Q1: Tell me about yourself.

My name is **Mohd Shahid**, and I have over **7.5 years of experience in IT**. I'm originally from **Delhi, India**, and I hold a **Bachelor's degree from Faizabad University**. I've also completed certifications like **AWS Solutions Architect – Associate** and **Microsoft Azure Administrator**.

I started my career in **2017 as an IT Support Technician**, then worked as a **System Administrator** at **Delhivery** and **Motherson-Sumi**, managing servers and networks. I later joined **Genpact** as a **Desktop Support Engineer**, where I got the opportunity to move into the **Cloud domain**.

Currently, I'm working as a **Cloud Specialist at Genpact**, where I manage **AWS infrastructure**, build **CI/CD pipelines**, automate using **Terraform and Ansible**, and work with **Kubernetes** for container orchestration. I'm passionate about automation and improving system performance.

Q2: What are your roles and responsibilities? (Interview-style answer)

In my current role as a Cloud Specialist at Genpact, I'm responsible for:

- Designing, deploying, and managing cloud infrastructure on AWS using best practices.
- Implementing CI/CD pipelines using Jenkins, GitLab CI, and AWS CodePipeline to automate deployments and improve delivery speed.
- Using Infrastructure as Code with Terraform to provision and manage cloud resources.
- Deploying and managing Kubernetes (EKS) clusters for containerized workloads.
- Monitoring system health and performance through CloudWatch and Prometheus.
- Participating in migration projects to transition legacy systems to AWS.
- Collaborating with development teams to ensure DevOps best practices are followed across the SDLC.

Q3: What do you know about DevOps? Which components have you worked on?

DevOps is a cultural and technical approach that aims to bridge the gap between development and operations through automation, collaboration, and continuous feedback.

Components I’ve worked on include:

- **CI/CD**: Jenkins, GitLab CI, AWS CodePipeline
- **Containerization**: Docker, Kubernetes (EKS)
- **IaC (Infrastructure as Code)**: Terraform, Ansible
- **Cloud Platforms**: AWS (EC2, S3, VPC, IAM, CloudWatch), Azure
- **Version Control**: Git, GitHub, GitLab
- **Monitoring & Logging**: Prometheus, ELK Stack, CloudWatch
- **Scripting**: Bash and Python for automation

I’ve used these tools to implement scalable infrastructure, automate deployments, and improve reliability across cloud environments.

Q4: I want to create and checkout a branch at the same time.

You can use the following Git command:

```
git checkout -b <branch-name>
```

This command creates a new branch and switches to it immediately. For example:

```
git checkout -b feature/login-api
```

Q5: What is Git Rebase?

git rebase is a Git command used to integrate changes from one branch into another in a linear and clean way. Instead of creating a merge commit like git merge, rebase rewrites the commit history by placing your changes on top of another branch.

Use case:
If you’re on a feature branch and want to apply your changes on top of the latest main branch, you can use:

```
git checkout feature-branch

git rebase main
```

This keeps the history clean and avoids unnecessary merge commits.

Q6: What is Git Fetch? Why not Git Pull?

git fetch downloads changes from the remote repository but doesn’t merge them into your working branch. It’s useful for reviewing updates before applying them.

```
git fetch origin
```

On the other hand, git pull is a combination of fetch and merge. It directly applies the changes, which may cause conflicts if not reviewed.

Why use git fetch over git pull?
Using git fetch gives you more control and avoids unwanted merge conflicts. It allows you to inspect what's changed before merging:

```
git fetch

git diff origin/main

git merge origin/main
```

Q7: What is Task Definition in ECS?

In Amazon ECS (Elastic Container Service), a *Task Definition* is like a blueprint for running containers. It defines:

- The Docker image to use.
- CPU and memory requirements.
- Port mappings.
- Environment variables.
- IAM roles.
- Logging configuration.

Each ECS task or service uses a task definition to understand how to run the container(s). It’s version-controlled, so you can roll back to earlier configurations.