

# Git and GitLab assignment

## Main Task: GitHub & GitLab Collaboration & Workflow Setup

You are part of a DevOps team working on two different platforms – GitHub and GitLab. Your team wants to ensure smooth development, proper access control, and repository mirroring between the two platforms.

## Part 2 : GitLab Tasks

### Subtask 4: GitLab Repository Setup

#### 1. Create a private repository on GitLab.

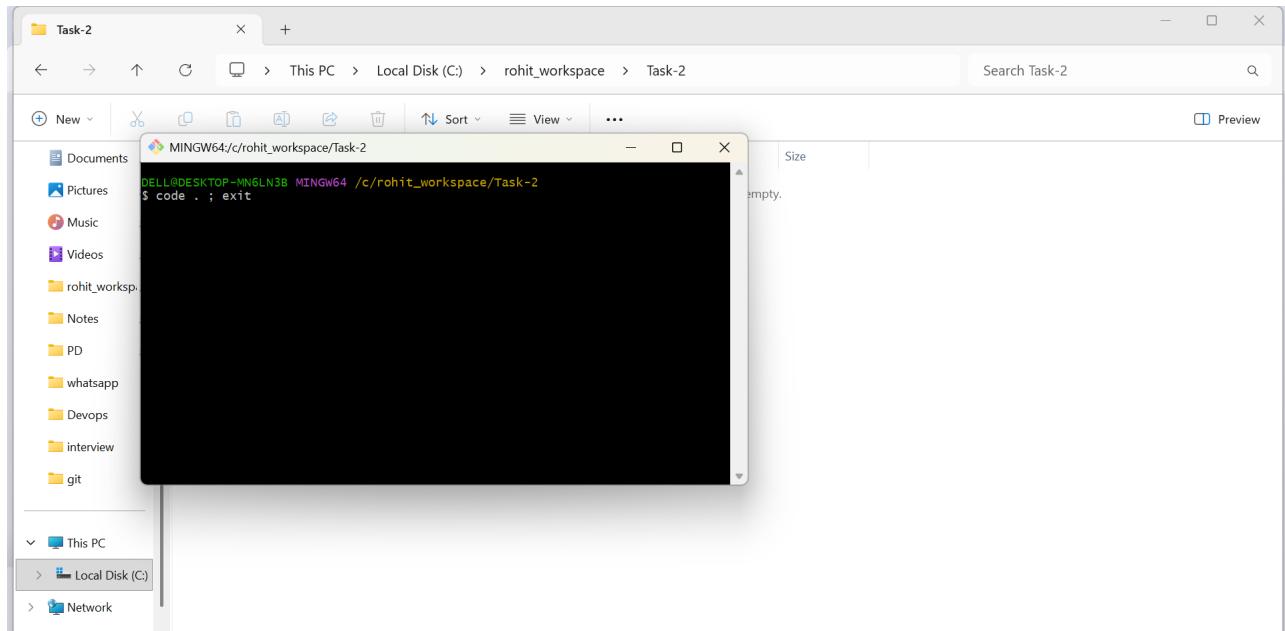
- Created On Private Repo On GitLab

#### 2. Clone it on your local machine using SSH (not HTTPS).

- Copy The URL SSH URL

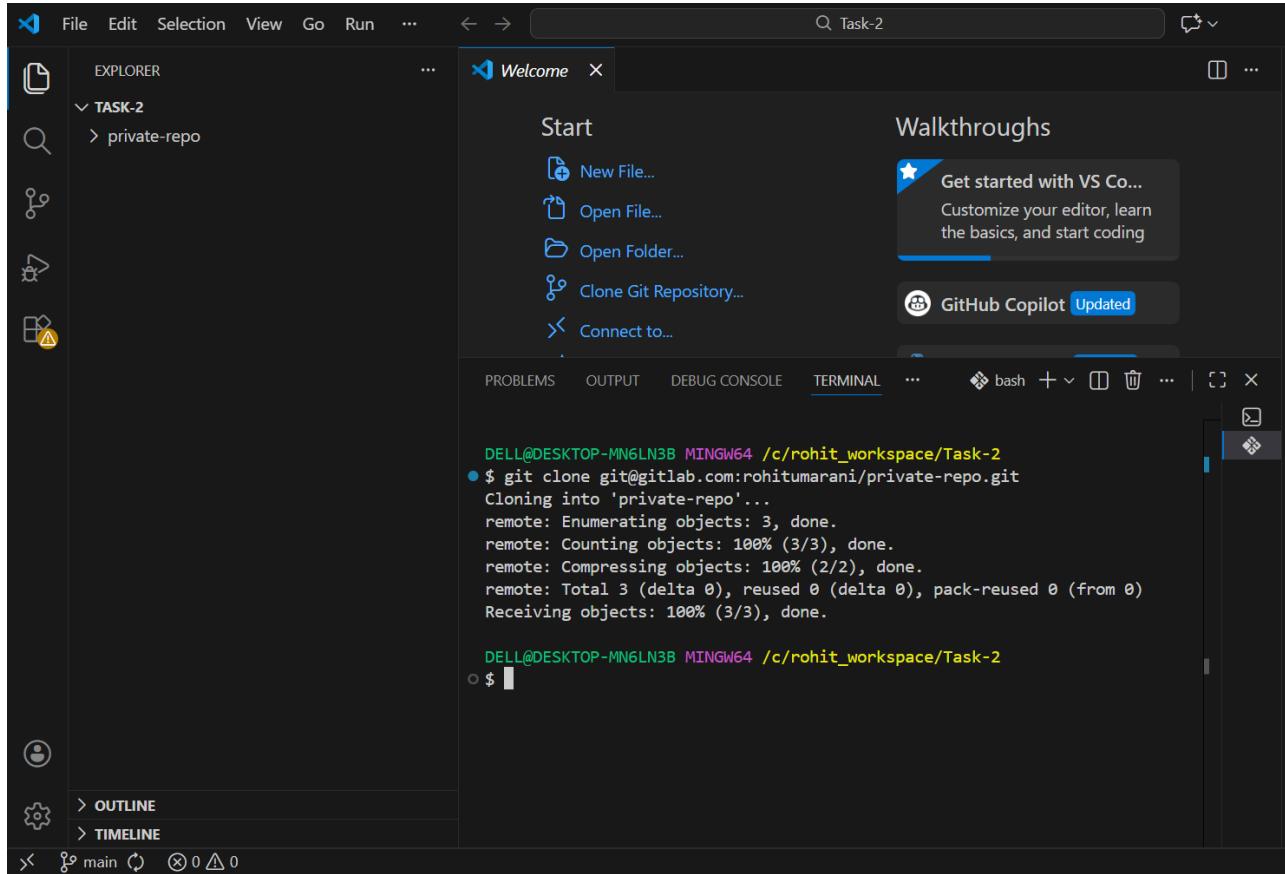
Name	Last commit	Last update
README.md	Initial commit	2 minutes ago
README.md		

- Create One Folder Into Local Machine

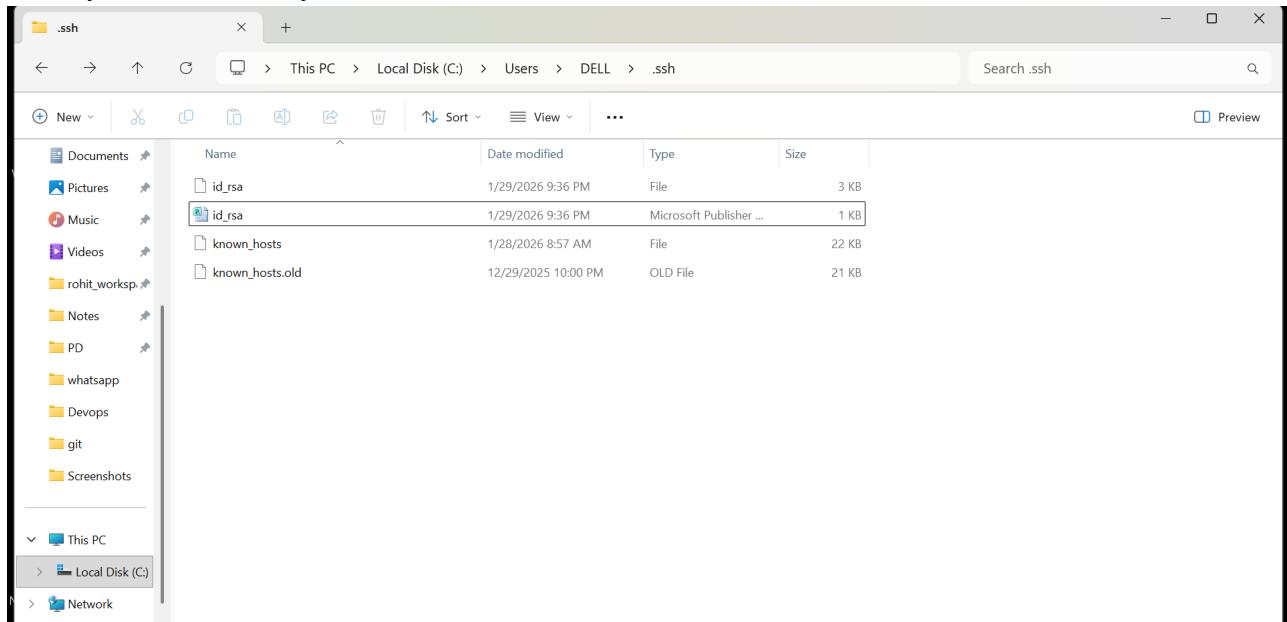


- Use Command For Clone Private Repo To Local Machine

```
git clone git@gitlab.com:rohitumarani/private-repo.git
```



- Already Created SSH Key



How to check the Path:

C:\Users\DELL\.ssh

- SSH Key On GitLab

The screenshot shows the GitLab User Settings page at [gitlab.com/~user\\_settings/ssh\\_keys](https://gitlab.com/~user_settings/ssh_keys). The sidebar is open, showing options like Applications, Integration accounts, Personal access tokens, Emails, Password, Notifications, SSH Keys (which is selected and highlighted in blue), GPG keys, Preferences, Comment templates, Active sessions, Authentication log, Usage quotas, What's new, Help, and Collapse sidebar. The main content area is titled "User Settings / SSH Keys". It includes a search bar and a "SSH Keys" section with the following information:

Your SSH keys 1							
Title	Key	Usage type	Created	Last used	Expires	Action	
DELL@DESKTOP-MN6LN3B	74:c1:ca:bb:3e:a1:90:21:23:cc:1c:57:9e:f3:44:66	Authentication & Signing	5 days ago	7 minutes ago	2027-01-29	<button>Revoke</button>	

### 3. Create a simple project structure (e.g., src/app.py , docs/guide.md).

- Created Src Folder inside that app.py file

The screenshot shows the VS Code interface with the following details:

- EXPLORER** view: A repository named "TASK-2" is selected, containing a folder "private-repo". Inside "private-repo", there is a file "app.py" and a file "README.md".
- WELCOME** view: Shows options like "Start", "Walkthroughs", and "Recent". Walkthroughs include "Get started with VS Code" and "GitHub Copilot".
- TERMINAL** view: Displays the command history for creating a "src" directory:
 

```
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2
$ cd private-repo/
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo (main)
$ mkdir src
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo (main)
$ cd src
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo/src (main)
$ touch app.py
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo/src (main)
$ 
```

- Created docs Folder inside that guide.md file

The screenshot shows the VS Code interface with the following details:

- EXPLORER** view: A repository named "TASK-2" is selected, containing a folder "private-repo". Inside "private-repo", there is a folder "docs" which contains a file "guide.md", and a folder "src" which contains a file "app.py" and a file "README.md".
- TERMINAL** view: Displays the command history for creating a "docs" directory:
 

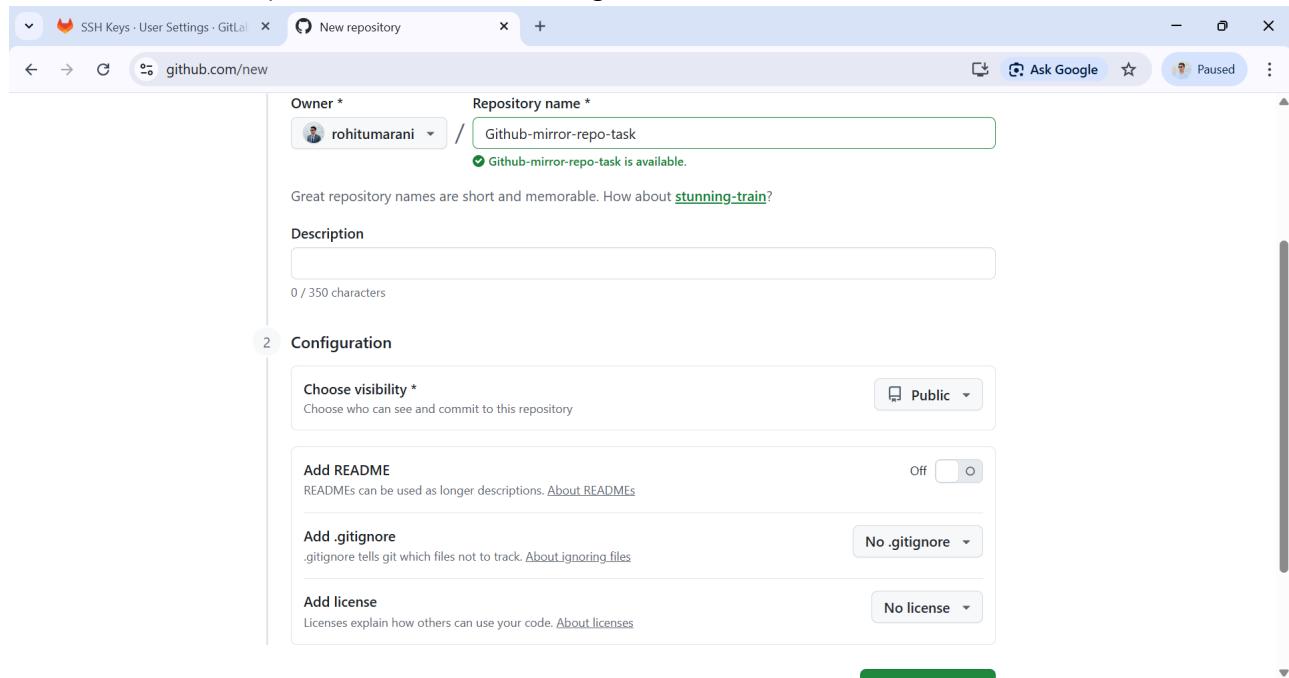
```
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo (main)
$ mkdir docs
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo (main)
$ cd docs
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo/docs (main)
$ touch guide.md
DELL@DESKTOP-MN6LN3B MINGW64 /c/rohit_workspace/Task-2/private-repo/docs (main)
$ 
```

## Subtask 5: Repository Mirroring

### Create a mirror setup:

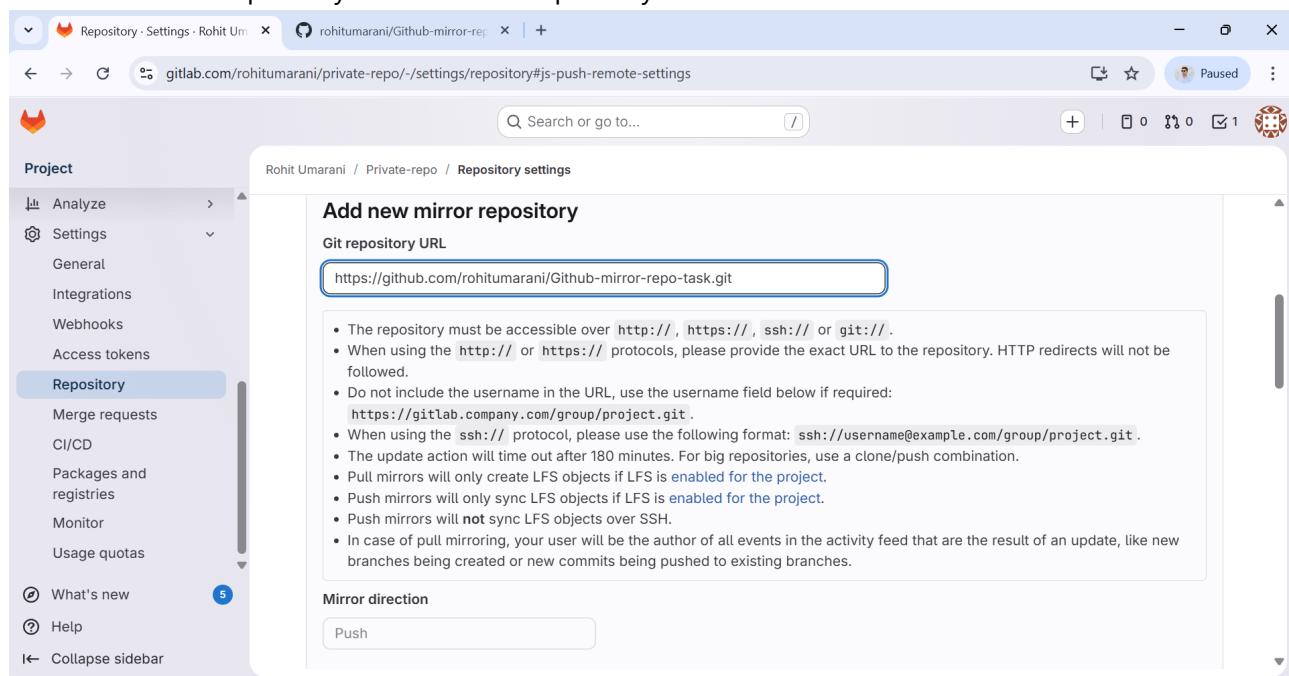
#### 1. Set the GitHub private repo as the mirror of your GitLab repo.

- Created One New Repo On Github For Mirroring



The screenshot shows the GitHub 'New repository' creation interface. The 'Owner' dropdown is set to 'rohitumarani'. The 'Repository name' field contains 'Github-mirror-repo-task', with a note below it stating 'Github-mirror-repo-task is available.' A link to 'stunning-train' is provided as a suggestion. The 'Description' field is empty, with a note indicating 0 / 350 characters. The 'Configuration' section is expanded, showing options for visibility ('Public'), README ('Off'), .gitignore ('No .gitignore'), and license ('No license'). A progress bar at the bottom indicates the task is 100% complete.

- Add This Github Repository to the GitLab Repository



The screenshot shows the 'Add new mirror repository' configuration in GitLab. The 'Git repository URL' field contains 'https://github.com/rohitumarani/Github-mirror-repo-task.git'. A detailed list of instructions follows:

- The repository must be accessible over `http://`, `https://`, `ssh://` or `git://`.
- When using the `http://` or `https://` protocols, please provide the exact URL to the repository. HTTP redirects will not be followed.
- Do not include the username in the URL, use the username field below if required: `https://gitlab.company.com/group/project.git`.
- When using the `ssh://` protocol, please use the following format: `ssh://username@example.com/group/project.git`.
- The update action will time out after 180 minutes. For big repositories, use a clone/push combination.
- Pull mirrors will only create LFS objects if LFS is enabled for the project.
- Push mirrors will only sync LFS objects if LFS is enabled for the project.
- Push mirrors will not sync LFS objects over SSH.
- In case of pull mirroring, your user will be the author of all events in the activity feed that are the result of an update, like new branches being created or new commits being pushed to existing branches.

The 'Mirror direction' field is set to 'Push'.

- Create Personal Access Token. It Required For Password Authentication

The screenshot shows the GitHub 'Personal access tokens (classic)' page. On the left, there's a sidebar with 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens' (which is selected). Under 'Personal access tokens', there are 'Fine-grained tokens' and 'Tokens (classic)'. The main area displays two tokens:

- new-devops-mirror-repo**: Scopes: admin:enterprise, admin:gpg\_key, admin:org, admin:org\_hook, admin:public\_key, admin:repo\_hook, admin:ssh\_signing\_key, audit\_log, codespace, copilot, delete:packages, delete\_repo, gist, notifications, project, repo, user, workflow. Status: Never used. Expires on Sat, Feb 28 2026.
- mirror-repo**: Scopes: admin:enterprise, admin:gpg\_key, admin:org, admin:org\_hook, admin:public\_key, admin:repo\_hook, admin:ssh\_signing\_key, audit\_log, codespace, copilot, delete:packages, delete\_repo, gist, notifications, project, repo, user, workflow, write:discussion, write:network\_configurations, write:packages. Status: Never used. Expires on Sun, Mar 29 2026.

A button 'Generate new token' is visible at the top right.

- After That Paste That Token And Setup Up GitLab and GitHub

The screenshot shows the GitLab 'Repository settings' page for a repository named 'Private-repo'. The sidebar on the left has 'Repository' selected. The main form is titled 'Mirror repository' and contains the following fields:

- Mirror direction**: Push
- Authentication method**: Username and Password
- Username**: rohitumarani
- Password**: (redacted)
- Keep divergent refs**: (unchecked)
- Mirror only protected branches**: (unchecked)

At the bottom are 'Mirror repository' and 'Cancel' buttons.

## 2. Push some changes to GitLab and verify if the changes reflect in GitHub automatically.

- Push the Changes

The screenshot shows the VS Code interface with the terminal tab selected. The terminal window displays the command \$ git push -u origin main followed by its execution output. The output shows the process of enumerating objects, calculating delta compression, and writing objects to the repository.

```

DELL@DESKTOP-MNGLN3B MINGW64 /c/rohit_workspace/Task-2/private-repo (main)
$ git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 343 bytes | 114.00 KiB/s, done.
Writing objects: 100% (4/4), 343 bytes | 114.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
branch 'main' set up to track 'origin/main'.
branch 'main' set up to track 'origin/main'.
branch 'main' set up to track 'origin/main'.

```

### Use Command For Push

```
git push -u origin main
```

- Check On GitLab Changes

The screenshot shows a web browser displaying a GitLab project page. The sidebar on the left shows pinned issues and merge requests. The main content area shows the project's repository details, including the creation date (February 04, 2026) and a recent commit named "second commit" by Rohit Umarani. The commit details show it was authored 21 seconds ago and has a commit hash of 0ac57a56. Below the commit, a table lists files with their last commits and update times.

Name	Last commit	Last update
docs	first commit	10 minutes ago
src	second commit	21 seconds ago
README.md	Initial commit	57 minutes ago
README.md		

- Check On Github Changes

The screenshot shows a GitHub repository named "Github-mirror-repo-task". The repository is public and has 3 commits. The commits are:

- rohitumarani second commit (0ac57a5 · 2 minutes ago)
- docs first commit (12 minutes ago)
- src second commit (2 minutes ago)
- Initial commit (1 hour ago)

The repository has 0 stars, 0 forks, and 0 releases. The README file is visible.

## Subtask 6: Access Control

### Invite your friend to the GitLab private repository:

- Invite Member to the Repository

The screenshot shows a GitLab project settings page for "Private-repo". In the "Members" section, there is one member listed: "Pranav Khade" (pranav-khade-tech). The "Invite members" dialog is open, showing the input field "pranav-khade-tech" and a suggestion "Pranav Khade pranav-khade-tech". The dialog also includes fields for "Role" (set to "Guest"), "Access expiration date (optional)", and "Expiration date".

- Assign them the Guest role initially, observe the access.

The screenshot shows the GitLab 'Members' page for a project. On the left, a sidebar menu is open with 'Members' selected. The main area displays a single member entry for 'Pranav Khade'. The member's profile includes their GitHub icon, name, handle (@pranav-khade-tech), and role, which is currently set to 'Guest'. Below the member's details, there are sections for 'Account', 'Source', 'Role' (set to 'Guest'), 'Expiration' (with an input field for 'Expiration date'), 'Activity' (listing specific activity dates), and 'Actions' (with a three-dot menu). At the bottom right of the main content area, there is a small message: 'Rohit Umarani It's you!'

- Then change their role to Developer, and let them push one file.

This screenshot shows the same GitLab 'Members' page, but with a 'Change role' modal window open over the main content. The modal has a title 'Change role' and a list of roles: Guest, Planner, Reporter, and Developer. The 'Developer' role is selected and highlighted with a blue background. Below the list is a 'Description' section with a note about the Guest role and its limitations. At the bottom of the modal are 'Update role' and 'Cancel' buttons. The rest of the interface, including the sidebar and other member entries, remains visible in the background.

## Role is Change To the Devolper

The screenshot shows a GitLab interface for a project named 'Members - Rohit Umarani / Private-repo'. The sidebar on the left is collapsed, and the main content area is titled 'Members' with a count of 2. A search bar at the top right contains the placeholder 'Search or go to...'. Below the search bar are buttons for 'Filter members' and 'Account'. The member list displays one user: Pranav Khade (@pranav-khade-tech), represented by a green profile icon. The details for Pranav Khade show him as a 'Direct member by Rohit Umarani' with the role 'Developer'. Below the member list, sections for 'Source' and 'Expiration' are visible, along with an 'Activity' section showing recent logins on Jan 27, 2026, Feb 04, 2026, and Feb 03, 2026.