Task Documentation

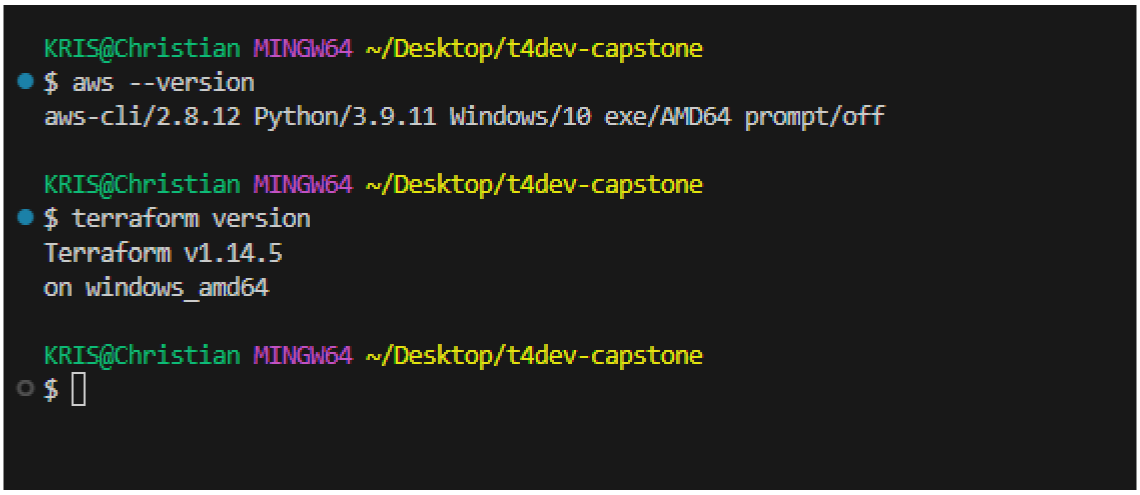
This Document shows step-by-step task done with screenshot

Infrastructure Provisioning (Terraform)

Prerequisites

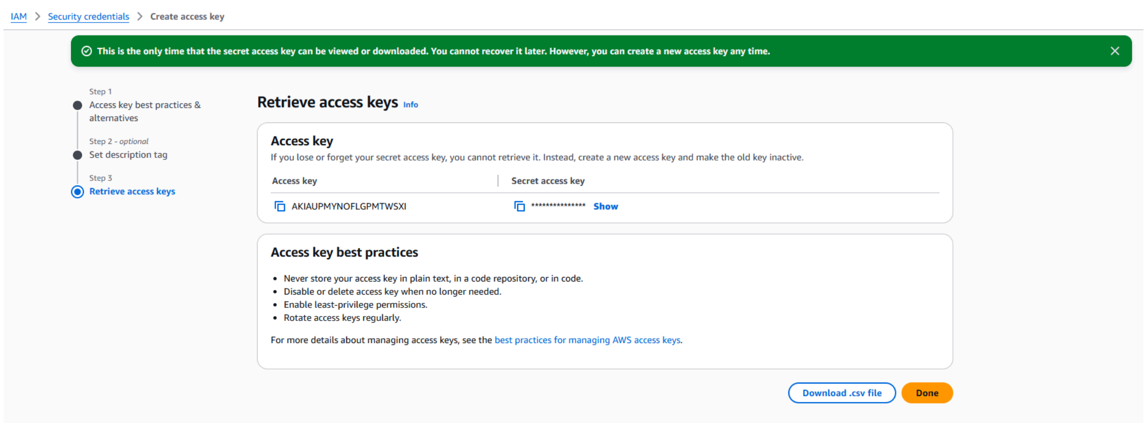
I already have the following:

- **AWS Account**
- **AWS CLI installed in my local machine. Run to check:** `aws --version`
- **Terraform installed in my local machine. Run to check:** `terraform version`



```
KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$ aws --version
aws-cli/2.8.12 Python/3.9.11 Windows/10 exe/AMD64 prompt/off

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$ terraform version
Terraform v1.14.5
on windows_amd64

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$ 
```

In my AWS Account, I created an access key and secret key for terraform to use.



IAM > Security credentials > Create access key

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAUPMYNOFLGPMTWSXI	*************** Show

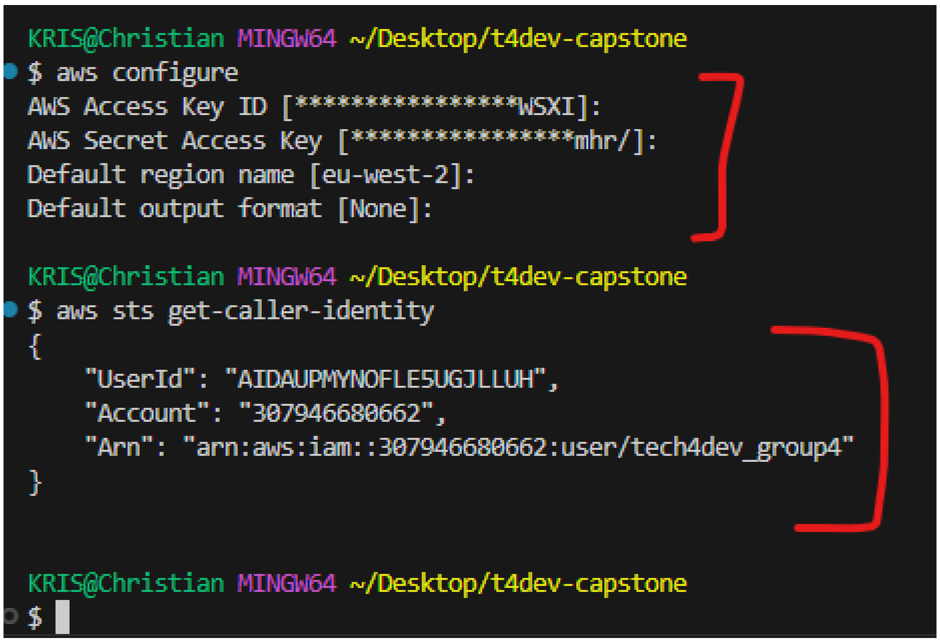
Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

Then I configure the aws cli to use the access key and secret key.

`aws configure`

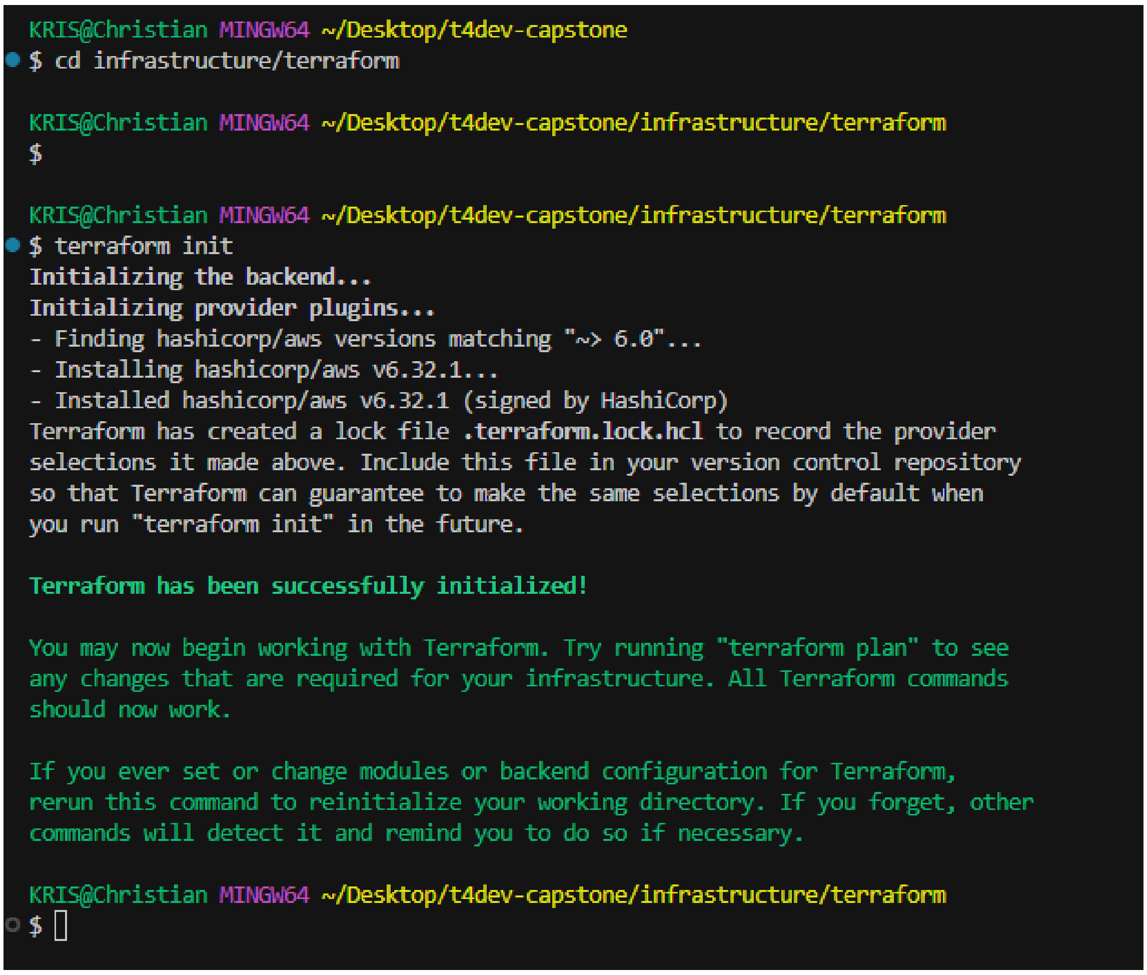


```
KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$ aws configure
AWS Access Key ID [****************WSXI]:
AWS Secret Access Key [****************mhr/]:
Default region name [eu-west-2]:
Default output format [None]:

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$ aws sts get-caller-identity
{
    "UserId": "AIDAUPMYNOFLE5UGJLLUH",
    "Account": "307946680662",
    "Arn": "arn:aws:iam::307946680662:user/tech4dev_group4"
}


KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$
```

Steps

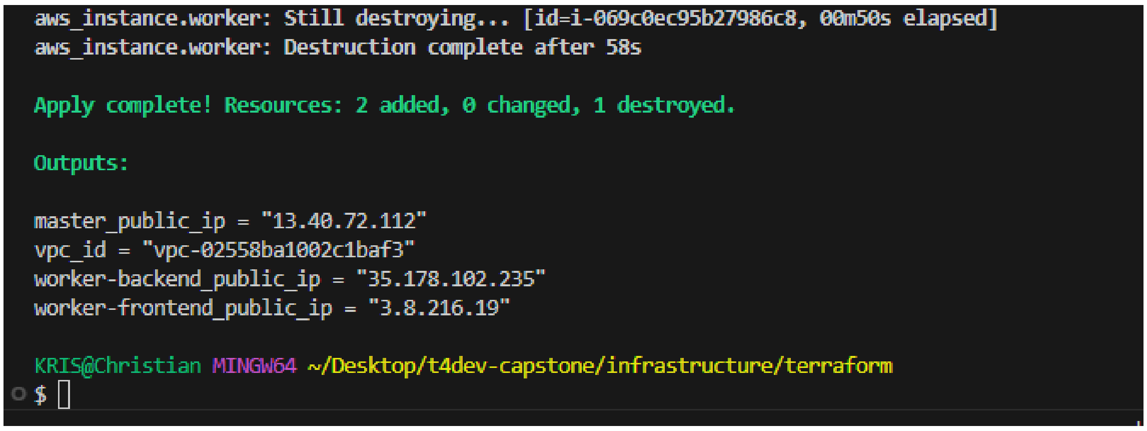
1. Goto the terraform folder and Initialize Terraform

```
cd infrastructure/terraform
terraform init
```

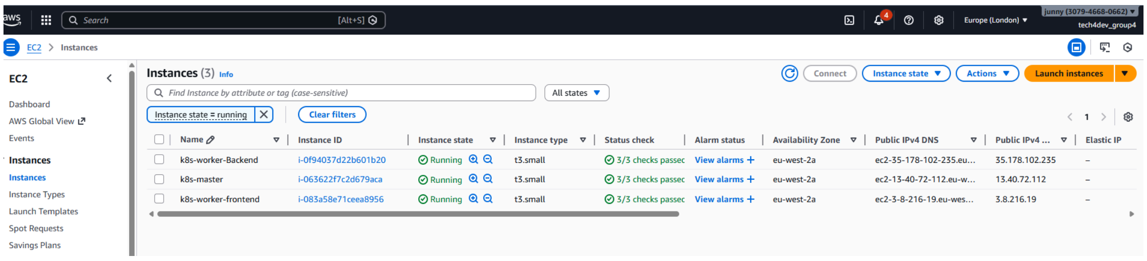


```
KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone
$ cd infrastructure/terraform

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone/infrastructure/terraform
$

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone/infrastructure/terraform
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 6.0"...
- Installing hashicorp/aws v6.32.1...
- Installed hashicorp/aws v6.32.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone/infrastructure/terraform
$
```

2. **Apply Terraform** `terraform apply`



```
aws_instance.worker: Still destroying... [id=i-069c0ec95b27986c8, 00m50s elapsed]
aws_instance.worker: Destruction complete after 58s

Apply complete! Resources: 2 added, 0 changed, 1 destroyed.

Outputs:

master_public_ip = "13.40.72.112"
vpc_id = "vpc-02558ba1002c1baf3"
worker-backend_public_ip = "35.178.102.235"
worker-frontend_public_ip = "3.8.216.19"

KRIS@Christian MINGW64 ~/Desktop/t4dev-capstone/infrastructure/terraform
$
```



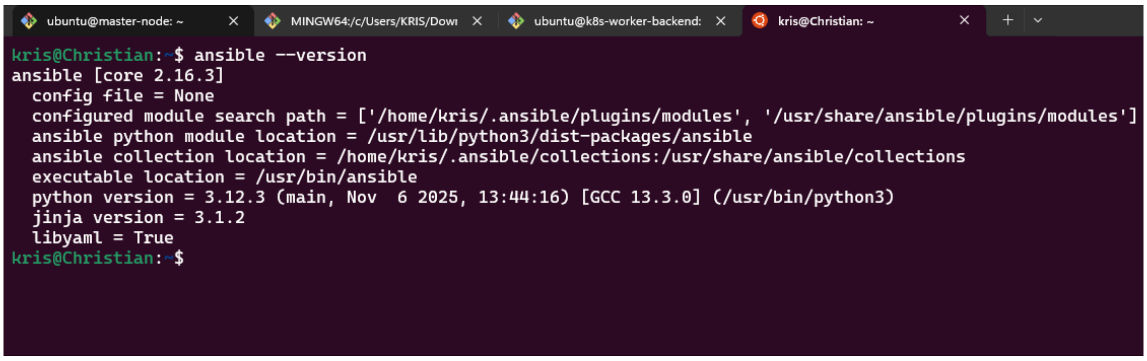
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
k8s-worker-Backend	i-0f94037d22b601b20	Running	t3.small	3/3 checks passec	View alarms +	eu-west-2a	ec2-35-178-102-235.eu...	35.178.102.235	–
k8s-master	i-063622f7c2d679aca	Running	t3.small	3/3 checks passec	View alarms +	eu-west-2a	ec2-13-40-72-112.eu-w...	13.40.72.112	–
k8s-worker-frontend	i-083a58e71ceea8956	Running	t3.small	3/3 checks passec	View alarms +	eu-west-2a	ec2-3-8-216-19.eu-wes...	3.8.216.19	–

Configuration setup (Ansible)

Prerequisites

I already have the following:

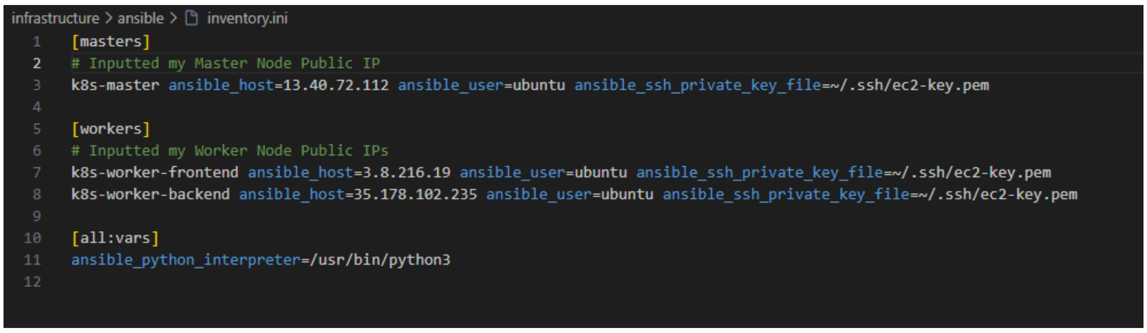
- **Ansible installed in my local machine via wsl. Run to check:** `ansible --version`



```
kris@Christian:~$ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/kris/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/kris/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Nov  6 2025, 13:44:16) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
kris@Christian:~$
```

Steps

1. Goto the ansible folder and update the inventory file `cd infrastructure/ansible`



infrastructure > ansible > inventory.ini

```
[masters]
# Inputted my Master Node Public IP
k8s-master ansible_host=13.40.72.112 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/ec2-key.pem

[workers]
# Inputted my Worker Node Public IPs
k8s-worker-frontend ansible_host=3.8.216.19 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/ec2-key.pem
k8s-worker-backend ansible_host=35.178.102.235 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/ec2-key.pem

[all:vars]
ansible_python_interpreter=/usr/bin/python3

```

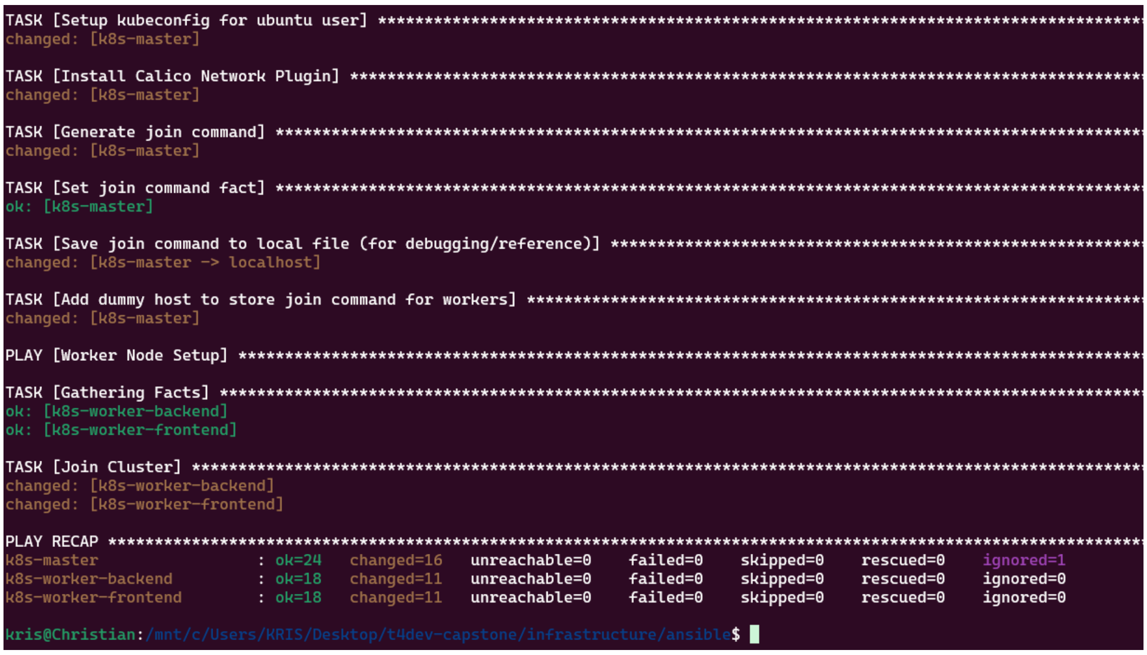
Create a playbook to setup kubernetes cluster on the EC2 instances.

```
---
- name: Kubernetes Common Configuration
  hosts: all
  become: yes
  tasks:
    - name: Disable swap
      shell: |
        swapoff -a
        sed -i '/ swap / s/^/#/' /etc/fstab

    - name: Load kernel modules
      copy:
        content: |
          overlay
          br_netfilter
        dest: /etc/modules-load.d/k8s.conf

    - name: Modprobe modules
      shell: |
        modprobe overlay
        modprobe br_netfilter

    - name: Apply sysctl params
      copy:
        content: |
          net.bridge.bridge-nf-call-iptables  = 1
          net.bridge.bridge-nf-call-ip6tables = 1
          net.ipv4.ip_forward                 = 1
        dest: /etc/sysctl.d/k8s.conf

    - name: Apply sysctl
      command: sysctl --system

    - name: Install dependencies
      apt:
        name:
          - ca-certificates
          - curl
          - gnupg
          - lsb-release
          - apt-transport-https
        state: present
        update_cache: yes

    - name: Add Docker GPG key
      shell: |
        mkdir -p /etc/apt/keyrings
        curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o
/etc/apt/keyrings/docker.gpg --yes
        chmod a+r /etc/apt/keyrings/docker.gpg

    - name: Add Docker repository
      shell: |
        echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

```
    - name: Install containerd
      apt:
        name: containerd.io
        state: present
        update_cache: yes

    - name: Configure containerd
      shell: |
        mkdir -p /etc/containerd
        containerd config default | tee /etc/containerd/config.toml
        sed -i 's/SystemdCgroup = false/SystemdCgroup = true/' /etc/containerd/config.toml
      notify: restart containerd

    - name: Add Kubernetes GPG key
      shell: |
        curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | gpg --dearmor -o
/etc/apt/keyrings/kubernetes-apt-keyring.gpg --yes

    - name: Add Kubernetes repository
      shell: |
        echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /' | tee /etc/apt/sources.list.d/kubernetes.list

    - name: Install Kubernetes components
      apt:
        name:
          - kubelet
          - kubeadm
          - kubectl
        state: present
        update_cache: yes

    - name: Hold Kubernetes packages
      shell: apt-mark hold kubelet kubeadm kubectl

  handlers:
    - name: restart containerd
      service:
        name: containerd
        state: restarted
        enabled: yes

- name: Master Node Setup
  hosts: masters
  become: yes
  tasks:
    - name: Initialize Kubernetes Control Plane
      command: kubeadm init --pod-network-cidr=192.168.0.0/16
      register: kubeadm_output
      ignore_errors: yes # Ignore if already initialized

    - name: Setup kubeconfig for ubuntu user
```

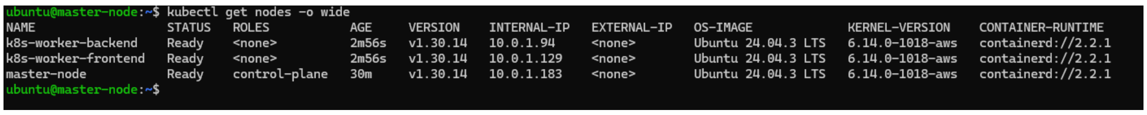
```
      shell: |
        mkdir -p /home/ubuntu/.kube
        cp /etc/kubernetes/admin.conf /home/ubuntu/.kube/config
        chown $(id -u ubuntu):$(id -g ubuntu) /home/ubuntu/.kube/config

    - name: Install Calico Network Plugin
      command: kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml --
kubeconfig=/etc/kubernetes/admin.conf

    - name: Generate join command
      command: kubeadm token create --print-join-command
      register: join_command_raw

    - name: Set join command fact
      set_fact:
        join_command: "{{ join_command_raw.stdout }}"

    - name: Save join command to local file (for debugging/reference)
      local_action: copy content="{{ join_command_raw.stdout }}" dest=./join_command.sh
      become: no

    - name: Add dummy host to store join command for workers
      add_host:
        name: "K8S_TOKEN_HOLDER"
        join_command: "{{ join_command_raw.stdout }}"

- name: Worker Node Setup
  hosts: workers
  become: yes
  tasks:
    - name: Join Cluster
      shell: "{{ hostvars['K8S_TOKEN_HOLDER']['join_command'] }}"
      args:
        executable: /bin/bash
      when: hostvars['K8S_TOKEN_HOLDER'] is defined
```

2. Run the playbook `ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook -i inventory.ini playbook.yml`



```
TASK [Setup kubeconfig for ubuntu user] ******************************************************************************
changed: [k8s-master]

TASK [Install Calico Network Plugin] *********************************************************************************
changed: [k8s-master]

TASK [Generate join command] *****************************************************************************************
changed: [k8s-master]

TASK [Set join command fact] *****************************************************************************************
ok: [k8s-master]

TASK [Save join command to local file (for debugging/reference)] *****************************************************
changed: [k8s-master -> localhost]

TASK [Add dummy host to store join command for workers] **************************************************************
changed: [k8s-master]

PLAY [Worker Node Setup] *********************************************************************************************

TASK [Gathering Facts] ***********************************************************************************************
ok: [k8s-worker-backend]
ok: [k8s-worker-frontend]

TASK [Join Cluster] **************************************************************************************************
changed: [k8s-worker-backend]
changed: [k8s-worker-frontend]

PLAY RECAP ***********************************************************************************************************
k8s-master                 : ok=24   changed=16   unreachable=0    failed=0    skipped=0    rescued=0    ignored=1
k8s-worker-backend         : ok=18   changed=11   unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
k8s-worker-frontend        : ok=18   changed=11   unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

kris@Christian:/mnt/c/Users/KRIS/Desktop/t4dev-capstone/infrastructure/ansible$
```

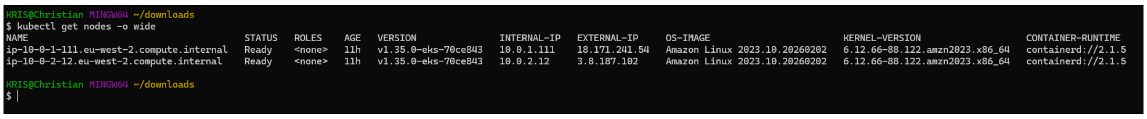
3. Check the status of the cluster created in ec2 `kubectl get nodes`



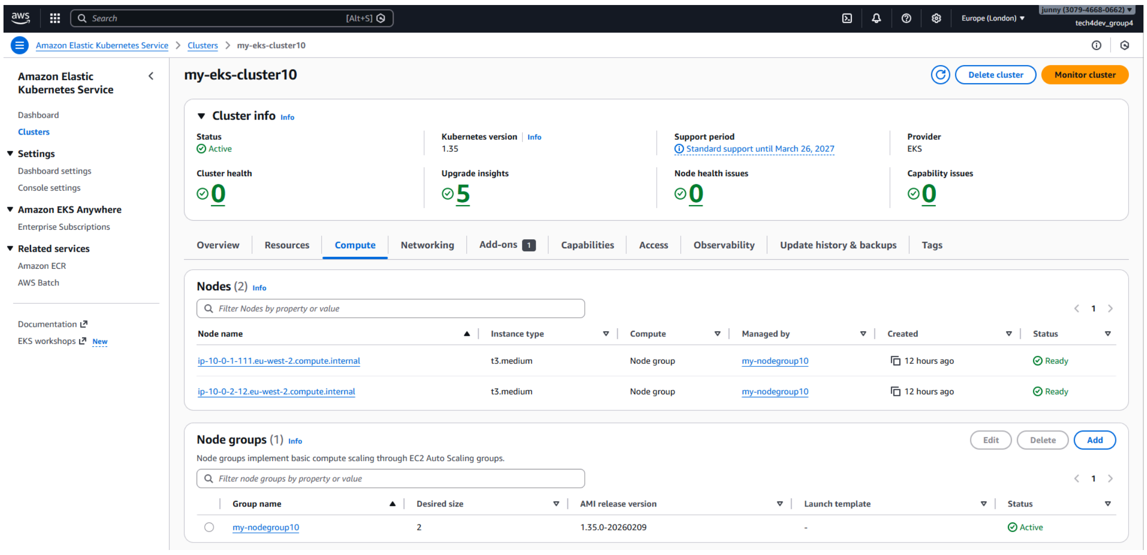
```
ubuntu@master-node:~$ kubectl get nodes -o wide
NAME                  STATUS   ROLES           AGE     VERSION    INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION    CONTAINER-RUNTIME
k8s-worker-backend    Ready    <none>          2m56s   v1.30.14   10.0.1.94     <none>        Ubuntu 24.04.3 LTS   6.14.0-1018-aws   containerd://2.2.1
k8s-worker-frontend   Ready    <none>          2m56s   v1.30.14   10.0.1.129    <none>        Ubuntu 24.04.3 LTS   6.14.0-1018-aws   containerd://2.2.1
master-node           Ready    control-plane   30m     v1.30.14   10.0.1.183    <none>        Ubuntu 24.04.3 LTS   6.14.0-1018-aws   containerd://2.2.1
ubuntu@master-node:~$
```

NOTE: I terminate the EC2 instance and use AWS EKS Cluster

4. AWS EKS Cluster



```
KRIS@Christian MINGW64 ~/downloads
$ kubectl get nodes -o wide
NAME                                       STATUS   ROLES    AGE   VERSION               INTERNAL-IP   EXTERNAL-IP     OS-IMAGE                         KERNEL-VERSION                    CONTAINER-RUNTIME
ip-10-0-1-111.eu-west-2.compute.internal   Ready    <none>   11h   v1.35.0-eks-70ce843   10.0.1.111    18.171.241.54   Amazon Linux 2023.10.20260202   6.12.66-88.122.amzn2023.x86_64   containerd://2.1.5
ip-10-0-2-12.eu-west-2.compute.internal    Ready    <none>   11h   v1.35.0-eks-70ce843   10.0.2.12     3.8.187.102     Amazon Linux 2023.10.20260202   6.12.66-88.122.amzn2023.x86_64   containerd://2.1.5

KRIS@Christian MINGW64 ~/downloads
$ |
```


my-eks-cluster10

Cluster info

Status	Kubernetes version	Support period	Provider
Active	1.35	Standard support until March 26, 2027	EKS

Cluster health	Upgrade insights	Node health issues	Capability issues
0	5	0	0

Nodes (2)

Node name	Instance type	Compute	Managed by	Created	Status
ip-10-0-1-111.eu-west-2.compute.internal	t3.medium	Node group	my-nodegroup10	12 hours ago	Ready
ip-10-0-2-12.eu-west-2.compute.internal	t3.medium	Node group	my-nodegroup10	12 hours ago	Ready

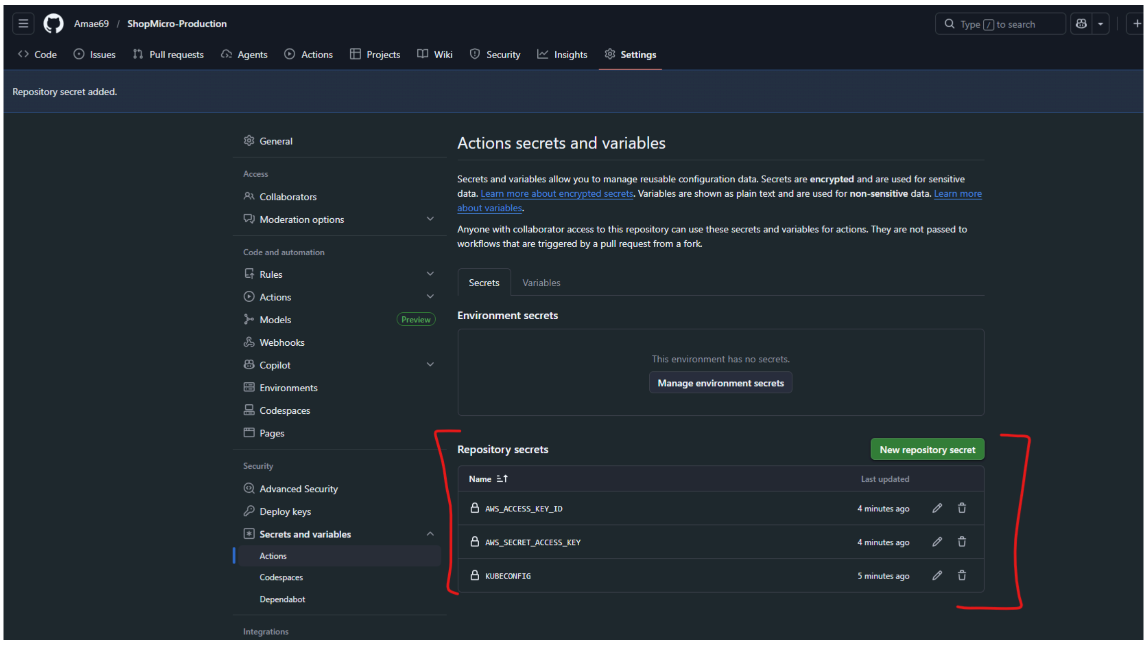
Node groups (1)

Node groups implement basic compute scaling through EC2 Auto Scaling groups.

Group name	Desired size	AMI release version	Launch template	Status
my-nodegroup10	2	1.35.0-20260209	-	Active

On my GitHub, I created a repository named ShopMicro-Production and added my kube congig data and AWS access key and secret key as secrets in the repository settings.

On ks8 master node i run `cat ~/.kube/config` to get the kube config data and paste it in the repository settings secrets.


Amae69 / ShopMicro-Production

Repository secret added.

Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. Learn more about encrypted secrets. Variables are shown as plain text and are used for **non-sensitive** data. Learn more about variables.

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Environment secrets

This environment has no secrets.

Repository secrets

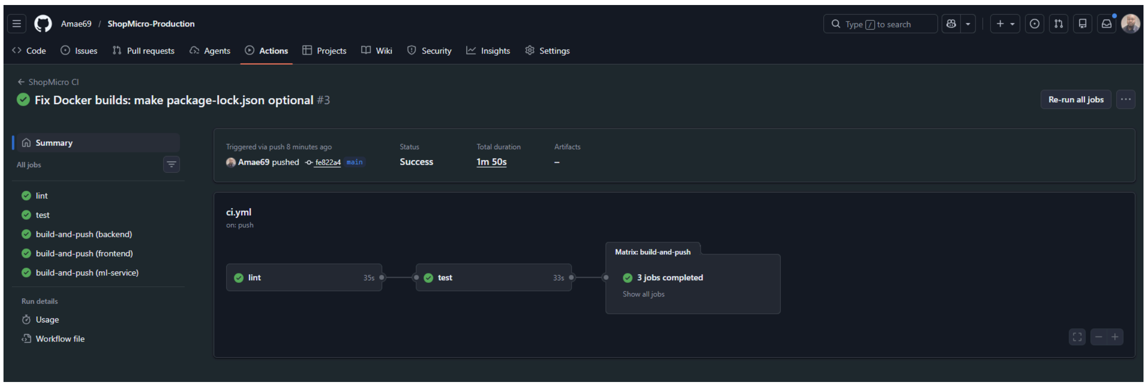
Name	Last updated
AWS_ACCESS_KEY_ID	4 minutes ago
AWS_SECRET_ACCESS_KEY	4 minutes ago
KUBECONFIG	5 minutes ago

CI/CD Pipeline Implementation

I implemented a complete GitHub Actions workflow suite:

- **ci.yml:**

Triggers on Push/PR. Runs linting (Node/Python), Unit Tests, and builds/pushes Docker images to GHCR.



Fix Docker builds: make package-lock.json optional #3

Triggered via push 8 minutes ago
Amae69 pushed fe822a4 main

Status
Success

Total duration
1m 50s

Artifacts
–

- lint
- test
- build-and-push (backend)
- build-and-push (frontend)
- build-and-push (ml-service)

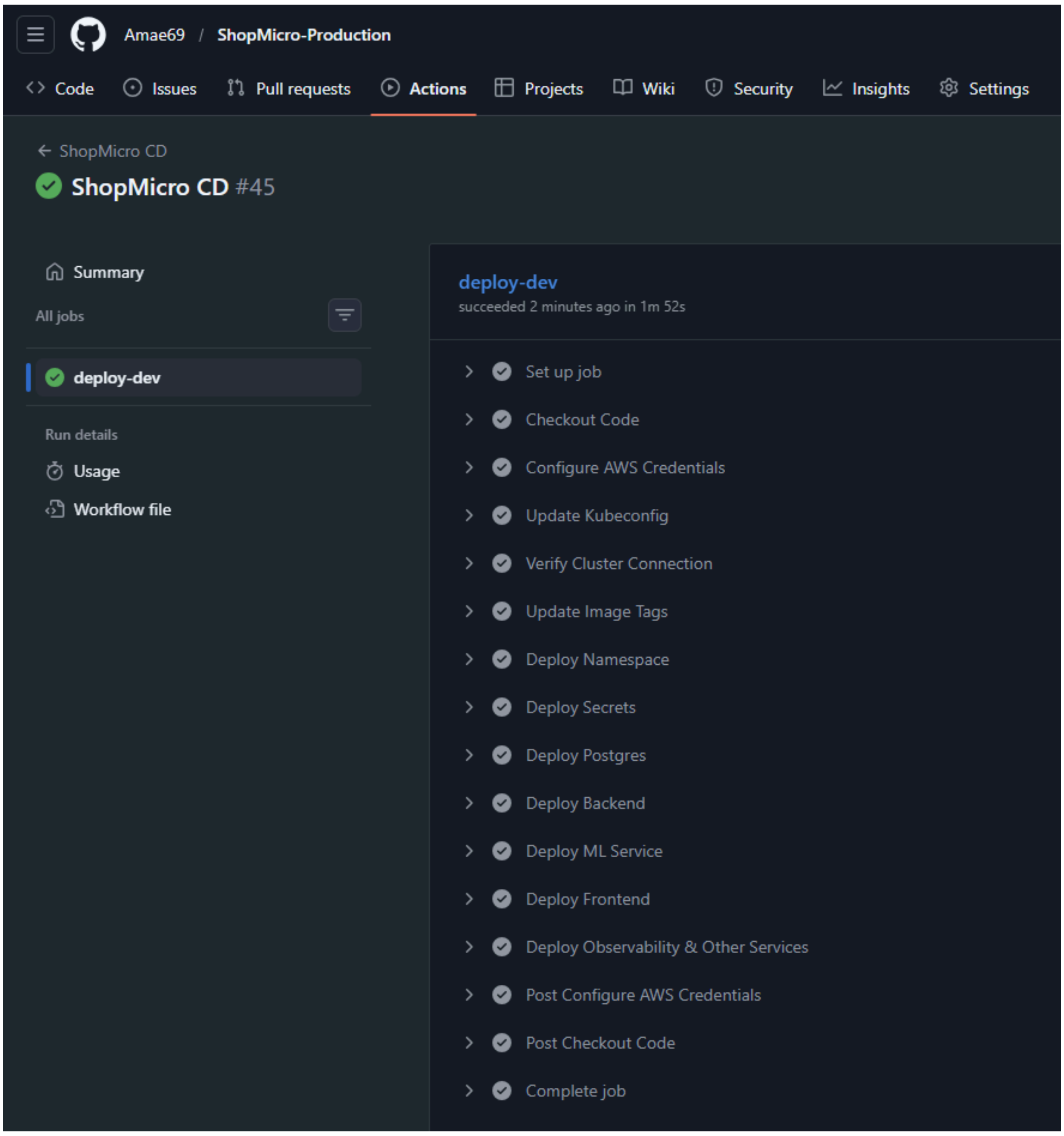
Run details
- Usage
- Workflow file

ci.yml
on: push

lint 35s
test 33s
Matrix: build-and-push
3 jobs completed
Show all jobs

- **cd.yml:**

Triggers after CI success on main. Deploys the application to my Kubernetes cluster.



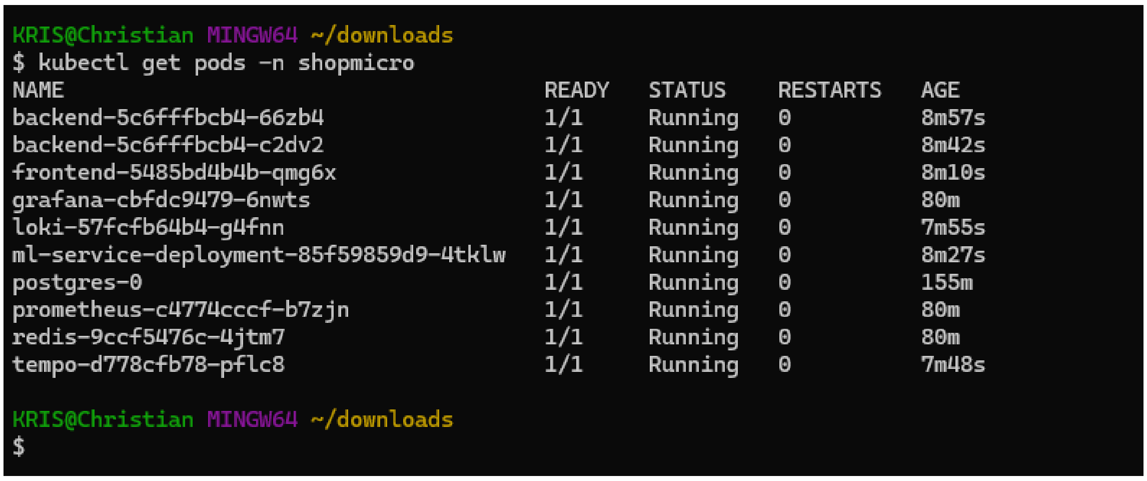
ShopMicro CD #45

deploy-dev

deploy-dev
succeeded 2 minutes ago in 1m 52s

- Set up job
- Checkout Code
- Configure AWS Credentials
- Update Kubeconfig
- Verify Cluster Connection
- Update Image Tags
- Deploy Namespace
- Deploy Secrets
- Deploy Postgres
- Deploy Backend
- Deploy ML Service
- Deploy Frontend
- Deploy Observability & Other Services
- Post Configure AWS Credentials
- Post Checkout Code
- Complete job

`kubectl get pod -n shopmicro` to check running pod on cluster



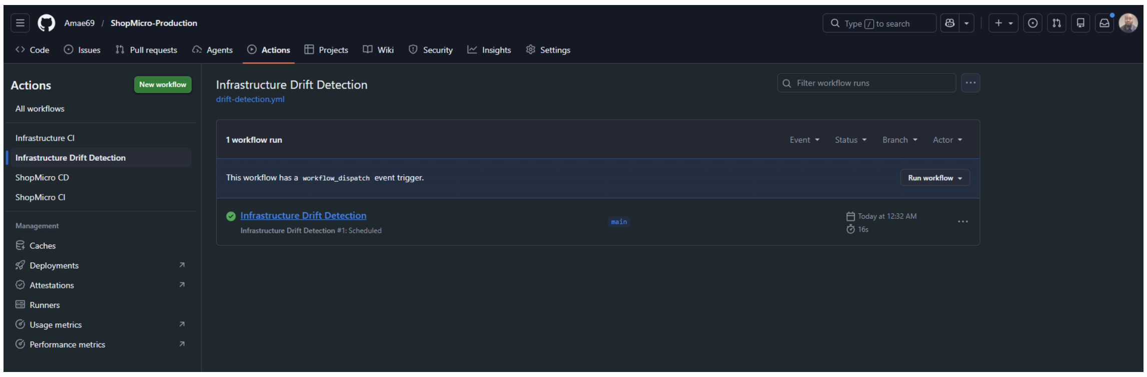
```
KRIS@Christian MINGW64 ~/downloads
$ kubectl get pods -n shopmicro
NAME                                       READY   STATUS    RESTARTS   AGE
backend-5c6fffbcb4-66zb4                   1/1     Running   0          8m57s
backend-5c6fffbcb4-c2dv2                   1/1     Running   0          8m42s
frontend-5485bd4b4b-qmg6x                  1/1     Running   0          8m10s
grafana-cbfdc9479-6nwts                    1/1     Running   0          80m
loki-57fcfb64b4-g4fnn                      1/1     Running   0          7m55s
ml-service-deployment-85f59859d9-4tklw     1/1     Running   0          8m27s
postgres-0                                 1/1     Running   0          155m
prometheus-c4774cccf-b7zjn                 1/1     Running   0          80m
redis-9ccf5476c-4jtm7                      1/1     Running   0          80m
tempo-d778cfb78-pflc8                      1/1     Running   0          7m48s

KRIS@Christian MINGW64 ~/downloads
$
```

- **iac-ci.yml:**

Runs terraform fmt, terraform validate, tflint, and OPA policy checks on infrastructure changes.

- **drift-detection.yml:**

Runs daily at 8am to check for infrastructure drift. Status: ✅ Implemented. Requires Secrets setup in GitHub.



Infrastructure Drift Detection
drift-detection.yml

1 workflow run

This workflow has a `workflow_dispatch` event trigger.

Infrastructure Drift Detection
Infrastructure Drift Detection #1: Scheduled
main
Today at 12:32 AM
16s

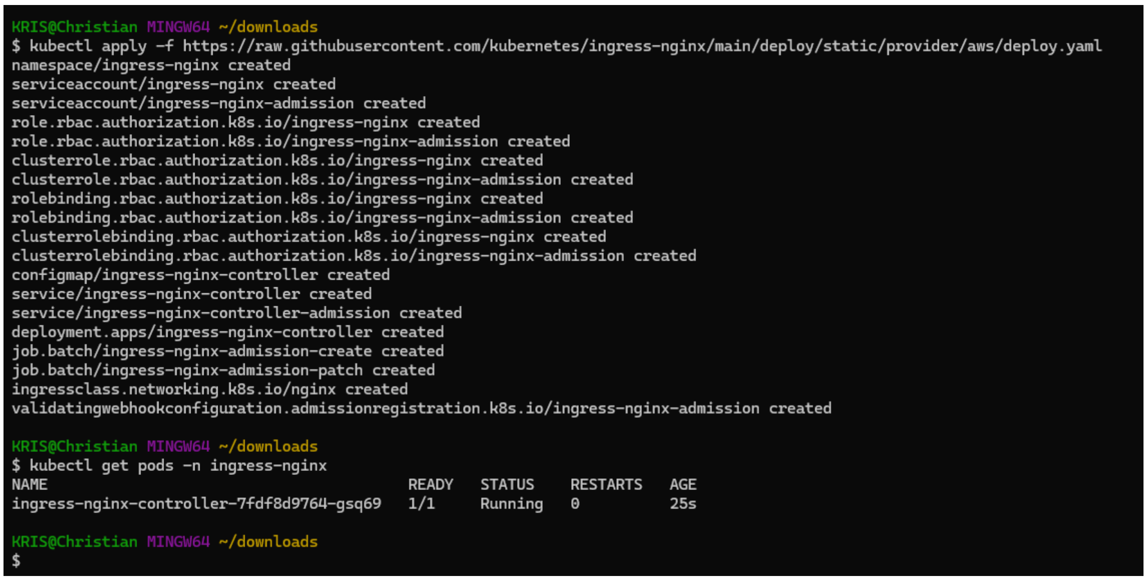
Configure ingress

Install NGINX Ingress Controller:

The Ingress resource **(k8s/ingress.yaml)** requires a controller to work. Install it on EKS

`kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/main/deploy/static/provider/aws/deploy.yaml`

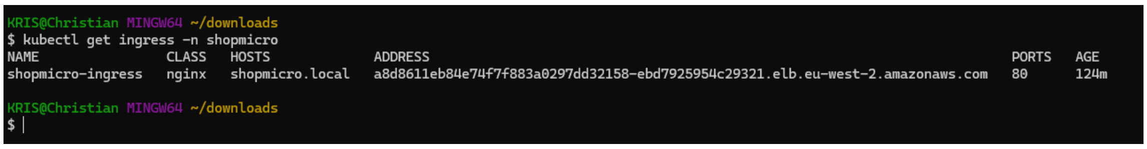
confirm: `kubectl get pods -n ingress-nginx`



```
KRIS@Christian MINGW64 ~/downloads
$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/main/deploy/static/provider/aws/deploy.yaml
namespace/ingress-nginx created
serviceaccount/ingress-nginx created
serviceaccount/ingress-nginx-admission created
role.rbac.authorization.k8s.io/ingress-nginx created
role.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrole.rbac.authorization.k8s.io/ingress-nginx created
clusterrole.rbac.authorization.k8s.io/ingress-nginx-admission created
rolebinding.rbac.authorization.k8s.io/ingress-nginx created
rolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
configmap/ingress-nginx-controller created
service/ingress-nginx-controller created
service/ingress-nginx-controller-admission created
deployment.apps/ingress-nginx-controller created
job.batch/ingress-nginx-admission-create created
job.batch/ingress-nginx-admission-patch created
ingressclass.networking.k8s.io/nginx created
validatingwebhookconfiguration.admissionregistration.k8s.io/ingress-nginx-admission created

KRIS@Christian MINGW64 ~/downloads
$ kubectl get pods -n ingress-nginx
NAME                                        READY   STATUS    RESTARTS   AGE
ingress-nginx-controller-7fdf8d9764-gsq69   1/1     Running   0          25s

KRIS@Christian MINGW64 ~/downloads
$
```

Apply Ingress Resource: Apply the ingress.yaml file to expose the application externally.

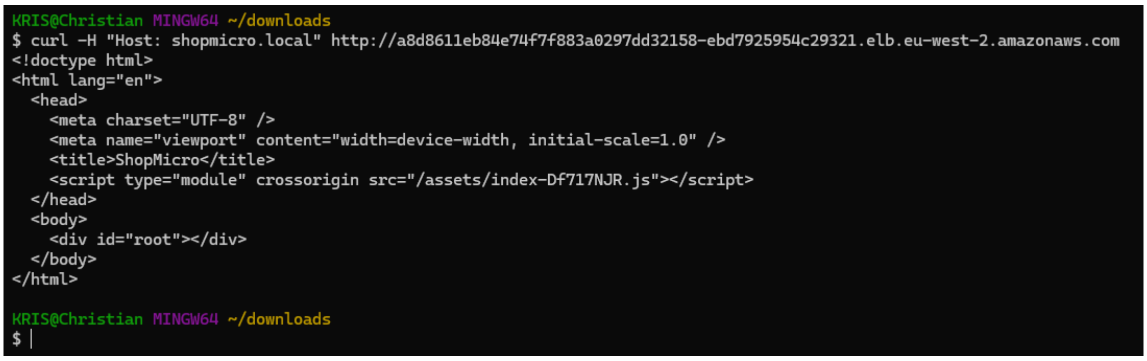
confirm: `kubectl get ingress -n shopmicro`



```
KRIS@Christian MINGW64 ~/downloads
$ kubectl get ingress -n shopmicro
NAME                CLASS   HOSTS             ADDRESS                                                                   PORTS   AGE
shopmicro-ingress   nginx   shopmicro.local   a8d8611eb84e74f7f883a0297dd32158-ebd7925954c29321.elb.eu-west-2.amazonaws.com   80      124m

KRIS@Christian MINGW64 ~/downloads
$ |
```

Test the frontend is working through ingress:

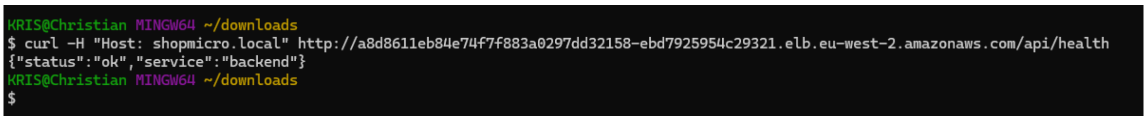
`curl -H "Host: shopmicro.local" http://a8d8611eb84e74f7f883a0297dd32158-ebd7925954c29321.elb.eu-west-2.amazonaws.com`



```
KRIS@Christian MINGW64 ~/downloads
$ curl -H "Host: shopmicro.local" http://a8d8611eb84e74f7f883a0297dd32158-ebd7925954c29321.elb.eu-west-2.amazonaws.com
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>ShopMicro</title>
    <script type="module" crossorigin src="/assets/index-Df717NJR.js"></script>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>

KRIS@Christian MINGW64 ~/downloads
$ |
```

Test the backend is working through ingress

`curl -H "Host: shopmicro.local" http://a8d8611eb84e74f7f883a0297dd32158-ebd7925954c29321.elb.eu-west-2.amazonaws.com/api/health`



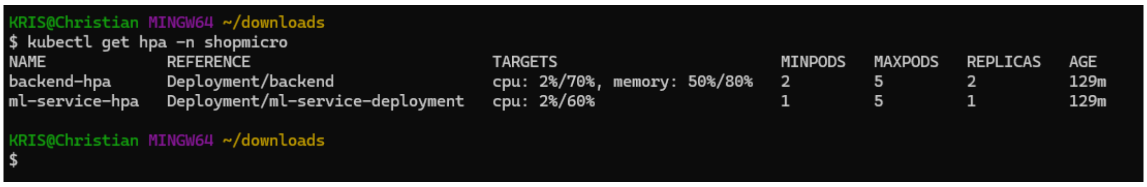
```
KRIS@Christian MINGW64 ~/downloads
$ curl -H "Host: shopmicro.local" http://a8d8611eb84e74f7f883a0297dd32158-ebd7925954c29321.elb.eu-west-2.amazonaws.com/api/health
{"status":"ok","service":"backend"}
KRIS@Christian MINGW64 ~/downloads
$
```

Horizontal Pod Auto Scaling:

The HPA is now deployed. Check its status (requires `metrics-server`):

NOTE: if the cpu, or memory threshold is reached, pod will automatically scaled till it get to its set maxpods

```
kubectl get hpa -n shopmicro
```



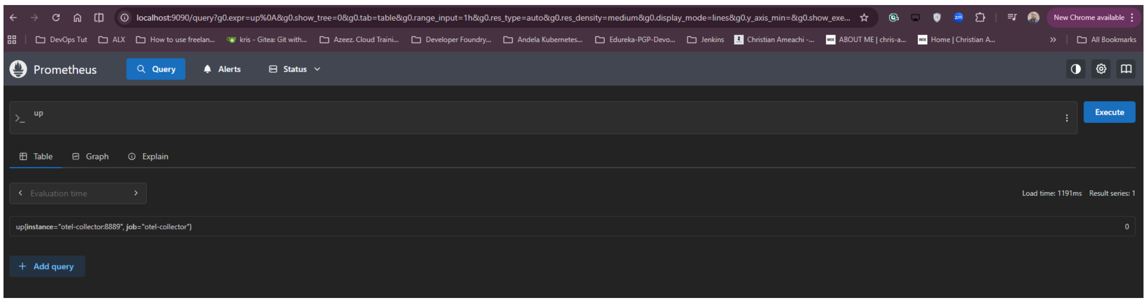
```
KRIS@Christian MINGW64 ~/downloads
$ kubectl get hpa -n shopmicro
NAME             REFERENCE                          TARGETS                        MINPODS   MAXPODS   REPLICAS   AGE
backend-hpa      Deployment/backend                 cpu: 2%/70%, memory: 50%/80%   2         5         2          129m
ml-service-hpa   Deployment/ml-service-deployment   cpu: 2%/60%                    1         5         1          129m

KRIS@Christian MINGW64 ~/downloads
$
```

Monitoring (Prometheus & Grafana)

Using portforwarding to test prometheus and grafan access on the web, given that their services are clusterIP and can only be access internally

Portforward prometheus: `kubectl port-forward svc/prometheus 9090:9090 -n shopmicro`

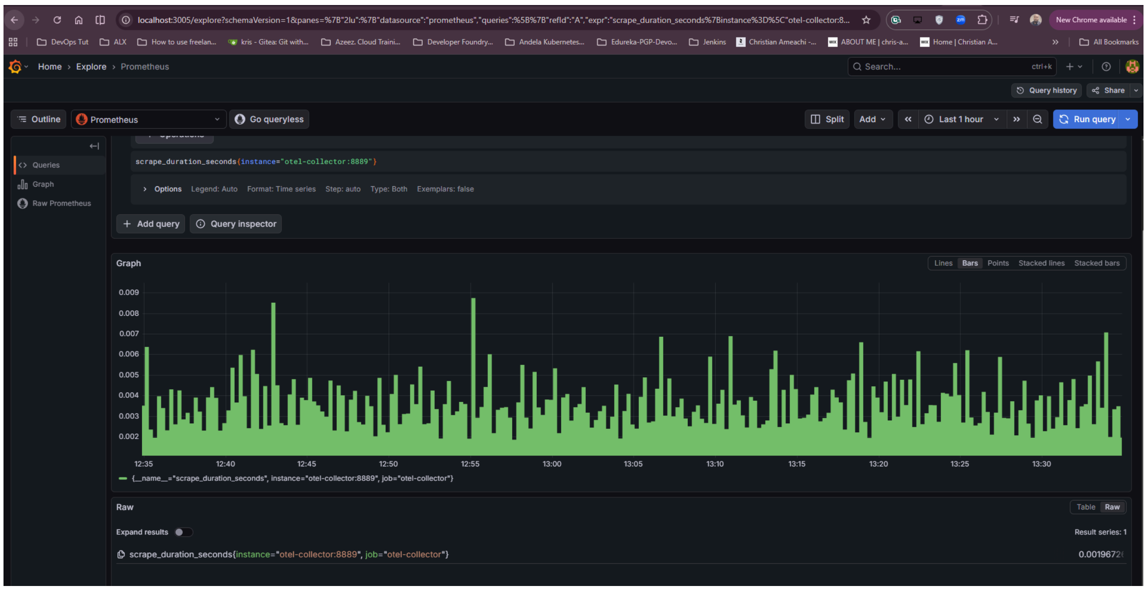
On browser: http://localhost:9090



```
localhost:9090/query?g0.expr=up%0A&g0.show_tree=0&g0.tab=table&g0.range_input=1h&g0.res_type=auto&g0.res_density=medium&g0.display_mode=lines&g0.y_axis_min=&g0.show_exe...

up

Load time: 1191ms   Result series: 1
up{instance="otel-collector:8889", job="otel-collector"}   0
```

Portforward grafana: `kubectl port-forward svc/grafana 3005:3000 -n shopmicro`

On browser: http://localhost:3005



```
scrape_duration_seconds{instance="otel-collector:8889"}
```

Options Legend: Auto Format: Time series Step: auto Type: Both Exemplars: false

{__name__="scrape_duration_seconds", instance="otel-collector:8889", job="otel-collector"}

Expand results

Result series: 1

```
scrape_duration_seconds{instance="otel-collector:8889", job="otel-collector"}    0.0019672
```

Rollback Proof Demonstration

To demonstrate a successful rollback, I performed the following steps to simulate a failed deployment and recover:

1. **Deploy a "Broken" Version**:

   Manually update the backend to use a non-existent image tag. This will trigger a rolling update that fails (ImagePullBackOff).

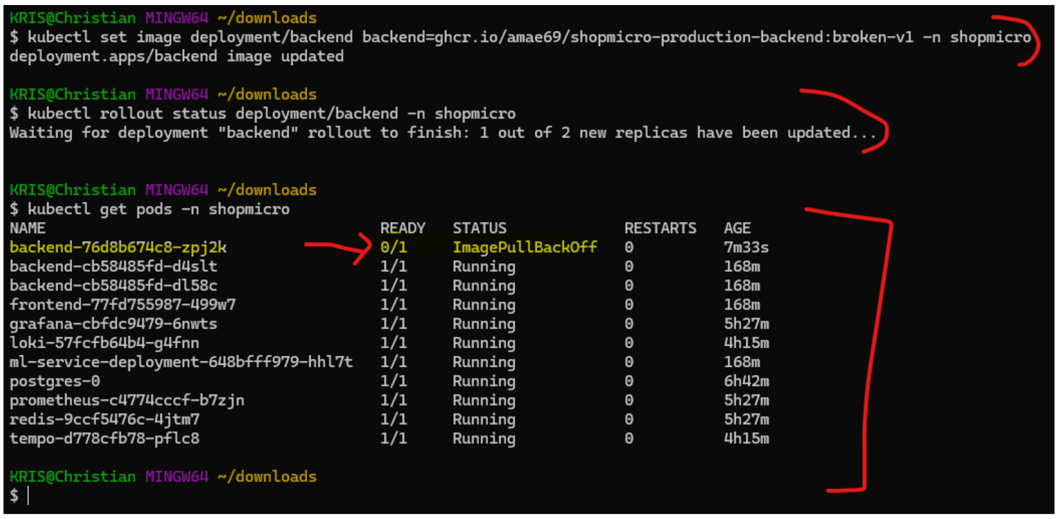
   ```
   kubectl set image deployment/backend backend=ghcr.io/amae69/shopmicro-production-
   backend:broken-v1 -n shopmicro
   ```

2. **Observe the Failure**:

   Check the rollout status. You will see it hang or show errors.

   ```
   kubectl rollout status deployment/backend -n shopmicro
   # Pods will likely show 'ImagePullBackOff'
   kubectl get pods -n shopmicro
   ```

   Screenshot showing "Evidence of Failure".



```
KRIS@Christian MINGW64 ~/downloads
$ kubectl set image deployment/backend backend=ghcr.io/amae69/shopmicro-production-backend:broken-v1 -n shopmicro
deployment.apps/backend image updated

KRIS@Christian MINGW64 ~/downloads
$ kubectl rollout status deployment/backend -n shopmicro
Waiting for deployment "backend" rollout to finish: 1 out of 2 new replicas have been updated...


KRIS@Christian MINGW64 ~/downloads
$ kubectl get pods -n shopmicro
NAME                                      READY   STATUS             RESTARTS   AGE
backend-76d8b674c8-zpj2k                  0/1     ImagePullBackOff   0          7m33s
backend-cb58485fd-d4slt                   1/1     Running            0          168m
backend-cb58485fd-dl58c                   1/1     Running            0          168m
frontend-77fd755987-499w7                 1/1     Running            0          168m
grafana-cbfdc9479-6nwts                   1/1     Running            0          5h27m
loki-57fcfb64b4-g4fnn                     1/1     Running            0          4h15m
ml-service-deployment-648bfff979-hhl7t    1/1     Running            0          168m
postgres-0                                1/1     Running            0          6h42m
prometheus-c4774cccf-b7zjn                1/1     Running            0          5h27m
redis-9ccf5476c-4jtm7                     1/1     Running            0          5h27m
tempo-d778cfb78-pflc8                     1/1     Running            0          4h15m

KRIS@Christian MINGW64 ~/downloads
$ |
```

3. **Perform the Rollback**:

   Use the `undo` command to revert to the last stable state.

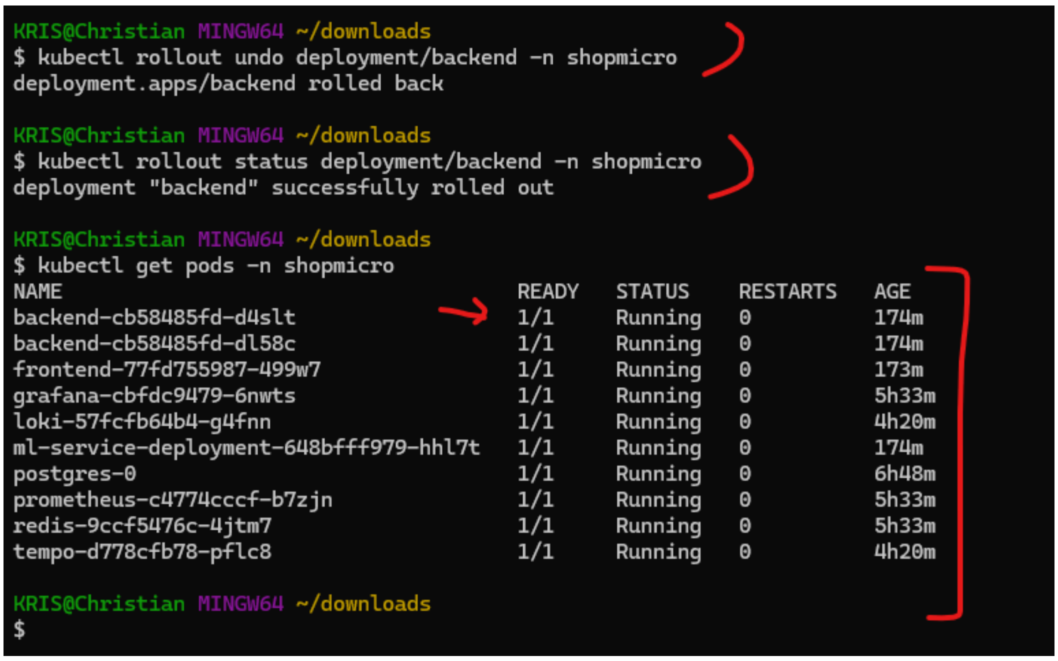
   ```
   kubectl rollout undo deployment/backend -n shopmicro
   ```

4. **Verify Restoration**:

   Confirm the deployment is back to a healthy state.

   ```
   kubectl rollout status deployment/backend -n shopmicro
   kubectl get pods -n shopmicro
   ```

   Screenshot showing "Rollback Proof".



```
KRIS@Christian MINGW64 ~/downloads
$ kubectl rollout undo deployment/backend -n shopmicro
deployment.apps/backend rolled back

KRIS@Christian MINGW64 ~/downloads
$ kubectl rollout status deployment/backend -n shopmicro
deployment "backend" successfully rolled out

KRIS@Christian MINGW64 ~/downloads
$ kubectl get pods -n shopmicro
NAME                                      READY   STATUS    RESTARTS   AGE
backend-cb58485fd-d4slt                   1/1     Running   0          174m
backend-cb58485fd-dl58c                   1/1     Running   0          174m
frontend-77fd755987-499w7                 1/1     Running   0          173m
grafana-cbfdc9479-6nwts                   1/1     Running   0          5h33m
loki-57fcfb64b4-g4fnn                     1/1     Running   0          4h20m
ml-service-deployment-648bfff979-hhl7t    1/1     Running   0          174m
postgres-0                                1/1     Running   0          6h48m
prometheus-c4774cccf-b7zjn                1/1     Running   0          5h33m
redis-9ccf5476c-4jtm7                     1/1     Running   0          5h33m
tempo-d778cfb78-pflc8                     1/1     Running   0          4h20m

KRIS@Christian MINGW64 ~/downloads
$
```

END ...