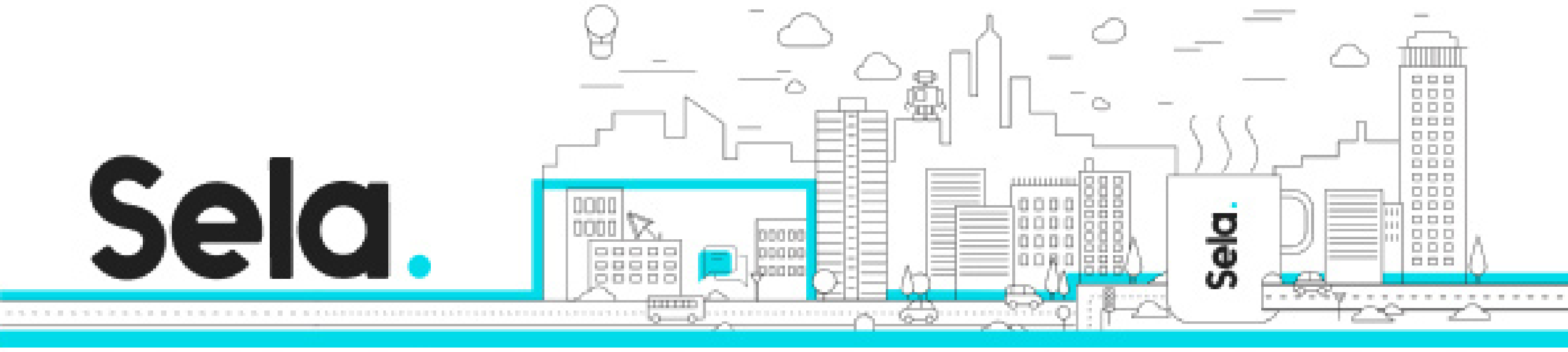
AZ400

Designing and Implementing Microsoft DevOps solutions

college@sela.co.il 03-6176666



Designing and Implementing Microsoft DevOps solutions

AZ400 - Version: 2

5 days Course

Description:

This course provides the knowledge and skills to design and implement DevOps processes and practices. Students will learn how to plan for DevOps, use source control, scale Git for an enterprise, consolidate artifacts, design a dependency management strategy, manage secrets, implement continuous integration, implement a container build strategy, design a release strategy, set up a release management workflow, implement a deployment pattern, and optimize feedback mechanisms.

Intended audience:

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

Prerequisites:

Cloud computing concepts, including an understanding of PaaS, SaaS, and IaaS implementations.

Both Azure administration and Azure development with proven expertise in at least one of these areas.

Version control, Agile software development, and core software development principles. It would be helpful to have experience in an organization that delivers software.

Objectives:

Plan for the transformation with shared goals and timelines

Select a project and identify project metrics and Key Performance Indicators (KPI's)

Create a team and agile organizational structure

Design a tool integration strategy

Design a license management strategy (e.g., Azure DevOps and GitHub users)

Design a strategy for end-to-end traceability from work items to working software

Design an authentication and access strategy

Design a strategy for integrating on-premises and cloud resources

Describe the benefits of using Source Control

Describe Azure Repos and GitHub

Migrate from TFVC to Git

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

Build organizational knowledge on code quality

Explain how to structure Git repos

Describe Git branching workflows

Leverage pull requests for collaboration and code reviews

Leverage Git hooks for automation

Use Git to foster inner source across the organization

Explain the role of Azure Pipelines and its components

Configure Agents for use in Azure Pipelines

Explain why continuous integration matters

Implement continuous integration using Azure Pipelines

Define Site Reliability Engineering

Design processes to measure end-user satisfaction and analyze user feedback

Design processes to automate application analytics

Manage alerts and reduce meaningless and non-actionable alerts

Carry out blameless retrospectives and create a just culture

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

Implement compliance and security in your application infrastructure
Describe the potential challenges with integrating open-source software
Inspect open-source software packages for security and license compliance
Manage organizational security and compliance policies
Integrate license and vulnerability scans into build and deployment pipelines
Configure build pipelines to access package security and license ratings

Topics:

Module 1: Get started on a DevOps transformation journey

- Introduction to DevOps
- Choose the right project
- Describe team structures
- Migrate to DevOps
- Introduction to source control
- Describe types of source control systems
- Work with Azure Repos and GitHub

Module 2: Work with Git for enterprise DevOps

- Structure your Git Repo
- Manage Git branches and workflows
- Collaborate with pull requests in Azure Repos
- Explore Git hooks
- Plan fostering inner source
- Manage Git repositories

Module 3: Implement CI with Azure Pipelines and GitHub Actions

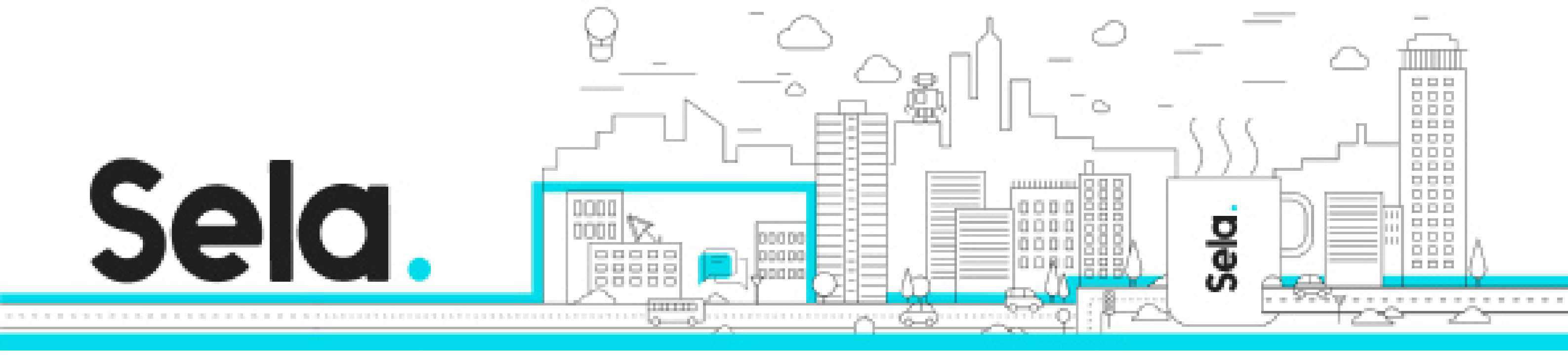
- Explore Azure Pipelines
- Manage Azure Pipeline agents and pools
- Describe pipelines and concurrency
- Explore Continuous integration
- Implement a pipeline strategy
- Integrate with Azure Pipelines
- Introduction to GitHub Actions
- Learn continuous integration with GitHub Actions

Module 4: Design and implement a release strategy

- Introduction to continuous delivery
- Explore release strategy recommendations
- Build a high-quality release pipeline
- Introduction to deployment patterns
- Implement blue-green deployment and feature toggles
- Implement canary releases and dark launching
- Implement A-B testing and progressive exposure deployment

Module 5: Implement a secure continuous deployment using Azure Pipelines

- Create a release pipeline
- Configure and provision environments
- Manage and modularize tasks and templates
- Automate inspection of health
- Introduction to security development process
- Manage application configuration data
- Integrate with identity management systems
- Implement application configuration



Module 6: Manage infrastructure as code using Azure, DSC, and third-party tools

- Explore infrastructure as code and configuration management
- Create Azure resources using Azure Resource Manager templates
- Create Azure resources by using Azure CLI
- Explore Azure Automation with DevOps
- Implement Desired State Configuration (DSC)
- Introduction to Chef and Puppet
- Implement Ansible
- Implement Terraform

Module 7: Design and implement a dependency management strategy

- Explore package dependencies
- Understand package management
- Migrate, consolidating and secure artifacts
- Implement a versioning strategy

Module 8: Create and manage containers using Docker and Kubernetes

- Design a container build strategy
- Implement Docker multi-stage builds
- Implement Azure Kubernetes Service (AKS)
- Explore Kubernetes tooling
- Integrate AKS with Pipelines

Module 9: Implement continuous feedback

- Design a container build strategy
- Implement Docker multi-stage builds
- Implement Azure Kubernetes Service (AKS)
- Explore Kubernetes tooling
- Integrate AKS with Pipelines

Module 10: Implement security and validate code bases for compliance

- Understand security in the Pipeline
- Introduction to Azure Security Center
- Implement open-source software
- Manage anti-malware and anti-spam policies
- Integrate license and vulnerability scans
- Identify technical debt