**Git & GitHub**

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

* + ¿Qué es GitHub?

GitHub es una página web que trabaja con Git que ofrece el servicio de alojar proyectos y repositorios en la web además de tener el propio sistema de control de versiones de Git.

* + ¿Cuáles son los beneficios de usar GitHub?

Los beneficios de GitHub son que puedas acceder a cualquier proyecto que este subido en la web desde cualquier computador o maquina con acceso a internet y por cualquier persona que desee acceder al proyecto, repositorio, etc. Que este subido en GitHub, además, la misma pagina web tiene sus funcionalidades como revisión de código, arreglar errores menores en el proyecto, la creación de un repositorio, crear ramas y casi todas las funcionalidades de Git desde la propia pagina web para facilitar proyectos cooperativos.

* + ¿Cómo crear un repositorio en GitHub?

En la pagina web de GitHub en el apartado de home del perfil debemos tocar el botón el de new, esto abrirá una pestaña que nos pedirá la información que deseamos que contenga el repositorio, tanto como nombre, descripciones del proyecto y si será un proyecto publico a todo el mundo o solo para el creador del mismo, una vez configurado todo se debe tocar en la parte inferior derecha de la pestaña el botón de créate repository, una vez presionado el botón el proyecto ya estará alojado en la pagina web.

* + ¿Cuáles son los tipos de ramas en Git?

Principalmente en cada proyecto pueden haber 6 tipos de ramas en el proyecto, la primera rama “main” o “master” es por lo general donde se encontrara el proyecto que este funcionando o se este utilizando terminado; la segunda rama es de desarrollo en la que cada rama de desarrollo que se cree será para agregar funcionalidades o características nuevas al proyecto que ya este funcionando en main pero sin afectar al mismo en caso de que surjan errores en la etapa de desarrollo, también dentro de estas ramas de desarrollo se pueden generar otro tipo de ramas que sea una rama diferente para cada característica o funcionalidad que luego se incorporaran todas juntas en la rama de desarrollo; el cuarto tipo de rama llamado “bugfix” se crea para la corrección de errores o eliminación de bug, como su nombre indica del proyecto, de la rama main o principal o de la rama de desarrollo también; el quinto tipo de rama son llamadas “release” son aquellas ramas con el fin de unificar e incorporar la rama desarrollo con la rama main para probar o corregir errores finales antes de incorporar directamente en la rama main; el sexto y último tipo es la rama “experimental” que es una rama similar a la de desarrollo, con el fin de agregar funcionalidades al proyecto pero esta vez con apuntes y enfoques diferentes del proyecto principal, a diferencia de la rama de desarrollo en esta rama se prueban cosas nuevas y diferentes a las del proyecto que no necesariamente tienen o serán incorporadas en el proyecto final a no que dicha funcionalidad o característica.

* + ¿Cómo crear una rama en Git?

Para crear una rama en Git se utilizará el comando por consola git Branch “nombre de la rama”

* + ¿Cómo cambiar a una rama en Git?
  + ¿Cómo fusionar ramas en Git?
  + ¿Cómo crear un commit en Git?
  + ¿Cómo enviar un commit a GitHub?
  + ¿Qué es un repositorio remoto?
  + ¿Cómo agregar un repositorio remoto a Git?
  + ¿Cómo empujar cambios a un repositorio remoto?
  + ¿Cómo tirar de cambios de un repositorio remoto?
  + ¿Qué es un fork de repositorio?
  + ¿Cómo crear un fork de un repositorio?
  + ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
  + ¿Cómo aceptar una solicitud de extracción?
  + ¿Qué es un etiqueta en Git?
  + ¿Cómo crear una etiqueta en Git?
  + ¿Cómo enviar una etiqueta a GitHub?
  + ¿Qué es un historial de Git?
  + ¿Cómo ver el historial de Git?
  + ¿Cómo buscar en el historial de Git?
  + ¿Cómo borrar el historial de Git?

* + ¿Qué es un repositorio privado en GitHub?
  + ¿Cómo crear un repositorio privado en GitHub?
  + ¿Cómo invitar a alguien a un repositorio privado en GitHub?
  + ¿Qué es un repositorio público en GitHub?
  + ¿Cómo crear un repositorio público en GitHub?
  + ¿Cómo compartir un repositorio público en GitHub?
  + ¿Qué es un repositorio de código en GitHub?
  + ¿Cómo crear un repositorio de código en GitHub?
  + ¿Cómo compartir un repositorio de código en GitHub?
  + ¿Qué es un repositorio de documentación en GitHub?
  + ¿Cómo crear un repositorio de documentación en GitHub?
  + ¿Cómo compartir un repositorio de documentación en GitHub?
  + ¿Qué es un repositorio de proyecto en GitHub?
  + ¿Cómo usar Git para trabajar con otros?

1. **Realizar la siguiente actividad:**

* + Crear un repositorio.

* + - Dale un nombre al repositorio. o Elije el repositorio sea público.
    - Inicializa el repositorio con un archivo.

* + Agregando un Archivo

* + - Crea un archivo simple, por ejemplo, "mi-archivo.txt".
    - Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
    - Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

* + Creando Branchs

* + - Crear una Branch o Realizar cambios o agregar un archivo o Subir la Branch

1. Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

* + Ve a GitHub e inicia sesión en tu cuenta.
  + Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
  + Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
  + Opcionalmente, añade una descripción.
  + Marca la opción "Initialize this repository with a README".
  + Haz clic en "Create repository".

Paso 2: Clonar el repositorio a tu máquina local

* + Copia la URL del repositorio (usualmente algo como https://github.com/tuusuario/conflict-exercise.git).
  + Abre la terminal o línea de comandos en tu máquina.
  + Clona el repositorio usando el comando: git clone https://github.com/tuusuario/conflict-exercise.git
  + Entra en el directorio del repositorio: cd conflict-exercise

Paso 3: Crear una nueva rama y editar un archivo

* + Crea una nueva rama llamada feature-branch:

git checkout -b feature-branch

* + Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:

Este es un cambio en la feature branch.

* + Guarda los cambios y haz un commit: git add README.md git commit -m "Added a line in feature-branch"

Paso 4: Volver a la rama principal y editar el mismo archivo

* + Cambia de vuelta a la rama principal (main):

git checkout main

* + Edita el archivo README.md de nuevo, añadiendo una línea diferente:

Este es un cambio en la main branch.

* + Guarda los cambios y haz un commit: git add README.md git commit -m "Added a line in main branch"

Paso 5: Hacer un merge y generar un conflicto

* + Intenta hacer un merge de la feature-branch en la rama main:

git merge feature-branch

* + Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

Paso 6: Resolver el conflicto

* + Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

Copiar código

<<<<<<< HEAD

Este es un cambio en la main branch.

=======

Este es un cambio en la feature branch.

>>>>>>> feature-branch

* + Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
  + Edita el archivo para resolver el conflicto y guarda los cambios.
  + Añade el archivo resuelto y completa el merge:

git add README.md git commit -m "Resolved merge conflict"

Paso 7: Subir los cambios a GitHub

* + Sube los cambios de la rama main al repositorio remoto en GitHub:

git push origin main

* + También sube la feature-branch si deseas:

git push origin feature-branch

Paso 8: Verificar en GitHub

* + Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
  + Puedes revisar el historial de commits para ver el conflicto y su resolución.

**Entrega**:

* + Responder las preguntas en el documento de Word.

* + Colocar Apellido y Nombre.

* + Copiar los enlaces al proyecto de GitHub.

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