

# git/GitHub for Developers

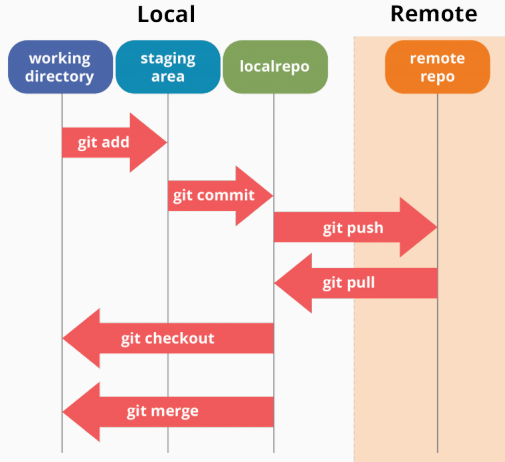
---

February 15, 2024

Engineers for Exploration, UC San Diego

# Introduction

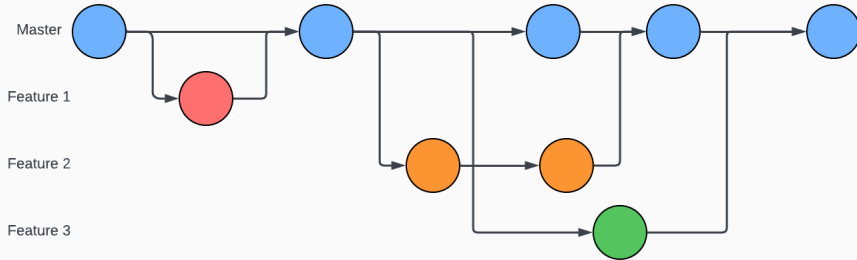
- **Git vs. GitHub:** Distributed VCS vs. collaboration platform.
- **Purpose:** Enhances project management and teamwork.



- Feature Branch Workflow
- Gitflow Workflow
- Fork Workflow

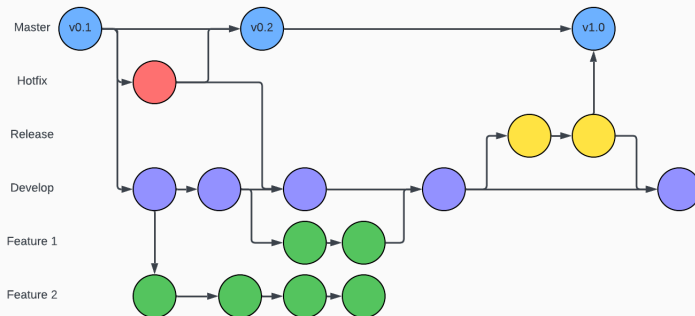
# Feature Branch Workflow

- Develop each feature in its own branch.
- Merges via pull requests for code review.
- Keeps main branch stable, encourages collaboration.
- Ideal for projects with simultaneous feature development.



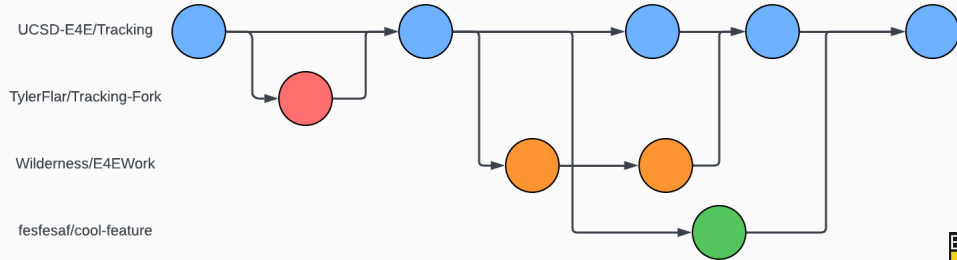
# Gitflow Workflow

- Structured model: development, features, releases, hotfixes.
- Systematic release management, clear branch roles.
- Tracks progress efficiently, supports parallel releases.
- Suited for scheduled release cycles.



# Fork Workflow

- Developers work on personal repository copies.
- Changes proposed via pull requests.
- Encourages external contributions, safe experimentation.
- Ideal for open-source and large collaborations.



- Marks significant project milestones.
- Useful for release points, use semantic versioning.
- Lightweight tags: `git tag tagname`.
- Annotated tags: `git tag -a tagname "message"`.
- List/delete tags: `git tag`, `git tag -d tagname`.

# Pull Requests (PR) Management

- Keep PRs small for easy review.
- Use checklists for consistent reviews.
- Automate tests and checks via GitHub Actions.



- Draft new release, choose git tag.
- Add release notes describing changes.
- Bundles code, executables, and assets.
- Detailed notes inform users of updates.
- Example: <https://github.com/HumanSignal/label-studio/releases>

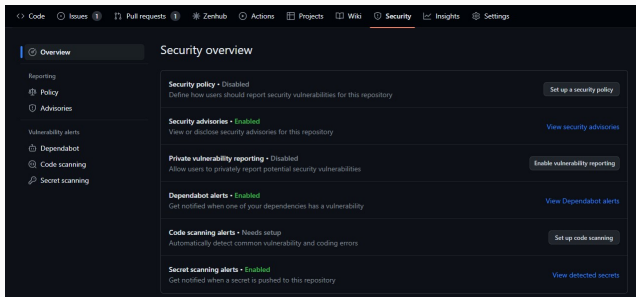
- Triggered by GitHub events (push, PRs).
- Workflows combine actions in YAML files.
- Runs on GitHub-hosted or self-hosted runners.
- Automates tests and deployment on PR merge.
- Auto-assigns issues, auto-labels PRs by path.

# Putting it Together

- Combine Tags, Releases, and GitHub Actions.
- Interact with source code.
- Example: <https://github.com/TylerFlar/MinecraftDiscord-CrossChat>

Protection against:

- Vulnerable dependencies
- Some code vulnerabilities
- Some committed secrets



- Interactive Rebase\*: Rewrite history, edit commits.
- Stashing\*: Temporarily shelf changes for tasks.
- Cherry-picking: Apply commit to another branch.
- \* Not recommended for shared/remote repositories.

# Cherry-picking

- *git cherry-pick commit-hash*

