Types of Tests to Execute When Deploying to Staging Environment

This lesson discusses the type of tests we might need to run when deploying to the staging environment.

We'll cover the following

I often see that teams I work with get confused about the objectives of each type of test and that naturally leads to those tests being run in incorrect locations and at the wrong time. Don't get your hopes too high. If you think that I'll give you the precise definition of each type of test, you're wrong. Instead, I'll simplify things by splitting them into three groups.

Static validation

The first group of tests consists of those that don't rely on live applications. I'll call them *static validation*, and they can be unit tests, static analysis, or any other type that needs only code. Given that we do not need to install our application for those types of tests, we can run them as soon as we check out the code and before we even build our binaries.

Application-specific tests

The second group is the *application-specific tests*. For those, we do need to deploy a new release first, but we do not need the whole system. Those tests tend to rely heavily on mocks and stubs. In some cases, that is not possible or practical, and we might need to deploy a few other applications to make the tests work. While I could argue that mocks should replace all "real" application dependencies in this phase, I am also aware that not all applications are designed to support that.

> The critical thing to note is that the application-specific tests do not need the whole system. Their goal is not to validate whether the system works as a whole, but whether the features of an application behave as expected.

Since containers are immutable, we can expect an app to behave the same no matter what environment it's running in. Given that definition, those types of tests are run inside the pipeline of that application, just after the step that deploys the new release.

System-wide validation

The third group of tests is *system-wide validations*. We might want to check whether one live application integrates with other live applications. We might want to confirm that the performance of the system as a whole is within established thresholds. There can be many other things we might want to validate on the level of the whole system. What matters is that the tests in this phase are expensive. They tend to be slower than others, and they tend to need more resources. What we should not do while running system-wide validations is to repeat the checks we already did. We do not run the tests that already passed, and we try to keep those in this phase limited to what really matters (mostly integration and performance).

Why am I explaining the groups of tests we should run?

The answer lies in the *system-wide validations*. Those are the tests that do not belong to an application, but to the pipelines in charge of deploying new releases to environments. We are about to explore one such pipeline, and we might need to add some tests.

In the next lesson, we will explore the staging environment in more detail.