Actividad 2.3: supuesto práctico para

el uso de Git y Github

**Supuesto Práctico Complejo: Proyecto Colaborativo en GitHub**

Imagina que eres parte de un equipo de desarrollo que trabaja en un proyecto de software

en GitHub. El proyecto está en constante evolución, y cada miembro del equipo es

responsable de diferentes funcionalidades. A lo largo de este ejercicio, te enfrentarás a

varias situaciones en las que necesitarás aplicar múltiples comandos y estrategias para

gestionar de manera efectiva el proyecto.

**Escenario:**

Tu equipo está trabajando en una aplicación web llamada **"Task Manager"**, que permite a

los usuarios gestionar tareas y proyectos. Se te ha asignado la tarea de desarrollar una

**nueva funcionalidad** para la aplicación, la cual consiste en **añadir un sistema de**

**notificaciones** que informe a los usuarios cuando una tarea está a punto de vencer.

El proyecto está alojado en GitHub en un **repositorio remoto** al cual todos los

desarrolladores contribuyen. Antes de comenzar a trabajar, debes asegurarte de que tu

entorno local está sincronizado con el último código y seguir un flujo de trabajo adecuado

para evitar conflictos con el trabajo de tus compañeros.

**Tareas que Debes Completar:**

**1. Clonar el Repositorio del Proyecto**

El proyecto está alojado en GitHub, y aún no tienes una copia local del mismo. El primer

paso es clonar el repositorio remoto a tu máquina para comenzar a trabajar.

● **Instrucciones**:

○ Clona el repositorio remoto llamado task-manager a tu equipo.

**2. Crear una Nueva Rama para Desarrollar la Funcionalidad**

Para no afectar la rama principal (main), necesitas crear una nueva rama donde puedas

desarrollar la funcionalidad de notificaciones de manera aislada.

● **Instrucciones**:

○ Crea una nueva rama llamada notificaciones.

○ Cámbiate a esa rama y empieza a trabajar desde allí.

**3. Realizar Cambios en el Código**

Realiza los cambios necesarios en el código para implementar el sistema de notificaciones.

A medida que desarrolles la funcionalidad, haz commits pequeños y descriptivos para

registrar tu progreso.

● **Instrucciones**:

○ Realiza al menos dos cambios importantes en los archivos del proyecto

relacionados con las notificaciones. Por ejemplo, haz cambios en el archivo

notificaciones.js y archivo.html

○ Añade estos cambios al área de preparación (staging area) y realiza un

commit con un mensaje descriptivo.

■ Haz un commit diferente para cada uno de los archivos modificados

**4. Sincronizar los Cambios con el Repositorio Remoto**

Mientras estabas trabajando en tu funcionalidad, el equipo realizó actualizaciones

importantes en la rama principal. Antes de subir tus cambios, es crucial que integres las

últimas actualizaciones para evitar conflictos.

● **Instrucciones**:

○ Cambia a la rama main y descarga las actualizaciones más recientes del

repositorio remoto.

○ Integra estos cambios en tu rama notificaciones.

**5. Resolver Conflictos de Fusión**

Al fusionar los cambios de la rama main en tu rama notificaciones, Git detecta un

conflicto en el archivo notificaciones.js, ya que otro desarrollador también trabajó en

una funcionalidad relacionada. Tu tarea es resolver este conflicto.

● **Instrucciones**:

○ Identifica el conflicto y edita manualmente el archivo notificaciones.js

para resolver el problema.

○ Añade el archivo al staging area y realiza un commit con la resolución del

conflicto.

**6. Guardar Cambios Temporales**

Estás a punto de finalizar tu trabajo en la funcionalidad de notificaciones cuando recibes una

solicitud urgente para corregir un error en la rama main. No quieres perder tus cambios

actuales, pero tampoco estás listo para hacer un commit.

● **Instrucciones**:

○ Guarda temporalmente tus cambios en la rama notificaciones sin hacer

un commit.

○ Cambia a la rama main para corregir el error.

**7. Corregir un Bug en la Rama Principal**

Una vez en la rama main, realiza los cambios necesarios para corregir el error que fue

reportado. Asegúrate de realizar un commit con un mensaje claro y descriptivo. El archivo

sobre el que corregir el bug es botones.html

● **Instrucciones**:

○ Realiza la corrección del bug y haz un commit con un mensaje que explique

qué has corregido.

**8. Volver a tu Trabajo en la Funcionalidad de Notificaciones**

Después de corregir el bug en main, necesitas retomar tu trabajo en la funcionalidad de

notificaciones. Para hacerlo, debes volver a la rama notificaciones y restaurar los

cambios que guardaste temporalmente.

● **Instrucciones**:

○ Cambia a la rama notificaciones y recupera los cambios que guardaste

previamente.

**9. Finalizar la Funcionalidad y Subirla al Repositorio Remoto**

Una vez que hayas completado el desarrollo de la funcionalidad de notificaciones, debes

subir tu rama al repositorio remoto en GitHub para que el equipo pueda revisarla.

● **Instrucciones**:

○ Sube la rama notificaciones al repositorio remoto.

**10. Revisar y Fusionar el Pull Request**

Tu equipo ha revisado tus cambios y ha aceptado integrarlas en main. Procede a fusionar la

rama notificaciones con main.

● **Instrucciones**:

○ Fusiona la rama notificaciones con main y **elimina** la rama una vez que

los cambios se hayan incorporado.