



AZ400 - Version: 1
01 May 2021

Azure DevOps Engineer Expert



Azure DevOps Engineer Expert

AZ400 - Version: 1

 7 days Course

Description:

Azure DevOps professionals combine people, process, and technologies to continuously deliver valuable products and services that meet end user needs and business objectives.

This 7-days course is comprised of 7 separate 1- day courses. This course will help you prepare to the Microsoft AZ-400 exam. You can review the daily syllabuses in the links below:
</br> Day 1: Implementing DevOps Development Processes

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t01a&branchName=165>

</br> Day 2:

Implementing Continuous Integration

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t02a&branchName=165>

</br> Day 3: Implementing Continuous Delivery

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t03a&branchName=165>

</br> Day 4:

Implementing Dependency Management

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t04a&branchName=165>

</br> Day 5: Implementing Application Infrastructure

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t05a&branchName=165>

</br> Day 6: Implementing Continuous Feedback

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t06a&branchName=165>

</br> Day 7: Designing a DevOps Strategy

<https://scc.sela.co.il/Syl/Syllabus/Info?courseCode=az400t07a&branchName=165>

</br>

Intended audience:

Students in this course are interested in implementing dependency management or in passing the Microsoft Azure DevOps Solutions certification exam.

Prerequisites:

Fundamental knowledge about Azure, version control, Agile software development, and core software development principles. It would be helpful to have experience in an organization that delivers software.

Objectives:

Describe the benefits of using Source Control

Migrate from TFVC to Git

Scale Git for Enterprise DevOps

Implement and manage build infrastructure

Manage application config and secrets

Implement a mobile DevOps strategy

Continuous Integration Overview

Implementing a Build Strategy

Manage code quality including: technical debt SonarCloud, and other tooling solutions.

Manage security policies with open source, OWASP, and WhiteSource Bolt.

Manage code quality including: technical debt, SonarCloud, and other tooling solutions.

Implement a container strategy including how containers are different from virtual machines and how microservices use containers.

Implement containers using Docker.

Differentiate between a release and a deployment

Define the components of a release pipeline

Explain things to consider when designing your release strategy

Classify a release versus a release process and outline how to control the quality of both

Describe the principle of release gates and how to deal with release notes and documentation

Explain deployment patterns, both in the traditional sense and in the modern sense

Choose a release management tool

Explain the terminology used in Azure DevOps and other Release Management Tooling

Describe what a Build and Release task is, what it can do, and some available deployment tasks

Classify an Agent, Agent Queue, and Agent Pool

Explain why you sometimes need multiple release jobs in one release pipeline

Differentiate between multi-agent and multi-configuration release job

Use release variables and stage variables in your release pipeline

Deploy to an environment securely using a service connection

Embed testing in the pipeline

List the different ways to inspect the health of your pipeline and release by using alerts, service hooks, and reports

Create a release gate

Describe deployment patterns

Implement Blue Green Deployment

Implement Canary Release

Implement Progressive Exposure Deployment

Recommend artifact management tools and practices

Abstract common packages to enable sharing and reuse

Inspect codebase to identify code dependencies that can be converted to packages

Identify and recommend standardized package types and versions across the solution

Refactor existing build pipelines to implement version strategy that publishes packages

Manage security and compliance

Inspect open source software packages for security and license compliance to align with corporate standards

Configure build pipeline to access package security and license rating

Configure secure access to package feeds

Apply infrastructure and configuration as code principles

Deploy and manage infrastructure using Microsoft automation technologies such as ARM templates, PowerShell, and Azure CLI

Describe deployment models and services that are available with AzureDeploy and configure a Managed Kubernetes cluster

Deploy and configure infrastructure using 3rd party tools and services with Azure, such as Chef, Puppet, Ansible, SaltStack, and Terraform

Define an infrastructure and configuration strategy and appropriate toolset for a release pipeline and application infrastructure

Implement compliance and security in your application infrastructure

Design practices to measure end-user satisfaction

Design processes to capture and analyze user feedback from external sources

Design routing for client application crash report data

Recommend monitoring tools and technologies
Recommend system and feature usage tracking tools
Configure crash report integration for client applications
Develop monitoring and status dashboards
Implement routing for client application crash report data
Implement tools to track system usage, feature usage, and flow
Integrate and configure ticketing systems with development team's work management
Analyze alerts to establish a baseline
Analyze telemetry to establish a baseline
Perform live site reviews and capture feedback for system outages
Perform ongoing tuning to reduce meaningless or non-actionable alerts
Plan for the transformation with shared goals and timelines
Select a project and identify project metrics and KPIs
Create a team and agile organizational structure
Develop a project quality strategy
Plan for secure development practices and compliance rules.
Migrate and consolidate artifacts
Migrate and integrate source control measures

Topics:

Day 1

- Module 1: Getting started with Source Control
 - What is Source Control?
 - Benefits of Source Control
 - Types of Source Control systems
 - Introduction to Azure Repos
 - Migrating from Team Foundation Version Control (TFVC) to Git
 - Authenticating to your Git Repos
- Module 2: Scaling git for enterprise DevOps
 - How to structure your Git repo
 - Git Branching workflows
 - Collaborating with Pull Requests
 - Why care about GitHooks?
 - Fostering Internal Open Source

- Git Version
- Public projects
- Files in Git
- Module 3: Implement & Manage Build Infrastructure
 - The concept of pipelines in DevOps
 - Azure Pipelines
 - Evaluate use of Hosted vs Private Agents
 - Agent pools
 - Pipelines and concurrency
 - Azure DevOps and Open Source projects
 - Azure Pipelines YAML vs Visual Designer
 - Setup private agents
 - Integrate Jenkins with Azure Pipelines
 - Integration external source control with Azure Pipelines
 - Analyze and integrate Docker multi-stage builds
- Module 4: Managing application config & secrets
 - Introduction to Security
 - Implement secure and compliant development process
 - Rethinking application config data
 - Manage secrets, tokens, and certificates
 - Implement tools for managing security and compliance in a pipeline
- Module 5: Implement a mobile DevOps strategy
 - Introduction to Mobile DevOps
 - Introduction to Visual Studio App Center
 - Manage mobile target device sets and distribution groups
 - Manage target UI test device sets
 - Provision tester devices for deployment
 - Create public and private distribution groups

Day 2

- Module 1: Implementing Continuous Integration in an Azure DevOps Pipeline
 - Continuous Integration Overview
 - Implementing a Build Strategy
 - Lab : Enabling Continuous Integration with Azure Pipelines
 - Lab : Creating a Jenkins Build Job and Triggering CI

- Module 2: Managing Code Quality and Security Policies
 - Managing Code Quality
 - Managing Security Policies
 - Lab : Managing Technical Debt with Azure DevOps and SonarCloud
 - Lab : Checking Vulnerabilities using WhiteSource Bolt and Azure DevOps
- Module 3: Implementing a Container Build Strategy
 - Implementing a Container Build Strategy
 - Lab : Existing .NET Applications with Azure and Docker Images

Day 3

- Module 1: Design a Release Strategy
 - Lessons
 - Introduction to Continuous Delivery
 - Release strategy recommendations
 - Building a High Quality Release pipeline
 - Choosing a deployment pattern
 - Choosing the right release management tool
 - Lab : Building a release strategy
- Module 2: Set up a Release Management Workflow
 - Create a Release Pipeline
 - Provision and Configure Environments
 - Manage and Modularize Tasks and Templates
 - Integrate Secrets with the release pipeline
 - Configure Automated Integration and Functional Test Automation
 - Automate Inspection of Health
 - Lab : Automating your infrastructure deployments in the Cloud with Terraform and Azure Pipelines
 - Lab : Setting up secrets in the pipeline with Azure Key vault
 - Lab : Setting up and Running Load Tests
 - Lab : Setting up and Running Functional Tests
 - Lab : Using Azure Monitor as release gate
 - Lab : Creating a Release Dashboard
- Module 3: Implement an appropriate deployment pattern
 - Introduction to Deployment Patterns
 - Implement Blue Green Deployment

- Feature Toggles
- Canary Releases
- Dark Launching
- AB Testing
- Progressive Exposure Deployment
- Lab : Blue-Green Deployments
- Lab : Traffic Manager

Day 4

- Module 1: Designing a Dependency Management Strategy
 - Introduction
 - Packaging dependencies
 - Package management
 - Implement a versioning strategy
 - Lab : Updating packages
- Module 2: Manage security and compliance
 - Introduction
 - Package security
 - Open source software
 - Integrating license and vulnerability scans

Day 5

- Module 1: Infrastructure and Configuration Azure Tools
 - Learning Objectives
 - Infrastructure as Code and Configuration Management
 - Create Azure Resources using ARM Templates
 - Create Azure Resources using Azure CLI
 - Create Azure Resources using Azure PowerShell
 - Additional Automation Tools
 - Version Control
 - Lab Deploy to Azure using ARM templates
 - Module Review Questions
- Module 2: Azure Deployment Models and Services
 - Learning Objectives

- Deployment Models and Options
- Azure Infrastructure-as-a-Service (IaaS) Services
- Azure Automation with DevOps
- Desired State Configuration (DSC)
- Azure Platform-as-a-Service (PaaS) services
- Azure Service Fabric
- Lab Azure Automation - IaaS or PaaS deployment
- Module Review Questions
- Module 3: Create and Manage Kubernetes Service Infrastructure
 - Learning Objectives
 - Azure Kubernetes Service
 - Lab Deploy and Scale AKS Cluster
 - Module Review Questions
- Module 4: Third Party and Open Source Tools available with Azure
 - Learning Objectives
 - Chef
 - Puppet
 - Ansible
 - Cloud-Init
 - Terraform
 - Lab Provision and configure an App in Azure Using X
 - Module Review Questions
- Module 5: Implement Compliance and Security in your Infrastructure
 - Security and Compliance Principles with DevOps
 - Azure Security Center
 - Lab Integrate a scanning extension or tool in an Azure DevOps pipeline/security center
 - Module Review Questions
- Module 6: Course Completion
 - Final Exam

Day 6

- Module 1: Recommend and design system feedback mechanisms
 - The inner loop
 - Continuous Experimentation mindset

- Design practices to measure end-user satisfaction
- Design processes to capture and analyze user feedback
- Design process to automate application analytics
- Lab : Integration between Azure DevOps and Teams
- Module 2: Implement process for routing system feedback to development teams
 - Implement tools to track system usage, feature usage, and flow
 - Implement routing for mobile application crash report data
 - Develop monitoring and status dashboards
 - Integrate and configure ticketing systems
- Module 3: Optimize feedback mechanisms
 - Site Reliability Engineering
 - Analyze telemetry to establish a baseline
 - Perform ongoing tuning to reduce meaningless or non-actionable alerts
 - Analyze alerts to establish a baseline
 - Blameless PostMortems and a Just Culture

Day 7

- Module 1: Planning for DevOps
 - Transformation Planning
 - Project Selection
 - Team Structures
- Module 2: Planning for Quality and Security
 - Planning a Quality Strategy
 - Planning Secure Development
- Module 3: Migrating and Consolidating Artifacts and Tools
 - Migrating and Consolidating Artifacts
 - Migrating and Integrating Source Control