**📦 Project Deliverables**

**1. Source Code Repository**

**GitHub/GitLab Repository:**

**https://github.com/your-username/k8s-assignment**

**(Replace with your actual repository URL)**

**Included:**

**All Kubernetes YAML files (yaml directory)**

**Dockerfile**

**Application source code**

**README file**

**2. Docker Image**

**Docker Hub URL:**

**https://hub.docker.com/r/your-dockerhub-username/k8s-assignment**

**(Replace with your actual Docker Hub image URL)**

**3. Service API Tier URL**

**API URL (via Ingress):**

**http://k8s-api.local/api/employee**

**(Replace with your actual Ingress/external URL)**

**4. Screen Recording Video**

**Video Link:**

**[Google Drive/YouTube/OneDrive link to your screen recording]**

**(Upload your video and paste the link here)**

**What to show in the video:**

**All Kubernetes objects deployed and running (kubectl get all)**

**API call retrieving records from the database (e.g., using curl or browser)**

**Kill the API pod and show it regenerates (kubectl delete pod <api-pod-name>)**

**Kill the database pod and show it regenerates and retains data (kubectl delete pod <db-pod-name>)**

**📖 Documentation**

**Requirement Understanding**

**This project demonstrates a cloud-native microservices deployment using Kubernetes. The solution includes an API tier and a backend database, with all configuration externalized using ConfigMaps and Secrets. The API is exposed externally via Ingress, and all inter-tier communication uses Kubernetes service discovery (not Pod IPs).**

**Assumptions**

**The Kubernetes cluster is pre-provisioned and accessible.**

**Docker images are built and pushed to Docker Hub.**

**NGINX Ingress controller is installed in the cluster.**

**The user has access to update their local hosts file for Ingress testing.**

**Persistent storage is available for the database.**

**Solution Overview**

**API and DB are deployed as separate pods using Deployments/StatefulSets.**

**All configuration (including DB credentials) is externalized using ConfigMaps and Secrets.**

**API tier is exposed externally using an Ingress resource, not a LoadBalancer.**

**Pods are managed by Kubernetes, ensuring self-healing and high availability.**

**Database data is persisted using a PersistentVolumeClaim.**

**Docker images are built from the provided Dockerfile and hosted on Docker Hub.**

**Justification for the Resources Utilized**

**Deployments/StatefulSets:**

**Used for managing stateless (API) and stateful (DB) workloads, ensuring scalability and resilience.**

**ConfigMaps & Secrets:**

**Enable secure, environment-specific configuration without code changes.**

**ClusterIP Services:**

**Used for internal communication, following best practices to avoid Pod IPs.**

**Ingress:**

**Provides a single entry point for external traffic, supporting host-based routing and TLS termination.**

**PersistentVolumeClaim:**

**Ensures database data is retained across pod restarts or rescheduling.**

**NGINX Ingress Controller:**

**Industry-standard, widely supported, and easy to automate in any cloud environment.**

**# K8s Assignment**

**## Repository**

[https://github.com/your-username/k8s-assignment](https://github.com/your-username/k8s-assignment)

**## Docker Image**

[https://hub.docker.com/r/your-dockerhub-username/k8s-assignment](https://hub.docker.com/r/your-dockerhub-username/k8s-assignment)

**## API URL**

[http://k8s-api.local/api/employee](http://k8s-api.local/api/employee)

**## Demo Video**

[Google Drive/YouTube/OneDrive link]

**## How to Deploy**

1. Clone the repo and cd into the project.

2. Build and push the Docker image.

3. Apply all Kubernetes YAMLs in the `yaml/` directory.

4. Update your hosts file to map the Ingress IP to `k8s-api.local`.

5. Access the API via browser or curl.

**## Kubernetes Objects**

- Deployment/StatefulSet for API and DB

- ConfigMap and Secret for configuration

- ClusterIP Services for internal communication

- Ingress for external access

- PersistentVolumeClaim for DB data

**## Self-Healing Demo**

- Delete an API pod: `kubectl delete pod <api-pod>`

- Delete a DB pod: `kubectl delete pod <db-pod>`

- Both will be automatically recreated by Kubernetes, and DB data will persist.

**## Contact**

For any questions, open an issue or contact [your-email@example.com].