Git Repository Setups Steps

1. Create a folder

- Open your terminal or command prompt and navigate to the directory where you want to create your project folder. Then run the following command to create a folder:
- **mkdir my_project**
- This command creates a folder named "my_project" in the current directory.
- To navigate into the newly created folder, run:
- **cd my_project**

2. Initialize a Git repository

- To initialize a Git repository in the "my_project" folder, run the following command:
- **git init**
- This command initializes an empty Git repository in the current directory.

3. Create a remote repository

- To create a remote repository, you can use a hosting service like GitHub, GitLab, or Bitbucket. Sign in to your chosen hosting service, create a new repository, and follow the instructions to set it up.

4. Add the remote repository URL

- Once you have created a remote repository, you need to add the remote repository URL to your local repository. Run the following command:
- git remote add origin <remote_repository_url>
- Replace <remote_repository_url> with the URL of your remote repository. For example:
- **git remote add origin https://github.com/your-username/my_project.git**

5. Create or add files to the project folder

- You can create or add files directly within the "my_project" folder using your preferred text editor or by using command-line tools.

6. Stage the files for commit

- To stage the files you want to commit, you need to add them to the Git staging area. In this case, we want to add all the files in the current directory. Run the following command:
- **git add .**
- The '.' represents the current directory. This command adds all the files and changes in the current directory to the staging area.
- You can also specify individual files or directories if you don't want to add everything. For example:
- **git add demo.py**
- This command adds only the " demo.py " file to the staging area.

7. Commit the changes

- After staging the files, you need to commit them to the Git repository. A commit is a snapshot of your project's current state. Run the following command to commit the changes:
- **git commit -m "Initial commit"**
- The -m flag is used to provide a commit message. In this example, the commit message is "Initial commit" to describe the first commit of the project.

8. Push your changes to the remote repository

- Finally, you can push your local commits to the remote repository. Run the following command:
- **git push -u origin master**
- This command pushes the commits from your local repository's master branch to the origin remote repository. If you are using a different branch,

Additional Git Commands

1. git restore Demo.txt

- This command restores the file "Demo.txt" to its previous state.
- It undoes any changes made to the file since the last commit.

2. git branch

- This command lists all the branches in the repository.
- In this case, there is only one branch named "master".

3. git branch demoBranch

- This command creates a new branch named "demoBranch".
- It allows for parallel development and isolates changes from the "master" branch.

4. git checkout demoBranch

- This command switches the current branch to "demoBranch".
- The prompt indicates the branch change.

5. git status

- This command displays the current status of the repository. It shows that there are changes in the working directory and untracked files.

6. git merge animation:

- This command merges the changes from the one branch into the current branch.
- The merge command is used to integrate changes from one branch into another.

7. git add .gitignore

- The .gitignore file is typically used to specify which files or directories should be ignored by Git.
- It allows you to exclude certain files or patterns from being tracked, such as temporary files, build artifacts, or sensitive information.