Microsoft Designing and Implementing Microsoft DevOps solutions

Duration: 4 Days **Course Code: M-AZ400** **Version: A**

Overview:

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

Target Audience:

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

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
- 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

Prerequisites:

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.

Testing and Certification

-

Content:

Module 1: Get started on a DevOps transformation journey

Module 1 Lessons

- Introduction to DevOps
- Choose the right project
- Describe team structures
- Choose the DevOps tools
- Plan Agile with GitHub Projects and Azure Boards
- Introduction to source control
- Describe types of source control systems
- Work with Azure Repos and GitHub

Lab 1: Agile planning and portfolio management with Azure Boards

Lab 2: Version controlling with Git in Azure Repos

After completing Module 1, students will be able to:

- Understand what DevOps is and the steps to accomplish it
- Identify teams to implement the process
- Plan for the transformation with shared goals and timelines
- Plan and define timelines for goals
- Understand different projects and systems to guide the journey
- Select a project to start the DevOps transformation
- Identify groups to minimize initial resistance
- Identify project metrics and Key Performance Indicators (KPI's)
- Understand agile practices and principles of agile development
- Create a team and agile organizational structure

Module 2: Development for enterprise DevOps

Module 2 Lessons

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

Lab 3: Version controlling with Git in Azure

will be able to:

- Describe Azure Pipelines
- Explain the role of Azure Pipelines and its components
- Decide Pipeline automation responsibility
- Understand Azure Pipeline key terms
- Choose between Microsoft-hosted and self-hosted agents
- Install and configure Azure pipelines Agents
- Configure agent pools
- Make the agents and pools secure
- Use and estimate parallel jobs
- Explain continuous delivery (CD)
- Implement continuous delivery in your development cycle
- Understand releases and deployment
- Identify project opportunities to apply CD
- Explain things to consider when designing your release strategy
- Define the components of a release pipeline and use artifact sources
- Create a release approval plan
- Implement release gates
- Differentiate between a release and a deployment
- 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
- Implement release jobs
- Differentiate between multi-agent and multi-configuration release job
- Provision and configure target environment
- Deploy to an environment securely using a service connection
- Configure functional test automation and run availability tests
- Setup test infrastructure
- Use and manage task and variable groups
- Understand how to deploy your environment
- Plan your environment configuration
- Choose between imperative versus declarative configuration
- Explain idempotent configuration
- Create Azure resources using ARM templates
- Understand ARM templates and template components
- Manage dependencies and secrets in templates
- Organize and modularize templates
- Create Azure resources using Azure CLI
- Identify SQL injection attack
- Understand DevSecOps
- Implement pipeline security
- Understand threat modeling

will be able to:

- Describe Azure Pipelines
- Explain the role of Azure Pipelines and its components
- Decide Pipeline automation responsibility
- Understand Azure Pipeline key terms
- Choose between Microsoft-hosted and self-hosted agents
- Install and configure Azure pipelines Agents
- Configure agent pools
- Make the agents and pools secure
- Use and estimate parallel jobs
- Explain continuous delivery (CD)
- Implement continuous delivery in your development cycle
- Understand releases and deployment
- Identify project opportunities to apply CD
- Explain things to consider when designing your release strategy
- Define the components of a release pipeline and use artifact sources
- Create a release approval plan
- Implement release gates
- Differentiate between a release and a deployment
- 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
- Implement release jobs
- Differentiate between multi-agent and multi-configuration release job
- Provision and configure target environment
- Deploy to an environment securely using a service connection
- Configure functional test automation and run availability tests
- Setup test infrastructure
- Use and manage task and variable groups
- Understand how to deploy your environment
- Plan your environment configuration
- Choose between imperative versus declarative configuration
- Explain idempotent configuration
- Create Azure resources using ARM templates
- Understand ARM templates and template components
- Manage dependencies and secrets in templates
- Organize and modularize templates
- Create Azure resources using Azure CLI
- Identify SQL injection attack
- Understand DevSecOps
- Implement pipeline security
- Understand threat modeling
- Implement open-source software
- Explain corporate concerns for

Repos

After completing Module 2, students will be able to:

- Understand Git repositories
- Implement mono repo or multiple repos
- Explain how to structure Git Repos
- Implement a change log
- Describe Git branching workflows
- Implement feature branches
- Implement GitFlow
- Fork a repo
- Leverage pull requests for collaboration and code reviews
- Give feedback using pull requests

Module 3: Implement CI with Azure Pipelines and GitHub Actions

Module 3 Lessons

- 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
- Design a container build strategy

Lab 4: Configuring agent pools and understanding pipeline styles

Lab 5: Enabling continuous integration with Azure Pipelines

Lab 6: Integrating external source control with Azure Pipelines

Lab 7: Implementing GitHub Actions by using DevOps Starter

Lab 8: Deploying Docker Containers to Azure App Service web apps

After completing Module 3, students

- Implement open-source software
- Explain corporate concerns for open-source components
- Describe open-source licenses
- Understand the license implications and ratings
- Work with Static and Dynamic Analyzers
- Configure Microsoft Defender for Cloud
- Define dependency management strategy
- Identify dependencies
- Describe elements and componentization of a dependency management
- Scan your codebase for dependencies
- Implement package management
- Manage package feed
- Consume and create packages
- Publish packages
- Identify artifact repositories
- Migrate and integrate artifact repositories

Module 4: Design and implement a release strategy

Module 4 Lessons

- Introduction to continuous delivery
- Create a release pipeline
- Explore release strategy recommendations
- Provision and test environments
- Manage and modularize tasks and templates
- Automate inspection of health

Lab 9: Creating a release dashboard

Lab 10: Controlling deployments using Release Gates

After completing Module 4, students

will be able to:

- Describe Azure Pipelines
- Explain the role of Azure Pipelines and its components
- Decide Pipeline automation responsibility
- Understand Azure Pipeline key terms
- Choose between Microsoft-hosted and self-hosted agents
- Install and configure Azure pipelines Agents
- Configure agent pools
- Make the agents and pools secure
- Use and estimate parallel jobs
- Explain continuous delivery (CD)
- Implement continuous delivery in your development cycle
- Understand releases and deployment
- Identify project opportunities to apply CD

open-source components

- Describe open-source licenses
- Understand the license implications and ratings
- Work with Static and Dynamic Analyzers
- Configure Microsoft Defender for Cloud
- Define dependency management strategy
- Identify dependencies
- Describe elements and componentization of a dependency management
- Scan your codebase for dependencies
- Implement package management
- Manage package feed
- Consume and create packages
- Publish packages
- Identify artifact repositories
- Migrate and integrate artifact repositories

Module 7: Implement security and validate code bases for compliance

Module 7 Lessons

- Introduction to Secure DevOps
- Implement open-source software
- Software Composition Analysis
- Static analyzers
- OWASP and Dynamic Analyzers
- Security Monitoring and Governance

Lab 15: Implement security and compliance in Azure Pipelines

Lab 16: Managing technical debt with SonarQube and Azure DevOps

After completing Module 7, students

will be able to:

- Describe Azure Pipelines
- Explain the role of Azure Pipelines and its components
- Decide Pipeline automation responsibility
- Understand Azure Pipeline key terms
- Choose between Microsoft-hosted and self-hosted agents
- Install and configure Azure pipelines Agents
- Configure agent pools
- Make the agents and pools secure
- Use and estimate parallel jobs
- Explain continuous delivery (CD)
- Implement continuous delivery in your development cycle
- Understand releases and deployment
- Identify project opportunities to apply CD
- Explain things to consider when designing your release strategy
- Define the components of a release pipeline and use artifact sources

- Explain things to consider when designing your release strategy
- Define the components of a release pipeline and use artifact sources
- Create a release approval plan
- Implement release gates
- Differentiate between a release and a deployment
- 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
- Implement release jobs
- Differentiate between multi-agent and multi-configuration release job
- Provision and configure target environment
- Deploy to an environment securely using a service connection
- Configure functional test automation and run availability tests
- Setup test infrastructure
- Use and manage task and variable groups
- Understand how to deploy your environment
- Plan your environment configuration
- Choose between imperative versus declarative configuration
- Explain idempotent configuration
- Create Azure resources using ARM templates
- Understand ARM templates and template components
- Manage dependencies and secrets in templates
- Organize and modularize templates
- Create Azure resources using Azure CLI
- Identify SQL injection attack
- Understand DevSecOps
- Implement pipeline security
- Understand threat modeling
- Implement open-source software
- Explain corporate concerns for open-source components
- Describe open-source licenses
- Understand the license implications and ratings
- Work with Static and Dynamic Analyzers
- Configure Microsoft Defender for Cloud
- Define dependency management strategy
- Identify dependencies
- Describe elements and componentization of a dependency management
- Scan your codebase for dependencies
- Implement package management
- Manage package feed
- Consume and create packages
- Publish packages
- Identify artifact repositories
- Migrate and integrate artifact repositories

- Create a release approval plan
- Implement release gates
- Differentiate between a release and a deployment
- 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
- Implement release jobs
- Differentiate between multi-agent and multi-configuration release job
- Provision and configure target environment
- Deploy to an environment securely using a service connection
- Configure functional test automation and run availability tests
- Setup test infrastructure
- Use and manage task and variable groups
- Understand how to deploy your environment
- Plan your environment configuration
- Choose between imperative versus declarative configuration
- Explain idempotent configuration
- Create Azure resources using ARM templates
- Understand ARM templates and template components
- Manage dependencies and secrets in templates
- Organize and modularize templates
- Create Azure resources using Azure CLI
- Identify SQL injection attack
- Understand DevSecOps
- Implement pipeline security
- Understand threat modeling
- Implement open-source software
- Explain corporate concerns for open-source components
- Describe open-source licenses
- Understand the license implications and ratings
- Work with Static and Dynamic Analyzers
- Configure Microsoft Defender for Cloud
- Define dependency management strategy
- Identify dependencies
- Describe elements and componentization of a dependency management
- Scan your codebase for dependencies
- Implement package management
- Manage package feed
- Consume and create packages
- Publish packages
- Identify artifact repositories
- Migrate and integrate artifact repositories

Module 8: Design and implement a dependency management strategy

Module 8 Lessons

- Explore package dependencies

Module 5: Implement a secure continuous deployment using Azure Pipelines

Module 5 Lessons

- 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
- Integrate with identity management systems
- Manage application configuration data

Lab 11: Configuring pipelines as code with YAML

Lab 12: Setting up and running functional tests

Lab 13: Integrating Azure Key Vault with Azure DevOps

After completing Module 5, students

will be able to:

- Describe Azure Pipelines
- Explain the role of Azure Pipelines and its components
- Decide Pipeline automation responsibility
- Understand Azure Pipeline key terms
- Choose between Microsoft-hosted and self-hosted agents
- Install and configure Azure pipelines Agents
- Configure agent pools
- Make the agents and pools secure
- Use and estimate parallel jobs
- Explain continuous delivery (CD)
- Implement continuous delivery in your development cycle
- Understand releases and deployment
- Identify project opportunities to apply CD
- Explain things to consider when designing your release strategy
- Define the components of a release pipeline and use artifact sources
- Create a release approval plan
- Implement release gates
- Differentiate between a release and a deployment
- 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
- Understand package management
- Migrate, consolidate, and secure artifacts
- Implement a versioning strategy
- Introduction to GitHub Packages

Lab 17: Package management with Azure Artifacts

After completing Module 8, students

will be able to:

- Describe Azure Pipelines
- Explain the role of Azure Pipelines and its components
- Decide Pipeline automation responsibility
- Understand Azure Pipeline key terms
- Choose between Microsoft-hosted and self-hosted agents
- Install and configure Azure pipelines Agents
- Configure agent pools
- Make the agents and pools secure
- Use and estimate parallel jobs
- Explain continuous delivery (CD)
- Implement continuous delivery in your development cycle
- Understand releases and deployment
- Identify project opportunities to apply CD
- Explain things to consider when designing your release strategy
- Define the components of a release pipeline and use artifact sources
- Create a release approval plan
- Implement release gates
- Differentiate between a release and a deployment
- 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
- Implement release jobs
- Differentiate between multi-agent and multi-configuration release job
- Provision and configure target environment
- Deploy to an environment securely using a service connection
- Configure functional test automation and run availability tests
- Setup test infrastructure
- Use and manage task and variable groups
- Understand how to deploy your environment
- Plan your environment configuration
- Choose between imperative versus declarative configuration
- Explain idempotent configuration
- Create Azure resources using ARM templates
- Understand ARM templates and template components

- Implement release jobs
- Differentiate between multi-agent and multi-configuration release job
- Provision and configure target environment
- Deploy to an environment securely using a service connection
- Configure functional test automation and run availability tests
- Setup test infrastructure
- Use and manage task and variable groups
- Understand how to deploy your environment
- Plan your environment configuration
- Choose between imperative versus declarative configuration
- Explain idempotent configuration
- Create Azure resources using ARM templates
- Understand ARM templates and template components
- Manage dependencies and secrets in templates
- Organize and modularize templates
- Create Azure resources using Azure CLI
- Identify SQL injection attack
- Understand DevSecOps
- Implement pipeline security
- Understand threat modeling
- Implement open-source software
- Explain corporate concerns for open-source components
- Describe open-source licenses
- Understand the license implications and ratings
- Work with Static and Dynamic Analyzers
- Configure Microsoft Defender for Cloud
- Define dependency management strategy
- Identify dependencies
- Describe elements and componentization of a dependency management
- Scan your codebase for dependencies
- Implement package management
- Manage package feed
- Consume and create packages
- Publish packages
- Identify artifact repositories
- Migrate and integrate artifact repositories

Module 6: Manage infrastructure as code using Azure and DSC

Module 6 Lessons

- 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
- Manage dependencies and secrets in templates
- Organize and modularize templates
- Create Azure resources using Azure CLI
- Identify SQL injection attack
- Understand DevSecOps
- Implement pipeline security
- Understand threat modeling
- Implement open-source software
- Explain corporate concerns for open-source components
- Describe open-source licenses
- Understand the license implications and ratings
- Work with Static and Dynamic Analyzers
- Configure Microsoft Defender for Cloud
- Define dependency management strategy
- Identify dependencies
- Describe elements and componentization of a dependency management
- Scan your codebase for dependencies
- Implement package management
- Manage package feed
- Consume and create packages
- Publish packages
- Identify artifact repositories
- Migrate and integrate artifact repositories

Module 9: Implement continuous feedback

Module 9 Lessons

- Implement tools to track usage and flow
- Develop monitor and status dashboards
- Share knowledge within teams
- Design processes to automate application analytics
- Manage alerts, Blameless retrospectives and a just culture

Lab 18: Monitoring application performance with Application Insights

Lab 19: Integration between Azure DevOps and Microsoft Teams

Lab 20: Sharing Team Knowledge using Azure Project Wikis

After completing Module 9, students will be able to:

- Implement tools to track feedback
- Plan for continuous monitoring
- Implement Application Insights
- Use Kusto Query Language (KQL)
- Implement routing for mobile applications
- Configure App Center Diagnostics
- Configure alerts
- Create a bug tracker
- Configure Azure Dashboards

(DSC)
- Implement Bicep

Lab 14: Azure deployments using Azure Resource Manager templates

After completing Module 6, students

- Work with View Designer in Azure Monitor

Further Information:

For More information, or to book your course, please call us on Head Office 01189 123456 / Northern Office 0113 242 5931

info@globalknowledge.co.uk

www.globalknowledge.com/en-gb/

Global Knowledge, Mulberry Business Park, Fishponds Road, Wokingham Berkshire RG41 2GY UK