

Continuous Integration and Continuous Deployment (CI/CD)

What is CI/CD?

- CI = Continuous Integration
 - DevOps best practice: regularly merge changes into a repository and then run builds and tests automatically.
- CD = either Continuous Delivery or Continuous Deployment
- Continuous Delivery
 - produce software in short, frequent cycles
 - You can release the software at any time
 - Process of deployment is simple and repeatable (though still manual)
- Continuous Deployment
 - Goes one step further than Cont. Delivery
 - Every change that passes all stages of production is released automatically to the customers.
 - You can release faster and releases are less risky because the changes are smaller
 - But you need a great testing culture in your company

GitHub Actions for CI/CD

- GitHub Actions are a CI/CD tool integrated into GitHub.
- Features: workflows, jobs, steps, and secrets.

Git Branching Strategies

- Branching in Git lets developers work on features, fix bugs, or safely experiment with new ideas in isolated environments within the same repository.
- Common strategies: Feature branches, Gitflow, and GitHub flow.

Feature Branches

- Feature branches are temporary branches created from the main code line for developing new features or fixes.
- Once the feature is completed and tested, it gets merged back into the main branch, often after a code review.
- This strategy keeps the main branch free from unstable code and makes it easier to manage multiple features in development.
- It's crucial to frequently sync these branches with the main branch to avoid complex merges later on.

Gitflow Workflow

- Gitflow is a more structured branching strategy, extending the feature branch model with specific roles for different branches.
- Five types of branches in Gitflow: feature, develop, release, hotfix, and master.
- Develop and feature branches are for development, release branches for preparing a release, hotfix branches for quick fixes, and master for stable releases.
- The flow supports a robust release cycle, with development and releases progressing in parallel.
- Gitflow's explicit structure is excellent for managing complex projects but might be too rigid for smaller teams or projects.

GitHub Flow

- GitHub Flow is a lightweight, branch-based workflow that supports teams working on and deploying regularly.
- In GitHub Flow, anything in the main branch is deployable, and new features are created in branches, submitted via pull requests, and then merged after review.
- With GitHub Flow, deployments are frequent, and the process is straightforward, aligning with continuous delivery.

Tips for Choosing a Strategy

- Smaller projects with fewer developers might prefer the simplicity of feature branches and/or GitHub workflow.
- For teams with frequent releases and hotfixes, Gitflow offers a more controlled workflow.
- Consider how your team collaborates and the level of Git expertise when choosing a strategy.
- Your choice of CI/CD tools and project management software can also influence the optimal branching strategy.

Summary and Takeaways

- CI/CD is essential for modern software development for rapid, reliable releases.
- GitHub Actions provides a convenient way to implement CI/CD right from your repository.
- An effective Git branching strategy can smoothly integrate with your CI/CD pipeline.
- Finally, I covered practical tips for making sure your app is deployed securely.