# **Standard Operating Procedure (SOP) Mobile App Regression Release Process**

## **1. Introduction**

This document outlines the Standard Operating Procedure (SOP) for the regression testing and mobile application release process. The purpose of this SOP is to ensure that every mobile application release is of perfect quality, stable, free from regressions of existing functionalities, and meets all business and technical requirements before being launched to end-users.

## **2. Objectives**

* To ensure existing functionalities continue to work correctly after code changes (new features, bug fixes, refactoring).
* To identify and report regressions as early as possible in the development cycle.
* To establish clear criteria for regression pass/fail and release decisions.
* To minimize risks and negative impacts on users after release.
* To increase confidence in the quality of released products.

## **3. Scope**

This SOP applies to all regression testing performed before mobile application (iOS and Android) releases to the production environment, and also covers the release decision-making process and post-release monitoring.

## **4. Definitions**

* **Regression:** The emergence of bugs or defects in functionalities that previously worked correctly.
* **Regression Cycle:** The period dedicated to executing regression test cases.
* **Release Candidate Build:** A version of the application considered stable and ready for final regression testing before release.
* **Entry Criteria:** Conditions that must be met before regression testing can begin.
* **Exit Criteria:** Conditions that must be met for regression testing to be considered complete and successful.
* **Post-Release Monitoring:** Monitoring application performance and stability after public launch.

## **5. Process Phases**

### **5.1. Regression Preparation**

When the Regression Cycle Begins:

The regression cycle begins immediately after a stable Release Candidate Build, free from known critical (blocking) bugs, has been made available by the Development team. This typically occurs after the initial feature testing or sanity testing phases are complete.

**Entry Criteria:**

* All critical bugs (Priority 1 & 2) from the previous feature testing cycle have been fixed and verified.
* The Release Candidate Build is available and successfully installed in the testing environment.
* The build has passed initial smoke tests by the QA team (core functionality works).
* All relevant regression test cases have been updated and are ready for execution.
* The testing environment (devices, network connection, backend) is stable and ready.

**Teams Involved:**

* QA Engineers (Primary Responsible)
* Development Team (For quick bug fixing and technical support)
* Product Owners (For functionality clarification)

**Environment:**

* Staging/UAT (User Acceptance Testing) environment that replicates the production environment as closely as possible.
* Physical mobile devices and/or emulators/simulators covering various OS versions and relevant device models.

### **5.2. Regression Cycle**

When is Our Regression Cycle:

The regression cycle is a critical phase that occurs after the release candidate build is submitted and before the final release decision is made. It is the last checkpoint to ensure stability.

The regression cycle will be conducted every two weeks.

Duration:

The duration of the regression cycle must be determined based on the complexity of changes, the size of the regression scope, and release deadlines. Generally, it can last from 1 day to 3 full working days, depending on the release scale.

**Types of Testing:**

* **Full Regression Testing:** Executing the entire comprehensive regression test case suite. Performed for major releases or releases involving significant changes to the core architecture.
* **Partial Regression Testing:** Executing a subset of regression test cases focusing on areas affected by recent changes and the most frequently used core functionalities. Performed for minor releases or hotfixes.
* **Automated Regression Testing:** Running automated regression test suites regularly (e.g., nightly or after every major commit). Automated results must be closely monitored.
* **Manual Regression Testing:** Performed for test cases that are difficult to automate, or for end-to-end validation and user experience.

**Tools:**

* **Test Management Tool:** JIRA, TestRail, Azure DevOps (for managing test cases, execution, and reporting).
* **Bug Tracking Tool:** JIRA, Bugzilla (for reporting and tracking bugs).
* **Automation Frameworks:** Appium, Espresso (Android), XCUITest (iOS) (if automated test cases exist).
* **Device Farm:** Firebase Test Lab, BrowserStack, Sauce Labs (for testing on various devices).

**Bug Reporting:**

* Any regressions found must be immediately reported in the bug tracking tool with complete details (replication steps, expected vs. actual result, screenshots, logs, priority, severity).
* High-priority bugs (P1/P2) must be immediately communicated to the Development and Product teams.

### **5.3. Regression Exit Criteria**

**Pass Conditions:**

* All planned regression test cases have been executed.
* No open P1 (Critical/Blocker) bugs.
* The number of open P2 (Major) bugs is below the agreed threshold (e.g., maximum 1-2 P2 bugs that do not affect core functionality).
* No P3 (Minor) or P4 (Cosmetic) bugs significantly affecting user experience.
* Regression test case pass rate reaches the agreed target (e.g., >98%).
* All relevant stakeholders have approved the regression test results.

**Fail Conditions:**

* New P1 (Critical/Blocker) bugs are found.
* The number of open P2 (Major) bugs exceeds the agreed threshold.
* Regression test case pass rate is below the agreed target.
* Testing environment is unstable.
* Key stakeholders do not approve the test results.

### **5.4. Release Decision**

When We Can Release:

The release decision can be made only after all Regression Exit Criteria are met and there is unanimous agreement from the core team (QA Lead, Product Manager, Development Lead, Release Manager).

**Under What Conditions Can We Release:**

* All regression exit criteria have been met.
* No known bugs that could cause significant financial, reputational, or security impact.
* The potential impact of any remaining minor bugs has been understood and accepted by Product and Management.
* All builds have been signed and are ready for distribution.
* A rollback plan (if post-release issues occur) has been prepared and communicated.
* Support and marketing teams have been informed and are ready.

**Roles and Responsibilities:**

* **QA Lead:** Provides final quality status report, recommends release or postponement based on regression results.
* **Product Manager:** Approves release based on feature readiness and risk acceptance.
* **Development Lead:** Approves release based on code stability and rollback capability.
* **Release Manager:** Coordinates the release process and communication.

### **5.5. Release Process**

**Technical Steps:**

* **Final Build Verification:** Final verification of the build to be released (e.g., hash, signature, version).
* **Deployment to App Store/Play Store:** Uploading the build to distribution platforms (App Store Connect, Google Play Console).
* **Staged Rollout (Optional but Recommended):** Gradually launching the application to a small percentage of users first (e.g., 1%, 5%, 10%) to monitor initial performance and feedback before a full release.
* **Go-Live:** Full release to all users.

**Communication:**

* Inform all stakeholders (Support, Marketing, Sales teams) about the release schedule and included features.
* Prepare external communication materials (release notes, announcements).

### **5.6. Post-Release Monitoring**

**Objective:**

* To detect critical issues that may have slipped through testing as early as possible.
* To monitor application performance in the production environment.
* To collect user feedback.

**Key Metrics:**

* **Crash Rate:** The rate at which the application crashes. Target: <0.1% sessions.
* **ANR (Application Not Responding) Rate:** The rate at which the application becomes unresponsive.
* **User Reviews/Ratings:** Changes in ratings on the App Store/Play Store.
* **API Error Rate:** The error rate for backend API calls.
* **Latency:** Application response time.
* **Resource Usage:** Battery, CPU, memory consumption.
* **User Feedback:** Through support channels, social media.
* **Conversion Rates / Key Business Metrics:** Whether the release positively or negatively impacts key business metrics.

**Monitoring Tools:**

* **Crash Reporting:** Firebase Crashlytics, Sentry, Bugsnag.
* **APM (Application Performance Monitoring):** New Relic, Datadog, Dynatrace.
* **Analytics Tools:** Google Analytics, Firebase Analytics, Mixpanel.
* **Customer Support Platforms:** Zendesk, Intercom.

**Issue Handling:**

* The Monitoring team must be vigilant and monitor metrics in real-time after release.
* If critical bugs are detected, immediately investigate.
* If bugs have a widespread impact, consider hotfix or rollback options (reverting to a previous version).
* Communicate status and solutions to stakeholders and users if necessary.

### **5.7. Evaluation and Retrospective**

**Objective:**

* To analyze the success of the release process.
* To identify what went well and what needs improvement.
* To learn from mistakes and successes.

**When Conducted:**

* Within **1-3 days after a full release** (especially for major releases).
* Periodically for minor releases.

**Output:**

* Retrospective Report covering:
  + Release summary and its outcomes.
  + List of issues encountered and how they were resolved.
  + Post-release quality metrics.
  + Lessons learned.
  + **Action Items:** Concrete list of improvements to be implemented in the next release cycle.

## **6. Appendices**

* [Link to] Test Plan Template
* [Link to] Bug Report Guidelines
* [Link to] Release Checklist
* [Link to] Rollback Plan