# ----------Use GIT BASH--------------



Install git first

Step 2:start open git bash

pwd - print working directory

ls - List files also ls -l to get list

cd directoryname - Change directory like cd desktop

touch filename - Make file

rm filename - Remove file

rm -r directoryname - Remove directory

clear - ear what’s on the screen

nano - create a file and write in it

../ - go up one level

Admin@MyLaptop MINGW64 ~

$ git --version

git version 2.46.0.windows.1

Admin@MyLaptop MINGW64 ~

$ pwd

/c/Users/Admin

Admin@MyLaptop MINGW64 ~

$ ls //to see list of all folders in the particular directlory

**AppData**/

**'Application Data'**@

**Contacts**/

**Cookies**@

**Desktop**/

**Documents**/

**Downloads**/

**Favorites**/

**IntelGraphicsProfiles**/

**Links**/

**'Local Settings'**@

**Music**/

**'My Documents'**@

NTUSER.DAT

NTUSER.DAT{db4548c7-0a30-11ef-b378-b5b3cc685af1}.TM.blf

NTUSER.DAT{db4548c7-0a30-11ef-b378-b5b3cc685af1}.TMContainer00000000000000000001.regtrans-ms

NTUSER.DAT{db4548c7-0a30-11ef-b378-b5b3cc685af1}.TMContainer00000000000000000002.regtrans-ms

**NetHood**@

**OneDrive**/

**Pictures**/

**Postman**/

**PrintHood**@

**Recent**@

**'Saved Games'**/

**Searches**/

**SendTo**@

**'Start Menu'**@

**Templates**@

**Tracing**/

**Videos**/

**eclipse**/

**eclipse-workspace**/

**letslearngit**/

ntuser.dat.LOG1

ntuser.dat.LOG2

ntuser.ini

package-lock.json

**socailmedia-master**/

Admin@MyLaptop MINGW64 ~

$ cd Desktop //to change directory and get to to the directory that we want

Admin@MyLaptop MINGW64 ~/Desktop

$ cd ../ //just to go back to location just one step back

Admin@MyLaptop MINGW64 ~

$ pwd //give us the path where we are at presently from root

/c/Users/Admin

Admin@MyLaptop MINGW64 ~

$ cd Desktop

Admin@MyLaptop MINGW64 ~/Desktop

$ mkdir test //to create folder

Admin@MyLaptop MINGW64 ~/Desktop

$ cd test //similary to get in the folder

Admin@MyLaptop MINGW64 ~/Desktop/test

$ touch index.html //to create file in the new directory or existing directory i.e folder

Admin@MyLaptop MINGW64 ~/Desktop/test

$ rm index.html //to remove or delete file

Admin@MyLaptop MINGW64 ~/Desktop/test

$ cd ../

Admin@MyLaptop MINGW64 ~/Desktop

$ rm test

rm: cannot remove 'test': Is a directory

Admin@MyLaptop MINGW64 ~/Desktop

$ cwd

bash: cwd: command not found

Admin@MyLaptop MINGW64 ~/Desktop

$ ls

09ac94260d534ad5175b.svg

**Brave.lnk**\*

**Discord.lnk**\*

**'Eclipse IDE for Java Developers - 2024-06.lnk'**\*

**'Econut - Chrome.lnk'**\*

'Fast-Track to Full Spectrum Software Engineering – Get SDE Ready.url'

**GitHub.lnk**\*

**'Google Meet.lnk'**\*

**'Lively Wallpaper.lnk'**\*

**'Microsoft Edge.lnk'**\*

**'New folder'**/

**Postman.lnk**\*

**'WPS Office.lnk'**\*

**YouTube.lnk**\*

**'Zoom Workplace.lnk'**\*

**'backend java springboot'**/

**'card mind game'**/

**clock**/

**'company logo'**/

desktop.ini

**dsa**/

**java**/

**'java dsa'**/

**'java practice for learners'**/

**'js practice'**/

**memories**/

**product**/

**socailmedia-master**/

**tailwindproject**/

**test**/

**'tic-tac-to game js'**/

**'web dev open'**/

Admin@MyLaptop MINGW64 ~/Desktop

$

# NOW playing with Git Commands

# Configuring Git

Use git bash for using gits or any commands





Example my Github name

Git-hub name:[Poorna-chandra2000](https://github.com/Poorna-chandra2000)

Must be in Camel case

*Poorna Chandra2000*

**Admin@MyLaptop MINGW64 ~**

**$ git config --global user.name "Poorna Chandra2000" //gihub name**

**Admin@MyLaptop MINGW64 ~**

**$ git config --global user.email "poornachandra473@gmail.com" //github user.email**

**Admin@MyLaptop MINGW64 ~**

**$ git config –list //to get list of what we configured**

diff.astextplain.textconv=astextplain

filter.lfs.clean=git-lfs clean -- %f

filter.lfs.smudge=git-lfs smudge -- %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

http.sslbackend=openssl

http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt

core.autocrlf=true

core.fscache=true

core.symlinks=false

pull.rebase=false

credential.helper=manager

credential.https://dev.azure.com.usehttppath=true

init.defaultbranch=master

user.name=Poorna Chandra2000

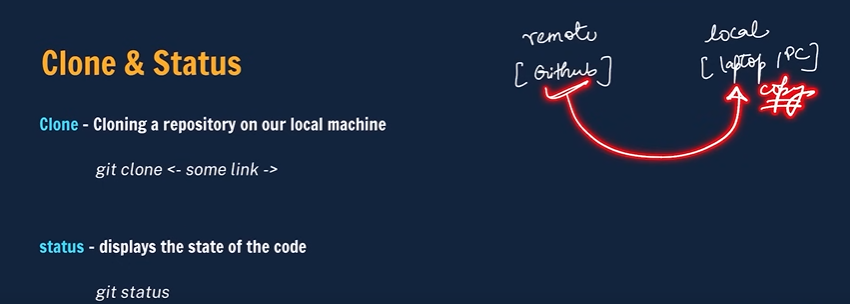
user.email=poornachandra473@gmail.com

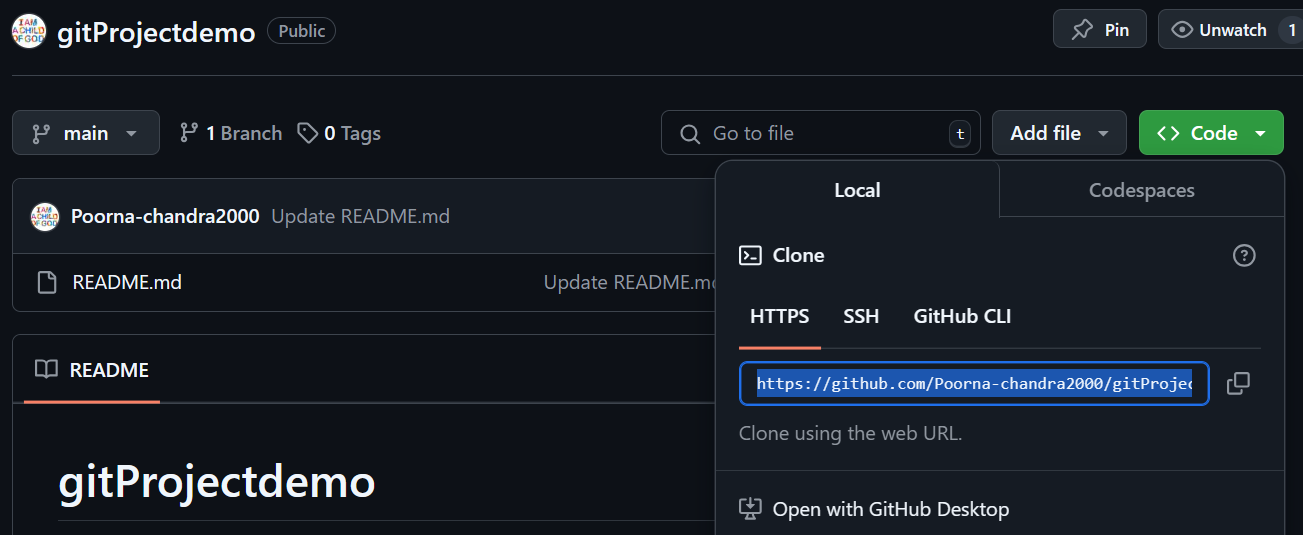
# Use git in editors instead for commands

# Now the best way to use git is through Editors like github or any platform

Always prefer cloning method

# Basic commands







**git clone** ***https..link*** from your github or anyones

ex-git clone <https://github.com/Poorna-chandra2000/gitProjectdemo.git>

C:\Users\Admin\Desktop\git Demo>git clone **https://github.com/Poorna-chandra2000/gitProjectdemo.git**

Cloning into 'gitProjectdemo'...

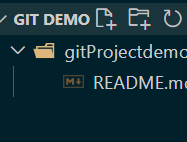
remote: Enumerating objects: 6, done.

remote: Counting objects: 100% (6/6), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

Receiving objects: 100% (6/6), done.

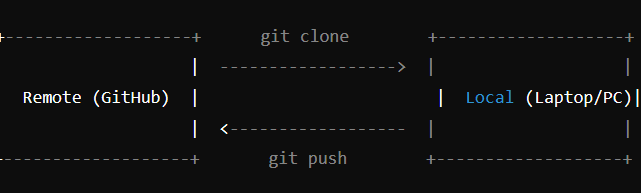
as we can see the files of gihb are now being cloned in local machine(viewing through VS code)

**Cloning** is the process of copying a repository (a collection of files and their version history) from a remote location (like GitHub) to your local machine (your laptop or PC).

1. **Steps to Clone:**
2. **Remote Repository**: This is where the code is stored online, for example, on GitHub.
3. **Local Machine**: This is your computer where you want to copy the repository.
4. **Clone Command**: You use the command git clone <link> to copy the repository from GitHub (remote) to your local machine.
5. **Git Status:**

* After cloning, you can use the git status command to check the state of the files in your repository.

1. **Simple Diagram Explanation:**

* **Remote (GitHub)**: This is shown on the left side of the diagram.
* **Local (Laptop/PC)**: This is on the right side of the diagram.
* **Arrow**: Indicates the direction of cloning, showing the repository being copied from GitHub to your local machine
* 

1. Cls/clear command to clear terminal
2. Admin@MyLaptop MINGW64 ~/Desktop/git Demo

$ **cd gitProjectdemo //now change direcetory to the directory you want remember that directory must be present (use Tab)shortcut so terminal will automatically direct you to thet directory is present**

1. Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

$ **ls //to see whats there in the chosen directory**

1. README.md
2. Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)
3. $ **ls -a //its also shows if there are any hidden files**
4. ./ ../ .git/ README.md

**Git Status**

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

$ git status //gives status of the application as below means updates or auditing

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clear

Now let us learn how to modify code

After modifying what happens

We just need to do this

Modify things on vs code which has clones…it will mark as M or m if

Any modifications are made..

Check status and it shows modified

Therefore we need to

Modify

|

Add

|

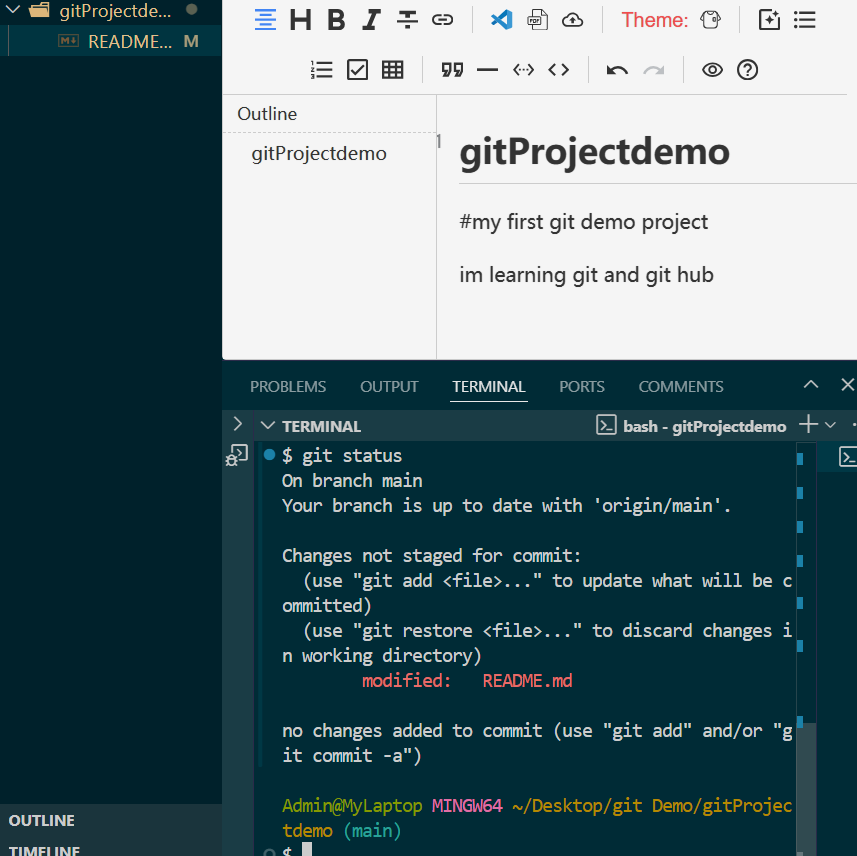
Commit

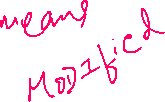
|

Push to github

Follow these steps only after modification

Must fisrt add and thn commit





Now lets add new Html file to check what kind of status it gives

Originally in github project that we cloned had no html file

Now let us create new html file in local machine and check

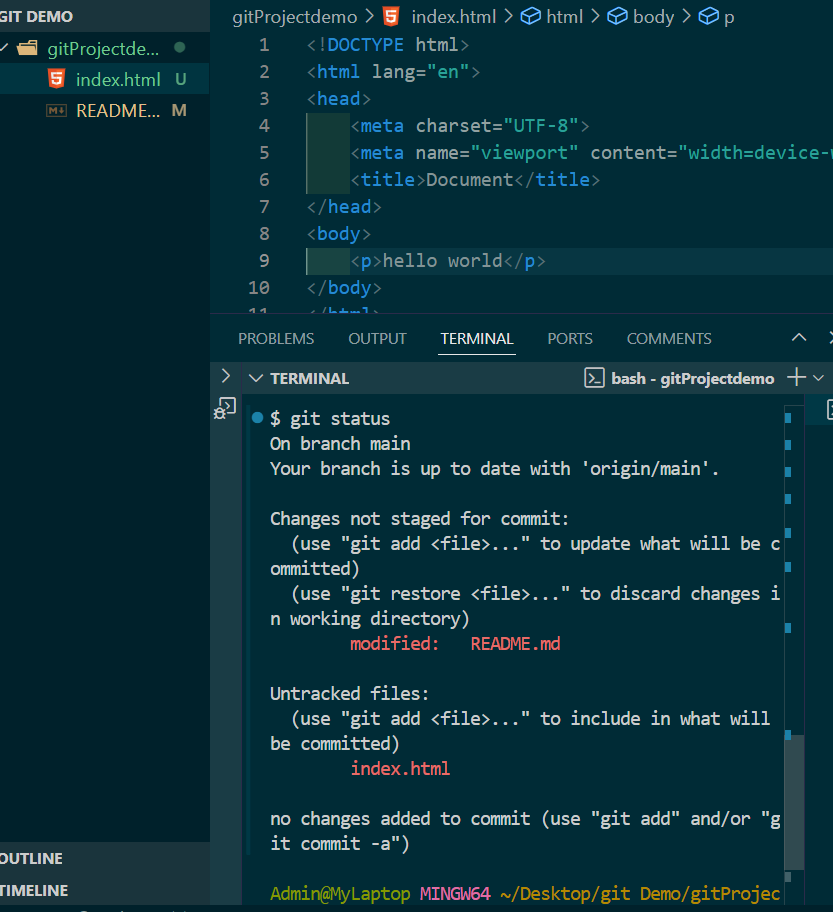
Lets see what status we get

The status also shows how to restore discard any changes made

And also how to add

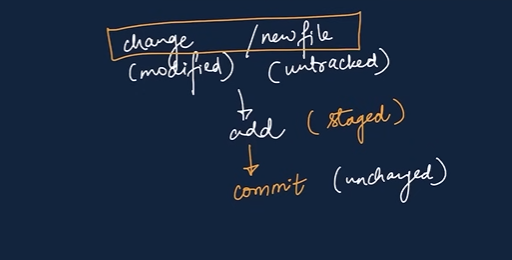
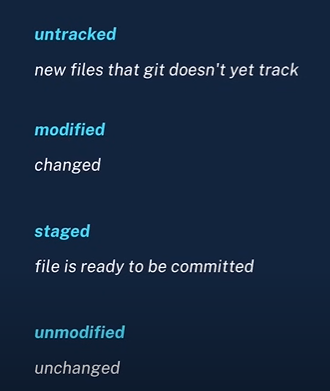


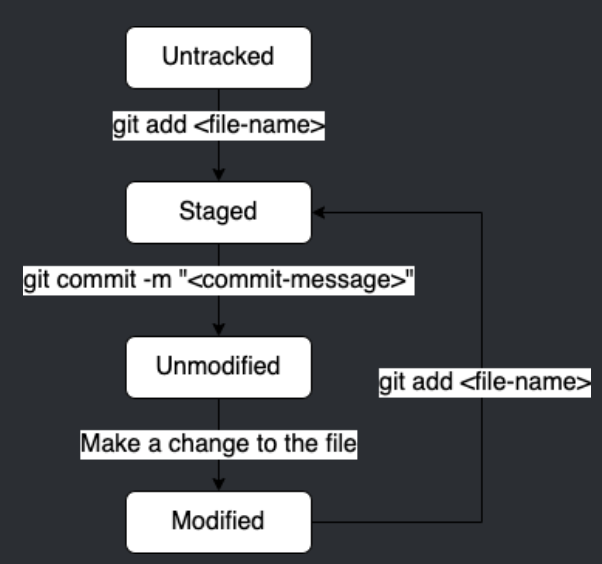
U stands for Untracked



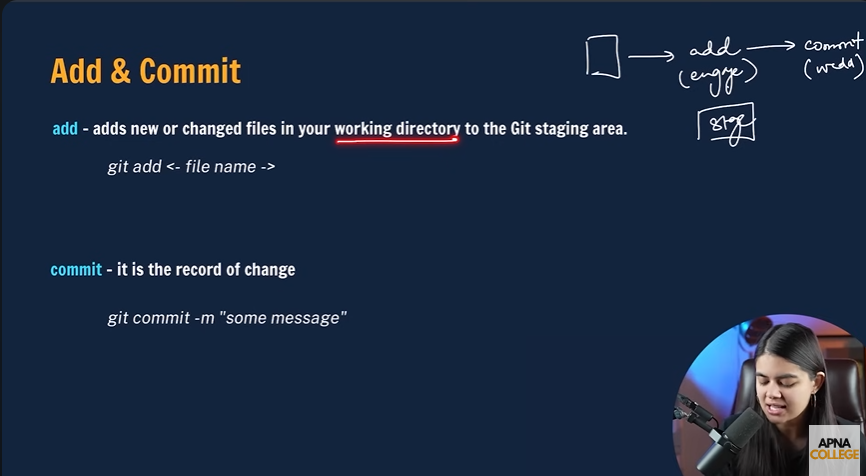


# Types of git status we may get





# Add and Commit



Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

$ Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

***$ git add index.html* //add file………….**

warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

////////////////////////////////////////////////////////////////////////////////////////////////////////////

**$ git status**

On branch main

Your branch is up to date with 'origin/main'.

Changes to be committed:

(use "git restore --staged <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: README.md

Now (git add .)

(git add .) the ‘.’ Dot represents all files canbe added at the same time

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

**$ git add .**

///////////////////////////////////////////////////////////////////////////////////////////////////////////

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

**$ git status**

On branch main

Your branch is up to date with 'origin/main'.

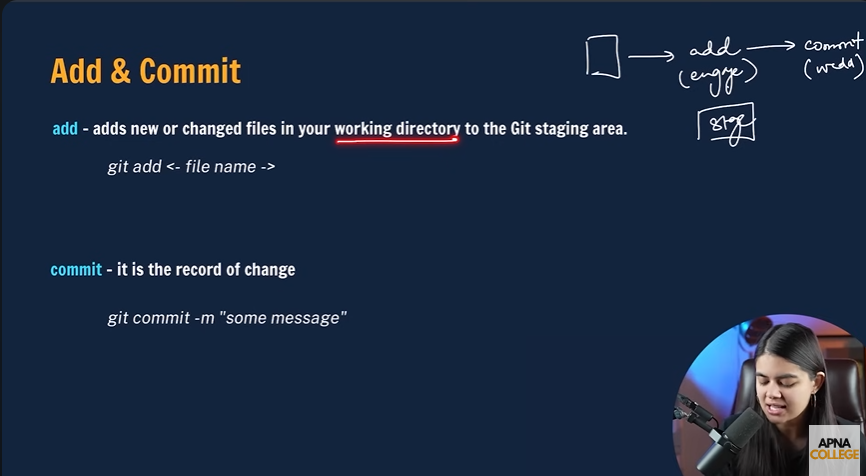
Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: README.md

new file: index.html

# After adding we commit



Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main)

**$ git commit -m "Added new file index.html with a paragraph,also changed modified read-me file"**

[main 260faf1] Added new file index.html with a paragraph,also changed modified read-me file

2 files changed, 14 insertions(+)

create mode 100644 index.html

we have just commit but no changes will be done to original file on githup

there is one more step we need push

# Push to see changes in Github

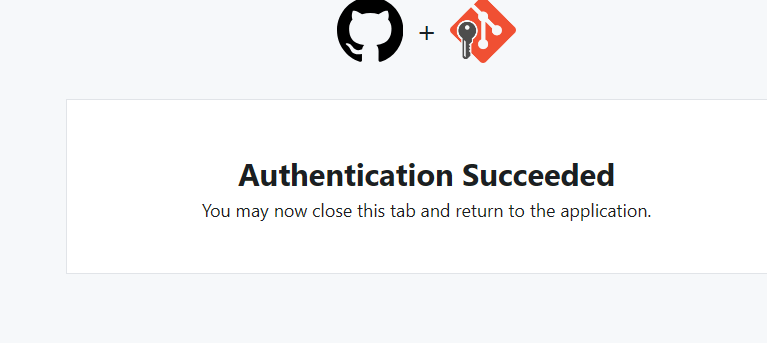


When we use this command for the first time ,github as for permission to authorize..just login when it asks and authorize…it will only ask for first time…or may be occationally

To Normalize above command to type less

Just use git push -u origin main //-u defines we will be using same origin for long term

From next step just use git push//just this after running previous command

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/gitProjectdemo (main) 

Now the the canges are pushed from local machine to remote machine

i.e is my computer to to github

$ git push origin main //main is main branch i.e is remote to original

info: please complete authentication in your browser...

Enumerating objects: 6, done.

Counting objects: 100% (6/6), done.

Delta compression using up to 8 threads

Compressing objects: 100% (4/4), done.

Writing objects: 100% (4/4), 561 bytes | 561.00 KiB/s, done.

Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

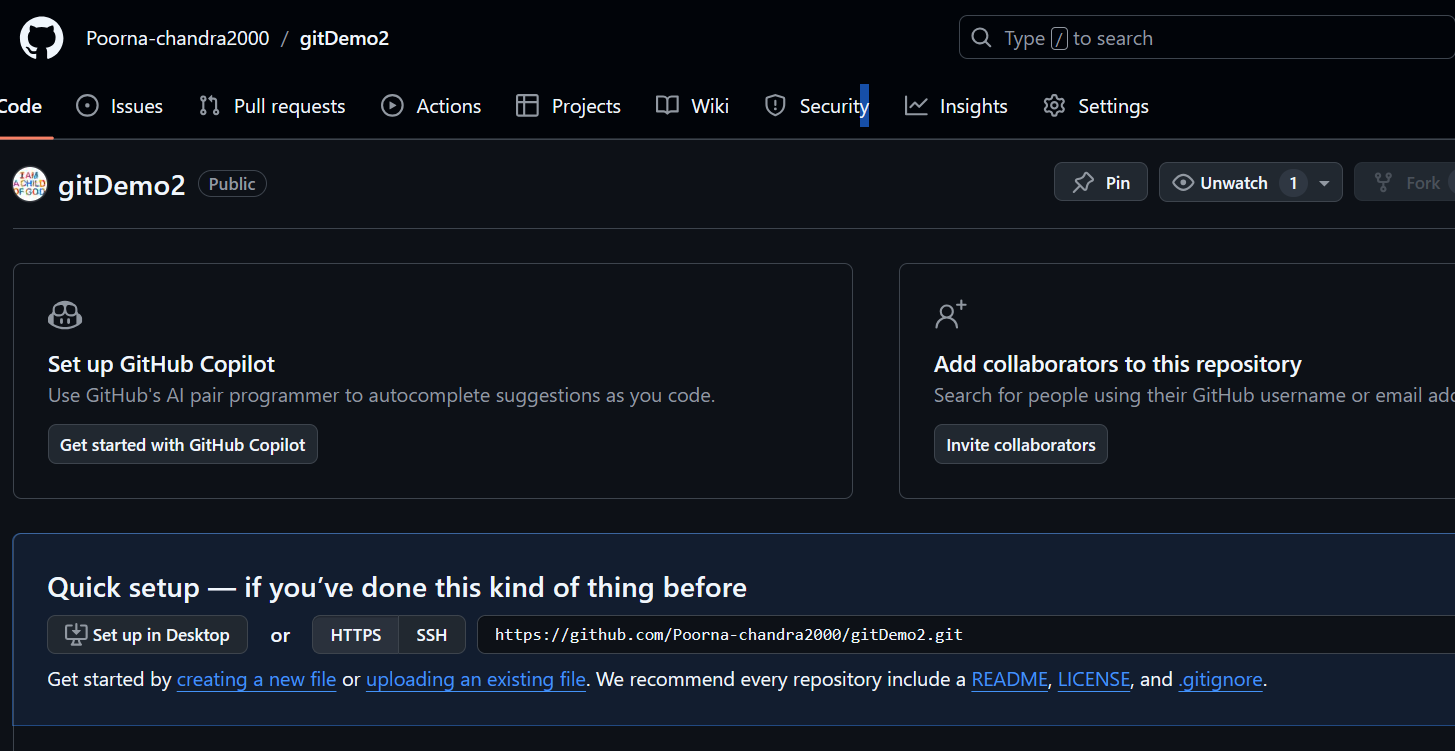
To https://github.com/Poorna-chandra2000/gitProjectdemo.git

14920bb..260faf1 main -> main

# WHAT IF NEW PROJECT IS CREATED ON LOCAL MACHINE AND YOU NEED TO PUSH THAT TO GITHUB REPOSITORY

**Step1 :create repository and keep its empty don’t add any read me file as we have to clone first and do stuffs:**

Just create repository and keep it empty





**Step2:Create a new Folder with new files on a Local Machine**

Using following commands on ScreenShot

Admin@MyLaptop MINGW64 ~/Desktop/git Demo

$ ../

bash: ../: Is a directory

Admin@MyLaptop MINGW64 ~/Desktop/git Demo

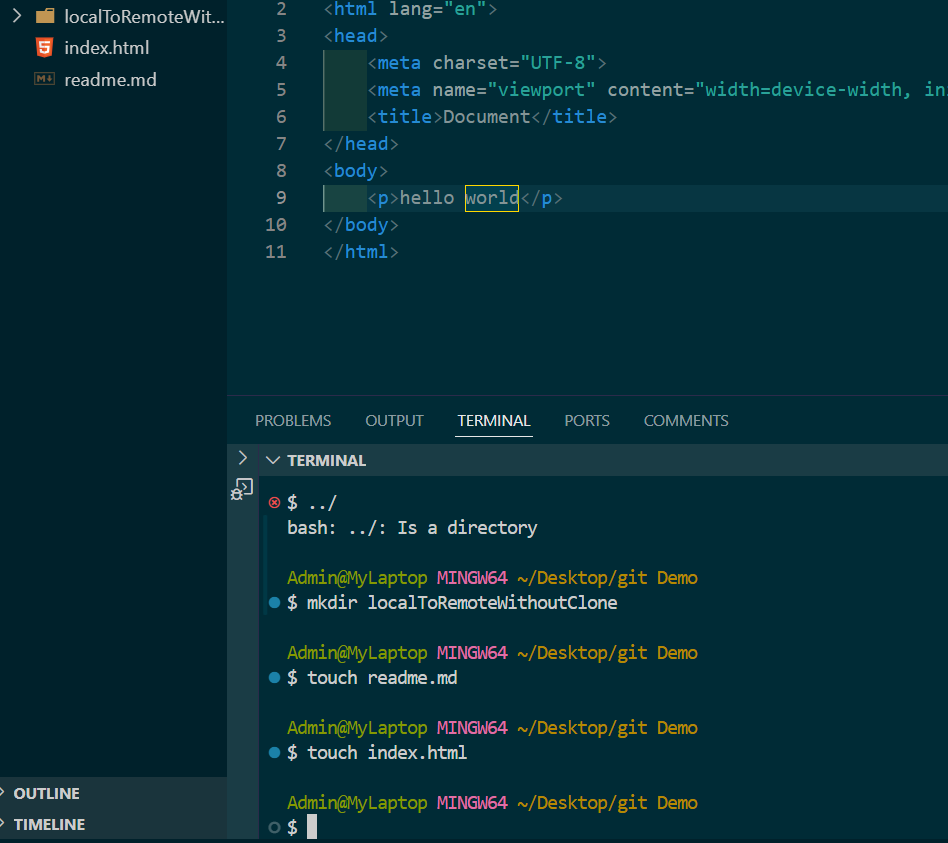
$ mkdir localToRemoteWithoutClone

Admin@MyLaptop MINGW64 ~/Desktop/git Demo

$ touch readme.md

Admin@MyLaptop MINGW64 ~/Desktop/git Demo

$ touch index.html



*Step 3:lets start pushing the new files that we created to empty newly created repository from local machine without cloning*



***Step one :perform these commands on Local machine within directory and commit on local machine***

Same exact steps to be followed

$ **git init**

Initialized empty Git repository in C:/Users/Admin/Desktop/git Demo/.git/

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (master)

$ **git status**

On branch master

No commits yet

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (master)

$ **git add .**

warning: adding embedded git repository: gitProjectdemo

hint: You've added another git repository inside your current repository.

hint: Clones of the outer repository will not contain the contents of

hint: See "git help submodule" for more information.

hint: Disable this message with "git config advice.addEmbeddedRepo false"

warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (master)

$ **git commit -m "Add initial files"**

[master (root-commit) 1a9e7f1] Add initial files

3 files changed, 12 insertions(+)

create mode 160000 gitProjectdemo

create mode 100644 index.html

create mode 100644 readme.md

***Step two:Lets start uploading the files to Remote Machine i.e GitHub repository that was empty***

1. **…or create a new repository on the command line**

echo "# gitDemo2" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/Poorna-chandra2000/gitDemo2.git

git push -u origin main

1. **…or push an existing repository from the command line**

git remote add origin https://github.com/Poorna-chandra2000/gitDemo2.git

git branch -M main

git push -u origin main

now since we have already commited

git branch -M main

git remote add origin https://github.com/Poorna-chandra2000/gitDemo2.git

git push -u origin main

***Now if you modify things again after pushing(Existing)***

***Add->commit->set remote origin->branch-M main->git push(if you previously used -u else git push -u origin main)***

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/localToRemoteWithoutClone (main)

$ git add .

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/localToRemoteWithoutClone (main)

$ git commit -m "new commint"

On branch main

Your branch is up to date with 'origin/main'.

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/localToRemoteWithoutClone (main)

git remote add origin <https://github.com/Poorna-chandra2000/gitDemo2.git>

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/localToRemoteWithoutClone (main)

$ git branch -M main

Admin@MyLaptop MINGW64 ~/Desktop/git Demo/localToRemoteWithoutClone (main)

git push -u origin main

for exsiting only

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (main)

$ **git add index.html**

warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (main)

$ **git add readme.md**

warning: in the working copy of 'readme.md', LF will be replaced by CRLF the next time Git touches it

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (main)

**$ git commit -m "modified files"**

[main 47db7bd] modified files

2 files changed, 2 insertions(+)

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (main)

**$ git remote -v**

origin https://github.com/Poorna-chandra2000/gitDemo2.git (fetch)

origin https://github.com/Poorna-chandra2000/gitDemo2.git (push)

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (main)

$ **git push origin main**

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 8 threads

Compressing objects: 100% (4/4), done.

Writing objects: 100% (4/4), 497 bytes | 497.00 KiB/s, done.

Total 4 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

To https://github.com/Poorna-chandra2000/gitDemo2.git

1a9e7f1..47db7bd main -> main

Admin@MyLaptop MINGW64 ~/Desktop/git Demo (main)

* + Make sure this directory contains files, and then add them individually using:

bash

Copy code

git add localToRemoteWithoutClone/\*

* + Alternatively, if you don't need to include this directory, you can remove it from the tracking list:

bash

Copy code

git rm -r --cached localToRemoteWithoutClone/

1. **Staging Files for Commit**:
   * The message Changes not staged for commit indicates that you've modified the index.html file, but you haven't staged it for commit.
   * Stage the modified file using:

bash

Copy code

git add index.html

* + Then commit the changes:

bash

Copy code

git commit -m "Updated index.html"

1. **Remote Already Exists**:
   * The error remote origin already exists occurs because you've already added a remote repository named origin.
   * To check your remotes, use:

bash

Copy code

git remote -v

* + If you need to change the remote URL, you can update it using:

bash

Copy code

git remote set-url origin https://github.com/Poorna-chandra2000/gitDemo2.git

1. **Pushing Changes**:
   * After staging and committing the changes, push them to your remote repository using:

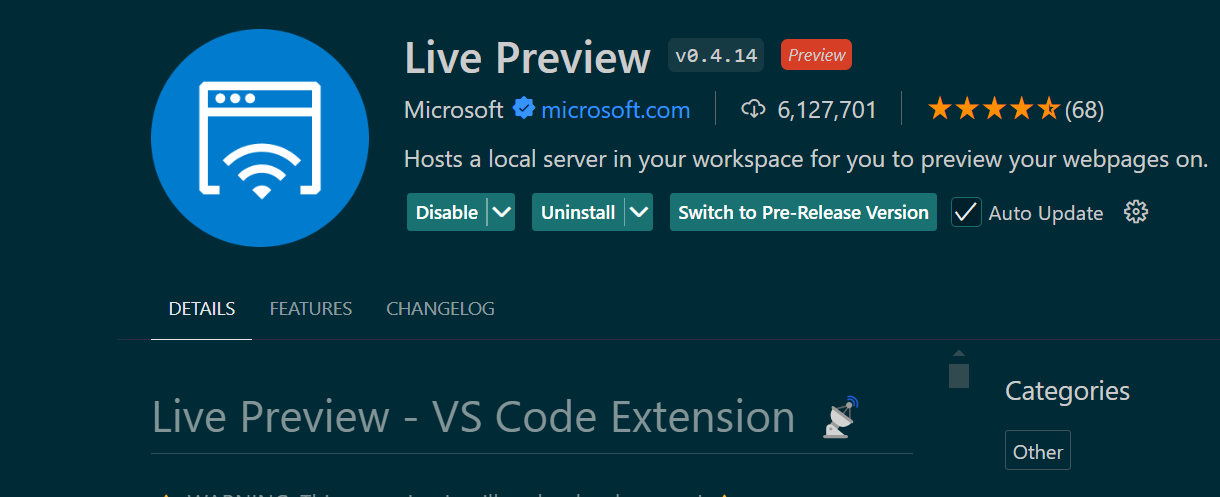
bash

Copy code

git push origin main

If you follow these steps, you should be able to resolve the issues and successfully push your changes. Let me know if you encounter any further problems!

### Vs Code Setup



**Install these VSCode Extensions**

Note, if the extensions are not installed, no problem, these are optional and would only help in your coding

* Code Snippets: [**VS Code JavaScript (ES6) snippets**](https://marketplace.visualstudio.com/items?itemName=xabikos.JavaScriptSnippets)
* Syntax Highlighting: [**Babel JavaScript**](https://marketplace.visualstudio.com/items?itemName=mgmcdermott.vscode-language-babel)Linter Extensions: *Linting is the process of running a program that analyses the code for potential errors.* Linter programs automatically find basic mistakes and tell you where they are and how to fix them. One of the most used by the VSCode community is:
* [**ESLint**](https://marketplace.visualstudio.com/items?itemName=dbaeumer.vscode-eslint).
* Node Extensions
  + [**npm**](https://marketplace.visualstudio.com/items?itemName=eg2.vscode-npm-script)
  + [**npm Intellisense**](https://marketplace.visualstudio.com/items?itemName=christian-kohler.npm-intellisense)
  + [**Path Intellisense**](https://marketplace.visualstudio.com/items?itemName=christian-kohler.path-intellisense)
* Formatting (There are other extensions for formatting, like Beautify, but we recommend installing this one)
  + [**Prettier**](https://marketplace.visualstudio.com/items?itemName=esbenp.prettier-vscode)
* One essential extension you will need for the first part of the course is Live Server. *This will allow you to easily see your webpage in the browser and all your changes will be automatically updated there*.
  + [**Live Server**](https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer)
* Eventually in your code you will be nesting statements and using lots of brackets and parenthesis. This extension will bring some color so you can easily identify the matching ones.
  + [**Bracket Pair Colorizer**](https://marketplace.visualstudio.com/items?itemName=CoenraadS.bracket-pair-colorizer-2)

To see additional extensions, which you don't need now, but you might want to install later, follow this [**link**](https://www.sitepoint.com/vs-code-extensions-javascript-developers/).

o make ls work in Visual Studio Code (VS Code) on Windows, you'll need to use a shell that supports Unix-like commands, such as **Git Bash** or **Windows Subsystem for Linux (WSL)**. Here's how to set it up:

# Option 1: Using Git Bash in VS Code

1. **Install Git Bash**:
   * If you haven't already, download and install [Git for Windows](https://git-scm.com/download/win). It comes with Git Bash, which supports ls and other Unix commands.
2. **Configure VS Code to Use Git Bash**:
   * Open VS Code.
   * Press Ctrl + Shift + P to open the Command Palette.
   * Type Terminal: Select Default Profile and select it.
   * From the list, choose **Git Bash**.
   * Now, when you open a new terminal in VS Code (using Ctrl + ), it will default to Git Bash, and you can use ls` to list files and directories.

Let me explain this step-by-step:

1. **1. Open Visual Studio Code**

* Start by opening Visual Studio Code (VS Code) on your computer.

1. **2. Open the Command Palette**

* Press the following keys together: Ctrl + Shift + P (on Windows/Linux) or Cmd + Shift + P (on macOS).
* This will open the **Command Palette** at the top of the VS Code window. The Command Palette is a quick way to access various commands and settings in VS Code.

1. **3. Type "Terminal: Select Default Profile"**

* Once the Command Palette is open, start typing Terminal: Select Default Profile.
* As you type, you'll see options related to your input appear.

1. **4. Select "Terminal: Select Default Profile"**

* When you see **"Terminal: Select Default Profile"** in the list of options, click on it or press Enter.

1. **5. Choose "Git Bash" (or another shell)**

* After selecting "Terminal: Select Default Profile," a list of available terminal profiles will appear.
* In this list, you should see options like **Command Prompt**, **PowerShell**, and **Git Bash** (if you have Git Bash installed).
* Click on **Git Bash** to set it as the default terminal.

1. **6. Open a New Terminal**

* Now, you can open a new terminal in VS Code by pressing Ctrl + ` (the backtick key).
* The terminal will open, and it should now be using **Git Bash**. You can confirm this by looking at the terminal prompt or typing ls to see if it lists your files.

1. **Option 2: Using Windows Subsystem for Linux (WSL) in VS Code**
2. **Install WSL**:
   * Open PowerShell as Administrator and run the following command:

bash

Copy code

wsl --install

* + Follow the instructions to install and set up your preferred Linux distribution.

1. **Configure VS Code to Use WSL**:
   * Install the [Remote - WSL](https://marketplace.visualstudio.com/items?itemName=ms-vscode-remote.remote-wsl) extension in VS Code.
   * Open a WSL terminal by clicking on the green remote indicator in the lower left corner of VS Code and selecting **New WSL Window**.
   * Now you can use ls and other Unix commands within this terminal.
2. **Option 3: Using PowerShell with Unix-Like Commands**
3. **Install Windows PowerShell Core (Optional)**:
   * Install [PowerShell Core](https://github.com/PowerShell/PowerShell) if you want an improved experience over the built-in PowerShell.
4. **Use ls in PowerShell**:
   * By default, PowerShell supports ls as an alias for Get-ChildItem. You can simply open a terminal in VS Code and use ls:

bash

Copy code

ls

1. **Summary:**

* **Git Bash**: Offers a Unix-like shell in Windows.
* **WSL**: Provides a full Linux environment in Windows.
* **PowerShell**: Supports ls by default as an alias for listing files.

Choose the method that best fits your workflow!