Cuaderno de ejercicios

Tema 1

El objetivo de estos ejercicios es aprender a utilizar los comandos básicos de control de versiones (Git) en entorno local, remoto y de manera colaborativa. En la última parte también se pide realizar una pequeña aplicación y desplegarla en un servidor web local, en máquina virtual y en Docker.

Entrega:

Debes entregar un documento donde indiques paso a paso las instrucciones (y capturas de pantalla si lo necesitas) que has empleado para resolver cada ejercicio, así como cualquier aclaración que quieras hacer constar.

La idea de este documento es que te sirva de guía de uso de Git para futuros ejercicios y actividades.

Git (trabajo individual)

Trabajo en local

1. **Inicialización de un repositorio**
    - Crea un nuevo directorio llamado mi-proyecto y conviértelo en un repositorio de Git. Dentro de este directorio, crea una subcarpeta llamada src y un archivo index.php con un mensaje de saludo.
2. **Añadir archivos al seguimiento**
    - Crea un archivo functions.php en la carpeta src con una función PHP que retorne un saludo. Agrega este archivo al área de preparación.
3. **Confirmar cambios (commit)**
    - Realiza un commit con un mensaje que describa la creación del archivo functions.php.
4. **Visualización del historial de commits**
    - Consulta el historial de commits de tu repositorio y muestra sólo los mensajes de commit.
5. **Ignorar archivos**
    - Crea un archivo .gitignore en el directorio raíz y configura Git para que ignore los archivos con la extensión .log.
6. **Modificar un archivo y realizar otro commit**
    - Edita el archivo index.php para que incluya y utilice la función de functions.php para mostrar el saludo. Realiza un commit describiendo los cambios.
7. **Comparar cambios**
    - Crea un fichero README.md en el directorio raíz y añádelo al repositorio. Haz un cambio en el fichero y muestra las diferencias entre el área de trabajo y el área de preparación. A continuación, añade el fichero al área de preparación y muestra las diferencias entre el área de preparación y el último commit.
8. **Eliminar archivos**
    - Elimina el archivo index.php y realiza un commit para reflejar el cambio.

Trabajo en local y ramas

9. **Crear y cambiar de ramas**
    - Crea una nueva rama llamada funcionalidad-1 y cambia a esa rama. En esta rama, añade una nueva función en functions.php que retorne un mensaje personalizado.
10. **Hacer commits en la nueva rama**
    - Realiza un commit en la rama funcionalidad-1 describiendo los cambios realizados en functions.php.

11. **Fusionar ramas (merge)**
    - Vuelve a la rama principal (main o master) y fusiona los cambios de la rama funcionalidad-1.
12. **Resolver conflictos de fusión**
    - Crea un conflicto de fusión editando functions.php en la rama main y funcionalidad-1, luego fusiónalas y resuelve el conflicto.
13. **Rebase**
    - Realiza un rebase de la rama funcionalidad-1 sobre main y explica las diferencias con merge.

Trabajo en remoto con GitHub

14. **Crear un repositorio en GitHub**
    - Crea un nuevo repositorio en GitHub llamado mi-proyecto y conéctalo a tu repositorio local. Realiza un push de tus cambios locales al repositorio remoto.
15. **Clonar un repositorio**
    - Clona un repositorio existente de GitHub a tu máquina local. Añade un nuevo archivo README.md con la descripción del proyecto y realiza un push de los cambios al repositorio remoto.
16. **Trabajar con ramas en remoto**
    - Crea una nueva rama en tu repositorio local llamada funcionalidad-2 y publícala en GitHub. Realiza cambios en functions.php en la nueva rama y realiza un push de los cambios.
17. **Realizar pull de cambios**
    - Modifica desde GitHub el fichero README.md. Obtén los últimos cambios desde el repositorio remoto y actualiza tu repositorio local.
18. **Fork y pull request**
    - Haz un fork de un repositorio en GitHub de algún compañero (o de otra cuenta de GitHub que tengas), clona tu fork localmente, realiza cambios en index.php, y crea un pull request.
19. **Revisar pull requests**
    - Revisa y comenta el pull request en GitHub que haya creado tu compañero (o tú mismo, si utilizas dos cuentas de GitHub). Aprueba o solicita cambios según sea necesario.

Git (trabajo en equipo)

En este bloque de ejercicios hay que trabajar por parejas.

Vais a desarrollar una sencilla aplicación web de tipo calculadora. Debe ser lo más básica posible, pero debe contar con una interfaz web que permita introducir 2 números y realizar operaciones de suma, resta, multiplicación o división y mostrar el resultado. El proyecto deberá tener 3 ramas, una rama para la interfaz, otra para realizar la suma y la resta, y otra para realizar la multiplicación y la división. Una vez estén todas las ramas finalizadas, se deberá hacer un merge a la rama main.

Deberéis crear un repositorio de GitHub para realizar los ejercicios.

1. **Inicializar el repositorio y configurar el entorno**
    - Uno de los integrantes del equipo (A) crea un nuevo repositorio en GitHub llamado calculadora-web y lo inicializa con un archivo README.md. Luego, clona el repositorio en su máquina local e invita al otro integrante (B) como colaborador.
    - Clonar el repositorio en la máquina local de B.
2. **Estructurar el proyecto**
    - "A" crea la estructura inicial del proyecto en la rama main con los archivos y carpetas necesarios para una aplicación web simple (HTML, CSS, y PHP).
        - index.html para la interfaz de usuario.
        - styles.css para el estilo.
        - calculadora.php para manejar las operaciones de la calculadora.
3. **Crear las ramas de trabajo**
    - "A" crea tres ramas adicionales desde main:
        - funcionalidad/interfaz
        - funcionalidad/suma-resta
        - funcionalidad/multiplicacion-division
4. **Implementar la interfaz**
    - "B" cambia a la rama funcionalidad/interfaz y desarrolla la interfaz de usuario en index.html, incluyendo un formulario con dos campos numéricos y botones para las operaciones de suma, resta, multiplicación y división, así como un campo para el resultado.
5. **Implementar suma y resta**
    - "A" cambia a la rama funcionalidad/suma-resta y desarrolla las funcionalidades de suma y resta en calculadora.php, asegurándose de que se manejen las solicitudes del formulario de index.html.

6. **Implementar multiplicación y división**
    - “B” cambia a la rama funcionalidad/multiplicacion-division y desarrolla las funcionalidades de multiplicación y división en calculadora.php, asegurándose de que se manejen las solicitudes del formulario de index.html.
7. **Revisión y merge de la interfaz**
    - “A” revisa los cambios en la rama funcionalidad/interfaz, realiza un merge a main y resuelve cualquier conflicto.
8. **Revisión y merge de suma y resta**
    - “B” revisa los cambios en la rama funcionalidad/suma-resta, realiza un merge a main y resuelve cualquier conflicto.
9. **Revisión y merge de multiplicación y división**
    - “A” revisa los cambios en la rama funcionalidad/multiplicacion-division, realiza un merge a main y resuelve cualquier conflicto.
10. **Merge final y prueba de la aplicación**
    - Ambos integrantes se aseguran de que la rama main tiene la última versión del código de todas las ramas, realizan pruebas finales de la aplicación, y actualizan el archivo README.md con instrucciones de uso.

Pruebas

1. Prueba la aplicación en localhost. Hazlo de dos formas, con el servidor web Apache y con el servidor web incorporado (built-in) de PHP.
2. Prueba la aplicación en la máquina virtual Ubuntu (accede primero desde el navegador de Ubuntu y luego desde el navegador de la máquina host). Hazlo también de dos formas, con el servidor web Apache y con el servidor web incorporado (built-in) de PHP.
3. Crea un Docker personalizado con tu aplicación y luego ejecútalo en localhost (hazlo de manera similar a cómo lo has hecho en el ejemplo práctico del Tema 1).