Programacion II

Tecnologías WEB: HTML, CSS y JavaScript

Tecnicatura Universitaria en Programación - UTN

Introducción a la Materia

Objetivo de la Materia. ¿Qué se aprenderá durante el cursado?

Entorno de desarrollo: Navegador (preferentemente Chrome o Firefox), VS Code y cuenta en GitHub.

Canales de Comunicación: CVG, Zoom y Discord.

Hoja de Ruta:

U1: Introducción: Cómo funciona Internet y la Web

U2: HTML5

U3: CSS3

U4: JavaScript

U5: Python

U6: Flask

Evaluaciones.

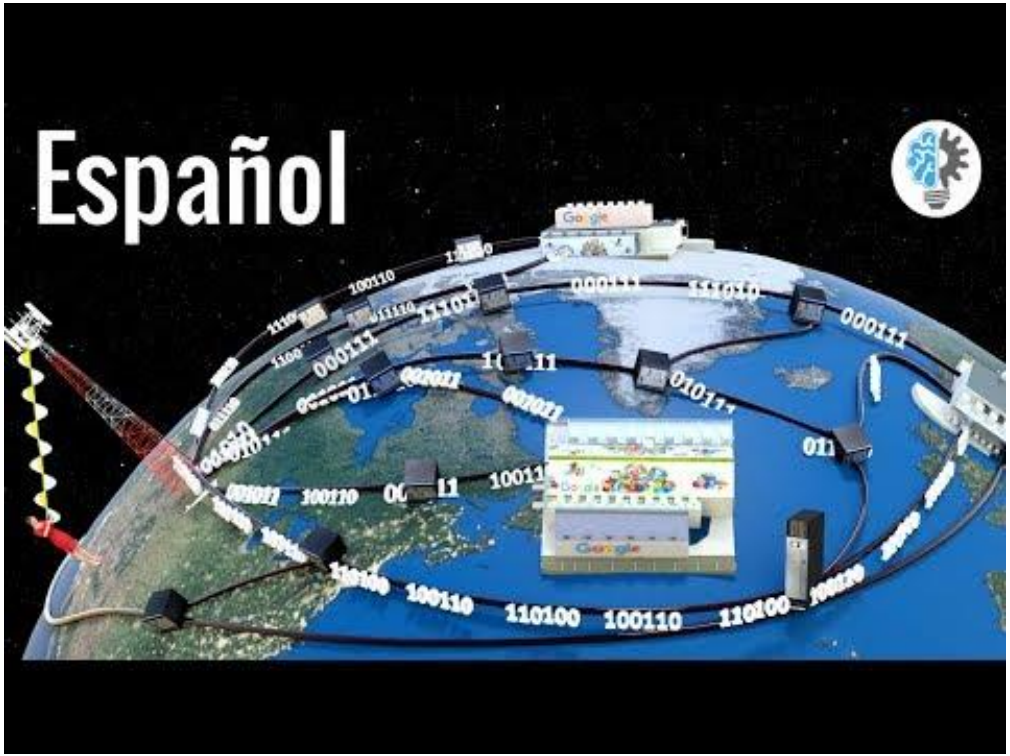
Trabajo práctico integrador.

Internet es la columna vertebral de la Web, la infraestructura técnica que hace posible la Web. En su forma más básica, Internet es una gran red de computadoras que se comunican todas juntas.

Todo comenzó en la década de **1960**, como una forma para que entidades gubernamentales (científicas y militares) compartieran información.

Con el calentamiento de la Guerra Fría, en **1969** EE.UU. crea la **ARPANET** (Red de Agencias de Proyectos de Investigación Avanzada), la red que finalmente evolucionó hasta convertirse en lo que ahora conocemos como Internet.

El **1 de enero de 1983** se considera el nacimiento oficial de Internet ya que se lanzó el protocolo TCP/IP que permitió que diferentes tipos de computadoras compartieran información entre sí. ARPANET adopta este protocolo y quedó establecido el lenguaje universal de comunicación entre todas las computadoras.



Fuente: https://www.youtube.com/watch?v=rw41W8crZ_Y

Resumen - Conceptos más importantes:

- Conjunto descentralizado de redes interconectadas que funciona como una red lógica única.
- Infraestructura de comunicación: antenas, satélites, cables submarinos, routers, entre otros.
- **ARPANET** fue la primera red de computadoras.
- **TCP/IP:** el Protocolo de Control de Transmisión y el Protocolo de Internet son protocolos de comunicación que definen cómo deben viajar los datos a través de Internet. Divide los datos en paquetes y los une entre origen y destino.
- Todo dispositivo conectado a Internet tiene un identificador único, de ahí las **IP públicas** (servidores) e **IP Privadas** (clientes)

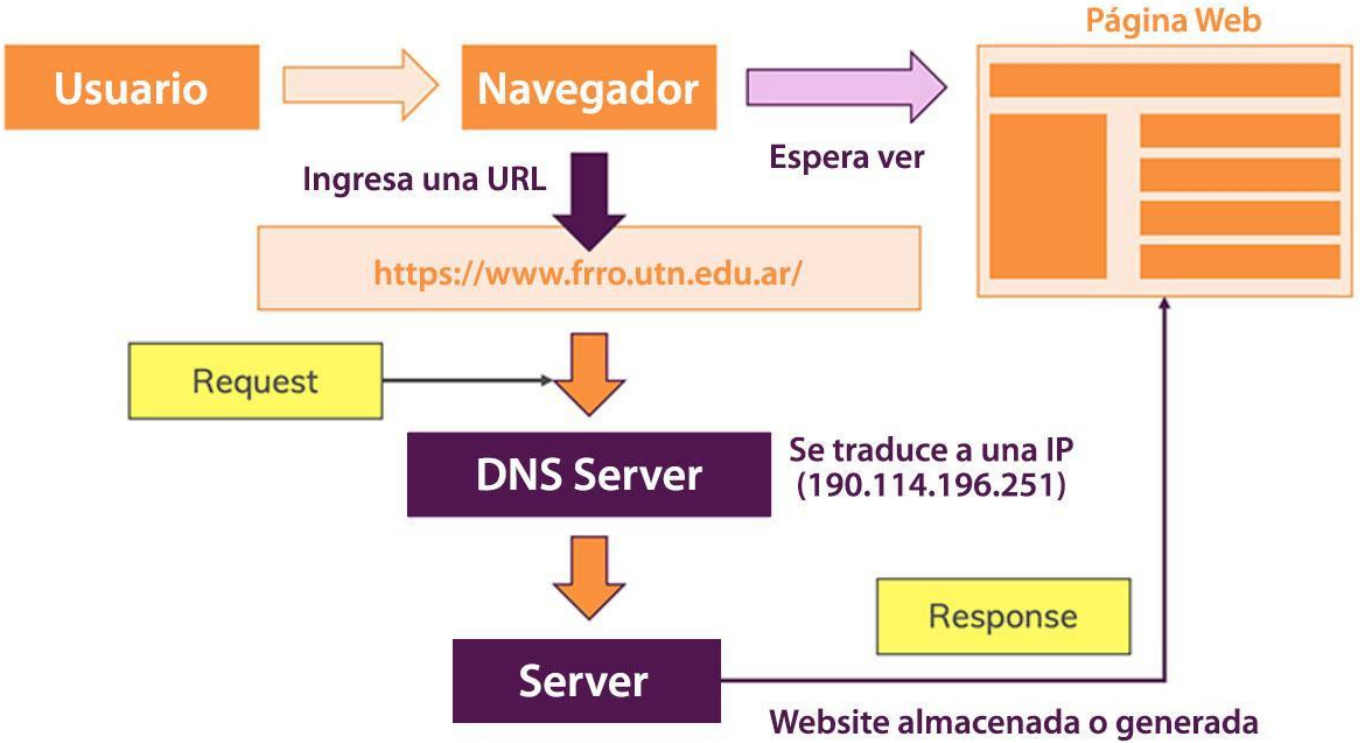
Esta teoría no es esencial para escribir código web a corto plazo, pero en poco tiempo comenzará a beneficiarse realmente al comprender lo que sucede en segundo plano.

El funcionamiento de la web proporciona una vista simplificada de lo que sucede cuando visualiza una página web en un navegador web en su computadora o teléfono.

En marzo de **1989**, Tim Berners-Lee con la idea de que las computadoras puedan compartir información entre sí, presenta un documento con las bases que fue rechazado por su jefe. Aún así, en octubre de **1990** Tim había escrito las tres tecnologías fundamentales que siguen siendo la base de la web actual (y que es posible que haya visto aparecer en partes de su navegador web):

- **HTML**: lenguaje de marcado de hipertexto. El lenguaje de marcado (formato) para la web.
- **URI/URL**: Identificador uniforme de recursos. Una especie de "dirección" que es única y que se utiliza para identificar cada recurso en la web.
- **HTTP/HTTPS**: Protocolo de transferencia de hipertexto. Permite la recuperación de recursos vinculados de toda la web.

Fuentes: https://webfoundation.org/, https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works

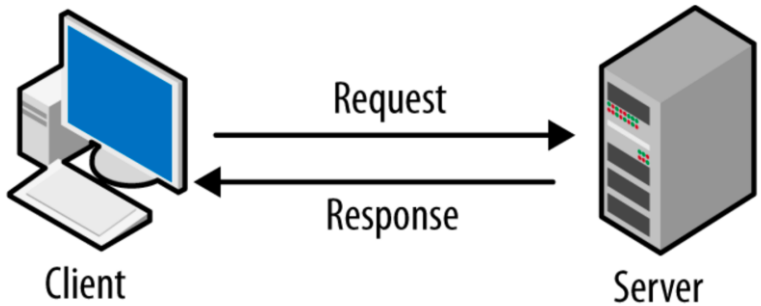

Página Web
Usuario
Navegador
Espera ver
Ingresa una URL
https://www.frro.utn.edu.ar/
Request
DNS Server
Se traduce a una IP
(190.114.196.251)
Response
Server
Website almacenada o generada

Resumen - Conceptos más importantes:

- **Conexión a Internet**: le permite enviar y recibir datos en la web.
- **DNS:** los servidores de nombres de dominio son como una libreta de direcciones para sitios web. Cuando escribe una dirección web (**URL**) en su navegador, el navegador busca en el DNS para encontrar la dirección real (**IP**) del sitio web.
- Se basa en una **arquitectura cliente-servidor**, utilizando los protocolos y herramientas que provee internet para su acceso.
- **Archivos de componentes**: un sitio web se compone de muchos archivos diferentes. Estos archivos vienen en dos tipos principales:
    - **Archivos de código**: los sitios web se crean principalmente a partir de **HTML, CSS y JavaScript**.
    - **Assets (activos)**: este es un nombre colectivo para todas las demás cosas que componen un sitio web, como imágenes, música, videos, documentos de Word y archivos PDF.

Esta arquitectura propone un sistema informático distribuido, en el cual las tareas se dividen entre la máquina **servidor** y la máquina **cliente**.

- Los **clientes** solicitan un servicio/información.
- Los **servidores** se ocupan de recibir esas peticiones, procesar la solicitud y proporcionar una respuesta acorde.



Las ventajas que propone este modelo son:

- Mejor distribución de la información.
- Posibilidad de compartir recursos a diferentes plataformas.
- Capacidad de acceder y procesar datos en ubicaciones remotas.
- Fácil mantenimiento.
- Control centralizado de seguridad.

Algunas desventajas son:

- Sobrecarga de servidores.
- Al ser una arquitectura centralizada en el servidor, si este falla, ningún cliente podrá satisfacer sus pedidos.

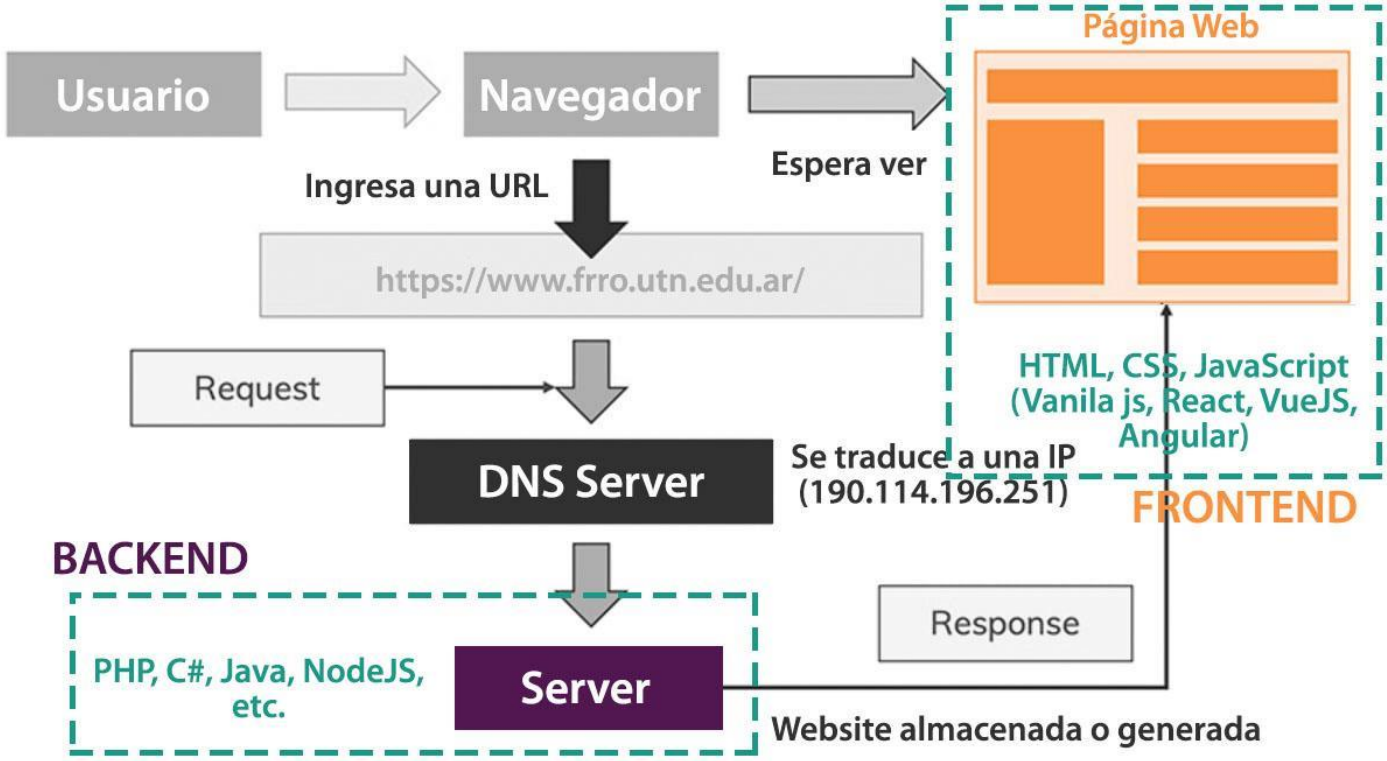
Es un **software** que permite acceso a la web.

Un usuario de la web ingresa la dirección donde está alojado el documento/recurso deseado, el navegador se ocupa de realizar las peticiones al servidor donde se encuentra el contenido y renderiza el mismo en la pantalla.

Componentes de los navegadores:

- Interfaz de usuario.
- Motor de navegación.
- Motor de renderización.
- Interfaz de red.
- Intérprete de JS.
- Persistencia de datos.

Mas info: https://www.html5rocks.com/es/tutorials/internals/howbrowserswork/


Página Web
Usuario
Navegador
Espera ver
Ingresa una URL
https://www.frro.utn.edu.ar/
Request
HTML, CSS, JavaScript
(Vanila js, React, VueJS,
Angular)
DNS Server
Se traduce a una IP
(190.114.196.251)
FRONTEND
BACKEND
Response
PHP, C#, Java, NodeJS,
etc.
Server
Website almacenada o generada

Backend

- Aplicaciones que corren/se ejecutan en el servidor.
- Se encargan de procesar las peticiones y generar el contenido que entregará a los clientes (aplicaciones Frontend) o a otros Backends.
- Tienen comunicación directa con las bases de datos y recursos comunes a todos los clientes.
- Lenguajes/frameworks:
  - PHP (Laravel, Symfony)
  - Javascript (NodeJS, Express)
  - C# (ASP.NET)
  - Java (Spring)

Frontend

- Aplicaciones que corren/se ejecutan del lado del cliente.
  - Ejemplos: Páginas WEB, Aplicaciones.
- Es la parte del sistema que interactúa directamente con el usuario.
- Algunos frameworks CSS:
  - Bootstrap
  - Tailwind
  - Bulma
- Algunos frameworks JavaScript:
  - jQuery
  - React
  - VueJS
  - Angular

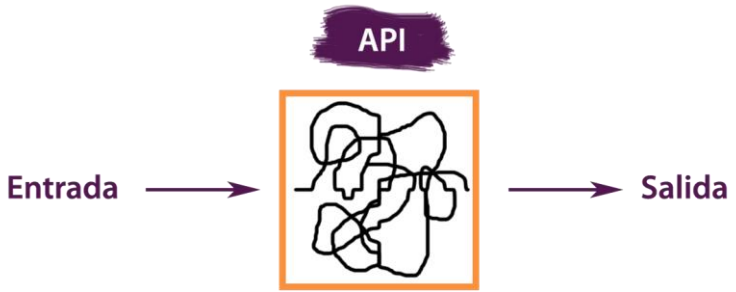
Fullstack: desarrollo de aplicaciones Backend y Frontend.

API: Interfaz de Programación de Aplicaciones.

Permite la comunicación entre dos sistemas desarrollados en lenguajes completamente diferentes.

Su objetivo principal es que diferentes sistemas puedan compartir información/funcionalidades sin saber exactamente cómo se procesa esa información o cómo se lleva a cabo la funcionalidad.

En otras palabras, es un lenguaje en común que definen dos sistemas para comunicarse entre sí.

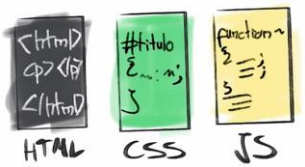


Ejemplos de APIs

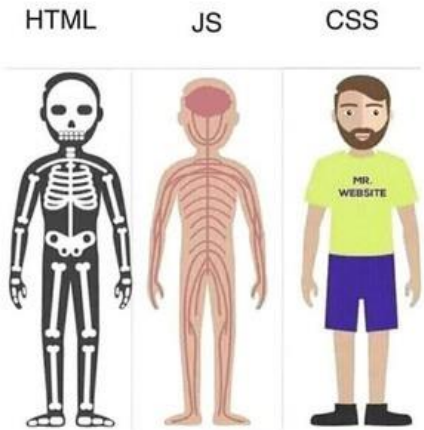
APIs Web - API REST

Mas info: https://developer.mozilla.org/es/docs/Learn/JavaScript/Client-side_web_APIs/Introduction

Introducción a los lenguajes de programación

- Lenguajes de marcado (**HTML**, XML, JSON)
- Lenguajes de hoja de estilo (**CSS**)
- Lenguajes interpretados (**JS**, Python)
- Lenguaj 

C++, Java)



HTML es un **lenguaje de marcado** que utiliza una serie de códigos llamados **etiquetas** que van definiendo los elementos que componen una página web, es decir, su estructura: texto, imágenes, etc. Estas etiquetas serán **interpretadas** por un programa navegador de internet que mostrará adecuadamente la página web al usuario.

CSS “Hojas de Estilo en Cascada” es un lenguaje de diseño gráfico y permite **aplicar estilos** (color, posición, tamaño, etc.) a los distintos elementos HTML de las páginas web, de modo que los títulos, listas y párrafos pueden verse igual en todas y cada una de las páginas.

Javascript fue creado para “**dar vida a las páginas web**”.

Los programas en este lenguaje se llaman **scripts**. Se pueden escribir directamente en el HTML de una página web y ejecutarse automáticamente a medida que se carga la página.

Los scripts se proporcionan y ejecutan como texto plano. **No necesitan preparación especial o compilación para correr**.