Kubernetes Concepts: Pods, ReplicaSets, and YAML Manifest Files

In Kubernetes, **pods** are deployed on nodes, and the number of pods you can create depends on the number of available nodes in your cluster. To manage these resources effectively, YAML manifest files are used instead of running ad-hoc commands in real-time. Here's a concise guide to help you understand and practice these concepts:

YAML Manifest Files

- YAML is a data structure representation format that supports hierarchical data.
- It is widely used in Kubernetes to define configurations for various resources.

YAML uses key-value pairs for data representation, and lists can be created as follows:
yaml

```yaml
name: Jithendar
names:
  - Jithendar
  - Ramagiri
```

- Indentation is critical in YAML files, and dictionaries allow property modifications while lists must follow a sequence.

Example: Pod Manifest File
yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
  labels:
    env: dev
    app: facebook
spec:
  containers:
    - name: my-container
      image: <your-image-name>
```

Save the above file as mypod.yaml and apply it using:
bash

```bash
kubectl apply -f mypod.yaml
```

Important Notes on Pods:

- Every pod has a unique ID and IP address.
- When a pod is deleted, it won't automatically recreate unless you use a **ReplicaSet** or similar controller.

Example: ReplicaSet Manifest File

yaml

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-replicaset
  labels:
    app: facebook
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: facebook
    spec:
      containers:
        - name: my-container
          image: <your-image-name>
```

Save the file as myreplicaset.yaml and apply it using:

bash

Copy code

```
kubectl apply -f myreplicaset.yaml
```

Managing ReplicaSets:

Check the status of the ReplicaSet with:

bash

```
kubectl get rs
```

View all resources, including pods and ReplicaSets, with:

bash

```
kubectl get all
```

Delete a pod:

bash

```
kubectl delete pod <pod-name>
```

- The ReplicaSet will automatically create a new pod to maintain the desired replica count.

Scaling ReplicaSets:

- Update the desired replica count in the YAML file and reapply it.

Alternatively, scale up or scale down without editing the file using ad-hoc commands:
bash

```
kubectl scale rs <replicaset-name> --replicas=<desired-count>
```

Deleting a ReplicaSet:
To delete a ReplicaSet (and its pods):
bash

```
kubectl delete rs <replicaset-name>
```

Tips:

- Use **Visual Studio Code** with the YAML extension to easily write and validate manifest files.
- Practice these commands and YAML structures to master Kubernetes resource management.

This structured approach ensures efficient handling of Kubernetes resources while scaling and maintaining desired states. Let me know if you need further examples or clarifications!