Maslak Mahallesi, Maslak Meydan Sk. No:5. Spring Giz Plaza. Maslak/İstanbul
+90 212 282 7700 - info@bilginc.com

Microsoft Azure DevOps Solutions

Learn via: **Classroom / Virtual Classroom / Online**

Duration: **5 Gün**

Overview

Bu {*eğitim} şunları sunmaktadır:

- DevOps proseslerini uygulamaya koymak için gereken bilgi ve beceriler. Katılımcılar, kaynak kontrolünü nasıl kullanacaklarını, işletmeler için Git'i nasıl ölçeklendireceklerini ve bir altyapıyı nasıl uygulamaya koyacaklarını ve yöneteceklerini öğreneceklerdir.
- Kesintisiz entegrasyonun DevOps uygulamalarını uygulamaya koymak için gereken bilgi ve beceriler. Katılımcılar, bir Azure DevOps ortamında kesintisiz entegrasyonu nasıl uygulamaya koyacaklarını, kod kalitesini ve güvenlik ilkelerini nasıl yöneteceklerini ve bir konteyner oluşturma stratejisini nasıl uygulamaya koyacaklarını öğreneceklerdir.
- Kesintisiz sunumu uygulamaya koymak için gereken bilgi ve beceriler. Katılımcılar, bir sürüm stratejisini nasıl tasarlayacaklarını, bir sürüm yönetimi iş akışını nasıl kuracaklarını ve uygun bir kurulum düzenini nasıl hayata geçireceklerini öğreneceklerdir.
- Bağımlılık yönetimini uygulamaya koymak için gereken bilgi ve beceriler. Katılımcılar bir bağımlılık yönetimi stretjisini nasıl tasarlayacaklarını ve güvenlik ve uyumu nasıl yöneteceklerini öğreneceklerdir.
- DevOps ortamlarında uygulama altyapısını kurmak için gereken bilgi ve beceriler. Katılımcılar, altyapısı kod ve yapılandırma yönetimi olarak nasıl uygulamaya koyacaklarını, ortak otomasyon araçlarını kullanarak Azure altyapısını nasıl geliştireceklerini ve çeşitli Azure servislerini ve kurulum metodolojilerini kullanarak bir uygulama altyapısını nasıl kuracaklarını öğreneceklerdir. Katılımcılar ayrıca uyum ve güvenliği bu sürüm planlarına dahil etmek için Chef ve Puppet gibi üçüncü şahıs kurulum araçlarını Azure ile nasıl entegre edeceklerini de öğreneceklerdir.
- Kesintisiz geri bildirimi uygulamaya koymak için gereken bilgi ve beceriler. Katılımcılar sistem geri bildirim mekanizmalarını nasıl önerip tasarlayacaklarını, sistem geri bildirimini geliştirme ekiplerine yönlendirmeye yönelik bir süreci nasıl uygulamaya koyacaklarını ve geri bildirim mekanizmalarını nasıl en verimli hale getireceklerini öğreneceklerdir.
- DevOps stratejisini tasarlamak koymak için gereken bilgi ve beceriler. Katılımcılar dönüşüm için planlamayı nasıl yapacaklarını, proje nasıl seçeceklerini ve ekip yapılarını nasıl oluşturacaklarını öğreneceklerdir. Katılımcılar ayrıca kalite ve güvenlik stratejilerini nasıl geliştireceklerini de öğrenecekler ve nesnelerin taşınması ve konsolidasyonunun ve kaynak kontrolünün planlanması da kapsama dahil olacaktır.

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.

Who Should Attend

Bu {*eğitim} katılmak isteyenler, DevOps süreçlerinin uygulamaya konulmasına veya Microsoft Azure DevOps Çözümleri sertifikasyon sınavına katılmaya ilgi gösteren kişilerdir.

What You Will Learn

After completing this course, students will be able to:

- Describe the benefits of using source control
- Migrate from TFVC to Git
- Scale Git for Enterprise DevOps
- Implement and manage build infrastructure
- Manage application config & secrets
- Implement a mobile DevOps strategy
- Explain why continuous integration matters
- Implement continuous integration using Azure DevOps
- Configure builds and the options available
- Create an automated build workflow
- Integrate other build tooling with Azure DevOps

- Create hybrid build processes
- Describe what is meant by code quality and how it is measured
- Detect code smells
- Integrate automated tests for code quality
- Report on code coverage during testing
- Add tooling to measure technical debt
- Detect open source and other licensing issues
- Implement a container build strategy
- 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
- Recommend artifact management tools and practices
- Abstract common packages to enable sharing and reuse
- 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
- 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
- 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
- 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 system
- 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
- Plan for the transformation with shared goals and timelines.
- Select a project and identify project metrics and KPIs.
- Create a team and agile organizational structure.
- Develop a project quality strategy.
- Plan for secure development practices and compliance rules.
- Migrate and consolidate artifacts.
- Migrate and integrate source control measures.

Outline

Module 1: Getting started with Source Control

Lessons

-
- What is Source Control?
- Benefits of Source Control
- Types of source control systems
- Introduction to Azure Repos
- Migrating from TFVC to Git
- Authenticating to your Git Repos

Module 2: Scaling git for enterprise DevOps

Lessons

- How to structure your git repo
- Git Branching workflows
- Collaborating with Pull Requests
- Why care about GitHooks?
- Fostering Internal Open Source
- Git Version
- Public projects
- Files in Git

Module 3: Implement & Manage Build Infrastructure

Lessons

- The concept of pipelines in DevOps
- Azure Pipelines
- Evaluate use of Hosted vs Private Agents
- Agent pools
- Pipelines & Concurrency
- Azure DevOps and Open Source projects
- Azure Pipelines YAML vs Visual Designer
- Setup private agents
- Integrate Jenkins with Azure Pipelines
- Integration external source control with Azure Pipelines
- Analyze & Integrate Docker multi-stage builds

Module 4: Managing application config & secrets

Lessons

- Introduction to Security
- Implement secure & compliant development process
- Rethinking application config data
- Manage secrets, tokens & certificates
- Implement tools for managing security and compliance in a pipeline

Module 5: Implement a mobile DevOps strategy

Lessons

- Introduction to Mobile DevOps
- Introduction to Visual Studio App Center
- Manage mobile target device sets and distribution groups
- Manage target UI test device sets
- Provision tester devices for deployment
- Create public and private distribution groups

Module 6: Implementing Continuous Integration in an Azure DevOps Pipeline

In this module, you'll be introduced to continuous integration principles including: benefits, challenges, build best practices, and implementation steps. You will also learn about implementing a build strategy with workflows, triggers, agents, and tools.

Lessons

- Continuous Integration Overview
- Implementing a Build Strategy

Lab : Enabling Continuous Integration with Azure Pipelines

Lab : Creating a Jenkins Build Job and Triggering CI

Module 7: Managing Code Quality and Security Policies

In this module, you will be learn how to manage code quality including: technical debt, SonarCloud, and other tooling solutions. You will also learn how to manage security policies with open source, OWASP, and WhiteSource Bolt.

Lessons

- Managing Code Quality
- Managing Security Policies

Lab : Managing Technical Debt with Azure DevOps and SonarCloud

Lab : Checking Vulnerabilities using WhiteSource Bolt and Azure DevOps

Module 8: Implementing a Container Build Strategy

In this module, you will learn how to implement a container strategy including how containers are different from virtual machines and how microservices use containers. You will also learn how to implement containers using Docker.

Lessons

- Implementing a Container Build Strategy

Lab : Existing .NET Applications with Azure and Docker Images

Module 9: Design a Release Strategy

Lessons

- Introduction to Continuous Delivery
- Release strategy recommendations
- Building a High Quality Release pipeline
- Choosing a deployment pattern
- Choosing the right release management tool

Lab : Building a release strategy

- 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

Module 10: Set up a Release Management Workflow

Lessons

- Create a Release Pipeline
- Provision and Configure Environments
- Manage And Modularize Tasks and Templates
- Integrate Secrets with the release pipeline
- Configure Automated Integration and Functional Test Automation
- Automate Inspection of Health

Lab : Automating your infrastructure deployments in the Cloud with Terraform and Azure Pipelines
Lab : Setting up secrets in the pipeline with Azure Key vault

Lab : Setting up and Running Load Tests
Lab : Setting up and Running Functional Tests
Lab : Using Azure Monitor as release gate
Lab : Creating a Release Dashboard

- 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

Module 11: Implement an appropriate deployment pattern

Lessons

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

Lab : Blue-Green Deployments
Lab : Traffic Manager

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

Module 12: Designing a Dependency Management Strategy

Lessons

- Introduction
- Packaging dependencies
- Package management
- Implement a versioning strategy

Lab : Updating packages

- Recommend artifact management tools and practices
- Abstract common packages to enable sharing and reuse
- 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

Module 13: Manage security and compliance

Lessons

-
- Introduction
- Package security
- Open source software
- Integrating license and vulnerability scans
- 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

Module 14: Infrastructure and Configuration Azure Tools

Lessons

- Learning Objectives
- Infrastructure as Code and Configuration Management
- Create Azure REsources using ARM Templates
- Create Azure Resources using Azure CLI
- Create Azure Resources by using Azure PowerShell
- Additional Automation Tools
- Version Control
- Lab Deploy to Azure using ARM templates
- Module Review Questions

Module 15: Azure Deployment Models and Services

Lessons

-
- Learning Objectives
- Deployment Models and Options
- Azure Infrastructure-as-a-Service (IaaS) Services
- Azure Automation with DevOps
- Desired State Configuration (DSC)
- Azure Platform-as-a-Service (PaaS) services
- Azure Service Fabric
- Lab Azure Automation - IaaS or PaaS deployment
- Moduel Review Questions

Module 16: Create and Manage Kubernetes Service Infrastructure

Lessons

-
- Learning Objectives
- Azure Kubernetes Service
- Lab Deploy and Scale AKS Cluster
- Module Review Questions

Module 17: Third Party and Open Source Tools available with Azure

Lessons

- Learning Objectives
- Chef
- Puppet
- Ansible
- Cloud-Init
- Terraform
- Lab Provision and configure an App in Azure Using X
- Module Review Questions

Module 18: Implement Compliance and Security in your Infrastructure

Lessons

- Security and Compliance Principles with DevOps
- Azure Security Center
- Lab Integrate a scanning extension or tool in an AZ DevOps pipeline/security center
- Module Review Questions

Module 19: Recommend and design system feedback mechanisms

Lessons

- The inner loop
- Continuous Experimentation mindset
- Design practices to measure end-user satisfaction

Module 20: Planning for DevOps

In this module, students will learn about transformation planning, project selection, and team structures.

Lessons

- Transformation Planning
- Project Selection
- Team Structures

Lab : Agile Planning and Portfolio Management with Azure Boards

Module 21: Planning for Quality and Security
In this module, students will learn about developing a quality strategy and planning for secure development.

Lessons

- Planning a Quality Strategy
- Planning Secure Development

Lab : Feature Flag Management with LaunchDarkly and AzureDevOps

Module 22: Migrating and Consolidating Artifacts and Tools

In this module, students will learn about migrating and consolidating artifacts, and migrating and integrating source control measures.

Lessons

- Migrating and Consolidating Artifacts
- Migrating and Integrating Source Control

Lab : Integrating Azure Repos and Azure Pipelines with Eclipse

- Design processes to capture and analyze user feedback
- Design process to automate application analytics

Lab : Integration between Azure DevOps and Teams
Lab : Feature Flags

- 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

Module 23: Implement process for routing system feedback to 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

Module 24: Optimize feedback mechanisms

Lessons

- Site Reliability Engineering
- Analyze telemetry to establish a baseline
- Perform ongoing tuning to reduce meaningless or non-actionable alerts

- Analyze alerts to establish a baseline
- Blameless PostMortems and a Just Culture