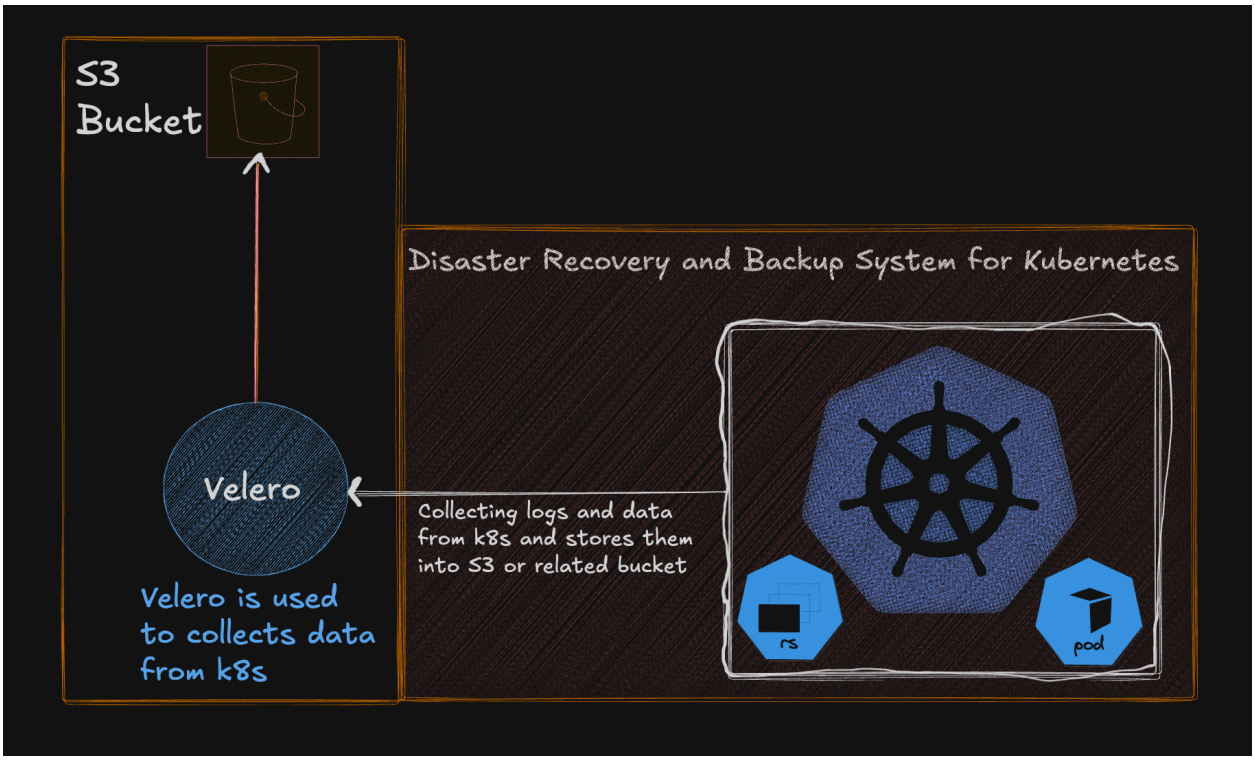
Project-2 → Kubernetes Disaster Recovery and backup using Velero.

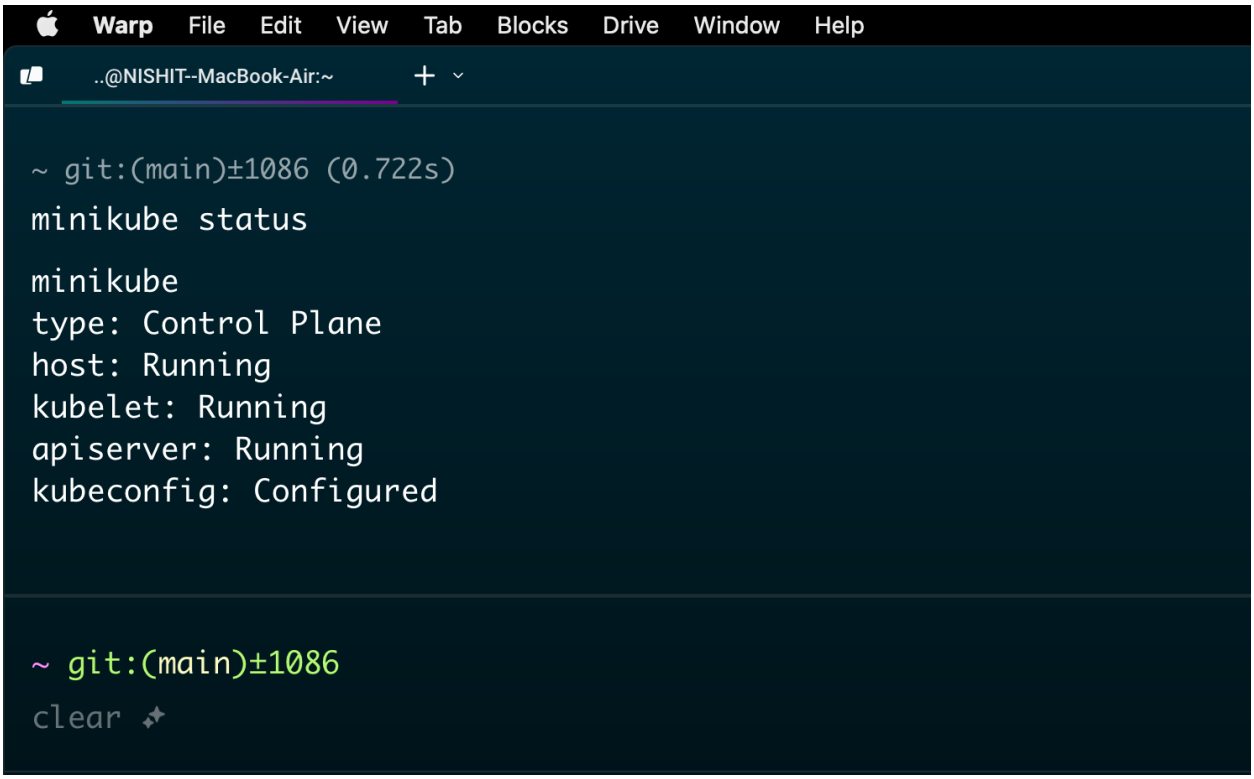
Architecture Diagram :-



Step 1 →

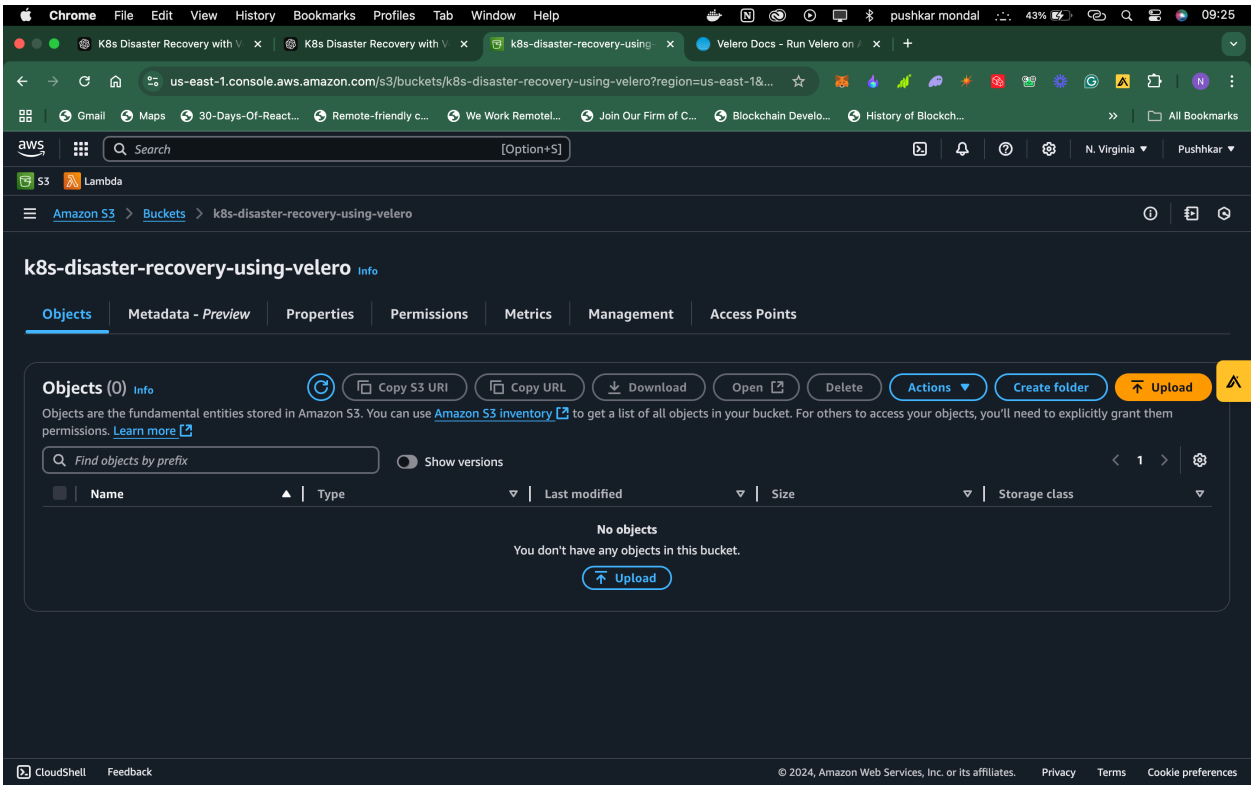
Firstly create a Kubernetes cluster to any of cloud provider such as **EKS, AKS, GKE** or you can simply setup **Minikube or Kind** cluster into your local computer.

- **For this tutorial I am using Minikube to setup a local cluster.**



Step 2 →

Now create a S3 bucket in your AWS Console.



- P.S:- Export your Bucket name and region to your console using →

```
export BUCKET=k8s-disaster-recovery-using-velero
 export REGION=us-east-1
```

- And also save your AWS Access Key and AWS Secret Access Key to some file.

Step 3 →

Install Velero into your local setup by clicking the below link according to your device configuration →

https://velero.io/docs/v1.8/basic-install/

Step 4 →

Now run this command

```
velero install \
    --provider aws \
    --plugins velero/velero-plugin-for-aws:v1.8.0 \
    --bucket $BUCKET \
    --secret-file ./credentials-velero \
    --backup-location-config region=$REGION \
    --snapshot-location-config region=$REGION
```

- In Velero docs it is not mentioned that you have to use the `--plugin` flag, otherwise you will get an error.

Step 5 →

- Create a namespace using `kubectl create ns <namespace_name>`
- Now create a simple deployment using yaml file or using command line.

  → If you are using command line you can use this command

  `k create deployment backup --image=nginx --replicas=2 -n backup`

Or if you using yaml file then you can use this yaml file below

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backup
  namespace: backup
spec:
  replicas: 2
  selector:
    matchLabels:
      app: backup
  template:
    metadata:
      labels:
```

```
        app: backup
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

- And then use `kubectl create -f <yaml_file_name>`

Step 6 →

Now create a service, i am using Minikube and I am using NodePort service.

`k expose deployment backup --name=backup-service --type=NodePort --port=80 -n backup`

Step 7→

Now you can use port forward to see if the server is running or not using below command →

`kubectl port-forward svc/backup-service 8000:80 -n backup`

Step 8→

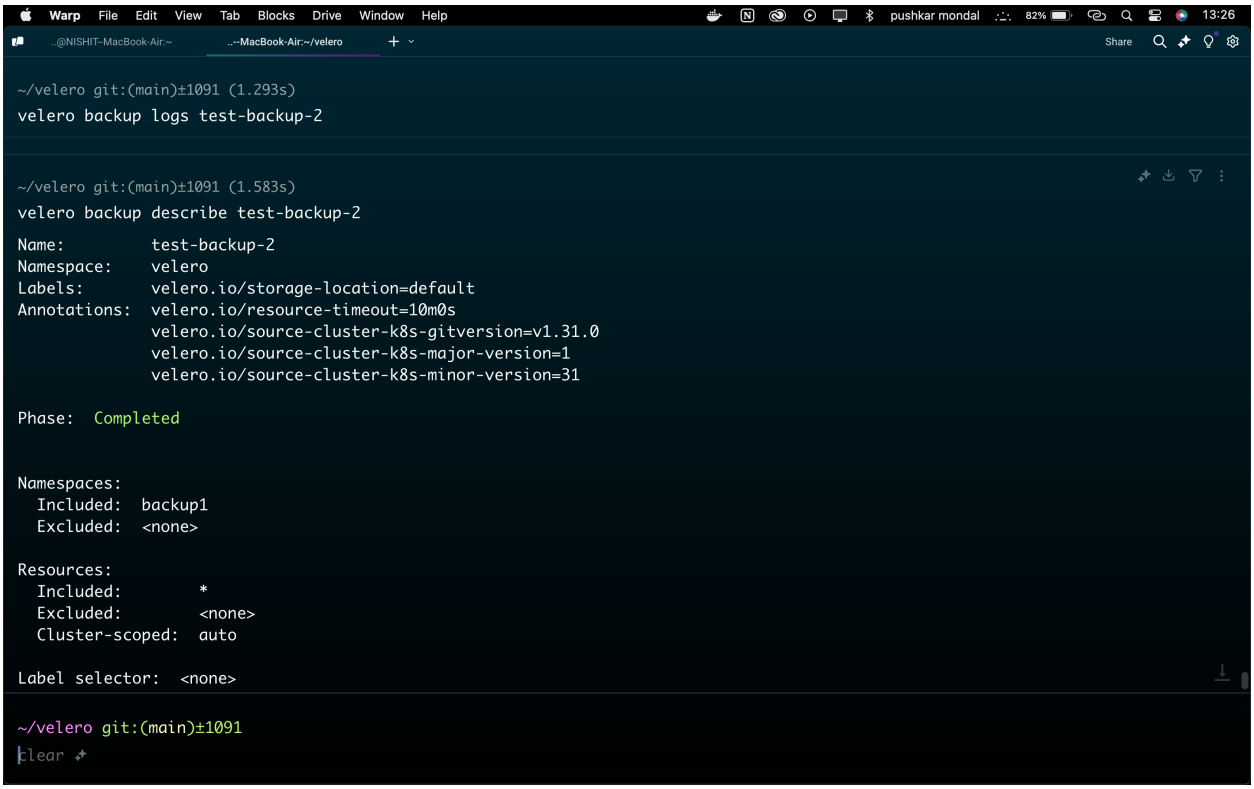
- Now create a namespace to any name as per you choice.
- Now run this below command to create a backup →

`velero backup create test-backup-2 --include-namespaces backup1`

- Now run this command to see is the backup is completed or not →

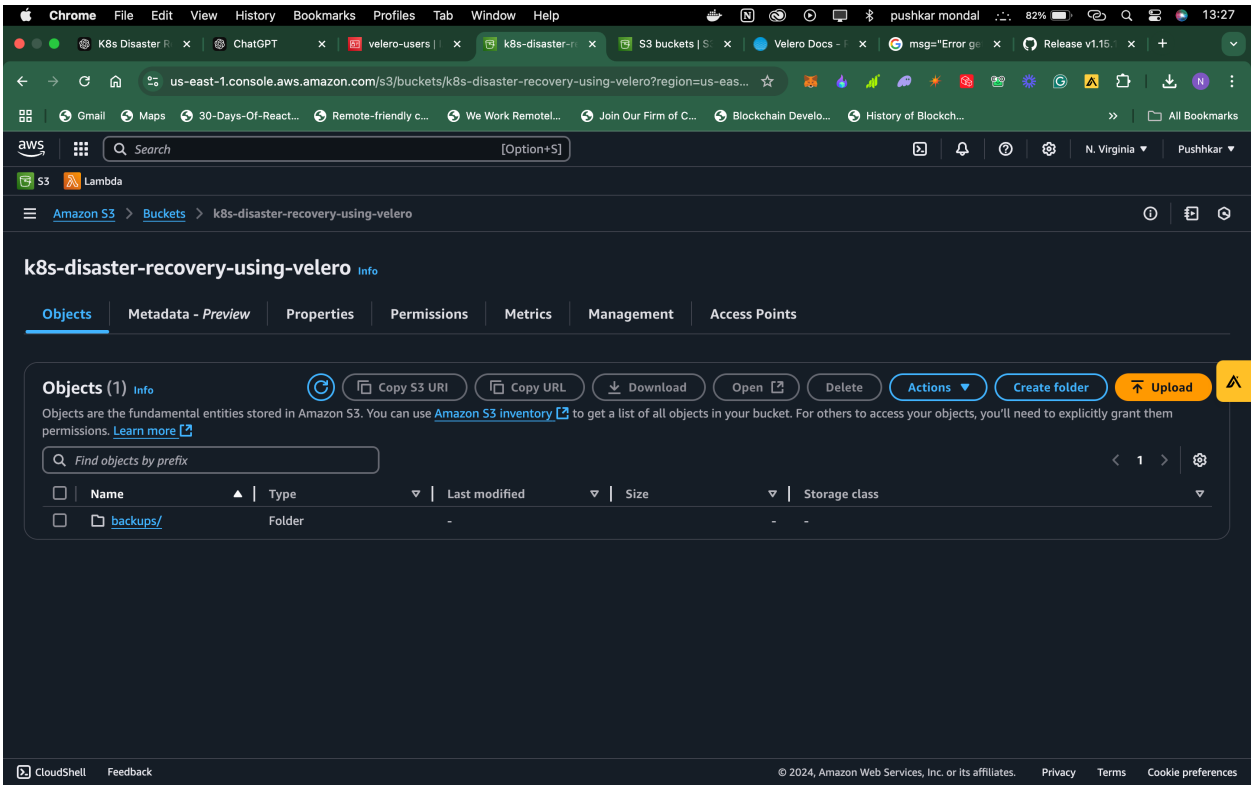
`velero backup describe test-backup-2`

It ill look like this →



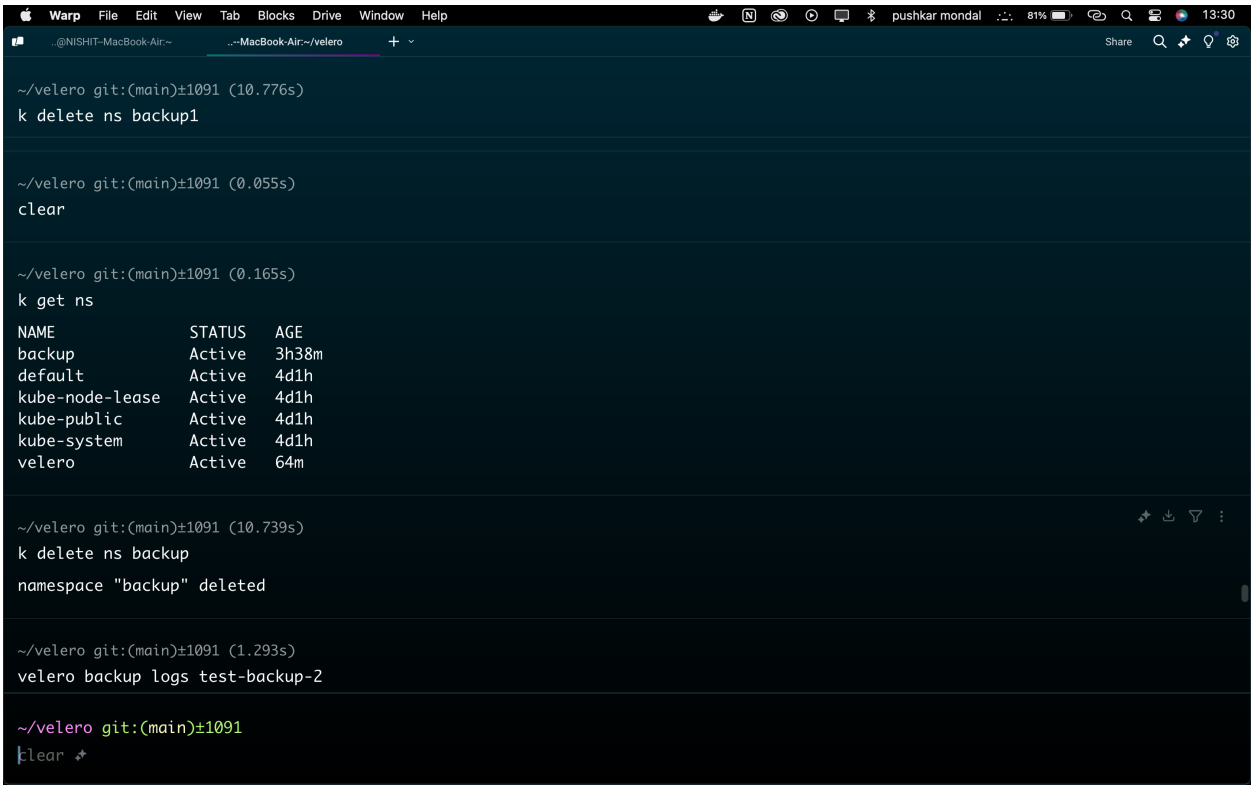
Step 9 →

Now go to your AWS console and then your S3 bucket and now your bucket look like this →



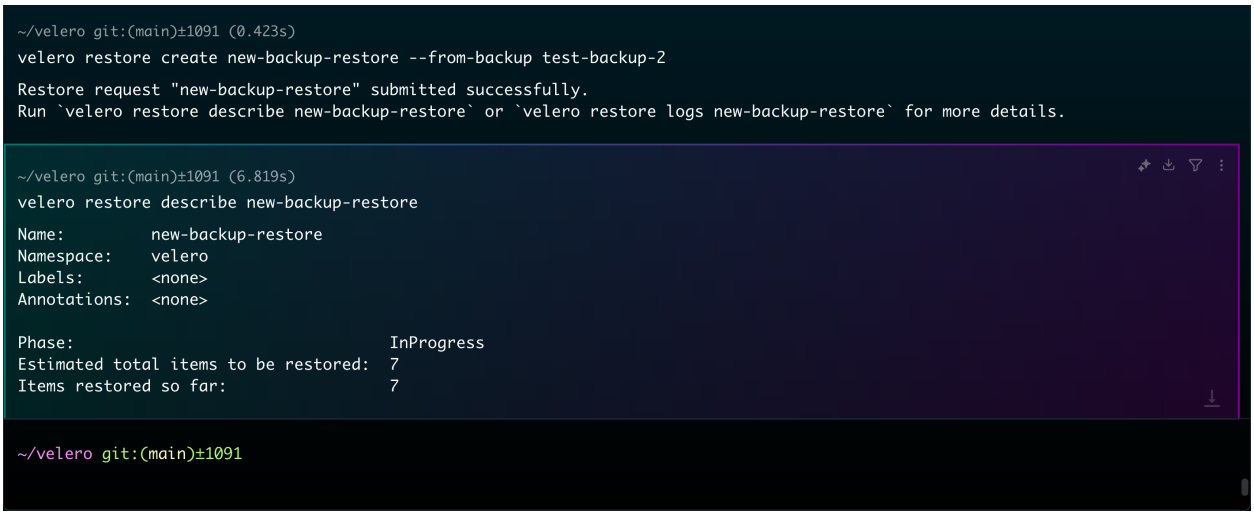
Step 10 →

Now delete the namespace that you have created for your application.

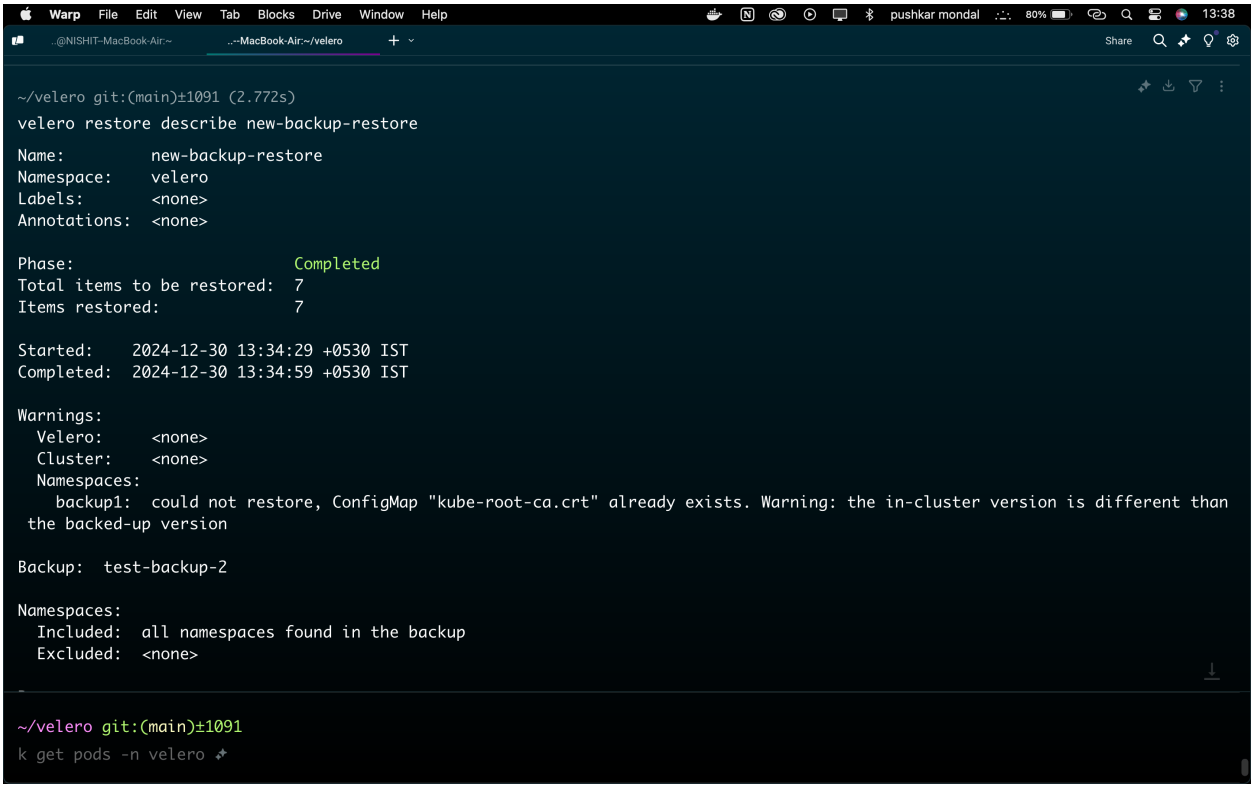


Step 11 →

Now run this below command to restore the backup again `velero restore create new-backup-restore --from-backup test-backup-2`

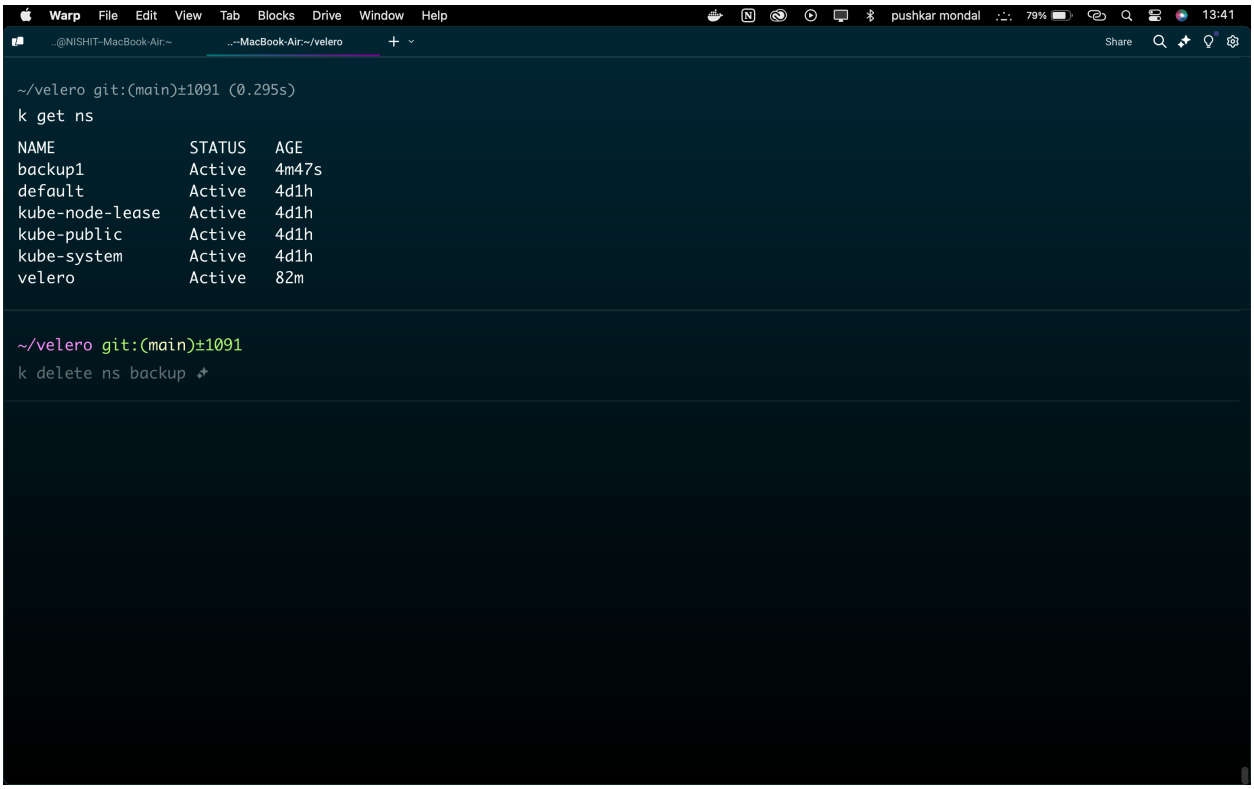


Now check the backup progress using this command `velero restore describe new-backup-restore`



Step 12 →

Now as you can see we restored our namespace again.


Warp File Edit View Tab Blocks Drive Window Help
pushkar mondal 79% 13:41
..@NISHIT-MacBook-Air:~
..~MacBook-Air:~/velero
Share
~/velero git:(main)±1091 (0.295s)
k get ns
NAME STATUS AGE
backup1 Active 4m47s
default Active 4d1h
kube-node-lease Active 4d1h
kube-public Active 4d1h
kube-system Active 4d1h
velero Active 82m
~/velero git:(main)±1091
k delete ns backup