

Hernan Andres Acosta

Desafío 14: Despliegue Automatizado con ArgoCD

Fecha de entrega: 12/05/2025

Objetivo

Implementar un despliegue automatizado utilizando ArgoCD para la aplicación desarrollada previamente (Desafío 12), que incluye una app NestJS y una base de datos MongoDB, empaquetada como un Helm Chart. Se busca aplicar los principios de GitOps para gestionar de forma eficiente los cambios desde un repositorio Git.

1. Instalación de ArgoCD en Minikube

Paso 1: Crear un cluster local (si no se dispone de uno)

```
minikube start
```

Paso 2: Instalar ArgoCD

```
kubectl create namespace argocd  
kubectl apply -n argocd -f  
https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
```

Paso 3: Exponer el dashboard de ArgoCD

```
kubectl port-forward svc/argocd-server -n argocd 8080:443
```

Acceder vía navegador: <https://localhost:8080>

Paso 4: Obtener la contraseña del admin

```
kubectl get secret argocd-initial-admin-secret -n argocd -o  
jsonpath="{.data.password}" | base64 -d
```

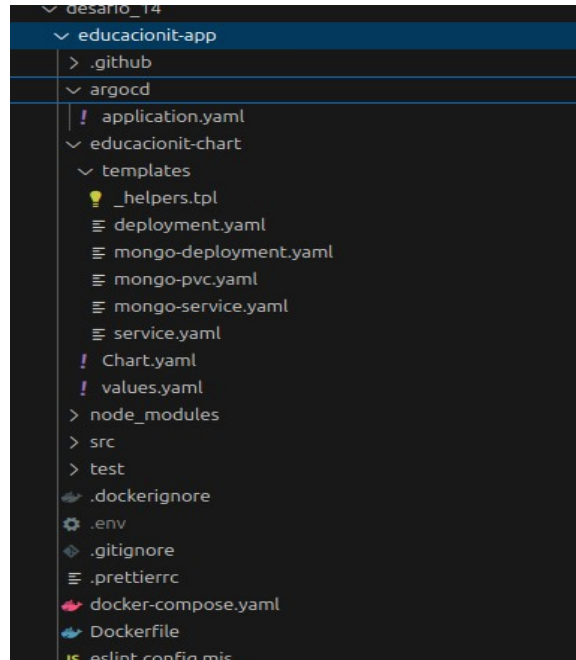
2. Preparación del Repositorio Git

Se utilizó el repositorio:

<https://github.com/hernan130/educacionit-app.git>

Allí se agregó la carpeta `educacionit-chart/` que contiene el Helm Chart de la app + MongoDB desarrollado en el Desafío 12.

Estructura:



3. Creación del Recurso Application en ArgoCD

Creamos el siguiente manifiesto YAML (`application.yaml`) para registrar la aplicación en ArgoCD:

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: educacionit-app
  namespace: argocd
spec:
  project: default
  source:
    repoURL: https://github.com/hernan130/educacionit-app.git
    targetRevision: main
    path: educacionit-chart
    helm:
      values: values.yaml
  destination:
    server: https://kubernetes.default.svc
    namespace: educacionit-chart
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
    syncOptions:
      - CreateNamespace=true
```

Aplicar el manifiesto

```
kubectl apply -f application.yaml
```

4. Validación del Despliegue

Verificamos que los recursos se hayan creado correctamente:

```
kubectl get all -n educacionit-chart
```

Resultado esperado:

- Pods de la app y MongoDB corriendo
- Services expuestos correctamente
- PVC para persistencia de MongoDB creado

```
herman@andres: ~/Documentos/Educacion_It/clase52/... x herman@andres: ~/Documentos/Educacion_It/clase52/... x herman@andres: ~/Documentos/Educacion_It/clase52/... x
herman@andres:~/Documentos/Educacion_It/clase52/desafio_14/educacionit-app$ kubectl get all -n educacionit-chart
NAME                                READY   STATUS    RESTARTS   AGE
pod/educacionit-app-cfd7b89f-sjrsj  1/1     Running   0           24m
pod/mongo-7dc6f78c88-7d9tv         1/1     Running   0           24m

NAME                                TYPE               CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/educacionit-app             NodePort           10.109.190.33   <none>           3000:30081/TCP   24m
service/mongo                       ClusterIP          10.108.233.62   <none>           27017/TCP        24m

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/educacionit-app      1/1     1             1           24m
deployment.apps/mongo                1/1     1             1           24m

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/educacionit-app-cfd7b89f  1         1         1       24m
replicaset.apps/mongo-7dc6f78c88         1         1         1       24m
```

Desde la interfaz web de ArgoCD:

- Estado: **Synced**
- Salud: **Healthy**
- Sin errores en la sección de logs o eventos

The screenshot shows the ArgoCD web interface. On the left is a dark sidebar with navigation links: Applications, Settings, User Info, and Documentation. Below these are application filters, including 'Favorites Only' and a 'SYNC STATUS' section with checkboxes for 'Unknown' (0), 'Synced' (1), and 'OutOfSync' (0). The main panel displays the 'educacionit-app' application details. It shows the project as 'default', labels, a 'Healthy' status with a green heart icon, and a 'Synced' status with a green checkmark icon. The repository is 'https://github.com/herman130/educacionit-app.git', the target revision is 'main', and the path is 'educacionit-chart'. The destination is 'in-cluster' and the namespace is 'educacionit-chart'. The application was created at '05/10/2025 12:46:47 (25 minutes ago)' and last synced at '05/10/2025 12:57:52 (14 minutes ago)'. At the bottom of the details panel are buttons for 'SYNC', 'C' (cancel), and a refresh icon.

5. Comandos Clave Utilizados

```
# Instalar el chart manualmente (verificación previa)
helm install educacionit educacionit-chart --namespace educacionit-chart --
create-namespace
```

```
# Comprobar recursos en el namespace
kubectl get all -n educacionit-chart
```

```
# Eliminar release si es necesario para testeo
helm uninstall educacionit -n educacionit-chart
```



Automatización de Despliegue

Gracias a la configuración de `syncPolicy.automated`, cualquier cambio en `values.yaml` o manifiestos dentro del chart se refleja automáticamente en el cluster al hacer `git push`.

Esto cumple con los principios de GitOps: el repositorio es la fuente de la verdad.

```
24
25   persistence:
26     enabled: true
27     size: 1Gi
28     storageClass: "" # usar el default de Minikube u otro
29
30   service:
31     type: NodePort
32     nodePort: 30081
```

You, 20 minutes ago • Cambio de puerto nodePort

```
24
25   persistence:
26     enabled: true
27     size: 1Gi
28     storageClass: "" # usar el default de Minikube u otro
29
30   service:
31     type: NodePort
32     nodePort: 30080
```

You, 1 second ago • Uncommitted changes

```

herman@andres:~/Documentos/Educacion_It/clase52/desafio_14/educacionit-app$ git add .
herman@andres:~/Documentos/Educacion_It/clase52/desafio_14/educacionit-app$ git commit -m "Cambio de puerto nodePort a 30080"
[main d8879be] Cambio de puerto nodePort a 30080
1 file changed, 1 insertion(+), 1 deletion(-)
herman@andres:~/Documentos/Educacion_It/clase52/desafio_14/educacionit-app$ git push origin main
Enumerando objetos: 7, listo.
Contando objetos: 100% (7/7), listo.
Compresión delta usando hasta 4 hilos.
Comprimiendo objetos: 100% (4/4), listo.
Escribiendo objetos: 100% (4/4), 358 bytes | 358.00 KiB/s, listo.
Total 4 (delta 3), reusados 0 (delta 0), pack-reusados 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/herman130/educacionit-app.git
2ca43e9..d8879be main -> main

```

```

herman@andres:~/Documentos/Educacion_It/clase52/desafio_14/educacionit-app$ kubectl get all -n educacionit-chart
NAME                                READY    STATUS    RESTARTS   AGE
pod/educacionit-app-cfd7b89f-sjrsj  1/1     Running   0           34m
pod/mongo-7dc6f78c88-7d9tv         1/1     Running   0           34m

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
service/educacionit-app             NodePort      10.109.190.33 <none>         3000:30080/TCP   34m
service/mongo                       ClusterIP     10.108.233.62 <none>         27017/TCP        34m

NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/educacionit-app      1/1     1             1            34m
deployment.apps/mongo               1/1     1             1            34m

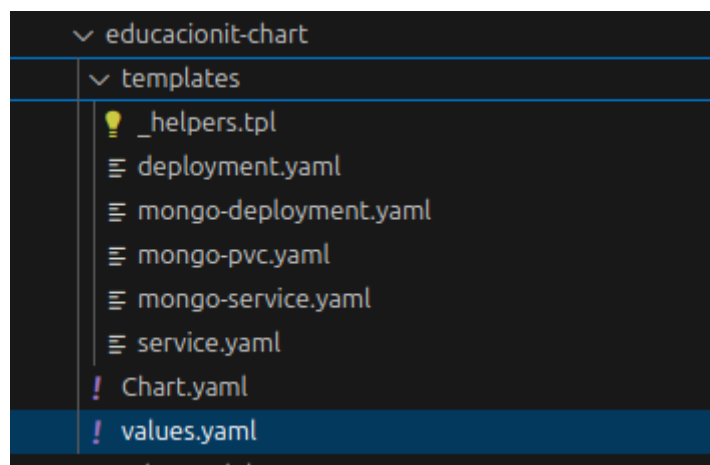
NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/educacionit-app-cfd7b89f  1          1          1        34m
replicaset.apps/mongo-7dc6f78c88          1          1          1        34m
herman@andres:~/Documentos/Educacion_It/clase52/desafio_14/educacionit-app$

```



Entregables

- Repositorio Git: <https://github.com/herman130/educacionit-app.git>
- Carpeta del Chart: educacionit-chart



- Recurso ArgoCD:
- application.yaml

```

1  apiVersion: argoproj.io/v1alpha1
2  kind: Application
3  metadata:
4    name: educacionit-app
5    namespace: argocd
6  spec:
7    project: default
8    source:
9      repoURL: https://github.com/hernan130/educacionit-a
10     targetRevision: main
11     path: educacionit-chart
12   destination:
13     server: https://kubernetes.default.svc
14     namespace: educacionit-chart
15   syncPolicy:
16     automated:
17       prune: true
18       selfHeal: true
19     syncOptions:
20       - CreateNamespace=true

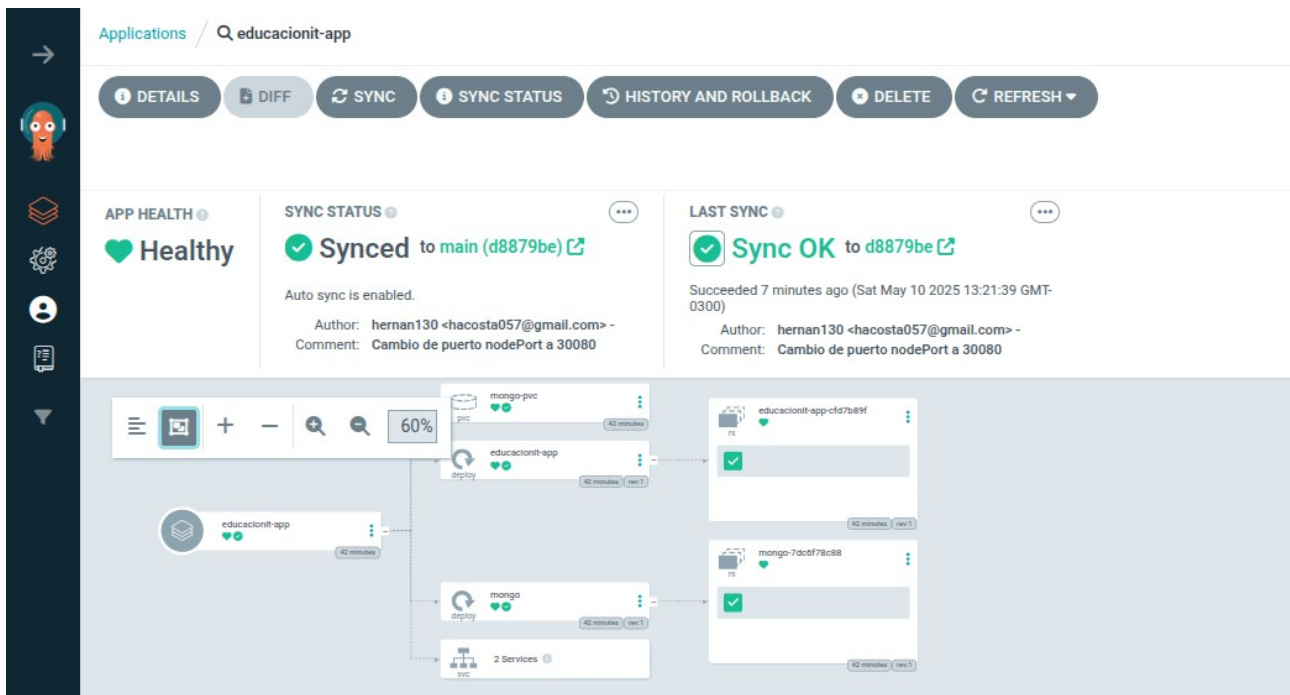
```

- Evidencia: pods corriendo,

pods corriendo

Nombre	Imágenes	Etiquetas	Nodo	Estado	Reinicios	Utilización de CPU (núcleos)	Utilización de memoria (octetos)	Fecha de creación
educacionit-app-cfd7b89f-sjrsj	herman1305/educacionit-app:latest	app: educacionit-app pod-template-hash: cfd7b89f	minikube	Running	0	-	-	40 m ago
mongo-7dc6f78c88-7d9tv	mongo:7.0	app: mongo pod-template-hash: 7dc6f78c88	minikube	Running	0	-	-	40 m ago

estado synced/healthy en ArgoCD,



✓ Para crear directamente la Aplicación desde la UI de ArgoCD

1. Accedé al panel web

Ingresá a <https://localhost:8080> (o la URL donde tengas expuesto ArgoCD).

2. Iniciá sesión

Usuario: admin

Contraseña: obtenida con:

```
bash
CopiarEditar
kubectl get secret argocd-initial-admin-secret -n argocd -o
jsonpath="{.data.password}" | base64 -d
```

3. Hací clic en NEW APP (arriba a la derecha)



4. Completá el formulario con los siguientes datos:

Campo	Valor
Application Name	educacionit-app
Project	default
Sync Policy	✓ Marcar "Auto-Sync" y "Self Heal"
Repository URL	https://github.com/hernan130/educacionit-app.git
Revision	main
Path	educacionit-chart (ruta donde está el Chart en tu repo)
Cluster	https://kubernetes.default.svc

Campo	Valor
Namespace	educacionit-chart (se creará automáticamente si marcás esa opción)
5. Avanzá con "Create"	

Una vez creada, ArgoCD detectará automáticamente los recursos del Helm Chart y los aplicará al clúster.

Verificá que aparezca:

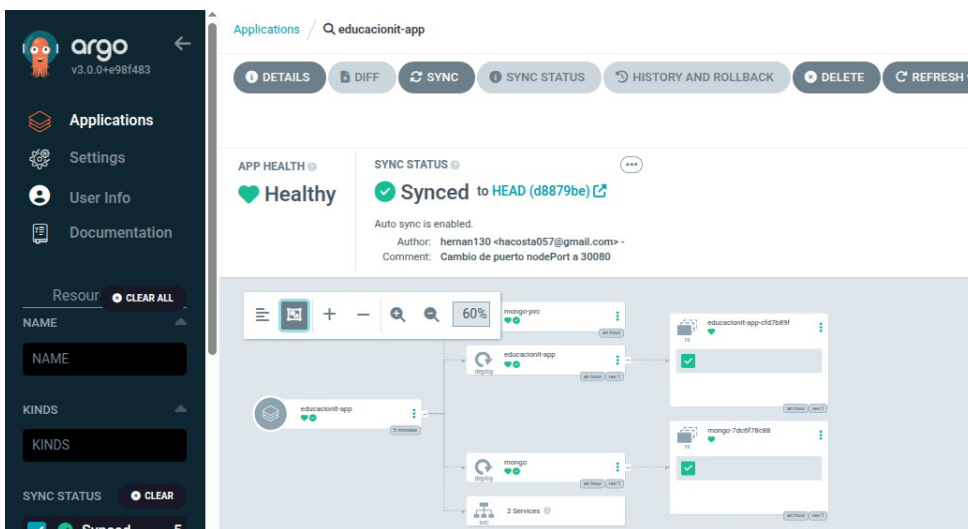
- Estado:  **Synced**
- Salud:  **Healthy**
- Todos los recursos desplegados en el namespace `educacionit-chart`

❌ Eliminar y recrear la app

Si querés hacer una creación limpia desde cero:

`kubectl delete application educacionit-app -n argocd` (desde la terminal)

Después, volvé a la UI y creala desde **NEW APP** como te indiqué anteriormente.



Conclusión Final

A través de este desafío, se logró implementar con éxito un flujo de despliegue automatizado utilizando ArgoCD y Helm sobre un clúster local de Minikube, aplicando los principios de GitOps. Esto permitió mantener una sincronización constante entre el estado deseado (almacenado en el repositorio Git) y el estado actual del clúster.

Este enfoque representa una práctica moderna y escalable de gestión de aplicaciones en Kubernetes, alineada con los estándares actuales del rol de DevOps Engineer.