# KUBERNETES 101

*from zero to cluster in 30 minutes or less*

# WHY?

- Connect Native: 2 processes + 1 db
- Standardized way to *orchestrate* systems
- Getting easier and cheaper to start

# PROGRAM

1. Demo app
2. Quick intro to Kubernetes
3. Live hacking

# DEMO: LAUNCHPAD

- Preview upcoming SpaceX launch
- 2 Rails processes
- 1 Postgres DB

# KUBERNETES

- System to automate deployments
- Keeps processes running on servers
- Simplifies connecting them together
- Standardized way to define infrastructure as code

# RESOURCES

**`Pod`**

akin to Docker container

**`Service`**

name things for discovery in cluster, expose ports

**`Ingress`**

expose HTTP services to outside world

# KEY PIECES

**`kubectl`**

command-line program to interact with cluster

**`kubeconfig`**

configuration/credential file from cloud provider

# DISCLAIMER

- Simplicity first, not best practices
- My goal for this

# HACKING

# KUBECONFIG

Goes in `~/.kube/config`

```
$ kubectl config use-context launchpad-staging
```

```
$ kubectl config use-context launchpad-production
```

# VIEW RESOURCES IN CLUSTER

```
$ kubectl get pods,services,ingresses
```

```
$ kubectl get all
```

# RUN FIRST POD

```
$ kubectl run web \
    --image=czak/launchpad \
    --env=RAILS_ENV=production \
    --env=RAILS_LOG_TO_STDOUT=true \
    --env=RAILS_SERVE_STATIC_FILES=true \
    --env=RAILS_MASTER_KEY=8bd9f3468d31f56f8cf23a952099f96b \
    --env=DATABASE_URL=postgres://launchpad:topsecret@database/la
    -- \
    bin/rails server
```

# CONNECT TO POD

```
$ kubectl exec -it web -- bin/rails console
```

# CONNECT TO POD PT. 2

```
$ kubectl port-forward web 3000
```

# POD IN YAML

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: web
  labels:
    app: web
spec:
  containers:
    - name: web
      image: czak/launchpad
      command: ["bin/rails", "server"]
      env:
        - name: RAILS_ENV
          value: production
        - name: RAILS_LOG_TO_STDOUT
          value: "true"
        - name: RAILS_SERVE_STATIC_FILES
          value: "true"
        - name: RAILS_MASTER_KEY
          value: 8bd9f3468d31f56f8cf23a952099f96b
        - name: DATABASE_URL
          value: postgres://launchpad:topsecret@database/launchpa
```

# RUN POD FROM YAML

```
$ kubectl apply -f web.yml
```

# CHECK LOGS

```
$ kubectl logs web
```

# RUN MORE PODS

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: database
  labels:
    app: database
spec:
  containers:
    - name: database
      image: postgres:13-alpine
      env:
        - name: POSTGRES_DB
          value: launchpad
        - name: POSTGRES_USER
          value: launchpad
        - name: POSTGRES_PASSWORD
          value: topsecret
```

# DECLARE DATABASE SERVICE

```yaml
apiVersion: v1
kind: Service
metadata:
  name: database
spec:
  selector:
    app: database
  ports:
    - port: 5432
```

# WEB → DATABASE

```
$ kubectl exec web -- bin/rails db:migrate
```

# DECLARE WEB SERVICE

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web
spec:
  selector:
    app: web
  ports:
    - port: 3000
```

# EXPOSE WEB SERVICE VIA INGRESS

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: web
spec:
  rules:
    - host: launchpad-staging.czak.pl
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: web
                port:
                  number: 3000
```

# RUN WORKER POD

```
apiVersion: v1
kind: Pod
metadata:
  name: worker
spec:
  containers:
    - name: worker
      image: czak/launchpad
      command: ["bin/rails", "runner", "bin/worker.rb"]
      env:
        - name: RAILS_ENV
          value: production
        - name: RAILS_LOG_TO_STDOUT
          value: "true"
        - name: RAILS_SERVE_STATIC_FILES
          value: "true"
        - name: RAILS_MASTER_KEY
          value: 8bd9f3468d31f56f8cf23a952099f96b
        - name: DATABASE_URL
          value: postgres://launchpad:topsecret@database/launchpa
```

# WHAT HAVE WE DONE?

1. Ran a Rails web server in a `web` pod
2. Ran a Postgres db in a `database` pod
3. Exposed `database` **internally** in the cluster via `Service`
4. Exposed `web` **externally** to the web via `Service+Ingress`
5. Ran a Rails runner process in a `worker` pod

# DEPLOY TO PRODUCTION

```
$ kubectl config use-context launchpad-production
$ kubectl apply -f production/
$ kubectl exec web -- bin/rails db:migrate
```

# NEXT STEPS

- Secrets, Sealed Secrets
- Deployments
- Automation

# THANK YOU :)