DevOps & Multi-Cloud Tools Interview Q&As

🔹 Section 1: DevOps + CI/CD Fundamentals

Q1. What is CI/CD?
Answer: CI/CD automates integration, testing, and deployment of code. CI ensures frequent code merges, CD delivers updates reliably to production.
💡 *Tip: Stress the benefits — speed, automation, and reduced human errors.*

Q2. CI vs CD (Delivery vs Deployment).
Answer: Continuous Delivery = production-ready builds needing manual approval. Continuous Deployment = auto-release to production without manual steps.
💡 *Tip: Interviewers like when you clarify both terms with real-life release examples.*

Q3. What are GitHub Actions?
Answer: A GitHub-native CI/CD service that automates build, test, and deploy workflows directly from repositories.
💡 *Tip: Mention YAML workflows and GitHub Marketplace actions.*

Q4. How do you secure secrets in pipelines?
Answer: Use encrypted stores (GitHub Secrets, Vault, AWS Secrets Manager). Never hard-code credentials.
💡 *Tip: Highlight "rotation of secrets" and compliance as best practices.*

Q5. GitHub Actions vs GitLab CI/CD.
Answer: GitHub Actions is lightweight and integrated with GitHub. GitLab CI/CD is enterprise-grade with built-in DevOps tools.
💡 *Tip: Show you understand trade-offs: simplicity (GitHub) vs enterprise features (GitLab).*

🔹 Section 2: Jenkins & Automation

Q6. What is Jenkins?
Answer: Jenkins is an open-source automation server for CI/CD pipelines with 1,800+ plugins.
💡 *Tip: Interviewers expect you to know it's "the most popular traditional CI/CD tool."*

Q7. Jenkins vs GitHub Actions.
Answer: Jenkins = customizable, plugin-rich, but self-managed. GitHub Actions = cloud-native and simpler for GitHub repos.
💡 *Tip: Add "Jenkins fits legacy setups; GitHub Actions fits cloud-native apps."*

Q8. Declarative vs Scripted pipelines in Jenkins.
Answer: Declarative = structured YAML-like syntax. Scripted = flexible Groovy scripting for advanced logic.
💡 *Tip: Say you'd prefer declarative for team adoption, scripted for complex logic.*

Q9. What are Jenkins Agents/Nodes?
Answer: Worker machines that execute jobs assigned by the master (controller). They allow distributed builds.
💡 *Tip: Explain how agents help scale CI/CD for large teams.*

Q10. How do you integrate Jenkins with AWS/Azure/GCP?
Answer: Use cloud plugins, SDKs, and credentials (e.g., AWS CodeDeploy plugin, Azure DevOps, GCP SDK).
💡 *Tip: Mention securing credentials with Jenkins Credentials Manager.*

🔹 Section 3: Cloud-Native DevOps Tools

Q11. What is Infrastructure as Code (IaC)?
Answer: IaC provisions infrastructure via code (Terraform,

CloudFormation) for repeatability and automation.
💡 *Tip: Say "treat infra like code" — versioned, reviewed, automated.*

Q12. Terraform vs CloudFormation.
Answer: Terraform = multi-cloud, modular, HCL-based.
CloudFormation = AWS-native only.
💡 *Tip: Stress Terraform's portability; CloudFormation's deep AWS integration.*

Q13. What is Kubernetes' role in DevOps?
Answer: Kubernetes orchestrates containers: scaling, load balancing, and recovery across clusters.
💡 *Tip: Be ready to give an example like "auto-scaling a microservices app."*

Q14. What is Helm, and how is it used?
Answer: Helm is a package manager for Kubernetes. It deploys apps using "charts" that simplify YAML configs.
💡 *Tip: Call Helm the "npm for Kubernetes."*

Q15. Docker vs Kubernetes in DevOps.
Answer: Docker = containerization. Kubernetes = orchestration and lifecycle management of containers.
💡 *Tip: Interviewers love when you say "Docker builds, K8s scales."*

Q16. What is a service mesh (e.g., Istio)?
Answer: Service mesh manages microservice communication with traffic routing, observability, and security.
💡 *Tip: Mention "east-west traffic" and zero-trust service-to-service auth.*

🔹 Section 4: Multi-Cloud Comparison

Q17. Compare EC2 vs Azure VMs vs Compute Engine.
Answer: All are VM services. EC2 = highly flexible, Azure VMs =

enterprise focus, Compute Engine = AI/ML integration.
💡 *Tip: Don't just list — explain which is better for which use case.*

Q18. Compare S3 vs Blob Storage vs Cloud Storage.
Answer: All are object storage. S3 = ecosystem leader, Blob = Microsoft-native, Cloud Storage = globally consistent.
💡 *Tip: Add one-liner: "S3 dominates adoption."*

Q19. Compare IAM models in AWS, Azure, and GCP.
Answer: AWS = IAM roles/policies. Azure = AD + RBAC. GCP = resource hierarchy (org > project > resource).
💡 *Tip: Always connect IAM to **least privilege** principle.*

Q20. EKS vs AKS vs GKE.
Answer: EKS = AWS, AKS = Azure, GKE = Google. GKE is most automated; AKS integrates with AD; EKS strongest AWS integration.
💡 *Tip: Say "GKE is most mature."*

Q21. Pros & cons of multi-cloud strategy.
Answer: Pros: no lock-in, high availability. Cons: cost, skill gaps, complexity.
💡 *Tip: Add "start hybrid, then move multi-cloud gradually."*

Q22. IAM challenges in multi-cloud setups.
Answer: IAM differs across clouds, leading to policy sprawl and duplication. Centralized identity is hard to enforce.
💡 *Tip: Suggest federated identity via SSO as a solution.*

🔹 Section 5: DevOps Practices Across Clouds

Q23. Blue-Green vs Canary deployments.
Answer: Blue-Green = two identical environments, switch traffic instantly. Canary = roll out to small users first.
💡 *Tip: Say Blue-Green is faster rollback; Canary is safer testing.*

Q24. Monitoring & logging in pipelines.
Answer: Use ELK, Prometheus/Grafana, CloudWatch, Stackdriver, or

Azure Monitor for metrics and logs.
💡 *Tip: Interviewers expect you to mention “alerting and dashboards.”*

Q25. GitOps vs DevOps.
Answer: DevOps = culture + practices. GitOps = uses Git as the single source of truth for infra + apps.
💡 *Tip: Keep it simple: “GitOps = DevOps powered by Git.”*

Q26. Secret management across AWS, Azure, GCP.
Answer: AWS Secrets Manager, Azure Key Vault, GCP Secret Manager. They ensure encryption, rotation, and controlled access.
💡 *Tip: Mention compliance (GDPR, SOC2) when talking secrets.*

Q27. Cost optimization in CI/CD pipelines.
Answer: Use caching, auto-scaling runners, on-demand/spot instances, and job pruning.
💡 *Tip: Add “monitor pipeline runtime to cut wasted compute.”*

Q28. Common DevOps challenges in multi-cloud.
Answer: Tool fragmentation, IAM complexity, inconsistent monitoring, and rising costs.
💡 *Tip: Suggest central monitoring + IaC as solutions.*