Azure DevOps - Full Audit Q&A

1. What is your rollback strategy if deployment fails?

Azure App Service supports deployment slots and version history. If deployment fails, we can quickly swap back to the previous version.

We also retain previous artifacts in Azure DevOps, allowing redeployment of any successful build.

Example:

- Use staging slot for safe deployment. If it fails, don't swap.
- Or use Azure DevOps to redeploy a previous successful build.

In simple terms: We ensure quick rollback using either swap mechanism or redeploying last known good version.

2. How do you handle environment-specific configurations?

We use variable groups and environment-specific parameter files in Azure DevOps.

For secure secrets, we use Azure Key Vault integrated with the pipeline.

Example:

- Dev environment uses 'dev.database.windows.net'
- Prod uses 'prod.database.windows.net'
- These values are injected using variable groups
- Sensitive values are fetched securely from Azure Key Vault

In simple terms: Each environment has its own config set. Secrets are stored in Key Vault for security.

3. How do you ensure secure access to your source code and pipelines?

We use GitHub with branch protection rules, mandatory reviews, and GitHub Advanced Security (GHAS).

Azure DevOps is integrated with Azure AD for RBAC and MFA.

Secrets are stored in Azure Key Vault and never hardcoded in pipelines.

4. How is monitoring and alerting handled post-deployment?

We use Azure Monitor and Application Insights for performance and availability monitoring.

Alerts are configured for key metrics such as CPU, memory, HTTP errors, etc.

Teams are notified via Email or Teams integration in case of failure.

5. How do you manage infrastructure as code (IaC)?

We use ARM/Bicep templates or Terraform to define and deploy infrastructure.

These scripts are versioned in GitHub and deployed via Azure DevOps pipelines.

Each environment has its own parameter file for consistency.

6. How do you implement quality checks during the CI process?

CI pipelines include static code analysis, unit tests, and code coverage reports.

Pull Requests require approvals and successful build validation before merge.

7. How do you handle secrets and sensitive values in pipelines?

Secrets are stored in Azure Key Vault.

Azure DevOps accesses these values securely using linked service connections and variable groups.

8. What tools do you use for DevOps automation and orchestration?

We use Azure DevOps for end-to-end CI/CD, GitHub for SCM and security, Terraform/Bicep for IaC, and Azure CLI/PowerShell for automation scripting.

9. How is deployment approval managed across environments?

We configure pre-deployment and post-deployment approval gates in Azure DevOps release pipelines.

Only authorized users or groups can approve promotion to next environments like staging or production.

10. How do you ensure traceability and auditability in DevOps?

GitHub provides full history of commits, pull requests, and reviews.

Azure DevOps retains detailed pipeline logs, build artifacts, and deployment history.

We also tag each deployment with build number and metadata for traceability.