Case Study: Single-Cloud DevOps Setup for “QuickShop”

Background

A small company called **QuickShop** is building a simple web application for selling products online.
They have:

- A **frontend** (React)
- A **backend API** (Node.js or Python)
- A **PostgreSQL database**

Currently, developers:

- Write code on their laptops
- Zip the code and send it to a freelancer
- The freelancer manually uploads it to a cloud VM

QuickShop wants to **move to a proper DevOps setup**.

Client Requirements :

1. **Cloud**
    - They want to host everything on **one cloud** (Azure).
    - In the future, they will be moving to scale the app to multi-cloud infra
2. **Architecture**
    - The app should run in **containers**.
    - They want one **Kubernetes cluster** to run:

        - frontend
        - backend
    - A database can be a managed DB service or a separate VM/container.
3. **Code & Repositories**
    - They **do not** want to expose code publicly.
    - All code must stay in **private repositories**.
4. **CI/CD**
    - They want an **automated deployment**:
        - When code is pushed, it should:
            - Build the app
            - Run tests
            - Build a Docker image
            - Deploy to Kubernetes (at least for “dev”)
    - They want **manual approval** before deploying to **production**.
5. **Security**
    - No public repos.
    - Secrets (DB password, API keys) should not be hardcoded in the code.
    - Only limited users should be able to deploy to production.
6. **Cost**
    - They don’t want the **cheapest unstable** option.
    - They are okay with paying **a reasonable amount for a stable and scalable setup**, but not enterprise-level expensive.

STUDENT TASK – DevOps Proposal Writing

Objective:

To train students on how real DevOps engineers understand client needs, ask clarifying questions, and prepare multiple solutions based on cost, efficiency, and scalability.

Background (Already Explained to Students)

You have been given the background and client requirements for a small company that wants to deploy their application on a single cloud.
They want:

- A **cluster-based** infrastructure
- Code stored in **private repositories**
- **Automated CI/CD**
- A setup that is **secure**, **scalable**, and **cost-justified**
- Zero lock-in, so the system can be extended to multi-cloud later

The client has *not* requested any specific tools or technologies.
It is your responsibility to choose them.

Your Task (Each Student Must Submit the Following)

1 Prepare TWO separate proposals for the client

Each proposal should be **2–3 pages max**

Proposal A – High Efficiency, Higher Cost

- Focus on **performance**, availability, scalability, maintainability.
- Choose components that give the best results even if they cost more.
- Example thought process (**not actual instructions**):
    - “Should I choose managed services or self-hosted?”
    - “Should I prefer HA clusters or a single node?”
    - “Should I add monitoring systems separately?”
    - “Should I add redundancy, backups, auto-scaling aggressively?”

Your architecture must reflect **quality > cost**.

Proposal B – Lower Cost, Lower Efficiency

- Focus on affordability and minimal infrastructure.
- Slight compromises on performance, speed, and security are acceptable *as long as the system works*.
- Example thought process (not actual instructions):
    - “Can we use a small cluster or VM-based setup?”
    - “Can we use simpler components?”
    - “Can we avoid expensive managed services?”

Your architecture must reflect **cost > efficiency**.

---

2 Students must decide all tools & technology stacks themselves

Do NOT ask the instructor “which tool should we use?”
The purpose is for you to **think like a DevOps engineer** and propose:

- Your own tools
- Your own cloud services
- Your own design
- Your own CI/CD strategy
- Your own security decisions
- Your own architecture diagrams

There is no wrong answer.
There is only **justification**.

If you can justify a tool, you can use it.

---

3 Each Student Must Prepare 5 Client Questions (Discovery Questions)

Before a real DevOps project begins, engineers always ask the client clarifying questions.

Each student must prepare **five (5)** smart questions that they would ask the client in a meeting.

These questions can be about:

- Logging strategy
- Deployment frequency
- Security expectations

- Backup and recovery
- Compliance
- Traffic load
- Testing process
- Repo access control
- Secrets management
- Environments (dev/stage/prod)
- Budget
- Monitoring
- Scalability needs

Examples (do NOT copy these — create your own):

- “How do you want us to manage logs and archival?”
- “Do you have an internal security policy for access control?”
- “What kind of backups and failover strategy do you expect?”
- “What is your peak traffic expectation?”
- “Do you require deployment approvals or fully automated deploys?”

Your questions should help you build a more accurate proposal.