

CI/CD con Jenkins y ArgoCD

GitOps

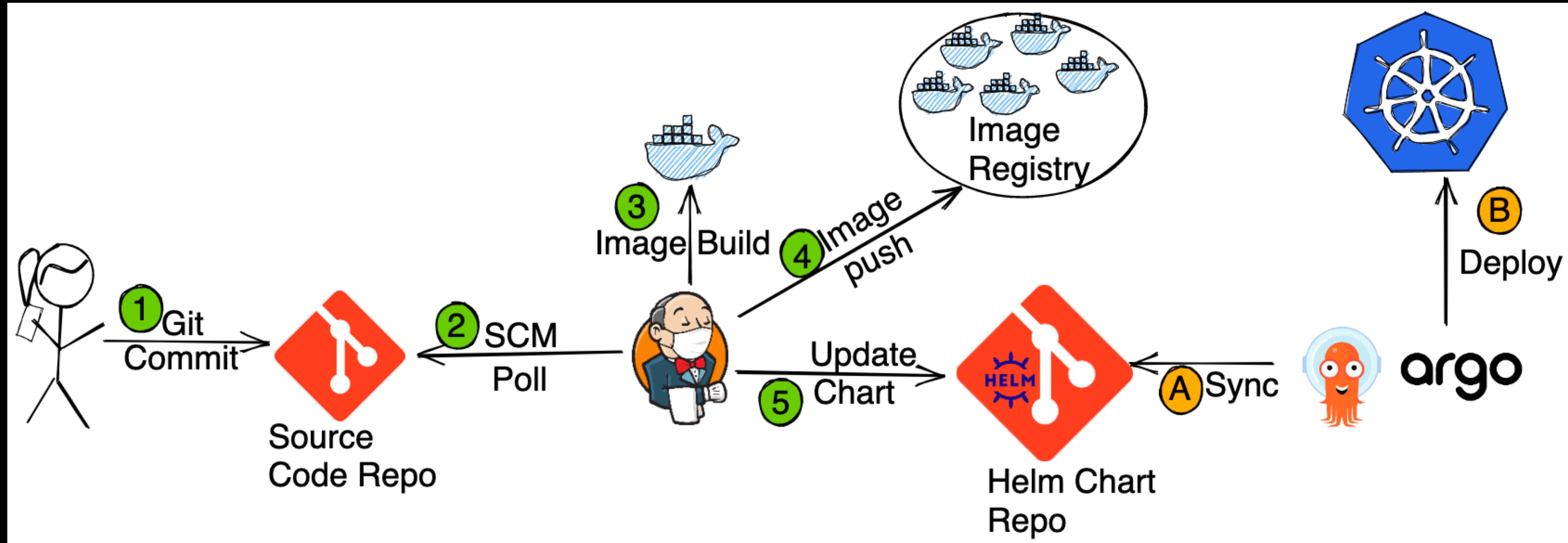
- Su nombre revela que nos gustaría administrar nuestras operaciones como implementación de aplicaciones, administración, escalamiento, etc. vía **Git**.
- Podemos commitear a Git y nuestra aplicación/infraestructura es instalada o actualizada.
- Hay muchas herramientas que soportan esto cómo: ArgoCD, Flux, entre otras. Veremos sólo **ArgoCD**.
- Las herramientas GitOps nos dan las siguientes características:
 - Declarativo
 - Observabilidad
 - Auditoría y compliance
 - Rollback

ArgoCD

<https://argoproj.github.io/cd/>

- Es uno de los proyectos bajo ArgoProj, el cual tiene otros proyecto como Argo Workflows, Argo Rollouts y Argo Events.

GitOps Workflow



Desplegando una aplicación con ArgoCD



Minikube

<https://minikube.sigs.k8s.io/docs/start/>

- curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-darwin-amd64
- sudo install minikube-darwin-amd64 /usr/local/bin/minikube

minikube start --memory 8GB --cpus=4

```
josediaz@MacBook-Pro-de-Jose:~ % minikube start --memory 8GB --cpus=4
Last login: Sun Feb 26 20:10:23 on ttys000
> minikube start --memory 8GB --cpus=4
😄 minikube v1.29.0 on Darwin 13.2.1
:+ Automatically selected the docker driver. Other choices: hyperkit, qemu2, virtualbox, ssh
📌 Using Docker Desktop driver with root privileges
👍 Starting control plane node minikube in cluster minikube
🌐 Pulling base image ...
💻 Downloading Kubernetes v1.26.1 preload ...
> preloaded-images-k8s-v18-v1...: 397.05 MiB / 397.05 MiB 100.00% 6.24 Mi
🔥 Creating docker container (CPUs=4, Memory=8192MB) ...
🐳 Preparing Kubernetes v1.26.1 on Docker 20.10.23 ...
  └ Generating certificates and keys ...
  └ Booting up control plane ...
  └ Configuring RBAC rules ...
🔗 Configuring bridge CNI (Container Networking Interface) ...
  └ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🔍 Verifying Kubernetes components...
⭐ Enabled addons: storage-provisioner, default-storageclass

❗ /Users/josediaz/bin/kubectl is version 1.23.7-eks-4721010, which may have incompatibilities with Kubernetes 1.26.1.
  └ Want kubectl v1.26.1? Try 'minikube kubectl -- get pods -A'
⚡ Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
~ > █ 1m 43s 20:16:28
```

Instalar ArgoCD

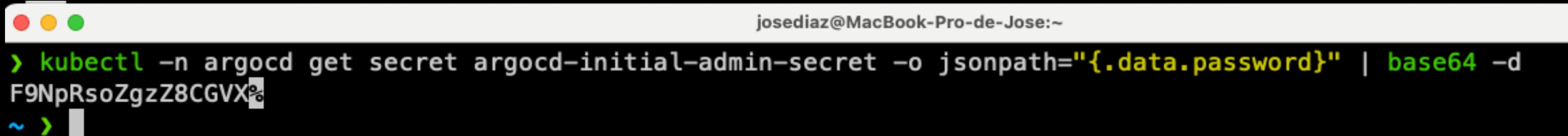
- kubectl create ns argocd
- kubectl apply -n argocd -f <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>
- kubectl get pods -n argocd

kubectl get pods -n argocd

```
josediaz@MacBook-Pro-de-Jose:~ % kubectl get pods -n argocd
configmap/argocd-ssh-known-hosts-cm created
configmap/argocd-tls-certs-cm created
secret/argocd-notifications-secret created
secret/argocd-secret created
service/argocd-applicationset-controller created
service/argocd-dex-server created
service/argocd-metrics created
service/argocd-notifications-controller-metrics created
service/argocd-redis created
service/argocd-repo-server created
service/argocd-server created
service/argocd-server-metrics created
deployment.apps/argocd-applicationset-controller created
deployment.apps/argocd-dex-server created
deployment.apps/argocd-notifications-controller created
deployment.apps/argocd-redis created
deployment.apps/argocd-repo-server created
deployment.apps/argocd-server created
statefulset.apps/argocd-application-controller created
networkpolicy.networking.k8s.io/argocd-application-controller-network-policy created
networkpolicy.networking.k8s.io/argocd-applicationset-controller-network-policy created
networkpolicy.networking.k8s.io/argocd-dex-server-network-policy created
networkpolicy.networking.k8s.io/argocd-notifications-controller-network-policy created
networkpolicy.networking.k8s.io/argocd-redis-network-policy created
networkpolicy.networking.k8s.io/argocd-repo-server-network-policy created
networkpolicy.networking.k8s.io/argocd-server-network-policy created
> kubectl get pods -n argocd
NAME                               READY   STATUS    RESTARTS   AGE
argocd-application-controller-0     1/1     Running   0          89s
argocd-applicationset-controller-9c64fc489-767w9 1/1     Running   0          89s
argocd-dex-server-5845b5c459-lprl4   1/1     Running   0          89s
argocd-notifications-controller-5c967844dc-kwgkk 1/1     Running   0          89s
argocd-redis-7b59bfff-5jbd8       1/1     Running   0          89s
argocd-repo-server-79566f874-p8wtd 1/1     Running   0          89s
argocd-server-5d5d5d9bbb-gfmvk     1/1     Running   0          89s
~ > % 20:27:15
```

Instalar ArgoCD

- Una vez que los pods están Running, podemos acceder a la UI con el usuario **admin**, pero, hay que ejecutar este comando para obtener el password:
- `kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath=".data.password" | base64 -d`



```
josediaz@MacBook-Pro-de-Jose:~
> kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath=".data.password" | base64 -d
F9NpRsoZgzz8CGVX%
```

Instalar ArgoCD

- Para acceder a la UI de ArgoCD ejecuta este comando:
- `kubectl port-forward svc/argocd-server -n argocd 8080:443`

```
> kubectl port-forward svc/argocd-server -n argocd 8080:443
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
```

Applications Tiles - Argo CD

Not Secure | https://localhost:8080/login?return_url=https%3A%2F%2Flocalhost%3A8080%2Fapplications

argohub

Username
admin

Password
.....

SIGN IN

Let's get stuff deployed!

The image shows a web browser window for the Argo CD application. The title bar says "Applications Tiles - Argo CD". The address bar shows a warning: "Not Secure | https://localhost:8080/login?return_url=https%3A%2F%2Flocalhost%3A8080%2Fapplications". The main content area has a dark blue background with a white star pattern. In the center is a large, friendly-looking orange cartoon octopus with big white eyes and a wide smile, standing on top of a large grey gear. To the right of the octopus is a login form with the "argohub" logo at the top. The form includes fields for "Username" (set to "admin") and "Password" (represented by a series of dots). A "SIGN IN" button is located below the password field. At the bottom left of the main area, the text "Let's get stuff deployed!" is displayed. The bottom right corner of the main area has the word "argo" in a smaller font.

Applications Tiles - Argo CD

Not Secure | https://localhost:8080/applications

Argo CD v2.6.0+acc554f

APPLICATIONS TILES

Log out

Applications

+ NEW APP SYNC APPS REFRESH APPS Search applications... /

No applications available to you just yet

Create new application to start managing resources in your cluster

CREATE APPLICATION

Applications

Settings

User Info

Documentation

←



Crear namespace app

- kubectl create ns app

```
Last login: Sun Feb 26 20:13:21 on ttys000
> kubectl create ns app
namespace/app created
~ > █
```

Pasos para crear una aplicación

- Forkear proyecto: <https://github.com/joedayz/rsvpapp-helm-cicd>
- Login a ArgoCD
- Clic en new app y actualizar con lo siguiente
- Dar como nombre de aplicación: rsvapp
- Seleccionar el proyecto. Por ahora escoger el nombre default, el cual es automáticamente presentado después de la instalación de ArgoCD
- Establecer SyncPolicy Automatic.
- Habilitar la opción Prune Resources
- Habilitar SELF HEAL

[CREATE](#)[CANCEL](#)

X

GENERAL

[EDIT AS YAML](#)

Application Name

rsvpapp

Project Name

default

SYNC POLICY

Automatic

▼

- PRUNE RESOURCES ⓘ
- SELF HEAL ⓘ

 SET DELETION FINALIZER ⓘ

SYNC OPTIONS

- SKIP SCHEMA VALIDATION
- PRUNE LAST
- RESPECT IGNORE DIFFERENCES

- AUTO-CREATE NAMESPACE
- APPLY OUT OF SYNC ONLY
- SERVER-SIDE APPLY

PRUNE PROPAGATION POLICY: foreground

▼

- REPLACE ⚠
- RETRY

Pasos para crear una aplicación

- Definir el repositorio del URL con la demo que se ha forkeado <https://github.com/joedayz/rsvapp-helm-cicd.git>
- Establecer revision **HEAD**
- El path es la ubicación del archivo de configuración. Escoger **.**
- Destination: Es el cluster donde se va a desplegar la aplicación.
- Selecciona el Cluster: <https://kubernetes.default.svc>.
- Selecciona el namespace: **app**
- Clic en **create**.

SOURCE

Repository URL

<https://github.com/joedayz/rsvpapp-helm-cicd.git>

GIT ▾

Revision

HEAD

Branches ▾



Path

.

DESTINATION

Cluster URL

<https://kubernetes.default.svc>

URL ▾

Namespace

app



rsvpapp



Project: default

Labels:

Status: ❤️ Healthy ✓ Synced

Repository: <https://github.com/joedayz/rsvpapp-helm-cicd.git>

Target Revi... HEAD

Path: -

Destination: in-cluster

Namespace: app

Created At: 02/26/2023 21:31:28 (a minute ago)

SYNC

REFRESH

DELETE

kubectl get pods,svc -n app

› kubectl get pods,svc -n app

NAME	READY	STATUS	RESTARTS	AGE
pod/rsvpapp-7456cb95f6-6lctz	1/1	Running	0	108s
pod/rsvpapp-7456cb95f6-cttr2	1/1	Running	0	108s
pod/rsvpapp-7456cb95f6-zd8cp	1/1	Running	0	108s
pod/rsvpapp-mongodb-546778fdb8-wnt7c	1/1	Running	0	108s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/rsvpapp	ClusterIP	10.108.65.29	<none>	80/TCP	108s
service/rsvpapp-mongodb	ClusterIP	10.108.47.98	<none>	27017/TCP	108s

~ › █

[APP DETAILS](#) [APP DIFF](#) [SYNC](#) [SYNC STATUS](#) [HISTORY AND ROLLBACK](#) [DELETE](#) [REFRESH](#) [Log out](#)APP HEALTH **Healthy**CURRENT SYNC STATUS **Synced**

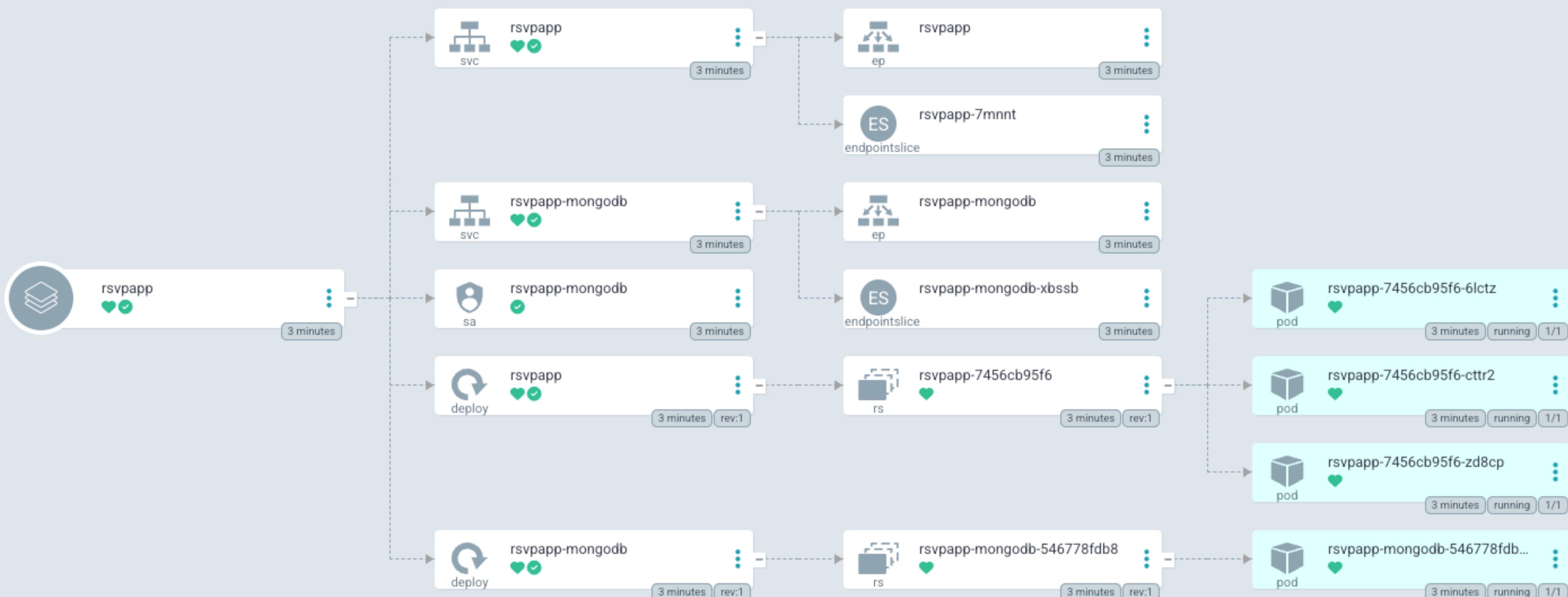
To HEAD (9c23383)

Author: Neependra Khare <neependra.khare@gmail.com> -
Comment: Update Chart.yaml
Auto sync is enabled.LAST SYNC RESULT **Sync OK**

To 9c23383

Succeeded 3 minutes ago (Sun Feb 26 2023 21:31:30 GMT-0500)
Author: Neependra Khare <neependra.khare@gmail.com> -
Comment: Update Chart.yaml

100%

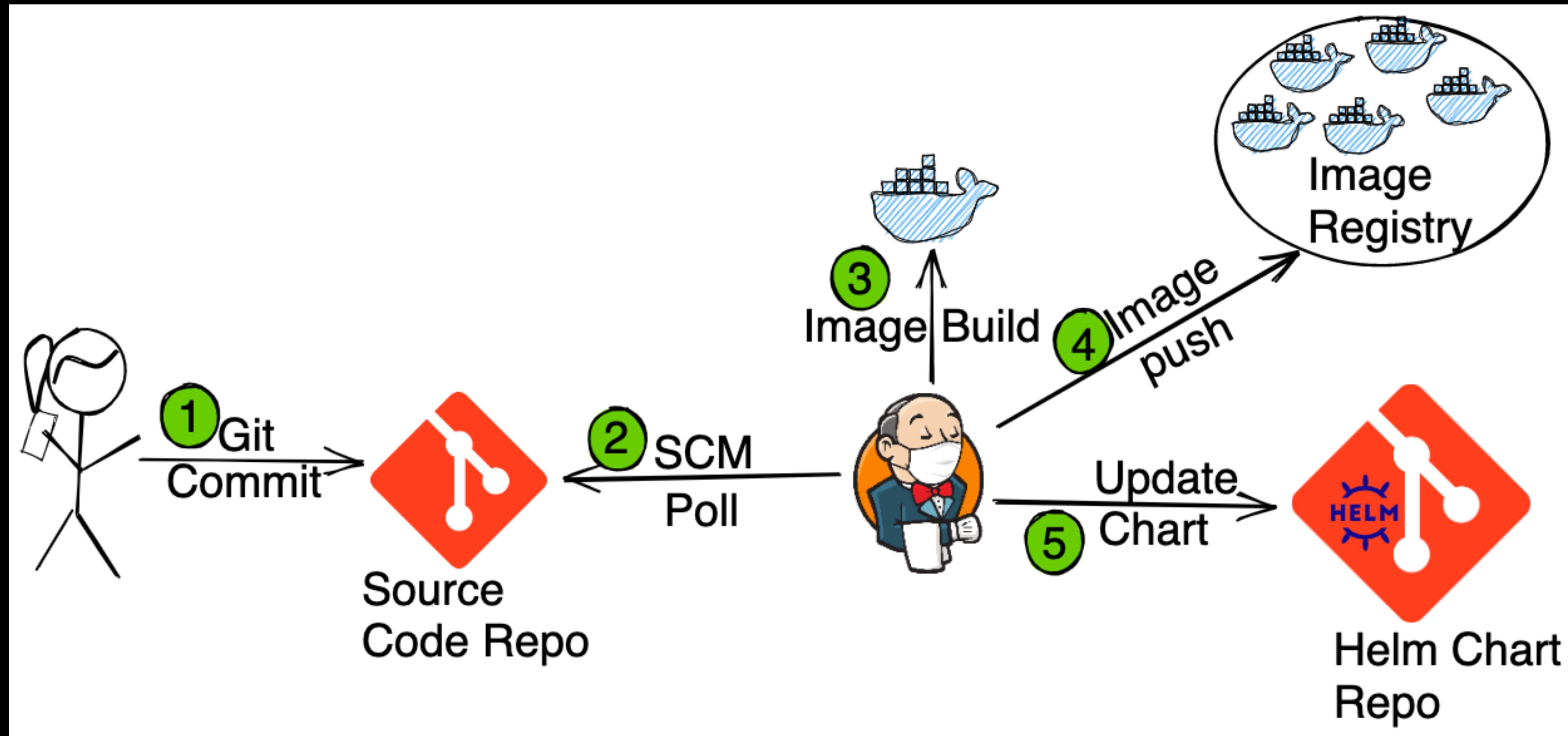


Crear el ingress para nuestro deployment

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: rsvp-ingress
spec:
  rules:
  - http:
    paths:
    - path: /
      pathType: Prefix
    backend:
      service:
        name: rsvpapp
        port:
          number: 80
```

```
> vi rsvp-ingress.yaml
> kubectl apply -f rsvp-ingress.yaml -n app
ingress.networking.k8s.io/rsvp-ingress created
~/Downloads > █
```

Configurar CI con Jenkins



Instalar jenkins

Jenkins-install.sh

```
FROM nginx:1.11-alpine ● | rsvp-ingress.yaml ✘ jenkins-install.sh ● | argo-install.yaml ✘

curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
chmod 700 get_helm.sh
./get_helm.sh
sleep 3
echo "installed helm"

helm repo add jenkins https://charts.jenkins.io

kubectl create ns jenkins

helm install jenkins \
--set controller.serviceType=NodePort,controller.image=teamcloudyuga/cyjenkins,controller.tag=2.289,controller.installPlugins=false,controller.nodePort=31000,persistence.enabled=false \
--namespace=jenkins \
jenkins/jenkins
echo "installed jenkins with helm "
```

Instalar jenkins

Jenkins-install.sh

```
> chmod +x jenkins-install.sh
> sh jenkins-install.sh
Helm v3.11.1 is already latest
installed helm
"jenkins" already exists with the same configuration, skipping
namespace/jenkins created
NAME: jenkins
LAST DEPLOYED: Sun Feb 26 21:42:09 2023
NAMESPACE: jenkins
STATUS: deployed
REVISION: 1
NOTES:
1. Get your 'admin' user password by running:
   kubectl exec --namespace jenkins -it svc/jenkins -c jenkins -- /bin/cat /run/secrets/additional/cha
rt-admin-password && echo
2. Get the Jenkins URL to visit by running these commands in the same shell:
   export NODE_PORT=$(kubectl get --namespace jenkins -o jsonpath=".spec.ports[0].nodePort" services
jenkins)
   export NODE_IP=$(kubectl get nodes --namespace jenkins -o jsonpath=".items[0].status.addresses[0].
address")
   echo http://$NODE_IP:$NODE_PORT
3. Login with the password from step 1 and the username: admin
4. Configure security realm and authorization strategy
5. Use Jenkins Configuration as Code by specifying configScripts in your values.yaml file, see docume
ntation: http://$NODE_IP:$NODE_PORT/configuration-as-code and examples: https://github.com/jenkinsci/
configuration-as-code-plugin/tree/master/demos
```

For more information on running Jenkins on Kubernetes, visit:
<https://cloud.google.com/solutions/jenkins-on-container-engine>

For more information about Jenkins Configuration as Code, visit:
<https://jenkins.io/projects/jcasc/>

```
#####
##### WARNING: Persistence is disabled!!! You will lose your data when #####
#####           the Jenkins pod is terminated. #####
#####
installed jenkins with helm
~/Downloads > |
```

Instalar jenkins

kubectl get pods -n jenkins

```
› kubectl get pods -n jenkins
NAME        READY   STATUS    RESTARTS   AGE
jenkins-0   2/2     Running   0          117s
~/Downloads › █
```

Instalar jenkins

Obtener el password del admin

- kubectl exec --namespace jenkins -it svc/jenkins -c jenkins -- /bin/cat /run/secrets/additional/chart-admin-password && echo

```
> kubectl exec --namespace jenkins -it svc/jenkins -c jenkins -- /bin/cat /run/secrets/additional/chart-admin-password && echo
```

```
OLMfcKjiug4I1esGbaJwQX  
~/Downloads > █
```

21:47:13

Instalar jenkins

Lanzar la UI de Jenkins

- minikube service jenkins -n jenkins

```
> minikube service jenkins -n jenkins
|-----|-----|-----|-----|
| NAMESPACE | NAME | TARGET PORT | URL |
|-----|-----|-----|-----|
| jenkins | jenkins | http/8080 | http://192.168.49.2:31000 |
|-----|-----|-----|-----|
🏃 Starting tunnel for service jenkins.
|-----|-----|-----|-----|
| NAMESPACE | NAME | TARGET PORT | URL |
|-----|-----|-----|-----|
| jenkins | jenkins | | http://127.0.0.1:61571 |
|-----|-----|-----|-----|
🎉 Opening service jenkins/jenkins in default browser...
❗ Because you are using a Docker driver on darwin, the terminal needs to be open to run it.
```

Sign in [Jenkins] X

127.0.0.1:61571/login?from=%2F ★



Welcome to Jenkins!

Username

Password

Sign in

Keep me signed in

Dashboard [Jenkins] X

127.0.0.1:61571

search ? 2 2 Jenkins Admin log out

Jenkins

Dashboard

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- New View

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job →

Build Queue ^
No builds in the queue.

Build Executor Status ^

REST API Jenkins 2.289.1