Informe del grupo

Integrantes: Armando Pérez, Gabriel Padilla y Nicely Meza

1. Pasos realizados

- creamos el repositorio con el nombre "app" en GitHub. se inicializó con README.md y gitignore.
- Se invitó a los demás colaboradores del grupo.
- Se configuró el proyecto creando el main.py y el task_model.py. Luego, se le hizo commit y push.
- Se crearon ramas individuales para cada estudiante.
- Cada uno implementó funcionalidades y resolvió conflictos. Todo esto a través de una rama grupal en la cual se fusionaron las demás ramas.
- Luego, se fusionó la rama grupo con el main.
- Por último, se eliminaron las ramas obsoletas.

2. Comandos Git:

- git clone <URL>: Clonar el repositorio
- git checkout -b estudiante1: Crear rama.
- git add <archivo>: añadir cambios.
- git commit -m "mensaje": Hacer commit.
- git push origin estudiante1: Empujar rama.
- git merge estudiante3: Fusionar rama con conflicto

- git push origin -delete estudiante1: Eliminar rama.

3. Conflictos:

- Conflicto en task_model.py entre is_completed/is_done y mark_as_complete / set_done.
- Solución: Combinamos métodos, manteniendo is_completed como atributo principal.

4. Contribuciones:

- Estudiante 1: Añadió mark_as_complete y actualizó main.py.
- Estudiante 2: Añadió delete_task y actualizó README.md.
- Estudiante 3: Simuló conflicto con set_done/remove_task y lo resolvió.

5. Reflexiones

- Si no se comunica de antemano en qué parte del proyecto estamos trabajando, podemos ocasionar un conflicto con alguien del equipo.
- aprendimos a crear ramas y coordinarlas
- se agiliza mucho el desarrollo colaborativo.
- coordinamos el trabajo en equipo comunicando la parte del proyecto sobre la que trabajaremos.
- Es importante usar pull Requests para asegurarse que no hayan conflictos y se dañe lo ya hecho.

- Aprendimos igualmente que los conflictos no son errores de Git sino que son señales que necesitan una decisión del equipo y es importante entender qué cambio es conveniente para el proyecto.
- El flujo de trabajo podría mejorarse con herramientas como CI/CD en la verificación automática del código cada vez que se hace un push o un pull detectando errores con anticipación antes de fusionar los cambios, lo que en consecuencia haría el trabajo mucho más rápido.