

Git:

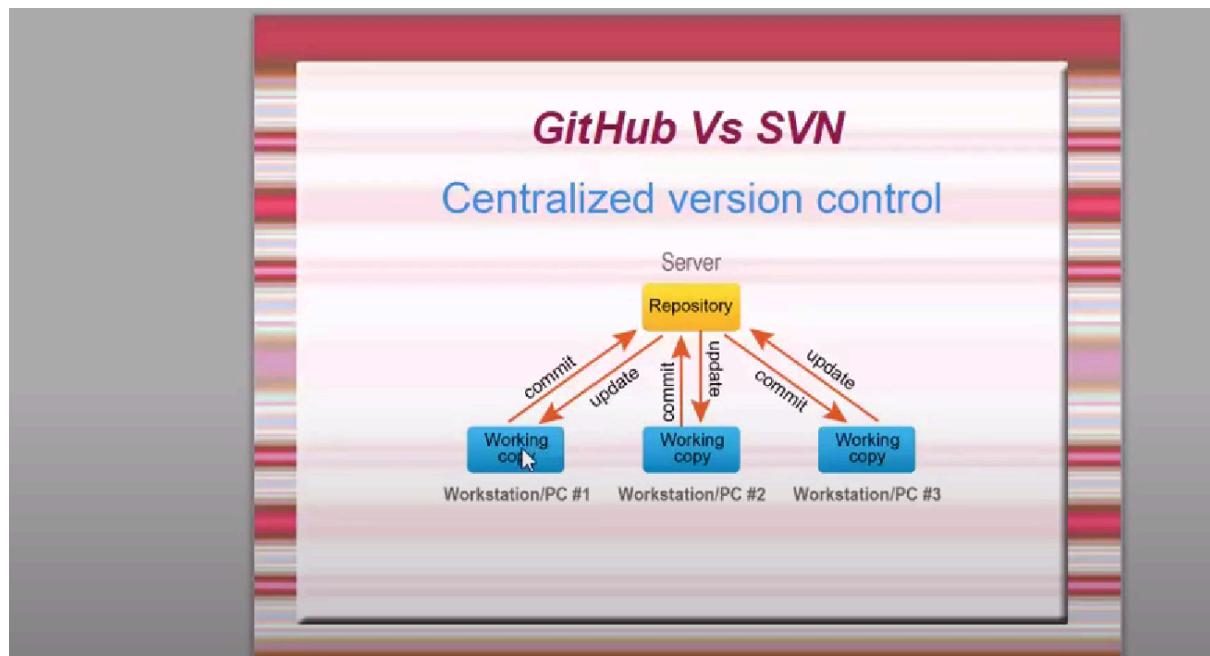
Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Difference between git and github:

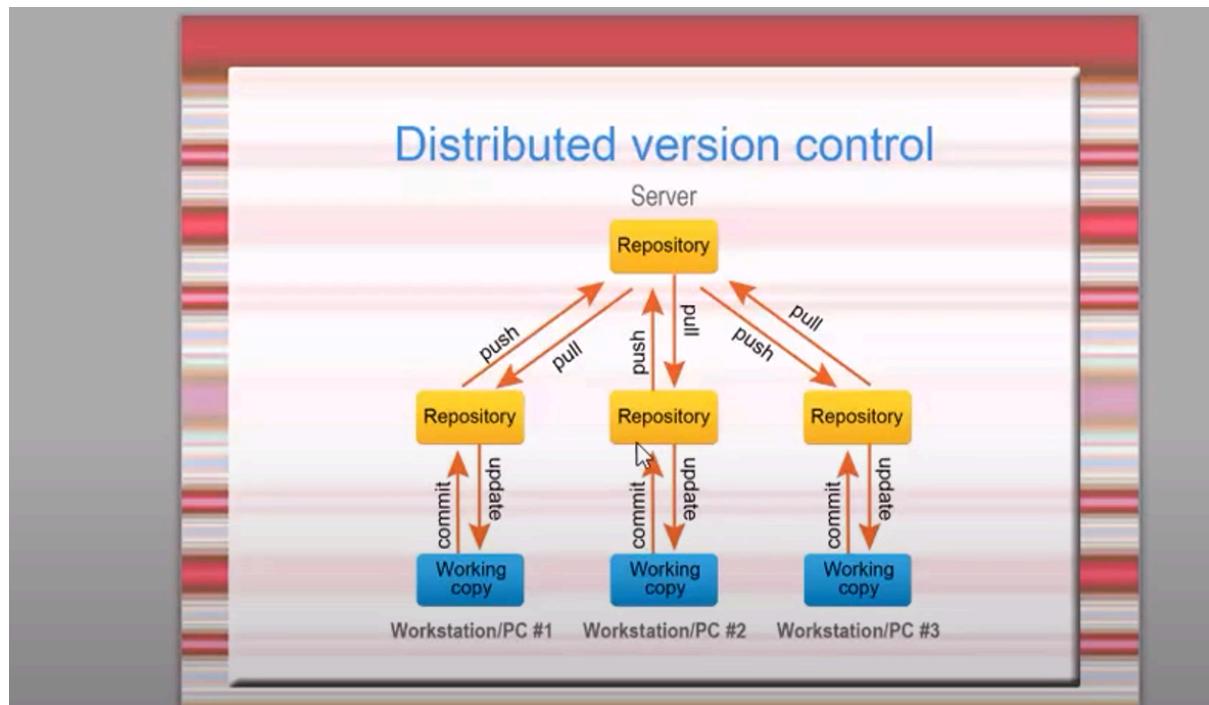
Git is a version control system that allows you to track changes to your code, and it is a great way to keep track of your work. It is also a great way to share your code with others, and it can be very useful when you are working on a project that involves a lot of collaboration.

Github is a web-based Git repository, that is. Cloud-based storage is offered by this hosting company. Git's distributed version control and source code management capabilities are all provided by GitHub, along with some additional features. Utilising Git facilitates collaboration. The public can access GitHub repositories as well. GitHub serves as a networking platform for online professionals since developers from all over the world may engage, contribute to one another's code, and edit or improve it. Social coding is another name for the process of participation and engagement.

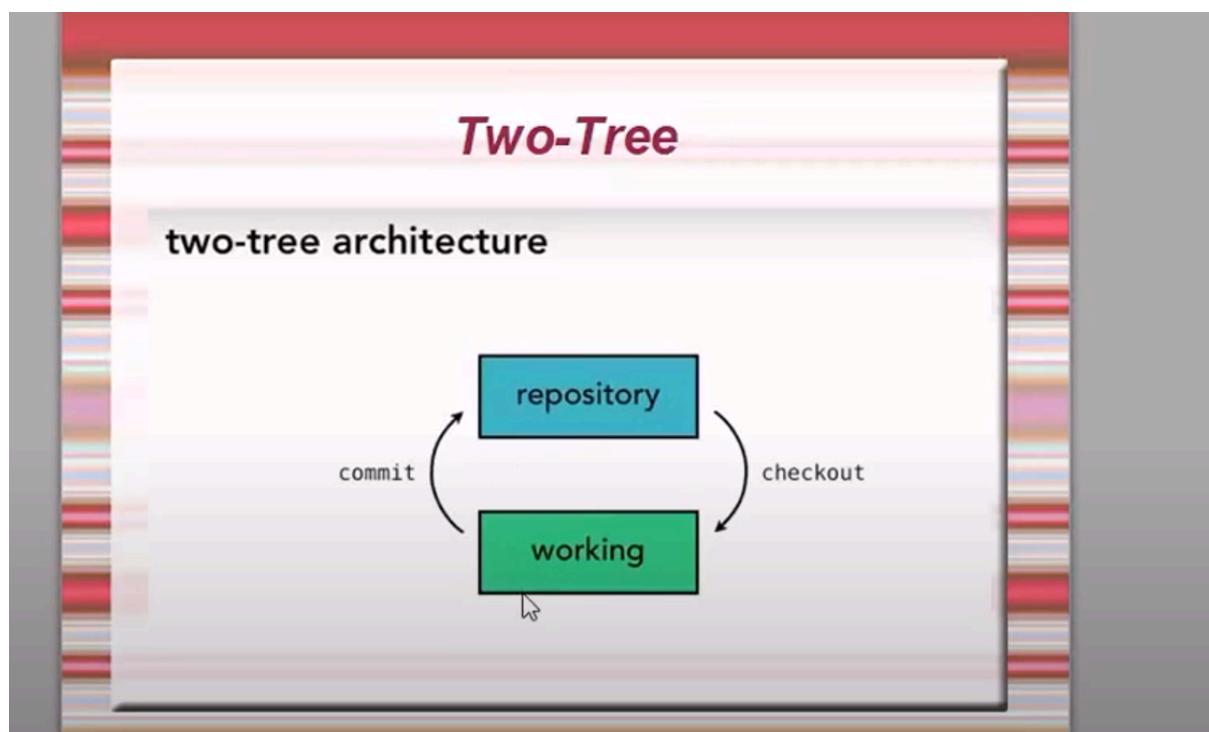
Difference between git and svn:



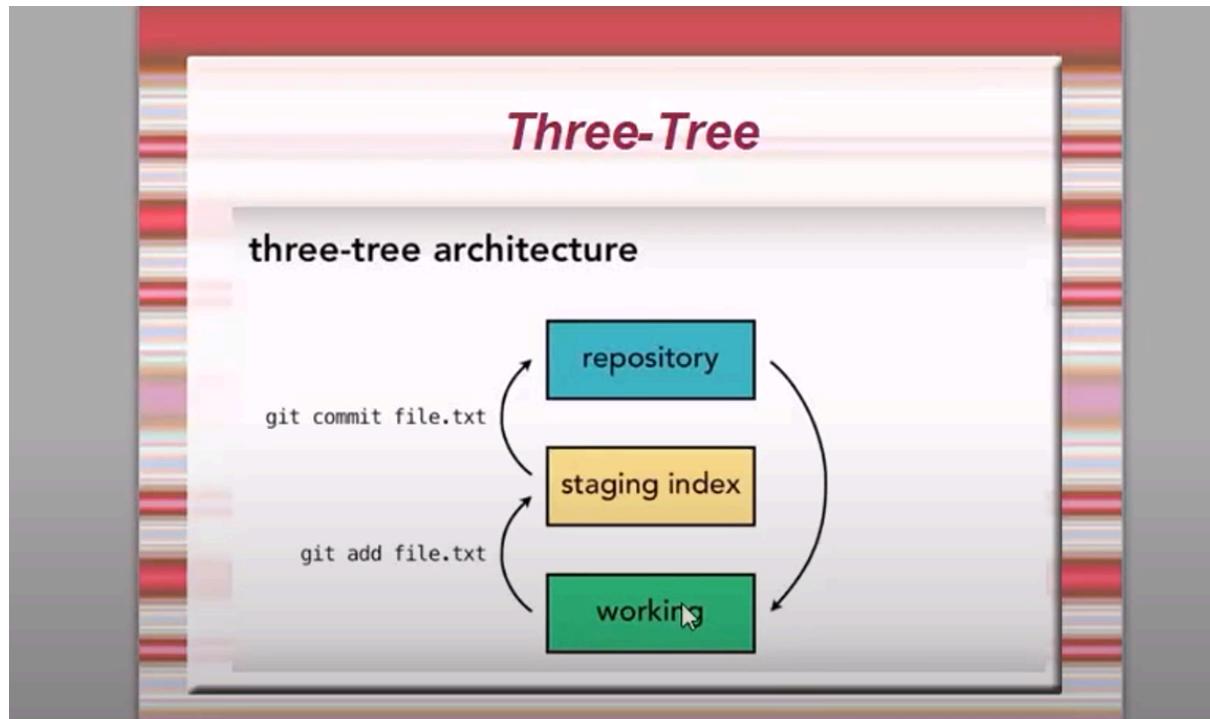
u must have internet connection.



SVN:



GIT:



Every changes there will be a version.

```
git log
commit d779105930fc... (HEAD -> master)
Author: basahota <techbasanta757@gmail.com>
Date: Thu Jun 28 12:50:28 2018 +0530

    testing

commit b83dd24527c8ec08fb08c993b35241facecc8303 (origin/master, origin/HEAD)
Author: Basant kumar Hota <basahota@users.noreply.github.com>
Date: Mon Jun 4 11:54:27 2018 +0530

    #code

commit d60a53bbb51a6e84d14e6640e03b164753dc3a32
Author: Basant kumar Hota <basahota@users.noreply.github.com>
Date: Mon Jun 4 11:53:48 2018 +0530

    Initial commit
```

Need to install git using below command

<https://git-scm.com/downloads>

The entire [Pro Git book](#) written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloads

[Mac OS X](#) [Windows](#) [Linux/Unix](#)

Older releases are available and the [Git source repository](#) is on GitHub.

GUI Clients

Git comes with built-in GUI tools (`git-gui`, `gitk`), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

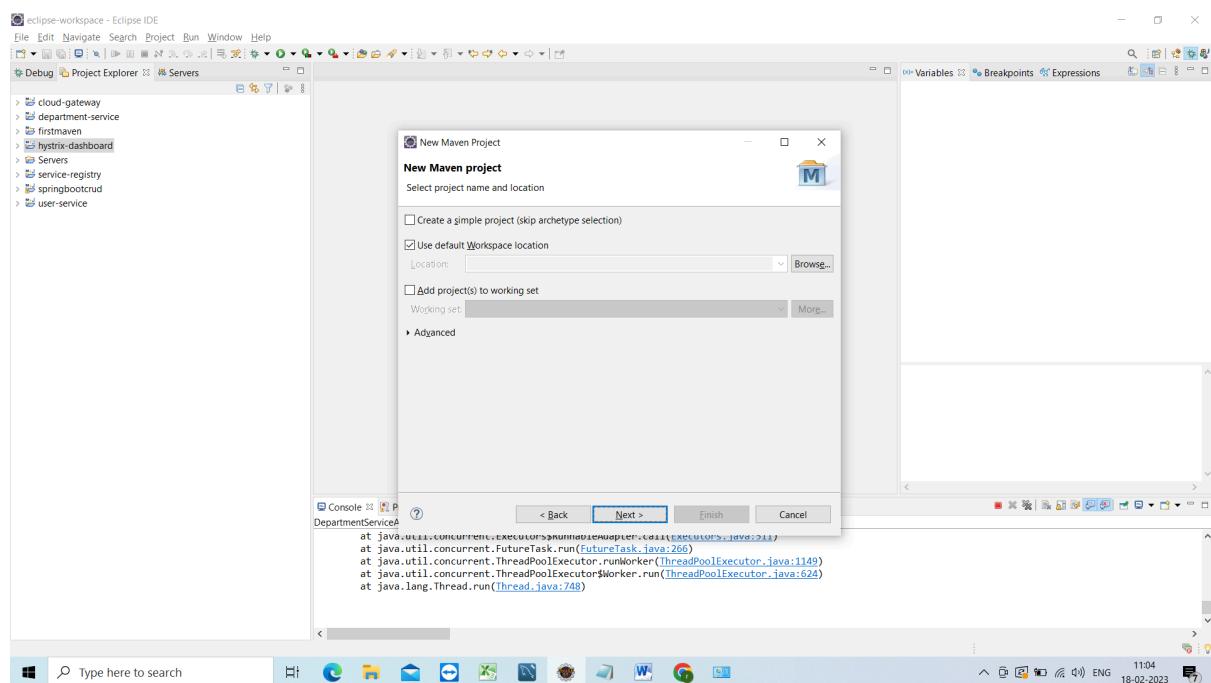
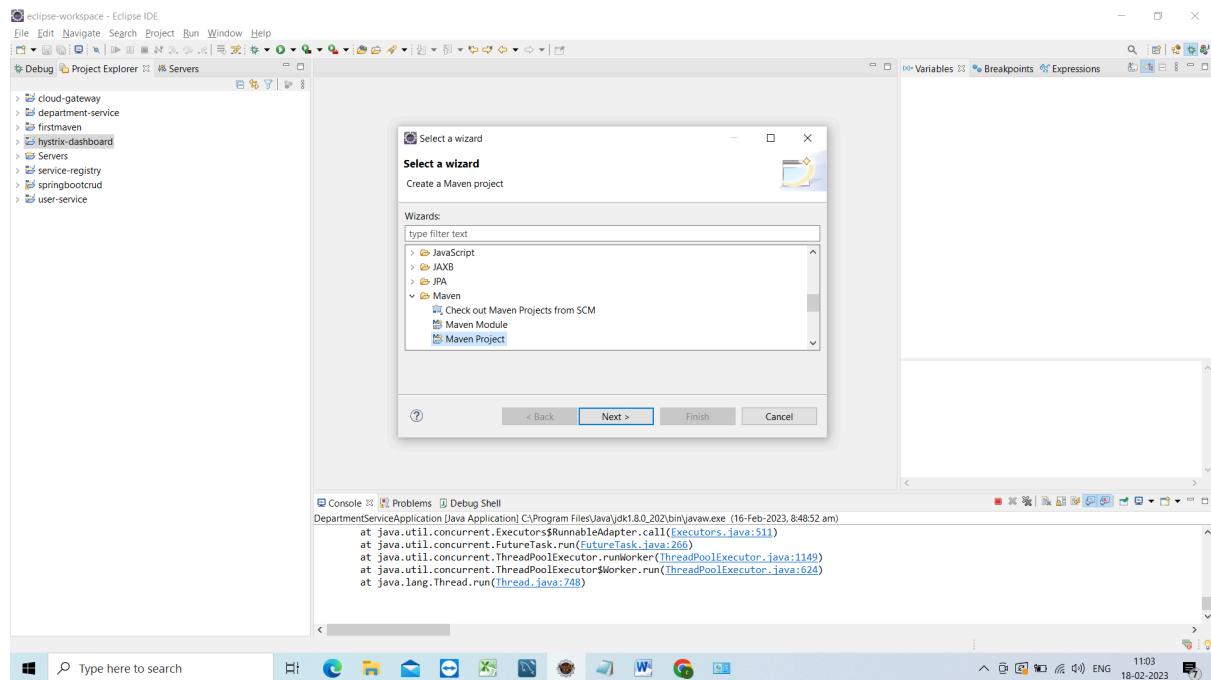
[View Logos →](#)

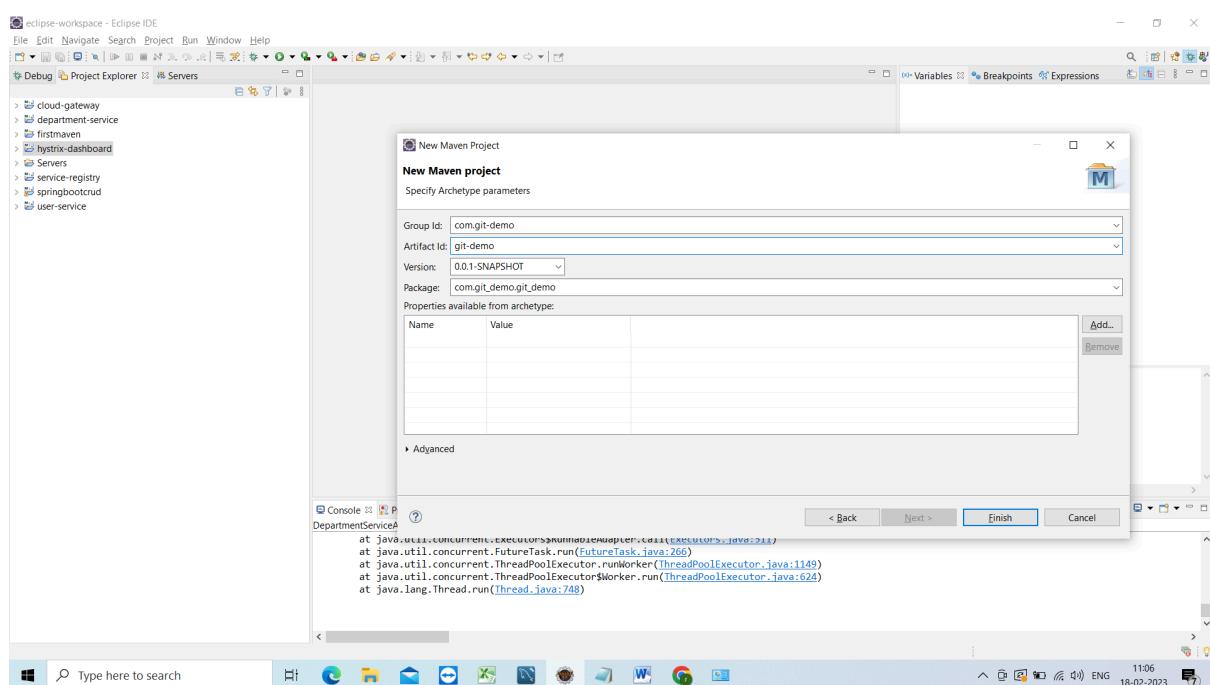
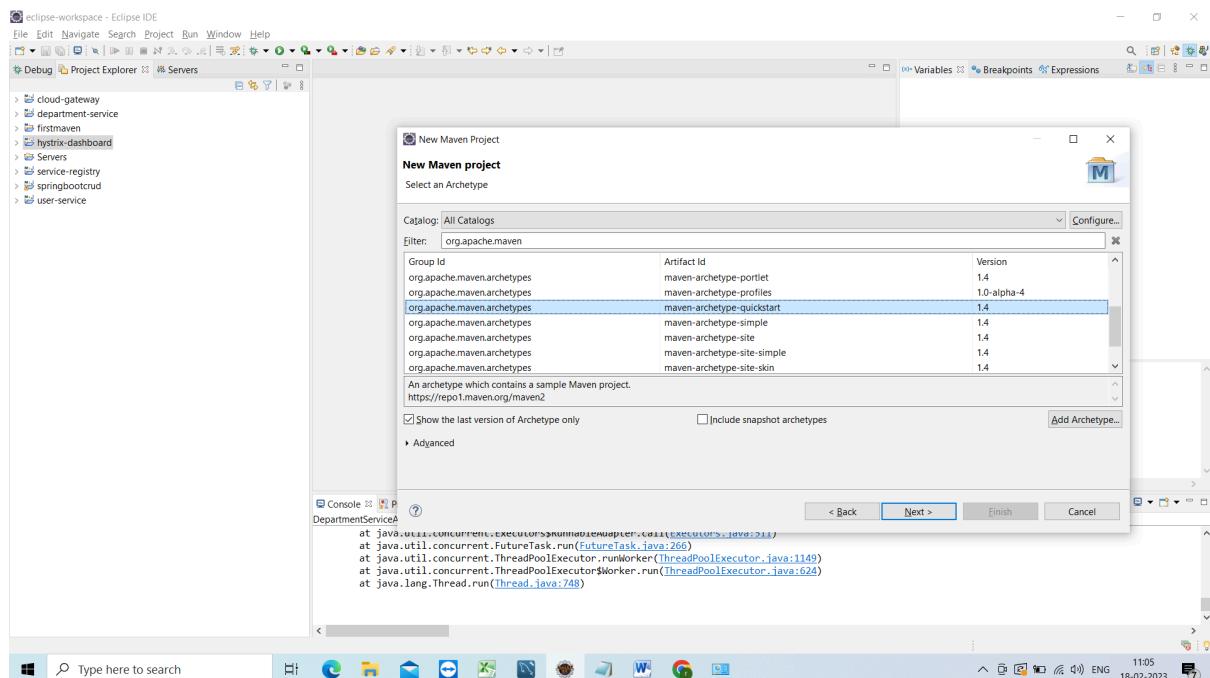
Git via Git

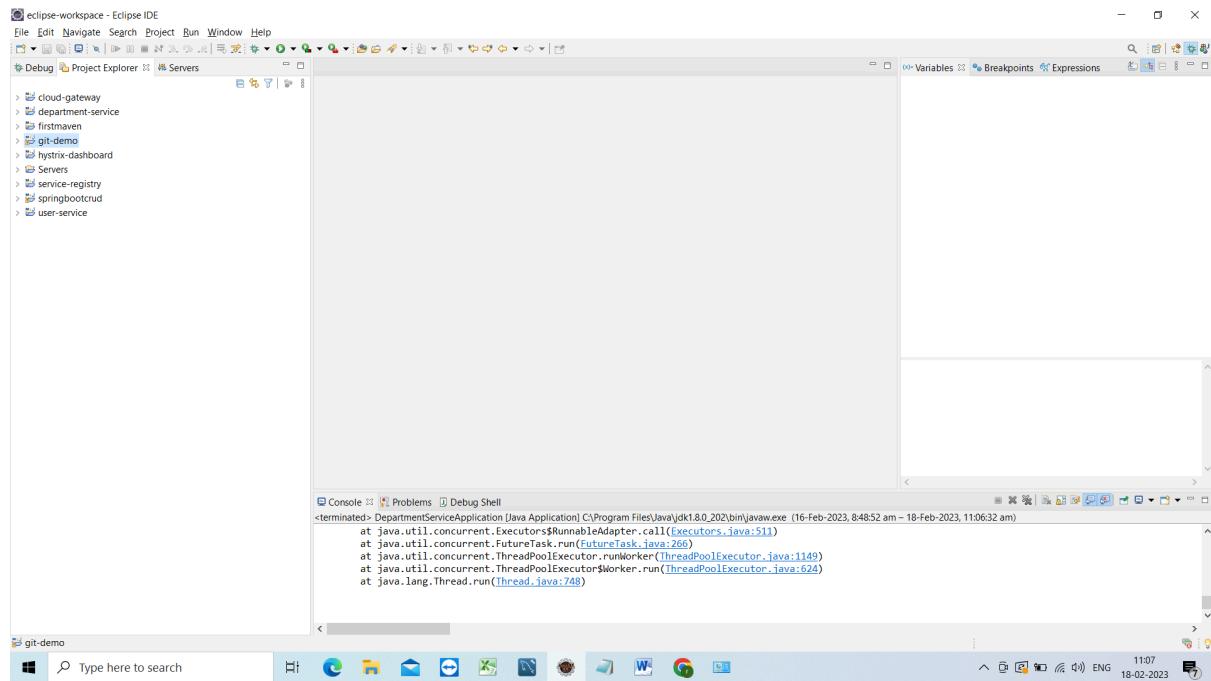
If you already have Git installed, you can get the latest development version via Git itself:

Click on download2.18.0 for windows

Lets create the maven project in eclipse



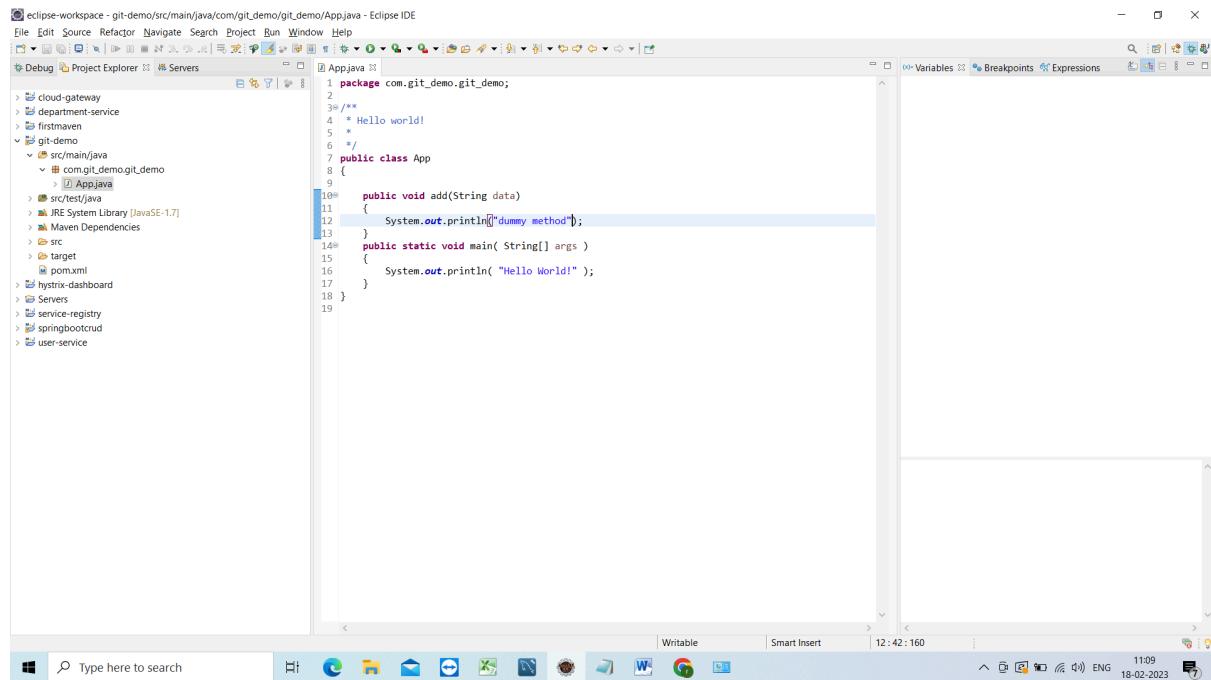




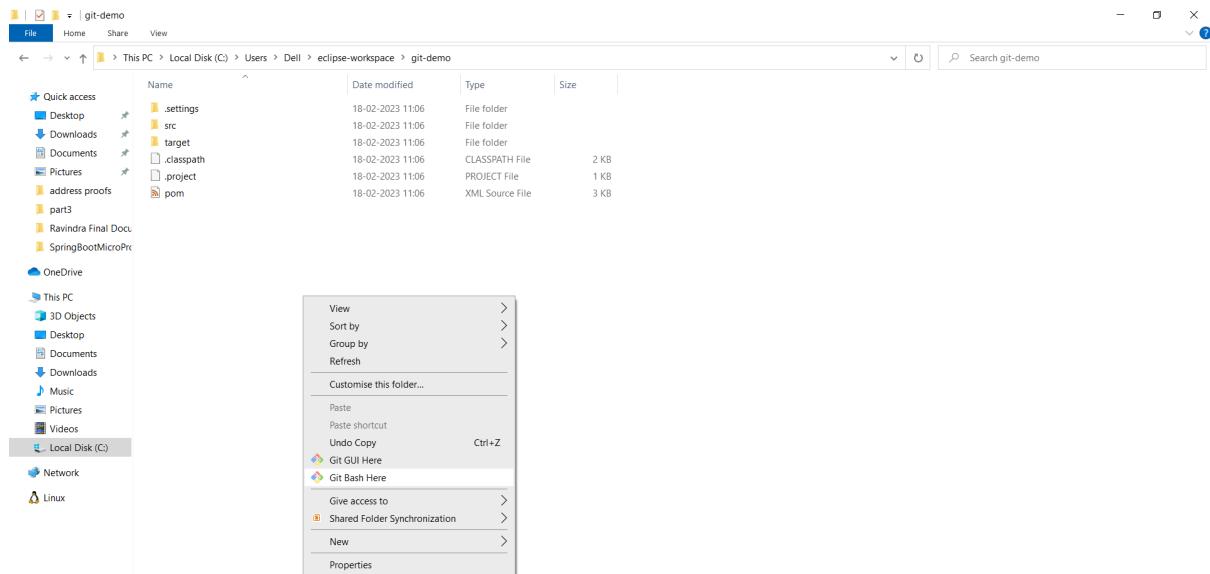
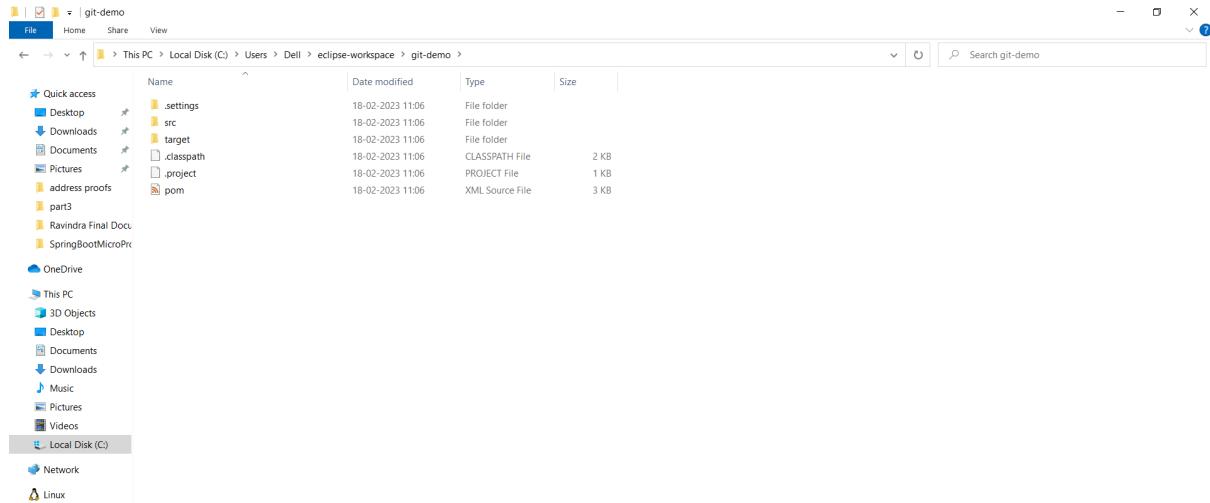
Project has been created.

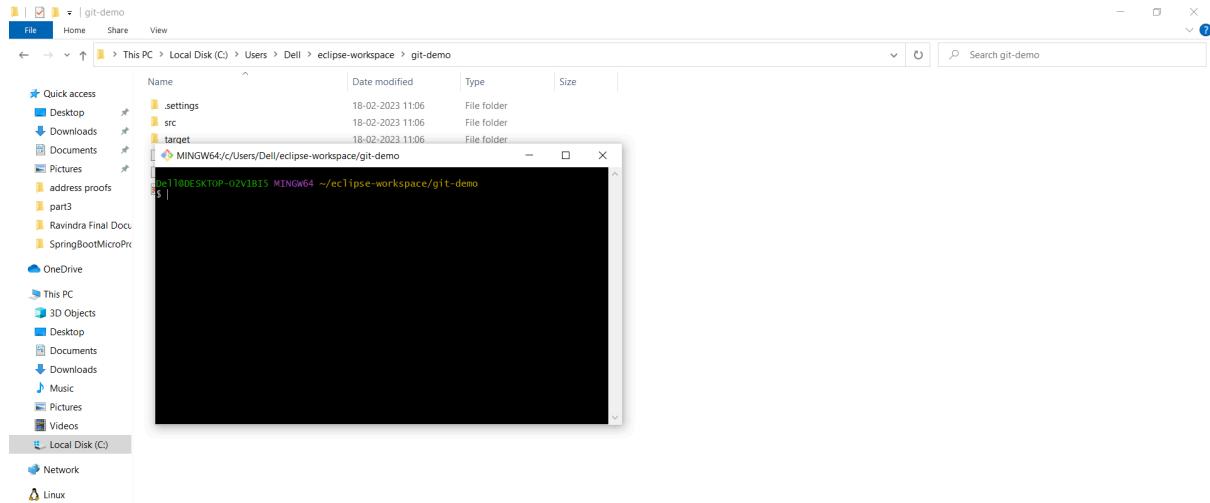
Create some dummy method

In APP.java



Now go to the project directory and do the git bash here

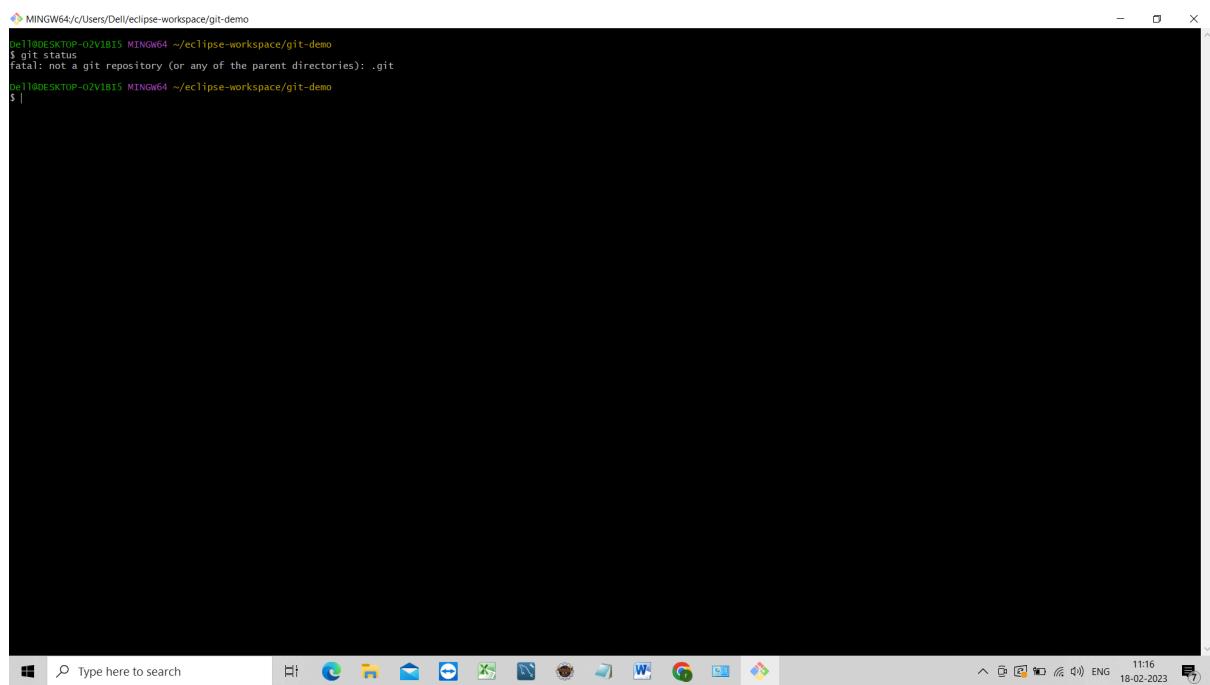




Now check any local repository created or not

For that need to use

Git status



Not created

To initialize the local repository we need to use command called

Git init

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git status
fatal: not a git repository (or any of the parent directories): .git
$ git init
Initialized empty Git repository in C:/Users/Dell/eclipse-workspace/git-demo/.git/
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .classpath
    .project
    settings/
    pom.xml
    src/
    target/
nothing added to commit but untracked files present (use "git add" to track)
$
```

Now check changes in the code

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git status
fatal: not a git repository (or any of the parent directories): .git
$ git init
Initialized empty Git repository in C:/Users/Dell/eclipse-workspace/git-demo/.git/
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .classpath
    .project
    settings/
    pom.xml
    src/
    target/
nothing added to commit but untracked files present (use "git add" to track)
$
```

Now we need to add those changes to staging area

Git add .

It will add all the files to the staging area

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git status
fatal: not a git repository (or any of the parent directories): .git
$ git init
Initialized empty Git repository in C:/Users/Dell/eclipse-workspace/git-demo/.git/
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .classpath
    .project
    settings/
    pom.xml
    src/
    target/

nothing added to commit but untracked files present (use "git add" to track)

$ git add .

$ git status
On branch master

$ |
```

Now check the git status

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git status
fatal: not a git repository (or any of the parent directories): .git
$ git init
Initialized empty Git repository in C:/Users/Dell/eclipse-workspace/git-demo/.git/
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .classpath
    .project
    settings/
    pom.xml
    src/
    target/

nothing added to commit but untracked files present (use "git add" to track)

$ git add .

$ git status
On branch master

$ |
```

Changes to be committed:

```
(use "git rm --cached <file>..." to unstage)
  new file:   .classpath
  new file:   .project
  new file:   .settings/org.eclipse.core.resourcesprefs
  new file:   .settings/org.eclipse.jdt.core.preferences
  new file:   .settings/org.eclipse.m2e.core.preferences
  new file:   pom.xml
  new file:   src/main/java/com/git_demo/git_demo/app.java
  new file:   src/main/java/com/git_demo/git_demo/appTest.java
  new file:   target/classes/META-INF/MANIFEST.MF
  new file:   target/classes/META-INF/maven/com.git_demo.git_demo/pom.properties
  new file:   target/classes/META-INF/maven/com.git_demo.git_demo/pom.xml
  new file:   target/classes/com/git_demo/git_demo/App.class
  new file:   target/test-classes/com/git_demo/git_demo/AppTest.class
```

```
$ |
```

Now add the all the files to the local repository for that need to use command called

Git commit –m “commit message any thing”

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git commit -m "First commit"
[master (root-commit) 14bc5cb] First commit
 13 files changed, 266 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/org.eclipse.core.resources_prefs
 create mode 100644 .settings/org.eclipse.jdt.core_prefs
 create mode 100644 .settings/org.eclipse.m2e.core_prefs
 create mode 100644 pom.xml
 create mode 100644 src/test/java/com/git_demo/git_demo/app.java
 create mode 100644 src/test/java/com/git_demo/git_demo/AppTest.java
 create mode 100644 target/classes/META-INF/MANIFEST.MF
 create mode 100644 target/classes/META-INF/maven/com.git_demo.git_demo/pom.properties
 create mode 100644 target/classes/META-INF/maven/com.git_demo.git_demo/pom.xml
 create mode 100644 target/classes/com.git_demo.git_demo/App.class
 create mode 100644 target/test-classes/com.git_demo.git_demo/AppTest.class
$
```

Now check the git status

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git commit -m "First commit"
[master (root-commit) 14bc5cb] First commit
 13 files changed, 266 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/org.eclipse.core.resources_prefs
 create mode 100644 .settings/org.eclipse.jdt.core_prefs
 create mode 100644 .settings/org.eclipse.m2e.core_prefs
 create mode 100644 pom.xml
 create mode 100644 src/test/java/com/git_demo/git_demo/app.java
 create mode 100644 src/test/java/com/git_demo/git_demo/AppTest.java
 create mode 100644 target/classes/META-INF/MANIFEST.MF
 create mode 100644 target/classes/META-INF/maven/com.git_demo.git_demo/pom.properties
 create mode 100644 target/classes/META-INF/maven/com.git_demo.git_demo/pom.xml
 create mode 100644 target/classes/com.git_demo.git_demo/App.class
 create mode 100644 target/test-classes/com.git_demo.git_demo/AppTest.class
$ git status
On branch master
nothing to commit, working tree clean
$
```

Now check git log

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
$ git commit -m "First commit"
[master (root-commit) 14bc8bc] First commit
 13 files changed, 266 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/org.eclipse.core.resources_prefs
 create mode 100644 .settings/org.eclipse.jdt.core_prefs
 create mode 100644 .settings/org.eclipse.m2e.core_prefs
 create mode 100644 pom.xml
 create mode 100644 src/main/java/com/git_demo/git_demo/app.java
 create mode 100644 src/test/java/com/git_demo/git_demo/AppTest.java
 create mode 100644 target/classes/META-INF/MANIFEST.MF
 create mode 100644 target/classes/META-INF/maven/com.git_demo.git_demo/pom.properties
 create mode 100644 target/classes/com.git_demo.git_demo/pom.xml
 create mode 100644 target/classes/com/git_demo/git_demo/App.class
 create mode 100644 target/test-classes/com/git_demo/git_demo/AppTest.class

DeSKTOP-02VIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
$ git status
On branch master
nothing to commit, working tree clean

DeSKTOP-02VIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
$ git log
commit 14bc8bc013fe039c87285ea04fc79a13148b (HEAD -> master)
Author: Simplilearn GitHub <siddam.bharat@simplilearn.net>
Date:   Sat Feb 18 11:24:17 2023 +0530

  first commit

DeSKTOP-02VIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
$ |
```

Checks has been created.

Now go to code do some changes

Now go to code do some changes

And again check git status

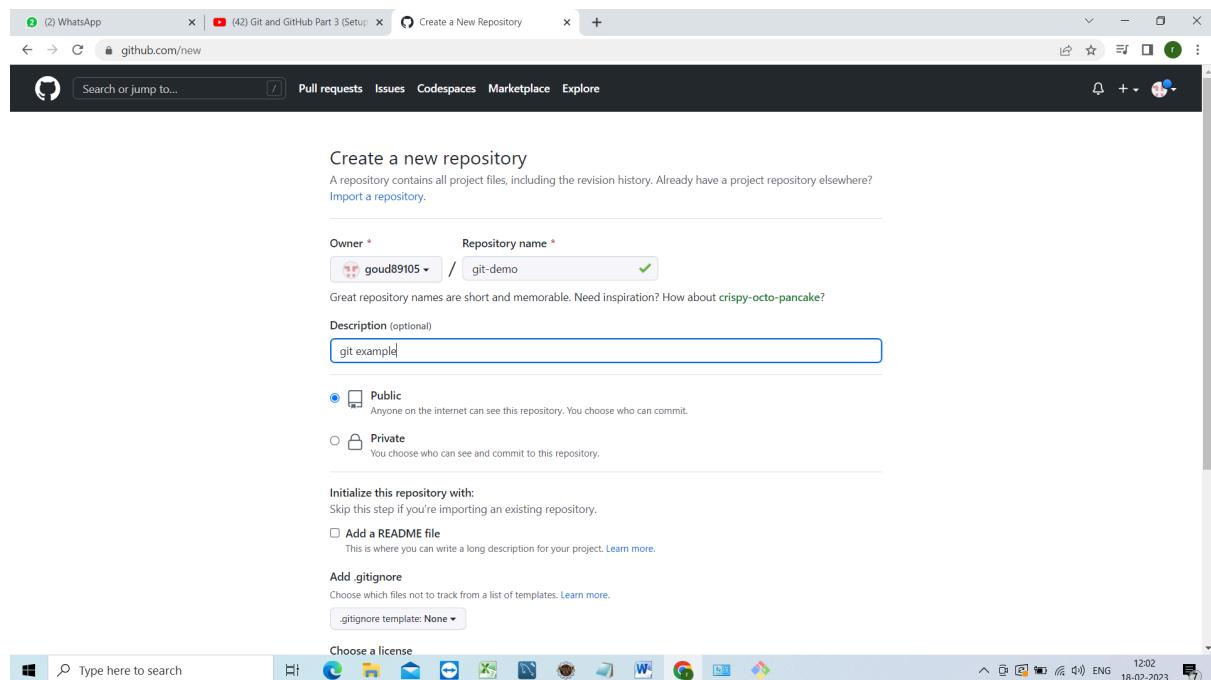
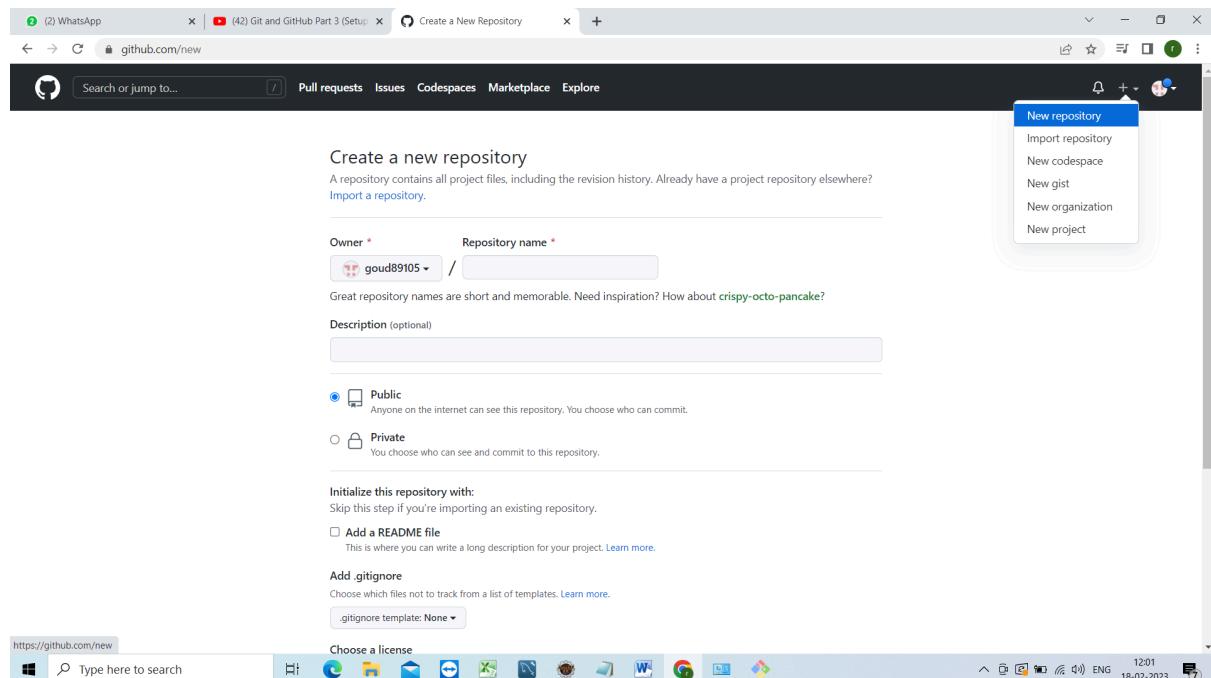
Use git add .

Git commit –m “message”

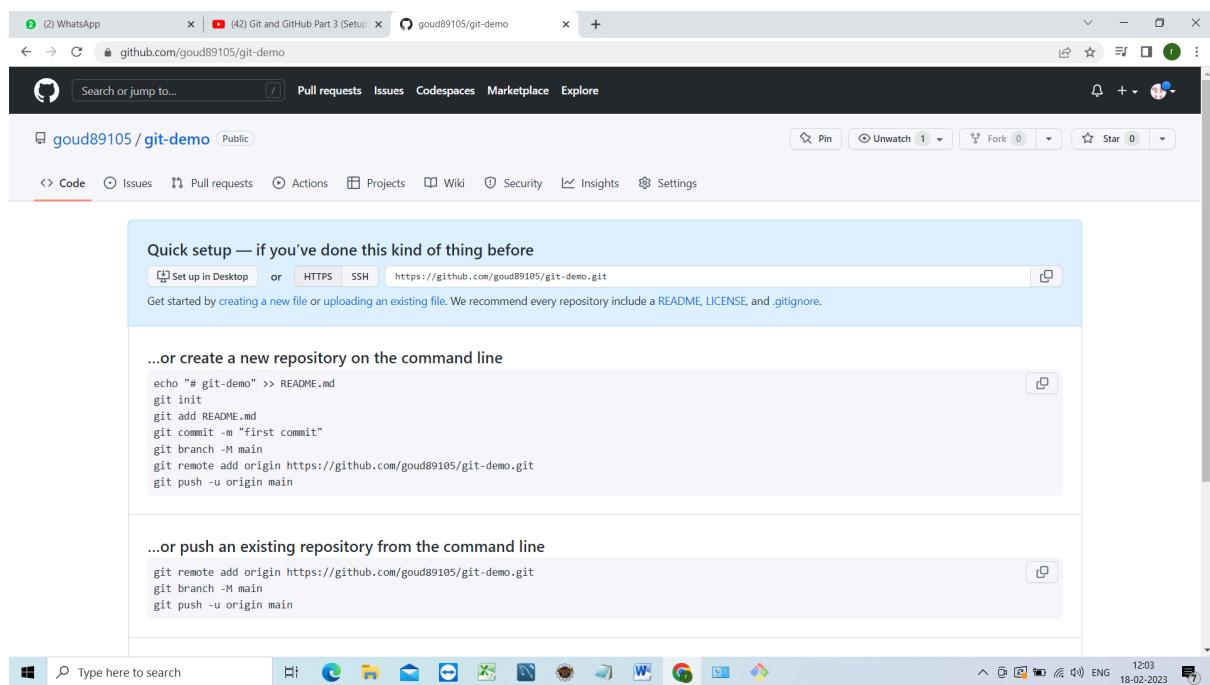
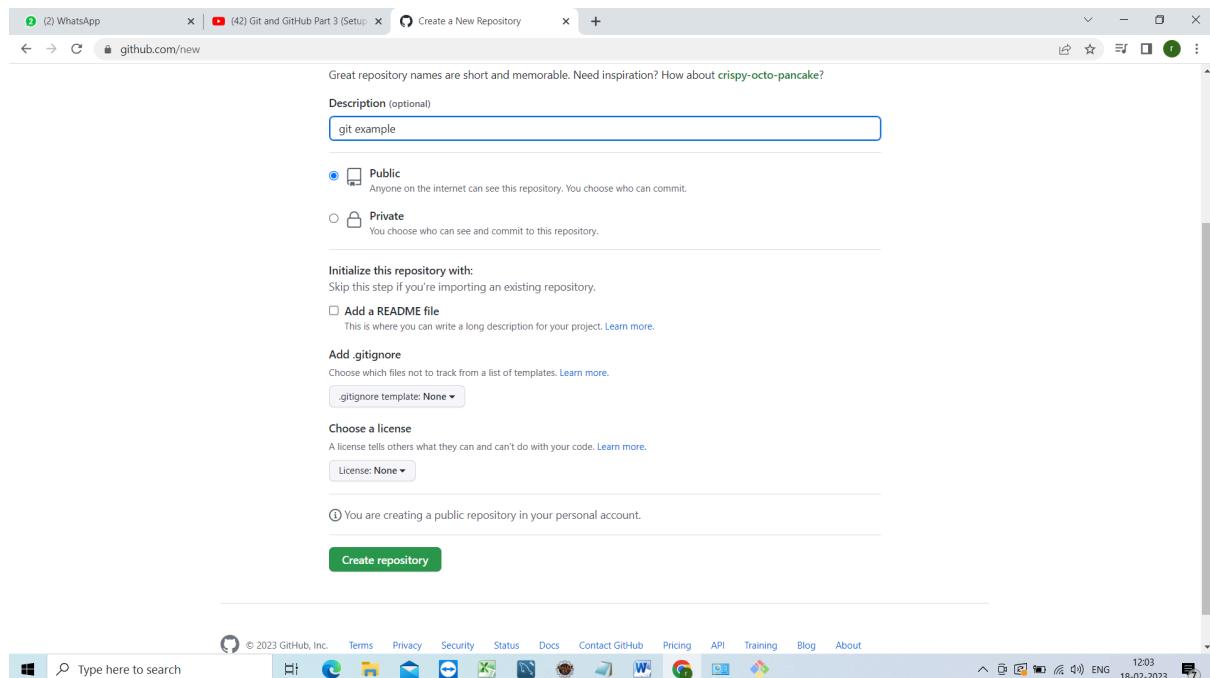
Next am going show u how to push local repository changes to remote repository.

For that u have to create the remote repository in github.

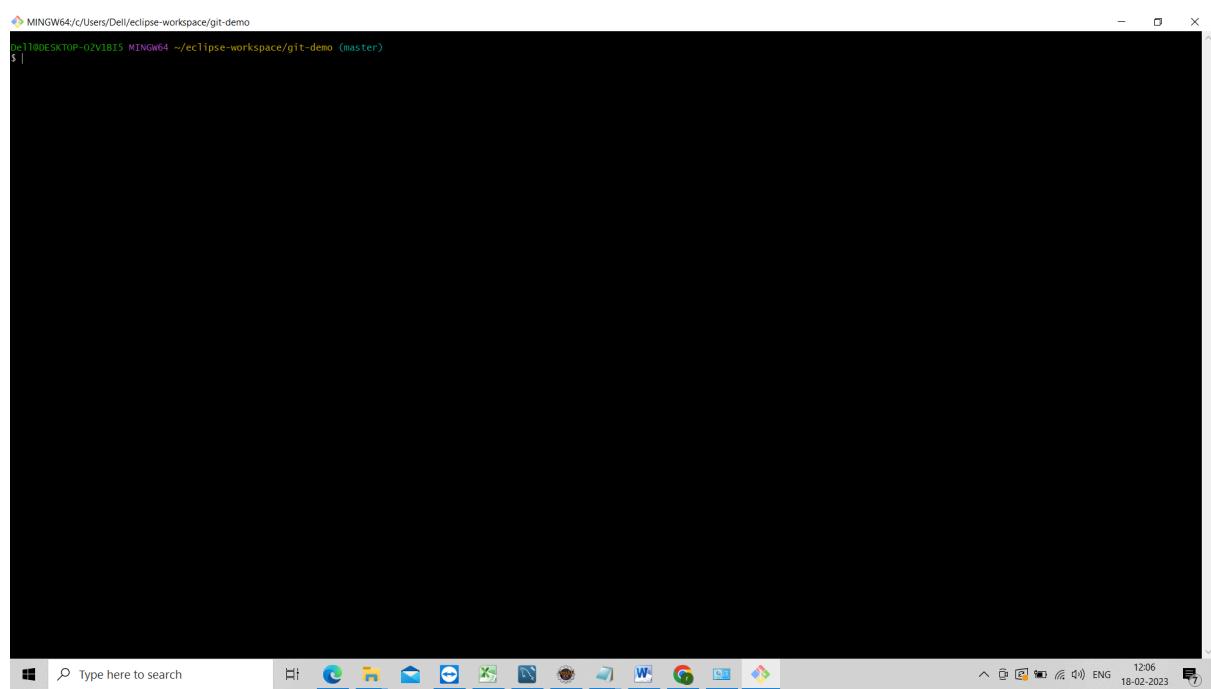
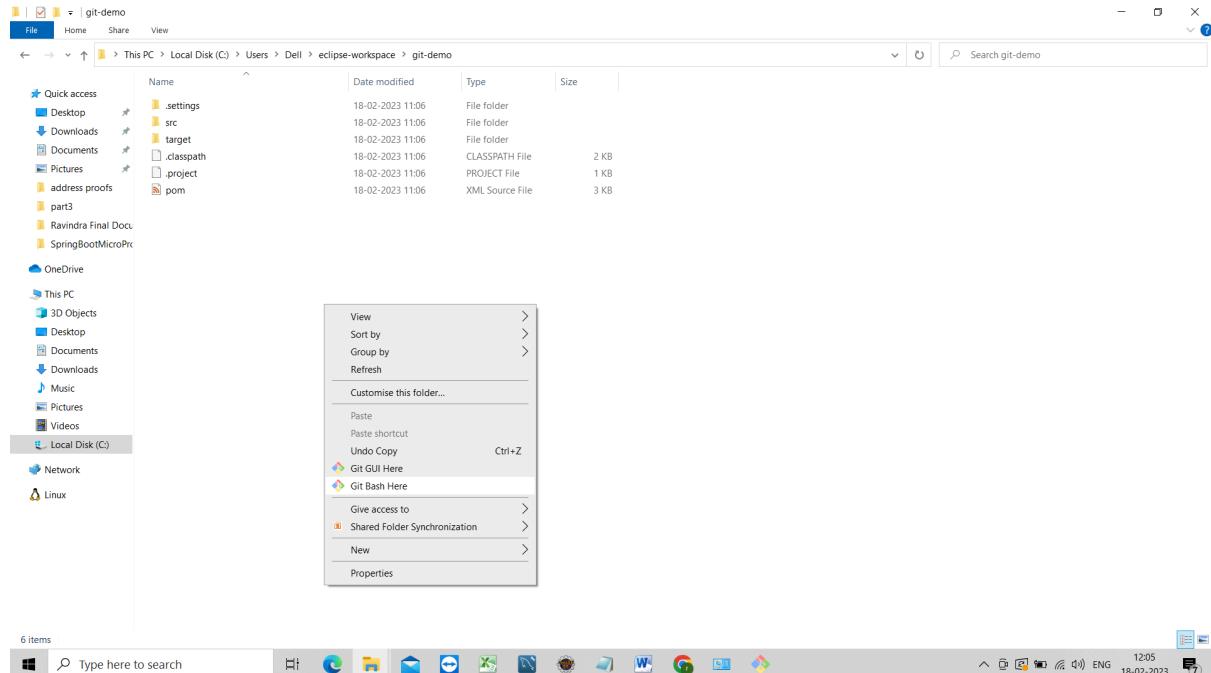
Click on + and select new repository



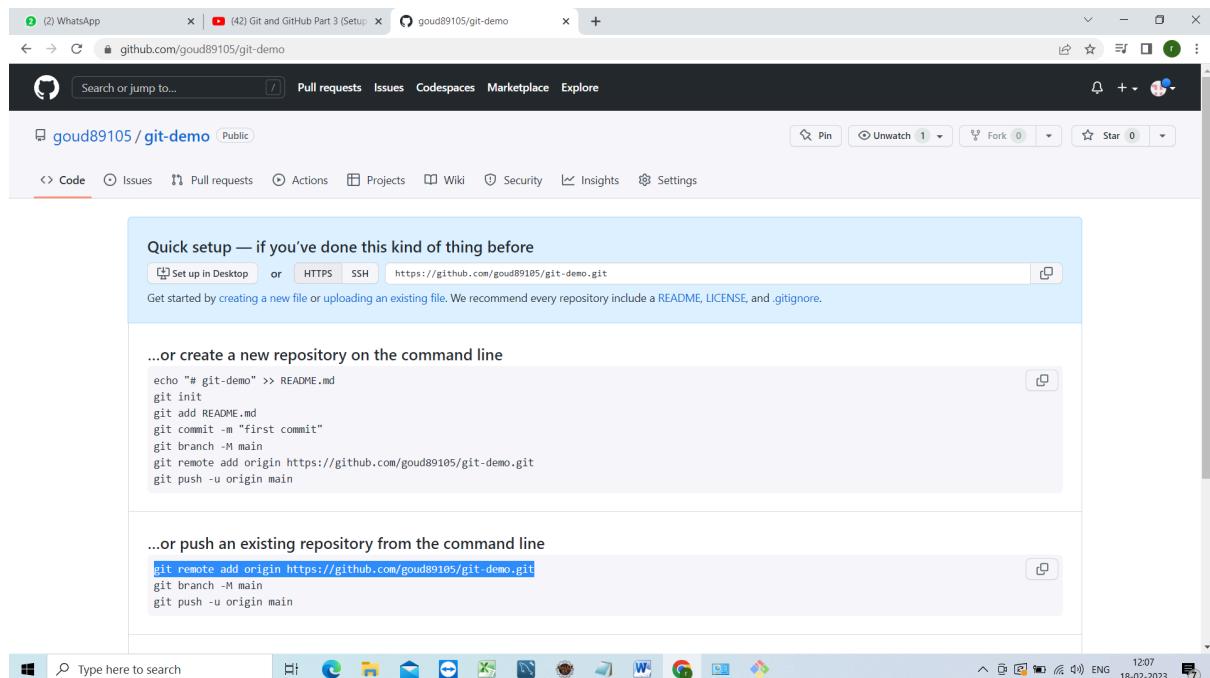
Click on create repository.



Now go to project directory which u have created previously do git bash from there.



We need to set from local repository to remote repository to set that use below command



git remote add origin <https://github.com/goud89105/git-demo.git>

A screenshot of a Windows terminal window titled 'MINGW64/c/Users/Dell/eclipse-workspace/git-demo'. The command 'git remote add origin https://github.com/goud89105/git-demo.git' is being typed into the terminal. The terminal window is located on a desktop with a taskbar at the bottom containing various icons.

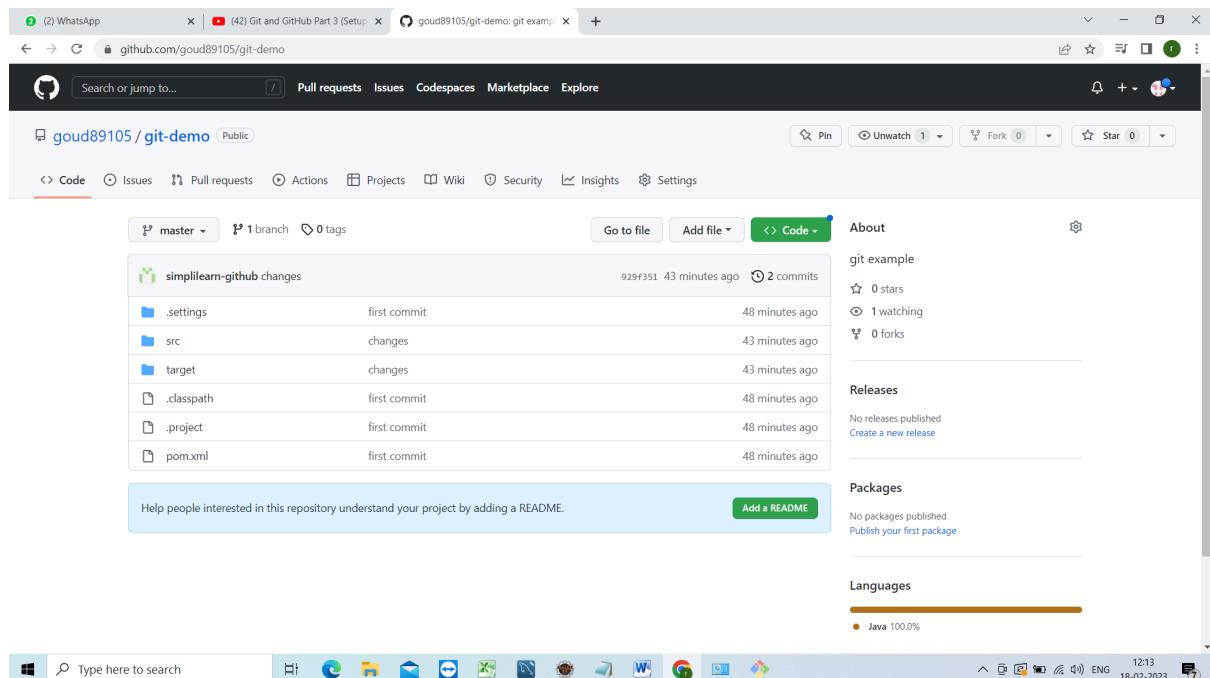
And enter

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo
DellDESKTOP-0ZVIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
$ git remote add origin https://github.com/goud89105/git-demo.git
DellDESKTOP-0ZVIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
```

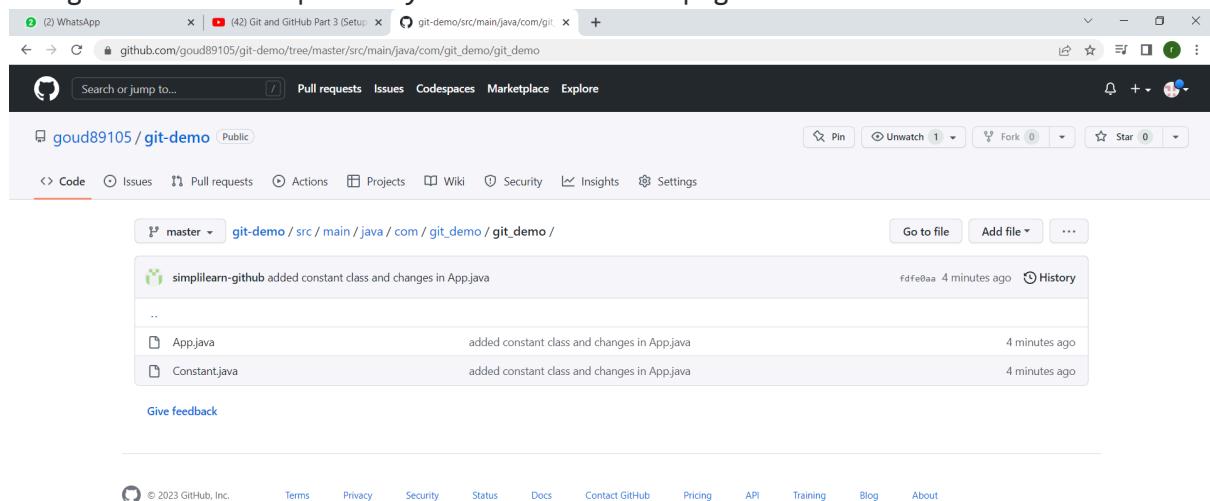
Now push our changes to remote repository
Use below command
Git push -u origin master

```
MINGW64/c/Users/Dell/eclipse-workspace/git-demo (master)
DellDESKTOP-0ZVIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
$ git remote add origin https://github.com/goud89105/git-demo.git
DellDESKTOP-0ZVIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
$ git push -u origin master
Enumerating objects: 54, done.
Counting objects: 100% (54/54), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (27/27), done.
Writing objects: 100% (54/54), 5.69 KiB | 647.00 KiB/s, done.
Total 54 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (27/27), done.
To https://github.com/goud89105/git-demo.git
 * [new branch] master -> master
branch 'master' set up to track 'origin/master'.
DellDESKTOP-0ZVIBIS MINGW64 ~/eclipse-workspace/git-demo (master)
```

Now our local changes push to git hub
Go to github and refresh the page

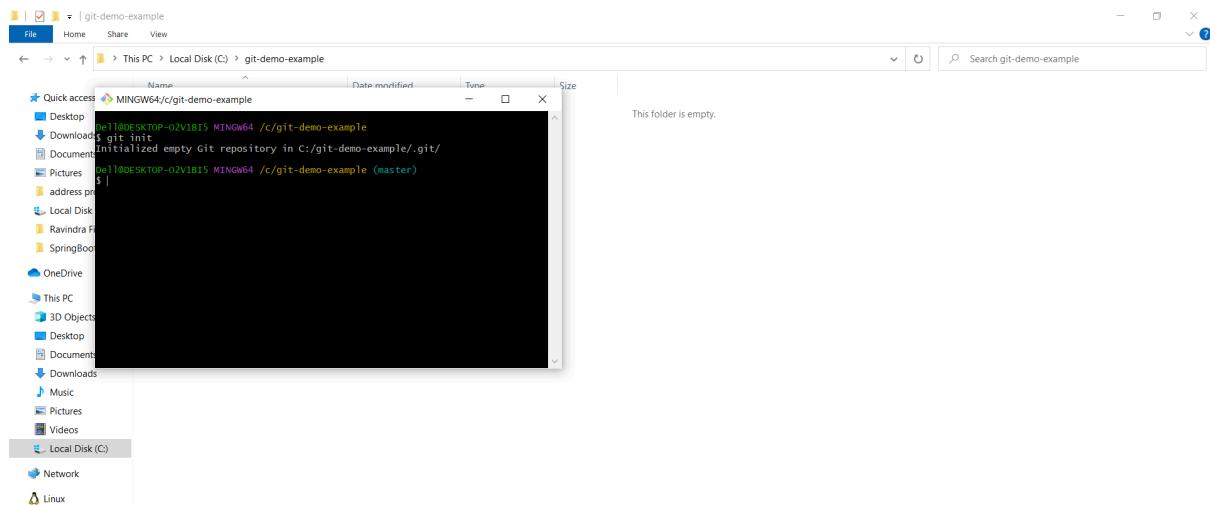
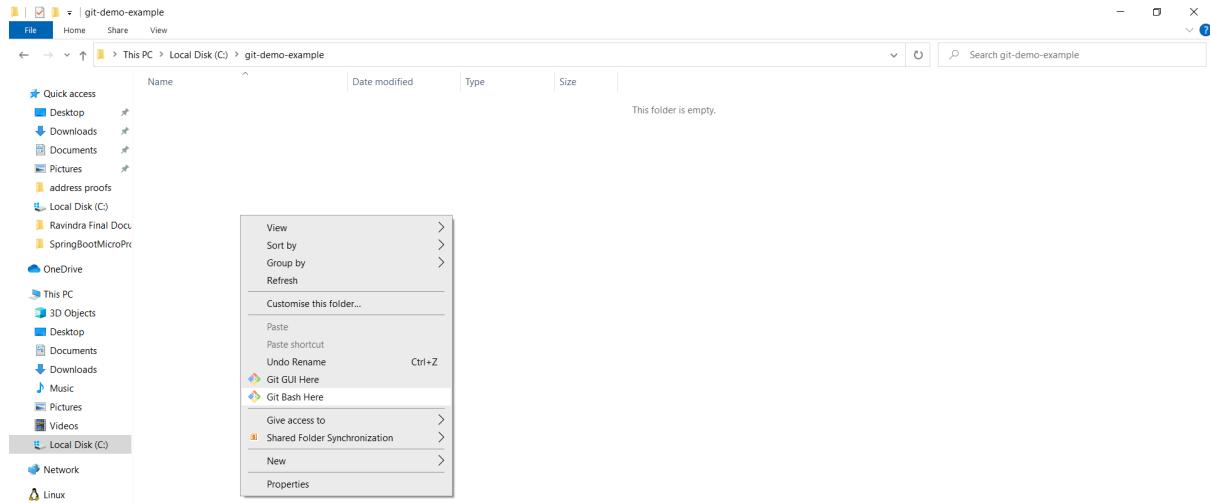


Now go to remote repository and refresh the page

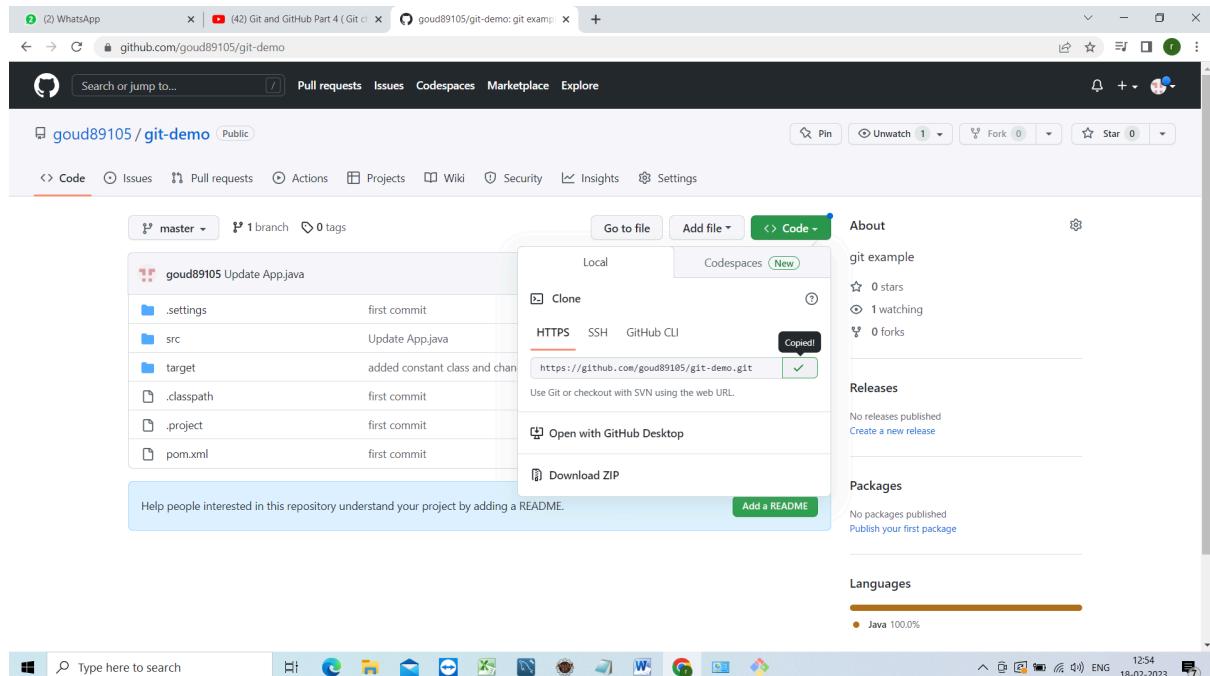


Now am going to show u how to checkout the code and how to update the code

Assume u have joined the project newly they have given git link I will show how to checkout the code from github
Frist go to the any drive and create the folder and open the git bash from that folder and do git init



And now go to git hub and copy the link from below https



Now go to git bash execute the below command
Git clone URL

<https://github.com/goud89105/git-demo.git>

```
MINGW64 /c/git-demo-example
$ git init
Initialized empty Git repository in c:/git-demo-example/.git/
MINGW64 /c/git-demo-example (master)
$ git clone https://github.com/goud89105/git-demo.git
Cloning into 'git-demo'...
remote: Enumerating objects: 80, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (37/37), done.
remote: Total 80 (delta 8), reused 69 (delta 6), pack-reused 0
Receiving objects: 100% (80/80), 8.30 KiB | 1.04 MiB/s, done.
Resolving deltas: 100% (8/8), done.
MINGW64 /c/git-demo-example (master)
$
```

Create the new branch

Git branch name of the branch

```
MINGW64/c/git-demo-example
$ git branch
* master
$ git branch phase2
$
```

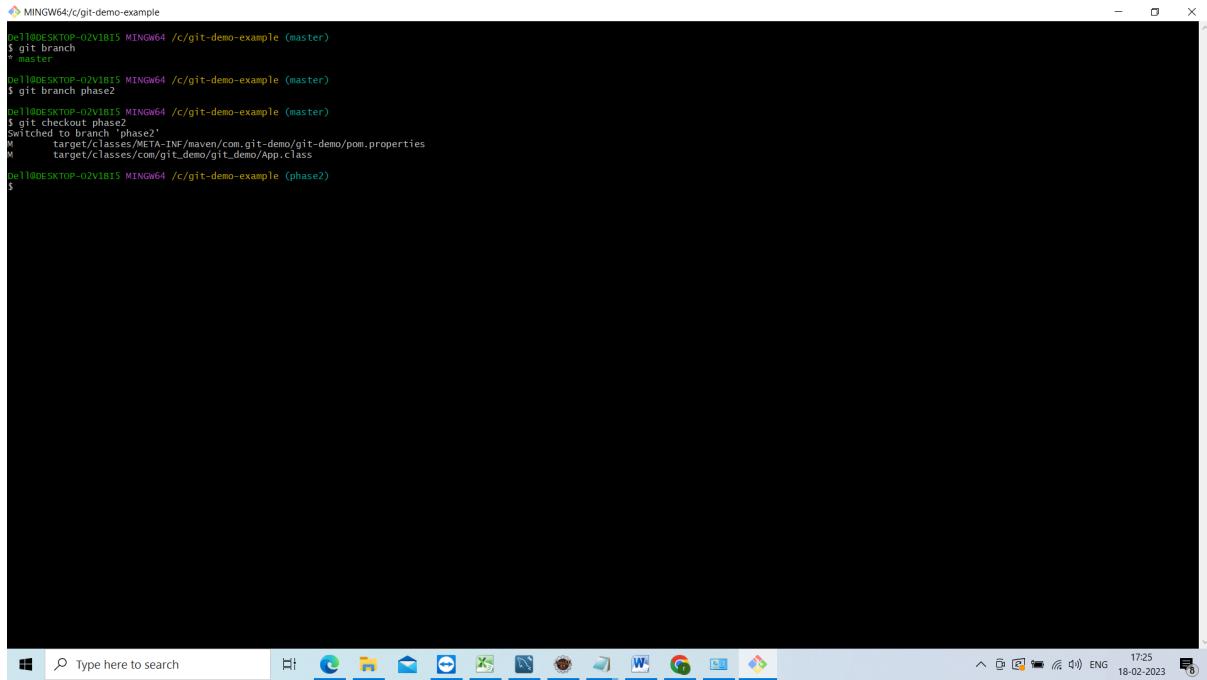


Now lets move to the

Git checkout phase2

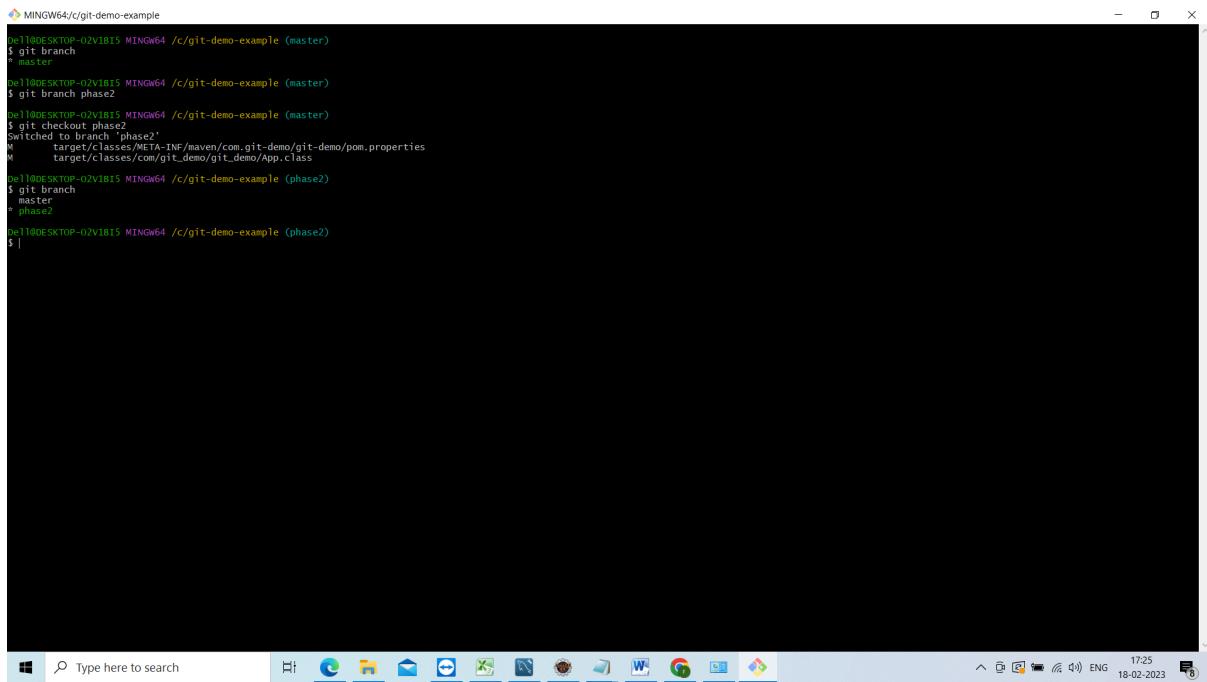
```
MINGW64/c/git-demo-example
$ git branch
* master
$ git branch phase2
$ git checkout phase2
```





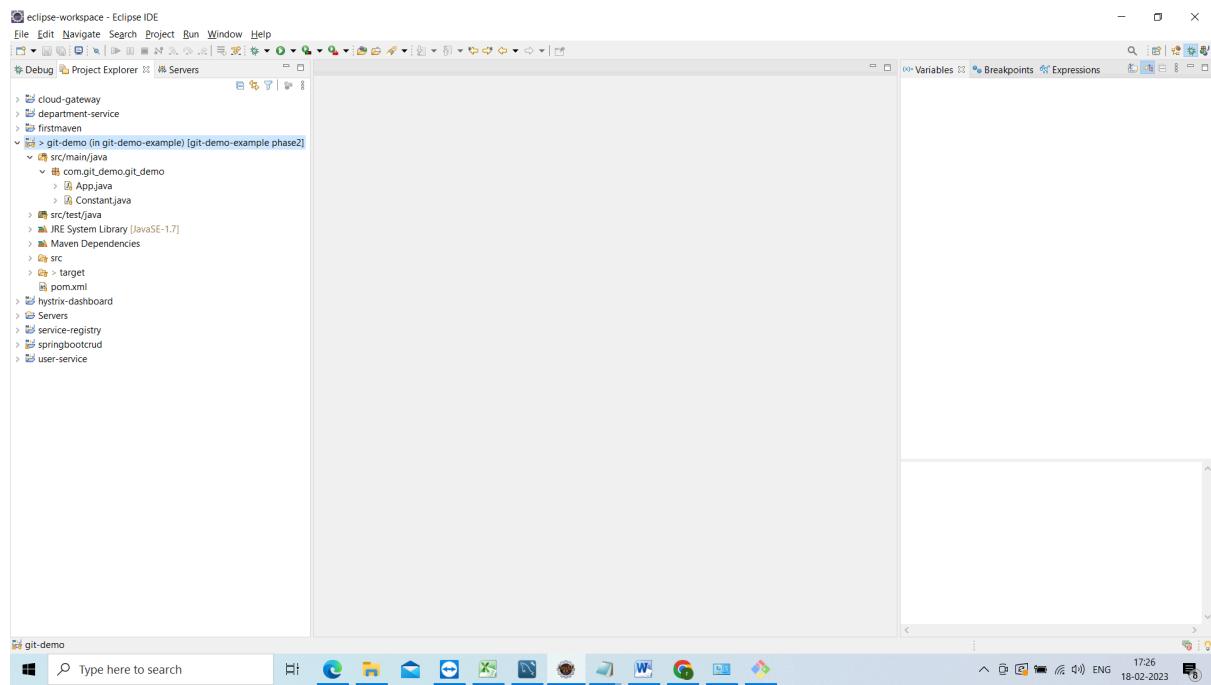
```
MINGW64/c/git-demo-example
$ git branch
* master
$ git checkout phase2
Switched to branch 'phase2'
$ git status
  # On branch phase2
  # Your branch is up-to-date with 'origin/phase2'.
  #
  nothing to commit, working tree clean
$ git branch phase2
$
```

Now check the branch

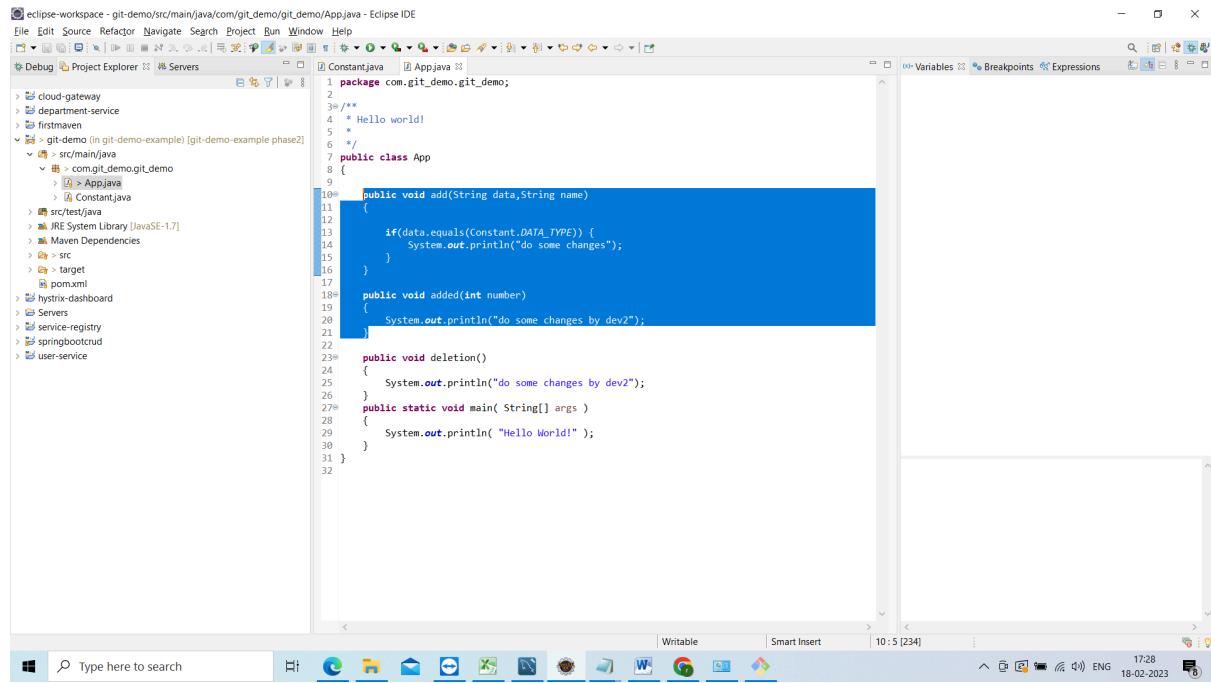


```
MINGW64/c/git-demo-example
$ git branch
* master
$ git checkout phase2
Switched to branch 'phase2'
$ git status
  # On branch phase2
  # Your branch is up-to-date with 'origin/phase2'.
  #
  nothing to commit, working tree clean
$ git branch phase2
$
```

Now go to eclipse see now we are in phase2 branch



Now do some changes those changes will be in phase2 branch



Now these changes not available in master branch

Go to git bash and see the changes in phase2

```
MINGW64/c/git-demo-example
$ git branch
* master
$ git checkout phase2
Switched to branch 'phase2'
$ git status
On branch phase2
Changes not staged for commit:
  (use "git add <file>" to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified: src/main/java/com/git_demo/git_demo/App.java
      modified: target/classes/META-INF/maven/com.git_demo/git_demo/pom.properties
      modified: target/classes/com/git_demo/git_demo/App.class

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

no changes added to commit (use "git add" and/or "git commit -a")
$
```

Now add it and commit

Git add .

Git commit -m "changes added"

```
MINGW64/c/git-demo-example
$ git branch
* master
$ git branch phase2
Switched to branch 'phase2'
$ git status
On branch phase2
Changes not staged for commit:
  (use "git add <file>" to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified: src/main/java/com/git_demo/git_demo/App.java
      modified: target/classes/META-INF/maven/com.git_demo/git_demo/pom.properties
      modified: target/classes/com/git_demo/git_demo/App.class

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

no changes added to commit (use "git add" and/or "git commit -a")
$ git add .
warning: LF will be replaced by CRLF in .gitignore.
The working copy will be replaced by the next time Git touches it.
$ git commit -m "changes added"
[phase2 22d638c] changes added
 4 files changed, 5 insertions(+), 4 deletions(-)
 create mode 100644 .gitignore
$
```

Now committed code in staging not in master and not in phase2

Now commit these changes to phase2

Git push origin phase2

```
MINGW64 /c/git-demo-example
$ git branch phase2
Switched to branch 'phase2'
$ git checkout phase2
Switched to branch 'phase2'
$ target/classes/META-INF/maven/com.git_demo/git-demo/pom.properties
$ target/classes/com.git_demo/App.class
$ git branch
* master
* phase2
$ git status
On branch phase2
Changes not staged for commit:
  (use "git add <file>" to update what will be committed)
    (use "git reset <file>" to discard changes in working directory)
      modified: src/main/java/com/git_demo/App.java
      modified: target/classes/META-INF/maven/com.git_demo/git-demo/pom.properties
      modified: target/classes/com.git_demo/App.class
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

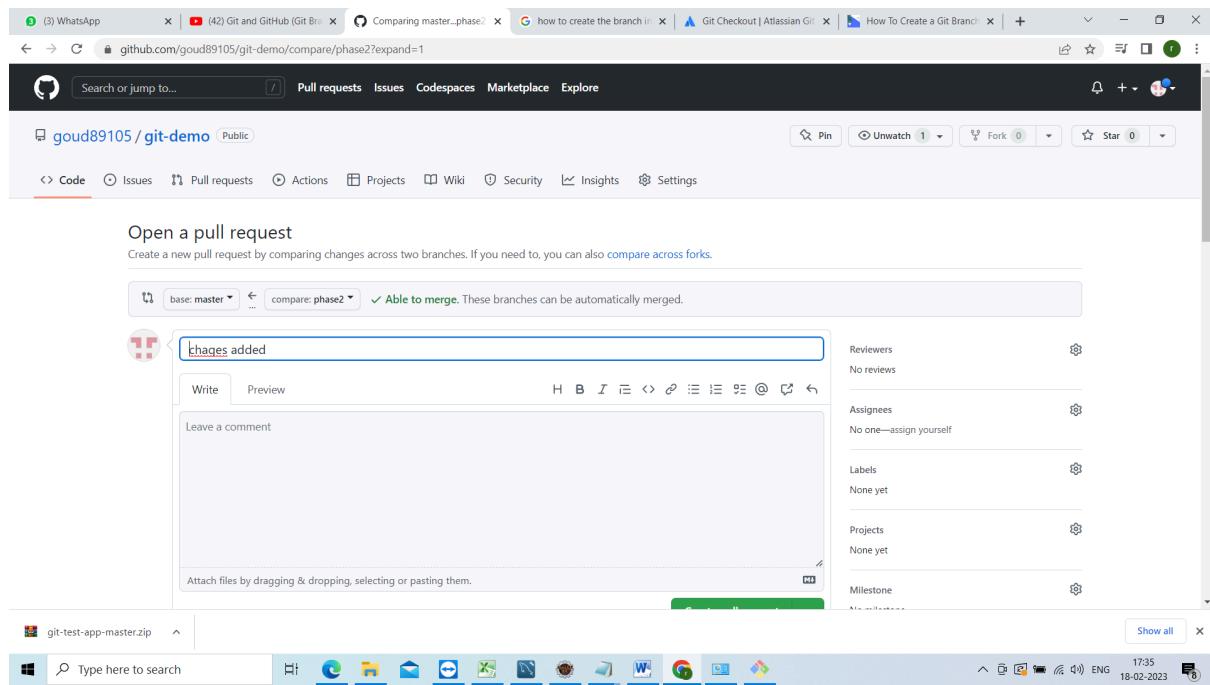
no changes added to commit (use "git add" and/or "git commit -a")

$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time git touches it
$ git commit -m "changes made"
[phase2 22d638c] changes added
 4 files changed, 5 insertions(+), 4 deletions(-)
 create mode 100644 .gitignore
$ git push origin phase2
Enumerating objects: 30, done.
Counting objects: 30 (local), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (21/21), 1.68 KiB | 573.00 KiB/s, done.
To https://github.com/goud89105/git-demo.git
 * [new branch] phase2 -> phase2
$
```

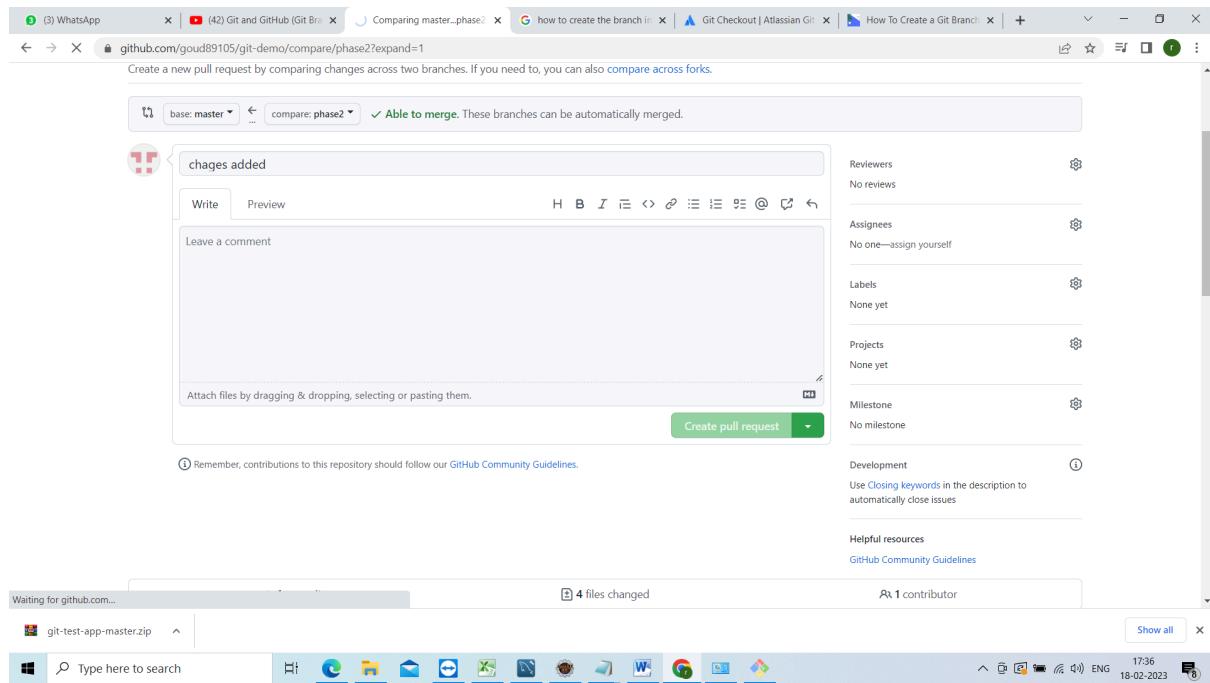
Now in git hub it will show the new branch phase2 go to git hub and see

The screenshot shows a GitHub repository page for 'goud89105 / git-demo'. The repository has 1 branch named 'phase2'. The 'About' section shows the repository is public and has 0 stars, 1 watching, and 0 forks. The 'Languages' section shows Java at 100.0%. The 'Code' tab is selected, displaying a list of commits from 'goud89105 developer2 changes' made 4 hours ago, including changes to .settings, src, target, .classpath, .project, and pom.xml.

Now compare and create the pull request



Now click on pull request



changes added #1

1 Open goud89105 wants to merge 1 commit into master from phase2

Conversation 0 Commits 1 Checks 0 Files changed 4

goud89105 commented now
No description provided.

22d638c

Add more commits by pushing to the phase2 branch on goud89105/git-demo.

Require approval from specific reviewers before merging
Branch protection rules ensure specific people approve pull requests before they're merged. Add rule

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.

git-test-app-master.zip

Now see this branch has no conflicts with the base branch

goud89105 commented 1 minute ago
No description provided.

22d638c

Add more commits by pushing to the phase2 branch on goud89105/git-demo.

Require approval from specific reviewers before merging
Branch protection rules ensure specific people approve pull requests before they're merged. Add rule

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.

Merge pull request

git-test-app-master.zip

Before that lets check the changes click on files chaged

changes added #1

goud89105 wants to merge 1 commit into `master` from `phase2`

Conversation 0 Commits 1 Checks 0 Files changed 4

goud89105 commented 2 minutes ago
No description provided.

chages added 22d638c

Add more commits by pushing to the `phase2` branch on `goud89105/git-demo`.

Require approval from specific reviewers before merging
Branch protection rules ensure specific people approve pull requests before they're merged. Add rule

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

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

Assignees: No one—assign yourself.

Labels: None yet.

Projects: None yet.

<https://github.com/goud89105/git-demo/pull/1/files>

git-test-app-master.zip

Type here to search

18-02-2023 17:38

changes added #1

Changes from all commits ▾ Conversations ▾

0 / 4 files viewed Review changes

Filter changed files

.gitignore

src/main/java/com/git_demo/git_d...
App.java

target/classes

META-INF/maven/com.git-demo/...
pom.properties

com/git_demo/git_demo
App.class

File filter ▾

Viewed ▾

src/main/java/com/git_demo/App.java

@@ -7,15 +7,15 @@

7 7 public class App

8 8 {

9 9

10 10 - public void add(String data)

11 11 + public void add(String data, String name)

12 12

13 13 if(data.equals(Constant.DATA_TYPE)) {

14 14 System.out.println("do some changes");

15 15 }

16 16 }

17 17

18 18 - public void added()

19 19 + public void added(int number)

20 20 System.out.println("do some changes by dev2");

21 21 }

... ... @@ -1,6 +1,6 @@

1 1 #Generated by Maven Integration for Eclipse

Viewed ▾

target/classes/META-INF/maven/com.git-demo/git-demo/pom.properties

... ... @@ -1,6 +1,6 @@

1 1 #Generated by Maven Integration for Eclipse

Show all

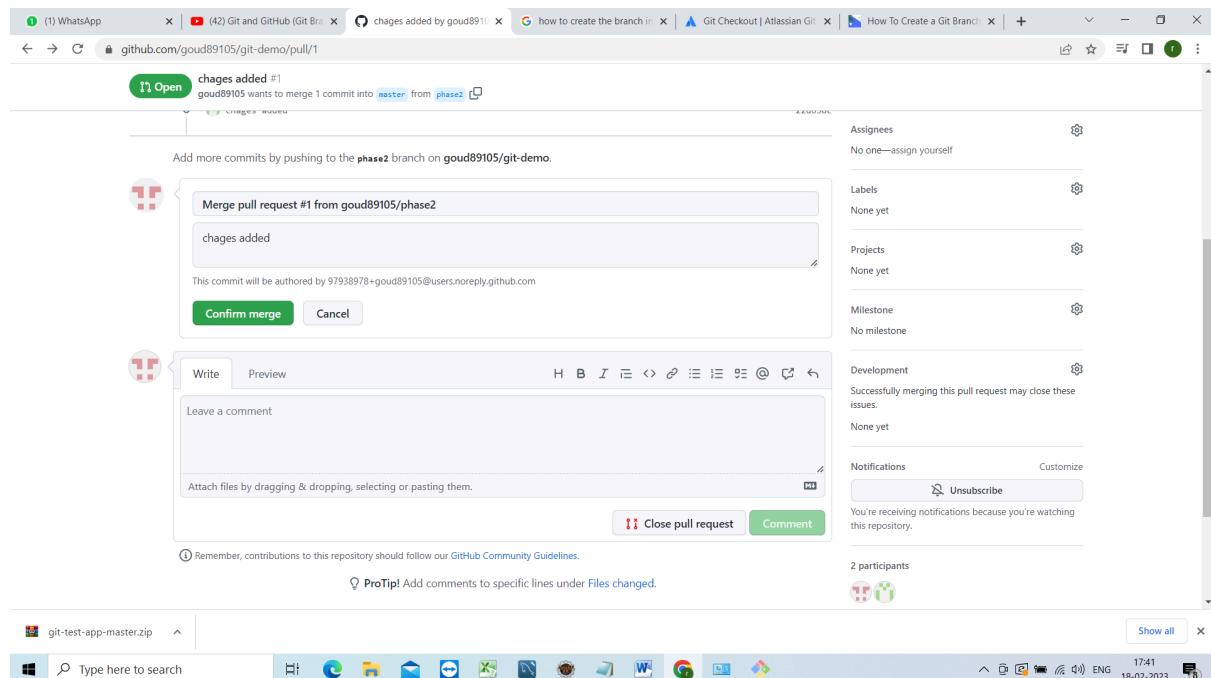
git-test-app-master.zip

Type here to search

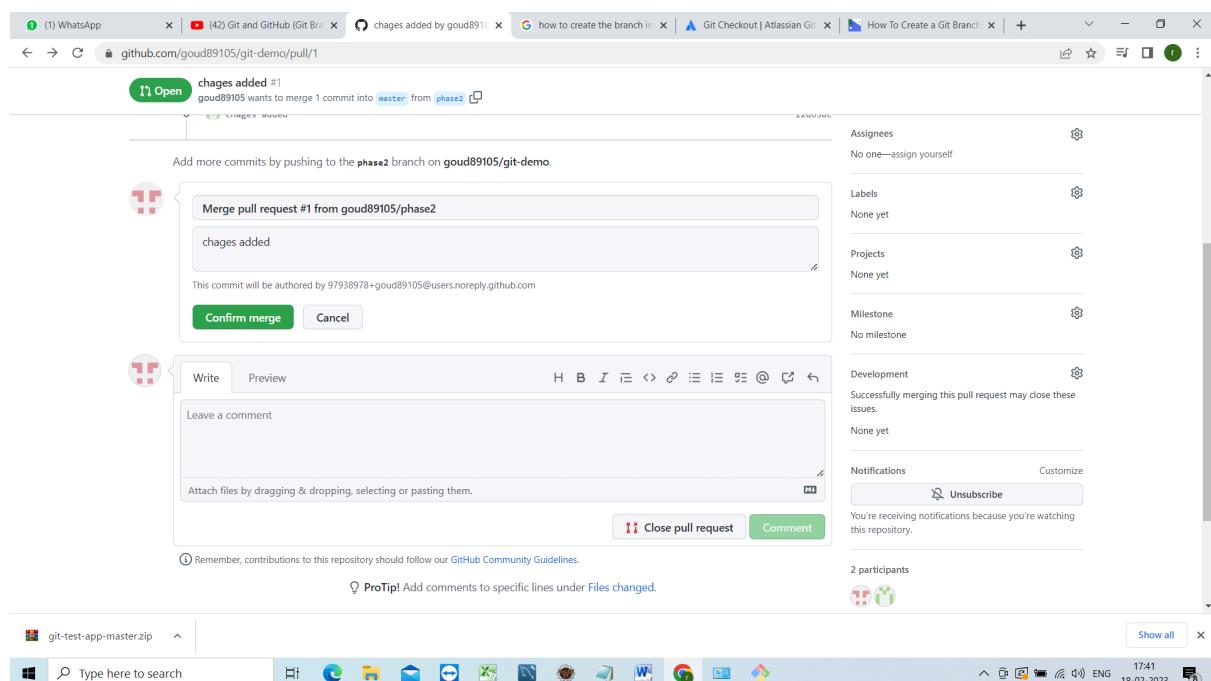
18-02-2023 17:39

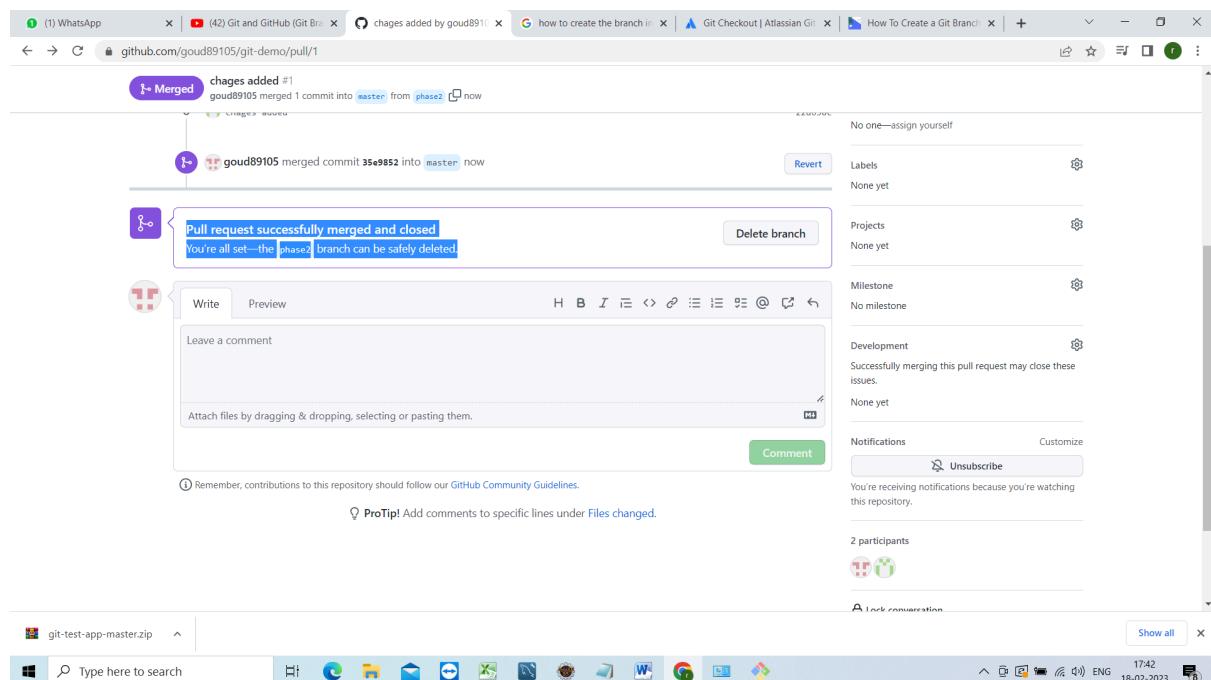
So that u can compare and can the review there

Now go back and click on merge pull request



Click on confirm merge

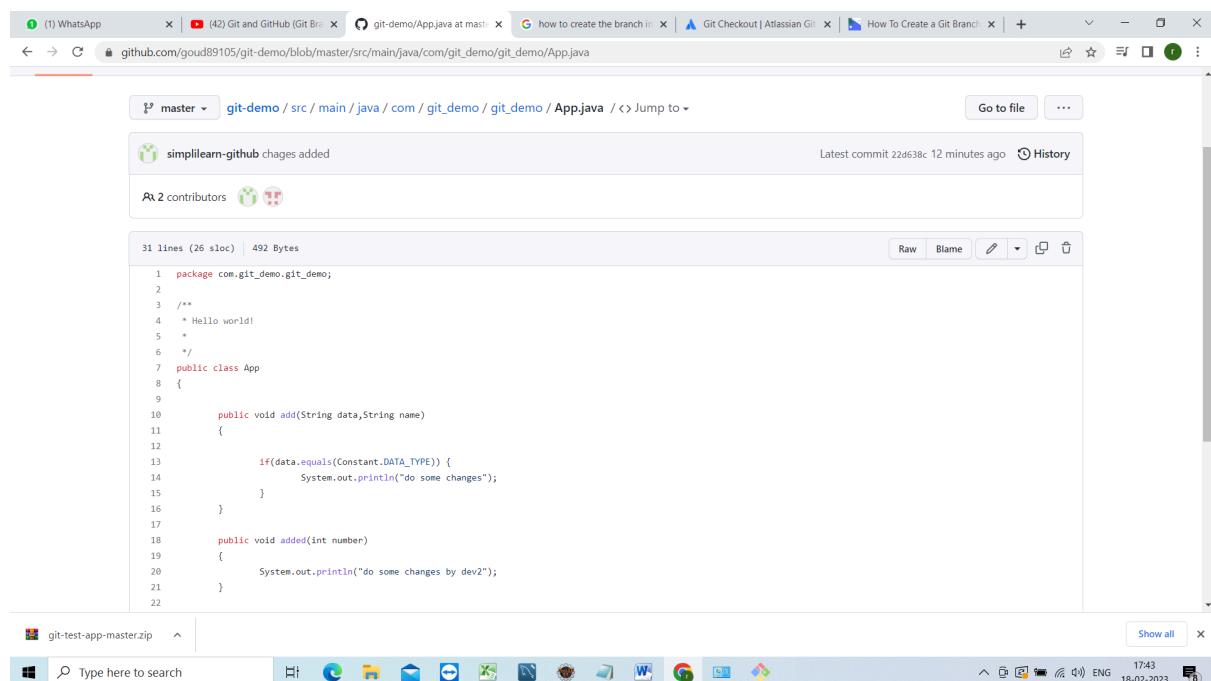




Now u have to give this pull request URL to ur team lead

<https://github.com/goud89105/git-demo/pull/1>

no go back and see this chages refleted or not



Now go to git bash changes are there in phase2 not in master changes wont come to master until u take pull

```
MINGW64/c/git-demo-example
$ git branch phase2
* phase2

MINGW64/c/git-demo-example (phase2)
$ git checkout phase2
Switched to branch 'phase2'
M target/classes/META-INF/maven/com.git_demo/git-demo/pom.properties
M target/classes/com/git_demo/git_demo/App.class

MINGW64/c/git-demo-example (phase2)
$ git branch
* phase2

MINGW64/c/git-demo-example (phase2)
$ git status
On branch phase2
Changes not staged for commit:
  (use "git add <file>" to update what will be committed)
  (use "git add --all <file>" to discard changes in working directory)
    modified: src/main/java/com/git_demo/git_demo/App.java
    modified: target/classes/META-INF/maven/com.git_demo/git_demo/pom.properties
    modified: target/classes/com/git_demo/git_demo/App.class

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

no changes added to commit (use "git add" and/or "git commit -a")

MINGW64/c/git-demo-example (phase2)
$ git commit -m "changes added"
[phase2 22d638c] changes added
 4 files changed, 5 insertions(+), 4 deletions(-)
 create mode 100644 .gitignore

MINGW64/c/git-demo-example (phase2)
$ git push origin phase2
Enumerating objects: 39, done.
Counting objects: 100% (39/39), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (21/21), 1.68 KiB | 573.00 KiB/s, done.
Total 21 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
remote:
remote: Create a pull request for 'phase2' on GitHub by visiting:
remote:   https://github.com/goud89105/git-demo/pull/new/phase2
remote:
To https://github.com/goud89105/git-demo.git
 * [new branch]  phase2 -> phase2

MINGW64/c/git-demo-example (phase2)
$
```

Switch to master branch

```
MINGW64/c/git-demo-example
$ git branch phase2
* phase2

MINGW64/c/git-demo-example (master)
$ git checkout phase2
Switched to branch 'phase2'
M target/classes/META-INF/maven/com.git_demo/git_demo/pom.properties
M target/classes/com.git_demo/git_demo/App.class

MINGW64/c/git-demo-example (phase2)
$ git branch
* phase2

MINGW64/c/git-demo-example (phase2)
$ git status
On branch phase2
Changes not staged for commit:
  (use "git add <file>" to update what will be committed)
  (use "git add --all <file>" to discard changes in working directory)
    modified: src/main/java/com/git_demo/git_demo/App.java
    modified: target/classes/META-INF/maven/com.git_demo/git_demo/pom.properties
    modified: target/classes/com/git_demo/git_demo/App.class

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

no changes added to commit (use "git add" and/or "git commit -a")

MINGW64/c/git-demo-example (phase2)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

MINGW64/c/git-demo-example (phase2)
$ git commit -m "changes added"
[phase2 22d638c] changes added
 4 files changed, 5 insertions(+), 4 deletions(-)
 create mode 100644 .gitignore

MINGW64/c/git-demo-example (phase2)
$ git push origin phase2
Enumerating objects: 39, done.
Counting objects: 100% (39/39), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (21/21), 1.68 KiB | 573.00 KiB/s, done.
Total 21 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
remote:
remote: Create a pull request for 'phase2' on GitHub by visiting:
remote:   https://github.com/goud89105/git-demo/pull/new/phase2
remote:
To https://github.com/goud89105/git-demo.git
 * [new branch]  phase2 -> phase2

MINGW64/c/git-demo-example (phase2)
$ git checkout master
```

```
MINGW64/c/git-demo-example
$ git checkout phase2
Switched to branch 'phase2'
M      target/classes/META-INF/maven/com.git_demo/git-demo/pom.properties
M      target/classes/com.git_demo/git_demo/App.class
$ git branch
* master
  phase2
$ git status
On branch phase2
Changes not staged for commit:
  (use "git add <file>" to update what will be committed)
  (use "git add --all" to update all changes in the working directory)
    modified:  src/main/java/com/git_demo/git_demo/App.java
    modified:  target/classes/META-INF/maven/com.git_demo/git-demo/pom.properties
    modified:  target/classes/com.git_demo/git_demo/App.class

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

no changes added to commit (use "git add" and/or "git commit -a")

$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
$ git commit -m "Initial commit"
[master 234638c] initial commit
 4 files changed, 5 insertions(+), 4 deletions(-)
 create mode 100644 .gitignore
$ git push origin phase2
Enumerating objects: 39, done.
Counting objects: 100% (39/39), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (23/23), 1.68 KiB | 573.00 KiB/s, done.
Total 23 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
remote:
remote: Create a pull request for 'phase2' on GitHub by visiting:
remote:   https://github.com/goud89105/git-demo/pull/new/phase2
remote:
remote: To https://github.com/goud89105/git-demo.git
 * [new branch]    phase2 -> phase2
$ git checkout master
Switched to branch 'master'
$
```

Now we are in master branch just clear the screen enter clear

```
MINGW64/c/git-demo-example
$ git checkout master
$
```

Now check git branch

A screenshot of a Windows desktop environment. At the top is a black terminal window titled 'MINGW64/c/git-demo-example'. It contains the following command-line session:

```
bell@DESKTOP-02V1815: MINGW64 /c/git-demo-example (master)
$ git branch
* master
  phase2
bell@DESKTOP-02V1815: MINGW64 /c/git-demo-example (master)
$
```

The taskbar below the terminal shows various pinned icons, including File Explorer, Task View, Start, Task Manager, and several browser and utility icons. The system tray indicates the date as 18-02-2023 and the time as 17:47.

Now go to eclipse and see we are in master branch changes not reflected to the master branch

A screenshot of the Eclipse IDE interface. The title bar reads 'eclipse-workspace - git-demo/src/main/java/com/git_demo/git_demo/App.java - Eclipse IDE'. The left side features the 'Project Explorer' view, which lists several projects and their components. The central area is a code editor displaying the 'App.java' file:

```
1 package com.git_demo.git_demo;
2
3 /**
4  * Hello world!
5  */
6 public class App
7 {
8     public void add(String data)
9     {
10        if(data.equals(Constant.DATA_TYPE))
11        {
12            System.out.println("do some changes");
13        }
14    }
15
16    public void added()
17    {
18        System.out.println("do some changes by dev2");
19    }
20
21    public void deletion()
22    {
23        System.out.println("do some changes by dev2");
24    }
25
26    public static void main(String[] args )
27    {
28        System.out.println( "Hello World!" );
29    }
30
31 }
```

The code editor has tabs for 'Constant.java' and 'App.java'. To the right of the code editor are toolbars for 'Variables', 'Breakpoints', and 'Expressions'. The bottom of the screen shows the Windows taskbar with pinned icons and the system tray indicating the date as 18-02-2023 and the time as 17:51.

Now go to git bash and do pull

The screenshot shows a Windows desktop environment. In the center is a terminal window titled 'MINGW64/c/git-demo-example'. The terminal displays a git pull command being run, showing progress and completion. The output includes:

```
git pull
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), done.
From https://github.com/goud89105/git-demo
   a1a8cb7..35e9852  master      -> origin/master
There is tracking information for the current branch.
  Please specify which branch you want to merge with.
  See git-pull(1) for details.

git pull <remote> <branch>
If you wish to set tracking information for this branch you can do so with:
  git branch --set-upstream-to=<origin>/<branch> master
```

At the bottom of the terminal window, there is a prompt '\$ |'.

Below the terminal, the Windows taskbar is visible, featuring the Start button, a search bar with the placeholder 'Type here to search', and several pinned application icons. The system tray shows the date and time as '18-02-2023 17:49'.

Now go to eclipse and see ur changes will be reflected in the master branch

Git Introduction

Git is a **distributed version control system** that helps developers track changes in their code, collaborate with others, and manage different versions of a project efficiently.

1. Create & Manage Repositories

A Git repository is a storage location where the history of changes in a project is maintained.

Create a New Repository

- Initialize a repository in an existing project

bash

```
git init
```

- Clone an existing repository

bash

```
git clone <repository_url>
```

Manage Repositories

- Check repository status

bash

```
git status
```

- View commit history

bash

```
git log
```

2. Repository Settings

Git repositories can be configured using settings like user credentials and ignored files.

- Set user details

bash

```
git config --global user.name "Your Name"  
git config --global user.email "your.email@example.com"
```

- **Ignore specific files** (Create a `.gitignore` file and add patterns)

bash

```
*.log  
node_modules/
```

- **View Git configurations**

bash

```
git config --list
```

3. Adding & Committing

Git tracks changes in **staging area** before committing.

- **Add files to staging**

bash

```
git add <file>
```

```
git add .
```

- **Commit changes**

bash

```
git commit -m "Commit message"
```

- **Amend last commit**

bash

```
git commit --amend -m "Updated commit message"
```

4. Branches

Branches allow multiple versions of a project to exist simultaneously.

- **Create a new branch**

bash

```
git branch <branch_name>
```

- **Switch to a branch**

bash

```
git checkout <branch_name>
```

- **Create and switch to a branch**

bash

```
git checkout -b <branch_name>
```

- **List all branches**

bash

```
git branch
```

- **Delete a branch**

bash

```
git branch -d <branch_name>
```

5. Stashing

Stashing temporarily saves changes without committing them.

- **Stash current changes**

bash

git stash

- **Apply stashed changes**

bash

CopyEdit

git stash pop

- **List stash entries**

bash

git stash list

- **Drop a stash entry**

bash

git stash drop

6. GitHub

GitHub is a cloud-based platform for hosting Git repositories.

- **Push local repository to GitHub**

bash

CopyEdit

git remote add origin <repository_url>

git push -u origin main

- **Clone a repository from GitHub**

bash

git clone <repository_url>

7. Fetching & Pulling

Fetching and pulling bring updates from remote repositories.

- **Fetch changes without merging**

bash

CopyEdit

git fetch

- **Pull changes and merge automatically**

bash

git pull

8. Git Collaboration

Collaboration involves multiple contributors working on a project.

- **Push changes to remote**

bash

CopyEdit

git push origin <branch_name>

- **Merge branches**

bash

git merge <branch_name>

- **Resolve merge conflicts manually in affected files, then commit changes.**
-

9. Git Tags

Tags mark specific points in a project's history.

- **Create a tag**

bash

CopyEdit

```
git tag <tag_name>
```

- **Push tags to remote**

bash

CopyEdit

```
git push origin <tag_name>
```

- **List all tags**

bash

```
git tag
```

10. Git Clone

Cloning copies a repository from a remote source.

- **Clone a repository**

bash

```
git clone <repository_url>
```

11. Git Forking

Forking creates a personal copy of another user's repository on GitHub.

- **Steps to Fork a repository:**

1. Open the repository on GitHub.
2. Click the "Fork" button.
3. Clone the forked repository to your local system:

bash

```
git clone <your_forked_repo_url>
```

12. Merge Requests

A Merge Request (or Pull Request on GitHub) is used to merge changes from one branch into another.

- **Steps to create a Merge Request:**

1. Push your changes to a branch.
2. On GitHub, navigate to your repository.
3. Click **New Pull Request**.
4. Select branches and submit.