***How do you deal with flaky tests?***

Below are the basic steps to be followed to deal with the flaky test

1. **Run the Test Regularly**

Once you write tests, run them regularly, preferably daily, or at least weekly.

After the same test runs multiple times, explore few things like ,average time it takes for the test to run from start to finish, the number of times the test passed, the actual functionality it is testing, how and when the test gets triggered, and other useful information.

1. **Identify the unstable Test**

The test can fail for multiple reasons, such as page load times, assertions, bad data, problems in the environment, synchronization problems, and much more. Analyse because the same test passes and fails intermittently. Checking error logs to understand the failure.

1. **Separate out flaky tests**

Analyse the Test results status from CI result. Once the failed test analysis is complete after multiple times, separate the flaky tests from the stable test suite.

1. **Fix Flaky Test one at a time**

Work on flaky tests one at a time. Check whether a test has any dependencies on other tests. Debug the root cause of the problem by commenting on some code, adding print and wait statements as needed, adding breakpoints, and constantly monitoring the logs.

1. **Improve and Re-Run**Once you fix a flaky test, run it multiple times to ensure it is passing. After consistent successful runs, add the fixed test back to the stable test suite. Rerun the stable test suite multiple times to ensure there are not any unexpected outcomes.

***Real Time Example* :** While Automating the Technical assignment , I could see the Captcha has been showing or gets pop up in the application inconsistently **.**

***Let's suppose there is a test pipeline taking about 1 hour to finish, what would you do to decrease the time of it?***

1. Use Parallel Test execution in your Framework with all the Thread Safety implementation along with the necessary thread counts.
2. Split your Test accordingly , use of best synchronization technique like Explicit wait and Use of CSS locators instead of Xpath which detects the elements fastly.
3. Use of OOPS Solid principle to implement high cohesive code which enhance the framework structure and optimize the Test execution.
4. Group the Test cases and use the best usage of TestNG framework and Best usage of Parameterization Technique using BDD or TestNG to run the same set of test cases in parallel .
5. Analyse the dependencies between the automation tool and application technology how it designed , based on that , Use the Strong Automation tool and Programming language which automates the application very fast by doing the Feasibility study with other tools.

***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***

1. The Tool should be capable of Validating the Huge amount of data between two servers

For Eg: When we suppose want to Validate the 1Lakh of data which got migrated from One Database server to another server . We need to compare the data got migrated successfully or not . Tool should support the huge data comparison which reduces time and increase Test productivity

1. Tool should adhere the Test Coverage, Reusability & Open Standards Tool should be capable of improving the business results with Increased test efficiency and Increased test effectiveness

* Testing efficiency is the average number of tests you can run for an hour of tester time. Higher testing efficiency drives down product development time and costs, improving your bottom line.
* Testing effectiveness is the rate at which your testing technology reveals bugs before your systems are released.

1. Tool should be capable of Integrating with the **Test Management tool for Eg Jira** such a way it should Execute the Test cases in Jira with Pass/Fail status which reduces the manual effort and increase the productivity in our project.
2. Tool should support Supports native plugins for the popular CI tools
3. Testers can import external testing libraries and let the tool perform the execution, avoiding the need for building a framework from scratch.
4. Tool should be capable of Creating 1lakh of users or any required data which needed a pre-requisite for loading into application. It will help in doing performance testing which reduces the time for performing any Load Test.