# Developer Experience on Continuous Delivery

## Building a CD system for k8s that developers LOVE

**Euccas Chen**

Software Engineer

**Tobi Ogunnaike**

Software Engineer

Photo by Markus Spiske

## Mission

# Bring everyone the inspiration to create a life they love!

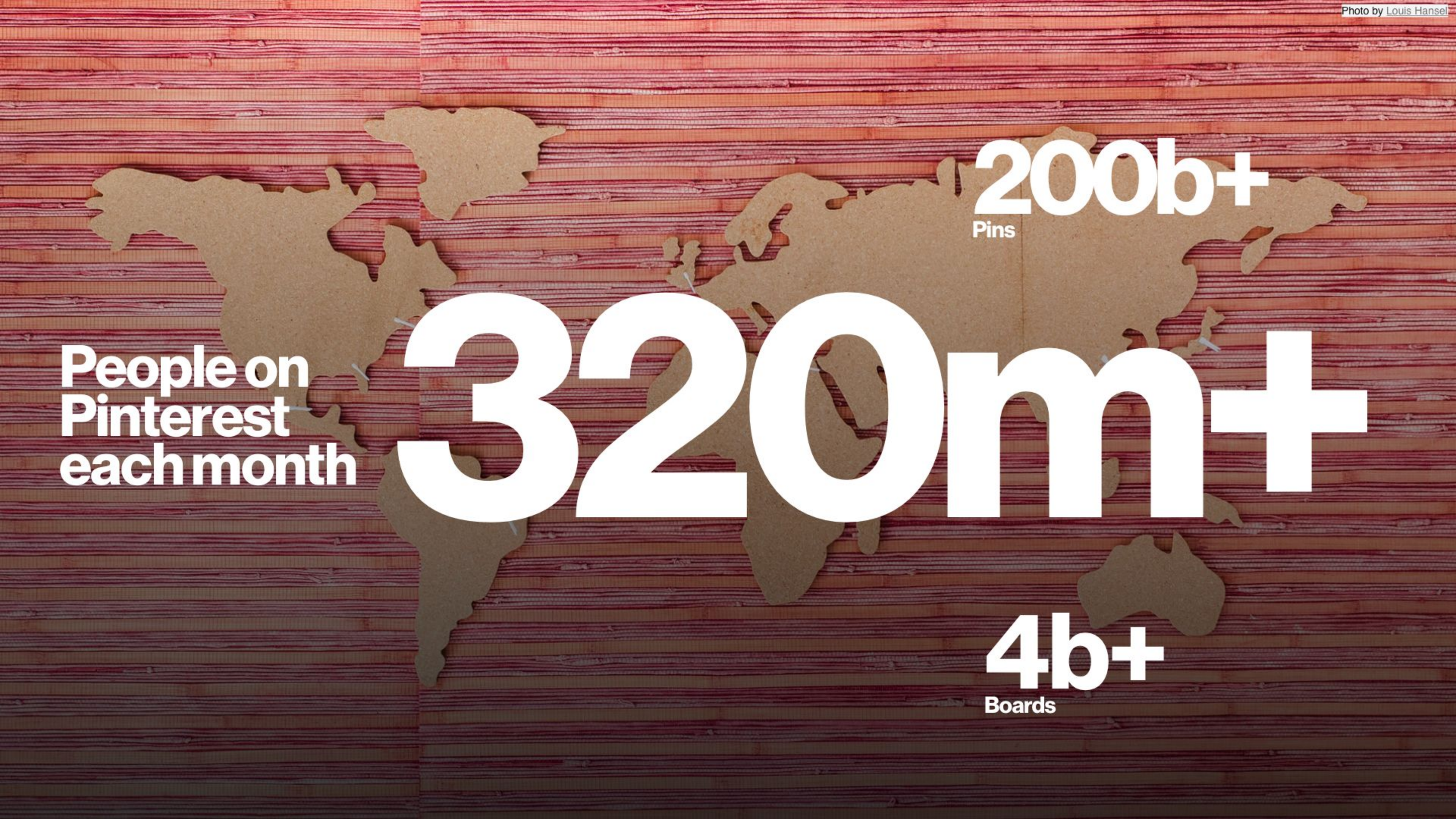


Photo by Louis Hansel

People on Pinterest each month

320m+

200b+

Pins

4b+

Boards

# Infrastructure Footprint

$O(10^5)$
of servers

$O(10^4)$
of deploys / month

$O(10^3)$
of services

## Engineering Productivity

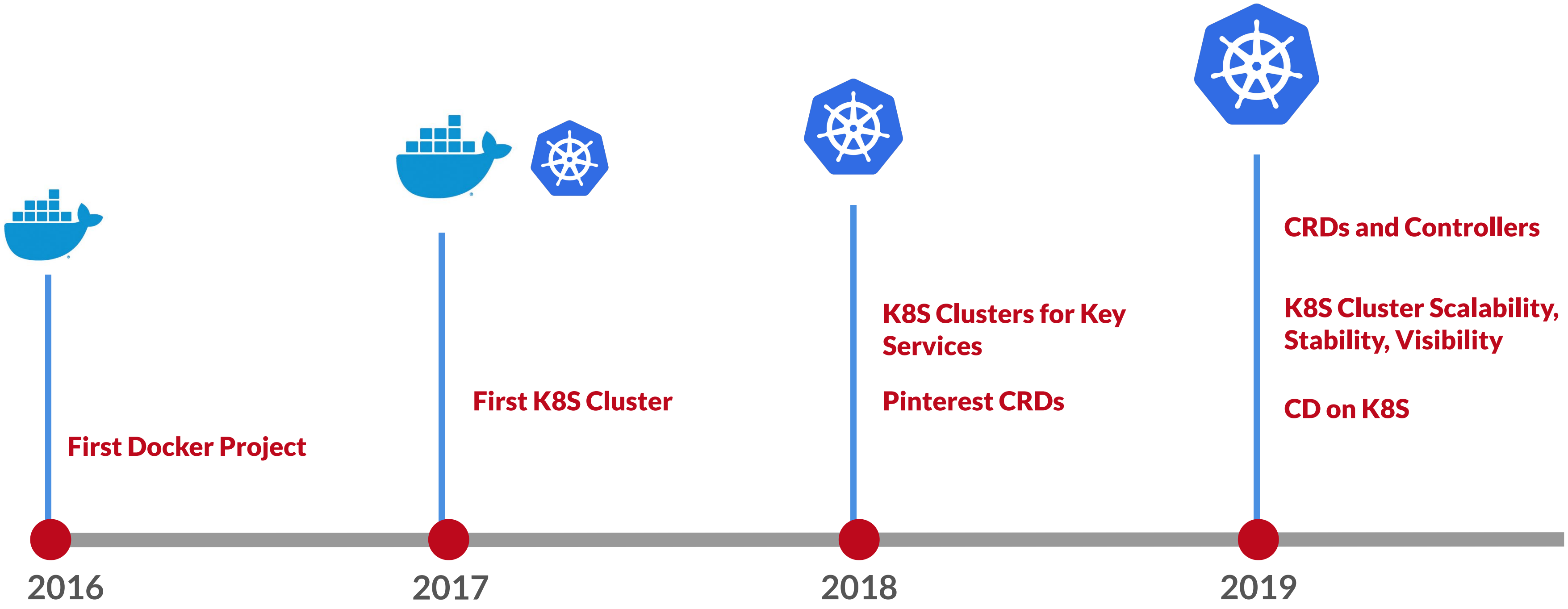
# Fast, safe & delightful path from an idea to production, without worrying about infrastructure

Photo by Markus Winkler

# Agenda

1. Build a CD system for k8s
2. Adoption and migration
3. Lessons we learned

# Kubernetes at Pinterest



2016

First Docker Project

2017

First K8S Cluster

2018

K8S Clusters for Key Services

Pinterest CRDs

2019

CRDs and Controllers

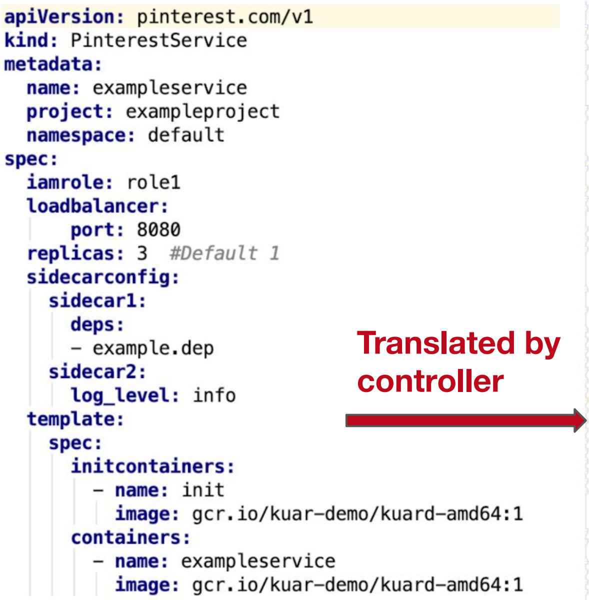
K8S Cluster Scalability, Stability, Visibility

CD on K8S

# Custom resources and controllers

## Pinterest CRD

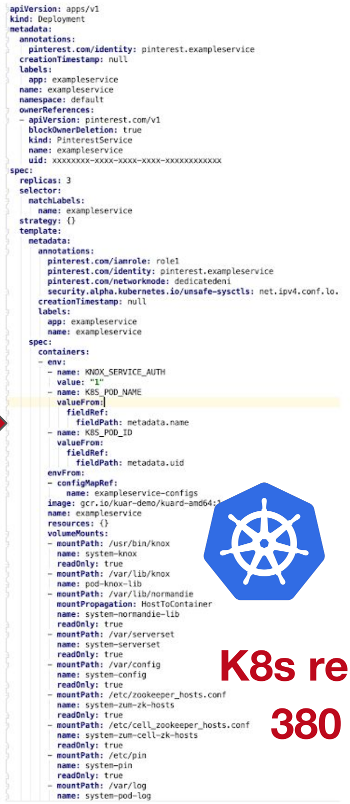
- Model unique workloads
- Inject runtime support
- Simplified config
- 6 CRD types



```yaml
apiVersion: pinterest.com/v1
kind: PinterestService
metadata:
  name: exampleservice
  project: exampleproject
  namespace: default
spec:
  iamrole: role1
  loadbalancer:
      port: 8080
  replicas: 3  #Default 1
  sidecarconfig:
    sidecar1:
      deps:
      - example.dep
    sidecar2:
      log_level: info
  template:
    spec:
      initcontainers:
        - name: init
          image: gcr.io/kuar-demo/kuard-amd64:1
      containers:
        - name: exampleservice
          image: gcr.io/kuar-demo/kuard-amd64:1
```

CRD, 25 lines

Translated by controller



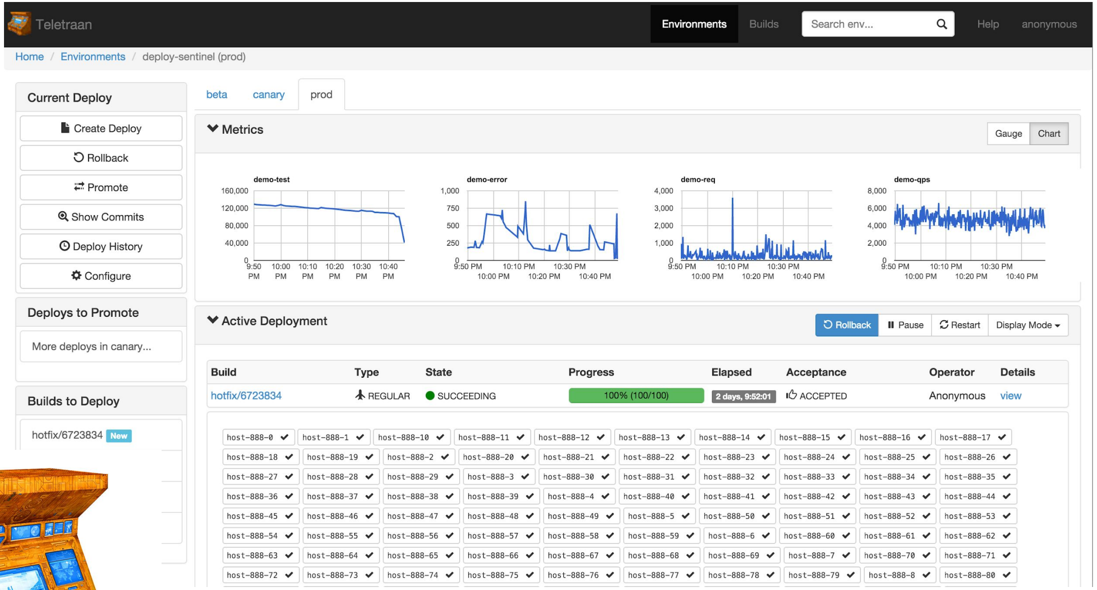
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    pinterest.com/identity: pinterest.exampleservice
  creationTimestamp: null
  labels:
    app: exampleservice
  name: exampleservice
  namespace: default
  ownerReferences:
  - apiVersion: pinterest.com/v1
    blockOwnerDeletion: true
    kind: PinterestService
    name: exampleservice
    uid: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
spec:
  replicas: 3
  selector:
    matchLabels:
      name: exampleservice
  strategy: {}
  template:
    metadata:
      annotations:
        pinterest.com/iamrole: role1
        pinterest.com/identity: pinterest.exampleservice
        pinterest.com/networkmode: dedicatedeni
        security.alpha.kubernetes.io/unsafe-sysctls: net.ipv4.conf.lo.
      creationTimestamp: null
      labels:
        app: exampleservice
        name: exampleservice
    spec:
      containers:
      - env:
        - name: KNOX_SERVICE_AUTH
          value: "1"
        - name: K8S_POD_NAME
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
        - name: K8S_POD_ID
          valueFrom:
            fieldRef:
              fieldPath: metadata.uid
        envFrom:
        - configMapRef:
            name: exampleservice-configs
        image: gcr.io/kuar-demo/kuard-amd64:1
        name: exampleservice
        resources: {}
        volumeMounts:
        - mountPath: /usr/bin/knox
          name: system-knox
          readOnly: true
        - mountPath: /var/lib/knox
          name: pod-knox-lib
        - mountPath: /var/lib/normandie
          mountPropagation: HostToContainer
          name: system-normandie-lib
          readOnly: true
        - mountPath: /var/serverset
          name: system-serverset
          readOnly: true
        - mountPath: /var/config
          name: system-config
          readOnly: true
        - mountPath: /etc/zookeeper_hosts.conf
          name: system-zum-zk-hosts
          readOnly: true
        - mountPath: /etc/cell_zookeeper_hosts.conf
          name: system-zum-cell-zk-hosts
          readOnly: true
        - mountPath: /etc/pin
          name: system-pin
          readOnly: true
        - mountPath: /var/log
          name: system-pod-log
```

K8s resource, 380 lines

# Current deployment system

## Teletraan

- Deploy code to VMs
- Running since 2016
- 1K services
- 5K deploys / day



Teletraan | Environments | Builds | Search env... | Help | anonymous

Home / Environments / deploy-sentinel (prod)

Current Deploy
- Create Deploy
- Rollback
- Promote
- Show Commits
- Deploy History
- Configure

Deploys to Promote
- More deploys in canary...

Builds to Deploy
- hotfix/6723834 New

beta | canary | prod

Metrics | Gauge | Chart

Active Deployment | Rollback | Pause | Restart | Display Mode

| Build | Type | State | Progress | Elapsed | Acceptance | Operator | Details |
|---|---|---|---|---|---|---|---|
| hotfix/6723834 | REGULAR | ● SUCCEEDING | 100% (100/100) | 2 days, 9:52:01 | ACCEPTED | Anonymous | view |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| host-888-0 ✓ | host-888-1 ✓ | host-888-10 ✓ | host-888-11 ✓ | host-888-12 ✓ | host-888-13 ✓ | host-888-14 ✓ | host-888-15 ✓ | host-888-16 ✓ | host-888-17 ✓ |
| host-888-18 ✓ | host-888-19 ✓ | host-888-2 ✓ | host-888-20 ✓ | host-888-21 ✓ | host-888-22 ✓ | host-888-23 ✓ | host-888-24 ✓ | host-888-25 ✓ | host-888-26 ✓ |
| host-888-27 ✓ | host-888-28 ✓ | host-888-29 ✓ | host-888-3 ✓ | host-888-30 ✓ | host-888-31 ✓ | host-888-32 ✓ | host-888-33 ✓ | host-888-34 ✓ | host-888-35 ✓ |
| host-888-36 ✓ | host-888-37 ✓ | host-888-38 ✓ | host-888-39 ✓ | host-888-4 ✓ | host-888-40 ✓ | host-888-41 ✓ | host-888-42 ✓ | host-888-43 ✓ | host-888-44 ✓ |
| host-888-45 ✓ | host-888-46 ✓ | host-888-47 ✓ | host-888-48 ✓ | host-888-49 ✓ | host-888-5 ✓ | host-888-50 ✓ | host-888-51 ✓ | host-888-52 ✓ | host-888-53 ✓ |
| host-888-54 ✓ | host-888-55 ✓ | host-888-56 ✓ | host-888-57 ✓ | host-888-58 ✓ | host-888-59 ✓ | host-888-6 ✓ | host-888-60 ✓ | host-888-61 ✓ | host-888-62 ✓ |
| host-888-63 ✓ | host-888-64 ✓ | host-888-65 ✓ | host-888-66 ✓ | host-888-67 ✓ | host-888-68 ✓ | host-888-69 ✓ | host-888-7 ✓ | host-888-70 ✓ | host-888-71 ✓ |
| host-888-72 ✓ | host-888-73 ✓ | host-888-74 ✓ | host-888-75 ✓ | host-888-76 ✓ | host-888-77 ✓ | host-888-78 ✓ | host-888-79 ✓ | host-888-8 ✓ | host-888-80 ✓ |

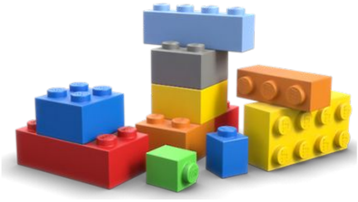
https://github.com/pinterest/teletraan

# Hermez { Design, Build }

Tl;dr: We are building a new Continuous Delivery system for Kubernetes at Pinterest.

# Deploying to k8s: Challenges

## What problems are we solving?



Complexity

Operational toil

640K RAM exceeded

Pinterest specific

# Deploying to k8s: What we want

## Make it easy

Abstract away complexity

Minimal configs

Single interface

## End to end

From code commits to deployment

Visibility

Debuggability

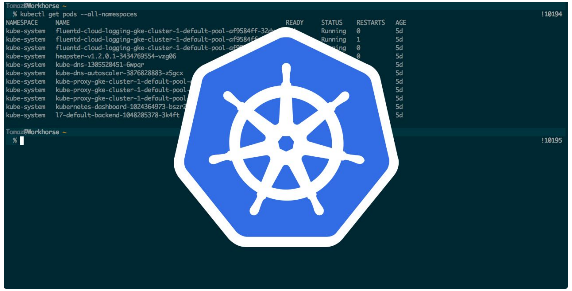
## Customization

Integrate with existing infra systems

Deployment pipelines

Migrate from Teletraan

# Existing Solutions ?



```
Tomaz@Workhorse ~
  % kubectl get pods --all-namespaces                                                              !10194
NAMESPACE     NAME                                                      READY     STATUS    RESTARTS   AGE
kube-system   fluentd-cloud-logging-gke-cluster-1-default-pool-af9584ff-32d           Running   0          5d
kube-system   fluentd-cloud-logging-gke-cluster-1-default-pool-af9584ff               Running   1          5d
kube-system   fluentd-cloud-logging-gke-cluster-1-default-pool-af95                   ing       0          5d
kube-system   heapster-v1.2.0.1-3434769554-vzg06                                                0          5d
kube-system   kube-dns-1305520451-6wpqr                                                                    5d
kube-system   kube-dns-autoscaler-3876828883-z5gcx                                                         5d
kube-system   kube-proxy-gke-cluster-1-default-pool-                                                       5d
kube-system   kube-proxy-gke-cluster-1-default-pool-                                                       5d
kube-system   kube-proxy-gke-cluster-1-default-pool                                                        5d
kube-system   kubernetes-dashboard-1024364973-bszr2                                                        5d
kube-system   l7-default-backend-1048205378-3k4ft                                                          5d

Tomaz@Workhorse ~
  % █                                                                                              !10195
```

kubectl

pcloud

U.S.
MAIL
43

# Introducing Hermez

1. The user-facing system for CD
2. Kubernetes first
3. Delightful developer experience

Hermez

Hermez

# Workloads

## Easy configuration

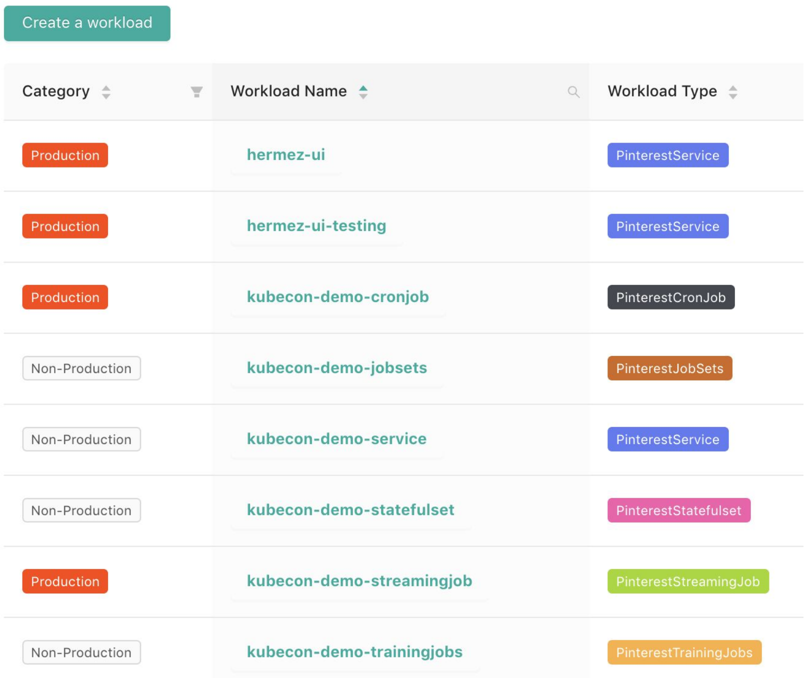
- Code repository
- K8s config file

## Workload types

- K8s: workload types defined by Pinterest CRDs
- Data streaming, Teletraan

## Operation support

- Workload healthiness, metrics, config change audit trail, authZ, notification



| Category | Workload Name | Workload Type |
|---|---|---|
| Production | hermez-ui | PinterestService |
| Production | hermez-ui-testing | PinterestService |
| Production | kubecon-demo-cronjob | PinterestCronJob |
| Non-Production | kubecon-demo-jobsets | PinterestJobSets |
| Non-Production | kubecon-demo-service | PinterestService |
| Non-Production | kubecon-demo-statefulset | PinterestStatefulset |
| Production | kubecon-demo-streamingjob | PinterestStreamingJob |
| Non-Production | kubecon-demo-trainingjobs | PinterestTrainingJobs |

Hermez

# Deployments

## Deploy commits and PRs

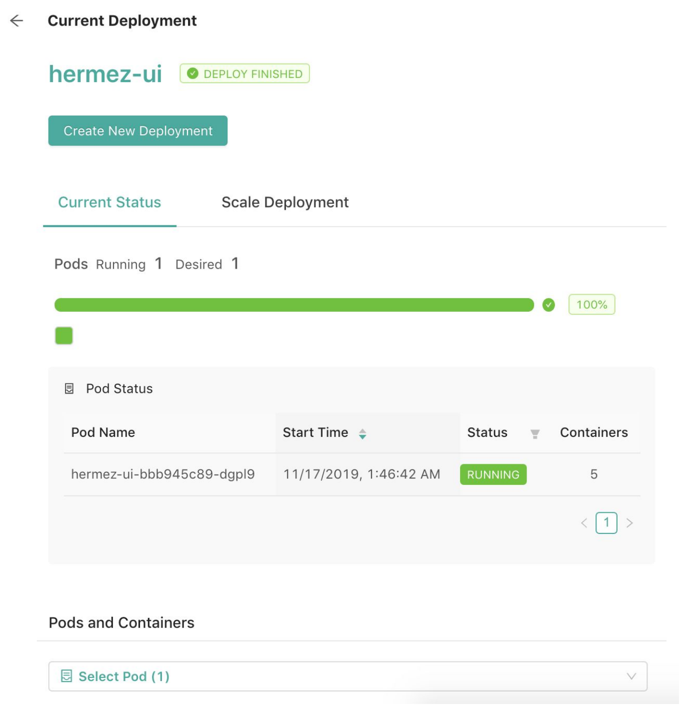
- Rollback, hotfix
- Scale a deployment: manual, auto-scaling

## Continuous Delivery pipelines

- Integrate with Spinnaker to run pipelines

## Observability

- Current running version, deployment details
- K8s: Pod and container status, events, logs
- Deployment history



← Current Deployment

hermez-ui DEPLOY FINISHED

Create New Deployment

Current Status | Scale Deployment

Pods Running 1 Desired 1

100%

Pod Status

| Pod Name | Start Time | Status | Containers |
|---|---|---|---|
| hermez-ui-bbb945c89-dgpl9 | 11/17/2019, 1:46:42 AM | RUNNING | 5 |

< 1 >

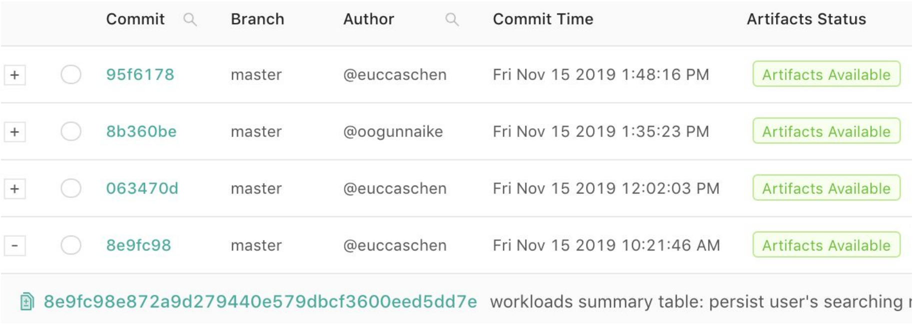
Pods and Containers

Select Pod (1)

# CI Integration

## Build pipelines

- Support individual service's repo and monorepos
- Build container images
- Publish k8s artifacts



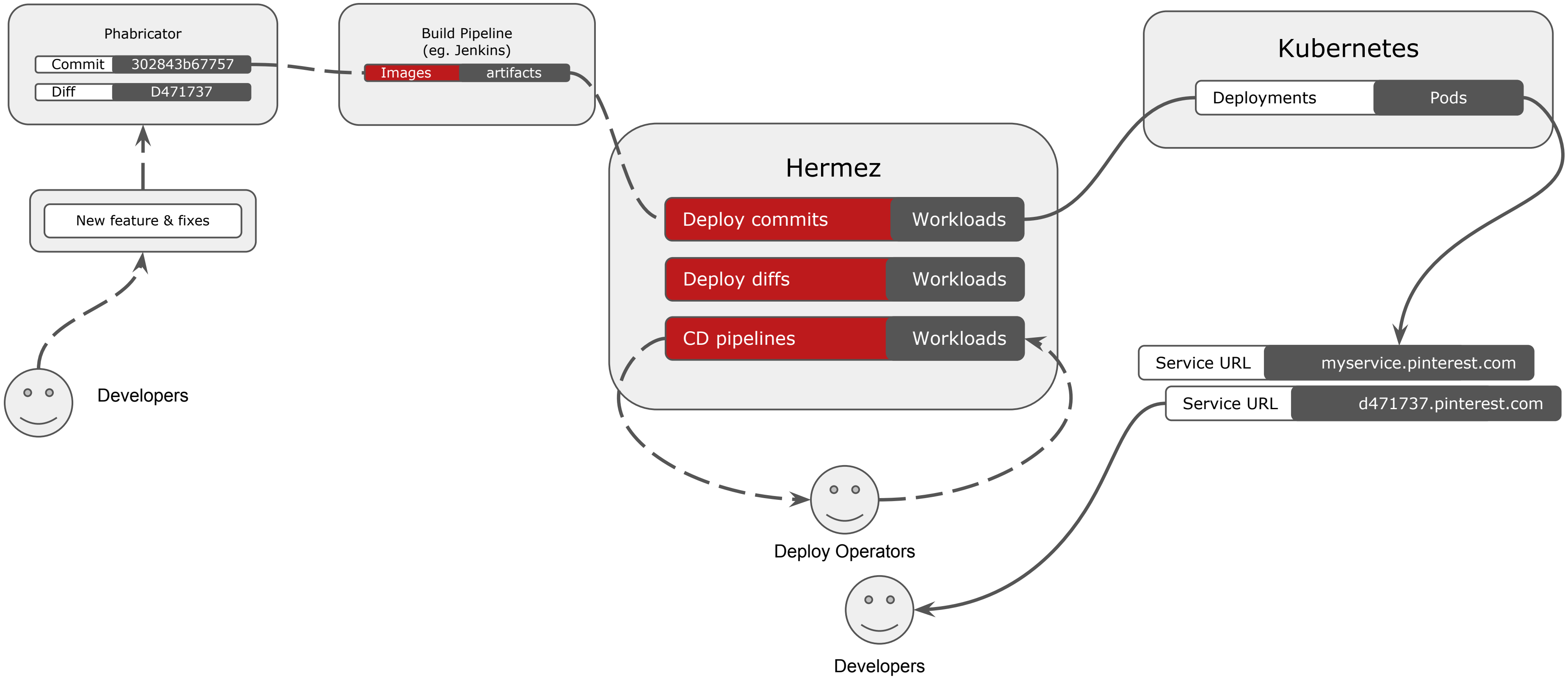
|  |  | Commit | Branch | Author | Commit Time | Artifacts Status |
|---|---|---|---|---|---|---|
| + | ○ | 95f6178 | master | @euccaschen | Fri Nov 15 2019 1:48:16 PM | Artifacts Available |
| + | ○ | 8b360be | master | @oogunnaike | Fri Nov 15 2019 1:35:23 PM | Artifacts Available |
| + | ○ | 063470d | master | @euccaschen | Fri Nov 15 2019 12:02:03 PM | Artifacts Available |
| - | ○ | 8e9fc98 | master | @euccaschen | Fri Nov 15 2019 10:21:46 AM | Artifacts Available |

8e9fc98e872a9d279440e579dbcf3600eed5dd7e workloads summary table: persist user's searching

## Bridge CI and CD

- Visualize the process of "from code commits to deployment"
- Logs for debugging
- Trigger build on-demand

# A developer’s experience



# Our path to a new CD system

1

## Design

User story

UI mockup

Feedback sessions with teams

2

## MVP

Minimal set of features

Hackathon

Gather feedback

3

## Dogfood

Use Hermez to deploy Hermez

Find early adopters

Iterate and learn

4

## Production

Break into smaller scopes

Onboard new services

Migrate existing services

# Hermez { Adoption, Migration }

Tl;dr: We are helping Pinterest engineering teams to deploy and migrate their services onto Kubernetes using the new CD system.

# Customer adoption is not easy

***“If you build it, they will come”***

**Said no successful product owner ever.**

# What can Hermez do for me?

And why should I care?

# Demo time for Service!

*Insert video link here*

# Service demo

## Call outs

1. **Deploy PRs with easily shareable URLs** #feature

2. **Easy integration with existing systems** *#minimal-config*

   build systems, artifact stores and docker registry

3. **Debuggability** *#dev-experience*

   container logs, pod status, workload metrics

# Demo time for Cronjob!

*Insert video link here*

# Cronjob demo

## Context - No single, recommended path for deploying cronjobs at Pinterest

### Call outs

1. **First class support for cronjob operation** *#feature*
   view cronjob schedule, execution history, next scheduled run in the UI

2. **Easy integration with existing systems** *#minimal-config*
   build systems, artifact stores and docker registry

3. **Debuggability** *#dev-experience*
   container logs, pod status, workload metrics

# How we prepared Hermez for adoption

1. Reduced scope
   Limited workload types (PinterestService, PinterestCronjob)

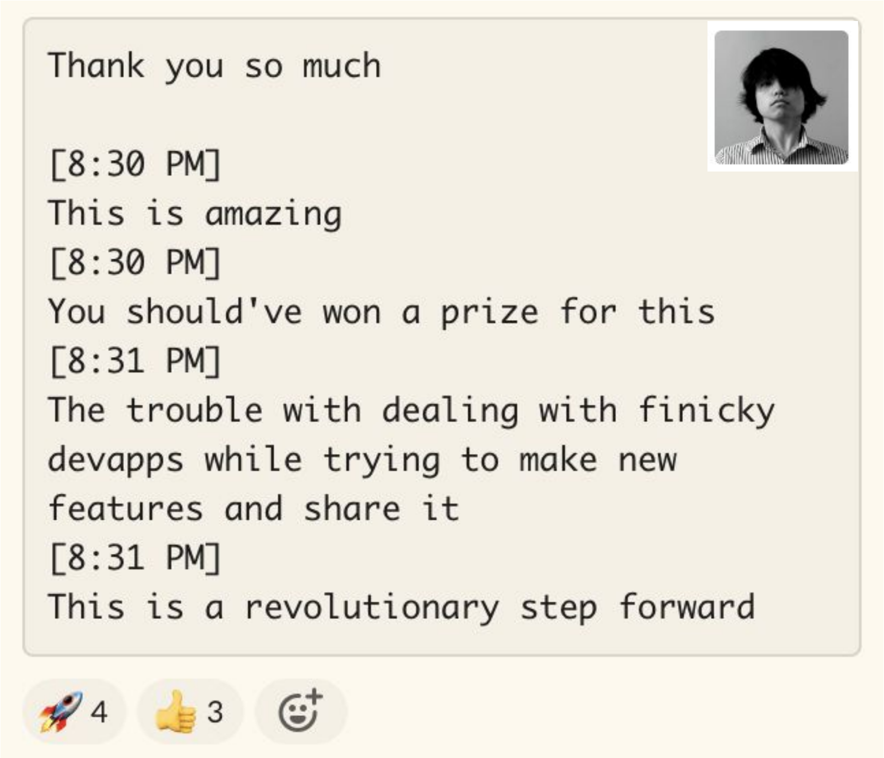
2. Partner with **early adopters**
   (SRE, Ads, Tools)

3. Evaluate feedback, iterate & improve

4. Knowledge sharing - onboard runbook, status updates, demos, brownbags

5. Self-service migration tools

# Homefeed + Hermez - a fairytale migration story



Thank you so much

[8:30 PM]
This is amazing
[8:30 PM]
You should've won a prize for this
[8:31 PM]
The trouble with dealing with finicky devapps while trying to make new features and share it
[8:31 PM]
This is a revolutionary step forward

🚀 4 👍 3

# Hermez { Learning }

Tl;dr: We want to share our experience and collaborate with the community.