Corporate Git Branching Strategies

By DevOps Shack

Branching strategies in software development are essential for managing code changes efficiently and ensuring a smooth workflow across different environments. Below, I'll explain three common branching strategies: Gitflow, GitHub Flow, and GitLab Flow, in the context of development (dev), quality assurance (QA), pre-production (PPD), production (Prod), and disaster recovery (DR) environments. I'll also cover how these strategies handle feature branches, bugfix branches, and hotfix branches.

1. Gitflow:

Environments:

- **Development (dev):** Daily work and ongoing development.
- **Quality Assurance (QA):** Integration testing and quality assurance.
- **Pre-Production (PPD):** Staging environment for final testing.
- **Production (Prod):** Live production environment.
- **Disaster Recovery (DR):** Backups and recovery procedures.

Branches:

- **Feature branches:**
  - Created from: `develop`
  - Merged into: `develop`
  - Purpose: Isolate new features for development and testing.
- **Bugfix branches:**
  - Created from: `develop`
  - Merged into: `develop` and `master`
  - Purpose: Fix bugs in the development and production branches.
- **Hotfix branches:**
  - Created from: `master`
  - Merged into: `develop` and `master`
  - Purpose: Quickly address critical issues in the production environment.

2. GitHub Flow:

Environments:

- **Development (dev):** Continuous deployment environment.

Branches:

- **Feature branches:**
  - Created from: `main` (or `master`)
  - Merged into: `main` (or `master`)
  - Purpose: Isolate new features for development and testing.
- **Bugfix branches:**
  - Created from: `main` (or `master`)
  - Merged into: `main` (or `master`)
  - Purpose: Fix bugs in the development and production branches.
- **Hotfix branches:**
  - Created from: `main` (or `master`)
  - Merged into: `main` (or `master`)
  - Purpose: Quickly address critical issues in the production environment.

3. GitLab Flow:

Environments:

- **Development (dev):** Continuous integration and delivery environment.

Branches:

- **Feature branches:**
  - Created from: `main`
  - Merged into: `main`
  - Purpose: Isolate new features for development and testing.
- **Bugfix branches:**
  - Created from: `main`
  - Merged into: `main`
  - Purpose: Fix bugs in the development and production branches.
- **Hotfix branches:**
  - Created from: `main`

    - Merged into: `main`
    - Purpose: Quickly address critical issues in the production environment.

General Notes:

- **Feature Development:**
    - Gitflow tends to isolate feature development more explicitly, making it suitable for larger projects.
    - GitHub Flow and GitLab Flow promote a simpler workflow, emphasizing continuous delivery.
- **Release Management:**
    - Gitflow has dedicated branches for releases and supports complex release management.
    - GitHub Flow and GitLab Flow release directly from the main branch.

Each branching strategy has its strengths and weaknesses, and the choice depends on the project's size, complexity, and team preferences. Additionally, tools like Git, GitHub, and GitLab provide flexibility for teams to adapt and customize their branching strategies as needed.