Lab - Git Essentials for a Devops Professional

Mission 04 - Git Essentials for a Devops Practitioner

Author: Gourav Shah

Publisher: School of Devops

Version : SFD105_v28.04.2024.01

With this lab you are going to learn the essentials of Git Workflow including ,

- How to add Global Configurations to Git
- How to start Revision Controlling your code with Git and GitHub
- How to check Logs, Status , Commit History, Staging Areas etc.
- How to get started with Git Branches and Remotes
- How to submit changes with Pull Requests Based Workflows
- How to enforce Branching Policies eg. Trunk Based Development Model
- How to Release your code using Semantic Versioning and by creating Tags.

Basic Git Operations

To begin learning Git, you need

- An account on GitHub. If you haven't yet signed up, begin by creating an account here GitHub: Let's build from here · GitHub.
- Git Client Installed. Refer to Git - Installing Git for installing git on to your system.

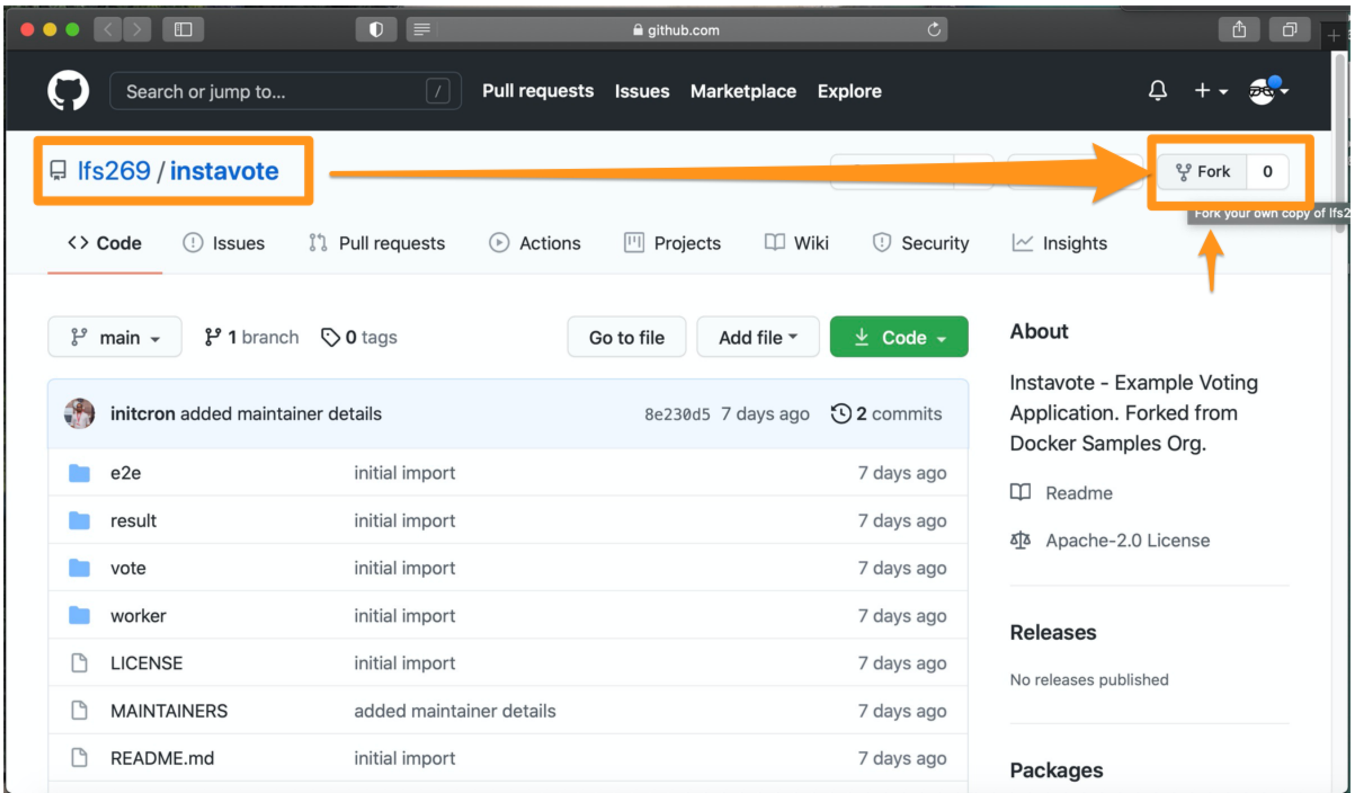
Begin by validating git client is installed by running

```
git
```

If you see an output with options to git utility, its installed on your system.

Forking and Cloning the Repo

Begin by creating a fork of GitHub - lfs269/instavote: Instavote - Example Voting Application. Forked from Docker Samples Org.



lfs269 / instavote

initcron added maintainer details 8e230d5 7 days ago 2 commits

e2e	initial import	7 days ago
result	initial import	7 days ago
vote	initial import	7 days ago
worker	initial import	7 days ago
LICENSE	initial import	7 days ago
MAINTAINERS	added maintainer details	7 days ago
README.md	initial import	7 days ago

About

Instavote - Example Voting Application. Forked from Docker Samples Org.

Readme

Apache-2.0 License

Releases

No releases published

Packages

Now clone the forked repository to your system as,

```
git clone https://github.com/xxxxxx/instavote.git
```

Where replace `xxxxxx` with actual GitHub account/org name that you forked the repo with.

Adding Global Configurations to Git

To set up configurations such as which user is committing to the repository etc. you would need to configure git , specifically add Global configuration (user specific).

To list existing config use,

```
git config
git config --list --global
```

To add essential configuration related to user,

```
git config --global user.name "Your Name"
git config --global user.email "name@example.com"
```

Ensure that the user name and email address match your GitHub account.

Validate the configurations by running,

```
git config --list --global
```

Basic Revision Control Operations

Change into the path where you have cloned the repository earlier. e.g.

```
cd instavote
```

```
ls
mkdir deploy
cd deploy
mkdir vote
cd vote
```

Now download the deployment and services code for vote app as,
`file: deployment.yaml`

```
wget -c https://gist.githubusercontent.com/initcron/
11ac44e45d94b8efc2fa7ca31e2162f7/raw/53c225de423465a54a1c72156e211232fbde90b6/
deployment.yaml
```

`file: service.yaml`

```
wget -c https://gist.githubusercontent.com/initcron/
954a91fcb49915cbbc56e395758d3469/raw/682d1a79e320a40dc28ff51894e7ffec57790113/
service.yaml
```

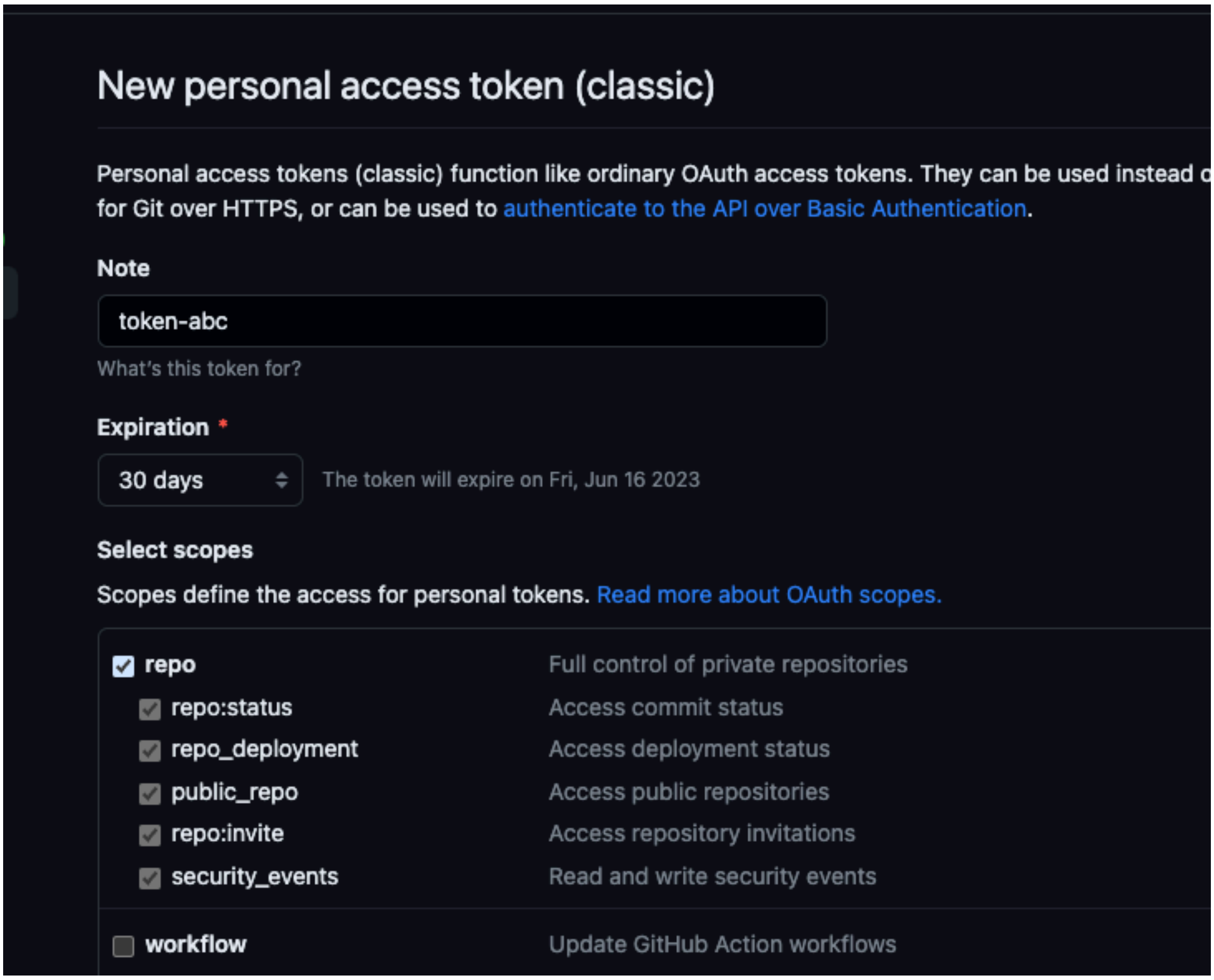
If you do not have `wget` installed, use the links above, copy over the files and add those to the location where you have created the vote/deploy folder as shown earlier.

Learn about Three Trees in Git and Start revision controlling these manifests (.yaml files) with git as,

```
cd ..
```

```
git status
git add *
git status
git commit -am "added deploy code for vote app"
git status
git log
git push origin main
```

When it asks for the password you should provide a token that you could generate from token's page Sign in to GitHub · GitHub as follows,



New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead o
for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note

token-abc

What's this token for?

Expiration *

30 days — The token will expire on Fri, Jun 16 2023

Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

✓ repo	Full control of private repositories
✓ repo:status	Access commit status
✓ repo_deployment	Access deployment status
✓ public_repo	Access public repositories
✓ repo:invite	Access repository invitations
✓ security_events	Read and write security events
☐ workflow	Update GitHub Action workflows

Ensure that you have provided name for the token and selected repo access. A token such as `ghp_JyHrBurgEUqJMPmEvbhvYblCg2gPux13dQ78` would be generated, which then can be used as a password. Do note this down somewhere securely if you want to use it later.

Validate the commit history by checking the log again as,

```
git log
```

Branching and Merging

To try creating new branch of development and switching to it as ,

```
git branch
git branch test
git branch
git log
git checkout test
git branch
git log
```

Alternately , you could have achieved this in one step using `git checkout -b test`

Its now time to add a file to the branch created above,

```
cd deploy
echo "This is a Deployment Code for Kubernetes" > README.md
git status
git add README.md
git status
git commit -am "added README for deploy code"
git status
git log
```

Try modifying the file now

```
echo "This code would be used by Flux to deploy to a kubernretes environment" >>
README.md

cat README.md
git status
git add README.md
git status
git commit -am "updated README"
git status
git log
```

To bring these changes into the main branch,

```
git checkout main
git merge test
```

To have it be reflected on GitHub, push the changes to it as,

```
git push origin main
```

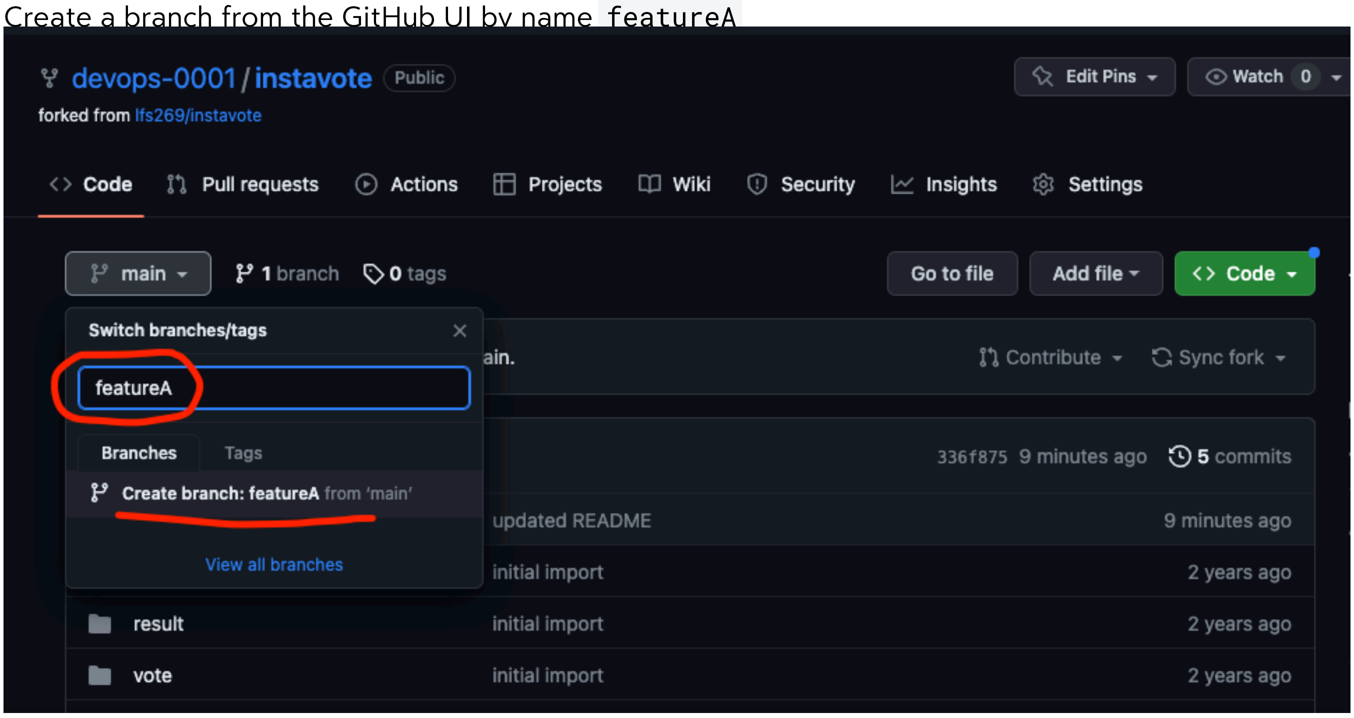
Ensure you are using token instead of password.

To delete the branch use,

```
git branch
git branch -D test
git branch
git log
```

Raising Pull Requests to Merge Changes

Create a branch from the GitHub UI by name `featureA`



On to your local workstation, pull the changes so that it starts tracking the remote branch, e.g.

```
# git pull origin
From https://github.com/devops-0001/instavote
 * [new branch]      featureA   -> origin/featureA
Already up to date.
```

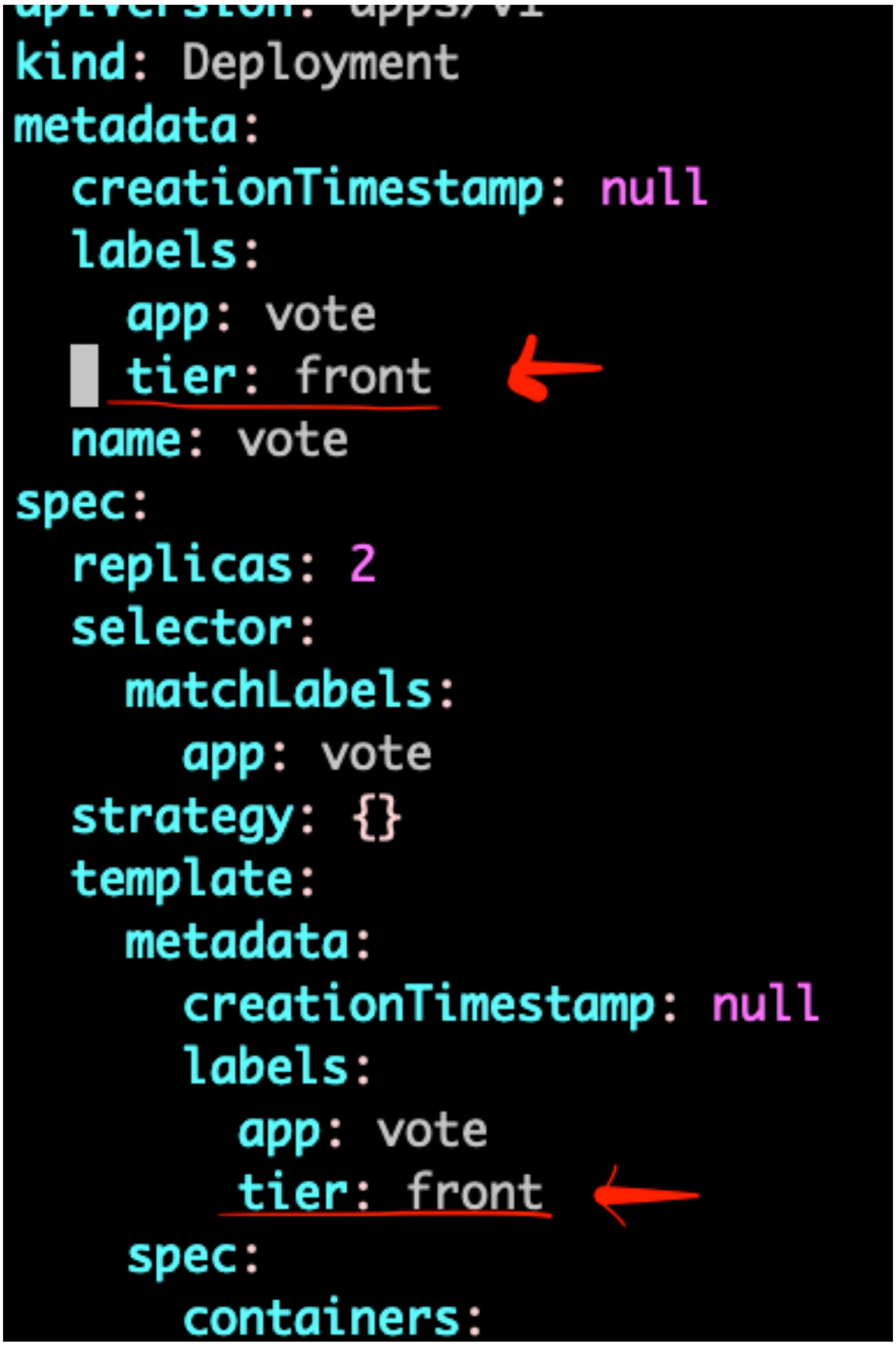
Switch to the newly created branch as,

```
git checkout featureA
git branch
```

Switch to `deploy` directory if not already

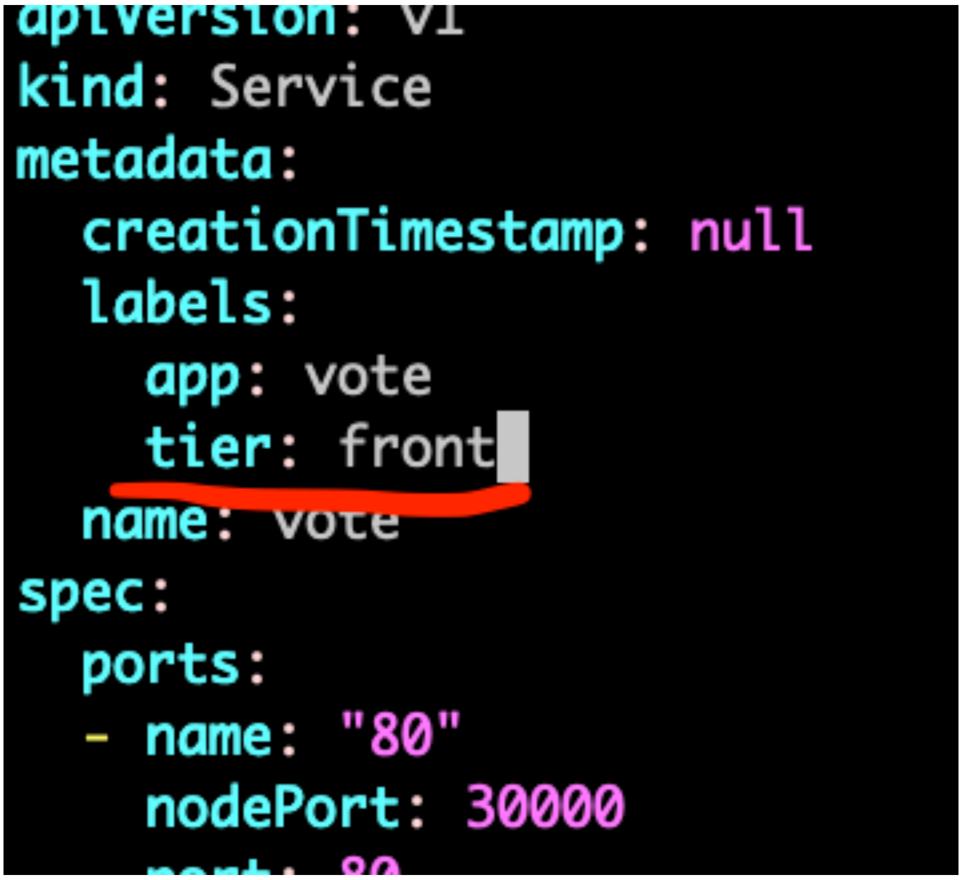
```
cd deploy
```

Edit `vote/deployment.yaml` to add labels as,



```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: vote
    tier: front
  name: vote
spec:
  replicas: 2
  selector:
    matchLabels:
      app: vote
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: vote
        tier: front
    spec:
      containers:
```

Edit `vim vote/service.yaml` to add labels as,

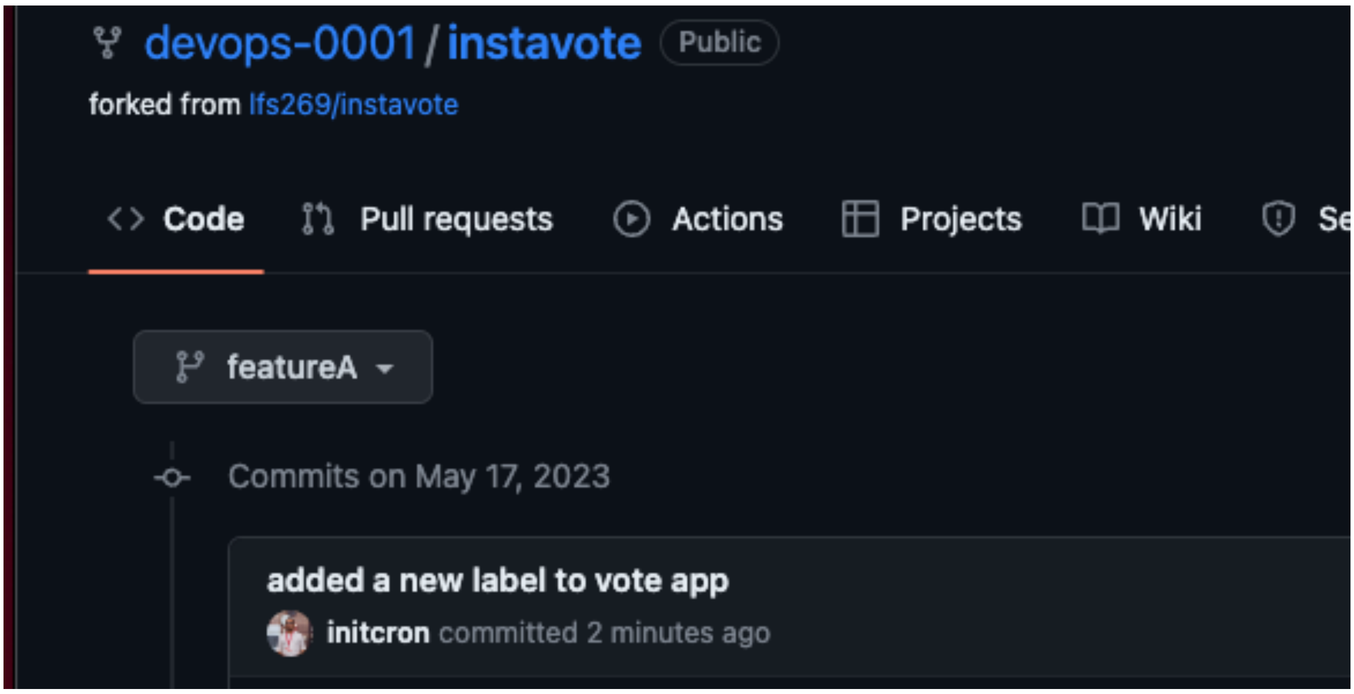


```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: vote
    tier: front
  name: vote
spec:
  ports:
  - name: "80"
    nodePort: 30000
    port: 80
```

Validate the changes

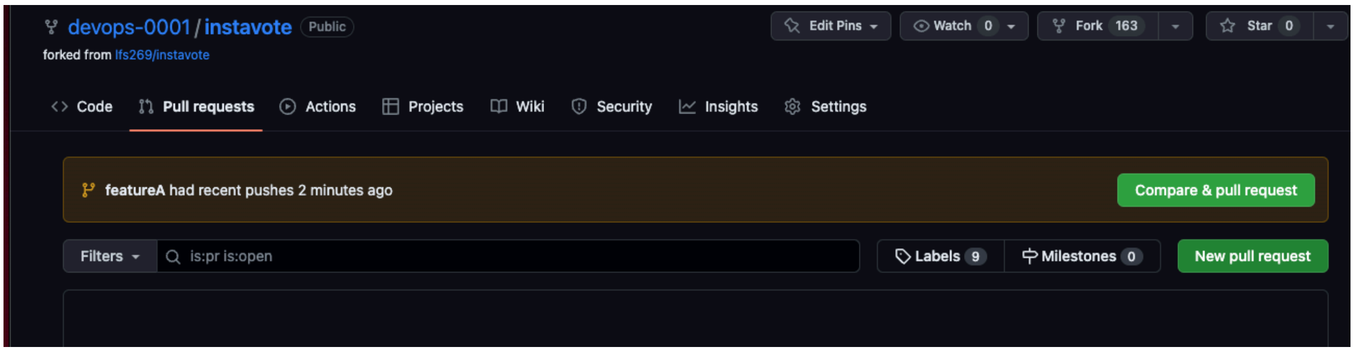
```
git diff
git log
git commit -am "added a new label to vote app"
git log
git push origin featureA
```

Verify the changes are available in the commit history as,


devops-0001 / instavote Public
forked from lfs269/instavote
Code
Pull requests
Actions
Projects
Wiki
featureA
Commits on May 17, 2023
added a new label to vote app
initcron committed 2 minutes ago

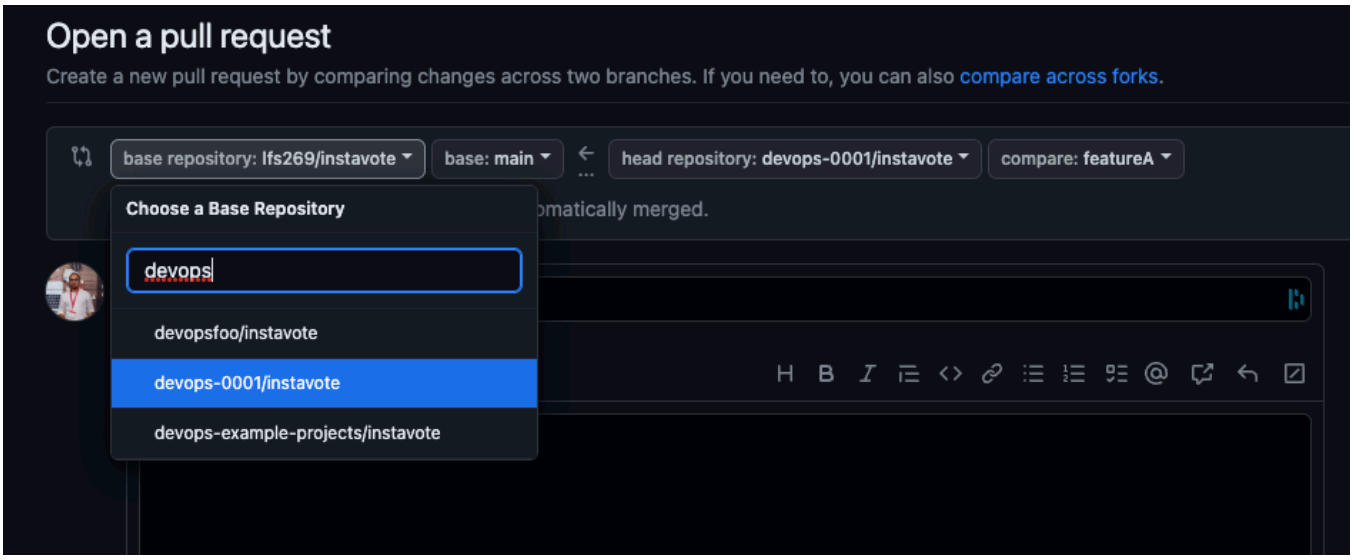
These changes are added to the feature branch as off now.

To bring these changes into the main line, raise a pull request by going to the `Pull Requests` tab and selecting `Compare and pull request` button.



One important consideration here is to select your own repository’s main branch while raising the pull request

e.g.


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base repository: lfs269/instavote · base: main · head repository: devops-0001/instavote · compare: featureA

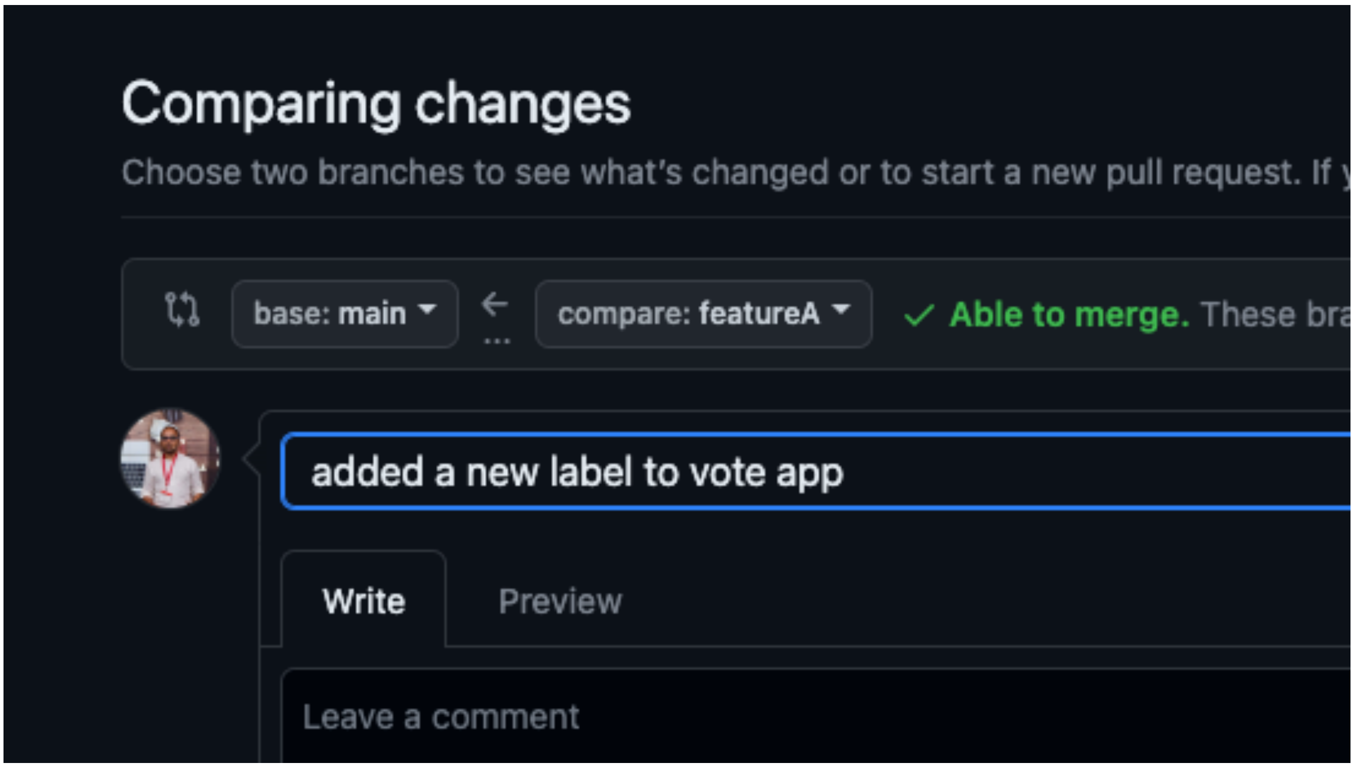
Choose a Base Repository

devops

devopsfoo/instavote

devops-0001/instavote

devops-example-projects/instavote

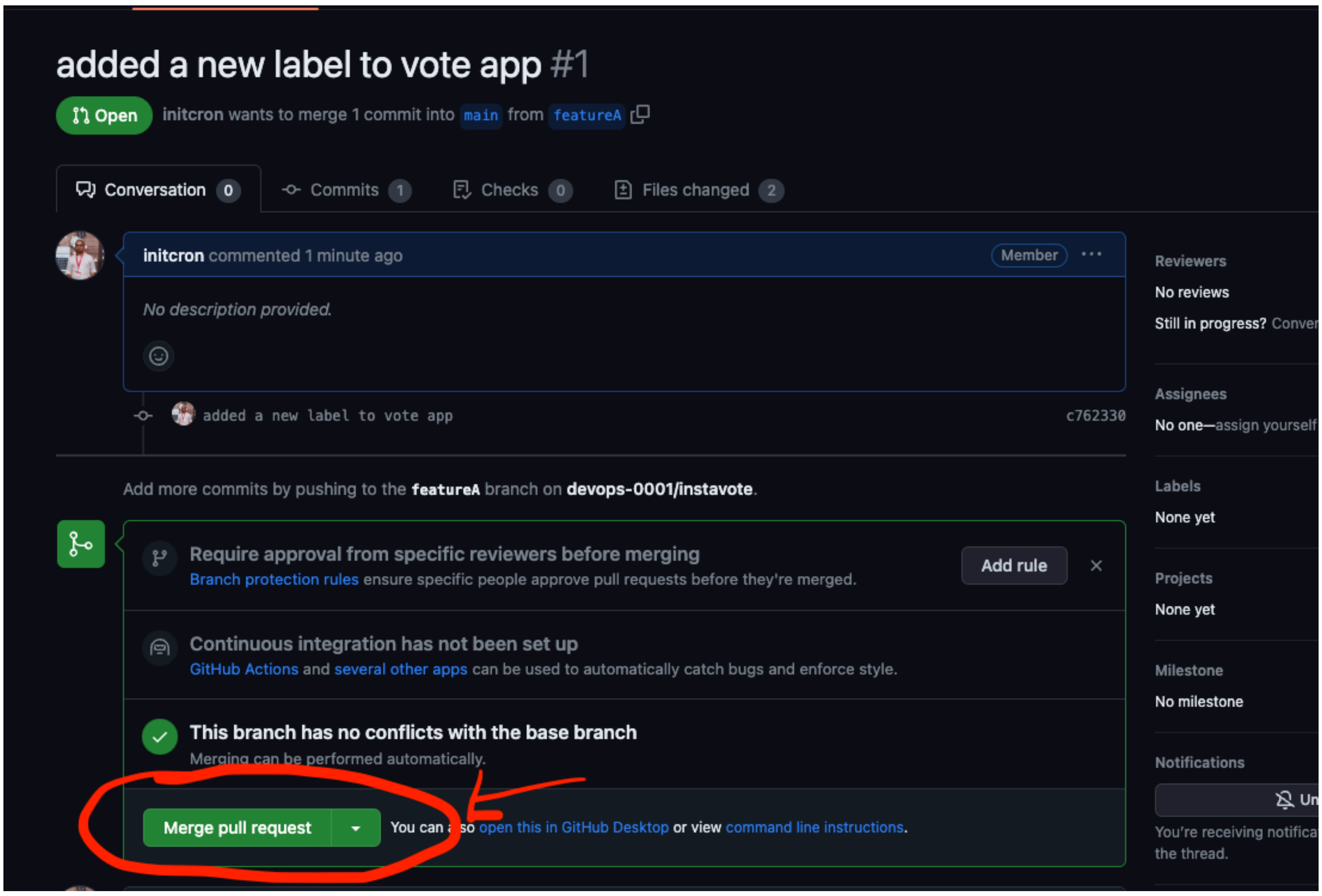

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If y

base: main · compare: featureA · ✓ Able to merge. These bra

added a new label to vote app

Finally, merge the pull request as,


added a new label to vote app #1

Open initcron wants to merge 1 commit into main from featureA

initcron commented 1 minute ago Member

No description provided.

added a new label to vote app c762330

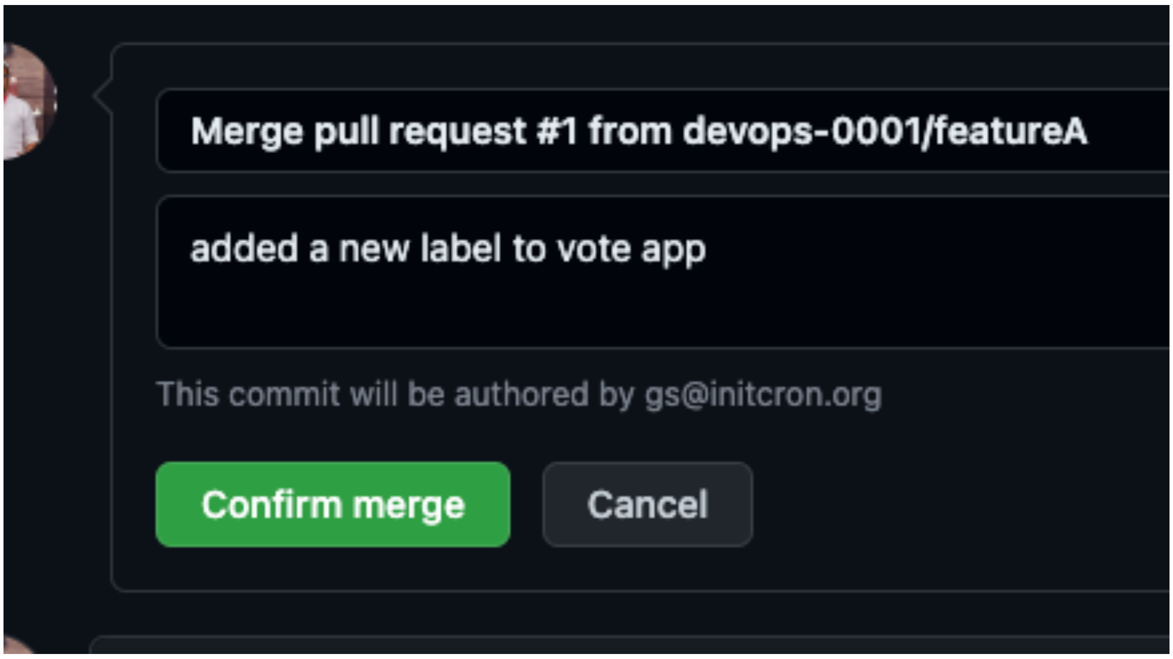
Add more commits by pushing to the featureA branch on devops-0001/instavote.

Require approval from specific reviewers before merging
Branch protection rules ensure specific people approve pull requests before they're merged.

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch
Merging can be performed automatically.

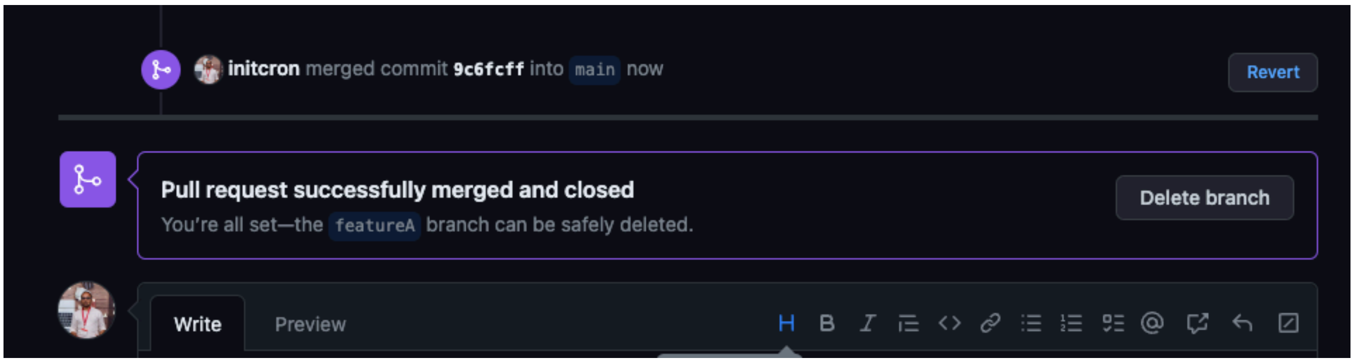
You can also open this in GitHub Desktop or view command line instructions.


Merge pull request #1 from devops-0001/featureA

added a new label to vote app

This commit will be authored by gs@initcron.org

After merging the changes, you could delete the feature branch


initcron merged commit 9c6fcff into main now

Pull request successfully merged and closed

You're all set—the featureA branch can be safely deleted.

To delete the branch locally and keep the code updated,

```
git checkout main
git branch -D featureA
git branch
git pull origin
```

Enforcing Trunk Based Development Model

Read about different branching models here

- Trunk Based Development Trunk Based Development
- GitHub Flow Understanding the GitHub flow · GitHub Guides
- Git Flow A successful Git branching model » nvie.com
- GitHub Flow vs Trunk Based https://trunkbaseddevelopment.com/alternative-branching-models/index.html#modern-claimed-high-throughput-branching-models

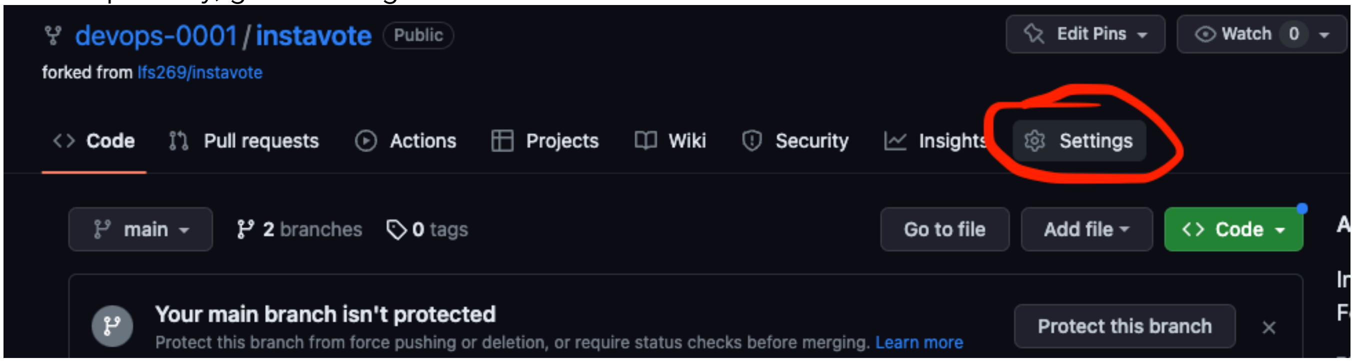
Making Changes Before Enforcing Branching Model

Before enforcing the Trunk Based Development Model, you can check if you are able to commit to the trunk (main branch) as,

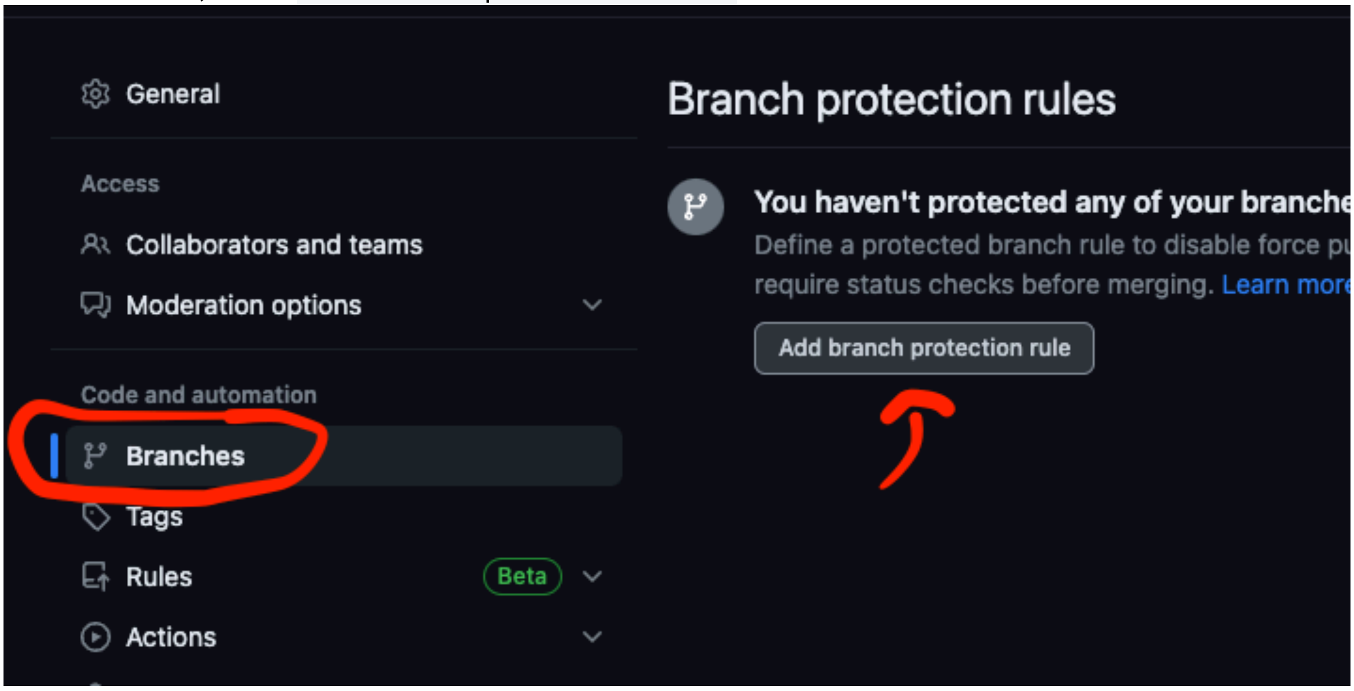
```
cd instavote/deploy
git log
git branch
echo "Test before enforcing branching model" >> README.md
git diff
git status
git commit -am "updating README"
git push origin main
git log
```

Adding Branch Protection Rule

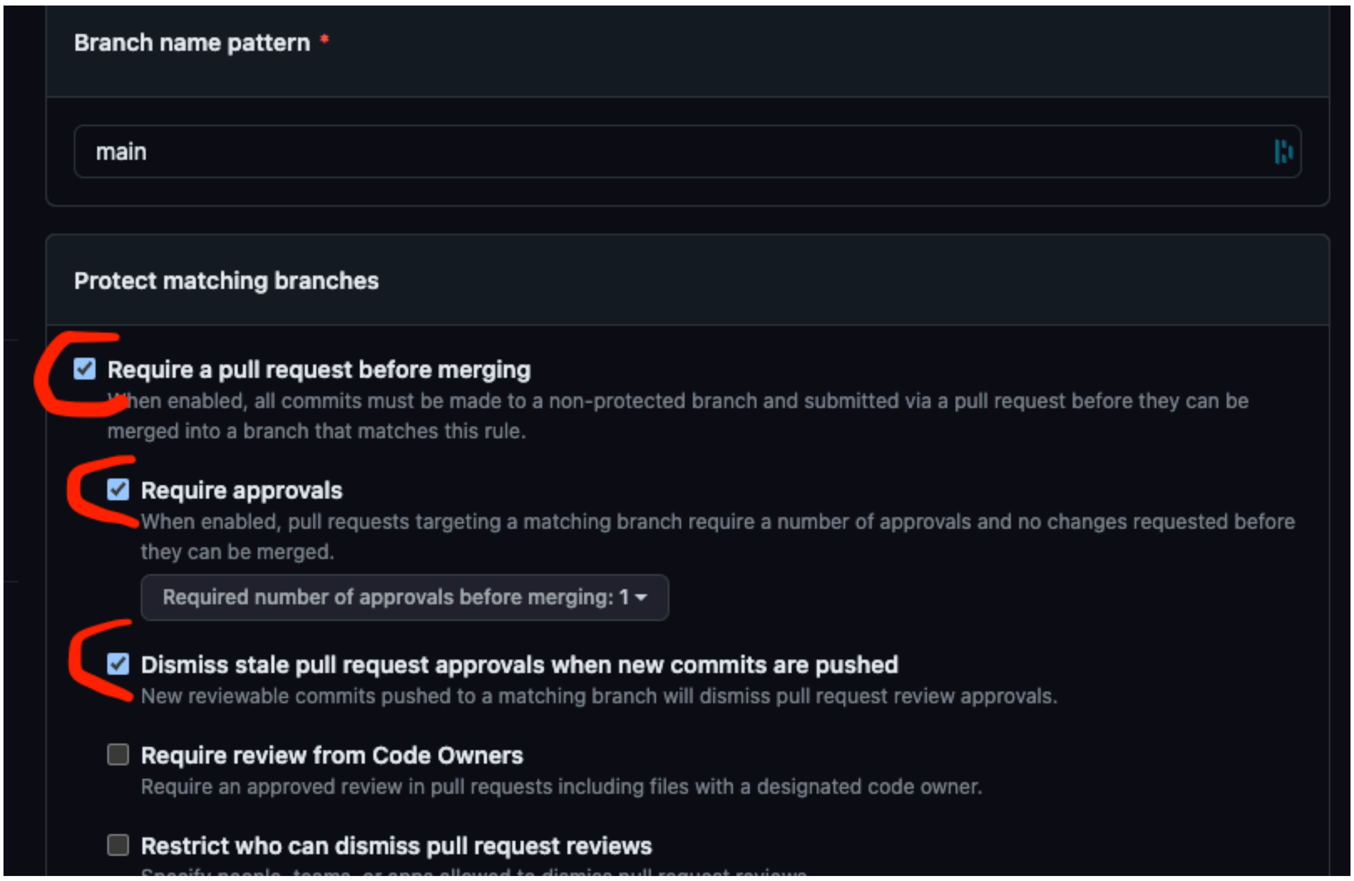
From repository, go to settings


devops-0001 / instavote Public
forked from lfs269/instavote
Edit Pins
Watch 0
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
2 branches
0 tags
Go to file
Add file
Code
Your main branch isn't protected
Protect this branch from force pushing or deletion, or require status checks before merging. Learn more
Protect this branch

From branches, select `Add branch protection rule`


General
Access
Collaborators and teams
Moderation options
Code and automation
Branches
Tags
Rules
Beta
Actions
Branch protection rules
You haven't protected any of your branche
Define a protected branch rule to disable force pu
require status checks before merging. Learn more
Add branch protection rule

Provide the branch name pattern as `main`, select the options as shown in the screenshot,



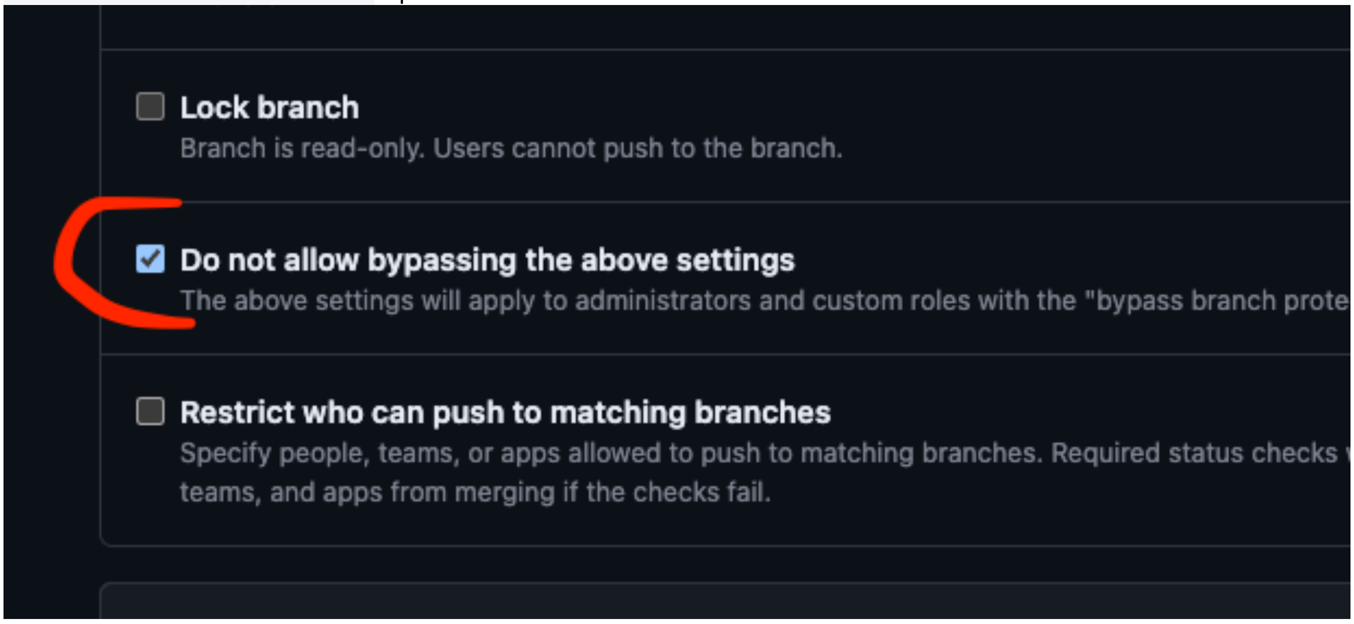
Branch name pattern *

main

Protect matching branches

- [✓] **Require a pull request before merging**
  When enabled, all commits must be made to a non-protected branch and submitted via a pull request before they can be merged into a branch that matches this rule.
  - [✓] **Require approvals**
    When enabled, pull requests targeting a matching branch require a number of approvals and no changes requested before they can be merged.
    Required number of approvals before merging: 1
  - [✓] **Dismiss stale pull request approvals when new commits are pushed**
    New reviewable commits pushed to a matching branch will dismiss pull request review approvals.
  - [ ] **Require review from Code Owners**
    Require an approved review in pull requests including files with a designated code owner.
  - [ ] **Restrict who can dismiss pull request reviews**

Ensure `Do not allow bypassing the above settings` which is now the replacement for `Include Administrators` option as seen earlier.



- [ ] **Lock branch**
  Branch is read-only. Users cannot push to the branch.
- [✓] **Do not allow bypassing the above settings**
  The above settings will apply to administrators and custom roles with the "bypass branch prote
- [ ] **Restrict who can push to matching branches**
  Specify people, teams, or apps allowed to push to matching branches. Required status checks
  teams, and apps from merging if the checks fail.

Proceed to `Create` the branch protection rule.

Committing Changes After adding Branch Protection Rule

```
cd instavote/deploy
git log
git branch
echo "Test after enforcing branching model" >> README.md
git diff
git status
git commit -am "updating README"
git push origin main
```

Now it throws an error such as

```
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote: error: GH006: Protected branch update failed for refs/heads/main.
remote: error: At least 1 approving review is required by reviewers with write
access.
To https://github.com/devops-0001/instavote.git
 ! [remote rejected] main -> main (protected branch hook declined)
error: failed to push some refs to 'https://github.com/devops-0001/
instavote.git'
roo
```

Which validates the branch protection rule, in turns a trunk based development model is being enforced.

Doing it the Right Way

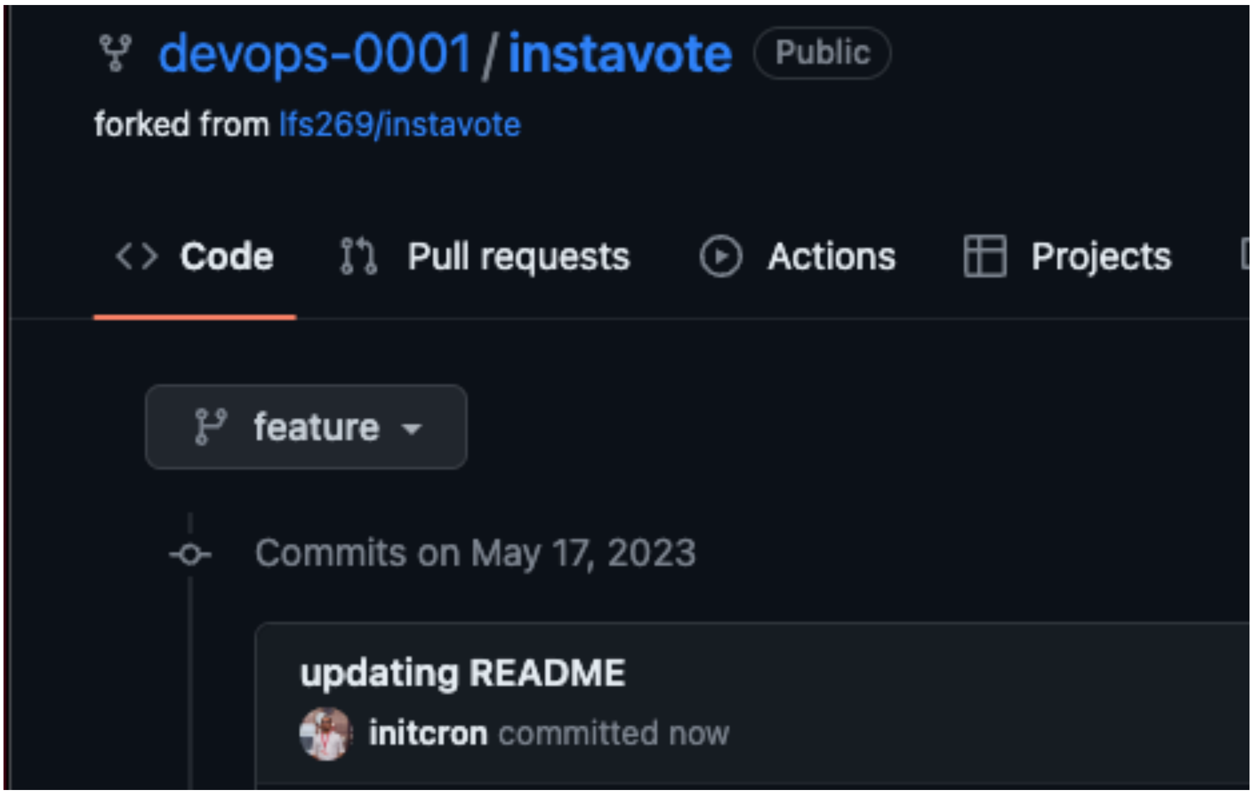
Begin by first resetting the changes,

```
git log
git reset --hard HEAD~1
git log
```

```
 git checkout -b feature
```

```
git branch
echo "Test after enforcing branching model" >> README.md
git diff
git status
git commit -am "updating README"
git push origin feature
```

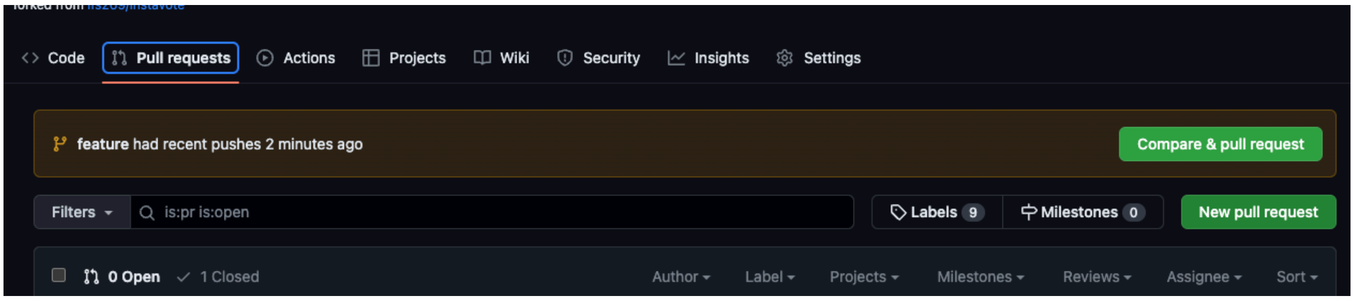
You would now see the changes pushed to the `feature` branch of the repository.



Commits on May 17, 2023

updating README
initcron committed now

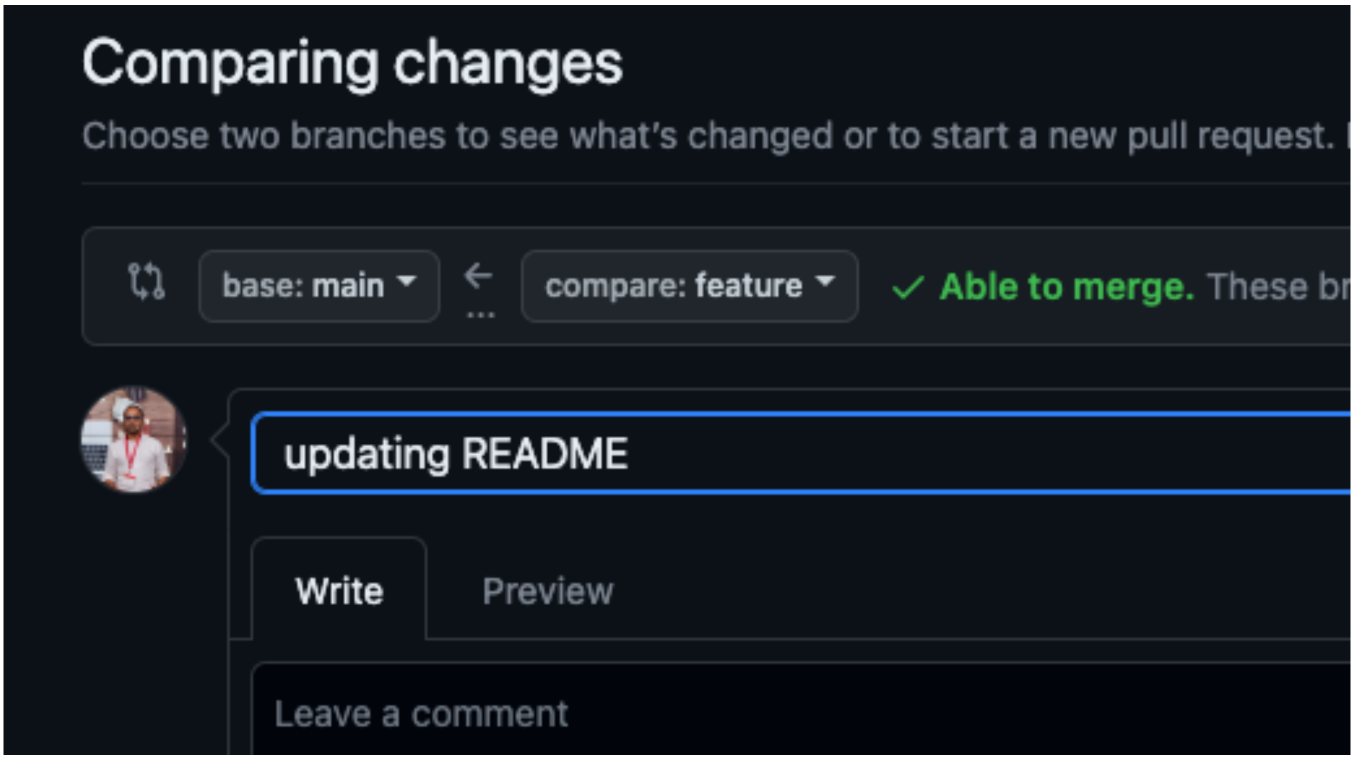
Proceed to create a pull request as earlier to merge the changes added to the feature branch into the main line of the code.



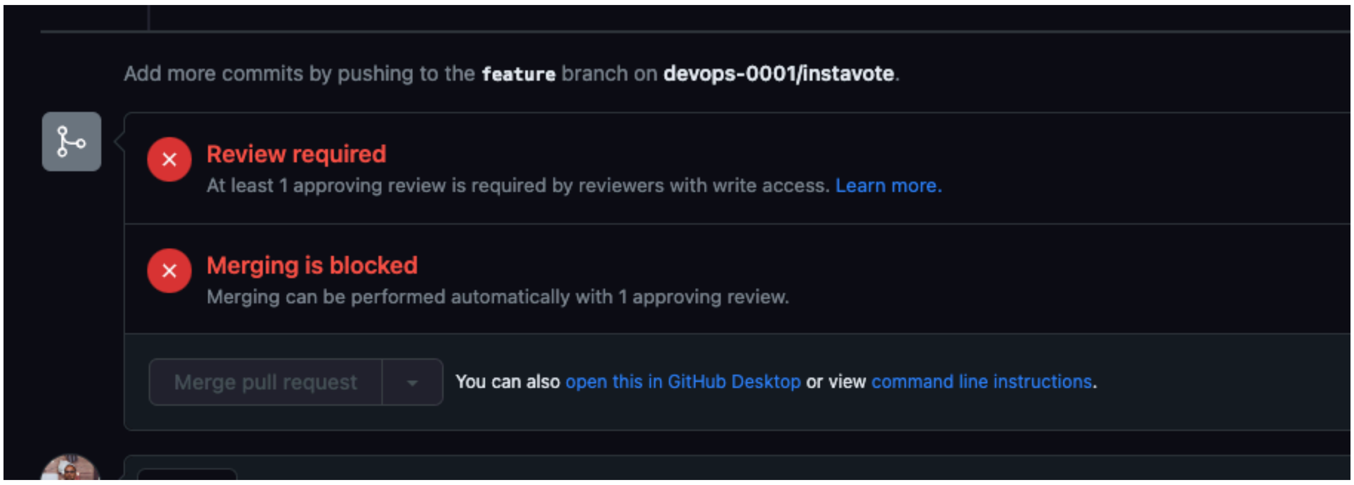
feature had recent pushes 2 minutes ago

is:pr is:open

0 Open ✓ 1 Closed


Comparing changes

Choose two branches to see what's changed or to start a new pull request.


Add more commits by pushing to the feature branch on devops-0001/instavote.

Review required
At least 1 approving review is required by reviewers with write access. Learn more.

Merging is blocked
Merging can be performed automatically with 1 approving review.

You can also open this in GitHub Desktop or view command line instructions.

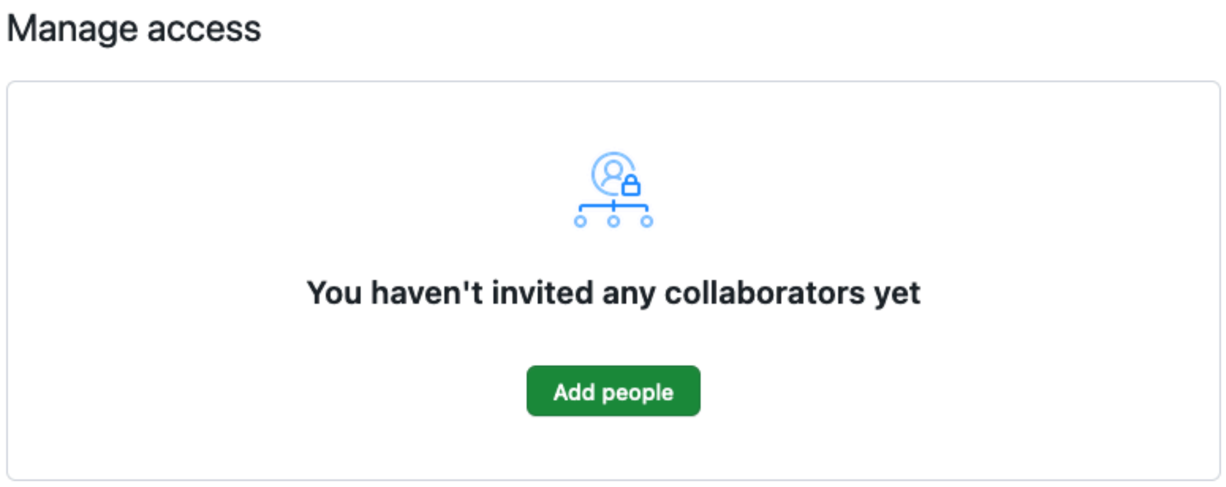
This time, after creating the pull requests, it shows the Merging is blocked, awaiting for a Review.

Enforcing Code Reviews

You could either add a reviewer, or decide to bypass this check by updating the branch protection rules.

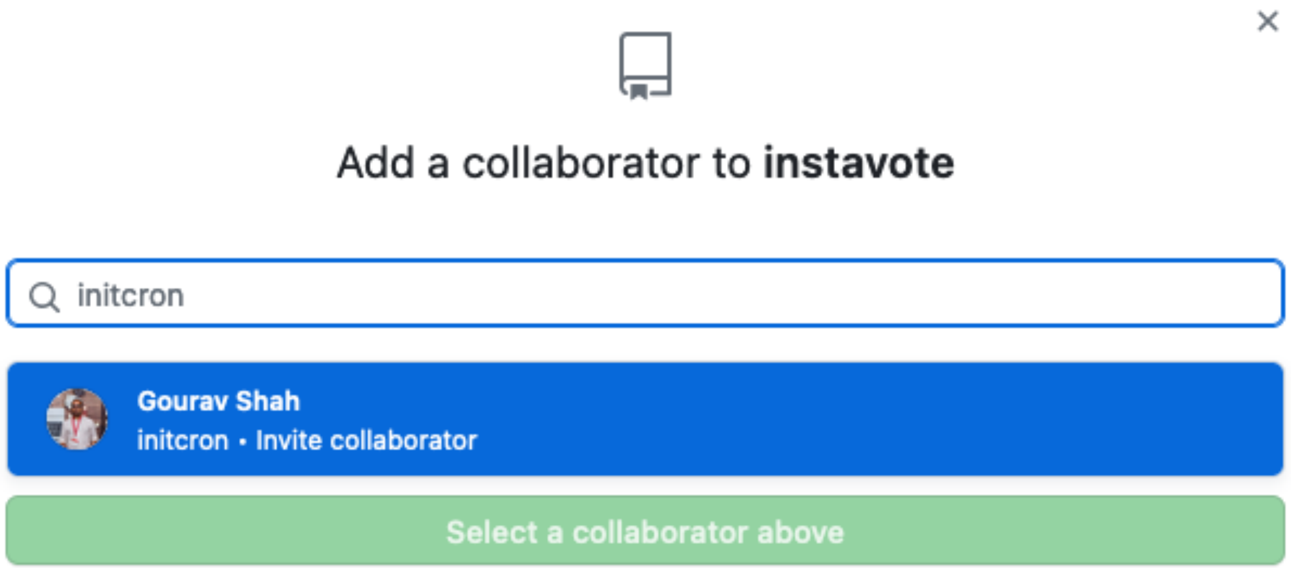
If you decide to add a reviewer, either create a second GitHub account, or find a peer who is ready to review your code.

Go to repository Settings => Collaborators

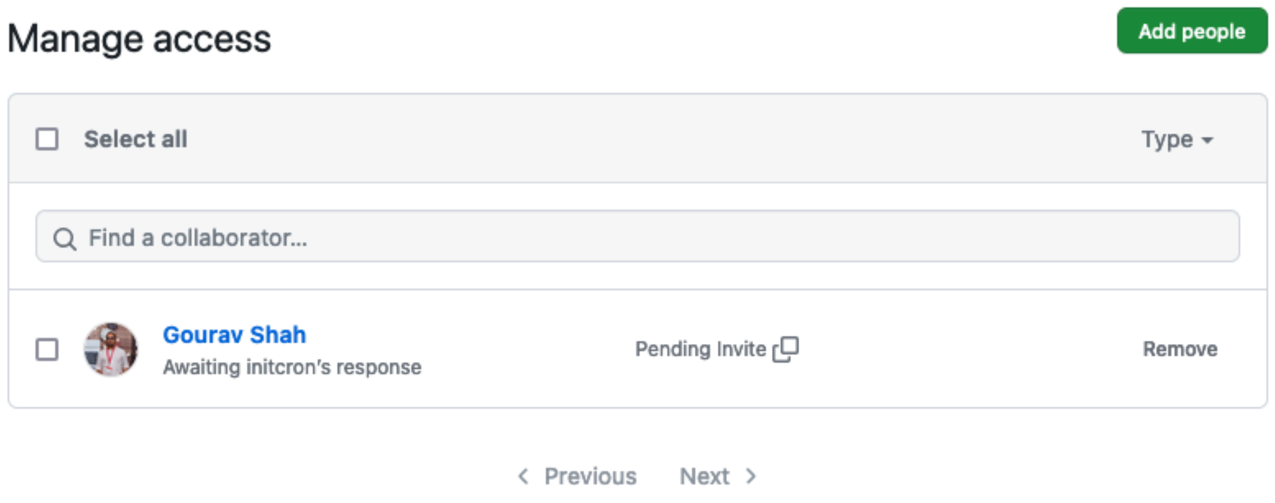


Then invite a collaborator to your repo by using Add People button shown as above.

Choose the GitHub username of the collaborator you would want to add and proceed.


Add a collaborator to instavote
initcron
Gourav Shah
initcron • Invite collaborator
Select a collaborator above

Tip: If you are not working as a team, create a second GitHub account and use that for collaboration/ review.


Manage access

Add people

Select all

Type

Find a collaborator…

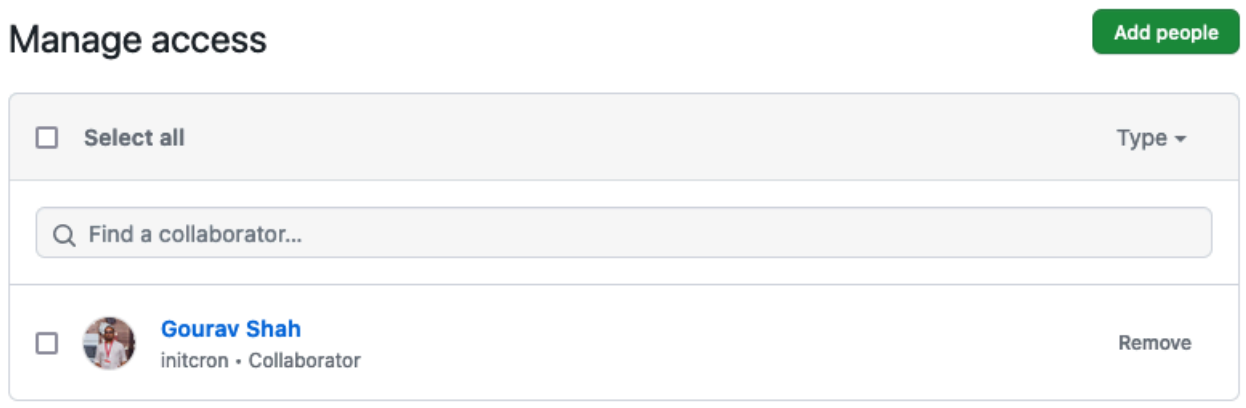
Gourav Shah
Awaiting initcron's response

Pending Invite

Remove

Previous Next

Once the collaborator accepts your request, go back to the pull request and add him/her as a reviewer


Manage access

Add people

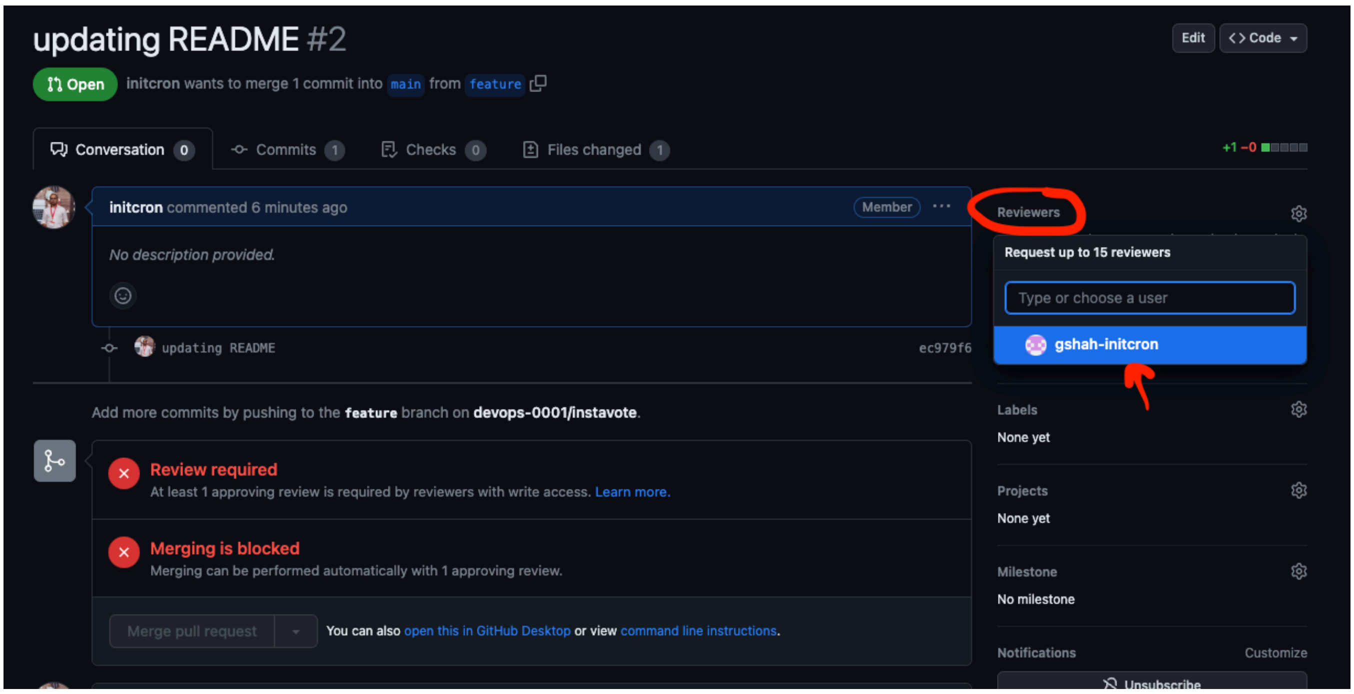
Select all

Type

Find a collaborator…

Gourav Shah
initcron • Collaborator

Remove


updating README #2

Edit   Code

Open   initcron wants to merge 1 commit into `main` from `feature`

Conversation 0   Commits 1   Checks 0   Files changed 1   +1 −0

initcron commented 6 minutes ago   Member

No description provided.

`updating README`   ec979f6

Add more commits by pushing to the **feature** branch on **devops-0001/instavote**.

Review required
At least 1 approving review is required by reviewers with write access. Learn more.

Merging is blocked
Merging can be performed automatically with 1 approving review.

Merge pull request   You can also open this in GitHub Desktop or view command line instructions.

Reviewers
Request up to 15 reviewers
Type or choose a user
gshah-initcron

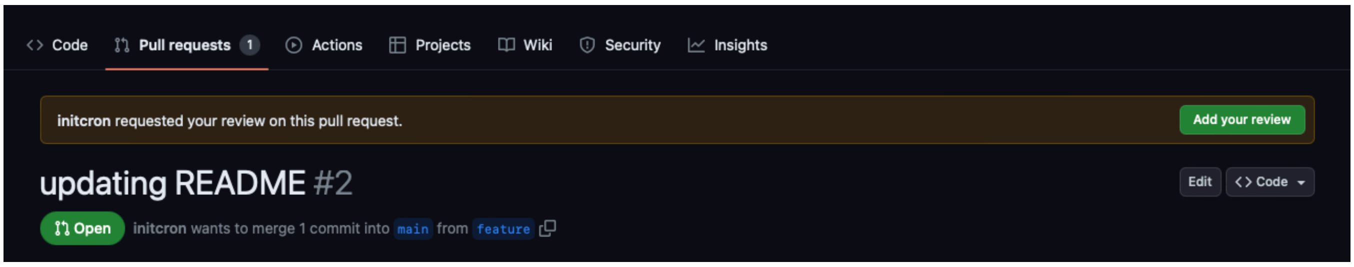
Labels
None yet

Projects
None yet

Milestone
No milestone

Notifications   Customize
Unsubscribe

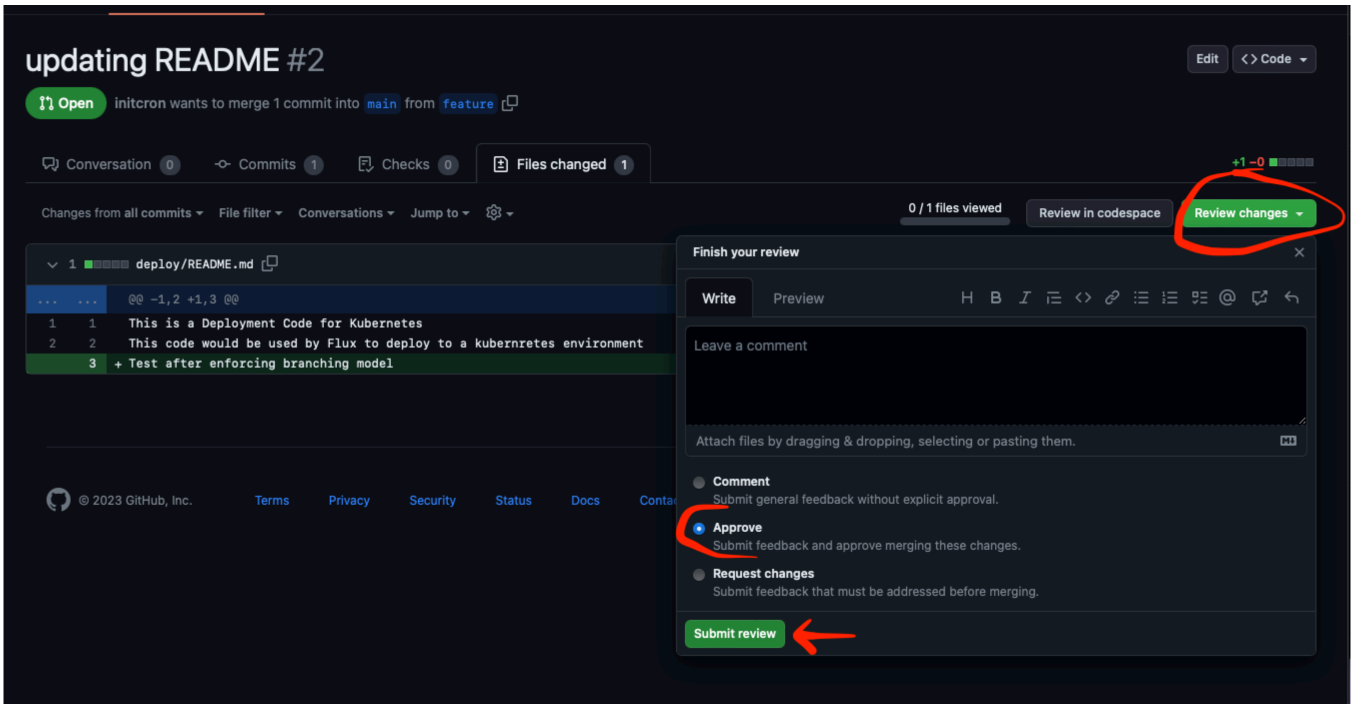
At this time, reviewer needs to add a review and approve the request for you to merge, e.g.



initcron requested your review on this pull request.

updating README #2

Open initcron wants to merge 1 commit into `main` from `feature`



updating README #2

Open initcron wants to merge 1 commit into `main` from `feature`

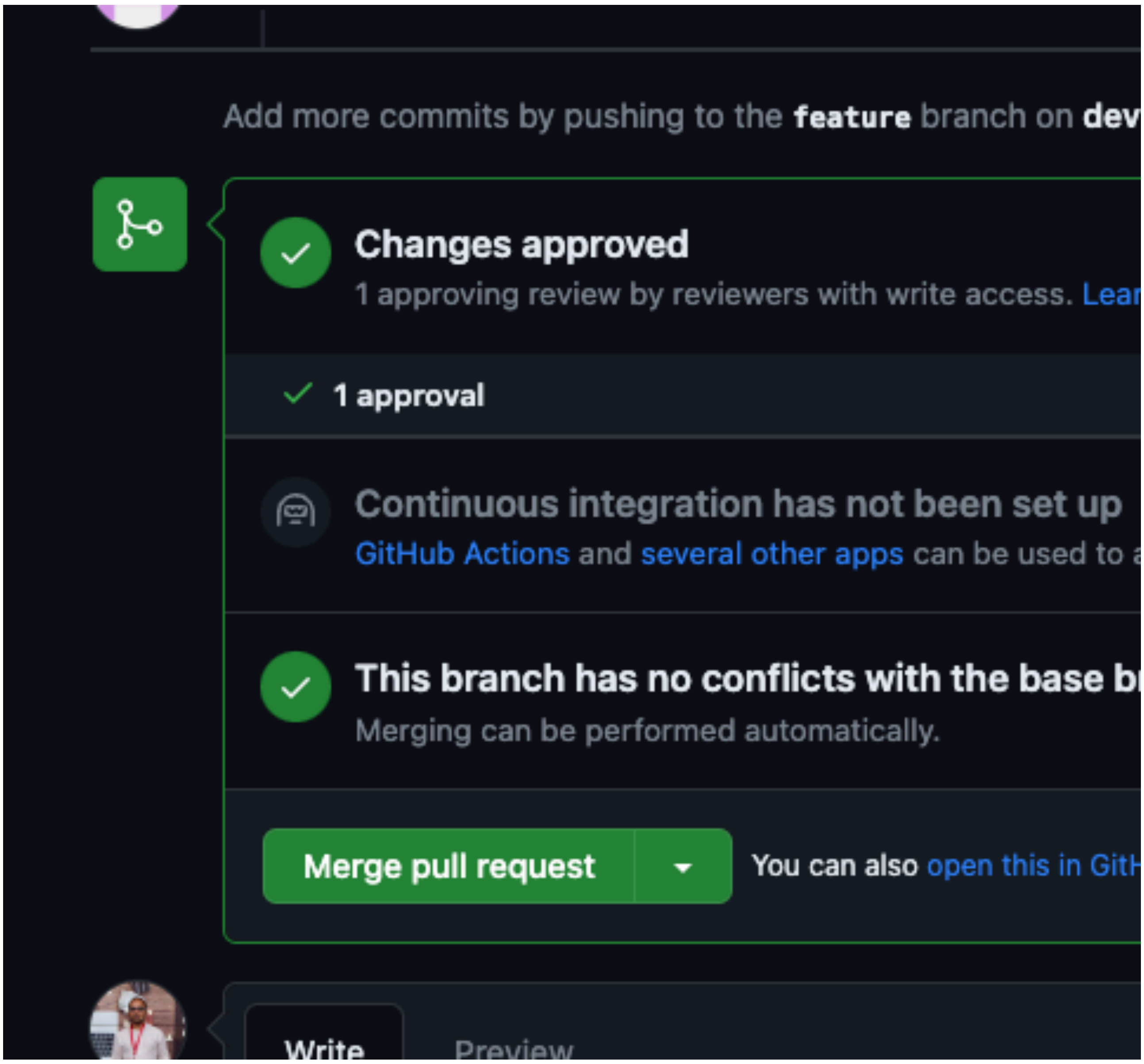
```
deploy/README.md
@@ -1,2 +1,3 @@
This is a Deployment Code for Kubernetes
This code would be used by Flux to deploy to a kubernretes environment
+ Test after enforcing branching model
```

Comment
Submit general feedback without explicit approval.

Approve [✓]
Submit feedback and approve merging these changes.

Request changes
Submit feedback that must be addressed before merging.

Once approved, the pull request will get updated accordingly and you should be able to merge changes.


Add more commits by pushing to the **feature** branch on dev

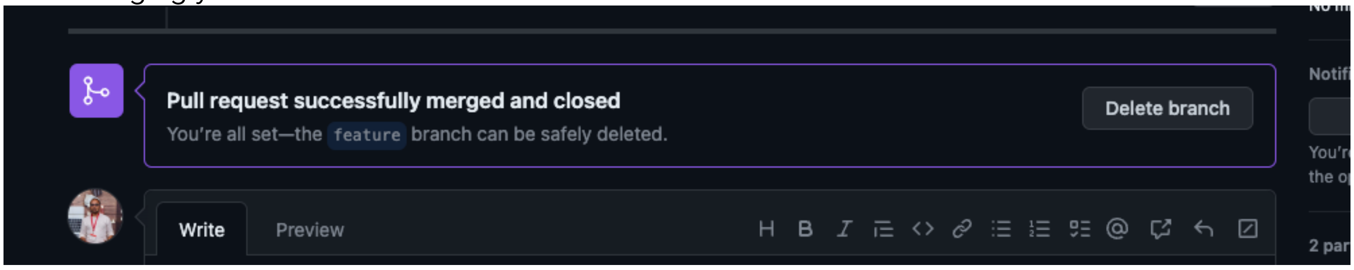
Changes approved
1 approving review by reviewers with write access. Lear

✓ **1 approval**

Continuous integration has not been set up
GitHub Actions and several other apps can be used to a

This branch has no conflicts with the base b
Merging can be performed automatically.

After merging you could delete the branch from GitHub


Pull request successfully merged and closed
You're all set—the `feature` branch can be safely deleted.

And then clean it up locally as,

```
git branch
git checkout main
git branch
git branch -D feature
git pull origin
```

What you see here is the enforcement of trunk based development model whereas,

- You are not allowed to make changes to the main line (trunk) directly.
- All changes are done via branches, brought in via pull requests, with enforcement of best practices such as code reviews etc.

Tagging Releases with Git Tag

Read about Git Tagging related topics here,

- Git Tagging Git - Tagging
- Semantic Versioning Semantic Versioning 2.0.0 | Semantic Versioning

```
git branch
```

Create a special branch to cut the releases from

```
git checkout -b releases/0.1
```

To create and list a tag with a patch release

```
git tag -a v0.1.0 -m "initial release"
git tag
```

```
git tag 4.0
git tag
```

To view the most recent tag

```
git show
```

To see the difference between a simple tag (4.0) and an annotated tag(v0.1.0) ,

```
git show 4.0
git show v0.1.0
```

Where annotated tag shows additional info such as

[sample output]

```
tag v0.1.0
Tagger: Gourav Shah <gs@initcron.org>
Date:   Wed May 17 07:21:03 2023 +0000

initial release
```

It is recommended to use annotated tags.

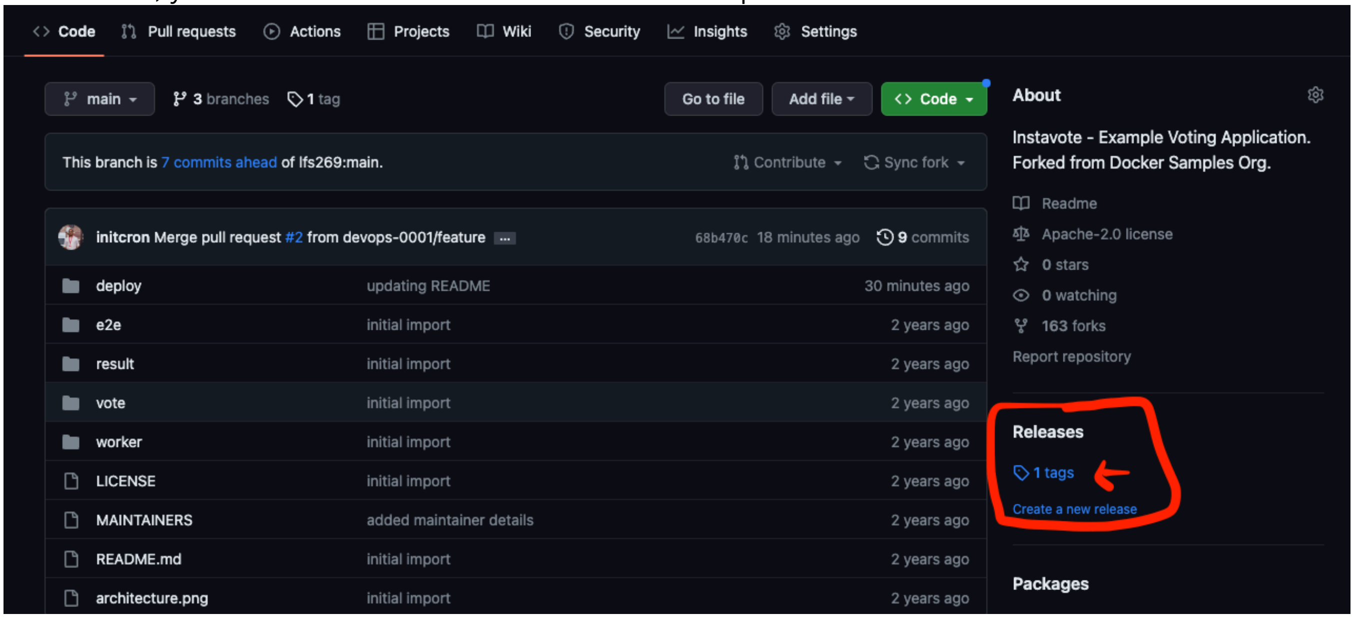
To delete the tag,

```
git tag
git tag -d 4.0
git tag
```

To push the tags to GitHub,

```
git push origin v0.1.0
```

At this time, you shall see a new release added to the repo

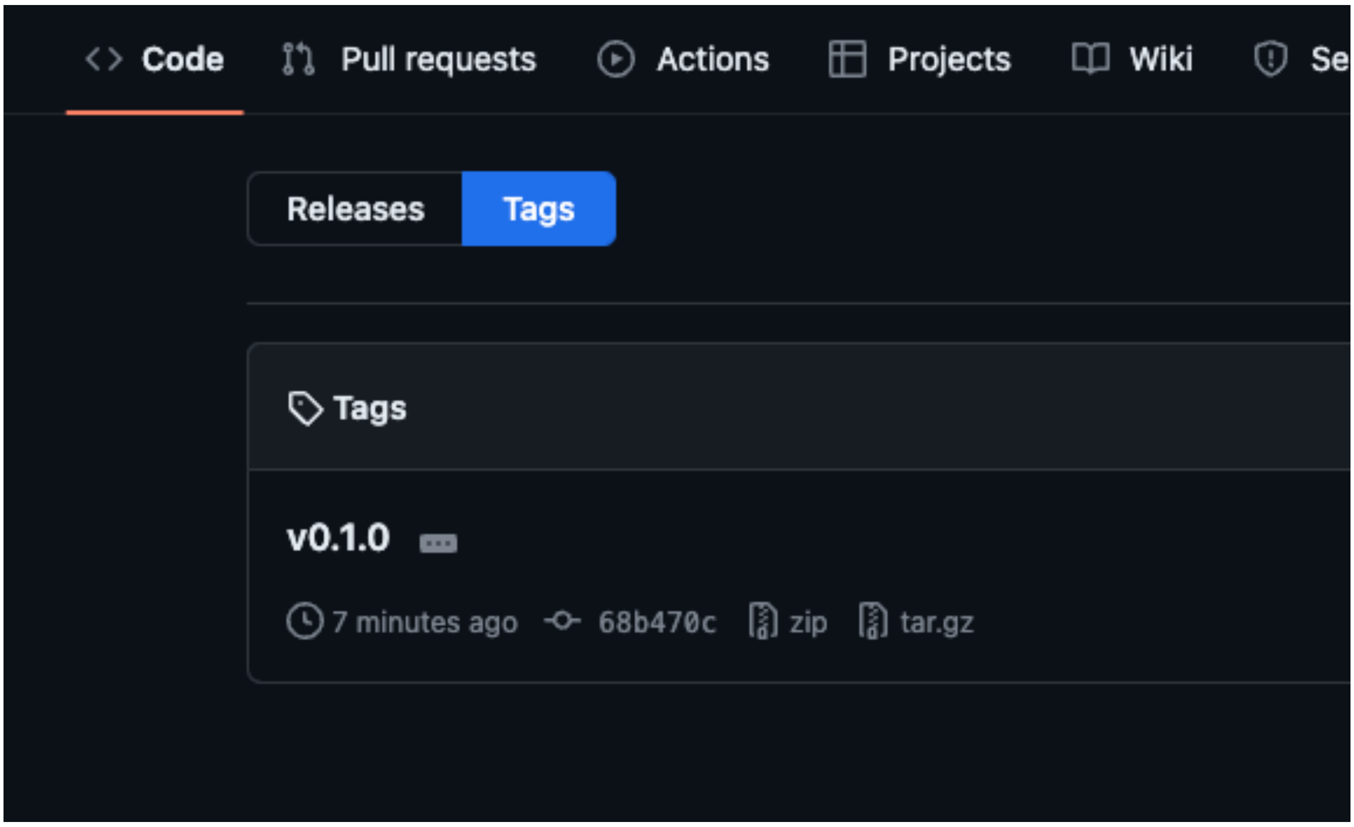


This branch is 7 commits ahead of lfs269:main.

Instavote - Example Voting Application. Forked from Docker Samples Org.

initcron Merge pull request #2 from devops-0001/feature	68b470c 18 minutes ago	9 commits
deploy	updating README	30 minutes ago
e2e	initial import	2 years ago
result	initial import	2 years ago
vote	initial import	2 years ago
worker	initial import	2 years ago
LICENSE	initial import	2 years ago
MAINTAINERS	added maintainer details	2 years ago
README.md	initial import	2 years ago
architecture.png	initial import	2 years ago

Selecting which shows more information and download options for that version as



And this is how you could create and release your product using semantic versioning scheme.

Summary

In this section you learnt the essentials of git for a Devops practitioner including how to start revisioning controlling your code, how to work with branches and remotes, how to enforce a branching model and also how to release your code with semantic versioning scheme.

#udbc/labs