



# Arquitecturas y lenguajes de programación en la Web

UD1: Tecnologías para el desarrollo de interfaces



# Objetivos de aprendizaje

- Sabrás qué **arquitectura** tienen las aplicaciones web
- Recordarás “qué pinta tiene” **JavaScript**
- Conocerás los principales **navegadores** Web y podrás compararlos
- Realizarás tu **primer script** en JavaScript en segundo curso

# Nacimiento de la Web



Tim Berners-Lee (**CERN**) (1989)

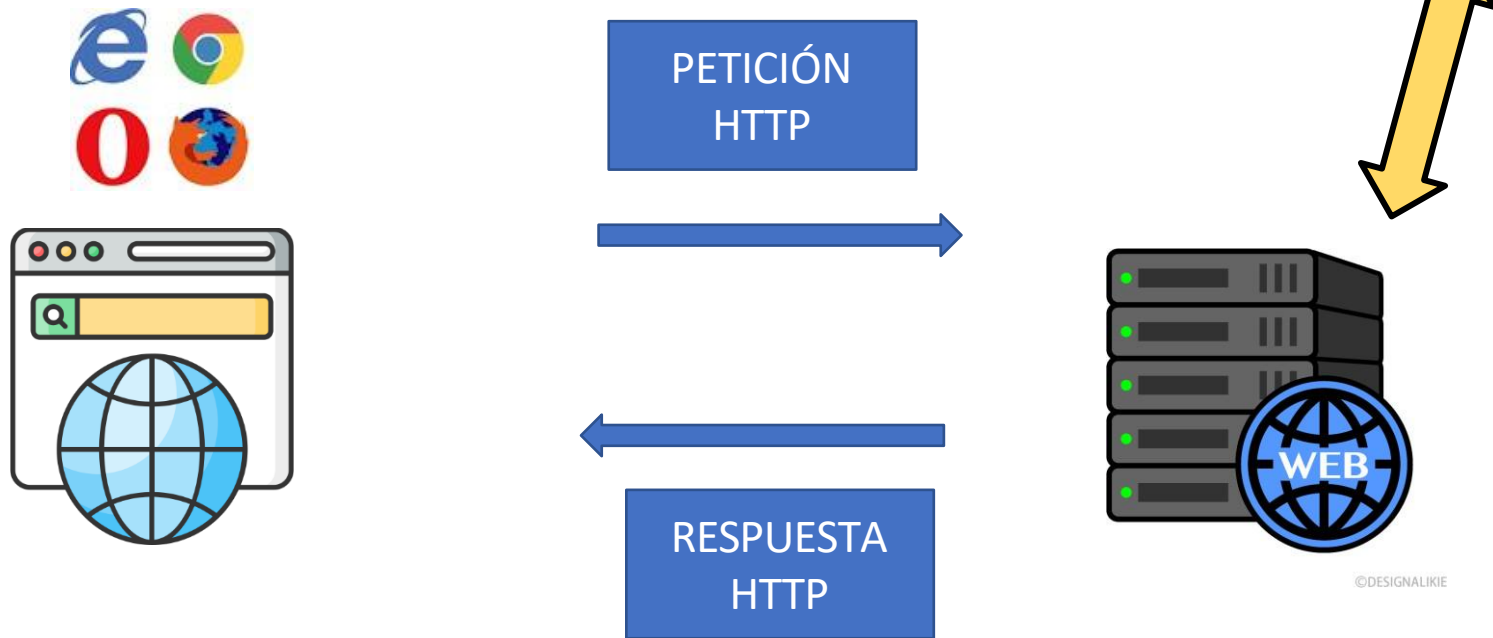
# Estandarización



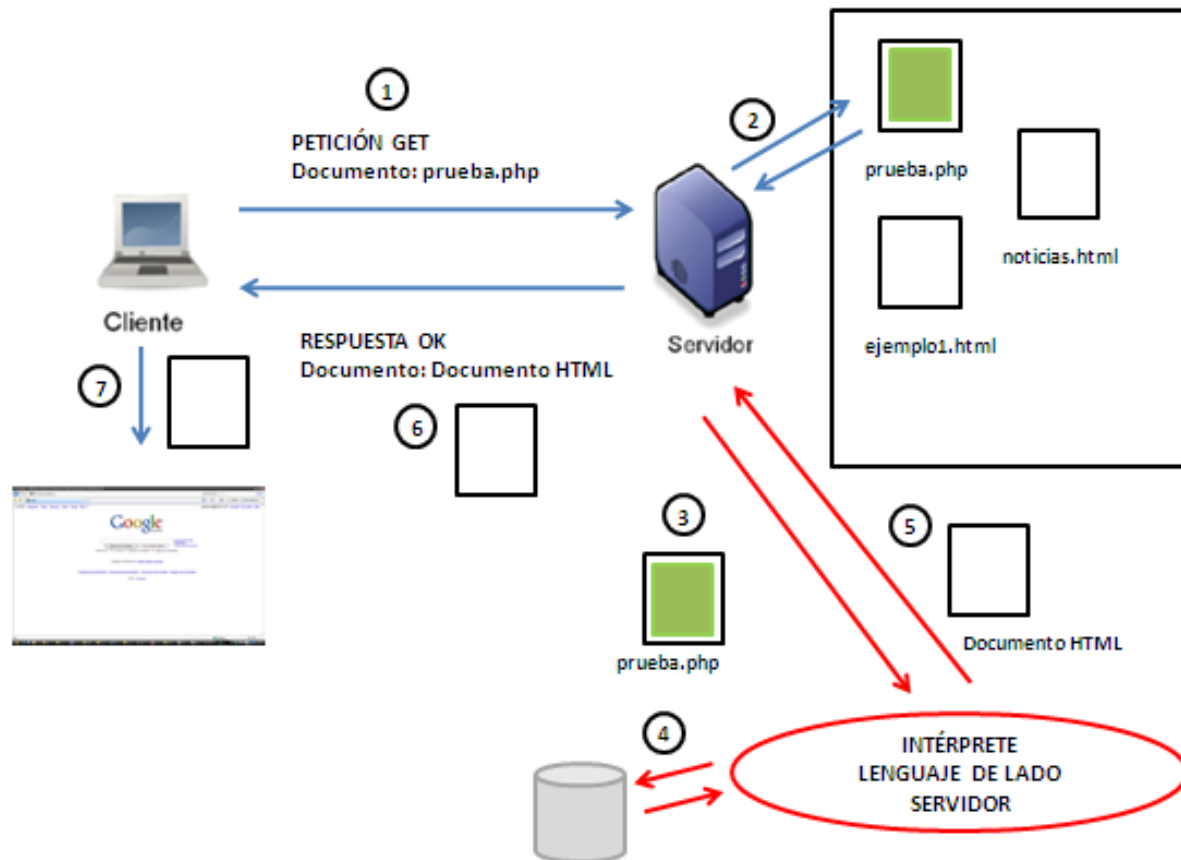
# Desarrollo Web

- Dividimos nuestro código en dos partes:
  - Parte servidora
    - Código que se ejecuta en un equipo remoto (servidor).
    - Funciones principales es el manejo de datos (acceso a bases de datos), autenticación, seguridad...
    - El usuario no tiene acceso al código de esta parte
  - Parte cliente
    - Código que se ejecuta en mi propio dispositivo (navegador Web)
- El diálogo entre cliente y servidor se establece a través del protocolo HTTP

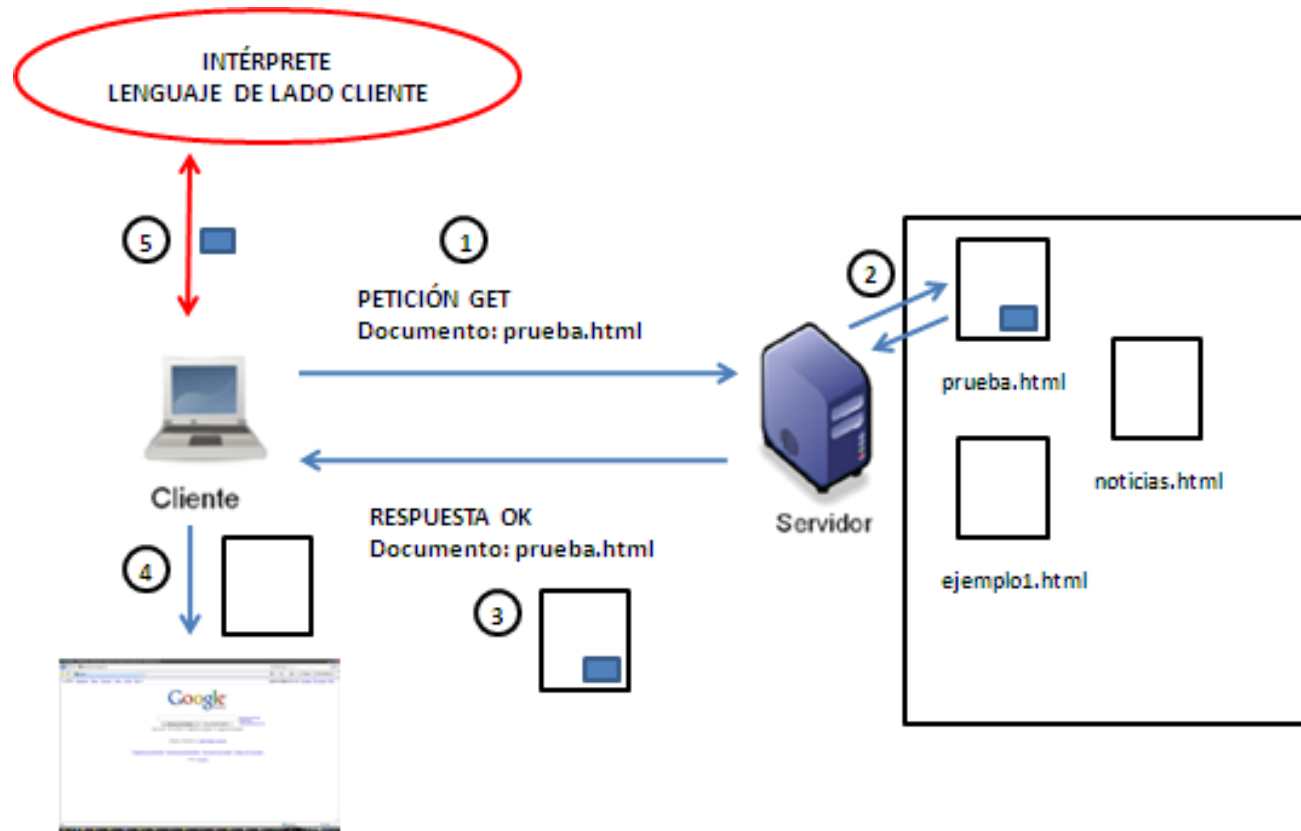
# Arquitectura Web



# Petición de una página Web



# Petición de una página Web



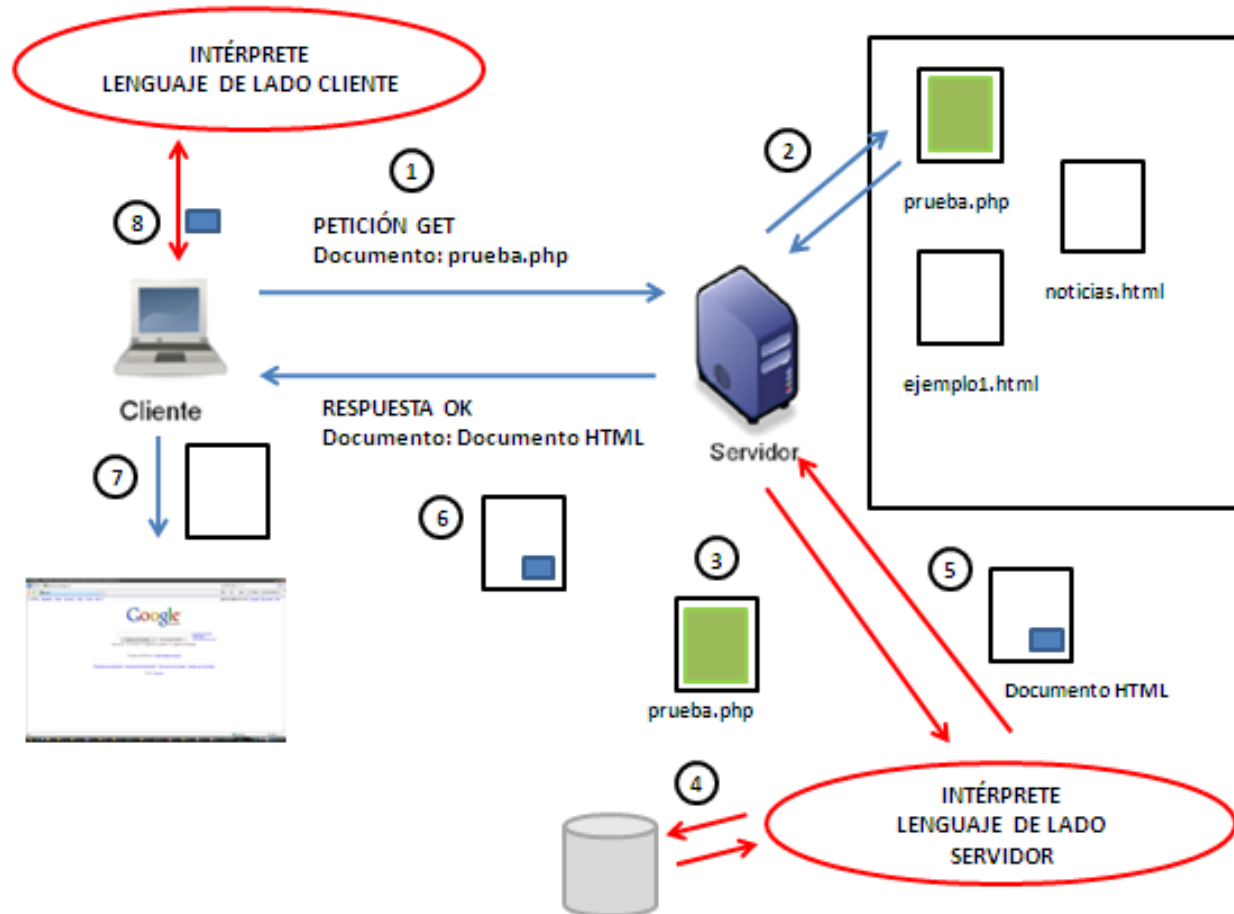
- NOTA: El cuadradito azul representa el código JavaScript



# ¿Cliente o Servidor?



# Cliente y Servidor



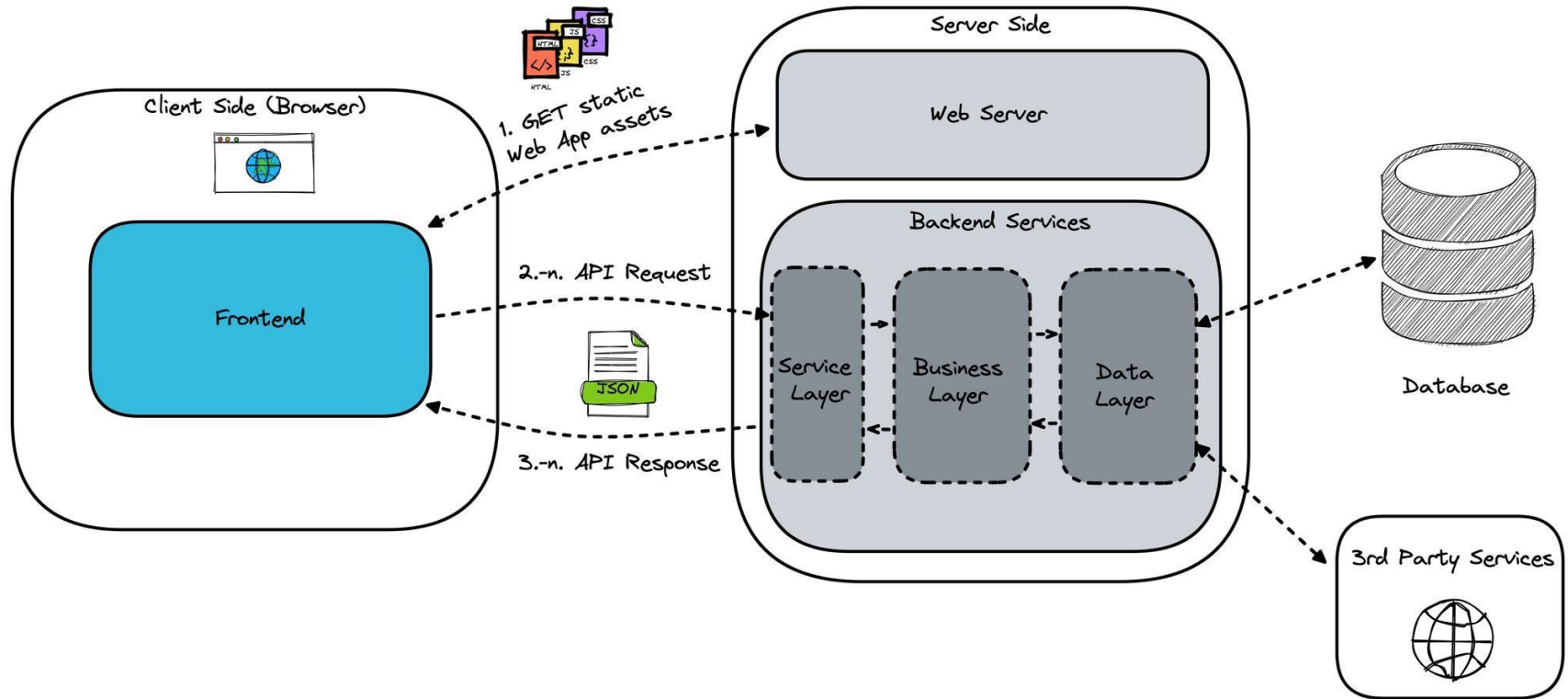
# AJAX

- Es una tecnología a medio camino entre cliente y servidor
- Evita tener que hacer nuevas peticiones de la página completa para cada recurso que se pide
  - Sólo se pide la parte que ha cambiado
  - No es necesario refrescar la página completa, permitiendo aplicaciones mucho más complejas

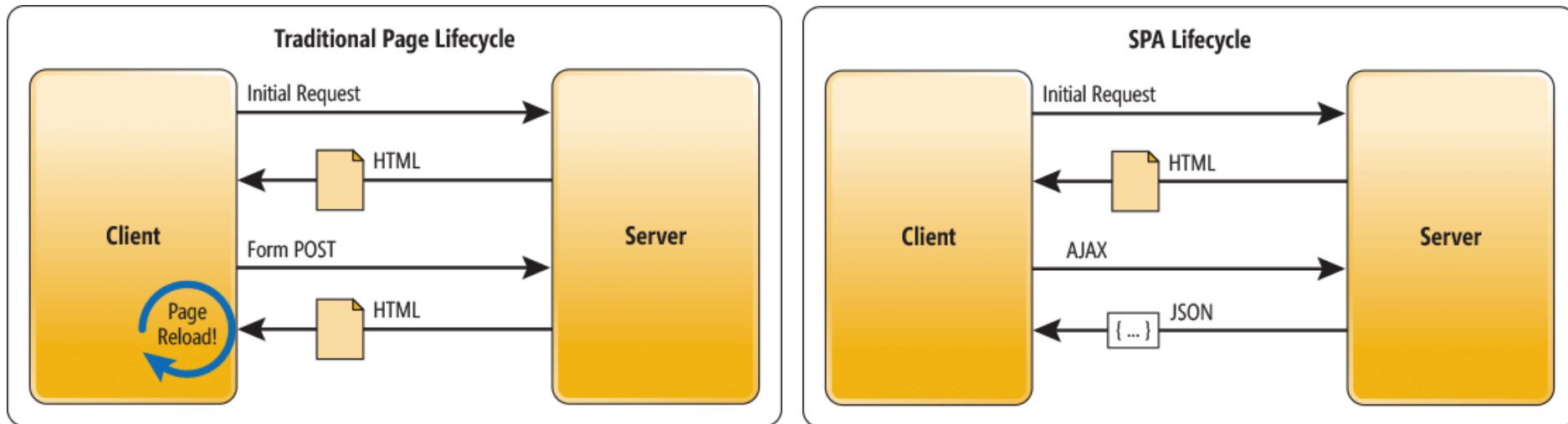
# Servicios web

- Arquitectura web cliente-servidor clásica
  - El cliente realiza llamadas al backend
  - La página se recarga
- Actualmente arquitectura SPA (*Single Page Application*)
  - El cliente gana mucho mayor peso
  - Programación reactiva que accede a servicios remotos REST que realizan las operaciones
    - Comunicación mediante JSON

# Petición de servicios web



# Comparativa



# Tecnologías Web

<p>Lado Cliente</p> 	
<p>Lado Servidor</p>  <small>©DESIGNALIKE</small>	

# Frameworks

- Un framework es un conjunto de utilidades probadas y validadas que me permiten solucionar problemas que me encuentro de manera común al realizar una aplicación.
  - Organización de archivos, separación de funcionalidad, manejo de rutas, seguridad, acceso a datos, manejo de usuarios....
- Cada plataforma de desarrollo suele tener su propio framework asociado.








# Frameworks JS

- Actualmente está muy extendido el uso de 3 frameworks JS
- Son similares pero cada uno tiene sus particularidades y curva de aprendizaje



# Resumiendo..

Mobile Apps	   Flutter  React Native  Xamarin  ionic
Web Frontend	    Bootstrap  jQuery  ANGULAR  React  Vue.js  BACKBONE.JS  ELECTRON
Web Backend	 node  Express  METEOR  GO  RAILS  .NET  .NET Core  Java  php  python  django
Databases	 MySQL  Microsoft SQL Server  PostgreSQL  mongoDB  OrientDB  Cassandra  DynamoDB  neo4j  elastic  redis 
Cloud/ Integration	 amazon web services™  Google Cloud Platform  Microsoft Azure  DigitalOcean  HEROKU