# Test Plan for Giphy Project

**Version:** 1.0
**Created:** 27.10.2020
**Last Updated:** 27.10.2020
**Created by:** Paula Mišić

## *Revision History*

| Version | Change Date | By | Description |
|---|---|---|---|
| 1.0 | 27.10.2020 | Paula Mišić | Creation of the first test plan version |

# 1 Introduction

This Test Plan, Version 1.0, provides the testing approach that will drive the testing of the Giphy Project.

The testing team members will need to understand each requirement and prepare corresponding test cases to ensure all the requirements are covered.

Furthermore, the team members should go through the application manually first to understand the application and the requirements.
After that, the tester is able to ensure product quality and that the application works in the way the user/client expects.

Giphy is a website that enables users to explore and share website GIFs and most popular animated GIFs.

## 1.1 Objectives

- Identify features to be tested
- Define roles and responsibilities
- Define passing and failing criteria for each feature
- Specify testing approaches to be used during testing
- Determine deliverables for the testing process
- Define environmental needs

## 1.2 Features to be tested

### 1.2.1 In Scope

This section details the features that will be included in the testing phase:

a.) Search – user searches for desired GIFs or stickers

b.) Login – user is able to login/logout

c.) Upload – user is able to upload GIFs

d.) Home Page

### 1.2.2 Out of Scope

Rest of the application

## 1.3 Roles and Responsibilities

| Role | Responsibilities | Resource Name |
|---|---|---|
| **Test team** | 1. Understand requirements<br>2. Writing and executing Test cases<br>3. Reviewing Test cases<br>5. Defect reporting and tracking<br>6. Confirmation and regression testing<br>7. Preparation of Test Data<br>8. Coordinate with Test Lead for any issues or problems encountered during test preparation or execution/defect handling. | XY |
| **Test Lead** | 1.Participation in the project plan creation/update process.<br>2.Planning and organization of test process for the release.<br>3.Coordinate with the test teamif any issues/problems encounter during testing.<br>4.Report test progress to the PM | XY |
| **Project Manager** | 1.Acts as a primary contact for development and the test team.<br>2. Responsible for Project schedule and the overall success of the project. | XY |

## 1.4 Item Pass/Fail criteria

All functionalities in the application under test (AUT) must meet the requirements specified.
If the feature does not comply with the documented requirements, the test case will fail.
Each test case includes an expected outcome to help determine whether the test case passed or failed.

# 2 Test Approach

## 2.1 General

- The testing objective is to confirm the functionality of the AUT based on requirements.
- The test team will start with exploratory testing once the build is ready for testing to ensure the application is stable for testing.
- Test Manager and the test team will define test cases and the test documentation in the test management tool used (e.g. JIRA, Confluence, Excel).
- Test Manager will review and sign-off all test cases prepared by the test team prior to start of test execution.
- The testing process should be performed on all browsers as difference in functionality may occur.
- If the tester finds a defect/bug in the application he will report the bug in the defect management tool used.
- All the defects should come along with a snapshot JPEG format and/or video with visible steps until the tester reaches the defect and any other information valuable for the developers.
- The bug lifecycle goes through following phases: new, assigned, open, fixed, test, verified, closed, reopen, duplicate, deferred, rejected.
- Test team members will assign severity and priority levels to all bugs. The test lead will be responsible to verify the correct level has been assigned to each bug.
- After developer fixes the bug, the tester needs to retest the issue and confirm that the bug is resolved (confirmation testing).
- To validate application functionality, all features will be tested to ensure all functions provide the expected output.

## 2.2 Test Techniques andTypes

Considering the scope of the project and the time limitations, following will be performed:

### 2.2.1 Exploratory testing

Exploratory testing is a test technique used for discovering, investigating and learning about the application.
Test cases are not created in advance - testers check the application on the fly.

The tester should explore the AUT to further create practical and useful tests for the successful testing of an application.

Furthermore, the tester should confirm the application is stable for testing and critical defects are removed before starting next levels of testing.

Scope: Search, Login, Upload, Home Page

Testers: Testing team

Tools: Manual testing without any test scripts

### 2.2.2 Functional testing

Testing type used to verify that each function of the application meets the specified requirements.
Every functionality of the system is tested by providing an input, verifying the output and comparing the actual results with the expected results.

The tester should interact with the AUT in a typical way a user would do to simulate the possible scenarios.
Every scenario will be mapped with a test case.

Scope: Search, Login, Upload, Home Page

Testers: Testing team

Tools: Manual testing followed by test cases

### 2.2.3 Non-functional testing

Testing type to check non-functional aspects of a software application.
Non-functional testing will be performed after functional testing.

The tester should check how the AUT behaves when too many users login simultaneously, if the application can handle stress or does the application work as intended when accessing through different browsers.

Scope: Search, Login, Upload, Home Page

Testers: Testing team

Tool: JMeter

### 2.2.4 API testing

API(Application Programming Interface) is a type of software testing that performs verification directly at the API level.

The purpose of API Testing is to check the functionality, reliability, performance, and security of the programming interfaces.
In API Testing, instead of using standard user inputs(keyboard) and outputs, software is used to send calls to the API, get output, and note down the system's response.

Scope:  Search, Upload, Login, Home Page

Testers: Testing team

Tools: IntelliJ, REST Assured, Postman

### 2.2.5 Automated testing

Software testing technique to test and compare the actual outcome with the expected outcome by writing test scripts or using any automation testing tool.

Test automation is used to automate repetitive tasks and other testing tasks which are difficult to perform manually.

In this project, functional test cases from section 5.2 will be automated.

Scope: Search, Upload, Login, Home Page

Testers: Testing team

Tools: IntelliJ, Selenium, TestNG

## 2.3 Test Completeness

- Planned test have been executed and passed associated manual and automated tests.
- 100% test coverage has been achieved.
- No critical or high severity defects are left outstanding.
- If the deadline is close and not all of the bugs are fixed, blocker bugs and bugs with high priority are the first to be resolved.

# 3 Test Deliverables

- Test plan
- Test cases
- Test report
- Bug report

# 4 Resource & Environment Needs

## 4.1 Test Management Tools

- Jira
- Confluence
- MS Excel, MS Word

## 4.2 SoftwareTestingTools

- IntelliJ
- Selenium
- TestNG
- REST Assured
- JMeter
- Postman

## 4.3 Test Environment

All tests to be performed will should run on following OS and browsers:

1.) Windows 10 – IE11, Firefox 81.0, Chrome 85.0

2.) Mac OS 11.0 – Safari 14.0, Firefox 81.0, Chrome 85.0

Devices:

1.) Iphone 7/8/X/XS/11- iOS version 14.1

2.) Samsung Galaxy Note 10+, Galaxy S10, Galaxy S9 – Android version 10

# 5 Test cases

## 5.1 Exploratory testing

Exploratory testing is carried out without any test scripts.

## 5.2 Functional testing

The following test cases will be created and executed against the application:

1.) Search

| Name | Description |
|---|---|
| TC1: Search – enter existing GIF name in the search field | Verify entering existing GIF name in the search field displays the results properly |
| TC2: Search – enter random numbers in search the field | Verify entering random numbers in the search field displays the proper message |
| TC3: Search – enter random special characters in the search field | Verify entering random special characters in the search field displays the proper message and the app does not crash |
| TC4: Search – enter space characters in search field | Verify entering space characters in the search field does not crash the app |
| TC5: Search – click on existing GIF | Verify clicking on a GIF as result of searching displays details |
| TC6: Search – click ‘Enter’ button after entering desired GIF name | Verify user is able to search when he enters the keyword and hits ‘Enter’ button on keyboard |
| TC7: Search – search suggestions | Verify word suggestions appear when typing the keyword in the search field |

2.) Login

| Name | Description/Result |
|---|---|
| TC1: Login– enter valid email address and password | Verify entering valid email address and password redirects to Profile Page |
| TC2: Login– enter not registered email address | Verify entering not registered email address displays a error message |
| TC3: Login– enter random special characters in email address field | Verify entering random special characters in the email address field displays an error message |
| TC4: Login– enter space characters in the email address field | Verify entering space characters in the email address field disables the ‘Login’ button |

| | |
|---|---|
| TC5: Login– enter invalid password | Verify entering invalid password displays an error message |
| TC6: Login– enter random special characters in password field | Verify entering random special characters in the password field displays an error message |
| TC7: Login– enter space characters in the password field | Verify entering space characters in the password field displays an error message |
| TC8: Login– try login with Social Media | Verify if clicking on buttons for social media login redirects to Facebook or Apple |
| TC9: Login– try password recover by entering invalid email address format | Verify entering an invalid email address format displays an error message and disables the ‘Send Email’ button |
| TC10: Login – try password recover by entering space characters in email address field | Verify entering space characters in the email address field disables the ‘Send Email’ button |
| TC11: Login– try password recover by entering registered email address | Verify entering a registered email address sends a recovery email |
| TC12: Profile Page – logout from profile | Verify clicking Logout button logouts the user from profile |

3.) Upload

| **Name** | **Description** |
|---|---|
| TC1: Upload –upload GIF with valid URL | Verify entering valid GIF URL displays the GIF and details properly |
| TC2: Upload – try GIF upload with entering invalid URL | Verify entering invalid URL displays an error message |
| TC3: Upload – try GIF upload without previously logging in | Verify clicking upload button without previously logging in displays the proper message |

4.) Home Page

| **Name** | **Description** |
|---|---|
| TC1: Home Page – click on ‘Reactions’ menu item | Verify clicking Reactions menu item redirects to the correct page |
| TC2:  Home Page – click on ‘Entertainment’ menu item | Verify clicking Entertainment menu item redirects to the correct page |
| TC3: Home Page – click on ‘Sports’ menu item | Verify clicking Sports menu item redirects to the correct page |
| TC4:  Home Page – click on ‘Stickers’ menu item | Verify clicking Stickers menu item redirects to the correct page |
| TC5:  Home Page – click on ‘Artists’ menu item | Verify clicking Artists menu item redirects to the correct page |

## 5.2 Non-functional testing

The following test cases will be created and executed against the application:

| **Name** | **Description** |
|---|---|
| TC1: Load Testing | Verify page load time is not be more than 5 seconds when up to 1000 users access it simultaneously |
| TC2: Cross Browser testing | Verify the page works as intended when accessing through different browsers |
| TC3: Stress testing | Check the page behavior under extreme conditions (e.g. a large number of users logged at the same time) |
| TC4: Usability testing | Verify the application adjusts to various screen sizes |

## 5.3 API testing

For API testing, an API Key was generated from the following web page:
https://developers.giphy.com/

The GET Method is used to request data from the specified resource.

API Endpoints:

1.) Search Endpoint –api.giphy.com/v1/gifs/search

Required parameters for this endpoint:

a.) api_key: string(required) – Giphy API Key generated as above described
b.) q: string(required) – Search query term or phrase
c.) limit: integer – the maximum number of objects to return(default 25)

2.) Get GIF by ID Endpoint - api.giphy.com/v1/gifs/{gif_id}

Required parameters for this endpoint:

a.) api_key: string(required) – Giphy API Key generated as above described
d.) gif_id: string(required) – The ID of the GIF you want details for

3.) Categories- api.giphy.com/v1/gifs/categories (List of GIF categories)

Required parameters for this endpoint:

a.) api_key: string(required) – Giphy API Key generated as above described

The following test cases will be created and executed against the application:

| Name | Description |
|---|---|
| TC1: Search – search for existing GIF | Validate request headers, response body and response code when searching for existing GIF |
| TC2:Search – search for non existing GIF | Validate response body and response code when searching for non existing GIF |
| TC3: Get by ID – request GIF by its ID | Validate response body and response code when requesting a specific GIF by sending its ID |
| TC4: Get by ID – request GIF by its ID when sending invalid ID | Validate response body and response code when sending invalid ID |
| TC5: Categories –send request for categories list | Validate response body and response code when requesting list of GIF categories |
| TC6: Invalid API Key | Validate response body and response code when entering invalid API Key |

**Note**: There are more endpoints, but the ones in scope for this project are the above listed

## 5.4 Automated testing

Automated test cases can be reused and the process of regression testing is accelerated with test automation.

Automated checks are a great way of confirming that the application still functions properly after changes are made to it.
It is possible that, when a new feature is added to an application or a bug is fixed, it impacts the functionality of the working software.
By running a set of automated tests when the application is updated, we can identify any new bugs introduced as a result of the changes without manually testing all of it.

Automated tests for the AUT will follow the functional test cases described above.