Obligatorio 2 SISTEMA DE GESTIÓN DE VENTAS CON GIT Y GITHUB

1. Automatización con Script Bash

Se creo un archivo llamado upload-semanal.sh que automatiza la confirmación y subida de cambios al repositorio, y además actualiza el README.md con un resumen semanal.

Que hace el script:

- Verifica si hay cambios (modificados, agregados o eliminados).
- Si no hay cambios: agrega una línea al README con la fecha y el mensaje “No hay cambios para subir”, hace commit y push.
- Si hay cambios: agrega todos los archivos, hace un commit, calcula cuantas líneas se modificaron, actualiza el README y hace push.

Comandos clave utilizados:

Git clone <URL-del-repositorio> : Clona un repositorio remoto a tu computadora local. Se usa para descargar el proyecto desde GitHub y trabajar localmente.

git diff --quiet : Verifica si hay **cambios sin confirmar** en el proyecto.
(No muestra nada, solo da un estado verdadero/falso.)

git add . : **Agrega todos los archivos modificados** al área de preparación (staging), para que estén listos para ser confirmados (commiteados).

git commit -m "mensaje" : Crea un **commit** con el mensaje indicado, guardando los cambios agregados previamente.

git push origin <rama> : **Sube los commits locales** al repositorio remoto (GitHub), en la rama especificada (por ejemplo: main, develop, feature/venta-productos, etc).

git rev-parse --abbrev-ref HEAD : Devuelve el **nombre de la rama actual**, útil para automatizar scripts sin tener que escribir la rama manualmente.

git diff --shortstat HEAD~1 HEAD : Muestra un **resumen de los cambios** entre el último commit (HEAD) y el anterior (HEAD~1).
Incluye cuántas **líneas se agregaron o borraron**.

2. Uso de GitFlow (estructura de ramas)

Se organizo el repositorio con GitFlow, creando ramas especificas:

- main: versión estable

- develop: versión de desarrollo

- feature/autenticación: login con validación

- feature/alta-productos: alta de productos

- feature/venta-productos: venta de productos

Comandos:

git checkout -b feature/autenticacion
Se usa para crear una nueva rama para desarrollar lo relacionado a dicha rama (en este caso, para autenticación).

git add . && git commit -m "mensaje"

git push origin feature/autenticacion : Sube la rama actual al repositorio remoto en GitHub, envia los commits de esa rama para hacer un pull request o compartir.

3. Desarrollo funcional del Sistema

Además de la automatización con Git y GitHub, se desarrolló un sistema funcional usando HTML y JavaScript, trabajando desde Visual Studio Code.

login.js: contiene la lógica para el ingreso al sistema.

productos.js: permite dar de alta productos con nombre, precio y stock. Se guarda la información y se muestra en un listado que se actualiza automáticamente.

ventas.js: gestiona la venta de productos. Permite elegir un producto y descontar stock.

Estas funcionalidades fueron trabajadas cada una en su propia rama (feature/autenticación, feature/alta-productos, feature/venta-productos) siguiendo la metodología Gitflow.

HTML: Se desarrolló una página HTML principal que centraliza la interacción del usuario con el sistema. Esta página:

-Está vinculada con los scripts JS creados.

-Permite simular un flujo completo: iniciar sesión, agregar productos y realizar ventas.

4. Pull Requests y Merge

Cada funcionalidad se hizo en ramas feature y se unio a develop con una Pull Request (PR).

- Pull Request: propone cambios para revision.

- Merge: une cambios a develop o main.

5. Plantillas de GitHub Issues

Se crearon 4 plantillas automáticas dentro de .github/ISSUE_TEMPLATE/, una por tipo:

- Documentación
- Mejora
- Feature Request
- Bug

Ejemplos de Issues

Se crearon 2 issues reales por categoria:

Bugs: - Error al comprar con stock 0
- Falla en validación del login
Mejoras: -Implementar listado de productos
- Aplicar CSS al sitio
Documentación: - Instrucciones de script
- Reseteo de productos al refrescar la pagina
Feature Request: - Categorías a productos
- Filtro por categoría