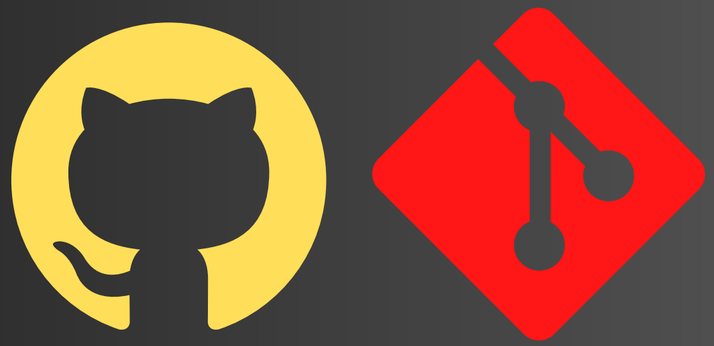
Git and GitHUB



What is Git?

Git is a distributed version control system (VCS).

Git is a tool that helps multiple people work on the same code project or documents by tracking and managing changes to the files.

Project Files

Project Files

Bug Fix

New Updates

Security Updates

Project Files

Bug Fix

New Updates

Security Updates

What is VCS?

Every time you make a change, whether it’s adding a sentence to a document or altering a line of code, the VCS records and saves the outcome.

project

project-1

project-2

project-final

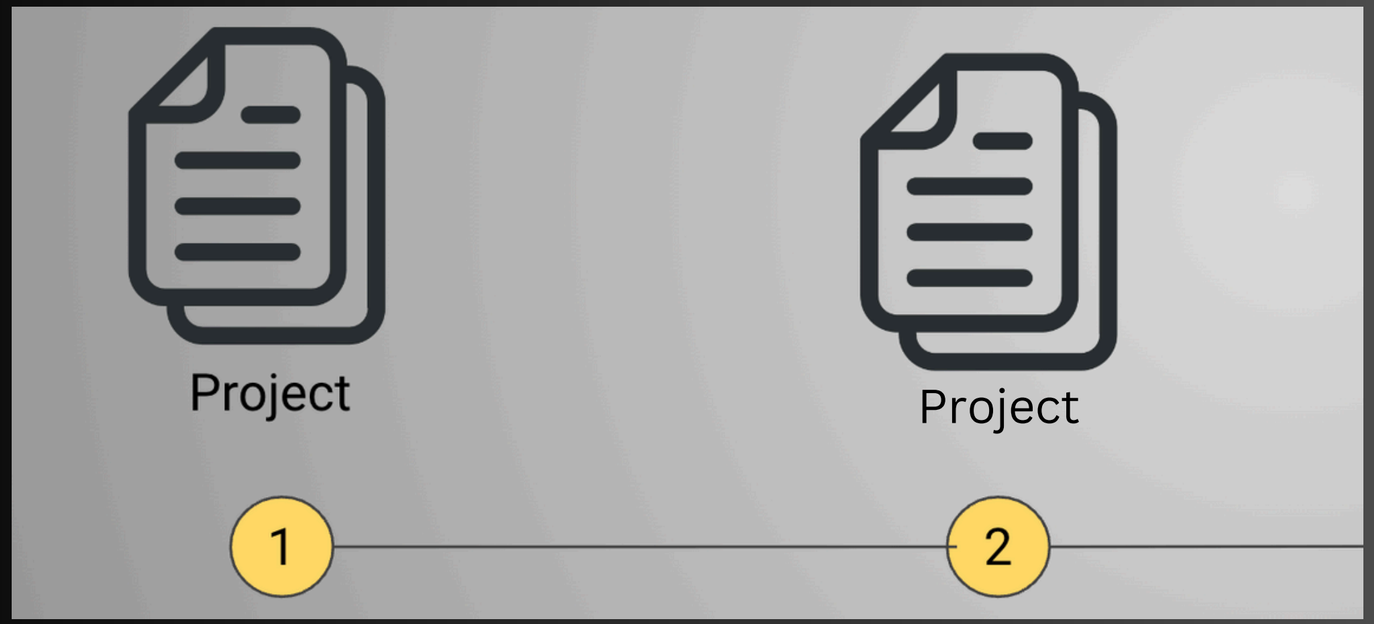
project-final1

project-finalfinal

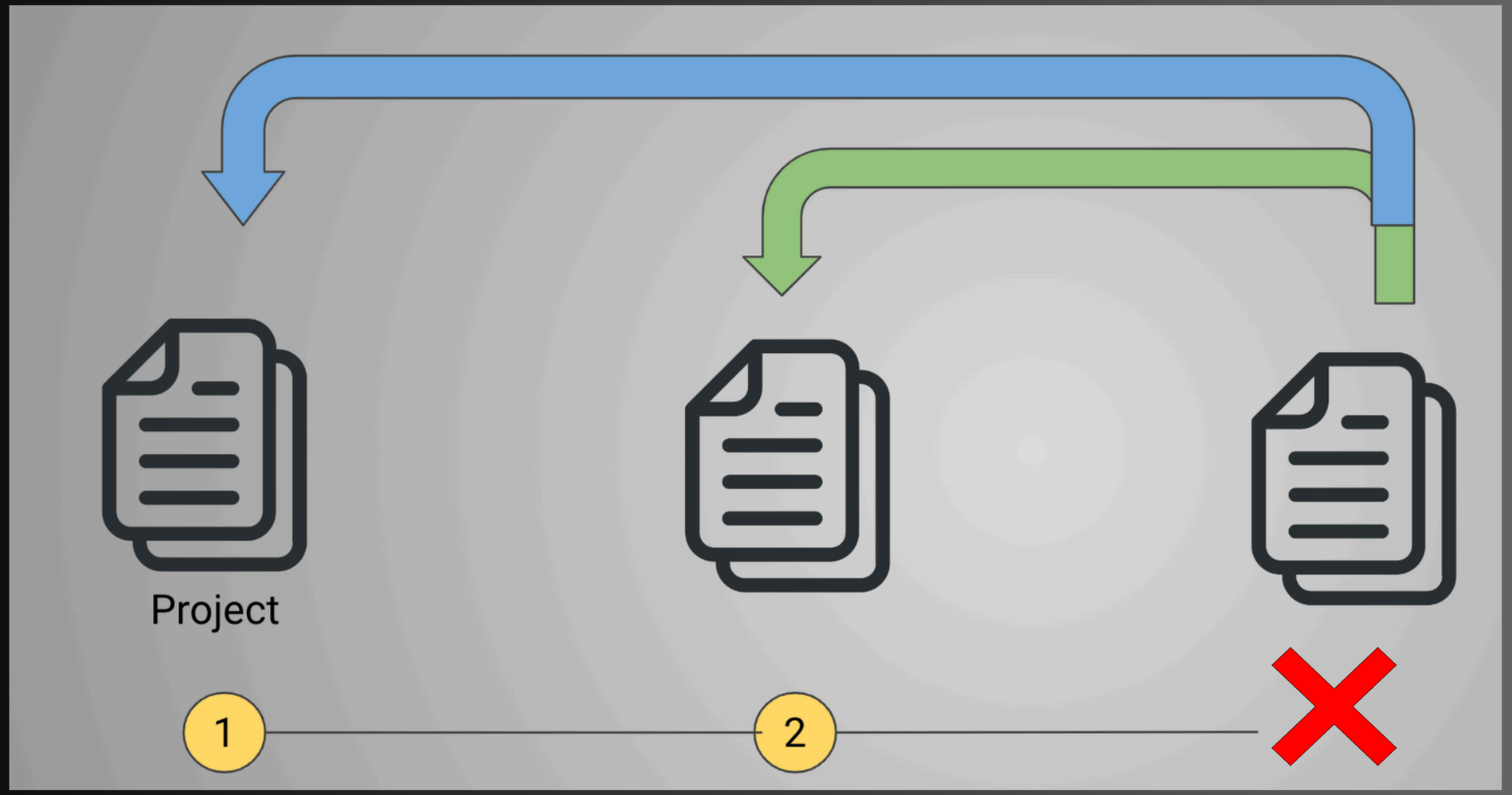
Project Files

Project

Project


Project
Project
1
2


Project
1
2


Project
1
2

Why VCS?

- **Backup and Restore:** Files are safe against accidental losses or mistakes.
- **Collaboration**: Multiple people can work on the same project simultaneously.
- **Branching and Merging**: Users can diverge from the main base of code, experiment, and then bring changes back in line without losing work.
- **Tracking Changes**: You can see specific changes made and by whom.

MPRASHANT

Installing Git

- Step-by-step installation of Git.
- Configuring Git with your username and email.

Configure Git with Username and Email

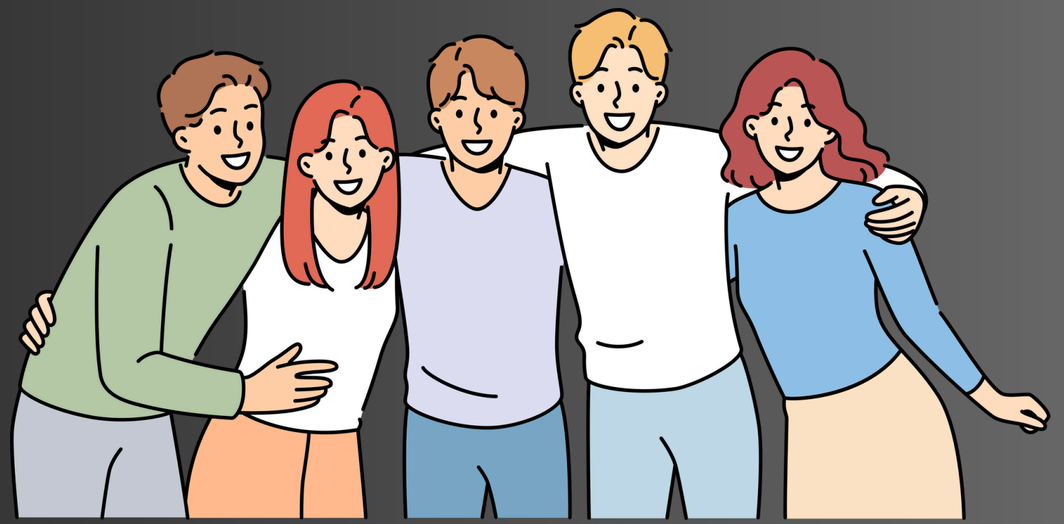
```
$ git config --global user.name "Paul Philips"
$ git config --global user.email paulphilips@email.com

$ git config --list
```

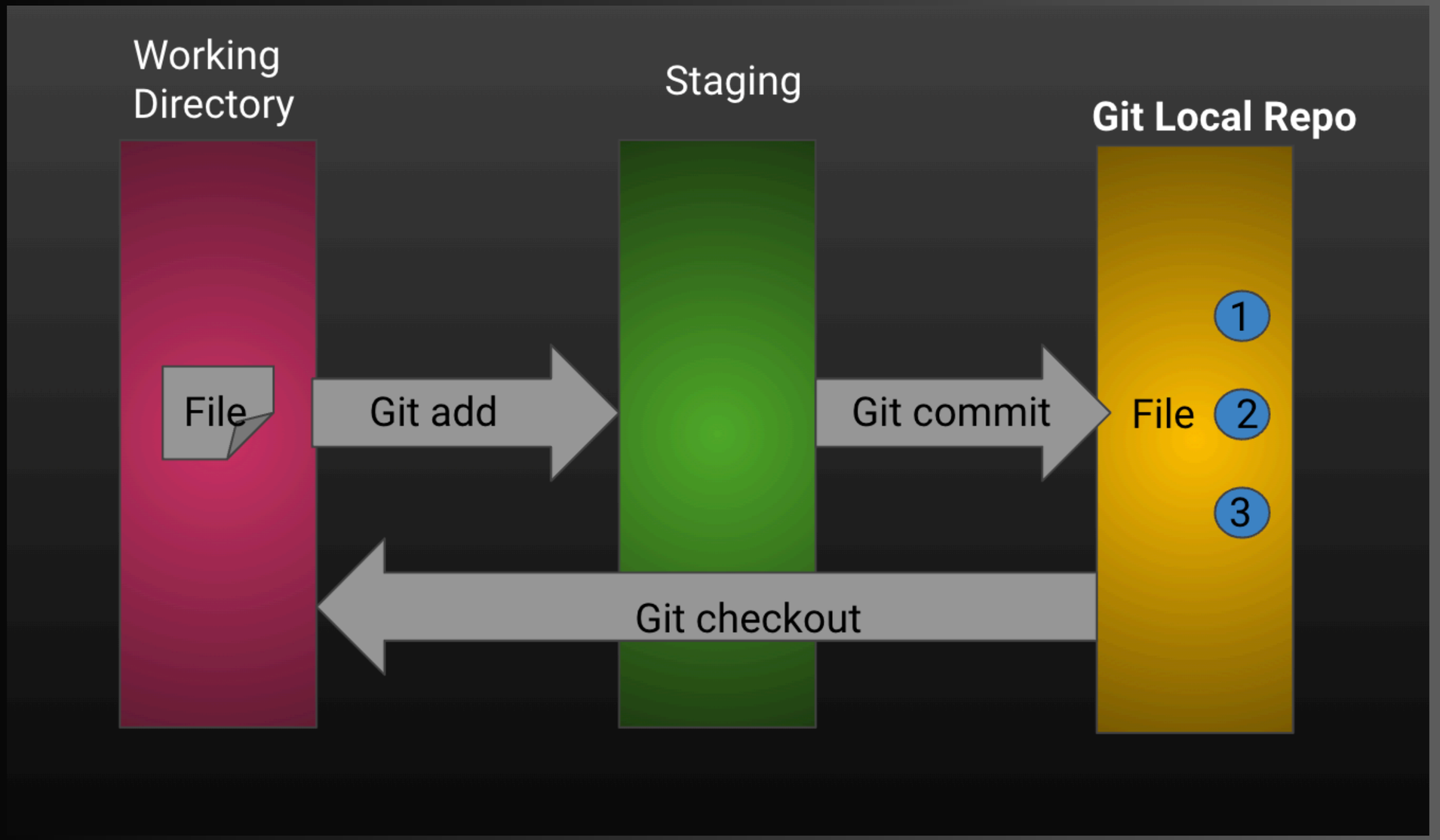
Creating Local Git Repo

```
$ git init
```

What is Git Commit?

A git commit is a command in Git that captures a snapshot of the project's currently staged changes, creating a permanent record in the repository's history.




Working Directory
Staging
Git Local Repo
File
Git add
Git commit
File
1
2
3
Git checkout

Negative Cases

If we made any change by mistake and save it

Case1: To undo changes, get the last successful change
git restore . or filename (. mean all files)

Case2: If we added the changes using git add then..
git restore --staged <file_path> # To unstage
git restore <file_path> # To discard changes in the working directory

Negative Cases

Case3: Added changes to staging area (didn't commit) after this added more changes to file

```
//To get the staged changes
git restore --worktree index.html
```

Case4: How about if we did commit also wrong files

```
git reset --soft HEAD^ (uncommit and keep the changes)
git reset --hard HEAD^ (uncommit and discard the changes)
```

MPRASHANT

Logging in Git

Useful Log Options

- git log -p -2 *(last two commit with diff)*
- git log --stat (*summary of changes*)

- git log --pretty=oneline
- git log --pretty=format:"%h - %an, %ar : %s"

- git log -S function_name

Useful Log Options

- git log --grep="fix bug" *(search commit msg)*

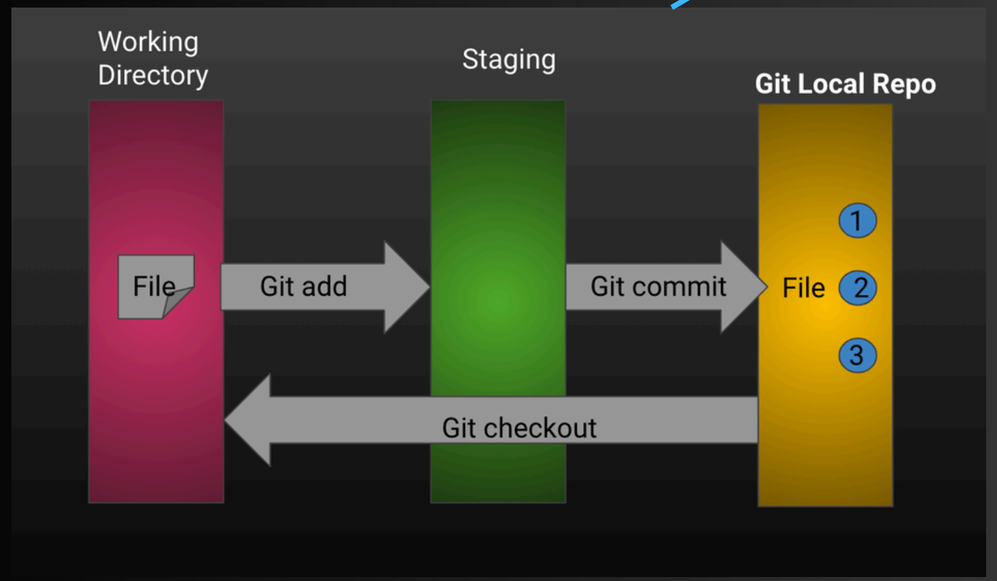
- git log --since="2024-01-01"
- git log --until="2024-01-01"

- git log --author="Paul"
- git log --no-merges

MPRASHANT

Remote Repo

A remote repository refers to a version of your project that resides on a network server or a hosted repository on the internet.


Working Directory
Staging
Git Local Repo
File
Git add
Git commit
File
1
2
3
Git checkout

Git HUB

Adding Remote Repo

```
git remote add origin <remote URL>
```

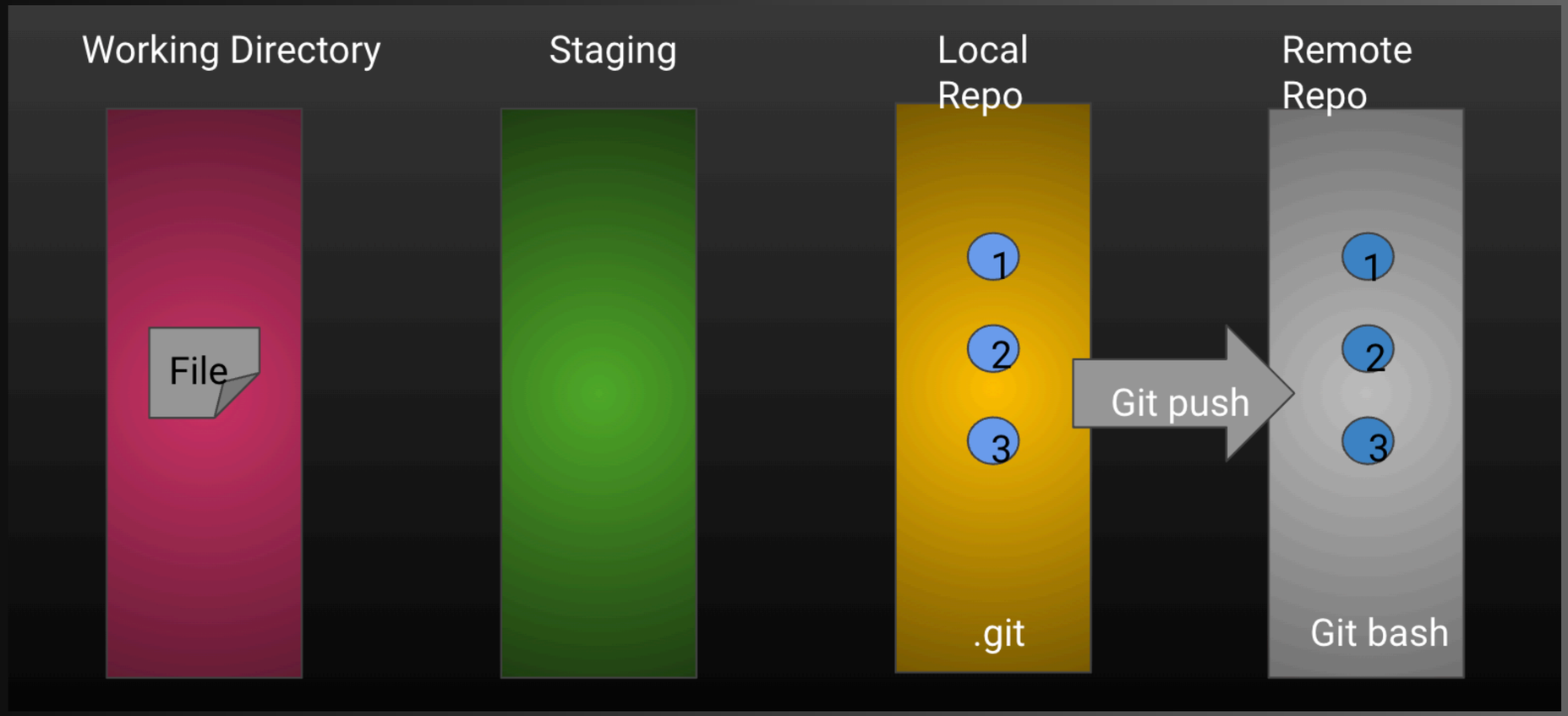
```
git push -u origin master
```

Working with Remote

To see the remote git link

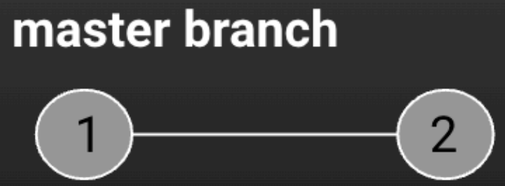
git remote
git remote -v
git remote show origin (to get more info about remote)

git clone
git pull

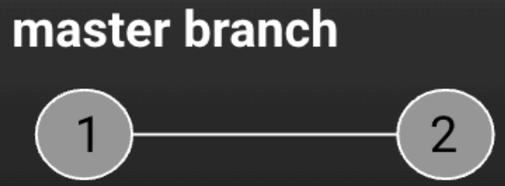

Working Directory
Staging
Local Repo
Remote Repo
File
1
2
3
Git push
1
2
3
.git
Git bash

Git status	To check for any change in the repo
Git diff	To see the difference of current version with last committed version
Git pull	To fetch the latest files from remote repo
Git add	To add the modified files in staging area
Git commit	To add the changes in git repo
Git push	To push the committed changes in Remote repo
Git log	To see the old commits history

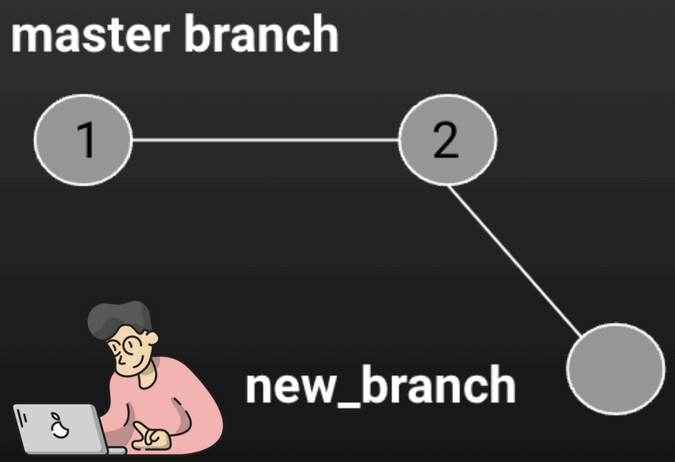
Branching and Merging



Branching and Merging

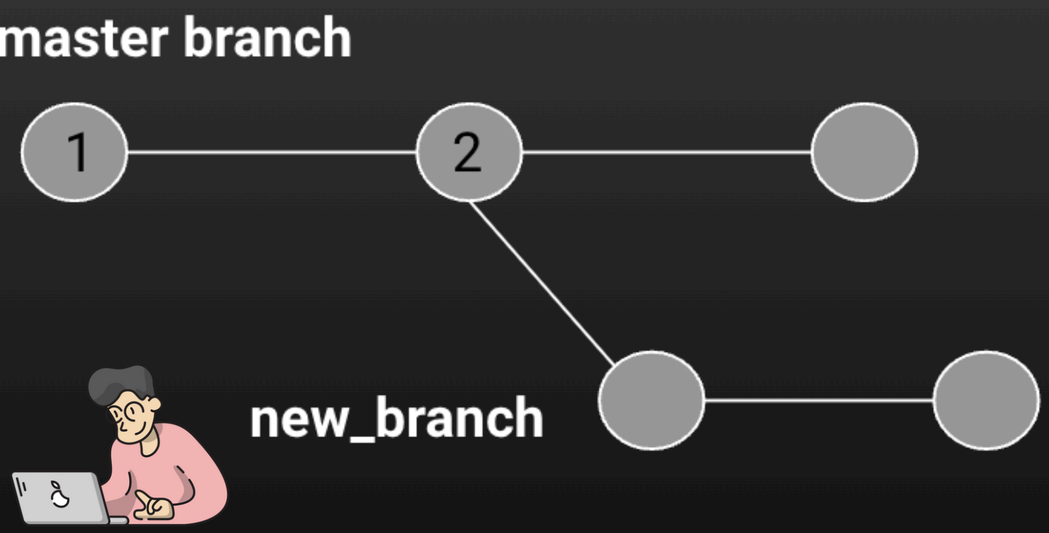


Branching and Merging



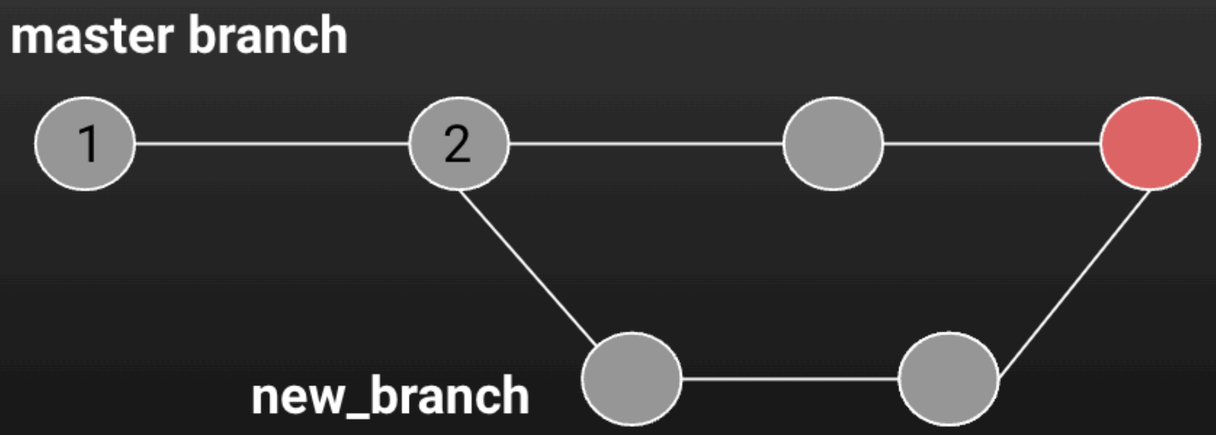

```
$ git branch new_branch
```

Branching and Merging




```
$ git branch new_branch
```

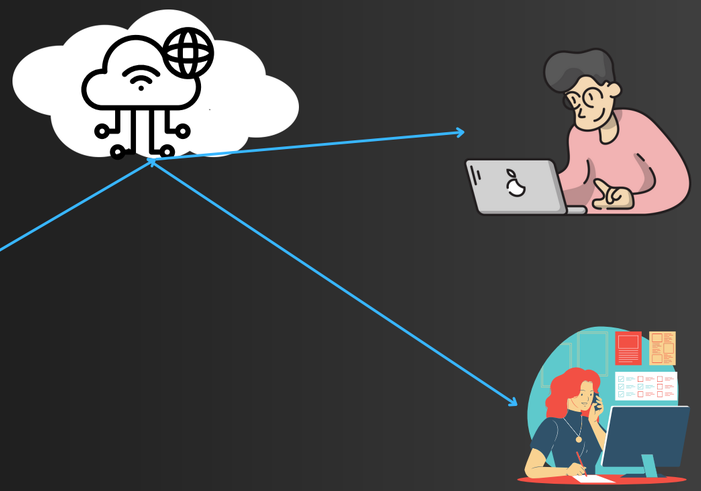
Branching and Merging

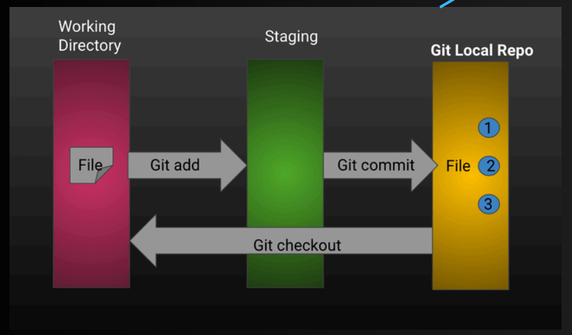



```
$ git checkout master
$ git merge new_branch
```

MPRASHANT

Git Forking and Pull Request





I can improve the code

MPRASHANT

Gitignore

Git Ignore

In case you don’t want to commit some files from your working directories then add those files name in .gitignore file.

Will be useful when your file contain confidential info.

Cloning

To get the remote repo into your local system.

```
git clone <remote_repo_link>
```

MPRASHANT

Git Clean

MPRASHANT

Git Tags

Git Tagging

To create annotated tags

```
git tag -a v1.0 -m “My version 1.0”
```

To show tags data

```
git show v1.0
```

To tag old commit in case you forgot

```
git tag -a v1.2 <commit_no>
```

Git Tagging

To delete a tag

git tag -d <tag_no>

Tags created remain local, to move it to remote:

git push origin v1.5

git push origin --tags (For all tags together)

In case in future, if you wanna give patch for v1.2 release let’s say

git checkout -b version2 v2.0.0 (it will create new branch also)

MPRASHANT

Managing access and permissions.

MPRASHANT

Using GitHub for Project Management

- Overview of GitHub features like Issues, GitHub Actions, and Projects.
- Tracking project progress using Kanban boards.
- Automating workflows with GitHub Actions.

- Securing your Git and GitHub accounts.
- Managing access and permissions.
- Monitoring repositories for vulnerabilities.