TEST REPORT

YouTube Platform

Sanduni Vihara

Sandunivihara228@gmail.com

Contents

1. Introduction

With the rapid growth of web-based platforms, ensuring both functional accuracy and system performance has become critical. YouTube, as one of the largest video-sharing platforms in the world, serves millions of users daily, making it an ideal candidate for testing automation and performance evaluation.

Functional testing ensures that the application behaves correctly from the user's perspective. Manual testing of dynamic websites like YouTube can be time-consuming and prone to errors due to repetitive tasks and complex user interactions. Selenium WebDriver addresses these challenges by providing a robust framework for automating browser-based tasks. Through Selenium, testers can simulate real user interactions, such as searching for videos, playing content, and interacting with UI elements, while capturing evidence through screenshots for verification and reporting.

Performance testing evaluates how a system performs under varying loads and stress conditions. While Selenium verifies functionality, it does not measure backend performance or server response times. Apache JMeter fills this gap by allowing testers to simulate multiple concurrent users, measure response times, and analyze system throughput. This information is essential to identify bottlenecks, optimize performance, and ensure scalability, especially for applications with high traffic like YouTube.

By combining Selenium and JMeter testing, this project highlights a comprehensive testing strategy. Functional correctness is validated alongside performance reliability, providing a complete picture of the system's quality. This approach is significant for developers, testers, and stakeholders as it ensures that web applications are both user-friendly and capable of handling high demand efficiently.

2. Test Plan

2.1. Objectives

The primary objective of this project is to automate basic interactions on the YouTube platform using Selenium WebDriver with Java and Jmeter, ensuring efficient and reliable testing of web functionalities. The specific objectives are:

1. Automate YouTube Navigation

    - ✓ To programmatically open the YouTube homepage and verify that it loads correctly, demonstrating automated browser control.

2. Automate Video Search Functionality

    - ✓ To simulate a user search by entering a query in the YouTube search bar.
    - ✓ To validate that relevant video search results are displayed, ensuring the search feature works as expected.

3. Automate Video Playback

    - ✓ To select and play the first video from the search results automatically.
    - ✓ To interact with video controls, such as play and pause, demonstrating control over dynamic web elements.

4. Capture Screenshots at Key Steps

    - ✓ To capture screenshots after each significant action (homepage load, search results, video page, and video playback).
    - ✓ To provide visual evidence for test execution, aiding in debugging and reporting.

5. Implement Dynamic Waiting for Elements

    - ✓ To use explicit waits for elements to become visible and clickable, ensuring stable test execution despite asynchronous content loading.

6. Enhance Test Reliability and Accuracy

    - ✓ To handle dynamically loaded web elements and YouTube's interactive interface effectively.
    - ✓ To ensure the automated tests are robust, repeatable, and can be used as a reference for future web automation tasks.

2.2. Aim & Scope

Aim

The aim of this project is to automate key functionalities of the YouTube platform using Selenium WebDriver with Java and performance using Jmeter, focusing on user interactions such as navigation, video search, and playback. The project seeks to demonstrate how web automation can efficiently perform tasks that would otherwise require manual testing, while also providing visual evidence through screenshots at each step.

Scope

The scope of this project includes:

1. Web Automation

    - ✓ Opening the YouTube homepage and verifying its successful load.
    - ✓ Searching for videos based on user-defined keywords.
    - ✓ Selecting and playing videos automatically.

2. Testing and Validation

    - ✓ Capturing screenshots at each major step to validate execution.
    - ✓ Ensuring elements are intractable using dynamic waits, which increases reliability.

3. Demonstration of Automation Skills

    - ✓ Applying Selenium WebDriver and Java programming to real-world web applications.
    - ✓ Interacting with dynamic web elements, such as video controls and search results.
    - ✓ To check the performance like load handling, response time and scalability.

4. Limitations

    - ✓ Login-protected features (like subscribing, commenting, or liking) are not included, as Google authentication adds complexity due to security restrictions.
    - ✓ The project focuses on automating core public features accessible without user credentials.

3. Testing Methodology

The testing methodology for this project integrates both functional testing using Selenium WebDriver and performance testing using Apache JMeter to ensure the YouTube application works correctly and efficiently under expected conditions.

3.1. Functional Testing with Selenium

Functional testing focuses on verifying that the YouTube platform behaves as expected from a user's perspective. The steps included:

3.1.1. Test Environment Setup

- Installed Java JDK and configured Selenium WebDriver with ChromeDriver.
- Prepared an IDE (IntelliJ IDEA/Eclipse) and configured project dependencies (Selenium and WebDriver libraries).

3.1.2. Test Case Design

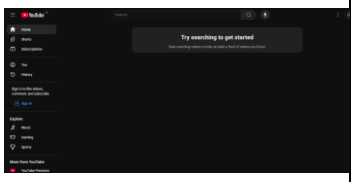
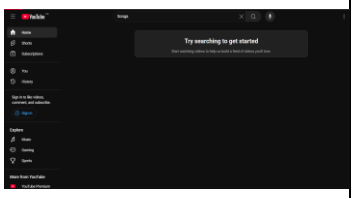
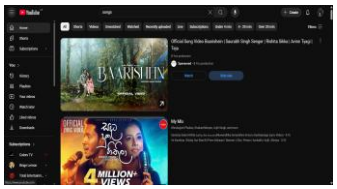
3.1.3. Test Execution

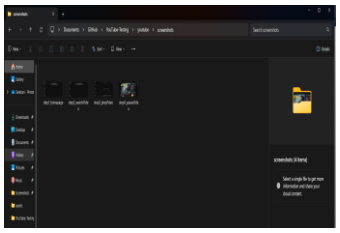
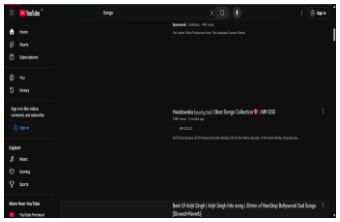
- Used WebDriverWait to handle dynamic content and ensure elements are interactable.
- Executed the automated tests sequentially for each step and validated expected behavior.

3.1.4. Result Verification

- Screenshots confirmed that each step completed successfully.
- Functional correctness was validated by observing video playback and UI interactions.

3.1.2.1. Test Cases

Test case Id	YT001					
Unit						
Assumption						
Test Case No:	Description	Input	Expected Output	Actual Output	Pass or Fail	Screenshots
001	Open YouTube home page	URL: www.youtube.com	YouTube homepage loads; title contains "YouTube"	YouTube homepage loads; title contains "YouTube"	Pass	
002	Search video	Keyword: "songs"	In search box display "songs"	In search box display "songs"	Pass	
003	Display videos based on the search Input	Enter "Submit" Button	Relevant video search results are displayed	Relevant video search results are displayed	Pass	
004	Click Third Video	Third video from search results	Video page loads and starts playing	Video page loads and starts playing	Pass	

005	Pause Video	Pause button click	Video pause accordingly	Video pause accordingly	Pass	
006	Take Screenshots	Steps of test execution (homepage, search, video play, pause)	Screenshots are saved with correct filenames	Screenshots are saved with correct filenames	Pass	
007	Verify Page Title	Page titles at homepage, search results, video page	Titles match expected content for each step	Titles match expected content for each step	Pass	

3.2. Performance Testing with Jmeter

Performance testing evaluates how the system behaves under varying load conditions. The steps included:

3.2.1. Test Environment Setup

- Installed Apache JMeter and configured it for HTTP request testing.

3.2.2. Test Plan Design

- Simulate Multiple Users**:** Created threads to simulate multiple concurrent searches.
- Measure Response Times**:** Collected metrics for page loading and video request response.
- Analyze Throughput: Checked how many requests the system could handle per second.

3.2.3. Test Execution

- Executed JMeter test plans in a controlled environment to avoid impacting public YouTube servers.
- Recorded response times and analyzed performance under different loads.

3.2.4. Result Analysis

- Identified potential bottlenecks in search response and page load time.
- Provided insights into system performance and scalability.

6. Conclusion

This project demonstrates a comprehensive approach to web application testing by combining Selenium WebDriver for functional testing and Apache JMeter for performance testing**.** Through Selenium, critical user interactions on the YouTube platform such as opening the homepage, searching for videos, playing content, and interacting with video controls were successfully automated and validated. Screenshots taken at each step provided clear visual evidence of test execution and results.

Complementing functional testing, JMeter enabled the assessment of performance aspects, such as response times and system behavior under simulated load conditions. Although direct load testing on public websites like YouTube is restricted, the methodology illustrates how performance testing can be applied to web applications in controlled environments.

Overall, this project highlights the importance of integrating functional and performance testing to ensure that web applications are both reliable and scalable. The combined use of Selenium and JMeter provides a powerful framework for automated testing, reduces manual effort, improves accuracy, and offers insights into both the user experience and system performance. This approach can be extended to other web applications, making it a valuable strategy for ensuring high-quality software delivery.