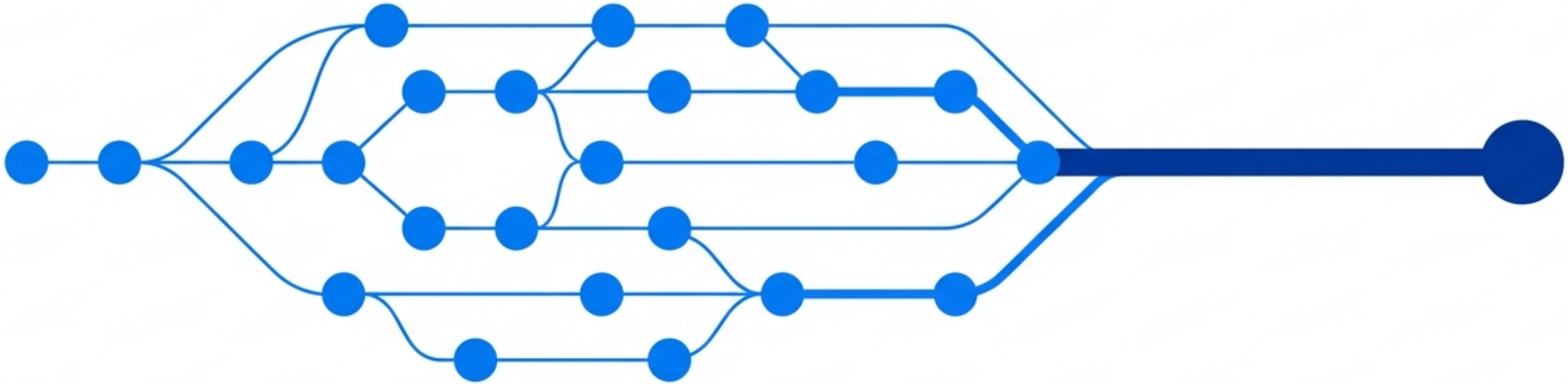
Collaborative Version Control

The Git & GitLab Handbook for Modern Teams

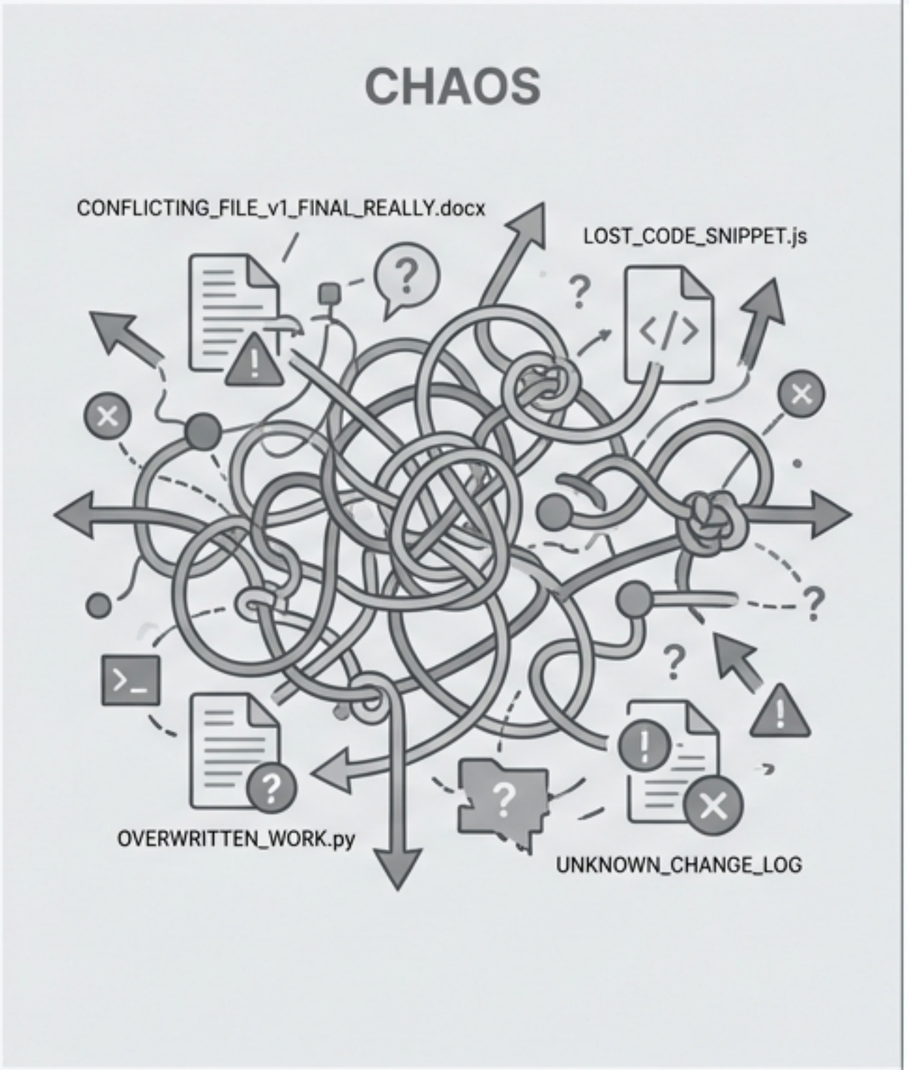


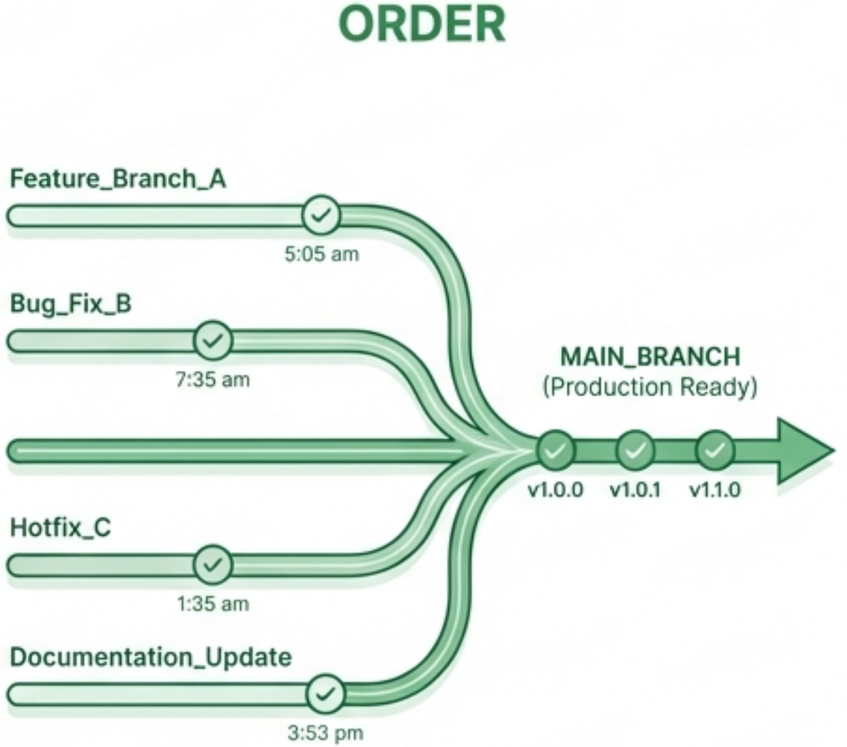
SELF-GUIDED HANDBOOK

Collaboration Without Chaos

Version control is the safety net for software teams. It tracks every change history, prevents lost code, and enables parallel work.

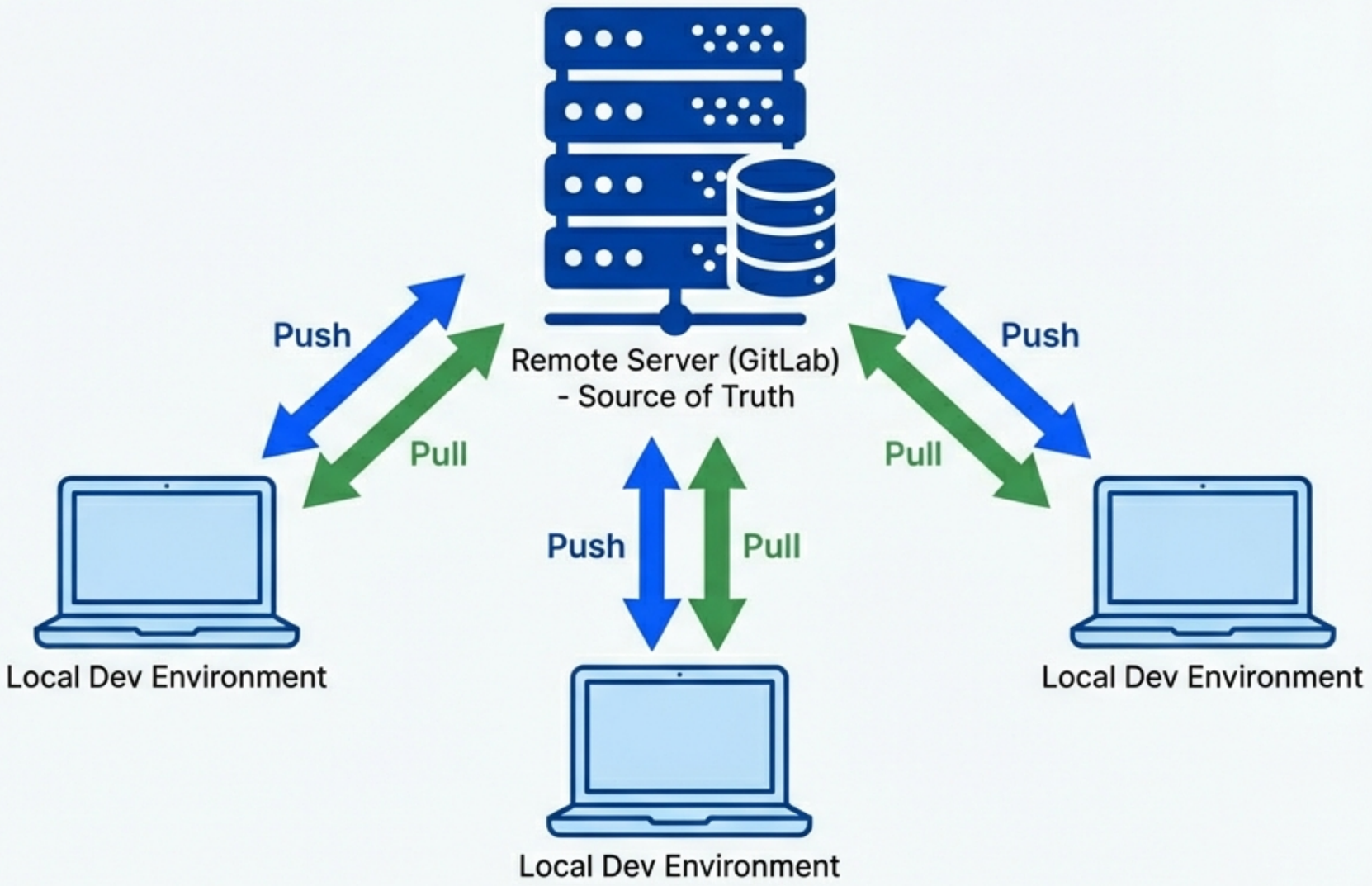
- **Traceability:** Who changed what, and when?
- **Safety:** Revert mistakes instantly.
- **Concurrency:** Developers work independently, then merge.





The Distributed Architecture

Git is a Distributed Version Control System (DVCS). Unlike older systems, every developer holds the full history of the project locally, not just a snapshot of the current files.



The Three Stages of File Tracking

Git uses a specific three-step holding pattern to move a file from “edited” to “permanently saved”.

Working Directory

Where you edit files.

Staging Area (Index)

The “on-deck” circle. Files marked for next commit.

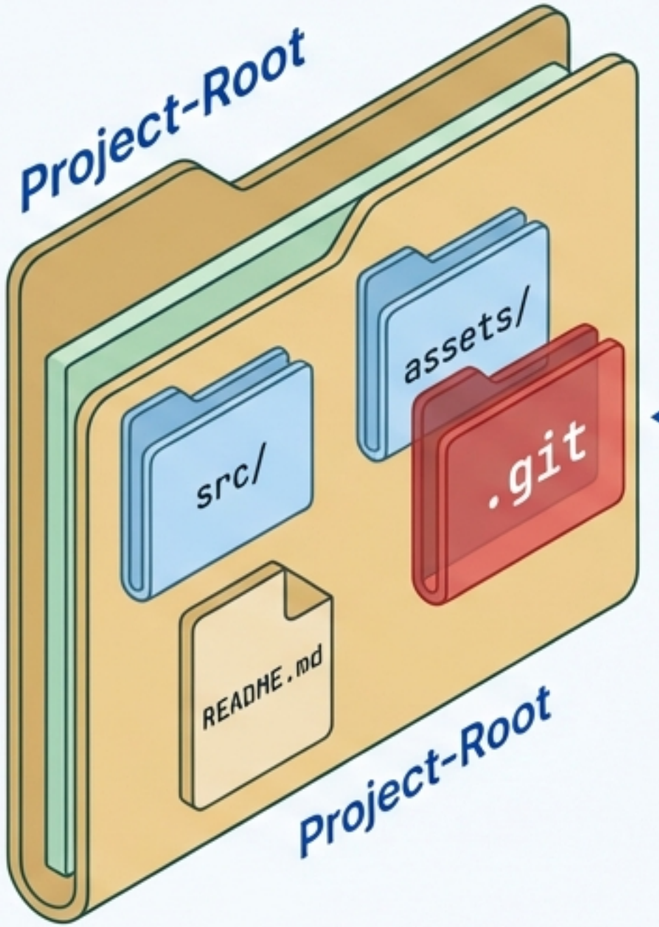
Local Repository

Permanent snapshots of project history.

The Repository: Your Project's Core

A Repository (Repo) is the project directory containing your source code and the configuration data.

- **Local Repo:** Stored on your machine.
- **Remote Repo:** Hosted on GitLab; the central collaboration point.

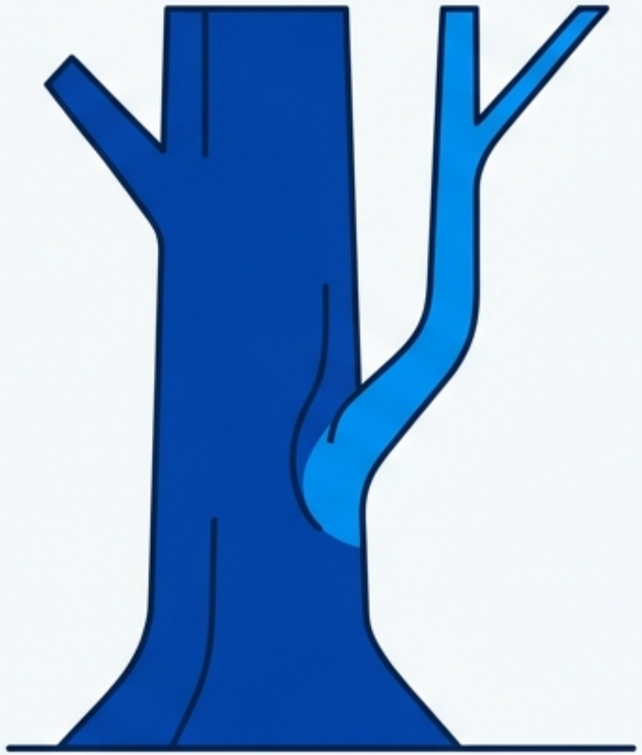


Hidden Directory. Contains all version history and config. Do not touch manually.

Isolation Strategies: Branches vs. Forks

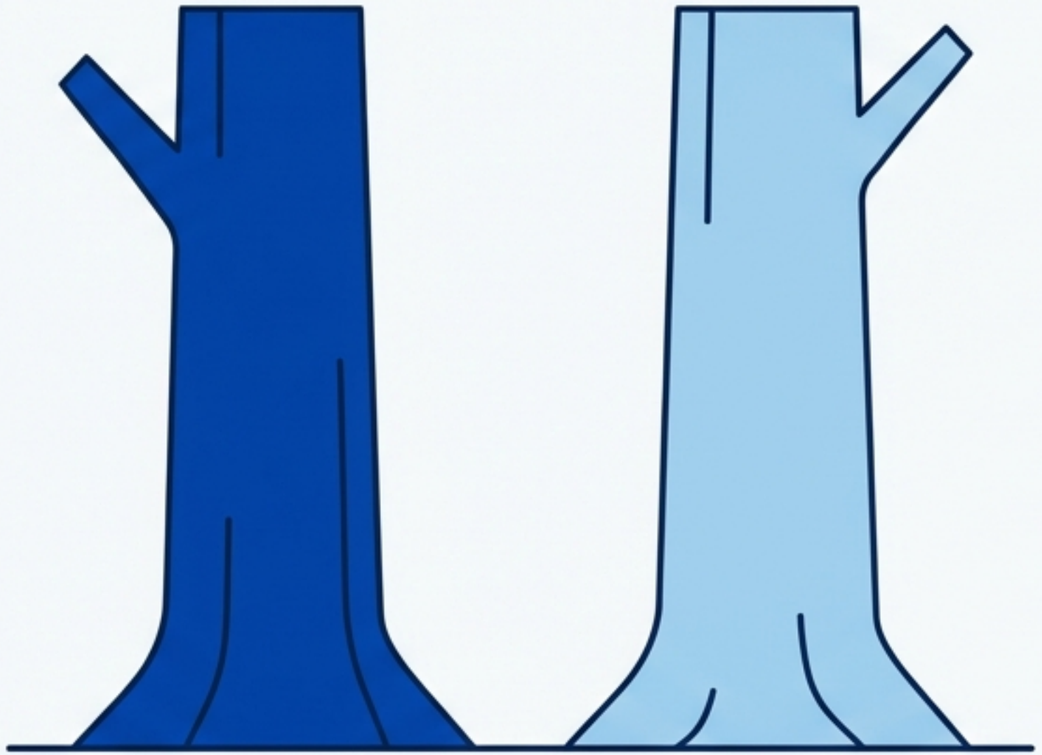
How do we work safely without breaking the main code?

Branching (Team Standard)



Parallel version within the *same repo. Used for daily team collaboration.

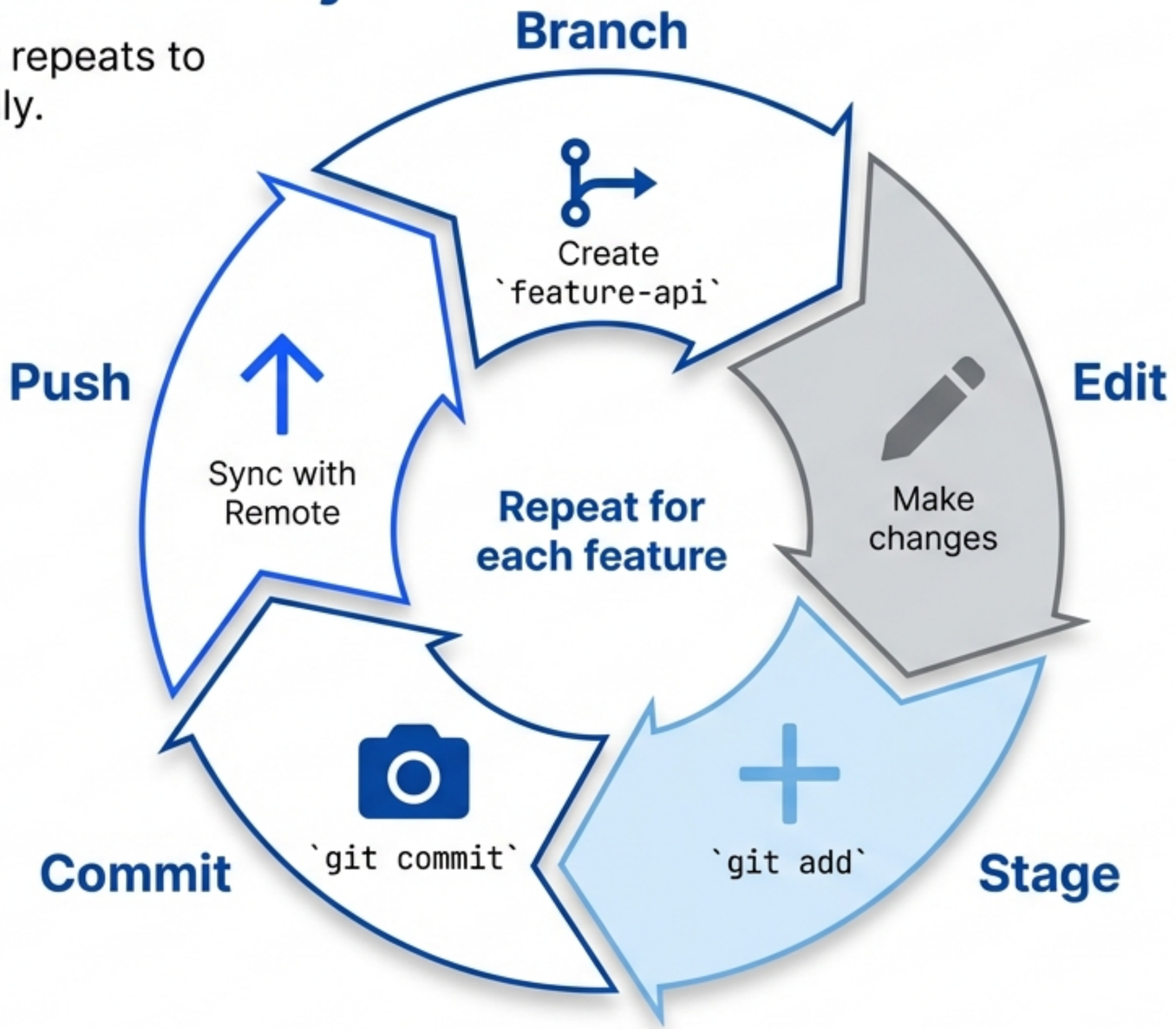
Forking (Open Source Standard)



A full copy of a *different* repo. Used when you don't have write access (e.g., Open Source).

The Daily Workflow Cycle

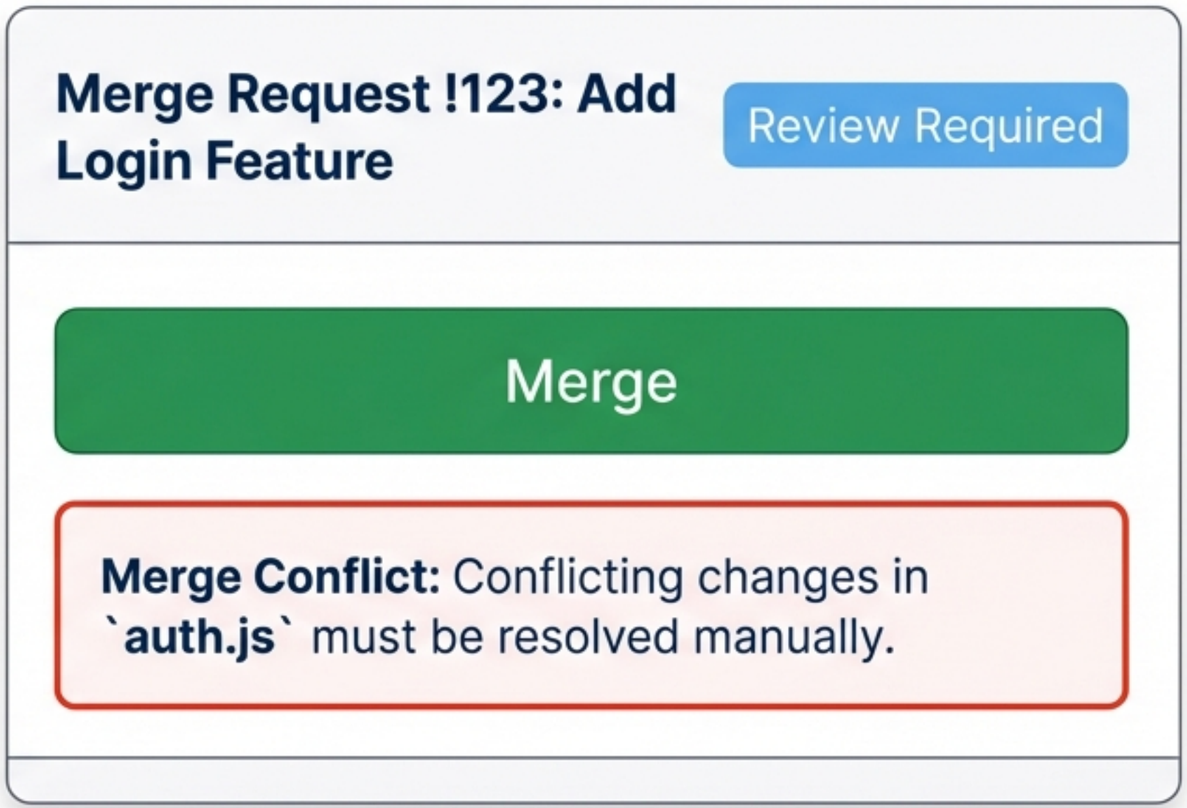
The loop a developer repeats to evolve a feature locally.



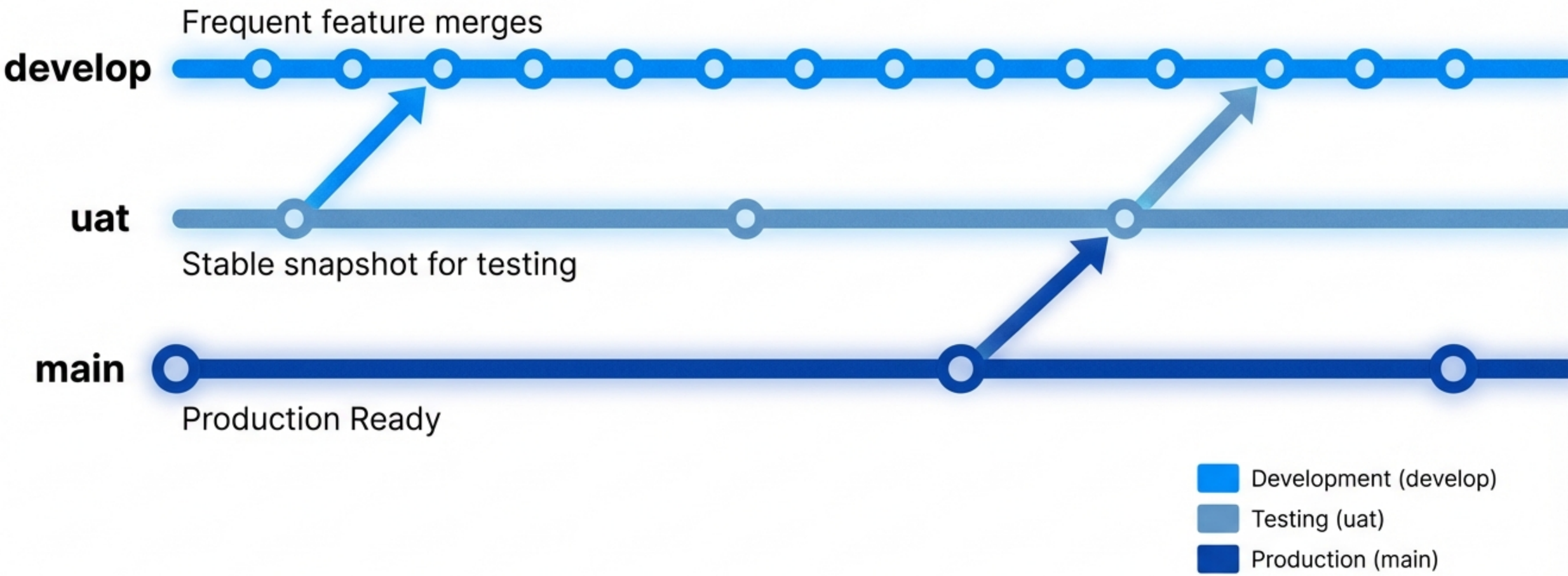
The Integration: Merge Requests

GitLab is the arena for review and ratification. Code is proposed via a Merge Request (MR).

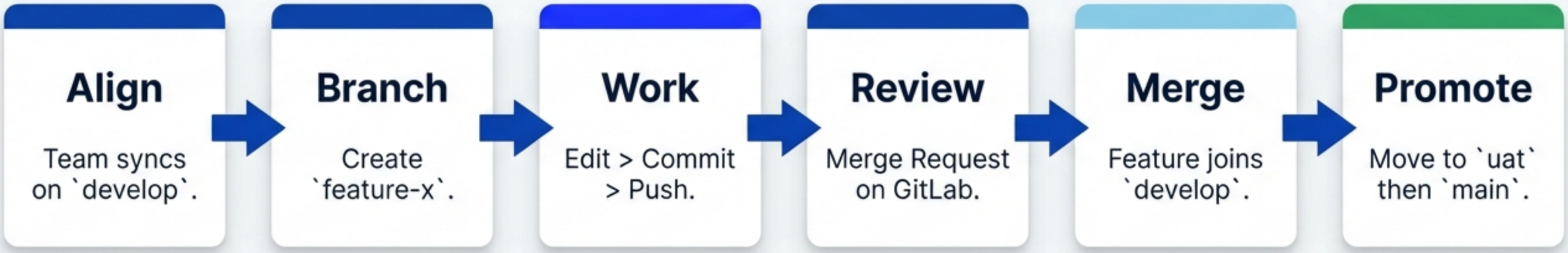
1. Dev opens MR.
2. Team reviews and comments.
3. Conflict resolution (if needed).
4. Code merged into source of truth.



Release Management Strategy



The Lifecycle: From Idea to Production

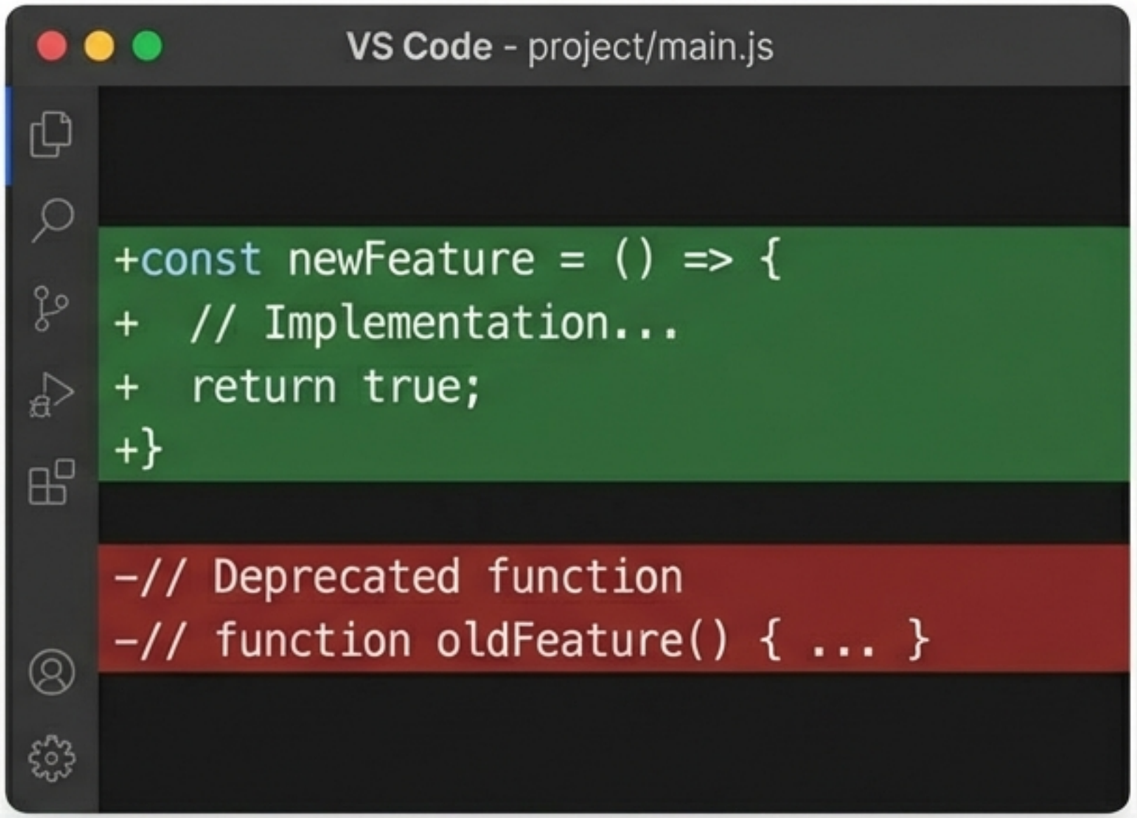


Essential Command Reference

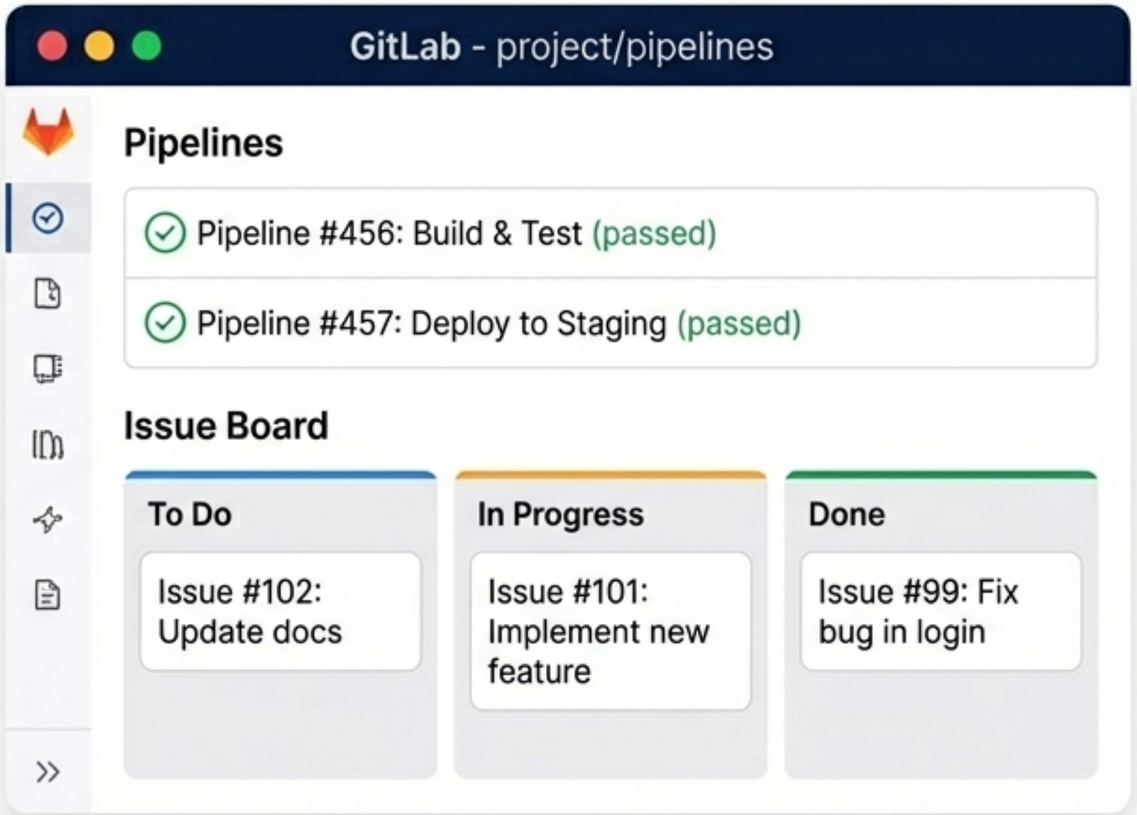
```
# Setup
git clone [url]

# Branching
git checkout -b [branch-name]

# Saving
git add .
git commit -m "message"

# Syncing
git push -u origin [branch-name]
git pull origin develop
```

Modern Tooling Integration

Integrating Git into your daily environment.



Local Editor (VS Code)): Writing, Committing, Branching.



Web Platform (GitLab): Reviews, CI/CD, Project Management.

A Shared Language of Collaboration

Git is more than just a history of code; it is the structure that allows teams to scale.

Structure **Traceability** **Confidence**

Collaborative Version Control Handbook