Tell me about your roles and responsibilities in the Project:

This is a broad question. The answer should be tailored to the specific project. Generally, it should cover your contributions to design, development, testing, and any other relevant tasks. Be specific with examples.

Have you created a framework from scratch? Explain the architecture:

If you have, describe the framework's purpose, components (e.g., test data management, reporting, logging), and how they interact. Mention the technologies used (e.g., Selenium, TestNG, Maven). If you haven't built one from scratch, discuss your experience working with existing frameworks and your understanding of their architecture.

Which design pattern have you used in the framework?

Common patterns in test automation frameworks include Page Object Model (POM), Factory, Singleton, and Observer. Explain the pattern you used and why it was appropriate.

What is 3 Amigos?

In Agile development, the 3 Amigos refers to a meeting involving the Business Analyst (representing the business), the Developer, and the Tester to discuss user stories and acceptance criteria *before* development begins. This helps ensure a shared understanding and reduces ambiguity.

Difference between Scrum and Kanban:

Scrum is a time-boxed, iterative framework with sprints, roles (Product Owner, Scrum Master, Development Team), and events (Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective). Kanban is a flow-based system with a focus on visualizing work, limiting work in progress (WIP), and continuous improvement. Kanban is more flexible and less prescriptive than Scrum.

Smoke vs. Sanity Testing, Regression vs. Retesting:

- **Smoke Testing:** A quick check to ensure the core functionality of the application is working after a build.
- **Sanity Testing:** A more focused test to verify that changes in a specific area haven't broken existing functionality.
- **Regression Testing:** Testing to ensure that new changes haven't negatively impacted existing features.
- **Retesting:** Testing to verify that defects that were previously found have been fixed

.

What are the important validations you will put on an API?

Common API validations include:

- Status codes (e.g., 200 OK, 400 Bad Request, 500 Internal Server Error)
- Response body structure and data types
- Headers (e.g., Content-Type)
- Authentication and authorization
- Error handling

Suppose you have 200 test cases and 49 tests failed; how would you collect the data of only failed test cases in TestNG? Best way?

TestNG provides reporting mechanisms. You can use listeners (like `ITestListener`) to capture test results. The `onTestFailure()` method of the listener will give you access to the failed test's information. You can then log this information to a file or generate a custom report.

What is the Maven Surefire Plugin?

The Maven Surefire Plugin is used to execute unit tests as part of the Maven build lifecycle. It generates reports of the test results.

Selenium and Web UI Testing:

Explain how you pick which test cases to automate? Prioritize based on:

- Frequency of use
- Criticality to the business
- Stability of the feature
- Repetitive tasks

Write code to find Broken Links in Selenium:

This involves iterating through all links on a page, sending HTTP requests to each link's URL, and checking the response status code. A 4xx or 5xx status code indicates a broken link.

How do you handle multiple frames?

Use `driver.switchTo().frame()` with the frame's name, ID, index, or WebElement. To switch back to the main document, use `driver.switchTo().defaultContent()`. To switch to the parent frame use `driver.switchTo().parentFrame()`.

You have 10 links; how will you print the title of each link using Selenium? Give the optimal approach:

Use `findElements()` to get a list of all link elements. Then, iterate through the list and use `getAttribute("href")` to get the link and `getText()` to get the link text (which often serves as the title).

In what cases do we need to use implicit/explicit waits?

- **Implicit Wait:** Tells WebDriver to poll the DOM for a certain amount of time when trying to find an element. Use it when elements take a relatively consistent amount of time to appear.
- **Explicit Wait:** Waits for a specific condition (e.g., element to be visible, clickable) before proceeding. Use it for elements that load dynamically or asynchronously.

What is a NullPointerException?

A `NullPointerException` is thrown when you try to access a member (method or variable) of an object that is `null` (i.e., not referencing any object in memory).

Difference between `driver.get()` vs. `driver.navigate().to()`:

Both load a URL. `driver.get()` waits for the page to fully load before proceeding. `driver.navigate().to()` doesn't necessarily wait for full page load and provides additional navigation methods (e.g., `back()`, `forward()`, `refresh()`).

How to handle Location popup in Selenium?

This depends on the popup. You might be able to use `Alert` handling (`driver.switchTo().alert()`) if it's a browser-level alert. If it's a webpage element, you'll need to locate and interact with it using standard Selenium methods.

How to handle dynamic dropdowns in Selenium?

Use `Select` class if the dropdown is a `<select>` element. Otherwise, you'll need to locate the dropdown element, click it to open the options, and then locate and click the desired option.

Difference Between pom.xml and testng.xml

-> To run test scripts in a Selenium framework, both pom.xml and testng.xml files play important roles, but for different purposes. The pom.xml file is used by Maven to manage dependencies, build configurations, and automate tasks, ensuring all necessary libraries (like Selenium and TestNG) are available. While you don't run the tests directly from pom.xml, it facilitates the build process and invokes TestNG when running tests through Maven (using commands like `mvn clean test`).

-> The testng.xml file, on the other hand, is used to define and configure test suites, classes, groups, and execution parameters for TestNG. It allows you to specify how tests should be executed and can be run directly from an IDE like Eclipse or IntelliJ. When running tests through Maven, the pom.xml will trigger the TestNG framework and execute the test scripts as defined in testng.xml.

TestNG Annotations Execution Order will be:-

-> @BeforeSuite

-> @BeforeTest

-> @BeforeClass

-> @BeforeMethod

-> @Test

-> @AfterMethod

-> @AfterClass

-> @AfterTest

-> @AfterSuite