Automated Testing Plan

for

OWASP Juice Shop

By:

Hoda Dehghanisanij

Michael Lum

Wayne Lin

California State University, Fullerton

Fall 2021

Professor Heckathorn

Table of Contents

[1. Introduction 2](#_Toc89700894)

[1.1 Scope 2](#_Toc89700895)

[1.2 purpose 2](#_Toc89700896)

[1.3 Application under test 2](#_Toc89700897)

[1.4 Automated Testing Software 3](#_Toc89700898)

[1.5 Testing Modes in Katalon 3](#_Toc89700899)

[2. Testing Approach 3](#_Toc89700900)

[3. Test Scenario 4](#_Toc89700901)

[4. Test Case 5](#_Toc89700902)

[5. Test Procedure 5](#_Toc89700903)

[5.1 Login Form 5](#_Toc89700904)

[5.2 User Registration Form 6](#_Toc89700905)

[5.3 Search Bar 9](#_Toc89700906)

[5.4 Basket 12](#_Toc89700907)

[6. Report 17](#_Toc89700908)

[7. Roles and Responsibilities 20](#_Toc89700909)

[8. Resources 21](#_Toc89700910)

# Introduction

## 1.1 Scope

The scope of this project is to use katalon studio to automate software testing and inspect the system considerations that test processes in determining system and software correctness, completeness, accuracy, consistency, and testability, as well as the relevant test documentation.

## 1.2 purpose

The purpose of this project is:

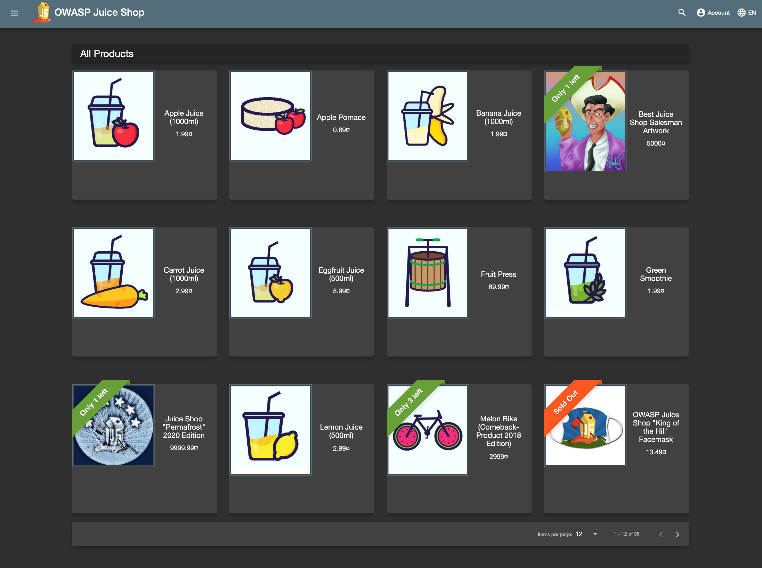
1. Define a framework for testing processes, activities, and tasks that is consistent.
2. Define the test tasks, required inputs, and required outputs
3. Identify the minimum test cases corresponding to integrity levels
4. Define the purpose and substance of test documentation such as the Test Design, Test Case, Test Procedure, Test Report, and Test Log.

## 1.3 Application under test

Open-source website description: OWASP Juice Shop

URL: https://juice-shop.herokuapp.com/#/

GitHub: https://github.com/OWASP/owasp.github.io



## 1.4 Automated Testing Software

Katalon Studio was used for automated software testing. Katalon studio has a user-friendly UI, and is created by Katalon, Inc. With a specific IDE interface for online websites, API, mobile, and desktop application testing, the product is developed on the open-source automation frameworks Selenium and Appium. It has two different ways to create test cases: a manual mode for non-technical users and a script mode for experienced testers to author automation tests with syntax highlight and intelligent code completion. [1]

Advantage of Katalon Studio:

* It's possible that Katalon Studio's users are inexperienced testers. With the manual mode, Katalon Studio hides all technical difficulties behind the scenes and delivers a user-friendly UI.[2]
* Katalon Studio is a unified bundle that comprises practically all required components, such as Java, Android SDK, Web drivers for browsers, and dependencies.[2]
* Object spy is supported by Katalon Studio for both web (using browser plugins for IE, Firefox, and Chrome) and mobile apps (by using the screenshot simulator). High lighting captured objects in AUT (Application Under Test) is also supported by Katalon Studio.[2]
* Test recording is the greatest approach to help users understand and quickly adapt to test automation. Katalon Studio can record and produce test scripts on both Web and mobile apps.[2]

## 1.5 Testing Modes in Katalon

Manual Mode allows users to setup test cases by manually executing the steps required to conduct testing. This process begins by creating a blank test case. Once blank test case is created, ‘Record Web’ is selected. Provide the URL to be tested and then click record. Katalon will open a browser and navigate to the URL provide. Katalon will now record the steps the user takes to test a feature. Upon completion of the testing, the user will stop recording. Katalon saves all the steps the user has taken. The user can then edit the steps as required in order to complete a test case. The user has now manually inputted the steps to complete a test case. The test case can be running by clicking on the ‘Run’ button.

Script Mode allows users to setup test cases by writing script and an experience user can provide the steps to complete test cases. Upon completion of script writing, Katalon Studio can execute to the script to conduct software testing by clicking on the ‘Run’ button.

Manual model was used for this project due to the ease of implementation.

# Testing Approach

Using Katalon Studio as an automated testing system.

Script Language: Groovy, a language on top of java – keyword based.

Operating System: Microsoft Windows - MacOS

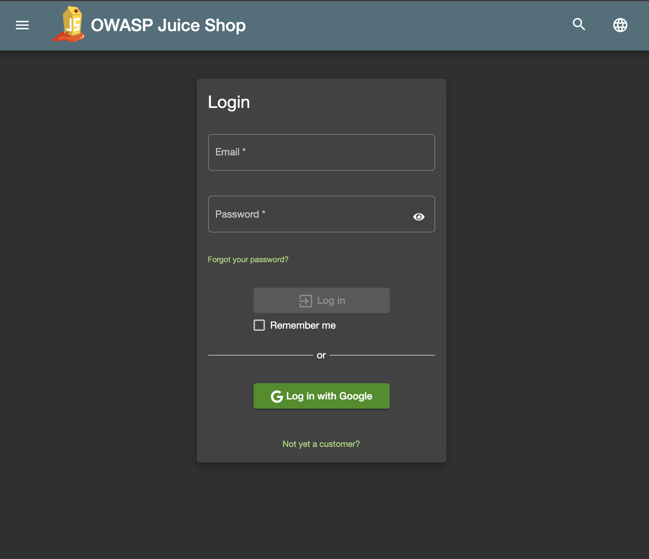
Browser Used: Chrome - Firefox

During this project, we’ll use Manual Mode for automated testing and tried to focus on the most critical functionality.

# Test Scenario

These areas in the application will be tested:

Module 1 - Login Form



The objective is to verify successful login with valid credentials and prevention of access with invalid credentials. As a prerequisite of this module, an account was created. This module will be conducted by attempting to login with valid credentials used to create an account. The module will be finalized with login attempts with invalid credentials. The conditions for this module to pass are:

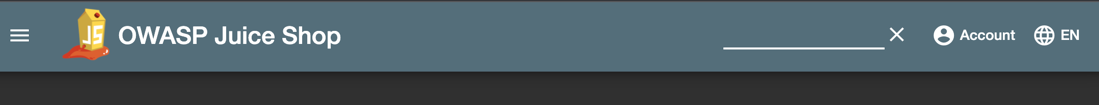
1. Login must be successful with valid credentials
2. Login must be unsuccessful with invalid credentials

Module 2 - Registration Form

Graphical user interface

Description automatically generated

Module 3 - Search Bar



Module 4 – Basket and Checkout

The components listed below are always present and included in every test case:

* Test Case ID
* Software Module
* Test Run History
* Steps
* Actual Input
* Expected Output

# Test Case

The main goal of this step is to determine whether a component of software passes or fails in terms of functionality. The scope of testing is limited to the Login Form, Registration Form, Search Bar, and Basket.

|  |  |  |
| --- | --- | --- |
| Module 1 – Login Form | | |
| Test Case | Pass | Fail |
| Invalid Logon | Passed |  |
| Test Valid Logon | Passed |  |
| Module 2 – Registration Form | | |
| Registration-001 | Passed |  |
| Registration-002 | Passed |  |
| Registration-003 | Passed |  |
| Module 3 – Search Bar | | |
| Search 1 - item name match | Passed |  |
| Search 2 - item desc match | Passed |  |
| Search 3 - no results | Passed |  |
| Module 4 – Basket / Checkout | | |
| Add to Basket 1 - no restriction | Passed |  |
| Add to Basket 2 - item limit | Passed |  |
| Add to Basket 3 - OOS | Passed |  |
| Modify Basket 4 - no restriction | Passed |  |
| Modify basket 5 - limit | Passed |  |
| Checkout / Order 6 - credit card | Passed |  |
| Checkout X - bad wallet balance |  | Failed |

|  |  |  |  |
| --- | --- | --- | --- |
| **Module No.** | **Module Description** | **Test Case** | **Pass/Fail** |
| 1 | Login Form | Test Valid Logon | Pass |
| Test Invalid Logon | Pass |
| 2 | Registration Form | Registration-001 | Pass |
| Registration-002 | Pass |
| Registration-003 | Pass |
| 3 | Search Bar | Search 1 - item name match | Pass |
| Search 2 - item desc match | Pass |
| Search 3 - no results | Pass |
| 4 | Basket / Checkout | Add to Basket 1 - no restriction | Pass |
| Add to Basket 2 - item limit | Pass |
| Add to Basket 3 - OOS | Pass |
| Modify Basket 4 - no restriction | Pass |
| Modify Basket 5 - limit | Pass |
| Checkout / Order 6 - credit card | Pass |
| 4X | Checkout Error | Checkout X – bad wallet balance | Fail |

# Test Procedure

## 5.1 Login Form

Prior to executing the test module, a valid account was created with OWASP Juice store. The valid account created for this test module utilized the following logon credentials:

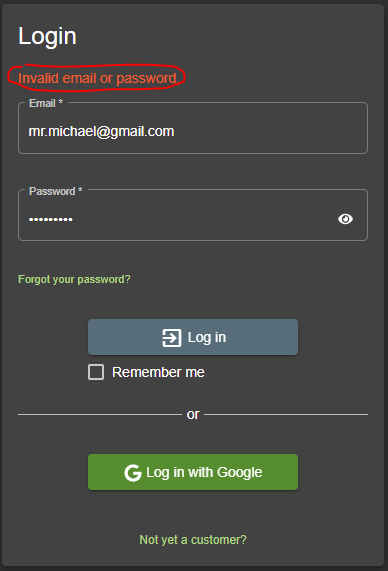
1. Email: mr.michael@gmail.com
2. Password: ILoveCats

This test module utilizes 3 equivalence classes (EC) and two test cases. The equivalence classes are representative of valid or invalid logon credentials. The ECs and test cases as defined below:

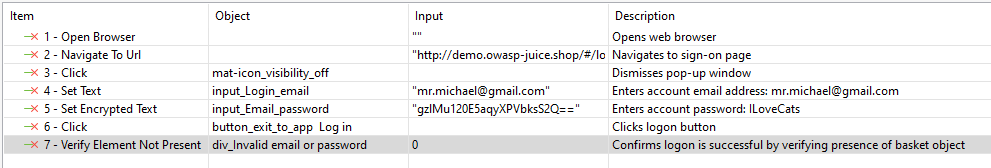
|  |  |  |
| --- | --- | --- |
| EC No | EC Credential Values | EC description |
| 1 | Email: mr.michael@gmail.com  Password: ILoveCats | This EC is representative of valid logon credentials for an existing account made with OWASP Juice Store. |
| 2 | Email: mr.michael@gmail.com  Password: IHateCats | This EC is representative of invalid logon credentials for an existing account made with OWASP Juice Store. The email address is for an existing account, but the password is invalid. |
| 3 | Email: mr.cat@gmail.com  Password: ILoveCats | This EC is representative of invalid logon credentials for an account that doesn’t exist. The email address is not registered with any OWASP Juice Store account, but the password is used by an existing account. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case No | Test Case Description | Acceptance Criteria | Valid logon ECs | Invalid logon ECs |
| 1 | Test valid logon credentials.  Ensures login is successful when provided valid login credentials for an existing account. | The message “Invalid email or password.” is not present | EC1 | N/A |
| 2 | Test invalid logon credentials.  Ensure login is not possible when provided an email for existing account and invalid password. Ensure login in is not possible when provided an email not linked to an existing account and password used by an account. | The message “Invalid email or password” is present for login attempts with:   1. Valid email and invalid password 2. Invalid email and valid password | N/A | EC2, EC3 |

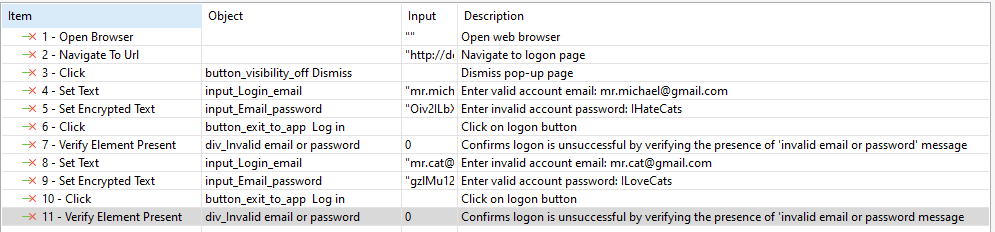
The acceptance criteria for both test cases involves confirming the presence of the “Invalid email or password” message shown in the image below.



The steps for test case no 1 are as follows taken from Katalon Studios:



The steps for test case no 2 are as follows taken from Katalon Studios:



## 5.2 User Registration Form

**Manual**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case Identifier | Random Input Value | Valid Equivalence Classes | Invalid Equivalence Classes |
| 1 | Hoda.com |  | EC2 |
| 2 | hoda@gmail.com | EC1 |  |
| 3 | H123 |  | EC3 |
| 4 | Hd12345 | EC2 |  |
| 5 | Hd12345 | EC5 |  |
| 6 | H12345 |  | EC6 |
| 7 | Mother's maiden name? | EC7 |  |
| 8 | Mom | EC8 |  |

For email field:

EC1. Email address is filled and has @, valid.

EC2. Email address is not filled or doesn’t have @, invalid.

For password field:

EC3. Password must be between 5-40 characters, valid.

EC4. Password is less than 5 or greater than 40 characters, invalid.

For repeat password field:

EC5. Repeat password is filled and matched to password, valid.

EC6. Repeat password is not filled or not matched to password, invalid.

For security question field:

EC7. Security questions are filled out and are chosen from dropdown menu, valid.

For answer security question field:

EC8. Security questions are not filled, invalid.

**Automated**

Test case Registration-001

Item to be tested:

Text

Description automatically generated

Log Viewer after running the test:

Graphical user interface, text, application

Description automatically generated

Test Case registration-002

Item to be tested:

Text

Description automatically generated

Log Viewer after running the test:

Text

Description automatically generated

Test Case registration-003

Item to be tested:

Text

Description automatically generated

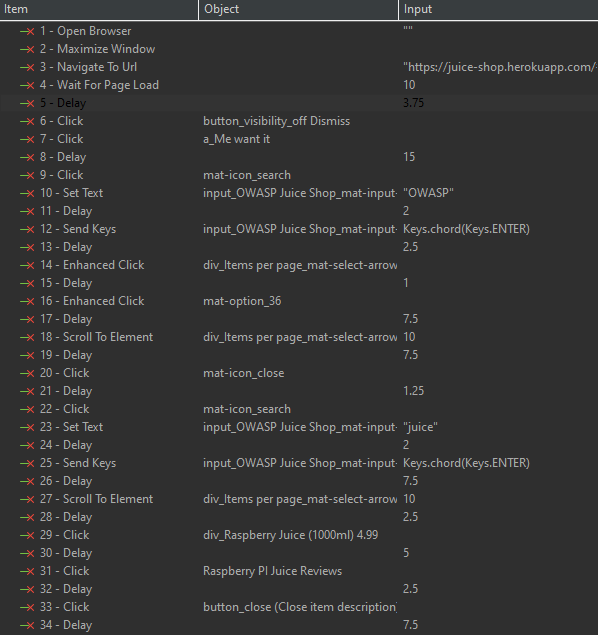
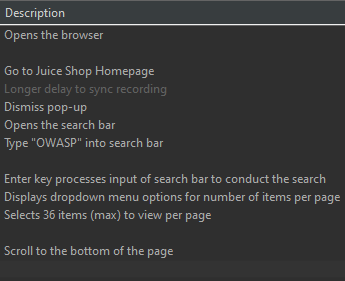
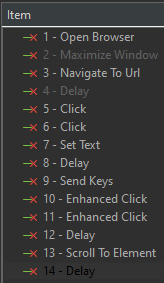
Log Viewer after running the test:

Graphical user interface, text, application

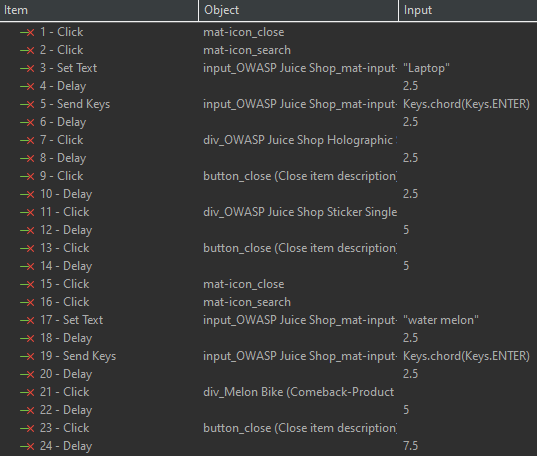
Description automatically generated

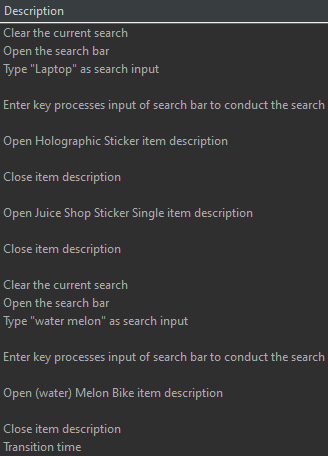
## 5.3 Search Bar

Test Case 1 – Search query matches item names

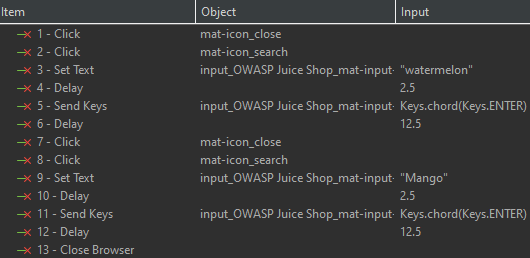
 

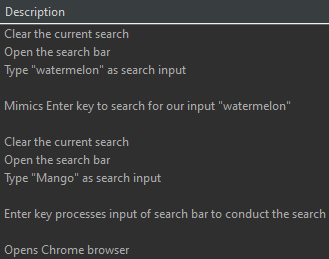
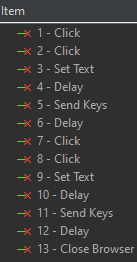
Test Case 2 – Search query matches item description



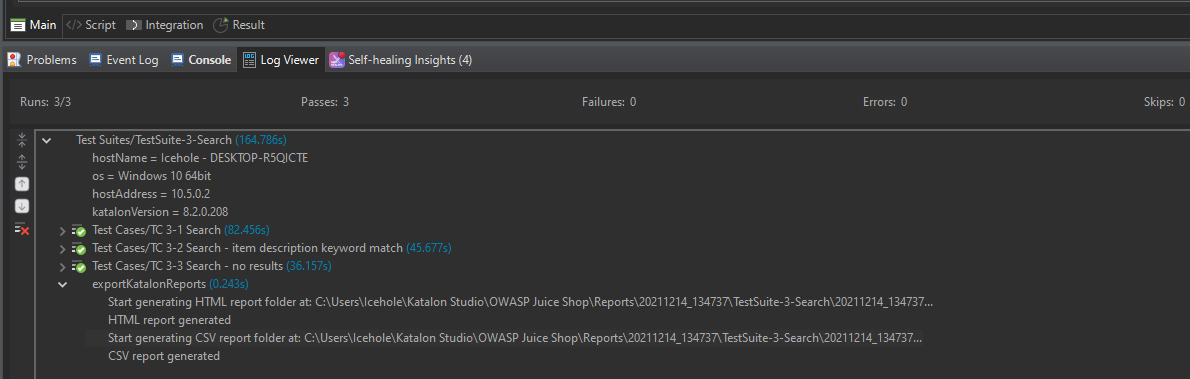


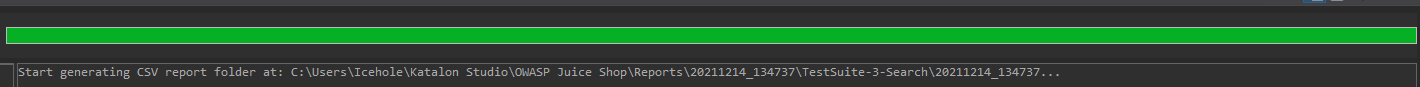
Test Case 3 – Search returns no results





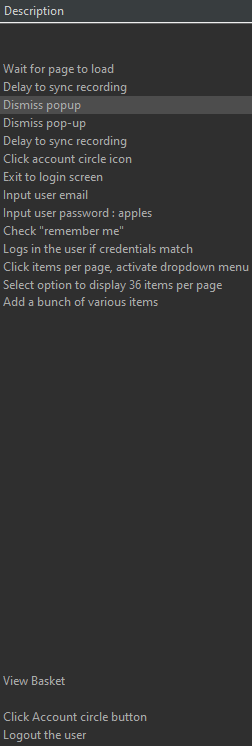
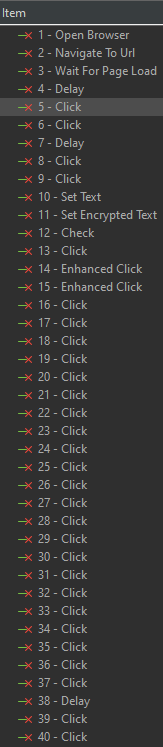
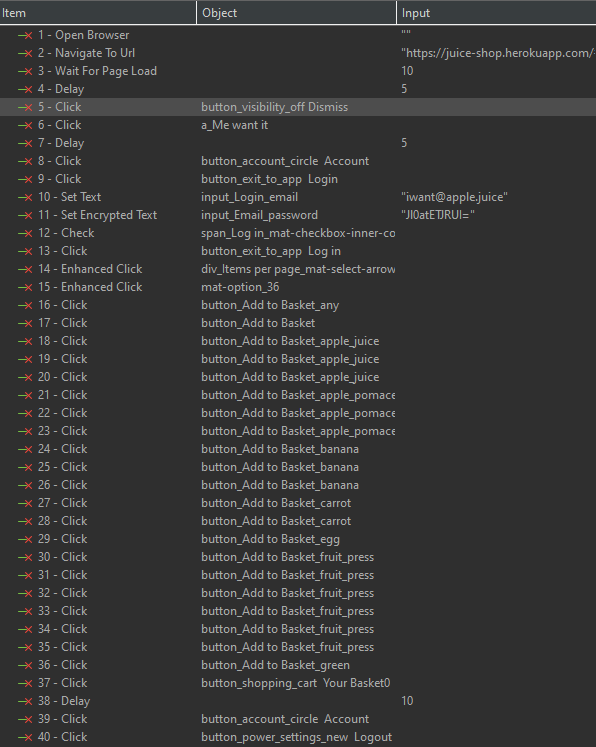
Log Viewer for Test Suite 3



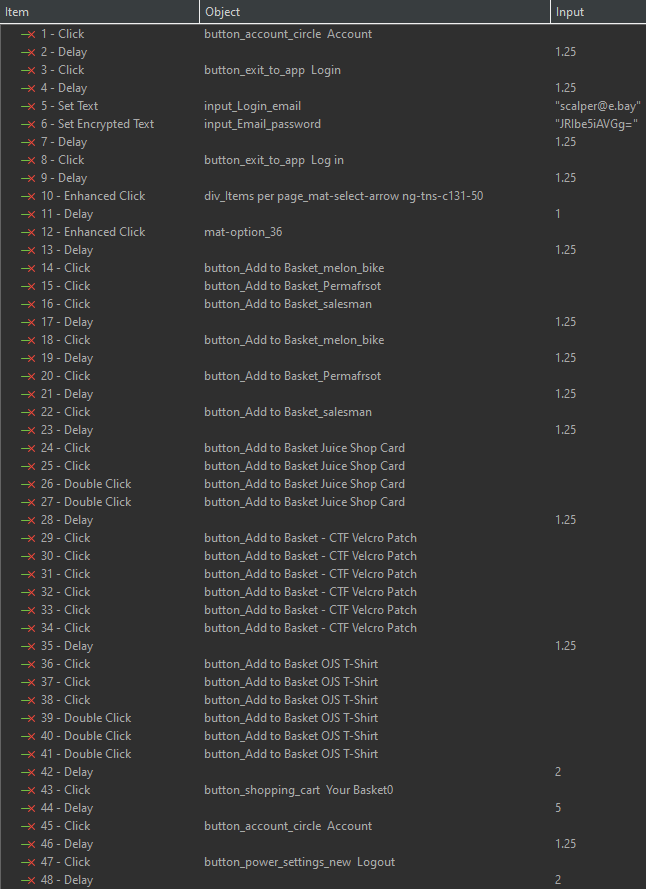


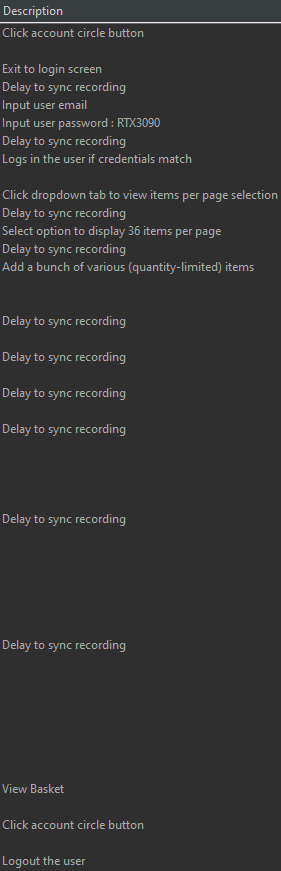
## **5.4 Basket and Checkout**

Test Case 1 – multiple quantities are in stock

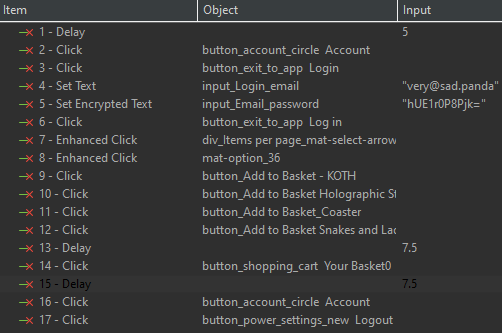


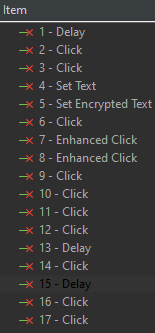
Test Case 2 – item limit of [1-5] enforced



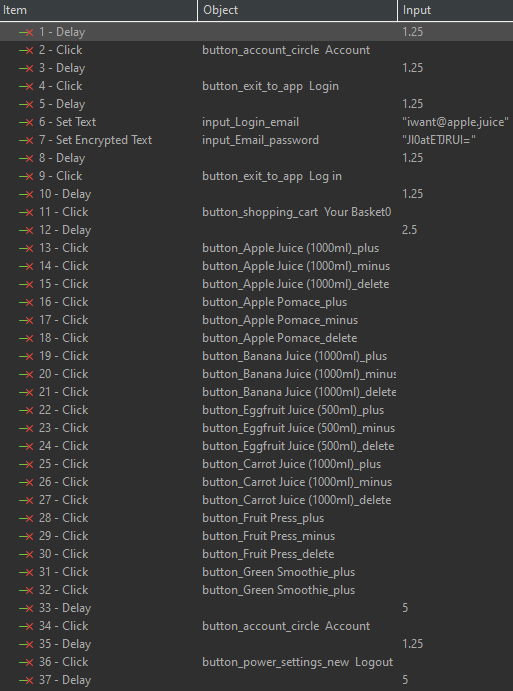


Test Case 3 – items SOLD OUT



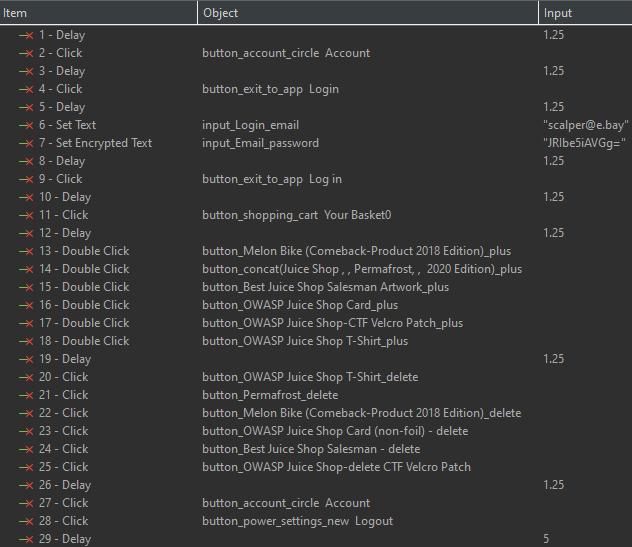


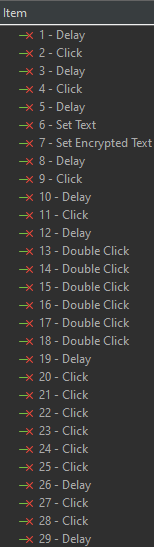
Test Case 4 – Modify Basket – multiple quantities



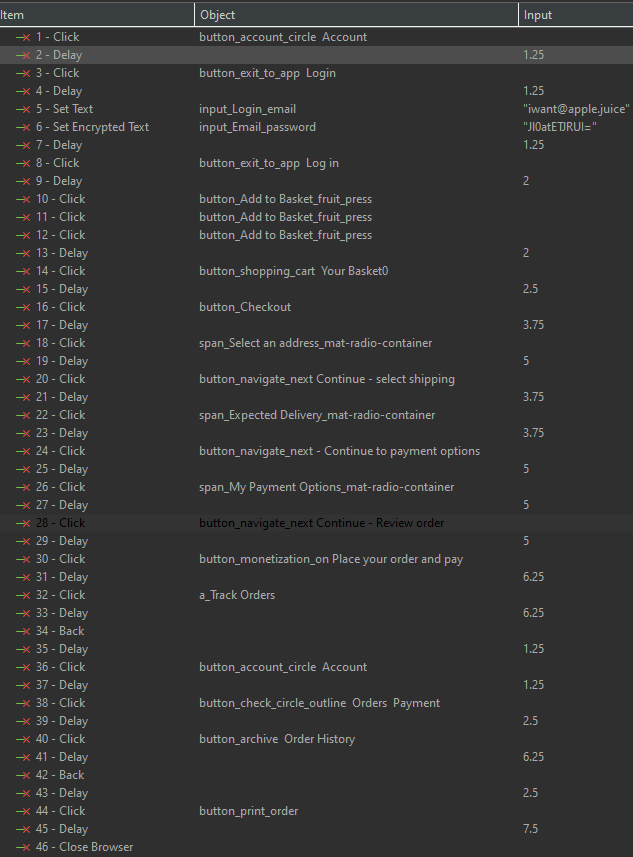


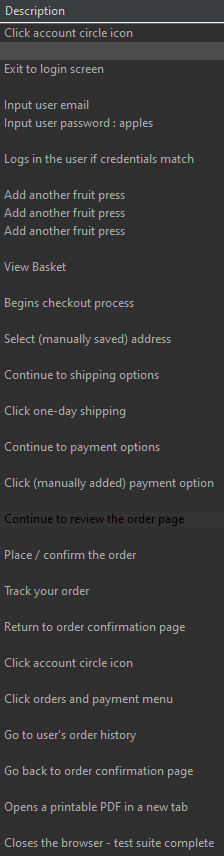
Test Case 5 – Modify basket – item limit restrictions



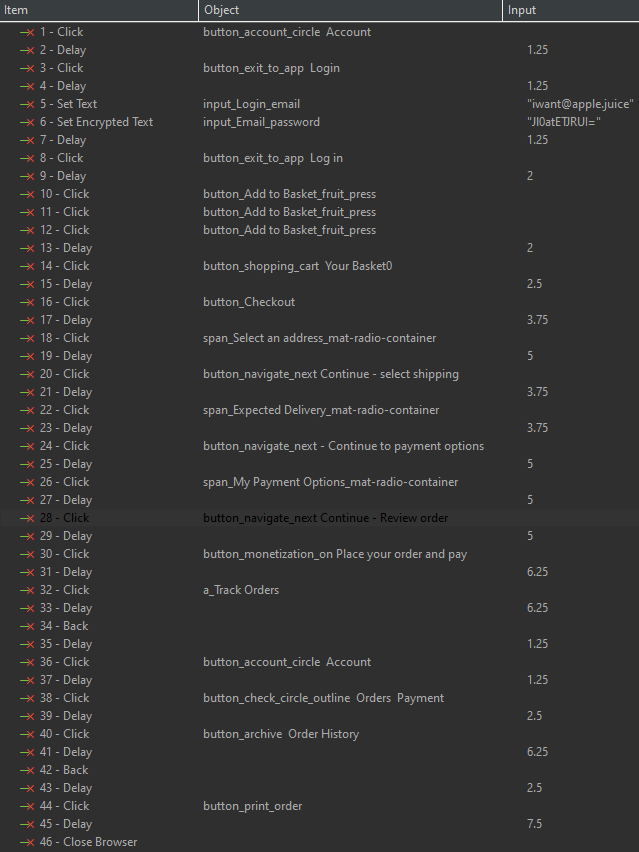


Test Case 6 - Checkout



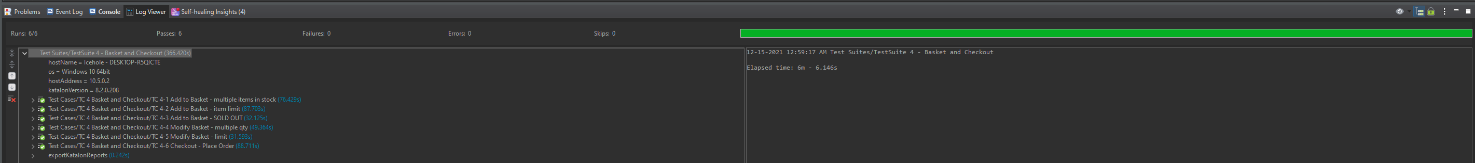
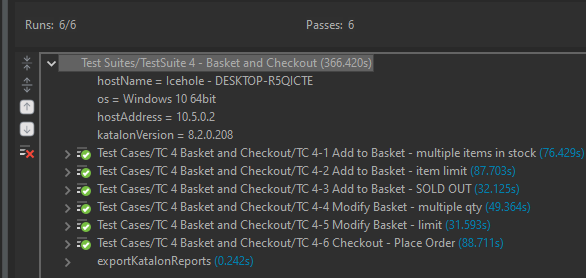


Test Case X (separate suite) – Error Case – Insufficient funds in digital wallet

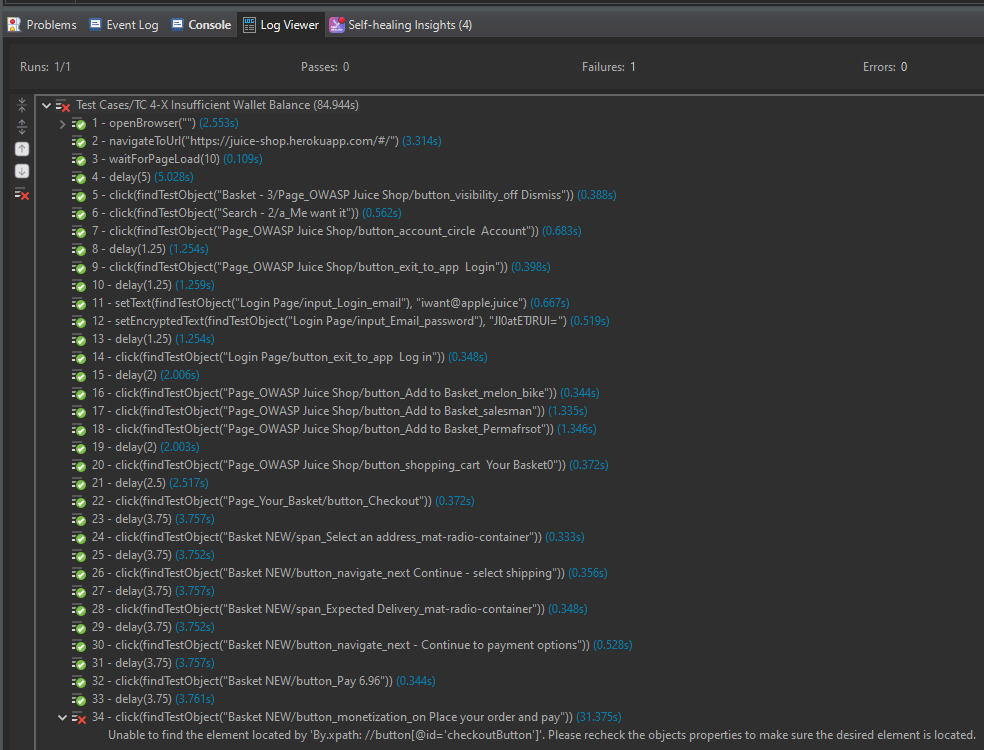




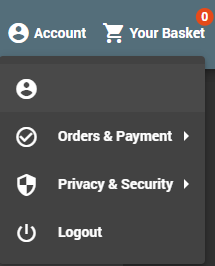
Log Viewer for Test Suite 4



Log Viewer for Test Suite 4X

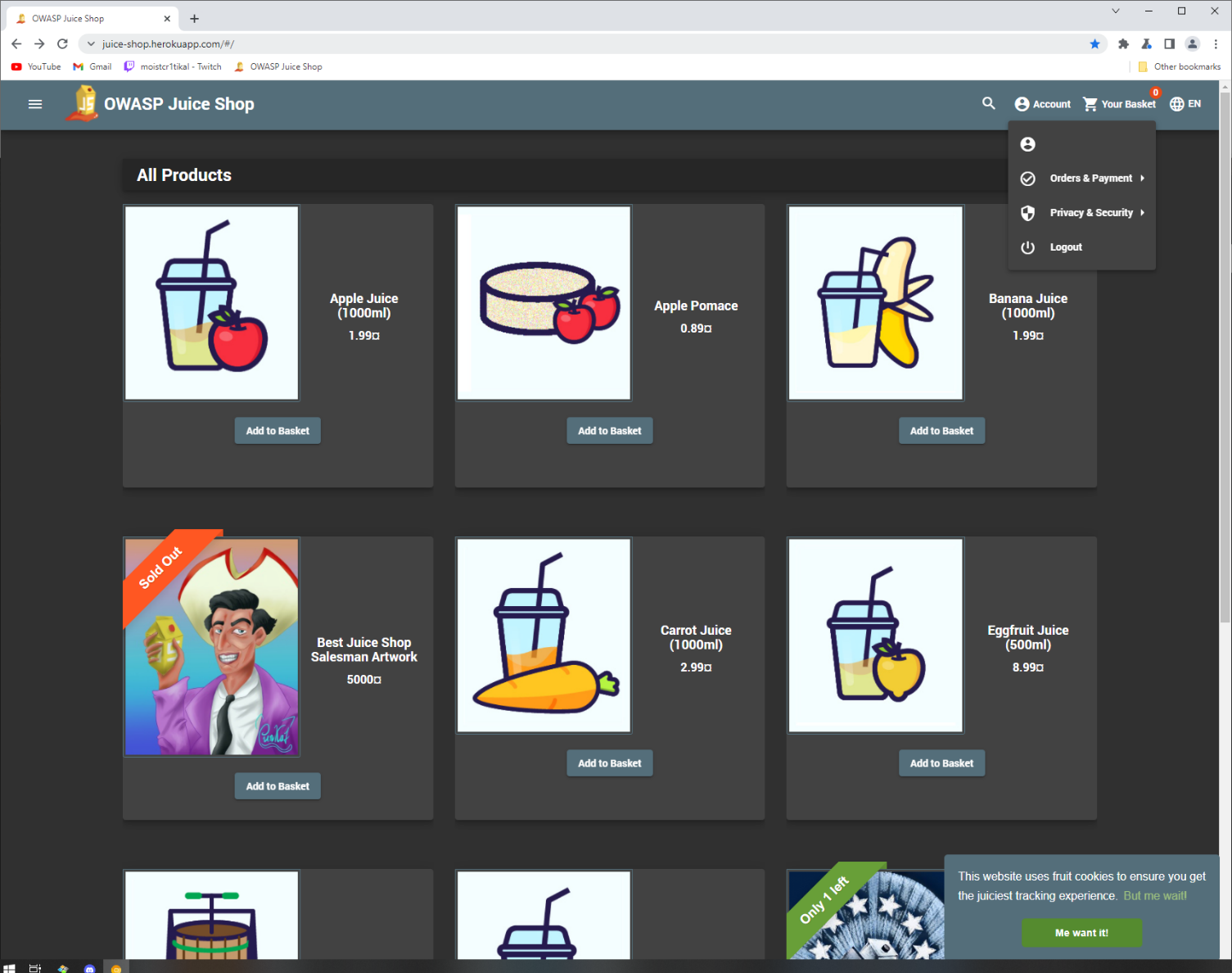


## **5.5 User Profile (Manual Test)**

BUG: The OWASP Juice Shop site flushes user data every few hours but leaves behind zombie accounts with blank usernames.

Trying to access the profile leads to a 500 Error.

Add to basket still shows but is nonfunctional.



Also see slides 9-11

## **5.6 Checkout Information (Manual Test)**

5.6.1 Add New Address

*Equivalence Classes*

EC1: all forms are filled in

EC2: at least one form is empty

EC3: Mobile number is between 10000000 – 9999999999 (8-10 digits no decimals)

EC4: Mobile number is 7 or fewer digits

EC5: Mobile number is 11 or more digits

EC6: Mobile number is a decimal

EC7: Zip code is between 1 and 8 characters

EC8: Zip code is longer than 8 characters

Valid Equivalence classes: EC1, EC3, EC7

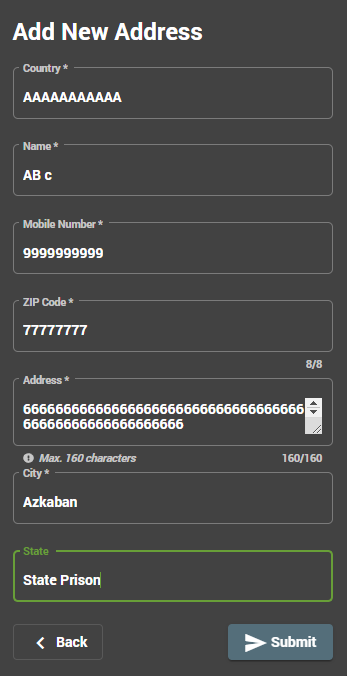
Invalid Equivalence classes: EC2, EC4, EC5, EC6 (no error message but cannot continue), EC8

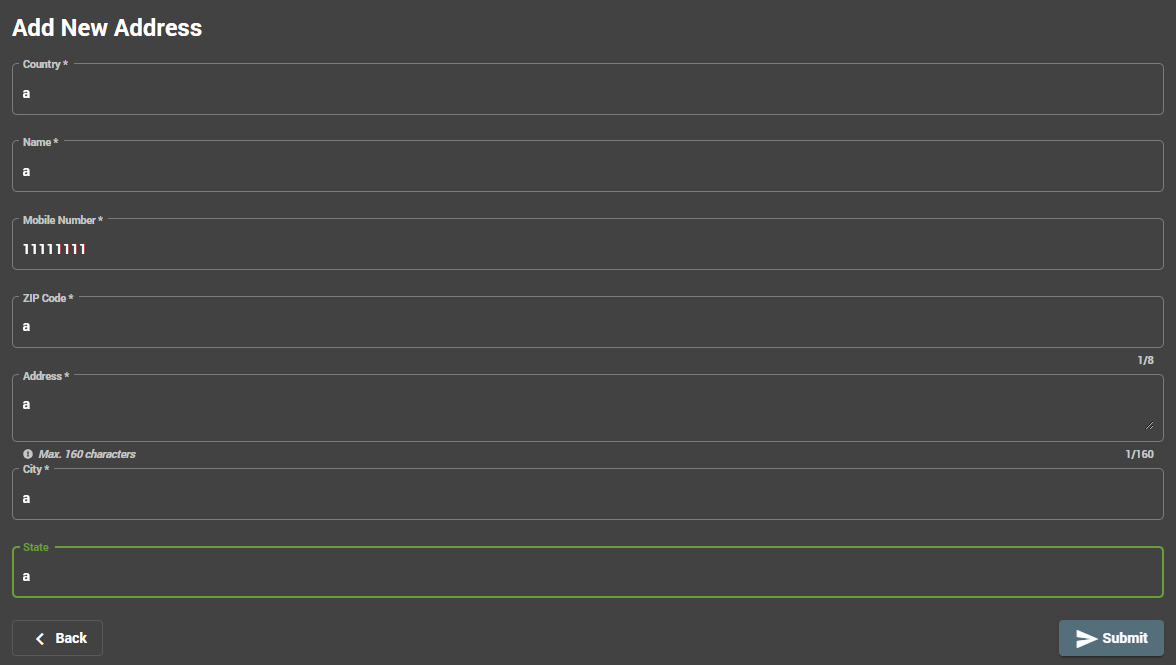
|  |  |  |
| --- | --- | --- |
| **Input Condition** | **Valid Equivalence Classes** | **Invalid Equivalence Classes** |
| All forms filled in | EC1 | EC2 |
| Mobile number is valid | EC3 | EC4, EC5, EC6 (no error msg.) |
| Zip code is valid | EC7 | EC8 |

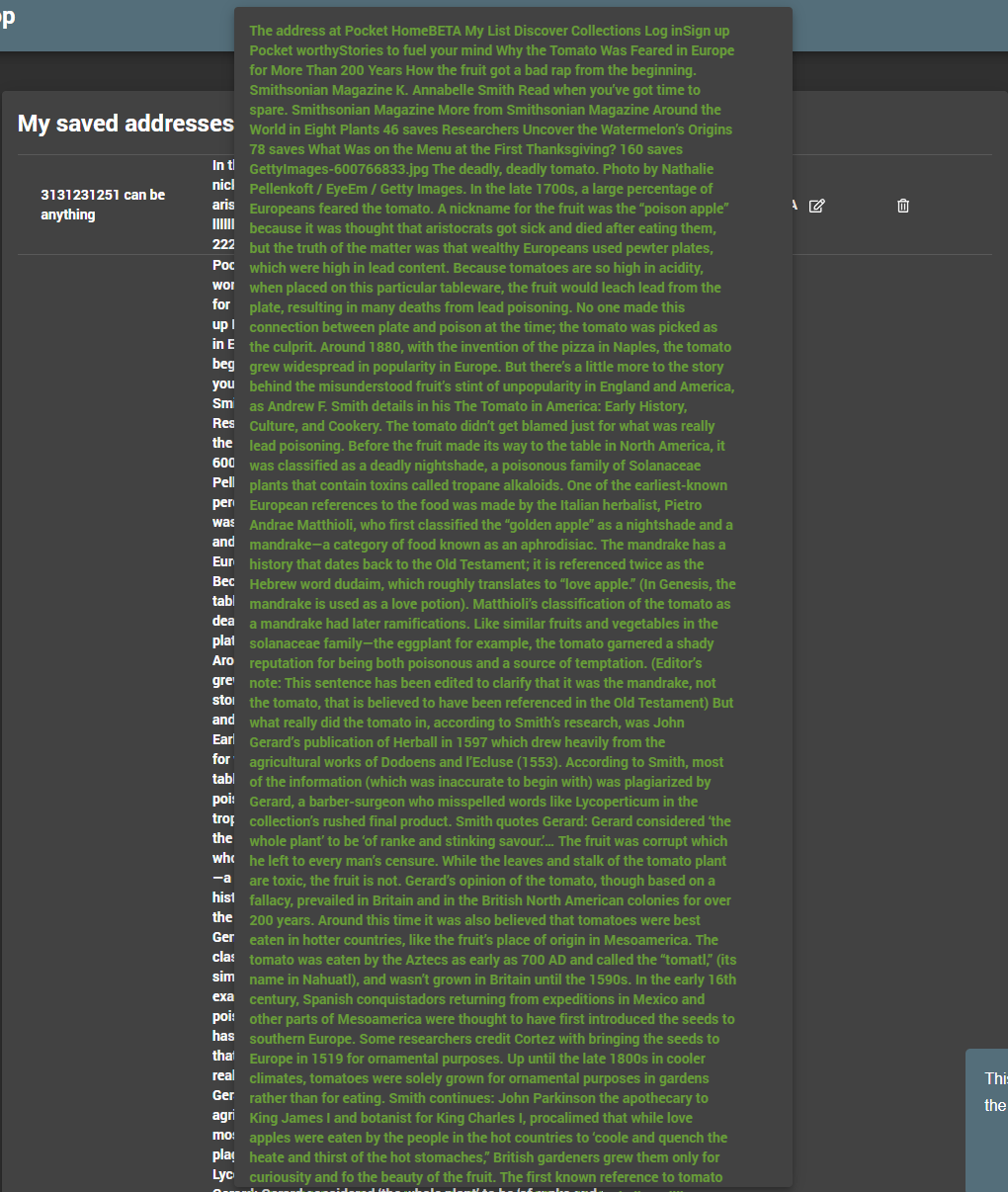
The real-world validity of the website’s address possibilities is suspect.

Bottom image: no length cutoffs for most fields may encourage spam

The address field has a hard limit of 160 characters.

 **Valid Case (EC1, EC3, EC7)** **Invalid Case (EC5, EC8)**

 **Valid Case (EC1, EC3, EC7)**

**Valid Case but spam**

5.6.2 Credit Card Information

*Equivalence Classes*

EC1: No fields are blank

EC2: One or more fields are blank

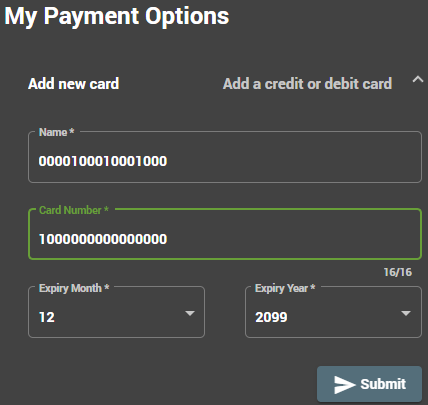
EC3: Card number is a 16-digit number between 1000000000000000 – 9999999999999999

EC4: Card number is a 16-digit number between 0000000000000000 – 0999999999999999

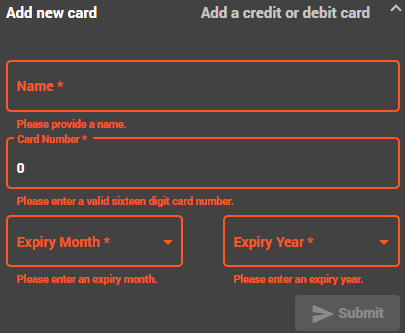
EC5: Card number has 15 or fewer digits

EC6: Card number has 17 or more digits

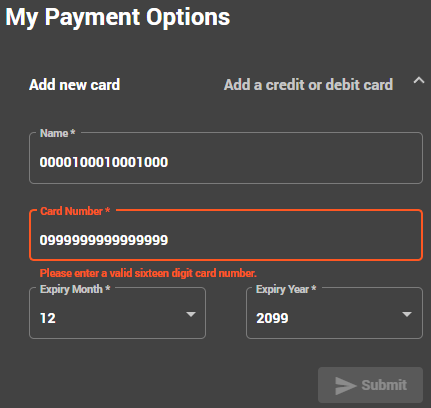
|  |  |  |
| --- | --- | --- |
| **Input Condition** | **Valid Equivalence Classes** | **Invalid Equivalence Classes** |
| The form is filled out | EC1 | EC2 |
| Card Number | EC3 | EC4, EC5, EC6 |



**Valid Case (EC1, EC3)**



**Invalid Case (EC2, EC5)**



**Invalid Case (EC4)**

## **5.7 Digital Wallet (Manual Test)**

Let x = the amount to reload the wallet by

Assumption: x is an integer or floating-point value

*Equivalence Classes*

EC1 = The form is filled out

EC2 = The form is blank

EC3 = {x: 10.00 <= x <= 1000.00}

EC4 = {x: x < 10.00}

EC5 = {x: x > 1000.00}

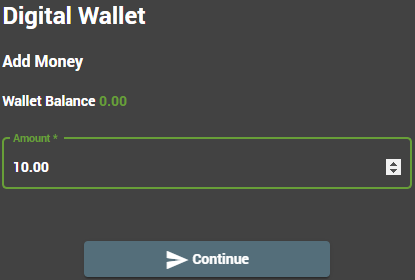
|  |  |  |
| --- | --- | --- |
| **Input Condition** | **Valid Equivalence Classes** | **Invalid Equivalence Classes** |
| The form is filled out | EC1 | EC2 |
| Balance reload amount | EC3 | EC4, EC5 |

*Boundary Values*

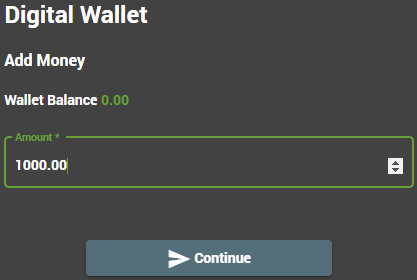
*LB:* 9.99, 10.00, 10.01

*UB:* 999.99, 1000.00, 1000.01

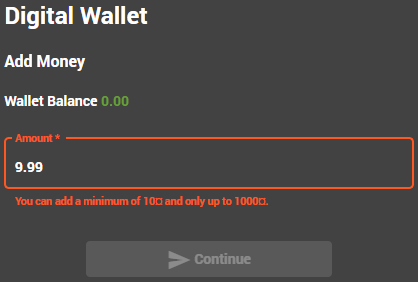
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case Identifier** | **Input Values** | **Valid equivalence classes and bounds covered** | **Invalid equivalence classes and bounds covered** |
| 1 | 10.00 | EC1, EC3(LB) |  |
| 2 | 1000.00 | EC1, EC3(UB) |  |
| 3 | 9.99 | EC1 | EC4(BLB) |
| 4 | 1000.01 | EC1 | EC4(AUB) |
| 5 | {nothing} |  | EC2 |
| 6 | 999.99 | EC1, EC3(ALB) |  |
| 7 | 10.01 | EC1, EC3(BUB) |  |



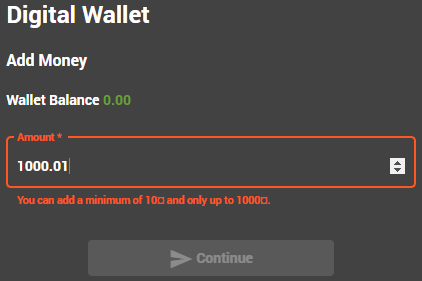
**TC1**



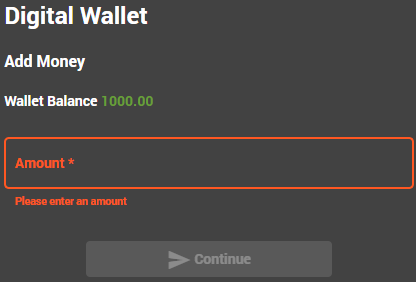
**TC2**



**TC3**



**TC4**



**TC5**

# Report

We grouped our test cases into four different categories:

TestSuite-1 related to test the Login Form.

TestSuite-2 related to test the Registration Form.

TestSuite-3 related to test the Search Bar.

TestSuite-4 related to test the Add to Basket, Modify Basket and Checkout functionality.

(TestSuite-4X) related to test the error case of Digital Wallet during Checkout.

Each TestSuite have a logical collection of test cases. (let me know the names of your test cases to put into this diagram)

TC 3-1 Search 1 - Item name match

TC 3-2 Search 2 - Item description match

TC 3-3 Search 3 - no results

TC 4-1 Add to basket 1 - no restriction

TC 4-2 Add to basket 2 - item limit

TC 4-3 Add to basket 3 - OOS

TC 4-4 Modify basket 4 - no restriction

TC 4-3 Modify basket 5 - limit

TC 4-6 Checkout / Order 6 – credit card

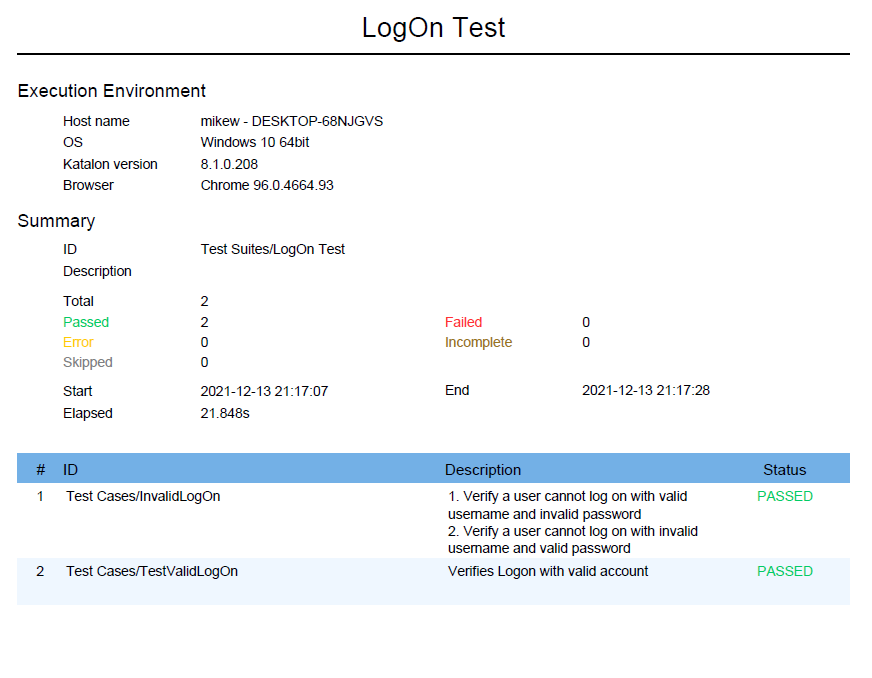
TC 4-X Checkout X – bad wallet balance

Diagram

Description automatically generated

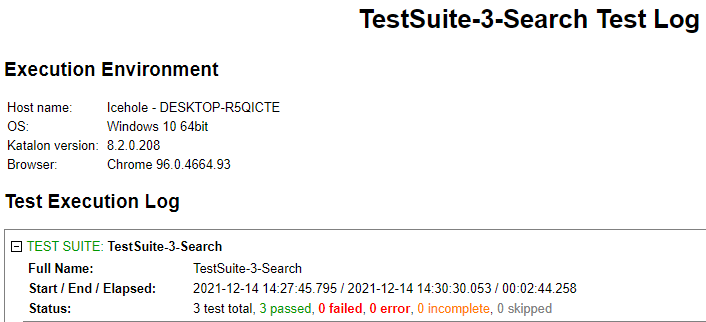
Katalon Studios provides test reports on the results of each test suite and related test cases. The results of the test suites from Katalon Studios are provided below:

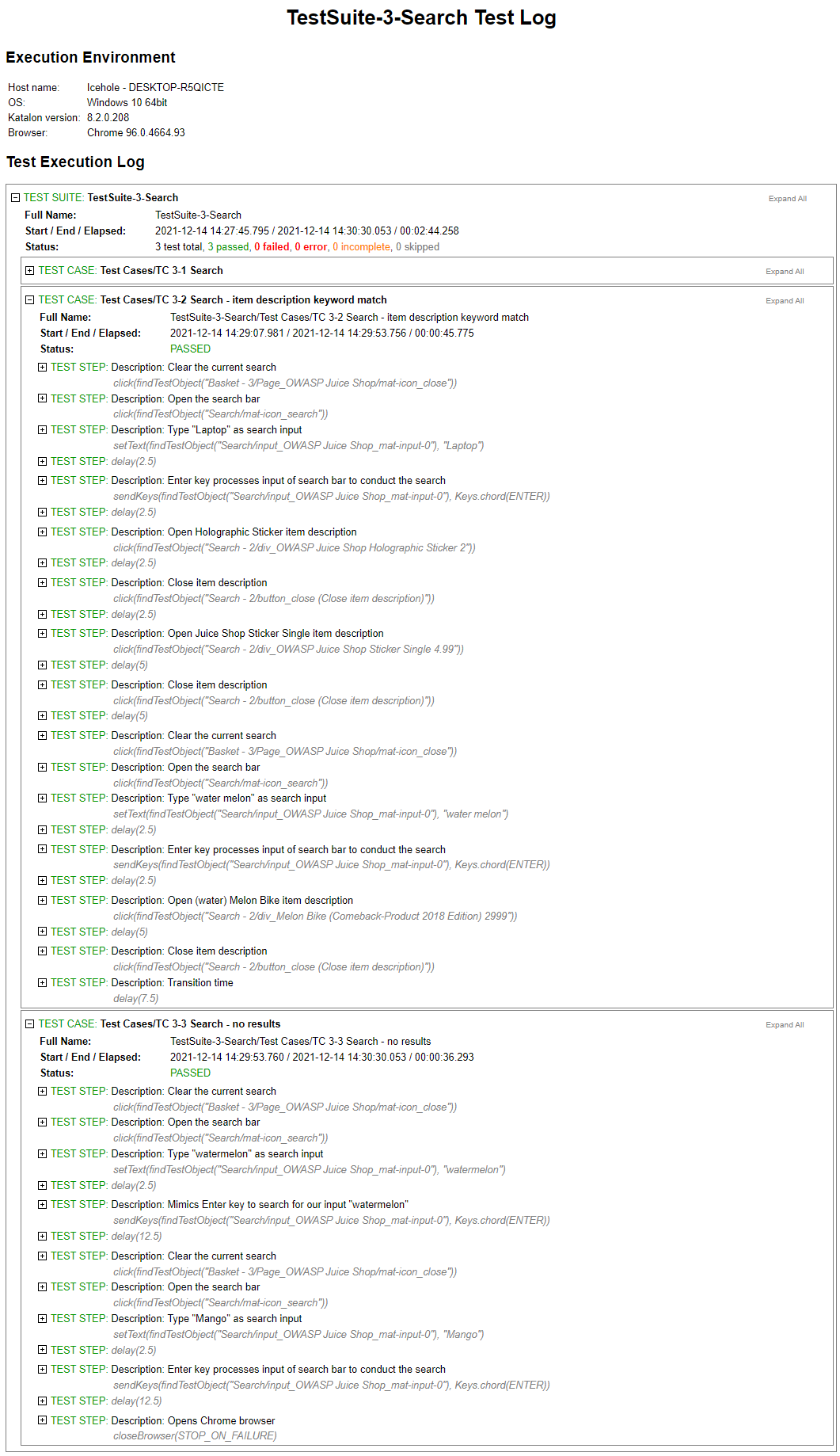
Module Test 1



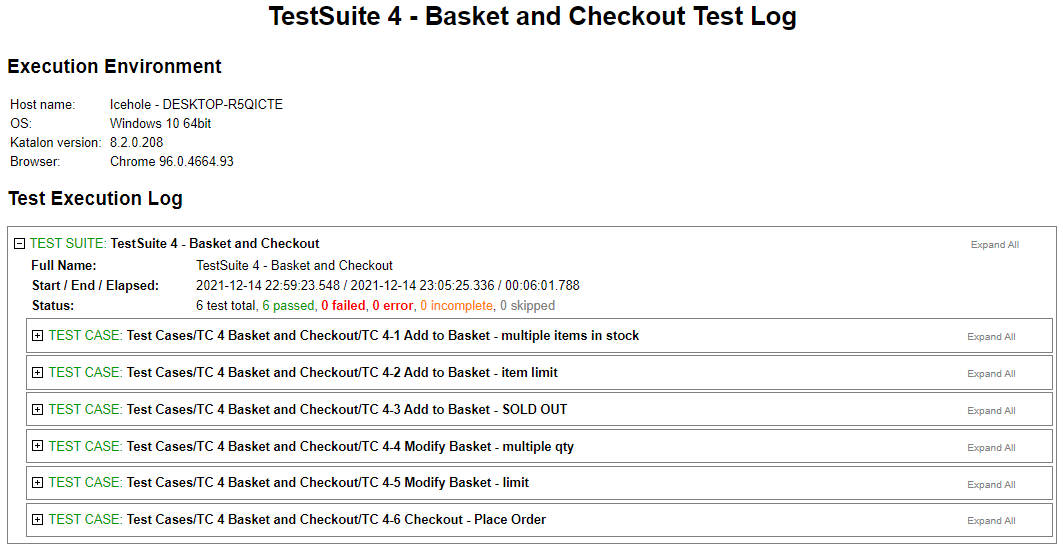
Timeline

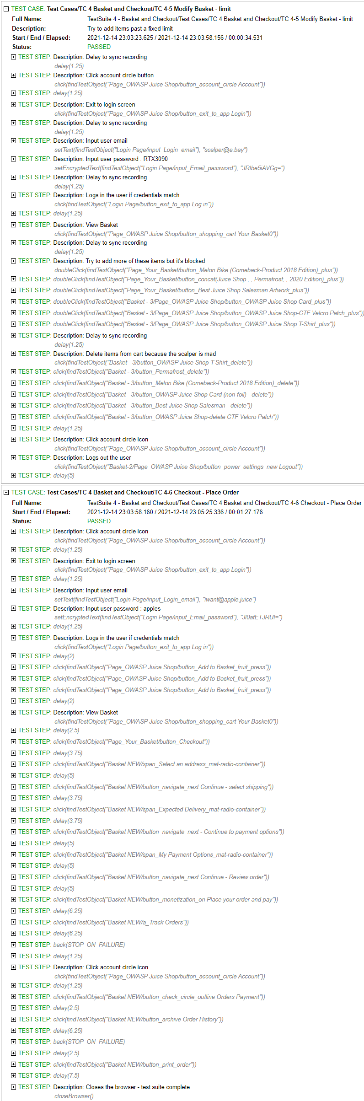
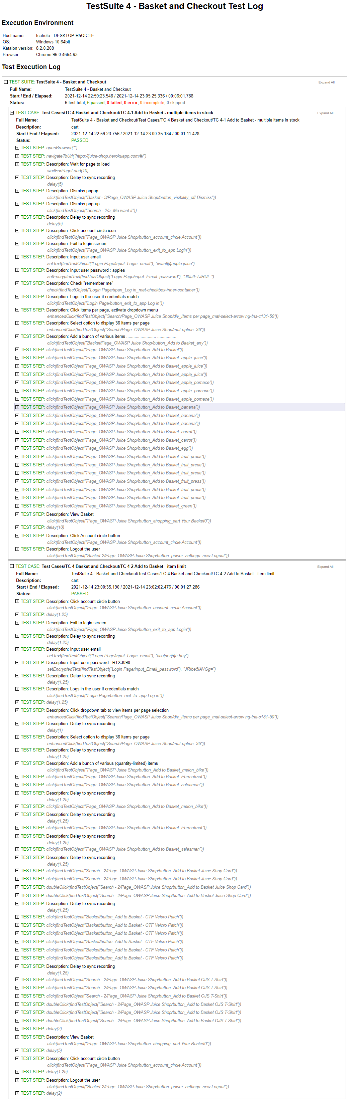
Description automatically generated with low confidence

Full Log

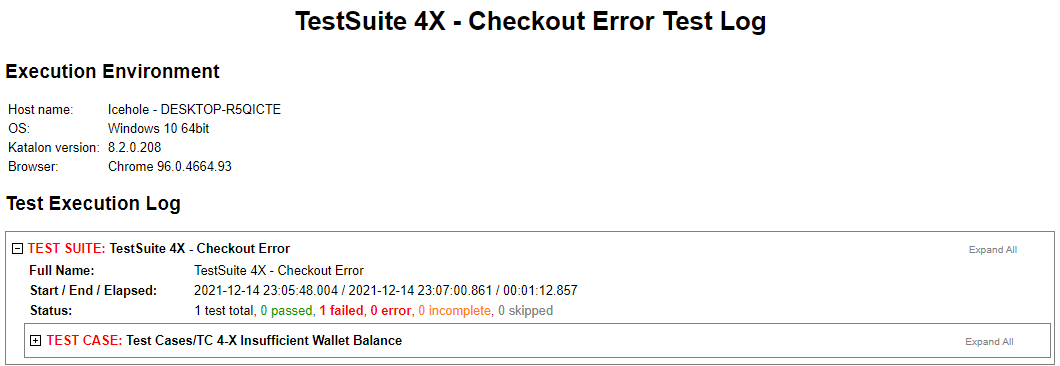


**Success Cases**

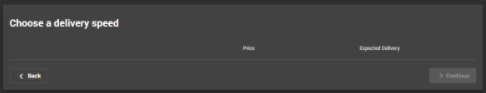
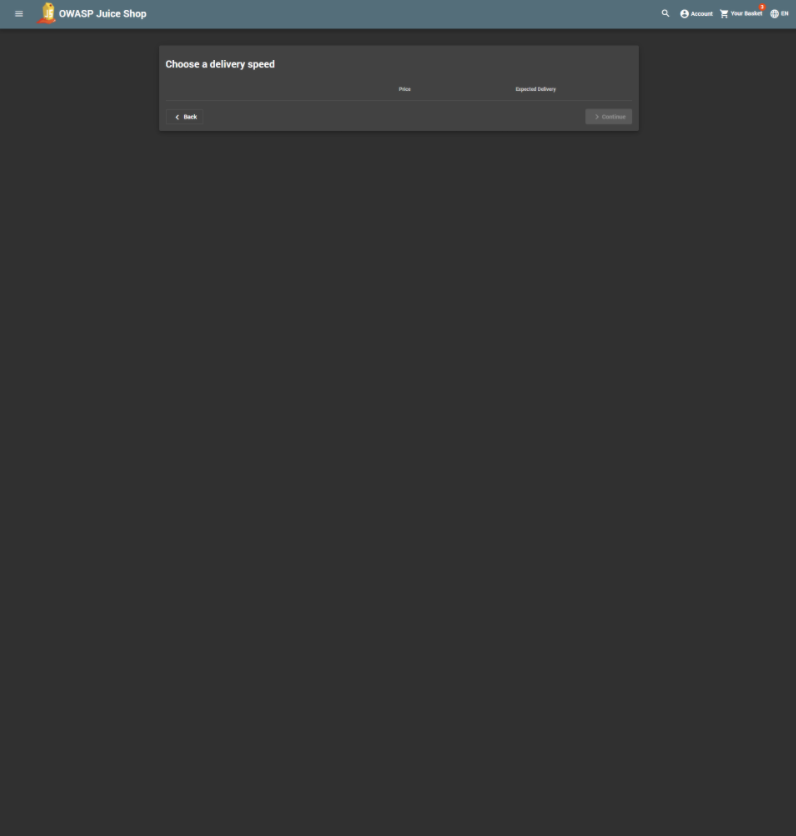
Full Log



**Failure Case 4X**



**Strange bug encountered while recording Test Suite 4X (NO delivery options)**



# Roles and Responsibilities

|  |  |
| --- | --- |
| Roles | Responsibilities |
| Test Leader: | * Design the testing approach, procedures and techniques * Identify, evaluate and choose the automated testing software * Design test documents templates * Resolve the issues in the test groups * Assign roles and provide schedule to testers |

|  |  |
| --- | --- |
| Roles | Responsibilities |
| Testers Group: | Michael Lum – Hoda Dehghanisanij – Wayne Lin |
| Tester 1: Michael | * Gather the test requirements for module 1 * Review the project documents to understand the workflow * Create the test documents test data and test cases for module 1 * Set up and verify the test environment * Automate the tests for module 1 * Participate in the review meetings * Prepare the test summary report for module 1 |
| Tester 2: Hoda | * Gather the test requirements for module 2 * Review the project documents to understand the workflow * Create the test documents test data and test cases for module 2 * Set up and verify the test environment * Automate the tests for module 2 * Participate in the review meetings * Provide the learning experience during the project and share it * Prepare the test summary report for module 2 |
| Tester 3: Wayne | * Supply the automated test materials for modules 3 and 4 * Review the project documents to understand the workflow * Create and verify test suites and test cases for modules 3 and 4 * Record and document automated testing on modules 3 and 4 * Perform manual testing on module 5, checkout and digital wallet * Participate in review meetings * Prepare the test summary report for modules 3, 4 and 5 |

# Resources

1. <https://en.wikipedia.org/wiki/Katalon_Studio>
2. <https://www.katalon.com/resources-center/blog/katalon-studio-vs-selenium-based-open-source-frameworks/>