

2025-10-03

Monday, April 18, 2022 12:29 PM

Tema: Clonar un repositorio en una organización de GitHub**Objetivos:**

- Describir el proceso básico de creación y configuración de un repositorio en GitHub.
- Explicar cómo clonar un repositorio y trabajar con él desde Visual Studio Code.
- Mostrar cómo realizar cambios, hacer commits y sincronizarlos con el repositorio remoto.
- Identificar problemas comunes de configuración de Git y su solución.

Desarrollo:**Paso 1: Creación del repositorio**

Primero, se crea el repositorio en la interfaz de GitHub. Se asigna un nombre único al proyecto en el campo "Repository name", se verifica el propietario y se define la visibilidad como "Pública". Una vez completados estos campos, se deja la configuración restante por defecto para finalizar la creación del espacio remoto.

Create a new repository
Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 **General**

Owner * julian-cz2004 / Repository name * camacho-2025-b-jacz-sw-gr2
camacho-2025-b-jacz-sw-gr2 is available.

Great repository names are short and memorable. How about [reimagined-octo-happiness?](#)

Description
0 / 350 characters

2 **Configuration**

Choose visibility * Public
Choose who can see and commit to this repository

Add README Off
READMEs can be used as longer descriptions. [About READMEs](#)

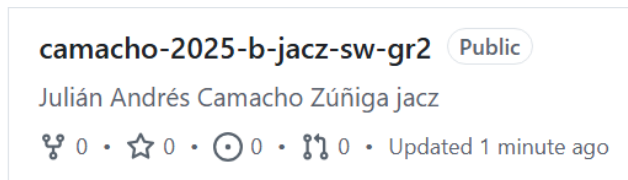
Add .gitignore No .gitignore
.gitignore tells git which files not to track. [About ignoring files](#)

Add license No license
Licenses explain how others can use your code. [About licenses](#)

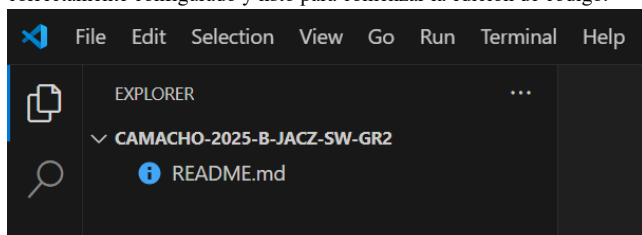
Create repository

Paso 2: Verificación del repositorio creado

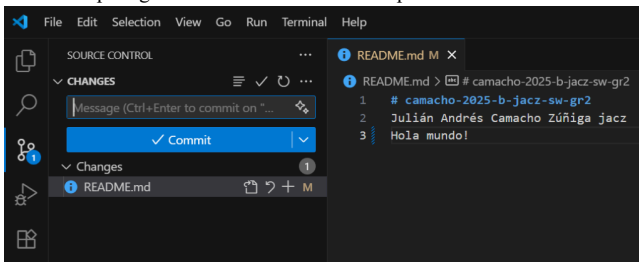
Una vez confirmado el formulario anterior, la plataforma redirige automáticamente a la cabecera del nuevo proyecto. En esta vista se confirma que el repositorio, denominado camacho-2025-b-jacz-sw-gr2, ha sido generado exitosamente dentro de la cuenta del usuario. Se observa la etiqueta "Public", lo que verifica que la visibilidad fue configurada correctamente, y el repositorio queda listo para las siguientes operaciones de clonado y gestión de archivos.

**Paso 3: Apertura del proyecto en Visual Studio Code**

Una vez clonado el repositorio en el equipo local, se abre la carpeta del proyecto en Visual Studio Code. En el panel lateral "Explorer" se verifica que el directorio CAMACHO-2025-B-JACZ-SW-GR2 está activo y contiene el archivo README.md. Esto confirma que el entorno de trabajo está correctamente configurado y listo para comenzar la edición de código.

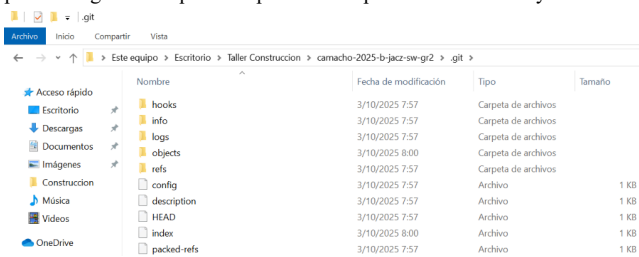
**Paso 4: Modificación de archivos y primer Commit**

Con el proyecto abierto en Visual Studio Code, se procede a editar el archivo README.md. En el editor, se agregan los datos requeridos, como el nombre del estudiante ("Julián Andrés Camacho Zúñiga") y un mensaje de prueba ("Hola mundo!"). El sistema de control de versiones detecta automáticamente estas modificaciones, lo cual se refleja en el panel "Source Control" donde el archivo aparece listado con la etiqueta "M" (Modificado). Finalmente, se escribe un mensaje descriptivo en el cuadro de texto y se presiona el botón "Commit" para guardar estos cambios en el repositorio local.



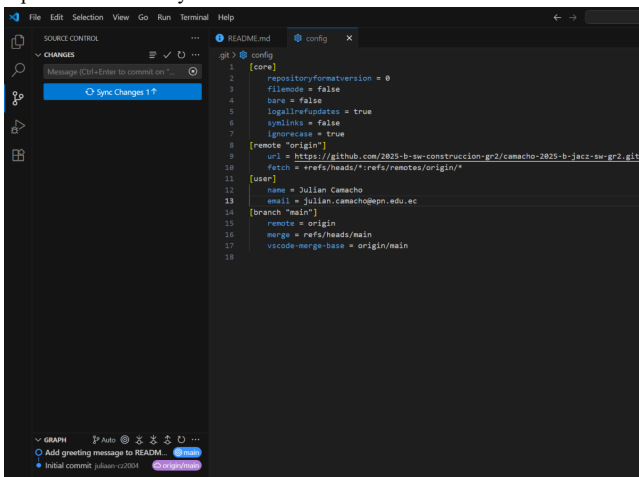
Paso 5: Verificación de la estructura Git local

Para confirmar que el repositorio se ha clonado correctamente a nivel de sistema, se accede a la carpeta del proyecto desde el explorador de archivos. Al habilitar la visualización de "elementos ocultos", se debe observar la carpeta .git. Este directorio es fundamental, ya que almacena toda la configuración del repositorio, el historial de cambios (logs) y las referencias (HEAD); su presencia garantiza que la carpeta es un repositorio Git válido y funcional.



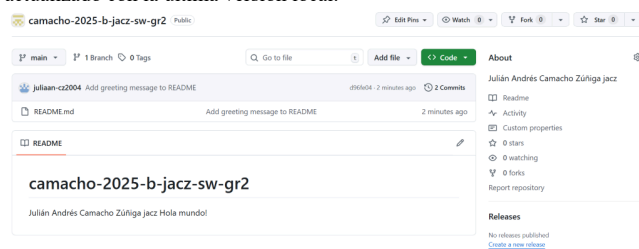
Paso 6: Sincronización con el remoto y verificación de credenciales

Finalmente, se procede a subir los cambios al repositorio remoto utilizando el botón "Sync Changes 1" en el panel de control de código, el cual indica que hay un commit local pendiente de envío. Adicionalmente, se abre el archivo de configuración .git/config para verificar técnicamente que la URL del origen ([remote "origin"]) y las credenciales de usuario (name y email) estén correctamente establecidas, asegurando así que la sincronización se dirija al repositorio correcto y con la autoría adecuada.



Paso 7: Verificación de la actualización en el repositorio remoto

Una vez realizada la sincronización desde Visual Studio Code, se regresa a la interfaz web de GitHub para confirmar la carga de los datos. En la vista principal del proyecto, se verifica que el archivo README.md ahora muestra el texto agregado ("Hola mundo!") y que el historial registra un nuevo cambio con el mensaje "Add greeting message to README". Esto confirma que el ciclo de trabajo se ha completado exitosamente y que el repositorio remoto está actualizado con la última versión local.



packages
No packages published
[Publish your first package](#)

Conclusiones:

- **Integración eficiente de herramientas:** La vinculación entre GitHub y Visual Studio Code simplifica significativamente el flujo de trabajo de desarrollo. Utilizar la interfaz gráfica del editor para gestionar operaciones como Stage, Commit y Sync permite a los desarrolladores centrarse en el código, minimizando la dependencia exclusiva de comandos de consola para tareas rutinarias.
- **Importancia de la configuración del entorno:** La verificación de la carpeta oculta .git y del archivo config demostró ser un paso crítico para asegurar la integridad del repositorio local. Validar correctamente las credenciales de usuario (user.name y user.email) y la URL del remoto (origin) es fundamental para prevenir errores de permisos o de autoría al momento de sincronizar cambios.
- **Ciclo completo de control de versiones:** Se ha comprobado exitosamente el ciclo de vida básico del software mediante Git: desde la creación del repositorio remoto y su clonación local, hasta la modificación de archivos y su posterior publicación. Este proceso garantiza que el trabajo realizado localmente esté respaldado y accesible en la nube, asegurando la trazabilidad de los cambios en el proyecto.

Bibliografía:

- S. Chacon y B. Straub, Pro Git, 2.^a ed. Apress, 2014. [En línea]. Disponible: <https://git-scm.com/book/es/v2>
- GitHub, "Inicio rápido para repositorios - Documentación de GitHub," GitHub Docs. [En línea]. Disponible: <https://docs.github.com/es/repositories/creating-and-managing-repositories/quickstart-for-repositories>
- GitLab, "¿Qué es el control de versiones de Git?," GitLab Topics. [En línea]. Disponible: <https://about.gitlab.com/es/topics/version-control/what-is-git-version-control/>
- D. van Hooland, S. Verborgh, y R. V. de Walle, "Introducción al control de versiones con GitHub Desktop," The Programming Historian en español, abr. 2017. [En línea]. Disponible: <https://programminghistorian.org/es/lecciones/retirada/introduccion-control-versiones-github-desktop>