A Project report

on

“EDUBUZZ”

with

Source Code Management

(CS181)

Submitted by

Team Member 1 . Yatin Dora 2110991591.

Team Member 2 . Aakash Goyal 2110991605.

Team Member 3 . Akash Chopra 2110991628.

Team Member 4 . Ayush Singla 2110991665.

Department of Computer Science & Engineering

Chitkara University Institute of Engineering and Technology, Punjab

Jan- June

(2021-22)

CHITKARA UNIVERSITY

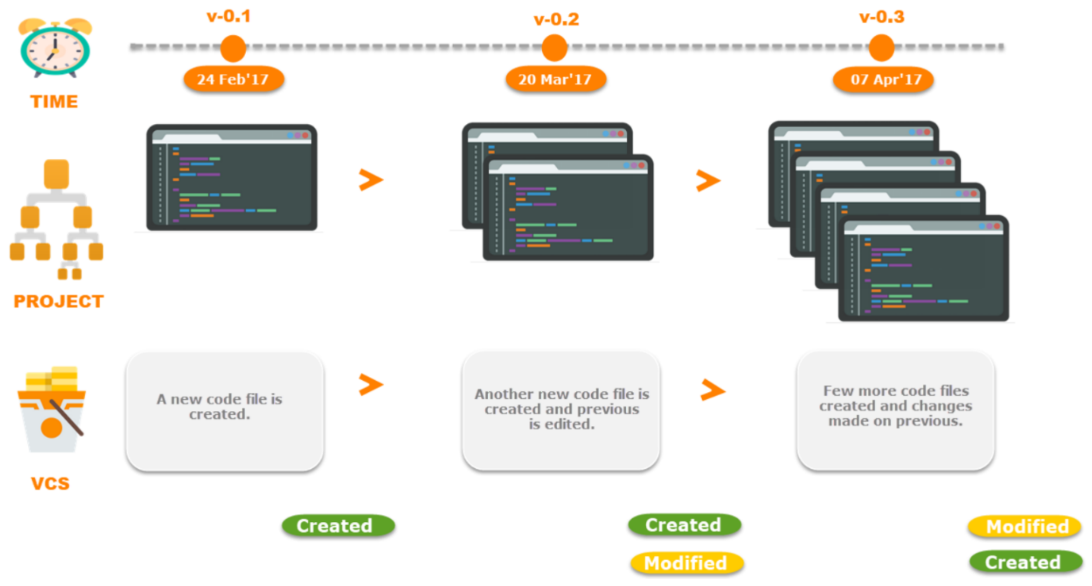
Institute/School Name	**Chitkara University Institute of Engineering and Technology**		
Department Name	**Department of Computer Science & Engineering**		
Programme Name	**Bachelor of Engineering (B.E.), Computer Science & Engineering**		
Course Name	**Source Code Management**	Session	**2021-22**
Course Code	**CS181**	Semester/Batch	**2nd/2021**
Vertical Name	**Alpha**	Group No	G-21
Course Coordinator	**Dr. Neeraj Singla**		
Faculty Name	**Dr. Shikha**		

CHITKARA UNIVERSITY

Table of Content

1. Version control with Git

What is “version control”, and why should you care? Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later. For the examples in this book, you will use software source code as the files being version controlled, though in reality you can do this with nearly any type of file on a computer. If you are a graphic or web designer and want to keep every version of an image or layout (which you would most certainly want to), a Version Control System (VCS) is a very wise thing to use. It allows you to revert selected files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more. Using a VCS also generally means that if you screw things up or lose files, you can easily recover. In addition, you get all this for very little overhead.


TIME
v-0.1
24 Feb'17
v-0.2
20 Mar'17
v-0.3
07 Apr'17
PROJECT
VCS
A new code file is created.
Another new code file is created and previous is edited.
Few more code files created and changes made on previous.
Created
Created
Modified
Modified
Created

Local Version Control Systems

Many people’s version-control method of choice is to copy files into another directory (perhaps a time-stamped directory, if they’re clever). This approach is very common because it is so simple, but it is also incredibly error prone. It is easy to forget which directory you’re in and accidentally write to the wrong file or copy over files you don’t mean to. To deal with this issue, programmers long ago developed local VCSs that had a simple database that kept all the changes to files under revision control.

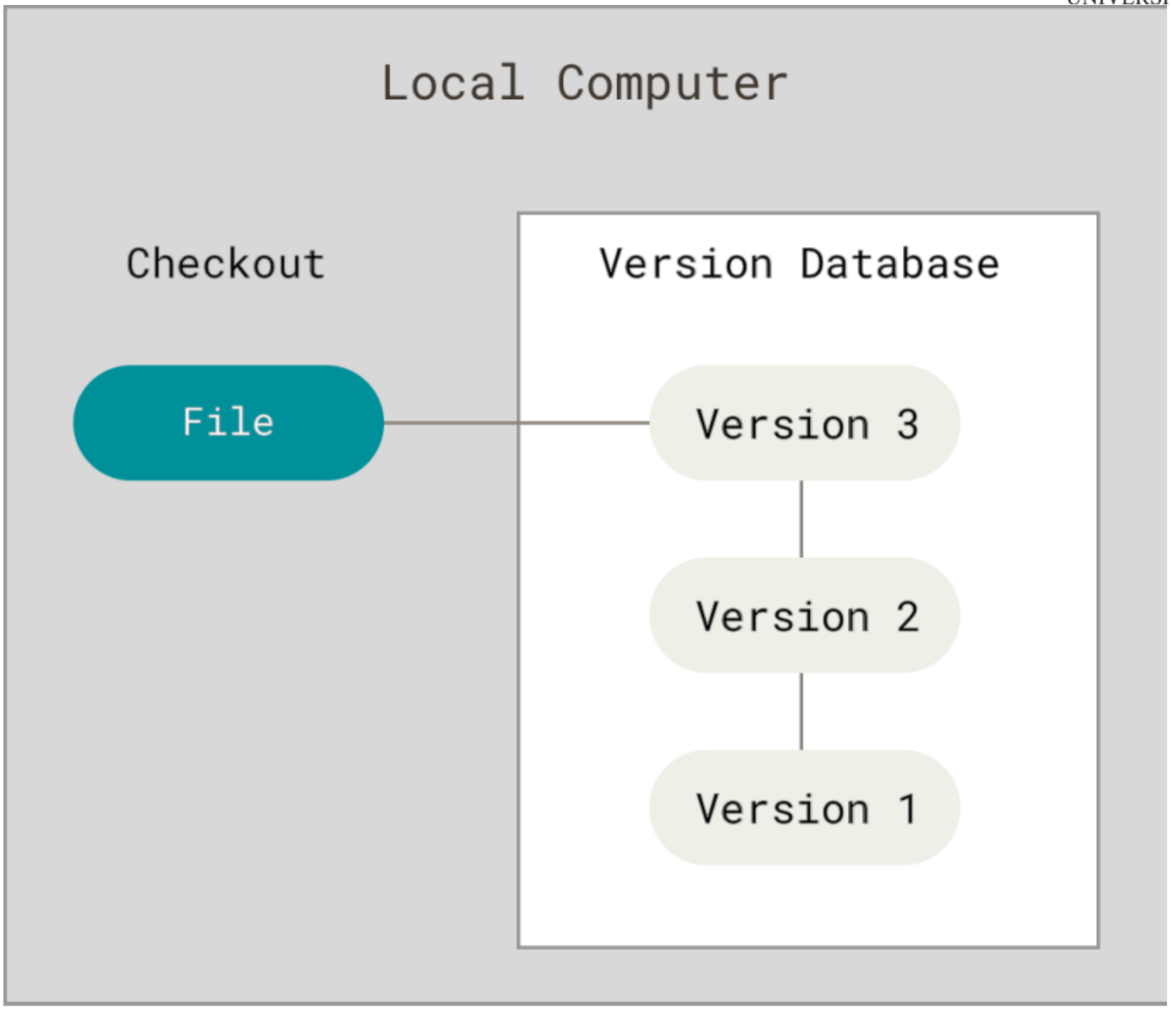


Figure 1. Local version control

One of the most popular VCS tools was a system called RCS, which is still distributed with many computers today. RCS works by keeping patch sets (that is, the differences between files) in a special format on disk; it can then re-create what any file looked like at any point in time by adding up all the patches.

Centralized Version Control Systems

The next major issue that people encounter is that they need to collaborate with developers on other systems. To deal with this problem, Centralized Version Control Systems (CVCSs) were developed. These systems (such as CVS, Subversion, and Perforce) have a single server that contains all the versioned files, and a number of clients that check out files from that central place. For many years, this has been the standard for version control.

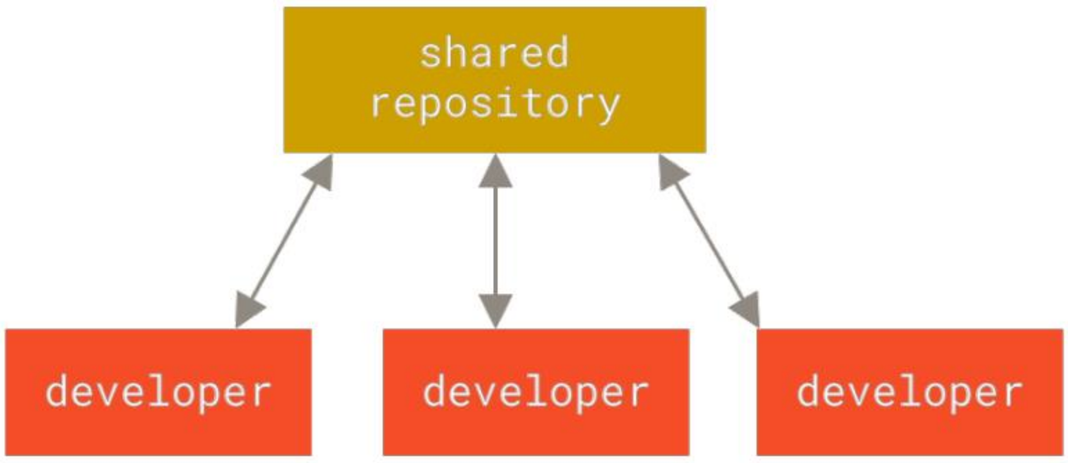


Figure 2. Centralized version control

This setup offers many advantages, especially over local VCSs. For example, everyone knows to a certain degree what everyone else on the project is doing. Administrators have fine-grained control over who can do what, and it's far easier to administer a CVCS than it is to deal with local databases on every client. However, this setup also has some serious downsides. The most obvious is the single point of failure that the centralized server represents. If that server goes down for an hour, then during that hour nobody can collaborate at all or save versioned changes to anything they're working on. If the hard disk the central database is on becomes corrupted, and proper backups haven't been kept, you lose absolutely everything — the entire history of the project except whatever single snapshots people happen to have on their local machines. Local VCSs suffer from this same problem — whenever you have the entire history of the project in a single place, you risk losing everything.

Distributed Version Control Systems

This is where Distributed Version Control Systems (DVCSs) step in. In a DVCS (such as Git, Mercurial, Bazaar or Darcs), clients don't just check out the latest snapshot of the files; rather, they fully mirror the repository, including its full history. Thus, if any server dies, and these systems were collaborating via that server, any of the client repositories can be copied back up to the server to restore it. Every clone is really a full backup of all the data.

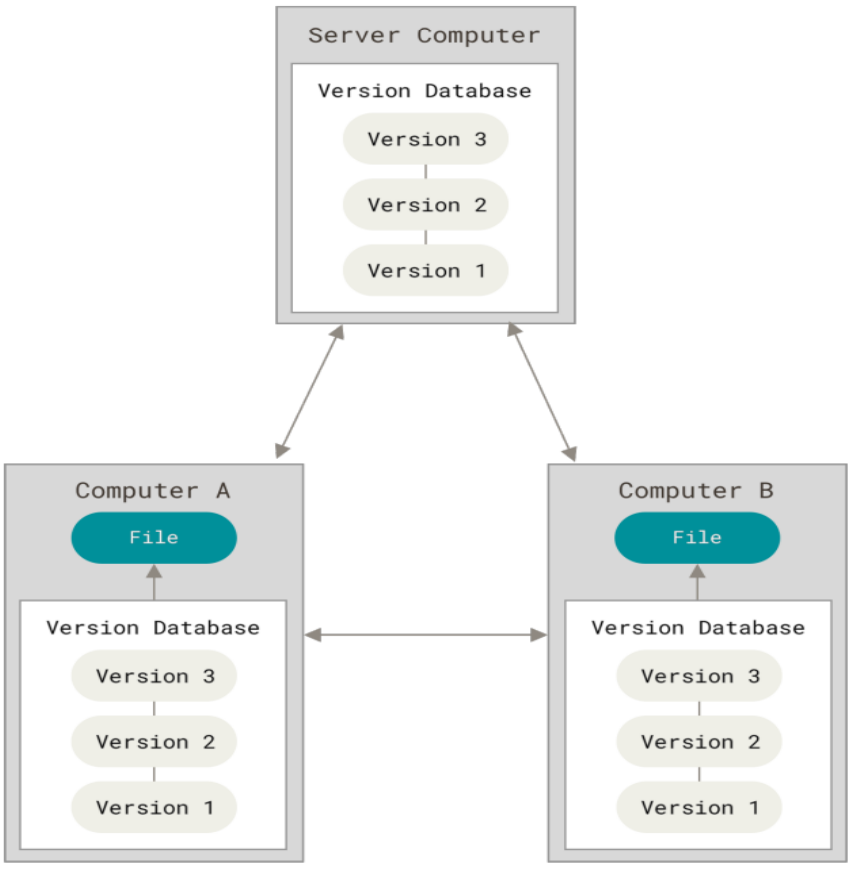


Figure 3. Distributed version control

Furthermore, many of these systems deal pretty well with having several remote repositories they can work with, so you can collaborate with different groups of people in different ways simultaneously within the same project. This allows you to set up several types of workflows that aren't possible in centralized systems, such as hierarchical models.

2. Problem Statement

Why Do We Need It?

Complex code development is impossible without a Version Control System (VCS). Version Control is used to track and control changes to source code. It is an essential tool to ensure the integrity of the codebase.

Why is Version Control Essential?

- Easy Modification of Codebase
- Reverting Error
- Improves visibility.
- Accelerates product delivery.

3.Objective

Version control system keeps track on changes made on a particular software and take a snapshot of every modification .Let's suppose if a team of developer add some new functionalities in a application and the updated version is not working properly so as version control system keeps track of our work so with the help of version control system we can omit the new changes and continue with the previous version .

Benefits of the version control system:

•Enhances the project development speed by providing efficient collaboration,

•Leverages the productivity, expedites product delivery, and skills of the employees through better communication and assistance,

•Reduce possibilities of errors and conflicts meanwhile project development through traceability to every small change,

•Employees or contributors of the project can contribute from anywhere irrespective of the different geographical locations through this VCS,

•For each different contributor to the project, a different working copy is maintained and not merged to the main file unless the working copy is validated. The most popular example is Git, Helix core, Microsoft TFS,

•Helps in recovery in case of any disaster or contingent situation,

•Informs us about Who, What, When, Why changes have been made.

4. Concepts and Commands

❖ Here are some commands which is used in GIT.

git config --global user.name “name”:

This command sets the author’s name to be used with your commits.

git config --global user.email “email”:

This command sets the author’s email address to be used with your commits.

git --version :

It is used to display the version of git bash.

pwd:

Gives the full pathname of the current working directory to the standard output

git init :

Used to initialize the repository.

touch “filename”:

It creates a new file.

Vi “filename”:

It creates and add content to a file.

git add –a:

Moves changes from the working directory to the staging area.

git commit -m “info of commit”:

This command records or snapshots the file permanently in the version history.

ls :

It gives the name of file present in the current folder.

git status:

This command lists all the files that have to be committed.

rm -rf .git:

This command is used delete the .git hidden folder.

git rm [file] :

This command deletes the file from your working directory and stages the deletion.

git log :

This command is used to list the version history for the current branch.

git branch:

This command lists all the local branches in the current repository.

git branch [branch name] :

This command creates a new branch.

git checkout [branch name] :

This command is used to switch from one branch to another.

git branch -m [branch name] :

This command is used to rename the branch.

git merge [branch name] :

This command is used to merge a command.

git branch -d [branch name] :

This command is used to soft delete a branch.

git branch -D [branch name] :

When you want to delete a branch without merging it. So, we use this command.

git log --oneline :

It gives the checkout information in one line.

git mv [file name] [file name] :

This command is used to rename a file.

git restore --staged [file name] :

This command is used to remove from staging area.

git log --oneline --graph :

It gives the checkout in the form of network graph.

git clone [URL] :

This command is used to obtain a repository from an existing URL.

git remote:

This command is used to connect your local repository to the remote server.

git push :

This command is used to push our repository.

git pull :

This command is used to pull a repository.

git remote remove [remote name] :

This command is to remove the remote.

git reset [checksome] :

This command is used to restore the commit upto the checksome.

git revert [checksome] :

This command is used to restore the commit upto the checksome and added a new checksome of revert.

.gitignore.txt :

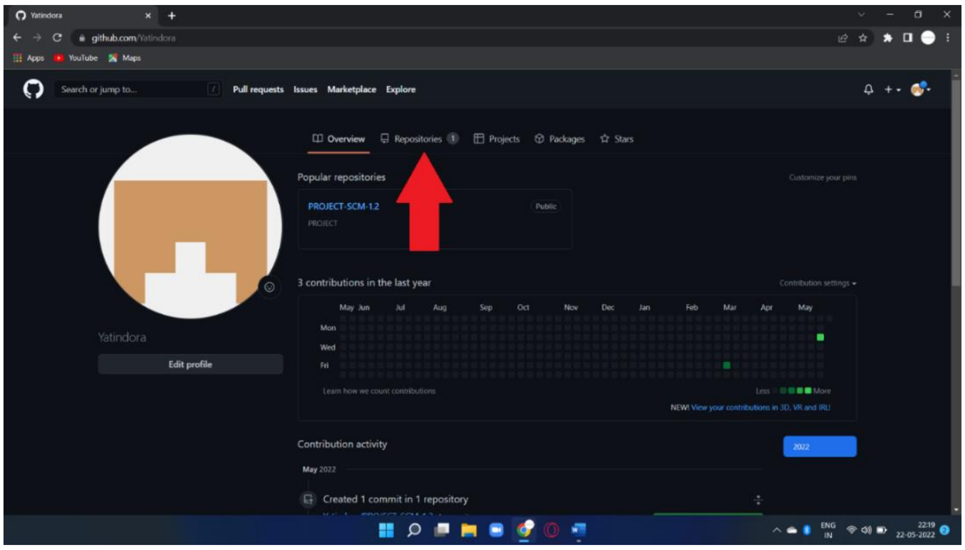
It is a text file used to ignore files which is not usable for us.

It ignore all the files which is inside .gitignote file.

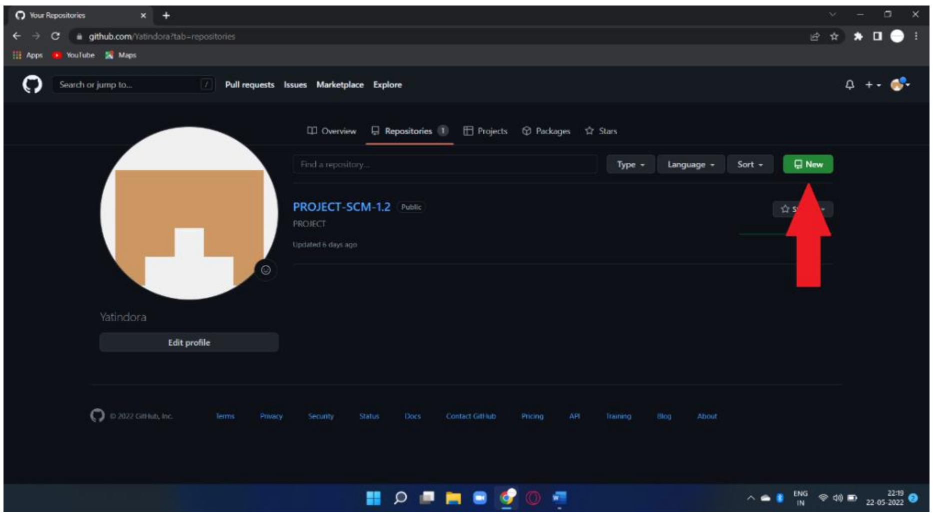
5.Workflow and Discussion.

Aim: Create a distributed Repository and add members in project team

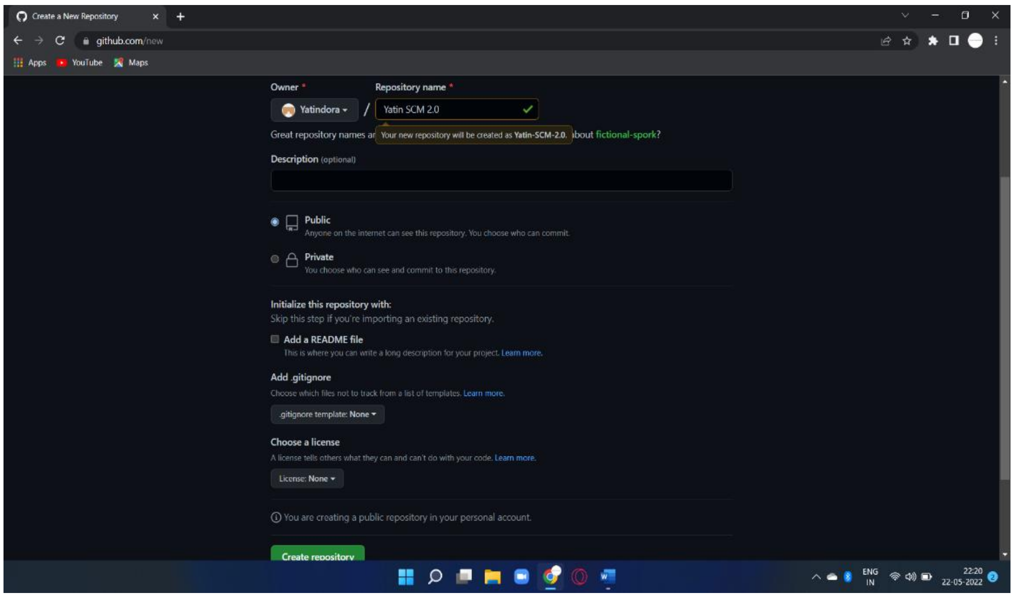
- Login to your GitHub account and you will land on the homepage as shown below. Click on Repositories option in the menu bar.



- Click on the 'New' button in the top right corner.



- Enter the Repository name and add the description of therepository.
- Select if you want the repository to be public or private.



Owner * Repository name *

Yatindora / Yatin SCM 2.0

Great repository names ar Your new repository will be created as Yatin-SCM-2.0. about fictional-spork?

Description (optional)

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

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

Initialize this repository with:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.

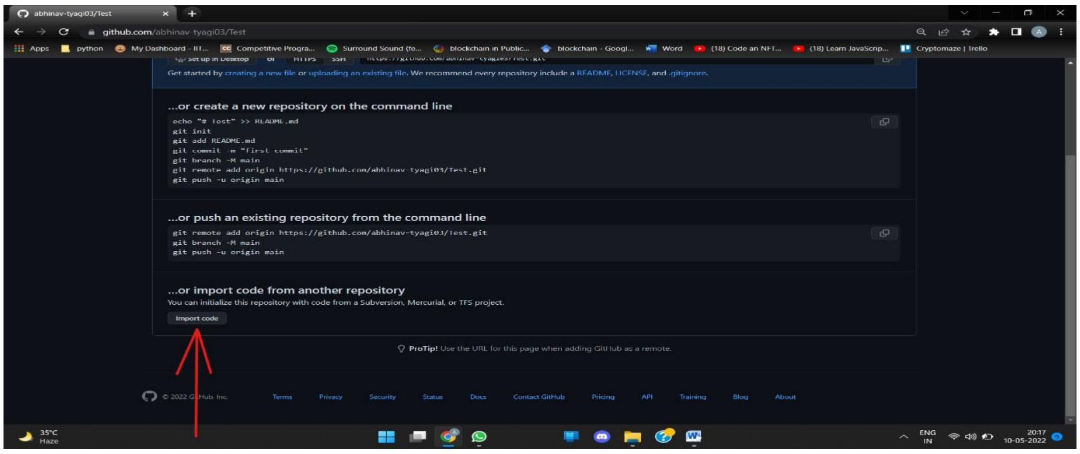
.gitignore template: None

Choose a license
A license tells others what they can and can't do with your code. Learn more.

License: None

You are creating a public repository in your personal account.

- If you want to import code from an existing repository selectthe



Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# Test" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/abhinav-tyagi03/Test.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/abhinav-tyagi03/Test.git
git branch -M main
git push -u origin main
```

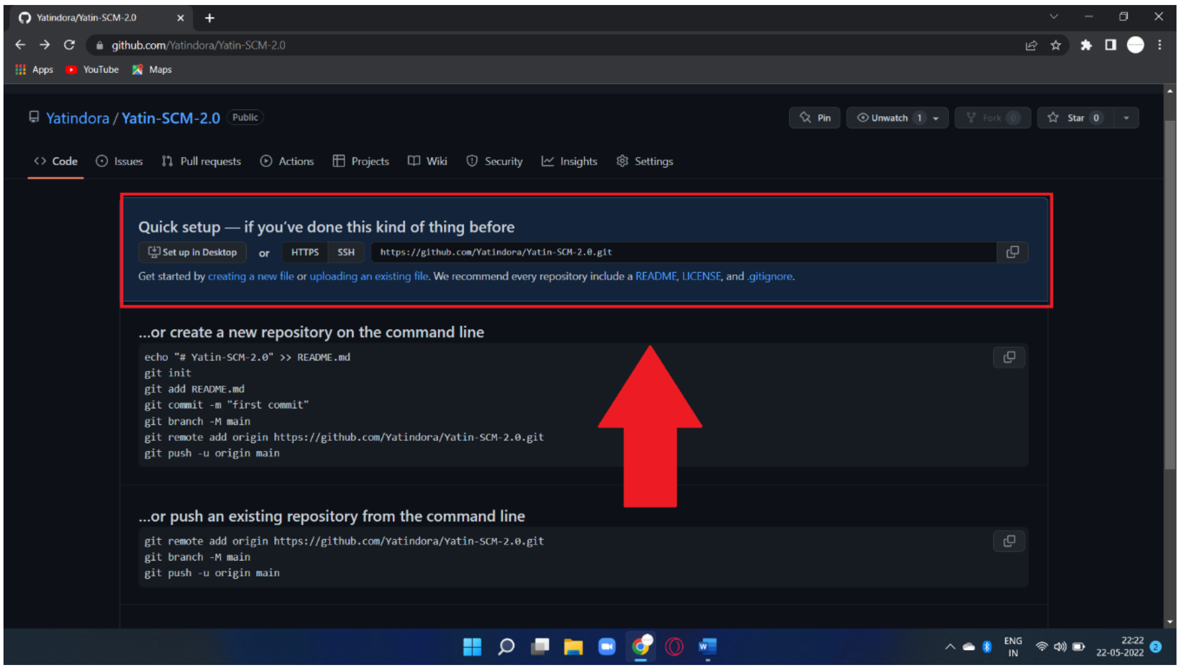
...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

ProTip! Use the URL for this page when adding GitHub as a remote.

import code option.

- To create a new file or upload an existing file into yourrepository select the option in the following box.



Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/Yatindora/Yatin-SCM-2.0.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

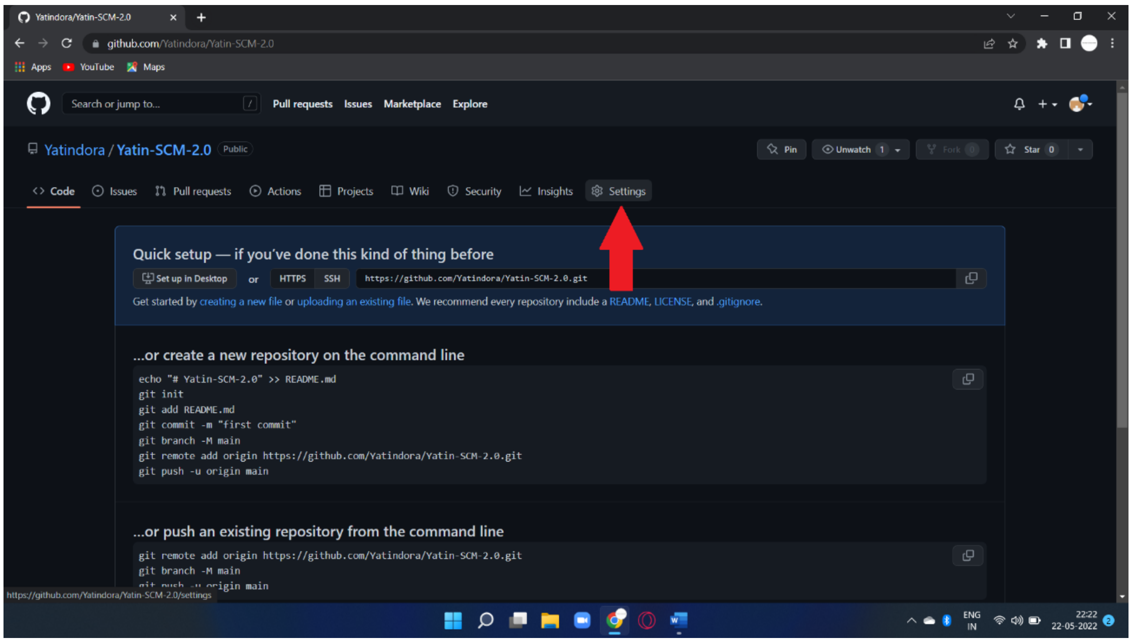
...or create a new repository on the command line

```
echo "# Yatin-SCM-2.0" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Yatindora/Yatin-SCM-2.0.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/Yatindora/Yatin-SCM-2.0.git
git branch -M main
git push -u origin main
```

- Now, you have created your repository successfully.
- To add members to your repository, open your repository andselect settings option in the navigation bar.



Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/Yatindora/Yatin-SCM-2.0.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

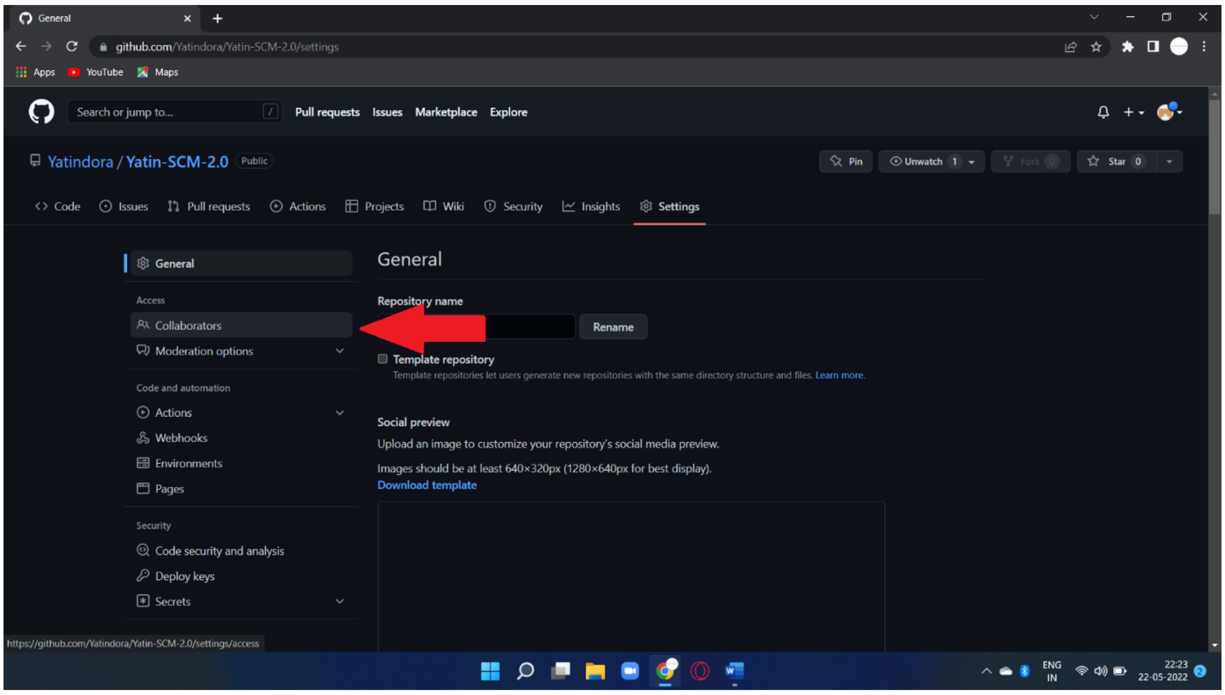
...or create a new repository on the command line

```
echo "# Yatin-SCM-2.0" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Yatindora/Yatin-SCM-2.0.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/Yatindora/Yatin-SCM-2.0.git
git branch -M main
git push -u origin main
```

- Click on Collaborators option under the access tab.


General
github.com/Yatindora/Yatin-SCM-2.0/settings
Pull requests Issues Marketplace Explore
Yatindora / Yatin-SCM-2.0 Public
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
General
Access
Collaborators
Moderation options
Code and automation
Actions
Webhooks
Environments
Pages
Security
Code security and analysis
Deploy keys
Secrets
Repository name
Rename
Template repository
Social preview

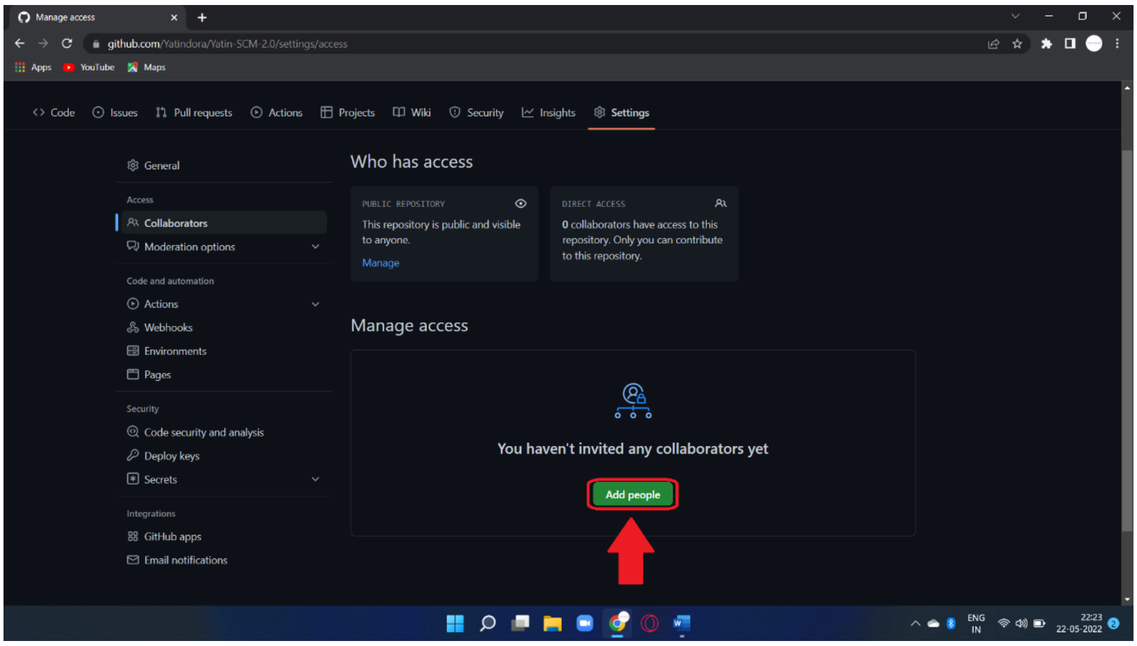
Template repositories let users generate new repositories with the same directory structure and files. Learn more.

Upload an image to customize your repository's social media preview.

Images should be at least 640×320px (1280×640px for best display).

Download template

- After clicking on collaborators GitHub asks you to enter yourpassword to confirm the access to the repository.
- After entering the password you can manage access andadd/remove team members to your project.
- To add members click on the add people option and searchthe id of your respective team member.

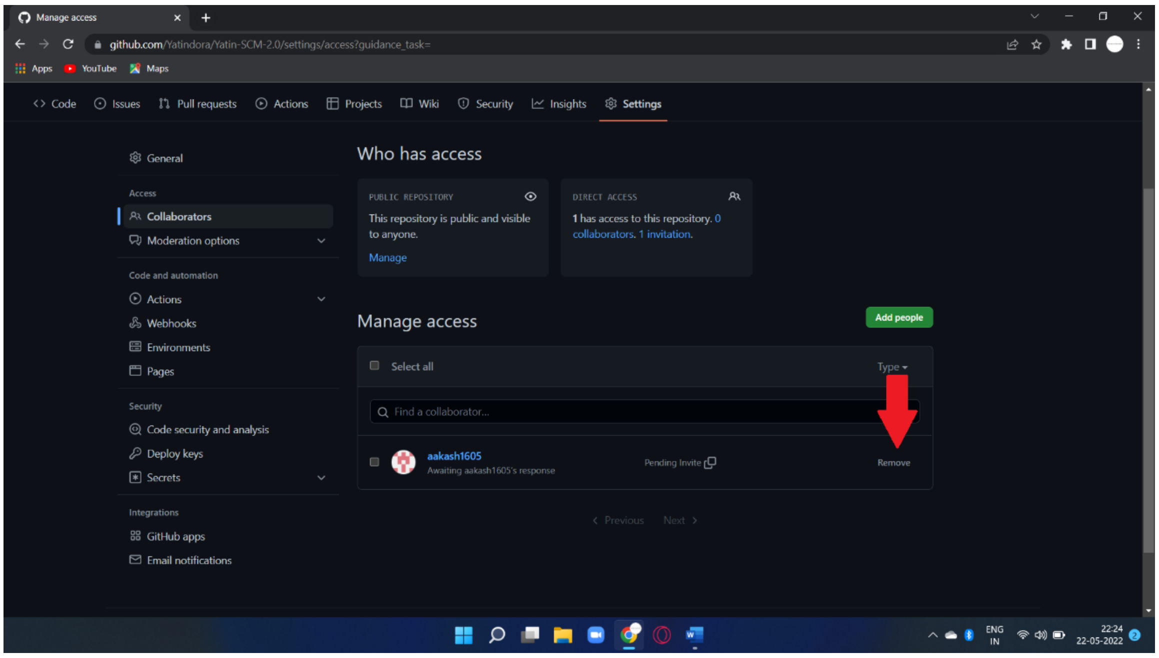


This repository is public and visible to anyone.

0 collaborators have access to this repository. Only you can contribute to this repository.

You haven't invited any collaborators yet

- To remove any member click on remove option available inthe last column of member's respective row.



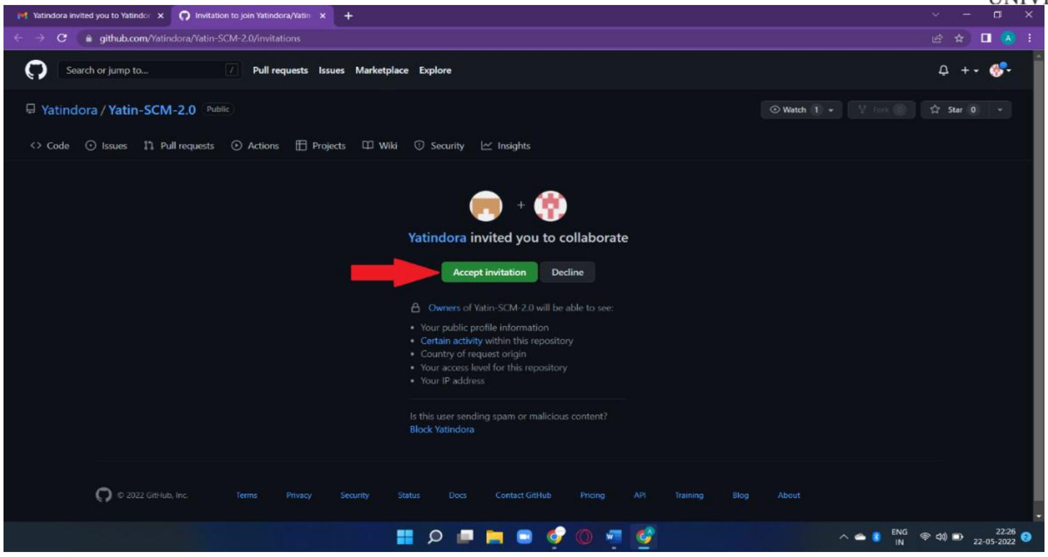
This repository is public and visible to anyone.

1 has access to this repository. 0 collaborators. 1 invitation.

Awaiting aakash1605's response

- To accept the invitation from your team member, open youremail registered with GitHub.
- You will receive an invitation mail from the repository owner.Open the email and click on accept invitation.

- You will be redirected to GitHub where you can either selectto accept or decline the invitation.



Yatindora / Yatin-SCM-2.0 Public

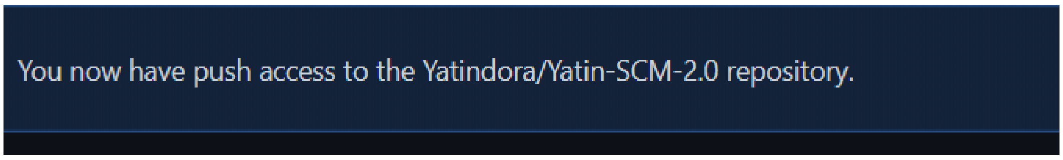
Yatindora invited you to collaborate

Owners of Yatin-SCM-2.0 will be able to see:

- Your public profile information
- Certain activity within this repository
- Country of request origin
- Your access level for this repository
- Your IP address

Is this user sending spam or malicious content?
Block Yatindora

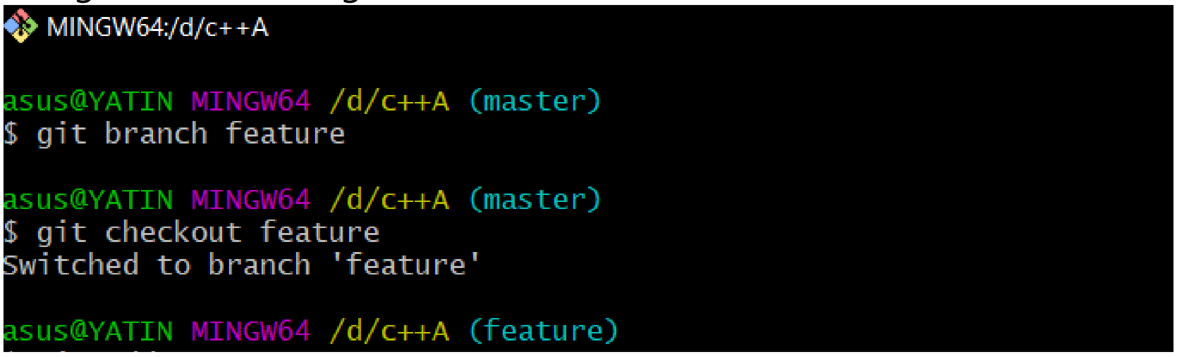
- You will be shown the option that you are now allowed topush


You now have push access to the Yatindora/Yatin-SCM-2.0 repository.

- Now all members are ready to contribute

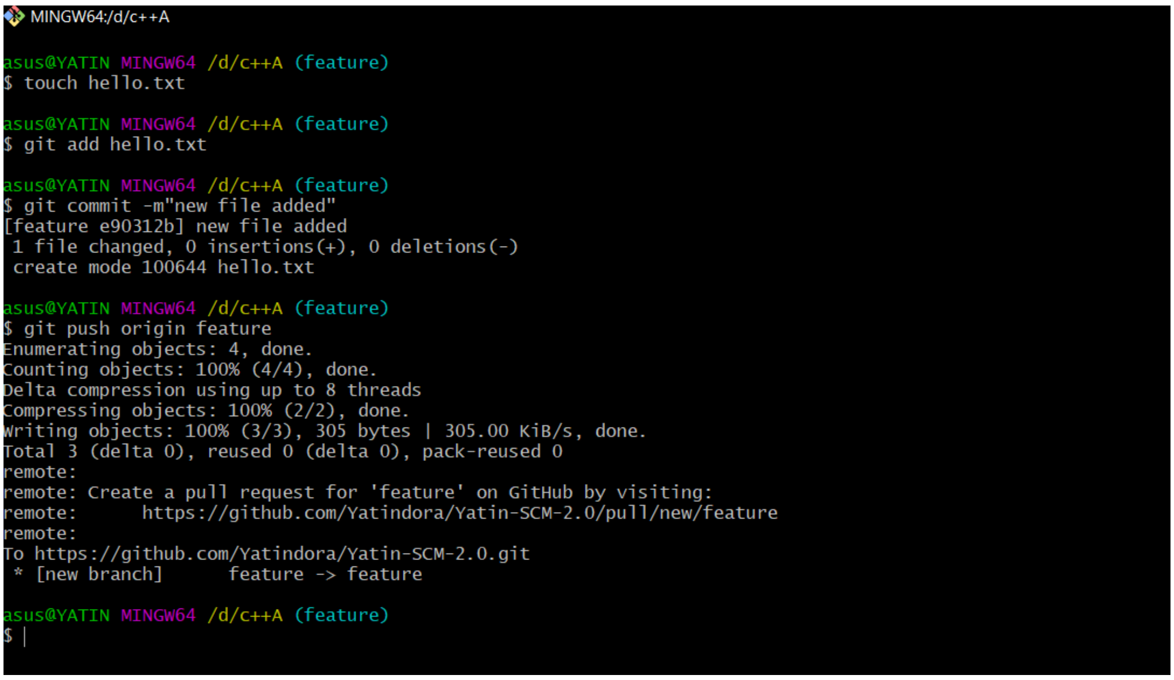
Aim: **Open And Close a Pull Request**

- To open a pull request we first have to make a new branch, by using gitbranch branchname option.
- After making new branch we add a file to the branch or make changes inthe existing file.



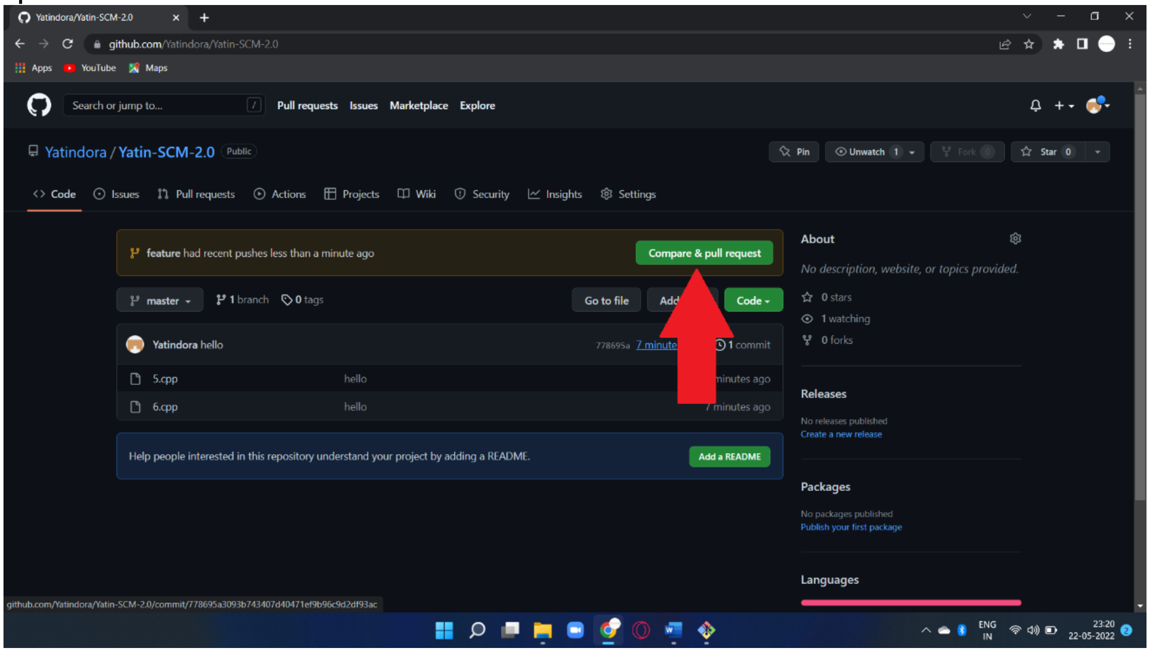
```
MINGW64:/d/c++A

asus@YATIN MINGW64 /d/c++A (master)
$ git branch feature

asus@YATIN MINGW64 /d/c++A (master)
$ git checkout feature
Switched to branch 'feature'

asus@YATIN MINGW64 /d/c++A (feature)
```

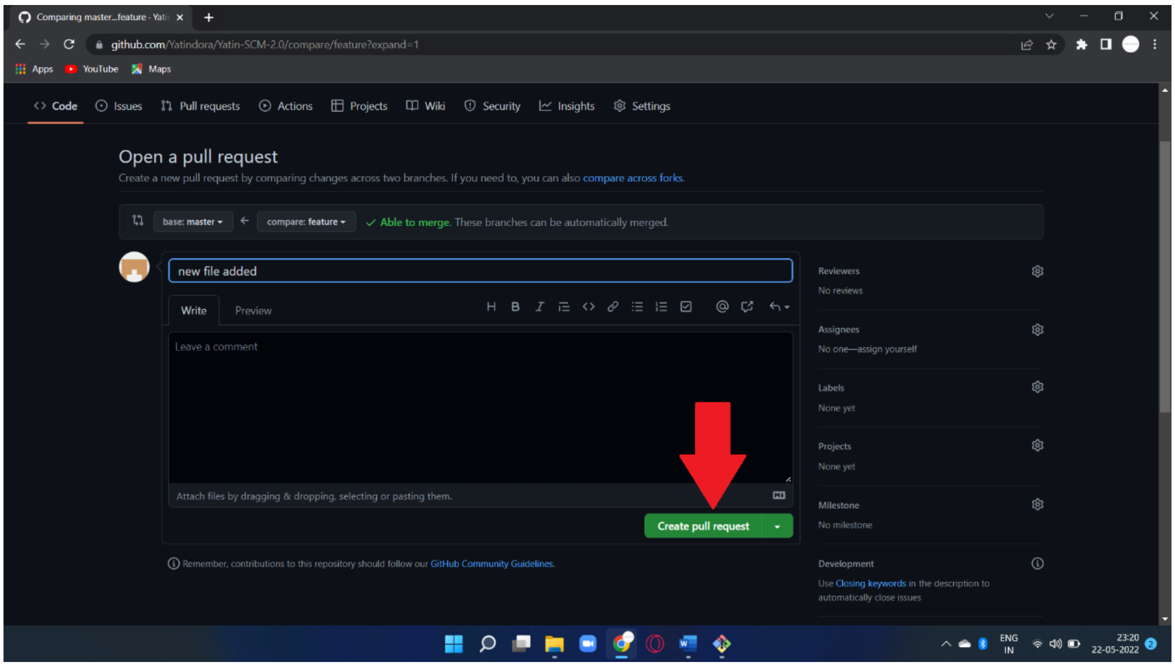
- Add and commit the changes to the local repository.
- Use git push origin branchname option to push the new branch to the mainrepository.



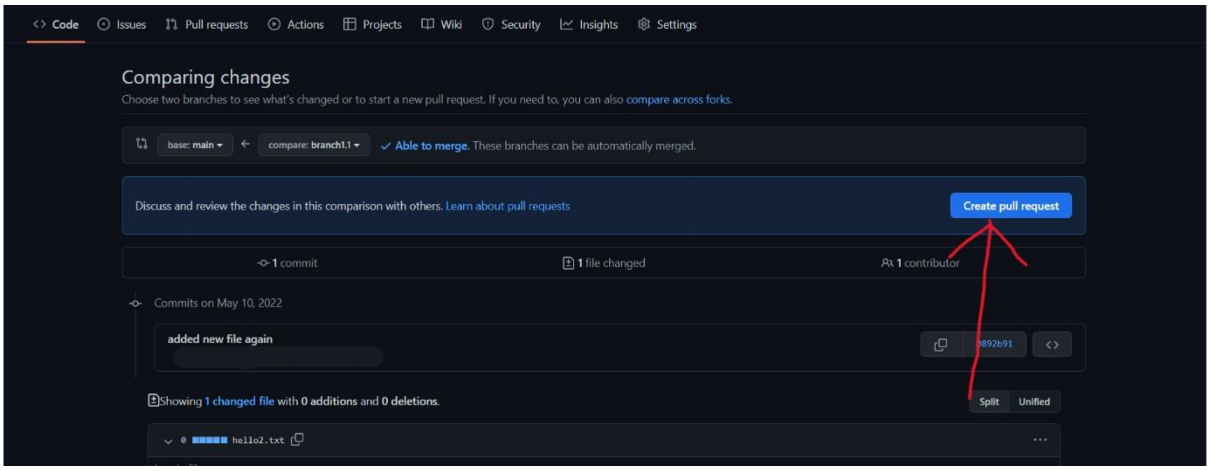
```
MINGW64:/d/c++A

asus@YATIN MINGW64 /d/c++A (feature)
$ touch hello.txt

asus@YATIN MINGW64 /d/c++A (feature)
$ git add hello.txt

asus@YATIN MINGW64 /d/c++A (feature)
$ git commit -m"new file added"
[feature e90312b] new file added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 hello.txt

asus@YATIN MINGW64 /d/c++A (feature)
$ git push origin feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 305 bytes | 305.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/Yatindora/Yatin-SCM-2.0/pull/new/feature
remote:
To https://github.com/Yatindora/Yatin-SCM-2.0.git
 * [new branch]      feature -> feature

asus@YATIN MINGW64 /d/c++A (feature)
$ |
```

- After pushing new branch GitHub will either automatically ask you tocreate a pull request or you can create your own pull request.


Yatindora / Yatin-SCM-2.0
feature had recent pushes less than a minute ago
Compare & pull request
master
Go to file
Code
Yatindora hello
5.cpp
6.cpp
Help people interested in this repository understand your project by adding a README.
Add a README


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: master
compare: feature
Able to merge. These branches can be automatically merged.
new file added
Leave a comment
Create pull request

- To create your own pull request click on pull request option.



Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

base: main ← compare: branch1.1 ✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. Learn about pull requests

Create pull request

1 commit · 1 file changed · 1 contributor

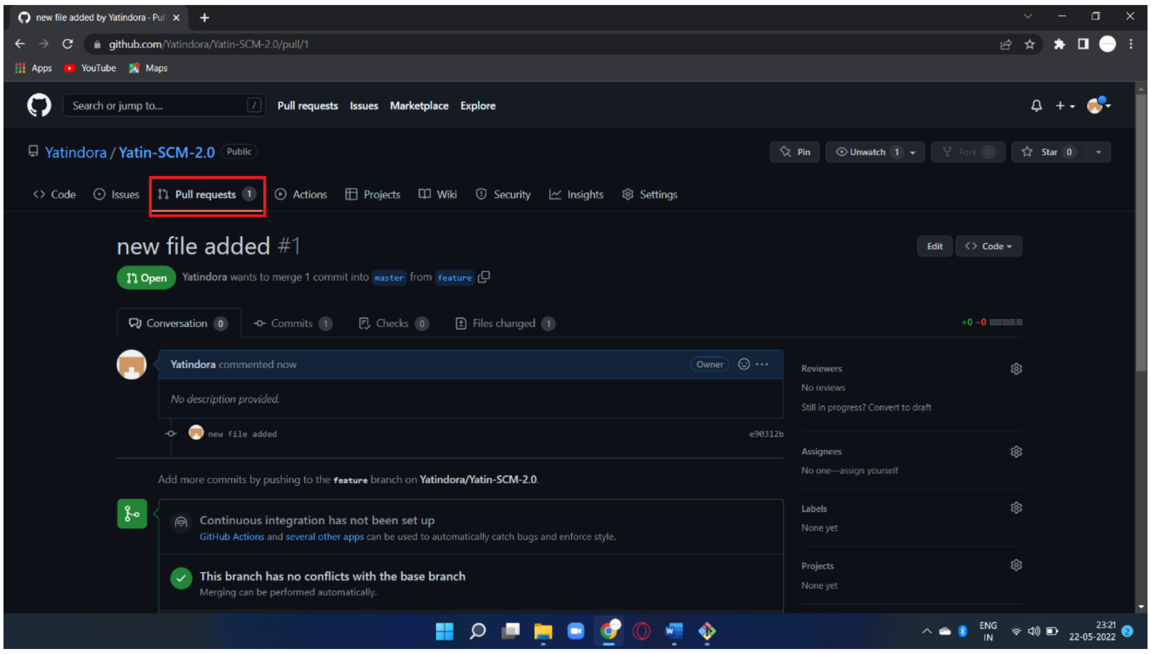
Commits on May 10, 2022

added new file again

Showing 1 changed file with 0 additions and 0 deletions.

hello2.txt

- GitHub will detect any conflicts and ask you to enter a description of yourpull request.



Yatindora / Yatin-SCM-2.0 Public

new file added #1

Open Yatindora wants to merge 1 commit into master from feature

Conversation 0 · Commits 1 · Checks 0 · Files changed 1

Yatindora commented now

No description provided.

new file added e90312b

Add more commits by pushing to the feature branch on Yatindora/Yatin-SCM-2.0.

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.

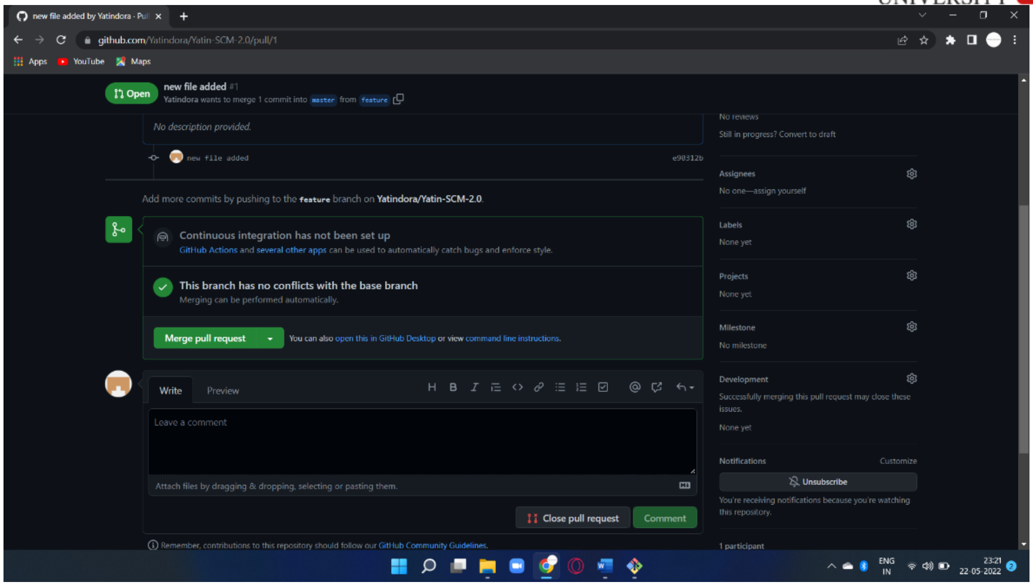
Reviewers: No reviews. Still in progress? Convert to draft

Assignees: No one—assign yourself

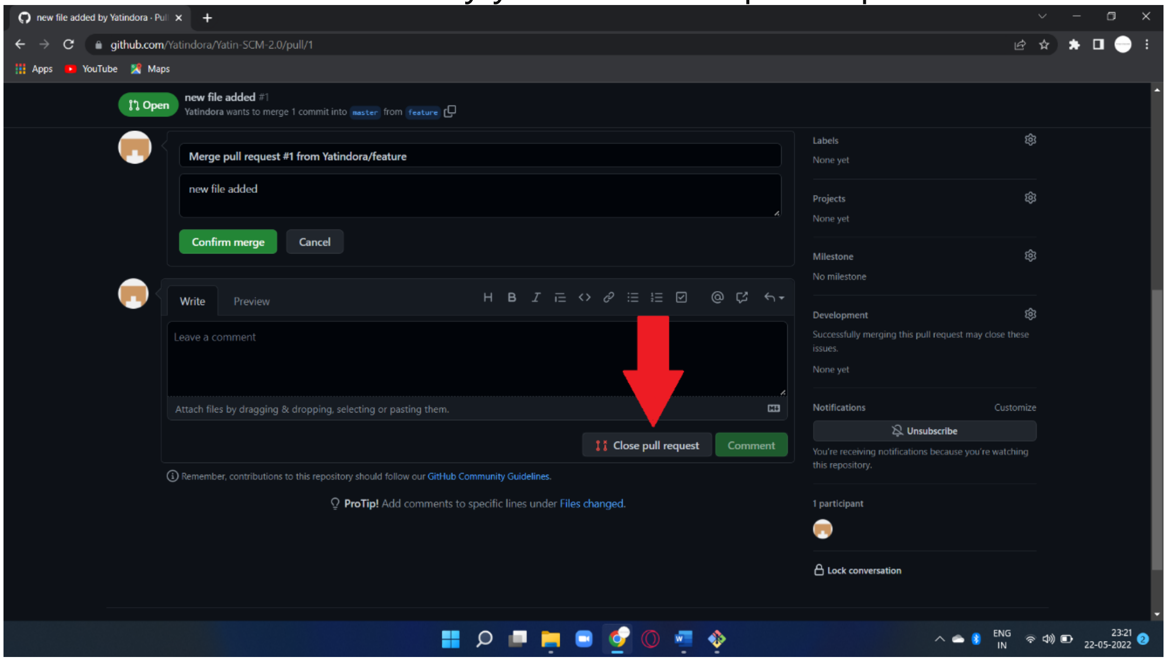
Labels: None yet

Projects: None yet

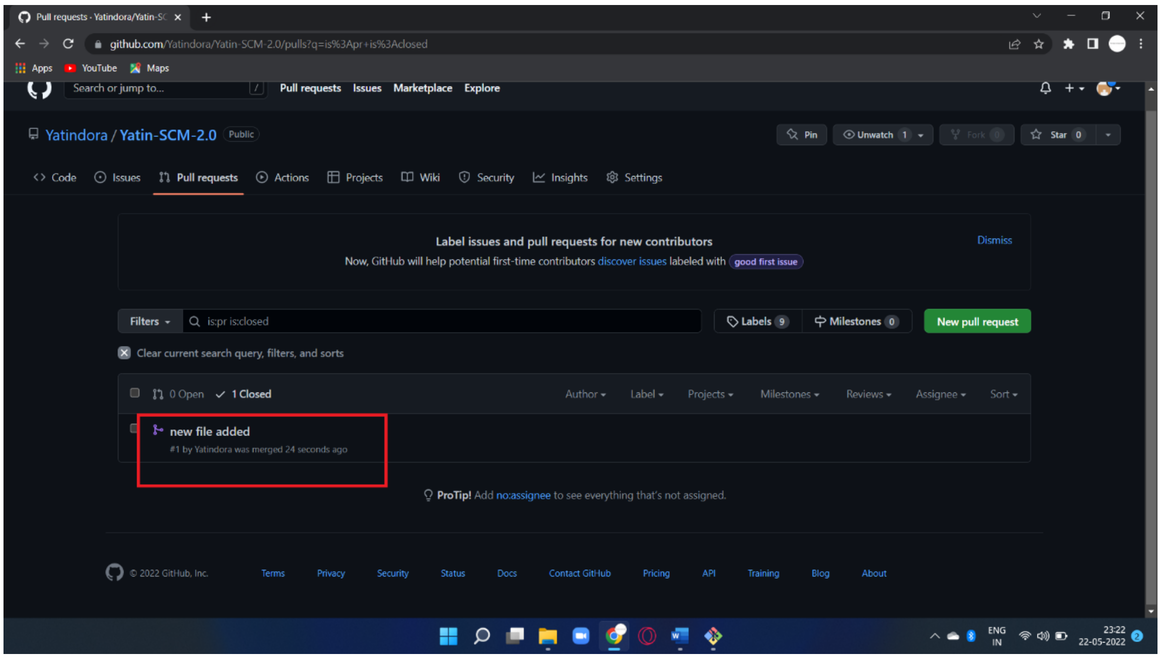
- After opening a pull request all the team members will be sent the requestif they want to merge or close the request.


new file added #1
Open
Yatindora wants to merge 1 commit into master from feature
No description provided.
new file added
Add more commits by pushing to the feature branch on Yatindora/Yatin-SCM-2.0.
Continuous integration has not been set up
This branch has no conflicts with the base branch
Merging can be performed automatically.
Merge pull request
Write
Preview
Leave a comment
Close pull request
Comment
Assignees
Labels
Projects
Milestone
Development
Notifications
Unsubscribe

- If the team member chooses not to merge your pull request they will closeyou're the pull request.
- To close the pull request simply click on close pull request and addcomment/ reason why you closed the pull request.



- You can see all the pull request generated and how they were dealt withby clicking on pull request option.



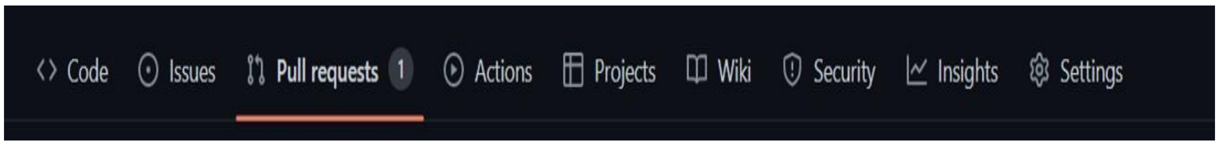
Pull requests · Yatindora/Yatin-SC
github.com/Yatindora/Yatin-SCM-2.0/pulls?q=is%3Apr+is%3Aclosed
Yatindora / Yatin-SCM-2.0 Public
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
Label issues and pull requests for new contributors
Now, GitHub will help potential first-time contributors discover issues labeled with good first issue
Dismiss
Filters is:pr is:closed Labels 9 Milestones 0 New pull request
Clear current search query, filters, and sorts
0 Open 1 Closed
Author Label Projects Milestones Reviews Assignee Sort
new file added
#1 by Yatindora was merged 24 seconds ago
ProTip! Add no:assignee to see everything that's not assigned.

Aim: **Create a pull request on a team member's repo and**

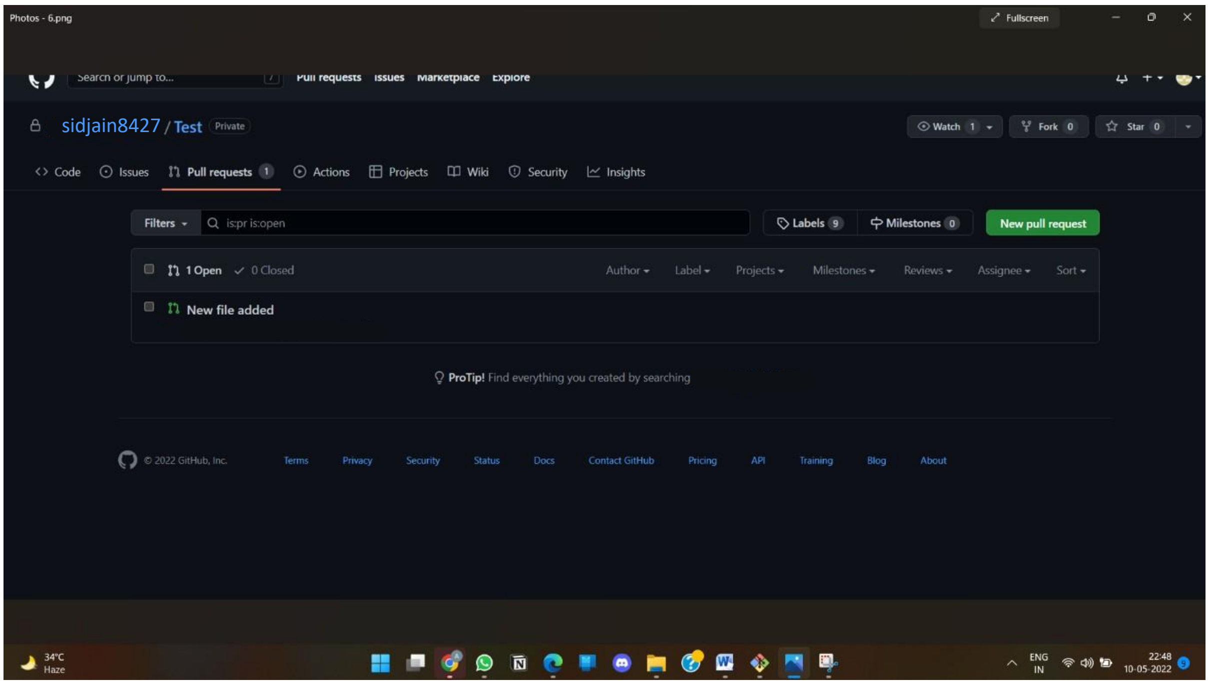
close pull requests generated by team members on own Repoas a maintainer

To create a pull request on a team member's repository and close requests byany other team members as a maintainer follow the procedure given below:-

- Do the required changes in the repository, add and commit thesechanges in the local repository in a new branch.
- Push the modified branch using git push origin branchname.
- Open a pull request by following the procedure from the aboveexperiment.
- The pull request will be created and will be visible to all the teammembers.
- Ask your team member to login to his/her Github account.



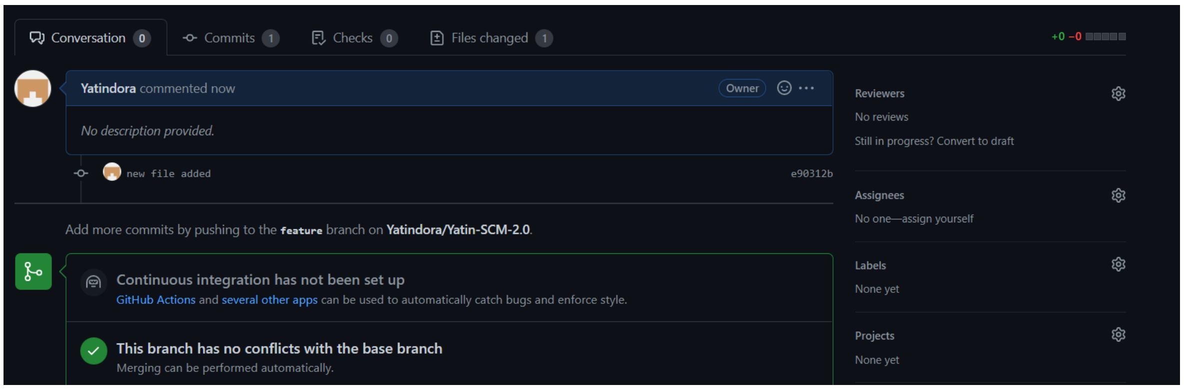
- They will notice a new notification in the pull request menu.
- Click on it. The pull request generated by you will be visible to them.



Photos - 6.png
sidjain8427 / Test Private
Code Issues Pull requests 1 Actions Projects Wiki Security Insights
Filters is:pr is:open
Labels 9 Milestones 0 New pull request
1 Open 0 Closed
New file added
ProTip! Find everything you created by searching

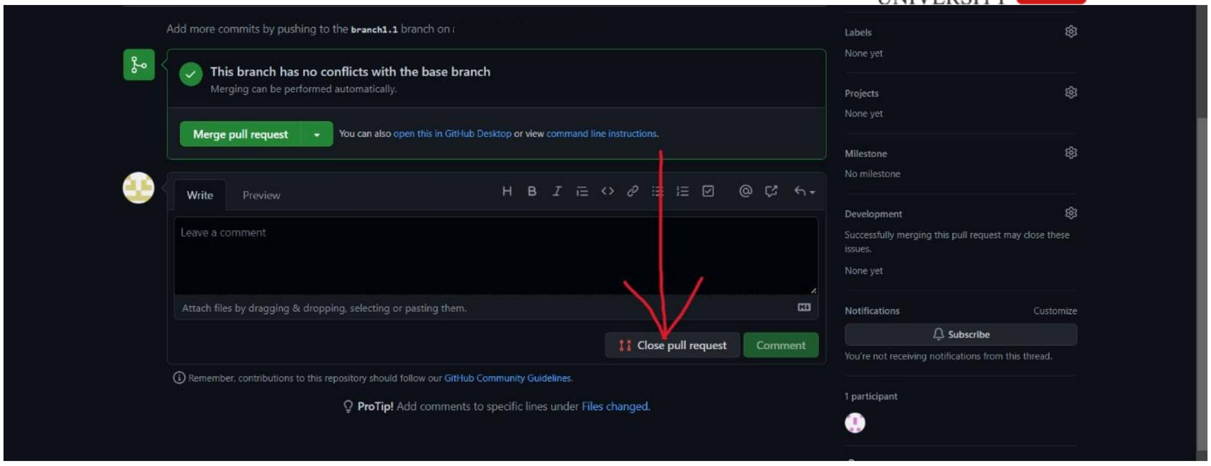
- Click on the pull request. Two option will be available, either to closethe pull request or Merge the request with the main branch.

- By selecting the merge branch option the main branch will get updatedfor all the team members.



Conversation 0 Commits 1 Checks 0 Files changed 1
Yatindora commented now Owner
No description provided.
new file added e90312b
Add more commits by pushing to the feature branch on Yatindora/Yatin-SCM-2.0.
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.
Reviewers No reviews Still in progress? Convert to draft
Assignees No one—assign yourself
Labels None yet
Projects None yet

- By selecting close the pull request the pull request is not accepted andnot merged with main branch.


Add more commits by pushing to the **branch1.1** branch on

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

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

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Close pull request Comment

Remember, contributions to this repository should follow our GitHub Community Guidelines.

ProTip! Add comments to specific lines under Files changed.

Labels
None yet

Projects
None yet

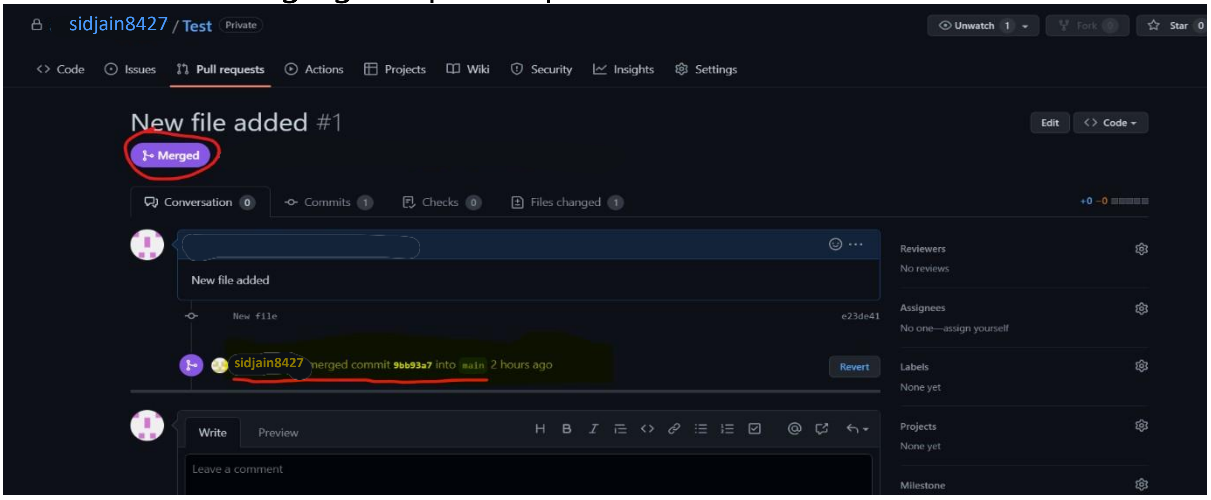
Milestone
No milestone

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

Notifications Customize
Subscribe
You're not receiving notifications from this thread.

1 participant

- The process is similar to closing and merging the pull request by you. Itsimply includes an external party to execute.
- The result of merging the pull request is shown below.


sidjain8427 / Test Private

Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

New file added #1

Merged

Edit Code

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

[redacted]

New file added

New file e23de41

sidjain8427 merged commit 9bb93a7 into main 2 hours ago Revert

Write Preview

Leave a comment

Reviewers
No reviews

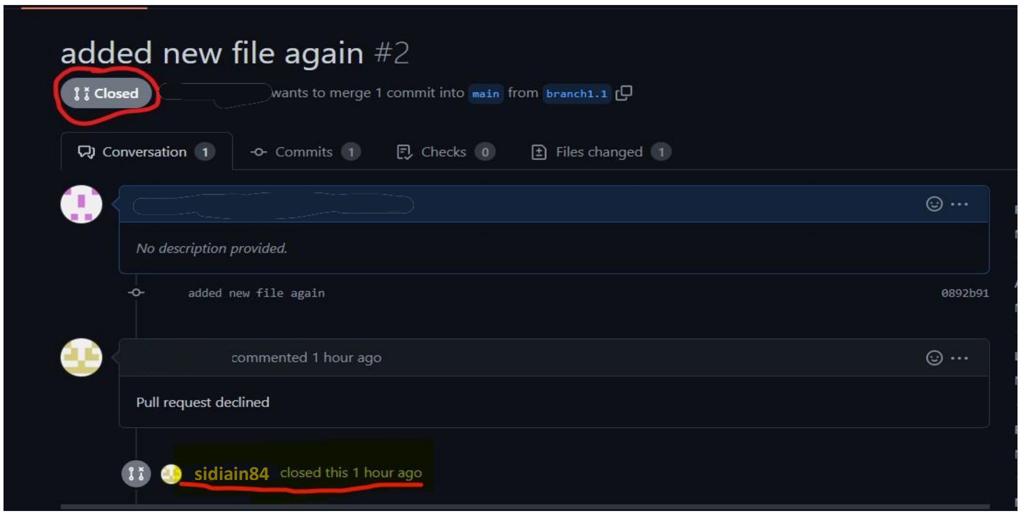
Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone

- The result of closing the request is shown below.


added new file again #2

Closed [redacted] wants to merge 1 commit into main from branch1.1

Conversation 1 Commits 1 Checks 0 Files changed 1

[redacted]

No description provided.

added new file again 0892b91

commented 1 hour ago

Pull request declined

sidiain84 closed this 1 hour ago

- Thus, we conclude opening and closing of pull request. We alsoconclude merging of the pull request to the main branch.

Aim: **Publish and print network graphs**

The network graph is one of the useful features for developers on GitHub. It is used to display the branch history of the entire repository network, including branches of the root repository and branches of forks that contain commits uniqueto the network.

A repository's graphs give you information on traffic, projects that depend onthe repository, contributors and commits to the repository, and a repository's forks and network. If you maintain a repository, you can use this data to get abetter understanding of who's using your repository and why they're using it.

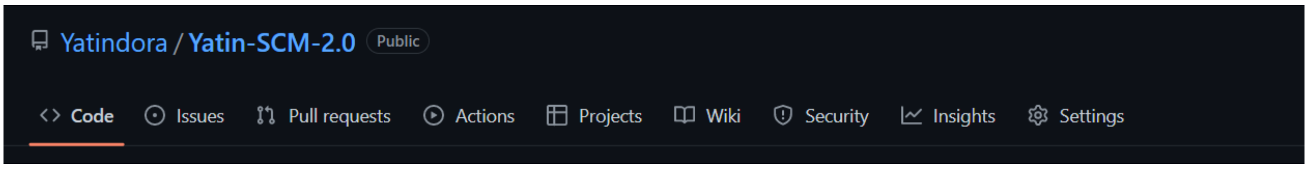
Some repository graphs are available only in public repositories with GitHubFree:

- Pulse
- Contributors
- Traffic
- Commits
- Code frequency
- Network

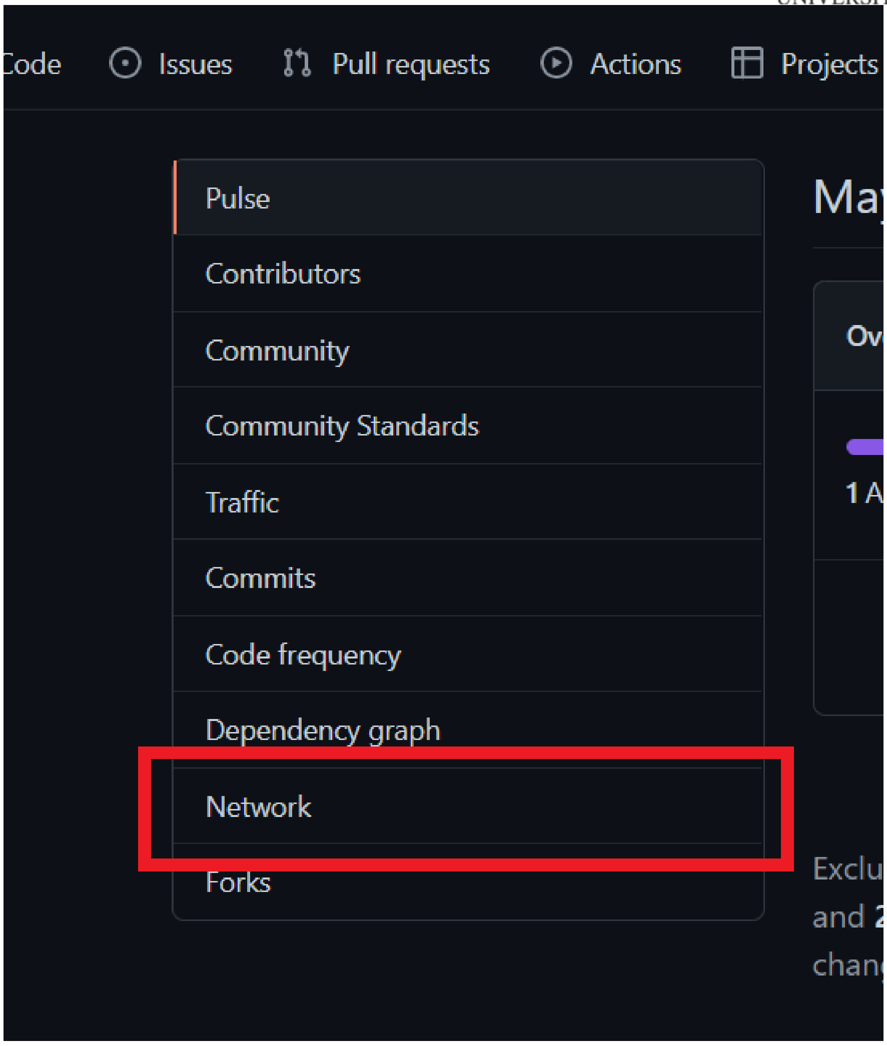
<u>Steps to acess network graphs of respective repository</u>

1. On GitHub.com, navigate to the main page of the repository.2.Under

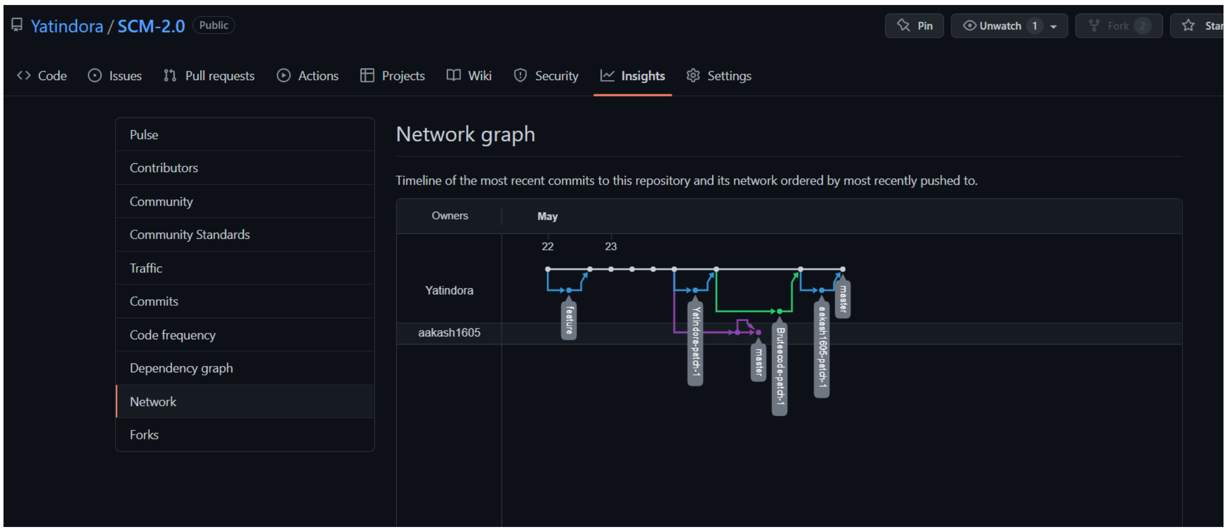
your repository name, click Insights.



3.At the left sidebar, click on Network.



You will get the network graph of your repository which displays the branchhistory of the entire repository network, including branches of the root repository and branches of forks that contain commits unique to the network.



Network graph

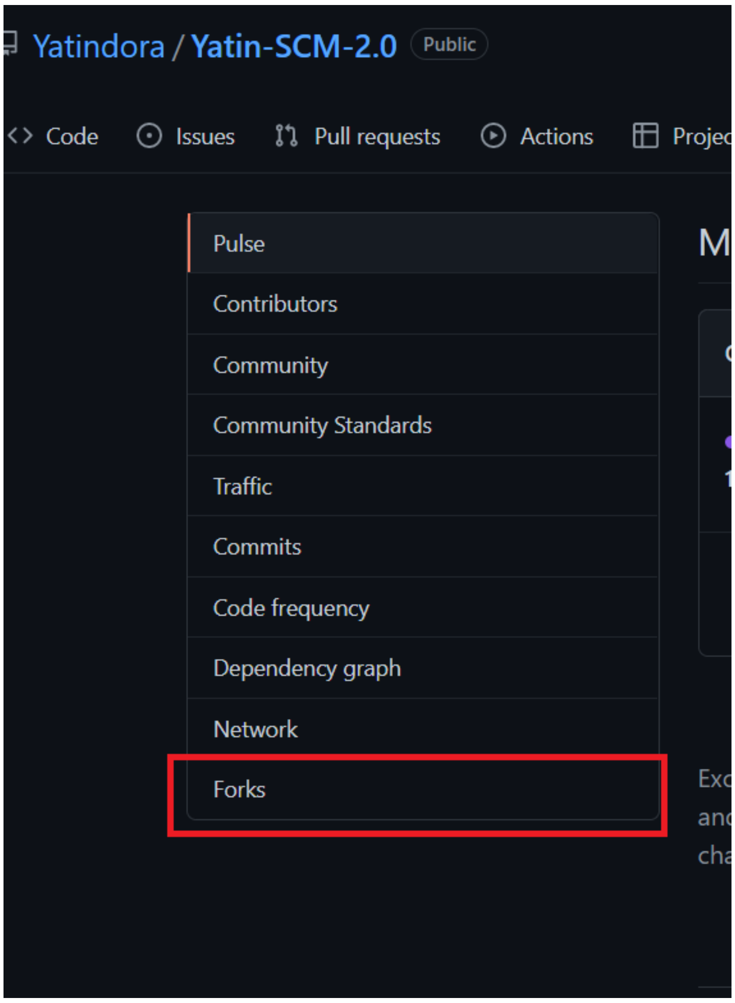
Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.

Listing the forks of a repository

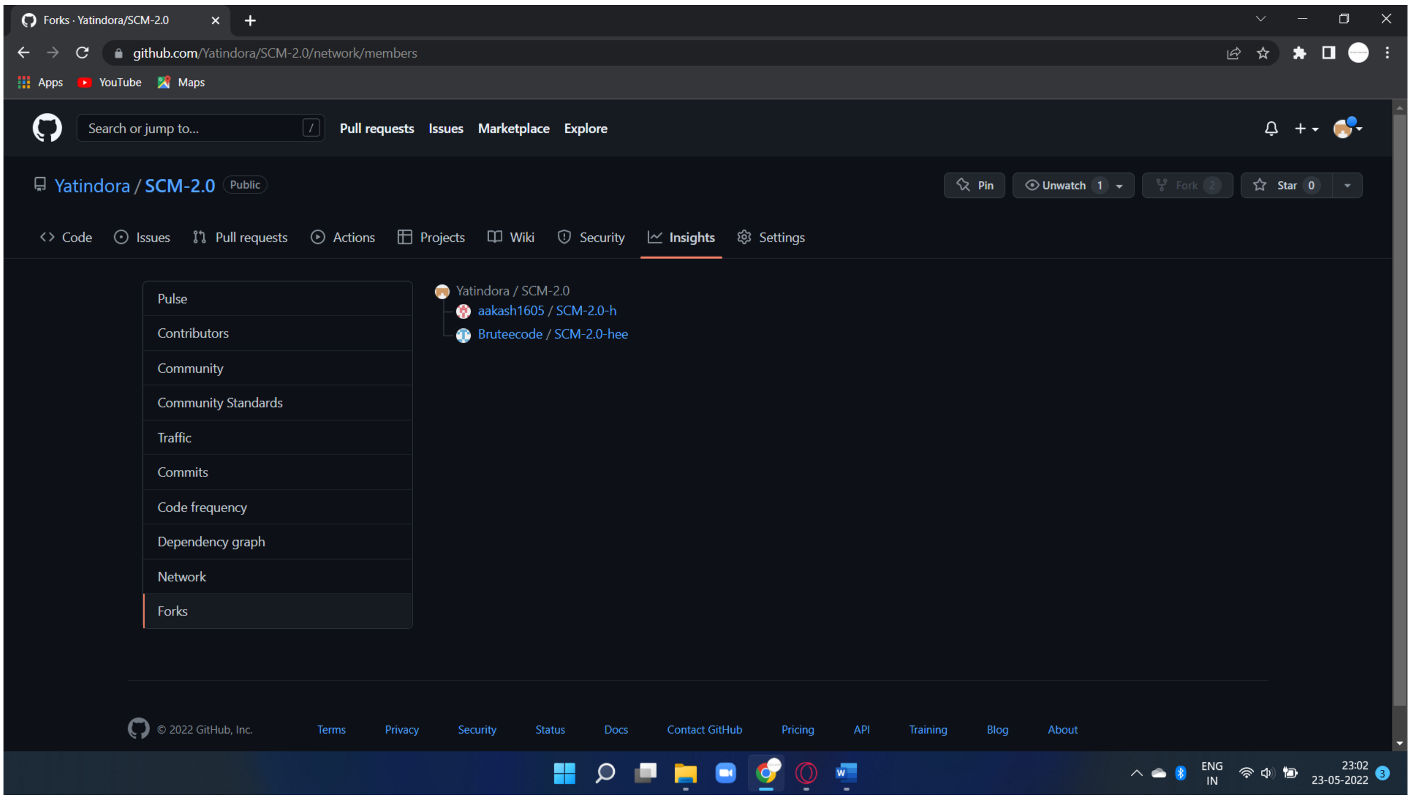
Forks are listed alphabetically by the username of the person who forked therepository

Clicking the number of forks shows you the full network. From there you canclick "members" to see who forked the repo

1. On GitHub.com, navigate to the main page of the repository.
2. Under your repository name, click Insights.



3. In the left sidebar, click Forks.



Here you can see all the forks

Viewing the dependencies of a repository

You can use the dependency graph to explore the code your repository depends on.

Almost all software relies on code developed and maintained by other developers, often known as a supply chain. For example, utilities, libraries, and frameworks. These dependencies are an integral part of your code and any bugs or vulnerabilities in them may affect your code. It's important to review and maintain these dependencies.

CHITKARA UNIVERSITY

6. Reference

S.No.	
1	https://www.geeksforgeeks.org/version-control-systems/
2	https://www.perforce.com/blog/vcs/what-is-version-control
3	https://medium.com/@lanceharvieruntime/version-control-why-do-we-need-it-1681f4888cec
4	https://en.wikipedia.org/wiki/Git
5	https://git-scm.com/
6	https://www.simplilearn.com/tutorials/git-tutorial/git-vs-github