comprehensive understanding of Git and GitHub

1. **Git?**
2. **Github?**
3. **ReadMe file?**
4. **Configuring git?**
5. **Git commands?**
6. **Workflow?**
7. **Git branch**
8. **Merging code**
9. **Pull command**
10. **Undoing Changes**

1. what is git?

before study Git we need to understand version control system.
Version control system is a tool that helps to track changes in code.

Git is a version control system and it is :

- ✓ Popular
- ✓ Free and Open source
- ✓ Fast and Scalable

- We primarily use Git for two main purposes:

1. Track the history
2. Collaboration

2. Github

- it is website that allow developers to stor and manage there code using Git.
- GitHub is a website where you can store your Git projects online, collaborate with others, and showcase your work.

- It's great for teamwork, open-source projects, and portfolio building.

3. ReadMe file

- A README is a text file (usually README.md) that explains your project.

In the README, we write to explain

- what the project is,
- what its name is,
- how to use it,
- why you created this project,
- what features it includes,
- and other related information.

- If you know basic HTML, you can make your README.md file look better and more organized.

4. Configuring Git

Configuring Git means setting up your Git environment so it knows who you are and how to behave.

Configuration Levels:

- System level: Settings for everyone using Git on the same computer.

Example: Default editor for all users.

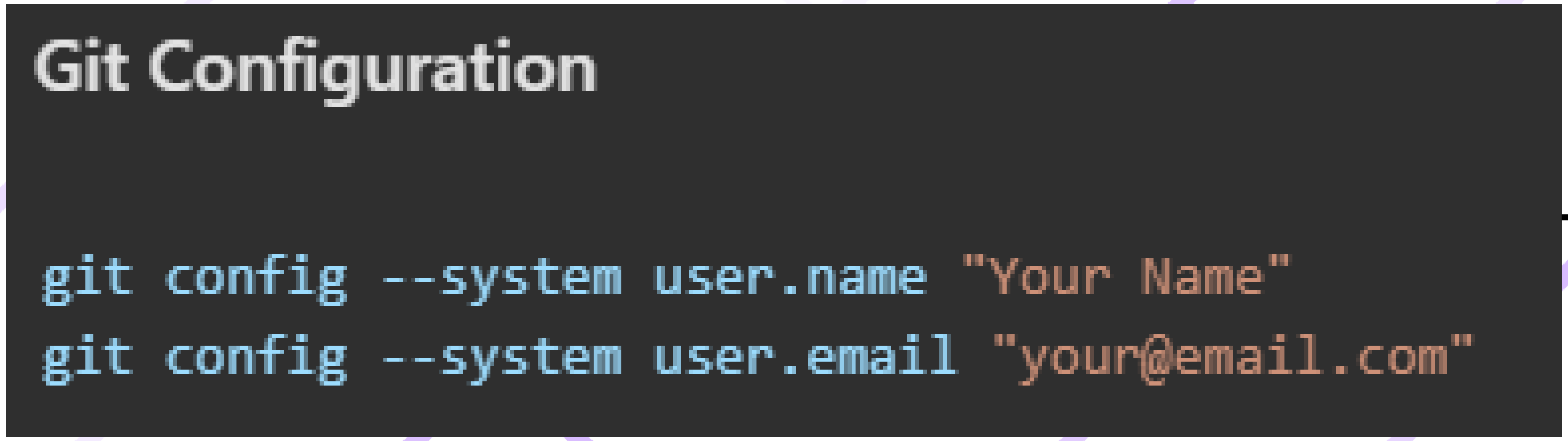
- Global level: Settings for you, the current user.

Example: Your name and email for all your projects.

- Local level: Settings for one specific project.

Example: Using a different username or email just for that repo.

System level (for all users on the computer):

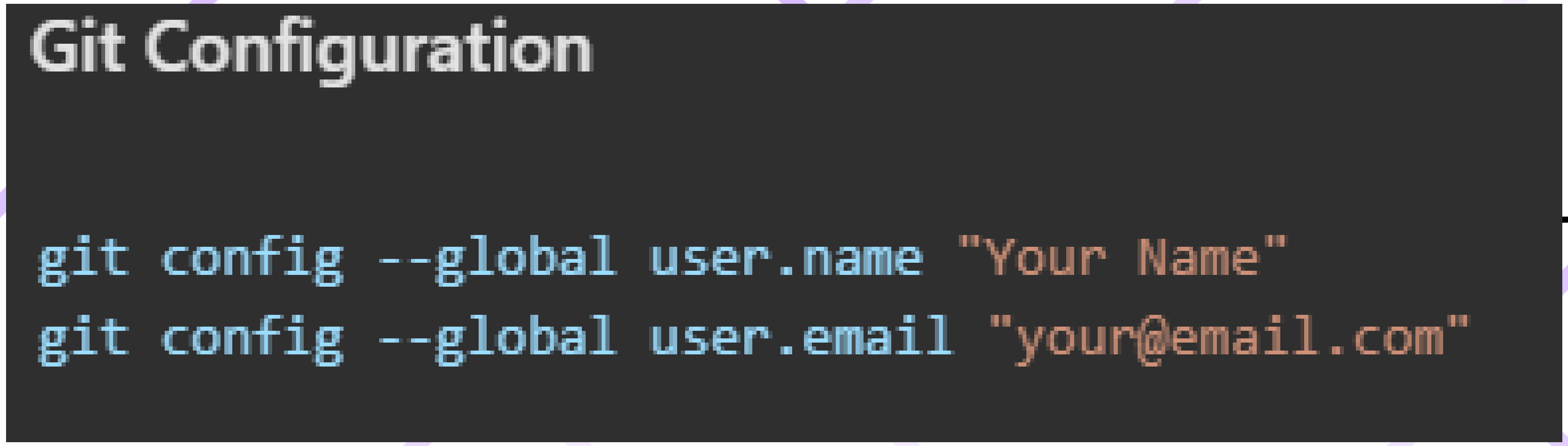


Git Configuration

```
git config --system user.name "Your Name"
git config --system user.email "your@email.com"
```

🔐 Needs admin /root access.

Global level (for your user account):

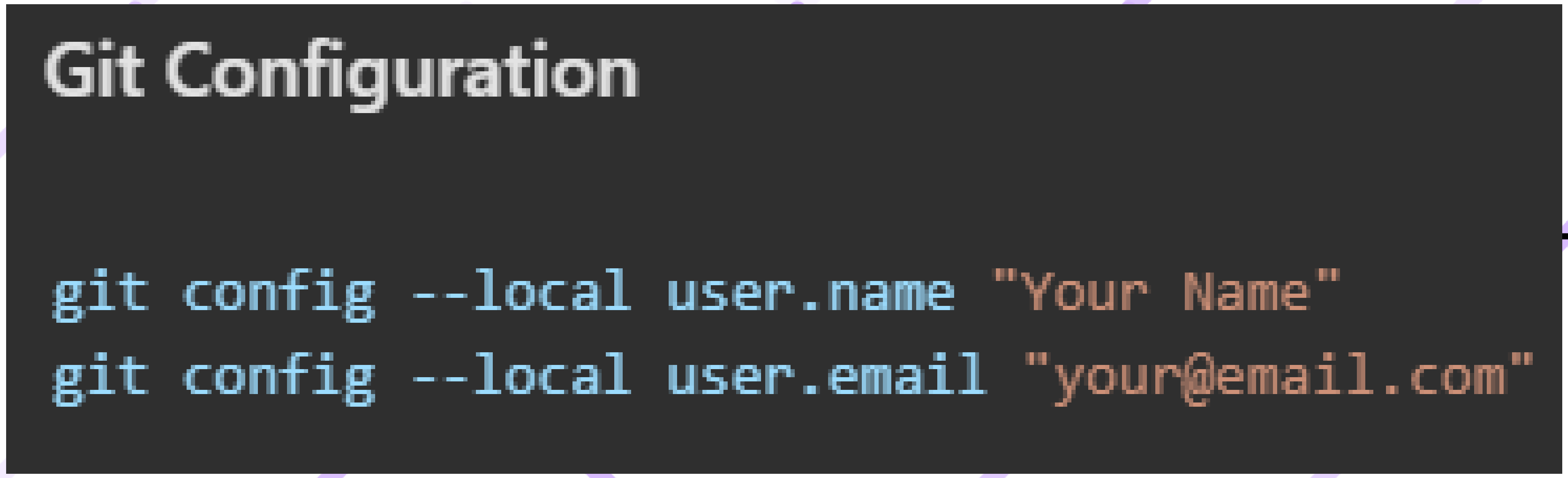


Git Configuration

```
git config --global user.name "Your Name"
git config --global user.email "your@email.com"
```

Most commonly used.

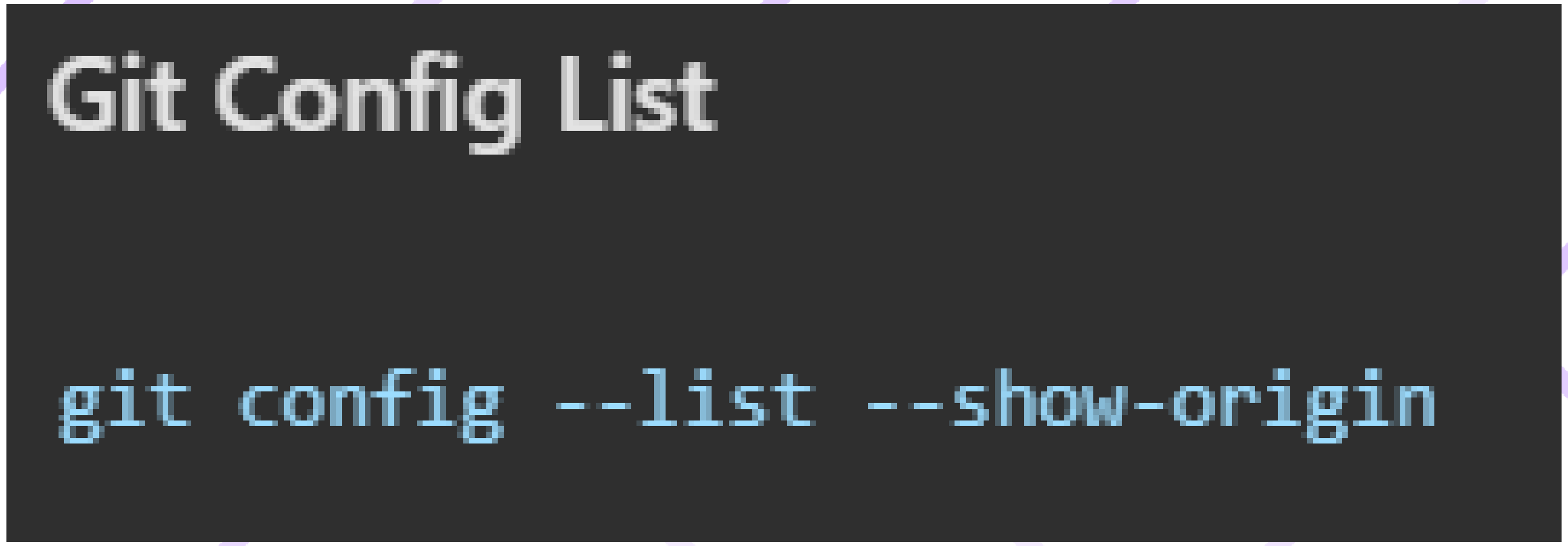
✅ Local level (for current project/repo):


Git Configuration

```
git config --local user.name "Your Name"
git config --local user.email "your@email.com"
```

💡 Run this inside the project folder.

☑️ You can view the config using:


Git Config List

```
git config --list --show-origin
```

5. Git commands

- git clone <url> ⟶ Cloning a repositery on your local machine.
- git status ⟶ Display the status of your code

Untracked ⟶	The new file that not tracked yet.
Modified ⟶	Changed
Unmodified ⟶	being tracked by Git and has no changes
Staged ⟶	files are ready to be committed

- git add <file name> ⟶ add changes or new file in working directory
- git add . ⟶ add all changes at once

- git commit -m "message" ⟶ It's record of changes.
- git push origin main ⟶ Upload local repo content to remote repo
- git init ⟶ Used to create new repo
- git remote add origine < link >
- git remote -v ⟶ To verify remote
- git branch ⟶ To check branch
- git branch -m main ⟶ To rename branch
- git push origin main
- git push -u origin main ⟶ we use -u for upstream , it use to work long time to overcome write of origin main.

6. Workflow

GitHub Repo

↓

Clone Repository
(git clone <url>)

↓

Make Changes
(edit files)

↓

Stage Changes
(git add .)

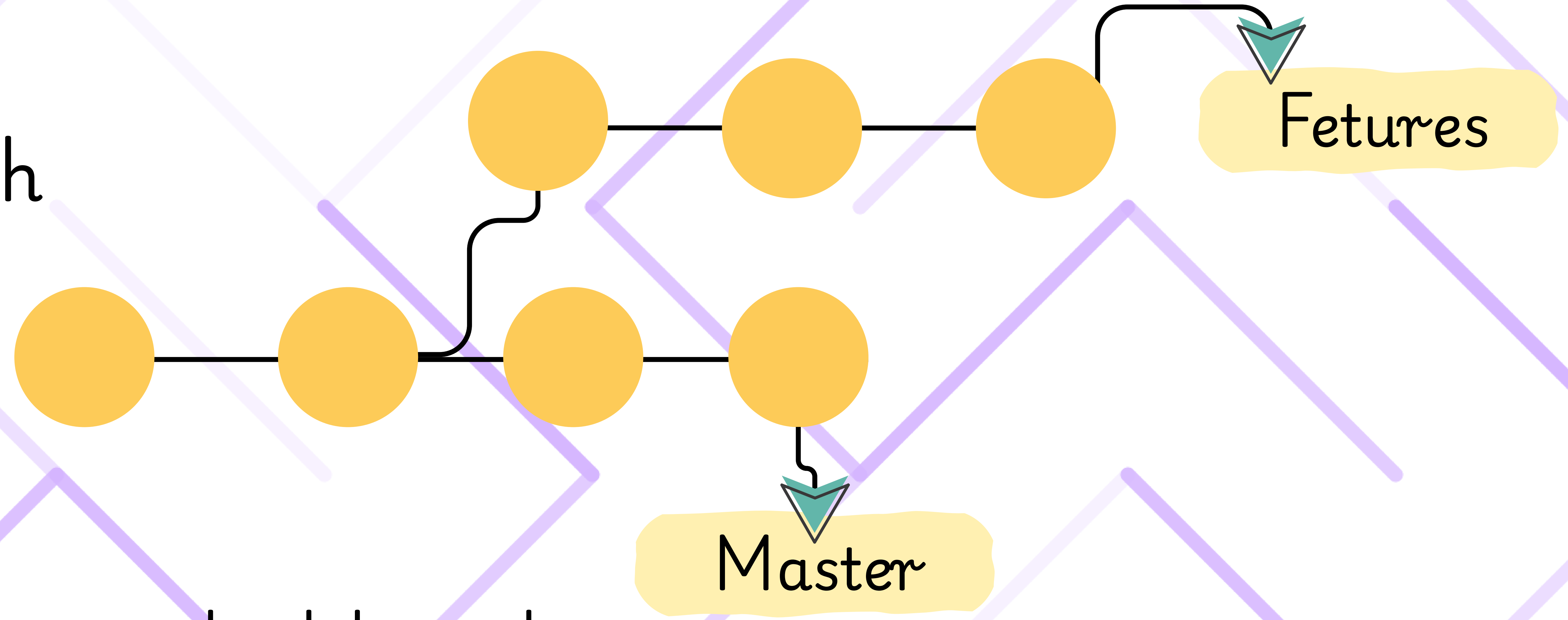
↓

Commit Changes
(git commit -m "message")

↓

Push to GitHub
(git push)

Git Branch



- git branch → to check branch
- git branch -m main → to rename branch
- git checkout <branch name> → to navigate (to go 2nd branch)
- git chckout -b <branch name> → to create new branch
- git branch -d <branch name> → to delet branch

Merging code

Two way to merge your code

- Way 1

  git diff <branch name> → to compare commits branches, files and more
  git merge <branch name> → to merge two branchs

- Way 2

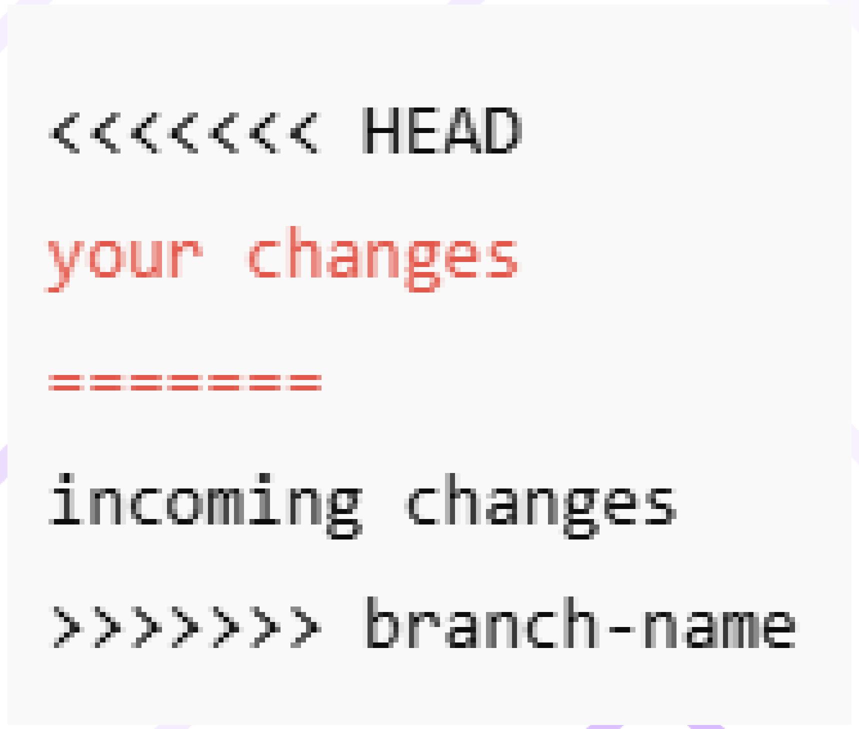
  create a pull request (PR) — It let you tell others about changes you've pushed to a branch in a repository on github.

Pull command

- git pull is used to download the latest changes from the remote repository (like GitHub) and merge them into your local project.
- git pull origin main → Pull (download and merge) the latest code from the main branch of the remote repo named origin.

Resolving Merge Conflict

- An event that takes place when git is unable to autometically resolve .
- git log → to check all commits



```
<<<<<<< HEAD
your changes
=======
incoming changes
>>>>>>> branch-name
```

Manually fix conflict

Edit the file, remove the <<<<<<<, =======, and >>>>>>>

After remove all this type of lines you can run the command

git add <file>

 git commit -m "Resolved merge conflict"

git push

Undoing Changes some changes add by mistake

Case 1 - Staged change → files are add but not commited

- git reset <filename> → for one file
- git reset → for all file

Case 2 - commited changes → for one commit

- git reset HEAD~1 → Removes the last commit
  Keeps the changes in unstaged state (you won't lose your work)

Case 3 - commited changes → for many commits

- git reset <commit-hash> → multiple commit se wapas jane ke liye hash use karte hai

- git reset -hard → delete all change or change remove ho jate hai

I HOPE YOU LEARN SOMETHING FROM HERE

THANK YOU