**Animation Movies**

**STP - Software Test Plan**

< Version 1. 0>

< February25, 2023>

<Raghda Knaane>

STP - Software Test Plan

TABLE OF CONTENTS

[**Revision History** 3](#_Toc129679047)

[**Document Approval**  3](#_Toc129679048)

[**1. Introduction**  4](#_Toc129679049)

[**2. Scope**  4](#_Toc129679050)

[**3. Objectives** 4](#_Toc129679051)

[**4.** **Testing Approach**  4](#_Toc129679052)

[**4.1.** **Testing Setup**  4](#_Toc129679053)

[**4.2.** **Testing Requirements**  5](#_Toc129679054)

[**4.3.** **Testing Strategy**  5](#_Toc129679055)

[**4.4. Unit Testing** 5](#_Toc129679056)

[**4.5.** **Integration Testing** 5](#_Toc129679057)

[**4.6.** **System Testing** 6](#_Toc129679058)

[**4.7.** **Acceptance Testing** 6](#_Toc129679059)

[**4.8.** **Testing Environment** 6](#_Toc129679060)

[**4.9.** **Testing Resources** 6](#_Toc129679061)

[**4.10.** **Test Deliverables** 6](#_Toc129679062)

[**5.** **Risk Assessment** 7](#_Toc129679063)

[**6.** **Priority and Severity** 7](#_Toc129679064)

[**7.** **Conclusion** 8](#_Toc129679065)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <25/02/2023> | <1.0> | SRS 1.0 | Raghda Knaane |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **Document Approval**

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  |  |  |  |
|  |  |  |  |

# **Introduction**

The purpose of this Software Test Plan (STP) is to specify the testing requirements for the Flask web application. This document outlines the approach, strategies, and resources required for testing the Flask web application, and aims to ensure that the application meets the requirements and specifications outlined in the software requirements document.

# **Scope**

The scope of this STP is to ensure that the Flask web application is thoroughly tested, and that all functionality, performance, security, and usability requirements are met. The testing will cover all modules and components of the application, including the user interface, backend, and database.

# **Objectives**

The objectives of this STP are:

* To verify that the Flask web application functions correctly and meets the specified requirements.
* To ensure that the application is user-friendly, secure, and performs well.
* To identify and report defects and issues found during testing.
* To ensure that the application is tested thoroughly and accurately.

# **Testing Approach**

The testing approach for the Flask web application will be a combination of manual and automated testing. The testing will be carried out in different stages, starting with unit testing, followed by integration testing, system testing, and acceptance testing.

## **Testing Setup**

 The testing setup for the Flask Movie App includes:

* Python
* Flask
* SQLAlchemy
* SQLite

## **Testing Requirements**

 The testing requirements for the Flask Movie App include:

* Test cases for all functions of the application.
* Test cases for all user interface elements.
* Test cases for different loads on the application.
* Test cases for different user roles (admin, user, guest).
* Test cases for different types of input (valid, invalid, malicious).

## **Testing Strategy**

* The testing strategy for the Flask Movie App includes:
* Black box testing: testing the functionality of the application without knowledge of the internal workings.
* White box testing: testing the functionality of the application with knowledge of the internal workings.
* Automated testing: testing the application using automated test scripts.
* Manual testing: testing the application manually.
* Regression testing: retesting the application after each change.
* Exploratory testing: testing the application by exploring its features.

## **Unit Testing**

The unit testing will be performed on each module and component of the application, including the backend and database. The unit testing will be performed using Python unittest and pytest frameworks, which will ensure that each unit of the code is tested individually and independently.

## **Integration Testing**

The integration testing will be performed to verify that the modules and components of the application work correctly together. This testing will be carried out after the unit testing, and it will test the integration of the backend, database, and the user interface. The integration testing will be performed manually and using automated testing tools, such as Selenium.

## **System Testing**

The system testing will be performed to verify that the application meets the specified requirements and is user-friendly, secure, and performs well. This testing will be performed manually and using automated testing tools, such as LoadRunner and JMeter, which will test the application's performance under different loads.

## **Acceptance Testing**

The acceptance testing will be performed to ensure that the application is ready for deployment. This testing will be carried out by the stakeholders, including the end-users, project managers, and developers, to verify that the application meets the requirements and specifications outlined in the software requirements document.

## **Testing Environment**

The testing environment for the Flask web application will be a combination of local and cloud-based environments. The local environment will be used for unit testing, integration testing, and system testing, while the cloud-based environment will be used for performance and load testing.

## **Testing Resources**

The resources required for testing the Flask web application include:

* Testing tools: Python unittest and pytest frameworks, Selenium.
* Testing environment: local and cloud-based environments.
* Testing personnel: testers, developers, and project managers.

## **Test Deliverables**

The test deliverables for the Flask web application include:

* Test plan document
* Test cases document
* Test summary report
* Defect reports
* Test logs

# **Risk Assessment**

 The following risks have been identified for the Flask Movie App:

* Security risks: risk of data breach, risk of unauthorized access.
* Performance risks: risk of slow response time, risk of server overload.
* Usability risks: risk of confusing user interface, risk of poor user experience.
* Compatibility risks: risk of incompatibility with different operating systems, risk of incompatibility with different browsers.
* The risks will be mitigated by:
* Implementing security measures such as encryption and authentication.
* Optimizing the application for better performance.
* Improving the user interface based on user feedback.
* Testing the application on different operating systems and browsers.

# **Priority and Severity**

|  |  |  |
| --- | --- | --- |
| **Severity** | **Priority** | **Description** |
| Critical | High | The bug causes a critical failure in the application, making it unusable. |
| High | High | The bug causes a major functionality failure,   making a significant portion of the application unusable. |
| Medium | Medium | The bug causes a moderate functionality failure, impacting a specific area of the application. |
| Low | Low | The bug causes a minor functionality failure, impacting a small area of the application. |
| Cosmetic | Low | The bug does not impact functionality but affects the visual   appearance or layout of the application. |

# **Conclusion**

The testing approach and strategies outlined in this Software Test Plan (STP) will ensure that the Flask web application is thoroughly tested and meets the specified requirements and specifications. The testing will cover all modules and components of the application, and will be performed using a combination of manual and automated testing techniques. The test deliverables will be used to track and report the progress of the testing, and to ensure that the application is ready for deployment.