# **DRAFT: Bug Reporting Procedure**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ver.** | **Date** | **Description** | **Changed by** |
| **1.96** | **2/8/15** | **Bug reporting procedure** | **Rami** |

|  |  |  |
| --- | --- | --- |
| **Stage** | **Role** | **Name** |
| **Author/Lead** | QA Lead | **Nahum** |
| **Reviewer** | Release Manager/  Product Manager/  Engineering Lead | **Rami** |
| **Approver** | Director QA |  |

**Table of Contents**  
**1** **Introduction**   
1.1 Purpose   
1.2 Scope   
1.3 Roles & Responsibilities (in the context of this process)   
1.4 References   
1.4.1 Zendesk P-level table (Priority reported by customer/CS/SE)   
**Table 1.4.1 – Zendesk P-Levels (Priority reported by customer/CS/SE)**   
1.4.2 Zendesk Case Status table   
**Table 1.4.2 – Zendesk Case Status**   
1.4.3 Priority table (QA reported)   
**Table 1.4.3 – Priority levels (QA reported)**   
1.4.4 Bug tracking Status table   
**Table 1.4.4 – Bug Status in Jira**   
1.4.5 Bug Severity table (Planning to redefine as per +<https://jira.vidyo.com/browse/AM-2027+> )   
**2** **Procedure (Steps)**   
2.1 Bug inception through Customer/SE/CS   
2.2 Bug inception through QA/DEV/Others   
2.3 Bug Monitoring and Reviews   
2.4 Bug Resolution   
2.5 Bug Validation & Closure   
2.5.1 Bugs in RETURNED state:   
2.5.2 Bugs in PASSED UNIT TEST / PENDING TEST Q state:   
2.6 Bug closure - CS   
**3** **Bug Workflow chart**   
**4** **Jira Bug Reporting Guidelines**   
4.1 Before opening new bug in Jira collect following information:   
4.2 Enter new Bug in Jira and provide following details:   
4.3 Edit Bug in JIRA and provide following optional details if necessary:   
**Appendix E – Document Change Control**

# **Introduction**

## **Purpose**

To streamline the bug reporting and resolution process among the different departments, and ensure the proper handover and accountability throughout the lifecycle of the bug.

## **Scope**

The procedure covers the entire bug lifecycle, from inception through closure, and details the actions taken by the participants (CS, QA, DEV, PM) and triggers for those actions.

## **Roles & Responsibilities (in the context of this process)**

CS – Main interface with customer/SE  
QA – Bug identification, validation and prioritizing  
Tier 4 – Responsible for reporting bugs in JIRA based on field cases/incidents.  
DEV – Bug resolution  
PM – (Product Manager) Requirement definition, deciding on GAB & Fix version

## **References**

The listed tables detail the reported bug attributes:

### **Zendesk P-level table (Priority reported by customer/CS/SE)**

As reported by the customer/CS/SE (if initiated in the field or internal beta). The importance of this ranking is in reflecting the customer specific experience.   
**Table 1.4.1 – Zendesk P-Levels (Priority reported by customer/CS/SE)**

|  |  |  |
| --- | --- | --- |
| **SL** | **Description** | **Workaround** |
| P0 | Complete Outage | Immediate solution |
| P1 | Partial Outage | One of the next versions or a patch |
| P2 | Degraded by system performance | Future versions |
| P3 | Degraded by system performance | Future versions |
| P4 | Request for information? | Future versions |

### **Zendesk Case Status table**

This is managed by CS & Tier 4, indicating the bug status to be reported to the customer/Case Owner. Tier-4 starts working the cases after it reaches Pending QA. All other States are listed here for sake of completeness.  
**Table 1.4.2 – Zendesk Case Status**

|  |  |
| --- | --- |
| **Status** | **Description** |
| NEW | New incident opened, additional information required |
| Pending-NJ CS Eval | Enough relevant information exists to handle. |
| In Process-NJ CS Eval | CS is actively evaluating the case |
| Pending – Customer FB | Used by CS to collect customer feedback |
| Pending – Case Owner FB | Pending Case Owner Feedback is typically used when a case is returned by Tier4 because there is no bug or additional info is needed from the customer |
| Pending QA | Tier 4 starting point |
| Pending RnD Eval | Escalated by Tier 4 from "Pending QA" once the incident reported by CS is confirmed to be a bug |
| Pending Resolution | A fix has been delivered into a build (but not verified) or a fix is being targeted for a future build. |
| Pending GA | Used for cases where the fix has been validated by QA |
| Pending Upgrade | This state is used when the release has gone GA and customers can be instructed to upgrade |
| On Hold | Unable to continue without external information / action (e.g. happened once, cannot reproduce) |
| Cancelled | Case withdrawn |
| Closed | Case closed |

### **Bug tracking Status table**

Bug status indicates the current state of the bug within its lifecycle in the bug tracking system.   
**Table 1.4.4 – Bug Status in Jira**

|  |  |
| --- | --- |
| **Status** | **Description** |
| NEW | Newly entered bug and assigned to PM/Dev lead |
| ASSIGNED | Bug has been reviewed by PM/DEV lead and assigned to a Developer |
| RETURNED | No clear requirements, invalid scenario. Returned to QA |
| PASSED UNIT TEST | Bug has been resolved and unit tested by DEV (and waiting for build.) |
| PENDING TEST (Q) | ~~QA is working on validating the fix.~~ (To be replaced by Build is ready for testing) |
| REOPENED | Bug resolution has not been qualified. Bug is reassigned to DEV |
| CLOSED | Bug has been resolved by DEV and qualified by QA. |
| CLOSED – No change | Bug has been closed without any code change. Invalid Bug, change in requirement, etc. |
| CLOSED - Duplicate | Another bug exists for same/similar scenario. |
| ARCHIVED | Old, irrelevant bugs from non-supported versions |

### **Bug Severity table (Planning to redefine as per** [**https://jira.vidyo.com/browse/AM-2027**](https://jira.vidyo.com/browse/AM-2027) **)**

Severity is the weighed result of all decision factors (i.e. Priority & Environmental factors). This is the most important rank determining the DEV priorities in resolving open bugs.

# **Procedure (Steps)**

A bug can be initiated by an SE, QA, DEV or other source.

## **Bug inception through Customer/SE/CS**

1. If CS cannot resolve/dismiss the Zendesk cases reported by customer, the case is escalated to *PENDING QA* state.
2. Tier 4 team will process cases in *PENDING QA* state and try to reproduce them. If bug cannot be reproduced/user error, QA will change Zendesk status to *PENDING CASE OWNER FEEDBACK* and provide the pertinent information. If the issue is reproduced, Tier 4 creates Bug in Jira with status *NEW*, with Priority level, (see table 1.4.2) **assigned to product PM, no Fix version**.
3. For certain cases which are not reproducible but have sufficient information for R&D to proceed, Tier 4 will open Bug with all the relevant information.

*Note: Bug Priority level may be different than Zendesk P-levels, based on QA evaluation of the bug.*

## **Bug inception through QA/DEV/Others**

1. Bug initiated by QA, entered on bug tracking system with status NEW, assigned to DEV lead (which is automatic assignee in Jira) with assigned Priority level (see table 1.4.2).
2. QA may initiate an incident report same as bug but Reproducible field is marked as No and Bug assigned to QA. Then assigned QA engineer should try to reproduce it later and update Bug with necessary information and assign bug to DEV.
3. Bug initiated by DEV/Others, entered in bug tracking system with status NEW, assigned to DEV lead.

## **Bug Monitoring and Reviews**

1. Depending on project phase Bug reviews will be conducted at least once every 2 weeks for all GAB, Blockers with QA, DEV, PM & CS teams.
2. If DEV/PM questions a specific bug priority or GAB assignment, QA Lead will provide reasoning/consider modification. However, only PM is authorized to modify bug priority or Fix Version/s.

## **Bug Resolution**

1. If DEV is unable to reproduce the bug or root cause of the bug based on logs or code review, send back the Bug to QA with request for more information (assigned to QA Lead or QA person initiating the bug). If able to reproduce, continue to next step.
2. If DEV is able to reproduce the bug or fix the bug based on logs or code review, DEV produces a fix and performs unit test.
3. DEV updates bug status to *PASSED UNIT TEST* in the bug tracking system, and adds relevant details on the bug fix (e.g. expected build number/ETA in QA, specifics of what has been resolved and how).
4. DEV updates the Change Log accordingly.
5. DEV moves bug to *PENDING TEST Q* when build is delivered to QA

## **Bug Validation & Closure**

### **Bugs in RETURNED state:**

Development/PM needs QA to re-examine this issue and provide more information, QA should try to reproduce this bug in the current build. If not reproducible in the current build then QA should try it on the build the issue was observed. In all the cases QA should provide the information back to developer/PM.

* If reproducible ASSIGN the bug to DEV with additional information.
* If observed in previous build but not current : Should be closed as with fix
* If not observed in previous also then should be closed as without fix.

### **Bugs in PASSED UNIT TEST / PENDING TEST Q state:**

* Development has resolved this issue and QA need to validate the fix. QA starts validating bugs in *PENDING TEST Q* state staring with GAB and Priority. If issue is fixed, move to CLOSED with proper comments and update fields Closed in Tag (QA) and Fix Validated in Environment.
* If issue is not fixed REOPEN the issue with proper information and new logs
* If bug having multiple issues is partially fixed, update comments for each issue. If majority of the issues are fixed, close the existing Bug and raise a new Bug for remaining issues.

1. If validated on a specific tag but not yet available in GA, Zendesk status is changed to *PENDING GA*
2. After GA, Zendesk status is changed to *PENDING UPGRADE*

## **Bug closure - CS**

1. CS updates customer/SE on resolution via Zendesk.

# **Bug Workflow chart**

# **Jira Bug Reporting Guidelines**

## **Before opening new bug in Jira collect following information:**

1. Collect the logs/traces for all non UI issue and screenshot/movie clip for UI issues.
2. Reproduce the issues few more times to know the Reproducibility
3. Try to reproduce it on other environment with same build.
4. Try to reproduce with previous Release / Build if necessary to isolate the component/module.
5. Search in Jira if same/similar issue is already reported
6. If bug exists in RETURNED/CLOSED state, REOPEN the issue with current build, environment, logs, Screenshots.
7. If bug exists in OPEN state, add this incident and update the issue with current build, environment, logs, Screenshots.
8. If issue does not exist in Jira proceed to next section.

## **Enter new Bug in Jira and provide following details:**

1. **Project:** Select correct Project
2. **Issue Type:** Select Issue Type as "Bug"
3. **Summary:** Give a short and precise description of the bug. This Summary will be displayed in the table when searching for issues. A good summary should quickly and uniquely identify a bug and should be possible to tell what this bug is about. You can prefix the summary with feature /OS/Hardware/Browser to quickly identify the context.
4. **Priority:** Refer to section 1.4.3
5. **Blocks:** Select one options below (listed in the order of Severity)

**GA Blocker:** For GA blocker   
**Beta Blocker:** external beta blocker (Higher than Main blocker)  
**Main Blocker:** external public beta blocker   
**Main2 Blocker:** Not ready for internal use.  
**Best Effort:** internal beta blocker.  
\*\*Any feature not matching with Requirement can be a GA blocker.

1. **Component/s:** Provide the component impacted by the issue. From the list, select the components the bug is related to, to the best of your knowledge. You can select multiple options
   * 1. **Room Models:** Only applicable for PanoRoom or VidyoRoom bugs
2. **Detected in Tag** field is now deprecated. You can use the data that is already exist in this field (read only), but don't add data.
3. **Tag Number**: Input accepted is one number. In case multiple versions are listed in Affected Version/s field, please mention the tag number that corresponds with the latest affected version.
   * 1. **Reproduced in Environment:** Enter the Environment where the issue was observed. eg. Windows XP, Internet Explorer 7, QA1 portal. Hardware type if applicable.
     2. **Reproducible:** Provide the proper reproducibility.
     3. **Affects Version:** this is a multi-choice field that has the same values as Fix Version field. e.g. "3.5", "3.5.2").
     4. **Fix Version/s:** Leave this blank for Zendesk bugs. For all other bugs mostly it will be same as observed, When product is nearing GA, Fix version/s can be next version (TBC with PM)
     5. **Assignee:** Leave it as "Automatic ".
4. **Reporter:** Leave it as default (yourself)
5. **Description:** Give following information in Description

Overview: Give detailed description of the Bug  
Steps to Reproduce: Provide list of steps for reproducing the issue.  
Expected Result:  
Observed Result:  
Reproducibility: Additional details to support info in Reproducible field.  
Date & Time of observation: Provide the time when the issue was observed, it will help developer to look into the logs  
Related Issue: If this issue is related to any other JIRA issue (bug, task, requirement, MMF) provide the Jira number of related issue. Many bugs which are similar it would be very helpful if QA could mark such MRs as "relate to" to provide more context to the developer reviewing the bug and to the PM reviewing the severity and assessing how widespread the issue is.  
Requirement/Reference: Provide the requirement (if available) reference like link to PRD, MMF, Guides, Test Plans, email confirmation, etc.  
Include other relevant information like your screen resolution, special plugins, peripherals, network conditions, etc.

1. **Attachments:** Attach the logs, screen shot, short movie clip, Diagnostics reports, crash logs etc. Refer to product specific log requirement.

## **Edit Bug in JIRA and provide following optional details if necessary:**

1. **Customer Detected, Customer Name, Zendesk case #, Zendesk Priority:** Enter relevant information for bugs opened as a result of Zendesk case.
2. **Organization:** Enter TATA\_Bug for MRs opened by Tata team
3. **Bug Exists in previous Anchor:** Verify if this issue was observed in previous GA/Anchor build and update this field. (Strongly recommended for test blockers and GA blockers)
4. **Last version known to work:** This is important field to get statistics for broken features.

## **Known Issues / Knowledge Base list (refers to QA Managers)**

In case a defect should go into the Release Notes, mark the field Knowledge Base as "Internal + External". Then create a filter than lists all the issues that have "Internal + External" in the Knowledge Base field and publish it under the master PDM for the project in Tech Pubs tab -> KB Link field (ask the relevant Project Manager in case you are not sure).