# Docker Lab 12 — Highly Available Services in Docker Swarm

**Author: Dr. Sandeep Kumar Sharma**

---

## Lab Description

Lab 12 focuses on deploying **Highly Available (HA) services** in a Docker Swarm cluster. You will learn how Swarm distributes workloads, ensures availability, handles node failures, and keeps applications running 24/7.

You will create a multi-replica service, apply placement constraints, configure restart policies, and validate failover behavior.

---

## Topics Covered

- What is a Highly Available (HA) service?
- Replicated mode vs Global mode
- Placement constraints & preferences
- Restart policies (failure handling)
- Node drain & automatic rescheduling
- Handling Swarm node failures

---

## Learning Objectives

By the end of this lab, you will be able to: - Deploy services with high availability - Use constraints to control where services run - Configure Swarm to recover from failures - Demonstrate container rescheduling on node failure

---

## Learning Outcomes

After completing this lab, you will: - Understand how Swarm ensures zero downtime - Control placement of services across nodes - Handle node failures gracefully

---

# Section 1 — Prerequisites

You must have a **Swarm cluster** from Lab 11: - 1 Manager - 2 Workers

---

# Section 2 — Deploy a Highly Available Web Service

On **manager node**:

```
docker service create
  --name sandeep-ha-web
  --replicas 5
  --publish 8080:80
  nginx
```

Verify:

```
docker service ls
docker service ps sandeep-ha-web
```

You should see 5 tasks spread across all nodes.

---

# Section 3 — Apply Placement Constraints

Run only on worker nodes:

```
docker service update
  --constraint-add 'node.role==worker'
  sandeep-ha-web
```

Run on specific node:

```
docker service update
  --constraint-add 'node.hostname==swarm-worker1'
  sandeep-ha-web
```

Remove constraint:

```
docker service update --constraint-rm 'node.hostname==swarm-worker1' sandeep-ha-web
```

---

# Section 4 — Configure Restart Policies

```
docker service update
  --restart-condition any
  --restart-delay 5s
  --restart-max-attempts 3
  sandeep-ha-web
```

---

# Section 5 — Simulate Node Failure

Drain worker node:

```
docker node update --availability drain swarm-worker1
```

Observe automatic rescheduling:

```
docker service ps sandeep-ha-web
```

Set back to active:

```
docker node update --availability active swarm-worker1
```

---

# Section 6 — Scaling the HA Service

```
docker service scale sandeep-ha-web=10
```

Verify again:

```
docker service ps sandeep-ha-web
```

---

# Section 7 — Cleanup

```
docker service rm sandeep-ha-web
```

---

# Summary

You successfully deployed and tested a highly available distributed service in Docker Swarm. You used constraints, restart policies, scaling, and node failover to ensure reliability and uptime.