**1. Basics**

How do you deal with flaky tests?

* I re-execute the tests multiple times, change their execution order, and visualize how they succeed and fail over time.
* I divide the faky tests into separate test suite and try to figure out the exact root cause for the test to get failed often and fix them.
* And then add the fixed test to the previous test suite.
* In this way, I eliminate the flaky tests.

Let's suppose there is a test pipeline taking about 1 hour to finish, what would you

do to decrease the time of it?

* As I know, if I run the test cases sequentially, they consume more time. Hence, I will run the test cases in parallel.
* I will minimize the number of commands required to execute a test case and then the run time will reduce accordingly.
* For more reduction of test run time, I will use Explicit waits rather than Implicit waits so that the commands in the script execute as soon as possible without any wastage of time.

Imagine you have the possibility to ask software engineers to develop tools for

you that will increase your productivity as full-stack QA, please describe to them

your requirements?

I will ask software engineers to develop a tool which daily calendar for my meetings like stand ups, planning and retrospective meetings all together in the tool so that I don’t miss any of the meetings.

I’ll ask for a tool which changes the code written by me in Java or C# to the required programming language for the project without any errors so that it saves my energy and time.

A tool which connects Devops to any version control tool like Git and together connect to a Continuous Integration tool like Jenkins.

**2.** **Test Case Challenge**

Prepare a test plan for testing the login feature of your application, considering the

following requirements:

● The feature is implemented on iOS, Android and Web and is backend driven

● The mobile apps are native

● The login feature consists of a login form with email and password input and a login button

● After input correct credentials, you land in the main screen of your application

Your testing plan need to answer the following questions:

**1. What are you going to test?**

Common test cases for testing the application on different platforms:

* Enter wrong email, password and click on ‘Login’ button – shows error message or remaining number of attempts left.
* Enter correct email and wrong password or vice-versa and click on ‘Login’ button – behaves same as before.
* Enter correct email and password and click on ‘Login’ button – redirects to the main screen of the application.

Each client (Android/IOS/Web) gives the exact same response.

Upon launching a change from the backend, all clients can immediately reflect the change and provide in the exact same manner.

Mobile always has a versioning problem, as we know users don't always update their apps. This also must be majorly considered while testing.

**2. What would you automate and at which layer of the testing pyramid will you place it?**

Login feature for Web or Mobile(iOS/Android) can be automated and has no issues at all even if the data is retrieved from the backend.

But, given that backend can change a response instantly (and dynamically for different inputs), ensuring clients can support these ever-changing responses is challenging.

I place it in End-To-End testing layer.