here we are providing AWS devops engineer roadmap and tools information and interview skills and sample resume

1.ROAD MAP
2.TOOLS INFORMATION
3.INTERVIEW PREPARATION
4.RESUME

DevOps is About:
Designing systems that ensure smooth deployments
Scaling infrastructure effortlessly
Maintaining reliable operations
Detecting issues early
Enabling seamless team collaboration

Tools are Just Enablers
They assist in achieving these goals, but they don't define DevOps.

True DevOps Focuses On:
Smart automation
Proactive monitoring
Efficient scaling
Strong security
Continuous improvement

First we are looking ROad map

1.How software development works
2.operating system LINUX basics and admin level
3.Cloud AWS
4.version control system git and github
5.containerization Docker
6.CI/CD pipeline Jenkins
7.Container orchestration Kubernetes
8.Scripting Language
9.Infrastructure as a code Terraform
10.avoid provisioning and configuration management ansible.
11.Monitoring tools promothesus and grafana

1.NOTE-:
-->Since DevOps covers the whole software development lifecycle, it means you work with lots of technologies. Plus DevOps is still evolving and there are lots of new tools being developed all the time.
So you have to be comfortable with constantly learning and expanding your knowledge, even after you have become a DevOps engineer

NOTE-:
As a DevOps engineer you will not be

programming the application, but as you are working closely with the development team to improve and automate tasks for them, you need to understand the concepts.

-->You have to understand what the whole software development lifecycle covers from idea to code, all the way to releasing it to the end users!

-->How developers work and collaborate

-->How applications are configured (Build & packaging Tools)

-->Automated testing and test scopes

2.Operating system Linux Concepts
-->basic commands
-->shell commands
-->ssh key management
-->linux file system
-->permissions
-->Configure Firewalls to secure access
-->Understand how IP addresses, ports and DNS works
-->Load Balancers
-->Proxies
-->HTTP/HTTPS

3.Cloud AWS
-->first we have to understand virtual server works
-->EC2
-->IAM
-->S3
-->RDS
-->ECS
-->EKS
-->ECR
-->CLOUD FRONT
-->CLOUD WATCH
-->CLOUD FORMATION
-->CODE PIPELINE
-->CODE COMMIT
-->VPC
-->ROUTE53
-->AWS LAMBDA
-->AWS CLODTRAIL
-->AWS ELASTIC BEANSTALK....etc

4.version control system git and github
NOTE-:
Git is a CLI Tool, which you install locally. It enables the tracking of changes in the source code and enables better collaboration on code.

YOU HAVE TO LEARN

-->Your files are stored centrally in a remote Git repository on the web. Most popular Git repositories are GitHub and GitLab.

-->GITHUB WORKFLOWS

-->The core Git commands, like git clone, git branch, git pull/push, git merge etc But also how to collaborate on a project, like create pull requests, code reviews, branching

-->continuous integration

5.Containerzation Docker

NOTE-:As containers have become the new standard of software packaging, you will most probably run your application as a container

you need to understand:

-->concepts of virtualization
-->concepts of containerization
-->how to manage containerized applications on a server.

you have to learn:
-->Run containers
-->Inspect active containers
-->Docker Networking
-->Persist data with Docker Volumes
-->Dockerize apps using Dockerfiles
-->Run multiple containers using
-->Docker-Compose
-->Docker Repository
-->Docker hub
-->ECS AND ECR

6.CI/CD pipeline Jenkins

-->CI/CD(continuous integration and continuous delivery and continuous deployment) is the main tool in DevOps.
-->In DevOps, all code changes, like new features or bug fixes, need to be integrated in the existing application and deployed for the end user continuously and in an automated way

You have to learn:
-->Setting up the CI/CD server
-->Integrate code repository to trigger pipeline automatically
-->Build Tools & Package Manager Tools to execute the tests and package the application
-->Configuring artifact repositories (like Nexus) and integrate with pipeline

7.Container orchestration Kubernetes
NOTE:
Container orchestration tools like Kubernetes, automate the deployment, scaling and management of containerized applications.

You have to learn

-->How Kubernetes works
-->How to administer and manage the K8s cluster
-->How to deploy applications on K8s
-->Learn core components like, Deployment, Service, ConfigMap, Secret, StatefulSet, Ingress
-->Kubernetes CLI (Kubectl)
-->Persisting data with K8s Volumes Namespaces

8.SCRIPTING LANGUAGE
-->python
-->shell scripting

9.Infrastructure as a code Terraform
NOTE:
Manually creating and maintaining infrastructure is time consuming and error prone. Especially when you need to replicate the infrastructure, e.g. for a Development, Testing and Production environment.
In DevOps, we want to automate as much as possible and that’s where Infrastructure as Code comes into the software

you have to learn Teraform concepts

11.Monitoring tools promothesus and grafana
-->Once software is in production, it is important to monitor it to track the performance, discover problems in your infrastructure and the application.

-->So one of your responsibilities as a DevOps engineer is to
*setup software monitoring
*setup infrastructure monitoring, e.g. for your Kubernetes cluster and underlying servers
*visualize the data

You have to learn:
-->Prometheus:
A popular monitoring and alerting tool
-->Grafana:
Analytics and interactive visualization tool
-->ELK Stack:
A popular log management stack