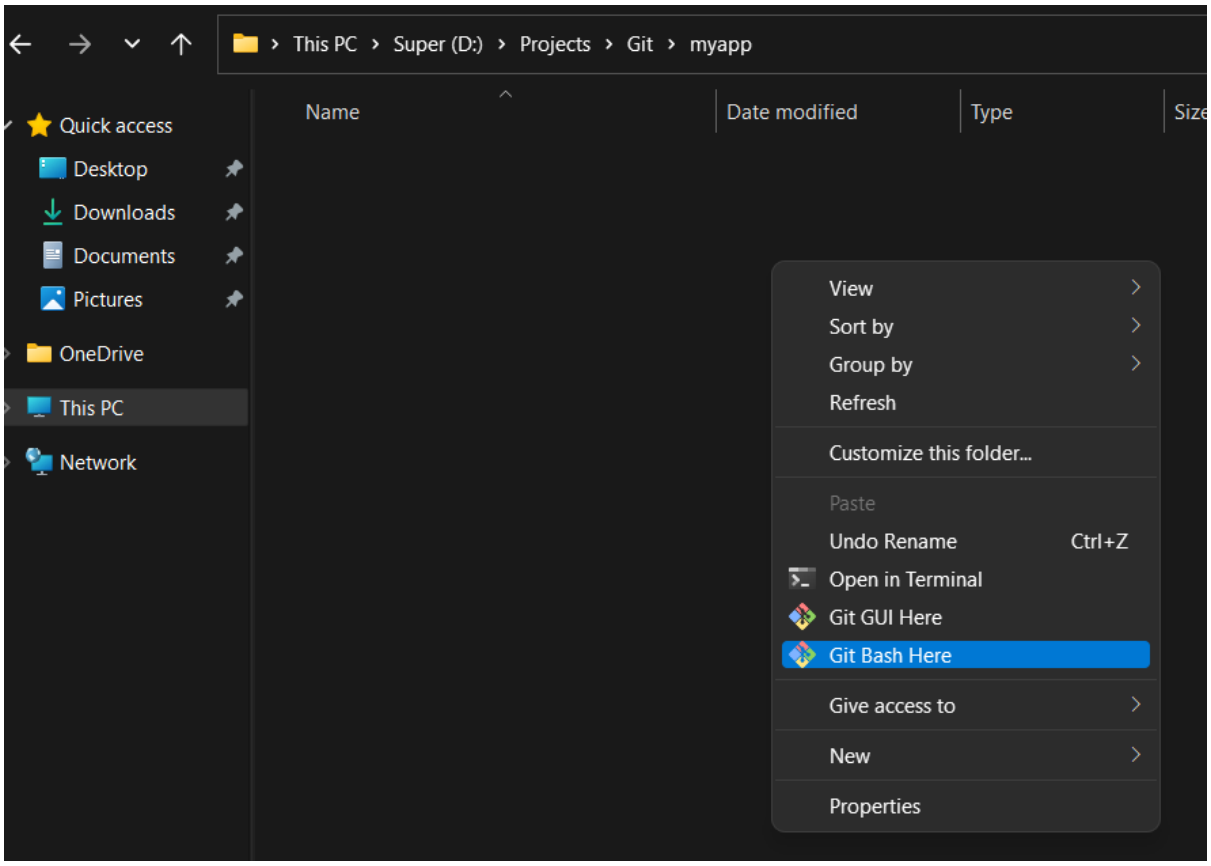
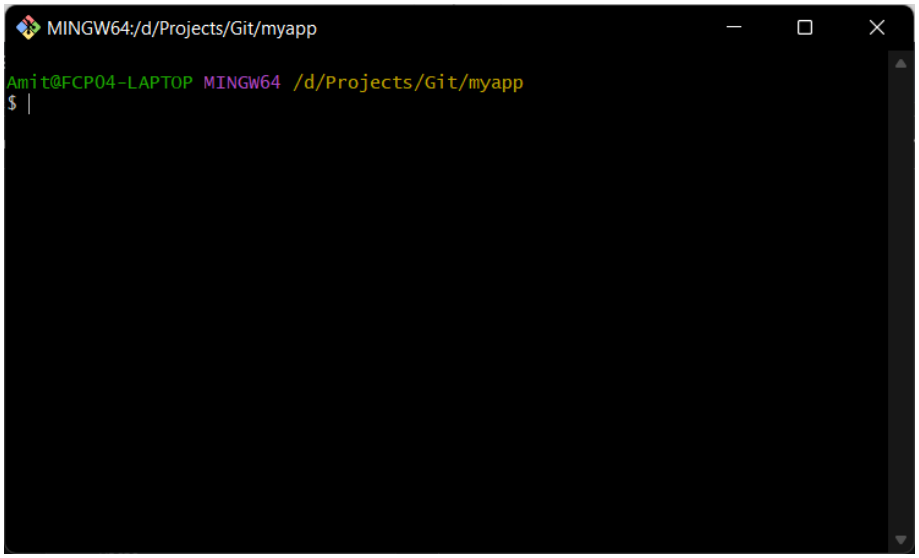
Git hands-on

Create a folder named “myapp”

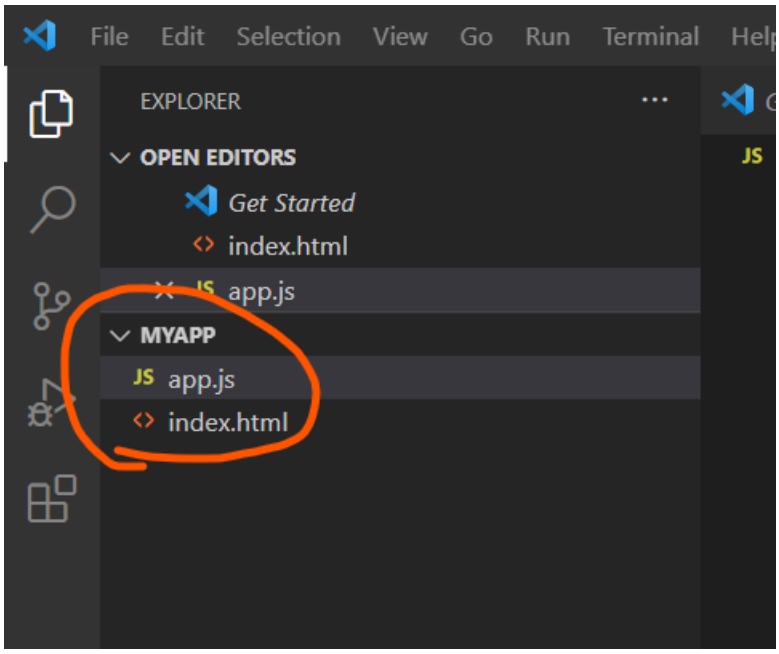
In windows explorer, navigate to the myapp folder, right click and select “Git Bash Here”

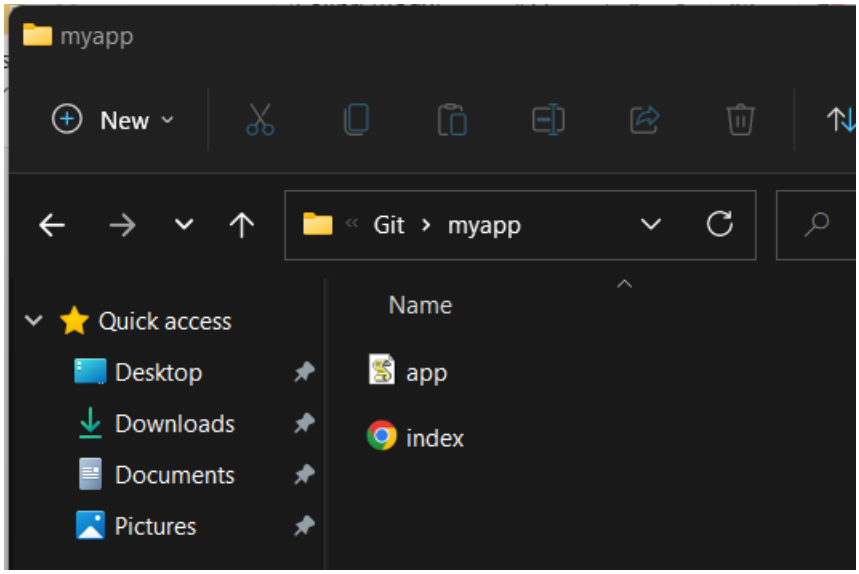


It will open Git Bash as shown below

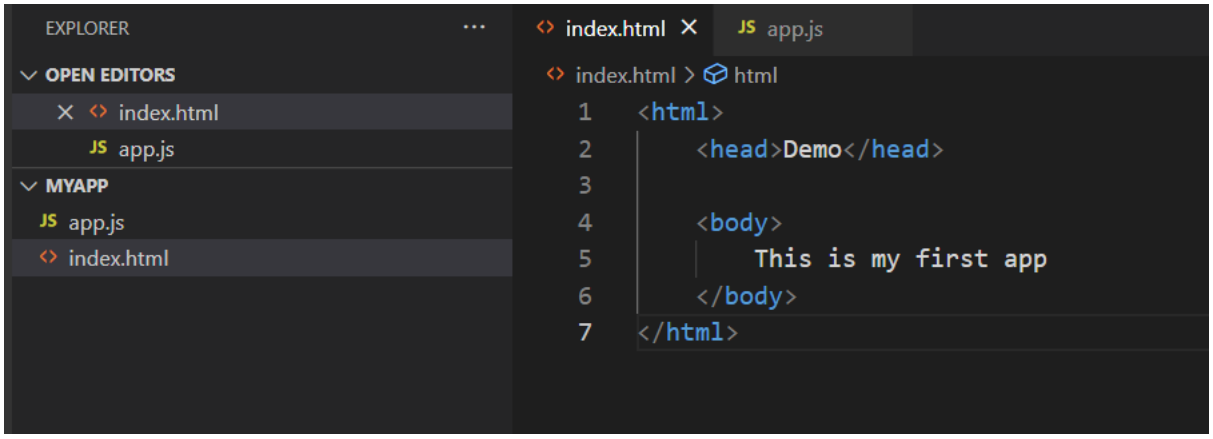


Create few files in the folder



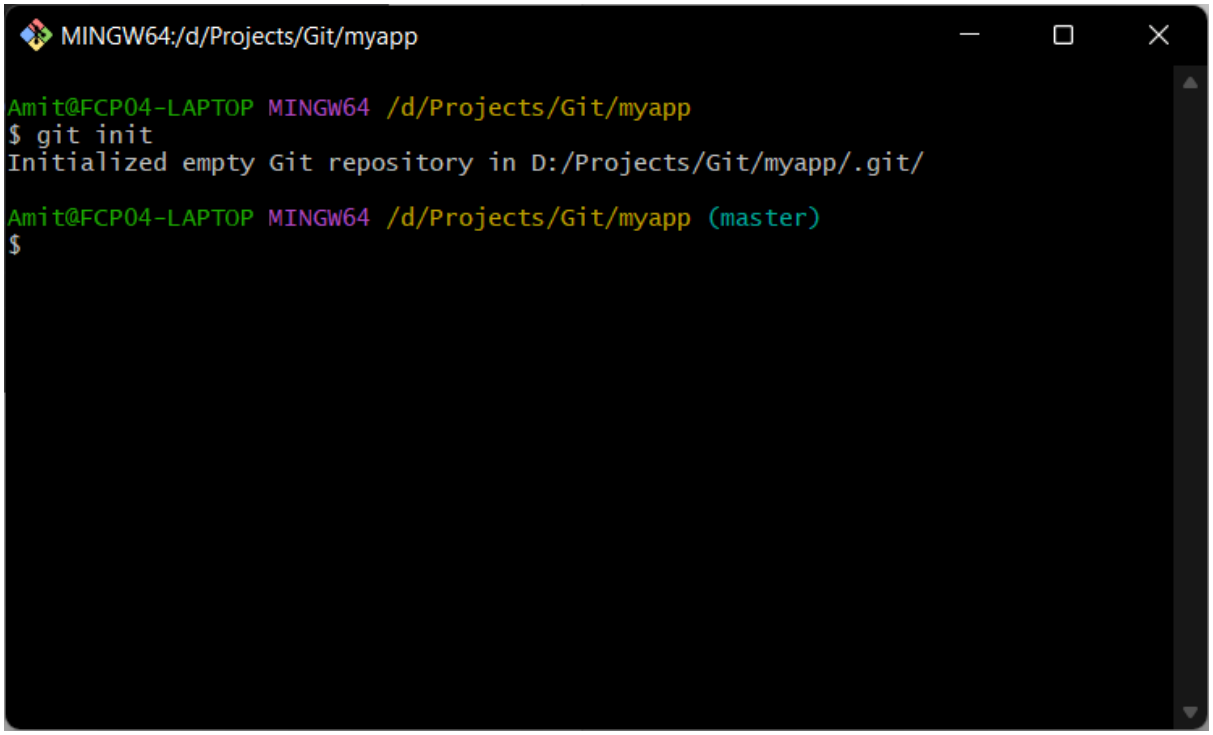


Add some code



Now we wish to initialize the folder as a GIT repository

Issue the $ git init command as shown below:



It’s and hidden folder

So we can unhide and check it

View -> Show -> Hidden items

Thus a new local repository is created for our app

You may want to add your name and email id to Git. To do that use the config command

```
MINGW64:/d/Projects/Git/myapp

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp
$ git init
Initialized empty Git repository in D:/Projects/Git/myapp/.git/

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git config --global user.name 'Amit Borkar'

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git config --global user.email 'amitg.borkar@gmail.com'

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

Now let's add our index.html file to our local repository

```
MINGW64:/d/Projects/Git/myapp

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp
$ git init
Initialized empty Git repository in D:/Projects/Git/myapp/.git/

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git config --global user.name 'Amit Borkar'

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git config --global user.email 'amitg.borkar@gmail.com'

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git add index.html

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

Now if you observe, it doesn't give us any response as such, but it has added our index.html file to the staging area

Let's check what is available in our staging area: use command $ git status

```
MINGW64:/d/Projects/Git/myapp

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git config --global user.email 'amitg.borkar@gmail.com'

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git add index.html

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

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


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

Observe it is showing us

- Changes to be committed (means file(s) available in staging area)
- Untracked files (means file(s) which are not part of staging area yet)

To add files

- git add index.html → to add single file
- git add *.html → to add all files of a certain type
- git add . → to add all files

NOTE: we have not yet committed our files to the repo.

Let's make some changes to the index.html file

```
index.html M ×   JS app.js U
index.html > html > body
1  <html>
2      <head>Demo</head>
3
4      <body>
5          This is my first app !!
6      </body>
7  </html>
```

And now if we check the status:

```
MINGW64:/d/Projects/Git/myapp

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

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:   index.html

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


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

Let's do a "git add ."

```
MINGW64:/d/Projects/Git/myapp
        modified:   index.html

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


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git add .

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   app.js
        new file:   index.html


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

All files are in staging area !!!

COMMIT

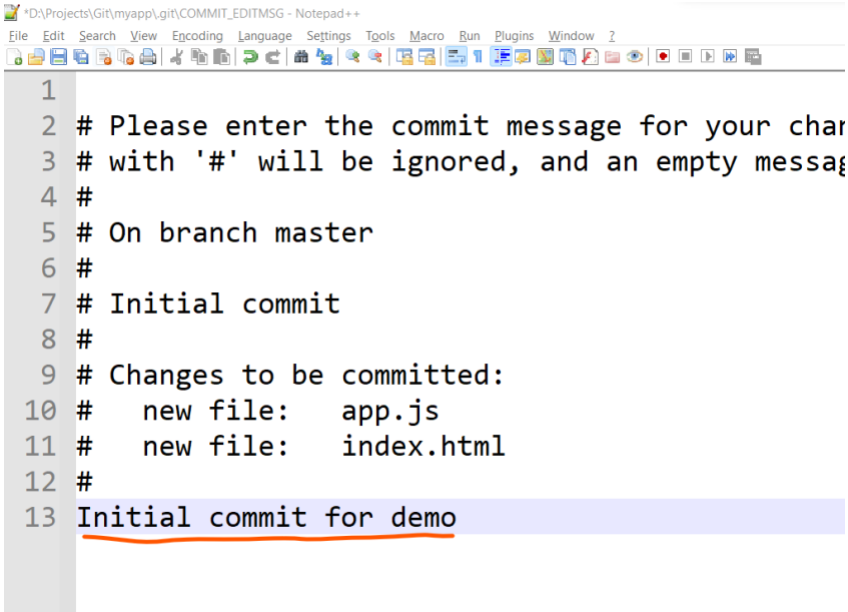
Let's understand how do we commit our changes to the Git (local repo)

```
MINGW64:/d/Projects/Git/myapp

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


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git add .

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   app.js
        new file:   index.html


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git commit
hint: Waiting for your editor to close the file... |
```

It will open a file in Notepad++ (remember during installation we had set Notepad++ as the default)

```
D:\Projects\Git\myapp\.git\COMMIT_EDITMSG - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

1
2  # Please enter the commit message for your changes. Lines starting
3  # with '#' will be ignored, and an empty message aborts the commit.
4  #
5  # On branch master
6  #
7  # Initial commit
8  #
9  # Changes to be committed:
10 #   new file:   app.js
11 #   new file:   index.html
12 #
13
```

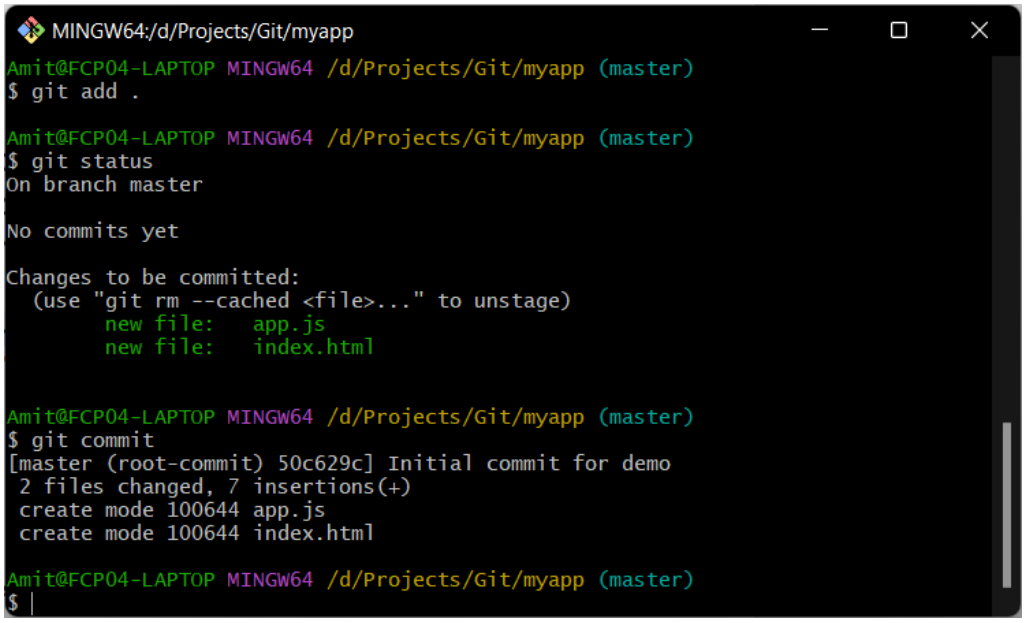
Add a comment as shown below:



Save the file

Close the Editor (Remember the Git bash is expecting us to close the editor, before it can start committing)

After closing the editor, you will see the following messages in the Git Bash



$ git status → Nothing to commit

```
MINGW64:/d/Projects/Git/myapp
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   app.js
        new file:   index.html


Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git commit
[master (root-commit) 50c629c] Initial commit for demo
 2 files changed, 7 insertions(+)
 create mode 100644 app.js
 create mode 100644 index.html

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master
nothing to commit, working tree clean

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ |
```

Next we will see how to skip the editing of file during commit:

Add some code to app.js

```
index.html    app.js  M ×
app.js
1   console.log('Hello World');
2
```

git status

```
MINGW64:/d/Projects/Git/myapp
Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git commit
[master (root-commit) 50c629c] Initial commit for demo
 2 files changed, 7 insertions(+)
 create mode 100644 app.js
 create mode 100644 index.html

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master
nothing to commit, working tree clean

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master
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:   app.js

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

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

git add .

Now, let's commit as shown below:

```
MINGW64:/d/Projects/Git/myapp
$ git status
On branch master
nothing to commit, working tree clean

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git status
On branch master
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:   app.js

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

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git add .

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git commit -m 'Changed app.js'
[master 5506d3d] Changed app.js
 1 file changed, 1 insertion(+)

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ |
```

Note: For any files that need to be ignored follow the steps

- Create a file named ".gitignore"
- Add the name of the file(s) to this file E.g "log.txt"
- Git will ignore all the files that are listed in ".gitignore" file
- To ignore an entire folder, add the folder name to file e.g "/dir1"
- Use "*.txt" to ignore all text files etc…for other options see documentation

Branch

We can create a new branch, whenever we want to work on some functionality and not want to make any changes to the main branch

$ git branch login → Create a new branch

Currently we are at master branch. To switch to login branch

$ git checkout login → to change to login branch

Any new creation, updates are now done on this branch only (and not master)

Let's do 2 things:

- Create a new file (login.html), while we are on this branch
- Edit the index.html file while we are on login branch

git add .

git commit -m 'login form'

Now when you switch back to 'master' branch observe what is shows in windows explorer

- The login.html file will no longer be visible
- The index.html file will not have changes that you made while you were on the 'login' branch

MERGE:

Imagine we have implemented the login functionality and we are ready to merge,

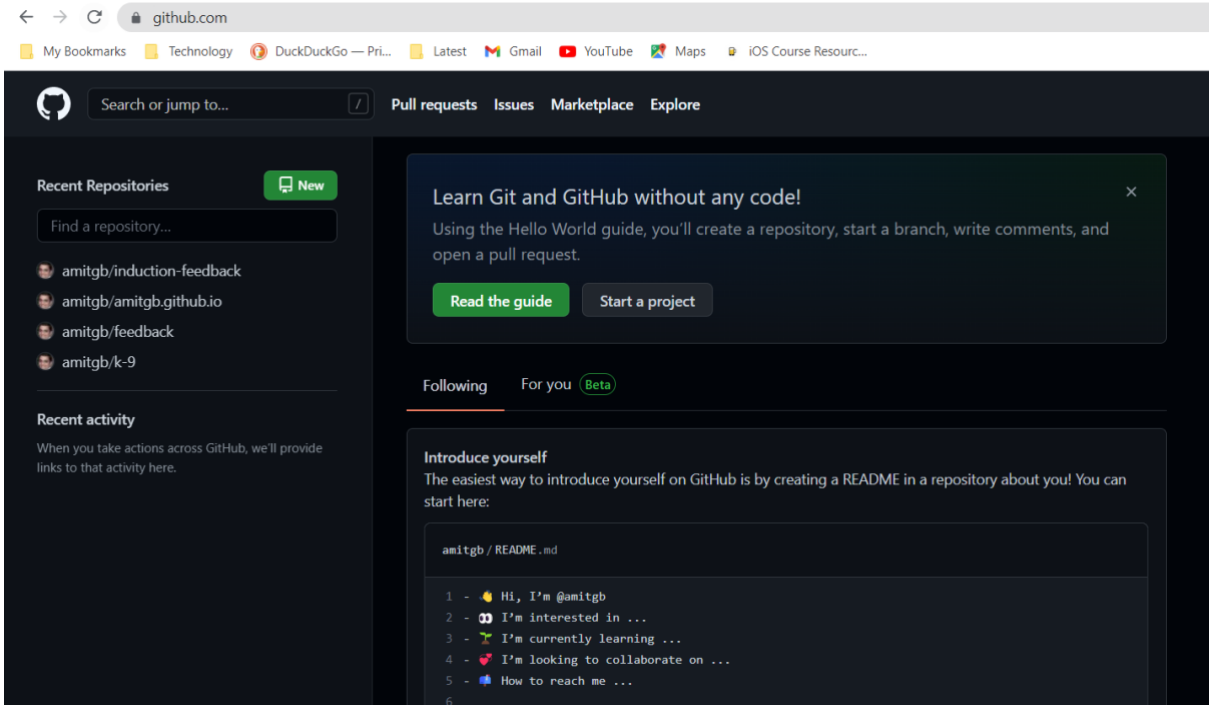
While on master branch issue:

- git merge login

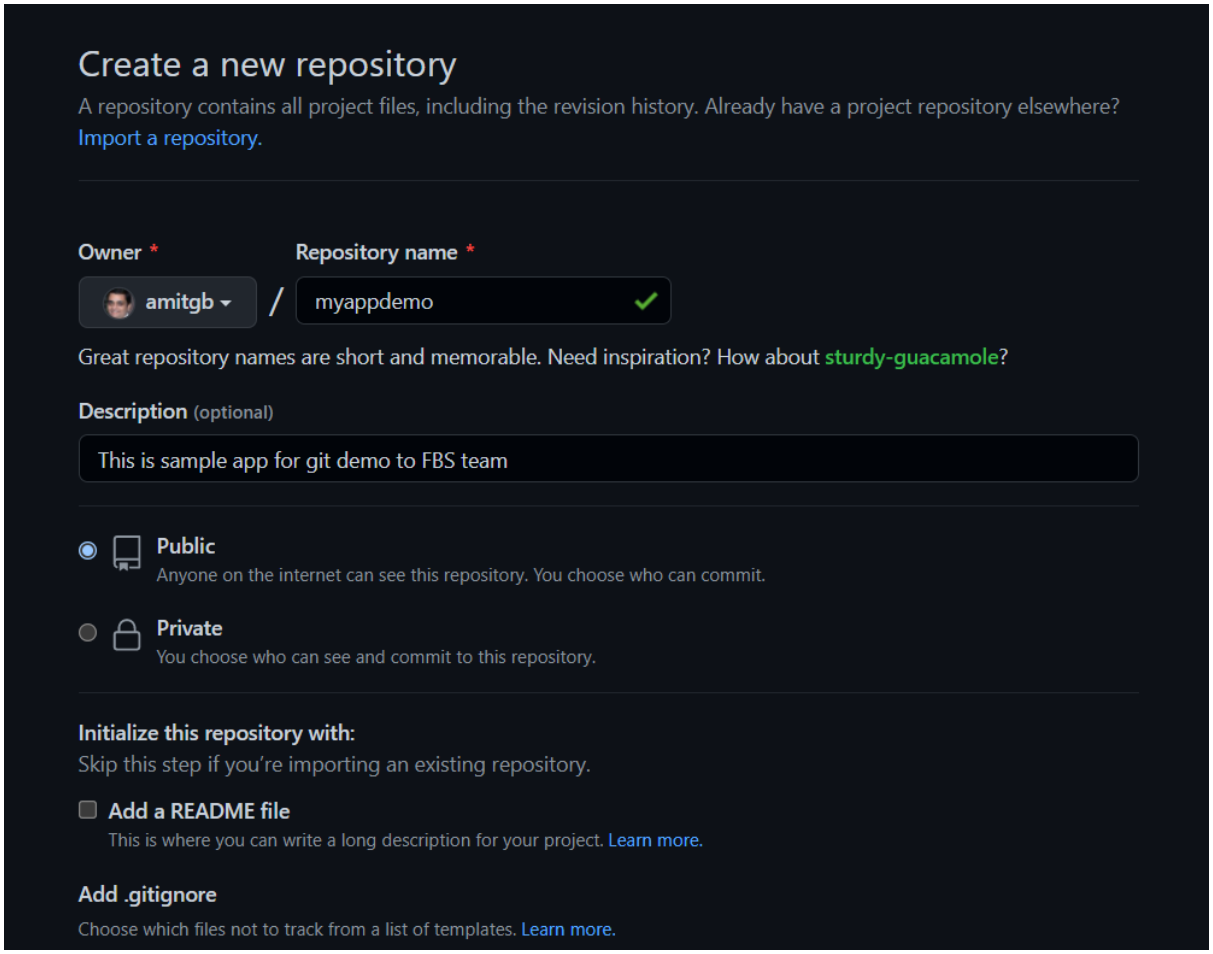
Note: When we are the only one working on our project, we wont need merging functionality

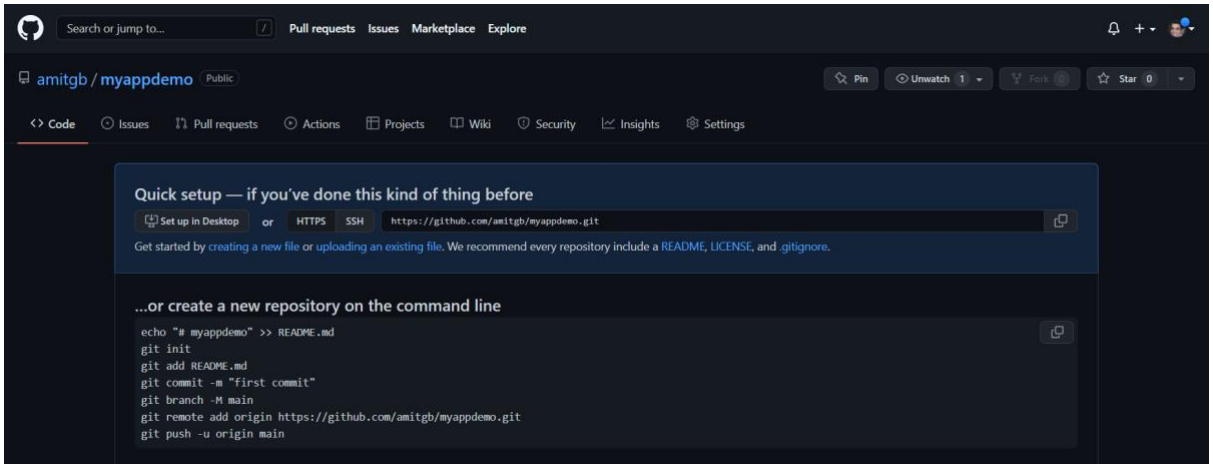
Working with Remote Repositories

github.com



Create a new repository:



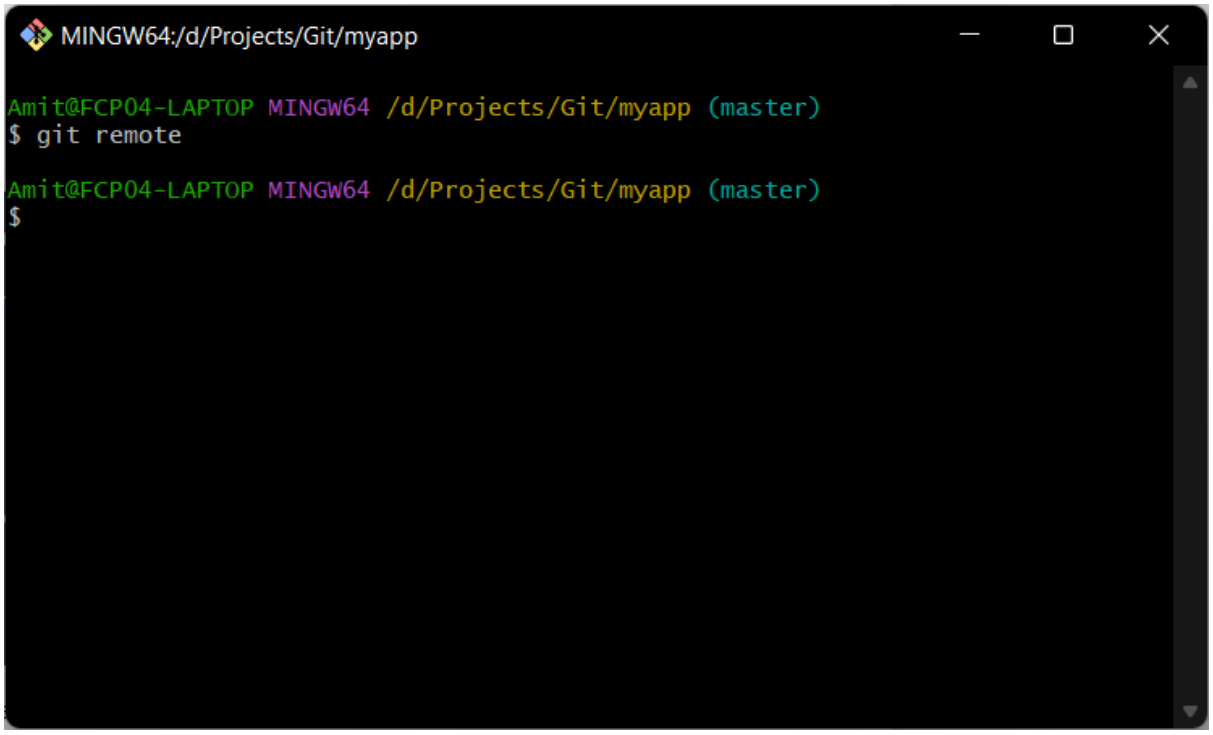


A good idea is to add a readme.md file if your repo is on public forum. This will provide info about your project to others

It shows some help like:

We only need to do the last part (2 statements)

Let's check in our Bash which remote repositories do we have?



As you can see we have none !!

Next, let's copy the command from github:

```
git remote add origin https://github.com/amitgb/myappdemo.git
```

Add the remote repository:

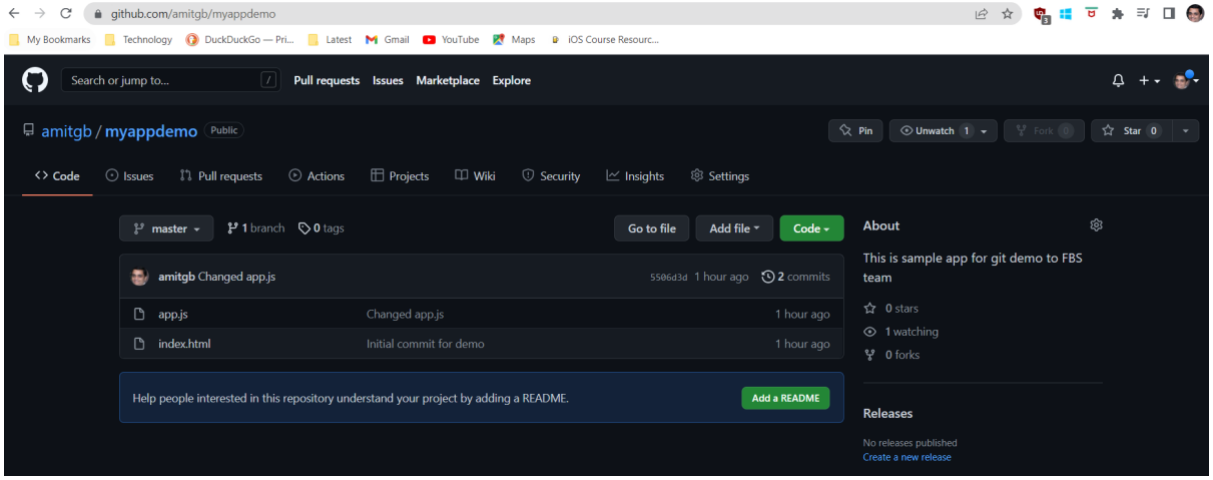
```
MINGW64:/d/Projects/Git/myapp

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git remote

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git remote add origin https://github.com/amitgb/myappdemo.git

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git remote
origin

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

Now let's push our local repo to remote repo:

```
git push -u origin master
(NOTE: In github the name is 'main' whereas in git (locally) we have the name
as 'master')

When the command is issued it will ask for authentication, which could be done
based on your browser (git will open a small window which will allow you to
select 'browser based' or 'token' or something similar. Selecting browser will
open a link in your browser and then it will ask you to login. Once
successfully logged in, the push command will be successful)
```

```
Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git push -u origin master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 600 bytes | 600.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/amitgb/myappdemo.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```

Refresh your GitHub page to see following:



Other Commands:

To check logs:

```
Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$ git log index.html
commit 5457b53b66320d488f0746a6367714fe46044d5e (HEAD -> master)
Author: Amit Borkar <amitg.borkar@gmail.com>
Date:   Mon Apr 25 11:14:18 2022 +0530

    Added new line

commit 50c629c3200a9aeebad5ad704c1629b4eee67c3a
Author: Amit Borkar <amitg.borkar@gmail.com>
Date:   Mon Apr 25 09:57:39 2022 +0530

    Initial commit for demo

Amit@FCP04-LAPTOP MINGW64 /d/Projects/Git/myapp (master)
$
```