



Testing G

Contents

2.1 Requirement Analysis	4
2.2 Test Planning.....	4
2.3 Test Design	4
2.4 Test Setup	5
2.5 Test Execution	5
2.6 Defect Reporting.....	6
2.7 Test Closure.....	6
3. Types of Testing.....	6
3.2 Usability Testing.....	7
3.3 Performance Testing	8
3.4 Compatibility Testing	8
3.5 Security Testing.....	9
3.6 Accessibility Testing.....	9
3.7 Localization Testing.....	9
3.8 Interrupt Testing.....	10
4. Manual Testing Guidelines.....	10
4.1 Prepare Test Cases	11
4.2 Perform Exploratory Testing	11
4.3 Check Device Compatibility	12
4.4 Test Responsiveness.....	12
4.5 Conduct Regression Testing.....	12
4.6 Test User Interface.....	13
4.7 Document Issues.....	13

5. Free Testing Tools for Manual Testing	13
5.1 Functional Testing Tools	13
5.2 Cross-Device Testing Tools.....	14
5.3 Performance Testing Tools	14
5.4 Accessibility Testing Tools	14
5.5 Security Testing Tools.....	14
5.6 UI/UX Testing Tools	14
6. Best Practices	15
6.1 Test on Real Devices.....	15
6.2 Prioritize Critical Test Cases.....	15
6.3 Involve Real Users	15
6.4 Document Everything.....	16
6.5 Continuous Testing.....	16
7. Conclusion.....	16

Guidelines For Android App

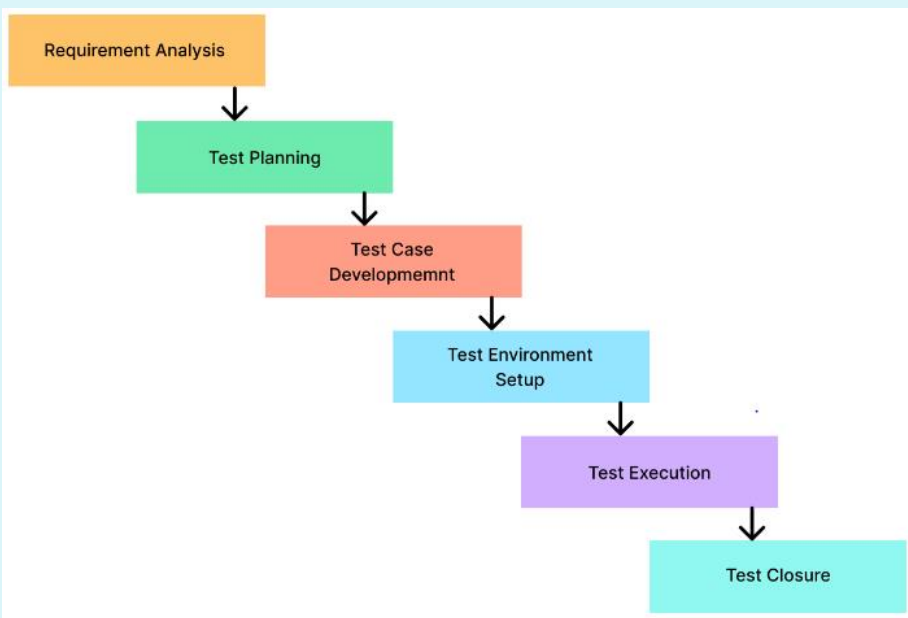
Detailed Testing Guidelines for Android App Testing

1. Introduction

Manual testing of Android applications is essential to ensure that the app delivers a seamless user experience, functions correctly, and is free from bugs and vulnerabilities. This document outlines comprehensive guidelines for manually testing Android applications, covering all key phases, types of testing, and best practices. Additionally, it includes a list of free tools that can aid testers in conducting manual testing effectively.

2. Testing Phases

The Android app testing process can be divided into several key phases:



2.1 Requirement Analysis

Objective: Understand the scope, functionality, and performance expectations of the Android app.

Activities:

Review project documentation, including functional and non-functional requirements.

Identify critical features, modules, and areas that require special attention.

Gather information on target audience, supported devices, OS versions, and expected usage scenarios.

Output: A clear understanding of what needs to be tested, including key focus areas.

2.2 Test Planning

Objective: Plan the testing process, defining the scope, objectives, and resources required.

Activities:

Develop a detailed test plan document outlining testing objectives, strategies, scope, resources, schedule, and risk analysis.

Identify the testing environments, devices, tools, and required skills.

Prioritize test cases based on critical functionalities and business impact.

Output: A test plan document that guides the entire testing process.

2.3 Test Design

Objective: Create detailed test cases and scenarios to be executed during testing.

Activities:

Design test cases based on the requirements, covering all possible user interactions and edge cases.

Develop test data and input values for various scenarios.

Review and validate test cases with stakeholders to ensure they align with the requirements.

Output: A set of detailed test cases ready for execution.

2.4 Test Setup

Objective: Prepare the testing environment, including setting up devices and configuring testing tools.

Activities:

Install the Android app on various devices and OS versions to be tested.

Configure testing tools and ensure they are properly integrated with the testing environment.

Verify that all devices are connected and functioning correctly.

Output: A fully prepared and functional testing environment.

2.5 Test Execution

Objective: Execute the designed test cases and document the results.

Activities:

Perform manual testing by executing the test cases on the target devices.

Record the actual outcomes and compare them with the expected results.

Identify and document any discrepancies, defects, or unexpected behavior.

Output: Test execution results, including a list of identified issues.

2.6 Defect Reporting

Objective: Document and communicate any defects or issues found during testing.

Activities:

Log defects in a defect tracking system with detailed descriptions, steps to reproduce, severity, and screenshots if applicable.

Assign defects to the appropriate team members for resolution.

Retest the resolved issues and update the defect status accordingly.

Output: A defect log that tracks the status of all identified issues.

2.7 Test Closure

Objective: Conclude the testing phase and ensure all critical issues are resolved.

Activities:

Review the test execution results to ensure all critical test cases have passed.

Conduct a final round of regression testing to confirm that recent changes haven't introduced new issues.

Prepare a test summary report detailing the overall test results, including pass/fail rates, defect statistics, and key findings.

Output: A test closure report that confirms the Android app is ready for production.

3. Types of Testing

To ensure the Android app's overall quality, various types of testing should be performed:

Types of Mobile App Testing



3.1 Functional Testing

Objective: Ensure that all features and functionalities work as intended according to the requirements.

Activities:

Test all user interactions, such as clicking buttons, submitting forms, and navigating through different app screens.

Validate that input fields accept and process data correctly.

Verify that business logic and workflows are implemented correctly.

Check the integration of third-party services, such as payment gateways, APIs, etc.

3.2 Usability Testing

Objective: Assess the app's user interface (UI) and user experience (UX) to ensure it is intuitive and user-friendly.

Activities:

Evaluate the ease of navigation and the logical flow of the app.

Test the readability of content, including font size, color contrast, and spacing.

Observe user interactions to identify potential usability issues, such as confusing layouts or unclear instructions.

Collect feedback from real users, if possible, to gain insights into the user experience.

3.3 Performance Testing

Objective: Test the app's speed, responsiveness, and stability under various conditions.

Activities:

Monitor app load times and identify any performance bottlenecks.

Test the app's response time during high load scenarios, such as handling multiple users or large data volumes.

Evaluate the app's performance on different devices, including low-end and high-end devices.

Analyze the app's battery consumption and memory usage.

3.4 Compatibility Testing

Objective: Ensure the app functions correctly across different Android devices, OS versions, and screen sizes.

Activities:

Test the app on various devices with different screen sizes, resolutions, and hardware configurations.

Verify that the app runs smoothly on different Android OS versions, from older to the latest.

Check for any device-specific issues, such as UI misalignment, crashes, or slow performance.

3.5 Security Testing

Objective: Identify and address potential security vulnerabilities in the Android app.

Activities:

Test for common security threats like SQL injection, cross-site scripting (XSS), and data leaks.

Validate that sensitive data, such as passwords and personal information, is properly encrypted and stored.

Ensure secure communication protocols (e.g., HTTPS) and proper session management.

Verify that user authentication and authorization mechanisms are robust and secure.

3.6 Accessibility Testing

Objective: Ensure the app is accessible to users with disabilities and complies with accessibility standards.

Activities:

Validate that the app meets accessibility guidelines

Test the app's compatibility with screen readers and other assistive technologies.

Check for appropriate use of alternative text for images, keyboard navigation support, and color contrast.

Ensure that forms and other interactive elements are accessible and easy to use for all users.

3.7 Localization Testing

Objective: Verify that the app's content is correctly translated and adapted for different languages and regions.

Activities:

Test the app in different languages to ensure accurate translation and proper display of localized content.

Check for any UI issues that arise due to text expansion or contraction in different languages.

Validate that currency, date, and time formats are correctly localized.

Verify that the app complies with regional regulations and standards.

3.8 Interrupt Testing

Objective: Ensure the app handles interruptions gracefully without affecting its functionality.

Activities:

Simulate interruptions such as incoming calls, SMS, or notifications while using the app.

Test how the app behaves when the device is put to sleep, woken up, or when the app is minimized and then resumed.

Check the app's behavior during network connectivity changes, such as switching between Wi-Fi and mobile data.

4. Manual Testing Guidelines

Manual testing is essential for assessing the app's usability, functionality, and overall user experience.

PROJECT NAME	mobile app testing									
Created by	NIKHIL NIGAM									
Current Date	22/11/2023									
TC No	Test scenario	Test Criteria	Test Steps	Role	Used Credentials	EXPECTED RESULT	ACTUAL RESULT	TESTING STATUS	REMARK/Attachment	TESTER NAME
	login after re install(home screen functionality)	CHECK USER ABLE TO SEE PROFILE Icon OR NAME	3) USER GET OTP 4) ENTER OTP 5) LOGIN WITH NUMBER AND PASSWORD 6) see PROFILE ICON		id-8400442832	1) USER SHOULD ABLE TO SEE HIS/HER PROFILE ICON AND PHOTO OR NAME	1) USER IS NOT ABLE TO SEE HIS/HER PROFILE ICON AND PHOTO OR NAME	Fail	profile icon	nikhil nigam
	login after re install(home screen functionality)	CHECK USER ABLE TO CHANGE MOBILE NUMBER	1) ENTER A LOGIN NUMBER 3) USER GET OTP 4) ENTER OTP 5) LOGIN WITH NUMBER AND PASSWORD 6) PROFILE ICON		id-8400442832	1) USER SHOULD ABLE TO CHANGE MOBILE NUMBER	1) USER IS NOT ABLE TO CHANGE MOBILE NUMBER	Pass		nikhil nigam
	login after re install(home screen functionality)	user is able to change his/her name or change user details	1) ENTER A LOGIN NUMBER 3) USER GET OTP 4) ENTER OTP 5) LOGIN WITH NUMBER AND PASSWORD 6) PROFILE ICON		id-8400442832	user should able to change his/her details	user is not able to change his/her details	Pass		nikhil nigam
	login after re install(home screen functionality)	verify user able to use changed password	1) ENTER A LOGIN NUMBER 3) USER GET OTP 4) ENTER OTP 5) LOGIN WITH NUMBER AND PASSWORD 6) PROFILE ICON		id-8400442832	user should able to enter changed password	user is not able to enter changed password	Fail		nikhil nigam
	login after re install(home screen functionality)	check all assigned project show or not	1) ENTER A LOGIN NUMBER 3) USER GET OTP 4) ENTER OTP 5) LOGIN WITH NUMBER AND PASSWORD 6) PROFILE ICON		id-8400442832	user should able to see all assigned project in user number	user is able to see all assigned project in user number	Pass		nikhil nigam

4.1 Prepare Test Cases

Objective: Develop detailed test cases that cover all functionalities and edge cases.

Activities:

Write test cases that clearly describe the steps to be performed and the expected outcomes.

Ensure test cases cover all user interactions, including positive, negative, and edge cases.

Organize test cases by functionality or module for easy execution.

Output: A comprehensive set of test cases ready for execution.

4.2 Perform Exploratory Testing

Objective: Discover unexpected issues by testing without predefined test cases.

Activities:

Explore the app freely, trying different combinations of actions and inputs.

Test unusual or edge case scenarios that may not be covered by standard test cases.

Use your intuition and experience to identify potential problem areas.

Output: A list of any issues discovered during exploratory testing.

4.3 Check Device Compatibility

Objective: Ensure the app functions correctly on different Android devices.

Activities:

Execute test cases on a variety of devices with different screen sizes, resolutions, and hardware specifications.

Identify and document any device-specific issues, such as UI problems or performance issues.

Output: A compatibility report highlighting any device-specific issues.

4.4 Test Responsiveness

Objective: Verify that the app adapts properly to different screen sizes and resolutions.

Activities:

Test the app on multiple devices, including smartphones and tablets with different screen sizes and resolutions.

Ensure that the app's UI elements are properly aligned and that content is displayed correctly.

Check for any issues related to scrolling, zooming, or orientation changes.

Output: A report on the app's responsiveness across different screen sizes and orientations.

4.5 Conduct Regression Testing

Objective: Ensure that recent changes or fixes haven't introduced new issues.

Activities:

Re-execute previously passed test cases after any code changes or updates.

Focus on areas that are most likely to be affected by the recent changes.

Verify that any fixed issues remain resolved and that no new issues have been introduced.

Output: A regression test report indicating whether any new issues were introduced.

4.6 Test User Interface

Objective: Ensure that the app's UI is consistent, intuitive, and visually appealing.

Activities:

Verify the alignment, spacing, and size of UI elements across different devices.

Check the consistency of fonts, colors, icons, and other design elements.

Test the app in different lighting conditions to ensure visibility and readability.

Output: A UI test report detailing any inconsistencies or visual issues.

4.7 Document Issues

Objective: Record any bugs or issues found during testing with detailed information.

Activities:

Log each issue with a clear description, steps to reproduce, expected and actual outcomes, and severity.

Include screenshots or video recordings if necessary to illustrate the issue.

Assign issues to the appropriate team members for resolution and track their progress.

Output: A detailed defect log.

5. Free Testing Tools for Manual Testing

5.1 Functional Testing Tools

Apache JMeter is a popular open-source software testing tool primarily used for load testing and measuring the functional behavior of software applications. It stands out for

its ability to simulate heavy loads on servers, networks, or objects to analyze overall performance under different load types.

5.2 Cross-Device Testing Tools

In cross-device testing, we test an application across different devices. We ensure the application provides a smooth user experience and functions well on other devices like mobile phones, tablets, and desktops. So, no matter what device the user has, the application should behave consistently.

5.3 Performance Testing Tools

Performance testing is the practice of evaluating how a system performs in terms of responsiveness and stability under a particular workload. Performance tests are typically executed to examine speed, robustness, reliability, and application size

5.4 Accessibility Testing Tools

Accessibility Scanner:

A tool that suggests improvements to make your app more accessible to users with disabilities.

Helps identify areas where the app may be difficult to use for people with visual or motor impairments.

5.5 Security Testing Tools

QA in Security Testing is crucial because it's like having a security guard for your digital world. It checks if your software is strong enough to keep hackers out. Without QA, hackers could sneak in and cause problems. With QA, we can find and fix any weaknesses before they become big issues

5.6 UI/UX Testing Tools

UI/UX testing is the process of verifying the functionalities of the user interface (UI) and keeping the user experience (UX) in check. It helps explore critical bugs that can quickly magnify and may hinder performing basic operations on a web application.

6. Best Practices

To ensure thorough and effective Android app testing, consider the following best practices:

6.1 Test on Real Devices

Objective: Ensure that the app is tested on real devices in addition to emulators.

Activities:

Test the app on a wide range of physical devices, including different brands, screen sizes, and hardware configurations.

Use device farms or cloud-based testing services if physical devices are not available.

Outcome: A more accurate assessment of the app's performance and functionality in real-world conditions.

6.2 Prioritize Critical Test Cases

Objective: Focus on testing the most critical functionalities first.

Activities:

Identify and prioritize test cases based on the app's core functionalities and business impact.

Allocate more time and resources to testing high-risk areas that could severely affect user experience if they fail.

Outcome: Early identification and resolution of critical issues that could impact the app's success.

6.3 Involve Real Users

Objective: Gather feedback from real users to improve the app's usability and functionality.

Activities:

Conduct beta testing or user acceptance testing with a group of real users.

Collect and analyze user feedback to identify areas for improvement.

Incorporate user suggestions into the app's development and testing process.

Outcome: A user-centric app that meets the needs and expectations of its target audience.

6.4 Document Everything

Objective: Keep comprehensive records of all testing activities.

Activities:

Document test cases, test execution results, and defects in a centralized repository.

Keep a log of all testing activities, including the date, time, and results of each test.

Share test documentation with the development team to ensure transparency and collaboration.

Outcome: Clear and accessible records that can be referenced throughout the project.

6.5 Continuous Testing

Objective: Regularly test the app throughout the development process.

Activities:

Perform continuous testing as new features are developed and integrated into the app.

Conduct regression testing after each code change to ensure stability.

Use feedback loops to quickly identify and resolve issues during development.

Outcome: A high-quality app that is stable, reliable, and ready for release.

7.

Conclusion

By following these detailed guidelines and utilizing the recommended tools, testers can ensure comprehensive and effective Android app testing. This will lead to a more reliable, user-friendly, and high-performing Android application. Consistent application of best practices, thorough documentation, and ongoing collaboration are key to successful Android app quality assurance.