Actividad: Control de versiones de software con Git

Objetivos

Parte 1: Inicializar Git

Parte 2: Preparar y confirmar un archivo en el repositorio de Git

Parte 3: Gestión del archivo y seguimiento de los cambios

Parte 4: Ramas y fusión

Parte 5: Gestión de conflictos

Parte 6: Integración de Git con GitHub

Aspectos básicos/Situación.

En esta actividad, explorará los fundamentos del sistema de control de versiones distribuido Git, incluyendo la mayoría de las feature que necesita conocer para colaborar en un proyecto de software. También integrará su repositorio Git local con el repositorio GitHub basado en la nube.

Inicializando Git

En esta parte, inicializará un repositorio de Git.

Abrir una terminal donde vas a realizar la actividad.

Inicializar un repositorio de Git.

Utilizar el comando **ls** para mostrar el directorio actual. Recuerde que comandos distinguen entre mayúsculas y minúsculas.

~$ **ls**

A continuación, configure la información de usuario que se utilizará para este repositorio local (obvia si ya has realizado esto). Esto asociará su información con el trabajo que contribuye a un repositorio local. Utilice su nombre en lugar de "Usuario test" para el nombre entre comillas. Utilice @example.com para su dirección de correo electrónico.

Nota: Esta configuración puede ser cualquier cosa que desee en este momento. Sin embargo, cuando restablezca estos valores globales, usará el nombre de usuario de su cuenta de GitHub. Si lo desea, puede usar su nombre de usuario de GitHub ahora.

~$ **git config —user.name global "Usuario test "**

~$ **git config —global user.email s@ejemplo.com**

En cualquier momento, puede revisar esta configuración con el comando **git config —list**.

~$ **git config —list**

User.name= Usuario test

user.email= s@ejemplo.com

Haga un directorio **git-intro** y cambie el directorio en él:

$ **mkdir git-intro**

$ **cd git-intro**

/git-intro$

Utilice el comando **git init** para inicializar el directorio actual (git-intro) como repositorio de Git. El mensaje que se muestra indica que ha creado un repositorio local dentro del proyecto contenido en el directorio oculto **.git**. Aquí es donde se encuentra todo el historial de cambios. Puedes verlo con el comando **ls -a**.

/git-intro$ **git init**

Repositorio Git vacío inicializado en /home/.../git-intro/.git/

/git-intro$ **ls -a**

. .. .git

...

A medida que trabaje en su proyecto, querrá comprobar qué archivos han cambiado. Esto es útil cuando está enviando archivos al repositorio, y no desea comprometerlos todos. El comando **git status** muestra los archivos modificados en el directorio de trabajo que se almacenan en etapas para su siguiente confirmación.

/git-intro$ git status

Este mensaje le dice:

Estas en una rama master.

El mensaje de confirmación es la confirmación inicial.

No hay cambios que comprometer

Verá que el estado de su repositorio cambiará una vez que se agreguen archivos y se comience a realizar cambios.

/git-intro$ **git status**

En la rama master

Aún no hay confirmaciones

Nada que confirmar (create/copy files and use "git add" to track)

/git-intro$

Preparando y confirmando un archivo en el repositorio

En esta parte creará un archivo, pondrá en escena ese archivo y confirmará ese archivo en el repositorio de Git.

Crear un archivo.

El repositorio **git-intro** se creó, pero está vacío. Usando el comando **echo**, crea el archivo CC3S2.txt con la información contenida entre comillas.

/git-intro$ **echo "Estoy en camino para pasar el examen CC3S2" >CC3S2.txt**

git-intro$

Utilizar el comando **ls -la** para verificar el archivo, así como el directorio.git, que están en el directorio **git intro**. A continuación, utilice **cat** para mostrar el contenido de CC3S2.txt.

/git-intro$ **ls -la**

total 16

drwxrwxr-x 3 ... 4096 Abr 17 20:38.

drwxrwxr-x 5 ... 4096 Abr 17 19:50...

-rw-rw-r— 1 ... 48 Abr 17 20:38 CC3S2.txt

drwxrwxr-x 7 ... 4096 Abr 17 19:57 .git

git-intro$ **cat CC3S2.txt**

Estoy de camino a aprobar el examen CC3S2

/git-intro$

Examinar el estado del repositorio.

Examinar el estado del repositorio usando el **estado de git**. Observar que Git encontró el nuevo archivo en el directorio y sabe que no se rastreó.

/git-intro$ **git status**

En la rama master.

Aún no hay confirmaciones

Archivos sin seguimiento:

(use "git add <file>..." para incluir en lo que se confirmará)

CC3S2.txt

No se agregó nada a confirmar, pero hay archivos sin seguimiento presentes (use "git add" para rastrear)

/git-intro$

Poner en ejecución el archivo.

A continuación, use el comando **git add** para "ejecutar" el archivo CC3S2.txt. La ejecución es una fase intermedia previa a enviar un archivo al repositorio con el comando **git commit**. Este comando crea el contenido del archivo inmediatamente al mismo tiempo de que el comando se haya introducido. Cualquier cambio en el archivo requiere otro comando **git add** antes de confirmar el archivo.

/git-intro$ **git add CC3S2.txt**

Usando el comando **git status** nuevamente, observe los cambios por etapas que se muestran como "new.file:CC3S2.txt".

/git-intro$ **git status**

En la rama master

Aún no hay confirmaciones

Cambios que deben comprometerse:

(use "git rm —cache <file>..."to unstage)

new file: CC3S2.txt

/git-intro$

Confirmando un archivo.

Ahora que ha puesto sus cambios en ejecución, tiene que confirmarlos para que Git sepa que quiere comenzar a rastrear esos cambios. Confirmar su contenido por etapas como una nueva confirmación instantánea mediante el comando **git commit**. El mensaje **-m** permitirá agregar un mensaje explicando los cambios realizados. Observe la combinación de número y letra resaltada en la salida. Este es el ID de confirmación. Cada confirmación se identifica mediante un hash SHA1 único. El ID de confirmación es los primeros 7 caracteres del hash de confirmación completo. Su ID de confirmación será diferente al mostrado.

git-intro$ **git commit -m "Confirmando CC3S2.txt para iniciar seguimiento de cambios"**

[master (root-commit) b510f8e] Confirmando el archivoCC3S2.txt para comenzar el seguimiento de los cambios

1 archivo cambiado, 1 inserción (+)

create mode 100644 CC3S2.txt

/git-intro$

Ver el historial de confirmación.

Utilice el comando **git log** para mostrar todas las confirmaciones en el historial de la rama actual. De forma predeterminada, todas las confirmaciones son realizadas en la rama máster. (Las ramas se discutirán más adelante). La primera línea es el hash de confirmación con el ID de confirmación como los primeros 7 caracteres.

El archivo está comprometido con la rama master. A continuación, se indica su nombre y dirección de correo electrónico, la fecha de la confirmación y el mensaje que ha incluido con la confirmación.

/git-intro$ **git log**

commit b510f8e5f9f63c97432d108a0413567552c07356 (HEAD -> master)

Autor: Usuario test < sample@example.com >

Fecha: Sáb Abr 1 18:03:28 2023 +0000

Confirmar CC3S2.txt para comenzar el seguimiento de los cambios

/git-intro$

Modificación del archivo y seguimiento de los cambios

En esta parte, modificará un archivo, pondrá en ejecución el archivo, confirmará el archivo y verificará los cambios en el repositorio.

Modificar el archivo.

Realice un cambio en CC3S2.txt mediante el comando **echo**. Asegúrese de usar ">>" para agregar el archivo existente. El ">" sobrescribirá el archivo existente. Utilice el comando **cat** para ver el archivo modificado.

/git-intro$ **echo "Estoy empezando a entender Git!" >>CC3S2.txt**

Utilice el comando **cat** para ver el archivo modificado.

/git-intro$ **cat CC3S2.txt**

Estoy de camino a aprobar el examen de ...

¡Estoy empezando a entender a Git!

/git-intro$

Verificar el cambio en el repositorio.

Verificar el cambio en el repositorio usando el comando **git status**.

/git-intro$ **git status**

En la rama master.

Cambios no preparados para la confirmación

(use "git add <file>..."para actualizar lo que se comprometerá)

(use "git restore <file>..."para descartar los cambios en el directorio de trabajo)

modificado: CC3S2.txt

No se agregaron cambios a la confirmación (use "git add" y/o "git commit -a")

/git-intro$

Ejecución del archivo modificado.

El archivo modificado tendrá que ser puesto en escena de nuevo antes de que pueda ser confirmado usando el comando **git add** nuevamente.

/git-intro$ **git add CC3S2.txt**

Confirmar el archivo por ejecución.

Confirmar el archivo por ejecución usando el comando **git commit**. Observe el nuevo ID de confirmación.

/git-intro$ **git commit -m "Agregada línea adicional al archivo"**

[master 9f5c4c5] Se agregó una línea adicional al archivo

1 archivo cambiado, 1 inserción (+)

/git-intro$

Compruebe los cambios en el repositorio.

Utilice el comando **git log** de nuevo para mostrar todas las confirmaciones. Observe que el registro contiene la entrada de confirmación original junto con la entrada para la confirmación que acaba de realizar. La última confirmación se muestra primero. La salida resalta el ID de confirmación (los primeros 7 caracteres del hash SHA1), la fecha/hora de la confirmación y el mensaje de la confirmación para cada entrada.

/git-intro$ **git log**

commit 9f5c4c5d630e88abe2a873fe48144e25ebe7bd6a (HEAD ->master)

Autor: Usuario test< sample@example.com >

Fecha: Sáb Abr 1 19:17:50 2023 +0000

Se agregó una línea adicional al archivo

commit b510f8e5f9f63c97432d108a0413567552c07356

Autor: Usuario test< sample@example.com >

Date: Sab Apr 1 18:03:28 2023 +0000

ConfirmarCC3S2.txt para comenzar el seguimiento de los cambios

/git-intro$

Cuando tiene varias entradas en el registro, puede comparar las dos confirmaciones usando el comando **git diff** agregando el ID de confirmación original primero y el último ID después: **git diff** <commit ID original> <commit ID latest>. Deberá usar sus ID de confirmación. El signo "+" al final, seguido del texto, indica el contenido que se agregó al archivo.

/git-intro$ **git diff b510f8e 9f5c4c5**

diff —git a/CC3S2.txt b/CC3S2.txt

índice 93cd3fb.. 085273f 100644

— a/CC3S2.txt

+++ b/CC3S2.txt

@@ -1 +1,2 @@

Estoy de camino a aprobar el examen CC3S2

¡Estoy empezando a entender a Git!

/git-intro$

Ramas y fusiones.

Cuando se crea un repositorio, los archivos se colocan automáticamente en una rama llamada **master**. Siempre que sea posible, se recomienda utilizar ramas en lugar de actualizar directamente la rama master. El uso de ramas se utiliza para que pueda realizar cambios en otra área sin afectar a la rama master. Esto se hace para ayudar a evitar actualizaciones accidentales que podrían sobrescribir el código existente.

En esta parte, creará una nueva rama, retirará la rama, hará cambios en la rama, ejecutará y confirmará la rama, fusionará los cambios de esa rama en la rama master y luego eliminará la rama.

Crear una nueva rama

Cree una nueva rama llamada **feature** usando el comando **git branch**<branch-name>

/git-intro$ git branch feature

Verificar la rama actual

Utilice el comando **git branch** sin nombre de rama para mostrar todas las ramas de este repositorio. El "*" junto a la rama master indica que se trata de la rama actual, la rama que está actualmente "desprotegida".

/git-intro$ **branch git**

feature

* master

/git-intro$

Verifique la nueva rama.

Utilizar el comando **git checkout** <branch-name> para cambiar a la rama feature.

/git-intro$ **git checkout feature**

Verificar la rama actual.

Verificar que haya cambiado la rama feature usando el comando **git branch**. Tenga en cuenta el "*" junto a la rama feature Esta es ahora la *rama de trabajo*.

/git-intro$ **git branch**

* feature

master

/git-intro$

Añadir una nueva línea de texto al archivo CC3S2.txt, utilizando nuevamente el comando **echo** con los signos ">>".

/git-intro$ **echo "Este texto se agregó originalmente en la rama feature" >>CC3S2.txt**

Comprobar que la línea se ha añadido al archivo mediante el comando **cat**.

/git-intro$ **cat CC3S2.txt**

Estoy de camino a aprobar el examen CC3S2

¡Estoy empezando a entender a Git!

Este texto se agregó originalmente en la rama feature

/git-intro$

Presenta el archivo modificado en la rama feature.

Presentar el archivo actualizado a la rama feature actual.

/git-intro$ **git addCC3S2.txt**

Utilice el comando **git status** y observe que el archivo modificadoCC3S2.txt está en la rama *feature*.

/git-intro$ **git status**

En la rama feature.

Cambios que se deben confirmar:

(use "git restore —staged <file>..."para quitarlo de ejecución)

modificado: CC3S2.txt

/git-intro$

Confirmar el archivo por ejecución en la rama feature

Confirmar el archivo por ejecución usando el comando **git commit**. Observe el nuevo ID de confirmación y su mensaje.

/git-intro$ **git commit -m "Agregado una tercera línea en la rama feature"**

[feature cd828a7] Se agregó una tercera línea en la rama feature

1 archivo cambiado, 1 inserción (+)

/git-intro$

Use el comando **git log** para mostrar todas las confirmaciones, incluida la confirmación que acaba de hacer en la rama *feature*. La confirmación previa se realizó dentro de la rama *master.*

/git-intro$ **git log**

commit cd828a73102cf308981d6290113c358cbd387620 (HEAD -> ...)

Autor: Usuario test< sample@example.com >

Fecha: Sáb Abr 1 22:59:48 2023 +0000

Se agregó una tercera línea en la rama feature

commit 9f5c4c5d630e88abe2a873fe48144e25ebe7bd6a (master)

Autor: Usuario test< sample@example.com >

Fecha: Sáb Abr 1 19:17:50 2023 +0000

Se agregó una línea adicional al archivo

commit b510f8e5f9f63c97432d108a0413567552c07356

Autor: Usuario de ejemplo < sample@example.com >

Date: Sáb Apr 1 18:03:28 2023 +0000

ConfirmarCC3S2.txt para comenzar el seguimiento de los cambios

/git-intro$

Verificación de la rama master.

Cambie a la rama master usando el comando **git checkout** *master* y verifique la rama de trabajo actual usando el comando **git branch**.

/git-intro$ **git checkout master**

Cambiado a la rama 'master'

/git-intro$ **git branch**

Feature

*master

/git-intro$

Combinar el contenido del archivo de feature con la rama master.

Las ramas se utilizan a menudo al implementar nuevas característica o correcciones. Los miembros del equipo pueden enviarlos para su revisión, y luego, una vez verificados, pueden ser arrastrados a la base de código principal: la rama master.

Combine el contenido (conocido como el historial) de la rama feature en la rama master usando el comando **git merge** <branch-name>. El nombre de la rama es la rama de la que se extraen los historiales a la rama actual. La salida muestra que se ha cambiado un archivo con una línea insertada.

/git-intro$ **git merge feature**

Actualización de 9f5c4c5.. cd828a7

Avance rápido.

CC3S2.txt | 1 +

1 archivo cambiado, 1 inserción (+)

/git-intro$

Compruebe el contenido anexado al archivo CC3S2.txt en la rama master mediante el comando **cat**.

/git-intro$ **cat CC3S2.txt**

Estoy de camino a aprobar el examen CC3S2

¡Estoy empezando a entender a Git!

Este texto se agregó originalmente en la rama feature

/git-intro$

Eliminar una rama.

Verifique que la rama **feature** aún esté disponible usando el comando **git branch**.

/git-intro$ **git branch**

feature

* master

/git-intro$

Eliminar la rama **feature** usando el comando <branch-name> **git branch -d**

/git-intro$ **git branch -d feature**

Deleted branch feature (was cd828a7)...

/git-intro$

Verifique que la rama feature ya no esté disponible usando el comando **git branch**.

/git-intro$ **git branch**

* master

/git-intro$

Manejo de conflictos de fusión.

A veces, puede experimentar un conflicto de fusión. Esto es cuando es posible que haya realizado cambios superpuestos en un archivo, y Git no puede combinar automáticamente los cambios.

En esta parte, creará una rama test, modificará su contenido, pondrá en ejecución y confirmará la rama test, cambiará a la rama master, modificará el contenido de nuevo, pondrá en ejecución y comprometerá la rama master, intentará fusionar ramas, localizar y resolver el conflicto, poner en escena y confirmar la rama master de nuevo, y verificará su confirmación.

Crear una nueva prueba de rama.

Crear una nueva **rama test**

/git-intro$ **git branch test**

Revisar la rama test

Cambiar a la rama test

/git-intro$ **git checkout test**
Cambiado a la rama de ‘test'
/git-intro$

Verificar que la rama de trabajo sea la rama **test**.

/git-intro$ **git branch**
master
* test
/git-intro$

Comprobar el contenido actual de CC3S2.txt.

Comprobar el contenido actual del archivo CC3S2.txt. Observar que la primera línea incluya la palabra "Python".

/git-intro$ **cat CC3S2.txt**
Estoy de camino a aprobar el examen CC3S2 Python
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama feature
/git-intro$

Modificar el contenido de CC3S2.txt en la rama test.

Utilizar el comando **sed** para cambiar la palabra "Python " a "Java" en el archivo CC3S2.txt.

/git-intro$ **sed -i 's/Python/Java/'CC3S2.txt**

Comprobar el contenido del CC3S2.txt modificado en la rama test.

Comprobar el cambio en el archivoCC3S2.txt.

/git-intro$ **cat CC3S2.txt**
Estoy en camino a aprobar el examen CC3S2 Java
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama feature
/git-intro$

Ponga en el stage y confirme la rama test

Ponga en el stage y confirme el archivo con un solo comando **git commit -a**. La opción **-a** sólo afecta a los archivos modificados y eliminados. No afecta a los archivos nuevos.

/git-intro$ **git commit -a -m "Cambiar Python a Java"**
[test b6130a6] Change Python a Java
1 file changed, 1 insertion(+), 1 deletion(-)
/git-intro$

Revisar la rama master.

Cambiar a la rama master

/git-intro$ **git checkout master**
Cambiado a la rama 'master'
/git-intro$

Compruebe que la rama master es la rama de trabajo actual.

/git-intro$ **git branch**
* master
test
/git-intro$

Modificar el contenido de CC3S2.txt en la rama master.

Utilice el comando **sed** para cambiar la palabra "Python" a “Java" en el archivo CC3S2.txt.

/git-intro$ **sed -i 's/Python/Java/'CC3S2.txt**

Compruebe el contenido del archivoCC3S2.txt modificado en la rama master.

Verificar el cambio en el archivo.

/git-intro$ **cat CC3S2.txt**
Estoy en camino a aprobar el examen CC3S2 Java
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama feature
/git-intro$

Ponga en ejecución y confirme la rama master.

Ponga en ejecución y confirme el archivo con un solo comando **git commit -a.**

/git-intro$ **git commit -a -m "Cambiado Python a Java"**
[master 72996c0] Cambiado Python a Java
1 file changed, 1 insertion(+), 1 deletion(-)
/git-intro$

Intentar fusionar la rama test en la rama master.

Intentar combinar el historial de rama test en la rama master.

/git-intro$ **git merge test**
Combinación automática deCC3S2.txt

CONFLICT (content): Merge conflict in CC3S2.txt

Error en la fusión automática; corrige los conflictos y, a continuación, confirma el resultado.

/git-intro$

Encuentra el conflicto.

Utilice el comando **git log** para ver las confirmaciones. Observe que la versión HEAD es la rama master. Esto será útil en el siguiente paso.

/git-intro$ **git log**

commit 72996c09fa0ac5dd0b8ab9ec9f8530ae2c5c4eb6 (HEAD -> master)

Autor: Usuario test < sample@example.com >

Fecha: Dom Abr 19 00:36:05 2023 +0000

Cambiado Python a Java

<output omitted>

Usar el comando **cat** para ver los contenidos del archivo CC3S2.txt. El archivo ahora contiene información para ayudarle a encontrar el conflicto. La versión HEAD (rama master) que contiene la palabra "CC3S2" está en conflicto con la versión de la rama test y la palabra "Java".

/git-intro$ **catCC3S2.txt**

<<<<<<< HEAD

Estoy en camino a aprobar el examen CC3S2 Python

=======

Estoy en camino a aprobar el examen CC3S2 Java

>>>>>>> test

¡Estoy empezando a entender a Git!

Este texto se agregó originalmente en la rama feature

/git-intro$

Editar manualmente el archivo CC3S2.txt para eliminar el texto en conflicto.

Utilice el comando **vim** para editar el archivo.

/git-intro$ **vim CC3S2.txt**

Utilice las flechas arriba y abajo para seleccionar la línea de texto adecuada. Presione **dd** (delete) en las siguientes líneas que se resaltan. **dd** eliminará la línea en la que está el cursor.

<<<<<<< HEAD

Estoy en camino a aprobar el examen CC3S2 Python

=======

Estoy en camino a aprobar el examen CC3S2 Java

>>>>>>> test

¡Estoy empezando a entender a Git!

Este texto se agregó originalmente en la rama feature

- Guarde sus cambios en vim presionando **ESC** (la tecla de escape) y luego escribiendo **:** (dos puntos) seguido de **wq** y presione enter.

ESC

:

wq

<Enter or Return>

Leer: https://danielmiessler.com/p/vim/

Comprobar las ediciones de CC3S2.txt en la rama master.

Comprobar los cambios mediante el comando **cat**.

/git-intro$ **cat CC3S2.txt**

Estoy en camino a aprobar el examen CC3S2 Python

¡Estoy empezando a entender a Git!

Este texto se agregó originalmente en la rama feature

/git-intro$

Ponga en ejecución y confirme la rama master

Ponga en ejecución y confirme el archivo con un solo comando **git commit -a.**

/git-intro$ **git add CC3S2.txt**

/git-intro$ **git commit -a -m "Fusionada manualmente desde la rama test"**

[master 22d3da4] Combinado manualmente desde la rama test

/git-intro$

Verificar la confirmación.

Usar el comando **git log** para mirar las confirmaciones. Si es necesario, puede usar **q** para salir de la pantalla del registro de git.

/git-intro$ **git log**

commit 22d3da41e00549ce69dc145a84884af6a1697734 (HEAD -> master)

Fusión: 72996c0 b6130a6

Autor: Usuario test< sample@example.com >

Fecha: Dom Abr 19 01:09:53 2023 +0000

Fusionada manualmente desde la rama test

<output omitted>

Integración de Git con GitHub

Hasta ahora, todos los cambios que ha realizado en su archivo se han almacenado en su máquina local. Git se ejecuta localmente y no requiere ningún servidor de archivos central ni servicio de alojamiento basado en la nube. Git permite a un usuario almacenar y administrar archivos localmente.

Aunque Git es útil para un solo usuario, integrar el repositorio local de Git con un servidor basado en la nube como GitHub es útil cuando se trabaja dentro de un equipo. Cada miembro del equipo

mantiene una copia en el repositorio de su máquina local y actualiza el repositorio central basado en la nube para compartir cualquier cambio.

Hay bastantes servicios populares de Git, incluyendo GitHub, Stash de Atlassian y GitLab. Debido a que es fácilmente accesible, usará GitHub en estos ejemplos.

Crear una cuenta GitHub.

Si no lo ha hecho anteriormente, vaya a github.com y cree una cuenta de GitHub.

Inicie sesión en su cuenta de GitHub - Crear un repositorio.

Inicie sesión en su cuenta de GitHub.

Crear un repositorio.

- Seleccione el botón "**Nuevo repositorio**" o haga clic en el ícono "**+**" en la esquina superior derecha y seleccione "**Nuevo repositorio**".

Cree un repositorio (repositorio de prueba) utilizando la siguiente información:

Nombre del repositorio: CC32S2**-study-team**

Descripción: **Trabajando juntos para aprobar el examen CC3S2**

Público/Privado: **Privado**

- Seleccionar: **Crear repositorio**

Crear un nuevo directorio cc3s2-study-team.

- Si aún no está en el directorio **git-intro**, cámbielo ahora.

~$ **cd/git-intro**

Crear un nuevo directorio llamado cc3s2-study-team. El directorio no tiene que coincidir con el nombre como repositorio.

/git-intro$ **mkdir cc3s2-study-team**

Cambie el directorio a cc3s2-study-team.

Utilice el comando **cd** para cambiar los directorios a **cc3s2-study-team**.

/git-intro$ **cd cc3s2-study-team**
/git-intro/cc3s2-study-team$

Copie el archivo trabajado en la actividad

Utilice el comando **cp** para copiar el archivo CC3S2.txt del directorio principal **git-intro** al subdirectorio **cc3s2-study-team**. Los dos puntos y una barra diagonal anterior al nombre del archivo indican el directorio principal. El espacio y el punto que siguen al nombre del archivo indica que se debe copiar el archivo en el directorio actual con el mismo nombre de archivo.

/git-intro/cc3s2-study-team$ **cp../CC3S2.txt.**

- Verifique que el archivo se haya copiado con el comando **ls** y el contenido del archivo con el comando **cat**.

/git-intro/cc3s2-study-team$ **ls**

CC3S2.txt

/git-intro/cc3s2-study-team$ **cat CC3S2.txt**

Estoy en camino a aprobar el examen C3S2 Python

¡Estoy empezando a entender a Git!

Este texto se agregó originalmente en la rama feature

/git-intro/cc3s2-study-team$

Inicializar un nuevo repositorio de Git.

Utilice el comando **git init** para inicializar el directorio actual (cc3s2-study-team) como repositorio de Git. El mensaje que se muestra indica que ha creado un repositorio local dentro del proyecto contenido en el directorio oculto **.git**. Aquí es donde se encuentra todo el historial de cambios.

/git-intro/cc3s2-study-team$ **git init**

Repositorio Git vacío inicializado en /home/.../git-intro/cc3s2-study-team/.git/

/git-intro/cc3s2-study-team$

- A continuación, compruebe sus variables globales de git con el comando **git config —list**.

/git-intro/cc3s2-study-team$ **git config —list**

User.name=...

user.email=...

core.repositoryformatversion=0

core.filemode=true

core.bare=false

core.logallrefupdates=true

/git-intro/cc3s2-study-team$

- Si las variables user.name y user.email no coinciden con sus credenciales de GitHub, cámbielas ahora.

~$ **git config --global user.name** *GitHub Username***"**

~$ **git config --global user.email** *GitHub-email-address*

Apunte el repositorio de Git al repositorio de GitHub.

- Utilice el comando **git remote add** para agregar una URL de Git como un alias remoto. El valor "origen" apunta al repositorio recién creado en GitHub. Utilice su nombre de usuario de GitHub en la ruta URL para *github-username*.

Nota: Su nombre de usuario distingue entre mayúsculas y minúsculas.

/git-intro/cc3s2-study-team$ **git remote add origin https://github.com/*github-username*/cc3s2-study-team.git**

- Verifique que el **remote** se esté ejecutando en github.com.

/git-intro/cc3s2-study-team$ **git remote —verbose**

origen https://github.com/username/cc3s2-study-team.git ...

origen https://github.com/username/cc3s2-study-team.git ...

/git-intro/cc3s2-study-team$

- Ver el **registro de git**. El error indica que no hay confirmaciones.

/git-intro/cc3s2-study-team$ **git log**

Fatal: Su rama actual 'master ' no tiene ninguna confirmación todavía

Ponga en el stage y confirme el archivo CC3S2.txt.

Utilice el comando **git add** para poner en el stage el archivo CC3S2.txt.

/git-intro/cc3s2-study-team$ **git addCC3S2.txt**

Utilice el comando **git commit** para confirmar el archivoCC3S2.txt.

/git-intro/cc3s2-study-team$ **git commit -m "Agrega CC3S2.txt file al cc3s2-study-team"**

[master (root-commit) c60635f] Agrega CC3S2.txt file al cc3s2-study-team

1 archivo cambiado, 3 insersiones(+)

create mode 100644CC3S2.txt

/git-intro/cc3s2-study-team$

Verificar la confirmación.

- Usar el comando **git log** para mirar las confirmaciones.

/git-intro/cc3s2-study-team$ **git log**

commit c60635fe4a1f85667641afb9373e7f49a287bdd6 (HEAD -> master)

Autor: nombre de usuario < user@example.com >

Fecha: Lun Abr 2 02:48:21 2020 +0000

Agregar archivo CC3S2.txt al cc3s2-study-team

/git-intro/cc3s2-study-team$

- Utilice el comando **git status** para ver la información de estado. La frase "working tree clean" significa que Git ha comparado tu lista de archivos con lo que le has dicho a Git, y es una pizarra limpia sin nada nuevo que informar.

/git-intro/cc3s2-study-team$ **git status**

rama master.

Nada que confirmar, working tree clean

/git-intro/cc3s2-study-team$

Enviar (push) el archivo de Git a GitHub.

Utilice el comando **git push origin** para enviar (push) el archivo a su repositorio de GitHub. Se le pedirá un nombre de usuario y una contraseña (token), que será la que utilizó para crear su cuenta de GitHub.

/git-intro/cc3s2-study-team$ **git push origin**

Nombre de usuario para 'https://github.com': ***nombre de usuario***

Token para 'https://username@github.com':token

Objetos enumerados: 3, hecho.

Contando objetos 100% (3/3), hecho.

Compresión Delta usando hasta 2 hilos.

Compresión de objetos: 100% (2/2), hecho.

Escribiendo objetos: 100% (2/2), hecho.

Total 3 (delta 0), reused 0 (delta 0)

A https://github.com/username/cc3s2-study-team.git

* [nueva rama] master -> master

/git-intro/cc3s2-study-team$

Nota: Si, después de introducir su nombre de usuario y contraseña, obtiene un error fatal que indica que no se encuentra el repositorio, lo más probable es que haya enviado una URL incorrecta. Deberá revertir su comando **git add** con el comando **git remote rm origin**.

Verifique el archivo en GitHub.

Vaya a su cuenta de GitHub y en "Repositorios" seleccione nombre de **usuario/cc3s2-study-team**.

Debería ver que el archivo CC3S2.txt se ha agregado a este repositorio de GitHub. Haga clic en el archivo para ver el contenido.

Tareas adicional: Revisa el siguiente tutorial https://www.datacamp.com/es/tutorial/git-push-pull y con tu equipo de trabajo realicen experimentos de práctica para futuras evaluaciones.