Awesome Testing

Contents

AI & Testing

UI Testing

Performance Testing

CI/CD

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

AI in Testing: From ChatGPT to Test Code Generation

A practical workshop on prompt engineering, AI-assisted test design, and daily QA workflows with ChatGPT and modern AI IDEs.

Level: Foundations

Duration: 3 days

Mode: remote, onsite

Languages: PL, EN

Tooling: ChatGPT, Cursor, Copilot, Codex, Playwright, JavaScript, TypeScript, Playwright MCP, AI Agents

Details: https://awesome-testing.com/workshops/ai-chatgpt-test-code

Key outcomes

- Design effective prompts for test tasks and bug analysis
- Generate and validate test scenarios and test data with AI
- Use AI assistants safely in coding and review workflows

Full program

Learning objective

AI is becoming a core part of modern software engineering workflows. This workshop equips testers with practical knowledge and hands-on skills to use tools such as ChatGPT and AI-powered IDEs (for example Cursor) in daily quality engineering work.

After completing the workshop, participants will be able to:

- understand how large language models (LLMs) like ChatGPT work,
- create effective prompts that improve testing productivity,
- use AI responsibly when generating and reviewing automated test code.

Scope

- LLM fundamentals and practical understanding of ChatGPT
- Key concepts: AI, NLP, LLMs, prompts
- Model mechanics: tokenization, embeddings, attention mechanism
- Transformer architecture and model training basics
- Prompt engineering methods and best practices
  - role prompting ("act as")
  - few-shot prompting
  - chain-of-thought style reasoning
  - challenge/critique prompts
  - language strategy for prompting
- Testing-focused AI applications
  - test data generation
  - test case generation
  - pair-testing with AI
  - using AI to accelerate learning and analysis
- Practical exercise: generating CI setup (GitHub Actions) with AI
- AI IDE workflows (Cursor and similar)

    - AI-assisted coding and refactoring for tests
    - semantic search and working with technical documentation
- End-to-end workshop labs
    - building automated tests with Playwright and JavaScript/TypeScript
    - evaluating what to delegate to AI vs what to keep manual
- Risk and quality controls
    - common AI failure modes and hallucinations
    - safe usage patterns in test engineering
- Next-step learning paths, including API-level AI usage

Preparation

Who should attend?

The workshop is intended for testers and QA engineers who know programming fundamentals (preferably JavaScript).

What to prepare

Participants should bring a laptop prepared according to trainer instructions provided before the workshop.

Teaching methods

The training is predominantly hands-on workshop work supported by short theory modules. Participants learn through practical exercises and guided implementation.

Training materials

- workshop presentation,
- ready-to-run code examples (organized in branches/modules),
- working notes and reference links,
- additional guidance materials used during labs.

Benefits

- Strong practical understanding of LLMs in a testing context.
- Ability to design prompts that improve output quality and usefulness.
- Better use of AI IDEs for test development and maintenance.
- Hands-on experience generating and validating automated tests with AI support.
- Clear understanding of AI risks and how to apply safe team practices.
- A practical workflow that can be reused in real QA projects.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Playwright + LLM + MCP for API and UI Testing

Hands-on training that combines Playwright, MCP server integrations, and AI tools to accelerate exploratory and automated testing.

Level: Advanced

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: ChatGPT, Cursor, Copilot, Codex, Playwright, MCP, JavaScript, TypeScript, Playwright MCP, AI agents

Details: https://awesome-testing.com/workshops/playwright-llm-mcp

Key outcomes

- Connect AI tools with Playwright through MCP and tool calling
- Automate API and UI test flows with controlled AI assistance
- Evaluate productivity gains and reliability risks in AI-driven testing

Full program

Learning objective

This training shows how to connect a Model Context Protocol (MCP) server for Playwright with AI tooling (e.g., GitHub Copilot / Cursor) and modern LLM approaches such as tool calling (function calling) and Retrieval-Augmented Generation (RAG). The goal is to accelerate exploratory testing and the development of automated tests for web applications.

By the end of the course, participants will be able to:

- Explain what tool calling (function calling) and RAG are, and how they apply to software testing.
- Describe the role of MCP and its architecture in the Playwright ecosystem.
- Run LLM-assisted exploratory testing sessions.
- Generate and maintain Playwright tests (UI and API) with AI assistance.
- Assess when AI support improves productivity and when it increases risk.

Agenda

1) Modern LLM techniques

- LLM fundamentals (how tools like ChatGPT operate)
- Tool calling / tools: connecting the model to external capabilities
- Retrieval-Augmented Generation (RAG): improving answer quality using internal documentation
- AI agents: concepts, theory, and practical applications

2) MCP and Playwright

- MCP server architecture
- Installation, configuration, and integration with Playwright (JS/TS)
- API overview: accessibility snapshots, navigation, and element actions

3) MCP in hands-on testing

- Prompt-driven exploratory testing
- Generating test cases from user stories

4) Test automation with AI

- Using Copilot / Cursor for test automation
- Connecting Copilot / Cursor with MCP: a "prompt → code → run → fix" workflow
- Practical workshops:
  - Automating REST API checks with Playwright
  - Automating UI tests with MCP + Playwright
- Productivity review: what to write manually vs what to generate

5) Risks, ethics, and maintainability

- Model hallucinations vs test stability and quality
- Versioning prompts and ensuring reproducibility
- Cost analysis and return on investment
- MCP and Playwright roadmap: what to watch next

Prerequisites

Who is this for?

The course is intended for testers who have basic programming skills in JavaScript/TypeScript and understand Playwright fundamentals. Familiarity with Git is recommended.

What to prepare

- A laptop with Node.js (LTS), Git, and VS Code
- An AI coding tool configured (e.g., GitHub Copilot or Cursor)

A detailed setup checklist is provided after registration.

Teaching methods

Workshops (~65%) plus lecture-style explanation (~35%). Each practical block ends with a short retrospective and knowledge sharing. Participants leave with working code and a prompt history.

Training materials

- Slide deck (PDF)
- Git repository with branches for each module
- Prompt notebook (Markdown)
- A curated board with links and theory

Benefits

- Practical skill: controlling a browser via MCP, increasingly in demand.
- Faster automation development with AI support.
- Stronger exploratory testing with AI as an assistant.
- Better risk management: checklists and guidance on when not to trust LLM outputs.
- Knowledge sharing with peers: prompts, templates, and repository patterns.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Playwright Fundamentals

From zero to productive Playwright test development with TypeScript, debugging techniques, and CI-ready foundations.

Level: Foundations

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: Playwright, JavaScript, TypeScript

Details: https://awesome-testing.com/workshops/playwright-fundamentals

Key outcomes

- Build a clean Playwright project structure
- Write stable UI and basic API tests
- Use traces and artifacts to debug failures

Full program

Overview

A practical entry-level training that takes participants from first Playwright tests to a clean, maintainable automation baseline ready for team work and CI usage.

What you will learn

- Build a Playwright project from scratch with clear structure
- Implement stable UI tests and basic API checks
- Use traces, screenshots, and logs for debugging
- Prepare a suite for basic CI execution

Workshop agenda

Day 1: Core Playwright workflow

- Installation and project bootstrap
- Test anatomy, selectors, and assertions
- Browser contexts, waits, and stability basics
- First end-to-end test scenarios

Day 2: Maintainability and team readiness

- Reusable helpers, fixtures, and organization patterns
- Data handling and environment configuration
- Debugging failures with artifacts
- CI-ready setup and practical team conventions

Prerequisites

Basic testing knowledge is recommended. Prior Playwright experience is not required.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Playwright Advanced: Stability, Scale, CI

Optimize flaky suites, improve architecture with fixtures, and scale execution with parallelism and sharding.

Level: Intermediate

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: Playwright, JavaScript, TypeScript, GitHub Actions

Details: https://awesome-testing.com/workshops/playwright-advanced

Key outcomes

- Reduce flakiness with trace-driven diagnosis
- Refactor test architecture around fixtures
- Scale runtime safely in CI pipelines

Full program

Overview

Advanced Playwright workshop for teams that already have automated suites and want to improve reliability, architecture, and execution speed. The training focuses on reducing flakiness and scaling test delivery in CI.

What you will learn

- Diagnose and reduce flaky behavior using traces and artifacts
- Refactor test architecture with fixtures and reusable abstractions
- Scale execution with parallelism, sharding, and environment strategy
- Improve CI feedback quality and maintainability

Workshop agenda

Day 1: Stability and architecture

- Root causes of instability in mature test suites
- Trace-driven debugging and failure classification
- Fixture patterns for cleaner and more maintainable tests
- Test data and state management at scale

Day 2: Scaling in CI

- Runtime optimization and parallel execution strategy
- Sharding, retries, and selective test execution patterns
- CI diagnostics and fast feedback loops
- Governance for long-term suite health

Prerequisites

Hands-on Playwright experience and understanding of CI basics are required.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Cypress + JavaScript Foundations

Live, practical Cypress training focused on real project setup, UI/API testing, and productive JavaScript test workflows.

Level: Foundations

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: Cypress, JavaScript, TypeScript

Details: https://awesome-testing.com/workshops/cypress-javascript-foundations

Key outcomes

- Set up a full Cypress testing environment
- Write maintainable end-to-end and API tests
- Work with modern front-end test tooling in day-to-day QA

Full program

Overview

Hands-on training for building practical Cypress skills from project setup to maintainable test implementation. Participants work with realistic UI and API scenarios and learn how to create an effective day-to-day automation workflow.

What you will learn

- Set up a Cypress project with a clean JavaScript test structure
- Write maintainable UI and API tests
- Use selectors, fixtures, and test data effectively
- Run tests locally and in basic CI workflows

Workshop agenda

Module 1: Foundations and setup

- Cypress architecture and core execution model
- Local environment setup and project bootstrap
- Test structure, naming, and maintainability basics

Module 2: UI and API practice

- Reliable UI interactions and assertions
- API testing basics and response validation
- Working with fixtures, mocks, and reusable helpers

Module 3: Workflow and quality

- Debugging and failure analysis
- Simple CI integration and reporting
- Common anti-patterns and stabilization guidelines

Prerequisites

Basic JavaScript understanding is recommended. No prior Cypress experience is required.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Cypress Advanced: Isolation, TypeScript, CI

Advanced Cypress workshop covering isolated tests, TypeScript, CI pipelines, flake management, and optional visual testing.

Level: Advanced

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: Cypress, JavaScript, TypeScript, GitHub Actions, Percy

Details: https://awesome-testing.com/workshops/cypress-advanced

Key outcomes

- Implement robust isolated UI testing practices
- Integrate Cypress execution into CI/CD workflows
- Use advanced patterns for scaling and reliability

Full program

Overview

Advanced Cypress training for teams that already run UI automation and want stronger reliability, maintainability, and CI scalability. The workshop targets flaky behavior, architecture quality, and effective test execution pipelines.

What you will learn

- Build isolated, deterministic Cypress tests
- Apply TypeScript patterns for stronger test code quality
- Integrate Cypress suites into scalable CI workflows
- Reduce flakes using diagnostics and stabilization techniques

Workshop agenda

Day 1: Test architecture and reliability

- Isolation principles and deterministic state setup
- Component and end-to-end strategy alignment
- TypeScript structure for reusable test utilities
- Patterns for selectors, fixtures, and test data control

Day 2: CI scaling and operational quality

- Pipeline optimization for runtime and stability
- Flake diagnostics, retries, and quarantine policies
- Parallel execution and environment orchestration
- Optional visual regression integration and governance

Prerequisites

Solid hands-on Cypress experience and basic familiarity with CI workflows.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Selenium WebDriver Intermediate: Framework Design

A hands-on workshop on building a maintainable Selenium test framework, covering architecture, reusable patterns, reporting, and Selenium Grid execution.

Level: Intermediate

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: Selenium, Java, Selenium Grid

Details: https://awesome-testing.com/workshops/selenium-framework-intermediate

Key outcomes

- Design and build maintainable Selenium framework modules
- Apply Page Object and test-data patterns to create reusable tests
- Run, debug, and troubleshoot tests locally and in remote environments (including Selenium Grid)

Full program

Learning objective

Selenium WebDriver is a free tool used to automate web applications.

This workshop expands your Selenium WebDriver skills and focuses on building a practical, maintainable test framework. You will learn how to optimise your day-to-day work, apply advanced testing techniques, and follow proven best practices for automated test development.

The course is primarily delivered using Java. For closed groups, it can also be delivered using Python, C#, or JavaScript. The training is workshop-based: for each topic, participants implement real framework components.

Scope

- Selenium WebDriver: extending core concepts and techniques
- Framework design and architecture
- Project structure
- Creating core framework components (Framework Core)
- Test configuration and execution
  - Properties
  - Parameters
- Test parametrisation with test data
  - DataFactory
  - DataReader
- Utilities
  - Helpers and common functions
- Automatic test logging
- Automated test reporting
- Automatic screenshots

- Integrations with external libraries and tools
- Design patterns
  - Page Object Pattern and PageFactory (advanced usage)
- Remote execution
  - RemoteWebDriver and Selenium Grid
  - Server and remote node setup
  - Running tests on remote machines
- Troubleshooting common framework issues
- Testing an application using the created framework

How to prepare

Who should attend?

This training is intended for participants who:

- have basic programming skills in Java, JavaScript, or Python, and
- understand Selenium WebDriver fundamentals.

It is suitable for manual testers transitioning into automation, junior and intermediate automation testers, and software developers.

Preparation

Please bring your own laptop prepared according to the trainer's instructions.

Teaching methods

The training is predominantly hands-on, supported by short lecture-style explanations. Participants learn the tool by completing practical tasks.

Training materials

You will receive a complete set of training materials (presentation and supporting resources) and full access to electronic materials on the edu.ittraining.pl platform.

Benefits

- You will deepen your knowledge of automated testing with Selenium WebDriver.
- You will learn advanced techniques and capabilities of the tool.
- You will be able to build your own web testing framework or extend an existing one.
- You will strengthen your professional profile and support career growth in test automation.

Further development path

A recommended skills-development path can be proposed via related Selenium courses. You can also attend this course independently.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Performance Testing with Gatling + Java

Workshop focused on methodology, load modeling, Gatling implementation, and decision-ready performance reporting.

Level: Intermediate

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: Gatling, Java

Details: https://awesome-testing.com/workshops/performance-gatling-java

Key outcomes

- Plan performance tests from business goals
- Implement robust scenarios in Gatling + Java
- Present SLA/SLO-oriented reports for technical and business stakeholders

Full program

Overview

A practical workshop on planning, implementing, and analyzing performance tests with Gatling and Java. The program combines load testing methodology with hands-on scripting and report interpretation oriented to SLA/SLO decisions.

What you will learn

- Translate business goals into measurable performance test objectives
- Build reusable load scenarios in Gatling + Java
- Work with realistic traffic models and dynamic data
- Communicate results clearly to technical and business stakeholders

Workshop agenda

Day 1: Methodology and scenario design

- Performance testing goals, scope, and risk framing
- Workload modeling and user behavior patterns
- Gatling project structure and script fundamentals
- Assertions and thresholds aligned with service objectives

Day 2: Realism and reporting

- Correlation, dynamic tokens, and data feeders
- Scenario hardening for repeatable runs
- Report analysis: latency, throughput, error rates, saturation
- Converting technical results into release recommendations

Prerequisites

Basic knowledge of APIs/HTTP and Java fundamentals is recommended.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Performance Testing with k6 + JavaScript

Design and execute practical load tests in k6 with realistic data, correlation, and business-facing reporting.

Level: Intermediate

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: k6, JavaScript, TypeScript

Details: https://awesome-testing.com/workshops/performance-k6-javascript

Key outcomes

- Build reusable k6 performance test scenarios
- Use realistic feeders and token correlation
- Interpret metrics and communicate outcomes to stakeholders

Full program

Overview

This workshop teaches practical performance testing in k6 with JavaScript, from scenario design to execution and business-focused reporting. Participants build reusable scripts and learn how to interpret key metrics for release readiness.

What you will learn

- Design effective load and stress scenarios in k6
- Create reusable JavaScript-based performance scripts
- Handle realistic data, token correlation, and runtime configuration
- Interpret metrics and turn results into actionable recommendations

Workshop agenda

Day 1: Foundations and implementation

- Core performance concepts and objective definition
- k6 setup, project structure, and script lifecycle
- Modeling user journeys and traffic distribution
- Assertions, thresholds, and baseline run strategy

Day 2: Advanced execution and analysis

- Correlation and dynamic data handling
- Environment parameterization and repeatable execution
- Deep analysis of latency, errors, and throughput patterns
- Stakeholder-oriented reporting and optimization priorities

Prerequisites

Basic JavaScript knowledge and familiarity with web/API fundamentals are recommended.

Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

CI/CD and Test Observability

Practical CI/CD workshop for test execution pipelines, Dockerized runs, quality gates, and failure diagnosis workflows.

Level: Intermediate

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: GitHub Actions, Docker, Jenkins

Details: https://awesome-testing.com/workshops/cicd-observability

Key outcomes

- Build test pipelines in GitHub Actions and Jenkins
- Containerize test runs for reproducible execution
- Implement observability and actionable quality gates

Full program

Overview

A practical workshop on building reliable test delivery pipelines with CI/CD, containers, and observability. The program emphasizes repeatable execution, clear diagnostics, and quality gates that support release decisions.

What you will learn

- Build and structure test pipelines in GitHub Actions and Jenkins
- Containerize test execution for reproducibility across environments
- Implement actionable quality gates and failure triage routines
- Improve visibility into pipeline and test-suite health

Workshop agenda

Day 1: CI/CD pipeline architecture

- Pipeline design for test stages and environments
- Dockerized test execution patterns
- Trigger strategy for PR, nightly, and release workflows
- Artifact management, retries, and baseline stability practices

Day 2: Observability and quality controls

- Capturing logs, traces, screenshots, and reports for diagnosis
- Test result analytics and trend interpretation
- Quality gates based on risk, coverage, and release confidence
- Incident-style triage flow for failed builds and flaky suites

Prerequisites

Participants should understand basic automated testing and version control workflows.