3. Build and release pipelines

└ **3.5 Implement deployment solutions**

└ **3.5.1 Feature Flags with Azure App Configuration**

---

1. What are feature flags and why are they used in DevOps?
2. What is Azure App Configuration and how does it support feature management?
3. How do you create and manage feature flags in Azure App Configuration?
4. How do you enable or disable a feature flag at runtime without redeploying code?
5. What is the Feature Manager library and how is it used in .NET applications?
6. How can feature flags be targeted for specific users or environments?
7. How do you implement feature flag evaluation in Azure DevOps pipelines?
8. What are best practices for rolling out and rolling back features using flags?
9. How can feature flag usage be audited and monitored?
10. What are the risks or anti-patterns when using feature flags in production?

---

1. **What are feature flags and why are they used in DevOps?**
Feature flags are switches in application code to enable or disable features at runtime. They allow teams to release code to production safely, test in production, perform gradual rollouts, and roll back features instantly without redeployment.

---

2. **What is Azure App Configuration and how does it support feature management?**
Azure App Configuration is a centralized service for application settings and feature flags. Its Feature Management capability allows dynamic creation, updating, and targeting of feature flags for apps at scale, decoupling feature rollout from code releases.

---

3. **How do you create and manage feature flags in Azure App Configuration?**
Feature flags are created and managed in the *Azure* portal under *App Configuration > Feature Manager*. Flags can be enabled, disabled, or targeted, and settings are available via client libraries or REST API.

---

4. **How do you enable or disable a feature flag at runtime without redeploying code?**
Change the flag's state in *Azure App Configuration*. The application, using the *Feature Management SDK*, will check for flag changes at runtime (polling or refresh interval) and update behavior instantly, without requiring a redeployment.

---

5. **What is the Feature Manager library and how is it used in .NET applications?**
The *Microsoft.FeatureManagement* library is a .NET SDK that integrates with *Azure App Configuration*. Add and configure it in your app to query and evaluate feature flags dynamically using dependency injection, middleware, or attributes.

---

6. **How can feature flags be targeted for specific users or environments?**
Azure App Configuration supports filters (e.g., user targeting, time windows, percentage rollout). Configure these filters in *Feature Manager* to enable features for selected users, groups, or environments based on context.

---

7. **How do you implement feature flag evaluation in Azure DevOps pipelines?**
Use the *Azure App Configuration* task in *Azure Pipelines* to retrieve flag values as pipeline variables. This allows pipeline stages, jobs, or tasks to conditionally execute based on the flag state in *App Configuration*.

8. **What are best practices for rolling out and rolling back features using flags?**

- Use gradual exposure (percentage rollout, user targeting),
- monitor impact,
- and have clear rollback procedures.
- Remove obsolete flags quickly to avoid technical debt and keep the configuration clean.

---

9. **How can feature flag usage be audited and monitored?**

Audit changes using *Azure Activity Logs* and *App Configuration* revision history. Monitor feature usage and application impact using *Application Insights* or custom telemetry tied to flag evaluation points.

---

11. **What are the risks or anti-patterns when using feature flags in production?**

Common risks include

- flag overload,
- leaving dead flags,
- inconsistent flag states across services,
- and insufficient monitoring.

These can lead to increased complexity, technical debt, and production issues if not managed properly.