How to Leverage Kubernetes Architecture for Developer Independence

Omer Kahani, Riskified

DEVOPS WORLD by CloudBees

About Me

- Software developer for 10+ years in various teams
- 6 years at Riskified
- 2 years a cloud platform developer
- Lead the DevOps Culture
- Design the developers experience
- Argo community

Kubernetes is an opportunity to make drastic changes

Our Drastic Change

Developers independence = Do more actions on their own

Our Drastic Change

Developers independence = Do more actions on their own

- Reducing learning curve
- Safe & secure process

Developers

Taught To:

- Write code

Not:

- How the production environment is designed
- How to deploy microservices

Developers

Taught To:

- Write Code

Not:

e production environment is ned

deploy micro services

Today

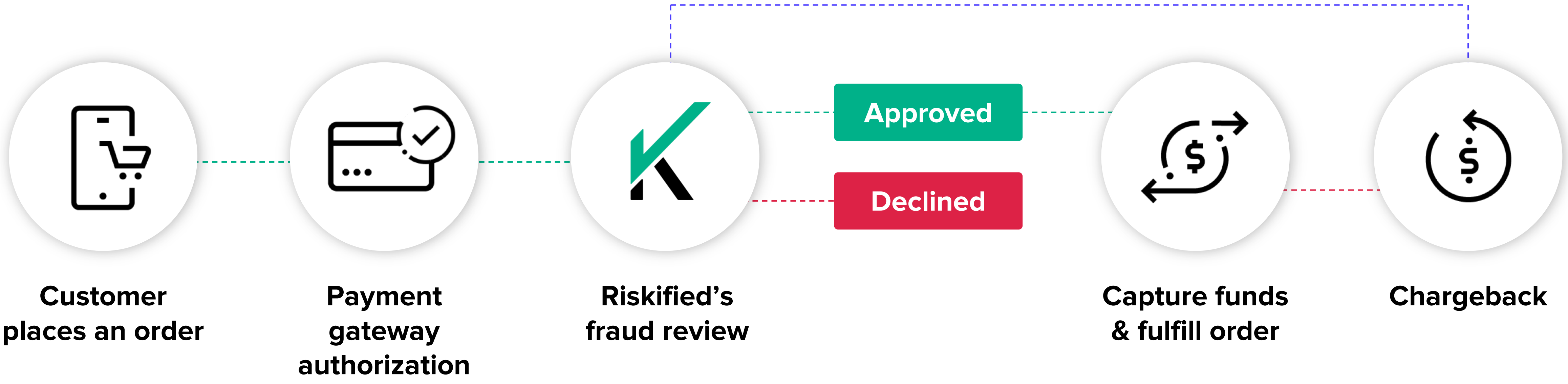
- Creating a new service

Today

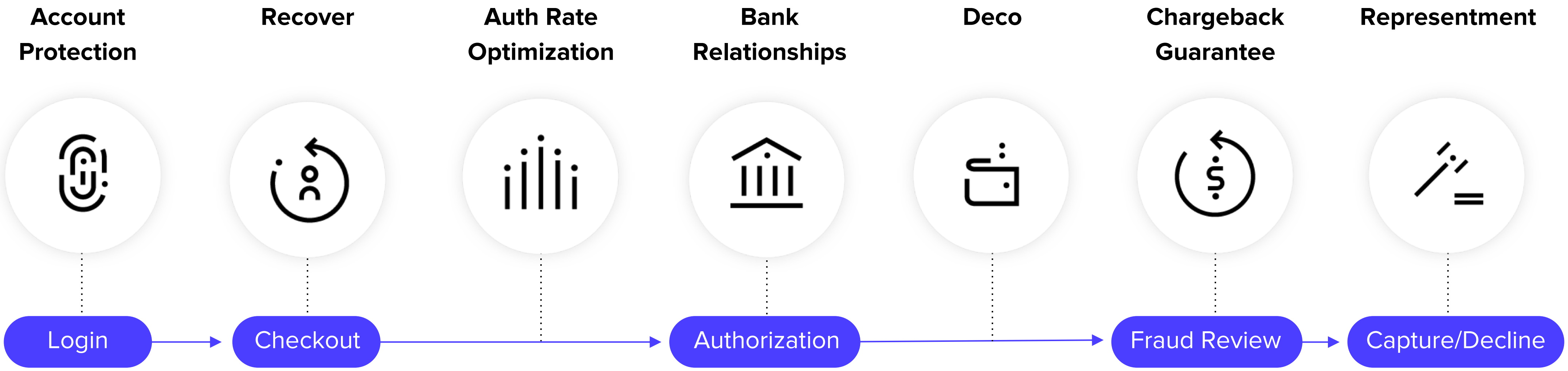
- Creating a new service
- Reduce magic
- Understand, trust, operate

riskified

Riskified develops powerful machine-learning algorithms that recognize legitimate customers.



riskified End-to-End Solution



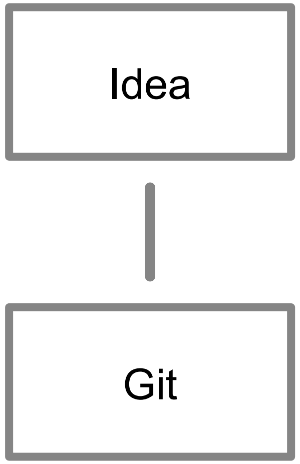
Riskified Growth

- More teams, more services
- Operations is a bottleneck
- We have a chance to change it

New Service Steps

Idea

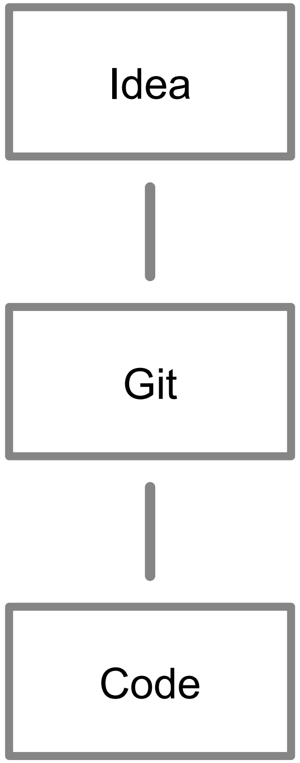
New Service Steps



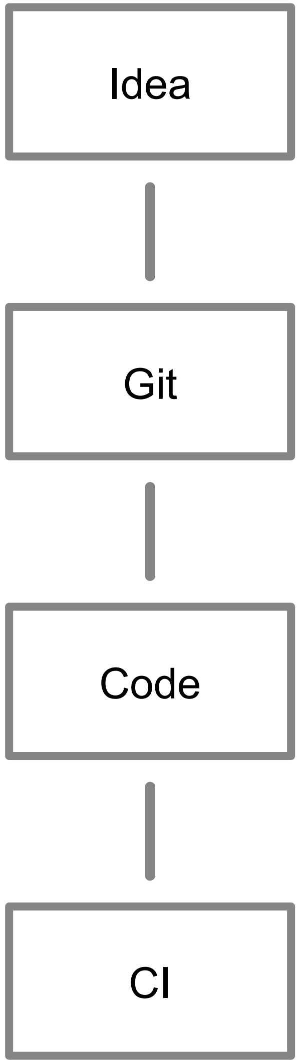
Idea

Git

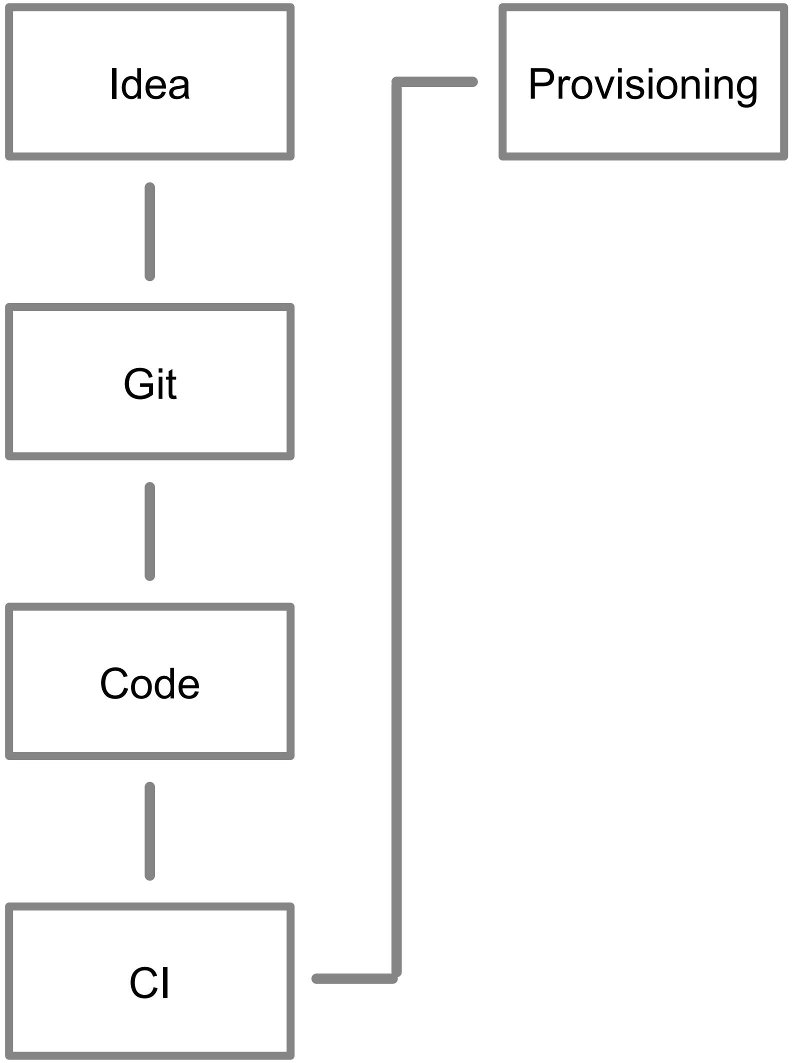
New Service Steps



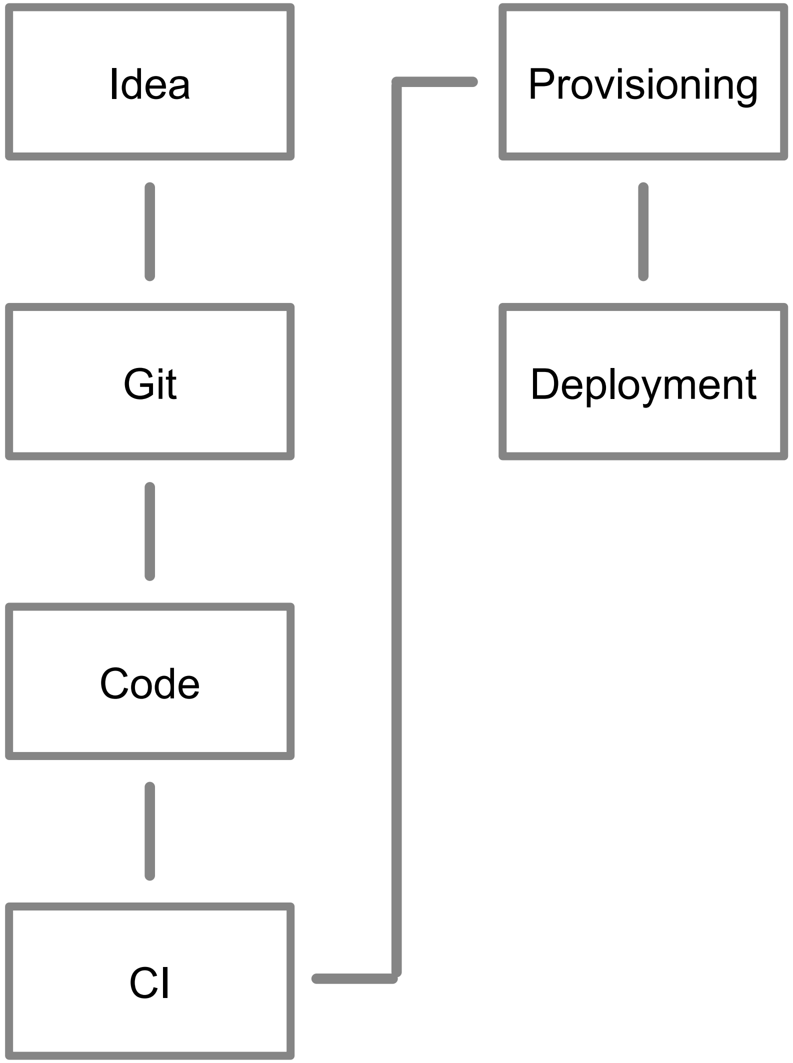
New Service Steps



New Service Steps



New Service Steps



Idea

Git

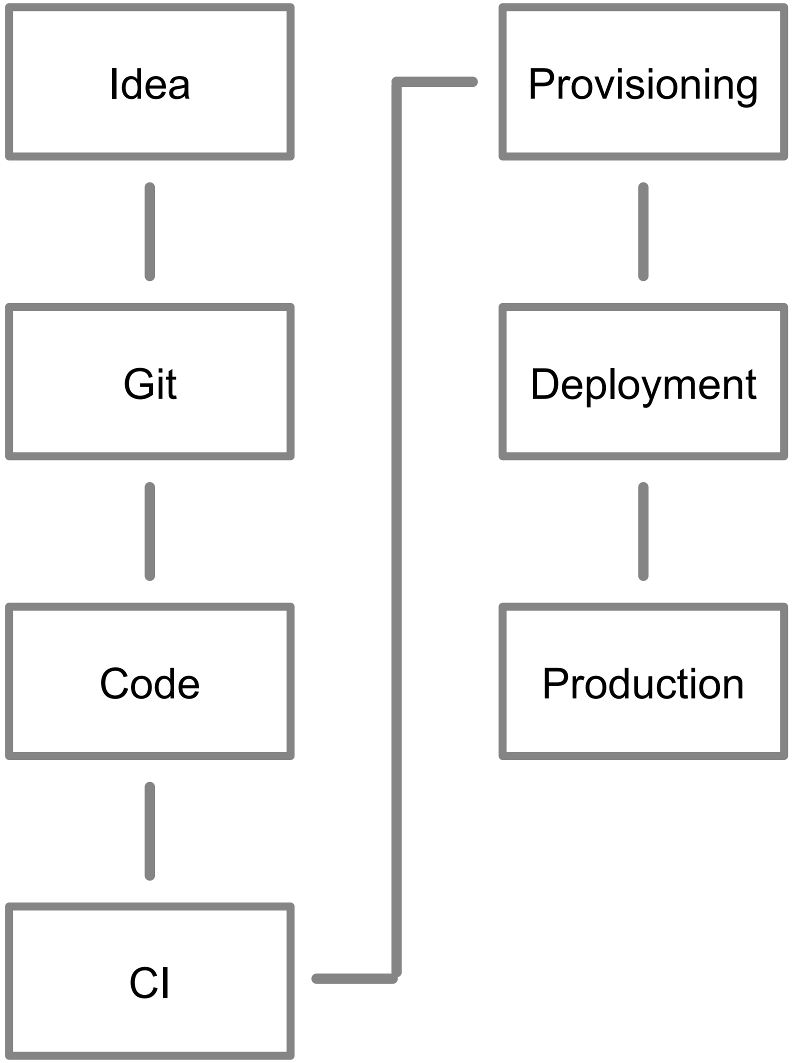
Code

CI

Provisioning

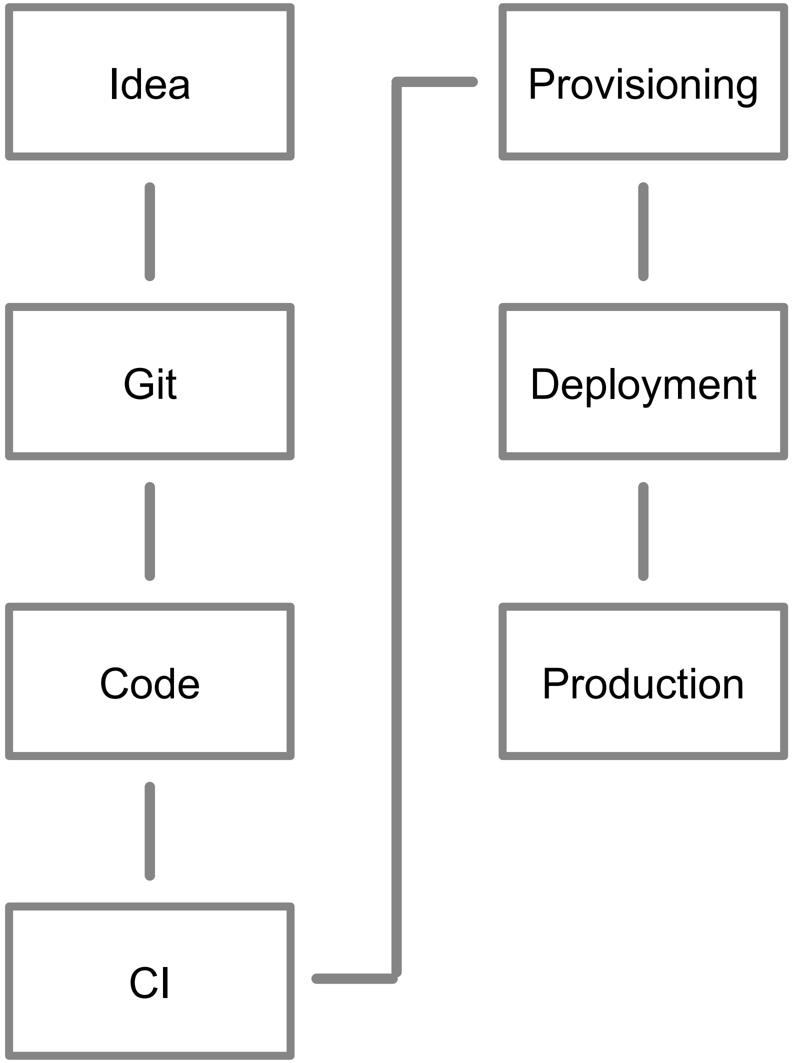
Deployment

New Service Steps



New Service Steps

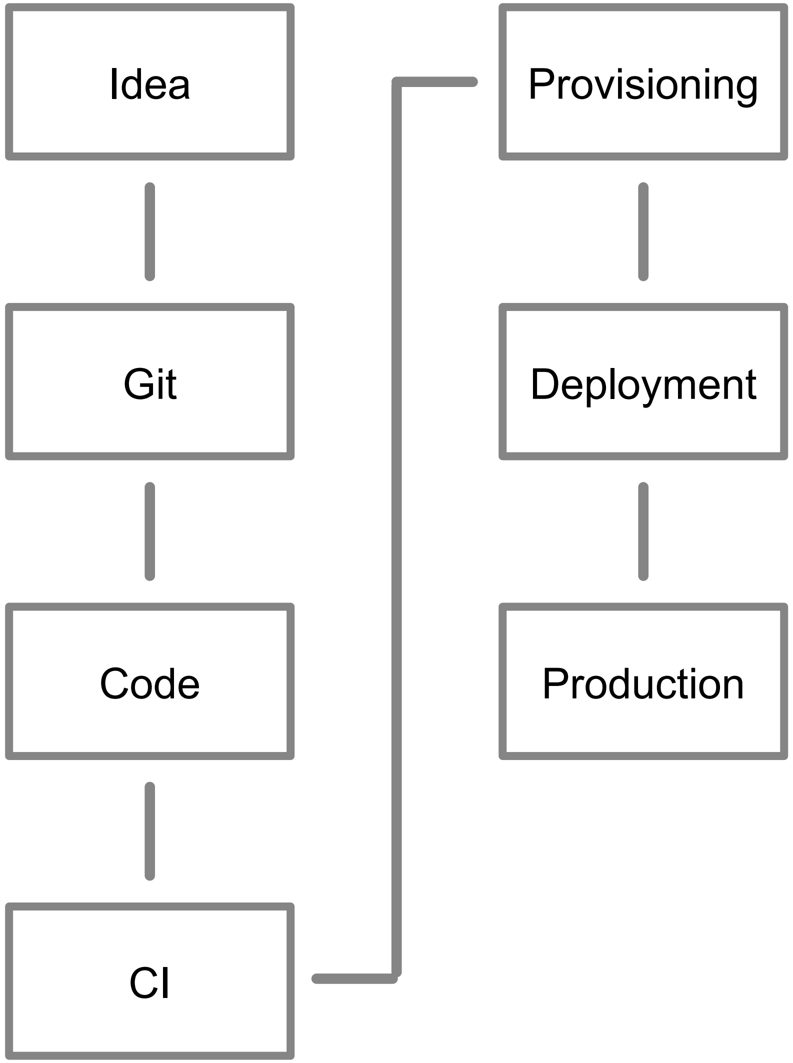
- First: map the steps



New Service Steps

- First: map the steps

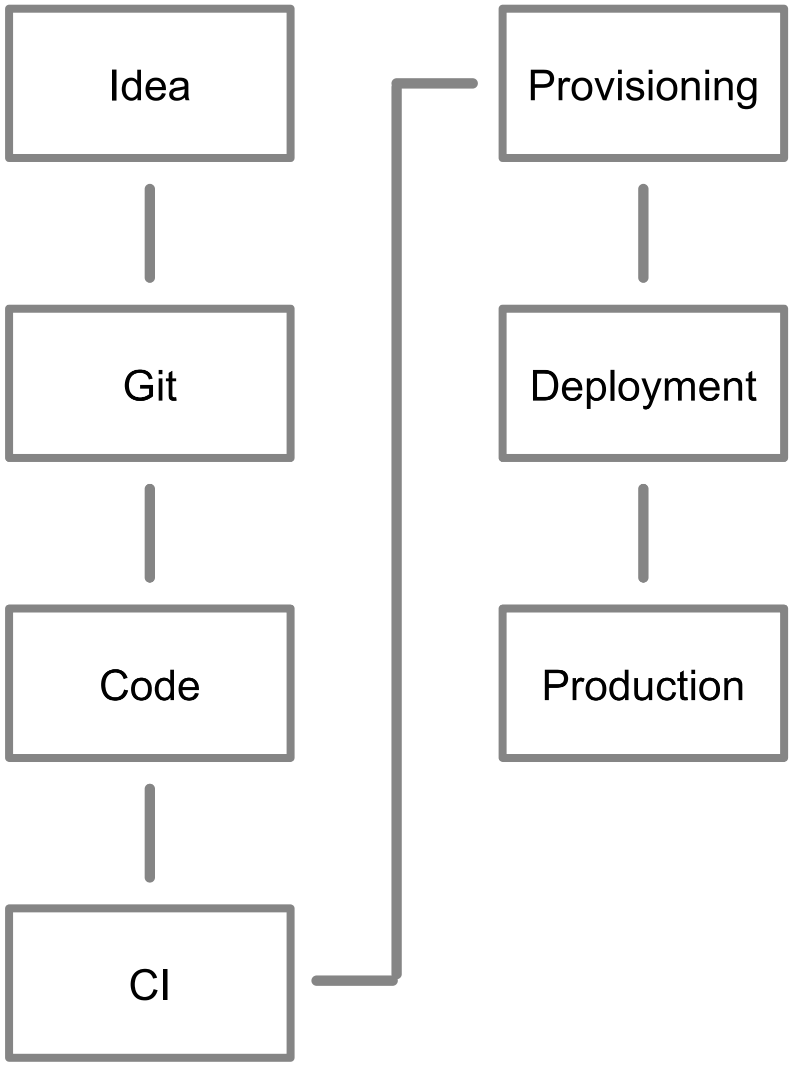
- Goal: developers focus on code



DEVOPSWORLD by CloudBees

New Service Steps

- First: map the steps
- Goal: developers focus on code
- Developers know GitOps - embrace it



Git Repository

- Hard process can’t work

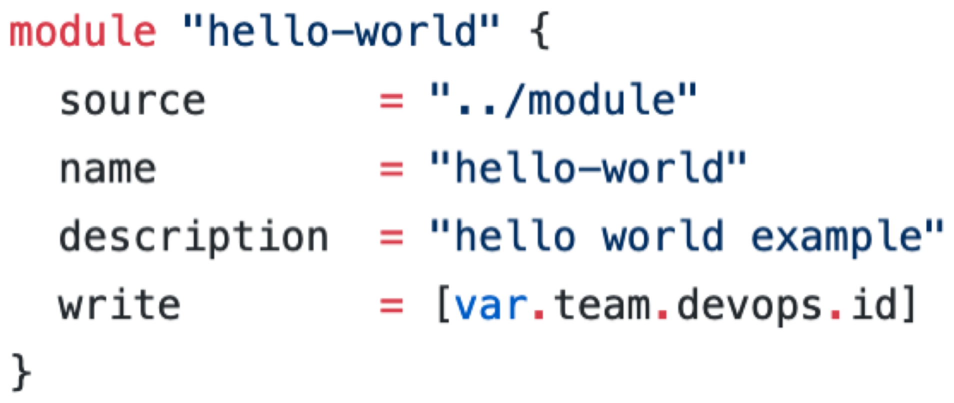
Git Repository

- Hard process can’t work
- Too many options:
  - Branch protection
  - PR checks
  - Name convention
  - Permission

Git Repository

- Hard process can’t work
- Too many options:
  - Branch protection
  - PR checks
  - Name convention
  - Permission
- Terraform internal model

A new git repository - developer experience



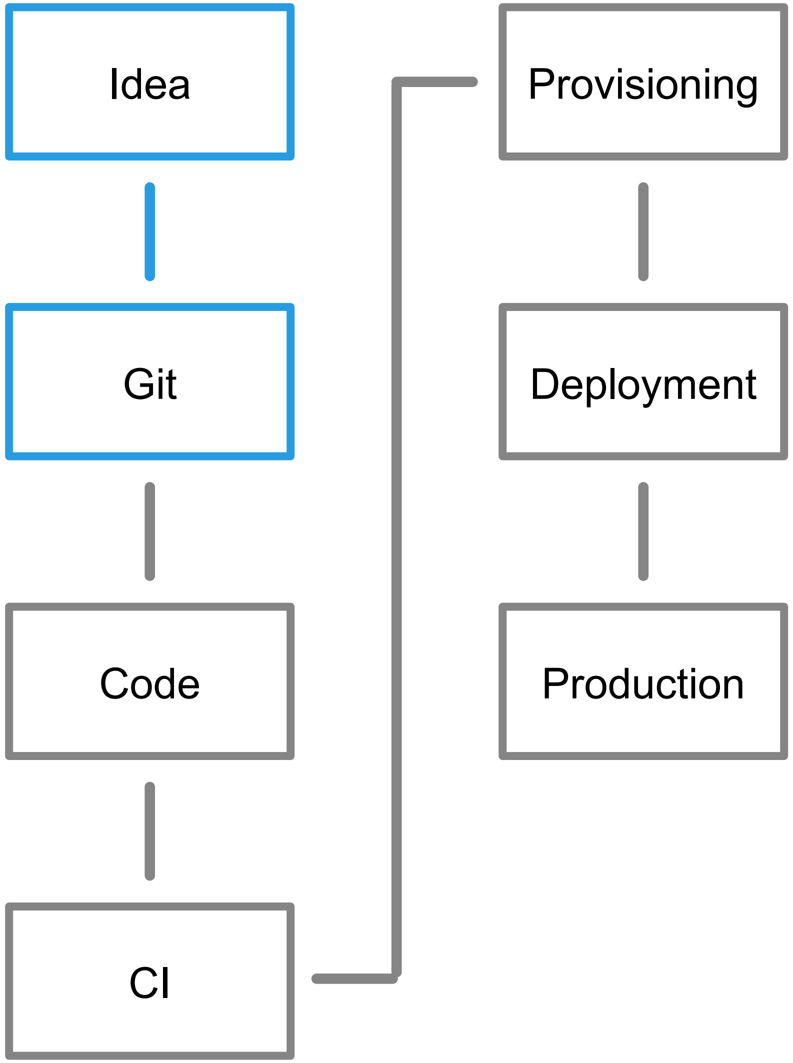
```
module "hello-world" {
  source      = "../module"
  name        = "hello-world"
  description = "hello world example"
  write       = [var.team.devops.id]
}
```

Code & Integration

- Hello World template
  - Running Hello World example
  - Docker file
  - CircleCI file

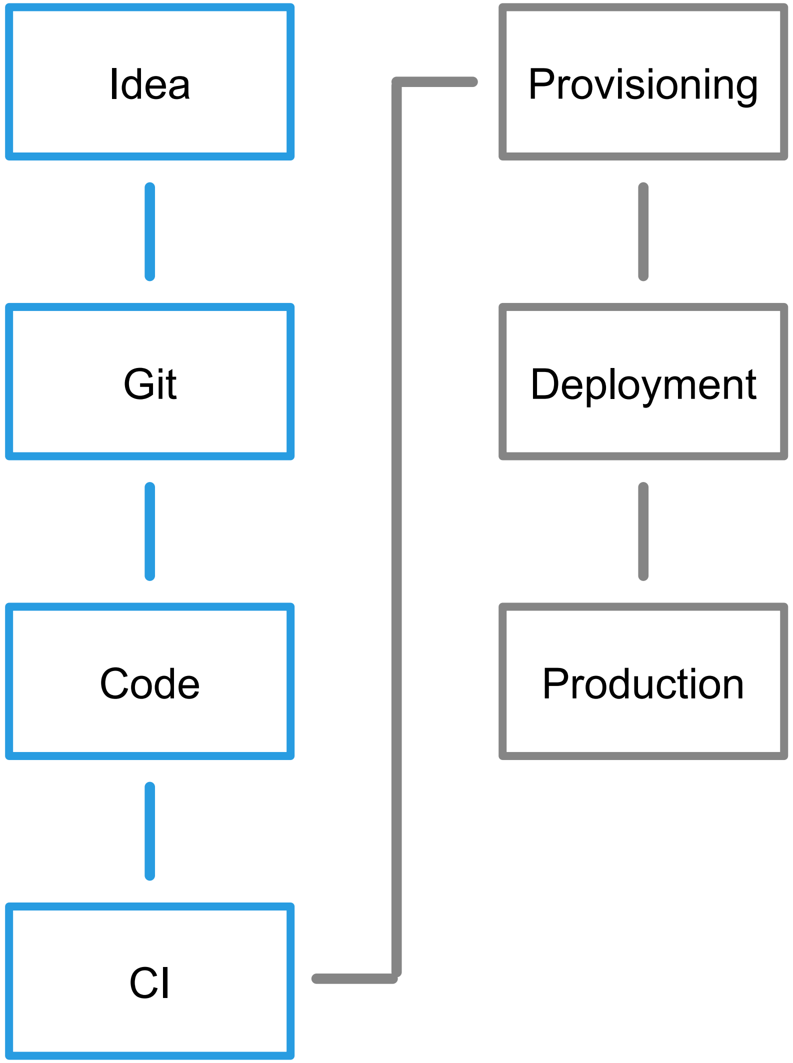
Recap

- Git - Terraform model



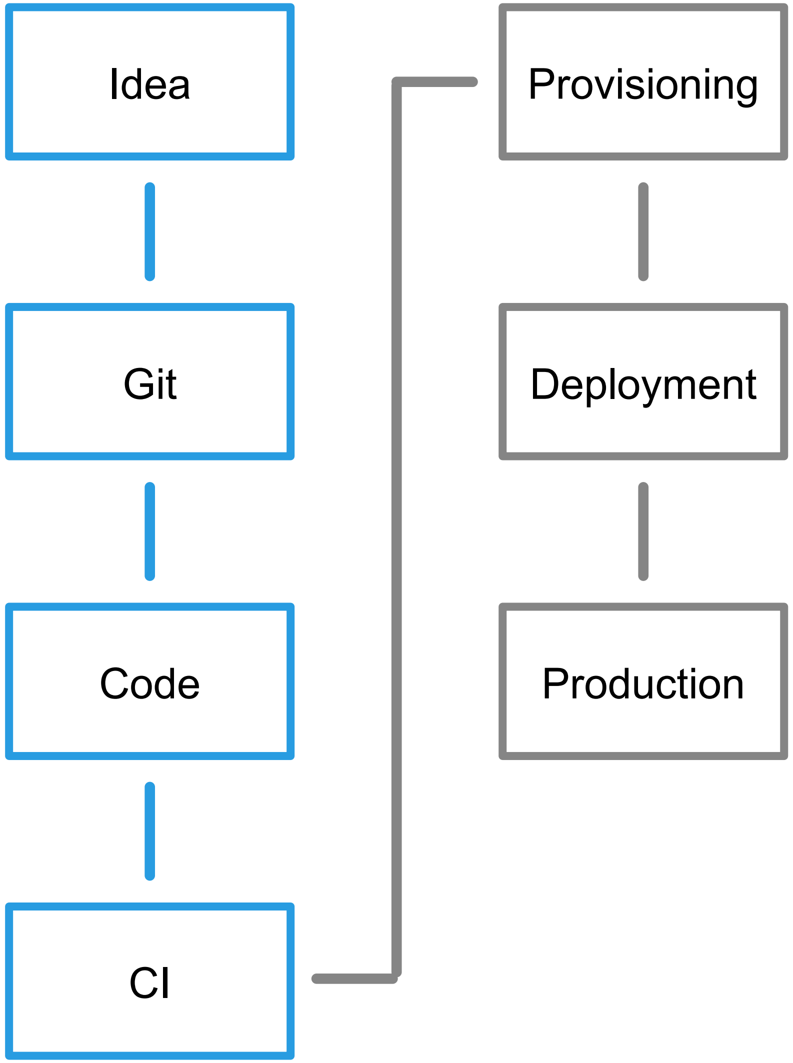
Recap

- Git - Terraform model
- Code - Hello World template
- CI - Hello World template



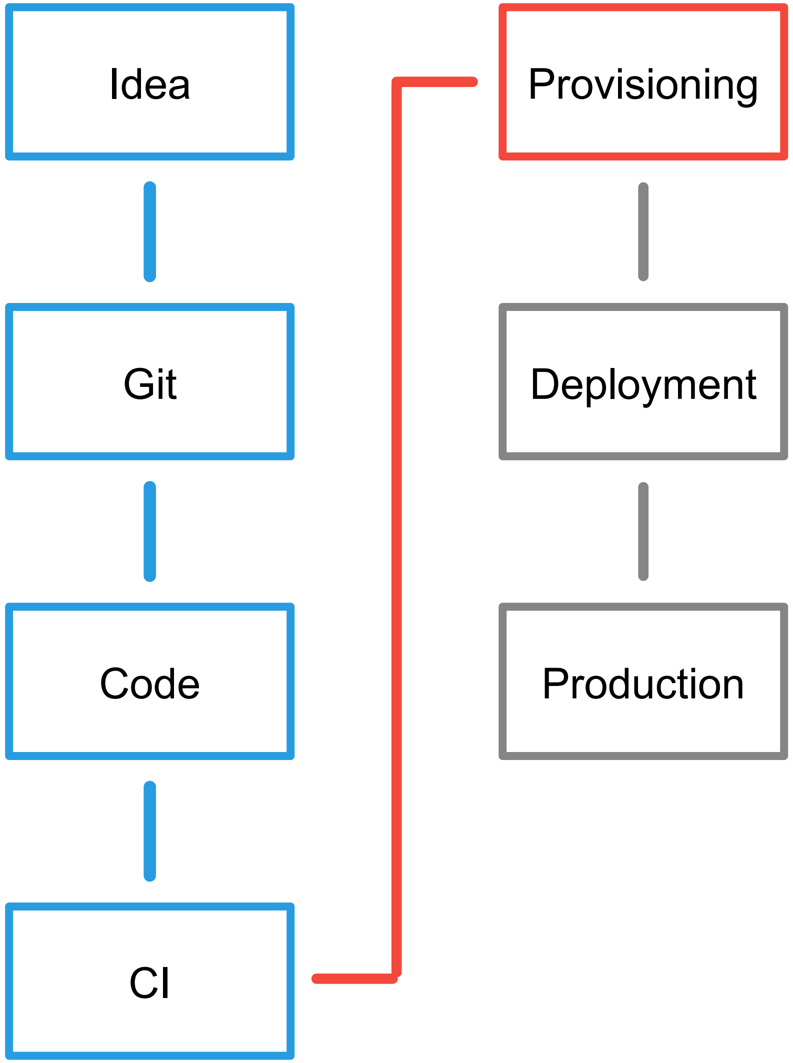
Recap

- Git - Terraform model
- Code - Hello World template
- CI - Hello World template
- Hide complex options
- Enable start coding faster
- Use organization standards


Idea
Git
Code
CI
Provisioning
Deployment
Production

Next Step - Provisioning

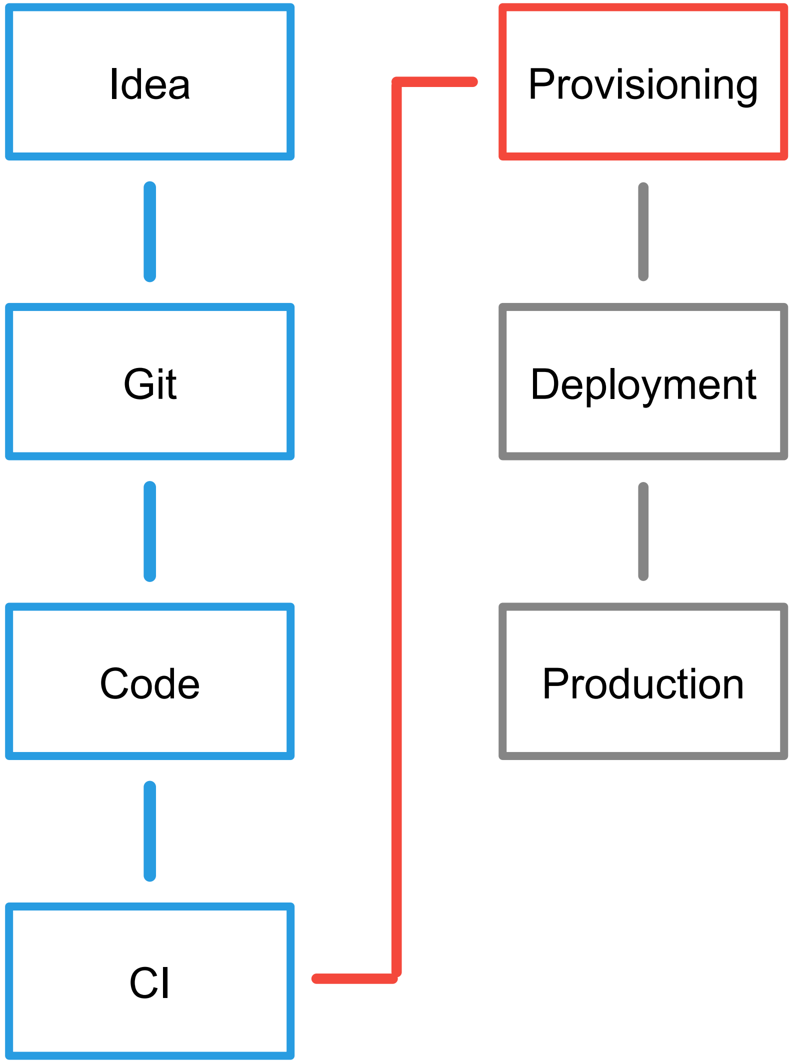
- Setting up the resources for the service
- Move ownership from operations to developers



Next Step - Provisioning

- Setting up the resources for the service
- Move ownership from operations to developers

1. Helm - which
2. Cluster architecture - where



Provisioning in Kubernetes

Provisioning in Kubernetes

- Kubernetes has a lot of objects
  - Deployment
    - Liveness
    - Readiness
    - Strategy
    - Node affinity

Provisioning in Kubernetes

- Kubernetes has a lot of objects
  - Deployment
    - Liveness
    - Readiness
    - Strategy
    - Node affinity
  - Service

Provisioning in Kubernetes

- Kubernetes has a lot of objects
  - Deployment
    - Liveness
    - Readiness
    - Strategy
    - Node affinity
  - Service
  - Service account
  - Secret
  - Volume
  - PDB
  - HPA
  - VPA
  - Ingress

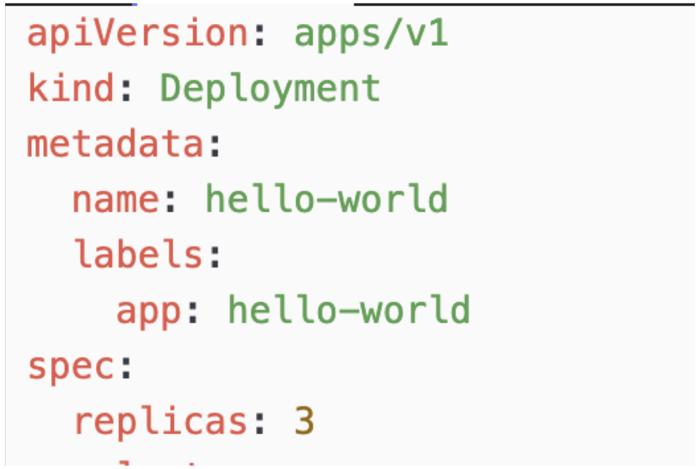
Provisioning in Kubernetes

- Kubernetes has a lot of objects
  - Deployment
    - Liveness
    - Readiness
    - Strategy
    - Node affinity
  - Service
  - Service account
  - Secret
  - Volume
  - PDB
  - HPA
  - VPA
  - Ingress
  - ...

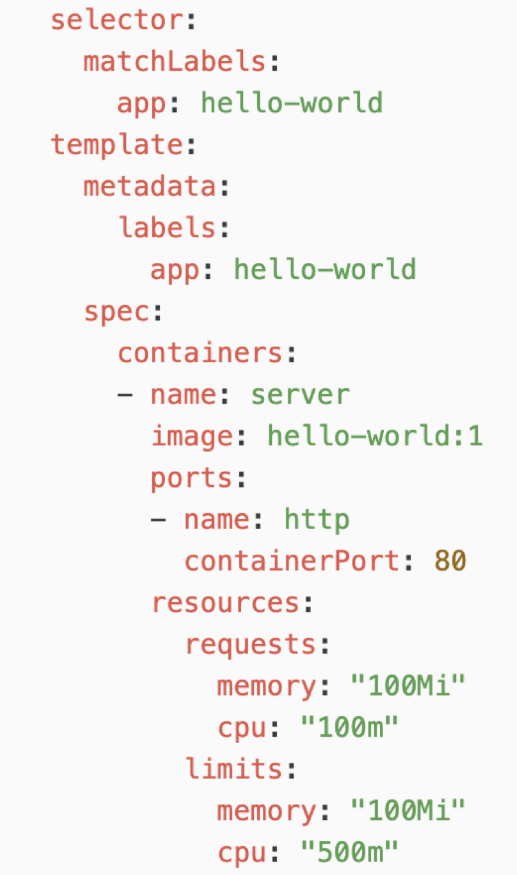
Provisioning - Hello World Example

- Hello World example
  - **Deployment - pods & containers**
  - **Service - load-balancer**

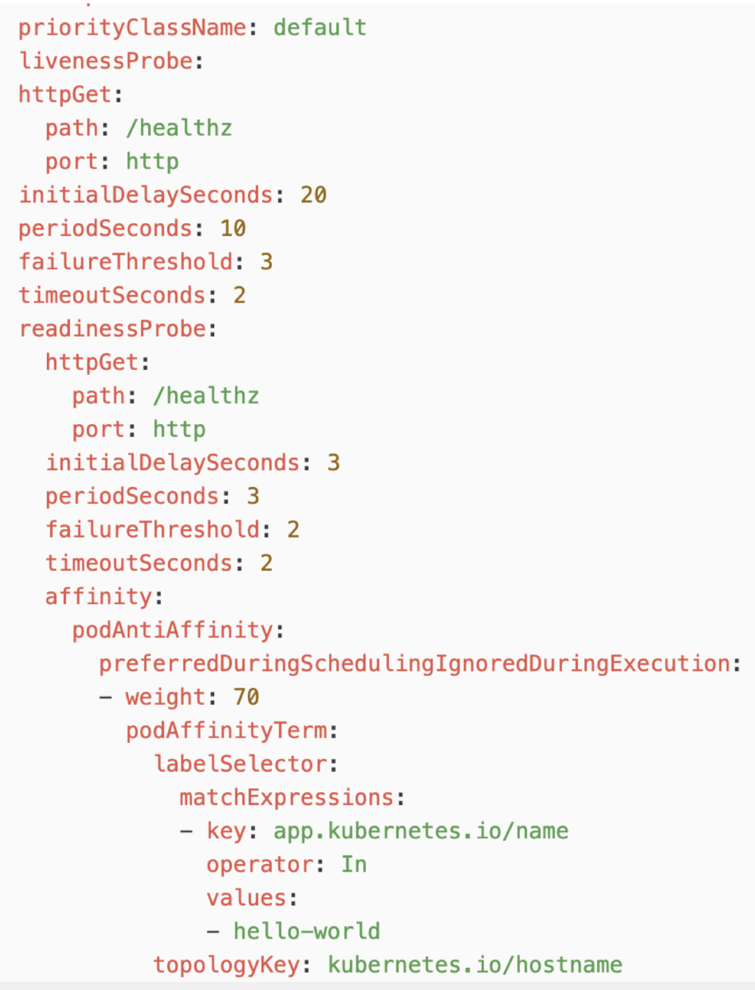
Provisioning - The Problem



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
  labels:
    app: hello-world
spec:
  replicas: 3
```

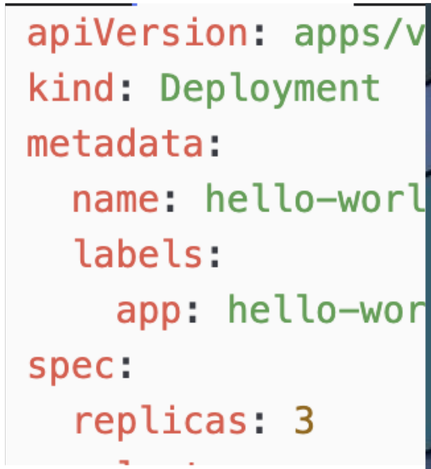


```yaml
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - name: server
        image: hello-world:1
        ports:
        - name: http
          containerPort: 80
        resources:
          requests:
            memory: "100Mi"
            cpu: "100m"
          limits:
            memory: "100Mi"
            cpu: "500m"
```

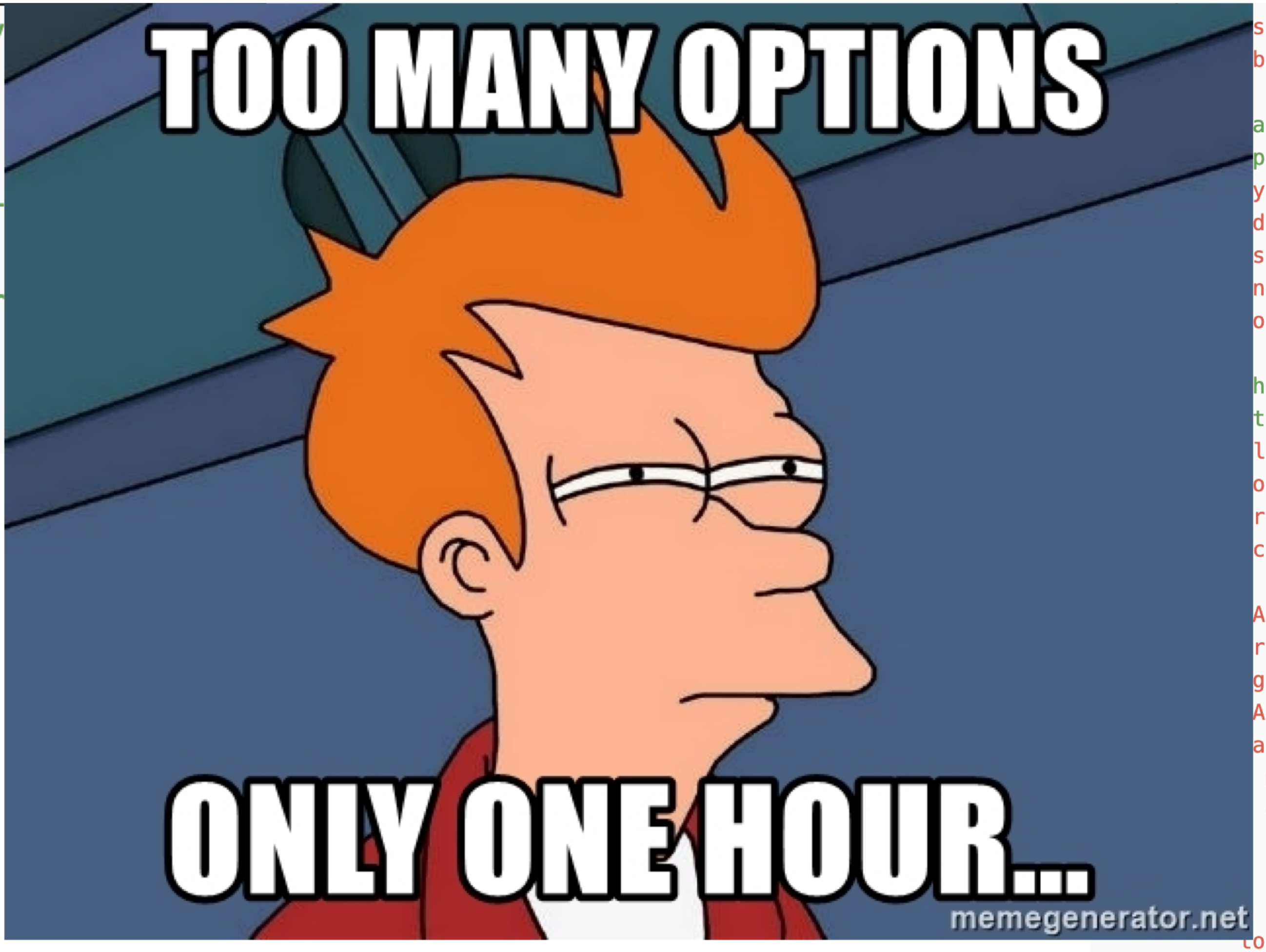


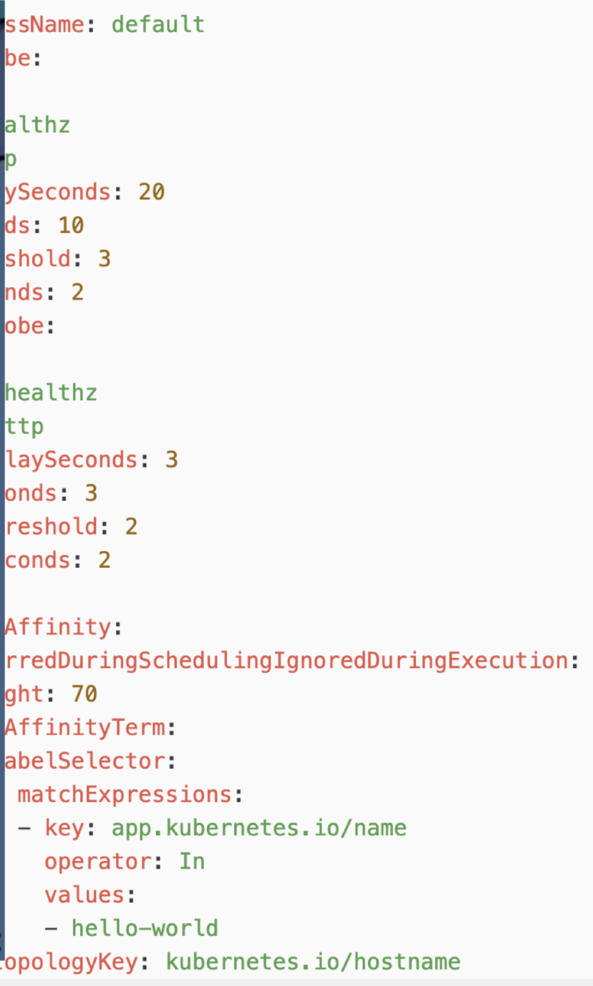
```yaml
priorityClassName: default
livenessProbe:
httpGet:
  path: /healthz
  port: http
initialDelaySeconds: 20
periodSeconds: 10
failureThreshold: 3
timeoutSeconds: 2
readinessProbe:
  httpGet:
    path: /healthz
    port: http
  initialDelaySeconds: 3
  periodSeconds: 3
  failureThreshold: 2
  timeoutSeconds: 2
  affinity:
    podAntiAffinity:
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 70
        podAffinityTerm:
          labelSelector:
            matchExpressions:
            - key: app.kubernetes.io/name
              operator: In
              values:
              - hello-world
          topologyKey: kubernetes.io/hostname
```

Provisioning - The Problem



```yaml
apiVersion: apps/v
kind: Deployment
metadata:
  name: hello-worl
  labels:
    app: hello-wor
spec:
  replicas: 3
```





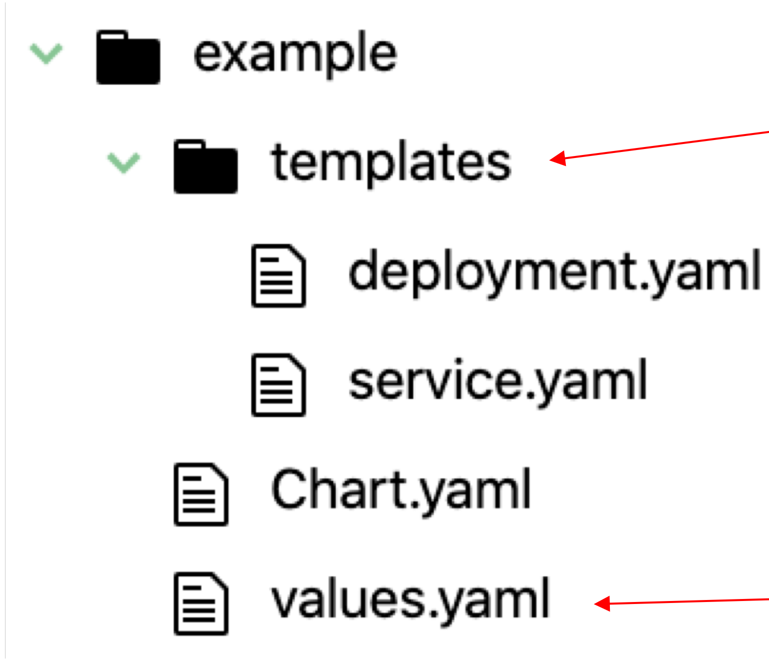
```yaml
ssName: default
be:

althz
p
ySeconds: 20
ds: 10
shold: 3
nds: 2
obe:

healthz
ttp
laySeconds: 3
onds: 3
reshold: 2
conds: 2

Affinity:
rredDuringSchedulingIgnoredDuringExecution:
ght: 70
AffinityTerm:
abelSelector:
 matchExpressions:
 - key: app.kubernetes.io/name
   operator: In
   values:
   - hello-world
opologyKey: kubernetes.io/hostname
```

DEVOPSWORLD by CloudBees

Helm - Provisioning Solution

- Helm is a package manager:
  - Template
  - Manage
- CNCF graduated project

Helm Chart

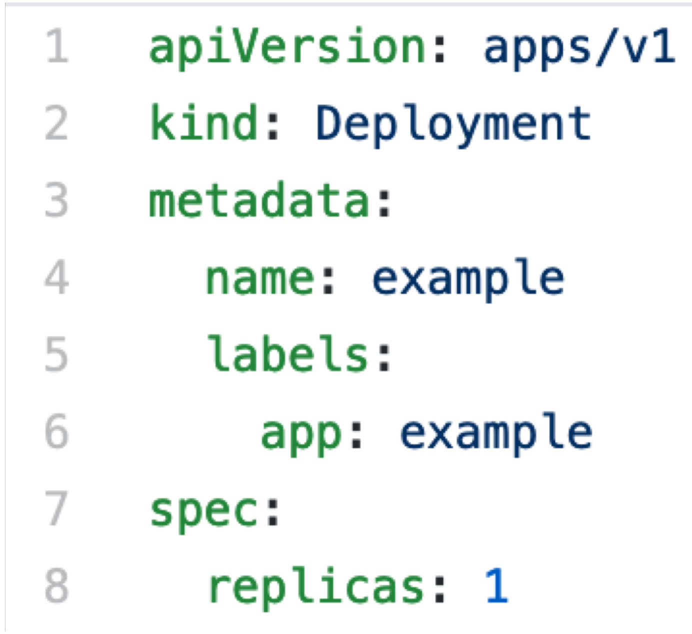


```
example
    templates
        deployment.yaml
        service.yaml
    Chart.yaml
    values.yaml
```

Template YAMLs

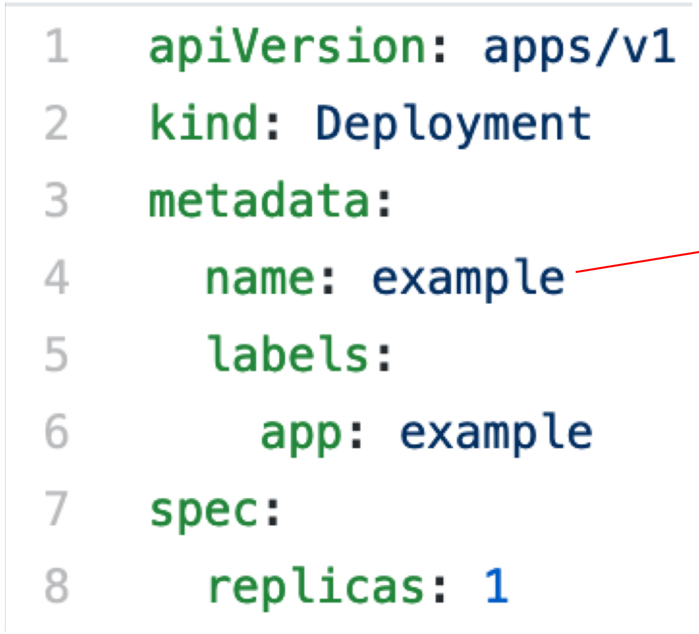
Values files

Helm Abstract Chart



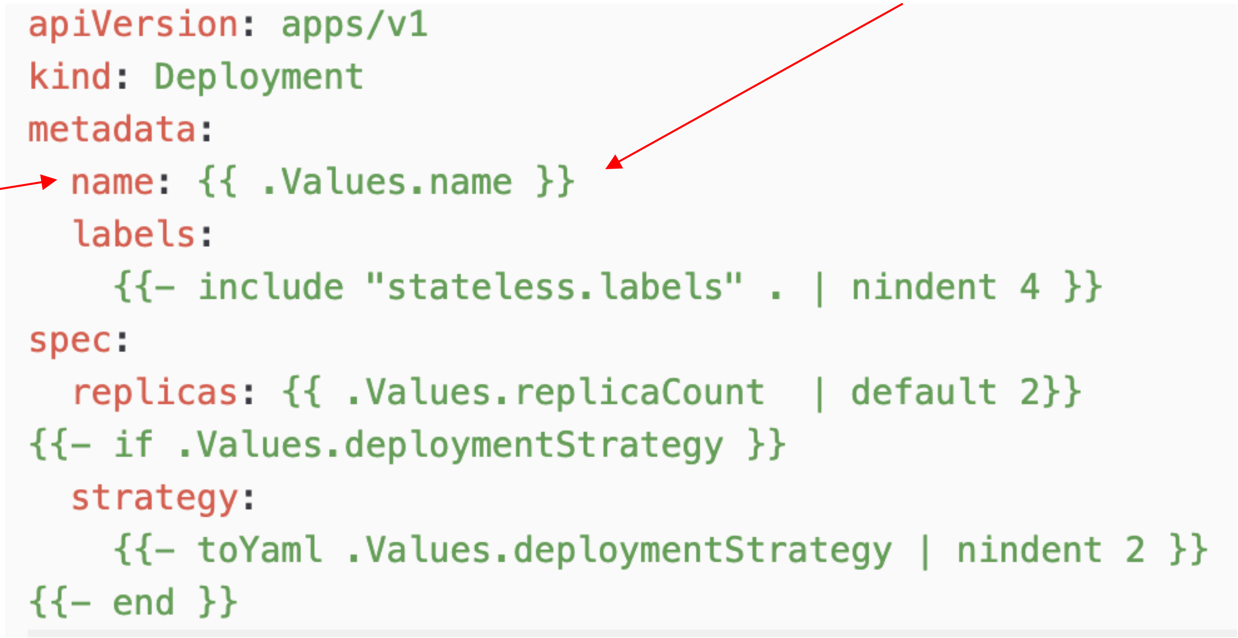
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example
  labels:
    app: example
spec:
  replicas: 1
```

Helm Abstract Chart



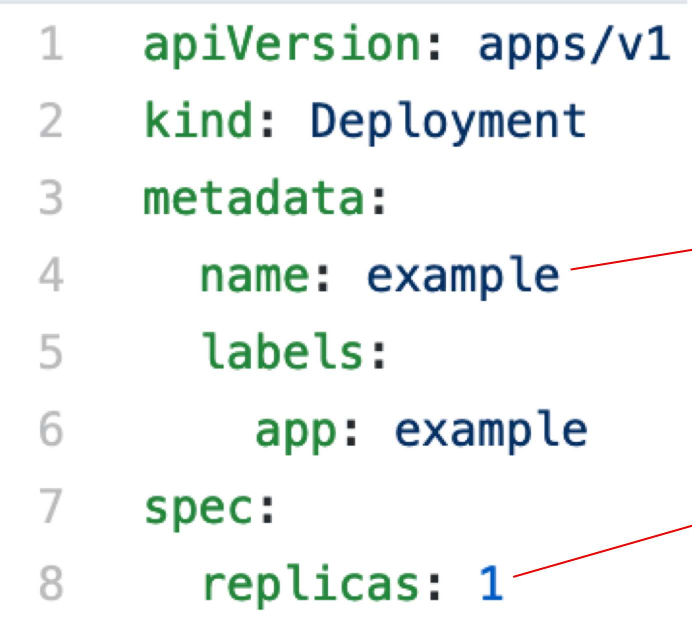
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example
  labels:
    app: example
spec:
  replicas: 1
```

Mandatory value

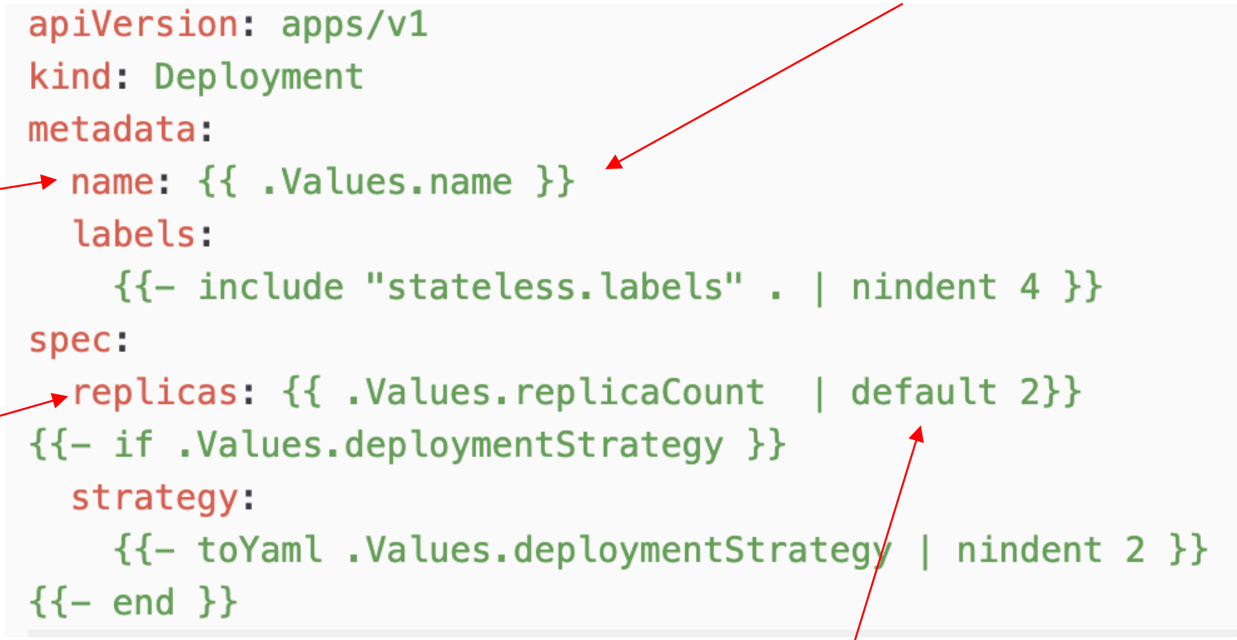


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ .Values.name }}
  labels:
    {{- include "stateless.labels" . | nindent 4 }}
spec:
  replicas: {{ .Values.replicaCount  | default 2}}
{{- if .Values.deploymentStrategy }}
  strategy:
    {{- toYaml .Values.deploymentStrategy | nindent 2 }}
{{- end }}
```

Helm Abstract Chart



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example
  labels:
    app: example
spec:
  replicas: 1
```



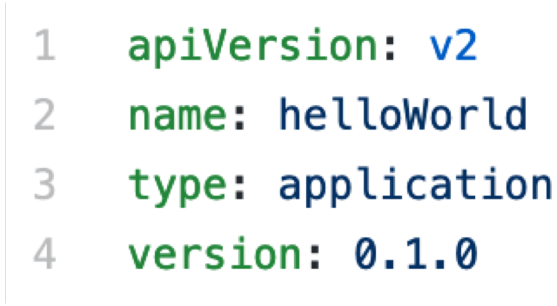
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ .Values.name }}
  labels:
    {{- include "stateless.labels" . | nindent 4 }}
spec:
  replicas: {{ .Values.replicaCount  | default 2}}
{{- if .Values.deploymentStrategy }}
  strategy:
    {{- toYaml .Values.deploymentStrategy | nindent 2 }}
{{- end }}
```

Mandatory value

Default value

Helm Abstract Chart - Use Case

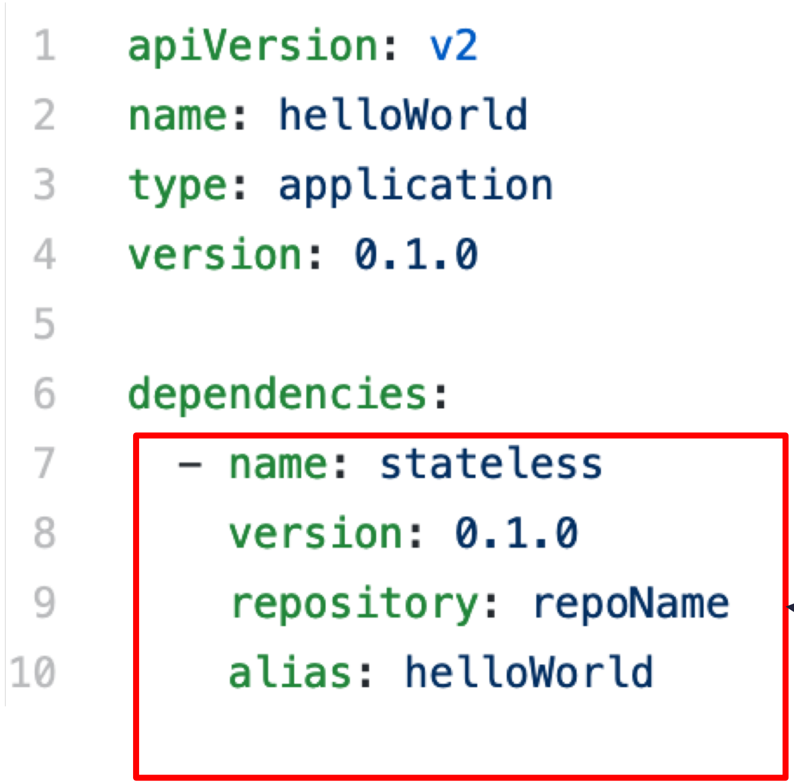
chart.yaml



```yaml
apiVersion: v2
name: helloWorld
type: application
version: 0.1.0
```

Helm Abstract Chart - Use Case

chart.yaml



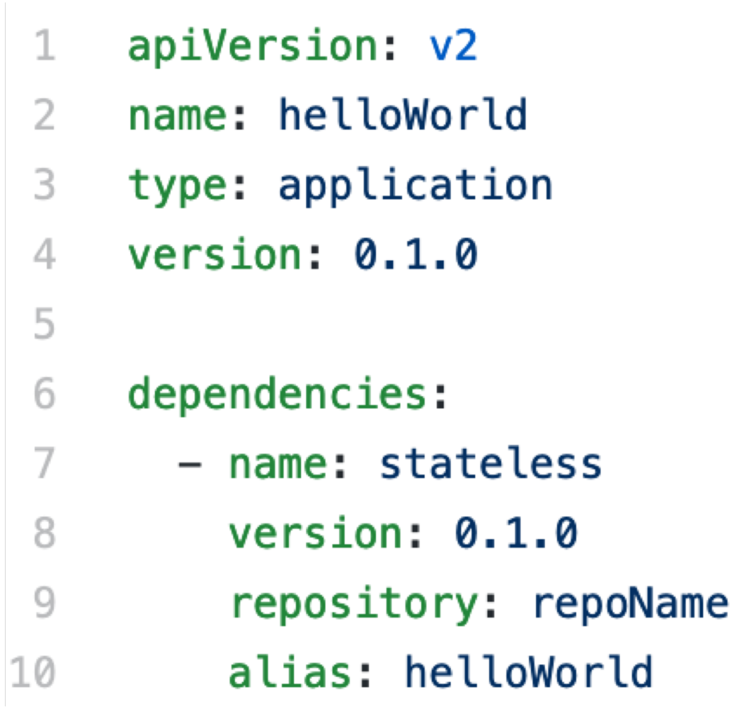
```yaml
apiVersion: v2
name: helloWorld
type: application
version: 0.1.0

dependencies:
  - name: stateless
    version: 0.1.0
    repository: repoName
    alias: helloWorld
```

Require
other chart

DEVOPSWORLD
by CloudBees

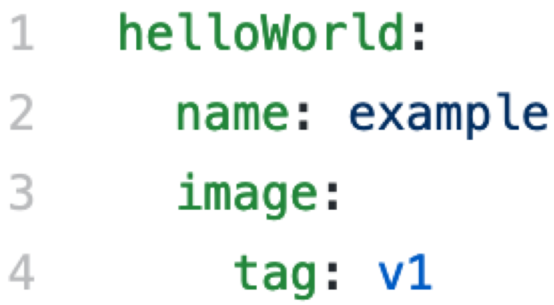
Helm Abstract Chart - Use Case

chart.yaml



```yaml
apiVersion: v2
name: helloWorld
type: application
version: 0.1.0

dependencies:
  - name: stateless
    version: 0.1.0
    repository: repoName
    alias: helloWorld
```

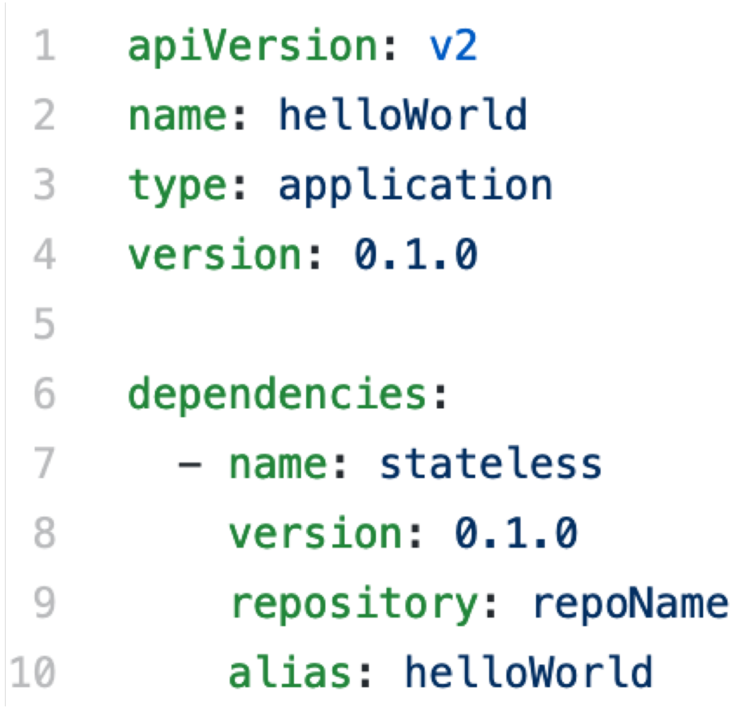
values.yaml



```yaml
helloWorld:
  name: example
  image:
    tag: v1
```

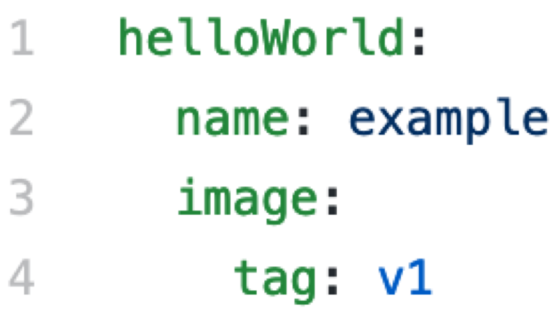
Helm Abstract Chart - Use Case

chart.yaml



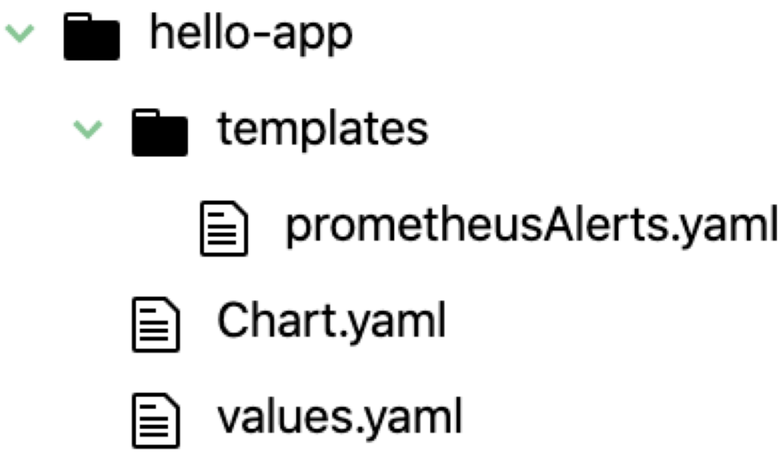
```yaml
apiVersion: v2
name: helloWorld
type: application
version: 0.1.0

dependencies:
  - name: stateless
    version: 0.1.0
    repository: repoName
    alias: helloWorld
```

values.yaml



```yaml
helloWorld:
  name: example
  image:
    tag: v1
```

Extension



```
hello-app
    templates
        prometheusAlerts.yaml
    Chart.yaml
    values.yaml
```

Helm Abstract Chart - API

- Helm charts are the platform API
- Semantic versioning

Helm Abstract Chart - API

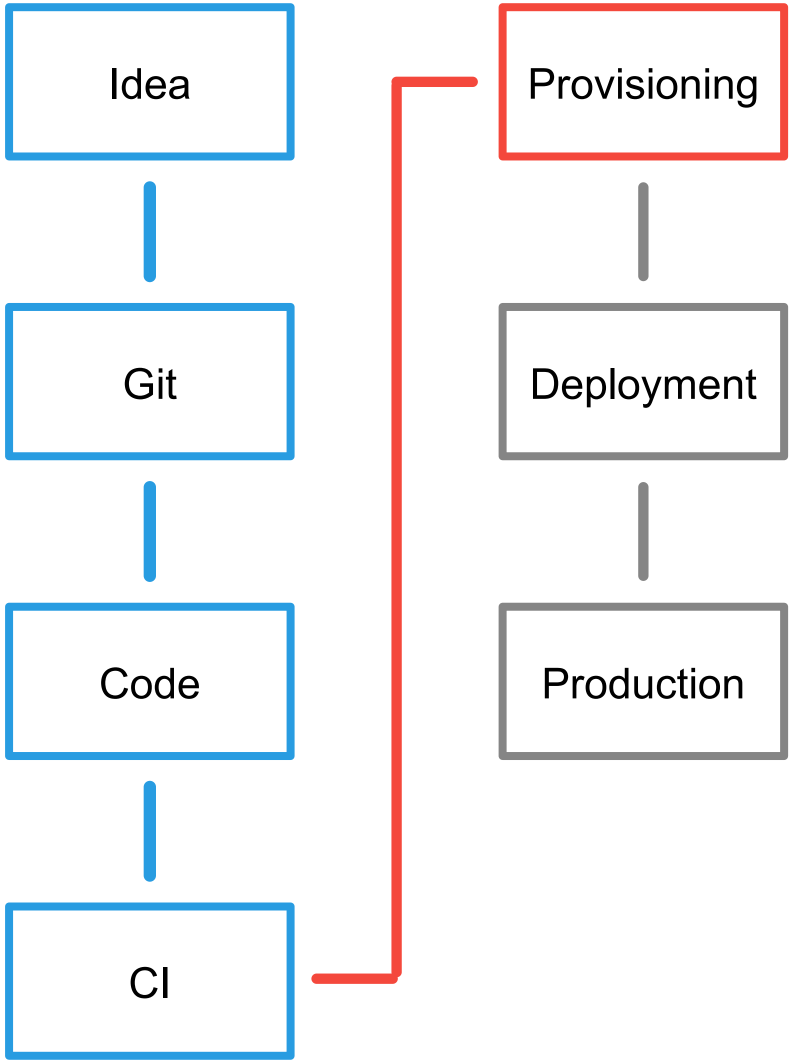
- Helm charts are the platform API
- Semantic versioning
- Guidelines

1. New objects should be added only when absolutely necessary by more than one consumer.
2. Variable name should be in camelCase

Next Step - Provisioning

- Setting up the resources for the service
- Move ownership from operations to developers

1. Helm
2. **Cluster architecture**



Cluster Architecture

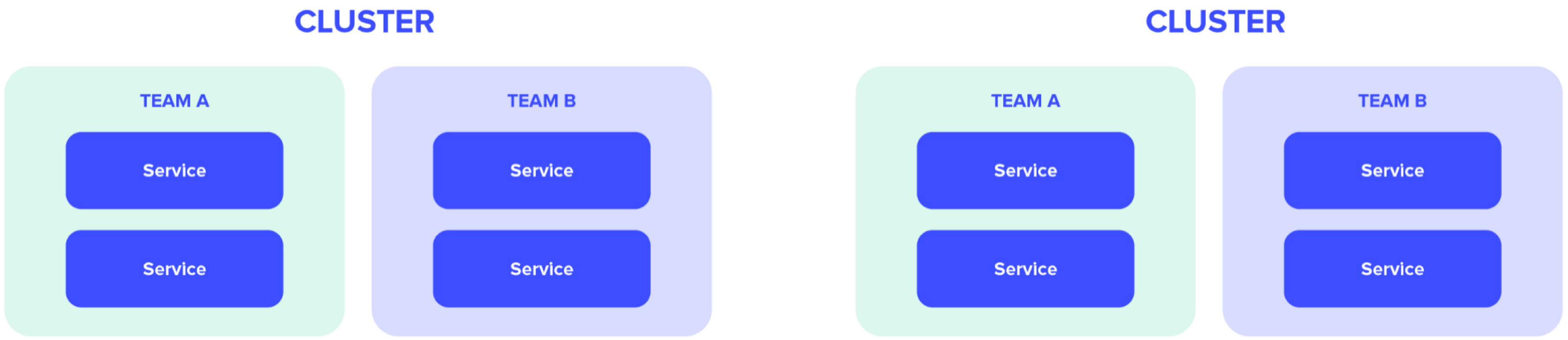
- Security & reliability
- Developers independence

Cluster Architecture

- Cluster per environment
- Namespace per team
- Namespace limits

Cluster Architecture - Cluster per Environment

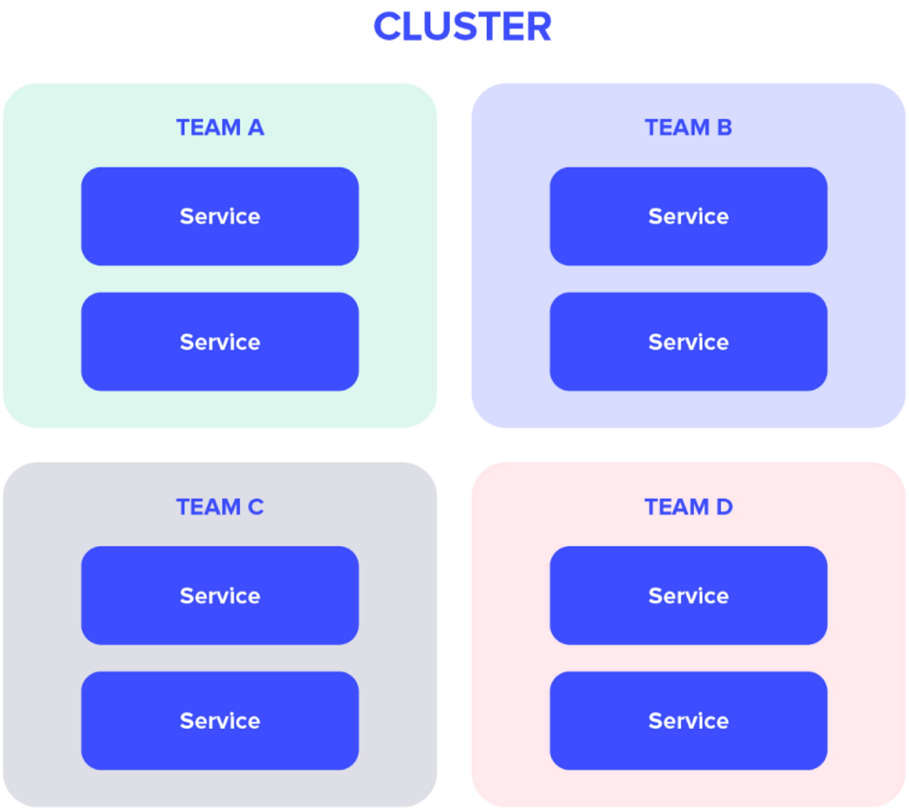
- Staging can’t cause problems in production



Cluster Architecture - Namespace per Team

- Separate services by owners

- Developers can only change services they own

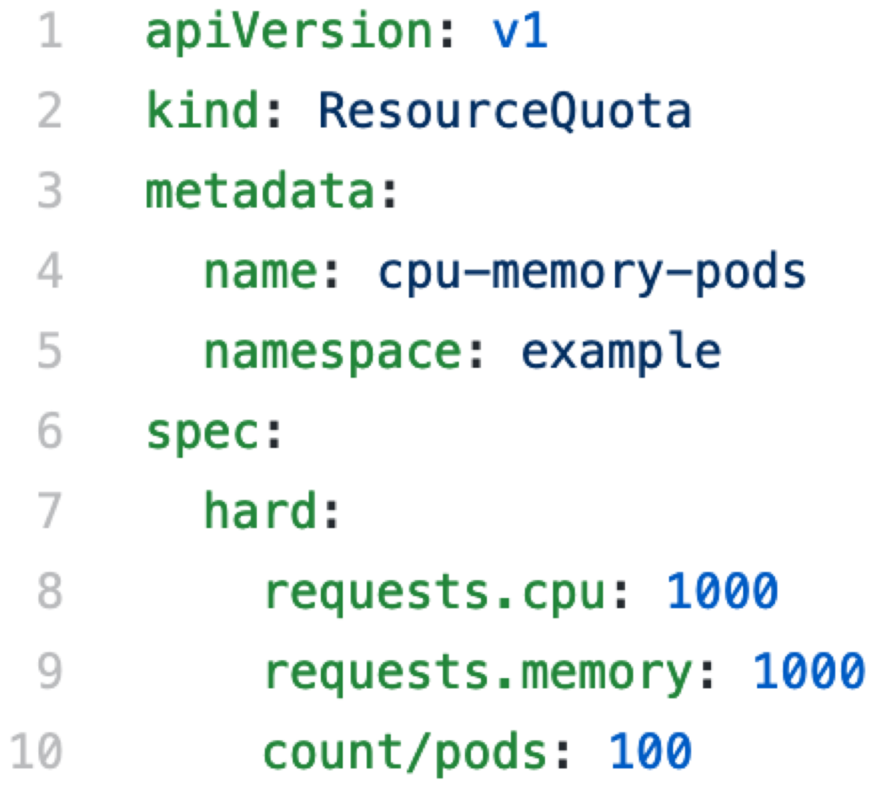


Cluster Architecture - Namespace Limits

- Limit CPU, Memory, Pods

- Prevent mistakes

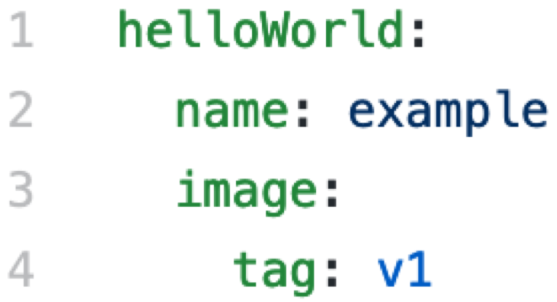
Limit Example



```
apiVersion: v1
kind: ResourceQuota
metadata:
  name: cpu-memory-pods
  namespace: example
spec:
  hard:
    requests.cpu: 1000
    requests.memory: 1000
    count/pods: 100
```

Value File per Environment

values.yaml

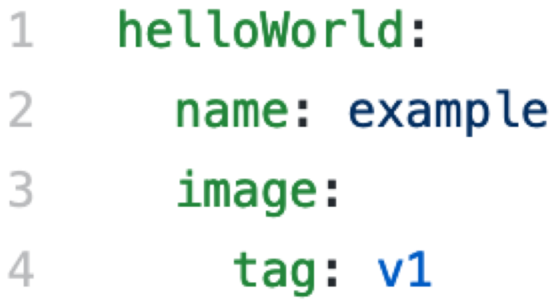


```yaml
helloWorld:
  name: example
  image:
    tag: v1
```

Value File per Environment

values-global.yaml

values.yaml

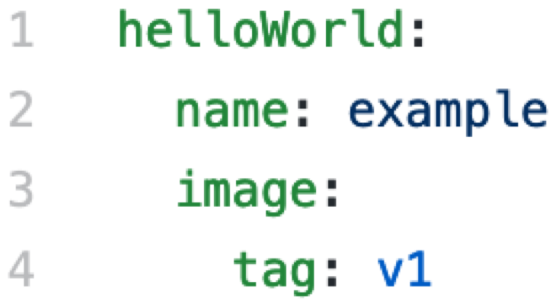


```
helloWorld:
  name: example
  image:
    tag: v1
```

values-staging.yaml

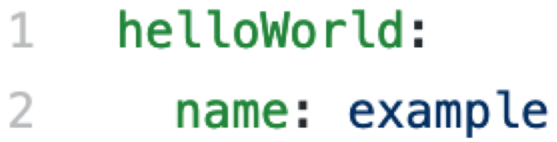
Value File per Environment

values.yaml



```yaml
helloWorld:
  name: example
  image:
    tag: v1
```

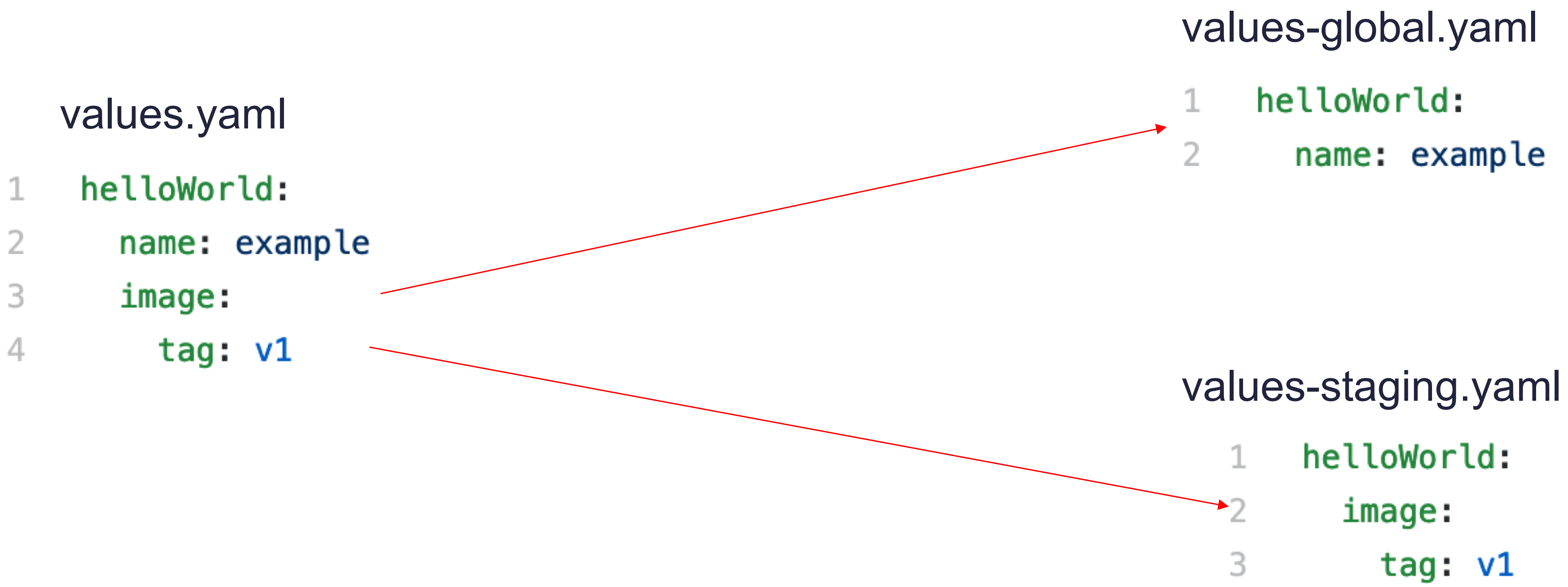
values-global.yaml



```yaml
helloWorld:
  name: example
```

values-staging.yaml

Value File per Environment



values.yaml

```yaml
helloWorld:
  name: example
  image:
    tag: v1
```

values-global.yaml

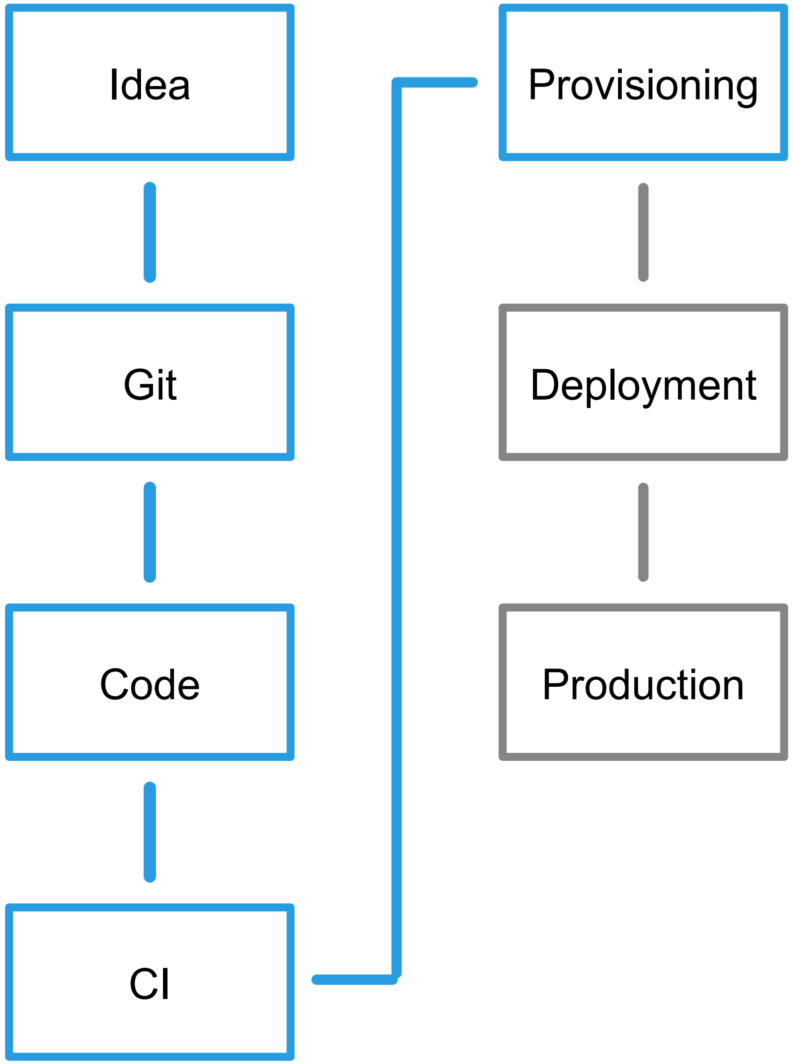
```yaml
helloWorld:
  name: example
```

values-staging.yaml

```yaml
helloWorld:
  image:
    tag: v1
```

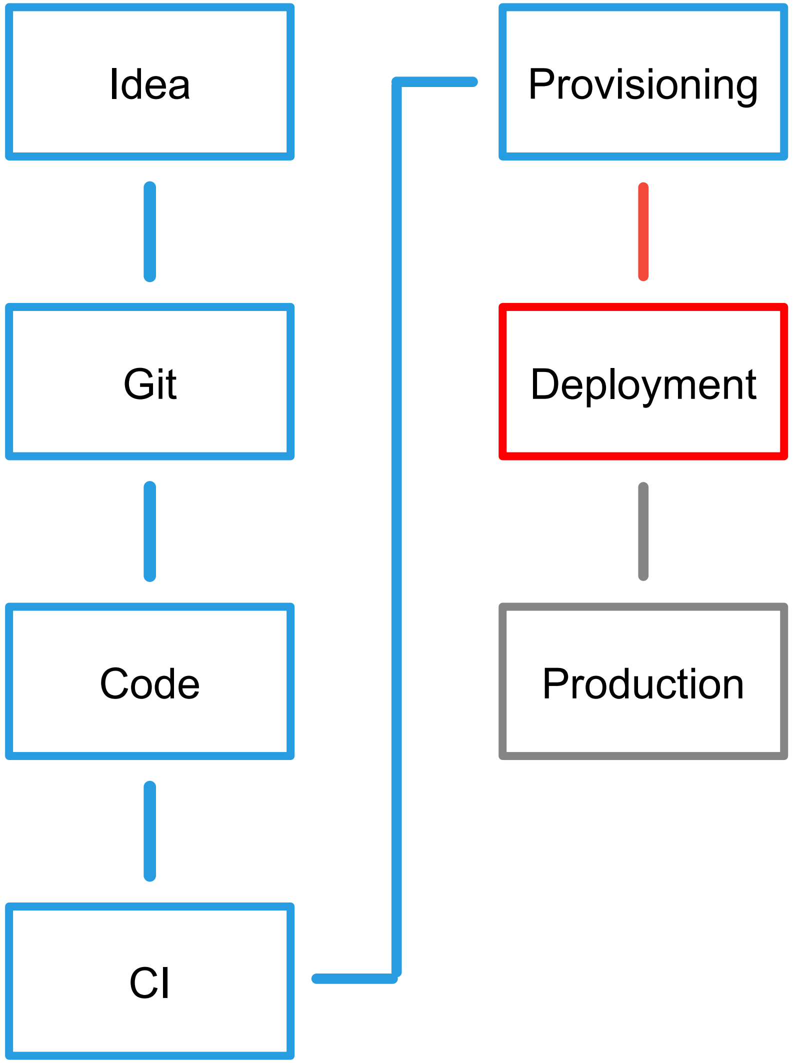
Recap

- Git, Code, CI - single file
- Provisioning
  - Helm chart - add few values
  - Cluster and Namespace



Next Step - Deployment

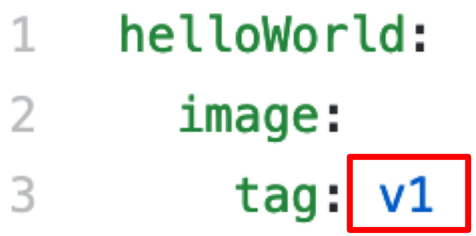
- Git, Code, CI - single file
- Provisioning
  - Helm chart - add few values
  - Cluster and Namespace
- Deployment pipeline



DEVOPSWORLD by CloudBees

Deployment

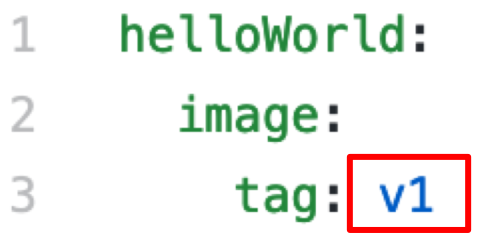
Change the tag value in the right value file



```
helloWorld:
  image:
    tag: v1
```

Deployment

Change the tag value in the right value file



```yaml
helloWorld:
  image:
    tag: v1
```

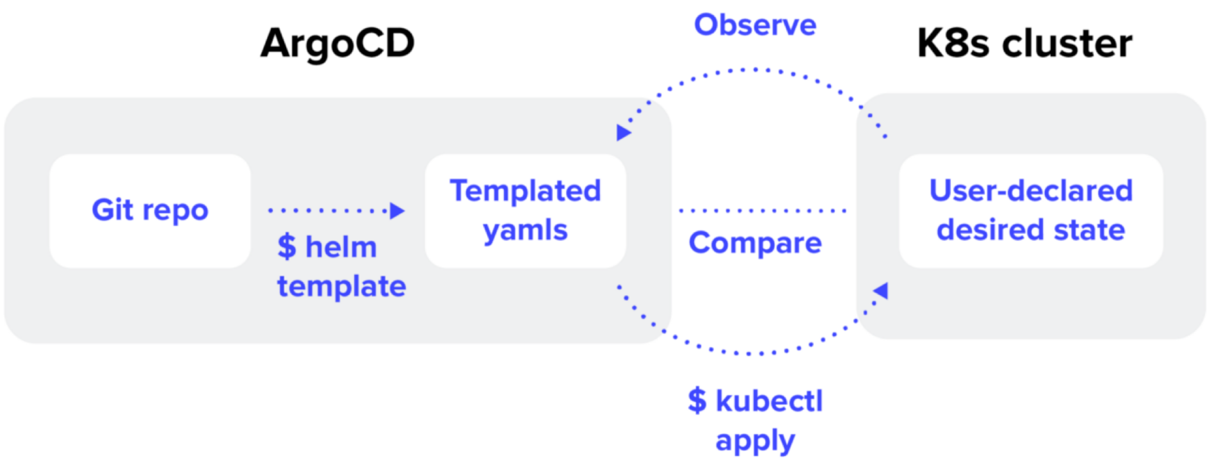
“How to apply the change?”

Deployment - ArgoCD

- GitOps continuous delivery tool

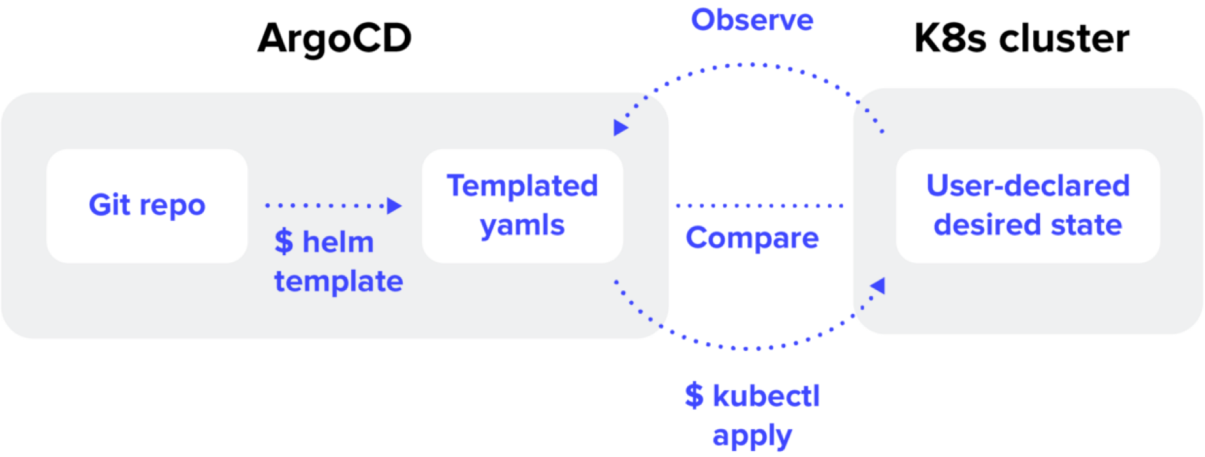
Deployment - ArgoCD

- GitOps continuous delivery tool
- Controller / git watcher

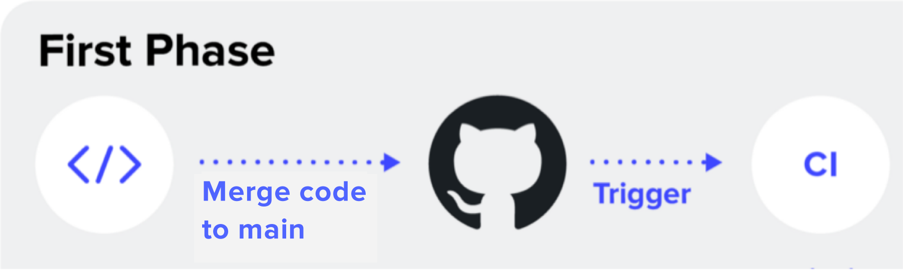


Deployment - ArgoCD

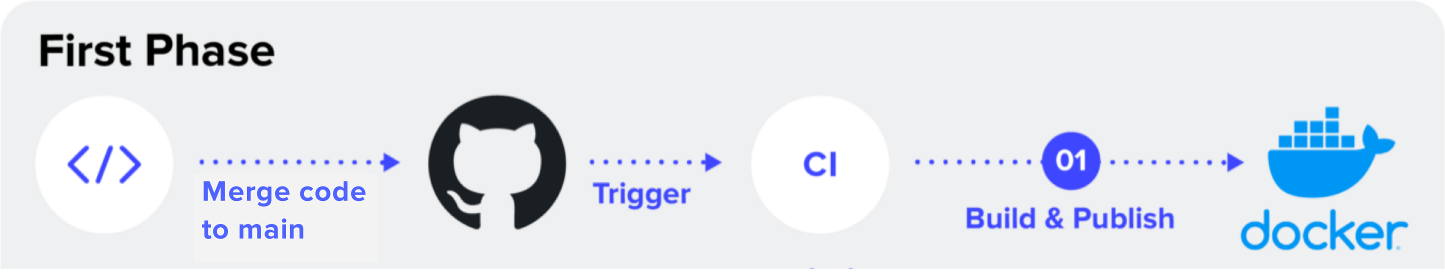
- GitOps continuous delivery tool
- Controller / git watcher
- Supports multiple templating tools



The Deployment Process

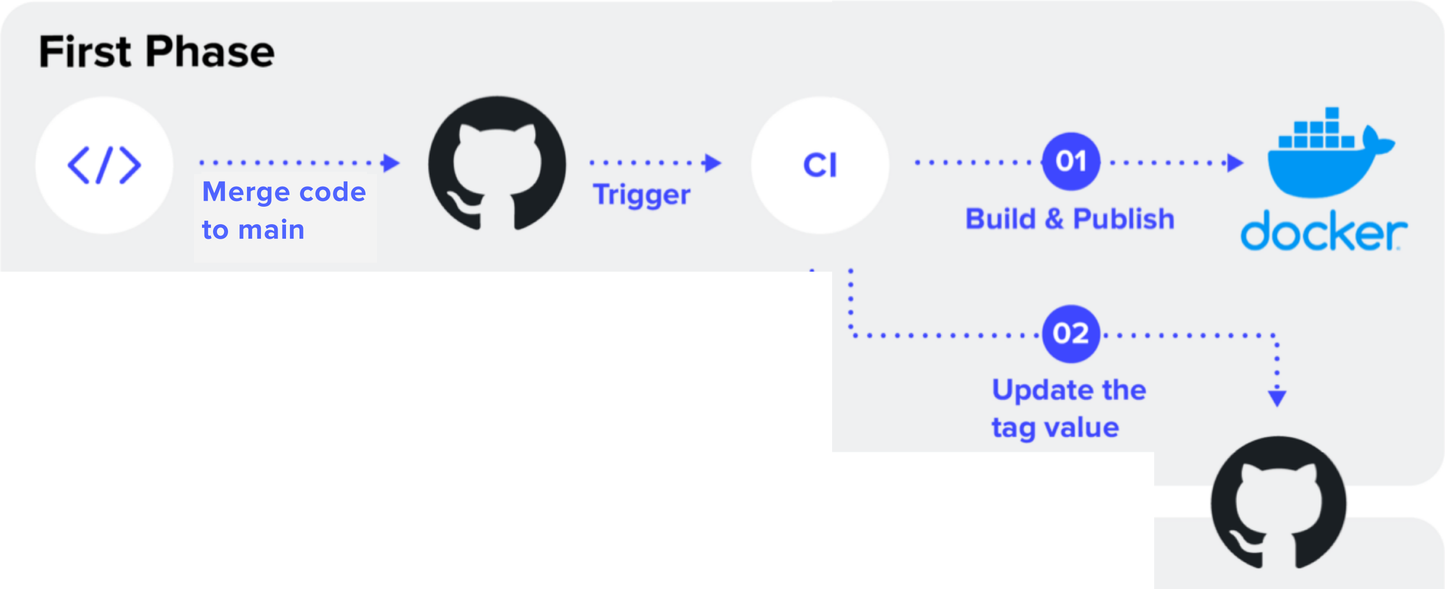


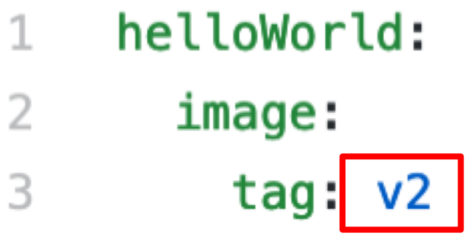
The Deployment Process



helloWorld:v2

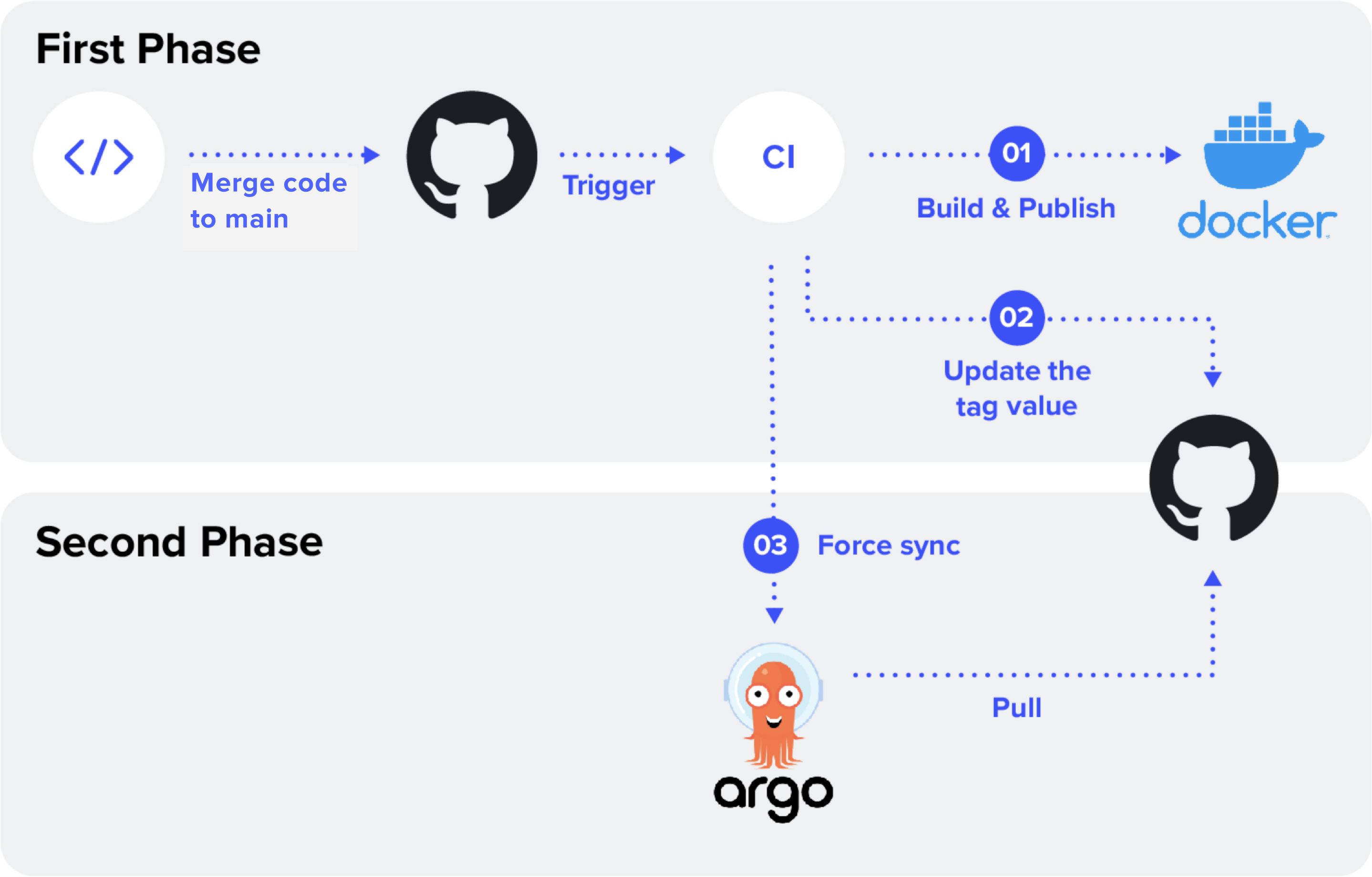
The Deployment Process



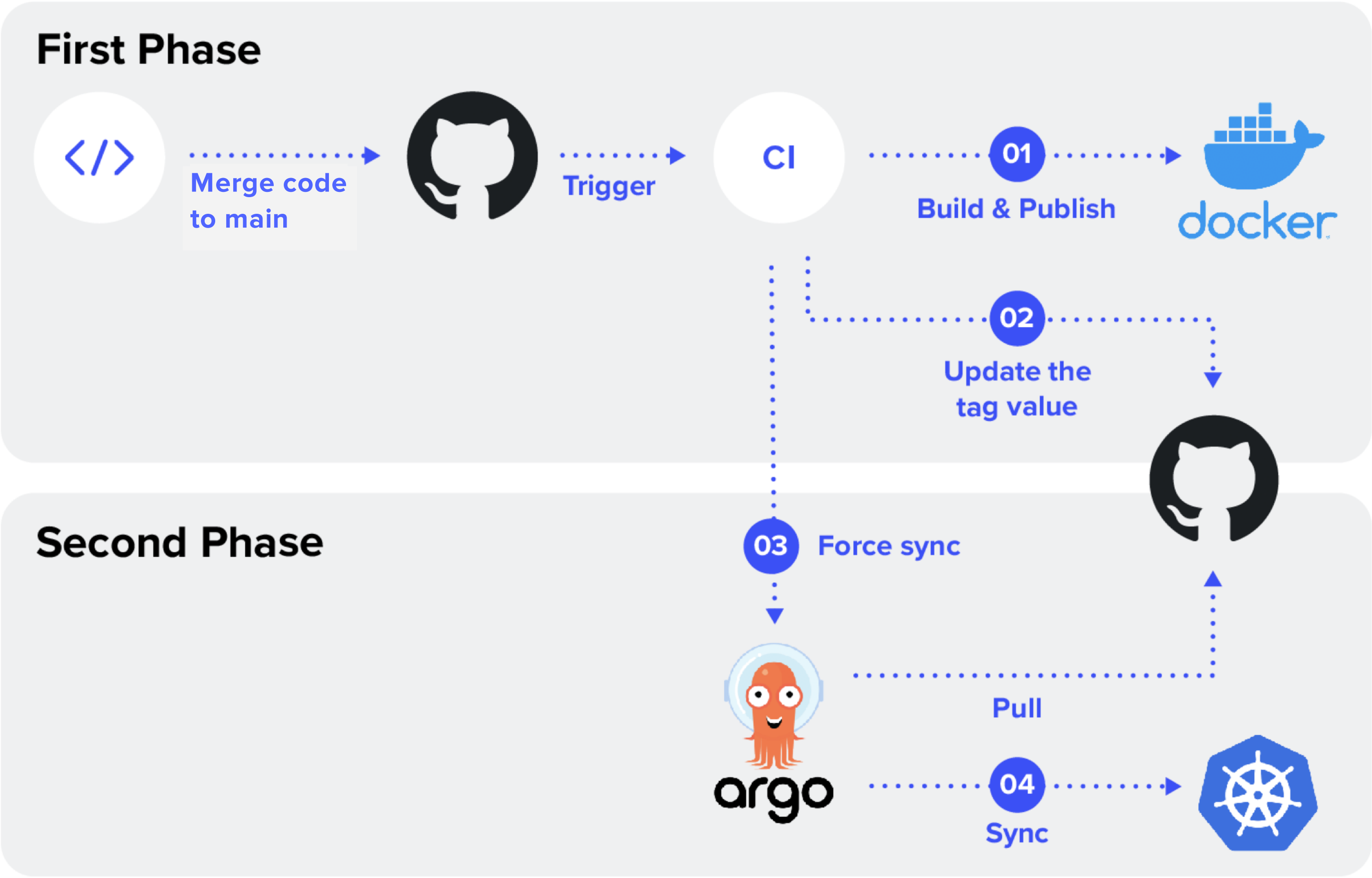


```
helloWorld:
  image:
    tag: v2
```

The Deployment Process

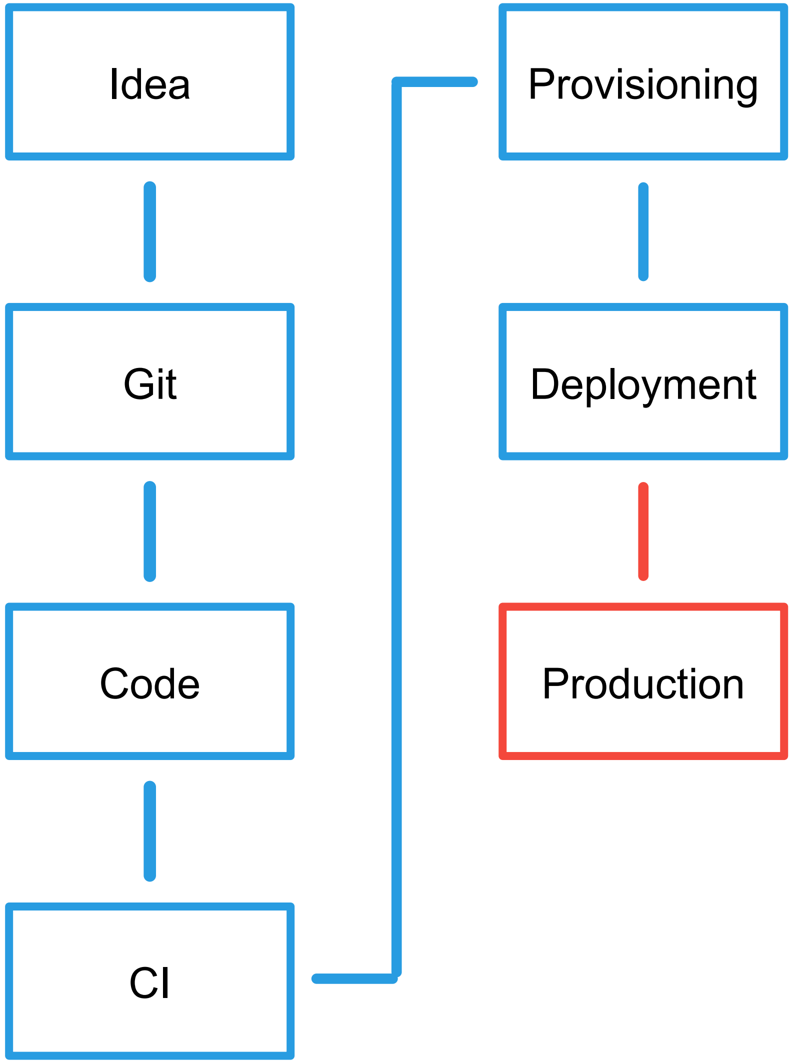


The Deployment Process



Next Step - Production

- Git, Code, CI - single file
- Provisioning
  - Helm chart - add few values
  - Cluster and Namespace
- Code deployment pipeline
  - ArgoCD
- Does my application works?



Production UI

- Helm hides the complicated parts of Kubernetes

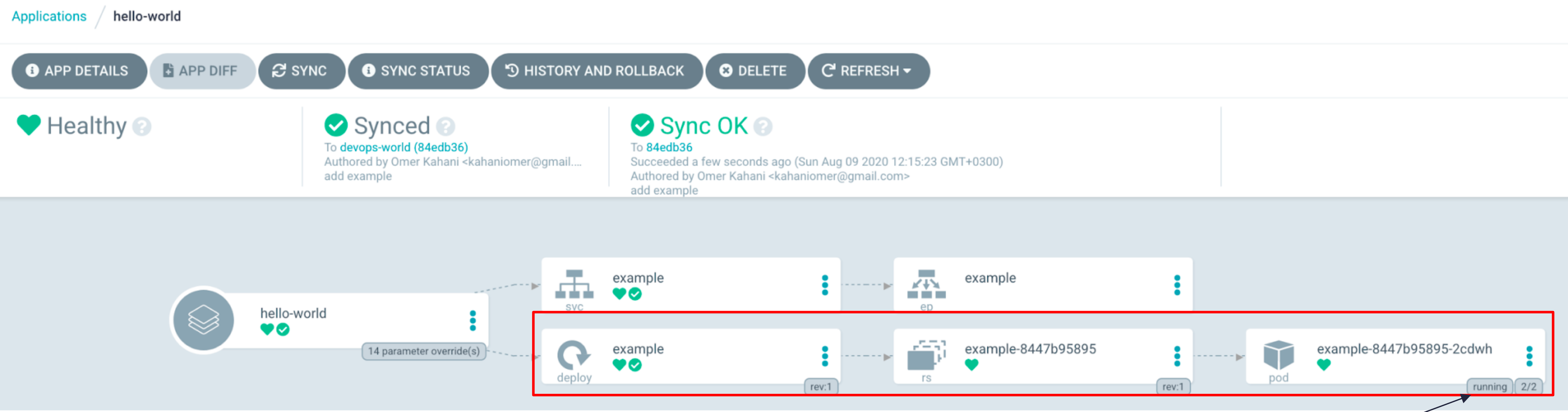
Production UI

- Helm hides the complicated parts of Kubernetes
- Good for the learning curve

Production UI

- Helm hides the complicated parts of Kubernetes
- Good for the learning curve
- Bad in production

ArgoCD - UI



Applications / hello-world

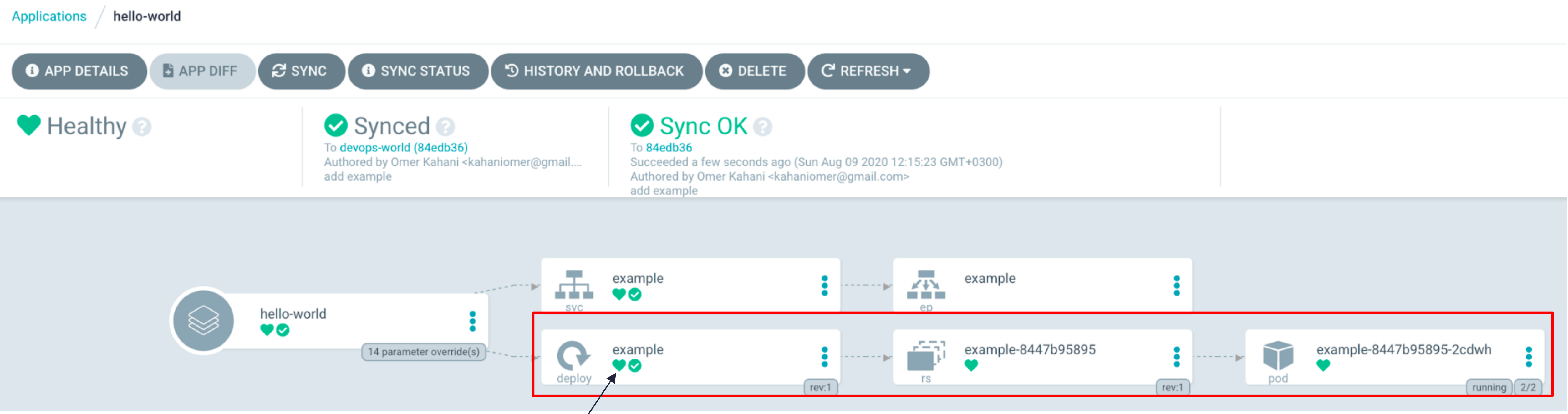
Healthy

Synced
To devops-world (84edb36)
Authored by Omer Kahani <kahaniomer@gmail....
add example

Sync OK
To 84edb36
Succeeded a few seconds ago (Sun Aug 09 2020 12:15:23 GMT+0300)
Authored by Omer Kahani <kahaniomer@gmail.com>
add example

Pod status
(running, error, init…)

ArgoCD - UI



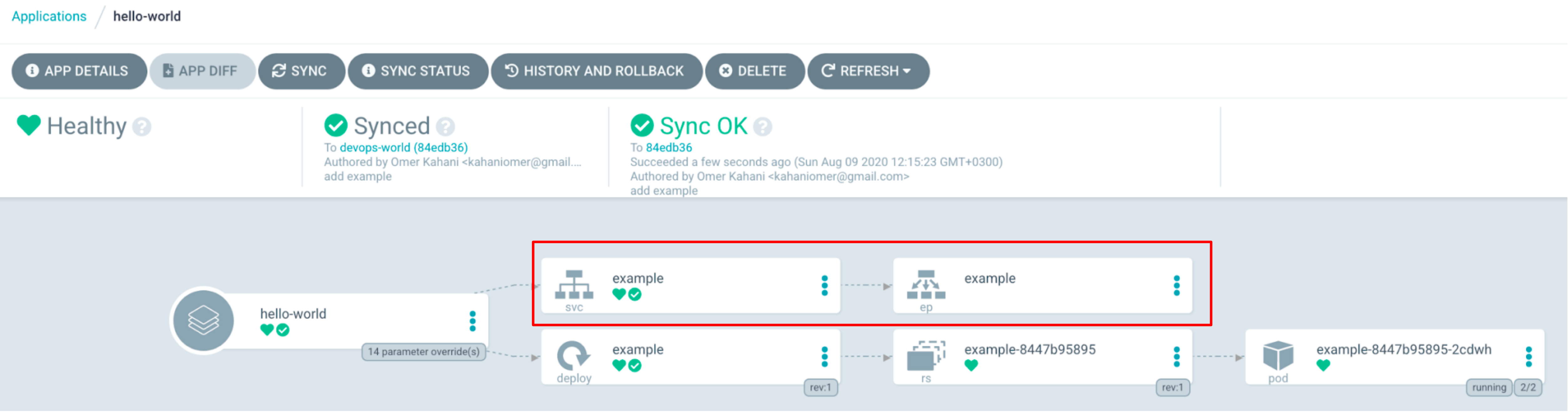
Healthy

Synced
To devops-world (84edb36)
Authored by Omer Kahani <kahaniomer@gmail....
add example

Sync OK
To 84edb36
Succeeded a few seconds ago (Sun Aug 09 2020 12:15:23 GMT+0300)
Authored by Omer Kahani <kahaniomer@gmail.com>
add example

Deployment Health
(green / red)

ArgoCD - UI



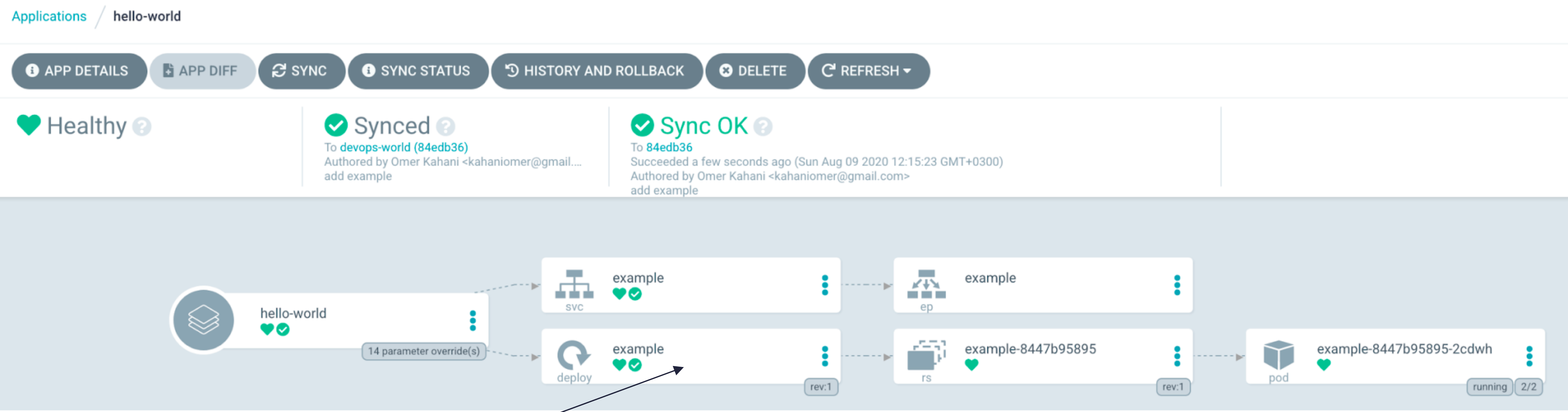
Applications / hello-world

Healthy

Synced
To devops-world (84edb36)
Authored by Omer Kahani <kahaniomer@gmail....
add example

Sync OK
To 84edb36
Succeeded a few seconds ago (Sun Aug 09 2020 12:15:23 GMT+0300)
Authored by Omer Kahani <kahaniomer@gmail.com>
add example

ArgoCD - UI



Applications / hello-world

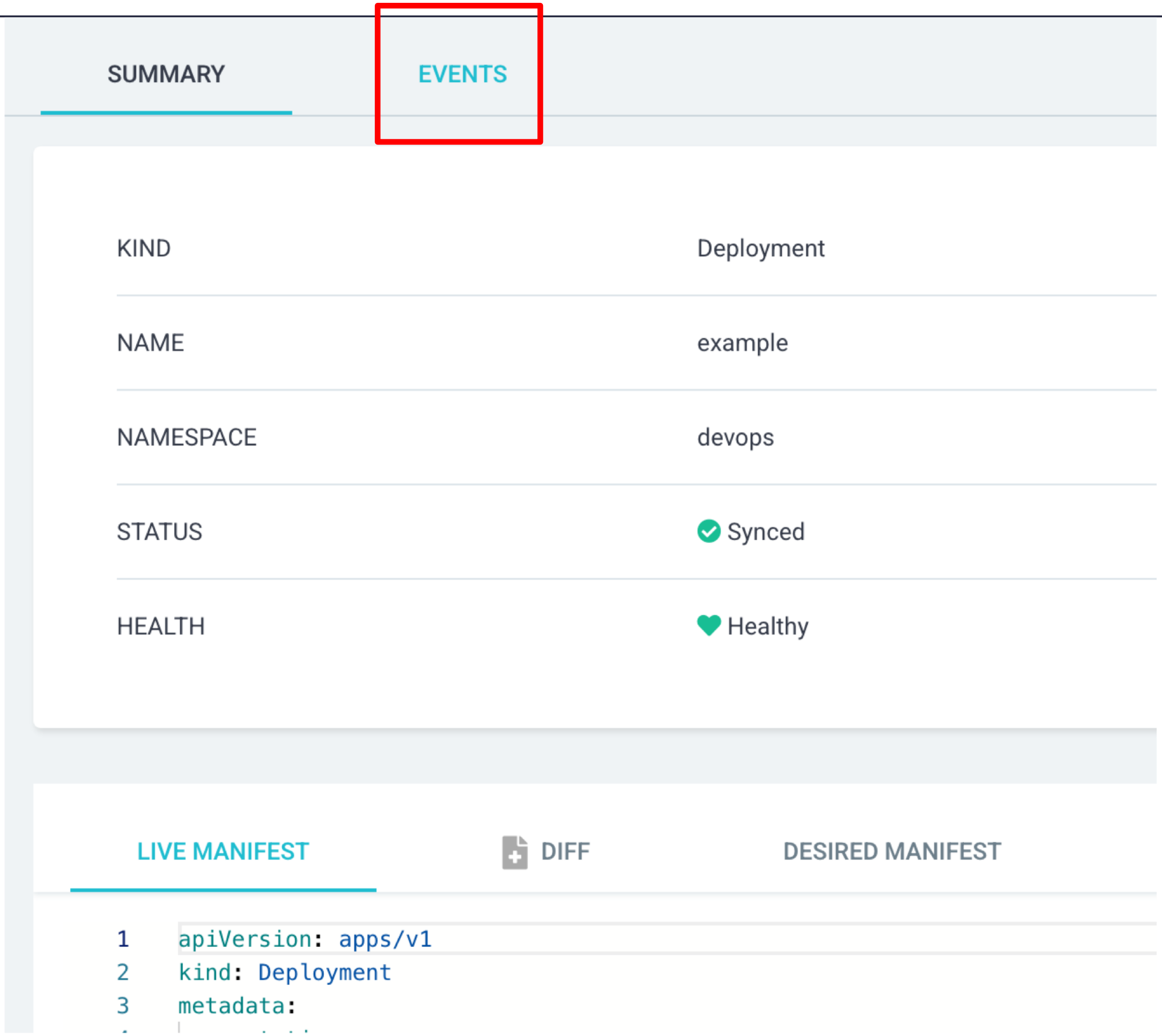
Healthy

Synced
To devops-world (84edb36)
Authored by Omer Kahani <kahaniomer@gmail....
add example

Sync OK
To 84edb36
Succeeded a few seconds ago (Sun Aug 09 2020 12:15:23 GMT+0300)
Authored by Omer Kahani <kahaniomer@gmail.com>
add example

Click for more...

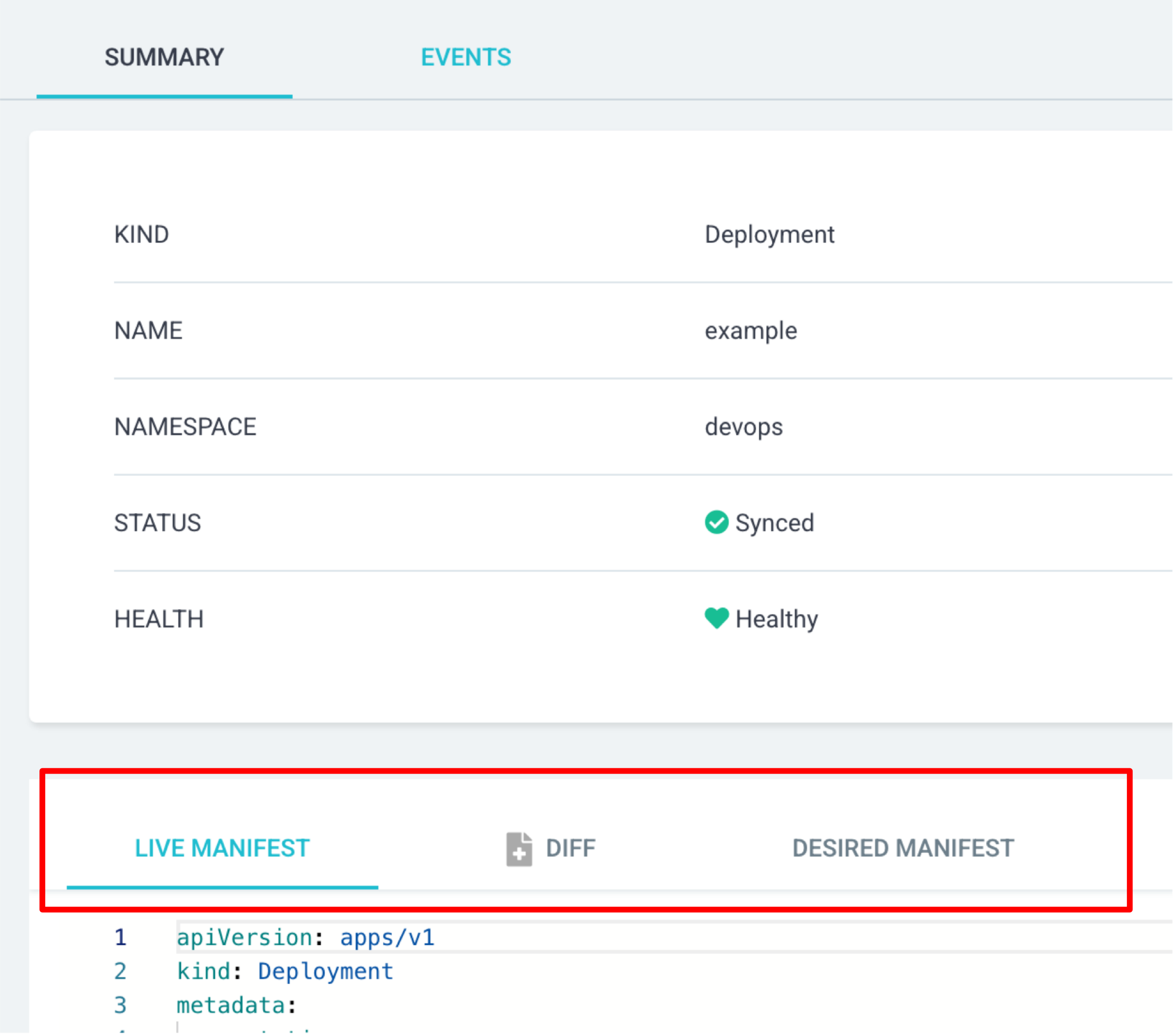
ArgoCD - UI



KIND	Deployment
NAME	example
NAMESPACE	devops
STATUS	✓ Synced
HEALTH	Healthy

```
apiVersion: apps/v1
kind: Deployment
metadata:
```

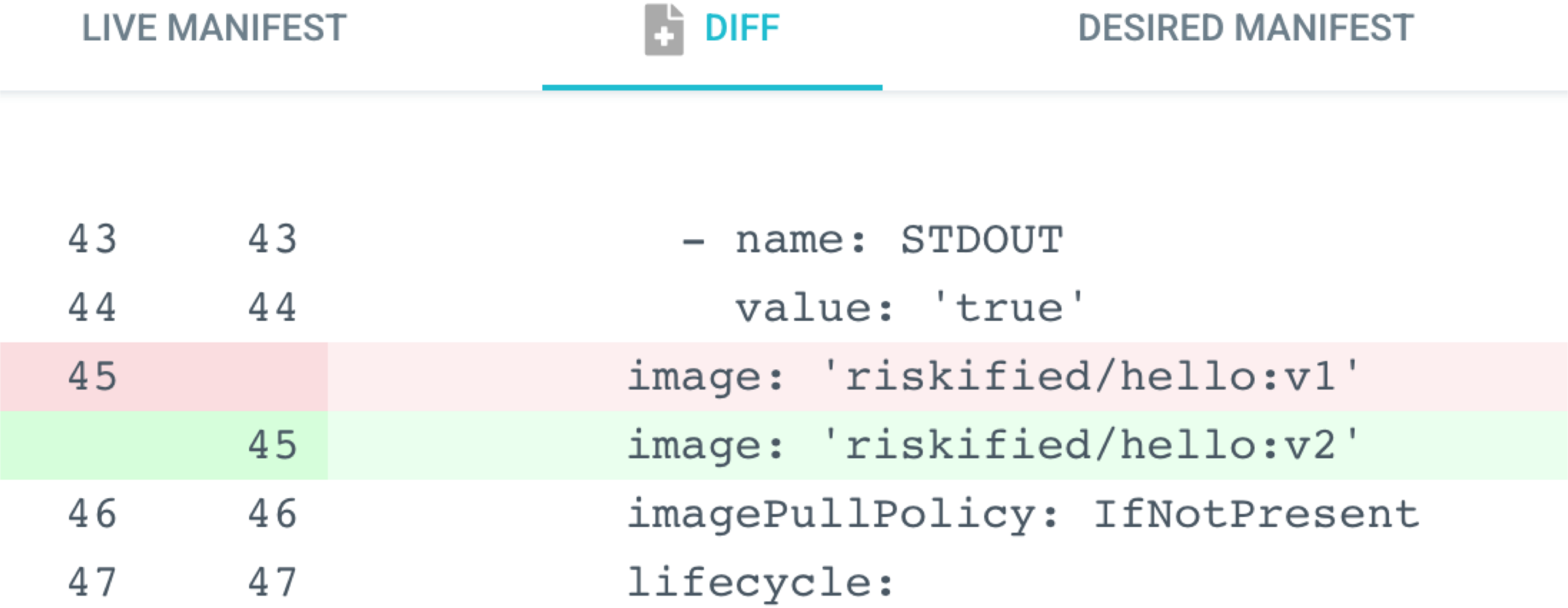
ArgoCD - UI



KIND	Deployment
NAME	example
NAMESPACE	devops
STATUS	✓ Synced
HEALTH	♥ Healthy

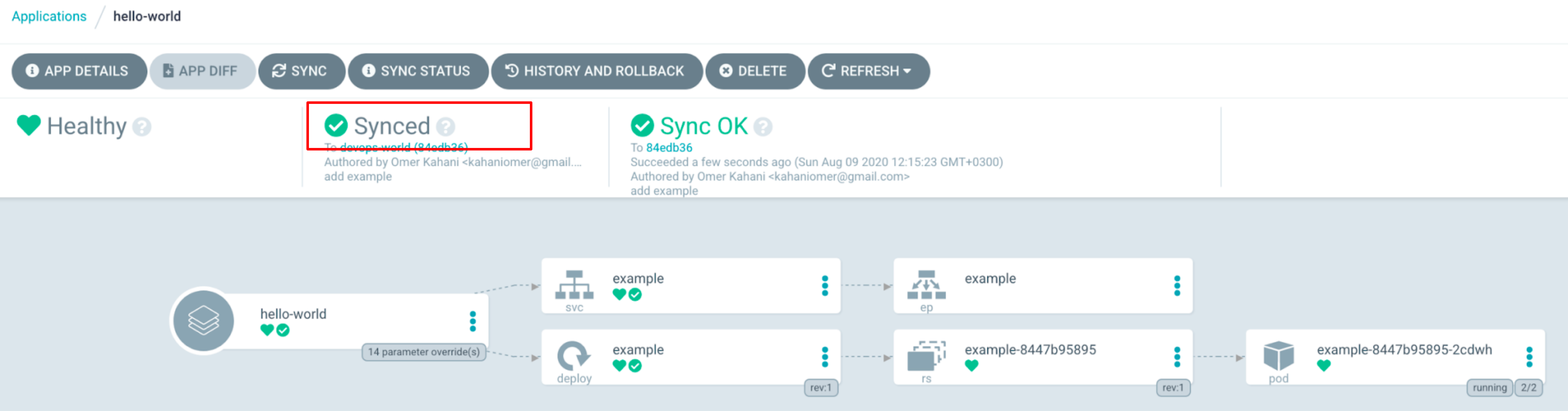
```
apiVersion: apps/v1
kind: Deployment
metadata:
```

ArgoCD - UI



```
43      43              - name: STDOUT
44      44                value: 'true'
45                  image: 'riskified/hello:v1'
        45          image: 'riskified/hello:v2'
46      46          imagePullPolicy: IfNotPresent
47      47          lifecycle:
```

ArgoCD - UI

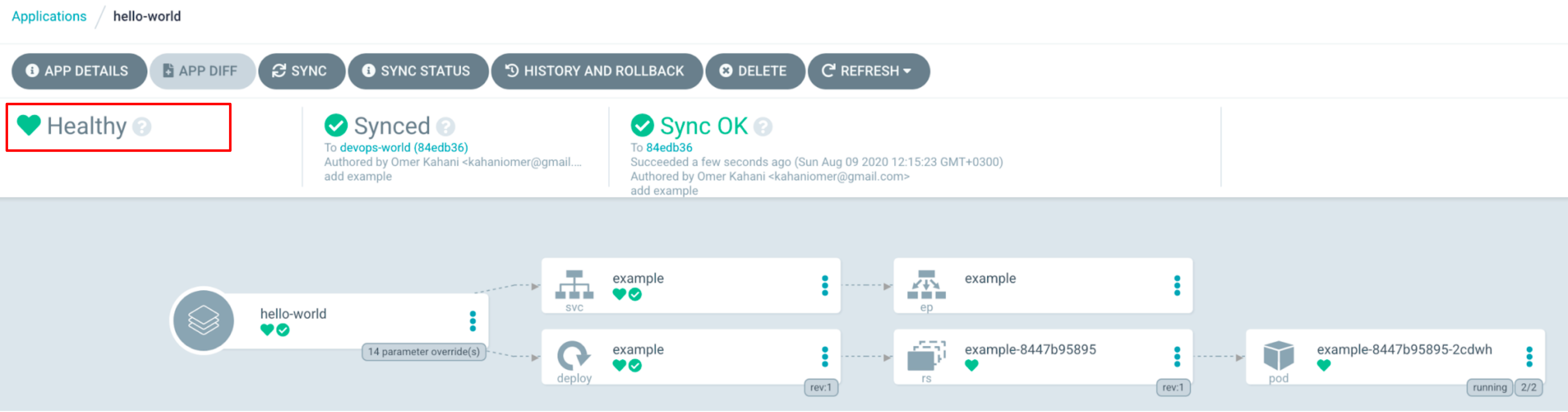


Healthy

Synced
To devops-world (84edb36)
Authored by Omer Kahani <kahaniomer@gmail....
add example

Sync OK
To 84edb36
Succeeded a few seconds ago (Sun Aug 09 2020 12:15:23 GMT+0300)
Authored by Omer Kahani <kahaniomer@gmail.com>
add example

ArgoCD - UI



Applications / hello-world

Healthy

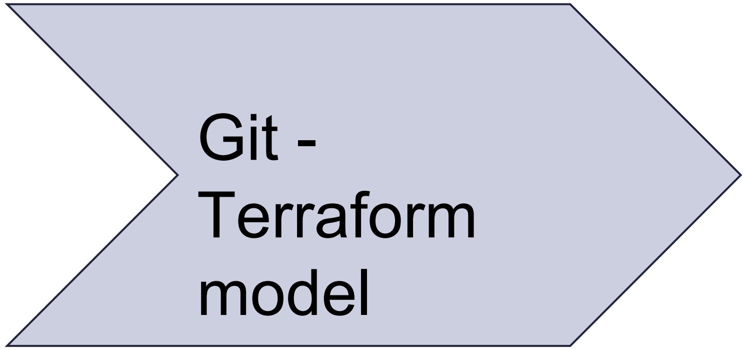
Synced
To devops-world (84edb36)
Authored by Omer Kahani <kahaniomer@gmail....
add example

Sync OK
To 84edb36
Succeeded a few seconds ago (Sun Aug 09 2020 12:15:23 GMT+0300)
Authored by Omer Kahani <kahaniomer@gmail.com>
add example

Summary

Summary

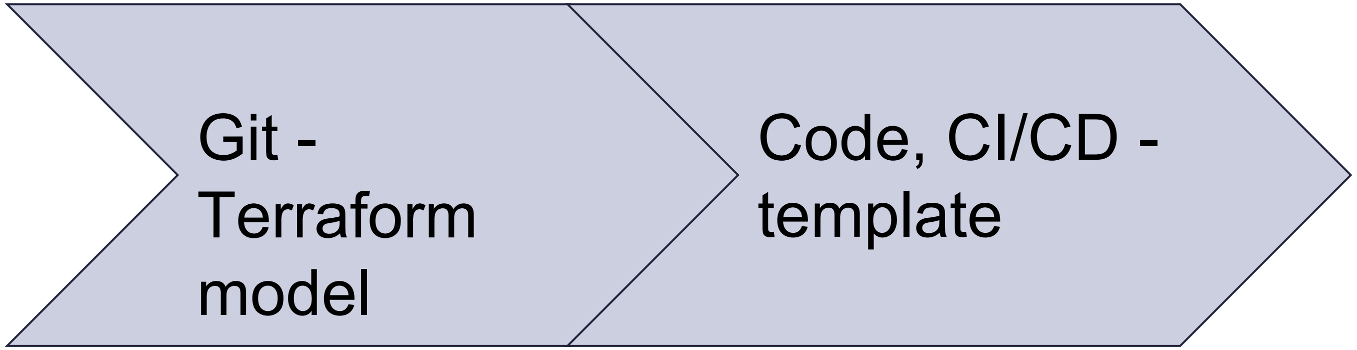
- 3 steps to a working service



Git - Terraform model

Summary

- 3 steps to a working service

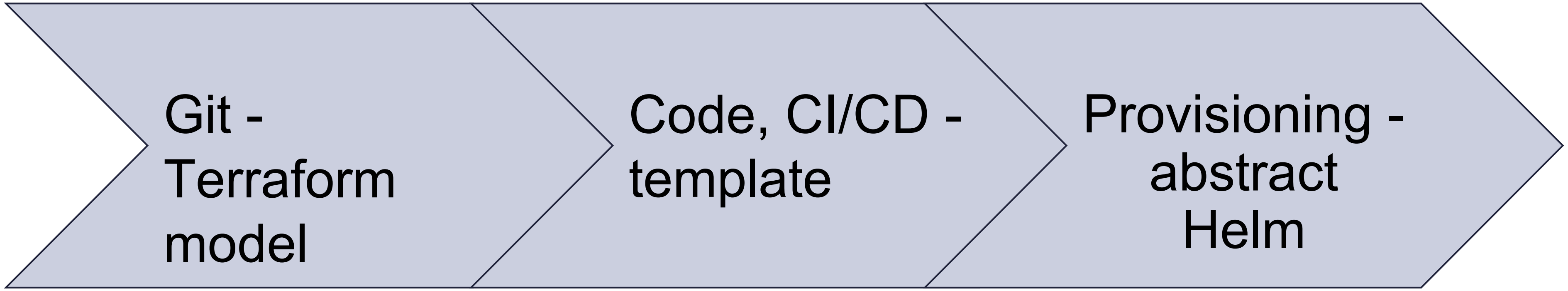


Git - Terraform model

Code, CI/CD - template

Summary

- 3 steps to a working service



Git - Terraform model

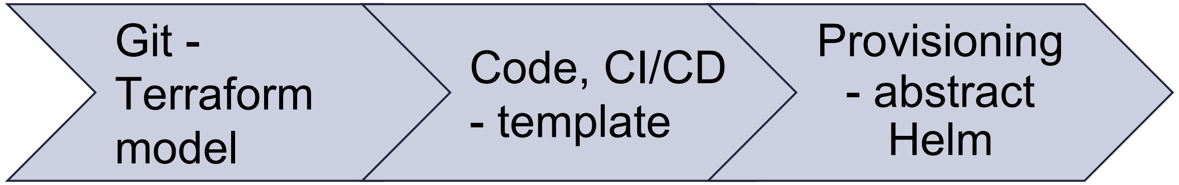
Code, CI/CD - template

Provisioning - abstract Helm

DEVOPSWORLD by CloudBees

Summary

- 3 steps to a working service
- Abstract to reduce learning curve



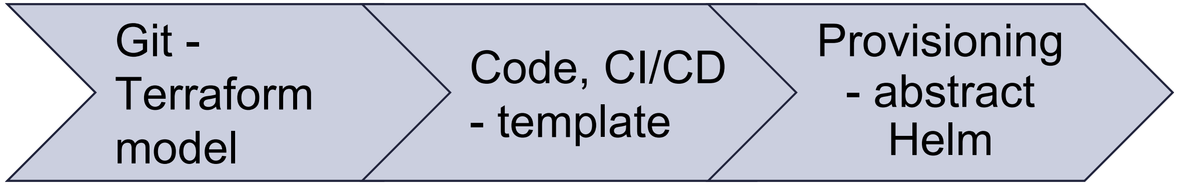
Git - Terraform model

Code, CI/CD - template

Provisioning - abstract Helm

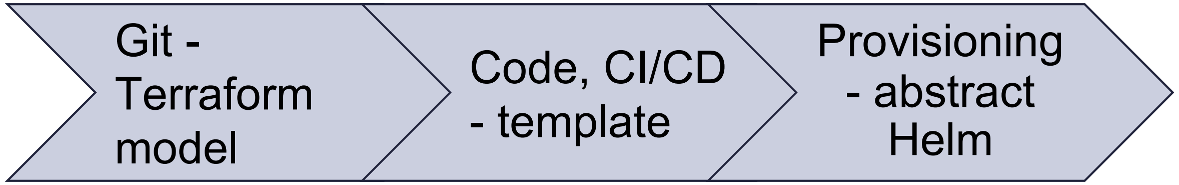
Summary

- 3 steps to a working service
- Abstract to reduce learning curve
- Boundaries to ensure reliability



Summary

- 3 steps to a working service
- Abstract to reduce learning curve
- Boundaries to ensure reliability
- GitOps



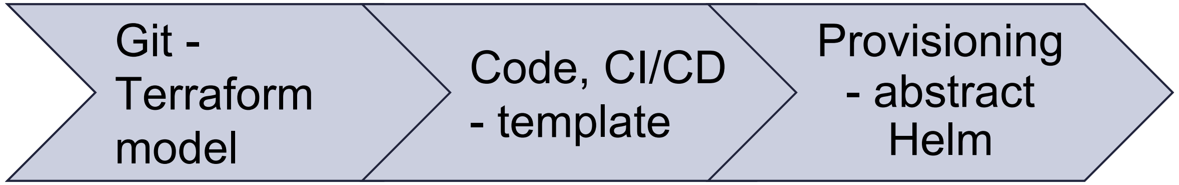
Git - Terraform model

Code, CI/CD - template

Provisioning - abstract Helm

Summary

- 3 steps to a working service
- Abstract to reduce learning curve
- Boundaries to ensure reliability
- GitOps
- Safe, explainable process
- Developer independence



Git - Terraform model

Code, CI/CD - template

Provisioning - abstract Helm

Thank You

- Find out more:

  https://medium.com/@kahaniomer

  https://medium.com/riskified-technology

- Reach out:

  Twitter: @OmerKahani

  Slack: Kubernetes, CNCF, Argo

DEVOPSWORLD by CloudBees