Universidad Tecnológica Nacional

Facultad Regional San Nicolás

Tecnicatura Universitaria en Programación

Trabajo Práctico 2 – Trabajo Colaborativo

Tadeo Ressio

02 de abril de 2025, Córdoba, Argentina

Actividades

Actividad 1:

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

• ¿Qué es GitHub?

GitHub es una plataforma que se basa en el sistema de control de versiones Git con el cual podemos almacenar repositorios, compartirlos y trabajar en equipo de forma colaborativa.

Con GitHub podremos almacenar nuestros proyectos y podremos seguirlos desarrollando en otros dispositivos. Así mismo, podremos compartir nuestros programas a cualquier persona a través de internet, lo que nos permitirá dar a conocer nuestro trabajo así como compartir herramientas útiles en caso de precisarlo. Además, es fundamental para trabajar de manera colaborativa sobre un proyecto sin la necesidad de estar físicamente en el mismo espacio o de crear redes privadas para almacenar este tipo de información. Por último, es importante destacar el propio manejo de versiones, el cual nos permite tener un control sobre nuestro software para volver atrás en caso de tener inconvenientes o de poder desarrollar funcionalidades de forma paralela.

Si bien hay otros servicios similares tales como GitLabn o BitBucket, GitHub destaca por su uso extendido tanto en ámbitos privados como profesionales y empresariales. Hoy en día es el predominante dentro de la industria por lo que es importante saber cómo funciona.

• ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio en GitHub es importante entender la diferencia entre un repositorio local y un repositorio remoto. El repositorio local es una carpeta dentro del dispositivo que estamos utilizando y a la cual la definamos como tal. Luego, la idea es

relacionar dicho repositorio local con otro repositorio remoto almacenado en los servidores de GitHub. Haciendo esto, podremos sincronizar los cambios y que las modificaciones en el repositorio local se vean reflejadas en el repositorio remoto.

Entonces, para crear un repositorio remoto en GitHub, debemos acceder al propio sitio de GitHub (en realidad también existe la aplicación GitHub Desktop pero ignoraremos el uso de la misma). Luego, debemos loguearnos (o crear una cuenta en su defecto). Una vez hemos ingresado con nuestro usuario, podemos crear un repositorio remoto desde el símbolo "+" en la barra de navegación arriba a la derecha y seleccionando "New repository". Luego, deberemos elegir un nombre y algunas configuraciones básicas tales como si el repositorio será privado o público.

Con esto ya habremos creado un repositorio en GitHub el cual podremos clonar en nuestro dispositivo.

• ¿Cómo crear una rama en Git?

Para crear una rama en Git se debe usar el comando ***git branch***. Este comando, ejecutado sin ningún otro argumento, nos indicará la lista de ramas de dicho repositorio. Si, en cambio, utilizamos algo como: ***git branch nombreRama***, con esto crearemos una nueva rama (siempre y cuando no existe previamente otra rama con el mismo nombre).

Cabe aclarar que, por ejemplo GitHub, permite crear ramas desde su sitio web o desde GitHub Desktop.

• ¿Cómo cambiar a una rama en Git?

Para movernos a una rama, deberemos ejecutar el comando ***git checkout nombreRama***. Esto nos llevará a la rama indicada.

Es importante destacar que hoy en día también se utiliza el comando ***git switch nombreRama*** el cual cumple básicamente la misma función**.**

• ¿Cómo fusionar ramas en Git?

Para fusionar ramas en Git, se utiliza el comando ***git merge***.

Primero debemos posicionarnos en la rama en la que le queremos incluir los cambios de otra. Una vez nos hemos posicionado en la rama correcta, deberemos ejecutar ***git merge nombreRamaATraerCambios***. Esto traerá todos los nuevos cambios (archivos nuevos, carpetas nuevas, código dentro de archivos, etc.) y los fusionará con los datos previos de la rama en la que estamos posicionados.

Cabe aclarar que es normal que esto traiga "conflictos", es decir cambios en un mismo archivo por el cual deberemos seleccionar una de las dos versiones.

• ¿Cómo crear un commit en Git?

Para crear un commit, primero debemos utilizar el comando ***git add .***, el cual actualizará todos los cambios en el Staging Area (esta sirve específicamente para almacenar los cambios no incluidos todavía en un commit).

Luego, podremos proceder a ejecutar el comando ***git commit -m "Comentario de commit"***. Haciendo esto, hemos actualizado todos los cambios a la rama en la que estamos trabajando pero de manera local, luego tocaría actualizar dichos cambios con el repositorio remoto.

Cabe aclarar que el argumento -m es justamente para poder indicar un comentario que sirva como referencia para entender el commit a futuro. En cualquier caso, los commits crean un identificador alfanumérico que será el inequívoco para reconocerlos.

• ¿Cómo enviar un commit a GitHub?

Para enviar un commit a GitHub, debemos utilizar el comando ***git push origin nombreRama***. Esto actualizará todos los commits creados previamente en el repositorio remoto.

Cabe aclarar que el argumento ***origin***, sirve para hacer referencia a la URL del repositorio en el que estamos trabajando sin necesidad de incluirla manualmente.

• ¿Qué es un repositorio remoto?

Así como he aclarado en la primera pregunta, un repositorio remoto es un lugar donde se almacena un proyecto de programación en la nube. Este nos permite acceder al mismo, descargarlo y/o modificarlo ya sea por el propio creador del repositorio o por otros usuarios según la configuración de permisos que se elija.

Actualmente es una herramienta fundamental para el desarrollo de software a nivel profesional.

• ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git podemos utilizar el siguiente comando: ***git remote add origin <URL-del-repositorio>***. En este caso, la URL correspondería al repositorio remoto anteriormente creado desde GitHub.

• ¿Cómo empujar cambios a un repositorio remoto?

Para empujar cambios a un repositorio remoto hay que seguir los pasos mencionados para hacer un ***push***. Es decir que hay que ejecutar, primero ***git add .*** y, luego, realizar ***git commit***. Por último, se puede ejecutar el comando ***git push origin nombreRama***.

• ¿Cómo tirar de cambios de un repositorio remoto?

Para tirar de cambios de un repositorio remoto hay que volver a un ***commit*** anterior. Primero, para traer los cambios de los ***commits*** anteriores, debemos ejecutar ***git pull origin nombreRama***. Luego, para ver los ***commits*** existentes debemos ejecutar ***git log***. Por último, debemos ejecutar ***git checkout idCommit*** para volver a dicha etapa.

• ¿Qué es un fork de repositorio?

Al hacer ***fork*** de un repositorio de GitHub, lo que estamos haciendo es crear una copia de dicho repositorio en nuestra propia cuenta de GitHub. Luego, podremos trabajar sobre este nuevo repositorio y, si así lo quisiéramos, podríamos proponer que los cambios que hicimos se actualicen en el repositorio original.

• ¿Cómo crear un fork de un repositorio?

Si bien se puede hacer por consola, para hacer un ***fork*** de un repositorio en GitHub, debemos acceder a la página dónde se encuentre el repositorio en sí. Una vez allí, debemos seleccionar la opción ***fork***. Esto nos llevará a otra vista donde podremos seleccionar el nombre que tendrá en nuestra cuenta y, además, podremos incluir una descripción del mismo.

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

Para hacer una ***pull request***, debemos hacer ***fork*** del repositorio, clonarlo en nuestro dispositivo, realizar cambios, realizar un ***commit*** y, finalmente, hacer un ***push.***

Una vez hayamos hecho esto, luego podremos ir a la copia de nuestro repositorio dentro del sitio de GitHub y seleccionar la opción ***open pull request***. Con esto, el usuario propietario del repositorio original, podrá revisar dicha petición y aceptarla en caso de que así lo considere.

• ¿Cómo aceptar una solicitud de extracción?

Para aceptar las ***pulls request***, debemos acceder al repositorio en el sitio de GitHub. Una vez allí, deberemos seleccionar la opción ***pull request***. En esta página veremos todas las peticiones realizadas por otros usuarios y podremos revisarlas debidamente. Se mostrarán los cambios en el código y se podrá aceptar o rechazar dicha petición.

En caso de querer aceptar los cambios, se debe seleccionar la opción ***merge pull request*** que aparece por debajo.

• ¿Qué es un etiqueta en Git?

Una etiqueta en Git es un identificador en el historial de ***commits***, el cual nos servirá para reconocer puntos importantes en el desarrollo. Se usa para marcar puntos de inflexión en el proyecto para que, en caso de tener que volver a ellos, sea más facil.

• ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta, debemos usar el comando ***git tag -a nombreTag -m "Comentario tag".*** Esto nos permitirá acceder a este punto del desarrollo de nuestro programa a futuro.

Cabe aclarar que el argumento -a sirve para indicar que será “anotted tag”, es decir que se incluirán datos como el mensaje, la fecha y hora, etc. La -m, al igual que para los ***commits***, indica que se incluirá un mensaje descriptivo.

• ¿Cómo enviar una etiqueta a GitHub?

Para enviar una etiqueta al repositorio remoto de GitHub, debemos utilizar, nuevamente, el comando ***push*** pero esta vez indicando el nombre del tag. Ej.: ***git push origin nombreTag*.**

• ¿Qué es un historial de Git?

El historial de Git hace referencia a todos los cambios que fueron sucediendo entre los distintos ***pushs*** que se han hecho en el proyecto. Es decir que es un historial de los cambios del programa.

• ¿Cómo ver el historial de Git?

Para ver el historial de Git, debemos usar el comando ***git log***. Esto nos mostrará todos los commits realizados en sus distintas ramas.

• ¿Cómo buscar en el historial de Git?

Se puede buscar commits específicos dentro del ***log*** de Git utilizando comandos. Algunos son:

git log --grep="palabra_clave": el comando grep es muy conocido por fuera de Git para buscar texto dentro de archivos.

git log --author="Nombre del autor": con esto podremos buscar los ***commits*** realizados por un usuario específico.

git log --since="2024-01-01" --until="2024-03-31": así podremos buscar todos los ***commits*** dentro de un período de tiempo.

• ¿Cómo borrar el historial de Git?

Si se quiere borrar todo el historial de un proyecto de Git, se puede crear una rama y reemplazar la original.

Sin embargo, también es posible borrar ***commits*** específicos. Para esto usaremos el comando ***git rebase -i nombreCommit***. Cabe aclarar que el nombre del ***commit*** en realidad sería su ***hash***, es decir un identificador que se crea a la hora de realizar dicho commit.

• ¿Qué es un repositorio privado en GitHub?

Un repositorio privado es un repositorio albergado en una cuenta de GitHub pero que no es visible para otro usuario que no sea el propio creador o las personas que este haya definido para darles visibilidad.

• ¿Cómo crear un repositorio privado en GitHub?

A la hora de crear un repositorio se consulta por si se debe hacer privado o público. Igualmente es posible realizarlo, luego, ingresando ***settings*** del repositorio y yendo a la zona inferior donde podremos cambiar la visibilidad o, además, eliminar el repositorio.

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

Para invitar a alguien a un repositorio privado en GitHub, es necesario invitar a dicho usuario como ***colaborador***. Para esto debemos acceder a nuestro repositorio e ir a ***settings.*** Una vez allí, debemos seleccionar la opción de ***collaborators*** que figura en la zona izquierda de la pantalla. Por último, clickeando ***add people*** podremos buscar por nombre al usuario deseado.

• ¿Qué es un repositorio público en GitHub?

Un repositorio público en GitHub es un repositorio remoto cuya visibilidad está habilitada para cualquier usuario sin ningún tipo de restricción. Esto implica que otros usuarios puedan descargar dicho repositorio y utilizarlo indiscriminadamente.

• ¿Cómo crear un repositorio público en GitHub?

Para crear un repositorio público en Github, al igual que para hacerlo privado, es una pregunta que se realiza a la hora de crear el repositorio. Igualmente se puede cambiar la visibilidad a futuro.

• ¿Cómo compartir un repositorio público en GitHub?

Para compartir un repositorio público en GitHub es tan simple como copiar la URL que se indica a la hora de posicionarnos sobre el proyecto dentro del sitio web.

Actividad 2:

A continuación, comparto el enlace al repositorio donde se podrán ver las acciones solicitadas en el TP:
https://github.com/tadeoressio/Programacion1_TP2_Repo1.git

Actividad 3:

A continuación, comparto el enlace al repositorio donde se podrán ver las acciones solicitadas en el TP:
https://github.com/tadeoressio/Programacion1_TP2_Repo2.git