

CI/CD Pipeline Süreçleri ve Uygulaması

CI/CD Pipeline Süreçleri ve Uygulaması

CI/CD Nedir?

Azure Nedir?

GitHub Nedir?

Örnek Proje İncelemesi

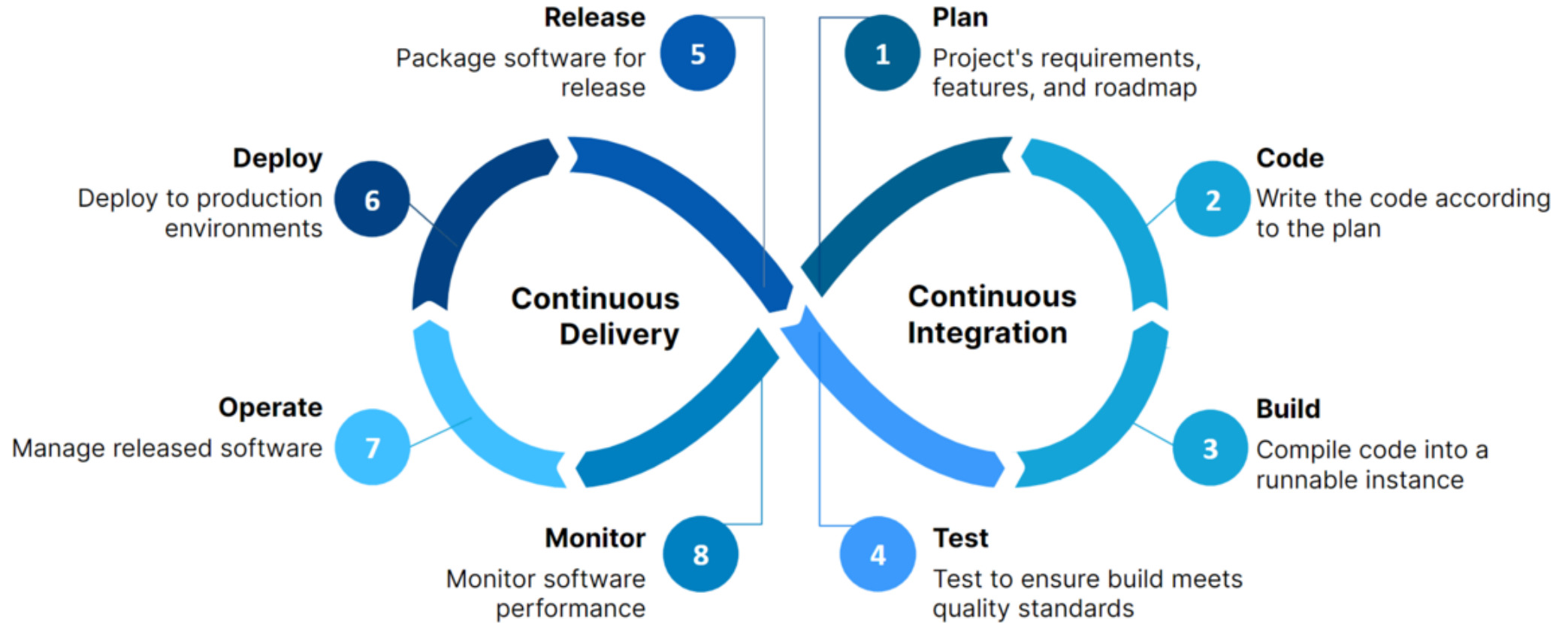
GitHub Action & Azure App Service ile CI/CD Uygulaması



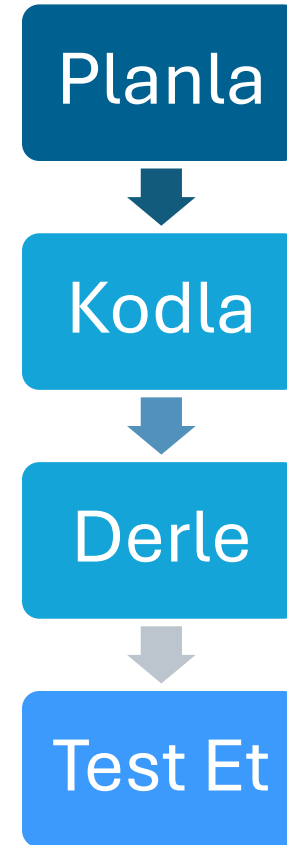
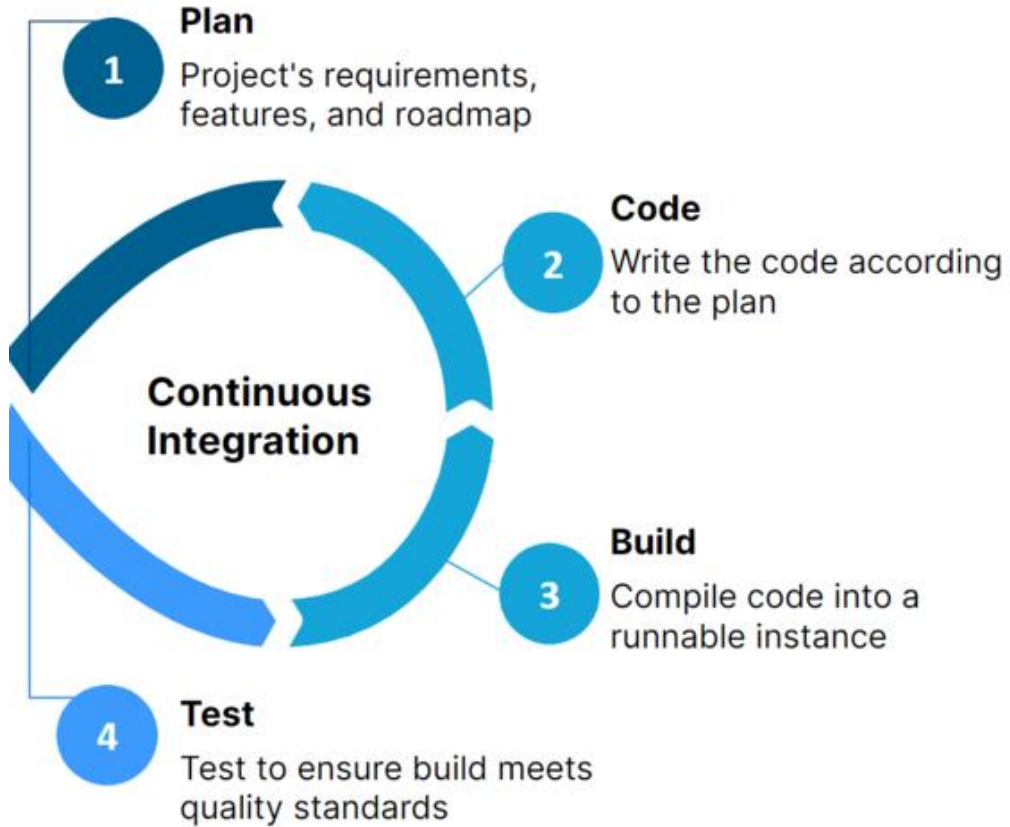
Kadir Berat GÜVENTÜRK

LEAD .NET DEVELOPER & PROJECT
MANAGER AT PİKSEL AKADEMİ

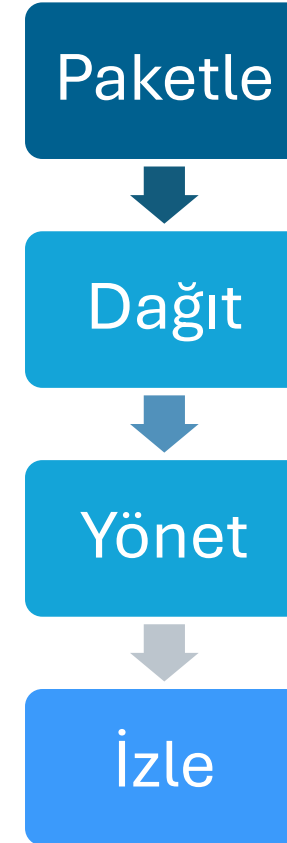
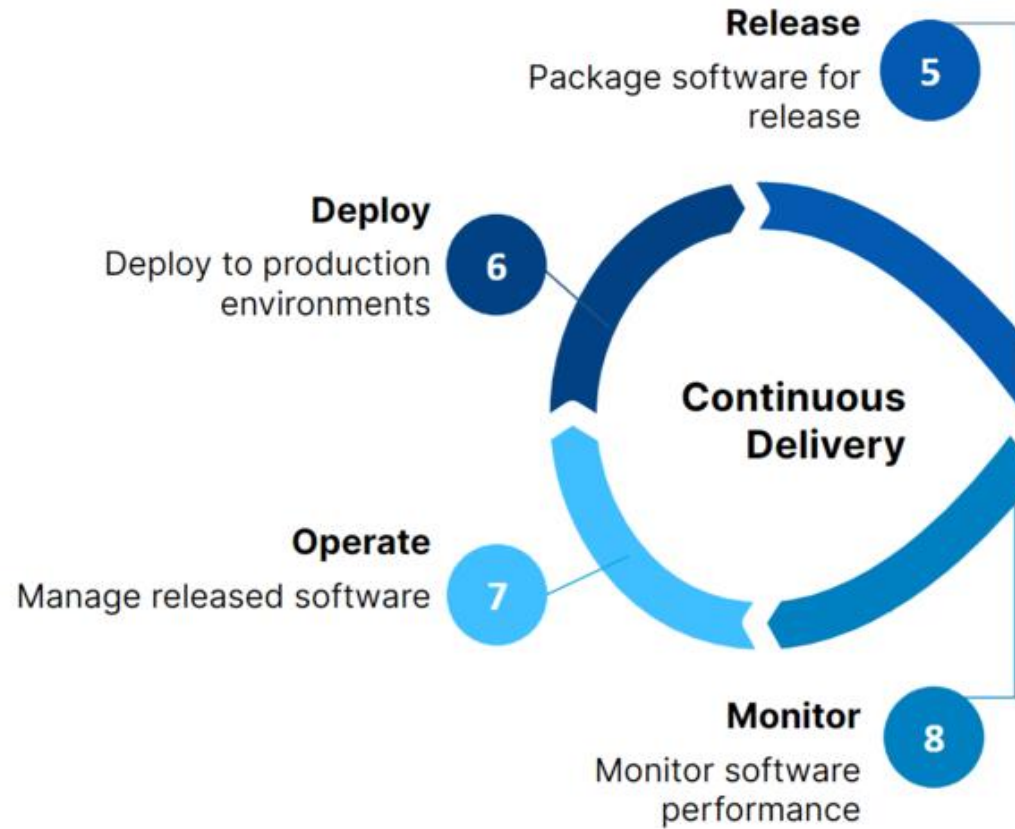
CI/CD Nedir?



CI Nedir?



CD Nedir?



Bazı CI/CD Araçları



Azure DevOps



GitHub Actions



Jenkins



GitLab



Bamboo



circleci



harness



TEKTON



Argo CD



Buildkite



Spinnaker



TeamCity



Travis CI



CODESHIP



go[®]



codefresh

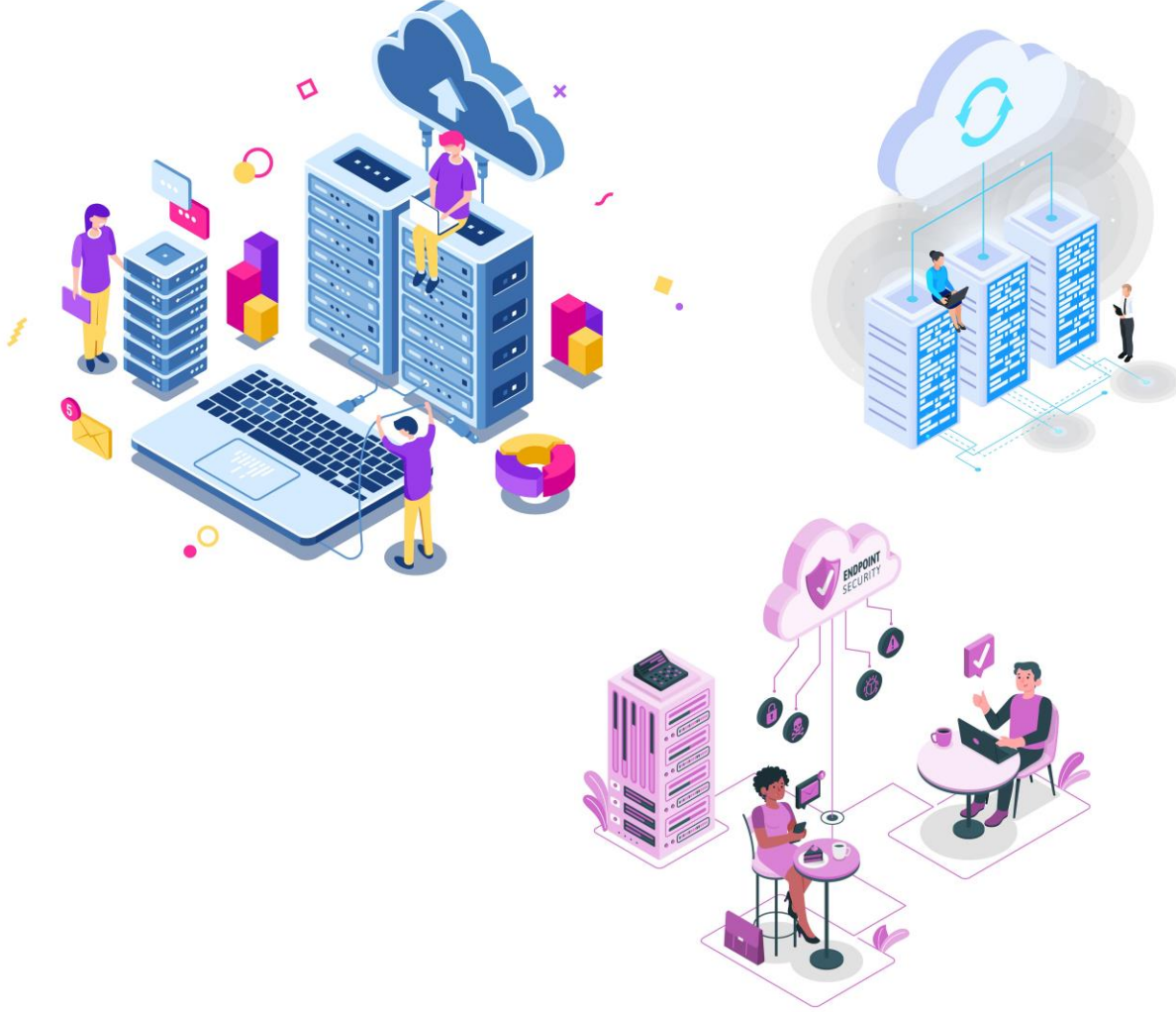
Gerçek Hayatta CI/CD

-
- PR'da otomatik build + test
 - Staging/Preview ortamlarda doğrulama
 - Prod dağıtım (slot / blue-green / canary)
 - DB migration ve config yönetimi
 - NuGet / Docker paket yayınlama
 - Monitoring + rollback ile güvenli geri dönüş

CI/CD Q&A

Microsoft Azure

Azure Nedir?



Azure Virtual Machines

-
- Otomatik ölçeklendirme
 - Birden çok işletim sistemi desteği
 - Yerleşik izleme ve yönetim
 - Hızlı yedekleme ve geri yükleme
 - Hızlandırılmış performans
 - Yapay zeka ve yüksek performanslı bilgi işlem (HPC)

Azure Dedicated Host

-
- Azure VM'lerinizi çalıştıran sunucu altyapısı üzerinde denetim ve görünürlük sağlar
 - İş yüklerinizi ayrılmış bir sunucuda dağıtarak uyumluluk gereksinimlerinin karşılanmasına yardımcı olur
 - İhtiyacınız olan işlemci sayısı, VM serileri ve VM boyutlarıyla Ayrılmış Konak SKU'larına sahiptir
 - Windows Server ve SQL Server için yalnızca Azure'da yararlanabileceğiniz fiyatlar ve avantajlar sunar

Azure App Service

- Dağıtımı ve ölçeklendirmeyi otomatikleştirin
 - Yüksek kullanılabilirlik ve dayanıklılık elde edin
 - Sıfır Güven ilkelerini benimseyin
 - Tercih ettiğiniz dillerde geliştirin
 - Azure App Service üzerinde WordPress siteleri oluşturun
 - Tek kiracılı App Service Ortamları ile uygulamaları güvenli şekilde çalıştırın
-

Azure Container Apps

-
- Uygulamaları isteğe bağlı olarak ölçeklendirin
 - Gelişmiş ağ iletişimi
 - Gelişmiş güvenlik ve idare
 - Açık kaynak temel ve eklentiler
 - İsteddiğiniz yerde dağıtın

Azure Kubernetes Service

-
- Basitleştirilmiş Kubernetes deneyimi
 - Bütünleşik izleme ve günlük kaydı
 - Gelişmiş güvenlik ve idare denetimleri
 - Seçki olarak sunulan koddan buluta deneyimi
 - Buluttan uca dağıtımlar
 - Güvenli kapsayıcı tedarik zincirleri
 - Gelişmiş konteyner ağ mimarisi
 - Çok kümeli yönetim

Azure Arc

- Tek merkezden yönetim
 - On-prem & multi-cloud desteği
 - Policy & governance
 - Güvenlik entegrasyonu
 - Envanter & izleme
 - Güncelleme/konfig yönetimi
 - Arc-enabled Kubernetes
 - Arc-enabled data services
-

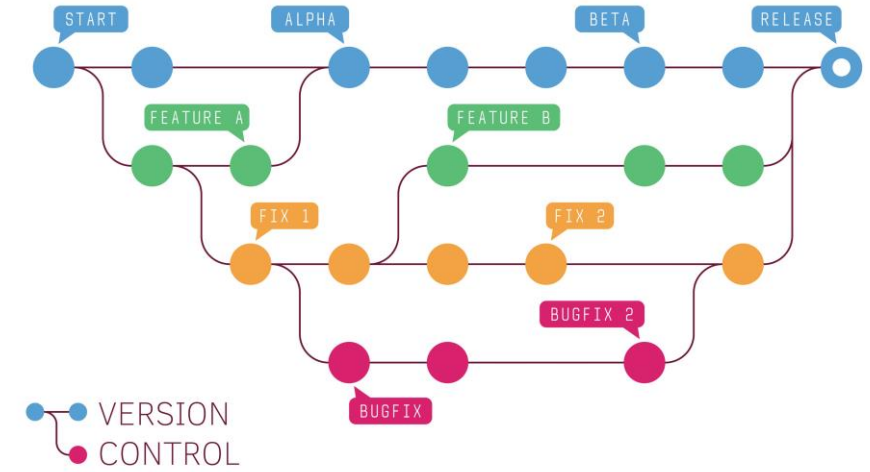
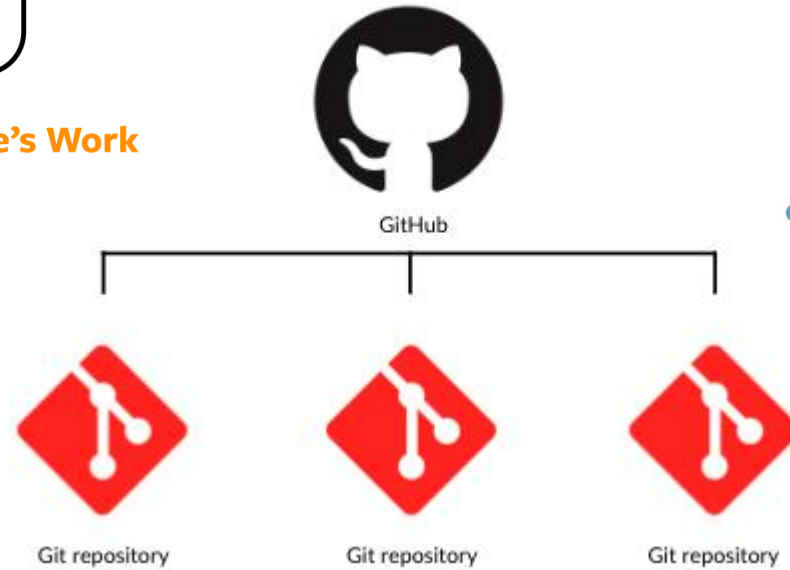
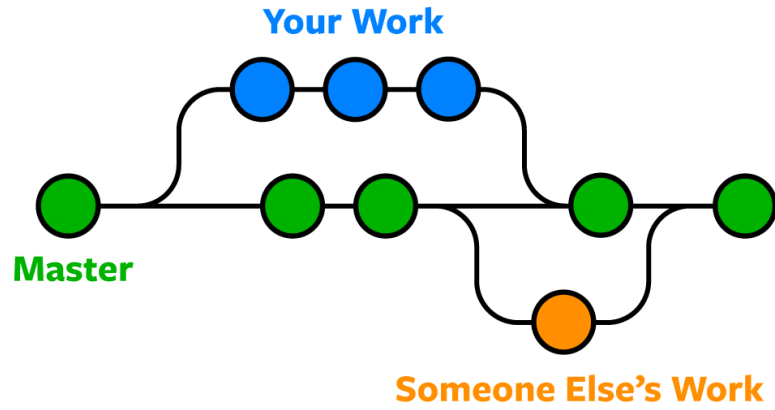
Azure DevOps

-
- Azure Repos
 - Azure Boards
 - Azure Artifacts
 - Azure Pipelines
 - Azure Test Plans

Azure Q&A

GitHub

GitHub Nedir?



GitHub Repositories

-
- Sürüm Geçmişi
 - Dal Yapısı
 - Çekme İstekleri
 - Hata Takibi
 - Dokümantasyon
 - Sürüm Yayınlama
 - Pages
 - Wikis

GitHub Projects

-
- Board/Kanban
 - Görevler
 - Yapılacaklar Listesi
 - Planlama
 - Durum Takibi
 - Sorumlu Atama
 - Etiketler & Öncelikler
 - Farklı Sayfa Tasarımları
 - Otomasyon

GitHub Actions

- İş Akışı
 - Tetikleyiciler
 - İşler
 - Adımlar
 - Konak Makine
 - Hazır Paketler
 - Gizli Bilgiler
 - Çıktılar
-

GitHub Q&A

CI/CD Örnek Proje

Proje Gereksinimleri

-
- Visual Studio 2026
 - .NET 10
 - GitHub hesabı (Repository ve Actions için)
 - Microsoft ve Azure hesapları (Azure App Service için)

Proje Tanıtımı

-
- Yönetim Panelli Web Sitesi
 - In-Memory veya MSSSQL database tercihi.
 - Yönetim panelinin Area içerisinde veya harici bir web uygulaması olarak yer aldığı iki farklı senaryo.
 - Panel üzerinden düzenlenebilen sayfa metinleri.
 - Projeler için temel entegrasyon (smoke) testleri.

Proje Yapısı

-
- 1. Senaryo: Solution içerisinde tek bir web uygulamasının bulunması.
 - 2. Senaryo: Solution içerisinde birden çok web uygulamasının bulunması.



GitHub Actions

Workflow Yapısı

-
- **name:** Workflow'un görünen adı.
 - **on:** Workflow'un **ne zaman** çalışacağını belirler.
 - **env:** Workflow genelinde kullanılan ortam değişkenleri.
 - **permission:** Workflow için **GitHub Token** yetkilerini belirler.
 - **concurrency:** Workflow için çakışmaları önlemek adına eşzamanlılık kontrolü.

Workflow Yapısı

-
- **jobs:** Workflow içindeki **işlerin** listesi.
 - **runs-on:** Job'un çalışacağı runner'ı seçer.
 - **steps:** Job'un içinde sırayla çalışan adımlar.
 - **uses:** Hazır action'ları kullanmamızı sağlar.
 - **run:** Terminal komutu çalıştırır.
 - **with:** uses ile kullanılarak action'a parametre geçirmek için kullanılır.
 - **if:** job veya step'leri koşullu çalıştırmamızı sağlar.

Workflow Yapısı

-
- **needs:** Bir job'un başka bir job bitmeden başlamasını engeller.
 - **strategy+matrix:** Aynı job'u farklı kombinasyonlarda çalıştırmaya veya tekrar eden yapıda çalıştırmaya yarar.
 - **secrets:** Gizli değerlerle çalışmamızı sağlar.
 - **artifacts:** Buil çıktısı gibi dosyaları job'lar veya action'lar arası taşımamızı sağlayan paketler.
 - **environments:** Projenin deploy ortamları.

Tetikleme Stratejileri (on)

-
- **push**: Bir branch'e push olunca çalışır.
 - **pull_request**: PR açılınca/güncellenince/merge edilince çalışır.
 - **workflow_dispatch**: Manuel olarak çalıştırılır.
 - **schedule**: Zamanlanmış olarak çalışır.
 - **release**: Release yapılıncaya çalışır.
 - **workflow_run**: Başka bir workflow bitince çalışır.
 - **workflow_call**: Başka bir workflow tarafından çağırılarak çalışır.

Push Filtreleri

-
- **branches:** Sadece belirli branch'lerde çalıştır.
 - **branches-ignore:** Bazı branch'leri hariç tut.
 - **tags:** Tag push'larında çalıştır.
 - **tags-ignore:** Bazı tag'ları hariç tut.
 - **paths:** Sadece belirli dosya/klasör değiştiyse çalıştır.
 - **paths-ignore:** Belirli yollar değişse bile çalıştırma.

Pull Request Filtreleri

-
- **branches:** Sadece belirli branch'lerde çalıştır.
 - **branches-ignore:** Bazı branch'leri hariç tut.
 - **paths:** Sadece belirli dosya/klasör değiştiyse çalıştır.
 - **paths-ignore:** Belirli yollar değişse bile çalıştırma.
 - **types:** Pull Request durumuna göre (opened, synchronize, reopened, closed) çalıştır.

Permissions

-
- **read:** okuma (listeleme/indirme)
 - **read-all:** Tüm hedefler için read yetkisi.
 - **write:** yazma (push, issue/comment, release oluşturma vb.)
 - **write-all:** Tüm hedefler için write yetkisi.
 - **none:** yetki yok

Permission Scopes

-
- **contents:** Repo içeriği
 - **pull-requests:** PR ile ilişkili işlemler
 - **issues:** Issue işlemleri
 - **checks:** Raporlama senaryoları için check run
 - **statuses:** commit durumu işlemleri
 - **actions:** action'larla ilgili işlemler
 - **packages:** GitHub Packages publish/pull işlemleri
 - **deployments:** Deployment kaydı oluşturma
 - **id-token:** OIDC ile cloud'a şifresiz oturum açma

Job Anahtarları

-
- **name:** Job'un görünen adı
 - **if:** Job'un koşullu çalışması
 - **needs:** Bu job başlamadan önce bitmesi gereken job
 - **runs-on:** Runner seçimi (ubuntu-latest, windows-latest, self-hosted)
 - **permissions:** Sadece bu job için yetkiler
 - **env:** Sadece bu job için ortam değişkenleri
 - **environment:** Deploy ortamı

Job Anahtarları

-
- **concurrency**: Bu job için eşzamanlılık/çakışma kontrolü
 - **timeout-minutes**: Job'un maksimum çalışma süresi
 - **strategy+matrix**: Aynı job için farklı kombinasyonlar
 - **continue-on-error**: Job hata olsa bile workflow devam eder
 - **defaults**: Job içindeki **run** adımlarının varsayılan shell'i

Job Anahtarları

-
- **container:** Job'u container içinde çalıştırma. (Docker tabanlı build/test)
 - **services:** Job'a bağlı servis container'ları
 - **steps:** Job'un içerisinde sırayla çalışan adımlar
 - **outputs:** Job'un ürettiği çıktıları diğer Job'lara aktarma
 - **uses:** Başka bir workflow'u Job olarak çağırma

Step Anahtarları

-
- **name:** Step'in görünen adı
 - **id:** Step'in benzersiz kimliği
 - **uses:** Hazır action çalıştırır
 - **run:** Shell komutu çalıştırır
 - **with:** uses ile çağırıldığında action'a parametre gönderir
 - **env:** Sadece bu adım için ortam değişkenleri
 - **if:** Step'i koşula bağlı çalıştır

Step Anahtarları

-
- **shell:** run komutunun hangi shell ile çalışacağı (bash, pwsh, cmd)
 - **working-directory:** Komutların çalışacağı dizin
 - **continue-on-error:** Step hata olsa bile Job devam eder
 - **timeout-minutes:** Step'in maksimum çalışma süresi

Bazı Hazır Action'lar

-
- `actions/checkout@v4`
 - `actions/cache@v*`
 - `actions/upload-artifact@v*`
 - `actions/download-artifact@v*`
 - `actions/github-script@v*`

Bazı Hazır Action'lar

- `actions/setup-dotnet@v*`
 - `actions/setup-node@v*`
 - `actions/setup-java@v*`
 - `actions/setup-python@v*`
 - `actions/setup-go@v*`
 - `ruby/setup-ruby@v1`
-

Bazı Hazır Action'lar

-
- `docker/login-action@v*`
 - `docker/build-push-action@v*`
 - `Azure/login@v*`
 - `aws-actions/configure-aws-credentials@v*`
 - `google-github-actions/auth@v*`

Bazı Hazır Action'lar

-
- `github/super-linter@v*`
 - `codecov/codecov-action@v*`
 - `dependabot/fetch-metadata@v*`
 - `dorny/paths-filter@v3`

GitHub Actions Q&A

Örnek Proje Üzerinde GitHub Actions & Azure App Service Uygulaması

Azure DevOps Pipelines

Pipeline Yapısı

-
- **name:** Pipeline'ın görünen adı.
 - **trigger:** Push/commit tetikleme (CI).
 - **pr:** Pull Request doğrulama tetikleme.
 - **variables:** Pipeline genelinde değişkenler (secret/normal).
 - **pool:** Agent seçimi (Microsoft-hosted / self-hosted).
 - **stages:** Build/Deploy gibi aşamalar.
 - **resources:** Repo, pipeline, container vb. bağımlılıklar.

Pipeline Yapısı

-
- **stage**: Mantıksal aşama (Build, Deploy).
 - **jobs**: Stage içindeki işler (paralel/seri).
 - **steps**: Job içindeki adımlar.
 - **task**: Hazır görev (UseDotNet@2, DotNetCoreCLI@2, AzureWebApp@1).
 - **script**: Bash/PowerShell komutları.
 - **displayName**: UI'da görünen ad (stage/job/step).

Pipeline Yapısı

-
- **Multi-stage** YAML: CI + CD aynı dosyada yönetim.
 - **Artifacts**: publish/download ile paket paylaşımı.
 - **conditions**: succeeded(), failed(), eq(), startsWith().
 - **templates**: YAML şablonlarıyla tekrar kullanılabilir parçalar.
 - **environments**: onaylar + checks (prod gate).
 - **variable groups**: ortak konfigürasyon ve yetkilendirme.

Tetikleme Stratejileri (trigger/pr)

-
- **trigger**: branch/path filtreli CI.
 - **pr**: PR validation (autoCancel).
 - **schedules**: cron tabanlı planlı çalıştırma.
 - **resources**: başka pipeline tamamlanınca tetikleme.
 - **manual**: Run pipeline (parametrelili çalıştırma).

Branch & Path Filtreleri

-
- **branches:** include / exclude
 - **paths:** include / exclude
 - **Monorepo'da** sadece ilgili klasör değişince çalıştırma.
 - Örnek: include: main, develop | exclude: docs/*, *.md

Pull Request Filtreleri (pr)

-
- **branches:** include / exclude
 - **paths:** include / exclude
 - **autoCancel:** true (yeni commit gelince eskisini iptal).
 - **Branch policy** ile PR üzerinde build/test zorunluluğu.

Service Connections & Permissions

-
- **Service connection:** Azure aboneliğine erişim (ARM).
 - **Pipeline identity:** Project Build Service hesabı.
 - **Least privilege:** sadece gerekli subscription/resource group.
 - Secure files / variable group izinleri (kim kullanabilir?).
 - Environment approvals & checks ile prod koruması.

Yetki Kapsamları (Scopes)

-
- **Job authorization scope:** “current project” ile sınırla.
 - **Protected resources:** service connection/variable group için ayrı yetkiler.
 - **Secret’lar** otomatik maskelenir; log’a yazdırma riskleri.
 - **Key Vault** entegrasyonu ile secret’ları merkezileştirme.

Stage Anahtarları

-
- **stages:** []
 - **stage:** stage adı
 - **displayName:** görünen ad
 - **dependsOn:** bağımlı stage'ler
 - **condition:** koşullu çalışma
 - **variables:** stage değişkenleri
 - **jobs:** []

Job Anahtarları

-
- **jobs:** []
 - **job:** job adı
 - **displayName:** görünen ad
 - **pool:** agent seçimi
 - **dependsOn:** bağımlı job'lar
 - **condition:** koşullu çalışma
 - **timeoutInMinutes:** zaman aşımı
 - **workspace:** clean (temiz çalışma alanı)

Job Anahtarları (Matrix)

-
- **strategy.matrix**: tek YAML ile çoklu varyasyon (api/panel).
 - **maxParallel**: paralellik limiti.
 - **Matrix** değerleri ile path / csproj / env yönetimi.
 - Aynı pipeline ile 2 uygulama x2 senaryo yönetimi.

Step/Task Anahtarları

- **steps:** []
- **task:** GörevAdı@versiyon (UseDotNet@2, DotNetCoreCLI@2, AzureWebApp@1)
- **inputs:** görev parametreleri
- **displayName:** görünen ad
- **condition:** koşullu çalışma
- **env:** step bazlı ortam değişkenleri
- **continueOnError:** hata olsa da devam et

Script Adımları & Değişken Setleme

-
- **script** / **bash** / **pwsh** ile komut çalıştırma.
 - Çıktıdan değişken setleme: `##vso[task.setvariable variable=...]`.
 - Fail davranışı: `exit code`, `failOnStdErr`.
 - Test sonuçları: `PublishTestResults@2`.

Bazı Hazır Task'ler (Build/Test)

-
- UseDotNet@2 (.NET SDK/Runtime).
 - DotNetCoreCLI@2
(restore/build/test/publish/custom).
 - NuGetToolInstaller@1 / NuGetCommand@2.
 - PublishTestResults@2.

Bazı Hazır Task'ler (Artifacts/P aket)

-
- PublishPipelineArtifact@1 / DownloadPipelineArtifact@2.
 - publish / download anahtar sözcükleri (kısayol).
 - ArchiveFiles@2 (zip paket).
 - CopyFiles@2.

Bazı Hazır Task'ler (Deploy: Azure)

-
- AzureWebApp@1 (App Service + slot).
 - AzureCLI@2 (özelleştirilmiş deploy/script).
 - FileTransform@2 (config transform / JSON substitution).
 - Azure Key Vault / variable group ile secret injection.

Bazı Hazır Task'ler (Deploy: FTP/SSH)

-
- FtpUpload@2 (FTP ile dosya yükleme).
 - SSH@0 (uzak makinede komut çalıştırma).
 - CopyFilesOverSSH@0 (çıktıyı SSH ile kopyalama).
 - WindowsMachineFileCopy@2 (Windows hedefe dosya kopyalama).

Azure DevOps Pipelines Q&A

Örnek Proje Üzerinde Azure DevOps Pipelines & Azure App Service Uygulaması

Teşekkürler

kadir@pikselakademi.com
linkedin.com/in/kadirguventurk