# **Standard Operating Procedure (SOP)**

# **Software Bug Reporting Process**

**Table of Contents**

[1. Document Control](#) 1

[2. Purpose](#) 1

[3. Scope](#) 1

[4. Definitions](#) 2

[5. Roles and Responsibilities](#) 2

[6. Procedure](#) 3

[6.1. Bug Identification and Initial Verification (QA Engineer)](#) 3

[6.2. Bug Reporting (QA Engineer / Client Success)](#) 3

[6.3. Bug Triage and Prioritization (Product Manager/Project Manager)](#) 4

[6.4. Bug Resolution (Developer)](#) 4

[6.5. Bug Retesting and Verification (QA Engineer)](#) 5

[6.6. Bug Closure (QA Engineer)](#) 5

[7. Flowchart (Optional but Recommended)](#) 5

[8. Related Documents](#) 6

[9. Revision History](#) 6

## **1. Document Control**

* **Document ID:** SOP-QA-001
* **Version:** 1.0
* **Effective Date:** [23rd June 2015]

## **2. Purpose**

This Standard Operating Procedure (SOP) outlines the standardized process for identifying, documenting, reporting, and tracking software defects (bugs) within **Practice League's** powerful solutions designed to simplify and enhance law firm operations. Adherence to this procedure ensures consistent and efficient bug management, leading to enhanced reliability, data integrity, and operational efficiency for law firms utilizing our platform, ultimately contributing to improved software quality and faster resolution times for critical legal workflows.

## **3. Scope**

This SOP applies to all individuals involved in the testing, development, and quality assurance of **Practice League's legal practice management software**. This includes Quality Assurance (QA) Engineers, Software Developers, Product Managers, Project Managers, and Client Success/Support teams who may receive initial bug reports from law firm users.

## **4. Definitions**

* **Bug/Defect:** An error, flaw, failure, or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways, particularly impacting legal workflows or data integrity within the Practice League platform.
* **Bug Report:** A formal document or entry in a bug tracking system detailing a discovered bug.
* **Bug Tracking System (BTS):** Software used to log, track, and manage bugs (e.g., Jira, Azure DevOps, Asana, GitHub Issues).
* **Severity:** The impact of the bug on the system's functionality, user experience, or potential legal/compliance implications (e.g., Critical, Major, Minor, Cosmetic).
* **Priority:** The urgency with which a bug needs to be fixed, considering its impact on critical legal operations, data security, and client usability (e.g., High, Medium, Low).
* **Legal Workflow:** A sequence of tasks or processes commonly performed by law firms, such as case management, document generation, billing, or client communication, facilitated by Practice League software.

## **5. Roles and Responsibilities**

* **QA Engineer:**
  + Identifies, verifies, and thoroughly documents bugs, with a keen understanding of their potential impact on legal workflows.
  + Creates detailed bug reports in the BTS, ensuring all necessary context for reproduction and resolution.
  + Performs retesting of fixed bugs, validating against original requirements and potential side effects.
  + Closes verified bugs after successful retesting and regression.
* **Developer:**
  + Reviews assigned bug reports, reproduces, and diagnoses bugs, focusing on root cause analysis.
  + Develops and implements robust and secure bug fixes, ensuring data integrity and system stability.
  + Performs thorough unit and integration testing of fixes.
  + Updates bug status in the BTS and provides clear resolution notes.
* **Product Manager/Project Manager:**
  + Prioritizes bugs based on their impact on legal operations, client satisfaction, business objectives, and regulatory compliance.
  + Monitors overall bug resolution progress and ensures alignment with release cycles.
  + Communicates bug status and impact to internal stakeholders and, where necessary, to affected law firm clients.
* **Client Success/Support Specialist:**
  + Serves as the first point of contact for law firm clients reporting issues.
  + Gathers initial bug details from clients and logs preliminary reports in the BTS.
  + Communicates bug status updates and resolutions back to the client.

## **6. Procedure**

### **6.1. Bug Identification and Initial Verification (QA Engineer)**

1. **Execute Test Cases:** Perform testing as per defined test plans, exploratory testing, or user acceptance testing scenarios, with a focus on critical legal workflows.
2. **Identify Discrepancy:** Observe any behavior that deviates from expected functionality, legal requirements, or design specifications.
3. **Reproduce Bug:** Attempt to consistently reproduce the bug across various environments and configurations relevant to law firm clients.
4. **Isolate Issue:** Determine the specific steps, data (e.g., case data, client details), and environment conditions (e.g., browser, OS, specific Practice League module version) that trigger the bug.

### **6.2. Bug Reporting (QA Engineer / Client Success)**

1. **Access BTS:** Log in to the designated Bug Tracking System (e.g., Jira).
2. **Create New Issue:** Select "Create Issue" and choose the "Bug" issue type.
3. **Populate Fields:** Fill in the following mandatory fields:
   * **Summary:** A concise, descriptive title of the bug, clearly stating the affected functionality (e.g., "Document generation fails for specific client template").
   * **Description:**
     + **Steps to Reproduce:** Numbered, clear, and precise steps to reproduce the bug. Include specific data inputs or configurations if relevant.
     + **Expected Result:** What the system *should* do, based on requirements or legal expectations.
     + **Actual Result:** What the system *actually* does, detailing the erroneous behavior.
     + **Environment:** Specific Practice League module, browser, OS, build/version, and any relevant test data or client configuration.
     + **Impact on Legal Workflow:** Briefly describe how this bug affects a law firm's daily operations or compliance.
   * **Attachments:** Include screenshots, video recordings, console logs, or relevant error messages that illustrate the bug. Ensure sensitive client data is anonymized before attachment.
   * **Severity:** Assign based on impact (e.g., Critical - data loss, system crash; Major - key feature unusable; Minor - usability issue; Cosmetic - visual glitch).
   * **Priority:** Assign initial priority (e.g., High - blocks critical legal work; Medium - impairs efficiency; Low - minor inconvenience).
   * **Component/Module:** Specify the affected part of the Practice League platform (e.g., "Case Management," "Billing," "Document Automation").
   * **Assignee:** Initially assign it to the relevant Product Manager or Team Lead for triage.
   * **Labels/Tags:** Add relevant tags (e.g., document-automation, billing-error, data-integrity, security).
4. **Submit Report:** Save the bug report in the BTS.

### **6.3. Bug Triage and Prioritization (Product Manager/Project Manager)**

1. **Review New Bugs:** Regularly review newly reported bugs in the BTS, paying close attention to their potential impact on law firm operations and data.
2. **Validate and Clarify:** Ensure the bug report is clear, reproducible, and complete. Request clarification from the QA Engineer or Client Success Specialist if needed.
3. **Assess Impact:** Evaluate the bug's severity, its impact on critical legal workflows, potential data integrity issues, and any compliance risks.
4. **Assign Priority:** Adjust the priority based on business impact, client urgency, regulatory requirements, and resource availability.
5. **Assign Developer:** Assign the bug to the appropriate Developer or development team for resolution.
6. **Update Status:** Change bug status to "Open," "In Progress," or "Backlog" as appropriate.

### **6.4. Bug Resolution (Developer)**

1. **Reproduce and Analyze:** Reproduce the bug in the development environment and perform a thorough root cause analysis, considering system architecture and data flow.
2. **Develop Fix:** Implement the necessary code changes to resolve the bug, adhering to secure coding practices and ensuring data integrity.
3. **Unit & Integration Testing:** Perform comprehensive unit tests and integration tests to ensure the fix works as intended and does not introduce new issues or regressions.
4. **Code Review:** For critical bugs or complex fixes, ensure a peer code review is conducted to maintain code quality and identify potential issues.
5. **Update Status:** Change bug status to "Resolved" or "Ready for QA" in the BTS.
6. **Add Resolution Notes:** Provide clear and concise details about the fix in the bug report (e.g., "Addressed race condition in multi-user document editing; implemented transactional updates for billing records").

### **6.5. Bug Retesting and Verification (QA Engineer)**

1. **Review Resolved Bugs:** Access bugs with "Resolved" or "Ready for QA" status in the BTS.
2. **Deploy Build:** Ensure the fixed build is deployed to the QA environment, reflecting the latest code changes.
3. **Retest:** Follow the "Steps to Reproduce" from the original bug report to rigorously verify the fix.
4. **Regression Testing:** Perform targeted regression testing in the affected module and related functionalities to ensure no new issues were introduced and existing features remain stable.
5. **User Acceptance Testing (UAT) (Optional but Recommended for Major Bugs):** Coordinate with Product Management or Client Success for UAT if the bug had significant client impact, using mock law firm data.
6. **Update Status:**
   * If fixed and verified: Change status to "Closed."
   * If not fixed or new issues found: Reopen the bug, add detailed comments explaining the failure, and change status back to "Reopened" or "In Progress."

### **6.6. Bug Closure (QA Engineer)**

1. **Confirm Closure:** Ensure the bug has been successfully retested, verified, and all associated regression checks have passed.
2. **Final Status:** Set the bug status to "Closed" in the BTS.
3. **Archive (if applicable):** Once closed, the bug report is archived for historical reference and future analysis.

## **7. Flowchart (Optional but Recommended)**

(You can describe a simple flowchart here or create one visually if you have the tools)

A visual representation of the bug reporting lifecycle within Practice League, from identification by QA or Client Success to final closure, highlighting key decision points and handoffs between teams.

## **8. Related Documents**

* Practice League Software Testing Strategy Document
* Test Plan Template
* Release Management SOP
* Data Security and Privacy Policy
* Compliance Guidelines for Legal Software
* Client Feedback and Support Process

## **9. Revision History**

| **Version** | **Date** | **Description** |
| --- | --- | --- |
| 1.0 | [Date] | Initial Draft |