Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Semester: (Fall, Year: 2024), B.Sc. in CSE (Day)

"ShopEase (E-commerce Website)"

Course Title: Software Testing and Quality Assurance Lab

Course Code: CSE 454

Section: 211D1

Students Details

Name	ID
JOY BAIDYA	211002142

Submission Date: 22/12/2024

Course Teacher's Name: Sudip Chandra Ghoshal

[For teacher's use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

TABLE OF CONTENTS

List of Figures iv

List of Tables 1

1 Introduction 1

1.1 Overview 1

1.2 Problem Domain 1

1.3 Motivation 2

1.4 Testing Methodologies 3

1.5 Testing Features 4

1.5.1 Manual Testing and Bug Tracking 4

1.5.2 Automated Testing 4

1.5.3 Performance Testing 4

1.5.4 API Testing 5

1.5.5 Black-Box Testing 5

1.5.6 Regression Testing 5

1.5.7 Boundary Value Analysis and Equivalence Partitioning 5

1.6 Tools and Technologies 6

2 Design/Development/Implementation of the Project 7

2.1 Introduction 7

2.2 Project Details 7

2.2.1 Home Page 7

2.2.2 Register Page 8

2.2.3 Login Page 9

2.2.4 Product Page 10
2.2.5 Details Page 11
2.2.6 About Us Page 12
2.2.7 Contact Us Page 13
2.2.8 Footer Page 14
2.3 Testing Details 15
2.3.1 Test Case (Manual Testing) 15
2.3.2 Performance Testing 19
2.3.3 Performance Metrics Observed During Load Testing 21
2.3.4 Load Testing Execution for Website 22
2.3.5 Stress Testing with JMeter 23
2.3.6 Endurance Testing with JMeter 25
2.3.7 Spike testing Testing with JMeter 26
2.3.8 API Testing 29
2.3.9 Automation Testing 31
2.3.10 Security Testing 54

3 Conclusion 55
3.1 Conclusion 55
3.2 Limitations 55
3.3 Future Scope 56

List of Figures

1.1 SCRUM Process 4
1.2 Black-Box Testing 5

2.1 Home Page 8
2.2 Register Page 9
2.3 Login Page 10
2.4 Product Page 11
2.5 Details Page 12
2.6 About Us Page 13
2.7 Contact Us Page 14
2.8 Footer Page 14
2.9 Test case (Manual Test for ShopEase website) 17
2.10 Test case (Manual Test for ShopEase website) 17
2.11 Test case (Manual Test for ShopEase website) 18
2.12 Test case (Manual Test for ShopEase website) 18
2.13 Test case (Manual Test for ShopEase website) 19
2.14 Test case (Manual Test for ShopEase website) 19
2.15 Performance Testing 20
2.16 The Load testing for web site 23
2.17 The Stress testing for web site 24
2.18 The Endurance testing for web site 25
2.19 The Spike testing for web site 27
2.20 API testing for this ShopEase 29

2.21 Verify Home Page Loads 31
2.22 Validate Title Match for Home Page 32
2.23 Verify Registration Modal Popup 33
2.24 Verify Registration Modal Popup Result 33
2.25 Verify Successful Registration Process 34
2.26 Verify Successful Registration Process Result 35
2.27 Verify Validation for Missing Information 36
2.28 Verify Validation for Missing Information Result 36
2.29 Boundary Value Analysis and Equivalence Partitioning 37
2.30 Boundary Value Analysis and Equivalence Partitioning 38
2.31 Boundary Value Analysis and Equivalence Partitioning successfull Result 39
2.32 Boundary Value Analysis and Equivalence Partitioning 40
2.33 Invalid Boundary Value Analysis and Equivalence Partitioning 40
2.34 valid SignUp.....Test (error occur: ”Infinity password takes.) 41
2.35 Valid Login Popup and Authentication 42
2.36 Password is correct but Email is In-correct 43
2.37 Password is incorrect but Email is correct. 44
2.38 Password is incorrect and Email is incorrect. 45
2.39 Password is empty and Email is empty. 46
2.40 SeeAllProduct 47
2.41 Navigate to Products 48
2.42 Verify Search Functionality 49
2.43 Product Category Filter Functionality 50
2.44 Validate Price Range Filter 51
2.45 Validate Price Range Filter Rusult 52
2.46 Validate High-to-Low And Low-to- High Sorting 53
2.47 Security Testing 54

Chapter 1

Introduction

1.1 Overview

The ShopEase e-commerce platform is designed to provide a seamless shopping experience to users. This testing phase aims to ensure the system's functionality, performance, security, and reliability while employing modern tools, languages, and methodologies. The key focus is to validate all critical modules such as user management, product catalog, cart, checkout, and payment gateways.

1.2 Problem Domain

The ShopEase e-commerce platform requires comprehensive testing to validate its critical functionalities and user-centric features.

- **User Management:** Ensuring reliable processes for user registration, authentication, and account management.
- **Product Catalog:** Verifying the accuracy of product listings, proper categorization, and efficient search capabilities.

The testing phase aims to guarantee a seamless and secure shopping experience through rigorous evaluation.

Testing Objectives

The ShopEase e-commerce platform's testing aims to ensure its quality and reliability. Key objectives include:

a. **Functionality Testing:** Validate core features like user management, product catalog, cart, checkout, and payment gateways.

b. **Performance Testing:** Ensure the platform is responsive and stable under varying loads.

c. **Security Testing:** Identify vulnerabilities to protect user data and transactions.

d. **Usability Testing:** Confirm the platform is intuitive and provides a consistent experience across devices.

e. **Integration Testing:** Verify that third-party services, like payment gateways, function correctly.

These objectives help deliver a seamless and dependable shopping experience.

1.3 Motivation

The motivation behind testing the ShopEase e-commerce platform lies in the critical role it plays in delivering a reliable and seamless shopping experience. With the growing reliance on online shopping, it is essential to ensure that the platform operates flawlessly across all key modules.

a. **User Trust:** Robust user management builds trust by ensuring secure registration, authentication, and account handling. A system that protects user data and provides an intuitive experience encourages repeat usage.

b. **Product Accessibility:** Accurate product cataloging, efficient categorization, and intuitive search functionality allow users to easily find and purchase desired items. This enhances user satisfaction and drives business growth.

c. **User Experience:** Usability refers to how easily and efficiently users can interact with a product or system. It focuses on user satisfaction, ease of learning, efficiency, error recovery, and overall experience.

By addressing these areas through meticulous testing, ShopEase aims to establish itself as a reliable e-commerce solution that prioritizes user satisfaction and operational excellence.

1.4 Testing Methodologies

Agile Methodologies

Agile methodologies are widely adopted for testing. There are various Agile frameworks, including:

- **Scrum:** Divides projects into short sprints (1-4 weeks), with roles like Scrum Master and Product Owner to ensure focus and track progress.
- **Kanban:** Focuses on visualizing workflows and continuous delivery, without fixed-length sprints.
- **Extreme Programming (XP):** Emphasizes engineering practices like continuous integration and pair programming for better quality.
- **Lean:** Focuses on reducing waste and maximizing value, optimizing processes to focus on what matters most to the customer.

Scrum Overview

1.1 figure illustrates the SCRUM Process, an Agile framework for iterative development. It starts with the Product Backlog, followed by Sprint Planning, and a Sprint (1-4 weeks) where the team works on selected tasks. The process includes Daily Standups, a Sprint Review Retrospective, and ends with Finished Work.

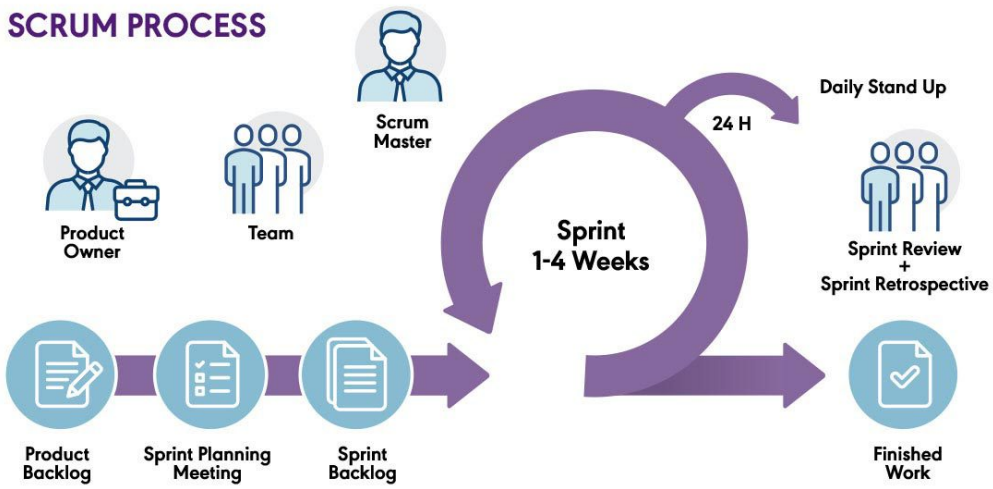


Figure 1.1: SCRUM Process

1.5 Testing Features

1.5.1 Manual Testing and Bug Tracking

I was involved in executing test cases manually, identifying bugs, and reporting them using tools like Jira. This task sharpened my attention to detail and my ability to track and document defects accurately.

1.5.2 Automated Testing

I applied my understanding of test automation by writing and executing automated test scripts using Selenium. This involved developing scripts to automate repetitive testing tasks and integrating these tests into a Jenkins CI/CD pipeline to ensure code quality across the team's workflow.

1.5.3 Performance Testing

Performance Testing evaluates how a system behaves under various workloads, ensuring speed, scalability, and stability. It helps identify bottlenecks, set benchmarks, and

optimize performance for both normal and extreme conditions.

1.5.4 API Testing

Using Postman, I tested the APIs of various software applications, ensuring that data exchanges between systems functioned as expected. I also wrote automation scripts for API tests, which further strengthened my understanding of end-to-end software testing.

1.5.5 Black-Box Testing

1.2 figure explains Black-Box Testing, a software testing technique that evaluates system functionality without knowledge of internal code structure. It highlights five key techniques: Equivalent Partitioning, Boundary Value Analysis, Decision Table Testing, State Transition Testing, and Use Case Testing, which help ensure software reliability by testing different input scenarios.

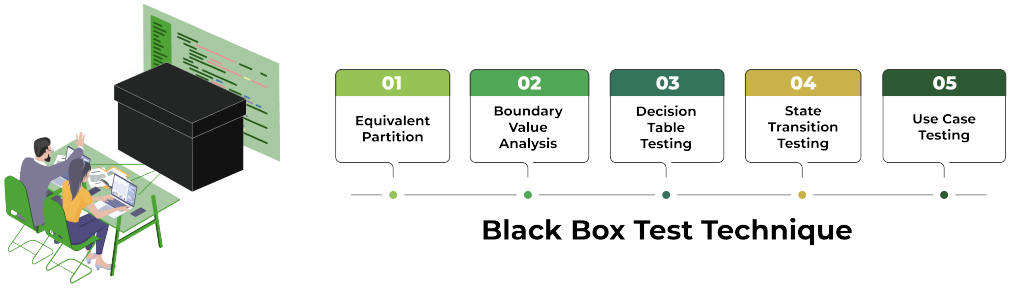


Figure 1.2: Black-Box Testing

1.5.6 Regression Testing

Ensuring that recent changes do not disrupt existing functionality.

1.5.7 Boundary Value Analysis and Equivalence Partitioning

Boundary Value Analysis (BVA) focuses on testing the boundaries of input ranges. Errors often occur at the edges rather than in the middle of input ranges.

1.6 Tools and Technologies

Testing Tools:

- Personal Computer
- Web Browser
- Testing tool kit - Selenium
- Netbeans
- Selenium (Automation Testing)
- JMeter (Performance Testing)
- Postman (API Testing)
- Test Management Tools: Jira for issue tracking and test case management

Languages Used:

- Java for Selenium-based testing

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

We have developed a modern e-commerce website that is fully responsive and user-friendly, utilizing technologies such as HTML, CSS, Tailwind CSS, React.js, JavaScript, Node.js, Express.js, Firebase, and MongoDB. This project offers a range of features, including dynamic search functionality, category-based filtering, price range sorting (from high to low and low to high), and a customizable dark mode for enhanced usability. Designed with a focus on performance and accessibility, the website ensures a seamless shopping experience across all devices, making it both visually appealing and functional.

2.2 Project Details

2.2.1 Home Page

2.1 figure explains The home page of the e-commerce website "ShopEase" features a clean and fully responsive design. It includes a navigation bar at the top for easy access to different sections, followed by an interactive slider showcasing key highlights

or promotions and some option for filtering products.

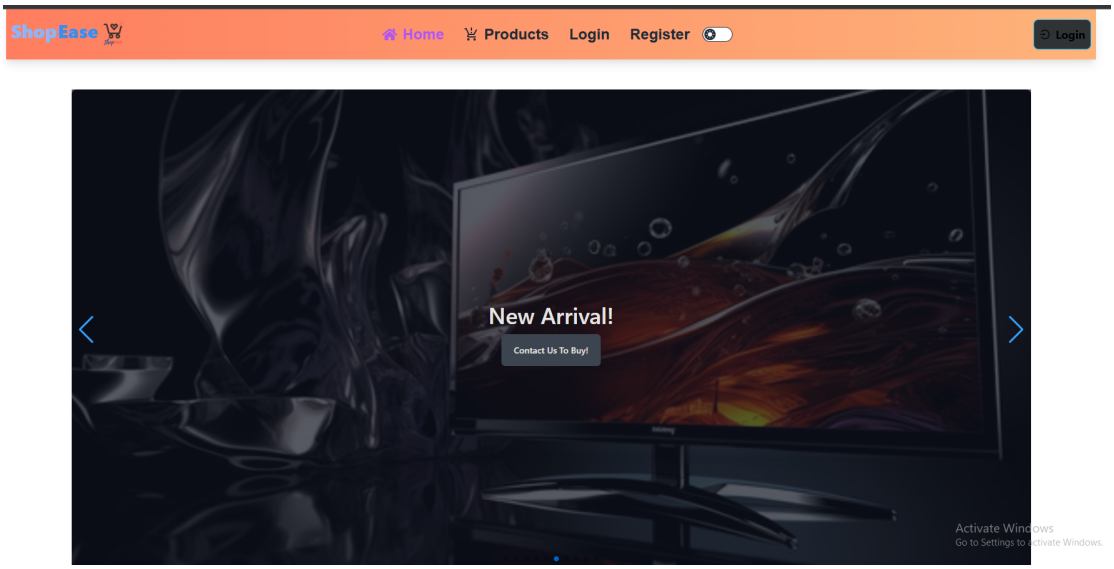


Figure 2.1: Home Page

Below the slider, a selection of product items is displayed to give users a preview of the offerings. The page is completed with a footer at the bottom, providing additional links and essential information. The layout ensures a seamless user experience across all devices.

2.2.2 Register Page

2.2 figure explains The register page provides a simple and user-friendly interface for new users to create an account. It includes fields for essential details such as name, email, password, and any additional required information. The layout is fully responsive, ensuring accessibility on all devices. Validation messages guide users to enter correct information, and options like Google authentication may also be included for convenience. The page prioritizes security and ease of use to streamline the registration process.

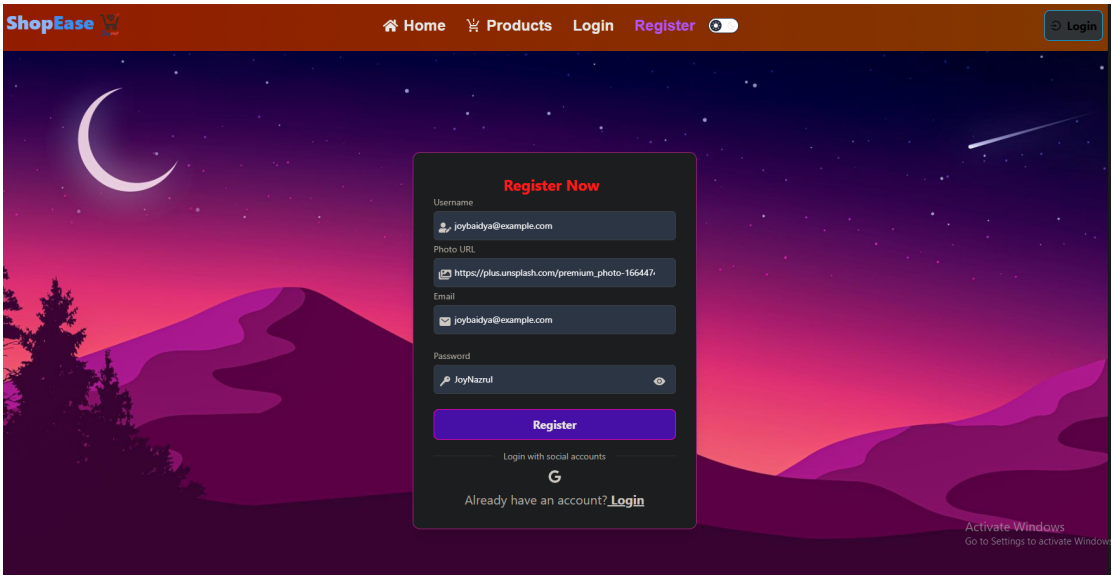


Figure 2.2: Register Page

2.2.3 Login Page

2.3 figure explains The login page is designed with simplicity and responsiveness in mind, ensuring a smooth user experience across all devices. It features a clean layout with fields for email and password input, along with options for Google authentication or other login methods. Error messages are displayed for invalid inputs to guide users effectively. The page ensures secure authentication while maintaining a user-friendly interface.

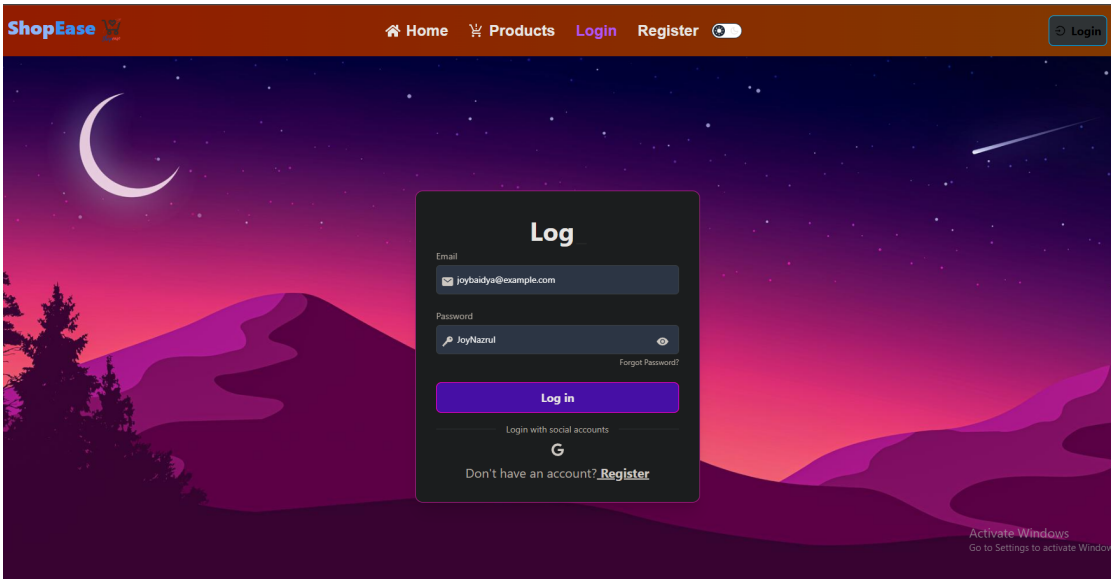


Figure 2.3: Login Page

2.2.4 Product Page

2.4 figure explains The product page displays all products in a grid layout, with each product presented in a card format. Each card includes key information such as the product name, image, price, and category. The responsive design ensures the grid adapts seamlessly to different screen sizes, providing a consistent and user-friendly browsing experience. Users can easily explore the products through intuitive filtering, sorting, and search options, making it simple to find items of interest.

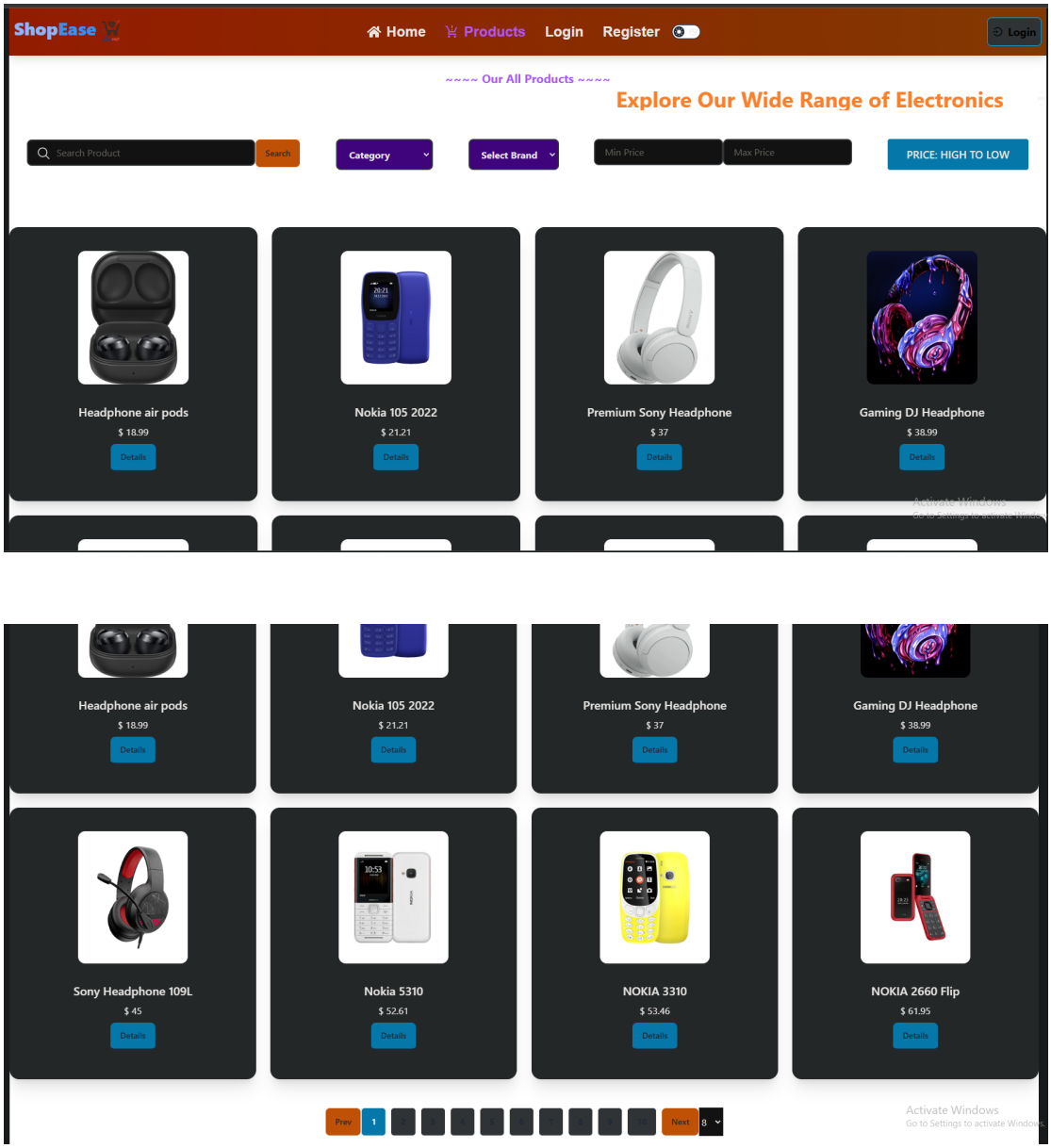


Figure 2.4: Product Page

2.2.5 Details Page

2.5 figure explains The details page provides comprehensive information about a selected product in a clean and responsive design. It includes a larger product image, detailed description, price, category, and additional features if applicable. Designed for a seamless user experience, the layout ensures that all information is easily accessible and visually appealing across all devices. From this page user can back to home page by

clicking on the button “Back to home”. The details page offers comprehensive product information with a clean, responsive design, including a large image, description, price, category, and features, along with a ”Back to home” button for easy navigation.

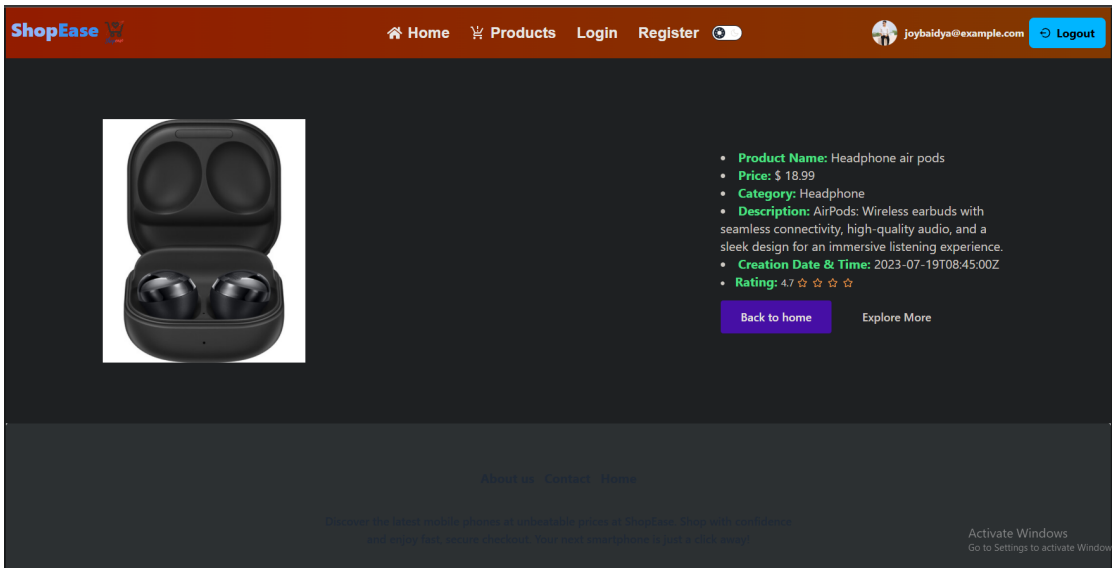


Figure 2.5: Details Page

2.2.6 About Us Page

2.6 figure explains The ”About Us” page offers an insightful overview of the website, its mission, and the team behind it. It highlights the core values, vision, and purpose of the e-commerce platform, focusing on how it aims to provide customers with a seamless shopping experience. The page is designed with a clean and responsive layout, featuring sections that detail the company’s history, goals, and commitment to customer satisfaction. It also includes contact information and links to social media for users to stay connected.

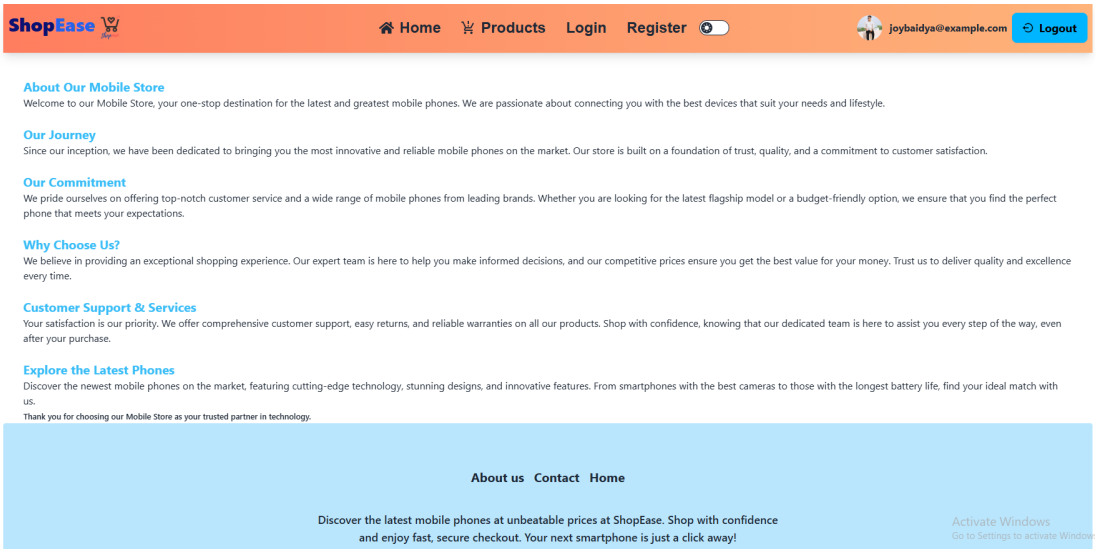


Figure 2.6: About Us Page

2.2.7 Contact Us Page

2.7 figure explainsThe ”Contact Us” page provides users with multiple ways to reach out for support or inquiries. It includes a simple contact form with fields for name, email, subject, and message, making it easy for users to send their questions or feedback. Additionally, the page displays contact details such as phone numbers, email addresses, and the physical address, along with social media links for further communication. Fully responsive, the page ensures a seamless experience across all devices, helping users quickly get in touch.

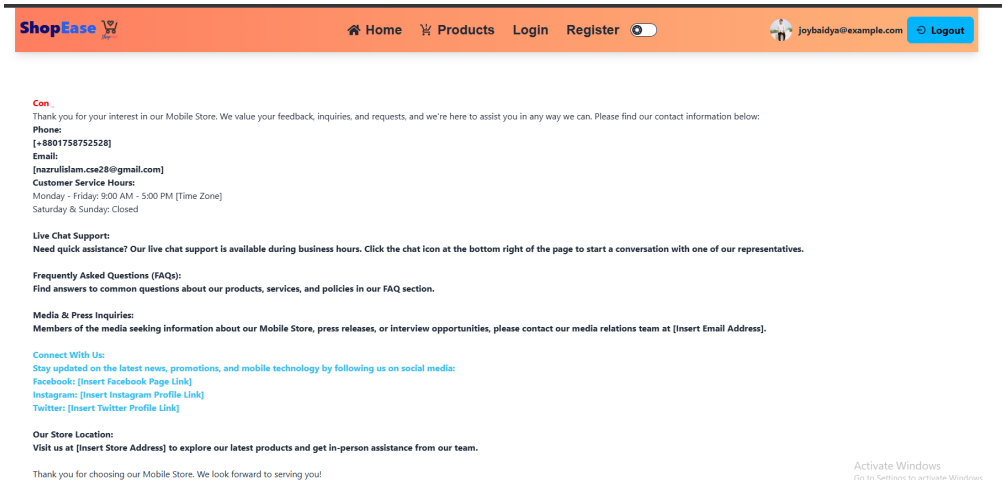


Figure 2.7: Contact Us Page

2.2.8 Footer Page

2.8 figure explains The footer section of the website is designed to provide essential information and navigation links in a clean and organized layout. It includes quick links to important pages, contact information, social media icons for easy connectivity, and copyright details. Fully responsive, the footer adapts seamlessly to different screen sizes, ensuring consistent usability across all devices. It serves as a concluding element of the website, offering users convenient access to additional resources and support.

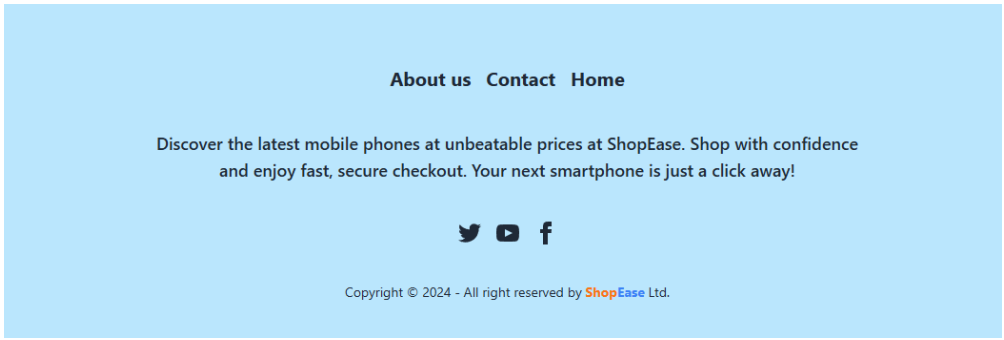


Figure 2.8: Footer Page

2.3 Testing Details

2.3.1 Test Case (Manual Testing)

A **test case** is a detailed document or set of steps designed to verify whether a specific feature or functionality of a software application is working as intended. It includes input conditions, execution steps, expected results, and actual results. Test cases serve as a guide for testers to systematically evaluate a system and ensure it meets requirements.

Components of a Test Case

- **Test Case ID** A unique identifier for the test case.
- **Test Description** A summary of the purpose of the test.
- **Preconditions** Any setup or conditions that must be met before the test is executed.
- **Test Steps** A sequence of actions to be performed.
- **Test Data** The inputs required for the test.
- **Expected Result** The outcome expected if the software works as intended.
- **Actual Result** The outcome observed during testing.
- **Pass/Fail Status** Indicates whether the test case passed or failed.

Manual Test Case

A manual test case is a type of test case executed manually by a tester without the use of automation tools. Testers follow the predefined steps to verify whether the application behaves as expected.

Example of a Manual Test Case

Test Case ID TC_001

Test Description Verify login functionality with valid credentials.

Preconditions

- User is on the login page.

Test Steps

a. Enter a valid username in the ”Username” field.

b. Enter a valid password in the ”Password” field.

c. Click the ”Login” button.

Test Data

- Username: `testuser`
- Password: `password123`

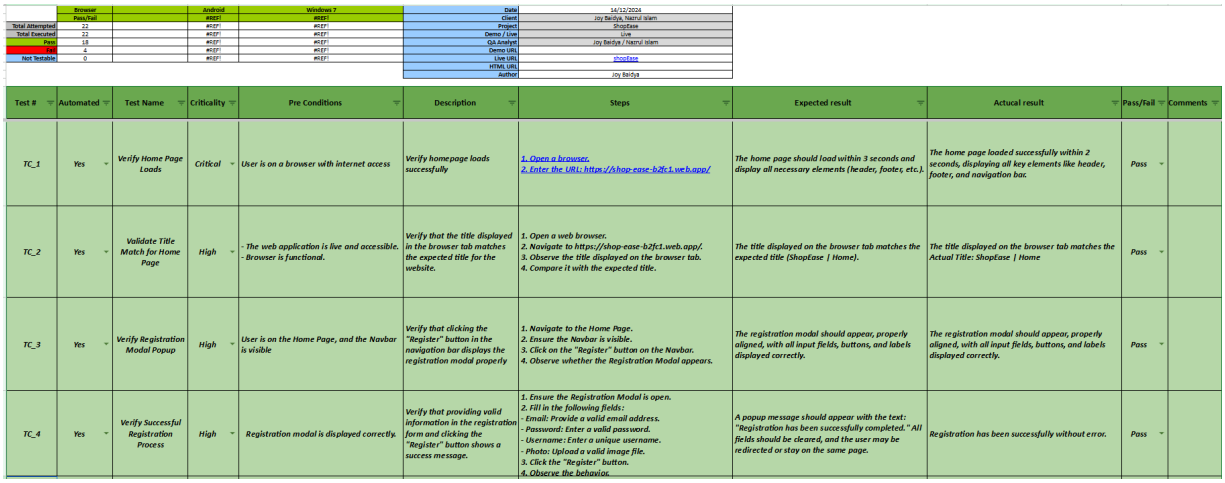
Expected Result

- User is redirected to the homepage.
- A welcome message is displayed.

Actual Result (To be filled in during execution.)

Status (To be marked as Pass/Fail based on the actual result.)

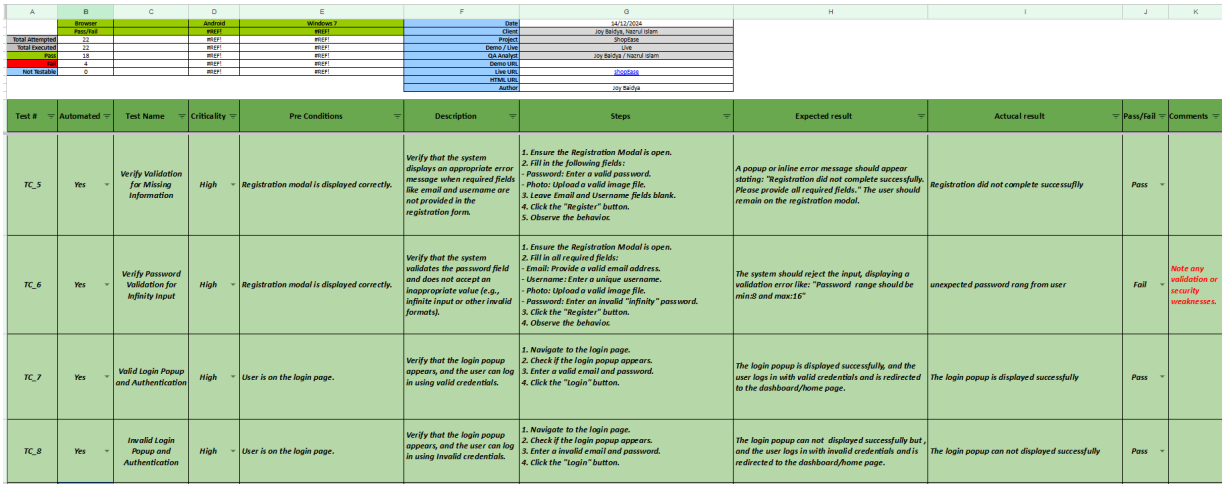
Test case (Manual Test for ShopEase website)



	Browser		Android	Windows 7	Date	14/12/2024
	Pass/Fail		#REF!	#REF!	Client	Joy Baidya, Nazrul Islam
Total Attempted	22		#REF!	#REF!	Project	Shopease
Total Executed	22		#REF!	#REF!	Demo / Live	Live
Pass	18		#REF!	#REF!	QA Analyst	Joy Baidya / Nazrul Islam
Fail	4		#REF!	#REF!	Demo URL	
Not Testable	0		#REF!	#REF!	Live URL	shopease
					HTML URL	
					Author	Joy Baidya

Test #	Automated	Test Name	Criticality	Pre Conditions	Description	Steps	Expected result	Actucal result	Pass/Fail	Comments
TC_1	Yes	Verify Home Page Loads	Critical	User is on a browser with internet access	Verify homepage loads successfully	1. Open a browser. 2. Enter the URL: https://shop-ease-b2fc1.web.app/	The home page should load within 3 seconds and display all necessary elements (header, footer, etc.).	The home page loaded successfully within 2 seconds, displaying all key elements like header, footer, and navigation bar.	Pass	
TC_2	Yes	Validate Title Match for Home Page	High	- The web application is live and accessible. - Browser is functional.	Verify that the title displayed in the browser tab matches the expected title for the website.	1. Open a web browser. 2. Navigate to https://shop-ease-b2fc1.web.app/. 3. Observe the title displayed on the browser tab. 4. Compare it with the expected title.	The title displayed on the browser tab matches the expected title (ShopEase | Home).	The title displayed on the browser tab matches the Actual Title: ShopEase | Home	Pass	
TC_3	Yes	Verify Registration Modal Popup	High	User is on the Home Page, and the Navbar is visible	Verify that clicking the "Register" button in the navigation bar displays the registration modal properly	1. Navigate to the Home Page. 2. Ensure the Navbar is visible. 3. Click on the "Register" button on the Navbar. 4. Observe whether the Registration Modal appears.	The registration modal should appear, properly aligned, with all input fields, buttons, and labels displayed correctly.	The registration modal should appear, properly aligned, with all input fields, buttons, and labels displayed correctly.	Pass	
TC_4	Yes	Verify Successful Registration Process	High	Registration modal is displayed correctly.	Verify that providing valid information in the registration form and clicking the "Register" button shows a success message.	1. Ensure the Registration Modal is open. 2. Fill in the following fields: - Email: Provide a valid email address. - Password: Enter a valid password. - Username: Enter a unique username. - Photo: Upload a valid image file. 3. Click the "Register" button. 4. Observe the behavior.	A popup message should appear with the text: "Registration has been successfully completed." All fields should be cleared, and the user may be redirected or stay on the same page.	Registration has been successfully without error.	Pass	

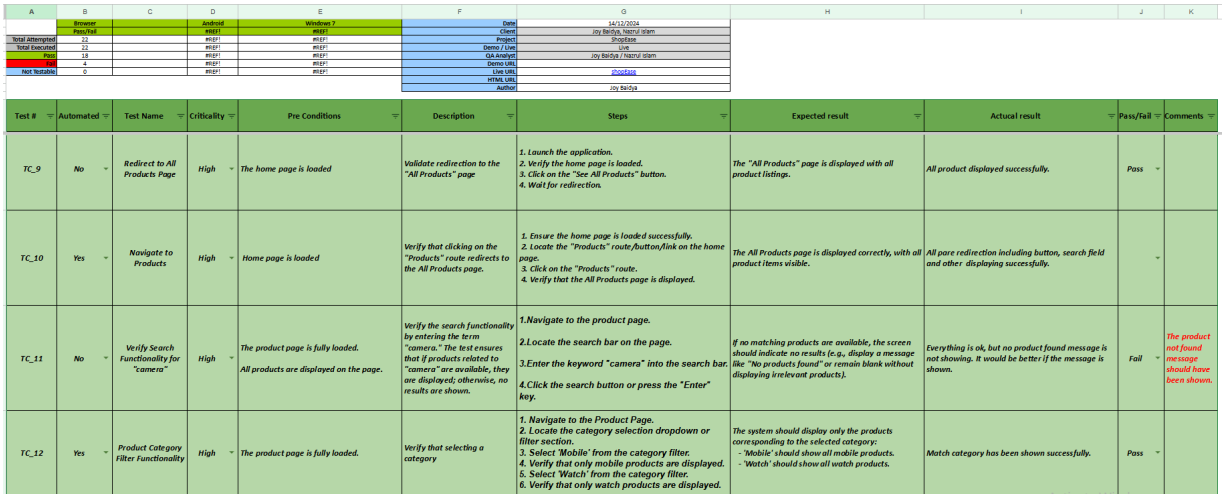
Figure 2.9: Test case (Manual Test for ShopEase website)



	Browser		Android	Windows 7	Date	14/12/2024
	Pass/Fail		#REF!	#REF!	Client	Joy Baidya, Nazrul Islam
Total Attempted	22		#REF!	#REF!	Project	Shopease
Total Executed	22		#REF!	#REF!	Demo / Live	Live
Pass	18		#REF!	#REF!	QA Analyst	Joy Baidya / Nazrul Islam
Fail	4		#REF!	#REF!	Demo URL	
Not Testable	0		#REF!	#REF!	Live URL	shopease
					HTML URL	
					Author	Joy Baidya

Test #	Automated	Test Name	Criticality	Pre Conditions	Description	Steps	Expected result	Actucal result	Pass/Fail	Comments
TC_5	Yes	Verify Validation for Missing Information	High	Registration modal is displayed correctly.	Verify that the system displays an appropriate error message when required fields like email and username are not provided in the registration form.	1. Ensure the Registration Modal is open. 2. Fill in the following fields: - Password: Enter a valid password. - Photo: Upload a valid image file. 3. Leave Email and Username fields blank. 4. Click the "Register" button. 5. Observe the behavior.	A popup or inline error message should appear stating: "Registration did not complete successfully. Please provide all required fields." The user should remain on the registration modal.	Registration did not complete successuflly	Pass	
TC_6	Yes	Verify Password Validation for Infinity Input	High	Registration modal is displayed correctly.	Verify that the system validates the password field and does not accept an inappropriate value (e.g., infinite input or other invalid formats).	1. Ensure the Registration Modal is open. 2. Fill in all required fields: - Email: Provide a valid email address. - Username: Enter a unique username. - Photo: Upload a valid image file. - Password: Enter an invalid "infinity" password. 3. Click the "Register" button. 4. Observe the behavior.	The system should reject the input, displaying a validation error like: "Password range should be min:8 and max:16"	unexpected password rang from user	Fail	Note any validation or security weaknesses.
TC_7	Yes	Valid Login Popup and Authentication	High	User is on the login page.	Verify that the login popup appears, and the user can log in using valid credentials.	1. Navigate to the login page. 2. Check if the login popup appears. 3. Enter a valid email and password. 4. Click the "Login" button.	The login popup is displayed successfully, and the user logs in with valid credentials and is redirected to the dashboard/home page.	The login popup is displayed successfully	Pass	
TC_8	Yes	Invalid Login Popup and Authentication	High	User is on the login page.	Verify that the login popup appears, and the user can log in using Invalid credentials.	1. Navigate to the login page. 2. Check if the login popup appears. 3. Enter a invalid email and password. 4. Click the "Login" button.	The login popup can not displayed successfully but , and the user logs in with invalid credentials and is redirected to the dashboard/home page.	The login popup can not displayed successfully	Pass	

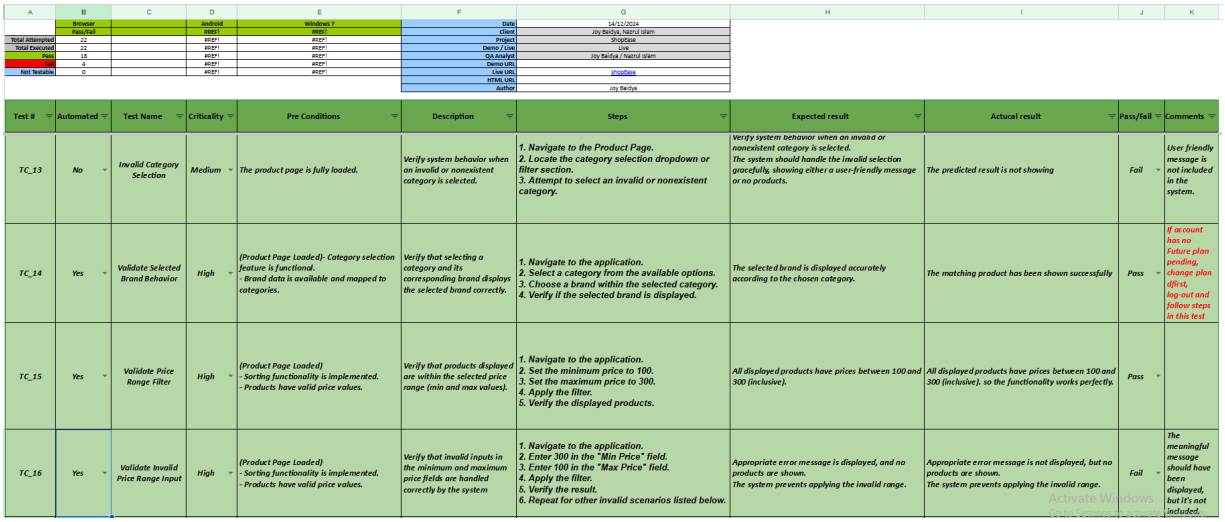
Figure 2.10: Test case (Manual Test for ShopEase website)



	Browser		Android	Windows 7	Date	14/12/2024					
	Pass/Fail		#REF!	#REF!	Client	Joy Baidya, Nazrul Islam					
Total Attempted	22		#REF!	#REF!	Project	Shopease					
Total Executed	22		#REF!	#REF!	Demo / Live	Live					
Pass	18		#REF!	#REF!	QA Analyst	Joy Baidya / Nazrul Islam					
Fail	4		#REF!	#REF!	Demo URL						
Not Testable	0		#REF!	#REF!	Live URL	shopease					
					HTML URL						
					Author	Joy Baidya					

Test #	Automated	Test Name	Criticality	Pre Conditions	Description	Steps	Expected result	Actucal result	Pass/Fail	Comments
TC_9	No	Redirect to All Products Page	High	The home page is loaded	Validate redirection to the "All Products" page	1. Launch the application. 2. Verify the home page is loaded. 3. Click on the "See All Products" button. 4. Wait for redirection.	The "All Products" page is displayed with all product listings.	All product displayed successfully.	Pass	
TC_10	Yes	Navigate to Products	High	Home page is loaded	Verify that clicking on the "Products" route redirects to the All Products page.	1. Ensure the home page is loaded successfully. 2. Locate the "Products" route/button/link on the home page. 3. Click on the "Products" route. 4. Verify that the All Products page is displayed.	The All Products page is displayed correctly, with all product items visible.	All pare redirection including button, search field and other displaying successfully.		
TC_11	No	Verify Search Functionality for "camera"	High	The product page is fully loaded. All products are displayed on the page.	Verify the search functionality by entering the term "camera." The test ensures that if products related to "camera" are available, they are displayed; otherwise, no results are shown.	1.Navigate to the product page. 2.Locate the search bar on the page. 3.Enter the keyword "camera" into the search bar. 4.Click the search button or press the "Enter" key.	If no matching products are available, the screen should indicate no results (e.g., display a message like "No products found" or remain blank without displaying irrelevant products).	Everything is ok, but no product found message is not showing. It would be better if the message is shown.	Fail	The product not found message should have been shown.
TC_12	Yes	Product Category Filter Functionality	High	The product page is fully loaded.	Verify that selecting a category	1. Navigate to the Product Page. 2. Locate the category selection dropdown or filter section. 3. Select 'Mobile' from the category filter. 4. Verify that only mobile products are displayed. 5. Select 'Watch' from the category filter. 6. Verify that only watch products are displayed.	The system should display only the products corresponding to the selected category: - 'Mobile' should show all mobile products. - 'Watch' should show all watch products.	Match category has been shown successfully.	Pass	

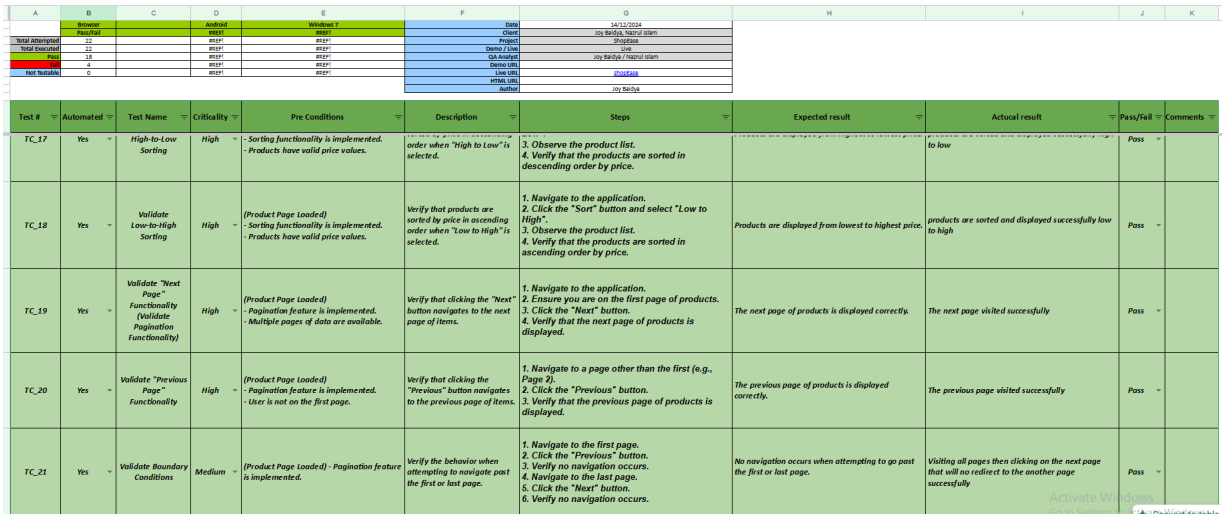
Figure 2.11: Test case (Manual Test for ShopEase website)



	Browser		Android	Windows 7	Date	14/12/2024					
	Pass/Fail		#REF!	#REF!	Client	Joy Baidya, Nazrul Islam					
Total Attempted	22		#REF!	#REF!	Project	Shopease					
Total Executed	22		#REF!	#REF!	Demo / Live	Live					
Pass	18		#REF!	#REF!	QA Analyst	Joy Baidya / Nazrul Islam					
Fail	4		#REF!	#REF!	Demo URL						
Not Testable	0		#REF!	#REF!	Live URL	shopease					
					HTML URL						
					Author	Joy Baidya					

Test #	Automated	Test Name	Criticality	Pre Conditions	Description	Steps	Expected result	Actucal result	Pass/Fail	Comments
TC_13	No	Invalid Category Selection	Medium	The product page is fully loaded.	Verify system behavior when an invalid or nonexistent category is selected.	1. Navigate to the Product Page. 2. Locate the category selection dropdown or filter section. 3. Attempt to select an invalid or nonexistent category.	Verify system behavior when an invalid or nonexistent category is selected. The system should handle the invalid selection gracefully, showing either a user-friendly message or no products.	The predicted result is not showing	Fail	User friendly message is not included in the system.
TC_14	Yes	Validate Selected Brand Behavior	High	(Product Page Loaded)- Category selection feature is functional. - Brand data is available and mapped to categories.	Verify that selecting a category and its corresponding brand displays the selected brand correctly.	1. Navigate to the application. 2. Select a category from the available options. 3. Choose a brand within the selected category. 4. Verify if the selected brand is displayed.	The selected brand is displayed accurately according to the chosen category.	The matching product has been shown successfully	Pass	If account has no Future plan pending, change plan dfirst, log-out and follow steps in this test
TC_15	Yes	Validate Price Range Filter	High	(Product Page Loaded) - Sorting functionality is implemented. - Products have valid price values.	Verify that products displayed are within the selected price range (min and max values).	1. Navigate to the application. 2. Set the minimum price to 100. 3. Set the maximum price to 300. 4. Apply the filter. 5. Verify the displayed products.	All displayed products have prices between 100 and 300 (inclusive).	All displayed products have prices between 100 and 300 (inclusive). so the functionality works perfectly.	Pass	
TC_16	Yes	Validate Invalid Price Range Input	High	(Product Page Loaded) - Sorting functionality is implemented. - Products have valid price values.	Verify that invalid inputs in the minimum and maximum price fields are handled correctly by the system	1. Navigate to the application. 2. Enter 300 in the "Min Price" field. 3. Enter 100 in the "Max Price" field. 4. Apply the filter. 5. Verify the result. 6. Repeat for other invalid scenarios listed below.	Appropriate error message is displayed, and no products are shown. The system prevents applying the invalid range.	Appropriate error message is not displayed, but no products are shown. The system prevents applying the invalid range.	Fail	The meaningful message should have been displayed, but it's not included.

Figure 2.12: Test case (Manual Test for ShopEase website)



	Browser		Android	Windows 7
	Pass/Fail		#REF!	#REF!
Total Attempted	22		#REF!	#REF!
Total Executed	22		#REF!	#REF!
Pass	18		#REF!	#REF!
Fail	4		#REF!	#REF!
Not Testable	0		#REF!	#REF!

Date	14/12/2024
Client	Joy Baidya, Nazrul Islam
Project	Shopease
Demo / Live	Live
QA Analyst	Joy Baidya / Nazrul Islam
Demo URL	
Live URL	shopease
HTML URL	
Author	Joy Baidya

Test #	Automated	Test Name	Criticality	Pre Conditions	Description	Steps	Expected result	Actucal result	Pass/Fail	Comments
TC_17	Yes	High-to-Low Sorting	High	- Sorting functionality is implemented. - Products have valid price values.	order when "High to Low" is selected.	3. Observe the product list. 4. Verify that the products are sorted in descending order by price.		to low	Pass	
TC_18	Yes	Validate Low-to-High Sorting	High	(Product Page Loaded) - Sorting functionality is implemented. - Products have valid price values.	Verify that products are sorted by price in ascending order when "Low to High" is selected.	1. Navigate to the application. 2. Click the "Sort" button and select "Low to High". 3. Observe the product list. 4. Verify that the products are sorted in ascending order by price.	Products are displayed from lowest to highest price.	products are sorted and displayed successfully low to high	Pass	
TC_19	Yes	Validate "Next Page" Functionality (Validate Pagination Functionality)	High	(Product Page Loaded) - Pagination feature is implemented. - Multiple pages of data are available.	Verify that clicking the "Next" button navigates to the next page of items.	1. Navigate to the application. 2. Ensure you are on the first page of products. 3. Click the "Next" button. 4. Verify that the next page of products is displayed.	The next page of products is displayed correctly.	The next page visited successfully	Pass	
TC_20	Yes	Validate "Previous Page" Functionality	High	(Product Page Loaded) - Pagination feature is implemented. - User is not on the first page.	Verify that clicking the "Previous" button navigates to the previous page of items.	1. Navigate to a page other than the first (e.g., Page 2). 2. Click the "Previous" button. 3. Verify that the previous page of products is displayed.	The previous page of products is displayed correctly.	The previous page visited successfully	Pass	
TC_21	Yes	Validate Boundary Conditions	Medium	(Product Page Loaded) - Pagination feature is implemented.	Verify the behavior when attempting to navigate past the first or last page.	1. Navigate to the first page. 2. Click the "Previous" button. 3. Verify no navigation occurs. 4. Navigate to the last page. 5. Click the "Next" button. 6. Verify no navigation occurs.	No navigation occurs when attempting to go past the first or last page.	Visiting all pages then clicking on the next page that will no redirect to the another page successfully	Pass	

Figure 2.13: Test case (Manual Test for ShopEase website)

	Browser		Android	Windows 7
	Pass/Fail		#REF!	#REF!
Total Attempted	22		#REF!	#REF!
Total Executed	22		#REF!	#REF!
Pass	18		#REF!	#REF!
Fail	4		#REF!	#REF!
Not Testable	0		#REF!	#REF!

Date	14/12/2024
Client	Joy Baidya, Nazrul Islam
Project	Shopease
Demo / Live	Live
QA Analyst	Joy Baidya / Nazrul Islam
Demo URL	
Live URL	shopease
HTML URL	
Author	Joy Baidya

Test #	Automated	Test Name	Criticality	Pre Conditions	Description	Steps	Expected result	Actucal result	Pass/Fail	Comments
		Functionality)								
TC_20	Yes	Validate "Previous Page" Functionality	High	(Product Page Loaded) - Pagination feature is implemented. - User is not on the first page.	Verify that clicking the "Previous" button navigates to the previous page of items.	1. Navigate to a page other than the first (e.g., Page 2). 2. Click the "Previous" button. 3. Verify that the previous page of products is displayed.	The previous page of products is displayed correctly.	The previous page visited successfully	Pass	
TC_21	Yes	Validate Boundary Conditions	Medium	(Product Page Loaded) - Pagination feature is implemented.	Verify the behavior when attempting to navigate past the first or last page.	1. Navigate to the first page. 2. Click the "Previous" button. 3. Verify no navigation occurs. 4. Navigate to the last page. 5. Click the "Next" button. 6. Verify no navigation occurs.	No navigation occurs when attempting to go past the first or last page.	Visiting all pages then clicking on the next page that will no redirect to the another page successfully	Pass	
TC_22	Yes	Validate Dark Mode Toggle On	High	- Dark mode functionality is implemented. - Toggle button is visible and clickable.	Verify that clicking the dark mode button toggles the screen to dark mode.	1. Navigate to the application. 2. Locate the "Dark Mode" toggle button. 3. Click the toggle button. 4. Verify that the screen switches to dark mode.	The screen switches to dark mode (background and text colors are updated accordingly)	The button toggle successfully and the screen switches to dark.	Pass	
TC_23	Yes	Validate Dark Mode Toggle Off	High	- Dark mode is currently active.	Verify that clicking the dark mode button again toggles the screen back to light mode.	1. While in dark mode, click the toggle button again. 2. Verify that the screen switches back to light mode.	The screen switches back to light mode (default background and text colors are restored).	The button toggle successfully and the screen switches to light	Pass	

Figure 2.14: Test case (Manual Test for ShopEase website)

2.3.2 Performance Testing

2.15 figure describes Performance Testing types used to evaluate system behavior under different conditions. Load Testing checks performance under expected workload, while Stress Testing examines performance under extreme loads. Endurance Testing (Soak Testing) assesses long-duration stability, and Spike Testing analyzes system response to sudden user surges. Volume Testing evaluates performance with large data volumes.

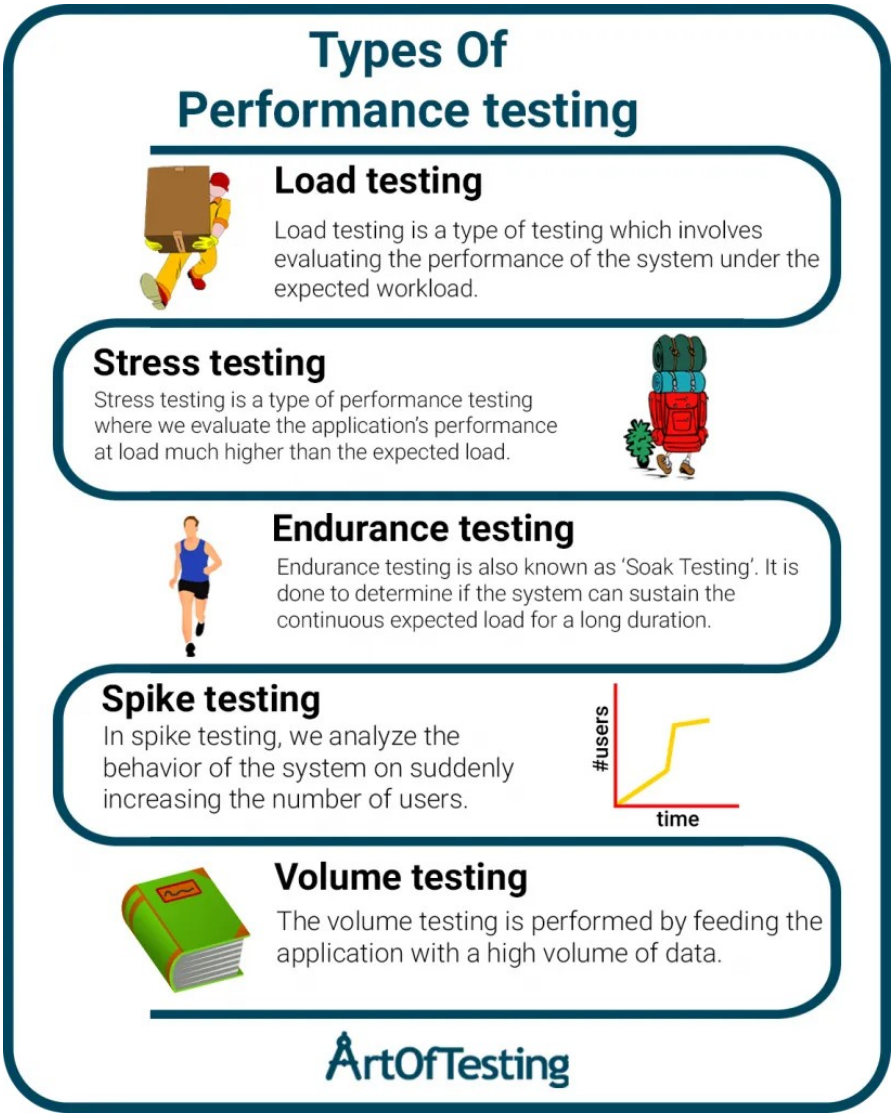


Figure 2.15: Performance Testing

How to Perform Performance Testing?

Conducting Performance Testing on a Web Application using Apache JMeter involves the following steps:

Steps to Perform Performance Testing in JMeter

- **Test Plan:** Create a Test Plan as the container for your tests.
- **Thread Group:** Add a Thread Group to define virtual users and ramp-up time.
- **HTTP Requests:** Include HTTP Request elements to simulate user actions.

- **Assertions:** Validate responses with assertions to ensure functionality.
- **Timers:** Add Timers to simulate real-world user delays.
- **Listeners:** Use Listeners to collect and analyze test data.
- **Parameters:** Configure threads, ramp-up, and duration for your test.
- **Run Test:** Save and execute the test to simulate user load.
- **Analyze Results:** Review metrics like response time, throughput, and errors.

The Load testing using the Jemter

Test Configuration

- **Tool Used:** Apache JMeter
- **Test Scenario:** Simulating concurrent users to test maximum load capacity.
- **Maximum Load Tested:** 3800 concurrent users
- **Test Duration:** 5 minutes
- **System Under Test (SUT):** ShopEase Website - Product Page
- **Environment:**
  - **Server:** Cloud-based Hosting
  - **Protocol:** HTTP/HTTPS
  - **Network:** Standard Broadband (10 Mbps)

2.3.3 Performance Metrics Observed During Load Testing

The 2.1 table presents **performance metrics from load testing**, including **max concurrent users (4100), response times, error rate (0.7%), throughput (650 requests/sec),**

CPU utilization (85%), and memory usage (78%).It highlights **how the system handles peak load**, with **minor errors, stable memory performance, and high CPU utilization**. Overall, **the website performed efficiently under stress conditions**.

Table 2.1: Performance Metrics Observed During Load Testing

Metric	Value	Observation
Max Concurrent Users	4100	Website performed under peak load.
Average Response Time	2.5 seconds	Slight delay observed at high concurrency.
Peak Response Time	5.8 seconds	Response time peaked under maximum load.
Error Rate	0.7%	Minimal errors occurred during peak load.
Throughput	650 requests/second	Requests processed efficiently.
CPU Utilization	85%	High CPU usage observed during max load.
Memory Usage	78%	Stable memory performance under load.

2.3.4 Load Testing Execution for Website

The 2.16 figure shows a **load testing execution using JMeter**,displaying **HTTP request results, response times, and thread performance**.The interface provides **detailed logs of processed requests**, helping analyze system efficiency. It visually supports the **observations from the table, confirming system behavior under load**.

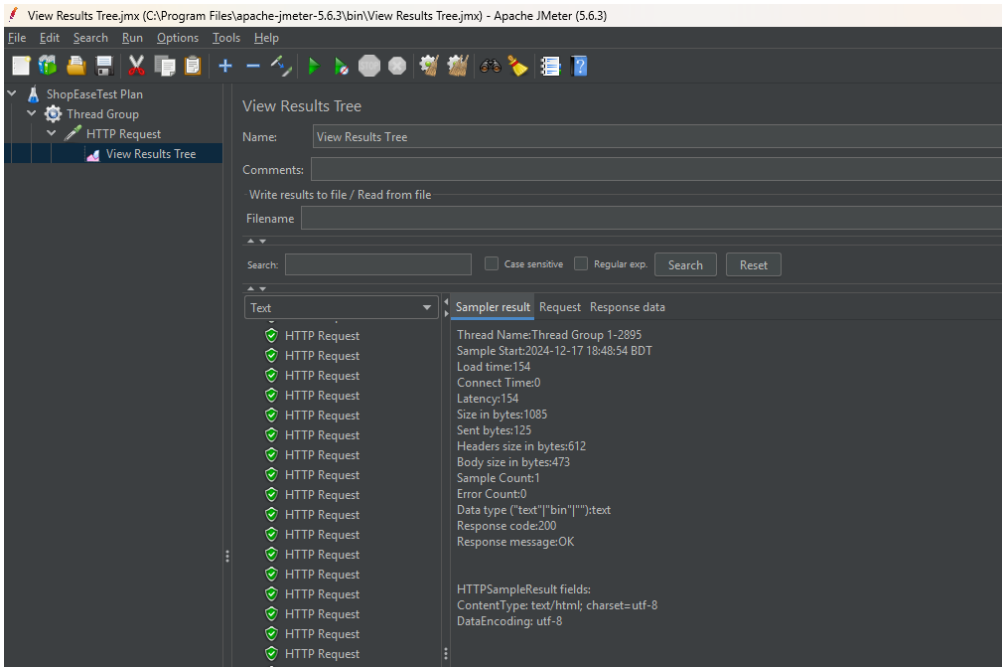


Figure 2.16: The Load testing for web site

2.3.5 Stress Testing with JMeter

Thread Group Settings

The following settings were configured for the stress test:

- **Number of Threads:** 5000
- **Ramp-Up Period:** 60 seconds
- **Loop Count:** Infinite (or 1000 iterations)

HTTP Request Configuration

- **Protocol:** HTTPS
- **Server Name:** www.shopease.com
- **Path:** /products

The 2.17 figure illustrates *stress testing* of a website using **Apache JMeter**, a performance testing tool. The test simulates multiple user requests to analyze system stability under high load conditions. The results highlight performance issues, including system crashes at peak loads.

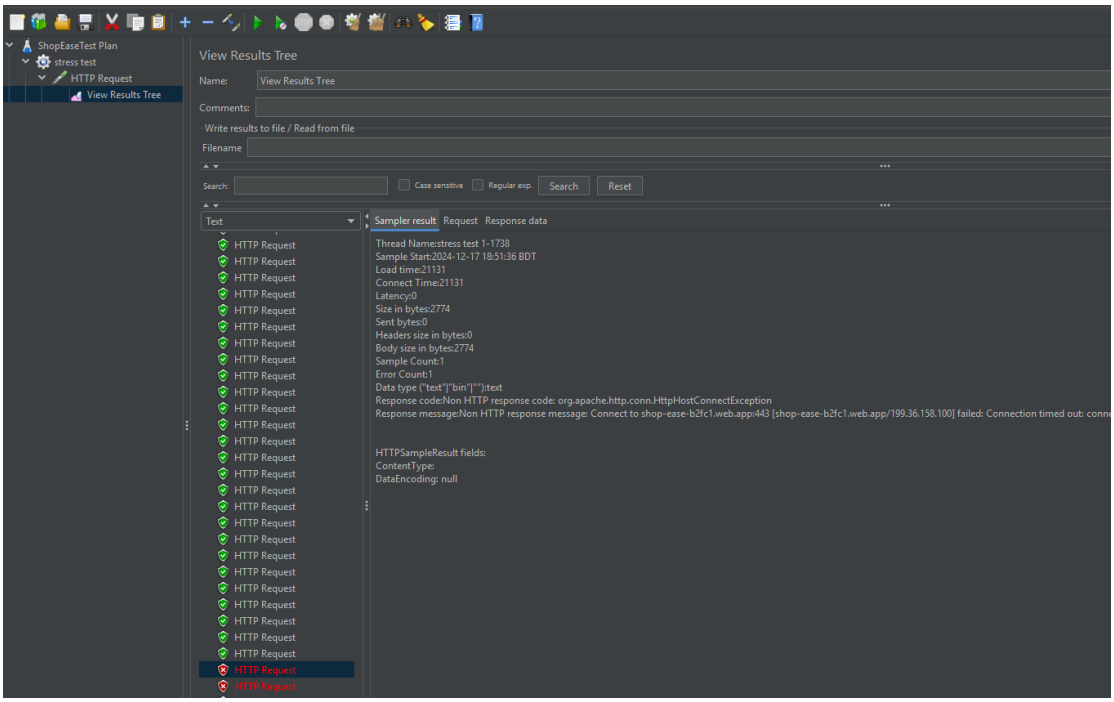


Figure 2.17: The Stress testing for web site

Table 2.2: Stress Test Observations

Metric	**Value**	**Observation**
Maximum Load	4000 users	Server crashed at 4000 concurrent users.
Peak Response Time	8 seconds	Response time peaked under maximum load.
Error Rate	5%	Errors increased significantly after 4500 users.
CPU Utilization	95%	CPU reached critical usage levels.
Memory Usage	90%	Memory usage near capacity under load.

The 2.2 table presents key *stress test metrics*, including **maximum load, response**

time, error rate, CPU, and memory usage. The system crashed at **4000 concurrent users**, and the response time peaked at **8 seconds**. Additionally, **CPU and memory utilization** reached critical levels, causing errors beyond **4500 users**.

2.3.6 Endurance Testing with JMeter

The2.18 figure illustrates Endurance Testing of a website using Apache JMeter. It shows multiple HTTP requests being sent continuously to evaluate system performance over an extended period. The test helps identify potential memory leaks, performance degradation, or failures under prolonged load.

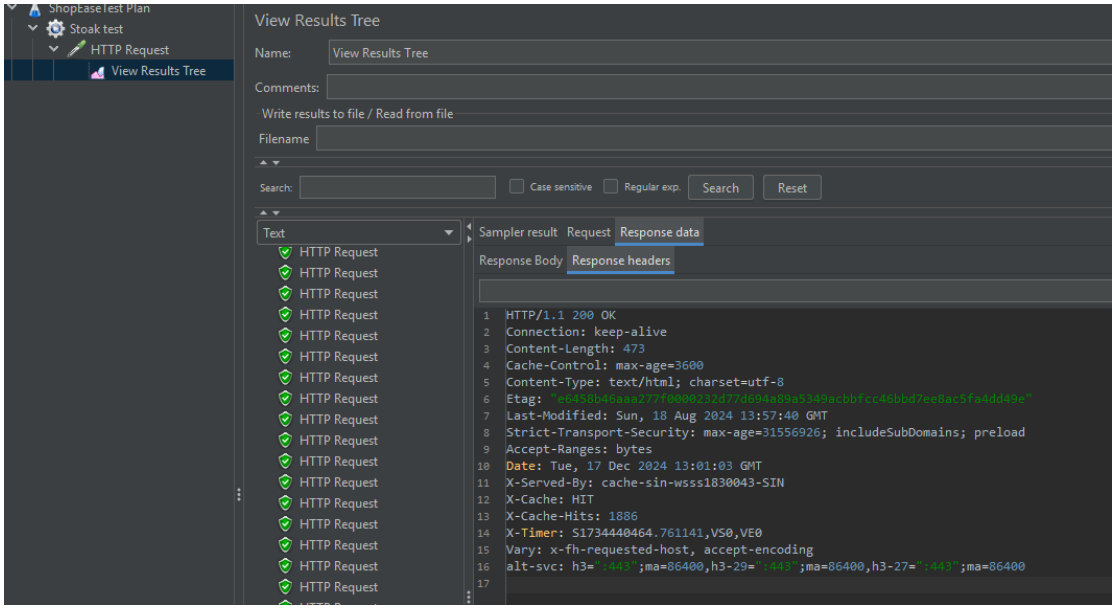


Figure 2.18: The Endurance testing for web site

The 2.3 table endurance test, conducted for **3 hours** with **3800 concurrent users**, demonstrated the system's stability under sustained load. The **error rate was 2%**, showing a slight increase over time. **CPU utilization remained at 85%**, and **memory usage gradually increased to 80%**, indicating stable resource consumption. The **average response time was 4 seconds**, staying within acceptable limits. Most importantly, **the system did not crash**, proving its reliability during prolonged operation.

Table 2.3: Endurance Testing Results with JMeter

Metric	Value	Observation
Test Duration	3 hours	The test ran continuously under sustained load.
Concurrent Users	3800 users	System maintained performance up to 3800 users.
Error Rate	2%	A minor increase in errors after 3 hours.
CPU Utilization	85%	CPU usage remained stable under load.
Memory Usage	80%	Memory consumption increased gradually.
Response Time	4 seconds (average)	Average response time remained within limits.
System Crash	No crash	The system remained stable throughout the test.

2.3.7 Spike testing Testing with JMeter

The 2.19 figer Spike testing with JMeter involves testing a system's reaction to a sudden, sharp increase in load. The test helps to identify the system's ability to handle rapid spikes in traffic, ensuring that it can recover without crashing or significant performance degradation.

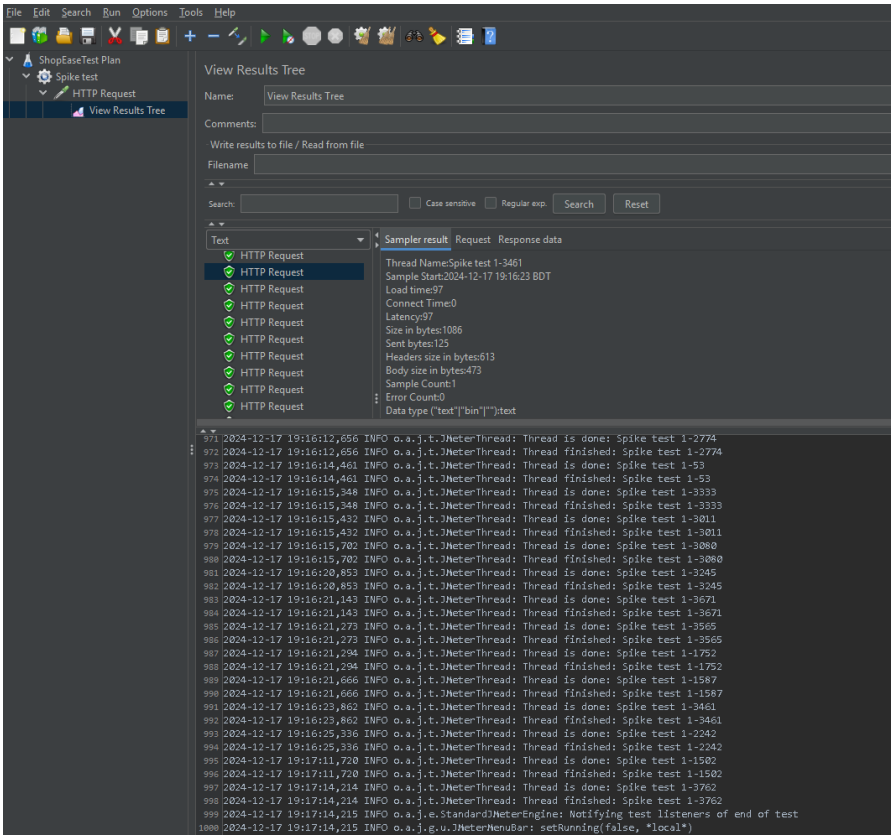


Figure 2.19: The Spike testing for web site

The 2.4 table The spike test involved a sudden increase in load from 500 to 4000 concurrent users, revealing that the system experienced significant response time spikes and error rates due to high CPU and memory utilization, peaking at 98% and 95%, respectively. However, the system recovered within 2 minutes, returning to normal performance after the spike.

Table 2.4: Spike Testing Results with JMeter

Metric	Value	Observation
Initial Load	500 users	System operated normally under baseline load.
Spike Load	4000 users	Sudden increase to 5000 concurrent users.
Response Time	10 seconds (peak)	Response time spiked significantly during the load.
Error Rate	12%	Errors increased sharply during the spike.
CPU Utilization	98%	CPU usage reached critical levels under spike load.
Memory Usage	95%	Memory usage neared full capacity.
System Recovery	2 minutes	System recovered to normal performance post spike.

Performance Testing Total Results

The 2.5 table Shows result for type of performance testing in this web side:

Table 2.5: Performance Testing Results

Test Type	Objective	Load Condition	Result (Users)
Load Testing	Evaluate system performance under expected load.	Steady load up to **3,800 users**	**Successful**
Stress Testing	Test system under extreme load to find its breakpoint.	Pushed beyond **3,850 users**	Crashed at 3850
Endurance Testing	Assess system stability over a prolonged period under normal load.	Continuous **3,800 users** for 3 hours	**Stable**
Spike Testing	Test system behavior under sudden, extreme load increases.	Sudden jump to **3,800 users**	Performance Degraded

2.3.8 API Testing

Steps for Executing API Testing

a. **Create API testing requirements:** Identify the purpose, functionalities, and expected outcomes of the API to ensure proper testing coverage.

b. **Establish the API test environment:** Set up the necessary infrastructure, tools, and configurations to facilitate API testing. **Make a trial API call** Perform an initial API call to verify connectivity and to understand the response structure.

c. **Define the input parameters:** Specify the required inputs such as request headers, body, and query parameters.

d. **Create API test cases:** Develop comprehensive test cases to cover different scenarios such as positive, negative, and edge cases.

API Testing Tools

- Postman

2.20 figer is API testing for this ShopEase

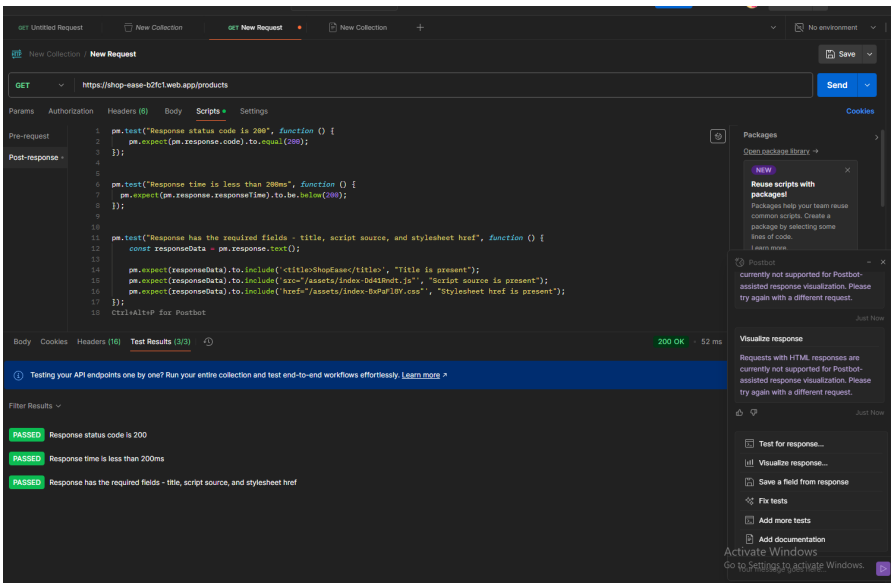


Figure 2.20: API testing for this ShopEase

API Test Description

Request Details

- **Method:** GET
- **Endpoint:** `https://shop-ease-b2fc1.web.app/products`

Tests Performed

a. **Response Status Code Validation:**

- Ensures the response status code is 200.

b. **Response Time Validation:**

- Validates that the response time is less than 200 ms.

c. **Response Content Validation:**

- Verifies that the response contains the following fields:
  - HTML `<title>` tag: `<title>ShopEase</title>`
  - Script source: `src="/assets/index-Dd41Rndt.js"`
  - Stylesheet link: `href="/assets/index-BxPaFlBy.css"`

Test Results

Table 2.6: Test Results

Test Attribute	**Status**	**Details**
Response Code	PASSED	200
Response Time	PASSED	52 ms
Required Fields Validation	PASSED	-

2.3.9 Automation Testing

Automation testing is the use of specialized software tools to automate the execution of test cases, improving efficiency and accuracy. It helps in performing repetitive tests, saves time, and ensures consistent results. However, it requires an initial investment in tools, scripts, and maintenance over time.

Automation Testing Implementation and Result

Figure 2.21 Verify Home Page Loads

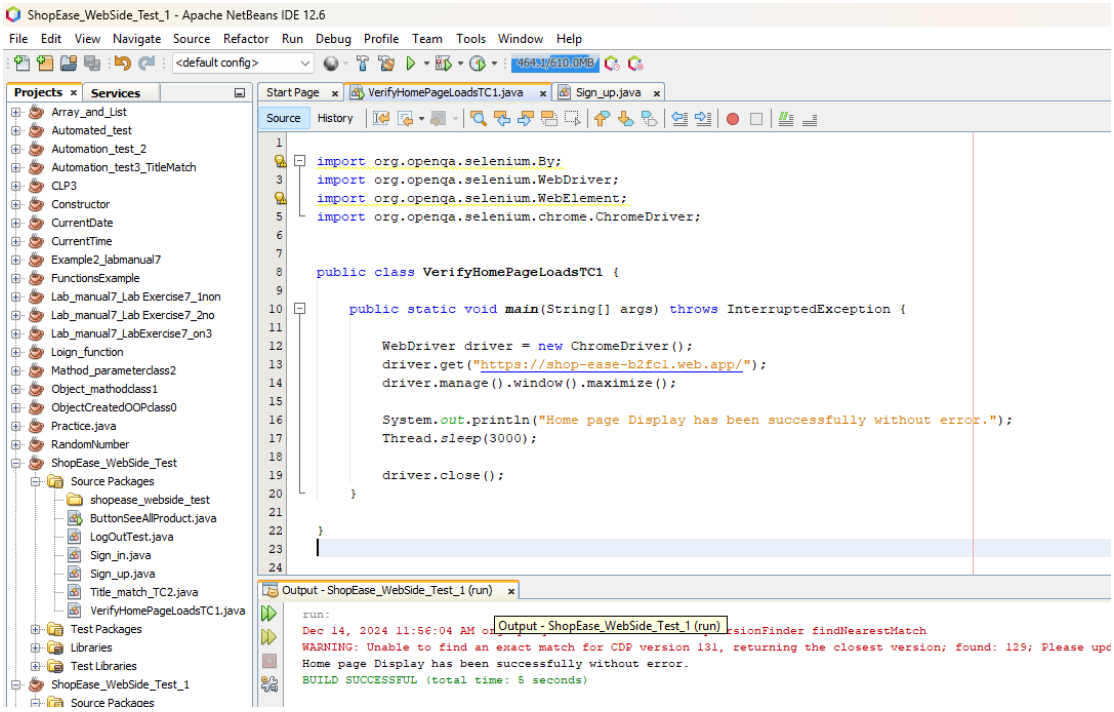


Figure 2.21: Verify Home Page Loads

- **Expected result:**The home page should load within 3 seconds and display all necessary elements (header, footer, etc.)
- **Actucal result:**The home page loaded successfully within 2 seconds, displaying all key elements like header, footer, and navigation bar.
- **Pass:** Expected result **Equal** Actucal result

Figure 2.22 Validate Title Match for Home Page

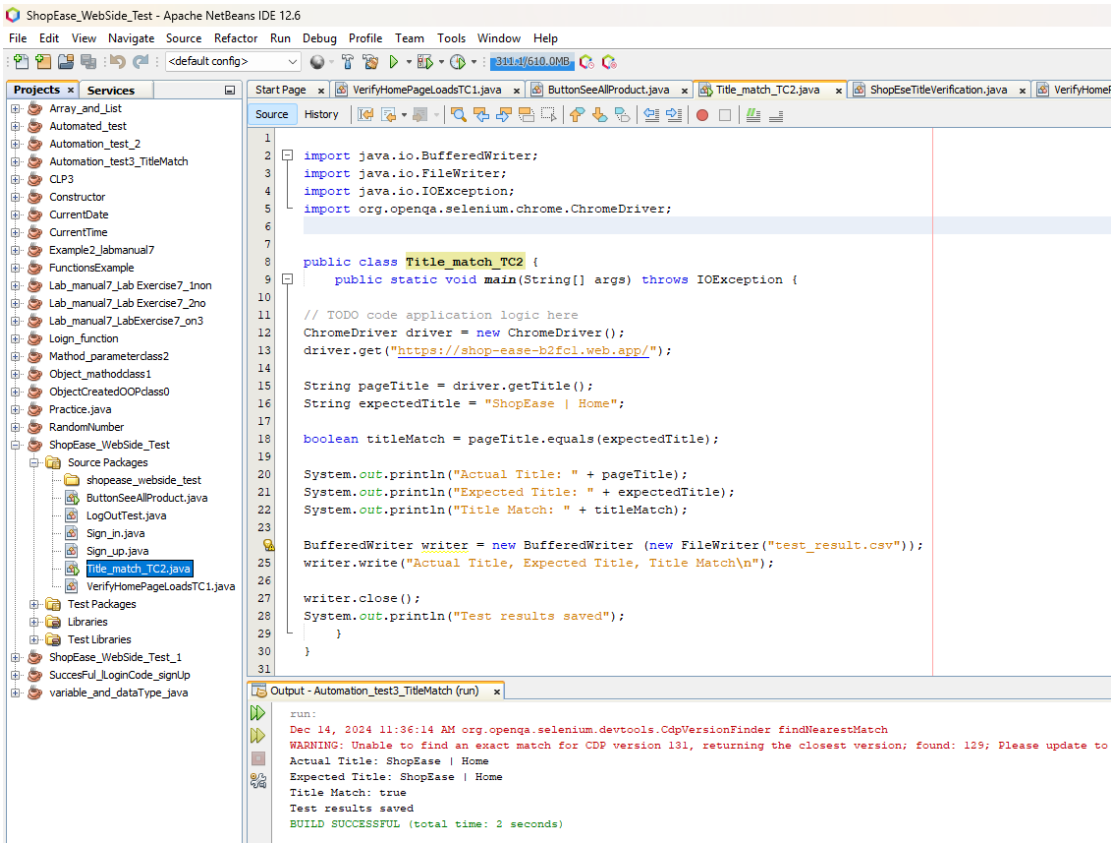


Figure 2.22: Validate Title Match for Home Page

- **Expected result:**he title displayed on the browser tab matches the expected title (ShopEase — Home).
- **Actucal result:**The title displayed on the browser tab matches the Actual Title: ShopEase — Home
- **Pass:** Expected result **Equal** Actucal result

Figure 2.23 Verify Registration Modal Popup

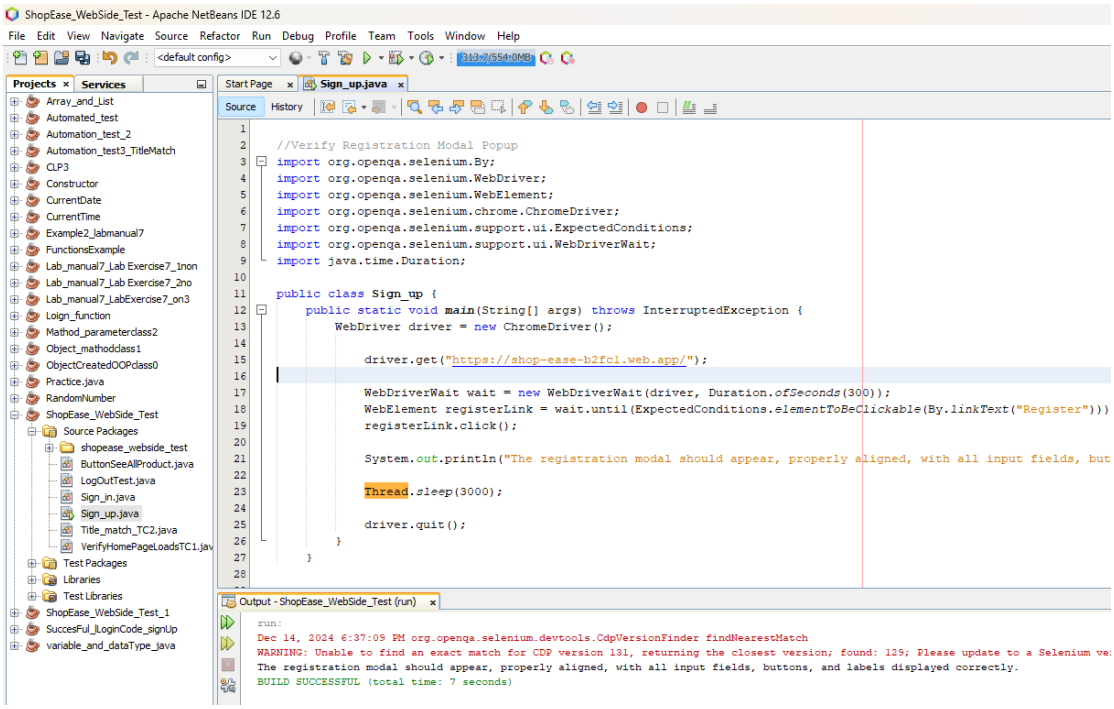


Figure 2.23: Verify Registration Modal Popup

Verify that clicking the ”Register” button in the navigation bar displays the registration modal properly

Figure 2.24 Verify Registration Modal Popup Result

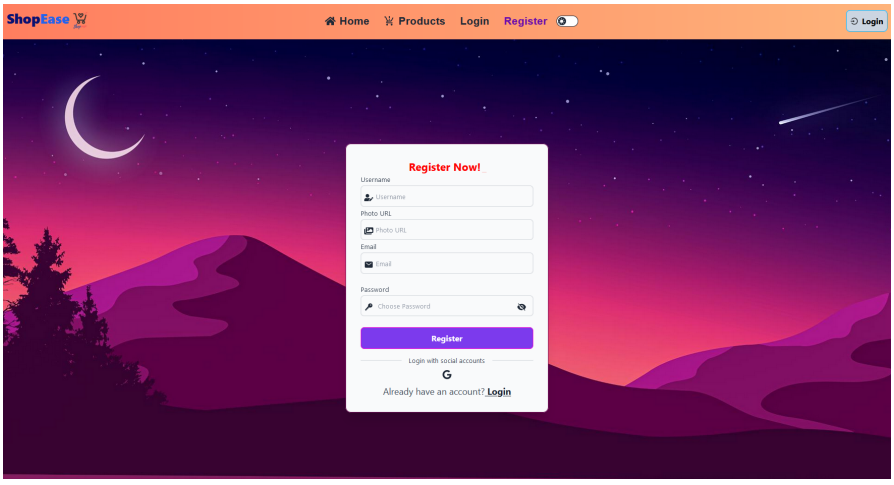


Figure 2.24: Verify Registration Modal Popup Result

- **Expected result:**The registration modal should appear correctly aligned, with all input fields, buttons, and labels correctly displayed.

- **Actucal result:**The registration modal should appear, properly aligned, with all input fields, buttons, and labels displayed correctly.
- **Pass:** Expected result **Equal** Actucal result

Figure 2.25 Verify Successful Registration Process

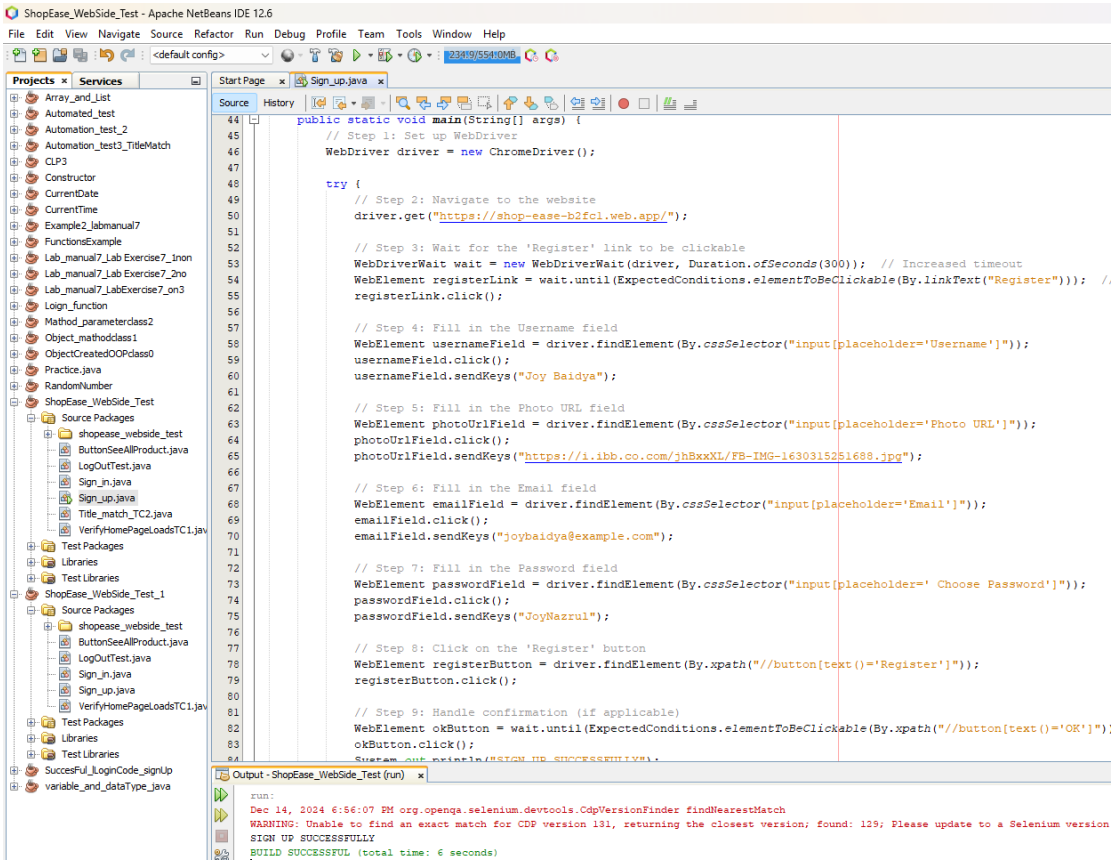


Figure 2.25: Verify Successful Registration Process

Verify that providing valid information in the registration form and clicking the ”Register” button shows a success message.

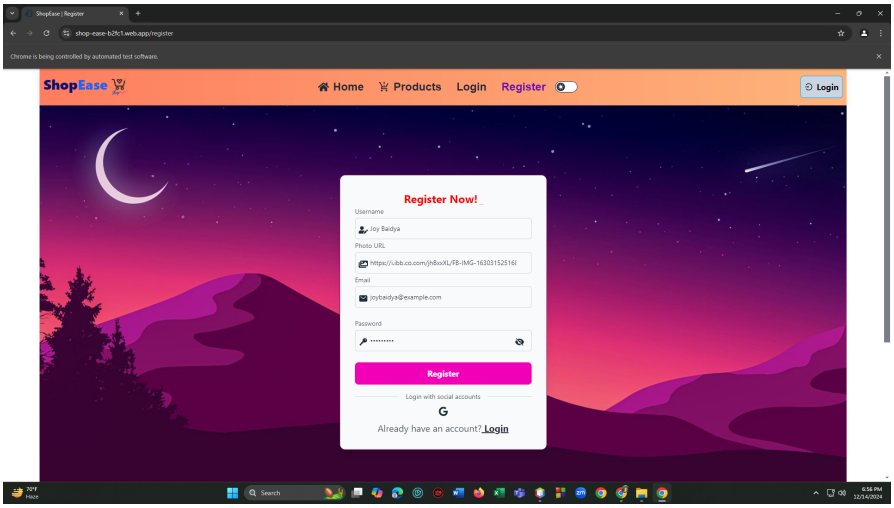


Figure 2.26: Verify Successful Registration Process Result

- **Expected result:**A popup message should appear with the text: ”Registration has been successfully completed.” All fields should be cleared, and the user may be redirected or stay on the same page.
- **Actucal result:**Registration has been successfully without error.
- **Pass:** Expected result **Equal** Actucal result

Figure 2.27 Verify Validation for Missing Information

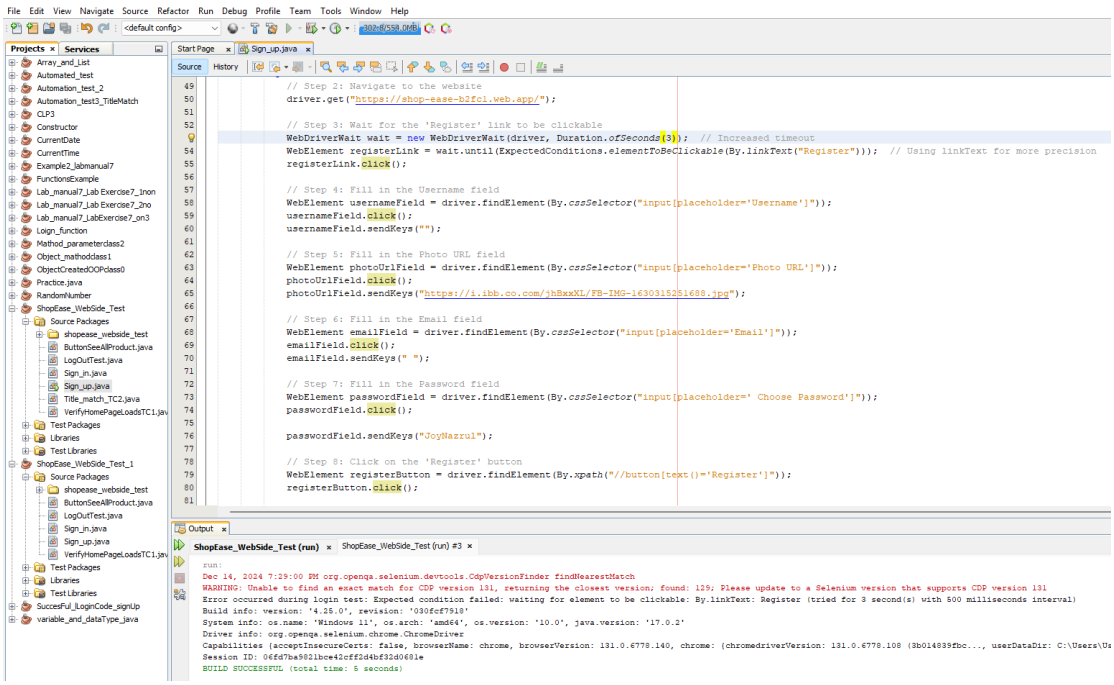


Figure 2.27: Verify Validation for Missing Information

Verify that the system displays an appropriate error message when required fields like email and username are not provided in the registration form.

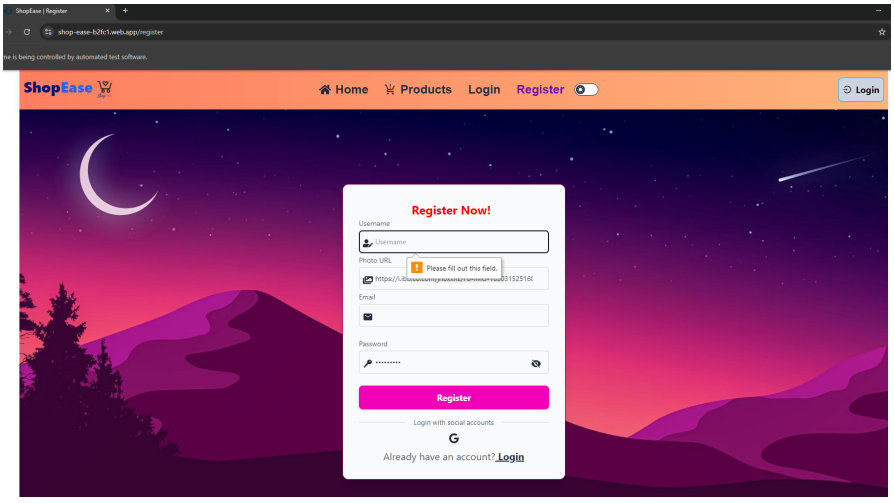


Figure 2.28: Verify Validation for Missing Information Result

- **Expected result:**A popup or inline error message should appear stating: ”Registration did not complete successfully. Please provide all required fields.” The user should remain on the registration modal.

- **Actucal result:**Registration did not complete successuflly
- **Pass:** Expected result **Equal** Actucal result

Figure 2.29 Valid Boundary value testing and Equivalence testing for sign Up

Boundary Value Analysis (BVA) focuses on testing the boundaries of input ranges. The idea is that errors often occur at the edges rather than in the middle of input ranges.

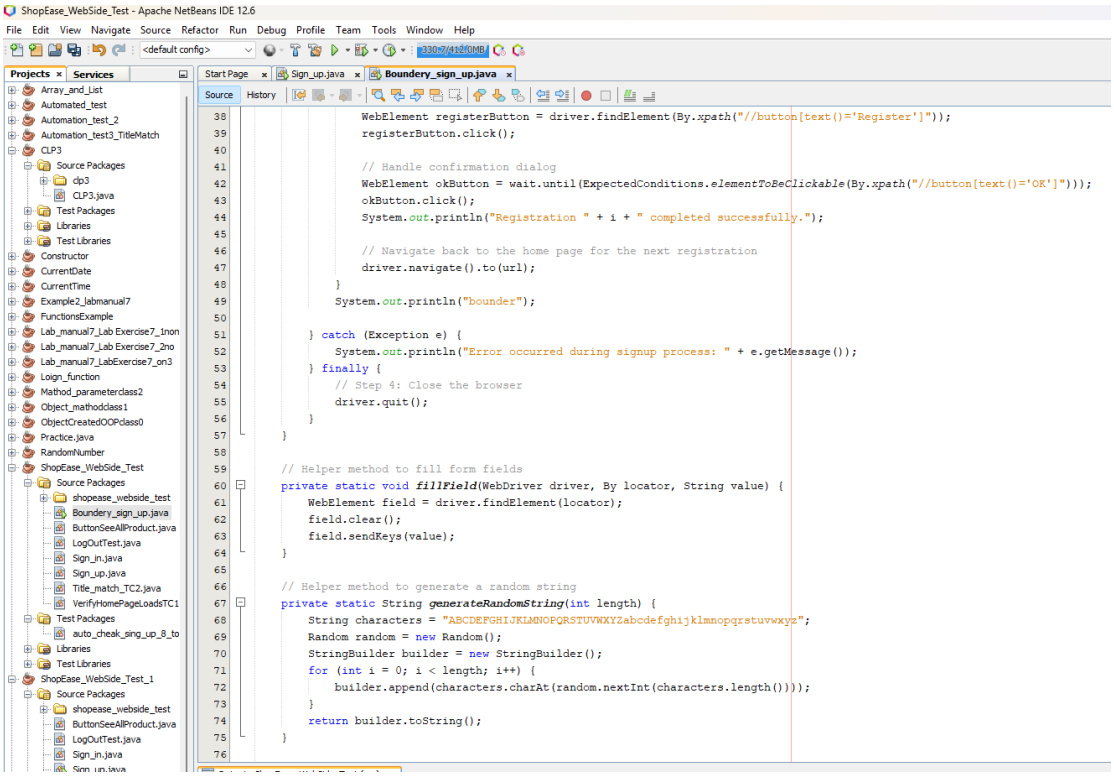


Figure 2.29: Boundary Value Analysis and Equivalence Partitioning

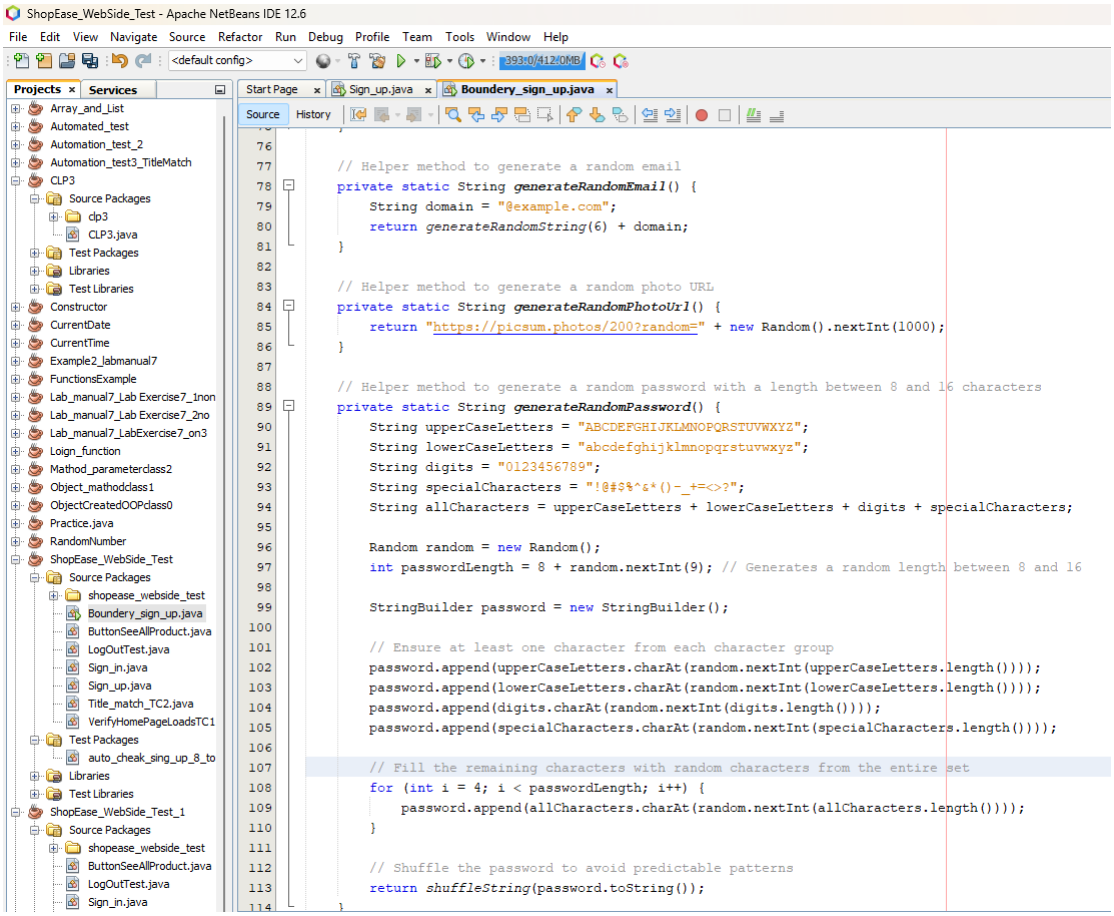


Figure 2.30: Boundary Value Analysis and Equivalence Partitioning

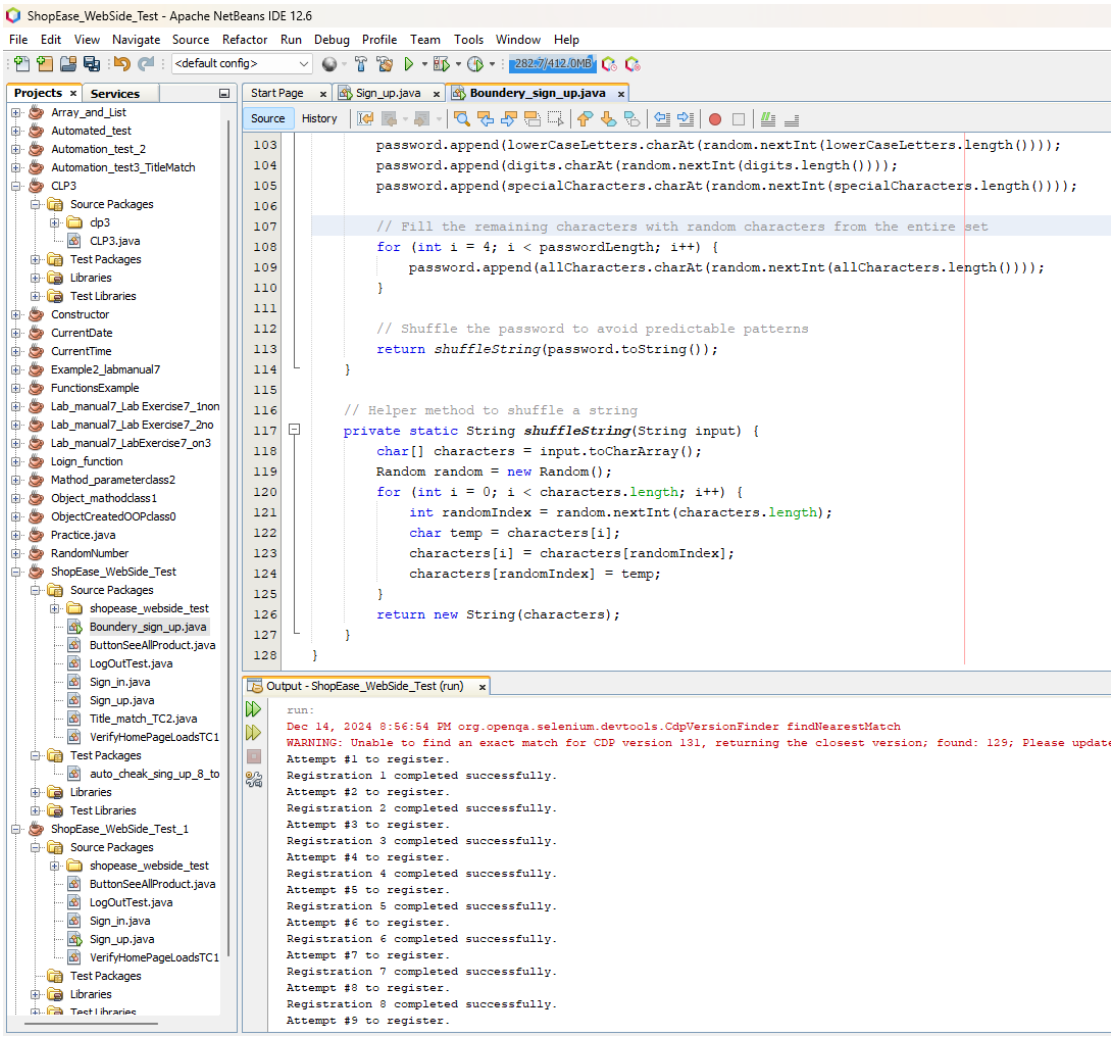


Figure 2.31: Boundary Value Analysis and Equivalence Partitioning successfull Result

- **Expected result:** Valid condition successfull Tase case
- **Actucal result:** Valid condition successfull Tese case
- **Pass:** Expected result **Equal** Actucal result

To catch defects that occur at boundary conditions, which are common places for errors in software.

Figure 2.32 Verify Password Validation for Infinity Input

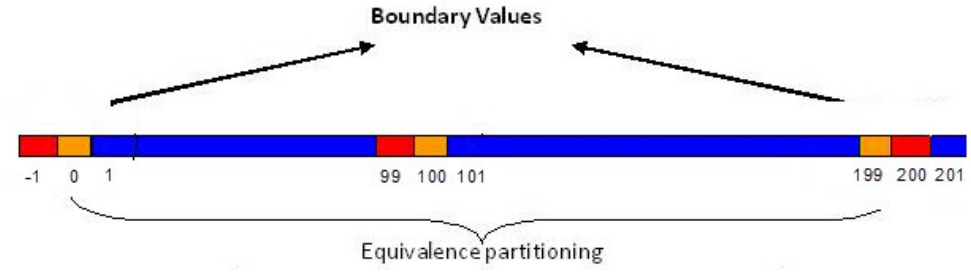


Figure 2.32: Boundary Value Analysis and Equivalence Partitioning

PasswordField.sendKeys: JoyNazrulhjffhjdskfhjkdsfhdshfjkdshfjkdshfljdshfjd-shfjdsklhfjdshlkjdshjdskhfkjdshfjdsfhjdskhjdshjskhfdjsfhjdsfhjdsfhjdskhfjkdsfjdshfjdskfhjdsh

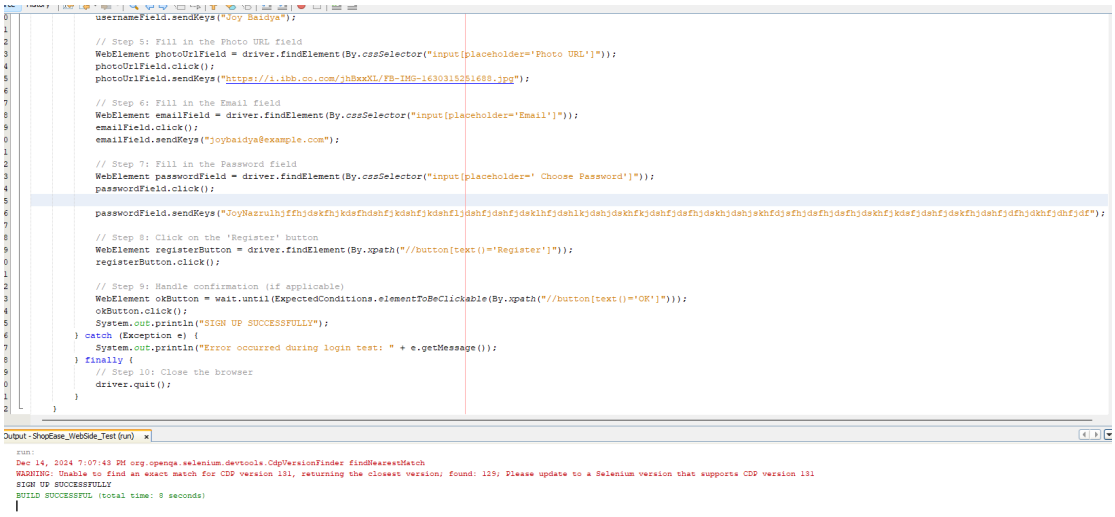


Figure 2.33: Invalid Boundary Value Analysis and Equivalence Partitioning

verify that the system validates the password field and does not accept an inappropriate value (e.g., infinite input or other invalid formats).

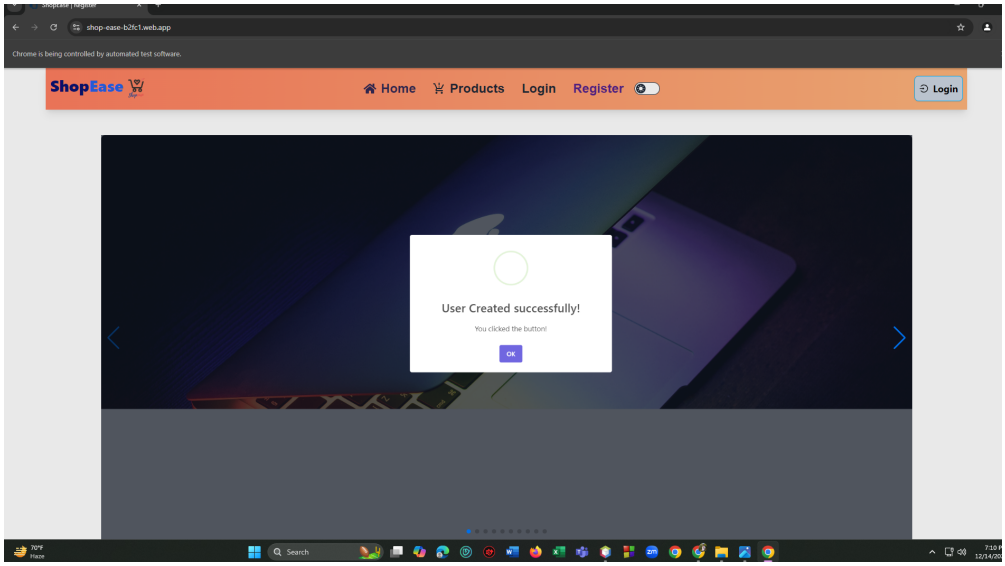
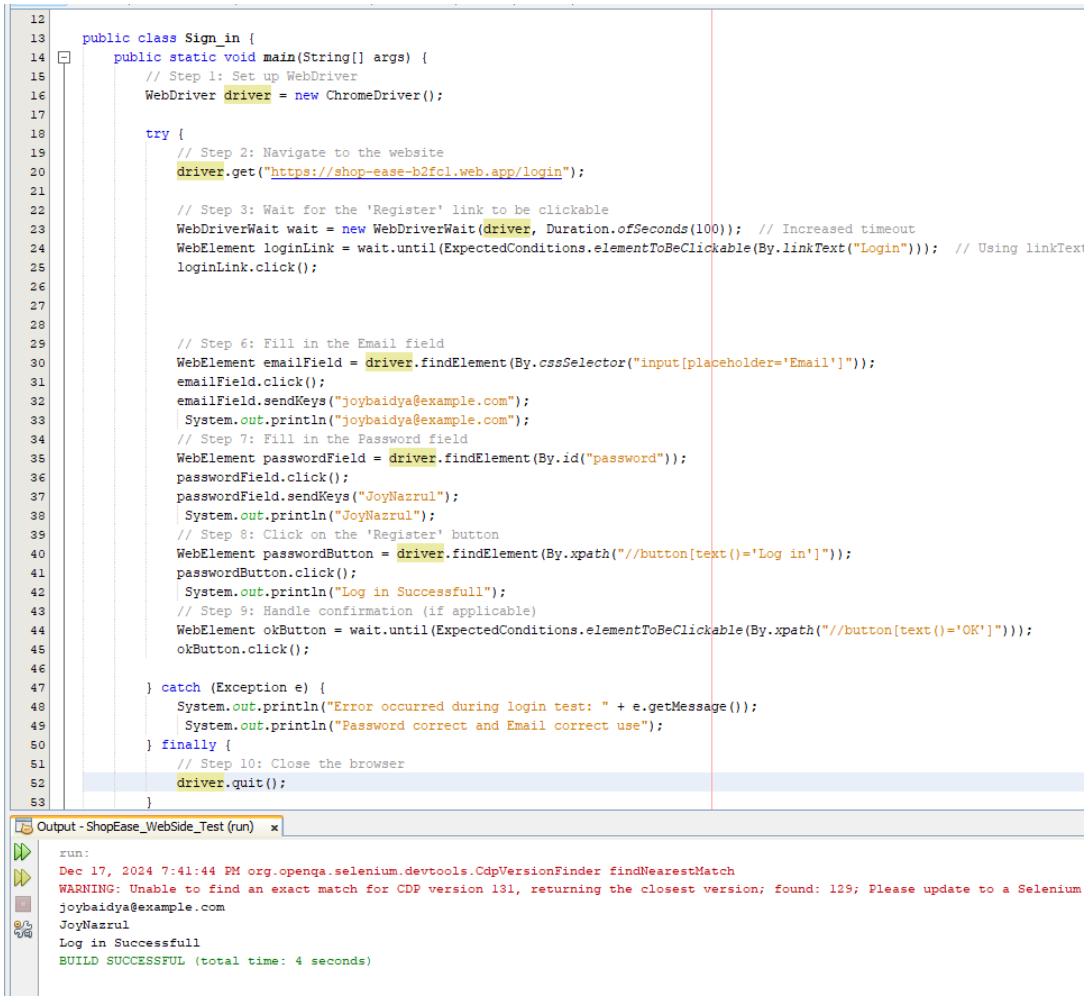


Figure 2.34: valid SignUp.....Test (error occur: ”Infinity password takes.)

Any validation or security weaknesses.

- **Expected result:**The system should reject the input, displaying a validation error like: ”Password range should be min:8 and max:16”
- **Actucal result:** unexpected password rang from user
- **Fail:** Expected result **NOT Equal** Actucal result

2.3.9.1 Valid Sign In



```
public class Sign_in {
    public static void main(String[] args) {
        // Step 1: Set up WebDriver
        WebDriver driver = new ChromeDriver();

        try {
            // Step 2: Navigate to the website
            driver.get("https://shop-ease-b2fc1.web.app/login");

            // Step 3: Wait for the 'Register' link to be clickable
            WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(100));  // Increased timeout
            WebElement loginLink = wait.until(ExpectedConditions.elementToBeClickable(By.linkText("Login")));  // Using linkText
            loginLink.click();



            // Step 6: Fill in the Email field
            WebElement emailField = driver.findElement(By.cssSelector("input[placeholder='Email']"));
            emailField.click();
            emailField.sendKeys("joybaidya@example.com");
             System.out.println("joybaidya@example.com");
            // Step 7: Fill in the Password field
            WebElement passwordField = driver.findElement(By.id("password"));
            passwordField.click();
            passwordField.sendKeys("JoyNazrul");
             System.out.println("JoyNazrul");
            // Step 8: Click on the 'Register' button
            WebElement passwordButton = driver.findElement(By.xpath("//button[text()='Log in']"));
            passwordButton.click();
             System.out.println("Log in Successfull");
            // Step 9: Handle confirmation (if applicable)
            WebElement okButton = wait.until(ExpectedConditions.elementToBeClickable(By.xpath("//button[text()='OK']")));
            okButton.click();

        } catch (Exception e) {
            System.out.println("Error occurred during login test: " + e.getMessage());
             System.out.println("Password correct and Email correct use");
        } finally {
            // Step 10: Close the browser
            driver.quit();
        }
```

```
Output - ShopEase_WebSide_Test (run)
run:
Dec 17, 2024 7:41:44 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 131, returning the closest version; found: 129; Please update to a Selenium
joybaidya@example.com
JoyNazrul
Log in Successfull
BUILD SUCCESSFUL (total time: 4 seconds)
```

Figure 2.35: Valid Login Popup and Authentication

Verify that the login popup appears and the user can log in using valid credentials.

- **Expected result:**The login popup is displayed successfully, and the user logs in with valid credentials and is redirected to the dashboard/home page.
- **Actucal result:**The login popup is displayed successfully
- **Pass:** Expected result **Equal** Actucal result

Table 2.7: Invalid Sign-in Scenarios

Case No.	Condition	Result
1	Password is correct but Email is in-correct.	Invalid Sign-in
2	Password is incorrect but Email is correct.	Invalid Sign-in
3	Password is incorrect and Email is incorrect.	Invalid Sign-in
4	Password is empty and Email is empty.	Invalid Sign-in

2.3.9.2 Invalid Sign In

```
        try {
            // Step 2: Navigate to the website
            driver.get("https://shop-ease-b2fc1.web.app/login");

            // Step 3: Wait for the 'Register' link to be clickable
            WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));  // Increased timeout
            WebElement loginLink = wait.until(ExpectedConditions.elementToBeClickable(By.linkText("Login")));
            loginLink.click();
            // Step 6: Fill in the Email field
            WebElement emailField = driver.findElement(By.cssSelector("input[placeholder='Email']"));
            emailField.click();
            emailField.sendKeys("bangladesh12@gmail.com");
            System.out.println("incurrect email= bangladesh12@gmail.com");
            // Step 7: Fill in the Password field
            WebElement passwordField = driver.findElement(By.id("password"));
            passwordField.click();
            passwordField.sendKeys("JoyNazrul");
             System.out.println("correct password");
            // Step 8: Click on the 'Register' button
            WebElement passwordButton = driver.findElement(By.xpath("//button[text()='Log in']"));
            passwordButton.click();

            // Step 9: Handle confirmation (if applicable)
            WebElement okButton = wait.until(ExpectedConditions.elementToBeClickable(By.xpath("//button[text(
            okButton.click();

        } catch (Exception e) {
            System.out.println("Error occurred during login test: " + e.getMessage());
            System.out.println("Password Correct and Email Incorrect use");
```

```
Output - ShopEase_WebSide_Test (run)
Dec 17, 2024 8:00:20 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 131, returning the closest version; found: 129; Please update to a Selenium version that
incurrect email= bangladesh12@gmail.com
correct password
Error occurred during login test: Expected condition failed: waiting for element to be clickable: By.xpath: //button[text()='OK'] (tried for 10
Build info: version: '4.25.0', revision: '030fcf7918'
System info: os.name: 'Windows 11', os.arch: 'amd64', os.version: '10.0', java.version: '17.0.2'
Driver info: org.openqa.selenium.chrome.ChromeDriver
Capabilities {acceptInsecureCerts: false, browserName: chrome, browserVersion: 131.0.6778.140, chrome: {chromedriverVersion: 131.0.6778.108 (3b0
Session ID: c6ffb32db835d86aae9fc3b784d22fcc
Password Correct and Email Incorrect use
BUILD SUCCESSFUL (total time: 13 seconds)
```

Figure 2.36: Password is correct but Email is In-correct

Verify that the login popup appears, and the user can log in using Invalid credentials.

- **Expected result:**The login popup can not displayed successfully but , and the user logs in with invalid credentials and is redirected to the dashboard/home page.
- **Actucal result:**The login popup can not displayed successfully
- **Pass:** Expected result **Equal** Actucal result

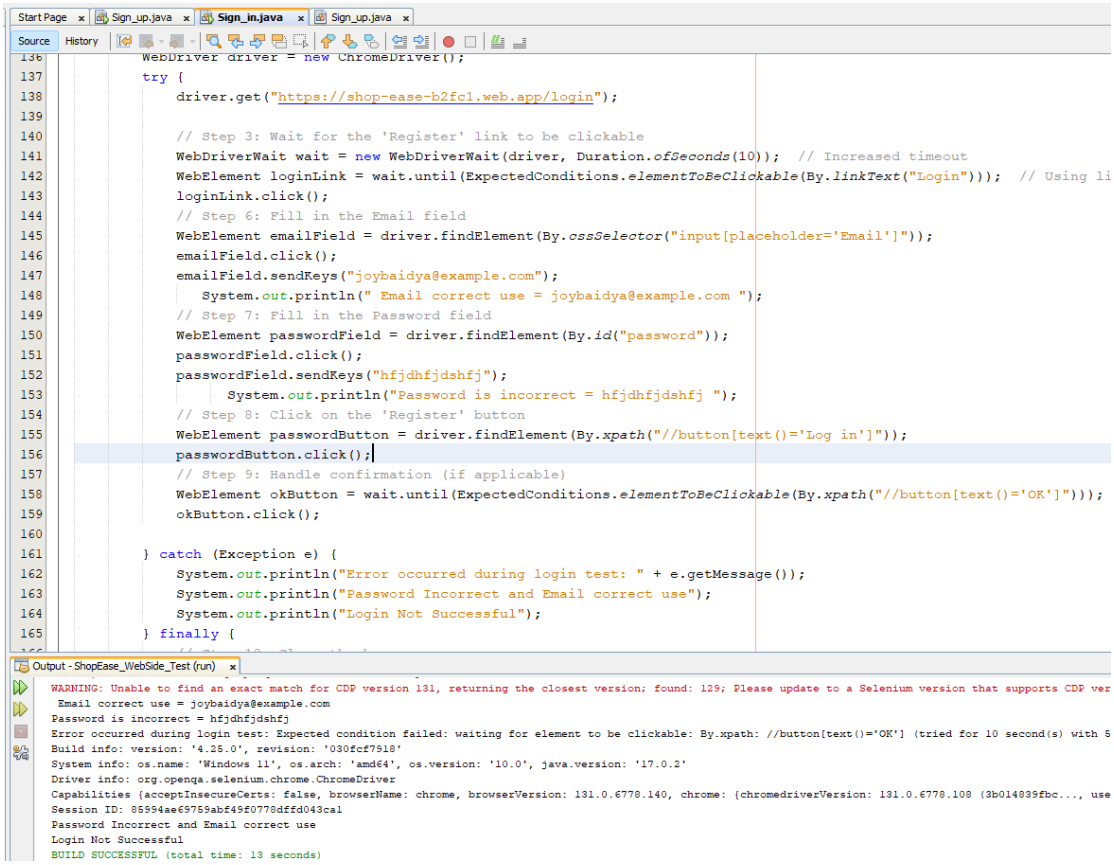


Figure 2.37: Password is incorrect but Email is correct.

Verify that the login popup appears, and the user can log in using Invalid credentials.

- **Expected result:**The login popup can not displayed successfully but , and the user logs in with invalid credentials and is redirected to the dashboard/home page.
- **Actucal result:**The login popup can not displayed successfully.

- **Pass:** Expected result **Equal** Actucal result

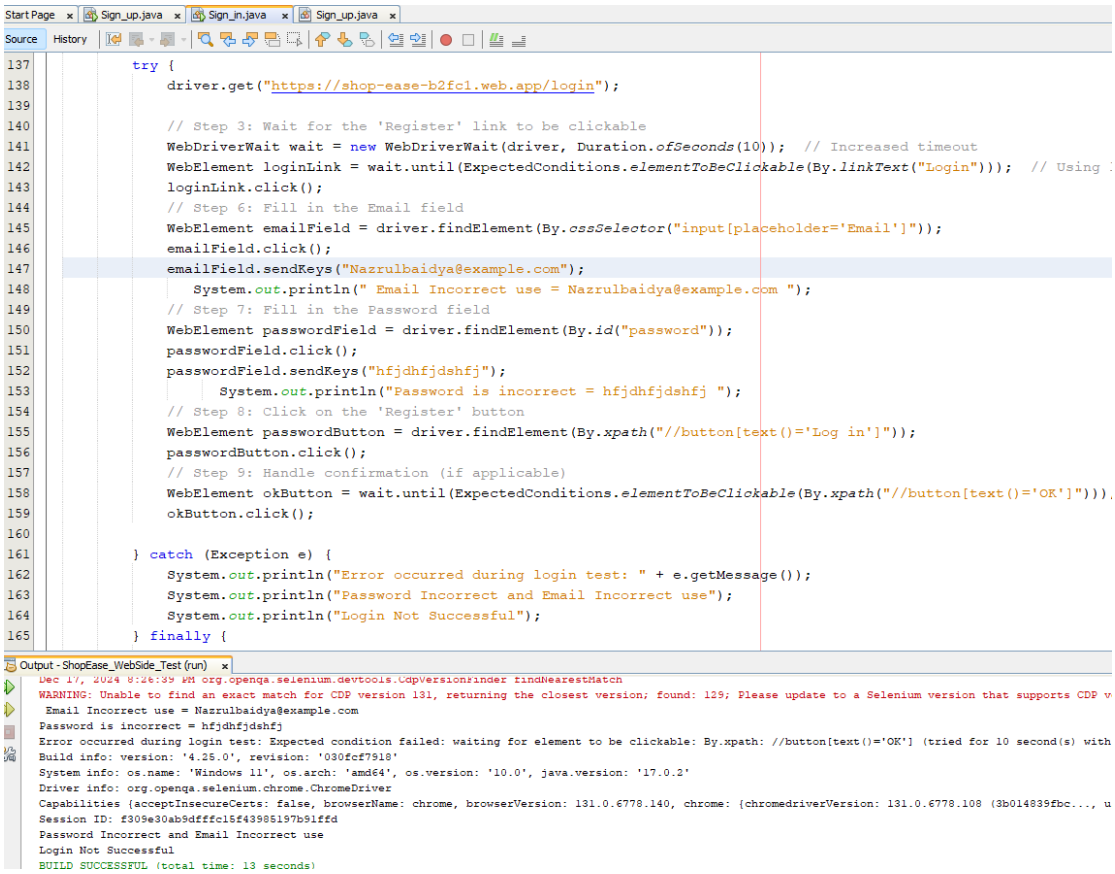


Figure 2.38: Password is incorrect and Email is incorrect.

Verify that the login popup appears, and the user can log in using Invalid credentials.

- **Expected result:**The login popup can not displayed successfully but , and the user logs in with invalid credentials and is redirected to the dashboard/home page.
- **Actucal result:**The login popup can not displayed successfully
- **Pass:** Expected result **Equal** Actucal result

Figure 2.39: Password is empty and Email is empty.

Verify that the login popup appears, and the user can log in using Invalid credentials.

- **Expected result:**The login popup can not displayed successfully but , and the user logs in with invalid credentials and is redirected to the dashboard/home page.
- **Actucal result:**The login popup can not displayed successfully
- **Pass:** Expected result **Equal** Actucal result

2.3.9.3 ButtonSeeAllProduct Tc9

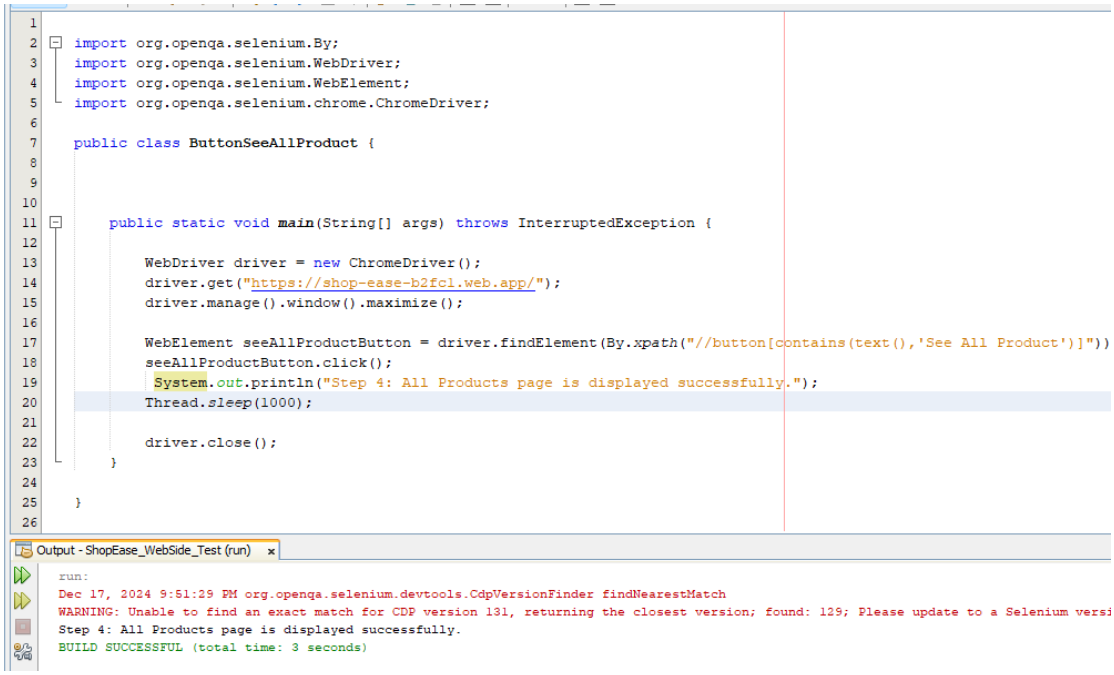


Figure 2.40: SeeAllProduct

Validate redirection to the ”All Products” page

- **Expected result:**The ”All Products” page is displayed with all product listings.
- **Actucal result:**All product displayed successfully.
- **Pass:** Expected result **Equal** Actucal result

2.3.9.4 Navigate to Products tc10

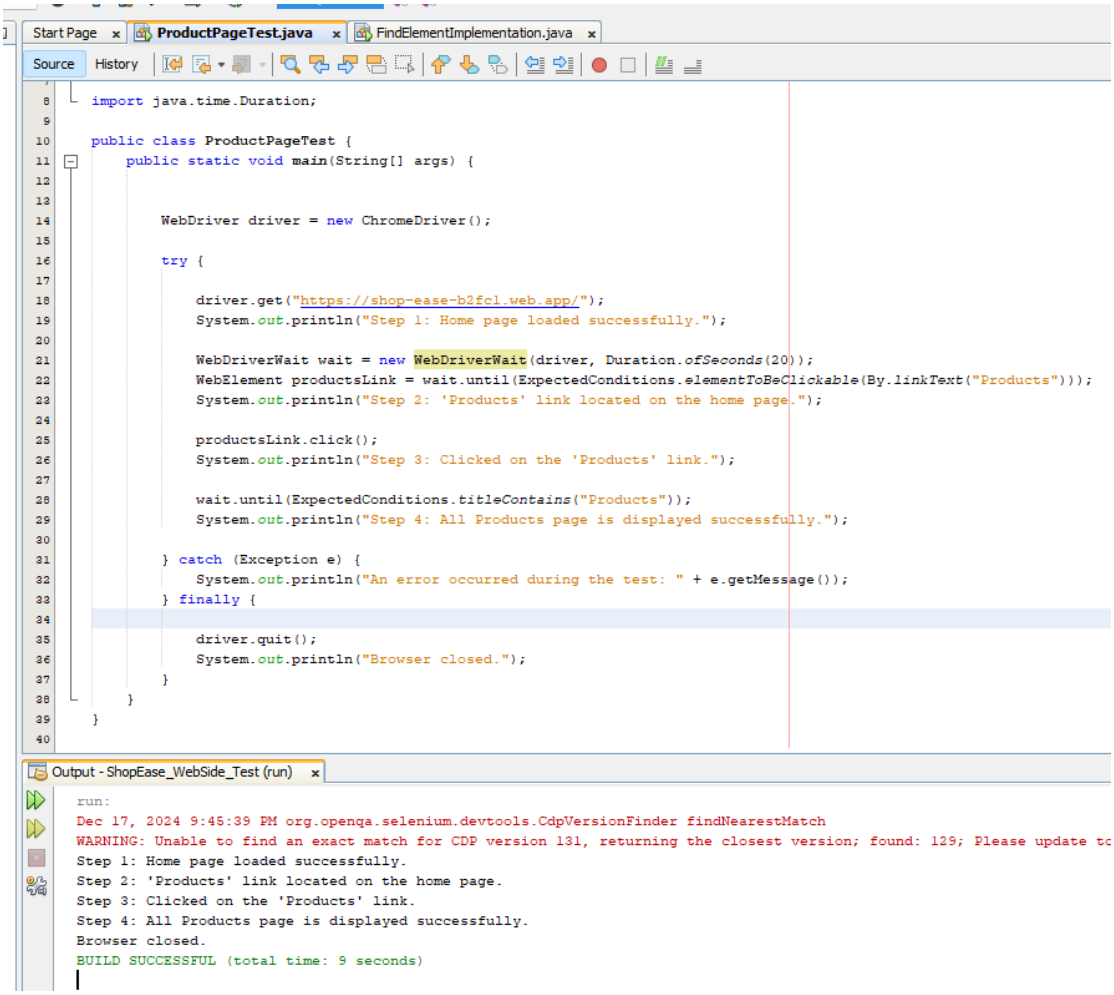


Figure 2.41: Navigate to Products

Verify that clicking on the ”Products” route redirects to the All Products page.

- **Expected result:**The All Products page is displayed correctly, with all product items visible.
- **Actucal result:**All pare redirection including button, search field and other displaying successfully.
- **Pass:** Expected result **Equal** Actucal result

2.3.9.5 Verify Search Functionality Tc11

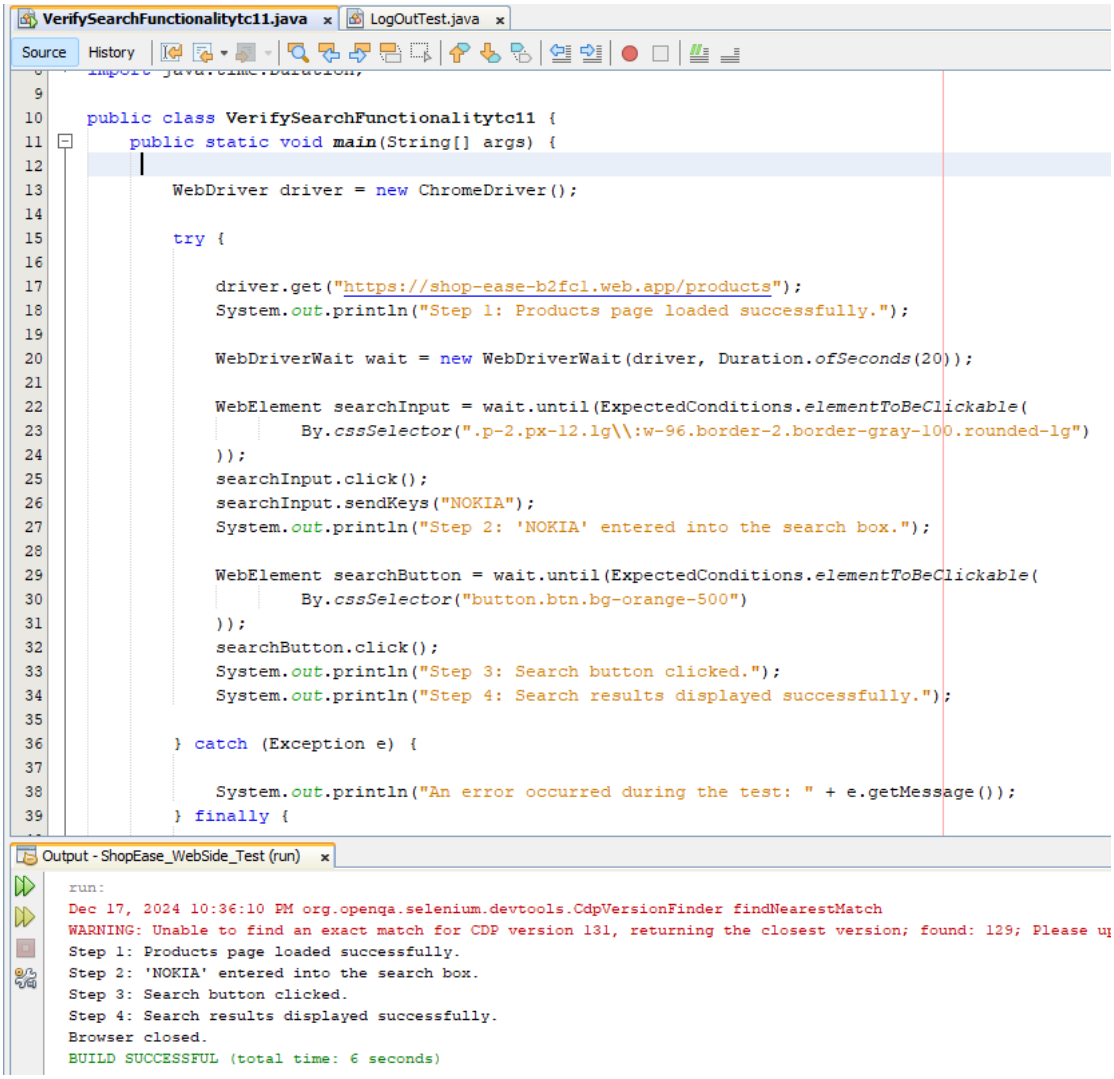


Figure 2.42: Verify Search Functionality

Verify the search functionality by entering the term ”camera.” The test ensures that if products related to ”camera” are available, they are displayed; otherwise, no results are shown.

- **Expected result:**If no matching products are available, the screen should indicate no results (e.g., display a message like ”No products found” or remain blank without displaying irrelevant products).

- **Actucal result:** Everything is ok, but no product found message is not showing. It would be better if the message is shown.
- **Fail:** Expected result **Not Equal** Actucal result
- **Comment:** The product not found message should have been shown.

2.3.9.6 Product Category Filter Functionality Tc12

```
12  public class ProductCategoryFilterFunctionality {
13      public static void main(String[] args) throws InterruptedException {
14          // Step 1: Initialize WebDriver
15          WebDriver driver = new ChromeDriver();
16
17          try {
18              // Step 2: Open the target web page
19              driver.get("https://shop-ease-b2fc1.web.app/products");
20              System.out.println("Step 1: Products page loaded successfully.");
21
22              // Step 3: Define an explicit wait
23              WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(20));
24
25              // Step 6: Locate the 'Category' dropdown and select 'Laptop'
26              WebElement categoryDropdown = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector
27          ("select.p-2.px-4.font-bold.border-2.bg-purple-400.border-gray-100.rounded-lg")));
28             Select categorySelect = new Select(categoryDropdown);
29             categorySelect.selectByValue("Laptop");
30             System.out.println("Step 5: 'Laptop' selected from Category dropdown.");
31
32              // Step 7: Locate the 'Brand' dropdown and select 'Samsung'
33                 WebElement brandDropdown = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector
34          ("select.p-2.px-4.font-bold.border-2.bg-purple-400.border-gray-100.rounded-lg.mt-4.lg\\:mt-0")));
35                Select brandSelect = new Select(brandDropdown);
36                brandSelect.selectByValue("Samsung");
37                System.out.println("Step 6: 'Samsung' selected from Brand dropdown.");
38
39          } catch (Exception e) {
40              System.out.println("An error occurred during the test: " + e.getMessage());
41              e.printStackTrace(); // Print detailed exception stack trace for debugging
42          } finally {
43              Thread.sleep(7000);
44              // Step 7: Close the browser
45              driver.quit();
46              System.out.println("Browser closed.");
47          }
48      }
49  }
```

```
Output - ShopEase_WebSide_Test (run)
run:
Dec 18, 2024 1:35:17 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 131, returning the closest version; found: 129; Please update to a Selenium vers
Step 1: Products page loaded successfully.
Step 5: 'Laptop' selected from Category dropdown.
Step 6: 'Samsung' selected from Brand dropdown.
Browser closed.
BUILD SUCCESSFUL (total time: 11 seconds)
```

Figure 2.43: Product Category Filter Functionality

Verify that selecting a category

- **Expected result:**The system should display only the products corresponding to the selected category: - 'Mobile' should show all mobile products. - 'Watch'

should show all watch products.

- **Actucal result:** Match category has been shown successfully.
- **Pass:** Expected result **Equal** Actucal result

2.3.9.7 Validate Price Range Filter

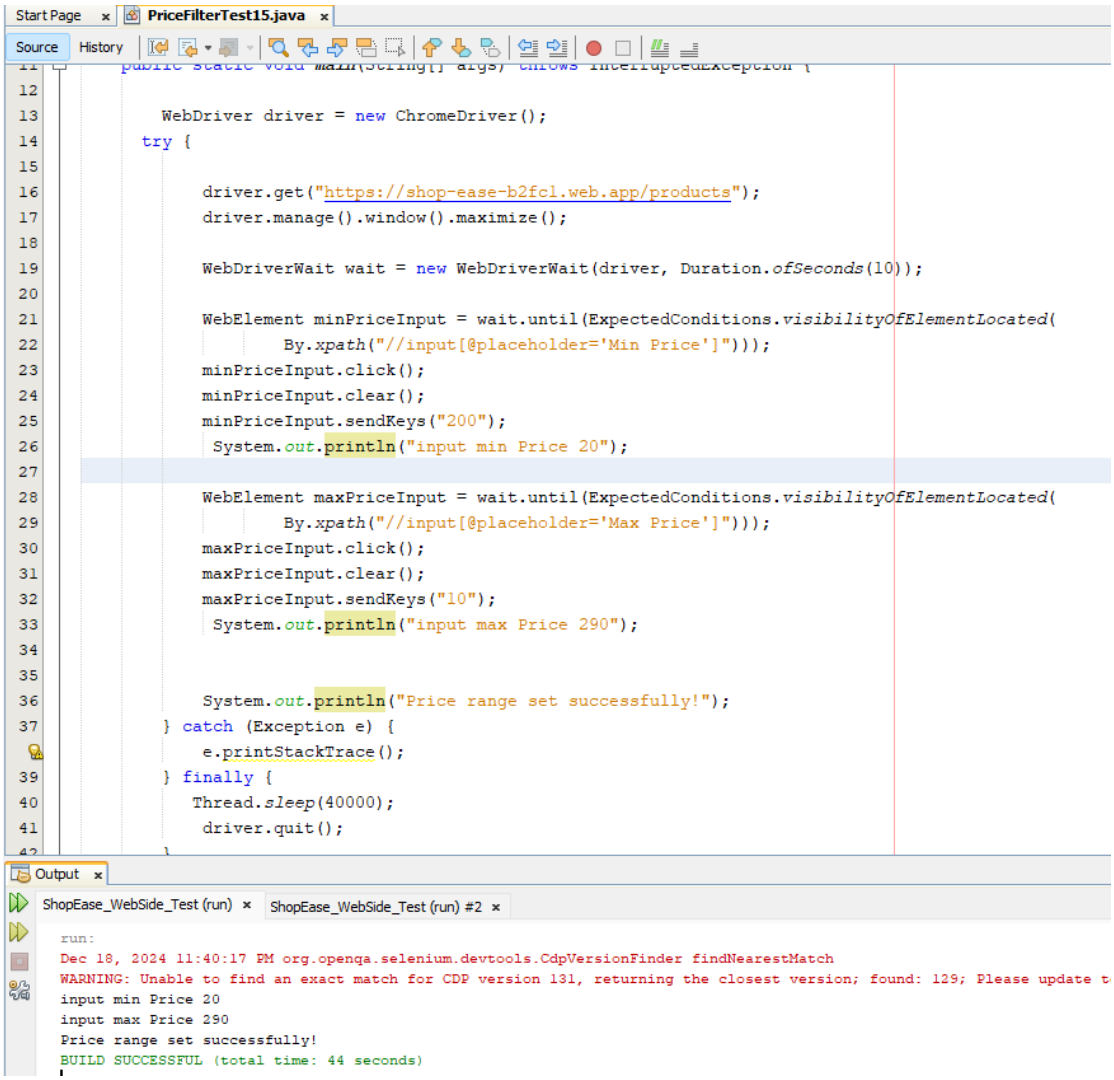


Figure 2.44: Validate Price Range Filter

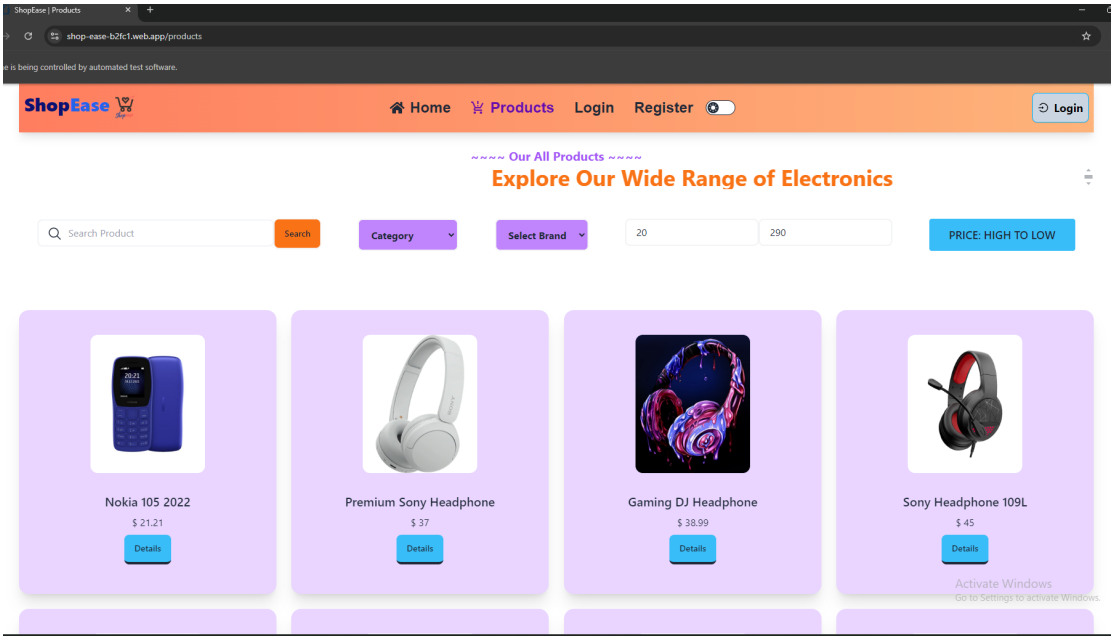


Figure 2.45: Validate Price Range Filter Rusult

Verify that products displayed are within the selected price range (min and max values).

- **Expected result:**All displayed products have prices between 100 and 300 (inclusive).
- **Actucal result:**All displayed products have prices between 100 and 300 (inclusive). so the functionality works perfectly.
- **Pass:** Expected result **Equal** Actucal result.

  BUT Verify that invalid inputs in the minimum and maximum price fields are handled correctly by the system then
- The meaningful message should have been displayed, but it's not included.

2.3.9.8 Validate High Low SortingTc17,18

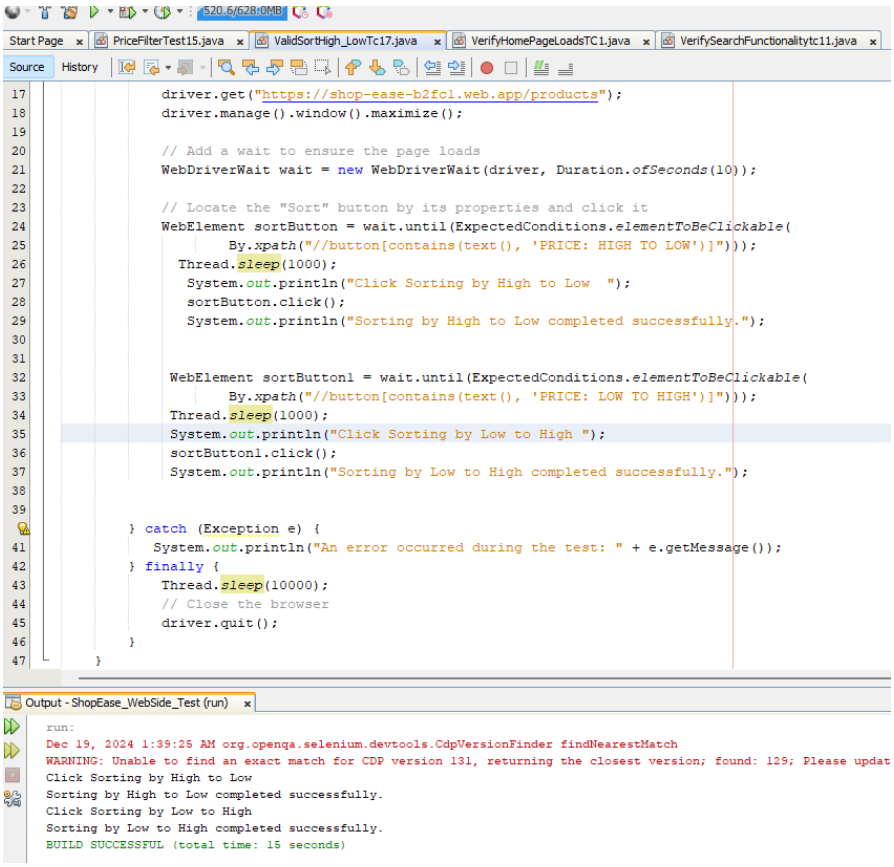


Figure 2.46: Validate High-to-Low And Low-to- High Sorting

Verify that products are sorted by price in descending order when ”High to Low” is selected.Verify that products are sorted by price in ascending order when ”Low to High” is selected.

- **Expected result:** Fast cleck Products are displayed from highest to lowest price then Products are displayed from lowest to highest price.
- **Actucal result:** products are sorted and displayed successfully high to low or products are sorted and displayed successfully low to high
- **Pass:** Expected result **Equal** Actucal result.

2.3.10 Security Testing

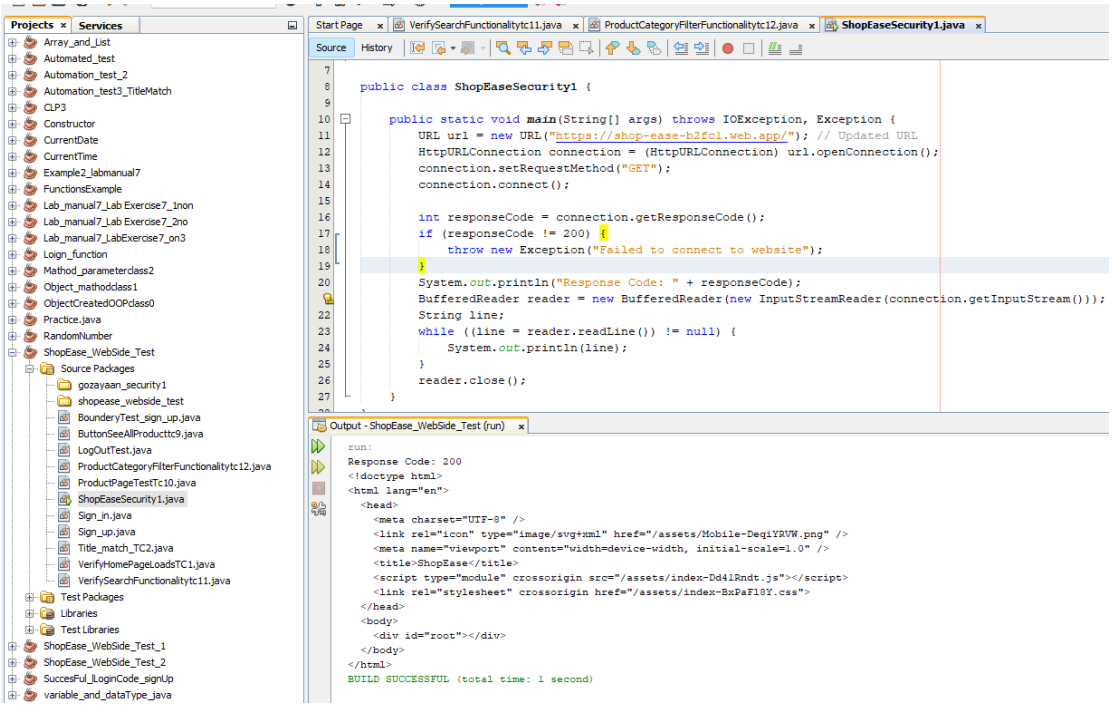


Figure 2.47: Security Testing

Website Security Test Result

2.8Table: The test was performed on the URL `https://shop-ease-b2fc1.web.app/` to validate the connection and fetch the response. Below are the key results:

Table 2.8: Security Test Results for ShopEase Website

Test Metric	**Observation**
Connection Status	Response Code: 200 (Successful connection to the website).
Response Content	The HTML structure of the webpage was retrieved successfully, including meta tags and links.
Execution Time	The test completed successfully in 1 second.

Chapter 3

Conclusion

3.1 Conclusion

In this project, I gained practical experience in various software testing techniques, including manual testing, automated testing, performance testing, and API testing. I utilized tools like Selenium, JMeter, Postman, and Jira to ensure the quality and reliability of the *ShopEase* e-commerce platform.

Through manual testing, I identified and documented bugs effectively, while automated testing with Selenium streamlined repetitive tasks. Performance testing with JMeter helped optimize the system's stability and scalability, and API testing using Postman ensured smooth data exchanges between components.

By following Agile methodologies such as Scrum and Kanban, I improved collaboration and ensured timely delivery of results. Overall, this project enhanced my skills in testing tools and methodologies, preparing me for real-world software quality assurance challenges.

3.2 Limitations

- **Limited Test Coverage**: Automated tests focused on repetitive tasks, leaving complex scenarios for manual testing.

- **Time Constraints**: Insufficient time restricted thorough exploratory testing and performance benchmarking.
- **Environment Gaps**: The testing setup lacked real-world user conditions, affecting performance accuracy.

3.3 Future Scope

- **Extended Automation**: Increase automated test coverage for diverse scenarios.
- **Enhanced API Testing**: Focus on advanced cases, including error handling and stress testing.
- **Performance Optimization**: Use cloud-based tools for scalability and real-world simulations.
- **Security Testing**: Integrate tools to identify vulnerabilities and improve system security.

References

- https://www.guru99.com/manual-testing.html
- https://shop-ease-b2fc1.web.app/
- https://shop-ease-b2fc1.web.app/login
- https://artoftesting.com/types-of-performance-testing
- https://www.guru99.com/selenium-tutorial.html
- https://www.atlassian.com/software/jira
- https://www.postman.com/api-evangelist/plentymarkets/folder/ttpufno/login

---

THE END