EPFL

EMBEDDED SYSTEMS LABORATORY

Introduction to git part 4

EPFL - Embedded Systems Laboratory

2025

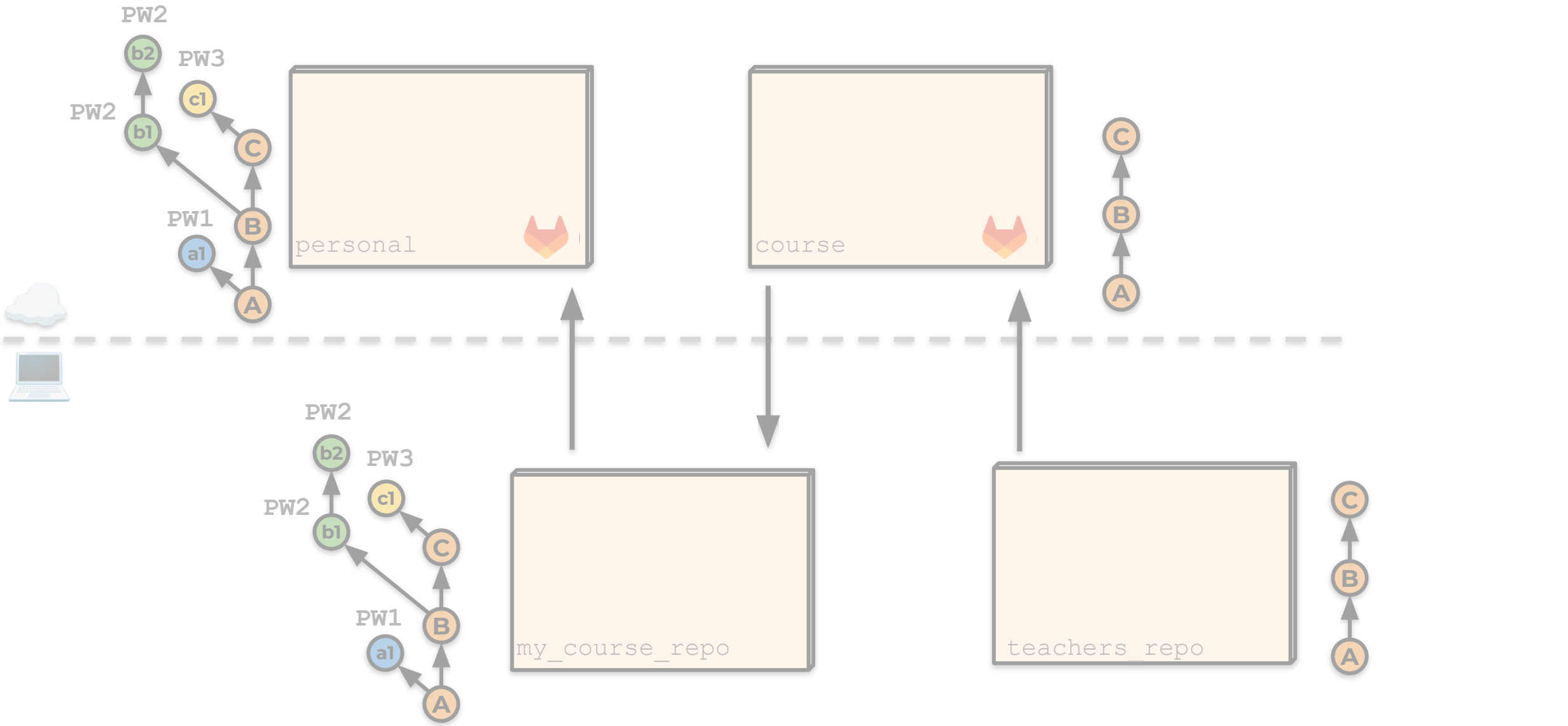
Introduction to git
introduction and basic use

Working alone
for any project you have

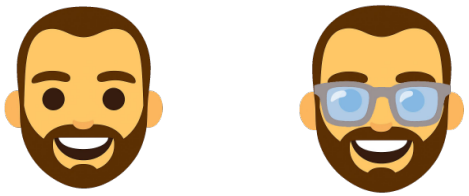
Using other’s work
e.g. the practical works

Working with others
e.g. your final project

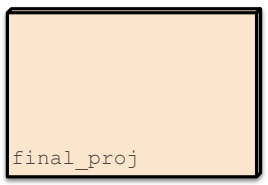
↩Handling the PWs



Handling a Project

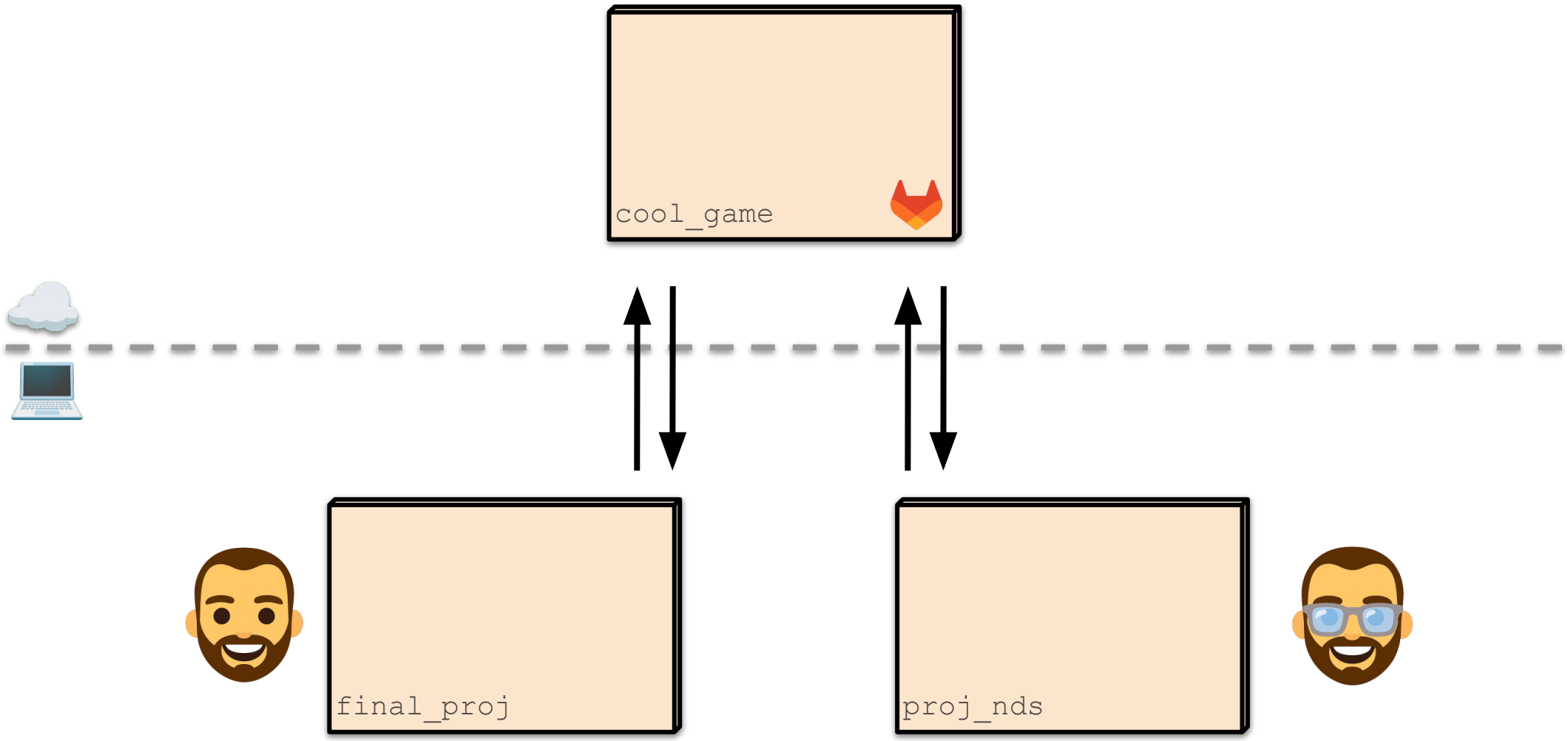


Handling a Project



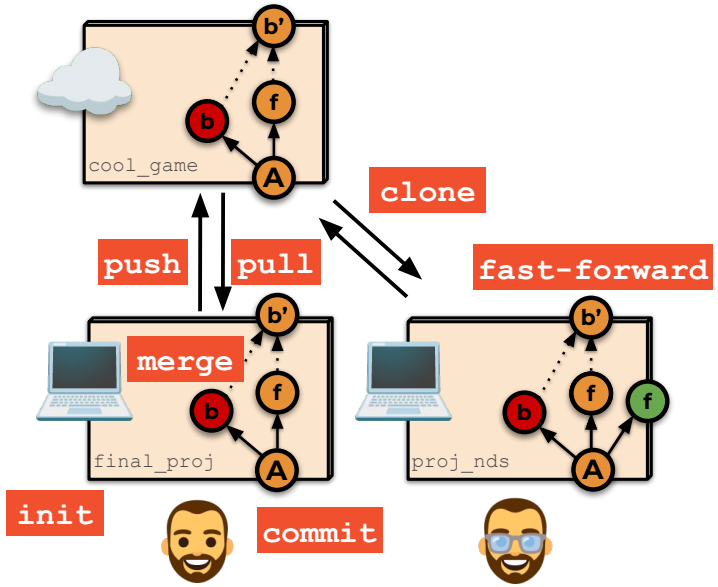
Handling a Project

Handling a Project



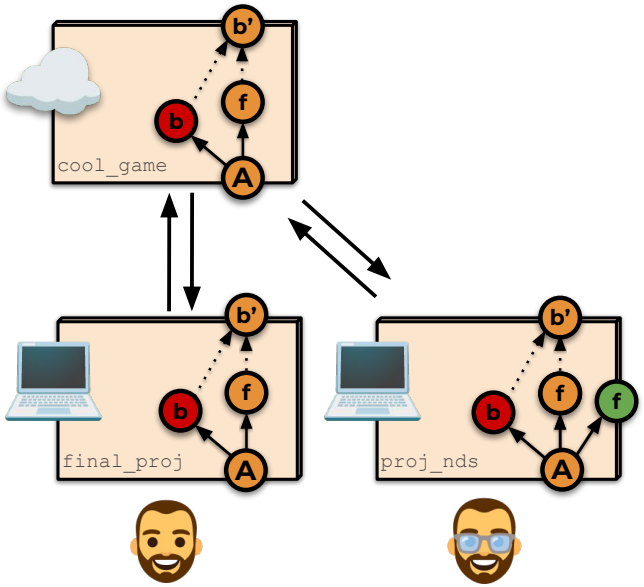
Handling a Project

Sharing a single remote repo, merging locally

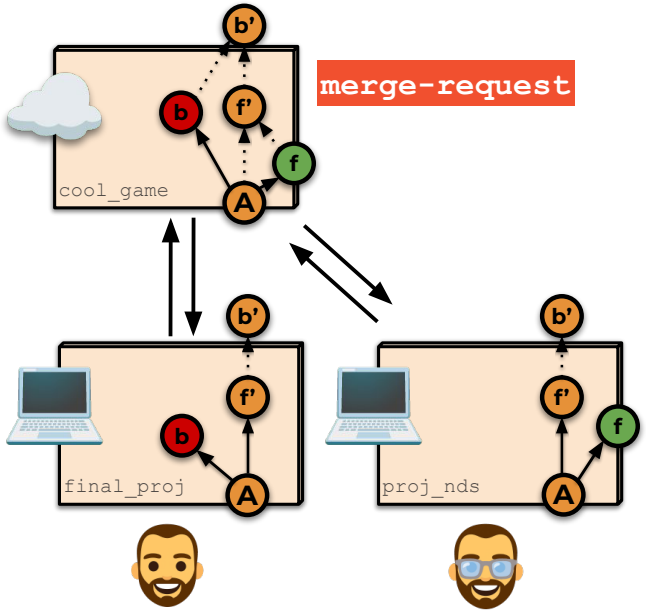


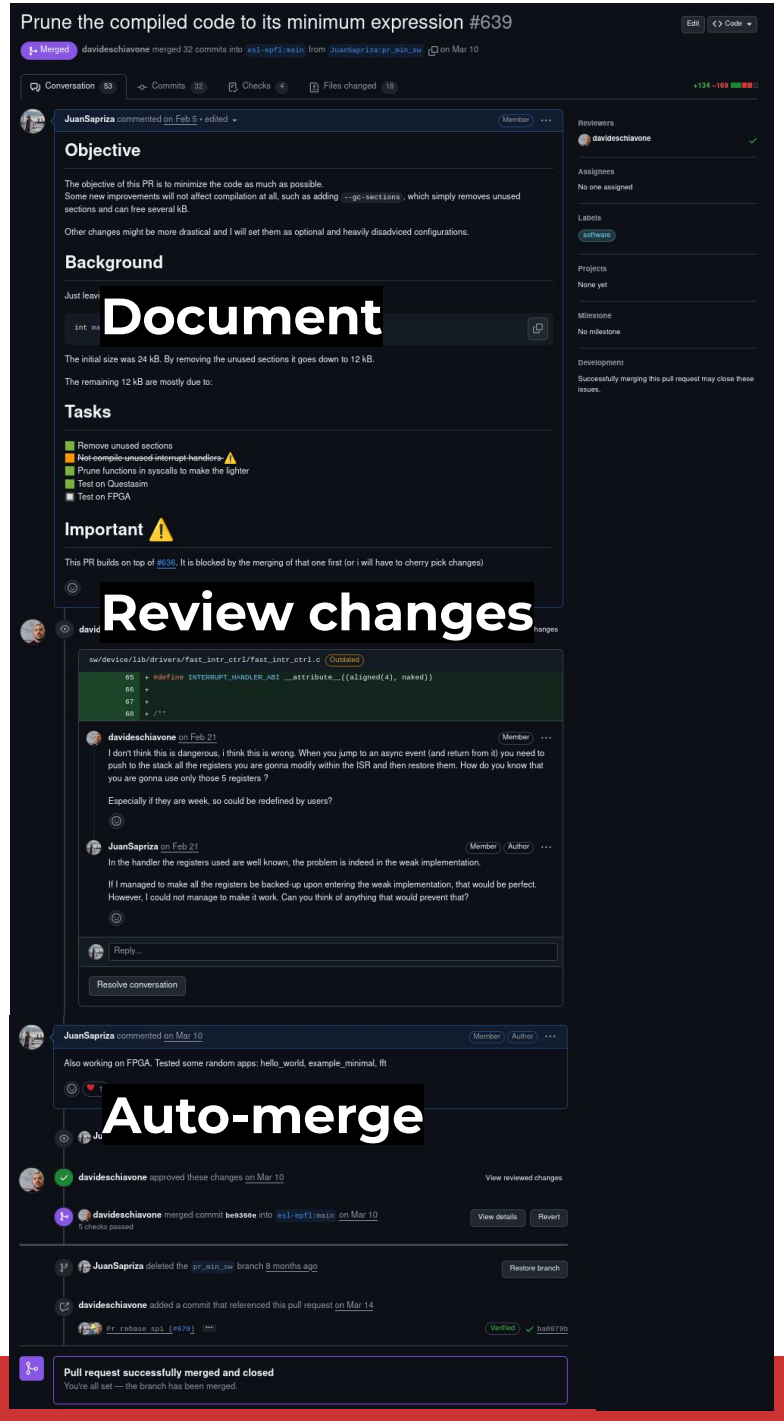
Handling a Project

Sharing a single remote repo, merging locally



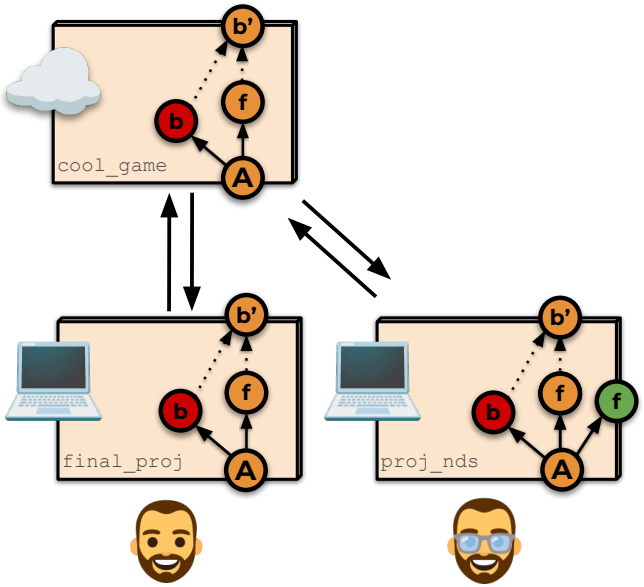
Sharing a single remote repo, merge requests



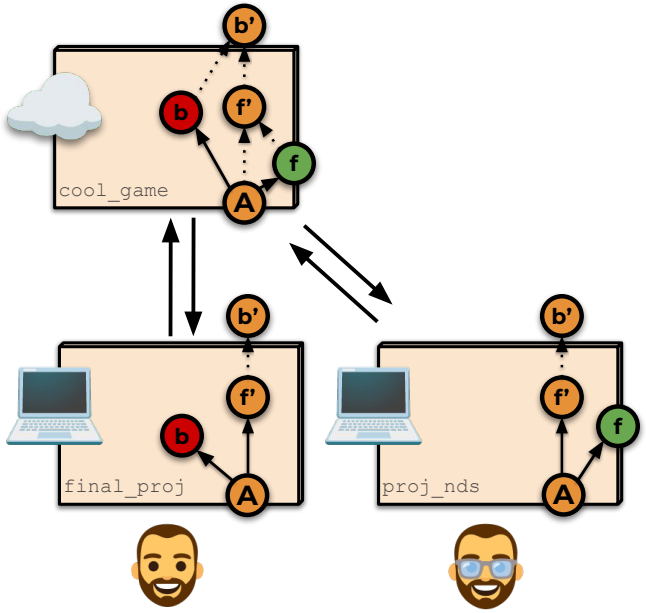


Handling a Project

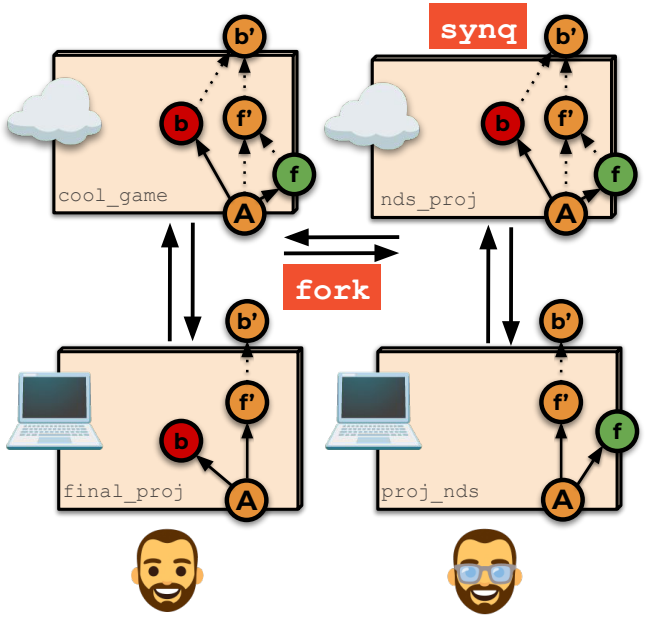
Sharing a single remote repo, merging locally



Sharing a single remote repo, merge requests

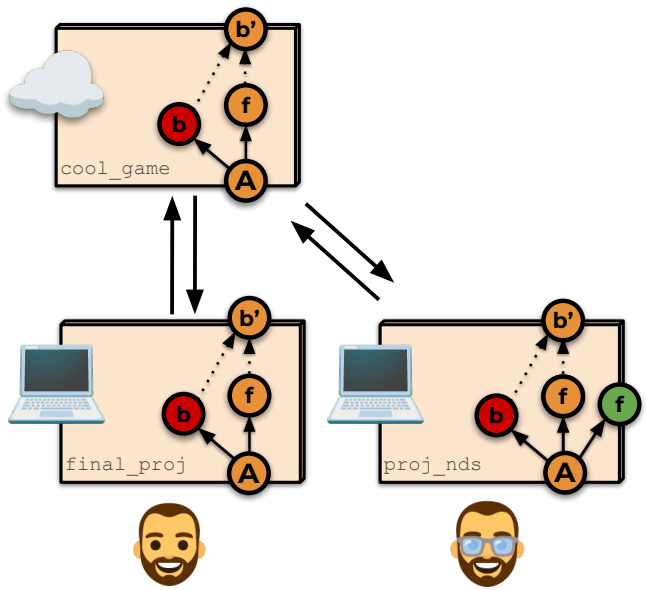


Fork of main repo, merge requests



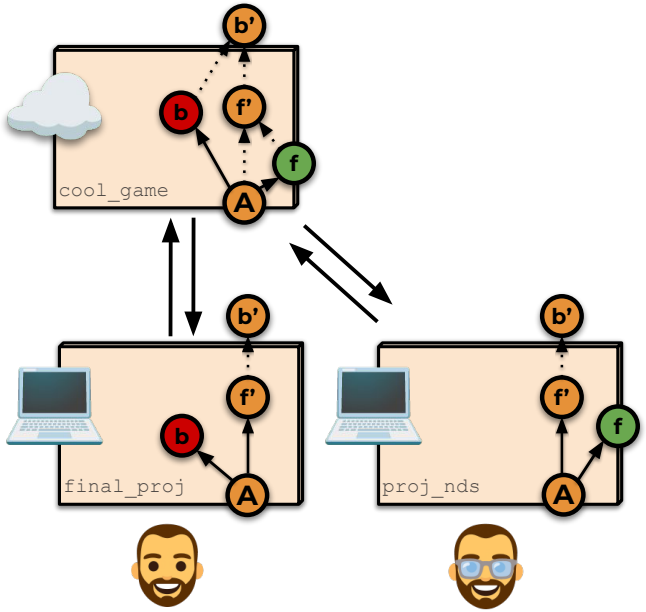
Handling a Project

Sharing a single remote repo, merging locally



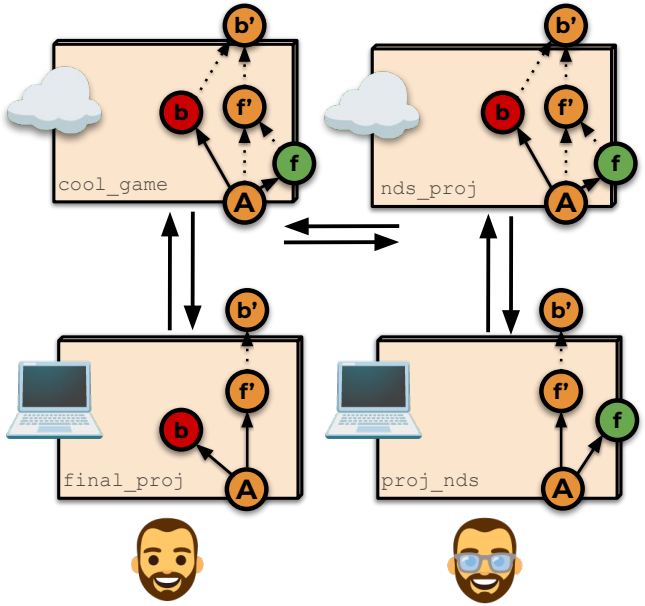
Lots of manual merging

Sharing a single remote repo, merge requests



Manual merging only for conflicts

Fork of main repo, merge requests



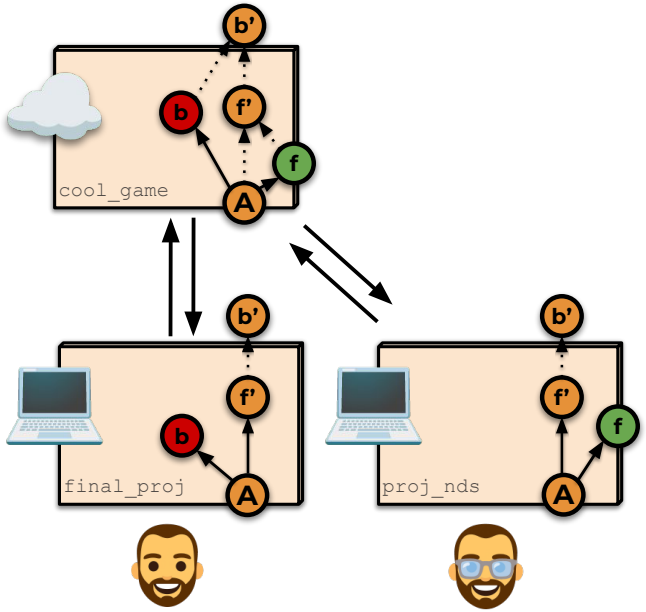
For larger projects

Demo

Ignoring files

- Share a repo
- Open an MR
- Reviewing an MR
- Merging an MR
- Solving conflicts

Sharing a single remote repo, merge requests



Manual merging only for conflicts

Demo - cheatsheet

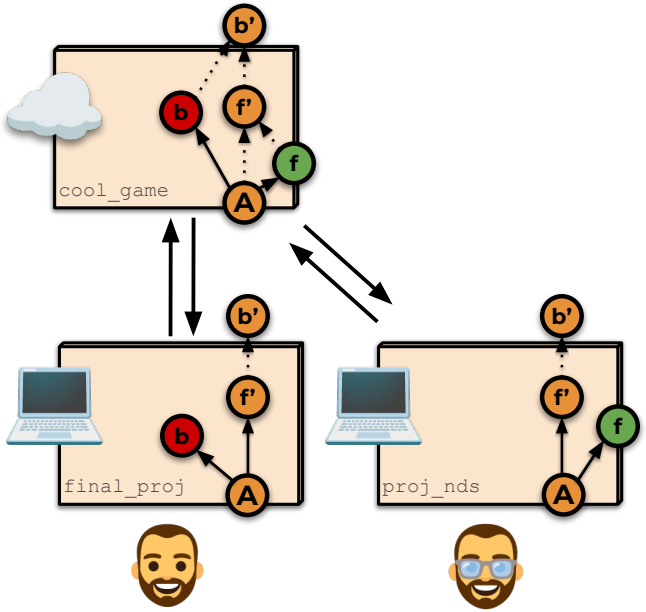
```
        # Create the local repo
mkdir Juan                          # Create a new folder
git init                            # Turn it into a repo

        # Create the remote repo
# New proyect/repository
# Create Blank repository
# Uncheck add Readme
# ADD MEMBERS! Manage > Members
# Code > copy SSH URL

        # Add a commit and push
git remote add origin <url>
echo hello > a.txt
git add a.txt
git commit -m “First commit”
git push origin main

        # Clone the repo as someone else (“Juano”)
cd ../
git clone <url> Juano
git checkout -b feature
echo bye > b.txt
git add b.txt
git commit -m “Added a feature in b”
git push origin feature             # Check how a new branch appears in the remote

        # Create a Merge Request
# Ctrl+click on the link to create MR
# Mark as draft and add a description
# Create it

cd ../Juan/
# commit, restore or stash changes
git checkout -b feature origin/feature
# You can commit and push, but be careful! Juano will need to pull these changes to continue working.
# Add comments or and a review

# Add another commit and push
echo Good byeee > b.txt
git commit -am “Emphasized”
git push origin feature

# Check changes, mark as ready, and merge!

        # See the effects of merging remotely
git fetch
git log --all --oneline --graph     # Note that main is behind, and that a merge was performed remotely
git checkout main
git pull                            # Update main branch

        # Do another MR, but using squash commit
git checkout -b other_feature
# Do 2 commits on this branch
git push origin other_feature

# Create MR
# Accept and squash commits

git log --all --oneline --graph     # Note that changes are missing
git fetch
git log --all --oneline --graph     # Note that changes in “Other” are not the ones merged!
                                    # Also note that the feature branch was not deleted??
git fetch --prune
git log --all --oneline --graph

        # Make changes as the other colleague
cd ../Juan
git fetch
git log --all --oneline --graph     # Notice how behind we are
git checkout -b someother_fefature
echo something > b.txt              # Add a conflicting change
echo something else > c.txt
git add b.txt c.txt
git commit -m “some cool feature”
git push origin someother_feature

# Create a MR
# Conflicts will prevent the merging: accept “theirs”

git fetch –prune
git log --all --oneline --graph     # Notice how the remote merged main on someother_feature, and viceversa
```

Demo - try it yourself

Ignoring files

- Share a repo
- Open an MR
- Reviewing an MR
- Merging an MR
- Solving conflicts

Sharing a single remote repo, merge requests



Manual merging only for conflicts

Thank you!

EPFL - Embedded Systems Laboratory