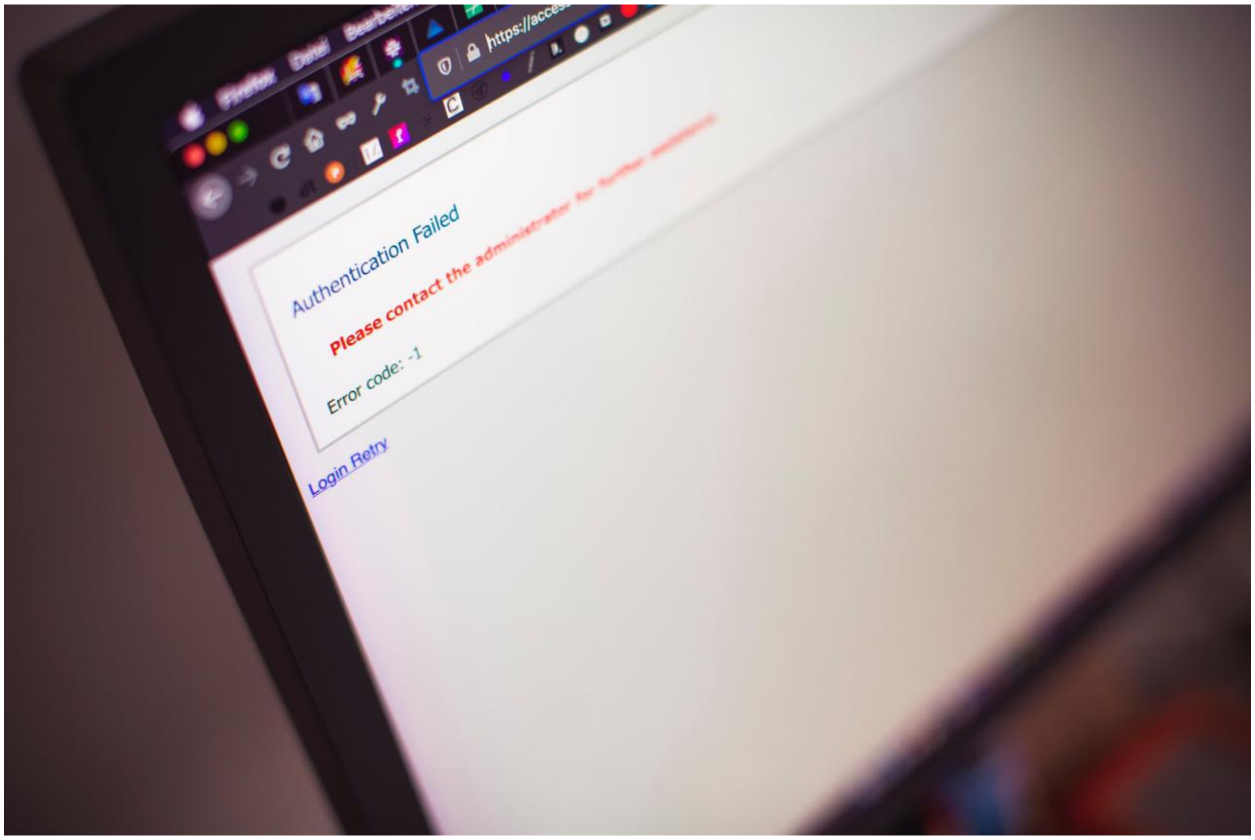
How To Write a Bug Ticket Using Our Best Practices Template



Authentication Failed

Please contact the administrator for further assistance

Error code: -1

Login Retry

Whether you work as a QA officer or a project manager or even a product owner or a developer, you might have to submit a **bug ticket** or report on your project. These tickets are essential as they help identify bugs and give sufficient information to the developers to fix them. But how exactly do you write a good bug report?

Any good bug report should include a proper title, priority, system information, a summary, clear instructions to reproduce the detected bug, expected behavior from the system, and proof or a screenshot of the bug in action. There are a lot of things you can add to your bug report and also different things you can omit. But overall, this should be enough.

You have to keep in mind that often bug reports are also read by the companies owning the products. These people might not have enough technical knowledge to understand complex technical jargon. So your report needs to be clear enough for everyone to understand. Not just engineers and developers. Let’s see how you can write such a report with **a bug ticket template**.

What Makes a Good Bug Ticket?

A good bug report is effective enough to solve the problem. It should clearly describe the bug and help the engineers address the issue. The report should separate the bugs from other bugs and also include steps to recreate the bug. Without these elements, a report can be very confusing for all parties involved.

The difference between a good bug report and a bad one is often how clear it is. A good bug report should have clear, concise titles and identification number that separates the bug from other bugs. It should be specific and to the point, not a descriptive essay.

For a bug report to be effective, it should allow the engineers to reproduce the bug. There should be one more URL included in the ticket, so that extra information can be gained on the bug environment.

A single ticket should only address a single bug. This simplifies the problem and makes it easier to close reports once fixed. The ticket should also be as unique as possible. This means you should make sure that you're not just reporting a duplicate bug or a bug that has already been reported. Check the database for all known bugs to see if any match.

Again, this is where the bug identification and title are really helpful. A proper bug ticket can be distinguished from other tickets just by the title alone.

The tone of the report should be informative. Don't try to be commanding or offensive. You should never assume that the developer has made a mistake that caused the bug. Be respectful and avoid any unnecessary personal opinions.

Your bug ticket should most importantly communicate "**how**" the test was performed and "**where**" the problem occurred. It should pinpoint the bug in the system as best as possible.

If you have an idea, then you should also mention effect of the bug. What would the damage be in terms of user experience, or even profits and revenues if the bug is not fixed? This creates urgency and can help clients understand the gravity of the issue.

How Do You Write an Effective Bug Report?

There are a lot of different elements that you have to keep in mind when writing an effective bug report. It's pretty easy to forget one or two of them or not have consistency in your reports. That's why to always write the best bug reports for all products, you should consider following a bug ticket template.

This template is supposed to be an overall outline of your report and you can omit a few points when you don't need them. Each product's bug report might have slightly different contents, but the overall structure should be the same. Let’s go over how you can use this template for your bug reports.

Bug Report Template

- **Title**

This should be clear and concise so that anyone who reads it can understand what the bug is. A proper title can clear a lot of confusion and helps reduce redundancy. Try to be precise and short, but don’t leave out any important information trying to make it concise.

You should also practice including a Bug ID or Bug Number and ensure that your whole team follows it. Assigning an ID to each bug can help identify them easily and keep track of whether it’s been addressed before.

- **Priority**

Not all bugs are of the same importance. Some are more minor and don’t cause a lot of damage, while some are so critical, they can make the whole system crash. So, it’s important to prioritize your bug reports, so the developers know which bugs to address first.

A bug can be a Critical, Blocker, Minor, Major, Trivial, or just a suggestion. You can assign them priorities accordingly, following company protocols. Most bug tickets prioritize bugs on a scale of P1 to P5. You can also mention what the impact of the bug is.

- **System Information**

A bug may behave differently in different systems and environments. So it's important to mention what kind of an environment the bug occurs in. This helps the developers recreate the bug so they can analyze it better.

Mention the operating system that you use, the hardware specifications, which browser you used, whether it was on a PC platform or a mobile, and so on. Include any relevant information about the testing device.

- **Summary of the Bug**

This should describe the problem that you encountered. Be precise and accurate and mention all that went wrong.

You should be careful to only include facts and leave out any opinions. Anyone who reads the report should be able to understand what went wrong. You can include technical details here, but try to avoid too much jargon.

- **Expected Behavior**

After the bug summary, you should mention what the consequences of the problem are. This is to further clarify any confusion.
Mention exactly how the system should have performed, if the bug were not present. Specify what happened in comparison to what you expected to happen.

- **Steps to Reproduce the Bug**

Often when it comes to fixing the bug, the developers cannot recreate it. It might only happen once you take certain actions. Try to include relevant test data, if the bug only occurs with certain data.

It's important to mention a step-by-step procedure of what led to the bug. You should attempt to reproduce the bug at least 3 times yourself, before reporting it. So, you know it's not a one-time thing.

- **Screenshot or Proof**

Especially for front-end problems, a screenshot can be very precise to explain what went wrong. For more complex bugs you can even include a screen recording to show which steps you take that lead to the bug.

For backend issues, a screenshot might not help as much if you aren’t accessing the code. But it might still help to narrow down the cause.

Conclusion

If you follow this overall template of a **bug ticket**, you should have a clear outline of exactly what to include in your report. When the problem is identified properly, it makes it much easier to solve and saves a lot of time for everyone. Following a template allows you to have a checklist of everything you need to include, without forgetting any key points.

Hope this helped you enough to understand the scenario. Have a nice day!