Práctico 2: Git y GitHub

Alumno: Enzo Chavez

Actividades

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

• ¿Qué es GitHub?

GitHub es una comunidad donde se puede compartir repositorios de forma pública o privada. Allí podemos, aparte de cargar nuestros repositorios, ver los de otras personas, guardarlos o clonarlos.

• ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio nuevo en GitHub nos posicionamos en nuestro perfil de GitHub y desde allí hacer click en repositorios, luego en “New” y ahí cargamos el nombre del repositorio, con que perfil lo vamos a crear, la descripción y si va a ser público o privado y hacemos click en “Create repository”.

• ¿Cómo crear una rama en Git?

Para crear una rama en Git es necesario posicionarnos en la terminal y desde ahí escribimos “git branch “ y el nombre de la rama nueva que vayamos a crear.

• ¿Cómo cambiar a una rama en Git?

Para cambiar de rama en Git vamos a escribir en la terminal: “git checkout “ y el nombre de la rama a la que queremos cambiar

• ¿Cómo fusionar ramas en Git?

Para fusionar ramas en Git vamos a escribir en la terminal: “git merge” y el nombre de la rama que queremos fusionar a la que nos encontramos posicionados.

• ¿Cómo crear un commit en Git?

Para crear un commit en Git vamos a escribir en la terminal: “git commit”. Es importante guardar los cambios agregándoles un “titulo” asi si debemos volver a un cambio guardado podemos saber rápidamente en que cambio quedò de la siguiente manera: “git commit -m “y aca escribimos una breve descripción del cambio”.

• ¿Cómo enviar un commit a GitHub?

Para poder subir los cambios al repositorio remoto debemos escribir en nuestra terminal: “git push origin main”, el main en este es caso un ejemplo de si se quiere enviar el trabajo de la rama main, sino se cambiaría por el nombre de la rama que deseamos enviar.

• ¿Qué es un repositorio remoto?

Un repositorio remoto es un servidor que contiene copias de los proyectos que se le cargan.

• ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git debemos desde la terminal escribir: “git remote add origin “ y la url/link del repositorio de GitHub que queremos agregar.

• ¿Cómo empujar cambios a un repositorio remoto?
Para empujar los cambios a un repositorio remoto debemos escribir en la terminal: “git push origin main”, el main en este caso es un ejemplo de si se quiere enviar el trabajo de la rama main, sino se cambiaría por el nombre de la rama que deseamos enviar.

• ¿Cómo tirar de cambios de un repositorio remoto?
Para traer cambios de un repositorio remoto lo que debemos escribir en la terminal es: “git pull”

• ¿Qué es un fork de repositorio?
El fork de un repositorio que se utiliza para crear una copia de un repositorio remoto, en esta nosotros vamos a poder trabajar y hacer modificaciones sin afectar al original de donde realizamos la copia.

• ¿Cómo crear un fork de un repositorio?
Para crear un fork de un repositorio es necesario posicionarnos en la página de GitHub donde esté cargado el repositorio y desde allí hacer click en Fork, luego seleccionamos con que cuenta realizaremos la copia, que nombre y descripción le pondremos a la misma y hacemos click en “Create fork”

• ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
Una vez que realizamos los cambios, que hicimos commit y que realizamos el push a nuestra copia del repositorio remoto, en la página principal del fork que creamos en GitHub aparecerá un cartel que dice “Compare and pull request”. Seleccionamos a que repositorio queremos aplicar los cambios y por último hacemos click en “Create pull request”

• ¿Cómo aceptar una solicitud de extracción?
Nos posicionamos en la página principal del repositorio remoto y desde alli vamos a la sección Pull request, ahí veremos cargadas las solicitudes y podremos si queremos incorporar los cambios, hacer click en “Merge pull request” y “Confirm merge”, y sino hacemos click en “Close pull request” para descartar.

• ¿Qué es una etiqueta en Git?
Una etiqueta en Git sirve para marcar un momento en la historia del proyecto. Esto puede tener distintos propósitos como por ejemplo el de identificar versiones a lo largo del código.

• ¿Cómo crear una etiqueta en Git?
Para crear una etiqueta en Git simplemente debemos escribir en la terminal: “git tag “ y añadimos el nombre de la etiqueta, por ejemplo: “git tag v1.0.0”

• ¿Cómo enviar una etiqueta a GitHub?
Para enviar una etiqueta a Github debemos, una vez que tengamos creada la etiqueta, escribir en la terminal: “git push origin “ y el nombre de la etiqueta. Por ejemplo, siguiendo con la creación de la etiqueta de la pregunta anterior, escribimos en la terminal: “git push origin tag v1.0.0” o si queremos enviar todos las etiquetas creadas sería: “git push - - tags”

• ¿Qué es un historial de Git?
El historial de git es un registro de todos los “commits” que se realizaron a lo largo del proceso de creación del proyecto, ahí podemos ver quien fue el autor y fecha y hora de su creación.

• ¿Cómo ver el historial de Git?
Para poder ver y revisar el historial de Git es necesario que en nuestra terminal escribimos: “git log”

• ¿Cómo buscar en el historial de Git?
Para buscar en específico en el historial de Git podemos utilizar el hash del commit, este es un identificador único del commit que hayamos realizado y podemos escribir en la terminal “git log “ y a continuación los primeros 7 caracteres del hash para abreviarlo

• ¿Cómo borrar el historial de Git?
Para borrar el historial de Git tenemos que escribir por la consola el siguiente comando: “git reset”, y de esa forma vamos a sacar del stage todos los archivos y carpetas del proyecto

• ¿Qué es un repositorio privado en GitHub?
Un repositorio privado en GitHub es un repositorio que no permite que cualquier persona en internet tenga acceso a ella

• ¿Cómo crear un repositorio privado en GitHub?
Para crear un repositorio privado en GitHub nos posicionamos en nuestro perfil de GitHub y desde allí hacer click en repositorios, luego en “New” y ahí cargamos el nombre del repositorio, con que perfil lo vamos a crear, la descripción, seleccionamos la opción “Private” y hacemos click en “Create repository”.

• ¿Cómo invitar a alguien a un repositorio privado en GitHub?
Para poder invitar a alguien debemos posicionarnos en la página principal de repositorio privado al que queremos invitar gente, seleccionar “Settings”, luego “Collaborators” y ahí hacemos click en “Add people” para escribir los nombres de las cuentas de GitHub que queremos que participen en nuestro repositorio privado.

• ¿Qué es un repositorio público en GitHub?
Un repositorio público en GitHub es un repositorio que permite que cualquier persona tenga acceso al proyecto

• ¿Cómo crear un repositorio público en GitHub?
Para crear un repositorio público en GitHub nos posicionamos en nuestro perfil de GitHub y desde allí hacer click en repositorios, luego en “New” y ahí cargamos el nombre del repositorio, con que perfil lo vamos a crear, la descripción, seleccionamos la opción “Public” y hacemos click en “Create repository”.

• **¿Cómo compartir un repositorio público en GitHub?**
Un repositorio público se puede compartir simplemente con el envío del link a la página principal del repositorio o se puede invitar a colaboradores yendo a la sección “Settings”, luego a “Collaborators” y ahí se carga la gente a la que se desea invitar al proyecto.

2) Realizar la siguiente actividad:

• Crear un repositorio.
o Dale un nombre al repositorio.
o Elije el repositorio sea público.
o Inicializa el repositorio con un archivo.

• Agregando un Archivo
o Crea un archivo simple, por ejemplo, "mi-archivo.txt".
o Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
o Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

• Creando Branchs
o Crear una Branch
o Realizar cambios o agregar un archivo
o Subir la Branch

Link al repositorio: https://github.com/EnzooChavez/Actividad2-TP2

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.

Link al repositorio: https://github.com/EnzooChavez/conflict-exercise