Betting in Testing

04 April 2019

“I’ve completed my testing of this feature,

and I think it’s ready to ship”

“Are you willing to bet on that?”

No, Don’t worry, I’m not going to list various ways you could test the feature better or things you might have forgotten.

Instead, I recommend you to ask yourself that question next time you believe you are finished.

Why? It might cause you to analyse your belief more critically. We arrive at a decision usually by means of a mixture of emotion, convention and reason. Considering the question of whether the feature and the app are good enough as a bet is likely to make you use a more evidence-based approach.

Why do I think I am done here? Would I bet money/reputation on it? I have a checklist stuck to one of my screens, that I read and contemplate when I get to this point. When you have considered the options, you may decide to check some more things or ship the app. Either could be the right decision.

Then the app fails…

The next day you log on and find that the feature is broken. It turns out the programmer and you had missed the bug.

Firstly, before you beat yourself up, you may have made the right decision. Even though the feature was broken - it may have been appropriate to test the way you did.

Or it may be that you need to update your checklist, skills, or automated tests etc.

How could I fail to catch the bug and yet have made the right decisions?

1. You and the rest of the team made choices on the best information you had available. You stopped and thought about what was the right thing to do. Based on that decision you shipped.
2. As you could not test all the behaviours/permutations that the app was subject to, you made an educated guess based on the data available. You used an approach that suited the system as you knew it, and you would make the same decision again.

Assuming that you made the wrong choice because it had the wrong outcome is called Resulting. It is similar to the hindsight bias and can change our future views and behaviour.

Example:

If the chance of the app failing was 0.1% (in reality we can’t usually place % on these things, but for the sake of argument…) then we might have been able to ship 1000 times and likely only seen that sort of bug once. If that’s the sort of risk profile our Product Owner is happy with, then we made the right call.

It could have been a greater risk to the business to not have that feature deployed. (Think regulatory deadlines, rival product launches etc)

In summary, every time you test, you are gambling with your time. You can use your knowledge and expertise to help make the right bets. Sometimes those bets don’t pay off.

You may miss a bug because you need to update your checklists, skills, knowledge or automated tests. Or it could be that you made a judgement call that is right 99.99% of the time, but just not that time and your test approach was correct.

Thinking this way can help focus your work on productive and valuable behaviours. For example, We won’t panic and write too many automated tests, slowing down our team’s delivery schedule. Or we may decide to examine the impact of the code change failing, rather than just looking at IF it is failing. Will it just affect a low priority system? Or will it have a catastrophic effect?