Q. Difference between Selenium and Playwright

- Selenium is highly mature, supports multiple languages, and has extensiv e browser support with a large community.
- Playwright is newer but offers modern features, easier setup, built-in parallel execution, and robust network interception capabilities.

Q. What is Playwright and what are its main features?

- Playwright is an open-source automation library for end-to-end testing and browser automation. It supports Chromium, WebKit, and Firefox with a single API.
- Its main features include cross-browser testing, parallel testing, network interception, and support for headless and headful modes.

Q. How do you install Playwright in a JavaScript project?

- We can install Playwright using npm.
- By entering the command -: npm install @playwright/test

Q. What is the difference between headless and headful modes in Playwright?

- Headless mode runs the browser without a user interface, which is useful for automated testing and CI/CD pipelines.
- Headful mode runs the browser with a user interface, allowing you to visually interact with the browser during testing.

Q. How do you handle browser contexts in Playwright?

- Browser contexts allow us to isolate different parts of our application.
- We can create a new context using **page.context()** and switch between contexts as needed.

Q. What is the purpose of network interception in Playwright?

- Network interception allows us to intercept and modify network requests and responses.
- This is useful for testing APIs, mocking responses, and debugging network issues.

Q. How do you run Playwright tests in parallel?

- We can run tests in parallel by using the **--workers option** with the playwright test command.
- Example, **npm playwright test --workers 4** will run tests in parallel using 4 workers.

Q. What are Playwright selectors and how do you use them?

Playwright selectors are used to locate elements on a web page. They can be based on CSS, XPath, or text content. Ex: page.locator('text=Example Domain')

Q. How do you handle timeouts in Playwright tests?

We can handle timeouts using the **page.waitFor()** method, which waits for a specific condition to be met before proceeding.

Ex-: await page.waitFor('selector:example')

Q. Difference between async and await

- async is a keyword used to declare an asynchronous function. It allows us to write functions that return promises.
- await is a keyword used inside an async function to pause the execution and wait for a promise to resolve or reject.

Q. How do you write a basic Playwright test in JavaScript?

```
const { test } = require('@playwright/test');

test('basic test', async ({ page }) => {
  await page.goto('https://example.com');
  await expect(page.title()).toBe('Example Domain');
```

Q. How do you take screenshots in Playwright?

We can take screenshots using the **page.screenshot()** method. For example:

```
const { test } = require('@playwright/test');

test('take screenshot', async ({ page }) => {
  await page.goto('https://example.com');
  await page.screenshot({ path: 'example.png' });
```