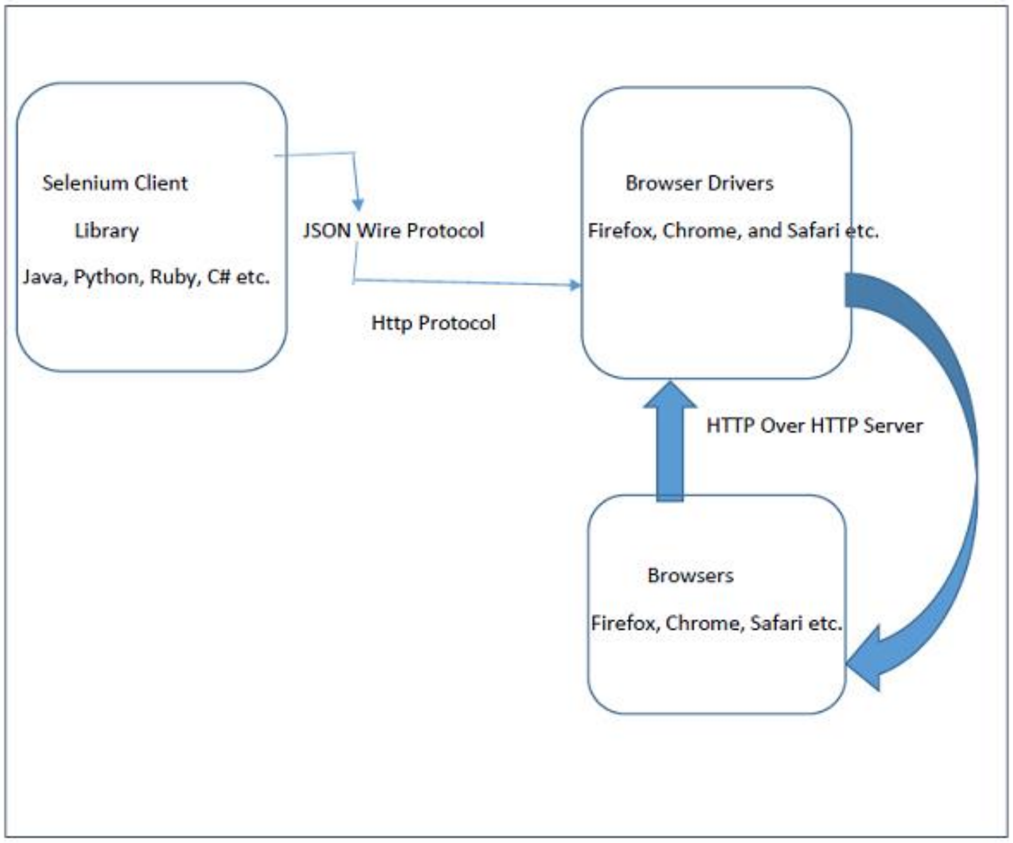
Question-6

What is the Selenium Web Driver Architecture?

Selenium Web Driver architecture in a simplified diagram is described below:



Let us now understand the Selenium Web Driver Architecture. Selenium WebDriver API enables interaction between browsers and browser drivers. This architecture consists of four layers namely the Selenium Client Library, JSON Wire Protocol, Browser Drivers and Browsers.

- Selenium Client Library consists of languages like Java, Ruby, Python, C# and so on. After the test cases are triggered, entire Selenium code will be converted to Json format.
- JSON stands for Javascript Object Notation. It takes up the task of transferring information from the server to the client. JSON Wire Protocol is primarily responsible for transfer of data between HTTP servers. Generated Json is made available to browser drivers through http Protocol.
- Each browser has a specific browser driver. Browser drivers interact with its respective browsers and execute the commands by interpreting Json which they received from the browser. As soon as the browser driver gets any instructions, they run them on the browser. Then the response is given back in the form of HTTP response.

Question-4:

Test Automation Frameworks Type:

Each test automation framework has its own architecture, advantages, and disadvantages. Some of these test automation frameworks are:

1) Linear Automation Framework

The linear Automation framework is commonly used in the testing of small applications. This framework is also called as a Record and playback framework.

Pros: There is no need to write custom code, so expertise in test automation is not necessary.

Cons: The data is hardcoded in the test script; hence, the test cases cannot be re-run with multiple sets. You need to make some changes if the data is altered.

2) Modular Driven Framework

In this Framework, the tester can create test scripts module wise by breaking down the whole application into smaller modules as per the client requirements and create test scripts individually.

Pros: Modular driven framework ensures the division of scripts that leads to easier maintenance and scalability. You can write independent test scripts.

Cons: The modular driven framework requires additional time in analyzing the test cases and identifying reusable flows.

3) Behavior Driven Development Framework

Behavior Driven Development framework is to create a platform, which allows every person, like Developers, Testers, business analyst, etc., to participate actively. It also increases collaboration between the tester and the developers on your project.

Pros: You can use non-technical, natural language to create test specifications on this behavior-driven testing.

Cons: To work with this framework, sufficient technical skills as well as prior experience in Test driven development is required.

4) Data-driven Testing Framework

Generally, Test Data is read from the external files like Excel Files, Text Files, CSV Files, ODBC Sources, DAO Objects and they are loaded into the variables inside the Test Script. The data-driven framework allows us to create test automation scripts by passing different sets of test data.

Pros: It reduces the number of scripts required. Hence, multiple scenarios can be tested in less code.

Cons: You will need a highly experienced tester who should be proficient in various programming languages to completely utilize the design of this framework.

5) The Keyword-Driven Testing Framework

The keyword-Driven Testing framework is also known as table-driven testing. This framework is suitable only for small projects or applications. The automation test scripts performed are based on the keywords specified in the excel sheet of the project.

Pros: A single keyword can be used across multiple test scripts, so the code is reusable.

Cons: The initial cost of setting up the framework is high, and it is time-consuming & complex.

6) The Hybrid test Automation Framework

Hybrid Framework is used to combine the benefits of Keyword Driven and Data-Driven frameworks.

Pros: This type leverages the advantages of all kinds of related frameworks.

Cons: Tests are fully scripted in a Hybrid Testing Framework thus increases the automation effort.

All of these test automation frameworks can be effectively used to handle the code in a systematic way, which can be reviewed by a third person easily. You can choose the framework as per your project requirements, team expertise, time as well as budget. Test automation frameworks boost productivity with standardization. By adopting the framework, you can ensure maximum test coverage in your development process.

Question-3

There are three popular test automation development models:

Record and Playback Model: This is the simplest and most straightforward test automation model. It involves recording user actions and keystrokes using a tool and playing back the recorded script to reproduce the same steps during the testing phase. The recorded test script can be executed multiple times, and its results can be analyzed for any issues or bugs. This model is useful for testing simple applications or for quickly creating test scripts. However, it can be less effective for complex applications, as it may not be able to handle complex test cases or dynamic changes in the application under test.

Keyword-Driven Model: This model is a more structured approach to test automation development. It involves breaking down test cases into smaller, reusable components called keywords, which represent specific actions or functions that the application can perform. Each keyword is associated with a specific set of test data and is executed using a test automation tool. This model allows for better modularity, reusability, and maintainability of test scripts, as keywords can be reused across multiple test cases. It also allows for better scalability and flexibility in test automation.

Data-Driven Model: This model is focused on testing an application using a range of test data inputs. It involves separating the test data from the test script and executing the test script multiple times using different sets of test data. This model is particularly useful for testing applications that require multiple input combinations and where the behavior of the application changes based on different input values. It can also help identify and isolate defects related to specific input data values. The data-driven model can be combined with other test automation models, such as the keyword-driven model, to create a more powerful test automation framework.

Question-2

Test Automation Life Cycle Diagram

Test Planning: In this phase, the testing team decides which tests to automate and which testing tools and frameworks to use. The team also creates a test plan that outlines the goals, scope, timelines, and resources required for test automation.

Test Script Development: This phase involves creating test scripts using the chosen testing tools and frameworks. The scripts are designed to replicate user actions, validate functionality, and capture the expected results.

Test Script Execution: In this phase, the test scripts are executed using the automated testing tool. The tool simulates user interactions with the application, executes the test cases, and compares the actual results with the expected results.

Test Results Analysis: Once the test scripts are executed, the results are analyzed to identify any defects or bugs. The testing team reviews the test logs and reports to isolate issues and categorize them based on their severity.

Defect Reporting: Any defects or issues identified during the testing phase are reported to the development team using a bug tracking tool. The reports include details such as the steps to reproduce the defect, screenshots, and logs.

Defect Resolution: The development team reviews the bug reports and works on resolving the issues. Once a fix is developed, it is tested by the testing team to ensure that the defect has been resolved.

Test Maintenance: Test automation is an ongoing process that requires continuous maintenance. The testing team must update the test scripts as the application evolves, and new features are added. Maintenance activities also include updating the test environment, test data, and test scripts.

Overall, the test automation life cycle is a continuous process that involves planning, development, execution, analysis, and maintenance of test scripts. The process helps ensure that software applications meet their quality objectives while reducing the time and cost involved in manual testing.

Question-7, 8

Method	**Syntax**	**Description**
By ID	driver.findElement(By.id (<element ID>))	Locates an element using the ID attribute
By name	driver.findElement(By.name (<element name>))	Locates an element using the Name attribute

By class name	driver.findElement(By.className (<element class>))	Locates an element using the Class attribute
By tag name	driver.findElement(By.tagName (<htmltagname>))	Locates an element using the HTML tag
By link text	driver.findElement(By.linkText (<linktext>))	Locates a link using link text
By partial link text	driver.findElement(By.partialLinkText (<linktext>))	Locates a link using the link's partial text
By CSS	driver.findElement(By.cssSelector (<css selector>))	Locates an element using the CSS selector
By XPath	driver.findElement(By.xpath (<xpath>))	Locates an element using XPath query