Tecnicatura Superior en Programación 

UTN-FRM

Laboratorio de Computación I – Git & GitHub

* **Git & GitHub**

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

1)¿Qué es GitHub?

GitHub es un servicio donde se pueden alojar repositorios de control de versiones, donde se puede almacenar y gestionar proyectos de software. Es gratuito y de código abierto que permite a los usuarios trabajar juntos en proyectos, compartir código y colaborar de forma eficiente.

2)¿Cuáles son los beneficios de usar GitHub?

Los beneficios de usar GitHub son:

1. Facilidad de uso: es muy fácil de usar, sobre todo para desarrolladores que no conocen esta herramienta de control de versiones.
2. Escalabilidad: puede escalar para satisfacer las necesidades de cualquier proyecto, desde personales hasta de grandes empresas.
3. Seguridad: ofrece protección para los proyectos de los usuarios.
4. Comunidad: cuenta con una gran comunidad de desarrolladores que colaboran entre ellos.

3) ¿Cómo crear un repositorio en GitHub?

Un repositorio es un contenedor para tu código. Cuando se crea un repositorio, se crea una copia del código en los servidores de GitHub. Se puede clonar el repositorio en el ordenador para trabajar en él localmente.

Una vez clonado el repositorio, se puede empezar a trabajar en el código. Se puede hacer cambios en el código y guardarlos como "commits". Se pueden enviar los commits a GitHub con el comando "push". También se puede recibir los cambios que otros han hecho en el código con el comando "pull".

4)¿Cuáles son los tipos de ramas en Git?

Se utilizan básicamente 5 ramas : Master, Desarrollo, QA, Features, y Hotfix.

1. Master: es la rama principal. Contiene el repositorio de lo que se encuentra en producción.
2. Desarrollo: Es una rama sacada del master. Es una rama de integración. Luego que se realice la integración y se corrijan los errores (en caso de que haya alguno), luego de que se encuentre estable, se puede hacer un merge de desarrollo sobre la rama QA.
3. Features: cada nueva funcionalidad se debe realizar en una rama nueva específica para esa funcionalidad. Sale de la rama Desarrollo. Una vez que la funcionalidad esté desarrollada, se hace un merge sobre la rama de Desarrollo, donde se integrará con el resto de las funcionalidades.
4. QA: es una rama sacada del desarrollo que fue probado y que pasa a QA para que el usuario final realice la prueba de aceptación.
5. Hotfix: son bugs que surgen en producción, por lo que se deben arreglar e implementar en producción de forma urgente. Por eso son ramas sacadas de Master. Una vez corregido el error, se debe hacer un merge (une las ramas), sobre el Master. Luego, para que no quede desactualizada la versión, se debe hacer un merge sobre QA y luego sobre Desarrollo.

5) ¿Cómo crear una rama en Git?

Para crear una rama se utiliza el comando ”git branch” seguido del nombre de la nueva rama. Por ejemplo **git branch nombre\_de\_tu\_branch**

6) ¿Cómo cambiar a una rama en Git?

Para moverse entre ramas usamos el comando “git checkout” seguido del nombre de la rama que queremos que sea la activa.

**git checkout nombre\_de\_tu\_branch**.

7)¿Cómo fusionar ramas en Git?

Para fusionar ramas se utiliza el comando “git merge” seguido del nombre de la rama que se desea fusionar.

**git merge nombre\_de\_tu\_branch**.

8)¿Cómo crear un commit en Git?

Para crear un commit se utiliza el comando **git commit -m “mensaje del comentario”.** Este comando guarda los cambios que hayas realizado y los añade al historial de versiones del proyecto con el mensaje especificado.

9)¿Cómo enviar un commit a GitHub?

Para enviar un commit a GitHub, primero se debe agregar el repositorio remoto utilizando.

**git remote add origin URL\_del\_repositorio**.

Luego utiliza **git push origin nombre\_de\_la\_rama**; para enviar los commits de una rama específica al repositorio remoto en GitHub.

10) ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión de un repositorio de código almacenado en un servidor en línea, GitHub es uno, que permite a los desarrolladores colaborar y compartir sus códigos de trabajo.

11) ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto se utiliza el comando **git remote add nombre\_del\_repositorio URL\_repositorio**.

12)¿Cómo empujar cambios a un repositorio remoto?

Se puede usar el comando “git push” seguido del nombre del repositorio remoto y la rama que se desea empujar los cambios

13)¿Cómo tirar de cambios de un repositorio remoto?

Se puede usar el comando “git pull”, seguido del nombre del repositorio remoto y la rama que se desea traer al repositorio local.

14) ¿Qué es un fork de repositorio?

Un fork de un repositorio es una copia independiente de un repositorio en la cuenta de GitHub. Permite realizar cambios en el repositorio copiado sin afectar al repositorio original.

15)¿Cómo crear un fork de un repositorio?

Para crear un fork de un repositorio hay que visitar el repositorio que se desea copiar, luego hay que hacer click en el botón “Fork” en la esquina superior derecha de la página. Esto creará una copia del repositorio en la cuenta propia que uno mismo tiene en GitHub.

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

Para enviar una solicitud de extracción (pull request) a un repositorio se debe visitar el repositorio original y hacer click en la pestaña “Pull requests”. Luego, hacer click en el botón “New pull request” y seguir los pasos para seleccionar la rama y los cambios que se desea enviar. Por último, se hace click en “Create pull request” para enviar la solicitud al dueño del repositorio.

17) ¿Cómo aceptar una solicitud de extracción?

Para aceptar una solicitud de extracción se debe ir al repositorio en cuestión y hacer click en la pestaña “Pull request”. Luego seleccionar la solicitud de extracción que se desee aceptar y revisar los cambios propuestos. Si uno está de acuerdo, se hace click en el botón “Merge” para fusionar los cambios en el repositorio. Finalmente se confirma la fusión y se cierra la solicitud de extracción si así se desea.

18) ¿Qué es una etiqueta en Git?

En Git, una etiqueta o tag es una forma de marcar un punto específico en la historia del repositorio. Las etiquetas se utilizan típicamente para marcar versiones o hitos importantes en el desarrollo del proyecto.

19)¿Cómo crear una etiqueta en Git?

Para crear una etiqueta se puede usar el comando “git tag” seguido de un nombre para la etiqueta y el commit al que se desea asociar. Por ejemplo, se puede usar **“git tag -a nombre de la etiqueta -m ‘Mensaje de la etiqueta’ commit\_id”** para crear una etiqueta anotada. También se puede usar **“git tag nombre\_etiqueta commit\_id”** para crear una etiqueta ligera.

20) ¿Cómo enviar una etiqueta a GitHub?

Se puede usar el comando “git push” seguido del nombre del repositorio remoto y el nombre de la etiqueta que se desea enviar. Por ejemplo, se puede usar **“git push repst nombre\_etiqueta”** para enviar una etiqueta al repositorio remoto llamado ‘repst’.

21)¿Qué es un historial de Git?

Es una secuencia de instantáneas (commits) que representan el estado de un repositorio en diferentes momentos del tiempo. Cada instantánea contiene los cambios realizados en el repositorio desde la última instantánea, lo que permite rastrear la evolución del proyecto a lo largo del tiempo.

22) ¿Cómo ver el historial de Git?

Se puede usar el comando “git log”. Mostrará una lista de todos los commits en orden inverso cronológicamente, es decir, desde el mas reciente. Se pueden usar varias opciones para personalizar la salida, como ‘--oneline’ para mostrar cada commit en una sola línea o ‘--author’ para mostrar solo los commits realizados por un autor especifico.

23) ¿Cómo buscar en el historial de Git?

Se puede utilizar el comando “git log” junto con la opción ‘--grep’ seguida de una palabra clave. Esto filtrara los commits que coincidan con la búsqueda especificada en el mensaje del commit. Por ejemplo: se puede usar **“git log –grep=’palabra clave’ ”.**

24) ¿Cómo borrar el historial de Git?

Borrar todo el historial de Git no es recomendable, ya que puede causar problemas en el repositorio. Si se desea borrar un commit específico, se puede usar el comando “git revert” para crear un nuevo commit que deshaga los cambios realizados en el commit original. Si se necesita eliminar todo el historial de un repositorio, se puede crear un nuevo repositorio vacío y empujar los archivos relevantes a él. Sin embargo, esto eliminará todo el historial del repositorio original.

25) ¿Qué es un repositorio privado en GitHub?

Es un espacio de almacenamiento en línea donde se puede guardar y gestionar un código de forma segura, limitando el acceso solo a las personas autorizadas.

26) ¿Cómo crear un repositorio privado en GitHub?

Primero hay que iniciar sesión en nuestra cuenta, hacer click en el botón "+", seleccionar "Nuevo repositorio", ingresar el nombre y la descripción, elegir la opción "Privado" y hacer click en "Crear repositorio".

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

Hay que ir al repositorio, hacer click en la pestaña "Configuración", seleccionar "Colaboradores" en el menú izquierdo, ingresar el nombre de usuario o la dirección de correo electrónico de la persona que se desea invitar y hacer click en "Agregar colaborador".

28) ¿Qué es un repositorio público en GitHub?

Es un espacio de almacenamiento en línea donde se puede guardar y gestionar código de forma gratuita y accesible al público en general. Cualquier persona puede ver, descargar y contribuir al código almacenado en un repositorio público.

29) ¿Cómo crear un repositorio público en GitHub?

Son los mismos pasos que uno hace para crear un repositorio privado, hay que iniciar sesión en nuestra cuenta, hacer click en el botón "+" en la esquina superior derecha del perfil y seleccionar "New repository". Luego, establecer un nombre, marcar la opción "Public" y hacer click en "Create repository".

30) ¿Cómo compartir un repositorio público en GitHub?

Simplemente se debe compartir el enlace del repositorio con otras personas. También se puede hacer click en el botón "Fork" en la página del repositorio para crear una copia personalizada en su propia cuenta de GitHub.

31) ¿Qué es un repositorio de código en GitHub?

Es un espacio en el que se almacena y gestiona el código fuente de un proyecto. Permite a los desarrolladores colaborar, realizar seguimiento de cambios, revisar y fusionar código, y llevar un historial de versiones.

32) ¿Cómo crear un repositorio de código en GitHub?

Para crear un repositorio de código en GitHub, se debe iniciar sesión en la cuenta, hacer click en el botón "+" en la esquina superior derecha del perfil y seleccionar "New repository". Luego, establecer un nombre para el repositorio, agregar una descripción opcional, marcar la opción "Public" o "Private", y hacer click en "Create repository".

33)¿Cómo compartir un repositorio de código en GitHub?

Simplemente se comparte el enlace del repositorio con otras personas. También se pueden agregar colaboradores al repositorio para que contribuyan al proyecto. Para hacerlo, hay que ir a la pestaña "Settings" del repositorio, seleccionar "Collaborators" en el menú de la izquierda y agregar los nombres de usuario de los colaboradores.

34) ¿Qué es un repositorio de documentación en GitHub?

Es un espacio donde se almacena y gestiona la documentación de un proyecto. Puede incluir guías, tutoriales, manuales, especificaciones técnicas y cualquier otro tipo de documentación relacionada con el proyecto. Permite a los colaboradores contribuir, revisar y mantener actualizada la documentación de manera colaborativa.

35) ¿Cómo crear un repositorio de documentación en GitHub?

Hay que iniciar sesión en la cuenta, hacer click en el botón "+" en la esquina superior derecha del perfil y seleccionar "New repository". Luego, establecer un nombre para el repositorio de documentación, agregar una descripción opcional y marcar la opción "Public" o "Private". Además, se puede seleccionar la opción "Initialize this repository with a README" para crear un archivo README.md inicial que se puede editar con la documentación. Finalmente, hacer click en "Create repository".

36)¿Cómo compartir un repositorio de documentación en GitHub?

Solo hay que compartir el enlace del repositorio con otras personas. También se pueden agregar colaboradores al repositorio para que puedan contribuir y editar la documentación. También se pueden utilizar las funciones de control de versiones y solicitudes de extracción para llevar un seguimiento de los cambios y revisar las actualizaciones realizadas por los colaboradores.

37) ¿Qué es un repositorio de proyecto en GitHub?

Es donde se almacena y gestiona el código fuente, la documentación y cualquier otro recurso relacionado con un proyecto. Permite a los desarrolladores trabajar en equipo, llevar un historial de versiones, realizar seguimiento de problemas y solicitudes de extracción, y colaborar en la construcción del proyecto.

38) ¿Cómo usar Git para trabajar con otros?

Para usar Git y colaborar con otros en un proyecto, primero hay que clonar el repositorio en nuestra máquina local usando el comando "git clone [URL del repositorio]". Luego, realizar cambios en los archivos y utilizar los comandos "git add" para agregar los cambios, "git commit" para confirmarlos y "git push" para enviarlos al repositorio remoto. Para incorporar los cambios de otros colaboradores, se utiliza "git pull" para actualizar el repositorio local. Si hay conflictos, se resuelven y se realiza un nuevo commit.

**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.

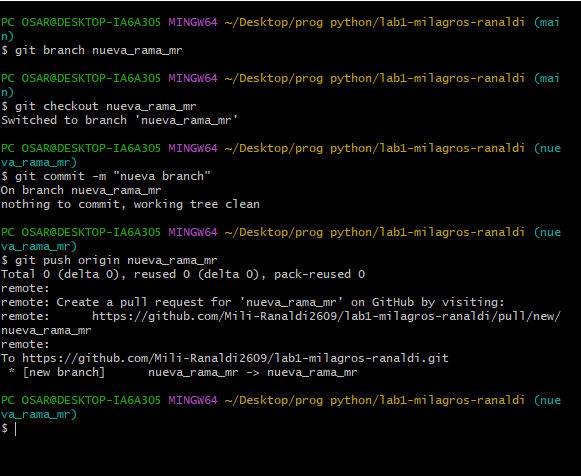
o Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

• 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.