# The pains of GitOps 1.0

And the vision for GitOps 2.0

**KOSTIS KAPELONIS**

# Agenda

- What is GitOps
- Scope and lack of standards
- Promotion to different environments
- Observability and visibility
- Mixing CI and CD
- GitOps at scale
- Proposed solutions
- The future of GitOps

# What is GitOps

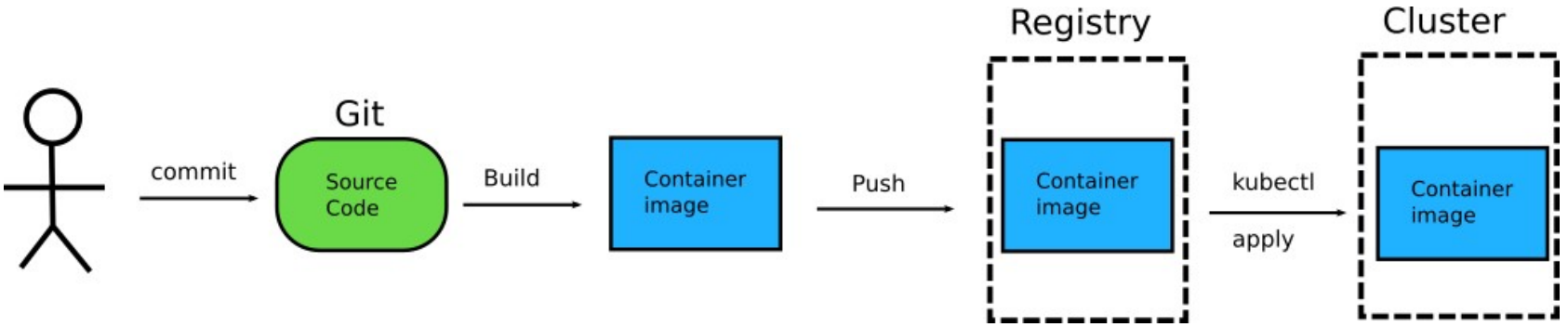
## Everything in Git
Git is single source of truth

## Declarative deployments
Deploy by matching Git state with cluster state

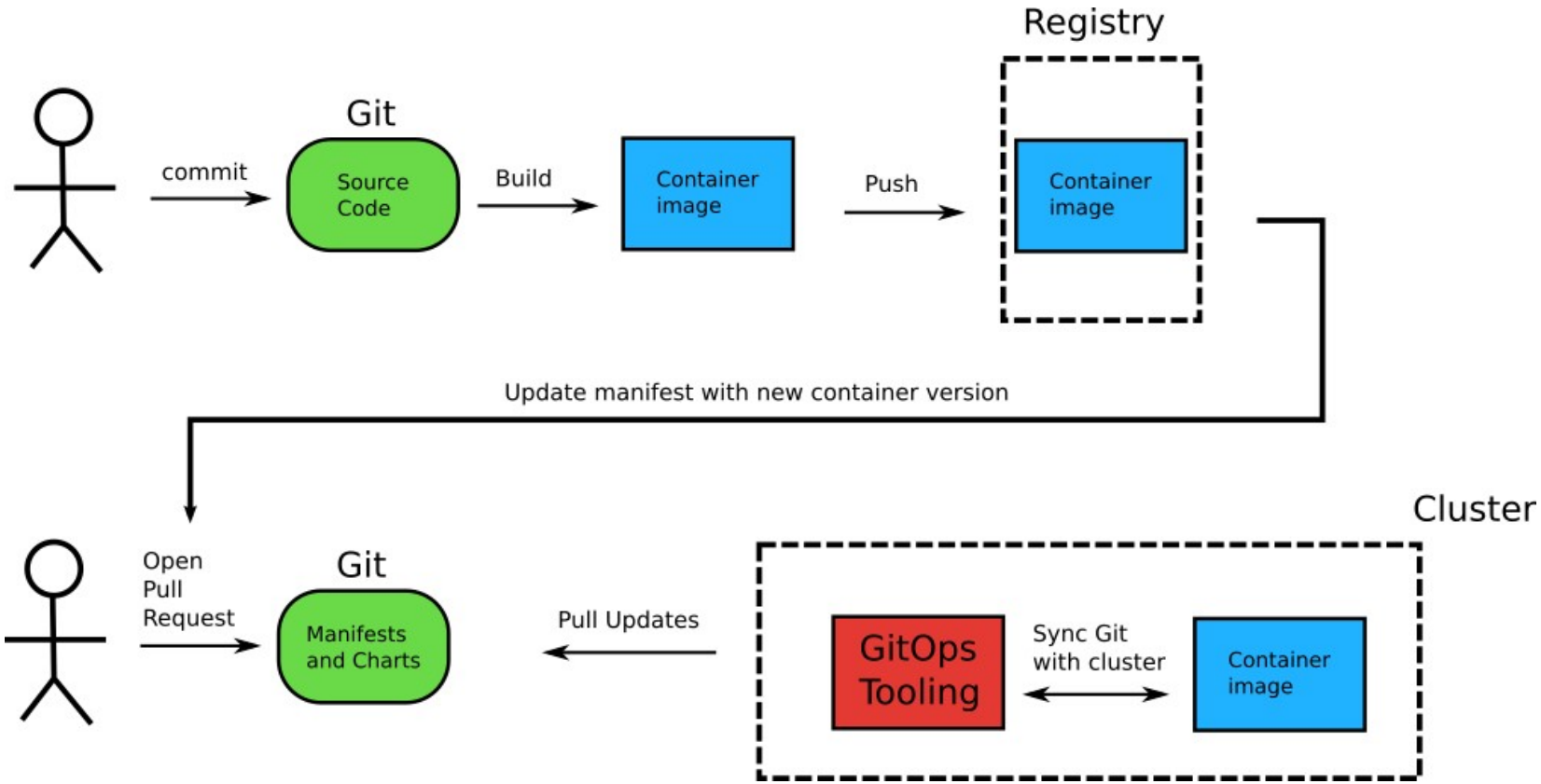
## No adhoc actions
Operations by Pull Request

https://www.weave.works/blog/gitops-operations-by-pull-request

## Traditional deployment



## GitOps deployment



# GitOps advantages

## Git state = cluster state

Git is single source of truth

## Git history = deployment history

Deploy by matching Git state with cluster state

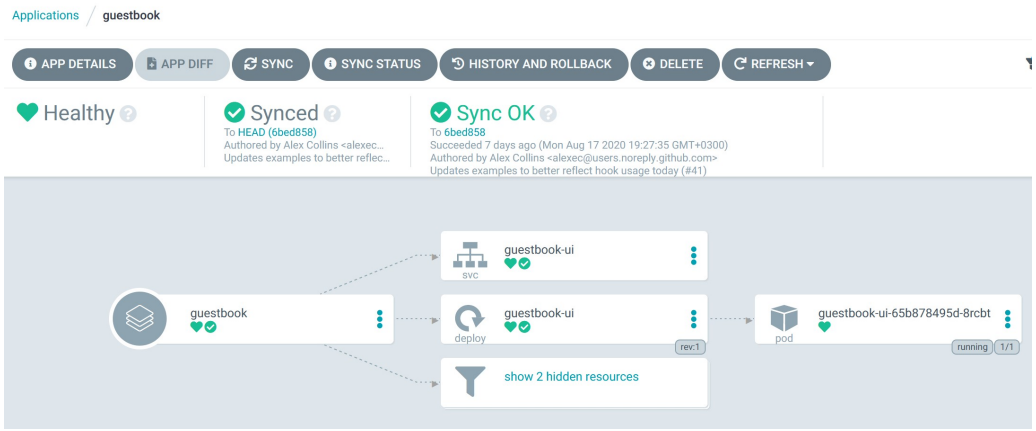
## Deploy/rollback with Git tools

Operations by Pull Request

## GitOps tools

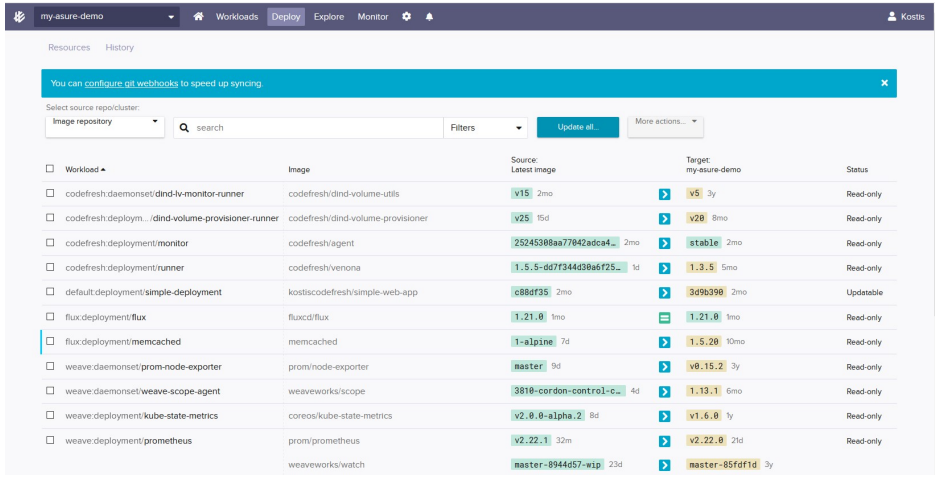
# ArgoCD

https://argoproj.github.io/argo-cd/




# Flux V2

https://toolkit.fluxcd.io/



## Applied GitOps

# Solve configuration drift

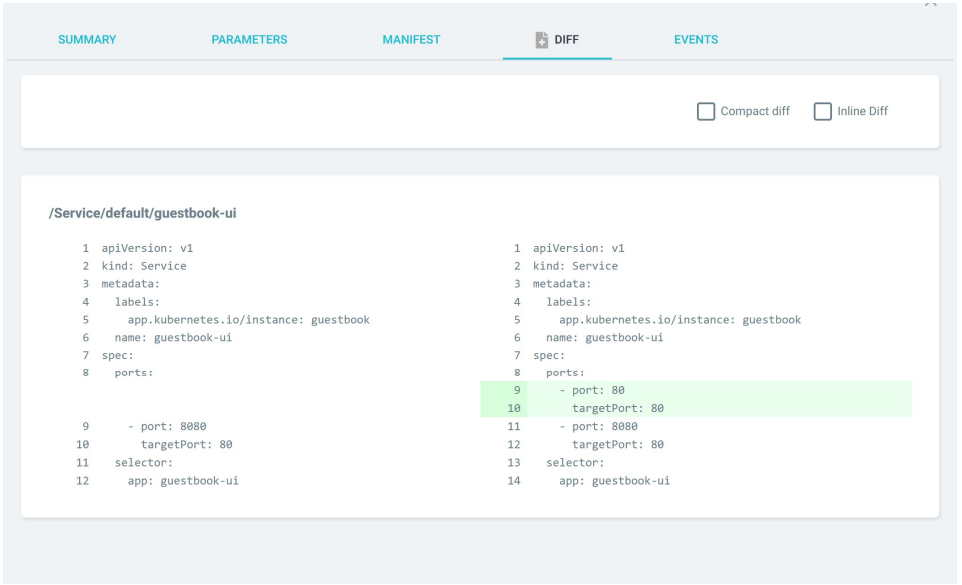
Know what changed in the cluster

# Use Git for everything

…and not just for source code

# Deployment is pull request

Merging a pull request starts the deployment



# GitOps challenges

What lies beneath the surface

# GitOps gaps

## Scope

GitOps covers only deployments

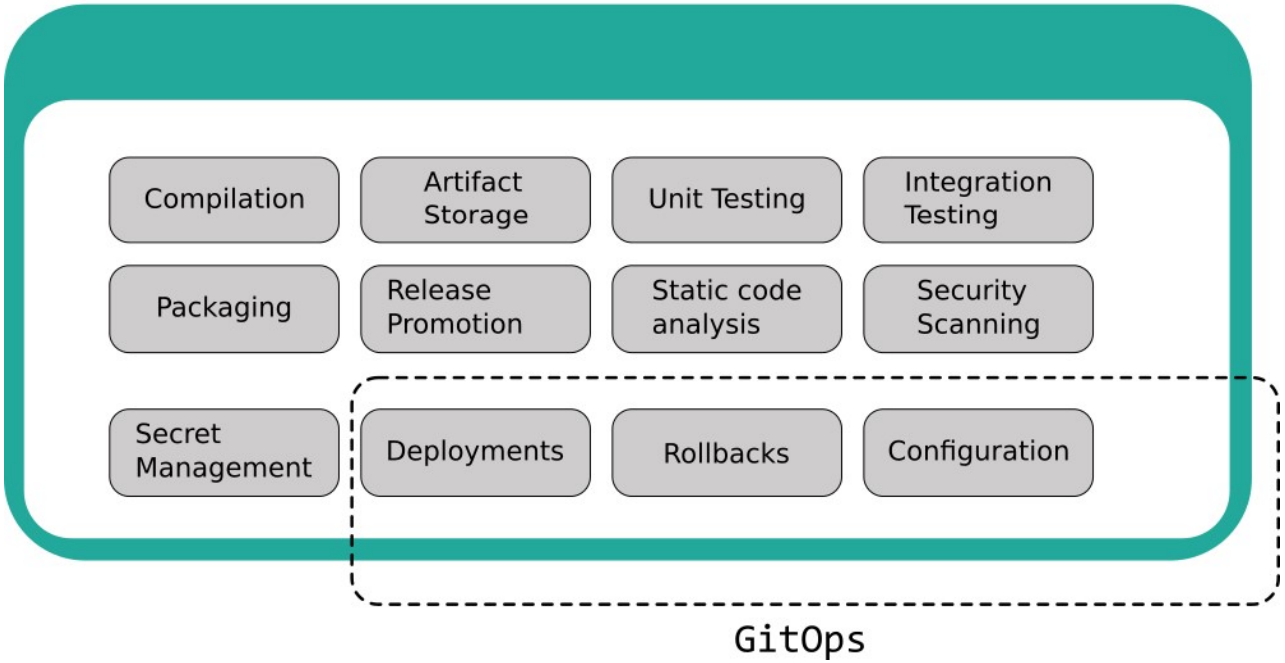
## How to do Rollbacks

No best practice

## How to handle secrets

No best practice

## GitOps only covers deployments



No documented best practices

## How to handle secrets

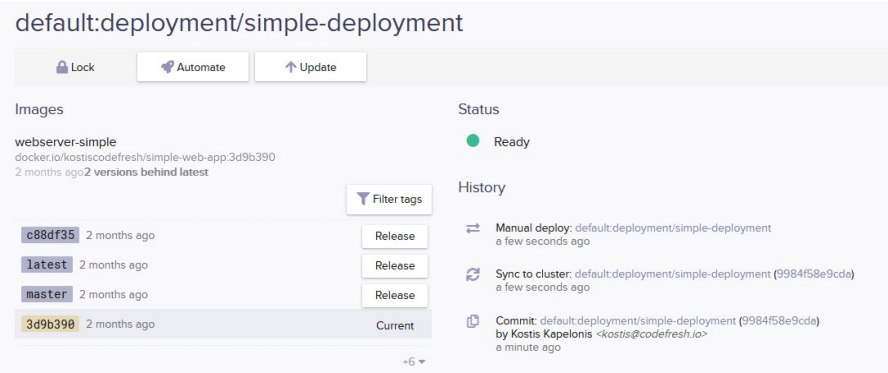
- Store in Git?
- Fetch from Hashicorp Vault?
- Use Bitnami Sealed secrets
- Use Mozilla Sops?

## How to do rollbacks?

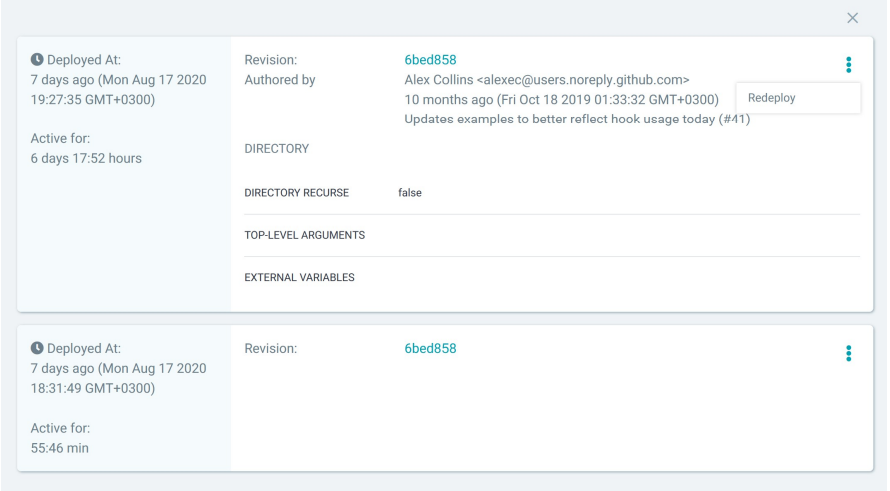
- Manually git revert?
- Manually git reset?
- Automatically go to previous hash?
- Automaticaly go to previous container image?

## Rolling back

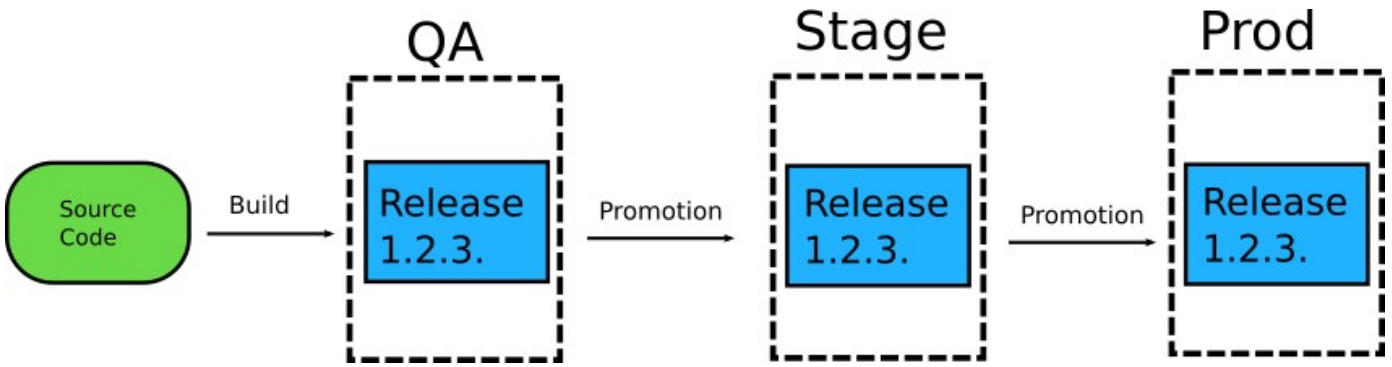
# Flux rollback




# ArgoCD rollback



# Environment promotion



# Environment promotion

**GassiestFunInTheWest** 1 point · 1 year ago

Do you run multiple environments? How do you orchestrate releases through and across environments?

Give Award Share Report Save

Tacticus 11 points · 1 year ago

How do you handle promotion through different environments or rollback in the event of a bad service deploy?

Give Award Share Report Save

coderanger 3 points · 1 year ago

Branches, different envs are configured to watch different branches of the gitops repos. Promotion is opening a PR to merge from one to the next. Rollback is undoing that merge.

Give Award Share Report Save

Tacticus 1 point · 1 year ago

So you don't have any automated progression through environments. manual branch merges are it?

Give Award Share Report Save

Tacticus 7 points · 1 year ago

Gitops tools really need to find a better way of doing multi environment promotion.

Give Award Share Report Save

GassiestFunInTheWest 3 points · 1 year ago

Agreed. Without this, GitOps is strictly worse than CD pipelines. From what I can see though, it's not something that the GitOps movement is trying to address.

As such, I'm going to stick with push-based CD pipelines.

Give Award Share Report Save

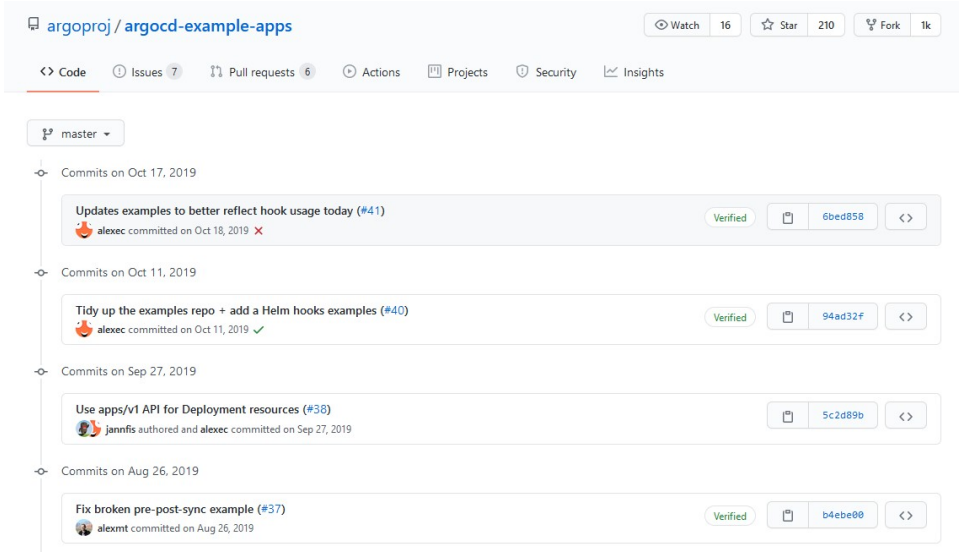
First environment

Commit to App repo
Create container
Push to registry
Commit to manifest repo
Open Pull Request
Merge Pull Request
Sync Cluster

Second environment

?

# Observability




## No info on features

GitOps covers only deployments

## Git history is problematic

No info on unit tests, or testing results

## How to see multiple environments

And what features are deployed

Typical questions in a team

## Cluster questions

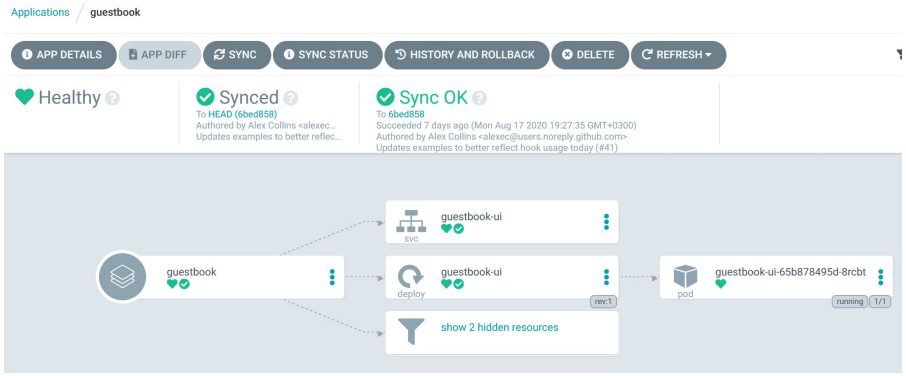
- Is the application healthy?
- How many replicas?
- Is traffic routing correct?
- When was the last deployment?
- Is Git state same as cluster state?

## Application questions

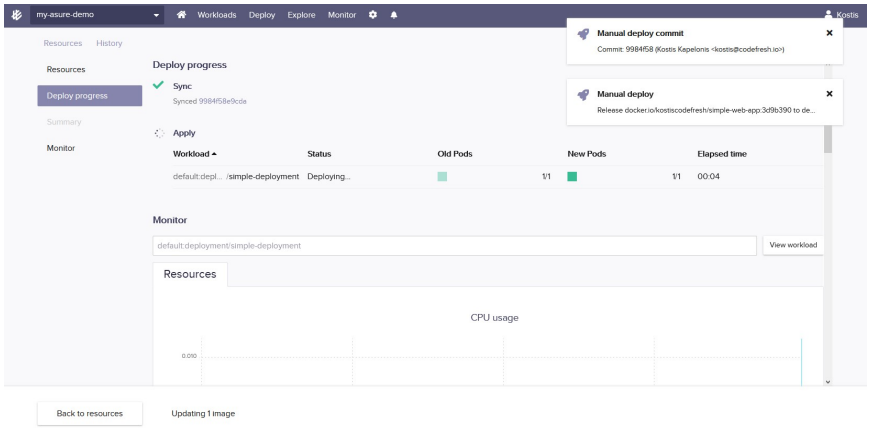
- Does production have feature X?
- Is bug Y present in staging?
- What features were deployed yesterday?
- What was the last feature deployed 2 hours ago?
- How long did feature Z stay in the staging environment before going to production?

## Answering cluster questions

# ArgoCD



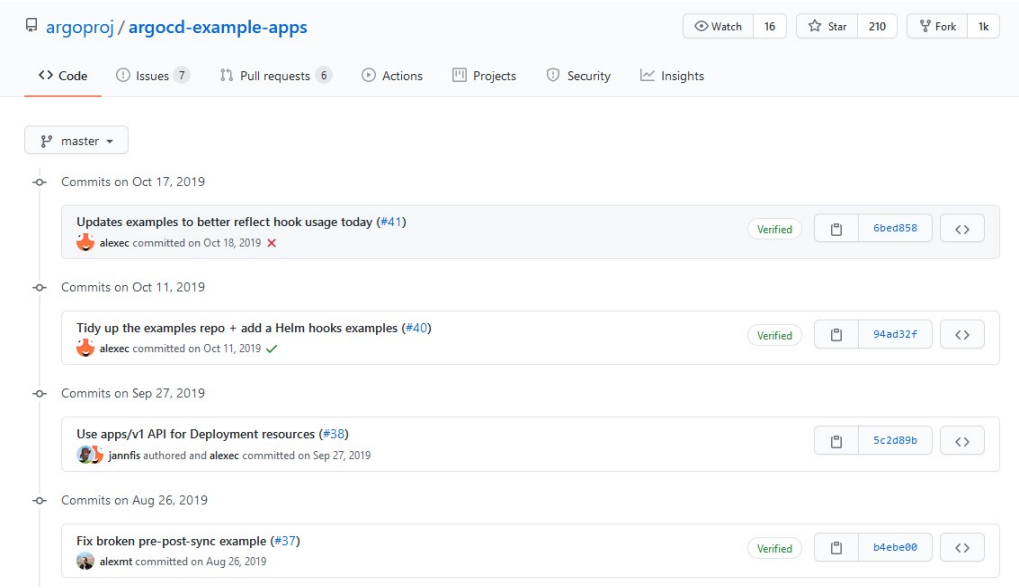

# Flux



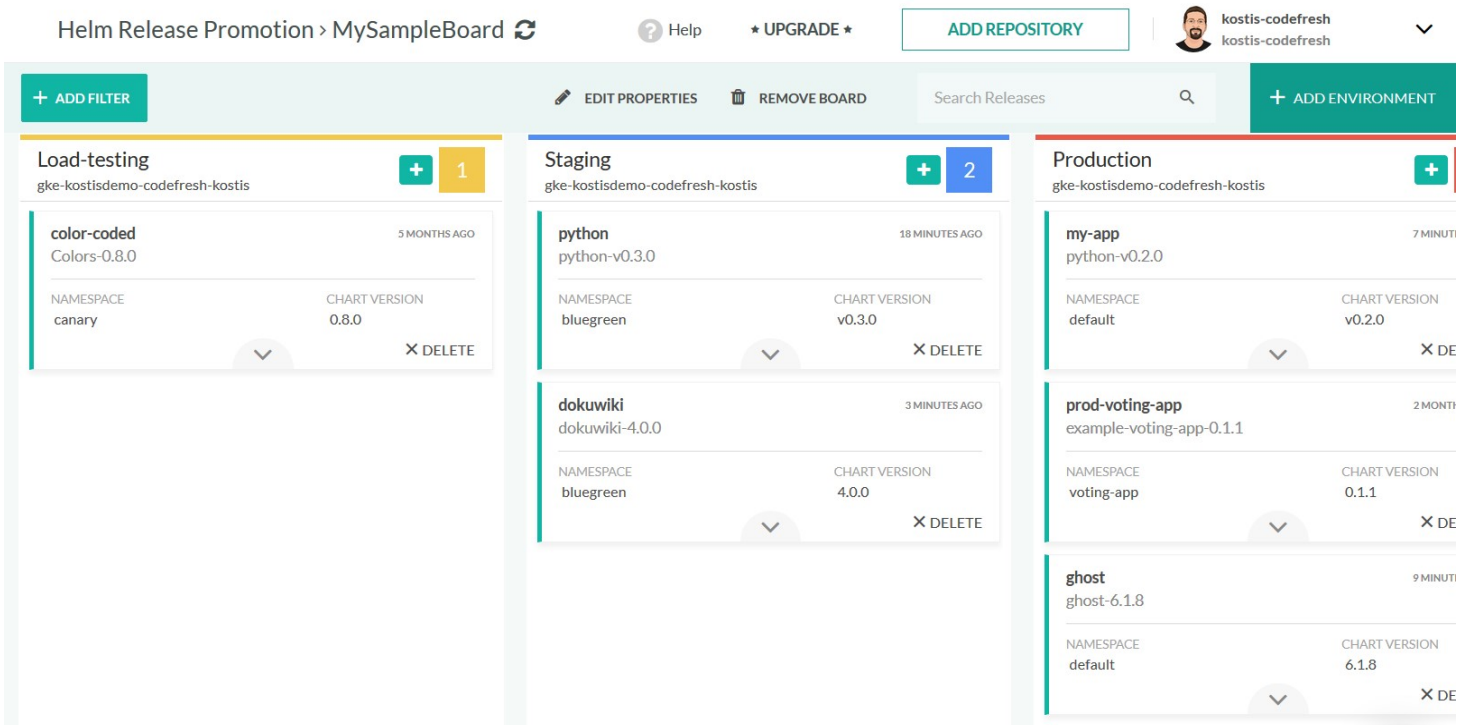
# Observability

## Git history != observability

- Hard to search
- Hard to reason
- Hard to correlate hashes with features
- No information on testing, security scanning, static analysis



# Multiple environments



# Mixing CI and CD

## How to do smoke tests

How to rollback depending on test result?

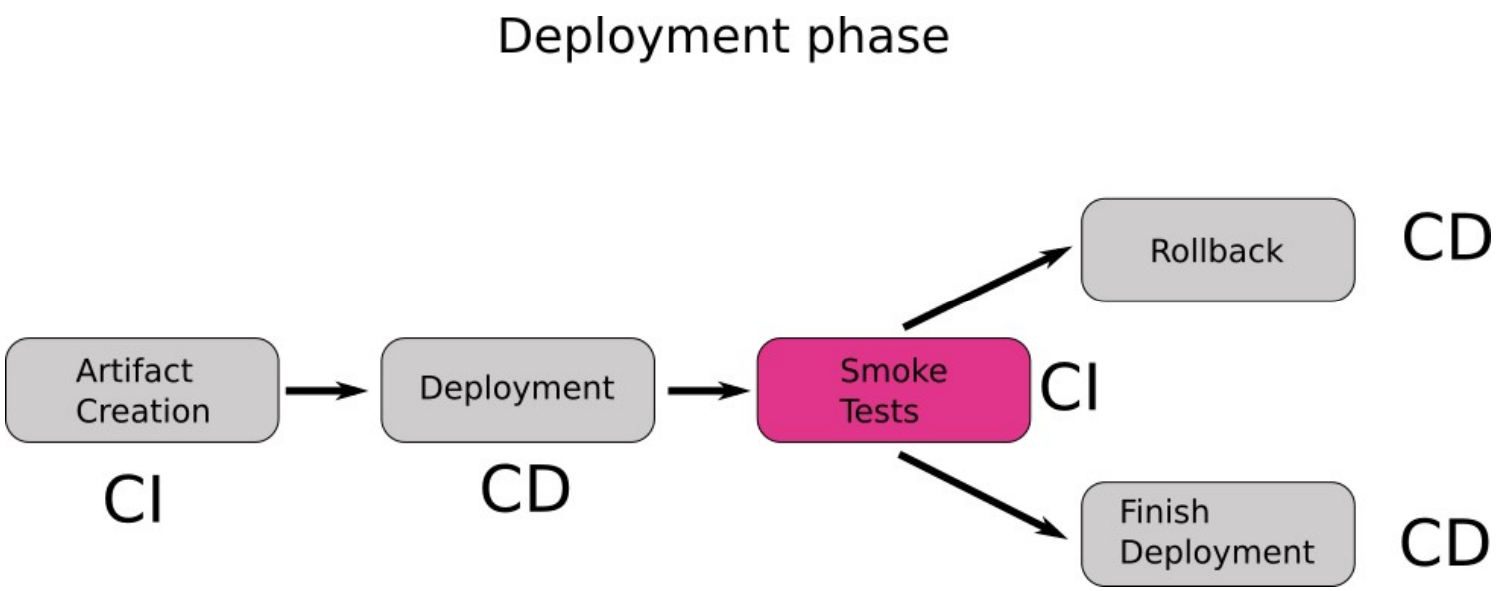
## How to do parallel deployments

How to deploy to US, EU, Asia at the same time?

## Pre and Post deployment steps

How to send a slack message after deployment?

## Rollback on smoke tests



## Other scenarios currently not handled by GitOps tools

### Pre-deployment

- Check network access
- Human Approval
- Mark deployment in metrics
- Validate container image
- Disable alerts

### Post-deployment

- Send slack message
- Mark deployment in metrics
- Trigger another deployment
- Cleanup resources

# GitOps at scale

## Multiple environments
Understand correlation of multiple deployments

## Dynamic resources break
Deploy by matching Git state with cluster state

## No adhoc actions
Operations by Pull Request only

GitOps at scale

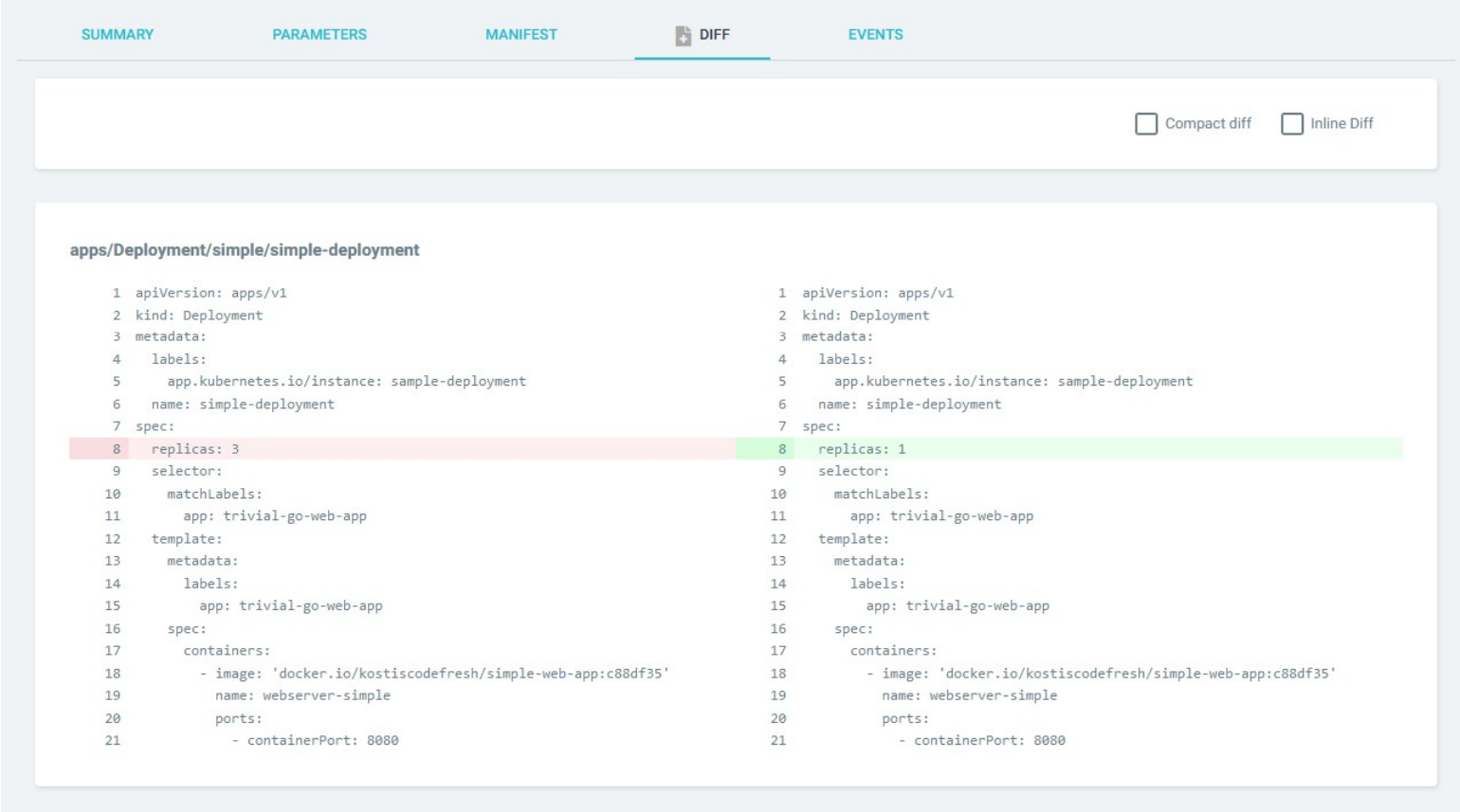
## For 10 environments

- 10 distinct Git repositories?
- Multiple parent-child repositories?
- 1 Git repository with 10 folders?
- 1 Git repository with 10 branches?

## Other problems

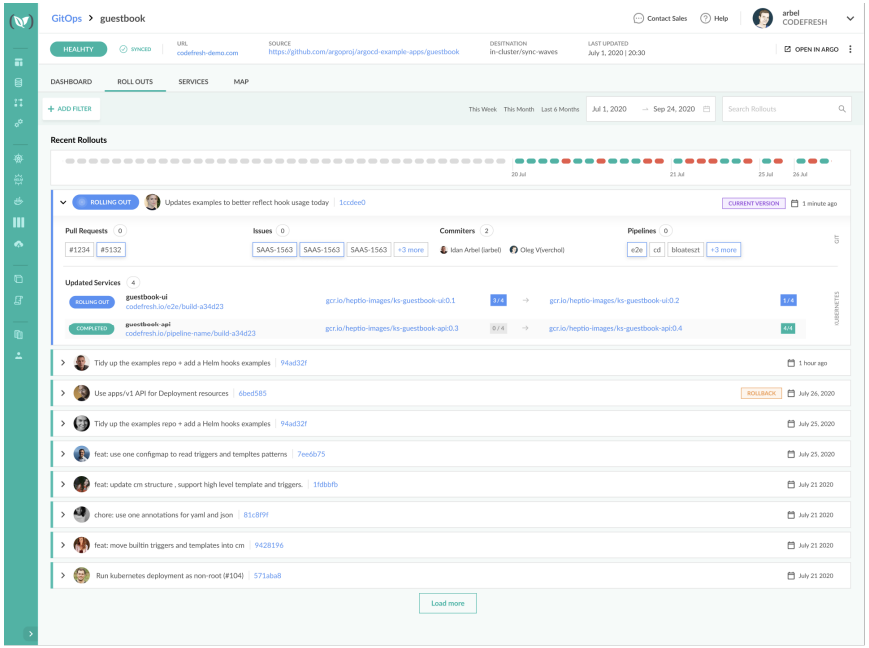
- How to find common settings?
- How to promote releases?
- How to co-ordinate deployments
- How to handle auditing?

## GitOps and dynamic resources



# A vision for GitOps 2.0

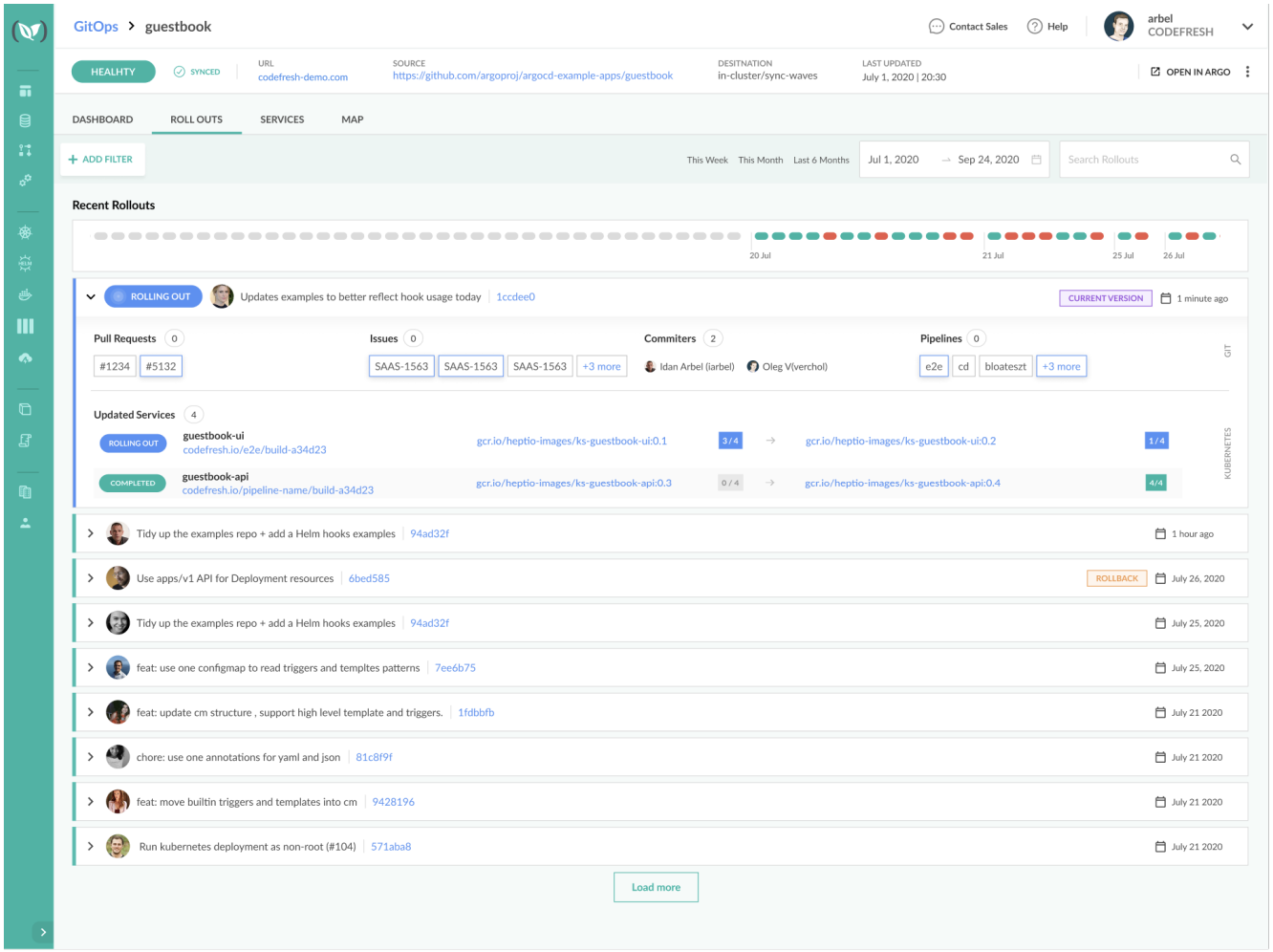
Solving the challenges



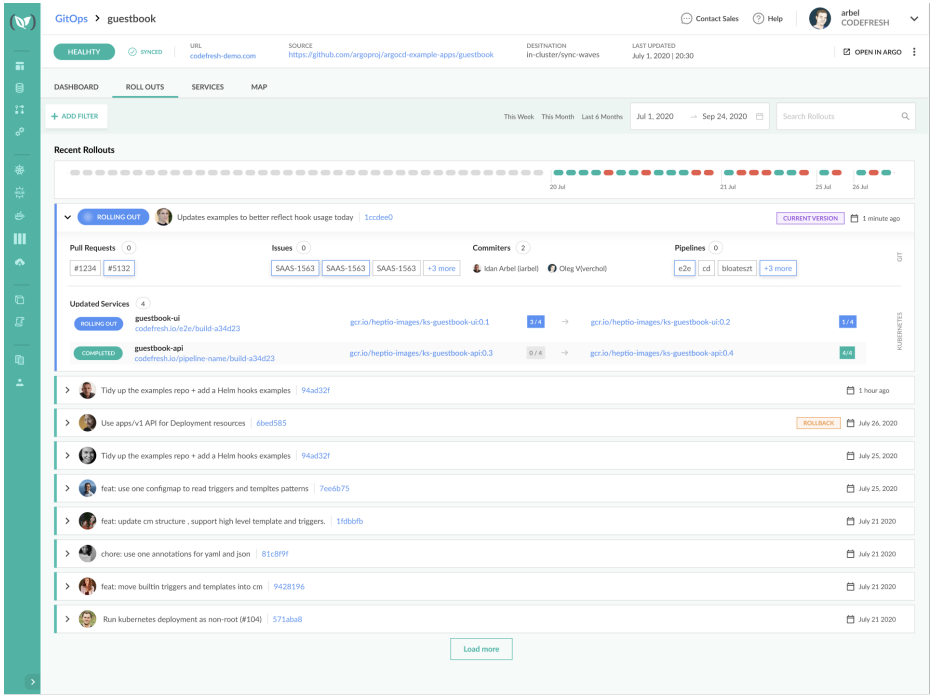

# Codefresh GitOps

Based on ArgoCD

# GitOps dashboard



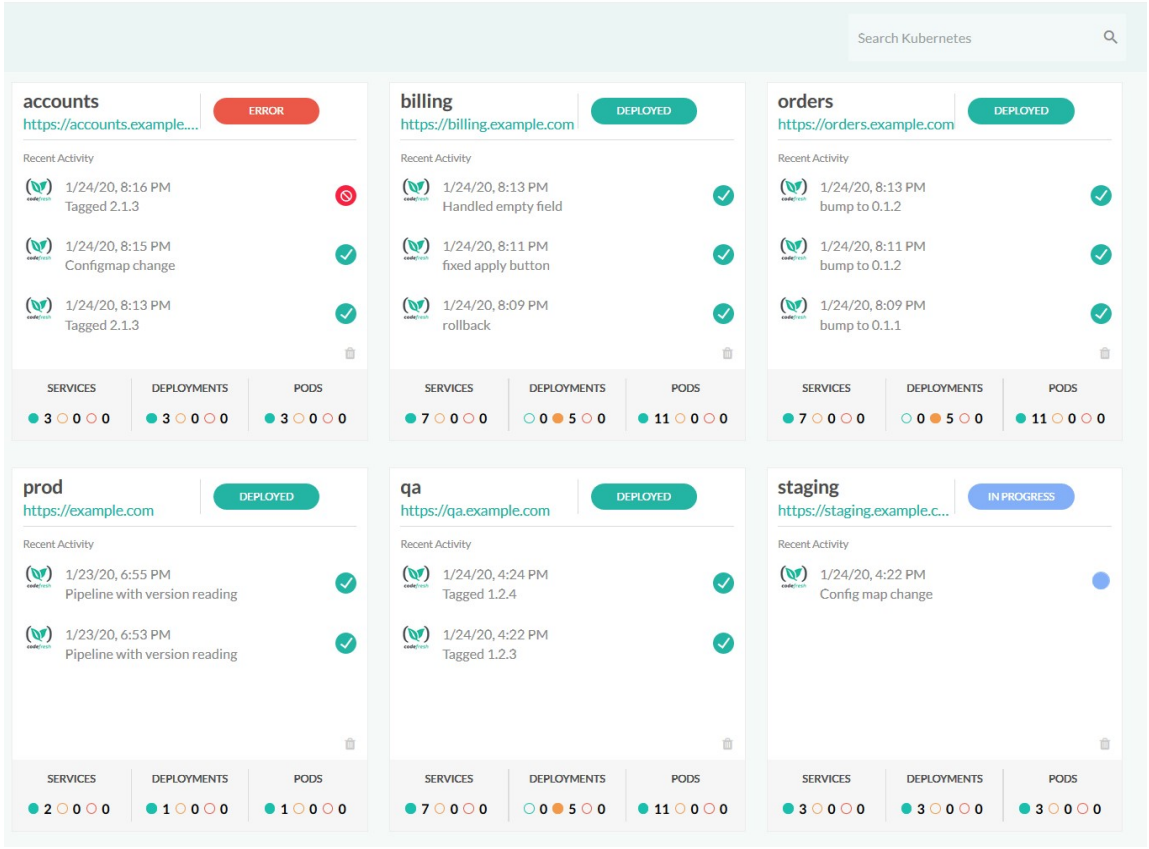
## Easily correlate features and deployments



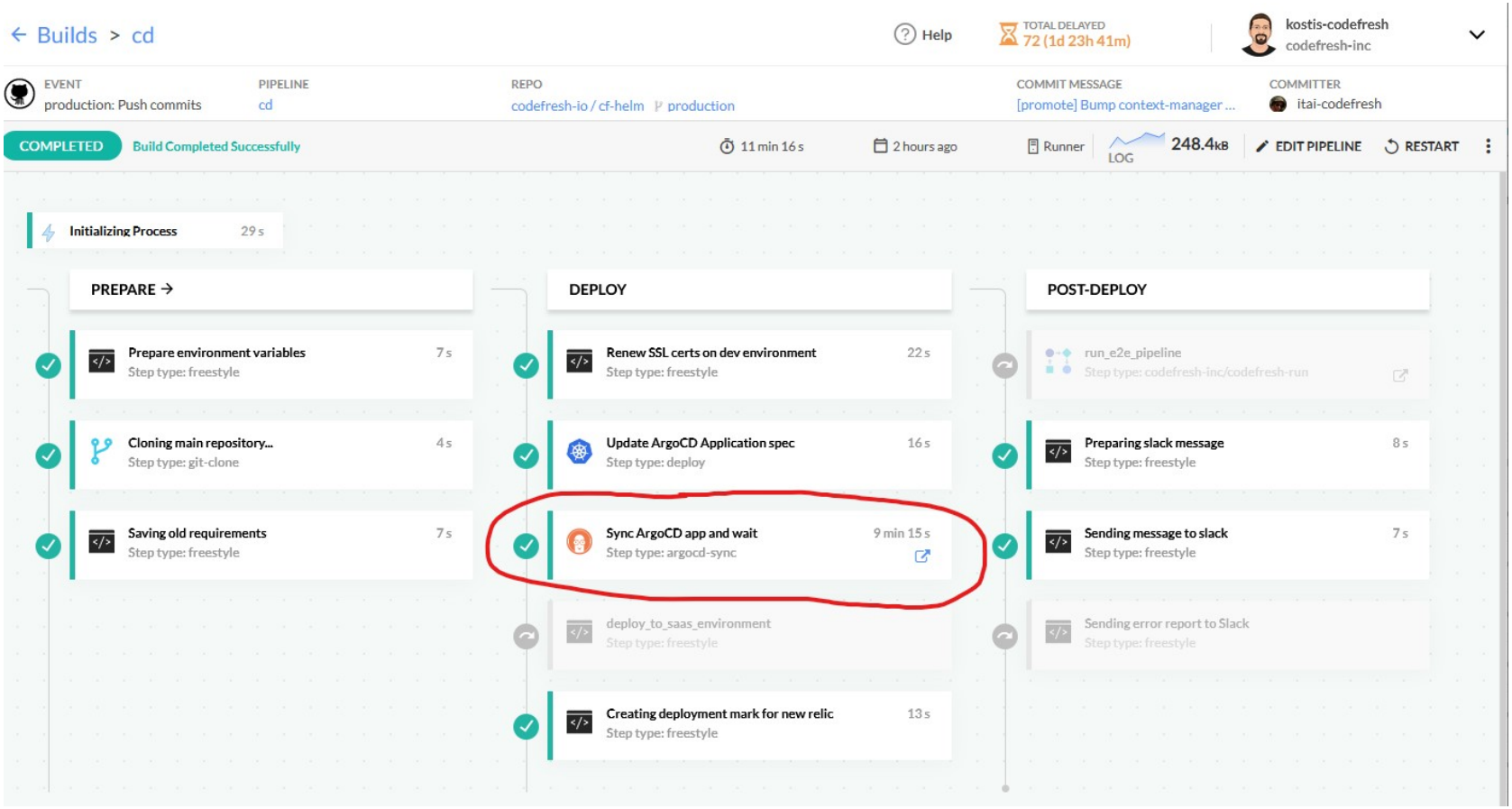

# Codefresh GitOps

- See which features are in deployment
- See which Pull Requests affected a deployment
- Inspect result of unit tests and security scans
- Understand affected Kubernetes services

# Multi-environment dashboard



## Pre/Post deployment steps



# The GitOps working group

A CNCF community project

## The GitOps working group





### An open working group

The founders of the GitOps Working Group are creating a neutral working group to clearly define a principle-lead meaning of GitOps to better enable interoperability between tools. With a clear definition, GitOps Certification Programs for individuals will also be possible.

https://github.com/fluxcd/gitops-working-group/

**GitOps challenges and a vision for GitOps 2.0**


https://codefresh.io/devops/pains-gitops-1-0/
https://codefresh.io/devops/vision-gitops-2-0/
https://codefresh.io/devops/entering-gitops-2-0/
https://codefresh.io/devops/launching-future-devops-gitops-2-0/
https://codefresh.io/devops/announcing-gitops-working-group/