Lab 7: Using Git in a Team

GW CS 2541: Database Systems and Team Projects - 2026

Prof. Gabe Parmer, Emil Abbasov, Bella Dayrit, Sydney Berritt, Max Eichholz, Aditya Arjun

Announcements

1. The git logs and readMe for all 3 activities will be graded.
2. All project reminders will be on the website. You will regret not starting the project early.

Why learn Git? Why do we care?

- Your typical git workflow at this point
  - Pull and push to a single main branch
- Real world gitflow is much more complicated D:
  - Every SWE position will involve teams. You need this skill
- This should go on your **resume**, and will come up in interviews
  - Most directly determines how much your team will appreciate you (return offer or nah)

Working in teams

- New Workflow
  - Create a new branch for each feature (ex. Order cost function)
  - Each team member implements a feature on that branch, pushing and pulling as per usual
  - Branches get merged into main
  - Merge conflicts are resolved
- Main represents the unified consistent base
- Feature branches allow for evolution that does not disrupt the workflow of your teams

Activity 0: Comparing workflows

5 Minutes

- Discuss in tables:
  1. How is the new flow different from how you typically use git/github?
  2. Why do you think the additional complexity is "worth it" and common?
- Further reading here

Incoming Git-uations You'll Find Yourself In

- What if two people make changes to the same line of code?
  - Git doesn't know which change is "correct"
  - So we must choose for it
  - If we are working with additional people on the same project, this problem will happen much more often.
- The standard CS curriculum doesn't expose these ideas till your Senior Capstone Design, but by then, you'll want to have work experience

Useful commands

- See Git cheat sheet for a more detailed description

List existing branches

```
git branch -a
```

Make a new branch **&** switch to it

```
git checkout -b new_branch
```

Switch to main branch

```
git checkout main
```

View branch's staged & unstaged changes

```
git status
```

Get latest version from parent branch

```
git pull
```

Get latest version from different branch

```
git pull origin main
```

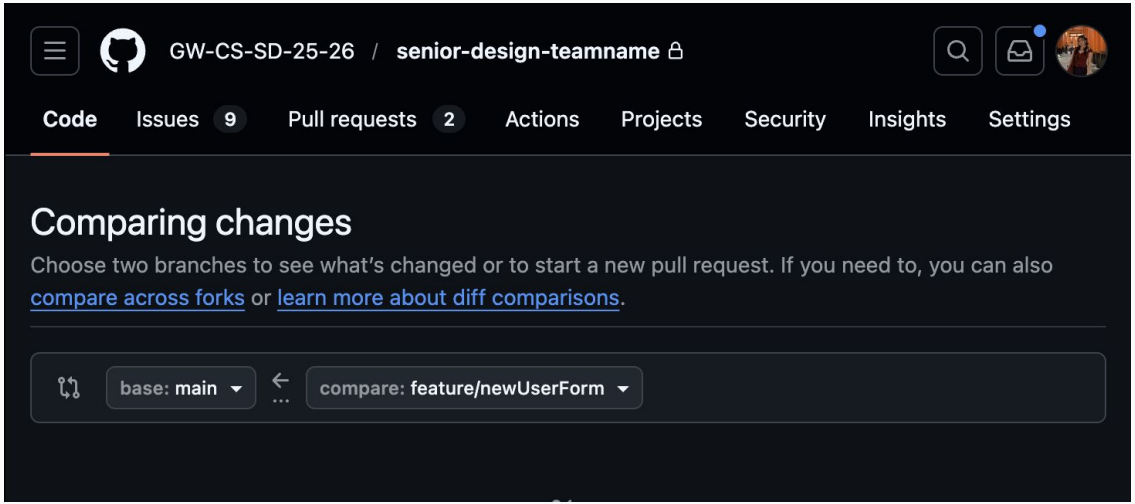
Activity 1: Working on separate branches

20 Minutes

- See README in the git classroom for instructions.

Default Git Merge/ Pull Request Workflow

- When you want to merge your feature branch to main, prepare your pull request on GitHub's interface
- Because we want to merge WITH main (add your feature into the main branch), the base should be main



Preparing your merge here, will allow you to handle merge conflicts in GitHub's visual code editor because github will only allow it if it can produce it cleanly

Remember to locally checkout into your main branch to pull these changes locally after completing your merge !

Activity 2: Using Git Merge

20 Minutes

- See README in the git classroom for instructions.

Next: Using git pull --rebase origin main

The workflow behind this command is what's most commonly used in real world software engineering teams! This command is run whenou're on your feature branch in the git command line

What does git pull –rebase origin main do?

1. It identifies your feature branches unique commits
2. Moves your branch pointer to the latest main
3. Reapplies your unique commits one by one as new commits on top of the latest version of main

Keywords:

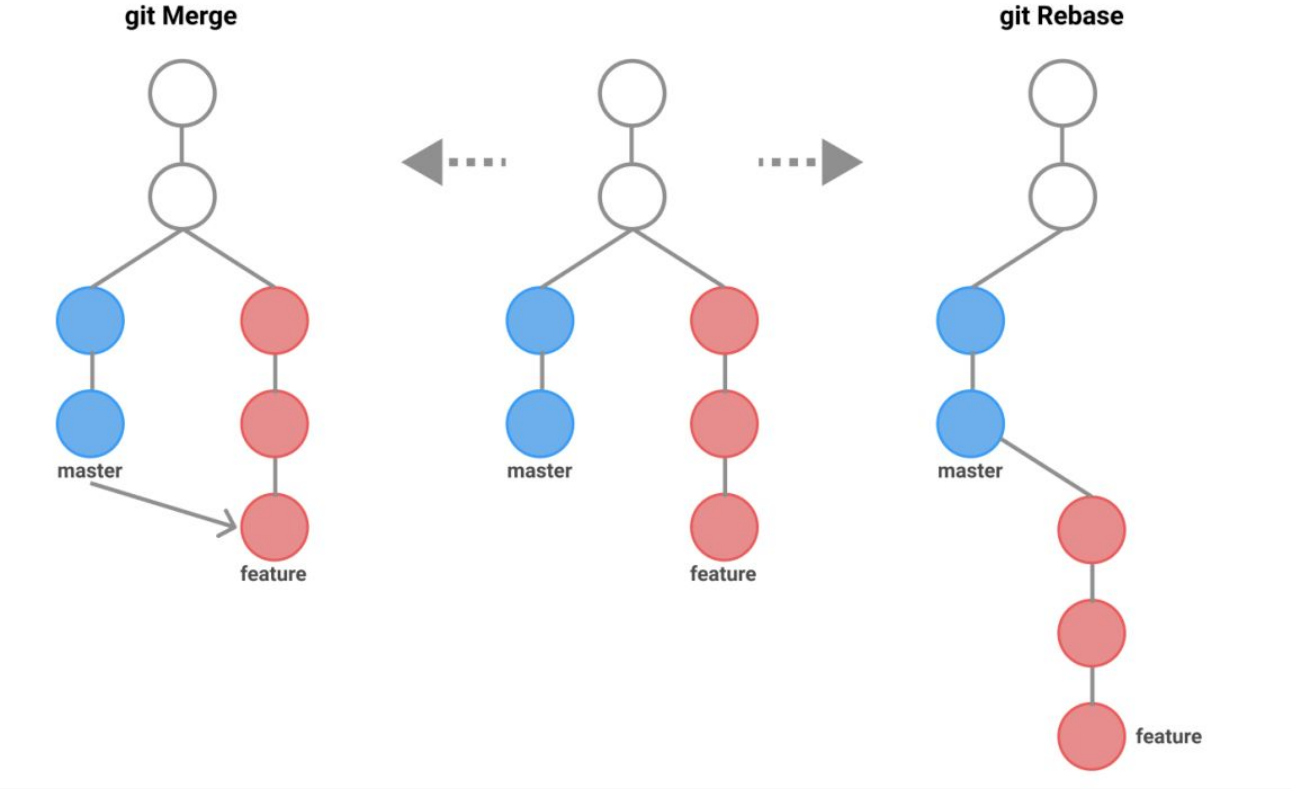
- Origin: the remote repository (IMPORTANT to ensure we're rebasing our REMOTE main)
- Main: the branch on the remote that we're updating from

Why use git pull –rebase origin main

Benefits:

1. Cleaner linear history
2. Easier to debug when looking back at previous versions of your code repository
    a. Eg. a team member integrated their feature to main but this has resulted in the whole application breaking
3. Prepares you for professional software engineering

Git merge feature-branch vs. Git pull --rebase



Edge Cases: git pull --rebase origin main

What about when we run into a merge conflict when you're running the git pull —rebase origin main command?

> You run: git pull --rebase origin main
> You see: CONFLICT (content): Merge conflict in file.js
> error: could not apply abc1234...

What You Do Next:

1. Fix the conflict in the file in VS Code
2. Git add the file.py that you just fixed
3. Run: git rebase —continue
    a. Tells git you resolved the conflict and to continue applying your commits to main

Activity 3: Using Git pull –rebase

20 Minutes

- See README in the git classroom for instructions.
- Stop by to see a TA before leaving when you are finished