## 🔹 ****KUBERNETES (Advanced Level)****

**1. Architecture & Internals**

* Explain the architecture of Kubernetes: role of kube-apiserver, kubelet, etcd, scheduler.
* How does the control plane work?
* What happens when you run kubectl apply?
* How does service discovery and networking work internally?

**2. Scheduling and Resource Management**

* How does Kubernetes decide where to schedule a pod?
* Difference between requests and limits? What happens if a pod exceeds its limit?

**3. Deployments and Rolling Updates**

* How do rolling updates work under the hood?
* How would you perform a zero-downtime deployment?
* What are readiness and liveness probes? Why are they important?

**4. Debugging & Failures**

* A pod is in CrashLoopBackOff — how do you debug?
* How would you troubleshoot network issues between pods?

**5. Advanced Concepts**

* How do you use **operators** or **Custom Resource Definitions (CRDs)**?
* How do **init containers**, **sidecars**, or **volumes** work?
* How do you handle secrets and config maps securely?
* Differences between **ReplicaSet**, **StatefulSet**, and **DaemonSet**?

**6. Security and RBAC**

* How do you set up RBAC?
* What’s a PodSecurityPolicy or admission controller?
* How do you ensure isolation between namespaces?

## 🔹 ****DOCKER (Advanced Level)****

**1. Docker Internals**

* What happens when you run docker run?
* What are layers in Docker images and how is caching handled?
* How does the union file system work?

**2. Optimization**

* How do you reduce image size?
* Difference between COPY and ADD, CMD vs ENTRYPOINT.

**3. Docker Networking**

* How do bridge, host, and overlay networks work?
* How would you connect two containers without using --link?

**4. Debugging & Performance**

* How do you debug a container that’s failing to start?
* How do you check and limit CPU/RAM usage of a container?

**5. Security**

* How do you scan Docker images for vulnerabilities?
* How do you run containers as non-root users?

## 🔹 ****GITLAB CI/CD (Advanced Level)****

**1. Deep Understanding of** .gitlab-ci.yml

* Explain how stages, jobs, and artifacts work.
* What is the difference between rules, only/except, and when?

**2. Pipelines & Runners**

* How do GitLab Runners work? Difference between shell, Docker, Kubernetes runners?
* How would you run CI/CD jobs in parallel?

**3. Advanced Scenarios**

* How do you set up **multi-environment** deployment (dev/stage/prod)?
* How do you do canary deployments or blue-green deployments in GitLab?
* How do you use GitLab with Kubernetes for Auto DevOps?

**4. Security and Secrets Management**

* How do you store and use secrets securely?
* How do you prevent secret leakage in logs?

**5. Optimization & Caching**

* How do you cache dependencies between jobs?
* How do you speed up slow pipelines?