**Documentation**

**1️⃣ Requirement Understanding**

The system consists of:

* **Service/API Tier**:
  + 4 replicas
  + Rolling updates enabled
  + Exposed outside the cluster via LoadBalancer
  + No persistent storage required
* **Database Tier**:
  + 1 replica
  + No rolling updates (Recreate strategy)
  + Persistent storage enabled
* **CI/CD**:
  + GitHub Actions builds and pushes Docker image to Docker Hub
  + Deploys manifests to GKE

**2️⃣ Assumptions**

* Kubernetes cluster is **Google Kubernetes Engine (GKE)**
* Docker Hub stores container images
* MongoDB credentials:
  + Stored securely in ConfigMap and Secret
* API endpoint is publicly accessible via LoadBalancer

**3️⃣ Solution Overview**

**Components:**

1. **Product Service (API Tier)**:
   * Spring Boot REST API
   * Deployment with 4 replicas
   * LoadBalancer service for public access
   * Rolling update strategy
2. **MongoDB (Database Tier)**:
   * Single replica
   * ClusterIP service for internal access
   * PVC for persistence
   * Recreate strategy to avoid corruption
3. **CI/CD (GitHub Actions)**:
   * On push to main, build Docker image
   * Push image to Docker Hub
   * Deploy manifests using kubectl apply

**4️⃣ Justification for Resources Utilized**

* **4 API Replicas** → High availability and load balancing
* **1 DB Replica** → Simplicity for single instance DB
* **Rolling Update for API** → Zero-downtime deployments
* **Recreate Strategy for DB** → Ensures data consistency
* **PVC for MongoDB** → Retains data after restarts
* **LoadBalancer for API** → Public access
* **ClusterIP for MongoDB** → Internal security