This is an example of a set of test strategy ideas. It's a raw set of rough notes from a one-hour brainstorm I did in 2018 on how I might think about testing a login process. It's intended as somewhat disgruntled reply to those examples of a script "testing" a login page by checking for the 'you are now logged in' message. There are lots of things that can go wrong with the login process, and around it.

A couple of things to mention. First, these are *rough* notes; *very* rough. I have barely cleaned them up; just enough to make them semi-readable. To be useful, they would need refinement, or to be tossed out and replaced with something clearer and cleaner. On the other hand, *it's a start*.

The second thing to mention about these notes is that they're incomplete! They are *by no means* done. You'll notice tons of empty nodes and dangling ideas on the map. There could be lots more things to question or to cover in some context or another.

Third, I'm not a expert in aspects of testing (particularly security and performance) that would require deep expertise here. I'm not doing this stuff every day. If I were, you can bet that I would immerse myself in those aspects of testing and risk investigation. I'd hang out with performance testers and security testers and programmers, and I'd collect stories about risks and problems from the Net.

Fourth, to answer the inevitable question, No, we wouldn't consider all of these things on every product, and certainly not on every build. But we might choose to pull out some of these ideas and use them to guide or influence our testing from time to time.

The key in all this is to recognize two contexts for testing. In some cases, we do want to test quickly, in a shallow but useful way, when that helps us to maintain a good pace of development and not slow down.

In other circumstances, we may need to test deeply to look for subtle, hidden, rare, intermittent, emergent bugs. Deep testing, however, is time-consuming and disruptive to programming work, so we don't even try to do it all the time, or on every build. But if there's money or value or health or safety or reputation on the line, we'll want to do deep testing to find the important problems that matter to our customers and to the business.

Michael Bolton