Git and Github

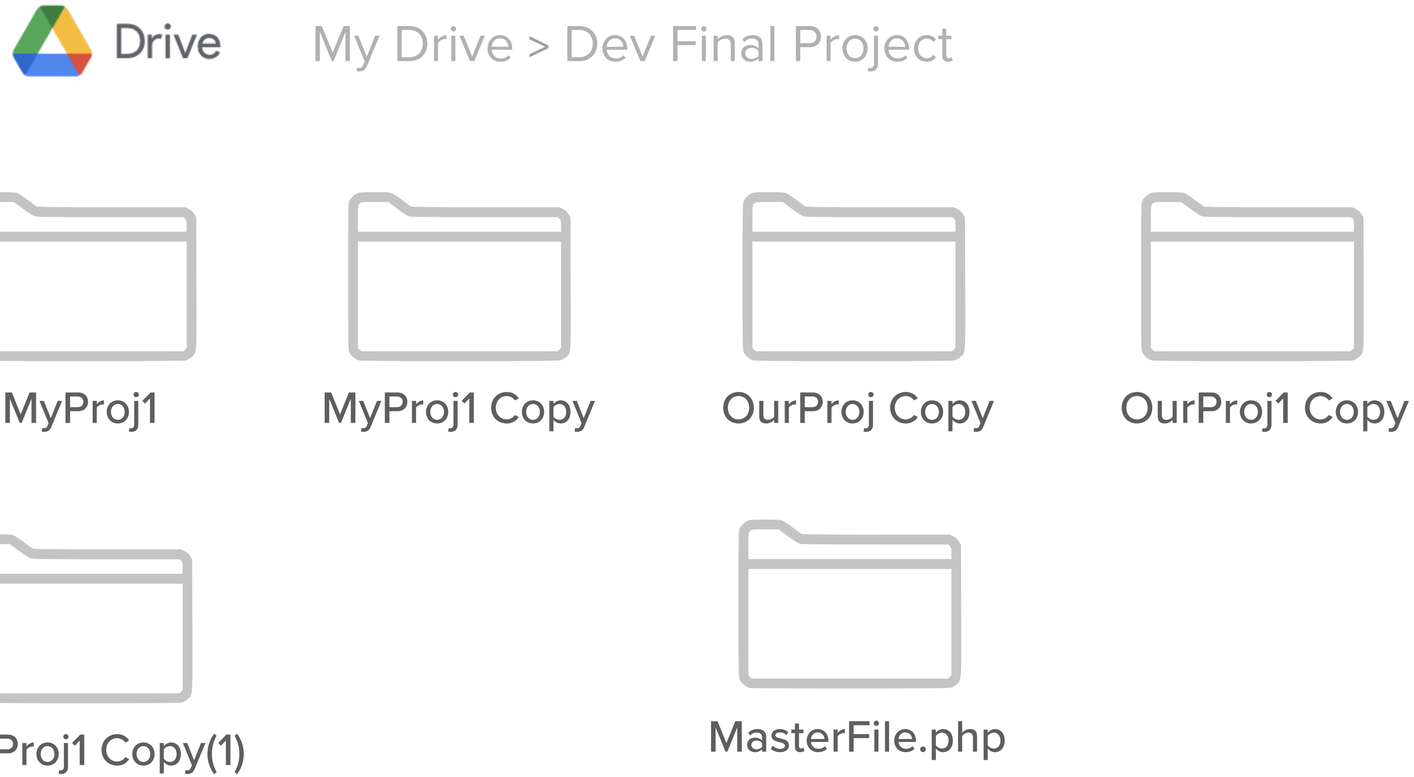
Anupam Batra

Suppose you want to build a website.....

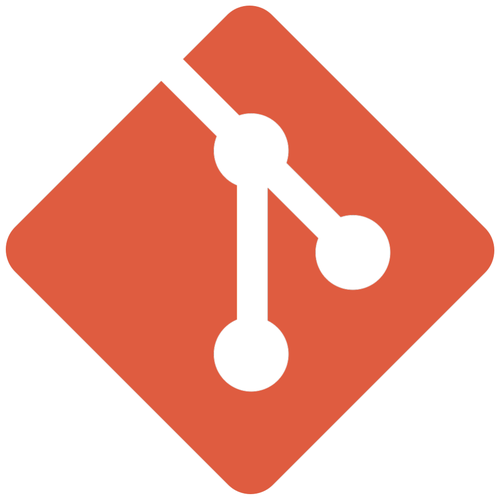
(or anything really)

- Well, you probably will need more than 1 iteration (in fact, likely a lot more than 1) and you'll probably want to keep track of those changes
- You'll want to collaborate with a team, and allow each person to make edits eventually, you'll want to merge all those changes
- And, you'll want a central (potentially public?) location to save those files so that you can access them later

That might end up looking a little something like this



Perhaps a little chaotic....



Enter Git

An open source version control system (VCS) that lets you manage and keep track of your code history

(Git takes care of versioning for you, and allows you to easily collaborate and merge files. Generally speaking, it takes care of tracking changes between files and allows you to merge/revert as desired)

Git

VS

Github

Command line tool invented in 2005

Run and maintained on your local system

Provides core VCS functionality

“Hub for Git” invented in 2008 (now owned by MSFT)

Hosted online through the cloud

Built in user management + extra features

Basically, a tool for managing your “Git”, and so much more

Repository (Repo)

A set of files or folders for a given project (usually the top level folder)

This is sort of like the ‘project name’

Important Terminology + Functions

Commit

A snapshot of your code folder at the time of the commit

Basically saying LOCALLY SAVE the current state of my code, and commit to memory the changes I've made

This is a LOCAL SAVE, and does not change items on the server

Each requires a commit message, summarizing the changes you've made

Important Terminology + Functions

Push

Sends your commit (or commits) to the server repository

A push requires a commit before it, but you can often “commit and push” together

A push is how you actually get your content to the server!

Important Terminology + Functions

Fetch / Pull

Fetch gathers any commits from the branch you are working on. Pull will fetch and update your local codebase

Fetch will check for/grab any changes, while pull is required to make those changes show up in your local code

Basically, fetch is to commit as push is to pull

Important Terminology + Functions

Branch

A fork from your ‘primary’ code. Creates a duplicate of your code that you can then independently edit

Any changes you make in a branch won’t affect the main codebase

This is great for bug fixes, teamwork, and new features (without messing up known working code)

Pull Request

Open a pull request to start the process of merging a branch with another

These let you tell others the changes you've made, and allows them to be reviewed

Eventually, someone will be responsible for understanding and approving the changes, to merge them into the main branch

Important Terminology + Functions

Checkout

How you switch between different branches

You ‘check out’ the appropriate branch you need to work on

For PHPStorm - make sure you checkout a “Remote Branch,” not a local one!

Overview

Clone	Copy from a cloud repository to your computer (local files)
Fetch + Pull	Get files from a cloud repo and add them to your computer (local files)
Commit + Push	Save changes to your computer (local files) and add them to the cloud repo
Branch	Duplicate and split a main repo as a playground (making an editable copy)
Pull Request	Ask for a branch to be merged back into the main codebase
Checkout	Switch between branches (different, copied versions of code) for you to work on

Demo Time

Recommended Workflow (High Level)

1. Configure your repository on Github online
   Make sure you share it with everyone!
2. Clone your repo via Git Desktop, and build your file structure
3. Code, commit, push, pull
4. Pull Request and Merge