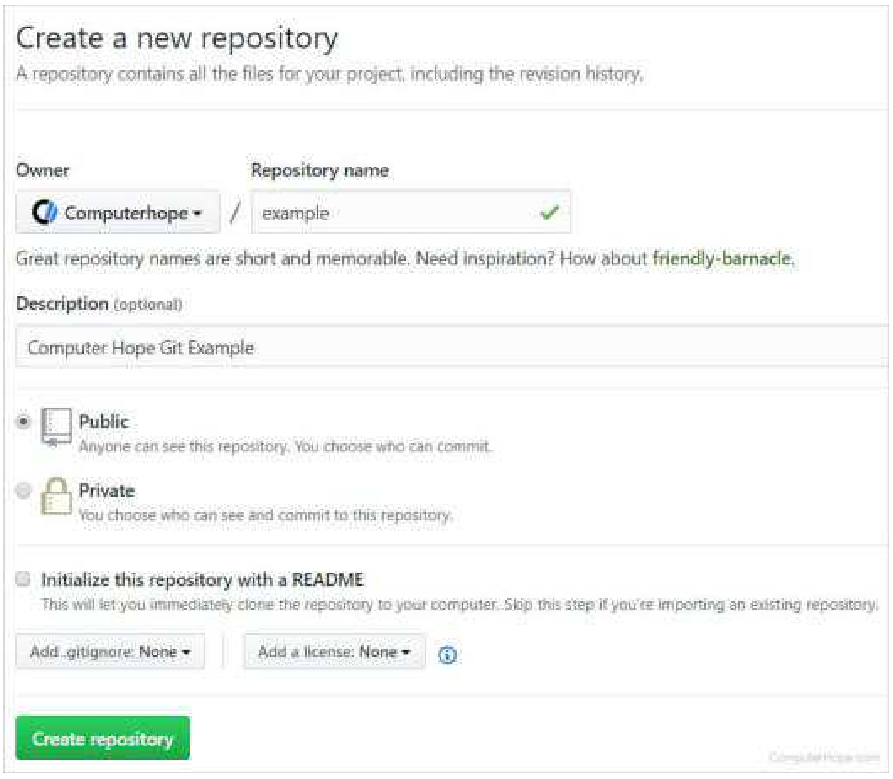
The Github Workflow a.k.a. The GitHub Flow

Admin:

1. Creates a new repo with branches ‘master’ and ‘dev’ on Github
2. Clones the Github repo (git clone https://github.com/…)
3. works on the local copy
4. Creates a react project (or a vanilla project with just the files index.html index.js style.css)
5. add comments for sections in the files, e.g. style.css: /* nav start */ /* nav end */ /* main start */ /* main end */
6. Adds all files, commits and pushes to Github (git add .; git commit -m “initial”; git push
7. adds the collaborators on Github
8. accepts the pull requests by the collaborators



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: Computerhope ▾ / Repository name: example ✓

Great repository names are short and memorable. Need inspiration? How about friendly-barnacle.

Description (optional): Computer Hope Git Example

[✓ Public] Anyone can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Collaborators:

1. copy the repo link from Github to clone the repo - git clone https://github.com/…

2. git checkout -b “nav” (create my feature branch nav)

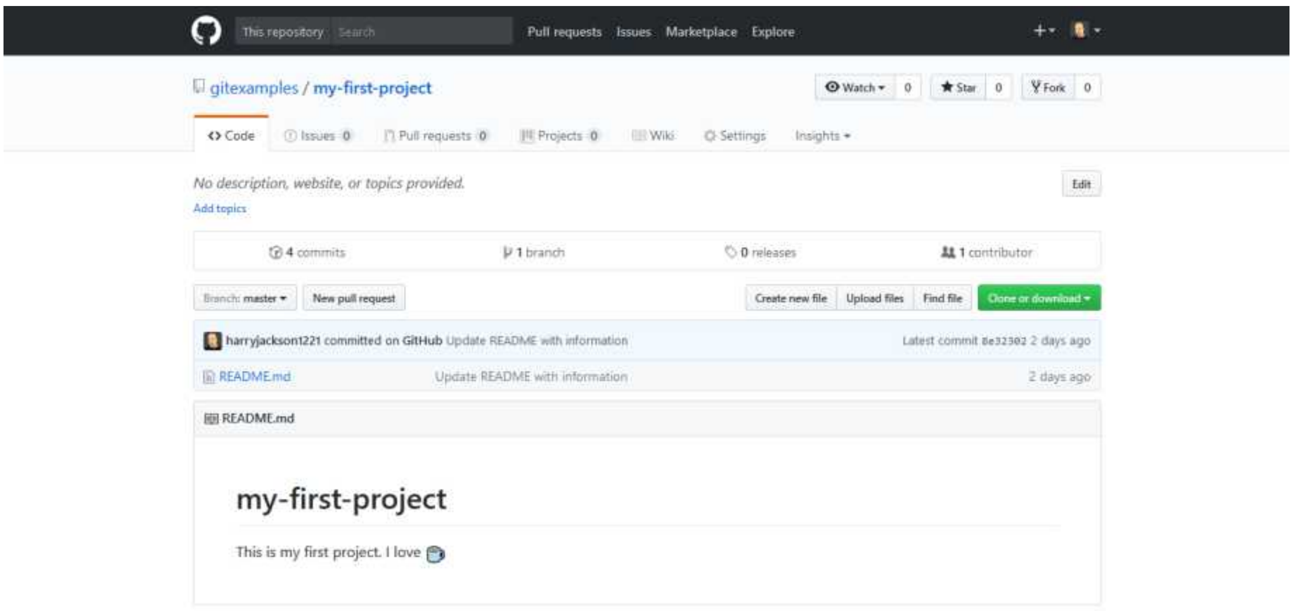
3. git add .   git commit -m “init feature branch”   git push (error message because branch does not exist yet on Github repo)

4. git push --upstream … (push and create branch on Github)

5. pull the start working on my feature branch (nav)

6. git add .   git commit -m “added navbar”   git push

7. on Github: merge the new branch with the main branch



gitexamples / my-first-project

No description, website, or topics provided.

Add topics

4 commits | 1 branch | 0 releases | 1 contributor

harryjackson1221 committed on GitHub Update README with information   Latest commit 8e32302 2 days ago

README.md   Update README with information   2 days ago

README.md

my-first-project

This is my first project. I love

Git Cheat sheet SETUP

CONFIG

`git config --global user.name “[firstname lastname]”` set a name that is identifiable for credit when review version history

`git config --global user.email “[valid-email]”`
set an email address that will be associated with each history marker

`git config --global color.ui auto`
set automatic command line coloring for Git for easy reviewing

SETUP & INIT

Configuring user information, initializing and cloning repositories

`git init`

initialize an existing directory as a Git repository

`git clone [url]`
retrieve an entire repository from a hosted location via URL

STAGE & SNAPSHOT

Working with snapshots and the Git staging area

`git status`

show modified files in working directory, staged for your next commit

`git add [file]`
add a file as it looks now to your next commit (staging area)

`git add .`
add all modified files to the staging area

```
  git reset [file]
```

unstage a file while retaining the changes in working directory

`git diff`

diff of what is changed but not staged

`git diff --staged`
diff of what is staged but not yet committed

`git commit -m “[descriptive message]”`
commit your staged content as a new commit snapshot

BRANCH & MERGE

Isolating work in branches, changing context, and integrating changes

`git branch`
list your branches. a * will appear next to the currently active branch

`git branch [branch-name]`
create a new branch at the current commit

`git checkout -b [branch-name]` create a new branch and check it out

```
  git merge [branch]
```

merge the specified branch’s history into the current one

`git log`
show all commits in the current branch’s history

INSPECT & COMPARE

Examining logs, diffs and object information

`git log`
show the commit history for the currently active branch

`git log branchB..branchA`
show the commits on branchA that are not on branchB

`git diff branchB...branchA`

show the diff of what is in branchA that is not in branchB

`git log --follow [file]`
show the commits that changed file, even across renames

SHARE & UPDATE

Retrieving updates from another repository and updating local repos

`git remote add [alias] [url]` add a git URL as an alias

`git fetch [alias]`
fetch down all the branches from that Git remote

`git merge [alias]/[branch]`
merge a remote branch into your current branch to bring it up to date

`git push [alias] [branch], e.g. git origin main` Transmit local branch commits to the remote repository branch

`git pull`

fetch and merge any commits from the tracking remote branch