+++ date = '2026-02-16T18:03:45-08:00' draft = false title = 'Practica_0: Manejo de repositorio' +++

secciones

- Uso markdown
- Uso de Git y Github
- Uso de HUGO y Github Pages

descripcion de la practica

Esta practica consiste en aprende que es y com outilizar una herramienta muy util para crear texto para reportes o incluso paguinas web y como se utiliza com oe su sintaxis ademas del uso de repositorios en github diversos comando necesarios para usar git como guardar nuestro codigo en un repositorio en github, por ultimo el como convinar nuestros conocimientos en el huso de github y markdown para para crea una paguina web estatica atravez de otra rerramientas llamada HUGO y el uso de github pages.

Desarrollo

Seccion 1: Uso de markdown

Que es Markdown?

Markdown es un lenguaje de marcado ligero que se utiliza para dar formato a textos de manera sencilla y rápida, usando una sintaxis fácil de leer y escribir. Fue creado en 2004 por John Gruber con el objetivo de permitir que las personas escribieran contenido con formato sin necesidad de usar código HTML complejo.

La principal característica de Markdown es que el texto original sigue siendo legible incluso sin procesarse. Es decir, puedes entender el contenido aunque no se haya convertido a su formato final (como una página web).

Como se utiliza?

Markdown funciona mediante el uso de símbolos especiales que indican cómo debe mostrarse el texto cuando se convierte a un formato final, como HTML, PDF o una página web.

Esos símbolos no cambian el contenido del texto en sí, sino que actúan como “instrucciones” para darle formato.

Cuando escribes en Markdown, el archivo generalmente tiene extensión .md. Luego, un procesador de Markdown interpreta esos símbolos y los transforma en código HTML automáticamente.

sintaxis

Markdown usa símbolos simples para agregar y darformato al texto que se ingresa como por ejemplo:

1. Titulos y encavezados

```
# Titulo 1
## Titulo 2
### Titulo 3
```

```
#### Titulo 4
...
```

2. Texto en negritas, cursiva y Tachado

```
**texto en negritas**
*cursiva*
***negrita y cursiva***
~~texto tachado~~`
```

3. Listas

```
*Elemento 1
    *Elemento 1.1
    *Elemento 1.2
*Elemento 2

+Elemento 1
    +Elemento 1.1
    +Elemento 1.2
+Elemento 2
```

Numeradas

```
1. Elemento 1
2. Elemento 2
3. Elemento 3
```

4. Enlaces

```
[nombre del enlace](URL)
```

5. Imagen

```
![texto alternaivo](imagen.jpg)
```

6. comentarios

```
<!-- comentario-->
```

Seccion 2: Uso de Git y Github

Que es Git y Github?

Git es un sistema de control de versiones distribuido creado por Linus Torvalds que permite registrar, organizar y gestionar los cambios realizados en un proyecto, especialmente en código fuente, manteniendo un historial completo de modificaciones y facilitando el trabajo en equipo mediante ramas y fusiones.

por otro lado, GitHub es una plataforma en línea, propiedad de Microsoft, que utiliza Git para alojar repositorios en la nube, permitiendo almacenar proyectos, compartirlos con otras personas y colaborar de forma remota, siendo la principal diferencia que Git es la herramienta de control de versiones que funciona localmente y GitHub es el servicio web donde esos proyectos pueden publicarse y gestionarse en internet.

Como se utiliza?

Git se utiliza desde la terminal para llevar el control de versiones de un proyecto; permite registrar cambios, organizarlos mediante confirmaciones (commits), trabajar con distintas ramas y sincronizar el proyecto con un repositorio remoto como GitHub para compartirlo o respaldarlo en línea.

Sintaxis

```
git init: inicia un nuevo repositorio en la carpeta actual
git add <archivo>: agrega un archivo especifico y lo deja listo para guardarse
git add .: agrega todos los archivos modificados
git commit -m "mensaje descriptivo": guarda todos los cambios seleccionados con
add
git status: Muestra los archvos modificados sin guardar y los ya guardados
git log: muestra el historial de commits realizados en el programa
git branch <nombre_rama>: Crea una nueva rama, que permite trabajar en cambios sin
afectar la rama principal
git checkout <nombre_rama>: Cambia a la rama indicada para trabajar en ella
git merge <nombre_rama>: Combina los cambios de la rama especificada con la rama
actual.
git remote add origin <URL>: Vincula el repositorio local con un repositorio
remoto
git push origin main: Sube los commits de la rama main al repositorio remoto.
git pull origin main: Descarga y actualiza el repositorio local con los cambios
que estén en el repositorio remoto.
```

Seccion 2: Uso de HUGO y Github Pages

Que es HUGO y github pages

Hugo es un generador de sitios web estáticos que permite crear páginas web a partir de archivos escritos en Markdown, organizándolos mediante plantillas y temas para generar automáticamente un sitio completo en formato HTML, CSS y JavaScript listo para publicarse; funciona de manera local en la computadora del desarrollador y convierte el contenido en un sitio web rápido y optimizado sin necesidad de bases de datos ni servidores complejos.

Por otro lado, GitHub Pages es un servicio de alojamiento web ofrecido por GitHub que permite publicar sitios web estáticos directamente desde un repositorio, facilitando el despliegue de proyectos personales, portafolios o documentación sin necesidad de contratar un hosting externo, siendo la diferencia principal que Hugo se utiliza para crear y generar el sitio web, mientras que GitHub Pages se utiliza para alojarlo y hacerlo accesible en internet.

Como funciona

Hugo funciona generando un sitio web estático a partir de archivos de contenido escritos generalmente en Markdown; el proceso comienza creando un proyecto con hugo new site, después se agrega un tema que define el diseño visual, se crean páginas o entradas con hugo new, y al ejecutar hugo server el programa construye automáticamente el sitio y lo muestra en un servidor local para previsualizarlo; cuando el sitio está listo, el comando hugo genera una carpeta llamada public que contiene todos los archivos HTML finales listos para publicarse.

Por su parte, GitHub Pages funciona tomando los archivos estáticos almacenados en un repositorio de GitHub y publicándolos como un sitio web accesible mediante una URL; una vez que el repositorio está configurado para usar GitHub Pages, la plataforma detecta los archivos generados (por ejemplo, los creados por Hugo) y los despliega automáticamente, permitiendo que el sitio esté disponible en internet sin necesidad de configurar un servidor propio.

Sintaxis Basico

```
hugo new site nombre_sitio:  Crea un nuevo proyecto de sitio web.
cd nombre_sitio: entra a la carpeta del sitio
hugo new posts/mi-post.md: Crea una nueva página o entrada en formato Markdown.
hugo server: Inicia un servidor local para previsualizar el sitio.
```

Conclusiones

En conclusión, esta práctica permitió comprender y aplicar herramientas fundamentales en el desarrollo web moderno. El uso de Git como sistema de control de versiones facilitó la organización y el seguimiento de los cambios en el proyecto, mientras que GitHub permitió almacenar y compartir el repositorio en la nube, favoreciendo el trabajo colaborativo y el respaldo del código. Por su parte, Markdown simplificó la creación de contenido estructurado mediante una sintaxis clara y ligera. Estos conocimientos se integraron con Hugo, un generador de sitios estáticos que transforma archivos Markdown en páginas web funcionales, y con GitHub Pages, que hizo posible publicar el sitio en internet de manera sencilla y gratuita. En conjunto, estas herramientas fueron utiles para pasar desde la escritura de contenido hasta la publicación de una página web completa, comprendiendo el flujo real de desarrollo, control de versiones y despliegue profesional de un proyecto.