This also includes steps on how to connect the local repository to a remote repository on GitHub and how to use several essential Git commands listed below.

**### Git Practice Instructions for First-Time Students**

**#### \*\*Part 1: Creating a Local Repository\*\***

1. \*\*Open PyCharm IDE\*\*:

   - Launch PyCharm on your laptop.

   - Open or create a new project where you wish to initialize the Git repository.

2. \*\*Open Terminal in PyCharm\*\*:

   - Navigate to the bottom of the PyCharm window and click on the "Terminal" tab to open the command line interface within PyCharm.

3. \*\*Navigate to Your Project Directory\*\*:

   - **Use the `cd**` command to change directories to the specific folder where you want to initialize the repository. For example:

     ```

**cd path/to/your/project**

     ```

4. \*\*Initialize the Git Repository\*\*:

   - In the terminal, type the following command to initialize a new Git repository:

     ```

**git init**

     ```

**5. \*\*Verify Repository Initialization\*\*:**

   - Enter the following command to confirm that the directory is now a Git repository and to see the status of the files:

     ```

**git status**

     ```

   - This command will show you the files in the working directory that are not yet tracked by Git.

**#### \*\*Part 2: Linking Local Repository to GitHub\*\***

6. \*\*Create a Remote Repository on GitHub\*\*:

   - Log into your GitHub account.

   - Navigate to the Repositories tab and click on the "New" button.

   - Name your repository and set it to public or private as preferred. Do not initialize with a README, .gitignore, or license.

   - Once created, copy the remote repository URL provided by GitHub.

7. \*\*Connect Your Local Repository to the Remote Repository\*\*:

   - Back in the PyCharm terminal, link the local repository to your remote GitHub repository with the following command:

     ```

     git remote add origin [URL]

     ```

   - Replace `[URL]` with the URL you copied from GitHub.

**#### \*\*Part 3: Basic Git Operations\*\***

8. \*\*Add Files to Staging Area\*\*:

   - To track new files or changes, use:

     ```

**git add .**

     ```

   - This command adds all files in the directory to the staging area.

9. \*\*Commit Changes\*\*:

   - Commit the changes in the staging area with a message describing what was changed:

     ```

**git commit -m "Initial commit"**

     ```

**10. \*\*Push Changes to GitHub\*\*:**

    - Push your committed changes to GitHub with:

      ```

**git push origin master**

      ```

    - If you are using the main branch (recent GitHub repositories default to 'main' instead of 'master'), use:

      ```

**git push origin main**

      ```

**11. \*\*Check Status\*\*:**

    - After each command, you can check the status of your files and the synchronization with the remote repository:

      ```

**git status**

      ```

**12. \*\*Pull Latest Changes from GitHub\*\*:**

    - If there are any changes on the GitHub repository that need to be reflected locally, use:

      ```

**git pull origin main**

      ```

These steps demonstrates to the students  how to initialize a local Git repository in PyCharm, then connect it to a remote GitHub repository, and perform basic Git command operations to manage their project files. We will test these instructions on Friday.