🎭 Day 14 – Playwright Automation (Python + Java Theory Special)

🔹 1. What is Playwright?

- Open-source test automation framework by Microsoft.
- Supports Python, Java, JavaScript, .NET.
- Designed for end-to-end testing with modern web apps.
- Works across Chromium, Firefox, WebKit (Cross-browser support).

🔹 2. Why Playwright over Selenium?

✅ Faster execution with headless browsers.
✅ Auto-waiting (no need to explicitly write waits in many cases).
✅ Network interception (mock APIs, block requests).
✅ Built-in parallel execution.
✅ Easy handling of iframes, multiple tabs & popups.
✅ Provides video & screenshot recording out of the box.

🔹 3. Playwright Architecture

- Uses browser drivers bundled inside → no external setup like Selenium Grid.
- Supports multiple languages but uses a common protocol.
- Has Playwright Test Runner (JS) but can integrate with Pytest / JUnit/TestNG.

🔹 4. Installation

Python:

```
pip install playwright
playwright install
```

Java (Maven Dependency):

```
<dependency>
  <groupId>com.microsoft.playwright</groupId>
  <artifactId>playwright</artifactId>
  <version>1.43.0</version>
</dependency>
```

🔹 5. First Test Example

Python:

```
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    browser = p.chromium.launch(headless=False)
    page = browser.new_page()
    page.goto("https://example.com")
    print(page.title())
    browser.close()
```

Java:

```
import com.microsoft.playwright.*;

public class PlaywrightTest {
    public static void main(String[] args) {
        try (Playwright playwright = Playwright.create()) {
            Browser browser = playwright.chromium().launch(new
BrowserType.LaunchOptions().setHeadless(false));
```

```
            Page page = browser.newPage();
            page.navigate("https://example.com");
            System.out.println(page.title());
            browser.close();
        }
    }
}
```

🔹 6. Key Features in Playwright

- **Auto-waiting → waits for elements to be ready.**
- **Selectors → CSS, XPath, text, role-based locators.**
- **Frames & Multiple Tabs handling.**
- **Network mocking → simulate slow responses, errors.**
- **Screenshots & Video Recording.**
- **Cross-platform testing → Linux, Windows, macOS.**

🔹 7. Playwright vs Selenium (Quick Comparison)

Feature	**Selenium**	**Playwright**
Speed	**Slower**	**Faster**
Auto-wait	❌ **No**	✅ **Yes**
Cross-browser	✅ **Yes**	✅ **Yes**
Mobile emulation	**Limited**	✅ **Advanced**
API Mocking	❌ **No**	✅ **Yes**
Parallel Execution	**With TestNG/JUnit**	✅ **Built-in**

🔹 8. Testing Use Cases

- **Automating login & user workflows.**
- **Verifying cross-browser compatibility.**
- **Capturing screenshots & videos for debugging.**
- **End-to-end testing with CI/CD pipelines.**
- **API + UI combined testing (network interception).**

🔹 9. Integrations

- **Python → Playwright + Pytest for reporting.**
- **Java → Playwright + TestNG/JUnit for structured testing.**
- **Can run in GitHub Actions, Jenkins, Azure DevOps, GitLab CI.**

🔹 10. Interview Points (Playwright Theory)

❓ **What browsers does Playwright support?**
❓ **Difference between Selenium & Playwright?**
❓ **How does Playwright handle waits?**
❓ **Can Playwright run in headless mode?**
❓ **How do you capture network requests in Playwright?**