QE – Automation Training Program

3-Week Intensive Training | Debug-Oriented | Code-First | AI-Enhanced

Duration: 3 Weeks (15 Working Days)

Daily Schedule: 8 Hours

Instruction: 5 Hours

Hands-on Practice: 3 Hours

Total Training: 120 Hours

Program Overview

This intensive program is designed to build strong foundations in Quality Engineering and modern test automation through hands-on learning and real-world projects.

Week 1: Foundations & Web Testing

Day 1: QE Fundamentals & TypeScript for Test Automation

- Overview of Quality Engineering and its role in software delivery
- Manual vs Automation Testing
- Test Automation Pyramid
- Development environment setup (Node.js, npm, VS Code, Git)
- JavaScript fundamentals recap (variables, data types, operators, control flow)
- TypeScript fundamentals (syntax, types, type inference, type annotations)
- Functions in TypeScript (regular, arrow functions, function types)
- Interfaces, types, and classes for test automation
- Arrays, objects, and destructuring
- VS Code debugging basics (breakpoints, watch, call stack)

Practice: TypeScript coding exercises with debugging

Day 2: Asynchronous JavaScript, TypeScript & Node.js

- Understanding Synchronous vs Asynchronous programming
- Callbacks and callback hell in JavaScript
- Promises: Creating, consuming, chaining (.then, .catch, .finally)
- async/await: Modern asynchronous programming
- Error handling with try-catch in async functions
- Node.js fundamentals (modules: CommonJS vs ES6, require vs import)
- Node.js file system operations (fs module)
- npm and package.json management
- Environment variables and configuration

Practice: Build async CLI tool with Promises and async/await, debugging

Day 3: Playwright Basics

- Playwright overview and key advantages
- Architecture and setup
- Understanding async/await in Playwright context
- Writing first automated test
- Locator strategies (getByRole, getByText, getByTestId, CSS, XPath)
- Basic assertions and test structure
- Debugging with Playwright Inspector and trace viewer

Practice: Automate basic login scenarios with debugging

Day 4: Advanced Playwright – Web Automation

- Page Object Model (POM) design
- Playwright Fixtures (built-in and custom fixtures)
- Data-driven testing (reading from JSON, CSV, external files)
- Handling forms and various web elements
- Browser contexts and page management
- Debugging, screenshots, and video recording

Practice: Create a POM-based framework with fixtures and data-driven tests

Day 5: Playwright – Advanced Scenarios

- page.on() event handling (console, dialog, request, response)
- Network monitoring and request/response interception
- Handling iframes and Shadow DOM
- File upload and download automation
- Alerts and dialogs
- Authentication and cookie handling

Practice: Automate end-to-end web flows with debugging

Week 2: API Testing, Cypress & Mobile Testing

Day 6: Playwright API Testing Fundamentals

- REST API concepts and architecture
- HTTP methods and status codes
- Playwright API Testing (@playwright/test API context)
- Request and response validation
- Debugging API tests

Practice: Automate CRUD operations using Playwright API

Day 7: Advanced Playwright API Testing

- Authentication mechanisms (Basic, Bearer, OAuth)
- API chaining and data dependencies
- Schema validation
- Combining UI and API testing in Playwright
- Data-driven API tests

Practice: Build end-to-end API automation suite with Playwright

Day 8: Cypress Fundamentals

- Cypress architecture and workflow
- Cypress vs Playwright comparison
- Project setup and configuration
- Commands, assertions, and test structure
- Fixtures and data-driven testing in Cypress
- Debugging with Cypress Test Runner

Practice: Build a basic Cypress test suite

Day 9: Advanced Cypress & API Testing

- Custom commands and fixtures
- Data-driven testing in Cypress
- Network request interception with cy.intercept()
- API testing with Cypress
- Environment variables and plugins
- Introduction to API performance testing

Practice: Develop a comprehensive Cypress test suite

Day 10: Mobile Testing with WebDriverIO

- Mobile testing concepts and challenges
- Native, Hybrid, and Web applications
- Introduction to WebDriverIO
- BrowserStack App Automate setup with WebDriverIO
- Debugging mobile tests

Practice: Automate basic mobile test scenarios with WebDriverIO

Week 3: Cloud Testing, CI/CD, AI & Capstone Project

Day 11: BrowserStack Integration

- Cross-browser and cross-device testing strategy
- Integrating Playwright, Cypress, and WebDriverIO with BrowserStack
- Parallel test execution
- Debugging cloud test failures

Practice: Execute tests across multiple browsers and devices

Day 12: AWS for Test Automation

- AWS fundamentals for QA engineers
- EC2 for test execution
- S3 for test artifacts storage
- Lambda for serverless test execution

Practice: Deploy an automation framework on AWS

Day 13: CI/CD & Test Reporting

- CI/CD concepts and workflows
- GitHub Actions for automation pipelines
- Running tests in CI environments
- Test reporting using Allure and Mochawesome
- ReportPortal integration for open-source test reporting
- Real-time test analytics and dashboards

Practice: Configure a complete CI/CD pipeline with ReportPortal

Day 14: AI in Test Automation

- Overview of AI concepts in test automation
- AI-powered locators and self-healing tests (Playwright)
- AI testing capabilities in Cypress
- MCP (Model Context Protocol) for test automation
- AI Agents for intelligent test generation
- Browser assistants and AI-driven debugging

Practice: Implement AI-powered test scenarios

Day 15: Framework Design & Capstone Project

Morning Session

- Automation framework architecture and best practices
- Configuration and data-driven testing
- Build an end-to-end automation framework
- Web, API, and Mobile testing integration
- BrowserStack, CI/CD, and ReportPortal integration

Afternoon Session

- Project presentation and evaluation
- Code review and feedback
- Career guidance and Q&A
- Certification distribution

Learning Outcomes

- Master TypeScript, JavaScript, and Node.js for test automation development
- Apply asynchronous programming (Callbacks, Promises, async/await) in test automation
- Apply advanced debugging techniques across all test layers
- Perform web automation with Playwright and Cypress using fixtures and data-driven testing
- Implement page.on() event handling and network monitoring
- Build API automation using Playwright API testing
- Automate mobile applications using WebDriverIO and BrowserStack
- Integrate automation with CI/CD pipelines and ReportPortal
- Build scalable, maintainable automation frameworks
- Leverage AI concepts, MCP, and AI Agents in test automation
- Apply industry best practices and debug-oriented approaches

Assessment Approach

- Daily hands-on assignments
- Weekly coding challenges
- Final capstone project
- Code quality and best practices evaluation

Training Methodology

- 40% Conceptual learning
- 60% Hands-on practice and projects
- Debug-oriented approach with live troubleshooting
- Live coding demonstrations
- Real-world scenarios and case studies
- Strong emphasis on JavaScript, TypeScript, and Node.js coding proficiency
- Deep focus on asynchronous programming patterns for test automation
- Peer code reviews and debugging exercises
- Individual mentoring and doubt-clearing sessions