

```
> "I committed locally but haven't pushed yet. I want to combine the last two commits into a single commit with a new message. suggest what i can write
• The user is asking for help with git. They want to:
  1. Combine the last two commits into one
  2. Write a new message for the combined commit

This is a git workflow question. They haven't pushed yet, so they can safely use interactive rebase or git reset to combine commits.

The user is asking for suggestions on:
  1. How to combine the last two commits
  2. What message to write

I should:
  1. First, show them their last two commits so I can see what they contain
  2. Suggest a git command to combine them (interactive rebase is the standard approach)
  3. Suggest a commit message based on what the commits are about
```

- To combine your last two commits into one, use **interactive rebase**:
`git rebase -i HEAD~2`
This will open an editor. Change the second commit from **pick** to **squash** (or **s**), save, and then you'll be prompted to write a new combined message.
For the commit message, consider following this format depending on what you changed:
 - If it's a feature: **feat: <short description of what was added>**
 - If it's a bug fix: **fix: <what was fixed>**
 - If it's a refactor: **refactor: <what was improved>**
 - General format: **<type>: <concise description>**Add a body if needed:
`feat: add user authentication system`
 - Implemented JWT-based login
 - Added password hashing with bcrypt
 - Created session managementQuick command if you want to be more direct:
`git reset --soft HEAD~2
git commit -m "Your new combined message"`
What type of changes are in your two commits? That'll help me suggest a more specific message.

.The response is highly effective, offering two accurate and separate solutions: an interactive rebase (the recommended, secure method for squashing commits) and a soft reset (a quick alternative for merging recent commits). A valuable addition is the explanation of commit message conventions, providing the user with practical flexibility based on their Git proficiency.