

Week 2 : Git and GitHub

Assignment 5 : Initial Git Setup (Local only)

Task1 : Create a local Project Directory.

Step1: Create a directory with the name, 'CodeTrack'.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2]
$ mkdir CodeTrack
```

Step2 : Change the working directory to CodeTrack

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2]
$ cd CodeTrack
```

Step3: Initialize the git repository

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:     git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:     git branch -m <name>
Initialized empty Git repository in /home/cybercena/Desktop/DevOps_with_Cohort/week-2/CodeTrack/.git/
```

Step 4 : check if .git folder exists or not to confirm the repo was successfully initialized.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ ls -a
.  ..  .git
```

Task2 : configure git locally for CodeTrack

[Tip: If you'll push to GitHub and want to keep your email private, use GitHub's noreply email (e.g., 12345678+username@users.noreply.github.com).]

Step 1: Configure the username locally only for this repo.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git config --local user.name "cybercena"
```

Step 2: Configure the email locally only for this repo.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git config --local user.email "cybercena@users.noreply.github.com"
```

Task 3 : configure globally (optional, in case if you need to configure for all the repos)

you need to use ‘--global’ instead of ‘--local’ in same command as we used before to configure globally

Step 1: configure the username for all the repos on the system.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git config --global user.name "cybercena"
```

Step 2: Configure the email for global repos on the system.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git config --global user.email "cybercena@users.noreply.github.com"
```

Step 3 : check the configuration emails, and username for the github repo.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git config --global --list
user.name=cybercena
user.email=cybercena@users.noreply.github.com
```

Why do we choose local and global configuration?

Global configuration and local configuration in GitHub are used for different purposes. We can create a repository with both configuration methods. We can use local configuration to set ourselves as the owner for a specific repository only, and global configuration to set ourselves as the owner of all the repositories we create on the system.

Assignment 6 : Tracking and Staging Changes in CodeTrack Project

Task 1: Modify the content of Repo

Step 1: Navigate to the Repository for writing code

Step 2: Create index.html and style.css file using touch command.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ touch index.html style.css
```

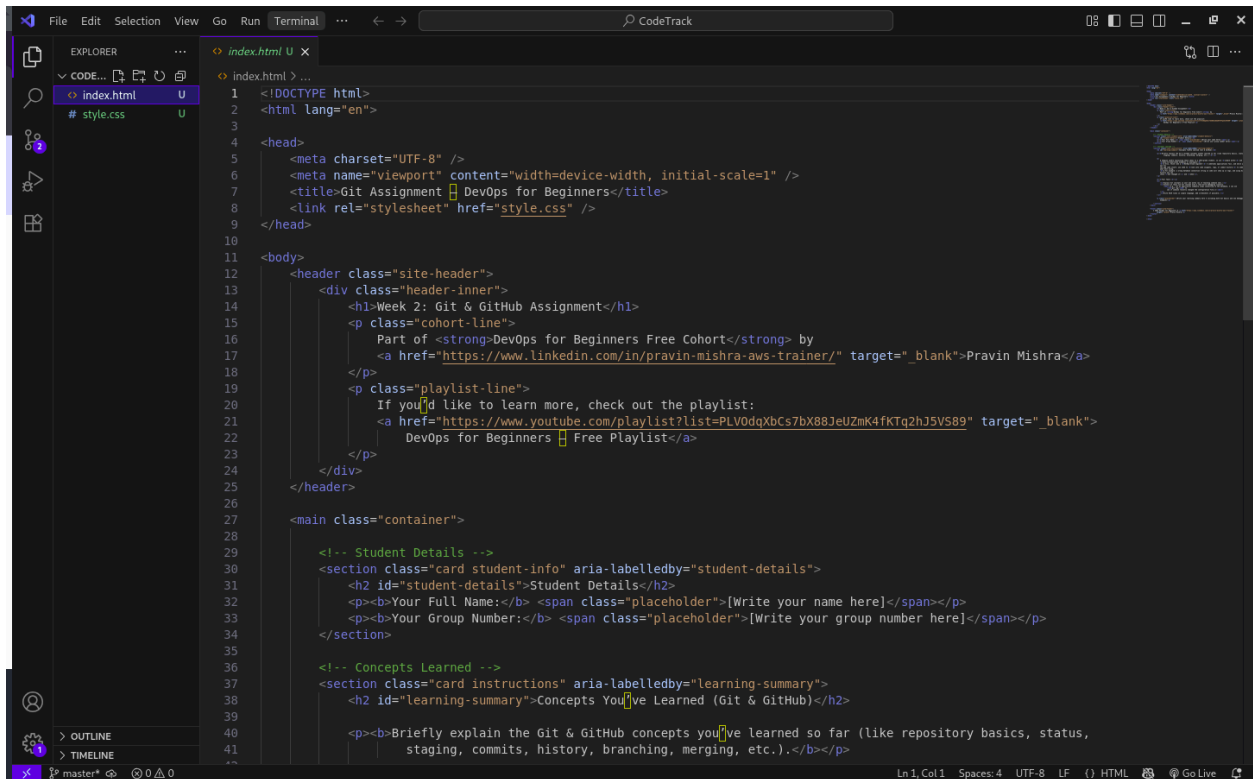
Step3: Copy the content of index.html and style.css from this repo :

<https://github.com/pravinmishraaws/Week-2---Git-GitHub-Assignment>

Step 3: Open VSCode to edit the content of files of the current repo as per the instructions.

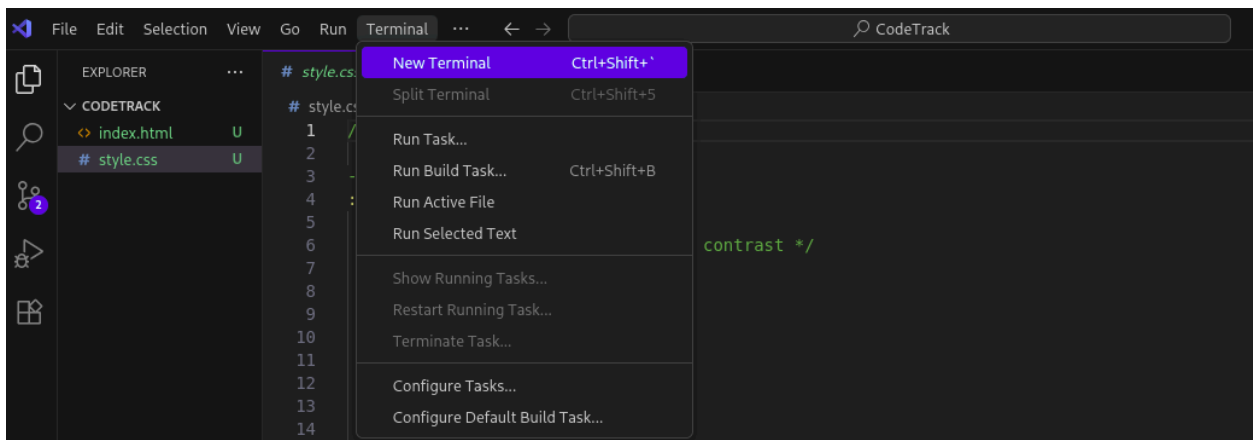
```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ code .
```

Step 4: Paste the code you copy from github repo to your local files.



```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8" />
6   <meta name="viewport" content="width=device-width, initial-scale=1" />
7   <title>Git Assignment | DevOps for Beginners</title>
8   <link rel="stylesheet" href="style.css" />
9 </head>
10
11 <body>
12   <header class="site-header">
13     <div class="header-inner">
14       <h1>Week 2: Git & GitHub Assignment</h1>
15       <p class="cohort-line">
16         Part of <strong>DevOps for Beginners Free Cohort</strong> by
17         <a href="https://www.linkedin.com/in/pravin-mishra-aws-trainer/" target="_blank">Pravin Mishra</a>
18       </p>
19       <p class="playlist-line">
20         If you'd like to learn more, check out the playlist:
21         <a href="https://www.youtube.com/playlist?list=PLV0dqXbCs7bX883eUzmK4fKTq2hJ5VS89" target="_blank">
22           DevOps for Beginners | Free Playlist</a>
23       </p>
24     </div>
25   </header>
26
27   <main class="container">
28
29     <!-- Student Details -->
30     <section class="card student-info" aria-labelledby="student-details">
31       <h2 id="student-details">Student Details</h2>
32       <p><b>Your Full Name:</b> <span class="placeholder">[Write your name here]</span></p>
33       <p><b>Your Group Number:</b> <span class="placeholder">[Write your group number here]</span></p>
34     </section>
35
36     <!-- Concepts Learned -->
37     <section class="card instructions" aria-labelledby="learning-summary">
38       <h2 id="learning-summary">Concepts You've Learned (Git & GitHub)</h2>
39
40       <p><b>Briefly explain the Git & GitHub concepts you've learned so far (like repository basics, status,
41         staging, commits, history, branching, merging, etc.).</b></p>
```

Step 5 : Open the inbuilt terminal of VSCode.



```
File Edit Selection View Go Run Terminal ...
# style.css
# style.css
1 /
2
3
4
5
6
7
8
9
10
11
12
13
14
```

Step 6 : Check the status of the local repo by using the ‘git status’ command.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git status
On branch master

No commits yet

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

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

(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$
```

Step 7: add all the untracked files by using the 'git add .' command. '.' represents the current working directory and it will add all the untracked files.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git add .
```

Step 8: check the status of files on the repository.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git status
On branch master

No commits yet

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

Step 9: Commit the changes with meaningful messages by using, 'git commit -m "messages"'

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git commit -m "Initial commit - Adding index.html and style.css"
[master (root-commit) 7f39437] Initial commit - Adding index.html and style.css
2 files changed, 301 insertions(+)
create mode 100644 index.html
create mode 100644 style.css
```

Step 10 : Verify the commit history

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git log --oneline
7f39437 (HEAD -> master) Initial commit - Adding index.html and style.css
```

Step 11: Open the index.html file on browser and modify the content of the file as per instructions.



Step 12: After modifying the contents, check the repository status.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ 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:   index.html

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

Step 13: Add the untracked files to repo.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git add index.html
```

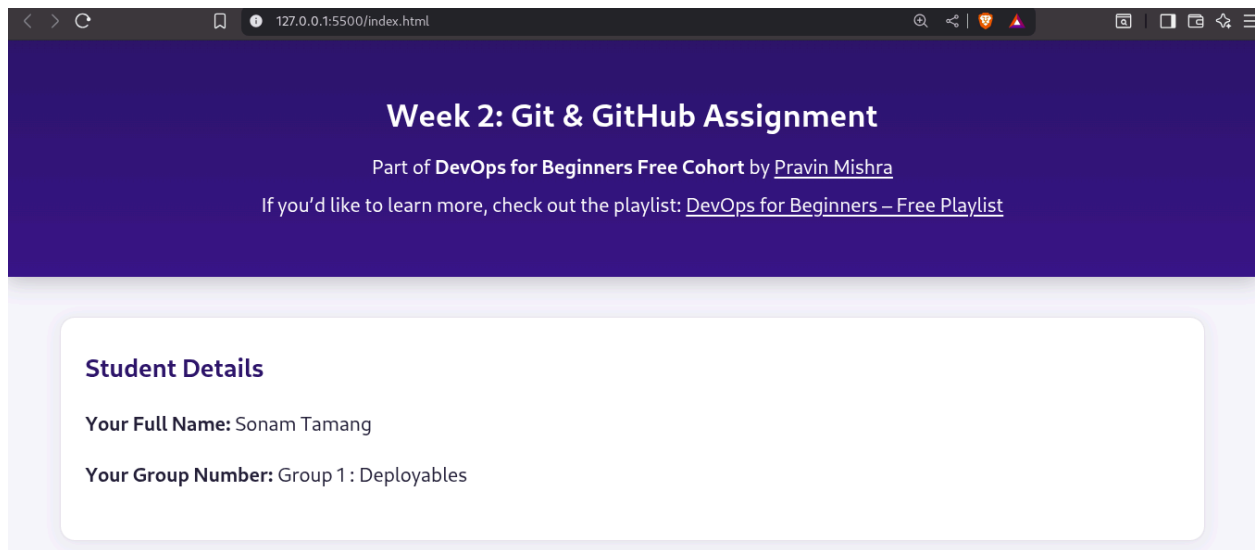
Step 14: Commit the files with meaningful messages.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git commit -m "updating the contents in index.html file"
[master 516843e] updating the contents in index.html file
1 file changed, 30 insertions(+), 3 deletions(-)
```

Step 15: verify the commit history (optional)

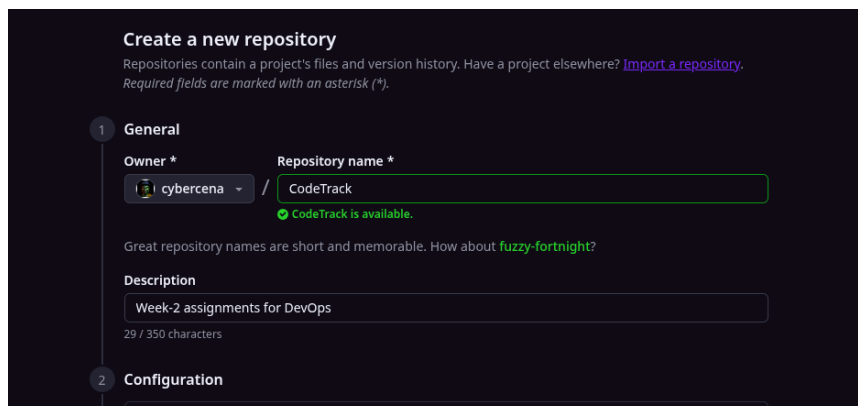
```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git log --oneline
516843e (HEAD -> master) updating the contents in index.html file
7f39437 Initial commit - Adding index.html and style.css
```

Step 15: verify the changes by opening index.html file.

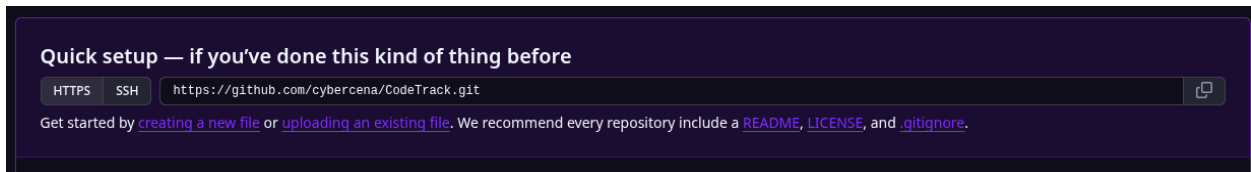


Let's convert our local repo to remote repo, we will make our repo remote by publishing it to Github.

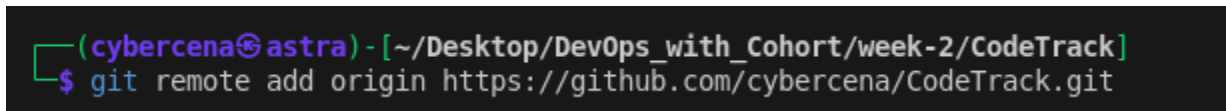
Step 1: create a Github account if you don't have one. If you already have a github account then log into the github and create a new repository with the project name.(make it publicly visible)



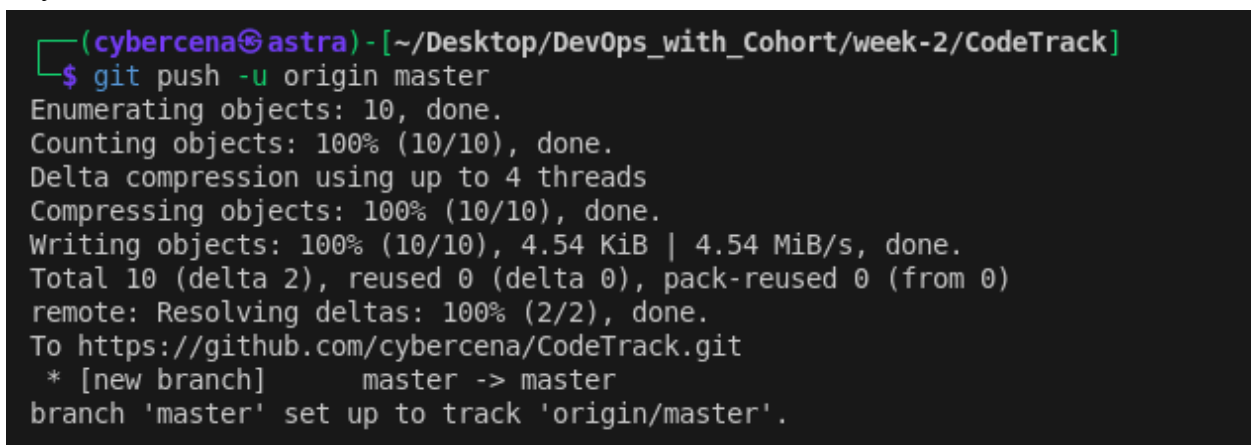
Step 2: copy the url for setup.



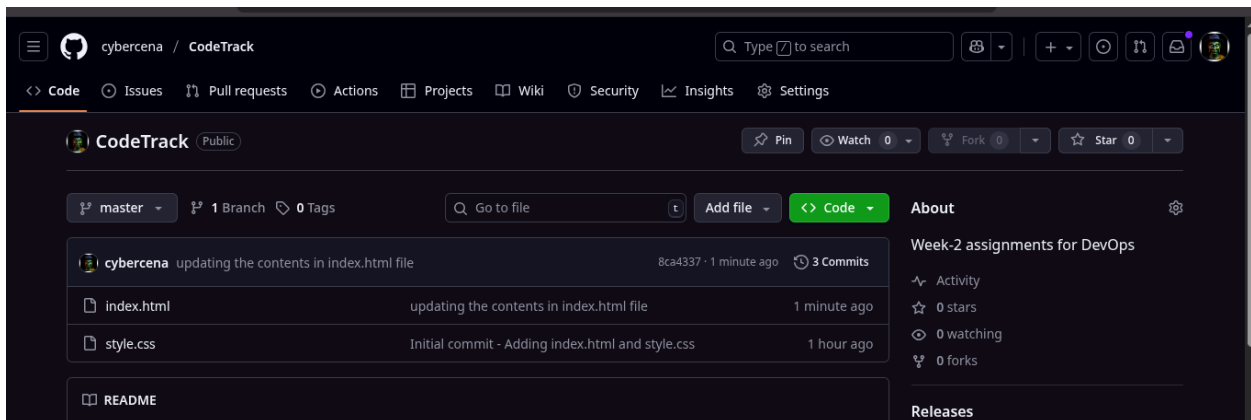
Step 3: Navigate to your local repo and run the command, 'git remote add origin <url>.



Step 4: push your code to github by using the command, 'git push -u origin <branchname>'. It may be main or master.



Step 5: Check the github repo , if the source codes are updated or not.



Now, we successfully uploaded our local repo to github.

Task 2 : Deploy the application on EC2 instance.

Step1 : launch EC2 instance for ubuntu machine.

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name
week2-linux [Add additional tags](#)

Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux macOS **Ubuntu** Windows Red Hat SUSE Linux Debian

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Summary

Number of instances [Info](#)
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.8.2...[read more](#)
ami-00ca32bbc84273381

Virtual server type (instance type)
t3.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

[Cancel](#) [Launch instance](#) [Preview code](#)

Step 2 : Generate a Private key

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.
week2
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

☒ RSA
RSA encrypted private and public key pair

☐ ED25519
ED25519 encrypted private and public key pair

Private key file format

☒ .pem
For use with OpenSSH

☐ .ppk
For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

[Cancel](#) [Create key pair](#)

Step 3 : Allow HTTP and HTTPS traffic to access web content and SSH to get remote access into the machine from our local machine using SSH protocol.

Firewall (security groups) | [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-1' with the following rules:

- ☒ Allow SSH traffic from
Helps you connect to your instance Anywhere
0.0.0.0/0
- ☒ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
- ☒ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

Step 4 : change the permission of private key to read only for owners and no permissions to groups and others.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2]
$ chmod 400 week2.pem
```

Step 5 : Go to SSH client and copy the command to access the machine.

aws | Search | [Alt+S] | United States (N. Virginia) | >sonam

EC2 > Instances > i-0346089fc2d1b970d > Connect to instance

Connect [Info](#)
Connect to an instance using the browser-based client.

EC2 Instance Connect | Session Manager | **SSH client** | EC2 serial console

Instance ID
i-0346089fc2d1b970d (week2-linux)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is week2.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
`chmod 400 "week2.pem"`
4. Connect to your instance using its Public DNS:
ec2-98-83-159-229.compute-1.amazonaws.com

Example:
`ssh -i "week2.pem" ubuntu@ec2-98-83-159-229.compute-1.amazonaws.com`

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Step 6: Access the machine by using SSH protocol with private key. Command : ssh -i <privatekey> username@hostname

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2]
$ ssh -i "week2.pem" ubuntu@ec2-98-83-159-229.compute-1.amazonaws.com
The authenticity of host 'ec2-98-83-159-229.compute-1.amazonaws.com (98.83.159.229)' can't be established.
ED25519 key fingerprint is SHA256:znc6bE9myb7rrnt/R/Fl/bPwZtJuuEdNJ1/UVgZQkN0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? █
```

Step 7 : check who you are by using 'whoami' command (optional)

```
ubuntu@ip-172-31-40-249:~$ whoami
ubuntu
ubuntu@ip-172-31-40-249:~$
```

Step 8: Update the system and Repository

```
ubuntu@ip-172-31-40-249:~$ sudo apt update
```

Step 9: install the nginx server

```
ubuntu@ip-172-31-40-249:~$ sudo apt install -y nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
Suggested packages:
  fcgiwrap nginx-doc ssl-cert
```

Step 10: Clone the github repo we created earlier ,<https://github.com/cybercena/CodeTrack.git>

```
ubuntu@ip-172-31-40-249:~$ git clone https://github.com/cybercena/CodeTrack.git
Cloning into 'CodeTrack'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10 (delta 2), reused 10 (delta 2), pack-reused 0 (from 0)
Receiving objects: 100% (10/10), 4.54 KiB | 1.51 MiB/s, done.
Resolving deltas: 100% (2/2), done.
ubuntu@ip-172-31-40-249:~$
```

Step 11: Start the NGINX server.

```
ubuntu@ip-172-31-40-249:~$ sudo systemctl start nginx
ubuntu@ip-172-31-40-249:~$
```

Step 12: Navigate to the CodeTrack folder and check the files. Change the ownership permission of the /var/www/html folder to www-data user and group.

```
ubuntu@ip-172-31-40-249:~$ ls
CodeTrack
ubuntu@ip-172-31-40-249:~$ cd CodeTrack
ubuntu@ip-172-31-40-249:~/CodeTrack$ ls
index.html  style.css
ubuntu@ip-172-31-40-249:~/CodeTrack$ sudo chown -R www-data:www-data /var/www/html
ubuntu@ip-172-31-40-249:~/CodeTrack$ sudo chmod -R 755 /var/www/html
ubuntu@ip-172-31-40-249:~/CodeTrack$
```

Step 13: Remove the content of /var/www/html and copy the content of CodeTrack to /var/www/html.

```
ubuntu@ip-172-31-40-249:~/CodeTrack$ cd /var/www/html
ubuntu@ip-172-31-40-249:/var/www/html$ ls
index.html  style.css
ubuntu@ip-172-31-40-249:/var/www/html$
```

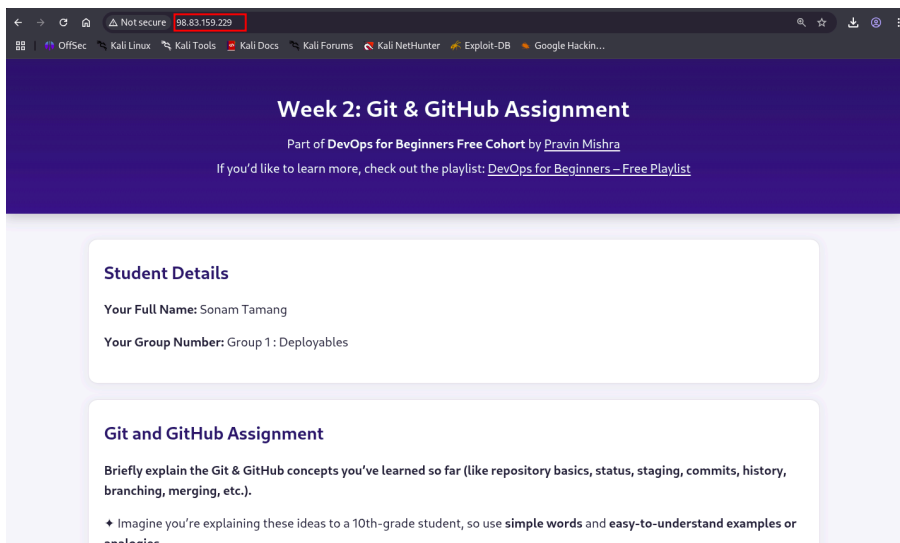
Step 14: Restart the NGINX server

```
ubuntu@ip-172-31-40-249:/var/www/html$ sudo systemctl restart nginx
ubuntu@ip-172-31-40-249:/var/www/html$
```

Step 15 : Get the Public IP address of EC2 instance by using , 'curl ifconfig.me'.

```
ubuntu@ip-172-31-40-249:/var/www/html$ curl ifconfig.me
98.83.159.229ubuntu@ip-172-31-40-249:/var/www/html$
```

Step 16 : Visit the ip address on your browser and you will get your site ready.



[Note: I have hosted this site using cloudflare on my subdomain, week2.sonam.info.np]

Assignment 7 : Branching workflow - Add & Verify a Contact Page

Step 1 : Navigate to your existing CodeTrack Project from the last assignment and check the repo status and branch we are working on.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git branch
* master

(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$
```

Step 2: Create and switch to a feature branch

Creating a new branch 'feature/contact-page'

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git branch feature/contact-page
```

Switch to 'feature/contact-page' branch from 'master' branch.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git checkout feature/contact-page
Switched to branch 'feature/contact-page'
```

Checking the current branch

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git branch
* feature/contact-page
  master
```

it shows we are on feature/contact-page branch

Step 3: Add contact.html in the current branch

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ nano contact.html
```

Writing the code inside Contact.html

```
GNU nano 8.4 contact.html *
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Contact - CodeTrack</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Contact Us</h1>
  <p>Email: ping@sonam.info.np</p>
  <p>Website: Sonam.info.np</p>
</body>
</html>
```

Stage and commit contact.html

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git add contact.html

(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git commit -m "feature(contact): add contact page with email and phone"
[feature/contact-page 645dba1] feature(contact): add contact page with email and phone
1 file changed, 13 insertions(+)
create mode 100644 contact.html
```

Step 4: Add a link to the contact page in index.html (work from same branch)

```
GNU nano 8.4 index.html *
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Git Assignment - DevOps for Beginners</title>
  <link rel="stylesheet" href="style.css" />
</head>
<body>
  <header class="site-header">
    <div class="header-inner">
      <h1>Week 2: Git & GitHub Assignment</h1>
      <p class="cohort-line">
        Part of <strong>DevOps for Beginners Free Cohort</strong> by
        <a href="https://www.linkedin.com/in/pravin-mishra-aws-trainer/" target="_blank">Pravin Mishra</a>
      </p>
      <p class="playlist-line">
        If you'd like to learn more, check out the playlist:
        <a href="https://www.youtube.com/playlist?list=PLV0dqXbCs7bX88JeUZmk4fKTq2hJ5VS89" target="_blank">
          DevOps for Beginners - Free Playlist</a>
      </p>
      <p class="playlist-line">
        Want to reach us? Visit the
        <a href="contact.html">Contact Page</a>.
      </p>
    </div>
  </header>
  <main class="container">
```

Saving index.html file and commit with messages

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git add index.html

(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git commit -m "feature(nav): add contact page link to index.html"
```

Step 5: Switch back to master or main branch

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
```

There is no contact.html file in the main branch.

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ ls
index.html  style.css
```

Step 6 : Merge the feature branch into main or master branch

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git merge feature/contact-page
Updating 8ca4337..2e66092
Fast-forward
 contact.html | 13 ++++++
 index.html   |  4 ++++
 2 files changed, 17 insertions(+)
 create mode 100644 contact.html
```

We can see the contact.html file after merging the feature/contact-page branch

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ ls
contact.html  index.html  style.css
```

Now, we can see the contact link on index.html ,

Week 2: Git & GitHub Assignment

Part of [DevOps for Beginners Free Cohort](#) by [Pravin Mishra](#)

If you'd like to learn more, check out the playlist: [DevOps for Beginners – Free Playlist](#)

Want to reach us? Visit the [Contact Page](#).

Student Details

Your Full Name: Sonam Tamang

Your Group Number: Group 1 : Deployables

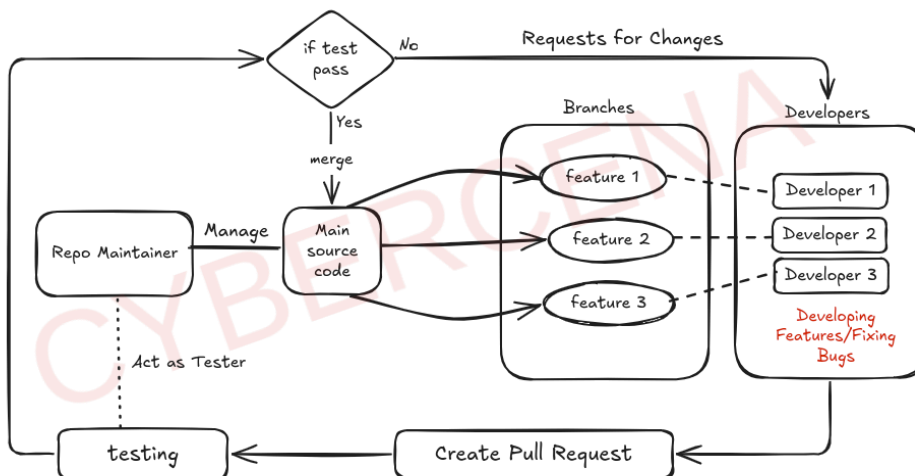
Step 7 : Inspect the history of repo

```
(cybercena@astra) - [~/Desktop/DevOps_with_Cohort/week-2/CodeTrack]
$ git log --oneline --graph --decorate --all
* 2e66092 (HEAD -> master, feature/contact-page) feature(nav): add contact page link to index.html
* 645dba1 feature(contact): add contact page with email and phone
* 8ca4337 (origin/master) updating the contents in index.html file
* 516843e updating the contents in index.html file
* 7f39437 Initial commit - Adding index.html and style.css
```

If you want to delete the branch you can use ‘git branch -d <branch-name>’.

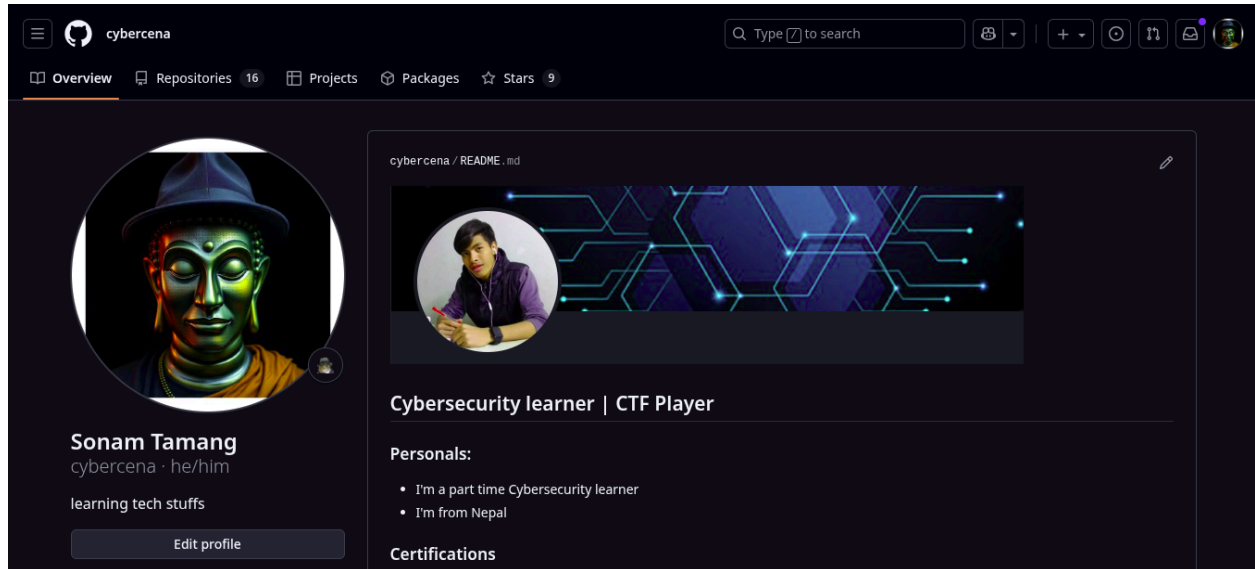
Explain why the link wasn’t visible before the merge and why it appears after.

The change code exists on the feature/contact-page not on the master branch. It will only appear if we merge the branch on the main source code. After we merged it, it appeared in the index.html file. You can see the github work flow from below image:

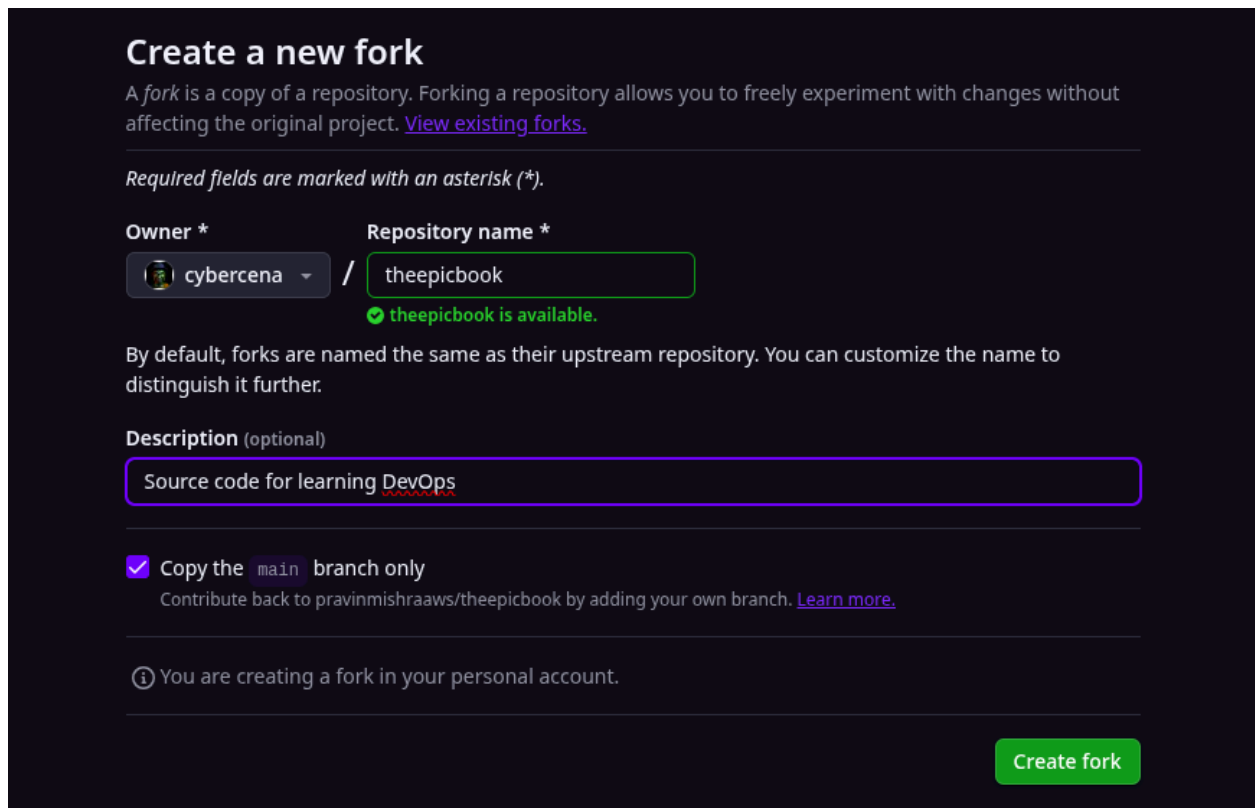


Assignment 8 : Setting up Github for CodeTrack

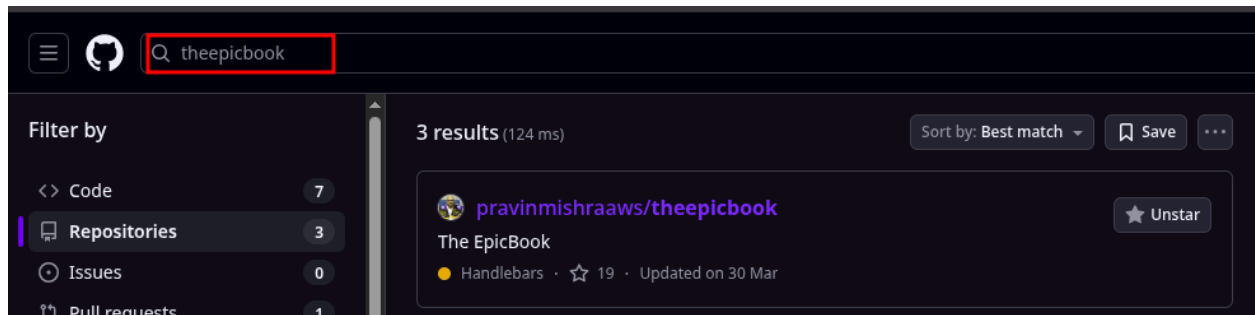
Task 1 : Create Github account and Setup Profile.



Task 2: Fork the repository you want to make your own copy



Task 3: Starred the repository that interests you.



[Note : I already have Github account so, i don't create and setup it again]

Why is it important to have a professional GitHub profile as a developer ?

A professional GitHub profile is important because it acts as a portfolio to showcase your coding skills and projects. It helps build credibility by demonstrating clean code, documentation, and consistent contributions. Employers often review GitHub profiles to evaluate a developer's experience and technical abilities. It also allows you to collaborate with others, contribute to open-source projects, and learn from real-world coding challenges. Overall, a strong GitHub presence increases your visibility and opportunities in the developer community.