training code: AZ-400 / ENG DL 4d / EN

Designing and Implementing Microsoft DevOps solutions

Authorized Microsoft Designing and Implementing Microsoft DevOps solutions **AZ-400** Distance Learning training.

Target audience:

- administrator
- developer
- DevOps engineer

Purpose of the training

The training is addressed to people interested in improving their knowledge and skills from DevOps processes, as well as people interested in taking Microsoft Azure DevOps Solutions certification exam.

The course is recommended to:

- developers
- administrators participating in projects related to developing new applications.

The course includes such issues as :

- making use of source code control, scaling Git for enterprise, as well as implementing and managin infrastructure.
- implementing continuous integration in Azure DevOps stream, managing code quality and security rules, as well as container building strategy.
- designing release strategy, configuring workflow of version management and implementing appropriate implementation pattern.
- designing strategy of dependency management, as well as security and compliance management.
- implementing infrastructure of code and infrastructure, providing Azure infrastructure using popular automation tools, implementing application infrastructure using various Azure platform services, as well as implementation methods, integrating external tools to implement Azure platform such as Chef and Puppet to involve compliance and security in release stream.
- designing system feedback mechanisms, implementing the process of conveying system opinons to

development teams and optimising feedback mechanisms.
- planning transformations, choosing the project and creating team structure, developing quality and security strategy, planning migration and consolidation of artefacts and source control

Benefits of completing the training

Gaining knowledge and practical skills from Office 365.
Including:

- Advantages of using source control, migration from TFVC to Git, scaling Git for Enterprise DevOps, implementing and managing compilation infrfastructure, application configuration management and sensitive data storage,DevOps mobile strategy
- Why continuous integration matters, implementing continuous integration using Azure DevOps, Configuring compilation and available options, creating automated workflows, integrating other tools for compilation with Azure DevOps platform, developing processes of hybrid code quality and methods of its measurment, detecting suspicious code, integrating automated tests of code quality, report related to code range during testing, tools to measure technical debt, detecting problems with Open Source licenses, as well as other licensing issues, implementing the strategy of building containers
- Differences between release and implementation, defining release stream components, about the things which you have to pay attention to when we design release strategies, classifying releases compared to the proces of releasing and controlling their quality, dealing with comments on release and documentation both in traditional and contemporary meaning, selecting release management tools, terminology used in Azure DevOps and other tools for release, understanding what Build and Release is, classifying the agent, queue and pool of agents, the necessity of several release tasks in one release stream, differentiating between release tasks for many agents and many configurations, the use of changeable versions and stage variables in release stream, remote implementation in environment using service connection, embedding testing in preparation, various ways of verifying the state of stream and release using alerts, service hooks and reports, creating release gateway, implementation patterns, implementing Blue Green deployment, Canary Release deployment, implementing progressive exposition
- Tools and practices of managing components, generalising packages which enable sharing and reuse, verifying base of codes to identify code dependencies which might be converted to packages, Identifying and recommending standard types and versions of packages in the whole solution, existing streams to build refactor to implement version strategy which publishes packages, security and compliance management, verifying open source software packages in terms of security and compliance with licenses to adjust tchem to corporate standards, configuing structuring stream to safely access package channels
- Using infrastructure and configuration as a code rule, implementing and managing infrastructure using Microsoft automation technology, such as ARM schemes, PowerShell and Azure command line

interafce, implementation models and services available in Azure, implementing and configuring Managed Kubernetes cluster, placing and configuring infrastructure using tools and services of other companies using Azure platforms, such as Chef, Puppet, Ansible, SaltStack and Terraform, infrastructure strategy and configuration, as well as appropriate tool sets for release stream and implementing compliance and security in application infrastructure

- Project practices to measure end-user satisfaction, designng processes to intercept and analyse users' opinions from external sources, routing design for data of Client's application failure report, recommended monitoring tools and technologies, suggested tools to track the use of system and functions, configuring integration failure reports for Client's applications, developing monitoring and state panels, routing implementation for data of Client's application failure, implementing tools to track the use of the system, the use of functions and flow, integration and configuration of ticketing systems with the development team work management system, analysing alerts to determine baseline, analysing telemetry to determine baseline, reviewing current websites nad intercepting opinions on system failures, performing constant tuning to diminish irrelevant or unfit for operation alerts
- Planning transformation with shared goals and timelines, choosing the project and specifying project indicators, as well as KPI indicators, creating the team and effective organisational structure, designing project quality strategy, planning secure development practice and compliance rule, migration and artefact consolidation, migrating and integrating source control means

Examination method

The exam is on-line. You can enroll at: https://home.pearsonvue.com/Clients/Microsoft.aspx

Exam description

After the **AZ-400** course, you can take **Microsoft** certification **exams**:an Authorized Test Center,online being monitored by an offsite proctor. Details on the website:
https://docs.microsoft.com/en-us/learn/certifications/exams/az-400

Expected Listener Preparation

- Fundamental knowledge about Azure platform, version control, rozwoju oprogramowania Agile software development and basic software development rules. Experience in organisations providing software is strongly recommended

- Experience in IDE environment as well as certain knowlege on Azure portal are advised
- An ability to use materials in English
- previous courses: AZ-104, AA_10961

To make work more convenient and training more effective we suggest using additional screen. Lack of extra screen does not make it impossible to participate in the training, but significantly influences the convenience of work during classes

Information and requirements conerning participation in distance learning trainings is available at: https://www.altkomakademia.pl/distance-learning/#FAQ

Training Language

- **Training**: English
- **Materials**: English

Training Includes

* electronic handbook available at: https://learn.microsoft.com/pl-pl/training/
* access to Altkom Akademia student portal

Duration

4 days / 29 hours

Training agenda

1. Implementing DevOps Development Processes
2. First steps with source control
   - What is source control?
   - Benefits from source control

- Types of source control systems
- Azure Repos
- Migration from TFVC to Git
- Authentication to Git repository

3. Scaling git for enterprise DevOps
   - How to organise git repository
   - Branched Git workflows
   - Cooperation with downloaded conclusions
   - Why should we care about GitHooks?
   - Supporting internal open source
   - Git version
   - Public projects
   - Files in Git
4. Implementing and managing compilation infrastructure
   - Stream concept in DevOps
   - Streams in Azure
   - Evaluating the use of hosted private agents
   - Pools of agents
   - Stream and concurrence
   - Azure DevOps and Open Source projects
   - Azure Pipelines YAML vs Visual Designer
   - Configuring private agents
   - Integrating Jenkins with Azure Pipelines
   - Integrating external navigation source with Azure Pipelines
   - Analysing and integrating multi-degree Docker compilation
5. Managing application configuration and ins and outs
   - Introduction to security
   - Implementing secure and compliant development process
   - Reanalysing application configuration data
   - Managing ins and outs, tokens and certificates
   - Implementing security management tools and compliance in preparation
6. Implementing DevOps mobile strategy
   - Introduction to Mobile DevOps
   - Introduction to Visual Studio application center
   - Managing mobile sets of target tools and distribution groups
   - Managing target sets of devices testing user’s interface
   - Testing devices for implementation
   - Creating public and private distribution groups

7. Implementing Continuous Integration
8. Implementing continuous integration in Azure DevOps stream
    - Discussing continuous integration
    - Implementing creation strategy
9. Code quality management and security strategies
    - Code quality management
    - Security rules management
10. Implementing container creation strategy
    - Implementing container creation strategy
11. Implementing Continuous Delivery
12. Release strategy project
    - Introduction to continuous delivery
    - Suggestions concerning release strategy
    - Building high quality stream
    - Selecting implementation pattern
    - Selecting appropriate tool for version management
13. Configuring version management workflow
    - Creating release stream
    - Delivering and Configuring environments
    - Managing and modularisation of tasks and schemes
    - Integrating ins and outs with release stream
    - Configuring automated integration and functional test automation
    - Automating condition of inspections
14. Implementing appropriate pattern of implementation
    - Introduction to implementation patterns
    - Implementing Blue Green deployment
    - Function switches
    - Canary Release
    - Dark Launching
    - AB testing
    - Progressive exposition
15. Implementing Dependency Management
16. Designing dependency management strategy
    - Introduction
    - Dependency packaging
    - Package management
    - Versioning strategy introduction
17. Security and compliance management

- Introduction
- Securing the package
- Open source software
- Scanning licenses and gaps in securities integration
- Verifying open source software packages in terms of security and compatibility with license to adjust tchem to corporate standards
- Configuring structure stream to gain access to security package and license evaluation
- Configuring secure access to package channels

18. Implementing Application Infrastructure
19. Infrastructue and configuration of Azure tool
    - Infrastructure as a code and configuration management
    - Creating Azure Resources using ARM templates
    - Creating Azure platform resources using Azure command line interface
    - Creating Azure resources using Azure PowerShell
    - Additional automation tools
    - Version control
20. Models and Azure implementation services
    - Models and implementation options
    - Azure Infrastructure-as-a-Service (IaaS)
    - Azure Automation with DevOps
    - Demanded State Configuration (DSC)
    - Usługi Azure Platform-as-a-Service (PaaS)
    - Azure Service Fabric
21. Creating and managing Kubernetes service infrastructureKubernetes
    - Azure Kubernetes service
22. Other company tools and Open Source tools available from Azure
    - Chef
    - Puppet
    - Ansible
    - Cloud-Init
    - Terraform
23. Implementing compatibility and security in its infrastructure
    - Rules of security and compatibility with DevOps
    - Azure security center
24. Course summary
25. Implementing Continuous Feedback
26. Recommended and designed system feedback mechanisms
    - Inner loop

- Constant empirical experiment
- Project practices to measure end-user satisfaction
- Designing processes to intercept and analyse users' opinions
- Designing process to automate application analysis

27. Implementing processes of conveying system opinions to programming teams
    - Implementing tools to track the use of the system
    - Routing implementation for data of mobile application failure report
    - Developing monitoring and state dashboards
    - Integrating and configuring ticketing systems
28. Feedback mechanisms optimisation
    - Website reliability engineering
    - Telemetry analysis to specify baseline
    - Performing constant tuning to diminish irrelevant or unfit for operation alerts
    - Alert analysis to specify baseline
    - Flameless postmortems and decent culture
29. Designing a DevOps Strategy
30. DevOps planning
    - Planning transformation
    - Project selection
    - Team structures
31. Quality and security planning
    - Planning quality strategy
    - Planning secure development
32. Migrating and consolidating artefacts and tools
    - Migrating and consolidating artefacts
    - Migrating and integrating source control