**Setting up Minikube on Windows 11 with Docker Driver: Challenges, Solutions, and Installation Process**

**Introduction:**

In this document, we'll discuss setting up Minikube, a local Kubernetes cluster, on a Windows 11 system using Docker as the primary driver. We'll also cover the installation process for Kubernetes (`kubectl`), Argo CD, and Argo Rollouts.

**Installation Process:**

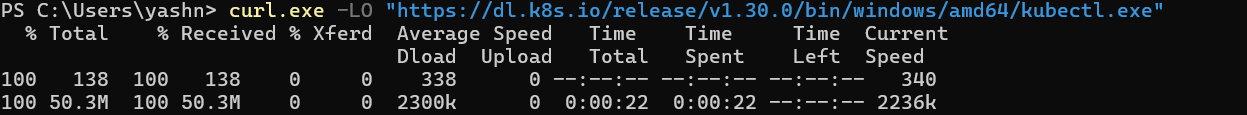
**1. Installing Kubernetes (kubectl):**

**Install kubectl binary with curl on Windows**

**Step1:** Download the latest 1.30 patch release: [kubectl 1.30.0](https://dl.k8s.io/release/v1.30.0/bin/windows/amd64/kubectl.exe).

Or if you have curl installed, use this command:

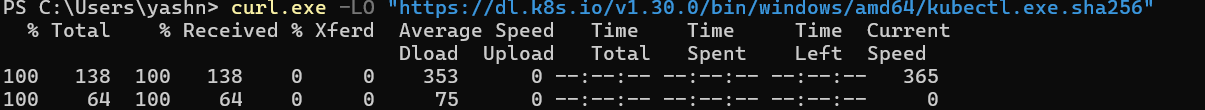
curl.exe -LO "https://dl.k8s.io/release/v1.30.0/bin/windows/amd64/kubect

****

**Step2:** Validate the binary (optional)

Download the kubectl checksum file:

curl.exe -LO "https://dl.k8s.io/v1.30.0/bin/windows/amd64/kubectl.exe.sha256



Validate the kubectl binary against the checksum file:

* Using Command Prompt to manually compare CertUtil's output to the checksum file downloaded:

CertUtil -hashfile kubectl.exe SHA256

**type** kubectl.exe.sha256

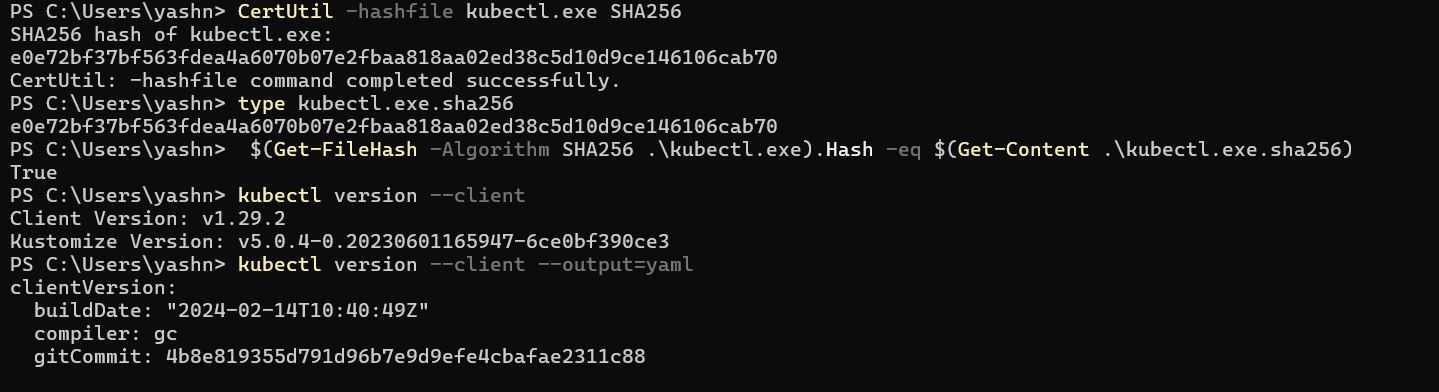
Using PowerShell to automate the verification using the -eq operator to get a True or False result:

$(Get-FileHash -Algorithm SHA256 .\kubectl.exe).Hash -eq $(Get-Co

1. Test to ensure the version of kubectl is the same as downloaded:
2. kubectl version --client

Or use this for detailed view of version:

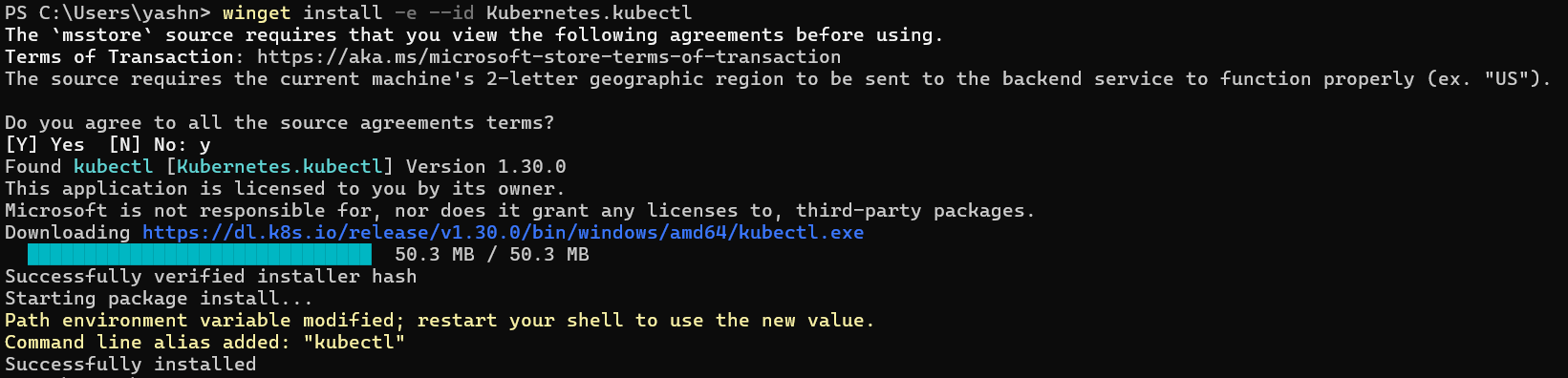
kubectl version --client --output=yaml

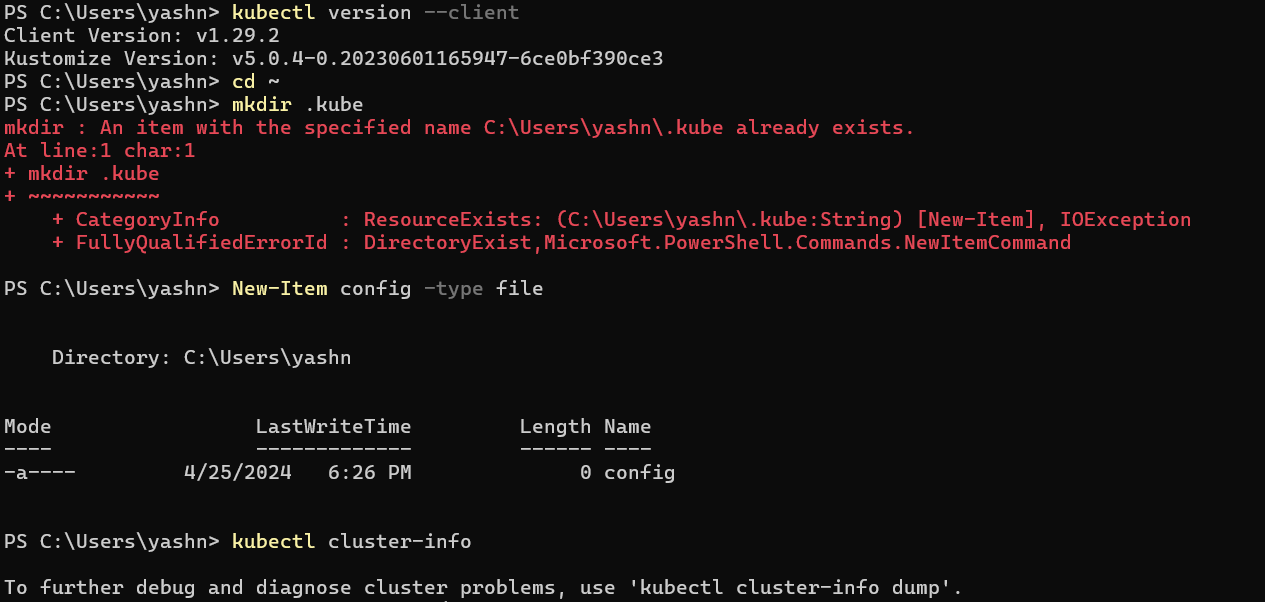


**Install on Windows using winget**

1. To install kubectl on Windows you can use either [Chocolatey](https://chocolatey.org/) package manager, [Scoop](https://scoop.sh/) command-line installer, or [winget](https://learn.microsoft.com/en-us/windows/package-manager/winget/) package manager.
   * [winget](https://kubernetes.io/docs/tasks/tools/install-kubectl-windows/#kubectl-win-install-2)
2. winget install -e --id Kubernetes.kubectl
3. Test to ensure the version you installed is up-to-date:
4. kubectl version --client
5. Navigate to your home directory:
6. *# If you're using cmd.exe, run: cd %USERPROFILE%*
7. cd ~
8. Create the .kube directory:
9. mkdir .kube
10. Change to the .kube directory you just created:
11. cd .kube
12. Configure kubectl to use a remote Kubernetes cluster:

New-Item config -type file





(My file already exists )

Check that kubectl is properly configured by getting the cluster state:

kubectl cluster-info

If you see a URL response, kubectl is correctly configured to access your cluster.

If you see a message similar to the following, kubectl is not configured correctly or is not able to connect to a Kubernetes cluster.

The connection to the server <server-name:port> was refused - did you specify the right host or port?

If kubectl cluster-info returns the url response but you can't access your cluster, to check whether it is configured properly, use:

kubectl cluster-info dump

**Install Docker Desktop for Windows:**

- Download and install Docker Desktop for Windows from the official Docker website.

- Follow the installation instructions and ensure Docker Desktop is running.

**2. Setting up Minikube with Docker Driver:**

**Step 1: Install Minikube CLI:**

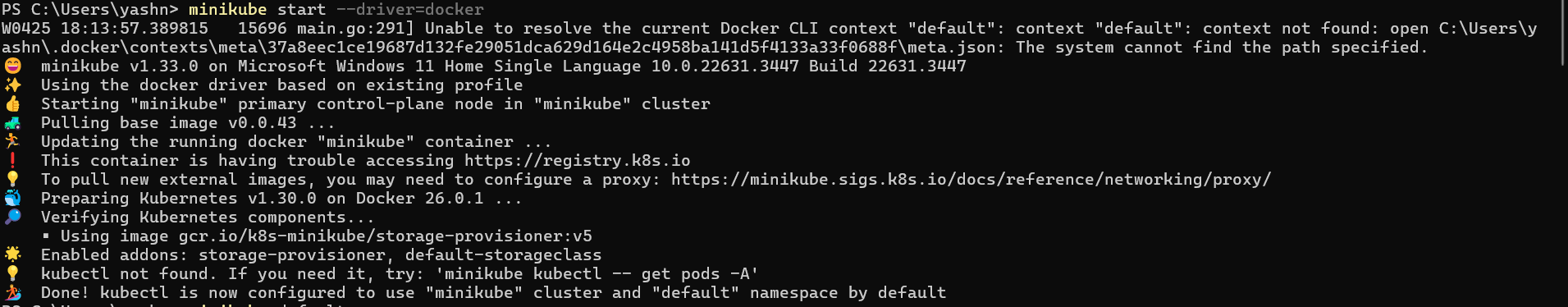
- Download and install the Minikube CLI for Windows from the official Minikube GitHub releases page.

**Step 2: Start Minikube with Docker Driver:**

- Open Command Prompt or PowerShell as Administrator.

- Run the following command to start Minikube using the Docker driver:

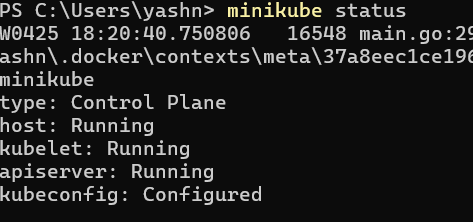
Minikube start –driver=docker

****

**Step 3: Verify Minikube status:**

After Minikube has started successfully, verify the status of the Minikube cluster:

Minikube status

****

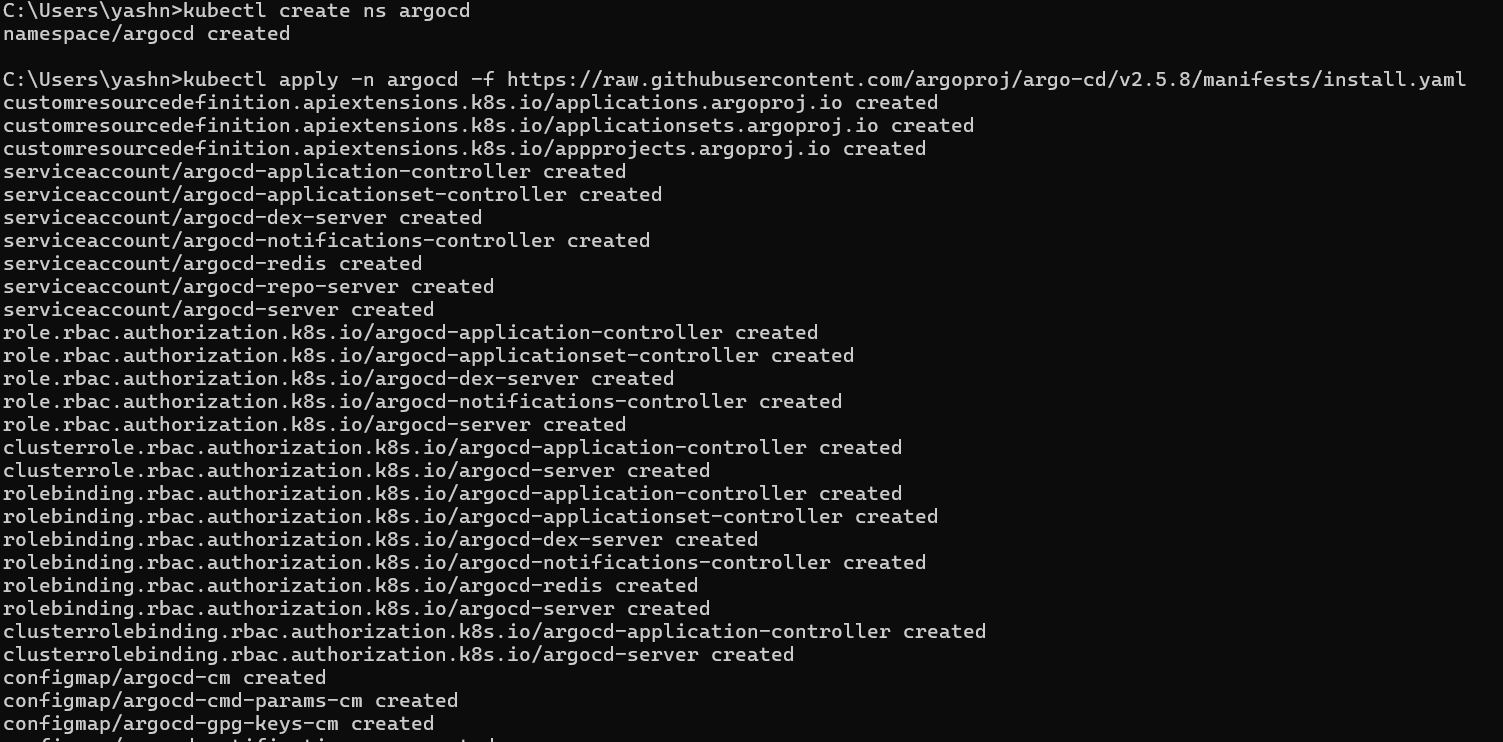
**3. Installing Argo CD and Argo Rollouts:**

**Step 1: Install Argo CD:**

Apply the Argo CD manifests using `kubectl`:

Kubectl create namespace argocd

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

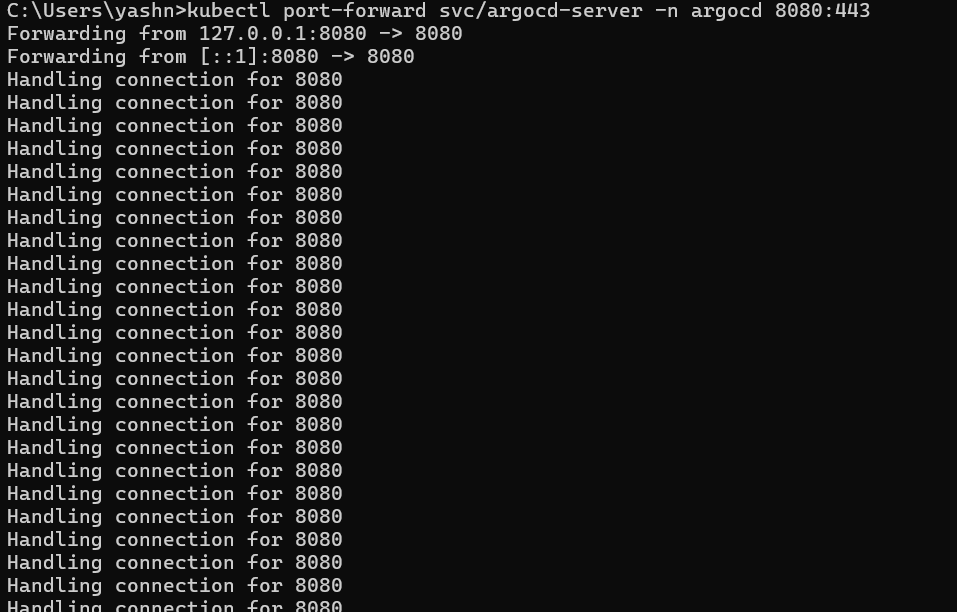
****

**Step 2: Access Argo CD Web UI:**

Port-forward the Argo CD server to access the web UI:

kubectl get all -n argocd

Kubectl port-forward svc/argocd-server -n argocd 8080:443

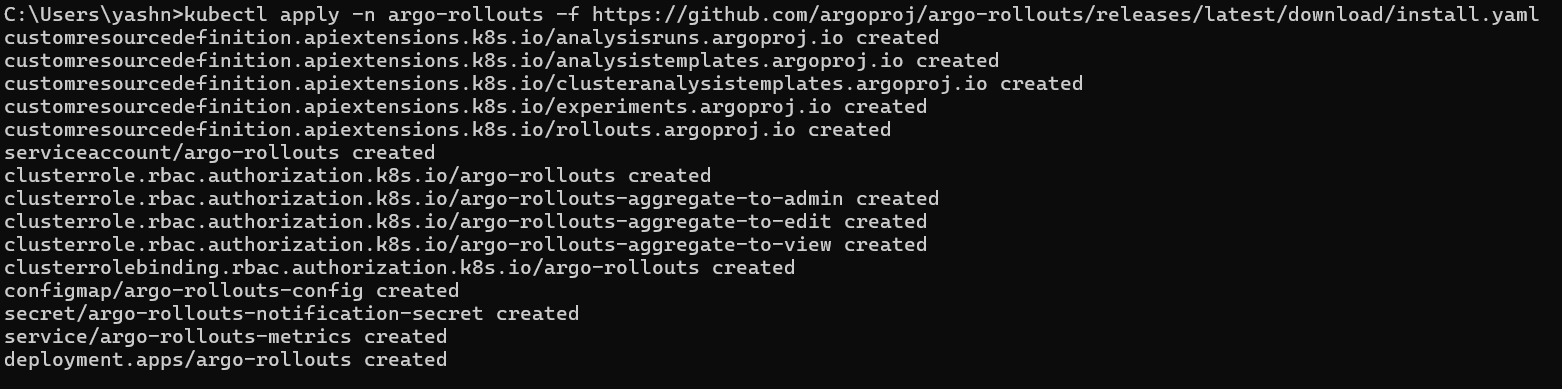
****

Open a web browser and navigate to `https://localhost:8080` to access the Argo CD dashboard (username: `admin`, password: retrieve using `kubectl`).

**Step 3: Install Argo Rollouts (Canary):**

- Apply the Argo Rollouts CRDs and controller using `kubectl`:

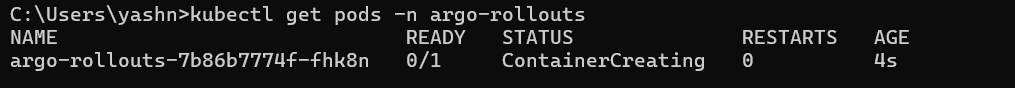
Kubectl apply -n argo-rollouts -f https://github.com/argoproj/argo-rollouts/releases/latest/download/install.yaml

****

**Step 4: Verify Argo Rollouts installation:**

- Check the deployment of Argo Rollouts:

Kubectl get pods -n argo-rollouts

****

**Step 5: Setting up rollout manifests**

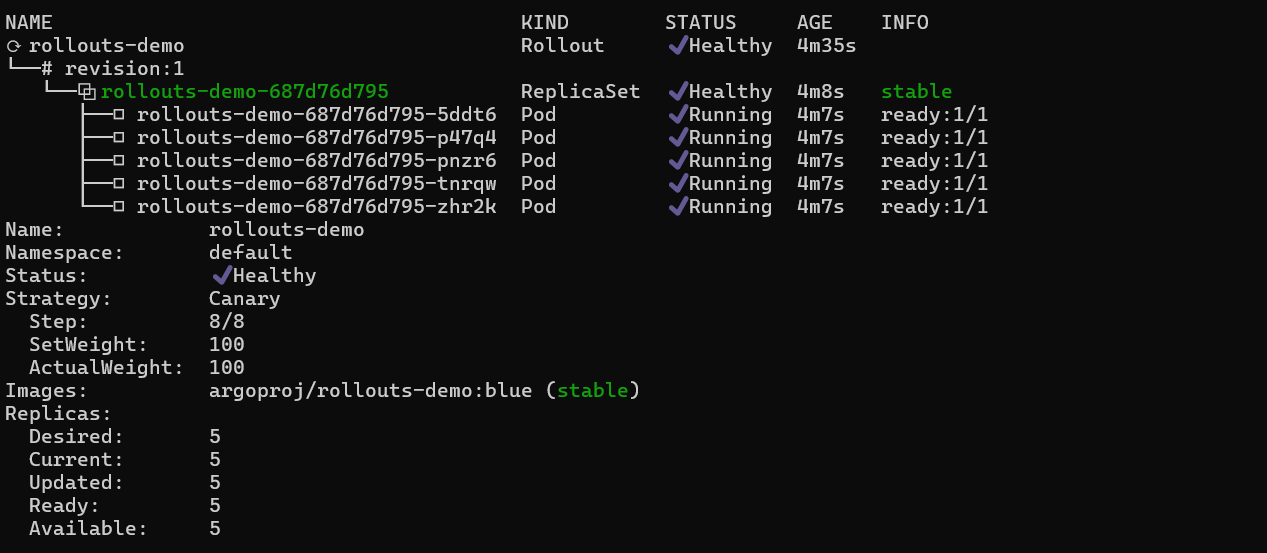
Now create your rollout manifests for services.yaml and rollout.yaml and use code

kubectl apply -f <https://raw.githubusercontent.com/argoproj/argo-rollouts/master/docs/getting-started/basic/rollout.yaml>

kubectl apply -f https://raw.githubusercontent.com/argoproj/argo-rollouts/master/docs/getting-started/basic/service.yaml

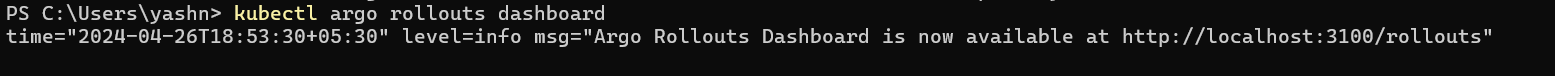
**Step 6: Execute argo rollout**

kubectl argo rollouts get rollout rollouts-demo --watch

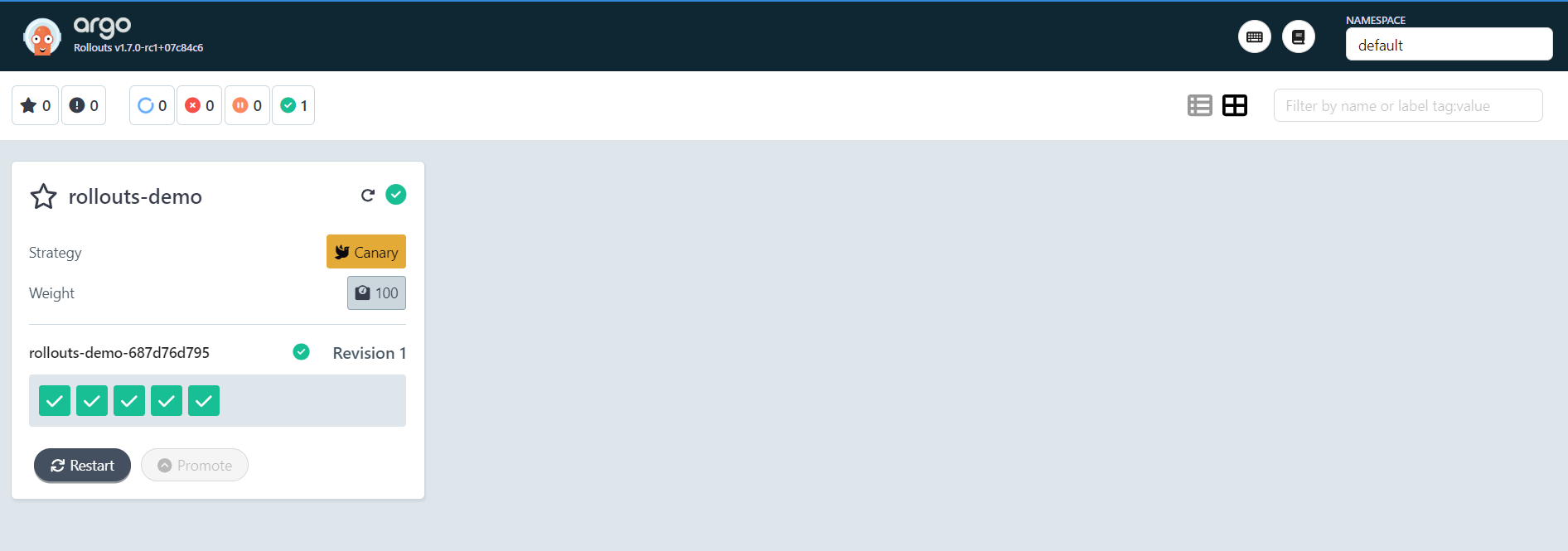
**Step 7: Execute argo rollout dashboard**

Execute argo rollout dashboard to host it locally

kubectl argo rollouts dashboard



**This is how the dash board looks**

****

**Cleaning Up everything**

**1.Delete Argo Rollouts Resources:**

Delete any existing Rollouts, Analysis Templates, Experiments, or other Argo Rollouts resources in your Kubernetes cluster using kubectl delete commands. For example:

kubectl delete rollouts --all

kubectl delete analysistemplates --all

kubectl delete experiments --all

**2.Delete Argo CD Resources (if installed):**

If you've installed Argo CD, delete its resources using kubectl delete. For example:

kubectl delete -n argocd applications -all

**3.Delete Minikube Cluster:**

If you're using Minikube, delete the Minikube cluster to clean up all associated resources. Run:

minikube delete

**4.Clean Up Docker Containers (if necessary):**

If you have Docker containers running locally, you can clean them up using Docker commands. For example:

Docker ps -aq | xargs docker rm -f