VERSION CONTROL

ACM-W Code Jam Resources

LEARNING OBJECTIVES

01

GIT BASICS

Understand how to take a real project and integrate it with Git/GitHub.

02

AESTHETICS

Write well-crafted commits that’s intuitive and informative

03

GIT WORKFLOW

How everything comes together to peacefully develop

Introduction to Version Control

→ **Save** and document edits (Git)

→ **Review** previous versions (Git)

→ **Backup** your code remotely (GitHub)

→ **Share** and **discover** code (GitHub)

Introduction to Version Control

→ MacOS / Unix

a. install brew from brew.sh
b. $ brew install git

→ Windows

a. Install “Git” from https://git-scm.com/
b. Heaven-forbid do not use Github Desktop :)

GIT STATUS

For short, “gst” on terminal macOS

Find out what changes have been made locally on your workstation. What branch you’re on, and any warnings that Git is imposing.

GIT ADD .

Stages or adds all files to be ready for commit

The “.” means every changed file in your local directory. Can also add individual files. “*” wildcards are also allowed. Most commonly used is “git add .” to stage all changes.

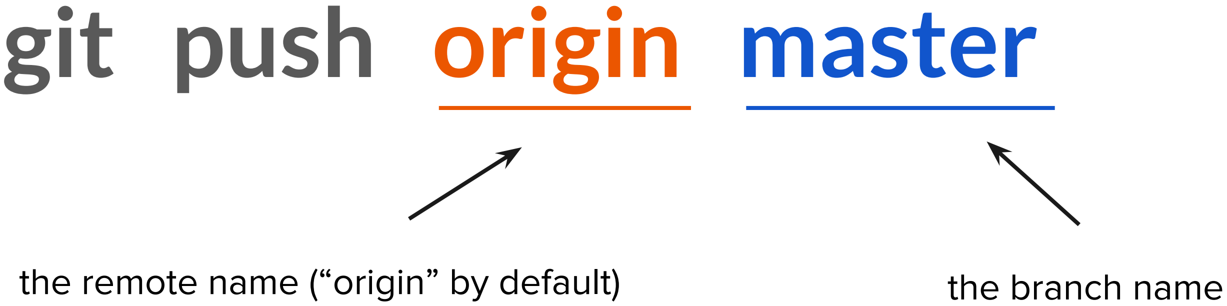
GIT COMMIT -M “<insert>”

Saves staged changes into a saved *clump* that’s ready to be pushed online to Github.

GIT PUSH ORIGIN <insert>

Pushes all committed changes to online repository

<insert> denotes a branch name. Origin denotes the remote name on Github

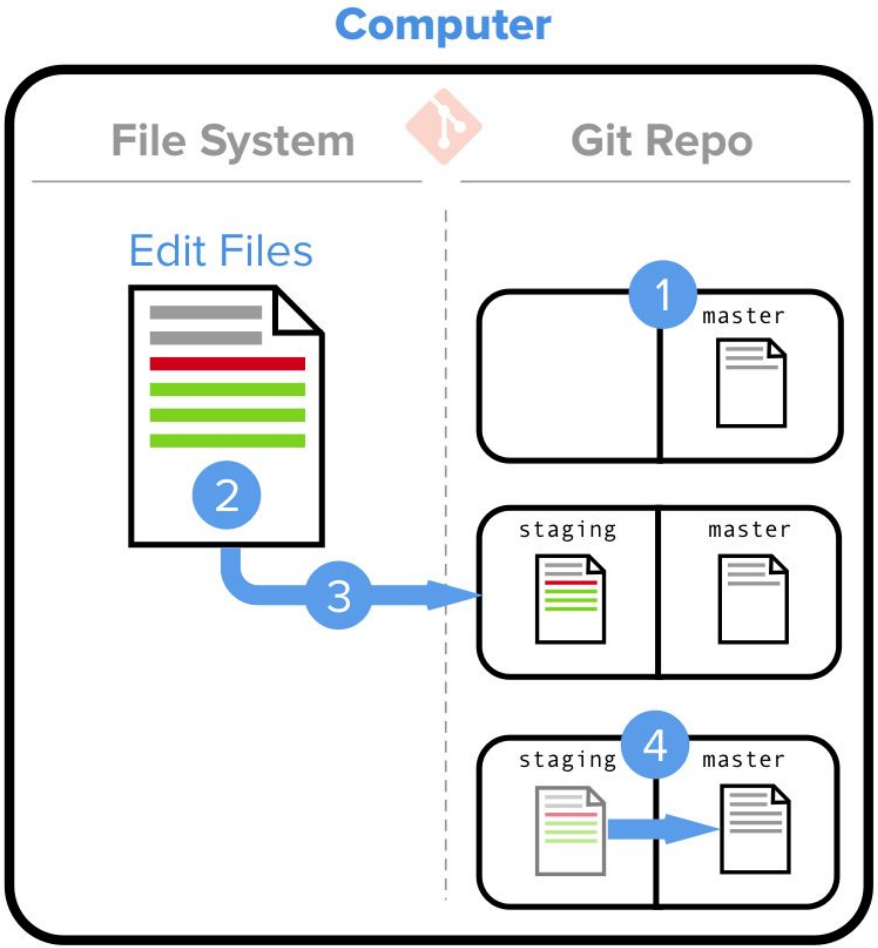


```
git push origin master
```

the remote name (“origin” by default)

the branch name

Git Workflow



1 `git init`

(or)

`git clone <repo-url>`

2 Edit Files Locally

3 `git add <files> <folders>`

4 `git commit -m "message"`

WHAT IS A BRANCH, CLONE, AND FORK?

BRANCH

Used for feature branching, avoiding merge conflicts

CLONE

Copies online repo to your local directory and make changes directly (with permission)

FORK

A copy of the same repository, but is often linked to collaborate on the original one via only pull requests

CLONE vs. FORK

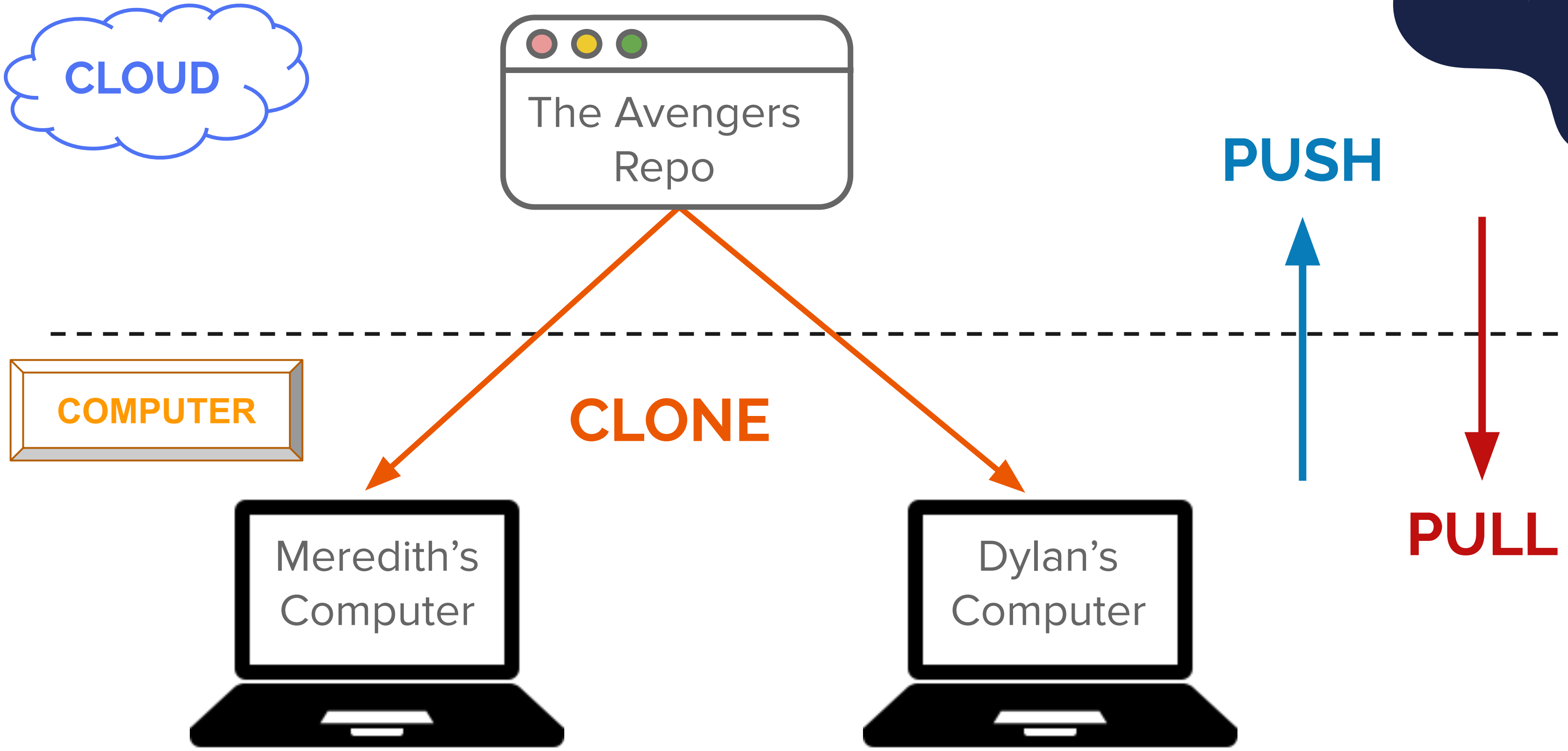
A **clone** is a *local copy* (on your computer) of a remote repository (on GitHub). Use a **clone** when:

- You need to copy **your own** repository from GitHub to your computer.
- You want to contribute to a **shared** repository where you are a collaborator (the owner has given you access to edit its contents).

A **fork** is a *new repository* that is a copy of another repo on GitHub. Both repos exist on GitHub (not your computer). Use a **fork** when:

- You want to contribute to a **shared** repository on GitHub where you are ***not*** a collaborator (such as an open-source repo, like React).

CLONE vs. FORK



CLOUD

The Avengers Repo

PUSH

COMPUTER

CLONE

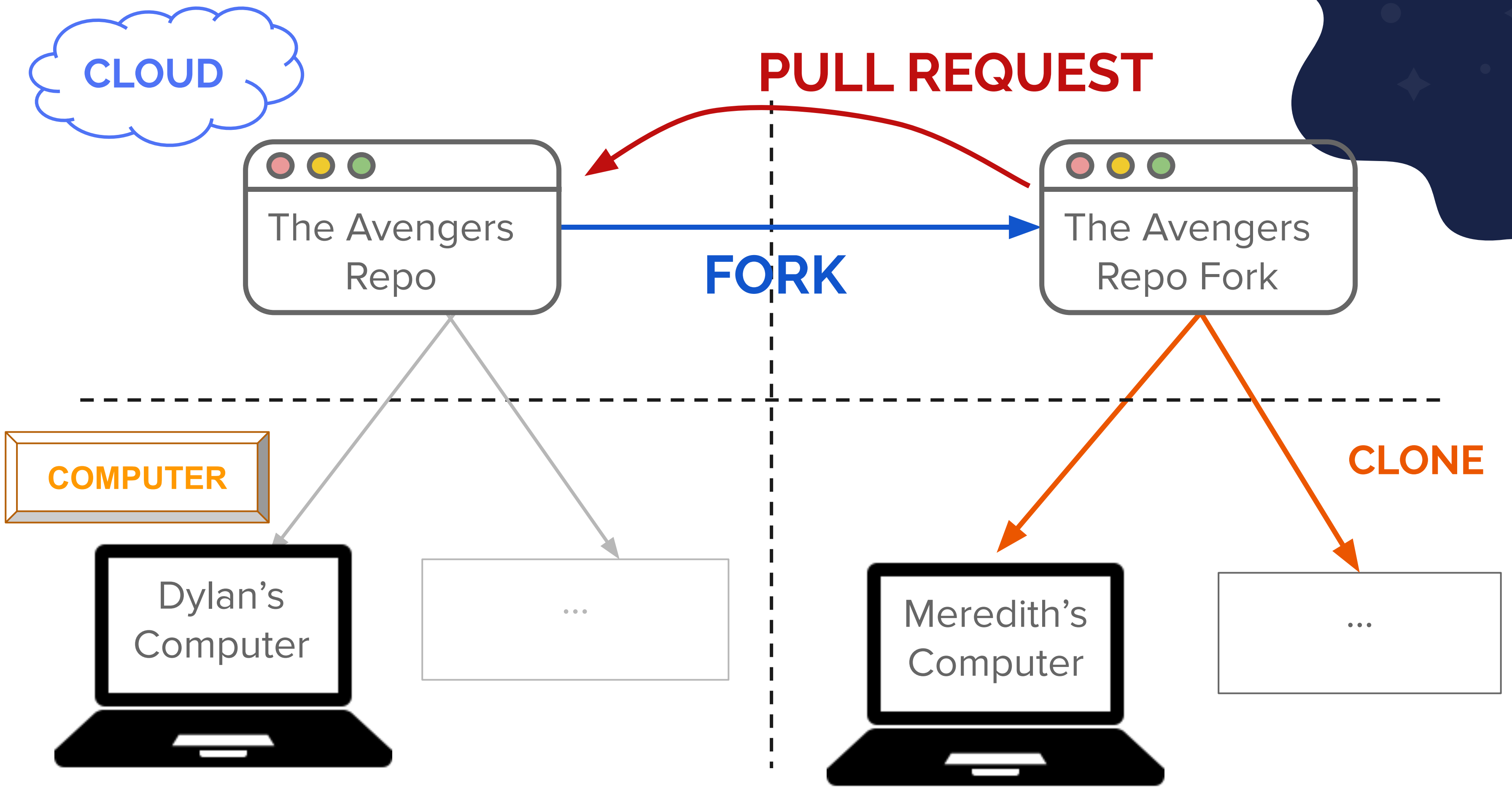
PULL

Meredith's Computer

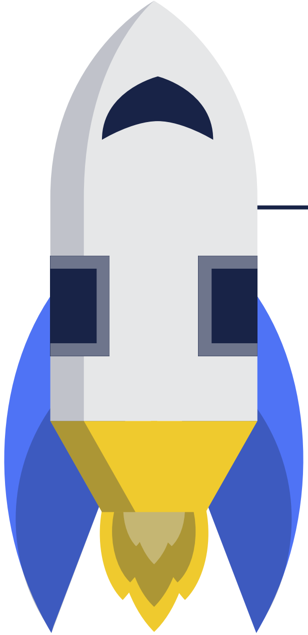
Dylan's Computer

What if I’m not a collaborator?

This is where forking comes in!


CLOUD
PULL REQUEST
The Avengers Repo
FORK
The Avengers Repo Fork
COMPUTER
CLONE
Dylan's Computer
...
Meredith's Computer
...

GIT BRANCHES



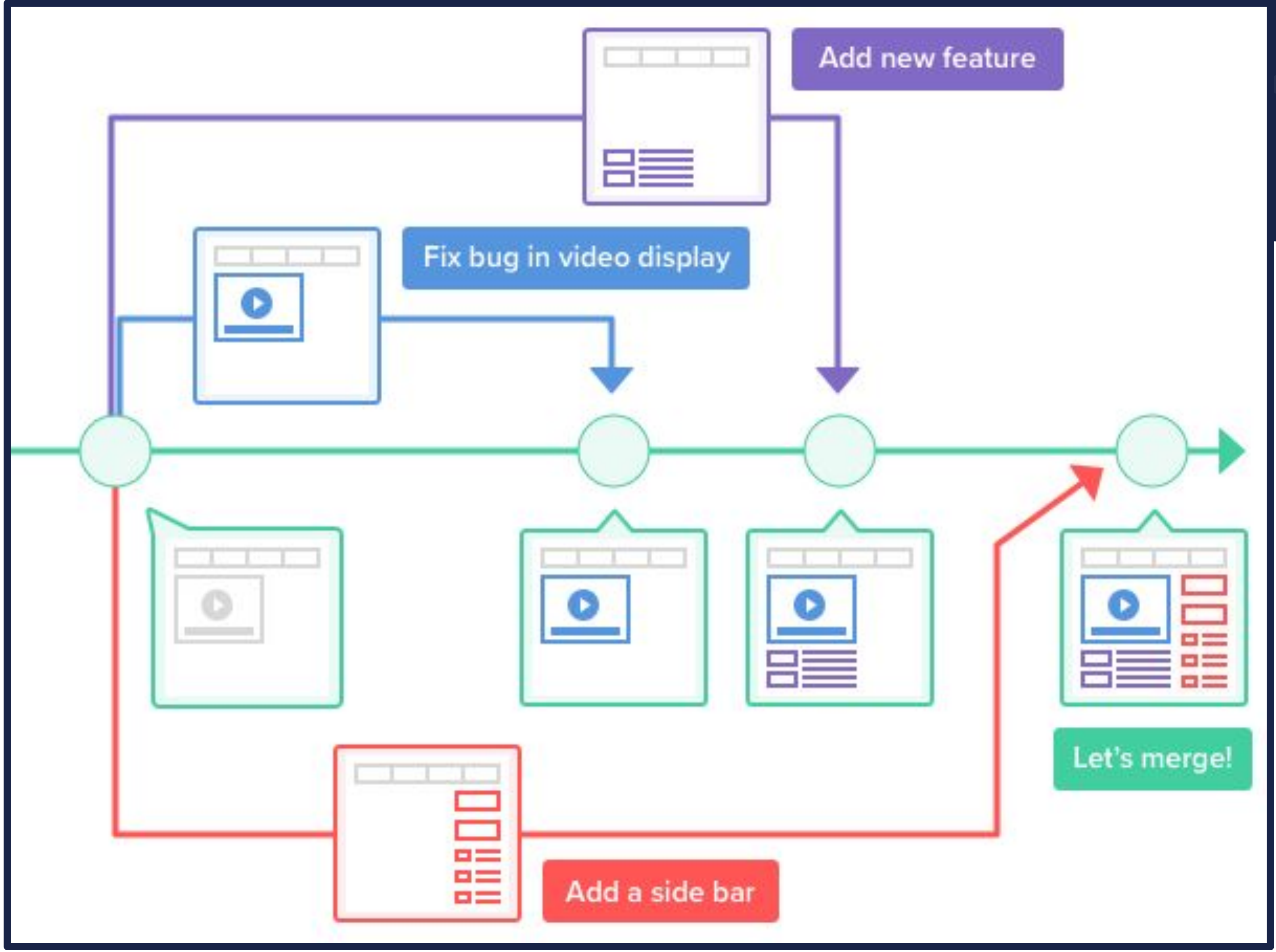
Definition

a ***separate*** *set of code changes* that build off of a previous version of the project

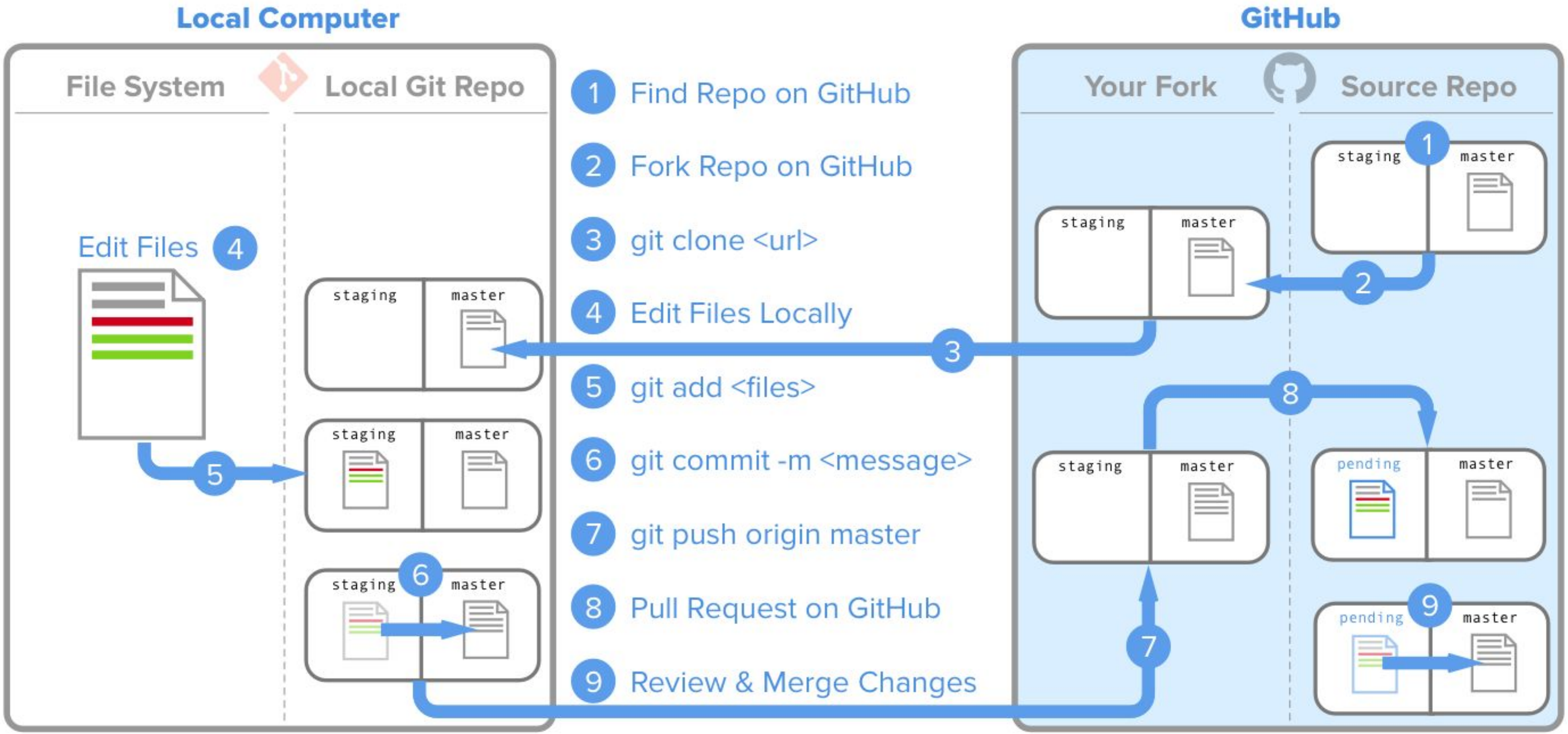
What it allows for your project...

Isolates development work without affecting other branches in the repository

Work on multiple features in parallel without disturbing the other


Add new feature
Fix bug in video display
Let's merge!
Add a side bar

MARVEL
SUPER HERO
SQUAD



1 Find Repo on GitHub

2 Fork Repo on GitHub

3 git clone <url>

4 Edit Files Locally

5 git add <files>

6 git commit -m <message>

7 git push origin master

8 Pull Request on GitHub

9 Review & Merge Changes

Writing Good Commit Messages

The git commit subject line should always be able to complete the following sentence:

`If applied, this commit will` *`<your subject line here>`*`.`

Example: **Add** Multi-Factor Authentication to login

Guidelines to Commit Messages

→ Use **imperative** mood in the subject

→ Wrap lines at ~**70** characters

→ Start with **capital letters**

Informative Commit Messages

→ **Brief** description of change in the first line

→ **Describe why** the change was made

→ **Never assume** the reviewer understands the original problem

→ **Describe** any limitations of the current code

Going Forward

Organizers and judges will require all teams to have a Gitlab repository!

Gitlab and Github are very similar.. But if you have any more questions, the help desk will gladly help you out further.

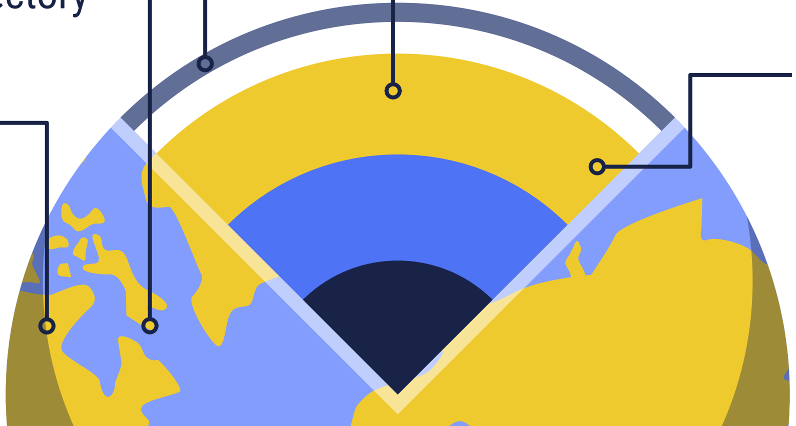
Resources

Easy Guide to Solving Merge Conflicts

Branching Guide

Forking and Clone Guide

FINAL PROCESS



Step 0

Initiate Git repository on your workstation

Step 1

Make, stage, and commit changes in git directory

Step 2

Publish it online using “git push” for collaboration

Step 3

Collaborate via forks or cloning the repository

Step 4-5

Create, review, and merge a pull request

THANK YOU!

Any questions?

Discord: harryy#4796
Email: hzhu20@georgefox.edu

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon** and infographics & images by **Freepik**