AZ-400: Designing and Implementing Microsoft DevOps solutions

Microsoft - Azure Apps & Infrastructure

Promoção: **Curso com 20% de desconto**Aproveite já!

Live Training (também disponível em presencial)

- **Localidade:** Imprimir Curso
- **Data:** 30 May 2022
- **Preço:** 1940 € (Os valores apresentados não incluem IVA. Oferta de IVA a particulares e estudantes.)
- **Horário:** Laboral das das 9h00 às 18h00
- **Nível:** Avançado
- **Duração:** 40h

Sobre o curso

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.

Destinatários

- Students in this course are interested in designing and implementing DevOps processes or in passing the Microsoft Azure DevOps Solutions certification exam.

Objetivos

- 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

---

Pré-requisitos

Successful learners will have prior knowledge and understanding of:

- 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.

- If you are new to Azure and cloud computing, consider take AZ-900: Azure Fundamentals
- If you are new to Azure Administration, consider take AZ-104: Microsoft Azure Administrator and AZ-010: Azure Administration for AWS SysOps
- If you are new to Azure Developer, consider take AZ-204: Developing Solutions for Microsoft Azure

---

Programa

- Planning for DevOps
- Getting Started with Source Control
- Managing Technical Debt
- Working with Git for Enterprise DevOps
- Configuring Azure Pipelines
- Implementing Continuous Integration using Azure Pipelines
- Managing Application Configuration and Secrets
- Implementing Continuous Integration with GitHub Actions
- Designing and Implementing a Dependency Management Strategy
- Designing a Release Strategy
- Implementing Continuous Deployment using Azure Pipelines
- Implementing an Appropriate Deployment Pattern
- Managing Infrastructure and Configuration using Azure Tools
- Third Party Infrastructure as Code Tools Available with Azure
- Managing Containers using Docker
- Creating and Managing Kubernetes Service Infrastructure
- Implementing Feedback for Development Teams
- Implementing System Feedback Mechanisms
- Implementing Security in DevOps Projects
- Validating Code Bases for Compliance

Module 1: Planning for DevOps

Lessons

- Transformation Planning
- Project Selection
- Team Structures
- Migrating to Azure DevOps

Lab : Agile Planning and Portfolio Management with Azure Boards

After completing this module, students will be able to:

- 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

Module 2: Getting Started with Source Control

Lessons

- What is Source Control
- Benefits of Source Control
- Types of Source Control Systems
- Introduction to Azure Repos
- Introduction to GitHub
- Migrating from Team Foundation Version Control (TFVC) to Git in Azure Repos

Lab : Version Controlling with Git in Azure Repos

After completing this module, students will be able to:

- Describe the benefits of using Source Control
- Describe Azure Repos and GitHub
- Migrate from TFVC to Git

Module 3: Managing Technical Debt

Lessons

- Identifying Technical Debt
- Knowledge Sharing within Teams
- Modernizing Development Environments with Codespaces

Lab : Sharing Team Knowledge using Azure Project Wikis

After completing this module, students will be able to:

- Manage code quality including technical debt SonarCloud, and other tooling solutions
- Build organizational knowledge on code quality

Module 4: Working with Git for Enterprise DevOps

Lessons

- How to Structure Your Git Repo
- Git Branching Workflows
- Collaborating with Pull Requests in Azure Repos
- Why Care About Git Hooks
- Fostering Inner Source
- Managing Git Repositories

Lab : Version Controlling with Git in Azure Repos

After completing this module, students will be able to:

- 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

Module 5: Configuring Azure Pipelines

Lessons

- The Concept of Pipelines in DevOps
- Azure Pipelines
- Evaluate use of Hosted versus Self-Hosted Agents
- Agent Pools
- Pipelines and Concurrency
- Azure DevOps and Open-Source Projects (Public Projects)
- Azure Pipelines YAML versus Visual Designer

Lab : Configuring Agent Pools and Understanding Pipeline Styles

After completing this module, students will be able to:

- Explain the role of Azure Pipelines and its components
- Configure Agents for use in Azure Pipelines

Module 6: Implementing Continuous Integration using Azure Pipelines

Lessons

- Continuous Integration Overview
- Implementing a Build Strategy

- Integration with Azure Pipelines
- Integrating External Source Control with Azure Pipelines
- Set Up Self-Hosted Agents

Lab : Enabling Continuous Integration with Azure Pipelines

Lab : Integrating External Source Control with Azure Pipelines

After completing this module, students will be able to:

- Explain why continuous integration matters
- Implement continuous integration using Azure Pipelines

Module 7: Managing Application Configuration and Secrets

Lessons

- Introduction to Security
- Implement a Secure Development Process
- Rethinking Application Configuration Data
- Manage Secrets, Tokens, and Certificates
- Integrating with Identity Management Systems
- Implementing Application Configuration

Lab : Integrating Azure Key Vault with Azure DevOps

After completing this module, students will be able to:

- Manage application configuration and secrets
- Integrate Azure Key Vault with a pipeline

Module 8: Implementing Continuous Integration with GitHub Actions

Lessons

- GitHub Actions
- Continuous Integration with GitHub Actions
- Securing Secrets for GitHub Actions

Lab : GitHub Actions Continuous Integration

After completing this module, students will be able to:

- Create and work with GitHub Actions and Workflows
- Implement Continuous Integration with GitHub Actions

Module 9: Designing and Implementing a Dependency Management Strategy

Lessons

- Packaging Dependencies
- Package Management
- Migrating and Consolidating Artifacts
- Package Security
- Implementing a Versioning Strategy

Lab : Package Management with Azure Artifacts

After completing this module, students will be able to:

- Recommend artifact management tools and practices
- Abstract common packages to enable sharing and reuse
- Migrate and consolidate artifacts
- Migrate and integrate source control measures

Module 10: Designing a Release Strategy

Lessons

- Introduction to Continuous Delivery
- Release Strategy Recommendations
- Building a High-Quality Release pipeline
- Choosing the Right Release Management Tool

Lab : Controlling Deployments using Release Gates

Lab : Creating a Release Dashboard

After completing this module, students will be able to:

- 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
- Choose a release management tool

Module 11: Implementing Continuous Deployment using Azure Pipelines

Lessons

- Create a Release Pipeline
- Provision and Configure Environments
- Manage and Modularize Tasks and Templates
- Configure Automated Integration and Functional Test Automation
- Automate Inspection of Health

Lab : Configuring Pipelines as Code with YAML

Lab : Setting up and Running Functional Tests

After completing this module, students will be able to:

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

Module 12: Implementing an Appropriate Deployment Pattern

Lessons

- Introduction to Deployment Patterns
- Implement Blue Green Deployment
- Feature Toggles
- Canary Releases
- Dark Launching
- AB Testing
- Progressive Exposure Deployment

Lab : Feature Flag Management with LaunchDarkly and Azure DevOps

After completing this module, students will be able to:

- Describe deployment patterns
- Implement Blue Green Deployment
- Implement Canary Release
- Implement Progressive Exposure Deployment

Module 13: Managing Infrastructure and Configuration using Azure Tools

Lessons

- Infrastructure as Code and Configuration Management
- Create Azure Resources using ARM Templates
- Create Azure Resources using Azure CLI
- Azure Automation with DevOps
- Desired State Configuration (DSC)

Lab : Azure Deployments using Resource Manager Templates

After completing this module, students will be able to:

- Apply infrastructure and configuration as code principles.
- Deploy and manage infrastructure using Microsoft automation technologies such as ARM templates, PowerShell, and Azure CLI

Module 14: Third Party Infrastructure as Code Tools Available with Azure

Lessons

- Chef
- Puppet
- Ansible
- Terraform

Lab : Automating Infrastructure Deployments in the Cloud with Terraform and Azure Pipelines

Lab : Deploying Apps with Chef on Azure

Lab : Deploy App with Puppet on Azure

Lab : Ansible with Azure

After completing this module, students will be able to:

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

Module 15: Managing Containers using Docker

Lessons

- Implementing a Container Build Strategy
- Implementing Docker Multi-Stage Builds

Lab : Modernizing Existing ASP.NET Apps with Azure

After completing this module, students will be able to:

- Implement a container strategy including how containers are different from virtual machines and how microservices use containers
- Implement containers using Docker
- Implement Docker multi-stage builds

Module 16: Creating and Managing Kubernetes Service Infrastructure

Lessons

- Azure Kubernetes Service
- Kubernetes Tooling
- Integrating AKS with Pipelines

Lab : Deploying a Multi-Container Application to Azure Kubernetes Service

After completing this module, students will be able to:

- Deploy and configure a Managed Kubernetes cluster

Module 17: Implementing Feedback for Development Teams

Lessons

- 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

Lab : Monitoring Application Performance with Application Insights

After completing this module, students will be able to:

- Implement tools to track system usage, feature usage, and flow
- Configure crash report integration for client applications
- Implement routing for client application crash report data
- Develop monitoring and status dashboards
- Integrate and configure ticketing systems with development team’s work management

Module 18: Implementing System Feedback Mechanisms

Lessons

- Site Reliability Engineering
- Design Practices to Measure End-User Satisfaction
- Design Processes to Capture and Analyze User Feedback
- Design Processes to Automate Application Analytics
- Managing Alerts
- Blameless Retrospectives and a Just Culture

Lab : Integration between Azure DevOps and Teams

After completing this module, students will be able to:

- 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

Module 19: Implementing Security in DevOps Projects

Lessons

- Security in the Pipeline
- Azure Security Center

Lab : Implement Security and Compliance in an Azure DevOps Pipeline

After completing this module, students will be able to:

- Define an infrastructure and configuration strategy and appropriate toolset for a release pipeline and application infrastructure
- Implement compliance and security in your application infrastructure

Module 20: Validating Code Bases for Compliance

Lessons

- Open-Source Software
- Managing Security and Compliance Policies
- Integrating License and Vulnerability Scans

Lab : Managing Technical Debt with SonarQube and Azure DevOps

After completing this module, students will be able to:

- 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