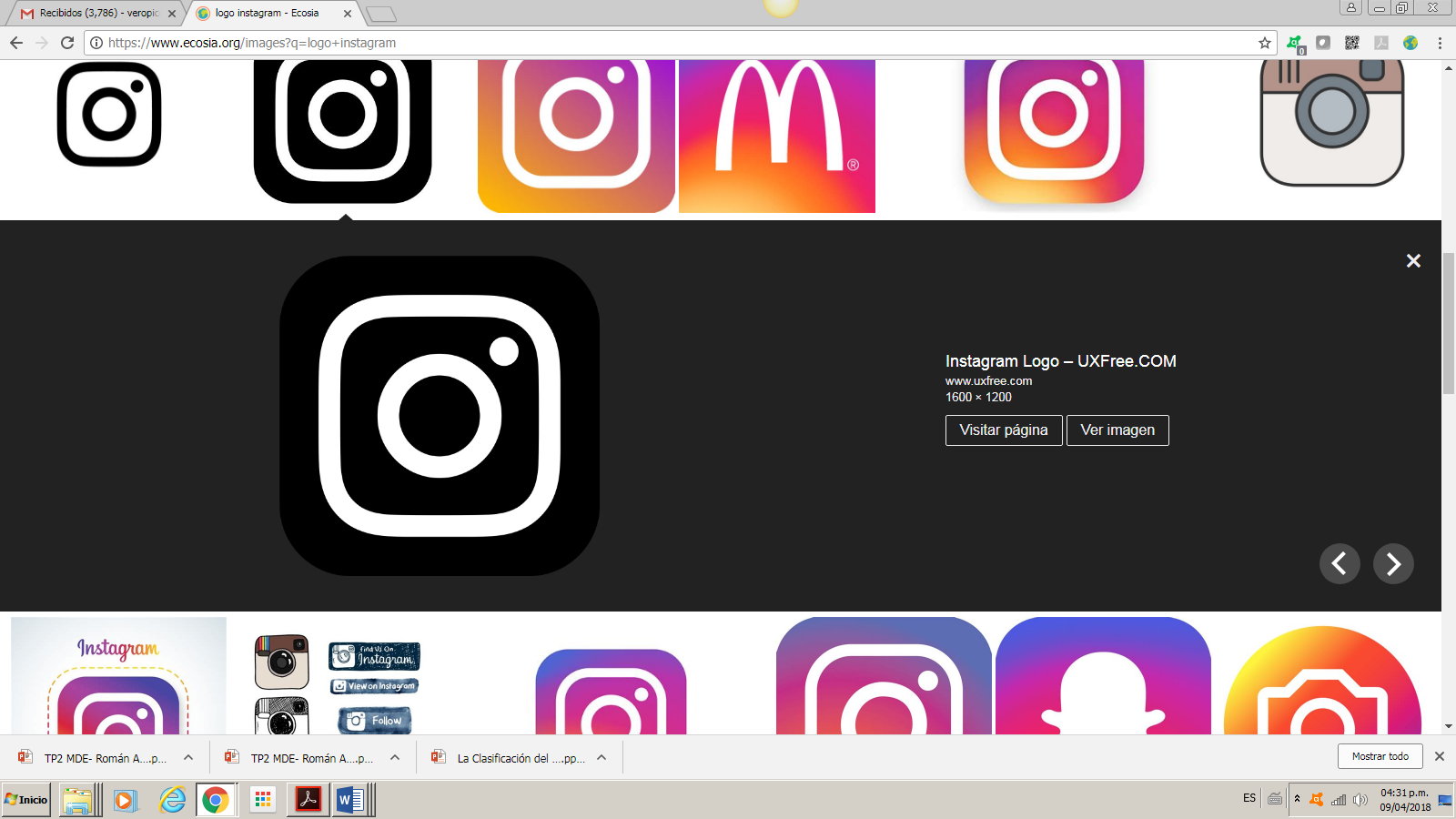
Carrera: **TÉCNICO SUPERIOR EN ANÁLISIS FUNCIONAL DE SISTEMAS INFORMÁTICOS**

Cátedra: **PRÁCTICAS PROFESIONALIZANTES II**

Curso: **3° Año**

Profesor: **Verónica M. Piccini** [veropiccini@gmail.com](mailto:veropiccini@gmail.com) [veropiccini@hotmail.com](mailto:veropiccini@hotmail.com) **** **@veropiccini**

**A16 GIT - GITHUB**

# **git: REPOSITORIO - CONTROL DE VERSIONES – DESARROLLO COLABORATIVO**

**GIT permite guardar ´puntos en el tiempo´o snapshot o capturas de pantalla del proyecto en distintos momentos.**

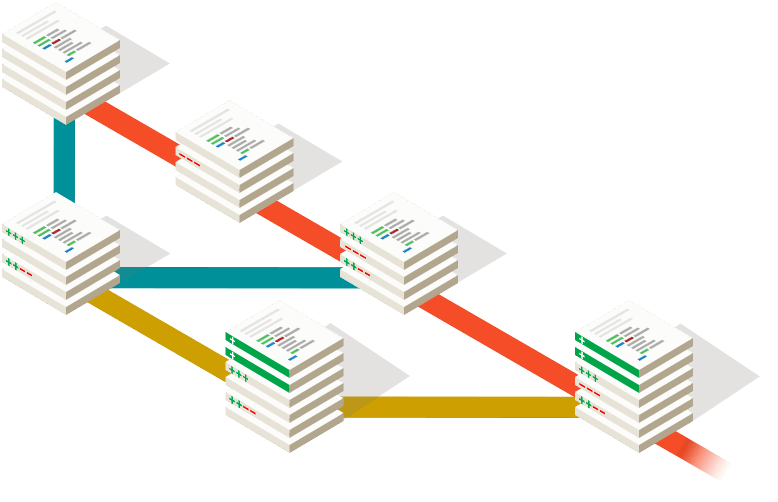


1. **git es un software de VCS (sistema de control de versiones local) que permite a los desarrolladores guardar instantáneas de sus proyectos a lo largo del tiempo. Generalmente es mejor para uso individual.**
2. **GitHub es una plataforma basada en la web que incorpora las características de control de versiones de git para que puedan ser utilizadas de forma colaborativa. También incluye características de gestión de proyectos y equipos, así como oportunidades para la creación de redes y la codificación social.**

**git** es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

El kernel Linux fue nombrado en honor a su creador, Linus Torvalds. Cuando escribió git, quiso dar con una palabra corta, fácil de recordar y de tres letras. La razón por la que eligió "git” podría haber sido dado por Linus Torvalds cuando escribió la primera versión: una combinación aleatoria de tres letras que es pronunciable, y que en realidad no es utilizada por ningún comando UNIX común. El hecho de que sea una mala pronunciación de "get" puede o no ser relevante.

Link Oficial: <https://git-scm.com/>



**EL CONTROL DE VERSIONES - git**

Imaginemos que nos despertamos un día y tenemos que comenzar con un proyecto nuevo. Este nuevo proyecto consiste en sólo un archivo fuente. Todos los cambios que le podamos hacer a ese archivo durante el día, lo más probable es que puedan ser deshechos. Al día siguiente continuamos con el desarrollo de nuestro proyecto y mejoramos nuestro archivo pero en el intento nuestro programa empieza a presentar errores. Necesitamos volver nuestro proyecto a una versión anterior, donde no había errores pero ahora estamos restringidos a solo poder deshacer los cambios hechos durante el día.

**git** entonces nos ayuda con este aspecto del desarrollo. Nos ofrece herramientas para poder gestionar cada una de las etapas y versiones por las que va transitando un proyecto de desarrollo. Por otro lado puede realizar, pero sin estar restringido un **Desarrollo Colaborativo** .

**EL DESARROLLO COLABORATIVO GitHub**

El **desarrollo colaborativo de software** es un modelo de desarrollo de software cuyas bases son la disponibilidad pública del código y la comunicación vía Internet. Este modelo se hizo popular a raíz de su uso para el desarrollo de Linux en 1991.

Teniendo como contexto a **git** , podríamos decir que el **Desarrollo Colaborativo** nos proporciona herramientas para poder desarrollar entre un gran número de individuos de una manera más fácil , menos propensa a errores y rápida de implementar. De esta manera siempre tenemos la opción de, por medio de algún cliente , publicar nuestro código junto con todas las etapas y versiones que nos llevó el proyecto para que otras personas puedan sumar y aportar nuevas ideas a nuestro repositorio.

# **GitHub PLATAFORMA DE DESARROLLO COLABORATIVO**

**GitHub** es la herramienta de versionado más utilizada del mercado, a través del uso de repositorios de datos y asegurando el código fuente, con un desarrollo colaborativo

Link Oficial: <https://github.com/>



**GitHub** es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. Anteriormente era conocida como Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena generalmente de forma pública. El 4 de junio de 2018, Microsoft compró GitHubpor la cantidad de 7500 millones de dólares, GitHub continúa siendo la plataforma más importante de colaboración para proyectos de código abierto.



La consola se utiliza de forma intensiva para la ejecución conjunta de casos y simulaciones, realiza versionados locales y remotos, ejecuta workflows y migraciones de un repositorio SVN a GIT, evalúa situaciones tales como el uso standard de un desarrollador, situaciones críticas de emergencia y riesgo, y atención urgente a problemas productivos, simula pérdidas de código, errores de manipulación, reestructuraciones masivas en el repositorio.

Algunas de sus funcionalidades son:

* Versionado de archivos desde cero
* Utilizar registro de cambios
* Trabajar con Ramas. Flujos de trabajo con Ramas
* Estrategia de unificación de trabajo
* Depuración de trabajo
* Manejo avanzado de GIT
* Repositorios centralizados y distribuidos
* Migraciones hacia repositorios GIT
* Detección temprana de problemas
* Reestructuraciones complejas en repositorio

# **VERSIONADO DEL SOFTWARE**

El **versionado de software es una opción de gran utilidad para mantener organizadas las labores** y establecer comunicación entre la organización y sus usuarios.

El proceso de versionado de software se refiere a una opción que permite identificar de forma**unívoca las nuevas features (características)  o funciones, actualizaciones y/o los fixes (correcciones)  corregidos.** De modo que los cambios incluidos en el versionado de un software deben contar con la característica de **aumentar las funcionalidades del sistema y solucionar sus errores**, sin modificar la aplicación o producto a nivel esencial.

Cabe destacar que **estas actualizaciones del sistema forman parte de la vida útil de un software,** por lo que se requiere mantener un control de estas nuevas versiones lanzadas.

## Características del versionado de software

1. **Permite la identificación de bugs (errores) o problemas de funcionamiento** a partir de una versión determinada.
2. Proporciona la hoja de ruta de la aplicación o programa, lo que **permite observar el avance y se actualice de acuerdo a lo estipulado**.
3. **Fija siempre las versiones de las librerías, así como de los componentes** que se usan. Esto ayudará a evitar los errores de utilización frente a la actualización de estos por parte del sistema o aplicación
4. **Incluye cambios en el sistema, en su estructura de database*,* en los protocolos**implementados y demás utilidades.

# **PRIMEROS PASOS CON GIT – DESCARGA E INSTALACIÓN DEL SOFTWARE**

# **PRIMER PASO- DESCARGA E INSTALACIÓN DEL SOFTWARE**

Link Oficial de descarga:

<https://git-scm.com/>

Manual y Ayuda de Comandos GIT

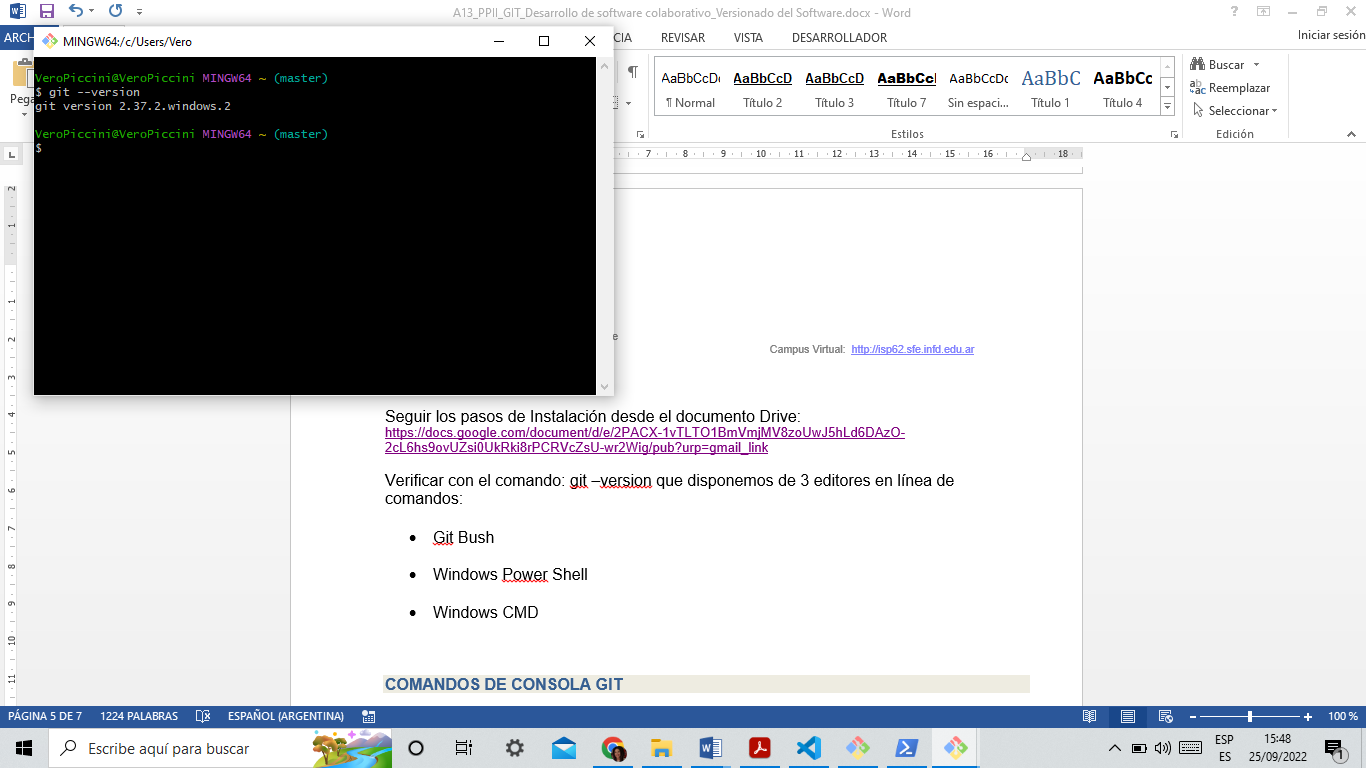
<https://git-scm.com/docs>

Seguir los pasos de Instalación desde el documento Drive:

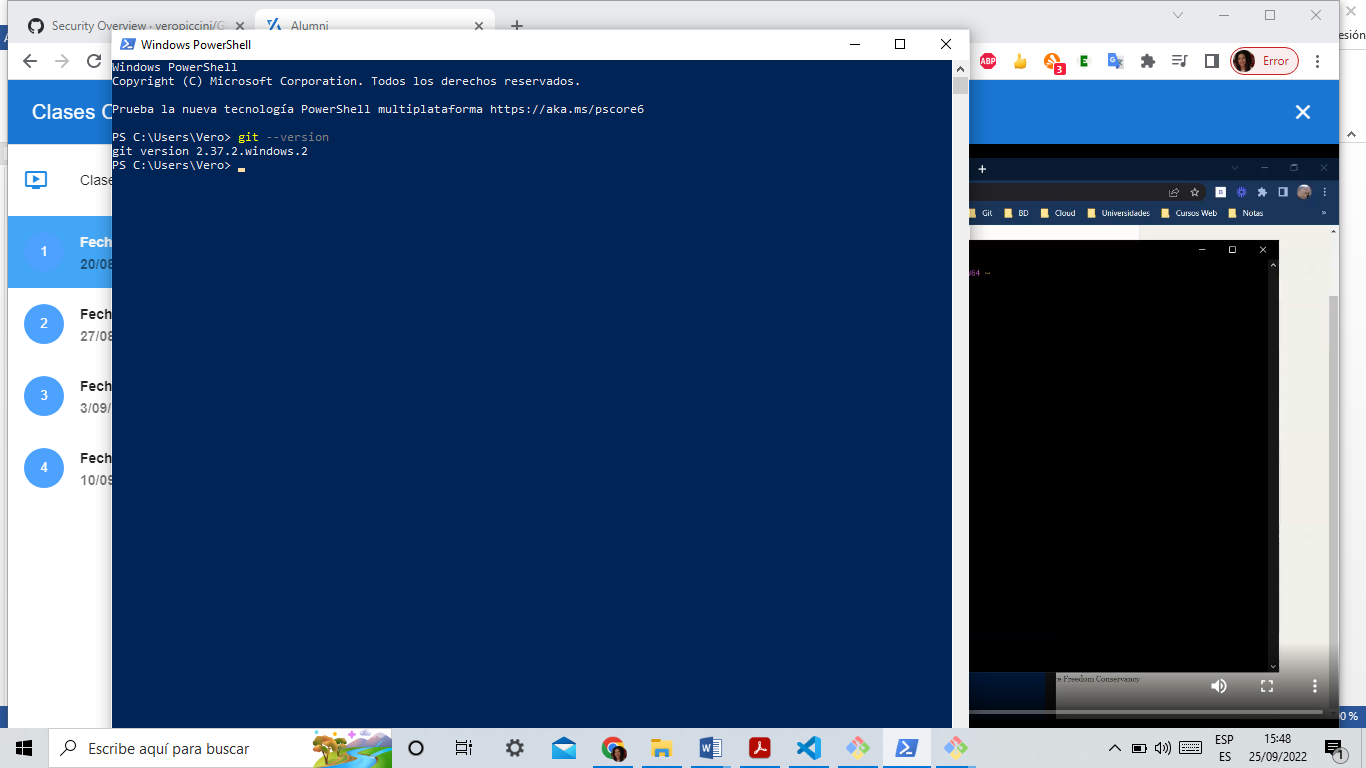
<https://docs.google.com/document/d/e/2PACX-1vTLTO1BmVmjMV8zoUwJ5hLd6DAzO-2cL6hs9ovUZsi0UkRki8rPCRVcZsU-wr2Wig/pub?urp=gmail_link>

Verificar con el comando: **git –version** que disponemos de 3 editores en línea de comandos:

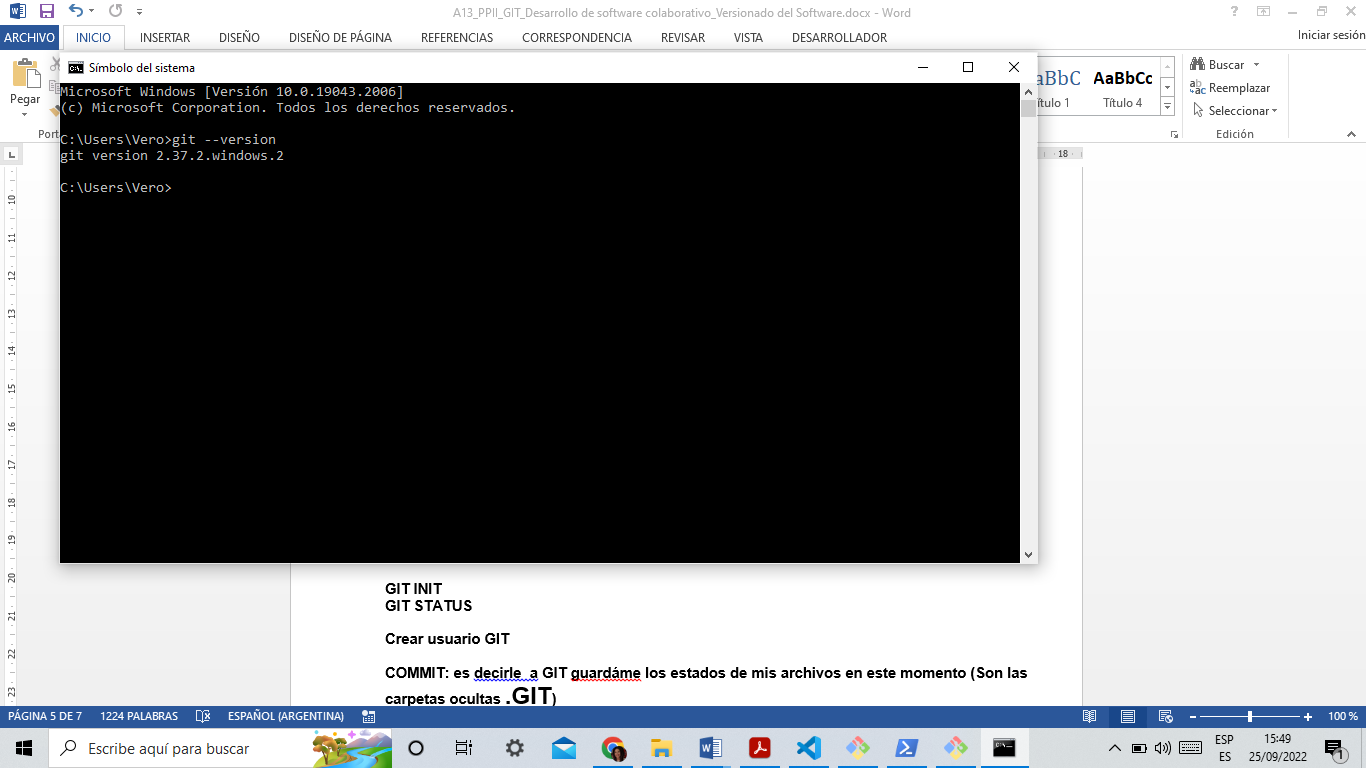
* Git Bush



* Windows Power Shell



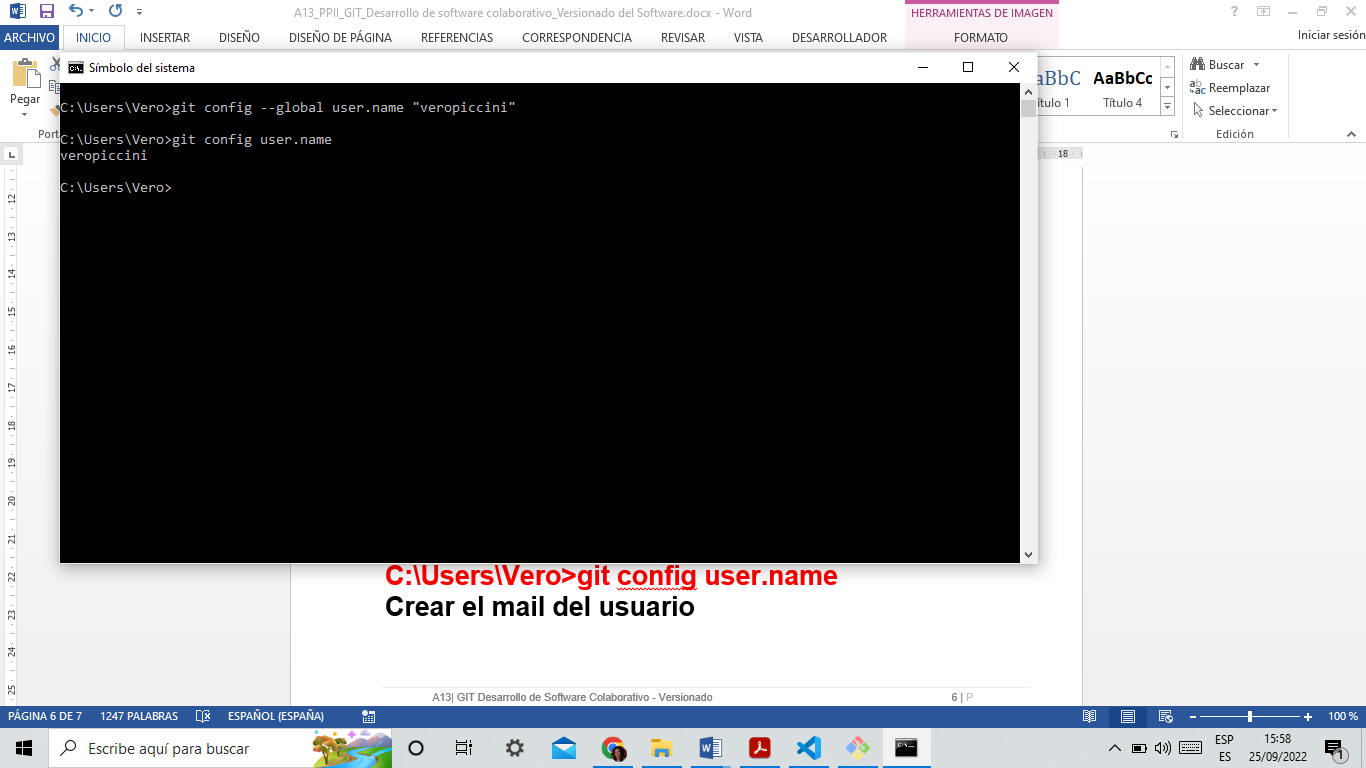
* Windows CMD



# **SEGUNDO PASO- CREAR USUARIOS QUE USEN git EN LA COMPUTADORA**

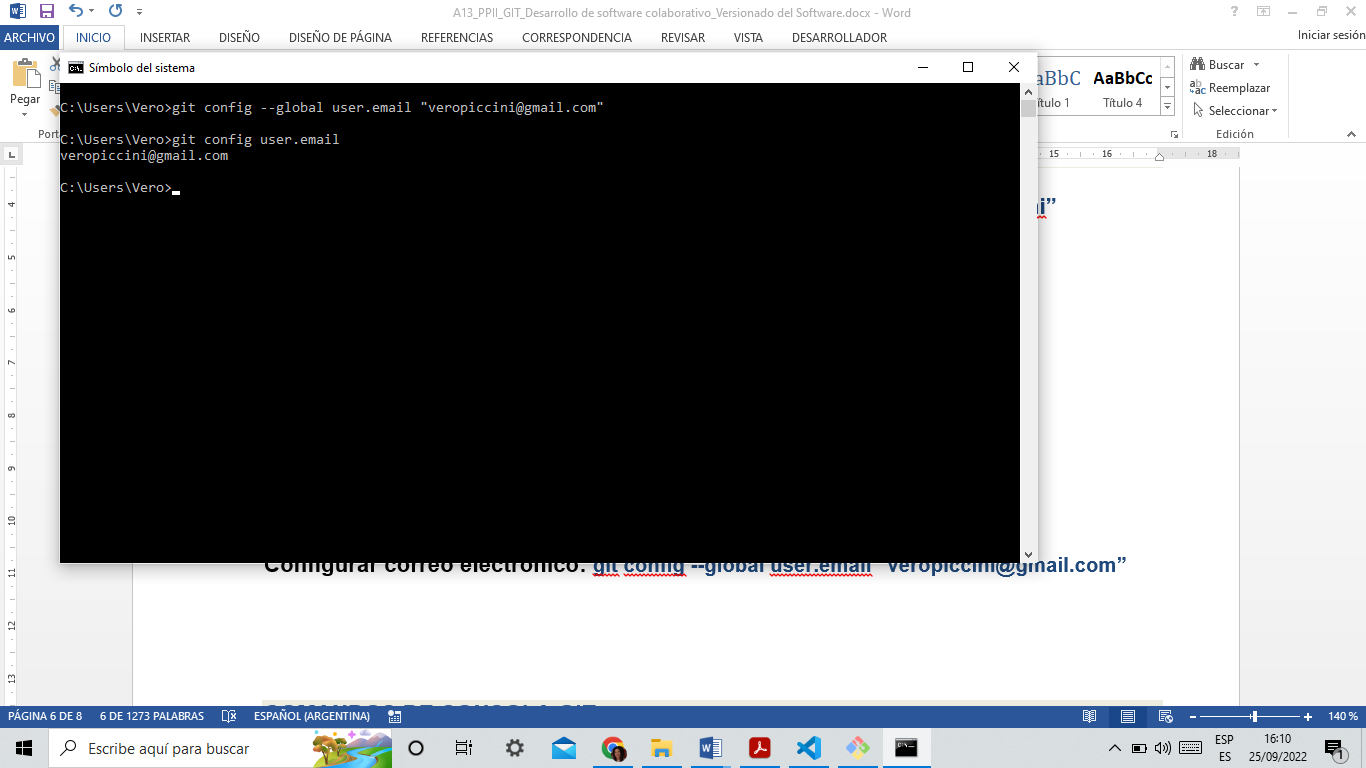
**Configurar nombre de usuario: git config --global user.name “veropiccini”**

Y comprobamos la creación del usuario con **git config user.name**



**Configurar correo electrónico: git config --global user.email “veropiccini@gmail.com”**

**git config user.email**



# **TERCER PASO- IR A UN EDITOR DE CÓDIGO. Por Ej: VISUAL STUDIO CODE**

**Ir a la carpeta creada en el escritorio, botón derecho, Abrir con Visual Studio Code**

**\*\*\* Si Visual Studio Code está en inglés podemos instalar una extensión para traducir al español**

**Link:** <https://marketplace.visualstudio.com/items?itemName=MS-CEINTL.vscode-language-pack-es>

# **COMANDOS DE CONSOLA git**

**Crear una carpeta en el escritorio, botón derecho y clic en Abrir en Visual Studio Code, o abrirla desde Visual Studio Code, Menú Archivo, Abrir carpeta…**

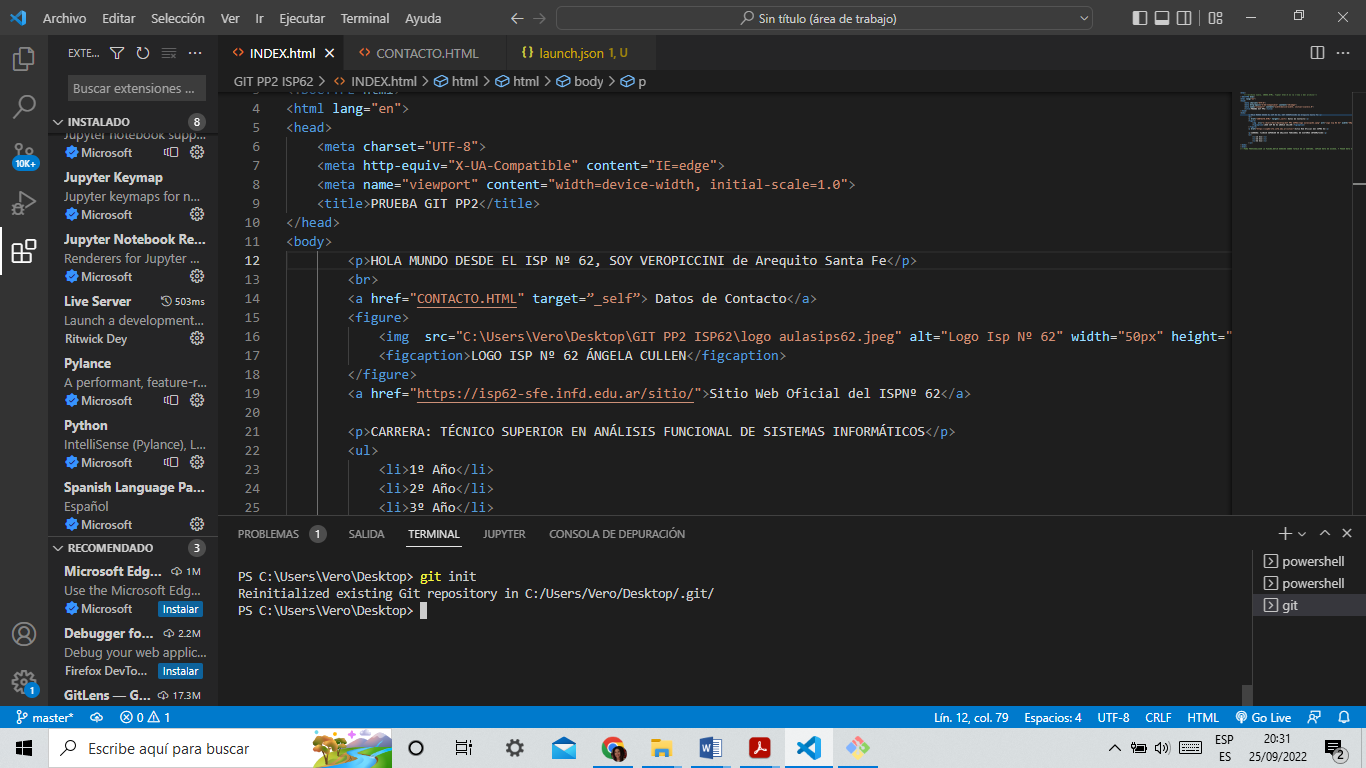
**Recomendación: CREAR UNA CARPETA POR CADA PROYECTO**

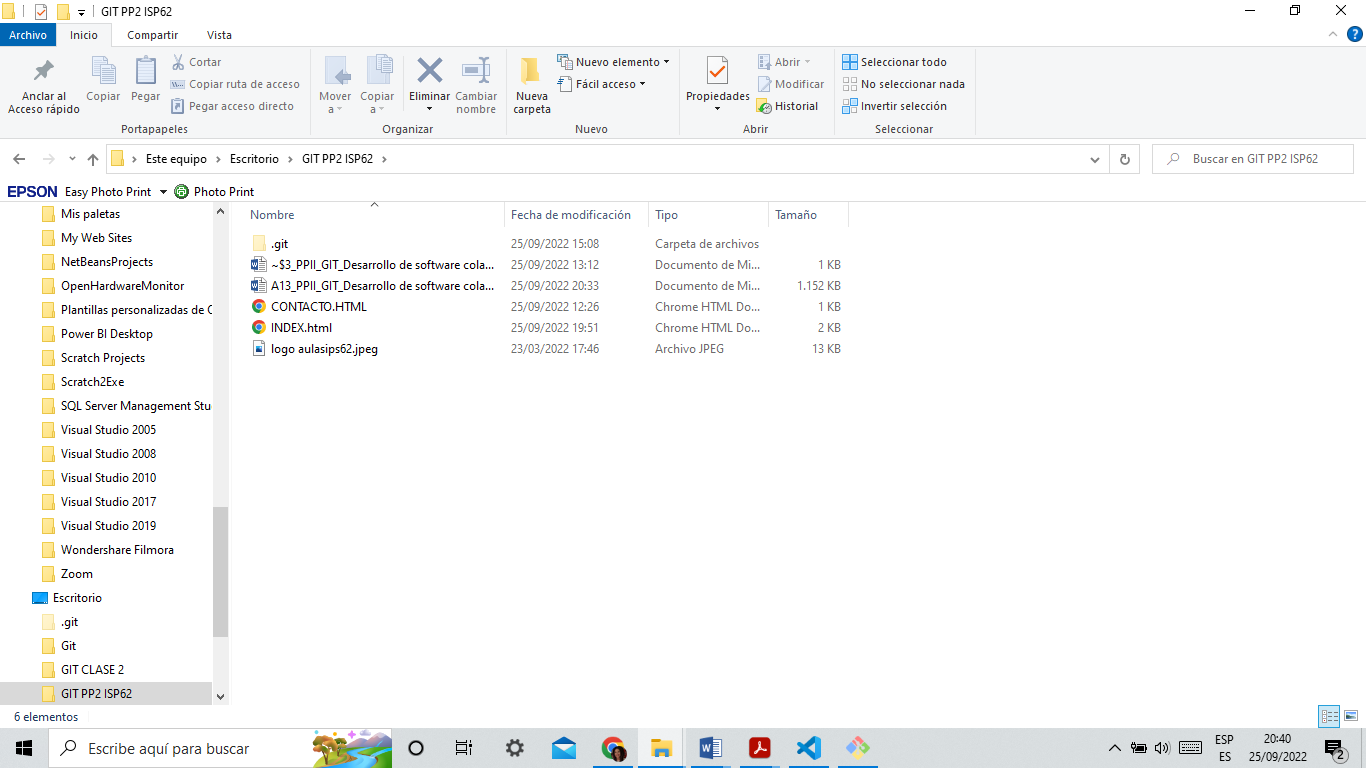
**Usar para ejecutar los comandos la Terminal de Visual Studio Code, o el Git Bush**

**git init : Inicializa y crea un repositorio del proyecto (una carpeta oculta .git dentro de la carpeta del proyecto)**

**Se realiza una sola vez git init al comienzo!!!**

**Cls Limpia la pantalla de la terminal o línea de comandos**





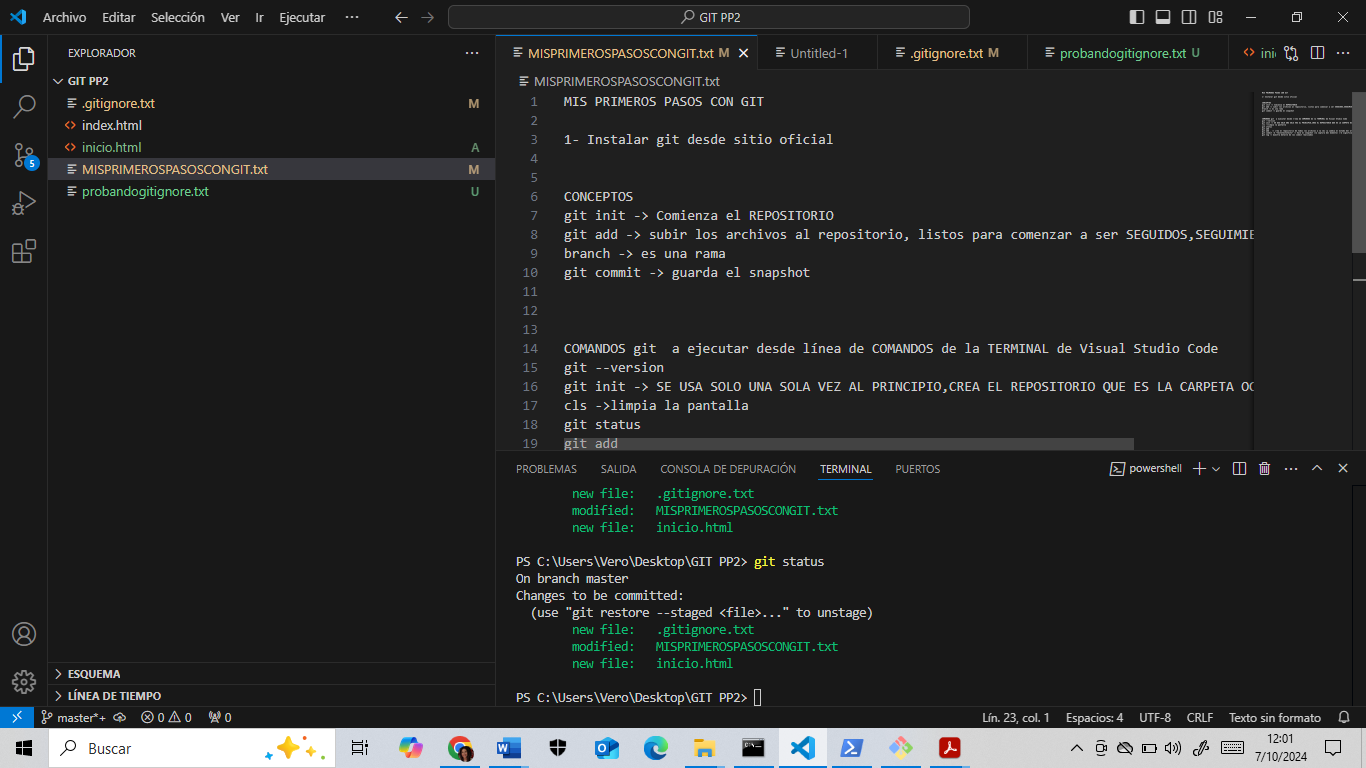
**Git add [nombre de archivo] git add index.html**

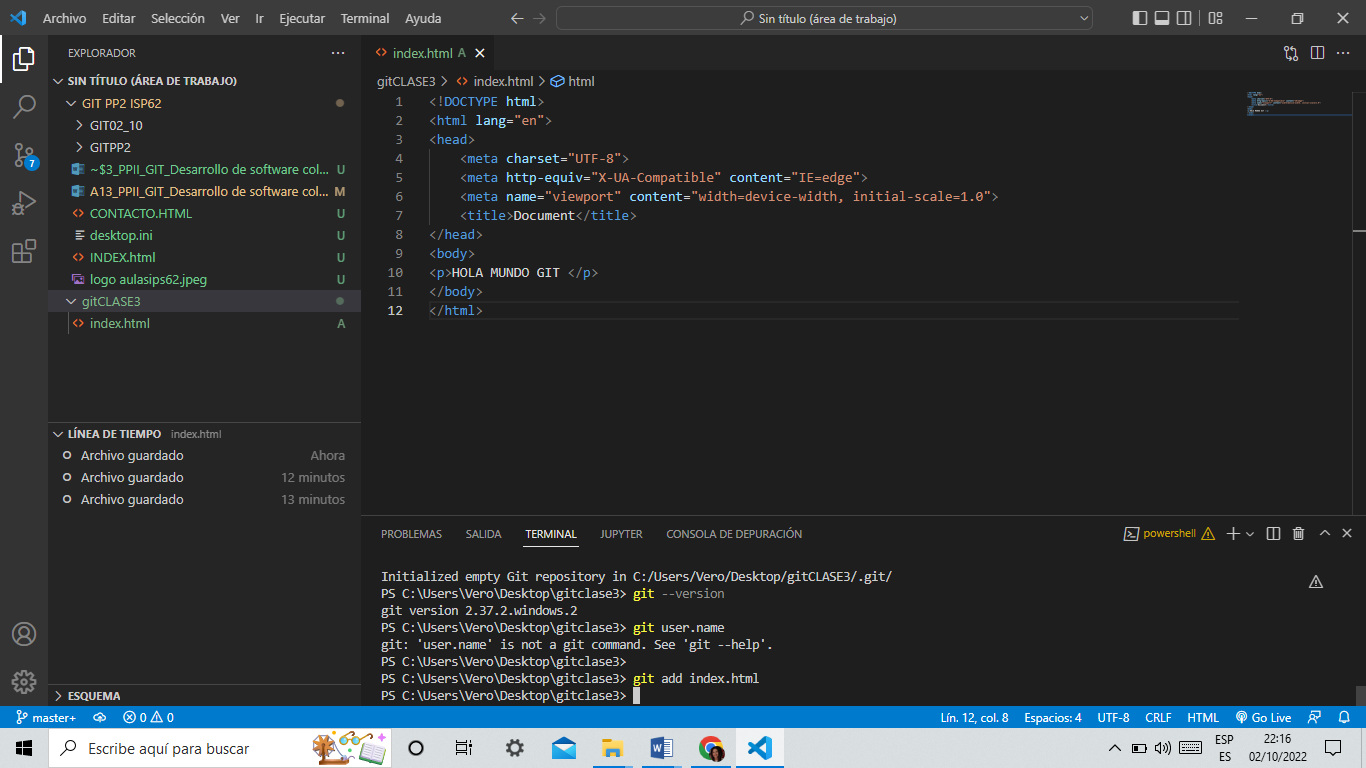
**Git add . (git add . (punto): para agregar todos los archivos a git a la vez**

**Para que algún archivo quede sin hacer el seguimiento de git (A) se crea un archivo de texto en la carpeta llamado .gitignore y dentro escribo los nombres de los archivos. Por ej.: misapuntes.txt y si necesitamos que no siga una carpeta tipeamos una / y el nombre de la carpeta. Por ej.: /personal**

**Como git no hace seguimiento de una carpeta vacía por default. Si queremos mostrar sí o sí su contenido, debemos crear un archivo .gitkeep**

**Cuando los archivos tienen la U significa (untracked) sin trackear, sin seguir o sin seguimiento de git. Luego de ejecutar el git add pasan al estado A**





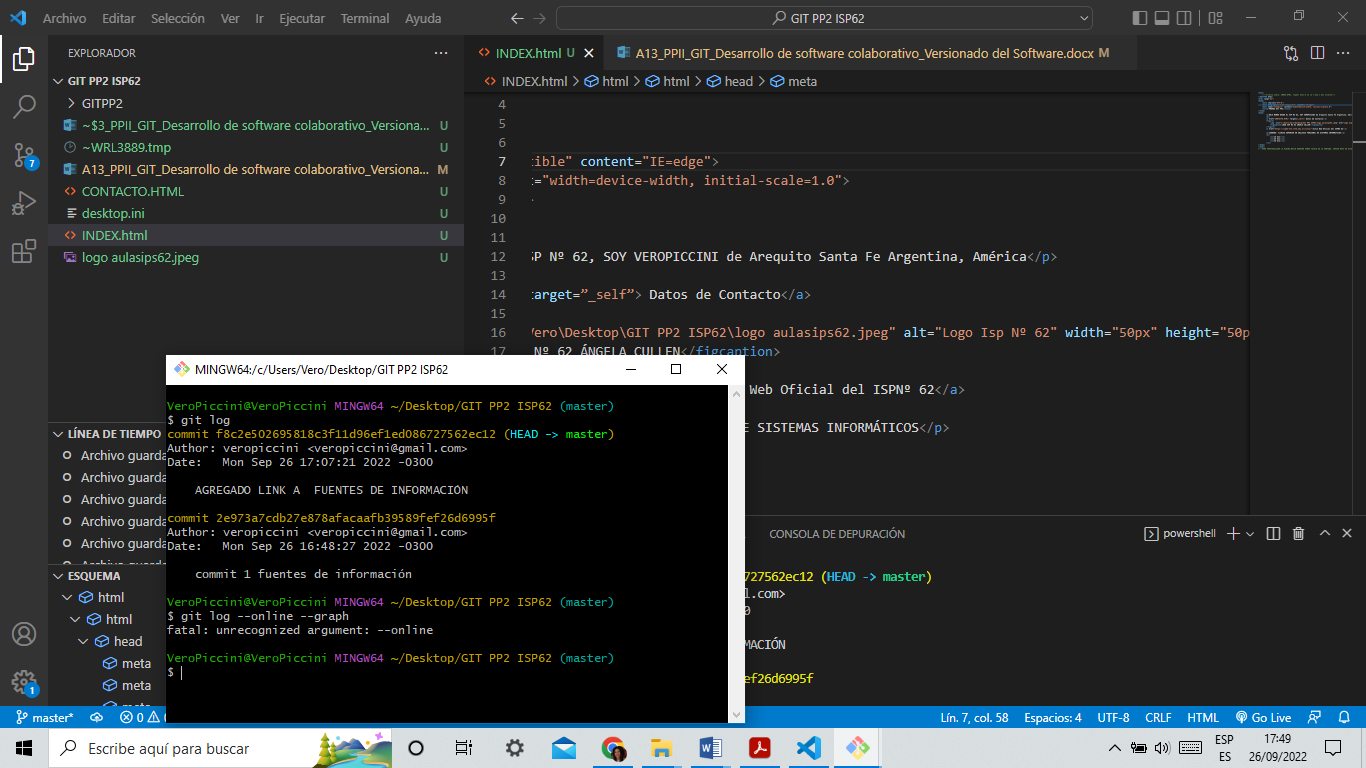
**Git status: es un pedido de info del estado del repositorio**

**Git add [nombre de archivo] git add index.html**

**Git add . (git add . (punto): para agregar todos los archivos a git a la vez**

**Git commit : git commit –m “primer commit en index.html” Saca la captura, el snapshot**

**Git log: muestra información de los comentarios de los commits realizados**



**COMMIT: es decirle a GIT guardáme los estados de mis archivos en este momento (se guarda en la carpeta oculta .git)**

**CREAR UN REPOSITORIO NUEVO EN GITHUB y COPIAR EL LINK**

**git remote add origin https://github.com/veropiccini/gitpp2\_2022.git**

**git remote add origin** [**https://github.com/veropiccini/gitclase3.git**](https://github.com/veropiccini/gitclase3.git) **(el link es del repositorio creado en GitHub**

**PS C:\Users\Vero\Desktop\gitclase3> git remote add origin** [**https://github.com/veropiccini/gitclase3.git**](https://github.com/veropiccini/gitclase3.git)

**git remote –v (muestra si hay repositorios online)**

**git push origin master (agrega todo al repositorio remoto)**

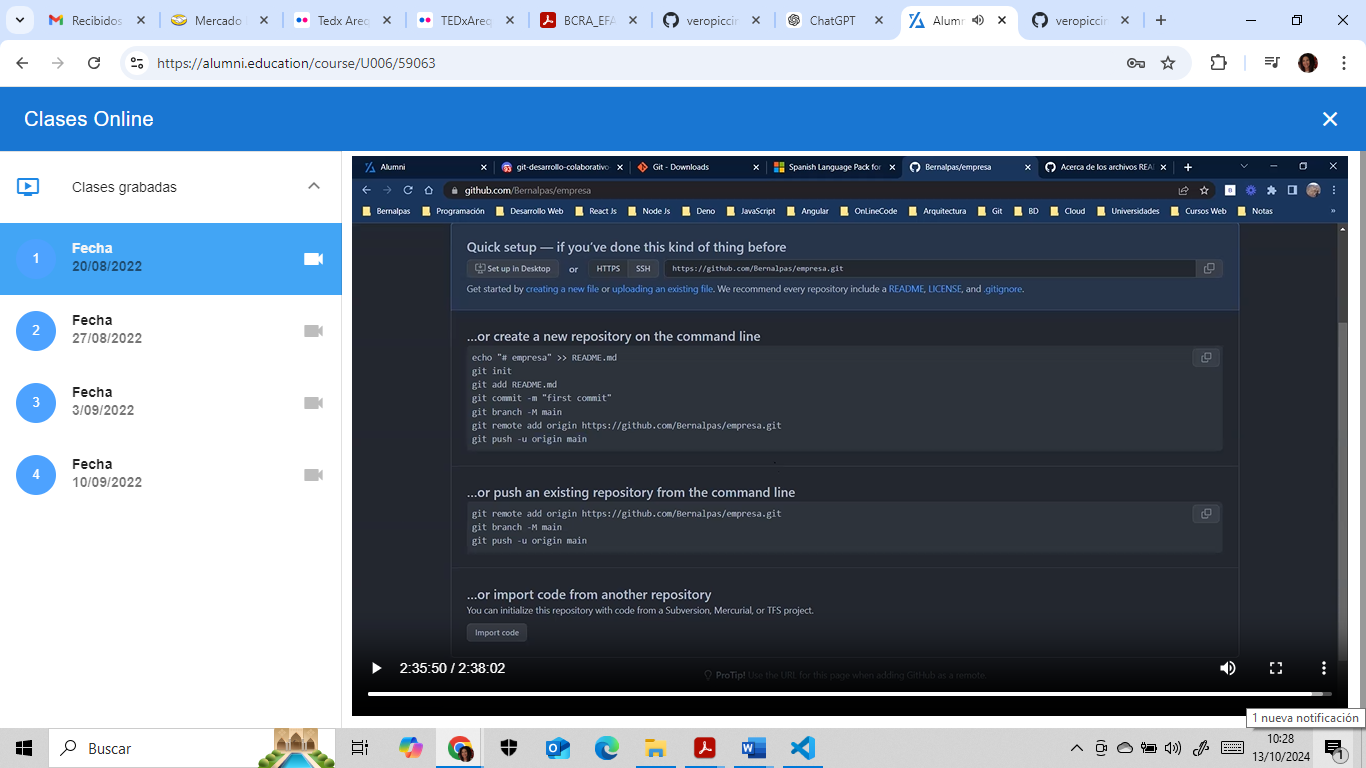
**git push –u origin master (agrego todo al repositorio remoto solicitando nombre usuario y mail todas las veces)**

**git pull (Compara si hay actualizaciones en el código, hacerlo antes de empezar a trabajar)**

**git status**

**MERGE es unir dos ramas o (branch)**

**CHECKOUT permite pararse en los COMMIT que realicé git checkout**



**GUIA DE ACTIVIDADES A16**

**A16.1) Instalación de la plataforma git y registro en github**

**A16.2) Crear un repositorio para guardar el archivo de documento del INFORME FINAL DE PRÁCTICA PROFESIONALIZANTE.**

**A16.3) Crear un repositorio a utilizar con los archivos del código fuente del software a presentar como Proyecto Final de la carrera TÉCNICO SUPERIOR EN ANÀLISIS FUNCIONAL DE SISTEMAS INFORMÀTICOS.**

**FUENTES DE INFORMACIÓN**

<https://gitforwindows.org/>

<https://www.youtube.com/watch?v=NwlZZfcv400> Introducción al Versionado con Git

<https://keepcoding.io/blog/que-es-un-commit-en-git/> comando COMMIT

<https://www.youtube.com/watch?v=59XL48wr6Ow> Versionado GitHub y Word

<https://www.youtube.com/watch?v=qdec2M4NwT0> Conectar Visual Code, git y github

<https://www.freecodecamp.org/espanol/news/guia-para-principiantes-de-git-y-github/>

<https://kinsta.com/es/base-de-conocimiento/git-vs-github/> Diferencias entre git y GitHub

[Licencia Creative Commons](http://creativecommons.org/licenses/by-nc-sa/3.0/)  
git/GITGUB Desarrollo Software Colaborativo - Versionado por  [Veronica Piccini](http://creativecommons.org/choose/www.tecnologiasvp.com.ar) se encuentra bajo una Licencia [Creative Commons Atribución-NoComercial-CompartirDerivadasIgual 3.0 Unported](http://creativecommons.org/licenses/by-nc-sa/3.0/).