Learning Objectives of CP-AAT

"Knowledge with experience is power; certification is just a by-product"

What is CP-AAT?

CP-AAT stands for "**Certified Professional - Automation Agile Testing**" certification. The course is applicable for all roles and not just "testers". Knowledge, experience & certification is consciously designed to focus on "agile testing" and automation of testing in agile and DevOps and **not** just on "agile testers" or "testers".

How is it useful?

CP-AAT prepares you for utilizing automation effectively in an agile project. It helps you realize the tester's role in planning and strategizing for Testing particularly BDD Driven Test Automation. CP-AAT is designed specifically for working professionals who are already working in agile and DevOps practices and want to leverage TDD and BDD practices in optimizing the test specification and test automation. The course covers **BDD in details using Cucumber and various facets of Gherkin (DSL)**. All the concepts are driven using real user stories and case studies which involve participants to write Gherkin Feature files and then code step files both for Unit and End to End level. The program introduces and goes in depth learning for practices / tools such as **TDD, BDD, Cucumber, Junit and Selenium.**

Am I Eligible?

Anyone having 1-year experience in agile or testing is eligible for CP-AAT. Having prior knowledge of any programing language will come in very handy. If you know Selenium and you have cleared CP-SAT, this program would help you expand your test automation knowledge into agile automation.

Duration?

Learning Program designed around CP-AAT can be completed in 2-3 days (total of 14-21 hours days) depending on the participants background and readiness.

The CP-AAT exam is an open exam and can be taken by anyone who is working in Agile and is familiar in BDD using Cucumber.

CP-AAT Exam

CP-AAT exam is three hours exam. One-hour theory exam with multiple choice questions and a two-hour practical exam covering the concepts of TDD, BDD using Cucumber, Junit and Selenium. Getting more than 60% in both the exams is necessary to clear the CP-AAT exam ang get all the necessary requirements for getting the certificate fulfilled.

CP-AAT Learning Objective Version

This is version 2.0 of CP-AAT learning objective. The program has gone through a major revision. Version 2.0 learning objective has taken out sections on Fitness. The CP-AAT program now covers in depth BDD using Cucumber including all the Gherkin keywords. The CP-AAT program can now be taken using Specflow if the technology of choice is C# and .Net. Relevant sections of Cucumber will change to Specflow.

Learning Objectives of CP-AAT:

1. Agile Recap

- 1.1. Warm up Agile Quiz
- 1.2. Agile Manifesto – quick look
- 1.3. Agile in Practice (Video), Models a quick look
- 1.4. Understanding the challenges associated with defects creeping in due to requirement not understood well

2. User Stories, Testing, Test Pyramid and Need for Test Automation in Agile, Automated Tests in Agile – Test First Approach

- 2.1. Understanding User stories
- 2.2. When are user stories written?
- 2.3. Story Hierarchy (Epics, Features and Stories)
- 2.4. Origination of User stories
- 2.5. Testing Agile in comparison to legacy testing in phase
- 2.6. Tests in Agile
- 2.7. Acceptance Criteria and Acceptance Tests
- 2.8. Practical exercise on writing acceptance tests for a given story – understanding the value of examples
- 2.9. Testing Strategy and Role of Testing in Agile
- 2.10. Increase in load of testing due to iterations and the importance of Automation
- 2.11. Testing Pyramid, importance of Unit and API testing. What is Unit testing, Integration Testing and End to End Testing
- 2.12. Automated Tests in Agile - Test First Approach (TDD / ATDD / BDD)

3. TDD and Unit Testing

3.1. What is Unit Testing

- Sample Unit test case
- Unit Testing Frameworks

3.2. TDD

- How does TDD Work
- Red, Green, Blue Cycle
- TDD practical

3.3. TDD Case Study

- For a given problem statement implement TDD

3.4. Understanding importance of Static Code Analysis and Code Coverage

3.5. TDD, ATDD, BDD – understanding the differences and similarities among the three

4. BDD

- What is BDD
- BDD practices
  - Discovery
  - Formulation
  - Automation
- Human Language Support
- BDD Tools
- BDD Myths

5. Introduction to Cucumber

- Behavior in Feature document
- Step definition
- Human Language Support
- Cucumber for Java – Cucumber JVM
- Testing Possibilities
- Specification by Examples

5.1. Installation and configuration for Cucumber

- Installing Eclipse
- Creating Maven project
- Maven dependencies for Cucumber and other tools used in the program

- Installing Cucumber and Natural Plugins

5.2. Implementing BDD using Cucumber

- Preparing Features File having test scenarios
- Writing a step definition
- Understanding Gherkin Keywords – Feature, Scenario, Given, When, Then, And, But and usage in features files and the associated glue code in step file
- Passing parameters in Step Functions
- JUnit Test generation in Cucumber
- Using Assertions to report failure
- Running simple feature/Step scenario
- Building a simple test case
- **BDD Case Study** - For the Given User Story, please create necessary Cucumber feature files and step file
- **Corelate how BDD helped in improving the already implemented TDD source**
- **Learn the importance of specification by examples**

6. Cucumber and Gherkin Deep Dive

- Scenario Outline and Examples
- Case study on Scenario Outline and Examples - extending the already created feature file and how data driven approach works
- Doc String Delimiters in Gherkin
- Step Tables practical
    - Understanding List of Maps
    - How Datatables work
- Writing Runner Class, Cucumber Options for
    - Pretty, Tags and Glue
    - Cucumber HTML Reports
    - Features
    - Glue
- Running features from command line using Maven and Runner class
- Tags - organize your features and scenarios
    - Tag Inheritance
    - Logical expressions in Tags
    - Not
    - And
    - Or
    - Using Tags in CucumberOptions (Runner Class)

7. Cucumber and Selenium for End to End Tests, CI using Jenkins

- Configuring Eclipse for Selenium
  - Selenium Setup and integration in the existing project
  - Getting the Selenium utility classes for driver invocation
- practical for UI / End to End Tests using Cucumber and Selenium
  - Simple User story from the End to End perspective
  - More complex end to end user stories
- Understanding how all of this can be tied together using a CI Tool like Jenkins.

8. More BDD

- More User stories and practical. Practice all BDD fundamentals throughout the course using more examples from the case study.