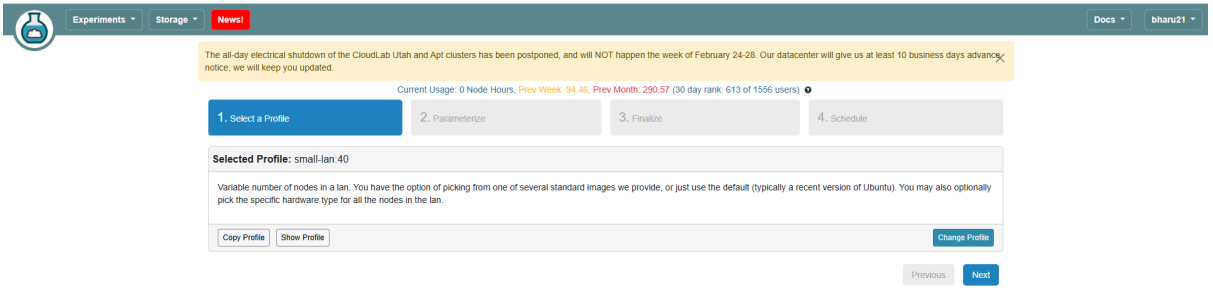
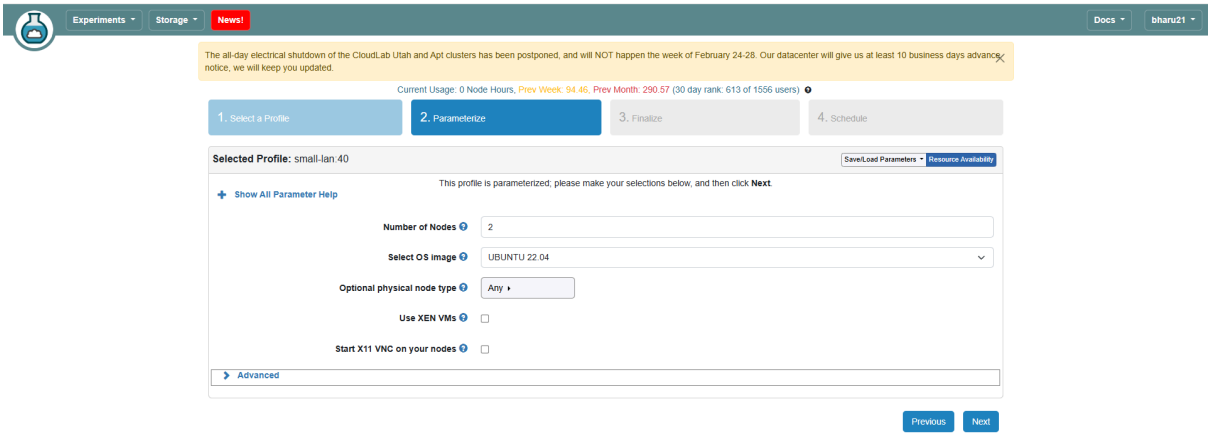
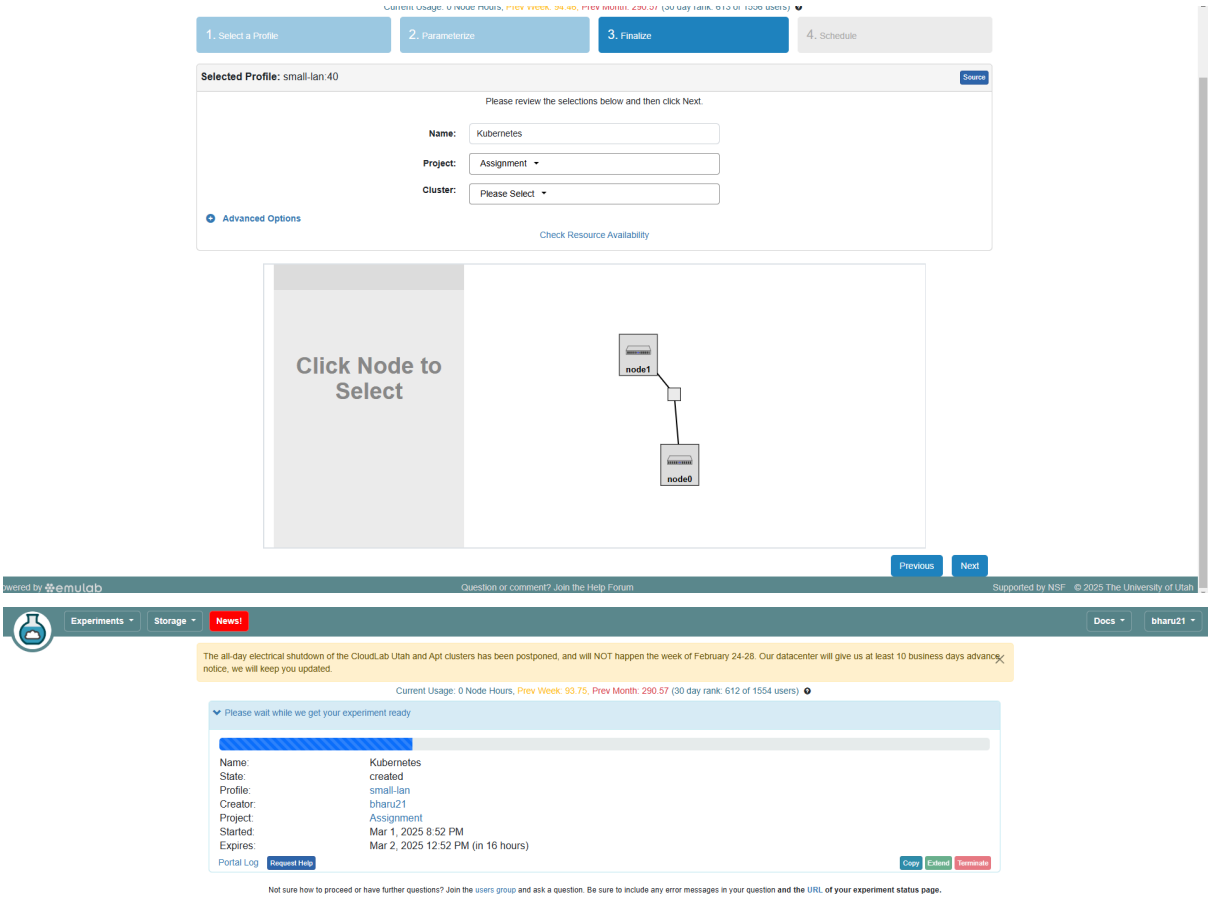
Assignment 5: Container Orchestration

Initial setup:



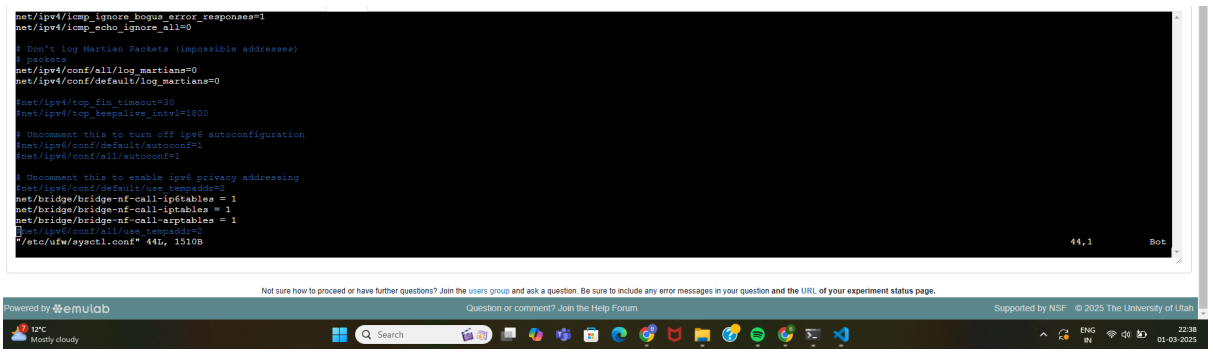




Prepare the VM:

Before setting up Kubernetes, the virtual machines (VMs) need to be **updated** and **configured**.
Each node (master and worker) should have:

- The **container runtime (`containerd`)** installed, which allows running containers.
- **Essential Kubernetes components (`kubeadm`, `kubelet`, and `kubectl`)** installed to create and manage the cluster.



Create a cluster:

```
root@node0:/users/bharu21# sudo rm -rf /etc/apt/keyrings/kubernetes-archive-keyring.gpg
root@node0:/users/bharu21# sudo apt update && sudo apt upgrade -y
Hit:1 http://repos.emulab.net/emulab/ubuntu jammy InRelease
Hit:2 http://repos.emulab.net/grub-backports/ubuntu jammy InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu jammy InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:5 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:6 http://us.archive.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
Get more security updates through Ubuntu Pro with 'esm-apps' enabled:
```

```
root@node0:/users/bharu21# sudo apt install -y apt-transport-https ca-certificates curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203~22.04.1).
curl is already the newest version (7.81.0-1ubuntu1.20).
apt-transport-https is already the newest version (2.4.13).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
root@node0:/users/bharu21# sudo mkdir -p /etc/apt/keyrings
root@node0:/users/bharu21# curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo tee /etc/apt/keyrings/kubernetes-archive-keyring.gpg > /dev/null
root@node0:/users/bharu21# echo "deb [signed-by=/etc/apt/keyrings/kubernetes-archive-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /" | sudo tee /etc/apt/sources.list.d/kuberne
deb [signed-by=/etc/apt/keyrings/kubernetes-archive-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /
root@node0:/users/bharu21# sudo apt update
```

```
root@node0:/users/bharu21# kubeadm init --pod-network-cidr=192.168.0.0/16
I0302 13:11:11.304531   35922 version.go:256] remote version is much newer: v1.32.2; falling back to: stable-1.28
[init] Using Kubernetes version: v1.28.15
[preflight] Running pre-flight checks
        [WARNING Swap]: swap is enabled; production deployments should disable swap unless testing the NodeSwap feature gate of the kubelet
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
```

Not sure how to proceed or have further questions? Join the users group and ask a question. Be sure to include any error messages in your question and the URL of your experiment status page.

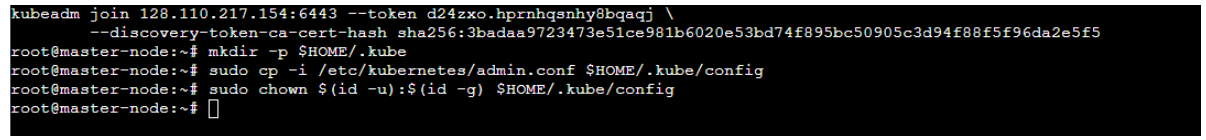
```
root@node0:/users/bharu21# sudo systemctl start docker
```

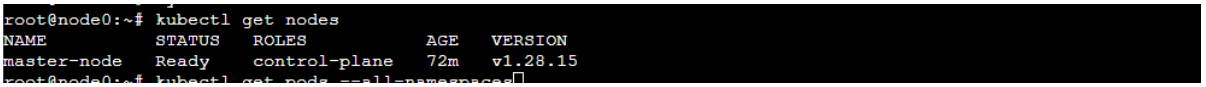
```
root@node1:/users/bharu21# sudo kubeadm init --pod-network-cidr=192.168.0.0/16
I0302 17:52:48.392512    9480 version.go:256] remote version is much newer: v1.32.2; falling back to: stable-1.28
[init] Using Kubernetes version: v1.28.15
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
```

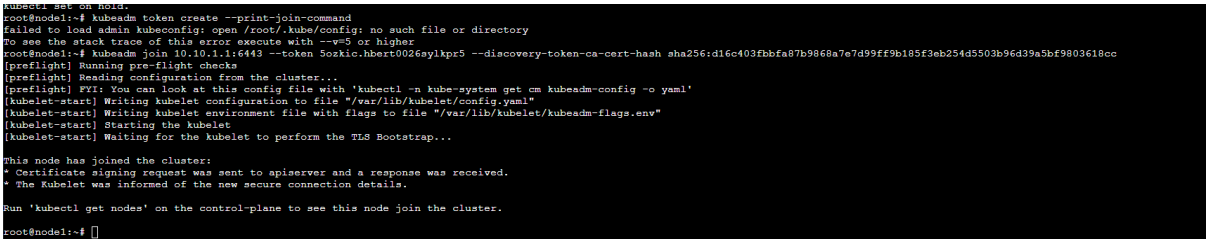
```
Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 128.110.217.154:6443 --token d24zxo.hprnhqsnhy8bqaqj \
        --discovery-token-ca-cert-hash sha256:3badaa9723473e51ce981b6020e53bd74f895bc50905c3d94f88f5f96da2e5f5
root@master-node:~#
```

Starting the cluster:

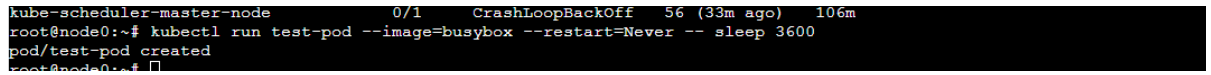
The master node generates a `join` command, which the worker node executes to become part of the cluster.
Once joined, the worker node must appear in `kubectl get nodes` as `Ready`.
Kubernetes requires a Container Network Interface (CNI) plugin for pod-to-pod communication.
Options include Calico (recommended) or Flannel.
The CNI plugin allows worker nodes to send and receive traffic between pods.

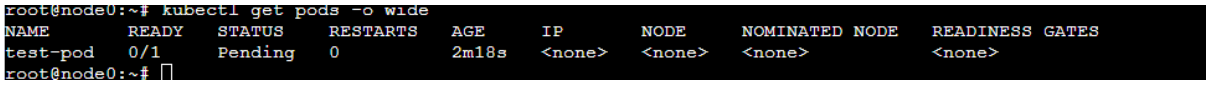






Deploy test pod:
A simple test pod is created to verify if the worker node can schedule and run workloads.
If the pod starts on the worker node, it confirms that the cluster is functioning correctly.





2. Deployment.yml of NGNIX

A Deployment is created to manage two replicas of Nginx.
The Deployment ensures high availability, meaning if a pod fails, Kubernetes automatically replaces it.
The `deployment.yaml` file defines:

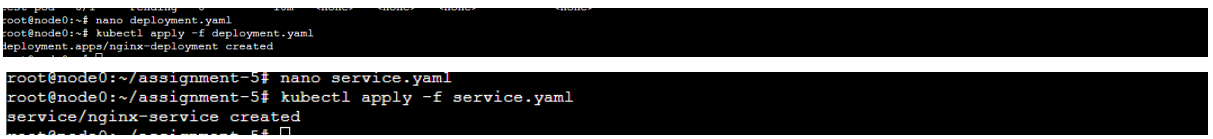
- The number of replicas (`2`).
- The Nginx image to be used.
- The port (`80`) on which Nginx runs.

Deployment.yml:

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

Service.yml

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



```
service/nginx-service created
root@node0:~/assignment-5# kubectl get svc
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP      10.96.0.1       <none>        443/TCP        152m
nginx-service   LoadBalancer   10.99.116.65    <pending>     80:31536/TCP   32s
root@node0:~/assignment-5#
```