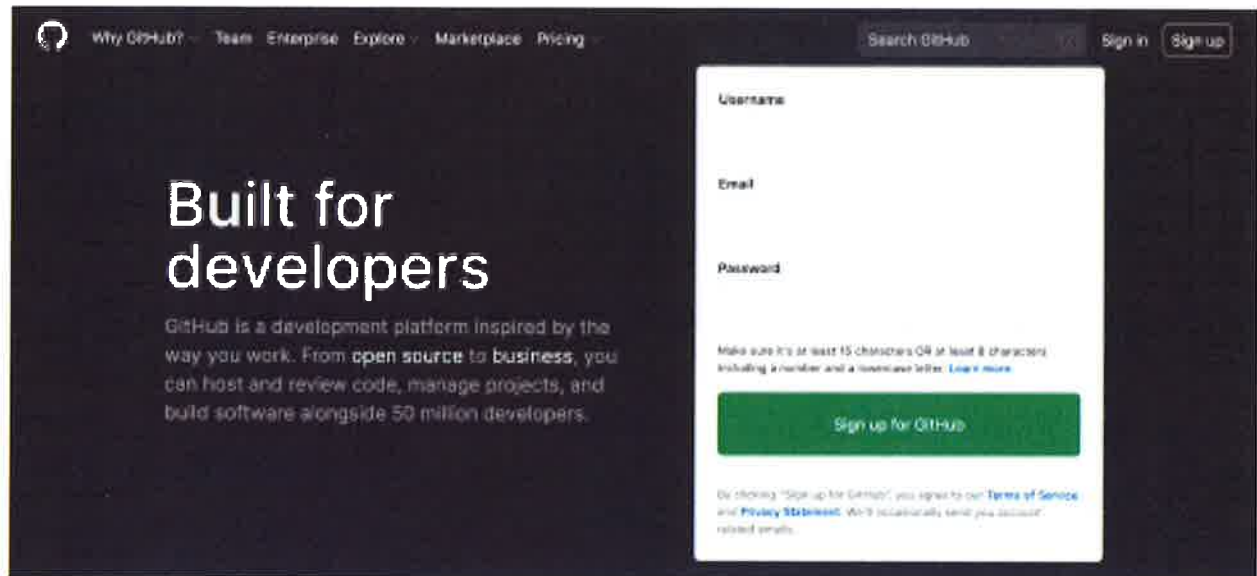
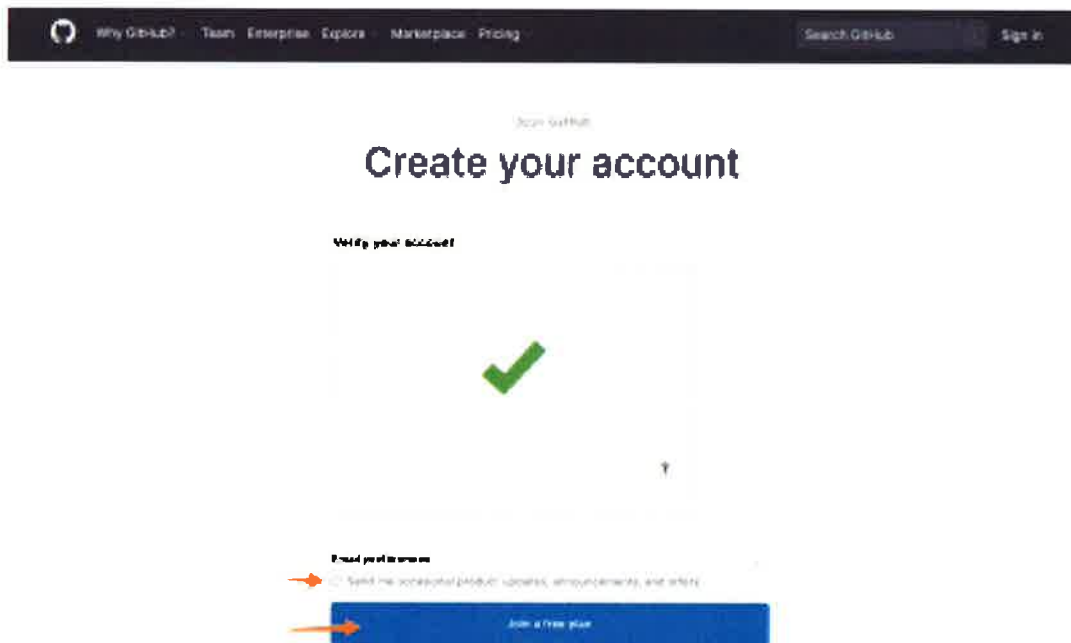


Step 1: Create a GitHub Account (it's free)



The screenshot shows the GitHub sign-up page. On the left, there is a dark blue header with the GitHub logo and navigation links: "Why GitHub?", "Team", "Enterprise", "Explore", "Marketplace", and "Pricing". Below the header, the text reads "Built for developers" in large white font, followed by a paragraph: "GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 50 million developers." On the right, there is a white sign-up form with fields for "Username", "Email", and "Password". Below the password field, there is a note: "Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more](#)". A green button labeled "Sign up for GitHub" is positioned below the form. At the bottom of the form, there is a small disclaimer: "By choosing 'Sign up for GitHub' you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account-related emails."

You will be asked to verify your account and choose whether you want them to send you occasional emails.

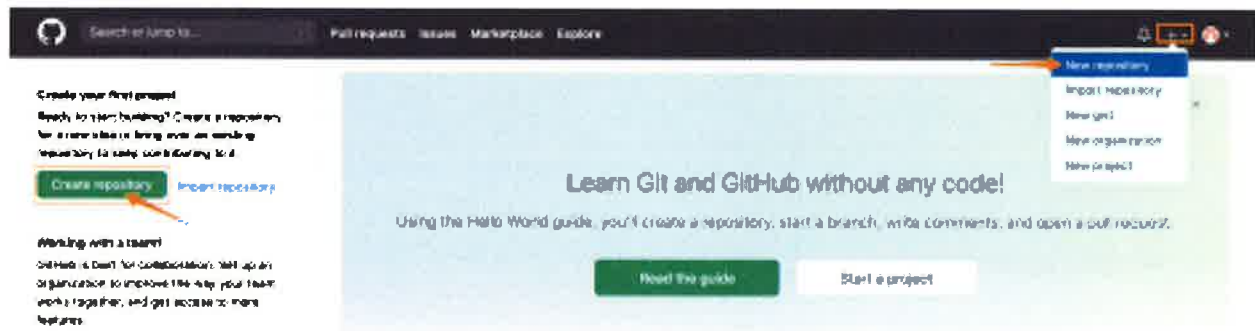


The screenshot shows the GitHub account verification page. At the top, there is a dark blue header with the GitHub logo and navigation links: "Why GitHub?", "Team", "Enterprise", "Explore", "Marketplace", and "Pricing". Below the header, the text reads "Create your account" in large black font. Underneath, there is a section titled "Verify your account" with a large green checkmark. Below the checkmark, there is a section titled "Final preferences" with two radio buttons. The first radio button is selected and has an orange arrow pointing to it; the text next to it is "Send me occasional GitHub updates, announcements, and alerts". The second radio button is unselected. Below the radio buttons, there is a blue button labeled "Join a free plan" with an orange arrow pointing to it.

You will also be asked a few additional questions to set up your account preferences, and you will need to verify your email address.

Once you have set up your account, you will be ready to create your first repository.

Step 2: Create a New Repository




In the name field enter “Demo” and then click the “Create Repository” button at the bottom.

Create a new repository

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

Owner * Repository name *

 jeffbeem2 - / Demo ✓

Great repository names are short and memorable. Need inspiration? How about expert-garbanzo?

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.

Initialize this repository with:

Skip this step if you're importing an existing repository.

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

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

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

Now we will follow the option to use the Command Line. You will do this on our Virtual Machine.

Quick setup — if you've done this kind of thing before

Setup in Docker [or](#) [HTTPS](#) [SSH](#) [https://github.com/jellibeer2/Demo.git](#)

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# Demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/jellibeer2/Demo.git
git push -u origin main
```

...or push an existing repository from the command line

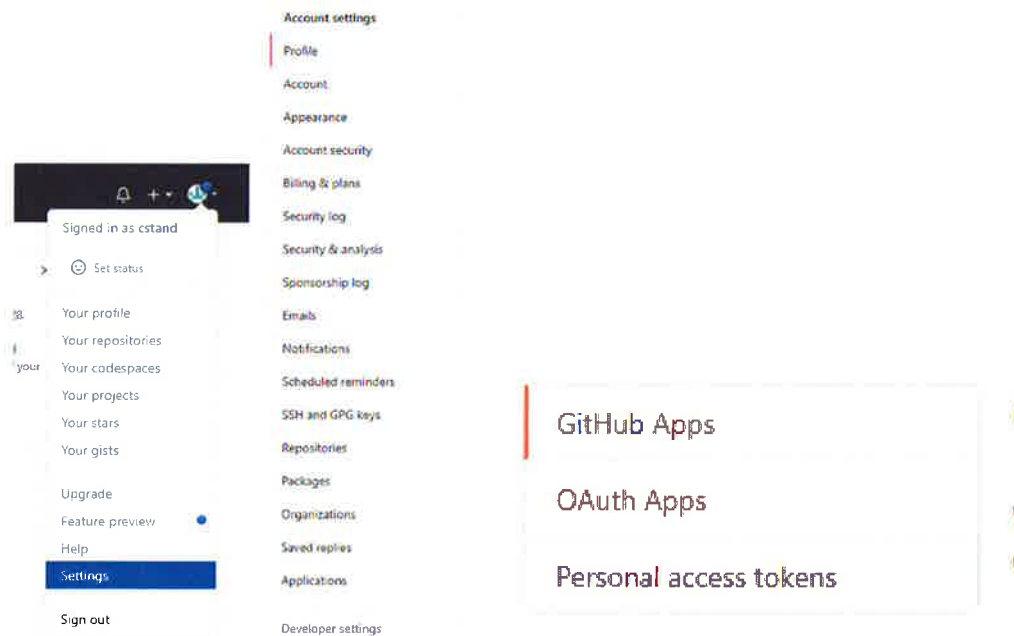
```
git remote add origin https://github.com/jellibeer2/Demo.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can import the repository with code from a Subversion, Mercurial, or TFS project.

Import code

You will need to create an access token to connect to github. Click on your icon and choose settings. Near the bottom, you should click on Developer Settings. Then choose Personal access tokens.



You need to create a token. This is only done once so you should copy your token and save it somewhere. (Note: if you forget your token, you can generate a new one. Also, connecting to Github won't be on the test)

The following screenshot shows the settings you can use to generate your token. Give your token a name (I chose it 1100). You will only need the token for 60 days. Make sure to select repo. Then at the bottom click to generate your new token. Make sure to save it somewhere.

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

IT 1100

What's this token for?

Expiration *

60 days



The token will expire on Fri, Dec 10 2021

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

- | | |
|---|--------------------------------------|
| <input checked="" type="checkbox"/> repo | Full control of private repositories |
| <input checked="" type="checkbox"/> repo:status | Access commit status |
| <input checked="" type="checkbox"/> repo_deployment | Access deployment status |
| <input checked="" type="checkbox"/> public_repo | Access public repositories |
| <input checked="" type="checkbox"/> repo:invite | Access repository invitations |
| <input checked="" type="checkbox"/> security_events | Read and write security events |

Do the following instructions on your CLI

You will need to install git on your CLI. Remember to update first.

```
sudo apt update
sudo apt install git
```

Create a new directory in your home folder on your CLI Virtual Machine called Demo and then move into your Demo directory.

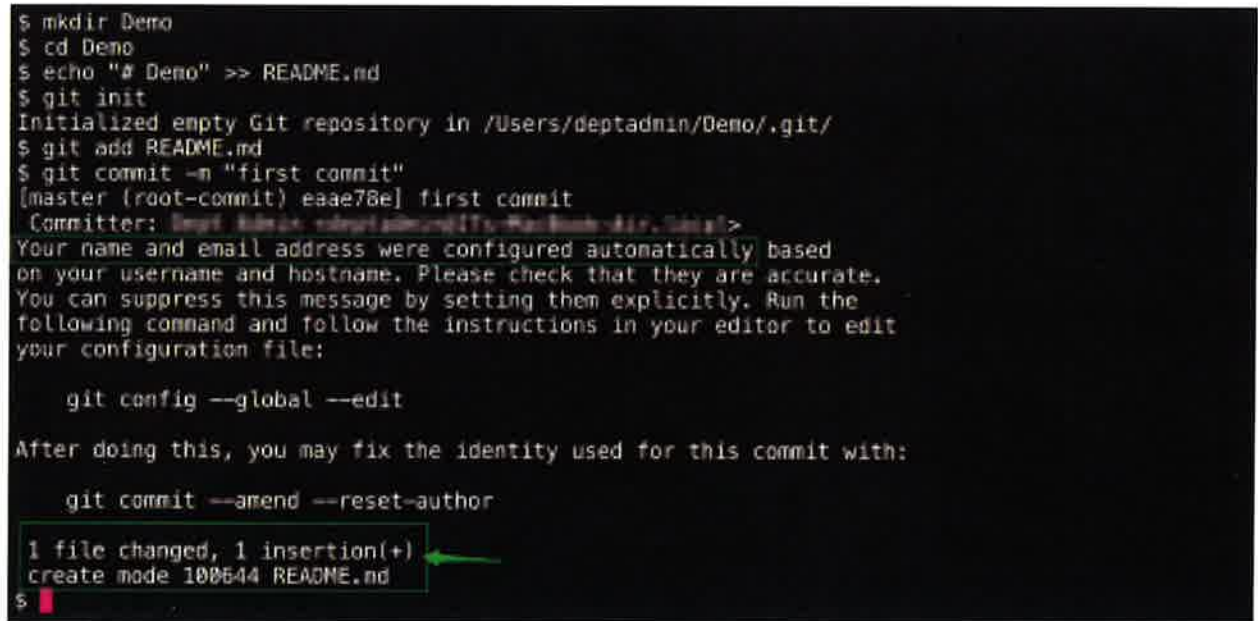
...or create a new repository on the command line

```
echo "# Demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/jellibeen2/Demo.git
git push -u origin main
```



```
$ mkdir Demo
$ cd Demo
$
```

Once inside the Demo directory, run each command found in the GitHub instructions, one line at a time, until you reach the `git commit -m "first commit"` command.



```
$ mkdir Demo
$ cd Demo
$ echo "# Demo" >> README.md
$ git init
Initialized empty Git repository in /Users/deptadmin/Demo/.git/
$ git add README.md
$ git commit -m "first commit"
[master (root-commit) ea4e78e] first commit
Committer: dept admin <deptadmin@172.16.17.101>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+)
create mode 100644 README.md
$
```

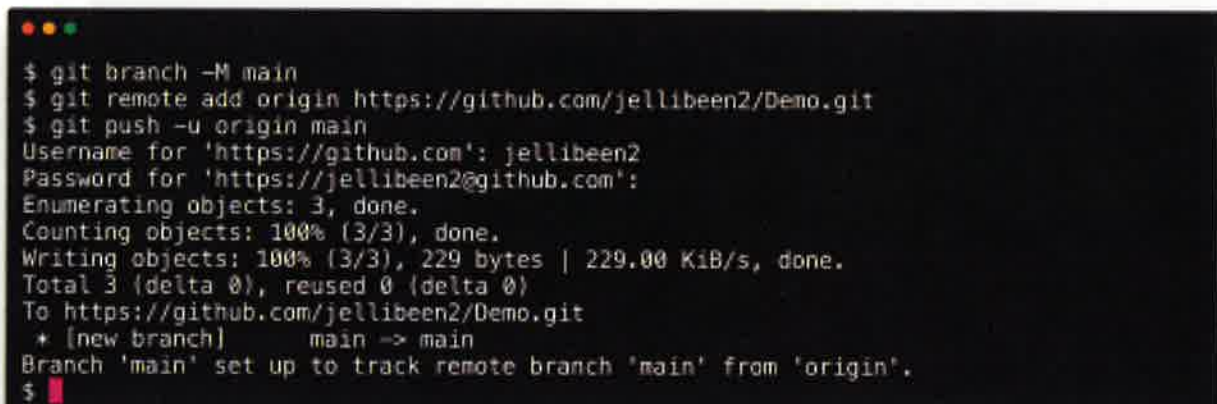
If you do not see this message, and instead get a message notifying you that your name and email address are not set, you will need to set them using the following two commands:

```
git config --global user.email "youremail@here"  
git config --global user.name "your github username"
```

Then continue running the remaining 3 commands.

You will replace `jellibeen2` with your username on github.

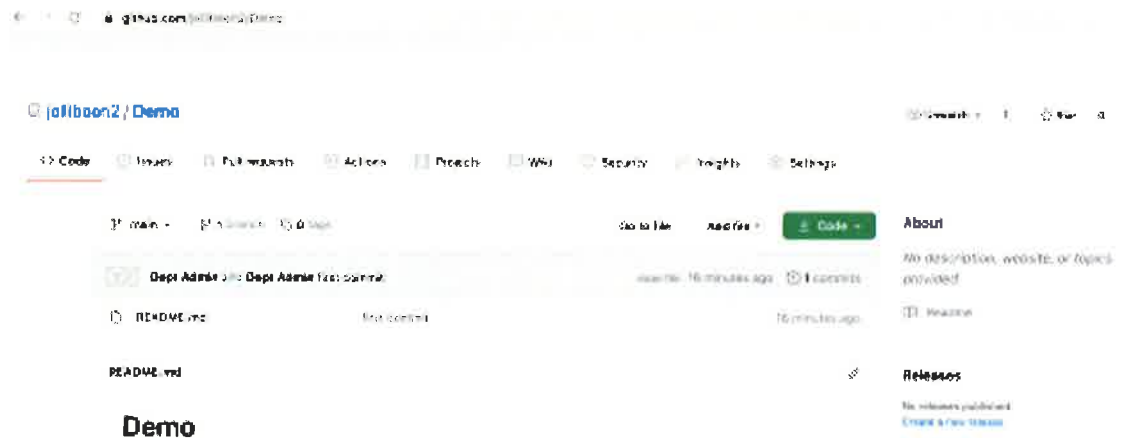
```
...or create a new repository on the command line  
  
echo "# Demo" >> README.md  
git init  
git add README.md  
git commit -m "first commit"  
git branch -M main  
git remote add origin https://github.com/jellibeen2/Demo.git  
git push -u origin main
```



```
$ git branch -M main  
$ git remote add origin https://github.com/jellibeen2/Demo.git  
$ git push -u origin main  
Username for 'https://github.com': jellibeen2  
Password for 'https://jellibeen2@github.com':  
Enumerating objects: 3, done.  
Counting objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 229 bytes | 229.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0)  
To https://github.com/jellibeen2/Demo.git  
+ [new branch]      main -> main  
Branch 'main' set up to track remote branch 'main' from 'origin'.  
$
```

Once you have finished running all the commands listed in the GitHub instructions, you should be able to refresh the page and view your new repository.

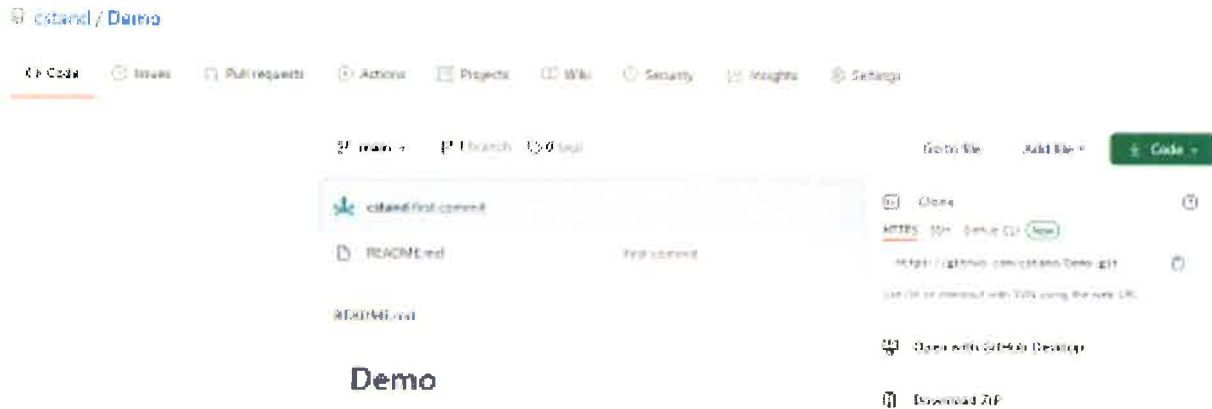
SCREENSHOT #1



PART II – Cloning

Step 1: Remove your Demo directory on your CLI. You may have to enter 'yes' a few times in order to remove it.

Step 2: Go to your Demo repository on github and click on the green download Code button.



Step 3: Copy the URL to the clipboard and then paste it into the following command (cstand should be your personal username):

```
git clone https://github.com/cstand/Demo.git
```

```
carol@carol-f20-GUI:~$ git clone https://github.com/cstand/Demo.git
Cloning into 'Demo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
carol@carol-f20-GUI:~$
```

Step 4: We are going to modify the files and then resubmit them to git hub. Do the following commands on your CLI.

```
cd ~/Demo
ls
cat README.md
echo "Added another line to README.md" >> README.md
cat README.md
git status
```

```
carol@carol-f20-GUI:~$ cd ~/Demo
carol@carol-f20-GUI:~/Demo$ ls
README.md
carol@carol-f20-GUI:~/Demo$ cat README.md
# Demo
carol@carol-f20-GUI:~/Demo$ echo "Added another line to README.md" >> README.md
carol@carol-f20-GUI:~/Demo$ cat README.md
# Demo
Added another line to README.md
carol@carol-f20-GUI:~/Demo$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
carol@carol-f20-GUI:~/Demo$
```


Step 5: Learn some terminology. Important terms will be underlined and highlighted in Yellow. Look in the above image for the following text.

Changes not staged for commit:

This tells you that the file listed in red has been changed but has not been staged. You stage a file by adding it using git add.

Before we stage it, lets see what has changed. We do this with the git diff command. git diff

```
carol@carol-f20-GUI:~/Demo$ git diff
diff --git a/README.md b/README.md
index 0805455..3bd615d 100644
--- a/README.md
+++ b/README.md
@@ -1,2 @@
 # Demo
+Added another line to README.md
carol@carol-f20-GUI:~/Demo$
```

This reports that you added a newline to README.md since the last time this was synced up to github. Now we want to stage the updated README.md file using git add. git add README.md

Step 6: Check the status of your repository using git status.

git status

```
carol@carol-f20-GUI:~/Demo$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   README.md

carol@carol-f20-GUI:~/Demo$
```

Look in the screenshot for the following words: Changes to be committed

We first **stage** and then **commit**. We will do that using the following commit command:
`git commit -m "Updated Readme file"`

The words in the quotes describe what changes were made between the last commit and this commit. You can word it however you would like.

```
carol@carol-f20-GUI:~/Demo$ git commit -m "Updated Readme file"
[main 293f0b0] Updated Readme file
 1 file changed, 1 insertion(+)
carol@carol-f20-GUI:~/Demo$
```

Step 7: Finally, we need to send these changes to github. We do this with a **push** command.
`git push -u origin main`

You should see the following in your CLI.

```
carol@carol-f20-GUI:~/Demo$ git push -u origin main
Username for 'https://github.com': cstand
Password for 'https://cstand@github.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 284 bytes | 284.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/cstand/Demo.git
   b74bbbc..293f0b0  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
carol@carol-f20-GUI:~/Demo$
```

Refresh your github page to see the changes reflected there.

cstand / Demo

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags

Go to file Add file + Code

cstand Updated Readme file 293f0b0 5 minutes ago 2 commits 5 minutes ago

README.md Updated Readme file

README.md

Demo

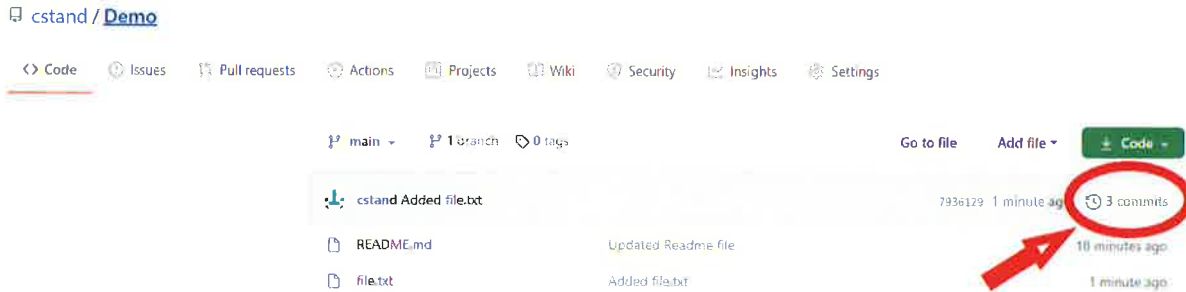
Added another line to README.md

Notice that it shows 2 commits.

Step 8: Add a new file to Git. Do the following in your CLI.

```
echo "This is a new file" >> file.txt
cat file.txt
git status
```

Now repeat the steps we did on steps 5-7: stage, commit, and then push. Your github account should then be:



Notice that there are now 3 commits.

Step 9: Delete a file. Suppose that file.txt has an error so we want to delete it. Delete it on your CLI and then check your git status.

```
carol@carol-f20-GUI:~/Demo$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

       deleted:    file.txt

no changes added to commit (use "git add" and/or "git commit -a")
carol@carol-f20-GUI:~/Demo$
```

This looks similar to the one when we added file.txt. We will need to go through the same commands to stage, commit, and push as we did when we added the file.

```
carol@carol-f20-GUI:~/Demo$ git add file.txt
carol@carol-f20-GUI:~/Demo$ git commit -m "Removed file.txt"
[main bf4a43a] Removed file.txt
 1 file changed, 1 deletion(-)
 delete mode 100644 file.txt
carol@carol-f20-GUI:~/Demo$ git push -u origin main
Username for 'https://github.com': cstand
Password for 'https://cstand@github.com':
Counting objects: 2, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (1/1), done.
Writing objects: 100% (2/2), 237 bytes | 237.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0)
To https://github.com/cstand/Demo.git
   7936129..bf4a43a  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
carol@carol-f20-GUI:~/Demo$
```

Now check your github and take a screenshot. Make sure you show the whole git hub screen. It should have 4 commits.

SCREENSHOT #2

