

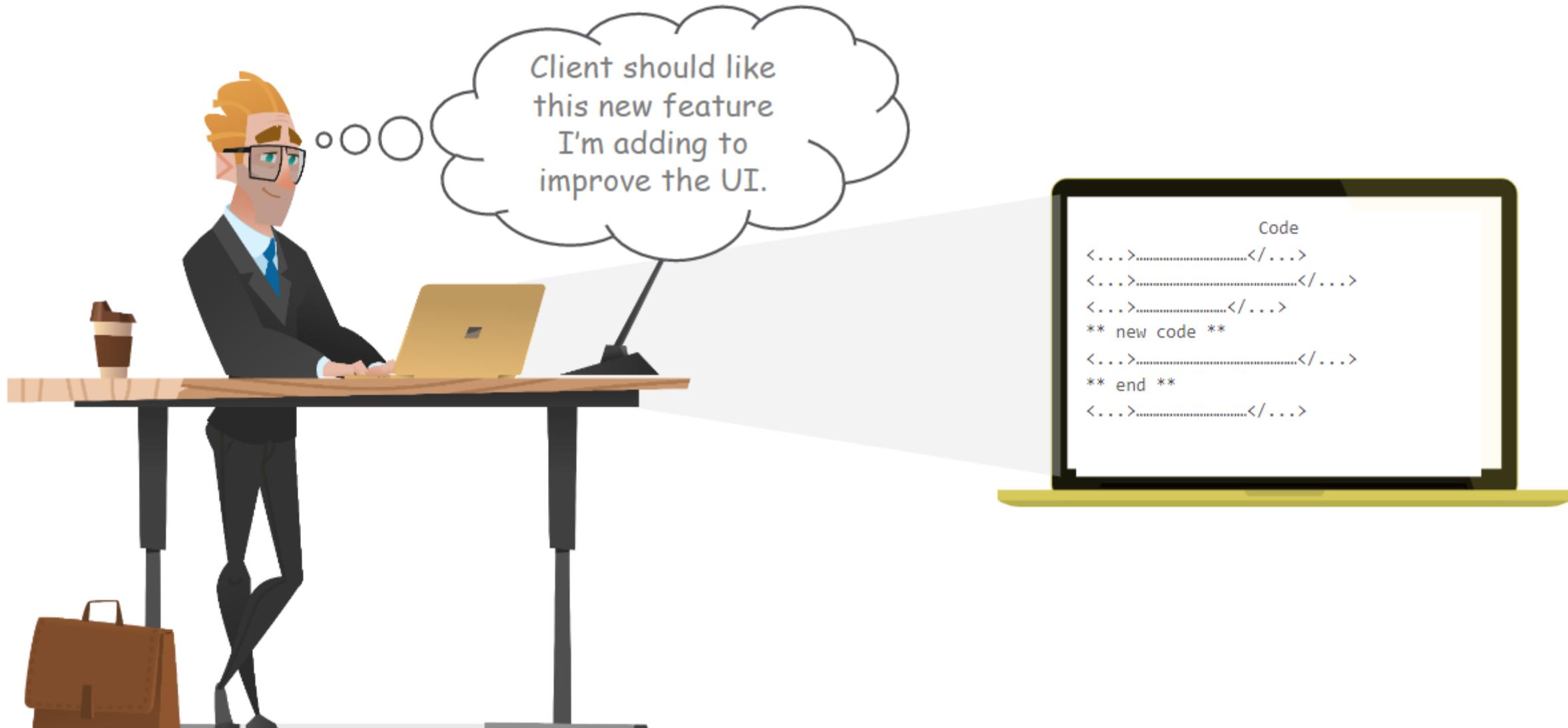
Distributed Version Control System. DVCS



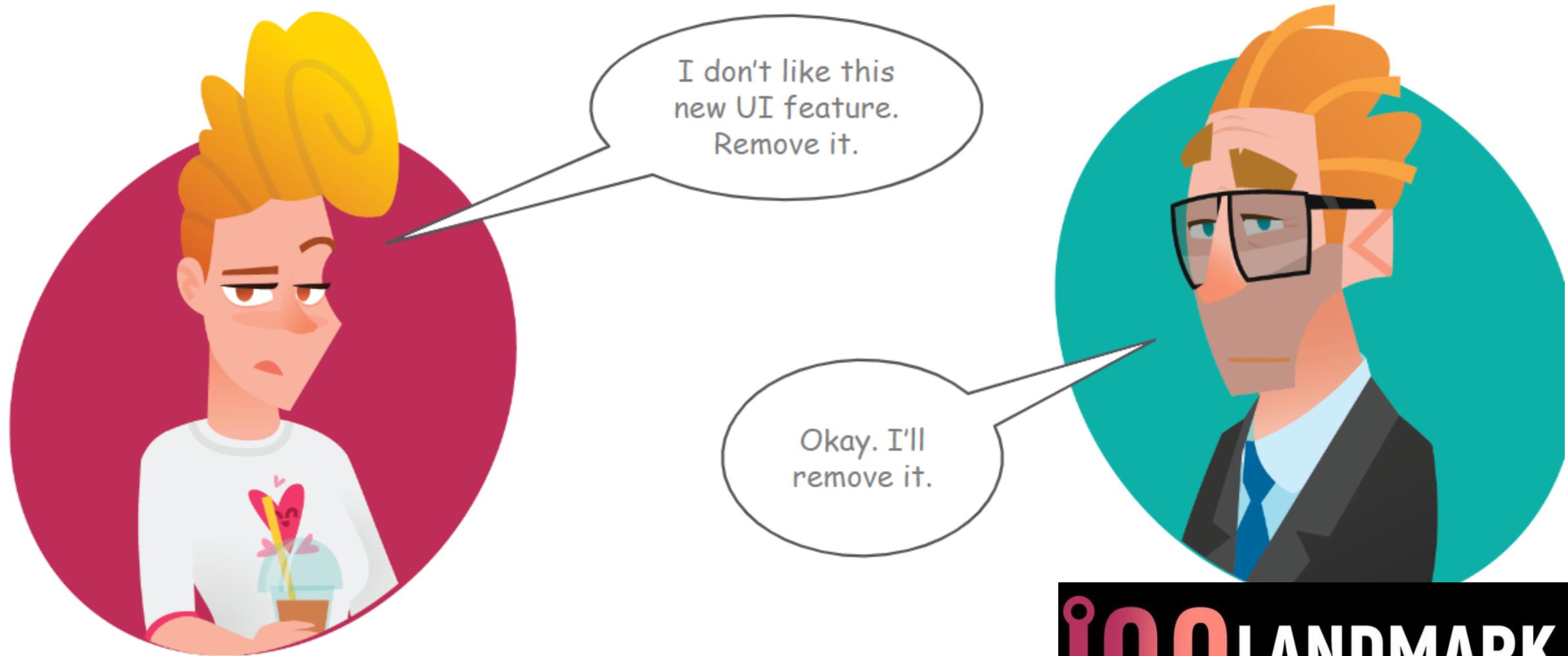
git



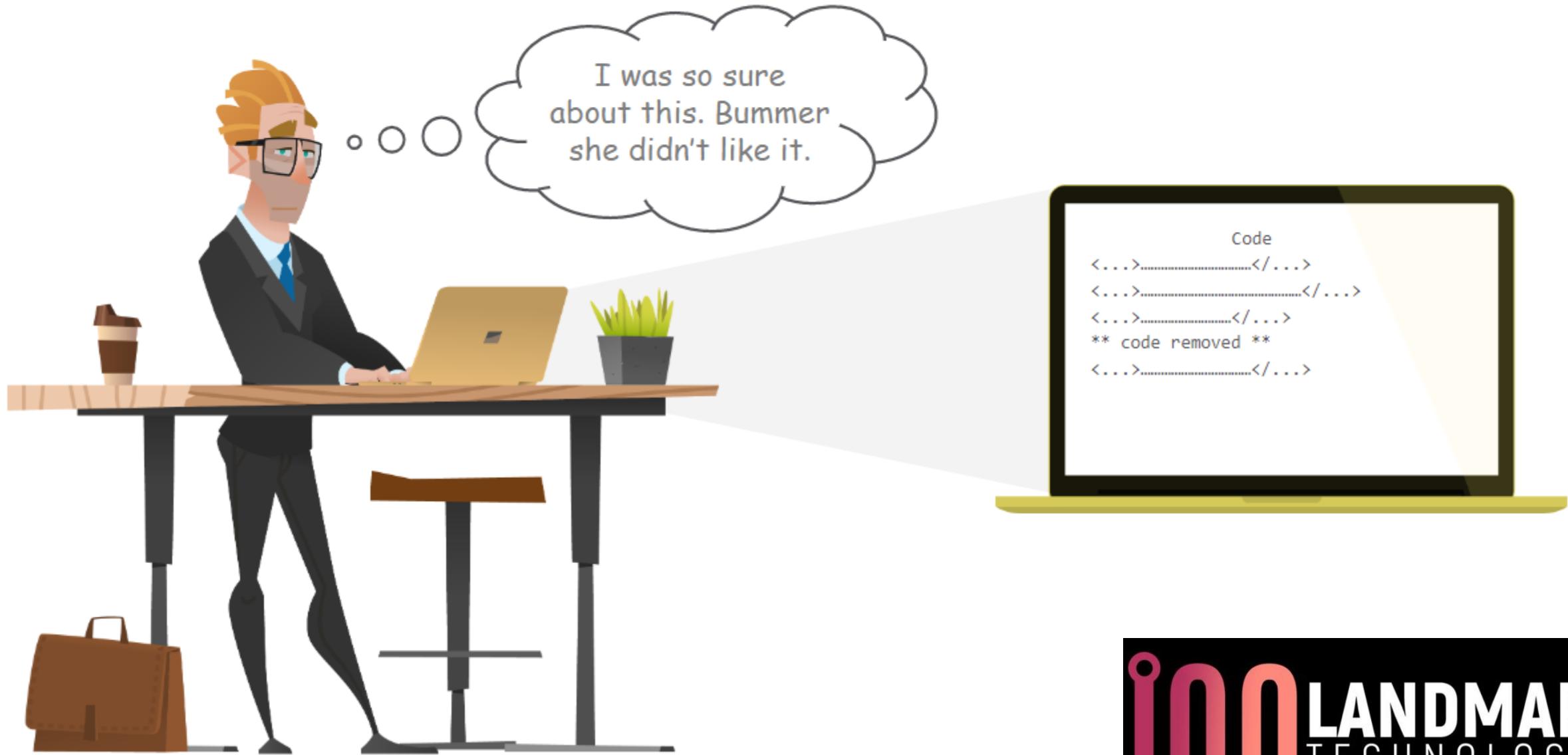
Dave Working On A Project



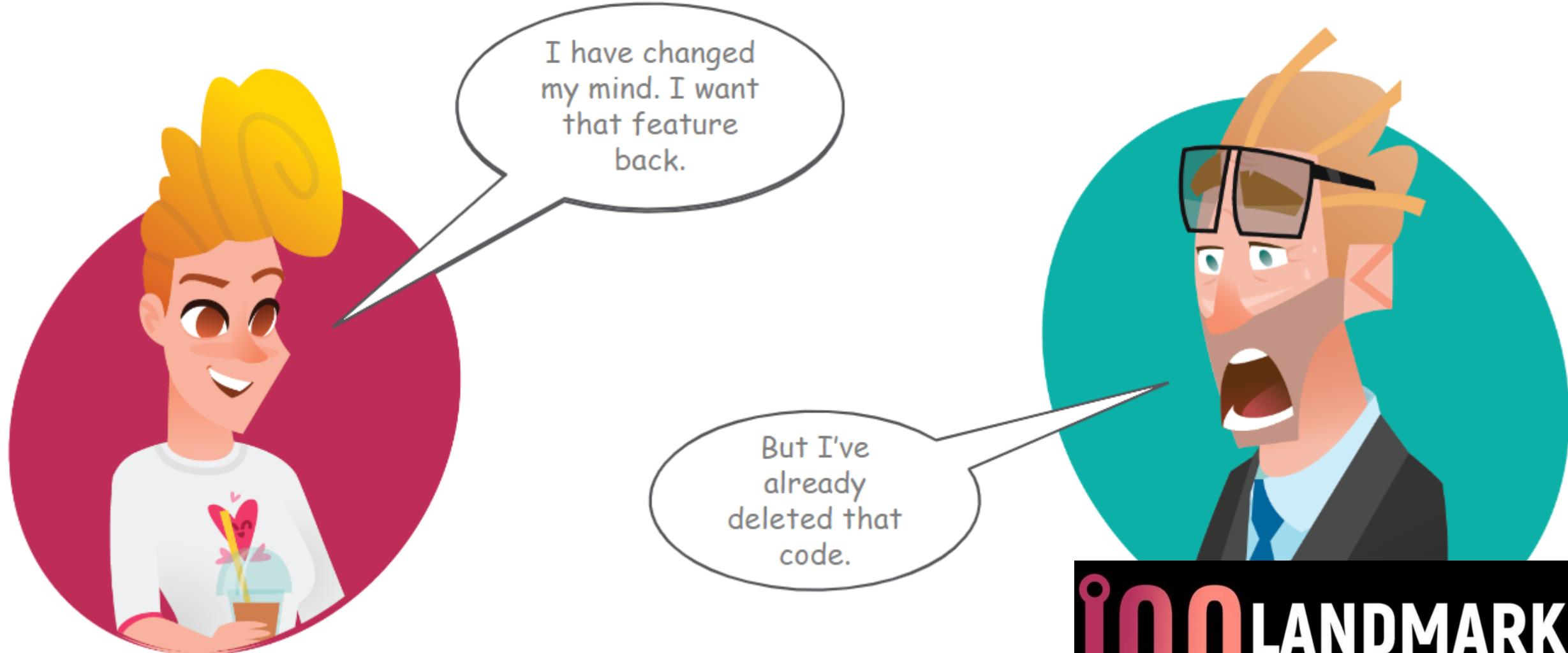
Disapproval From Client



Dave Changes The Code



Client Changes Her Mind



Dave Writes The New Code Again

-



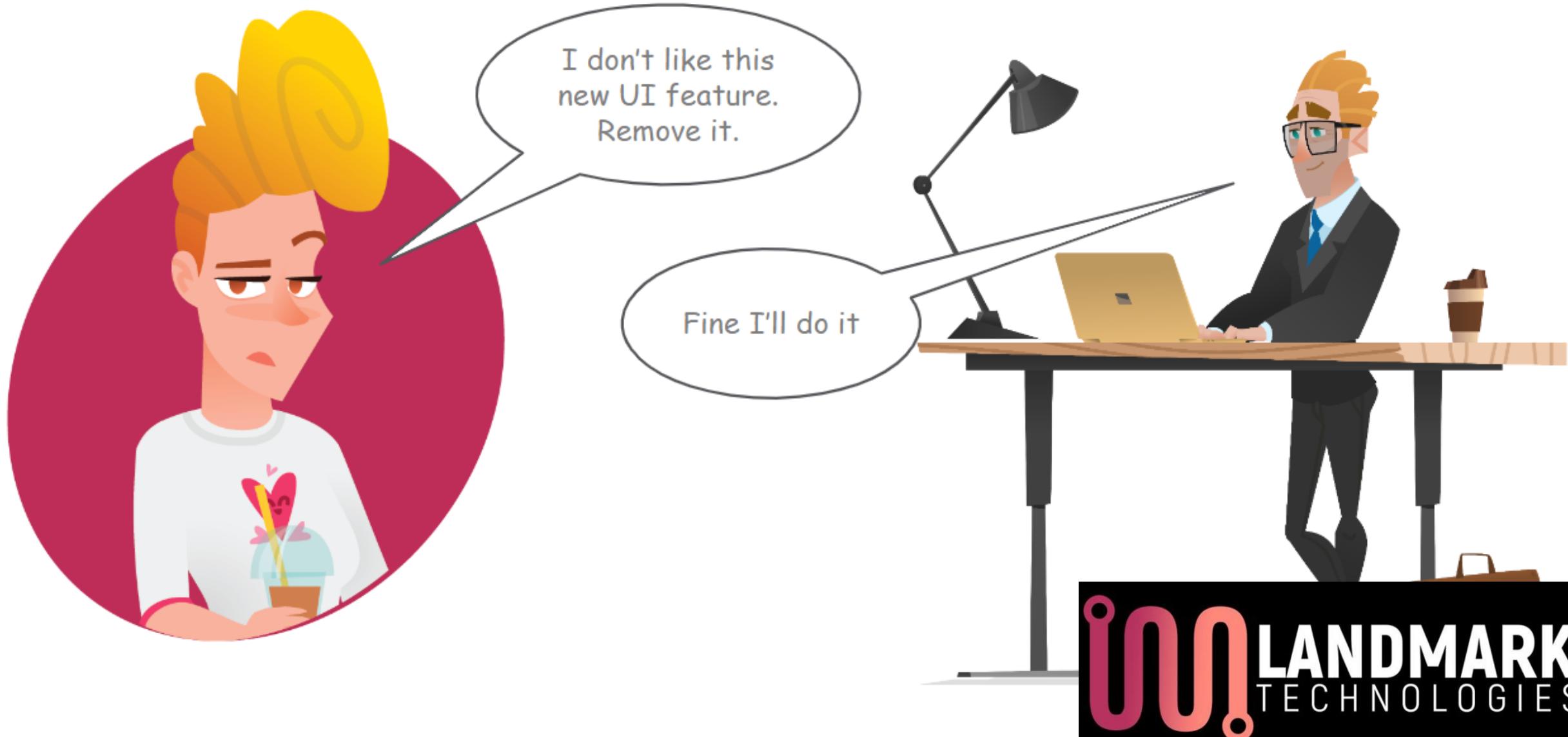
Issues Without Version Control

- Once saved, all the changes made in the files are permanent and cannot be reverted back
- No record of what was done and by whom
- Downtime that can occur because of a faulty update could cost Millions in losses

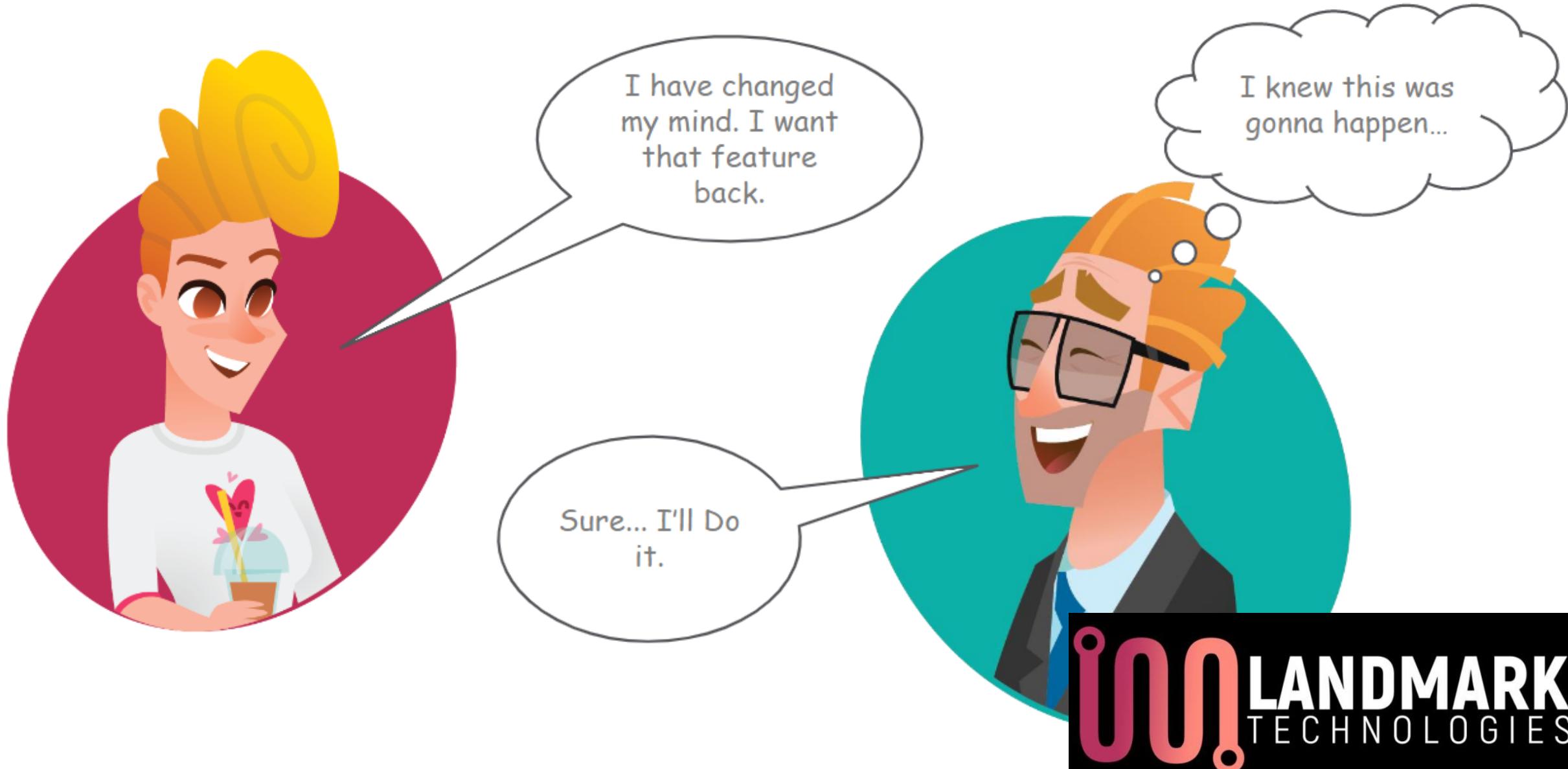


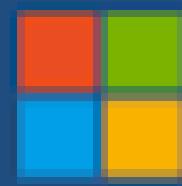
Same Scenario
With
**VERSION
CONTROL**

With Version Control: Disapproval From Client



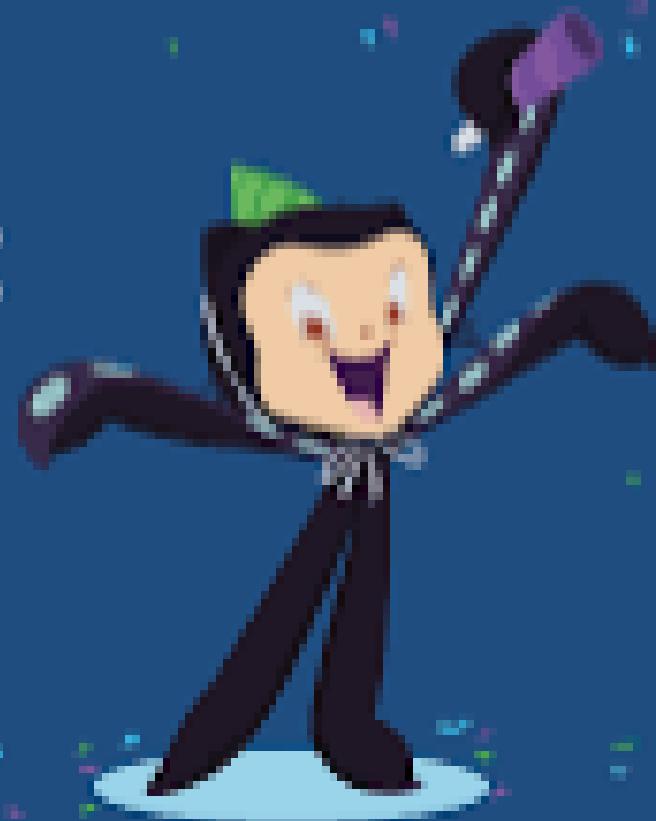
With Version Control: Client Wants The New Feature





Microsoft

GitHub

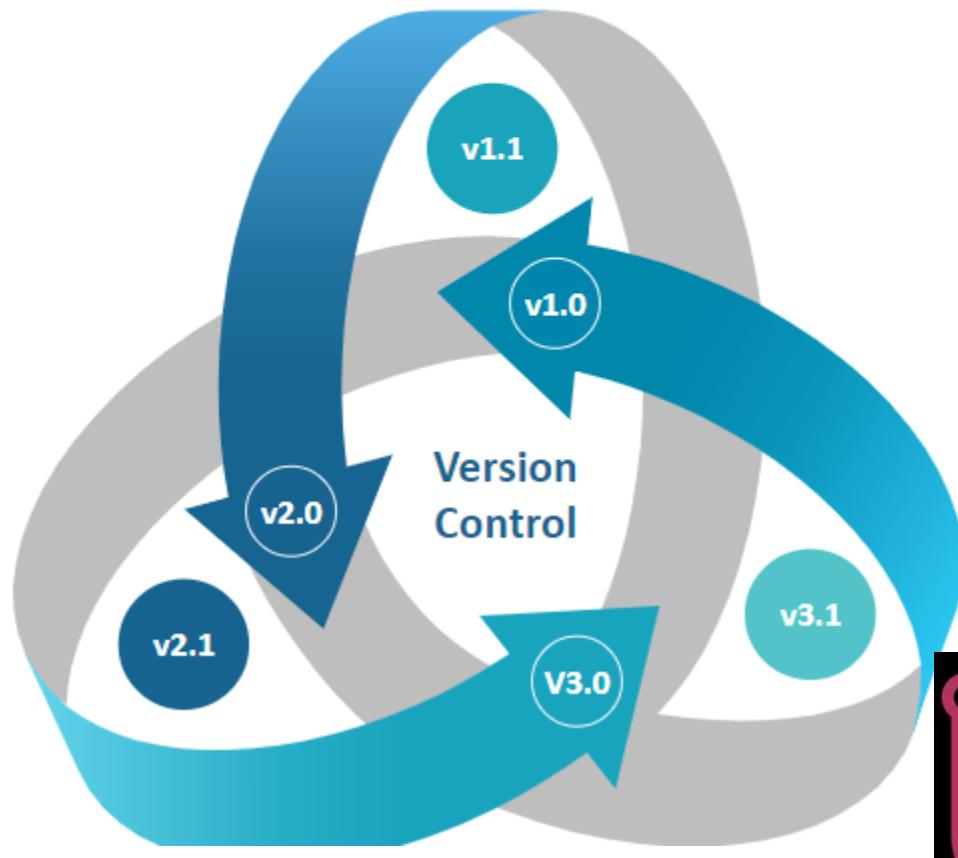


Version Control Saves The Day



What Is Version Control?

Version Control is a system that documents changes made to a file or a set of files. It allows multiple users to manage multiple revisions of the same unit of information. It is a snapshot of your project over time.

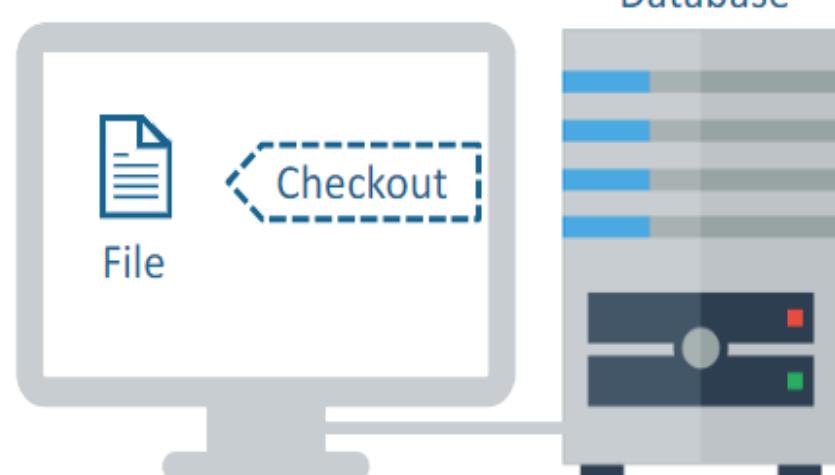


Version Control: Types

- 1 Local
- 2 Centralized
- 3 Distributed

Local Version Control (LVC)

- The practice of having the Version Database in the local computer
- Local database keeps a record of the changes made to files in version database

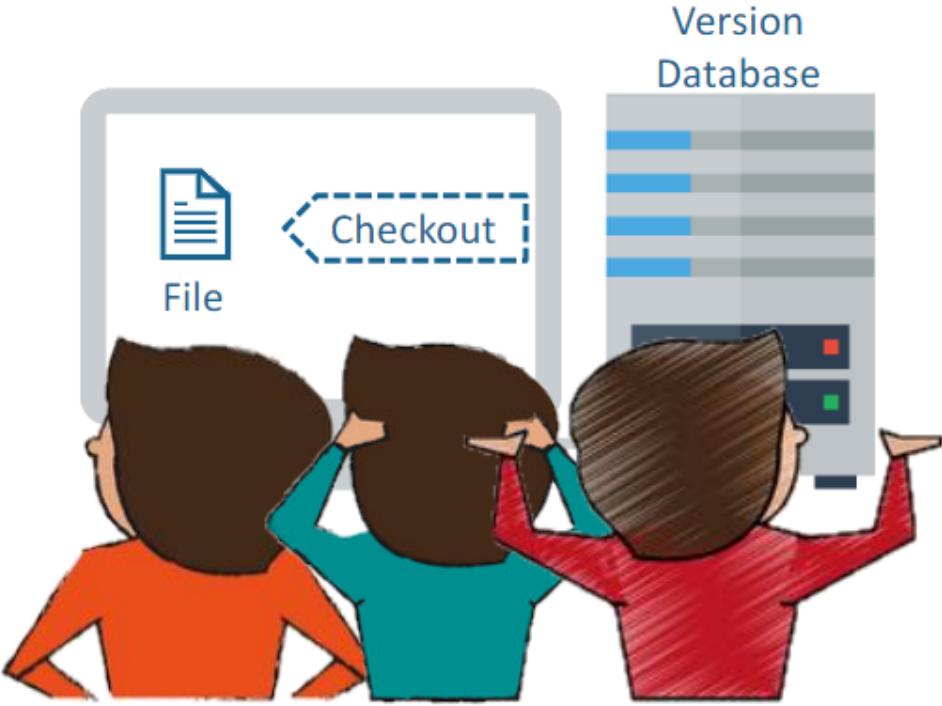


Version Control: Types

- 1 Local
- 2 Centralized
- 3 Distributed

Local Version Control: Issue

- **Issue:** Multiple people parallelly working on the same project
- **Solution:** Centralized Version Control

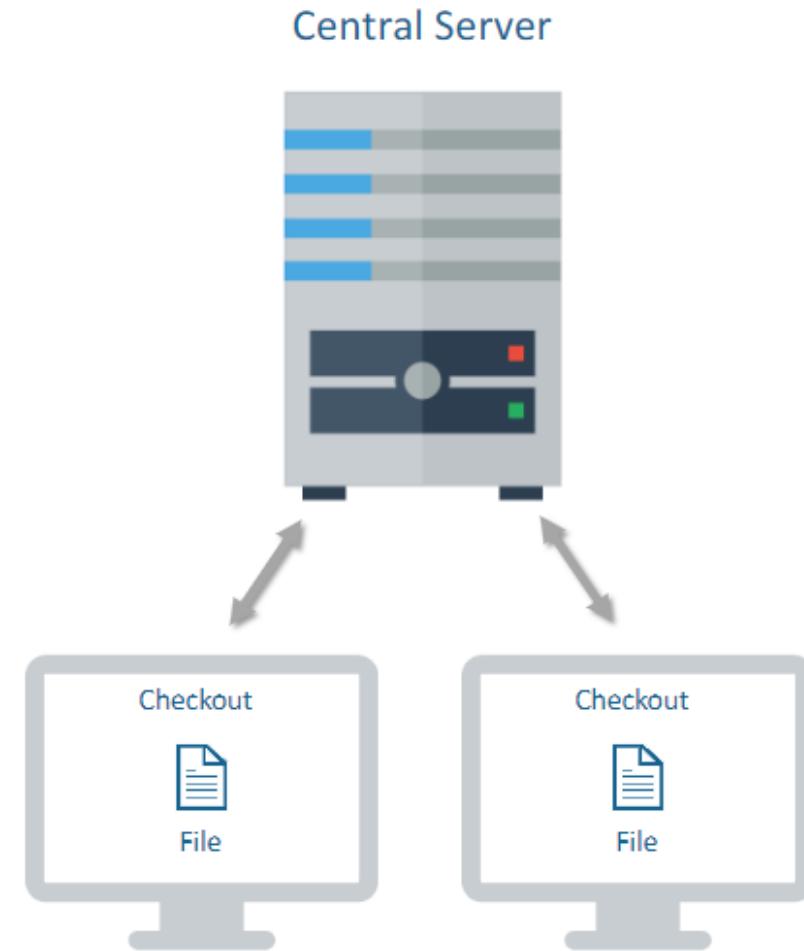


Version Control: Types

- 1 Local
- 2 Centralized
- 3 Distributed

Centralized Version Control (CVC)

- Local Version Control's issues are resolved by Centralized Version Control
- In CVC, a central repository is maintained where all the versioned files are kept
- Now users can checkout, and check-in files from their different computers at any time

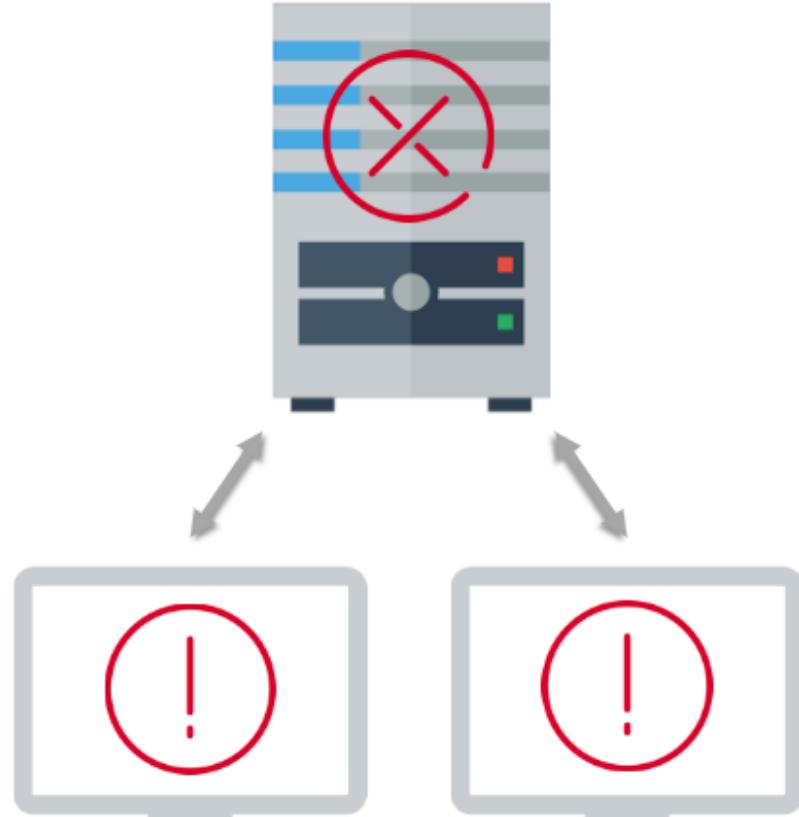


Version Control: Types

- 1 Local
- 2 Centralized
- 3 Distributed

Centralized Version Control: Issue

- **Issue:** In case of central server failure whole system goes down
- **Solution:** Distributed Version Control

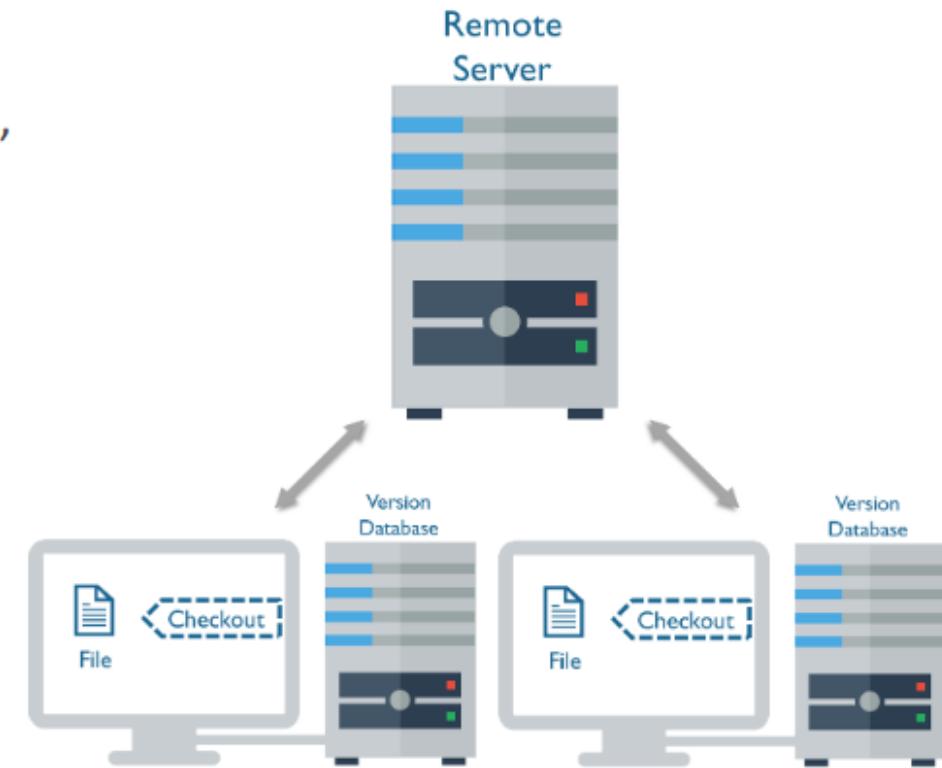


Version Control: Types

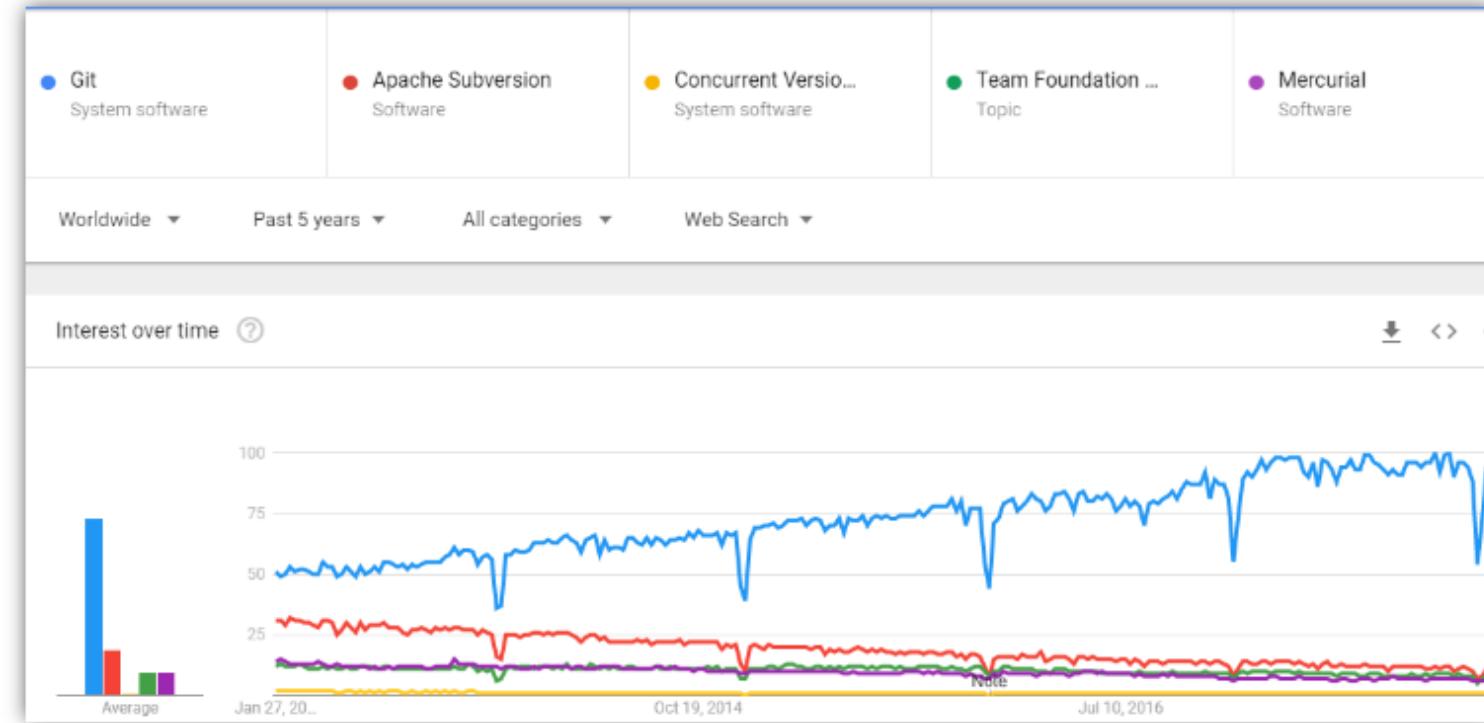
- 1 Local
- 2 Centralized
- 3 Distributed

Distributed Version Control

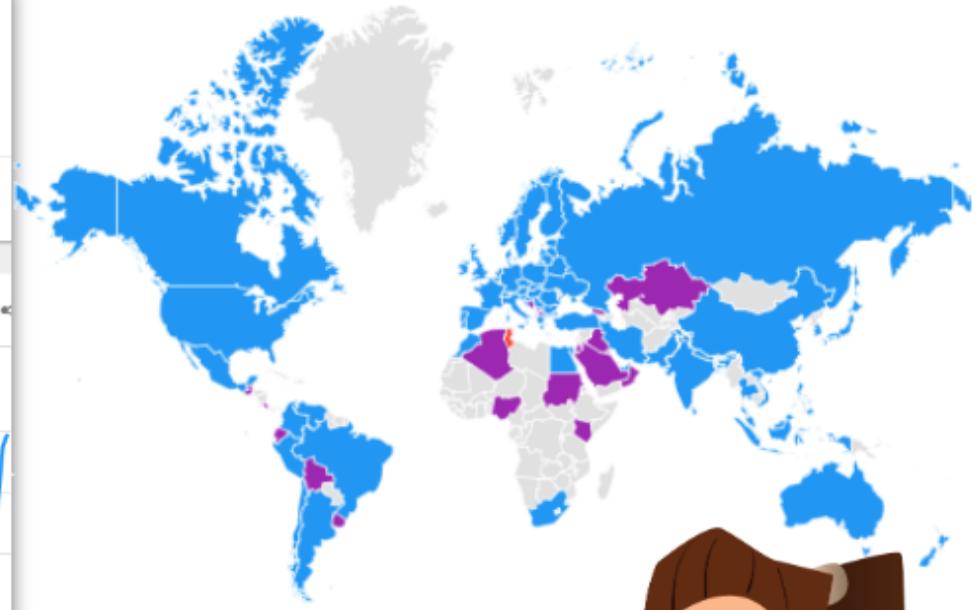
- Version Database is stored at every users' local system and at remote server
- Users manipulate the local files and then upload the changes to the remote server
- If any of the server dies, a client server can be used to restore



Why Git?



Note: The above graph shows the usability of the popular VCS tools in the past 5 years



Why Git is a Clear Winner?

Snapshots

Git records changes made to a file rather than file itself. That means if a file isn't changed it isn't stored again.



Distributed

Every user has his own copy of the repository data stored locally allowing full functionality even on disconnection.



Integrity

Check-sum before storing ensures that you can't make any changes to anything without Git recording that change.



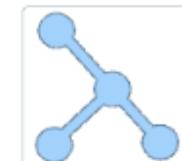
Fast Operations

Almost every operation on git is local, hence the speed offered by Git is lightening fast compared to other VCS's.



Branch Handling

Every collaborator's working directory is in itself a branch. Different branches can be merged with ease.



Robust

Nearly every task in Git is undo-able and it is really hard to loose any change or data in Git.



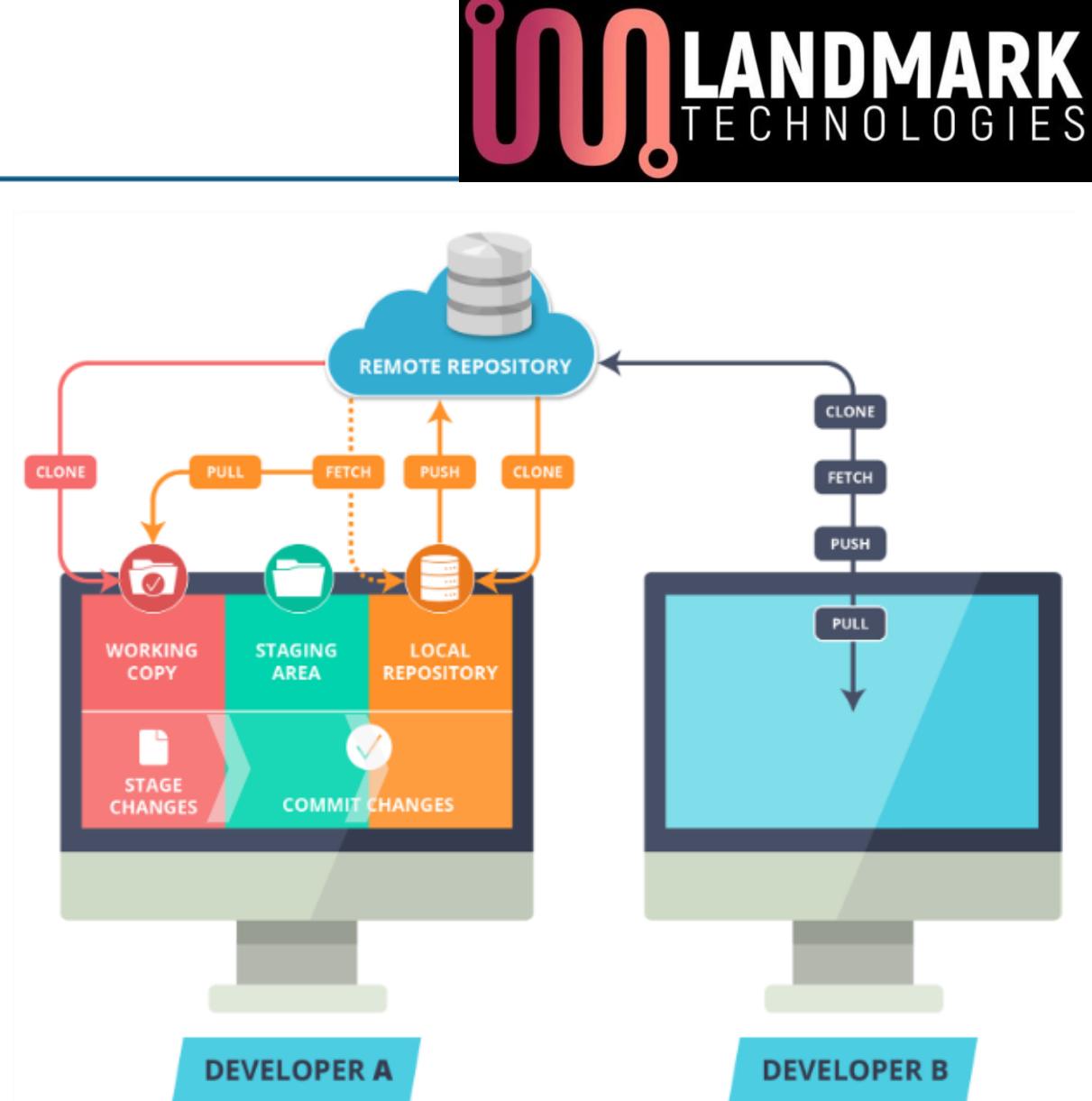
What is Git?

Git is an open source Distributed Version Control System(DVCS) which records changes made to the files laying emphasis on **speed, data integrity** and **distributed, non-linear workflows**



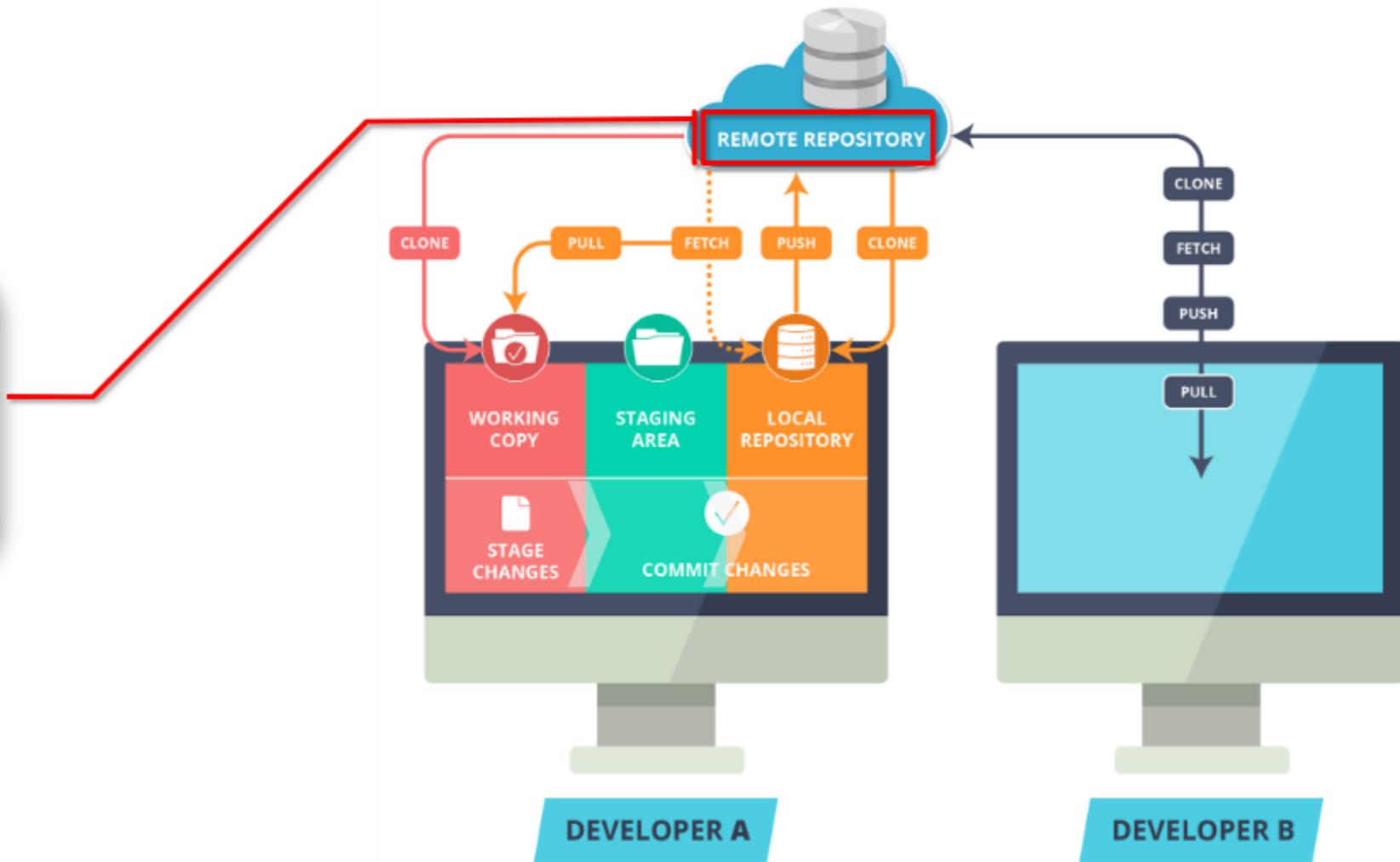
The Git File Workflow

- Use Git workflow to manage your project effectively
- Working with set of guidelines increases Git's consistency and productivity



The Git File Workflow

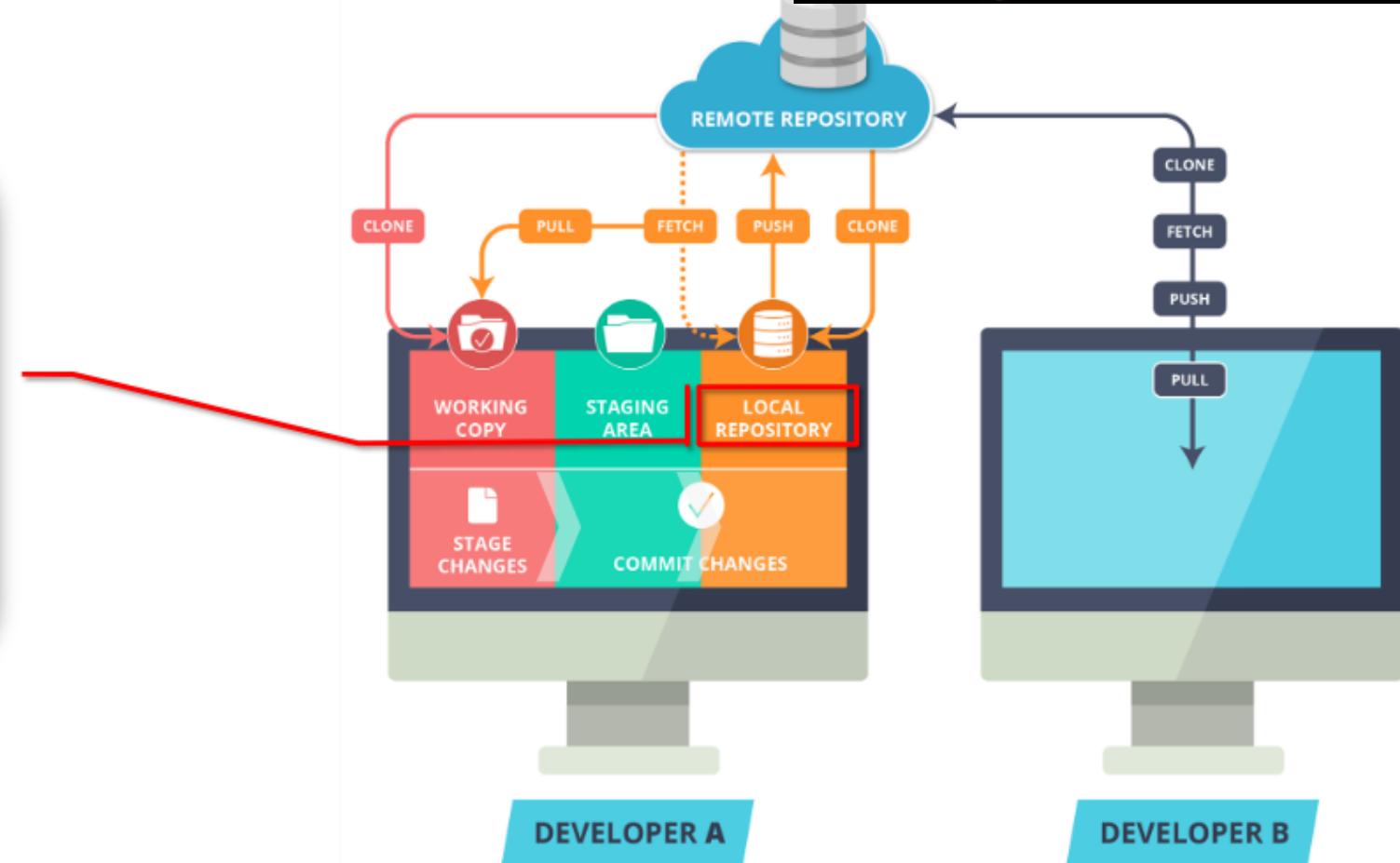
The Remote Repository is the server where all the collaborators upload changes made to the files



The Git File Workflow



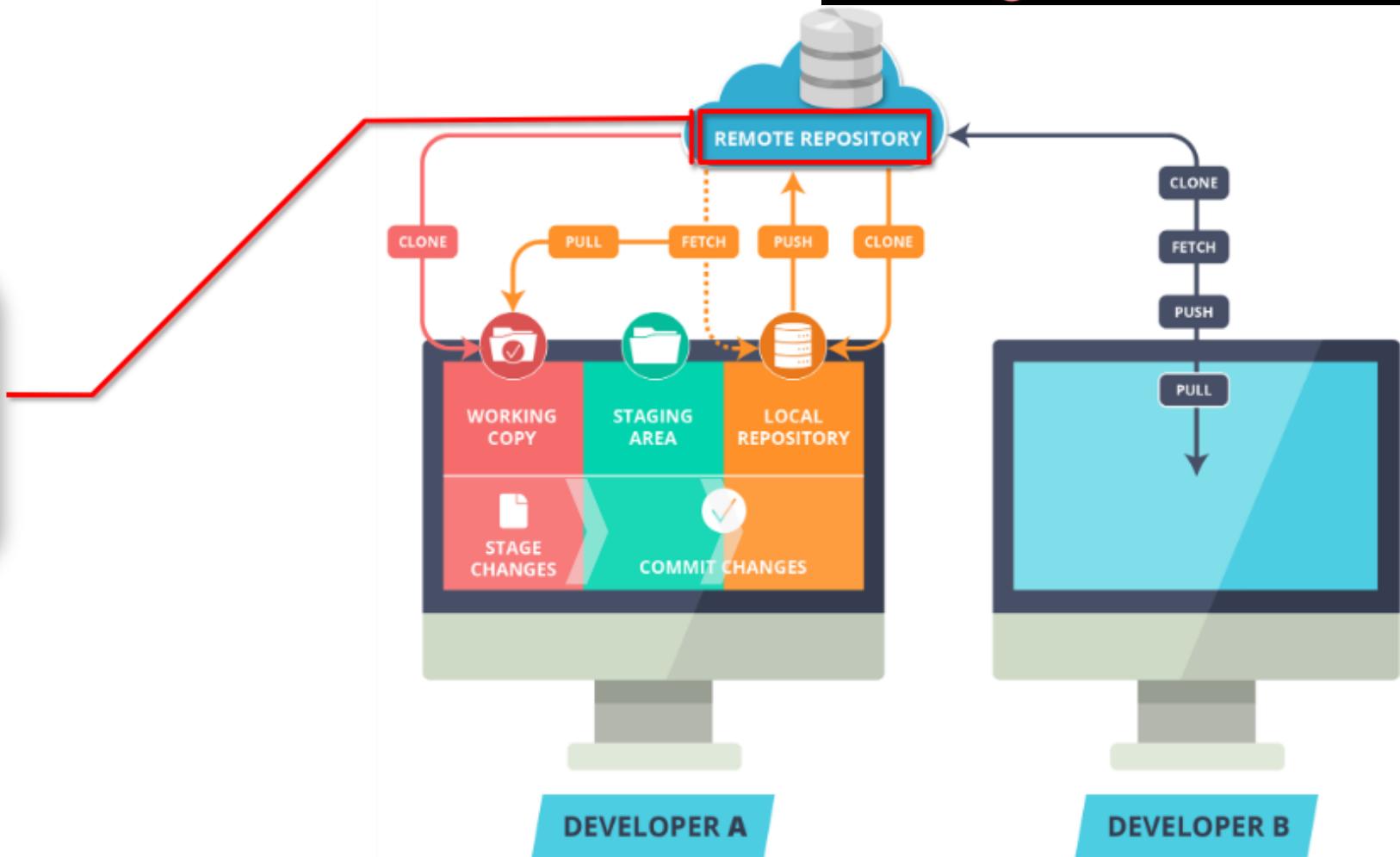
- “Local Repository” is user’s copy of the Version Database
- The user accesses all the files through local repository and then push the change made to the “Remote Repository”



The Git File Workflow



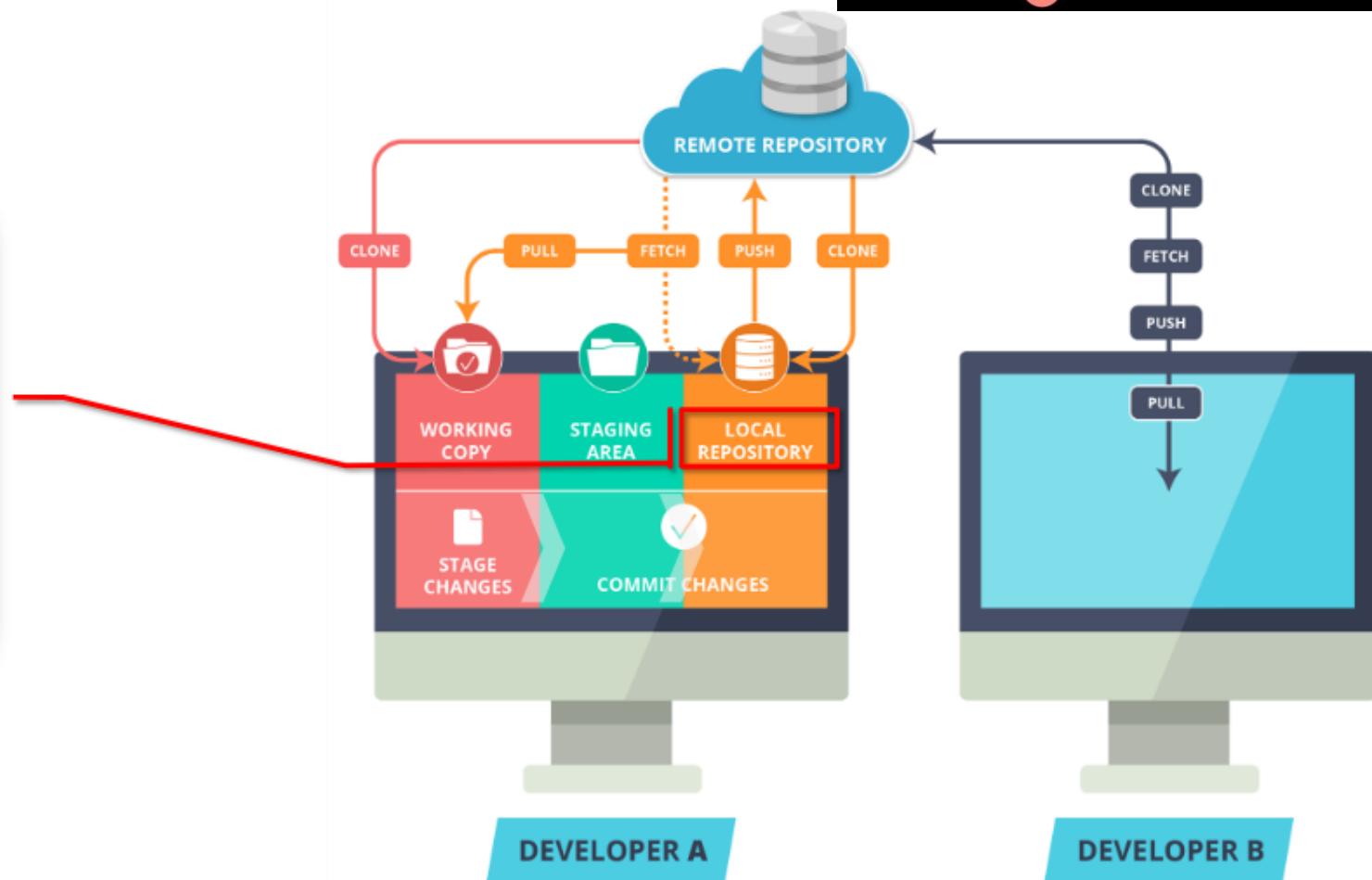
The Remote Repository is the server where all the collaborators upload changes made to the files



The Git File Workflow



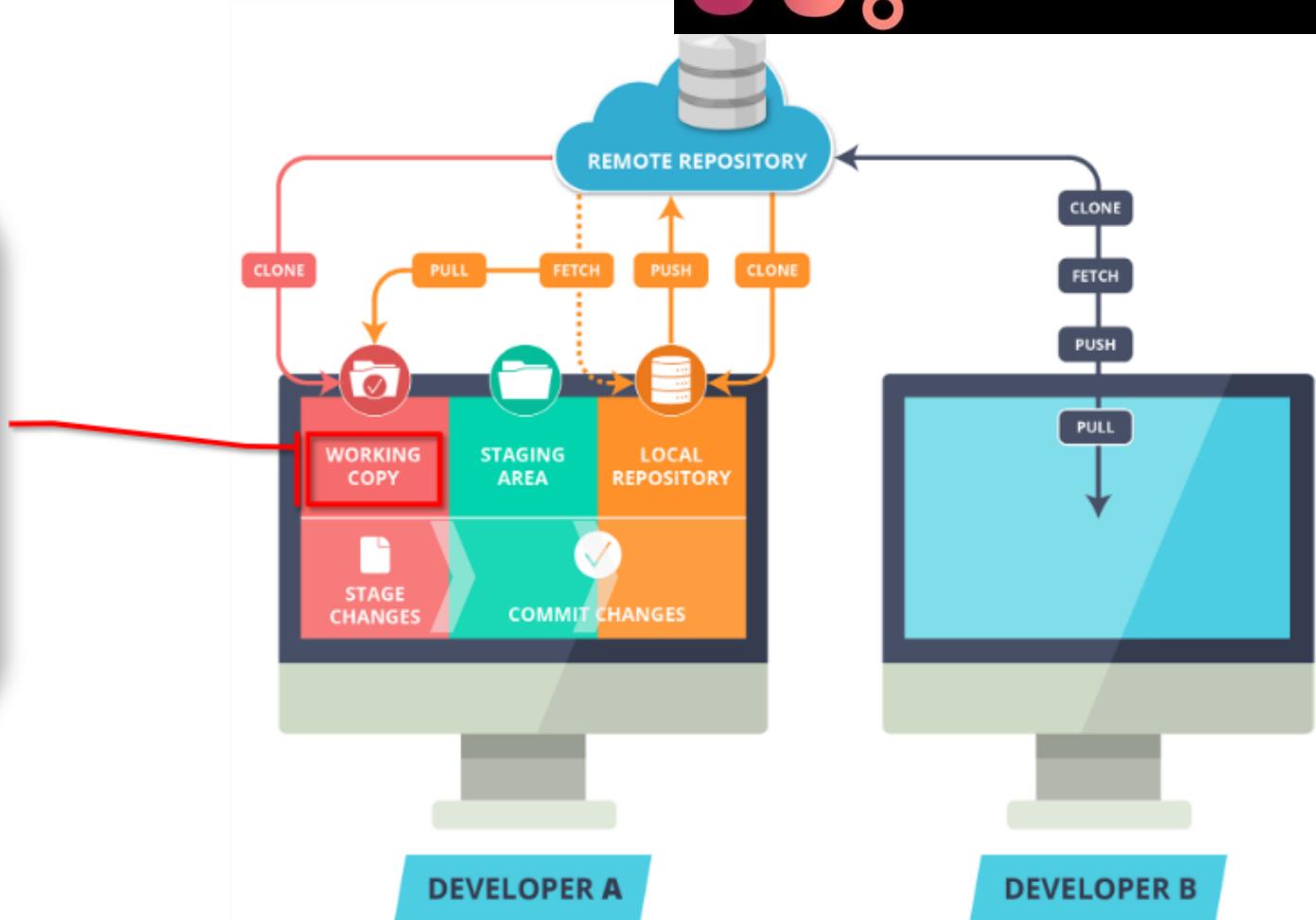
- “Local Repository” is user’s copy of the Version Database
- The user accesses all the files through local repository and then push the change made to the “Remote Repository”



The Git File Workflow



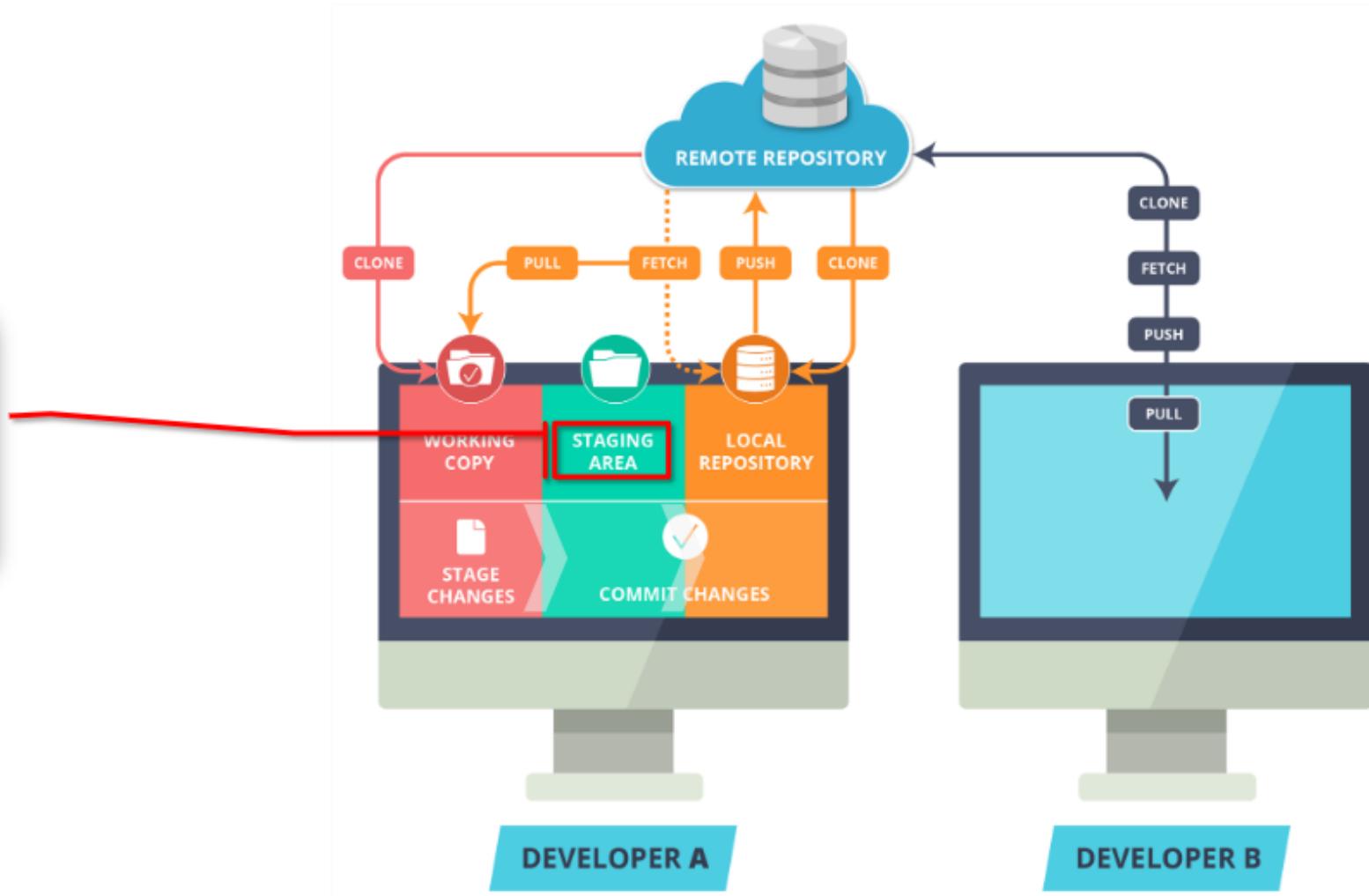
- “**Workspace**” is user’s active directory
- The user modifies existing files and creates new files in this space. Git tracks these changes compared to your Local Repository



The Git File Workflow



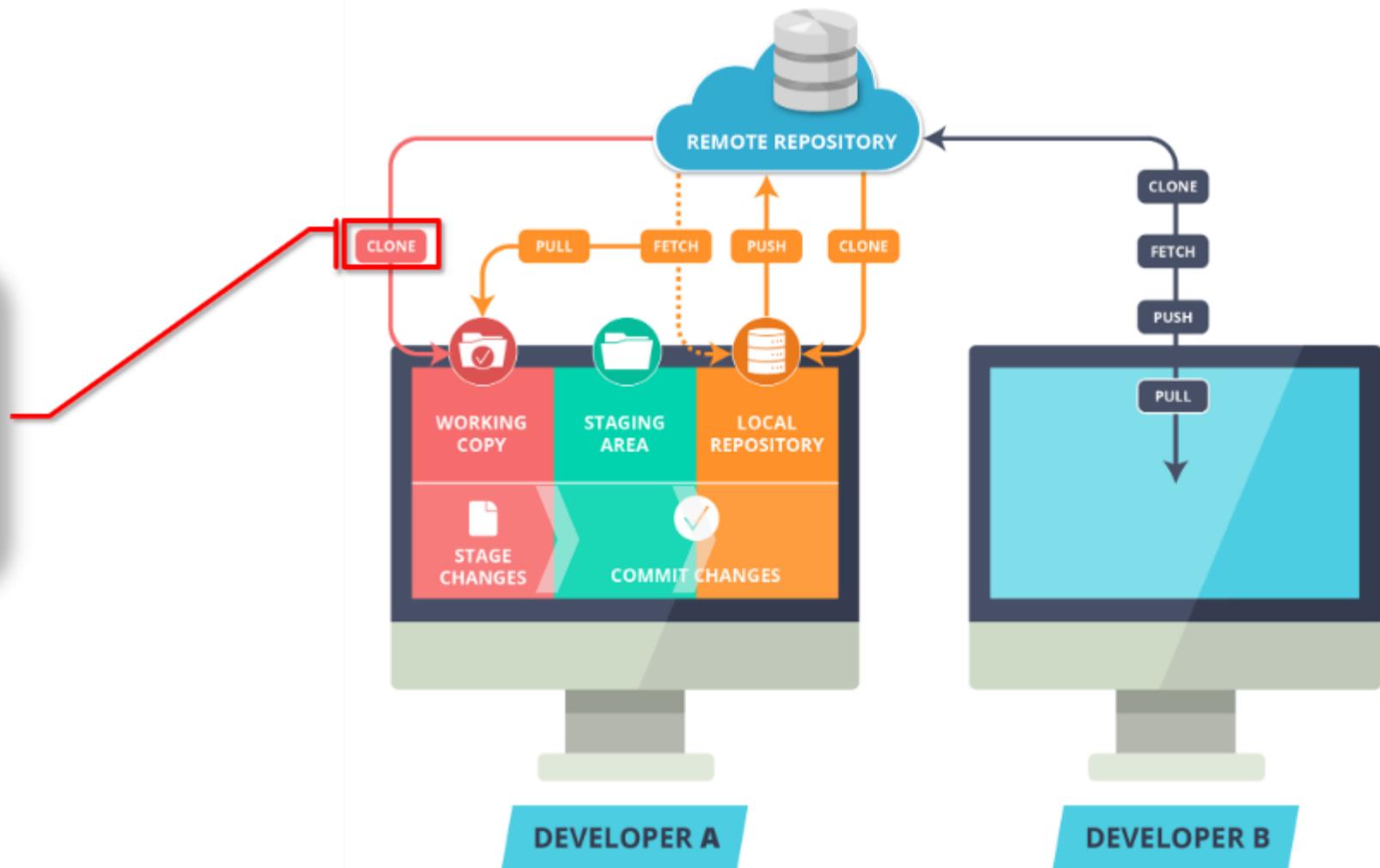
Stage is a place where all the modified files marked to be committed are placed.



The Git File Workflow

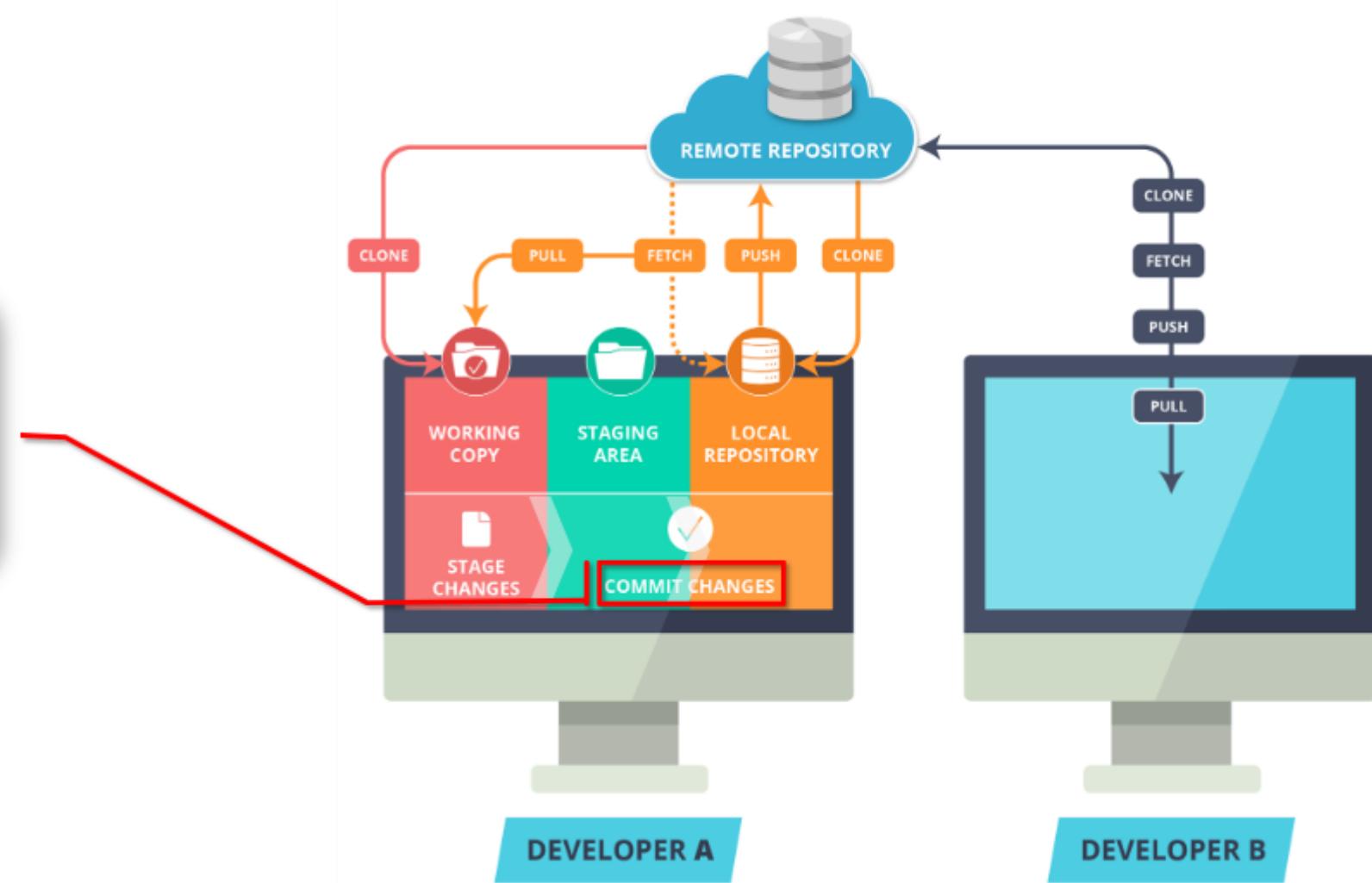


Clone command creates a copy of an existing Remote Repository inside the Local Repository.



The Git File Workflow

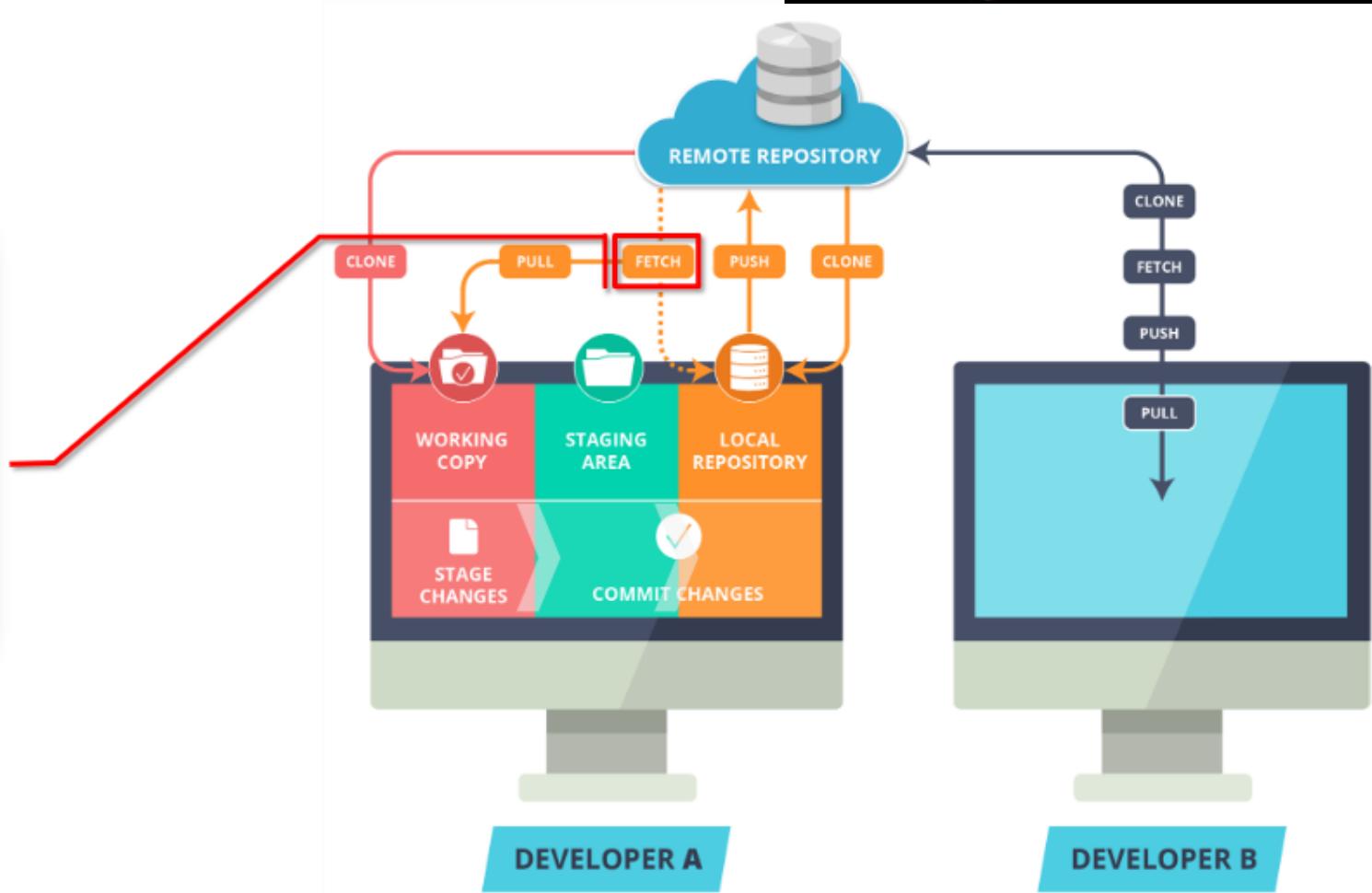
Commit command commits all the files in the staging area to the local repository.



The Git File Workflow



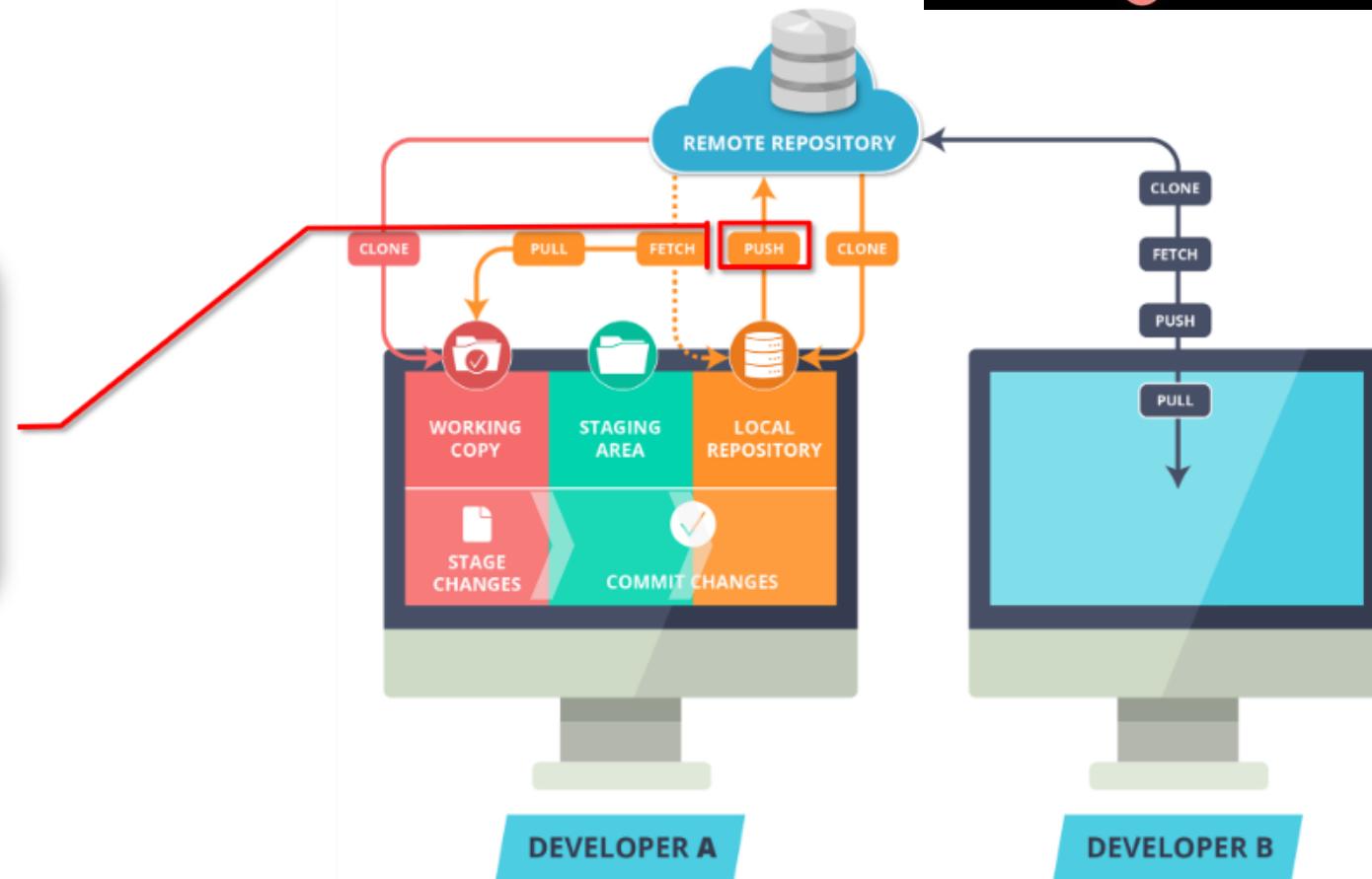
Fetch command collects the changes made in the Remote repository and copies them to the Local Repository. This command doesn't affect our Workspace.



The Git File Workflow

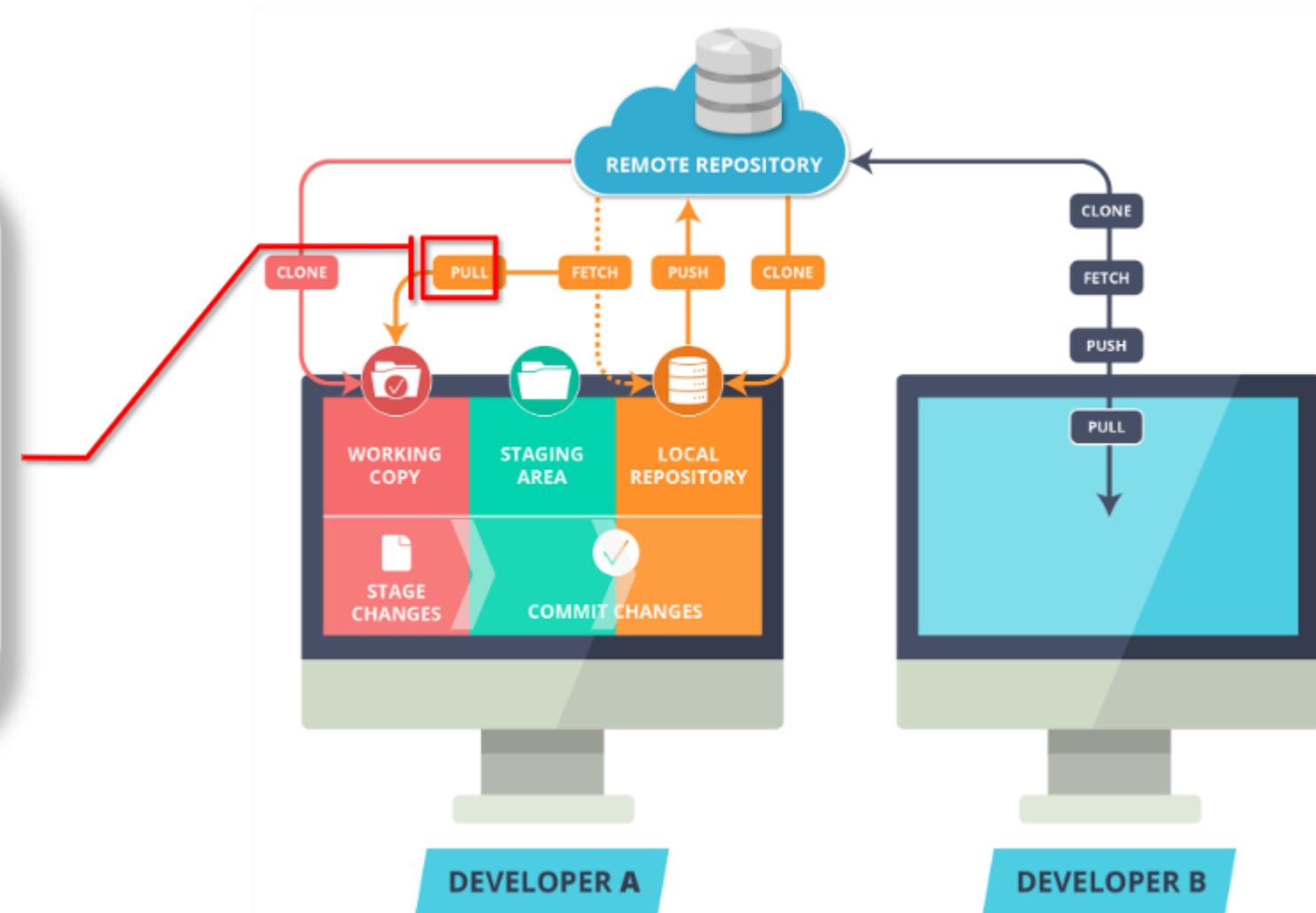


Push command pushes all the changes made in the Local Repository to the Remote Repository

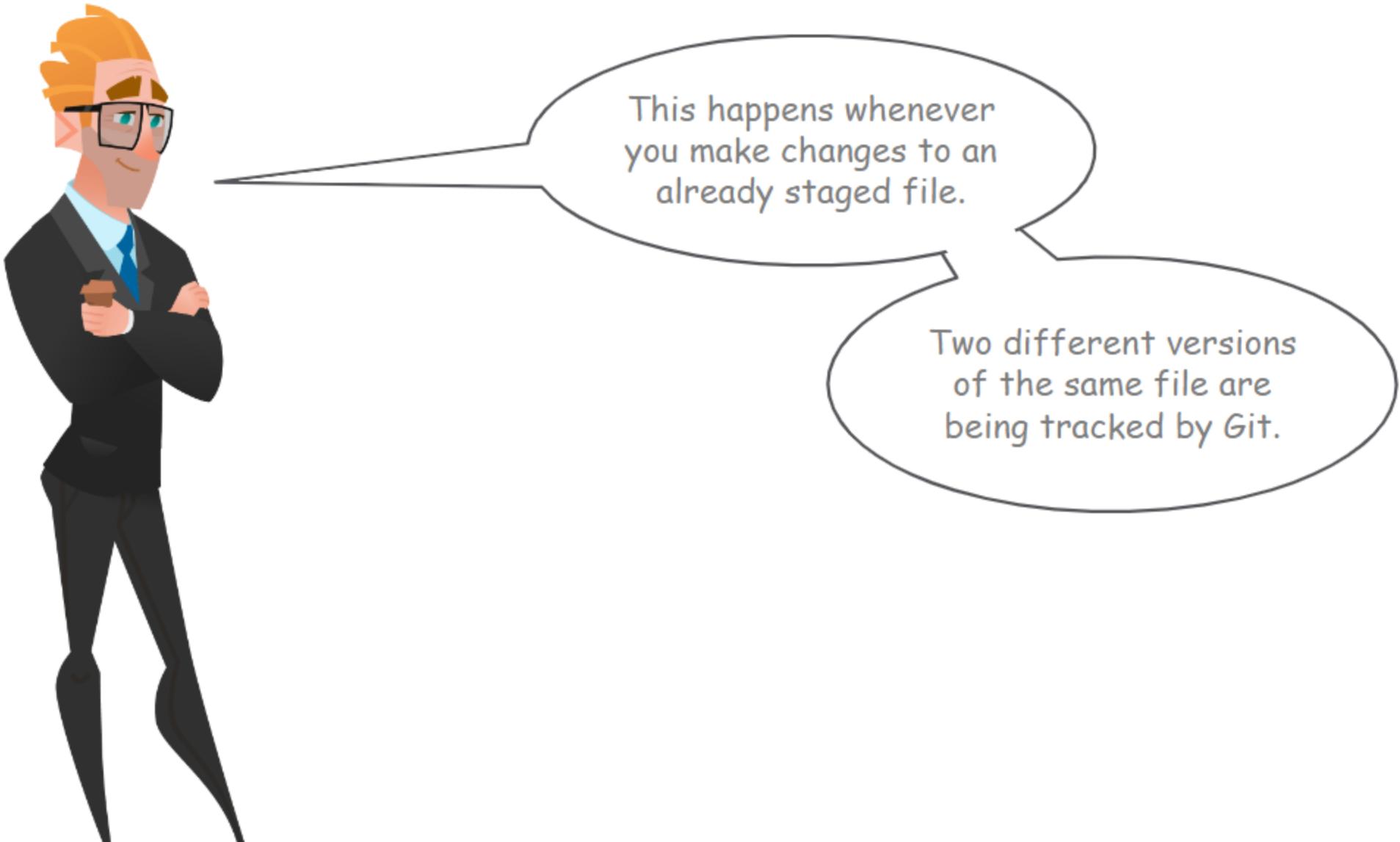


The Git File Workflow

- Pull like Fetch, gets all the changes from the remote repository and copies them to the Local Repository
- Pull merges those changes to the current working directory



Why a File is both staged and unstaged?



Removing Files From Repository



- The `git rm` command deletes the file from git repository as well as users system

Syntax: `git rm <filename>`

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git rm Demo.class
```

- To remove the file from git repository but not from the system `--cached` option

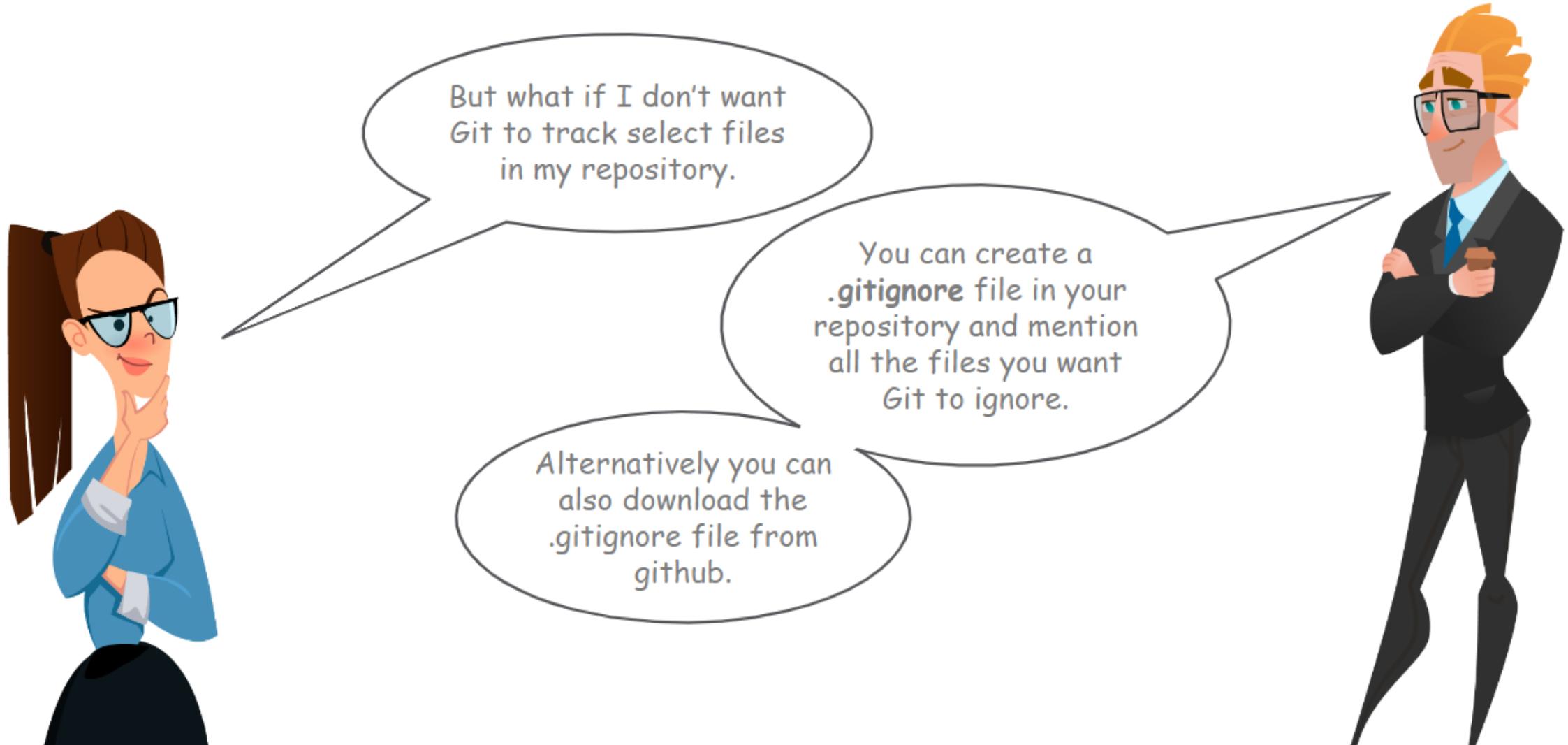
Syntax: `git rm --cached <filename>`

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git rm --cached Demo.class
rm 'Demo.class'
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

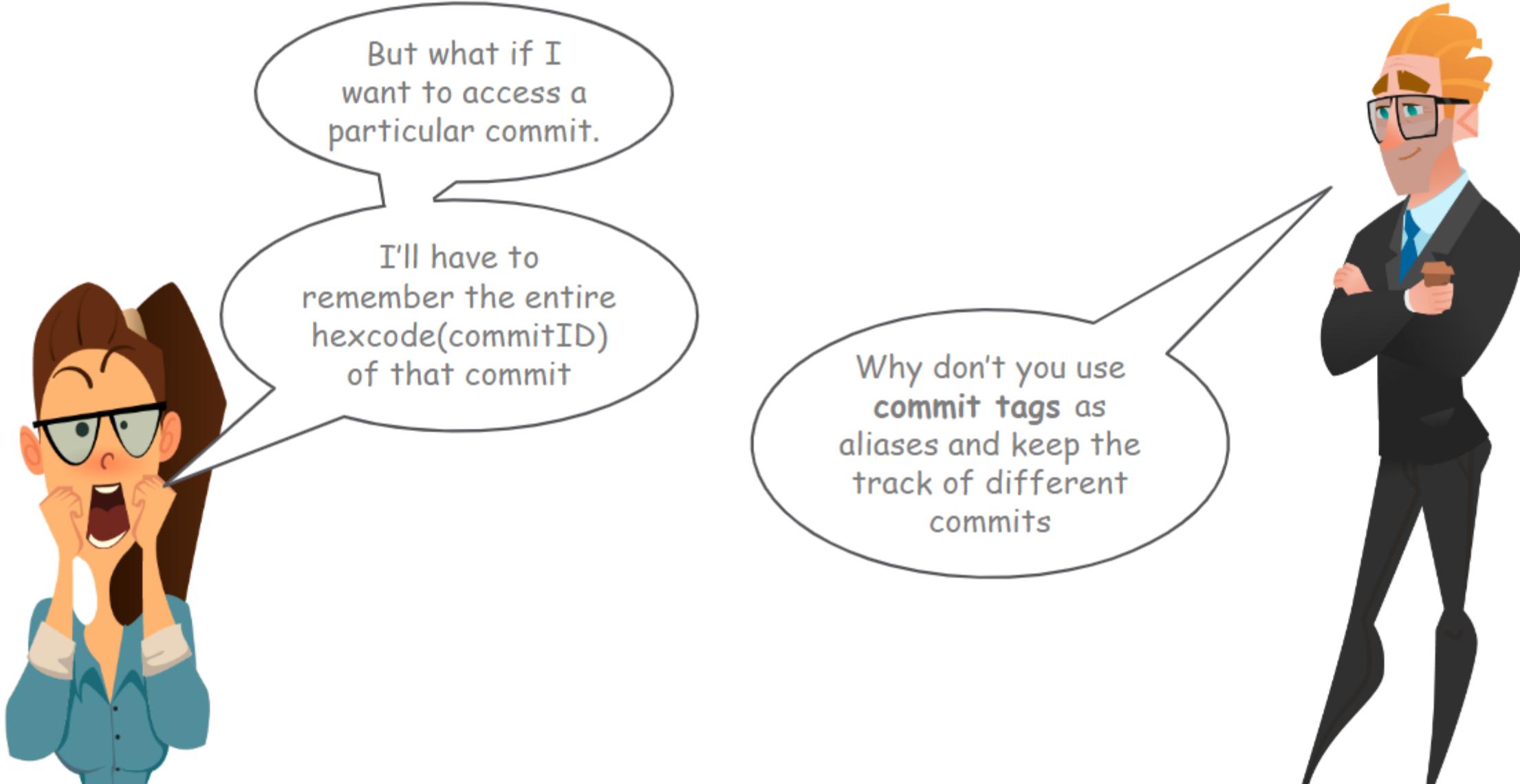
- An error shows up if you try to delete a staged file
- You can force remove a staged file by using `-f` flag

Syntax: `git rm -f <filename>`

Ignoring Files



Commit Tags



Commit: Git Tag

- Commit tags provide an alias for commitID

Syntax: git tag --a <annotation> --m <message>

- You can also view all the tags you have created

Syntax: git tag

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git tag -a v1.3 -m 'Version 1.3'  
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git tag  
v1.3  
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

Commit Tags



- Adding a tag to one of the previous commits

Syntax: `git tag --a <annotation> <commit id> --m <message>`

- Commit id can be obtained from git logs
- All these tags can be viewed in git

Syntax: `git show <tag-name>`

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git tag -a v0.3 11cf392d -m 'Added tag to a previous commit'
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git show v0.3
tag v0.3
Tagger: amrjeet <amrinderjeet.singh@edureka.co>
Date:   Wed Jan 31 11:51:32 2018 +0530

Added tag to a previous commit

commit 11cf392dc4307d8446fdc4419e8d9c3674549385
Author: amrjeet <amrinderjeet.singh@edureka.co>
Date:   Tue Jan 30 15:30:56 2018 +0530

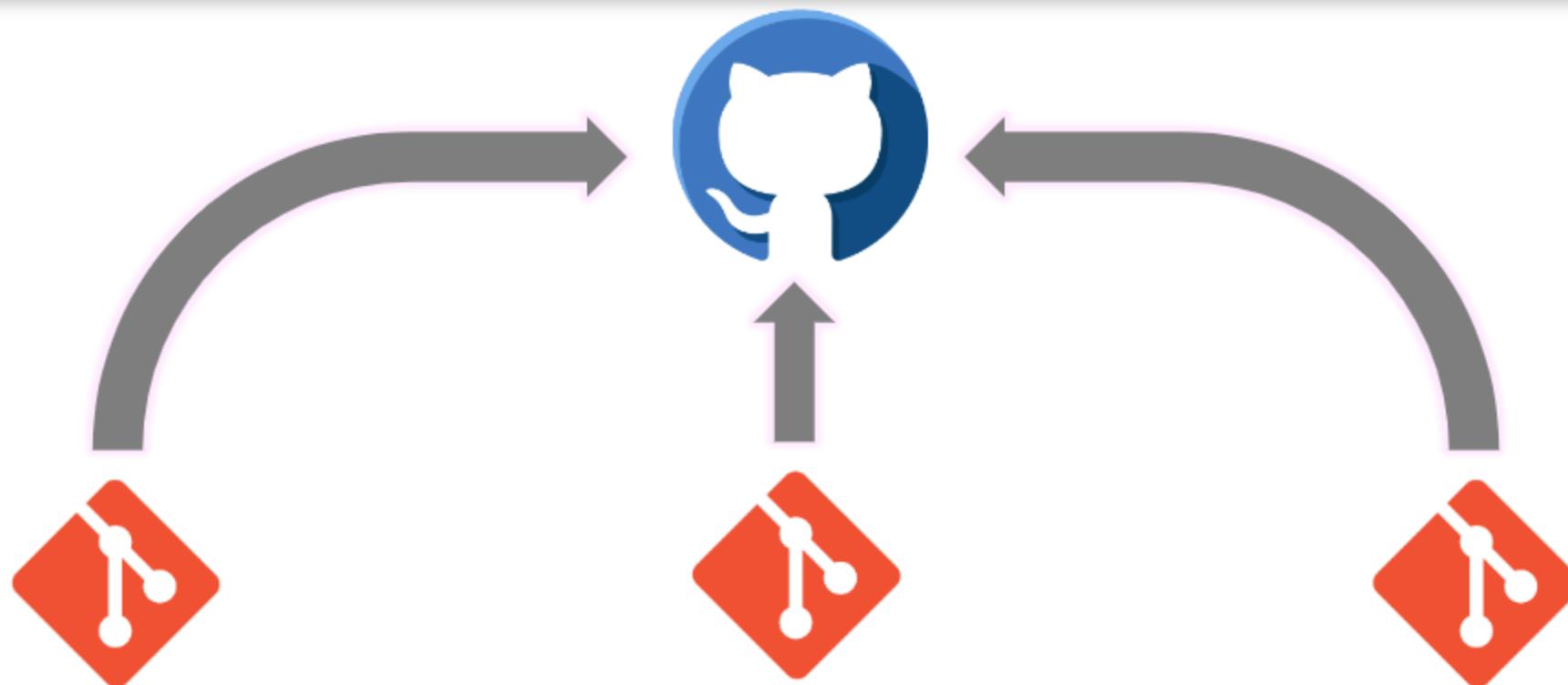
    removed .class extension files

diff --git a/Demo.class b/Demo.class
deleted file mode 100644
index fa53ecb..0000000
Binary files a/Demo.class and /dev/null differ
diff --git a/Demo2.class b/Demo2.class
deleted file mode 100644
index 81c13a3..0000000
Binary files a/Demo2.class and /dev/null differ
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

What Is A Remote Repository?

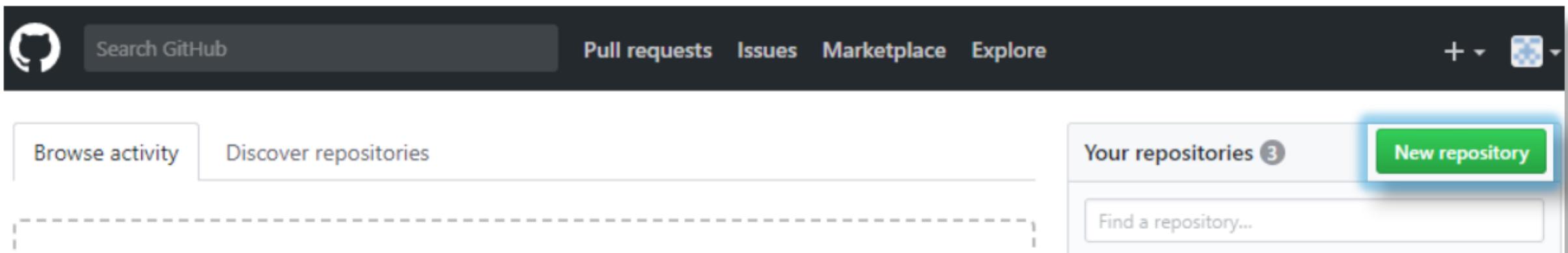


Mostly the users work on a local repository. But in order to collaborate with other people, we use a remote repository. A remote repository is place where the users upload and share their commits with other collaborators.



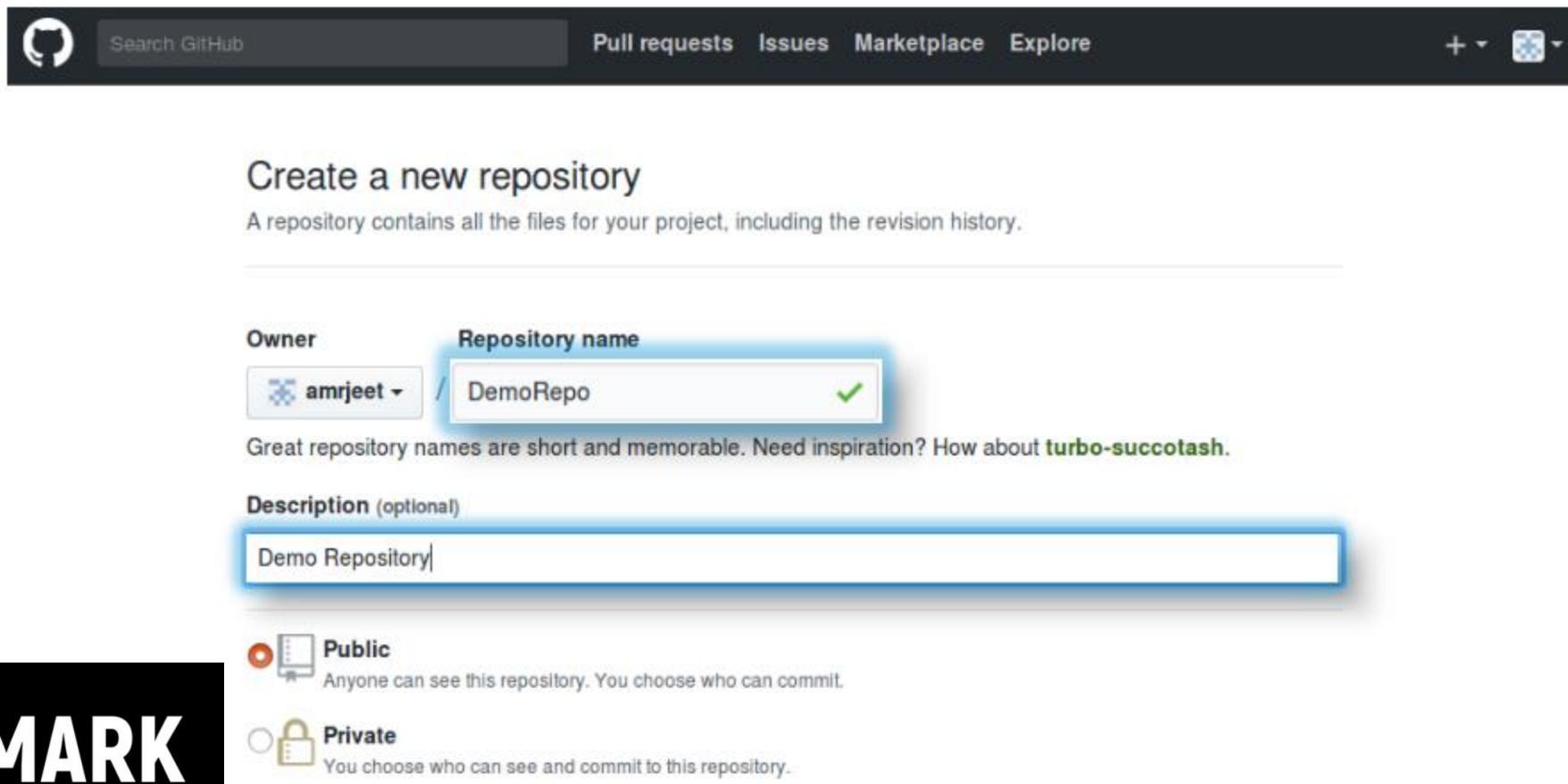
Creating A Remote Repository: New Repository

- Sign-up at github.com
- Click on New repository to create a new repository



Creating A Remote Repository: Adding Description

- Under **Repository name**, give a name to your repository
- Give some **Description** about your repository under **Description** section.



Creating A Remote Repository: Create Repository

- For a free repository choose public
- For a private repository a monthly premium needs to be paid
- Finally click on Create Repository

Description (optional)

Public
Anyone can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾ Add a license: **None** ▾ ⓘ

Create repository

Adding Remote To Local



To add Remote repository to local use **git add remote** followed by remote link

Syntax: `git add remote origin <remote link>`

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git remote add origin https://github.com/amrjeet/DemoRepo.git  
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

The screenshot shows a GitHub repository page for 'amrjeet / DemoRepo'. At the top, there's a navigation bar with links for 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the navigation bar, the repository name 'amrjeet / DemoRepo' is displayed, along with 'Watch 0', 'Star 0', and 'Fork 0' buttons. A horizontal menu bar includes 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Wiki', 'Insights', and 'Settings'. The main content area features a 'Quick setup — if you've done this kind of thing before' section. It contains a button to 'Set up in Desktop' or links for 'HTTPS' and 'SSH' with the URL 'https://github.com/amrjeet/DemoRepo.git'. A red arrow points from the text 'Remote link' to the HTTPS URL. Below this, a note says 'We recommend every repository include a README, LICENSE, and .gitignore.' At the bottom, there's a link to '...or create a new repository on the command line'.

Push Local Repository To Remote



To push **Local repository** to remote use **push** command

Syntax: git push origin master

Origin is used
as an alias for
your remote

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git push origin master
Username for 'https://github.com': amrjeet
Password for 'https://amrjeet@github.com':
Counting objects: 11, done.
Compressing objects: 100% (10/10), done.
Writing objects: 100% (11/11), 1.61 KiB | 0 bytes/s, done.
Total 11 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/amrjeet/DemoRepo.git
 * [new branch]      master -> master
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

Refers to master
branch in local repo

Pushing Tags



Tags can be pushed, viewed and shared on Remote

Syntax: `git push origin --tags`

A screenshot of a GitHub repository page titled "amrjeet / DemoRepo". The top navigation bar includes links for "Pull requests", "Issues", "Marketplace", and "Explore". Below the title, there are buttons for "Watch" (0), "Star" (0), and "Fork" (0). The main content area shows two releases: "v1.3" (9 minutes ago) and "v0.3" (5 minutes ago). Each release entry includes a commit hash, a "zip" link, and a "tar.gz" link.

Release Version	Published	Commit Hash	Downloads
v1.3	9 minutes ago	a26d84e	zip tar.gz
v0.3	5 minutes ago	11cf392	zip tar.gz

Push Local Repository To Remote



The changes can be seen in Remote repository

This screenshot shows a GitHub repository page for 'amrjeet / DemoRepo'. The repository has 3 commits, 1 branch, 0 releases, and 1 contributor. The latest commit was made 11 hours ago. The commits added or modified 'Demo.java' and 'Demo2.java' files. A call-to-action at the bottom encourages adding a README file.

amrjeet / DemoRepo

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Demo Repository Add topics Edit

3 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

amrjeet removed .class extension files Latest commit 11cf392 Jan 30, 2018

Demo.java New files added and Demo.java modified 11 hours ago

Demo2.java New files added and Demo.java modified 11 hours ago

Add a README

Working On Remote



Files can be created and edited on remote

A screenshot of a GitHub repository page. At the top, there are buttons for 'Branch: master ▾', 'New pull request', 'Create new file' (which is highlighted with a blue border), 'Upload files', 'Find file', and 'Clone or download ▾'. Below this, a commit message from 'amrjeet' is shown: 'removed .class extension files'. To the right, it says 'Latest commit 11cf392 Jan 30, 2018'. Below the commit message, a file named 'Demo.java' is listed with the status 'New files added and Demo.java modified' and a timestamp '11 hours ago'.

A screenshot of a GitHub repository page for 'amrjeet / DemoRepo'. The repository has 0 issues, 0 pull requests, 0 projects, and 0 wiki pages. It has 0 stars and 0 forks. The 'Code' tab is selected. A search bar shows 'DemoRepo / RepoFile' with the placeholder 'or cancel'. Below the search bar, there are buttons for 'Edit new file' (selected) and 'Preview'. The preview area contains the text: '1 This is just an example'.

Working On Remote



These files can then be committed on the remote

The screenshot shows a GitHub interface. At the top, a modal window titled "Commit new file" is open, containing a message box with the text "Added and committed file from remote repository" and a larger text area for an optional extended description. Below these are two radio button options: one selected ("Commit directly to the master branch") and another ("Create a new branch for this commit and start a pull request"). At the bottom of the modal are "Commit new file" and "Cancel" buttons. Below the modal, the main repository page for "Demo Repository" is visible. It shows 4 commits, 1 branch, 0 releases, and 1 contributor. The contributor is listed as "amrjeet". The commit history includes three entries: "Added and committed file from remote repository" (by amrjeet, Jan 31, 2018), "New files added and Demo.java modified" (by amrjeet, 12 hours ago), and "New files added and Demo.java modified" (by amrjeet, 12 hours ago). A fourth commit, "Added and committed file from remote repository" (by amrjeet, just now), is highlighted with a blue border. The repository page also features standard navigation buttons like "Edit", "Branch: master", "New pull request", and "Clone or download".

File	Description	Time
Demo.java	New files added and Demo.java modified	12 hours ago
Demo2.java	New files added and Demo.java modified	12 hours ago
RepoFile	Added and committed file from remote repository	just now

Remote List



To list all the remotes attached to your Local repository

Syntax: git remote -v

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git remote -v
origin  https://github.com/amrjeet/DemoRepo.git (fetch)
origin  https://github.com/amrjeet/DemoRepo.git (push)
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ █
```

Fetch

- **Fetch** command copies the changes from remote to local repository

Syntax: git fetch origin

- Fetch **does not** affect the present working directory

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git fetch origin
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ ls
Demo2.class  Demo2.java  Demo.class  Demo.java
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

Present working
directory not affected

Pull

- Pull copies all the changes from remote to local repository
- It then merges the changes with the present working directory

Syntax: git pull origin

```
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ git pull origin
From https://github.com/amrjeet/DemoRepo
 * branch            HEAD      -> FETCH_HEAD
Updating 11cf392..6ffe2e
Fast-forward
 RepoFile | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 RepoFile
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$ ls
Demo2.class  Demo2.java  Demo.class  Demo.java  RepoFile
amrinderjeet@amrinderjeet-VirtualBox:~/Documents/demo$
```

Working directory merged
with pulled repository

Thanks.