3. Build and release pipelines
└ **3.1 Package management and testing strategy**
└ **3.1.1 Quality/release gates (security, governance)**

1. What are quality gates and release gates in Azure Pipelines?
2. How do you configure a quality gate in a pipeline?
3. What types of policies or checks are commonly used as gates?
4. How do you implement security scanning (e.g., SCA, SAST) as a pipeline gate?
5. How do you enforce governance and compliance requirements in a release pipeline?
6. What is an approval gate and how is it set up?
7. How can you integrate external tools (e.g., SonarCloud, WhiteSource, Checkmarx) as gates?
8. What happens when a gate fails during a release?
9. How do you audit and report on gate results for compliance?
10. What are best practices for designing effective release and quality gates?

1. What are quality gates and release gates in Azure Pipelines?
Quality gates and release gates are automated checks within *Azure Pipelines* that block progression of builds or releases until specified conditions are met.

- Quality gates run during the build (e.g., code coverage, test pass rate),
- while release gates execute before or after deployment stages (e.g., approvals, external service checks).

2. How do you configure a quality gate in a pipeline?
Define checks or conditions (e.g., code coverage threshold, static analysis results) in the pipeline YAML or classic editor. In YAML, use the `condition` attribute for steps, or specify required checks in branch policies for PR validation.

3. What types of policies or checks are commonly used as gates?
Common gates include:

- Code coverage thresholds
- Static code analysis (SAST)
- Open-source dependency scanning (SCA)
- Linting
- Required manual approvals
- Work item linking
- Deployment health checks
- External service integrations (e.g., SonarCloud)

4. How do you implement security scanning (e.g., SCA, SAST) as a pipeline gate?
Add tasks for security scanning tools (e.g., SonarCloudPrepare, Trivy, OWASP Dependency-Check) in the pipeline. Configure the tool to fail the build or block the release if vulnerabilities or policy violations are detected.

5. How do you enforce governance and compliance requirements in a release pipeline?

- Use environment checks and approvals,
- integrate mandatory gate tasks (e.g., approval steps, sign-off requirements),
- and ensure artifacts and deployments meet compliance criteria before advancing to production.
- Configure auditing and tracking via Azure Pipelines environments and release history.

---

6. What is an approval gate and how is it set up?

An approval gate requires one or more authorized users to approve deployment before the pipeline can proceed. Configure this in the Azure Pipeline environment or stage by adding manual approval checks and specifying approvers in the pipeline settings.

---

7. How can you integrate external tools (e.g., SonarCloud, WhiteSource, Checkmarx) as gates?

Add *marketplace* tasks or custom scripts for the tool to your pipeline. Use the tool's exit codes or results to determine pass/fail status. For gated releases, configure an environment check using the tool's REST API or output.

---

8. What happens when a gate fails during a release?

If a gate fails, the pipeline halts at that stage. No further deployments occur until the issue is resolved and the gate is re-evaluated or manually overridden (if permitted).

---

9. How do you audit and report on gate results for compliance?

Gate results are logged in the pipeline run history and environment checks. Use

- Azure DevOps audit logs,
- downloadable run logs,
- and built-in reporting features

to demonstrate compliance during audits.

---

10. What are best practices for designing effective release and quality gates?

Use automated gates for security, quality, and compliance.

- Avoid overloading pipelines with too many manual approvals.
- Regularly review and update gate criteria to match evolving requirements.
- Integrate external scanning tools natively.
- Ensure gate failures produce clear, actionable feedback.