**TEST PLAN**

**Spotify – Web Player**

[**https://open.spotify.com/**](https://open.spotify.com/)

Table of Contents

1. INTRODUCTION

2. SUB-TASKS

2.1. WEBSITE MANUAL TEST

2.2 Automation Testing

2.3 Performance Testing

2.4 Security Testing

2.5 API Testing

3. TESTS SCHEDULE AND ESTIMATED TIME

4.APPROVALS

**1. INTRODUCTION**

The Test Plan has been created to facilitate communication within the team members. This document describes approaches and methodologies that will apply to the modules: “Main page”, “Sign Up”, “Log In”, “Menu”, “Search”, “Playlist”, “Liked songs” of the website <https://open.spotify.com/>. This document has clearly identified sub-tasks of website's module testing, types of testing, what devices, environments, and tools to be used. It includes test cases with expected and actual results, schedule and executing estimated time each of sub-tasks and entire project.

**2. SUB-TASKS**

**2.1. Manual tests**

Execute manual testing for the Spotify – Web Player based on the next types of testing:

- GUI;

- Exploratory testing;

- Functional Positive testing:

- Functional Negative testing;

- Functional Ad-hoc testing.

GUI Testing:

GUI testing includes testing the UI part. It covers look and feel, error messages, spelling mistakes, GUI guideline violations.

Exploratory Testing:

Exploratory testing will include a type of software testing where Test cases are not created in advance but QA check system on the fly.

2.1.1. Manual Functional Testing

Manual Functional testing will be performed to check the functions of application. The functional testing is carried out by feeding the input and validates the output from the application. Manual Functional testing will be executed after Exploratory test is completed.

**Scope**: modules “Sign Up”, “Log In”, “Menu”, “Search”, “Playlist”, “Liked songs”

**Environment**: OS: Windows 11/64, Browsers: Chrome, Mozilla Firefox, Microsoft Edge

*POSITIVE TESTS* check whether an application behaves as expected with positive inputs.

The site must be accessible at the following URL: https://open.spotify.com/

“Sign up” module.

User can sign up with valid email.

“Log in” module.

User can log in with valid existing email and correct password.

“Menu” module

Verify that “Home” button leads to the right page.

Verify that “Search” button leads to the right page.

Verify that “Your Library” collapses and expands left side menu with playlists.

“Search” module.

User can find specific item (artist, song, album, audiobook).

“Playlist” module.

User can create a playlist.

User can change the name of playlist.

User can create the name of a playlist using any kind of characters.

User can add a description to the playlist.

User can add a song to the playlist.

User can delete a song from playlist.

User can delete the playlist.

“Liked songs” module.

User can add a song to “liked songs” list.

User can delete a song from “liked songs” list.

*NEGATIVE TESTS:* invalid data is inserted to compare the output against the given input.

User can NOT log in with email that wasn’t been signed up.

User can NOT log in with wrong password.

User can NOT log in without password.

User can NOT create the name of playlist with more than 100 characters.

*AD-HOC TESTS* include an informal testing type with an aim to break the system.

User can NOT sign up with invalid email.

**2.2 Automation Testing**

This test focuses on creating automation scripts based on manual test cases. This test will be carried out after manual testing is done and all critical issues are resolved.

**Scope**: The same as in Manual Functional Testing

**Tools**: PyCharm as a main IDE, Python as a main language, Selenium WebDriver as a main framework for test scripts automation, and BrowserStack to run tests in various environments

**2.3 Performance Testing**

This test is performed to measure the speed, responsiveness, stability of the site and how well the page is built for optimal performance. This test will be carried out after manual testing is done and all critical issues are resolved. After testing reports will be generated.

**Scope**: https://open.spotify.com

**Tools**: GTMetrix, SpeedLab, Lighthouse

**2.4 Security Testing**

This test is performed to reveal current or potential security vulnerabilities. This test will be carried out after manual testing is done and all critical issues are resolved. After testing reports will be generate.

**Scope**: https://open.spotify.com

**Tools**: The testing team should use Snyk and Mozilla Observatory

**2.5 API Testing**

API tests are performed to determine if the developed APIs meet the expectations when it comes to the functionality, performance, reliability and security of the website. This test will be carried out after manual testing is done and all critical issues are resolved.

**Scope**: https://open.spotify.com

**Tools**: Postman API as a platform for building and using APIs and JavaScript as a language for test scripts.

* GET. Get Current User's Profile
* GET. Get User's Profile by ID
* POST. Create Playlist
* GET. Get User's Playlists to check playlist was created successfully
* PUT. Change Playlist Details
* GET. Check User's Playlists Was Changed
* POST. Add Items to Playlist
* GET. Get Playlist Items
* DELETE. Remove Playlist Items
* GET. Check Playlist Items Were Deleted
* PUT. Save Tracks for Current User
* GET. Get User's Saved Tracks
* DELETE. Remove User's Saved Tracks
* GET. Check User's Saved Tracks Were Deleted
* GET. Get User's Profile with Unexisted ID
* POST. Playlist name more than 100 characters
* POST. Playlist name with 0 characters
* POST. Add Items to Playlist With Empty Body

**3. TESTS SCHEDULE AND ESTIMATED TIME**

|  |  |  |  |
| --- | --- | --- | --- |
| Sub-Task Name | Start | Est. time (hours) | * Finish |
| Manual Testing |  |  |  |
| Automation Testing |  |  |  |
| Performance testing |  |  |  |
| Security testing |  |  |  |
| API Testing |  |  |  |

Test Execution

QA -

Test phase -

Status –

**4. APPROVALS**

|  |  |  |
| --- | --- | --- |
|  | Project Manager | QA Lead |
| Name |  |  |
| Signature |  |  |