

Git Cheat – Sheet

Use this document in the event you need to fix a commit message. But please be careful when committing changes and make sure you commit message is in line with PM's practices to avoid having to fix it again once the PR is open and/or in review.

Before you do anything, you need to have the latest develop branch on your local machine:

```
git checkout develop  
git pull
```

Then checkout your feature branch, check on github in the commit tab how many commits your branch has, and how far back your bad commit is, let say it's the 3rd one from the most recent commit. Then you need to rebase the last 3 commits:

```
git rebase -i HEAD~3  //3 here is the num of commits to go back
```

Once you run this command, a text as follow should open

```
pick 4557759 contact info form (#62)  
pick 1978e68 implement contact information form layout  
pick 8987f36 fix unit tests
```

To fix a commit message, you need to replace the word “pick” with “reword”. Also if you see any commits with a number between parenthesis, like the first one on the snippet above, those belong to other PRs and need to be removed by replacing “pick” with “drop”.

Once you save and close this file. You will see a file open for each commit that needs to be fixed where you can update the commit messages. Once you are done, it should complete the rebase and output a message “rebase completed successfully”.

Now that bad commit messages have been fixed, you can either merge develop to your branch locally or clicking the merge button in github after pushing your changes.

To you push your changes, you need to use the `--force` flag, since no code change was done. Otherwise, git won't push and will display a message saying that no changes were made that need to be pushed.

```
git push --force
```

Note, if you run into any permission error trying to push with `--force` flag, please contact your team lead and they will escalate to client. Everyone should have permission to push with `--force` flag.

Before doing any development, please make sure to go over the GPG setup successfully to make sure all your commits are signed.

But, if you need to sign passed commits on an existing PR/Branch, run this command:

```
git rebase --exec 'git commit --amend --no-edit -n -S' -i  
develop
```

This will sign any unsigned commits all the way to develop, but you can replace develop by a commit hash to sign commits up to (but excluding) that commit.

If you need to sign a single commit, just run this command:

```
git commit --amend --no-edit -S <commit hash>
```

Then push with `--force` flag, as show above.