Trabajo Practico 2: Git y GitHub

Nombre: Salvatierra Ramon Ezequiel

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

- ***¿Qué es GitHub?***

  GitHub es una plataforma de repositorios de código para alojar proyectos mediante un sistema de control de versiones llamado Git

- ***¿Cómo crear un repositorio en GitHub?***

  Para crear un repositorio en GitHub debemos seguir los siguientes pasos:
  1. Crear una cuenta en GitHub.
  2. Ingresar al signo más (+) que se encuentra arriba a la derecha en la pantalla, y luego ingresar en donde dice “New repository”.
  3. Desde allí elegimos un nombre para el repositorio y distintas. configuraciones y le damos en “Create repository”.
  4. Y listo ya tenemos el repositorio.

- ***¿Cómo crear una rama en Git?***

  Para crear una rama en Git, desde la consola escribimos “git branch nueva_rama”, donde nueva_rama es el nombre de la nueva rama.

- ***¿Cómo cambiar a una rama en Git?***

  Para cambiar de rama en git debemos ingresar en la consola “git checkout rama_destino”, donde rama_destino es el nombre de la rama hacia donde nos dirigimos.

- ***¿Cómo fusionar ramas en Git?***

  Para fusionar ramas en Git debemos ubicarnos en la rama en donde se va a realizar la fusión, escribimos en la consola “git merge rama_a_fusionar”, si no hay conflictos, se realizara la fusión sin problemas, pero cuando hay conflictos debemos elegir como solucionarlo.

- ***¿Cómo crear un commit en Git?***

  Para hacer un commit en Git, primero debemos agregar los cambios que van a formar parte del commit con “git add .”, luego hacemos el commit con “git commit –m ‘realizando cambios’, debemos ingresar un mensaje que describa los cambios que realizamos.”

- ***¿Cómo enviar un commit a GitHub?***

  Para enviar los commit realizados en el repositorio local al repositorio en GitHub debemos usar “git pull origin rama”, donde “rama”, es el nombre de la rama a donde se dirigirán los cambios

- ***¿Qué es un repositorio remoto?***

  Un repositorio remoto es un repositorio de código que se encuentra en la nube, es decir que no se encuentra en nuestra PC, sino en un servidor de alojamiento de código.

- ***¿Cómo agregar un repositorio remoto a Git?***

  Para agregar un repositorio remoto a Git, debemos clonar el repositorio remoto hacia nuestra PC para poder trabajar localmente, para ello debemos escribir en la consola “git clone url_del_repositorio”.

- ***¿Cómo empujar cambios a un repositorio remoto?***

  Para enviar los cambios que realicemos en el código, hacia un repositorio remoto primero clonamos el repositorio, y luego de realizarle los cambios, escribimos en la consola: “git pull origin rama”, donde rama es el nombre de la rama hacia donde se dirigirán los cambios.

- ***¿Cómo tirar de cambios de un repositorio remoto?***

  Para traer los cambios que se encuentran en el repositorio remoto debemos usar “git pull origin nombre_rama”, donde nombre_rama es el nombre de la rama de donde traerá los cambios.

- ***¿Qué es un fork de repositorio?***

Un fork de un repositorio es crear una “copia” de un repositorio de otro usuario y guardarla en nuestros repositorios

- ***¿Cómo crear un fork de un repositorio?***

  Para crear un fork de un repositorio ingresamos al repositorio en github, desde allí, presionamos arriba a la derecha en donde dice “fork”, nos pedira que le pongamos un nombre, podríamos dejarle el que tiene, luego le damos a create fork. Y listo ya tendríamos el repositorio accesible desde nuestra cuenta.

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

  Para realizar una solicitud de pull request debemos seguir los siguientes pasos:
  1. Ingresar a nuestro repositorio forkeado
  2. Ingresar a la pestaña “pull request”
  3. Clicar en “new pull request”
  4. Desde alli comparamos los cambios entre las ramas y le damos a “create new pull request”

- ***¿Cómo aceptar una solicitud de extracción?***

  Para aceptar una solicitud de extracción, debemos seguir los siguientes pasos:
  1. Ingresar a nuestro repositorio
  2. Ingresar a la pestaña de pull request
  3. Seleccionar la pull request que se desea aceptar
  4. Revisar los cambios
  5. Aprobar la solicitud con el botón approve

- ***¿Qué es una etiqueta en Git?***

  Una etiqueta en Git es un nombre que se le da a un commit para poder identificarlo fácilmente más adelante

- ***¿Cómo crear una etiqueta en Git?***

Para etiquetar el último commit usamos:
git tag nombre-de-la-etiqueta
Para etiquetar un commit en específico:
git tag -a nombre-de-la-etiqueta id-del-commit -m "Mensaje"

- ***¿Cómo enviar una etiqueta a GitHub?***

  Para enviar una sola etiqueta a GitHub:
  git push origin nombre-de-la-etiqueta
  Para enviar todas las etiquetas:
  git push origin –tags

- ***¿Qué es un historial de Git?***

  El historial de Git es básicamente la lista de todos los cambios que se han hecho en el proyecto, en el orden en que se realizaron.

- ***¿Cómo ver el historial de Git?***

  Para ver el historial completo de Git utilizamos:
  git log
  Para ver un historial resumido:
  git log –oneline

- ***¿Cómo buscar en el historial de Git?***

  Usando *git log* con la opción *--grep=””,* puedes buscar una palabra clave en los mensajes de commit.

- ***¿Cómo borrar el historial de Git?***

  Para borrar el historial solo localmente (sin afectar GitHub):

  rm -rf .git

  git init

- ***¿Qué es un repositorio privado en GitHub?***

Un repositorio privado en GitHub es un repositorio al que solo pueden acceder el dueño del repositorio y los usuarios que este autorice

- ***¿Cómo crear un repositorio privado en GitHub?***

  Para crear un repositorio privado en GitHub, podemos seguir los mismos pasos para crear un repositorio, pero al momento de llegar a la sección de configuraciones, marcamos la opción “private”.

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

  1. Ingresamos a GitHub y vamos al repositorio privado
  2. Hacemos clic en la pestaña "Settings" del repositorio
  3. En el menú lateral izquierdo, hacemos clic en "Collaborators"
  4. Presionamos el botón "add people"
  5. Escribimos el nombre de usuario o correo de GitHub de la persona que deseamos invitar.
  6. Seleccionamos al usuario correcto de la lista que aparece y hacemos clic en "Add".
  7. La persona va a recibir una invitación y deberá aceptarla para poder acceder al repositorio

- ***¿Qué es un repositorio público en GitHub?***

  Un repositorio público en GitHub es un espacio donde podemos almacenar y compartir código que cualquiera puede ver, clonar y descargar

- ***¿Cómo crear un repositorio público en GitHub?***

  Para crear un repositorio público en GitHub, seguimos los mismos pasos mencionados sobre cómo crear un repositorio, pero al momento de llegar a la sección donde ingresamos el nombre del repositorio y las distintas configuraciones, dejamos marcada la opción public.

- ***¿Cómo compartir un repositorio público en GitHub?***

  Para compartir un repositorio público en GitHub es tan simple como compartir el enlace al repositorio

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 Branch**
  - **Crear una Branch**
  - **Realizar cambios o agregar un archivo**
  - **Subir la Branch**

Repositorio de la actividad realizada:

https://github.com/ezesalvatierra/repo-tp2-a2

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.**

Repositorio de la actividad resuelta:

https://github.com/ezesalvatierra/conflict-exercise