Lecture 4: Intro to Git & GitHub

Economía Laboral

Junghanss, Juan Cruz

Universidad del CEMA

2nd Semester 2024

Contents

Today's lecture content:

- Intro to Beamer presentations in LaTeX
- Intro to GitHub

Introduction to Beamer in LaTeX

LaTeX se puede usar para lo que necesitemos, ya sean artículos, presentaciones, CVs, etc. La idea es que hoy veamos un poco acerca de las presentaciones (clase "Beamer").

- Beamer es la clase de documento usada para presentaciones (tipo Powerpoint).
- Existen cientos de templates y estilos, no siempre tiene que ser el clásico beamer académico (como este).
- Se trabaja por frames (diapositivas) y se setean parámetros que aplican para todas.

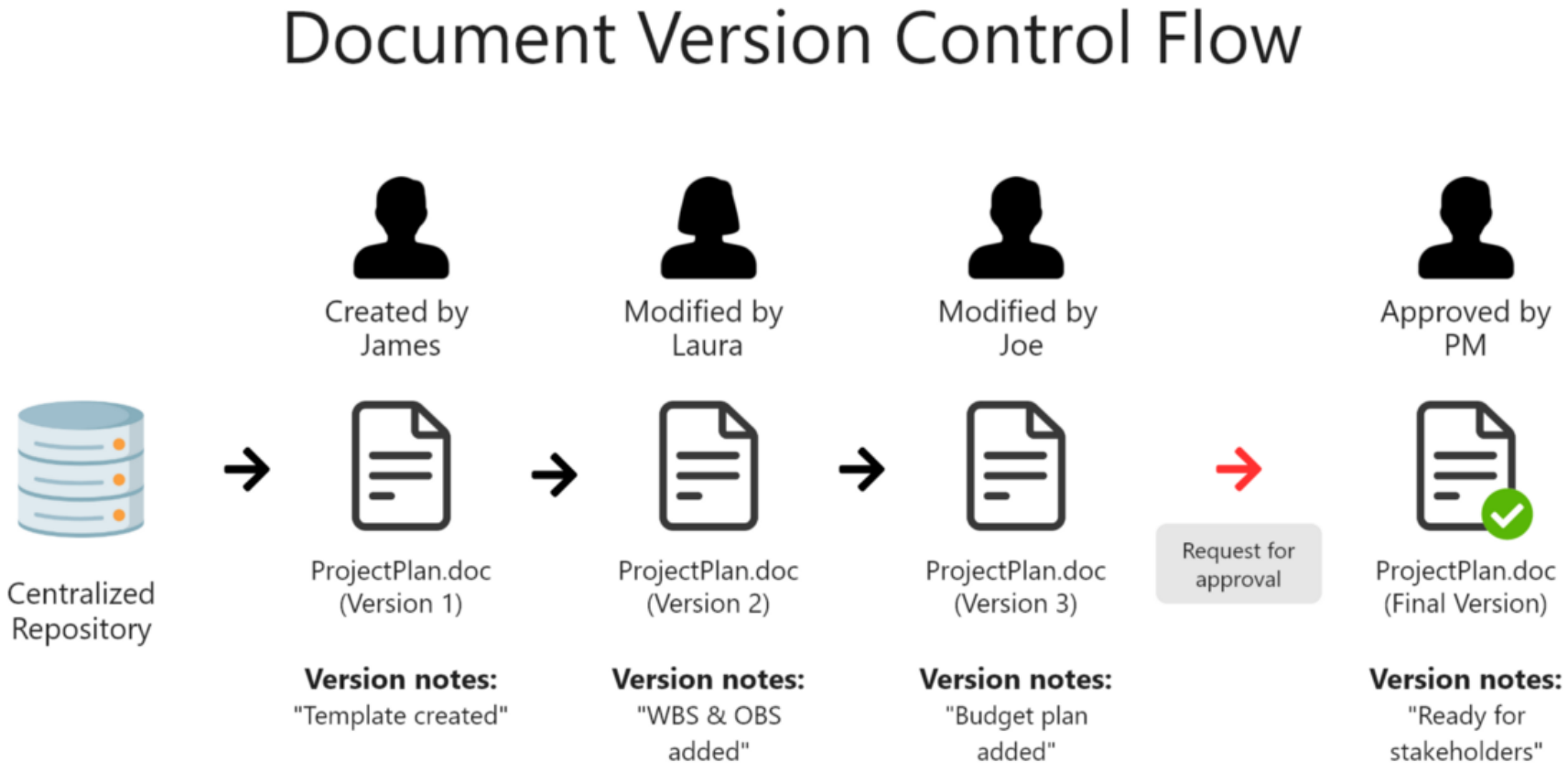
Veamos el template en la website.

Introduction to Git

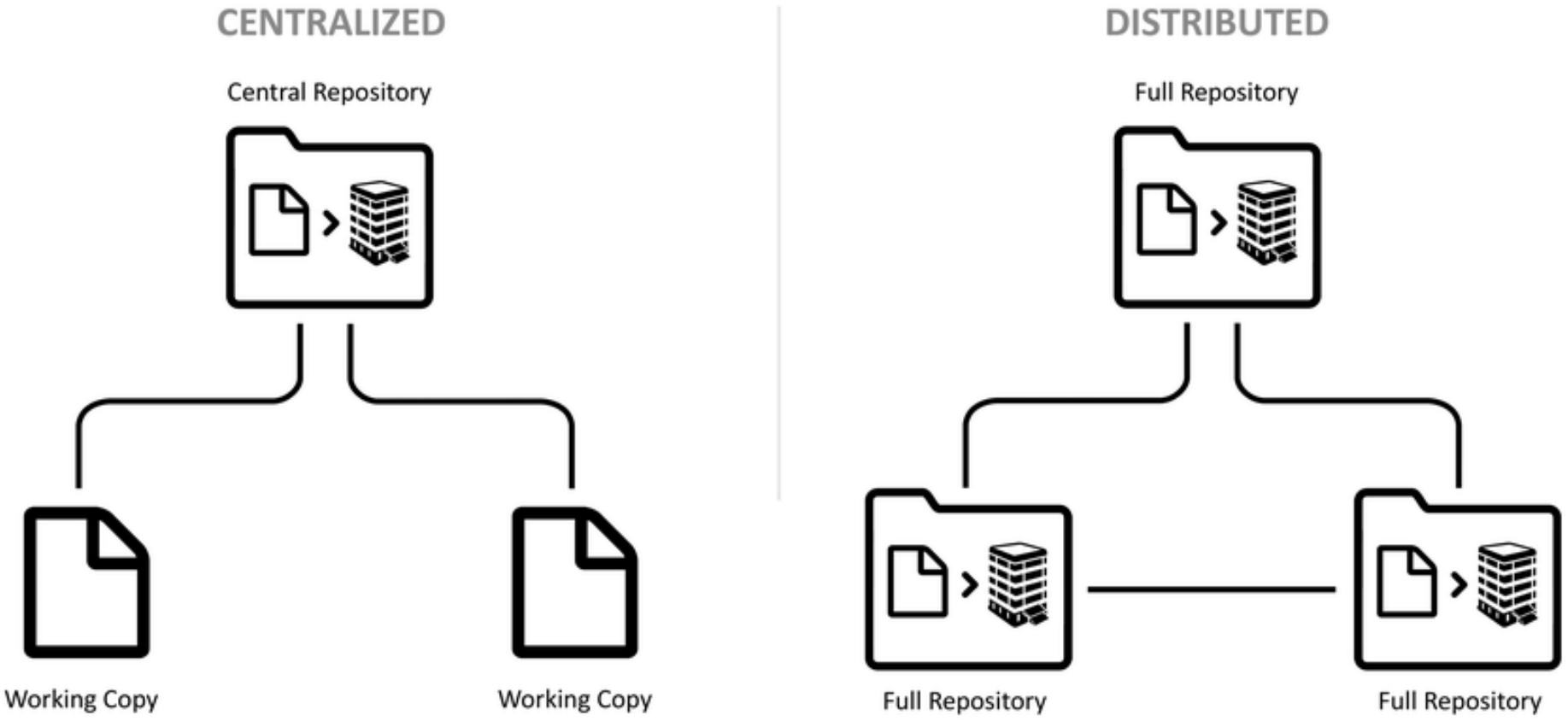
Although we want to use GitHub, we have to begin by learning about **Git**.

Git is a free and open source **distributed version control system** designed to handle everything.

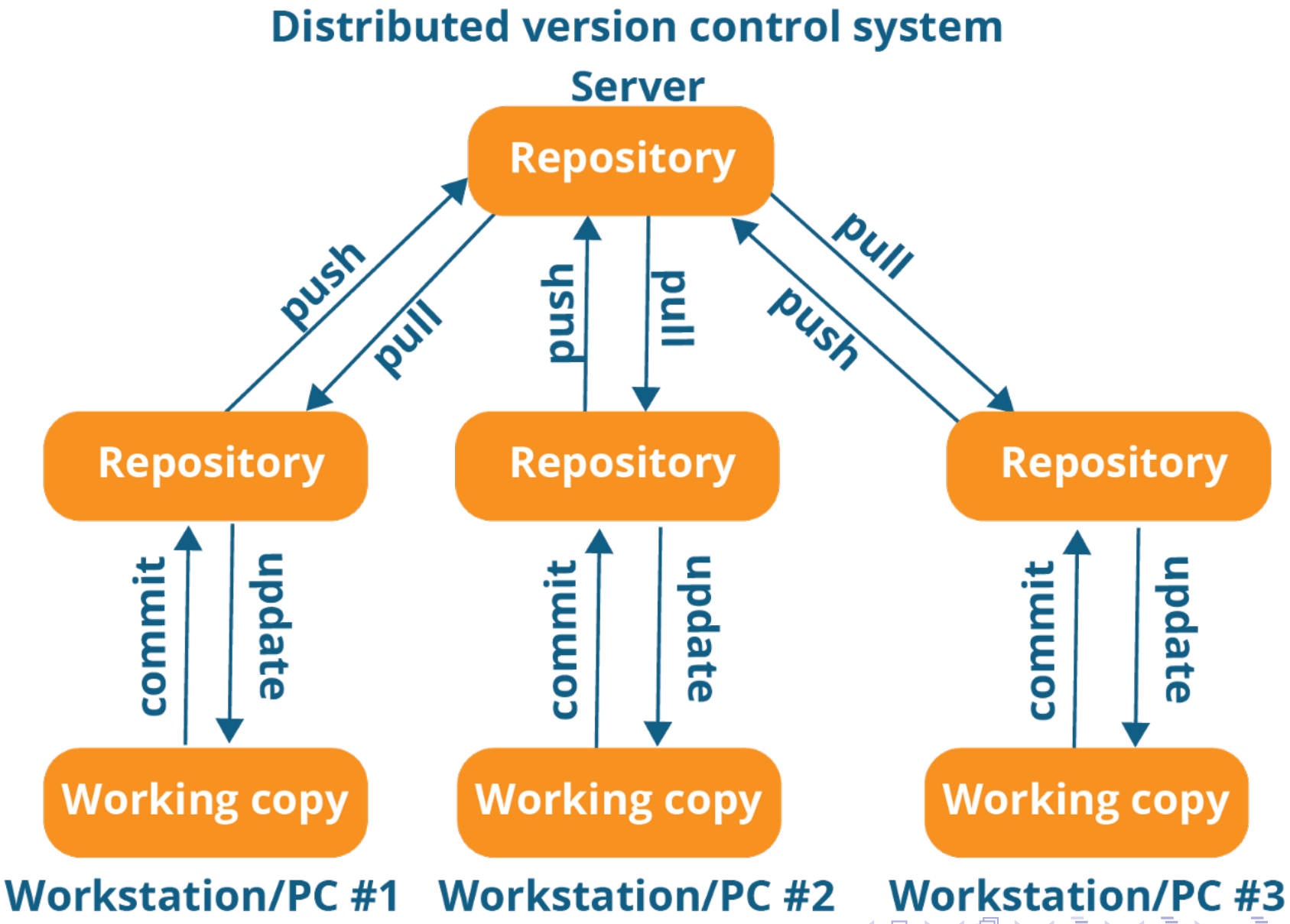
- **Version control** is a system that records changes to a file or set of files over time so that you can recall specific versions later.
- A **Distributed** version control system (in opposite to centralized) is a form of version control in which the complete codebase, including its full history, is mirrored on every developer's computer.


Document Version Control Flow
Created by James
Modified by Laura
Modified by Joe
Approved by PM
Centralized Repository
ProjectPlan.doc (Version 1)
ProjectPlan.doc (Version 2)
ProjectPlan.doc (Version 3)
Request for approval
ProjectPlan.doc (Final Version)
Version notes: "Template created"
Version notes: "WBS & OBS added"
Version notes: "Budget plan added"
Version notes: "Ready for stakeholders"

Introduction to Git



Introduction to Git

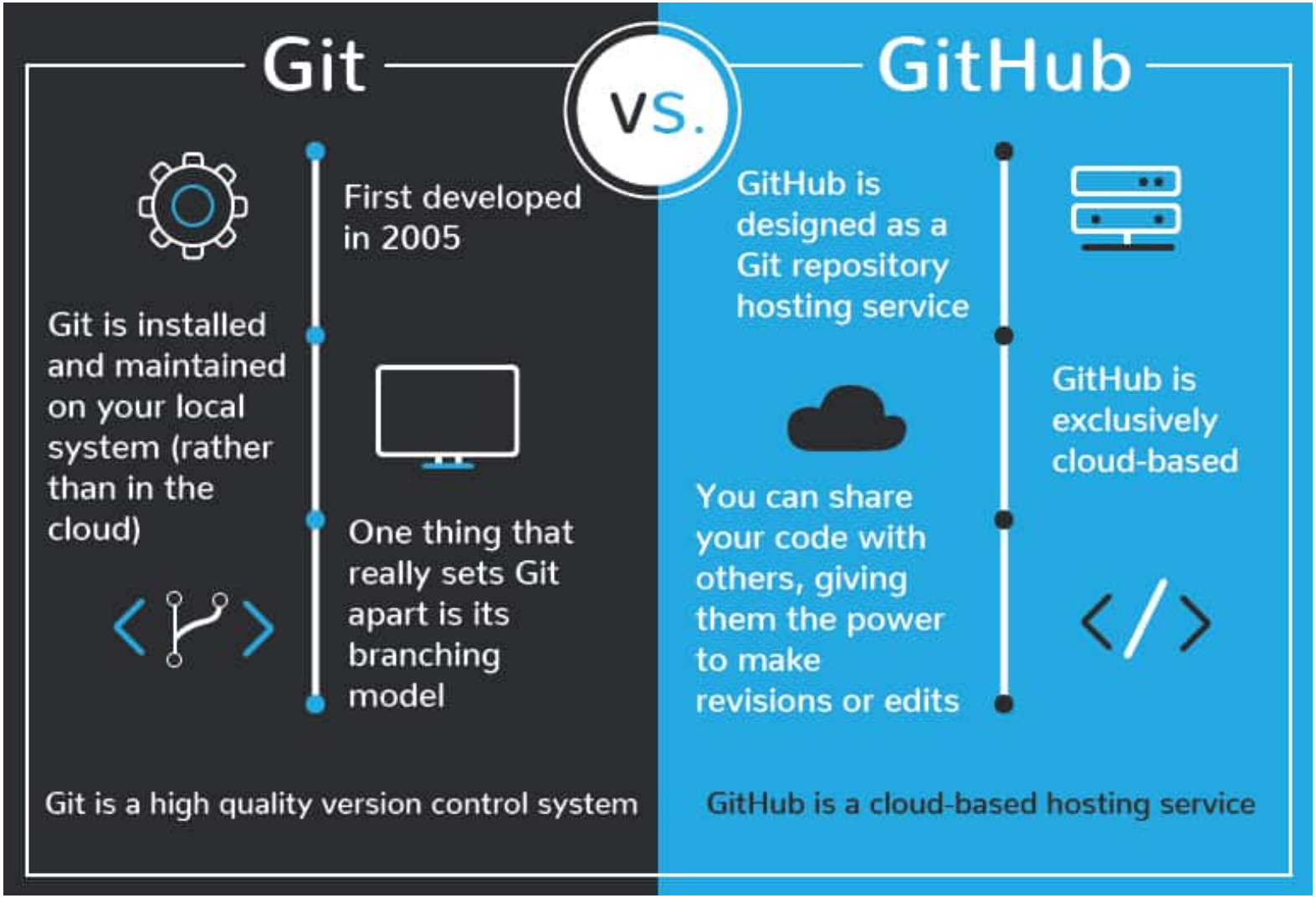


Introduction to GitHub

So, what is GitHub?

- **Website and cloud-based hosting service** for software development and version control using Git.
- GitHub's interface is user-friendly enough so even novice coders can take advantage of Git.
- Some people even use GitHub to manage other types of projects, like writing books.

Introduction to GitHub



Introduction to GitHub

There are for sure companies and developers that choose other cloud hosting service providers for Git:

- GitLab
- Bitbucket
- GitBucket
- AWS CodeCommit
- Google Cloud Source Repositories
- etc.

Introduction to GitHub - Advantages

Advantages of GitHub:

- One of the most popular
- User-friendly interface
- Benefits for students

El mejor consejo que les puedo dar: **creen una cuenta con su email de la universidad para ganar los beneficios de estudiante**

https://education.github.com/pack

Introduction to GitHub - Motivation

Motivation: ¿Para qué nos sirve GitHub si no somos programadores?

- Podemos trabajar colaborativamente sobre los proyectos que impliquen programar (en cualquier lenguaje y en cualquier contexto: universidad, trabajo, etc.)
- Podemos mostrar (a empresas/universidades) lo que hemos estudiado sobre programación.
- Podemos armar un portafolio de proyectos (econometría, data analytics, data science, etc.)
- Podemos construir y tener activa una website propia.

Introduction to GitHub - Motivation

Vayamos a GitHub y veamos qué clase de repositorios (relacionados a economía, finanzas y otras disciplinas) podemos encontrar navegando en la plataforma.

Introduction to GitHub - Hands On

En esta clase vamos a ver cómo se instala Git & GitHub en la computadora y cómo funcionan, pero por cuestiones logísticas nos vamos a tener que enfocar en **cómo funciona y se usa**.

Esto significa que van a tener que seguir los pasos para instalarlo en sus casas. Por ese motivo dejo los siguientes links de utilidad como soporte:

- Manual de Git y GitHub para usuarios de R
- YouTube: Cómo descargar Git en Windows (Importante)

De todas maneras, en nuestra website va a quedar resumida la introducción a Git y su instalación.

Register an account

Step 1: Register your account in https://github.com/

- Username advice: incorporate your actual name and pick an username you will be comfortable revealing to your future boss.
- Student account: you can register yourself with another email and later on add your student's email to get the benefits.

Install the GitHub client

Step 1.2: Install the GitHub desktop client (we'll use it a the end).

GitHub offers a free Git client, GitHub Desktop, for Windows and macOS.

GitHub Desktop is aimed at beginners who want the most useful features of Git front and center.

Install Git

Step 2: Install Git

A complete guide for Windows, Mac, etc. can be found here.
There are some ways to install it:

- Using the latest installer.
- Using the command prompt terminal (cmd)[1].

For Windows OS is more difficult (see tutorial here).
Open later the terminal and type "git version" to check if it was installed.

[1] Check the appendix if you don't know what this is.

Set your Git account

Step 3: Set your GitHub account in Git

Type the following in the shell:

Example

```
git config --global user.name 'nombre'
git config --global user.email 'correo'
git config --global --list
git config --global init.defaultBranch main
```

Recommendation: type just "git" and see all the available usage options.

Set your Git credentials - Access token for HTTPS

Step 4: Set your GitHub credentials to communicate with the remote server.

1. Go to github.com/settings/tokens and click "Generate token" (without expiration, etc.).
2. Copy the generated PAT to your clipboard.
3. Provide this PAT next time a Git operation asks for your password (see next step to create a Repo) or...
   Provide this PAT now in R using gitcreds package (usar el código de clase)

Creating our first Repo

Step 5: Create the first Repository.

Recall that a Git repository is the container for a project that is tracked by Git. Our first repository can be created in two ways:

1. We create a local folder in our computer $\Longrightarrow$ and later **initialize** a Repo (in that folder) with the shell (*we'll do this*).
2. We create the Repo directly in GitHub $\Longrightarrow$ and later **clone** it in our computer.

Initializing a Git Repository

Initializing a Repository: Go to the Shell and change you directory to the local folder where you want to have the Repo, then type the following:

To initialize it:

Example

```
git init
```

To check if it's ok:

Example

```
git status
```

Cloning a Git Repository

Clone a Remote Repository: Go to the Shell and change you directory to the local folder where you want to have the Repo, then type the following:

To clone the remote repo:

Example

```
git clone https://github.com/USERNAME/REPOSITORY.git
```

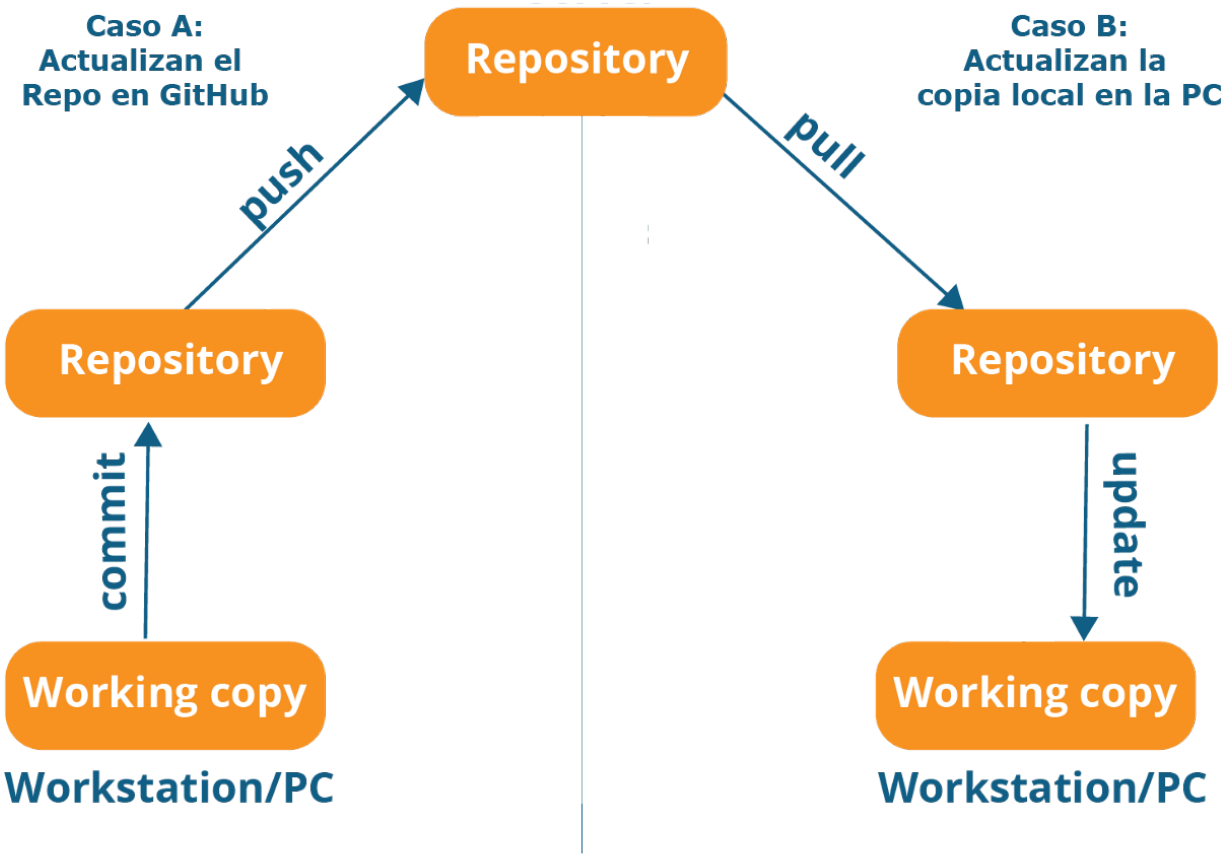
To check if it's ok:

Example

```
git status
```

Push/Pull the Git Repository

Step 6: Pulling and pushing the changes you want.
Recall that you can modify files in your local Repo folder and **push** those changes to your Repo in GitHub. And if someone made changes in the GitHub Repo you can update your local Repo folder when you **pull**.



Push/Pull the Git Repository

Step 6: Pulling and pushing the changes you want.

To **push** the changes:

Example

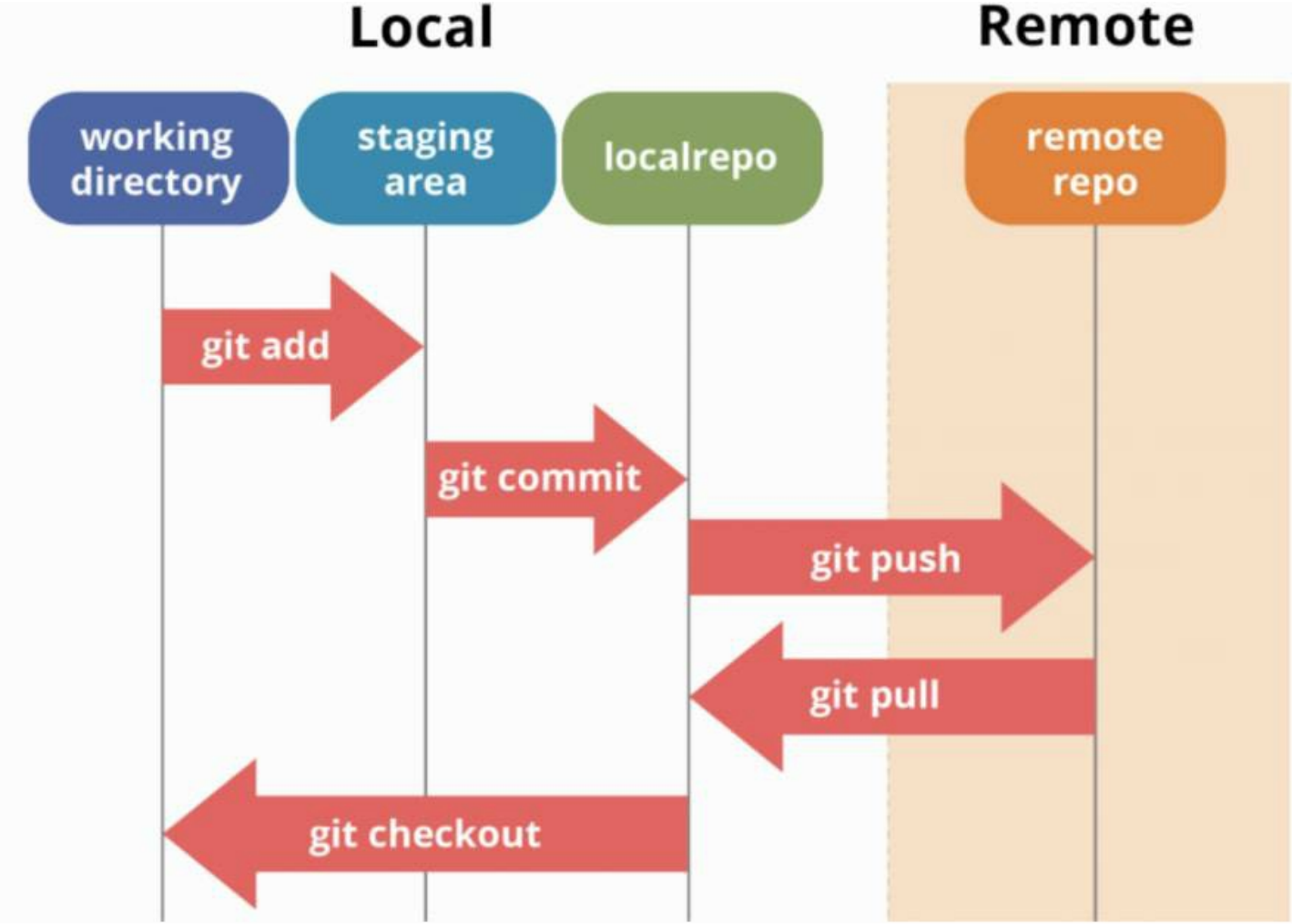
```
git add -A #Agregamos todos los archivos
git commit -m "aca va el mensaje"
git push origin master
```

To **pull** the changes:

Example

```
git pull origin master
```

Push/Pull the Git Repository



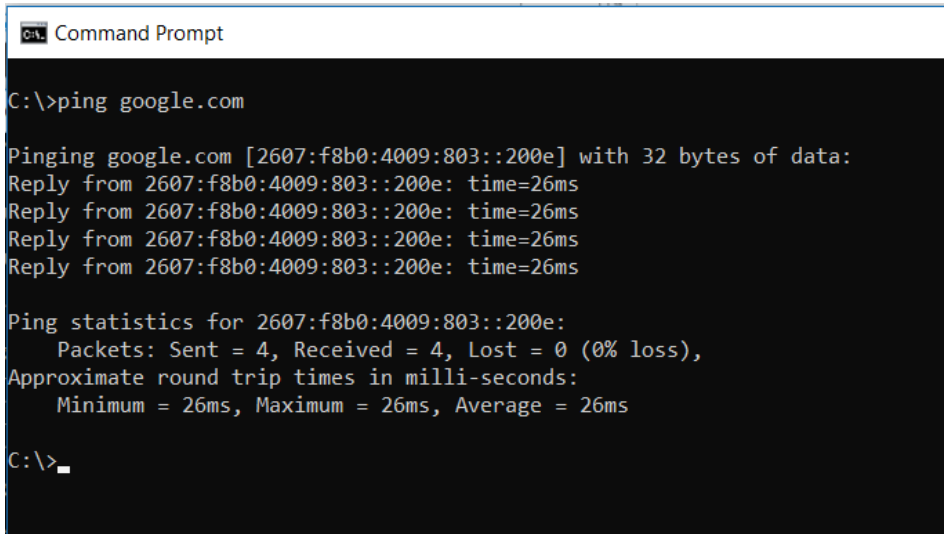
Push/Pull the Git Repository

Hint: the "*staging area*" is like a rough draft space, it's where you can git add the version of a file or multiple files that you want to save in your next commit (in other words it's the next version of your project).

What the Shell?

Appendix: working with the shell. Más info.

- Es un programa cuya función es correr otros programas.
- Expone los servicios del sistema operativo directamente al usuario.
- También llamada como consola, terminal, command line, cmd, etc.
- Se puede abrir desde Rstudio o el IDE con el que trabajen.



What the Shell?

Appendix: working with the shell.

Los comandos varían entre sistemas operativos, pero para Windows los más usados son:

- "dir" (directory) para listar todas las carpetas existentes.
- "cd" (change directory) entra en un directorio/carpeta.
- "cd .." regresa un directorio atrás.
- "mkdir" (make directory) para crear una carpeta.
- con "." preserva la ruta (path) del directorio actual. Ver ejemplo.

What the Shell?

Appendix: working with the shell.

Ejemplo de acceso a distintos directorios/carpetas:

```
C:\Users>cd ".\jcjun\Documents"    input
C:\Users\jcjun\Documents>_    output directory
```

```
C:\Users\jcjun\Documents>cd ..    input
C:\Users\jcjun>    output directory
```