



Day 2

Day 2

- Github Actions
- Kustomize
- Advanced deployment
- Storage

Day 2



Day 2



(c) <https://aws.plainenglish.io/kubernetes-deployments-for-micro-service-applications-fa3236592c9f>

Github Actions & Openshift Deployment Lab

- Register / Login Github <https://github.com/>
- Config:
 - Go to Github
 - Create public repository
 - Actions -> new Workflow -> set up a workflow

Github Actions Example

```
name: Simple Echo and List Workflow with Git Pull

on:
  push:
    branches:
      - main

jobs:
  echo-and-list-job:
    runs-on: ubuntu-latest
    steps:
      - name: Set up Git
        run: |
          sudo apt-get update
          sudo apt-get install -y git

      - name: Clone Repository using Git
        run: |
          git clone https://github.com/${{ github.repository }}.git
          cd $(basename ${{ github.repository }})
          git pull

      - name: Echo Hello World
        run: echo "Hello, World!"

      - name: List Files
        run: ls *
```

Github Actions Example

```
name: Simple Echo and List Workflow with Git Pull

on:
  push:
    branches:
      - main

jobs:
  echo-and-list-job:
    runs-on: ubuntu-latest
    steps:
      - name: Set up Git
        run: |
          sudo apt-get update
          sudo apt-get install -y git

      - name: Clone Repository using Git
        run: |
          git clone https://github.com/${{ github.repository }}.git
          cd $(basename ${{ github.repository }})
          git pull

      - name: Echo Hello World
        run: echo "Hello, World!"

      - name: List Files
        run: ls *
```

```
name: Simple Echo and List Workflow

on:
  push:
    branches:
      - main

jobs:
  echo-and-list-job:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout Repository
        uses: actions/checkout@v4

      - name: Echo Hello World
        run: echo "Hello, World!"

      - name: List Files
        run: ls *
```

Github Actions Example

```
name: Simple Echo and List Workflow with Git Pull

on:
  push:
    branches:
      - main

jobs:
  echo-and-list-job:
    runs-on: ubuntu-latest
    steps:
      - name: Set up Git
        run: |
          sudo apt-get update
          sudo apt-get install -y git

      - name: Clone Repository using Git
        run: |
          git clone https://github.com/${{ github.repository }}.git
          cd $(basename ${{ github.repository }})
          git pull

      - name: Echo Hello World
        run: echo "Hello, World!"

      - name: List Files
        run: ls *
```

```
name: Simple Echo and List Workflow

on:
  push:
    branches:
      - main

jobs:
  echo-and-list-job:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout Repository
        uses: actions/checkout@v4

      - name: Echo Hello World
        run: echo "Hello, World!"

      - name: List Files
        run: ls *
```

<https://github.com/marketplace>

Actions Example Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/0.workflow
```

- Actions -> new Workflow -> set up a workflow
- Copy & paste helloworld.yaml
- Commit

Actions Example Lab

- Actions -> new Workflow -> set up a workflow
- Copy & paste helloworld.yaml
- Commit

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/0.workflow
```

```
name: Hello World

on:
  push:
    branches:
      - main

jobs:
  hello:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout Repository
        uses: actions/checkout@v4

      - name: Say Hello
        run: echo "Hello, World!"

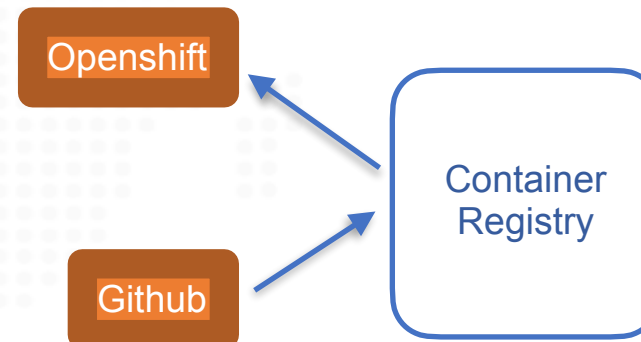
      - name: list files
        run: ls -a
```

Github Actions Container Registry

- Github Container Registry -> create Github Packages-Token
 - User-Settings -> Developer -> PAT -> Classic

```
oc create secret docker-registry ghcr-pull-secret \  
--docker-server=ghcr.io \  
--docker-username=GITHUB_USERNAME \  
--docker-password=GITHUB_TOKEN \  
--docker-email=GITHUB_EMAIL
```

- Create Repository Secret:
 - Go to GitHub repository -> Settings
-> Secrets and Variables „Actions“
 - GHCR_TOKEN



Actions Container Image Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/0.workflow
```

- Actions -> new Workflow -> set up a workflow
- Copy & paste dockerbasic.yaml
- Set correct Dockerfile path:
e.g. ./Day1/container/Dockerfile
- Commit

Actions Container Image Lab

- Actions -> new Workflow -> set up a workflow
- Copy & paste dockerbasic.yaml
- Set correct Dockerfile path:
e.g. ./Day1/container/Dockerfile
- Commit

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/0.workflow
```

```
name: Dockerbasic

env:
...
jobs:

  docker-basic:
    name: Build Dockerimage
    runs-on: ubuntu-latest
    environment: production

  ...
  - name: Build from Dockerfile
    id: build-image
    uses: redhat-actions/buildah-build@v2
    with:
      image: ${ env.APP_NAME }
      tags: ${ env.IMAGE_TAGS }

    dockerfiles: |
      ./Dockerfile
  ...
```


Github Actions vs Openshift

- Login Openshift <https://developers.redhat.com/developer-sandbox>
- Go to Openshift dashboard „copy login command“
 - Copy the data
 - Go to GitHub repository -> Settings -> Secrets and Variables „Actions“
- Create Repository Secrets:
 - OPENSIFT_SERVER
 - OPENSIFT_NAMESPACE
 - OPENSIFT_TOKEN

Actions deployment Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/0.workflow
```

- Actions -> new Workflow -> set up a workflow
- Copy & paste openshift.yaml
- Commit

Actions deployment Lab

git clone <https://github.com/kolinrr/openshift.git>

cd openshift/Day2/0.workflow

- Actions -> new Workflow -> set up a workflow
- Copy & paste openshift.yaml
- Commit

```
...  
- name: my deployment  
  run: |  
    export IMAGE=${{ steps.push-image.outputs.registry-path }}  
    export OPENSIFT_NAMESPACE=${{ env.OPENSIFT_NAMESPACE }}  
  
    echo ${{ steps.push-image.outputs.registry-path }}  
  
    sed -i 's|IMAGE_PLACEHOLDER|"$IMAGE"|g' Day2/deployment/deployment.yaml  
  
    oc apply -f Day2/deployment/deployment.yaml -n $OPENSIFT_NAMESPACE  
    oc apply -f Day1/network/service.yaml -n $OPENSIFT_NAMESPACE  
  
    oc get pods  
    oc get svc  
...
```



Lunch Break: 30 minutes

Kustomize 1

```
git clone https://github.com/kolnrr/openshift.git
```

```
cd openshift/Day2/1.kustomize
```

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```



```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```


Kustomize 1

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```



```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```

```
git clone https://github.com/kolinnr/openshift.git
```

```
cd openshift/Day2/1.kustomize
```

```
my-kustomize-project/
├── base/
│   ├── configmap.yaml
│   └── kustomization.yaml
├── overlays/
│   ├── prod/
│   │   └── kustomization.yaml
│   └── test/
│       └── kustomization.yaml
```

Kustomize 1

```
git clone https://github.com/kolnrr/openshift.git
```

```
cd openshift/Day2/1.kustomize
```

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```

```
#base/kustomization.yaml
resources:
- configmap.yaml
```

```
my-kustomize-project/
├── base/
│   ├── configmap.yaml
│   └── kustomization.yaml
├── overlays/
│   ├── prod/
│   │   └── kustomization.yaml
│   └── test/
│       └── kustomization.yaml
```

Kustomize 1

```
git clone https://github.com/kolnrr/openshift.git
```

```
cd openshift/Day2/1.kustomize
```

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```

```
#base/kustomization.yaml
resources:
- configmap.yaml
```

```
#overlays/prod/kustomization.yaml
bases:
- ../../base
```

```
my-kustomize-project/
├── base/
│   ├── configmap.yaml
│   └── kustomization.yaml
└── overlays/
    ├── prod/
│   │   └── kustomization.yaml
    └── test/
        └── kustomization.yaml
```

Kustomize 1

```
git clone https://github.com/kolinnr/openshift.git
```

```
cd openshift/Day2/1.kustomize
```

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```

```
#base/kustomization.yaml
resources:
- configmap.yaml
```

```
#overlays/prod/kustomization.yaml
bases:
- ../../base
```

```
#overlays/test/kustomization.yaml
bases:
- ../../base
nameSuffix: „-test"
```

```
my-kustomize-project/
├── base/
│   ├── configmap.yaml
│   └── kustomization.yaml
├── overlays/
│   ├── prod/
│   │   └── kustomization.yaml
│   └── test/
│       └── kustomization.yaml
```

Kustomize 1

git clone https://github.com/kolnrr/openshift.git

cd openshift/Day2/1.kustomize

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```

```
#base/kustomization.yaml
resources:
- configmap.yaml
```

kustomize build overlays/prod

```
#overlays/prod/kustomization.yaml
bases:
- ../../base
```

```
#overlays/test/kustomization.yaml
bases:
- ../../base
nameSuffix: „-test"
```

```
my-kustomize-project/
├── base/
│   ├── configmap.yaml
│   └── kustomization.yaml
├── overlays/
│   ├── prod/
│   │   └── kustomization.yaml
│   └── test/
│       └── kustomization.yaml
```


Kustomize 1

git clone https://github.com/kolnrr/openshift.git

cd openshift/Day2/1.kustomize

- Template manipulation of objects

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config
data:
  ENV: "prod"
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: app-config-test
data:
  ENV: "prod"
```

```
#base/kustomization.yaml
resources:
- configmap.yaml
```

kustomize build overlays/prod

kustomize build overlays/test

```
#overlays/prod/kustomization.yaml
bases:
- ../../base
```

```
#overlays/test/kustomization.yaml
bases:
- ../../base
nameSuffix: „-test"
```

```
my-kustomize-project/
├── base/
│   ├── configmap.yaml
│   └── kustomization.yaml
├── overlays/
│   ├── prod/
│   │   └── kustomization.yaml
│   └── test/
│       └── kustomization.yaml
```

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/2.kustomize
```

Kustomize 2

Extended template manipulation of objects

- Deploy 2 more application for prod- and test-environment
With different resource limits

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/2.kustomize
```

Kustomize 2

Extended template manipulation of objects

- Deploy 2 more application for prod- and test-environment
- Resource limits

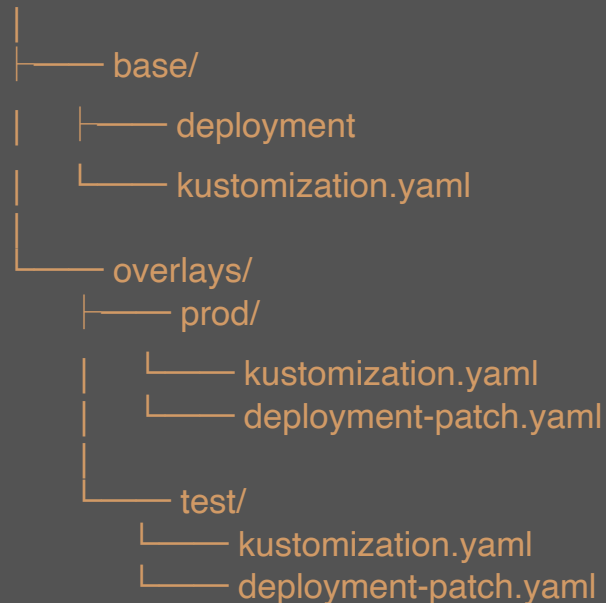
```
3.kustomize/  
├── base/  
│   ├── deployment  
│   └── kustomization.yaml  
└── overlays/  
    ├── prod/  
    │   ├── kustomization.yaml  
    │   └── deployment-patch.yaml  
    └── test/  
        ├── kustomization.yaml  
        └── deployment-patch.yaml
```

Kustomize 2

Extended template manifest

- Deploy 2 more applications to prod- and test-environment

3.kustomize/



```
#base/deployment.yaml
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_TCP_PORT
              value: "3306"
          ports:
            - containerPort: 3306
          imagePullPolicy: IfNotPresent
```

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/2.kustomize
```

Kustomize 2

Extended template manifest

- Deploy 2 more applications in prod- and test-environment

```
3.kustomize/
├── base/
│   ├── deployment
│   └── kustomization.yaml
├── overlays/
│   ├── prod/
│   │   ├── kustomization.yaml
│   │   └── deployment-patch.yaml
│   └── test/
│       ├── kustomization.yaml
│       └── deployment-patch.yaml
```

```
#base/deployment.yaml

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_TCP_PORT
              value: "3306"
          ports:
            - containerPort: 3306
          imagePullPolicy: IfNotPresent
```

```
#base/kustomization.yaml

resources:
- deployment.yaml
```

git clone <https://github.com/kolinrr/openshift.git>

cd openshift/Day2/2.kustomize

Kustomize 2

Extended template manifest

- Deploy 2 more applications in prod- and test-environment

```
3.kustomize/  
├── base/  
│   ├── deployment  
│   └── kustomization.yaml  
└── overlays/  
    ├── prod/  
    │   ├── kustomization.yaml  
    │   └── deployment-patch.yaml  
    └── test/  
        ├── kustomization.yaml  
        └── deployment-patch.yaml
```

```
#base/deployment.yaml  
  
apiVersion: apps/v1  
kind: StatefulSet  
metadata:  
  name: mysql  
spec:  
  serviceName: mysql  
  replicas: 1  
  selector:  
    matchLabels:  
      app: mysql  
  template:  
    metadata:  
      labels:  
        app: mysql  
    spec:  
      containers:  
        - name: mysql  
          image: mysql:latest  
          env:  
            - name: MYSQL_TCP_PORT  
              value: "3306"  
          ports:  
            - containerPort: 3306  
          imagePullPolicy: IfNotPresent
```

```
#base/kustomization.yaml  
  
resources:  
  - deployment.yaml
```

git clone <https://github.com/kolinrr/openshift.git>

cd openshift/Day2/2.kustomize

```
#base/overlays/prod/deployment-patch.yaml  
  
apiVersion: apps/v1  
kind: StatefulSet  
metadata:  
  name: mysql  
spec:  
  template:  
    spec:  
      containers:  
        - name: mysql  
          resources:  
            requests:  
              cpu: "500m"  
              memory: "512Mi"  
            limits:  
              cpu: "1000m"  
              memory: "1024Mi"
```

Kustomize 2

Extended template manifest

- Deploy 2 more applications in prod- and test-environment

```
3.kustomize/
├── base/
│   ├── deployment
│   └── kustomization.yaml
└── overlays/
    ├── prod/
    │   ├── kustomization.yaml
    │   └── deployment-patch.yaml
    └── test/
        ├── kustomization.yaml
        └── deployment-patch.yaml
```

```
#base/deployment.yaml

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_TCP_PORT
              value: "3306"
          ports:
            - containerPort: 3306
          imagePullPolicy: IfNotPresent
```

```
#base/kustomization.yaml

resources:
- deployment.yaml
```

```
git clone https://github.com/kolinrr/openshift.git
cd openshift/Day2/2.kustomize
```

```
#base/overlays/prod/deployment-patch.yaml

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  template:
    spec:
      containers:
        - name: mysql
          resources:
            requests:
              cpu: "500m"
              memory: "512Mi"
            limits:
              cpu: "1000m"
              memory: "1024Mi"
```

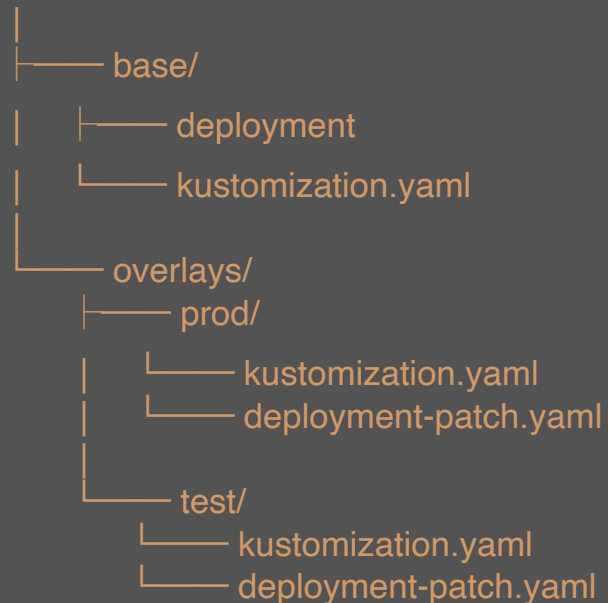
```
#overlays/test/kustomization.yaml

resources:
- ../../base

apiVersion: kustomize.config.k8s.io/v1alpha1
kind: Kustomization
patches:
- path: deployment-patch.yaml
```

Kustomize 2

3.kustomize/



```
#base/deployment.yaml
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_TCP_PORT
              value: "3306"
          ports:
            - containerPort: 3306
          imagePullPolicy: IfNotPresent
```

```
#base/kustomization.yaml
```

```
resources:
  - deployment.yaml
```

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/2.kustomize
```

```
#base/overlays/prod/deployment-patch.yaml
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  template:
    spec:
      containers:
        - name: mysql
          resources:
            requests:
              cpu: "500m"
              memory: "512Mi"
            limits:
              cpu: "1000m"
              memory: "1024Mi"
```

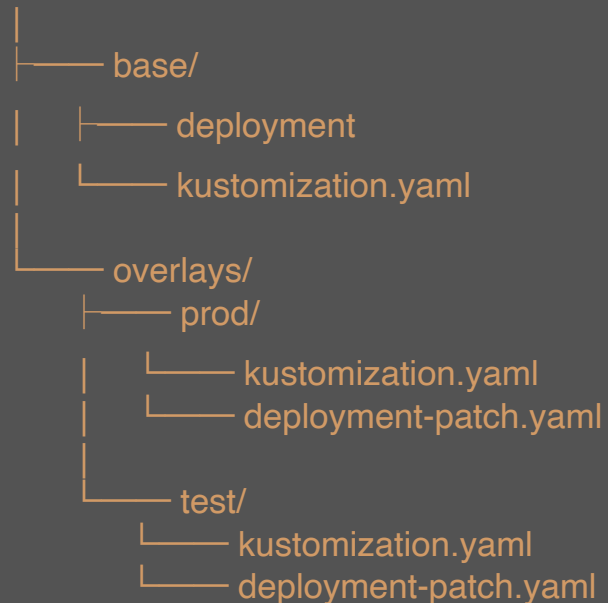
```
#overlays/test/kustomization.yaml
```

```
resources:
  - ../../base
```

```
apiVersion: kustomize.config.k8s.io/
kind: Kustomization
patches:
  - path: deployment-patch.yaml
```

Kustomize 2

3.kustomize/



```
#base/deployment.yaml
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_TCP_PORT
              value: "3306"
          ports:
            - containerPort: 3306
          imagePullPolicy: IfNotPresent
```

kustomize build overlays/prod

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/2.kustomize
```

```
#base/overlays/prod/deployment-patch.yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  template:
    spec:
      containers:
        - name: mysql
          resources:
            requests:
              cpu: "500m"
              memory: "512Mi"
            limits:
              cpu: "1000m"
              memory: "1024Mi"
```

```
#base/kustomization.yaml
```

```
resources:
  - deployment.yaml
```

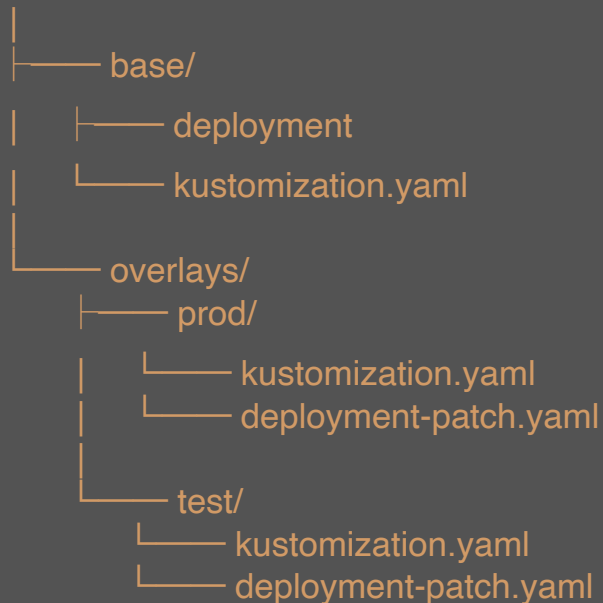
```
#overlays/test/kustomization.yaml
```

```
resources:
  - ../../base
```

```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
patches:
  - path: deployment-patch.yaml
```

Kustomize 2

3.kustomize/



```
#base/deployment.yaml
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          env:
            - name: MYSQL_TCP_PORT
              value: "3306"
          ports:
            - containerPort: 3306
          imagePullPolicy: IfNotPresent
```

kustomize build overlays/prod

kustomize build overlays/test

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/2.kustomize
```

```
#base/overlays/prod/deployment-patch.yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  template:
    spec:
      containers:
        - name: mysql
          resources:
            requests:
              cpu: "500m"
              memory: "512Mi"
            limits:
              cpu: "1000m"
              memory: "1024Mi"
```

```
#base/kustomization.yaml
```

```
resources:
  - deployment.yaml
```

```
#overlays/test/kustomization.yaml
```

```
resources:
  - ../../base
```

```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
patches:
  - path: deployment-patch.yaml
```

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/3.kustomize
```

Kustomize 3

Extended template manipulation of objects

- Deploy 2 more application for prod- and test-environment with different configmaps and secrets

Kustomize 3

```
git clone https://github.com/kolnrr/openshift.git
```

```
cd openshift/Day2/3.kustomize
```


Kustomize 3

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/3.kustomize
```

```
kustomize build overlays/prod
```

Kustomize 3

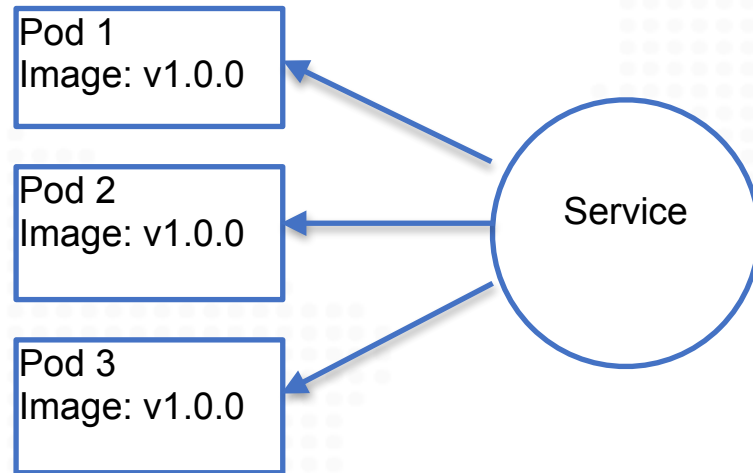
```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/3.kustomize
```

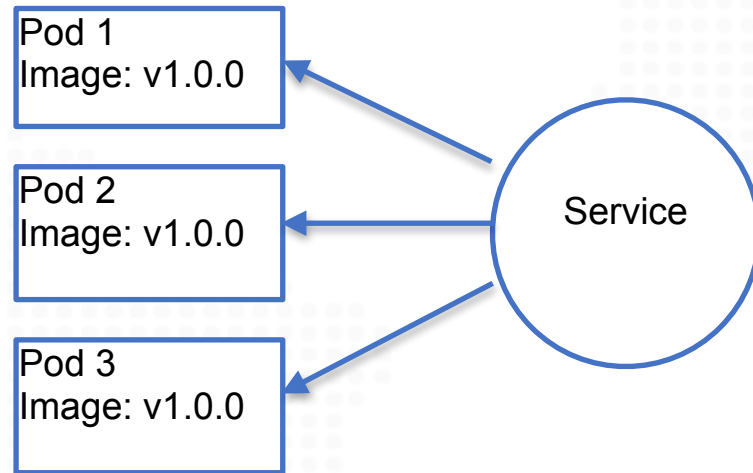
```
kustomize build overlays/prod
```

```
kustomize build overlays/test
```

Rolling updates

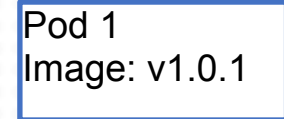
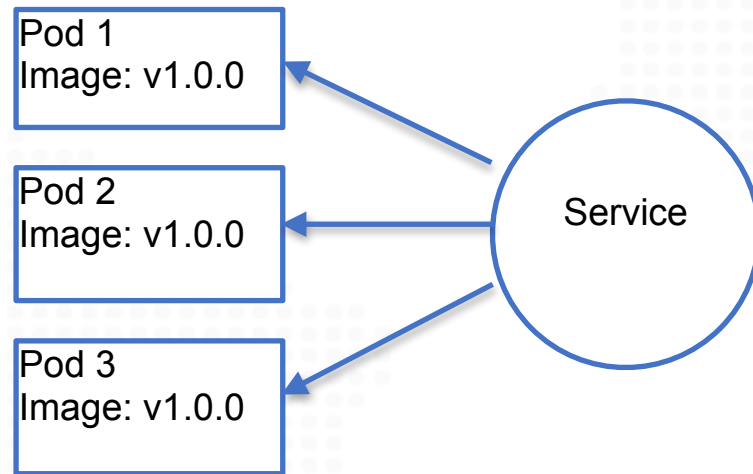


Rolling updates



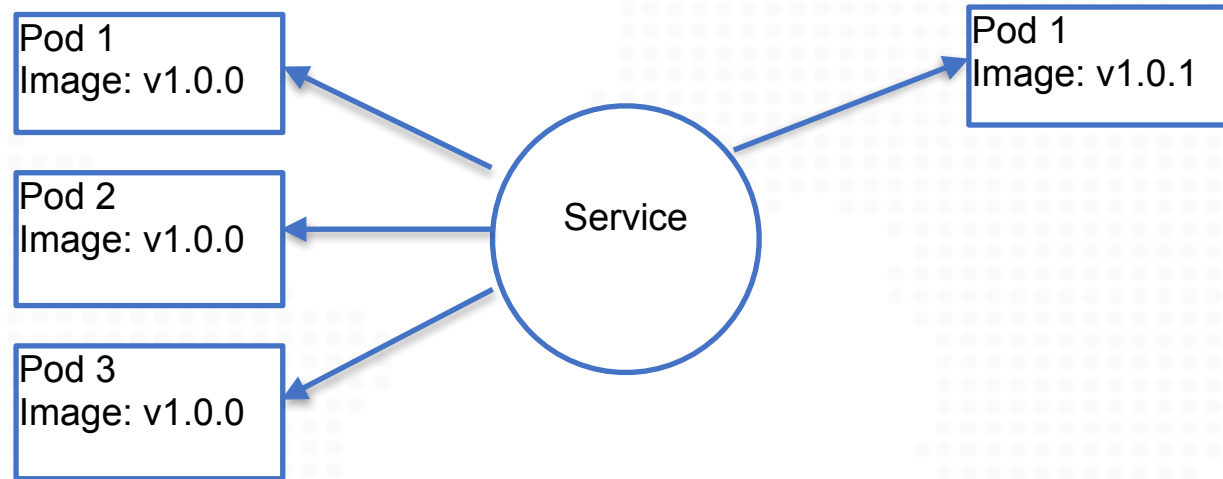
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



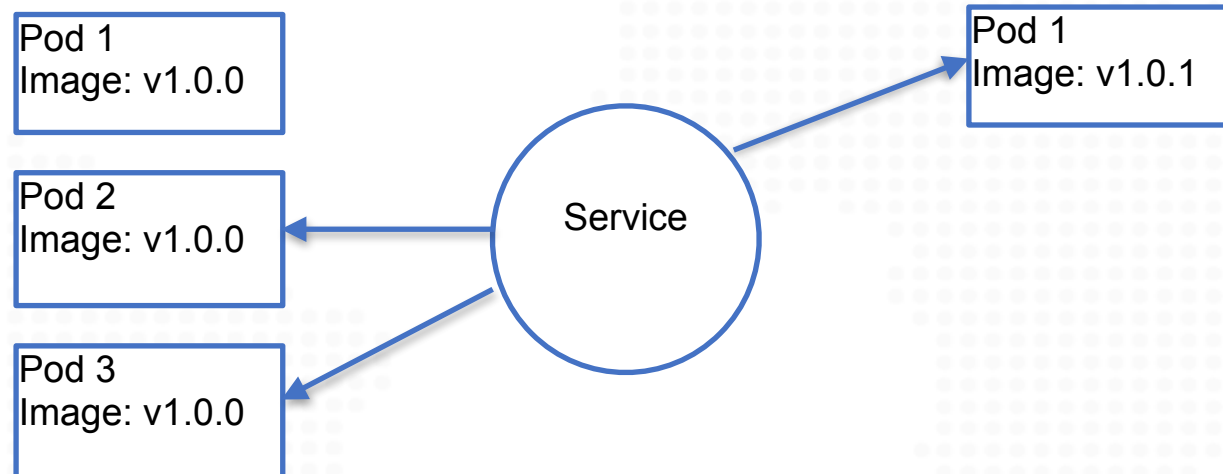
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



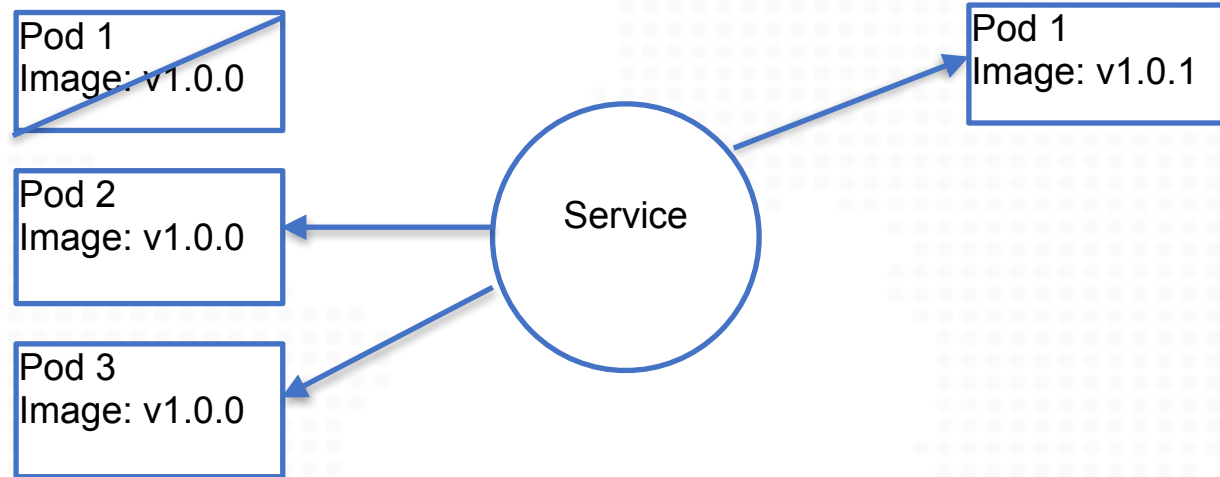
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



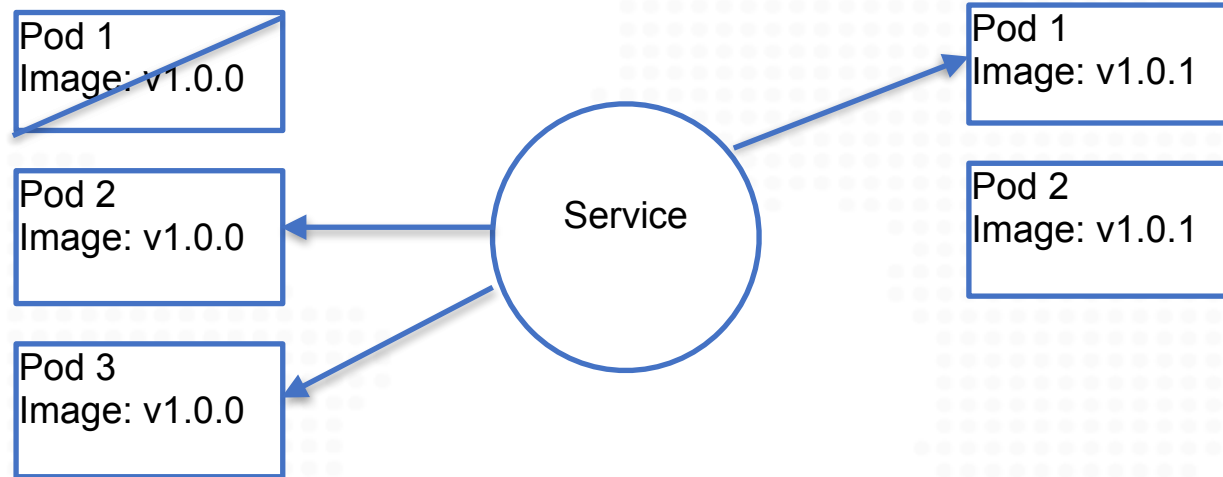
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```


Rolling updates



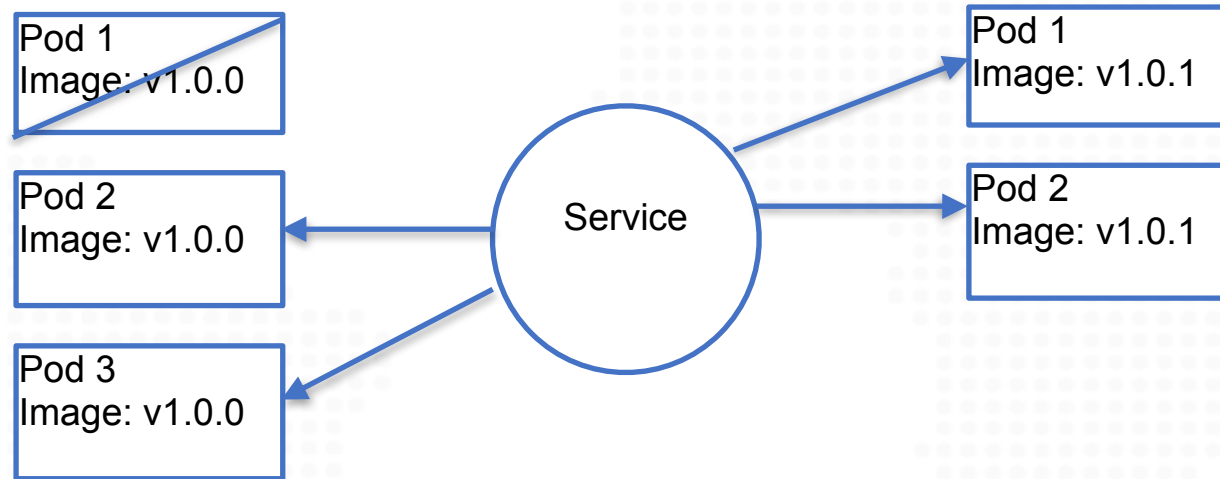
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



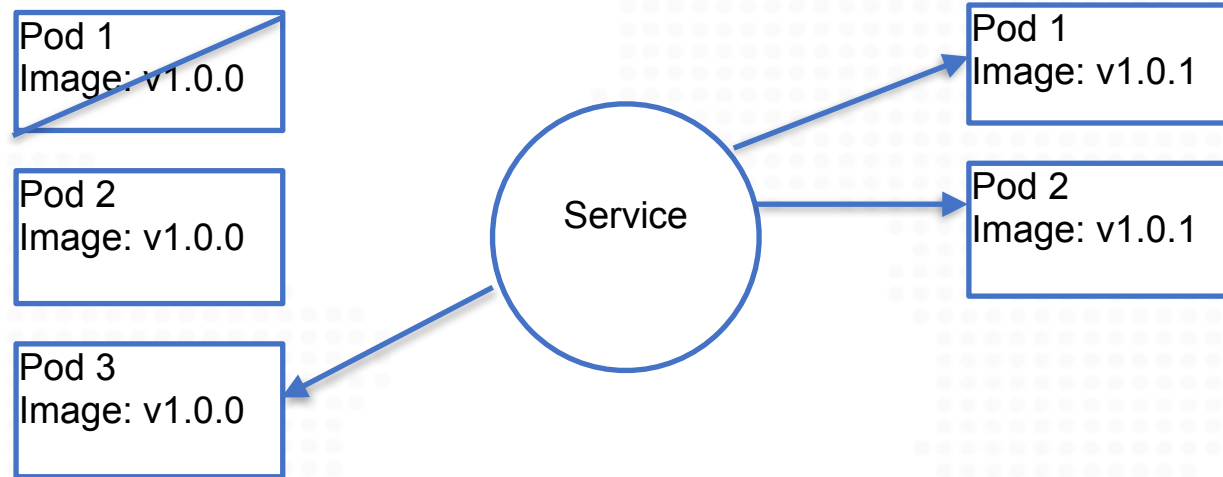
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



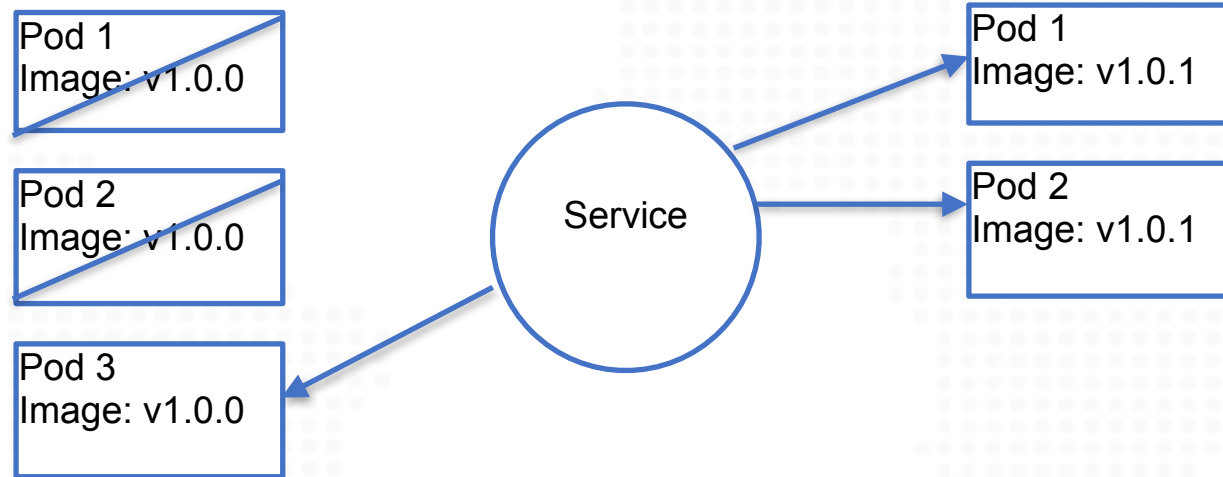
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



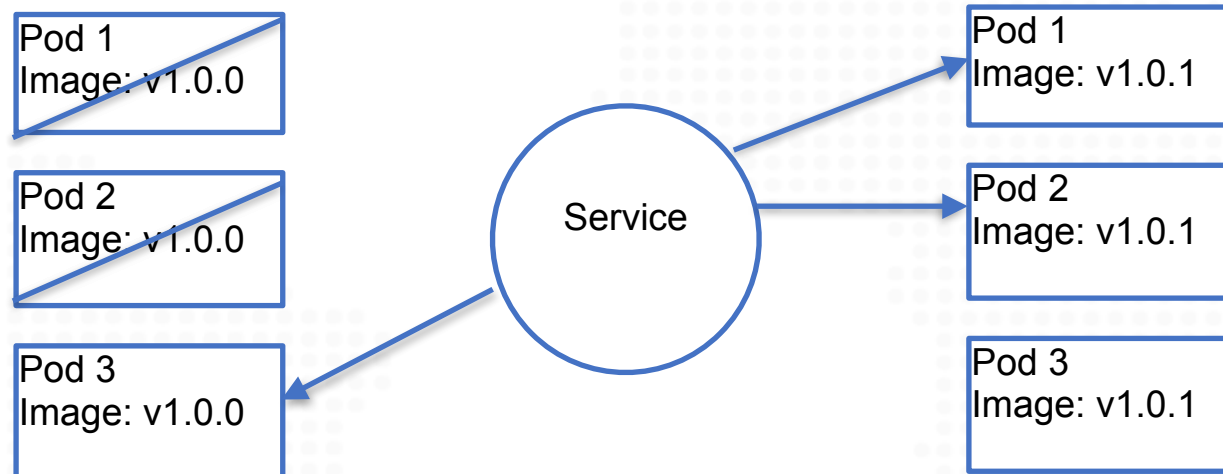
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



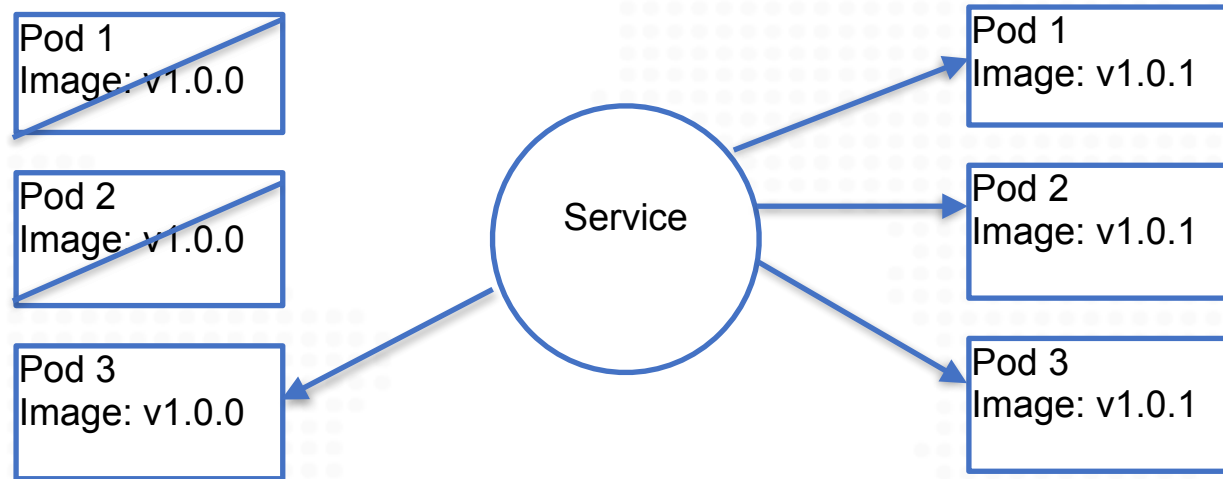
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



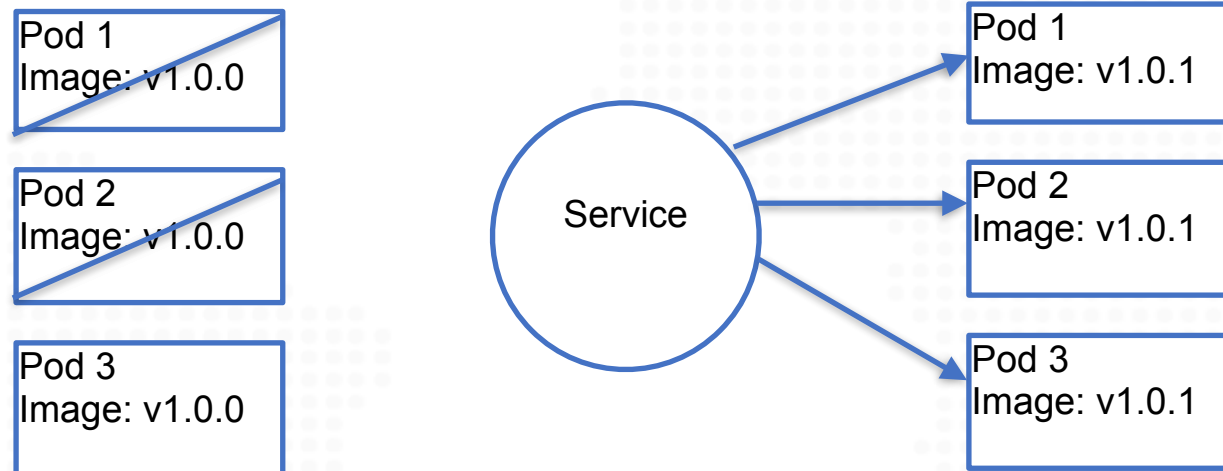
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



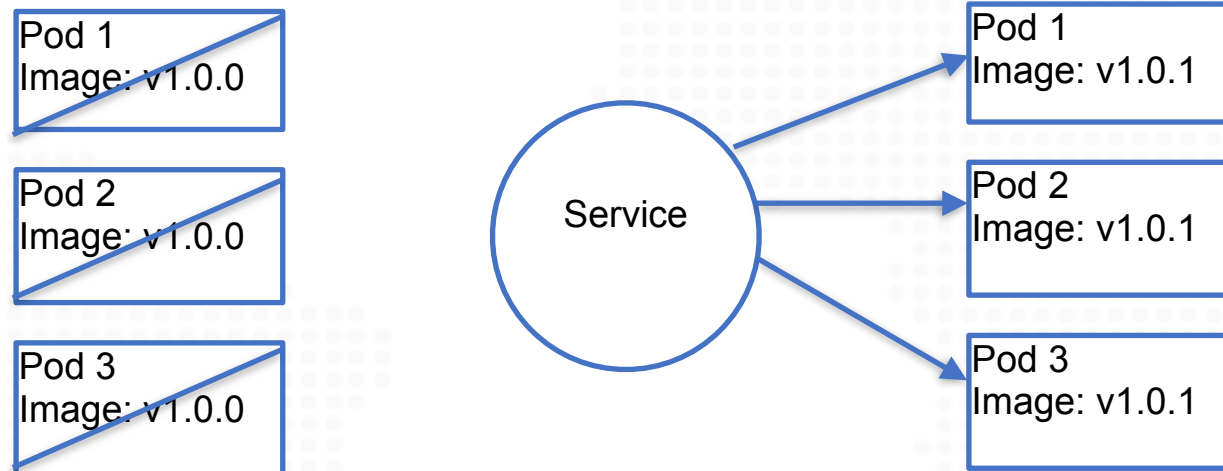
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```


Rolling updates



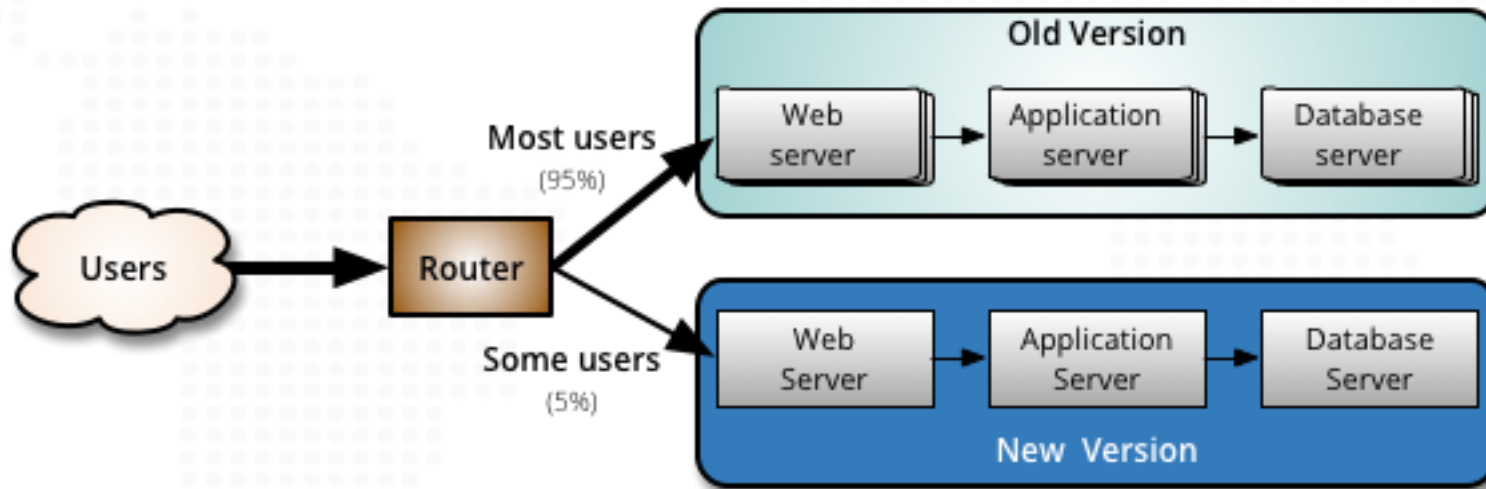
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

Rolling updates



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app:1.0.1
```

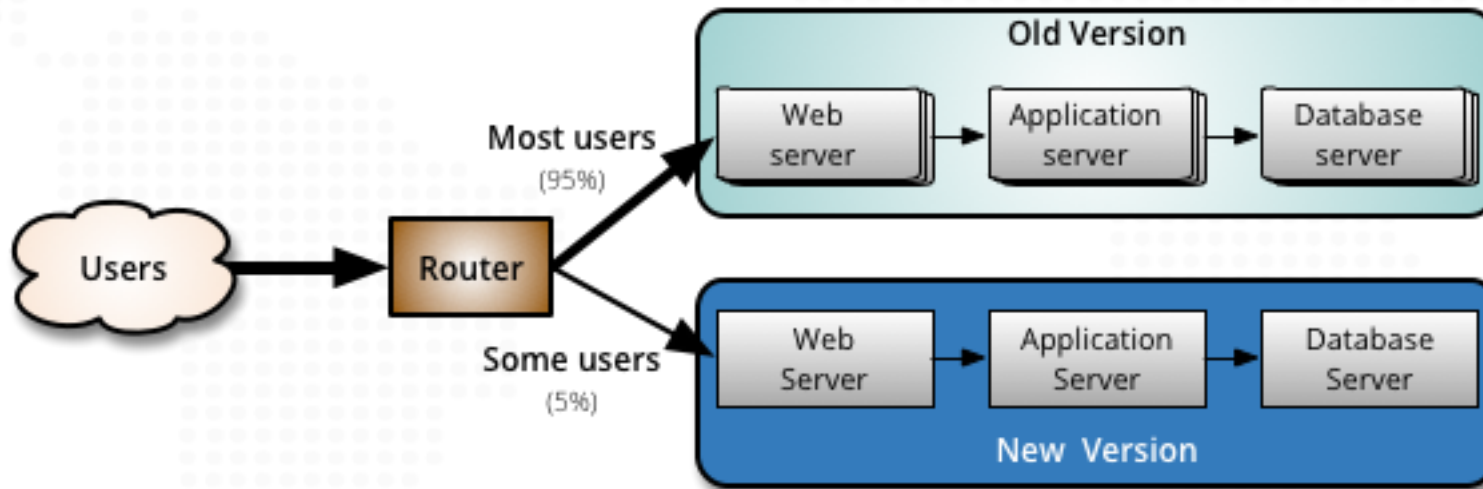
Canary release



(c) <https://martinfowler.com/bliki/CanaryRelease.html>

Canary release

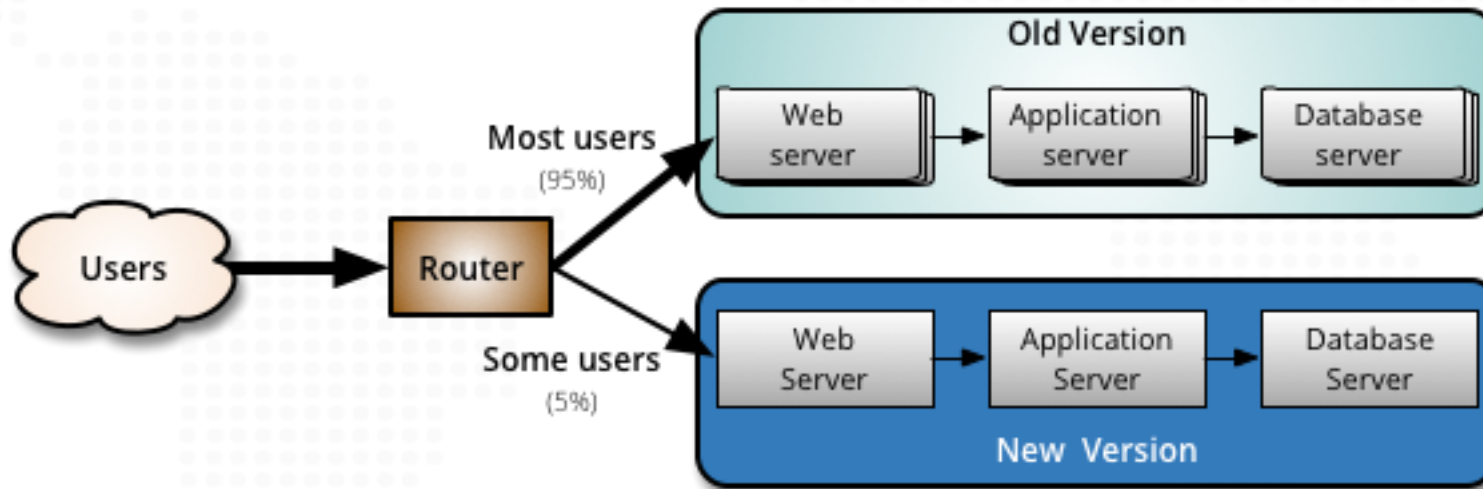
- 8 Users connects to Production-Release
- 2 Users to Production V2



(c) <https://martinfowler.com/bliki/CanaryRelease.html>

Canary release

- 8 Users connects to Production-Release
- 2 Users to Production V2

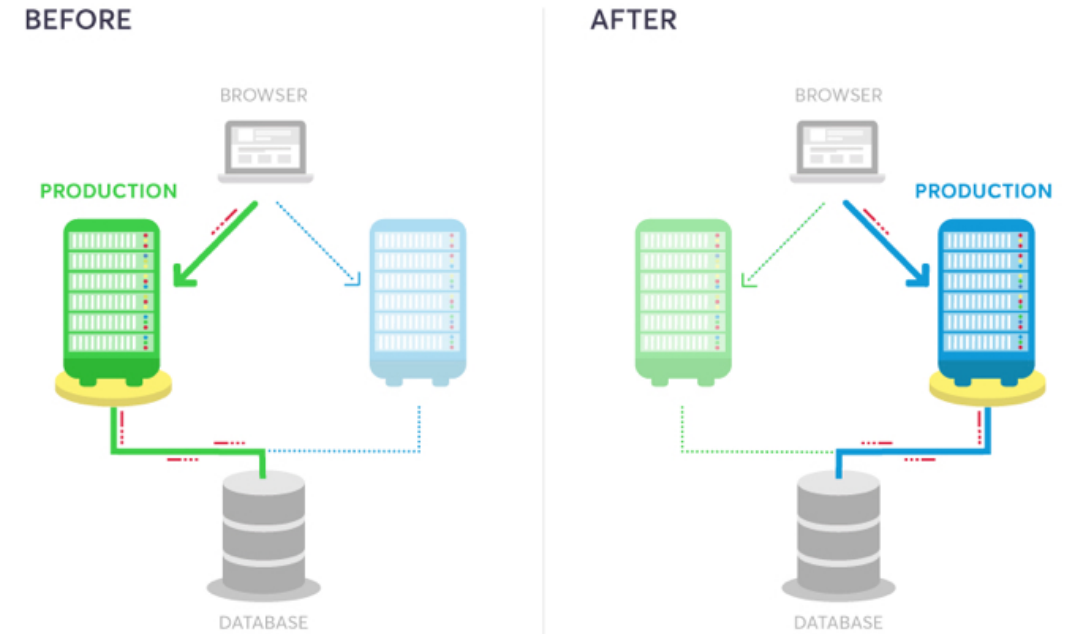


```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: myapp
  namespace: mynamespace
spec:
  to:
    kind: Service
    name: service-prod
    weight: 80

  alternateBackends:
    - kind: Service
      name: service-prod-v2
      weight: 20
  port:
    targetPort: 8080
```

(c) <https://martinfowler.com/bliki/CanaryRelease.html>

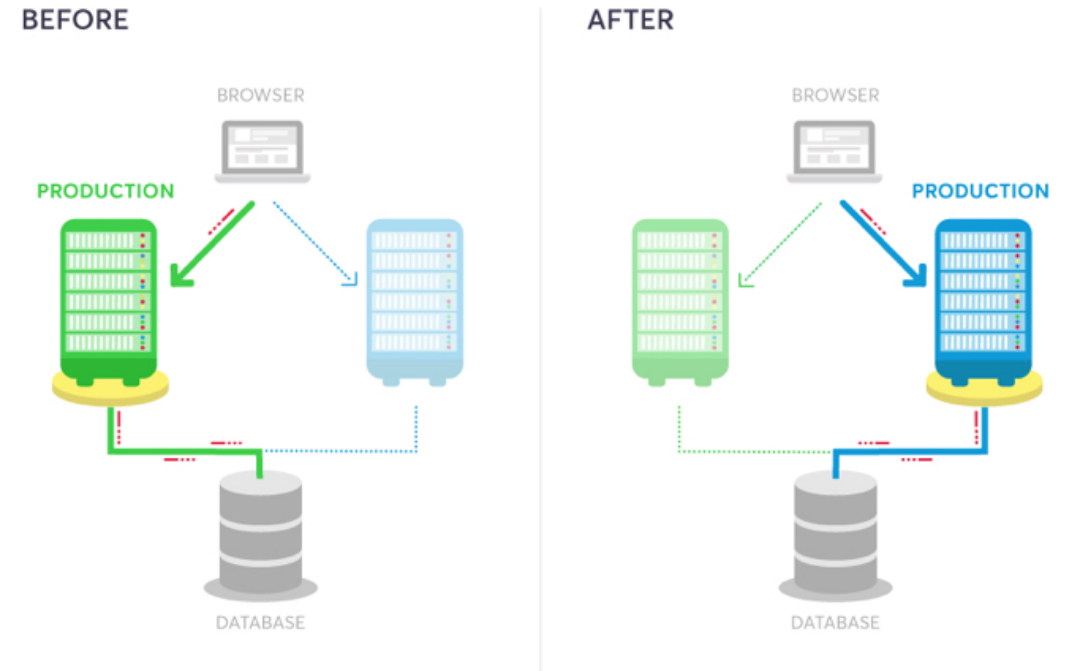
Blue-green deployments



(c) <https://www.abtasty.com/de/blog/blue-green-deployments/>

Blue-green deployments

- Only one part of the system will be replaced



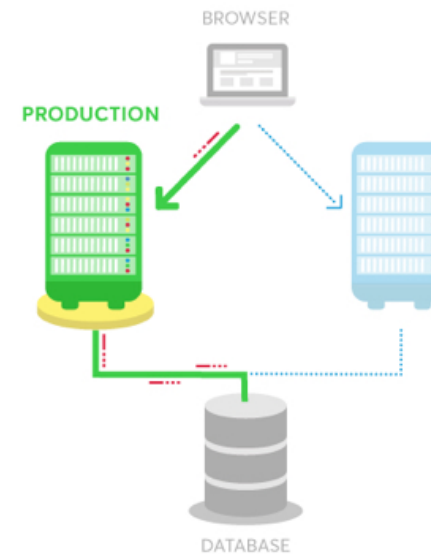
(c) <https://www.abtasty.com/de/blog/blue-green-deployments/>

Blue-green deployments

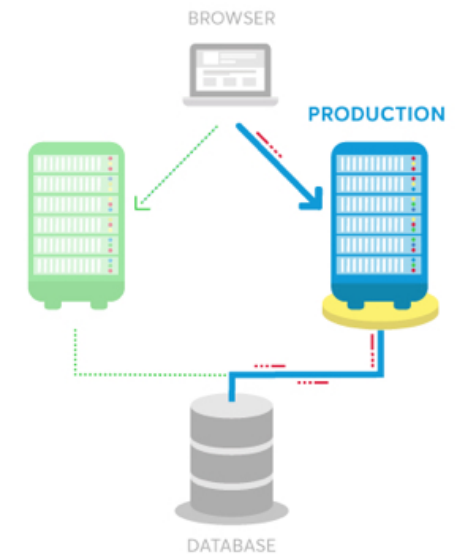
- Only one part of the system will be replaced

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: myapp
  namespace: mynamespace
spec:
  to:
    kind: Service
    name: service-green
  port:
    targetPort: 8080
```

BEFORE



AFTER



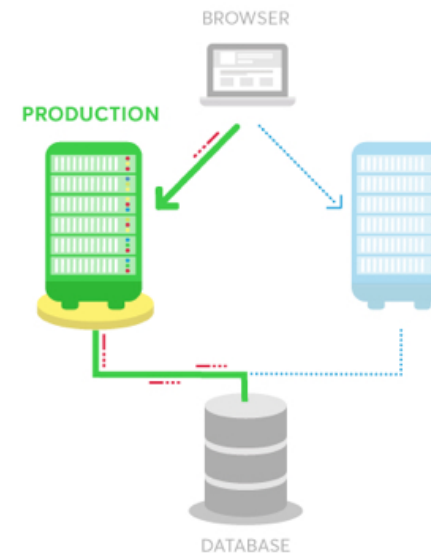
(c) <https://www.abtasty.com/de/blog/blue-green-deployments/>

Blue-green deployments

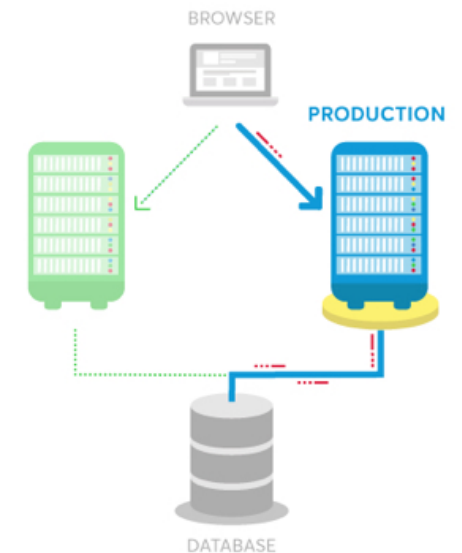
- Only one part of the system will be replaced

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: myapp
  namespace: mynamespace
spec:
  to:
    kind: Service
    name: service-green
  port:
    targetPort: 8080
```

BEFORE



AFTER



(c) <https://www.abtasty.com/de/blog/blue-green-deployments/>

Blue-green deployments

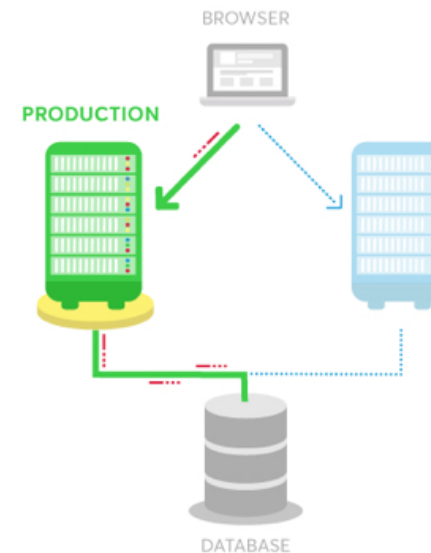
- Only one part of the system will be replaced

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: myapp
  namespace: mynamespace
spec:
  to:
    kind: Service
    name: service-green
port:
  targetPort: 8080
```

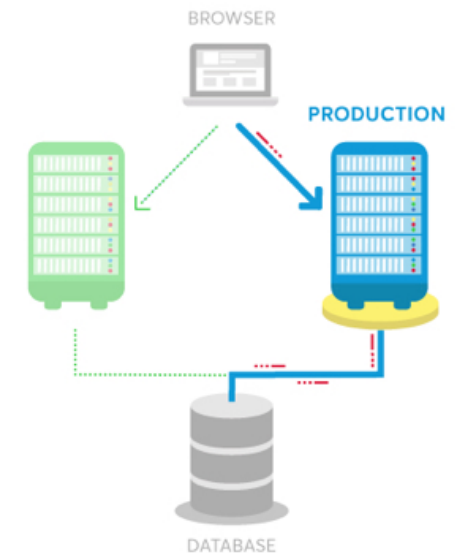


```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: myapp
  namespace: mynamespace
spec:
  to:
    kind: Service
    name: service-blue
port:
  targetPort: 8080
```

BEFORE

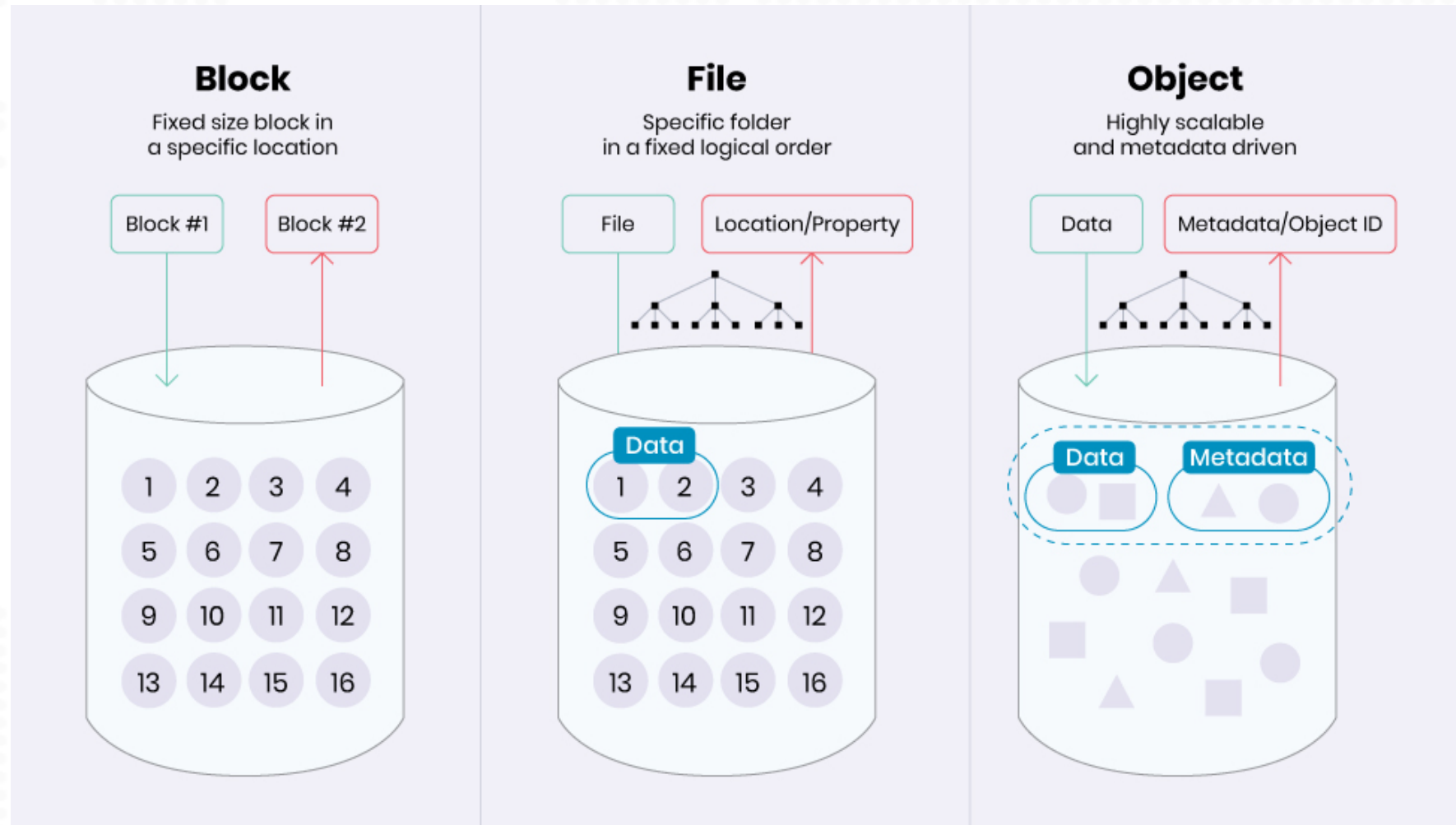


AFTER



(c) <https://www.abtasty.com/de/blog/blue-green-deployments/>

Storage



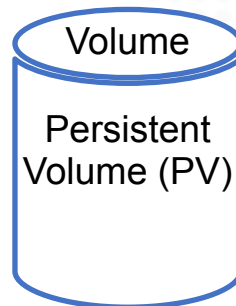
(c) <https://accuweb.cloud/blog/cloud-storage-object-v-s-block-v-s-file-storage/>

Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups

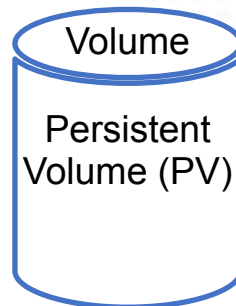
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



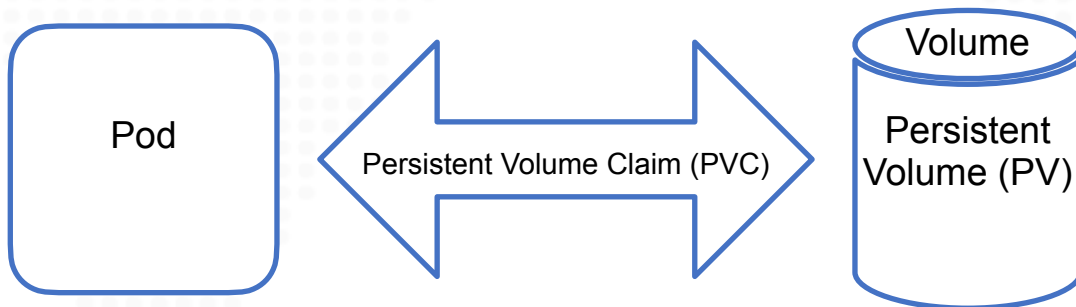
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



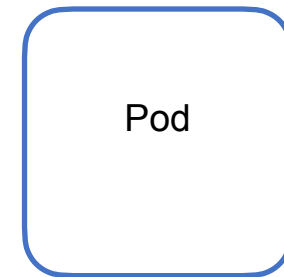
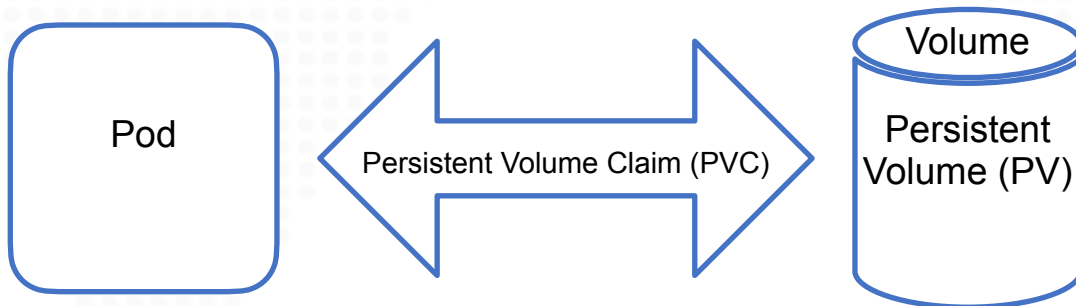
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



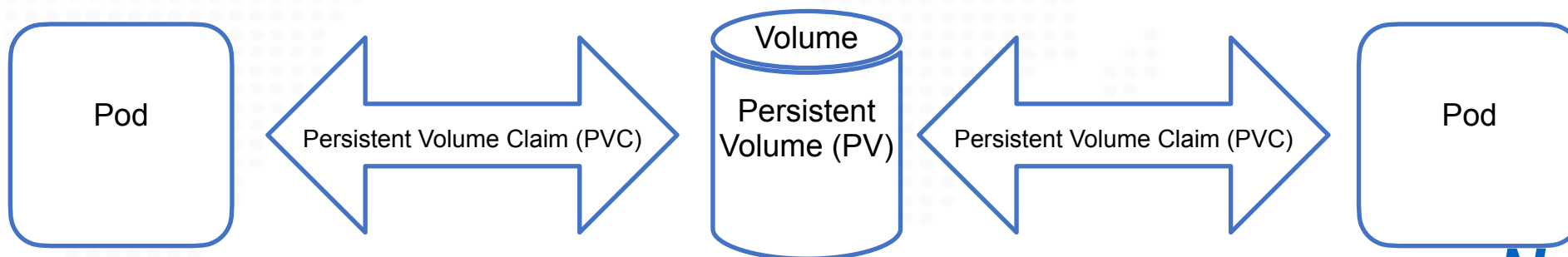
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



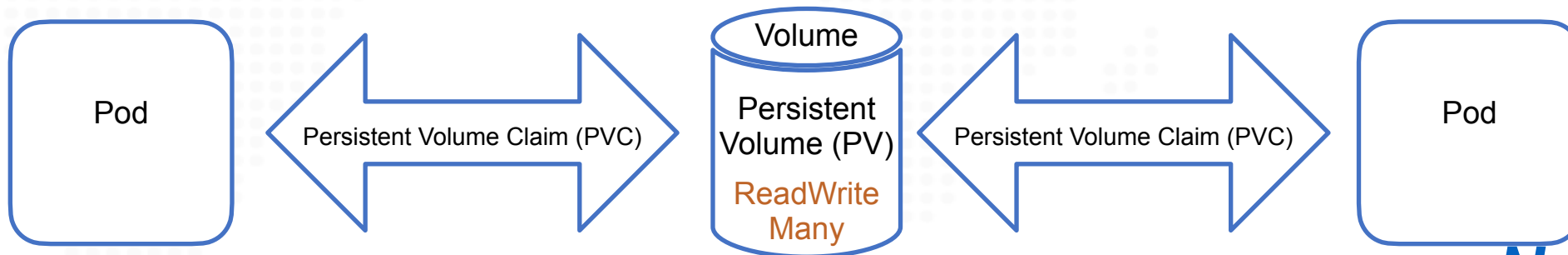
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



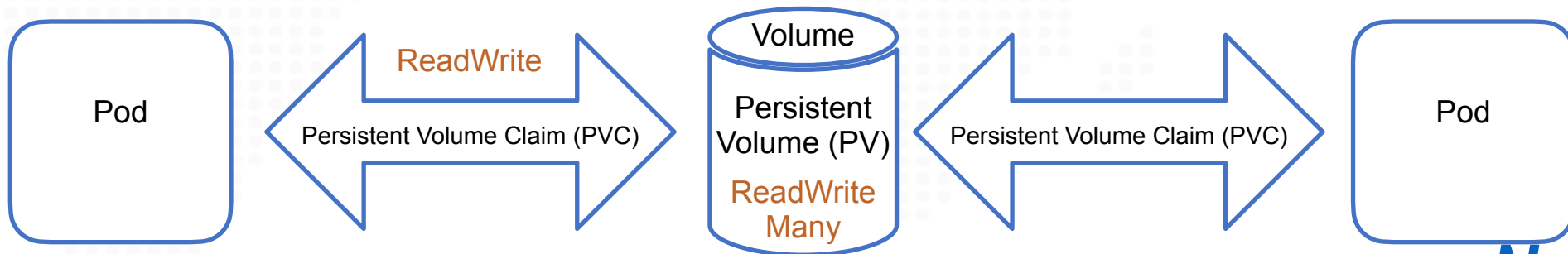
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



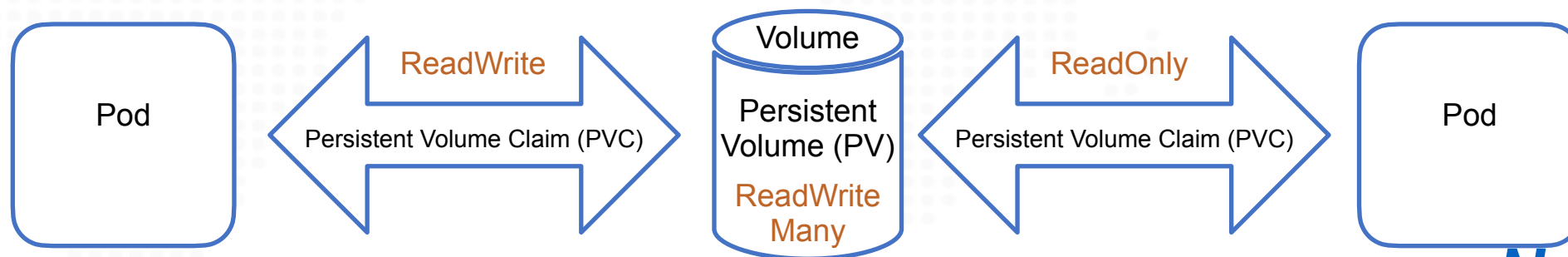
Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



Storage Management

- Access Types: Read Write | Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
 - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



Storage Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/4.storage
```

Storage Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/4.storage
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-data
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: gp3-csi
  resources:
    requests:
      storage: 5Gi
```

Storage Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/4.storage
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mysql-data
              mountPath: /var/lib/mysql
            - name: mysql-logs
              mountPath: /var/log/mysql
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-data
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: gp3-csi
  resources:
    requests:
      storage: 5Gi
```


Storage Lab

```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/4.storage
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mysql-data
              mountPath: /var/lib/mysql
            - name: mysql-logs
              mountPath: /var/log/mysql
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-data
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: gp3-csi
  resources:
    requests:
      storage: 5Gi
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-logs
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 1Gi
```

Storage Lab

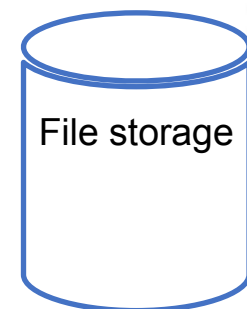
git clone <https://github.com/kolinrr/openshift.git>

cd openshift/Day2/4.storage

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mysql-data
              mountPath: /var/lib/mysql
            - name: mysql-logs
              mountPath: /var/log/mysql
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-data
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: gp3-csi
  resources:
    requests:
      storage: 5Gi
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-logs
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 1Gi
```



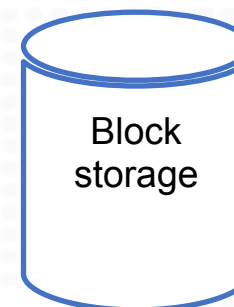
Storage Lab

git clone <https://github.com/kolinrr/openshift.git>

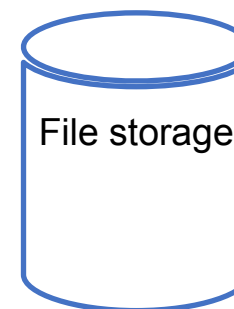
cd openshift/Day2/4.storage

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mysql-data
              mountPath: /var/lib/mysql
            - name: mysql-logs
              mountPath: /var/log/mysql
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-data
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: gp3-csi
  resources:
    requests:
      storage: 5Gi
```



```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-logs
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 1Gi
```



Storage Lab

```
$ cd $HOME/openshift/Day2/4.storage
```

```
//Compare the differences between 1. to 3.
```

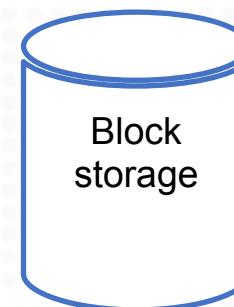
```
$ oc apply -f pvc-block-storage.yaml  
$ oc apply -f pvc-file-storage.yaml  
$ oc apply -f secret.yaml  
$ oc apply -f stateful.yaml
```

```
apiVersion: apps/v1  
kind: StatefulSet  
metadata:  
  name: mysql  
spec:  
  serviceName: mysql  
  replicas: 1  
  selector:  
    matchLabels:  
      app: mysql  
  template:  
    metadata:  
      labels:  
        app: mysql  
    spec:  
      containers:  
        - name: mysql  
          image: mysql:latest  
          ports:  
            - containerPort: 3306  
          volumeMounts:  
            - name: mysql-data  
              mountPath: /var/lib/mysql  
            - name: mysql-logs  
              mountPath: /var/log/mysql
```

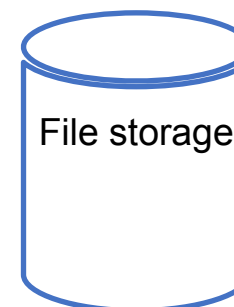
```
git clone https://github.com/kolinrr/openshift.git
```

```
cd openshift/Day2/4.storage
```

```
apiVersion: v1  
kind: PersistentVolumeClaim  
metadata:  
  name: mysql-data  
spec:  
  accessModes:  
    - ReadWriteOnce  
  storageClassName: gp3-csi  
resources:  
  requests:  
    storage: 5Gi
```



```
apiVersion: v1  
kind: PersistentVolumeClaim  
metadata:  
  name: mysql-logs  
spec:  
  accessModes:  
    - ReadWriteMany  
  storageClassName: efs-sc  
resources:  
  requests:  
    storage: 1Gi
```





Finish!

Questions?