

## Lesson 03 Demo 05

### Rebasing in Git

**Objective:** To perform rebase in Git for integrating changes from one branch to another while maintaining a linear commit history

**Tools required:** Git and GitHub

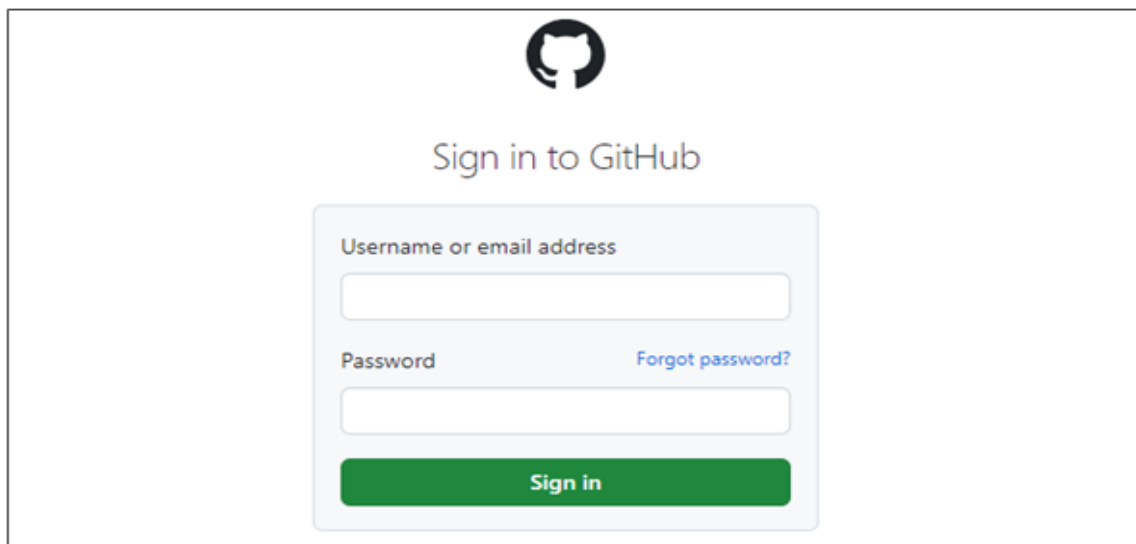
**Prerequisites:** None

Steps to be followed:

1. Create a repository
2. Clone the Git repository and rebase the branch to integrate the changes

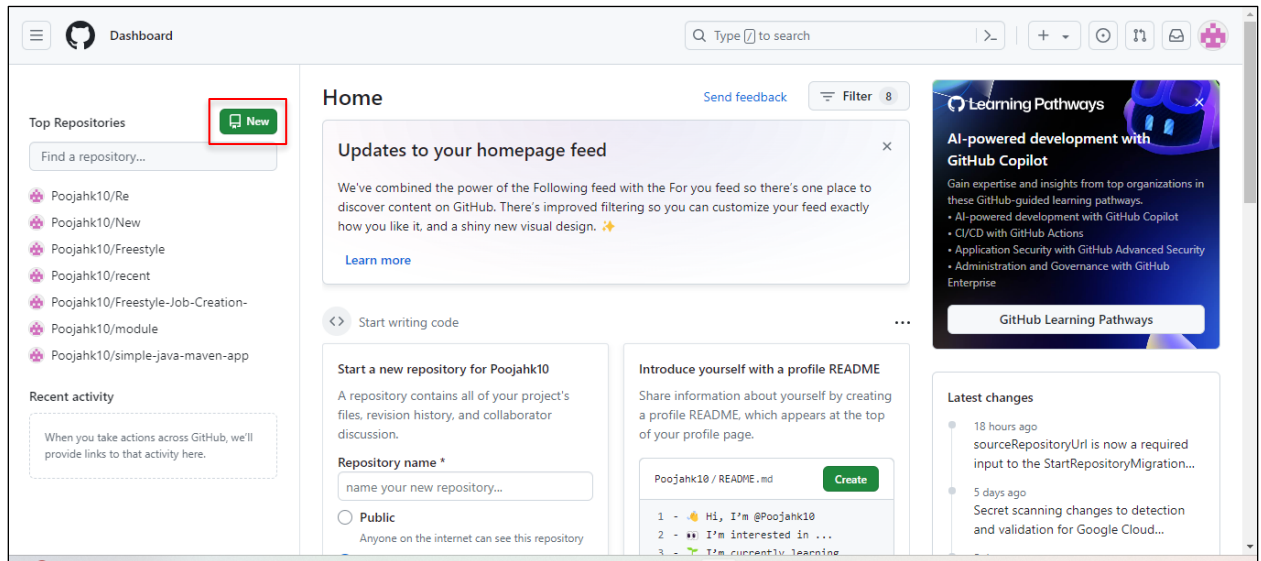
#### Step 1: Create a repository

- 1.1 Open the browser in your practice lab, navigate to **github.com**, and sign in to your account as shown in the screenshot below:



**Note:** If you do not have a GitHub account, visit the official website at <https://github.com/signup> and create a new account

1.2 Click on the **New** button as shown in the screenshot below:




1.3 Enter a desired name for your repository and choose **Public** as shown in the screenshot below:

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*


**Owner \*** **Repository name \***


 Poojahk10 /

✓ Rebase is available.


Great repository names are short and memorable. Need inspiration? How about [verbose-umbrella](#) ?


**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.

1.4 Click on the **Add a README file** checkbox and then click on **Create repository** as shown in the screenshot below:

☒  **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:

☒ **Add a README file**

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore


.gitignore template: None ▾


Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None ▾

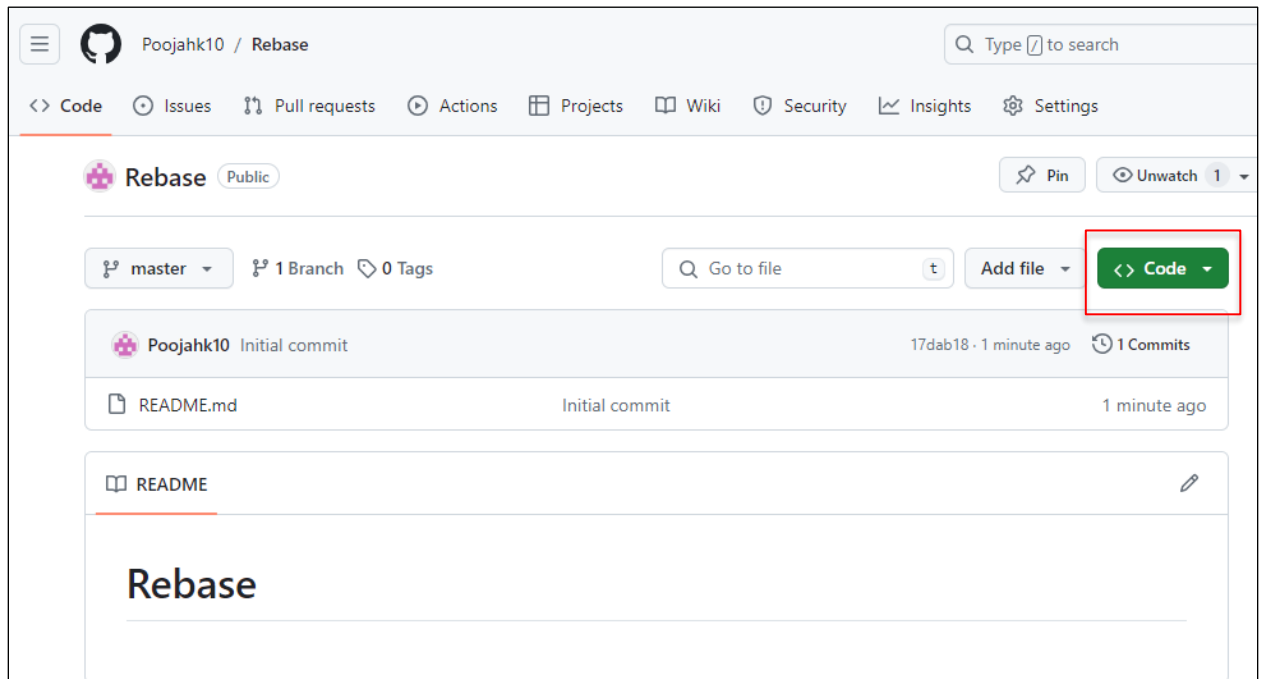
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  master as the default branch. Change the default name in your [settings](#).

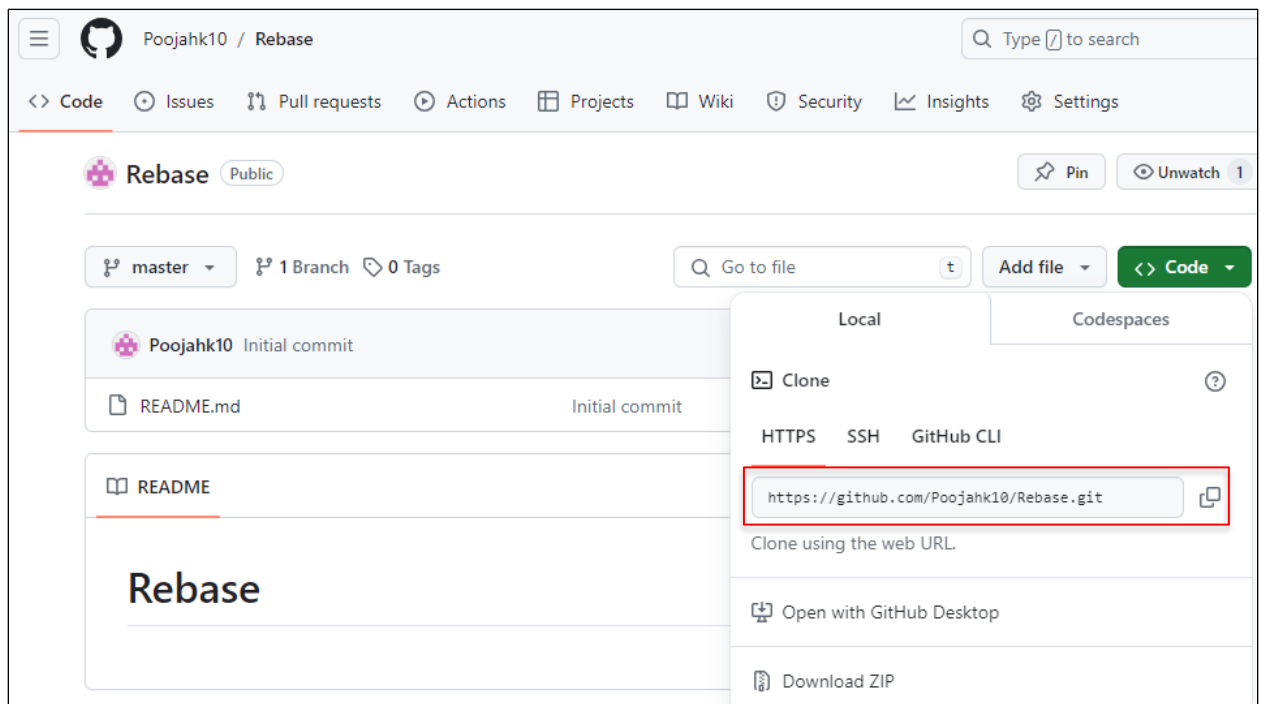
 You are creating a public repository in your personal account.

Create repository

1.5 Click on **Code** as shown in the screenshot below:



1.6 Copy the repository URL as shown in the screenshot below:



## Step 2: Clone the Git repository and rebase the branch to integrate the changes

- 2.1 Open the Linux terminal in your lab and clone the repository using the following command:

**git clone <RepositoryURL>**

```
syedsharozsimpl@ip-172-31-40-171:~$ git clone https://github.com/Poojahk10/Rebase.git
Cloning into 'Rebase'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
syedsharozsimpl@ip-172-31-40-171:~$
```

**Note:** Add the URL of your GitHub account repository in the place of <RepositoryURL>

- 2.2 Navigate inside the repository that you had created using the following command:

**cd RepositoryName/**

```
syedsharozsimpl@ip-172-31-40-171:~$ cd Rebase/
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ █
```

- 2.2 List the branches using the following command:

**git branch**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git branch
* master
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ █
```

- 2.3 Create a new branch using the following command:

**git branch feature1**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git branch feature1
syedsharozsimpl@ip-172-31-40-171:~/Rebase$
```

- 2.4 Switch to the branch that you created in the previous step using the following command:

**git checkout feature1**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git checkout feature1
Switched to branch 'feature1'
syedsharozsimpl@ip-172-31-40-171:~/Rebase$
```

2.5 List the branches using the following command:

**git branch**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git branch
* feature1
  master
syedsharozsimpl@ip-172-31-40-171:~/Rebase$
```

2.6 Create a file, stage the changes, and commit the staged changes in **feature1** branch using the following commands:

**touch f1**

**git add .**

**git commit -m "f1 added"**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ touch f1
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git add .
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git commit -m "f1 added"
[feature1 60e5b1e] f1 added
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 f1
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ █
```

2.7 Create another file, stage the changes, and commit the staged changes using the following commands:

**touch f11**

**git add .**

**git commit -m "f11 added"**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ touch f11
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git add .
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git commit -m "f11 added"
[feature1 c34960e] f11 added
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 f11
```

2.8 Run the following command to display the commit history of a repository:

**git log**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git log
commit c34960e3532ee3b0e1dd5e17d0e9f52287fe7bb8 (HEAD -> feature1)
Author: Your Name <you@example.com>
Date:   Tue Apr 16 17:07:37 2024 +0000

    f11 added

commit 8bf228fc5bebc9573586a3bf99f0245383a4268d
Author: Your Name <you@example.com>
Date:   Tue Apr 16 17:03:54 2024 +0000

    f1 added
```

2.9 Switch to the master branch using the following command:

**git checkout master**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
syedsharozsimpl@ip-172-31-40-171:~/Rebase$
```

2.10 List the branches using the following command:

**git branch**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git branch
  feature1
* master
syedsharozsimpl@ip-172-31-40-171:~/Rebase$
```

- 2.11 Create a file inside the master branch, stage the changes, and commit the staged changes using the following commands:

**touch f22**

**git add .**

**git commit -m "f22 added"**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ touch f22
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git add .
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git commit -m "f22 added"
[master 63f3858] f22 added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 f22
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ █
```

- 2.12 Run the following command to display the commit history of a repository:

**git log**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git log
commit 63f38589e5daff12f9ccc0b61b5a493b68225697 (HEAD -> master)
Author: Your Name <you@example.com>
Date:   Tue Apr 16 17:11:44 2024 +0000

    f22 added

commit 17dab1877dead8ae2b6ede61624388125dc649c3 (origin/master, origin/HEAD)
Author: Poojahk10 <153164498+Poojahk10@users.noreply.github.com>
Date:   Tue Apr 16 22:20:14 2024 +0530

    Initial commit
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ █
```

- 2.13 Run the following command to rebase both branches:

**git rebase feature1 master**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git rebase feature1 master
Successfully rebased and updated refs/heads/master.
syedsharozsimpl@ip-172-31-40-171:~/Rebase$
```



2.14 Run the following command to check the commit history of a repository after rebasing:  
**git log**

```
syedsharozsimpl@ip-172-31-40-171:~/Rebase$ git log
commit 4dbce2ecaf9a18cab913cfc986084b6cbbbee328a (HEAD -> master)
Author: Your Name <you@example.com>
Date:   Tue Apr 16 17:11:44 2024 +0000

    f22 added

commit c34960e3532ee3b0e1dd5e17d0e9f52287fe7bb8 (feature1)
Author: Your Name <you@example.com>
Date:   Tue Apr 16 17:07:37 2024 +0000

    f11 added

commit 8bf228fc5bebc9573586a3bf99f0245383a4268d
Author: Your Name <you@example.com>
Date:   Tue Apr 16 17:03:54 2024 +0000

    f1 added
```

By following these steps, you have successfully performed a rebasing in Git to integrate changes from one branch into another for seamless collaboration and code management.