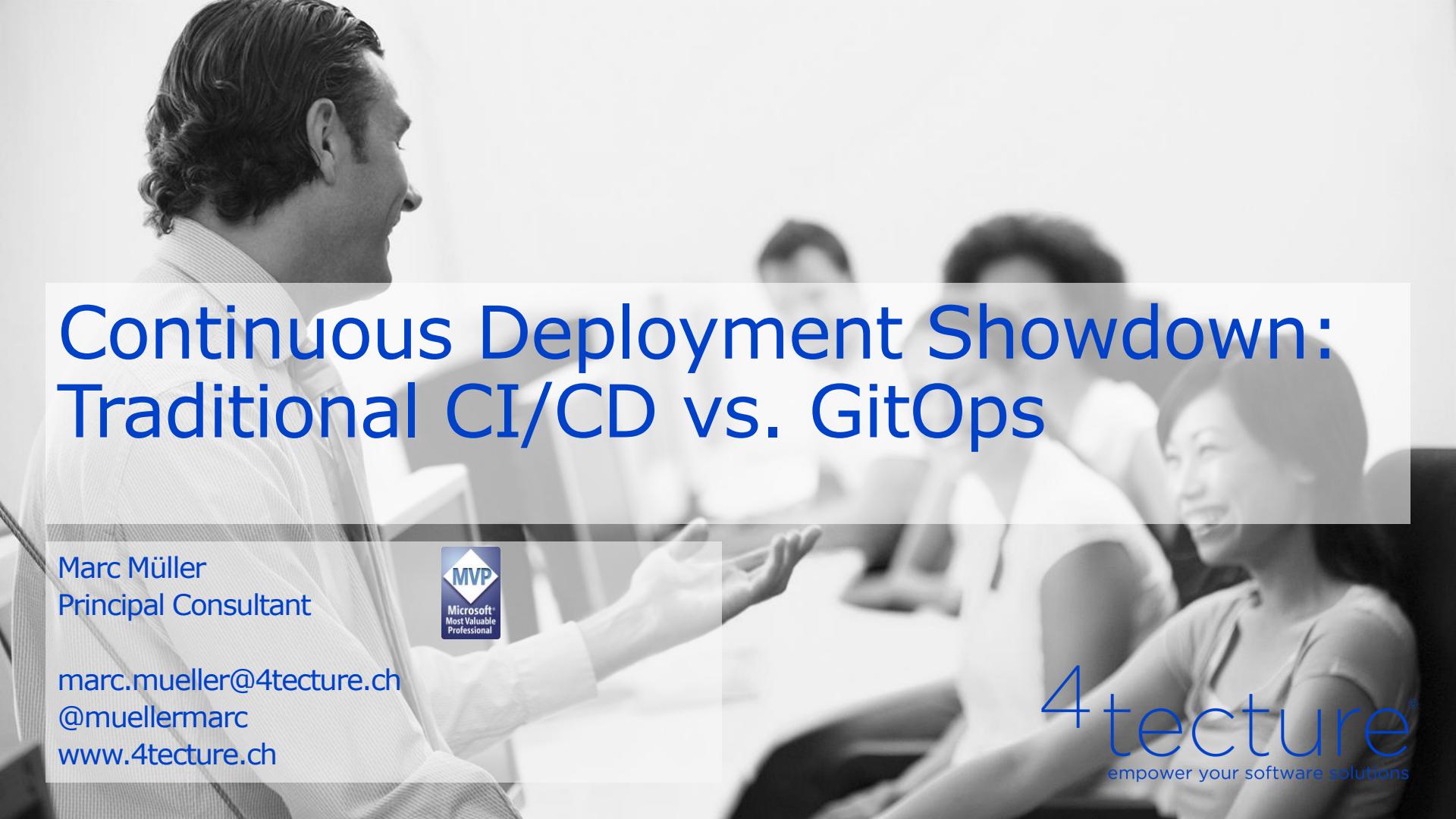


Continuous Deployment Showdown: Traditional CI/CD vs. GitOps



Marc Müller
Principal Consultant



marc.mueller@4tecture.ch
@muellermarc
www.4tecture.ch

4tecture®
empower your software solutions



About me:

Marc Müller
Principal Consultant
@muellermarc



4tecture[®]
empower your software solutions

Our Products:

Multi-Tenant OpenID
Connect Identity Provider



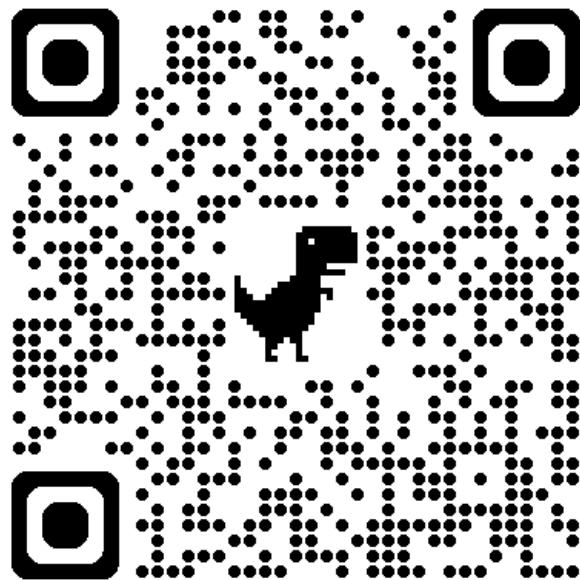
www.proauth.net

Enterprise Application
Framework for .NET



www.reafx.net

Slide Download



<https://www.4tecture.ch/events/dwx24cicdgitops>

Agenda

- Intro
- Setup
- Implementing the Deployment
- Staging
- Automatic Updates
- Cluster Setup with Azure Pipelines
- Conclusion



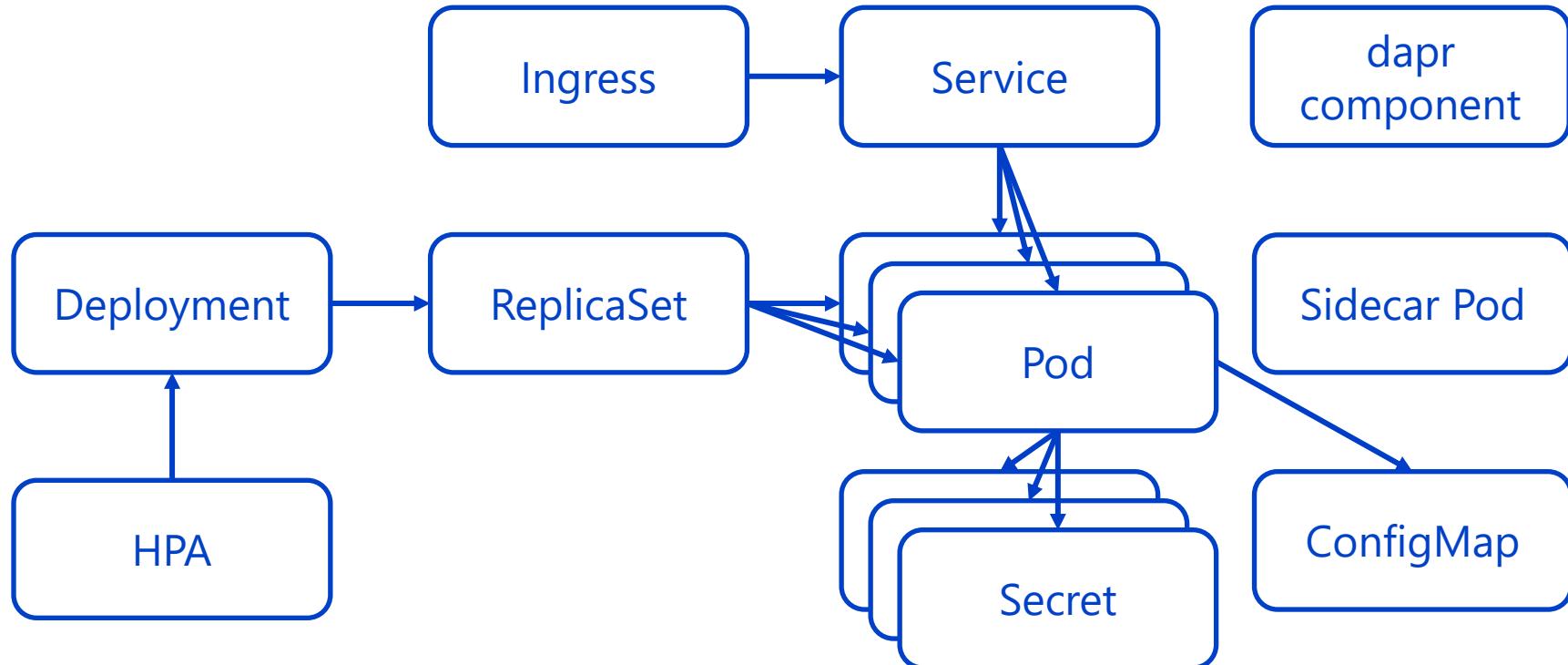
Intro

4tecture®
empower your software solutions

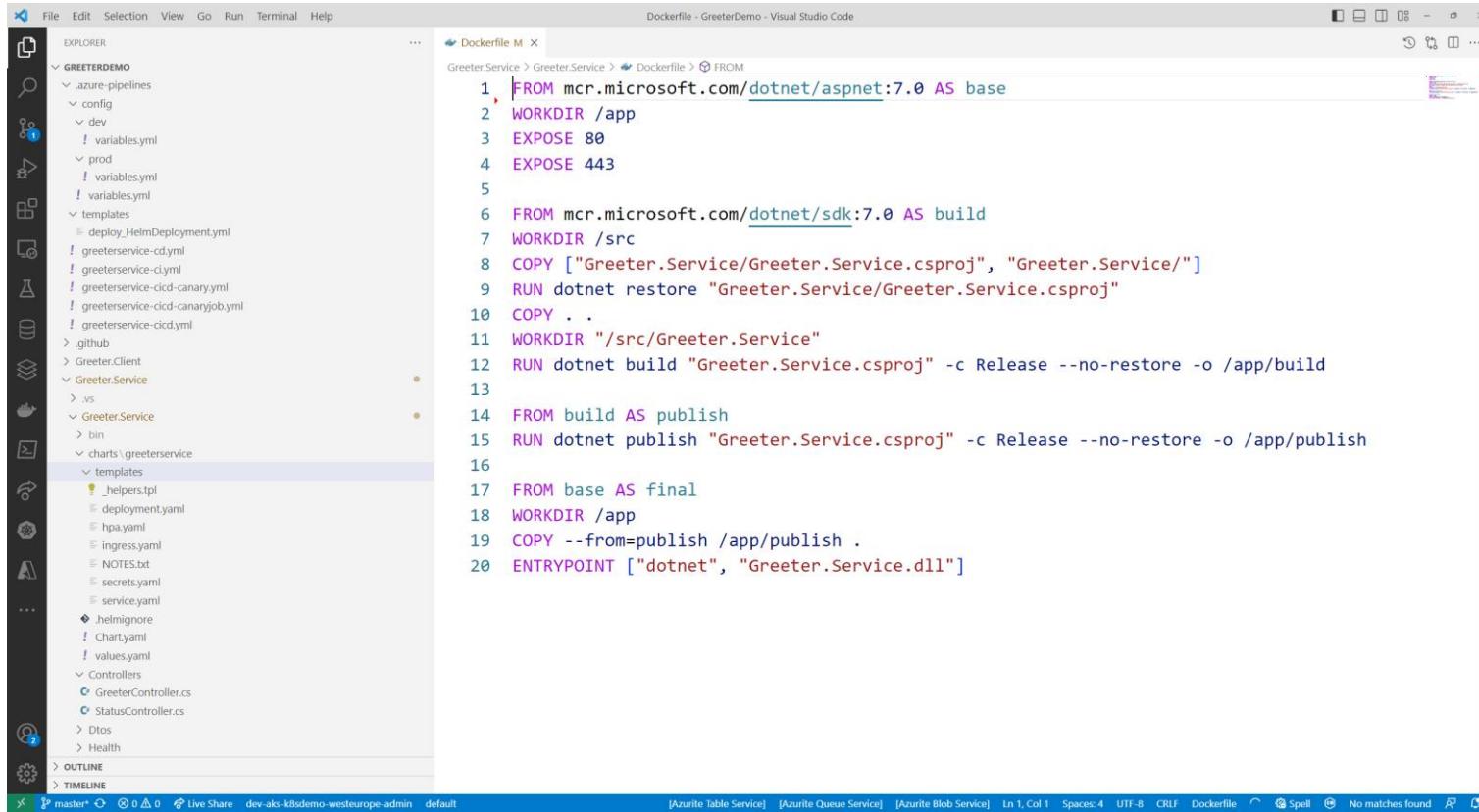
Our playground...

- ASP.NET Core Application
- Dockerized
- Kubernetes deployment environment

Kubernetes Deployment



We focus on deployment ...this is the build



The screenshot shows a Visual Studio Code window with the title "Dockerfile - GreeterDemo - Visual Studio Code". The left sidebar displays the project structure under "EXPLORER". The "GREETERDEMO" folder contains subfolders like ".azure-pipelines", "config", "dev", "prod", "templates", and ".github". The "Greeter.Service" folder contains "vs", "bin", and "charts/greeterservice". The "charts/greeterservice" folder has a "templates" subfolder containing files such as "_helpers.tpl", "deployment.yaml", "hpa.yaml", "ingress.yaml", "NOTES.txt", "secrets.yaml", "service.yaml", ".helmignore", "Chart.yaml", and "values.yaml". The main editor area shows the Dockerfile content:

```
FROM mcr.microsoft.com/dotnet/aspnet:7.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:7.0 AS build
WORKDIR /src
COPY ["Greeter.Service/Greeter.Service.csproj", "Greeter.Service/"]
RUN dotnet restore "Greeter.Service/Greeter.Service.csproj"
COPY .
WORKDIR "/src/Greeter.Service"
RUN dotnet build "Greeter.Service.csproj" -c Release --no-restore -o /app/build
FROM build AS publish
RUN dotnet publish "Greeter.Service.csproj" -c Release --no-restore -o /app/publish
FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "Greeter.Service.dll"]
```

At the bottom, the status bar shows the file is on "master" branch, has 0 changes, and is connected to "dev-aks-k8sdemo-westereurope-admin" in "default" mode. It also lists "Live Share", "Azurite Table Service", "Azurite Queue Service", "Azurite Blob Service", "In 1, Col 1", "Spaces: 4", "UTF-8", "CRLF", "Dockerfile", "Spell", and "No matches found".

Kustomize

- **Standalone tool**
 - for customizing Kubernetes objects
 - by using a declarative configuration language
- **Built-in in kubectl since 1.14**
- **kustomization.yaml → processes resources**
- **Bases and overlays**
- **Patches, ConfigMaps and Secrets, Variable substitution**

Helm

- Package Manager for Kubernetes
- Define, install and upgrade Kubernetes applications
- Configuration based on «Charts»

Helm Features

- **Manage Complexity**
 - Charts describe even the most complex apps
 - Provide repeatable application installation
 - Serve as a single point of authority
- **Easy Updates**
 - Take the pain out of updates
 - In-place upgrades
 - Custom hooks.
- **Simple Sharing**
 - Versioning
 - Easy to share, and host on public or private servers
- **Rollbacks**
 - Use helm rollback to roll back to an older version of a release with ease.



KEEP
CALM
AND

GIT COMMIT
GIT PUSH ORIGIN MAIN

GitOps

- **Declarative Configuration:**
 - System state is described in a declarative manner, usually using files like YAML.
- **Version-Controlled System State:**
 - Desired system state is stored in a version control system, typically Git.
 - Git serves as the single source of truth.
- **Automated Delivery:**
 - Changes in the Git repository trigger automatic system updates.
- **Software Agents Ensure System Convergence:**
 - Tools continuously monitor and ensure the actual system state matches the desired state in the Git repository.
 - Any discrepancies lead the system to self-correct to match the desired state.
- **Infrastructure as Code (IaC):**
 - Infrastructure setup and configuration are defined and version-controlled as code.
- **Enhanced Visibility and Traceability:**
 - All changes are tracked, auditable, and can be rolled back using Git's capabilities.
- **Reduced Manual Intervention:**
 - Direct changes to production are discouraged.
 - Changes are made via pull requests or commits to the repository.

GitOps

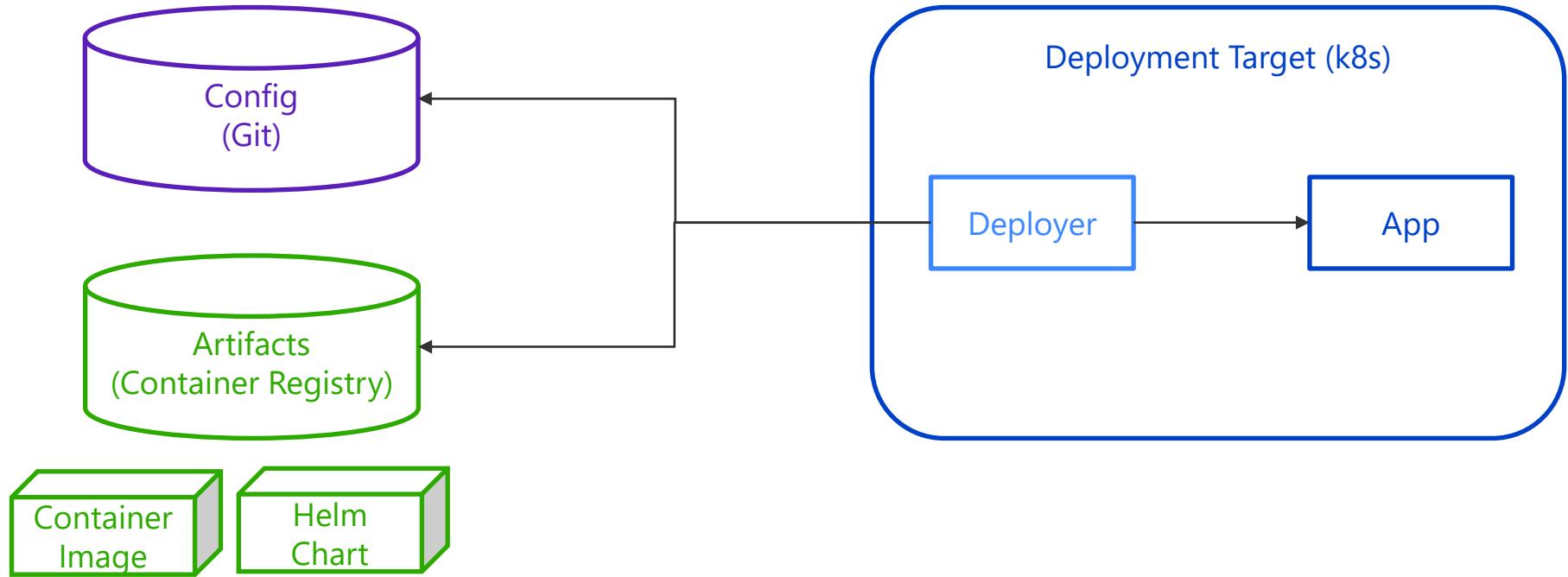
By following these principles, GitOps aims to make operations and deployment more consistent, repeatable, and secure.

Setup



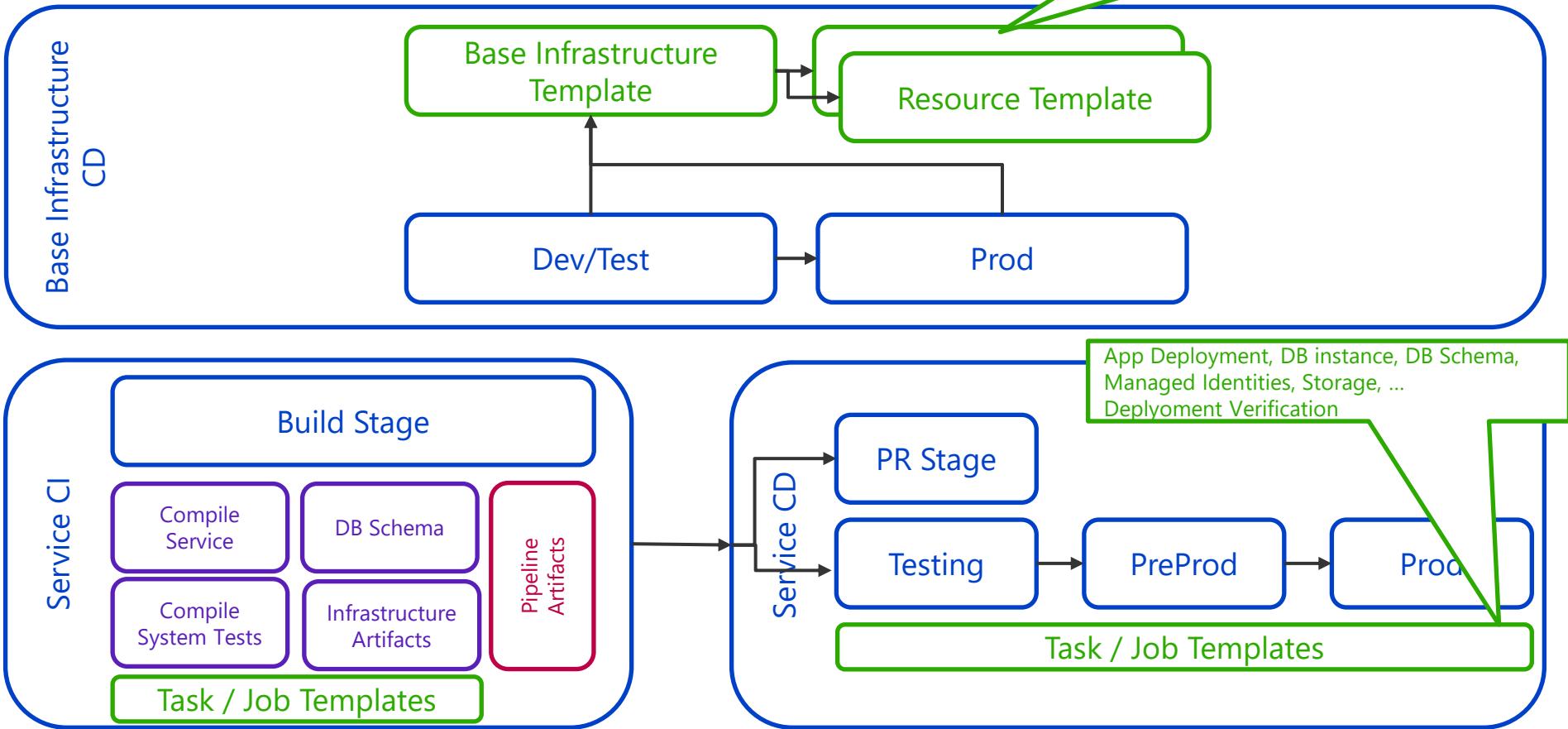
4tecture®
empower your software solutions

General Setup

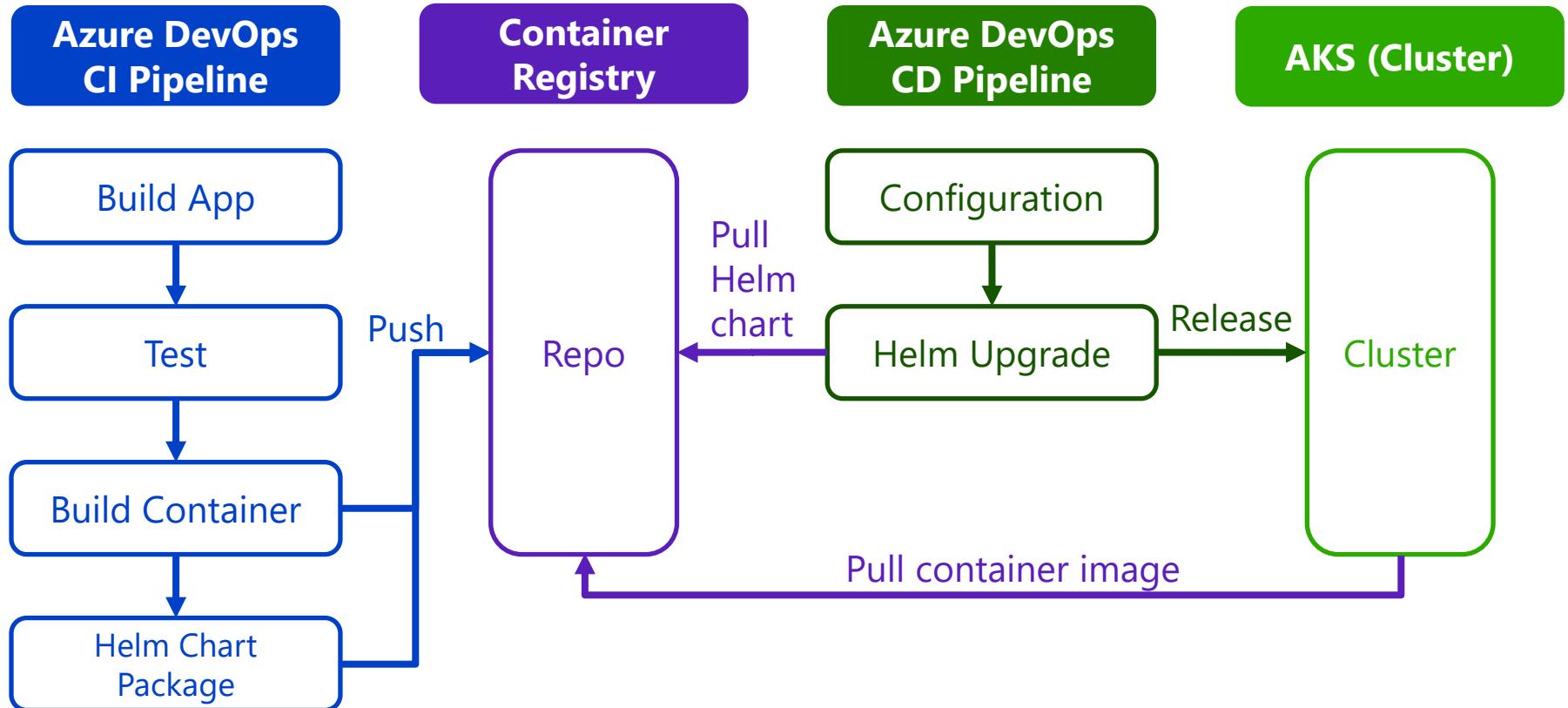


Azure Pipelines

Pipeline Design



Azure DevOps & AKS CI / CD



Configuration as Code

- Pipelines, Templates, Variables / Values files stored in Git
- Do not store secrets in Git!
- Mono-Repo / Repo per Service
- Staging is implemented by
 - Feature Branches → PR → PR Deployment with QA → PR Approval / Integration
 - Main Branch → Pre Production → Production (checks and approvals, deployment rings)



EXPLORER

INFRASTRUCTURE
>.vscode
azure
aks.bicep
arm.old.bicep
{} arm_old.json
{} k8senvironment-dev.parameters.json
k8senvironment.bicep
{} k8senvironment.parameters.json
rbac.bicep
registry.bicep
sql.bicep
vnet.bicep
deploymentagent
charts/deploymentagent
templates
agent.yaml
agentsecret.yaml
.helmignore
Chart.yaml
values.yaml
Dockerfile
README.md
start.sh
kubernetes-config
linuxworker
scripts
.gitignore
azure-pipelines.yml
README.md

Dockerfile U X

deploymentagent > Dockerfile > ...

```
1 FROM ubuntu:20.04
2 RUN DEBIAN_FRONTEND=noninteractive apt-get update
3 RUN DEBIAN_FRONTEND=noninteractive apt-get upgrade -y
4
5 RUN DEBIAN_FRONTEND=noninteractive apt-get install -y -qq --no-install-recommends \
6     apt-transport-https \
7     apt-utils \
8     ca-certificates \
9     curl \
10    git \
11    iputils-ping \
12    jq \
13    lsb-release \
14    software-properties-common \
15    libicu66
16
17 RUN curl -sL https://aka.ms/InstallAzureCLIDeb | bash
18
19 # Can be 'linux-x64', 'linux-arm64', 'linux-arm', 'rhel.6-x64'.
20 ENV TARGETARCH=linux-x64
21
22 # Install custom tooling
23 RUN apt-get install -y wget apt-transport-https software-properties-common \
24     && wget -q https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb \
25     && dpkg -i packages-microsoft-prod.deb \
26     && apt-get update \
27     && apt-get install -y powershell \
28     && pwsh -Command Install-Module SqlServer -AllowPrerelease -force \
29     && apt-get install -y --no-install-recommends unzip \
30     && rm -rf /var/lib/apt/lists/* \
31     && wget -q -O /opt/sqlpackage.zip https://go.microsoft.com/fwlink/?linkid=2236425 && unzip -qq /opt/sqlpackage.zip -d /opt/sqlpackage && chmod +x /opt/s
32     && curl -sL https://aka.ms/InstallAzureCLIDeb | bash
33
34
35 WORKDIR /azp
36
37 COPY ./start.sh .
38 RUN chmod +x start.sh
39
40 ENTRYPOINT [ "./start.sh" ]
```

EXPLORER ... Dockerfile agent.yaml X

deploymentagent > charts > deploymentagent > templates > agent.yaml > [map] > apiVersion

```
1  apiVersion: apps/v1
2  kind: StatefulSet
3  metadata:
4    name: {{ .Chart.Name }}
5    namespace: {{ .Release.Namespace }}
6  labels:
7    chart: {{ .Chart.Name }}
8    version: {{ .Chart.Version }}
9  spec:
10    replicas: {{ .Values.replicas }}
11    serviceName: {{ .Chart.Name }}
12    selector:
13      matchLabels:
14        app: {{ .Chart.Name }}
15    template:
16      metadata:
17        labels:
18          app: {{ .Chart.Name }}
19    spec:
20      containers:
21        - name: {{ .Chart.Name }}
22        image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
23        imagePullPolicy: {{ .Values.image.pullPolicy }}
24      env:
25        - name: AZP_TOKEN
26          valueFrom:
27            secretKeyRef:
28              name: azp-agent-secret
29              key: token
30        - name: AZP_URL
31          value: {{ .Values.agent.url }}
32        - name: AZP_POOL
33          value: {{ .Values.agent.pool | default "kubernetes-azp-agents" }}
34          {{- if .Values.agent.name }}
35            - name: AZP_AGENT_NAME
36              value: {{ .Values.agent.name }}
37          {{- else }}
38            - name: AZP_AGENT_NAME
39              valueFrom:
40                fieldRef:
41                  fieldPath: metadata.name
42          {{- end }}
43        - name: AZP_WORK
44          value: {{ .Values.agent.work | default "/workspace" }}
45          {{- range $key, $value := .Values.extraEnv }}
46            - name: {{ $key }}
47              value: {{ $value | quote }}
48          {{- end }}
49      volumes:
50        - name: workspace
51          mountPath: {{ .Values.agent.work | default "/workspace" }}
```

OUTLINE 50

TIMELINE 51

feature/deploymentagent 0 △ 0 Live Share dev-aks-k8sdemo-westeuropo-admin default [Azurite Table Service] [Azurite Queue Service] [Azurite Blob Service] Ln 1, Col 1 Spaces: 2 UTF-8 CRLF helm-template 🔍 Spell No matches found 🔍 🔍

hêlñ uřgsáđê ïnştjáll' áćđôđêrl'ôýnêntjáğêntj
çháştjş dêrl'ôýnêntjáğêntj
ñáñêşrâçê áćđôáğêntj
çseájtê ñáñêşrâçê
şêtj áğêntj řôôl' l''şđêñôđêrl'ôýnêntj
şêtj áğêntj tñolêñ yy.yy.

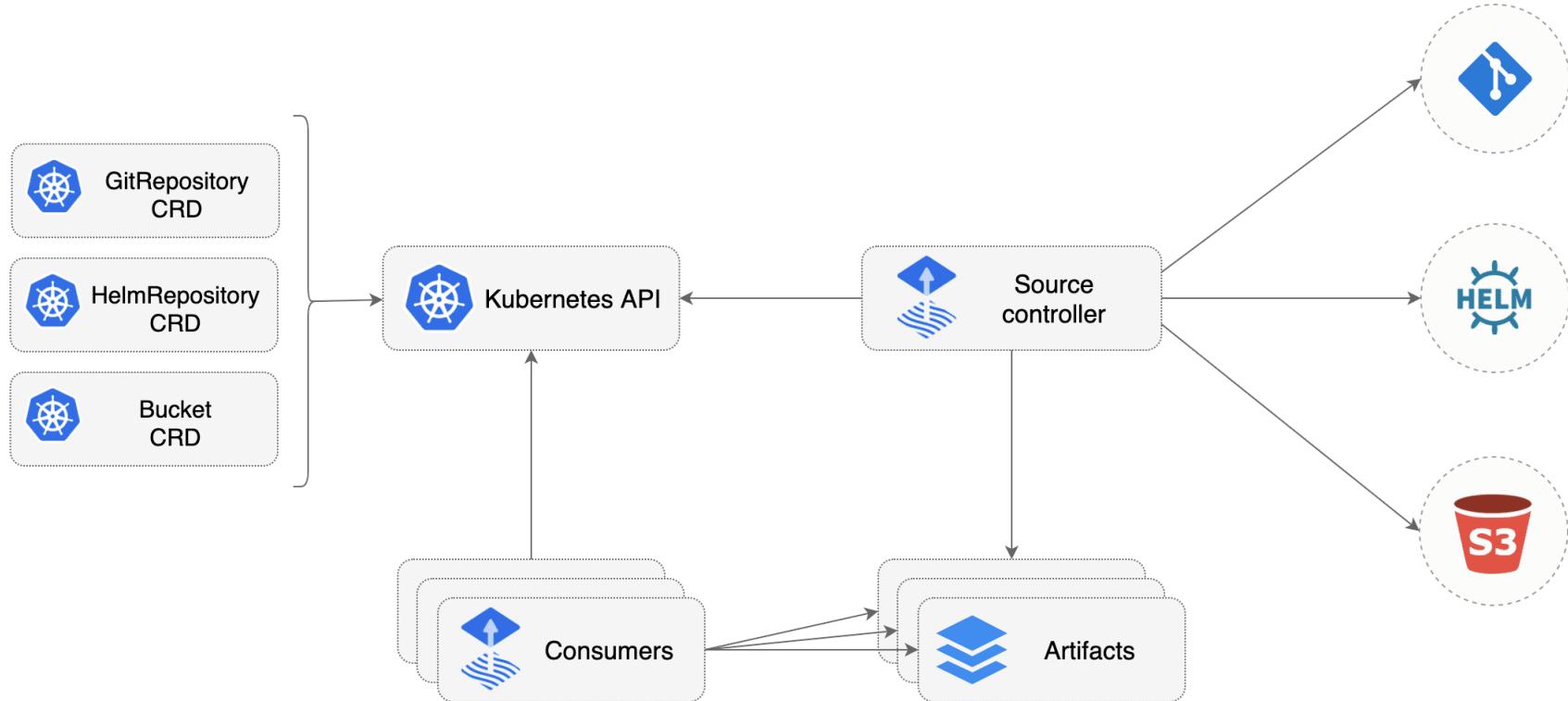
The screenshot shows the Azure DevOps Kubernetes extension interface. On the left, there's a sidebar with various navigation items like Overview, Pods, Deployments, etc. The main area shows a table for 'Pods' with one item. The pod details show it's named 'deploymentagent-0', belongs to the 'azdoagent' namespace, has one container, and is controlled by a StatefulSet. It's running on node 'aks-linux1-14208566-vmss000000'. The logs pane below shows the deployment agent starting up, including steps for determining matching agents, downloading and extracting the agent, and configuring it. It also includes license agreements and connection logs for the agent.

Flux

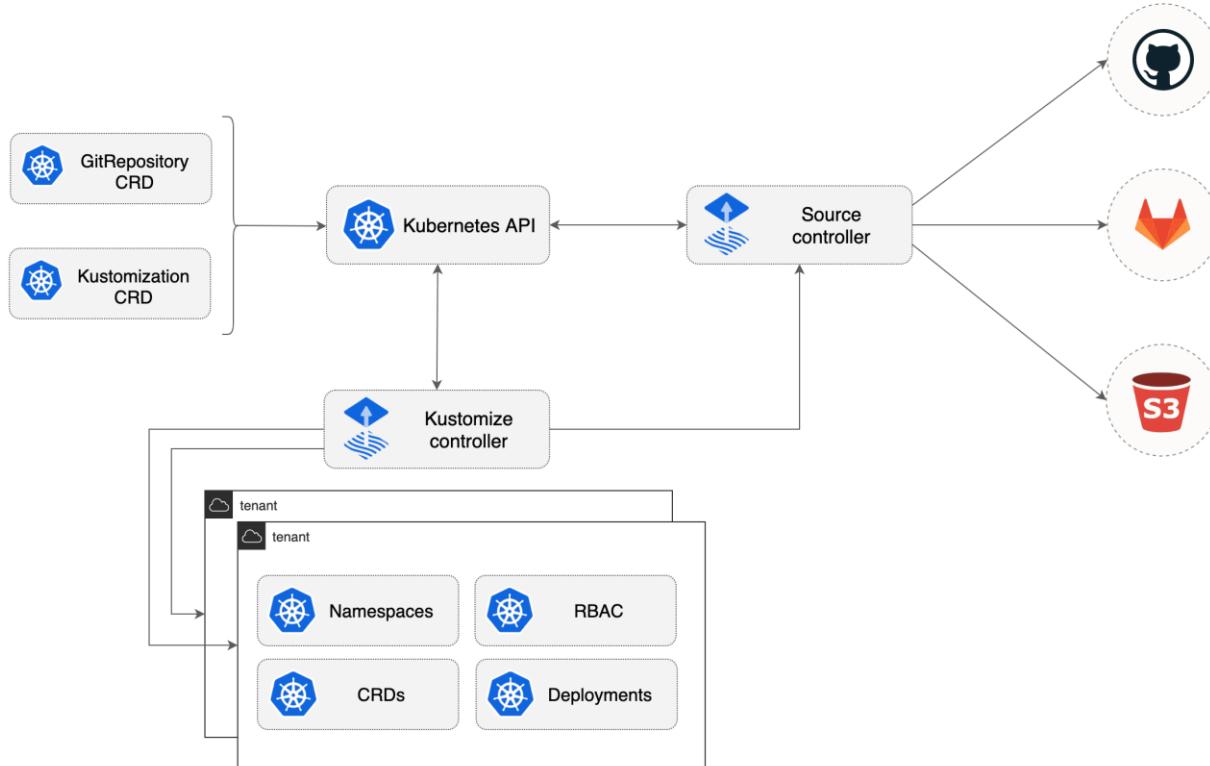
FluxCD

- Open-source GitOps toolkit for Kubernetes
- Part of the CNCF
- Monitors Git repositories
- Automatically applies the changes to the cluster

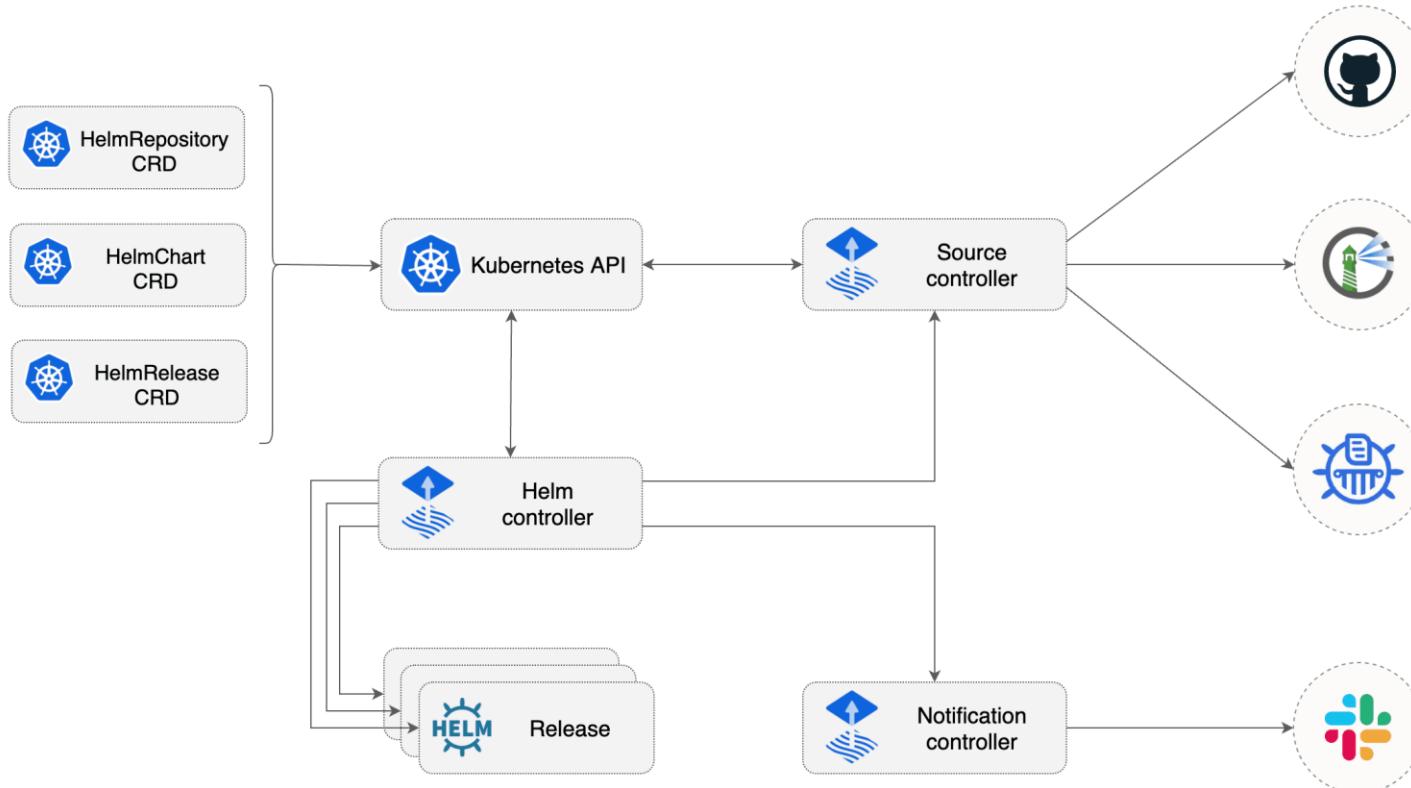
Flux Source Controller



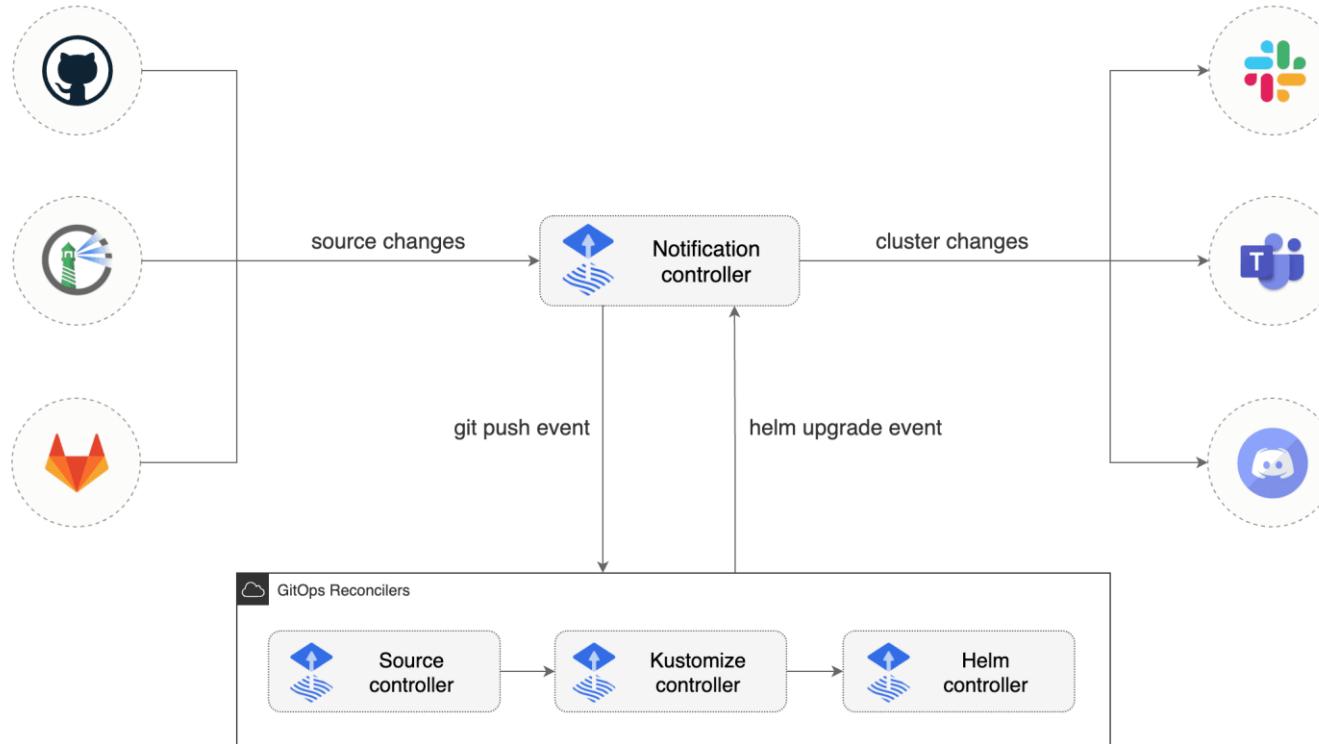
Flux Kustomize Controller



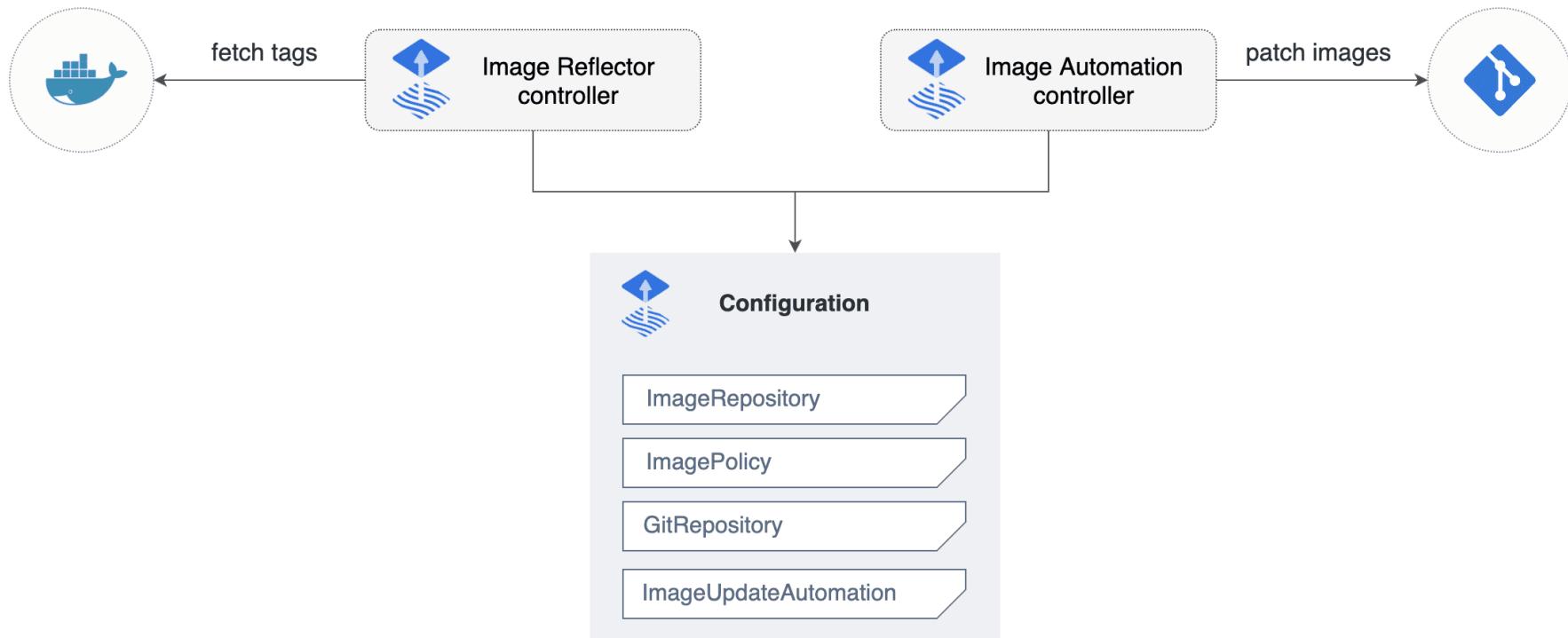
Flux Helm Controller



Flux Notification Controller

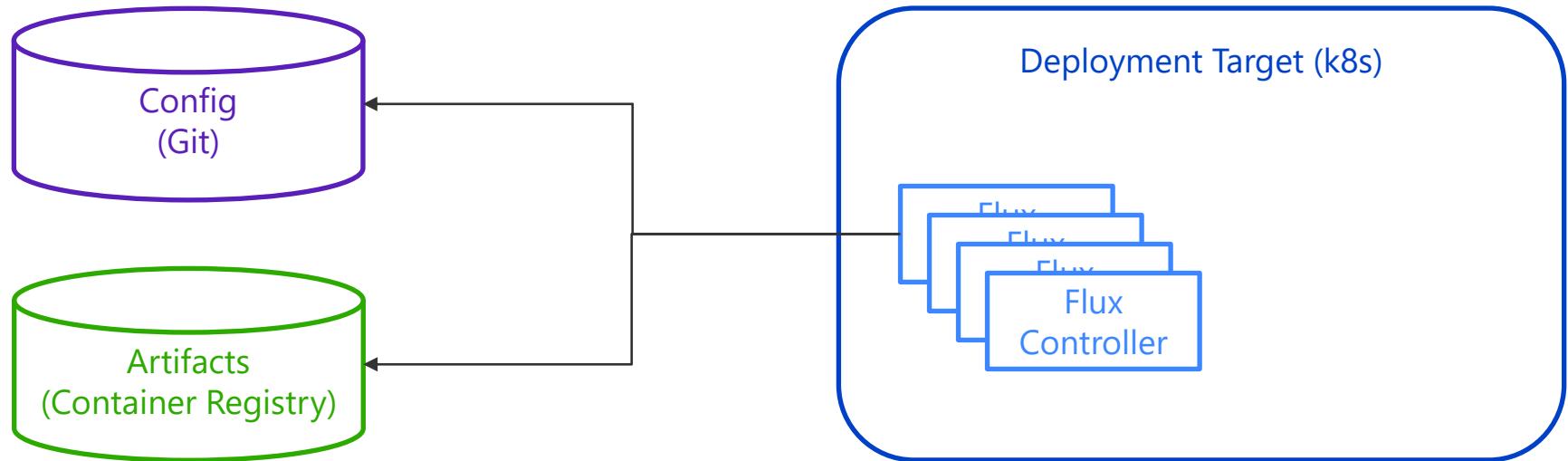


Flux Image Update Automation



Setup Flux

- Install the CLI
- Installation in k8s
 - flux bootstrap
 - flux install
 - Add kustomizations, sources, ...



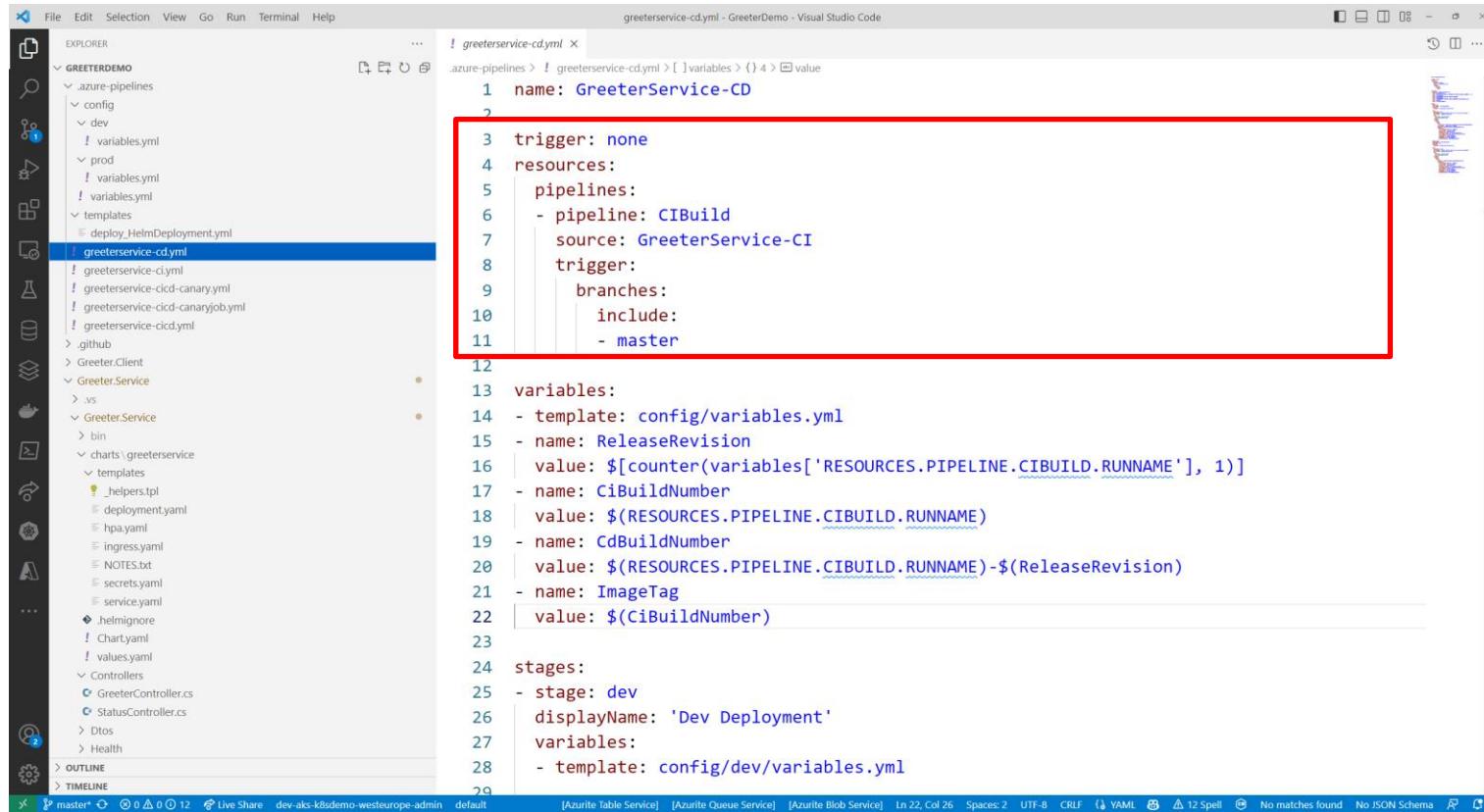
A close-up, low-angle shot of several rowers in a racing shell. Their hands are gripping yellow and black oars, which are angled downwards. The water is visible at the bottom, showing ripples and spray. The rowers are wearing blue and red athletic gear.

Implementing the Deployment

4tecture®
empower your software solutions

Azure Pipelines

CD Pipeline with CI Resource

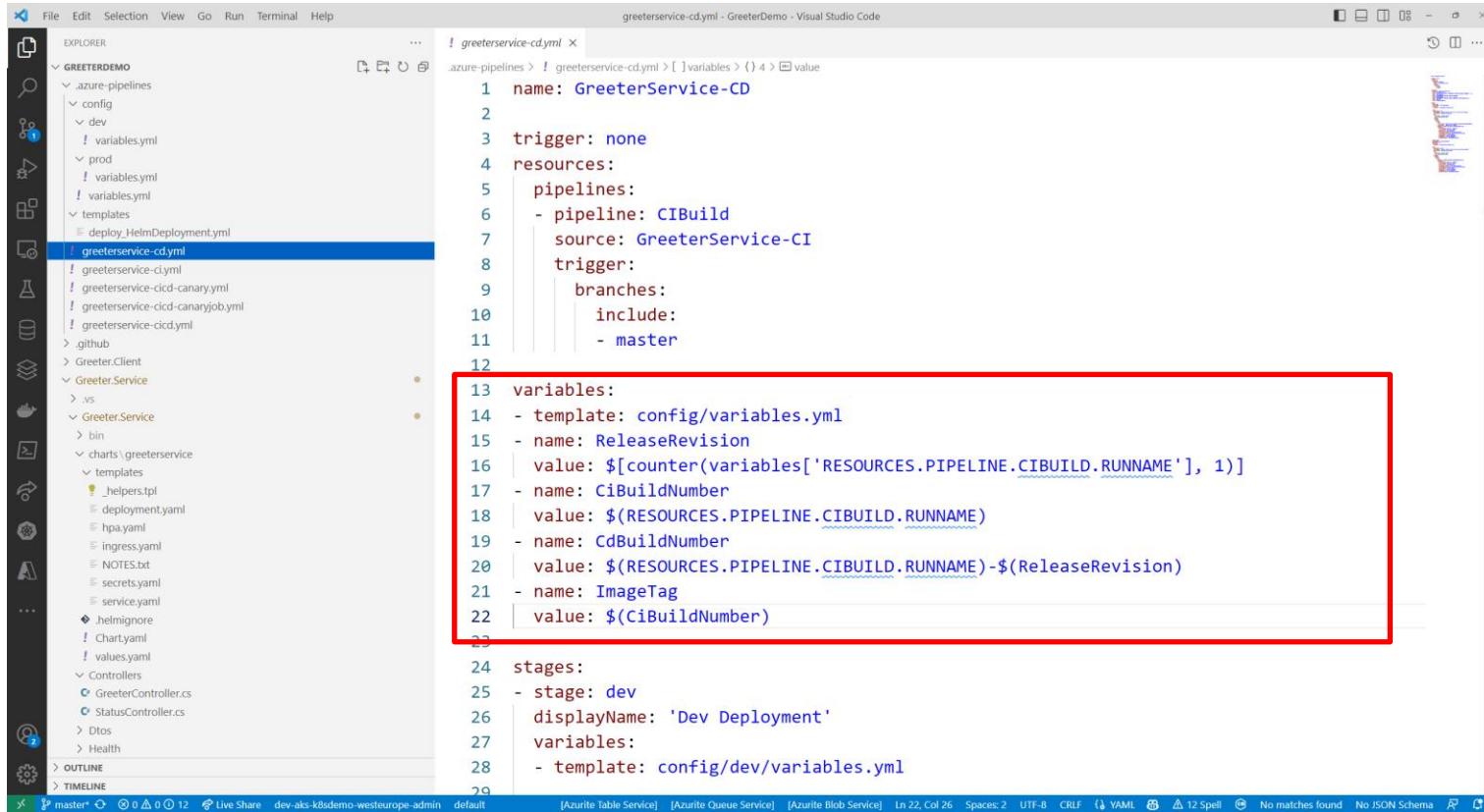


```
greeterservice-cd.yml - GreeterDemo - Visual Studio Code

File Edit Selection View Go Run Terminal Help
greeterservice-cd.yml - GreeterDemo - Visual Studio Code
greeterservice-cd.yml
1 name: GreeterService-CD
2
3 trigger: none
4 resources:
5   pipelines:
6     - pipeline: CIBuild
7       source: GreeterService-CI
8       trigger:
9         branches:
10           include:
11             - master
12
13 variables:
14   - template: config/variables.yml
15   - name: ReleaseRevision
16     value: ${[counter(variables['RESOURCES.PIPELINE.CIBUILD.RUNNAME'], 1)]}
17   - name: CiBuildNumber
18     value: ${RESOURCES.PIPELINE.CIBUILD.RUNNAME}
19   - name: CdBuildNumber
20     value: ${RESOURCES.PIPELINE.CIBUILD.RUNNAME}-${ReleaseRevision}
21   - name: ImageTag
22     value: ${CiBuildNumber}
23
24 stages:
25   - stage: dev
26     displayName: 'Dev Deployment'
27     variables:
28       - template: config/dev/variables.yml
29

master* 0 0 0 0 12 Live Share dev-aks-k8sdemo-westeu-admin default [Azure Table Service] [Azure Queue Service] [Azure Blob Service] In 22, Col 26 Spaces: 2 UTF-8 CRLF YAML No matches found No JSON Schema
```

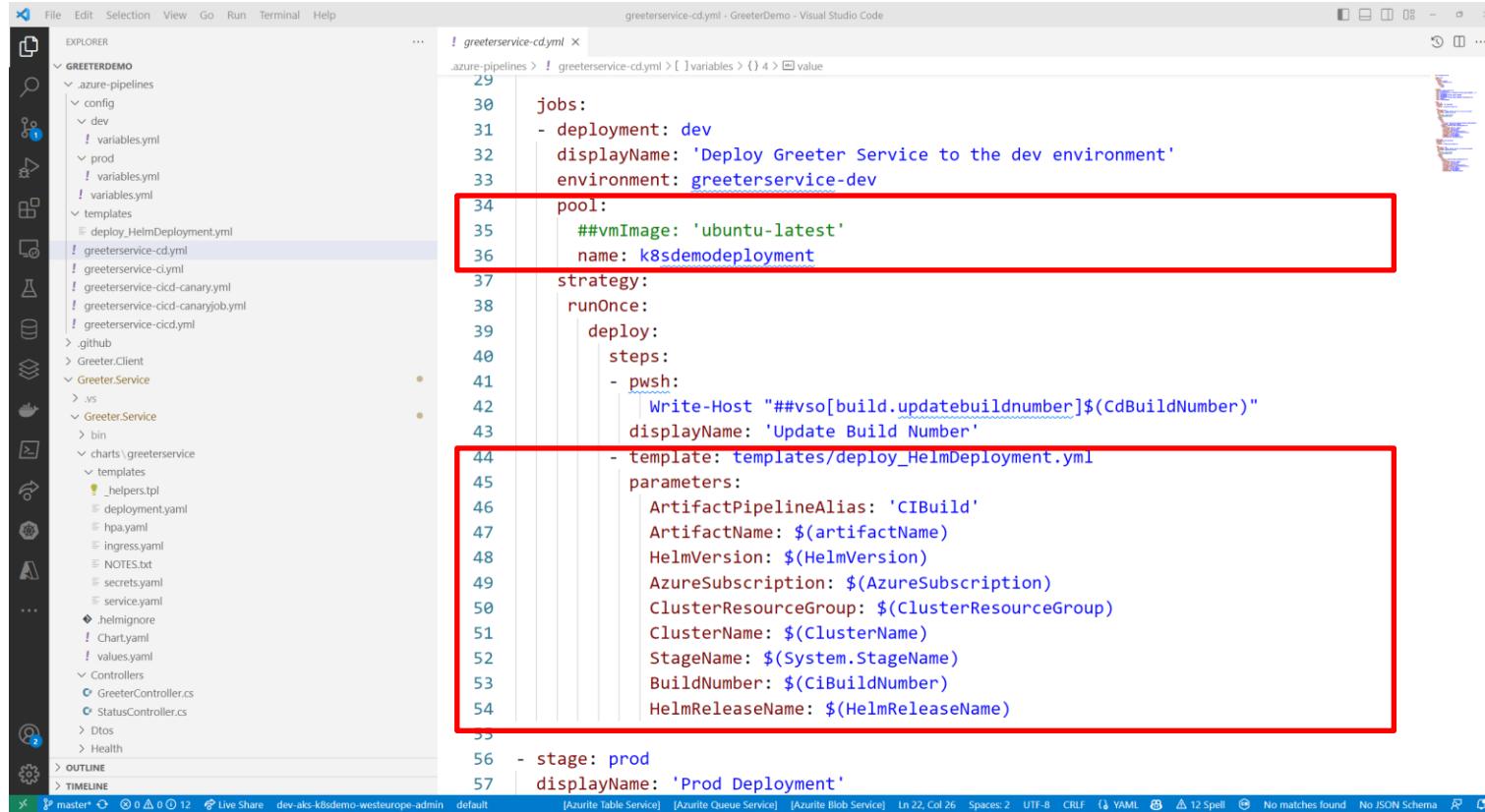
Define Versioning



The screenshot shows the Visual Studio Code interface with the file `greeterservice-cd.yml` open in the editor. The file is part of a project named `GreeterDemo`, which contains an `azure-pipelines` folder with multiple pipeline configurations for different environments (dev, prod) and stages (canary, canaryjob). The `greeterservice-cd.yml` file itself defines a CI/CD pipeline for the `GreeterService`. A red box highlights the `variables:` section, which contains definitions for `ReleaseRevision`, `CiBuildNumber`, `CdBuildNumber`, and `ImageTag`, all derived from the pipeline run number. The bottom of the screen shows the standard VS Code status bar with tabs, file paths, and other development tools.

```
1 name: GreeterService-CD
2
3 trigger: none
4 resources:
5   pipelines:
6     - pipeline: CIBuild
7       source: GreeterService-CI
8       trigger:
9         branches:
10           include:
11             - master
12
13 variables:
14   - template: config/variables.yml
15   - name: ReleaseRevision
16   value: $[counter(variables['RESOURCES.PIPELINE.CIBUILD.RUNNAME'], 1)]
17   - name: CiBuildNumber
18   value: $(RESOURCES.PIPELINE.CIBUILD.RUNNAME)
19   - name: CdBuildNumber
20   value: $(RESOURCES.PIPELINE.CIBUILD.RUNNAME)-$(ReleaseRevision)
21   - name: ImageTag
22   value: $(CiBuildNumber)
23
24 stages:
25   - stage: dev
26     displayName: 'Dev Deployment'
27     variables:
28       - template: config/dev/variables.yml
```

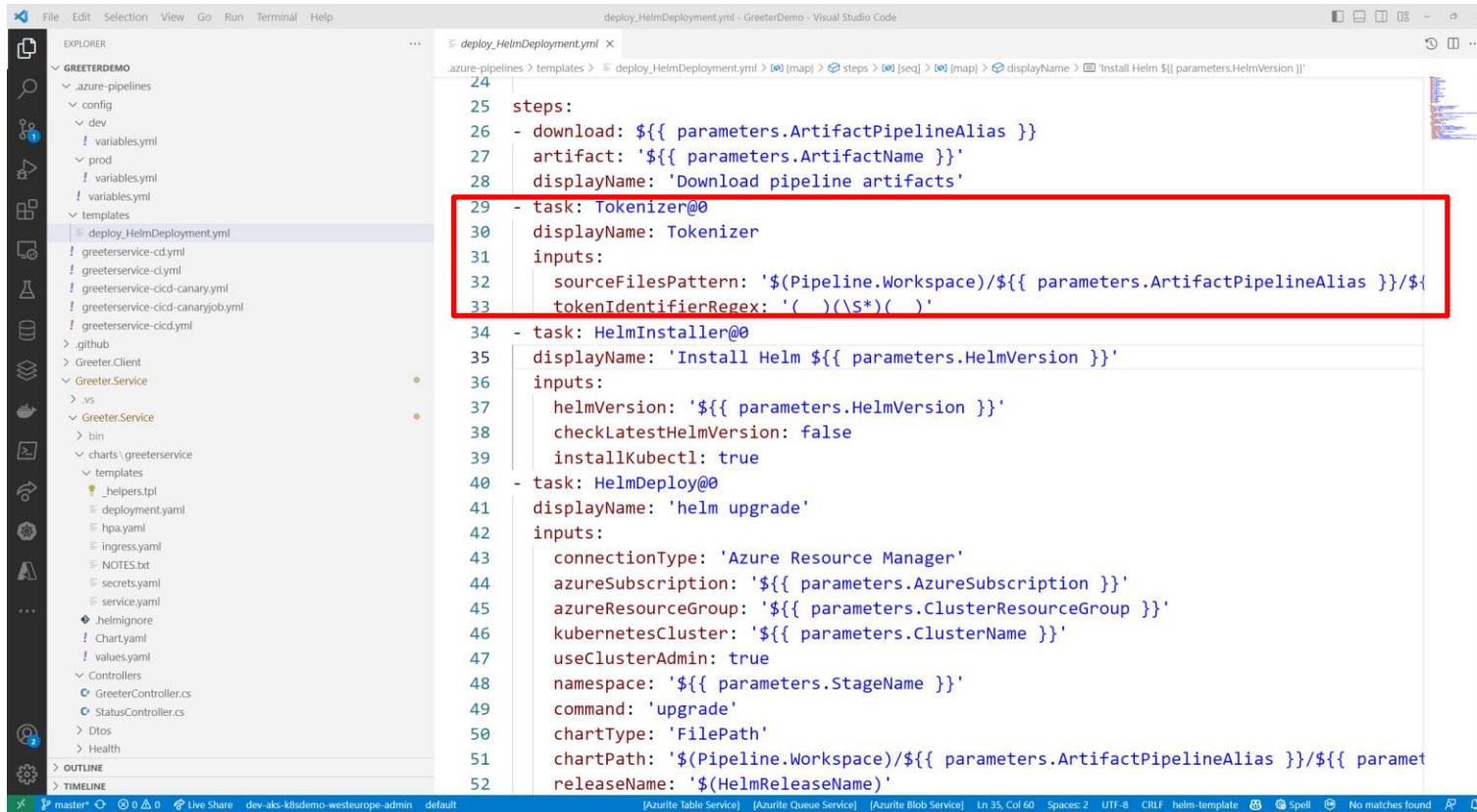
Deployment Job



```
greeterservice-cd.yml - GreeterDemo - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
GREETERDEMO
.azure-pipelines
  config
    dev
      variables.yml
    prod
      variables.yml
    templates
      deploy_HelmDeployment.yml
      greeterservice-cd.yml
      greeterservice-clyml
      greeterservice-cicd-canary.yml
      greeterservice-cicd-canaryjob.yml
      greeterservice-cicd.yml
    .github
    Greeter.Client
    Greeter.Service
      vs
      Greeter.Service
        bin
        charts/greeterservice
          templates
            _helpers.tpl
            deployment.yaml
            hpa.yaml
            ingress.yaml
            NOTES.txt
            secrets.yaml
            service.yaml
            .helmignore
            Chart.yaml
            values.yaml
        Controllers
        GreeterController.cs
        StatusController.cs
        Dtos
        Health
OUTLINE
TIMELINE
29
30   jobs:
31     - deployment: dev
32       displayName: 'Deploy Greeter Service to the dev environment'
33       environment: greeterservice-dev
34       pool:
35         ##vmImage: 'ubuntu-latest'
36         name: k8sdemodeployment
37       strategy:
38         runOnce:
39           deploy:
40             steps:
41               - pwsh:
42                 Write-Host "##vso[build.updatebuildnumber]$(CdBuildNumber)"
43                 displayName: 'Update Build Number'
44             - template: templates/deploy_HelmDeployment.yaml
45               parameters:
46                 ArtifactPipelineAlias: 'CIBuild'
47                 ArtifactName: $(artifactName)
48                 HelmVersion: $(HelmVersion)
49                 AzureSubscription: $(AzureSubscription)
50                 ClusterResourceGroup: $(ClusterResourceGroup)
51                 ClusterName: $(ClusterName)
52                 StageName: $(System.StageName)
53                 BuildNumber: $(CiBuildNumber)
54                 HelmReleaseName: $(HelmReleaseName)
55
56     - stage: prod
57       displayName: 'Prod Deployment'
```

The screenshot shows an Azure DevOps pipeline YAML file named `greeterservice-cd.yml` in Visual Studio Code. The file defines a deployment job for the `dev` environment. The job uses an Ubuntu VM (`k8sdemodeployment`) and the `deploy_HelmDeployment` template. The template parameters include `CIBuild`, `artifactName`, `HelmVersion`, `AzureSubscription`, `ClusterResourceGroup`, `ClusterName`, `StageName`, `BuildNumber`, and `HelmReleaseName`. The entire `pool` and `template`/`parameters` sections are highlighted with a red box.

Deployment



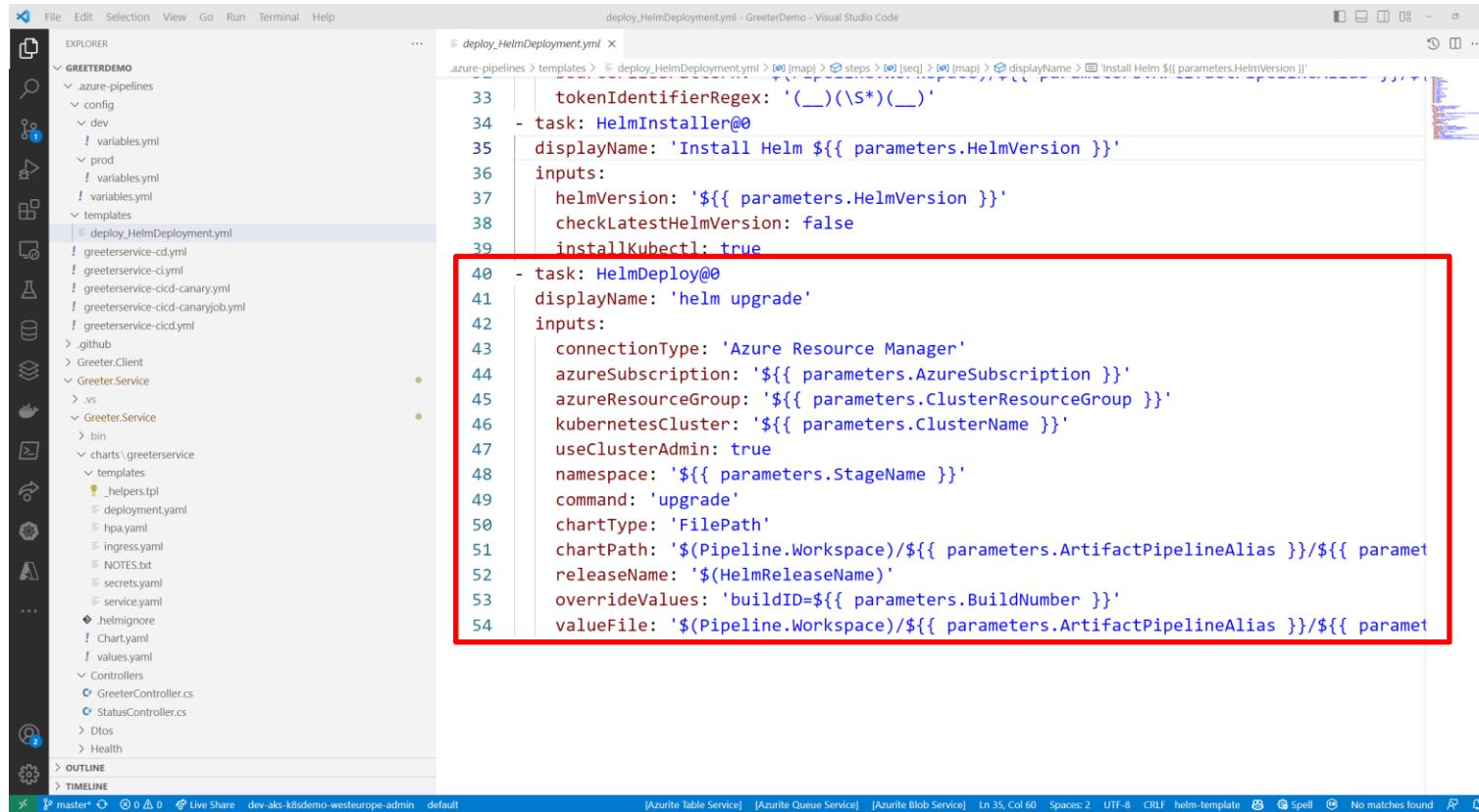
The screenshot shows the Visual Studio Code interface with an Azure DevOps pipeline YAML file open. The file is titled `deploy_HelmDeployment.yml` and is part of a project named `GreeterDemo`. The pipeline template includes several steps, with the `Tokenizer` task highlighted by a red box.

```
24
25 steps:
26 - download: ${{ parameters.ArtifactPipelineAlias }}
27 artifact: '${{ parameters.ArtifactName }}'
28 displayName: 'Download pipeline artifacts'
29 - task: Tokenizer@0
30 displayName: Tokenizer
31 inputs:
32   sourceFilesPattern: '$(Pipeline.Workspace)/${{ parameters.ArtifactPipelineAlias }}/${{ tokenIdentifierRegex: '(\ )(\S*)(\ )' }}
```

The highlighted section of the code defines the `Tokenizer` task. It specifies the task name as `Tokenizer@0`, sets the `displayName` to `'Tokenizer'`, and defines the `inputs` as a pattern for source files. The pattern uses backreferences to capture groups in the identifier regex `'(\)(\S*)(\)'`.

At the bottom of the code editor, the status bar displays various details about the workspace, including the current branch (`master`), repository (`dev-aks-k8sdemo-west-europe-admin`), and default profile (`default`). The status bar also shows the number of changes (0), the number of staged files (0), and the number of unstaged files (0). Other status indicators include `Live Share`, `azurite` services (Table, Queue, Blob), and file encoding (`UTF-8 CRLF`).

Deployment

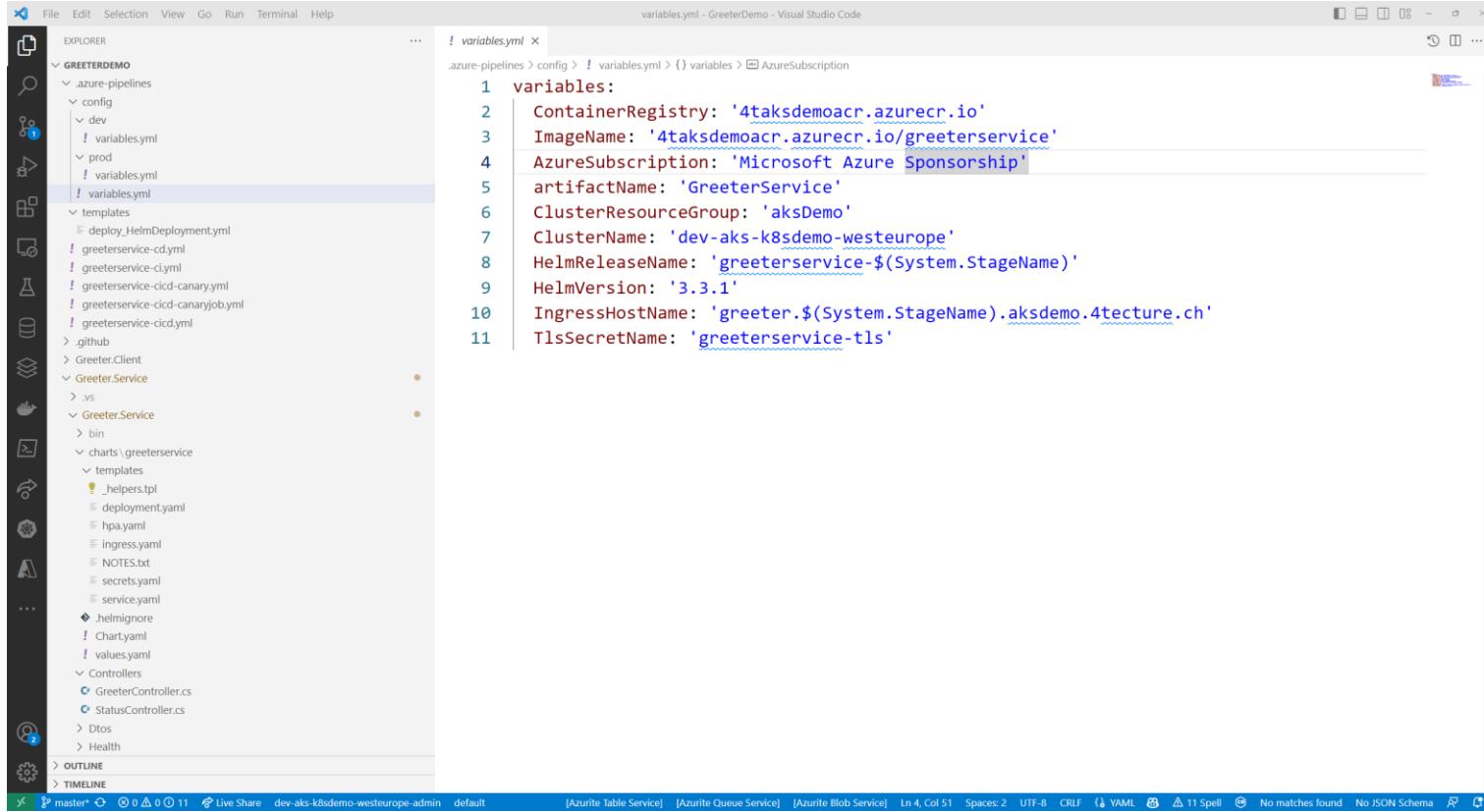


The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows a project structure under ".GREETERDEMO". Key items include ".azure-pipelines", "templates", and "charts/greeter-service".
- Code Editor (Right):** Displays a YAML file named "deploy_HelmDeployment.yml". A red box highlights the section of the code starting at line 40.
- Code Content (Red Boxed Area):**

```
33     tokenIdentifierRegex: '(_)(\s*)(_)'
34   - task: HelmInstaller@0
35     displayName: 'Install Helm ${{ parameters.HelmVersion }}'
36   inputs:
37     helmVersion: '${{ parameters.HelmVersion }}'
38     checkLatestHelmVersion: false
39     installKubectl: true
40   - task: HelmDeploy@0
41     displayName: 'helm upgrade'
42   inputs:
43     connectionType: 'Azure Resource Manager'
44     azureSubscription: '${{ parameters.AzureSubscription }}'
45     azureResourceGroup: '${{ parameters.ClusterResourceGroup }}'
46     kubernetesCluster: '${{ parameters.ClusterName }}'
47     useClusterAdmin: true
48     namespace: '${{ parameters.StageName }}'
49     command: 'upgrade'
50     chartType: 'FilePath'
51     chartPath: '${Pipeline.Workspace}/${{ parameters.ArtifactPipelineAlias }}/${{ parameters.ArtifactPipelineAlias }}'
52     releaseName: '$(HelmReleaseName)'
53     overrideValues: 'buildID=${{ parameters.BuildNumber }}'
54     valueFile: '${Pipeline.Workspace}/${{ parameters.ArtifactPipelineAlias }}/${{ parameters.ArtifactPipelineAlias }}'
```
- Bottom Status Bar:** Shows the current branch ("master"), file status, and other development information.

Configuration



The screenshot shows the Visual Studio Code interface with the following details:

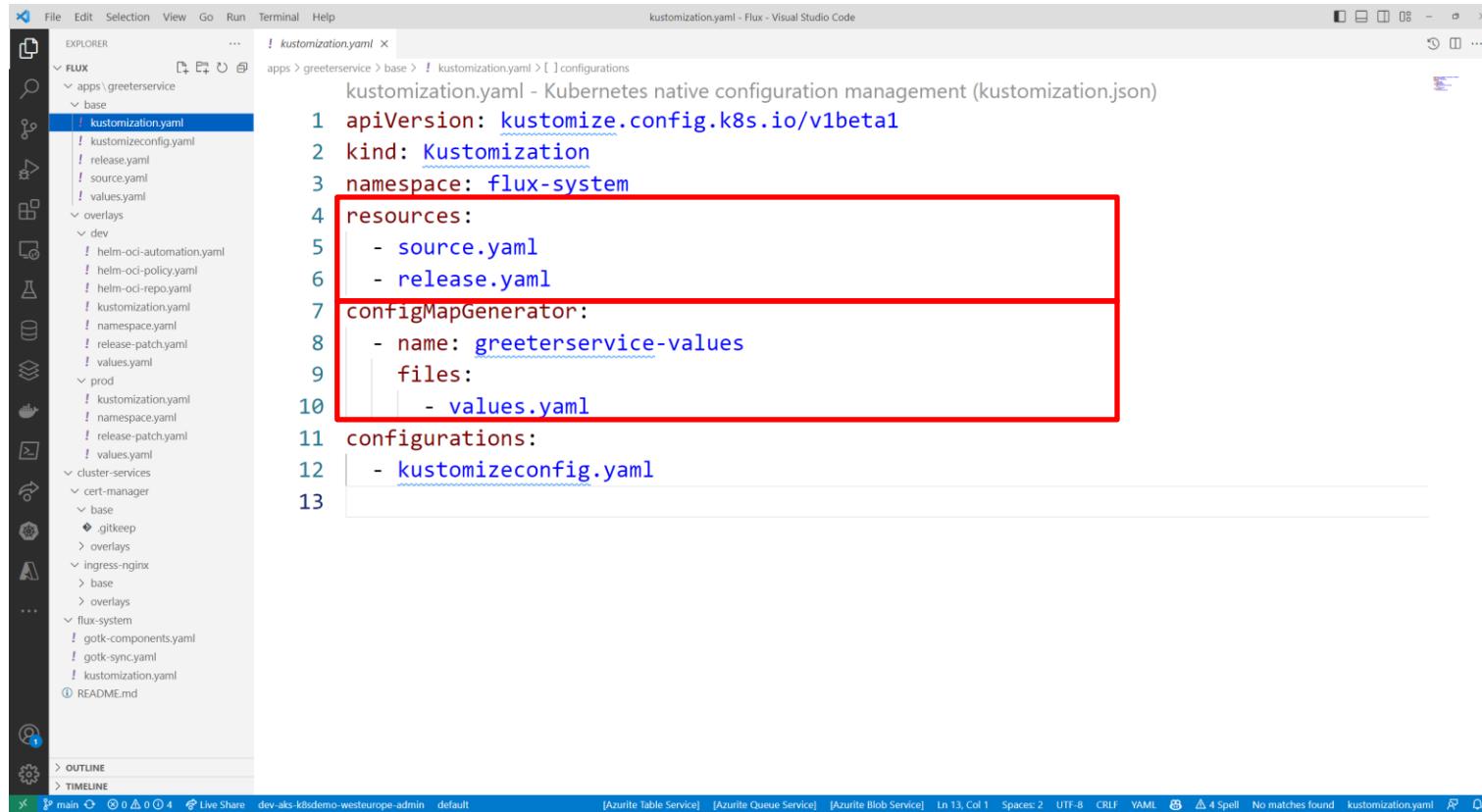
- File Explorer (Left):** Shows the project structure under the 'GREETERDEMO' folder. It includes 'azure-pipelines' (with 'config' and 'variables.yml' files for 'dev' and 'prod'), 'templates' (with multiple Helm-related files), '.github', 'Greeter.Client', 'Greeter.Service' (with 'vs', 'bin', and 'charts/greeterservice' subfolders containing various YAML files like deployment.yaml, hpa.yaml, ingress.yaml, NOTES.txt, secrets.yaml, service.yaml, and Chart.yaml), 'Controllers' (with GreeterController.cs and StatusController.cs), 'Dtots', and 'Health'. There are also 'OUTLINE' and 'TIMELINE' sections.
- Editor (Center):** Displays the 'variables.yml' file content. The code is as follows:

```
1 variables:
2   ContainerRegistry: '4taksdemoacr.azurecr.io'
3   ImageName: '4taksdemoacr.azurecr.io/greeterservice'
4   AzureSubscription: 'Microsoft Azure Sponsorship'
5   artifactName: 'GreeterService'
6   ClusterResourceGroup: 'aksDemo'
7   ClusterName: 'dev-aks-k8sdemo-westeurope'
8   HelmReleaseName: 'greeterservice-$(System.StageName)'
9   HelmVersion: '3.3.1'
10  IngressHostName: 'greeter.$(System.StageName).aksdemo.4tecture.ch'
11  TlsSecretName: 'greeterservice-tls'
```

The code editor has syntax highlighting for YAML and includes status indicators at the bottom: master*, 0 0 0 0 11, Live Share, dev-aks-k8sdemo-westeurope-admin, default, [Azurite Table Service], [Azurite Queue Service], [Azurite Blob Service], In 4, Col 51, Spaces 2, UTF-8, CRLF, YAML, 11 Spell, No matches found, No JSON Schema.

Flux

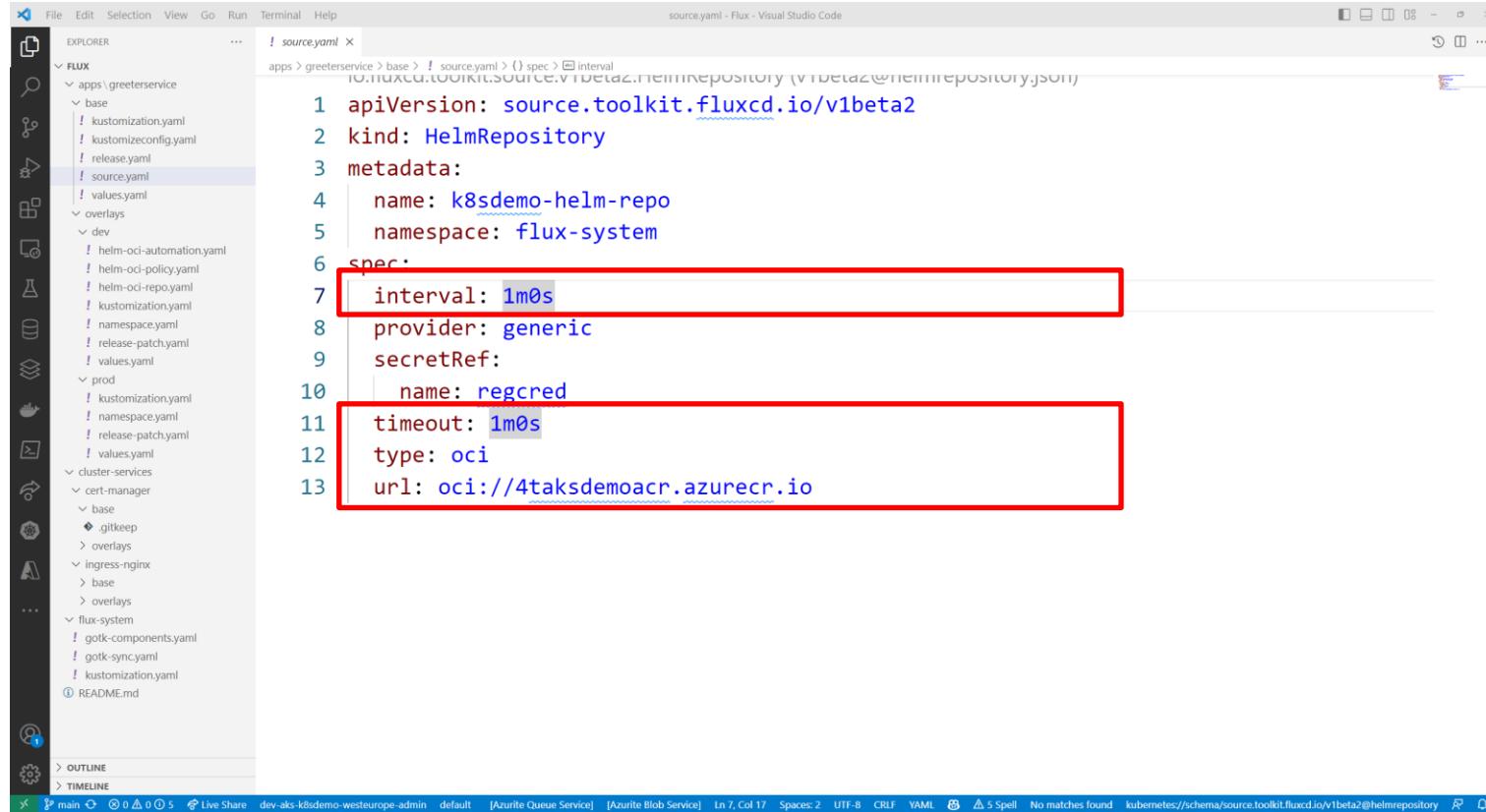
Kustomization



```
! kustomization.yaml x
apps > greeterservice > base > ! kustomization.yaml > [ ] configurations
kustomization.yaml - Kubernetes native configuration management (kustomization.json)
1 apiVersion: kustomize.config.k8s.io/v1beta1
2 kind: Kustomization
3 namespace: flux-system
4 resources:
5   - source.yaml
6   - release.yaml
7 configMapGenerator:
8   - name: greeterservice-values
9     files:
10      - values.yaml
11 configurations:
12   - kustomizeconfig.yaml
13
```

The screenshot shows a Visual Studio Code window with the file `kustomization.yaml` open. The left sidebar displays a tree view of the project structure under the `FLUX` category, including `base`, `overlays` (with `dev` and `prod` subfolders), and `cluster-services`. The right pane shows the YAML content of the `kustomization.yaml` file. A red box highlights the `resources:` section and the `configMapGenerator:` section, which includes a `name` field set to `greeterservice-values` and a `files` field containing a reference to `values.yaml`.

Helm Repository



The screenshot shows a Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Explorer:** Shows a tree view of a directory structure under the **FLUX** folder:
 - apps\greeterservice**: base, kustomization.yaml, kustomizeconfig.yaml, release.yaml, source.yaml, values.yaml.
 - overlays**: dev, helm-oci-automation.yaml, helm-oci-policy.yaml, helm-oci-repo.yaml, kustomization.yaml, namespace.yaml, release-patch.yaml, values.yaml.
 - prod**: kustomization.yaml, namespace.yaml, release-patch.yaml, values.yaml.
 - cluster-services**: cert-manager, ingress-nginx, flux-system.
 - flux-system**: gotk-components.yaml, gotk-sync.yaml, kustomization.yaml.
- Source Editor:** The active file is `source.yaml`. The code defines a `HelmRepository` resource with the following configuration:

```
apiVersion: source.toolkit.fluxcd.io/v1beta2
kind: HelmRepository
metadata:
  name: k8sdemo-helm-repo
  namespace: flux-system
spec:
  interval: 1m0s
  provider: generic
  secretRef:
    name: regcred
  timeout: 1m0s
  type: oci
  url: oci://4taksdemoacr.azurecr.io
```

The `interval`, `timeout`, and `url` fields are highlighted with red boxes.
- Bottom Status Bar:** Shows the current file path: `dev-aks-k8sdemo-westeurope-admin default [Azurite Queue Service] [Azurite Blob Service]`, line number `Ln 7, Col 17`, and other status indicators like `Spaces: 2`, `UTF-8`, `CRLF`, `YAML`, `5 Spell`, and `No matches found`.

Helm Release

```
! release.yaml x
apps > greeterservice > base > ! release.yaml > ...
io.fluxcd.toolkit.helm.v2beta1.HelmRelease (v2beta1@helmrelease.json)

1 apiVersion: helm.toolkit.fluxcd.io/v2beta1
2 kind: HelmRelease
3 metadata:
4   name: greeterservice-helm-release
5   namespace: flux-system
6 spec:
7   chart:
8     spec:
9       chart: helm/greeterservice
10      reconcileStrategy: ChartVersion
11      sourceRef:
12        kind: HelmRepository
13        name: k8sdemo-helm-repo
14        version: '0.0.1'
15      interval: 1m0s
16      targetNamespace: replaceme
17      valuesFrom:
18        - kind: ConfigMap
19          name: greeterservice-values
20        - kind: ConfigMap
21          name: greeterservice-overlay-values
22
```

The screenshot shows a Visual Studio Code window with the file "release.yaml" open. The file is a Helm Release manifest. A red box highlights the "chart" section, which contains the "spec" block. Another red box highlights the "valuesFrom" section, which contains two entries for "ConfigMap". The "targetNamespace" field is also highlighted with a red box.

Configuration

The screenshot shows a Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Explorer:** Shows a tree view of the project structure under the "FLUX" category, including "apps/greeterservice/base", "overlays/dev", "overlays/prod", and "cluster-services/cert-manager/base". A file named "values.yaml" is selected.
- Editor:** The main editor area displays a YAML configuration file named "values.yaml". The content includes:

```
replicaCount: 1
envvariables:
  servicesettings__EnablePrimeNumberCalculation: false
  servicesettings__message: Hello from FluxCD
hpa:
  averageCpuUtilization: 50
  enabled: false
  maxReplicas: 3
  minReplicas: 1
image:
  repository: 4taksdemoacr.azurecr.io/greeterservice
ingress:
  hosts:
    - greeter.replaceme.aksdemo.4tecture.ch
  tls:
    - hosts:
        - greeter.replaceme.aksdemo.4tecture.ch
secretName: greeterservice-tls
```
- Bottom Status Bar:** Shows file paths like "main", "dev-aks-k8sdemo-westeurop-admin", "default", and various status indicators such as "Live Share", "Spaces: 2", "UTF-8", "YAML", "13 Spell", and "No matches found".

Unique Resource Names

The screenshot shows a Visual Studio Code interface with two code editors and a sidebar.

Top Editor: The file is `kustomization.yaml`. It defines a `Kustomization` resource for the `flux-system` namespace. It includes a `configMapGenerator` block that creates a `greeterservice-values` Config Map from a `values.yaml` file. This entire block is highlighted with a red rectangle.

```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
namespace: flux-system
resources:
- source.yaml
- release.yaml
configMapGenerator:
- name: greeterservice-values
  files:
  - values.yaml
configurations:
- kustomizeconfig.yaml
```

Bottom Editor: The file is `kustomizeconfig.yaml`. It defines a `nameReference` block that points to a `ConfigMap` named `greeterservice-values`. This entire block is highlighted with a red rectangle.

```
nameReference:
- kind: ConfigMap
  version: v1
  fieldSpecs:
  - path: spec/valuesFrom/name
    kind: HelmRelease
- kind: Secret
  version: v1
  fieldSpecs:
  - path: spec/valuesFrom/name
    kind: HelmRelease
```

Sidebar: The sidebar is titled "Config Maps" and lists several entries:

- Name
- greeterservice-overlay-values-dev-m54kd5ft4g
- greeterservice-overlay-values-prod-ft899gm78k
- greeterservice-values-dev-7hf7g2hf25
- greeterservice-values-prod-7hf7g2hf25

Staging

A close-up, low-angle shot of a team of rowers in a boat, viewed from the stern. The rowers are wearing blue and red athletic gear. Their oars are extended, dipping into the water. The background is a bright, slightly overexposed sky. The word "Staging" is overlaid in large, bold, blue letters across the middle of the image.

4tecture®
empower your software solutions

Azure Pipelines

Pipeline Stages

Azure DevOps 4tecture-demo / k8sDemo / Pipelines / GreeterService-CD / 0.1.111-fixingcd0011-1

Search Run new ⋮

k8sDemo +

#0.1.111-fixingcd0011-1 • fix image tag
GreeterService-CD

This run will be cleaned up after 1 month based on your project settings.

Summary Environments Scans

Manually run by  Marc Müller View 264 changes

Repository and version Time started and elapsed Related Tests and coverage

GreeterDemo Yesterday at 2:09 PM 0 work items Get started

feature/fixingcd c7819c1a 1m 15s 1 consumed

Stages Jobs

Dev Deployment  1 job completed 26s

Prod Deployment  1 job completed 26s

Project settings «

Environments

Azure DevOps 4tecture-demo / k8sDemo / Pipelines / GreeterService-CD / 0.1.111-fixingcd0011-1

Search Run new ⋮

k8sDemo +

Overview Boards Repos Pipelines Pipelines Environments Releases Library Task groups Deployment groups Test Plans Artifacts

#0.1.111-fixingcd0011-1 • fix image tag
GreeterService-CD

This run will be cleaned up after 1 month based on your project settings.

Summary Environments Scans

greeterservice-dev

Jobs	Resource	Duration
Deploy Greeter Service to the d...		24s

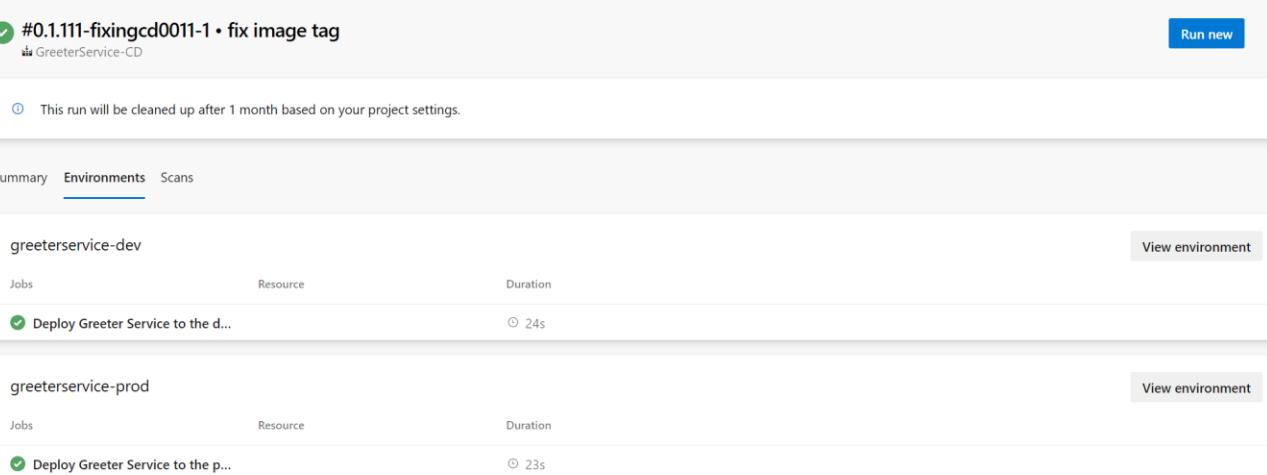
View environment

greeterservice-prod

Jobs	Resource	Duration
Deploy Greeter Service to the p...		23s

View environment

Project settings «



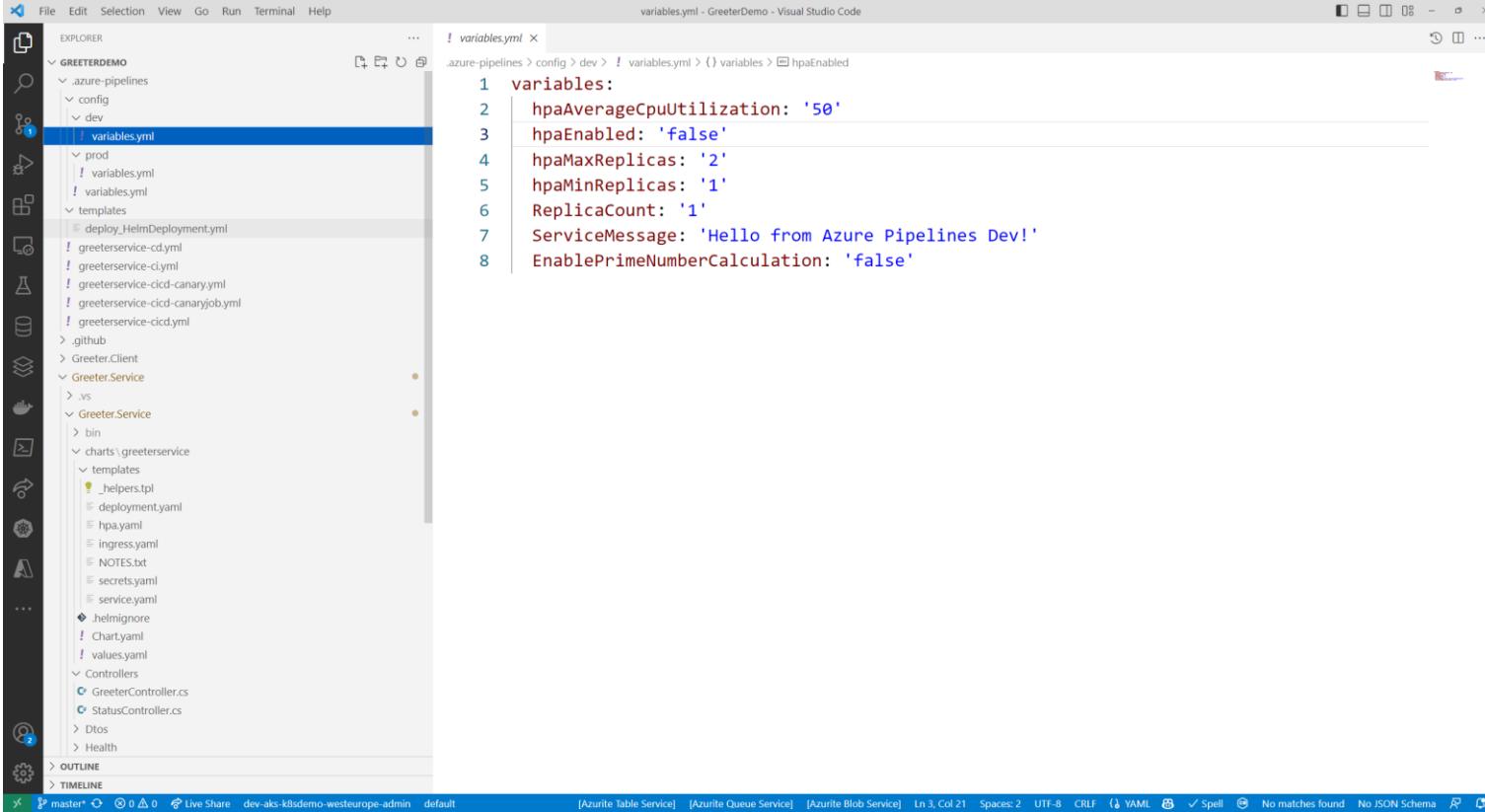
Dedicated Settings per Stage

```
greeterservice-ci.yml
stages:
- stage: dev
  displayName: 'Dev Deployment'
  variables:
    - template: config/dev/variables.yml

jobs:
- deployment: dev
  displayName: 'Deploy Greeter Service to the dev environment'
  environment: greeterservice-dev
  pool:
    ##vmImage: 'ubuntu-latest'
    name: k8sdemodeployment
  strategy:
    runOnce:
      deploy:
        steps:
- pswsh:
    Write-Host "#vso[build.updatebuildnumber]$(CdBuildNumber)"
    displayName: 'Update Build Number'
- template: templates/deploy_HelmDeployment.yml
  parameters:
    ArtifactPipelineAlias: 'CIBuild'
    ArtifacName: $(artifactName)
    HelmVersion: $(HelmVersion)
    AzureSubscription: $(AzureSubscription)
    ClusterResourceGroup: $(ClusterResourceGroup)
    ClusterName: $(ClusterName)
    StageName: $(System.StageName)
    BuildNumber: $(CIBuildNumber)
    HelmReleaseName: $(HelmReleaseName)

- stage: prod
  displayName: 'Prod Deployment'
```

Dedicated Settings per Stage



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under the workspace `GREETERDEMO`. It includes `.azure-pipelines`, `charts/greeterservice` (with files like `_helpers.tpl`, `deployment.yaml`, `hpa.yaml`, `ingress.yaml`, `NOTES.txt`, `secrets.yaml`, `service.yaml`, `.helmignore`, `Chart.yaml`, and `values.yaml`), `Controllers`, `GreeterController.cs`, `StatusController.cs`, `Dtos`, and `Health`.
- Editor (Center):** Displays the `variables.yml` file content. The file defines variables for the `dev` stage of the pipeline.
- Bottom Status Bar:** Shows the current branch as `master*`, the commit hash as `0 0 0`, and the repository name as `dev-aks-k8sdemo-westeurope-admin`. It also indicates the active tab is `default`.
- Bottom Activity Bar:** Includes icons for Live Share, Azure Table Service, Azure Queue Service, Azure Blob Service, Spaces: 2, UTF-8, CRLF, YAML, Spell, No matches found, and No JSON Schema.

```
variables.yml
variables:
  hpaAverageCpuUtilization: '50'
  hpaEnabled: 'false'
  hpaMaxReplicas: '2'
  hpaMinReplicas: '1'
  ReplicaCount: '1'
  ServiceMessage: 'Hello from Azure Pipelines Dev!'
  EnablePrimeNumberCalculation: 'false'
```

Flux

Structuring Best Practices

- Use base configuration with overlays
- Overlay config maps / secrets
- Overlay objects with patches

- apps

- application1

- base

- overlays

- dev

- prod

- application2

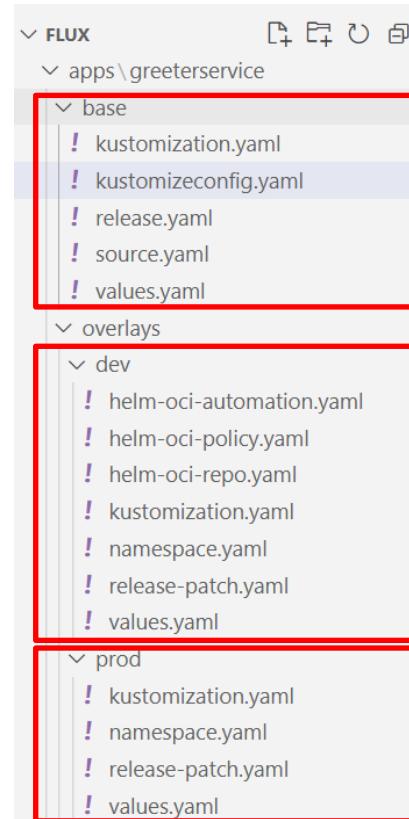
- base

- overlays

- dev

- prod

Structure



Overlay Kustomization

The screenshot shows a Visual Studio Code window with the file `kustomization.yaml` open. The left sidebar displays a file tree with a structure under the `FLUX` folder:

- `apps\greeterservice`
 - `base`
 - `! kustomization.yaml`
 - `! kustomizeconfig.yaml`
 - `! release.yaml`
 - `! source.yaml`
 - `! values.yaml`
 - `overlays`
 - `dev`
 - `! helm-oci-automation.yaml`
 - `! helm-oci-policy.yaml`
 - `! helm-oci-repo.yaml`
 - `! kustomization.yaml`
 - `! namespace.yaml`
 - `! release-patch.yaml`
 - `! values.yaml`
- `prod`
 - `! kustomization.yaml`
 - `! namespace.yaml`
 - `! release-patch.yaml`
 - `! values.yaml`
- `cluster-services`
- `cert-manager`
 - `base`
 - `! gitkeep`
 - `overlays`
 - `ingress-nginx`
 - `base`
 - `overlays`
- `flux-system`
 - `! gotk-components.yaml`
 - `! gotk-sync.yaml`
 - `! kustomization.yaml`

- `README.md`
- `OUTLINE`
- `TIMELINE`

```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
namespace: flux-system
nameSuffix: -dev
resources:
  - ../../base
  - namespace.yaml
  - helm-oci-repo.yaml
  - helm-oci-policy.yaml
  - helm-oci-automation.yaml
  - configMapGenerator:
    - name: greeterservice-overlay-values
      files:
        - values.yaml
patchesStrategicMerge:
  - release-patch.yaml
```

Overlay Values

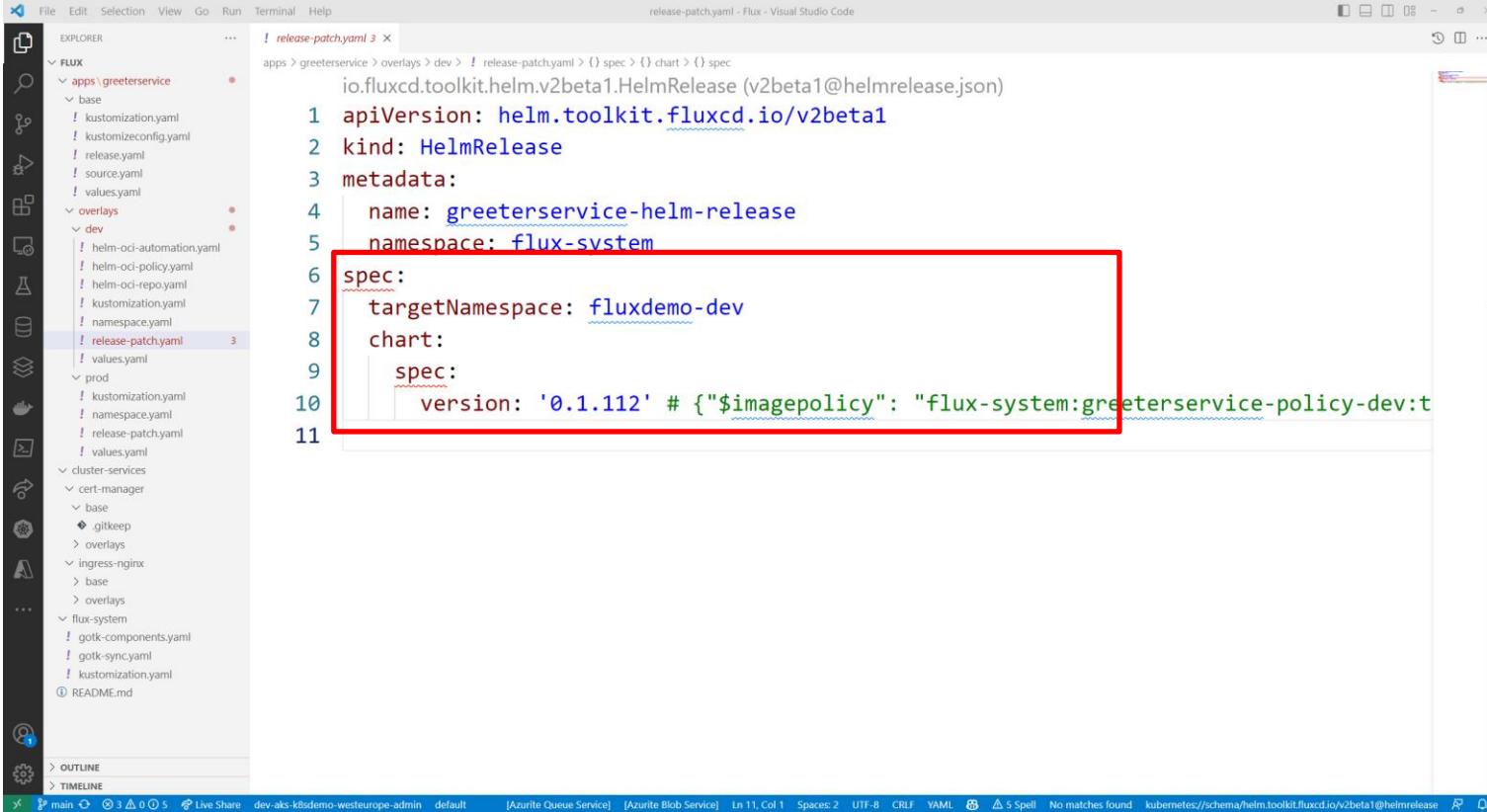
The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Explorer:** Shows a tree view of the project structure under the 'FLUX' category. The 'values.yaml' file is selected in the 'apps\greeter-service\overlays\dev' folder.
- Editor:** The file 'values.yaml' is open, displaying YAML configuration for a 'greeter' service. The code includes environment variables, ingress configurations, and a secret name.
- Bottom Status Bar:** Shows file paths (main, dev-aks-k8sdemo-west-europe-admin), default profile, and various status icons (Azurite Table Service, Azurite Queue Service, Azurite Blob Service, Line 3, Col 52, Spaces: 2, CRLF, YAML, 11 Spell, No matches found, No JSON Schema).

```
! values.yaml x
apps > greeter-service > overlays > dev > ! values.yaml > {} envvariables > servicesettings_message

1 envvariables:
2   servicesettings_EnablePrimeNumberCalculation: false
3   servicesettings_message: Hello from fluxdemo-dev
4 ingress:
5   hosts:
6     - greeter.fluxdemo-dev.aksdemo.4tecture.ch
7   tls:
8     - hosts:
9       - greeter.fluxdemo-dev.aksdemo.4tecture.ch
10  secretName: greeterservice-tls
```

Patch Release

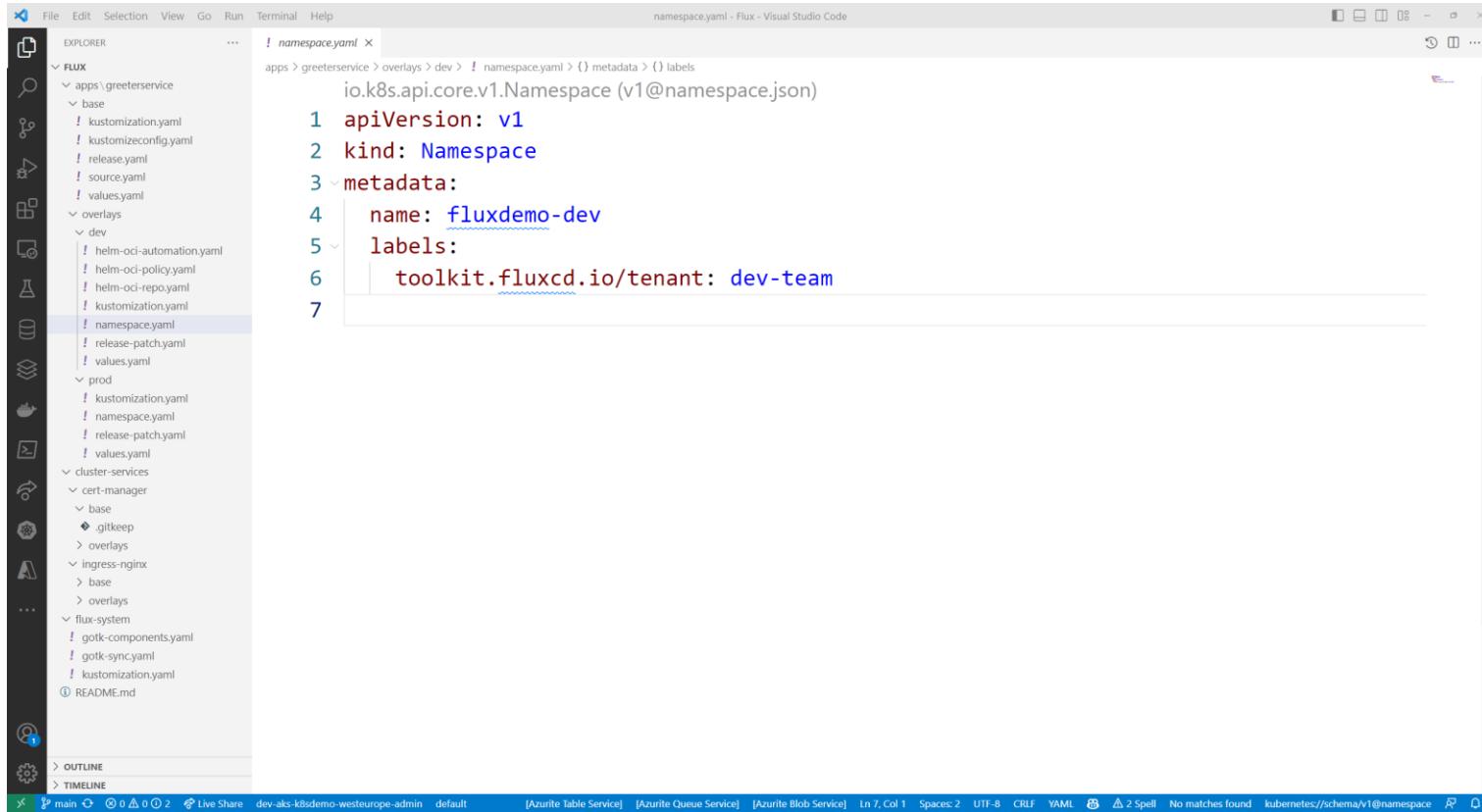


```
! release-patch.yaml 3 ×
apps > greeterservice > overlays > dev > ! release-patch.yaml > {} spec > {} chart > {} spec
io.fluxcd.toolkit.helm.v2beta1.HelmRelease (v2beta1@helmrelease.json)

1 apiVersion: helm.toolkit.fluxcd.io/v2beta1
2 kind: HelmRelease
3 metadata:
4   name: greeterservice-helm-release
5   namespace: flux-system
6 spec:
7   targetNamespace: fluxdemo-dev
8   chart:
9     spec:
10       version: '0.1.112' # {"$imagepolicy": "flux-system:greeterservice-policy-dev:t
11
```

The screenshot shows a Visual Studio Code window with the title "release-patch.yaml - Flux - Visual Studio Code". The left sidebar displays a file tree under the "EXPLORER" tab, showing a directory structure for "FLUX" with subfolders like "apps", "overlays", "dev", and "prod". The "release-patch.yaml" file is selected in the tree. The main editor area shows the YAML content of the file. A red rectangular box highlights the entire "spec:" block, which includes the "targetNamespace", "chart:", and "spec:" sections. The status bar at the bottom provides various file and system information.

Add dedicated objects



The screenshot shows a Visual Studio Code window with the file `namespace.yaml` open. The file content is as follows:

```
! namespace.yaml x
apps > greeterservice > overlays > dev > ! namespace.yaml > {} metadata > {} labels
io.k8s.api.core.v1.Namespace (v1@namespace.json)

1 apiVersion: v1
2 kind: Namespace
3 metadata:
4   name: fluxdemo-dev
5   labels:
6     toolkit.fluxcd.io/tenant: dev-team
7
```

The code editor interface includes a sidebar with icons for file operations like Open, Save, Find, and Replace, and sections for Explorer, Outline, and Timeline. The status bar at the bottom provides information about the file path, encoding, and other settings.



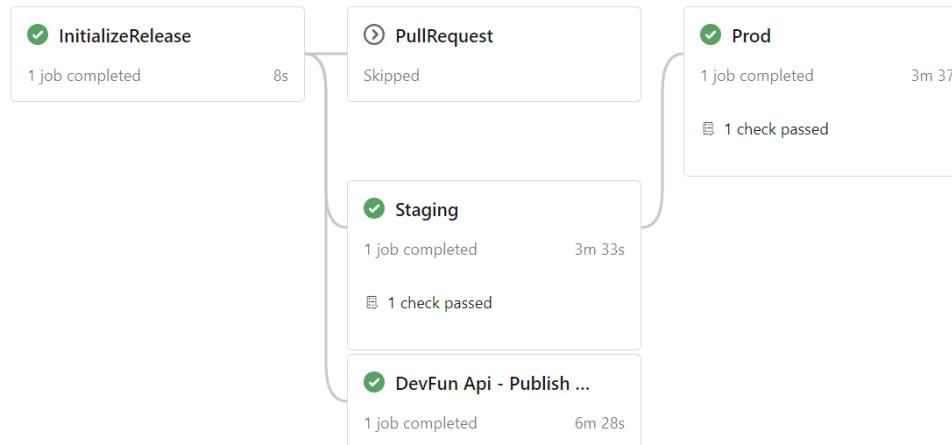
Automatic Updates

4tecture®
empower your software solutions

Azure Pipelines

Out-of-the box: triggers and branches

- Pipeline Triggers on Source Update
- Use a branching concept and PR workflow
- PR deployments
- Checks and Approvals



Flux

GitOps

- Git Repo defines what is deployed
- No update without new commit

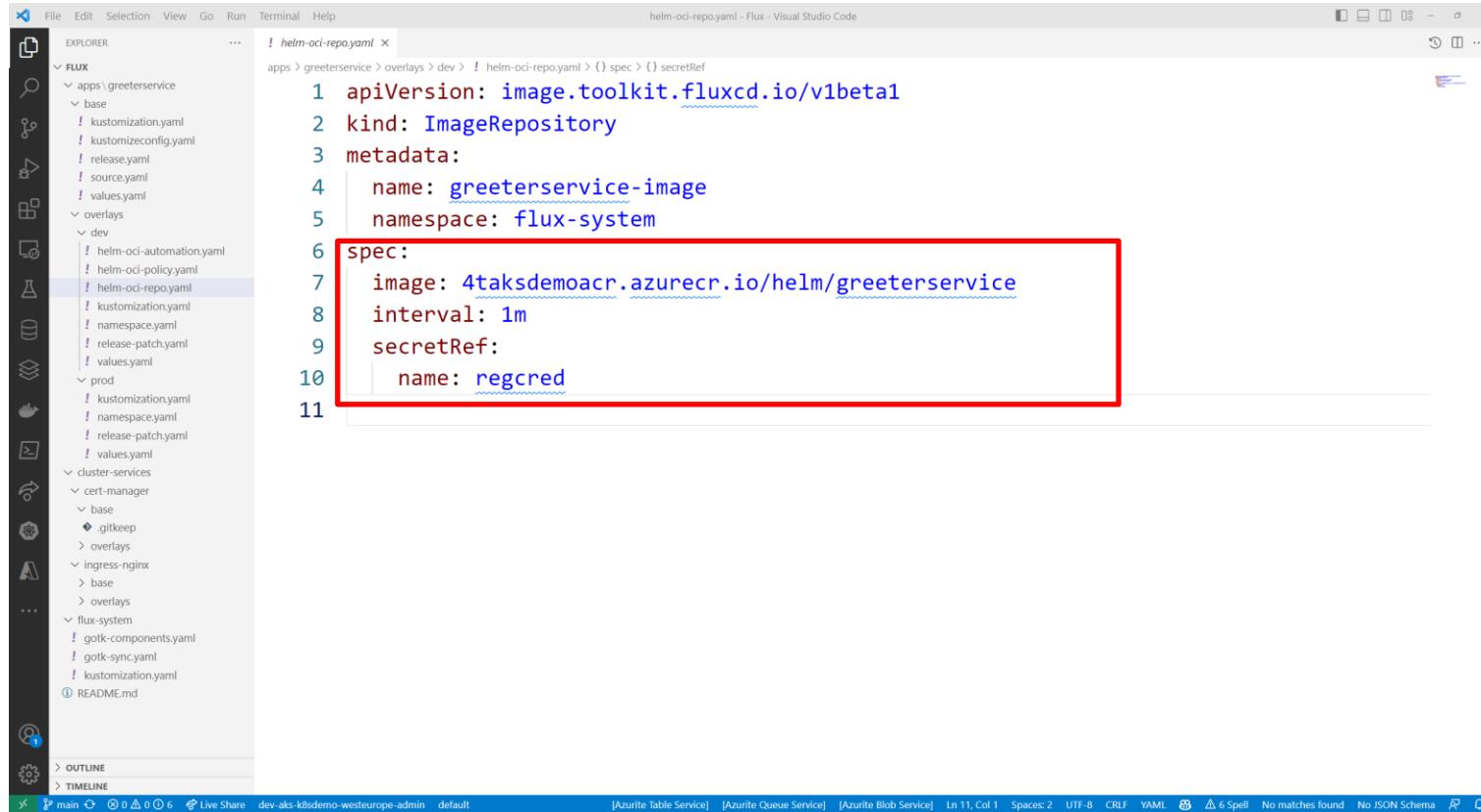
Repo Updates

- Task after successful CI
- Listen for new images / Helm charts

Flux Image Automation

- Monitor Container Registry
- Analyze updates based on policy
- Change configuration – commit change in GIT repo
- Standard GitOps flow is triggered

Image Repository



The screenshot shows a Visual Studio Code window with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** helm-oci-repo.yaml - Flux - Visual Studio Code.
- Explorer View:** Shows a tree structure under the FLUX category:
 - apps\greeterservice
 - base
 - kustomization.yaml
 - kustomizeconfig.yaml
 - release.yaml
 - source.yaml
 - values.yaml
 - overlays
 - dev
 - helm-oci-automation.yaml
 - helm-oci-policy.yaml
 - helm-oci-repo.yaml
 - kustomization.yaml
 - namespace.yaml
 - release-patch.yaml
 - values.yaml
 - prod
 - kustomization.yaml
 - namespace.yaml
 - release-patch.yaml
 - values.yaml
 - cluster-services
 - cert-manager
 - base
 - .gitkeep
 - overlays
 - ingress-nginx
 - base
 - overlays
 - flux-system
 - gotk-components.yaml
 - gotk-sync.yaml
 - kustomization.yaml
 - ... (Ellipsis)
 - Editor View:** The file content is a YAML configuration for an ImageRepository resource:

```
apiVersion: image.toolkit.fluxcd.io/v1beta1
kind: ImageRepository
metadata:
  name: greeterservice-image
  namespace: flux-system
spec:
  image: 4taksdemoacr.azurecr.io/helm/greeterservice
  interval: 1m
  secretRef:
    name: regcred
```

The spec block is highlighted with a red rectangle.
 - Bottom Status Bar:** Shows file paths (main, dev-aks-k8sdemo-westeurope-admin), default, and various status icons (e.g., Live Share, Spell Check).
 - Bottom Right:** Includes links to Azure services (Table Service, Queue Service, Blob Service), line numbers (Ln 11, Col 1), and other code editor settings (Spaces: 2, CRLF, YAML, Spell Check).

Image Policy

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows a project structure under the **FLUX** folder. The **apps/greeterservice** folder contains **base**, **overlays**, and **dev** subfolders. The **dev** folder contains files like **helm-oci-automation.yaml**, **helm-oci-policy.yaml**, etc.
- Editor (Center):** Displays the **helm-oci-policy.yaml** file content. The file is a YAML document defining an **ImagePolicy** named **greeterservice-policy** in the **flux-system** namespace. It specifies an **imageRepositoryRef** named **greeterservice-image-dev** and a **policy** section with a **semver** range of **0.1.x**. The **policy** section is highlighted with a red rectangle.
- Bottom Status Bar:** Shows the file path **dev-aks-k8sdemo-westeurope-admin_default/helm-oci-policy.yaml**, the line number **Ln 8, Col 35**, and the character count **Spaces: 2, UTR-B, CRLF, YAML**.

```
apiVersion: image.toolkit.fluxcd.io/v1beta1
kind: ImagePolicy
metadata:
  name: greeterservice-policy
  namespace: flux-system
spec:
  imageRepositoryRef:
    name: greeterservice-image-dev
  policy:
    semver:
      range: 0.1.x
```

Image Update Automation

The screenshot shows a Visual Studio Code window with the file `helm-oci-automation.yaml` open. The file is a `ImageUpdateAutomation` resource defined in the `image.toolkit.fluxcd.io/v1beta1` API version. The code is annotated with several sections highlighted by red boxes:

```
apiVersion: image.toolkit.fluxcd.io/v1beta1
kind: ImageUpdateAutomation
metadata:
  name: greeterservice-imageautomation
  namespace: flux-system
spec:
  interval: 30m
  sourceRef:
    kind: GitRepository
    name: flux-system
  git:
    checkout:
      ref:
        branch: main
    commit:
      author:
        email: fluxcdbot@4tecture.ch
        name: fluxcdbot
      messageTemplate: '{{range .Updated.Images}}{{println .}}{{end}}'
    push:
      branch: main
    update:
      path: ./apps/greeterservice
      strategy: Setters
```

The highlighted sections are:

- checkout:** Contains the `ref: branch: main` configuration.
- commit:** Contains the `author: email: fluxcdbot@4tecture.ch` and `messageTemplate: '{{range .Updated.Images}}{{println .}}{{end}}'` configurations.
- push:** Contains the `branch: main` configuration.
- update:** Contains the `path: ./apps/greeterservice` and `strategy: Setters` configurations.

The left sidebar shows the project structure under the `FLUX` folder, including `apps/greeterservice`, `cluster-services`, and `flux-system`. The bottom status bar shows the file path `dev-aks-k8sdemo-westeuropa-admin default` and various status icons.

Update markers

```
release-patch.yaml - Flux - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
FLUX
  apps\greeterservice
    base
      ! kustomization.yaml
      ! kustomizeconfig.yaml
      ! release.yaml
      ! source.yaml
      ! values.yaml
    overlays
      dev
        ! helm-oci-automation.yaml
        ! helm-oci-policy.yaml
        ! helm-oci-repo.yaml
        ! kustomization.yaml
        ! namespace.yaml
      release-patch.yaml
        ! values.yaml
      prod
        ! kustomization.yaml
        ! namespace.yaml
        ! release-patch.yaml
        ! values.yaml
    cluster-services
      cert-manager
        base
          .gitkeep
        overlays
        ingress-nginx
        base
        overlays
      flux-system
        ! gotk-components.yaml
        ! gotk-sync.yaml
        ! kustomization.yaml
      README.md
OUTLINE
TIMELINE
COMMENTS
release-patch.yaml
apps > greeterservice > overlays > dev > ! release-patch.yaml > {} spec > {} chart > {} spec
io.fluxcd.toolkit.helm.v2beta1.HelmRelease (v2beta1@helmrelease.json)
1 apiVersion: helm.toolkit.fluxcd.io/v2beta1
2 kind: HelmRelease
3 metadata:
4   name: greeterservice-helm-release
5   namespace: flux-system
6 spec:
7   targetNamespace: fluxdemo-dev
8   chart:
9     spec:
10       version: '0.1.112' # {"$imagepolicy": "flux-system:greeterservice-policy-dev:tag"}
11
```

Updates

The screenshot shows the Azure DevOps GitHub interface for a repository named 'k8sDemo'. The left sidebar navigation bar includes links for Overview, Boards, Repos, Files, Commits, Pushes, Branches, Tags, Pull requests, Advanced Security, Pipelines, Test Plans, Artifacts, and Project settings.

The main content area displays a commit history for the 'main' branch. The commits are listed in chronological order from top to bottom:

- 4taksdemoacr.azurecr.io/helm/greeterservice:0.1.112 (47655f20) - Marc Müller - Yesterday at 4:37 PM
- fixing suffixes for automation (46280ea9) - Marc Müller - Yesterday at 4:26 PM
- update dev kustomization (73415c55) - Marc Müller - Yesterday at 4:17 PM
- add image automation (81c6c778) - Marc Müller - Yesterday at 4:08 PM
- fix dev version (9e89fe32) - Marc Müller - Yesterday at 2:47 PM
- update dev (fc1d11e4) - Marc Müller - Yesterday at 2:44 PM
- update prod version (97a7fdd5) - Marc Müller - Yesterday at 2:41 PM
- update readme (e1bb2924) - Marc Müller - Yesterday at 2:36 PM
- add suffixes (bd3eea4f) - Marc Müller - Thu at 10:11 PM
- fix message (6cd9781a) - Marc Müller - Thu at 9:44 PM
- fix urls (ad295b32) - Marc Müller - Thu at 9:44 PM
- fix namespaces (0e27f32e) - Marc Müller - Thu at 9:40 PM
- fix naming in patch (7677674) - Marc Müller - Thu at 9:38 PM

A red box highlights the first commit, which corresponds to the commit shown in the commit status bar at the bottom of the screen.

Updates

Azure DevOps 4tecture-demo / k8sDemo / Repos / Commits / Flux

Search

k8sDemo

Overview Boards

Repos Files

Commits Pushes Branches Tags Pull requests Advanced Security

Pipelines Test Plans Artifacts

Project settings

4taksdemoacr.azurecr.io/helm/greeterservice:0.1.112

47655f20 fluxdbot committed Yesterday main

Files Details

Parent 1 → This commit Filter 1 changed file

Side-by-side View

Flux

apps/greeterservice/overlays/dev

release-patch.yaml

release-patch.yaml -1+2

/apps/greeterservice/overlays/dev/release-patch.yaml

```
targetNamespace: fluxdemo-dev
chart:
spec:
version: '0.1.111' # {"imagepolicy": "flux-system:greeterservice"}
```

```
targetNamespace: fluxdemo-dev
chart:
spec:
version: '0.1.112' # {"imagepolicy": "flux-system:greeterservice"}
```

The screenshot shows a commit detail page in Azure DevOps for a repository named 'k8sDemo'. The commit is titled '4taksdemoacr.azurecr.io/helm/greeterservice:0.1.112' and was made by 'fluxdbot' yesterday. The 'Details' tab is selected, showing a single changed file, 'release-patch.yaml'. The file is located at '/apps/greeterservice/overlays/dev/release-patch.yaml'. The diff view shows a change from version '0.1.111' to '0.1.112', with the new version highlighted in green. The commit message includes a comment about the image policy: '# {"imagepolicy": "flux-system:greeterservice"}'.

A close-up, low-angle shot of several rowers in a racing shell. Their hands are gripping yellow oars with black blades, which are partially submerged in water. The rowers are wearing blue and red athletic gear. The background is blurred, suggesting motion on a body of water.

Cluster Setup with Azure Pipelines

4tecture®
empower your software solutions

Deploy Cluster Services

- Cluster setup has a lot of Helm chart deployments
 - Configuration as Code
 - Pipeline automation on configuration object
-
- Runtime Parameters / Parameters support objects
 - Define your cluster configuration as object

Parameter with all charts

Azure DevOps 4tecture-demo / k8sDemo / Pipelines

k8sDemo +

Overview Boards Repos Pipelines Pipelines Environments Releases Library Task groups Deployment groups Test Plans Artifacts

← AzureK8SInfrastructure

master Infrastructure / azure-pipelines.yml *

Variables Save ...

Show assistant

```
28 parameters:
29   - name: ChartsToInstall
30     type: object
31     default:
32       - name: cert-manager
33         chartUrl: https://charts.jetstack.io
34         repoName: cert-manager
35         version: v1.12.2
36         namespace: cert-manager
37         valuesFile: cert-manager.yaml
38       - name: aad-pod-identity
39         chartUrl: https://raw.githubusercontent.com/Azure/aad-pod-identity/master/charts
40         repoName: aad-pod-identity
41         version: 4.1.18
42         namespace: kube-system
43         valuesFile:
44       - name: ingress-nginx
45         chartUrl: https://kubernetes.github.io/ingress-nginx
46         repoName: ingress-nginx
47         version: 4.7.0
48         namespace: ingress-nginx
49         valuesFile: ingress-nginx.yaml
50       - name: redis
51         chartUrl: https://charts.bitnami.com/bitnami
52         repoName: bitnami
53         version: 17.11.6
54         namespace: redis
55         valuesFile: redis.yaml
56       - name: dapr
57         chartUrl: https://dapr.github.io/helm-charts/
58         repoName: dapr
59         version: 1.11.0
60         namespace: dapr-system
61         valuesFile: dapr.yaml
62       - name: loki-stack
63         chartUrl: https://grafana.github.io/helm-charts
```

Project settings <>



Configuration as Code

The screenshot shows the Azure DevOps interface for a repository named 'k8sDemo'. The left sidebar is the navigation menu, and the main area is the code editor for the 'ingress-nginx.yml' file.

Repository Structure:

- Root: k8sDemo
- Overview
- Boards
- Repos** (selected)
- Commits
- Pushes
- Branches
- Tags
- Pull requests
- Advanced Security
- Pipelines
- Test Plans
- Artifacts

File Structure:

- Infrastructure
 - azure
 - kubernetes-config
 - charts
 - manifests
 - valuesfiles
 - cert-manager.yaml
 - dapr.yaml
 - ingress-nginx.yaml** (selected)
 - loki-stack.yaml
 - prometheus-adapter.yaml
 - redis.yaml
 - linuxworker
 - scripts
 - .gitignore
 - azure-pipelines.yaml
- README.md

Code Editor Content:

```
1 controller:
2   # Will add custom configuration options to Nginx https://kubernetes.github.io/ingress-nginx/user-guide/nginx-configuration/configuring/
3   config:
4     proxy-buffers: "8 64k"
5     proxy-buffer-size: 64k
6     http2-max-field-size: 64k
7     http2-max-header-size: 64k
8     large-client-header-buffers: "4 64k"
9
10  service:
11    annotations:
12      | service.beta.kubernetes.io/azure-load-balancer-health-probe-request-path: /healthz
13
14  ## Annotations to be added to controller pods
15  ##
16  podAnnotations:
17    | prometheus.io/scrape: true
18    | prometheus.io/port: 10254
19
20  replicaCount: 2
21
22  ## Node labels for controller pod assignment
23  ## Ref: https://kubernetes.io/docs/user-guide/node-selection/
24  ##
25  nodeSelector:
26    | kubernetes.io/os: linux
27
28  admissionWebhooks:
29    patch:
30      nodeSelector:
31        | kubernetes.io/os: linux
32
33  metrics:
34    enabled: true
```

Add and update Helm Repos

The screenshot shows the Azure DevOps Pipeline Editor interface. On the left, a sidebar lists project navigation options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The 'Pipelines' option is selected. The main area displays a YAML pipeline configuration file named 'azure-pipelines.yml'. A search bar and various UI controls are visible at the top right.

The pipeline file contains two sections highlighted with red boxes:

```
263     - ${{ each.chart.in.parametersChartsToInstall }}:
264       task: HelmDeploy@0
265       displayName: "Helm Repo Add - ${chart.name}"
266       inputs:
267         connectionType: 'Azure Resource Manager'
268         azureSubscription: '${{ AzureSubscription }}'
269         azureResourceGroup: '${{ AksResourceGroupName }}'
270         kubernetesCluster: '${{ AksClusterName }}'
271         useClusterAdmin: true
272         command: 'repo'
273         arguments: 'add ${chart.repoName} ${chart.chartUrl}'
274
275
276     - task: HelmDeploy@0
277       displayName: "Helm Repo Update"
278       inputs:
279         connectionType: 'Azure Resource Manager'
280         azureSubscription: '${{ AzureSubscription }}'
281         azureResourceGroup: '${{ AksResourceGroupName }}'
282         kubernetesCluster: '${{ AksClusterName }}'
283         useClusterAdmin: true
284         command: 'repo'
285         arguments: 'update'
286
287     task: Kubernetes@1
```

The first section (lines 263-274) defines a task to add a Helm repository. It specifies the task type as 'HelmDeploy@0', a display name, and various inputs including connection type ('Azure Resource Manager'), subscription ('\${{ AzureSubscription }}'), resource group ('\${{ AksResourceGroupName }}'), cluster ('\${{ AksClusterName }}'), and command ('repo'). The arguments are set to 'add \${chart.repoName} \${chart.chartUrl}'.

The second section (lines 276-285) defines a similar task to update the Helm repository. It uses the same parameters and command ('repo') but with the argument 'update'.

Install helm charts

The screenshot shows the Azure DevOps Pipeline Editor interface. On the left, there's a sidebar with project navigation (k8sDemo), pipeline management (Pipelines, Pipelines, Environments, Releases, Library, Task groups, Deployment groups), test plans (Test Plans), and artifacts (Artifacts). The main area displays the 'Infrastructure / azure-pipelines.yml' file for the 'master' branch of the 'AzureK8SInfrastructure' pipeline. A red box highlights the Helm deployment task configuration. The code snippet below shows the task definition and its inputs.

```
299    - ${{ each chart in parametersChartsToInstall }}:
300      task: HelmDeploy@0
301      displayName: "Install ${{ chart.name }}"
302      inputs:
303        connectionType: 'Azure Resource Manager'
304        azureSubscription: '${{ AzureSubscription }}'
305        azureResourceGroup: '${{ AksResourceGroupName }}'
306        kubernetesCluster: '${{ AksClusterName }}'
307        useClusterAdmin: true
308        command: 'upgrade'
309        chartType: 'Name'
310        chartName: '${{ chart.repoName }}/${{ chart.name }}'
311        chartVersion: '${{ chart.version }}'
312        releaseName: '${{ chart.name }}'
313        ${{ if chart.valuesFile }}:
314          valueFile: ${Build.SourcesDirectory}/kubernetes-config/valuesfiles/${{ chart.valuesFile }}
315        namespace: ${{ chart.namespace }}
316        arguments: '--create-namespace'
317
318      pwsh: |
319        $trainingWorkEnvironmentsJson = @"
320        ${{ convertToJson(parameters.TrainingWorkEnvironment) }}
321        @@
322        $trainingWorkEnvironments = $trainingWorkEnvironmentsJson | ConvertFrom-Json
323        $index = 0
324        $setValues = ""
325        foreach($workEnv in $trainingWorkEnvironments){
```

Conclusion



4tecture®
empower your software solutions

Comparison

Pipelines

- Everything in Git
- Versatile
- Full DevOps Lifecycle
- End-to-end Traceability
- Integrated Test Automation

Flux

- Everything in Git
- K8s Deployments
- Focus on Continuous Deployment
- Independent / Various Sources

Conclusion

- Both provide a state-of-the-art deployment
- Git is the source of truth
- We use
 - Azure Pipelines: Classic Development to Deployment (CI/CD)
 - Flux: disconnected from development / prod cluster deployment



Q & A

4tecture®
empower your software solutions

Thank you for your attention!

If you have any questions do not hesitate to contact us:

4ecture GmbH
Industriestrasse 25
CH-8604 Volketswil

+41 44 508 37 00
info@4ecture.ch
www.4ecture.ch

Marc Müller
Principal Consultant

marc.mueller@4ecture.ch
[@muellermarc](https://twitter.com/muellermarc)
www.powerofdevops.com

A close-up photograph of several hands reaching towards a central wooden puzzle piece. The puzzle piece is light-colored wood with a dark green and red section. The hands belong to different people, suggesting collaboration. The background is blurred.

4 tecture[©]

empower your software solutions