***GIT and GIT HUB***

**A red and blue logo

Description automatically generated**

***Assessment Weightage & Type:***

***60% Portfolio Coursework***

***Year:2024***

* *Student Name:* ***Khushi Kumari***
* *UWE ID:* ***24030167***
* *Module code:* ***UFCFL1-15-0***
* *Level:* ***Level 3)***
* *Faculty:* ***Faculty of Environment & Technology***
* *Department:* ***FET Department of Computer Science & Creative***

***Technology***

**Getting Started with GIT- Local Setup**

**Introduction:**

This guide will walk you through setting up Git locally on your computer. Git is a version control system that allows you to track changes in your code or files over time. We'll cover downloading and installing Git, creating a local repository, and configuring your user information.

**Prerequisites**

* A computer with an internet connection

**Steps**

1. **Download Git:**
   * Open your web browser and navigate to the official Git downloads page: <https://www.git-scm.com/downloads>.
   * Download the latest version of Git for your operating system (Windows, macOS, or Linux). Choose the appropriate installer based on your system.

**Install Git:**

* Double-click the downloaded installer file and follow the on-screen instructions to complete the installation.
* **Windows users:** During installation, ensure you check the option to "Use Git from the Windows Command Prompt".

**Create a Folder and Open Git Bash:**

* Open your File Explorer (Windows) or Finder (macOS).
* Create a new folder on your Desktop and name it "L3".
* Right-click on the "L3" folder and select "Git Bash Here" from the context menu. This assumes Git is configured to use Bash as the default terminal emulator.



**Check Git Version:**

* In the Git Bash window, type the following command and press Enter:

git --version



This command will display the installed Git version on your system.

**Create Files:**

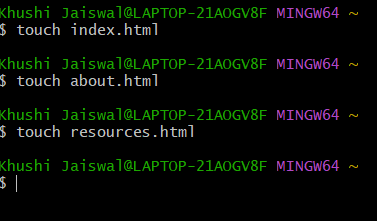
* In the Git Bash window, type the following commands one by one, pressing Enter after each:

Bash

touch index.html

touch about.html

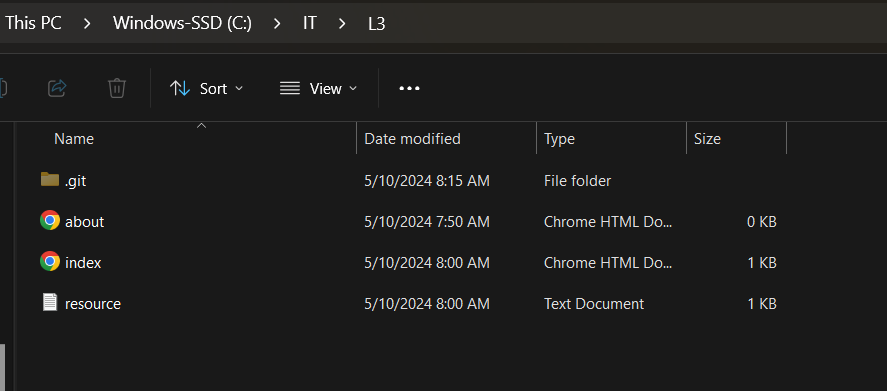
touch resources.txt



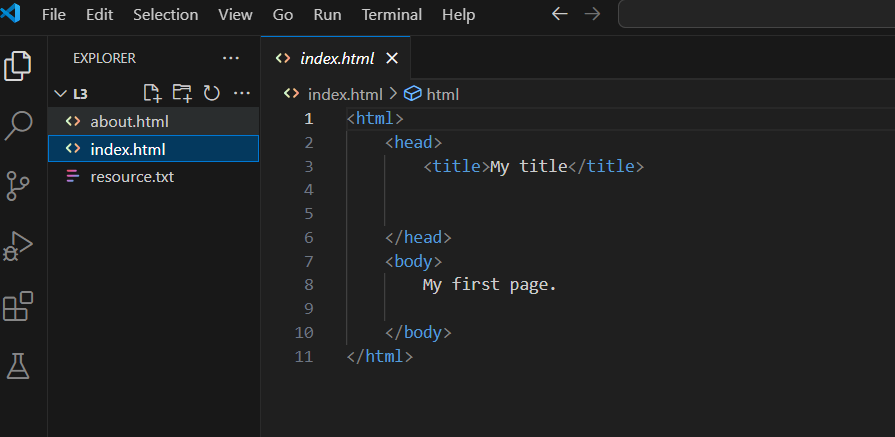
* These commands create three empty files (index.html, about.html, and resources.txt) inside your "L3" folder.

 **Verify Files:**

* Open your File Explorer (Windows) or Finder (macOS) and navigate to the "L3" folder. You should see the three files created in the previous step.

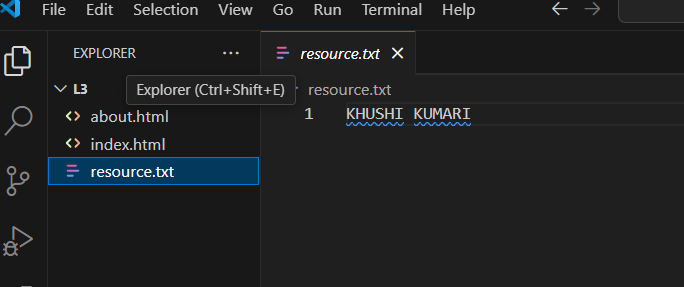


**Edit index.html:**

* Open "index.html" in a text editor like Notepad or Notepad++.
* Add the following line of code and save the file: ****

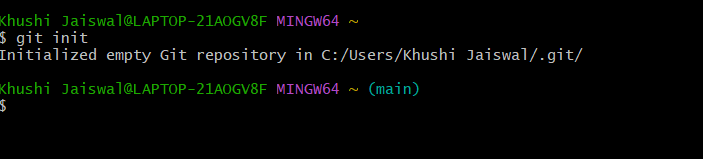
 **Edit resources.txt:**

* Open "resources.txt" in your text editor.
* Write your name and save the file.



**Initialize Git Repository:**

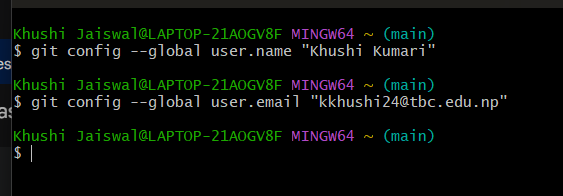
* In the Git Bash window, type the following command and press Enter:



This initializes a Git repository in your "L3" folder. You'll see a new folder named ".git" appear, which stores Git's internal data.

**10. Configure Username and Email:**

* In the Git Bash window, type the following commands one by one, replacing <Your Name> and <yourmail@gmail.com> with your actual information:



1. **Check Working Directory Status:**

* In the Git Bash window, type the following command and press Enter:

**git status**

**Repositories:**

In Git version control, repositories are the fundamental units that store and manage your project's history. They come in two main flavors:

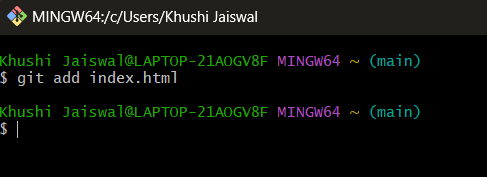
 **Local Repository:** Your workspace on your computer. It has your project files and a hidden folder tracking changes. Think of it as your project's notebook.

 **Remote Repository:** Lives on a server (like GitHub) and acts as a central hub. It stores the project's complete history for collaboration and backup. Imagine it as a secure online library for your project's notebook.

**Adding and Committing Changes**

1. **Add Files:**

* There are two ways to add files to the Git repository:
  + **Adding specific files:** Use the following command, replacing <filename> with the actual filename (e.g., git add index.html).



* **Adding all modified and new files:** Use the following command to add all changes in your working directory:

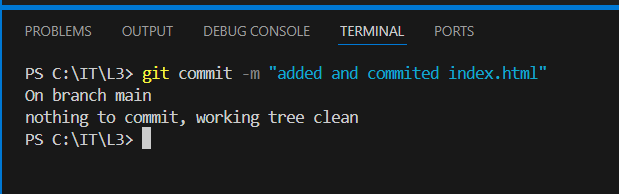
git add .

1. **Check Status Again:**

* After adding files, use the git status command again to see the current status.
* This should now show the added files under the "Changes to be committed" section.

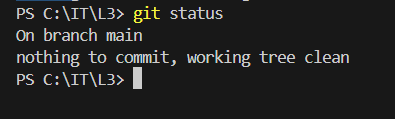
1. **Commit Your Changes:**

To save the changes permanently in your Git repository, use the git commit command. You can optionally add a commit message to describe the changes.



1. **Verify Clean Status:**

* Use git status one last time:

****\

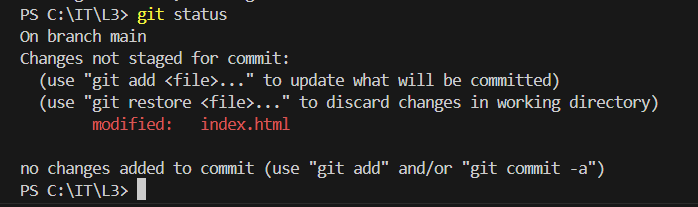
This should show "nothing to commit, working tree clean" indicating all changes are currently staged and committed.

**Making More Changes:**

* Edit your files (index.html or resources.txt) and add some content.

**Check Status After Changes:**

* Run git status again to see the newly made changes:



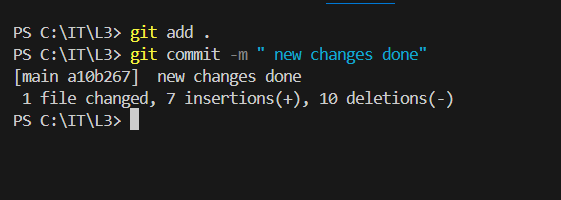
This will show "changes not staged for commit" as you haven't added these new changes to the repository yet.

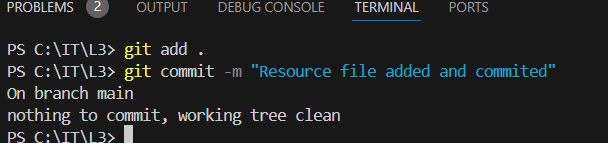
1. **Committing New Changes:**

* Use git commit with a descriptive message to commit the newly added changes.

1. **Checking Status After Commit:**

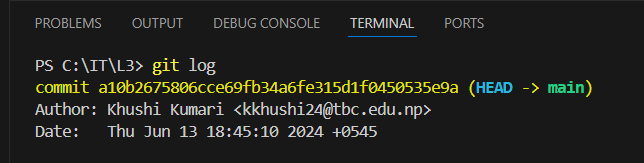
* Run git status to verify that the working directory is clean again.





**Viewing Commit History:**

* Use the following command to see the history of your commits:



**Git & GitHub - Connecting Your Local Repository**

**Connecting your local Git repository with a remote repository on GitHub allows you to:**

* Collaborate with others on projects.
* Track changes and versions of your code or files over time.
* Create backups of your work in the cloud.

**Here's a step-by-step guide for connecting your local repository to GitHub:**

**1. Create a GitHub Account:**

* Head over to <https://github.com/index> and create a free account if you don't have one already.

**2. Connect Local Repository to GitHub:**

* Open your Git Bash window (assuming you're in your project folder).

**a. Identify Remote Repository:**

* + Log in to your GitHub account and navigate to your newly created repository.
  + The repository URL will be displayed in the address bar of your browser. It will look something like https://github.com/<username>/<repository-name>.git. Copy this URL.

**b. Add Remote Repository:**

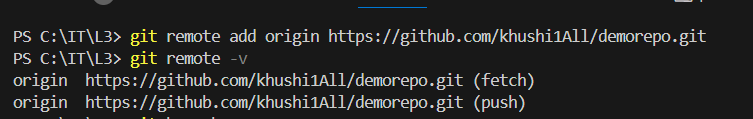
* + In your Git Bash window, type the following command, replacing <remote-name> with a name for the remote (e.g., origin) and pasting the copied URL in place of <URL>.

git remote add <remote-name> <URL>

**c. Verify Remote:**

Use the following command to list all configured remotes and verify the connection.

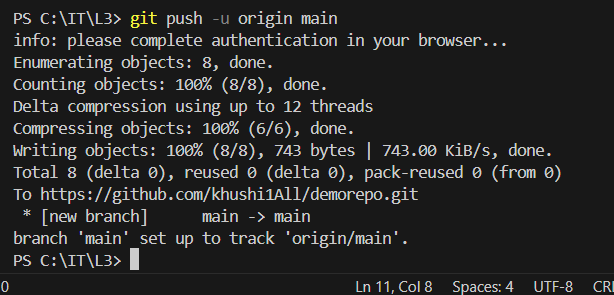
git remote –v



**Push Local Commits to GitHub:**

* Use the following command to push your local commits to the remote repository on GitHub:

git push -u origin main



**Once the push is successful:**

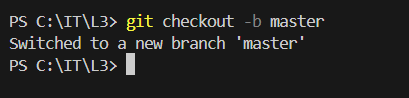
* You'll see your files uploaded to your GitHub repository.
* Refresh your GitHub repository page in your web browser to verify.

**Branching & Merging:**

**. Create a New Branch:**

* Use the following command to create a new branch named <branch-name>:

git checkout -b <branch-name>



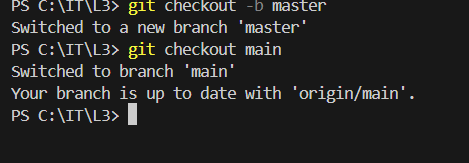
**Make Changes in the Branch:**

* Edit your files and make changes as needed.
* Stage and commit your changes like you did earlier in Task 1.

**3. Switch Back to Main Branch:**

* Use the following command to switch back to the main branch

git checkout main



**4. Merge the Branch:**

* Use the following command to merge the changes from your feature branch (<branch-name>) into the main branch

**. Delete the Branch (Optional):**

* Once your feature is merged and verified, you can delete the branch using the following command:

git branch -d <branch-name>