**Bug reporting guideline**

IMPORTANT NOTE:

All of the data in the file below is just a template. Respect the structure but adapt it to the project, bug tracking system and customer requirements. It must not be followed by the letter. Some bug tracking systems might not have all the fields mentioned below, others might have extra fields. The most important thing is to try and standardize the bug reporting process and make sure that if a person from the outside comes and sees your issue, they can understand it and reproduce it.

* **Bug life cycle:**

Below you have a flow chart describing the life cycle of an issue from the moment it is reported to the moment it is fixed and closed. Please note however that this is a guideline and depending on the bug tracking software and project requirements the name of the stages or the actual flow might be different.



* **Priority/Severity**

**Priority:** concern with application from the business point of view. It answers: How quickly we need to fix the bug? Or how soon the bug should get fixed? **Severity:** concern with functionality of application. It deals with the impact of the bug on the application. It answers: How much the bug is affecting the functionality of the application?

Examples:

* High Priority and Low Severity:  
  Company logo is not properly displayed on their website.
* High Priority and High Severity:  
  Suppose you are doing online shopping and filled payment information, but after submitting the form, you get a message like "Order has been cancelled."
* Low Priority and High Severity:  
  If we have a typical scenario in which the application get crashed, but that scenario exists rarely.
* Low Priority and Low Severity:  
  There is a mistake like "You have registered success" instead of successfully, success is written
* **Bug reporting template:**

(Component or area tested – Environment it occurs on*\**) A few words describing the issue

**Actual:**

Detailed information about the issue. What the application did when the steps have been performed.

Please see the *attached/following screenshot/video* for more details*\*\**.

Reproducibility: Issue occurred x times out of x attempts*\*\*\**.

**Expected:**

What the application should have done, if there was no bug.

**Steps to reproduce:**

Minimized, easy-to-follow steps that will trigger the described problem. Include any special setup steps.

\* - If the environment is not relevant do not mention it

\*\* - Recommended. When situation demands it.

\*\*\* - This field is only required if the issue does not have a 100% reproducibility rate

* **Language**

Please make sure you use appropriate and professional language when writing the issues. This also applies when providing feedback to a previously reported issue. Provide clear and concise information when asked to do so.

* **General tips to take into consideration**
* The first aim of a bug report is to let the programmer see the failure with their own eyes. If you can't be with them to make it fail in front of them, give them detailed instructions so that they can make it fail for themselves.
* In case the first aim doesn't succeed, and the programmer *can't* see it failing themselves, the second aim of a bug report is to describe what went wrong. Describe everything in detail. State what you saw, and also state what you expected to see. Write down the error messages, *especially* if they have numbers in.
* By all means try to diagnose the fault yourself if you think you can, but if you do, you should still report the symptoms as well.
* Be ready to provide extra information if the programmer needs it. If they didn't need it, they wouldn't be asking for it. They aren't being deliberately awkward. Have version numbers at your fingertips, because they will probably be needed.
* Write clearly. Say what you mean, and make sure it can't be misinterpreted.
* Above all, *be precise*. Programmers like precision.

**Example no. 1 of an issue following the above template:**

(Login - Chrome) Certain user is unable to complete the login process

Actual:

It was observed that even though the user “admin” using the password “123” exists in the database, when using the credentials to login, an error message is received “Wrong credentials” and cannot complete the login process.

Please note that this issue occurs only on Google Chrome.

Reproducibility: This issue occurred 3 times out of 10 attempts.

Please see the following screenshot for more details:

[http://www.linktoscreenshot.com](http://www.linktoscreenshot.com/)

Expected:

The user should be redirected to the dashboard after using the mentioned credentials.

Steps to reproduce:

* Access <http://www.testedwebsite.com/login> on Google Chrome
* Login using the username “admin” and password “123”
* Observe that an error message appears and the user is unable to complete the login process

**Example no.2 of an issue following the above template:**

(BackOffice) No validation for mandatory fields on the Questions page

Actual:  
When the user presses the “Create” button for a blank page the mandatory fields are shown highlighted with errors. It was observed that for the Questions page in the Questionnaire section, there is no validation for the “Answers” field even though it is marked with an asterisk(\*).  
Please observe the attached screenshot/video for more details.

Expected:  
All fields marked with \*(asterisk) should have validation when left blank and completing the page.

Steps to reproduce:  
1. Access <http://www.testedwebsite.com/login> on any browser  
2. Login using an administrative account (username: admin, password:123)  
3. Go to Questionnaire > Questions from the left side navigation menu  
4. Click on the “Add new” button from the top right of the page  
5. On the new page scroll down and click on the “Create” button  
6. Observe that the “Answers” field does not have validation and returns no error even though it is marked as mandatory