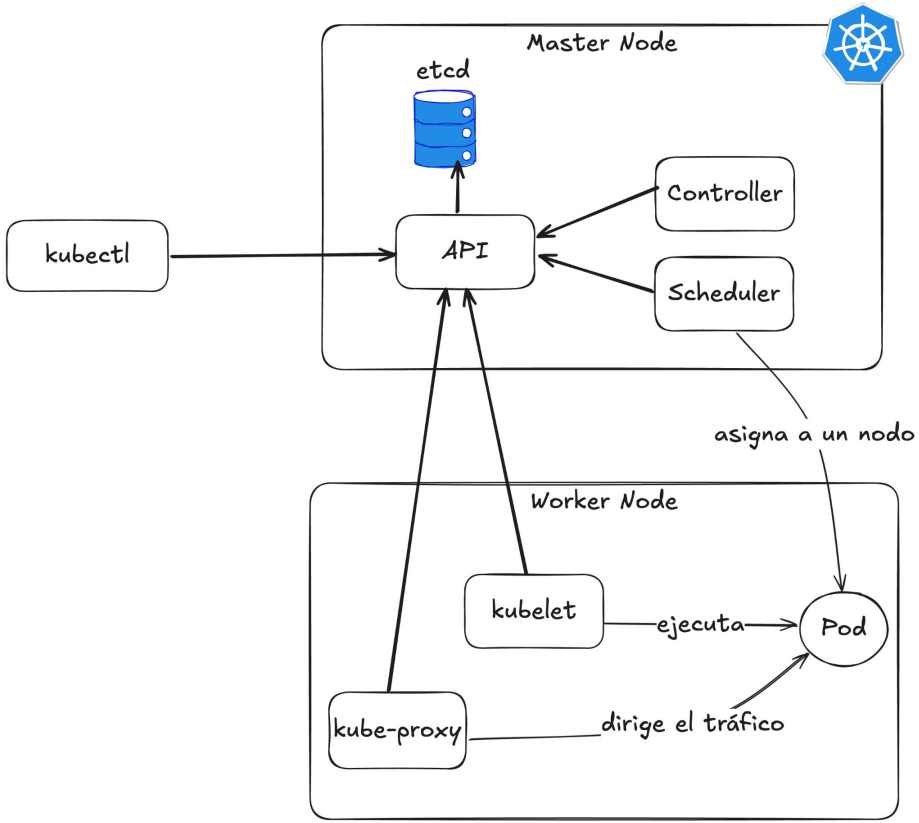
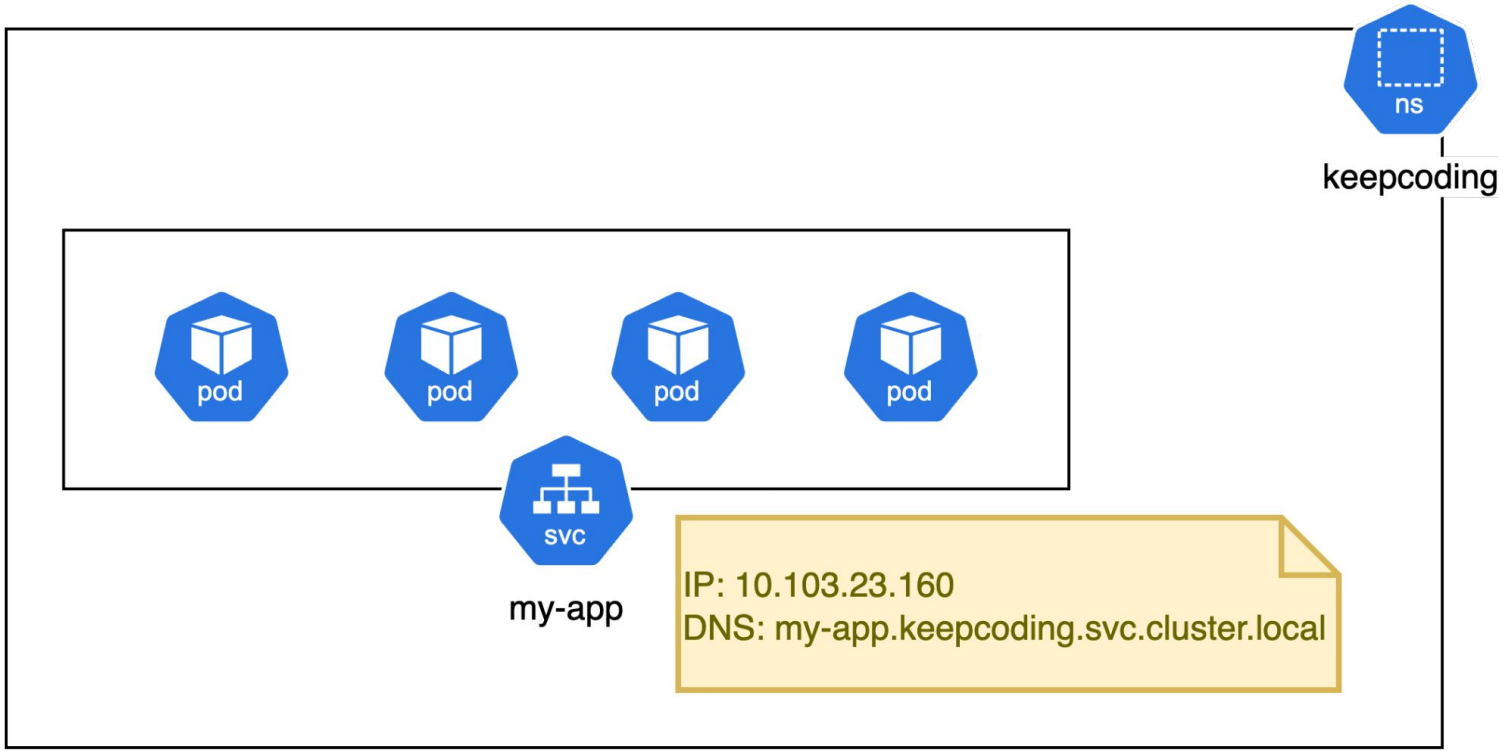
Repaso

KEEPCODING
Tech School

Arquitectura



Networking



Pods

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    app: myapp
spec:
  containers:
  - name: myapp-container
    image: busybox
    command: ['sh', '-c']
    args:
      - |
        echo 'Hello Kubernetes!'
        sleep 30
```

```
$ kubectl apply -f pod.yaml
$ kubectl get pods
$ kubectl describe pod myapp-pod
$ kubectl logs myapp-pod
```

Pods - Liveness & Readiness probes

```
apiversion: v1
kind: Pod
metadata:
  labels:
    test: liveness
  name: liveness-http
spec:
  containers:
  - name: liveness
    image: registry.k8s.io/e2e-test-images/agnhost:2.40
    args:
    - liveness
    livenessProbe:
      httpGet:
        path: /healthz
        port: 8080
        httpHeaders:
        - name: Custom-Header
          value: Awesome
      initialDelaySeconds: 3
      periodSeconds: 3
```

```
apiversion: v1
kind: Pod
metadata:
  labels:
    test: readiness
  name: readiness-http
spec:
  containers:
  - name: readiness
    image: registry.k8s.io/e2e-test-images/agnhost:2.40
    args:
    - readiness
    readinessProbe:
      httpGet:
        path: /healthz
        port: 8080
        httpHeaders:
        - name: Custom-Header
          value: Awesome
      initialDelaySeconds: 3
      periodSeconds: 3
```

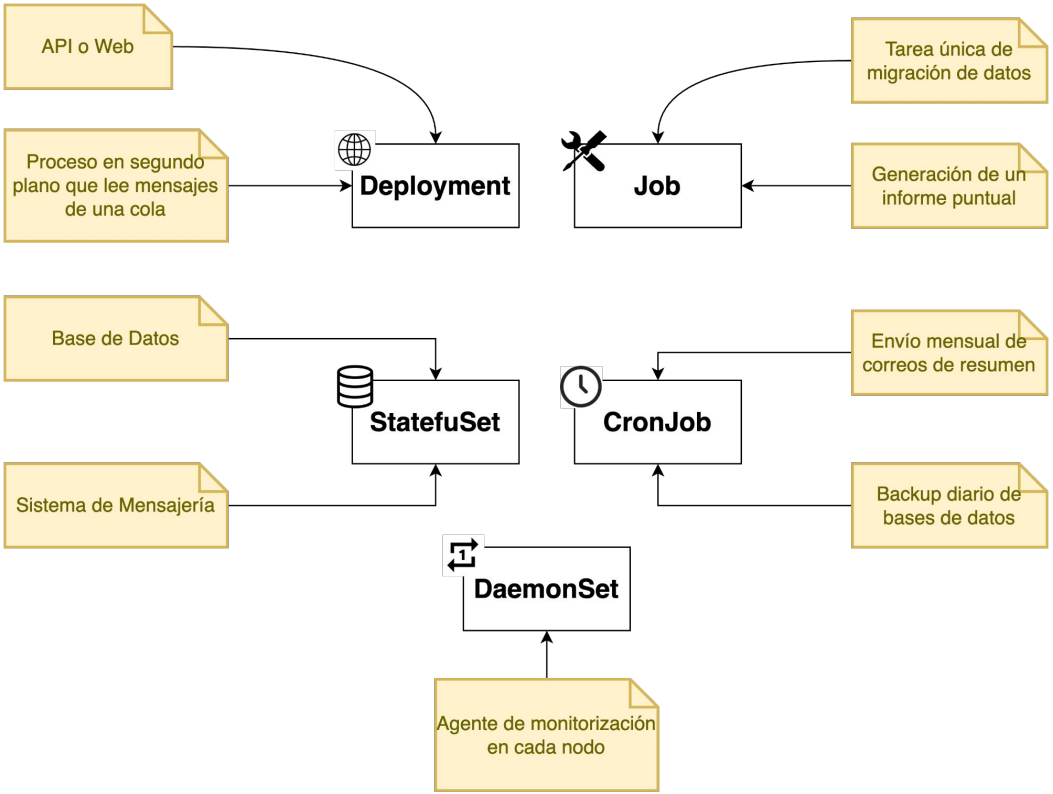
ConfigMaps & Secrets

```
apiVersion: v1
kind: Pod
metadata:
  name: env-configmap
spec:
  containers:
    - name: app
      command: ["/bin/sh", "-c", "printenv"]
      image: busybox:latest
      envFrom:
        - configMapRef:
            name: myconfigmap
        - secretRef:
            name: test-secret
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: myconfigmap
data:
  username: k8s-admin
  access_level: "1"
```

```
apiVersion: v1
kind: Secret
metadata:
  name: test-secret
data:
  username: bXktYXBw
  password: Mzk1MjgkdmRnN0pi
```

Workloads



Workloads

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  serviceName: mysql
  replicas: 3
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        image: mysql:5.7
        ports:
        - containerPort: 3306
        volumeMounts:
        - name: data
          mountPath: /var/lib/mysql
          subPath: mysql
volumeClaimTemplates:
  - metadata:
      name: data
    spec:
      accessModes: ["ReadWriteOnce"]
      resources:
        requests:
          storage: 10Gi
```

Resources - CPU & Memory

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
```

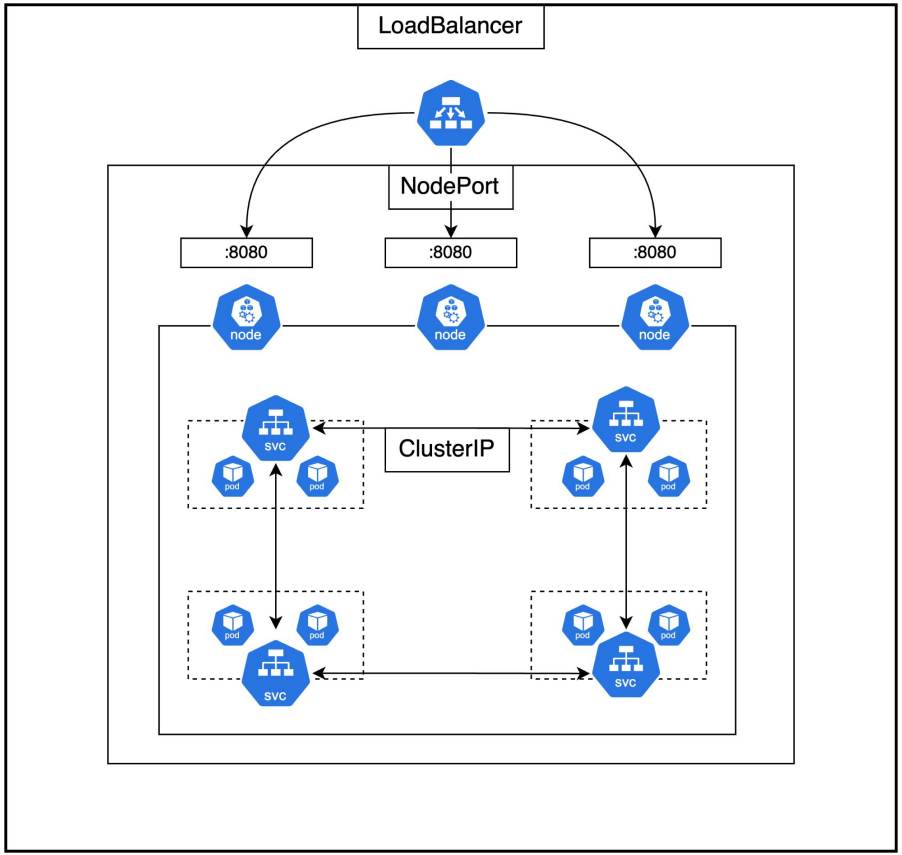
```
$ kubectl top pods
$ kubectl top nodes
```

Auto-escalado horizontal de Pods

```yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: hpa-example
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: myapp
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 70
```

https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/

Services



Services

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app.kubernetes.io/name: proxy
  ports:
  - name: name-of-service-port
    protocol: TCP
    port: 80
    targetPort: 8080
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    app.kubernetes.io/name: proxy
spec:
  containers:
  - name: nginx
    image: nginx:stable
    ports:
      - containerPort: 8080
```

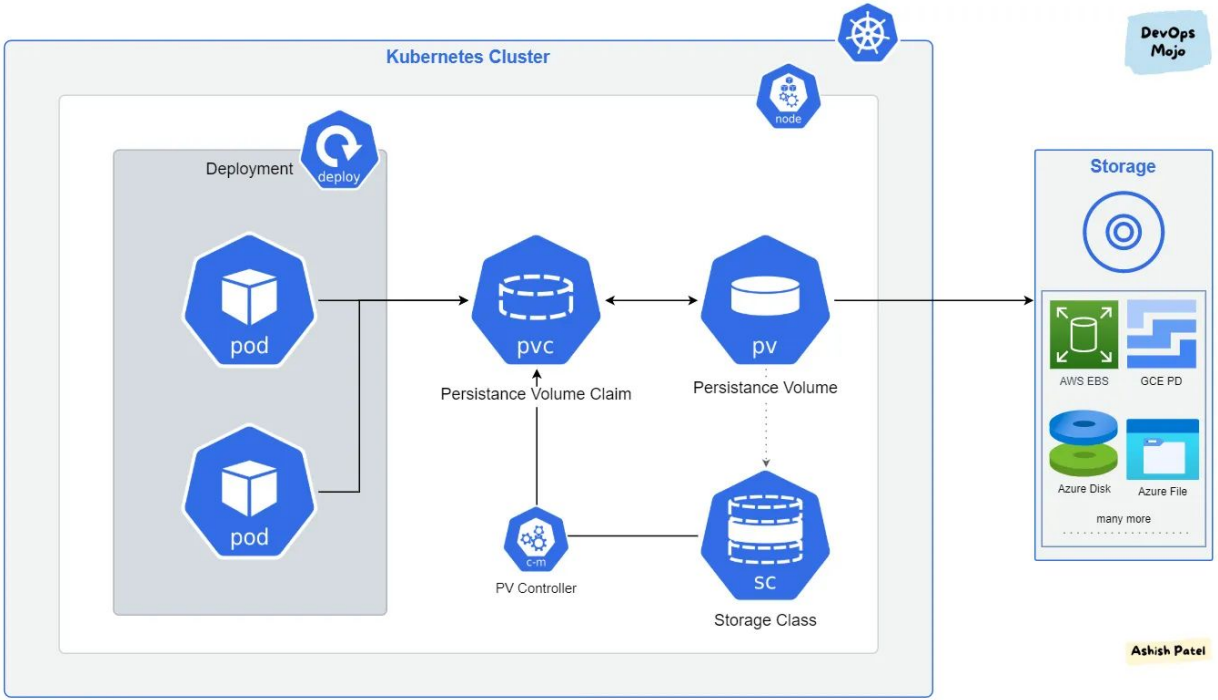
StorageClasses & PVC

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard
provisioner: k8s.io/minikube-hostpath
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mariadb-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

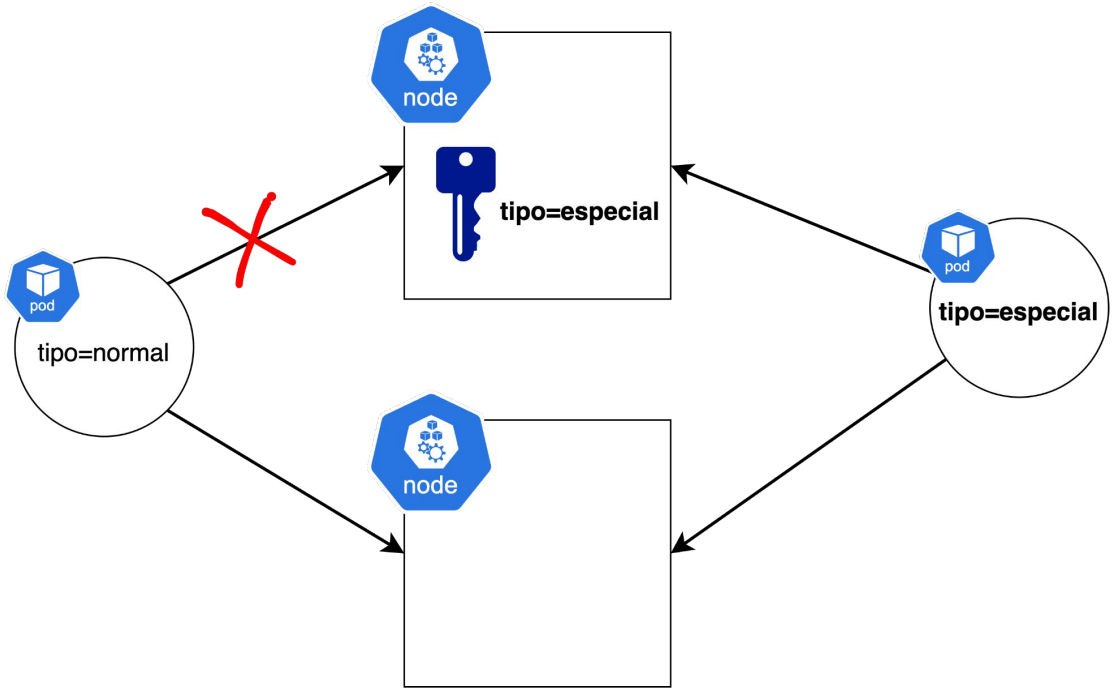
```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mariadb
spec:
...
      containers:
        - name: mariadb
          image: mariadb:10.5
          ports:
            - containerPort: 3306
          volumeMounts:
            - name: mariadb-storage
              mountPath: /var/lib/mysql
      volumes:
        - name: mariadb-storage
          persistentVolumeClaim:
            claimName: mariadb-pvc
```

StorageClasses & PVC



https://medium.com/devops-mojo/kubernetes-storage-options-overview-persistent-volumes-pv-claims-pvc-and-storageclass-sc-k8s-storage-df71ca0fccc3

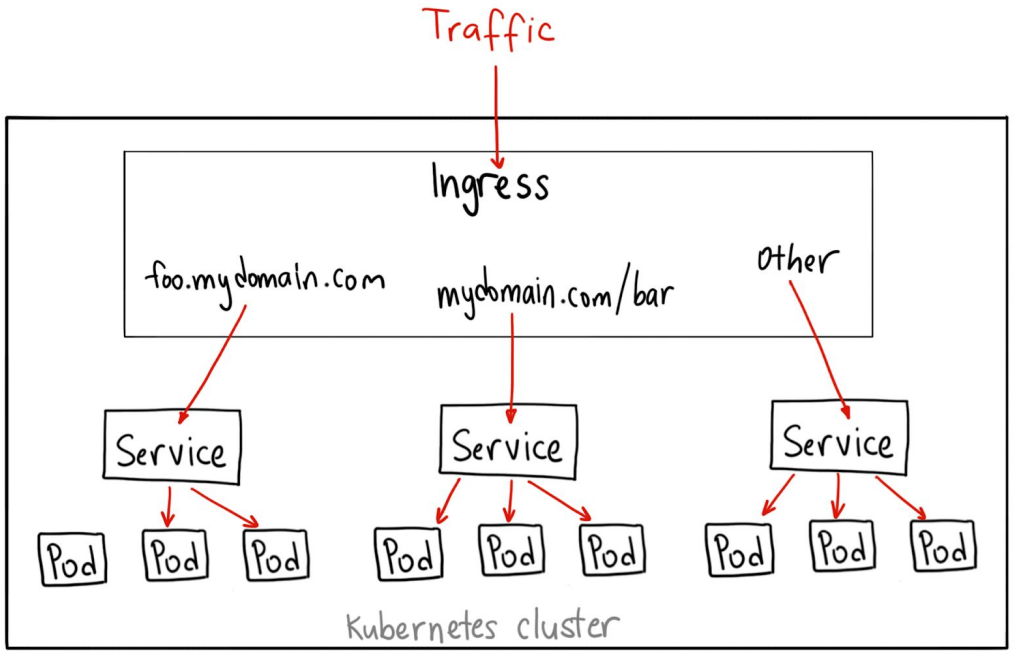
Taints & Tolerations



https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/

Ingress - Cómo exponemos nuestras apps?

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: myapp
spec:
  rules:
    - host: myapp-127-0-0-1.nip.io
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: myservice
                port:
                  number: 80
```



Helm