SOMAIYA VIDYAVIHAR UNIVERSITY

Somaiya TRUST

K J Somaiya School of Engineering

Department of Computer Engineering
Course: DevOps

IA – 1: Case Study

Tool: **ArgoCD**

Name: Aditya Kothari

Batch: C-3

Roll no.: 16010122329

Date: 29/09/2025

1. INTRODUCTION

ArgoCD is a Kubernetes-native continuous delivery tool that implements GitOps, using Git repositories as the source of truth. Unlike traditional pipelines, ArgoCD continuously monitors Git for changes and automatically applies updates to the cluster.

2. Problem Statement

Manual kubectl apply workflows are error-prone and lack visibility. We need:

- A declarative approach driven by Git
- Automated synchronization to keep clusters in desired state
- Real-time dashboard for monitoring and control

3. Objectives

- Provision a local Kubernetes environment (Minikube)
- Install ArgoCD with kubectl
- Deploy an NGINX sample app from Git
- Demonstrate automatic sync on Git commits
- Demonstrate manual rollback via UI
- Compare ArgoCD to mainstream CI/CD tools

- Discuss advantages and limitations

4. Implementation and Demonstration

4.1 Minikube Setup

1. minikube start --driver=docker

   Starts a single-node Kubernetes cluster in Docker.

2. minikube status

   Verifies control plane, kubelet, and kubeconfig status.

3. kubectl get nodes

   Lists node(s) showing READY status and Kubernetes version.

4. kubectl get pods -A

   Displays all pods across namespaces to confirm cluster health.

4.2 ArgoCD Installation

1. kubectl create namespace argocd

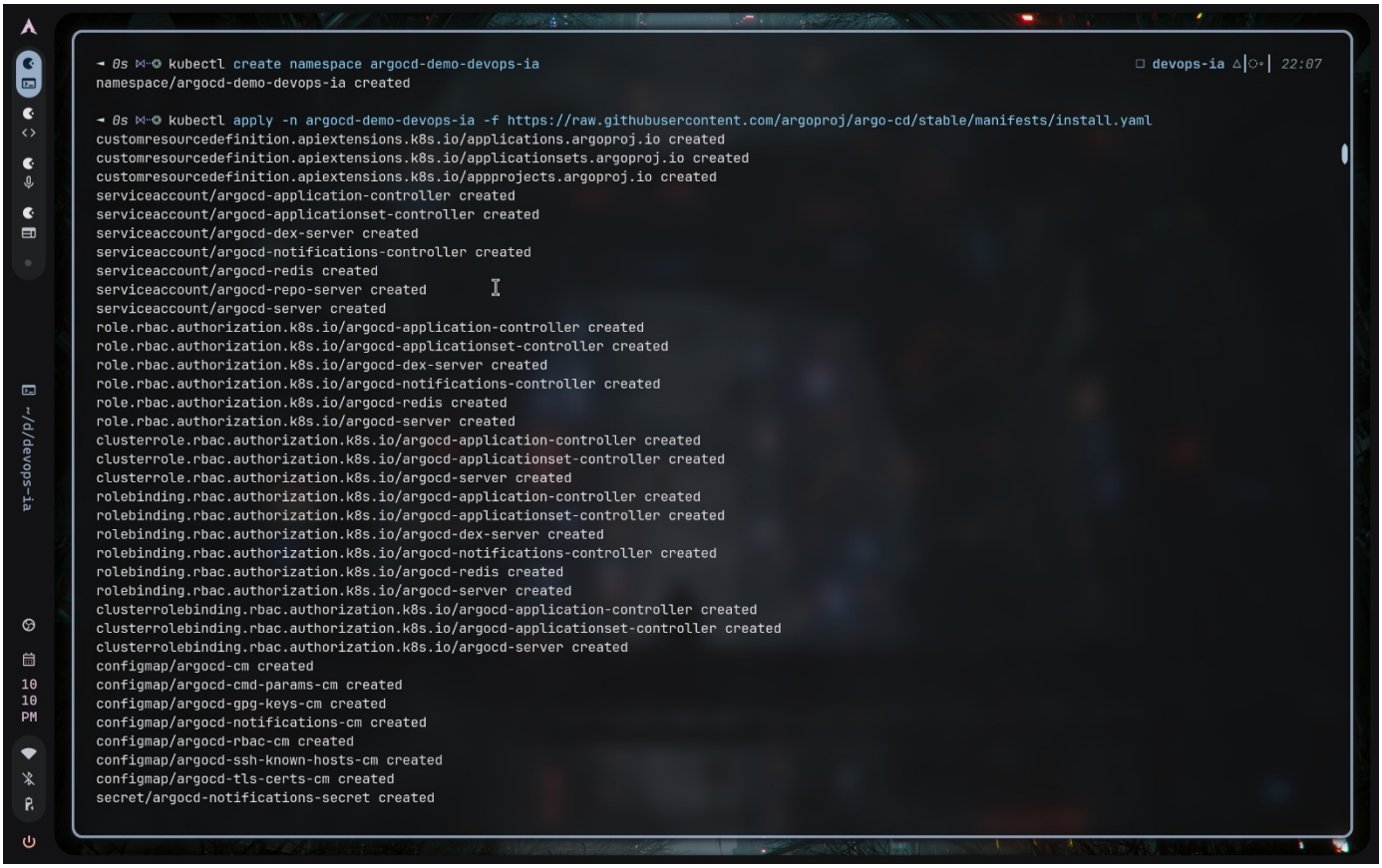
   Creates the ‘argocd’ namespace.

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

   Installs ArgoCD CRDs, deployments, StatefulSet, and services.

3. kubectl get pods -n argocd

   Checks that argocd-server, repo-server, application-controller, etc., are Running.



```
◄ 0s ⋈-⊙ kubectl create namespace argocd-demo-devops-ia                    □ devops-ia △|○·| 22:07
namespace/argocd-demo-devops-ia created

◄ 0s ⋈-⊙ kubectl apply -n argocd-demo-devops-ia -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
customresourcedefinition.apiextensions.k8s.io/applications.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/applicationsets.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/appprojects.argoproj.io created
serviceaccount/argocd-application-controller created
serviceaccount/argocd-applicationset-controller created
serviceaccount/argocd-dex-server created
serviceaccount/argocd-notifications-controller created
serviceaccount/argocd-redis created
serviceaccount/argocd-repo-server created
serviceaccount/argocd-server created
role.rbac.authorization.k8s.io/argocd-application-controller created
role.rbac.authorization.k8s.io/argocd-applicationset-controller created
role.rbac.authorization.k8s.io/argocd-dex-server created
role.rbac.authorization.k8s.io/argocd-notifications-controller created
role.rbac.authorization.k8s.io/argocd-redis created
role.rbac.authorization.k8s.io/argocd-server created
clusterrole.rbac.authorization.k8s.io/argocd-application-controller created
clusterrole.rbac.authorization.k8s.io/argocd-applicationset-controller created
clusterrole.rbac.authorization.k8s.io/argocd-server created
rolebinding.rbac.authorization.k8s.io/argocd-application-controller created
rolebinding.rbac.authorization.k8s.io/argocd-applicationset-controller created
rolebinding.rbac.authorization.k8s.io/argocd-dex-server created
rolebinding.rbac.authorization.k8s.io/argocd-notifications-controller created
rolebinding.rbac.authorization.k8s.io/argocd-redis created
rolebinding.rbac.authorization.k8s.io/argocd-server created
clusterrolebinding.rbac.authorization.k8s.io/argocd-application-controller created
clusterrolebinding.rbac.authorization.k8s.io/argocd-applicationset-controller created
clusterrolebinding.rbac.authorization.k8s.io/argocd-server created
configmap/argocd-cm created
configmap/argocd-cmd-params-cm created
configmap/argocd-gpg-keys-cm created
configmap/argocd-notifications-cm created
configmap/argocd-rbac-cm created
configmap/argocd-ssh-known-hosts-cm created
configmap/argocd-tls-certs-cm created
secret/argocd-notifications-secret created
```

```
configmap/argocd-notifications-cm created
configmap/argocd-rbac-cm created
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

◄ 1s ⋈·⊙ kubectl -n argocd-demo-devops-ia get pods                    □ devops-ia △|○·| 22:09
NAME                                                  READY   STATUS              RESTARTS   AGE
argocd-application-controller-0                       0/1     ContainerCreating   0          25s
argocd-applicationset-controller-6d87c9f7b-x5hcz      0/1     ContainerCreating   0          25s
argocd-dex-server-69ccc5b979-j9pvz                    0/1     PodInitializing     0          25s
argocd-notifications-controller-545bddc587-9x9hh      0/1     ContainerCreating   0          25s
argocd-redis-7fb68457d5-hl92b                         0/1     Init:0/1            0          25s
argocd-repo-server-6dcd5dccdd-n5jrd                   0/1     Init:0/1            0          25s
argocd-server-7fc5bf6f76-ls8tp                        0/1     ContainerCreating   0          25s

◄ 0s ⋈·⊙ |                                                           □ devops-ia △|○·| 22:10
```

4.3 Accessing ArgoCD UI

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

   Forwards local port 8080 to ArgoCD server port 443.

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

   Retrieves the initial admin password.

3. argocd login localhost:8080 --username admin --password <password>

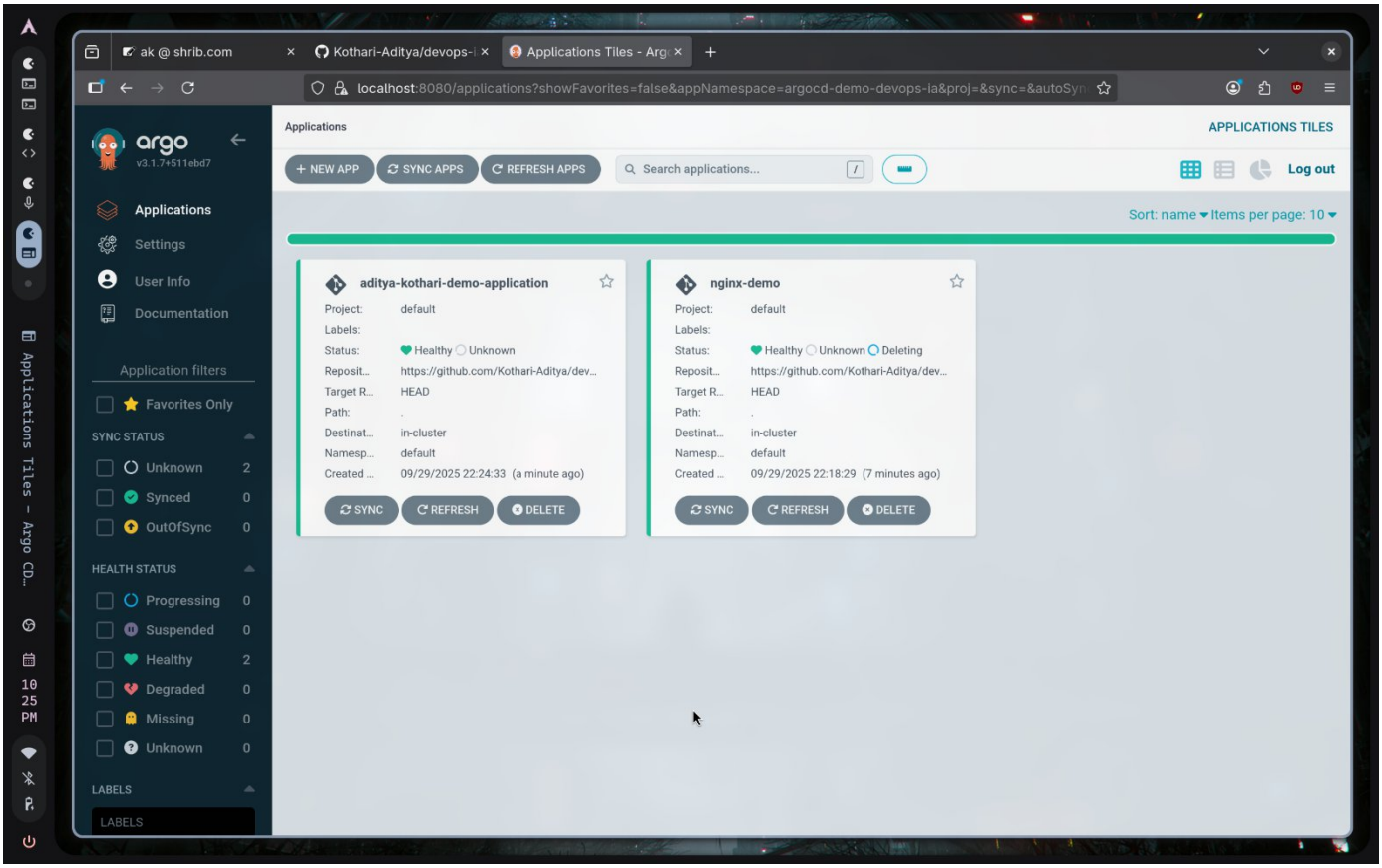
   Logs into ArgoCD CLI pointing to local UI.

```
◄ 0s ⋈ ⊙ kubectl port-forward svc/argocd-server -n argocd-demo-devops-
ia 8080:443                                                  □ devops-ia
 △|⊙*| 22:11
Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
Handling connection for 8080
Handling connection for 8080
E0929 22:14:23.917693   18630 portforward.go:404] "Unhandled Error" er
r="error copying from local connection to remote stream: writeto tcp4
127.0.0.1:8080->127.0.0.1:43296: read tcp4 127.0.0.1:8080->127.0.0.1:4
3296: read: connection reset by peer"
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
```

```
◄ 0s ⊙ kubectl -n argocd-demo-devops-ia get secret argocd-initial-admi
n-secret -o jsonpath="{.data.password}" | base64 -d; echo
-mssaYDbm2AS-2AO

◄ 0s ⊙ |                                                  ⌂ 22:14
```

```
127.0.0.1:8080->127.0.0.1:43296: read tcp4 127.0.0.1:8080->127.0.0.1:4
3296: read: connection reset by peer"
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
E0929 22:20:37.939747   18630 portforward.go:391] "Unhandled Error" er
r="error copying from remote stream to local connection: readfrom tcp4
 127.0.0.1:8080->127.0.0.1:33188: write tcp4 127.0.0.1:8080->127.0.0.1
:33188: write: broken pipe"
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
E0929 22:22:38.134373   18630 portforward.go:391] "Unhandled Error" er
r="error copying from remote stream to local connection: readfrom tcp4
 127.0.0.1:8080->127.0.0.1:58256: write tcp4 127.0.0.1:8080->127.0.0.1
:58256: write: broken pipe"
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
E0929 22:23:25.906231   18630 portforward.go:391] "Unhandled Error" er
r="error copying from remote stream to local connection: readfrom tcp4
 127.0.0.1:8080->127.0.0.1:36862: write tcp4 127.0.0.1:8080->127.0.0.1
:36862: write: broken pipe"
Handling connection for 8080
```

```
◄ 0s ⊙ kubectl apply -f ~/dev/devops-ia/app.yaml              ⌂ 22:23
application.argoproj.io/aditya-kothari-demo-application created

◄ 0s ⊙ kubectl -n argocd-demo-devops-ia get applications      ⌂ 22:24
NAME                              SYNC STATUS   HEALTH STATUS
aditya-kothari-demo-application   Unknown       Healthy
nginx-demo                        Unknown       Healthy

◄ 0s ⊙ |                                                      ⌂ 22:24
```



4.4 Git Repository Manifests

- deployment.yaml: Defines NGINX Deployment with replicas: 1.
- service.yaml: Exposes NGINX via ClusterIP on port 80.

deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-demo
  labels:
    app: nginx-demo
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx-demo
  template:
    metadata:
      labels:
        app: nginx-demo
    spec:
      containers:
      - name: nginx
        image: nginx:stable
        ports:
        - containerPort: 80
```

service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-demo
spec:
  selector:
    app: nginx-demo
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: ClusterIP
```

4.5 ArgoCD Application Definition

Local file app.yaml configures:

- Git repository URL, path, and revision
- Destination cluster URL and namespace
- Automated sync policy (prune: true, selfHeal: false)

app.yaml

```yaml
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: aditya-demo-app
  namespace: argocd-devops-demo
spec:
  project: default
  source:
    repoURL: https://github.com/Kothari-Aditya/devops-ia-argo-cd-demo.git
    targetRevision: HEAD
    path: .
  destination:
    server: https://kubernetes.default.svc
    namespace: default
  syncPolicy:
    automated:
      prune: true
      selfHeal: false
    syncOptions:
    - CreateNamespace=true
```

4.6 Deploy ArgoCD Application

1. kubectl create namespace argocd-devops-demo

Namespace for the new application.

2. kubectl apply -f fix-rbac.yaml

Grants cluster-admin role to argocd-application-controller service account.

3. kubectl rollout restart statefulset argocd-application-controller -n argocd

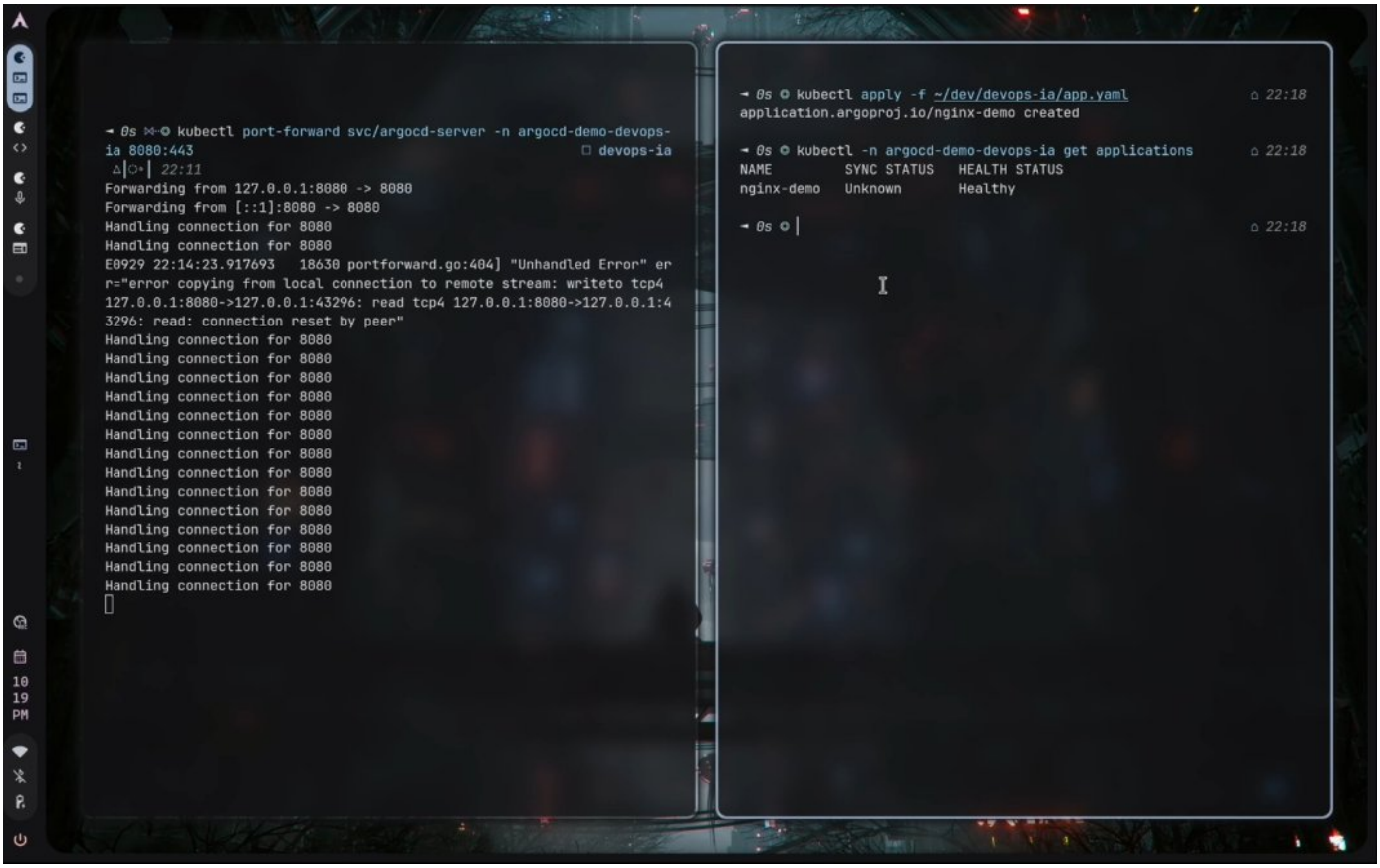
Restarts controller to apply RBAC changes.

4. kubectl apply -f app.yaml

Registers the Application resource with ArgoCD.

5. kubectl get applications -n argocd-devops-demo

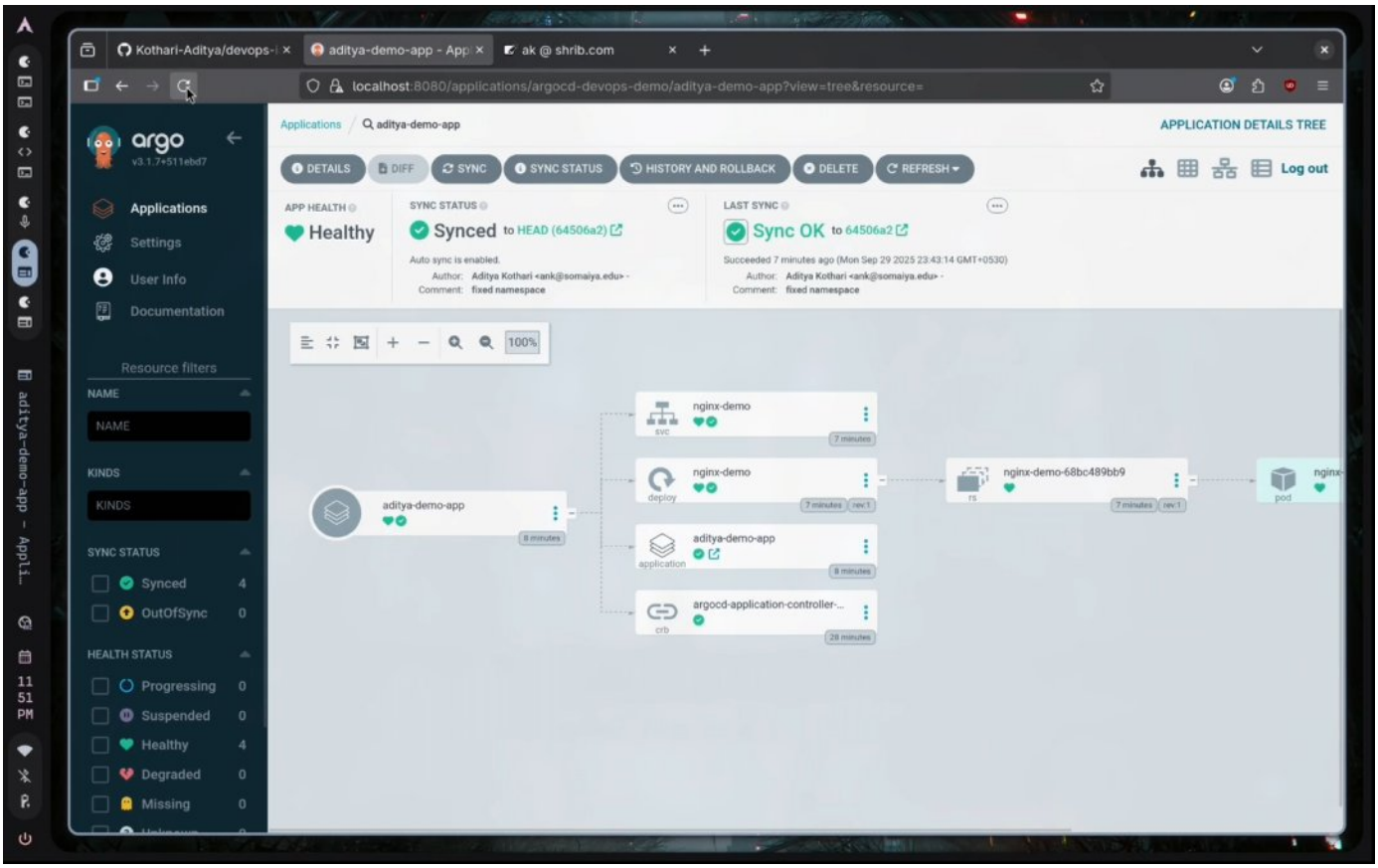
Confirms the application 'aditya-demo-app' is known to ArgoCD.



5. Core Features Demonstration

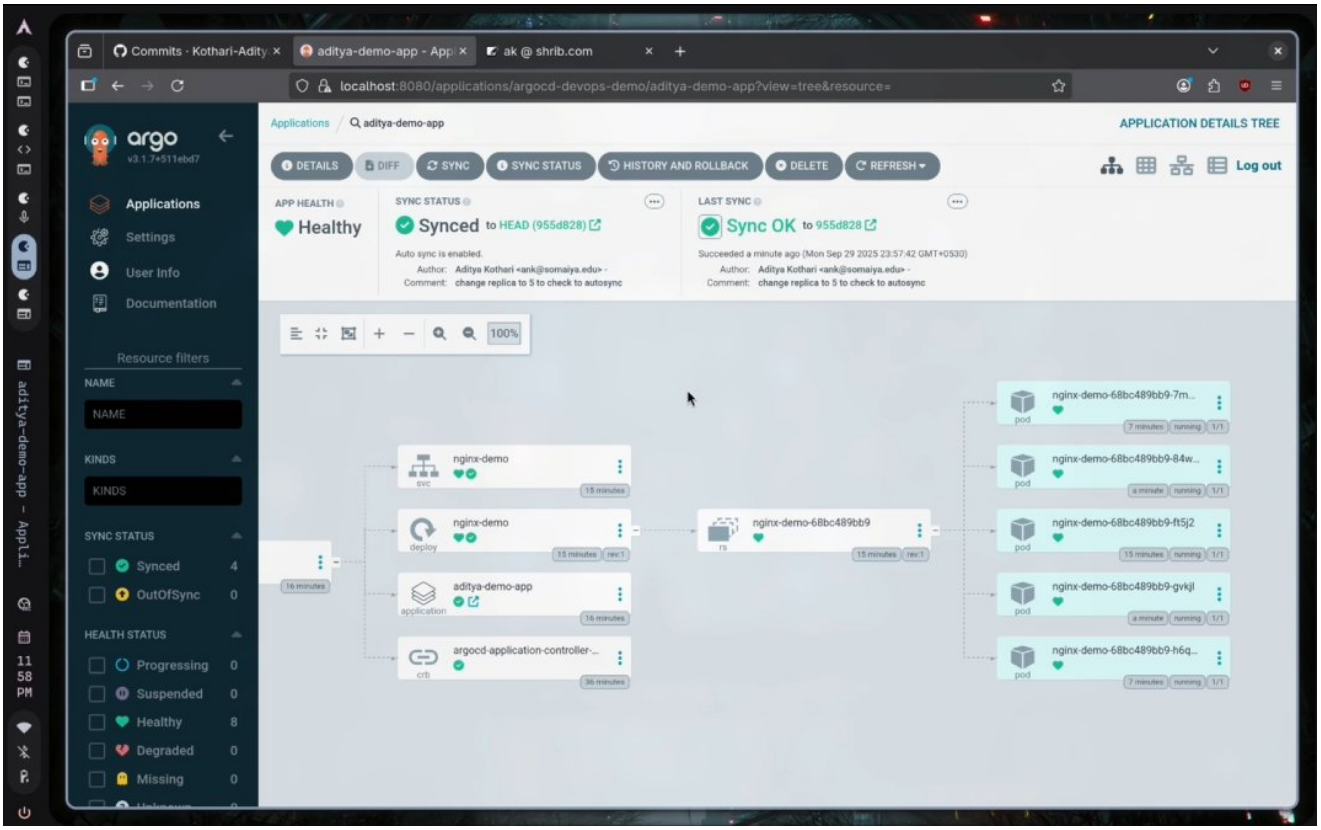
5.1 Initial Sync & Health

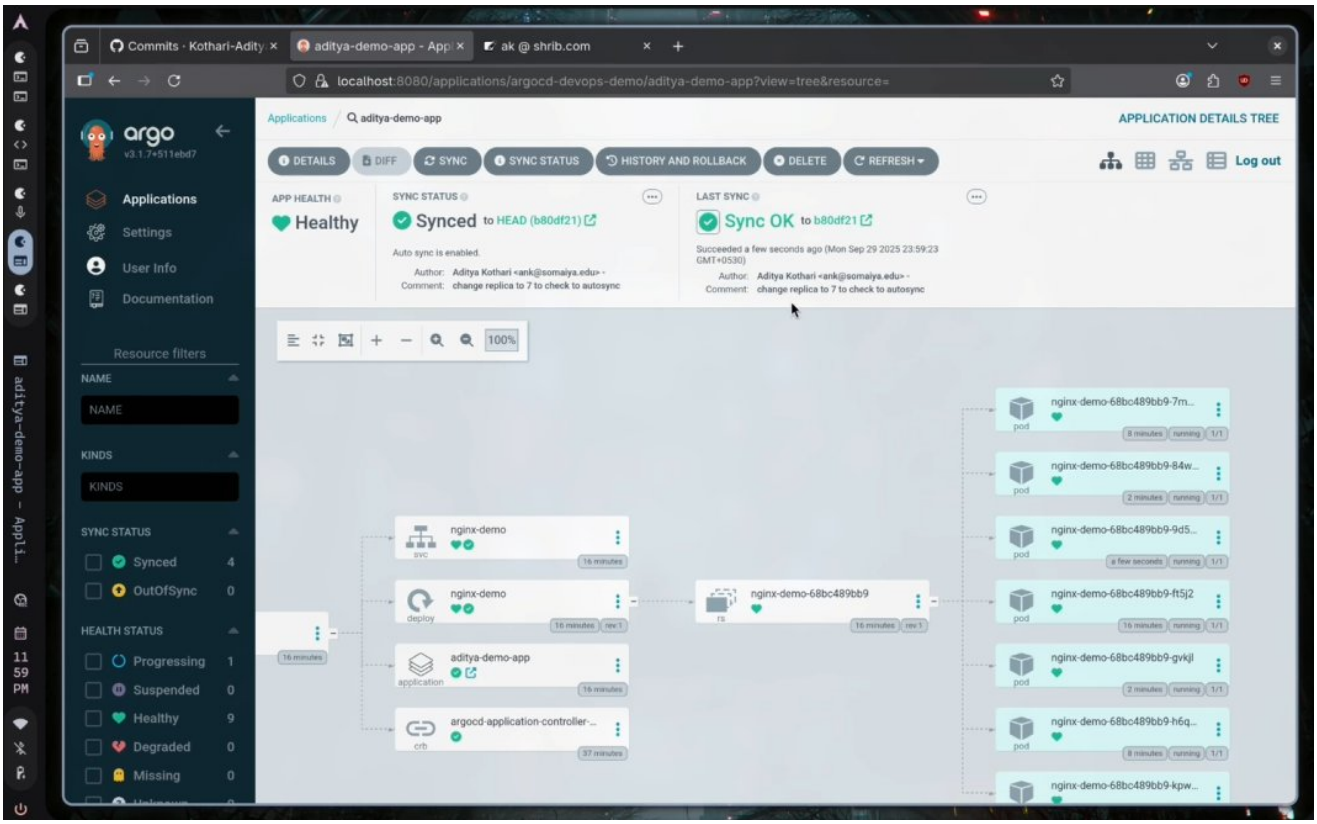
- UI automatically syncs manifests, deploying NGINX.
- Resource tree shows deployment, service, replica set, pod.



5.2 Auto-Sync on Commit

1. Update deployment.yaml to replicas: From 5 to 7.
2. Commit & push triggers ArgoCD to detect diff.
3. UI transitions: OutOfSync → Synced.
4. Kubernetes pods scale to 7.


Commits · Kothari-Adity
aditya-demo-app - App
ak @ shrlb.com
localhost:8080/applications/argocd-devops-demo/aditya-demo-app?view=tree&resource=
argo
v3.1.7+511ebd7
Applications
Settings
User Info
Documentation
Resource filters
NAME
KINDS
SYNC STATUS
Synced 4
OutOfSync 0
HEALTH STATUS
Progressing 0
Suspended 0
Healthy 8
Degraded 0
Missing 0
Applications / aditya-demo-app
APPLICATION DETAILS TREE
DETAILS
DIFF
SYNC
SYNC STATUS
HISTORY AND ROLLBACK
DELETE
REFRESH
Log out
APP HEALTH
Healthy
SYNC STATUS
Synced to HEAD (955d828)
Auto sync is enabled.
Author: Aditya Kothari <ank@somaiya.edu> -
Comment: change replica to 5 to check to autosync
LAST SYNC
Sync OK to 955d828
Succeeded a minute ago (Mon Sep 29 2025 23:57:42 GMT+0530)
Author: Aditya Kothari <ank@somaiya.edu> -
Comment: change replica to 5 to check to autosync
100%
nginx-demo
nginx-demo
aditya-demo-app
argocd-application-controller-...
nginx-demo-68bc489bb9
nginx-demo-68bc489bb9-7m...
nginx-demo-68bc489bb9-84w...
nginx-demo-68bc489bb9-ft5j2
nginx-demo-68bc489bb9-gvkjl
nginx-demo-68bc489bb9-h6q...

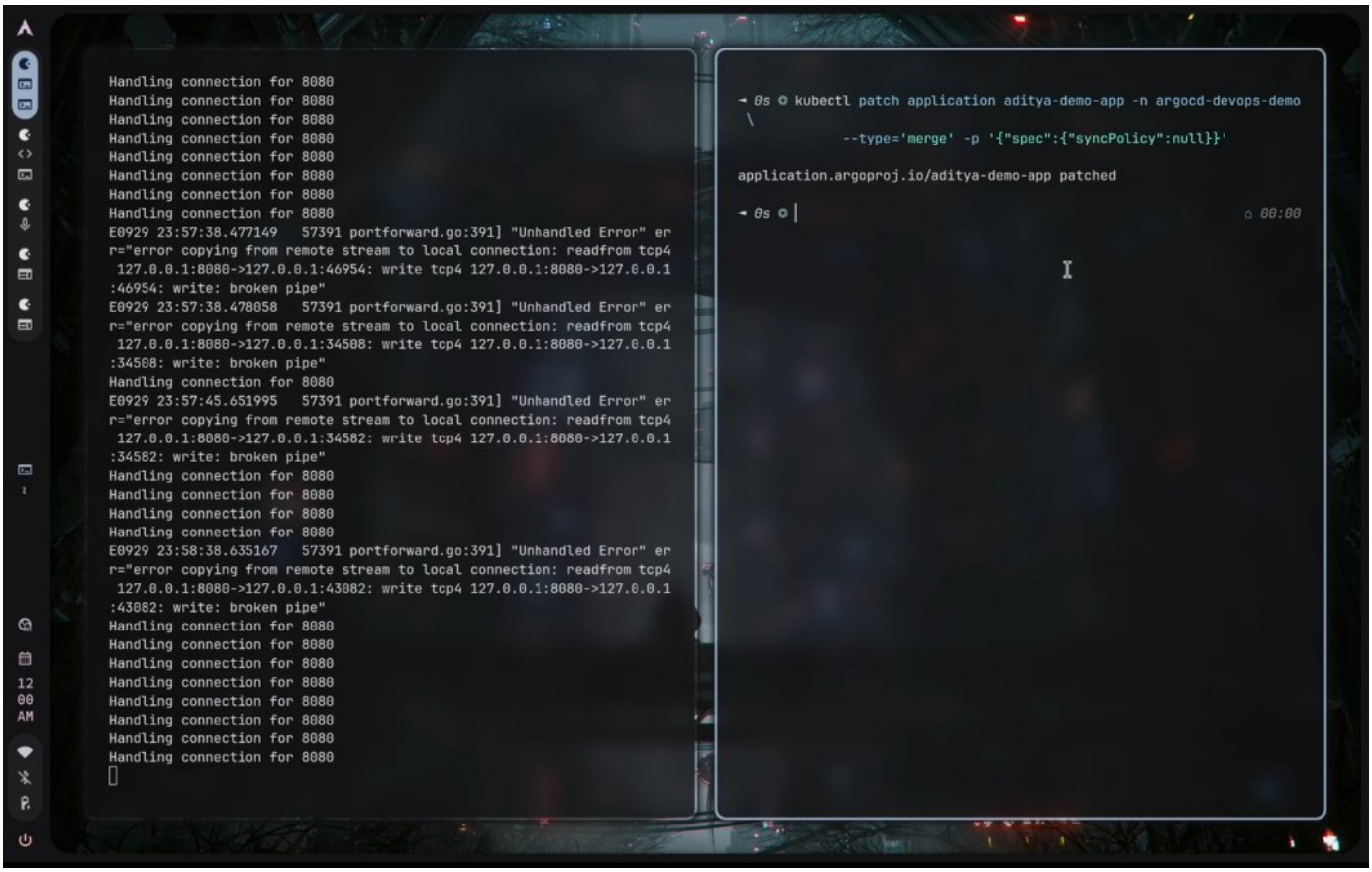

Commits · Kothari-Adity
aditya-demo-app - App
ak @ shrlb.com
localhost:8080/applications/argocd-devops-demo/aditya-demo-app?view=tree&resource=
argo
v3.1.7+511ebd7
Applications
Settings
User Info
Documentation
Resource filters
NAME
KINDS
SYNC STATUS
Synced 4
OutOfSync 0
HEALTH STATUS
Progressing 1
Suspended 0
Healthy 9
Degraded 0
Missing 0
Applications / aditya-demo-app
APPLICATION DETAILS TREE
DETAILS
DIFF
SYNC
SYNC STATUS
HISTORY AND ROLLBACK
DELETE
REFRESH
Log out
APP HEALTH
Healthy
SYNC STATUS
Synced to HEAD (b80df21)
Auto sync is enabled.
Author: Aditya Kothari <ank@somaiya.edu> -
Comment: change replica to 7 to check to autosync
LAST SYNC
Sync OK to b80df21
Succeeded a few seconds ago (Mon Sep 29 2025 23:59:23 GMT+0530)
Author: Aditya Kothari <ank@somaiya.edu> -
Comment: change replica to 7 to check to autosync
100%
nginx-demo
nginx-demo
aditya-demo-app
argocd-application-controller-...
nginx-demo-68bc489bb9
nginx-demo-68bc489bb9-7m...
nginx-demo-68bc489bb9-84w...
nginx-demo-68bc489bb9-9d5...
nginx-demo-68bc489bb9-ft5j2
nginx-demo-68bc489bb9-gvkjl
nginx-demo-68bc489bb9-h6q...
nginx-demo-68bc489bb9-kpw...

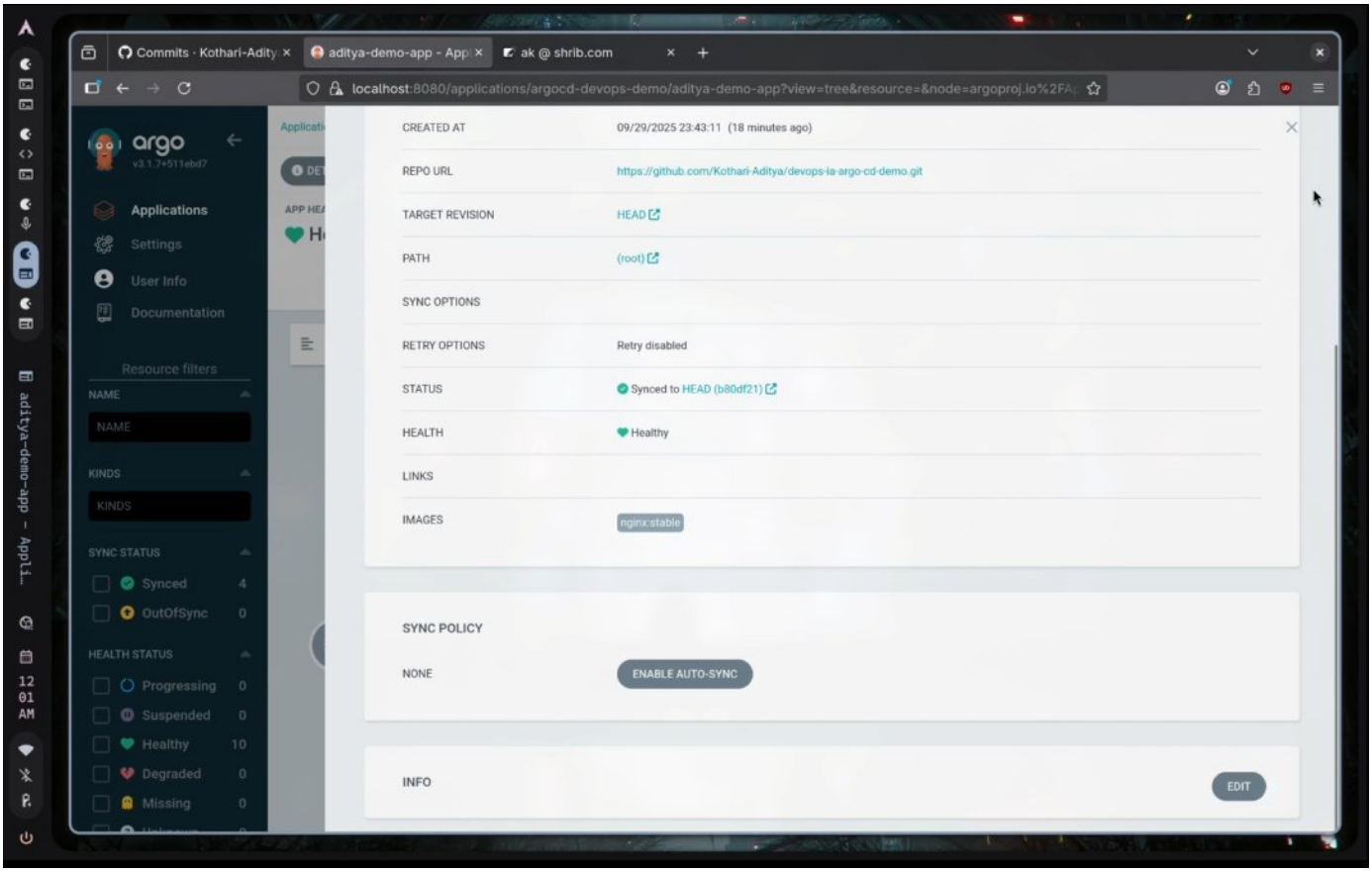
5.3 Manual Sync (Rollback)

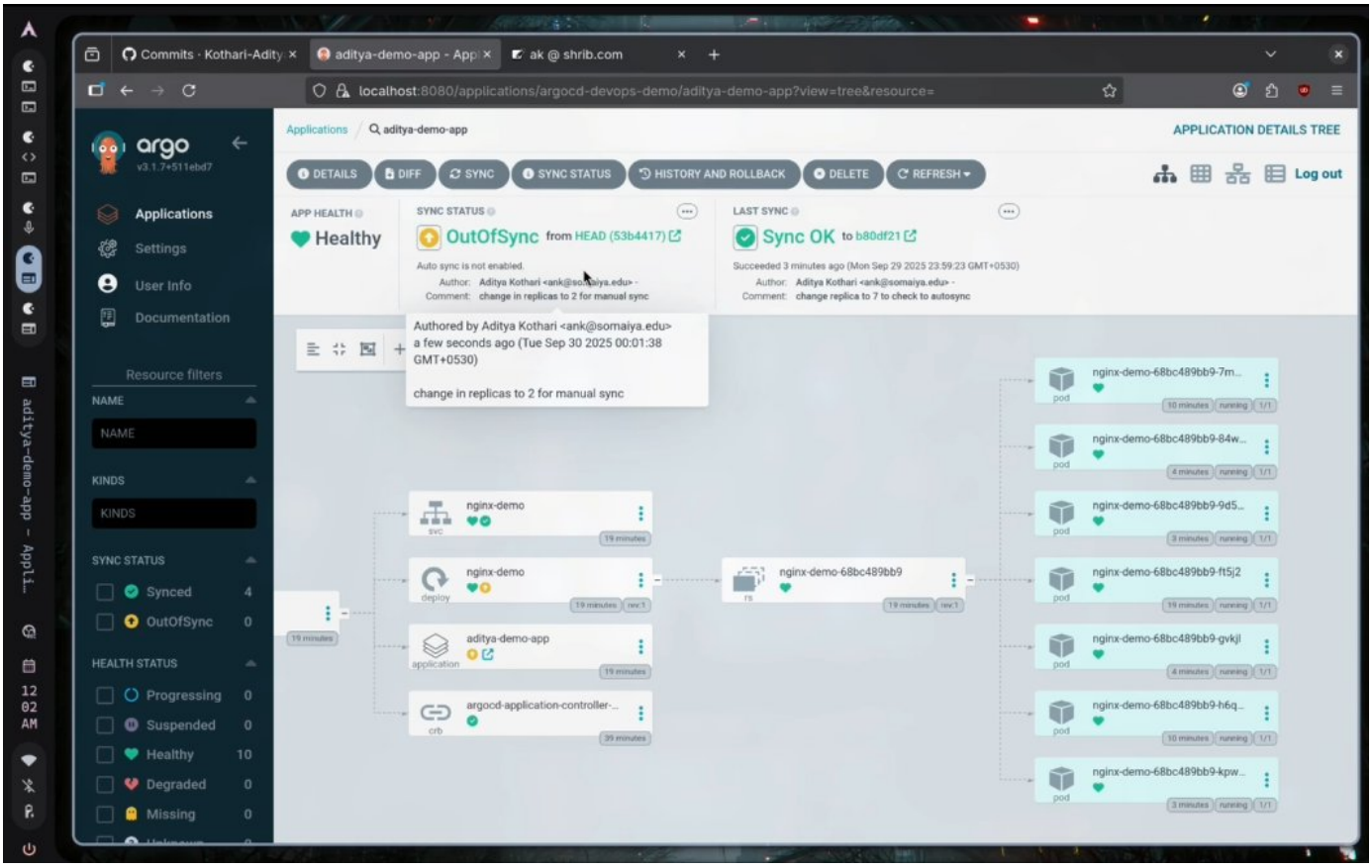
1. Disable automated syncing:

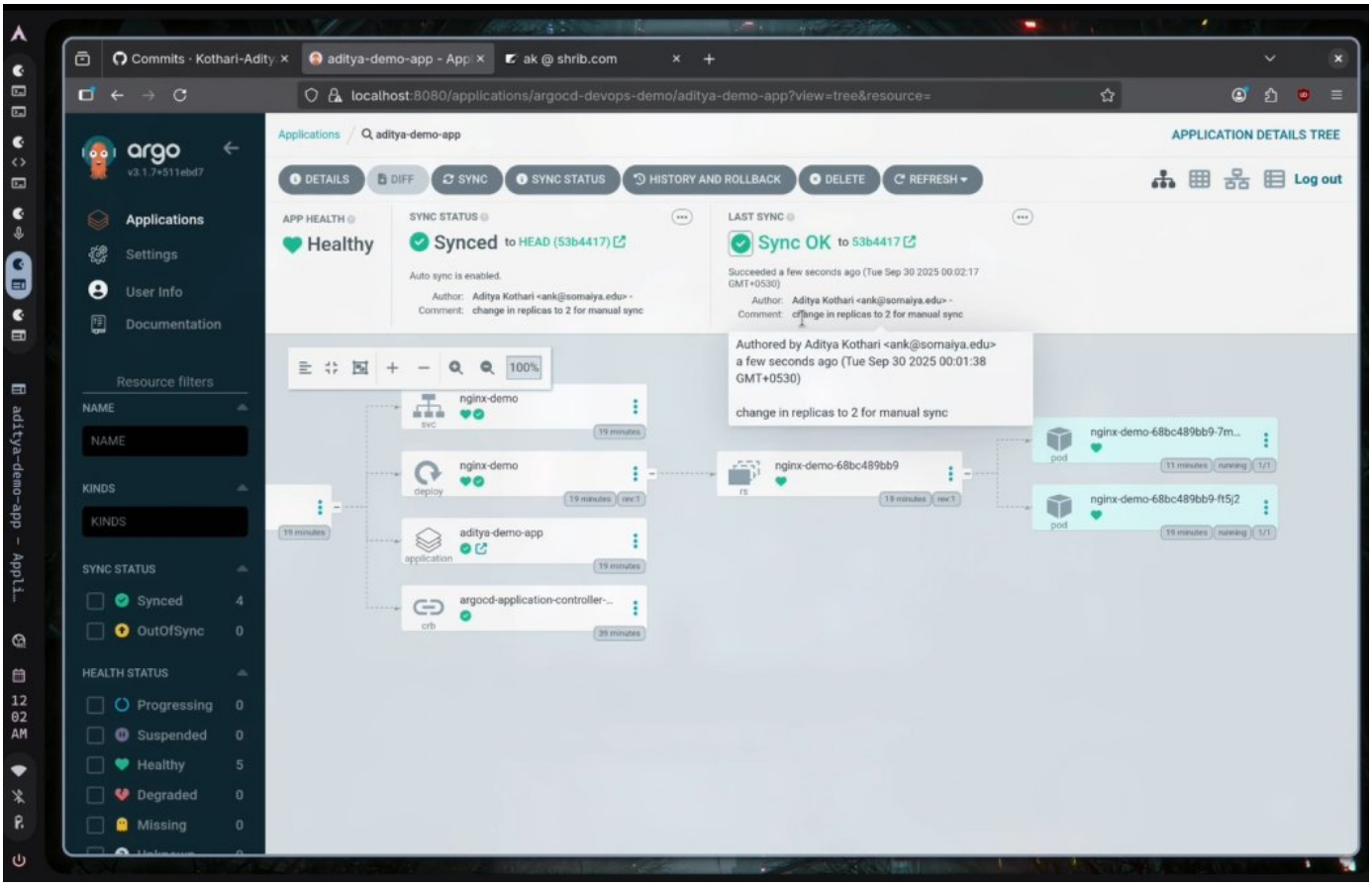
   kubectl patch application aditya-demo-app -n argocd-devops-demo --type merge -p '{"spec":{"syncPolicy":null}}'

2. Revert replicas to 2 from 7, commit & push.
3. Click **Sync** in UI to apply manual rollback.




Commits · Kothari-Adity
aditya-demo-app - Appl
ak @ shrib.com
localhost:8080/applications/argocd-devops-demo/aditya-demo-app?view=tree&resource=&node=argoproj.io%2FA
argo
v3.1.7+511ebd7
Applications
Settings
User Info
Documentation
Resource filters
NAME
KINDS
SYNC STATUS
Synced 4
OutOfSync 0
HEALTH STATUS
Progressing 0
Suspended 0
Healthy 10
Degraded 0
Missing 0
CREATED AT
09/29/2025 23:43:11 (18 minutes ago)
REPO URL
https://github.com/Kothari-Aditya/devops-ia-argo-cd-demo.git
TARGET REVISION
HEAD
PATH
(root)
SYNC OPTIONS
RETRY OPTIONS
Retry disabled
STATUS
Synced to HEAD (b80df21)
HEALTH
Healthy
LINKS
IMAGES
nginx:stable
SYNC POLICY
NONE
ENABLE AUTO-SYNC
INFO
EDIT


Commits · Kothari-Adity
aditya-demo-app - Appl
ak @ shrib.com
localhost:8080/applications/argocd-devops-demo/aditya-demo-app?view=tree&resource=
argo
v3.1.7+511ebd7
Applications
Settings
User Info
Documentation
Resource filters
NAME
KINDS
SYNC STATUS
Synced 4
OutOfSync 0
HEALTH STATUS
Progressing 0
Suspended 0
Healthy 10
Degraded 0
Missing 0
Applications / aditya-demo-app
APPLICATION DETAILS TREE
DETAILS
DIFF
SYNC
SYNC STATUS
HISTORY AND ROLLBACK
DELETE
REFRESH
Log out
APP HEALTH
Healthy
SYNC STATUS
OutOfSync from HEAD (53b4417)
Auto sync is not enabled.
Author: Aditya Kothari <ank@somaiya.edu> -
Comment: change in replicas to 2 for manual sync
LAST SYNC
Sync OK to b80df21
Succeeded 3 minutes ago (Mon Sep 29 2025 23:59:23 GMT+0530)
Author: Aditya Kothari <ank@somaiya.edu> -
Comment: change replica to 7 to check to autosync
Authored by Aditya Kothari <ank@somaiya.edu>
a few seconds ago (Tue Sep 30 2025 00:01:38 GMT+0530)
change in replicas to 2 for manual sync
nginx-demo
svc
nginx-demo
deploy
aditya-demo-app
application
argocd-application-controller-...
crb
nginx-demo-68bc489bb9
rs
nginx-demo-68bc489bb9-7m...
nginx-demo-68bc489bb9-84w...
nginx-demo-68bc489bb9-9d5...
nginx-demo-68bc489bb9-ft5j2
nginx-demo-68bc489bb9-gvkjl
nginx-demo-68bc489bb9-h6q...
nginx-demo-68bc489bb9-kpw...
pod



6. Comparison with Mainstream tools

Feature	Jenkins + kubectl	GitHub Actions + kubectl	ArgoCD
Declarative config	Partial (pipelines)	Partial (workflows)	Full (K8s CRDs)
Continuous sync	No	No	Yes
Drift detection	Manual	Manual	Automated
UI visualization	Limited	Limited	Built-in Dashboard
Rollback via UI	No	No	Yes

7. Advantages & Limitations

Advantages

- **True Declarative GitOps**: Entire app state defined in Git; promotes version control and audit trails.
- **Continuous Reconciliation**: ArgoCD constantly ensures cluster matches Git state, preventing drift.
- **Unified UI**: Centralized dashboard displaying health, sync status, and resource tree for rapid troubleshooting.
- **Built-in Rollback**: One-click rollback to any previous Git revision via UI, simplifying recovery.

Limitations

- **RBAC Complexity**: Initial setup requires careful cluster-role and role-binding configurations to grant controller permissions.
- **Learning Curve**: Understanding CRDs, sync policies, and custom resources adds complexity compared to simple scripting.
- **Resource Overhead**: Additional ArgoCD services (repo-server, server, controller) run as pods, consuming cluster resources.
- **Limited Non-Git Sources**: Primarily designed for Git-based manifests; less native support for other artifact types without extra tooling.

8. Conclusion

ArgoCD provides a robust GitOps experience, automating deployments and drift detection while offering a rich UI and easy rollback. Its declarative, continuous sync model significantly reduces manual toil and errors compared to imperative workflows.

9. References

- ArgoCD Documentation: argo-cd.readthedocs.io
- Kubernetes kubectl Reference: kubernetes.io/docs/reference/kubectl/
- GitOps Principles: www.gitops.tech