

ubuubun

Nombre: Yurben Ariel Puc tuz.		Matrícula: 3089082
Nombre del curso: Kubernetes y monitoreo		Nombre del profesor: Mariana Olivas
Módulo: 1		Actividad: 3
Fecha: 06/02/2026		
Bibliografía: <ul style="list-style-type: none">• <i>dayrafcarpio@gmail.com. (2025, August 8). CPU: Qué es, Función y Características del Procesador Guía 2025. PPS.</i> https://ppstech.mx/blog/cpu-que-es-y-cual-es-la-funcion-del-procesador/• <i>Glosarix. (2025, March 4). Solicitudes de Recursos de Pod - Glosarix.</i> https://glosarix.com/glossary/solicitudes-de-recursos-de-pod/• <i>Kubernetes: Recursos de Pod, resources.requests y resources.limits, cgroups de Linux. (n.d.).</i> https://www.q2bstudio.com/nuestro-blog/24180/kubernetes-recursos-de-pod-resources-requests-y-resources-limits-cgroups-de-linux• <i>I Mejor Esfuerzo (Best Effort) - 20 VPN's basadas en Firewall. (n.d.).</i> https://1library.co/article/i-mejor-esfuerzo-best-effort-vpn-basadas-firewall.q2n4w6pq• <i>Pod quality of service classes. (2026, January 10). Kubernetes.</i> https://kubernetes.io/docs/concepts/workloads/pods/pod-qos/?utm_source=copilot.com		

Actividad 3

1. Para cada uno de los siguientes recursos de Kubernetes, investiga en internet ejemplos de códigos de configuración y describe la utilidad de cada uno de ellos:
 1. Unidad Central de Procesamiento (CPU) – núcleos.
 1. interpretar, procesar y ejecutar las instrucciones de los programas y del sistema operativo. Piénselo como el director de una orquesta: no toca todos los instrumentos, pero lee la partitura (el software) y da las órdenes precisas a cada músico (los otros componentes como la memoria RAM o el disco duro) para que todo funcione en perfecta armonía y al ritmo adecuado.
 2. Memoria (MEM) – bytes.
 1. Se refiere a la capacidad y el rendimiento de la memoria RAM en un sistema informático. La memoria RAM es crucial para el funcionamiento eficiente de los sistemas, ya que permite al procesador acceder rápidamente a los datos y realizar operaciones sin ralentizar el sistema.
 3. Recursos de pods solicitados.
 1. Son fundamentales para la gestión eficiente de los recursos en un clúster, permitiendo al sistema programar Pods en nodos que tengan suficiente capacidad para satisfacer las necesidades.
 4. Limitar recursos de pod.
 1. Fija el consumo máximo que un contenedor puede usar: para memoria esto determina `memory.max` en el `cgroup` y si se supera puede activarse el OOM Killer; para CPU esto fija `cpu.max` o `cpu.cfs_quota` y provoca throttling si el grupo consume su cuota en un periodo.
2. Con base en las tres clases de Quality of Service (QoS) mencionados a continuación, investiga y ejemplifica, para cada una de ellas, un caso real de aplicaciones y escribe el código usado para poder utilizarla en el desarrollo de 3 sistemas (puedes referenciar a través de códigos de ejemplos usados en empresas que ya lo hayan implementado).
 1. **BestEffort.**
 1. Empresas de streaming como Netflix han usado *BestEffort* para tareas de análisis interno o procesamiento batch que no requieren prioridad. Si el nodo se queda sin recursos, estos pods son los primeros en ser eliminados.
 2. **Código de ejemplo:**

```
apiVersion: v1
kind: Pod
metadata:
  name: besteffort-pod
spec:
  containers:
  - name: analytics-job
```

```
image: busybox
command: ["sh", "-c", "echo Procesando logs && sleep
3600"]
```

2. **Guaranteed.**

1. En fintechs como Nubank o BBVA, los servicios críticos de pagos se ejecutan con **Guaranteed** para asegurar que siempre tengan los recursos necesarios.

2. **Código:**

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: guaranteed-pod
```

```
spec:
```

```
  containers:
```

```
    - name: payment-service
```

```
      image: myregistry/payment-api:latest
```

```
      resources:
```

```
        requests:
```

```
          cpu: "500m"
```

```
          memory: "512Mi"
```

```
        limits:
```

```
          cpu: "500m"
```

```
          memory: "512Mi"
```

```
      ports:
```

```
        - containerPort: 8080
```

3. **Burstable.**

1. En e-commerce como Amazon, los servicios de recomendación de productos se configuran como **Burstable**. Tienen recursos mínimos garantizados, pero pueden usar más si hay disponibilidad.

2. **Código;**

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: burstable-pod
```

```
spec:
```

```
  containers:
```

```
    - name: recommendation-engine
```

```
      image: myregistry/recommendation:latest
```

```
      resources:
```

```
        requests:
```

```
          cpu: "250m"
```

```
          memory: "256Mi"
```

```
        limits:
```

```
          cpu: "1"
```

```
          memory: "1Gi"
```

```
      ports:
```

```
        - containerPort: 5000
```