Finite-State Machine and Input Domain Partitioning Testing for a Video-Sharing Platform

Name: Minho Song

SMUID: 49408418

Abstract

The absence of formal testing in a newly developed video-sharing platform posed significant risks to its reliability and user experience, particularly in critical functionalities such as video upload, user management, and playback. This project applies two formal black-box testing techniques to meticulously evaluate the platform: Finite State Machine testing, and Input Domain Partitioning and Boundary testing. FSM testing focuses on verifying correct state transitions across various modules. Input Domain Partitioning explores input validation across multiple input fields. The testing identifies a number of defects, which are classified using the Orthogonal Defect Classification (ODC) framework. A two-way analysis reveals critical areas affecting the reliability of the system and the core functionalities, guiding targeted improvements. Implementation of error handlings, input validation mechanisms and logic modifications are performed as follow-up activities. A series of retest confirmed the effectiveness of the fixes. The project emphasizes the necessity of formal testing methodologies for enhanced reliability and usability of complex web applications that involves user interactions.

1. Introduction

1.1. Background

The video-sharing platform has been fully developed individually but has not yet undergone any formal testing except several ad-hoc testing. Although no specific defects or issues have been identified at this point, the absence of testing poses a risk of potential problems during the program operation. Because of the characteristic of video-sharing platforms, the platform's main focus is on video-upload and playback features. These functions must be thoroughly tested. The lack of previous testing raises concerns about potential defects in state transitions between processes or inappropriate input handling that could lead to unexpected behaviors or system crash.

In addition, provided that this project was built independently, obtaining user-usage data for operational profiles such as Musa OP or Markov OP seems infeasible. As a result, coverage-based testing techniques are more appropriate to simulate typical user interactions and identify potential defect injections in the application. This initial testing activity will function as the primary method that verifies the expected system behaviors under normal and edge-case conditions.

1.2. Problem

The absence of formal testing for the video-sharing platform's critical functionalities presents risks of defects that could affect the system's reliability and user experience. Key functionalities, like video upload and playback, need thorough testing to identify potential issues in state transitions and input handling.

1.3 . Objective

The primary object of this project is to apply two formal black-box testing techniques-Finite State Machine (FSM) testing and Input Domain Partitioning with Boundary testing- to the video-sharing platform. These techniques will make sure thorough coverage of the system's key features, validate expected behaviors, and identify defects for subsequent analysis and resolution.

2. Testing Strategy

2.1. Finite State Machine Testing

FSM testing will be applied for state transitions both between processes and within each process to make sure functionalities as expected. The FSM-based approach has an advantage for this video-sharing platform since the application has sequential processes where errors in transitions can lead to major malfunctions.

2.1.1. FSM Models

Due to the characteristics of web-based software, there were a few challenges in constructing FSM for the video-sharing platform, such as multiple entry points, asynchronous interactions, unpredictable user navigation that leads to state explosion. In order to avoid state explosion and keep FSM manageable, the application has been separated into individual functional units (modules) where each module has its own FSM.

However, there are cross-module transitions influenced by login status. To reflect these conditions, different abbreviations are used for states, such as H1 for the logged-out homepage and H2 for the logged-in homepage. This approach allows the FSM to maintain modularity while accounting for user authentication status. In addition, Playback for Video Viewing and Recording for Video Management are

treated as internal events within the states. Even though they play core functions in the platform, these actions do not lead to a page transition. The model focuses on the transitions between stages of the playback and recording process while maintaining the simplicity of modular states.

The modules are: Homepage and Navigation module (H), Authentication module(A), Search module(S), Profile management module(P), Video management module(VM), Recording(R), and Video viewing module(VV).

Each module has its own FSM. Each FSM is detailed in **appendix 1**:

Table 1: Module to Module Transitions

To / From	Homepage (H1, H2)	Authentication (A)	Search (S)	Profile Management (P)	Video Management (VM)	Video Viewing (VV)
Homepage (H1)	0	**(log/join, -)**	**(srch, -)**	0	0	**(vid, -)**
Homepage (H2)	0	0	**(srch, -)**	**(prof, -)**	**(up, crtvid) (myvid, -)**	**(vid, -)**
Authentication (A1, A2, A3)	**(slog, -) (git, -) (sjoin, acc)**	0	0	0	0	0
Search (S1, S2, S4)	0	0	0	0	0	**(vid, -)**
Profile Management (P1, P2)	**(chgpwd, -)**	0	0	0	0	0
Video Management (VM1, VM2, VM3)	**(sedit, -)**	0	0	0	0	**(vid, -)**
Video Viewing (VV1, VV2)	**(del, delvid)**	0	0	0	**(edit, -)**	0

Click login	**log**	Click video	**vid**	click Edit profile	**prof**	Successful Login	**slog**
Click Join	**join**	Click upload	**up**	Click logout	**out**	Successful Join	**sjoin**
Click search icon	**srch**	Click avatar	**myvid**	Login with github	**git**	Successful password change	**chgpwd**
Successful video edit	**sedit**	Click Edit video	**edit**	Click Delete video	**del**	Account created	**acc**
Video created	**crtvid**	Video deleted	**delvid**				

2.2. Input Domain Partitioning and Boundary testing

Input Domain Partitioning will focus on input validation. This technique will partition the input space into distinct sub-domains to check whether the system behaves properly under all input conditions such as valid, invalid, and boundary conditions.

2.2.1. Input Domain partition boundary testing model

In this video-sharing platform, there are multiple input fields across different modules such as authentication, profile management, video management, and search functionality. By analyzing and partitioning the input spaces and applying boundary testing techniques, this testing model aims to detect defects related to handling of the inputs in the system.

2.2.1.1. Input identification

Through reading source code, several input fields are reused across different pages. For example, username is used in join, login, and edit profile page while title is used in upload and edit video page. Given this, these reused input fields are treated as one input domain for each unique field. This strategy will improve efficiency of testing by removing redundancy.

The input spaces identified through code review are as below:

1. **Authentication**: Username, Password
2. **Profile Management**: Name, Email, Location, Avatar
3. **Video Management**: Title, Description, Hash tags, Video, Thumbnail
4. **Search**: Search query
5. **Comment**: Comment text

2.2.1.2. Sub-domain and domain analysis

The input domain analysis for this video-sharing platform was conducted. Each input space is examined and classified into sub-domains. Sub-domains are classified based on constraints specific to each input filed, such as length, format, and content limits. Both valid and invalid input values were considered in the classification for comprehensive coverage of the input space. The summary of the sub-domains is in the **appendix 4**.

Each input domain is partitioned into valid and invalid sub-domains based on the constraints identified through code review. This classification forms the basis for the subsequent domain analysis.

For comprehensive coverage of sub-domains, Weak N x 1 strategy is utilized for both one-dimensional and multi-dimensional input spaces. Initially EPC was considered for one dimensional input space and Weak N x 1 for multi-dimensional input space. However, Weak N x 1 strategy is chosen for all domains because EPC causes an exponential increase in testing points as the number of dimension rises while Weak N x 1 has the same number of testing points as EPC has for one dimensional input field. Testing points from domain analysis is in the **appendix 5**.

2.3. Orthogonal Defect Classification (ODC)

For the defect analysis, Orthogonal Defect Classification (ODC) will be used for classification of detected defects and two-way analysis will be performed on the classified data. Since the application involves complex features, it is important not only to detect defects but also track down their causes, impact, and distribution.

3. Test cases Development

3.1. Overview and objective of the test

The testing strategy for the video-sharing platform encompasses two distinct but complementary approaches: Finite State Machine(FSM) testing and Input Domain Partitioning and Boundary testing. Both techniques are design to cover crucial aspects of the platform and validate its functionality under a wide range of conditions.

FSM testing is used to verify state transitions. The FSM modules covered include 'Homepage and Navigation', 'Authentication', 'Search', 'Profile Management',' Video Management', and 'Video Viewing'. Each test case evaluates specific user interactions and ensures the system transitions between states properly and responds with the expected output. The focus is to validate that the platform's user interface and system functionality behave as described in the FSM models. The test also verifies cross-module transitions like logging in and the functionality of core actions such as playback, upload, record and video deletion.

Input Domain Partitioning and Boundary testing is used to validate the input fields across various parts of the platform, such as user authentication, video uploads, search functionality, and comments. The focus is to validate that the input handling mechanisms for each input field behave as expected. This includes detecting improper input handling and checking correct input process and responses with proper error messages with invalid input.

3.1. Test Cases for FSM testing

Detailed test cases are developed for each FSM module to evaluate specific user interactions, the system transitions between states, and the expected output. The key test cases representing the core

functionality of the video-sharing platform are summarized in **Table 2**. The whole set of 35 FSM test cases will be attached in the **appendix 2**:

Table 2: Key test cases for FSM testing

Case ID	Module	Description	Expected Result
H-01	Homepage	User clicks Login	Redirect to login page
H-02		User clicks Join	Redirect to join page
H-05		User clicks Upload video	Redirect to upload video page
A-01	Authentication	User enters valid credentials on login page	Redirect to logged-in homepage
A-03		User creates new account with valid details	Redirect to login page
S-01	Search	User submits valid search query	Display search results
P-01	Profile Mgmt	User updates profile with valid info	Profile updated. Redirect to edit profile page
VM-01	Video Mgmt	User upload valid video	Upload video, Redirect to logged-in homepage
VV-01	Video Viewing	User clicks play icon	Video starts playing
VV-05		User clicks 'Delete Video'	Redirect to logged-in homepage with video deleted

3.2. Test Case for Input Domain Partitioning

The test cases for Input Domain Partitioning are derived from testing points identified through domain analysis. Each testing point forms a test case, resulting in **105** total test cases generated through Weak N x 1 strategy. The number of testing cases follows (n+1) x b + 1. However, since the categorical domain (valid/invalid characters) test points cover an interior point, there is no need to include one extra interior point in the test cases. An example of test cases for the Username input field is provided in **Table 3**, with the complete set of test cases included in the **appendix 6**.

Table 3: Example test cases for Username

Case ID	Input field	Input value	Description	Expected output
TC-01	Username	Length = 5, valid characters	“abcde”	Accepted
TC-02	Username	Length = 5, valid characters	“fghij”	Accepted
TC-03	Username	Length = 4, valid characters	“abcd”	Rejected
TC-04	Username	Length=20, valid characters	“abcde12345-ABCDE1234”	Accepted
TC-05	Username	Length=20, valid characters	“ABCDE123456-abcde123”	Accepted
TC-06	Username	Length=21, valid characters	“ABCDE123456-abcde1234”	Rejected
TC-07	Username	Length=10, valid characters	“abcde12345”	Accepted
TC-08	Username	Length=16, valid characters	“abcde12345-ABCDE”	Accepted
TC-09	Username	Length=15, invalid characters	“!cde12345-ABCDE”	Rejected

4. Test Execution

4.1. Testing environment

- Web browser: Microsoft Edge
- Backend: NodeJS
- Database: MongoDB
- Platform: Localhost

4.2. Testing Methodology

Test results will be manually recorded with pass/fail criteria. A test case will be marked as pass if the actual result matches the expected result exactly. Otherwise, any deviation from the expected result will be marked as fail.

5. Test Results

The testing process focused on validating two key aspects of the video-sharing platform. Each test case followed the description section outlined in the test case, and the results were manually recorded based

on the actual behavior of the system with comparison to the expected behavior. The detailed test results are in **the appendix 3(FSM)** and **appendix 7(Input Domain Partitioning)**.

5.1. FSM Testing Results

The testing was conducted to validate the core functionalities of the video-sharing platform by simulating various user interactions based on the modular FSM models. The goal was to cover system responses to user inputs, transitions between states as expected and functional consistency.

Out of 35 test cases, 29 cases passed (denoted as 'P'), while 6 cases failed (denoted as 'F'). The failing cases were primarily related to functional issues, such as issues with GitHub authentication, handling invalid video formats, and app crashes (e.g., while playing a video without a thumbnail).

5.2. Input Domain Partitioning and Boundary Testing Results

The testing evaluated how well the platform handles different inputs within the identified input fields. The inputs were analyzed by partitioning them into valid and invalid sub-domains, focusing on boundary conditions and edge cases. The Weak N x 1 strategy was applied to identify the minimum necessary testing points while comprehensively testing each input fields.

Out of 105 test cases, 92 cases passed, while 13 cases failed. The failing cases were due to the regex incompatible to the language (pug), allowing invalid format uploads, system crashes when dealing with file uploads exceeding size limits, password change, and handling hash tags.

5.3. Combined Test Results

The combined pass rate is 86.4% (121 out of 140 test cases). The test results highlight a few critical areas of the platform that require attention. The areas include:

1. **User authentication**: Problems with GitHub integration and other user authentication mechanisms.
2. **File Upload handling**: Crashes during video and thumbnail uploads, especially with large-file handling as well as invalid format uploads.
3. **Input validation**: Problems with validating special characters, file formats, and enforcing size limits on upload files.
4. **Session logic**: Password change causes system crash due to premature session termination due to logic error.

The more detailed analysis of the test results will be explored in the Result Analysis.

6. Result analysis

6.1. Defect Classification Using ODC Framework

Using Orthogonal Defect Classification (ODC) framework, defects are classified into categories based on their *impact*, *trigger*, *source*, and *severity*. Each failed test case is analyzed and classified by the four categories. Detailed analysis of each test case is in **appendix 8**.

The identified types of impact are ***Usability***, ***Functionality***, and ***Reliability***. The severity of defects is classified into three categories depending on how much it impacts on the system: ***Critical***, ***Major***, and ***Minor***. Defect with critical severity causes system crash or failure of critical functions, and defect with Major severity affects user experience due to unexpected functionality but does not crash the system while defect with Minor severity influences user experience in minor ways but does not stop any functionality such as inconsistent UIs.

6.2. Two-way analysis

Two-way analysis is performed to analyze the distribution and criticality of defects against system functionality, which is represented in **Table 4**. The chart (**Figure 1**) places defect severity on the X-axis, and defect-impact on the Y-axis. This bubble chart presents how defects with highest severity correlate with the most impactful areas of the system. The bubble size shows the number of defects for each severity-impact combination.

Table 4: Two-way analysis results: Interactions between impact and severity

Impact	Severity		
	Critical	Major	Minor
Functionality	0	11	0
Reliability	6	0	0
Usability	0	8	2

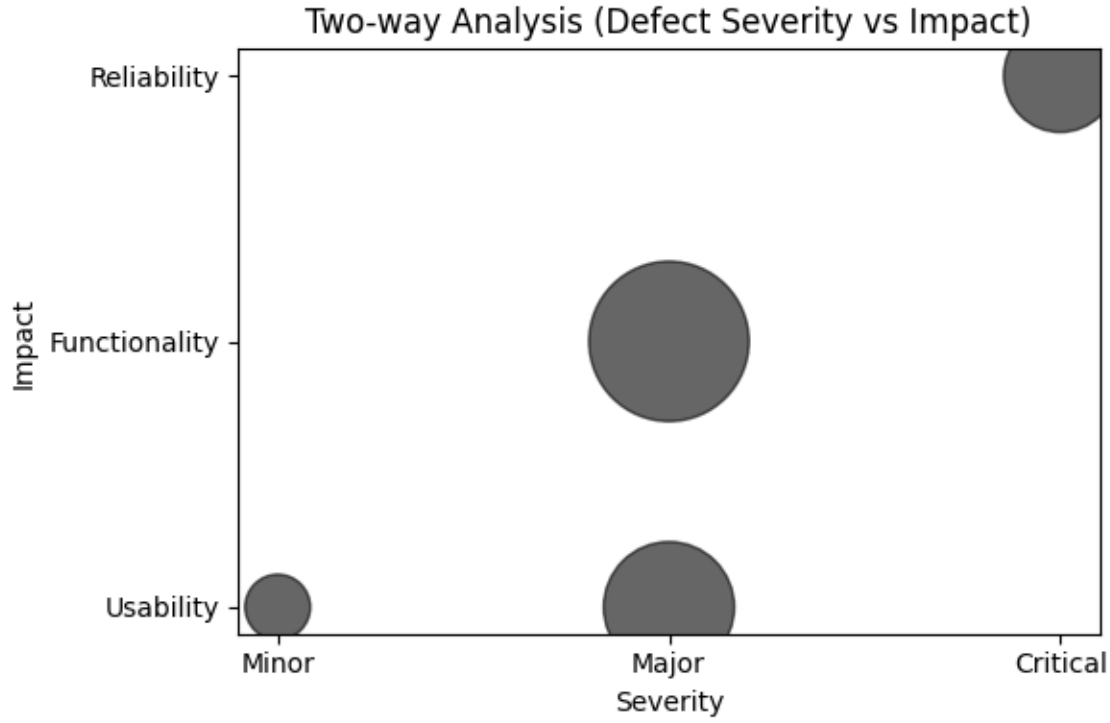

Figure 1: Two-way analysis bubble chart on defect severity against defect impact.

The two-way analysis chart illustrates that Critical defects predominantly affect Reliability, as seen in cases where app crashes when file size exceeds the limits, password change, viewing videos without a thumbnail while Major defects tend to impact Functionality, especially with improper input validation. Minor defects, on the other hand, are mostly related to Usability, such as the absence of a default display on certain pages.

7. Lesson learned

The testing and analysis of the video-sharing platform shows several key insights regarding the system's functionality, reliability and user experience.

First of all, reliability issues occur at high-risk areas such as file upload, change password, and playback. Media playback, uploading video, and user management are core functions of the video-sharing platform. App crashes at those functions emphasizes the need for enhanced error handling mechanisms.

Secondly, input validation is crucial for preventing system crashes. A significant number of failures are detected from improper input validations in areas such as handling files exceeding size limits. Such app crashes could have been prevented in advance with defect containment strategies such as error handling, lowering the severity of the defects with increased reliability.

Third, the third-party integration requires testing in multiple environments. Defects related to GitHub authentication highlights the importance of testing in different environments such as production-like conditions. During the test process, limitations in the test environment prevented full validation of external integrations.

Lastly, minor details that significantly impact user experience, such as incompatible regex expressions in pug template, could have been effectively blocked with error source removal strategies such as proper education on the language in the beginning of development.

8. Follow-up activity

From the lessons learned from this testing process, follow-up activities are performed to ensure comprehensive coverage of the video-sharing platform.

Immediate actions required to address the critical defects causing system crashes have been performed. File type and size filter has been implemented in Middleware with better error-handling mechanisms to deal with system crashes and acceptance of invalid file types related to file uploads.

Given the numerous issues related to input validation, the input validation mechanisms at both front-end and back-end level have been revised. In addition, regular expressions are modified to be compatible with the front-end rendering framework (pug).

To address the logic errors observed in session handling during password change and Github authentication, session management logic is refactored so that the user sessions are not prematurely destroyed before necessary actions. In addition, integration test with Github in production-like environment (Fly.io) has been conducted to resolve authentication issues.

In addition, minor issues such as missing default display on the search page and no reset on the completion of viewing a video have been handled by implementing a default display for the search page and reset logic to the video player.

Once all the follow-up activities have been implemented, a comprehensive retest of the application is performed. All relevant test cases from both models are re-executed. The results are in the **appendix 9**.

There was one test case that did not pass the re-test, which had been fixed and gone through the second re-test to pass.

9. Conclusion

The application of FSM and Input Domain Partitioning with Boundary Testing provided comprehensive coverage of the video-sharing platform's functionalities and input handling mechanisms. The testing identified critical defects that were subsequently addressed, enhancing the platform's reliability and usability. The project highlights the importance of formal testing methodologies in software development, particularly for complex applications involving user interactions and file handling.

10. Summary

This project focused on the formal testing of a video-sharing platform that did not undergo structured testing process. Recognizing potential risks associated with untested functionalities like video upload and playback, two black-box testing techniques were employed: Finite State Machine Testing and Input Domain Partitioning and Boundary Testing. FSM was to verify the correctness of state transitions within and between the platform's modules. Input Domain Partitioning and Boundary Testing was to evaluate the platform's input handling by partitioning input fields into sub-domains and testing boundary conditions to detect defects related to input validation.

The testing revealed several defects, which were classified using the Orthogonal Defect Classification (ODC) framework. A two-way analysis provided insights into their impact and severity, clearly displaying critical areas affecting reliability and functionality. Follow-up activities addressed the identified defects through implementing error handlings, input validation mechanisms, and logic modifications. A series of retest confirmed the resolution of the issues. This project demonstrates the effectiveness of formal

testing methodologies in improving software reliability and proves the importance of systematic testing in software development.

Appendix

1. FSM model

1. Homepage and Navigation module(H)

Current state	Input	Next state	Output
H1(logged-out)	User clicks 'Login'	Login page(A1)	Redirect to login page
H1(logged-out)	User clicks 'Join'	Join page(A3)	Redirect to Join page
H1(logged-out)	User clicks search icon	Search page(S1)	Redirect to Search page
H1(logged-out)	User clicks video	VV1(logged-out)	Redirect to Video Viewing page
H1(logged-out)	User clicks logo	H1(logged-out)	Redirect to logged-out Home(H1)
H2(logged-in)	User clicks logo	H2(logged-in)	Redirect to logged-in Home(H2)
H2(logged-in)	User clicks search icon	S4(logged-in)	Redirect to logged-in Search page
H2(logged-in)	User clicks 'Upload Video'	Upload page(VM1)	Redirect to Video Upload page
H2(logged-in)	User clicks 'Edit Profile'	Edit Profile page(P1)	Redirect to Edit Profile page
H2(logged-in)	User clicks avatar icon	My Video page(VM3)	Redirect to My Video page
H2(logged-in)	User clicks video	VV2(logged-in)	Redirect to Video Viewing page
H2(logged-in)	User clicks 'Log out'	H1(logged-out)	Redirect to logged out Home(H1)

2. Authentication Module

Current state	Input	Next state	Output
A1(Login Page)	User select 'Login with Github'	GitHub Auth(A2)	Redirect to GitHub authentication page
A2(GitHub Auth)	Github authentication succeeds	H2(logged-in)	Redirect to logged-in Home(H2)
A2(Github Auth)	Github authentication fails	Login Page(A1)	Redirect to login page
A1(Login Page)	User provide valid id and password	H2(logged-in)	Redirect to logged-in Home(H2)
A1(Login Page)	User provide invalid id and password	Login Page(A1)	Redirect to login page with error message
A3(Join Page)	User provide valid account details	Login Page(A1)	Redirect to login page
A3(Join Page)	User provide invalid account details	Join Error	Show error message on Join page

3. Search Module

Current state	Input	Next state	Output
S1(Search Page)	User submits a valid search query	Search Results(S2)	Display search results
S1(Search Page)	User submits a search query with no results	No Search Results(S3)	Show “No Video Found” Message
S2(Search Results)	User clicks a video	VV1(logged-out)	Redirect to Video Viewing page
S4(Logged-in Search Page)	User clicks a video	VV2(logged-in)	Redirect to Video Viewing page

4. Profile management module

Current state	Input	Next state	Output
P1(Edit Profile page)	User clicks ‘Update Profile’	Edit Profile page(P1)	Redirect to Edit Profile page with updated information
P1(Edit Profile page)	User clicks ‘Change Password’	Change Password page(P2)	Redirect to Change Password page
P2(Change Password page)	User submits valid password change	H1(logged-out)	Redirect to logged out Home(H1)
P2(Change Password page)	User submits invalid password change	Password change error	Show error message on Change Password page

5. Video Management Module

Current state	Input	Next state	Output
VM1(Upload page)	User uploads a video successfully	H2(logged-in)	Redirect to logged-in Home(H2)
VM1(Upload page)	User submits invalid video upload	Video Upload Error	Show error message on Upload Video page
VM1(Upload page)	User clicks ‘Start Recording’	Recording(VM1)	Start Recording for 5 seconds and inner text on button changes to ‘Download’
VM1(Recording)	User clicks ‘Download’	Download(VM1)	Transcode and download video and thumbnail image
VM2(Edit Video)	User edits video successfully	VV2(logged-in)	Redirect to Video Viewing page with success message
VM2(Edit Video)	User submits invalid video edits	Video Edit Error	Show error message on Edit video page
VM3(My video Page)	User clicks a video	VV2(logged-in)	Redirect to Video Viewing page

6. Video Viewing Module

Current state	Input	Next state	Output
VV1 (logged-out)	User plays video	Video Playing(VV1)	Video starts playing
VV1 (play)	User pauses video	VV1 (Video Paused)	Video pauses
VV2(logged-in)	User adds a comment	VV2(logged-in)	Comment is added successfully
VV2(logged-in)	User adds an invalid comment	Video viewing error message	Show error message on video viewing page
VV2(logged-in)	User clicks 'Edit Video'	Edit Video(VM2)	Redirect to Edit Video
VV2(logged-in)	User clicks 'Delete video'	H2(logged-in)	Delete video and redirect to logged-in Home(H2)

2. Test Case for FSM model

Case ID	Case	Condition	Input	Expected Result
H-01	Login from homepage	User logged out	User clicks 'Login'	Redirects to login page
H-02	Join from homepage	User logged out	User clicks 'Join'	Redirect to join page
H-03	Search from logged out homepage	User logged out	User clicks the search icon	Redirect to search page
H-04	View a video from logged out homepage	User logged out	User clicks on a video	Redirect to video viewing page
H-05	Navigate to upload page	User logged in	User clicks 'Upload Video'	Redirect to video upload page
H-06	Navigate to edit profile	User logged in	User clicks 'Edit profile'	Redirect to edit profile page
H-07	Navigate to my video	User logged in	User clicks avatar	Redirect to my video page
H-08	Log out	User logged in	User clicks 'Log out'	Redirect to logged out homepage with a message
H-9	Back to homepage	Other states	User clicks logo	Redirect to homepage
A-01	Login with valid credentials	User logged out on login page	User enters valid credentials and click 'Login'	Redirects to logged-in homepage
A-02	Login with invalid credentials	User logged out on login page	User enters invalid credentials and click 'Login'	Remain on login page with an error message
A-03	Create a new account with valid details	User logged out on join page	User enters valid details and click 'Join'	Redirect to login page
A-04	Create a new account with invalid details	User logged out on join page	User enters invalid details and click 'Join'	Remain on join page with an error message
A-05	Login success with Github authentication	User logged out on login page	User clicks 'Continue with Github' and succeeds in authentication	Redirect to logged in homepage
A-06	Login failure with Github authentication	User logged out on login page	User clicks 'Continue with Github' and fails in authentication	Redirect to login page

Case ID	Case	Condition	Input	Expected Result
S-01	Submit valid search query	User logged out on search page	User submits a query	Display search results
S-02	Submit query with no results	User on search page	User submits a query with no results	Display “No Video Found”
S-03	View video from search results	User logged in on search results	User clicks a video from search results	Redirect to video viewing page(logged in)
S-04	View video from search results	User logged out on search results	User clicks a video from search results	Redirect to video viewing page(logged out)
P-01	Update profile with valid information	User on edit profile page	User updates with valid information	Remain on edit profile page. Profile updated
P-02	Navigate to change password	User on edit profile page	User clicks ‘Change Password’	Redirect to change password page
P-03	Change password with valid input	User on change password page	User submits valid password change	Redirect to logged out homepage
P-04	Change password with invalid input	User on change password page	User submits invalid password change	Remain on change password page with an error message
VM-01	Upload valid video	User on video upload page	Upload valid video	Upload video. Redirect to logged in homepage
VM-02	Upload invalid video	User on video upload page	Upload invalid video	Remain on upload page with an error message
VM-03	Start and Complete recording	User on video upload page	User clicks ‘start recording’, record for 5 sec, user download recording	Transcode and download recording and thumbnail image
VM-04	Edit video metadata successful	User on edit video page	User submits valid video metadata	Redirect to video viewing page with updated metadata
VM-05	Edit video metadata with invalid input	User on edit video page	User submits invalid video metadata	Remain on edit video page with error message
VM-06	View video from my video page	User on my video page	User clicks on video	Redirect to video viewing page(logged in)
VV-01	Play video	User on video viewing page	User clicks play icon	Video starts playing
VV-02	Pause video	User on video viewing page	User clicks pause icon	Video pauses
VV-03	Adding valid comment	User logged in on video viewing page	User adds a valid comment	Remain on video viewing page with comment displayed
VV-04	Adding invalid comment	User logged in on video viewing page	User adds invalid comment	Invalid input cannot be entered
VV-05	Delete video	User logged in on video viewing page	User clicks ‘Delete video’	Redirect to logged in homepage with video deleted
VV-06	Delete comment	User logged in on video viewing page	User clicks ‘X’ mark beside comment	Comment deleted from display

3. Test Results for FSM

Case ID	Input	Expected Result	Actual Result	Status	Note
H-01	User clicks 'Login'	Redirects to login page	Redirects to login page	P	
H-02	User clicks 'Join'	Redirect to join page	Redirect to join page	P	
H-03	User clicks the search icon	Redirect to search page	Redirect to search page	F	Default Search page not set up('No Video Found' display)
H-04	User clicks on a video	Redirect to video viewing page	Redirect to video viewing page	P	
H-05	User clicks 'Upload Video'	Redirect to video upload page	Redirect to video upload page	P	
H-06	User clicks 'Edit profile'	Redirect to edit profile page	Redirect to edit profile page	P	
H-07	User clicks avatar	Redirect to my video page	Redirect to my video page	P	
H-08	User clicks 'Log out'	Redirect to logged out homepage with a message	Redirect to logged out homepage with a message	P	
H-09	User clicks logo	Redirect to homepage	Redirect to homepage	P	
A-01	User enters valid credentials and click 'Login'	Redirects to logged-in homepage	Redirects to logged-in homepage	P	
A-02	User enters invalid credentials and click 'Login'	Remain on login page with an error message	Remain on login page with an error message	P	
A-03	User enters valid details and click 'Join'	Redirect to login page	Redirect to login page	P	
A-04	User enters invalid details and click 'Join'	Remain on join page with an error message	Remain on join page with an error message	P	
A-05	User clicks 'Continue with Github' and succeeds in authentication	Redirect to logged in homepage	Test environment does not allow redirection to gitHub	F	
A-06	User clicks 'Continue with Github' and fails in authentication	Redirect to login page	Test environment does not allow redirection to gitHub	F	
S-01	User submits a query	Display search results	Display search results	P	
S-02	User submits a query with no results	Display "No Video Found"	Display "No Video Found"	P	
S-03	User clicks a video from search results	Redirect to video viewing page(logged in)	Redirect to video viewing page(logged in)	P	
S-04	User clicks a video from search results	Redirect to video viewing page(logged out)	Redirect to video viewing page(logged out)	P	
P-01	User updates with valid information	Remain on edit profile page. Profile updated	Remain on edit profile page. Profile updated	P	

Case ID	Input	Expected Result	Actual Result	Status	Note
P-02	User clicks 'Change Password'	Redirect to change password page	Redirect to change password page	P	
P-03	User submits valid password change	Redirect to logged out homepage	App crash due to logical defect in code	F	
P-04	User submits invalid password change	Remain on change password page with an error message	Remain on change password page with an error message	P	
VM-01	Upload valid video	Upload video. Redirect to logged in homepage	Upload video. Redirect to logged in homepage	P	
VM-02	Upload invalid video	Remain on upload page with an error message	Upload video. Redirect to logged in homepage	F	Allow invalid format for upload(functional defect)
VM-03	User clicks 'start recording', record for 5 sec, user download recording	Transcode and download recording and thumbnail image	Transcode and download recording and thumbnail image	P	
VM-04	User submits valid video metadata	Redirect to video viewing page with updated metadata	Redirect to video viewing page with updated metadata	P	
VM-05	User submits invalid video metadata	Remain on edit video page with error message	Remain on edit video page with error message	P	
VM-06	User clicks on video	Redirect to video viewing page(logged in)	Redirect to video viewing page(logged in)	P	
VV-01	User clicks play icon	Video starts playing	Video starts playing	F	1.App crash after playing video without thumbnail(logical defect in the code) 2.Playback does not initialize after the completion of viewing.
VV-02	User clicks pause icon	Video pauses	Video pauses	P	
VV-03	User adds a valid comment	Remain on video viewing page with comment displayed	Remain on video viewing page with comment displayed	P	
VV-04	User adds invalid comment	Invalid input cannot be entered	Invalid input cannot be entered	P	
VV-05	User clicks 'Delete video'	Redirect to logged in homepage with video deleted	Redirect to logged in homepage with video deleted	P	
VV-06	User clicks 'X' mark beside comment	Comment deleted from display	Comment deleted from display	P	

4. Sub-domain and domain analysis

Domain	Constraint	Sub domain	
		Valid	Invalid
Username	Length: [5, 20] Text: [a-zA-Z_-]	1. Length [5, 20] 2. [a-zA-Z_-]	1. Length < 5 2. Length > 20 3. Illegal characters
Password	Length: [10, 20] Text: at least one uppercase and one special character	1. Length [10, 20] 2. Password with at least one uppercase and a special char.	1. Length <10 2. Length > 20 3. Missing uppercase or special character
Name	Length: [2, 30] Text: [a-zA-Z\s]	1. Length [2,30] 2. [a-zA-Z\s]	1. Length < 2 2. Length > 30 3. Illegal characters
Email	Format: email Length: ≤50	1. valid email format	1. invalid email format 2. length > 50
Location	Length: [2, 50] Text: [a-zA-Z0-9\s,]	1. Length [2,50] 2. [a-zA-Z0-9\s,]	1. Length < 2 2. Length > 50 3. Illegal characters
Avatar	Format: image Size max: 3MB	1. Image format 2. Size ≤ 3MB	1. Other format 2. Size > 3MB
Title	Length: [3, 80] Text: [a-zA-Z09\s_-]	1. Length [3, 80] 2. [a-zA-Z09\s_-]	1. Length < 3 2. Length > 80 3. Illegal character
Description	Length: [20, 255]	1. Length [20, 255]	1. Length < 20 2. Length > 255
Hash tags	Length: [1,200] Hashlength: [1,20] Max hashtags: 10 Text: [a-zA-Z09]	1. Length [1,200] 2. Hash length:[1,20] 3. Num of hash:[1, 10] 4. [a-zA-Z09]	1. No hash 2. Length > 200 3. Hash length > 20 4. Num of hash > 10 5. Illegal characters
Video	Format: video Size max: 10MB	1. Video format 2. Size ≤ 10MB	1. Other format 2. Size > 10MB
Thumbnail	Format: image Size max: 3MB	1. Image format 2. Size ≤ 3MB	1. Other format 2. Size > 3MB
Search	Length: [3, 80] Text: [a-zA-Z09\s]	1. 1. Length [3, 80] 2. [a-zA-Z09\s]	1. Length < 3 2. Length > 80 3. Illegal character
Comment	Length: [5, 150]	1. Length [5, 150]	1. Length > 150 2. Length < 5

5. Testing points from Domain Analysis

Input field	Testing point	Type
Username	Length = 5, valid characters	ON point(min, valid)
	Length = 5, valid characters	ON point(min, valid)
	Length = 4, valid characters	OFF point(**under**, valid)
	Length = 20, valid characters	ON point(max, valid)
	Length = 20, valid characters	ON point(max, valid)
	Length = 21, valid characters	OFF point(**over**, valid)
	Length = 10, valid characters	ON point(valid, valid)
	Length = 16, valid characters	ON point(valid, valid)
	Length = 15, invalid characters	OFF point(valid, **invalid**)
Password	Length = 10, valid characters	ON point(min, valid)
	Length = 10, valid characters	ON point(min, valid)
	Length = 9, valid characters	OFF point(**under**, valid)
	Length = 20, valid characters	ON point(max, valid)
	Length = 20, valid characters	ON point(max, valid)
	Length = 21, valid characters	OFF point(**over**, valid)
	Length = 13, valid characters	ON point(valid, valid)
	Length = 18, valid characters	ON point(valid, valid)
	Length = 16, invalid characters	OFF point(valid, **invalid**)
Name	Length = 2, valid characters	ON point(min, valid)
	Length = 2, valid characters	ON point(min, valid)
	Length = 1, valid characters	OFF point(**under**, valid)
	Length = 30, valid characters	ON point(max, valid)
	Length = 30, valid characters	ON point(max, valid)
	Length = 31, valid characters	OFF point(**over**, valid)
	Length = 10, valid characters	ON point(valid, valid)
	Length = 20, valid characters	ON point(valid, valid)
	Length = 17, invalid characters	OFF point(valid, **invalid**)
Email	Length = 50, valid format	ON point(max, valid)
	Length = 50, valid format	ON point(max, valid)
	Length = 51, valid format	OFF point(**over**, valid)
	Length = 20, valid format	ON point(valid, valid)
	Length = 40, valid format	ON point(valid, valid)
	Length = 30. Invalid format	OFF point(valid, **invalid**)

Input field	Testing point	Type
Location	Length = 2, valid characters	ON point(min, valid)
	Length = 2, valid characters	ON point(min, valid)
	Length = 1, valid characters	OFF point(**under**, valid)
	Length = 50, valid characters	ON point(max, valid)
	Length = 50, valid characters	ON point(max, valid)
	Length = 51, valid characters	OFF point(**over**, valid)
	Length = 20, valid characters	ON point(valid, valid)
	Length = 40, valid characters	ON point(valid, valid)
	Length = 30, invalid characters	OFF point(valid, **invalid**)
Avatar	Size = 3MB, valid format	ON point(max, valid)
	Size = 3MB, valid format	ON point(max, valid)
	Size = 3.1MB, valid format	OFF point(**over**, valid)
	Size = 1.5MB, valid format	ON point(valid, valid)
	Size = 2.5MB, valid format	ON point(valid, valid)
	Size = 2MB, invalid format	OFF point(valid, **invalid**)
Title	Length = 3, valid characters	ON point(min, valid)
	Length = 3, valid characters	ON point(min, valid)
	Length = 2, valid characters	OFF point(**under**, valid)
	Length = 80, valid characters	ON point(max, valid)
	Length = 80, valid characters	ON point(max, valid)
	Length = 81, valid characters	OFF point(**over**, valid)
	Length = 20, valid characters	ON point(valid, valid)
	Length = 50, valid characters	ON point(valid, valid)
	Length = 40, invalid characters	OFF point(valid, **invalid**)
Description	Length = 20	ON point(min)
	Length = 255	ON point(max)
	Length = 19	OFF point(below min)
	Length = 256	OFF point(above max)
	Length = 100	Interior point
Hash tags	Length = 1, valid char, valid hash num, valid hash len	ON point(min, valid)
	Length = 1, valid char, valid hash num, valid hash len	ON point(min, valid)
	No hash	OFF point(**under of all**)
	Length = 200, valid char, valid	ON point(max, valid)

Input field	Testing point	Type
	hash num, valid hash len	
	Length = 200, valid char, valid hash num, valid hash len	ON point(max, valid)
	Length = 201, valid char, valid hash num, valid hash len	OFF point(**over**, valid)
	Hash Length = 20, valid char, valid hash num, valid len	ON point(max, valid)
	Hash Length = 20, valid char, valid hash num, valid len	ON point(max, valid)
	Hash Length = 21, valid char, valid hash num, valid len	OFF point(**over**, valid)
	Num of hash = 1, valid char, valid len, valid hash len	ON point(min, valid)
	Num of hash = 1, valid char, valid len, valid hash len	ON point(min, valid)
	Num of hash = 10, valid char, valid len, valid hash len	ON point(max, valid)
	Num of hash = 10, valid char, valid len, valid hash len	ON point(max, valid)
	Num of hash = 11, valid char, valid len, valid hash len	OFF point(**over**, valid)
	Valid char, valid len, valid hash len, valid hash num	ON point(valid, valid)
	Diff Valid char, valid len, valid hash len, valid hash num	ON point(valid, valid)
	Inalid char, valid len, valid hash len, valid hash num	OFF point(valid, **invalid**)
Video	Size = 10MB, valid format	ON point(max, valid)
	Size = 10MB, valid format	ON point(max, valid)
	Size = 10.1MB, valid format	OFF point(**over**, valid)
	Size = 3MB, valid format	ON point(valid, valid)
	Size = 8MB, valid format	ON point(valid, valid)
	Size = 5MB, invalid format	OFF point(valid, **invalid**)
Thumbnail	Size = 3MB, valid format	ON point(max, valid)
	Size = 3MB, valid format	ON point(max, valid)
	Size = 3.1MB, valid format	OFF point(**over**, valid)

Input field	Testing point	Type
	Size = 1MB, , valid format	ON point(valid, valid)
	Size = 2.5MB, valid format	ON point(valid, valid)
	Size = 2MB, invalid format	OFF point(valid, **invalid**)
Search	Length = 3, valid characters	ON point(min, valid)
	Length = 3, valid characters	ON point(min, valid)
	Length = 2, valid characters	OFF point(**under**, valid)
	Length = 80, valid characters	ON point(max, valid)
	Length = 80, valid characters	ON point(max, valid)
	Length = 81, valid characters	OFF point(**over**, valid)
	Length = 20, valid characters	ON point(valid, valid)
	Length = 40, valid characters	ON point(valid, valid)
	Length = 30, invalid characters	OFF point(valid, **invalid**)
Comment	Length = 5	ON point(min)
	Length = 150	ON point(max)
	Length = 4	OFF point(below min)
	Length = 151	OFF point(above max)
	Length = 50	Interior point

6. Test Cases for Input domain boundary testing

Case ID	Input field	Test point	Input Description	Expected output
TC-01	Username	Length = 5, valid characters	“abcde”	Accepted
TC-02	Username	Length = 5, valid characters	“fghij”	Accepted
TC-03	Username	Length = 4, valid characters	“abcd”	Rejected
TC-04	Username	Length = 20, valid characters	“abcde12345-ABCDE1234”	Accepted
TC-05	Username	Length = 20, valid characters	“ABCDE123456-abcde123”	Accepted
TC-06	Username	Length = 21, valid characters	“ABCDE123456-abcde1234”	Rejected
TC-07	Username	Length = 10, valid characters	“abcde12345”	Accepted
TC-08	Username	Length = 16, valid characters	“abcde12345-ABCDE”	Accepted
TC-09	Username	Length = 15, invalid characters	“!cde12345-ABCDE”	Rejected
TC-10	Password	Length = 10, valid characters	“Passw@1234”	Accepted
TC-11	Password	Length = 10, valid characters	“pASSW@1234”	Accepted
TC-12	Password	Length = 9, valid characters	“Passw@123”	Rejected
TC-13	Password	Length = 20, valid characters	“Passwordtest!1234567”	Accepted
TC-14	Password	Length = 20, valid characters	“passwordTEST!1234567”	Accepted
TC-15	Password	Length = 21, valid characters	“Passwordtest!12345678”	Rejected
TC-16	Password	Length = 13, valid characters	“PasswordTest!”	Accepted
TC-17	Password	Length = 18, valid characters	“PasswordTest!12345”	Accepted
TC-18	Password	Length = 16, invalid characters	“passwordtest1123”	Rejected
TC-19	Name	Length = 2, valid characters	“ab”	Rejected
TC-20	Name	Length = 2, valid characters	“cd”	Accepted
TC-21	Name	Length = 1, valid characters	“A”	Rejected

Case ID	Input field	Test point	Input Description	Expected output
TC-22	Name	Length = 30, valid characters	“Last name First name Middlenam”	Accepted
TC-23	Name	Length = 30, valid characters	“First name Last name Middlenam”	Accepted
TC-24	Name	Length = 31, valid characters	“First name Last name Middlename”	Rejected
TC-25	Name	Length = 10, valid characters	“First name”	Accepted
TC-26	Name	Length = 20, valid characters	“First name last name”	Accepted
TC-27	Name	Length = 17, invalid characters	“First last mid123”	Rejected
TC-28	Email	Length = 50, valid format	“emailtest1emailtest2emailtest3emailtest4@valid.com”	Accepted
TC-29	Email	Length = 50, valid format	“emailtest1emailtest2emailtest3emailtest4@email.com”	Accepted
TC-30	Email	Length = 51, valid format	“emailtest1emailtest2emailtest3emailtest4@invalid.kr”	Rejected
TC-31	Email	Length = 20, valid format	“emailtest1@valid.com”	Accepted
TC-32	Email	Length = 40, valid format	“emailtest1emailtest2emailtest3@valid.com”	Accepted
TC-33	Email	Length = 30, Invalid format	“emailtest1emailtest2invalid.co”	Rejected
TC-34	Email	Length = 2, valid characters	“WA”	Accepted
TC-35	Email	Length = 2, valid characters	“TX”	Accepted
TC-36	Location	Length = 1, valid characters	“A”	Rejected
TC-37	Location	Length = 50, valid characters	“street name, unit, city, state, country, zip code,”	Accepted
TC-38	Location	Length = 50, valid characters	“street name, city, county, state, country, zip cod”	Accepted
TC-39	Location	Length = 51, valid characters	“street name, city, county, state, country, zip code”	Rejected
TC-40	Location	Length = 20, valid characters	“street, city, state,”	Accepted
TC-41	Location	Length = 40, valid characters	“streetname, city, county, state, country”	Accepted
TC-42	Location	Length = 30, invalid characters	“street#, unit#, state, country”	Rejected
TC-43	Avatar	Size = 3MB, valid format	3MB image file(JPG)	Accepted
TC-44	Avatar	Size = 3MB, valid format	3MB image file(PNG)	Accepted
TC-45	Avatar	Size = 3.1MB, valid format	3.1MB image file(JPG)	Rejected
TC-46	Avatar	Size = 1.5MB, valid format	1.5MB image file(JPG)	Accepted
TC-47	Avatar	Size = 2.5MB, valid format	2.5MB image file(JPG)	Accepted

Case ID	Input field	Test point	Input Description	Expected output
TC-48	Avatar	Size = 2MB, invalid format	2MB video file(MP4)	Rejected
TC-49	Title	Length = 3, valid characters	“abc”	Accepted
TC-50	Title	Length = 3, valid characters	“def”	Accepted
TC-51	Title	Length = 2, valid characters	“ab”	Rejected
TC-52	Title	Length = 80, valid characters	“Video title example with maximum characters allowed Video title example with max”	Accepted
TC-53	Title	Length = 80, valid characters	“second title example with maximum characters allowed Video title example with ma”	Accepted
TC-54	Title	Length = 81, valid characters	“over length title example with over length characters allowed Video title exampl”	Rejected
TC-55	Title	Length = 20, valid characters	“on-point example for”	Accepted
TC-56	Title	Length = 50, valid characters	“another on point 123”	Accepted
TC-57	Title	Length = 40, invalid characters	“Video title example with special char!@#”	Rejected
TC-58	Description	Length = 20	“is valid description”	Accepted
TC-59	Description	Length = 255	“Exploring the endless potential of software testing, this project delves into comprehensive methods such as finite-state machine testing, boundary partitioning, and defect analysis to ensure robust performance across critical components of a video-sharing”	Accepted
TC-60	Description	Length = 19	“invalid description”	Rejected
TC-61	Description	Length = 256	“Exploring the endless potential of software testing, this project delves into comprehensive methods such as finite-state machine testing, boundary partitioning, and defect analysis to ensure robust performance across critical components of a video-sharing.”	Rejected
TC-62	Description	Length = 100	“This is testing the reliability of a video-sharing platform using FSM and input boundary techniques.”	Accepted
TC-63	Hash tags	Length = 1, valid char, valid hash num, valid hash len	“a”	Accepted

Case ID	Input field	Test point	Input Description	Expected output
TC-64	Hash tags	Length = 1, valid char, valid hash num, valid hash len	“b”	Accepted
TC-65	Hash tags	No hash	“”	Rejected
TC-66	Hash tags	Length = 200, valid char, valid hash num, valid hash len	“#SoftwareTestingprog, #QualityAssurancesti, #BugTrackingprogam, #Automationprogram, #RegressionTesting, #FiniteStateMachine, #BoundaryTesting, #DefectAnalysis, #CodeCoveragetest, #TestDrivenDevelopmen”	Accepted
TC-67	Hash tags	Length = 200, valid char, valid hash num, valid hash len	“#SoftwareTestingprog, #QualityAssurancesti, #BugTrackingprogam, #Automationprogram, #RegressionTesting, #FiniteStateMachine, #BoundaryTesting, #DefectAnalysis, #CodeCoveragetest, #Thisisadifferenthash”	Accepted
TC-68	Hash tags	Length = 201, valid char, valid hash num, valid hash len	“#SoftwareTestingprog, #QualityAssurancesti, #BugTrackingprogam, #Automationprogram, #RegressionTesting, #FiniteStateMachine, #BoundaryTesting, #DefectAnalysis, #CodeCoveragetest, #Thisviolateslength”	Rejected
TC-69	Hash tags	Hash Length = 20, valid char, valid hash num, valid len	“#hashlengthistwenty”	Accepted
TC-70	Hash tags	Hash Length = 20, valid char, valid hash num, valid len	“#thisonesalsotwenty”	Accepted
TC-71	Hash tags	Hash Length = 21, valid char, valid hash num, valid len	“#twentyoneistooolong”	Rejected
TC-72	Hash tags	Num of hash = 1, valid char, valid len, valid hash len	“#a”	Accepted
TC-73	Hash tags	Num of hash = 1, valid char, valid len, valid hash len	“#b”	Accepted
TC-74	Hash tags	Num of hash = 10, valid char, valid len, valid hash len	“#A, #B, #C, #D, #E, #F, #G, #H, #I, #J”	Accepted
TC-75	Hash tags	Num of hash = 10, valid char, valid len, valid hash len	“#K, #L, #M, #N, #O, #P, #Q, #R, #S, #T”	Accepted
TC-76	Hash tags	Num of hash = 11, valid char, valid len, valid hash len	“#A, #B, #C, #D, #E, #F, #G, #H, #I, #J, #K”	Rejected
TC-77	Hash tags	Valid char, valid len, valid hash len, valid hash num	“#this, #hashtag, #is,#correct”	Accepted
TC-78	Hash tags	Valid char, valid len, valid hash len, valid hash num	“#this, #hash, #is, #valid”	Accepted

Case ID	Input field	Test point	Input Description	Expected output
TC-79	Hash tags	Invalid char, valid len, valid hash len, valid hash num	“#hashtag@!”	Rejected
TC-80	Video	Size = 10MB, valid format	10MB video file(MP4)	Accepted
TC-81	Video	Size = 10MB, valid format	10MB video file (MPEG)	Accepted
TC-82	Video	Size = 10.1MB, valid format	10.1MB video file(MP4)	Rejected
TC-83	Video	Size = 3MB, valid format	3MB video file(MP4)	Accepted
TC-84	Video	Size = 8MB, valid format	8MB video file(MP4)	Accepted
TC-85	Video	Size = 5MB, invalid format	5MB image file(JPG)	Rejected
TC-86	Thumbnail	Size = 3MB, valid format	3MB image file(JPG)	Accepted
TC-87	Thumbnail	Size = 3MB, valid format	3MB image file(PNG)	Accepted
TC-88	Thumbnail	Size = 3.1MB, valid format	3.1MB image file(JPG)	Rejected
TC-89	Thumbnail	Size = 1MB, , valid format	1MB image file(JPG)	Accepted
TC-90	Thumbnail	Size = 2.5MB, valid format	2.5MB image file(JPG)	Accepted
TC-91	Thumbnail	Size = 2MB, invalid format	2MB video file(MP4)	Rejected
TC-92	Search	Length = 3, valid characters	“abc”	Accepted
TC-93	Search	Length = 3, valid characters	“def”	Accepted
TC-94	Search	Length = 2, valid characters	“ab”	Rejected
TC-95	Search	Length = 80, valid characters	“Software testing quality assurance bug tracking automation FSM boundary testing1”	Accepted
TC-96	Search	Length = 80, valid characters	“software quality assurance bug tracking testing automation FSM boundary testing2”	Accepted
TC-97	Search	Length = 81, valid characters	“Software testing quality assurance bug tracking automation FSM boundary testing12”	Rejected
TC-98	Search	Length = 20, valid characters	“Software qual assure”	Accepted
TC-99	Search	Length = 40, valid characters	“Software quality assurance bug tracking3”	Accepted
TC-100	Search	Length = 30, invalid characters	“Software quality assurance !@#”	Rejected
TC-101	Comment	Length = 5	“abcde”	Accepted
TC-102	Comment	Length = 150	“This platform offers an intuitive interface for uploading, watching, and managing videos. I appreciate the smooth performance and the various features”	Accepted
TC-103	Comment	Length = 4	“abcd”	Rejected
TC-104	Comment	Length = 151	“This platform offers an intuitive interface for uploading, watching, and managing videos. I appreciate the smooth performance and the various features.”	Rejected
TC-105	Comment	Length = 50	“This platform offers an intuitive interface for up”	Accepted

7. Testing results for Input Domain Partitioning and Boundary Testing

Case ID	Input field	Test point	Expected Result	Actual Result	Status	Note
TC-01	Username	Length = 5, valid characters	Accepted	Accepted	P	
TC-02	Username	Length = 5, valid characters	Accepted	Accepted	P	
TC-03	Username	Length = 4, valid characters	Rejected	Rejected	P	
TC-04	Username	Length = 20, valid characters	Accepted	Accepted	P	
TC-05	Username	Length = 20, valid characters	Accepted	Accepted	P	
TC-06	Username	Length = 21, valid characters	Rejected	Rejected	P	
TC-07	Username	Length = 10, valid characters	Accepted	Accepted	P	
TC-08	Username	Length = 16, valid characters	Accepted	Accepted	P	
TC-09	Username	Length = 15, invalid characters	Rejected	Rejected	F	Incompatible Regex
TC-10	Password	Length = 10, valid characters	Accepted	Accepted	P	
TC-11	Password	Length = 10, valid characters	Accepted	Accepted	P	
TC-12	Password	Length = 9, valid characters	Rejected	Rejected	F	Incompatible Regex
TC-13	Password	Length = 20, valid characters	Accepted	Accepted	P	
TC-14	Password	Length = 20, valid characters	Accepted	Accepted	P	
TC-15	Password	Length = 21, valid characters	Rejected	Rejected	P	
TC-16	Password	Length = 13, valid characters	Accepted	Accepted	P	
TC-17	Password	Length = 18, valid characters	Accepted	Accepted	P	
TC-18	Password	Length = 16, invalid characters	Rejected	Rejected	P	
TC-19	Name	Length = 2, valid characters	Accepted	Accepted	P	
TC-20	Name	Length = 2, valid characters	Accepted	Accepted	P	
TC-21	Name	Length = 1, valid characters	Rejected	Rejected	P	

Case ID	Input field	Test point	Expected Result	Actual Result	Status	Note
TC-22	Name	Length = 30, valid characters	Accepted	Accepted	F	Incompatible Regex
TC-23	Name	Length = 30, valid characters	Accepted	Accepted	F	Incompatible Regex
TC-24	Name	Length = 31, valid characters	Rejected	Rejected	P	
TC-25	Name	Length = 10, valid characters	Accepted	Accepted	P	
TC-26	Name	Length = 20, valid characters	Accepted	Accepted	P	
TC-27	Name	Length = 17, invalid characters	Rejected	Rejected	P	
TC-28	Email	Length = 50, valid format	Accepted	Accepted	P	
TC-29	Email	Length = 50, valid format	Accepted	Accepted	P	
TC-30	Email	Length = 51, valid format	Rejected	Rejected	P	
TC-31	Email	Length = 20, valid format	Accepted	Accepted	P	
TC-32	Email	Length = 40, valid format	Accepted	Accepted	P	
TC-33	Email	Length = 30. Invalid format	Rejected	Rejected	P	
TC-34	Location	Length = 2, valid characters	Accepted	Accepted	P	
TC-35	Location	Length = 2, valid characters	Accepted	Accepted	P	
TC-36	Location	Length = 1, valid characters	Rejected	Rejected	P	
TC-37	Location	Length = 50, valid characters	Accepted	Rejected	F	Incompatible Regex
TC-38	Location	Length = 50, valid characters	Accepted	Rejected	F	Incompatible Regex
TC-39	Location	Length = 51, valid characters	Rejected	Rejected	P	
TC-40	Location	Length = 20, valid characters	Accepted	Accepted	P	
TC-41	Location	Length = 40, valid characters	Accepted	Accepted	P	
TC-42	Location	Length = 30, invalid characters	Rejected	Rejected	P	
TC-43	Avatar	Size = 3MB, valid format	Accepted	Accepted	P	

Case ID	Input field	Test point	Expected Result	Actual Result	Status	Note
TC-44	Avatar	Size = 3MB, valid format	Accepted	Accepted	P	
TC-45	Avatar	Size = 3.1MB, valid format	Rejected	Rejected	F	App crash
TC-46	Avatar	Size = 1.5MB, valid format	Accepted	Accepted	P	
TC-47	Avatar	Size = 2.5MB, valid format	Accepted	Accepted	P	
TC-48	Avatar	Size = 2MB, invalid format	Rejected	Rejected	P	
TC-49	Title	Length = 3, valid characters	Accepted	Accepted	P	
TC-50	Title	Length = 3, valid characters	Accepted	Accepted	P	
TC-51	Title	Length = 2, invalid characters	Rejected	Rejected	P	
TC-52	Title	Length = 80, valid characters	Accepted	Accepted	P	
TC-53	Title	Length = 80, valid characters	Accepted	Accepted	P	
TC-54	Title	Length = 81, invalid characters	Rejected	Accepted	F	Incompatible Regex
TC-55	Title	Length = 20, valid characters	Accepted	Accepted	P	
TC-56	Title	Length = 50, valid characters	Accepted	Accepted	P	
TC-57	Title	Length = 40, invalid characters	Rejected	Accepted	F	Incompatible Regex
TC-58	Description	Length = 20	Accepted	Accepted	P	
TC-59	Description	Length = 255	Accepted	Accepted	P	
TC-60	Description	Length = 19	Rejected	Rejected	P	
TC-61	Description	Length = 256	Rejected	Rejected	P	
TC-62	Description	Length = 100	Accepted	Accepted	P	
TC-63	Hash tags	Length = 1, valid char, valid hash num, valid hash len	Accepted	Accepted	P	
TC-64	Hash tags	Length = 1, valid char, valid hash num, valid hash len	Accepted	Accepted	P	
TC-65	Hash tags	No hash	Rejected	Rejected	P	
TC-66	Hash tags	Length = 200, valid char, valid hash num, valid hash len	Accepted	Accepted	P	
TC-67	Hash tags	Length = 200, valid	Accepted	Accepted	P	

Case ID	Input field	Test point	Expected Result	Actual Result	Status	Note
		char, valid hash num, valid hash len				
TC-68	Hash tags	Length = 201, valid char, valid hash num, valid hash len	Rejected	Rejected	P	
TC-69	Hash tags	Hash Length = 20, valid char, valid hash num, valid len	Accepted	Accepted	P	
TC-70	Hash tags	Hash Length = 20, valid char, valid hash num, valid len	Accepted	Accepted	P	
TC-71	Hash tags	Hash Length = 21, valid char, valid hash num, valid len	Rejected	Rejected	P	
TC-72	Hash tags	Num of hash = 1, valid char, valid len, valid hash len	Accepted	Accepted	P	
TC-73	Hash tags	Num of hash = 1, valid char, valid len, valid hash len	Accepted	Accepted	P	
TC-74	Hash tags	Num of hash = 10, valid char, valid len, valid hash len	Accepted	Accepted	P	
TC-75	Hash tags	Num of hash = 10, valid char, valid len, valid hash len	Accepted	Accepted	P	
TC-76	Hash tags	Num of hash = 11, valid char, valid len, valid hash len	Rejected	Accepted	F	App crash
TC-77	Hash tags	Valid char, valid len, valid hash len, valid hash num	Accepted	Accepted	P	
TC-78	Hash tags	Valid char, valid len, valid hash len, valid hash num	Accepted	Accepted	P	
TC-79	Hash tags	Invalid char, valid len, valid hash len, valid hash num	Rejected	Rejected	P	
TC-80	Video	Size = 10MB, valid format	Accepted	Accepted	P	
TC-81	Video	Size = 10MB, valid format	Accepted	Accepted	P	
TC-82	Video	Size = 10.1MB, valid format	Rejected	Rejected	F	App crash

Case ID	Input field	Test point	Expected Result	Actual Result	Status	Note
TC-83	Video	Size = 3MB, valid format	Accepted	Accepted	P	
TC-84	Video	Size = 8MB, valid format	Accepted	Accepted	P	
TC-85	Video	Size = 5MB, invalid format	Rejected	Accepted	F	Internal logic issue
TC-86	Thumbnail	Size = 3MB, valid format	Accepted	Accepted	P	
TC-87	Thumbnail	Size = 3MB, valid format	Accepted	Accepted	P	
TC-88	Thumbnail	Size = 3.1MB, valid format	Rejected	Rejected	F	App crash
TC-89	Thumbnail	Size = 1MB, , valid format	Accepted	Accepted	P	
TC-90	Thumbnail	Size = 2.5MB, valid format	Accepted	Accepted	P	
TC-91	Thumbnail	Size = 2MB, invalid format	Rejected	Rejected	P	
TC-92	Search	Length = 3, valid characters	Accepted	Accepted	P	
TC-93	Search	Length = 3, valid characters	Accepted	Accepted	P	
TC-94	Search	Length = 2, valid characters	Rejected	Rejected	P	
TC-95	Search	Length = 80, valid characters	Accepted	Accepted	P	
TC-96	Search	Length = 80, valid characters	Accepted	Accepted	P	
TC-97	Search	Length = 81, valid characters	Rejected	Rejected	P	
TC-98	Search	Length = 20, valid characters	Accepted	Accepted	P	
TC-99	Search	Length = 40, valid characters	Accepted	Accepted	P	
TC-100	Search	Length = 30, invalid characters	Rejected	Rejected	P	
TC-101	Comment	Length = 5	Accepted	Accepted	P	
TC-102	Comment	Length = 150	Accepted	Accepted	P	
TC-103	Comment	Length = 4	Rejected	Rejected	P	
TC-104	Comment	Length = 151	Rejected	Rejected	P	
TC-105	Comment	Length = 50	Accepted	Accepted	P	
Number of Fails					13	
Number of Passes					92	

8. Result Analysis

FSM					
Issue	Case ID	Defect Impact	Defect Trigger	Defect Source	Defect Severity
Missing default display in search page	H-03	Usability	Missing default display for the state	Front-end rendering	Minor
Failed Github authentication	A-05, A-06	Functionality	Third-party integration, running platform	Back-end Integration	Major
App crash in password change	P-03	Reliability	Incorrect session handling	Back-end logic error	Critical
Invalid video format handling	VM-02	Functionality	Invalid file handling	Middleware missing file type filter	Major
App crash in video playback	VV-01	Reliability	Completion of viewing a video without thumbnail	Back-end logic not handling missing thumb data	Critical
No playback reset after completion of viewing	VV-01	Usability	Video viewing completion without reset	Back-end logic not handling reset	Minor
Input Domain Partition and Boundary					
Issue	Case ID	Defect Impact	Defect Trigger	Defect Source	Defect Severity
Incompatible Regex expression	TC-09, TC-12. TC-22, TC-23, TC-37, TC-38, TC-54, TC-57	Usability/ Functionality	Input validation with special character	Front-end using regex pattern not compatible with pug template	Major
App crash on exceeding file size limits	TC-45, TC-82, TC-88	Reliability	File uploads exceeding size limits	Middleware not properly handling size exceptions	Critical
App crash on handling hash tags	TC-76	Reliability	Input exceeding the allowed count of hash tags	Back-end logic failing to handle hash tag limits	Critical
Allowing uploading file with invalid format	TC-85	Functionality	Image file upload in video upload	Middleware not handling file type filter	Major

9. Results of re-execution of failed test cases

Case ID	Input	Expected Result	Actual Result	Status
H-03	User clicks the search icon	Redirect to search page	Redirect to search page	P
A-05	User clicks 'Continue with Github' and succeeds in authentication	Redirect to logged in homepage	Test environment does not allow redirection to gitHub	P
A-06	User clicks 'Continue with Github' and fails in authentication	Redirect to login page	Test environment does not allow redirection to gitHub	P
P-03	User submits valid password change	Redirect to logged out homepage	App crash due to logical defect in code	P
VM-02	Upload invalid video	Remain on upload page with an error message	Upload video. Redirect to logged in homepage	F →P (2^{nd} re-test)
VV-01	User clicks play icon	Video starts playing	Video starts playing	P

ase ID	Input field	Test point	Expected Result	Actual Result	Status
TC-09	Username	Length = 15, invalid characters	Rejected	Rejected	P
TC-12	Password	Length = 9, valid characters	Rejected	Rejected	P
TC-22	Name	Length = 30, valid characters	Accepted	Accepted	P
TC-23	Name	Length = 30, valid characters	Accepted	Accepted	P
TC-37	Location	Length = 50, valid characters	Rejected	Rejected	P
TC-38	Location	Length = 50, valid characters	Rejected	Rejected	P
TC-45	Avatar	Size = 3.1MB, valid format	Rejected	Rejected	P
TC-54	Title	Length = 81, invalid characters	Rejected	Rejected	P
TC-57	Title	Length = 40, invalid characters	Rejected	Rejected	P
TC-76	Hash tags	Num of hash = 11, valid char, valid len, valid hash len	Rejected	Rejected	P
TC-82	Video	Size = 10.1MB, valid format	Rejected	Rejected	P
TC-85	Video	Size = 5MB, invalid format	Rejected	Rejected	P
TC-88	Thumbnail	Size = 3.1MB, valid format	Rejected	Rejected	P