Laboratorio - Control de versiones de software con Git

Objetivos

Parte 1: Iniciar la Máquina Virtual (VM) DEVASC

Parte 2: Inicializar Git

Parte 3: Preparar y confirmar un archivo en el repositorio de Git

Parte 4: Gestión del archivo y seguimiento de los cambios

Parte 5: Sucursales y fusión

Parte 6: Gestión de conflictos de combinación

Parte 7: Integración de Git con GitHub

Aspectos básicos/Situación.

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

Recursos necesarios

- Una computadora con el sistema operativo de su elección.
- Virtual Box o VMWare.
- Máquina virtual (Virtual Machine) DEVASC.

Instrucciones

Parte 1: Iniciar la Máquina virtual (Virtual Machine) de DEVASC

Si no ha completado el laboratorio **- Instale el Entorno de Laboratorio de la Máquina Virtual**, hágalo ahora. Si se ha completado ya, inicie la máquina virtual DEVASC.

Parte 2: Inicializando Git

En esta parte, inicializará un repositorio de Git.

Paso 1: Abra una terminal en el laboratorio de la máquina virtual de DEVASC.

Doble click al icono del emulador de terminal en el escritorio.

Paso 2: Inicializar un repositorio de Git.

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

```
devasc@labvm:~$ ls
Descargar plantillas de música de públicas
Documentos de laboratorio, Imágenes instantáneas y videos.
devasc@labvm:~$
```

b. A continuación, configure la información de usuario que se utilizará para este repositorio local. Esto asociará su información con el trabajo que contribuye a un repositorio local. Utilice su nombre en lugar de "Usuario de prueba" 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 en la Parte 7, usará el nombre de usuario de su cuenta de GitHub. Si lo desea, puede usar su nombre de usuario de GitHub ahora.

```
devasc @labvm: ~$ git config –user.name global "SampleUser"
devasc @labvm: ~$ git config –global user.email sample@example.com
```

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

```
devasc @labvm: ~$ git config –list
User.name=SampleUser
user.email=sample@example.com
devasc@labvm:~$
```

d. Utilice el comando **cd** para navegar a la carpeta **devnet-src** :

```
devasc @labvm: ~$ cd labs/devnet-src/
devasc @labvm: ~/labs/devnet-src$
```

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

```
devasc @labvm: ~/labs/devnet-src$ mkdir git-intro
devasc @labvm: ~/labs/devnet-src$ cd git-intro
devasc @labvm: ~/labs/devnet-src/git-intro$
```

f. 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**.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git init
Repositorio Git vacío inicializado en /home/devasc/labs/devnet-src/git-intro/.git/
devasc @labvm: ~/labs/devnet-src/git-intro$ ls -a
.  ..  .git
devasc@labvm:~/labs/devnet-src/git-intro$
```

g. 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.

Este mensaje le dice:

- Estas en una rama maestra. (Los niveles se discuten más adelante en este laboratorio)
- 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git status
En la rama maestra

Aún no hay confirmaciones

Nada que confirmar (create/copy files and use "git add" to track)
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Parte 3: 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.

Paso 1: Crear un archivo.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ echo "Estoy en camino para pasar el examen DEVASC de Cisco" > DEVASC.txt
devasc@labvm:~/labs/devnet-src/git-intro$
```

b. Utilzar 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 **DEVASC.txt**.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ ls -la
total 16
drwxrwxr-x 3 devasc devasc 4096 Abr 17 20:38.
drwxrwxr-x 5 devasc devasc 4096 Abr 17 19:50..
-rw-rw-r— 1 devasc devasc 48 abr 17 20:38 DEVASC.txt
drwxrwxr-x 7 devasc devasc 4096 Abr 17 19:57 .git
evasc @labvm: ~/src/git-intro$ cat DEVASC.txt
Estoy de camino a aprobar el examen DEVASC de Cisco
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 2: 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git status
En la rama maestra.

Aún no hay confirmaciones

Archivos sin seguimiento:
(use "git add <file>..." para incluir en lo que se confirmará)
      DEVASC.txt

No se agregó nada a confirmar, pero hay archivos sin seguimiento presentes (use "git add" para rastrear)
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 3: Ponga en ejecución el archivo.

a. A continuación, use el comando **git add** para "ejecutar" el archivo **DEVASC.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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git add DEVASC.txt
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git status
```

```
En la rama maestra

Aún no hay confirmaciones

Cambios que deben comprometerse:
  (use "git rm —cache <file>..."to unstage)
        new file: DEVASC.txt

devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 4: 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.

```
devasc@labvm:~/labs/devnet-src/git-intro$ git commit -m "Committing DEVASC.txt to begin tracking changes"
[master (root-commit) b510f8e] Confirmando el archivo DEVASC.txt para comenzar el seguimiento de los cambios
1 archivo cambiado, 1 inserción (+)
create mode 100644 DEVASC.txt
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 5: 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 principal. (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 maestra. 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git log
commit b510f8e5f9f63c97432d108a0413567552c07356 (HEAD -> master)
Autor: Usuario de prueba < sample@example.com >
Fecha: Sáb Abr 18 18:03:28 2020 +0000

Confirmar DEVASC.txt para comenzar el seguimiento de los cambios
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Parte 4: 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.

Paso 1: Modificar el archivo.

a. Realice un cambio en DEVASC.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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$echo "Estoy empezando a entender Git!" >> DEVASC.txt
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy de camino a aprobar el examen DEVASC de Cisco
¡Estoy empezando a entender a Git!
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 2: Verificar el cambio en el repositorio.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git status
En la rama maestra.
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: DEVASC.txt

No se agregaron cambios a la confirmación (use "git add" y/o "git commit -a")
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 3: 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git add DEVASC.txt
```

Paso 4: Confirmar el archivo por ejecución.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git commit -m "Agregada línea adicional al archivo"
[master 9f5c4c5] Se agregó una línea adicional al archivo
1 archivo cambiado, 1 inserción (+)
devasc@labvm:~/labs/devnet-src/git-intro$
```

Paso 5: Compruebe los cambios en el repositorio.

a. 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git log
commit 9f5c4c5d630e88abe2a873fe48144e25ebe7bd6a (HEAD ->master)
Autor: Usuario de prueba< sample@example.com >
Fecha: Sáb Abr 18 19:17:50 2020 +0000

Se agregó una línea adicional al archivo

commit b510f8e5f9f63c97432d108a0413567552c07356
Autor: Usuario de prueba< sample@example.com >
Date:   Sat Apr 18 18:03:28 2020 +0000
```

```
Confirmar DEVASC.txt para comenzar el seguimiento de los cambios
devasc @labvm: ~/labs/devnet-src/git-intro$
```

b. 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git diff b510f8e 9f5c4c5
diff –git a/DEVASC.txt b/DEVASC.txt
índice 93cd3fb.. 085273f 100644
– a/DEVASC.txt
+++ b/DEVASC.txt
@@ -1 +1,2 @@
Estoy de camino a aprobar el examen DEVASC de Cisco
¡Estoy empezando a entender a Git!
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Parte 5: 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 maestra. La derivación se utiliza para que pueda realizar cambios en otra área sin afectar a la rama maestra. 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 principal y luego eliminará la rama.

Paso 1: Crear una nueva rama

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ característica de la rama git
```

Paso 2: 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 principal indica que se trata de la rama actual, la rama que está actualmente "desprotegida".

```
devasc @labvm: ~/labs/devnet-src/git-intro$ branch git
  feature
* master
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 3: Verifique la nueva rama.

Utilizar el comando **git checkout** <branch-name> para cambiar las *características* de la rama.

```
devasc@labvm:~/labs/devnet-src/git-intro$ git checkout feature
```

Paso 4: Verificar la rama actual.

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

```
devasc@labvm:~/labs/devnet-src/git-intro$ git branch
* feature
master
```

```
devasc @labvm: ~/labs/devnet-src/git-intro$
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ echo "Este texto se agregó originalmente en la rama de características" >> DEVASC.txt
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy de camino a aprobar el examen DEVASC de Cisco
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 5: Presente el archivo modificado en la rama de características.

a. Presentar el archivo actualizado a la rama de *características* actual.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git add DEVASC.txt
```

b. Utilice el comando **git status** y observe que el archivo modificado **DEVASC.txt** está en la rama *características*.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git status
En la rama característica.
Cambios que se deben confirmar:
(use "git restore –staged <file>..."para quitarlo de ejecución)
       modificado: DEVASC.txt

devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 6: Confirmar el archivo por ejecución en la rama de características

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git commit -m "Agregado una tercera línea en la rama de características"
[característica cd828a7] Se agregó una tercera línea en la rama de características
1 archivo cambiado, 1 inserción (+)
devasc@labvm:~/labs/devnet-src/git-intro$
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git log
commit cd828a73102cf308981d6290113c358cbd387620 (HEAD -> función)
Autor: Usuario de prueba< sample@example.com >
Fecha: Sáb Abr 18 22:59:48 2020 +0000

Se agregó una tercera línea en la rama de entidad

commit 9f5c4c5d630e88abe2a873fe48144e25ebe7bd6a (master)
Autor: Usuario de prueba< sample@example.com >
Fecha: Sáb Abr 18 19:17:50 2020 +0000
```

```
Se agregó una línea adicional al archivo

commit b510f8e5f9f63c97432d108a0413567552c07356
Autor: Usuario de ejemplo < sample@example.com >
Date:   Sat Apr 18 18:03:28 2020 +0000

Confirmar DEVASC.txt para comenzar el seguimiento de los cambios
devasc@labvm:~/labs/devnet-src/git-intro$
```

Paso 7: Verificación de la rama maestra.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git checkout master
Cambiado a la rama 'maestra'
devasc@labvm:~/labs/devnet-src/git-intro$ git branch
Características
*master
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 8: Combinar el contenido del archivo de las características con la rama maestra.

a. Las ramas se utilizan a menudo al implementar nuevas características 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 maestra.

   Combine el contenido (conocido como el historial) de la rama de características en la rama maestra usando el comando <branch-name> X **git merge**. 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.

```
devasc@labvm:~/labs/devnet-src/git-intro$ git merge feature
Actualización de 9f5c4c5.. cd828a7
Avance rápido.
 DEVASC.txt | 1 +
1 archivo cambiado, 1 inserción (+)
devasc @labvm: ~/labs/devnet-src/git-intro$
```

b. Compruebe el contenido anexado al archivo DEVASC.txt en la rama maestra mediante el comando **cat**.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy de camino a aprobar el examen DEVASC de Cisco
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 9: Eliminar una rama.

a. Verifique que la rama de **características** aún esté disponible usando el comando **git branch**.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git branch
feature
* master
devasc@labvm:~/labs/devnet-src/git-intro$
```

b. Eliminar la rama de **características** usando el comando <branch-name> **git branch -d**. .

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git branch -d feature
Deleted branch feature (was cd828a7)..
devasc@labvm:~/labs/devnet-src/git-intro$
```

c. Verifique que la rama de características ya no esté disponible usando el comando **git branch**.

```
devasc@labvm:~/labs/devnet-src/git-intro$ git branch
* master
devasc@labvm:~/labs/devnet-src/git-intro$
```

Parte 6: 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 de prueba, modificará su contenido, pondrá en ejecución y confirmará la rama de prueba, cambiará a la rama maestra, modificará el contenido de nuevo, pondrá en ejecución y comprometerá la rama maestra, intentará fusionar ramas, localizar y resolver el conflicto, poner en escena y confirmar la rama maestra de nuevo, y verificará su confirmación.

Paso 1: Crear una nueva prueba de rama.

Crear una nueva **rama de prueba**

```
devasc@labvm:~/labs/devnet-src/git-intro$ git branch test
```

Paso 2: Revisar la rama de prueba

a. Cambiar a la rama de **prueba**

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git checkout test
Cambiado a la rama de ‘prueba'
devasc @labvm: ~/labs/devnet-src/git-intro$
```

b. Verificar que la rama de trabajo sea la rama **de prueba**.

```
devasc@labvm:~/labs/devnet-src/git-intro$ git branch
master
* test
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 3: Comprobar el contenido actual de DEVASC.txt.

Comprobar el contenido actual del archivo **DEVASC.txt**. Observar que la primera línea incluya la palabra "Cisco".

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy de camino a aprobar el examen DEVASC de Cisco
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc@labvm:~/labs/devnet-src/git-intro$
```

Paso 4: Modificar el contenido de DEVASC.txt en la rama de prueba.

Utilizar el comando **sed** para cambiar la palabra "Cisco" a "NetACad" en el archivo **DEVASC.txt**.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ sed -i 's/cisco/NetACad/'
DEVASC.txt
```

Paso 5: Comprobar el contenido del DEVASC.txt modificado en la rama de prueba.

Comprobar el cambio en el archivo **DEVASC.txt**.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy en camino a aprobar el examen DEVASC de NetACad
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 6: Ponga en escena y confirme la rama de prueba

Ponga en escena 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git commit -a -m "Cambiar Cisco a
NetACad"
[test b6130a6] Change Cisco a NetACad
1 file changed, 1 insertion(+), 1 deletion(-)
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 7: Revisar la rama principal.

a. Cambiar a la rama **principal**

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git checkout master
Cambiado a la rama 'maestra'
devasc @labvm: ~/labs/devnet-src/git-intro$
```

b. Compruebe que la rama principal es la rama de trabajo actual.

```
devasc@labvm:~/labs/devnet-src/git-intro$ git branch
* master
test
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 8: Modificar el contenido de DEVASC.txt en la rama principal.

Utilice el comando **sed** para cambiar la palabra "Cisco" a "DevNet" en el archivo DEVASC.txt.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ sed -i 's/cisco/devnet/'
DEVASC.txt
```

Paso 9: Compruebe el contenido del archivo DEVASC.txt modificado en la rama principal.

Verificar el cambio en el archivo.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy en camino a aprobar el examen DevNet DEVASC
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc@labvm:~/labs/devnet-src/git-intro$
```

Paso 10: Ponga en ejecución y confirme la rama de maestra.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git commit -a -m "Cambiado Cisco a
DevNet"
```

```
[master 72996c0] Cambiado Cisco a DevNet
1 file changed, 1 insertion(+), 1 deletion(-)
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 11: Intentar fusionar la rama de prueba en la rama maestra.

Intentar combinar el historial de rama de prueba en la rama maestra.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git merge test
Combinación automática de DEVASC.txt
CONFLICT (content): Merge conflict in DEVASC.txt
Error en la fusión automática; corrige los conflictos y, a continuación, confirma el
resultado.
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 12: Encuentra el conflicto.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git log
commit 72996c09fa0ac5dd0b8ab9ec9f8530ae2c5c4eb6 (HEAD -> master)
Autor: Usuario de prueba < sample@example.com >
Fecha: Dom Abr 19 00:36:05 2020 +0000

    Cambiado Cisco a DevNet

<output omitted>
```

b. Usar el comando **cat** para ver los contenidos del archivo DEVASC.txt. El archivo ahora contiene información para ayudarle a encontrar el conflicto. La versión HEAD (rama maestra) que contiene la palabra "DevNet" está en conflicto con la versión de la rama de prueba y la palabra "NetACad".

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
<<<<<<< HEAD
Estoy en camino a aprobar el examen de DevNet DEVASC
=======
Estoy en camino a aprobar el examen DEVASC de NetACad
>>>>>>> test
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 13: Editar manualmente el archivo DEVASC.txt para eliminar el texto en conflicto.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ vim DEVASC.txt
```

b. 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 de DevNet DEVASC
=======
Estoy en camino a aprobar el examen de DEVASC de NetACad
>>>>>>> test
```

```
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
```

c. 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>
```

Paso 14: Comprobar las ediciones de DEVASC.txt en la rama principal.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cat DEVASC.txt
Estoy en camino a aprobar el examen DevNet DEVASC
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc@labvm:~/labs/devnet-src/git-intro$
```

Paso 15: Ponga en ejecución y confirme la rama de maestra

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git add DEVASC.txt
devasc @labvm: ~/labs/devnet-src/git-intro$ git commit -a -m "Fusionada
manualmente desde la rama de prueba"
[master 22d3da4] Combinado manualmente desde la rama de prueba
devasc @labvm: ~/labs/devnet-src/git-intro$
```

Paso 16: 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro$ git log
commit 22d3da41e00549ce69dc145a84884af6a1697734 (HEAD -> master)
Fusión: 72996c0 b6130a6
Autor: Usuario de prueba< sample@example.com >
Fecha: Dom Abr 19 01:09:53 2020 +0000

    Fusionada manualmente desde la rama de prueba
<output omitted>
```

Parte 7: 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.

Paso 1: Crear una cuenta GitHub.

Si no lo ha hecho anteriormente, vaya a github.com y cree una cuenta de GitHub. Si tiene una cuenta de GitHub, vaya al paso 2.

Paso 2: Inicie sesión en su cuenta de GitHub - Crear un repositorio.

Inicie sesión en su cuenta de GitHub.

Paso 3: Crear un repositorio.

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

b. Cree un repositorio utilizando la siguiente información:

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

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

Público/Privado: **Privado**

c. Seleccionar: **Crear repositorio**

Paso 4: Crear un nuevo directorio devasc-study-team.

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

```
devasc @labvm: ~$ cd ~/labs/devnet-src/git-intro
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ mkdir devasc-study-team
```

Paso 5: Cambie el directorio a devasc-study-team.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro$ cd devasc-study-team
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$
```

Paso 6: Copie el archivo DEVASC.

a. Utilice el comando **cp** para copiar el **DEVASC.txt** del directorio principal **git-intro** al subdirectorio **devasc-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.

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ cp../DEVASC.txt.
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ ls
DEVASC.txt
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ cat DEVASC.txt
Estoy en camino a aprobar el examen DevNet DEVASC
¡Estoy empezando a entender a Git!
Este texto se agregó originalmente en la rama de características
devasc@labvm:~/labs/devnet-src/git-intro/devasc-study-team$
```

Paso 7: Inicializar un nuevo repositorio de Git.

a. Utilice el comando **git init** para inicializar el directorio actual (devasc-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.

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git init
Repositorio Git vacío inicializado en /home/devasc/src/git-intro/devasc-study-team/.git/
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git config –list
User.name=SampleUser
user.email=sample@example.com
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$
```

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

```
devasc @labvm: ~$ git config --global user.name GitHub Username"
devasc @labvm: ~$ git config --global user.email GitHub-email-address
```

Paso 8: Apunte el repositorio de Git al repositorio de GitHub.

a. 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git remote add origin https://github.com/github-username/devasc-study-team.git
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git remote –verbose
origen https://github.com/username/devasc-study-team.git (buscar)
origen https://github.com/username/devasc-study-team.git (push)
devasc@labvm:~/labs/devnet-src/git-intro/devasc-study-team$
```

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

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git log
Fatal: Su rama actual 'master o principal' no tiene ninguna confirmación todavía
```

Paso 9: Ponga en escena y confirme el archivo DEVASC.txt.

a. Utilice el comando **git add** para poner en escena el archivo DEVASC.txt.

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git add DEVASC.txt
```

b. Utilice el comando **git commit** para confirmar el archivo DEVASC.txt.

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git commit -m "Add DEVASC.txt file to devasc-study-team"
```

```
[master (root-commit) c60635f] Add DEVASC.txt file to devasc-study-team
1 archivo cambiado, 3 insersiones(+)
create mode 100644 DEVASC.txt
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$
```

Paso 10: Verificar la confirmación.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git log
commit c60635fe4a1f85667641afb9373e7f49a287bdd6 (HEAD -> master)
Autor: nombre de usuario < user@example.com >
Fecha: Lun Abr 20 02:48:21 2020 +0000

    Agregar archivo DEVASC.txt a devasc-study-team
devasc@labvm:~/labs/devnet-src/git-intro/devasc-study-team$
```

b. 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.

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git status
rama maestra.
Nada que confirmar, working tree clean
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$
```

Paso 11: Enviar (push) el archivo de Git a GitHub.

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

```
devasc @labvm: ~/labs/devnet-src/git-intro/devasc-study-team$ git push origin master
Nombre de usuario para 'https://github.com': nombre de usuario
Contraseña para 'https://username@github.com':contraseña
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/devasc-study-team.git
* [nueva rama] maestro -> maestro
devasc@labvm:~/labs/devnet-src/git-intro/devasc-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**.

Paso 12: Verifique el archivo en GitHub.

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

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