# AWS Interview Questions and Answers

## 1. What are the differences between EC2, ECS, EKS, and Lambda? In which scenario would you choose one over the other?

EC2 (Elastic Compute Cloud) provides resizable compute capacity in the cloud and is ideal for full control over the operating system and software stack.  
ECS (Elastic Container Service) is a container orchestration service for running Docker containers.  
EKS (Elastic Kubernetes Service) is a managed Kubernetes service for running containerized applications.  
Lambda is a serverless compute service that runs code in response to events and automatically manages the compute resources.  
Choose EC2 for traditional applications, ECS/EKS for containerized workloads, and Lambda for event-driven or short-lived tasks.

## 2. How do you set up high availability and fault tolerance in AWS for a web application?

Use multiple Availability Zones (AZs) for redundancy.  
Deploy load balancers (ELB) to distribute traffic.  
Use Auto Scaling Groups to maintain healthy instances.  
Store data in highly available services like Amazon RDS with Multi-AZ or Amazon S3.  
Implement Route 53 for DNS failover.

## 3. Explain the difference between scaling vertically and horizontally in AWS.

Vertical scaling involves increasing the capacity of a single instance (e.g., upgrading EC2 instance type).  
Horizontal scaling involves adding more instances to distribute the load.  
Horizontal scaling is preferred for fault tolerance and scalability.

## 4. How would you implement a disaster recovery plan in AWS?

Identify critical resources and define RTO and RPO.  
Use backups (e.g., Amazon S3, RDS snapshots).  
Implement cross-region replication.  
Use infrastructure as code (e.g., CloudFormation) for quick recovery.  
Test the recovery process regularly.

## 5. Can you walk through setting up an Auto Scaling Group (ASG) with load balancers?

1. Create a launch template or configuration.  
2. Define the Auto Scaling Group with desired capacity and scaling policies.  
3. Attach the ASG to a load balancer (ALB or ELB).  
4. Configure health checks and monitoring.  
5. Test scaling behavior under load.

## 6. How do IAM roles and policies differ from security groups in AWS?

IAM roles and policies manage permissions for users and services to perform actions on AWS resources.  
Security groups act as virtual firewalls controlling inbound and outbound traffic to resources like EC2.  
IAM is for access control; security groups are for network security.

## 7. What is the difference between S3 Standard, S3 Intelligent-Tiering, and S3 Glacier?

S3 Standard is for frequently accessed data.  
S3 Intelligent-Tiering automatically moves data between access tiers based on usage.  
S3 Glacier is for archival storage with retrieval times ranging from minutes to hours.

## 8. How would you secure sensitive data stored in AWS S3?

Enable encryption (SSE-S3, SSE-KMS).  
Use bucket policies and IAM policies to restrict access.  
Enable logging and monitoring with CloudTrail.  
Use VPC endpoints for private access.  
Enable MFA Delete for extra protection.

## 9. What are VPC Peering and Transit Gateway? When would you use them?

VPC Peering connects two VPCs for private communication.  
Transit Gateway connects multiple VPCs and on-prem networks through a central hub.  
Use peering for simple connections; use Transit Gateway for scalable, hub-and-spoke architecture.

## 10. Explain how you would set up monitoring and alerting using CloudWatch for a production application

1. Create CloudWatch metrics and dashboards.  
2. Set up CloudWatch Alarms for thresholds.  
3. Use CloudWatch Logs for application and system logs.  
4. Integrate with SNS for notifications.  
5. Use CloudWatch Insights for log analysis.

# DevOps Interview Questions and Answers

## 1. What are the key differences between Continuous Integration, Continuous Delivery, and Continuous Deployment?

Continuous Integration (CI) involves automatically building and testing code every time a developer commits changes. Continuous Delivery (CD) ensures that code changes are automatically prepared for release to production. Continuous Deployment goes a step further by automatically deploying every change that passes the automated tests to production without manual intervention.

## 2. How do you ensure zero-downtime deployment in a microservices environment?

Zero-downtime deployment can be achieved using strategies like Blue-Green Deployment, Canary Releases, and rolling updates. These approaches allow new versions of services to be deployed alongside the old ones and gradually shift traffic, ensuring uninterrupted service.

## 3. Can you explain how Infrastructure as Code (IaC) improves reliability in DevOps practices?

IaC allows infrastructure to be defined and managed using code, which improves reliability by enabling version control, repeatability, and automation. Tools like Terraform and AWS CloudFormation help ensure consistent environments across development, testing, and production.

## 4. What are some best practices you follow while writing CI/CD pipelines?

Best practices include keeping pipelines modular, using version control for pipeline definitions, incorporating automated testing and security scans, setting up notifications for failures, and ensuring pipelines are fast and efficient.

## 5. How would you handle secrets management in a DevOps setup?

Secrets should be stored securely using tools like HashiCorp Vault, AWS Secrets Manager, or Kubernetes Secrets. Access should be restricted using IAM policies, and secrets should never be hardcoded in code or configuration files.

## 6. In case of a failed deployment, how would you design a rollback strategy?

Rollback strategies include maintaining previous versions of applications, using Blue-Green deployments to switch traffic back, and implementing automated rollback mechanisms in CI/CD pipelines based on health checks and monitoring.

## 7. What’s the difference between Blue-Green and Canary deployment strategies, and when would you choose one over the other?

Blue-Green deployment involves running two identical environments and switching traffic between them. Canary deployment gradually rolls out changes to a small subset of users before full deployment. Canary is preferred for testing new features with minimal risk, while Blue-Green is ideal for quick rollback.

## 8. How do you monitor and log microservices effectively in production?

Effective monitoring involves using tools like Prometheus, Grafana, and CloudWatch for metrics, and ELK Stack or Fluentd for centralized logging. Distributed tracing tools like Jaeger or Zipkin help track requests across services.

## 9. Explain the importance of container orchestration and why Kubernetes is widely used.

Container orchestration automates deployment, scaling, and management of containerized applications. Kubernetes is widely used due to its robust ecosystem, scalability, self-healing capabilities, and support for declarative configuration.

## 10. How would you optimize a slow CI/CD pipeline in a large-scale project?

Optimization strategies include parallelizing jobs, caching dependencies, using incremental builds, reducing unnecessary steps, and monitoring pipeline performance to identify bottlenecks.

1. What will you do if a Jenkins pipeline fails?  
   Answer: Check Jenkins logs → Identify stage of failure → Fix configuration/code issue → Re-run the pipeline. If infra-related, verify Terraform or Kubernetes changes before redeploying.
2. 2. How do you handle a failed deployment in Kubernetes?  
   Answer: Use kubectl describe pod and kubectl logs to check errors → If critical, rollback with kubectl rollout undo deployment <name> → Fix and redeploy.
3. 3. How do you manage Terraform state in a team?  
   Answer: Store state in remote backend (Azure Blob, GCS bucket, Terraform Cloud) → Enable state locking to prevent conflicts → Use versioning for recovery.
4. 4. You changed a Terraform variable and want to see the impact before applying. What do you do?  
   Answer: Run terraform plan to preview changes before applying.
5. 5. How do you ensure zero downtime deployment in Kubernetes?  
   Answer: Use RollingUpdate strategy in deployments, configure readiness probes, and keep replicas running until new pods are healthy.
6. 6. How do you roll back in Jenkins if a deployment causes issues?  
   Answer: Keep artifact versioning → Redeploy the last stable build from Jenkins → Or trigger rollback pipeline.  
   7. What if Terraform state file gets corrupted or deleted?  
   Answer: Restore from remote backend version history (e.g., GCS versioning, Azure Blob snapshots) → If not available, use terraform import to rebuild state.
7. 8. How do you secure secrets in pipelines?  
   Answer: Use Jenkins credentials manager, Vault, or cloud secret managers (GCP Secret Manager, Azure Key Vault) instead of storing secrets in code.
8. 9. How do you monitor Kubernetes clusters?  
   Answer: Use Prometheus + Grafana for metrics, ELK/EFK stack for logs, and Kubernetes liveness/readiness probes for pod health.
9. 10. What will you do if a pod is stuck in CrashLoopBackOff?  
   Answer: Run kubectl describe pod and kubectl logs → Check startup script, image, or config issue → Fix error → Redeploy.
10. 11. How do you manage infrastructure across multiple environments (Dev/QA/Prod) using Terraform?  
    Answer: Use workspaces or separate state files with environment-specific variables.
11. 12. How do you optimize CI/CD pipelines in Jenkins?  
    Answer: Use parallel stages, caching (e.g., Docker layers, Maven cache), and parameterized builds to save time.
12. 13. How do you perform blue-green deployment in Kubernetes?  
    Answer: Run two environments (Blue = current, Green = new) → Route traffic to Green only after successful validation → Rollback to Blue if issues occur.
13. 14. How do you troubleshoot high CPU usage on a Linux server?  
    Answer: Use top, htop, vmstat, and iostat → Identify process → Kill/fix process → Scale infra if required.
14. 15. How do you automate infrastructure scaling in cloud?  
    Answer: Configure Auto Scaling Groups in GCP (Instance Groups) or Azure (VM Scale Sets) → Integrate with Terraform for automation
15. What if a Terraform apply fails halfway?  
    Answer: Terraform creates a partial state → Fix the error → Run terraform plan again → Apply → Or use terraform refresh to sync state.  
      
      
    How do you manage multi-cloud deployments (GCP + Azure)?  
    Answer: Use Terraform with multiple providers → Create modules for each cloud → Keep separate state files for GCP and Azure.  
      
      
    How do you handle a disk full issue in Linux?  
    Answer: Run df -h to check usage → Clear logs from /var/log → Remove unused Docker images/containers → Expand disk if required.  
      
      
    How do you implement rollback in Terraform?  
    Answer: Use version control to revert .tf files → Run terraform apply again → Restore previous infra state from remote backend version.  
      
      
    How do you ensure security in DevOps pipelines?  
    Answer:  
    Scan code with SonarQube  
    Scan images with Trivy/Anchore  
    Use IAM least privilege in GCP/Azure  
    Store secrets in Secret Manager/Key Vault  
    Enable audit logging
16. How do you manage Terraform modules for reusability?  
    Answer: Create modules for common components (VPC, VM, AKS/GKE, IAM) → Store them in Git → Call modules in different projects with version control.  
      
      
    What if two team members run Terraform apply at the same time?  
    Answer: Use remote backend with state locking (like Azure Blob with locking or GCS with locking) → This prevents conflicts.  
      
      
    How do you perform Canary Deployment in Kubernetes?  
    Answer: Deploy a new version to a small % of users → Use Istio/NGINX Ingress for traffic routing → Gradually increase traffic → Rollback if errors.  
      
      
    How do you troubleshoot “ImagePullBackOff” in Kubernetes?  
    Answer:  
    Check if image exists in registry.  
    Validate credentials/secret for private registry.  
    Verify image tag.  
    Fix and redeploy.  
      
      
    How do you integrate Jenkins with GitHub?  
    Answer: Configure GitHub webhook → Connect Jenkins job to repo → Trigger builds automatically on code push/PR.  
      
      
    How do you perform zero-downtime DB migration in CI/CD?  
    Answer: Use Liquibase/Flyway migration scripts → Apply backward-compatible schema changes → Deploy app → Apply destructive changes only later.  
      
      
    What if a Jenkins agent node goes offline?  
    Answer: Check agent logs → Restart service → Verify connectivity with master → Add auto-scaling slaves (Kubernetes or cloud VMs).  
      
      
    How do you set resource limits in Kubernetes?  
    Answer: Define requests & limits in pod spec → Ensures fair resource allocation and prevents pod from consuming all CPU/memory.  
      
      
    How do you implement CI/CD for Terraform?  
    Answer:  
    Use Jenkins/GitHub Actions → terraform fmt, terraform validate, terraform plan, approval stage, then terraform apply.  
    Store state in backend.  
      
      
    How do you monitor logs in Kubernetes?  
    Answer: Use kubectl logs for quick debugging → For centralized logging, use EFK (Elasticsearch + Fluentd + Kibana) or Loki + Grafana.