**Paso 1: Establecer main como la base limpia**

La rama main será nuestra "versión de oro", la que irá a producción. Primero, nos aseguramos de que estamos en ella y que está actualizada con lo que tienes en GitHub.

Generated bash

# Comando: git checkout main

# Propósito: Cambia tu espacio de trabajo a la rama 'main'.

# Si ya estás en ella, no hará nada. Si estabas en otra, cargará

# la última versión de los archivos de 'main'.

git checkout main

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

Generated bash

# Comando: git pull origin main

# Propósito: "Tira" de los últimos cambios de la rama 'main' desde GitHub

# y los fusiona en tu 'main' local. Esto asegura que partes de

# la versión más reciente que existe en la nube.

git pull origin main

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

En este punto, estás en main y tienes la versión más actualizada del código "oficial".

**Paso 2: Crear la Rama de Desarrollo (develop)**

Ahora vamos a crear tu espacio de trabajo principal. develop será una copia de main donde vivirán todas las nuevas funcionalidades y correcciones antes de ser consideradas "listas para producción".

Generated bash

# Comando: git checkout -b develop

# Propósito: Es una combinación de dos comandos:

# 1. 'git branch develop': Crea una nueva rama llamada 'develop'.

# 2. 'git checkout develop': Cambia inmediatamente a esa nueva rama.

# El resultado es que ahora estás en una nueva rama 'develop'

# que es una copia exacta de 'main' en este momento.

git checkout -b develop

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

Tu terminal ahora te indicará que estás en la rama develop.

**Paso 3: Publicar develop en GitHub**

La nueva rama develop solo existe en tu computadora. Necesitas subirla a tu repositorio de GitHub para que exista en la nube y puedas colaborar o simplemente tener una copia de seguridad.

Generated bash

# Comando: git push -u origin develop

# Propósito:

# 1. 'git push': "Empuja" o sube tus cambios a GitHub.

# 2. 'origin develop': Le especifica que quieres subir la rama 'develop'

# al repositorio remoto llamado 'origin' (que es el alias estándar para GitHub).

# 3. '-u' (o '--set-upstream'): Este es un paso crucial que solo haces la

# primera vez que subes una rama. Crea un "enlace" permanente entre tu

# rama 'develop' local y la rama 'develop' en GitHub. Gracias a esto,

# en el futuro solo necesitarás escribir 'git push'.

git push -u origin develop

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

¡Felicidades! Tu estructura de ramas profesional ya está configurada.

**Parte 4: El Flujo de Trabajo Diario (Cómo trabajar y subir cambios)**

Este es el ciclo que repetirás una y otra vez para cada nueva tarea.

**Escenario: Quieres añadir una nueva sección de "Reportes" a tu aplicación.**

**Etapa A: Preparar tu espacio de trabajo**

1. **Siempre empieza en develop y actualízala.**

Generated bash

# Vuelve a tu rama de desarrollo (si no estás ya en ella)

git checkout develop

# Descarga los últimos cambios de 'develop' desde GitHub.

# Es vital si trabajas en equipo para tener las últimas actualizaciones.

git pull origin develop

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

1. **Crea una rama específica para tu nueva tarea.** Nunca trabajes directamente en develop.

Generated bash

# Crea y cambia a una nueva rama. Nómbrala de forma descriptiva.

# El prefijo "feature/" es una convención común.

git checkout -b feature/agregar-reportes

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

**Etapa B: Desarrollar (El trabajo real)**

1. **Escribe tu código:** Haz todos los cambios necesarios en los archivos de tu proyecto (crea componentes, modifica la API, etc.).
2. **Guarda tu progreso con commit:** Un commit es como un punto de guardado. Haz tantos como necesites. Es bueno hacer commits pequeños y enfocados.

Generated bash

# Paso 1: Añade los archivos que has modificado al "área de preparación".

# El punto '.' significa "todos los archivos modificados en esta carpeta y subcarpetas".

git add .

# Paso 2: Crea el punto de guardado con un mensaje que describa el cambio.

git commit -m "Crea el endpoint base para el reporte de ventas"

# ... sigue trabajando ...

git add .

git commit -m "Diseña la tabla de reportes en React"

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

1. **(Opcional) Sube tu rama de feature a GitHub:** Esto es útil como copia de seguridad o si quieres que un compañero revise tu trabajo antes de integrarlo.

Generated bash

git push origin feature/agregar-reportes

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

**Etapa C: Integrar tus cambios (Terminaste la tarea)**

1. **Vuelve a la rama develop:**

Generated bash

git checkout develop

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

1. **Fusiona los cambios:** Trae todo el trabajo que hiciste en tu rama feature/agregar-reportes y únelo a develop.

Generated bash

git merge feature/agregar-reportes

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

1. **Sube el resultado a GitHub:** Ahora develop contiene tu nueva funcionalidad. Súbela a la nube.

Generated bash

# Como ya usamos '-u' antes, ahora solo necesitas 'git push'

git push origin develop

Use code [with caution](https://support.google.com/legal/answer/13505487).Bash

En este momento, tu nueva sección de "Reportes" está integrada en la línea de desarrollo principal del proyecto, pero **aún no está en producción.**

**Parte 5: El Despliegue (Cómo subir a producción "cuando quieras")**

Este es un paso especial que solo realizas cuando estás seguro de que los cambios en develop son estables y están listos para ser publicados. **No se hace desde la terminal de VS Code, sino en la interfaz web de GitHub.**

1. **Abre GitHub:** Ve a la página de tu repositorio en github.com.
2. **Crea un "Pull Request":** Verás un mensaje sugiriendo crear un Pull Request desde develop. Haz clic en él.
   * **Base:** Asegúrate de que la rama base sea main.
   * **Compare:** Asegúrate de que la rama a comparar sea develop.
3. **Revisa los Cambios:** Dale un título y una descripción al Pull Request. Revisa la lista de archivos modificados para una última confirmación.
4. **"Merge Pull Request":** Cuando estés listo, haz clic en el botón verde para fusionar.

**¡Acción-Reacción!**

En el momento en que el Pull Request se fusione, Git copiará todo el código de develop a main. **Render**, que está vigilando tu rama main, detectará esta actualización y **automáticamente comenzará a desplegar la nueva versión de tu aplicación.**

Con este flujo, tienes el control absoluto. Desarrollas libremente en ramas de feature, integras en develop, y solo cuando tú decides, fusionas a main para lanzar los cambios al mundo.