TEST PLAN FOR WIKIPEDIA PROJECT

Test plan prepared by Sergei Smirnov

Table of Content.

1. Objects……..3
1.1 Objects…….3
1.2 Purpose……....3
1.3 Team Members……3
2. Scope……..4
2.1 Scope……....4
2.2 Test approaches……..4
3. Test Types……...5
4. Bug severity list……..6
5. Requirements……7
5.1 Requirements……..7
5.2 Bug priority list……...8
6. Testing tools……....8
6.1 Testing tools…….8
6.2 Test Environment……....9
7. Test plan for Manual, Automation, API and Functional testing ……..9

Introduction

The test plan was developed to understand the goals, objectives of Wikipedia product testing. The plan shows what needs to be tested, what needs to be tested with, and what requirements are important in this product. Visitors to the site want a great site without any problems or defects.

1.1 Objectives

1.2 **The main purpose** of this test is to show what defects there are in the site, and that the requirements of the site are at the highest level. Any change in the documentation must be documented, and this information must be correct, written in real time, and the text must be readable.

1.3 Team Members

Member	**Tasks**
Test Manager	Manage the whole project, define projects directions.
Testers	Testing the entire product for defects, writing reports and checking the performance of the site
Developers	Implement the test cases, test program, test suite.

Test Administrator	Builds up and ensures test environment and assets are managed and maintained

2 Scope

2.1 All the features of website Wikipedia which were defined in software requirement specs need to be tested. Testing of all functional, application performance.

2.1 Test approaches

Types of testing that will be conducted as part of this test:

1. Website Performance testing
2. Manual testing
3. Automation testing
4. API testing

3 Test Types

In the Wikipedia project there are 7 types of testing that should be conducted:

1. Exploratory testing
2. Smoke testing
3. Functional testing
4. Positive testing
5. Negative testing
6. API testing
7. Performance testing

Exploratory testing:

Exploratory Testing is a type of software testing where Test cases are not created in advance but testers check system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a “thinking” activity.

Smoke testing

Smoke Testing is a software testing process that determines whether the deployed software build is stable or not. Smoke testing is a confirmation for QA team to proceed with further software testing. It consists of a minimal set of tests run on each build to test software functionalities.

Functional testing

Functional testing is a type of testing which verifies that each function of the software application operates in conformance with the requirement specification.

Positive testing

Positive testing is the type of testing that can be performed on the system by providing the valid data as input. It checks whether an application behaves as expected with positive inputs. This test is done to check the application that does what it is supposed to do.

Negative testing

Negative Testing is a software testing type used to check the software application for unexpected input data and conditions. Unexpected data or conditions can be anything from wrong data type to strong hacking attack. The purpose of negative testing is to prevent the software application from crashing due to negative inputs and improve the quality and stability.

API testing

API TESTING is a software testing type that validates Application Programming Interfaces (APIs). 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, you use software to send calls to the API, get output, and note down the system's response.

Performance testing

Performance Testing is a software testing process used for testing the speed, response time, stability, reliability, scalability and resource usage of a software application under particular workload. The main purpose of performance testing is to identify and eliminate the performance bottlenecks in the software application.

4. Bug severity list

Severity ID	Severity	Description
1	**Highest**	**The most difficult bugs level that break the site so much that it is impossible to use it, all functions of the site stop working**

2	**High**	**The level of which is also at a high level, but some functions of the site are still in working order, it becomes very difficult to use the site, but it is possible**
3	**Medium**	**Small bugs that interfere with the performance of the site, not serious problems in the work site**
4	**Low**	**Errors that may not affect the site, or slightly, but which you cannot see, errors in spelling or documentation**

5. Requirements.

5.1 Requirements All functions of the site should work and be at the highest level, no bugs should not be found on the site, all kinds of testing should be carried out and if defects are found, they must be corrected.

5.2 Bug priority list

Priority	Priority Level	Description
1	Highest	This bug must be fixed immediately.
2	High	These are important problems should be fixed as soon as possible.
3	Medium	The problem should be fixed within the time available.
4	Low	Fix these bugs after all other bugs have been fixed.
5	Lowest	Documentation errors.

6. Testing tools

6.1

Process	Tool
Test case creation	Microsoft Word, Microsoft Excel
Test case tracking	Jira
Test case execution	Manual, Selenium IDE

Test case management	**Microsoft Excel, Jira**
Defect management	**Microsoft Word, Jira**
Test reporting	**Jira**
API testing	**Postman**
Performance testing	**Lighthouse, Speed Lab, GT Metrix**
Automation testing	**Selenium IDE**

6.2 Test Environment

Test environment to be setup as per figure below:

- Windows 10: Chrome (Latest)
- Windows 11: Chrome (latest)

7. **Test plan for test types.**

Manual - Test the Wikipedia site to see if the language change works, also check that the login menu works, check that the search engine works, and that other Wikipedia projects work and load without problems, use the Microsoft Excel and Google Chrome.

Automation - To check that the tests made manually also work automatically and without any problems, use the Selenium IDE.

Performance - Check the performance of the site using Lighthouse, GT Metrix and Speed Lab. After all the tests, save the results and fill out the documentation in Microsoft Excel.

API - Check the functionality of API data on Wikipedia site using Postman, export the result and create a document in Microsoft excel with all the results.