

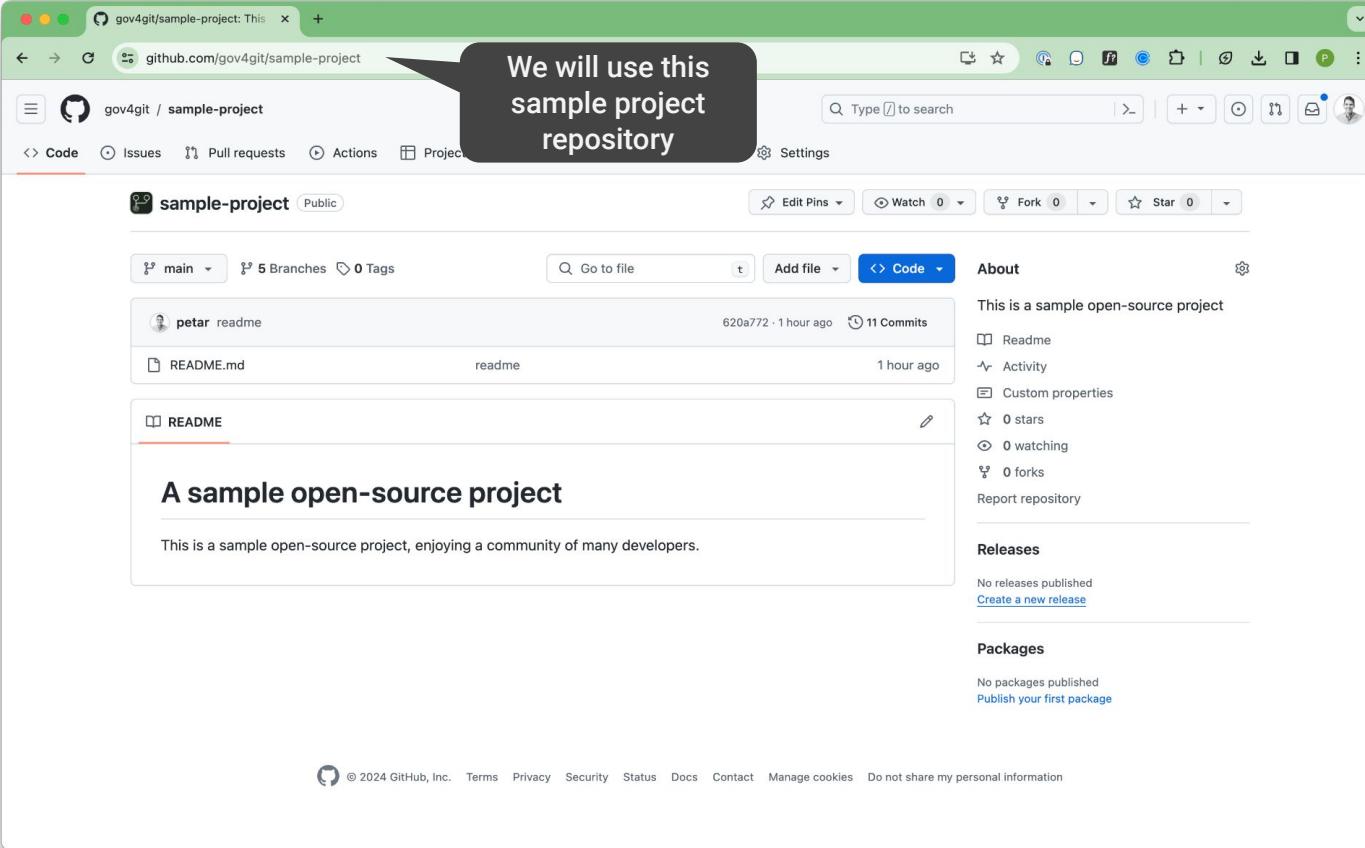
# Gov4Git

Deploy, manage and collaborate

# Deploy

For maintainers

# Pick any GitHub project that you want to govern with Gov4Git



We will use this sample project repository

sample-project Public

main 5 Branches 0 Tags

petar README 620a772 · 1 hour ago 11 Commits

README.md readme 1 hour ago

A sample open-source project

This is a sample open-source project, enjoying a community of many developers.

About

This is a sample open-source project

Readme Activity Custom properties

0 stars 0 watching 0 forks

Report repository

Releases

No releases published Create a new release

Packages

No packages published Publish your first package

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

# Install the Gov4Git desktop app

A screenshot of a web browser window displaying the GitHub page for the Gov4Git desktop application. The URL in the address bar is [github.com/gov4git/gov4git](https://github.com/gov4git/gov4git). The page content includes an introduction to Gov4Git, sections for installing the desktop app (Windows, macOS, Linux), and a terminal command for Linux installation. A callout bubble points to the introduction section with the text: "Installation instructions are on the Gov4Git GitHub page, and our website".

Installation instructions are on the Gov4Git GitHub page, and our website

**gov4git: Decentralized**

release passing

## Introduction

*gov4git* is a decentralized protocol for governing open-source communities based on git.

It is a wholistic framework for lifelong governance of open-source projects, which is secure, flexible, transparent, and pluralistic.

*gov4git* is designed to be practical and accessible. It requires git hosting as the only persistent infrastructure. It is easy (and continuously getting easier) to deploy by non-technical users, using an accompanying command-line client or a desktop app.

## Install the desktop app

All users (community organizers and community members) can use Gov4Git via our desktop application. Find the installation link for your OS below.

### Windows

Install the latest release [here](#).

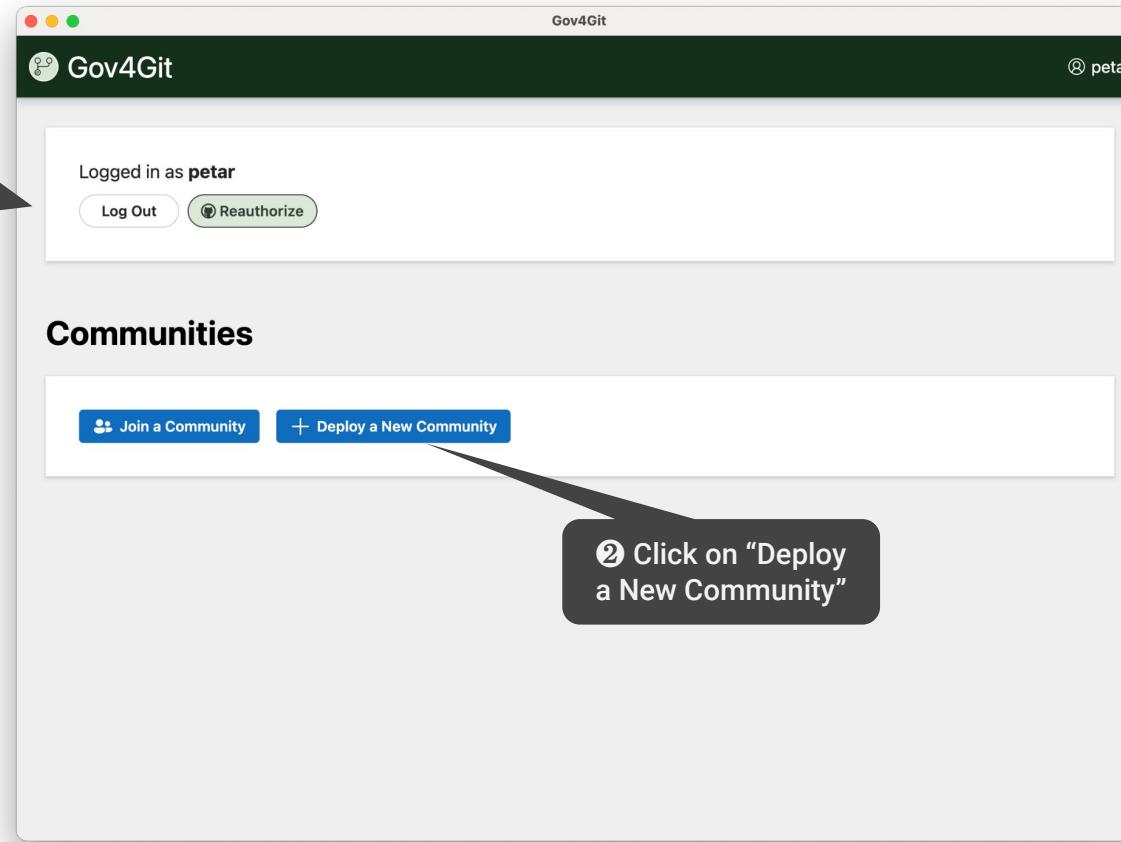
### macOS

Install the latest release [here](#).

### Linux

```
curl -sSfLO https://github.com/gov4git/desktop-application/releases/latest/download/gov4git-  
sudo chmod +x ./gov4git-desktop-app.AppImage  
./gov4git-desktop-app.AppImage
```

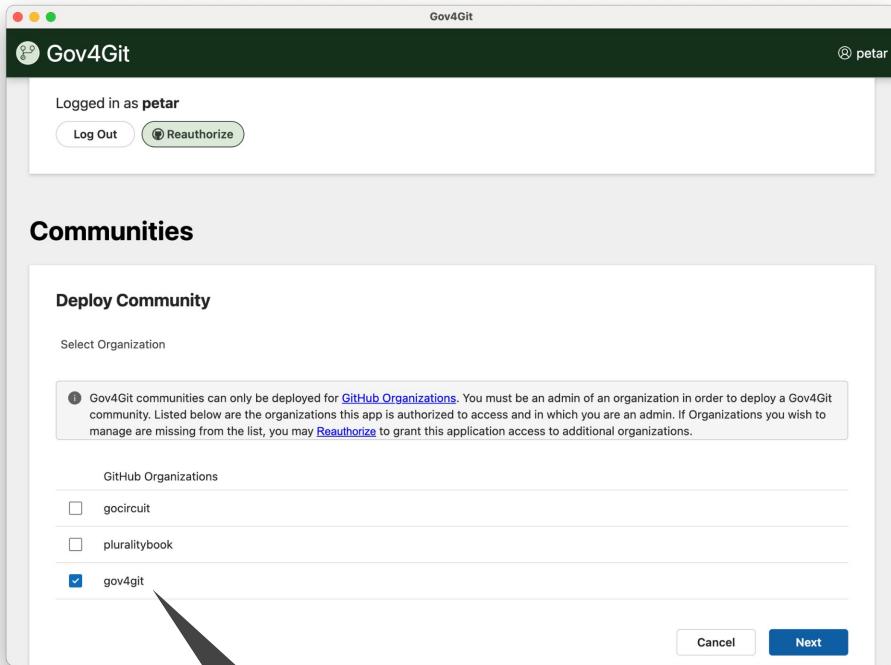
# Start the Gov4Git desktop app



① The app will invite you to authenticate with GitHub first. Then you will arrive at this screen.

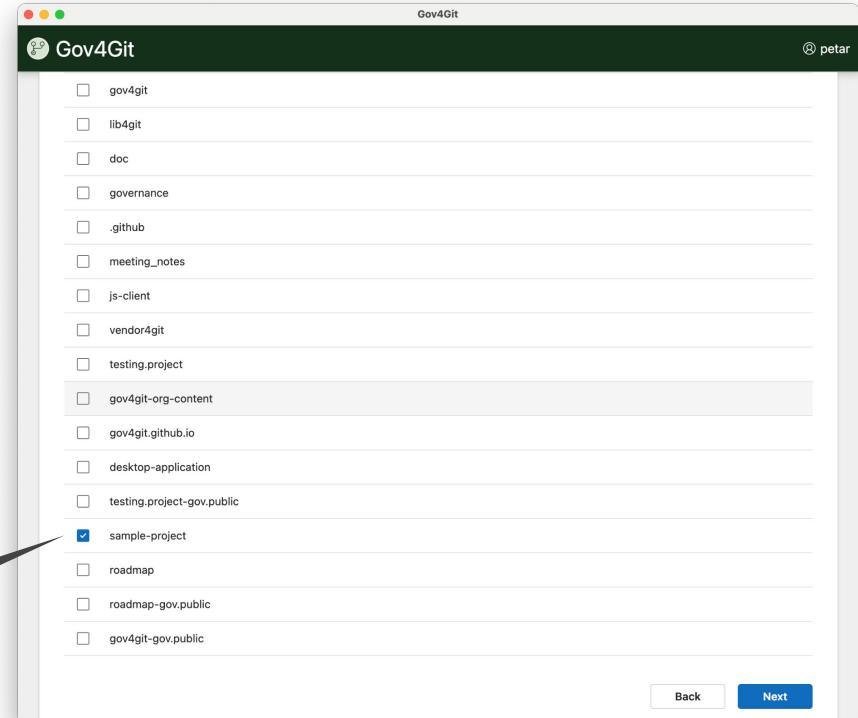
② Click on “Deploy a New Community”

# Specify which GitHub repo you want to govern



① Select the GitHub org of your project

② Then select the project repository



# Provide your maintainer GitHub credentials, and deploy

The screenshot shows the 'Generating a Personal Access Token' step of the Gov4Git setup. It includes a list of steps, a table of repository permissions, and a note about organization permissions. A large callout bubble contains instructions for entering a GitHub Personal Access Token.

Generating a Personal Access Token

1. Visit <https://github.com/settings/personal-access-tokens/new> to get started.
2. Provide a token name, expiration date, and description for the token.
3. Select **gov4git** as the Resource owner.
4. Select All repositories for the Repository access option.
5. Under Permissions, select the following Repository permissions.

Option	Access Level
Actions	Read and Write
Administration	Read and Write
Contents	Read and Write
Environments	Read and Write
Issues	Read and Write
Pull requests	Read and Write
Secrets	Read and Write
Variables	Read and Write
Workflows	Read and Write

and select the following Organization permissions

Option	Access Level
Members	Read

6. Select Generate token
7. Copy and paste the token below

Personal Access Token

.....

Back Next

① Finally, enter the Personal Access Token for the GitHub account which will run the governance bot.

It is a good idea to create a separate account for it, named something like: "sample-project-govbot"

The screenshot shows the 'Deploy Community' step of the Gov4Git setup. It displays the user is logged in as 'petar' and shows the 'Communities' section. A callout bubble indicates the 'Deploy' button.

Gov4Git

Logged in as **petar**

Log Out Reauthorize

## Communities

### Deploy Community

Select Organization > Select Repo > PAT > Deploy

Deploy Gov4Git for **gov4git/sample-project**

② Click "Deploy"

# Confirmation governance has been deployed

The screenshot shows the Gov4Git web application interface. At the top, there's a navigation bar with a user icon, the text "Gov4Git", and a notification badge for "petar 0". Below the header, it says "Logged in as petar" with "Log Out" and "Reauthorize" buttons. The main area is titled "Communities" and lists one item: "sample-project" with "Admin" status. Below this is a "Deploy Community" section with a "Select Organization > Select Repo > PAT > Deploy" flow. A green success message box at the bottom contains the text: "Success. A Gov4Git community has been deployed for gov4git/sample-project. You can view the newly created community repo at <https://github.com/gov4git/sample-project-gov.public>. Visit <https://github.com/gov4git/gov4git> for documentation on how to manage Gov4Git community repos.".

① You will get a confirmation screen.  
Now governance is deployed and operational.

② Deploying governance does not write to your project repository.

It creates two new repositories (one public, one private).

And installs the governance logic as a GitHub action in the public one.

# Manage

For maintainers

# Maintainers can issue credits to anyone

The screenshot shows a GitHub repository named "gov4git/sample-project" with an open issue creation form. The title field contains "issue credits to maintainer" and the description field contains "issue 200000 credits to @petar". The right sidebar shows the issue is labeled "gov4git:directive". Three numbered callouts point to specific parts of the interface:

- ① Start by issuing some credits to yourself (the maintainer).** Points to the title field.
- ② You do that by creating a new GitHub issue, containing a directive like: "Issue 200000 credits to @petar"** Points to the description field.
- ③ Then label the issue as "gov4git:directive" to let Gov4Git know that this issue is a command you want it to execute.** Points to the "Labels" section in the sidebar.

# Wait a couple of minutes

The screenshot shows a GitHub issue page for a repository named "gov4git / sample-project". The issue is titled "issue credits to maintainer #24" and is marked as "Open". A comment from the user "petar" is visible, stating "issue 200000 credits to @petar". Below this comment, another action is shown: "petar added the gov4git:directive label now". The right side of the screen displays various issue settings: Assignees (No one—assign yourself), Labels (gov4git:directive), Projects (None yet), Milestone (No milestone), Development (Create a branch for this issue or link a pull request.), and Notifications (Customize, Unsubscribe). A message at the bottom indicates that notifications are being received because the user authored the thread. A dark callout bubble on the left side of the screenshot contains two pieces of text:

After you create the issue, wait a couple of minutes.

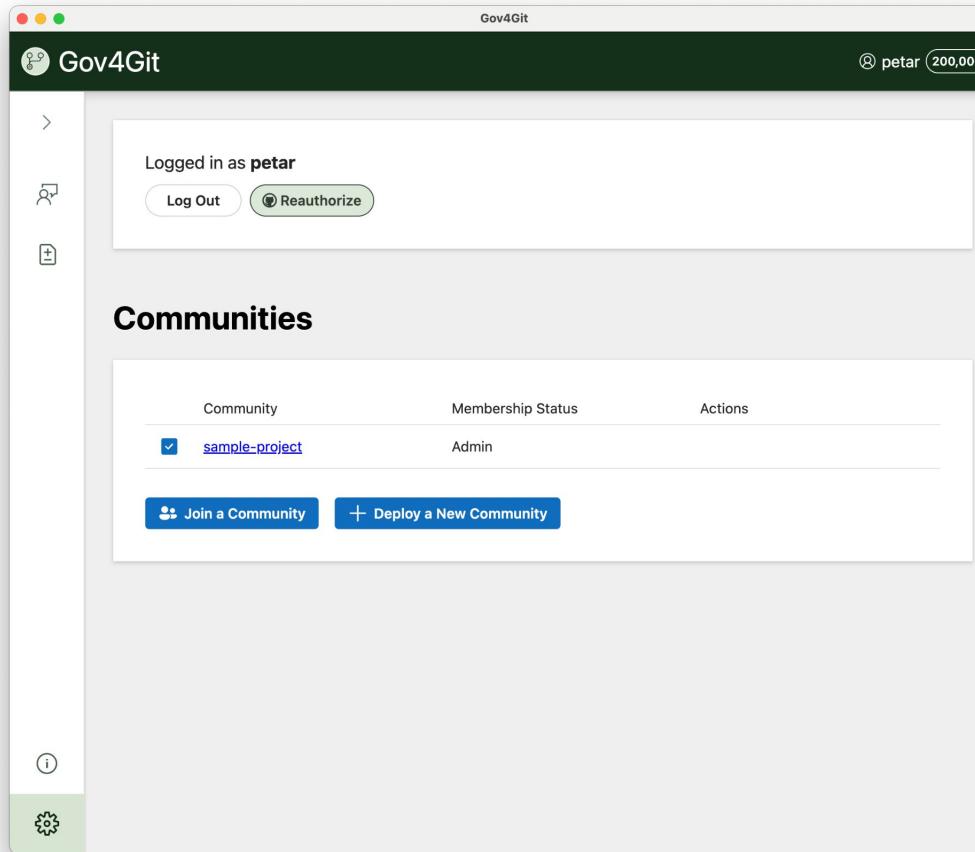
The governance bot wakes up every few minutes and performs its duties.

# You will see a confirmation when the task is completed

The screenshot shows a GitHub issue page for a sample project. The issue is titled "issue credits to maintainer #24" and is marked as "Closed". A comment from "petar" states: "issue 200000 credits to @petar". A reply from "gov4git-bot" says: "Follow up" and "Issued 200000 credits to member @petar.". The issue summary on the right indicates "2 participants" and shows the "gov4git:directive" label.

When the system processes your directive, it will produce a confirmation comment and close the issue.

# You can see your balance in the app too



If you go back to the desktop app, you will see your updated balance in the top right corner.

# Collaborate

For collaborators and maintainers

# Issues

# Collaborators create project issues freely

The screenshot shows a GitHub repository named "gov4git/sample-project". A collaborator has opened a new issue titled "our project needs a LICENSE". The issue description reads: "we need to pick an open-source license and create a LICENSE file". The "Labels" section shows a single label "gov4git:managed". A large callout bubble on the left states: "① Any collaborator can create a new project issue". Another callout bubble on the right states: "② After the issue is created, maintainers can label it as "gov4git:managed" to tell the governance system that you want it to manage this issue."

① Any collaborator can create a new project issue

② After the issue is created, maintainers can label it as "gov4git:managed" to tell the governance system that you want it to manage this issue.

gov4git / sample-project

New Issue · gov4git/sample-project · Issues 1 · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

Add a title

our project needs a LICENSE

Add a description

we need to pick an open-source license and create a LICENSE file

Markdown is supported Paste, drop, or click to add files

+ Add tasklist Submit new issue

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

# Wait until the system acknowledges the issue is under management

The screenshot shows a GitHub issue page for a sample project. The issue is titled "our project needs a LICENSE #25". It has one comment from user "petar" and one notice from "gov4git-bot". The right sidebar displays management details like assignees, labels, and projects.

**① In a couple of minutes, the system will confirm that the issue is now under management.**

**② It will provide some useful information, such as which management protocol is being used to manage this issue.**

**Assignees:** None yet — [assign yourself](#)

**Labels:** [gov4git:managed](#)

**Projects:** None yet

**Milestone:** No milestone

**Development:** [Create a branch for this](#)

**Unsubscribe:** You're receiving notifications because you're involved in the thread.

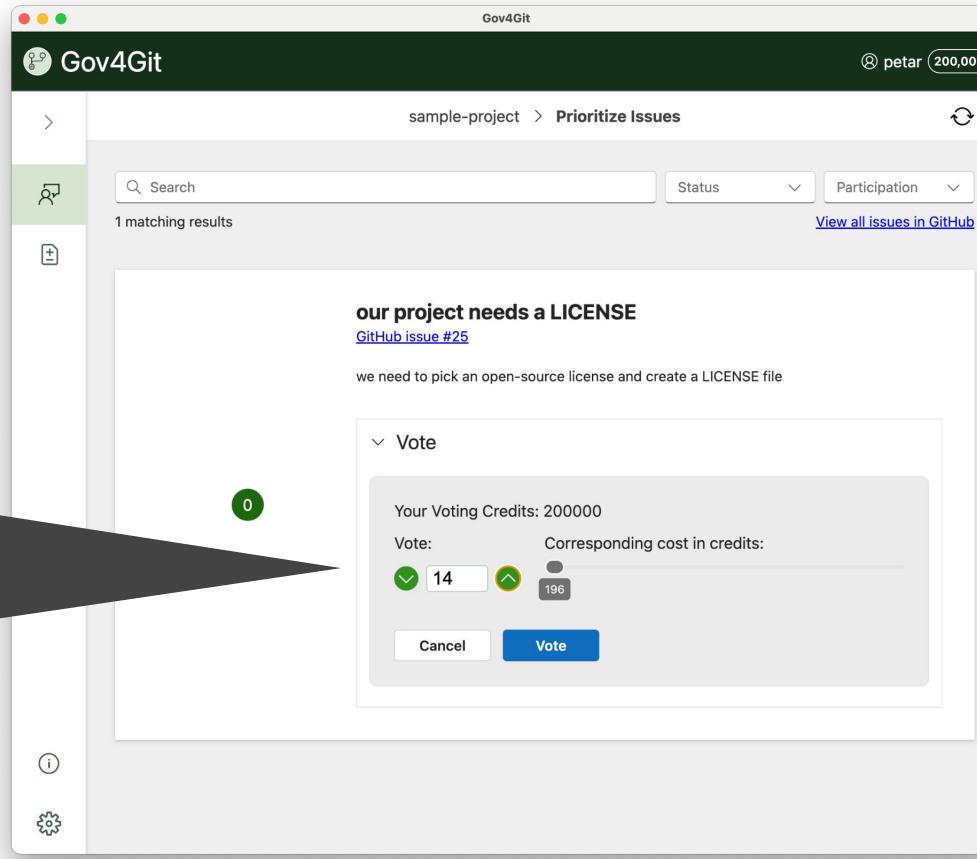
**Participants:** 2 participants

**Lock conversation:**

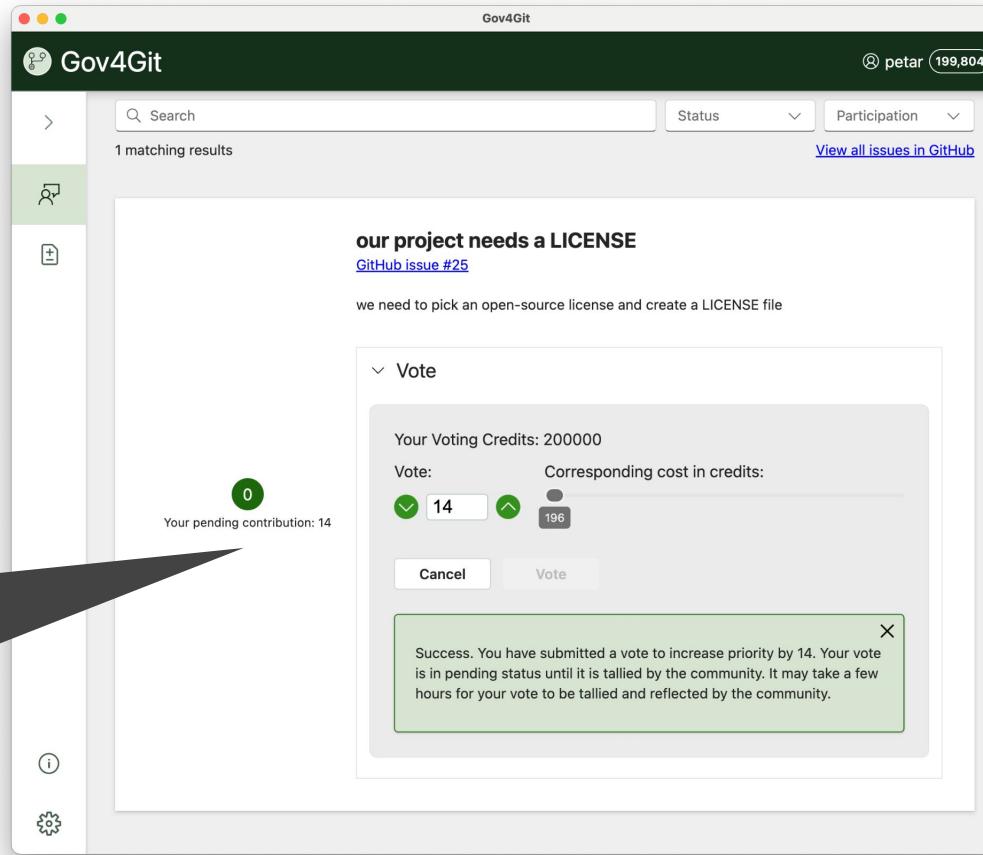
**Pin issue:**  ⓘ

**Transfer issue:** → ⓘ

# Collaborators can vote on managed issue



# Voting confirmation and pending votes



# Vote scores are also shown on the GitHub issue page

The screenshot shows a GitHub issue page for a project managed by Gov4Git. The issue is titled "our project needs a LICENSE #25" and was opened by petar 4 minutes ago with 2 comments.

The page displays Gov4Git notices and vote scores:

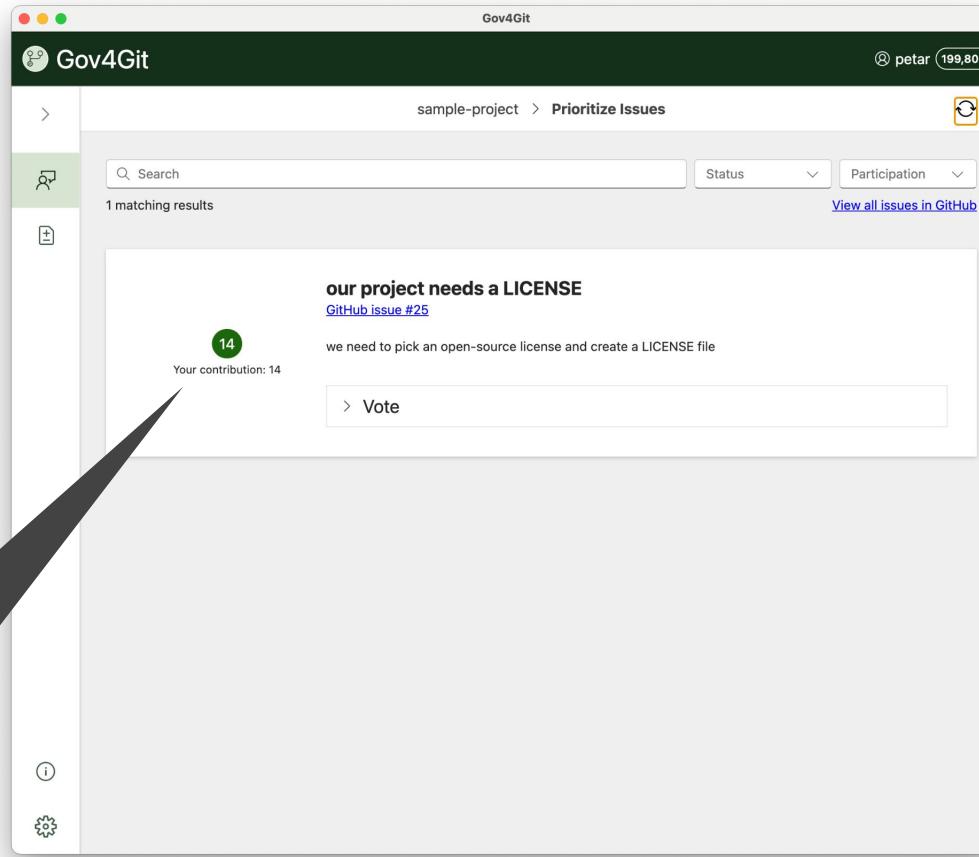
- Notice ynwuuij**: This issue's priority score is now 0.000000. The cost of priority is 0.000000. The projected bounty is now 0.000000.
- Notice bt42j4**: The set of eligible proposals claiming this issue is empty.
- gov4git-bot commented now**: Gov4Git notices
- Notice aw15b6**: This issue's priority score is now 14.000000. The cost of priority is 196.000000. The projected bounty is now 28.000000.
- Notice cjjq2d**: The set of eligible proposals claiming this issue is empty.

On the right side of the issue page, there are options for managing the issue: Lock conversation, Pin issue, Transfer issue, and Delete issue. The sidebar shows 2 participants.

A callout bubble on the left side of the screen contains the following text:

After new votes are processed, the system will post a comment on the issue's page with updated information about its priority score and bounty for its resolution.

# The desktop app also reflects your processed votes



# PRs

# Collaborators can freely create PRs that claim to address issue(s)

The screenshot shows a GitHub pull request creation interface. At the top, there's a green header bar with the URL [github.com/gov4git/sample-project/compare/main...petar-patch-4?quick\\_pull=1](https://github.com/gov4git/sample-project/compare/main...petar-patch-4?quick_pull=1). Below it, the repository name is `gov4git / sample-project`. The navigation bar includes **Code**, **Issues** (with 2), **Pull requests**, **Actions**, **Projects**, **Wiki**, **Security**, **Insights**, and **Settings**.

The main area is titled "Open a pull request". It says: "The change you just made was written to a new branch named `petar-patch-4`. Create a pull request below to propose these changes. [Learn more about diff comparisons here.](#)"

Below this, there are dropdowns for "base: main" and "compare: petar-patch-4", with a note: "Able to merge. These branches can be automatically merged."

The pull request form has two sections: "Add a title" and "Create LICENSE" (which is filled in). There's also an "Add a description" section containing the text: "I've added a reasonable license." and "Issues that this PR claims: - claims <https://github.com/gov4git/sample-project/issues/25>".

To the right of the form, there are several sections: "Reviewers" (No reviews), "Assignees" (No one—[assign yourself](#)), "Labels" (`gov4git:managed`), "Projects" (None yet), "Milestone" (None), "Development" (Use [Closing keywords](#) in the description to automatically close issues), and "Helpful resources" ([GitHub Community Guidelines](#)).

At the bottom, there's a "Create pull request" button and a note: "Remember, contributions to this repository should follow our [GitHub Community Guidelines](#)."

**①** Collaborators can create PRs that claim to resolve one or more issues.  
To claim an issue, the PR description must include the statement:  
"claims ISSUE\_URL"

**②** Once a PR is created, a maintainer has to label it as "gov4git:managed" to let the system know it should manage this PR.

# Wait for acknowledgement that the PR is under management

In a couple of minutes, the governance system will post a comment on the PR page, acknowledging that the PR is now under management.

The screenshot shows a GitHub pull request page for a sample project. The PR is titled "Create LICENSE #26" and is currently open. A comment from "petar" states: "I've added a reasonable license." Below this, the "Issues that this PR claims:" section lists a single item: "claims our project needs a LICENSE #25". A "Create LICENSE" commit is shown with a "Verified" status and hash "87c8f5c". The "Assignees" field is empty, and the "Labels" field contains "gov4git:managed". The "Projects" and "Milestone" fields are also empty. In the "Development" section, it says "Successfully merging this pull request may close these issues." The "Notifications" section indicates that the user is receiving notifications because they authored the thread, with an "Unsubscribe" button. The "Participants" section shows two users: "petar" and "gov4git-bot".

github.com/gov4git/sample-project/pull/26

Create LICENSE #26  
petar wants to merge 1 commit into `main` from `petar-patch-4`

**petar commented 1 minute ago**

I've added a reasonable license.

Issues that this PR claims:

- claims [our project needs a LICENSE #25](#)

**petar added the `gov4git:managed` label 1 minute ago**

**gov4git-bot mentioned this pull request 1 minute ago**

[our project needs a LICENSE #25](#)

**gov4git-bot commented 1 minute ago**

**Gov4Git notices**

On Tuesday, 05-Mar-24 22:37:47 UTC by Gov4Git dev

Notice [misz74](#)

Started managing this PR, using the Open-Source Management Protocol (Waimea), as Gov4Git proposal 26 with initial approval score of 0.000000.

This project is managed by [Gov4Git](#), a decentralized governance system for collaborative git projects. To participate in governance, install the [Gov4Git desktop app](#).

**Reviewers**  
No reviews  
Still in progress? Convert to draft

**Assignees**  
No one—assign yourself

**Labels**  
`gov4git:managed`

**Projects**  
None yet

**Milestone**  
No milestone

**Development**  
Successfully merging this pull request may close these issues.  
None yet

**Notifications** [Customize](#) [Unsubscribe](#)  
You're receiving notifications because you authored the thread.

**2 participants**

# The issue will be notified that there is a PR claiming to address it

A screenshot of a GitHub issue page for a sample project. The issue is titled "our project needs a LICENSE #25" and was opened by petar 11 minutes ago. A comment from petar mentions the issue 3 minutes ago. A reply from "gov4git-bot" 2 minutes ago includes a link to "Create LICENSE #26". A dark callout bubble points to this reply with the text: "The system will also post a comment on the issue's page, indicating that there is a PR claiming to address it."

our project needs a LICENSE #25  
petar opened this issue 11 minutes ago - 3 comments

petar mentioned this issue 3 minutes ago  
Create LICENSE #26

gov4git-bot commented 2 minutes ago

Gov4Git notices

On Tuesday, 05-Mar-24 22:37:47 UTC by Gov4Git dev

Notice hoo4pl

This issue was referenced by [#26](#), managed as Gov4Git proposal [26](#).

Add a comment

Write Preview

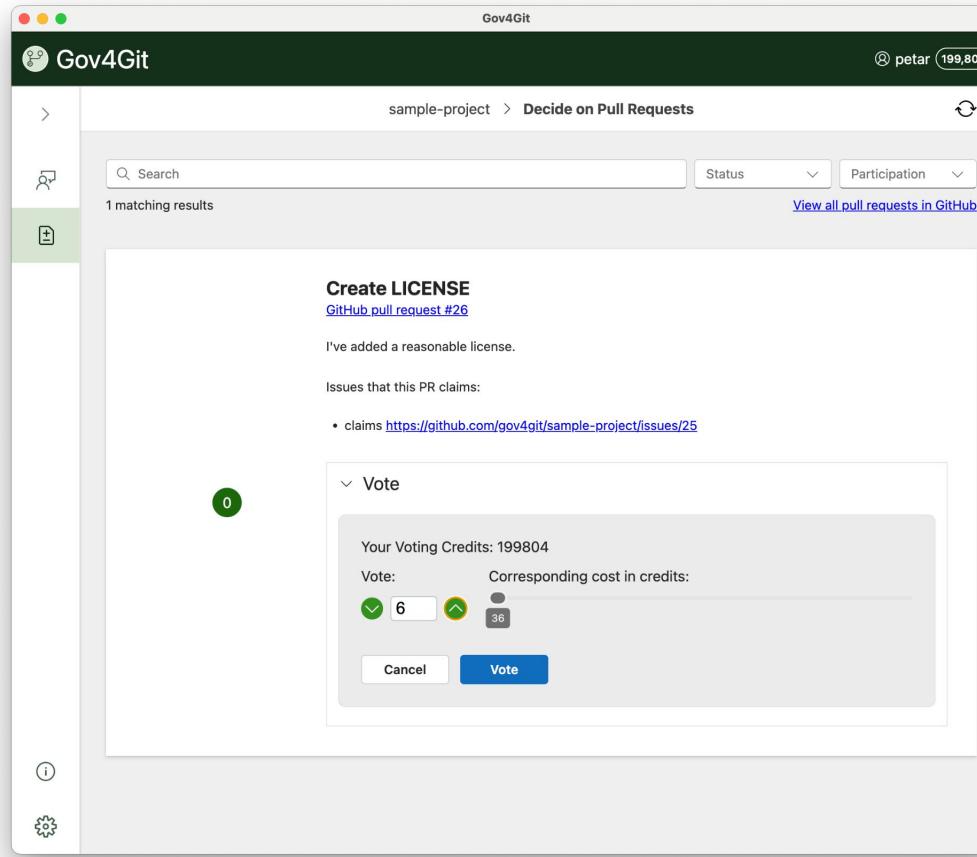
Add your comment here...

Markdown is supported Paste, drop, or click to add files

Close issue Comment

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

# Collaborators can now vote on the PR



All collaborators will see the PR in their desktop app, and be able to cast a vote.

Multiple votes (up or down) can be cast at any time, while the PR is open.

# PRs with positive votes are considered “eligible”

The screenshot shows a GitHub pull request page for a repository named "gov4git/sample-project". The PR is titled "Create LICENSE #26" and has an approval score of 6.000000. A comment from "gov4git-bot" indicates that the cost of review is 36.000000 and the projected bounty is 0.000000. Another comment from "Gov4Git notices" states that the set of eligible issues claimed by this PR is empty. A third comment from "qhudve" shows that the approval score is now 6.000000, the cost of review is 36.000000, and the projected bounty is now 28.000000. A fourth comment from "bstqin" indicates that the set of eligible issues claimed by this PR changed, listing one issue: "our project needs a LICENSE #25, managed as Gov4Git concern 25 with priority score of 14.000000". A note at the bottom of the page says "Add more commits by pushing to the petar-patch-4 branch on gov4git/sample-project."

① Votes cast on a PR will be reflected on the PR page, after the system processes them.

② As soon as a PR has a positive approval score (from votes), it is considered an “eligible” claim on the issue(s) it is claiming.

# An eligible PR will freeze an issue until the PR is accepted or rejected

The screenshot shows a GitHub issue page for a sample project. The issue is titled "our project needs a LICENSE #25" and is labeled "open". A comment from "gov4git-bot" is visible, followed by a reply from "Gov4Git notices". The "Gov4Git notices" reply contains several key pieces of information:

- This issue's priority score is now `14.000000`.
- The cost of priority is `196.000000`.
- The projected bounty is now `28.000000`.

Below this, under "Notice ax5gx3", it says "The set of eligible proposals claiming this issue changed:" with a link to "Create LICENSE #26". Under "Notice ijmpqg", it says "Freezing this issue as there are eligible PRs addressing it:" with another link to "Create LICENSE #26".

A large callout bubble on the left side of the screenshot contains the following text:

**①** The issue's page will also be updated to reflect that there is an "eligible" claim on it.

A large callout bubble on the right side of the screenshot contains the following text:

**②** When there is an "eligible" claim on an issue, the issue is "frozen", meaning that no one can cast votes on it any longer – until the PR addressing it is either accepted or rejected.

# Maintainers accept a PR by merging it

A screenshot of a GitHub pull request (PR) merge dialog. The URL in the address bar is [github.com/gov4git/sample-project/pull/26](https://github.com/gov4git/sample-project/pull/26). The title of the dialog is "Create LICENSE #26" and it says "I'm Open". It shows a message: "petar wants to merge 1 commit into `main` from `petar-patch-4`".

The main content area displays several notices:

- Notice `dhoxin`: This PR's approval score is now `6.000000`. The cost of review is `36.000000`. The projected bounty is now `0.000000`.
- Notice `qvudhe`: This PR's approval score is now `6.000000`. The cost of review is `36.000000`. The projected bounty is now `28.000000`.
- Notice `bstqin`: The set of eligible issues claimed by this PR changed:
  - [our project needs a LICENSE #25](#), managed as Gov4Git concern `25` with priority score of `14.000000`

At the bottom, there is a note: "Add more commits by pushing to the `petar-patch-4` branch on [gov4git/sample-project](#)".

A modal dialog is open at the bottom, titled "Merge pull request #26 from gov4git/petar-patch-4". It contains fields for "Merge pull request #26 from gov4git/petar-patch-4" and "Create LICENSE". A dropdown menu for "Choose which email address to associate with this commit" has "petarm@gmail.com" selected. At the bottom are "Confirm merge" and "Cancel" buttons.

**① Maintainers accept a PR by merging it.**

**Maintainers can reject a PR by closing it (unmerged).**

# Accepting a PR triggers distribution of awards and bounties

The screenshot shows a GitHub pull request page for a sample project. The PR has been merged, and the title is "Create LICENSE #26". The PR details section shows "petar merged commit 15a836b into main 1 minute ago". A comment from "gov4git-bot" is visible, stating "Gov4Git notices" and "Notice oypmtr". The text in the comment area includes:

- This PR, managed as Gov4Git proposal 26, has been accepted 🎉
- The PR approval score was 6.000000.
- Issues claimed and resolved by this PR were:
  - [Issue #25](#)
- The realized bounty for the author of this PR was 40.000000, comprising:
  - A priority bounty of 28.000000, and
  - A review bounty of 12.000000.
- PR reviewers were rewarded as follows:
  - Reviewer [@petar](#) was rewarded 36.000000 credits
- The cost of review of this PR was 36.000000.
- PR approval tally by reviewer was:
  - Reviewer [@petar](#) contributed 6.000000 votes

Two callout boxes highlight specific points:

- ① After a PR is merged, the governance system will award all participants (author and reviewing voters) and post a report on the PR page.**
- ② The details of awards, penalties, and refunds depend on the specific management protocol in use.**

# Merging a PR also updates the claimed issue(s)

The screenshot shows a GitHub issue page for a sample project. The issue, titled "our project needs a LICENSE #25", was opened by user "petar" and has 5 comments. A purple "Closed" button indicates the issue is now resolved. The last comment was made by "gov4git-bot" and states: "gov4git-bot closed this as completed now".

A large speech bubble on the left contains the following text:

**① Merging the PR will also result in distributing awards, penalties, or refunds to the collaborators who voted on the claimed issue, and the issue will be closed.**

A large speech bubble on the right contains the following text:

**② The details of awards, penalties, and refunds depend on the specific management protocol in use.**

The GitHub interface includes standard navigation elements like back, forward, and search, along with social sharing and issue filtering buttons.