

Test flow document



Table of Contents

[1 Introduction 3](#_Toc39160721)

[2 Web UI Project Details 3](#_Toc39160722)

[2.1 TinEye.com 4](#_Toc39160723)

[2.2 Test Process Details 4](#_Toc39160724)

[2.3 Architecture 5](#_Toc39160725)

[2.4 Class Diagram 6](#_Toc39160726)

[3 Test Case 1 7](#_Toc39160727)

[3.1 Summary 7](#_Toc39160728)

[3.2 Test Case Details 7](#_Toc39160729)

[3.3 Test Flow Diagram 7](#_Toc39160730)

[3.4 Test Flow Mapping on Application 7](#_Toc39160731)

[3.5 Automation Code Explanation 8](#_Toc39160732)

[3.6 Results Summary 8](#_Toc39160733)

[4 Test Case 2 9](#_Toc39160734)

[4.1 Summary 9](#_Toc39160735)

[4.2 Test Case Details 9](#_Toc39160736)

[4.3 Test Flow Diagram 9](#_Toc39160737)

[4.4 Test Flow Mapping on Application 9](#_Toc39160738)

[4.5 Automation Code Explanation 10](#_Toc39160739)

[4.6 Results Summary 10](#_Toc39160740)

[5 Test Case 3 11](#_Toc39160741)

[5.1 Summary 11](#_Toc39160742)

[5.2 Test Case Details 11](#_Toc39160743)

[5.3 Test Flow Diagram 11](#_Toc39160744)

[5.4 Test Flow Mapping on Application 11](#_Toc39160745)

[5.5 Automation Code Explanation 12](#_Toc39160746)

[5.6 Results Summary 12](#_Toc39160747)

[6 Test Case 4 13](#_Toc39160748)

[6.1 Summary 13](#_Toc39160749)

[6.2 Test Case Details 13](#_Toc39160750)

[6.3 Test Flow Diagram 13](#_Toc39160751)

[6.4 Test Flow Mapping on Application 13](#_Toc39160752)

[6.5 Automation Code Explanation 14](#_Toc39160753)

[6.6 Results Summary 14](#_Toc39160754)

[7 Test Case 5 15](#_Toc39160755)

[7.1 Summary 15](#_Toc39160756)

[7.2 Test Case Details 15](#_Toc39160757)

[7.3 Test Flow Diagram 15](#_Toc39160758)

[7.4 Test Flow Mapping on Application 15](#_Toc39160759)

[7.5 Automation Code Explanation 16](#_Toc39160760)

[7.6 Results Summary 16](#_Toc39160761)

# Introduction

Software test professionals often face challenges and complain that, writing test cases or test scripts takes enormous time and results into huge re-work when the specifications change in the rapid development or agile development environment.

In software testing, test cases are the core deliverables of the test artifacts, more like a source code that gets delivered as the end result of a software program. Often, project managers, test managers or internal stake holders advise tester to ignore test documentation in the rapid development environment and rather ask them to continue with only exploratory or random testing to minimize testing efforts, later resulting into several production issues. Avoiding test cases is as good or as bad a not writing the source code at all. Imagine a program without the source code or a PC without hardware!!!

Test flow is one of the solutions to particular situations aimed at reducing gigantic set of test cases that are often descriptive although enormously helpful. The Test flow gives producers, developers, and testers the capability to easily review, analyze, and provide feedback.

In Toptal evaluation project, test flow document will help evaluator to understand the test and process established for automation. Following section will describe overall process at high level and at detail level for different set of audience.

# Web UI Project Details

Automation of Web UI is one of 3 tasks for this evaluation project with following requirements.

|  |
| --- |
| The requirements for the test project are: 1) Write an automated test for a website or a desktop application.  \* You may pick any website or desktop application you wish.  \* You should have at least 5 test cases including these:     \* Search feature    \* Upload a file  \* Add an HTML report of test results  \* Report any bugs you find by writing a bug ticket  \* Please prepare a document with the test flows. The documents have to be clear both to the developer and to someone who is not familiar with the technology.  \* Run tests in a continuous integration tool and optionally in the cloud |

To accomplish this task I have selected <https://tineye.com/> as application under test and automated few of its features in compliance with requirements mentioned above.

## TinEye.com

Since TinEye.com is the website i shall use to demonstrate automation skills for this task, it’s worthwhile to have a little understanding about TinEye core business

TinEye is an image search and recognition company. TinEye is experts in computer vision, pattern recognition, neural networks and machine learning. Company mission is to make images searchable.

TinEye deliver image search and recognition solutions to the industries where searching images is mission critical. Today, TinEye's image recognition is used by millions of people and powers billions of searches across a wide range of industries.

## Test Process Details

To complete this automation task I have selected 5 random features of TinEye.com to automate. Here is the general description of the test scenarios. These test scenarios are selected in a way it complies with requirements, for example there are 2 tests which are related to search, there is one search which is related to file upload.

1. To verify, Log in attempt with invalid User Name and Password is Rejected with error message
2. To verify, if no image is selected and search is performed, app displays error message
3. To verify, Upload image from local storage is allowed
4. To verify, if Image URL is given, app is able to find the image
5. To verify, Landing Page list Required Elements

To describe each test case flow and its associated technical details, following aspects will be covered for each test case.

1. Test Case Details
2. Test Case Flow Diagram
3. Test Case Flow, mapped on Application under test images
4. Automation Code explanation
5. Results Summary

## Architecture

Page object model design pattern is used to automate web under test. Following figure shows how web pages which need to be test is kept separate from test cases, and test cases are kept separate from framework elements.

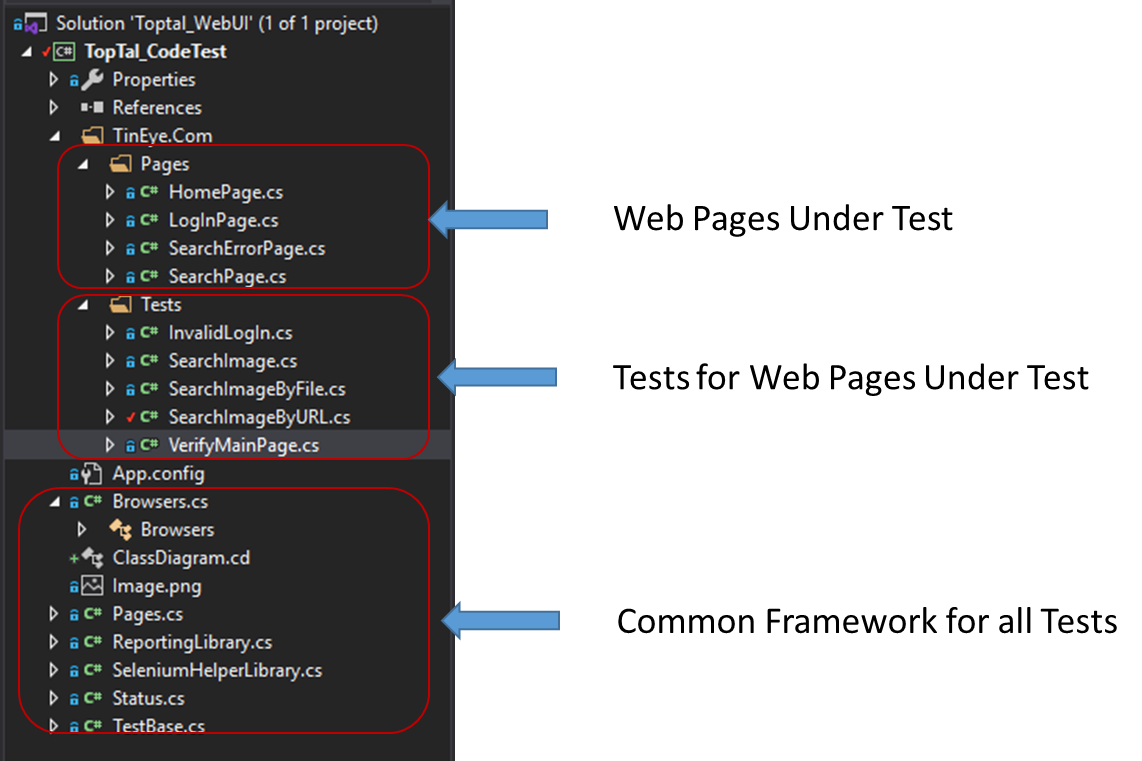


Figure 1: Page object Model of Project

## Class Diagram

Following is the class diagram for complete solution, showing how different classes are interacting

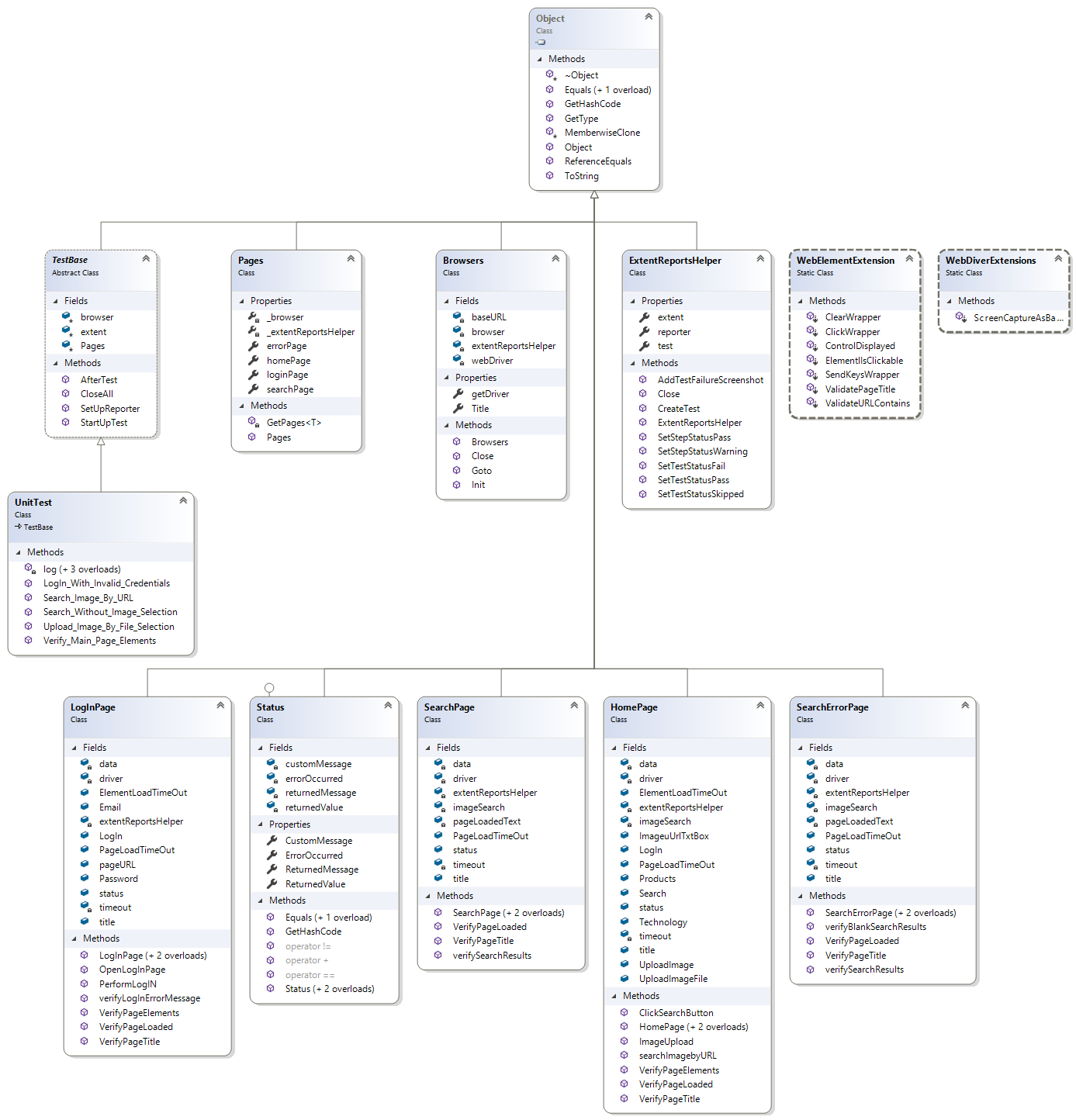


Figure 2: Class Diagram

# Test Case 1

## Summary

To verify, Log in attempt with invalid User Name and Password is Rejected with error message

## Test Case Details

**Test Steps:**

Navigate to Login Page

Verify Page is loaded

Verify Page Title

Verify Page Lists Email, Password and Login Button

Perform Login Operation with invalid Username and Password

Verify page remain same (Page title not changes)

Verify Login attempt failed message is generated

## Test Flow Diagram

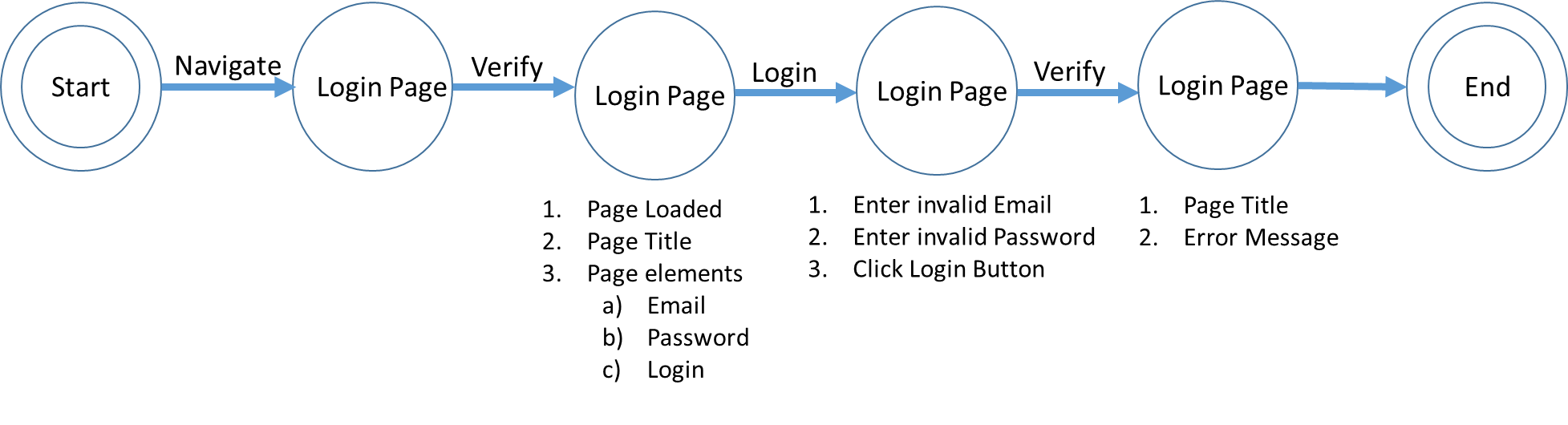


Figure 3: Test Flow Diagram – Test Case 1

## Test Flow Mapping on Application



Figure 4: Test Flow Map on Application – Test Case 1

## Automation Code Explanation

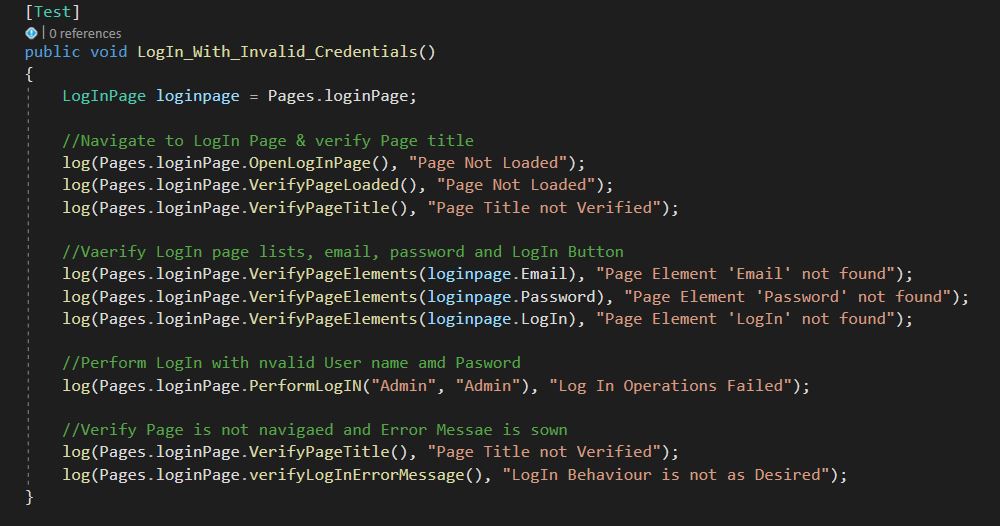


Figure 5: Code view – Test Case 1

## Results Summary

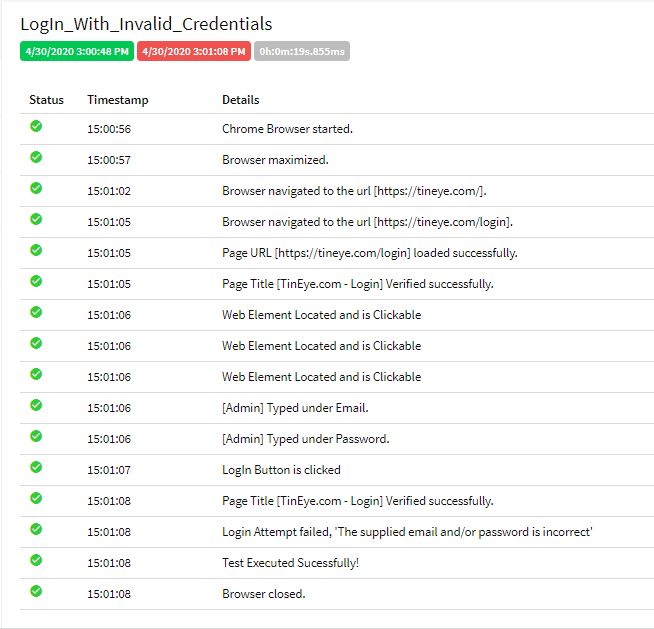


Figure 6: Test Report – Test Case 1

# Test Case 2

## Summary

To verify, if no image is selected and search is performed, app displays error message

## Test Case Details

**Test Steps:**

Navigate to Main Page

Verify Page loaded

Verify Page Title

Click Search Image button, without uploading any image

Verify page is navigated to search page

Verify Error Message is shown

## Test Flow Diagram

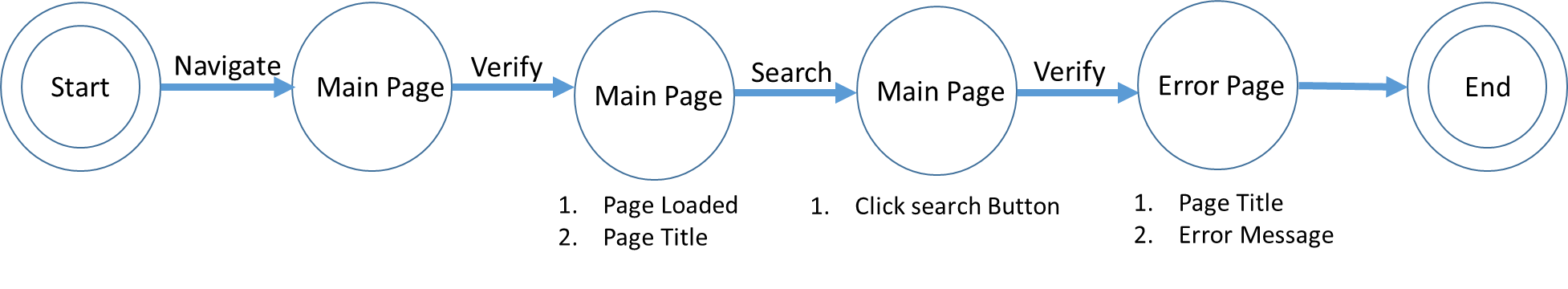


Figure 7: Test Flow Diagram – Test Case 2

## Test Flow Mapping on Application

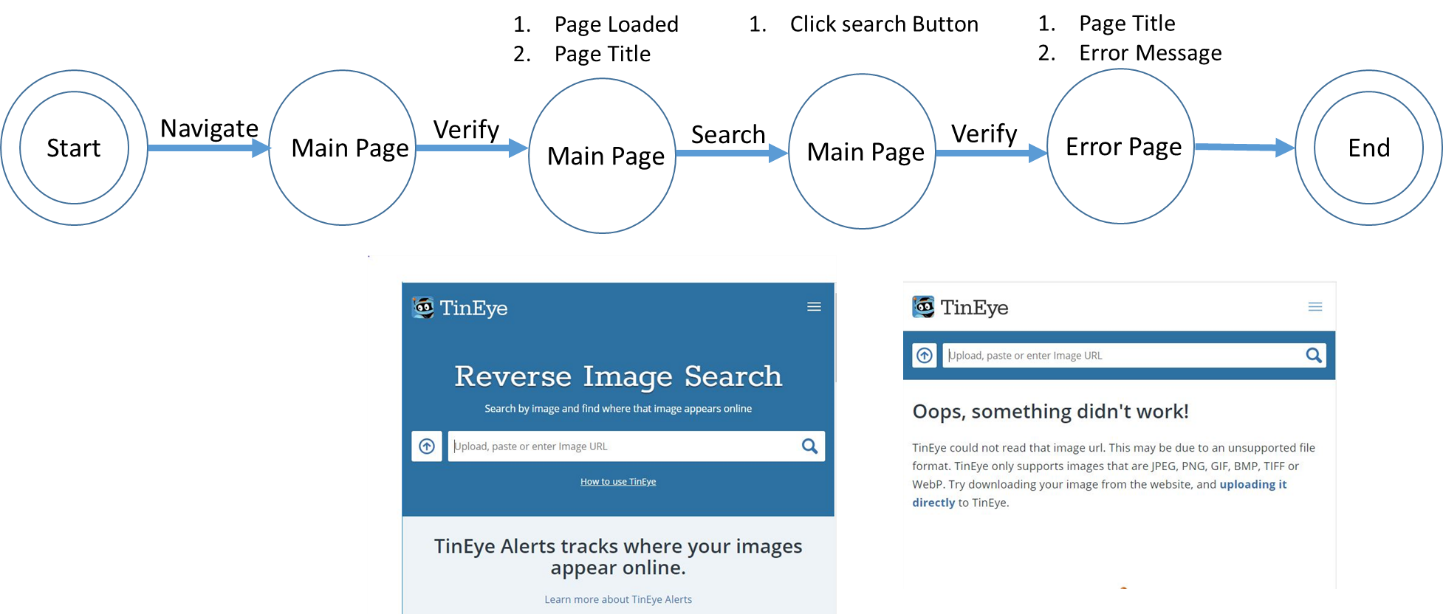


Figure 8: Test Flow Mapping on Application – Test Case 2

## Automation Code Explanation

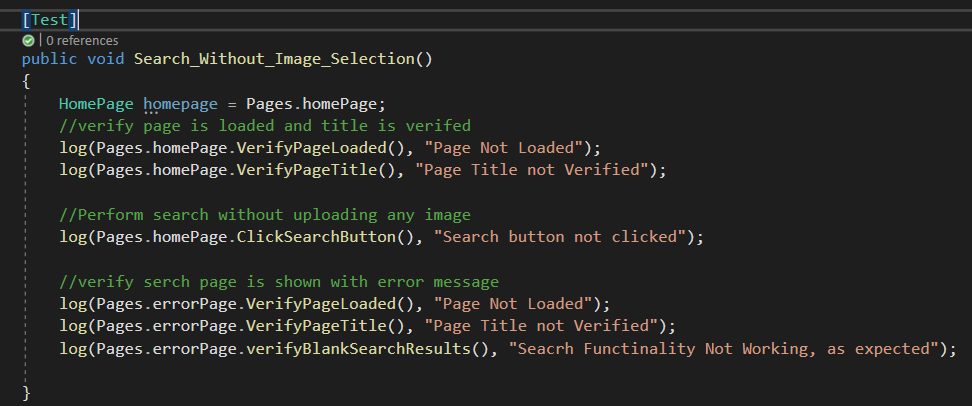


Figure 9: Code View – Test Case 2

## Results Summary

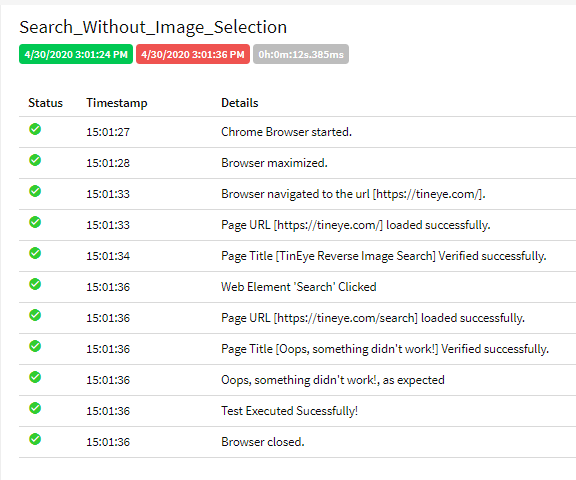


Figure 10: Test Report – Test Case 2

# Test Case 3

## Summary

To verify, Upload image from local storage is allowed

## Test Case Details

**Test Steps:**

Navigate to Main Page

Verify Page loaded

Verify Page Title

Upload a Unique Image

Verify Image is uploaded and Page navigated to Search page

Verify Unique image message is displayed

## Test Flow Diagram

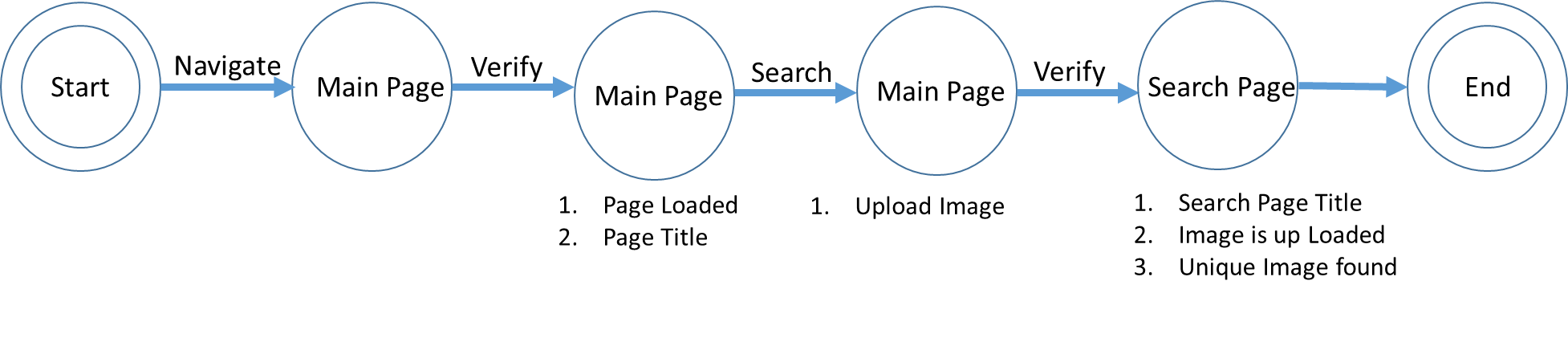
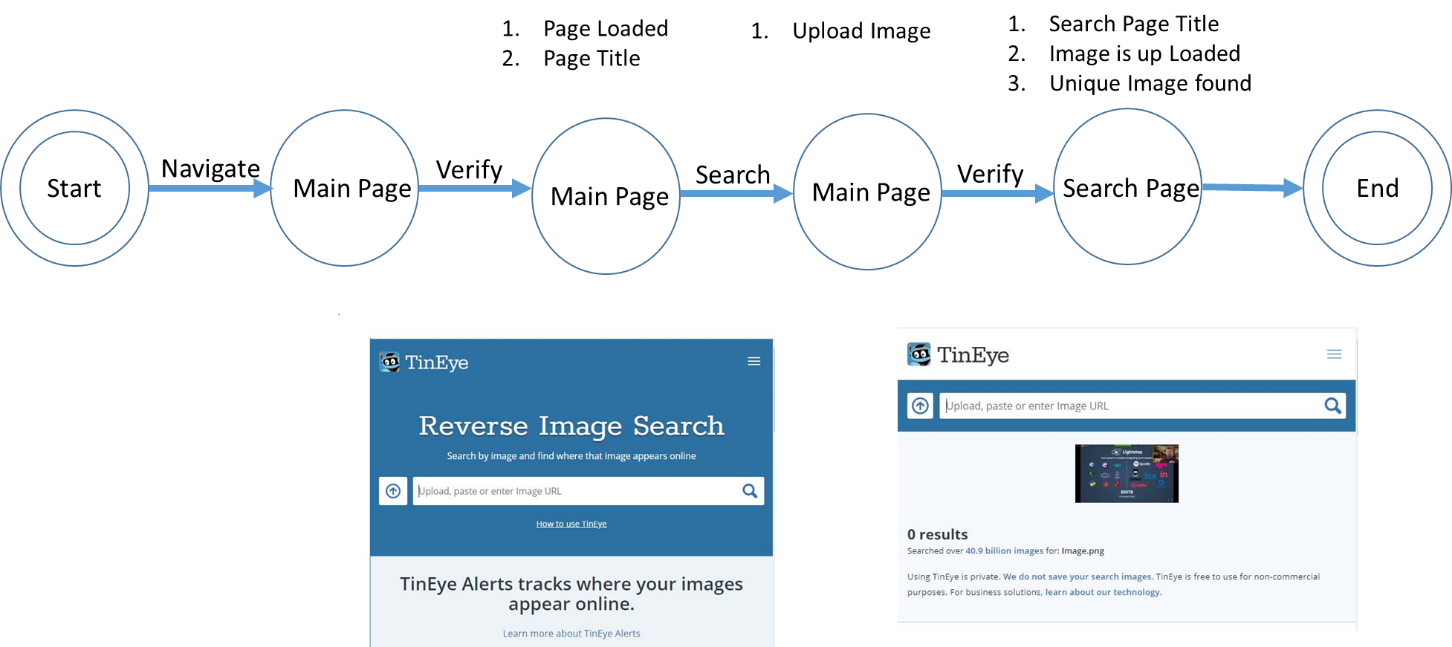


Figure 11: Test Flow Diagram – Test 3

## Test Flow Mapping on Application

 Figure 12: Test Flow Mapping on Application – Test 3

## Automation Code Explanation

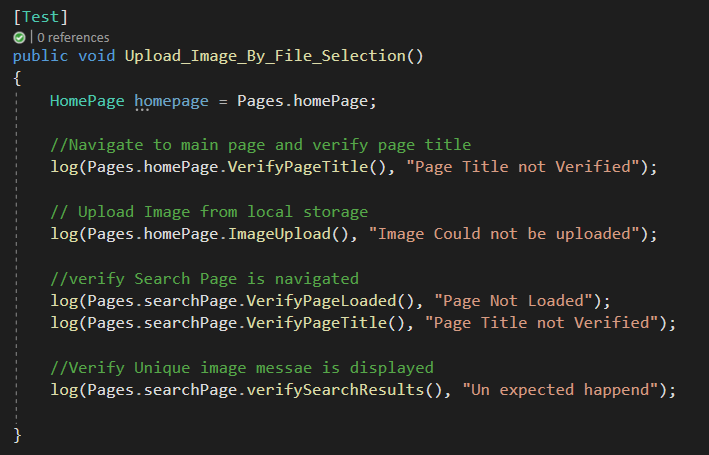


Figure 13: Code View – Test 3

## Results Summary

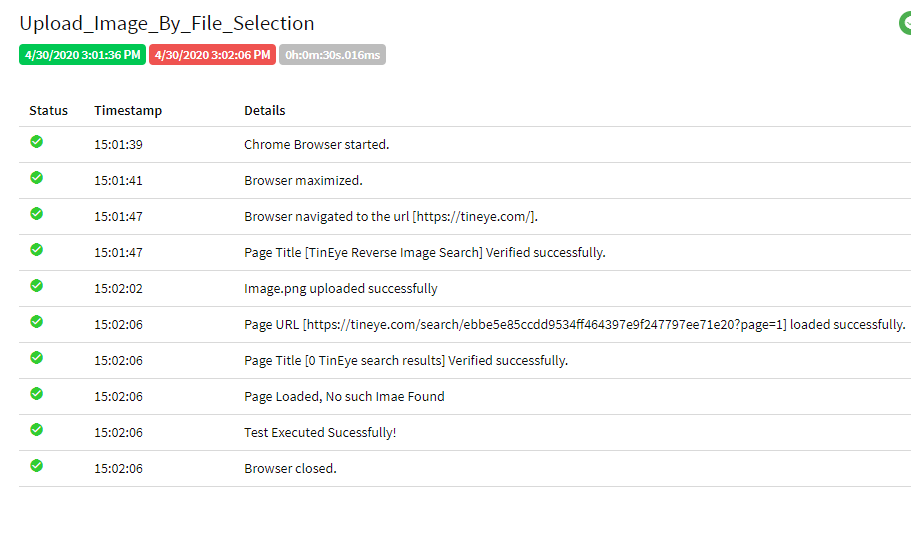


Figure 14: Test Report – Test 3

# Test Case 4

## Summary

To verify, if Image URL is given, app is able to find the image

## Test Case Details

**Test Steps:**

Navigate to Main Page

Verify Page loaded

Verify Page Title

Insert Image URL in search Box

Verify Page is navigated to search page

Verify image is searched without any error

## Test Flow Diagram

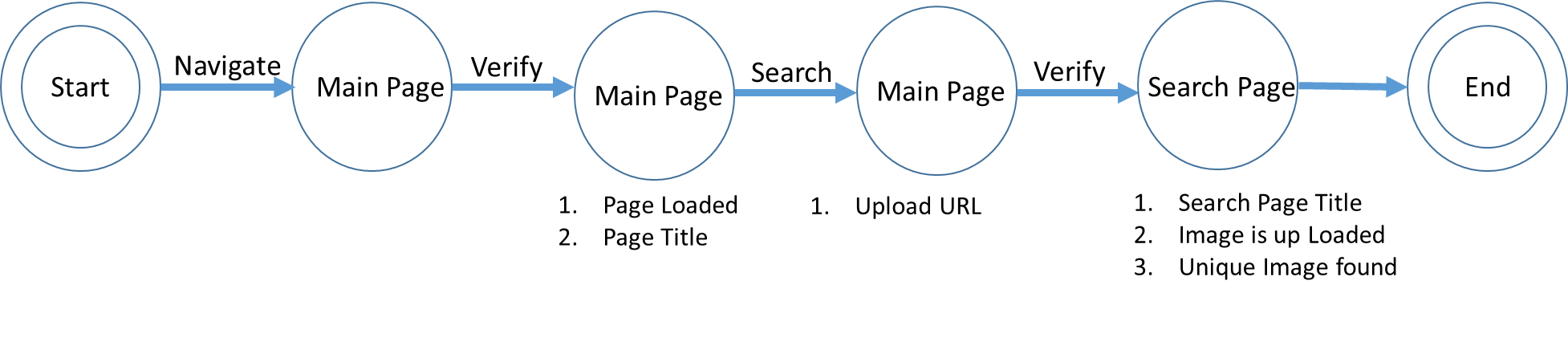


Figure 15: Test Flow Diagram – Test 4

## Test Flow Mapping on Application

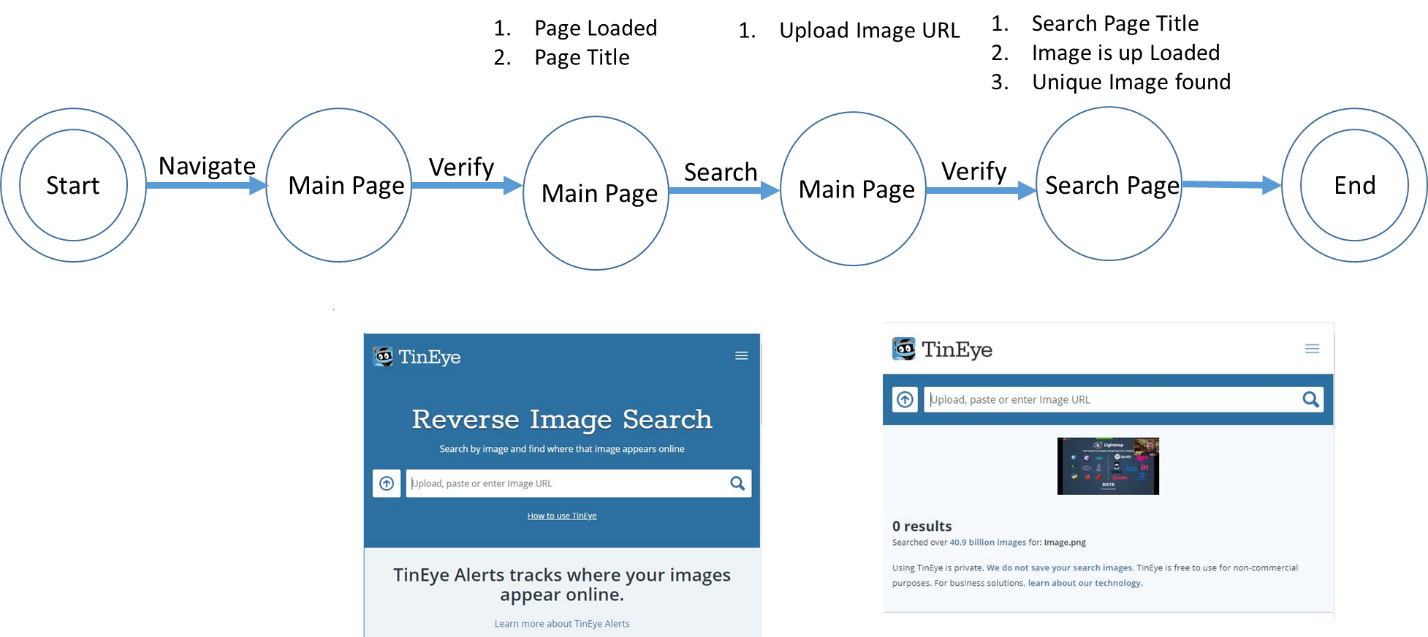


Figure 16: Test Flow Mapping on Application – Test 4

## Automation Code Explanation

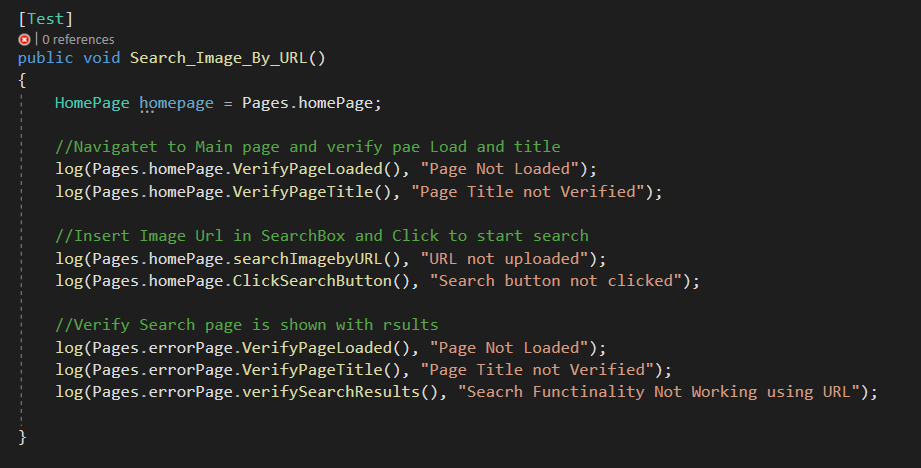


Figure 17: Code View – Test 4

## Results Summary

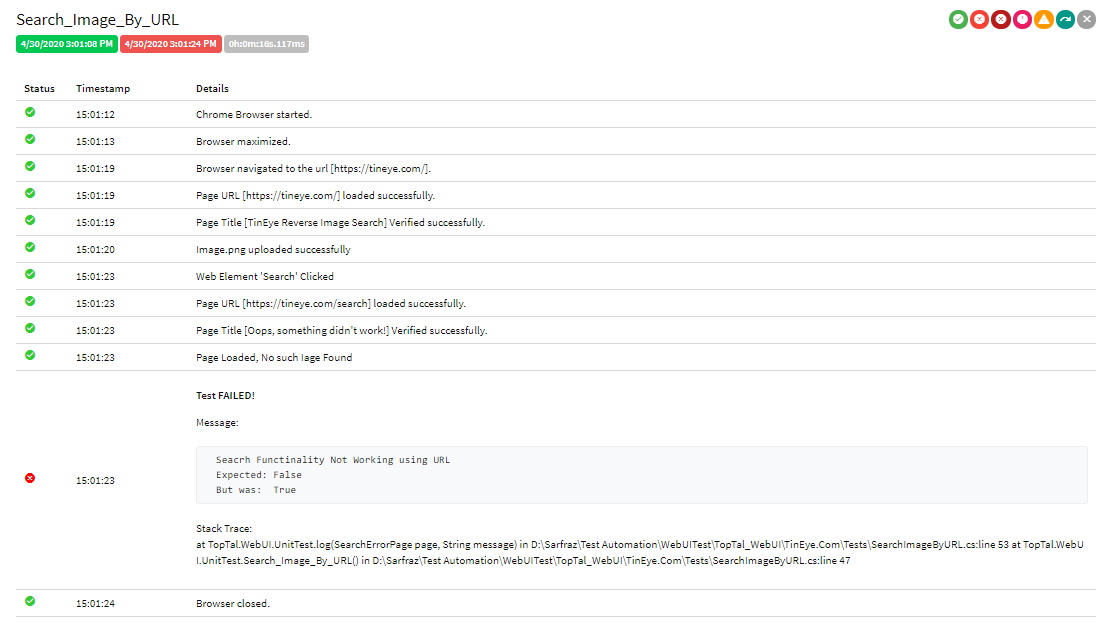


Figure 18: Test Report – Test 4

# Test Case 5

## Summary

To verify, Landing Page List Required Elements

## Test Case Details

**Test Steps:**

Navigate to Main Page

Verify Page loaded

Verify Page Title

Verify Page Lists Technology, Products and Login option

Verify to Search image, Page lists Textbox for Image Link

Verify to Search image, a button is provided to upload file

Verify User can click on Search button to start search

## Test Flow Diagram

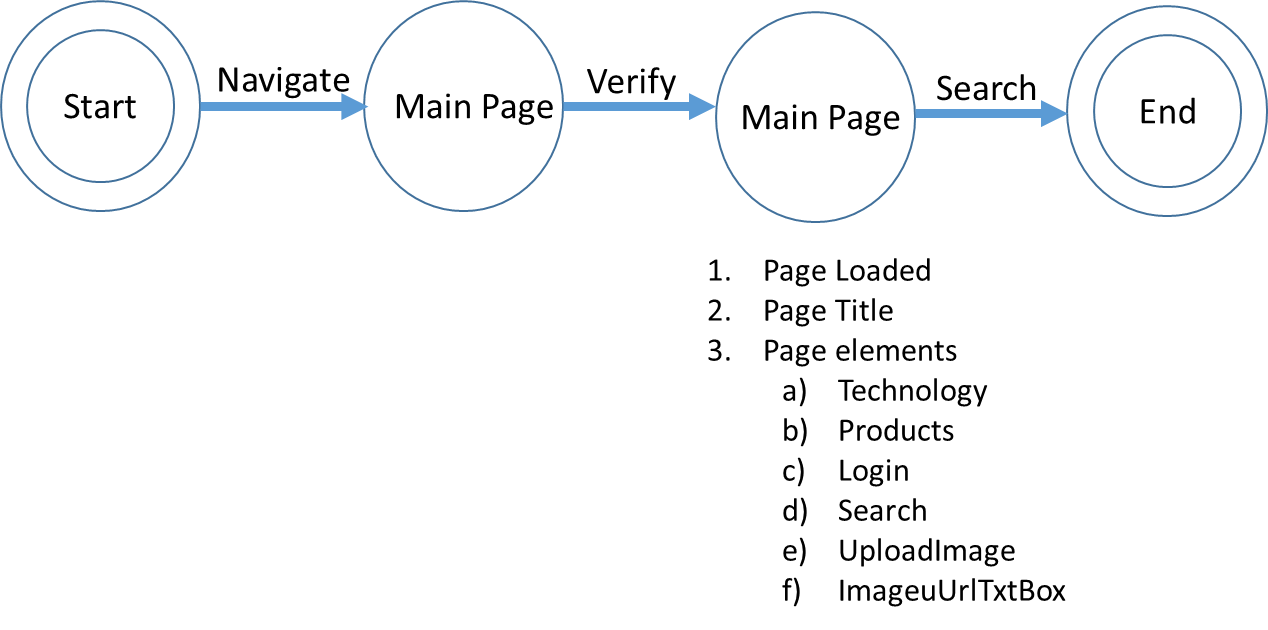


Figure 19: Test Flow Diagram – Test 5

## Test Flow Mapping on Application

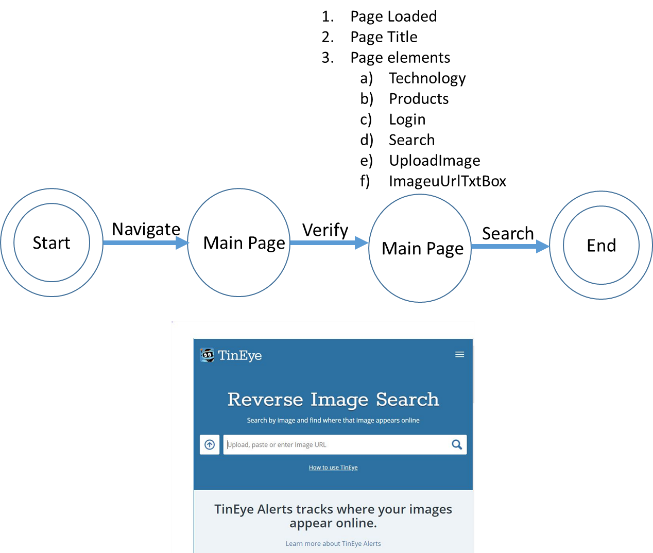


Figure 20: Test Flow Mapping on application– Test 5

## Automation Code Explanation

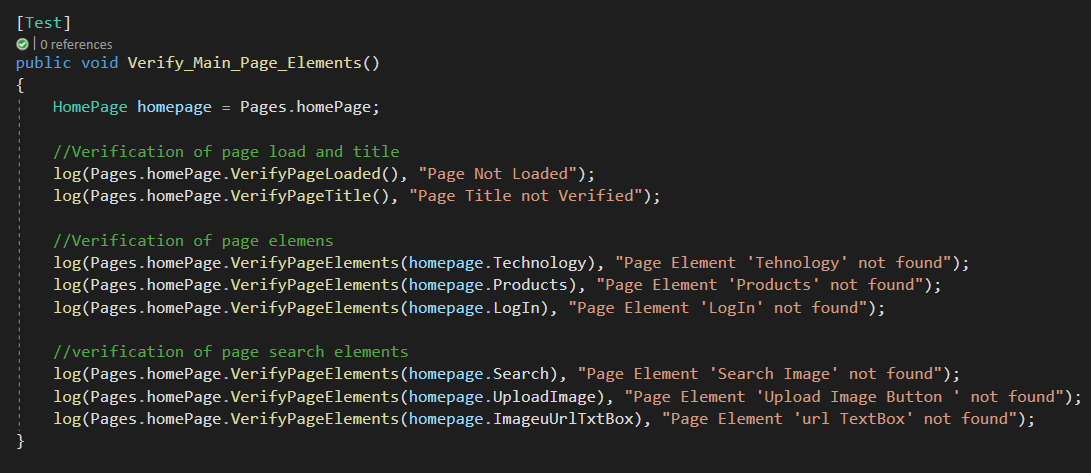


Figure 21: Code View– Test 5

## Results Summary

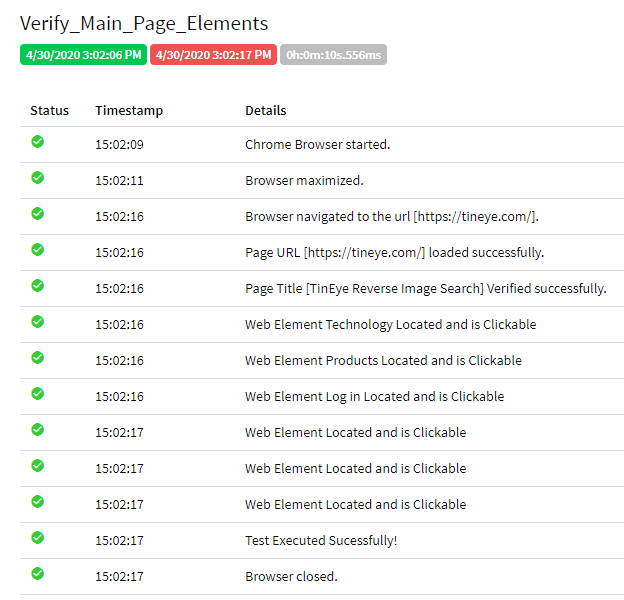


Figure 22: Test Result– Test 5