QA ADVANCED 4 DAYS

Selenium with Python

Build a strong foundation in web automation using Selenium with Python by understanding core concepts, environment setup, and locator strategies. Learn how to automate web testing, handle dynamic elements and waits, perform web scraping, and integrate Selenium with Python libraries for practical automation use cases.

Why Selenium (Python)?

Faster UI automation
Automate browsers with Selenium and Python.

Stable waits
Handle dynamic UIs with reliable synchronization.

Reusable design
Apply POM and data-driven patterns.

CI integration
Run Python Selenium suites in pipelines.

What you'll be able to do

- Understanding the basics of web automation and the role of Selenium and Python in it
- Learning how to set up a Selenium environment using Python and other necessary tools
- Understanding the different types of locators in Selenium
- Learning how to handle different types of web elements
- Understanding the different types of waits in Selenium
- Learning how to use Selenium to automate web testing
- Understanding how to use Selenium for web scraping and data extraction from websites.
- Learning how to use Selenium with other Python libraries

Who this is for

- Software Developers
- Software Testers
- Automation Engineers
- QA Engineers
- Web Developers

Curriculum

01 Foundations

- Apply foundations using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

02 Environment Setup

- Apply environment setup using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

03 Locators and Interactions

- Apply locators and interactions using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

04 Waits and Dynamic Elements

- Apply waits and dynamic elements using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

05 Alerts, Frames, and Windows

- Apply alerts, frames, and windows using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

06 POM and Design Patterns

- Apply pom and design patterns using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

07 Data-Driven Tests

- Apply data-driven tests using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

08 Frameworks and Reporting

- Apply frameworks and reporting using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

09 Cross-Browser and Parallel

- Apply cross-browser and parallel using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

10 CI/CD Integration

- Apply ci/cd integration using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks

11 Debugging and Best Practices

- Apply debugging and best practices using Selenium (Python) in practical test workflows
- Build and run hands-on exercises with realistic scenarios
- Validate results, logs, and failure signals
- Capture reusable patterns for team frameworks