Mobile application "Masik"

Test Strategy

Revision History

Date	Version	Author	Description
20.05	Version 1.1	Alona	Updated section on compatibility testing for iOS

Table of Contents

1. Scope and overview

The Masik mobile application is created for real cat lovers and connoisseurs of expertly captured cat moments in photos. The most important functionality of the application is the exchange of photos of cats, the ability to comment on photos, and find real cat masterpieces.

The application also promotes communication between people who share a love for cats. Users have the opportunity to add each other as friends, follow updates, share impressions, discuss interesting photos, organize real cat discussions.

The Masik application is a world where there are only cats, because it is forbidden to upload photos of other animals. True connoisseurs of feline grace and cuteness will be able to appreciate the easy, pleasant and intuitive interface of the application. Photos are divided into categories, so everyone will be able to easily find those that interest them the most. Personal information of users is protected.

The test strategy document will be reviewed and approved by the test team, test leader and project management.

The test strategy provides for functional, load, regression, security, compatibility and usability testing. Each type of testing will be conducted in accordance with the specified schedule and testing plan.

2. Test Approach

- Process of testing

Test design: The testing team will develop test cases and test scenarios based on the requirements and user stories. Test data will be identified, and test environments will be set up.

Test execution: The test cases will be executed using only manual testing techniques. The testing team will report and track defects, and retest fixed defects.

Test reporting: The testing team will prepare and share test reports that summarize the testing progress, the number of defects, and the overall quality of the application.

- Testing levels

Testing will be conducted at the following testing **levels**:

Unit Testing: Testing individual blocks or components of an application, such as functions or modules.

Integration Testing: Testing how different blocks/modules work together when they are integrated.

System Testing: Testing the entire application as a whole to ensure that all components are working properly.

Acceptance testing: Testing the application for compliance with requirements and its acceptability to end users.

- Roles and responsibilities of each team member

Project Manager:

Overseeing the working process, coordinating with stakeholders, and ensuring the overall progress aligns with project timelines.

Test Lead:

Leading the testing team, assigning tasks, and ensuring testing objectives are met.

Test team:

Executing test cases, reporting bugs, and ensuring the application meets quality standards.

Developers:

Providing support for bug fixes, addressing code-related issues, and ensuring the application functions as intended.

Business Analyst:

Providing domain knowledge, defining requirements, and ensuring that the application aligns with business objectives.

Stakeholder:

Reviewing test reports, providing feedback, and making decisions related to the application's quality.

- Types of Testing

The following **testing types** will be performed during the testing of the application:

Functional testing: This type of testing ensures that the application functions correctly according to the requirements of cats photos sharing functionality. It includes the testing of photos uploading, commenting on photos, liking photos, adding users to friends lists.

Usability testing: This type of testing focuses on user experience, ease of use, and user interface design specific to photo exchange application.

Performance testing: This type of testing evaluates the system's responsiveness, stability, and scalability under different load conditions for uploading photos, different number of users, durability of use of application.

Security testing: This type of testing ensures that the application is secure from unauthorized access, data breaches, and other security threats.

Compatibility testing: This type of testing checks if the application functions correctly across different operating systems, such as Android and iOs

- Adding new defects, re-testing, Defect triage, Regression Testing and test sign off

New defects discovered during testing will be promptly documented in the defect tracking system.

After defects are fixed, re-testing will be conducted to ensure the issues have been resolved.

Defects will be triaged based on severity and priority to determine the order of resolution.

Regression testing will be performed to verify that new changes do not adversely impact existing functionality.

The testing team will conduct a final review of test results and provide sign-off before the application is released.

3. Test Environment

The mobile application is designed to work on IOS 4.0 and higher versions and on Android 4.2.2 and higher versions.

For both Android and iOS test environments, the hardware requirements include:

Android:

Android devices with various screen sizes and versions for compatibility testing.

High-performance devices for performance testing.

Emulators and simulators for testing on virtual devices.

iOS:

Apple devices (iPhones, iPads) with different screen sizes and iOS versions.

Emulators and simulators provided by Xcode for testing iOS apps on virtual devices.

4. Testing Tools

For testing the mobile application will be used such testing tools:

XCTest and XCTestUI: Testing frameworks for iOS apps.

Espresso: Testing framework for Android apps.

As for test management tools:

Jira: For test case management and bug tracking.

TestRail: Test case management tool with reporting capabilities.

5. Release Control

A release calendar will be maintained to track the planned release dates, ensuring alignment with project milestones and timelines. While maintaining a continuous delivery approach, release planning may focus on setting smaller scopes for each release based on the priorities of the backlog items.

Comprehensive release notes will be maintained to track the changes, enhancements, bug fixes, and new features introduced in each release.

Regular Retrospectives: Conducting regular retrospectives to reflect on the release process, identify areas for improvement, and adjust the release control strategy accordingly.

6. Risk Analysis

Potential risks are device compatibility issues, security vulnerabilities, network connectivity problems, or performance concerns.

Risk: Limited device compatibility may result in user experience issues on certain devices.

Mitigation: Conduct thorough compatibility testing on a variety of Android and iOS devices with different screen sizes and resolutions to ensure a consistent user experience.

Risk: Insufficient network bandwidth leading to slow image uploads/download.

Mitigation: Perform performance testing under different network conditions to optimize image upload and download speeds.

Risk: Data security vulnerabilities may expose user information.

Mitigation: Prioritize thorough security testing to identify and address any potential security loopholes to protect user data from unauthorized access.

7. Review and Approvals

All activities outlined in this test strategy document will be reviewed and signed off by the test team, test lead, project manager, and relevant stakeholders.

For capturing and documenting review changes, there are tracked changes in the document, meeting minutes, and a separate change log.

Summary of review changes should be traced at the beginning of the document along with the approved date, name, and comments for transparency and accountability.

Stakeholder: Troitska Alona