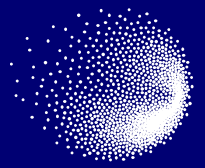


PSI Center for Scientific Computing, Theory and Data

Introduction to CI/CD

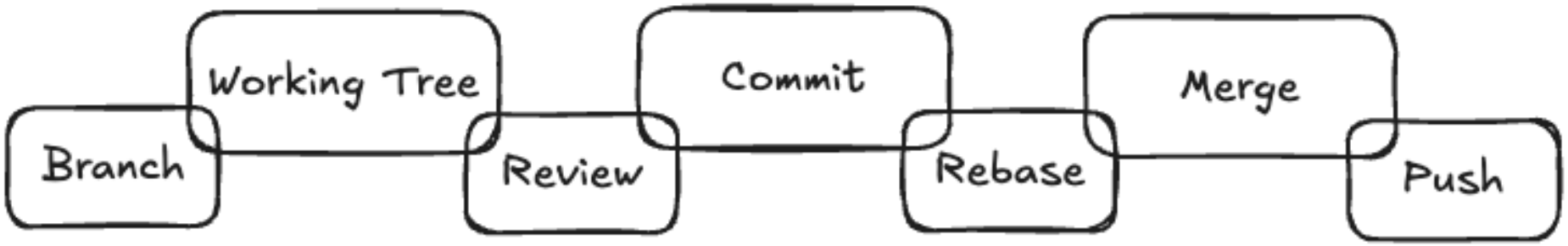
From a single commit to an automated release

Basil Bruhn
12/05/2026

Agenda

- Looking back in time
- Working on a Git Forge
- Issues and Planning
- Pull Requests and Reviews
- Continuous Integration
- Continuous Delivery
- Releases
- The Complete Workflow

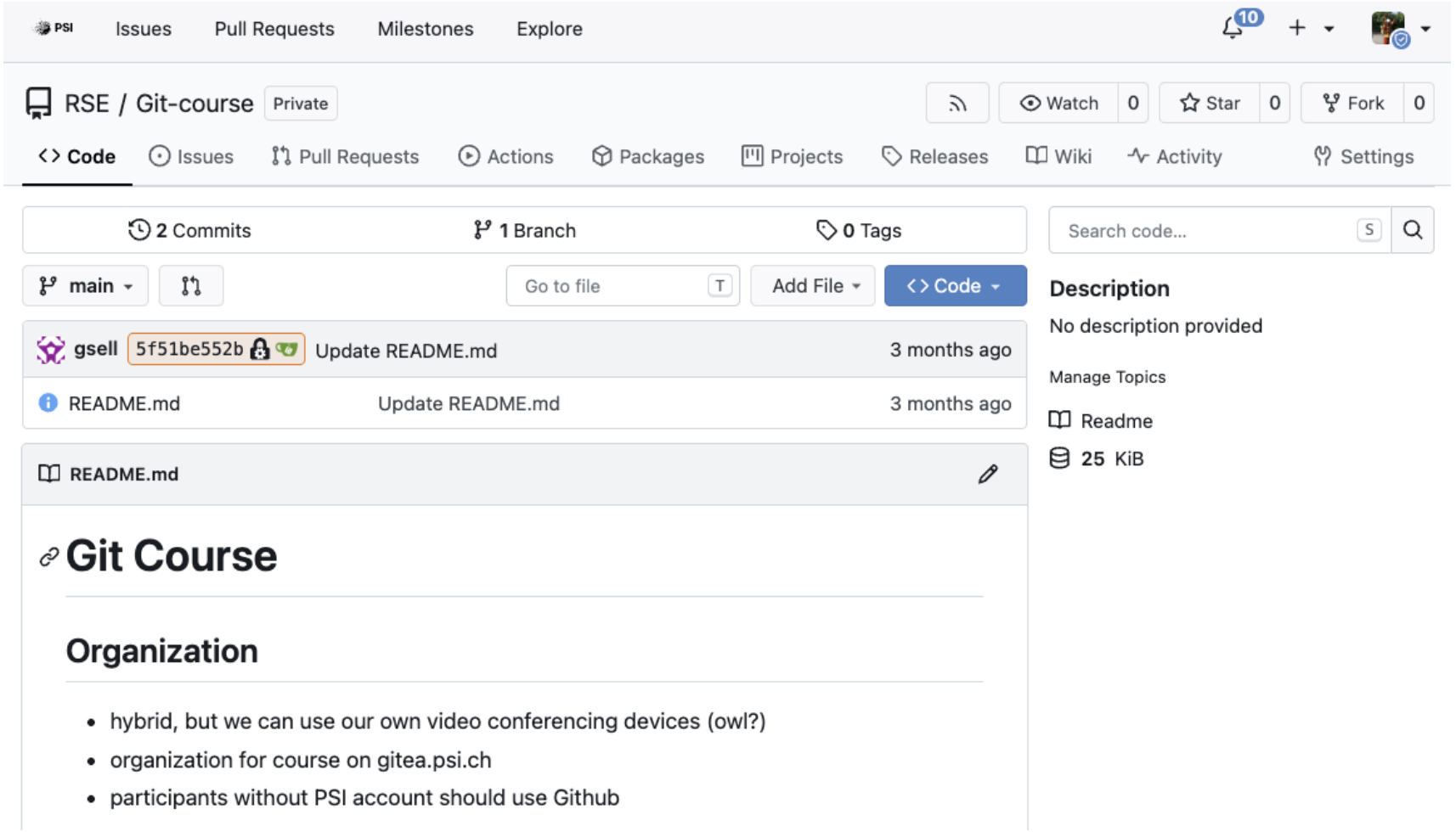
Looking back in time



But what happens after `git push`?

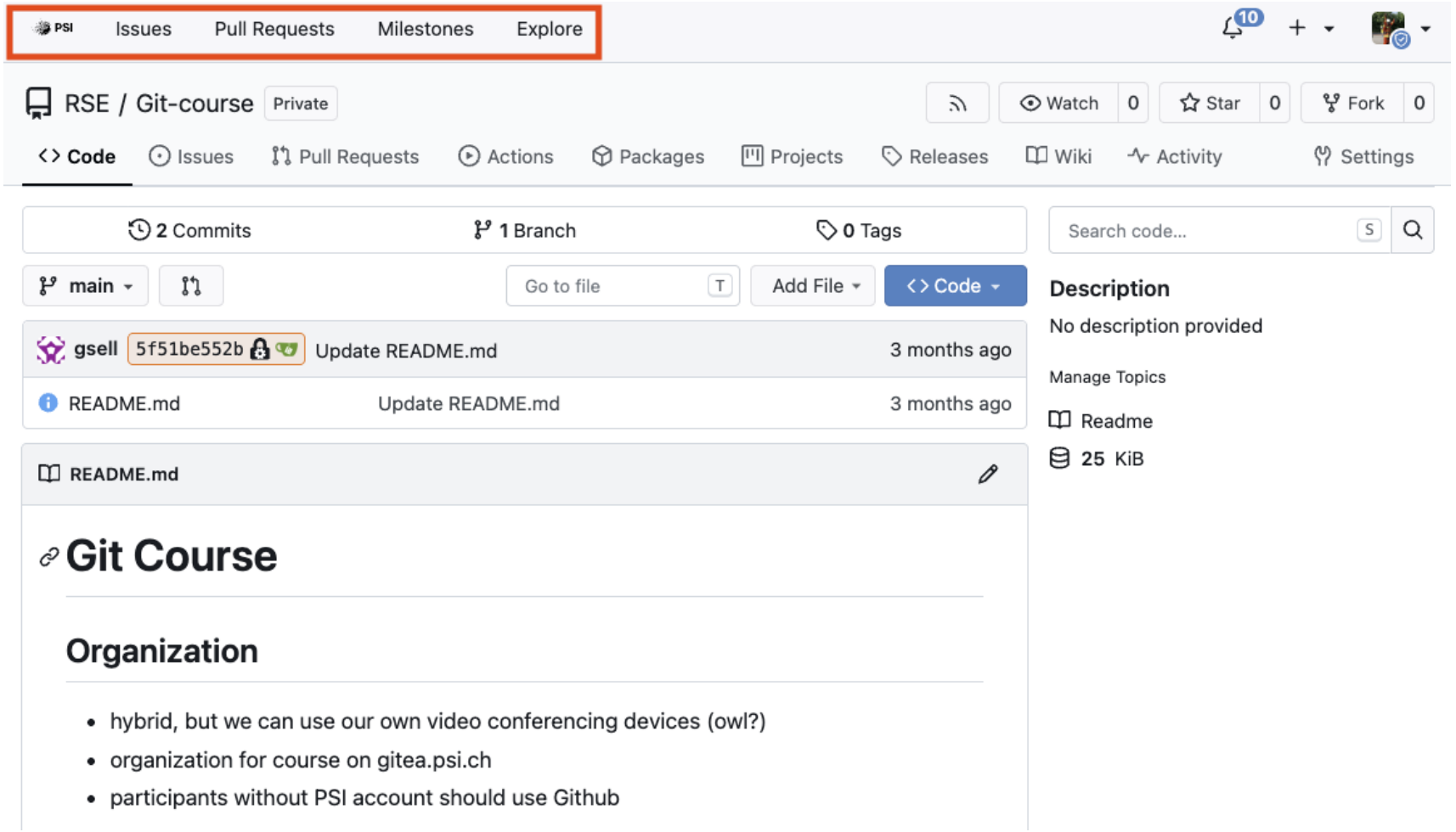
Collaboration on a Git Forge

What is a Git Forge?



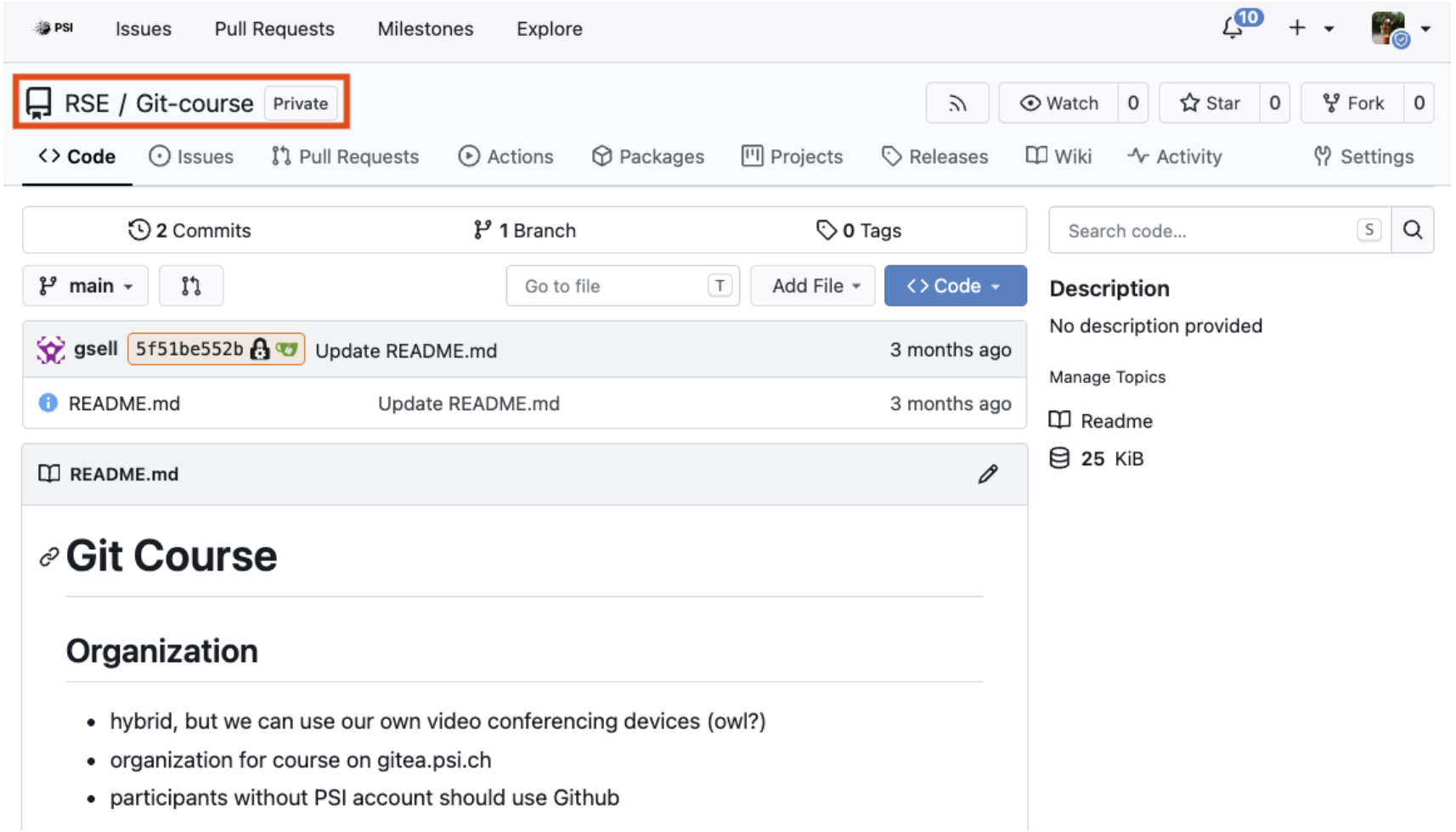
Collaboration on a Git Forge

What is a Git Forge?



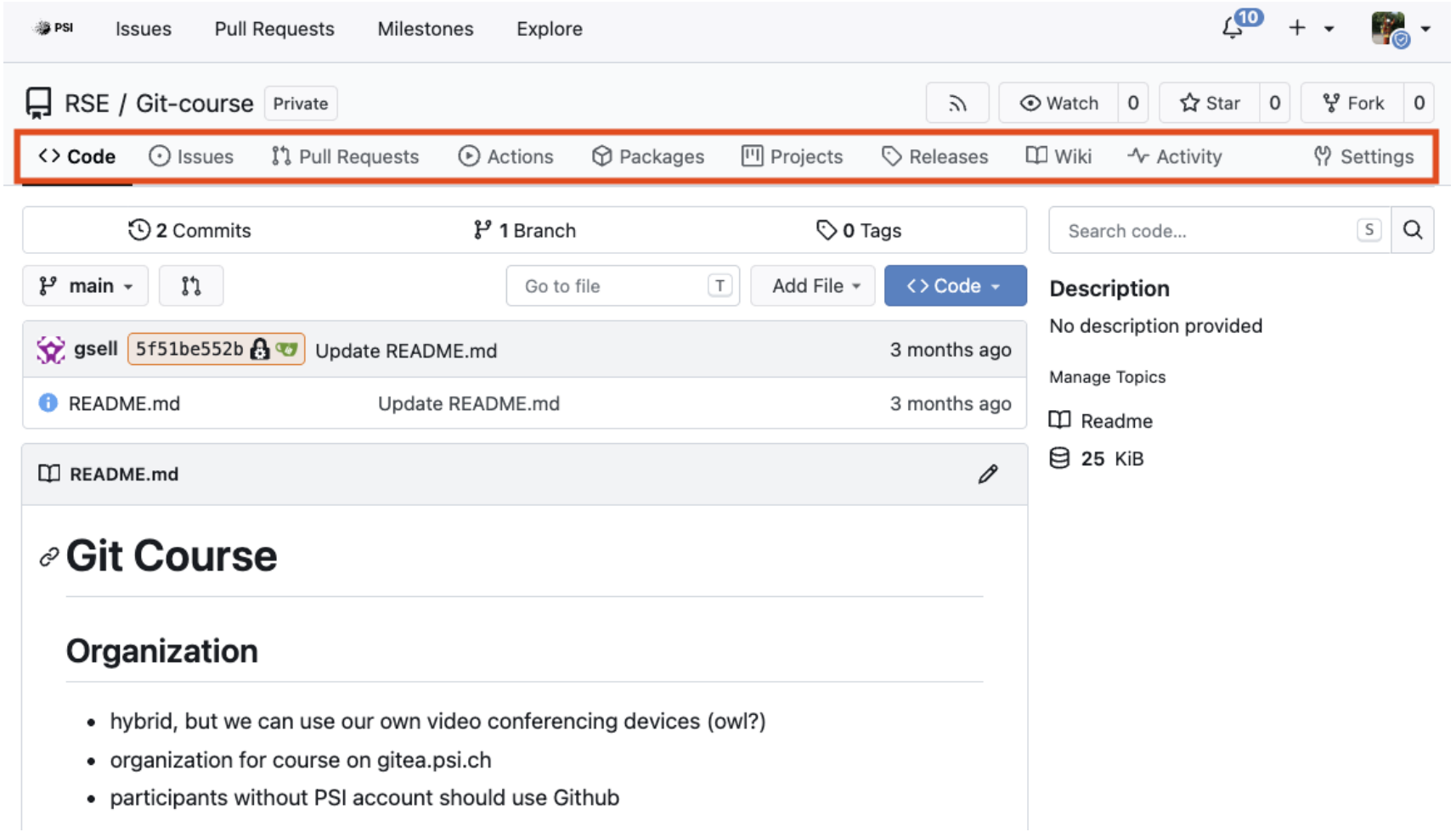
Collaboration on a Git Forge

What is a Git Forge?

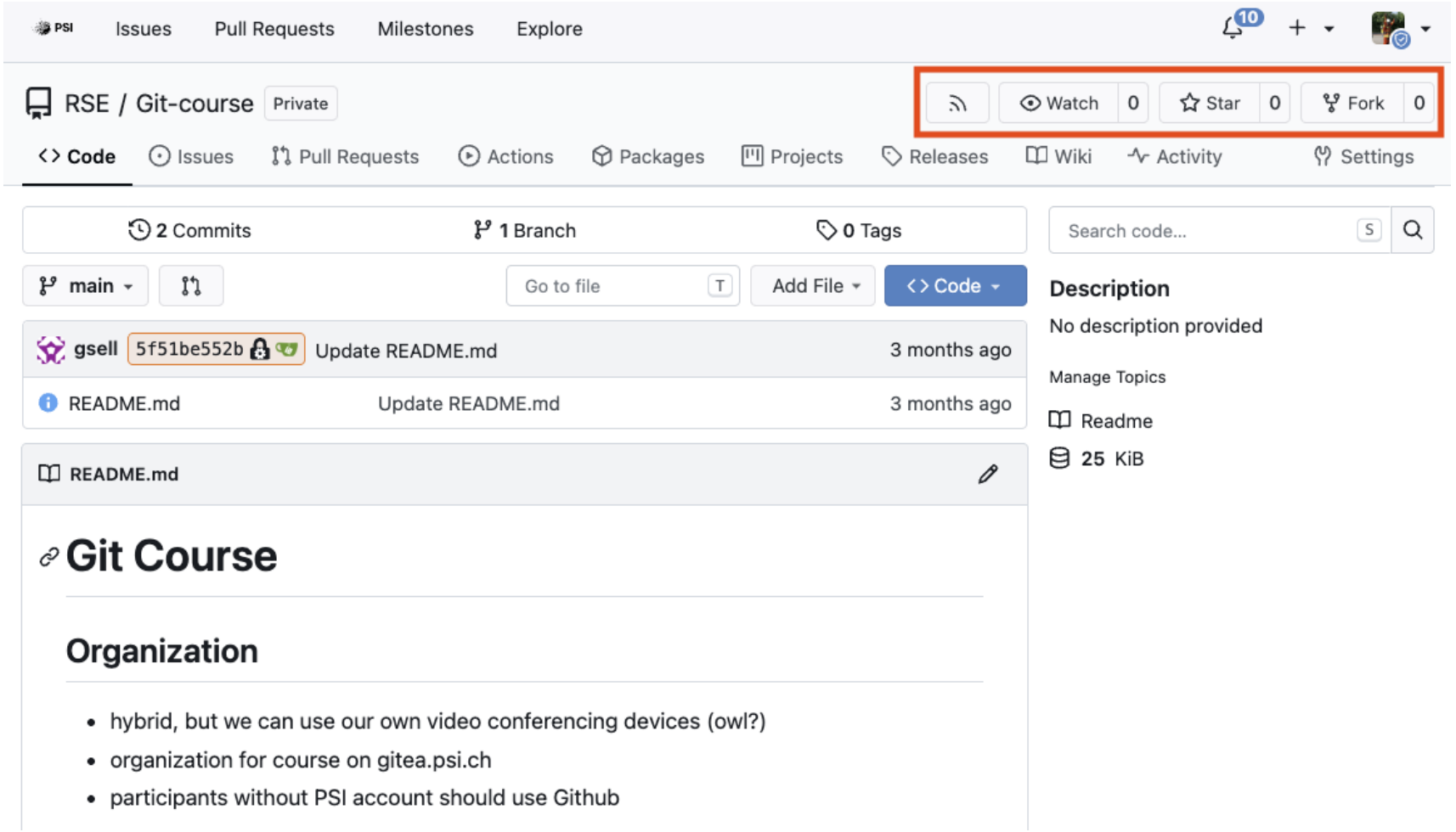


Collaboration on a Git Forge

What is a Git Forge?

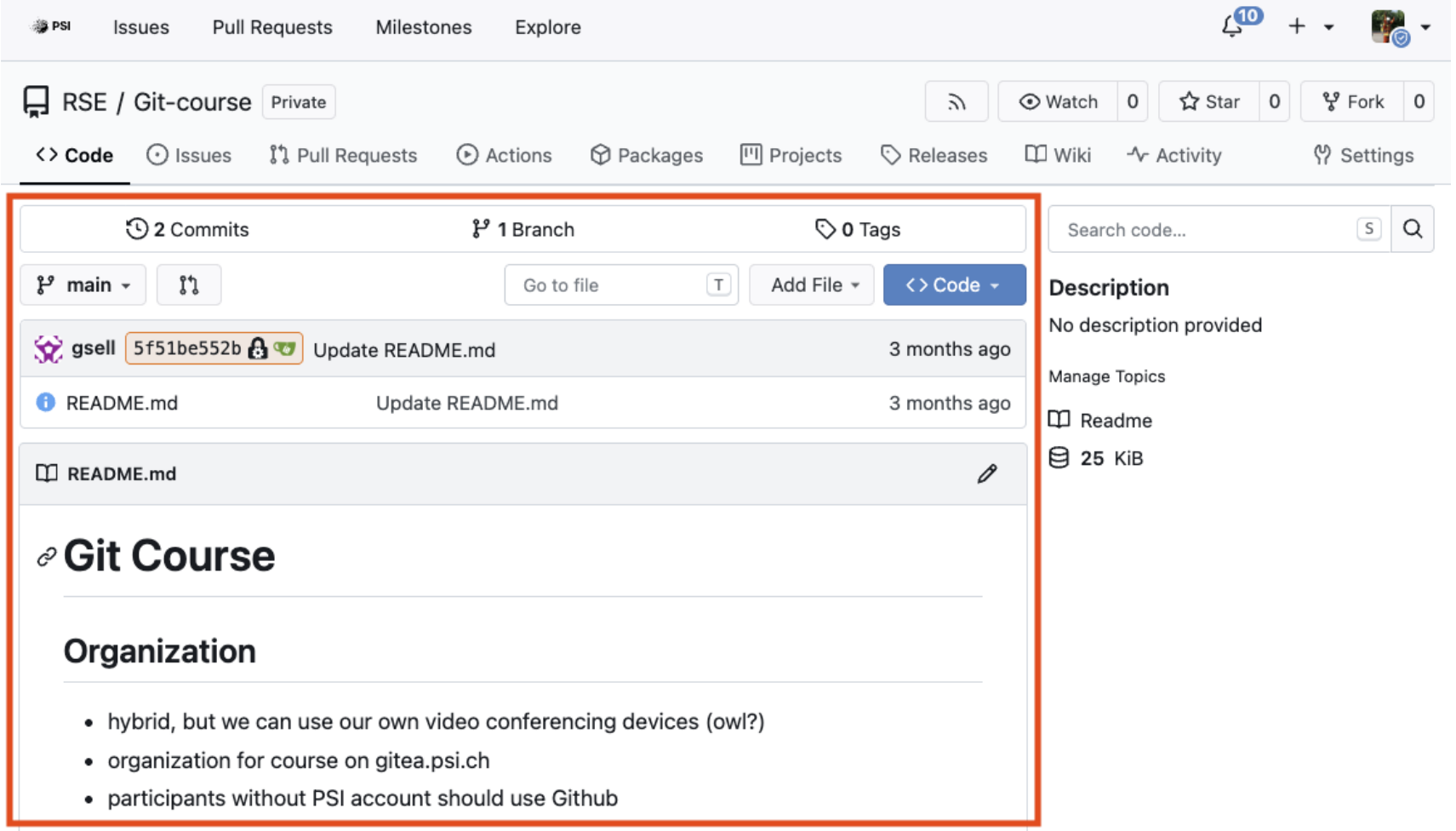


Collaboration on a Git Forge
What is a Git Forge?

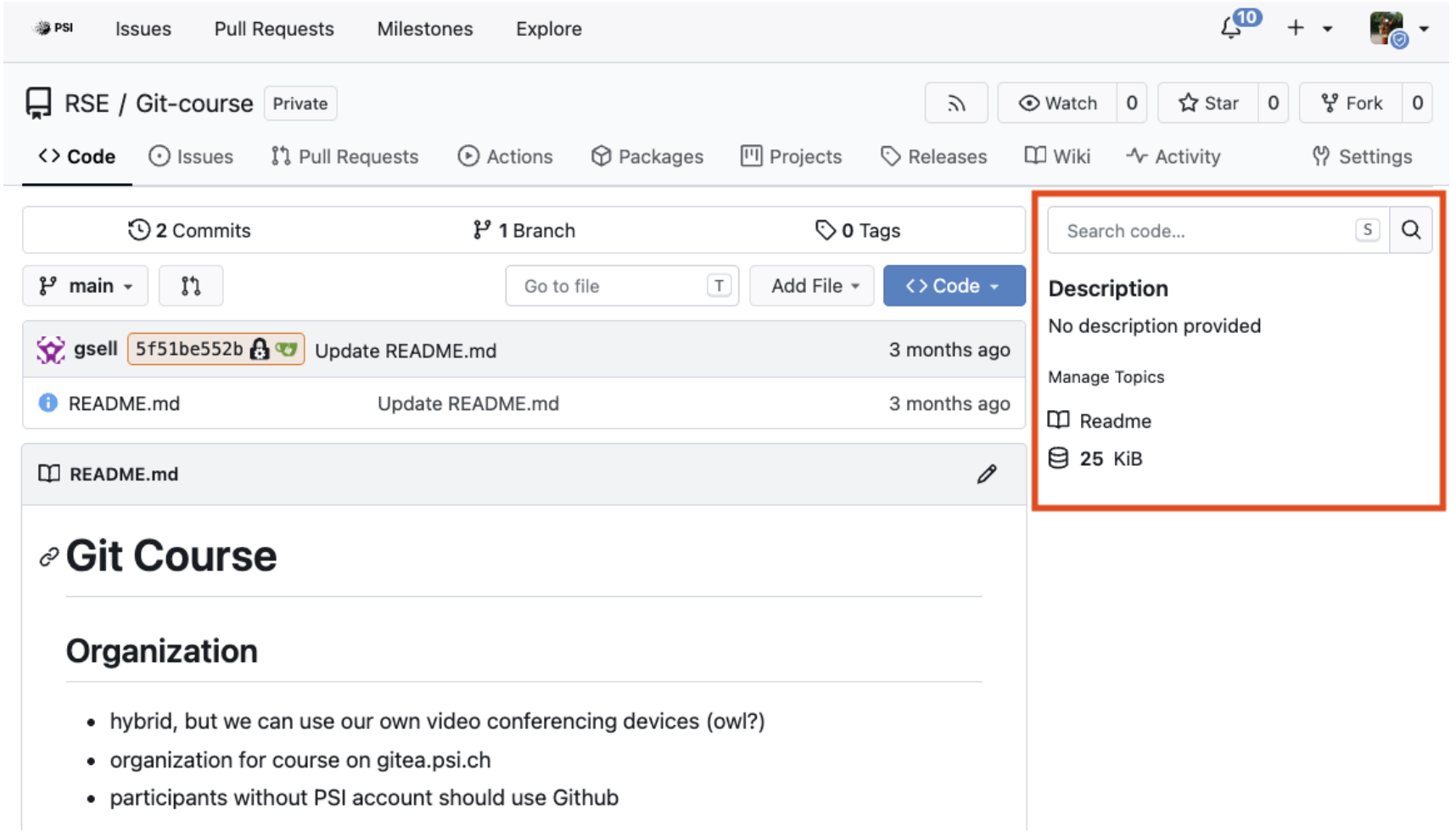


Collaboration on a Git Forge

What is a Git Forge?

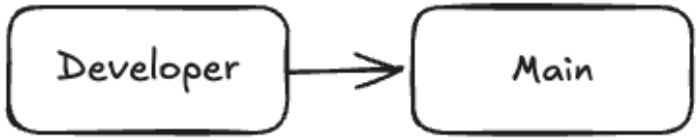


Collaboration on a Git Forge
What is a Git Forge?

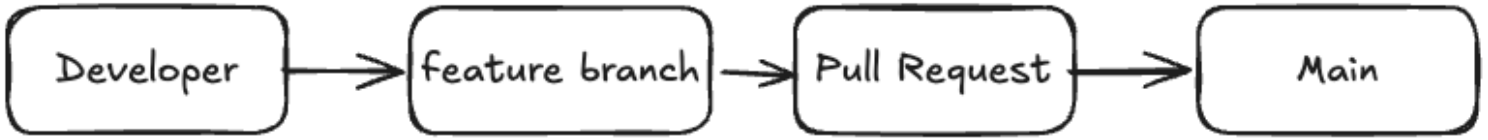


Collaboration on a Git Forge
Ways of working

- Direct commit

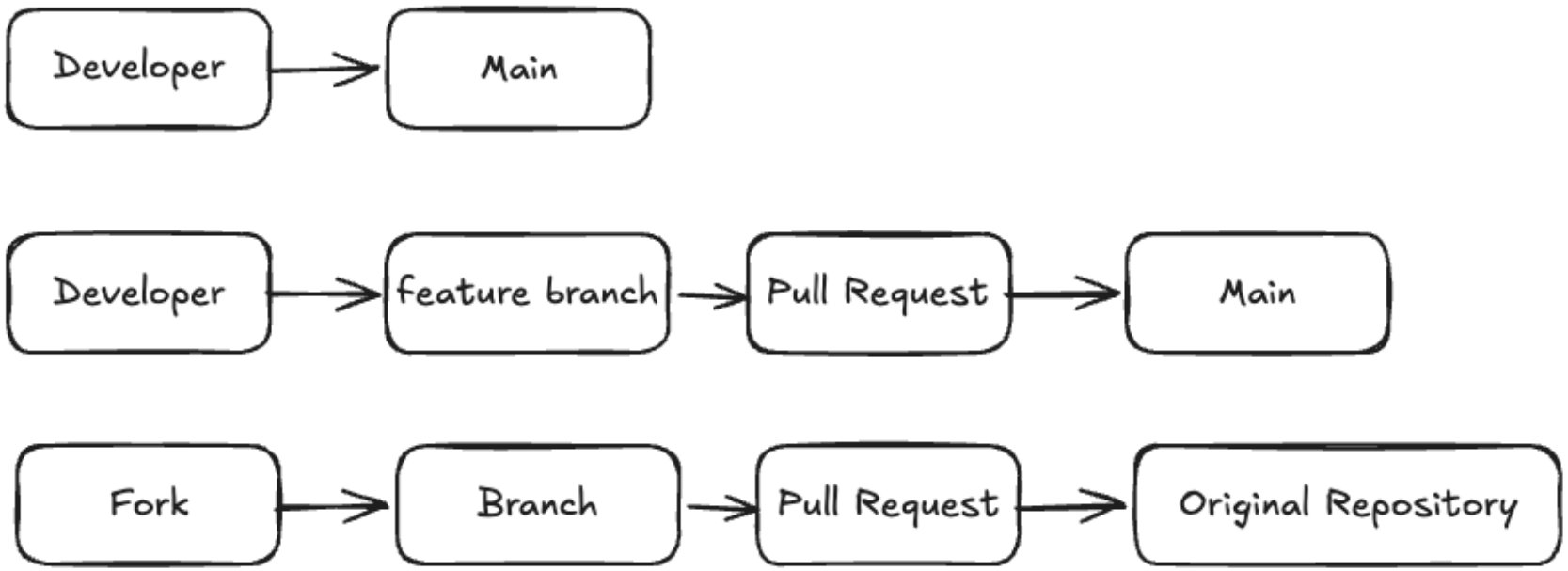


- Most common



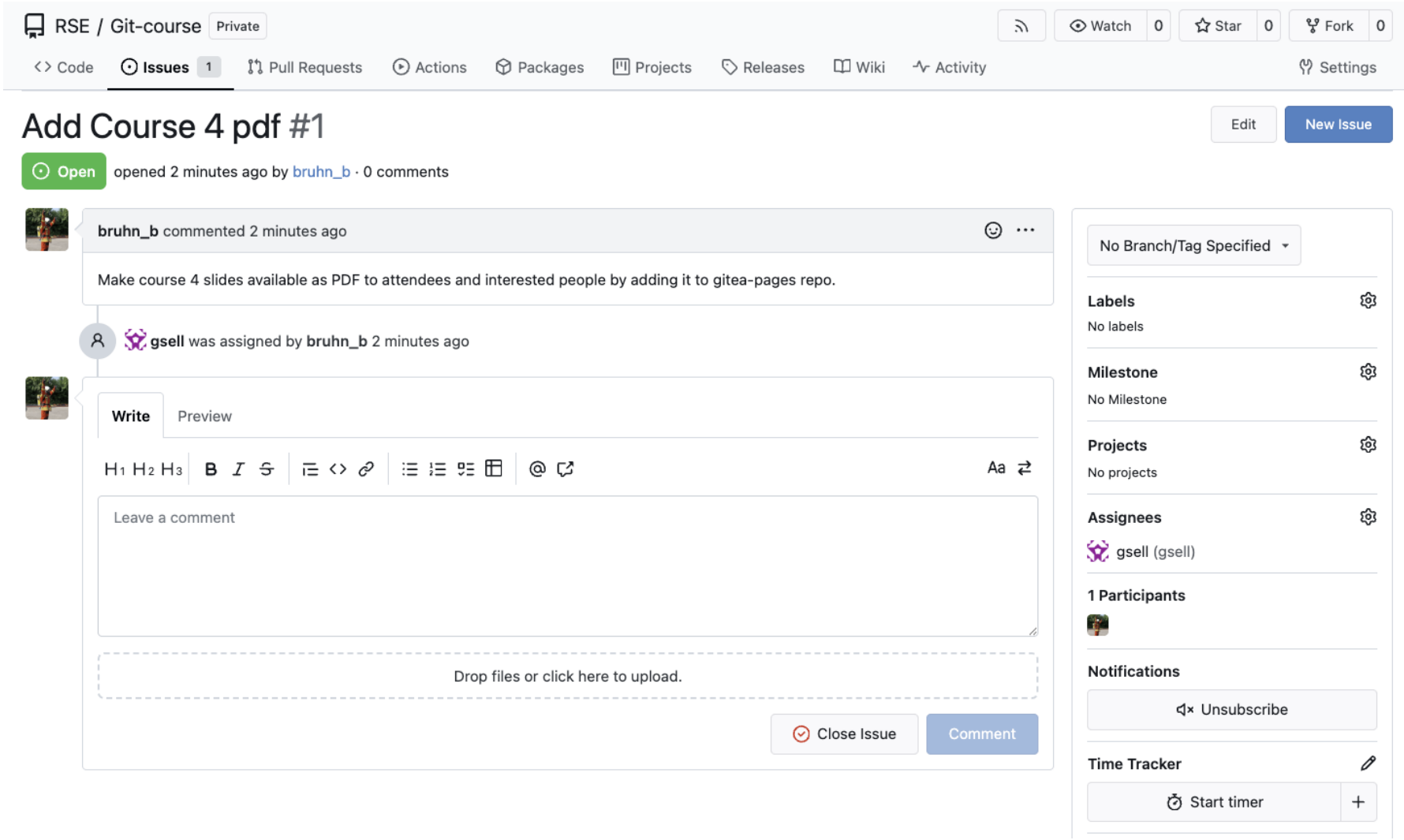
- Fork Workflow

Collaboration on a Git Forge
Ways of working



Issues and Planning

Issues



Issues and Planning
Linking Work

PSI

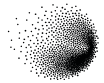
- Branch naming examples:
  - `issue1-course4-notes`
  - `copilot/fix-issue-1`
  - `release/v1.1`

Issues and Planning

Why use issues?

PSI

- Without issues:
  - Ideas disappear ☹
- With issues:
  - Ideas become work items!



A proposal to change history

Pull Requests and Reviews

Anatomy of a Pull Request

PSI

Repository Hygiene and Git LFS
Demo Setup

PSI

```
➜  research-notes git:(main) ✗ git status

➜  research-notes git:(main) ✗ git add .

➜  research-notes git:(main) ✗ git commit -m "Initial project
structure"
[main (root-commit) 20355af] Initial project structure
 3 files changed, 7 insertions(+)
 create mode 100644 README.md
 create mode 100644 analyze.py
 create mode 100644 notes.txt
```

Repository Hygiene and Git LFS

.gitignore again?

PSI

```
➜  research-notes git:(main) touch experiment_results.csv
➜  research-notes git:(main) ✗ touch analysis_output.tmp

➜  research-notes git:(main) ✗ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        analysis_output.tmp
        experiment_results.csv

nothing added to commit but untracked files present (use "git add" to
track)
```

Repository Hygiene and Git LFS

.gitignore again?

PSI

```
➜  research-notes git:(main) ✗ nvim .gitignore
*.tmp
*.csv

➜  research-notes git:(main) ✗ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore

nothing added to commit but untracked files present (use "git add" to
track)
```

Repository Hygiene and Git LFS

.gitignore again?

PSI

```
➜  research-notes git:(main) ✗ git add .gitignore

➜  research-notes git:(main) ✗ git commit -m "add repository hygiene
rules"
[main c46568f] add repository hygiene rules
 1 file changed, 2 insertions(+)
 create mode 100644 .gitignore
```

Repository Hygiene and Professional Workflows

PSI

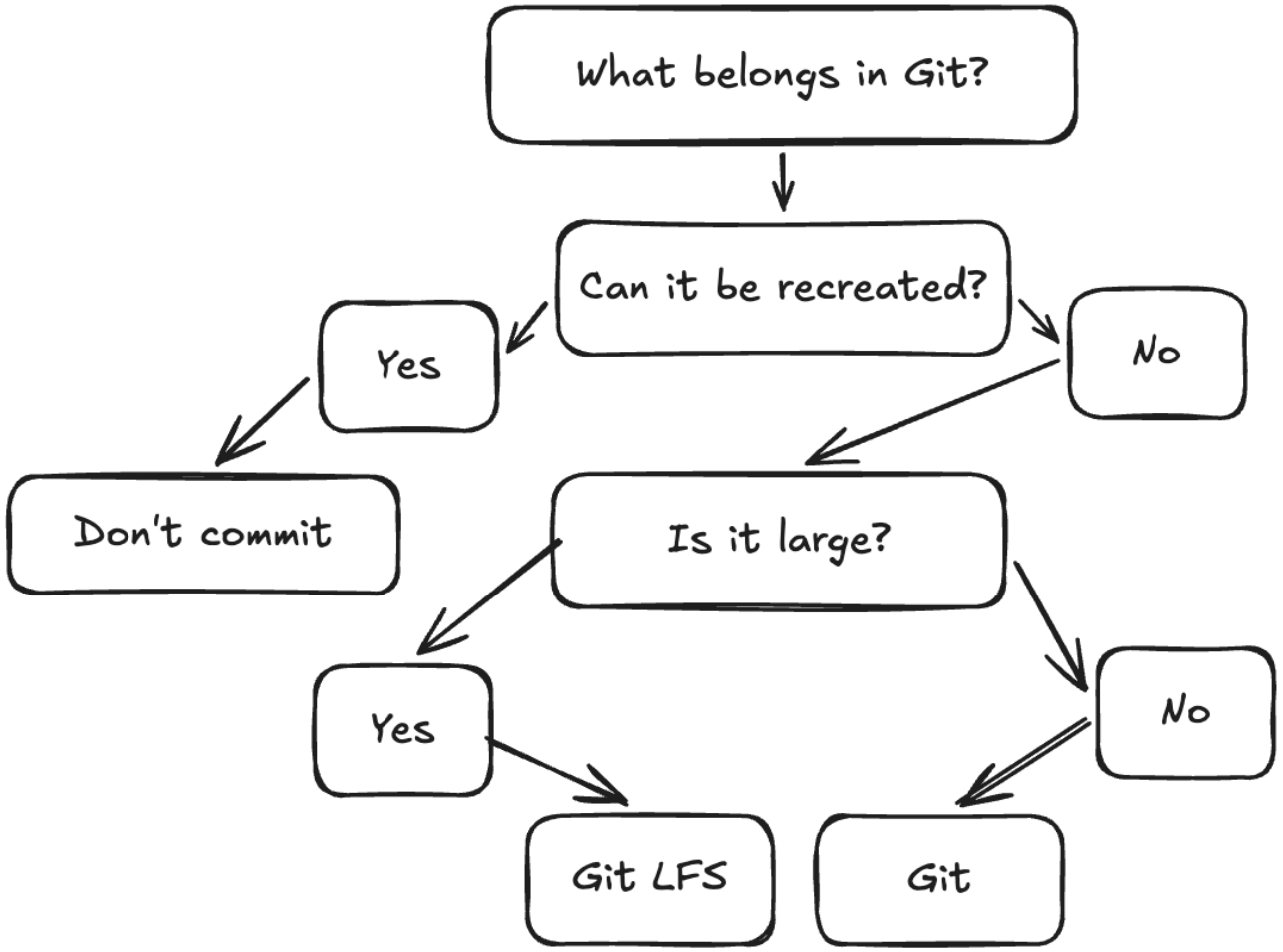
Good repository content

- source code
- scripts
- infrastructure code
- documentation
- configuration

Bad repository content

- caches
- dependencies
- temporary files
- logs
- secrets
- generated artifacts
- large binaries

Repository Hygiene and Professional Workflows



Repository Hygiene and Git LFS

git-lfs

PSI

```
➜  research-notes git:(main) dd if=/dev/zero of=dataset.bin bs=1M
count=500


➜  research-notes git:(main) ✗ ls -lh
total 1024024
-rw-r--r--@ 1 bruhn_b  staff     0B May 22 10:39 analysis_output.tmp
-rw-r--r--@ 1 bruhn_b  staff    22B May 22 10:13 analyze.py
-rw-r--r--@ 1 bruhn_b  staff   500M May 22 10:47 dataset.bin
-rw-r--r--@ 1 bruhn_b  staff     0B May 22 10:39 experiment_results.csv
-rw-r--r--@ 1 bruhn_b  staff    46B May 22 10:13 notes.txt
-rw-r--r--@ 1 bruhn_b  staff    58B May 22 10:13 README.md
```

Repository Hygiene and Git LFS

`git-lfs`

PSI

```
➜  research-notes git:(main) ✗ git lfs install
Updated Git hooks.
Git LFS initialized.

➜  research-notes git:(main) ✗ git lfs track *.bin
Tracking "dataset.bin”

➜  research-notes git:(main) ✗ cat .gitattributes
dataset.bin filter=lfs diff=lfs merge=lfs -text
```

Repository Hygiene and Git LFS

`git-lfs`

PSI

```
➜  research-notes git:(main) ✗ git add .gitattributes

➜  research-notes git:(main) ✗ git add dataset.bin

➜  research-notes git:(main) ✗ git commit -m "track datasets with git
lfs"
[main 78eae41] track datasets with git lfs
 2 files changed, 4 insertions(+)
 create mode 100644 .gitattributes
 create mode 100644 dataset.bin
```

Repository Hygiene and Git LFS
`git-lfs`

PSI

Good LFS candidates

- Datasets
- Firmware
- Media
- Scientific assets

Better commits and partial staging

```
➜  research-notes git:(main) echo "Humidity: 45%" >> notes.txt

➜  research-notes git:(main) ✗ echo 'print("Debug mode")' >>
analyze.py

➜  research-notes git:(main) ✗ echo "" >> README.md

➜  research-notes git:(main) ✗ echo "Added humidity measurements"
>> README.md
```

Better commits and partial staging

```
➜  research-notes git:(main) ✗ git diff
```

Better commits and partial staging

PSI

Bad commit

- Add feature
- Fix bug
- Update Docs
- Debug Output

Good commit

- One logical change

Better commits and partial staging

```
➜  research-notes git:(main) X git add -p
diff --git a/README.md b/README.md
index f61e178..92a937c 100644
--- a/README.md
+++ b/README.md
@@ -1,3 +1,5 @@
 # Research Notes

 Simple repository used for Git training
+
+Added humidity measurements
(1/1) Stage this hunk [y,n,q,a,d,e,p,?]y
```

Better commits and partial staging

```
diff --git a/analyze.py b/analyze.py
index 080d532..bcadad9 100644
--- a/analyze.py
+++ b/analyze.py
@@ -1 +1,2 @@
 print("Analyze data")
+print("Debug mode")
(1/1) Stage this hunk [y,n,q,a,d,e,p,?]? n
```

Better commits and partial staging

```
diff --git a/notes.txt b/notes.txt
index 21152c4..0d4e7e8 100644
--- a/notes.txt
+++ b/notes.txt
@@ -1,3 +1,4 @@
 Experiment 1
 Temperature: 20C
 Result: pending
+Humidity: 45%
(1/1) Stage this hunk [y,n,q,a,d,e,p,?]? y
```

Better commits and partial staging

```
➜  research-notes git:(main) ✗ git commit -m "add humidity measurements"
[main d28dbd3] add humidity measurements
 2 files changed, 3 insertions(+)
```

Better commits and partial staging

```
➜  research-notes git:(main) ✗ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   analyze.py
no changes added to commit (use "git add" and/or "git commit -a")
```

Real world interruptions

- Production issue
- Colleague needs help
- Wrong branch
- Need to switch context

Real world interruptions

```
➜  research-notes git:(main) ✗ git stash
Saved working directory and index state WIP on main: d28dbd3 add
humidity measurements

➜  research-notes git:(main) git status
On branch main
nothing to commit, working tree clean
```

Real world interruptions

```
➜  research-notes git:(main) git stash pop
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   analyze.py

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (f6f7990abba30648a1c355688ef234c3e478bb82)
```

Real world interruptions

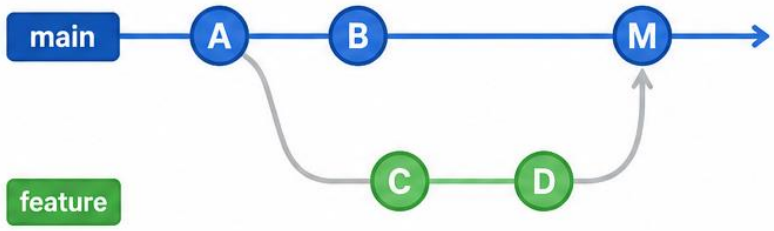
```
➜  research-notes git:(main) git restore analyze.py
```

Rebasing and history management

Merge vs Rebase

MERGE

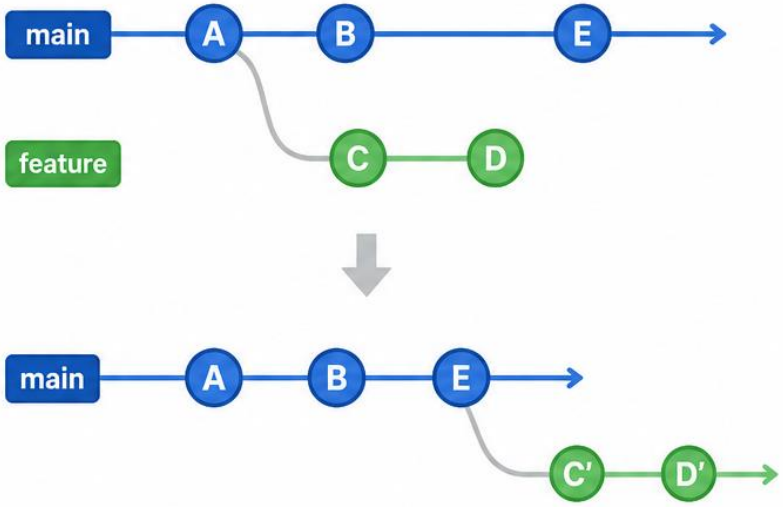
Combine timelines



Result:
Both histories are preserved.
A new merge commit (M) is created.

REBASE

Move timeline onto a newer base



Result:
Your changes (C, D) are replayed on top of the new base (E).
Timeline stays clean and linear.

Rebasing and history management

```
➜  research-notes git:(main) ✗ git switch -c experiment-results
Switched to a new branch 'experiment-results’

➜  research-notes git:(experiment-results) ✗ echo "Result:
successful" >> notes.txt

➜  research-notes git:(experiment-results) ✗ git add notes.txt

➜  research-notes git:(experiment-results) ✗ git commit -m
"record experiment result"
[experiment-results eb08b43] record experiment result
 1 file changed, 1 insertion(+)
```

Rebasing and history management

```
➜  research-notes git:(main) echo "Research phase: initial
testing" >> README.md

➜  research-notes git:(main) ✗ git add README.md

➜  research-notes git:(main) ✗ git commit -m "document research
phase"
[main f796e74] document research phase
 1 file changed, 1 insertion(+)
```

Rebasing and history management

```
➜  research-notes git:(main) git switch experiment-results
Switched to branch 'experiment-results’

➜  research-notes git:(experiment-results) git rebase main
Successfully rebased and updated refs/heads/experiment-results.
```

Rebasing and history management

What if both branches modify the same line?

- Conflict handling is the same idea as merge conflicts

Cherry-picking and selective history

Do you need the whole branch?

-> merge

Do you need one commit?

-> cherry-pick

Cherry-picking and selective history

```
➜  research-notes git:(main) git switch -c hotfix
Switched to a new branch 'hotfix’

➜  research-notes git:(hotfix) echo "Corrected temperature: 21C" >>
notes.txt

➜  research-notes git:(hotfix) ✗ git add notes.txt

➜  research-notes git:(hotfix) ✗ git commit -m "correct temperature
value”
[hotfix 574dead] correct temperature value
 1 file changed, 1 insertion(+)
```

Cherry-picking and selective history

```
➜  research-notes git:(hotfix) git switch main
Switched to branch 'main’

➜  research-notes git:(main) git cherry-pick 574dead
[main 6c8b6a8] correct temperature value
 Date: Fri May 22 13:54:53 2026 +0200
 1 file changed, 1 insertion(+)
```

Cherry-picking and selective history

```
➜  research-notes git:(main) cat notes.txt
Experiment 1
Temperature: 20C
Result: pending
Humidity: 45%
Research phase: initial testing
Corrected temperature: 21C
```

Mistakes happen ¯_(ツ)_/¯

- Delete a commit accidentally
- Reset the wrong branch
- Lost changes
- **Panic**

Recovery

```
➜  research-notes git:(main) git reset --hard HEAD~1
HEAD is now at 2d86fc7 document research phase

➜  research-notes git:(main) cat notes.txt
Experiment 1
Temperature: 20C
Result: pending
Humidity: 45%
Research phase: initial testing
```

Recovery

```
➜  research-notes git:(main) git reflog

➜  research-notes git:(main) git reset --hard HEAD@{1}

➜  research-notes git:(main) cat notes.txt
Experiment 1
Temperature: 20C
Result: pending
Humidity: 45%
Research phase: initial testing
Corrected temperature: 21C
```

Taking a glance into the future

- Fork workflows
- Pull / Merge requests
- Protected branches
- Code review workflows
- CI/CD pipelines

Questions!