Práctico 2: Git y GitHub

Alumno: Alejandro Lagos

Link actividades 2 y 3:

2) https://github.com/Alejandrovans/repositorios_git.git
3) https://github.com/Alejandrovans/conflict-exercise.git

Actividades

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

- ¿Qué es GitHub?

  Es una plataforma donde los desarrolladores pueden guardar, organizar y trabajar en sus proyectos de software. Es gratis y de código abierto. Facilita el trabajo en equipo, compartir código y colaborar de manera eficiente.

- ¿Cómo crear un repositorio en GitHub?

  Primero debemos ingresar a la página de github, luego crear una cuenta; podemos personalizar nuestro perfil y empezar a crear nuestros repositorios.

  En la parte izquiera de la pantalla presionamos en “New”. Nos aparecerá la opción de Crear un nuevo repositorio, le ponemos nombre y elegimos si queremos que sea publico o privado. Presionamos en “Create repository” y ya tendremos nuestro repositorio creado.

- ¿Cómo crear una rama en Git?

  Para crear una rama debemos incializar el comando “git branch”. Siempre comenzaremos en la rama principal de nuestro archivo, denominado MAIN o MASTER.

- ¿Cómo cambiar a una rama en Git?

  Para cambiar de una rama a otra usamos el comando “git checkout” seguido del nombre de la rama o ejercicio al que queremos ir, por ejemplo, si estamos en la rama MASTER y queremos cambiar a la rama OtraRama tendríamos que escribir “git checkout OtraRama”.

- ¿Cómo fusionar ramas en Git?

Para fusionar ramas debemos usar el comando “git merge”. Si queremos fusionar la rama Master con la rama OtraRama; En la terminal de nuestra rama MASTER debemos escribir “git merge OtraRama”

- ¿Cómo crear un commit en Git?

  Cuando realicemos los cambios en nuestro proyecto, usamos el comando git add para agregar los cambios, después usamos en comando “git commit” y agregamos un mensaje comentando los cambios. Por ej: git commit –m “se realizacon cambios en línea 3”.

- ¿Cómo enviar un commit a GitHub?

  Para enviar un commit debemos usar el comando “git push”. Si es la primera vez, debemos poner “git push –u origin master” (o main, en su defecto).

- ¿Qué es un repositorio remoto?

  Son versiones de nuestro proyecto que están hospedadas en una nube o host. Podemos subirlos con permiso solo de lectura o lectura y escritura para poder colaborar con otros desarrolladores, y enviar o traer datos para mejorar este proyecto.

- ¿Cómo agregar un repositorio remoto a Git?

  Primero debemos iniciar nuestro git (git init), luego agregar el remoto desde su origen junto con la url del repositorio: git remote add origin https://github.com/Alejandrovans/inicial_git.git

  Para verificar que ser agregego usamos el comando “git remote -v”

- ¿Cómo empujar cambios a un repositorio remoto?

  Para empujar cambios a un repositorio usamos el comando “git push origin” seguido del nombre de la rama que queremos empujar. EJ: git push origin RamaPrincipal

- ¿Cómo tirar de cambios de un repositorio remoto?

  Para tirar cambios de repositorio usamos el comando “git pull origin” y el nombre de la rama a tirar. Ej: git pull origin RamaSecundaria

- ¿Qué es un fork de repositorio?

  Es una copia de un repositorio, que nos permite modificar o experimentar con un proyecto original sin modificarlo directamente, nos sirve para proponer cambios en otros proyectos o probar otras opciones de código.

- ¿Cómo crear un fork de un repositorio?

  En GitHub elegimos el reposito del que queremos crear nuestro fork, y presionamos donde dice “Fork”. Seleccionamos el perfil en el que vamos a alojar nuestra copia y GitHub creara una copia en nuestra cuenta con su respectiva URL.

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

  En nuestra pantalla del repositorio, presionamos en PULL REQUEST, ahí elegimos “new pull request”, nos aparecerá un resumen de los cambios del repositorio. Elegimos “créate new pull request” y luego comentamos los cambios que hicimos.

- ¿Cómo aceptar una solicitud de extracción?

  En la pestaña PULL REQUEST revisamos los “filed changed” revisamos que no haya confilctos con nuestro repositorio y hacemos un “merge commit” seleccionando MERGE PULL REQUEST y CREAT A MERGE COMMIT.

- ¿Qué es un etiqueta en Git?

  Es una marca en un algún punto especifico del historial de commits para indicar versiones, que a diferencia de las branch estas no se mueven.

- ¿Cómo crear una etiqueta en Git?

  Hay dos opciones: ligera y anotada.

  Para ligera usamos “git tag” y la versión del software

  Para la anotada podemos anotar la versión (-a) y un mensaje (-m)

  Ej: git tag v1.0.2 // git tag –a v1.0.4 –m “ultima version estable”

- ¿Cómo enviar una etiqueta a GitHub?

  Creamos una etiqueta en el repositorio local, ej: (git tag v1.2) y La empujamos al repositorio remoto (git push origin v1.2)

- ¿Qué es un historial de Git?

  Es una secuencia donde vemos todos los cambios realizados en el repostiorio de git, donde cada cambio se guarda como commit, donde estos tienen la información sobre el proyecto y sus cambios en cada línea de código especifica.

- ¿Cómo ver el historial de Git?

  Para ver el historial básico usamos el comando “git log”.

- ¿Cómo buscar en el historial de Git?

Para buscar en el historial tenemos varios comandos que podemos usar dependiendo lo que queremos buscar algunos ejemplos son: “git log –oneline” donde vemos un historial compacto (una línea por commit), “git log --stat” vemos los cambios en los achivos, “git log -p” muestra los cambios línea por línea en cada commit.

- ¿Cómo borrar el historial de Git?

  El comando para borrar el historial es “git reset”

- ¿Qué es un repositorio privado en GitHub?

  Es un repositorio al cual solo tendrán acceso los usuarios autorizados por el creador del mismo. Sirve para proyectos en desarrollo o información sensible.

- ¿Cómo crear un repositorio privado en GitHub?

  En nuestra cuenta de GitHub donde creamos nuestros repositorios, seleccionamos “New Repository” y antes de crearlo seleccionamos la casilla PRIVATE y clickeamos en “Create repository”.

- ¿Cómo invitar a alguien a un repositorio privado en GitHub?

  Clickeamos “settings”, vamos donde dice “collaborators” y elegimos “add people”: ingresamos el nombre de usuario de quien queremos invitar y seleccionamos el nivel de acceso que queremos darle al colaborador; presionamos “add” y se envía la invitación.

- ¿Qué es un repositorio público en GitHub?

  Es un repositorio con acceso a cualquier usuario de GitHub par que pueda clonar, modificar o sugerir cambios al proyecto.

- ¿Cómo crear un repositorio público en GitHub?

2. En nuestra cuenta de GitHub donde creamos nuestros repositorios, seleccionamos “New Repository” y antes de crearlo seleccionamos la casilla PUBLIC y clickeamos en “Create repository”.

- ¿Cómo compartir un repositorio público en GitHub?

  La forma más sencilla y rápida es copiar la URL de nuestro repositorio y enviársela a quien queramos. En GitHub abrimos nuestro repositorio y copiamos la URL haciendo click en el botón “<>Code” a la derecha de la pantalla.

2) Realizar la siguiente actividad:

- Crear un repositorio.
    - Dale un nombre al repositorio. ○ Elije el repositorio sea público.
    - Inicializa el repositorio con un archivo.
- Agregando un Archivo
    - Crea un archivo simple, por ejemplo, "mi-archivo.txt".
    - Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
    - Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).
- Creando Branchs
    - Crear una Branch ○ Realizar cambios o agregar un archivo ○ Subir la Branch

3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

- Ve a GitHub e inicia sesión en tu cuenta.
- Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- Opcionalmente, añade una descripción.
- Marca la opción "Initialize this repository with a README".
- Haz clic en "Create repository".

Paso 2: Clonar el repositorio a tu máquina local

- Copia la URL del repositorio (usualmente algo como https://github.com/tuusuario/conflict-exercise.git).
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio usando el comando:

git clone https://github.com/tuusuario/conflict-exercise.git

- Entra en el directorio del repositorio: cd conflict-exercise

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch:

git checkout -b feature-branch

- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:

Este es un cambio en la feature branch.

- Guarda los cambios y haz un commit: git add README.md git commit -m "Added a line in feature-branch"

Paso 4: Volver a la rama principal y editar el mismo archivo

- Cambia de vuelta a la rama principal (main):

git checkout main

- Edita el archivo README.md de nuevo, añadiendo una línea diferente:

Este es un cambio en la main branch.

- Guarda los cambios y haz un commit: git add README.md git commit -m "Added a line in main branch"

Paso 5: Hacer un merge y generar un conflicto

- Intenta hacer un merge de la feature-branch en la rama main:

git merge feature-branch

- Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

Paso 6: Resolver el conflicto

- Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

<<<<<<< HEAD

Este es un cambio en la main branch.

=======

Este es un cambio en la feature branch.

>>>>>>> feature-branch

- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).
- Añade el archivo resuelto y completa el merge:

git add README.md git commit -m

"Resolved merge conflict"

Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub:

git push origin main

- También sube la feature-branch si deseas:

git push origin feature-branch

Paso 8: Verificar en GitHub

- Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
- Puedes revisar el historial de commits para ver el conflicto y su resolución.