apiVersion: apps/v1

kind: Deployment

metadata:

name: my-web-app

labels:

app: my-web-app

spec:

replicas: 3 # Указываем количество реплик для обеспечения отказоустойчивости

selector:

matchLabels:

app: my-web-app

template:

metadata:

labels:

app: my-web-app

spec:

# Обеспечиваем, чтобы поды размещались на разных нодах

affinity:

podAntiAffinity:

requiredDuringSchedulingIgnoredDuringExecution:

- labelSelector:

matchExpressions:

- key: app

operator: In

values:

- my-web-app

topologyKey: "kubernetes.io/hostname" # Размещаем на разных хостах

containers:

- name: web-app-container

image: my-web-app:latest # Укажите образ вашего приложения

resources:

requests:

memory: "128Mi" # Запрашиваем 128M памяти

cpu: "500m" # Запрашиваем 0.5 CPU на начальный этап

limits:

memory: "128Mi" # Ограничиваем использование памяти

cpu: "1" # Ограничиваем использование CPU

lifecycle:

# Задержка перед обработкой запросов до завершения инициализации

preStop:

exec:

command: ["/bin/sh", "-c", "sleep 5"] # Ожидание 5 секунд перед остановкой пода

readinessProbe:

httpGet:

path: /health # Проверка готовности по эндпоинту

port: 8080 # Укажите порт, на котором работает приложение

initialDelaySeconds: 10 # Задержка перед первой проверкой готовности

periodSeconds: 5 # Периодические проверки готовности

---

apiVersion: v1

kind: Service

metadata:

name: my-web-app-service

spec:

selector:

app: my-web-app

ports:

- protocol: TCP

port: 80 # Порт, по которому доступно приложение

targetPort: 8080 # Порт, на который будет направлен трафик

type: ClusterIP # Тип сервиса, доступного внутри кластера

---

apiVersion: autoscaling/v1

kind: HorizontalPodAutoscaler

metadata:

name: my-web-app-hpa

spec:

scaleTargetRef:

apiVersion: apps/v1

kind: Deployment

name: my-web-app # Имя нашего deployment

minReplicas: 3 # Минимальное количество реплик

maxReplicas: 5 # Максимальное количество реплик для горизонтального масштабирования

metrics:

- type: Resource

resource:

name: cpu

target:

type: Utilization

averageUtilization: 50 # Целевое использование CPU в процентах

1. **Минимальное потребление ресурсов от этого deployment’а**:

* Мы указываем resources.requests для CPU как 100m (0.1 CPU) и для памяти как 128Mi. Это минимальные ресурсы, которые под будет запрашивать. Таким образом, под будет использовать только необходимые ему ресурсы, что позволяет минимизировать потребление ресурсов в кластере.
* Установлены ограничения (limits) на ресурсы, чтобы предотвратить превышение допустимых значений.

1. **Размещение подов на разных нодах для отказоустойчивости**:

* Используя affinity и podAntiAffinity, мы указываем, что поды не должны размещаться на одной и той же ноде. Это увеличивает отказоустойчивость, поскольку если одна нода выходит из строя, другие ноды продолжают обрабатывать запросы.
* topologyKey: "kubernetes.io/hostname" гарантирует, что поды будут распределены по разным хостам.

3. **Чтобы под не обрабатывал запросы до завершения инициализации**:

* **Readiness Probe**: Используем readinessProbe, чтобы контролировать, когда под готов к обработке запросов. Если под еще не готов, сервис не будет направлять на него трафик. Здесь мы устанавливаем начальную задержку (initialDelaySeconds: 10), чтобы под успел инициализироваться перед первой проверкой.
* **preStop Hook**: Включение preStop hook обеспечивает задержку перед остановкой пода, что позволяет завершить все текущие запросы перед его отключением.