**GitHub**

GitHub is a web-based platform for version control and collaborations, allowing developers to store and manage their code, track changes and work with others on projects.

GitHub makes it’s easier to use Git’s features with a user-friendly interface and adds extra features like issue tacking, pull requests and project management tools.

**Key Concepts in GitHub:**

1. **Repository (Repo)**: A repository is a storage space where your project lives. You can store code, text files, images, or any type of data. Repositories can be public (viewable by everyone) or private (only accessible by specific people).
2. **Commit**: A commit is a snapshot of your repository at a specific point in time. Each commit saves the state of the project, and you can track the history of changes made.
3. **Branch**: A branch is a parallel version of your repository. You can work on different features or bug fixes in separate branches without affecting the main codebase (usually called the **master** or **main** branch).
4. **Pull Request**: This is a feature where you propose changes in one branch and ask for them to be merged into another branch. It allows for peer review and collaboration before changes are integrated into the main codebase.
5. **Fork**: A fork is a personal copy of another user's repository that you can make changes to without affecting the original repo.
6. **Merge**: This is the process of combining the changes from one branch into another branch.

**GitHub using**

1. Create GitHub and login
2. Click on + symbol it has right corner for creating repository
3. Open terminal or command prompt for cloning the repository

git clone <https://github.com/bhaskarreddyv123/programming.git> like this

1. Goto programming directory
2. Check status

**git status**

1. Add file ----------------> while adding file must be in the same repository

**git add filename**

OR

**git add .**

1. Commit the changes with a message

**git commit -m “message”**

1. After committing the changes, you need to push them to the GitHub repository

**git push origin branchname**

* origin refers to the GitHub repository.
* branchname is the branch you're working on (typically main or master)

1. After executing the push, it asks username and password

Username is your username

Password is passkey you want to generate

Passkey generation

* Open Settings
* In this open developer settings
* In this open personal access tokens
* In this click Tokens (classic)
* And now click on generate new token (Classic) and select all options given and click on generate token

**ghp\_zwOk4ktxRtpotQzdN2c4E1WItRry1i46ZoOH**

1. To pull updates from the GitHub repository to your local machine like directly added file now you want to see at terminal now use command

**git pull origin branchname**

1. Creating branch

**git checkout -b new\_branch\_name**

1. How to add file to branch

* **git branch** ---------> it shows in which branch is in by mark (\*)
* **git checkout branchname** ----------------> For shifting into branch
* Now add file by git add, git push

1. Merging Branches

To merge changes from one branch to another:

* Switch to the branch you want to merge into (e.g., main)

**git checkout main**

* Merge the other branch

**git merge new-branch-name**

1. Pull Request (PRs)

If you're working with others, you can create a pull request to propose changes:

* Push your branch to GitHub.
* Navigate to the repository on GitHub.
* Click **Pull Requests**, then **New Pull Request**.
* Select the branch you want to merge into, review your changes, and create the pull request.

1. **Git Stash for Managing Uncommitted Work**

**Stashing** allows you to save your uncommitted changes temporarily without committing them, so you can switch to another branch or work on a different task.

* Stash your changes:

**git stash**

* List stashed changes:

**git stash list**

* Apply the latest stash:

**git stash apply**

1. Other Common Git Functions

* **git log**: View the commit history.
* **git revert**: Revert a commit.
* **git reset**: Undo changes.
* **git stash**: Temporarily save changes without committing.
* **git remote add origin [URL]**: Connect your local repository to a remote one.

1. Delete repository
2. **Navigate to the Repository**

* Go to [GitHub](https://github.com/) and log in.
* From your dashboard, click on the repository you want to delete.

1. **Go to Repository Settings**

* Once inside the repository, click on the **Settings** tab located at the top of the page.

1. **Scroll Down to "Danger Zone"**

* Scroll to the bottom of the **Settings** page until you see the **Danger Zone** section.

1. **Click "Delete this repository"**

* In the **Danger Zone**, find the option that says **Delete this repository** and click on it.

1. **Confirm Deletion**

* A prompt will appear asking you to confirm the repository deletion. You’ll need to type the repository name exactly as shown to confirm the action (e.g., yourusername/repositoryname).
* Once confirmed, click the **Delete repository** button.

1. **Final Confirmation**

* GitHub may ask for your password to confirm the action.
* After confirming, the repository will be permanently deleted.

**Note**: Deleting a repository is a permanent action and cannot be undone. All data, commits, and branches in that repository will be lost unless you've backed it up locally.