CRAFTECH

Your DevOps team

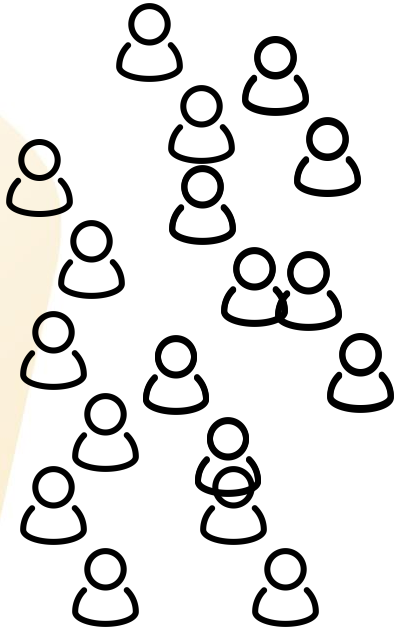
Módulo 4
Kubernetes

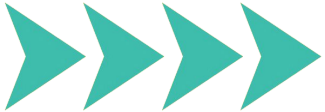
Repaso

- Volúmenes
- Redes
- Docker compose

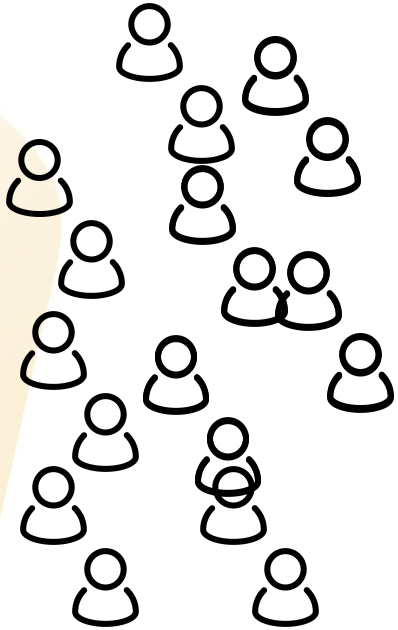
Nuevos problemas

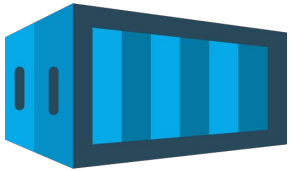
Tráfico

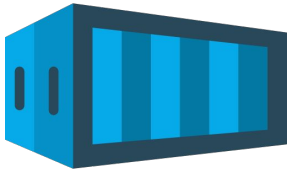




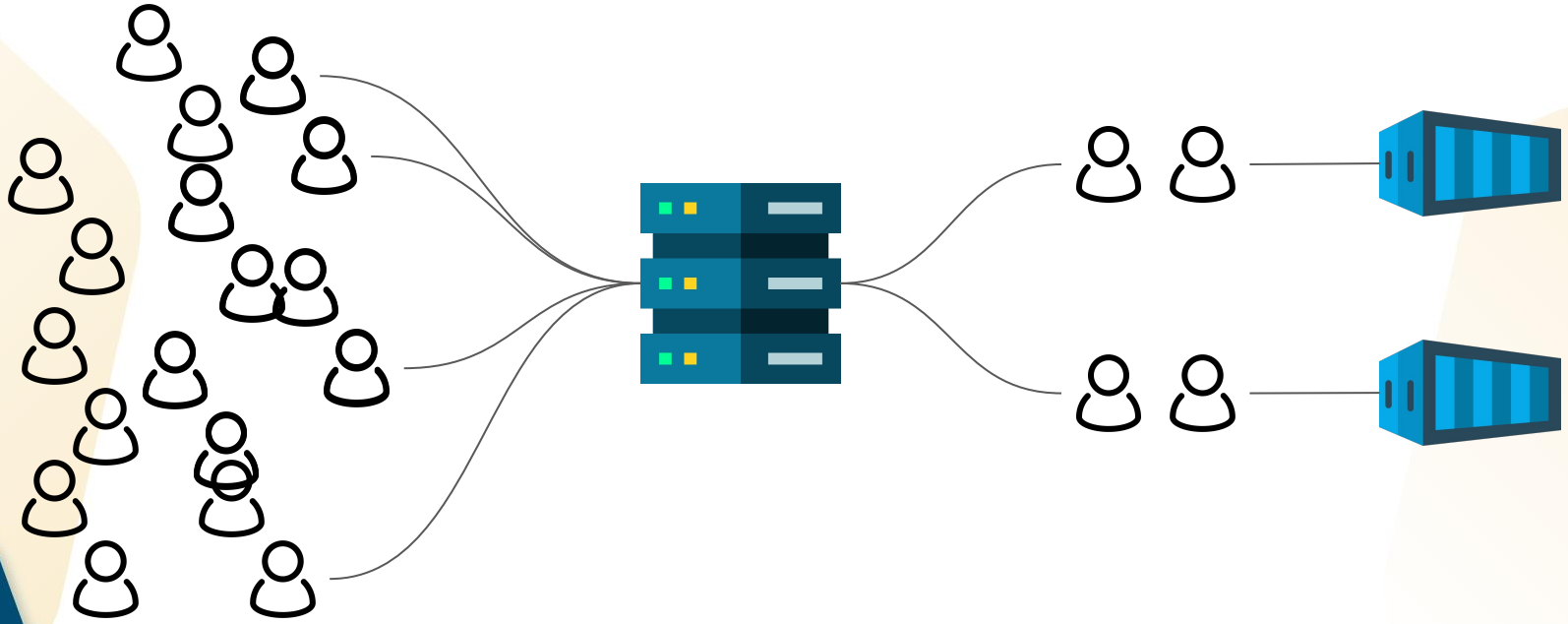
Tráfico



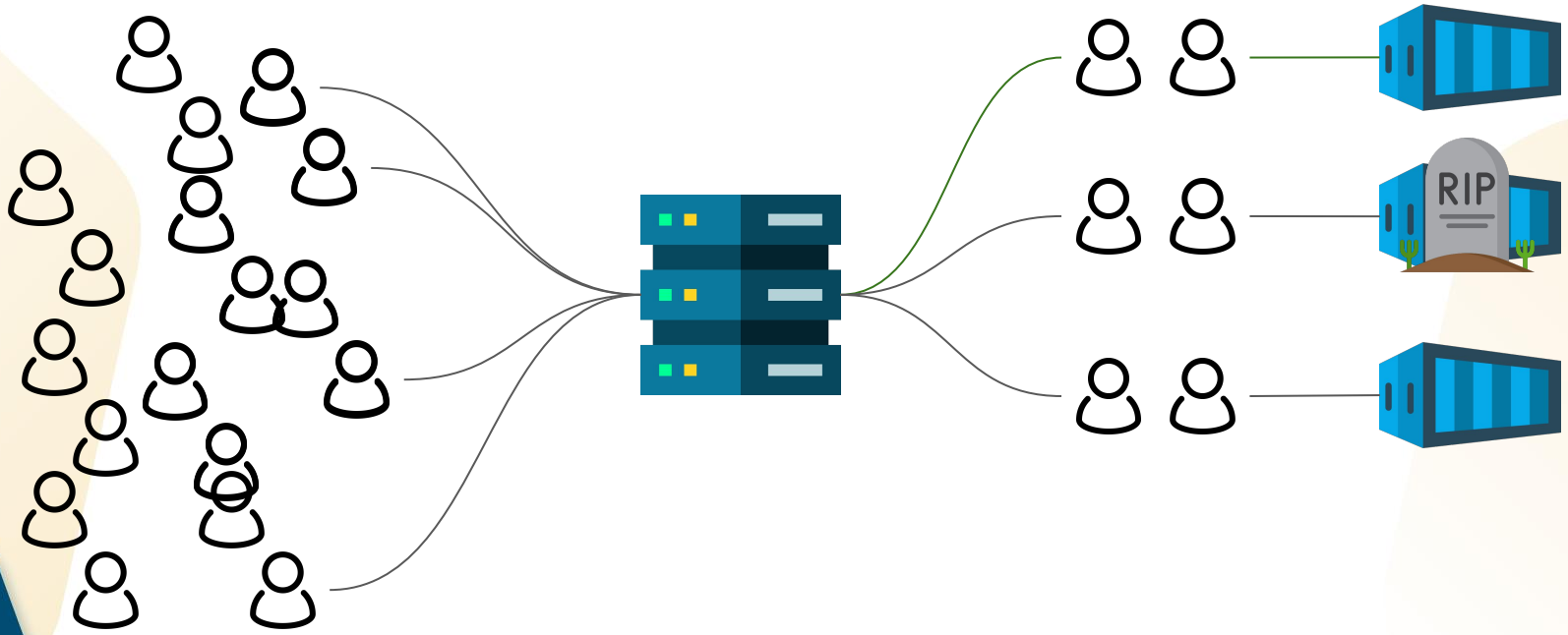




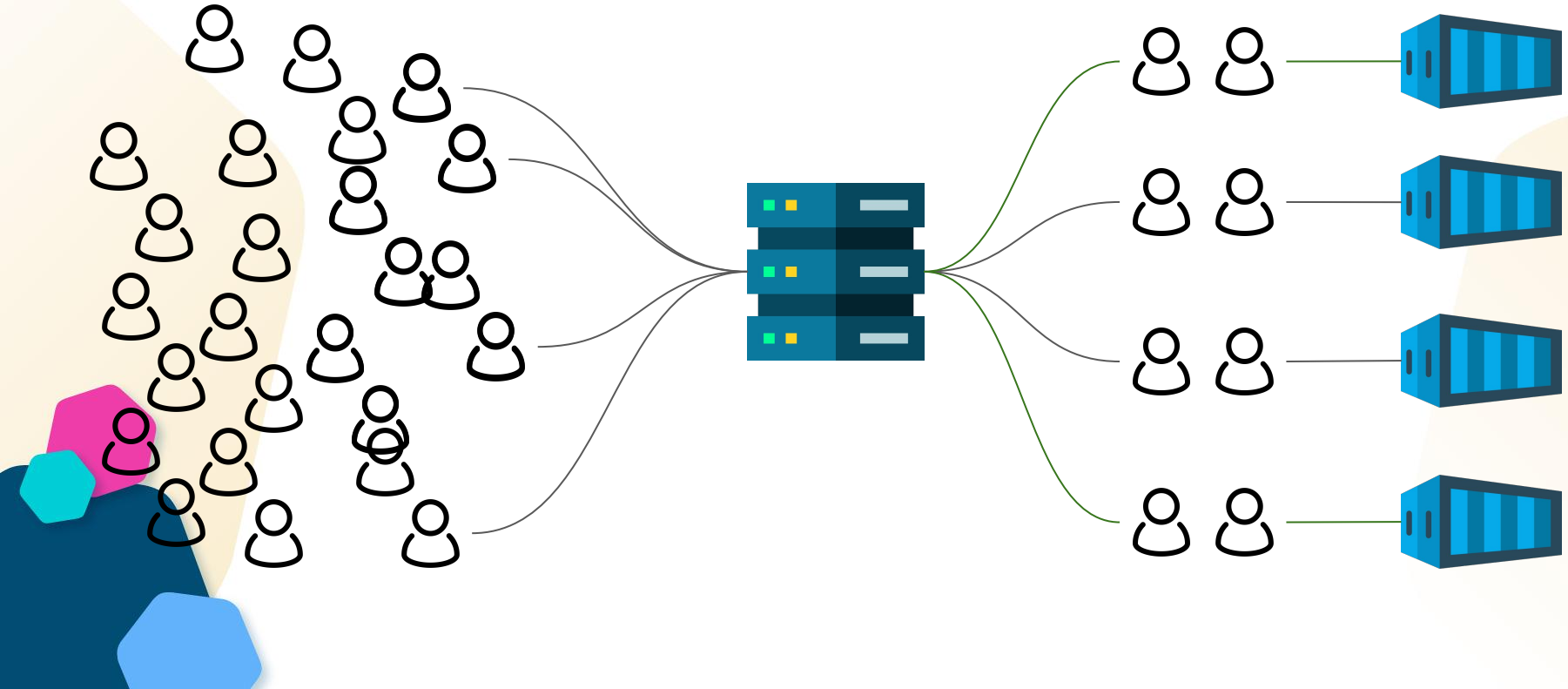
Balanceo de carga



Disponibilidad



Escalabilidad



Resumiendo...

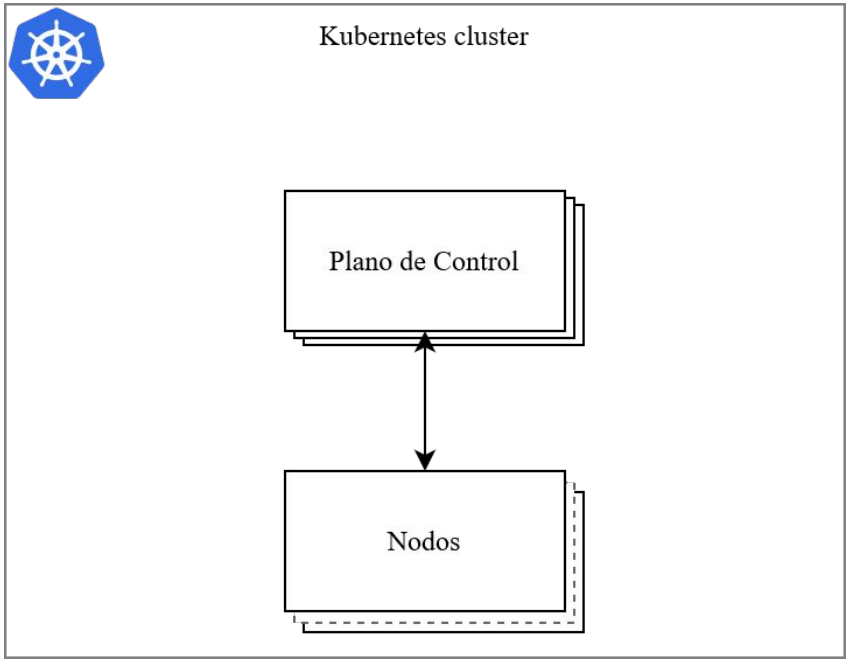
- Balanceo
  - Manual: configurar a qué contenedores se le puede enviar el tráfico.
- Disponibilidad
  - Solamente podemos controlar el reinicio del contenedor.
- Escalabilidad
  - Docker compose no lo soporta.

Orquestadores de contenedores

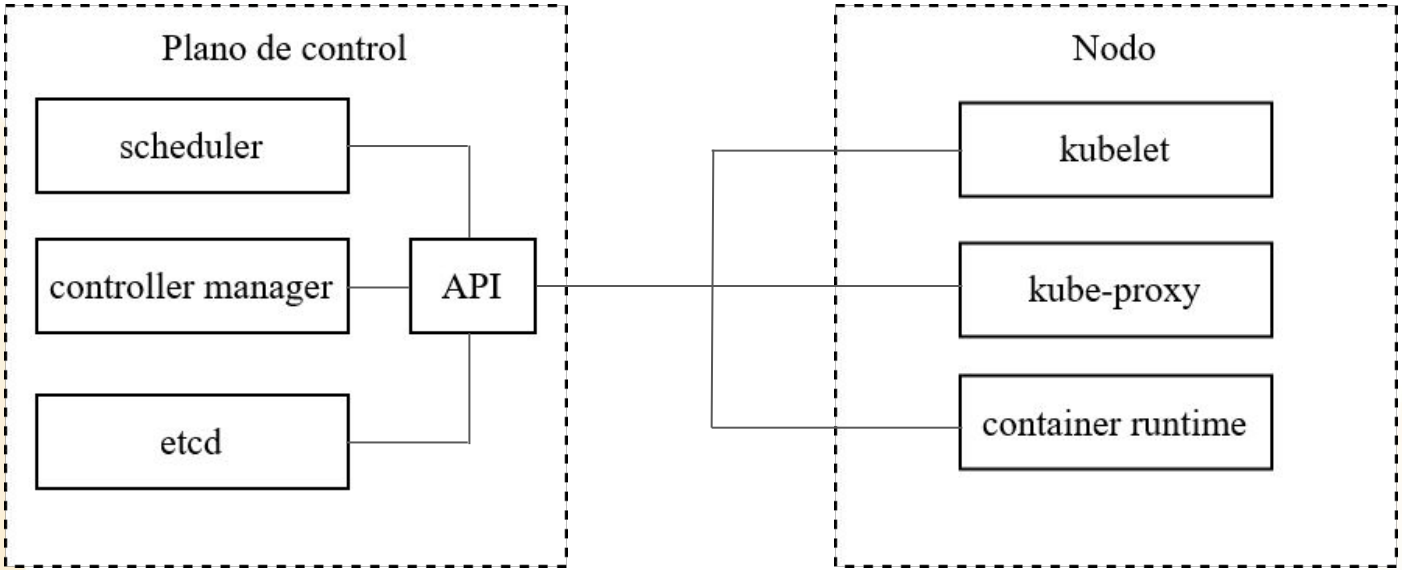
Kubernetes

- Sistema de despliegue de contenedores desarrollado por Google en 2014.
- Proviene del griego *κυβερνήτης*, que significa *piloto*.
- Versión 1.0 lanzada en 2015 y donado a la CNCF (*Cloud Native Computing Foundation*)
- Permite ganar agilidad y seguridad a la hora del despliegue de nuestro software.

Componentes



Componentes



Conceptos básicos

- Namespace
- Pod
- Service
- Ingress
- Deployment
- HPA

Namespace

- Clúster virtual
- Separación lógica
- Why?
  - Aislamiento
  - Organización
  - Manejo de permisos
  - Control de recursos
- Por defecto
  - default
  - kube-system
- Comandos:
  - Creación: `kubectl create namespace <NOMBRE>`
  - Obtención: `kubectl get namespace`

Pod

- Grupo de uno o más contenedores
  - Comparten storage, network y especificaciones de ejecución
  - Comunicación entre sí de forma local
- IP propia dentro del cluster
- Escalado horizontal y vertical

Pod

- Comandos:
  - Crear pod con archivo YAML:
    - `kubectl apply -f pod.yaml`
  - Crear pod sin archivo YAML:
    - `kubectl run nginx –it –image nginx`
    - `kubectl run nginx –it –rm –image nginx`

pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
   name: nginx
   labels:
     name: nginx
spec:
   containers:
   - name: nginx
     image: nginx
     ports:
     - containerPort: 80
```

Service

- Forma abstracta de exponer una aplicación que se ejecuta en un conjunto de Pods como un servicio de red.
- Tipos:
  - **ClusterIP** (default): utiliza una dirección IP interna del clúster.
  - **NodePort**: utiliza la dirección IP del nodo.
  - **LoadBalancer**: utiliza la IP del balanceador de carga, que enruta la solicitud a un *nodePort*.

Service

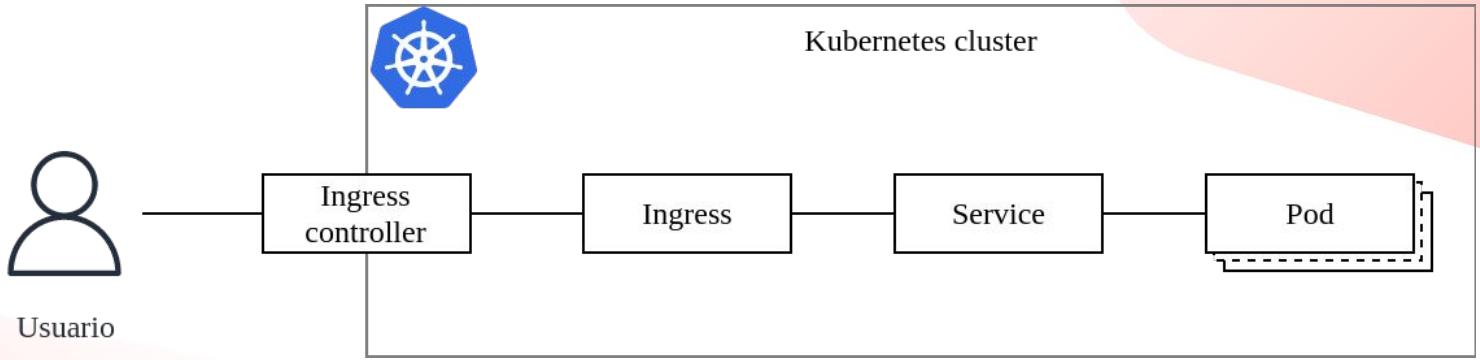
- Comandos:
  - Crear pod con archivo YAML:
    - **`kubectl apply -f service.yaml`**
  - Port-forwarding:
    - **`kubectl port-forward svc/nginx-svc <LOCAL_PORT>:<POD_PORT>`**

service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
   name: nginx-svc
spec:
   selector:
     name: nginx
   ports:
   - port: 80
     targetPort: 80
     protocol: TCP
```

Ingress

- Exponer mediante rutas HTTP/HTTPS los servicios internos del cluster.



Ingress

- Comandos:
  - Crear pod con archivo YAML:
    - **`kubectl apply -f ingress.yaml`**

ingress.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
   name: nginx-ingress
   annotations:
     <CONTROLLER_ANNOTATIONS>
spec:
  ingressClassName: nginx
  rules:
  - host: example.com
    http:
      paths:
      - path: /
      pathType: Prefix
      backend:
        service:
          name: nginx-svc
          port:
            number: 80
```

Juntando todo

pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
    name: nginx
    labels:
      name: nginx
spec:
    containers:
    - name: nginx
      image: nginx
      ports:
      - containerPort: 80
```

service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
    name: nginx-svc
spec:
    selector:
      name: nginx
    ports:
    - port: 80
      targetPort: 80
      protocol: TCP
```

ingress.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
    name: nginx-ingress
    annotations:
      <CONTROLLER_ANNOTATIONS>
spec:
  ingressClassName: nginx
  rules:
  - host: example.com
    http:
      paths:
      - path: /
      pathType: Prefix
      backend:
        service:
          name: nginx-svc
          port:
            number: 80
```

Deployment

- Administra los ReplicaSet.
- Define el estado deseado de un deploy.
- Recursos:
  - Request: recursos mínimos.
  - Límites: recursos máximos.
  - CPU:
    - **milicores**: 1 CPU == 1000m
  - Memoria:
    - **bytes**: 5G == 5000 MB
    - **bibytes**: 5Gi == 5368.7 MB

HPA

- *Horizontal Pod Autoscaling*
- Escalar automáticamente los pods de un Deployment según los recursos consumidos.

Muchos más...

- ReplicaSet
- StatefulSet
- DaemonSets
- Service Accounts
- Jobs
- Cronjobs

Documentación a todos estos recursos

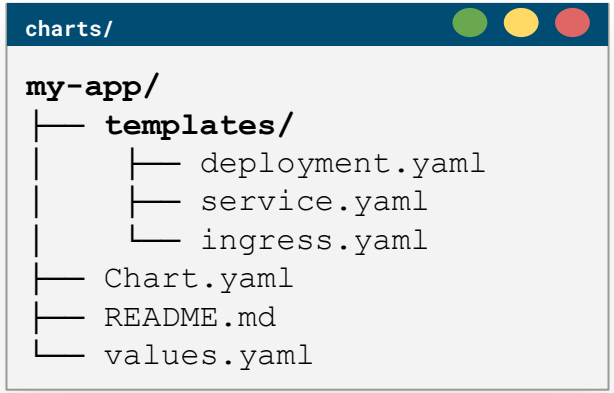
Helm

Manejador de paquetes para Kubernetes.

- Permite definir, instalar, actualizar y eliminar los recursos.

Helm

charts/


my-app/
templates/
deployment.yaml
service.yaml
ingress.yaml
Chart.yaml
README.md
values.yaml

¡Muchas gracias!