**Alumno: Rodolfo Nicolás Velasco Fessler**

**Repositorio de GitHub:** [**https://github.com/RodolfoVelasco1/Lab1-ComB**](https://github.com/RodolfoVelasco1/Lab1-ComB)

**LABORATORIO 1 – TP2**

**Git & GitHub**

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

**1- ¿Qué es GitHub?:**

GitHub es un servicio basado en la nube que aloja un sistema de control de versiones (VCS) llamado Git. Éste permite a los desarrolladores colaborar y realizar cambios en proyectos compartidos, a la vez que mantienen un seguimiento detallado de su progreso

**2 -¿Cuáles son los beneficios de usar GitHub?**

Es gratuita para usar y contribuir, es fácil de usar ya que tiene una interfaz intuitiva, es popular, es flexible, es segura, te permite colaborar con otros desarrolladores, puedes rastrear cambios en su código, puede crear versiones de su código, GitHub hace que sea fácil implementar su código en producción, puedes crear y compartir páginas web estáticas con GitHub Pages. Facilita compartir tus proyectos de una forma mucho más fácil y crear un portafolio

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

Creamos un nuevo repositorio en https://github.com. Nos creamos una cuenta. Posteriormente para crear un repositorio: vamos a nuestro perfil, hacemos click en el botón verde que dice “New” (o “Nuevo”), le damos nombre, una descripción, seleccionamos si va a ser un proyecto público o privado, y un README en caso de ser necesario. Le damos a crear repositorio y con esto ya tenemos el repositorio donde alojaremos nuestro proyecto.

**4 - ¿Cuáles son los tipos de ramas en Git?:**

En git podemos encontrar dos tipos de ramas:

* Las ramas principales: entre las que encontramos la rama Master.
* Las ramas auxiliares: entre las que encontramos Feature, Release, Hotfix

**5 -¿Cómo crear una rama en Git?:**

Para crear una rama en git, utilizamos el comando “git branch nombre-rama”

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

Usamos el comando git checkout nombre-rama o bien git checkout -b nombre-rama nos permite crear una rama y a su vez pasar a dicha rama.

**7 -¿Cómo fusionar ramas en Git?**

La herramienta git merge se utiliza para fusionar uno o más ramas dentro de la rama que tienes activa. Por ejemplo: creamos una rama secundaria, para poder fusionarla a la rama principal debemos situarnos en la misma y fusionarla con la rama seleccionada.

En algunas ocasiones, los procesos de fusión no suelen ser fluidos. Si hay modificaciones dispares en una misma porción de un mismo archivo en las dos ramas distintas que pretenden fusionar, Git no será capaz de fusionarlas directamente.

**8 -¿Cómo crear un commit en Git?**

Para crear un commit en git, se utiliza el comando git commit -m “Primer commit”

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

Puedes enviar los commits a los servidores de GitHub con el comando "push".

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

Los repositorios remotos son versiones de tu proyecto que están hospedadas en Internet o en cualquier otra red. Puedes tener varios de ellos, y en cada uno tendrás generalmente permisos de solo lectura o de lectura y escritura. Colaborar con otras personas implica gestionar estos repositorios remotos enviando y trayendo datos de ellos cada vez que necesites compartir tu trabajo.

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

Para añadir un remoto nuevo y asociarlo a un nombre que puedas referenciar fácilmente, ejecuta git remote add [nombre] [url]. A partir de ahora puedes usar el nombre pb en la línea de comandos en lugar de la URL entera.

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

Para obtener datos de tus proyectos remotos puedes ejecutar: $ git fetch [remote-name] El comando irá al proyecto remoto y se traerá todos los datos que aún no tienes de dicho remoto. Luego de hacer esto, tendrás referencias a todas las ramas del remoto, las cuales puedes combinar e inspeccionar cuando quieras.

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

Si quieres ver más información acerca de un remoto en particular, puedes ejecutar el comando git remote show [nombre-remoto]. El comando lista la URL del repositorio remoto y la información del rastreo de ramas. El comando te indica claramente que, si estás en la rama maestra y ejecutas el comando git pull, automáticamente combinará la rama maestra remota con tu rama local, luego de haber traído toda la información de ella. También lista todas las referencias remotas de las que ha traído datos.

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

Básicamente sirve para crear una copia de un repositorio en tu cuenta de usuario. Ese repositorio copiado será básicamente un clon del repositorio desde el que se hace el fork, pero a partir de entonces el fork vivirá en un espacio diferente y podrá evolucionar de manera distinta

Lo podemos entender como una rama externa de un repositorio, colocando esa rama en un nuevo repositorio controlado por otros usuarios. Una vez hecho el fork existirán dos repositorios distintos. Inicialmente uno era copia exacta del otro, pero a medida que se vaya desarrollando y publicando cambios en uno u otro repo, ambos repositorios podrán tender a ser tan distintos como quieran cada uno de los equipos de desarrollo

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

Buscamos algún repositorio que nos parezca interesante y yo quisiera tener ese proyecto en mi repositorio debería de hacer un fork dando click en Fork situado en la parte superior derecha de github. Una vez hecho comenzará el forking

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

Una vez clonado el proyecto de otro usuario, realizamos cambios en nuestra copia local y los guardamos haciendo un push. Si queremos que dichos cambios se realicen en el proyecto original deberemos realizar un pull request.

Para hacer el pull request nos dirigiremos a la solapa de Pull requests allí daremos click en new pull request, veremos una ventana donde se reflejarán los cambios que hemos hecho nosotros en comparación al repositorio original (el código original). Daremos click en Create pull request donde veremos el asunto (colocamos algún mensaje global) y más abajo podemos mencionar el porqué ese cambio que hemos realizado nosotros, sería considerado como algo que a el repositorio original le vendría bien agregarlo.

El autor del repositorio verá en sus pull requests el mensaje que le hemos enviado, para que lo pueda observar y si lo considera realizar el cambio pertinente (además de poder responderle al usuario que le ha propuesto ese cambio).

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

Una vez realizada una pull request, si el usuario original considera que esta modificación es buena y no genera conflictos con la rama maestra de su repositorio local remoto, puede clickear en “Merge pull request” y de esta manera sumará a su repositorio los cambios que hizo un usuario (en modo de ayuda).

**18 -¿Qué es una etiqueta en Git?**

Git tiene la posibilidad de etiquetar puntos específicos del historial como importantes. Esta funcionalidad se usa típicamente para marcar versiones de lanzamiento (v1.0, por ejemplo)

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

Para crear una etiqueta usamos el comando git tag número-version esto nos permite ordenar nuestro proyecto, cuando añadamos funcionalidades nuevas, corregir errores o una versión que sea incompatible con la anterior.

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

Debes enviar las etiquetas de forma explícita al servidor luego de que las hayas creado. Este proceso es similar al de compartir ramas remotas - puedes ejecutar git push origin [etiqueta] .

**21 - ¿Qué es un historial de Git?**

El historial de Git es la secuencia de commits (confirmaciones) que se han realizado en la historia de un proyecto, lo que nos permite examinar los estados que ha tenido el repositorio a lo largo del tiempo. Podemos ver el historial de confirmaciones utilizando el comando "git log" en la terminal de Git.

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

La herramienta más básica y potente para hacer esto es el comando git log . Por defecto, si no pasas ningún parámetro, git log lista las confirmaciones hechas sobre ese repositorio en orden cronológico inverso.

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

git log nos permite conocer todo el historial de un proyecto, con la información de la fecha, el autor y id de cada cambio.

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

Si es necesario eliminar un commit específico, se puede utilizar el comando "git reset" con la opción "--hard" seguida del identificador del commit anterior al que se desea eliminar. Esto eliminará el commit y todos los cambios realizados en él, incluyendo los archivos agregados o modificados.

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

Un repositorio privado en GitHub es un espacio de almacenamiento en línea que permite a los usuarios almacenar y compartir su código de forma privada.

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

Cuando creas un repositorio, puedes elegir si lo quieres hacer público o privado, solo basta con elegir la opción “private” y solo será visible por el usuario.

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

1. Acceder al repositorio privado en GitHub.

2. Hacer clic en "Settings" en la parte superior derecha de la página.

3. Seleccionar "Manage access" en el menú de la izquierda.

4. Hacer clic en "Invite a collaborator" en la parte superior derecha de la página.

5. Ingresar el nombre de usuario o la dirección de correo electrónico de la persona que se desea invitar.

6. Seleccionar el nivel de acceso que se desea otorgar (read, write o admin).

7. Hacer clic en "Add [nombre de usuario o dirección de correo electrónico]" para enviar la invitación.

Una vez que la persona acepte la invitación, podrá acceder al repositorio privado y colaborar en el proyecto según el nivel de acceso que se le haya otorgado.

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

Los repositorios públicos son accesibles para todos los usuarios de Internet. Solo tú, las personas con las que compartes el acceso explícitamente

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

Cuando creas un repositorio, puedes elegir si lo quieres hacer público o privado, solo basta con elegir la opción “public” y será accesible para todo el mundo que quiera visualizar el contenido del mismo

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

En el nombre del repositorio, haz clic en Configuración. Si no puedes ver la pestaña "Configuración", selecciona el menú desplegable y, a continuación, haz clic en Configuración. En la sección "Acceso" de la barra lateral, haz clic en Colaboradores. Haz clic en Agregar personas.

**31 - ¿Qué es un repositorio de código en GitHub?**

Es el lugar en el que se almacena y se puede realizar la distribución del código de una aplicación o un programa. Este debe ser un servidor seguro que utiliza sistemas de control de versiones.

**32 - ¿Cómo crear un repositorio de código en GitHub?**

1. Iniciar sesión en GitHub y hacer clic en el botón "New repository".

2. Ingresar un nombre para el repositorio y una descripción opcional.

3. Seleccionar la opción "Public" o "Private" según las necesidades del proyecto.

4. Seleccionar la opción "Initialize this repository with a README" para crear un archivo README.md en el repositorio.

5. Seleccionar la opción "Add .gitignore" para agregar un archivo .gitignore que ignore los archivos y carpetas que no se deben incluir en el repositorio.

6. Seleccionar la opción "Add a license" para agregar una licencia al repositorio.

7. Hacer clic en el botón "Create repository" para crear el repositorio.

8. Clonar el repositorio en la máquina local utilizando el comando "git clone" seguido del enlace del repositorio.

9. Agregar los archivos y carpetas del proyecto al repositorio utilizando los comandos "git add", "git commit" y "git push".

10. Colaborar con otros usuarios en el proyecto utilizando las solicitudes de extracción (pull requests) y las ramas (branches) en GitHub.

**33 - ¿Cómo compartir un repositorio de código en GitHub?**

Para compartir el repositorio con otros usuarios, puedes agregarlos como colaboradores en la configuración del repositorio en GitHub. También puedes compartir la URL del repositorio para que otros puedan clonarlo y trabajar en él.

**34 - ¿Qué es un repositorio de documentación en GitHub?**

Un repositorio de documentación en GitHub está específicamente diseñado para almacenar y gestionar documentos, manuales, guías, tutoriales y cualquier otro tipo de contenido informativo relacionado con un proyecto. Estos repositorios están pensados para mantener una documentación clara y organizada que puede ser utilizada por los colaboradores del proyecto, usuarios finales u otras partes interesadas.

**35 - ¿Cómo crear un repositorio de documentación en GitHub?**

Para crear un repositorio de documentación, primero debemos crear un repositorio en github: vamos a nuestro perfil de GitHub, hacemos click en el botón verde que dice “New” (o “nuevo”). Le damos un nombre descriptivo como "mi-proyecto-documentation" y una breve descripción para indicar que este repositorio está destinado a la documentación. Luego escogemos si el repositorio es público o privado y damos click en “Crear Repositorio”.

**36 - ¿Cómo compartir un repositorio de documentación en GitHub?**

Para compartir un repositorio podemos:

* Invitar colaboradores: si deseas compartir el repositorio de documentación a otras personas para que colaboren en la documentación. Vamos a la pestaña "Settings" (Configuración) en el repositorio. Hacemos click en "Manage access" (Administrar acceso) y vamos a "Invite a collaborator" (Invitar a un colaborador). Ingresamos el nombre de usuario o la dirección de correo electrónico de la persona que deseamos invitar.
* Compartir el enlace: también podemos compartir el enlace del repositorio con otros para que puedan acceder a él. Abrimos nuestro repositorio de documentación en GitHub, copiamos el enlace y se lo enviamos a la persona con la que deseamos compartir el repositorio de documentación.

**37 - ¿Qué es un repositorio de proyecto en GitHub?**

Un repositorio de proyecto en GitHub es un espacio de almacenamiento en línea que permite a los usuarios almacenar y compartir su código de forma pública o privada. Un repositorio de proyecto puede contener todo el código fuente, archivos de configuración, documentación y otros recursos necesarios para el proyecto.

**38 - ¿Cómo usar Git para trabajar con otros?**

1. Crear un repositorio en GitHub y compartir el enlace con los colaboradores.

2. Cada colaborador debe clonar el repositorio en su máquina local utilizando el comando "git clone" seguido del enlace del repositorio.

3. Cada colaborador debe crear una rama local para trabajar en ella utilizando el comando "git checkout -b [nombre de la rama]".

4. Cada colaborador debe realizar los cambios necesarios en su rama local y confirmarlos utilizando el comando "git commit".

5. Cada colaborador debe enviar sus cambios al repositorio remoto utilizando el comando "git push".

6. El propietario del repositorio debe revisar los cambios y fusionarlos en la rama principal utilizando el comando "git merge".

7. Si hay conflictos, se deben resolver antes de fusionar las ramas.

8. Los colaboradores deben actualizar su rama local con los cambios fusionados utilizando el comando "git pull".

Este flujo de trabajo permite a los colaboradores trabajar en paralelo en diferentes ramas y fusionar sus cambios de forma segura en el repositorio principal. También permite mantener un historial de versiones y revertir cambios si es necesario.

**2- Realizar la siguiente actividad:**

**Crear un repositorio.**

o Dale un nombre al repositorio.

o Elije el repositorio sea público.

o Inicializa el repositorio con un archivo.

**Agregando un Archivo**

o Crea un archivo simple, por ejemplo, "mi-archivo.txt".

o Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.

o Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

**Creando Branchs**

o Crear una Branch

o Realizar cambios o agregar un archivo

o Subir la Branch

**Entrega**:

* Responder las preguntas en el documento de Word.
* Colocar Apellido y Nombre.
* Copiar el enlace al proyecto de GitHub.

**Importante**: La entrega de este trabajo se realizará de forma individual.