Azure DevOps & CI/CD Interview Questions and Answers

1. How do you implement approval-based deployments in Azure Pipelines using Environments?

Azure Pipelines provide the concept of "Environments" where you can configure checks such as manual approvals before deploying to that environment.

Example: In a financial services application, we used environments for Dev, QA, UAT, and Prod. A stage in the pipeline deployed to the Prod environment, but it paused for manual approval from both the QA lead and Release Manager using environment checks before proceeding.

2. What's the purpose of pipeline artifacts and how are they different from build artifacts?

- Pipeline Artifacts are used to pass files between jobs or stages in a pipeline.
- Build Artifacts are outputs of a build that are stored and can be downloaded or deployed later.

Example: In our CI pipeline, compiled .jar files were stored as pipeline artifacts for integration tests. The same files were later published as build artifacts for deployment and rollback purposes.

3. How do you integrate Azure Repos with external Git providers like GitHub?

Azure DevOps supports GitHub integration using service connections. This allows pipelines to trigger builds on GitHub commits or pull requests.

Example: Our frontend code was hosted in GitHub, while backend was in Azure Repos. We used a GitHub service connection and set CI triggers for pull requests on main, enabling shared CI/CD pipelines across both repositories.

4. How do you use pipeline templates to enforce consistency across teams?

Pipeline templates allow reuse of YAML logic like build, test, and deployment steps.

Example: A centralized DevOps team created a template.yml for all microservices that included code scan, build, unit tests, and publishing artifacts. Every service team consumed this template

ensuring compliance and faster onboarding.

Azure Services & Integration

1. What is the difference between Azure Service Bus and Event Grid?

- Service Bus: Message broker, supports ordered delivery and retries (pull-based).
- Event Grid: Event notification system, used for lightweight, reactive patterns (push-based).

Example: For our order processing system, Service Bus managed order and payment service communication. For storage blob uploads, Event Grid was used to trigger an Azure Function that updated metadata in Cosmos DB.

2. How do you deploy applications to Azure Kubernetes Service (AKS) using Azure Pipelines?

Azure Pipelines can deploy to AKS using built-in kubectl, Helm, or azureCLI tasks.

Example: We containerized a .NET app, built the Docker image in Azure DevOps, pushed it to ACR, and deployed to AKS using a Helm chart in a deployment stage. Values files varied by environment.

3. How do you monitor Azure Functions for failures and performance?

Use Application Insights and Azure Monitor.

Example: A serverless function triggered by HTTP POST was instrumented with Application Insights. Logs showed cold start delay during early mornings. We added pre-warming and increased plan size to improve performance.

4. What is the role of Azure Front Door in DevOps workloads?

Azure Front Door offers global HTTP load balancing, HTTPS offload, WAF, and smart routing.

Example: For a globally used e-commerce app, Azure Front Door routed traffic to AKS deployments in East US and West Europe. It ensured minimal latency and failover during regional outages.

Infrastructure as Code (Terraform / Bicep)

1. How do you implement environment-specific configurations using Terraform?

Use multiple variable files or Terraform workspaces.

Example: We had dev.tfvars, qa.tfvars, and prod.tfvars. The pipeline dynamically selected the correct file based on the build stage and passed it to the terraform apply command.

2. How do you validate and test Terraform code before applying?

Use terraform validate, terraform plan, and tools like tflint, terratest.

Example: In our CI pipeline:

- terraform init
- terraform validate
- terraform plan
- Approval stage
- terraform apply

We also used Terratest to simulate resource creation and test outputs using Go.

3. What are Bicep modules and how do they improve reusability in IaC?

Modules are reusable .bicep files that encapsulate resource definitions.

Example: We created a vnet.bicep module defining VNet and subnets. It was reused across projects by just passing parameters like CIDR and tags, ensuring consistency and saving effort.

4. How do you integrate Terraform deployment into an Azure Pipeline securely?

Use service principals, secure variable groups, and Azure Key Vault.

Example: We stored the SP credentials in Key Vault. Azure Pipelines accessed them using AzureKeyVault@2. Terraform backend was Azure Storage with state locking via a shared key or SAS token.

Git & Repository Management

1. What's the difference between GitFlow and trunk-based development?

- GitFlow: Uses multiple branches for features, releases, hotfixes. Ideal for planned releases.

- Trunk-Based: Uses main as the single integration point. Encourages CI/CD.

Example: Our legacy ERP used GitFlow for monthly releases. A new ML-based recommendation engine adopted trunk-based development with short-lived branches and feature toggles, enabling rapid deployment and feedback.