Benjamin Ugarte **Comision 22**

Trabajo Práctico 2: Git y GitHub

Actividades

- ¿Qué es GitHub?

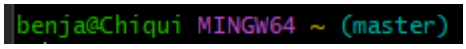
  Respuesta:
  GitHub es una plataforma web donde disponemos de un perfil, similar a una red social, permite a programadores y desarrolladores a subir códigos y proyectos de software (deforma pública o privada). Es como un espacio en la nube donde guardas y compartís el código de manera segura, también hace mas fácil trabajar en equipo, donde cada uno de los integrantes o cualquier otra persona puede ayudar o corregir errores de un código.

- ¿Cómo crear un repositorio en GitHub?

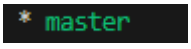
  Respuesta:
  Para crear un repositorio, primeramente se debe crear una cuenta, una vez creada y personalizada nos dirigimos al botón mas "+" y vamos a la opción "nuevo repositorio" (en ingles es "new repository"), los siguiente que se de hacer (cuando la pagina nos redirecciono y termino de cargar) es poner le un nombre al repositorio, puede ser cualquiera, hasta el mismo nombre del proyecto, después se personalizar, colocar una descripción, se puede poner en público, en privado, etc… Para finalizar apretamos "Crear Repositorio", y listo, ya esta creada, después ya debes sincronizarlo con el repositorio local de Git.

- ¿Cómo crear una rama en Git?

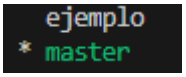
  Respuesta:
  Para crear una rama, hay que dirigirse o abrir una consola, un cmd, una terminal o también la misma terminal del vs code, una vez estando allí escribimos el comando "git branch" para saber en que ramas estamos y como se llama nuestra rama original.
  Puede tener varios nombres como main, master, etc.. Asi de vería de aparecer.

  El "master" es la rama original, en vs code aparece con un asterisco (*).

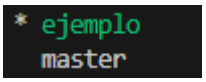
  

  Luego procedemos a crear una rama, escribimos el comando "git branch" pero luego ponemos un espacio y escribimos el nombre de la nueva rama por ejemplo "ejemplo", quedara así "git branch ejemplo" aprietas enter y ya está creada
  para confirmar escribimos "git branch" y te aparecerá así.

- ¿Cómo cambiar a una rama en Git?

  Respuesta:
  Para cambiar a ella escribimos el comando "git checkout ejemplo", para confirmar que estas en rama2 escribís el comando "git branch" y debería de aparecer así.



- ¿Cómo fusionar ramas en Git?

  Respuesta:
  Para fusionar las ramas debemos escribir el comando "git checkout (el nombre de la rama principal)", (si ya estas en tu rama principal omite este paso) una vez estando en la rama escribimos el comando "git merge ejemplo" y presionamos enter, eso procede a unir los cambios que se hicieron en la rama ejemplo a la rama principal master o main.

- ¿Cómo crear un commit en Git?

  Respuesta:
  Para crear un commit primera mente se debe guardar los cambios que se hicieron en el código el comando es "git add .", luego de haber echo eso el paso a seguir es el commit en git, que es para comentar que cambios hubo en el código
  para hacerlo hay que poner el comando "git commit -m "un texto de los cambios que hiciste"" aprietas enter y ya se creó el commit.

- ¿Cómo enviar un commit a GitHub?

  Respuesta:
  Un commit se envía de siguiente manera, primera mente se debe haber creado un repositorio remoto en GitHub, de ahí obtienes un URL https://github.com/tu-usuario/tu-repo.git, con esa URL te vas a la terminal vs code y pones el siguiente comando "git remote add origin https://github.com/tu-usuario/tu-repo.git" apretamos enter, escribimos otro comando "git push -u origin master" y listo.
  Si es la primera vez, te pide que inicies sesión en GitHub, asi puede Git estar conectada a tu cuenta de GitHub y subir los códigos sin problemas.

- ¿Qué es un repositorio remoto?

  Respuesta:
  Un repositorio remoto es una copia del codigo o proyecto de la cual está en constante cambio que va guardado en un servidor como una nube por ejemplo GitHub, GitLab, Bitbucket, entre otros, lo cual beneficia aquellos que trabajan en equipo o a distancia, también otras personas pueden ver el codigo si se lo guarda en público.
  También tiene el beneficio de ver y revisar el historial de cambios que hubieron en el código y hacer copias de seguridad.

- ¿Cómo agregar un repositorio remoto a Git?

  Respuesta:
  Para agregar un repositorio remoto en Git, se debe haber hecho con anterioridad en la terminal Git, la configuración en la ponemos nuestros datos de usuario local, a ver creado o iniciado un repositorio local y también a ver echo los siguientes comandos "git add ." "git commit -m "Un texto para el commit"".
  Lo siguiente que se debe hacer para agregar un repositorio remoto es escribir el siguiente comando "git remote add origin https://github.com/tu-usuario/tu-repo.git" y listo ya está el repositorio remoto agregado a Git.

- ¿Cómo empujar cambios a un repositorio remoto?

  Respuesta:
  Para pujar los cambios al repositorio remoto se debe escribir el siguiente comando "git push -u origin master" y listo ya se subieron los cambios a GitHub u cualquier otra plataforma de preferencia.

- ¿Cómo tirar de cambios de un repositorio remoto?

  Repuesta:
  Se debe abrir la termina git bash here o terminal de vs code, una vez allí escribimos el comando "git pull origin master" "origin" es el nombre del repositorio remoto, es muy importante acordarse el nombre que le hayamos puesto, es recomendable dejar "origin" para no generar con funciones futuras, y "master" es el nombre de la rama principal también lo podemos colocar el nombre de otra rama que estemos trabajando.

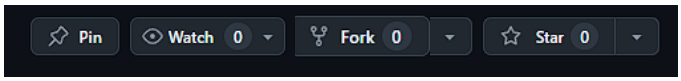
- ¿Qué es un fork de repositorio?

  Respuesta:
  Un fork es una copia de un repositorio existente creada por otra cuanta, la cual

con ella podemos hacer modificaciones sin tener que afectar al original, lo cual esto nos beneficia a no tener que descargar o clonar en el repositorio local ya que a la copia la podemos hacer desde la cuenta que tengamos en GitHub.

- ¿Cómo crear un fork de un repositorio?

  Respuesta:
  Para crear un fork se debe iniciar sesión o crear una cuenta en GitHub, una vez allí nos dirigimos al repositorio que nos hayan compartido entrando atreves de un link o URL. Estando ahí vamos al botón donde dice “forck” que esta ubicado en la parte superior derecha junto a otros botones más. Así



  Presionamos el botón, nos llevara a “Create a new fork”, allí nos pedirá algunas cosas pero por defecto ya está todo, salvo que se quiera cambiar algo como la descripción. Revisamos si esta todo correcto y luego apretamos el botón “create fork” esperamos unos segundos mientras se crea una copia del repositorio que hayamos elegido, una vez finalizada la carga ya tienes la copia en tu cuenta.

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

  Respuesta:
  Una solicitud de extracción (pull request) se usa para proponer cambios a un proyecto que está en GitHub. Es común cuando trabajás con proyectos colaborativos o querés contribuir a un repositorio que no te pertenece. Primero, tenés que hacer un “fork” del repositorio (crear una copia en tu cuenta), clonar esa copia a tu máquina local, crear una nueva rama (branch), hacer tus cambios y luego hacer push a esa rama. Después, desde tu repositorio en GitHub, vas a ver un botón que dice “Compare & pull request”. Hacés clic ahí, escribís una descripción de los cambios y hacés clic en **“**Create pull request”. El dueño del repositorio original podrá revisar y aceptar o rechazar los cambios.

- ¿Cómo aceptar una solicitud de extracción?

  Respuesta:
  Para aceptar una pull request, primero tenes que ser el dueño del repositorio o tener permisos para gestionarlo. Cuando alguien crea una pull request, te llegará una notificación. Entras al repositorio, vas a la pestaña **“**Pull requests**”**, seleccionas la que queres revisar, y podés ver todos los archivos que se modificaron. Si todo está bien, hacés clic en el botón verde “Merge pull request”, y después en “Confirm merge”. Eso une los cambios propuestos con tu rama principal (por lo general, main o master).

- ¿Qué es un etiqueta en Git?

  Respuesta:
  Una etiqueta (tag) en Git es como una marca o punto específico en la historia del proyecto que se usa para señalar algo importante, como una nueva versión (v1.0, v2.0, etc.). A diferencia de las ramas, las etiquetas no cambian, son fijas. Son útiles para saber en qué estado estaba el proyecto en un momento clave, por ejemplo cuando se hizo una versión estable.

- ¿Cómo crear una etiqueta en Git?

  Respuesta:
  Para crear una etiqueta, primero tenes que tener un repositorio con al menos un commit. En la terminal de VS Code o Git Bash, usas el siguiente comando “`git tag -a v1.0 -m "Versión 1.0 estable"`”. Eso crea una etiqueta llamada v1.0 con un mensaje descriptivo. Si no queres mensaje, podes usar “`git tag v1.0`” para hacer una etiqueta simple.

- ¿Cómo enviar una etiqueta a GitHub?

  Respuesta:
  Cuando cares una etiqueta local, no se sube automáticamente al repositorio remoto. Para subirla, usas este comando “`git push origin v1.0`” , si queres subir todas las etiquetas que hiciste, podes usar “`git push origin –tags`”. Después de eso, las vas a ver en GitHub, en la pestaña "Releases".

- ¿Qué es un historial de Git?

  Respuesta:
  El historial de Git es el registro de todos los cambios que se hicieron en el proyecto: commits, merges, ramas creadas, eliminadas, etc. Es como una línea de tiempo que muestra quién hizo qué cambios, cuándo y por qué. Este historial es muy útil para revisar el progreso, encontrar errores, o volver a una versión anterior del proyecto.

- ¿Cómo ver el historial de Git?

  Respuesta:
  Para ver el historial, simplemente escribís en la terminal de git “`git log`” Este comando muestra todos los commits con su ID, autor, fecha y mensaje. También podes usar este comando “`git log –oneline`” para ver un resumen más compacto.

En VS Code también podés ver el historial si tenés extensiones como GitLens, que muestran de forma visual quién modificó cada línea de código.

- ¿Cómo buscar en el historial de Git?

  Respuesta:
  Para buscar en el historial de Git podemos usar diferentes comandos según lo que necesitemos encontrar. El comando principal es "git log" que nos muestra todo el historial de commits con sus fechas, autores y mensajes para finalizar o salir del historial aprieta “q” que significa *quit* eso nos sacara del historial. Si queremos algo más resumido usamos "git log –oneline" que muestra cada commit en una sola línea. Para buscar commits que tengan una palabra específica escribimos "git log –grep="palabra"". Si necesitamos ver los cambios de un archivo en particular usamos "git log –nombre-archivo". También está el comando "git blame nombre-archivo" que nos muestra quién modificó cada línea y en que commit. Estos comandos son muy útiles cuando trabajamos en equipo o necesitamos recordar qué cambios hicimos anteriormente.

- ¿Cómo borrar el historial de Git?

  Respuesta:
  Para borrar el historial de Git debemos tener mucho cuidado porque es una acción que no se puede deshacer fácilmente. Si queremos crear un historial completamente nuevo, podemos usar "git checkout –orphan nueva-rama" para crear una rama sin historia, luego "git add ." para añadir todos los archivos, después "git commit –m "Nuevo inicio"" para crear el primer commit. Seguimos con "git branch -D main" para borrar la rama principal antigua, "git branch -m main" para renombrar nuestra nueva rama a main, y finalmente "git push -f origin main" para forzar la subida al repositorio remoto. Es importante saber que esto causará problemas a otros colaboradores si ya compartiste tu repositorio, así que debes avisarles antes de hacer esto.

- ¿Qué es un repositorio privado en GitHub?

  Respuesta:
  Un repositorio privado en GitHub es un espacio donde guardamos nuestro código que solo puede ser visto por nosotros y las personas que invitemos específicamente. Este tipo de repositorio es ideal cuando trabajamos en proyectos que no queremos que sean públicos, como código de empresas, proyectos personales o cualquier cosa que contenga información sensible. La principal ventaja es que mantenemos el control de quién puede ver y modificar nuestro

código. En la cuenta gratuita de GitHub tenemos un número limitado de colaboradores para repositorios privados, pero todas las otras funcionalidades son iguales a las de los repositorios públicos.

- ¿Cómo crear un repositorio privado en GitHub?

  Respuesta:
  Para crear un repositorio privado en GitHub primero debes iniciar sesión en tu cuenta de GitHub. Luego haces clic en el símbolo "+" que está en la esquina superior derecha y seleccionas "New repository". Escribes un nombre para tu repositorio y opcionalmente añades una descripción. Lo importante es que selecciones la opción "Private" en las opciones de visibilidad. Si quieres puedes marcar la casilla "Initialize this repository with a README" para crear un archivo inicial. Finalmente haces clic en "Create repository" y listo, ya tienes tu repositorio privado creado. Ahora solo tú puedes verlo hasta que invites a otras personas.

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

  Respuesta:
  Para invitar a alguien a tu repositorio privado debes ir a tu repositorio en GitHub y hacer clic en "Settings" que está en la parte superior. En el menú lateral izquierdo seleccionas "Collaborators" y luego haces clic en "Add people". Ahí puedes buscar a la persona por su nombre de usuario de GitHub, nombre completo o correo electrónico. Una vez que encuentres a la persona correcta, la seleccionas y haces clic en "Add". La persona recibirá una invitación por correo electrónico y debe aceptarla para poder acceder a tu repositorio. Puedes invitar a tantas personas como necesites dependiendo de tu plan de GitHub, y también puedes eliminar colaboradores en cualquier momento si ya no quieres que tengan acceso.

- ¿Qué es un repositorio público en GitHub?

  Respuesta:
  Un repositorio público en GitHub es un espacio donde el código que subimos puede ser visto por cualquier persona en internet, incluso sin tener cuenta en GitHub. Este tipo de repositorio es ideal para proyectos de código abierto, ejemplos educativos o cuando queremos compartir nuestro trabajo con la comunidad. Aunque cualquiera puede ver el código, solo tú y los colaboradores que autorices pueden hacer cambios directamente. Otras personas pueden sugerir cambios mediante "pull requests" que tú debes aprobar. Los repositorios públicos son excelentes para construir un portafolio de proyectos que puedes mostrar a posibles empleadores o para colaborar con otros desarrolladores alrededor del mundo.

- ¿Cómo crear un repositorio público en GitHub?

Respuesta:
Para crear un repositorio público en GitHub primero debes iniciar sesión en tu cuenta. Luego haces clic en el símbolo "+" en la esquina superior derecha y seleccionas "New repository". Escribes un nombre para tu repositorio y es recomendable añadir una descripción clara ya que será visible para todos. En las opciones de visibilidad seleccionas "Public". Es buena idea marcar la casilla "Initialize this repository with a README" para añadir información básica sobre tu proyecto. También puedes añadir una licencia que defina cómo otros pueden usar tu código y un archivo .gitignore según el tipo de proyecto. Finalmente haces clic en "Create repository" y ya está listo tu repositorio público para que cualquiera lo vea.

- ¿Cómo compartir un repositorio público en GitHub?

  Respuesta:
  Para compartir un repositorio público en GitHub simplemente necesitas compartir su URL. Puedes copiar la dirección directamente desde la barra del navegador cuando estás en la página de tu repositorio, algo como "https://github.com/benja-UG/proyect-". Esta URL la puedes enviar por correo electrónico, redes sociales, mensajes o cualquier medio que uses para comunicarte. También puedes usar el botón "Code" de color verde que aparece en la página del repositorio para copiar la URL que otros necesitarán para clonar tu proyecto. Cuando compartes un repositorio público, cualquier persona puede ver tu código, descargarlo como ZIP o clonarlo para usarlo, pero solo podrán hacer cambios si tú los aceptas a través de pull requests o los añades como colaboradores.