#_ Interview Preparation for DevOps Engineers Checklist

1. 🛠️ Infrastructure as Code (IaC):

- **Understanding IaC:**
  - Importance of IaC in automating infrastructure provisioning.
  - Benefits of declarative vs. imperative IaC approaches.
  - Tools like Terraform, CloudFormation, or Ansible for IaC.
- **Terraform Deep Dive:**
  - Writing and organizing Terraform configuration files.
  - Resource provisioning and management using HCL.
  - Terraform modules and remote state management.
- **Configuration Management (Optional):**
  - Introduction to tools like Ansible, Puppet, or Chef.
  - Automating server configuration and management.

Resources:

- Terraform Up & Running
- Ansible Documentation

2. ☁️ Cloud Services:

- **Cloud Providers:**
  - Understanding key services of major cloud providers (AWS, Azure, GCP).
  - Region, availability zones, and global infrastructure concepts.
- **AWS Services (Optional):**
  - In-depth knowledge of services like EC2, S3, RDS, etc.
  - IAM for access management and security.

Resources:

- AWS Documentation
- Azure Documentation
- GCP Documentation

By: Waleed Mousa

3. 📦 Containerization:

- **Docker Fundamentals:**
  - What containers are and their benefits in application deployment.
  - Docker commands for building, running, and managing containers.
  - Docker Compose for multi-container applications.
- **Orchestration Tools (Optional):**
  - Kubernetes for container orchestration.
  - Understanding pods, deployments, services, etc.

Resources:

- Docker Documentation
- Kubernetes Documentation

4. 🔄 Continuous Integration and Continuous Deployment (CI/CD):

- **CI/CD Pipelines:**
  - Designing and implementing automated CI/CD pipelines.
  - Version control integration (e.g., GitHub, GitLab, Bitbucket).
  - Continuous testing, integration, and deployment steps.
- **Jenkins (Optional):**
  - Setting up Jenkins pipelines and jobs.
  - Plugin integration and extensibility.

Resources:

- Jenkins Documentation

5. 🔐 Security and Compliance:

- **Security Best Practices:**
  - Understanding network security, firewalls, and encryption.
  - Implementing security groups, IAM roles, and policies.
  - Compliance standards like GDPR and HIPAA.
- **Infrastructure Security (Optional):**

    - Tools like Vault for secrets management.
    - Implementing secure access and authentication.

Resources:

- AWS Security Best Practices
- HashiCorp Vault Documentation

6. 📊 Monitoring and Logging:

- **Monitoring Solutions:**
    - Implementing monitoring using tools like Prometheus or Nagios.
    - Setting up alerts and notifications for incidents.
    - Dashboards and visualization for metrics.
- **Logging and Tracing (Optional):**
    - Centralized logging using tools like ELK stack or Splunk.
    - Distributed tracing for microservices.

Resources:

- Prometheus Documentation
- Elasticsearch, Logstash, Kibana (ELK) Stack

7. 🛡️ DevOps Culture and Collaboration:

- **DevOps Principles:**
    - Understanding the principles of collaboration, automation, measurement, and sharing.
    - Role of DevOps in bridging development and operations teams.
- **Agile and DevOps (Optional):**
    - Integrating Agile methodologies with DevOps practices.
    - CI/CD in Agile sprints.

Resources:

- The Phoenix Project
- DevOps Institute

8. 🧪 Test Automation and Infrastructure Testing:

- **Automated Testing:**
  - Importance of automated testing in CI/CD pipelines.
  - Types of testing (unit, integration, end-to-end) and their tools.
- **Infrastructure Testing (Optional):**
  - Testing IaC using tools like Terratest.
  - Validating infrastructure changes before deployment.

Resources:

- Test Automation University

9. 🛢️ Database Management and Automation:

- **Database Deployment:**
  - Automating database provisioning using IaC.
  - Managing database schema changes and migrations.
- **Database Tools (Optional):**
  - Tools like Liquibase or Flyway for database version control.
  - Monitoring and optimizing database performance.

Resources:

- Database Version Control with Liquibase

10. 🏎️ Performance Optimization and Scalability:

- **Scaling Strategies:**
  - Horizontal vs. vertical scaling considerations.
  - Load balancing and auto-scaling techniques.
- **Performance Tuning (Optional):**
  - Optimizing application and infrastructure for high traffic.
  - CDN and caching strategies.

Resources:

- High-Performance Web Sites

By: Waleed Mousa