

Desafío 14

Para este desafío número 14 los requisitos son los mismos que el número 13, tener Instalado **Minikube** y **kubectl** para la administración de los cluster.

```
Windows PowerShell
PS C:\Users\Michael> kubectl version
Client Version: v1.30.2
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
Unable to connect to the server: dial tcp 127.0.0.1:54239: connect: No connection could be made because the target machine actively refused it.
PS C:\Users\Michael> minikube version
minikube version: v1.34.0
commit: 210b148df93a80eb872ecbeb7e35281b3c582c61
PS C:\Users\Michael> |
```

Seguido de eso, desplegamos los componentes de **ArgoCD**, eso lo haremos en la consola de **Powershell** con el siguiente comando **kubectl apply -n argocd -f** <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>

```
PS C:\Users\Michael> kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
customresourcedefinition.apiextensions.k8s.io/applications.argoproj.io unchanged
customresourcedefinition.apiextensions.k8s.io/applicationsets.argoproj.io unchanged
customresourcedefinition.apiextensions.k8s.io/appprojects.argoproj.io unchanged
serviceaccount/argocd-application-controller unchanged
serviceaccount/argocd-applicationset-controller unchanged
serviceaccount/argocd-dex-server unchanged
serviceaccount/argocd-notifications-controller unchanged
serviceaccount/argocd-redis unchanged
serviceaccount/argocd-repo-server unchanged
serviceaccount/argocd-server unchanged
role.rbac.authorization.k8s.io/argocd-application-controller unchanged
role.rbac.authorization.k8s.io/argocd-applicationset-controller unchanged
role.rbac.authorization.k8s.io/argocd-dex-server unchanged
role.rbac.authorization.k8s.io/argocd-notifications-controller unchanged
role.rbac.authorization.k8s.io/argocd-redis unchanged
role.rbac.authorization.k8s.io/argocd-server unchanged
clusterrole.rbac.authorization.k8s.io/argocd-application-controller unchanged
clusterrole.rbac.authorization.k8s.io/argocd-applicationset-controller unchanged
```

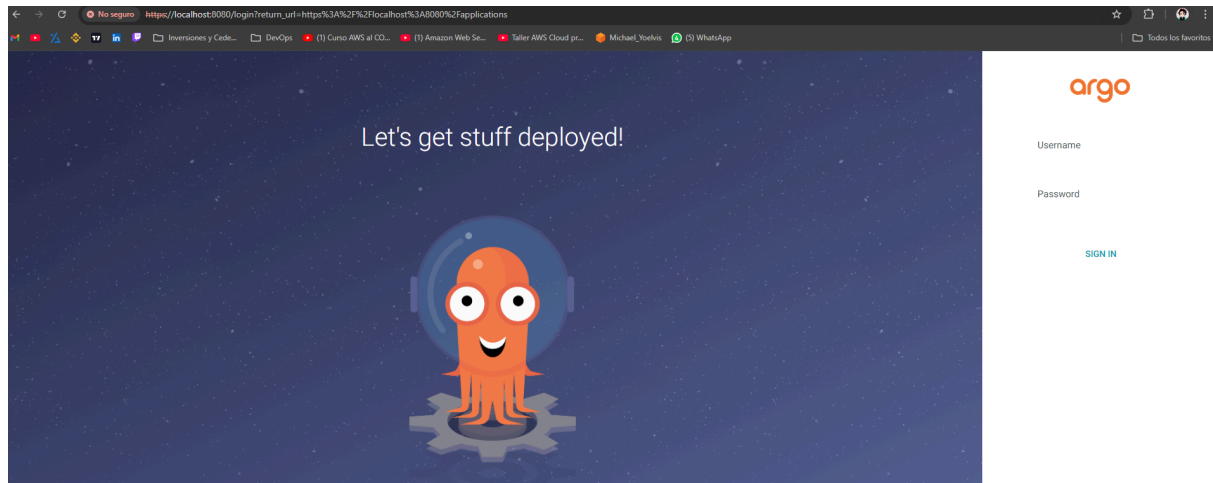
Seguido de estos pasos debemos Obtener la contraseña del usuario admin para acceder al dashboard de **ArgoCD**, eso lo hacemos con el comando **kubectl get secret argocd-initial-admin-secret -n argocd -o jsonpath="{.data.password}" | %{{System.Text.Encoding}::UTF8.GetString([System.Convert]::FromBase64String(\$_))}**

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

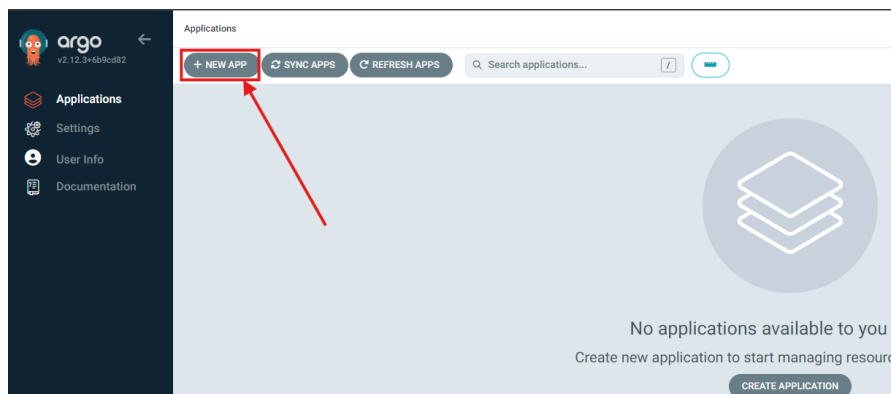
Luego exponemos el servicio de **ArgoCD** con el comando **kubectl port-forward svc/argocd-server -n argocd 8080:443**

```
PS C:\Users\Michael> kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
customresourcedefinition.apiextensions.k8s.io/applications.argoproj.io unchanged
customresourcedefinition.apiextensions.k8s.io/applicationsets.argoproj.io unchanged
customresourcedefinition.apiextensions.k8s.io/appprojects.argoproj.io unchanged
serviceaccount/argocd-application-controller unchanged
serviceaccount/argocd-applicationset-controller unchanged
serviceaccount/argocd-dex-server unchanged
serviceaccount/argocd-notifications-controller unchanged
serviceaccount/argocd-redis unchanged
serviceaccount/argocd-repo-server unchanged
serviceaccount/argocd-server unchanged
role.rbac.authorization.k8s.io/argocd-application-controller unchanged
role.rbac.authorization.k8s.io/argocd-applicationset-controller unchanged
role.rbac.authorization.k8s.io/argocd-dex-server unchanged
role.rbac.authorization.k8s.io/argocd-notifications-controller unchanged
role.rbac.authorization.k8s.io/argocd-redis unchanged
role.rbac.authorization.k8s.io/argocd-server unchanged
clusterrole.rbac.authorization.k8s.io/argocd-application-controller unchanged
clusterrole.rbac.authorization.k8s.io/argocd-applicationset-controller unchanged
```

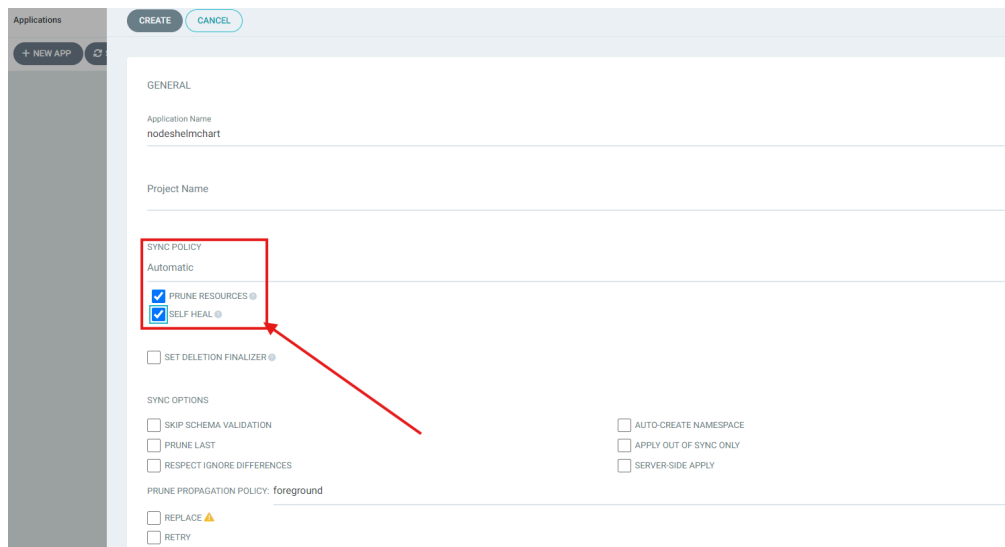
Desde el navegador nos dirigimos a nuestra dirección ip con el puerto 8080
<https://localhost:8080/> Usuario Admin y la clave la que en el paso anterior generamos.



Una vez dentro del **Dashboard** debemos crear nuestra App Seleccionamos New App



Seguimos con el completado de los parámetros solicitados, en **Sync Policy** tildar las 2 opciones para que así pueda actualizar el repositorio en base a nuestras actualizaciones



Debemos configurar que **ArgoCD** tenga acceso a nuestro repositorio de **Git** que contiene el **Deployment** a desplegar.

SOURCE

Repository URL

https://github.com/Kidbuut/Desafio-12

GIT ▼

Revision

HEAD

Branches ▼

Path

https://github.com/Kidbuut/Desafio-12/tree/main/App

Seguimos con mas llenado de parámetros

DESTINATION

Cluster URL

https://kubernetes.default.svc

URL ▼

Namespace

Default

Helm ▼

HELM

VALUES FILES

VALUES

PARAMETERS

autoscaling.enabled	false
autoscaling.maxReplicas	100
autoscaling.minReplicas	1
autoscaling.targetCPUUtilizationPercentage	80
env.MONGO_DB_NAME	test
env.MONGO_DB_PASS	mongo123
env.MONGO_DB_URI	mongodb://mongo:mongo123@mongodb-service:27017/test
env.MONGO_DB_USER	mongo
ingress.enabled	false
ingress.hosts[0].host	chart-example.local
mongodb.dbName	test

Al terminar la configuración y crear la App veremos cómo se despliegan nuestros nuestro deploy de **helmchart** y **nodejsapp**

