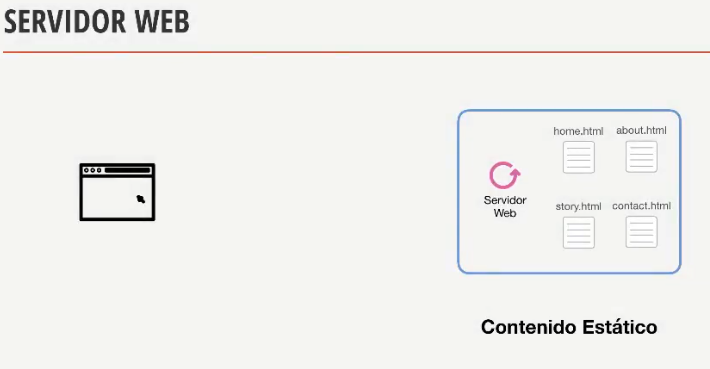
**Servidores y aplicaciones Web**

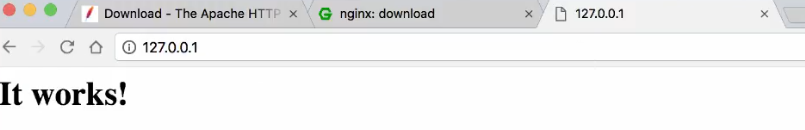


Son de código abierto y gratuitas.



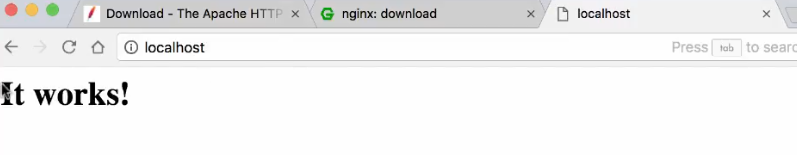
Un servidor es una aplicación que se instala en una máquina. Se ejecuta y sigue corriendo de fondo escuchando las nuevas peticiones que puedan surgir.

Si lo instalamos en nuestra propia computadora podemos acceder en una IP especial 127.0.0.1

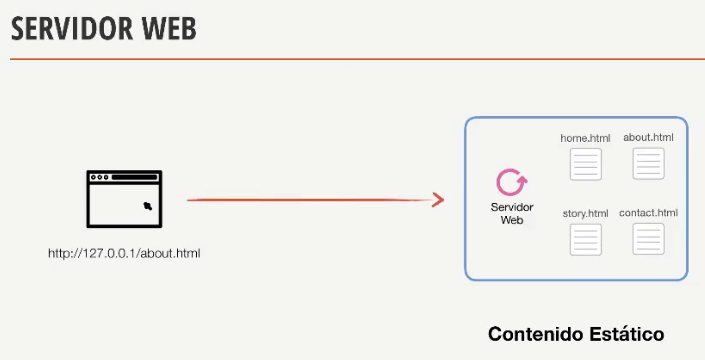


“It works “es la respuesta del servidor (Apaache en este caso).

Tambien podemos acceder con localhost

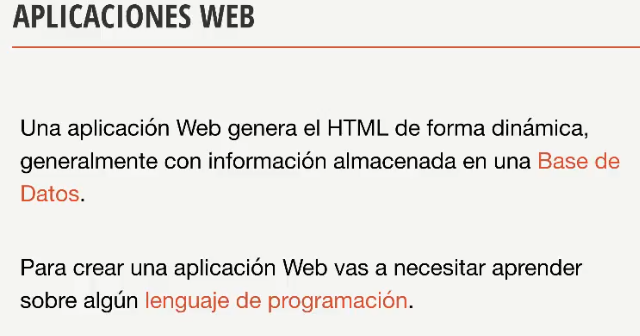


Cuando uno solicita un recurso, el servidor mira si existe ese archivo



Si lo tiene devuelve ese contenido. Si no lo encuentra devuelve el mensaje de error 404.

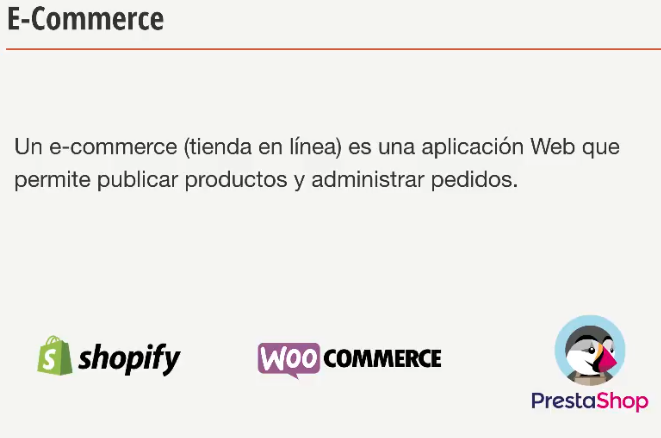
A esto se le llama **contenido estático**: la única forma de canviar el contenido html es entrar a ese archivo y modificarlo. Por otro lado, las **aplicaciones web** en vez de retornar el html que se encuentra en archivos, van a generar el html de forma **dinámica** (a través de lenguajes de programación)



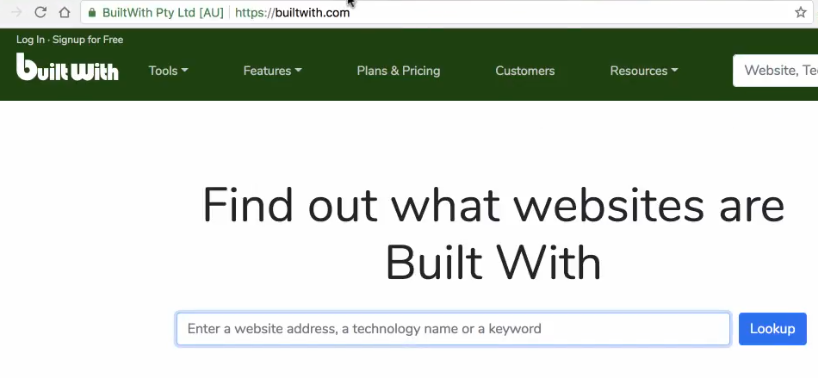
Las aplicaciones Web tienen dos formas de hacerlo:

CMS y e-comerce

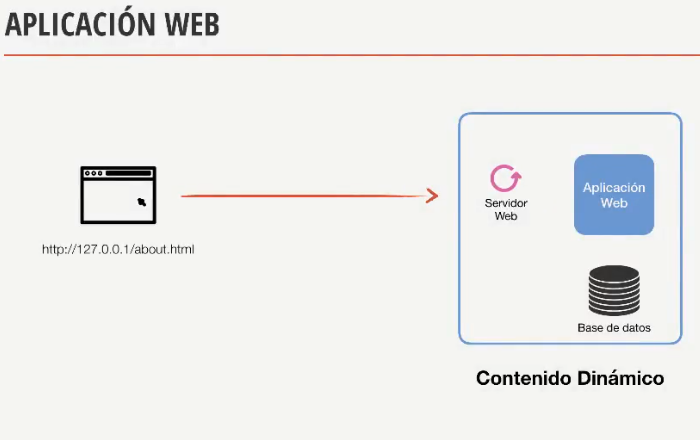




Para saber las tecnologías que está utilizando una página WEB:

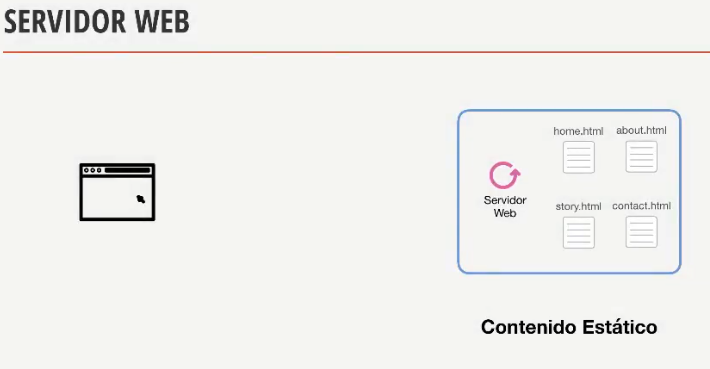


Nos dará información CMS, servidor, estadísticas de CMS y e-comerce..



En una **aplicación Web** además del servidor Web tendremos una aplicación Web para generar **contenido dinámico.**

En el caso de servidor Web solo devuelve **contenido estático**



El servidor normalemente tiene asignado el puerto 80. La web de mejorandola utiliza Node.js en el servidor



Satck más usual del servidor:

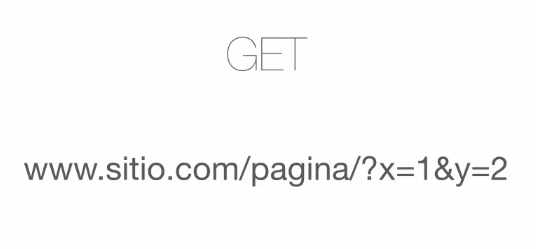
LAMP: Linux,Apache;Mysql;php



Técnicas de Conexión desde JS /HTML para manipular datos con el servidor



GET pasa variables a través de URL. Organizamos una serie de variables (hasta 1024 caracteres)



? variable=valor & variable2=valor2

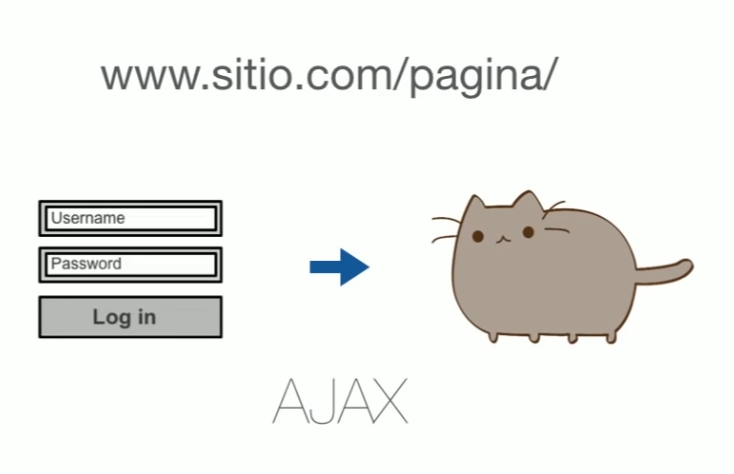
Hay que vigilar la seguridad con el método GET. Si queremos pasar contraseñas o datos sensibles por GET aparecerían en el URL. Aparecería por ejemplo en el historial.





POST no muestra datos en la URL. Los datos van en los headers HTTP.

|  |  |  |
| --- | --- | --- |
| **MÉTODO** | **CONCEPTO** | **OBSERVACIONES** |
| **GET** | GET lleva los datos de forma "visible" al cliente (navegador web). El medio de envío es la URL. Los datos los puede ver cualquiera. | Los datos son visibles por la URL, por ejemplo:  www.aprenderaprogramar.com/  action.php?nombre=pedro&apellidos1= gomez |
| **POST** | POST consiste en datos "ocultos" (porque el cliente no los ve) enviados por un formulario cuyo método de envío es post. Es adecuado para formularios. Los datos no son visibles. | La ventaja de usar POST es que estos datos no son visibles al usuario de la web. En el caso de usar get, el propio usuario podría modificar la URL escribiendo diferentes parámetros a los reales en su navegador, dando lugar a que la información tratada no sea la prevista. |



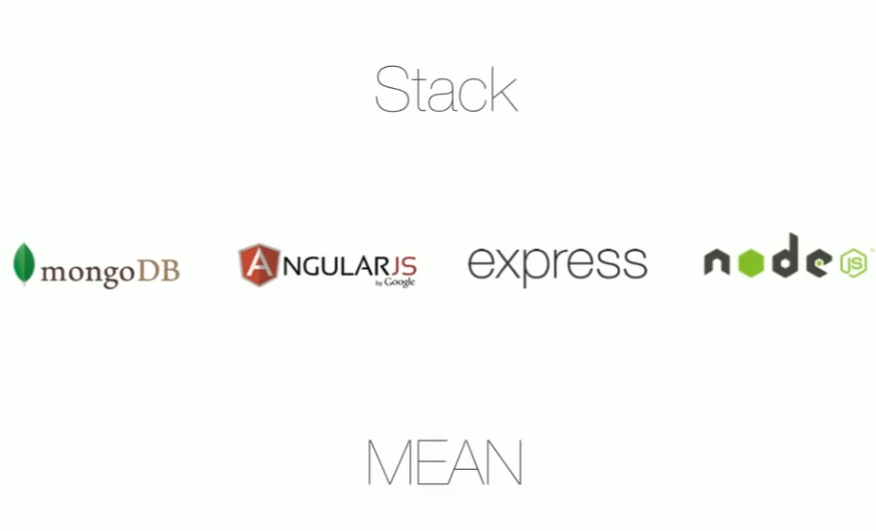
**AJAX**, acrónimo de ***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML* ([JavaScript](https://es.wikipedia.org/wiki/JavaScript) asíncrono y [XML](https://es.wikipedia.org/wiki/XML)), es una técnica de [desarrollo web](https://es.wikipedia.org/wiki/Desarrollo_web) para crear aplicaciones interactivas o [RIA](https://es.wikipedia.org/wiki/Rich_Internet_Application) (Rich Internet Applications). Estas aplicaciones se ejecutan en el [cliente](https://es.wikipedia.org/wiki/Cliente_(inform%C3%A1tica)), es decir, en el [navegador](https://es.wikipedia.org/wiki/Navegador_web) de los usuarios mientras se mantiene la **comunicación** [**asíncrona**](https://es.wikipedia.org/wiki/As%C3%ADncrono) con el **servidor en segundo plano**. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y [usabilidad](https://es.wikipedia.org/wiki/Usabilidad) en las aplicaciones.

Actualmente no se usa XML, se usa JSON.

Los tres métodos GET,PoST,AJAX no tienen un flujo constante con el servidor.

Sockets mantienen una conexión permanente en el servidor. **Socket** designa un concepto abstracto por el cual dos programas (posiblemente situados en computadoras distintas) pueden intercambiar cualquier flujo de datos, generalmente de manera fiable y ordenada.

Hay otros STAKS, como



Stack de mejorandola-la



STACK ultracomprimido para hacer volar un dron. El stack de desarrollo va a aser NODE.js que va correr en el lado del servidor pero como un servidor local en nuestro compuador. JS va a correr en el navegador. Como intermediario entre los dos usaremos EXPRESS que es la librería de NODE.js que me permite cargar elpuerto 80.

