Automation Testing using Selenium (Java)

Duration: 5 days
Prerequisites: Knowledge of Java Programming and Automation testing

Day 1 — Foundations & First Test

Objective: Get everyone from zero to one working test.

1. **Testing Basics**
   - Manual vs. automation, what to automate, stable test mindset
   - Anatomy of a web page: DOM, HTML, attributes, developer tools
2. **Environment Setup**
   - Create Maven project, add Selenium 4 + TestNG dependencies
   - Selenium Manager for browser drivers (no manual driver hassle)
3. **Your First WebDriver Test**
   - Launch browser, open URL, assert title/text, close browser
   - Understanding locator strategies: ID, name, CSS, XPath (when/why)

Labs

- L1: Project setup checklist—build & run a "Hello Selenium" test
- L2: Use DevTools to inspect elements and try 6 different locators
- L3: Write a simple "search box" test: type → submit → verify result

---

Day 2 — Interactions, Waits & Form Flows

Objective: Automate realistic user actions the safe way.

1. **Interacting with Elements**
   - Click, type, clear, dropdowns (Select), checkboxes, radio buttons
   - Handling alerts, basic frames/iframes, and new tabs/windows
2. **Making Tests Stable**
   - Why elements "aren't found"; implicit vs explicit waits; FluentWait basics
   - Common patterns for dynamic content
3. **Structuring Tests**
   - TestNG anatomy: annotations, assertions, simple test suite XML

Labs

- L4: Automate a login form (happy path + invalid creds) with explicit waits
- L5: Switch to an iframe, interact with a control, assert result
- L6: Window/tab handling: open a link in new tab → verify → return

---

Day 3 — Page Object Model (POM) & Data-Driven Basics

Objective: Improve readability and reusability.

1. **Why POM**
    - Pages, components, and responsibilities; avoiding duplication
2. **Building a Mini Framework (Beginner Level)**
    - Folder structure, base test, driver utility, config file (URL, browser)
    - Simple test data from CSV/Excel (Apache POI) or JSON
3. **Assertions & Reporting**
    - Soft vs hard asserts, capturing screenshots on failure
    - Intro to Allure or Extent Reports (basic setup)

Labs

- L7: Refactor login tests into POM (Page classes + tests)
- L8: Data-driven test: read 3 sets of creds from CSV/Excel and run the same test
- L9: Generate a report and attach screenshots for failed steps

---

Day 4 — Real-World Challenges & Good Practices

Objective: Handle flakiness and prepare tests for teamwork.

1. **Handling Tricky UIs**
    - Dynamic lists, calendars, file upload/download patterns
    - Stable locators: test IDs, CSS over fragile XPath, small helper methods
2. **Debugging & Maintenance**
    - Re-running failed tests, adding logs, screenshot strategies, waits checklists
3. **Cross-Browser & Headless (Local)**
    - Run on Chrome/Firefox/Edge; headless mode for quick smoke checks
4. **Team Basics**
    - Git essentials (clone, branch, commit, push, pull request)
    - Writing a good test case and naming conventions

Labs

- L10: File upload or calendar date selection with explicit waits
- L11: Run the suite on two browsers; produce a combined report
- L12: Fix a flaky test using improved locators and wait strategy

---

Day 5 — Capstone & Light CI Overview

Objective: Deliver a small but complete suite and know next steps.

1. Capstone Brief (choose one)
    - **E-Commerce Flow:** search → add to cart → mini checkout stub
    - **HR Portal Flow:** login → create record → edit → delete → verify
2. Build the Suite
    - 8–12 tests, POM structure, data-driven where suitable, screenshots & report
    - Smoke vs. regression tagging (TestNG groups)
3. Running in CI (Conceptual, Beginner Level)
    - Jenkins or GitHub Actions overview: trigger on push, run mvn test, archive report