Document Generated: 10/26/2025

Learning Style: Virtual Classroom

Technology:

Difficulty: Beginner

Course Duration: 1 Day

Implementing AI in Software Testing (TTAI2140)

About This Course:

AI is beginning to reshape how testing is planned, written, and maintained, and this course helps you build the skills to apply it in ways that actually make your work easier. Whether you are creating test cases, choosing what to run, or reviewing failures, you will learn how to use AI tools to speed things up, reduce repetitive

effort, and improve coverage. You will work hands-on with user-friendly AI tools generate test data, build tests from user stories, and support smarter decisions about what to test and when.

You will practice spotting flaky or redundant tests, creating self-healing flows, and using AI to explain what went wrong in a failing run. You will also explore how AI can predict risk based on commit history or past bugs, helping you focus on the areas that matter most. The course will show you how to plug AI into common testing workflows, including CI/CD tools like GitHub Actions, and how to write prompts that give you useful, accurate results. You will get examples, use cases, and guided labs that you can use right away in your own projects.

This expert-led, one-day course is designed for software testers who are new to AI but already familiar with core testing practices. It is about 50 percent hands-on, with labs built around common tasks that testers perform every day. Whether you are working in a manual, automated, or hybrid role, this course will help you start using AI in ways that are practical, helpful, and easy to build on.

Course Objectives:

The goal of this course is to help you build practical, hands-on skills for using AI in modern software testing. You will walk away with real experience using AI tools that can support faster, smarter, and more reliable testing in your day-to-day work.

By the end of this course, you will be able to:

- Use cutting-edge AI tools to generate realistic and varied test data tailored to your testing goals
- Select and prioritize test cases based on code changes, historical results, and risk indicators
- Generate UI, API, and functional test cases using plain-language prompts and AI coding tools
- Identify flaky or redundant tests and improve stability with self-healing and AI analytics
- Use AI to summarize test failures, detect patterns, and support faster triage
- Integrate AI-generated tests and insights into existing CI/CD workflows for real-time value

Audience:

This course is designed for software testers who are new to using AI and want to learn how to apply it confidently in real testing environments. It is ideal for QA professionals, test engineers, SDETs, or manual testers who want to add AI-assisted workflows to their skillset. The course is beginner-friendly when it comes to

AI, but assumes a basic understanding of software testing concepts.

Prerequisites:

This course is designed for experienced software testers who are new to using AI and want to learn how to apply it confidently in real testing environments. It is ideal for QA professionals, test engineers, SDETs, or manual testers who want to add AI-assisted workflows to their skillset. The course is beginner-friendly when it comes to AI, but assumes a basic understanding of software testing concepts.

Recommended skills before attending:

- Comfortable reading and reviewing test cases
- Familiarity with functional or UI testing practices
- Basic experience with common testing tools or frameworks (manual or automated)

Course Outline:

1. Introduction to AI in Software Testing

Get a clear understanding of how AI is changing software testing and how it complements—not replaces—your current workflow.

- What AI means in software testing
- Traditional vs AI-augmented workflows
- Key benefits: speed, coverage, accuracy
- AI in unit, integration, UI, and end-to-end testing
- Tool types for different user levels

2. Generating Test Data with AI

Discover how AI can help you create realistic, diverse, and structured test data faster and more effectively.

- Why high-quality test data matters
- Structured data with Mockaroo and Faker
- Using ChatGPT for edge-case inputs
- Choosing the right tool for your needs
- Keeping data anonymous yet realistic

3. Selecting Test Cases with AI

Learn how AI can help prioritize, streamline, and optimize your test suites based on real project data.

- Prioritize tests based on change history

- Remove redundant or low-value tests
- Select minimal test sets with high impact
- Generate tests from requirements
- Visualize impact with dashboards

4. AI-Enhanced Test Generation

See how AI tools can transform user stories and function headers into working test cases for UI, API, and more.

- Use AI for UI, API, and functional test cases
- Convert user stories into test scripts
- Improve outputs with prompt tuning
- Explore Copilot, Testim, and Codeium
- Integrate generated tests in IDEs

5. Smart Test Execution and Maintenance

Explore how AI improves test stability and maintenance with flaky test detection, self-healing flows, and visual testing.

- Detect and debug flaky tests
- Find test bottlenecks with analytics
- Use self-healing selectors
- Perform visual regression testing
- Track evolving test failures

6. Defect Prediction using AI

Use AI to find out where bugs are most likely to appear before they do, so you can test smarter, not harder.

- Forecast risk using test and code history
- Correlate bugs with churn and complexity
- Analyze commits using NLP
- Visualize risk zones with heatmaps
- Use insights in planning

7. Integrating AI into the Testing Workflow

Learn how to plug AI into your everyday test planning, CI/CD tools, and team reporting workflows.

- Identify AI entry points in the workflow
- Generate summaries and insights with ChatGPT
- Add AI to CI/CD (e.g., GitHub Actions)
- Use LLMs to triage test failures
- Standardize prompt best practices