What are different ways you ensure a Pod always runs?

To ensure a **Pod always runs in Kubernetes**, you can use one of the following approaches:

1. Use Deployments (Recommended)

A **Deployment** automatically manages and restarts Pods if they fail.

```
yaml
------
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 1  # Ensure at least one Pod is running
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-container
          image: my-image:latest
```

- **Kubernetes will restart the Pod** if it crashes.
- Use **replicas: 1** to ensure at least one instance always runs.

---

2. Use a DaemonSet (For Running on Every Node)

If you want a Pod to run on **every node**, use a **DaemonSet**:

```
yaml
------
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: my-daemon
spec:
```

```
  selector:
    matchLabels:
      app: my-daemon
  template:
    metadata:
      labels:
        app: my-daemon
    spec:
      containers:
        - name: my-container
          image: my-image:latest
```

- Ensures one Pod per node.
- Good for **monitoring agents, log collectors, etc.**

---

3. Set Restart Policy (For Single Pods, Not Recommended)

If using a standalone Pod (not managed by a Deployment), set `restartPolicy` to **Always**:

```
yaml
------
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  restartPolicy: Always  # Default setting
  containers:
    - name: my-container
      image: my-image:latest
```

- Works, but **not recommended**, as standalone Pods can be deleted manually.

In Kubernetes, "**CrashLoopBackOff**" signifies a pod that is repeatedly failing and restarting, with Kubernetes attempting to restart it with an increasing delay between restarts.

---

4. Use a PodDisruptionBudget (To Prevent Voluntary Disruptions)

If you're worried about **Pod evictions**, you can define a **PodDisruptionBudget (PDB)**:

```
yaml
------
apiVersion: policy/v1
kind: PodDisruptionBudget
metadata:
  name: my-pdb
spec:
  minAvailable: 1
  selector:
    matchLabels:
      app: my-app
```

- Ensures at least **one Pod is always available** during voluntary disruptions.

Best Approach?

✅ Use a **Deployment** if you need a single instance running all the time.
✅ Use a **DaemonSet** if the Pod must run on every node.
✅ Use a **PodDisruptionBudget** to prevent voluntary disruptions.

Which Kubernetes resource is best for ensuring that a Pod is always running, even if it crashes?

a) ConfigMap
b) Deployment
c) Service
d) Ingress

Answer: 🟢 b) Deployment

What is the default restart policy for a Pod in Kubernetes?

a) Never
b) OnFailure
c) Always
d) Manual

Answer: 🟢 c) Always

If you need to run a Pod on every node in a Kubernetes cluster, which resource should you use?

a) ReplicaSet
b) Deployment
c) DaemonSet
d) StatefulSet

Answer: 🟢 c) DaemonSet

Which Kubernetes feature ensures that at least a minimum number of Pods are always running during voluntary disruptions?

a) Horizontal Pod Autoscaler
b) PodDisruptionBudget
c) Liveness Probe
d) ConfigMap

Answer: 🟢 b) PodDisruptionBudget