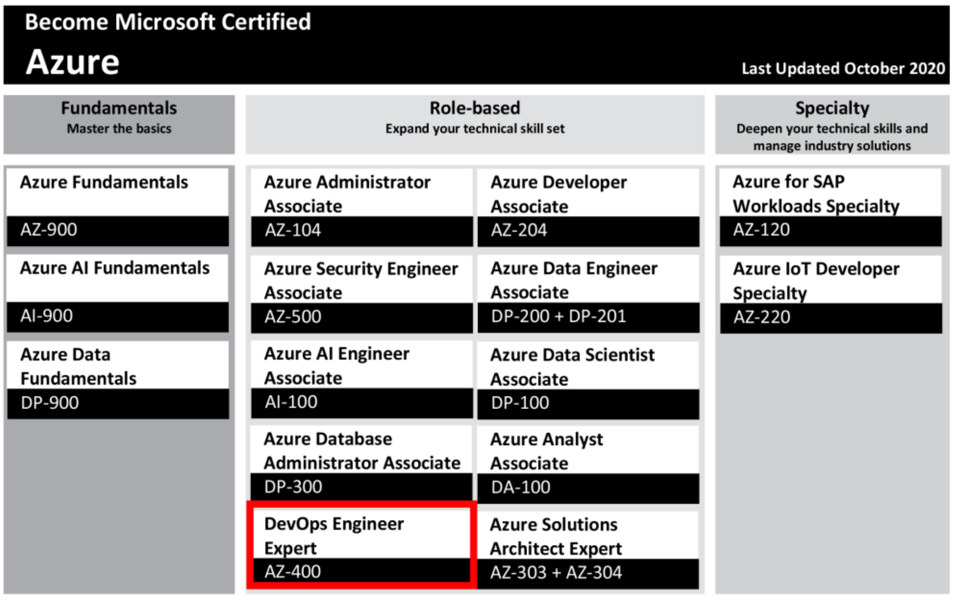
ISEIG
Institut Suisse d'Enseignement de l'Informatique de Gestion

Microsoft CERTIFIED
Partner
Learning Solutions

AZ-400 - Microsoft Certified: Azure DevOps Engineer Expert

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.

This course help prepare for the exam « AZ-400 – Designing and Implementing Microsoft DevOps Solutions » to obtain the title « Microsoft Certified: Azure DevOps Engineer Expert ».


Become Microsoft Certified
Azure
Last Updated October 2020

Fundamentals Master the basics	Role-based Expand your technical skill set		Specialty Deepen your technical skills and manage industry solutions
Azure Fundamentals AZ-900	Azure Administrator Associate AZ-104	Azure Developer Associate AZ-204	Azure for SAP Workloads Specialty AZ-120
Azure AI Fundamentals AI-900	Azure Security Engineer Associate AZ-500	Azure Data Engineer Associate DP-200 + DP-201	Azure IoT Developer Specialty AZ-220
Azure Data Fundamentals DP-900	Azure AI Engineer Associate AI-100	Azure Data Scientist Associate DP-100	
	Azure Database Administrator Associate DP-300	Azure Analyst Associate DA-100	
	DevOps Engineer Expert AZ-400	Azure Solutions Architect Expert AZ-303 + AZ-304	

Target Audience :

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

Objectives :

After completing this course, students will be able to:

- Plan for the transformation with shared goals and timelines
- Select a project and identify project metrics and KPIs
- Create a team and agile organization structure
- Describe the benefits of using Source Control
- Migrate from TFVC to Git
- Scale Git for Enterprise DevOps
- Recommend artifact management tools and practices
- Abstract common packages to enable sharing and reuse
- Migrate and consolidate artifacts
- Migrate and integrate source control measures
- Manage application config and secrets
- Develop a project quality strategy
- Plan for secure development practices and compliance rules
- Implement and manage build infrastructure

- Explain why continuous integration matters
- Implement continuous integration using Azure DevOps
- Manage code quality including: technical debt, SonarCloud, and other tooling solutions
- Manage security policies with open source, OWASP, and WhiteSource Bolt
- Implement a container strategy including how containers are different from virtual machines and how microservices use containers
- Implement containers using Docker
- 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
- 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
- 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
- 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
- Implement a mobile DevOps strategy
- 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 Azure
- Deploy 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
- 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

Prerequisites :

- Students should have 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.
- It is recommended that you have experience working in an IDE, as well as some knowledge of the Azure portal. However, students who may not have a technical background in these technologies, but who are curious about DevOps practices as a culture shift, should be able to follow the procedural and expository explanations of continuous integration regardless.
- Experience working in a software development or operations environment with either Windows or Linux would be helpful but is not essential.
- Students should also have knowledge of general application development and deployment processes.
- Also, it is recommended that you have experience working in an IDE, as well as some knowledge of the Azure portal. However, students who may not have a technical background in these technologies, but who are curious about DevOps practices as a culture shift, should be able to follow the procedural and expository explanations of continuous integration regardless.

Duration and Price

Courses - Modules	Jours	CHF	CHF/j
AZ-400 - Designing and Implementing Microsoft DevOps solutions	5	3'750.-	750.-

selon conditions générales. Le prix comprend toute la documentation distribuée.
Les cours se déroulent de 9 h 00 à 12 h 00 et 13 h 30 à 17 h 00

* Remise de 5 % au membre ADI, GRI, au diplômé(e) ISEIG CPF, titulaire du BFI, DFI ou certifié(e) MCSA, MCSE, MCSD formé(e) à l'ISEIG