

# GitHub Introduction

Anzony Quispe<sup>1</sup> & Alexander Quispe<sup>2</sup>

<sup>1</sup>Princeton University

<sup>2</sup>The World Bank

January 4, 2024

## Citation

These notes are based on the Lecture Notes of DIME Analytics - World Bank.

## Before Starting the class

- Do you have a GitHub account? If not, [click here](#) to sign up.
- Have you submitted your GitHub username to the host alexander.quispe@pucp.edu.pe?
- Have you installed GitHub Desktop? If not, [click here](#) and download it.
- Have you logged into GitHub Desktop at least once? If not, open GitHub Desktop and sign in with your GitHub account.
- Have you been invited to the main repository? And have you accepted the invitation?

# Version Control Issues



# Typical Issues

- Several copies of the same document.
- Problems in tracking changes.
- Accidental mistakes.
- Backups problems.
- Messy documents.
- Team working problems:
  - For Dropbox users, "Copy Conflict" may sounds familiar.
  - Who did change the code?
- No version history.

# Bad Alternative Solutions

- Name files using dates in filenames.
- Name files with version number.
- Email Threading.

# Git - The Best Solution

Git is a version control system developed by Linus Torvalds, the famous creator of the Linux operating system kernel. We can track changes in our files in a highly structured way. Benefits:

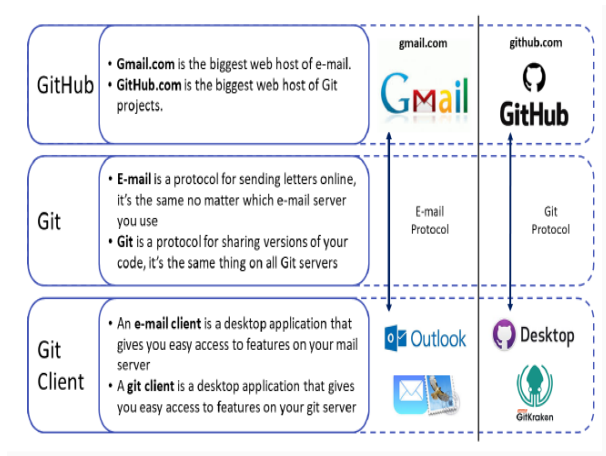
- A complete long-term change history of every file.
- In this workflow, we can work in teams. Anyone can edit; anyone can access previous versions and the most recent versions.
- It makes it easy to get excellent documentation of your project versions.
- Being able to trace each change made and the responsibility for that change.

GitHub is a hosting service. It provides a "home" for your Git-based projects on the internet.

- Showcase your work. Your projects are exposed on the internet and are easily shareable.
- You can easily track the development of any project.
- Everything is in the hosting service. Everything is going to be alright.



# Git/GitHub



# Key Concepts

- **Repository** : Set of files. It exists as a remote repository.
- **Clone** : It is a local copy of all the repository on our computer at that point in time.
- **Commit** : Snapshot of your repository. Git does not store data as a series of changesets or differences but instead as a series of **snapshots**.
- **Branch** : It is simply a lightweight movable pointer to one of these snapshots or commits. A independent line of development.
- **Pull Request** : It is an event when a contributor asks a maintainer of a repository to review code they want to merge into a project.

# Repository

Your project folder is called **repository** in GitHub.

[Overview](#) [Repositories 16](#) [Projects](#) [Packages](#)

Type ▾


Language ▾

Sort ▾


**ECO224** Public

☆ Star

Repositorio del curso PUCP Inferencia Causal y Machine Learning


 Jupyter Notebook

☆ 3

 10

1 issue needs help


Updated 17 hours ago



**ml\_book** Public

☆ Star

Updated 5 days ago

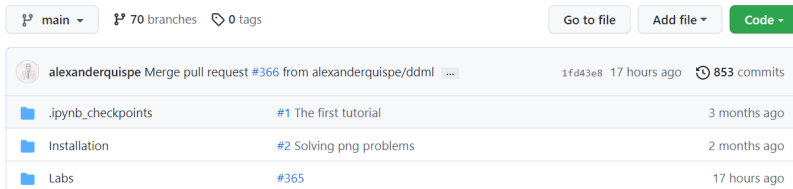


# Clone

It is not the same as downloading. When we clone a repository, we copy the complete history of that repository. Your computer uses Git software to download the version history. Additionally, we can make updates in the cloned repository that can be merged with the remote repository. When we download a repository, we do not copy the project history.

# How to clone?

- Go to the **main** page
- Click on green button **Code** or **download button**.



The screenshot shows the GitHub interface for the repository 'alexanderquispe'. At the top, there's a navigation bar with 'main' selected, '70 branches', and '0 tags'. To the right are buttons for 'Go to file', 'Add file', and a green 'Code' button. Below this, a merge pull request summary is shown: 'alexanderquispe Merge pull request #366 from alexanderquispe/ddml' with commit hash '1fd43e8' and '17 hours ago', and '853 commits'. A table of repository contents follows:

File/Folder	Commit/Link	Time
.ipynb_checkpoints	<a href="#">#1 The first tutorial</a>	3 months ago
Installation	<a href="#">#2 Solving png problems</a>	2 months ago
Labs	<a href="#">#365</a>	17 hours ago

# Commit

It is a snapshot of your repository. Git does not store data as a series of changesets or differences but instead as a series of **snapshots**. It has the metadata: author, date, message, parent commit, tree object hash, parent commit hash, and message. It indicates a significant modification to our version control system. Finally, a commit object references a snapshot.

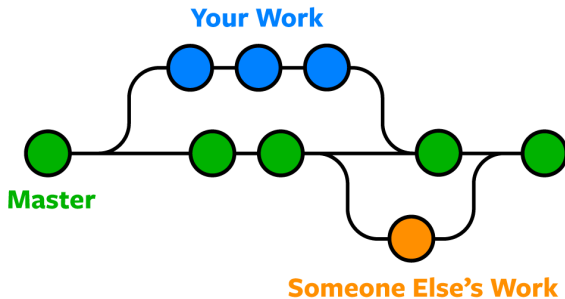
Each commit has an edit date and trace of who made that significant change.

Before we make a commit, we need to learn what is a branch.

We need to practice to learn. We are going to create a new file and store it in a specific folder.

# Branch

Using Branches is Git's "Killer" tool. This feature of Git is what makes it so powerful as a collaboration and version control tool. It allows you to refer to any snapshot of your project.



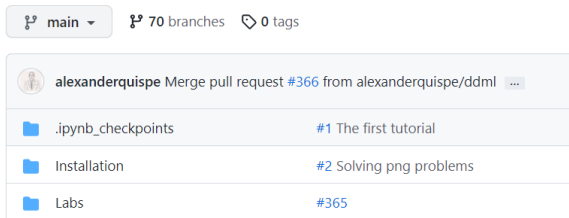
## Key questions about branches

- What happens to the contents of the folder on your computer when you check another branch in GitHub Desktop?
- What is the version that is cloned on your computer? Actually, all are cloned, but only one is shown, that branch you work on.



# Create a Branch

- Go to our repository in GitHub.



- Write your name in that field and click on Create branch: your\_name. Make sure it says "from master".
- Now another branch should appear with your\_name
- Go to this link to see your branch.

# Pull Request

PULL REQUEST = Request that your modifications be transferred to the **master branch**.

Usually, there is always a person who is called the **REPO MAINTAINER**, who has access to the **master branch**. Therefore, the only way to contribute to the **master branch** is through a **pull request**.

## How to do a Pull Request

- Go to the **Pull Request** section of our repository.
- Click on **New Pull Request**
- Select your branch
- Check if the modifications you are requesting for the master branch. **This step is important because in case of any conflict, you will not be able to unite your changes.** If everything is ok, then click on **Create Pull Request**.
- Finally click on **Create pull request** again.

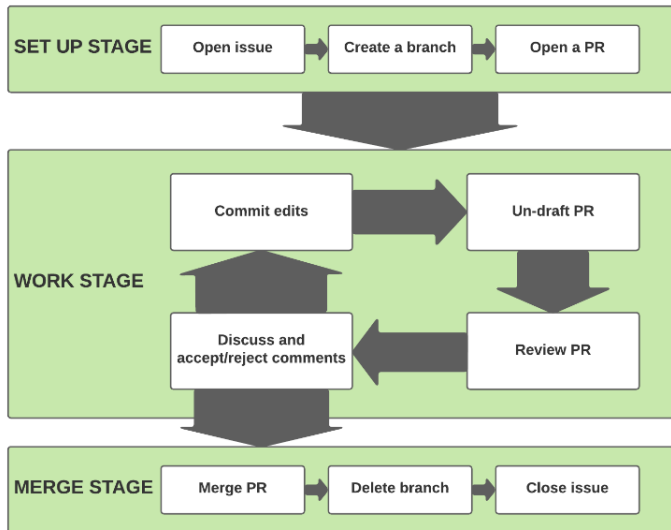
## Branch-PR-merge Cycle

This the best practice framework for implementing each task in a Pull Request.

It has three stages. In this course, all the students must complete a full branch-PR-merge cycle for all the tasks. It will be part of the evaluation.

- Set up Stage
- Work Stage
- Merge Stage

# Branch-PR-merge Cycle



# Set up Stage

This stage has three main sections.

- Open issue
- Create a branch
- Open a PR.



# Issues

We create a space in our repository to work on. This space is named as **Issue**.

- The issues allow us to document every significant change in our repository.
- We can assign and classify tasks.
- Team members can provide insights on how to solve a specific task.

<input type="checkbox"/>	Days to withdrawal <b>priority: low</b>	2
R297 opened on Dec 15, 2020 by kiryborry		
<input type="checkbox"/>	Code review	3
R299 opened on Dec 15, 2020 by kiryborry		
<input type="checkbox"/>	Decomposition extensive / intensive margin	1
R294 opened on Dec 15, 2020 by kiryborry		
<input type="checkbox"/>	Urban vs. rural <b>priority: high</b>	1
R298 opened on Dec 15, 2020 by kiryborry		
<input type="checkbox"/>	Combination variable - failure to withdraw and/or zero withdrawal recorded in tranche <b>priority: high</b>	6
R291 opened on Dec 14, 2020 by kiryborry		
<input type="checkbox"/>	Back of envelope estimate of cost of biometric ATMs (and ATM coverage overall)	
R296 opened on Dec 14, 2020 by kiryborry		
<input type="checkbox"/>	Gross vs. net interpretation	
R283 opened on Dec 11, 2020 by kiryborry		
<input type="checkbox"/>	Decomposing first stage <b>priority: medium</b>	5
R282 opened on Dec 11, 2020 by kiryborry		
<input type="checkbox"/>	Geocode all OPM HHs at the village level <b>priority: medium</b>	3
R278 opened on Nov 11, 2020 by amersjkal94		
<input type="checkbox"/>	Payment point proximity	0 of 2
R277 opened on Nov 4, 2020 by kiryborry		
<input type="checkbox"/>	Compare zero withdrawals in overall MIS vs. in matched sample w/OPM	
R276 opened on Nov 4, 2020 by kiryborry		
<input type="checkbox"/>	Shift into & out of zero withdrawals (breaking down extensive margin effect)	
R275 opened on Oct 26, 2020 by kiryborry		
<input type="checkbox"/>	Specifications with payment point data <b>OPM &amp; MIS priority: high</b>	2
R274 opened on Oct 21, 2020 by kiryborry		

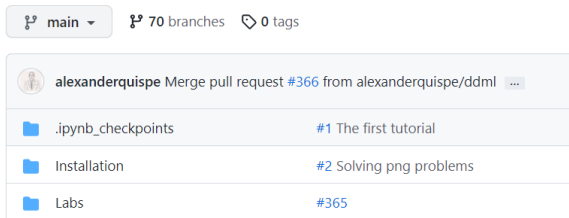
## How to create an Issue?

- Click on issue tab **New Issue**.
- Add descriptions and assign it properly.
- Add labels to facilitate future searches.



# Create a Branch

- Go to our repository in GitHub.



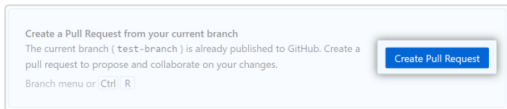
- Write your name in that field and click on Create branch: your\_name. Make sure it says "from master".
- Now another branch should appear with your\_name.
- Go to this link to see your branch.

## Open a Pull Request

- It creates a space on GitHub where the progress of this task can be followed.
- We create a *draft PR* that cannot be merged while in draft status.

# How to create a Pull Request

- Switch to the branch that you want to create a pull request for.
- Go to *Pull Request* tab.



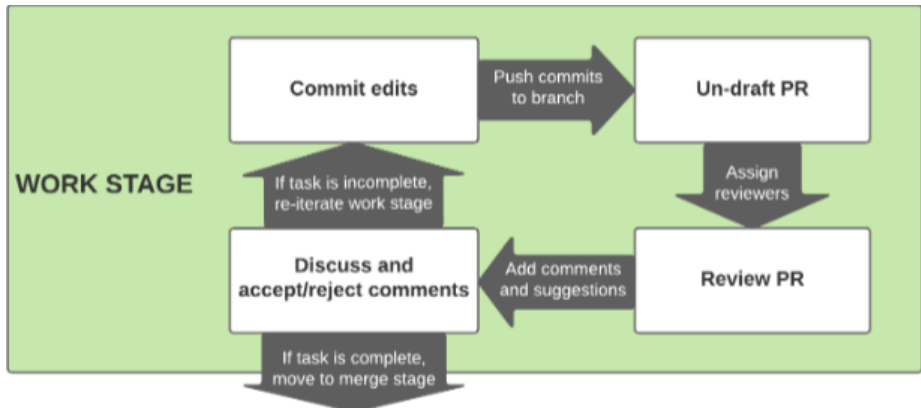
- Please, confirm that the branch in the base is the branch where you want to merge your changes.
- Name your **Pull Request**.
- We can label our **Pull Request**.

## Work Stage

Complete your tasks.

- Commit edits: Commit every significant change.
- The Pull Request will get every commit we made.
- Make commits until the task is completed.

# Work Stage



# Work Stage

- Un-draft Pull Request: Assign reviewers in case you work with a Repository Maintainer (RM).
- You can tag team members to be part of the discussion.

The screenshot displays a GitHub commit page for commit `5e6389e`. The commit message is: `@@ -27,38 +27,35 @@ La version que ustedes **unbran** al master branch no será la que envías cuando`. The commit includes several suggestions for improvement, such as "Dar un nombre al PR un nombre asociado al nombre del branch, pero un poco más descriptivo" and "Pueden agregar un label `**Work In Progress**`".

Below the commit message, there is a comment from `alexanderquispe` (Collaborator) stating: "Creo que deberías revisar nuevamente las líneas de código que has agregado." (I think you should review the lines of code you have added again).

At the bottom, there is a comment input area for `@antonioah23` with the text: "deberías revisar estas nuevas líneas que se han agregado." (you should review these new lines that have been added). The input area includes a "Write" tab, a "Preview" tab, and a "Comment on this commit" button.

## Merge Stage

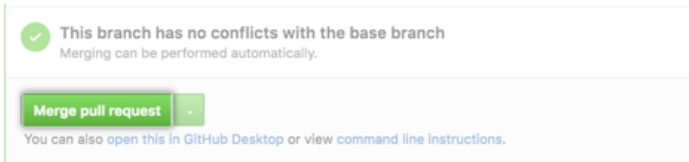
We merge the Pull Request only when we finished with the task.

- Your task must be properly documented.
- Future team members should be able to understand the state of the issue by reading your documentation.
- Delete the branch. We do not want to work on an outdated version of the repository. If you want to work in the same task in the future, you have to delete the branch and create a new one.

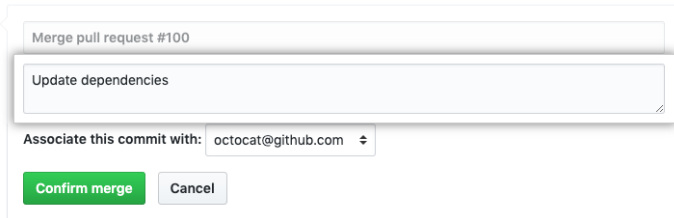


# Merge Stage

Merge your Pull Request.



Confirm your Merge.



Finally, close the issue when the task is done.