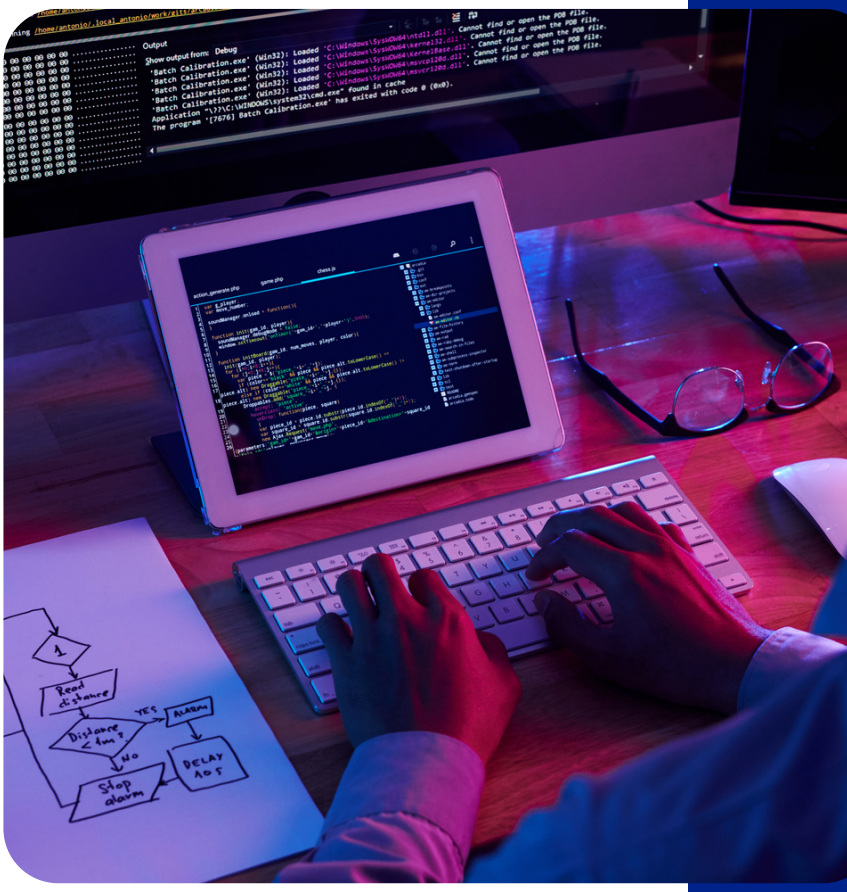
How you can avoid software bugs upfront



Most people who pay for a development team face the same absolute nightmare: bugs. And quite frankly, that doesn't speak highly of our industry. Are programmers a bunch of dreadfully ignorant nerds unable to comprehend the business problems of the real world? Most people switching developers or agencies have this very image in their heads; many of them develop symptoms of complete lack of trust and paranoia. Believe it or not, I am not exaggerating.

Two kinds of bugs

The fact is that bugs are pretty normal. It's true that they are not supposed to out of the blue take control over your current sprint, but a bug now and then is more than okay.Let us explain: there are 2 possible root causes for a bug. Being human is one of them. Developers aren't the machines they deal with, hence, human error simply can't be avoided completely. Every once in a while, a bug is sure to slip through Quality Assurance. It happens. The other reason is developers not fully understanding the goals you are trying to pursue. You can usually do a lot to minimize the risk of these misunderstandings, and it's not that hard.Note: Misunderstandings technically aren't even bugs, but you may view them as one so I'll devote a section to them here. In the end, we simply want to reopen less tasks — and misunderstandings are a huge part of the problem.

Eliminate Misunderstandings

- **Write Documentation**

Docs or it didn't happen. Imagine your project turning out to be a botched job solely because you didn't bother putting some nuances down. I'll post an article on how to write good documentation in the near future and will link it up here. For now — just make sure you write everything down, at least in bullet points.Don't get me wrong: it's absolutely appropriate to call up your dev team or agency and get verbal consultation on an issue. You should definitely do that if you have a problem to solve. Write down the outcome though and send it to your team in writing — best if you can insert it directly into their issue tracker so nothing gets lost in the mail.

- **Daily Standups**

If you don't have about 1–2 hours daily to have a standup meeting and check the delivered tasks for correctness, it's no wonder that bugs are accumulating and you end up with faulty work which hasn't been discovered before. Then, as a consequence, other components already built on top of it also lead to undesired functionality.Right now, go to your calendar and schedule 10 minutes every morning for a quick call with your development team. Ask the following three questions:What did you do yesterday?What will you be doing today?Did you face any problems?Ask these questions

regardless of whether you've just had a quick check on their progress in the issue tracker. If your team is able to give a 2-minute recap or overview, it means they truly understand your goals. If not, you are bound to notice it instantly. Then you can explain in greater detail and adjust the documentation.

Eliminate Human Error

- **Perform Quality Assurance**

Any complex application should have at least one QA engineer. Their task is to test every completed task according to its documentation. As I've mentioned above — human error may occur, and two heads are better than one. If you can afford a QA engineer who can write automated tests, that's even better. It's an investment that in the long run is able to drastically cut testing costs and make sure the core functions of your application won't break.

- **Don't Overload**

I know you're in a hurry. I understand there are deadlines. But the fact is: computer programming is a very demanding trade. It requires the programmer's full concentration, otherwise, results are bound to be mediocre. If your development team is working long hours or needs to "rush" the last features — human error is inevitable. If you rush it, you'll have more bugs, and you'll put more bug fixing tasks into the current sprint, which in its turn, will put even more pressure on your team — a deadly spiral down to a complete disaster. Thus, even if you can't meet your deadlines, try taking work off of your developers by re-prioritizing so they don't get tired or lose motivation. Trust me, you don't want that to happen.

Recap: How to Avoid Bugs

-As you've learnt, there are two root causes for undesired results: misunderstandings and human error.In order to avoid misunderstandings, write documentation and perform daily standups.In order to reduce human error, assign a quality assurance engineer to your team and don't overload your developers. Good luck with your development!