity OK                                                          13:57:48
14. end OK                                                                            13:57:48
15. close OK                                                                          13:57:48
'tc1' completed successfully                                                          13:57:48
```

Python Pytest:

```
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities
class TestNew1():
def setup_method(self, method):
self.driver = webdriver.Chrome()
self.vars = {}
def teardown_method(self, method):
self.driver.quit()
def test_new1(self):
self.driver.get("https://www.calculator.net/calorie-calculator.html")
self.vars["varGender"] = "F"
if self.driver.execute_script("return (arguments[0]===\"F\")", self.vars["varGender"]):
self.driver.find_element(By.CSS_SELECTOR, "td>.cbcontainer:nth-child(2)>.rbmark").click()
self.driver.find_element(By.ID, "cactivity").click()
dropdown = self.driver.find_element(By.ID, "cactivity")
dropdown.find_element(By.XPATH, "//option[. = 'Very Active: intense exercise 6-7 times/week']").click()
self.driver.find_element(By.ID, "cactivity").click()
else:
self.driver.find_element(By.CSS_SELECTOR, "td>.cbcntainer:nth-child(1)>.rbmark").click()
self.driver.find_element(By.ID, "cactivity").click()
dropdown = self.driver.find_element(By.ID, "cactivity")
dropdown.find_element(By.XPATH, "//option[. = '1']").click()
self.driver.find_element(By.ID, "cactivity").click()
self.driver.close()
```

14. Selenium IDE: Loops: Date: 01/01/2025



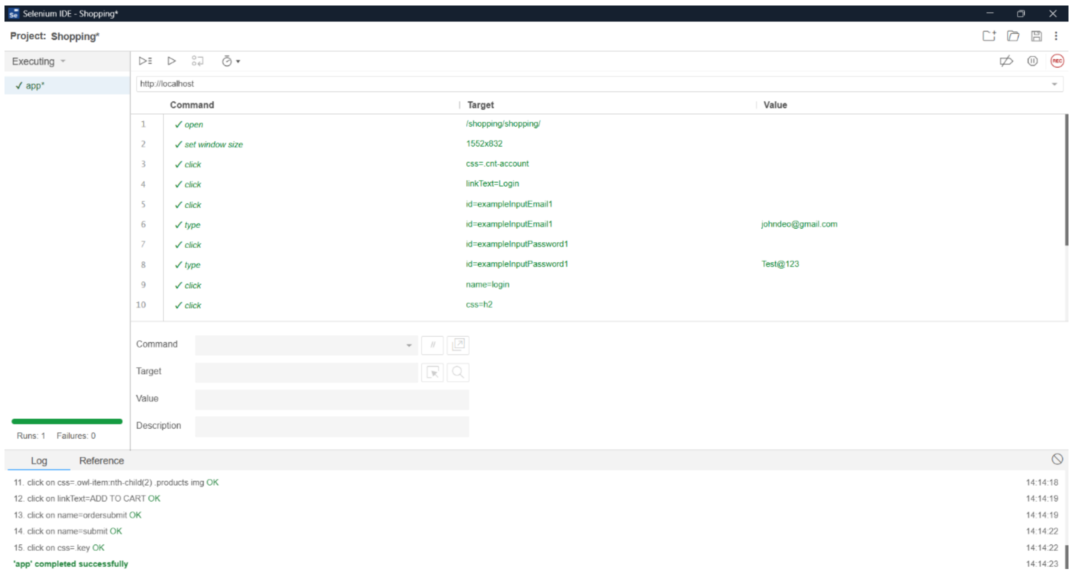
	Command	Target	Value
1	✓ execute script	return 0	counter
2	✓ do		
3	✓ execute script	return ${counter} + 10	counter
4	✓ echo	$ {counter}	
5	✓ repeat if	${counter} <50	
6	✓ echo	counter value is : ${counter}	

```
echo: $ {counter}                        14:05:33
echo: $ {counter}                        14:05:33
echo: $ {counter}                        14:05:33
echo: $ {counter}                        14:05:34
echo: counter value is : 50              14:05:34
'loop' completed successfully            14:05:34
```

Python Pytest:

```
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities
class TestLoop():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}
  def teardown_method(self, method):
    self.driver.quit()
  def test_loop(self):
    self.vars["counter"] = self.driver.execute_script("return 0")
    condition = True
    while condition:
      self.vars["counter"] = self.driver.execute_script("return arguments[0] + 10",
self.vars["counter"])
      print("$ {counter}")
      condition = self.driver.execute_script("return (arguments[0] <50)", self.vars["counter"])
    print("counter value is : {}".format(self.vars["counter"]))
```

15. Selenium IDE: For the Application: Date: 02/01/2025



	Command	Target	Value
1	✓ open	/shopping/shopping/	
2	✓ set window size	1552x832	
3	✓ click	css=.cnt-account	
4	✓ click	linkText=Login	
5	✓ click	id=exampleInputEmail1	
6	✓ type	id=exampleInputEmail1	johndeo@gmail.com
7	✓ click	id=exampleInputPassword1	
8	✓ type	id=exampleInputPassword1	Test@123
9	✓ click	name=login	
10	✓ click	css=h2	

```
11. click on css=.owl-item:nth-child(2) .products img OK        14:14:18
12. click on linkText=ADD TO CART OK                            14:14:19
13. click on name=ordersubmit OK                                14:14:19
14. click on name=submit OK                                     14:14:22
15. click on css=.key OK                                        14:14:22
'app' completed successfully                                    14:14:23
```

Python Pytest:

```
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities
class TestApp():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}
  def teardown_method(self, method):
    self.driver.quit()
  def test_app(self):
    self.driver.get("http://localhost/shopping/shopping/")
    self.driver.set_window_size(1552, 832)
    self.driver.find_element(By.CSS_SELECTOR, ".cnt-account").click()
    self.driver.find_element(By.LINK_TEXT, "Login").click()
    self.driver.find_element(By.ID, "exampleInputEmail1").click()
    self.driver.find_element(By.ID, "exampleInputEmail1").send_keys("johndeo@gmail.com")
    self.driver.find_element(By.ID, "exampleInputPassword1").click()
    self.driver.find_element(By.ID, "exampleInputPassword1").send_keys("Test@123")
    self.driver.find_element(By.NAME, "login").click()
    self.driver.find_element(By.CSS_SELECTOR, "h2").click()
    self.driver.find_element(By.CSS_SELECTOR, ".owl-item:nth-child(2) .products img").click()
    self.driver.find_element(By.LINK_TEXT, "ADD TO CART").click()
    self.driver.find_element(By.NAME, "ordersubmit").click()
    self.driver.find_element(By.NAME, "submit").click()
    self.driver.find_element(By.CSS_SELECTOR, ".key").click()
```

16. Selenium Web driver – Locating web element – id, name, link_text, partial_ link_text – Application: Date: 08/01/2025

```
from selenium import webdriver
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
driver.maximize_window()
element_by_id = driver.find_element(By.ID, "lightbox")
```

```
print("Element found by ID")
element_by_name = driver.find_element(By.NAME, "product")
print("Element found by Name")
element_by_link_text = driver.find_element(By.LINK_TEXT, "My Account")
print("Element found by Link Text")
element_by_partial_link_text = driver.find_element(By.PARTIAL_LINK_TEXT, "Account")
print("Element found by Partial Link Text")
driver.quit()
```

17. Selenium Web driver – Locating web element – id, name, link_text, partial_ link_text – Webpage: Date: 09/12/2025

```
from selenium import webdriver
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("https://row.gymshark.com/pages/shop-men")
driver.maximize_window()
element_by_id = driver.find_element(By.ID, "carousel-heading")
print("Element found by ID")
element_by_name = driver.find_element(By.NAME, "description")
print("Element found by Name")
element_by_link_text = driver.find_element(By.LINK_TEXT, "Accessibility Statement")
print("Element found by Link Text")
element_by_partial_link_text = driver.find_element(By.PARTIAL_LINK_TEXT, "Statement")
print("Element found by Partial Link Text")
driver.quit()
```

18. Selenium Web driver – Locating web element – class name and tag name – Application: Date: 15/01/2025

```
from selenium import webdriver
from time import sleep
import time
from selenium.webdriver import Keys
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
driver.maximize_window()
sleep(1)
driver.find_element(By.CLASS_NAME, "search-field").click()
driver.find_element(By.CLASS_NAME, "search-field").send_keys("Shoes")
driver.find_element(By.CLASS_NAME, "search-field").send_keys(Keys.ENTER)
element_by_tag_name = driver.find_element(By.TAG_NAME, "p")
print("\nElement found by Tag Name\n")
driver.quit()
```

19. Selenium Web driver – Locating web element – class name and tag name – Webpage Date: 16/01/2025

```
from selenium import webdriver
from time import sleep
import time
from selenium.webdriver import Keys
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("https://row.gymshark.com/pages/shop-men")
driver.maximize_window()
time.sleep(1)
driver.find_element(By.CLASS_NAME, "header_mobile-header-search__AfHPq").click()
time.sleep(1)
driver.find_element(By.ID, "gymshark-header-search-search-enter").send_keys("running")
driver.find_element(By.ID, "gymshark-header-search-search-enter").send_keys(Keys.ENTER)
element_by_tag_name = driver.find_element(By.TAG_NAME, "p")
print("\nElement found by Tag Name\n\n")
driver.quit()
```

20. Selenium Web driver – Locating web element – CSSSelector – Application: Date: 22/01/2025

```
from selenium import webdriver
import time
from selenium.webdriver import Keys
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
driver.maximize_window()
time.sleep(1) #Class Selector
driver.find_element(By.CSS_SELECTOR, "a .fa-sign-in").click()
time.sleep(3) #ID Selector
driver.find_element(By.CSS_SELECTOR,
'#exampleInputEmail1').send_keys("johndeo@gmail.com")
time.sleep(2) # Attribute Selector
driver.find_element(By.CSS_SELECTOR, "input[name='password']").send_keys("Test@123")
time.sleep(2) # Combinator Selector
driver.find_element(By.CSS_SELECTOR, 'form button[name="login"]').click()
driver.find_element(By.LINK_TEXT, "Shopping Portal").click()
time.sleep(2)print("Login Successful\n")
driver.quit()
```

21.Selenium Web driver – Locating web element – CSSSelector – Webpage: Date: 23/02/2025

```
from selenium import webdriver
import time
from selenium.webdriver import Keys
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("https://row.gymshark.com/pages/shop-men")
driver.maximize_window()
time.sleep(1)
driver.find_element(By.CSS_SELECTOR, ".header_action-bar-item-wrap__m_Rhz .icon-
user").click()
time.sleep(2) #Class Selector
driver.find_element(By.CSS_SELECTOR, "input.input.cec61d1fe.c54babdaf").click()
time.sleep(2) #ID Selector
driver.find_element(By.CSS_SELECTOR, "#username").send_keys("john@gmail.com")
time.sleep(2) #Attribute Selector
driver.find_element(By.CSS_SELECTOR, "input[id='password']").click()
time.sleep(2) #Combinators
driver.find_element(By.CSS_SELECTOR,"input[type='password'][autocomplete='current-
password']").send_keys("Test!@3")
time.sleep(2)
driver.find_element(By.CSS_SELECTOR, ".ulp-button-icon").click()
time.sleep(2)
driver.quit()
```

22. Selenium Web driver – Locating web element – xpath – Application: Date: 24/01/2025

```
from selenium import webdriver
import time
from selenium.webdriver import Keys
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
driver.maximize_window()
time.sleep(3)
driver.find_element(By.XPATH,
"/html/body/header/div[2]/div/div/div[2]/div/form/div/input").send_keys("Phone")
time.sleep(3)
driver.find_element(By.XPATH,
"/html/body/header/div[2]/div/div/div[2]/div/form/div/button").click()
time.sleep(3)
```

```
# product.send_key(Keys.ENTER)
driver.find_element(By.XPATH,
"/html/body/div[1]/div/div[1]/div[2]/div[2]/div/div/div/div/div/div/div/div[1]/div/a/img").click()
time.sleep(3)
driver.find_element(By.XPATH,"/html/body/div[2]/div/div[1]/div[2]/div[1]/div[2]/div/div[6]/div/di
v[3]/a").click()
time.sleep(3)
driver.find_element(By.XPATH,
"/html/body/div[2]/div/div[1]/div/div[4]/table/tbody/tr/td/div/button").click()
time.sleep(3)
driver.quit()
```

23. Selenium Web driver – Locating web element – xpath – Webpage:
Date: 29/01/2025

```
from selenium import webdriver
import time
from selenium.webdriver import Keys
from selenium.webdriver.common.by import By
driver = webdriver.Chrome()
driver.get("https://row.gymshark.com/pages/shop-men")
driver.maximize_window()
time.sleep(1)
driver.find_element(By.CSS_SELECTOR, ".header_mobile-header-search__AfHPq >
.search_search-trigger__U6dOW").click()
time.sleep(1)
driver.find_element(By.CSS_SELECTOR, "body")
time.sleep(1)
driver.find_element(By.ID, "gymshark-header-search-search-enter").send_keys("running")
time.sleep(2)
driver.find_element(By.ID, "gymshark-header-search-search-enter").send_keys(Keys.ENTER)
time.sleep(2)
driver.find_element(By.CLASS_NAME, "product-card_product-title-link__jDI6f").click()
time.sleep(3)
driver.find_element(By.CSS_SELECTOR, ".add-to-cart_section__GygMh:nth-child(5)
.size_size__saJiQ:nth-child(5)").click()
time.sleep(3)
driver.find_element(By.CSS_SELECTOR, ".add-to-cart_button-container__aKAUo:nth-child(6)
.button_success-overlay__aqLb7").click()
time.sleep(4)
driver.find_element(By.XPATH,
"/html/body/div[6]/div/div/div[2]/div[3]/div/div[4]/div/div/fieldset/input").click()
time.sleep(2)
driver.find_element(By.CLASS_NAME, "input_input__iz_H4").send_keys("PROMO")
```

```
time.sleep(2)
driver.find_element(By.XPATH,
"/html/body/div[6]/div/div/div[2]/div[3]/div/div[4]/div/div/button/span[2]").click()
time.sleep(2)
print("\nOrder Placed Successfully")
driver.quit()
```

24. Application commands and Browser commands: Date: 30/01/2025

```
from selenium import webdriver
driver = webdriver.Chrome()
import time
#application command
driver.get("http://localhost/shopping/shopping/")
time.sleep(3)
print("Title:", driver.title)
time.sleep(1)
current_url = driver.current_url
print("Current URL:", current_url)
time.sleep(1)
#browser command
driver.maximize_window()
time.sleep(3)
driver.minimize_window()
time.sleep(3)
driver.maximize_window()
time.sleep(3)
driver.quit()
```

25. Conditional Commands and Navigation Commands. Date: 31/01/2025

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
# Conditional Commands
```

```
element = driver.find_element(By.TAG_NAME, "body")
if element.is_displayed():
    print("Element is displayed")
if element.is_enabled():
    print("Element is enabled")
if element.is_selected():
    print("Element is selected")
time.sleep(3)
# Navigation Commands
driver.get("https://row.gymshark.com/pages/shop-men")
time.sleep(3)
print("Navigated to Application")
driver.back()
time.sleep(3)
print("Navigated back")
driver.forward()
time.sleep(4)
print("Navigated forward")
driver.refresh()
print("Page refreshed again")
driver.quit()
```

26. Difference between find_element and find_elements: Date: 05/02/2025

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
element = driver.find_element(By.TAG_NAME, "a")
print("Single element found by tag name:", element.text)
elements = driver.find_elements(By.TAG_NAME, "p")
print("\n\nMultiple elements found by tag name:")
for e in elements:
    print(e.text)
```

```
driver.close()
```

27. Difference between text and get_attribute: Date: 06/02/2025

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
element = driver.find_element(By.XPATH, "/html/body/header/div[2]/div/div/div[1]/div/a")
print("Element text:", element.text)
print("Element href attribute:", element.get_attribute("href"))
driver.quit()
```

28. Wait Commands Date: 07/02/2025

```
from selenium import webdriver
import time
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
driver = webdriver.Chrome()
driver.get("http://localhost/shopping/shopping/")
driver.maximize_window()
wait = WebDriverWait(driver, 10)
product_input = wait.until(EC.presence_of_element_located((By.CSS_SELECTOR,
"input[name=product]")))
product_input.send_keys("Book")
time.sleep(3)
print("Entered 'Book' in the product input field.")
search_button = wait.until(EC.element_to_be_clickable((By.CSS_SELECTOR,
"button[name=search]")))
search_button.click()
print("Clicked the search button.")
driver.quit()
```