

**Guidelines for Test Cases Execution**

|  |  |
| --- | --- |
| **Priority** | High |
| **Classification Information** | IV |
| **Document Version** | 1.0 |
| **CureMD Version** | 10g |
| **Creation Date** |  |
| **Author** |  |
| **Status** |  |
| **Approver** |  |

**Review History**

|  |  |  |
| --- | --- | --- |
| **DATE** | **DESCRIPTION** | **DONE BY** |
| 10/31/2018 | Test Case Execution Guidelines |  |
|  |  |  |
|  |  |  |

# Test Case Execution Contents:

While execution of Test Case Tester should add following contents:

1. **Actual Result:** Actual test result should be filled after test execution if ‘Status’ is Fail. Describe the system behavior after test execution.
2. **Status (Pass/Fail):** If actual result is not as per the expected result, then mark this test as failed. Otherwise, update it as **passed.**
3. **Test Executed By:** Name of the Tester who executed this test. To be filled only after test execution.
4. **Test Execution Date:** Date when the test Case was executed.
5. **Defect ID/Link:** If the Test Case status is failed then log a Defect in QC.

# Defect Reporting Guideline:

1. **Defect ID (M):** Unique ID is required for each Defect.
2. **Detected By (M):** Name of the Tester who is developing Test Cases.
3. **Detected on Date (M)**: Date when defect is logged.
4. **Subject (M)**: Area of Application (List is added in QC)
5. **Severity (M)**: Severity is defined as the degree of impact a defect has on the operation of the application being tested. Tester selects one of the following values:
   * **Critical**: This defect indicates complete shut-down of the process, nothing can proceed further.
   * **Major**: It causes some undesirable behavior, but the system is still functional. It may include Missing Functionality as well.
   * **Minor**: It won't cause any major break-down of the system.
6. **Test Cycle (M)**: Defines the cycle information in which defect identified. Tester selects one of the following values:
   * Document Review
   * Acceptance
   * 1st Cycle
   * 2nd Cycle
   * nth Cycle

Note: when ‘n’ crosses 3rd Cycle, a major review of the Design is required. An appropriate flag will be raised at this point. Management may decide to hold the Release and go back to the drawing board.

1. **Detected in Version (M)**: Indicates application version on which defect is identify.
2. **Assigned To (M)**: Name of the person to whom defect is to be assigned for further action.
3. **Module (M)**: Indicates the Application module information.
4. **Category (M)**: Differentiates the nature of the defect. Tester selects one of the following values:
   * **Crashing Defect**: When the page/area breaks down.
   * **Functionality Defect**: When requirements have been implemented incorrectly.
   * **Missing Functionality**: When a given requirement is missing.
   * **GUI Defect**: When Defects is related to GUI elements.
   * **Scripting Error**: When Defect is about the scripting errors.
   * **Functionality Suggestion**: When Defect is about the functional suggestion from Tester.
   * **GUI Suggestion**: When Defect is about the GUI suggestion from Tester.
   * **Performance**: When Defect is about not meeting the required Benchmark.
   * **Security Issue**: When Defect is about not meeting the security requirement.
   * **Question**: When the defect is of question nature. This value can only be selected when the CR state is ‘QA/Dev Review’ and the Test Cycle is ‘Document Review’.
5. **Priority (M)**: Priority is defined as the order in which a defect should be fixed.
   * **High:** The defect must be resolved as soon as possible as it affects the system severely and cannot be used until it is fixed.
   * **Medium**: During the normal course of the development activities defect should be resolved.
   * **Low**: The defect is an irritant but repair can be done once more serious defects have been fixed.
6. **Project (M)**: Indicates the project/CR reference against which Defect is identified.
7. **Server (M)**: Indicates server information on which defect is identified.
8. **Data Base (M)**: Indicates backend Data Base which is used to reproduce the Defect.
9. **Dev Team (M)**: Name of DEV Team to whom defect is to be assigned for further action.
10. **Status (M)**: Defines the current state of Defect. Can be one of the following:
    * **New:** When a Defect is generated initially.
    * **Fix on Dev:** When a Defect is fixed at Dev side.
    * **Fixed:** When Dev provides the fixes to Tester.
    * **Open:** When Defect is not fixed properly and needs to be reassigned back to Dev Team.
    * **Reopen:** When Defect is not fixed in subsequent iteration and needs to be reassigned to Dev Team again.
    * **Closed:** When Defect is fixed and verified by Tester. While selecting this status Tester must provide related comments in comment section.
    * **Next Release:** When Defect is not going to be fixed in current Release and will be fixed in future. While selecting this status DEV manager’s comments are mandatory.
    * **Tool Limitation:** When Defect can’t be fixed due to any Technical/Tool related limitations. While selecting this status DEV manager’s comments are mandatory.
    * **Rejected:** When reported Defect is not accepted by Dev/Analysis Team as Defect.
11. **Browser (O):** Tester selects the appropriate Browser if Defect is Browser specific.
12. **Required in Patch (M):** it’s a custom field used on need basis to take some run time decisions at Manager’s level. Default value of this field is ‘Y’. its scope is limited to internal Testing Cycles only.
13. **Summary (M)**: A short description about the Defect.
14. **Description (M)**: Defines the complete information about the Defect. This section will include following:
    * **Work Flow (M):** Tester Specify ‘WorkFlow’ in Description section.
    * **Pre-conditions (O)**: Any prerequisite that must be satisfied against Defect, if required.
    * **Description/Summary (M):** Specify ‘Summary’ against Defect. Summary should define the objective of Test Case/Defect.
    * **Steps to Reproduce:** Specify ‘Steps’ to Reproduce the reported Defect.
    * **Expected Result (M)**: Expected test results should be specified i.e. it should define the expected results of the test case.
    * **Actual Result (M):** Actual test result should be filled after test execution. Describe the system behavior after test execution.
15. **Attachments/References (O):** Attach screen shot against Defect if required.
16. **Comments (M/O):** User provides related comments in ‘Comment’ section while marking the Defect status. It is mandatory for the Tester to provide comments while marking the Defect status as ‘Closed’.

# Defect Reasons Guidelines:

Below is the defect reason which are selected by developer when marking bugs as **Fixed** in **QC**.

|  |  |  |
| --- | --- | --- |
| **Bug Category** | **Bug Reason** | **Description** |
| **01 DESIGN** | Incorrect Requirement | Problem occurred because functionality which was proposed/Implement is invalid as its not compliant with the same or other application domain/work flow or simply an invalid requirement due to any reason |
| Undocumented Specs | The bug was reported because things were done verbally between Analysis and Development but not communicated to QA in writing |
| Specs Modified | Problem occurred because functionality implemented is not similar to CR document and it is accepted as improved/better implementation over documented functionality |
| **02 CODE** | Design | The Problem raised due to Bad application design ~~and can’t be fixed permanently~~. Temporary fix provided for the particular situation or Bug fixed via design changed |
| Missing Code | A condition was left out |
| Missing Functionality | Functionality is not implemented or partially implemented |
| Rounding Error | The calculation formulae is ok but the value is being rounded, truncated. |
| Unhandled Exception | Unhandled exceptional scenario at potential problem areas |
| Multithreading/Deadlock | Race condition occurred in the server environment |
| Unit Testing | The area/page being updated was not fully reviewed after making changes.(This category includes GUI issues as well) |
| Ripples of a Recent Change | Updates in one module/page of the code caused a side effect in “another” module/page |
| Disturbed by other | Code changed by someone else affecting your portion of implementation |
| Regression | Problem already exist in the code/Feature and not occurred due to the recent change but fixed by dev with the recent change scope |
| **03 DATA** | Missing Elements | An external interface violated the specifications. An element or data value was expected but not found |
| Overflow | The data from the external system/user overflowed |
| Special Characters | Problem occurred due to entry of special characters which went unhandled/un-escaped |
| Migration | Problem occurred due to migration of data from external vendor to CureMD or from old version to newer CureMD version |
| **04 CONFIGURATION** | Hardware/Server Failure | Server was down/slow due to hardware failure or some other reason |
| Improper Deployment | Improper deployment by SCM/Systems department OR changes done directly/SCM ticket on live server was not synched locally in TFS |
| Incomplete check-in | Problem occurred because developer did not send complete upload. ASPs/SPs/DLL/Script missed in upload |
| Invalid/Wrong Index | Invalid or no index on DB |
| Merging Issue | Problem occurred due to code reverted or missed during Code Merging activity at any level |
| **05 TRAINING** | Not given to Developer | Problem occurred because developer was not trained on business/system domain or client /product requirements |
| Not given to QA Resource | Problem reported because QA resource was not trained on business/system domain or client /product requirements |
| **06 OTHER** | Duplicate Bug | Bug already reported by same or different QA resource |
| Forward Compatibility | Bug is occurring only on latest technology like IE, Chrome, Safari, Edge, Windows Vista, Windows 8 etc. |
| Not Reproducible | Bug reported in QA environment is not reproduced either in development environment or it’s automatically fixed somehow in QA environment later |
| Tool Limitation | Bug cannot be fixed due to technology/tools limitation being used |
| Rejected | Not a bug |
| QA Suggestion | A suggestion reported by QA |
| Obsolete | Reported Bug is no more valid in the current implementation or some work flow has been changed due to which there is no need to fix the reported Bug |
| Query Entertained | Reported Bug is a Query ask by QA against Document Review activity. (can only be used against the Question Type Bugs) |