جامعة بيرزيت

BIRZEIT UNIVERSITY

Software Testing And Quality Assurance

Comp438

Prepared by :

Asil Al Atrash	1211325
Hanan Murrar	1200201
Ibraheem Abu Hijleh	1203065

Instruction: Dr.Faisal Shehadeh

Section: 2

Table Of Contents

Introduction

This report includes the comprehensive quality assurance process applied to the YouTube website, selected as the system under test due to its extensive features, functionality, and widespread usage.

The purpose of this project is to ensure the quality, reliability, and performance of the YouTube website through systematic testing.

The testing process covers requirements analysis, test planning, test case development, execution, automation, and performance testing, the project aims to enhance user experience and verify the system capability to handle diverse scenarios effectively.

Additionally, this report highlights the tools, techniques, and methodologies utilized throughout the testing process and results.

Requirements Analysis

The YouTube website is a versatile system offering a variety of functionalities, analyzing both functional and nonfunctional requirements is essential to ensure good operation and comprehensive testing. [1]

Functional requirements:

Core functionalities:

- **R1: Video playback**
  - Users should be able to play, pause, rewind, and fast forward videos.
  - Support different video resolutions.
- **R2: Search functionality**
  - Users should be able to search for videos using keywords.
  - Provide autocomplete suggestions while typing in the search bar.
- **R3: Account management**
  - Users should be able to sign up, log in, and manage their profiles.
  - Enable features like viewing watch history and managing playlists.
- **R4: Video upload**
  - Users can upload videos in supported formats with specific size and duration limits.
  - Allow users to add titles, descriptions, tags.
- **R5: Comments**
  - Enable replies to comments and like, dislike comments.
- **R6: Interaction**
  - Allow users to like, dislike videos, and share videos.
- **R7: Recommendations**
  - Suggest videos based on the users viewing history and preferences.

- **R8: Live streaming**
  - Users should be able to broadcast live videos.
  - Users should be able to make real time chat during live streaming.
- **R9: Subscription**
  - Users can subscribe to channels to follow content updates.
- **R10: Notifications**
  - Users should be notified of new videos or live streams from subscribed channels.
- **R11: Captions and transcriptions**
  - Provide auto-generated captions in multiple languages.
  - Users should edit the caption of their own videos.
- **R12: Download in offline videos**
  - Users should be able to download videos for offline viewing.

Admin functionalities:

- **R13: Content moderation**
  - Admins should be able to ban or suspend user accounts violating terms of service.
  - Provide tools to flag inappropriate content.
  - Review and manage reported content.
- **R14: Analytics**
  - Provide content creators with video performance metrics like views, likes, engagement.

Non functional requirements:

Performance requirements:

- **R1: Scalability**
  - The system shall be able to handle millions of users worldwide without noticeable decrease in the performance.
- **R2: Caching**
  - frequently accessed videos and search should be cached to reduce loading time.

Security requirements:

- **R3: Fraud detection**
  - The system should contain algorithms to detect fake accounts and bot activity.
  - The system should prevent ad click fraud.
- **R4: Secure payment**
  - The system should provide a secure payment for premium features.

Maintainability requirements:

- **R5: Well documented and standardized database**
    - The system should be prepared for periodic updates.
    - The system should support the routine maintenance without disturbing user experience.

Compatibility requirements:

- **R6: Supports diversity**
    - The system should support various devices, operating systems, and browsers.
    - The system should be compatible with multiple video and audio codecs.

Test Plan

This section contains a structured test plan for the YouTube website.

Introduction:

- **Objective:** the purpose of this test plan is to verify the functionalities and performance of the YouTube website, ensuring it meets functional and non-functional requirements for users.

Scope:

The testing will cover the functional, performance, and usability aspects of the YouTube website, including the login, commenting, video playing, and searching features. This excludes advanced security testing and all non-user-facing features, such as backend analytics.

Test objectives:

- **Functional testing objectives:** verify the valid login functionality, valid comment submission, login with incorrect password, entering mismatched passwords into the reset form, search functionality verification, video playback validation, invalid file type upload, empty comment submission, notifications functionality, share video, verify notification access while offline, view history with invalid conditions.
- **Performance testing objectives:** ensure that YouTube can handle very concurrent users interacting with the system.
- **Usability testing objectives:** evaluate the ease of use of the search bar, video player, and comment sections.

Test strategy:

- **Approach:** the agile testing approach will be used with manual and automated testing.
- **Testing levels:** validate the overall functionality of the YouTube website.
- **Techniques:** the black-box testing for user scenario by Selenium, load testing using JMeter for concurrent users.

Test schedule:

- **Functional testing:** starts on [04/01/2025] and ends on [08/01/2025].
- **Performance testing:** starts on [10/01/2025] and ends on [12/01/2025].
- **Usability testing:** starts on [04/01/2025] and ends on [08/01/2025].

Test resources:

- **Hardware:** we will test the YouTube website on different types of devices to make sure it works well for everyone.
- **Software:** the Selenium for automated testing, JMeter for performance testing.
- **Personnel:** four team members, each assigned specific testing tasks.

Test deliverables:

- **Document structure:** test plan, test cases, bug reports, test automation, performance testing.

Risks and assumptions:

- **Assumptions:** the test environment will be fully set up and stable before testing begins.
- **Risks:** the test environment might not match the production environment exactly, leading to inaccurate results.

Dependencies:

- **External dependencies:** the testing schedule is dependent on the completion of development tasks.

Exit criteria:

- **Exit criteria testing will be considered complete when:** no critical issues are found in the final test, all functional, performance, security, and usability tests have been executed. [we can change this later]

Approvals:

- **The test plan requires approval from:** project manager, QA manager.

Appendices:

- **Glossary:** the test case is a specific set of inputs, execution conditions, and expected results developed to verify a particular feature.
- **References:** https://support.google.com/youtube#topic=9257498

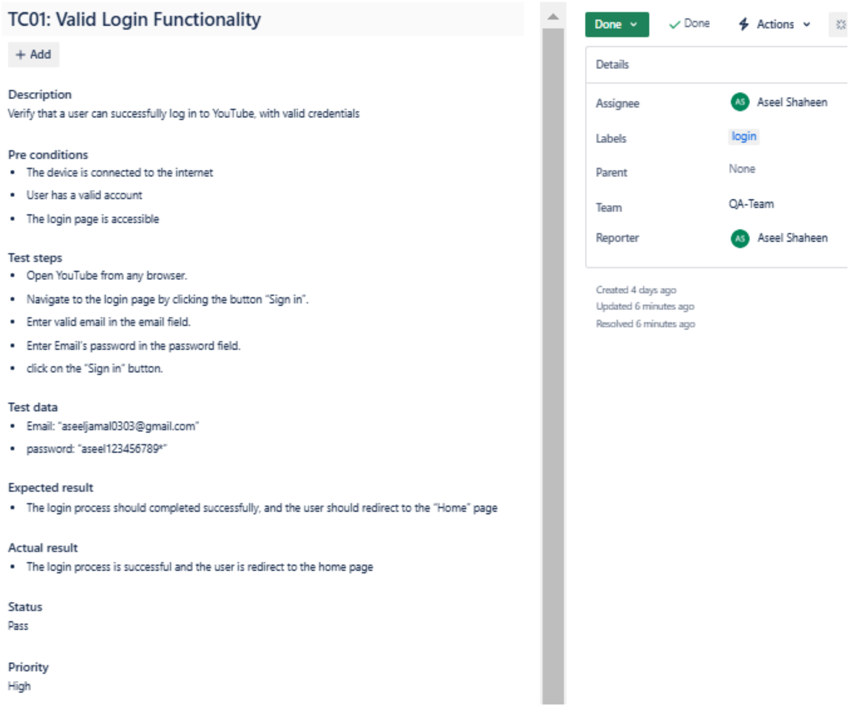
Test Casas & Test Execution

The section contains all the test cases, with each group member create 2 positive test cases and 2 negative test cases, the test cases were developed using well black box testing, to manage and document these test cases efficiently, the team utilized Jira, a specialized testing and project management tool.

Asil

Positive test case:

TC01: Valid login functionality (link in Jira)



TC01: Valid Login Functionality

Description
Verify that a user can successfully log in to YouTube, with valid credentials

Pre conditions
- The device is connected to the internet
- User has a valid account
- The login page is accessible

Test steps
- Open YouTube from any browser.
- Navigate to the login page by clicking the button "Sign in".
- Enter valid email in the email field.
- Enter Email's password in the password field.
- click on the "Sign in" button.

Test data
- Email: "aseeljamal0303@gmail.com"
- password: "aseel123456789*"

Expected result
- The login process should completed successfully, and the user should redirect to the "Home" page

Actual result
- The login process is successful and the user is redirect to the home page

Status
Pass

Priority
High

Details

Assignee	AS Aseel Shaheen
Labels	login
Parent	None
Team	QA-Team
Reporter	AS Aseel Shaheen

Created 4 days ago
Updated 6 minutes ago
Resolved 6 minutes ago

Analyse the result:

Date 05/01/2025

Expected result: the login process should be completed successfully, and the user should be redirected to the home page.

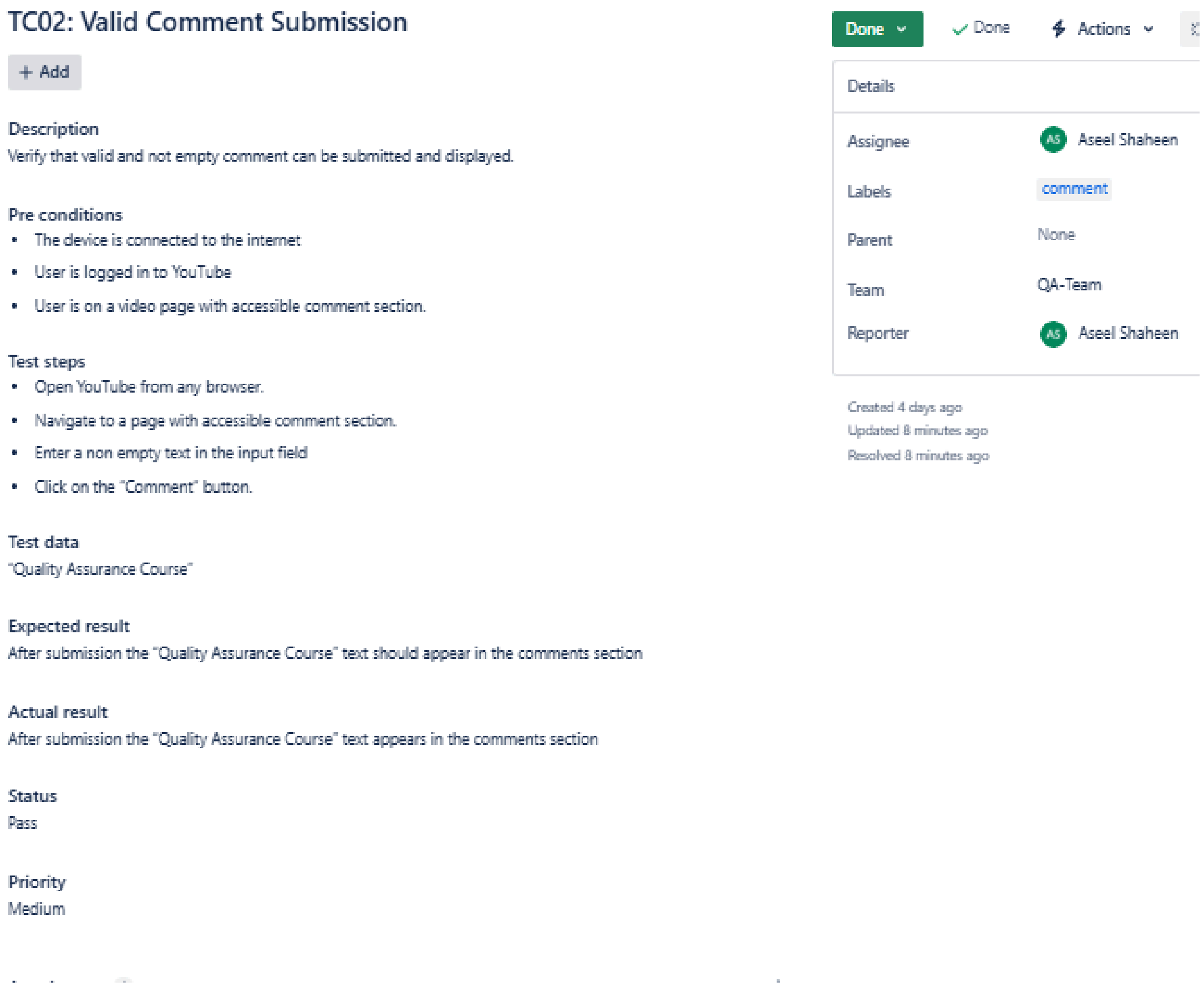
Actual result: after a successful login, the user is redirected to the home page.

Conclusion: it worked as expected by allowing the user to login with the correct email and password.

Detect bugs report:

No bugs detected.

TC02: Valid comment submission (link in Jira)



TC02: Valid Comment Submission

Description

Verify that valid and not empty comment can be submitted and displayed.

Pre conditions

- The device is connected to the internet
- User is logged in to YouTube
- User is on a video page with accessible comment section.

Test steps

- Open YouTube from any browser.
- Navigate to a page with accessible comment section.
- Enter a non empty text in the input field
- Click on the "Comment" button.

Test data

"Quality Assurance Course"

Expected result

After submission the "Quality Assurance Course" text should appear in the comments section

Actual result

After submission the "Quality Assurance Course" text appears in the comments section

Status

Pass

Priority

Medium

Details

Assignee	AS Aseel Shaheen
Labels	comment
Parent	None
Team	QA-Team
Reporter	AS Aseel Shaheen

Created 4 days ago
Updated 8 minutes ago
Resolved 8 minutes ago

Analyse the result:

Date 05/01/2025

Expected result: the comment should appear under the video in the comment section.

Actual result: the comment successfully appeared in the comment section.

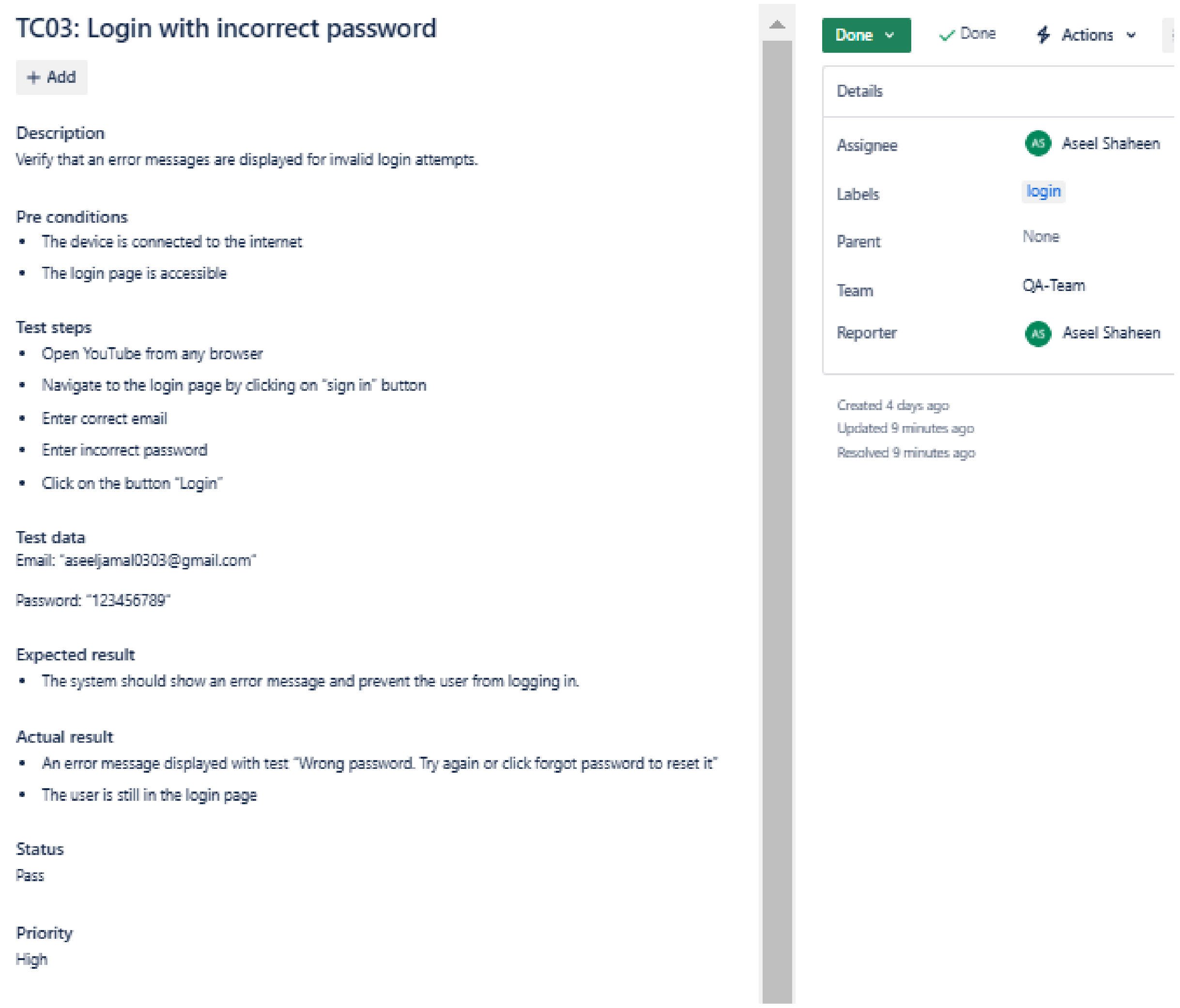
Conclusion: it worked as expected by allowing the user to submit an non-empty comment.

Detect bugs report:

No bugs detected.

Negative test case:

TC03: Login with incorrect password (link in Jira)



TC03: Login with incorrect password

Description

Verify that an error messages are displayed for invalid login attempts.

Pre conditions

- The device is connected to the internet
- The login page is accessible

Test steps

- Open YouTube from any browser
- Navigate to the login page by clicking on "sign in" button
- Enter correct email
- Enter incorrect password
- Click on the button "Login"

Test data

Email: "aseeljamal0303@gmail.com"

Password: "123456789"

Expected result

- The system should show an error message and prevent the user from logging in.

Actual result

- An error message displayed with test "Wrong password. Try again or click forgot password to reset it"
- The user is still in the login page

Status

Pass

Priority

High

Details

Assignee	AS Aseel Shaheen
Labels	login
Parent	None
Team	QA-Team
Reporter	AS Aseel Shaheen

Created 4 days ago

Updated 9 minutes ago

Resolved 9 minutes ago

Analyse the result:

Date 05/01/2025

Expected result: the system should show an error message and prevent the user from logging in.

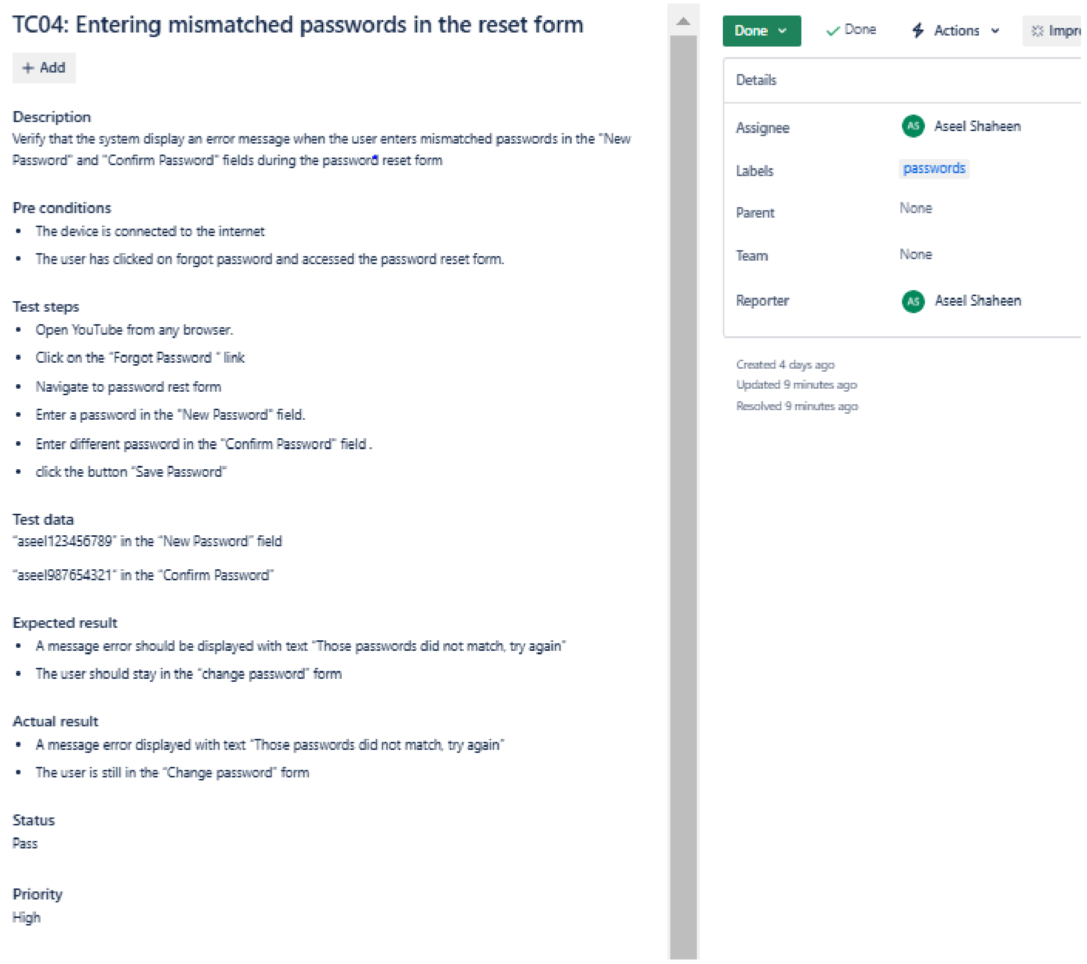
Actual result: an error message displayed with test “wrong password, try again or click forgot password to reset it” and the user is still in the login page.

Conclusion: the system worked as expected by preventing users from logging in and displaying an error message.

Detect bugs report:

No bugs detected.

TC04: Entering mismatched passwords into the reset form (link in Jira)



TC04: Entering mismatched passwords in the reset form

Description
Verify that the system display an error message when the user enters mismatched passwords in the "New Password" and "Confirm Password" fields during the password reset form

Pre conditions
- The device is connected to the internet
- The user has clicked on forgot password and accessed the password reset form.

Test steps
- Open YouTube from any browser.
- Click on the "Forgot Password " link
- Navigate to password rest form
- Enter a password in the "New Password" field.
- Enter different password in the "Confirm Password" field .
- click the button "Save Password"

Test data
"aseel123456789" in the "New Password" field

"aseel987654321" in the "Confirm Password"

Expected result
- A message error should be displayed with text "Those passwords did not match, try again"
- The user should stay in the "change password" form

Actual result
- A message error displayed with text "Those passwords did not match, try again"
- The user is still in the "Change password" form

Status
Pass

Priority
High

Details

Assignee	AS Aseel Shaheen
Labels	passwords
Parent	None
Team	None
Reporter	AS Aseel Shaheen

Created 4 days ago
Updated 9 minutes ago
Resolved 9 minutes ago

Analyse the result:

Date 05/01/2025

Expected result: the system should prevent the password updating and display an error message.

Actual result: a message error displayed with text "those passwords did not match, try again", the user is still in the "change password" form.

Conclusion: the system worked as expected by preventing the password update and displaying the error message.

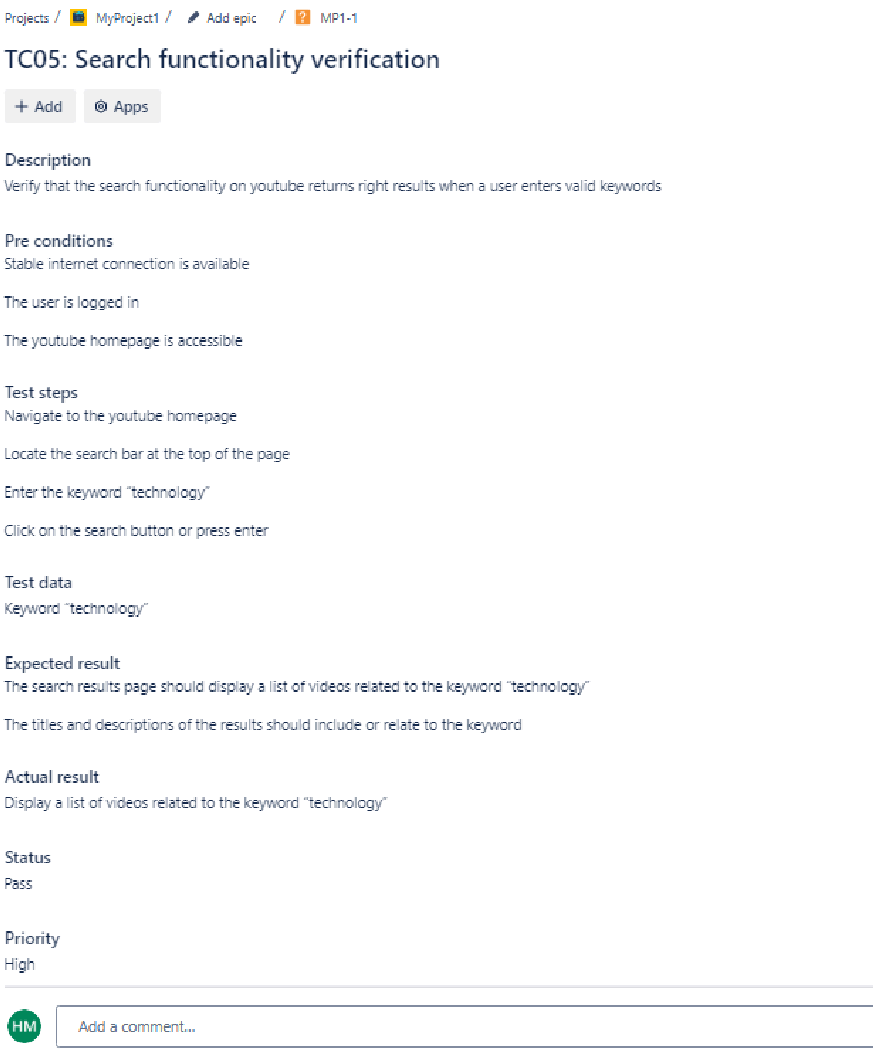
Detect bugs report:

No bugs detected.

Hanan

Positive test case:

TC05: Search functionality verification (link in Jira)



TC05: Search functionality verification

Description

Verify that the search functionality on youtube returns right results when a user enters valid keywords

Pre conditions

Stable internet connection is available

The user is logged in

The youtube homepage is accessible

Test steps

Navigate to the youtube homepage

Locate the search bar at the top of the page

Enter the keyword "technology"

Click on the search button or press enter

Test data

Keyword "technology"

Expected result

The search results page should display a list of videos related to the keyword "technology"

The titles and descriptions of the results should include or relate to the keyword

Actual result

Display a list of videos related to the keyword "technology"

Status

Pass

Priority

High

Analyze the results:

Date: 4/1/2025

Expected result: the search results page should display a list of videos related to the keyword “technology” and the titles and descriptions of the results should include or relate to the keyword.

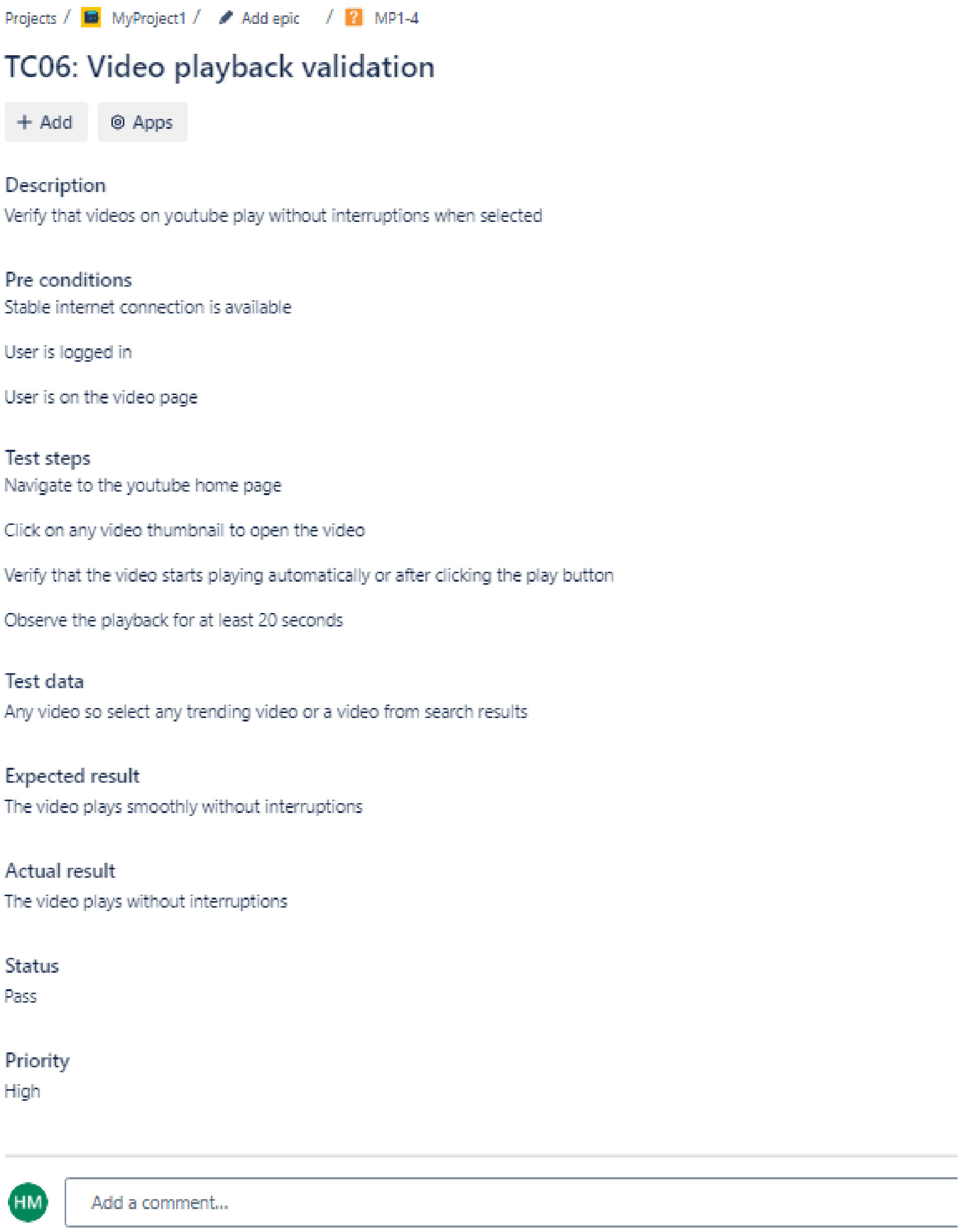
Actual result: display a list of videos related to the keyword “technology”.

Conclusion: it worked as expected, displaying relevant videos for the keyword "technology."

Detect bugs report:

No bugs detected.

TC06: Video playback validation (link in Jira)



TC06: Video playback validation

Description
Verify that videos on youtube play without interruptions when selected

Pre conditions
Stable internet connection is available

User is logged in

User is on the video page

Test steps
Navigate to the youtube home page

Click on any video thumbnail to open the video

Verify that the video starts playing automatically or after clicking the play button

Observe the playback for at least 20 seconds

Test data
Any video so select any trending video or a video from search results

Expected result
The video plays smoothly without interruptions

Actual result
The video plays without interruptions

Status
Pass

Priority
High

Analyze the results:

Date: 4/1/2025

Expected result: the video plays smoothly without interruptions.

Actual result: the video plays without interruptions.

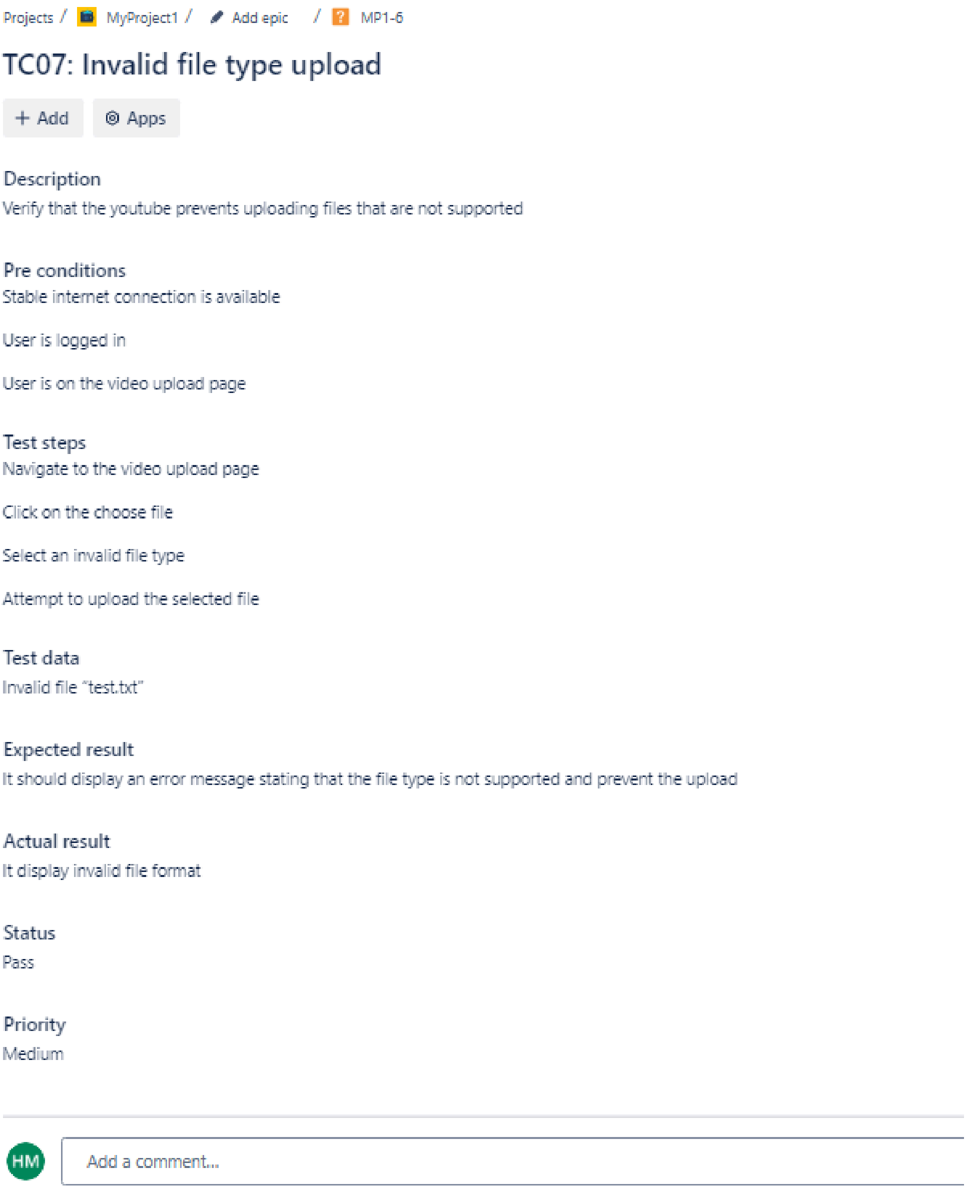
Conclusion: it worked as expected, the video played smoothly and without any interruptions.

Detect bugs report:

No bugs detected.

Negative test case:

TC07: Invalid file type upload (link in Jira)



TC07: Invalid file type upload

Description
Verify that the youtube prevents uploading files that are not supported

Pre conditions
Stable internet connection is available

User is logged in

User is on the video upload page

Test steps
Navigate to the video upload page

Click on the choose file

Select an invalid file type

Attempt to upload the selected file

Test data
Invalid file "test.txt"

Expected result
It should display an error message stating that the file type is not supported and prevent the upload

Actual result
It display invalid file format

Status
Pass

Priority
Medium

Analyze the results:

Date: 4/1/2025

Expected result: it should display an error message stating that the file type is not supported and prevent the upload.

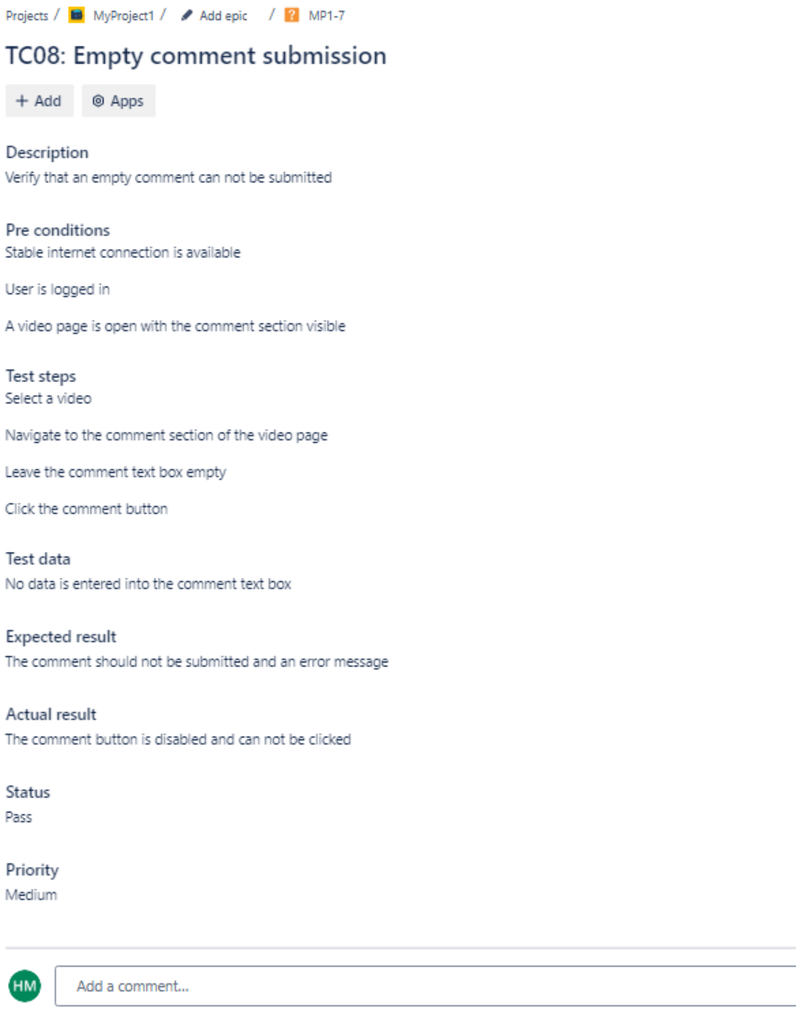
Actual result: it display invalid file format

Conclusion: the expected behavior was achieved, so the error message correctly prevents the upload.

Detect bugs report:

No bugs detected.

TC08: Empty comment submission (link in Jira)



Projects / MyProject1 / Add epic / MP1-7

TC08: Empty comment submission

Description
Verify that an empty comment can not be submitted

Pre conditions
Stable internet connection is available

User is logged in

A video page is open with the comment section visible

Test steps
Select a video

Navigate to the comment section of the video page

Leave the comment text box empty

Click the comment button

Test data
No data is entered into the comment text box

Expected result
The comment should not be submitted and an error message

Actual result
The comment button is disabled and can not be clicked

Status
Pass

Priority
Medium

Analyze the results:

Date: 4/1/2025

Expected result: the comment should not be submitted and an error message.

Actual result: the comment button is disabled and can not be clicked.

Conclusion: it worked as expected by disabling the comment button, preventing the empty comment from being submitted.

Detect bugs report:

Bug description: no error message displayed, although the button is disabled.

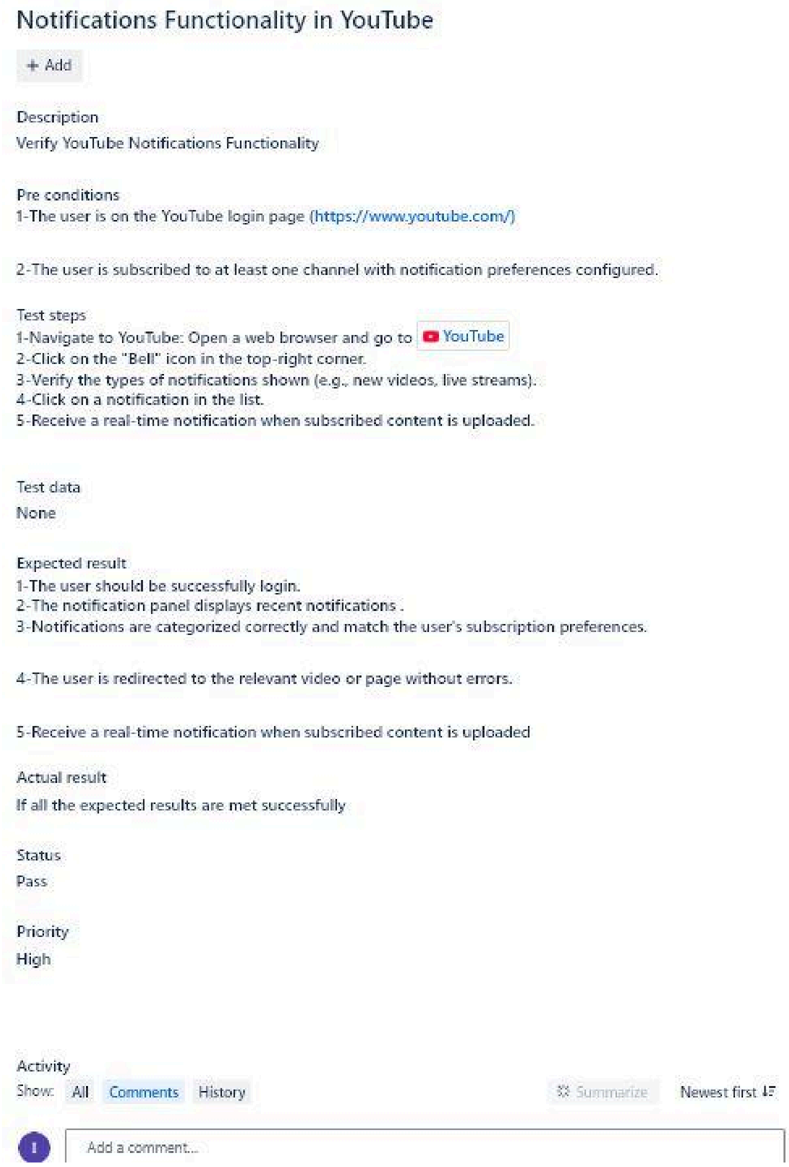
Expected result: the error message like comment cannot be empty, should appear along with the disabled button.

Actual result: the comment button is disabled, but no error message is shown to the user.

Ibraheem

Positive test case:

TC09: Notifications functionality (link in Jira)



Notifications Functionality in YouTube

Description
Verify YouTube Notifications Functionality

Pre conditions
1-The user is on the YouTube login page (https://www.youtube.com/)

2-The user is subscribed to at least one channel with notification preferences configured.

Test steps
1-Navigate to YouTube: Open a web browser and go to YouTube
2-Click on the "Bell" icon in the top-right corner.
3-Verify the types of notifications shown (e.g., new videos, live streams).
4-Click on a notification in the list.
5-Receive a real-time notification when subscribed content is uploaded.

Test data
None

Expected result
1-The user should be successfully login.
2-The notification panel displays recent notifications .
3-Notifications are categorized correctly and match the user's subscription preferences.

4-The user is redirected to the relevant video or page without errors.

5-Receive a real-time notification when subscribed content is uploaded

Actual result
If all the expected results are met successfully

Status
Pass

Priority
High

Activity

Analyze the results:

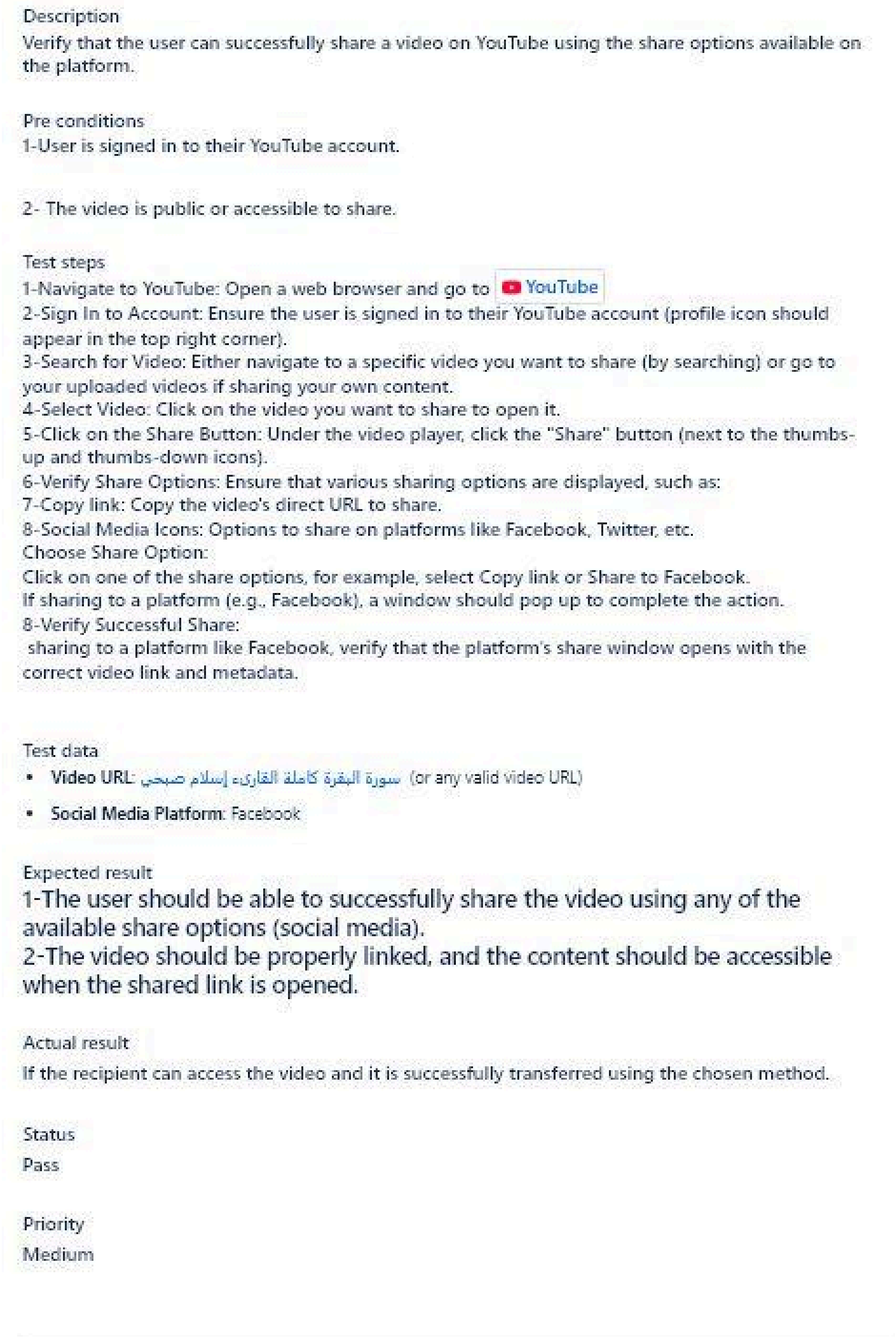
Expected result: the user should successfully login, the notification panel displays recent notifications, notifications are categorized correctly and match the users subscription preferences, the user is redirected to the relevant video or page without errors, receive a real-time notification when subscribed content is uploaded.

Actual result: if all the expected results are met successfully.

Detect bugs report:

No bugs detected.

TC10: Share video (link in Jira)



Description

Verify that the user can successfully share a video on YouTube using the share options available on the platform.

Pre conditions

1-User is signed in to their YouTube account.

2- The video is public or accessible to share.

Test steps

1-Navigate to YouTube: Open a web browser and go to YouTube
2-Sign In to Account: Ensure the user is signed in to their YouTube account (profile icon should appear in the top right corner).
3-Search for Video: Either navigate to a specific video you want to share (by searching) or go to your uploaded videos if sharing your own content.
4-Select Video: Click on the video you want to share to open it.
5-Click on the Share Button: Under the video player, click the "Share" button (next to the thumbs-up and thumbs-down icons).
6-Verify Share Options: Ensure that various sharing options are displayed, such as:
7-Copy link: Copy the video's direct URL to share.
8-Social Media Icons: Options to share on platforms like Facebook, Twitter, etc.
Choose Share Option:
Click on one of the share options, for example, select Copy link or Share to Facebook.
If sharing to a platform (e.g., Facebook), a window should pop up to complete the action.
8-Verify Successful Share:
sharing to a platform like Facebook, verify that the platform's share window opens with the correct video link and metadata.

Test data

- **Video URL:** سورة البقرة كاملة القارىء إسلام صبحي (or any valid video URL)
- **Social Media Platform:** Facebook

Expected result

1-The user should be able to successfully share the video using any of the available share options (social media).
2-The video should be properly linked, and the content should be accessible when the shared link is opened.

Actual result

If the recipient can access the video and it is successfully transferred using the chosen method.

Status

Pass

Priority

Medium

Analyze the results:

Expected result: the user should be able to successfully share the video using any of the available share options, the video should be properly linked, and the content should be accessible when the shared link is opened.

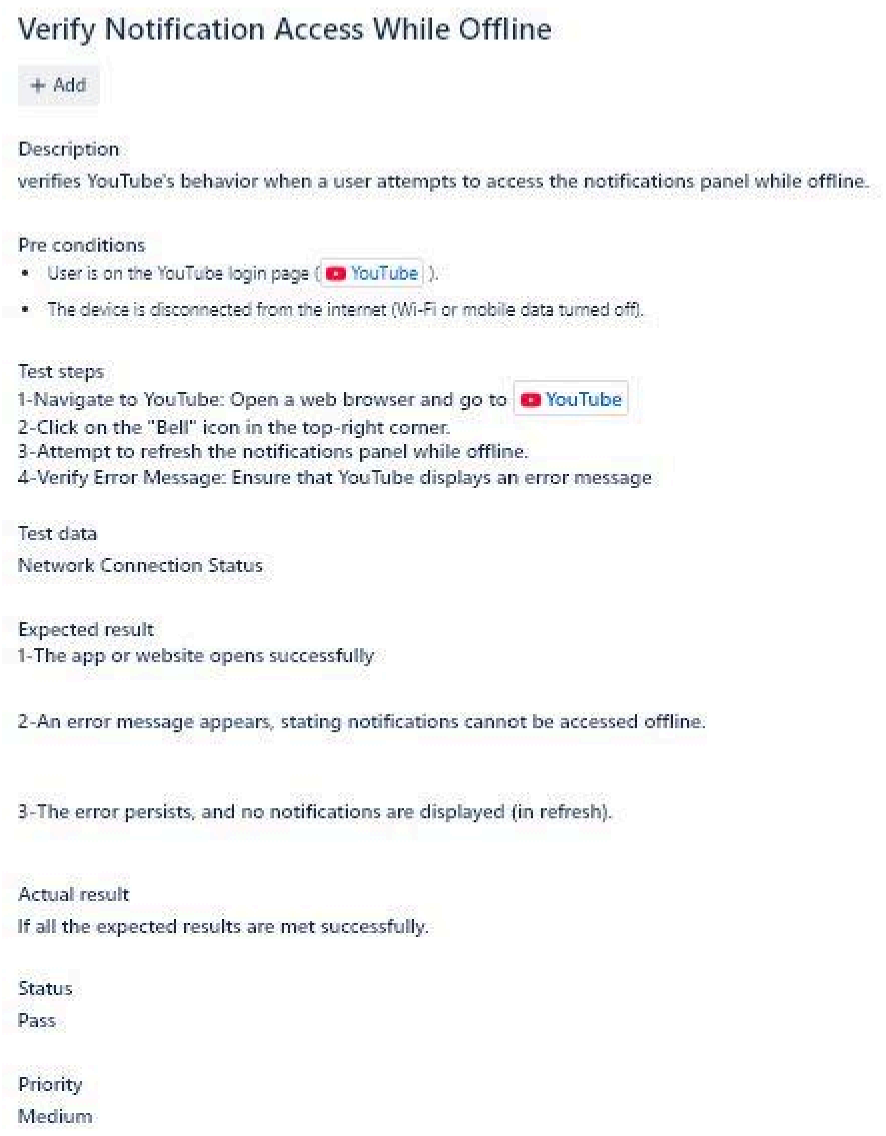
Actual result: if the recipient can access the video and it is successfully transferred using the chosen method.

Detect bugs report:

No bugs detected.

Negative test case:

TC11: Verify notification access while offline (link in Jira)



Verify Notification Access While Offline

Description
verifies YouTube's behavior when a user attempts to access the notifications panel while offline.

Pre conditions

- User is on the YouTube login page (▶ YouTube).
- The device is disconnected from the internet (Wi-Fi or mobile data turned off).

Test steps
1-Navigate to YouTube: Open a web browser and go to ▶ YouTube
2-Click on the "Bell" icon in the top-right corner.
3-Attempt to refresh the notifications panel while offline.
4-Verify Error Message: Ensure that YouTube displays an error message

Test data
Network Connection Status

Expected result
1-The app or website opens successfully

2-An error message appears, stating notifications cannot be accessed offline.

3-The error persists, and no notifications are displayed (in refresh).

Actual result
If all the expected results are met successfully.

Status
Pass

Priority
Medium

Analyze the results:

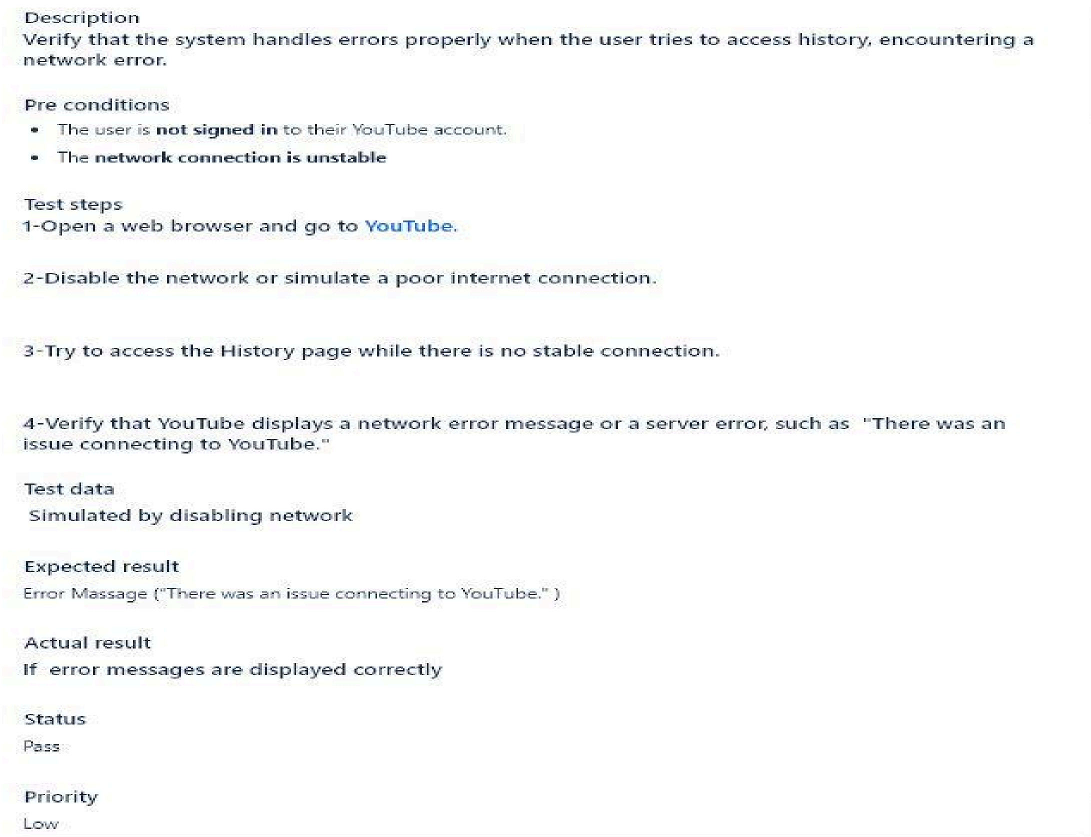
Expected result: the app or website opens successfully, an error message appears, stating notifications cannot be accessed offline, the error persists, and no notifications are displayed.

Actual result: if all the expected results are met successfully.

Detect bugs report:

No bugs detected.

TC12: View history with invalid conditions (link in Jira)



Description
Verify that the system handles errors properly when the user tries to access history, encountering a network error.

Pre conditions

- The user is **not signed in** to their YouTube account.
- The **network connection is unstable**

Test steps
1-Open a web browser and go to YouTube.

2-Disable the network or simulate a poor internet connection.

3-Try to access the History page while there is no stable connection.

4-Verify that YouTube displays a network error message or a server error, such as "There was an issue connecting to YouTube."

Test data
Simulated by disabling network

Expected result
Error Massage ("There was an issue connecting to YouTube.")

Actual result
If error messages are displayed correctly

Status
Pass

Priority
Low

Analyze the results:

Expected result: display error massage “there was an issue connecting to YouTube”.

Actual result: if error messages are displayed correctly.

Detect bugs report:

No bugs detected.

Requirement	Requirement description	Priority	Test case	Test result
R3: Account management	Handles user registration, login, and profile updates	High	TC01: Valid login functionality	Pass
R5:Comment	Allows users to find content using keywords	Medium	TC02:Valid comment submission	Pass
R3: Account management	Handles user registration, login, and profile updates	High	TC03: Login with incorrect password	Pass
R3: Account management	Handles user registration, login, and profile updates	High	TC04:Entering mismatched passwords in the reset form	Pass
R2: Search functionality	Allows users to find content using keywords	High	TC05: Search functionality verification	Pass
R1: Video playback	Enables viewing videos with playback controls.	High	TC06: Video playback validation	Pass
R4: Video upload	Facilitates uploading and storing videos	Medium	TC07: Invalid file type upload	Pass
R5: Comment	Supports user engagement through comments, likes, and replies	Medium	TC08: Empty comment submission	Pass
R10: Notification	Users should be notified of new videos or live streams from subscribed channels.	High	TC09: Notifications functionality	Pass
R6: Interaction	Allow users to like, dislike videos, and share videos	High	TC10: Share video	Pass
R10: Notification	Users should be notified of new videos or live streams from subscribed channels	Medium	TC11: Verify notification access while offline	Pass
R3: Account management	Enable features like viewing watch history and managing playlists	Low	TC12: View history with invalid conditions	Pass

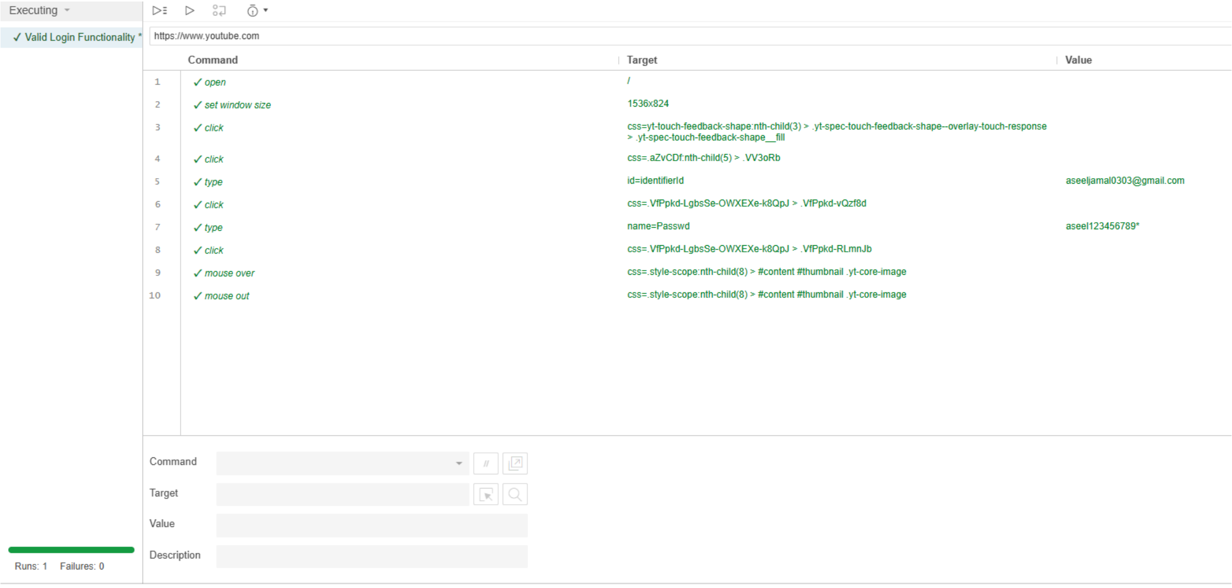
Table1: traceability matrix

Test Automation

This section contains automated test cases using Selenium, each group member implemented 2 test cases, ensuring efficient and reliable testing processes.

Asil

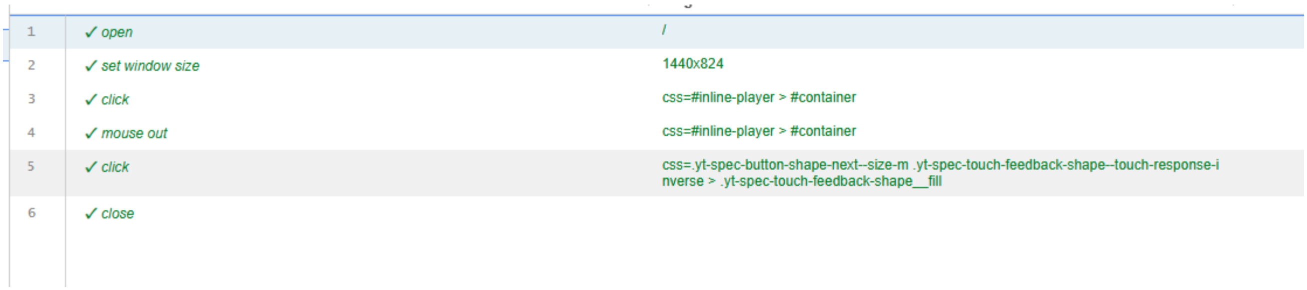
Valid login functionality



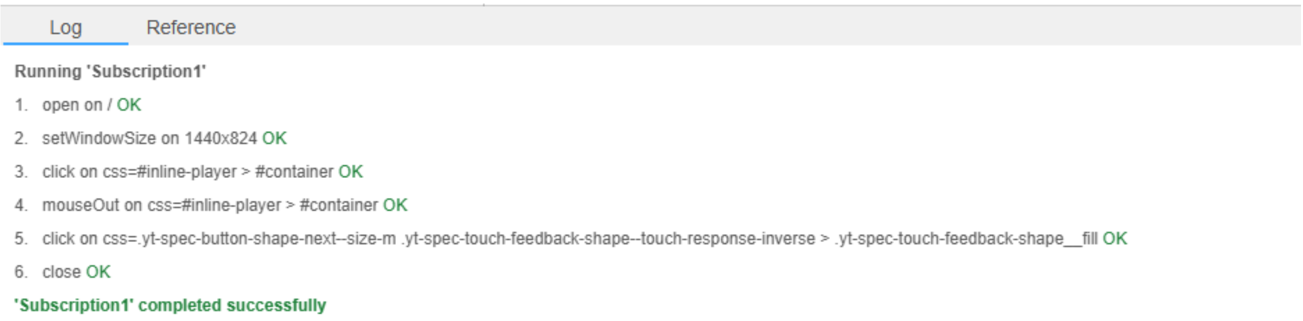
https://www.youtube.com

	Command	Target	Value
1	✓ open	/	
2	✓ set window size	1536x824	
3	✓ click	css=yt-touch-feedback-shape:nth-child(3) > .yt-spec-touch-feedback-shape--overlay-touch-response > .yt-spec-touch-feedback-shape__fill	
4	✓ click	css=.aZvCDf:nth-child(5) > .VV3oRb	
5	✓ type	id=identifierId	aseeljamal0303@gmail.com
6	✓ click	css=.VfPpkd-LgbsSe-OWXEXe-k8QpJ > .VfPpkd-vQzf8d	
7	✓ type	name=Passwd	aseel123456789*
8	✓ click	css=.VfPpkd-LgbsSe-OWXEXe-k8QpJ > .VfPpkd-RLmnJb	
9	✓ mouse over	css=.style-scope:nth-child(8) > #content #thumbnail .yt-core-image	
10	✓ mouse out	css=.style-scope:nth-child(8) > #content #thumbnail .yt-core-image	

Subscription



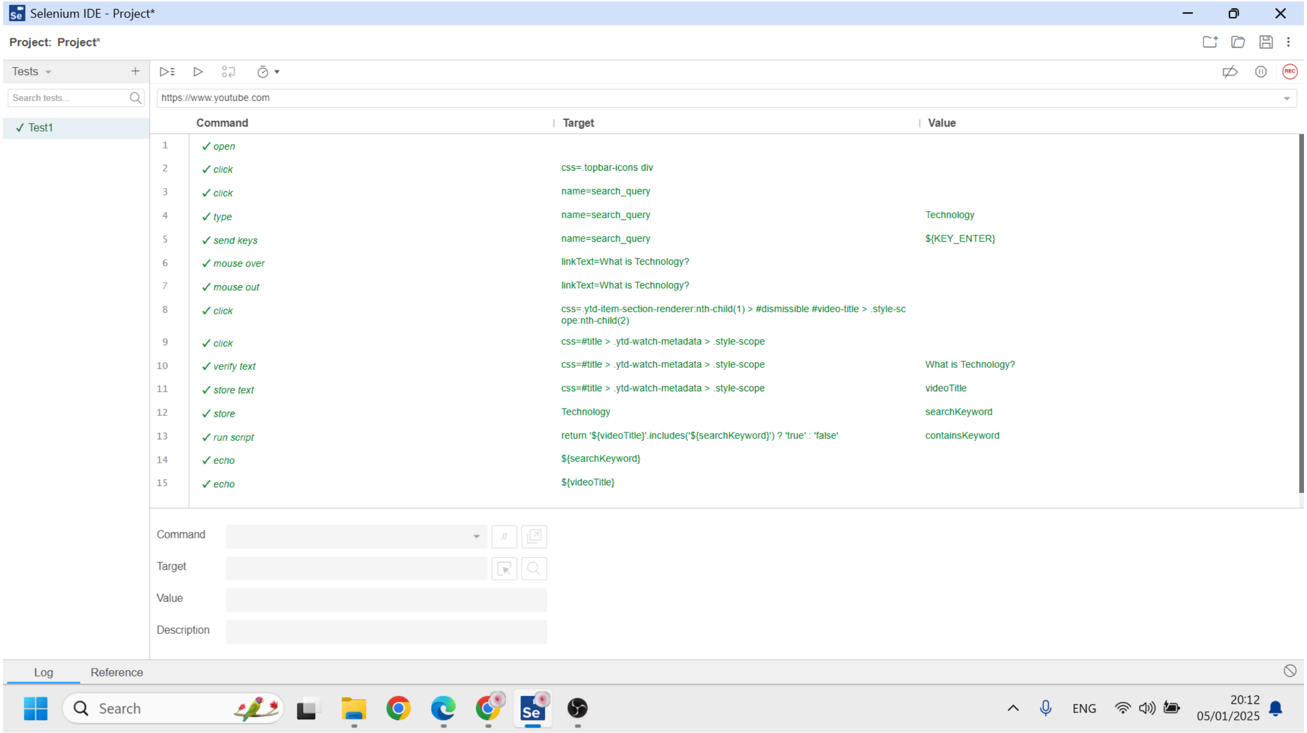
1	✓ open	/
2	✓ set window size	1440x824
3	✓ click	css=#inline-player > #container
4	✓ mouse out	css=#inline-player > #container
5	✓ click	css=.yt-spec-button-shape-next--size-m .yt-spec-touch-feedback-shape--touch-response-inverse > .yt-spec-touch-feedback-shape__fill
6	✓ close	



```
Running 'Subscription1'
1. open on / OK
2. setWindowSize on 1440x824 OK
3. click on css=#inline-player > #container OK
4. mouseOut on css=#inline-player > #container OK
5. click on css=.yt-spec-button-shape-next--size-m .yt-spec-touch-feedback-shape--touch-response-inverse > .yt-spec-touch-feedback-shape__fill OK
6. close OK
'Subscription1' completed successfully
```

Hanan

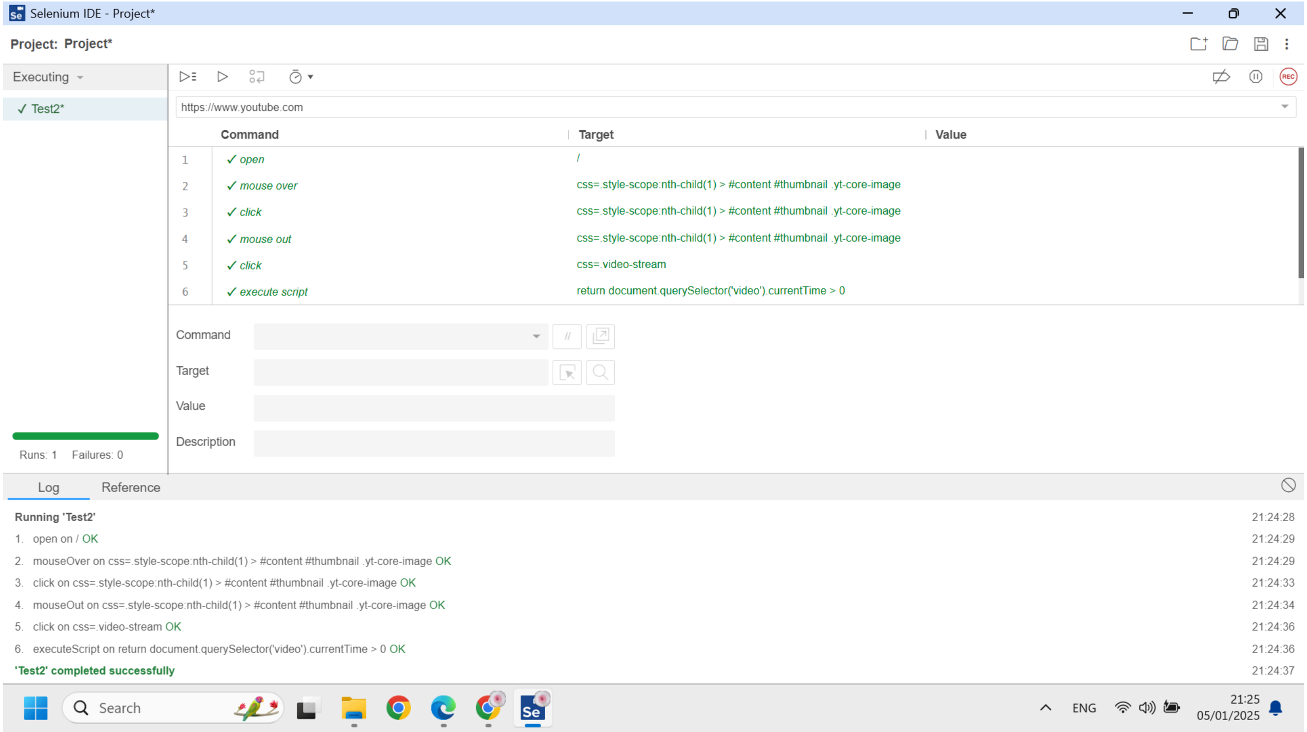
Search functionality verification



https://www.youtube.com

	Command	Target	Value
1	✓ open		
2	✓ click	css=.topbar-icons div	
3	✓ click	name=search_query	
4	✓ type	name=search_query	Technology
5	✓ send keys	name=search_query	${KEY_ENTER}
6	✓ mouse over	linkText=What is Technology?	
7	✓ mouse out	linkText=What is Technology?	
8	✓ click	css=.ytd-item-section-renderer:nth-child(1) > #dismissible #video-title > .style-scope:nth-child(2)	
9	✓ click	css=#title > .ytd-watch-metadata > .style-scope	
10	✓ verify text	css=#title > .ytd-watch-metadata > .style-scope	What is Technology?
11	✓ store text	css=#title > .ytd-watch-metadata > .style-scope	videoTitle
12	✓ store	Technology	searchKeyword
13	✓ run script	return '${videoTitle}'.includes('${searchKeyword}') ? 'true' : 'false'	containsKeyword
14	✓ echo	${searchKeyword}	
15	✓ echo	${videoTitle}	

Video playback validation



Selenium IDE - Project*

Project: Project*

Executing

✓ Test2*

https://www.youtube.com

	Command	Target	Value
1	✓ open	/	
2	✓ mouse over	css=.style-scope:nth-child(1) > #content #thumbnail .yt-core-image	
3	✓ click	css=.style-scope:nth-child(1) > #content #thumbnail .yt-core-image	
4	✓ mouse out	css=.style-scope:nth-child(1) > #content #thumbnail .yt-core-image	
5	✓ click	css=.video-stream	
6	✓ execute script	return document.querySelector('video').currentTime > 0	

Command

Target

Value

Description

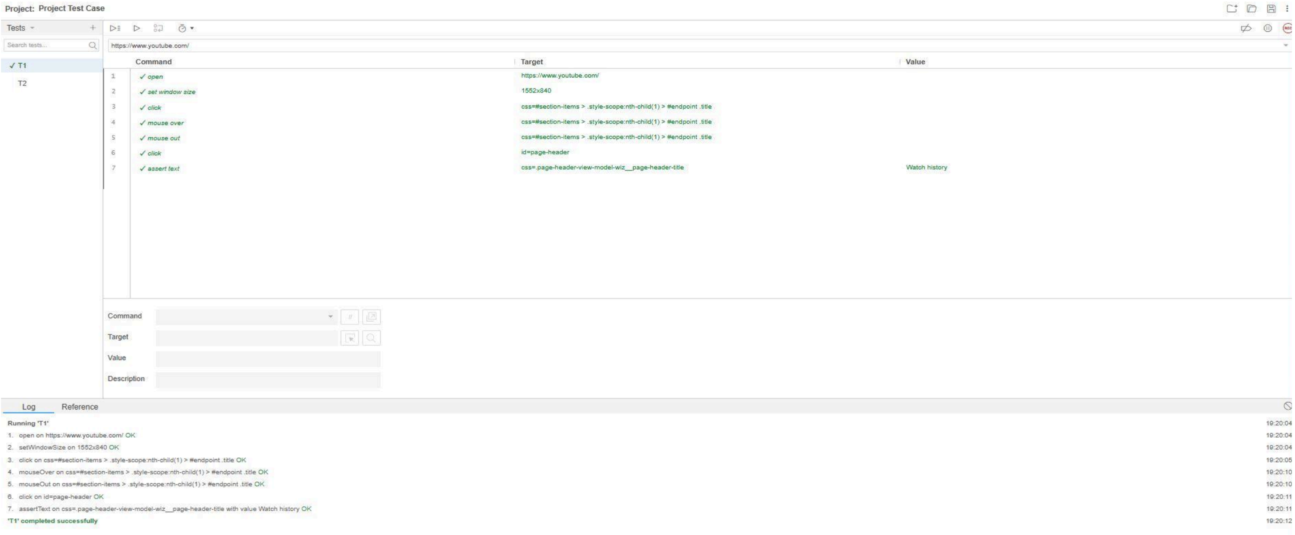
Runs: 1 Failures: 0

Log Reference

```
Running 'Test2'                                                                          21:24:28
1. open on / OK                                                                          21:24:29
2. mouseOver on css=.style-scope:nth-child(1) > #content #thumbnail .yt-core-image OK    21:24:29
3. click on css=.style-scope:nth-child(1) > #content #thumbnail .yt-core-image OK        21:24:33
4. mouseOut on css=.style-scope:nth-child(1) > #content #thumbnail .yt-core-image OK     21:24:34
5. click on css=.video-stream OK                                                         21:24:36
6. executeScript on return document.querySelector('video').currentTime > 0 OK            21:24:36
'Test2' completed successfully                                                           21:24:37
```

Ibraheem

View history with invalid conditions



Project: Project Test Case

Tests

Search tests...

✓ T1

T2

https://www.youtube.com/

	Command	Target	Value
1	✓ open	https://www.youtube.com/	
2	✓ set window size	1552x840	
3	✓ click	css=#section-items > .style-scope:nth-child(1) > #endpoint .title	
4	✓ mouse over	css=#section-items > .style-scope:nth-child(1) > #endpoint .title	
5	✓ mouse out	css=#section-items > .style-scope:nth-child(1) > #endpoint .title	
6	✓ click	id=page-header	
7	✓ assert text	css=.page-header-view-model-wiz__page-header-title	Watch history

Command

Target

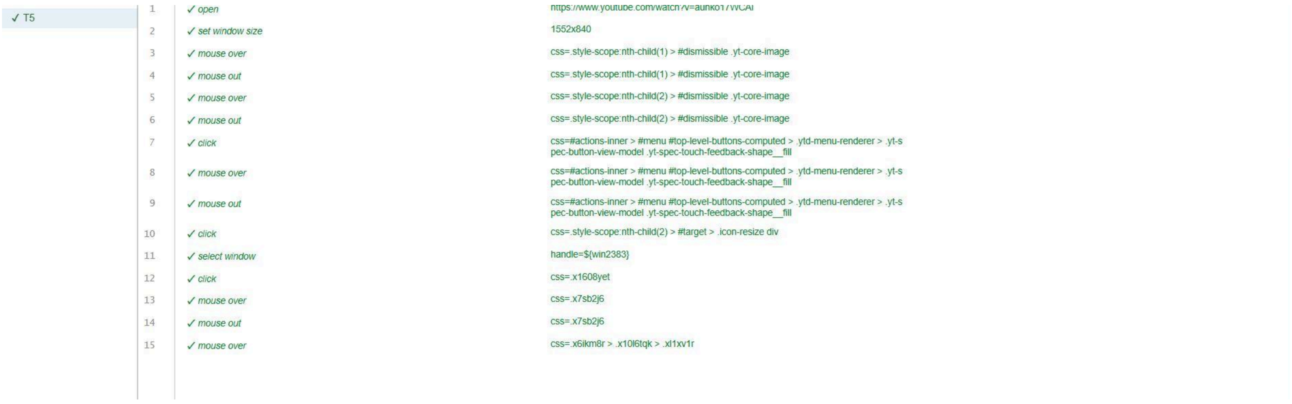
Value

Description

Log Reference

```
Running 'T1'                                                                                                19:20:04
1. open on https://www.youtube.com/ OK                                                                      19:20:04
2. setWindowSize on 1552x840 OK                                                                             19:20:04
3. click on css=#section-items > .style-scope:nth-child(1) > #endpoint .title OK                            19:20:05
4. mouseOver on css=#section-items > .style-scope:nth-child(1) > #endpoint .title OK                        19:20:10
5. mouseOut on css=#section-items > .style-scope:nth-child(1) > #endpoint .title OK                         19:20:10
6. click on id=page-header OK                                                                               19:20:11
7. assertText on css=.page-header-view-model-wiz__page-header-title with value Watch history OK             19:20:11
'T1' completed successfully                                                                                 19:20:12
```

Share video



✓ T5

1	✓ open	https://www.youtube.com/watch?v=aunko17WCAI
2	✓ set window size	1552x840
3	✓ mouse over	css=.style-scope:nth-child(1) > #dismissible .yt-core-image
4	✓ mouse out	css=.style-scope:nth-child(1) > #dismissible .yt-core-image
5	✓ mouse over	css=.style-scope:nth-child(2) > #dismissible .yt-core-image
6	✓ mouse out	css=.style-scope:nth-child(2) > #dismissible .yt-core-image
7	✓ click	css=#actions-inner > #menu #top-level-buttons-computed > .ytd-menu-renderer > .yt-spec-button-view-model .yt-spec-touch-feedback-shape__fill
8	✓ mouse over	css=#actions-inner > #menu #top-level-buttons-computed > .ytd-menu-renderer > .yt-spec-button-view-model .yt-spec-touch-feedback-shape__fill
9	✓ mouse out	css=#actions-inner > #menu #top-level-buttons-computed > .ytd-menu-renderer > .yt-spec-button-view-model .yt-spec-touch-feedback-shape__fill
10	✓ click	css=.style-scope:nth-child(2) > #target > .icon-resize div
11	✓ select window	handle=${win2383}
12	✓ click	css=.x1608yet
13	✓ mouse over	css=.x7sb2j6
14	✓ mouse out	css=.x7sb2j6
15	✓ mouse over	css=.x6ikm8r > .x10l6tqk > .xl1xv1r

Performance & Load Testing

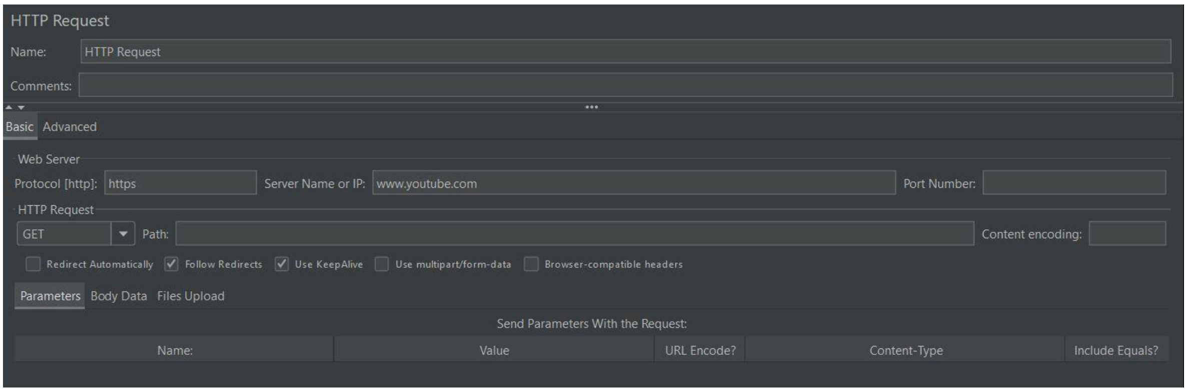
This section contains the performance and load testing for the YouTube website, using JMeter to evaluate the system performance under varying levels of user activity, the goal is to simulate real-world scenarios.

The following reliability testing scenarios were performed:

Load testing: measure the YouTube performance to **the main page** under expected loads to ensure it can handle the anticipated user base without degrading performance.

Thread properties:

- Number of user: 300
- Ramp-up period (in sec): 2
- loop count: 1



HTTP Request

Name: HTTP Request

Comments:

Basic Advanced

Web Server

Protocol [http]: https Server Name or IP: www.youtube.com Port Number:

HTTP Request

GET Path: Content encoding:

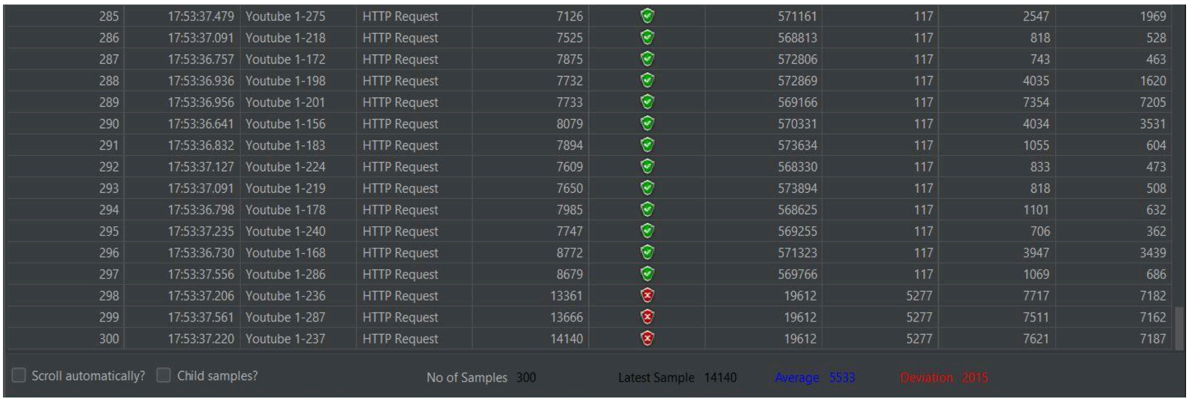
□ Redirect Automatically ☑ Follow Redirects ☑ Use KeepAlive □ Use multipart/form-data □ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	URL Encode?	Content-Type	Include Equals?

Figure: http request



285	17:53:37.479	Youtube 1-275	HTTP Request	7126	✓	571161	117	2547	1969
286	17:53:37.091	Youtube 1-218	HTTP Request	7525	✓	568813	117	818	528
287	17:53:36.757	Youtube 1-172	HTTP Request	7875	✓	572806	117	743	463
288	17:53:36.936	Youtube 1-198	HTTP Request	7732	✓	572869	117	4035	1620
289	17:53:36.956	Youtube 1-201	HTTP Request	7733	✓	569166	117	7354	7205
290	17:53:36.641	Youtube 1-156	HTTP Request	8079	✓	570331	117	4034	3531
291	17:53:36.832	Youtube 1-183	HTTP Request	7894	✓	573634	117	1055	604
292	17:53:37.127	Youtube 1-224	HTTP Request	7609	✓	568330	117	833	473
293	17:53:37.091	Youtube 1-219	HTTP Request	7650	✓	573894	117	818	508
294	17:53:36.798	Youtube 1-178	HTTP Request	7985	✓	568625	117	1101	632
295	17:53:37.235	Youtube 1-240	HTTP Request	7747	✓	569255	117	706	362
296	17:53:36.730	Youtube 1-168	HTTP Request	8772	✓	571323	117	3947	3439
297	17:53:37.556	Youtube 1-286	HTTP Request	8679	✓	569766	117	1069	686
298	17:53:37.206	Youtube 1-236	HTTP Request	13361	✗	19612	5277	7717	7182
299	17:53:37.561	Youtube 1-287	HTTP Request	13666	✗	19612	5277	7511	7162
300	17:53:37.220	Youtube 1-237	HTTP Request	14140	✗	19612	5277	7621	7187

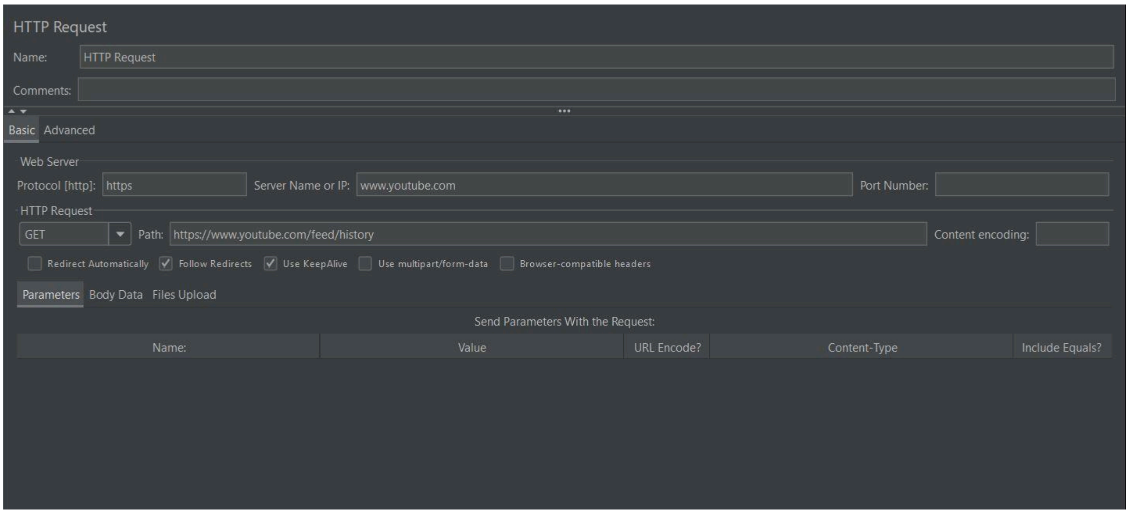
□ Scroll automatically? □ Child samples? No of Samples 300 Latest Sample 14140 Average 5533 Deviation 2015

Figure: view result table for 300 users

Load testing: measure the YouTube performance to **the history page** under expected loads to ensure it can handle the anticipated user base without degrading performance.

Thread properties:

- Number of user: 300
- Ramp-up period (in sec): 2
- loop count: 1



HTTP Request

Name: HTTP Request

Comments:

Basic Advanced

Web Server

Protocol [http]: https Server Name or IP: www.youtube.com Port Number:

HTTP Request

GET Path: https://www.youtube.com/feed/history Content encoding:

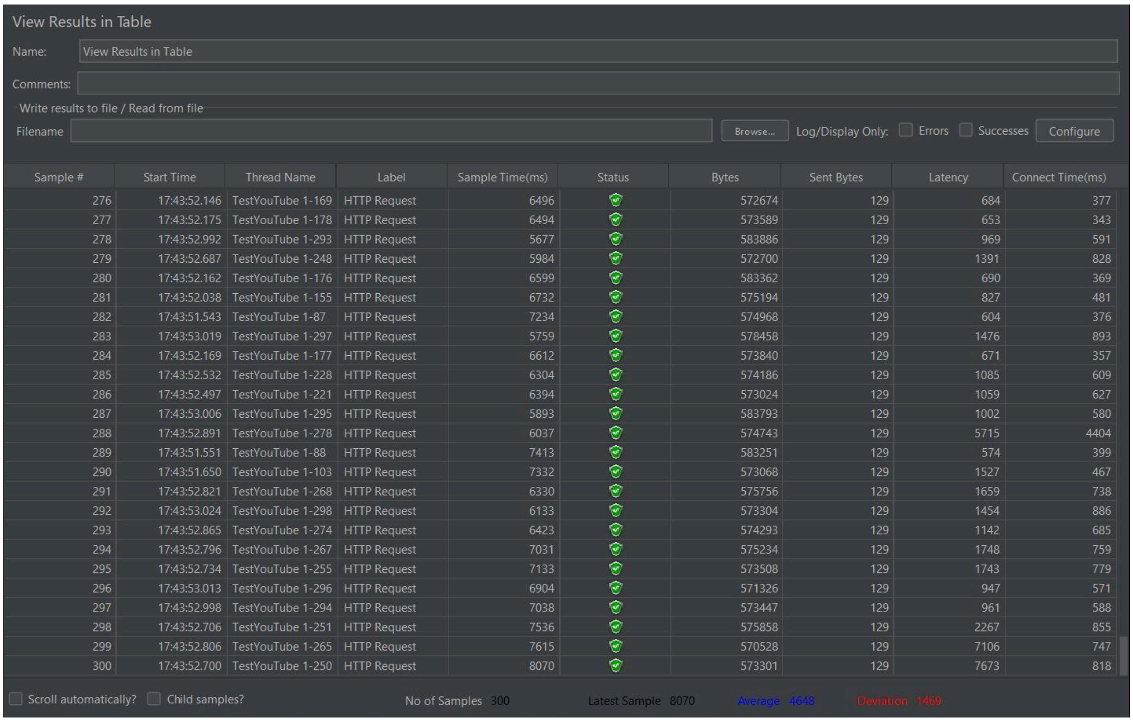
□ Redirect Automatically ☑ Follow Redirects ☑ Use KeepAlive □ Use multipart/form-data □ Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	URL Encode?	Content-Type	Include Equals?

Figure: http request



View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename Log/Display Only: □ Errors □ Successes

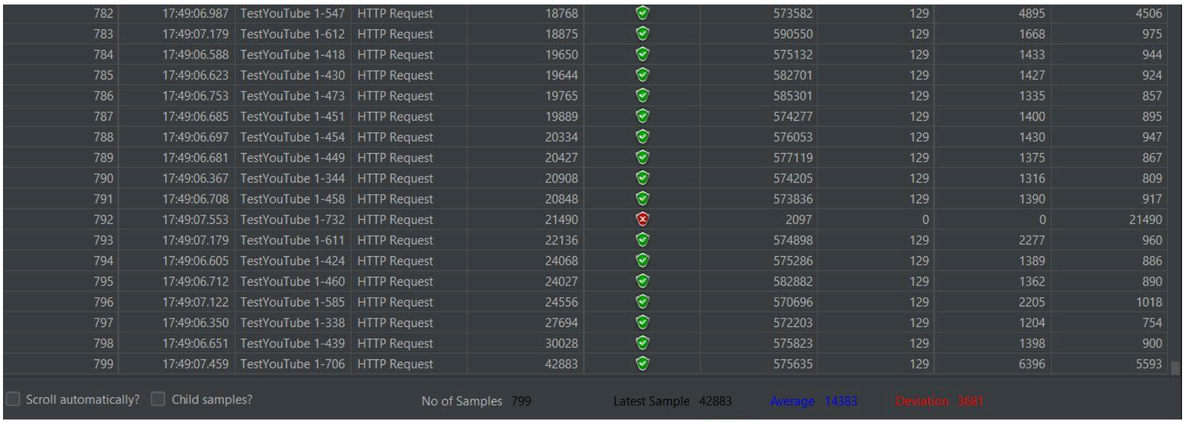
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
276	17:43:52.146	TestYouTube 1-169	HTTP Request	6496	✓	572674	129	684	377
277	17:43:52.175	TestYouTube 1-178	HTTP Request	6494	✓	573589	129	653	343
278	17:43:52.992	TestYouTube 1-293	HTTP Request	5677	✓	583886	129	969	591
279	17:43:52.687	TestYouTube 1-248	HTTP Request	5984	✓	572700	129	1391	828
280	17:43:52.162	TestYouTube 1-176	HTTP Request	6599	✓	583362	129	690	369
281	17:43:52.038	TestYouTube 1-155	HTTP Request	6732	✓	575194	129	827	481
282	17:43:51.543	TestYouTube 1-87	HTTP Request	7234	✓	574968	129	604	376
283	17:43:53.019	TestYouTube 1-297	HTTP Request	5759	✓	578458	129	1476	893
284	17:43:52.169	TestYouTube 1-177	HTTP Request	6612	✓	573840	129	671	357
285	17:43:52.532	TestYouTube 1-228	HTTP Request	6304	✓	574186	129	1085	609
286	17:43:52.497	TestYouTube 1-221	HTTP Request	6394	✓	573024	129	1059	627
287	17:43:53.006	TestYouTube 1-295	HTTP Request	5893	✓	583793	129	1002	580
288	17:43:52.891	TestYouTube 1-278	HTTP Request	6037	✓	574743	129	5715	4404
289	17:43:51.551	TestYouTube 1-88	HTTP Request	7413	✓	583251	129	574	399
290	17:43:51.650	TestYouTube 1-103	HTTP Request	7332	✓	573068	129	1527	467
291	17:43:52.821	TestYouTube 1-268	HTTP Request	6330	✓	575756	129	1659	738
292	17:43:53.024	TestYouTube 1-298	HTTP Request	6133	✓	573304	129	1454	886
293	17:43:52.865	TestYouTube 1-274	HTTP Request	6423	✓	574293	129	1142	685
294	17:43:52.796	TestYouTube 1-267	HTTP Request	7031	✓	575234	129	1748	759
295	17:43:52.734	TestYouTube 1-255	HTTP Request	7133	✓	573508	129	1743	779
296	17:43:53.013	TestYouTube 1-296	HTTP Request	6904	✓	571326	129	947	571
297	17:43:52.998	TestYouTube 1-294	HTTP Request	7038	✓	573447	129	961	588
298	17:43:52.706	TestYouTube 1-251	HTTP Request	7536	✓	575858	129	2267	855
299	17:43:52.806	TestYouTube 1-265	HTTP Request	7615	✓	570528	129	7106	747
300	17:43:52.700	TestYouTube 1-250	HTTP Request	8070	✓	573301	129	7673	818

□ Scroll automatically? □ Child samples? No of Samples 300 Latest Sample 8070 Average 4648 Deviation 1469

Figure: view result table for 300 users

Thread properties:

- Number of user: 800
- Ramp-up period (in sec): 2
- loop count: 1



782	17:49:06.987	TestYouTube 1-547	HTTP Request	18768	✓	573582	129	4895	4506
783	17:49:07.179	TestYouTube 1-612	HTTP Request	18875	✓	590550	129	1668	975
784	17:49:06.588	TestYouTube 1-418	HTTP Request	19650	✓	575132	129	1433	944
785	17:49:06.623	TestYouTube 1-430	HTTP Request	19644	✓	582701	129	1427	924
786	17:49:06.753	TestYouTube 1-473	HTTP Request	19765	✓	585301	129	1335	857
787	17:49:06.685	TestYouTube 1-451	HTTP Request	19889	✓	574277	129	1400	895
788	17:49:06.697	TestYouTube 1-454	HTTP Request	20334	✓	576053	129	1430	947
789	17:49:06.681	TestYouTube 1-449	HTTP Request	20427	✓	577119	129	1375	867
790	17:49:06.367	TestYouTube 1-344	HTTP Request	20908	✓	574205	129	1316	809
791	17:49:06.708	TestYouTube 1-458	HTTP Request	20848	✓	573836	129	1390	917
792	17:49:07.553	TestYouTube 1-732	HTTP Request	21490	✗	2097	0	0	21490
793	17:49:07.179	TestYouTube 1-611	HTTP Request	22136	✓	574898	129	2277	960
794	17:49:06.605	TestYouTube 1-424	HTTP Request	24068	✓	575286	129	1389	886
795	17:49:06.712	TestYouTube 1-460	HTTP Request	24027	✓	582882	129	1362	890
796	17:49:07.122	TestYouTube 1-585	HTTP Request	24556	✓	570696	129	2205	1018
797	17:49:06.350	TestYouTube 1-338	HTTP Request	27694	✓	572203	129	1204	754
798	17:49:06.651	TestYouTube 1-439	HTTP Request	30028	✓	575823	129	1398	900
799	17:49:07.459	TestYouTube 1-706	HTTP Request	42883	✓	575635	129	6396	5593

□ Scroll automatically? □ Child samples? No of Samples 799 Latest Sample 42883 Average 14383 Deviation 3681

Figure: view result table for 800 users

Challenges & Solutions

The YouTube QA project faced challenges like testing complex features, to manage tasks were divided, and detailed test cases were created, and performance testing required simulating high traffic, which was handled using tools like JMeter, these solutions helped ensure effective testing.

Conclusion

This project focused on testing the YouTube website to ensure its functionality, performance, and reliability.

By following a structured QA process, we evaluated the systems ability to handle different scenarios and provide a smooth user experience, the testing helped identify strengths and potential issues, ensuring that the website performs well under various conditions.

Overall, this project highlighted the importance of thorough testing in improving software quality and ensuring that YouTube remains a reliable and user-friendly system.