# Kubernetes & DevOps Documentation

GitHub Link: <https://github.com/yankit-chauhan/NAGP-2025-Devops>

## Requirement Understanding

The goal of this project is to design, containerize, and deploy a multi-tier application on Kubernetes consisting of:

* A **Service API tier** (microservice) that exposes REST endpoints and interacts with a database.
* A **Database tier** that provides persistent data storage and is not exposed externally.

Key requirements:

* The microservice must **fetch data from the database**.
* The **Docker image** of the microservice should be built and pushed to Docker Hub.
* All secrets and config must be handled securely via **Secrets** and **ConfigMaps**.
* The application should be deployed to a **Kubernetes cluster**, exposing only the microservice externally through **Ingress**.
* The application must support:
  + **Rolling updates** for the microservice tier.
  + **Data persistence** for the database.
  + **Auto-recreation** of pods without data loss.

## Assumptions

* The Kubernetes environment is hosted on **Google Kubernetes Engine (GKE)**.
* MySQL is used as the database.
* Java (Spring Boot) is used to build the microservice.
* The microservice connects to the database using JDBC.
* Docker images are hosted on **Docker Hub**.
* nip.io is used to simplify DNS resolution for Ingress.

## Solution Overview

**🎯 Microservice Tier**

* Built using **Gradle + Spring Boot + MySQL Connector**.
* Dockerized and hosted on Docker Hub.
* Configured using ConfigMap and Secret for clean separation.
* Exposed using **Ingress**.
* Deployment ensures **rolling updates** and scalability via replica count = 4.

**🎯 Database Tier**

* MySQL with a predefined table containing 5–10 records(defined in **ConfigMap**).
* Password stored using **Kubernetes Secret**.
* Config (host, database name, etc.) passed using **ConfigMap**.
* Mounted volume using **PersistentVolumeClaim (PVC)** to ensure **data persistence**.
* Only 1 replica (no rolling updates needed).

## Kubernetes Resources Breakdown

|  |  |
| --- | --- |
| Resource | Description |
| ConfigMap | Contains DB host, DB name, and user (excluding password). |
| Secret | Stores DB password securely. |
| PVC | Mounted to MySQL container to retain data. |
| Deployment | Two separate deployments: mysql (1 pod), app (4 pods). |
| Service | Internal ClusterIP for MySQL; ClusterIP for app used by Ingress |
| Ingress | Exposes the microservice to the internet via nip.io. |

## Justification for the Resources Utilized

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tier | Replicas | Storage | Exposure | Update Strategy |
| Microservice | 4 | None | Yes (Ingress) | Rolling updates |
| Database | 1 | PVC | No | Recreate only |

* **Service Type**:
  + MySQL → ClusterIP: internal-only access.
  + Microservice → ClusterIP + Ingress: publicly accessible through Ingress controller.
* **Persistent Storage**:
  + Only required for MySQL to retain data across pod restarts.
* **Ingress Controller**:
  + Used to map external requests to internal microservice.
  + Deployed via official NGINX Ingress YAML for GKE.