**POC Document: Blue-Green Deployment on AKS using Ingress Switching**

**1. Objective**

The purpose of this Proof of Concept (POC) is to demonstrate a Blue-Green deployment strategy on Azure Kubernetes Service (AKS) using an Ingress controller to switch traffic between two identical application environments (Blue and Green) with minimal to zero downtime.

**2. Overview of Blue-Green Deployment**

Blue-Green deployment maintains two identical production environments:

* **Blue** – The currently live environment serving production traffic.
* **Green** – The new version of the application to be deployed.

Traffic is switched from Blue to Green after validation, ensuring:

* Minimal downtime
* Easy rollback if issues occur
* No disruption to in-flight requests during the switch

**4. Tools & Technologies Used**

* **Azure Kubernetes Service (AKS)** – Managed Kubernetes cluster
* **NGINX Ingress Controller** – For routing and traffic switching
* **Azure Container Registry (ACR)** – For storing Docker images
* **Helm** – For managing Kubernetes manifests
* **kubectl** – For cluster management
* **Azure CLI** – For provisioning resources

**5. Implementation Steps**

**Deploy Blue Environment (blue-deployment.yaml)**

apiVersion: apps/v1

kind: Deployment

metadata:

name: blue-deployment

spec:

replicas: 2

selector:

matchLabels:

app: myapp

version: blue

template:

metadata:

labels:

app: myapp

version: blue

spec:

containers:

- name: myapp

image: myacr.azurecr.io/myapp:v1

ports:

- containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

name: blue-service

spec:

selector:

app: myapp

version: blue

ports:

- protocol: TCP

port: 80

targetPort: 80

**Deploy Green Environment (green-deployment.yaml)**

apiVersion: apps/v1

kind: Deployment

metadata:

name: green-deployment

spec:

replicas: 2

selector:

matchLabels:

app: myapp

version: green

template:

metadata:

labels:

app: myapp

version: green

spec:

containers:

- name: myapp

image: myacr.azurecr.io/myapp:v2

ports:

- containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

name: green-service

spec:

selector:

app: myapp

version: green

ports:

- protocol: TCP

port: 80

targetPort: 80

**Configure Ingress to Route to Blue (ingress.yaml)**

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: myapp-ingress

annotations:

kubernetes.io/ingress.class: nginx

spec:

rules:

- host: myapp.example.com

http:

paths:

- path: /

pathType: Prefix

backend:

service:

name: blue-service

port:

number: 80

**Validate Blue Deployment**

* Access http://myapp.example.com
* Ensure application shows **Blue version**
* **Switch Traffic to Green**
* Modify ingress.yaml to point to green-service:

backend:

service:

name: green-service

port:

number: 80

**Rollback if Needed**

Revert ingress.yaml to blue-service and re-apply.

**6. Testing & Validation**

* **Downtime check** – Used curl in a loop; no failed requests during switch
* **Version verification** – Page clearly displayed version info (v1 vs v2)
* **Rollback test** – Successful instant revert to Blue

**7. Key Benefits Observed**

* Near-zero downtime deployments
* Safe rollback option
* Parallel environment testing

**8. Limitations in POC**

* No database migration tested
* Health checks before traffic switch not implemented
* No automation pipeline integration in this POC

**9. Next Steps**

* Integrate with Azure DevOps/GitHub Actions
* Implement health checks
* Extend to multi-region deployment