Alumno: Bruno Tofaletti

Comisión: 22

Práctico 2: Git y GitHub

Objetivo:

El estudiante desarrollará competencias para trabajar con Git y GitHub, aplicando conceptos fundamentales de control de versiones, colaboración en proyectos y resolución de conflictos, en un entorno simulado y guiado.

Resultados de aprendizaje:

1. Comprender los conceptos básicos de Git y GitHub: Identificar y explicar los principales términos y procesos asociados con Git y GitHub, como repositorios, ramas, commits, forks, etiquetas y repositorios remotos.
2. Manejar comandos esenciales de Git: Ejecutar comandos básicos para crear, modificar, fusionar y gestionar ramas, commits y repositorios, tanto en local como en remoto.
3. Aplicar técnicas de colaboración en GitHub: Configurar y utilizar repositorios remotos, realizar forks, y gestionar pull requests para facilitar el trabajo colaborativo.
4. Resolver conflictos en un entorno de control de versiones: Identificar, analizar y solucionar conflictos de merge generados en un flujo de trabajo con múltiples ramas.

Actividades

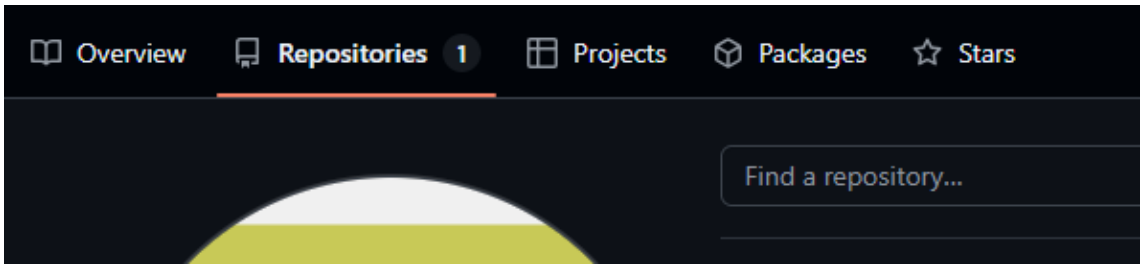
1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

- ¿Qué es GitHub?

  Es una plataforma basada en el sistema de control de versiones de Git. Permite tener repositorios con el código en la nube y de esa forma poder gestionar y colaborar con otros usuarios.

- ¿Cómo crear un repositorio en GitHub?

  Existen diferentes maneras para crear un repositorio, la más sencilla y directa es desde la misma plataforma GitHub, pestaña Repositories, click en New, luego se rellenas los campos a nuestra preferencia (Nombre, público o privado, descripción, si incluye un archivo Readme o no, entre otros).



Otra opción es desde el CLI, creando una carpeta en nuestra pc y con nuestro editor de código, abriendo una terminal y con el comando git init se inicializa un repositorio de forma local. Luego se realizan los comandos git add . para trackear nuestro archivo y git commit para confirmar los cambios. Por último, se ejecuta el comando git push para tenerlo en nuestra cuenta de GitHub.

- ¿Cómo crear una rama en Git?

  Desde la terminal de nuestro editor de código (En nuestro caso VS Code) ingresando en la terminal el comando git branch seguido del nombre que elijamos.

- ¿Cómo cambiar a una rama en Git?

  Ejecutando en la terminal el comando git checkout seguido del nombre de la rama destino

- ¿Cómo fusionar ramas en Git?

  Para fusionar ramas en Git es necesario posicionarnos en la rama original, ejecutar el comando git merge seguido del nombre de la rama a fusionar, de esta manera se añaden los cambios realizados siempre y cuando no se generen conflictos con el código en la rama original y tengamos que corregirlos.

- ¿Cómo crear un commit en Git?

  Para crear un commit en Git es necesario primero guardar nuestro código de forma local, luego añadir todos los cambios ejecutando el comando git add . , luego ejecutar git commit -m seguido de un comentario entre comillas detallando los cambios (-m es opcional pero siempre es recomendable hacerlo cuando se trabaja en conjunto).

- ¿Cómo enviar un commit a GitHub?

  Primero ejecutando el comando git commit -m para asegurar de tener los cambios de forma local y luego ejecutamos el comando git push -u origin master/main si es la primera vez que lo hacemos, luego simplemente git push

- ¿Qué es un repositorio remoto?

  Es una copia del repositorio Git local que se guarda en el servidor de la plataforma GitHub y de esta forma permite colaborar con otros y sincronizar los cambios.

- ¿Cómo agregar un repositorio remoto a Git?

  Se hace ejecutando el comando git remote add origin seguido del url del repositorio original en GitHub (ejemplo: https://github.com/usuario/repositorio.git).

- ¿Cómo empujar cambios a un repositorio remoto?

  Se hace ejecutando el comando git push origin seguido del nombre de la rama.

- ¿Cómo tirar de cambios de un repositorio remoto?

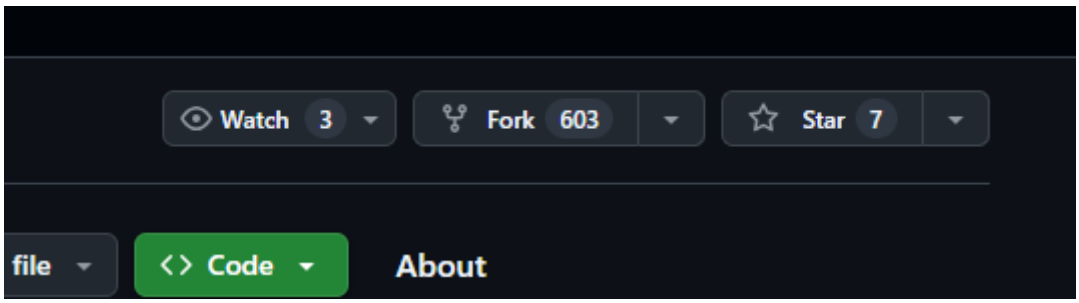
  Con el comando git pull origin seguido del nombre de la rama.

- ¿Qué es un fork de repositorio?

  Un fork es una copia de un repositorio en GitHub que no es de nuestra autoría, que nos permite modificar el código sin afectar el archivo original.

- ¿Cómo crear un fork de un repositorio?

  Accedemos al repositorio que queremos copiar y hacemos click en el botón Fork.



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

  Después de realizar los cambios en nuestro Fork, lo “pusheamos” a GitHub, una vez en la página nos dirigimos al repositorio y se hacemos click en Compare & Pull Request, completamos los datos adicionales y seleccionamos nuestra rama con los cabios (Esta debe estar apuntando a la rama original, main o master, que debe estar en la opción de base).

- ¿Cómo aceptar una solicitud de extracción?

  En la pestaña Pull Requests del repositorio aparece el pull solicitado, se revisan los cambios y una vez que estemos de acuerdo, hacemos click en Merge Pull Request y confirmamos. Luego opcional pero muy habitual se elimina la rama de GitHub.

- ¿Qué es un etiqueta en Git?

  Una etiqueta (Tag) es un referencia a un commit específico en el código, se usa principalmente para marcar versiones importantes en el historial. Existen dos tipos de etiquetas: Ligera (Se asigna solo una alias al commit), Anotada (Se incluyen nombre, correo, fecha, mensaje, etc).

- ¿Cómo crear una etiqueta en Git?

  Con el comando git tag seguido del nombre de la etiqueta.

- ¿Cómo enviar una etiqueta a GitHub?

  Con el comando git push origin seguido del nombre de la etiqueta, para enviar todas las etiquetas git push origin –tags.

- ¿Qué es un historial de Git?

  Es una línea temporal con todos los commit realizados en el repositorio, con los datos de autor, fechas y cambios.

- ¿Cómo ver el historial de Git?

  Para ver el historial se ejecuta el comando git log.

- ¿Cómo buscar en el historial de Git?

  Existen varias maneras de buscar en el historial. Se puede buscar por palabra clave en los mensajes de los commit con el comando git log --grep = seguido de la palabra clave entre comillas; Para buscar cambios en un archivo específico se ejecuta el comando git log -p seguido del nombre del archivo; Para buscar quién cambió una línea específica se ejecuta el comando git blame seguido del nombre del archivo; Para buscar por autor del commit se ejecuta git log --author = seguido del nombre del autor ente comillas; Para buscar por fecha se ejecuta git log --since = seguido de la fecha de inicio --until seguido de la fecha de finalización (ambas fechas entre comillas); Para buscar commit que modificaron una línea de código específica se ejecuta el comando git log -S seguido del fragmento de código en cuestión.

- ¿Cómo borrar el historial de Git?

  Para borrar el historial se ejecuta el comando git reset --hard hash.

- ¿Qué es un repositorio privado en GitHub?

  Un repositorio privado es aquel que solo es visible para el autor y/o los usuarios con permisos.

- ¿Cómo crear un repositorio privado en GitHub?

  De la misma manera que creamos un repositorio desde GitHub pero seleccionando la casilla de repositorio privado.

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

  Desde GitHub, entramos al repositorio, hacemos click en Setting, Collaborators y agregamos el nombre de usuario o email.

- ¿Qué es un repositorio público en GitHub?

  Es un repositorio visible para todos los usuarios de GitHub, que además se puede clonar.

- ¿Cómo crear un repositorio público en GitHub?

  De la misma manera que el repositorio privado pero seleccionando la opción de repositorio público.

- ¿Cómo compartir un repositorio público en GitHub?

  Compartiendo el link del repositorio.