Software Tester to DevOps Automated Tester

SKILLSOFT ASPIRE JOURNEY

skillsoft

Głównym wyzwaniem przed którym stają dziś organizacje na całym świecie jest konieczność ciągłego podnoszenia umiejętności i poziomu wiedzy w ślad za gwałtownym rozwojem nowych technologii i zmian na globalnym rynku.

Stały rozwój i podnoszenie kwalifikacji w IT od dawna jest już rzeczą oczywistą, a możliwość zapewnienia wsparcia specjalistom chcącym stale się rozwijać jest jedną z głównych kart przetargowych w walce o pracownika.

Na rynku liczą się dziś ludzie, którzy posiadają konkretne kompetencje i zestaw umiejętności pozwalający im wykonywać zadania efektywnie, a nie Ci z najdłuższym stażem pracy.

Dziś, bardziej niż kiedykolwiek w cenie jest umiejętność budowania ścieżki kariery dla profesjonalistów IT, którzy wciąż chcą się liczyć na rynku pracy.

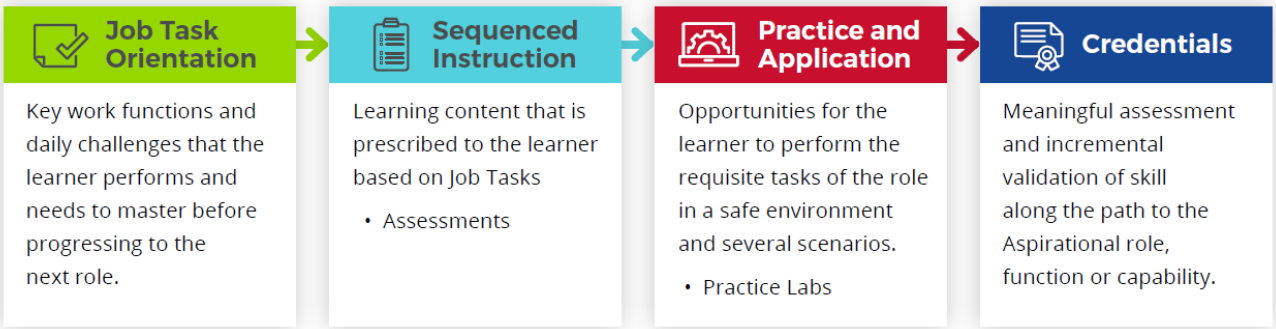
Skillsoft Aspire Journey stanowi odpowiedź na pytanie, jakie szkolenia muszę ukończyć, aby być przygotowanym do mojej wymarzonej pracy. Spośród kilkuset kanałów tematycznych dostępnych na naszej platformie szkoleniowej nasi specjaliści wybrali te, które naszym zdaniem najlepiej wyposażą uczących się w narzędzia potrzebne do realizacji zadań w nowej roli.

Skillsoft Aspire Journey to zestawy szkoleń i ćwiczeń w języku angielskim, które metodycznie, krok po kroku pozwalają specjalistom przejść od poziomu podstawowego do zaawansowanego.

Każda ścieżka zawiera szkolenia, laboratoria wirtualne, video i książki, które pomogą uczącym się osiągnąć pożądane kompetencje poświadczone certyfikatem.

Aspire Journey Model

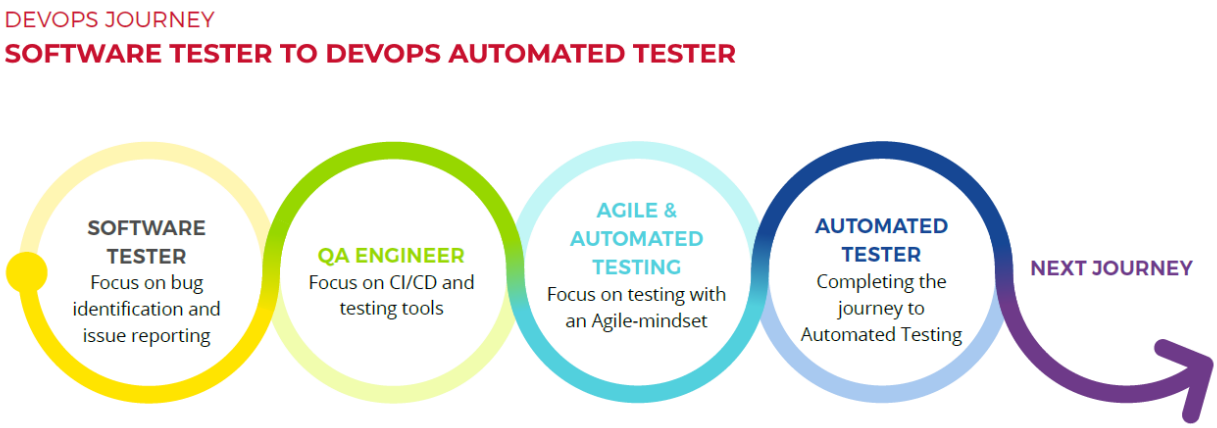
Cała ścieżka opiera się na 4-elementowym cyklu powtarzanym na kolejnych etapach nauki.



1. Określenie kluczowych funkcji i wyzwań, z którymi musi poradzić sobie uczący się w chwili obecnej, jak i tymi, z którymi przyjdzie mu się zmierzyć w nowej pracy.
2. Przejście zaprojektowanych ścieżek w proponowanej kolejności, wykonanie ćwiczeń i zaliczenie testów.
3. Przećwiczenie nowych umiejętności w kontrolowanym środowisku w oparciu o gotowe scenariusze działań. Laboratoria wirtualne Skillsoft
4. Certyfikat – zaliczenie testu końcowego na poziomie co najmniej 70% i uzyskanie certyfikatu potwierdzającego ukończenie danego etapu nauki.

Aspire Journey – Enterprise Developer to DevOps Engineer

Analizując trendy opisujące zachowanie użytkowników na naszych platformach szkoleniowych i współpracując ściśle z naszymi klientami na całym świecie Skillsoft wyselekcjonował najlepsze materiały szkoleniowe i ułożył je w ustrukturalizowaną ścieżkę rozwoju. Ścieżka zawiera ponad 72 godzin szkoleniowych.



6 courses
5h 25m 47s

- software testing for DevOps,
- navigating testing tools,
- test automation,
- continuous integration

7 courses
8h 31m 50s

- API management, unit testing,
- testing clean code,
- testing with Docker,
- automated testing

6 courses
6h 51m 50s

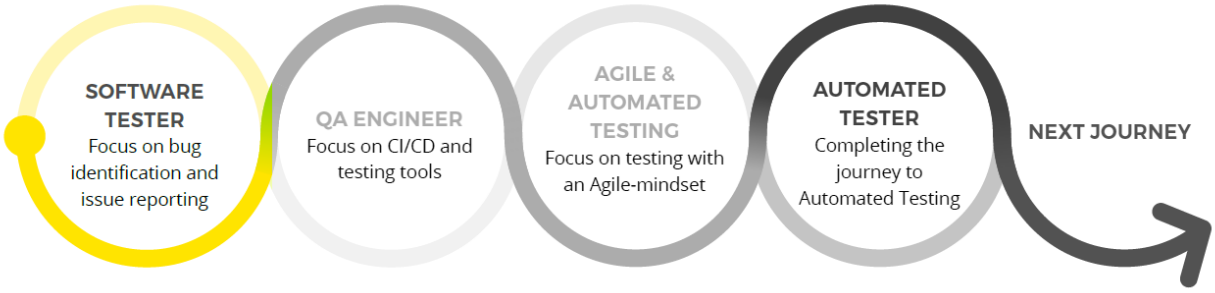
- manual vs automated testing, Agile
- software testing cycle, test driven development, end to end testing, CI/CD for autmated testing, and modyfying test frameworks

8 courses
7h 54m 50s

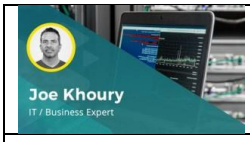
- testing with SoapUI, Cucumber, HP
- UFT, TestComplete, Python, and Selenium.

DEVOPS JOURNEY

SOFTWARE TESTER TO DEVOPS AUTOMATED TESTER



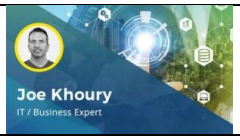
Track 1: Software Tester (duration: 5h 25m 47s)

 Software Testing for DevOps: Testing & DevOps Overview	 Software Testing for DevOps: DevOps Testing Pipeline
Objectives ▪ recognize key concepts underpinning the DevOps software development movement ▪ recognize the key principles behind the DevOps movement represented by the acronym CALMS ▪ describe the SDLC within a DevOps software development movement ▪ describe how Agile kicked off a changing approach to software testing culminating in DevOps ▪ recognize the key concepts of a DevOps pipeline ▪ describe how DevOps requires a different approach to software testing ▪ recognize key general software testing techniques that can comprise testing strategies ▪ recognize key software testing strategies and techniques in development within a DevOps culture ▪ recognize key software testing strategies and techniques in production within a DevOps culture ▪ recognize DevOps testing best practices ▪ use Jira for tracking customer feedback on software use ▪ describe DevOps, the CALMS acronym, and the SDLC under DevOps, and list common test strategies and DevOps testing best practice concepts	Objectives ▪ describe the concept of continuous integration ▪ recognize the benefits of continuous integration ▪ describe the concept of continuous delivery ▪ describe the concept of continuous deployment ▪ describe the concepts of continuous testing and test orchestration in a DevOps testing pipeline ▪ recognize the benefits related to continuous testing in a DevOps testing pipeline ▪ recognize the challenges related to continuous testing in a DevOps testing pipeline ▪ recognize some common tools used in the orchestration of continuous testing in a DevOps environment ▪ recognize key challenges when automating software testing ▪ work with Git for code versioning control ▪ work with the CircleCI continuous integration tool ▪ work with Docker containers for application packaging ▪ work with VirtualBox and Vagrant to provision infrastructure ▪ describe continuous integration, delivery, and testing, and list the benefits of continuous delivery and testing

Software Testing for DevOps: DevOps Test Tools

Objectives

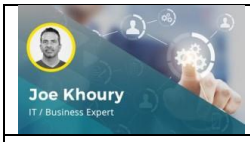
- recognize the various types of DevOps testing tools
- recognize key considerations when choosing DevOps test tools
- recognize the role that test automation plays in a DevOps culture
- recognize the role that test optimization plays in a DevOps culture
- recognize the importance of collaboration in a DevOps testing workflow
- describe the considerations necessary when generating a testing strategy in a DevOps culture
- work with test-driven development tools for automating acceptance tests
- work with performance and load testing tools
- work with Selenium to record and export user interactions with a Web browser
- work with Vagrant and Chef to create and configure infrastructure
- work with ChefSpec to generate and run unit tests against infrastructure
- describe types of DevOps test tools, list key factors for selecting DevOps test tools, describe the role that automation plays in DevOps, and work with Vagrant to create a test environment



Navigating Software Testing Tools: Types of Software Testing Tools

Objectives

- describe the various software testing tools for testing web applications, web services, system load, and databases
- describe the Selenium test automation tool and its advantages
- demonstrate how to use Selenium to test a web page
- describe Katalon Studio and the available web, API, and mobile testing modules
- demonstrate how to record and playback a web test using Katalon Studio
- demonstrate how to create an API test using Katalon Studio
- describe the benefits and features of SoapUI
- demonstrate how to perform REST API testing using SoapUI
- describe load testing and some of the tools available for performing load testing
- demonstrate how to perform load, stress, and endurance testing using NeoLoad
- describe how to regression test a relational database and available testing tools
- describe the features of unit testing tools and how to choose the best tool



Test Automation: Automated Software Testing

Objectives

- describe the steps to becoming an automated tester and challenges faced in automated testing
- describe the automated tester mindset and the goals of software testing
- describe the test plan for Agile automated testing and Agile testing strategies
- compare Agile and DevOps principles and define best practices for aligning test automation with both
- describe reasons why automated testing may not catch all risks
- describe the different types of automated testing including web applications, mobile devices, web service and data testing
- describe types of automated testing and goal of each test type
- compare the differences between automated and manual testing, the pros and cons of each method, and when to use each method
- describe automated testing, the automated testing process, and the scope of automated testing
- describe manual testing plans, tests cases, and defect reports
- describe the automated testing framework, framework guidelines, and types of software testing
- determine the best test cases to automate and how and when to test
-

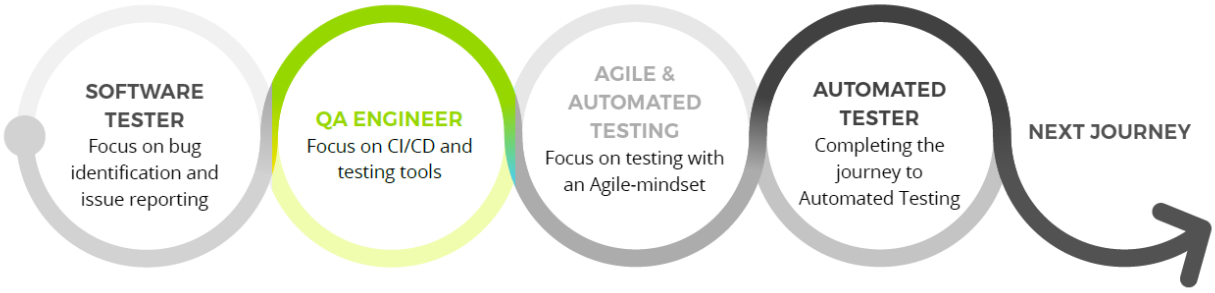
Exploring CI: Continuous Integration & CI Tools

Objectives

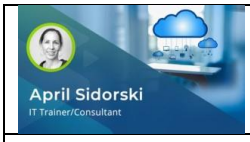
- discover the key concepts covered in this course
- describe the purpose of continuous integration and why it is important for software development
- describe the benefits of using continuous integration
- describe some of the best practices for using continuous integration
- describe common mistakes that companies make when trying to integrate and utilize continuous integration and how to avoid making those mistakes
- describe why continuous integration is crucial to developing high quality software systems and increasing customer satisfaction
- understand how continuous integration and automated testing are related and to how to create effective automated tests
- describe the stages of the continuous integration pipeline and the importance of each phase
- describe the continuous integration pipeline and how to configure it to efficiently run automated tests
- compare continuous integration, continuous delivery, and continuous deployment and how they are related to each other
- describe the features of common continuous integration tools
- work with CircleCI and GitHub for continuous integration
- work with Bamboo for continuous integration

DEVOPS JOURNEY

SOFTWARE TESTER TO DEVOPS AUTOMATED TESTER



Track 2: QA Specialist (duration: 8h 31m 50s)



API Management and DevOps

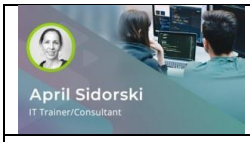
Objectives

- describe API and its typical lifecycle
- describe the different types of APIs and how they are managed
- describe API management and why it's required in software development
- describe the benefits of proper API management implementation
- recognize the role API management plays in the successful implementation of DevOps practices
- implement CI/CD pipelines for API management
- describe API management best practices
- provision and maximize API security
- recall API management solution tools
- implement API management with Azure API Management
- implement API management with Apigee API Management
- implement API management with AWS API Gateway
- recognize upcoming trends that are being adopted and tested for API management, with focus on AI and ML
-

Unit Testing in DevOps Software Development

Objectives

- describe unit testing, list the key properties of good unit testing programs, and differentiate between unit testing and integration testing
- recall the key components of unit testing frameworks and describe how unit testing frameworks can help developers write and execute tests, and review test results
- describe best practices for writing productive test cases and anti-patterns that should be avoided
- list unit testing tools that can be used to test Java and JavaScript programs, along with associated use case scenarios
- write and run unit tests using JsUnit and illustrate features provided in the JsUnit framework
- write unit test cases using QUnit and use assertions and callbacks in QUnit
- test server-side JavaScript using Mocha and Chai
- describe core testing and refactoring techniques that can help develop testable and maintainable code
- recognize guidelines and techniques that can help with writing trustworthy tests and approaches of managing bugs in tests
- create test cases and test suites using JUnit and illustrate the use of the Timeout and Ignore annotations
- use the @Test annotation and the invocationCount and threadPoolSize attributes to test web sites
- recognize the role of unit testing in test automation and use the AAA pattern to derive test cases
- recall the benefits of using test automation frameworks in the software delivery lifecycle
- describe the features of the popular test automation frameworks that can be used to automate test executions



Implementing Unit Testing Best Practices

Objectives

- discover the key concepts covered in this course
- describe Agile testing and testing strategies that can be adopted in the Agile software development process, with a focus on Agile Testing Quadrants
- list prominent unit testing patterns and scenarios where the patterns are applicable
- compare good and bad tests and specify the goals and approaches of writing good unit tests
- compare prominent code-driven unit testing frameworks that can be used to write unit tests in various programming languages
- implement the AAA unit testing pattern using JUnit and TestNG
- recognize the need for Test Double and describe how Dummy object, Fake object, stub class, and Mock object are used to facilitate test replacements
- apply best practices to the configuration and use of the Mockito API for implementing unit tests with Mock objects
- configure and write JUnit tests to test RESTful APIs with Mocha and Chai
- unit test web services using SOAP UI and custom code
- write unit tests using DBUnit to test data access objects that are written for database-driven applications
- implement unit testing using the test-driven development process based on sample scenarios
- implement automated testing with JUnit and Selenium to achieve enhanced browser compatibility

Role of Testing in Managing Clean Code

Objectives

- recognize common traits and approaches of identifying code that needs to be cleaned
- design applications with clean code
- list coding artifacts and describe best practices that should be adopted to create artifacts with clean code
- write applications that apply clean coding to artifacts like classes, interfaces, methods, and variable declarations
- recognize error handling styles that should be adopted for writing clean code
- manage code errors by adopting clean coding principles
- recall the significance of the unit testing F.I.R.S.T principle in writing clean tests
- recognize the fundamental principles of writing clean code, with focus on DRY, SOLID, and Law of Demeter
- recall the prominent testing strategies that should be adopted to maintain clean code
- describe the objective of testing clean code and the rules that are applied to implement clean tests
- write clean code in JavaScript
- recall TDD software development rules and the sequence of steps that should be followed to write clean code
- write applications by adopting the TDD principles of clean coding
-

Automated Testing with Docker

Objectives

- describe containerized testing and the benefits of using Docker for testing
- recognize the advantages and disadvantages of using Docker to set up architectures for automated testing
- configure and integrate GitHub repositories with Docker to enable automated build systems and continuous integration
- test automated builds on Docker Hub by executing tests when code is pushed or committed to GitHub
- configure continuous integration testing environments with Docker and Docker Compose
- dockerize tests and test environments and integrate them with test reporting tools
- recognize the features of test containers and list the prerequisites for setting up test containers
- recall patterns used for running tests inside Docker containers
- use Docker test containers in Java tests
- implement performance testing with Docker containers
- deploy Selenium Grid by running Selenium Hub and separate nodes for Chrome or Firefox
- integrate Selenium Grid with Docker and run Selenium scripts for testing

Kubernetes & Automation Testing

Objectives

- recognize the design principles and architecture behind Kubernetes and the essential components of Kubernetes master and Kubernetes worker
- list the essential objects and controllers that we can use to setup Kubernetes clusters
- run applications by creating Kubernetes deployment objects and scale the applications by increasing the replica count
- deploy three-tier web applications using Kubernetes and configure Kubernetes to ensure the front-end instances communicate with the back-end instances and the back-end instances communicate with the databases
- create single-zone clusters with the default features enabled in Google Kubernetes Engine and create volume resources in the clusters
- package web applications in Docker container images and run the container images on Google Kubernetes Engine clusters
- recognize the different ways that Kubernetes helps with software testing, with a focus on end-to-end testing
- recall the various types of testing that can be automated with Kubernetes and the advantages of deploying test containers in Kubernetes
- recognize when to use and when not to use Kubernetes for automation testing
- recognize the advantages and disadvantages of implementing test automation using Selenium Grid and Kubernetes
- set up test environments using Selenium Grid, Docker, and Kubernetes and spin up Chrome and Firefox nodes
- build test container images containing test files, libraries, and drivers
- use Selenium WebDriver to implement and execute end-to-end browser-related tests
- deploy test containers in Kubernetes clusters and publish the test results

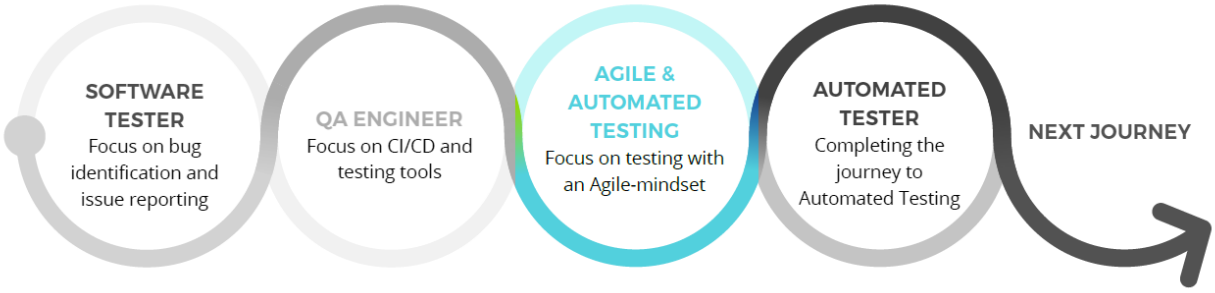
AWS Automated Testing: Environments & Services

Objectives

- describe how AWS-based environments are used for automated testing and the associated advantages and disadvantages
- list the prominent AWS development and testing tools
- recall best practices for using AWS to set up environments for automated testing
- set up an AWS environment for automated testing
- use AWS Device Farm for automated testing of iOS and Android applications
- use AWS Device Farm for automated testing of web applications
- work with AWS CodePipeline to automate the build and test phases of continuous delivery processes
- use AWS CodeBuild to add build and test automation in existing AWS CodePipeline pipelines
- work with AWS CodeDeploy to validate and debug applications before deploying them
- implement unit testing as a part of AWS CodeStar projects
- work with AWS CloudFormation to implement Infrastructure as Code for automated testing of applications
- use AWS OpsWorks for Chef Automate to automate testing for security and compliance

DEVOPS JOURNEY

SOFTWARE TESTER TO DEVOPS AUTOMATED TESTER



Track 3: QA Lead (duration: 6h 51m 50s)

Comparing Manual & Automated Testing

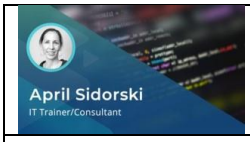
Objectives

- describe the SDLC, including its models and how its phases help to produce high-quality systems
- describe manual testing, including its types and goals
- list the frameworks that can be used to implement manual testing and describe their features, advantages, and disadvantages
- recognize the prominent models used in manual testing
- create test case templates for manual testing that apply best practices
- use Postman for manual testing
- describe automated testing, including its types and goals
- list frameworks that can be used to implement automated testing and describe its features, advantages, and disadvantages
- list frameworks that can be used to implement automated testing and describe its features, advantages, and disadvantages
- differentiate between manual and automated testing and describe where manual or automated testing should be implemented
- configure Selenium WebDriver for automation testing
- describe the preferred conditions where automated and manual testing can be implemented
- execute test cases with the implementation of automated testing using the QTP/UFT Linear Framework

Software Testing Cycle: Agile Testing

Objectives

- recognize the Agile methodologies that can be used to adopt the Agile testing paradigm
- recall the benefits and challenges associated with Agile testing and describe the best practices and guidelines that can be adopted to implement Agile testing practices
- recognize the lifecycle phases for testing within Agile frameworks
- use the Cucumber BDD framework to write acceptance tests
- describe approaches for tracking test progress and the product quality metrics that can be used to evaluate the outcomes of Agile test implementations
- define the responsibilities of testers in Agile projects and describe tools and products that can be used to set up Agile testing workflows
- recognize the different levels of tests that should be covered as a part of the Agile testing mechanism
- work with Cucumber-JVM and Selenium WebDriver in Java to implement Agile testing with Behavior Driven Development
- describe the role of Agile Testing Quadrants and how the quadrants reflect the What-Testing-When approach
- configure test management in Jira to implement Agile testing for REST APIs
- define the Whole Team Approach and list the different elements used to derive an Agile test strategy while applying the Agile methodology
- configure Agile project management using Scrum and Kanban methods with Jira



Test Driven Development: Implementing TDD

Objectives

- differentiate between SDLC and STLC, including features and uses
- describe TDD, its methodology, advantages, and usage scenarios
- describe AMDD, differentiate between TDD and AMDD, and recognize the edge AMDD has over TDD
- describe the implementation of test driven development in data science workflows
- refactor code using best practices
- implement TDD with the utilization of best practices
- describe tips for mastering TDD, the drawbacks of TDD, and the methodology that can be implemented in place of TDD
- implement TDD using the Jest Testing Framework
- implement BDD best practices
- implement TDD unit testing in Python
- apply the TDD process in microservices
- configure and use keywords in the Cucumber framework

End-to-End Testing Implementation

Objectives

- describe end-to-end testing and recognize the need for it
- recognize the processes involved in end-to-end testing and approaches for creating end-to-end test cases
- describe system testing and differentiate between system testing and end-to-end testing
- implement the basics of unit testing, taking test-driven development into consideration
- automate browsers
- automate testing using Selenium with Python
- recall the advantages and disadvantages of end-to-end testing
- configure Cypress and describe the advantages of working with Cypress
- test web components using Cypress with relatable use case scenarios
- describe continuous integration concepts and continuous integration servers from the perspective of testing
- implement integration testing using JUnit and Spring
- implement integration testing in a JavaScript framework based on real-life examples

Test Framework Modification

Objectives

- describe the generations of automated test frameworks, as well as their features, advantages, and disadvantages
- list the popular open source test automation frameworks with comparative modification granularity
- describe design considerations for modifying test frameworks to accelerate productivity
- recall best practises for modifying test frameworks to build Agile-friendly test automation frameworks
- recognize the benefits and risks associated with modifying test frameworks
- identify techniques that can be used to refactor code in test automation frameworks
- modify the Selenium framework by creating and destroying WebDriver objects using Selenium Core
- modify Selenium to implement Page Object Patterns using the Selenium Test class
- modify the report generation mechanism of testing frameworks using ExtentReport and Allure
- modify the methods of the Chai framework without destroying core functionality
- use Jasmine to implement a Behaviour-driven Development framework for testing JavaScript code
- recall the design patterns that can be implemented in automation testing frameworks to provide reliable test automation and improve test logics
- execute WebDriverIO in standalone mode and use its API to run automation testing

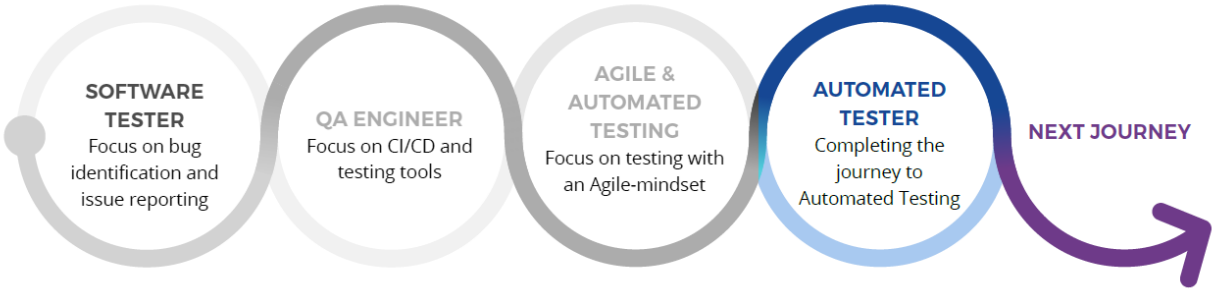
CI/CD Best Practices: Applying Automated Testing

Objectives

- recall the DevOps CI/CD process and describe approaches for switching from traditional practices to CI/CD practices
- recognize the risks and challenges associated with CI/CD and approaches for eliminating them
- describe CI/CD pipelines and the benefits of implementing automated testing for CI/CD
- classify the different CI/CD pipeline architectures that can be set up to implement automated testing
- describe CI/CD framework features that can help accelerate continuous integration with Jenkins test automation
- list CI/CD tools that are used to build next-gen delivery pipelines and their associated features
- configure Jenkins with the supported tools needed to set up continuous integration from the perspective of test automation
- use Git Hooks to automate development and deployment tasks
- build continuous integration and continuous delivery pipelines
- set up architectures of CI/CD pipelines for container automation
- recognize the core technologies and strategies that are used to implement continuous testing and deployment with containers
- implement continuous testing using Selenium and Jenkins

DEVOPS JOURNEY

SOFTWARE TESTER TO DEVOPS AUTOMATED TESTER



Track 4: Automated Tester (duration: 7h 54m 50s)

API Test Automation with SoapUI

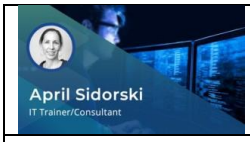
Objectives

- recall the features provided by SoapUI along with the concept of WSDL
- compare the differences between SoapUI and SoapUI NG Pro from the perspective of their supporting technologies and features
- configure SoapUI to test REST-based web services
- create Test Suites and Test Cases using SoapUI to test APIs
- demonstrate how to create projects in SoapUI
- demonstrate the procedure of adding WSDL to SoapUI projects
- define the concept of assertion and identify the assertions that are available to use in SoapUI
- create Test Suites in SoapUI projects and add assertions in the Test Suites
- demonstrate the process of creating REST API projects in SoapUI
- demonstrate the use of Groovy scripts along with the different types of operators in SoapUI
- integrate SoapUI Pro functional and security tests into CI/CD process
- demonstrate the steps involved in testing web service integration with SoapUI in Jenkins

BDD Testing with Cucumber

Objectives

- define the concept of Behaviour-Driven Development and list its important features
- compare the differences between BDD and TDD from the perspective of the advantages and disadvantages along with the prominent reason why we should use BDD
- recognize the features and advantages of Cucumber and compare the differences between Cucumber, Selenium, and ALM
- recall the overall sequence workflow and the testing stack of Cucumber
- define the concept of Gherkin and illustrate why we need to take it into account with focus on Gherkin syntax, important terms, and best practices
- configure Cucumber for BDD testing and illustrate the feature file & scenarios of BDD testing with Cucumber
- demonstrate how to work with Runner, Step definition, Parameter, and Gherkin script in Cucumber
- perform data-driven testing with Cucumber using data tables
- demonstrate the procedure of generating reports in Cucumber
- demonstrate how to use different hooks in Cucumber
- run Cucumber Feature file with TestNG
- write Gherkin script for API testing



HP UFT: Functional Test Automation

Objectives

- describe functional testing and its objectives
- list UFT features and the disadvantages associated with it
- define the testing workflows that can be used to implement the Unified Testing Process
- recall the various types of tests supported by UFT
- list HP QTP features and describe how conducting automated functional tests can help testers execute automated tests
- describe the role of object repository and differentiate between local and shared object repository
- record testing steps and use checkpoints
- set, identify, and use text checkpoints when testing web applications
- set and identify standard checkpoints when testing web applications
- use bitmap checkpoints on test components
- use Regular Expressions in UFT and recognize steps for handling dynamic property values in object repository
- declare transactions in the identified sections of a test

Test Automation with TestComplete

Objectives

- recall the basic concept, process, benefits, and tools that we can use for automation testing
- recognize the features of TestComplete and the different types of testing it supports
- describe the TestComplete Object Model and its relevance in automated testing
- describe name mapping, its uses, and the object identification criteria
- describe data-driven testing, as well as TestComplete data storage types and common tasks
- configure TestComplete for automation testing
- use the various options and features that are provided in the TestComplete IDE
- create or initialize projects in TestComplete
- record test cases for functional testing using TestComplete
- implement web testing in TestComplete and describe the objectives of the specifics involved in web testing
- create checkpoints for testing web applications with TestComplete
- test web applications with the use of test suites in TestComplete

Automation Testing for Python

Objectives

- recall the concept of automated testing and classify the different types of automation testing
- list the prominent frameworks that we can use to test Python-based applications
- list the features of DocTest, Nose, and UnitTest that can used to automate testing of Python-based applications
- implement unit testing in Pytest using the PyCharm IDE
- demonstrate the procedure to parameterize tests using Pytest
- configure the Robot Framework for Python testing
- demonstrate the procedure of writing Robot file and executing Python tests using the Robot Framework
- build and test APIs using Flask
- demonstrate the process of testing APIs built in Flask using Postman
- configure the Behave framework for Python testing
- implement BDD testing in Behave framework by writing feature files
- automate testing of web components using Selenium with Python

Automated Testing with Model-based Testing

Objectives

- define the concept of model-based testing along with the benefits and challenges associated with model-based testing
- classify the different types of model-based testing frameworks along with the different models that are used in model-based testing
- recognize the various techniques of model-based testing that can be used to implement effective quality assurance of software or application systems
- list the prominent tools that we can use to implement model-based testing along with their associated features
- configure the ModelJUnit framework in development environments to write and execute model-based tests
- write simple finite state machine or extended finite state machine models as Java classes, generate tests from those models, and measure various model coverage metrics
- demonstrate the steps involved in creating models using ModelJUnit and testing mock implementations using the created models
- install and configure MoMuT and then generate test cases from UML state machines
- configure the OSMO MBT Tool and demonstrate how to generate and execute test cases using test models
- demonstrate the different approach of using OSMO Tester to model data in model programs
- recognize the model-based testing workflow along with the prominent approaches of deploying model-based testing
- demonstrate the approach of implementing model-based testing using GraphWalker and Selenium

Selenium: Using Selenium for Continuous Testing

Objectives

- describe functional automation and the benefits and weakness of using Selenium for functional automation
- describe the architecture of Selenium, the components of Selenium Test Suite, and the basic elements of Selenium test scripts that are applicable to any test case
- install the Selenium IDE and configure the Firebug and Firepath add-ons
- recognize architectural components and features WebDriver and compare WebDriver with RC
- record test cases using Selenium IDE and export the recorded test cases using Java, JUnit 4, and WebDriver
- execute Selenium test scripts on Chrome and Firefox browsers
- describe locating strategies that can be used to specify the locations of web elements
- work with location strategies using Selenium locators to identify and locate web elements
- use of hard and soft assertions in Selenium to determine the state of applications
- list the major steps that are involved in building maintainable Selenium frameworks
- configure Selenium with Maven and Bamboo to implement continuous integration testing as a part of CI/CD pipelines
- integrate Jenkins with Selenium to set up testing as a part of CI/CD pipelines
- demonstrate the approach of automating scenarios such as hover menus and drag-and-drop controls using the Advanced User Interaction API of Selenium
- work with Selenium WebDriver Compatibility and cross-browser tests and automating in the cloud using Browserstack
- work with Selenium Headless Browser testing using PhantomJS and HTMLUnit browsers

Building & Implementing Test Automation Frameworks

Objectives

- list the different types of environments for software testing automation
- describe test automation frameworks and list key parameters for designing and developing them
- recognize prominent test automation frameworks and describe their features, advantages, and disadvantages
- list the different types of test automation frameworks that can be implemented using Selenium WebDriver
- build a Data Driven Framework using Selenium
- implement Data Driven Automation Framework using TestNG with the @dataProvider annotation
- implement a Keyword Driven Framework using Selenium WebDriver and Excel Test Suite
- build Hybrid Automation Framework using a collection of two or more frameworks
- set up Jenkins to run continuous integration tests from Maven projects using Selenium WebDriver
- simulate a combination of keystrokes, mouse movement, and window or control manipulation in order to automate non-web user interfaces in Selenium WebDriver
- use Docker and Selenium to build containerized test automation environments
- recall best practices for building test automation frameworks in Agile and DevOps

Productivity Tools for DevOps Engineer

Productivity Tools for DevOps Automated Testers

Optional

COURSE
Signing in & Setting up a Team
5

COURSE
Using the Conversation Tools
3

COURSE
Creating & Managing Projects
4

COURSE
Finding & Sharing Items
4

COURSE
Running Reports & Configuring Projects
4

COURSE
Configuring Spaces
8

COURSE
Sign-in & Setup
1

COURSE
Communication Tools
3

COURSE
Working with Groups
1

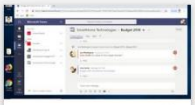
COURSE
Creating, Finding, & Sharing Information
1

COURSE
Configuring Convo
2

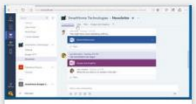
COURSE
The Convo iOS App
1

COURSE
Introducing the JIRA Platform
100

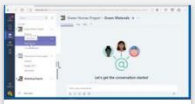
COURSE
Leveraging the JIRA Platform for Development Projects
57



COURSE
Sign-in & Setup
149

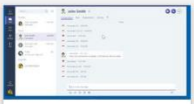


COURSE
Teams & Channels
121



COURSE
Conversation Tools
96

COURSE
Creating, Finding, & Sharing Information
88



COURSE
Call & Meeting Tools
77

COURSE
Signing in & Setting Up
14

COURSE
Using Channels
6

COURSE
Private Messaging & Communication Tools
10

COURSE
Creating, Finding & Sharing Information
5

COURSE
Configuring Slack
23

COURSE
Using the iOS App
3

COURSE
Setting Up
32

COURSE
Posting & Reacting to Status Updates
22

COURSE
Using Groups
4

COURSE
Collaborating & Communicating
34

COURSE
Configuring Networks
14

Bookshelf

Bookshelf Optional

BOOK

Software Testing: Concepts and Operations

22

BOOK

Software Testing Automation Tips: 50 Things Automation...

9

BOOK

Software Testing Foundations: A Study Guide...

6

BOOK

Guide to Advanced Software Testing, Second Edition

1

BOOK

Managing the Testing Process: Practical Tools and...

40

BOOK

Software Testing in the Cloud: Perspectives on an...

BOOK

Analytic Methods in Systems and Software Testing

BOOK

Creating Maintainable APIs: A Practical, Case-Study...

BOOK

Advanced API Security

9

BOOK

API Development: A Practical Guide for Busines...

1

BOOK

Practical API Architecture and Development with Azu...

14

BOOK

Generic Pipelines Using Docker: The DevOps Guide...

1

BOOK

Kubernetes in Action

18

BOOK

Amazon Web Services in Action, Second Edition

8

BOOK

AWS for Developers for Dummies

32

BOOK

Agile Metrics in Action: How to Measure and Improve...

1

BOOK

Agile Testing Foundations: An ISTQB Foundation Level...

27

BOOK

Python Continuous Integration and Delivery: A...

3

BOOK

Complete Guide to Test Automation: Techniques,...

3

BOOK

Cucumber Cookbook

27

BOOK

Testing Python: Applying Unit Testing, TDD, BDD and...

1

BOOK

Python Unit Test Automation: Practical...

4

BOOK

Hands-On Functional Test Automation: With Visual...

17

BOOK

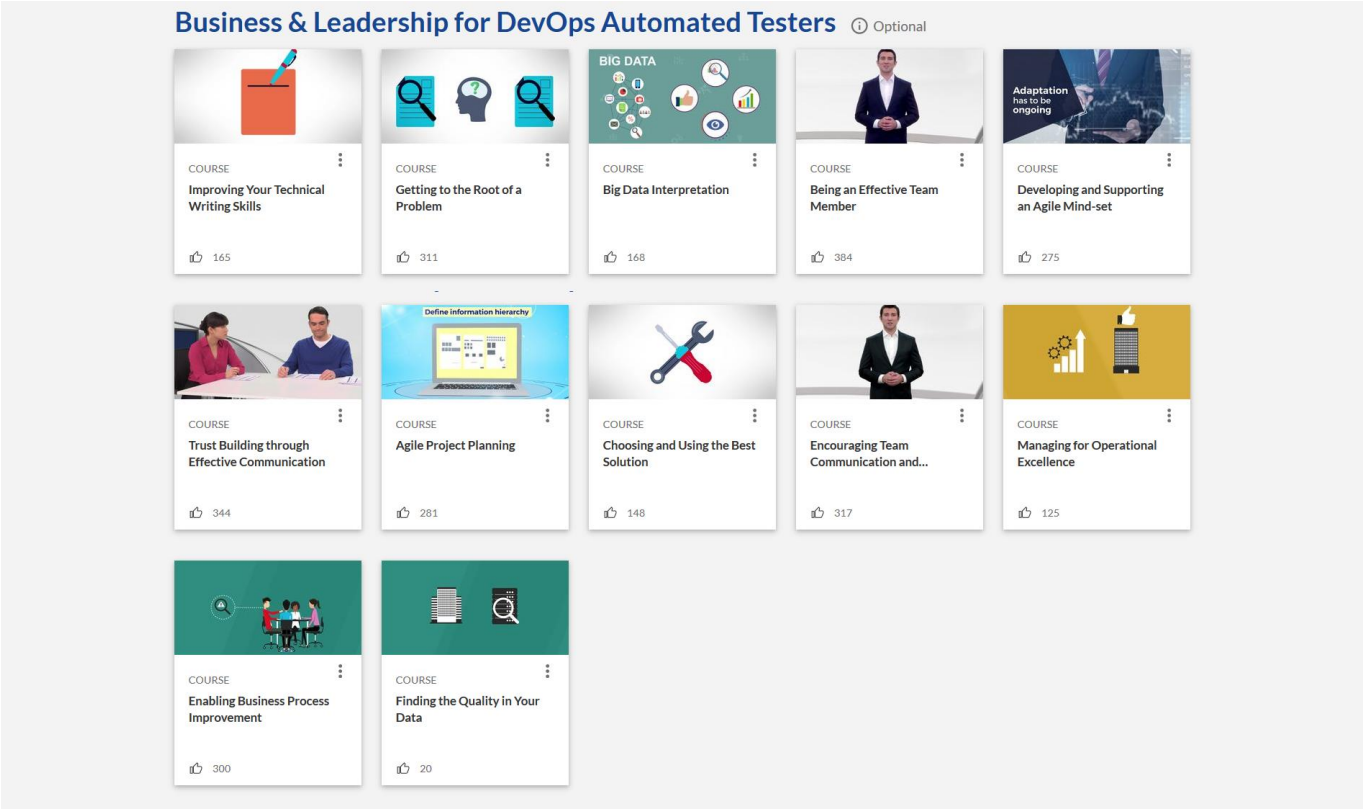
Java Unit Testing with JUnit 5: Test Driven Development...

11

BOOK

The Kitty Hawk Venture: A Novel About Continuous...

Business & Leadership for DevOps Engineer



FOLLOW US ON:

www.skilltech.pl

email: biuro@skilltech.pl

tel. +48 22 44 88 827

SkillTech
Technology hired for excellence