Glosario CH31

# Indice

[Frontend](#tfjhq7lczxfl)

[Backend](#9eid2irnas88)

[API](#fospf747wgii)

[Interfaz](#7dfdvuzbocd7)

[Lenguajes de programación](#du3jzi8lol5m)

[Internet](#hwfvpyge417z)

[Cliente/Servidor](#rseg9qwwfvs)

[Solicitudes HTTP](#m6qsk8smom9t)

[CRUD](#np1qszae3p85)

[Respuestas del servidor](#pn0xsubbx9e1)

[TCP/IP](#dot0i5wrub38)

**Frontend** la parte visual

**Ux / UI:** User Experiencie / User Interface

**HTML:** Hypertext Markup Language

**CSS:** Cascade Style Sheet

**JS:** JavaScript

**Bootstrap:** framework de diseño y desarrollo web

**Validaciones**

**Consumo de APIs**

**Lógica en el cliente**

**Interface web**

**Backend** lo que no vemos, el cerebro, lógica

**Java:** Lenguaje de programación en la parte del servidor

**SQL:** Structured Query Language, lenguaje para gestionar y manipular bases de datos

**Spring Boot:** framework de desarrollo que facilita la creación de aplicaciones Java. Reduce configuraciones complejas y permite centrarse en la lógica de negocio.

**Bases de datos**

**Rendimiento**

**Lógica en el servidor**

**Seguridad**

**Microservicios**

**API** Interfaz de programación de aplicaciones



**Interfaz** Intermediario, puente, conexión



**Sintaxis**: Forma de “escribir”

**Lenguajes de alto nivel:** se asemeja al lenguaje humano y utiliza muchas palabras (Java, Python)

**Lenguajes de bajo nivel:** lenguaje binario (0 y 1)

**Tipado:** cómo se declaran las variables

**Declarar las variables:** especificar nombre y tipo de variable

**Variable:** donde se guardan los datos de un programa

**Internet:** redes interconectadas

**WWW:** World Wide Web / Red Mundial

Red:

**Cliente:** quien solicita y recibe servicios o recursos

**Servidor:** proporciona servicios, recursos o información

**Protocolo:** Forma de hacer las cosas

**Servicios:** Información, recursos, imágenes, videos, etc.

**Base de datos:** Información categorizada y organizada

**Solicitudes GET - POST - PUT - DELETE**

**Get:** obtener

**Post:** agregar

**Put:** modificar

**Delete:** borrar

**CRUD:** **C**reate, **R**ead, **U**pdate, **D**elete, representa operaciones básicas para gestionar información en bases de datos

Códigos de respuesta para conexiones del Servidor

**200:** Aquí está la hamburguesa, ok

**301:** La hamburguesa la venden en otro restaurante, Movido permanentemente, Redirección

**302:** Movido temporalmente

**401:** No tiene permitido comprar hamburguesas, Unauthorized

**404:** Hamburguesa no encontrada, Not found

**500:** La cocina se prendió fuego, Internal server error

**El servidor** (quien prepara nuestra hamburguesa) toma todos los datos de nuestra petición (carne, pan, jitomate, lechuga, etc), la manda **al internet** (el encargado del mostrador) quien es quien entrega **nuestra petición** tal cual la pedimos (el cliente).

**TCP/IP**: protocolo de control de transmisión / protocolo de internet

**HTTP:** Protocolo de Transferencia de Hipertexto (Hypertext Transfer Protocol)

**Algoritmo:** conjunto de instrucciones o reglas que guían la realización de una tarea o resolución de un problema. Es como una receta que te dice exactamente qué hacer en cada paso para alcanzar un resultado deseado

**lista de comandos terminal**

**pwd:** print word directory te da una mejor explicaciónde donde estas trabajando (la ruta de dirección)

**ls:** te enlista los elementos que se encuentran en la ruta

-**parametros:** usas el - después del comando

- l : formato long

-a: all muestra todos

-R: hacer el comando recursivo

**clear:**  limpiar la ventana de comando

**history:** muestra un historial de los comando usados

**cd:** change directory/ cambiar de directorio **sintaxis: cd directorio**

parametro: .. (espacio punto punto) regresar un ruta del directorio

**mkdir:** make directory

**rmdir:** remove directory

el signo $ indica que esta listo para llevar a cabo un comando si no aparece esta llevando a cabo un proceso

**cp:** copiar sintaxis: cp Cama.txt ../CuartoMemo/

**rm:** remove

la estructura de un comando es:

que quiero hacer / como lo quiero hacer / a quien lo voy a hacer



**GIT:** sistema de control de versionado

Git hub permite en un solo archivo tener versiones de tal manera que si quieres puedes volver a algún punto en específico cada versión viene con un comentario y continuar el flujo hasta el punto deseado y volver a a cualquier versión posible que tenga un comentario

**Repositorio:** Es un proyecto que se encuentra en la nube. esto permite que varias personas trabajen en un mismo código, algo similar como documentos de drive

Comandos Git

Usamos el prefijo **git** para especificar

Principales comando para workflow

- **git init** haces que el directorio actual se convierta en un repositorio

- **git status** te dice el estado actual del repositorio

Ejemplo: dice que el archivo de glosario de terminos tecnicos no está trackeado

**- git add** agrega el archivo para poder trabajarlo

add . (“add espacio punto“) es para agregar a la mesa de trabajo todos los documentos de directorio

**git commit** se usa para agregar el comentario, pero nos abre el editor de texto que en este caso es vim

-m nos permite agregar el comentario seguido de el comando ej: git commit **-m** “se hicieron cambio a este archivo ”

**git push:** envía los cambios a la red para que quede de manera global

carpeta de proyecto estará sincronizada

ejemplo: git remote add origin <https://github.com/Angelhdz98/Glosario_Generation.git>

**git log:** Te muestra los commits realizados a los documentos

**git remote add origin:** es para poder decir cual es el repositorio de github con el que mi

nota:

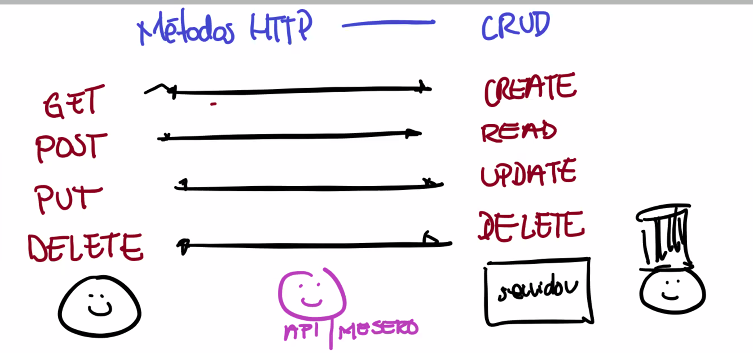
Parametros git

-- version: no dice la versión de git o programa del prefijo



21 de agosto de 2023

Introducción a paginas web

Recordando: todas las peticiones (métodos ) HTTP tienen su contra parte en el metodo CRUD

HTTP CRUD

Post Read

Get Create

Put Update

Delete Delete

“Ya que es muy diferente lo que el cliente con respecto a lo que se ejecuta en la cocina”

**Objetivos de hoy**; describir los componentes de una aplicación web y su funcionalidad:

HTML

CSS

JavaScript

Diferencia de una pagina web, un sitio web y una aplicación web

Terminología:

**Landing page:** “pagina de aterrizaje”

**Responsavidad:** que la pagina responda a los diferentes tamaños de pantalla

Pagina web: Es el elemento más sencillo cuando se habla de elementos web su propósito es solo brindar información en texto con un formato y estilo sencillo [ejemplo 1era pag web: <http://info.cern.ch> ] estas se pueden usar como landing page; Esta busca aterrizar a los usuarios para convertirlos en clientes potenciales. Busca atraer tu atención y recibir una respuesta para después ofrecerte un servicio ejemplo la pagina web de platzi: <https://platzi.com/>

Al pasar esta pagina uno se encuentra con mas interacción

por lo tanto deducimos que cuando trabajemos con muchas paginas web al desarrollar un sitio web

**sitio web:** es un conjunto de paginas web

dominio: dirección dentro del catalogo de la web /dato /dato /dato se llaman endpoint

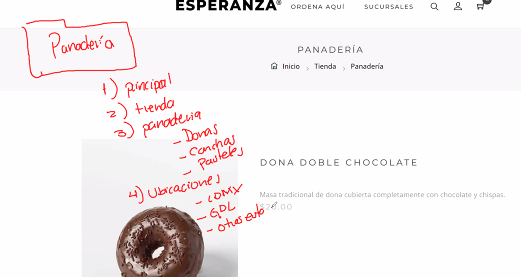
ejemplo:



DominioGeneral/endpoint/endpoint

Dominio = dirección principal

Es similar a una organización de las carpetas:



Todo esta conectado por ramificaciones

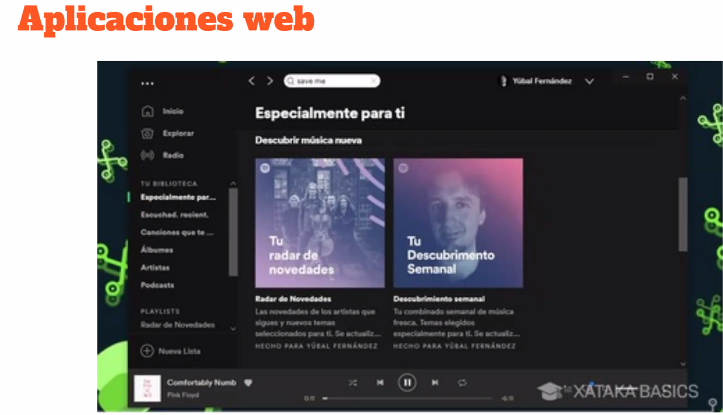
Aplicación WEB: no es solo para dar información y esta basada en las interacciones

* Usan muchísimas herramientas (puede visualizarlas con wappalyzer)
* Tienen responsividad

Notas :

Una pagina web es solo una pagina web: un sitio web son varias paginas web conectadas y un aplicación web es el mismo sitio web con funcionalidades y una parte clave de esta sería el servidor

La responsividad formará parte de todas nuestras tareas y aplicaciones



**HTML** es el esqueleto: encontramos termino similares al cuerpo: head, body, footer

**CSS**: skin le vamos a dar un mejor aspecto visual

**JS**: es la parte que da la funcionalidad y movimientos

Los navegadores suelen leer esto en este mismo orden

“Tener mucho cuidado que todas las cosas se carguen bien”

Por que si no el usuario puede creer que algo no funciona y se irá a otra lo que se traduce como perdidas monetarias

Preguntas de examen:

**HTML es un lenguaje de programación?** No es un lenguaje de etiquetado

HyperText Markup language (entrevista)

“Lenguaje de marcado de hipertexto“ con

**CSS:** Cascading Style Sheet estilo de hojas en cascada ) es un lenguaje de estilos

**JavaScript:**  es un lenguaje de programación

Con HTML podemos crear la estructura del cuerpo **head**,**body** y footer que no suele ser tomado com uno de los principales

**Head:** suele ir información que solo suele ver el navegador

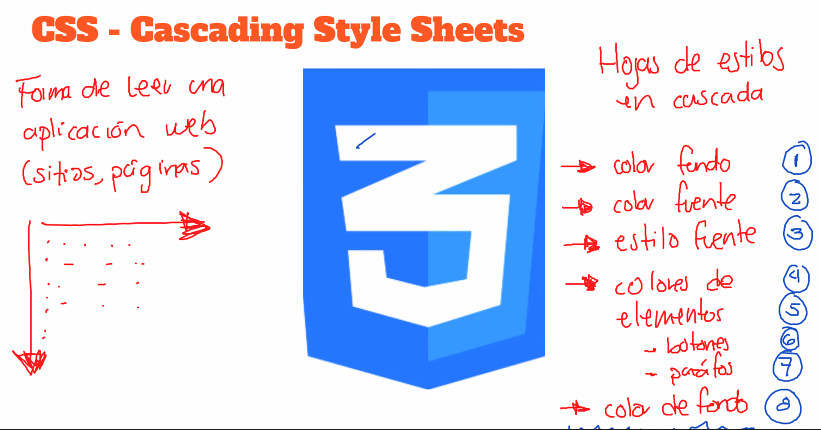
Idioma, caracteres y soporte de caracteres (Cuestiones de codificación y funcionalidad)

**Body:**  esto es lo que ve el usuario

Analogía: todos nos pueden ver y no saben que hay en nuestra casa pero si van ver nuestro cuerpo ropa y demás que mostramos

Footer: Se utiliza para datos adicionales (accesos, términos de privacidad )

**Etiquetas semánticas**: marca las secciones del cuerpo podemos segmetar el head el body el footer

**CSS:** Cascading Style sheets. Forma de leer una aplicación web

Las cosas se ejecutan a manera de cascada

Analogía de cuando te cambias:si te vas a poner un traje primero te pones la camiseta, luego la camisa de vestir luego la corbata y el saco y lo primero que verá alguien será el saco que fue lo ultimo que pusiste

**JavaScript:** Es diferente de Java pero si es un lenguaje de programación

“cuando estamos viendo una pagina, vemos una copia de la pagina que vive en el servidor

Cambios en pagina Santander para empezar a entender herramientas de inspector y HTML

**Extensiones instaladas**

Auto Close Tag: te cierra el tag con </name>

Auto Rename Tag: al poner el nombre te lo repite

Code Snapshot: te permite tomar capturas de tu Código

Live Server: hacer un servidor local para ver actualizaciones en tiempo real

Material Icon Theme:

Path Intellisense: te ayuda manejar entre las rutas

**HTML primeros pasos**

Las etiquetas se abren con “<nombreetiqueta> y se cierra con “</nombreetiqueta>”

Ejemplo de un etiquetado con HTML

Estructura básica html

**<!DOCTYPE html>** tipo de documento

**<html lang="en">** marca que es un archive html en ingles (language) lenguaje (en= english)

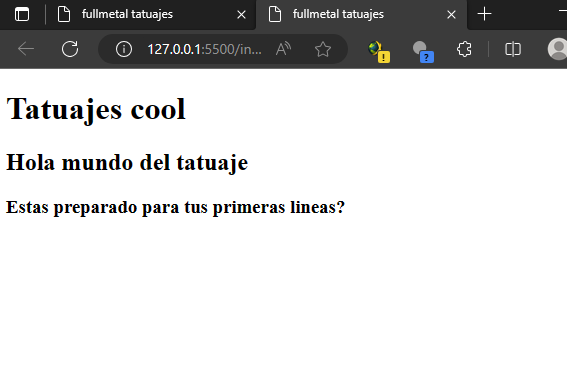
**<head>** (inicio de head)

**<meta charset="UTF-8">** Tipo de caracteres a utilizar

**<meta name="viewport"** ??  **content="width=device-width, initial-scale=1.0">**

**<title>Document</title>** nombre

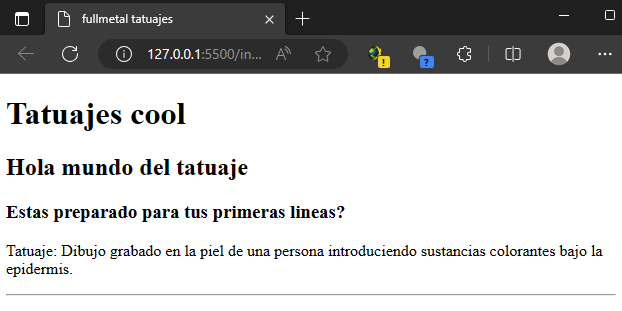
**</head>** cierre del head

**<body>**  abre body

**</body>** Cierra body

**</html>** cierra el archivo

**<h1>etiqueta de títulos </h1>** se usa de h1 a h6 pero solo se recomiendan los primeros 2 ya que el tercero ya pasa a ser muy pequeño

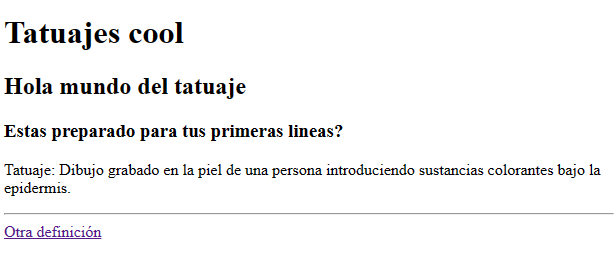
**. . . .**

**</h6>limite de etiqueta de título </h6>**

**<p> parrafos</p>**  es para poder usar párrafos

**<hr>** agrega una línea recta Recta Horizontal

“horizontal rule” (no lleva cierre)



**<a href=""></a>** Agregas un link a alguna otra pagina.

sintaxis

**<a href="dirccion.com/delURL"> mensaje en el hipervinculo</a>**

href es un atributo para agregar la dirección URL

target es un

Lista de atributos y propiedades: <https://developer.mozilla.org/es/docs/Web/HTML/Element>

Tarea del 21 de agosto de 2023 **¿Qué es una etiqueta semántica?**

Ayudan a definir la estructura del documento y permiten que las paginas web sean mejor indexadas por los buscadores.

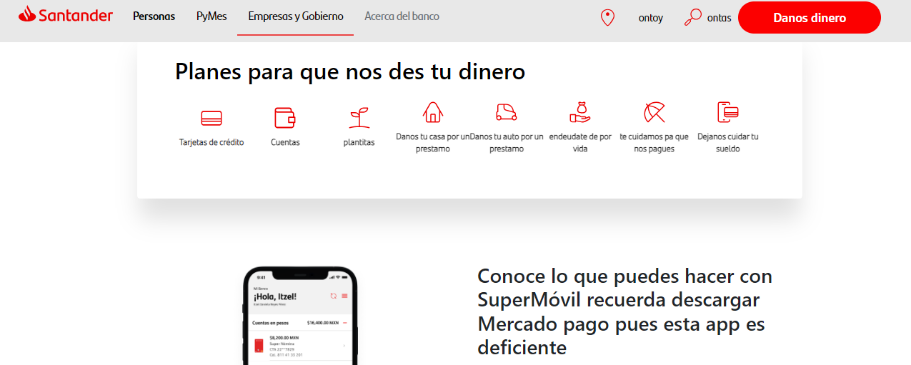
Se dice que una etiqueta es semántica si tiene que ver con el contenido es decir si nos informa de lo que se trata su contenido. Frente a las etiquetas semánticas tenemos otro tipo de etiqueta como las que afectan al formato como UL, a multimedia , como video, etc

Antes de las etiquetas semánticas el contenido era estructurado utilizando la etiqueta DIV que no daba información del contenido a menos que se use la propiedad ID o la propiedad CLASS

Las etiquetas semánticas más importantes son las que definen la estructura general del sitio estas son: HEADER, ARTICLE, SECTION, ASIDE, FOOTER y NAV

Aunque también existen etiquetas que se refieren a elementos concretos de la pagina como: FIGURE para imágenes y gráficos FIGCAPTION que es un pie de imagen, TIME para definir fechas y horas y MAIN, MARK, SUMARY y DETAILS





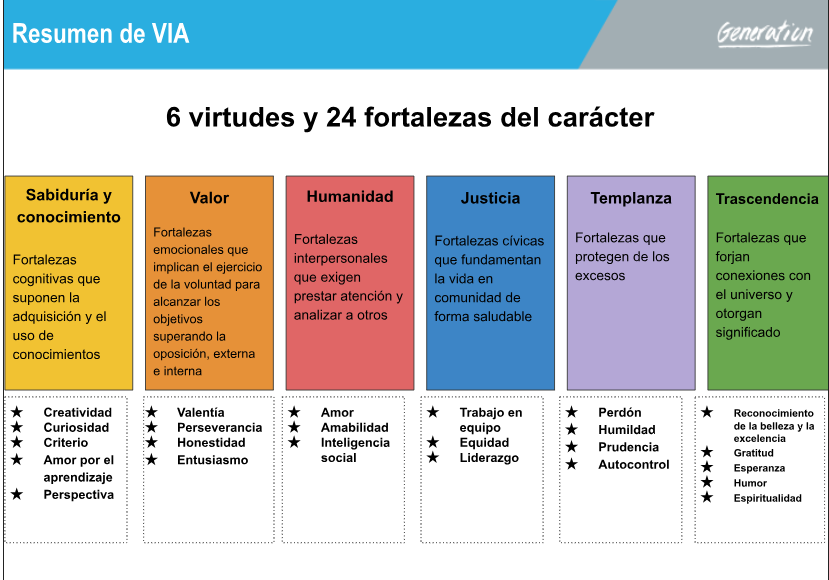
**Diferencia entre front end y back end**

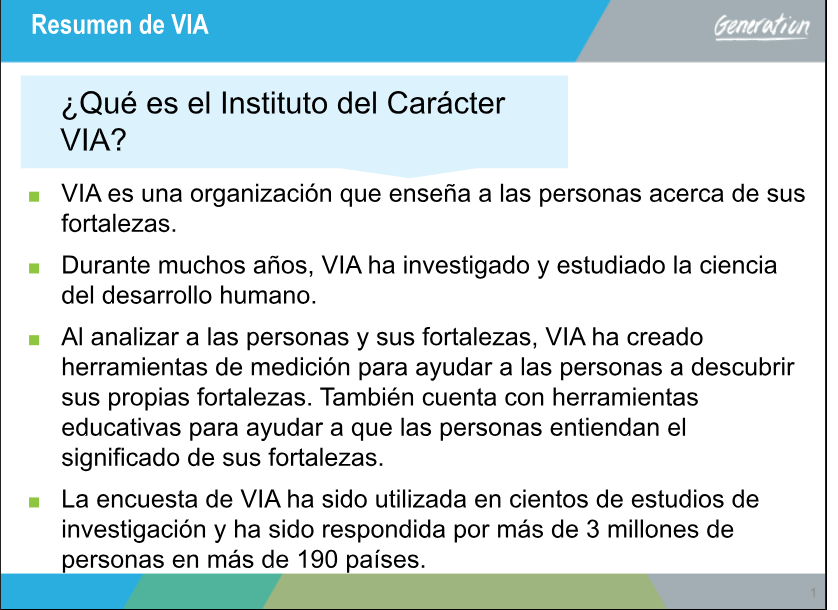
****

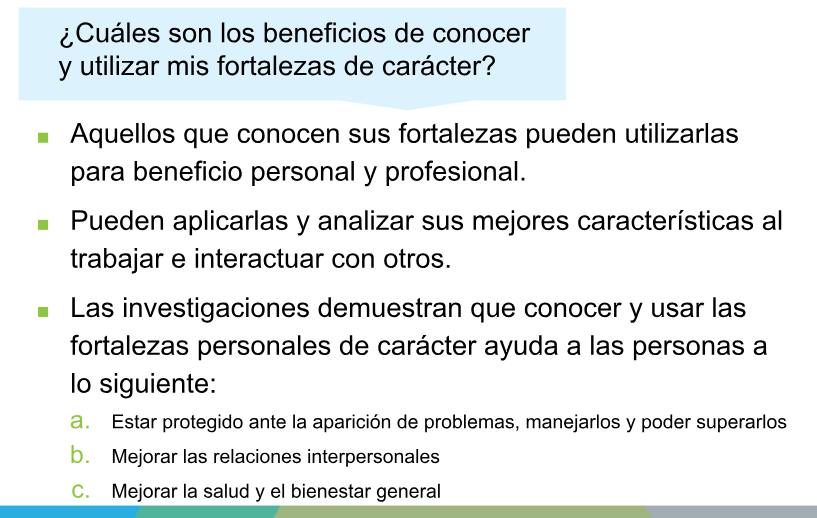
El inspector de métodos nos permite ver los métodos HTTP que realiza la computadora

Soft skils:

Fortalecas VIA



Que es VIA: 



Resultados prueba VIA

