|  | **HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY**  **INSTITUTE OF INTERNATIONAL EDUCATION** |
| --- | --- |

**FINAL ASSIGNMENT REPORT**

Major: **INFORMATION TECHNOLOGY**

Course code: **CMP179**

Course name: **Software Testing and Quality Assurance**

Lecturer’s name: MR. TRINH CONG NHUT

Student’s name:

1. HUYNH THANH DANH ID: 2180602030 Class: 21DTHQA2

Ho Chi Minh City

2025

**Table of contents**

[Prologue 4](#_gg7arqlu8tdm)

[**Chapter 1. Test Complete 5**](#_msnfb0tb0xie)

[Overview 5](#_yuvm6fdjihsw)

[Why use it? 5](#_baipu8mennt9)

[Create Project in Test Complete 6](#_od9jthhhb2w4)

[TestComplete User Interface 6](#_uq579i2vtt4s)

[Follow the steps to record the test: 7](#_okfjauhanmcy)

[Analyzing the test results 8](#_qcmvwztvry10)

[Creating a test on a desktop application 8](#_jveuli2ebma3)

[Analyze the recorded test 9](#_uzs0th9wbwba)

[**Chapter 2. TEST CASES 10**](#_bb9psvye75)

[1. Selenium IDE 10](#_1mjr13xp0pl5)

[What is Selenium IDE? 10](#_7zk366nhqk0)

[Installation and Usage 10](#_bo1c6cjnickb)

[Example and Limitations 10](#_z99dau8vct0x)

[Detailed Survey Note on Selenium IDE 10](#_3p8n01ahust0)

[Introduction to Selenium IDE 10](#_bmgzx5j1k40d)

[Features and Benefits 11](#_r6ijchsivnc5)

[Installation Process 11](#_l7x37weo08fb)

[Getting Started with Usage 11](#_wz2ffvwd3vxc)

[Example Test Case: Logging into a Web Application 12](#_e0etbk7stkdh)

[Advanced Usage and Customization 12](#_9tfm6q3i65t4)

[Limitations and When to Use 13](#_j948xp1rmclh)

[Comparative Context 13](#_rns1afwvf088)

[2.Application Under Test Introduction 13](#_u25bo4nev8gz)

[Home page: 14](#_3aw7xerll82w)

[Introduction of main function of website: 14](#_hwas2o7emdwc)

[3. Summary of test cases 16](#_efycf9kebf2h)

[4. Details of test cases 19](#_qow0brnx7eba)

[**Chapter 3. CONCLUSION 25**](#_u856kakxrdv7)

[**References 26**](#_notyhp4wq5cq)

[**Prologue**](#_k9tj7cbafp5f)

In the era of strong digitalization, e-commerce is growing, leading to the need to ensure software quality to bring the best online shopping experience to users. Functional testing of an e-commerce website like Grimm DC plays an important role in detecting errors, optimizing the shopping process and improving system performance.

This document presents detailed test cases, including testing the login, registration, search, filter, shopping cart, payment process and many other features of the Grimm DC website. Through these test scenarios, we aim to ensure the system operates smoothly, meets user requirements and improves the online shopping experience.

**Chapter 1. TEST COMPLETE**

Overview

Test Complete, developed by SmartBear Software, provides support for technologies such as: Net, Delphi, C++Builder, Java, Visual Basic, HTML5, Flash, Flex, Silverlight Desktop, Web and Mobile systems. TestComplete helps testers develop their test cases in various scripting languages ​​such as JavaScript, Python, VBScript, Delphi Script, JavaScript. It is available with two licenses and a free trial version valid for 30 days.

Why use it?

TestComplete provides a wide range of test automation capabilities.  
**Some of them are listed below:**

Keyword Testing: Uses the built-in Keyword editor so that testers can develop frameworks that control Keywords very easily.

Scripted Testing: Testers can write test scripts from scratch or modify the scripts recorded in the built-in editor

Test Record and Playback: Provides the basic mechanism of recording and playing back the initiated tests. The recorded test cases can be modified as needed

Integration to Bug Tracking Software: Integrates with various bug tracking software like Jira, Bugzilla, etc. It can be used to modify or create entries in bug tracking software using issue tracking templates

Data Driven Testing: Easily extract data from CSV files, database tables, Excel sheets, etc.

Test Visualizer: Taking screenshots during the test execution allows us to distinguish between the desired and actual screens.

**Minimum system requirements**

Operating system: Microsoft Windows XP Professional 32/64 bit.

Chip: Intel Core 2 Duo 2 GHz or higher

Ram: 2 GB RAM on other operating systems.

Hard disk: 1 GB free space for installation

Resolution: 1024 × 768 or higher.

Mouse or other pointing device

Install TestComplete

Download => TestComplete can be downloaded from the SmartBear homepage: https://smartbear.com/product/testcomplete/free-trial/

After downloading, follow these steps to install TestComplete

1) Double-click the downloaded TestComplete installation package. The software installation will start and the license agreement will be displayed.

2) Specify the path of the folder where you want to install the software.

3) Now, the welcome dialog is displayed asking for license activation, we can start by clicking on the 30-day trial license.

4) Then restart the computer, we have completed the TestComplete installation process.

## **Create Project in Test Complete**

Launch the application and you will see the start page.

Follow these steps to create a new Project.

1) Go to the file menu.

2) Click on New Menu from the menu.

3) Click on New Project.

4) Or alternatively, you can use the shortcut key (shift + ctrl + N) to create a New Project.

5) A window will appear, and you name the Project.

6) Click on Finish

7) So we have created the first Project in TestComplete.

## **TestComplete User Interface**

The TestComplete user interface is well organized and divided into different sections.

• The Project Explorer panel on the left side of the window, which displays the project suites, projects, and content

• The Workspace Panel is where you can modify and test

• The Test Visualizer panel (at the bottom) displays the screenshots saved during the execution of test cases.

Creating a Test Project on the Web

Let's start by creating the first test on a web project in TestComplete.

1) Select File | New | New Project as shown in the screenshot.

2) Enter the project name; you can also change the Location if needed.

3) Click the Next button.

Note: Since TestComplete supports different platforms like web, desktop, mobile, etc., we have to select the platform on which we are testing.

4) Initially, we are using the web testing application, so select "Functional testing of web pages" and click Next.

5) Once we have selected the platform in the previous step, the project creation wizard will point to the Test Visualizer page, where we can enable/disable the Test Visualizer functionality. Click Next

6) Now we need to specify the scripting language for the project. After selecting the project language, click Finish.

Now the Project for testing the web application is ready!

Recording the test for the web application

We will start with recording the test, where we will open in Google search engine and search for a query.

## **Follow the steps to record the test:**

1) Click Append to Test, as shown in the image below.

Note: TestComplete records the user action and usually the click, i.e. whenever the user clicks on any object, the id and reference are recorded.

2) A log is displayed as shown in the image below, which shows that the test recording has started. Now we are ready to execute.

3) Launch the browser, TestComplete recognizes the browser with its built-in special test command.

4) Navigate to this URL https://www.google.com

5) Type any query on the Google search box, for example test help software.

6) Click on the Stop button as shown in the image.

7) On clicking the Stop button, TestComplete will display the keyword editor where all the recorded keywords will be displayed.

8) To replay, the stored test cases just click on the Run Test button as shown in the image below.

## **Analyzing the test results**

Let's start analyzing the test results.

The running browser will launch the browser. It detects the launched browser with its built-in test functions and executes the test during the playback.

This command is used to wait for the page to load; Here we have opened the Google homepage, so that means the test execution is paused until the Google homepage has been fully loaded.

The following command is used to place the text in the Google search bar, we are using the test software as the keyword and hence the following text is displayed.

In the Test Visualizer, screenshots have been taken while executing the test, so that the tester can differentiate between the actual and expected screen output.

Note: Till now, we have only recorded a few basic steps. In fact, this is not a complete test. You will have to add/remove/customize steps to perform validation of the scripts that you need to execute.

## **Creating a test on a desktop application**

TestComplete supports both Web as well as Desktop based applications.

Let's start with creating a project on a Desktop based application.

Note: Close all open projects in TestComplete. Click File | Close.

1) Create new project-> File | New | New Project opens a new project wizard.

2) Specify a project name and location. Click Next.

*Note: Since we are running the test on a desktop application, select Windows application on the wizard. This will take us to the page where we can specify the platform of the project. Select Generic Windows application and click Next.*

Note: Since we are automating a desktop application, we need to specify the application to test in TestComplete

4) Click the Add button and specify the path of the project in the window that opens.

For demo purposes, we are only creating the test on notepad.exe.

5) Specify the path for the notepad.exe file on your computer For example: "C:\Users\Admin\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Accessories\Notepad.lnk".

6) Click OK. Then Next.

7) Select the required settings for the Test Visualizer. Click Next.

8) Select the scripting language. Click Finish.

Now we have created a project to record our test on a desktop application.

Recording a test for a desktop application

Once we have recorded our test on a Web project, recording our test for a Desktop application is simple.

1) Click Append to test.

2) A new notepad file will open.

3) You can select any text. For example "Software testing help."

4) Click the Stop button.

5) Close the notepad file.

6) To play back, simply click Run Test.

## **Analyze the recorded test**

Run Tested App is the command used to launch the application. Since we are testing on notepad.exe, the notepad name is displayed in the Operation Column. TestComplete will record the operation when the application is launched.

We have typed software testing help in the open notepad window, so the Edit command is used to set the text in the application.

**Chapter 2. TEST CASES**

1. **Selenium IDE**

**What is Selenium IDE?**

Selenium IDE (Integrated Development Environment) is a tool that helps automate testing for web applications. It records your actions on a website—like clicking buttons or filling forms—and plays them back to check if everything works as expected. It’s part of the Selenium suite, which is widely used for browser automation.

## **Installation and Usage**

You can install Selenium IDE from the Chrome Web Store, Firefox Add-ons store, or Microsoft Edge Add-ons store. Once installed, launch it from your browser’s menu bar, create a new test case, record your interactions, save it, and play it back to verify functionality. It’s great for beginners due to its ease of use.

## **Example and Limitations**

For example, you could record logging into a website, enter credentials, and check if it logs in successfully. However, for large or complex test projects, you might need Selenium WebDriver, as IDE is better for prototyping and simple tests.

## **Detailed Survey Note on Selenium IDE**

Selenium IDE (Integrated Development Environment) is an open-source tool designed to simplify the creation and execution of automated tests for web applications, particularly through its record-and-playback functionality. This survey note provides a comprehensive overview, covering its purpose, features, installation, usage, examples, and limitations, ensuring a thorough understanding for users seeking to leverage this tool for testing.

## **Introduction to Selenium IDE**

Selenium IDE is a browser extension that enables users to record their interactions with a web application and replay them to verify functionality. It is part of the broader Selenium suite, which includes tools like Selenium WebDriver and Selenium Grid for more advanced testing scenarios. The tool is particularly valuable for functional testing, ensuring that web applications behave as expected under various user interactions.

Research suggests that Selenium IDE is especially beneficial for those new to test automation, as it requires minimal programming knowledge. Its ease of use and accessibility make it a popular choice for creating quick, automated tests, with the evidence leaning toward its effectiveness in prototyping and basic testing needs.

**Features and Benefits**

Selenium IDE offers a range of features that enhance its utility for web testing:

* **Record and Playback:** Automatically generates test scripts based on user interactions, allowing for easy creation of test cases without extensive coding.
* **Multi-Browser Support:** Available for Google Chrome, Mozilla Firefox, and Microsoft Edge, ensuring cross-browser compatibility and testing.
* **Extensibility:** Can be extended with custom commands and integrated with third-party services through plugins, offering flexibility for advanced users.
* **Debugging Tools:** Includes features like setting breakpoints and pausing on exceptions, which aid in debugging test cases efficiently.
* **Control Flow Structures:** Supports commands like if, while, and times, enabling the creation of more complex test logic.
* **Command-Line Runner:** Allows running tests in parallel across different browser and operating system combinations, enhancing scalability.

The benefits include time savings through automation of repetitive tasks, consistency in test execution, and early detection of issues, which can significantly improve the quality of web applications.

**Installation Process**

Installing Selenium IDE is straightforward and browser-dependent:

* For Google Chrome, download it from the [Chrome Web Store](https://chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd).
* For Mozilla Firefox, find it in the [Firefox Add-ons store](https://addons.mozilla.org/en-US/firefox/addon/selenium-ide/).
* For Microsoft Edge, access it via the [Microsoft Edge Add-ons store](https://microsoftedge.microsoft.com/addons/detail/selenium-ide/egnjhciaieciobmggjffcdlbbipjdogg).

Once installed, launch the extension from the browser’s menu bar to begin creating test cases. The process is user-friendly, requiring no additional setup beyond the initial installation.

**Getting Started with Usage**

To start using Selenium IDE, follow these steps:

**Launch the IDE:** Access it from the browser’s menu bar after installation.

**Create a New Test Case:** Select “New Test Case” to begin recording.

**Record Interactions:** Navigate to the web application and perform the actions you want to automate, such as clicking buttons or entering text.

**Save the Test Case:** Save the recorded test with a meaningful name for future use.

**Play Back the Test Case:** Run the test to ensure it performs the recorded actions correctly, verifying the application’s functionality.

This process is intuitive, making it accessible for users with limited technical background, and it provides instant feedback, which is crucial for learning and debugging.

## **Example Test Case: Logging into a Web Application**

To illustrate, consider creating a test case for logging into a web application, such as Gmail:

**Objective:** Verify that the login functionality works as expected.

**Steps:**

Open the browser and navigate to the login page, e.g., [Gmail Login](https://accounts.google.com/).

Record entering the username in the username field.

Record entering the password in the password field.

Record clicking the “Sign In” button.

**Verification:** Ensure the test successfully logs in, checking for any errors or failures during playback.

This example demonstrates how Selenium IDE can automate repetitive tasks, saving time and ensuring consistency in testing.

## **Advanced Usage and Customization**

For users seeking more from Selenium IDE, advanced features include:

**Editing Test Cases:** Modify recorded steps to make them more robust, such as adding waits or changing element locators.

**Using Control Flow Commands:** Implement conditional logic, such as using “if” to handle different scenarios or “while” for loops, enhancing test complexity.

**Exporting Test Cases:** Export tests to other formats like Java, Python, or C#, allowing integration with continuous integration/continuous deployment (CI/CD) systems for larger projects. For instance, you can export to use with Selenium WebDriver for more advanced automation.

These capabilities make Selenium IDE versatile, catering to both beginners and those needing more sophisticated testing solutions.

## **Limitations and When to Use**

While Selenium IDE is powerful for certain use cases, it has limitations:

**Best Suited For:**

Quick, simple tests, ideal for prototyping and initial testing phases.

Basic functionality testing, such as form submissions or button clicks.

Users with minimal programming experience, due to its record-and-playback nature.

**Consider Alternatives For:**

Large test suites requiring extensive reporting and management, where Selenium WebDriver with frameworks like TestNG or JUnit might be more appropriate.

Complex test automation needs, such as handling dynamic content or advanced conditional checks, where IDE’s capabilities may be insufficient.

Integration with CI/CD pipelines for continuous testing, where exported scripts can be better managed with other tools.

The evidence leans toward using Selenium IDE for early-stage testing and prototyping, with a shift to more robust tools for mature projects.

## **Comparative Context**

Selenium IDE is part of the Selenium ecosystem, which includes Selenium WebDriver for programmatic test creation and Selenium Grid for scaling tests across multiple machines. While IDE is excellent for quick tests, WebDriver offers more flexibility for coding complex scenarios, and Grid enhances scalability for parallel execution. This positioning highlights IDE’s role as an entry point into the Selenium suite, with pathways for escalation as testing needs grow.

1. **Application Under Test Introduction**

Website: <https://grimmdc.com/>

Grimmdc.com is an e-commerce website designed to provide users with a comprehensive online shopping experience. It offers a variety of features to browse, select, and purchase products efficiently. Below is an introduction to its main functions, outlining how users can interact with the platform.

**Home page:**



**Introduction of main function of website:**

**1. Search**

Allows users to search for products using keywords.

Enables quick and easy access to specific items across the website.

**2. User Authentication**

**Log in**: Existing users can access their accounts to manage orders and preferences.

**Log out**: Ends the user session securely.

**Create account**: New users can register to save their details, track orders, and enjoy a personalized shopping experience.

**3. Product Filtering and Sorting**

**Filter**: Users can refine product listings based on criteria such as price, brand, size, color, and more, enhancing the shopping experience.

**Arrange**: Provides sorting options for product listings, including:

Ascending (e.g., price low to high).

Descending (e.g., price high to low).

Newest arrivals, popularity, and other relevant options.

**4. Product Detail**

Displays detailed information about individual products, including:

**Quantity**: Options to select how many items to purchase or view available stock.

**Add to cart**: Adds the product to the shopping cart for later checkout.

**Buy Now**: Allows immediate purchase without adding to the cart.

**Size, Color**: Options for products available in multiple variations.

**5. Shopping Cart**

The shopping cart functionality is split into two related sections:

**Your Cart**:

**Displays a summary of added products, including**:

**Product**: Name or description of the item.

**Unit price**: Price per item.

**Quantity**: Number of each item selected.

**Total**: Calculated cost for each product (unit price × quantity).

Features options such as:

**Update Cart**: Adjust quantities or remove items.

**Proceed to Checkout**: Move to the payment process.

**Note**: Add special instructions if needed.

**Delete Product**: Remove items from the cart.

**Cart** (Detailed View):

Expands on "Your Cart" with additional details:

**Product, Unit price, Quantity, Total**: Same as above.

**Update Cart, Proceed to Checkout, Note, Delete Product**: Same functionality as above.

**Transport**: Options for shipping methods.

**Note for shop owners**: A field to include messages or special requests for the seller.

**6. Payment Information**

The checkout page where users finalize their purchase by providing:

**Full name, Email, Phone**: Contact details.

**Address**: Shipping details, including Province, District, and Ward.

**Payment method**: Options such as credit card, PayPal, or other payment systems.

**Products in cart**: List of items being purchased.

**Total**: Final amount to pay.

**Coupon**: Field to apply discount codes.

**Provisional**: Temporary total before applying shipping or discounts.

**Shipping fee**: Cost of delivery based on selected transport options.

1. **Summary of test cases**

| **TC ID** | **TC Description** | **Author** | **Note** |
| --- | --- | --- | --- |
| TC-01 | Test login functionality, including validation for empty fields and invalid login credentials. | Danh | 0.1 |
| TC-02 | Test registration features, including authentication | Danh | 0.1 |
| TC-03 | Test the add product to cart feature on the website. | Danh | 0.15 |
| TC-04 | Check delete products in cart | Danh | 0.15 |
| TC-05 | Check the updated quantity of products in the shopping cart | Danh | 0.2 |
| TC-06 | Check exact search | Danh | 0.2 |
| TC-07 | Check the accuracy of the total value in the shopping cart | Danh | 0.15 |
| TC-8 | Incremental function testing | Danh | 0.1 |
| TC-9 | Check the search function to see how many products have "Sơ mi". | Danh | 0.1 |
| TC-10 | Test the add notes function in the shopping cart | Danh | 0.15 |
| TC-11 | Check and edit address information in the system | Danh | 0.2 |
| TC-12 | Test the purchase process, including adding products to cart, editing quantities, and handling quantity limits. | Danh | 0.15 |
| TC-13 | Check out the login process, add products to cart, checkout and verify total amount. | Danh | 0.1 |
| TC-14 | Check price filter works properly between 50,000 and 400,000 | Danh | 0.15 |
| TC-15 | Check color filter feature | Danh | 0.15 |
| TC-16 | Check the calculation of original and promotional prices of products in the "FIFTYPERCENT" category | Danh | 0.15 |
| TC-17 | Test the feature of viewing posts in the blog page. | Danh | 0.15 |
| TC-18 | Check the product filtering feature by category in the product page. | Danh | 0.15 |
| TC-19 | Check for low stock notifications on the product detail page. | Danh | 0.15 |
| TC-20 | Check the product size selection and navigate correctly to the page with the size filter applied. | Danh | 0.15 |
| TC-21 | Check product filtering by color and make sure the products displayed are in the selected color. | Danh | 0.1 |
| TC-22 | Check product price consistency when adding to cart on the Team Whales x Grimm DC page. | Danh | 0.15 |
| TC-23 | Check the accuracy of product prices when applying discount percentages on the Turban page | Danh | 0.15 |
| TC-24 | Check product prices in the list and detail pages in the Wallet category. Make sure the price in the list matches the price in the detail page. Check pagination if there are many products. | Danh | 0.1 |
| TC-25 | Check the discount percentage consistency between the product list and the product details in the Wallet catalog.  Make sure that the discount percentage in the list matches the percentage in the details page. | Danh | 0.1 |
| TC-26 | Check all products in the "Tote Bags" category are displayed and can be successfully opened from the list.  Verify the product exists and displays correctly in the detail page. | Danh | 0.15 |
| TC-27 | Test the product sorting by price feature on the category page.  Make sure that the product that appears first after sorting by "Price: Descending" has a higher price than the product that originally appeared first. | Danh | 0.1 |
| TC-28 | Check product price consistency on product page and in cart | Danh | 0.2 |
| TC-29 | Check out the product price sorting feature by "Price: Descending" on the product category page: | Danh | 0.2 |
| TC-30 | check location | Hùng | 0.3 |
|  |  |  |  |

1. **Details of test cases**

| **ID** | **Steps** | **Expected Result** | **Test**  **Result** |
| --- | --- | --- | --- |
| TC\_1 | Open the login page  Try to login without entering a password → Shows the expected error message.  Try to login without entering an email  Try to login with valid information → Login successfully.  Log out after successful login → Log out successfully. | Authentication messages are displayed correctly.  Login and logout work as expected. | PASS |
| TC\_2 | 1. Open the registration page  2. Registration tab  3. Try registering to see if the error message is displayed as expected.  4. Log out after successful login | Authentication messages are displayed correctly.  Login, Registration and logout work as expected. | PASS |
| TC\_3 | 1. Open the product collection page  2. Get the number of products displayed on the page  3. Click on the corresponding product  4. Click on the "Add to Cart" button  5. Close the "Added to Cart" notification  6. Reopen the collection page  7. Repeat until all products are added to the cart  8. End of loop | All products on the collection page should be added to the cart without errors.The "Added to Cart" notification should close successfully after each product is added.  The cart should reflect the correct total number of added products. The loop should complete without interruptions or failures. | PASS |
| TC\_4 | 1. Go to https://grimmdc.com/  2. Click on any product  3. Click [Add to Cart]  4. Open Cart  5. Click [Delete] | - Show message "Cart is empty"  - Number of products in cart = 0 | PASS |
| TC\_5 | 1. Go to https://grimmdc.com/  2. Click on any product  3. Click [Add to cart]  4. Enter quantity  5. Click [Update cart] | Total = Unit price \* Quantity | PASS |
| TC\_6 | 1. Go to https://grimmdc.com/  2. Click [Search]  3. Enter the keyword "Sony x Grimm DC // Vietnamese Artist Shirt" | Check if the search information matches the keyword | PASS |
| TC\_7 | 1. Go to the page:  https://grimmdc.com  2. Get the number of products available in the product list.  3. Reopen the shopping cart and get the total displayed value.  4. Get the number of products in the shopping cart.  5. Loop through each product in the shopping cart:  Compare the calculated total value with the displayed total value. | Match the total value and the sum of each product | PASS |
| TC\_8 | 1. Open the specified URL (https://grimmdc.com/).  2. Go to the "FIFTYPERCENT" page: Click on the "FIFTYPERCENT" link.  3. Select the sorting option: Ascending  4. Count the number of products  5. Check the sort by price:  Verify the results | Show correct increment value | PASS |
| TC\_9 | 1. Open the specified URL (https://grimmdc.com/).  2. Go to the search box: Assign the search keyword as " Sơ Mi".  3. Check the search results  4. Finish when you have checked all the products. | The search keyword Shirt is in all products that have the keyword “Sơ Mi” | PASS |
| TC\_10 | 1. Open the specified URL (https://grimmdc.com/).  2. Click on the search bar  3. Click on the search box enter the keyword "SONY" in the search bar  4. Select the product in the list  5. Scroll down to display the "Add to Cart" button  6. Click on the "Add to Cart" button  7. Click on the note box. Enter "TEST CASE FIRST TIME" in the note box  8. Click on the update cart button  Check the note content | The note "FIRST TEST CASE" is saved and displayed correctly in the shopping cart | PASS |
| TC\_11 | 1. Open the specified URL (https://grimmdc.com/).  2. Click on the user icon  3. Click on the "View Address" link  4. Click on "Edit"  5. Fill in the information.  6. Click on the Save button  7. Verify that the information has been updated successfully. | Check the address information edit in the system. | PASS |
| TC\_12 | 1. Open the specified URL (https://grimmdc.com/).  2. Search for the product named "Black Dragon Head."  3. Add the product to the cart and open the cart.  4. Check the quantity of the product in the cart.  5. Try increasing the quantity beyond the allowed  6. Resolve the warning message about the maximum quantity limit.  7. Confirm that the quantity does not exceed the limit. | Show warning when quantity limit is exceeded.  The final quantity remains at a maximum of 2. | PASS |
| TC\_13 | 1. Open the website  2. Log out (if logged in)  3. Log in with huynhto@gmail.com  4. Select a product from the FIFTYPERCENT category  5. Add the product to the cart  6. Open the cart and proceed to checkout  7. Save the value of:  Provisional (tam\_tinh)  Shipping fee (phi\_ship)  Total (tong\_tien)  8. Recalculate the total and compare it with the displayed total | Display the correct estimated value, shipping cost and total amount.  The calculated total and the displayed total must match. | PASS |
| TC\_14 | 1. Open the website  2. Click on the "FIFTYPERCENT" link  3. Open the sidebar  4. Select the price filter (3rd element)  5. Save the price of the first product  6. Check if the product exists  7. Verify if the price is within the allowed range (50,000 to 400,000) | The price of the product must be within the specified range. | PASS |
| TC\_15 | 1. Open the website  2. Click on the 3rd item menu  3. Open the sidebar  4. Select the "Green" color filter  5. Pause for 5 seconds to let the page load  6. Check if there are any products in the list  7. If there are no products, verify that the "No products match the request" message is displayed | Show "No products match request" message when there are no matching products | PASS |
| TC\_16 | 1. Open the website  2. Click on the "FIFTYPERCENT" link  3. Loop through each product in the list (from 1 to 10)  4. Scroll down the page to display the products  5. Wait for the product element to appear  6. Click on the product  7. Get the actual price and the discounted price  8. Calculate the original price = actual price + discounted price  9. Compare with the original price displayed on the page  10. Return to the product list and continue with the next product | Expected Result The calculated base price should match the base price displayed on the page. | PASS |
| TC\_17 | 1. Open the main page  2. Click on the "Blog" item on the menu  3. Count the total number of articles displayed on the page  4. Initialize the counter variable with the value 1  5. Start the loop until counter <= totalArticles.  6. Click on the current article  7. Wait for the article content to appear for 10 seconds  8. Check if the article content has been loaded  9. If the article content has been loaded.  10. Add the notification ${counter} viewed to the page  11. Wait for the notification to appear for 30 seconds  12. Check if the notification content is correct  13. Return to the article list page  14. Pause for 3 seconds  15. Increase the counter by 1  16. Repeat until all articles have been viewed  17. Add the notification "All articles have been viewed!" to the page  18. Verify the notification content | Posts are opened and displayed in full. The loop works correctly, processing each post without error. Content validation and notifications work as expected. | PASS |
| TC\_18 | 1. Open the product list page.  2. Click on the open filter icon  3. Select the product category (Croptop).  4. Check the selected category is correct  5. Count the number of products after filtering  6. Initialize the variable i = 1  7. Start the loop until i <= count  8. Get the product name from the list  9. Check the product name contains the keyword of the selected category (Croptop)  10. Confirm that the product name matches the keyword  11. Increase the value of i by 1  12. End the loop after all products have been checked | Product filtering works correctly according to categories. All filtered products match the search keyword. Filter results and product names are verified correctly. | PASS |
| TC\_19 | 1. Open the product page  2. Initialize variable i = 0  3. Initialize variable isErrorPresent = false.  4. Start looping until: i < 100  isErrorPresent != true  5. Check for the appearance of the "Add to Cart" button  6. Check if the error message (p > strong) appears  7. If there is an error message  8. If the error message appears in the modal, perform the following steps:  9. Check the appearance of the closing modal (popup)  10. End the loop when the notification is detected or has checked 100 times | The "Add to Cart" button works properly.Inventory notifications are displayed correctly. Modal displays and handles content correctly when a product is running low. | PASS |
| TC\_20 | 1. Open the home page  2. Click on the "See more" button  3. Open the left filter (sidebar)  4. Select the Ravolution x Grimm DC  5. Reopen the left filter  6. Select size L  7. Save the selected size (L)  8. Click on the sort option  9. Pause for 10 seconds  11. Count the number of displayed products  12. End the loop when all products have been checked | Size L is selected correctly. Size filter in URL is applied correctly. All products displayed correspond to the selected size. | PASS |
| TC\_21 | 1. Open the Ravolution x Grimm DC catalog page  2. Open the left filter (sidebar)  3. Select the color Black.  4. Save the Black value to the input variable  5. Pause for 5 seconds → Success (but can be optimized).  6. Count the number of products → 2 products  7. Increase the value of variable i to browse to the next product  8. End the loop when all products have been checked | Filtering products by color Black works correctly. All filtered products have the correct color. No errors when getting values ​​or navigating URLs. | PASS |
| TC\_22 | 1. Open Team Whales x Grimm DC category page  2. Count the number of products  3. Initialize counter variable i = 1  4. Start loop to check each product:  5. Select product  6. Get product price on detail page  7. Convert price from string to float  8. Add product to cart.  9. Verify product price in cart  "implicit locators are deprecated" locator explicitly.  10. Finish when all products are checked. | Product prices on the page and in the shopping cart match. All products are checked for accuracy. No logical or calculation errors | PASS |
| TC\_23 | 1. Open the grimmdc.com homepage.  2. Open the extended category  3. Open the sidebar  4. Select the "Turban" category  5. Count the number of products in the category  6. Initialize the counter variable i = 1  7. Finish when all products are checked | Product prices in the list and in the product details match. Discount percentage is applied correctly. Price calculation after discount works correctly. | PASS |
| TC\_24 | 1. Open the home page  2. Set the window size to 1552 x 832  3. Open the menu and select the "Wallet" category  4. Count the number of products in the category using executeScript  5. Close the browser | Prices in the list and in the product details match. Pagination works properly when there are multiple pages. All steps are accurate and stable | PASS |
| TC\_25 | 1. Open the home page → Success.  2. Set the window size to 1552 x 832 → Success.  3. Open the menu and select the "Wallet" category  4. Count the number of products in the category using executeScript  6. Initialize variable i = 1  5. Close the browser | The discount percentage in the list and in the product details match. All steps are accurate and stable. No errors or warnings. | PASS |
| TC\_26 | 1. Open the home page  2. Set the window size to 1552 x 832  3. Open the menu and select the "Tote Bag" category  6. Counting total products using executeScript  7. Initializing counter = 1  24. End the loop when all products have been viewed  25. Add notification All products have been viewed!  27. Close browser | Successfully opened and verified 5 products in the "Tote Bag" category. All products are displayed correctly and there are no errors during page loading. Notifications are fully displayed upon completion. | PASS |
| TC\_27 | 1. Open the home page .  2. Set the window size to 1552 x 832  3. Open the menu and select the category  4. Get the first product price before sorting (gia\_moi)  6. Verify the displayed price is correct  7. Click on the sort menu  8. Select Price: Descending from the list  9. Close browser | The price sorting feature by "Price: Descending" has been successfully tested. The price of the first product after sorting is higher than the price of the initial product. There are no errors in data loading and processing. | PASS |
| TC\_28 | 1. Open the browser  2. Select the FIFTYPERCENT category FIFTYPERCENT category is successfully opened.  3. Select the first product  4. Save and verify the product price on the detail  5. Add product to cart  6. Check the displayed cart successfully and displays product information.  7. Save and verify price in cart  8. Close notification | Check cart has been successfully displayed and product information is displayed. | PASS |
| TC\_29 | 1. Check Price Sort: Descending  2. Select "Price: Descending"  3. Get the first price (gia\_thap) → 1$ VND  4. Finish and close browser | Check price descending successfully | PASS |
| TC\_30 | 1. Open the page, set the window size  Check Address In Footer  3. Get address in footer (outside)  5. Verify address in footer  6. Go to "Contact" page  7. Get address in page (inside) 13c/9 Ky Dong,  9. Verify address in page  10. Close browser | Check that the address displayed on the Grimm DC website's home page and contact page match. | PASS |

**Chapter 3. CONCLUSION**

In this chapter, we examined the application of Selenium IDE as a tool for automating web application testing, with a particular focus on the e-commerce website grimmdc.com. Selenium IDE, known for its intuitive record-and-playback interface, offers an accessible entry point for beginners in test automation. It enables users to quickly create and execute test cases without requiring extensive programming knowledge, making it an ideal choice for initial explorations into automated testing.

The chapter demonstrated how Selenium IDE was used to test core functionalities of grimmdc.com, including search operations, user authentication (such as login, logout, and account creation), product filtering and sorting, and the shopping cart workflow (covering adding products, adjusting quantities, and completing checkout). These tests executed effectively for standard user interactions, underscoring Selenium IDE's capability to automate and validate straightforward processes reliably.

However, the testing process also exposed some limitations of Selenium IDE. When confronted with more complex or dynamic features—such as asynchronous content loading or intricate user interactions—the tool required manual script adjustments, revealing its constraints in handling advanced scenarios. This observation suggests that while Selenium IDE is well-suited for basic to moderately complex testing, more sophisticated web applications may necessitate advanced tools like Selenium WebDriver, which provides greater flexibility through programmatic control.

Nevertheless, the importance of automated testing in maintaining the quality and dependability of web applications like grimmdc.com remains clear. Selenium IDE offers significant benefits, including a user-friendly environment and instant feedback, which support rapid learning and test prototyping. Additionally, its feature to export scripts into various programming languages allows integration into broader testing frameworks, offering a pathway for scaling testing efforts as needs grow.

In summary, Selenium IDE proved to be a valuable tool for automating essential test cases on grimmdc.com, highlighting its strengths in simplicity and efficiency while also acknowledging its boundaries in complex scenarios. It stands as a vital resource for testers, fostering consistency and early issue detection, though users should be prepared to complement it with more robust solutions for comprehensive testing requirements.

**References**

Selenium IDE Official Documentation. (n.d.). Getting Started. Retrieved from [*https://www.selenium.dev/selenium-ide/docs/en/introduction/getting-started*](https://www.selenium.dev/selenium-ide/docs/en/introduction/getting-started)

grimmdc.com. (n.d.). Home Page. Retrieved from [*https://grimmdc.com/*](https://grimmdc.com/)

BrowserStack. (n.d.). What is Selenium IDE? Retrieved from [*https://www.browserstack.com/guide/what-is-selenium-ide*](https://www.browserstack.com/guide/what-is-selenium-ide)

Guru99. (n.d.). Introduction to Selenium IDE Tutorial. Retrieved from [*https://www.guru99.com/introduction-selenium-ide.html*](https://www.guru99.com/introduction-selenium-ide.html)

Sauce Labs. (n.d.). Selenium IDE Tutorial: Overview, Features & Benefits. Retrieved from [*https://saucelabs.com/resources/blog/selenium-ide-tutorial-overview-features-benefits*](https://saucelabs.com/resources/blog/selenium-ide-tutorial-overview-features-benefits)