

KUBERNETES



By:Mostafa Mahmoud Bahgat

LinkedIn:<https://www.linkedin.com/in/mostafamahmoudbahgat>

أنواع الـ Architecture Patterns

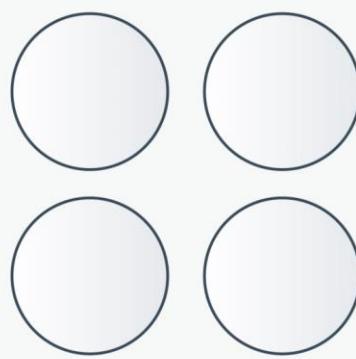
Microservices Architecture	SOA Service oriented Architecture	Monolithic Architecture
<p>هي عبارة عن Services صغيرة جداً ومتعددة بتقديمك ال APP بتاتعاًك وهذا كل ال Service معتمدة على ذاتها وممكن كل Deployed Service لوحدها وممكن اعمل Update لل Services من غير م اعمل Rebuilding كل APP</p>	<p>هنا يتم تقسيم ال Software الى Service متعددة لكن كلها مرتبطة بعض ومعتمدة بعض من خلال ال Set Of API</p>	<p>هو كان لفتره طويلة ال Design لاي Software وهنا ال App يكون عبارة عن Single Piece Of Software البرنامج كله قطعة واحدة ودا مخلي كل حاجه معتمدة على حاجه تاني وبالتالي عملية ال Scaling بتكون صعبه جداً</p>

Monolithic vs. SOA vs. Microservices



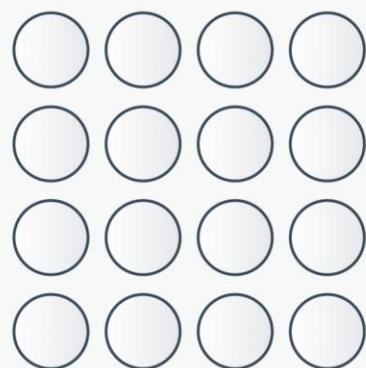
Monolithic

Single Unit



SOA

Coarse-grained



Microservices

Fine-grained

ال Containers هي أفضل حاجه عشان اعمل Deliver لـ Microservices وعشان اعمل كل ال Containers Orchestration دي تحتاج ال Containers Management وأشهرها هي ال Kubernetes وايضا يوجد ال Marathon-Nomad

Azure Services Fabric – Amazon Elastic Container Services(ECS) – Docker Swarm

امثله ع K8s as a service

Amazon Elastic Container Services(ECS)-

Azure K8s Service(AKS)-

Google K8s Engine(GKE)-

هي : Kubernetes بتعمل Containers Orchestration

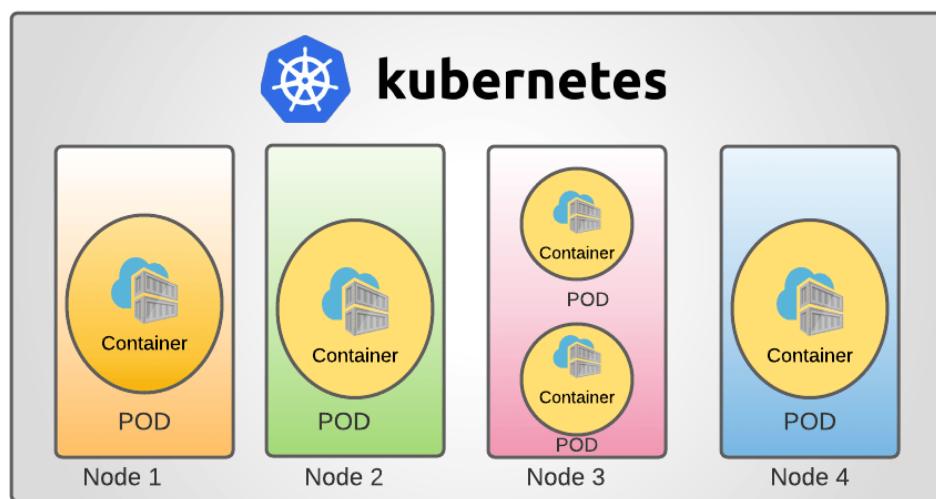
Containers App لـ automating-deployment-scaling-management

وهي كلمة يونانية تعني الشخص قائد السفينة وتخصار ال K8s (اول حرف k وآخر حرف s وبينهم 8 حروف) وتنطق Kate's

السؤال عن ادارتها هي مؤسسة ال Cloud native computing foundation(CNCF)

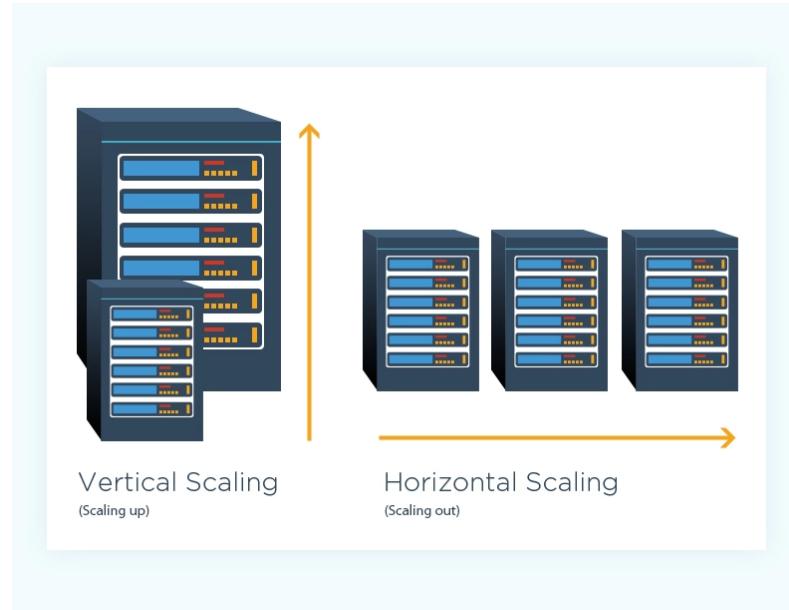
لية يحتاج اني استخدم k8s ?

لو عندي مثلا 3node وكل node بداخله 50 container فه تكون عمله ال manage لـ k8s .

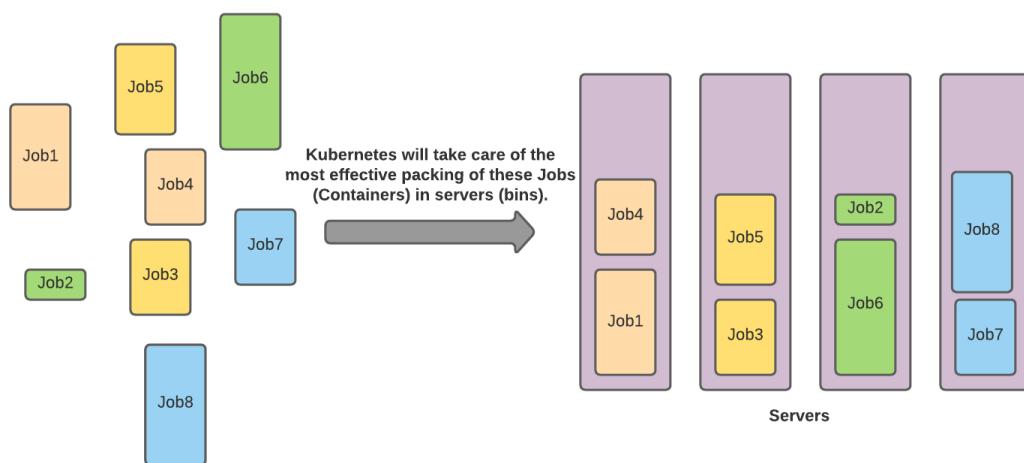


-مميزاتها

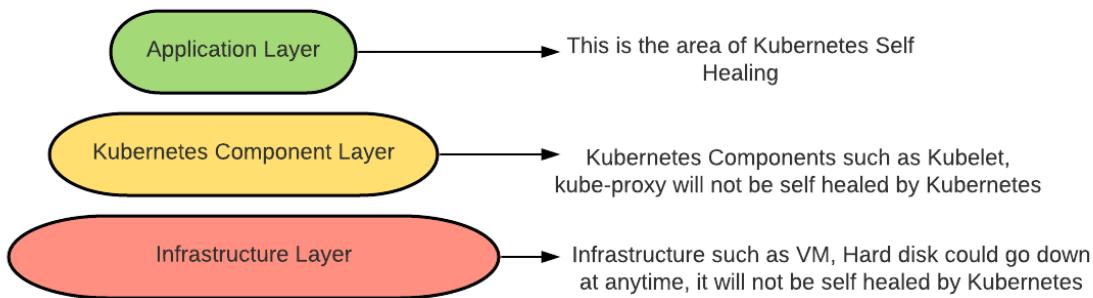
Scalability : بقدر اعمل pods لـ **Horizontal-vertical** بناءا ع مثلا استخدام ال CPU ال هي اني بزود عدد ال **Horizontal Recourses** وال vertical جهاز يبقى اتنين وهكذا



k8s : لما اجي اعمل pod جديـد لازم يكون داخل node فـ ال node من حيث ال available CPU and RAM وبتضعـ ال pod فـ ال المناسب من حيث ال RAM-CPU دـي (Resource)

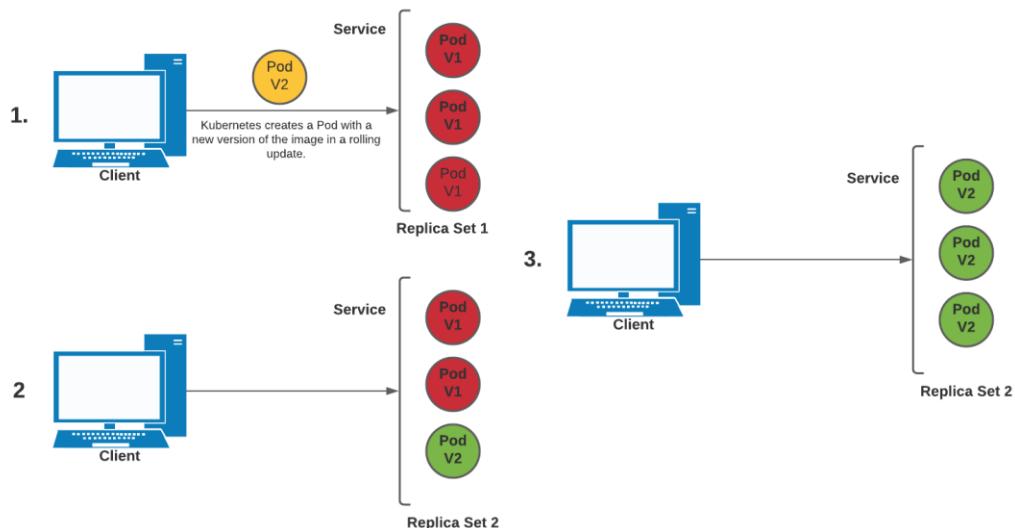


هنا بقدر اعمل Failed Node Reschedules Replaces Kill لـ Self-healing-3 بناء ع Unresponsive Rules معينه عشان يمنع الترافيك انه يوصل Running Unresponsive Containers (من الاخر لو لاقى pod failing بعد م كان ((pod and node) عشان يرجع يشتغل من تاني) هيعمله Restart



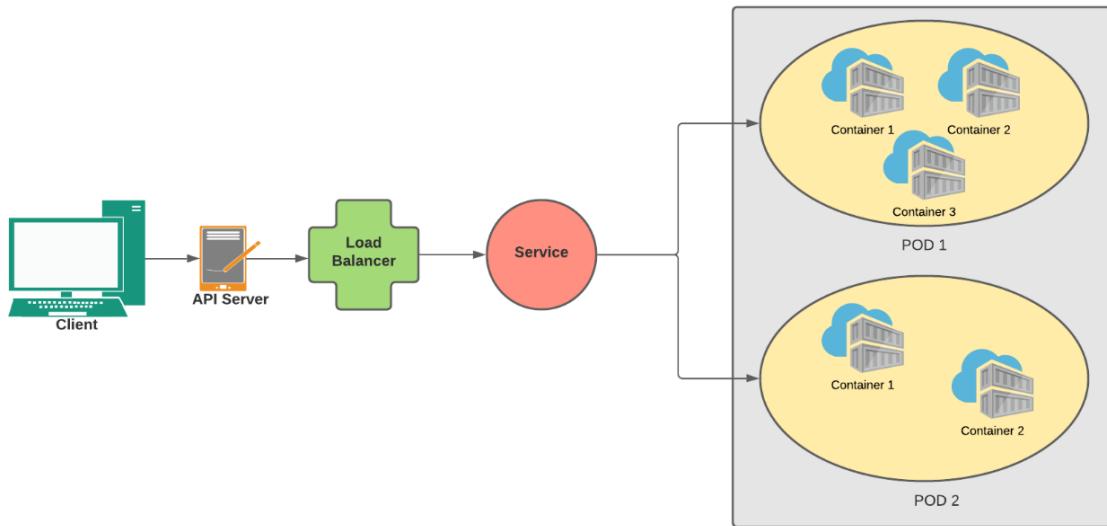
Rollouts and Rollbacks : بقدر اعمل Automated Rollouts and Rollbacks-4 Applications Monitoring configuration changes و Application Update

فلو عندي مثلا pod app v1 و عاوز اعمل update لـ v2 مش هيكون في downtime وه تكون عن طريق ال rolling update او ال rooking out و اقدر ارجع لـ v1 عن طريق ال rollback من غير downtime برضو

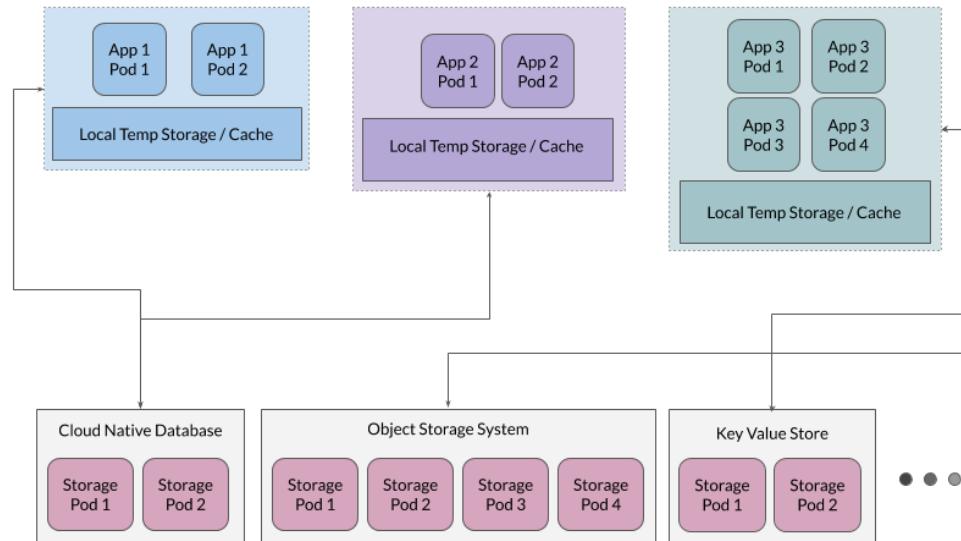


: Service discovery and load balancing-5

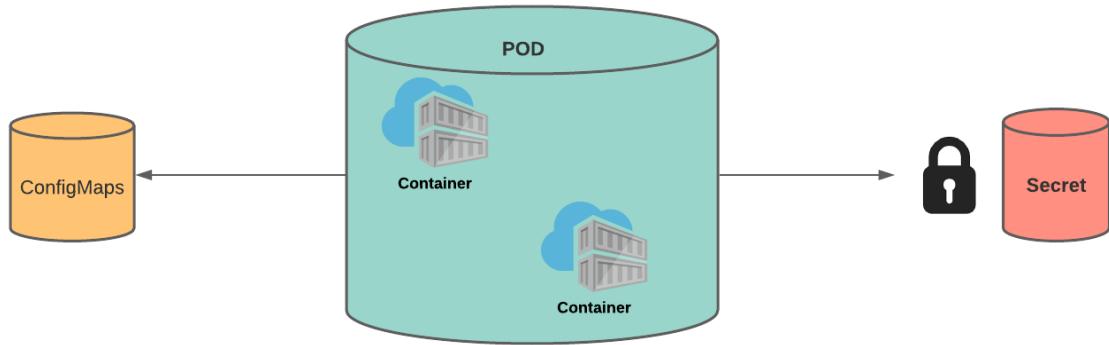
- ان ال k8s بتعمل expose لـ pod عن طريق ال DNS or IP
لو عندي pod الترافيك عليه عالي ف ال k8s بتعمل LB بيوزع ال Request على ال pods (عن طريق حاجه اسمها service)



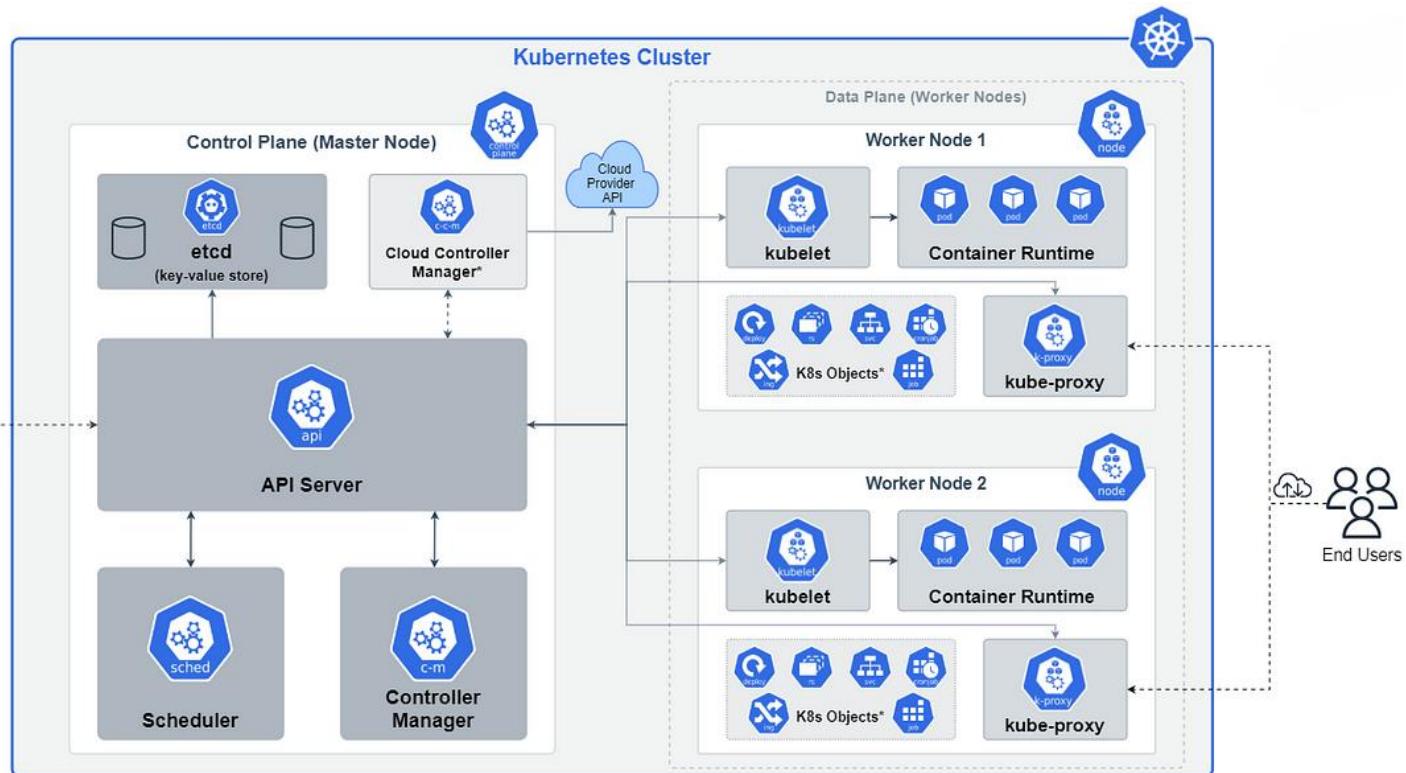
اقدر استخدم ال cloud provider-local storage عشان اخزن data بناء ال pod لان ال pod لو اتحذف ال data ال بداخله بتحذف معه



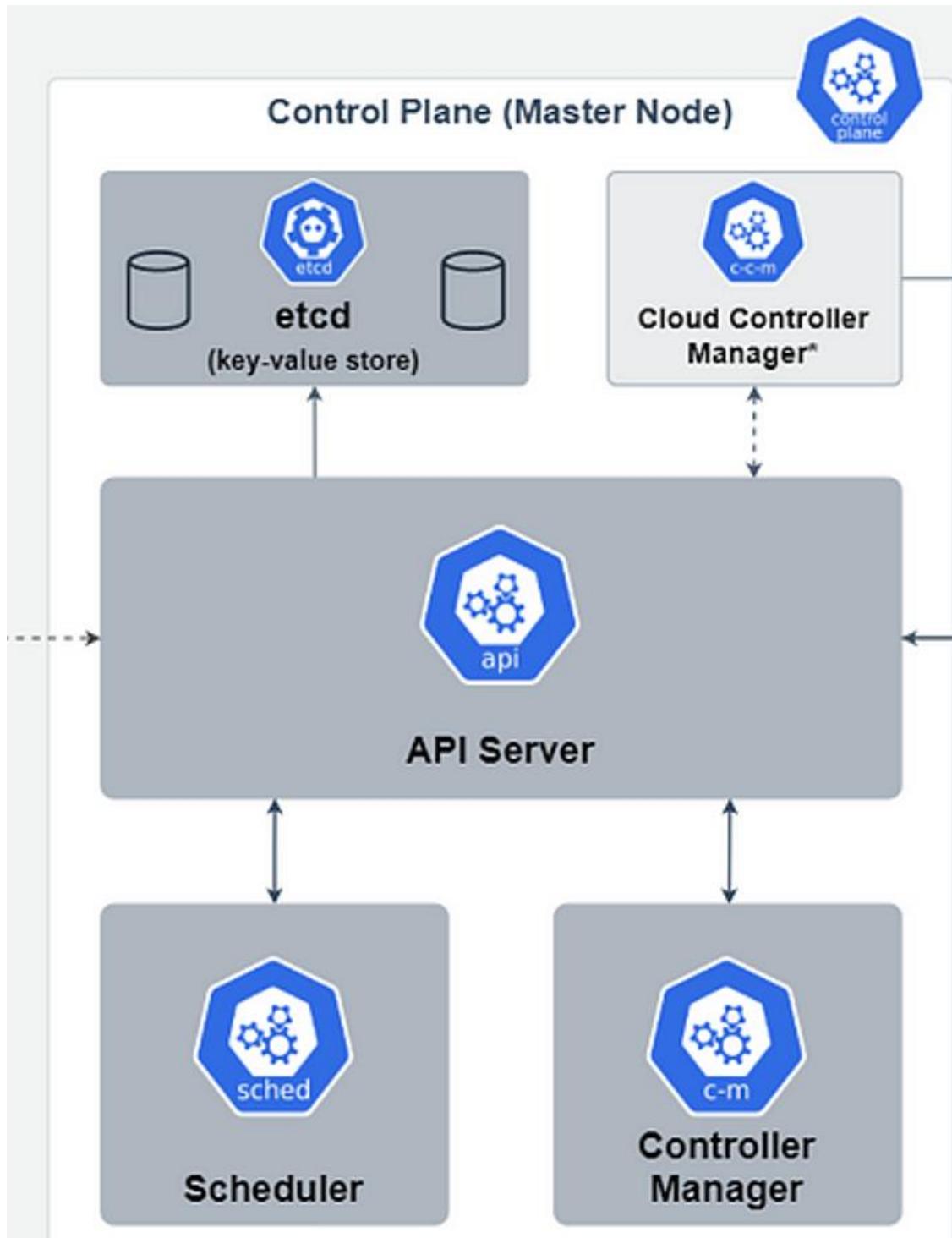
Management Secret and Configuration Management-7 : يقدر يعمل بعض الداتا Container image ويعزلها عن ال Username-password الحساسة ال فيها مثلا



Master Node وال Worker Node K8s K8s Architecture



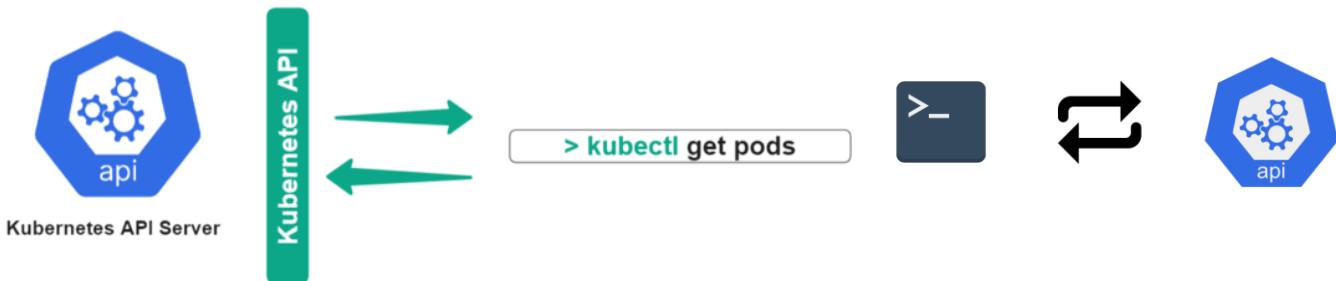
: هو بيوفرلي بيئه عمل ال Control Plane ودي المسؤولة عن التحكم او ال Master Node- Cluster في ال K8s Cluster يعني هو العقل المدبر لكل العمليات ال بتحصل داخل ال Cluster وعشان اتعامل مع ال User K8s Cluster لـ Requests ال بييعت API (هو ال بعمله بيـه لـ config cluster بتاعي) او Web UI او CLI او



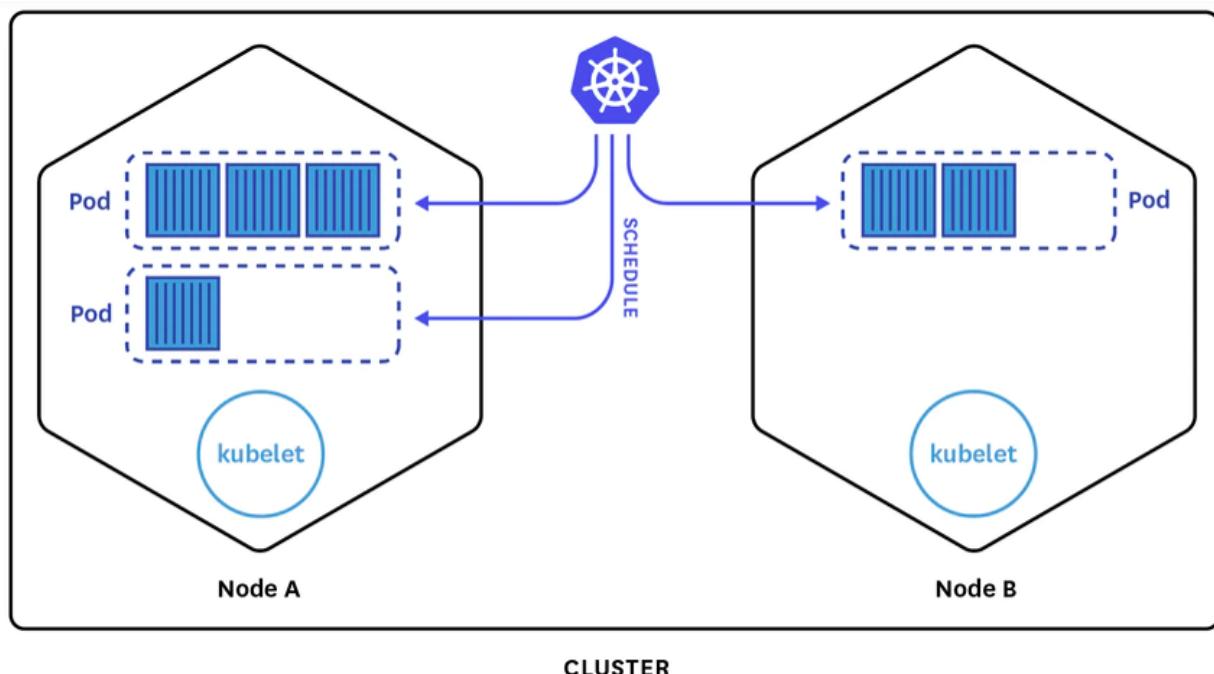
:Master Node Components

هو يعمل RESTful API Server-1 : بتاعته ال Calls لـ etcd data store من K8s Cluster Current state API Process بيقرا

(هو ال بي التواصل مع كل ال component ال داخل ال master وهو ال بي التواصل مع ال worker) عشان ال master يعرف يتواصل مع ال

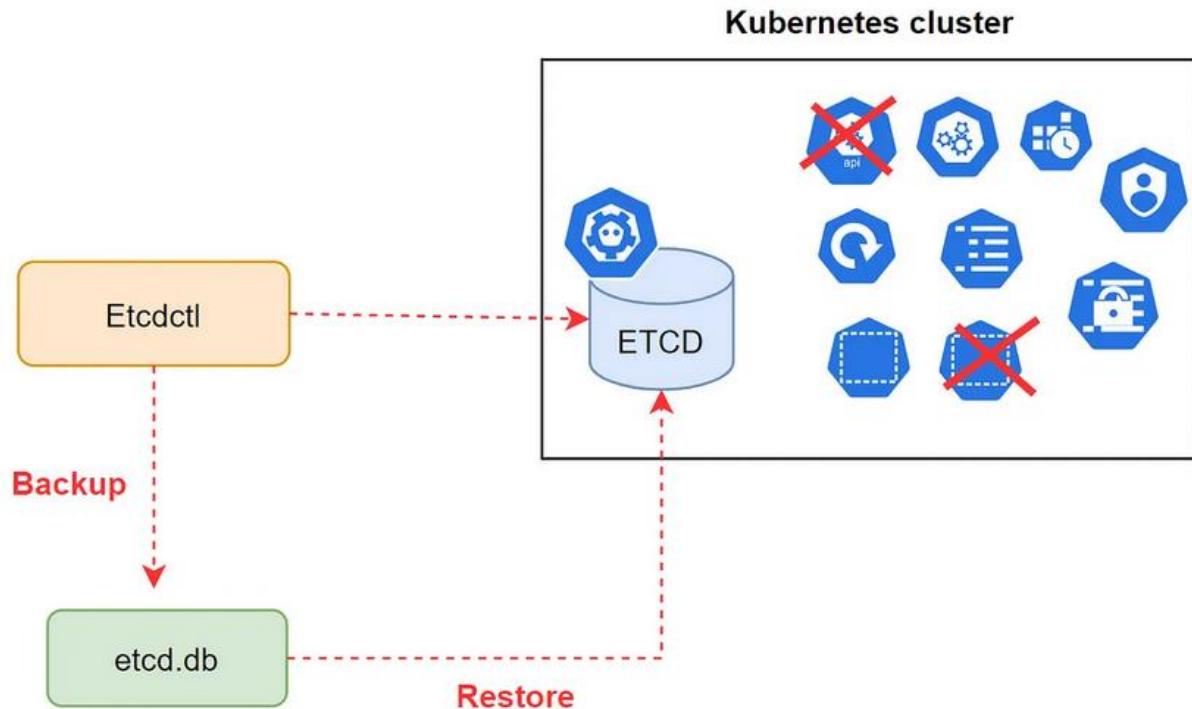


هو بيعمل Pods Assigns Nodes لـ Scheduler-2 يعني هو ال بيحدد ال pod المناسب لكل بناء ع بعض الشروط و Resources المتاحة (بيقول ال pod داي تعمله create على انهي (Node

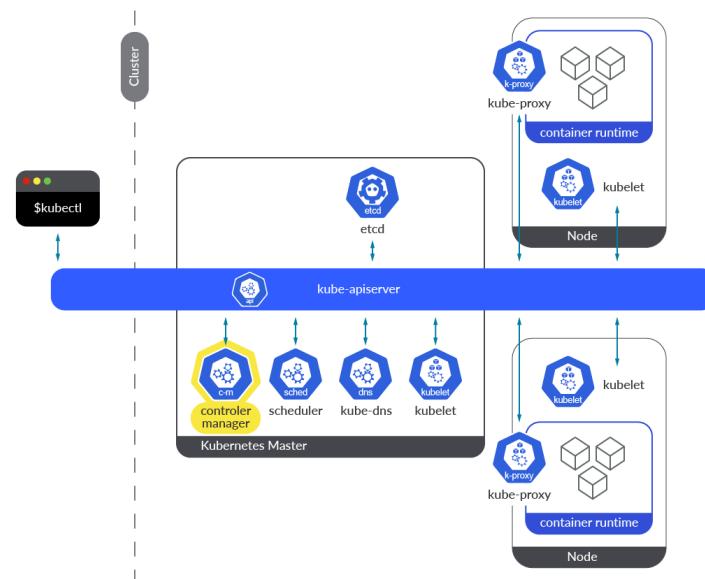


Distributes : هي نوع مناسب لـ Reliable Key-Value Store اسمها etcd-3 وال الوحيد الذي يتعامل مع ال etcd Data Store من خلال حاجة اسمها Backup-Snapshot-Restore ومن خلالها تقدر تعمل etcdctl

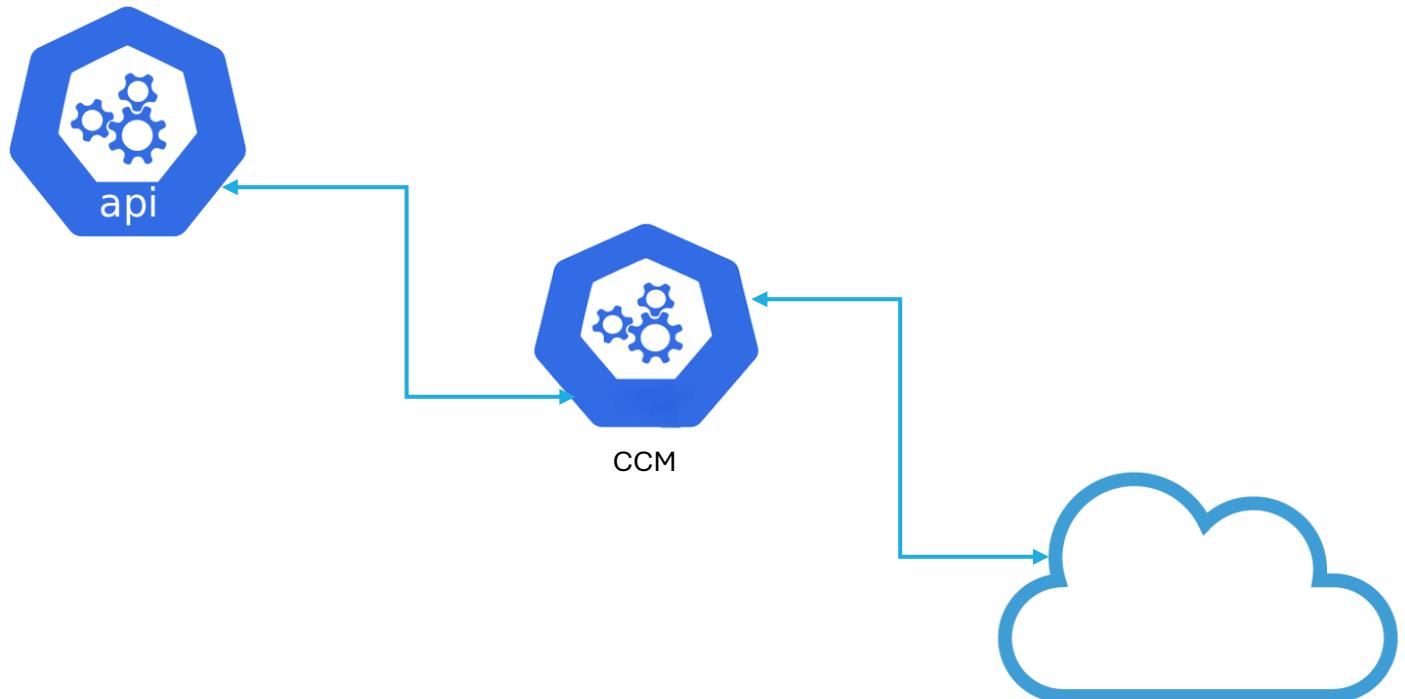
(هي DB تكون فيها كل ال data بتاع ال Cluster)



Design state : هو المسؤول ان يصل من Current state الي Controller Manager -4 من الحالة الحالية الي الحالة المرغوب فيها اني اوصلها ع K8s وهو ال بببدأ يتصرف لما تكون ال Nodes تكون Unavailable عشان يتأكد ان Pod زى ما احنا متوقعين يعني لو 3Container فى Pod بس ال شغال يحاول يشغلهم (كل دا من خلال التواصل مع ال API)

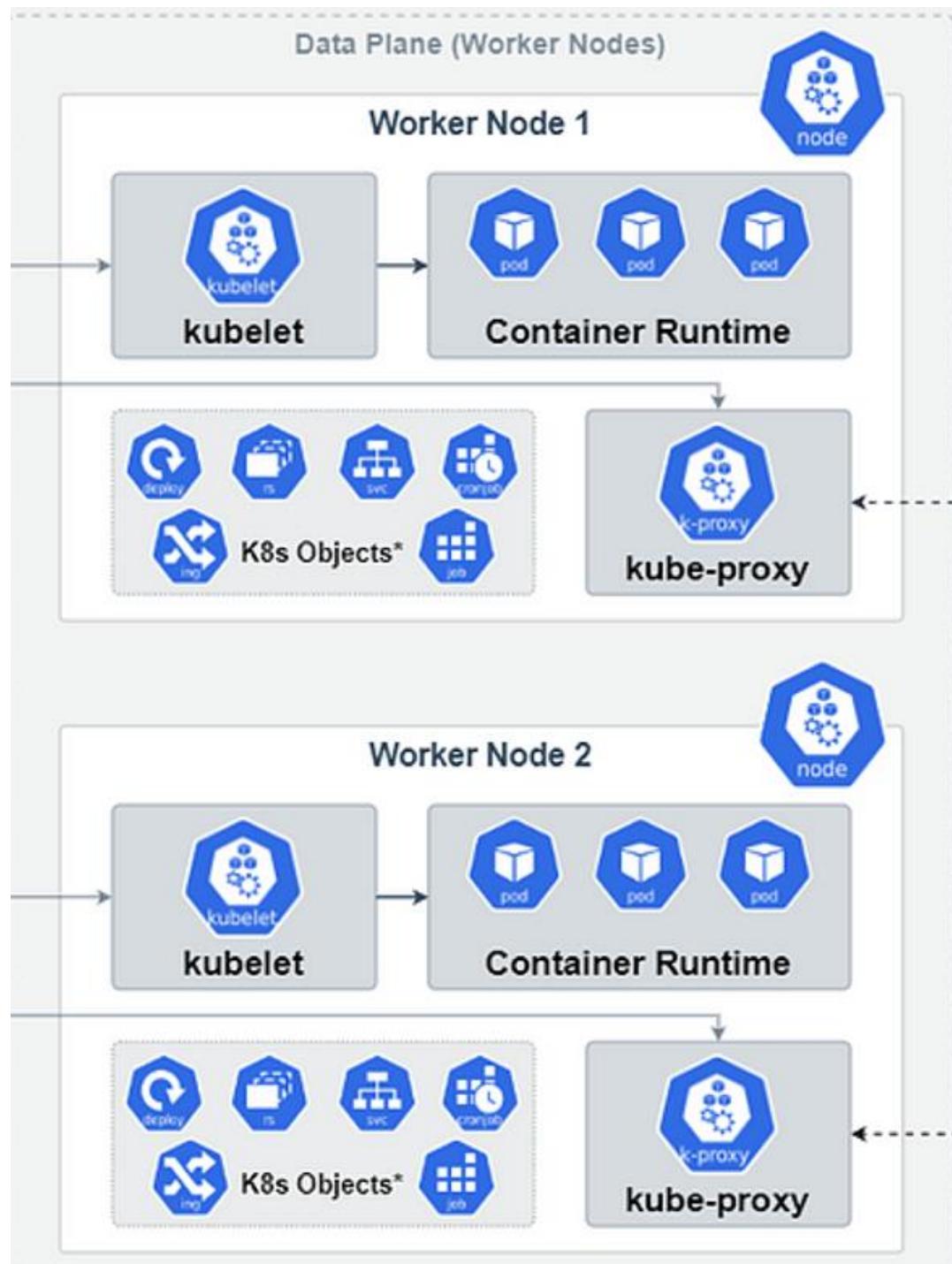


من خلاله بقدر اعمل communicate مع أي provider Cloud controller Manager-5



ال Worker Node : هو ال بيوفولي بيئه العمل لل Application من خلال ال Containerized و معمولها Encapsulated داخل ما يسمى بال Pods وال Pods دي اصغر K8s ودي ال بيكون بداخليها ال Containers بمعنى تاني ان ال Pods هي عباره عن Storage-Network Resources او اكتر بينهم لل Shared Container

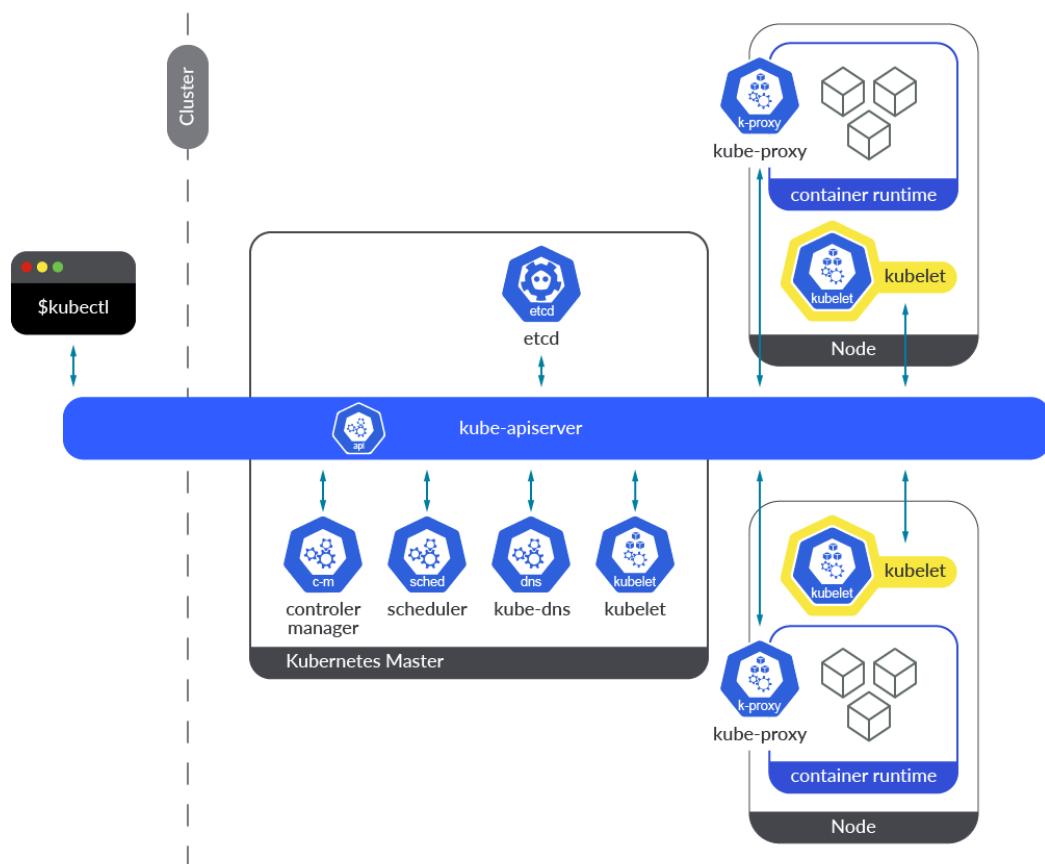
:Worker Node Components



هو kubelet-1 Communication Agent يقدر يعمل مع كل ال Running Nodes . هو ال موجوده في Master Node هو ال مستقبل Pod Definitions من Control Plane وهو ال بيتعامل Container Runtime مع API Server وهو ال Monitors Health لل Containers Resources .

ول يكن عاوز اعمل create ل pod فهكتب امر ال cli في create فلامر هيروح لل API وال Kubelet فيه عاوز عمل pod جديد في kubelet يبعث الامر للcontainer runtime وهو ال responsible عن create pod .

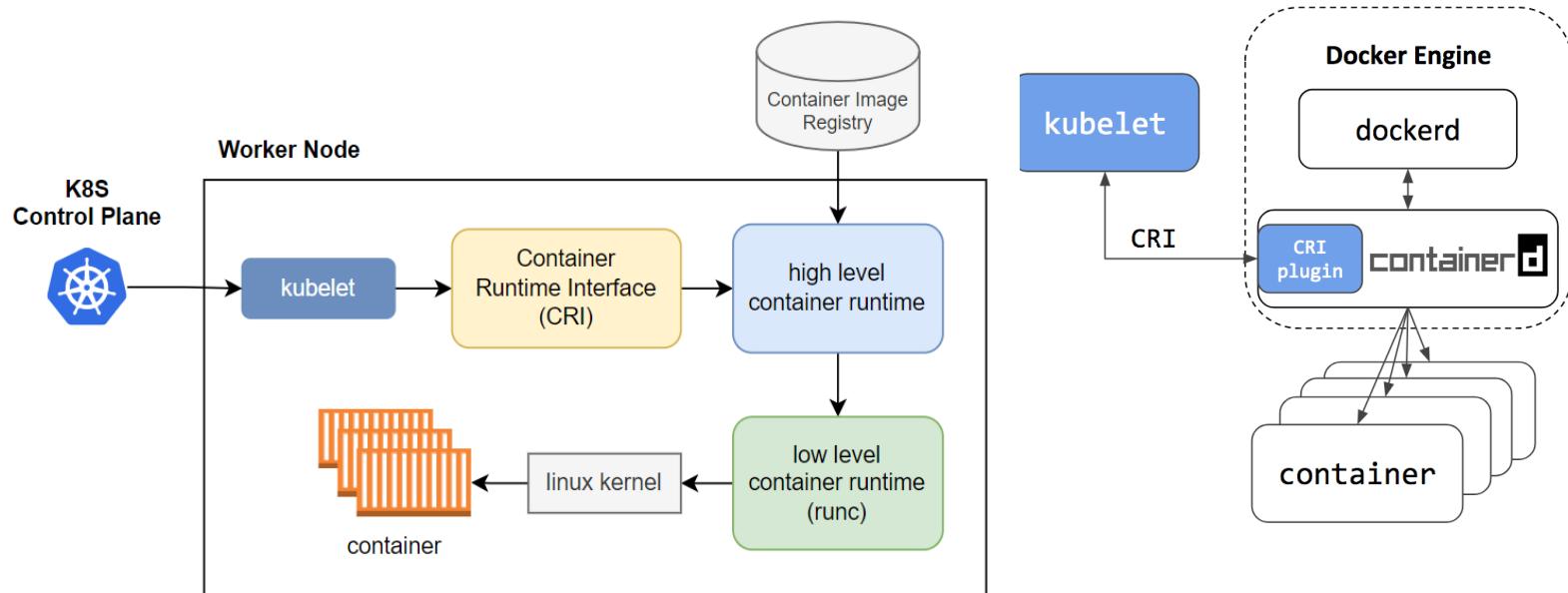
ويبعث كل فترة report لل API حالة ال pods



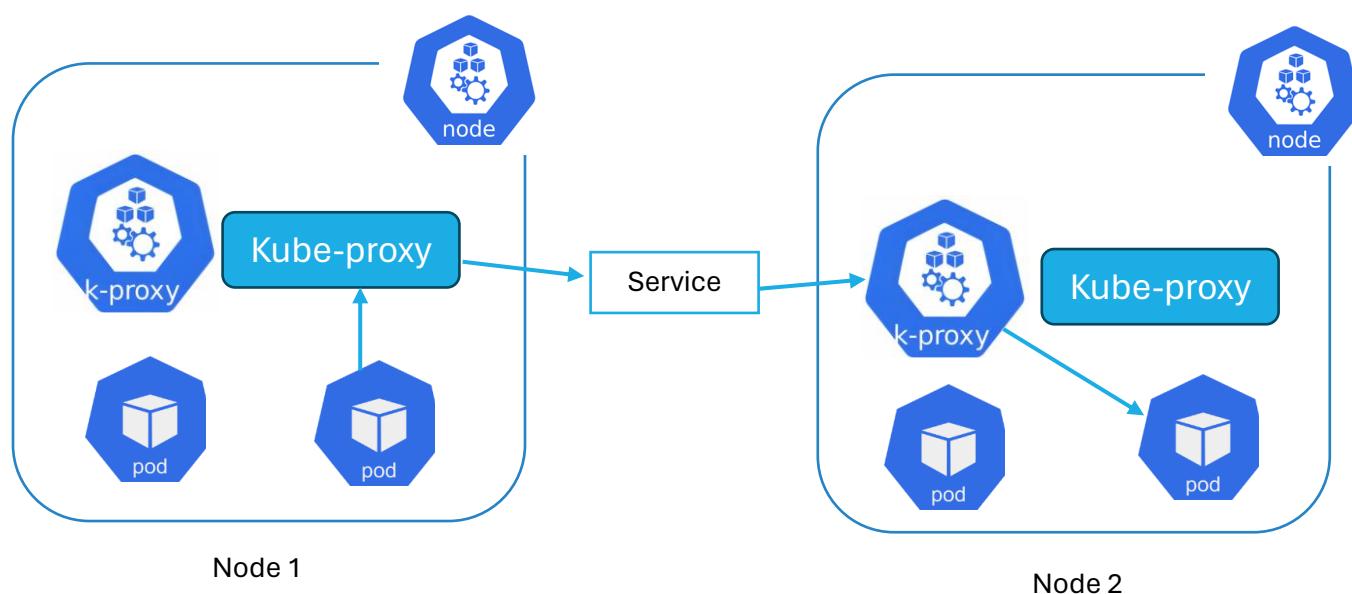
Pods : ودا ال من خلاله ال CRI(Container Runtime Interface) او Container Runtime-2
بيعمل Run لـContainers واشهر CRI في ال K8s هي ال

docker-3 CRI-o-2 Containerd-1

(باختصار ال CR هو ال بي عمل Create لـ Pods داخل ال worker node)



Network Agent (Kube-proxy) هو كل ال Nodes Run وبيكون على كل ال Network Rules Update-Maintenance عن ال Network Manage وهو شغال بال Round-Robin (بيعمل TCP-UPD-SCTP) لـ Network Communication بين ال pods وبين Worker node وبين Worker node في pod (الثاني)

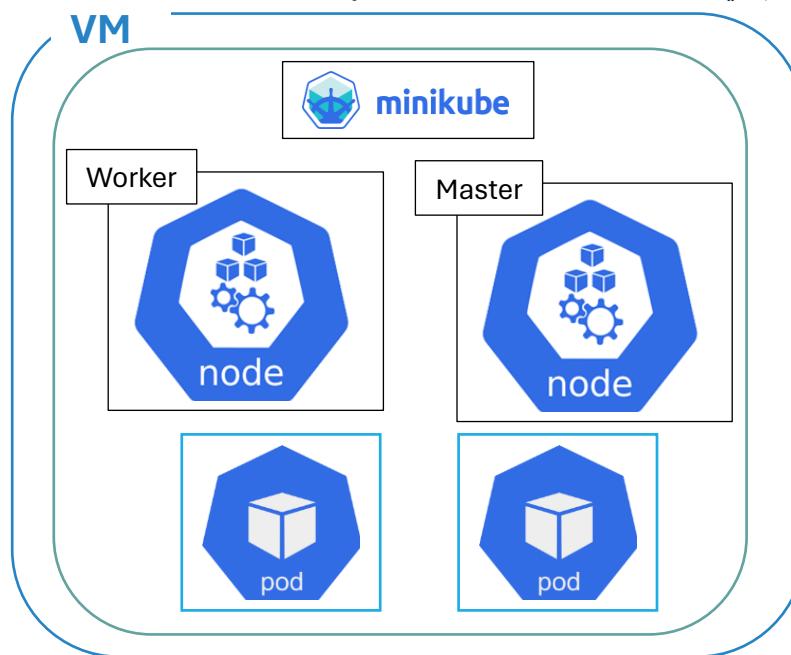


فلو ال pods ال فكل node هيتواصلو عن طريق ال k-proxy ولو pod1 عاوز يتواصل مع node2 في pod2 هيتواصلوا مع بعض عن طريق ال k-proxy لكل واحد عن طريق حاجه اسمها . service

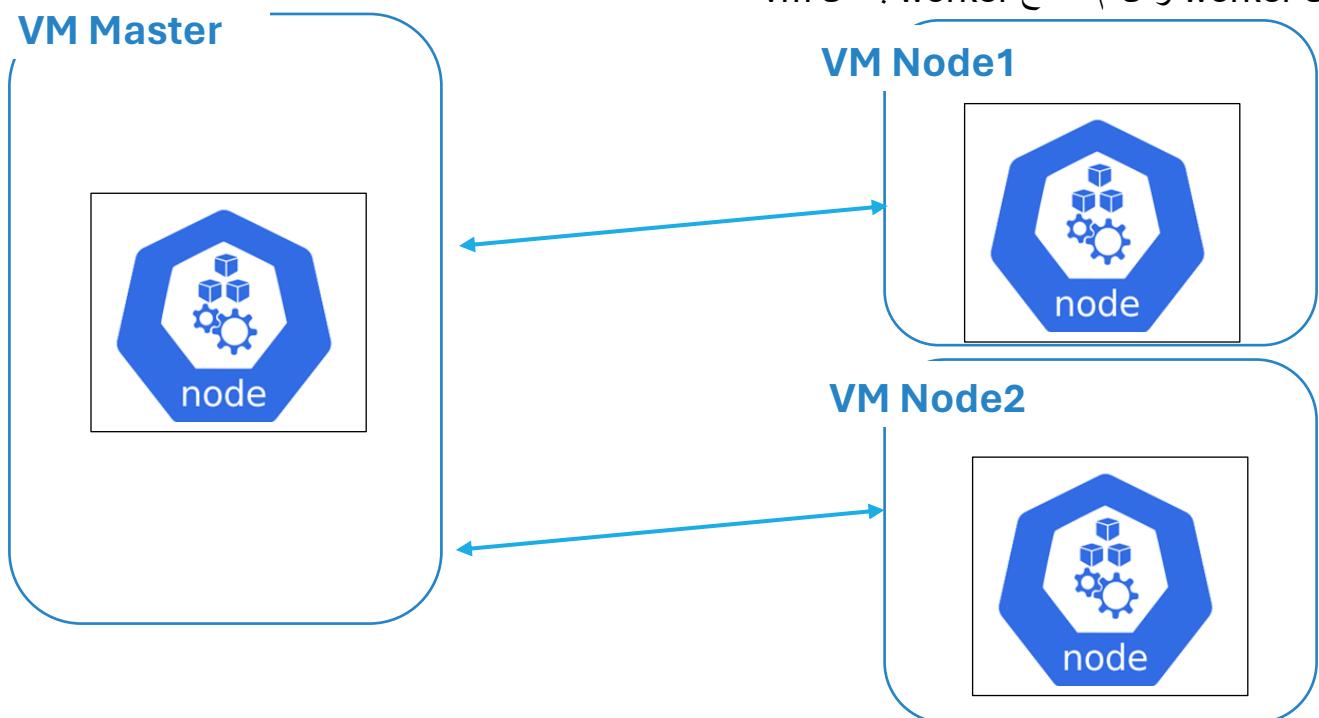
(minikube-kubeadm) install : Install K8s

-1 : بتكون على vm واحده بينزل عليها docker ويعدين بينزل عليها ال worker ولو م ينزل بيعمل master وال minikube

والطريقه مش بتستخدم في الشركات او ال production بتكون test فقط



ودي الطريقه المستخدمة في الشركات بيكون عندك vm هي ال Master و vm هي ال worker وكل م تحتاج vm بتعمل worker



```
mostafa@k8s:~$ kubectl get node
NAME      STATUS   ROLES      AGE   VERSION
minikube  Ready    control-plane  11m   v1.30.0
mostafa@k8s:~$
```

بعمل list بالي node ال عندي

```
mostafa@k8s:~$ minikube status
minikube
  type: Control Plane
  host: Running
  kubelet: Running
  apiserver: Running
  kubeconfig: Configured
```

```
mostafa@k8s:~$ █
```

بشواف ال بتاع ال minikube status

```
mostafa@k8s:~$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
mostafa@k8s:~$ █
```

بشواف معلومات عن ال cluster بتاعي

لو قفلت ال vm وفتحتها تاني هحتاج اكتب command عشان ال minikube ترجع running

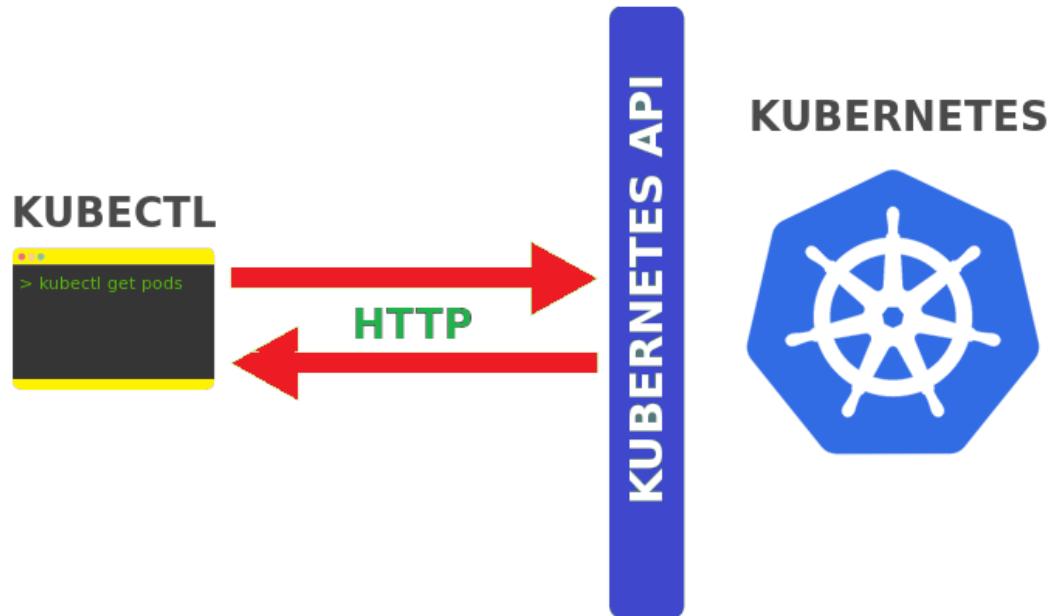
```
mostafa@k8s:~$ kubectl get nodes
Unable to connect to the server: dial tcp 192.168.49.2:8443: connect: no route to host
mostafa@k8s:~$ minikube start
* minikube v1.33.1 on Ubuntu 22.04
* Using the docker driver based on existing profile
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.44 ...
* Restarting existing docker container for "minikube" ...
* Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
  - Using image docker.io/kubernetesui/dashboard:v2.7.0
  - Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
* Some dashboard features require the metrics-server addon. To enable all features please run:

  minikube addons enable metrics-server

* Enabled addons: storage-provisioner, dashboard, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
mostafa@k8s:~$ kubectl get nodes
NAME      STATUS   ROLES      AGE   VERSION
minikube  Ready    control-plane  40m   v1.30.0
mostafa@k8s:~$ █
```

ال minikube start : command

الـ kubectl هو الـ من خلاله بقدر لـ deploy-manage-inspect وكمان من خلاله بشوف الـ logs



NAME	SHORTNAMES	APIVERSION	NAMESPACE	KIND
bindings		v1	true	Binding
componentstatuses	cs	v1	false	ComponentStatus
configmaps	cm	v1	true	ConfigMap
endpoints	ep	v1	true	Endpoints
events	ev	v1	true	Event
limitranges	limits	v1	true	LimitRange
namespaces	ns	v1	false	Namespace
nodes	no	v1	false	Node
persistentvolumeclaims	pvc	v1	true	PersistentVolumeClaim
persistentvolumes	pv	v1	false	PersistentVolume
pods	po	v1	true	Pod
podtemplates		v1	true	PodTemplate
replicationcontrollers	rc	v1	true	ReplicationController
resourcequotas	quota	v1	true	ResourceQuota
secrets		v1	true	Secret
serviceaccounts	sa	v1	true	ServiceAccount
services	svc	v1	true	Service
mutatingwebhookconfigurations		admissionregistration.k8s.io/v1	false	MutatingWebhookConfig
validatingadmissionpolicies		admissionregistration.k8s.io/v1	false	ValidatingAdmissionPol
validatingadmissionpolicybindings		admissionregistration.k8s.io/v1	false	ValidatingAdmissionPol
ing		admissionregistration.k8s.io/v1	false	ValidatingWebhookConfig
validatingwebhookconfigurations		admissionregistration.k8s.io/v1	false	ValidatingWebhookConfig
n				
customresourcedefinitions	crd,crds	apiextensions.k8s.io/v1	false	CustomResourceDefinitio
apiservices		apiregistration.k8s.io/v1	false	APIService
controllerrevisions		apps/v1	true	ControllerRevision
daemonsets	ds	apps/v1	true	DaemonSet
deployments	deploy	apps/v1	true	Deployment
replicasetss	rs	apps/v1	true	ReplicaSet
statefulsets	sts	apps/v1	true	StatefulSet
selfsubjectreviews		authentication.k8s.io/v1	false	SelfSubjectReview
tokenreviews		authentication.k8s.io/v1	false	TokenReview
localsubjectaccessreviews		authorization.k8s.io/v1	true	LocalSubjectAccessRevie
selfsubjectaccessreviews		authorization.k8s.io/v1	false	SelfSubjectAccessReview
selfsubjectrulesreviews		authorization.k8s.io/v1	false	SelfSubjectRulesReview
subjectaccessreviews		authorization.k8s.io/v1	false	SubjectAccessReview
horizontalpodautoscalers	hpa	autoscaling/v2	true	HorizontalPodAutoscaler
cronjobs	cj	batch/v1	true	CronJob
jobs		batch/v1	true	Job

يعمل k8s بكل الـ resources المتاحة داخل الـ cluster

```

mostafa@k8s:~$ kubectl --help
kubectl controls the Kubernetes cluster manager.

Find more information at: https://kubernetes.io/docs/reference/kubectl/

Basic Commands (Beginner):
  create      Create a resource from a file or from stdin
  expose      Take a replication controller, service, deployment or pod and expose it as a network endpoint
  run         Run a particular image on the cluster
  set          Set specific features on objects

Basic Commands (Intermediate):
  explain     Get documentation for a resource
  get         Display one or many resources
  edit        Edit a resource on the server
  delete      Delete resources by file names, stdin, resources and names, or by resources and selector

Deploy Commands:
  rollout    Manage the rollout of a resource
  scale      Set a new size for a deployment, replica set, or replication controller
  autoscale  Auto-scale a deployment, replica set, stateful set, or replication controller

Cluster Management Commands:
  certificate  Modify certificate resources
  cluster-info  Display cluster information
  top          Display resource (CPU/memory) usage
  cordon       Mark node as unschedulable
  uncordon    Mark node as schedulable
  drain        Drain node in preparation for maintenance
  taint        Update the taints on one or more nodes

```

بشوفر ال help بتاع ال k8s

```

mostafa@k8s:~$ kubectl get pods --all-namespaces
NAMESPACE      NAME                READY   STATUS    RESTARTS   AGE
kube-system   coredns-7db6d8ff4d-4b8zs   1/1    Running   3 (18m ago) 56m
kube-system   etcd-minikube           1/1    Running   3 (18m ago) 56m
kube-system   kube-apiserver-minikube  1/1    Running   3 (16m ago) 56m
kube-system   kube-controller-manager-minikube  1/1    Running   3 (18m ago) 56m
kube-system   kube-proxy-fvjm7        1/1    Running   3 (18m ago) 56m
kube-system   kube-scheduler-minikube  1/1    Running   3 (18m ago) 56m
kube-system   storage-provisioner    1/1    Running   6 (15m ago) 56m
kubernetes-dashboard dashboard-metrics-scraper-b5fc48f67-jlrtg  1/1    Running   3 (18m ago) 56m
kubernetes-dashboard kubernetes-dashboard-779776cb65-vbn56   1/1    Running   5 (15m ago) 56m
mostafa@k8s:~$ 

```

يعمل list بكل ال pods ال في ال namespaces ولو لاحظت ان دي كل ال components كانت في ال master node

```

mostafa@k8s:~$ kubectl run nginx --image=nginx
pod/nginx created
mostafa@k8s:~$ 

```

يعمل nginx image من اسمها pod

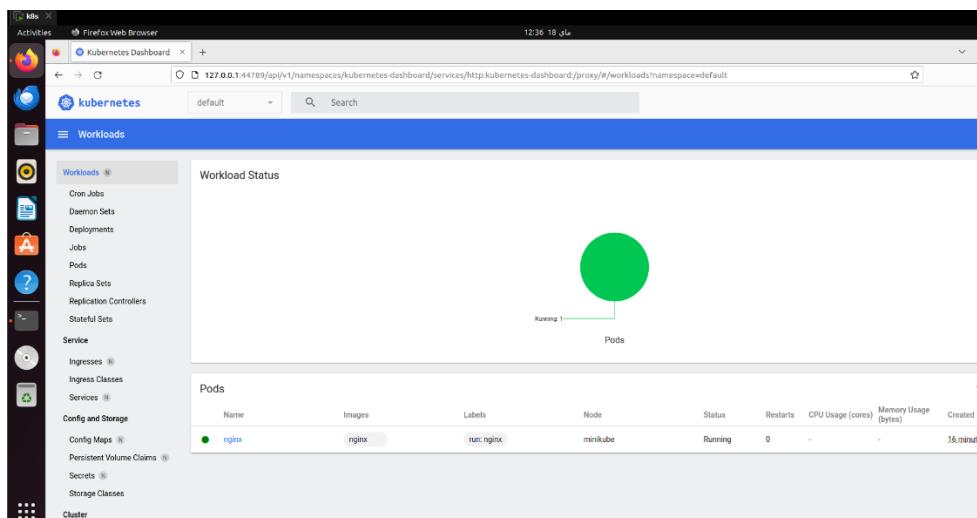
(طيب بيجب ال image دى منين ؟ من ال docker hub هو docker registry)

```
mostafa@k8s:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
nginx     1/1     Running   0          2m19s
mostafa@k8s:~$
```

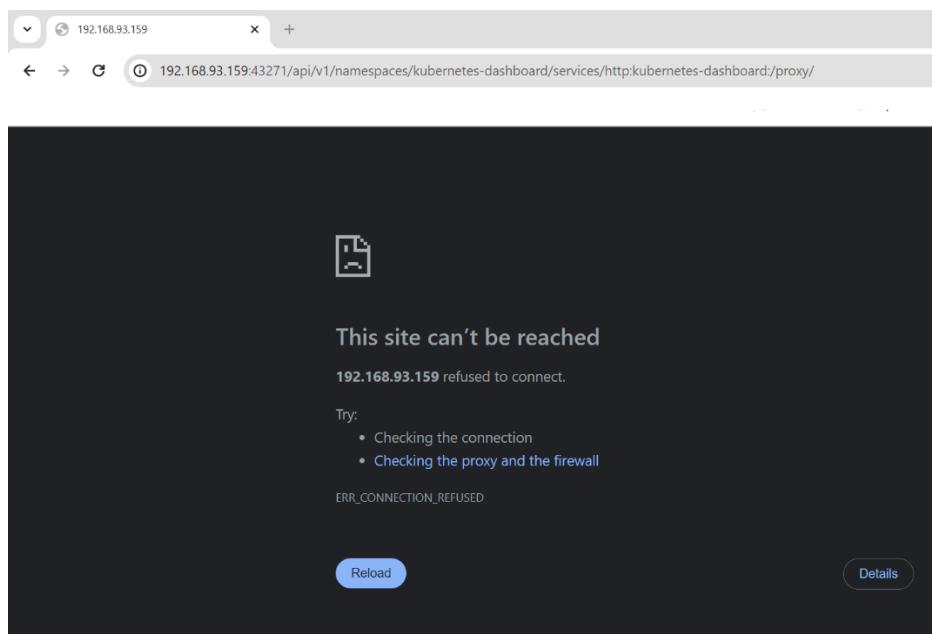
عمل list بال pods عندی

```
mostafa@k8s:~$ minikube dashboard
🟡 Verifying dashboard health ...
🟡 Launching proxy ...
🟡 Verifying proxy health ...
🟡 Opening http://127.0.0.1:44789/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
Gtk-Message: 12:35:31.597: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
```

عشان اشوف ال dashboard minikube هاخد اللينك دا وتكتبه فالمتصفح



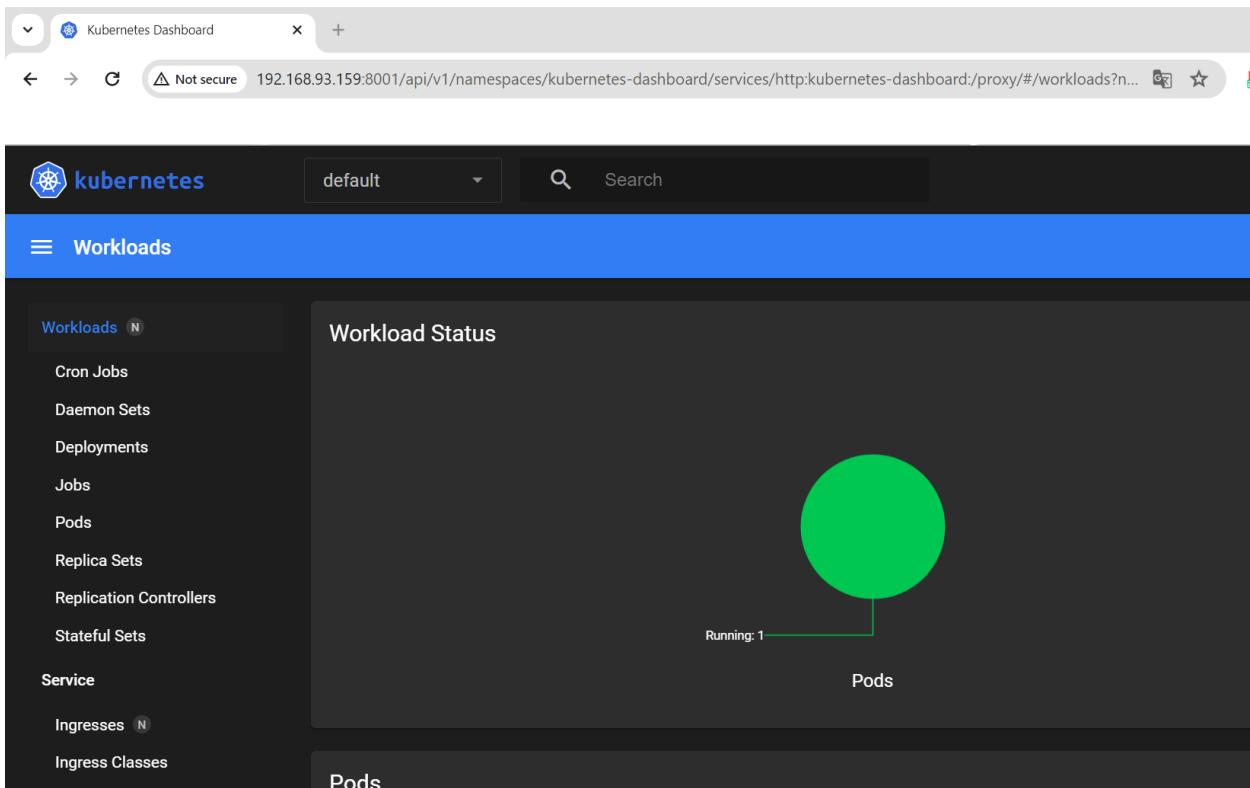
بس لازم افتحه على نفس ال vm ال عليها ال minikube لان لو فتحته من جهاز تاني مش هفتح



طيب لو عاوز افتحه من أي جهاز في ال network عندي هنكتب الامر دا

```
mostafa@k8s:~$ kubectl proxy --address='0.0.0.0' --disable-filter=true
W0518 12:39:16.852115    16015 proxy.go:177] Request filter disabled, your proxy is vulnerable to XSRF attacks, please be cautious
Starting to serve on [::]:8001
```

وخلی بالك هتفتح اللينك ع ال port دا 8001



```
mostafa@k8s:~$ kubectl delete pod test-774d7fd6bb-vmjnm
pod "test-774d7fd6bb-vmjnm" deleted
mostafa@k8s:~$
```

عمل `test-774d7fd6bb-vmjnm` دل `pod` delete لـ اسمه

: عشان اكتب `k8s yml file` في 4 حاجات أساسية لازم تكون موجوده :

version : بحدد ال `version` بتاعه وفي اكتر من `apiserver-1`

وبختار ال `version` على الحاجه ال انا محتاج اعملها فهل انا عاوز اعمل `pod` ف هستخدم

`V1` وهستخدم حاجه ثاني فهشوف هي في انهي `version` ودا مثال ع ال `versions` لان في `apiserver-1` كتيره `versions`

apiVersion

V1

Apps/v1

Networking.k8s.io/v1

storage.k8s.io/v1

pod

Deployment

Ingress

StorageClass

Namespace

Replicaset

ف لو عاوز اعمل pod هتسخدم v1 وهكذا

هنا بحدد نوع ال object الحتاج اعملها وليكن يحتاج اعمل pod kind-2

يعرف معلومات زيارة عن ال object ال هعمله هديله مثلا name معين او namespaces او labels

مواصفات ال object ال عاوز اعمله specification-4

مثال ع ال yaml file

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
    - name: nginx
      image: nginx:1.14.2
```

```

mostafa@k8s:~$ kubectl explain pod
KIND: Pod
VERSION: v1

DESCRIPTION:
Pod is a collection of containers that can run on a host. This resource is
created by clients and scheduled onto hosts.

FIELDS:
apiVersion <string>
APIVersion defines the versioned schema of this representation of an object.
Servers should convert recognized schemas to the latest internal value, and
may reject unrecognized values. More info:
https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

kind <string>
Kind is a string value representing the REST resource this object
represents. Servers may infer this from the endpoint the client submits
requests to. Cannot be updated. In CamelCase. More info:
https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds

metadata <ObjectMeta>
Standard object's metadata. More info:
https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

spec <PodSpec>
Specification of the desired behavior of the pod. More info:
https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status

status <PodStatus>
Most recently observed status of the pod. This data may not be up to date.
Populated by the system. Read-only. More info:
https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status

mostafa@k8s:~$ █

```

بیعرفنی از ای اکتب ال **yml file** الخاص بال **pod**

از ای ممکن اعمل **yml file** عشان اعمل **pod**
1- هعمل **.yml** ویکون **file**

```
mostafa@k8s:~$ vim pod.yml
```

2- هكتب ال **code** بتاعي

```

apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2

```

بقوله عاوز اعمل **pod** اسمها **nginx** من ال **image** **nginx:1.14.2**
3- هعمل **file** لـ **apply** دا

```

mostafa@k8s:~$ kubectl apply -f pod.yml
pod/nginx created
mostafa@k8s:~$ kubectl get pods
NAME      READY   STATUS            RESTARTS   AGE
nginx    0/1     ContainerCreating   0          18s
mostafa@k8s:~$ █

```

الفرق بين ال Imperative – Declarative

cli هي ال command ال بكتها في ال Imperative-

```
mostafa@k8s:~$ kubectl run redis --image=redis
pod/redis created
mostafa@k8s:~$ █
```

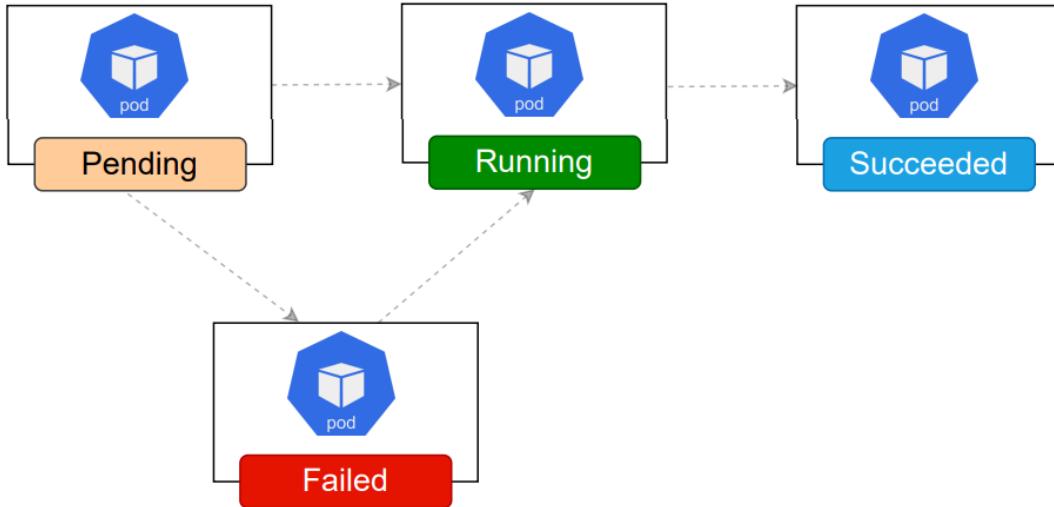
هي طريقة ال Declarative-

```
mostafa@k8s:~$ kubectl apply -f pod.yml
pod/redis2 created
mostafa@k8s:~$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: redis2
spec:
  containers:
    - name: redis
      image: redis
```

```
mostafa@k8s:~$ █
```

```
mostafa@k8s:~$ kubectl get pods
NAME     READY   STATUS    RESTARTS   AGE
redis   1/1     Running   0          2m31s
redis2  1/1     Running   0          70s
mostafa@k8s:~$ █
```

: Life cycle of a k8s pod



1- دى الفترة ال بيتم فيها انشاء ال pod وفيها بيتمن حاجتين

- ان ال Scheduler بيقرر ان ال pod هيتعمل في انهي

- ان ال kubelet هيدا يحمل ال image ال هيتعمل منها ال pod

2- بيحصل حاجه من اتنين لو الدنيا تمام ال status running ه تكون run بقا run وال running ال بداخله تمام و container

لو في اي خطأ هيحصل failed فممك ترجع تعدل الخطأ ال ممكن يكون خطأ مثلا في ال image او انت كاتب حاجه غلط وترجع تبني running

3- يعني ال pod تمام و عمل ال task succeeded

```
apiVersion: v1
kind: Pod
metadata:
  name: redis2
spec:
  containers:
    - name: redis
      image: redis
```

ال name ال في ال metadata هو اسم ال pod و الاسم ال في ال spec تحت ال container اسم ال container ال داخل ال pod

```
mostafa@k8s:~$ kubectl get pods -o wide
NAME    READY   STATUS    RESTARTS   AGE     IP           NODE   NOMINATED NODE   READINESS GATES
redis   1/1     Running   0          18m    10.244.0.20   minikube   <none>        <none>
redis2  1/1     Running   0          17m    10.244.0.21   minikube   <none>        <none>
mostafa@k8s:~$
```

عمل list لكل ال pod بمعلومات اكتر شويا

```
mostafa@k8s:~$ kubectl get pods -o yaml
apiVersion: v1
items:
- apiVersion: v1
  kind: Pod
  metadata:
    creationTimestamp: "2024-05-18T11:44:54Z"
    labels:
      run: redis
    name: redis
    namespace: default
    resourceVersion: "10403"
    uid: 842f9a2b-8b03-491a-b0a1-f202c0994cee
  spec:
    containers:
    - image: redis
      imagePullPolicy: Always
      name: redis
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
      volumeMounts:
      - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
        name: kube-api-access-hdd8m
        readOnly: true
    dnsPolicy: ClusterFirst
    enableServiceLinks: true
    nodeName: minikube
    preemptionPolicy: PreemptLowerPriority
    priority: 0
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    serviceAccount: default
    serviceAccountName: default
    terminationGracePeriodSeconds: 30
    tolerations:
    - effect: NoExecute
      key: node.kubernetes.io/not-ready
      operator: Exists
      tolerationSeconds: 300
    - effect: NoExecute
```

عرض ال pod بطريقه ال yml file

```
mostafa@k8s:~$ kubectl exec -it redis -- /bin/bash
root@redis:/data#
```

بدخل في ال bash بتاع ال pod ال اسمه redis

```
mostafa@k8s:~$ kubectl describe pod redis
Name:           redis
Namespace:      default
Priority:       0
Service Account: default
Node:           minikube/192.168.49.2
Start Time:     Sat, 18 May 2024 14:44:54 +0300
Labels:         run=redis
Annotations:    <none>
Status:         Running
IP:            10.244.0.20
IPs:
  IP: 10.244.0.20
Containers:
  redis:
    Container ID: docker://c6b8307652ad2e62a4d14507ee4b2160ddeabd5ac86544f99c7ab53f6564456f
    Image:          redis
    Image ID:      docker-pullable://redis@sha256:5a93f6b2e391b78e8bd3f9e7e1e06aeb5295043b4703fb88392835cec924a0
    Port:          <none>
    Host Port:    <none>
    State:         Running
      Started:   Sat, 18 May 2024 14:45:09 +0300
    Ready:         True
    Restart Count: 0
    Environment:   <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-hdd8m (ro)
Conditions:
  Type        Status
  PodReadyToStartContainers  True
  Initialized  True
  Ready        True
  ContainersReady  True
  PodScheduled  True
Volumes:
  kube-api-access-hdd8m:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:   kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:    true
  QoS Class:  BestEffort
  Node-Selectors: <none>
```

عرض معلومات كتير عن ال pod ال اسمه redis ومن ضمن المعلومات دي حاجه اسمها running ودي يتعرضلي زي log لـ pod دا والراحل ال مر بيها ال pod عشان يكون Event

Events:				
Type	Reason	Age	From	Message
Normal	Scheduled	28m	default-scheduler	Successfully assigned default/redis to minikube
Normal	Pulling	28m	kubelet	Pulling image "redis"
Normal	Pulled	28m	kubelet	Successfully pulled image "redis" in 14.549s (14.549s including waiting). Image size: 116496163 bytes.
Normal	Created	28m	kubelet	Created container redis
Normal	Started	28m	kubelet	Started container redis

```
mostafa@k8s:~$ #####impreative#####
mostafa@k8s:~$ kubectl edit pod redis
pod/redis edited
```

بعمل تعديل على ال pod ال اسمه redis بطريقة ال imperative

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: "2024-05-18T11:44:54Z"
  labels:
    run: redis
  name: redis
  namespace: default
  resourceVersion: "12450"
  uid: 842f9a2b-8b03-491a-b0a1-f202c0994cee
spec:
  containers:
    - image: redis:latst
      imagePullPolicy: Always
      name: redis
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
      volumeMounts:
        - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
          name: kube-api-access-hdd8m
            readOnly: true
```

```
apiVersion: v1
kind: Pod
metadata:
  name: redis2
spec:
  containers:
    - name: redis
      image: redis:7.2.4
```

بعمل تعديل على pod بطريقة ال declarative

```
mostafa@k8s:~$ kubectl apply -f pod.yml
pod/redis2 configured
mostafa@k8s:~$
```

لما برجع اعمل apply لـ file مش بيعمل pod جديد هو بيعرف ان فيه pod عنده معمول من ال file دا في بيعمل search في ال file عشان لو فيه أي تعديلات ينفذ التعديلات دي فقط

```
mostafa@k8s:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
redis2   1/1     Running   1 (44s ago)  49m
mostafa@k8s:~$
```

```
mostafa@k8s:~$ kubectl get pods -o yaml
apiVersion: v1
items:
- apiVersion: v1
  kind: Pod
  metadata:
    annotations:
      kubectl.kubernetes.io/last-applied-configuration: |
        {"apiVersion":"v1","kind":"Pod","metadata":{"annotations":{},"name":"redis2","namespace":"default"},"spec":{"containers":[{"image":"redis:7.2.4","name":"redis"}]}}
      creationTimestamp: "2024-05-18T11:46:15Z"
    name: redis2
    namespace: default
    resourceVersion: "12900"
    uid: ac8562a3-95c7-4341-8413-83ffb8f19aa6
```

التغييرات ال بتعمد ع ال pod بطرقه ال Declarative هي ال بتظهر في ال annotations لكن طريقه Imperative مش بيظهر التغييرات بتاعتتها في ال annotations

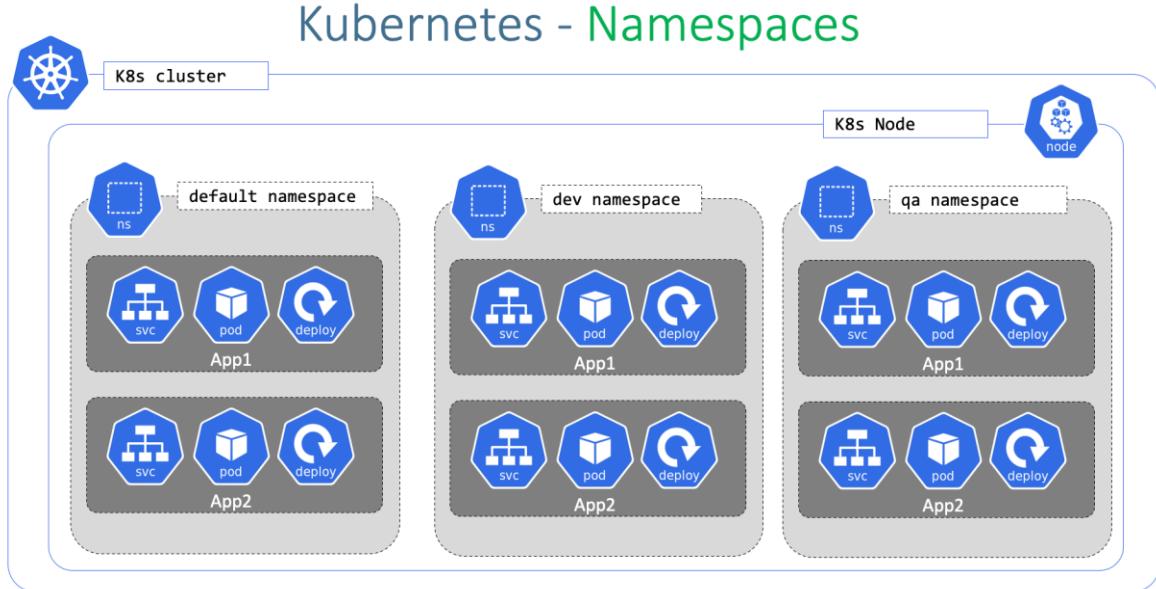
```
mostafa@k8s:~$ kubectl logs redis2
1:C 18 May 2024 12:35:34.591 * o000o000o000 Redis is starting o000o000o000
1:C 18 May 2024 12:35:34.591 * Redis version=7.2.4, bits=64, commit=00000000, modified=0, pid=1, just started
1:C 18 May 2024 12:35:34.591 # Warning: no config file specified, using the default config. In order to specify
conf
1:M 18 May 2024 12:35:34.592 * monotonic clock: POSIX clock_gettime
1:M 18 May 2024 12:35:34.592 * Running mode=standalone, port=6379.
1:M 18 May 2024 12:35:34.593 * Server initialized
1:M 18 May 2024 12:35:34.593 * Ready to accept connections tcp
mostafa@k8s:~$
```

بشوف ال logs بتاع ال pod ال اسمه redis2

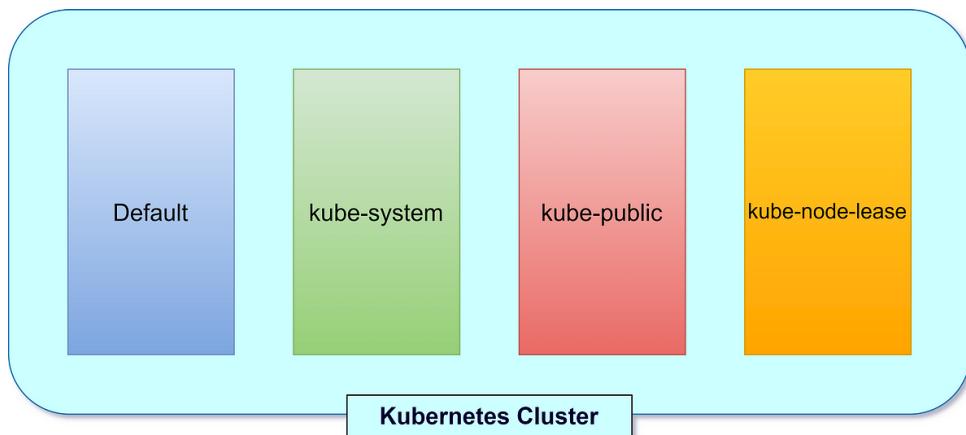
```
mostafa@k8s:~$ kubectl delete pod --all
pod "redis2" deleted
mostafa@k8s:~$ kubectl get pods
No resources found in default namespace.
mostafa@k8s:~$
```

بحذف كل ال pods ال عندي

cluster resources isolating ال داخل ال K8s Namespaces : من خلالها بعمل cluster معزولة عن أي حاجه داخل ال namespace دي بتكون داخل ال namespace اي حاجه تانية داخل ال cluster اقدر اعمل اكتر من NS داخل ال



يبكون في ال cluster 4 default namespace وهم :
 -1 default namespace
 -2 kube public
 -3 kube system
 -4 kube node lease



```
mostafa@k8s:~$ kubectl get namespaces
NAME          STATUS  AGE
default       Active  32s
kube-node-lease Active  32s
kube-public   Active  32s
kube-system   Active  32s
```

عمل list ns ال عندي

```
mostafa@k8s:~$ kubectl create namespace env
namespace/env created
mostafa@k8s:~$ 
mostafa@k8s:~$ 
mostafa@k8s:~$ 
```

عمل NS create ل اسمه env بطريقة ال Imperative

```
mostafa@k8s:~$ kubectl get namespaces
NAME          STATUS  AGE
default       Active  89s
env          Active  5s
kube-node-lease Active  89s
kube-public   Active  89s
kube-system   Active  89s
mostafa@k8s:~$ 
```

```
apiVersion: v1
kind: Namespace
metadata:
  name: production
```

عمل NS اسمه production بس بطريقة ال Declarative

```
mostafa@k8s:~$ kubectl apply -f ns.yml
namespace/production created
mostafa@k8s:~$ kubectl get namespaces
NAME          STATUS  AGE
default       Active  2m10s
env          Active  46s
kube-node-lease Active  2m10s
kube-public   Active  2m10s
kube-system   Active  2m10s
production    Active  6s
mostafa@k8s:~$ 
```

```
mostafa@k8s:~$ kubectl get pods
No resources found in default namespace.
mostafa@k8s:~$ █
```

كدا بعمل list لكل ال pods بس الموجودة في ال default namespace

```
mostafa@k8s:~$ kubectl get pods -n kube-system
NAME                               READY   STATUS    RESTARTS   AGE
coredns-7db6d8ff4d-4b8zs          1/1     Running   4 (3m49s ago)  31d
etcd-minikube                      1/1     Running   4 (3m49s ago)  31d
kube-apiserver-minikube           1/1     Running   4 (3m49s ago)  31d
kube-controller-manager-minikube  1/1     Running   4 (3m49s ago)  31d
kube-proxy-fvjm7                  1/1     Running   4 (3m49s ago)  31d
kube-scheduler-minikube           1/1     Running   4 (3m49s ago)  31d
storage-provisioner                1/1     Running   8 (3m10s ago)  31d
mostafa@k8s:~$ █
```

كدا بعمل list لكل ال pods بس الموجودة في ال NS ال اسمها kube-system

```
mostafa@k8s:~$ kubectl delete namespace env
namespace "env" deleted
```

عمل NS delete ل اسمه env

```
mostafa@k8s:~$ kubectl describe namespace production
Name:           production
Labels:         kubernetes.io/metadata.name=production
Annotations:   <none>
Status:        Active

No resource quota.

No LimitRange resource.
mostafa@k8s:~$ █
```

عرض معلومات اكتر عن ال NS ال اسمه production

```
mostafa@k8s:~$ kubectl run pod-ns --image=redis --namespace=production
pod/pod-ns created
mostafa@k8s:~$ █
```

كدا بعمل pod اسمه pod-ns من اسمها image redis ويكون في ال NameSpace production ال اسمه

```
mostafa@k8s:~$ kubectl get pod -n production
NAME      READY   STATUS    RESTARTS   AGE
pod-ns   1/1     Running   0          90s
mostafa@k8s:~$
```

عمل list بكل ال pods ال موجودة في ال NS اسمها production

```
apiVersion: v1
kind: Pod
metadata:
  name: nspod
  namespace: production
spec:
  containers:
    - name: nspod
      image: redis
  ~
  ~
```

عمل declarative production واعمله assign ل Namespace اسمه Production بطريقة ال

```
mostafa@k8s:~$ kubectl apply -f nspod.yml
pod/nspod created
mostafa@k8s:~$ kubectl get pod -n production
NAME      READY   STATUS    RESTARTS   AGE
nspod     1/1     Running   0          8s
pod-ns   1/1     Running   0          6m2s
mostafa@k8s:~$
```

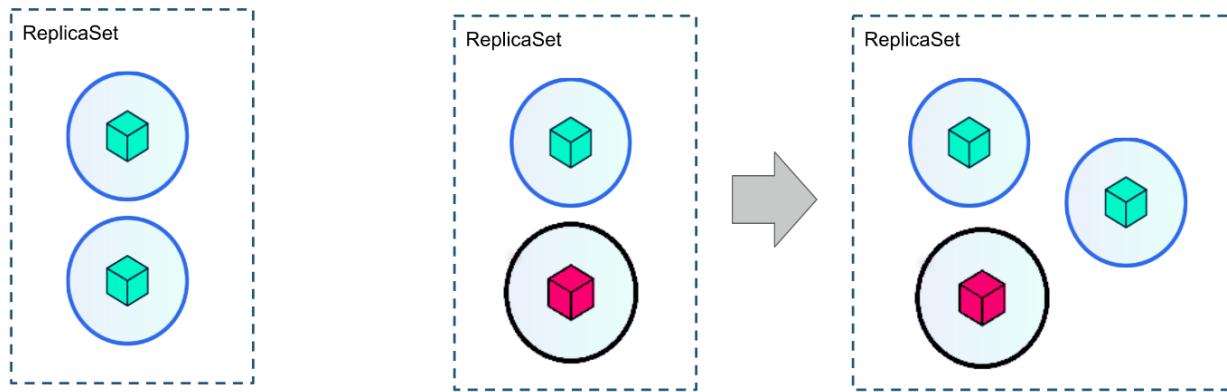
الاتنين واحد بيعملوا Replica : K8s ReplicaSets or K8s Replication Controller لكن ال Replication Controller هي طريقة قديمة

أي هي مهام ال ReplicaSets :

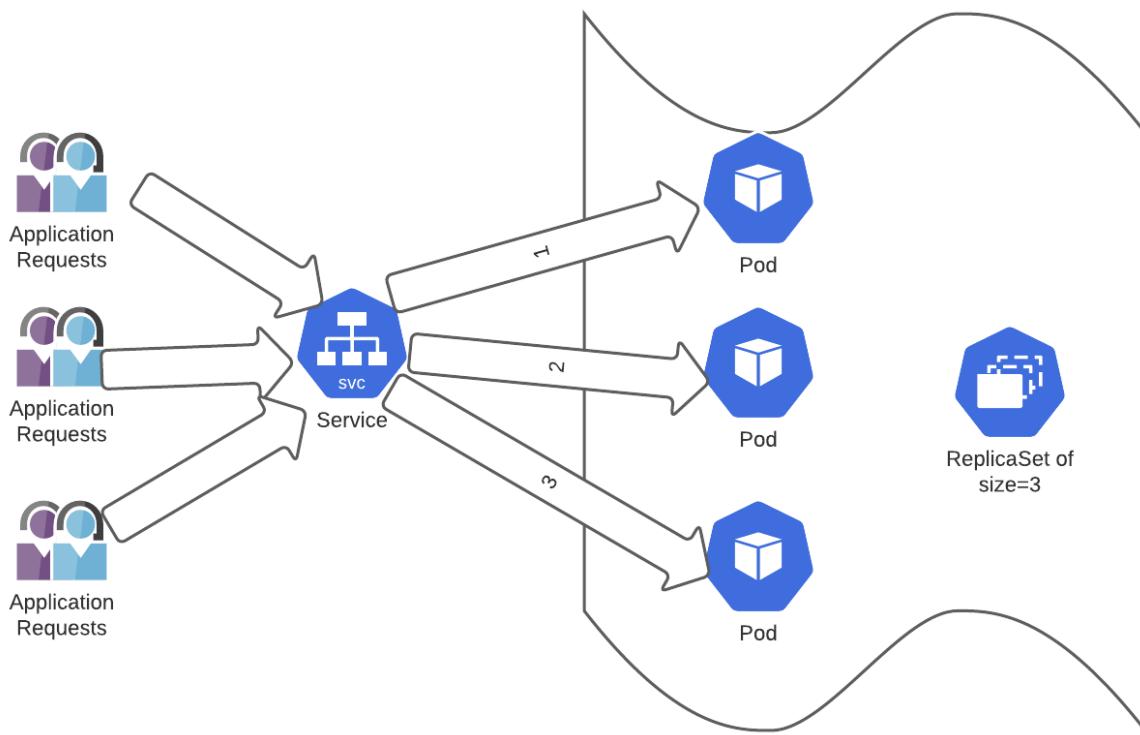
- ان أي pod بداخلها لو اتعلمه delete او حصل فيه failed تبدا تعمل pod جديد مكانه بنفس مواصفاته

- بتوزع ال Request بين ال Pods ال بداخلها يعني بتعمل بينهم

(load Balancing and High Availability)



مثال فالصوره الاولى من ناحيه اليسار عندي RS فيها 2pod – الصوره ال بعدها بتقول ان فيه فيهم حصله failed – الصوره الثالثه بتوضح وظيفة ال RS وهي أنها عملت Pod جديد بنفس مواصفات ال حصله Failed



هنا توضيح لـ RS ازاي بتوزع ال Request بين ال Pods

ازاي اكتب ال yml file الخاص بال RS

```

apiVersion: apps/v1      #1
kind: ReplicaSet          #2
metadata:
  name: website           #3
spec:
  replicas: 10            #6
  selector:
    matchLabels:
      env: prod             #9
  template:
    metadata:               #11
      labels:
        env: prod             #13
    spec:
      containers:            #15
      - name: redis            #16
        image: redis            #17

```

ال apiversion الخاص ب RS وهو ال apps/v1	1
ال kind اني بوضح ال object ال عاوز اعمله وهو ال RS	2
ال RS metadata الخاصة بال	3
ال name الخاص بال RS	4
ال spec الخاص بال RS	5
ال replication وهي عدد ال replicas ال هيكون عندي	6
ال selector خاص بال rs وبووضح فيه ال label	7
ال pod matchLabels بوضوح اسم ال label ال هيكون للpod	8
ال label هيكون اسم ال env: prod	9
ال pod Template وكل ال تحتها بتكون مواصفات ال pod	10
ال pod metadata الخاصة بال	11
ال labels ال هيكون ع ال pod ولازم دي تساوي ال matchLabels ال في selector ال في RS spec	12
ال matchLabels env: prod اسم ال label ولاحظ ان هو ال في ال	13
ال pod spec الخاصة بال	14
ال container ال هيكون داخل ال pod	15
اسم ال container ال هيكون داخل ال pod	16
ال Pod image ال هيتعمل منها ال	17

```
mostafa@k8s:~$ vim rs.yml
mostafa@k8s:~$ kubectl apply -f rs.yml
replicaset.apps/website created
mostafa@k8s:~$
```

عمل list لـ apply

```
mostafa@k8s:~$ kubectl get rs
NAME      DESIRED   CURRENT   READY   AGE
website   3          3          3       107s
mostafa@k8s:~$
```

عمل list لـ ReplicaSet ال عندي

```
mostafa@k8s:~$ kubectl get po
NAME        READY   STATUS    RESTARTS   AGE
website-bdrhq 1/1     Running   0          4m32s
website-rrm7k  1/1     Running   0          4m32s
website-wxjbn  1/1     Running   0          4m32s
mostafa@k8s:~$
```

عمل list لـ pod ال عندي

لاحظ ان اسم ال pod هو اسم ال RS + اسم ال pod

```
mostafa@k8s:~$ kubectl get pod --show-labels
NAME        READY   STATUS    RESTARTS   AGE   LABELS
website-bdrhq 1/1     Running   0          74m   env=prod
website-rrm7k  1/1     Running   0          74m   env=prod
website-wxjbn  1/1     Running   0          74m   env=prod
mostafa@k8s:~$
```

عمل list لـ pod ال عندي وعرض معاها ال label بتابعها

طيب لو عملت `pod delete` ل RS من ال هيحصل أي ؟

```
mostafa@k8s:~$ kubectl delete pod website-wxjbn
pod "website-wxjbn" deleted
mostafa@k8s:~$
```

عملت `pod delete`

```
mostafa@k8s:~$ kubectl get pod
NAME          READY   STATUS    RESTARTS   AGE
website-bdrhq 1/1     Running   0          82m
website-gzms9  1/1     Running   0          43s
website-rrm7k  1/1     Running   0          82m
mostafa@k8s:~$
```

عطول ال RS عملت مكانه `pod` جديد

```
mostafa@k8s:~$ kubectl scale rs website --replicas=6
replicaset.apps/website scaled
```

عمل `Scale up` لـ `Replica` هي اني بزود عدد ال `replica` لـ `6`

```
mostafa@k8s:~$ kubectl get rs
NAME      DESIRED   CURRENT   READY   AGE
website   6          6          6       94m
```

```
mostafa@k8s:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
website-bdrhq 1/1     Running   0          95m
website-bjwg8  1/1     Running   0          84s
website-fv5nc  1/1     Running   0          84s
website-gzms9  1/1     Running   0          13m
website-rrm7k  1/1     Running   0          95m
website-z546l  1/1     Running   0          84s
mostafa@k8s:~$
```

وقدر اعمل ال scale up عن طريق ال yml file وهي اني بعدل في الرقم الخاص بال replica

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: website
spec:
  replicas: 10
  selector:
    matchLabels:
      env: prod
  template:
    metadata:
      labels:
        env: prod
    spec:
      containers:
        - name: redis
          image: redis
```

خليل الرقم 10 وكان عندي pod 4 عشان يوصلهم لـ 10

```
mostafa@k8s:~$ vim rs.yml
mostafa@k8s:~$ kubectl apply -f rs.yml
replicaset.apps/website configured
mostafa@k8s:~$
```

```
mostafa@k8s:~$ kubectl get po
NAME           READY   STATUS    RESTARTS   AGE
website-bdrhq  1/1     Running   0          101m
website-bjwg8  1/1     Running   0          7m44s
website-bmm5x  1/1     Running   0          15s
website-bs679  1/1     Running   0          15s
website-fv5nc  1/1     Running   0          7m44s
website-gzms9  1/1     Running   0          20m
website-k4bd5  1/1     Running   0          15s
website-nv5kt  1/1     Running   0          15s
website-rrm7k  1/1     Running   0          101m
website-z546l  1/1     Running   0          7m44s
mostafa@k8s:~$
```

```
mostafa@k8s:~$ kubectl get rs
NAME   DESIRED   CURRENT   READY   AGE
website  10        10        10      103m
mostafa@k8s:~$
```

```
mostafa@k8s:~$ kubectl scale rs website --replicas=4
replicaset.apps/website scaled
mostafa@k8s:~$
```

عمل scale down وهي اني بقلل عدد ال Replica ال عندي من 10 ل 4

```
mostafa@k8s:~$ kubectl get rs
NAME      DESIRED   CURRENT   READY   AGE
website   4         4         4       106m
mostafa@k8s:~$
```

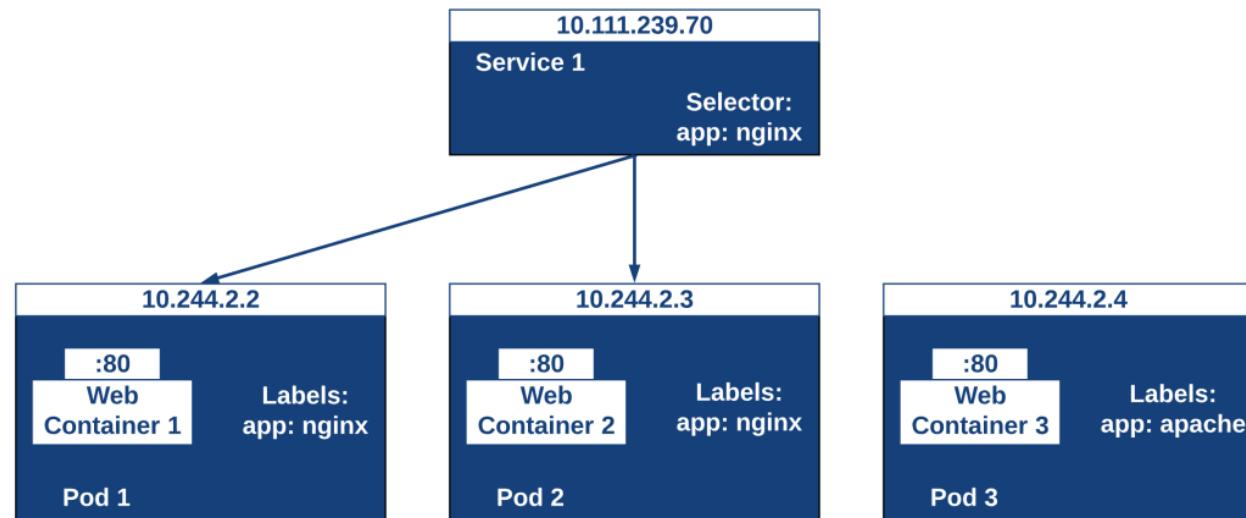
```
mostafa@k8s:~$ kubectl get pods
NAME        READY   STATUS    RESTARTS   AGE
website-bdrhq  1/1    Running   0          106m
website-gzms9  1/1    Running   0          25m
website-rrm7k  1/1    Running   0          106m
website-z546l  1/1    Running   0          12m
mostafa@k8s:~$
```

وقدر نفس الكلام اني اعمل ال scale down من ال yml file

```
mostafa@k8s:~$ kubectl delete rs website
replicaset.apps "website" deleted
mostafa@k8s:~$ kubectl get rs
No resources found in default namespace.
mostafa@k8s:~$
```

عمل website لـ Replica Set Delete

Labels in k8s هي عبارة key and value تكون attached لـ object زي ال pods وبقدر انظم ال object داخل ال k8s لمجموعات فرعية-ممكن اضع ال label وانا بـ create pods او بعد كذا اضيفه في ال runtime وال key يكون unique



```
mostafa@k8s:~$ kubectl label pod nginx project=test
pod/nginx labeled
mostafa@k8s:~$ █
```

عمل create label ل pod nginx key اسمه project value بتاعه كلمه بتاعتها كلمه test

```
mostafa@k8s:~$ kubectl get pod --show-labels
NAME      READY   STATUS    RESTARTS   AGE      LABELS
nginx     1/1     Running   0          5m8s    project=test,run=nginx
mostafa@k8s:~$
```

عمل list لل pod بالإضافة الى اظهار ال label الخاص بال

metadata : بتكتب في ال Label yml file

```
apiVersion: v1
kind: Pod
metadata:
  name: label-demo
  labels:
    environment: production
    app: nginx
spec:
  containers:
    - name: nginx
      image: nginx:1.14.2
      ports:
        - containerPort: 80
```

```
"metadata": {  
    "labels": {  
        "key1" : "value1",  
        "key2" : "value2"  
    }  
}
```

:Label Selectors

!= يعني بقوله ياما = او quality-based requirements-1

environment = production

tier != frontend

```
mostafa@k8s:~$ kubectl get pod --selector project=test
NAME      READY   STATUS    RESTARTS   AGE
nginx    1/1     Running   0          22m
mostafa@k8s:~$
```

```
mostafa@k8s:~$ kubectl get pod -l project=test
NAME      READY   STATUS    RESTARTS   AGE
nginx    1/1     Running   0          28m
mostafa@k8s:~$
```

عمل زي سيرش ع ال pod بتاعه label project=test بطريقه
quality-based requirements

'key in (value) ودي بتكون مختلفة شويا' set-based requirements-2

environment in (production, qa)
tier notin (frontend, backend)

```
mostafa@k8s:~$ kubectl get pod --selector 'project in (test)'
NAME      READY   STATUS    RESTARTS   AGE
nginx    1/1     Running   0          25m
mostafa@k8s:~$
```

```
mostafa@k8s:~$ kubectl get pod -l 'project in (test)'
NAME      READY   STATUS    RESTARTS   AGE
nginx    1/1     Running   0          29m
mostafa@k8s:~$
```

عمل زي سيرش ع ال pod بتاعه label project=test بطريقه
set-based requirements

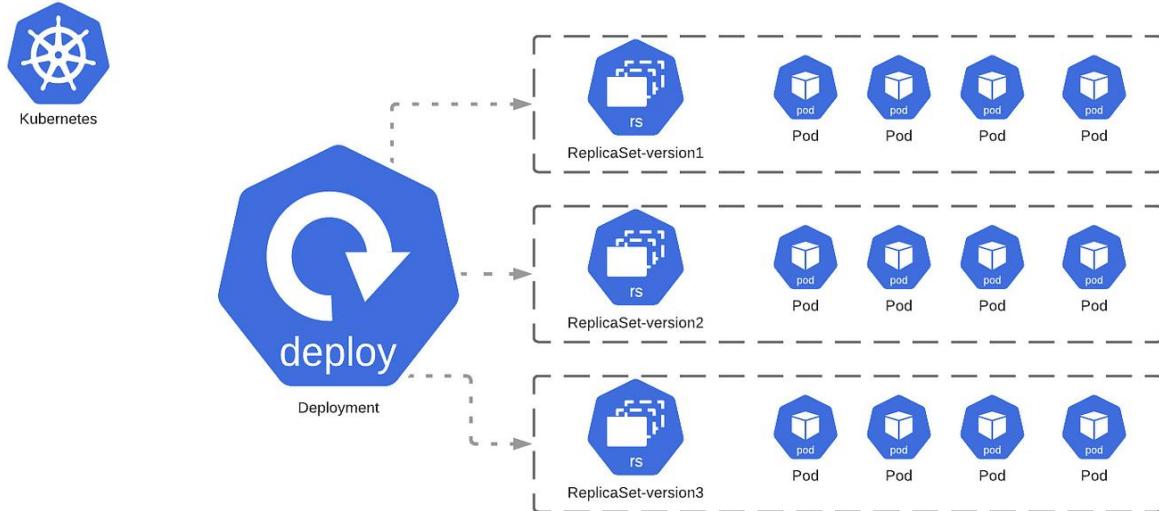
: selector yml file

```
selector:  
  component: redis
```

OR

```
"selector": {  
  "component" : "redis",  
}
```

تعتبر من ال object في ال k8s ووظيفتها انها بتعمل Manage لـ K8s Deployment desired state وال Current state وال Replicasets وبتحاول دايما تخليهم متساوين



```
mostafa@k8s:~$ kubectl create deploy testdeploy --image=redis --replicas=3
deployment.apps/testdeploy created
mostafa@k8s:~$
```

اعمل deploy اسمها testdeploy من image redis وعدد ال rs (ال هما ال pod ال تحت RS) هيكون 3

```
mostafa@k8s:~$ kubectl get deploy
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
testdeploy 3/3     3           3           45s
mostafa@k8s:~$
```

عمل list بكل ال deploy ال عندي

```
mostafa@k8s:~$ kubectl delete deploy testdeploy
deployment.apps "testdeploy" deleted
mostafa@k8s:~$
```

عمل delete لكل ال deploy ال عندي

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: db-deployment
  labels:
    DB: website
spec:
  replicas: 3
  selector:
    matchLabels:
      DB: website
  template:
    metadata:
      labels:
        DB: website
    spec:
      containers:
        - name: redis
          image: redis
```

بعمل اسمها deploy من db-deployment واسمها redis وعدد ال rs (ال هما ال pod ال تحت ال RS) هيكون 3 (عن طريق ال yml file)

```
mostafa@k8s:~$ kubectl apply -f deploy.yml
deployment.apps/db-deployment created
mostafa@k8s:~$
```

عمل yml file ل apply

```
mostafa@k8s:~$ kubectl get deploy
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
db-deployment  0/3     3            0           47s
mostafa@k8s:~$
```

عمل list ل deploy ال عندي

```
mostafa@k8s:~$ kubectl describe deploy db-deployment
Name:           db-deployment
Namespace:      default
CreationTimestamp: Mon, 20 May 2024 18:26:14 +0300
Labels:          DB=website
Annotations:    deployment.kubernetes.io/revision: 1
Selector:        DB=website
Replicas:       3 desired | 3 updated | 3 total | 2 available | 1 unavailable
StrategyType:   RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  DB=website
  Containers:
    redis:
      Image:      redis
      Port:       <none>
      Host Port: <none>
      Environment: <none>
      Mounts:     <none>
      Volumes:    <none>
```

عرض معلومات اكتر عن ال deploy ال اسمها db-deployment

```
mostafa@k8s:~$ kubectl get deploy
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
db-deployment 3/3      3           3           127m
mostafa@k8s:~$ 
mostafa@k8s:~$ 
mostafa@k8s:~$ kubectl get rs
NAME            DESIRED   CURRENT   READY   AGE
db-deployment-67d8c78bcb  3         3         3       127m
mostafa@k8s:~$ kubectl get po
NAME                           READY   STATUS    RESTARTS   AGE
db-deployment-67d8c78bcb-f2ss9  1/1     Running   0          128m
db-deployment-67d8c78bcb-gp5k6  1/1     Running   0          128m
db-deployment-67d8c78bcb-mtxb9  1/1     Running   0          128m
mostafa@k8s:~$
```

لما بعمل deploy جديدة بتبدا تعملني ال rs وال pod تحتها

```
mostafa@k8s:~$ kubectl delete rs db-deployment-67d8c78bcb
replicaset.apps "db-deployment-67d8c78bcb" deleted
mostafa@k8s:~$ kubectl get rs
NAME          DESIRED   CURRENT   READY   AGE
db-deployment-67d8c78bcb   3         3         1       3s
mostafa@k8s:~$ kubectl get po
NAME          READY   STATUS    RESTARTS   AGE
db-deployment-67d8c78bcb-mz7dt   1/1     Running   0          49s
db-deployment-67d8c78bcb-vbrvl   1/1     Running   0          49s
db-deployment-67d8c78bcb-ww9mk   1/1     Running   0          49s
mostafa@k8s:~$
```

لو حذفت RS تلقائياً ال deploy بيعمل واحدة مكانها

```
mostafa@k8s:~$ kubectl delete pod db-deployment-67d8c78bcb-ww9mk
pod "db-deployment-67d8c78bcb-ww9mk" deleted
mostafa@k8s:~$ kubectl get po
NAME          READY   STATUS    RESTARTS   AGE
db-deployment-67d8c78bcb-mz7dt   1/1     Running   0          5m10s
db-deployment-67d8c78bcb-rnm4b   1/1     Running   0          3s
db-deployment-67d8c78bcb-vbrvl   1/1     Running   0          5m10s
mostafa@k8s:~$
```

لو حذفت ال pod تلقائياً بيتعمل pod جديد

```
mostafa@k8s:~$ kubectl delete deploy db-deployment
deployment.apps "db-deployment" deleted
mostafa@k8s:~$ kubectl get deploy
No resources found in default namespace.
mostafa@k8s:~$ kubectl get rs
No resources found in default namespace.
mostafa@k8s:~$ kubectl get pod
No resources found in default namespace.
mostafa@k8s:~$
```

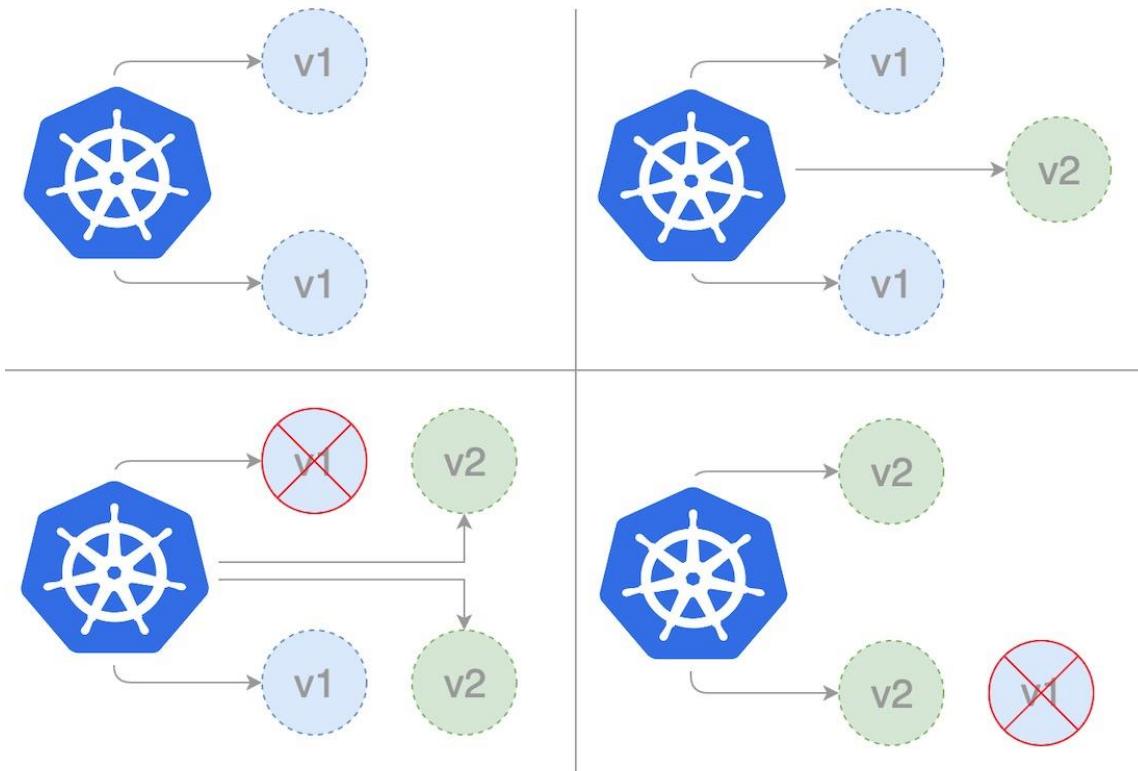
لو حذفت ال deploy كل ال تحتها سيتم حذفه

Deployment Strategies : يعني از اي تعمل update لـ APP شغال عندك من غير ما يكون فيه downtime – هي عبارة عن عمل Upgrade لـ APP بهدف انه يحصل من غير أي Automate process لـ updateing وعمل

أنواع ال Deployment Strategies :

Rolling Update Deployment-1 : هو النوع ال Default Replaces ى بتعمل Pods one by one ودا مفيد لأن باقي ال System بيكون Active لأن هو بيعمل ال Update في واحد فقط ويخلصه ويعمل لـ update تقل شويا لأن اثناء ال update بخسر بعض ال pos ال هي بيتعملها ال update

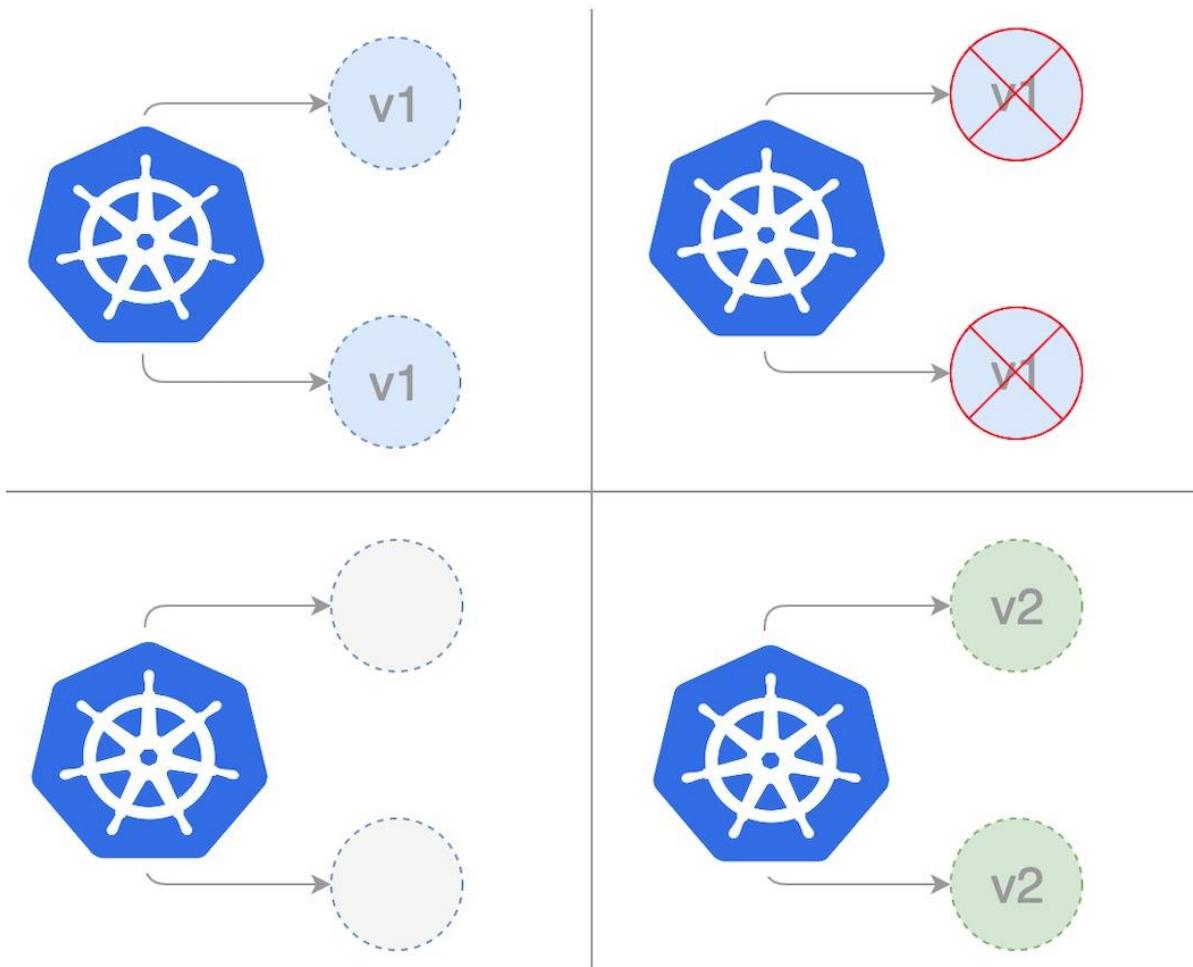
Max unavailable	Max surge
هنا بحدده عدد ال Pods ال هيتعملها ال update مرره واحدة مثلا 2pods او 4pods في ال update	بحدد ال max number من ال Pods ال يتعملاه ولبيكون لو عندي مثلا 50replica وعمل update ممكن ازود ال replica ديل عدد معين Performance لما اعمل ال update ميقلش



pod/recreate-deployment-d7998b49b-hwztz	0/1	ContainerCreating	0	19s
pod/recreate-deployment-d7998b49b-m2tv8	0/1	ContainerCreating	0	19s
pod/recreate-deployment-d7998b49b-z7g5c	1/1	Running	0	19s

كدا في pod واحد ال running والاتنين بيتعملهم update

Recreate update Deployment-2 : هنا بعمل shutdown لكل ال pods وتعمل replaces لـ new pods وهذا ال system هيكون فيه downtime لأن كل ال pods بتوقف

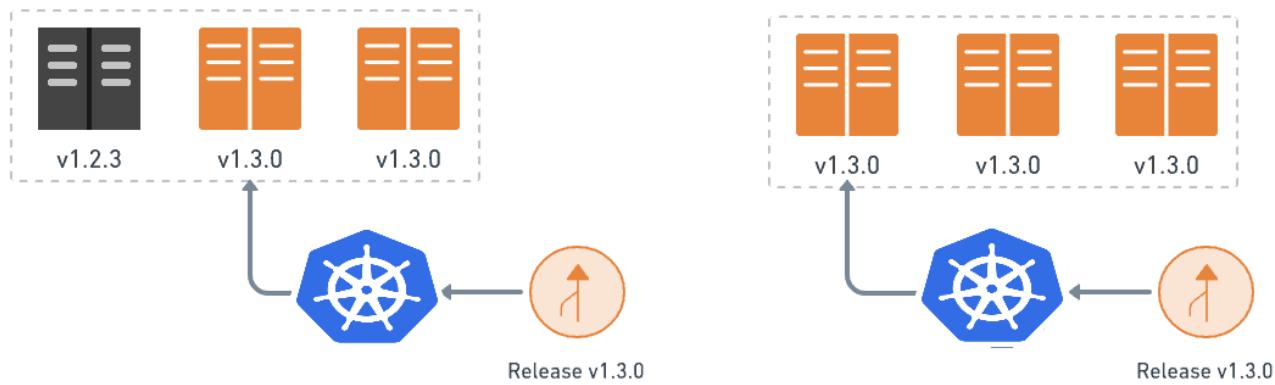
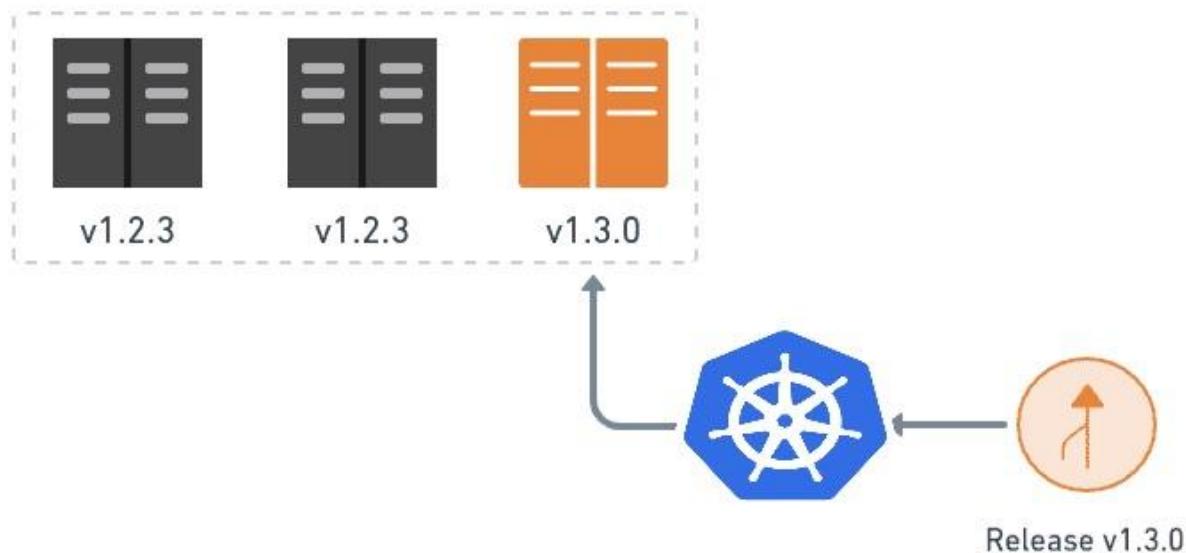


recreate-deployment-d7998b49b-hwztz	1/1	Terminating	0	20m
recreate-deployment-d7998b49b-m2tv8	1/1	Terminating	0	20m
recreate-deployment-d7998b49b-z7g5c	1/1	Terminating	0	20m

PS C:\Users\Devops> █

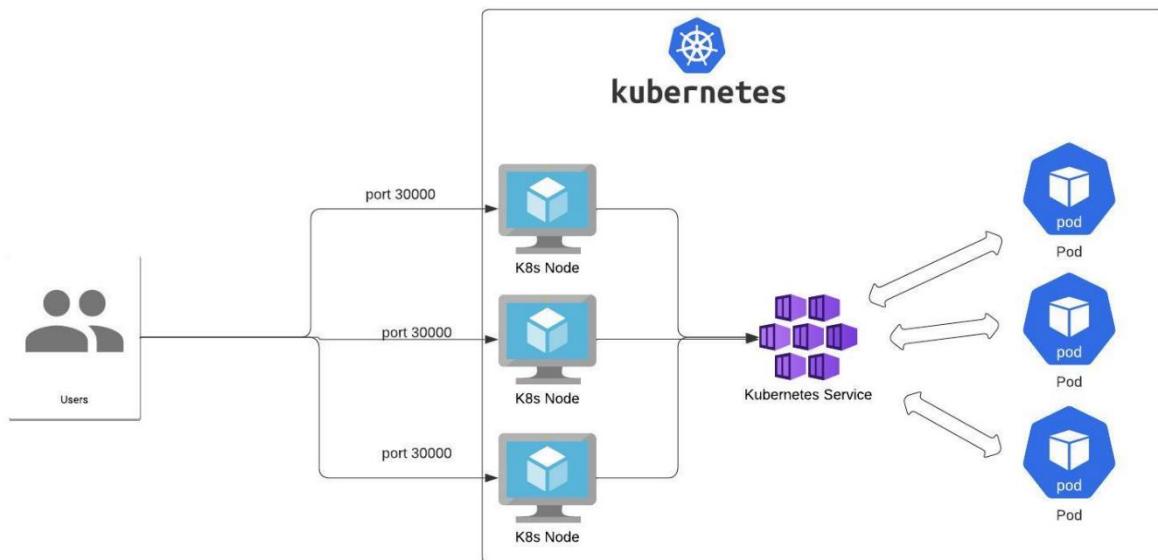
هنا بالظبط Partial update process: عباره عن Canary update Deployment strategy-3
كاني بعمل Test لـ APP بتاعي عن طريق الـ Users نفسهم من غير الالتزام اني اعمل Replace لكل الـ Pods

يعني بيتم انشاء بعض الـ Pods الجديدة وتخللي اغلب الـ pods القديمة وواحده واحده بغير فـ الـ Pods ما بين القديمة والجديدة.



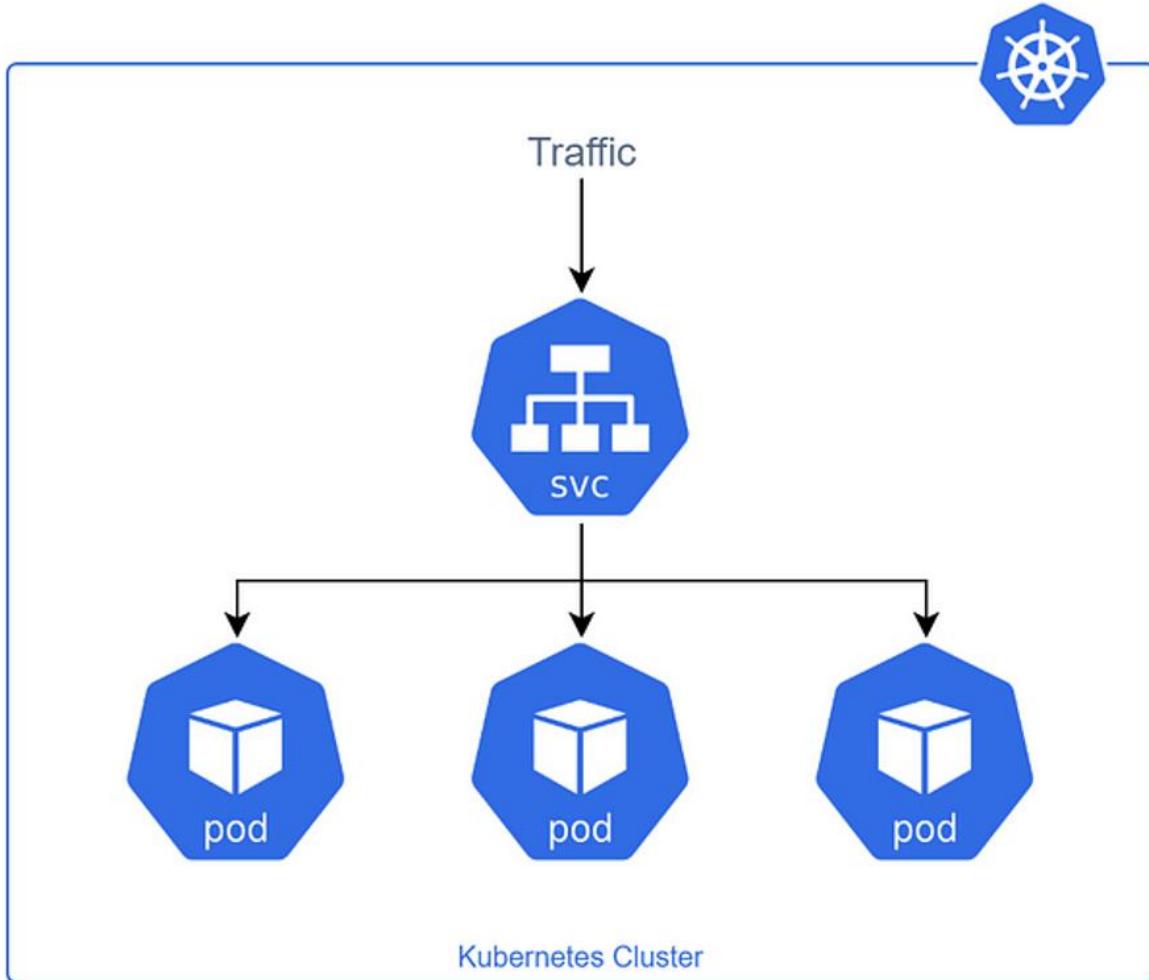
: K8s services

- من خلالها بقدر اعمل expose لـ Application ال داخل ال pod
- من خلالها بقدر اعمل Access لـ Application ال داخل ال pod من خارج ال Cluster
- لو عندي اكتر من deployment ال connection بينهم ه يكون عن طريق ال pods
- ال Services هي ال بتوزع الترافيك علي ال pods

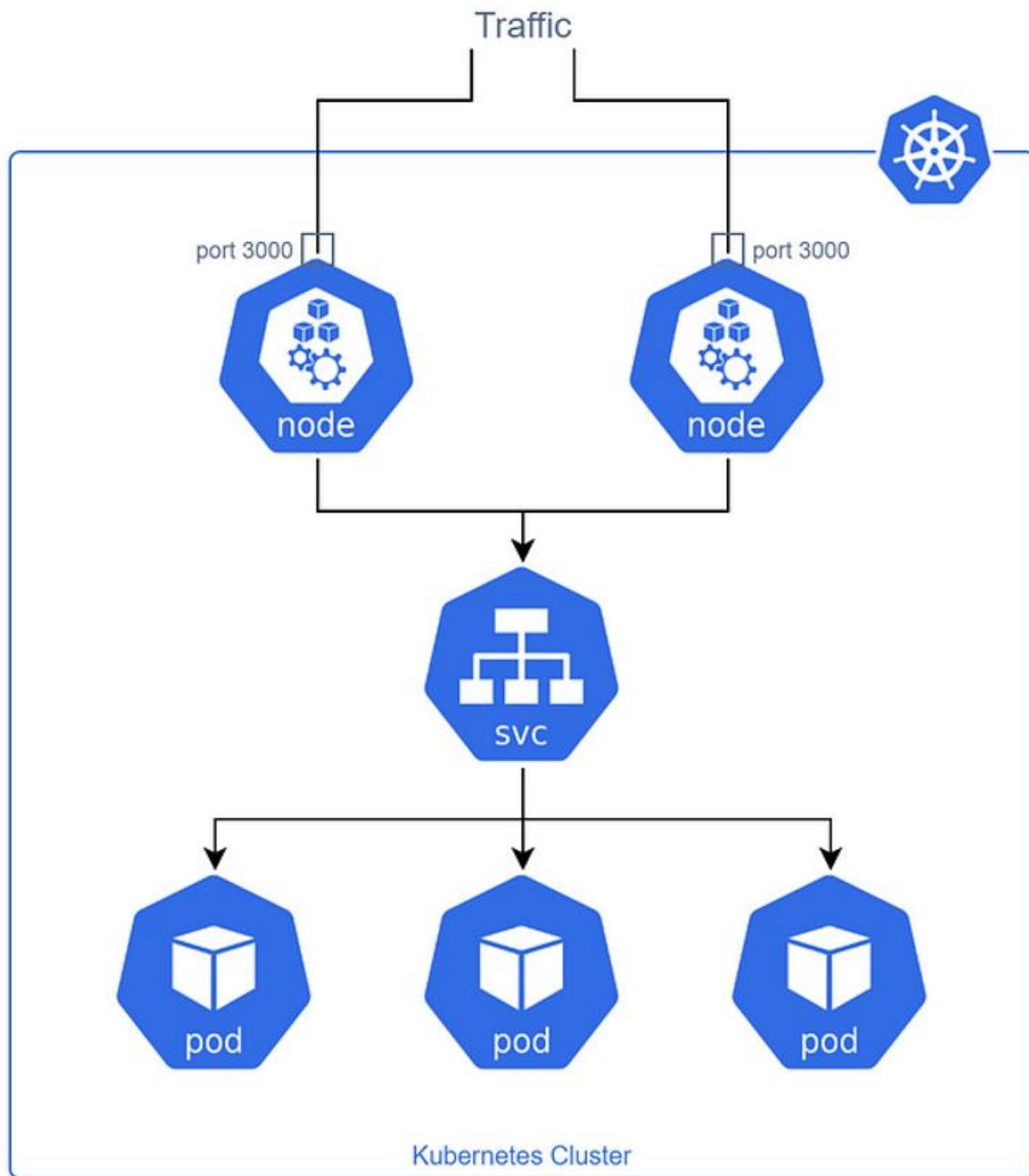


أنواع ال Services

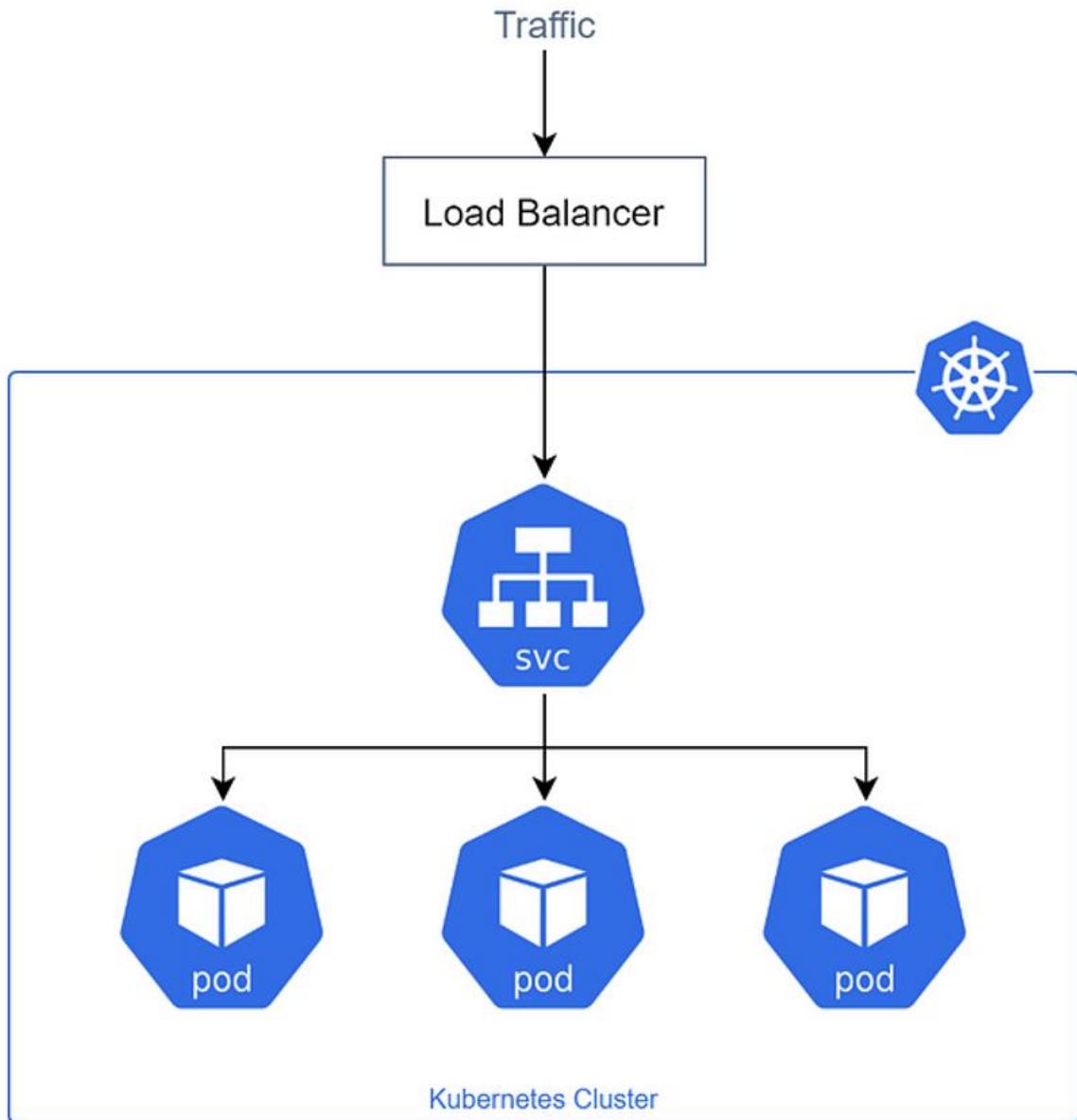
و دا معناه ان ال Services بتكون accessible من خلال ال Default ClusterIP-1
ومقدرش اعمل Request لـ pods من خارج ال cluster ال شغال عليه ،
(لو محدتش ال services فهتكون تلقائياً ال ClusterIP)



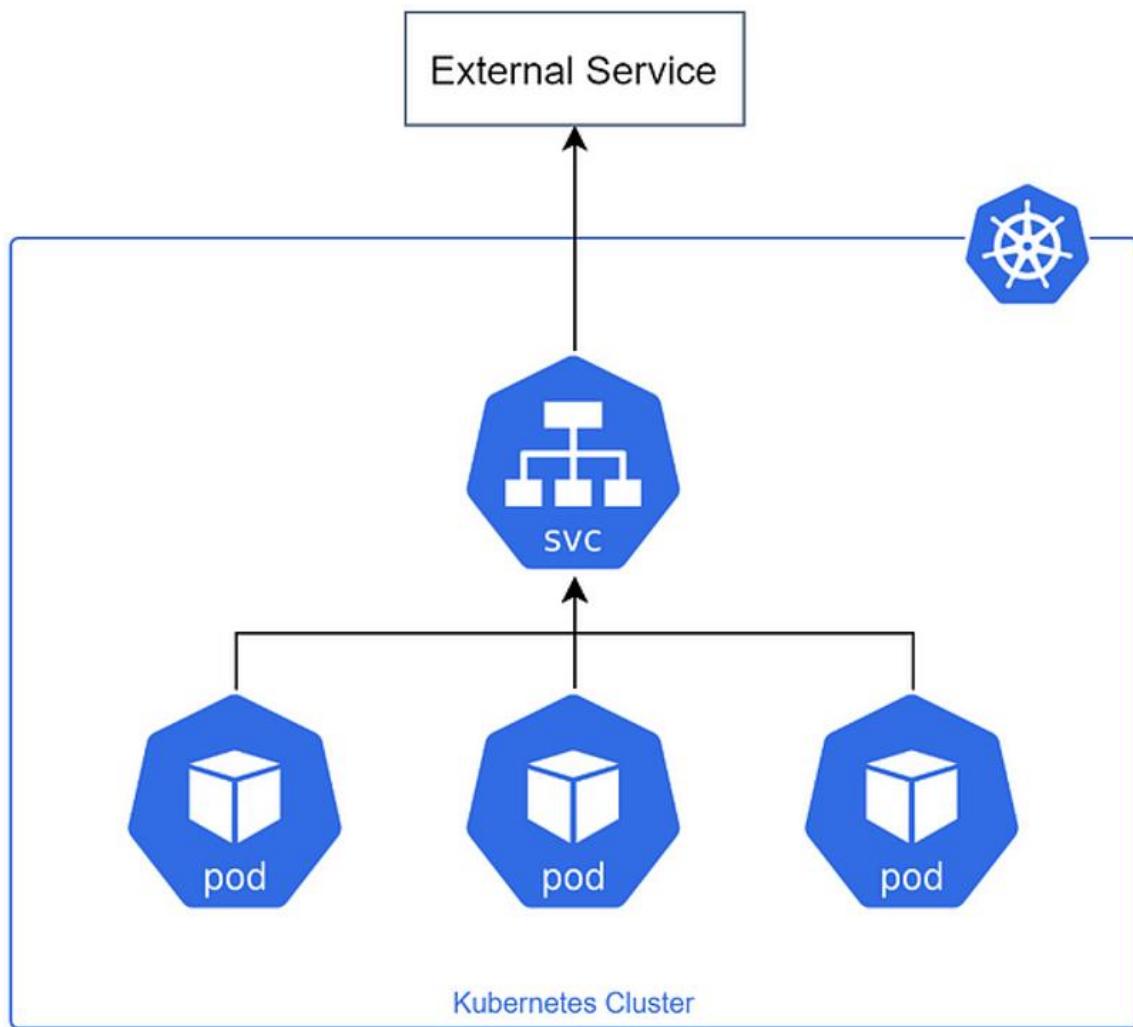
دی بتخلی ال static port تکون Services من accessible معین و معنی کدا ان ای
جای من خارج ال cluster هیشتغل و هیطلعله output Request



Cloud Services تکون accessible بشكل خارجي ع اي Load Balancer-3
LB Provider عنده



هنا مش بحدله أي ExternalName-4 Alias هو بيستخدم endpoint port ولا هو يوجهه (شبيه لـ DNS Services)



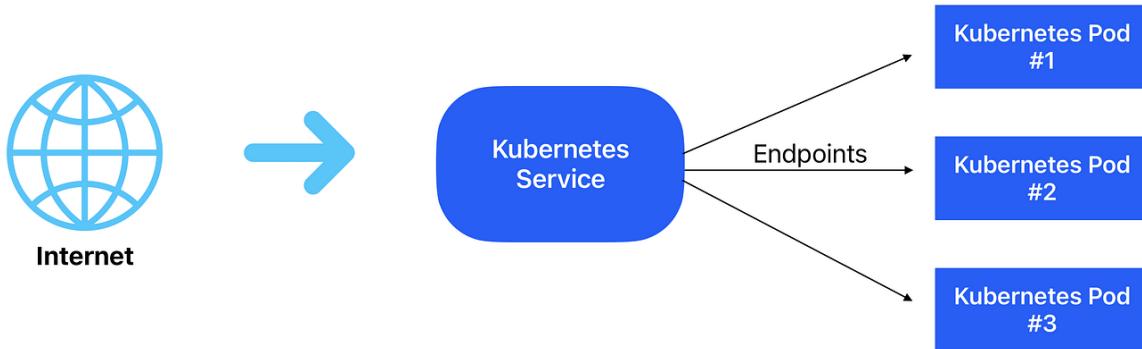
:ClusterIP

: K8s service yml file

```
apiVersion: v1          #1
kind: Service           #2
metadata:
  name: my-service    #3
spec:
  type: ClusterIP     #4
  selector:
    app: MyApp         #5
  ports:
    - port: 8080        #6
      targetPort: 80     #7
```

بوضح ال apiversion الخاص ب Service وهو ال v1	1
ال kind اني بوضح ال object ال عاوز اعمله وهو ال Service	2
ال metadata الخاصة بال Service	3
ال name الخاص بال Service	4
ال spec الخاص بال Service	5
نوع ال Service ال ه تكون عندي وهنا اختارت ال ClusterIP	6
ال selector خاص بال Service وبوضح فيه ال label	7
بوضح اسم ال label الخاص بال pod	8
ال ports بحدد ال service الخاصة بال pod وال pod	9
ال port دا ال خاص بال Services ال هقدر اعمل منه Access لـ pod	10
ال targetPort دا خاص بال container ال داخل ال pod وال ip دا بيكون برضو مكتوب في ال depoy file بيكون تحت ال spec في ال template	11

في حاجه اسمها endpoint ودي عباره عن ال ip الخاص بال pod + ال port الخاص بال pod
وعن طريقها ال services بتتواصل مع ال pods ويتوزع الترافيك



Endpoints: 10.244.0.38:80, 10.244.0.40:80, 10.244.0.41:80

```
mostafa@k8s:~$ kubectl expose deploy nginx --port=8080 --target-port=80
service/nginx exposed
mostafa@k8s:~$
```

عمل target port ل deploy (service) expose بيعمل nginx port 80 يكون target 80

```
mostafa@k8s:~$ kubectl get service
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP      2d21h
my-service   ClusterIP   10.99.12.0      <none>        80/TCP       40h
nginx       ClusterIP   10.101.165.253  <none>        8080/TCP     49s
mostafa@k8s:~$
```

عمل list لـ services عندى

```
mostafa@k8s:~$ kubectl describe svc nginx
Name:           nginx
Namespace:      default
Labels:          app=nginx
Annotations:    <none>
Selector:        app=nginx
Type:            ClusterIP
IP Family Policy: SingleStack
IP Families:    IPv4
IP:              10.101.165.253
IPs:             10.101.165.253
Port:            <unset>  8080/TCP
TargetPort:      80/TCP
Endpoints:       10.244.0.38:80,10.244.0.40:80,10.244.0.41:80
Session Affinity: None
Events:          <none>
mostafa@k8s:~$
```

عرض معلومات اكتر عن ال service ال اسمها nginx

```
mostafa@k8s:~$ kubectl get ep
NAME      ENDPOINTS   AGE
kubernetes  192.168.49.2:8443  2d21h
my-service <none>        40h
nginx     10.244.0.38:80,10.244.0.40:80,10.244.0.41:80  7m51s
mostafa@k8s:~$
```

عمل list بال endpoint علني

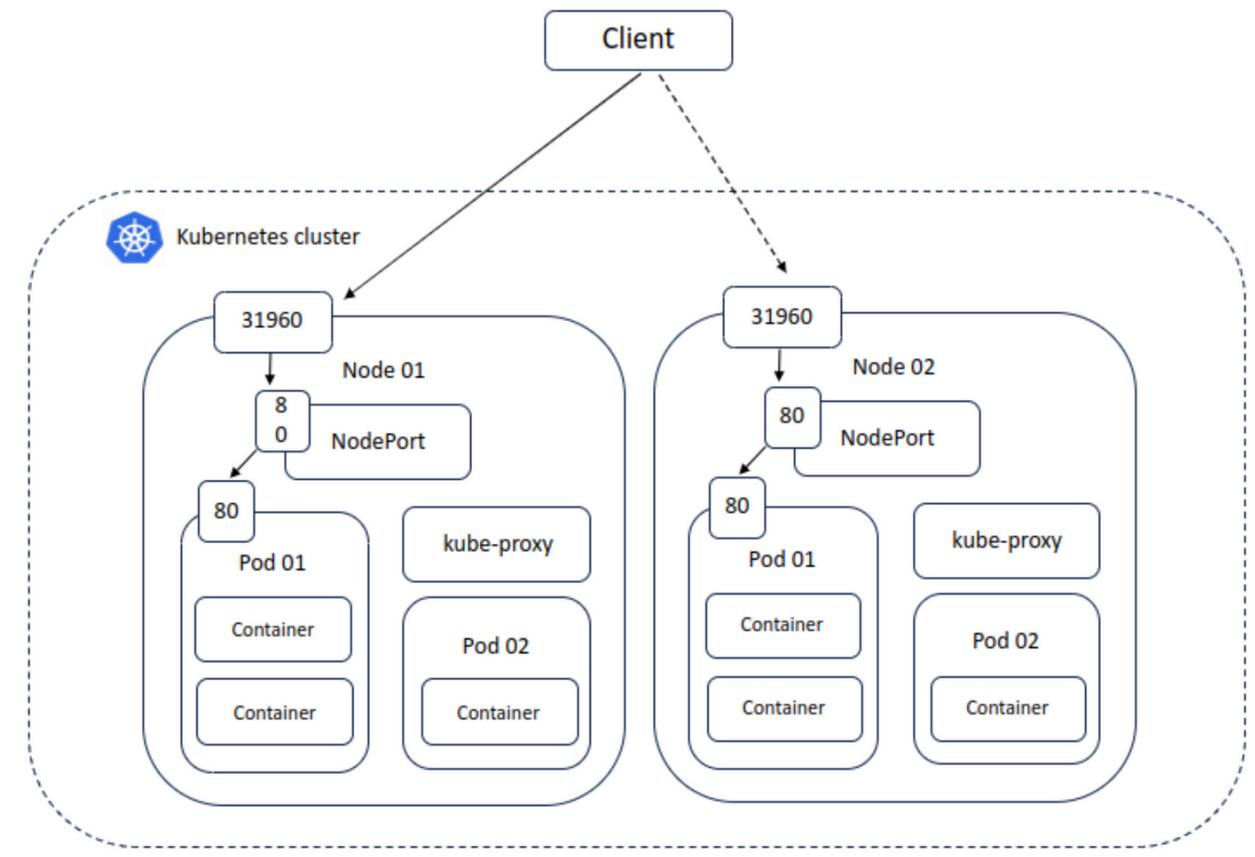
```
mostafa@k8s:~$ kubectl delete services nginx
service "nginx" deleted
mostafa@k8s:~$
```

عمل delete لـ service ال اسمها nginx

cluster: بتخليني اقدر access ال pod من خارج ال NodePort

-رنج ال ip بيكون بين 3000:32767 لازم يكون في الرنج دا

-فلو عندي ال ip بتاعها 192.168.1.50 وعملت service من نوع nodeport على
فعشان اقدر اعمل access ع ال pod هكتب ال ip بتاع ال node وال port بتاع
ال node برضو (192.168.1.50:3000)



: Nodeport yml file

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  type: NodePort
  selector:
    app: nginx
    env: nginx
  ports:
    - nodePort: 30000
      protocol: TCP
      port: 80
      targetPort: 80
```

ال nodePort : دا ال port الخاص ال Node

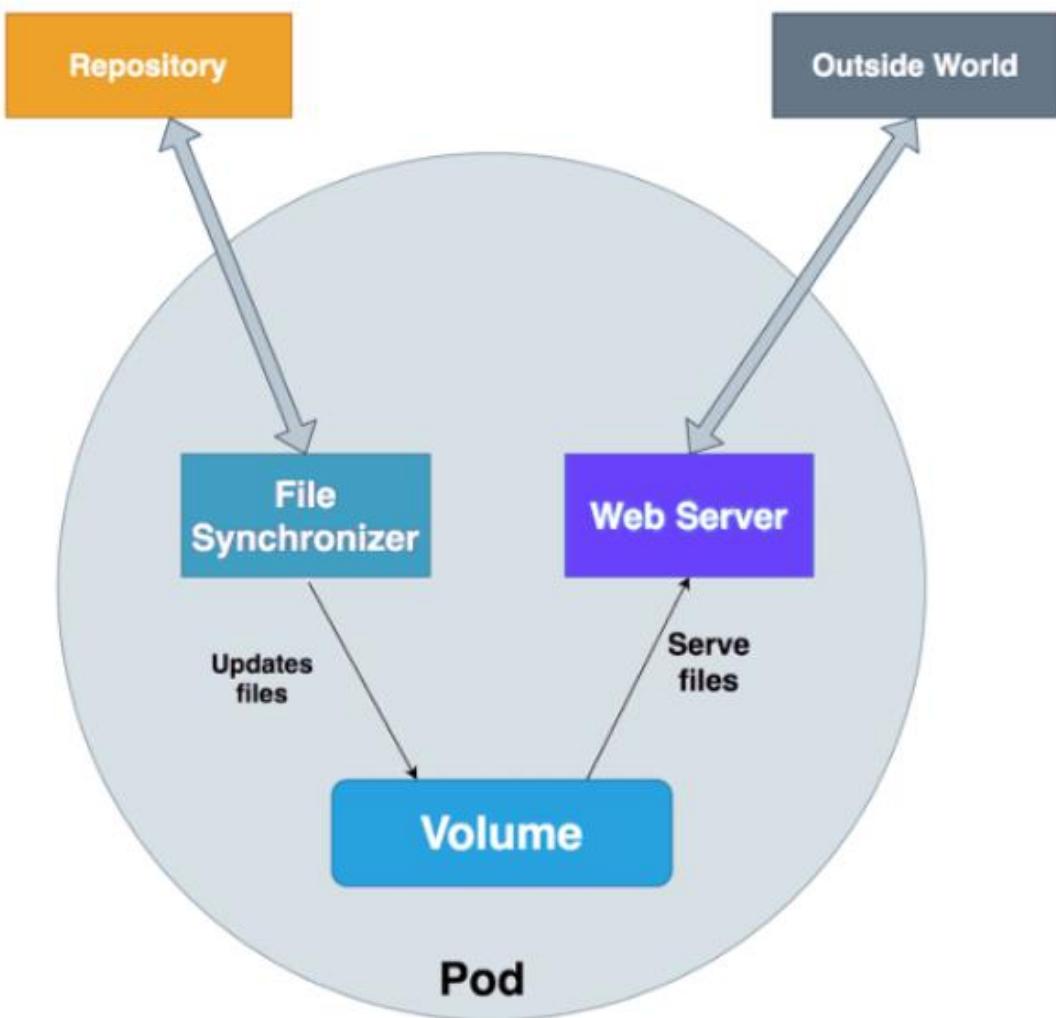
ال port : دا ال port الخاص ال service

ال targetPort دا ال port الخاص بال pod

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	3d4h
my-service	ClusterIP	10.99.12.0	<none>	80/TCP	47h
nginx	NodePort	10.107.221.35	<none>	80:30000/TCP	4m21s

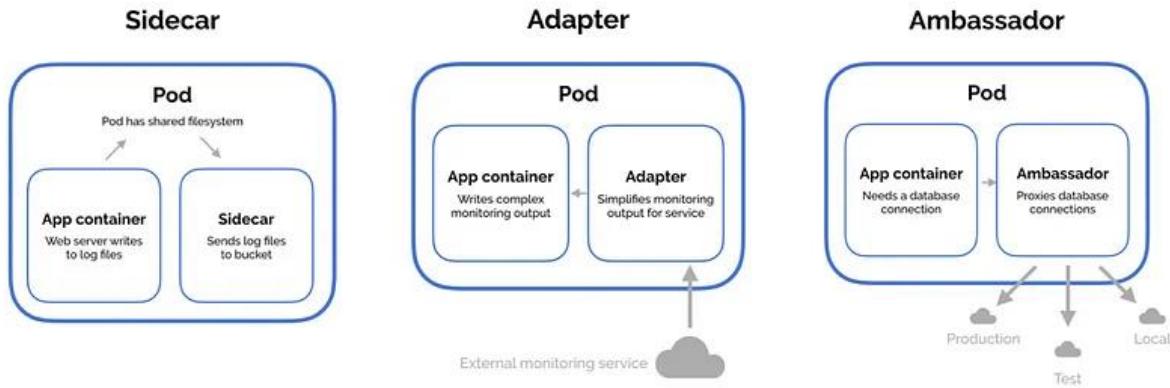
عمل list services و هلاقی ال config create ع ال اعملتها في ال yml file

ان يكون داخل ال Pod الواحد اكتر من share image : Multi-Container Pods in K8s Recourses

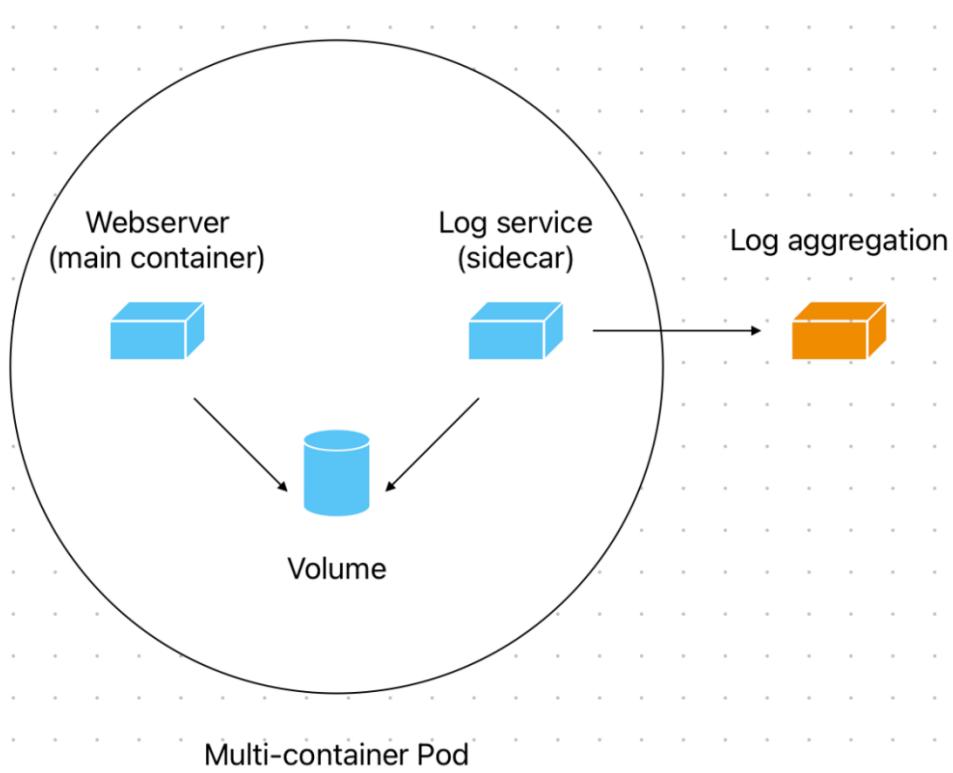


Multi-Container Pod

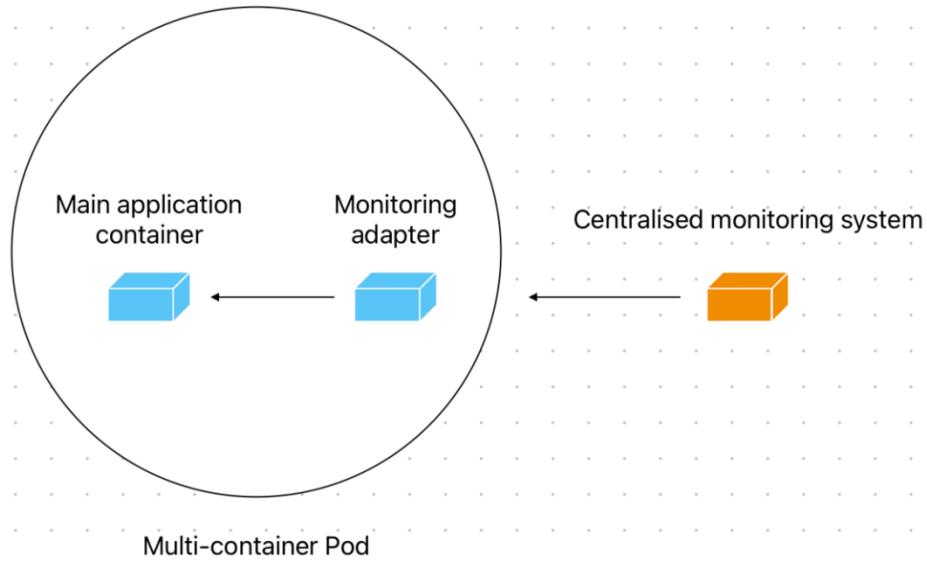
: Design-Patterns of Multi-Container Pods in K8s



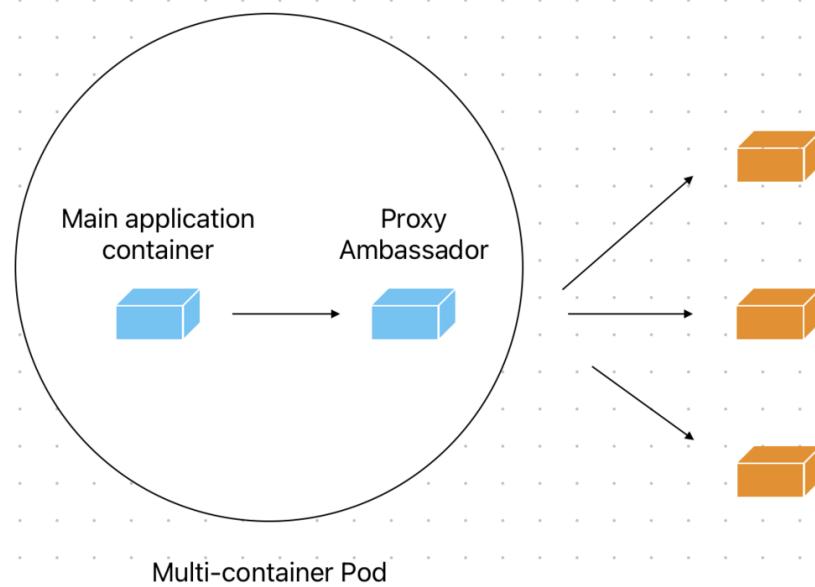
Sidecar-1 بيكون موجود فيه ال Main APP وهو موجود ال helper Process وال Helper Process دا بيكون له مسؤولية مهمة بس مش اساسية يعني اقدر اشغل ال APP من غيرها لكن هي مهمه عشان بتعمل Functionality تانية . يعني هو بتحسن فيه ال APP ويتزود ال Functionality بتاعتة .



transform او بيعمل external world لـ main container Connect : Adapter-2 من الـ main container output الي خارج الـ Pod کله .

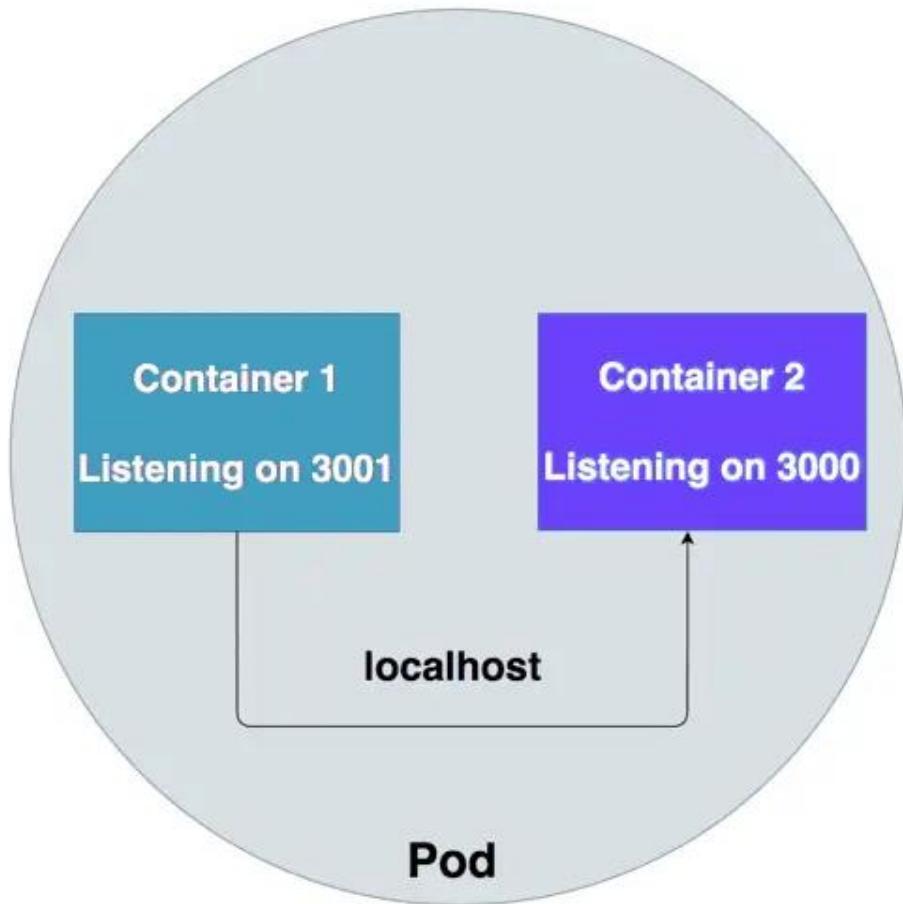


Ambassador -3 : برضو connect ال main container بالعالم الخارجي بس هنا ال connect requests يعني هو اشبه بال proxy بيسمح للContainers الثانية انهات ع ال port ال localhost او ال main container .

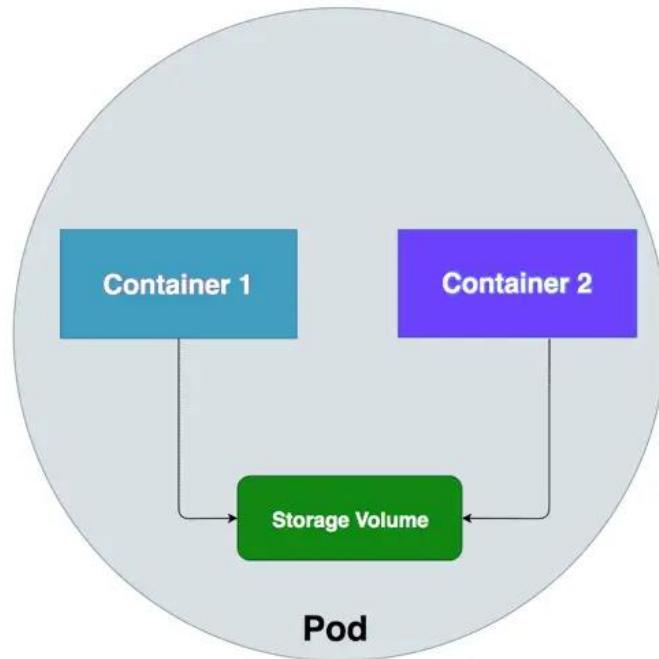


يعني از اي ال Commnunication inside A Multi Container Pod بيحصل بين ال Container دا خل ال Pod و دا ليه اكثـر من طريقة :

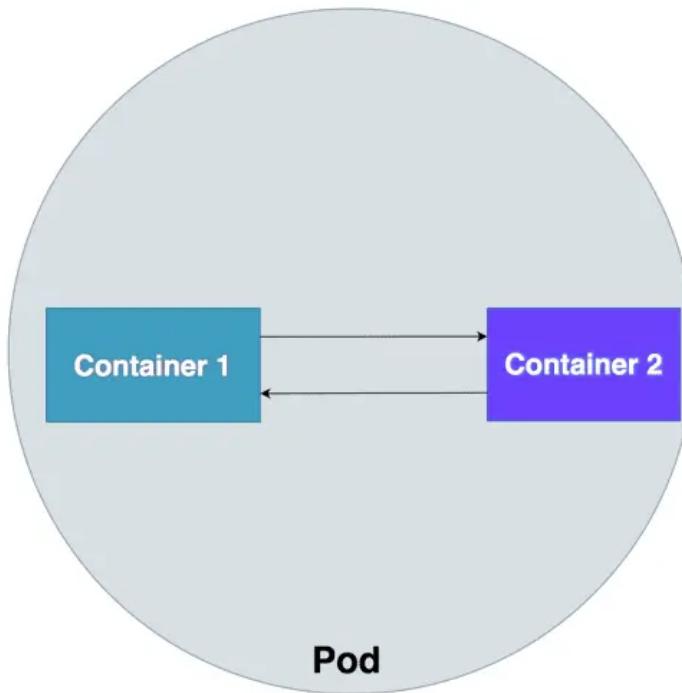
كل ال container ال موجودة في ال Pod بتتشارك ف نفس ال Shared Network NameSpace-1 لذلك كل ال container تقدر تتواصل مع بعض .



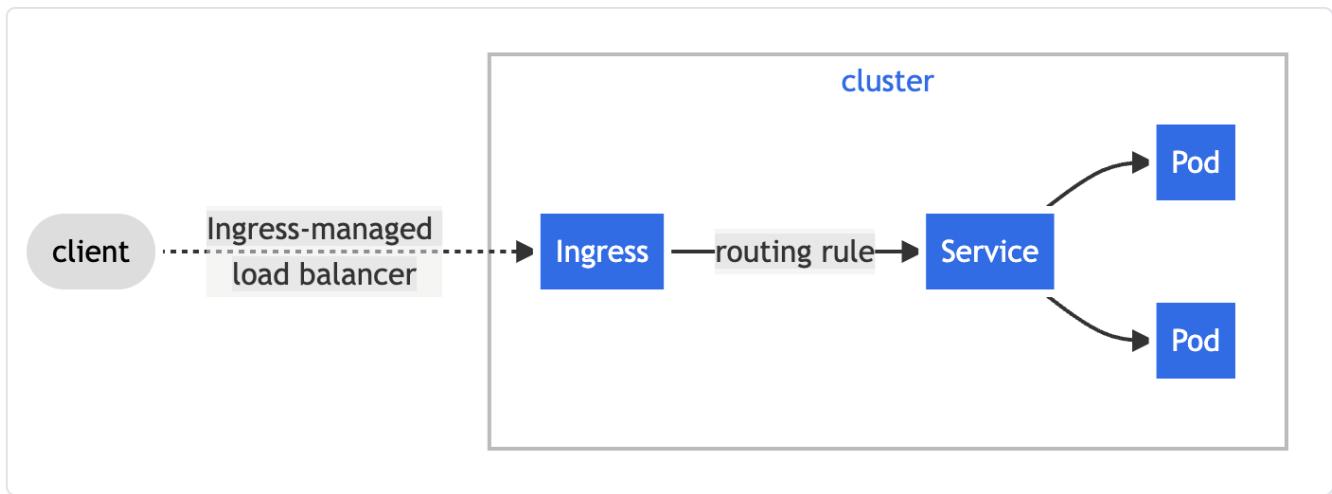
كل ال موجودة في ال Pod بتشارك نفس ال container Shared Storage Volumes-2 . (Storage) Volumes



ب يكون عندنا Shared Process NameSpace-3 موجود داخل ال pod اسمه Process NS Flag وبعدها تقدر ال container داخل ال pod تواصل مع بعض .



ال ingress object هي بيعمل allow access to k8s service من خارج ال cluster وتقدر تعمل rule لـ access دى عن طريقه connection define ازاي توصل لـ APP من خارج ال cluster الى داخل ال cluster وف كذا طريقه اعمل بيهما expose لـ service زري ClusterIP-NodePort-LoadBalancer



```

mostafa@k8s:~$ kubectl create ingress nginx-ingress --rule="/=nginx-ingress:80"
ingress.networking.k8s.io/nginx-ingress created
mostafa@k8s:~$ 
  
```

عمل nginx-ingress ل اسمه create

Ingress Controller	Ingress API
مسؤال عن عملية ال services بتاعه الدخول هو اشبه بال Load Balancer او يقوم بوظيفته لكن لا يعتبر LB	هو المسؤال عن عملية ال Exposing من خارج ال cluster

كل ingress path type path يكون له path type path متطابق وفي 3 أنواع

ال ingress class بيتعمله تتطابق بناءا ع ال Implementaion Specific-1

لازم ال path يكون matches case sensitivity exact-2

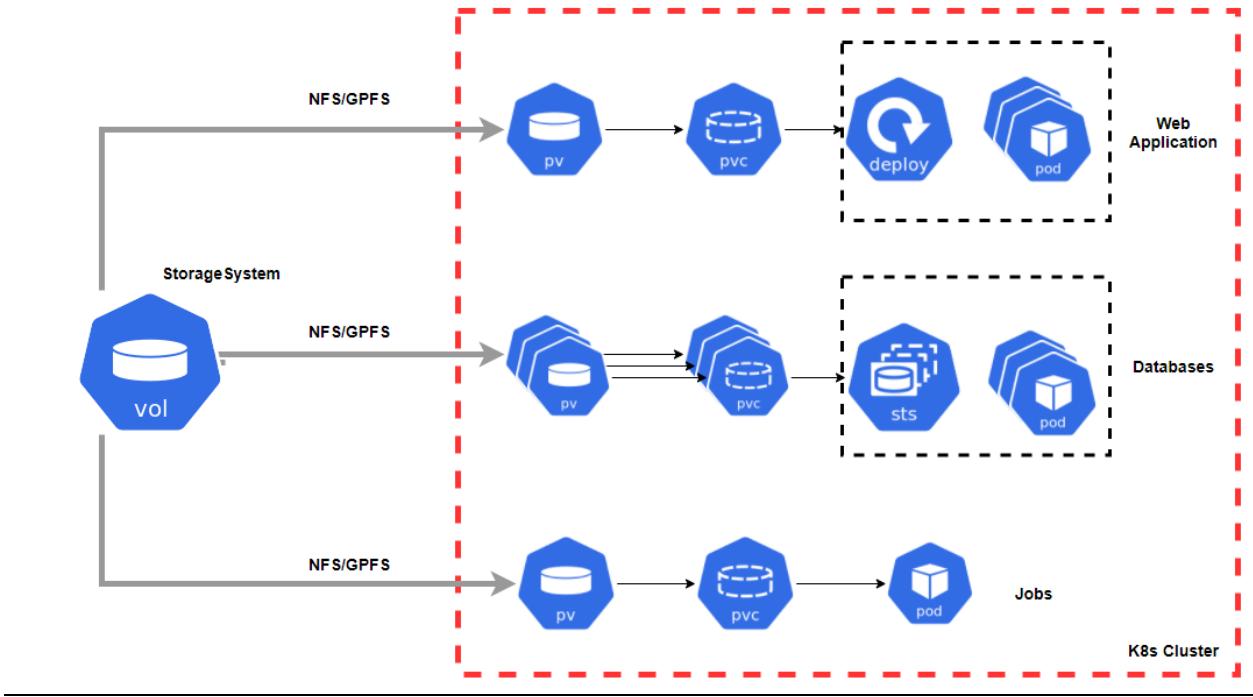
Exact	/foo	/bar	No
Exact	/foo	/foo/	No
Exact	/foo/	/foo	No

عکس ال path matches تماما يعني ال path مش لازم يكون case sensitivity exact-3 prefix-3

Prefix	/foo	/foo , /foo/	Yes
Prefix	/foo/	/foo , /foo/	Yes
Prefix	/aaa/bb	/aaa/bbb	No
Prefix	/aaa/bbb	/aaa/bbb	Yes

: K8s Storage

ال Storage هنا مبنيه على ال Volumes بمعنى ان ال Volumes تكون مجردة مش مرتبطة ب pod معين مثل .



```
mostafa@k8s:~$ kubectl explain pod.spec.volumes
KIND:     Pod
VERSION:  v1

FIELD: volumes <[ ]Volume>

DESCRIPTION:
List of volumes that can be mounted by containers belonging to the pod. More
info: https://kubernetes.io/docs/concepts/storage/volumes
Volume represents a named volume in a pod that may be accessed by any
container in the pod.

FIELDS:
awsElasticBlockStore  <AWSElasticBlockStoreVolumeSource>
awsElasticBlockStore represents an AWS Disk resource that is attached to a
kubelet's host machine and then exposed to the pod. More info:
https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore

azureDisk    <AzureDiskVolumeSource>
azureDisk represents an Azure Data Disk mount on the host and bind mount to
the pod.

azureFile    <AzureFileVolumeSource>
azureFile represents an Azure File Service mount on the host and bind mount
to the pod.

cephfs      <CephFSTVolumeSource>
cephFS represents a Ceph FS mount on the host that shares a pod's lifetime

cinder       <CinderVolumeSource>
cinder represents a cinder volume attached and mounted on kubelets host
machine. More info: https://examples.k8s.io/mysql-cinder-pd/README.md

configMap   <ConfigMapVolumeSource>
configMap represents a configMap that should populate this volume

csi         <CSIVolumeSource>
csi (Container Storage Interface) represents ephemeral storage that is
handled by certain external CSI drivers (Beta feature).
```

عمل لـ Volumes Explain

اقدر اعمل Volumes Access لل بطرقين

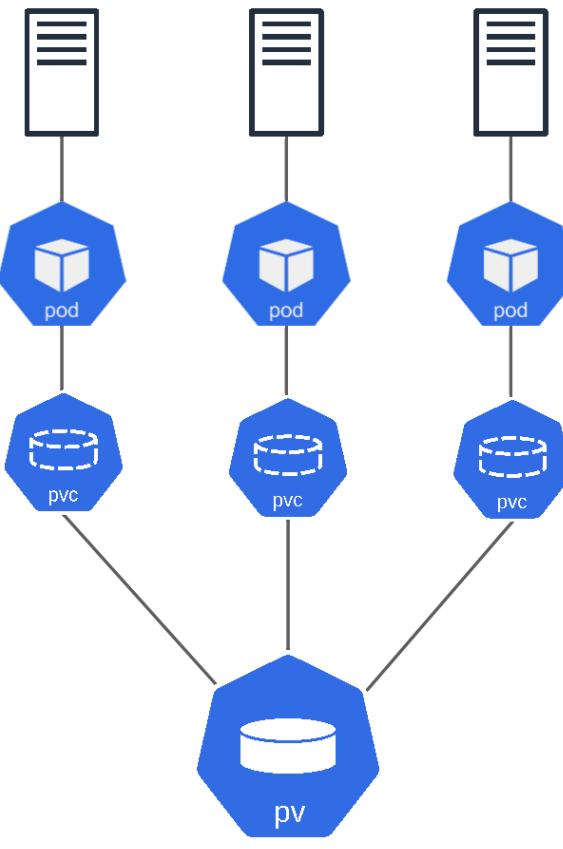
1- Ephemeral (non-Persistent) : هي temporary وهي ال default ويتكون متاحه طول م ال Short-Term Storage

2- Persistent Volumes : دyi عكس ال Non Support وب أنواع كتير منها

File Block Storage-Cloud Services-object Storage

ودي بتستخدم مع ال Long-Term Storage

k8s1 k8s2 k8s3



PersistentVolume

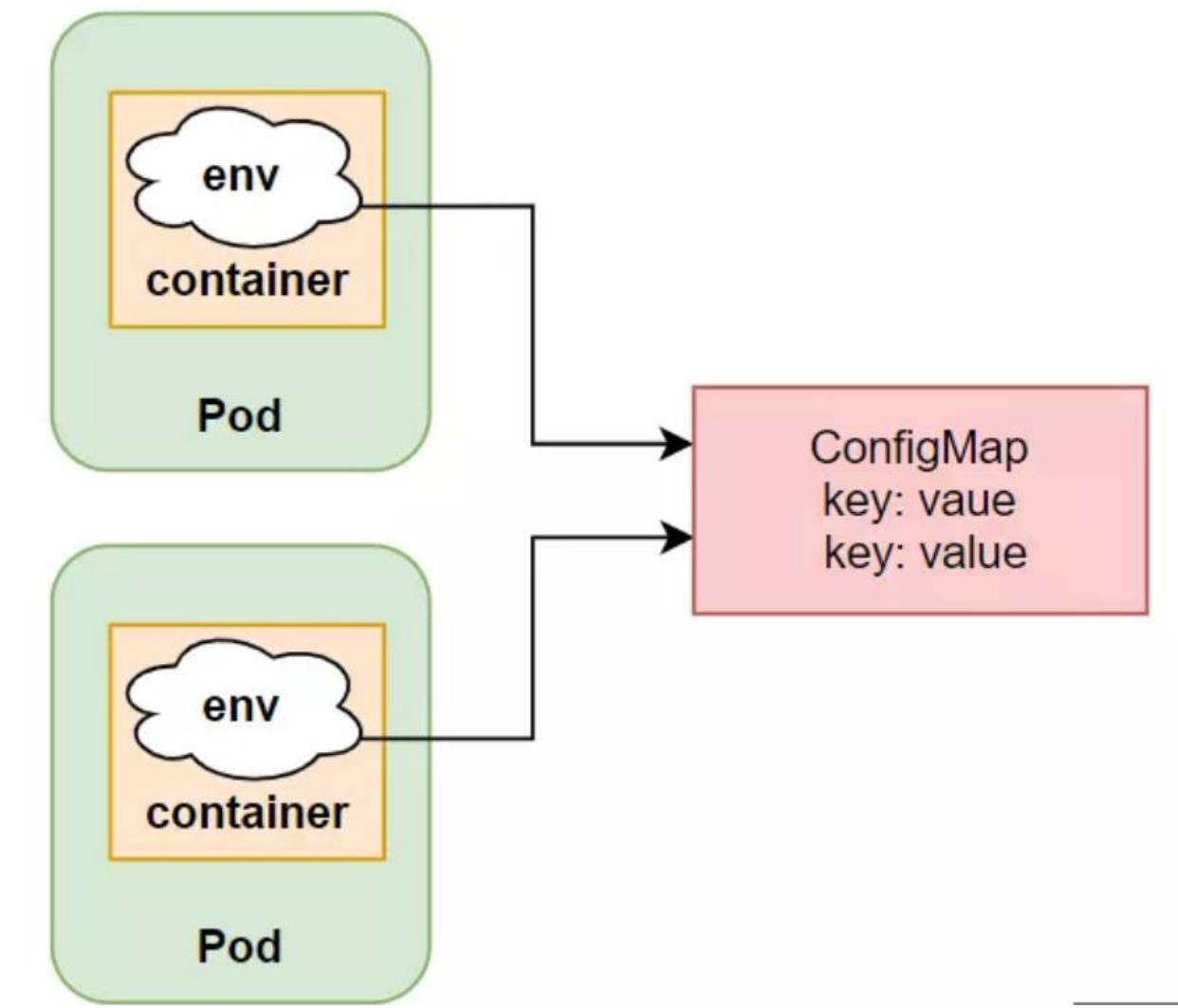
Persistent Volumes Claim(PVC)	Persistent Volumes(PV)
هو عبارة عن Storage Request لـ Pod من Users وبيتم تخصيصها لـ Admin	هي قطعة من ال Storage داخل ال Cluster هو ال بيوفرها وهي مستقلة عن ال Pod بيسخدمها يعني انك لو عملت Reference م بين PV دا وال Pod وبعد حين حذف ال Pod ال PV مش هيتحذف والبيانات هتفصل موجودة

Access Volumes Modes

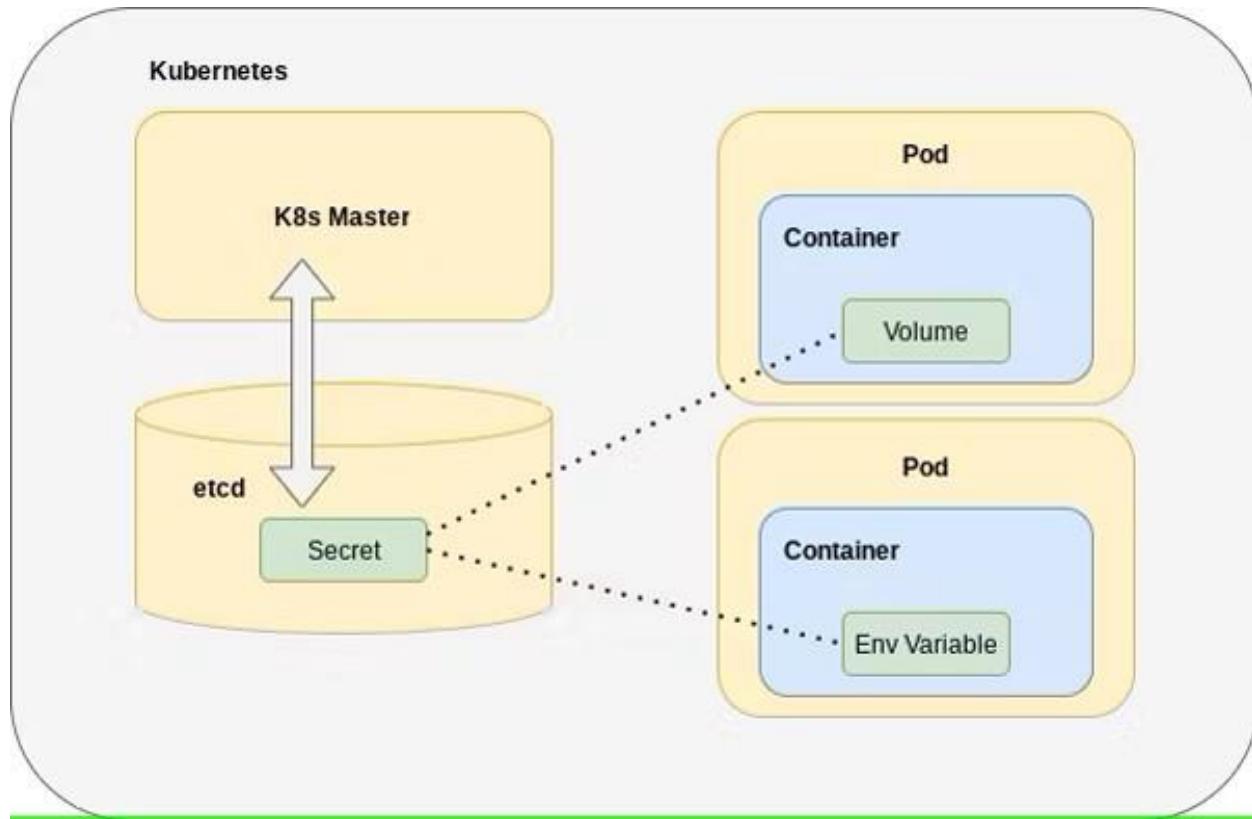
Read Write once Pod	Read only Many	Read Write Many	Read Write once
ال Volume يكون Single Pod لـ Access واحد فقط	ال Volum يكون Read Only فقط و اكثر من Node يقدر Read	اكثر من Node تقدر Write و Read لـ Volume	بسمح ب Access واحد هو ال Pod دا وكل ال Volume تقدر Read و Write من ال Volume دا لكن مش خارج ال Node المحدد



هي Object : Config Maps تانية موجودة يقدر يخليك تخزن Config Objects بيتم استخدامها بواسطة وبنكون عباره عن Key و Value



Secrets : هو عبة عن بيانات حساسة بيتخزنها زي ال Username-Password الهدف الرئيسي من استخدامها انك تقل خطورة انك تعمل exposing للبيانات دي لما تيجي تعمل لل Deploying App



```
mostafa@k8s:~$ kubectl create secret generic test-secret --from-literal='username=admin' --from-literal='password=ACD@123'
secret/test-secret created
mostafa@k8s:~$
```

عمل secret اسمه test-secret بال username admin و password اسمه ACD@123

ال Size بتاعها بيكون 1MB

ال أنواع

Opaque-1 : وهي ال Default في ال Secret

Service Account Token-2 : دا بيعتظر بال Token ال بتعمل Identify لل Service Account وبيتم انشاءه لما بيحصل Create لل Pods وتقدر تعمل للطريقه دي Disable

Basic Authentication Credentials-3 : هي عبارة عن Basic Authentication بس لازم تكون username and Password key

SSH Private Key-4 : بيحتفظ بالبيانات الضرورية عشان تنشأ SSH Connection ولازم تحتوي ع SSH Private Ke

النوع دا بتحفظ بال Associated Key وال المستخدم عشان ال TLS-5 دي ولازم
تناك اتن معاك ال TLS Cert وال TLS Key

Docker Registry : يتحفظ بال Credentials الخاصة بال Docker Config-6

Container Image

هي Token : Bootstrap Token-7
Node Bootstrap Token بتسخدم من خلال مثلاً كانك بتعمل Sign Config Maps

```
mostafa@k8s:~$ kubectl create secret generic -h
Create a secret based on a file, directory, or specified literal value.

A single secret may package one or more key/value pairs.

When creating a secret based on a file, the key will default to the basename of the file, and the value will default to the file content. If the basename is an invalid key or you wish to chose your own, you may specify an alternate key.

When creating a secret based on a directory, each file whose basename is a valid key in the directory will be packaged into the secret. Any directory entries except regular files are ignored (e.g. subdirectories, symlinks, devices, pipes, etc).

Examples:
# Create a new secret named my-secret with keys for each file in folder bar
kubectl create secret generic my-secret --from-file=path/to/bar

# Create a new secret named my-secret with specified keys instead of names on disk
kubectl create secret generic my-secret --from-file=ssh-privatekey=path/to/id_rsa
--from-file=ssh-publickey=path/to/id_rsa.pub

# Create a new secret named my-secret with key1=supersecret and key2=topsecret
kubectl create secret generic my-secret --from-literal=key1=supersecret --from-literal=key2=topsecret

# Create a new secret named my-secret using a combination of a file and a literal
kubectl create secret generic my-secret --from-file=ssh-privatekey=path/to/id_rsa --from-literal=passphrase=topsecret

# Create a new secret named my-secret from env files
kubectl create secret generic my-secret --from-env-file=path/to/foo.env --from-env-file=path/to/bar.env
```

عرض ال information بتاع Secret

By:Mostafa Mahmoud Bahgat

LinkedIn: <https://www.linkedin.com/in/mostafamahmoudbahgat>

