Git and GitHub

✨ *"GitHub Organization – A unified hub for teams to securely collaborate, manage, and ship code at scale."*

Description:

GitHub Organization is a collaborative workspace that brings together people, teams, and repositories under one umbrella. It enables centralized management of projects, user permissions, security policies, and workflows, making it ideal for teams and enterprises to build, review, and ship code efficiently and securely.

- **GitHub Organizations**
- **GitHub Repository's permissions**
- **GitHub Branches protection**
- **Collaborators (Give the Specific user permissions)**
- **Pull Requests (Branching & Merging)**

These are Some Specific Features/Toggles in the GitHub Organization:

- ❖ **Organization roles**
    1. **Role management**
    2. **Role Assignment**
- ❖ **Repository Roles**
- ❖ **Member Privileges**
- ❖ **Moderation**
    1. **Block Users**
    2. **Interaction Limits**
    3. **Code Review Limits**
    4. **Moderators**
- ❖ **Two Factor Authentication**

GitHub Organization:

GitHub and manage its privacy:

➔**Create a New Organization:**

- Log in to your GitHub account.
- Click your profile photo in the upper-right corner of any page, then click Settings.

- In the "Access" section of the sidebar, click Your Organizations.
- Click New organization next to the "Organizations" header.
- Follow the prompts to create your organization, including choosing a plan (GitHub Free, Team, or Enterprise).

➔Add users to the org with **role-based permissions**:

- **Read** → can only view code (still allows download unless combined with other controls).
- **Write** → can push commits.
- **Triage** → Manage issues/PRs without write.
- **Maintain** → repo settings (without admin).
- **Admin** → full control.

- Use **Teams** (e.g., *Dev*, *QA*, *Ops*) with repo-level permissions.
- Restrict **production repo** to *Admin + Ops team only*. Developers can be given access only to dev/staging repos.

➔**Prevent Direct Zip/Clone Access**

- By default, if a user has **read** access, they can download the repo as .zip or clone it.
- To block this, you must **avoid giving read access** at all.

Alternatives:

- **Private repositories**: Only explicitly added members can access.
- **Deploy Keys / GitHub Apps**: Give your CI/CD pipeline access without letting developers download the repo directly.
- **Code Owners**: Enforce code reviews before merges.
- **Branch Protection Rules**: Prevent force-push, require PR reviews, and restrict who can merge into production branches.

Note: **GitHub does not provide a direct "disable zip download" toggle. If someone has read access, they can always get the code.**

Workaround:

- Keep the **production repo private**.
- Developers work only on **mirror repos** or **fork of staging**.
- Only CI/CD pipelines (via tokens/keys) pull from the **production repo** to deploy.

➔**Manage Repository Visibility:**

- Within your organization, you can create repositories and set their visibility to Private. This ensures that only members of your organization, or specific teams within it, can access the repository's code and content.

➔**Control Member and Team Visibility:**

- You can choose to hide your individual membership within an organization by setting your visibility to Private in your

organization's "People" settings.

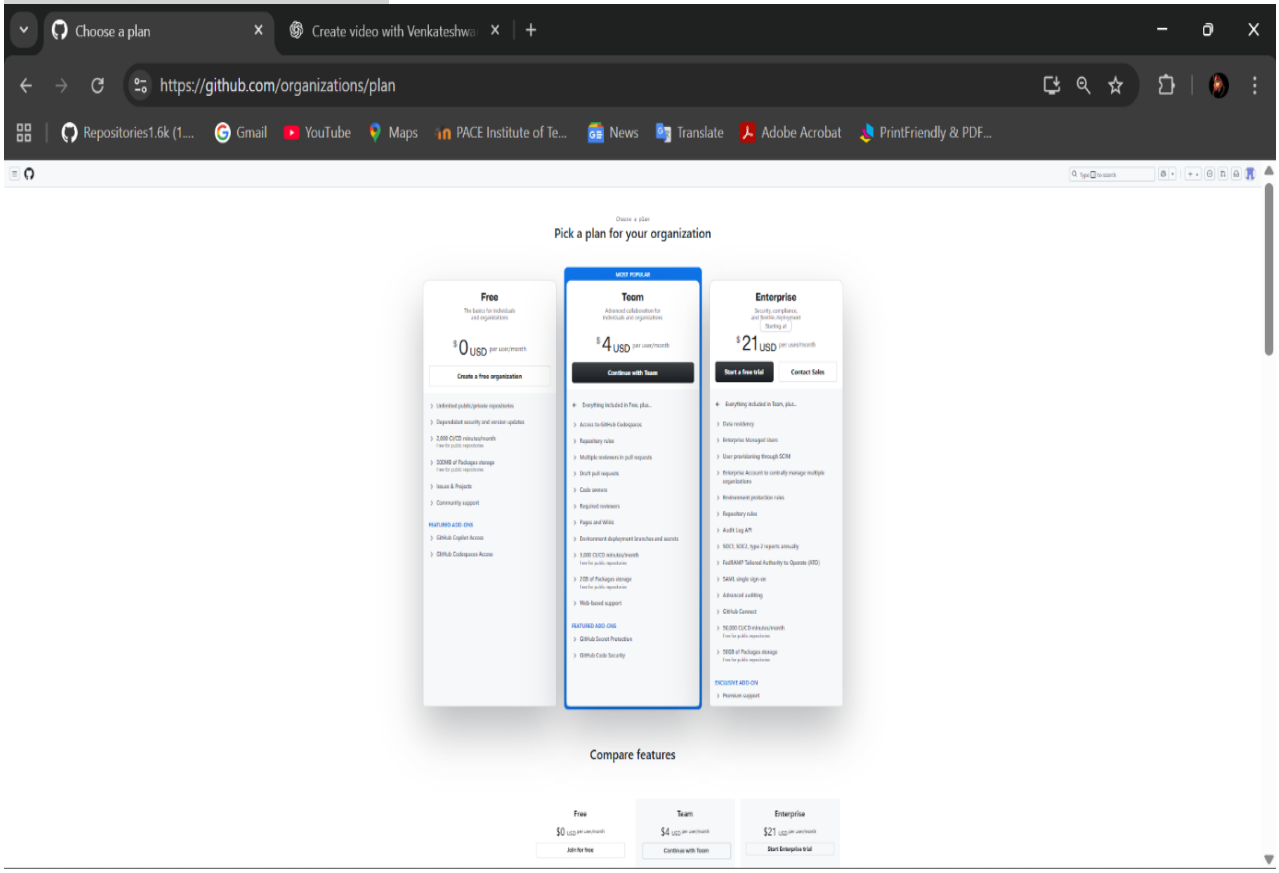
- You can also manage the visibility of teams within your organization, controlling who can see and interact with them.

➔**Fine-grained Access Control with Teams:**

- For private repositories within your organization, you can create teams and grant specific access levels (e.g., read, write, admin, maintain, triage) to those teams on a per-repository basis. This allows for precise control over who can access and contribute to your private projects.

By combining these features, you can create a highly secure and private environment for your projects and collaborations within a GitHub Organization

Step-By-Step Process:



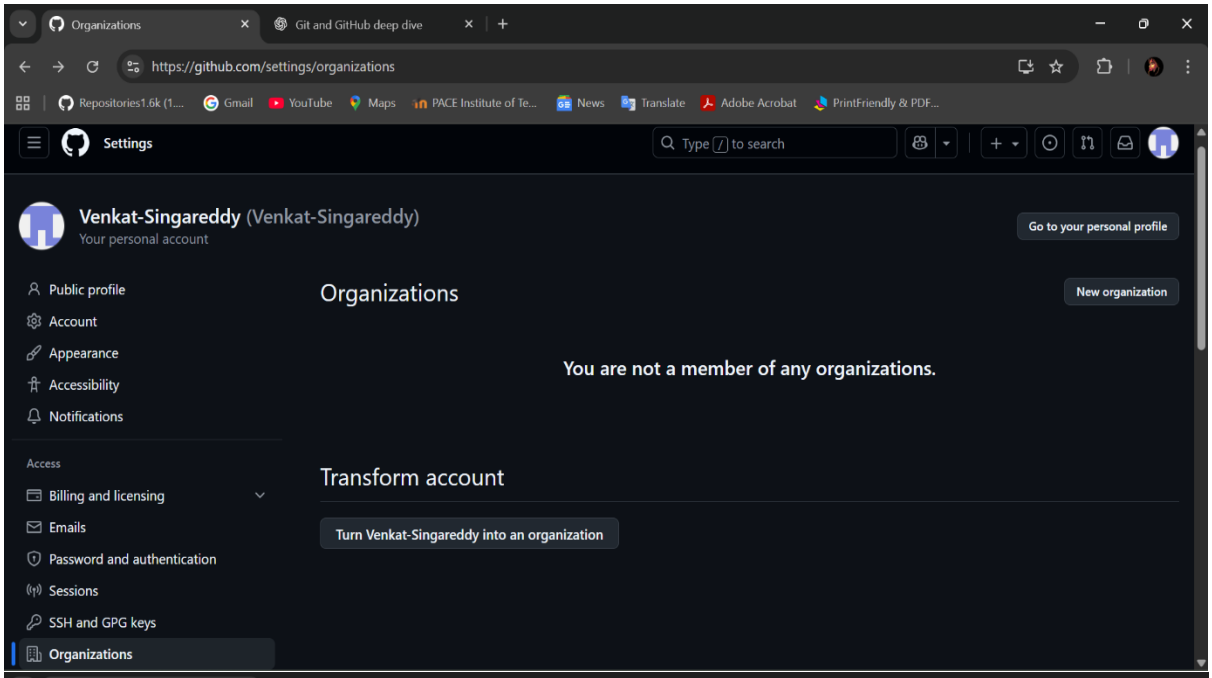
➔Here Select your organization it may be free, Team, Enterprise level

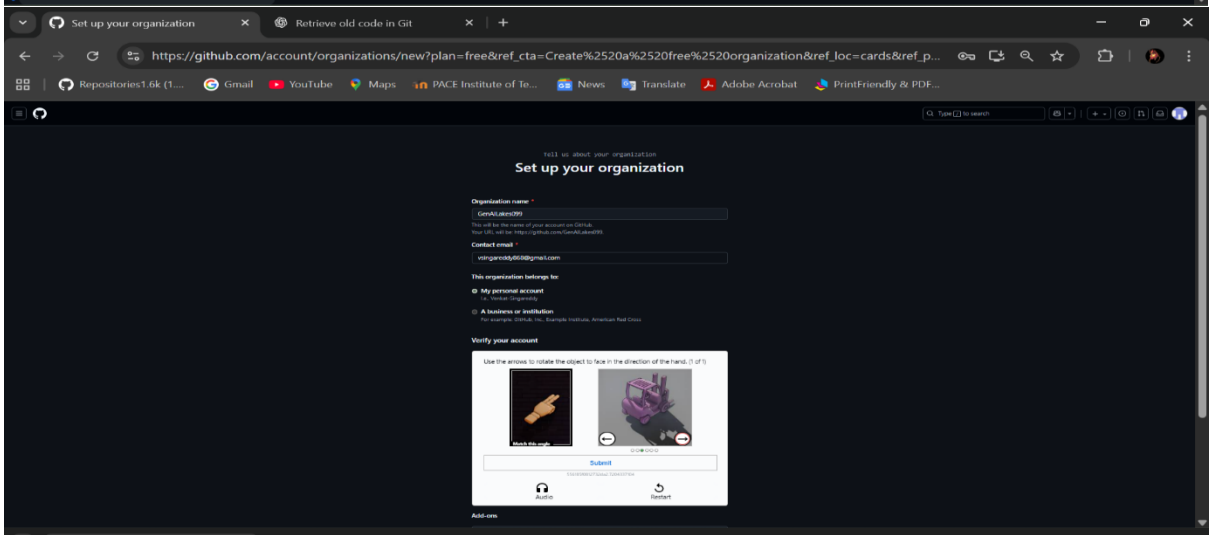
➔In the Enterprise level we have two types on-premises & off-premises Deployment

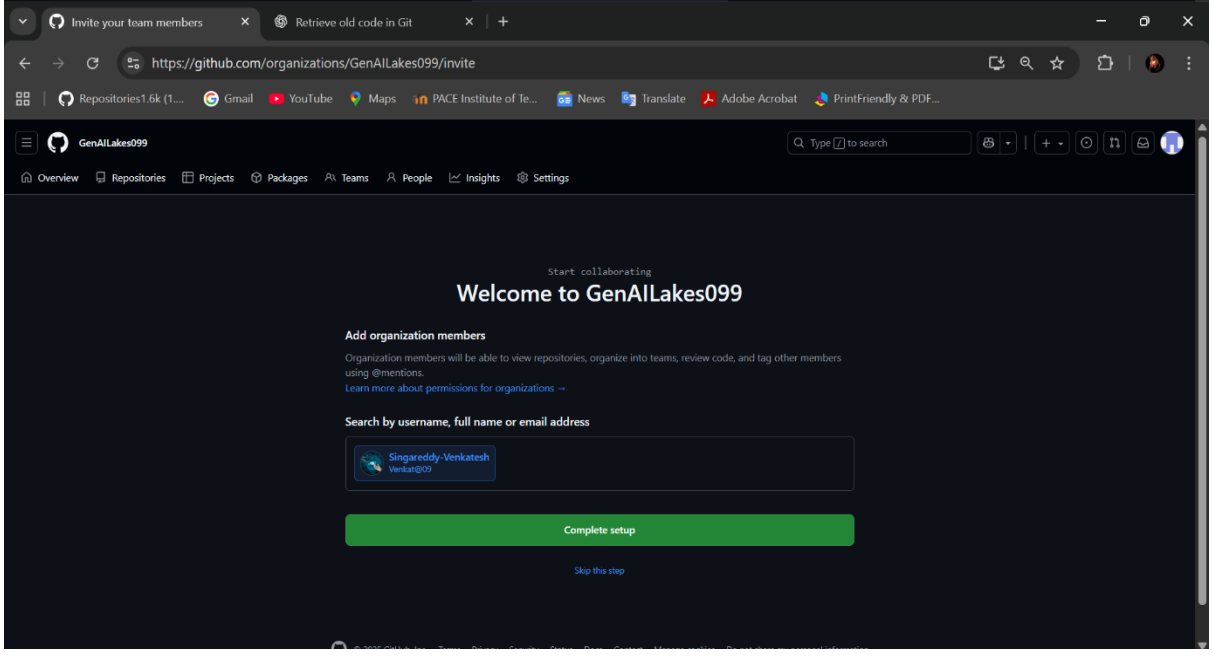
On-premises: In these we have GitHub website is going to use or GitHub Cloud.

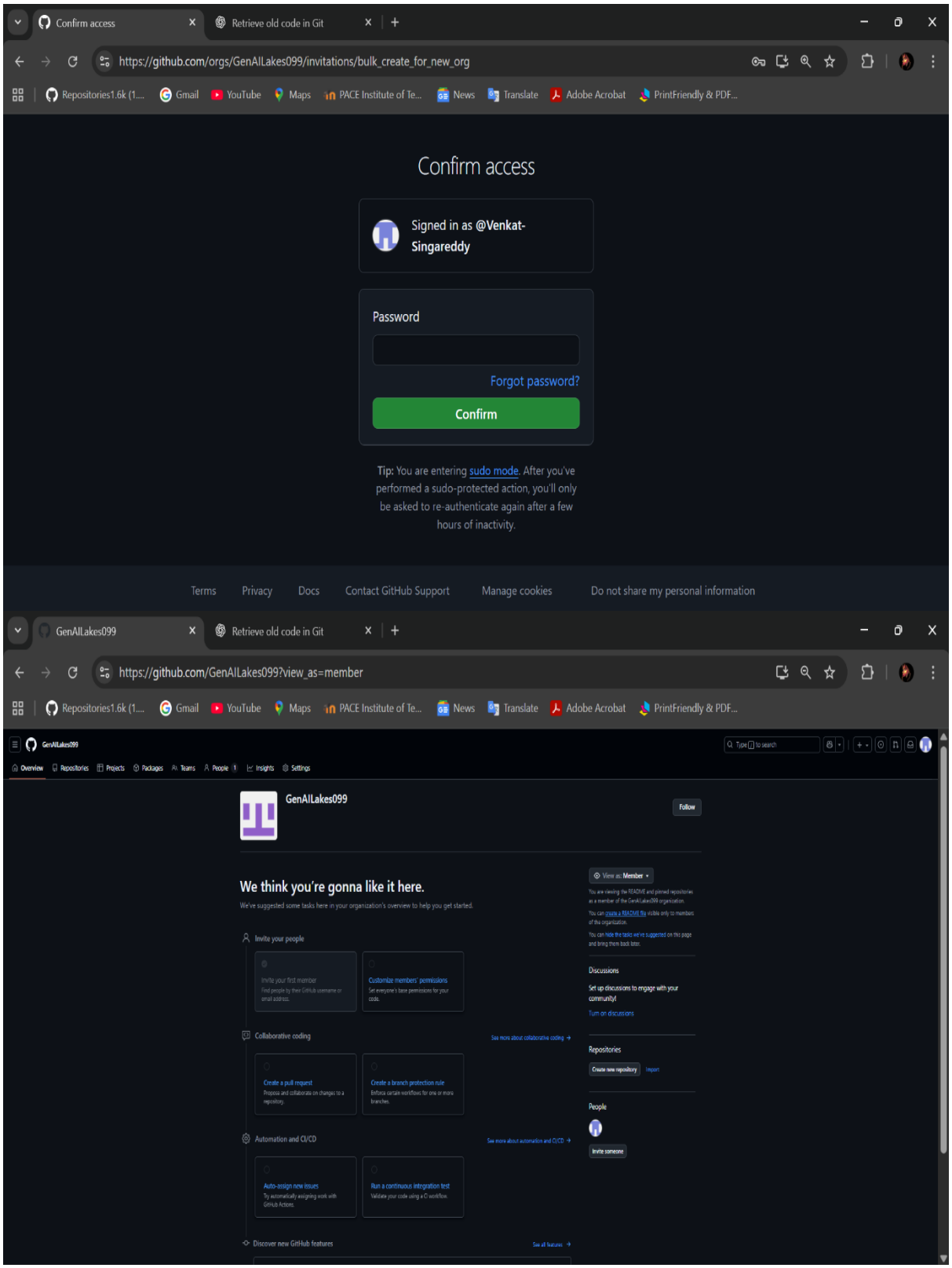
Off-premises: In these we have to use Physical Resources it may be physical servers or any other else.

➔Go to Organizations create new organization


Organizations
Git and GitHub deep dive
https://github.com/settings/organizations
Settings
Type / to search
Venkat-Singareddy (Venkat-Singareddy)
Your personal account
Go to your personal profile
Public profile
Account
Appearance
Accessibility
Notifications
Access
Billing and licensing
Emails
Password and authentication
Sessions
SSH and GPG keys
Organizations
Organizations
New organization
You are not a member of any organizations.
Transform account
Turn Venkat-Singareddy into an organization


Set up your organization
Retrieve old code in Git
https://github.com/account/organizations/new?plan=free&ref_cta=Create%2520a%2520free%2520organization&ref_loc=cards&ref_p...
Set up your organization
Organization name
GenAILakes099
Contact email
Verify your account
Submit

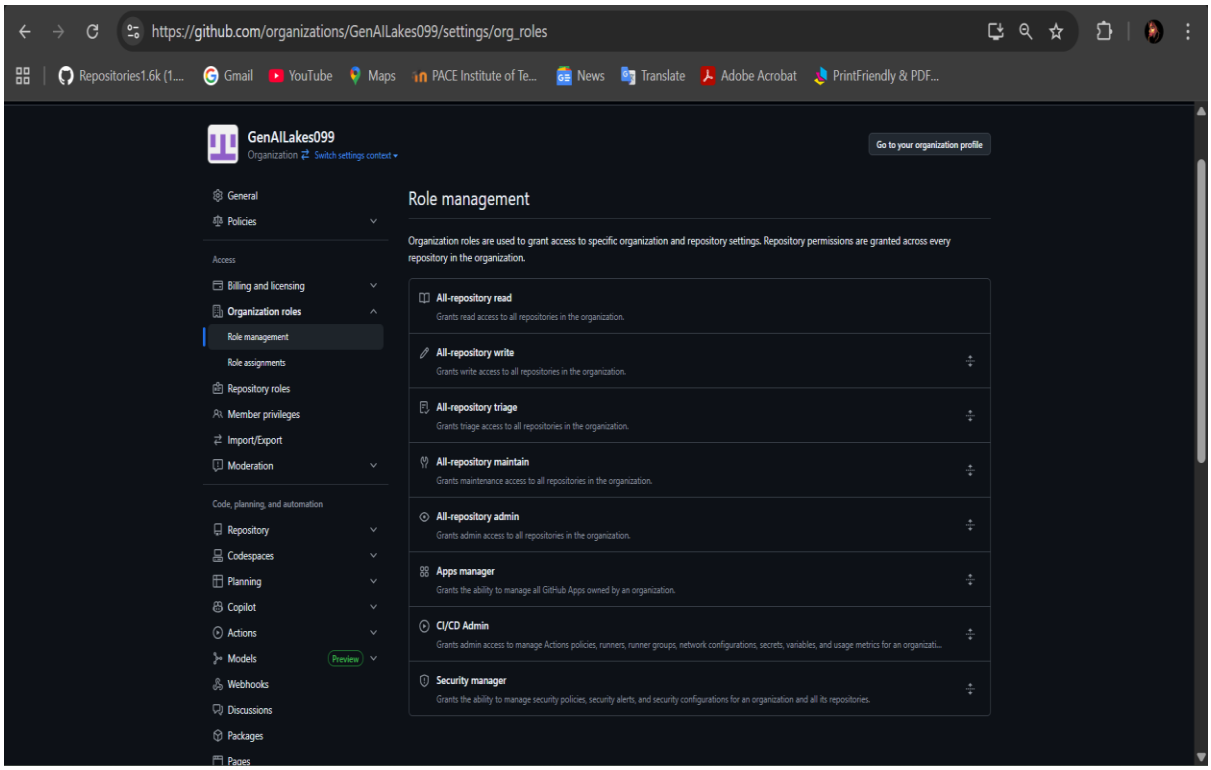

Invite your team members
Retrieve old code in Git
https://github.com/organizations/GenAILakes099/invite
GenAILakes099
Overview
Repositories
Projects
Packages
Teams
People
Insights
Settings
Start collaborating
Welcome to GenAILakes099
Add organization members
Organization members will be able to view repositories, organize into teams, review code, and tag other members using @mentions.
Learn more about permissions for organizations →
Search by username, full name or email address
Singareddy-Venkatesh
Venkat@03
Complete setup
Skip this step



❖ Organization roles

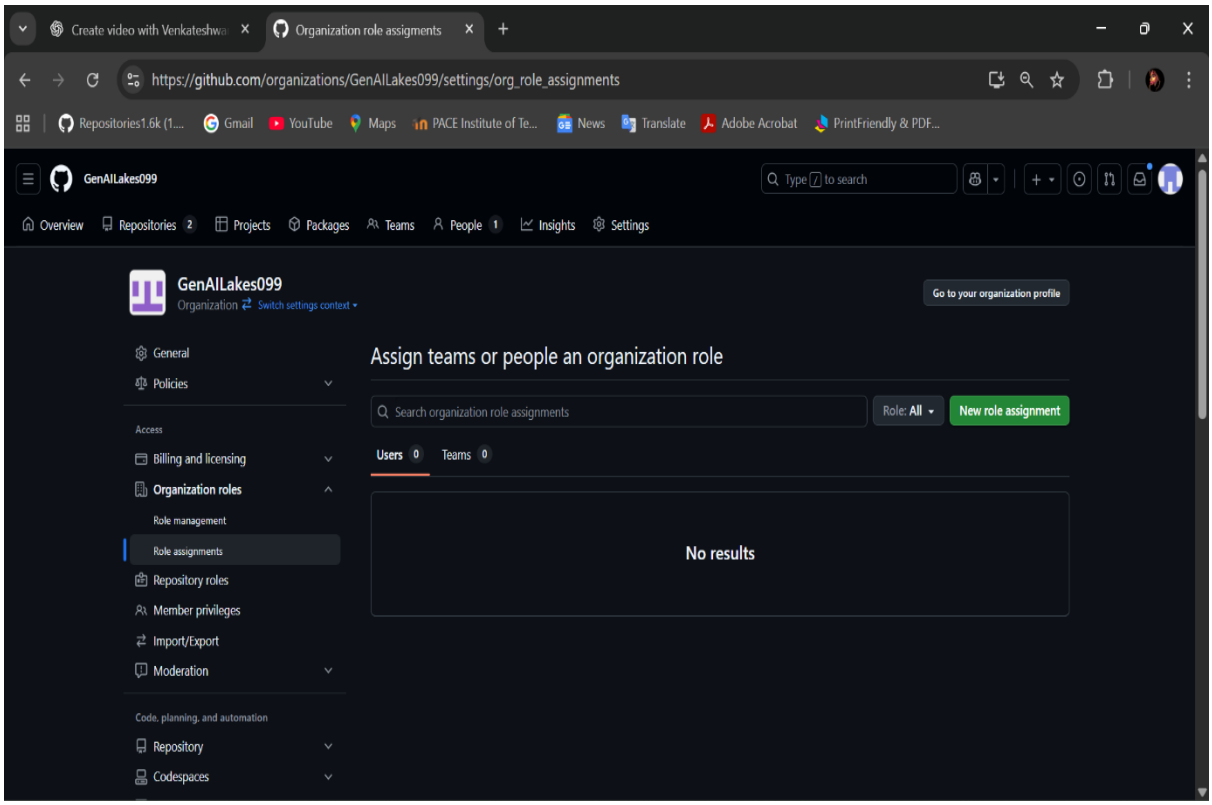
1. Role management

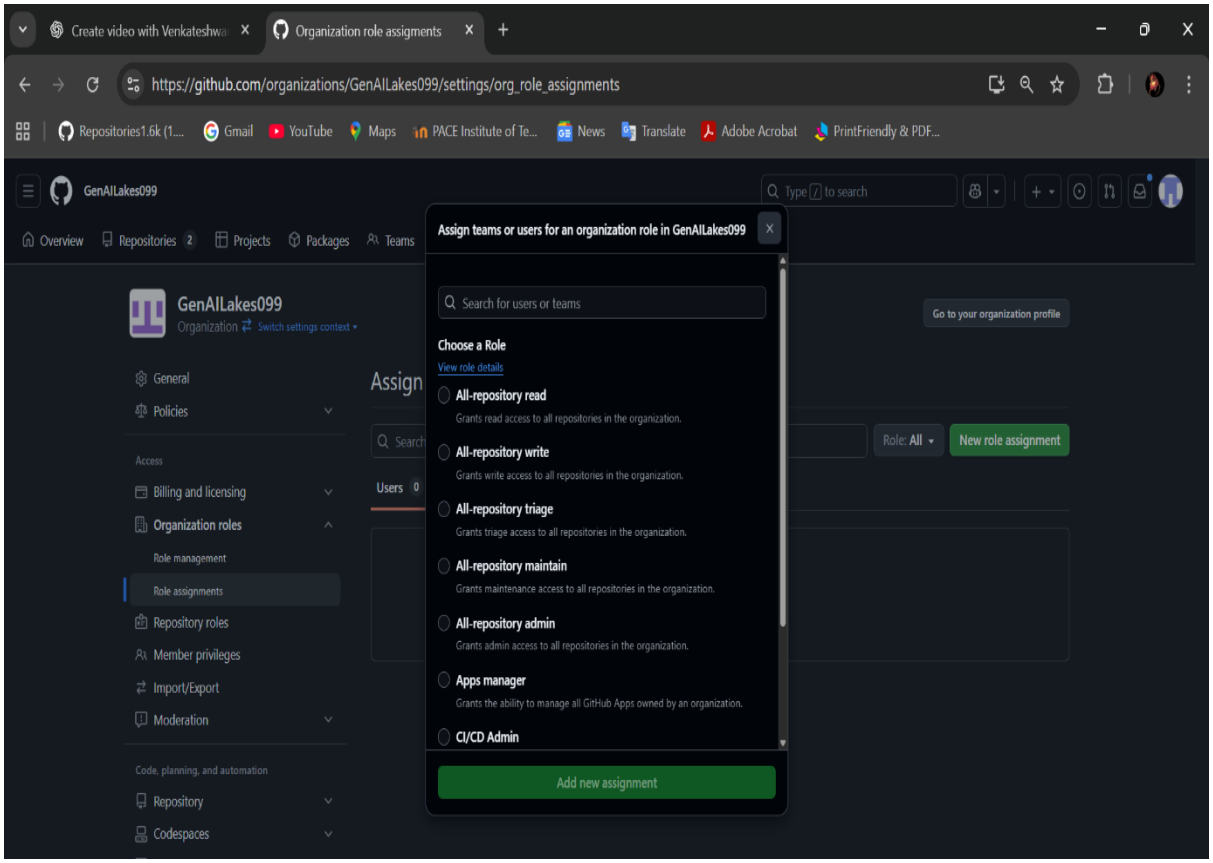
➔In Your Organization you can assign Role Management



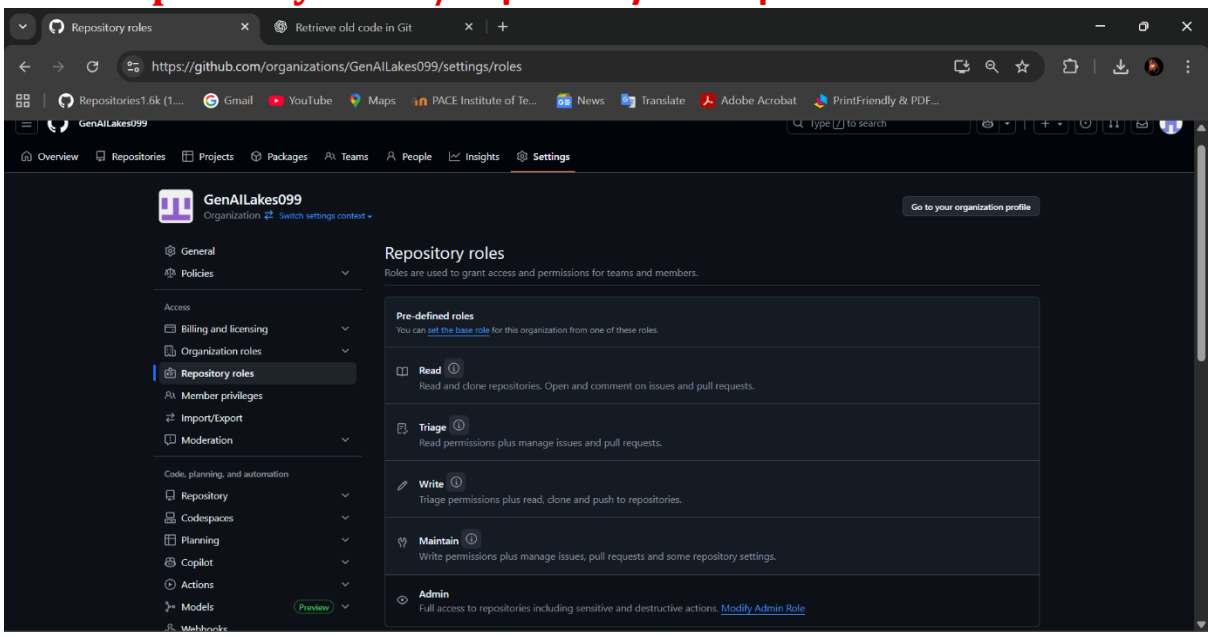
2. Role Assignment

➔Here you can assign specific role for users or team members





❖ Repository Roles/Repository Level permissions:

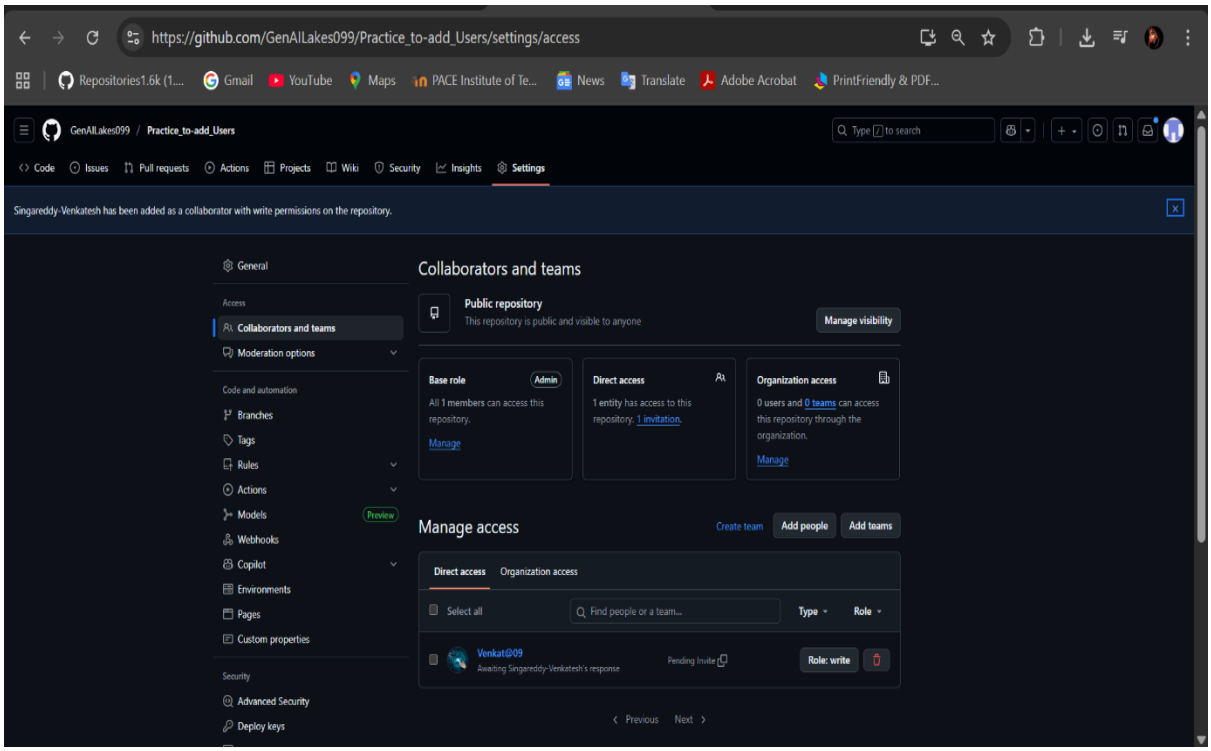


Collaborators & Team:

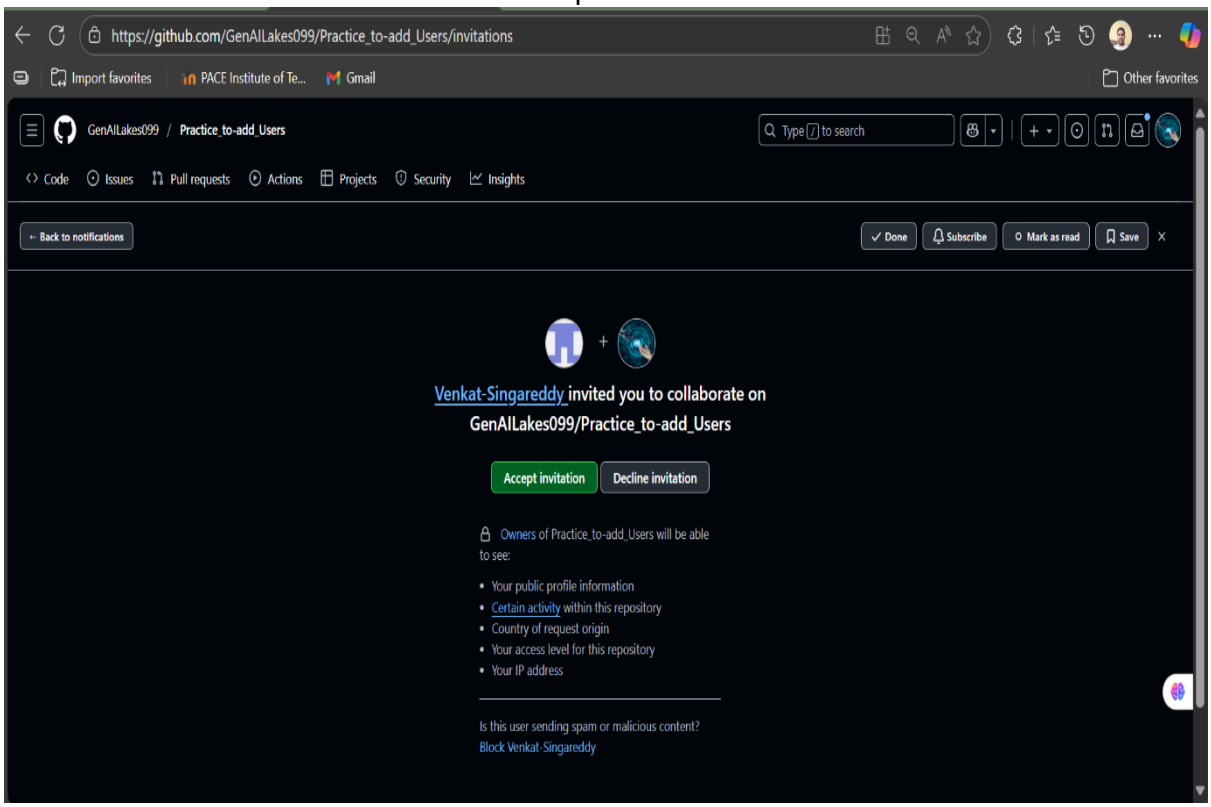
➔In the collaborators & team => manage access you can (**add peoples, create team, add teams**) by assigning the Permissions like (**Read, Write, Triage, Maintain, Admin**)

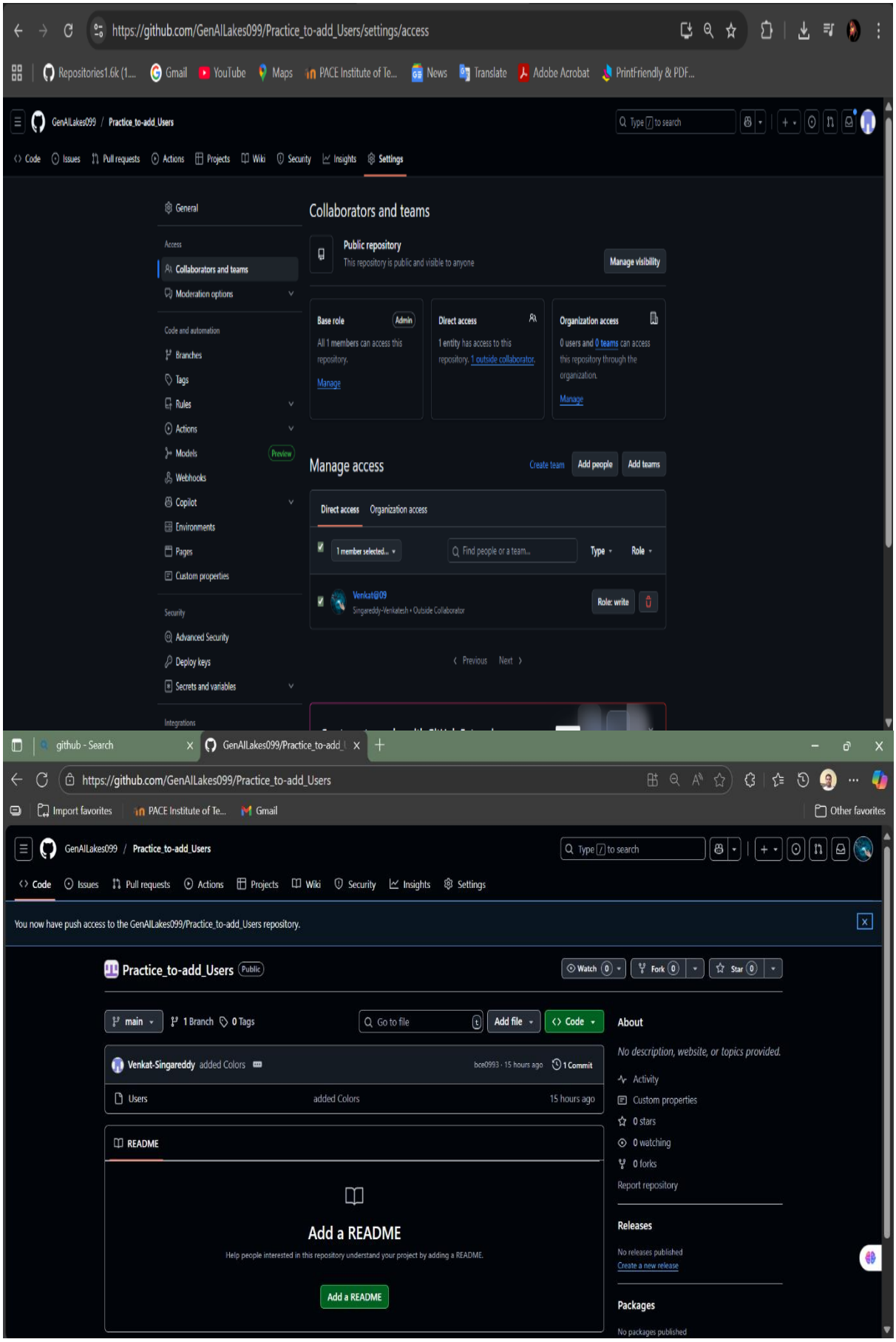
➔In the Add people invite the member by using the Email, GitHub Account, Phone Number) after you send invitation to the other side of the User can

accept or decline in there Github Account in the Notification.
➔After accepting the user, you can see in you **manage access** outside collaborators the previous pending notification will removed.
➔Now the User can also see your repository.

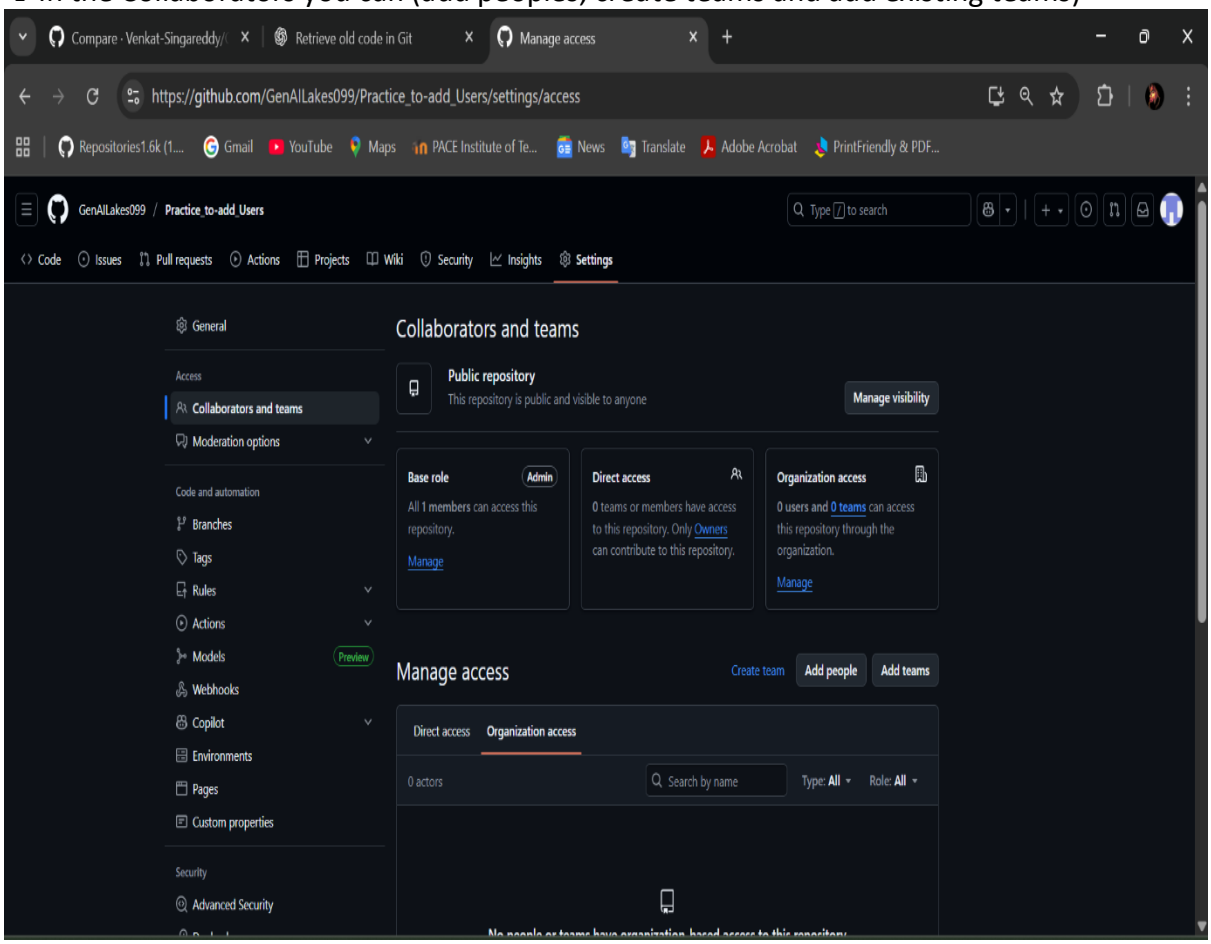


➔Another GitHub Account he want to Accept the Invitation




https://github.com/GenAILakes099/Practice_to-add_Users/settings/access
GenAILakes099 / Practice_to-add_Users
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
General
Access
Collaborators and teams
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Models
Webhooks
Copilot
Environments
Pages
Custom properties
Security
Advanced Security
Deploy keys
Secrets and variables
Integrations
Collaborators and teams
Public repository
This repository is public and visible to anyone
Manage visibility
Base role
Admin
All 1 members can access this repository.
Manage
Direct access
1 entity has access to this repository. 1 outside collaborator.
Organization access
0 users and 0 teams can access this repository through the organization.
Manage
Manage access
Create team
Add people
Add teams
Direct access
Organization access
1 member selected...
Find people or a team...
Type
Role
Venkat@09
Singareddy-Venkatesh • Outside Collaborator
Role: write
Previous
Next
github - Search
GenAILakes099/Practice_to-add_
https://github.com/GenAILakes099/Practice_to-add_Users
Import favorites
PACE Institute of Te...
Gmail
Other favorites
GenAILakes099 / Practice_to-add_Users
Type / to search
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
You now have push access to the GenAILakes099/Practice_to-add_Users repository.
Practice_to-add_Users
Public
Watch 0
Fork 0
Star 0
main
1 Branch
0 Tags
Go to file
Add file
Code
About
No description, website, or topics provided.
Venkat-Singareddy added Colors
bce0993 · 15 hours ago
1 Commit
Users
added Colors
15 hours ago
README
Add a README
Help people interested in this repository understand your project by adding a README.
Add a README
Activity
Custom properties
0 stars
0 watching
0 forks
Report repository
Releases
No releases published
Create a new release
Packages
No packages published

➔In the Collaborators you can (add peoples, create teams and add existing teams)



User Permissions:

Read

- Recommended for non-code contributors who want to view or discuss your project.
- Read and clone repositories. Open and comment on issues and pull requests.

Triage

- Recommended for contributors who need to manage issues and pull requests without write access.
- Read permissions plus manage issues and pull requests.

Write

- Recommended for contributors who actively push to your project.
- Triage permissions plus read, clone and push to repositories.

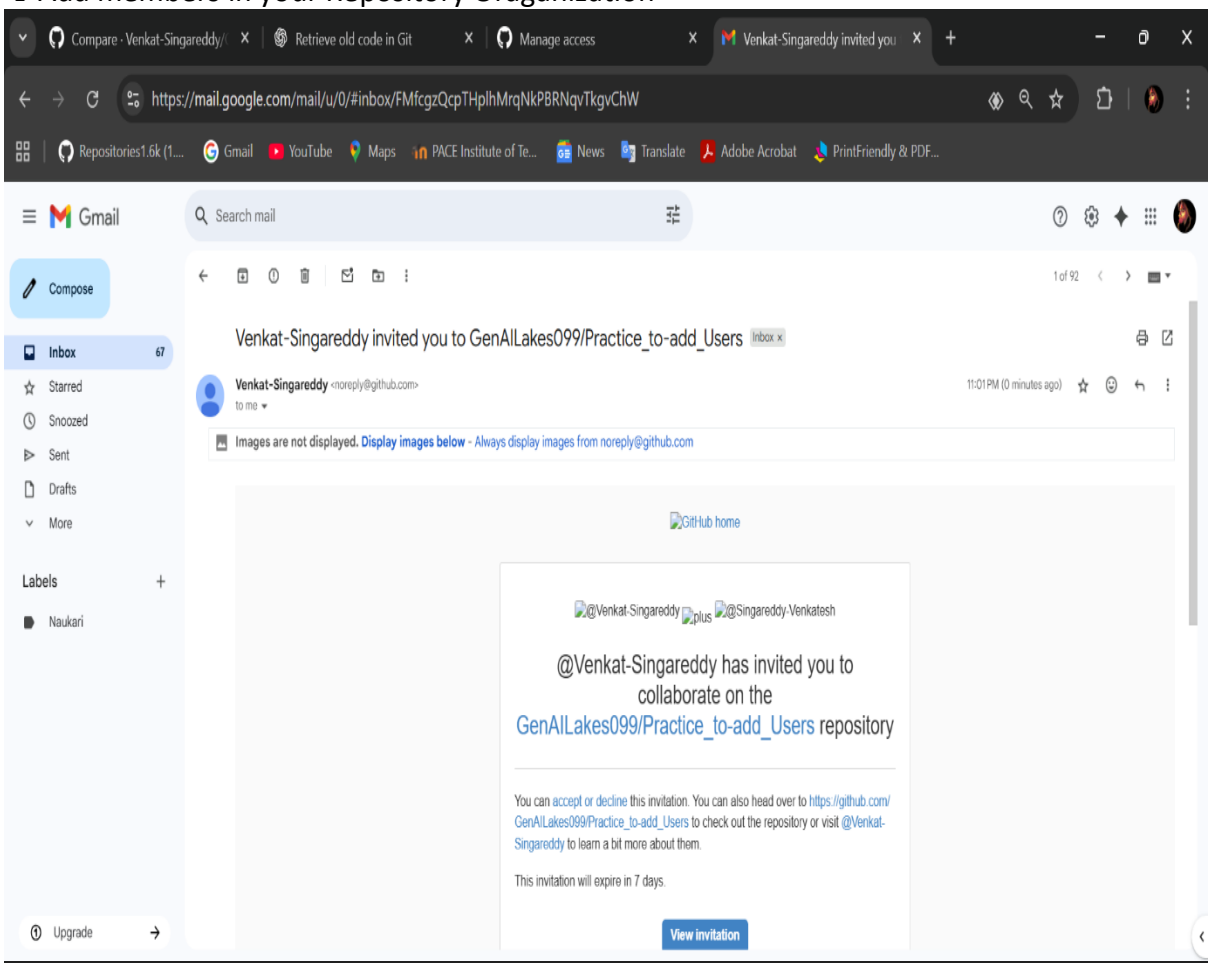
Maintain

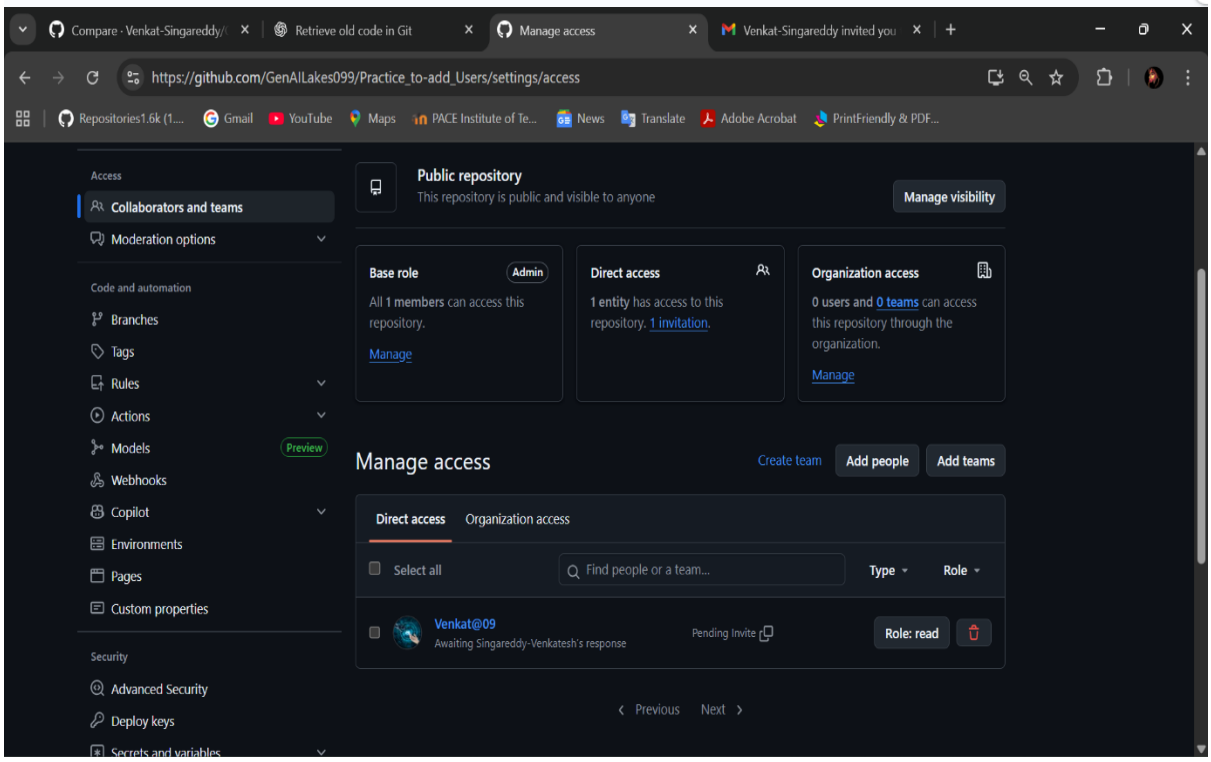
- Recommended for project managers who need to manage the repository without access to sensitive or destructive actions.
- Write permissions plus manage issues, pull requests and some repository settings.

Admin (Base role)

- Recommended for people who need full access to the project, including sensitive and destructive actions like managing security or deleting a repository.
- Full access to repositories including sensitive and destructive actions.

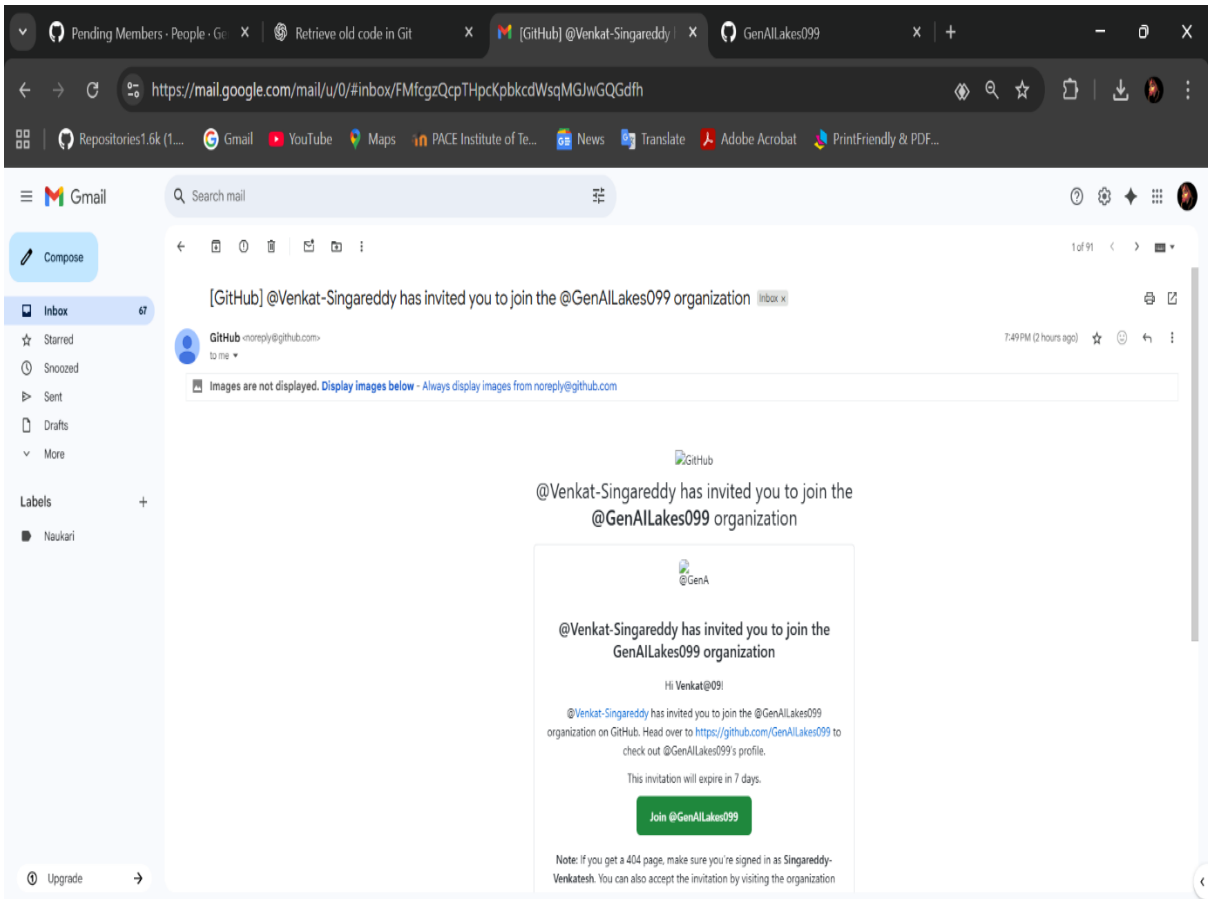
➔Add members in your Repository Oraganization

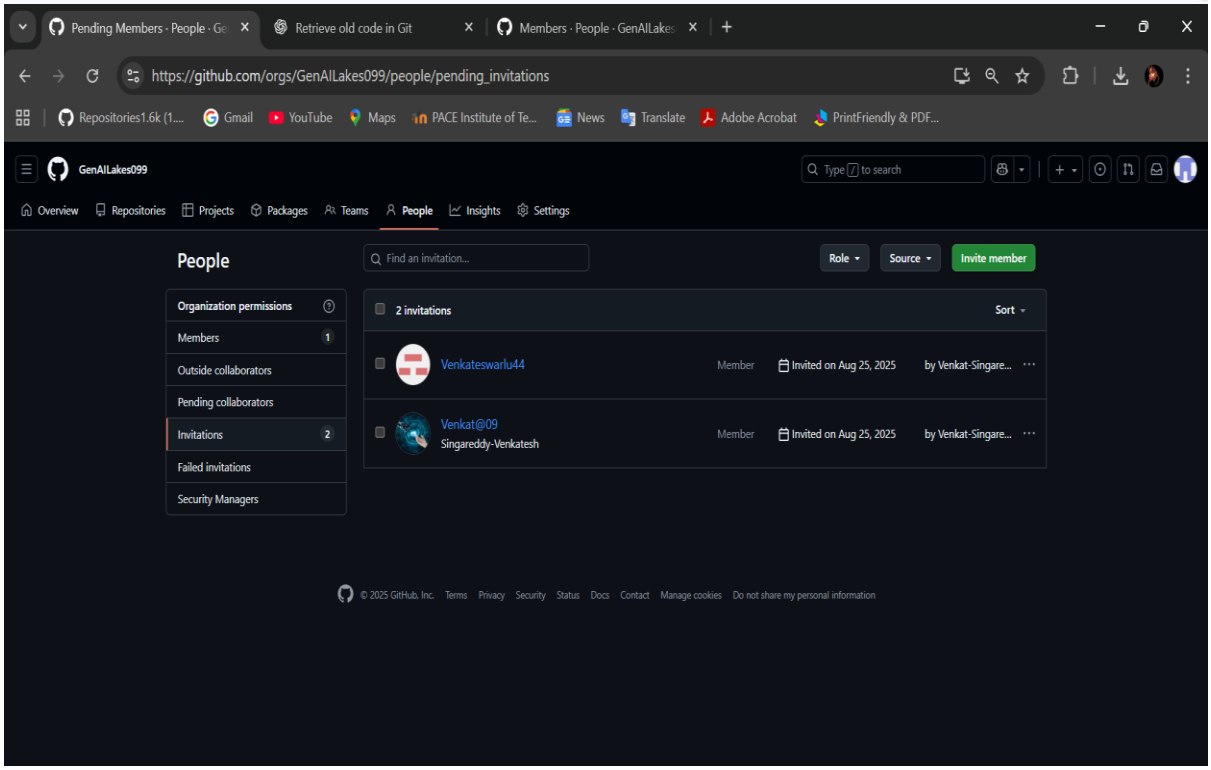




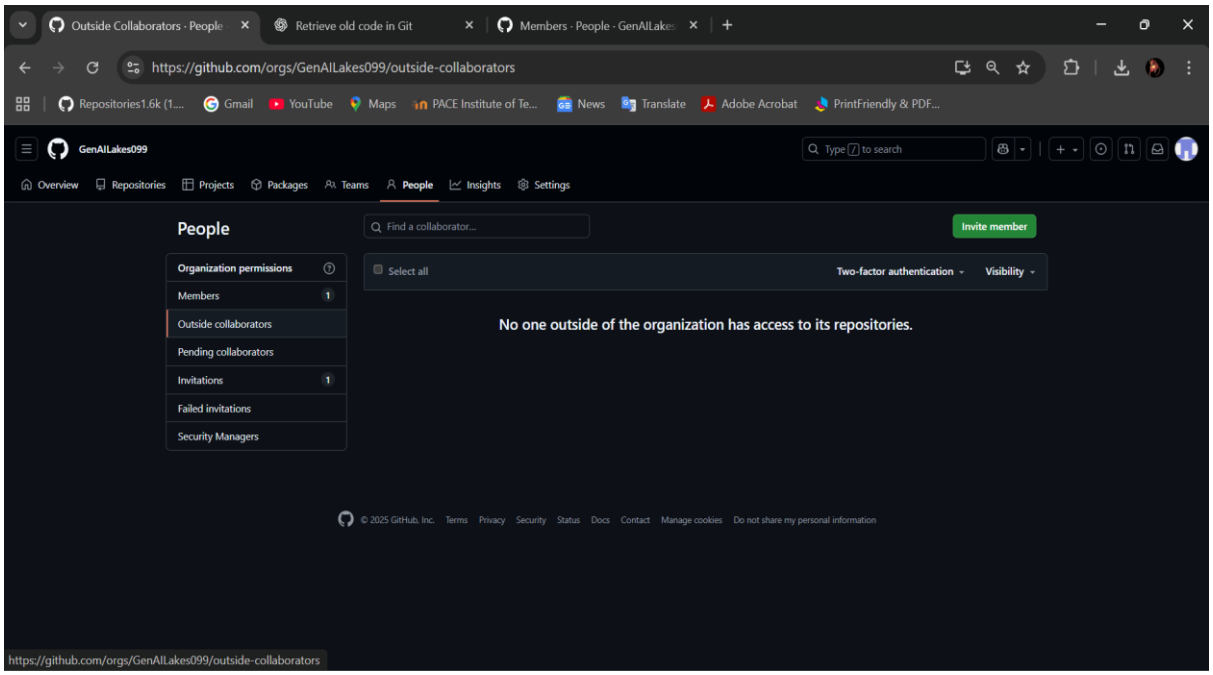
➔You can also change the role or delete the Access

➔You Added Two Members in your Organization (Best recommend a minimum of two people within each organization have the owner role based on the team members)





➔Outside Collaborators: **No one outside of the organization has access to its repositories we can restrict the outside users.**



Branch Level Protection:

➔**You can select and Protect your Repository Branchs of your Organization**

Branch rules:

- **Restrict creations:** Only allow users with bypass permission to create matching refs.
- **Restrict updates:** Only allow users with bypass permission to update matching refs.
- **Restrict deletions:** Only allow users with bypass permissions to delete matching refs.
- **Require linear history:** Prevent merge commits from being pushed to matching refs.
- **Require signed commits:** Commits pushed to matching refs must have verified signatures.
- **Require merge queue:** Merges must be performed via a merge queue.
- **Require deployments to succeed:** Choose which environments must be successfully deployed to before refs can be pushed into a ref that matches this rule.
- **Require a pull request before merging:** Require all commits be made to a non-target branch and submitted via a pull request before they can be merged.
- **Require status checks to pass:** Choose which status checks must pass before the ref is updated. When enabled, commits must first be pushed to another ref where the checks pass.
- **Block force pushes:** Prevent users with push access from force pushing to refs.
- **Require workflows to pass before merging:** Require all changes made to a targeted branch to pass the specified workflows before they can be merged.
- **Require code scanning results:** Choose which tools must provide code scanning results before the reference is updated. When configured, code scanning must be enabled and have results for both the commit and the reference being updated.

Restrictions (Enterprise)

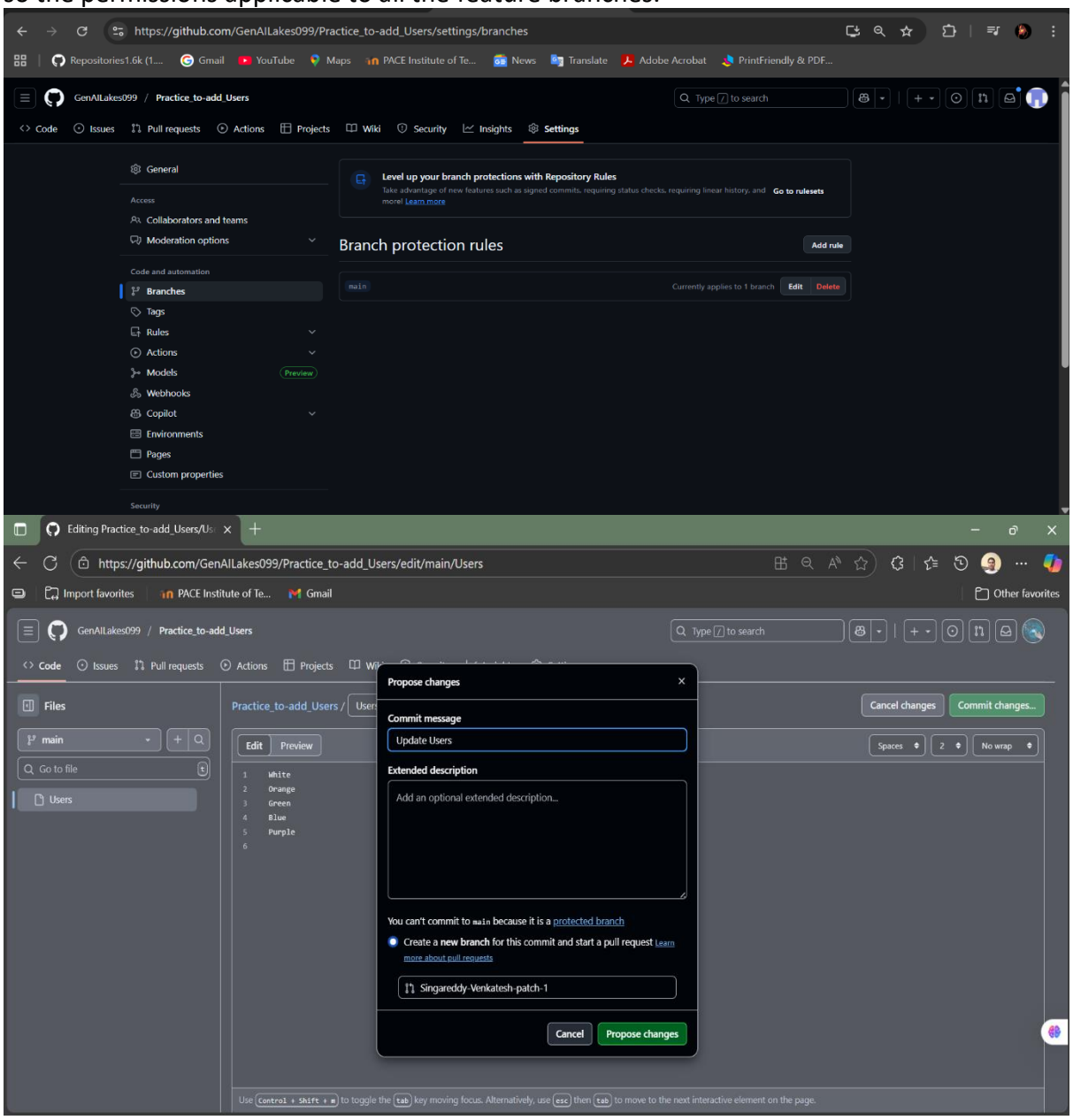
- **Restrict commit metadata:** Restrict commit author email addresses, committer email addresses, commit message content, and other metadata
- **Restrict branch names:** Restrict branch names

Branch Protection rule:

Note: Define branch rules to disable force pushing, prevent branches from being deleted, or require pull requests before merging.

- ❖ Add Branch Ruleset
- ❖ Add Classic Branch Protection Rule

➔In the branch ruleset in the “Ruleset name” suppose if you have multiple feature branches you have to give permissions to all the branches at a time simple type (feature*) so the permissions applicable to all the feature branches.

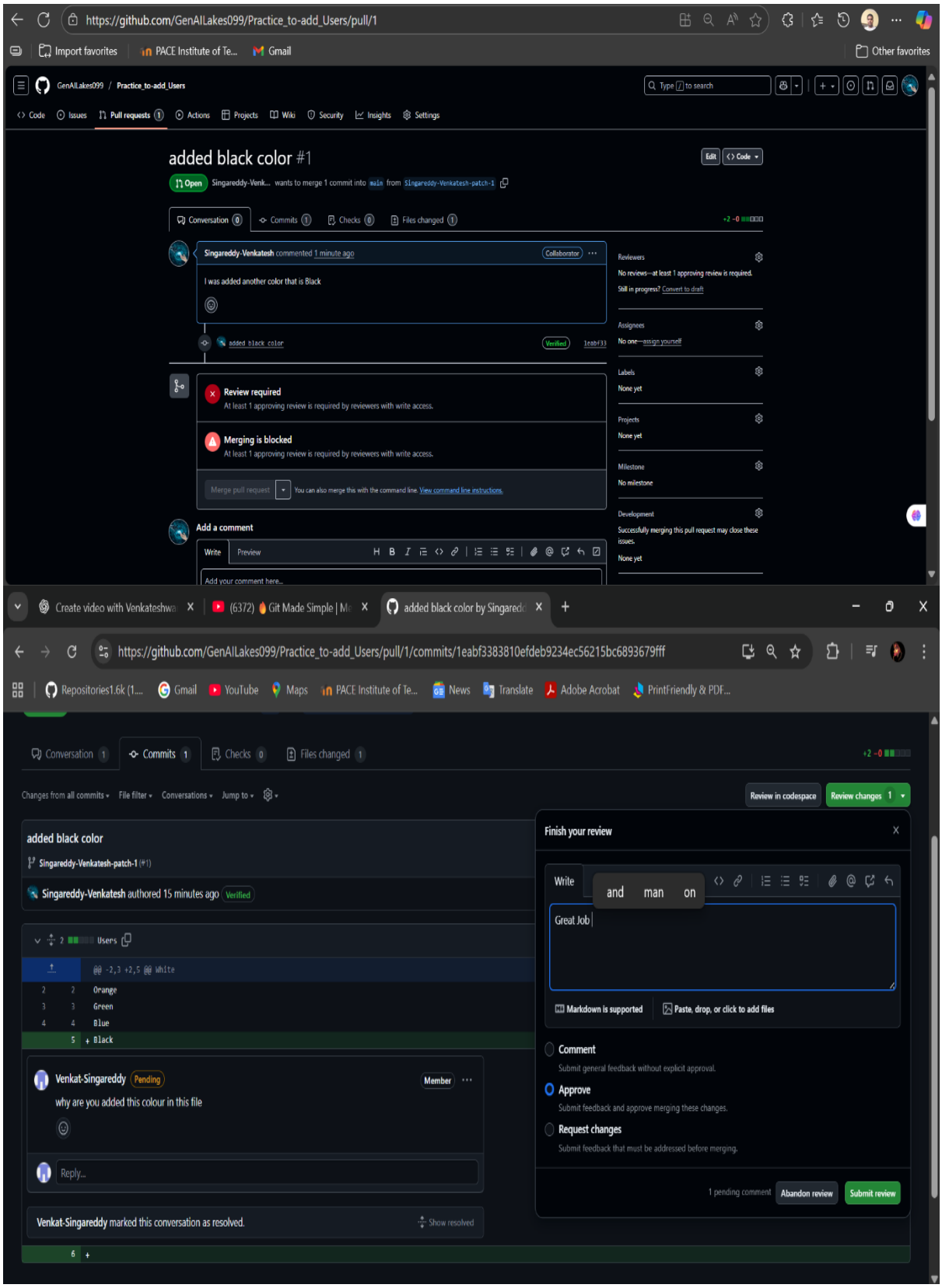


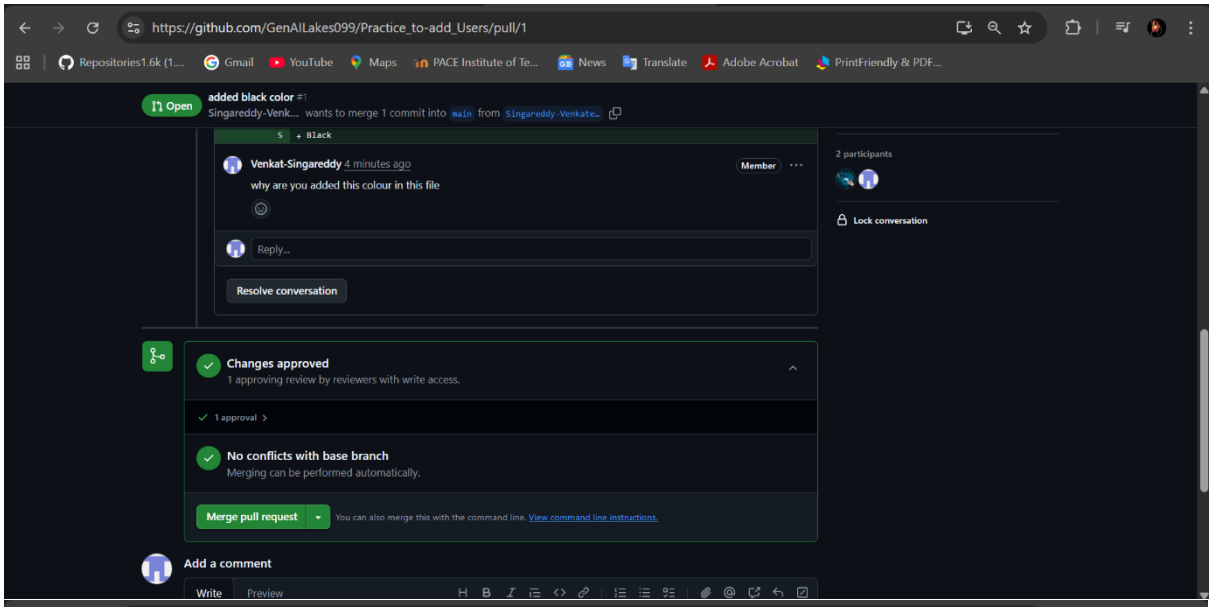
Note: In the above image it restricted to commit the code in the **main branch.**

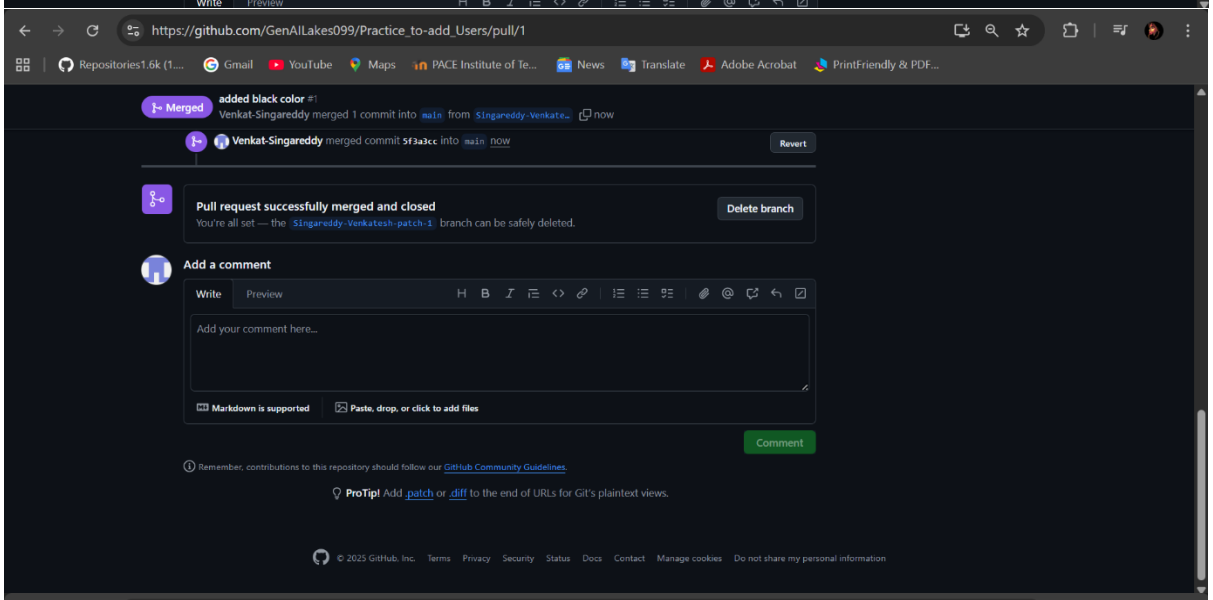
➔In the above scenario when you created a ruleset to Main/Master branch it is very important to protection in the production level, so when the collaborators of your organizations are tried to merge his code in the main branch it will be restrict.

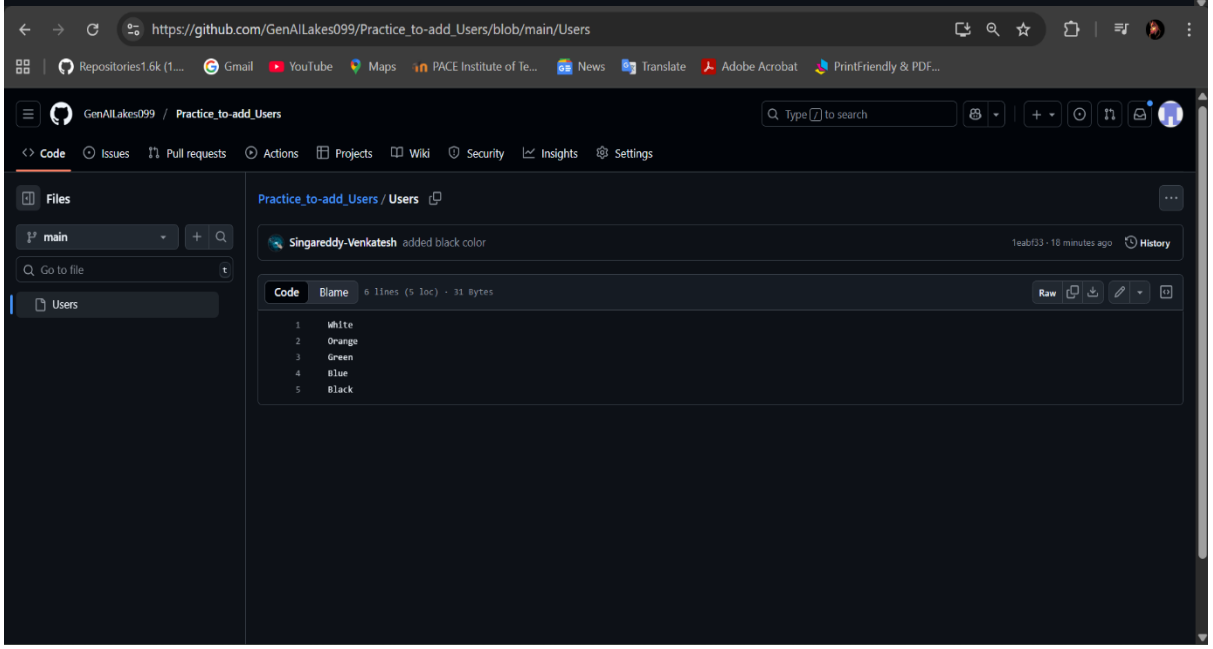
➔Without an approval from your **Admin,** you can’t make changes in the main branch so it is highly protected.

➔when the user makes pull request, so first the **Admin Access members** they can check and review the code before merging in the production environment here we can protect our production code.




https://github.com/GenAILakes099/Practice_to-add_Users/pull/1
Open
added black color #1
Singareddy-Venk... wants to merge 1 commit into main from Singareddy-Venkate...
+ Black
Venkat-Singareddy 4 minutes ago
Member
why are you added this colour in this file
Reply...
Resolve conversation
2 participants
Lock conversation
Changes approved
1 approving review by reviewers with write access.
1 approval
No conflicts with base branch
Merging can be performed automatically.
Merge pull request
You can also merge this with the command line. View command line instructions.
Add a comment
Write
Preview


https://github.com/GenAILakes099/Practice_to-add_Users/pull/1
Merged
added black color #1
Venkat-Singareddy merged 1 commit into main from Singareddy-Venkate... now
Venkat-Singareddy merged commit 5f3a3cc into main now
Revert
Pull request successfully merged and closed
You're all set — the Singareddy-Venkatesh-patch-1 branch can be safely deleted.
Delete branch
Add a comment
Write
Preview
Add your comment here...
Markdown is supported
Paste, drop, or click to add files
Comment
Remember, contributions to this repository should follow our GitHub Community Guidelines.
ProTip! Add .patch or .diff to the end of URLs for Git's plaintext views.
© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information


https://github.com/GenAILakes099/Practice_to-add_Users/blob/main/Users
GenAILakes099 / Practice_to-add_Users
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Files
main
Go to file
Users
Practice_to-add_Users / Users
Singareddy-Venkatesh added black color
1eabf33 · 18 minutes ago
History
Code
Blame
6 lines (5 loc) · 31 Bytes
Raw
1 White
2 Orange
3 Green
4 Blue
5 Black

Member Privileges

➔You can give the specific members Privileges like User permissions in your organization

- **Base permissions**

Base permissions to the organization's repositories apply to all members and excludes outside collaborators. Since organization members can have permissions from multiple sources, members and collaborators who have been granted a higher level of access than the base permissions will retain their higher permission privileges.

- **Repository creation**

Members will be able to create only selected repository types. Outside collaborators can never create repositories.

Public: Members will be able to create public repositories, visible to anyone.

Private: Members will be able to create private repositories, visible to organization members with permission.

- **Repository forking**

Allow forking of private repositories:

If enabled, forking is allowed on private and public repositories. If disabled, forking is only allowed on public repositories. This setting is also configurable per-repository.

- **Repository discussions**

Allow users with read access to create discussions:

If enabled, all users with read access can create and comment on discussions in GenAILakes099's repositories.

If disabled, discussion creation is limited to users with at least triage permission. Users with read access can still comment on discussions.

➢ Projects base permissions

Projects created by members will default to the selected role below.

- **Pages creation**

Members will be able to publish sites with only the selected access controls.

Public: Members will be able to create public sites, visible to anyone.
Private: Members will be able to create private sites, visible to anyone with permission.

- **Integration access requests**

Allow integration requests from outside collaborators:

Outside collaborators will be able to request access for GitHub or OAuth apps to access this organization and its resources.

➢ Admin repository permissions

- **Repository visibility change**

Allow members to change repository visibilities for this organization:

If enabled, members with admin permissions for the repository will be able to change its visibility. If disabled, only organization owners can change repository visibilities.

❖ **Repository deletion and transfer**

Allow members to delete or transfer repositories for this organization:

If enabled, members with admin permissions for the repository will be able to delete or transfer **public** and **private** repositories. If disabled, only organization owners can delete or transfer repositories.

❖ **Issue deletion**

Allow repository administrators to delete issues for this organization:

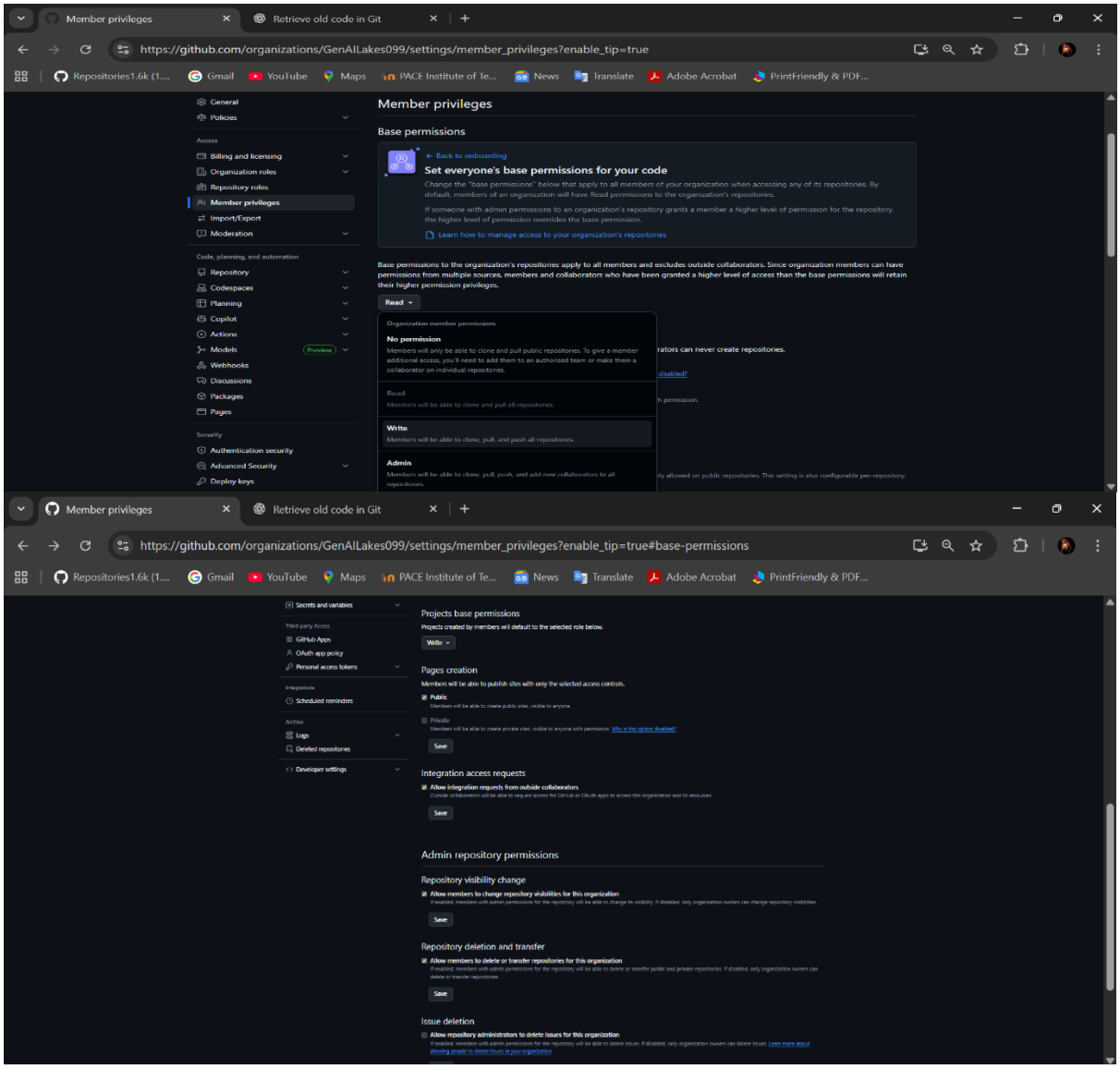
If enabled, members with admin permissions for the repository will be able to delete issues. If disabled, only organization owners can delete issues.

➢ **Member team permissions**

❖ **Team creation rules**

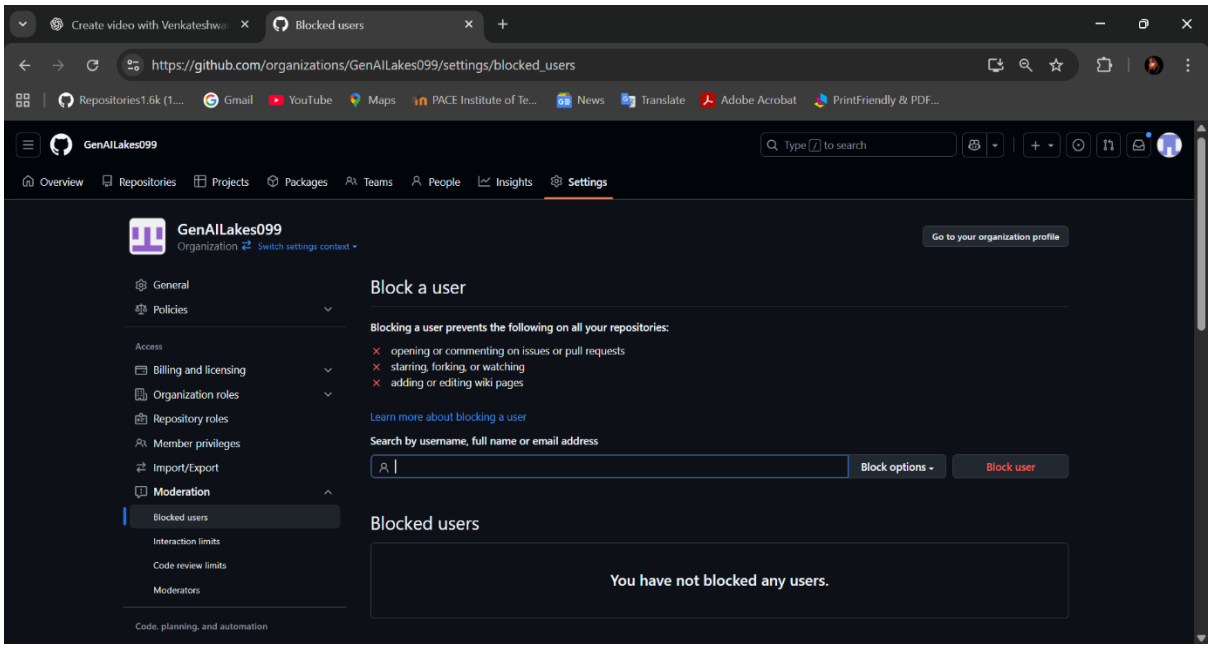
Allow members to create teams:

If enabled, any member of the organization will be able to create new teams. If disabled, only organization owners can create new teams.



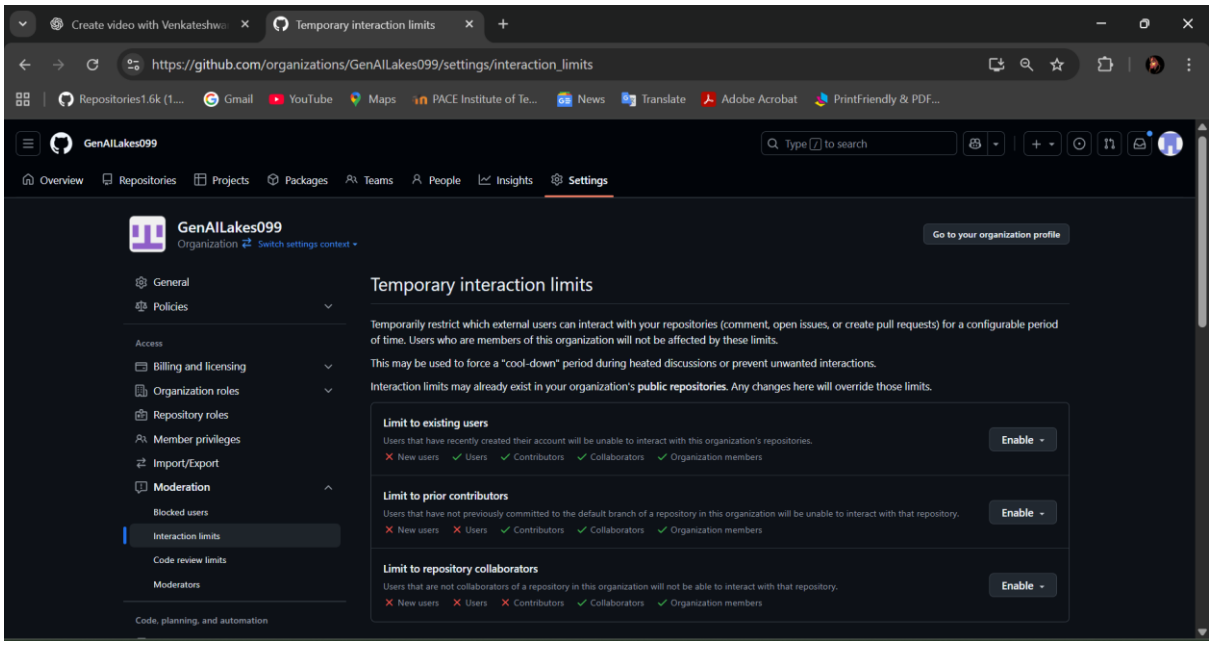
➢ Moderation

❖ Block Users

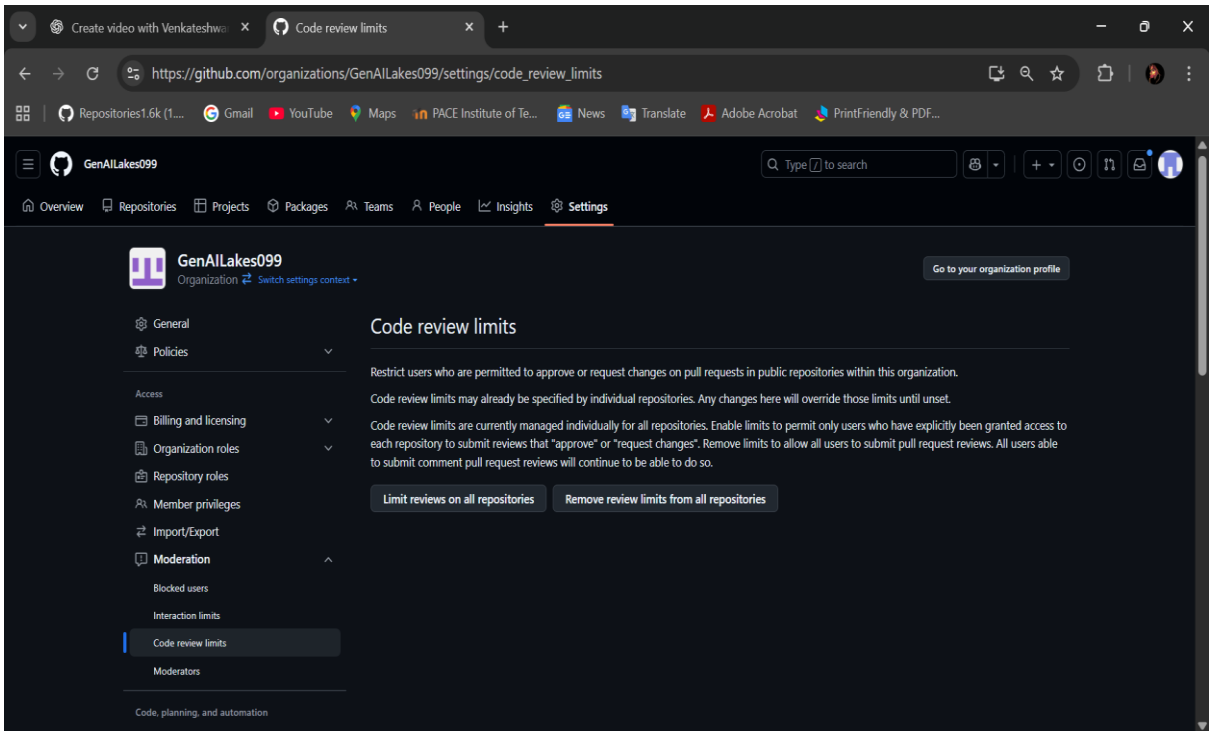


❖ Interaction Limits

- ❖ Temporarily restrict which external users can interact with your repositories (comment, open issues, or create pull requests) for a configurable period of time. Users who are members of this organization will not be affected by these limits.
- ❖ This may be used to force a "cool-down" period during heated discussions or prevent unwanted interactions.
- ❖ Interaction limits may already exist in your organization's **public repositories**. Any changes here will override those limits.

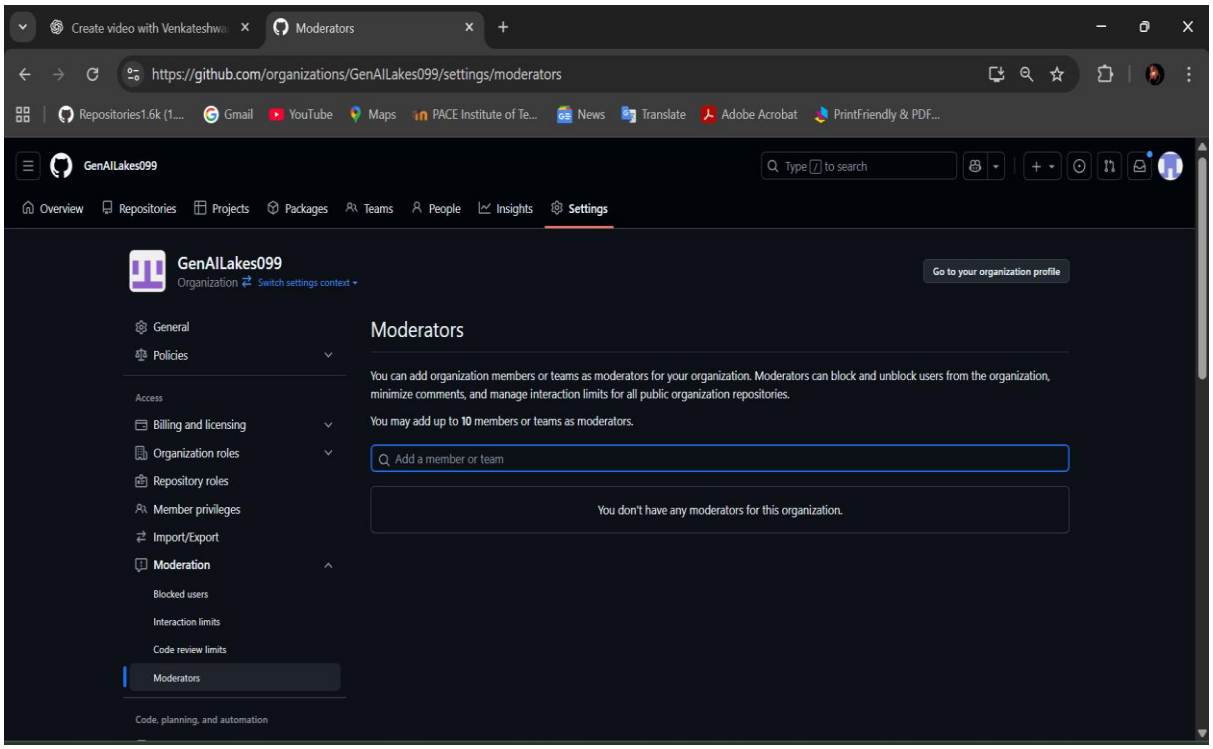


❖ Code Review Limits



➢ Moderators

- ❖ You can add organization members or teams as moderators for your organization. Moderators can block and unblock users from the organization, minimize comments, and manage interaction limits for all public organization repositories.
- ❖ You may add up to **10** members or teams as moderators.



Uses of Moderation in GitHub Organization:

1. Block Users

- What it does: Blocks specific users across the entire organization, not just one repository.
- Why useful:
  - Stops spammers or abusive contributors from opening issues, PRs, or commenting.
  - Provides consistency (blocked user is blocked everywhere in the org).
  - Saves admins from repeatedly blocking the same person repo by repo.

Example: If someone spams multiple repos in your org, blocking them once at the org level removes them everywhere.

---

2. Interaction Limits

- What it does: Restricts who can create issues, discussions, or PRs. Options include:
  - Everyone (default, open contribution)
  - Prior contributors only
  - Org members only
  - Members active for X days
- Why useful:
  - Prevents “drive-by” spam from new accounts.
  - Protects sensitive or enterprise projects.
  - Encourages higher-quality contributions.

Example: In an open-source project, you can allow only past contributors to open issues to reduce spam.

---

3. Code Review Limits

- What it does: Controls who can review pull requests within the organization.
  - Can restrict reviews to org members or specific teams.

    - Ensures only trusted users can approve code changes.
- Why useful:
    - Improves security (prevents malicious outsider reviews).
    - Keeps compliance with internal policies.
    - Ensures code quality by allowing reviews only from authorized maintainers.

Example: For a company repo, you can make sure only the DevOps or Security team can approve changes to Dockerfile.

---

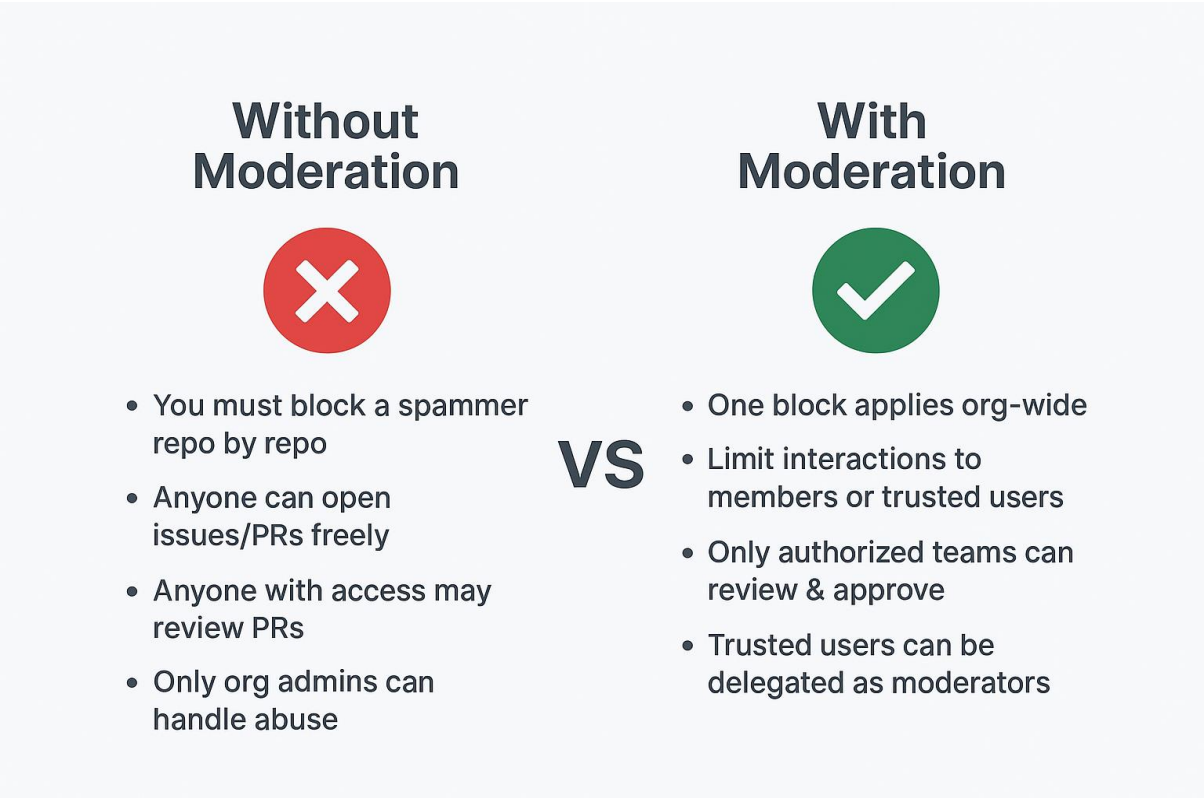
4. Moderators

- What it does: Assigns designated moderators (in addition to org admins).
    - Moderators can enforce rules, block/report users, and manage discussions/issues.
- Why useful:
    - Distributes responsibility (not just admins handle abuse).
    - Keeps community healthy in large open-source projects.
    - Provides quicker response to violations.

Example: In a big org, you can appoint trusted contributors as moderators to handle spam/abuse reports without giving them full admin rights.

Summary – Why Moderation is Useful for Your Org

- Block Users → Protect from repeat abusers' org-wide.
- Interaction Limits → Reduce spam & ensure only trusted contributors interact.
- Code Review Limits → Secure your review workflow, maintain code quality.
- Moderators → Share responsibility, keep your community safe and productive.



comparison table that clearly shows the difference between running a GitHub Organization without moderation vs with moderation

Feature	🚫 Without Moderation	✅ With Moderation
Block Users	You must block a spammer repo by repo. They can still attack other repos in the org.	One block applies org-wide, stopping abuse everywhere at once.
Interaction Limits	Anyone (including spam bots/new accounts) can open issues/PRs freely.	Limit interactions to members, prior contributors, or trusted users, reducing spam and noise.
Code Review Limits	Anyone with access may review PRs; risk of low-quality or malicious approvals.	Only authorized teams/org members can review & approve, ensuring secure and high-quality merges.
Moderators	Only org admins can handle abuse, leading to delays in large orgs.	Trusted users can be delegated as moderators, distributing responsibility and responding faster.

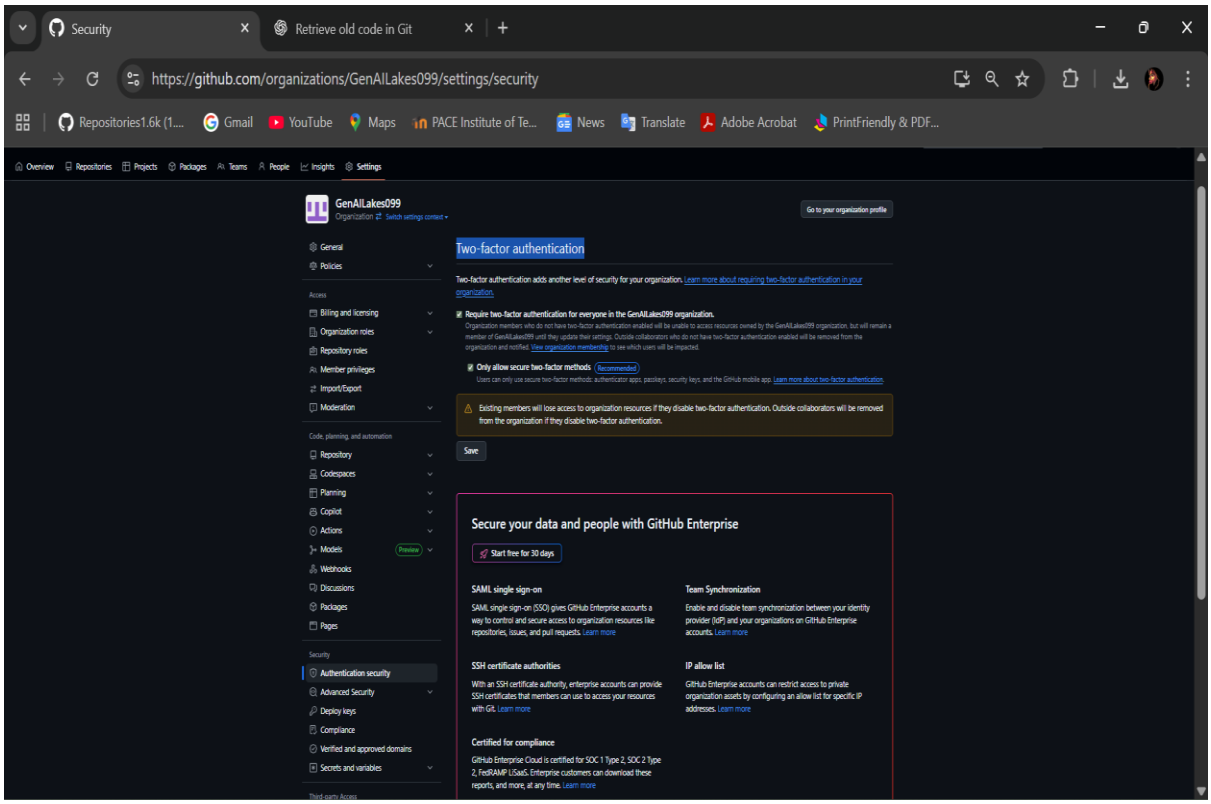
Feature	⃠ Without Moderation	✅ With Moderation
Community Safety	Higher risk of harassment, spam, and irrelevant issues/PRs.	Safer, more professional community with clear moderation policies.
Scalability	Admins overloaded with manual blocking/reporting.	Moderation scales well for large orgs or open-source communities.

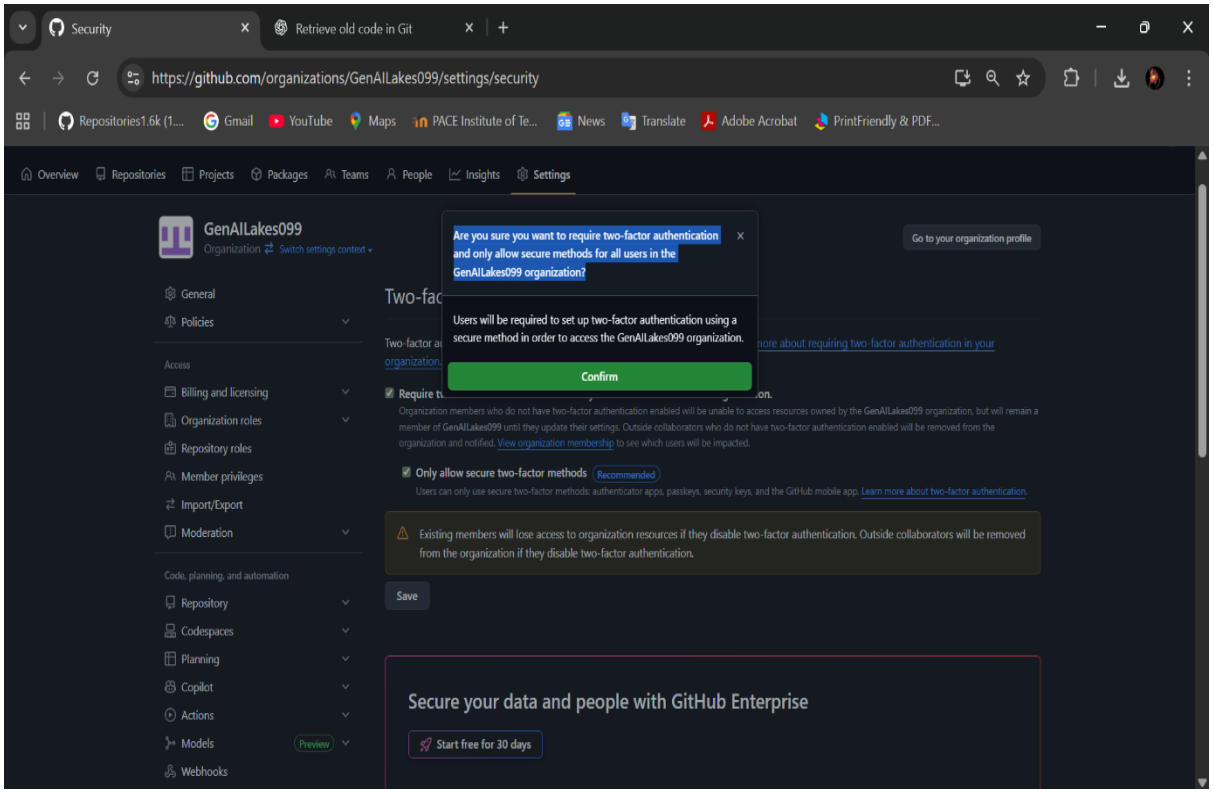
🎯 Takeaway

- Without Moderation → More open, but higher risk of spam, abuse, and security issues.
- With Moderation → Controlled, safer environment that scales well for both enterprise organizations and large open-source projects.

Two-factor authentication:

Go to the Organization ➔Security (Authentication Security) ➔ Two-factor authentication (Email, SMS, Passkey, etc)





POPUP: Are you sure you want to require two-factor authentication and only allow secure methods for all users in the GenAILakes099 organization.