Application “Kitten like”

**Test Strategy**

**Revision History**

| Date | Version | Author | Description |
| --- | --- | --- | --- |
| Jan 4, 23 | 1.0 | Bilokon U. |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[**1. Scope 3**](#_heading=h.gjdgxs)

[**2. Test Approach 4**](#_heading=h.30j0zll)

[**3. Test Environment 5**](#_heading=h.1fob9te)

[**4. Testing Tools 6**](#_heading=h.3znysh7)

[**5. Release Control 7**](#_heading=h.2et92p0)

[**6. Risk Analysis 8**](#_heading=h.tyjcwt)

[**7. Review and Approvals 9**](#_heading=h.3dy6vkm)

# Scope

Testing will cover the functional testing of the application “Kitten like”. Functionality for this release is detailed in the application's “Kitten like” Requirements specifications documents.

White-box testing is carried out by the development team. The scope and methods used are selected by the Senior Developer and approved together with the Product Owner and Senior QA Engineer.

Black-box testing is carried out by the Senior QA Engineer. The scope and methods used are selected by the Senior QA Engineer and approved together with the Product Owner.

The timelines are defined in both Test Plans.

This document shall be completed and used by the project test team to guide how testing will be managed for this project. The test effort will be prioritized and executed based on the project priorities as defined in the Project Plan and Requirements Specification. This is a living document that may be refined as the project progresses.

Product owner, Senior Developer and Senior QA Engineer will review the Test strategy.

Product Owner will approve this Test Strategy.

# Test Approach

The testing approach for this release should be executed in such a way as to ensure that the minimum set of features works properly and that the user interface conforms to the specific design.

2.1. Usability

Usability testing tests the ease with which users can learn and use a product.

2.2. Load testing

Load testing simulates multi-user access to an application to ensure that all functions can be used to perform specified requirements with no failures. The script will be written by the QA team and approved by the development staff. Found defects will be logged as such in the defect tracking tool.

2.3. Performance testing

Performance testing is conducted to evaluate the compliance of a system response time, and the ability to function in various operating environments. The approach to this will be manual testing.

2.4. Regression

Regression testing involves re-testing a previously tested program following modification to ensure that faults have not been introduced or uncovered as a result of the changes made. In this release this will be covered by the ongoing use of manual tests being executed after each successful build of the application, prior to release of the build for general testing use.

The Test Team will do a pass through all the test scripts that were developed for this project. This will encompass the re-testing of each item in each test script as well as the re-verification of each repaired defect that is decided on as an items to be regressed based on the severity of the defect.

Positive test cases will reflect that the application functions as expected. Negative test cases are tests that exercise the limits and boundaries outside the expected designs.

2.5. Recovery

Recovery testing forces the failure of the software in a variety of ways to verify that recovery is properly performed.

2.6. Security testing

Security testing evaluates whether the system meets its specified security objectives by attempting to break in or disable a system by improper acquisition of a password, bypassing security measures, browsing through insecure data, or overwhelming the application with requests.

2.7. Installation / Configuration testing

Installation / Configuration testing verifies that the system will install and function on all required operating platforms, under all specified configurations.

Application “Kitten Like” should be able to run on any mobile phone that has OS Android 8 or higher and OS iOS 8 or higher.

2.8. Documentation verification

Documentation verification involves reviewing for accuracy all supporting User Documentation, Help Files, and supplemental materials.

During the testing process team use bug-tracking system TestRail.

# Test Environment

* Define number of requirement and setup required for each environment
* Define backup of test data and restore strategy

# Testing Tools

* Automation and Test management tools needed for test execution
* Figure out number of open-source as well as commercial tools required, and determine how many users are supported on it and plan accordingly

# Release Control

* Release management plan with appropriate version history that will make sure test execution for all modification in that release

# Risk Analysis

* List all risks that you can estimate
* Give a clear plan to mitigate the risks also a contingency plan

# Review and Approvals

* All these activities are reviewed and sign off by the business team, project management, development team, etc.
* Summary of review changes should be traced at the beginning of the document along with approved date, name, and comment