👣 Crear y modificar ramas

En este paso, vamos a avanzar en el uso de las ramas para lograr un correcto flujo de trabajo de git. Recomendamos que una persona de la mesa de trabajo pueda compartir pantalla para que lo vayan resolviendo en equipo.

> 💡 ***Necesitamos tener abiertos Visual Studio Code y la consola de comandos en la carpeta "profile".***

📝 Actividad: Crear la rama develop y features

Para trabajar con el flujo de trabajo que propone git, debes:

- Partir desde la rama principal (main o master) y crea la rama develop
- Esta es la rama a la cual se fusionarán todas las vistas o funcionalidades desarrolladas.
- Desde la rama develop, crea las ramas de trabajo.
- Las ramas de trabajo serán las encargadas de desarrollar los avances del proyecto.

Comencemos con la creación de todas estas ramas:

1. En la terminal, verifica la rama principal actual con git branch. Recuerda que puede ser **main** o **master**.

2. Verifica estados con git status

3. Si no hay nada para commitear, ejecuta en la consola:

```
git checkout -b develop
```

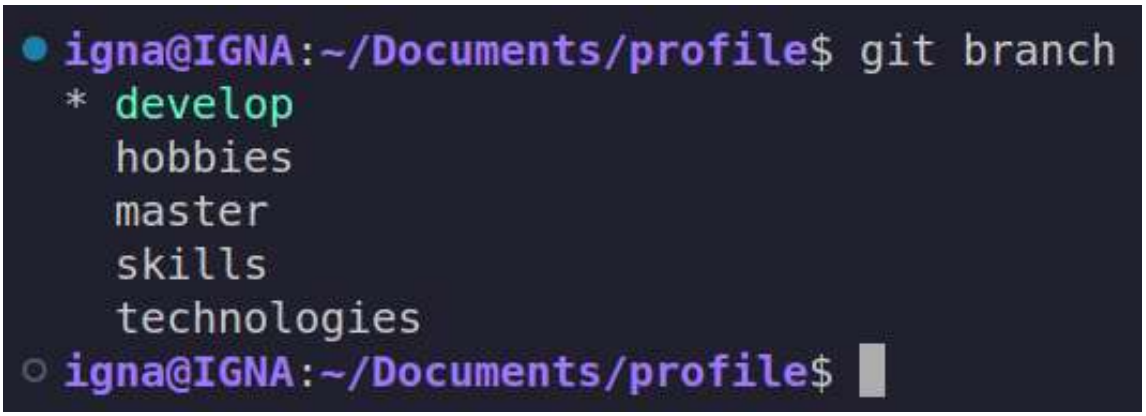
Este comando crea una rama exactamente igual a la principal main/master y automáticamente se mueve a la rama "develop".

La consola debería informar la correcta creación y cambio de rama con el mensaje: "Switched to a new branch 'develop'".

4. Desde esta rama hay que crear las ramas de trabajo de los próximos agregados del perfil. Vamos a crear 3 ramas con:

```
git branch technologies
git branch skills
git branch hobbies
git branch family
```

5. Verifica la creación de las ramas ejecutando git branch.



La consola me informa la rama en la que estoy trabajando con un color diferente. En este caso seguimos en la rama “develop”

6. Muévete a la rama de trabajo ejecutando git checkout technologies y define la sección “tecnologías”. A modo de ejemplo:

```
## Technologies
- Back-End: Proficient in Java, I bring to the table a strong understanding of server-sid
- Front-End: My expertise in Angular allows me to create interactive and user-friendly in
```

7. Prepara y “comittea” el archivo con los comandos correspondientes.

8. El siguiente paso sería subir los cambios, pero lo vamos a dejar para más adelante. Por ahora, cambia de rama para continuar con el desarrollo con **git checkout develop**

9. Ahora muévete a la otra rama de trabajo ejecutando **git checkout skills**

> ***Observa que se “borraron” las modificaciones realizadas en “technologies”. Cada rama de trabajo tiene avances de trabajo independientes. En un equipo real de trabajo, cada programador trabaja en el desarrollo de una rama de trabajo y luego las comparte para su corrección, aprobación e integración!***

Define la sección “habilidades”. A modo de ejemplo:

```
## Skills
- Problem-Solving: As a developer, I thrive on solving complex problems. I’m constantly e
```

10. Prepara y “comittea” el archivo con los comandos correspondientes.

11. Cambia de rama para continuar el desarrollo con git checkout develop

12. Por último, muévete a la rama de trabajo ejecutando git checkout hobbies

> ***Observa que se “borraron” las modificaciones realizadas en “skills”.***

Define la sección “hobbies”. A modo de ejemplo:

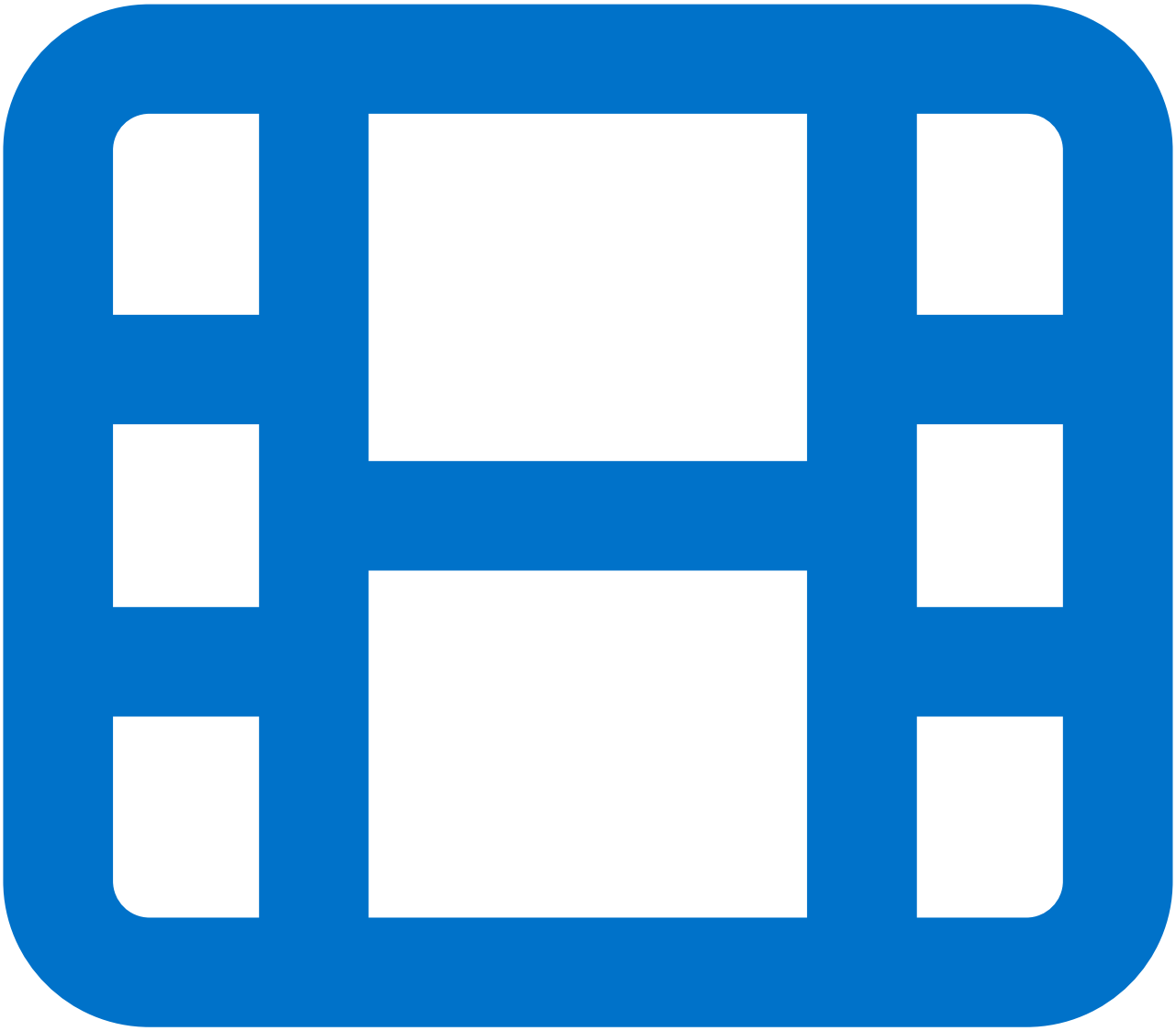
```
## Hobbies
- Dancing: ...
```

13. Prepara y “comittea” el archivo con los comandos correspondientes.

14. Cambia de rama para continuar el desarrollo con git checkout develop

📖 Material Audiovisual

Si tuviste dudas en alguna de las actividades, te invitamos a mirar el siguiente video donde se explica cómo solucionarlas:



Introducción a Git | Ramas | Egg