****

**PNH DEVOPS CKA 2023**

***Học viện PNH:*** [***www.pnh.vn***](http://www.pnh.vn) ***(Zalo: 0906289618 – 8A Tôn Thất Thuyết – Hà nội)***

|  |  |
| --- | --- |
| **Kubernetes CKA Q & A** | |
| 1 | #busy.yaml  apiVersion: v1  kind: Pod  metadata:  creationTimestamp: null  labels:  run: busybox  name: busybox-myapp  namespace: app-team1  spec:  serviceAccount: cicd-token  containers:  - image: radial/busyboxplus  name: busybox  command: ["sleep", "48000"]  resources: {}  dnsPolicy: ClusterFirst  restartPolicy: Always  status: {}  kubectl apply -f busy.yaml  kubectl create clusterrole deployment-clusterrole --verb=create,get --resource=deployments,statefulsets,daemonsets,pods  kubectl create serviceaccount cicd-token --namespace=app-team1  kubectl create rolebinding deployment-clusterrole --clusterrole=deployment-clusterrole --serviceaccount=app-team1:cicd-token --namespace=app-team1  kubectl exec -it busybox-myapp sh -n app-team1  KUBE\_TOKEN=$(cat /var/run/secrets/kubernetes.io/serviceaccount/token)  curl -sSk -H "Authorization: Bearer $KUBE\_TOKEN" \  https://$KUBERNETES\_SERVICE\_HOST:$KUBERNETES\_PORT\_443\_TCP\_PORT/api/v1/namespaces/app-team1/pods/ |
| 2 |  |
| 3 | root@master:/home/ubuntu# kubectl describe ns myapp  Name: myapp  Labels: kubernetes.io/metadata.name=myapp  Annotations: <none>  Status: Active  No resource quota.  No LimitRange resource.  root@master:/home/ubuntu# cat network.yaml  apiVersion: networking.k8s.io/v1  kind: NetworkPolicy  metadata:  name: allow-port-from-namespace  namespace: echo  spec:  podSelector: {}  policyTypes:  - Ingress  ingress:  - from:  - namespaceSelector:  matchLabels:  # sua tai day  kubernetes.io/metadata.name: myapp  - podSelector: {}  ports:  - protocol: TCP  port: 9000  #Tạo pod busy box trong namespace myapp  apiVersion: v1  kind: Pod  metadata:  creationTimestamp: null  labels:  run: busybox  name: busybox-myapp  namespace: myapp  spec:  serviceAccount: cicd-token  containers:  - image: radial/busyboxplus  name: busybox  command: ["sleep", "48000"]  resources: {}  dnsPolicy: ClusterFirst  restartPolicy: Always  status: {}  #Tạo pod nginx trong namespace echo , sửa cấu hình nginx , trong /etc/nginx/config.conf listen sang port 9000  kubectl exec -it busybox-myapp -n myapp sh  curl http://IP-nginx:9000 |
| 4 | Cài CSI driver Plugin Pod:  **git clone https://github.com/kubernetes-csi/csi-driver-host-path/**  Sau khi clone về thì chạy lệnh:  **./csi-driver-host-path/deploy/kubernetes-latest/deply.sh**  Sau khi đợi cài xong, kiểm tra: **kubectl get pod** mà thấy đã chạy đủ 8 containers trong Pod là OK    Tạo storageClaass: **nano** **storageclass.yml**  **apiVersion: storage.k8s.io/v1**  **kind: StorageClass**  **metadata:**  **name: csi-hostpath-sc**  **provisioner: hostpath.csi.k8s.io**  **reclaimPolicy: Delete**  **volumeBindingMode: Immediate**  **allowVolumeExpansion: true**  chạy lệnh: **kubectl apply -f storageclass.yml**  kiểm tra: **kubectl get sc**  Chạy lệnh sửa PVC  **k edit pvc pv-volume --record**    Tham khảo thêm:  https://kubernetes.io/docs/tasks/configure-pod-container/configure-persistent-volume-storage/#create-a-persistentvolume |
| 5 |  |
| 6 |  |
| 7 | Create a nginx pod with label env=test in engineering namespace  **Answer**:  Step 1: Tạo namespace nếu chưa tồn tại:  kubectl create namespace  engineering  Step 2:Tạo file yaml của pod  kubectl run nginx --image=nginx --restart=Never --labels=env=test --namespace=engineering --dry-run -o  yaml > nginx-pod.yaml  Step 3: khởi tọa pod  kubectl create -f nginx-pod.yaml |
| 8 | Given a partially-functioning Kubernetes cluster, identify symptoms of failure on the cluster. Determine the node, the failing service, and take actions to bring up the failed service and restore the health of the cluster. Ensure that any changes are made permanently.  Answer:  Step 1: Xác định nguyên nhân lỗi  Theo bài cụm k8s có thể có các node **bk8s-master-0** và **bk8s-node-0**, có thể phần config của các node này không đúng  Step 2:  Sau khi ssh vào node ssh <nodename> chuyển sang user root kiểm tra file cấu hình bằng lệnh vi /var/lib/kubelet/config.yml thường sẽ kiểm tra thông tin dòng **staticPodPath** sửa giá trị của biến này về /etc/kubernetes/manifests sau đo restart lại kubelet service systemctl restart kubelet |
| 9 | Create a pod with image nginx called nginx and allow traffic on port 80  Answer:  kubectl run nginx --image=nginx --restart=Never --port=80 |
| 10 | From the pod label name=cpu-utilizer, find pods running high CPU workloads and write the name of the  pod consuming most CPU to the file /opt/KUTR00102/KUTR00102.txt (which already exists).  Đề yêu cầu tìm pod ăn nhiều CPU nhất trong những pod có label name=cpu-utilizer và ghi vào file /opt/KUTR00102/KUTR00102.txt  Cách 1  kubectl top pods -l name=cpu-utilizer  (top để hiện thị dung lượng CPU và MEMORY nếu có cài metric server, -l để chọn hiển thị pod có label như mong muốn)  https://lh4.googleusercontent.com/v2gvws5WjYSrE45SIdyddZqSDHUsBBKIJybXKiSiRrJF2l4rHgBPw9-RkH1pG96cgAtNwIqdmYv_53IGG-Vaqae8UA9XwHfzprXzklmSzMNoVpqxtGMf7M3ZurtzNhLdhwDmllxh317FkWfR  Cop tên pod ăn nhiều cpu và mem nhất vào file /opt/KUTR00102/KUTR00102.txt |
| 11 | List all the pods sorted by created timestamp  Liệt kê danh sách các pod được sắp xếp theo mục “created timestamp”  kubectl get pods --sort-by=.metadata.creationTimestamp |
| 12 | Create a Kubernetes secret as follows:  Name: super-secret  password: bob  Create a pod named pod-secrets-via-file, using the redis Image, which mounts a secret named  super-secret at /secrets.  Create a second pod named pod-secrets-via-env, using the redis Image, which exports password as CONFIDENTIAL    Tạo một secret có thông tin như sau  name: super-secret  password: bob  kubectl create secret generic super-secret --from-literal=password=bob  Tạo một pod có tên là  “pod-secrets-via-file” dùng image “redis” và mount ở  “/secret”vào secret có tên là “super-secret”  apiVersion: v1  kind: Pod  metadata:    name: pod-secrets-via-file  spec:    containers:    - name: redis      image: redis      volumeMounts:      - name: secret-volume        mountPath: /secrets    volumes:    - name: secret-volume      secret:        secretName: super-secret  Tạo một pod thứ 2 có tên là “ pod-secrets-via-env” dùng image “redis” có tác dụng dùng biến giá trị trong secret như một biến môi trường có tên là CONFIDENTIAL lấy giá trị từ key “password” trong secret  apiVersion: v1  kind: Pod  metadata:    name: pod-secrets-via-env  spec:    containers:    - name: redis      image: redis      env:      - name: CONFIDENTIAL        valueFrom:          secretKeyRef:            name: super-secret            key: password |
| 13 | Print pod name and start time to “/opt/pod-status” file  $ sudo -i  # kubectl get pod -o=jsonpath='{range .items[\*]}{"Pod Name: "}{.metadata.name}{"\nStart Time: "}{.status.startTime}{"\n\n"}{end}' > /opt/pod-status  # cat /opt/pod-status  Pod Name: webserver  Start Time: 2023-06-07T07:22:20Z |
| 14 | Create a persistent volume with name app-data, of capacity 2Gi and access mode ReadWriteMany. The type of volume is hostPath and its location is /srv/app-data.  B1. create a yaml persistent-volume.yaml:  $ nano persistent-volume.yaml  apiVersion: v1  kind: PersistentVolume  metadata:    name: app-data  spec:    capacity:      storage: 2Gi    accessModes:      - ReadWriteMany    hostPath:      path: /srv/app-data  B2. you can create the persistent volume by running the following command:  kubectl apply -f persistent-volume.yaml  B4: Check pv  $ kubectl get pv  https://lh6.googleusercontent.com/5OBNBRMs_7nSAWKU6EWqTVEJYY03yeH1NXnCQEg_XILO5imT0D8CQ024VKXuD1s8F6c2JghlwgRElFao9CCq-KvBo7CQmv-JqVaSd1FlNmLKNkaqbDYnp1nHQu8qjNapeBkE361VNMuiN0Qo |
| 15 |  |
| 16 | Create a busybox pod and add add “sleep 3600” command  kubectl run busybox-pod --image=busybox --command -- sleep 3600  Kiểm tra :  kubectl exec -it busybox-pod -- sh  ps -ef | grep sleep |
| 17 | Create a pod named kucc8 with a single app container for each of the following images running inside (there may be between 1 and 4 images specified): nginx + redis + memcached  apiVersion: v1  kind: Pod  metadata:    name: kucc8  spec:    containers:    - name: nginx      image: nginx    - name: redis      image: redis    - name: memcached      image: memcached  Kiểm tra :  kubectl describe pod kucc8 |
| 18 | Set the node named ek8s-node-1 as unavailable and reschedule all the pods running on it  kubectl drain worker2 --ignore-daemonsets --force    Trước khi drain  https://lh5.googleusercontent.com/r9WgmjOA9oWXxnkZGopQxW3JUDBEAH8MigbbhxcGh087bK5jau8fFD-y9UVm-_2ZfVp3BEGsGRiHgVeA3bmAozesLH1RcAx95OC-BJ5TVkp_C2qE5Hsc5qKHLcKTW41Ps4ehC9TlDurj-5RI  Sau khi drain  https://lh4.googleusercontent.com/r4n6qFtxnGxymZQEeI6UcHBpYZPd7MwZWhs-KpMc40OBSHsnTG2Rpt835PJ7LYPOf7x4fQ5fHDFvQ062u2i-CZtkTGYWx44r4PC-pEVg1ZV3d-mB0MT679GlQpZKSSOUVrgjnnVAf8xFIFwH |
| 19 | Ensure a single instance of pod nginx is running on each node of the Kubernetes cluster where nginx also represents the Image name which has to be used.  Do not override any taints currently in place.  Use DaemonSet to complete this task and use ds-kusc00201 as DaemonSet name     1. Tạo file ds.yaml   apiVersion: apps/v1  kind: DaemonSet  metadata:    name: ds-kusc00201  spec:    selector:      matchLabels:        app: nginx    template:      metadata:        labels:          app: nginx      spec:        containers:        - name: nginx          image: nginx   1. Gõ lệnh: kubectl apply -f ds.yaml   https://lh3.googleusercontent.com/NrWTn0jH3Ri39maUBFOobS_guykIF42i0qLRPM7Sgf5lKisqRhQg_F5EBEqPYdZj6HV3y9SAkL0MvjtDHCTOAHnSZgWjWeO04QgUIQdEKokiOTB0RmXTjccL8TKPVFm-9YrappC3mJNlVVpW  => Daemonset sẽ tạo trên mỗi node 1 pod chạy nginx. |
| 20 | Score:7%  Task Create a new nginx Ingress resource as follows:  • Name: ping  • Namespace: ing-internal  • Exposing service hi on path /hi using service port 5678     1. Tạo môi trường namespace, service, pod: ***nano init.yaml***   *apiVersion: v1*  *kind: Namespace*  *metadata:*  *name: ing-internal*  *---*  *apiVersion: v1*  *kind: Pod*  *metadata:*  *name: nginx-pod*  *namespace: ing-internal*  *labels:*  *app: nginx*  *spec:*  *containers:*  *- name: nginx*  *image: nginx:latest*  *ports:*  *- containerPort: 80*  *---*  *apiVersion: v1*  *kind: Service*  *metadata:*  *name: hi*  *namespace: ing-internal*  *spec:*  *selector:*  *app: nginx*  *ports:*  *- protocol: TCP*  *port: 5678*  *targetPort: 80*  => *kubectl apply -f init.yaml*     1. Tạo ingress; ***nano ingress.yaml***   apiVersion: networking.k8s.io/v1  kind: Ingress  metadata:    name: ping    namespace: ing-internal  spec:    rules:      - host: test.vn        http:          paths:            - path: /hi              pathType: Prefix              backend:                service:                  name: hi                  port:                    number: 5678  *=> kubectl apply -f ingress.yaml*  https://lh5.googleusercontent.com/Kt3D7pkKgaEt2sutTHEyWtuOjj9odui2eaRHYCzgYZvHQORu3uxL7-VxEGkbfA_d_TV82-NrbBgaGjO2OANNFHzE9kniMqVBNs3iAn-YoC-GztfaFx41rWqMQgjVRgLd7wT2ye3TGl2Fh_gY |
| 21 | Scale the deployment webserver to 6 pods     1. tạo deployment webserver ***nano deployment.yaml***   apiVersion: apps/v1  kind: Deployment  metadata:    name: webserver  spec:    replicas: 1    selector:      matchLabels:        app: nginx    template:      metadata:        labels:          app: nginx      spec:        containers:          - name: nginx            image: nginx:latest            ports:              - containerPort: 80  ***=> kubectl apply -f deployment.yaml***     1. Scale deployment thành 6 pods   ***kubectl scale deploy webserver --replicas=6***  https://lh4.googleusercontent.com/vk0nKh6jg02jiICjDcquT1F-pApLOdVecouOpWoQ0DsSsRoglyuW9KeSSYfNVO79dGfL7-IeQT51gTvVGqbF2_ubjcscZEpNkQ5NIjBl0wr8TVH6Kx4vEX4WpY2nTudeYb-gf9NlxKrgh55k |
| 22 | **Check the Image version of nginx-dev pod using jsonpath**  $ kubectl get pod nginx-dev -o jsonpath='{.spec.containers[].image}{"\n"}'  **https://lh6.googleusercontent.com/Fv0Km2sIRLrJqXlyCE7MxdGaXGEywdHvbiBy5q91YUAE8kMLx3Qfre6QYvNYXaaUMi41L5BYji9Zp-a8g9bk6rxibqqPMGvUkQv424VBK7ltaS2E5ZJ5hB6T3yPGkMeZRtHNV6A3EExzuaBg** |
| 23 | **Create a deployment as follows:**  **=> Name: nginx-random**  **=> Exposed via a service nginx-random**  **=> Ensure that the service & pod are accessible via their respective DNS records**  **=> The container(s) within any pod(s) running as a part of this deployment should use the nginx Image.**  **=> Next, use the utility nslookup to look up the DNS records of the service & pod and write the output to */opt/KUNW00601/service.dns* and */opt/KUNW00601/pod.dns* respectively.**  $ kubectl create deploy nginx-random --image=nginx  $ kubectl expose deploy nginx-random --name=nginx-random --port=80 --target-port=80  $ kubectl create -f dns.yaml  $ kubectl get pod -o wide | grep nginx-random  $ kubectl exec -it busybox1 -- nslookup nginx-random  $ kubectl exec -it busybox1 -- nslookup nginx-random > /opt/KUNW00601/service.dns  $ kubectl exec -it busybox1 -- nslookup 172-20-5-4.default.pod  $ kubectl exec -it busybox1 -- nslookup 172-20-5-4.default.pod > /opt/KUNW00601/pod.dns  **https://lh6.googleusercontent.com/4w6FC4IEruCmmxEaUGnZkI9-ia2nj6PaWMYU7n-0XrH2D99Vdxd0zc9dvJykN3RV48N2ah1oKLXkuqBYjEf_dDxjrLl9MPc4nWc9S9TZ_VWhUnT69m-ZbvAhw3ARGzklVpkL6DQVkY50M3z0**  **File dns.yaml**  apiVersion: v1  kind: Pod  metadata:    name: busybox1    labels:      name: busybox  spec:    containers:      - image: busybox:1.28        command:          - sleep          - "3600"        name: busybox |
| 24 | **Scale the deployment “web server” (replace by nginx-random) to 3 pods.**  $ kubectl get deployment  $ kubectl scale deployment nginx-random --replicas=3  **https://lh4.googleusercontent.com/czWWFZDa-y_ay5lA2onU6NA0qSGseE6rkm2wFIf1xETU2Ij66W1RjznJlISx-us7fNTonN6w0dLbsEcOxaUCutLLfY7Didzt31z7xsYyWurso2Ok2-zb079wiXoeVZN5YB51vvB1L4r6pSlF** |
| 25 | List all the pods sorted by name  **Giải:**  kubectl get pods --help  kubectl get pods -A --sort-by=.metadata.name  https://lh4.googleusercontent.com/5QMofh9dka-3DGG23wEUiZ6Wq6wVjFZk11TkLnPHndTEgCcocxVel6iDiE6ctuggxQLG3NCcFtoc8QY7QBF4TPuqDDQNpwQiEqgEe1mdC6P9cqXQD7DNrvx3Ndw4Xbcq-Q2JmN88ARs7IZMF |
| 26 | Create a pod named kucc8 with a single app container for each of the following images running inside(there may be between 1 and 4 images specified): nginx + redis + memcached.  **Giải:**  vim kucc8-pod.yaml   |  | | --- | | apiVersion: v1  kind: Pod  metadata:    name: kucc8  spec:    containers:    - name: nginx      image: nginx    - name: redis      image: redis    - name: memcached      image: memcached |   kubectl apply -f kucc8-pod.yaml |
| 27 | Create 2 nginx image pods in which one of them is labelled with env=prod and another one labelled with env=dev and verify the same  **Giải:**  vim nginx-pod.yaml   |  | | --- | | apiVersion: v1  kind: Pod  metadata:    name: nginx-prod    labels:      env: prod  spec:    containers:    - name: nginx      image: nginx  ---  apiVersion: v1  kind: Pod  metadata:    name: nginx-dev    labels:      env: dev  spec:    containers:    - name: nginx      image: nginx |   kubectl apply -f nginx-pod.yaml  ===verify===  kubectl get pod --show-labels=true  kubectl get pod --show-labels=true -l env=prod  kubectl get pod --show-labels=true -l env=dev  https://lh4.googleusercontent.com/z8aWBIiEx0Iod3y9swK5E6O9Bct1BAuFDkOy1CkVzLXqfgGenk1o2Z0s_wBkrRV-A4X21_5i_6WZuduH2iWiEIcl-mTpq7J5qPsBCCfQJaX2vPXhZP2E7lttB3R2HysMQYzC43THV9t90Yhe |
| 28 | Create a pod that echo “hello world” and then exists. Have the pod deleted automatically when it’s completed  **Solution** |
| 29 | : For this item, you will have to ssh to the nodes ik8s-master-0 and ik8s-node-0 and complete all tasks on these nodes. Ensure that you return to the base node (hostname: node-1) when you have completed this item. Context As an administrator of a small development team, you have been asked to set up a Kubernetes cluster to test the viability of a new application.  **Solution**  **Remote SSH ik8s-master-0 sau đó thực hiện như sau:**  1, sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --apiserver-advertise-address=all --apiserver-advertise-address=IP-master  2, mkdir -p $HOME/.kube  3,sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  4, sudo chown $(id -u):$(id -g) $HOME/.kube/config  5, curl<https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/calico.yaml> -O  6. Làm theo hướng dẫn giống trong ảnh dưới    7, kubectl apply -f calico.yaml  **SSH ik8s-node-0: Thực hiện lệnh join**  sudo kubeadm join 10.0.2.15:6443 --token bno0o9.rgo7vx4oa8vdyj7v --discovery-token-ca-cert-hash sha256:b359377b4ef93d23e44c39832f9e49220b2c21eb501bafd40e52a1discovery-token-ca-cert-hash sha256:b359377b4ef93d23e44c39832f9e49220b2c21eb501bafd4520e52a1ef253ed8f6 |
| 30 | A Kubernetes worker node, named wk8s-node-0 is in state NotReady. Investigate why this is the case, and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made permanent.  **Solution**  **1, Check available resource**  Kubectl describe node wk8s-node-0  **2, Checking the kubelet process**  systemctl status kubelet  systemctl restart kubelet  **3, Checking the certificate:**  Openssl x509 -in /var/lib/kubelet/pki/kubelet.crt -text  **4, checking the kube-proxy pod**  Kubectl get pod -n kube-system |
| 31 |  |
| 32 |  |
| 33 |  |
| 34 | .kubectl get pod -A -o jsonpath='{range .items[\*]}{.metadata.name}{"\t"}{.metadata.namespace}{"\n"}{end}' |
| 35 | .kubectl get pods --namespace backend1 -o jsonpath='{.items[\*].spec.containers[\*].image}{"\n"}'  kubectl get pods --all-namespaces -o jsonpath="{.items[\*].spec.containers[\*].image}" |
| 36 | apiVersion: v1  kind: Pod  metadata:    name: "nginx-kusc00401"  spec:    nodeSelector:      disktype: ssd    containers:    - name: nginx      image: "nginx"      imagePullPolicy: IfNotPresent |
| 37 | Create an nginx pod and list the pod with different levels of verbosity  # Tạo mới một pod  **root@master:~# kubectl run nginx --image=nginx --restart=Always --port=80**  https://lh5.googleusercontent.com/Gn1kDTnMPNpOVb8cB3DC3Ay5Vs74tYGCF-cOMrC0_HtCOLRg_zJ9Fln6EkF3aym-ohPeto3wnLou-egSm23KMc-m_aX2AalrUnxEbzzw1txZRzJPZmsvip2UQ90x9FzAq_2iMuhHEWQdXpuR  **root@master:~# kubectl get pod -A**  https://lh5.googleusercontent.com/8k8tkUd8KVpk9mFiq1T4KLrfPo8CrX1X5AFwwZ-Qr6Ok6WFH0DnVx6oc-WqtwxWIuUoWXyN4LEr71w_MwfhDOmH_KhT7ASQbWu8rFBcOITV7em7ybs70vgJGt9l7Oro5WALcPwCaDeC8N1bp  # Liệt kê danh sách pod với các mức độ chi tiết khác nhau  **root@master:~# kubectl get pod nginx --v=7**  **root@master:~# kubectl get pod nginx --v=8**  **root@master:~# kubectl get pod nginx --v=9**  https://lh5.googleusercontent.com/IgP315CbL9m8CGOCImOdE3l_84fTT1BY1zmMupom2DN43rsHmOYMyVcuP7MuwS6z1eGRUXawVOzr3pDec-xFpaSXwEapDRPyCXYtuzl94MuVX3GuCG4iOB2rIAvk3Z35tms91Ee5IKocMUnQ |
| 38 | Check to see how many worker nodes are ready (not including nodes tainted NoSchedule) and write the number to /opt/KUCC00104/kucc00104.txt  # Kiểm tra danh sách nodes  **root@master:/opt/KUCC00104# kubectl get nodes -o wide**  https://lh3.googleusercontent.com/L11A4hTaL23rpW7Srfd8stJpOm07uCsSQTUXx1StT7d4Qso9SML18fNGOZ9kv2io_rKx2JAStoknj8Dnb-SGyntNjX_bdy-jrbR79g94_omA65QqyUBYyvjghb7OjuEZS-pVS61zHjX22hUW  # Nhập số node vào file /opt/KUCC00104/kucc00104.txt  **root@master:/opt/KUCC00104# echo 1 > kucc00104.txt**  **root@master:/opt/KUCC00104# cat kucc00104.txt**  https://lh4.googleusercontent.com/Yfz6emUsx8zDU3YPH9Ykm62gA3GLm_REaMkGMmkThkDly0tiD2-hvqKIKwb4c0-yPPrq2aCIaomZm_4DQDFfrq--JrIMcXY7BVuypE2A9y5nSucMvmyrWUIMEZrtSM6pwXlKYLYSs3g3RYyr |
| 39 | Schedule a Pod as follows:  https://lh5.googleusercontent.com/9Oi9j_1trpj2Cu3Nr3YpgXxh656p1gVN_RFYGJyOvqtlDdd-n--Ul6hn0D9Xg1MGIbls0MWYfuieghmHyN2C-aPVLFC6U1vHhI0gio7WwdOGkIyi1ykZFmdZ7m9lmuvTQydqPAFpIH0xL1yC   * Name: kucc1 * App Containers: 2 * Container Name/Images: nginx & consul   Do tài nguyên chỉ có 1 master & 1 worker, không chuẩn bị base-node nên tự tạo context mới & ký số + phân quyền ngay trên master node  # Tạo một context mới  **root@master:~/cert# kubectl config set-context thanhdt --cluster=kubernetes --user=thanhdt**  # Kiểm tra lại  **root@master:~/cert# kubectl config get-contexts**    https://lh4.googleusercontent.com/x8ULwN0Dh3kHNeBuTLIeFrj9m4a6VbdawoLfsNiSs8zVXqTGnjx73eqbkI9G_WUHfJikauwXaAE5B2nX9NtS1qxjEqzMMu7J2RcgFMzMGcNKYIZV7xrOKYnBjXRc-yPfaFJI8tW-qkRVZxTx  # Tạo Role bằng file yaml  https://lh5.googleusercontent.com/agKf_W_Ppj9z9LeZWJjGBlBzx0dpgF28e3Jf4lpOpJNMIjhYBaN1Mq1Jh_eP6NcHrAD53EtOs4Ev8Qv7V3e1lzflqBrPiDGqrrcP0BnfjBgRIbqByKVAgZF_LH1aRWY5CBHjB6mL_Gyei4JY  # Tạo Rolebinding bằng file yaml  https://lh5.googleusercontent.com/3JHr-aRmoZkHwNOhgJ7-ESaSQlmN3z2jRLhvb6Ty0wLJiR5MGnHfjQWvC10Lptddeh5ZWe9L0hBux_BGstSgpfkcBGbWzhtc2FJ53AbSEMPkYvZD7cpu6F8wuZrRZ543_4fAYa9znkDJq-Co  # Kiểm tra lại Role, Rolebinding  **root@master:~/RBAC# kubectl get role,rolebinding -A**  https://lh4.googleusercontent.com/a_g8G--2ocWt5dMJuJWUC7HrQ7vVp09_VU_qvcNZLtmSjK46VXfcz_L5Z-t5NkscT50f1wIISRtWp5x6-rrlRda2Mv6z6u1Y2a297DWBeGWx-TfUPCxjXqwio8qnNmCrVlYtF7oAU6shiKxa  # Kiểm tra rolebindings của một tài khoản và context cụ thể  **root@master:~/cert# kubectl describe rolebindings role-binding**  https://lh3.googleusercontent.com/O291YRZzczvM9hg59KolAfeFPWqr1l8pdL3aLKYBXdDRRwkvjF6lDLMt-8WUOkhCFpqGkBHjQK9zmTOSJStXjVMncA7P-8H9eVGp4Klq9XVQEPCRZnv7kUi-39CNeaGrUyRXZTHoWrnivyll  # Tạo key rsa  **root@master:~/cert# openssl genrsa -out thanhdt.key 2048**  **https://lh4.googleusercontent.com/VZGZMUJLTFbCRPOFMhzeDfEvhrTKRvYTQSBIR3eYCyQUmvVL_4fw4n8EbCP_AdfOlJjs8TFr6flDKxrKwfN1ZvKtesPR2xiX38cQ_IYeyddNBiQBBuH0I6kLTm8WlVDvS5Nf2aUcVjWxRn2_**  # Tạo CSR từ một private key & tạo Cert từ CSR  **root@master:~/cert# openssl req -new -key thanhdt.key -out thanhdt.csr -subj "/CN=thanhdt/O=cmc"**  **root@master:~/cert# openssl x509 -req -in thanhdt.csr -CA /etc/kubernetes/pki/ca.crt -CAkey /etc/kubernetes/pki/ca.key -CAcreateserial -out thanhdt.crt -days 500**  **https://lh6.googleusercontent.com/PTEVAf1DqnDiHOuu-m3j-lnmo9b60_oTd68IQXnKU6micb5zFmpHv9Gb9WSuPVhG_RU37uG0Uy764AfOoIqTv6bSJpBBi1ovovxn9pNBYyuFGUywjAZkzWVSj_C7Vr0GUm7LtO_TfzvEoCHC**  # Thiết lập thông tin chứng thực  **root@master:~/cert# kubectl config set-credentials thanhdt --client-certificate=/root/cert/thanhdt.crt --client-key=/root/cert/thanhdt.key**  **https://lh3.googleusercontent.com/vflOR5s6y-8q401tHqSPjTG7Pty3YPR6V5jwjhSJHTq4pO1RqVApydbIq1AzbSOipplNq5RE0_R0Vuo7dzLpLLsu1SHHc-U2AicTT2Bo85xEX0x6u7ekqtmAesfpUNvET38GXhSZwv25Mry_**  # Kiểm tra lại config  https://lh3.googleusercontent.com/ESPUBCJRLLxnH7dUNNtgh2Eu93fFBgfsd9wDP0aDmTUbcw3dagVBVtlTkAz-yOtrJ_WT_y95Ypf7ShSnLyWwVaVp4_NqeiIZbmsqW-rp5Nxrl9gcMiRgi9dD4E2U1eZOZe5nZyIVbp90cfc_  # Tạo file yaml với nội dung 1 Pod 2 container nginx & consul & apply  **root@master:~/pod# cat thanhpod.yaml**  https://lh5.googleusercontent.com/18ewp55G9ET_eWVsMgMx00QwO16uqWd2TxWhKg-C0bz5E4-gCvdOPsYkg1Aaqna1N5T5bvICt47FO55edtuZ_oiDgR7vnDCSdRknzq54mBnp6hsx2LQ90RA1AkjEKOVcNVfpElcUDicQx6h2  **root@master:~/pod# kubectl --context=thanhdt apply -f thanhpod.yaml**  https://lh4.googleusercontent.com/FtGZ4JSGIUWcy902kimvyJYMXaMa999WbGTnDakDhzVs9bQV6Cg1A-pRRM6HSgXEZZxBmpy_o7Hhdm85ftyoRoRdWDh8yiDZvmYHMqIPiWU76XjKo1XH7HwE0FLg5CUFwF7J0Dl33LkE3KDt  # Kiểm tra lại  **root@master:~/pod# kubectl get pod -A**  https://lh3.googleusercontent.com/ViFE24Qlz76BNC2-4PP0yRD8f5SFBJh3w_wjqgJPADJdjHYjehOvj4aqjy9XjVSDS-1TmHN1n-GfU-xqbSwBZOIfaiwyTYJsxbWe7FRx1Z4exAuRGWB2sIf8g3ctRz1ne7nm_WMsUDWLj8KP |
| 40 | **List the nginx pod with custom columns POD\_NAME and POD\_STATUS**  Đề yêu cầu liệt kê pod nginx với các cột tùy chỉnh POD\_NAME và POD\_STATUS.  **Answer:**  a, liệt kê với các app có label/app: nginx  kubectl get pods -l app=nginx -A -o custom columns=POD\_NAME:.metadata.name,POD\_STATUS:.status.phase  -l : filter những pods nào có label là app=nginx  -o : sử dụng để chỉ định định dạng đầu ra và custom-columns xác định các cột tùy chỉnh cũng như các giá trị tương ứng hiển thị.  https://lh3.googleusercontent.com/zfvSlNLSEYCAVQ1IA9Q1KfIXhCT87Fv31iIAef33pCyowi4CkNIhkn9bsFsOcebLSehDSsUz7Nk5FVKNXglw4EOFpW8cZXfZiln0EFd_RbDWvBRg6lXIlsFSonb7LRF41cqbU73z8QPnwZo  b, liệt kê trực tiếp pod  kubectl get pods  nginx-pod -o custom-columns=POD\_NAME:.metadata.name,POD\_STATUS:.status.phase  https://lh3.googleusercontent.com/LmiuMN5ZL2iNlIo8tYv6b85p94CMtkIF01g3QIasO20LLJTCmnPIignoIJ2MjpRcN8KFK0g5sfX7SgqY7vrTLU1xnGetBFNSrIBjUr09G_JW_2rJHXDKt-o5FRRbXNUSt19s1a0P3N-BzEs |
| 41 | **Get IP address of the pod – “nginx”**  Nhận địa chỉ ip của pod nginx  **Answer:**  a, chỉ nhận ip của pod nginx không hiển thị theo hàng cột  kubectl get pod nginx -o jsonpath='{.status.podIP}'  https://lh6.googleusercontent.com/fnbgPAuhLE96UTyaV4HXGy27MKgHkeAtUsXFKQ56q-FsZzel7TUpvZ2cNUwIvxuKlOfZUVpBcx0hOtYDdVHVwfTn1Rj93EU47Sv_Pivk4fEPt4H2zeWQ4mCmVh7dKmxH-Mfjmj3XvUHZhCk  B, nhận ip của pod nginx hiển thị theo hàng cột  kubectl get pod nginx -o custom-columns=POD\_NAME:.metadata.name,POD\_IP:.status.podIP  https://lh6.googleusercontent.com/j6iJB0lK_A0Vc9K_Iy63ZLKC4ooT2Ofkj9_y7TIUiBNYWAH444ibEQfk6Oqpi1YiKcm7kqAQGl3lyRKXJtO6KJ1jmMROHfr7Oh4WBz06TDt99y7bG7bhVkwCxatKfgbGjcv6wbca7Qfto-U |
| 42 | **Create a file: /opt/KUCC00302/kucc00302.txt that lists all pods that implement service baz in namespace development. The format of the file should be one pod name per line**  Tạo một tập tin:  /opt/KUCC00302/kucc00302.txt liệt kê tất cả các nhóm triển khai dịch vụ baz trong quá trình phát triển không gian tên. Định dạng của tệp phải là một tên nhóm trên mỗi dòng  **Answer:**  B1: check SELECTOR label của SVC  kubectl get svc -n development -o wide  https://lh3.googleusercontent.com/6NQ5kyzs8--nJS7g1HeAgyN7tOMNi1-MxkHtJVhlQlWr4kas-6a7IgYIzIzdSxN6bT4-P-sFVJLeJgmxOUSVtthCmjtWROEV8XR5_RoDyDZLxgaRlcGPIi6Yfc93EWk7iV_TIJrl08RdEdU  Xác định được svc baz selector label app=nginx  B2: lấy tên các pod ở namespace development có label app=nginx và ghi vào file /opt/KUCC00302/kucc00302.txt  kubectl get pod -l app=nginx -n development -o NAME > /opt/KUCC00302/kucc00302.txt  https://lh6.googleusercontent.com/-OCCLyNKK4mJICdNFkEEUA7PAuGt2ZJyty81d8rpU9hMj3i9vo5hmBdfdwWayQd9psku5_O29c5jnFJ-QSacKkTbIG7es0RpaGcgUdwNzWnwf9qHpXnDuZzwX7imYJ3v38WQ_0dqZClrZOc |
| 43 |  |
| 44 |  |
| 45 |  |
| 46 | **kubectl scale deployment.apps/presentation --replicas=6** |
| 47 | **kubectl create deploy nginx-app --image=nginx:1.11.10-alpine**  **kubectl set image deploy nginx-app nginx=nginx:1.11.13-alpine --record**  **kubectl rollout undo nginx-app** |
| 48 | **kubectl logs foo | grep unable-to-access-webservice > /opt/KULM00201/foo** |
| 49 | From the pod label name=cpu-utilizer, find pods running high CPU workloads and write the name of the pod consuming most CPU to the file /opt/KUTR00401/KUTR00401.txt (which already exists).  ======  => Từ pod có lable cpu-utilizer, tìm các pod đang chạy với tải CPU cao và ghi tên của pod tiêu thụ nhiều CPU nhất vào tệp /opt/KUTR00401/KUTR00401.txt (đã tồn tại)  => Giống câu 10  => Cần cài đặt thêm metric-server  kubectl top pods -l name=cpu-utilizer |
| 50 | Check to see how many nodes are ready (not including nodes tainted NoSchedule ) and write thenumberto /opt/KUSC00402/kusc00402.txt.   * Đếm số lượng các node có trạng thái "ready" hoạt động trong môi trường Kubernetes   kubectl describe nodes | grep ready|wc -l   * Xác định số lượng các node trong môi trường Kubernetes có taint (khoá) với tên là "noschedule" và ghi kết quả đếm đc vào file "/opt/KUSC00402/kusc00402.txt"   kubectl describe nodes | grep -i taint | grep -i noschedule |wc -l > /opt/KUSC00402/kusc00402.tx  https://lh5.googleusercontent.com/Zl-7Afiu6V1AeXP8tYgksTxKP7juQUWGdlrCeTYKuk0MeaALjj2glzxQ4POygE5oX7DIaRxhcVQ3ahiXqAI91P6DKIPpzTNURYCA4z5EJHrLRBR7DCgZlHHKf0QbJqwEJaNWI19d8NAn6bxf |
| 51 | List pod logs named “frontend” and search for the pattern “started” and write it to a file “/opt/error-logs”   * Liệt kê logs của pod có tên "frontend" và tìm kiếm pattern started" và ghi nó vào tệp "/opt/error-logs * kubectl logs frontend | grep -i “started” > /opt/error-logs   https://lh5.googleusercontent.com/mTlTsuSNI5NhDiGSjLK1YLLdQ7ADUT8UDan7aqGEdA9Um4eFeM--143zAEJeWOFNZbCvj7x3CN5wtmk5k1BOPEmJ3GD8xs5S0BO_B589QyrRShzNpbbzsobdU10xs-lc5qg-Xjp2FoOTuW3i |
| 52 | List all the pods sorted by name **Answer:** See the solution below. Explanation: kubectl get pods --sort-by=.metadata.name  https://lh4.googleusercontent.com/lAkG4m35PBvvyopcMcHayF3XCFvEl-E_lhPyASO_SlSd4ymaU7cIQS5RHrrzFSG9KVyXxhaSzLzoaNUUfeW-LnOTxAsHswGNxVmV6DRhUEJLQJBLhfxuA7jzodV5-VY5rIzB8ezpR9EAvcY |
| 53 | Task First, create a snapshot of the existing etcd instance running at https://127.0.0.1:2379, saving the snapshot to /srv/data/etcd-snapshot.db  Next, restore an existing, previous snapshot located at /var/lib/backup/etcd-snapshot-previo us.db   * #backup * Check file cấu hình etcd:   cat /etc/kubernetes/manifests/etcd.yaml    https://lh4.googleusercontent.com/vEsarjldsxCLwpw29QS08TkrhgFNP8jpZ57TBx25Ks1DO0eYU00RJtcY36yam45iDWbbSDTic7UkJ7_t5gasXPHC-XmULEq2FS4dAVu81Cnnpy74dGBfWMb-NeLD5pSfYvrf5VCD5-p1Bb4  Cài đặt etcdctl : Tham khảo  curl -LO <https://github.com/etcd-io/etcd/releases/download/v3.5.0/etcd-v3.5.0-linux-amd64.tar.gz>  tar -zxvf etcd-v3.5.0-linux-amd64.tar.gz  cd etcd-v3.5.0-linux-amd64/  https://lh4.googleusercontent.com/VovXl2HSdzErskqo1FLwcYYvfAg2h35KlhdDQruEr53OgpqnPLUycNzFFC2qpWACuuay83rimnIjyrLOQkwlCWtHY36tC93NnChx2Bc1BrZwT4_W-XrHvKgFL49dUYpelqaFm1lphwqpPdY  Tạo thư mục chứa file bk: mkdir   srv/data  ETCDCTL\_API=3 etcdctl --endpoints="https://10.38.21.41:2379" --cacert= /etc/kubernetes/pki/etcd/ca.crt --cert /etc/kubernetes/pki/etcd/server.crt --key /etc/kubernetes/pki/etcd/server.key snapshot save /srv/data/etcd-snapshot.db  https://lh4.googleusercontent.com/2dn9MXAJRpOV5l65Lz-5OHsfQF9Ppa0490lfoCqYQD9pQHhnPsBGqSF1jHOf6DJJB2nsLpRlDKIZM04GfNxjRsOYGZR3z_WJycdRp8VzJkrfy3TkynH3i9bnlsIWydOuEI3hUUm30bQLeHs   * Check sau khi tạo snapshot   ETCDCTL\_API=3 etcdctl snapshot status etcd-snapshot.db  https://lh6.googleusercontent.com/DWSIPmr47EdUyQZbTX4OdKu2wpDHWsJxO9dDW19SipkLXg9-6Cn9LbA-GXN5VMfnz7L-gqqvptl3U5md2OYfhYRul_Tea9yuEBrxFey3DX3c6-yJxX4DUoGOuccr-HUqGb9VTw  https://lh6.googleusercontent.com/k6Wy5XIj-iYPddBd629aC9diHKcTQBpqoBfdu7YkVDhTUAmbTdDTCvPKySd9dV6NcqSQGzXbskH9m7gvOVeOUWArqFPPcc8tUuKHAPGTG9DfeSDGq77aUypaZg1wRqvJF1gazNxshLaARb0  https://lh6.googleusercontent.com/mk6mjjIUVaVMo-T82ZuhZx9ZL_dNgo5I3HeS46qgMJnOUEKmeDYM8BqJyp4HvTveFG-Mf2NxXtnbKTFkPTWgDj-GgghVGPdcsIf8DxeMzwQpipPHkbV383O3zt6kLw1itGuIlj1cO2Lnm4I  #restore  cp etcd-snapshot.db /var/lib/backup  rm –rf /var/lib/etcd( danh nhanh lenh duoi)  ETCDCTL\_API=3 etcdctl --endpoints="https://10.38.21.41:2379" --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert /etc/kubernetes/pki/etcd/server.crt --key /etc/kubernetes/pki/etcd/server.key snapshot restore /var/lib/backup/etcd-snapshot-previoys.db  https://lh4.googleusercontent.com/6wpDdRgoAef3CETrD86-EE5x4b_KVxoPR_MQlno2PHvQ33jFa8BADZeUafrAwd13dWczPZ2WxGZzObvqYTaSYaYgGcZZ70p5m1D8ZJUa7-1PehK73hwNNG5qSqtWDNDknU5sz_ufMLkrF0o |
| 54 | List the nginx pod with custom columns POD\_NAME and POD\_STATUS **Answer:** See the solution below. Explanation: kubectl get po -o=custom-columns="POD\_NAME:.metadata.name, POD\_STATUS:.status.containerStatuses[].state"  https://lh3.googleusercontent.com/IHjaCIgQF3ANv2gamkJXdW0qIUuEgx9NJ_h0_9PABGfThVb-TWV5JjMJk1AxDk3p4YtSm7uFgyVpPu6YaehWZ9IC4iCIK5ksi_2Ri9KVCmKI61kE1G59Wrb-qOlbBwQ79ZDjZFXLKKmURC4  https://kubernetes.io/docs/tasks/administer-cluster/configure-upgrade-etcd/ |
| 55 | List all the pods showing name and namespace with a json path expression  Liệt kê các pods và show tên với namespace  bằng json path expression  ubuntu@master:~$ kubectl get pods -o=jsonpath="{.items[\*]['metadata.name', 'metadata.namespace']}" -A  https://lh5.googleusercontent.com/dOGC2cDpDZtXvAtRR6m2Ql6pr7oTgltVj1Ox1YenIlmOGxHlWjYoXJMzD9XxdBXZX62hzkcqfbpG-lUBCfBvbJobE3qSpWFPKOf9zcsVNvY68AqmZPm8fX3CXTcrN97Jt5XDNTcKzcYEZYFg |
| 56 | Create a pod that having 3 containers in it? (Multi-Container)  - Tạo mới 1 pod với file yaml có chứa 3 container. # vi kudu-pod.yaml    apiVersion: v1  kind: Pod  metadata:    name: kudu-pod  spec:    containers:    - name: first-container      image: nginx    - name: second-container      image: busybox      command: ['sh', '-c', 'echo Hello from the second container! && sleep 3600']    - name: third-container      image: alpine      command: ['sh', '-c', 'echo Hello from the third container! && sleep 3600']    https://lh4.googleusercontent.com/5KL3ba_oV3_1EZQpgJ6HwmiyTHRIbA2ASAza3NdHH7XXzYuUuXtVOZwNktk_ZV8ahraOyEDkjWkNmJUxklSXZL1VcWrxEe7GJP-WTb8sK9LGHUbSgZom02KK_NgJY1yjO9jCKn5KG4WdMLGL  https://lh3.googleusercontent.com/FME6Qx9Ca5ipUq5EKjHF5P3oCW_h_LZeEnAbdAw9DGzQj04AAHnScLWpMiXl4u2lNmgUmncCjY7mQGdGhhmXTSF8j1ejI9qbNixEm-WdPuak9yjKq2_Yu12sXtTraOb6Bd5ojSLxtRbLLj-v |
| 57 | Set the node named ek8s-node-1 as unavailable and reschedule all the pods running on it  Set node worker1 unavailable và reschedule lại cho các pods chạy trên node đó  https://lh6.googleusercontent.com/qzGXf9THXk6wA_kqQRldVySlP8wT1BfxpuspQNxX3NkdPWE-zleV7D9nakBjnqg5VPNTwrJKXaO7WS_kASl7_8W3UqKYf_3ZwvL9Wg2ofJqpZK18YLo92YlUaKxXD9jdvOzIQmpr3t4EOIr4 |
| 58 |  |
| 59 | A Kubernetes worker node, named wk8s-node-0 is in state NotReady. Investigate why this is the case,  and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made  permanent.  You can ssh to the failed node using:  [student@node-1] $ | ssh Wk8s-node-0  You can assume elevated privileges on the node with the following command:  [student@w8ks-node-0] $ | sudo –i  Một worker node tên là “ wk8s-node-0” có trạng thái chạy là “NotReady”, điều tra về trường hợp này  Case này thì đọc theo giải của nó chứ em không biết sao nó hỏng 🙂  Theo giải là họ ssh vào con worker rồi reset lại service kubelet là xong 😀  Cần gõ những lệnh sau:  kubectl config use-context wk8s  ssh Wk8s-node-0  sudo -i  systemctl restart kubelet  systemctl enable kubelet  Restart lại service kubelet  Exit  kubectl get nodes  https://lh3.googleusercontent.com/ENtzVfpxzG2VmFInTRwzsUeaK0i8jisGHNbb9XIgc_PEb4ffKR6ffvOedMgDIAYA-c-rwLg9j0f_L3_widWinv4A2Uk5i7sdwxN7j4gxAm2nPdKyw37j8nL5yWWp_mBC_QswxP2YGRtZCY4e |
| 60 | An existing Pod needs to be integrated into the Kubernetes built-in logging architecture (e. g. kubectl logs).Adding a streaming sidecar container is a good and common way to accomplish this requirement.  Task  Add a sidecar container named sidecar, using the busybox Image, to the existing Pod big-corp-app. The new sidecar container has to run the following command:  /bin/sh -c tail -n+1 -f /var/log/big-corp-app.log  Use a Volume, mounted at /var/log, to make the log file big-corp-app.log available to the sidecar container.  B1. Get pod as .yaml file  kubectl get pod -o yaml > big-corp-app.yaml  B2. Sửa lại file big-corp-app.yaml  Thêm 1 container sidecar với nội dung như sau:  - name: sidecar        image: busybox        args: [/bin/sh, -c, 'tail -n+1 -F /var/log/big-corp-app.log']        volumeMounts:          - name: logs            mountPath: /var/log  Sau khi sửa thì được file **big-corp-app.yaml**  với nội dung như sau:    spec:      containers:      - name: sidecar        image: busybox        args: [/bin/sh, -c, 'tail -n+1 -F /var/log/big-corp-app.log']        volumeMounts:          - name: logs            mountPath: /var/log      - args:        - /bin/sh        - -c        - |          i=0; while true; do            echo "$(date) INFO $i" >> /var/log/big-corp-app.log;            i=$((i+1));            sleep 1;          done        image: busybox        imagePullPolicy: Always        name: big-corp-app        resources:          limits:            cpu: 500m            memory: 128Mi          requests:            cpu: 500m            memory: 128Mi        terminationMessagePath: /dev/termination-log        terminationMessagePolicy: File        volumeMounts:        - mountPath: /var/log          name: logs        - mountPath: /var/run/secrets/kubernetes.io/serviceaccount          name: kube-api-access-5ws5g          readOnly: true    B3. Apply cấu hình mới  kubectl apply -f big-corp-app.yaml  B4. kiểm tra log trên container sidecar  kubectl logs big-corp-app sidecar  #out put  Fri Jun  9 05:22:10 UTC 2023 INFO 1810  Fri Jun  9 05:22:11 UTC 2023 INFO 1811  Fri Jun  9 05:22:12 UTC 2023 INFO 1812  Fri Jun  9 05:22:13 UTC 2023 INFO 1813  Fri Jun  9 05:22:14 UTC 2023 INFO 1814  Fri Jun  9 05:22:15 UTC 2023 INFO 1815  Fri Jun  9 05:22:16 UTC 2023 INFO 1816  Fri Jun  9 05:22:17 UTC 2023 INFO 1817  Fri Jun  9 05:22:18 UTC 2023 INFO 1818  Fri Jun  9 05:22:19 UTC 2023 INFO 1819  Fri Jun  9 05:22:20 UTC 2023 INFO 1820  Fri Jun  9 05:22:21 UTC 2023 INFO 1821  Fri Jun  9 05:22:22 UTC 2023 INFO 1822  Fri Jun  9 05:22:23 UTC 2023 INFO 1823  Fri Jun  9 05:22:24 UTC 2023 INFO 1824  Fri Jun  9 05:22:25 UTC 2023 INFO 1825  Fri Jun  9 05:22:26 UTC 2023 INFO 1826  Fri Jun  9 05:22:27 UTC 2023 INFO 1827  Fri Jun  9 05:22:28 UTC 2023 INFO 1828 |
| 61 | Create and configure the service front-end-service so it's accessible through NodePort and routes to the existing pod named front-end  kubectl expose pod nginx-pod --name=front-end-service --port=80 --target-port=80 --type=NodePort |
| 62 | **Get list of all pods in all namespaces and write it to file “/opt/pods-list.yaml”**  $ sudo touch /opt/pods-list.yaml && sudo chown ubuntu /opt/pods-list.yaml  $ kubectl get pod --all-namespaces > /opt/pods-list.yaml  $ cat /opt/pods-list.yaml  https://lh5.googleusercontent.com/ZprISzNYVNElP6Oc4hfNPXpi9oFr3VKSzwByt6vaMoA3K86XcCaFU9CWJIqWJdihNSHePGRx7L8CsEiGrpocgl365LR7VlltKqfOrHiq4RiHUAJoygRv4XFTxwdRBh-N9jYGLP1xyTsarzKO |
| 63 | Reconfigure the existing deployment front-end and add a port specification named http exposingport 80/tcp of the existing container nginx.  Create a new service named front-end-svc exposing the container port http.  Configure the new service to also expose the individual Pods via a NodePort on the nodes on which they are scheduled.  **Giải:**  kubectl edit deploy front-end -o yaml  Thêm đoạn dưới đây vào trong .spec.template.spec.containers   |  | | --- | | ports:       - containerPort: 80         name: http         protocol: TCP |   https://lh5.googleusercontent.com/K7rghTji1HbkGW-QitBgmAkx5hrWzzp4zQFBba_dK0SUBuY16RepIuckxAHSsET8L14AIJyOLX7h6uvrTf4rHl9RGXGyGs3zhM0-A29JW5orBEf9f-1gIDqabMJOggS3QChwaXtoBbUsde1P  vim front-end-svc.yaml   |  | | --- | | apiVersion: v1  kind: Service  metadata:    name: front-end-svc  spec:    selector:      app: front-end    type: NodePort    ports:    - name: http      port: 80      protocol: TCP      targetPort: http |   kubectl apply -f front-end-svc.yaml |
| 64 | List all persistent volumes sorted by capacity, saving the full kubectl output to /opt/KUCC00102/volume\_list. Use kubectl 's own functionality for sorting the output, and do not manipulate it  # Liệt kê tất cả các Persistent Volume sắp xếp theo dung lượng  **root@master:/opt/KUCC00102# kubectl get pv --sort-by=.spec.capacity.storage**  https://lh5.googleusercontent.com/rOTixy1Pf_Ky9RT9bhQvO9evLx-842dnh1Ok-8o_cS2TT_683M6uY_Tu4Zwpyh8cS8Cgni6GdYMe8wsbcUz5nZ0ydDIAJAt3Et7tYmO_220nPgAUmdVcQ5XJ8YlvnQzDb-jUf7RAFFsRWQuz  # Lưu đầu ra của câu lệnh vào một tập tin  **root@master:/opt/KUCC00102# kubectl get pv --sort-by=.spec.capacity.storage > /opt/KUCC00102/volume\_list** |
| 65 | **Create a persistent volume with name app-data , of capacity 1Gi and access mode ReadOnlyMany. The type of volume is hostPath and its location is /srv/app-d**  Tạo một ổ đĩa liên tục với tên app-data, dung lượng 1Gi và chế độ truy cập ReadOnlyMany. Các  loại âm lượng là hostPath và vị trí của nó là /srv/app-da  **Answer:**  Tạo file yaml: nano pv.yaml  piVersion: v1  kind: PersistentVolume  metadata:    name: app-data  spec:    capacity:      storage: 1Gi    accessModes:      - ReadOnlyMany    hostPath:      path: /srv/app-data  run file yaml:   kubectl apply -f pv.yaml -n development  https://lh6.googleusercontent.com/QqqIgdnFk7N1bUKLpYY44nfBKrB51yzGSJTuKxYiHohkQ9vCIYSSKCp3noGjPYUNEn8MAMs1nbOpHm-vzk7agt4JMfuYtGmMiwMiyNGd-GDrB-WLpOxx-zq5pGzpaK809W3VHEJBpOY-dQI |
| 66 |  |
| 67 | update Master:  #kubectl drain k8s-master --delete-local-data --ignore-daemonsets --force  sudo apt-mark unhold kubelet kubectl kubeadm  sudo apt-get install -y kubelet=1.26\* kubeadm=1.26\* kubectl=1.26\*  sudo kubeadm version  sudo kubeadm upgrade apply v1.26.5  sudo apt-mark hold kubelet kubectl kubeadm  sudo systemctl daemon-reload  sudo systemctl restart kubelet  kubectl uncordon master  kubectl get node  update Worker:  # Tren master  kubectl drain worker1 --ignore-daemonsets --delete-emptydir-data --force  # tren worker  sudo apt-mark unhold kubelet kubectl  sudo apt-get install -y kubelet=1.26\* kubectl=1.26\*  # Tren master  kubectl uncordon worker1 |
|  | **alias k=kubectl**  **alias kgp='k get po '**  **alias kgs='k get svc '**  **alias kgn='k get node '**  **alias kgd='k get deploy '**  **alias kn='k config set-context --current --namespace '**  **alias ka=' k apply -f '**  **alias kr=' k run --dry-run=client -oyaml --image '** |
|  | **www.pnh.vn** |