● **Why did you choose the framework you used?**

I choose TestNG (Test data development -TDD) because, TestNG is a testing framework inspired from JUnit and NUnit but introducing enhanced grouping and prioritized functionality that makes it more powerful and easier to use. This can be used for both Unit Testing and End to End testing.

It can be easily adopting page object and page factory automation design pattern for easy to maintain and integrate. It is very popular and widely used so, it has great community support for your problems and suggestions.

● **What were the advantages and disadvantages of your choice?**

Advantages of Using TestNG are:

1. It does not require any source code dependency.
2. Test cases can be grouped easily and prioritized easily.
3. Generating HTML report of test runs.
4. Parallel execution of test cases is possible.
5. It allows to define dependent test cases and each test case is independent to other test case.
6. Test logs can be generated.
7. Parameterization is possible.
8. Test cases can be executed on multiple browsers.
9. It can be easily integrated with Maven, Jenkins etc.
10. Assert class is used for verification in TestNG framework.

Disadvantages of using TestNG are:

1. Sometimes It becomes difficult understand for non-technical people because, it requires some kind knowledge about programming languages and framework flow.
2. Like any programming, there is a big difference between doing it and doing it well.  Writing good unit tests is an art form. This aspect of TDD is often not discussed, many managers tend to focus on metrics like code coverage; those metrics tell you nothing about the *quality* of the unit tests.
3. Early stage refactoring requires refactoring test classes as well.

**● What was the most complicated part?**

I think I don’t think any complicated part but application and Appium configuration & it related class and page element locators class should properly implement and carefully.

**● Which good coding practices did you follow when writing your tests?**

1. Commenting & Documentation
2. Consistent Indentation
3. Consistent Naming Scheme
4. Avoid deep nesting code.
5. Organize and maintain Files and Folder

**● How do you make sure your code is maintainable by other team members?**

Wiring code logic is always depending upon person to person but even in that I always make sure some following points in my minds –

1. Always try to follow most recommended frameworks and its procedure which has better community support like – page object modal and design patterns like page factory.
2. Provide code comment where it requires.
3. Always follow the folder structure and maintain it properly.
4. If possible then try to get with peer review from team members and discuss maintenance and others factors.