Day 2

NobleProg

Day 2

- Github Actions
- Kustomize
- Advanced deployment
- Storage

NobleProg

Day 2

NobleProg

Day 2

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

NobleProg

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

```yaml
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 *
```

NobleProg

Github Actions Example

```yaml
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 *
```

```yaml
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 *
```

NobleProg

Github Actions Example

```yaml
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 *
```

```yaml
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

NobleProg

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

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

```yaml
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
```

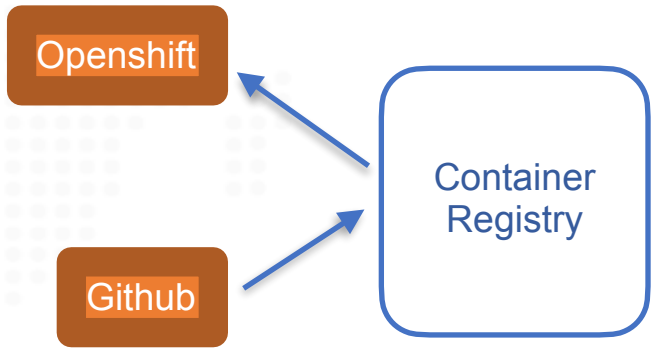
NobleProg

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



NobleProg

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

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

```
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
    …
```

NobleProg

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:
    - OPENSHIFT_SERVER
    - OPENSHIFT_NAMESPACE
    - OPENSHIFT_TOKEN

NobleProg

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=${{ steps.push-image.outputs.registry-path }}
        export OPENSHIFT_NAMESPACE=${{ env.OPENSHIFT_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 $OPENSHIFT_NAMESPACE
        oc apply -f Day1/network/service.yaml -n $OPENSGIFT_NAMSPACE

        oc get pods
        oc get svc
...
```

NobleProg

Lunch Break: 30 minutes

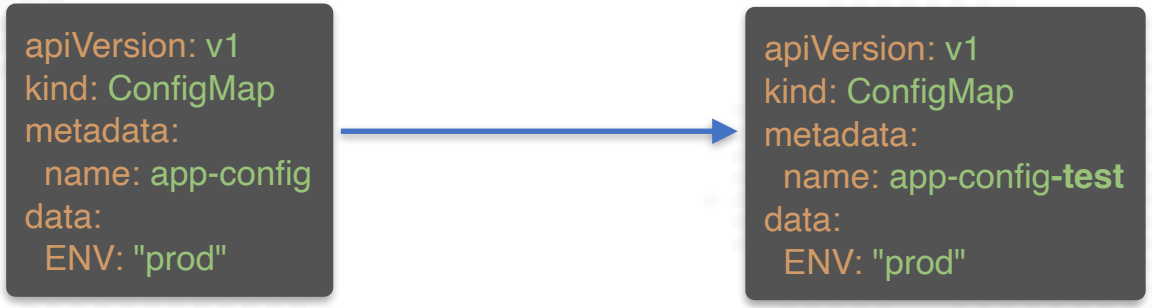
NobleProg

Kustomize 1

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

cd openshift/Day2/1.kustomize

- Template manipulation of objects



Kustomize 1

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

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

NobleProg

Kustomize 1

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

NobleProg

Kustomize 1

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

NobleProg

Kustomize 1

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

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

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

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

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

NobleProg

Kustomize 1

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

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

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

kustomize build overlays/prod

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

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

NobleProg

Kustomize 1

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

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

```
#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"
```

NobleProg

Kustomize 2

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

cd openshift/Day2/2.kustomize
```

Extended template manipulation of objects

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

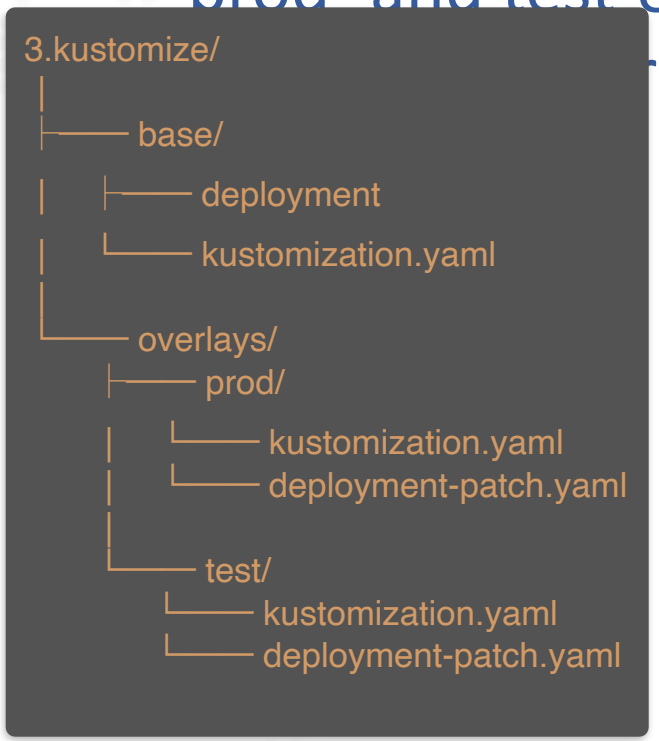
Kustomize 2

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

cd openshift/Day2/2.kustomize

Extended template manipulation of objects

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



NobleProg

Kustomize 2

Extended template mar s

- Deploy 2 more appli
  prod- and test-envir
  resou

```
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
```

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

cd openshift/Day2/2.kustomize

NobleProg

Kustomize 2

Extended template mar s

- Deploy 2 more appli
  prod- and test-envir
  resou

```
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

NobleProg

Kustomize 2

Extended template mar s

- Deploy 2 more appli
  prod- and test-envir
  resou

```
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"
```

NobleProg

Kustomize 2

Extended template mar s

- Deploy 2 more appli
  prod- and test-envir
  resou

```
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/
kind: Kustomization
patches:
- path: deployment-patch.yaml
```

Kustomize 2

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

```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
```

```yaml
#base/kustomization.yaml

resources:
  - deployment.yaml
```

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

cd openshift/Day2/2.kustomize

```yaml
#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"
```

```yaml
#overlays/test/kustomization.yaml
resources:
- ../../base

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

Kustomize 2

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

```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
```

```yaml
#base/kustomization.yaml

resources:
  - deployment.yaml
```

kustomize build overlays/prod

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

cd openshift/Day2/2.kustomize

```yaml
#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"
```

```yaml
#overlays/test/kustomization.yaml
resources:
- ../../base

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

Kustomize 2

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

```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
```

```yaml
#base/kustomization.yaml

resources:
  - deployment.yaml
```

kustomize build overlays/prod

kustomize build overlays/test

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

cd openshift/Day2/2.kustomize

```yaml
#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"
```

```yaml
#overlays/test/kustomization.yaml
resources:
- ../../base

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

Kustomize 3

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

cd openshift/Day2/3.kustomize
```

Extended template manipulation of objects

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

NobleProg

Kustomize 3

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

cd openshift/Day2/3.kustomize

NobleProg

Kustomize 3

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

cd openshift/Day2/3.kustomize

kustomize build overlays/prod

NobleProg

Kustomize 3

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

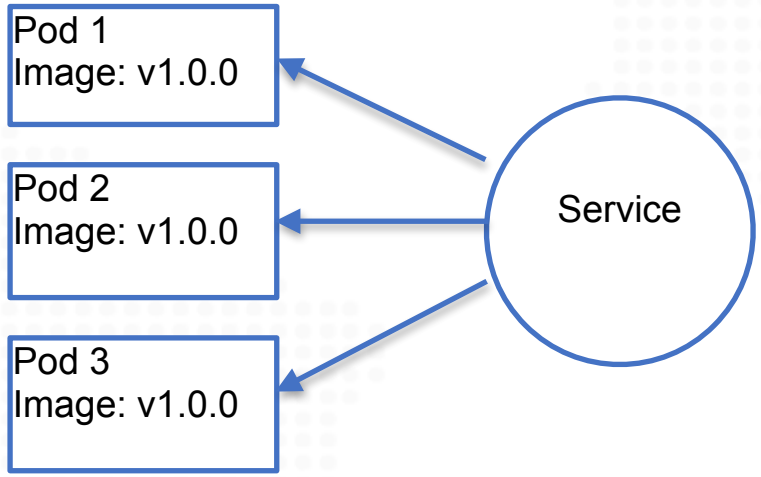
cd openshift/Day2/3.kustomize

kustomize build overlays/prod

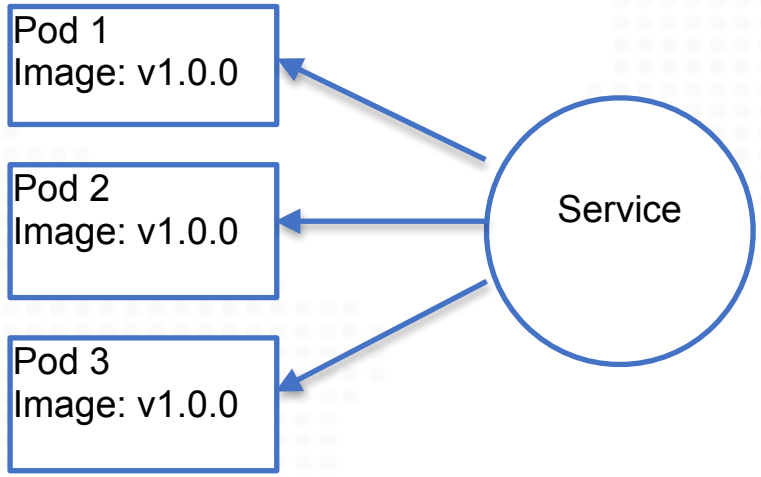
kustomize build overlays/test

NobleProg

Rolling updates



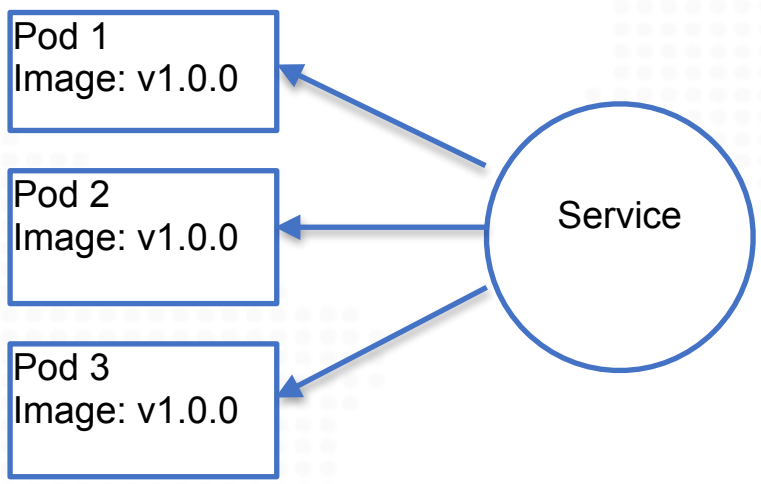
Rolling updates




```
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
```

NobleProg

Rolling updates

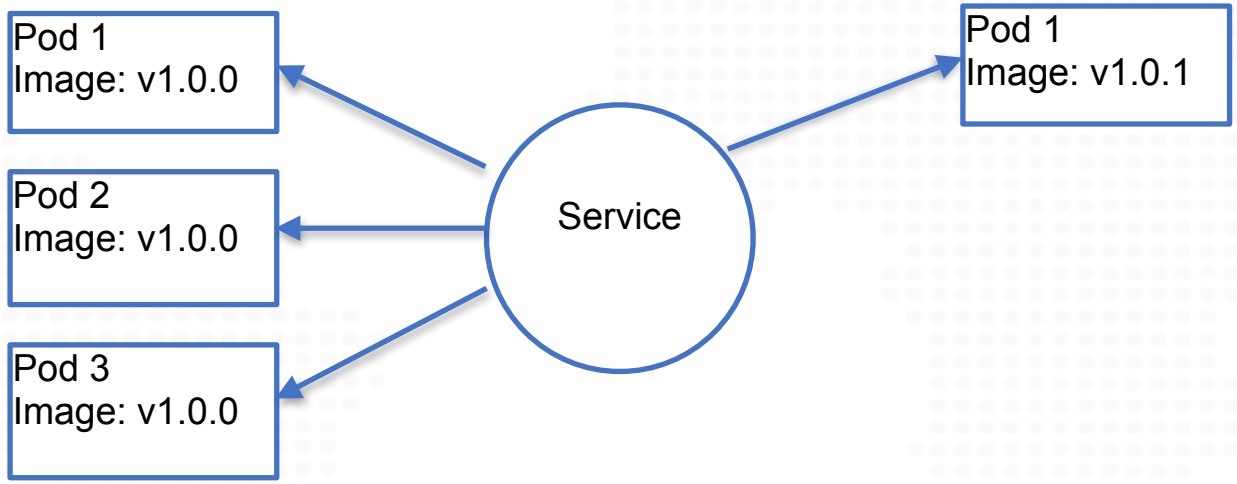


Pod 1
Image: v1.0.1

```
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
```

NobleProg

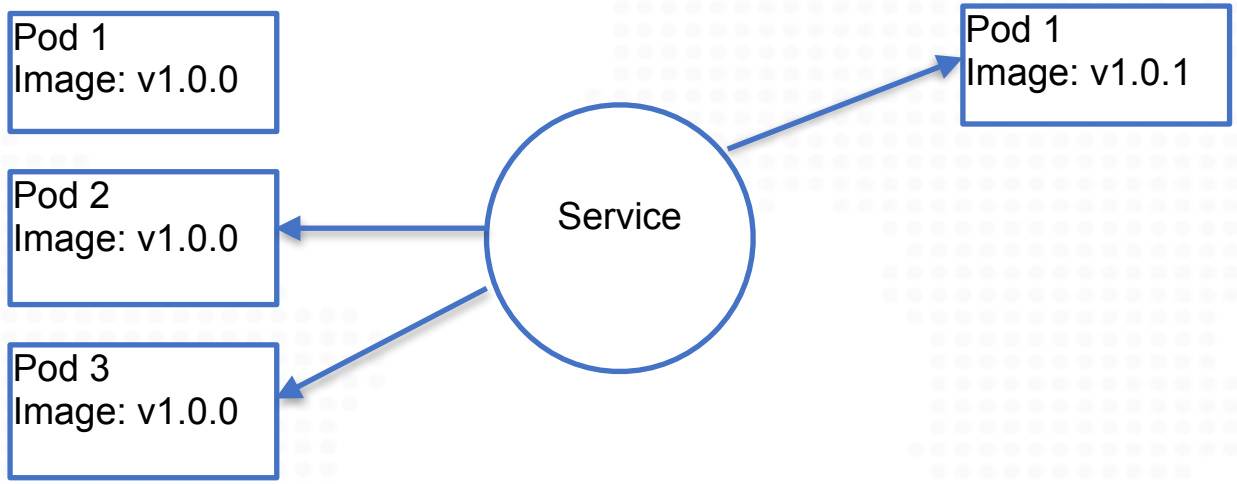
Rolling updates




```
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
```

NobleProg

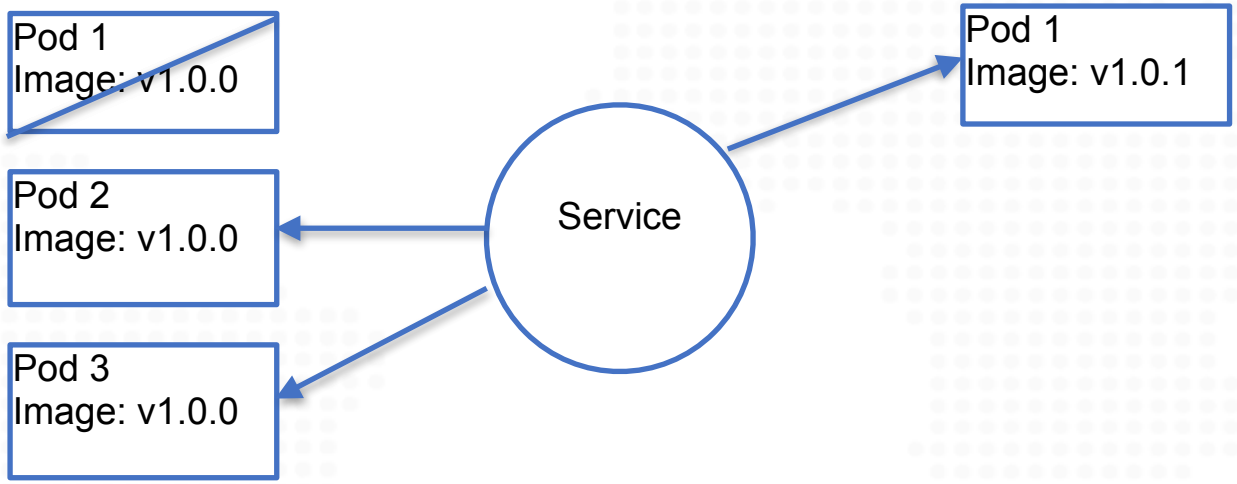
Rolling updates




```
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
```

NobleProg

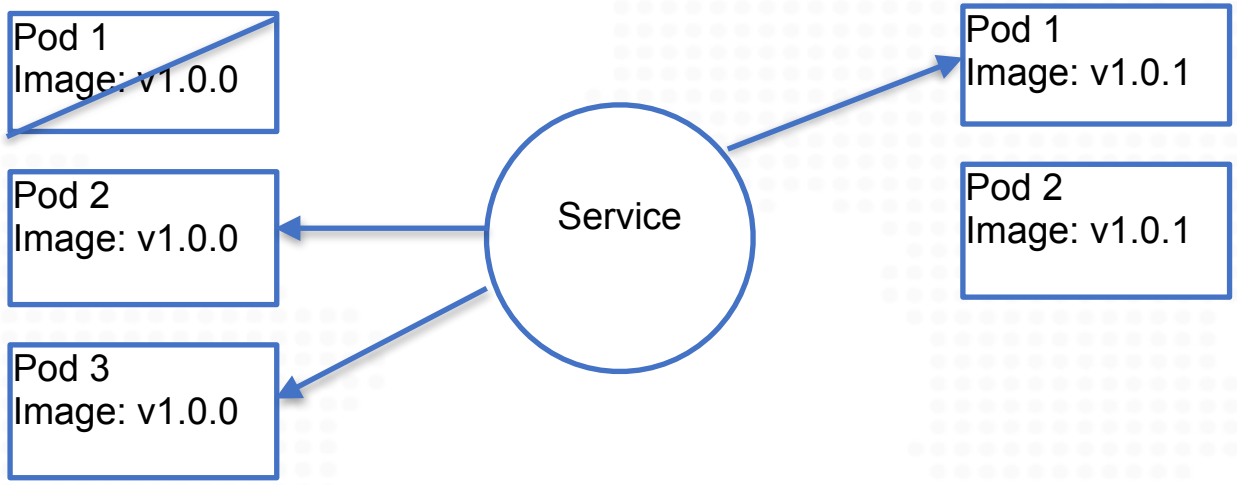
Rolling updates




```yaml
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
```

NobleProg

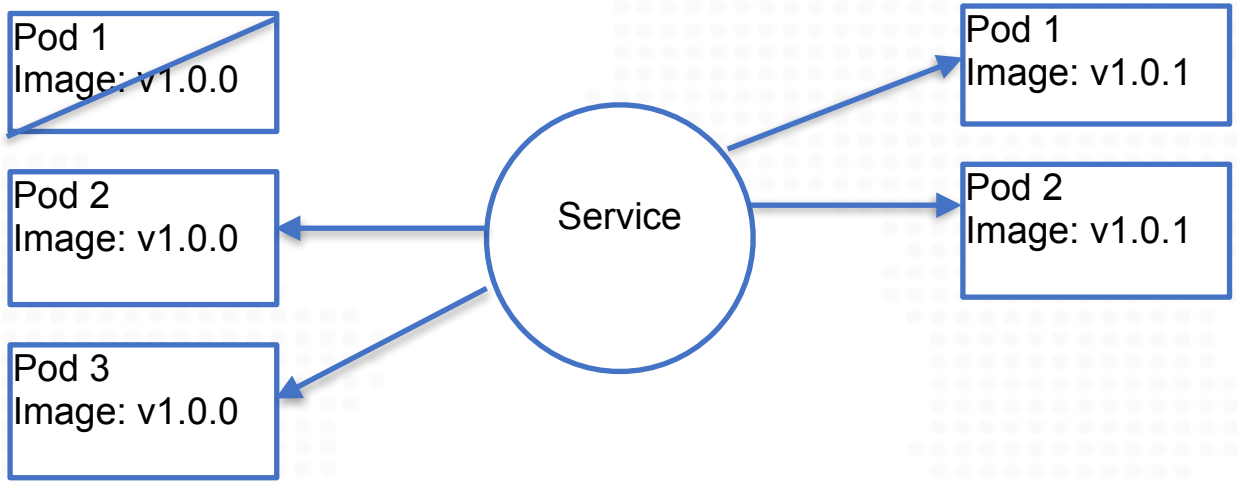
Rolling updates




```yaml
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
```

NobleProg

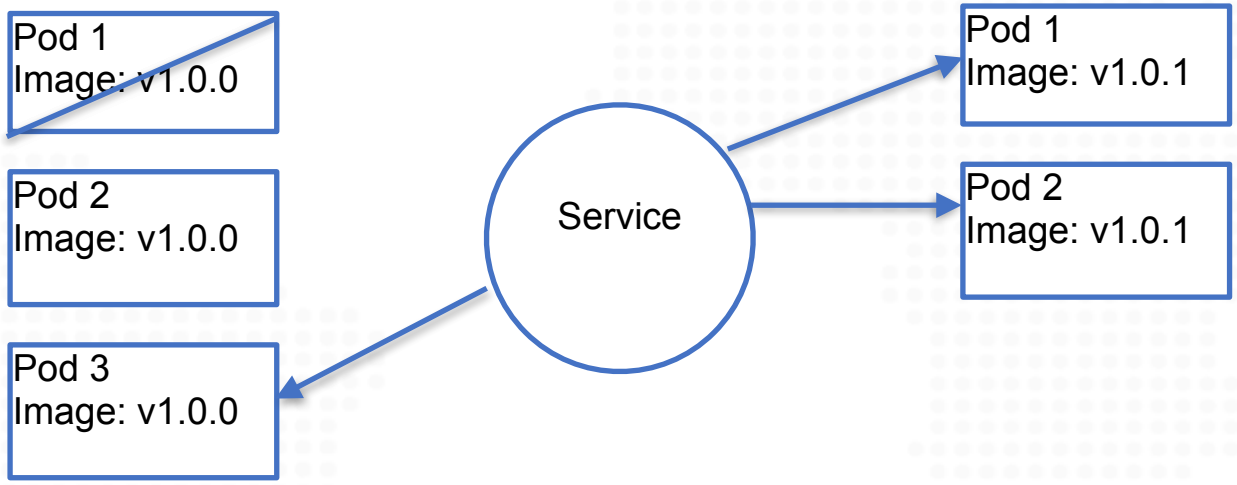
Rolling updates




```
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
```

NobleProg

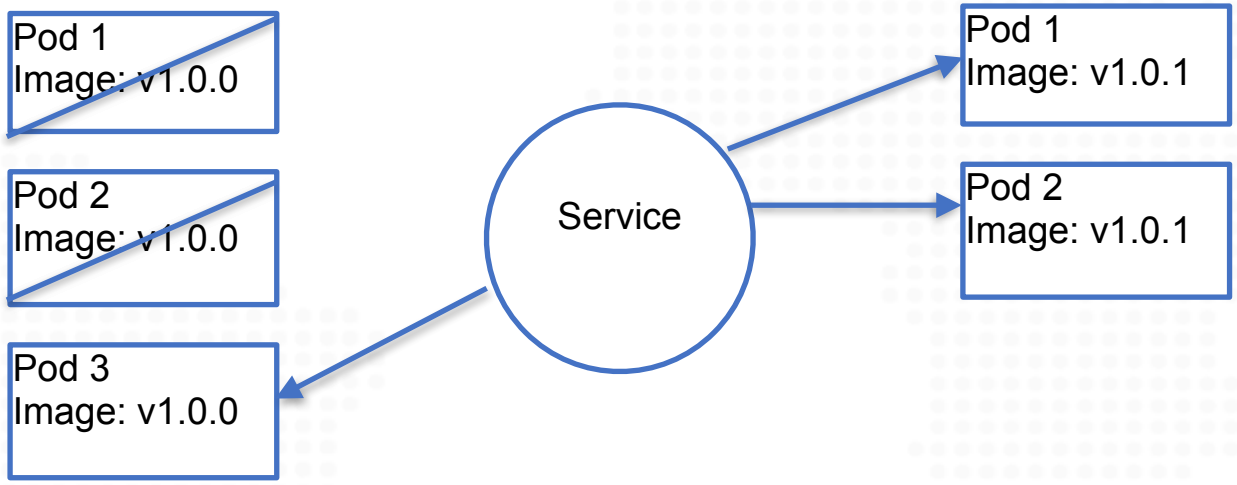
Rolling updates




```yaml
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
```

NobleProg

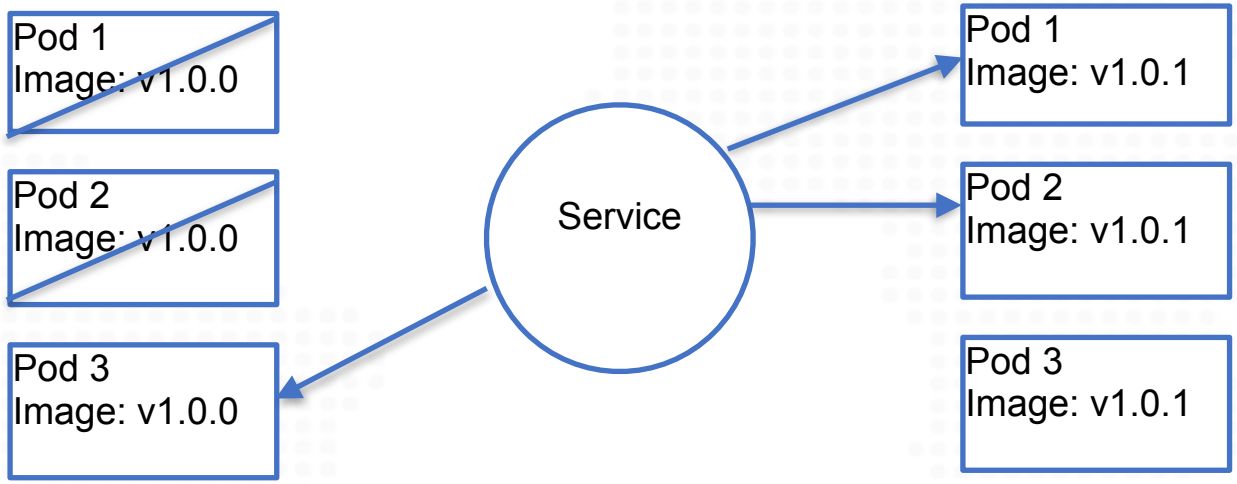
Rolling updates




```yaml
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
```

NobleProg

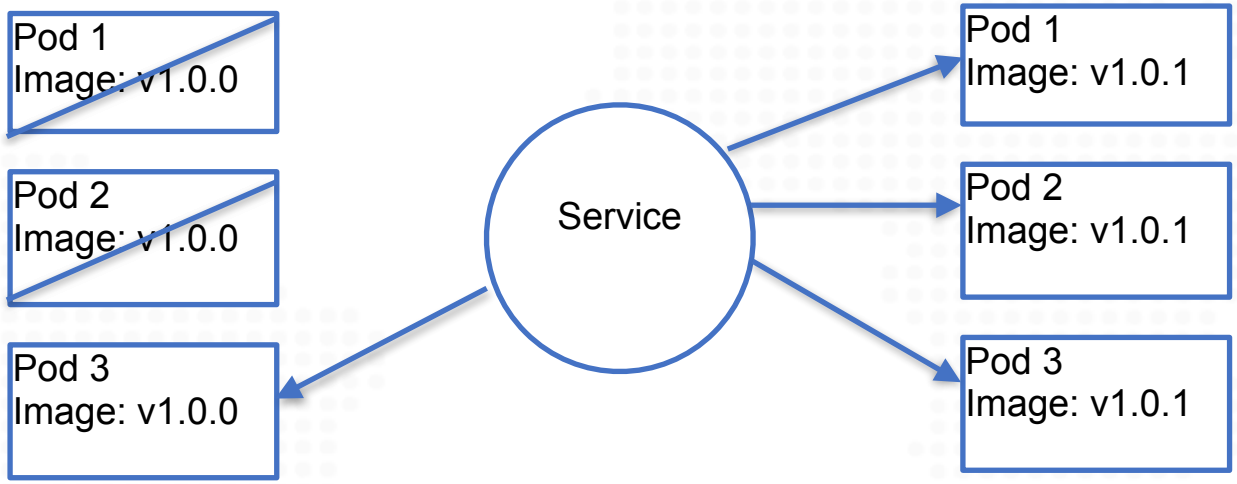
Rolling updates




```yaml
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
```

NobleProg

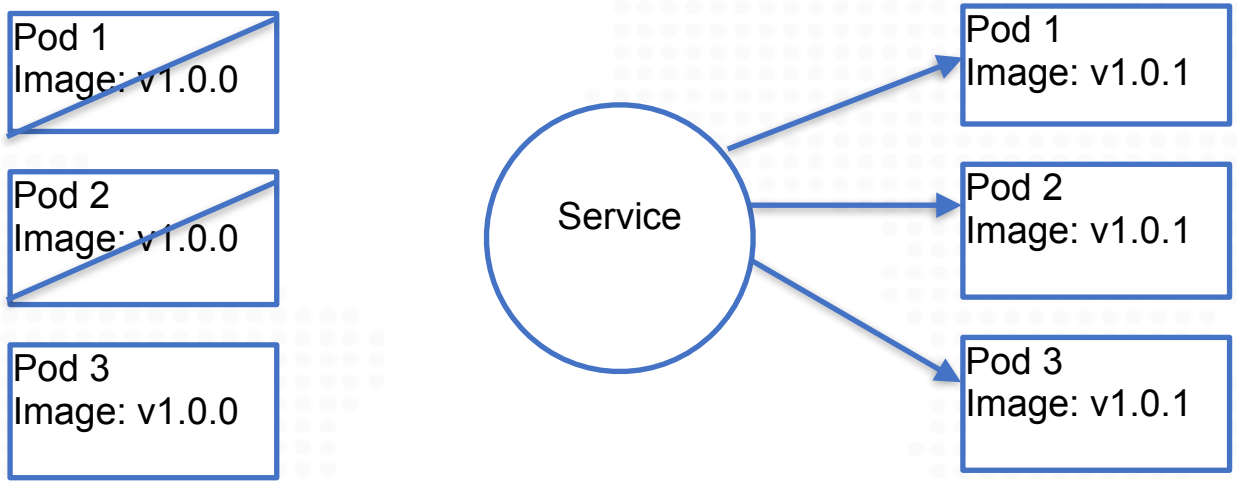
Rolling updates




```yaml
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
```

NobleProg

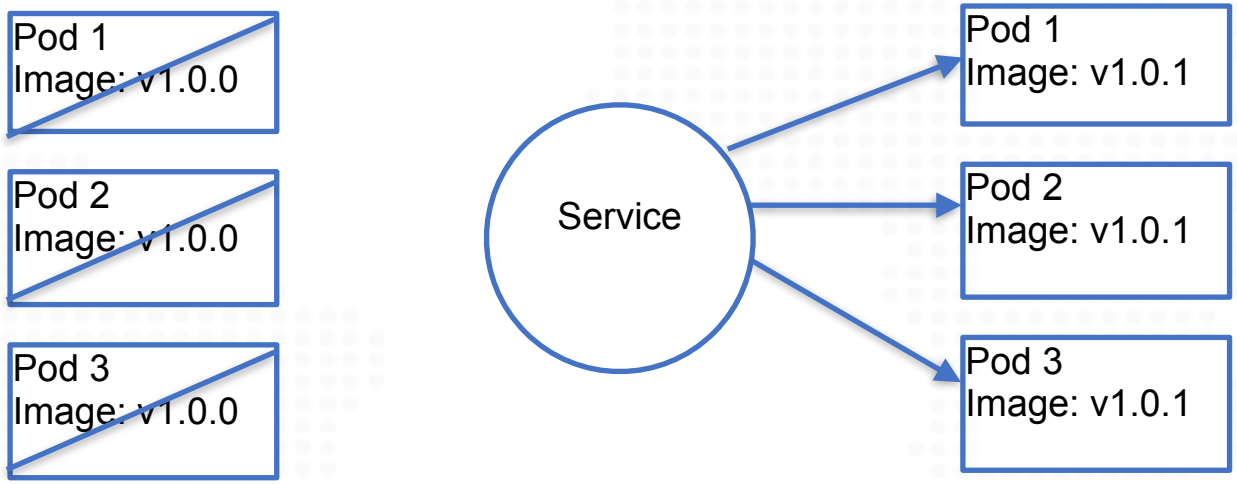
Rolling updates




```yaml
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
```

NobleProg

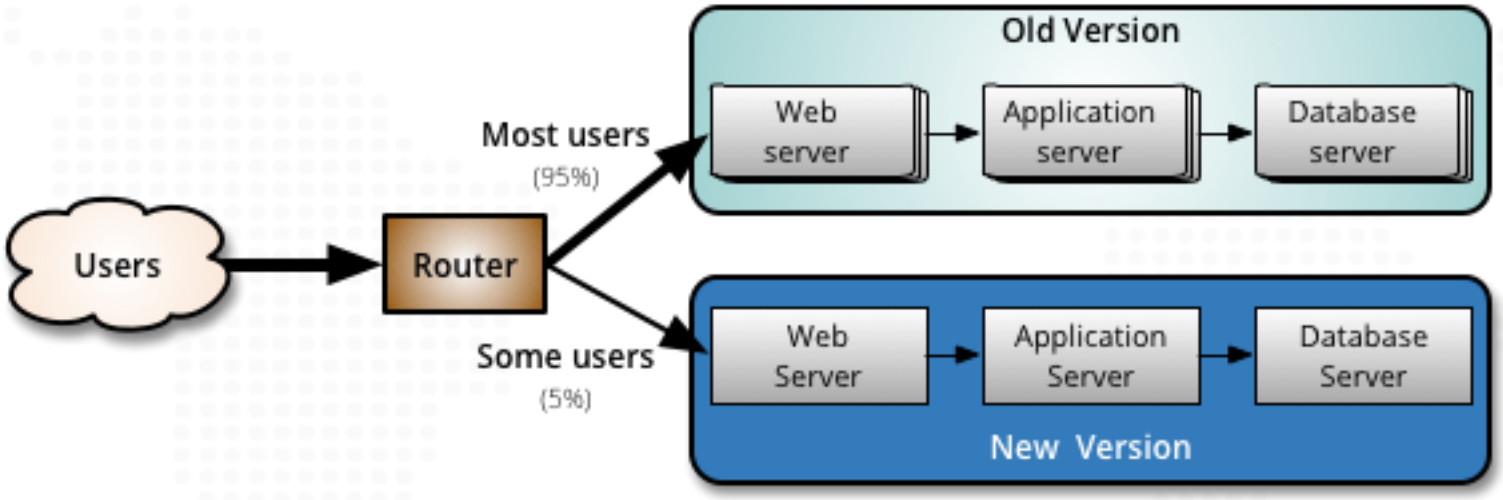
Rolling updates




```yaml
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
```

NobleProg

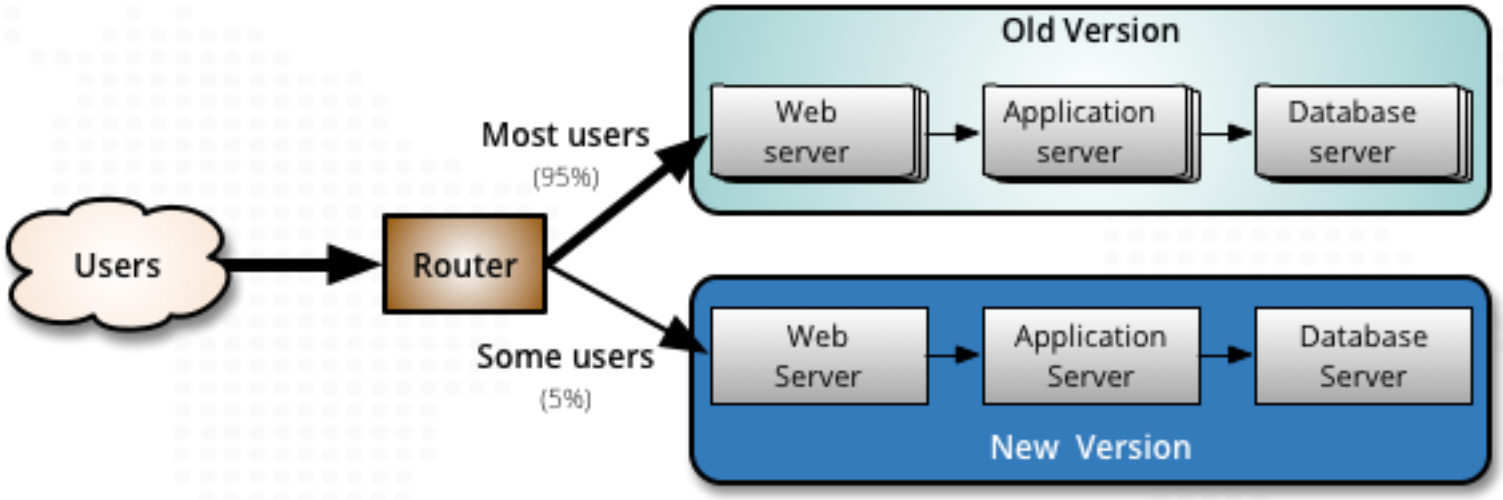
Canary release



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

Canary release

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

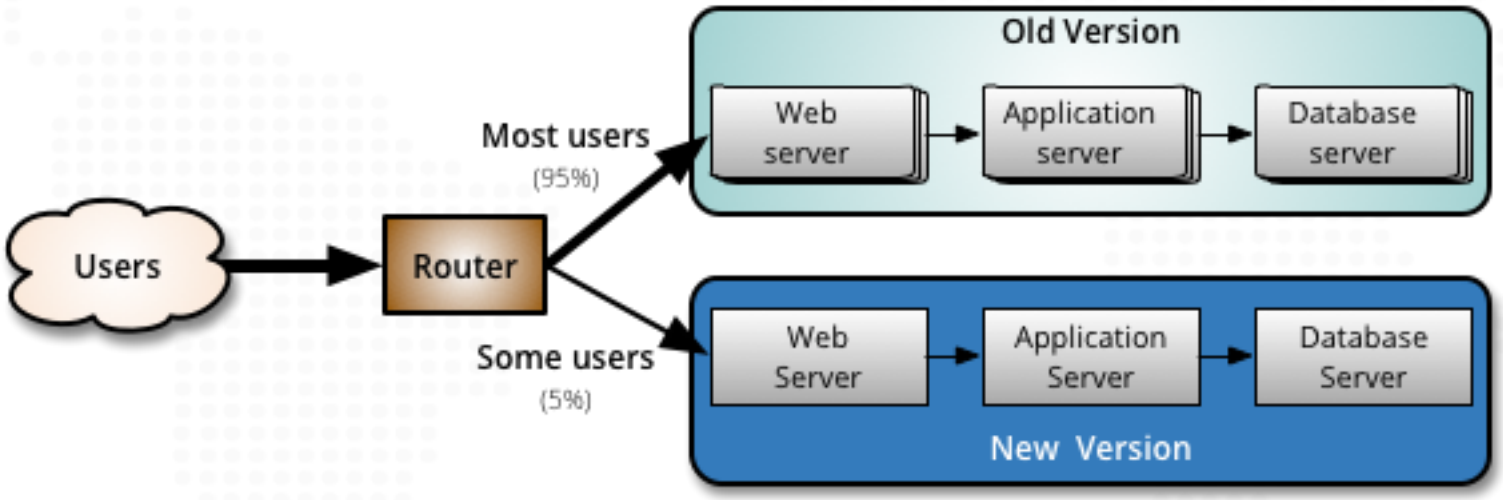


(c) https://martinfowler.com/bliki/CanaryRelease.html

Canary release

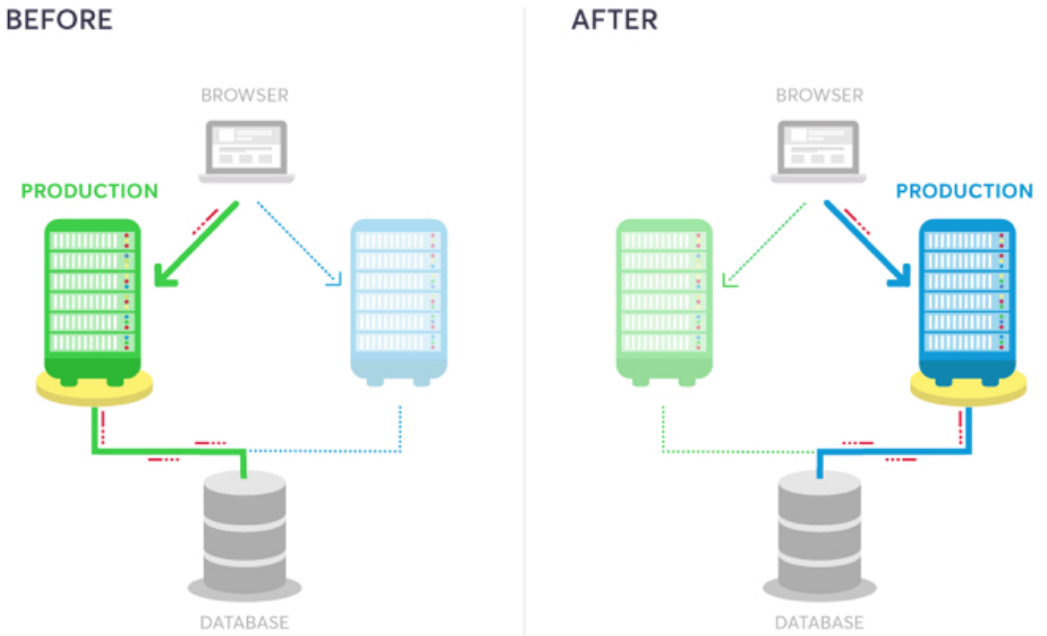
- 8 Users connects to Production-Release
- 2 Users to Production V2

```
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

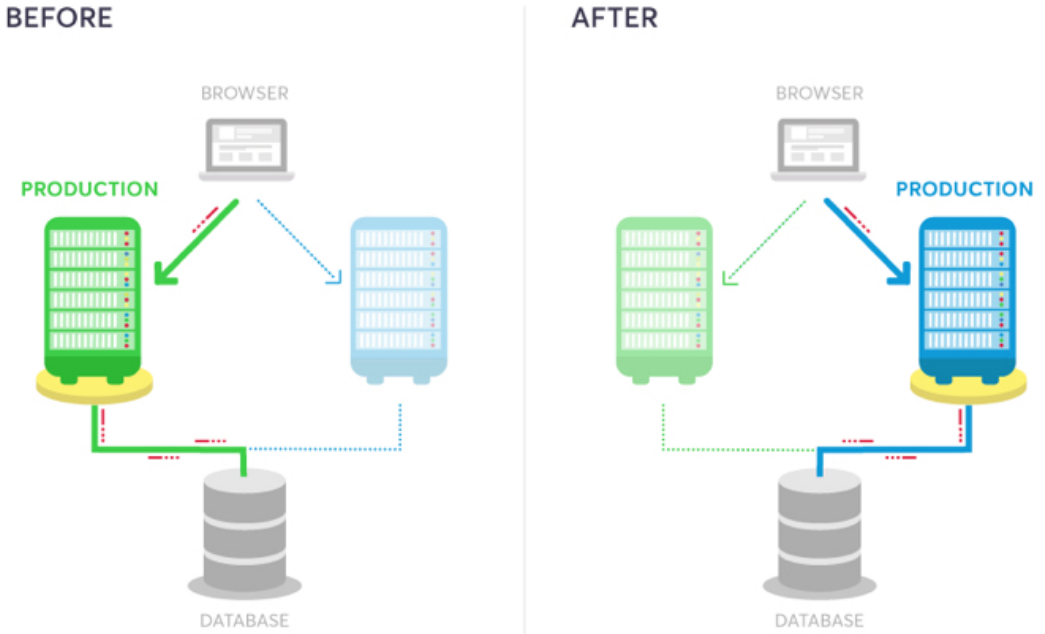


(c) https://www.abtasty.com/de/blog/blue-green-deployments/

NobleProg

Blue-green deployments

- Only one part of the system will be replaced

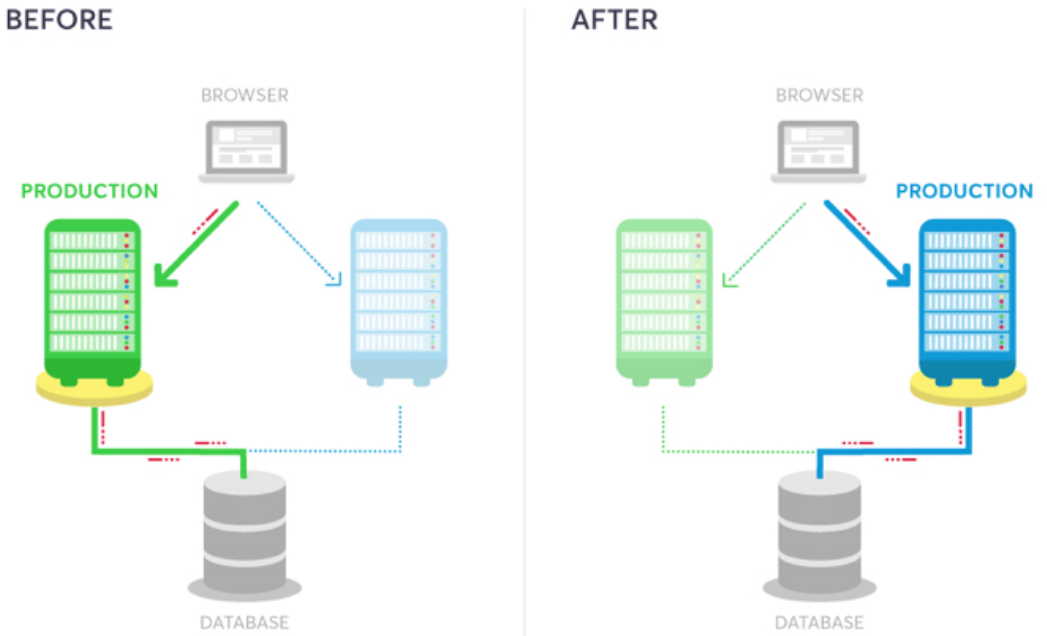


(c) https://www.abtasty.com/de/blog/blue-green-deployments/

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
```



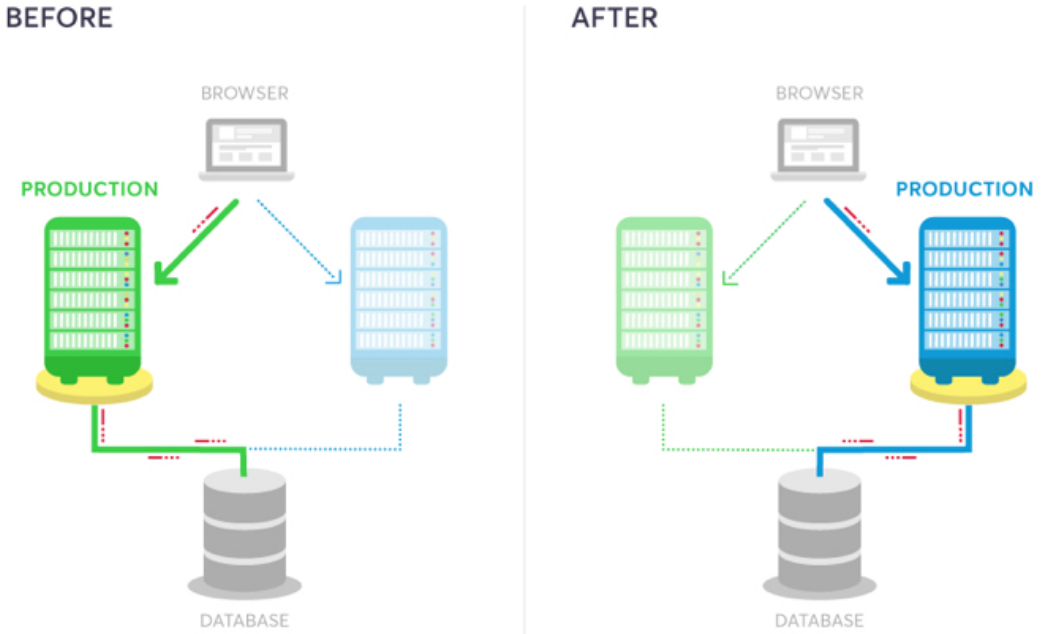
(c) https://www.abtasty.com/de/blog/blue-green-deployments/

NobleProg

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
```



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

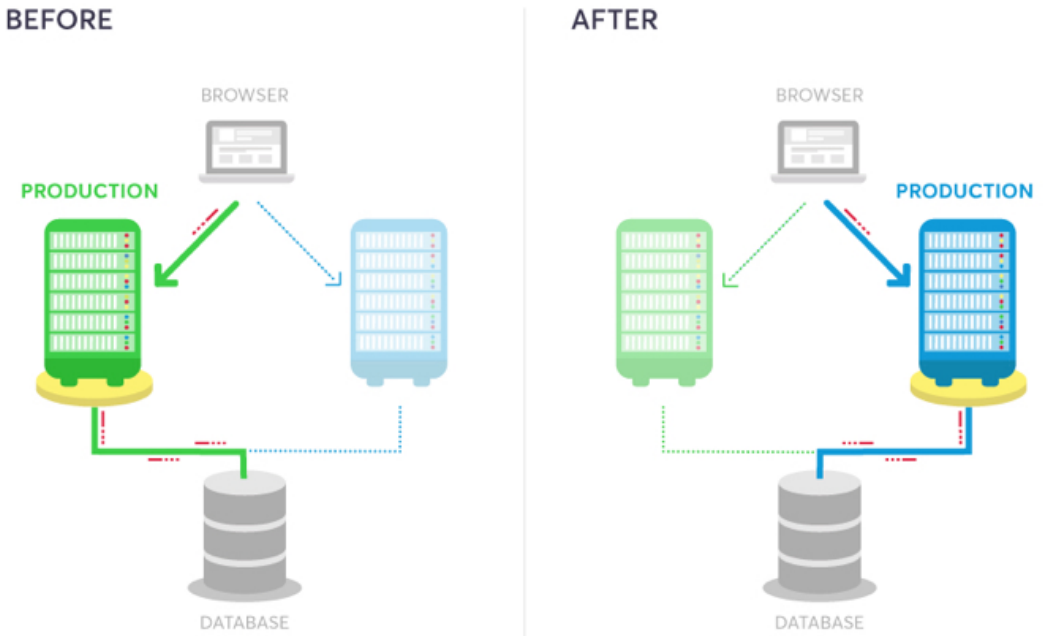
NobleProg

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
```

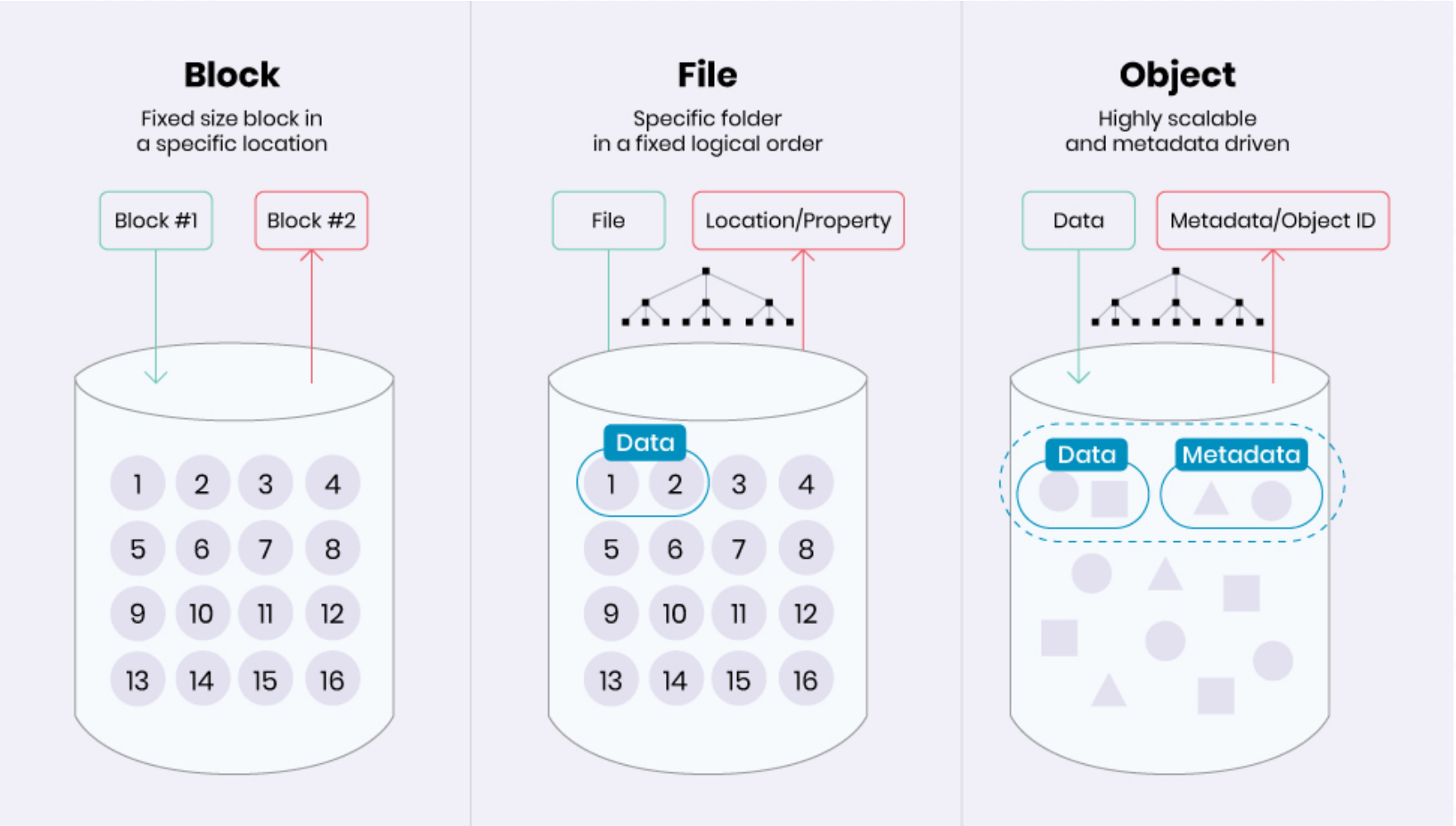
```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: myapp
  namespace: mynamespace
spec:
  to:
    kind: Service
    name: service-blue
  port:
    targetPort: 8080
```



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

NobleProg

Storage



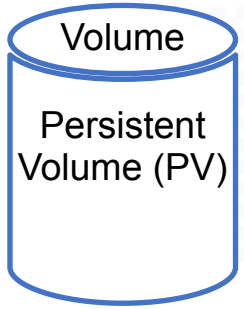
(c) https://accuweb.cloud/blog/cloud-storage-object-v-s-block-v-s-file-storage/

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

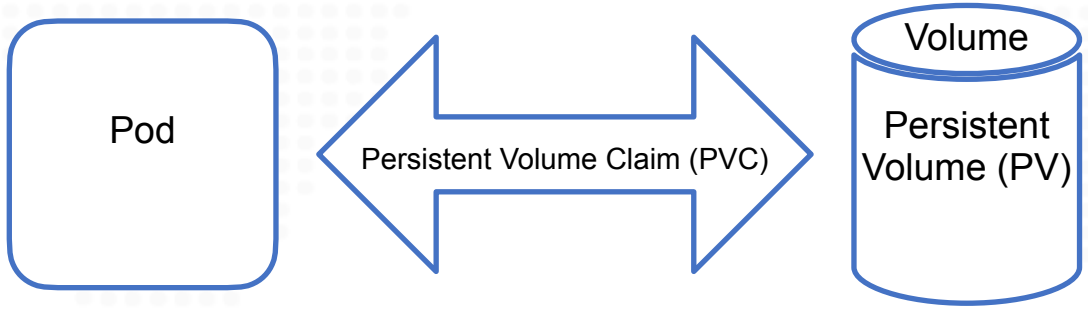


Storage Management

- 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

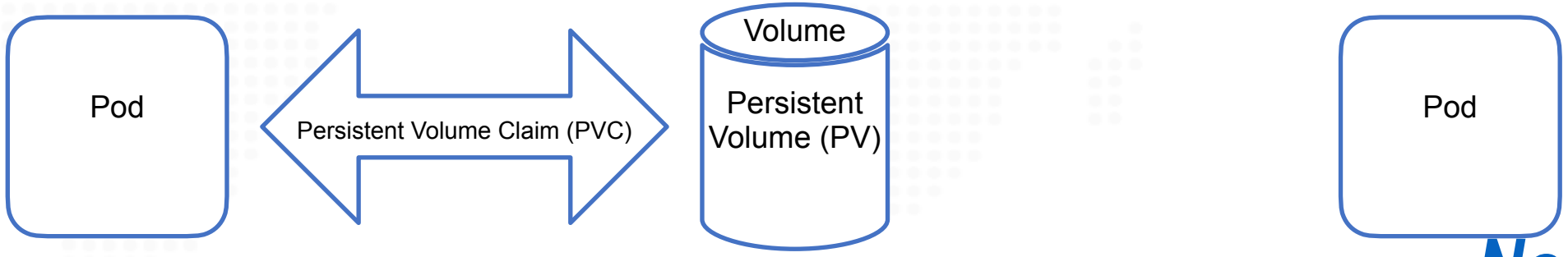
- Access Types: Read Write |Read Only | Once or Many
- Block Storage (RWO) -> Base storage layer
  - Faster, directly -> standard for Database Storage
- File Storage -> for shared data, logs (RWO, RWM)
- Object Storage (S3) -> uploads, backups



NobleProg

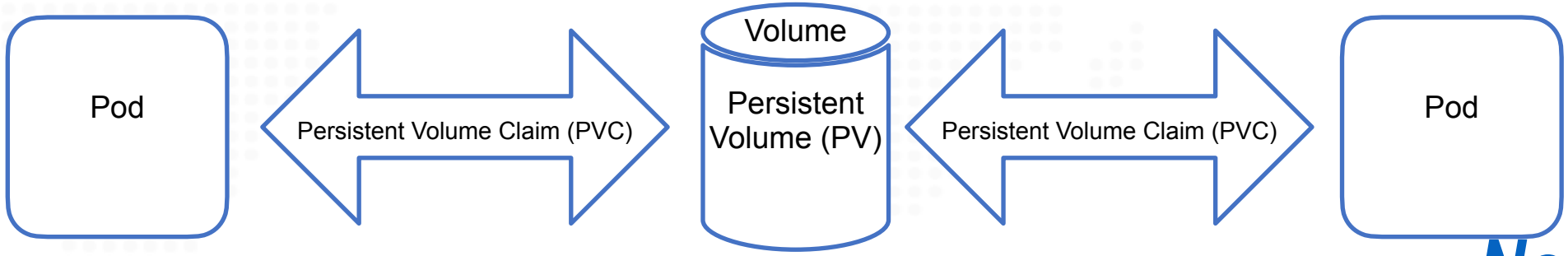
Storage Management

- 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



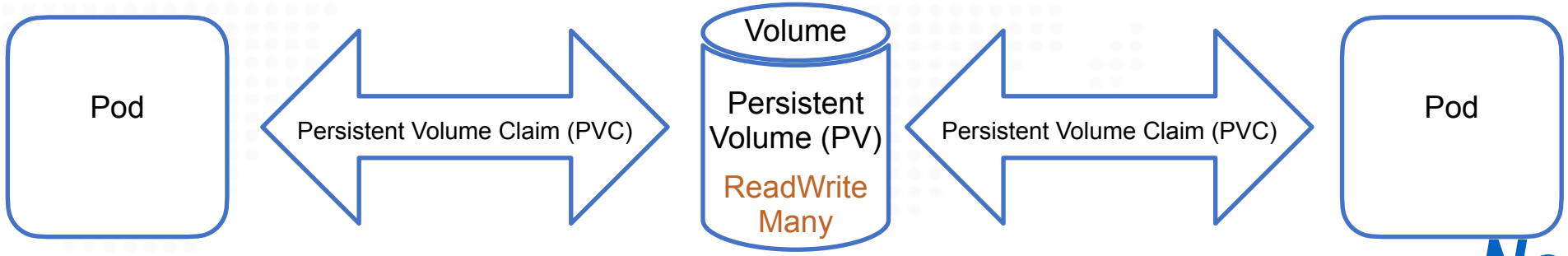
Storage Management

- 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



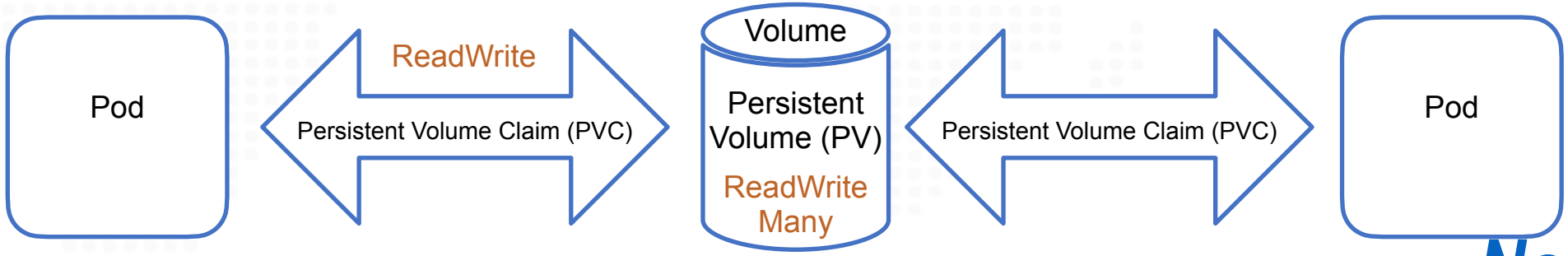
Storage Management

- 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



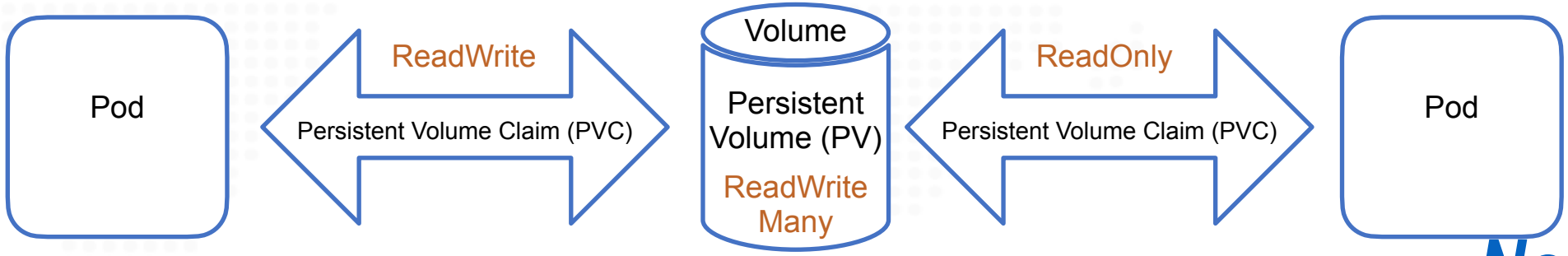
Storage Management

- 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

NobleProg

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
```

NobleProg

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
```

NobleProg

Storage Lab

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

cd openshift/Day2/4.storage

```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
```

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

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

NobleProg

Storage Lab

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

cd openshift/Day2/4.storage

```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
```

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

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

NobleProg

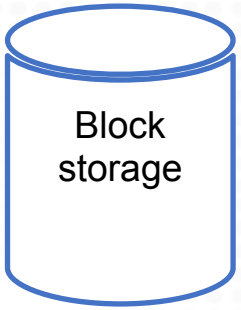
Storage Lab

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

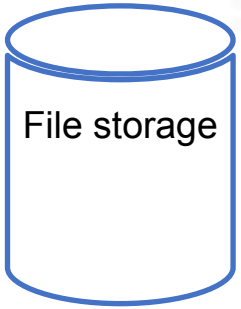
cd openshift/Day2/4.storage

```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
```

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




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



NobleProg

Storage Lab

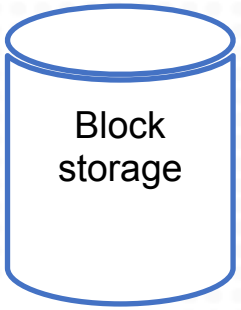
git clone https://github.com/kolinrr/openshift.git

cd openshift/Day2/4.storage

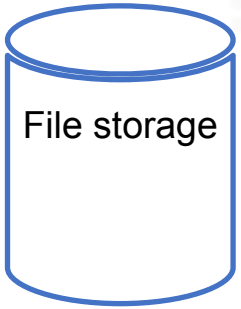
```
$ cd $HOME/openshift/Day2/4.storage

//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
```

```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
```

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




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



NobleProg

Finish!

Questions?