PRÁCTICO N.º 2: GIT & GITHUB

INTRODUCCIÓN

El estudiante desarrollará competencias para trabajar con Git y GitHub, aplicando conceptos fundamentales de control de versiones, colaboración en proyectos y resolución de conflictos, en un entorno simulado y guiado.

INFORMACIÓN PERSONAL

- Nombre: Jonatán X. Marizza
- Carrera: Tecnicatura de programación a distancia
- Comisión: M2025-16
- Email: Jonatan.marizza@gmail.com

DESARROLLO DEL TRABAJO

1. ¿QUE ES GIT HUB?

GitHub es una plataforma en línea que utiliza Git, un sistema de control de versiones distribuido desarrollado por Linus Torvalds. Ofrece un entorno colaborativo para que los desarrolladores compartan código, monitoreen cambios y contribuyan colectivamente a proyectos. Desde su establecimiento en 2008, GitHub ha experimentado un crecimiento significativo y se ha convertido en un repositorio central para proyectos de código abierto y repositorios privados.

2. ¿CÓMO CREAR UN REPOSITORIO EN GIT?

Para crear un repositorio en GIT, hay que seguir estos pasos:

PASO 1: INICIA SESIÓN EN GITHUB

Primero, iniciar sesión en tu cuenta de GitHub.

PASO 2: ACCEDE A LA OPCIÓN "NEW REPOSITORY"

Una vez iniciada sesión, ir a la esquina superior derecha de la pantalla y clic en el signo "+" y selecciona "New repository".

PASO 3: COMPLETA LOS DETALLES DEL REPOSITORIO

En la página siguiente, hay que completar algunos detalles para el nuevo repositorio:

- Repository Name: Escribe un nombre único para tu repositorio. Este nombre debe ser descriptivo y relevante para el proyecto.
- Description: Es opcional, pero puedes añadir una breve descripción del propósito o contenido del repositorio.
- Public or Private: Selecciona si deseas que tu repositorio sea público (visible para todos) o privado (solo visible para ti y los colaboradores que invites).

PASO 4: CONFIGURACIÓN INICIAL

Es posible inicializar el repositorio con un README, añadir un archivo .gitignore para excluir ciertos archivos del control de versiones, y elegir una licencia.

PASO 5: CREAR EL REPOSITORIO

Después de completar todos los detalles, clic en el botón "Create repository" para finalizar.

PASO 6: CLONAR EL REPOSITORIO

Una vez que el repositorio esté creado, hay que clonarlo en tu máquina local usando el siguiente comando:

- git clone [URL del repositorio]

PASO 7: AÑADIR ARCHIVOS Y HACER COMMITS

Ya con el repositorio clonado en tu local, podemos comenzar a añadir archivos, hacer cambios y realizar commits usando los siguientes comandos:

- git add [nombre del archivo]
- git commit -m "Mensaje descriptivo sobre el cambio realizado"
- git push origin [nombre de tu branch]

***Origin es por defecto el nombre que adopta localmente tu repositorio remoto cuando hacemos un clone, es como una label. Esta puede se cambiada con un: git remote rename origin [nombre nuevo].*

*Para verificar que realmente el label de tu repositorio remoto sea origin se puede ejecutar el comando git remote -v ***

3. ¿CÓMO CREAR UNA RAMA (BRANCH) EN GIT?

Para crear ramas en GIT, hay que seguir estos pasos:

PASO 1: CREAR UNA NUEVA RAMA

Para crear una nueva rama, ejecutar el siguiente comando:

- git branch [nombre de la nueva rama]

Este comando crea una nueva rama en tu repositorio local.

PASO 2: CAMBIAR A LA NUEVA RAMA

Después de crear la nueva rama, hay cambiar a esa rama para comenzar a trabajar en ella. Ejecutar el siguiente comando:

- git checkout [nombre de la nueva rama]

Ahora estamos trabajando en la nueva rama.

PASO 3: REALIZAR CAMBIOS Y COMMITS

Para realizar cambios en los archivos y luego añadirlos y hacer commits ejecutamos los siguientes comandos:

- git add [nombre del archivo] (solo agrega al commit el archivo especificado)
- git add . (agrega al commit todos los archivos modificados, el "." Indica que deben contemplarse todos los de la carpeta actual y subcarpetas)
- git add – u (agrega al commit todos los archivos modificados y eliminados, pero no agrega los nuevo no rastreados)
- git commit -m "Mensaje descriptivo sobre el cambio realizado"

PASO 4: SUBIR LA NUEVA RAMA AL REPOSITORIO REMOTO

Para subir la nueva rama al repositorio remoto, ejecuta el siguiente comando:

- git push origin [nombre de la nueva rama]

Esto enviará tu nueva rama al repositorio remoto, donde otros colaboradores podrán verla y trabajar en ella.

CONCLUSIÓN

Crear una nueva rama en GitHub es un proceso sencillo que permite un desarrollo organizado y colaborativo. Siguiendo estos pasos, podemos asegurarnos de que nuestro trabajo está bien gestionado y que podemos volver a la rama principal cuando los cambios estén listos para ser fusionados.

4. ¿CÓMO CAMBIAR ENTRE RAMAS EN GIT?

Para cambiar entre ramas en GIT, hay que seguir estos pasos:

PASO 1: LISTAR LAS RAMAS DISPONIBLES

Para ver todas las ramas disponibles en tu repositorio, ejecuta el siguiente comando:

- git branch

Este comando te mostrará una lista de todas las ramas, destacando la rama en la que actualmente te encuentras.

PASO 2: CAMBIAR A UNA RAMA ESPECÍFICA

Para cambiar a una rama específica, ejecuta el siguiente comando:

- git checkout [nombre de la rama]

Reemplaza [nombre de la rama] con el nombre de la rama a la que deseas cambiar.

PASO 3: CONFIRMAR EL CAMBIO DE RAMA

Puedes confirmar que has cambiado correctamente de rama utilizando nuevamente el comando git branch. La rama en la que te encuentras actualmente aparecerá destacada con un asterisco.

CONSEJO ADICIONAL

Si deseas cambiar y crear una nueva rama al mismo tiempo, puedes usar el siguiente comando:

- git checkout -b [nombre de la nueva rama]

Este comando cambiará a la nueva rama creada.

5. ¿CÓMO FUSIONAR RAMAS EN GIT?

Para fusionar ramas en Git, hay que seguir estos pasos:

PASO 1: CAMBIAR A LA RAMA PRINCIPAL

Asegurarse de estar en la rama principal (generalmente llamada main o master) donde deseas fusionar los cambios. Ejecuta el siguiente comando:

- git checkout main (o master)

PASO 2: ACTUALIZAR LA RAMA PRINCIPAL

Antes de fusionar, es recomendable actualizar la rama principal. Ejecuta:

- git pull origin main (o master)

PASO 3: FUSIONAR LA RAMA SECUNDARIA

Ahora, fusiona la rama secundaria en la rama principal. Ejecuta el siguiente comando:

- git merge [nombre-de-la-rama-secundaria]

Reemplaza [nombre-de-la-rama-secundaria] con el nombre de la rama que deseas fusionar.

PASO 4: RESOLVER CONFLICTOS

Si hay conflictos, Git lo notificará. Hay que abrir los archivos en conflicto, resolverlos manualmente y luego agregar los cambios resueltos:

- git add [archivo-resuelto]

Una vez que todos los conflictos estén resueltos, confirma la fusión con:

- git commit

PASO 5: VERIFICAR LA FUSIÓN

Finalmente, hay que verificar que la fusión se haya realizado correctamente utilizando el comando:

- git log

Este comando permite ver el historial de commits y confirmar que la fusión fue exitosa.

6. ¿CÓMO CREAR UN COMMIT EN GIT?

Para crear commit en Git, hay que seguir estos pasos:

PASO 1: VERIFICAR EL ESTADO DEL REPOSITORIO

Antes de crear un commit, es importante verificar el estado del repositorio para ver los cambios realizados. Ejecuta el comando:

- git status

PASO 2: AGREGAR CAMBIOS AL ÍNDICE

Agrega los archivos que deseas incluir en el commit utilizando:

- git add [archivo]

O para agregar todos los cambios:

- git add .

PASO 3: CREAR EL COMMIT

Una vez que los archivos están en el índice, crea un commit con el mensaje deseado:

- git commit -m "tu mensaje de commit"

PASO 4: VERIFICAR EL COMMIT

Para asegurar que el commit se realizó correctamente, es posible verificar el historial de commits con:

- git log

7. ¿CÓMO ENVIAR UN COMMIT A GITHUB?

Para enviar un commit a GitHub, hay que seguir estos pasos:

PASO 1: CONECTAR EL REPOSITORIO LOCAL CON GITHUB

Si no estás vinculado con tu GitHub, ejecuta el comando:

- git remote add origin [URL-del-repositorio]

PASO 2: ENVIAR LOS CAMBIOS A LA RAMA REMOTA

Para enviar los commits a la rama remota en GitHub, ejecuta el comando:

- git push origin

Si estás enviando cambios a la rama principal, el comando sería:

- git push origin main

Si es otra rama, reemplaza "main" con el nombre de tu rama específica.

PASO 3: VERIFICAR LOS CAMBIOS EN GITHUB

Asegurarse de que los cambios se han enviado correctamente revisando el repositorio en GitHub.

8. ¿QUÉ ES UN REPOSITORIO REMOTO?

Un repositorio remoto en GitHub es una copia de un proyecto alojada en un servidor remoto, que puede estar en internet o en un servidor externo. El repositorio remoto permite la colaboración entre varios desarrolladores, facilitando el intercambio y la integración de cambios en el código.

9. ¿CÓMO AGREGAR UN REPOSITORIO REMOTO A GIT?

Pasos para agregar un repositorio remoto a Git:

PASO 1: CREAR UN REPOSITORIO EN GITHUB

Ingresar a GitHub y crear un nuevo repositorio. Anotar la URL del repositorio.

PASO 2: INICIALIZAR TU PROYECTO LOCAL

Si el proyecto aún no está inicializado, ejecuta el siguiente comando:

- git init

PASO 3: AGREGAR EL REPOSITORIO REMOTO

Ejecutar el siguiente comando para vincular el proyecto local con el repositorio remoto:

- git remote add origin [URL_del_repositorio]

PASO 4: VERIFICAR LA CONEXIÓN

Asegurarse que el repositorio remoto se ha agregado correctamente con el comando:

- git remote -v

10. ¿CÓMO EMPUJAR CAMBIOS A UN REPOSITORIO REMOTO?

Para “empujar” (push) cambios a un repositorio remoto, hay que seguir los siguientes pasos:

PASO 1: REALIZAR CAMBIOS EN EL PROYECTO LOCAL

Realizar los cambios necesarios en tu proyecto usando un editor de código.

PASO 2: AGREGAR LOS CAMBIOS AL ÁREA DE PREPARACIÓN:

Ejecutar el siguiente comando para agregar los archivos modificados al área de preparación (staging):

- git add .

PASO 3: CONFIRMAR LOS CAMBIOS:

Ejecutar el siguiente comando para confirmar los cambios con un mensaje descriptivo:

- git commit -m "Descripción de los cambios realizados"

PASO 4: EMPUJAR CAMBIOS AL REPOSITORIO REMOTO:

Ejecutar el siguiente comando para enviar los cambios confirmados a la rama principal del repositorio remoto:

- git push origin main

PASO 5: VERIFICAR CAMBIOS EN EL REPOSITORIO REMOTO:

Acceder a GitHub y verificar que los cambios se hayan reflejado en el repositorio remoto. Otra buena práctica es consultar la lista de confirmaciones y los archivos modificados para asegurarte de que todo está correcto.

11. ¿CÓMO TIRAR DE CAMBIOS DE UN REPOSITORIO REMOTO?

Para “tirar de cambios” (pull) de un repositorio remoto, seguir los siguientes pasos:

PASO 1: TIRAR DE CAMBIOS DESDE EL REPOSITORIO REMOTO:

Ejecutar el siguiente comando para obtener los cambios realizados en el repositorio remoto y actualizarlos en el repositorio local:

- git pull origin [nombre_de_la_rama]

PASO 2: RESOLVER CONFLICTOS DE FUSIÓN:

Si hay conflictos de fusión, Git te los marcará para que los resuelvas.

Abrir los archivos afectados, realizar las modificaciones necesarias y confirmar los cambios.

PASO 3: VERIFICAR LA ACTUALIZACIÓN:

Una vez que los cambios se hayan “pulleado” y los conflictos resueltos, verificar que el repositorio local esté actualizado con los cambios del repositorio remoto. Es posible ejecutar git status para comprobar el estado de tu repositorio.

12. ¿QUÉ ES UN FORK DE REPOSITORIO?

Un fork en GitHub es una copia de un repositorio que se hace en la cuenta de un usuario. Permite modificar el código de forma independiente sin afectar el repositorio original. Con un fork, se puede experimentar, hacer cambios y enviar propuestas de mejora al proyecto original.

13. ¿CÓMO CREAR UN FORK DE UN REPOSITORIO?

Para realizar un fork de otro repositorio, es necesario seguir los siguientes pasos:

PASO 1: ACCEDER AL REPOSITORIO ORIGINAL

Ir al repositorio que deseas forkear en GitHub.

PASO 2: HACER CLIC EN EL BOTÓN DE "FORK"

En la parte superior derecha de la página del repositorio, hay un botón de "Fork". Hacer clic en él.

PASO 3: SELECCIONAR TU CUENTA O ORGANIZACIÓN

GitHub te preguntará si quieres forkear el repositorio en tu cuenta personal o en una organización de la que seas miembro. Seleccionar la opción adecuada.

PASO 4: ESPERAR A QUE SE COMPLETE EL FORK

GitHub creará una copia del repositorio en tu cuenta. Este proceso puede tardar unos segundos.

PASO 5: CLONAR EL FORK EN TU MÁQUINA LOCAL

Para trabajar con tu fork, clónalo en tu máquina local usando el comando:

- git clone [URL_del_fork]

14. ¿CÓMO ENVIAR UNA SOLICITUD DE EXTRACCIÓN (PULL REQUEST) A UN REPOSITORIO?

PASO 1: REALIZAR CAMBIOS EN TU RAMA

Realiza los cambios necesarios en la rama local. Asegurarse de que todo funciona correctamente realizando pruebas unitarias.

PASO 2: CONFIRMAR (COMMIT) LOS CAMBIOS

Ejecutar el siguiente comando para confirmar los cambios:

- git commit -m "Descripción de los cambios realizados"

PASO 3: ENVIAR LOS CAMBIOS A GITHUB

Ejecutar el siguiente comando para enviar los cambios a GitHub:

- git push [origin nombre_de_la_rama]

PASO 4: CREAR LA SOLICITUD DE EXTRACCIÓN

En el repositorio original de GitHub buscar un botón llamado"Compare & pull request". Hacer clic en él.

PASO 5: COMPLETAR LA SOLICITUD DE EXTRACCIÓN

Rellenar la información requerida, incluyendo una descripción clara de los cambios. Luego, haz clic en "Create pull request".

15. ¿CÓMO ACEPTAR UNA SOLICITUD DE EXTRACCIÓN?

Para aceptar un pull request debemos seguir los siguientes pasos:

1. Dirigirse al repositorio original en GitHub.

2. Buscar la pestaña "Pull Requests" y hacer clic en ella.

3. Encontrar la “solicitud de extracción” que se desea aceptar y haz clic en el título de la solicitud.

4. Revisar los cambios propuestos en la solicitud de extracción.

5. Si los cambios son satisfactorios, hacer clic en el botón "Merge pull request".

6. Confirmar la fusión haciendo clic en "Confirm merge".

7. Opcionalmente, eliminar la rama si ya no es necesaria, haciendo clic en "Delete branch".

16. ¿QUÉ ES UN ETIQUETA EN GIT?

Una etiqueta en Git es una referencia que apunta a un commit específico en la historia de un repositorio. Se utiliza principalmente para marcar versiones o hitos importantes en el desarrollo del proyecto. Las etiquetas son útiles para identificar puntos clave en el desarrollo y facilitar la navegación entre diferentes versiones del código.

17. ¿CÓMO CREAR UNA ETIQUETA EN GIT?

Para crear una etiquete hay que seguir los siguientes pasos:

1. Ir al repositorio clonado en la máquina local.

2. Asegurarse de estar en la rama correcta donde se quiere crear la etiqueta.

3. Crear etiqueta:

ETIQUETA LIGERA (LIGHTWEIGHT)

Es simplemente un nombre que apunta al commit actual (u otro que indiques):

Desde la rama o commit actual:

- git tag v1.0

Para etiquetar un commit específico:

- git tag v1.0 ab12cd3

✓ **No guarda** metadatos extra (autor, fecha o mensaje).

✓ Es como un “alias” inmutable al SHA del commit.

ETIQUETA ANOTADA (ANNOTATED)

Incluye metadatos (nombre del creador, fecha, mensaje) y está firmada opcionalmente:

- git tag -a v1.0 -m "Lanzamiento de la versión 1.0"

O directamente:

- git tag -a v1.0 -m "Lanzamiento de la versión 1.0" ab12cd3

a = annotated
m = mensaje de la etiqueta

Es posible firmarla con **-s** si se tiene configurado GPG (GNU Privacy Guard):

- git tag -s v1.0 -m "Firma de v1.0"

4. Confirmar que la etiqueta se ha creado correctamente usando git tag

18. ¿CÓMO ENVIAR UNA ETIQUETA A GITHUB?

Para enviar la etiqueta al repositorio remoto, usar: git push origin v1.0

19. ¿QUÉ ES UN HISTORIAL DE GIT?

Es una lista de todos los cambios realizados, quién los hizo, cuándo se hicieron y los mensajes de confirmación asociados. Es una herramienta fundamental para revisar el desarrollo del código y entender la evolución del proyecto.

20. ¿CÓMO VER EL HISTORIAL DE GIT?

Para ver el historial de GIT, hay que seguir los siguientes pasos:

Ejecutar el comando git log para ver una lista detallada de todos los cambios realizados, incluyendo quién los hizo, cuándo se hicieron y los mensajes de confirmación asociados.

Opcionalmente, es posible ejecutar los siguientes modificadores para personalizar la salida del historial:

- git log --oneline: muestra una versión simplificada del historial donde cada commit aparece en una sola línea.
- git log --graph: muestra un gráfico que representa visualmente la estructura del historial.

21. ¿CÓMO BUSCAR EN EL HISTORIAL DE GIT?

Para buscar en el historial de GIT, sigue los siguientes pasos:

1. EJECUTE EL COMANDO `GIT LOG`:

Para visualizar una lista detallada de todos los cambios realizados, incluyendo quién los realizó, cuándo se efectuaron y los mensajes de confirmación asociados.

2. USAR EL MODIFICADOR `--GREP`:

Seguido de la palabra clave que se quiere buscar. Por ejemplo, para localizar todos los commits que contienen la palabra "bug":

- git log --grep=bug

3. REALICE BÚSQUEDAS MÁS COMPLEJAS:

Usando expresiones regulares con el modificador `--grep`. Por ejemplo, para encontrar todos los commits que contienen palabras que comienzan con "fix":

- git log --grep="^fix"

4. BUSCAR POR AUTOR ESPECÍFICO:

utilice el modificador `--author` seguido del nombre del autor. Por ejemplo, para buscar commits efectuados por "Juan Pérez":

- git log --author="Juan Pérez"

5. BUSCAR EN LOS CAMBIOS INTRODUCIDOS EN LOS COMMITS:

(y no solo en los mensajes de confirmación), usar el modificador `-S` seguido de la palabra clave. Por ejemplo, para buscar cambios que contienen la palabra "error":

- git log -S error

22. ¿CÓMO BORRAR EL HISTORIAL DE GIT?

Para borrar el historial de Git de manera completa y permanente:

1. ELIMINAR EL HISTORIAL:

Para borrar todo el historial, primero eliminar el contenido de la carpeta .git. Para ello ejecutar el siguiente comando:

- rm -rf .git

2. INICIALIZAR UN NUEVO REPOSITORIO:

Después de eliminar la carpeta .git, hay que inicializar un nuevo repositorio:

- git init

3. AÑADIR TODOS LOS ARCHIVOS:

Añadir todos los archivos del proyecto al nuevo repositorio:

- git add .

4. REALIZAR EL PRIMER COMMIT:

Realizar el primer commit con los archivos añadidos:

- git commit -m "Primer commit tras borrar historial"

5. CONFIGURA EL REPOSITORIO REMOTO:

Añadir la URL del repositorio remoto nuevamente:

- git remote add origin

6. FORZAR EL PUSH:

Finalmente, realizar un push forzado para actualizar el repositorio remoto con el nuevo historial:

- git push -f origin

23. ¿QUÉ ES UN REPOSITORIO PRIVADO EN GITHUB?

Un repositorio privado en GitHub es un espacio de almacenamiento de código fuente al que solo tienen acceso los usuarios que el propietario del repositorio ha autorizado. A diferencia de los repositorios públicos, los repositorios privados no pueden ser vistos ni clonados por

personas no autorizadas, lo que proporciona una mayor seguridad y control sobre el contenido.

24. ¿CÓMO CREAR UN REPOSITORIO PRIVADO EN GITHUB?

1. Iniciar sesión en GitHub.

2. Hacer clic en el botón "New" en la esquina superior derecha de la página.

3. Introducir un nombre para el repositorio en el campo "Repository name".

4. Marcar la opción "Private" para que el repositorio sea privado.

5. Hacer clic en el botón "Create repository".

25. ¿CÓMO INVITAR A ALGUIEN A UN REPOSITORIO PRIVADO EN GITHUB?

1. Acceder GitHub y dirigirse al repositorio privado que se desea compartir.

2. En la página del repositorio, hacer clic en la opción “Settings” situada en la barra de menú del lado derecho.

3. En el menú de la izquierda, buscar y seleccionar la opción “Manage access”.

4. Hacer clic en el botón “Invite a collaborator”.

5. Introducir el nombre de usuario o el correo electrónico de la persona que deseamos invitar.

6. Seleccionar el usuario correcto de la lista que aparece y hacer clic en “Add collaborator”.

26. ¿QUÉ ES UN REPOSITORIO PÚBLICO EN GITHUB?

Un repositorio público en GitHub es un espacio de almacenamiento donde se puede guardar y compartir cualquier tipo de proyecto y código con otros usuarios de GitHub. Al ser público, cualquier persona que tenga acceso a GitHub puede ver, clonar y descargar el contenido del repositorio. Esto es útil para proyectos de código abierto y colaborativos donde se busca la contribución de una comunidad más amplia.

27. ¿CÓMO CREAR UN REPOSITORIO PÚBLICO EN GITHUB?

1. Dirigirse a GitHub.

2. En la esquina superior, hacer clic en el signo "más" (+) y seleccionar “New repository”.

3. Introducir un nombre para el nuevo repositorio en el campo “Repository name”.

5. Seleccionar la opción “Public” para que el repositorio sea visible para todos.

6. Marcar la casilla “Initialize this repository with a README” si deseas incluir un archivo README inicial.

7. Hacer clic en el botón “Create repository” para finalizar la creación del repositorio.

28. ¿CÓMO COMPARTIR UN REPOSITORIO PÚBLICO CON GITHUB?

Para compartir un repositorio público podemos directamente compartir la URL de este. O bien ir a:

Settings → **Collaborators & teams**.

Añade el nombre o email de tu colaborador y pulsa **Add collaborator**.

Esa persona recibirá una invitación y, al aceptarla, podrá hacer git push directamente.