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

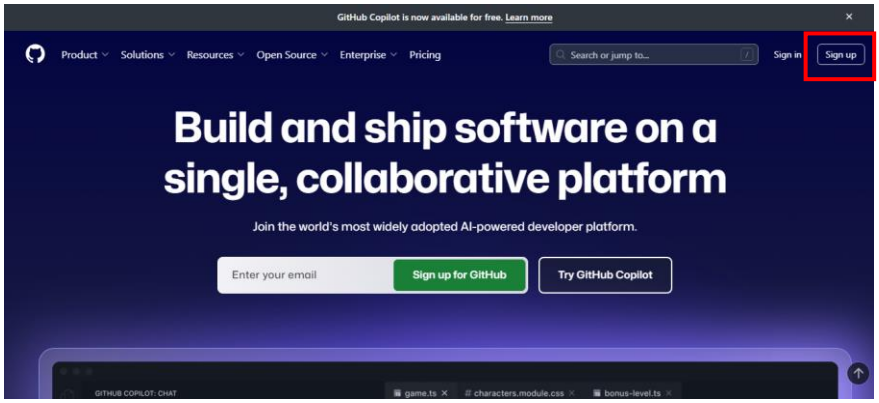
1) *Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas):*

- ¿Qué es GitHub?

  GitHub es una plataforma de desarrollo colaborativo que utiliza Git como sistema de control de versiones, es un sitio web y un servicio en la nube. Permite a los desarrolladores almacenar, gestionar y colaborar en proyectos de software. Además, ofrece características como seguimiento de problemas, revisiones de código y documentación, lo que facilita el trabajo en equipo.

- ¿Cómo crear un repositorio en GitHub?

  a) Inicia sesión en tu cuenta de GitHub. Si no tenés cuenta, entrá a la página github.com y creá una, has click en “sing up”.



Luego “create your acount”, luego se completan los datos personales solicitados y presionando “créate acount” ya se pueden subir los primeros repositorios.

b) Haz clic en el botón "New" o "Nuevo repositorio" en la parte superior derecha de la página.
c) Completa el formulario:
   - Nombre del repositorio: Elige un nombre único, puede coincidir con el nombre del repositorio local.
   - Descripción (opcional): Agrega una breve descripción de tu proyecto.
   - Visibilidad: Elige si quieres que sea público o privado.
d) Haz clic en "Create repository" para crear tu nuevo repositorio.
e) https://github.com/noeliasanchez90/Primer-repo-git.git
f) Una vez creado en primer repositorio en GITHUB nos indica
   …or create a new repository on the command line:
   echo "# Primer-repo-git" >> README.md
   git init
   git add README.md
   git commit -m "first commit"
   git branch -M main
   git remote add origin https://github.com/noeliasanchez90/Primer-repo-git.git
   git push -u origin main

   …or push an existing repository from the command line
   git remote add origin https://github.com/noeliasanchez90/Primer-repo-git.git
   git branch -M main
   git push -u origin main

- ¿Cómo crear una rama en Git?

  Si se ejecuta el comando > git Branch . nos imprime la lista de ramas que actualmente tiene nuestro repositorio, nos aparecerá probablemente que tenemos la rama main o master según como se haya definido en la instalación de git, y aparece con un * la rama en donde nos encontramos.

  Para crear una nueva rama, puedes usar el siguiente comando:

  >git Branch nombre-de-la-nueva-rama

  Luego para reconocer la nueva rama creada, ejecutamos el comando:

  >Git branch y nos devuleve las ramas existentes en el repositorio, recordar que el * nos indica sobre que rama estamos parados.

- ¿Cómo cambiar a una rama en Git?

  Para cambiarte a una rama existente, utiliza:

  >git checkout nombre-de-la-rama

- ¿Cómo fusionar ramas en Git?

  Primero, asegúrate de estar en la rama a la que deseas fusionar. Por lo general, es a la rama principal (master o main) o cualquier otra rama en la que estés integrando los cambios. Luego, usa:

  >git checkout master

>git merge nombre-de-la-rama-a-fusionar

Este comando incorpora los cambios de nuevaRama en master.

- ¿Cómo crear un commit en Git?

  Para crear un commit en Git, puedes usar el comando >git commit. Antes de usar git commit, debes usar >git add para seleccionar los cambios que quieres incluir.

  Pasos para crear un commit:

    Usar git add para seleccionar los cambios que quieres incluir

    Usar git commit para crear una instantánea de los cambios

    Incluir un mensaje con la opción -m

    Opción -m

      La opción -m significa mensaje

      El mensaje debe ser una breve descripción de los cambios

      El mensaje debe estar al final del comando y entre comillas

      Si no incluyes -m, se te pedirá que agregues un mensaje en tu editor de texto predeterminado

    Recomendaciones para los mensajes de commit

      Usa el verbo imperativo (Add, Change, Fix, Remove, …)

      No uses punto final ni puntos suspensivos

      Usa como máximo 50 caracteres

      Añade todo el contexto que sea necesario

      Usa un prefijo para tus commits

  Git considera cada commit como una versión segura de tu proyecto. Es recomendable ejecutar git commit regularmente y después de hacer cambios importantes.

- ¿Cómo enviar un commit a GitHub?

  Para enviar un commit a GitHub, puedes usar el comando >git push. Este comando inserta los cambios realizados en la rama local en un repositorio remoto.

    Paso 1 Asegúrate de estar en el directorio local que se creó cuando clonaste el repositorio.

    Paso 2 Agrega los archivos al "stage" para confirmarlos en el repositorio local.

    Paso 3 Confirma los cambios con git commit -m "mensaje de confirmación".

    Paso 4 Inserta los cambios en el repositorio remoto con git push.

  Para enviar un commit a GitHub, puedes usar el siguiente comando:

    >git add . para hacer un seguimiento de los cambios realizados en la carpeta

>git commit -m "mensaje de confirmación" para preparar los cambios para empujar a GitHub

>git push origin master para empujar los archivos a GitHub

En GitHub Docs puedes encontrar más información sobre cómo subir cambios a GitHub.

- ¿Qué es un repositorio remoto?

Un repositorio remoto es una copia de un proyecto que se encuentra alojada en una red o en internet. Se puede compartir entre varios miembros de un equipo.

Características

Se alojan en un servidor remoto, que puede estar en internet o en un servidor externo

Se pueden tener varios repositorios remotos

En cada repositorio se suelen tener permisos de lectura y escritura o solo de lectura

Se pueden clonar repositorios remotos para crear una versión local del repositorio en el equipo

Se pueden usar varios clones de un repositorio remoto para colaborar con miembros del equipo en el mismo proyecto

Comparación con repositorios locales

Los repositorios locales se alojan en un equipo local para un usuario individual. Los repositorios remotos son accesibles para todos los miembros del equipo.

Uso de repositorios remotos

Los repositorios son un tipo de almacenamiento digital centralizado que los desarrolladores utilizan para realizar y administrar cambios en el código fuente de una aplicación.

- ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, puedes usar el comando >git remote add en la terminal. Este comando se debe ejecutar dentro del directorio donde está el repositorio.

Paso a paso

1. Entrar al directorio donde está el repositorio
2. Ejecutar el comando >git remote add
3. Especificar un nombre remoto, por ejemplo, origin

También puedes clonar un repositorio remoto usando el comando >git clone. Para ello, necesitas la URL del repositorio.

Otros comandos de Git

- git init Crea un nuevo repositorio
- git add Añade un archivo al área de preparación del repositorio
- git commit Crea un commit nuevo con un mensaje que describa qué trabajo se ha hecho
- git push Envía los cambios al repositorio remoto
- git pull Extrae y descarga contenido desde un repositorio remoto
- git remote set-url Cambia la dirección URL del repositorio remoto de HTTPS a SSH

Para subir un repositorio local a un repositorio online, puedes crear un repositorio en un sitio online como GitHub o bitbucket.

- ¿Cómo empujar cambios a un repositorio remoto?

Para empujar cambios a un repositorio remoto con Git, puedes usar el comando >git push. Antes de hacerlo, debes asegurarte de haber confirmado los cambios en el repositorio local.

Sintaxis del comando

>git push <nombre del repositorio> <nombre de la rama>

Ejemplos de uso

- Para subir los cambios locales al repositorio en línea, puedes ejecutar git push origin main
- Para renombrar una rama, puedes usar git push REMOTE-NAME LOCAL-BRANCH-NAME:REMOTE-BRANCH-NAME
- Para subir una etiqueta única, puedes usar git push REMOTE-NAME TAG-NAME
- Para borrar una rama remota, puedes usar git push REMOTE-NAME :BRANCH-NAME

Otros comandos relacionados

- git remote add para agregar un repositorio remoto
- git remote para crear, ver, y eliminar conexiones con otros repositorios
- git pull para obtener y fusionar los últimos cambios del repositorio remoto
- git merge --abort para restaurar la rama al estado que tenía antes de extraerla

Para aprender más sobre el comando git push, puedes consultar la documentación de Git o ejecutar git push --help.

- ¿Cómo tirar de cambios de un repositorio remoto?

Para obtener los cambios de un repositorio remoto, puedes usar el comando >git fetch. Esto te permite recuperar las ramas de seguimiento remoto y etiquetas nuevas que hayan hecho otras personas.

Procedimiento

Abre una terminal

Escribe git fetch

Presiona Enter

GitHub Docs indica que al capturar desde un repositorio se obtienen todas las ramas de seguimiento remoto nuevas y etiquetas sin combinar estos cambios en las ramas propias.

Más información

Puedes agregar un remoto nuevo y luego recuperarlo.

Para quitar la dirección URL de un repositorio remoto del repositorio, puedes usar el comando git remote rm.

Para deshacer los cambios realizados en el historial de confirmaciones de un repositorio, puedes usar el comando git revert.

Puedes examinar el estado del repositorio mediante el comando git log.

- ¿Qué es un fork de repositorio?

Un fork de repositorio es una copia exacta de un repositorio original. El término en inglés es "fork" y en español se traduce como "bifurcación".

Los forks son útiles para:

Contribuir a proyectos sin tener permisos de escritura

Trabajar en un proyecto sin necesidad de acceder al repositorio central

Experimentar con nuevas características

Colaborar con otros desarrolladores

Para hacer un fork de un repositorio en GitHub, se puede:

Ir a la página del repositorio

Pulsar el botón "Fork" que se encuentra en la parte superior derecha de la página

Una vez que se tiene la copia, se pueden enviar los cambios al repositorio original para su aprobación. A este proceso se le llama "Pull Request".

Los forks tienen las siguientes características:

Se pueden crear repositorios idénticos pero independientes

Los cambios que se realicen en el repositorio original no se transmiten automáticamente a la copia

La copia puede ser clonada y utilizada como cualquier otro repositorio git

- ¿Cómo crear un fork de un repositorio?

Para crear un fork de un repositorio en GitHub o Bitbucket, se puede seguir estos pasos:

Ir a la página del repositorio que se desea clonar

Presionar el botón Fork

Elegir la cuenta o organización donde se desea crear la copia

Si se desea, se le puede dar un nombre y una descripción a la copia

Presionar Create fork

El fork es una copia exacta del repositorio original, pero con una URL diferente. Se puede usar para contribuir a un proyecto de forma segura.

Ventajas de usar un fork

El contribuyente trabaja con su propia copia, por lo que no necesita acceso al repositorio central

El propietario del repositorio puede gestionar fácilmente las contribuciones de muchas personas

Los cambios se pueden incorporar al repositorio original si el propietario los considera correctos

Cómo contribuir a un proyecto con un fork

Hacer los cambios y darles push al repositorio

Hacer un pull request con los cambios en alguna de las ramas del proyecto original

Esperar a que se aprueben o rechacen los cambios

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

  Para enviar una solicitud de extracción (pull request) a un repositorio en GitHub, puedes seguir estos pasos:

  Ir a la página del repositorio en GitHub

  Seleccionar la rama que contiene los cambios

  Hacer clic en "Comparar y solicitar extracción"

  Elegir la rama base y la rama de comparación

  Escribir un título y una descripción

  Hacer clic en "Crear solicitud de incorporación de cambios"

  Si la solicitud es aceptada, se recibirá un correo electrónico.

  Para enviar una solicitud de extracción en AWS CodeCommit, se puede ejecutar el comando create-pull-request. En este comando se debe especificar:

  El nombre de la solicitud de extracción

  La descripción de la solicitud de extracción

  La ramificación de origen

  La ramificación de destino

  Un token de idempotencia único generado por el cliente

En Bitbucket, se puede crear una solicitud de incorporación de cambios yendo al repositorio bifurcado y haciendo clic en el botón "Pull request".

- ¿Cómo aceptar una solicitud de extracción?

  Para aceptar una solicitud de extracción, puedes revisar los cambios, dejar comentarios y hacer clic en el botón "Aprobar".

  Paso 1: Revisar los cambios

  1. Abre la solicitud de extracción
  2. Revisa los cambios de código en todos los archivos modificados
  3. Añade comentarios a la solicitud

  Paso 2: Dejar comentarios

  - Puedes añadir un comentario desde la sección "Comentarios"
  - Puedes añadir un comentario a nivel de archivo
  - Puedes hacer clic en el símbolo " + " a la izquierda de una línea de código
  - Sé amable y constructivo con todos tus comentarios
  - Proporciona detalles en los comentarios
  - Ofrece orientación con cambios que contribuyan al crecimiento

  Paso 3: Aprobar la solicitud

  1. Haz clic en el botón "Revisar cambios" en la pestaña "Archivos modificados"
  2. Selecciona "Aprobar" en las opciones
  3. Envía tu revisión

  Puedes revisar y aprobar solicitudes de extracción en GitHub y Bitbucket Cloud.

- ¿Qué es un etiqueta en Git?

  En Git, una etiqueta es una marca que se aplica a una confirmación de un proyecto. Sirve para identificar un punto importante en el historial del repositorio.

  Usos de las etiquetas Identificar una nueva versión del proyecto, Marcar un cambio significativo en el código, Indicar el punto de la versión beta, Publicar una versión marcada.

  Tipos de etiquetas

  Etiquetas anotadas

  Contienen metadatos y se pueden firmar para verificar. Son la práctica recomendada.

  Etiquetas ligeras

  Apuntan a confirmaciones específicas y no contienen otra información. También se conocen como etiquetas blandas.

Creación de etiquetas

Para crear una etiqueta, se utiliza el comando git tag.

Para enviar etiquetas a un servidor compartido, se utiliza el comando git push origin <tagname>.

Para enviar todas las etiquetas que aún no estén en el servidor remoto, se utiliza la opción --tags del comando git push.

Acceso a la versión etiquetada

Una vez creada una etiqueta, se puede acceder a la versión etiquetada del código base.

Se puede volver a esa versión específica si es necesario.

- ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta en Git, puedes usar el comando git tag.

Paso 1: Especifica el nombre de la etiqueta. Por ejemplo, git tag v1.4.

Paso 2: Para crear una etiqueta anotada, usa la opción -a y especifica el mensaje de la etiqueta con la opción -m. Por ejemplo, git tag -a v1.4 -m 'my version 1.4'.

Paso 3: Para crear una etiqueta ligera, no uses las opciones -a, -s ni -m. Por ejemplo, git tag v1.4-lw.

Paso 4: Para listar las etiquetas almacenadas en un repositorio, puedes utilizar el comando git tag.

Paso 5: Para obtener una lista más personalizada, puedes utilizar la opción -l junto con una expresión comodín.

Paso 6: Para ver la información de una etiqueta, puedes usar el comando git show. Por ejemplo, git show v1.4.

Paso 7: Para enviar los tags creados desde local al repositorio GitHub, puedes lanzar el push con la opción –tags.

- ¿Cómo enviar una etiqueta a GitHub?

Para enviar etiquetas a GitHub, puedes usar el comando git push con la opción –tags. También puedes crear etiquetas desde la interfaz web de GitHub.

Paso a paso para enviar etiquetas a GitHub usando git push

1. Usa el comando git push
2. Agrega la opción –tags

Paso a paso para crear etiquetas desde la interfaz web de GitHub

1. Ve al enlace “releases”
2. Ve a la página principal del repositorio

Consideraciones

- De forma predeterminada, git push no envía etiquetas
- Para enviar varias etiquetas al mismo tiempo, usa la opción –tags en el comando git push
- Cuando otro usuario clone un repositorio o incorpore cambios en él, recibirá las nuevas etiquetas
- Cualquiera con acceso de clasificación en un repositorio puede aplicar y descartar etiquetas
- En GitHub Desktop, puedes crear una etiqueta haciendo clic derecho en la confirmación y seleccionando "Crear etiqueta"

• ¿Qué es un historial de Git?

El historial de Git es un registro de los cambios realizados en los archivos de un proyecto. Se almacena como un gráfico de instantáneas, llamadas confirmaciones, que pueden tener varios padres.

Características del historial de Git

- Cada confirmación muestra la fecha, el usuario, el SHA de confirmación, y un enlace para navegar al archivo.
- Las confirmaciones pueden tener varios elementos primarios, creando un historial similar a un gráfico.
- El historial de Git permite volver al proyecto para ver quiénes son los autores, averiguar dónde se introdujeron los errores y revertir los cambios problemáticos.

Cómo navegar por el historial de Git

- El comando git log muestra confirmaciones.
- La opción -L de git log muestra el historial de una función o línea de código.
- Se pueden modificar los parámetros de git log para dar formato a la salida de las confirmaciones y filtrar qué confirmaciones se incluyen en la salida.

Git es un sistema de control de versiones (VCS) que reside en la máquina local del usuario.

• ¿Cómo ver el historial de Git?

Para ver el historial de Git, puedes usar el comando git log o la interfaz de usuario de GitLab o GitHub.

Con el comando git log

- git log muestra las confirmaciones de un repositorio en orden cronológico inverso
- git log --oneline muestra los primeros siete caracteres del hash SHA-1 y el mensaje de confirmación
- git log --oneline --graph muestra el historial de confirmaciones en un gráfico ASCII

En la interfaz de usuario de GitLab

1. Selecciona Buscar o encuentra tu proyecto en la barra lateral izquierda
2. Selecciona Código > Repositorio
3. Vaya al archivo deseado en el repositorio
4. Selecciona Historial en la esquina superior derecha

En la interfaz de usuario de GitHub

1. En la pestaña "Historial", haz clic en la confirmación que quieres revisar
2. Haz clic en un archivo individual para ver los cambios realizados en ese archivo

En GitHub Docs también puedes ver el historial de cambios de una wiki.

- ¿Cómo buscar en el historial de Git?

  Para buscar en el historial de Git, puedes utilizar los comandos git log, git grep, git rev-list y git show.

  Comandos para buscar en el historial de Git

  - git log -L Muestra el historial de una función o línea de código
  - git grep <pattern> $(git rev-list --all) Busca un patrón en todo el historial de confirmaciones
  - git log -p -G <pattern> Busca confirmaciones que introduzcan o eliminen un patrón
  - git log -p -S <string> Busca confirmaciones que agreguen o eliminen una cadena específica
  - git log --grep=<pattern> Busca un patrón en los mensajes de confirmación
  - git grep Busca a través de cualquier árbol o directorio de trabajo con commit por una cadena o expresión regular

  Otras formas de ver el historial de Git

  - En la interfaz de usuario de GitLab, puedes ver el historial de un archivo seleccionando Código > Repositorio y luego Historial
  - En GitHub, puedes ver un historial detallado de cambios en un repositorio usando la vista de actividad
  - En Warp, puedes usar la función de búsqueda de comandos de IA para verificar el historial de confirmaciones de Git

- ¿Cómo borrar el historial de Git?

  Para borrar el historial de Git localmente, puedes borrar la carpeta .git. Sin embargo, si ya has hecho un push al repositorio remoto, el historial permanecerá ahí.

  Para eliminar un archivo del historial de Git, puedes reescribir la historia de tu repositorio. Para ello, puedes utilizar herramientas como git filter-repo o BFG Repo-Cleaner.

Procedimiento para eliminar un archivo del historial de Git:

1. Encontrar la ubicación del archivo dentro del repositorio.
2. Recorrer cada branch y commit.
3. Eliminar el archivo en cada branch.
4. Actualizar tu repositorio remoto.

Consideraciones importantes

- Antes de modificar el historial del repositorio, es importante comprender las implicaciones.
- Se recomienda combinar o cerrar todas las solicitudes de incorporación de cambios abiertas antes de quitar archivos del repositorio.
- Después de eliminar los datos sensibles, debes subir forzadamente tus cambios a GitHub.

• ¿Qué es un repositorio privado en GitHub?

Un repositorio privado en GitHub es un espacio de almacenamiento que solo es accesible para el usuario que lo crea y los que él autorice.

Características de los repositorios privados de GitHub

- Los repositorios privados están cifrados en reposo y en vuelo.
- Los propietarios de organizaciones pueden restringir la capacidad de cambiar la visibilidad de los proyectos.
- Los administradores de proyectos pueden gestionar el acceso de escritura y administración a su proyecto.
- Los repositorios privados son gratuitos.

Cómo crear un repositorio en GitHub

Para crear un repositorio en GitHub, se puede:

1. Ir a https://github.com/new.
2. Seguir las instrucciones del Inicio rápido para repositorios.

Cómo invitar colaboradores a un repositorio

Para invitar colaboradores a un repositorio, se puede:

1. Navegar hasta la página principal del repositorio.
2. Hacer clic en Configuración en el nombre del repositorio.
3. Hacer clic en Colaboradores en la sección "Acceso" de la barra lateral.
4. Hacer clic en Agregar personas.
5. Comenzar a teclear el nombre de la persona que se desea invitar.

6. Hacer clic en Agregar NOMBRE al REPOSITORIO.

- ¿Cómo crear un repositorio privado en GitHub?

  Para crear un repositorio privado en GitHub, puedes seguir estos pasos:

  1. Ir a GitHub
  2. En la esquina superior derecha, seleccionar Nuevo repositorio
  3. Escribir un nombre para el repositorio
  4. Elegir la visibilidad del repositorio
  5. Seleccionar Privado
  6. Hacer clic en Crear repositorio

  También puedes crear un repositorio usando una consulta de dirección URL. Por ejemplo, https://github.com/new?name=test_repo&owner=avocado_corp crea un repositorio llamado "test_repo" propiedad de la organización "avocado_corp".

  Para cambiar la visibilidad de un repositorio, puedes:

  1. Ir a la página principal del repositorio
  2. Hacer clic en Configuración en el nombre del repositorio
  3. En la sección "Zona de peligro", hacer clic en Cambiar visibilidad
  4. Seleccionar una visibilidad

  Un repositorio es un lugar para almacenar código, archivos, y el historial de revisiones de cada archivo. Los repositorios pueden ser públicos, internos, o privados.

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

  Para invitar a alguien a un repositorio privado en GitHub, puedes seguir estos pasos:

  1. Ir a la página principal del repositorio
  2. Hacer clic en Configuración
  3. En la barra lateral, hacer clic en Colaboradores y equipos
  4. Hacer clic en Agregar personas
  5. Escribir el nombre de usuario, el nombre completo o la dirección de correo electrónico de la persona a la que quieres invitar
  6. Elegir el rol de repositorio que quieres conceder a la persona
  7. Hacer clic en Agregar NOMBRE al REPOSITORIO

- ¿Qué es un repositorio público en GitHub?

  Un repositorio público en GitHub es un espacio en la nube donde se puede almacenar y compartir código, archivos y revisiones de forma gratuita y accesible para todos.

Los repositorios públicos se suelen usar para compartir software de código abierto. Para que un repositorio sea de código abierto, se debe generar una licencia que permita a otras personas usar, modificar y distribuir el software.

Los repositorios pueden ser públicos o privados. Los repositorios privados solo son accesibles para el usuario y las personas con las que comparte acceso.

Para crear un repositorio en GitHub, se puede:

1. Ir a https://github.com/new
2. Escribir un nombre corto y fácil de recordar
3. Agregar una descripción del repositorio
4. Elegir la visibilidad del repositorio
5. Seleccionar "Initialize this repository with a README"
6. Hacer clic en "Create repository"

Para cambiar la visibilidad de un repositorio, se puede:

1. Ir a la página principal del repositorio
2. Hacer clic en "Configuración" en el nombre del repositorio
3. Hacer clic en "Cambiar visibilidad" en la sección "Zona de peligro"
4. Seleccionar una visibilidad

- ¿Cómo crear un repositorio público en GitHub?

  Para crear un repositorio público en GitHub, puedes:

  1. Ir a GitHub
  2. En la esquina superior derecha, seleccionar Nuevo repositorio
  3. Escribir un nombre para el repositorio
  4. Añadir una descripción opcional
  5. Elegir la visibilidad del repositorio, en este caso pública
  6. Seleccionar Inicializar este repositorio con un README
  7. Hacer clic en Crear repositorio

  También puedes crear un repositorio usando una consulta de dirección URL. Por ejemplo, https://github.com/new?name=test_repo&owner=avocado_corp crea un repositorio llamado "test_repo" propiedad de la organización "avocado_corp".

  Un repositorio es un lugar para almacenar código, archivos, y el historial de revisiones de cada archivo. Los repositorios pueden ser públicos, internos, o privados.

  Para confirmar tu primer cambio en el repositorio, puedes:

  1. Abrir el editor de archivos

2. Escribir información personal en el cuadro de texto
3. Revisar los cambios
4. Hacer clic en Confirmar cambios
5. Escribir un mensaje de confirmación
6. Decidir si agregar la confirmación a la rama actual o a una nueva rama

- ¿Cómo compartir un repositorio público en GitHub?

  Para compartir un repositorio público en GitHub, puedes invitar a colaboradores o compartir la URL del repositorio.

  Invitar colaboradores

  1. Ir a la página principal del repositorio
  2. Hacer clic en Configuración
  3. En la sección "Acceso", hacer clic en Colaboradores y equipos
  4. Hacer clic en Agregar personas o Agregar equipos
  5. Escribir el nombre de la persona o equipo a invitar
  6. Seleccionar el rol de repositorio que se desea otorgar
  7. Hacer clic en Agregar NOMBRE AL REPOSITORIO

  Compartir la URL del repositorio

  1. Ir al repositorio que se desea compartir
  2. Hacer clic en el botón verde Código
  3. Hacer clic en el ícono de copia de la URL web
  4. Pegar el enlace a un sitio web

  Los repositorios públicos de GitHub se utilizan para compartir software de código abierto. Para que un repositorio sea de código abierto, se debe generar una licencia.