**Actividad 4 - Utilizando sistemas de control de versiones**

Autores:

Cristian Camilo Méndez Rodríguez - 100107009

Andrés Johan Vanegas García - 100142140

Corporación Universitaria Iberoamericana.

Facultad de Ingeniería, Ingeniería de Software.

Mantenimiento del software.

Ingeniero Jose David Ruíz Ariza.

Bogotá, Colombia

Mayo, 2024.

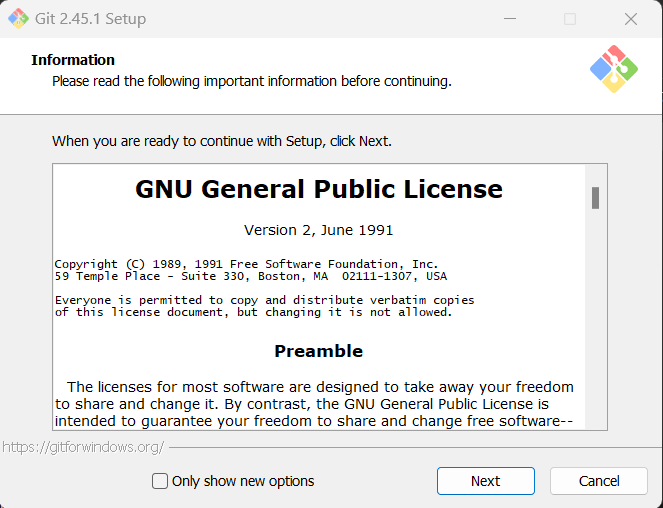
# Introducción.

En el desarrollo de software, los sistemas de control de versiones son herramientas esenciales que permiten gestionar y coordinar los cambios realizados en el código fuente. Entre los sistemas de control de versiones, Git se destaca como uno de los más populares y eficaces. Git es un sistema de control de versiones distribuido que permite a los desarrolladores trabajar de manera simultánea y mantener un historial completo de todas las modificaciones realizadas en el proyecto. Su capacidad para manejar ramificaciones y fusiones de código con eficiencia lo convierte en una herramienta ideal para proyectos colaborativos.

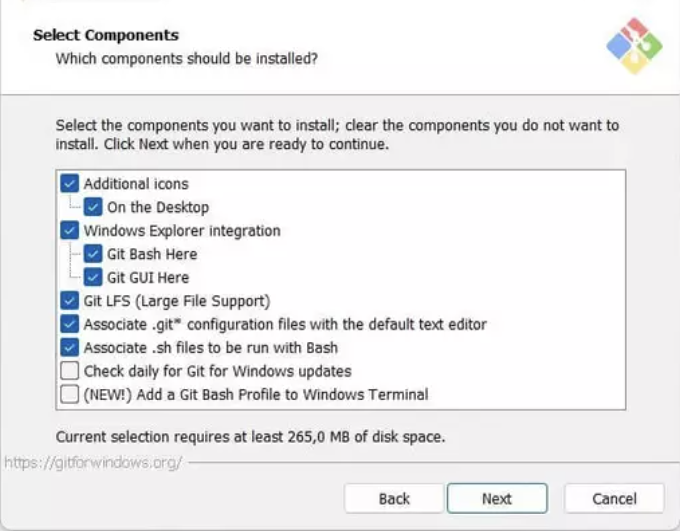
Por otro lado, GitHub es una plataforma que potencia el uso de Git al proporcionar un entorno en línea donde los desarrolladores pueden alojar sus repositorios de Git. GitHub no solo facilita el almacenamiento y la administración de proyectos, sino que también incluye características adicionales como la integración continua, la revisión de código, y la gestión de incidencias y proyectos. Al combinar Git con una interfaz accesible y herramientas colaborativas avanzadas, GitHub se ha convertido en una plataforma predilecta tanto para proyectos de código abierto como para desarrollos privados.

La importancia de Git y GitHub en la industria del software es innegable. Estos sistemas no solo mejoran la productividad y la organización del trabajo en equipo, sino que también fomentan la transparencia y la colaboración a nivel global. Con Git, los desarrolladores pueden experimentar y colaborar sin temor a perder trabajo, mientras que GitHub ofrece una comunidad y un ecosistema que enriquecen el proceso de desarrollo. Estas herramientas se han vuelto indispensables en el flujo de trabajo moderno de desarrollo de software, marcando una evolución significativa en la manera en que los equipos gestionan y desarrollan proyectos.

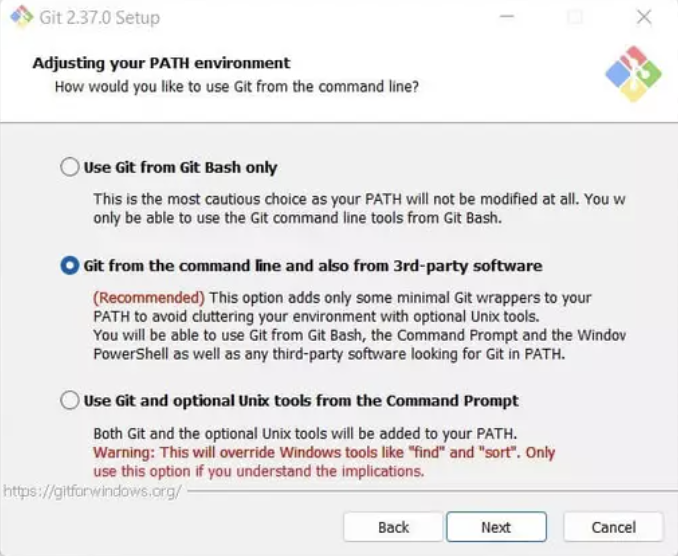
**1.** *Instalar Git:*



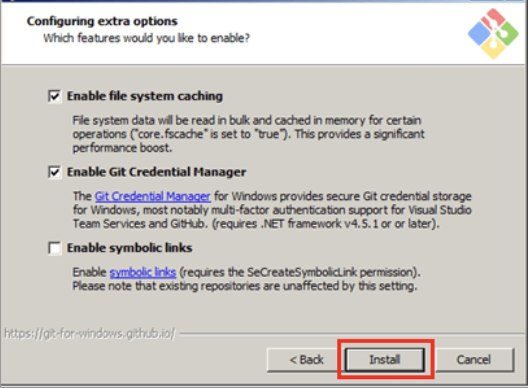
Damos click en “Next” para aceptar los términos de la licencia.



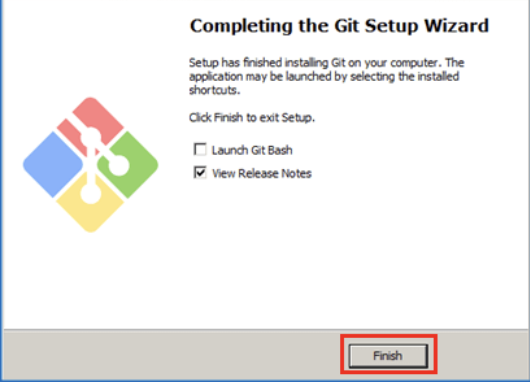
Dejamos las configuraciones por defecto para instalar un ícono en el escritorio, instalar la consola de Git y el entorno gráfico, así como par asociar los tipos de archivos correspondientes.



Marcamos la segunda opción para poder ejecutar Git desde cualquier consola de comandos.

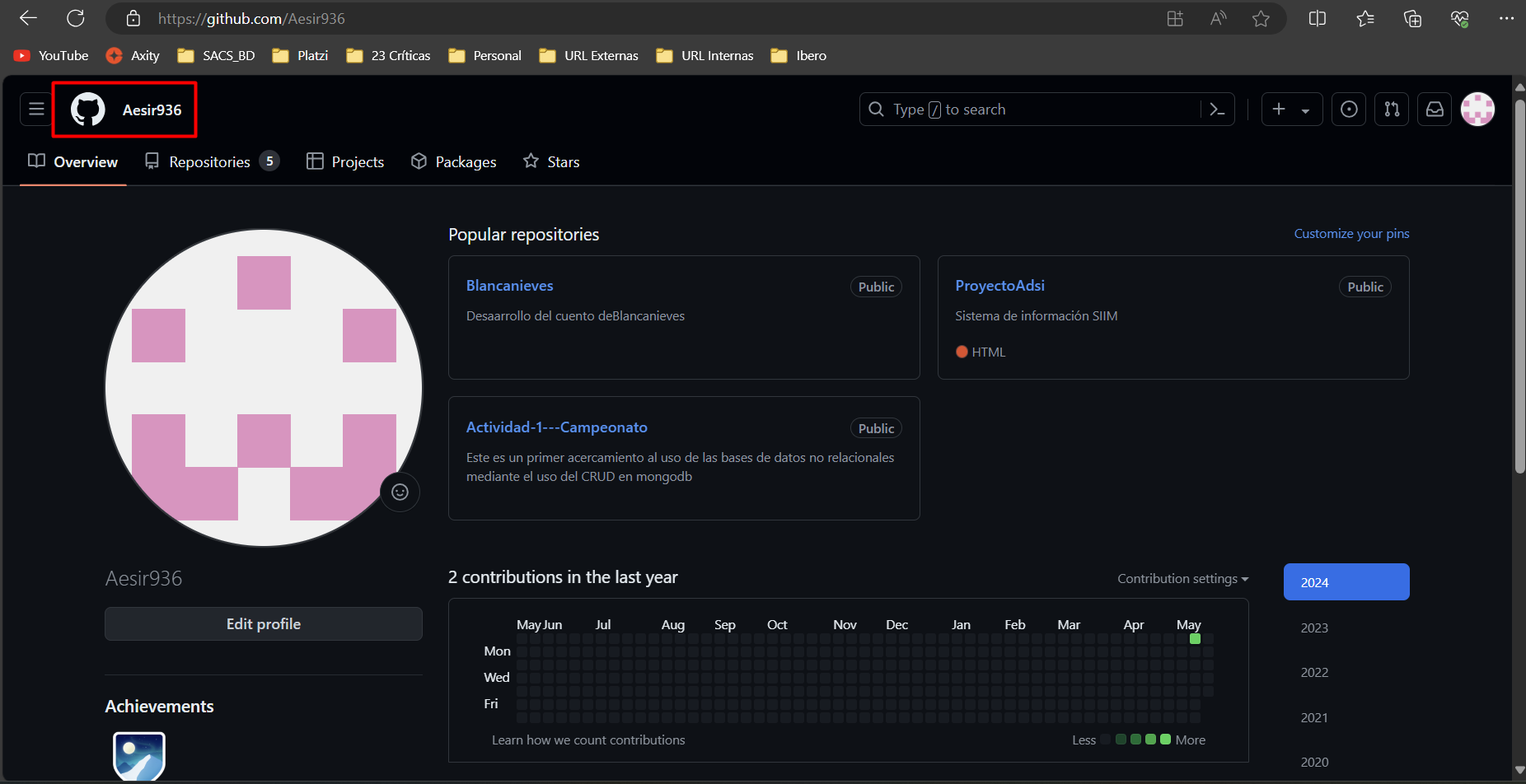


Dejamos las demás opciones avanzadas en la configuración por defecto y damos clic en “Install” para iniciar la instalación.



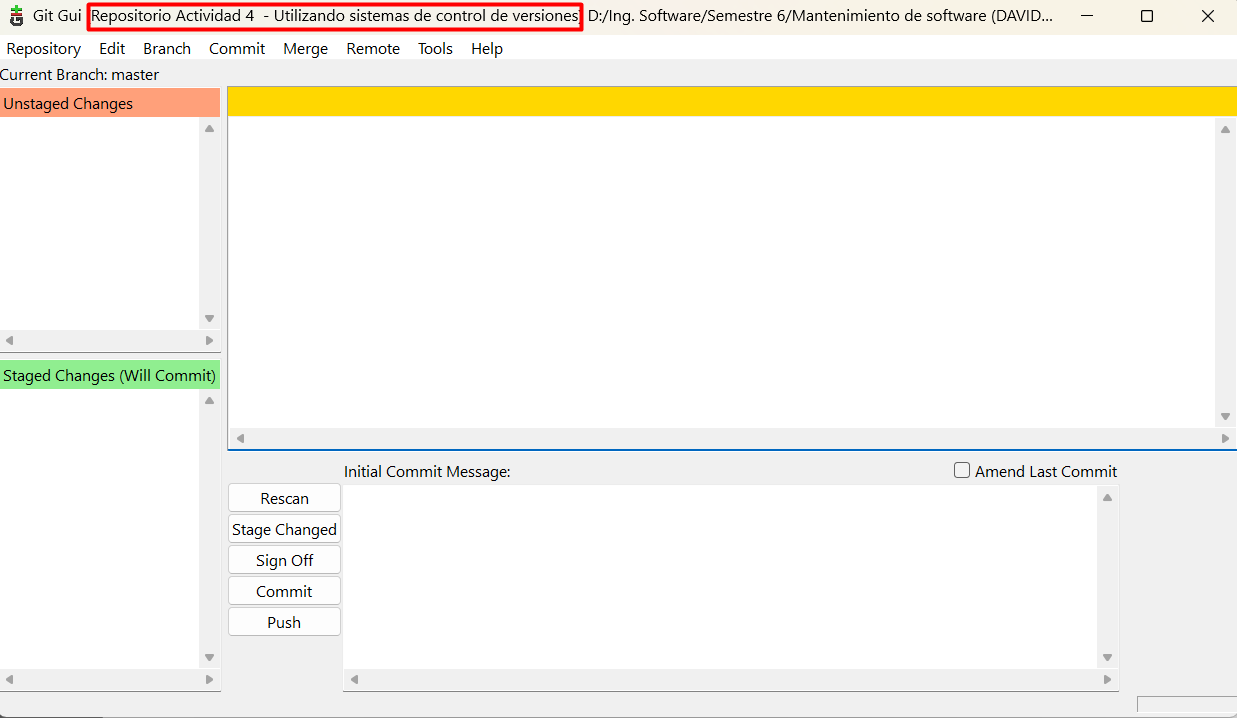
Una vez finalizada la instalación damos clic en “Finish” y ya tendremos Git de manera local en nuestras máquinas.

**2**. *Cuenta de Github:*

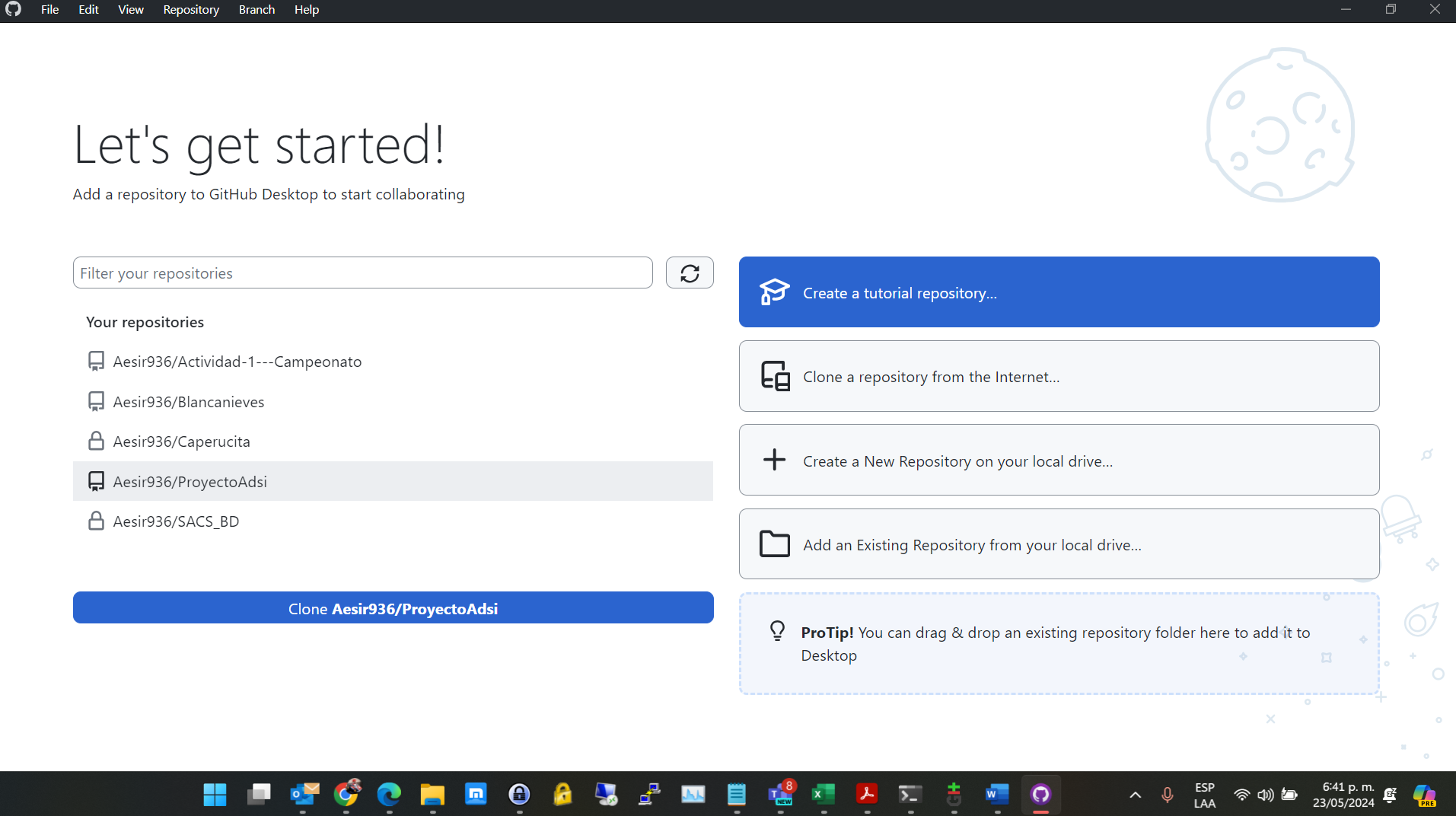


Ya se cuenta con un perfil en Github con el nombre de usuario “Aesir936” sobre el cual se trabajará.

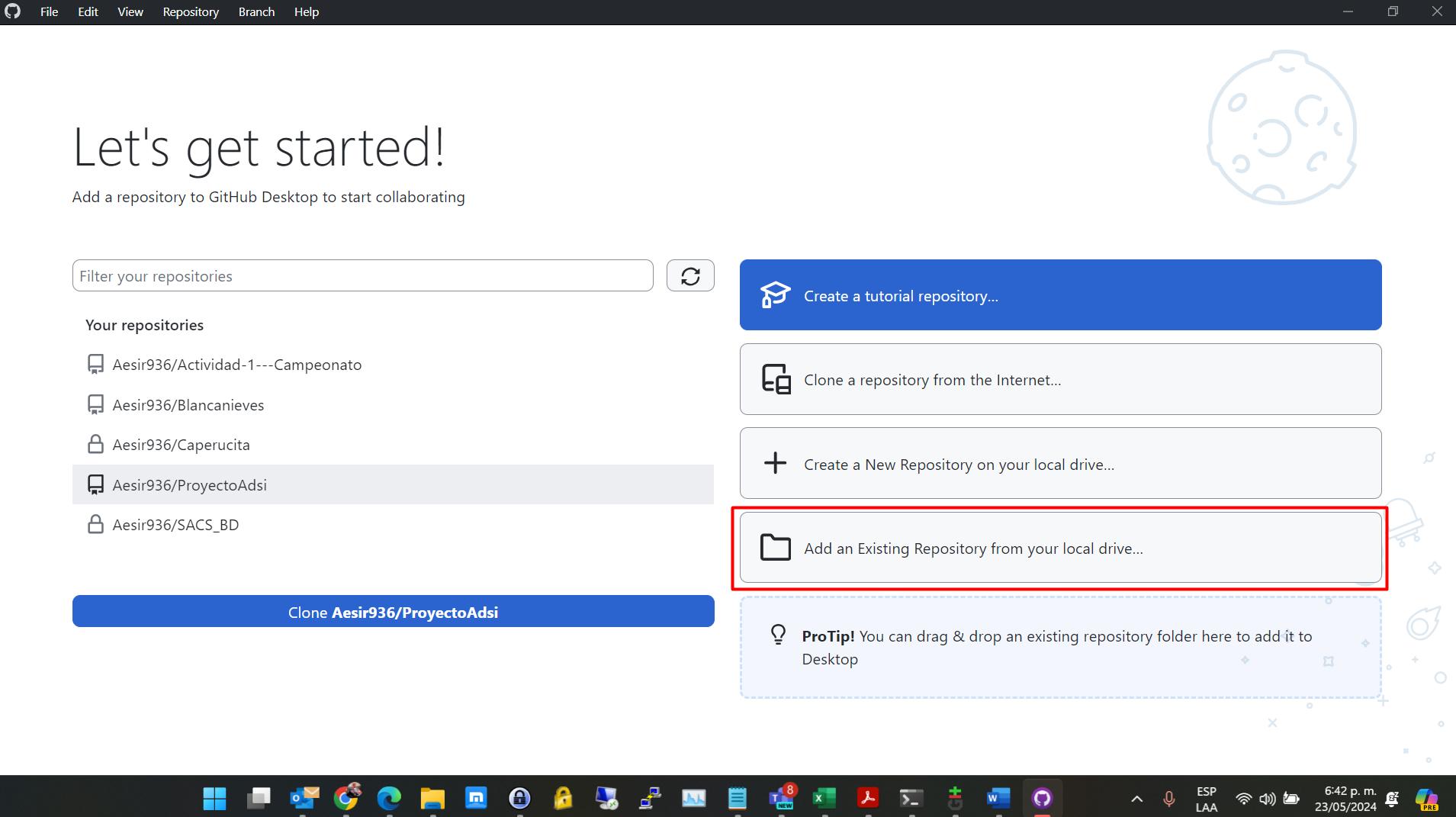
**3**. *Creación del repositorio y primeros comandos de configuración:*



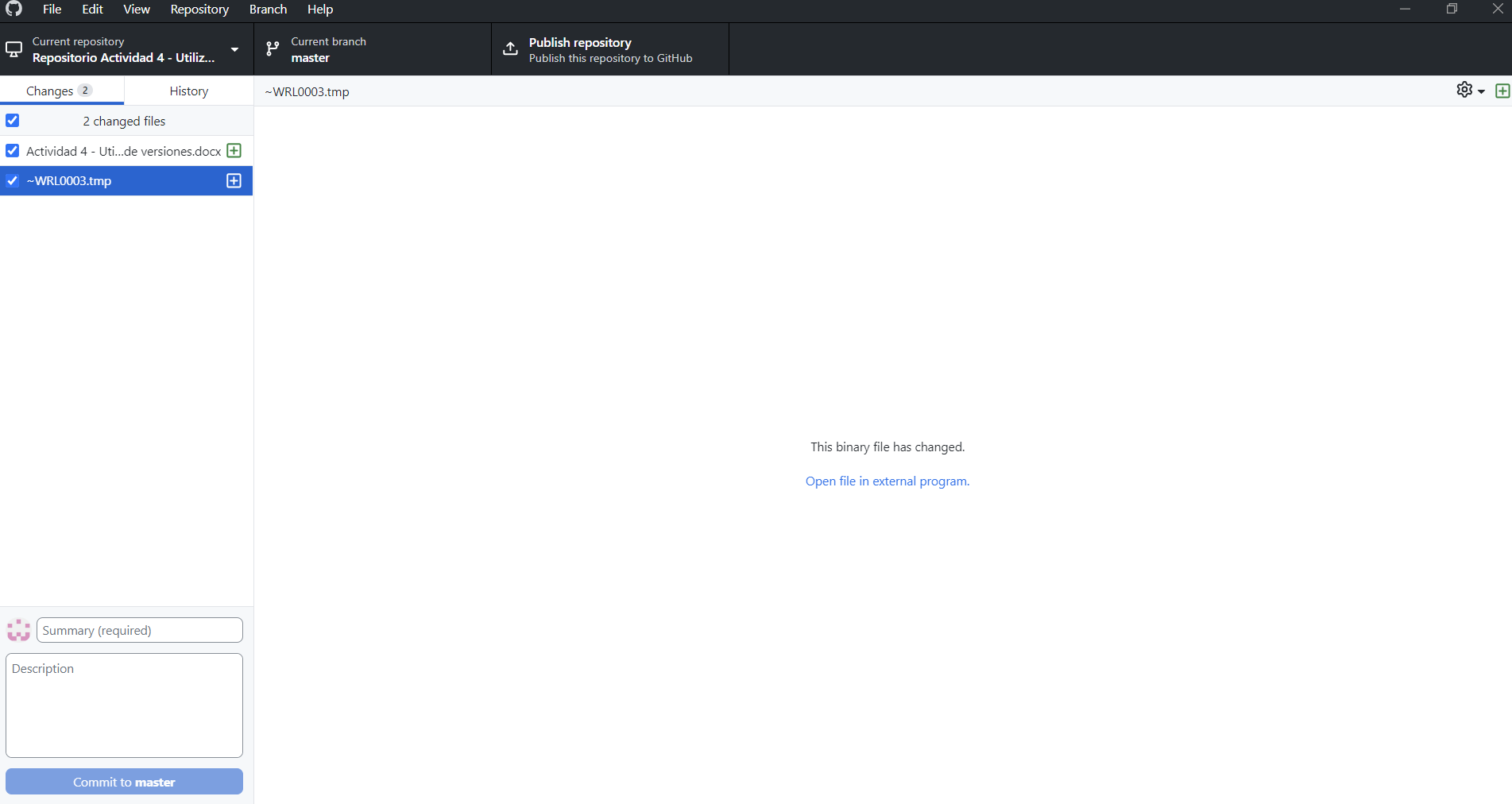
Se crea el repositorio correspondiente a la actividad 4 utilizando la interfaz gráfica incluida con Git “Git Gui”



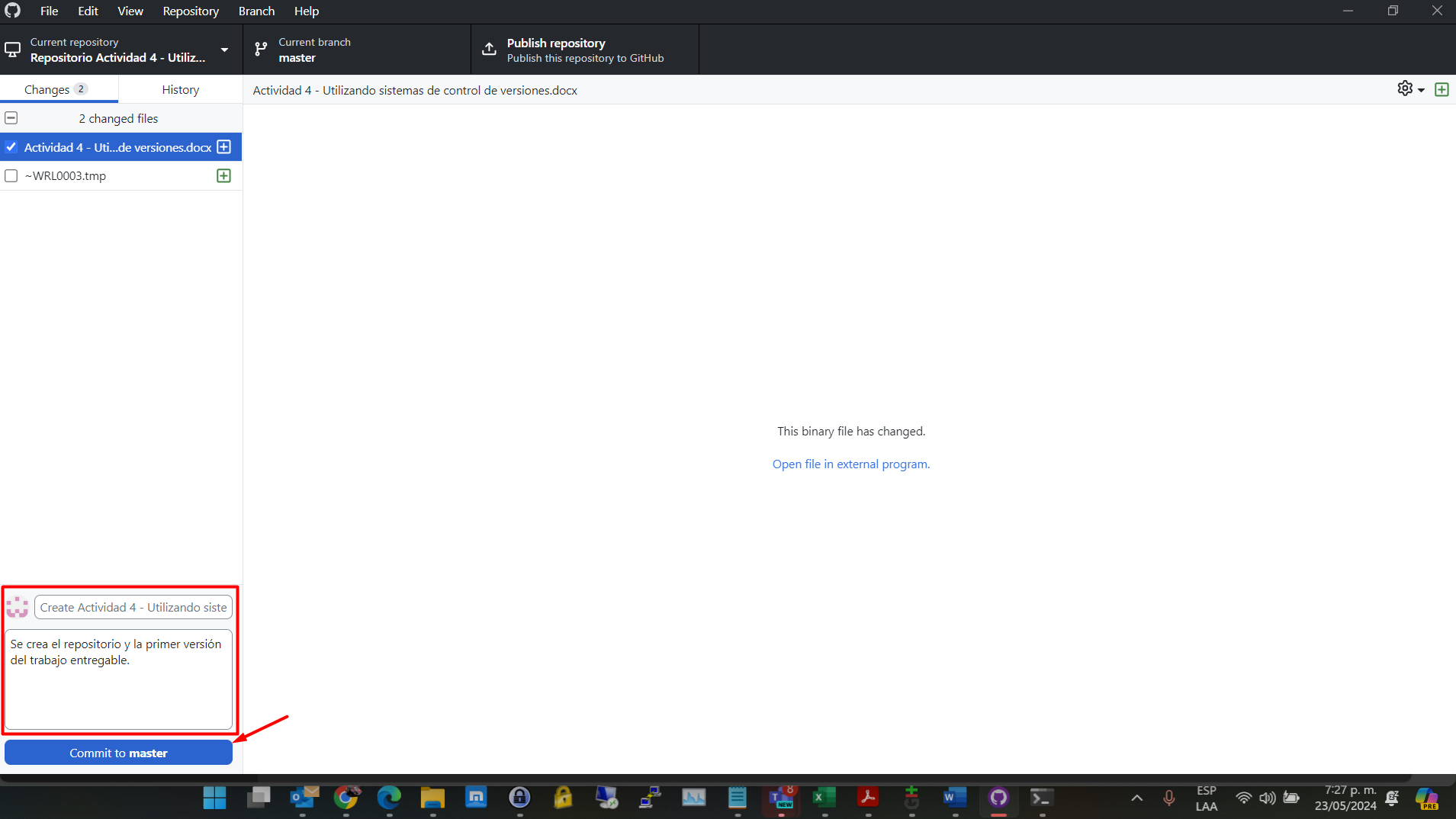
Realizamos la instalación de la versión de escritorio para Github y la sincronizamos con la cuenta anteriormente mencionada “Aesir936” podemos ver que ya existen algunos proyectos anteriores.



Agregamos a nuestra cuenta de Github el repositorio que acabamos de crear localmente. En este caso el programa internamente correrá el comando “git remote add origin https://github.com/Aesir936/Repositorio Actividad 4 - Utilizando sistemas de control de versiones.git”



Hacemos nuestro primer commit especificando en la descripción que se trata de la creación de del repositorio y la primer versión del entregable.



# Conclusiones.

En conclusión, el mantenimiento del software ha experimentado una notable evolución a lo largo del tiempo, adaptándose a los cambios en la industria y las necesidades de los usuarios. Desde sus primeras etapas centradas en la corrección de errores hasta su enfoque actual en la mejora continua y la innovación, el mantenimiento del software ha demostrado ser una parte esencial del ciclo de vida del desarrollo de software.

Además, la gestión eficaz del mantenimiento del software no solo implica abordar problemas y defectos existentes, sino también anticiparse a futuras necesidades y tendencias del mercado. Al invertir en prácticas de mantenimiento preventivo, adaptativo y evolutivo, las empresas pueden garantizar la longevidad y la competitividad de sus productos de software, brindando a los clientes una experiencia mejorada y cumpliendo con sus expectativas en constante cambio.

# Bibliografía.

Bourque, P. & Fairley, R. E. (Eds). (2014). Swebok. Guide to the Software Engineering Body of Knowledge - Versión 3.0. IEEE Computer Society.

Pérez Carvajal, R. J. (2016). Mantenimiento del software (UF1894). Málaga: IC Editorial.