

Quality Assurance Test Plan

***Google Translate***

**Classification:** Internal

**Owner:** WireApps

Contents

[**Document Change History** 3](#_Toc165789594)

[**1.** **Introduction** 4](#_Toc165789595)

[**2.** **Objectives** 4](#_Toc165789596)

[**3.** **Scope of Testing** 4](#_Toc165789597)

[**3.1** **In Scope** 4](#_Toc165789598)

[**3.1.1** **Test Coverage** 4](#_Toc165789599)

[**3.1.2** **Functional Requirements** 4](#_Toc165789602)

[**3.1.3** **Non-functional Requirements** 4](#_Toc165789606)

[**3.2** **Out of Scope** 5](#_Toc165789609)

[**4.** **Risks** 5](#_Toc165789610)

[**5.** **Assumptions** 5](#_Toc165789611)

[**6.** **Test Deliverables and Schedule** 5](#_Toc165789612)

[**7.** **Test Strategy** 5](#_Toc165789613)

[**7.1** **Test Design Techniques** 5](#_Toc165789618)

[**7.2** **Test Approach** 5](#_Toc165789619)

[**7.3** **Test Team** 6](#_Toc165789620)

[**8.** **QA Team Structure** 6](#_Toc165789621)

[**9.** **Test Environment** 6](#_Toc165789622)

[**10.** **Defect Management** 6](#_Toc165789623)

[**11.** **Defect Tracking Process** 7](#_Toc165789624)

[**12.** **Defect Status Life Cycle** 8](#_Toc165789625)

# **Document Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Requested by | Approved by | Document Version |
| Initial Document | Dilini Punchihewa |  | 1.0 |

# **Introduction**

The purpose of this document is to gather all information necessary to plan and control the test effort for Google Translator project & ensure the quality and functionality of the project.

# **Objectives**

This document supports the following objectives:

* Identify the items that should be targeted by the tests.
* Outline the testing approach that will be used.
* Specify the deliverable elements of the test project.

# **Scope of Testing**

## **In Scope**

### **Test Coverage**

The type of tests executed by QA team for this project are mentioned in the table below.

|  |
| --- |
| Type of Testing |
| * Smoke Testing |
| * Functional Testing |
| * Regression Testing |
| * UI / Usability Testing |
| * System Integration Testing |
| * Performance/Load Testing |
| * Security Testing |
| * Compatibility Testing |
| * UAT Testing |



### **Functional Requirements**

Functional requirements of the system are ranked based on the criticality.

|  |  |  |
| --- | --- | --- |
| No. | Functional Requirement | Criticality |
| 1 | FRS Web Application | N/A |
| 2 | FRS Mobile App | N/A |



### **Non-functional Requirements**

The following table contains non-functional requirements of the system.

|  |  |  |
| --- | --- | --- |
| No. | Non-Functional Requirement | Criticality |
| *1* | Usability Requirements | N/A |
| *2* | Reliability Requirement | N/A |
| *3* | Performance Requirements | N/A |
| *4* | Interface Requirements | N/A |



## **Out of Scope**

QA Team will not execute tests to verify the following features.

|  |  |
| --- | --- |
| No. | Out of Scope Functionalities |
| *1* | *Testing with all languages.* |
| *2* | *UX Testing* |
| *3* | *Test load exceeding the expected user size specified in the test environment.* |

# **Risks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Quality Risks | Frequency | Mitigation Strategy | Priority |
| Mobile App | Availability of devices for testing may impact the testing process | Medium | Google to assess and provide the required test devices | High |

# **Assumptions**

The plan is based on the following assumptions, which affect scope of testing:

* All the requirements are finalized and clearly written in stories.
* User stories are created in JIRA, with FRS reference.
* The given wireframes are considered final for QA test design/Test Execution.

# **Test Deliverables and Schedule**

* Test cases for manual testing.
* Test reports summarizing the results of all test phases.
* Recommendations for improvements based on test findings.

# **Test Strategy**



## **Test Design Techniques**

* Flow charts will be used for test scenario designing.
* Decision tables will be used to identify scenarios to ensure maximum test coverage in google translator applications.

## **Test Approach**

|  |  |
| --- | --- |
| Manual Testing | Testers will perform manual tests to evaluate translation usability, accuracy, and security. |

## **Test Team**

This section will include information about the QA team.

|  |  |
| --- | --- |
| QA Engineers | Dilini Punchihewa |

# **QA Team Structure**

The organization chart depicts the overall project organization with details of the QA team structure.

QA Architect

PM

QA Lead

Senior QA / QA Engineers

# **Test Environment**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Type | Environments Required | Specific Criteria (Hardware/Software) | Test Data | Test Devices |
| Functional Testing | production | Web browsers (Chrome, Firefox, Safari.),  operating systems (Windows, macOS, iOS, Android), | Sample texts data in multiple languages for translation testing | Computers and mobile devices |

# **Defect Management**

This section contains information about all the defect management related activities in the project.

# **Defect Tracking Process**

Have a Triage Meeting and set the Severity and Priority

Assign the defect to the developer

Create Defect in JIRA

QA verifies the fix and closes the defect

Developer fix it and releases to QA

* QA engineers will log the defects in the bug tracking tool and when the defects are found.
* Triage meetings will be held after completing the QA release to discuss the defects identified and to clarify any concerns the Development team may have. Priority of the defects will be decided and set.
* A developer will be assigned to each defect to analyze, tag as a valid defect, and to find the resolution. Once the defects are been fixed and they will be released to QA for testing on the relevant testing cycle (testing cycle 2 or 3).
* Defects released to QA will be retested and closed in the tool.
* If a defect is considered not to be a defect (unable to replicate/ existing requirement etc.) the QA Engineer who submitted the defect should withdraw it.
* If a defect identified is a requirement defect, it needs to be assigned to the appropriate BA to update the requirements document. Once the requirements document is updated the defect should be closed by the QA Engineer who submitted the defect.
* Defects related to erroneous requirements will be out of scope for QA testing.

# **Defect Status Life Cycle**

A diagram of a process

Description automatically generated

* Once a defect is raised it will be in open status.
* Open defect will be assigned to developer and developer will change it to dev in progress when he/she starts fixing it
* When the defect is fixed it will be changed to resolved status and released to QA
* QA will verify the defect fix and if it is fixed will change the status to closed else it will be re-open again
* Reopen defect will be assigned to the relevant developer and he/she will take it to fix by changing the status to dev in progress
* The same cycle will continue until the defect is closed.