El futuro digital es de todos

MinTIC

Universidad Industrial de Santander

MisiónTIC2022

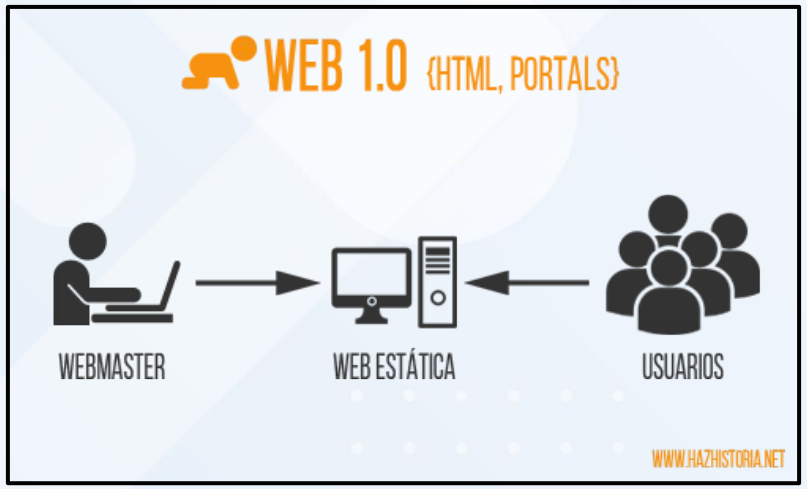
SPA

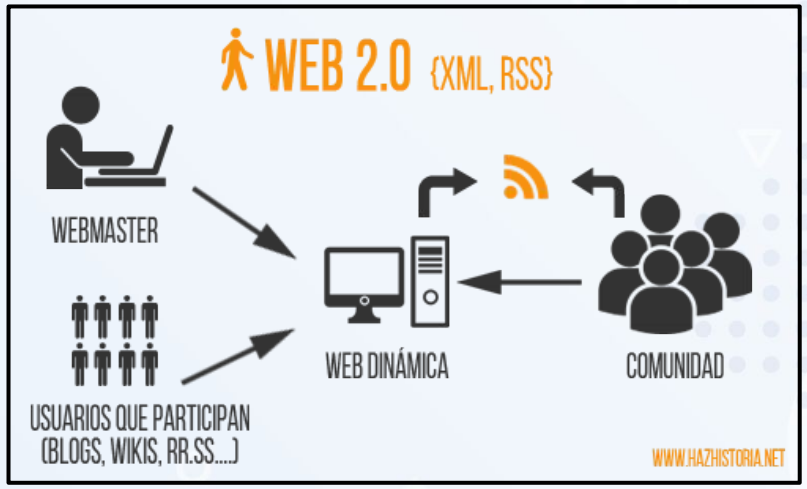
CONTENIDO

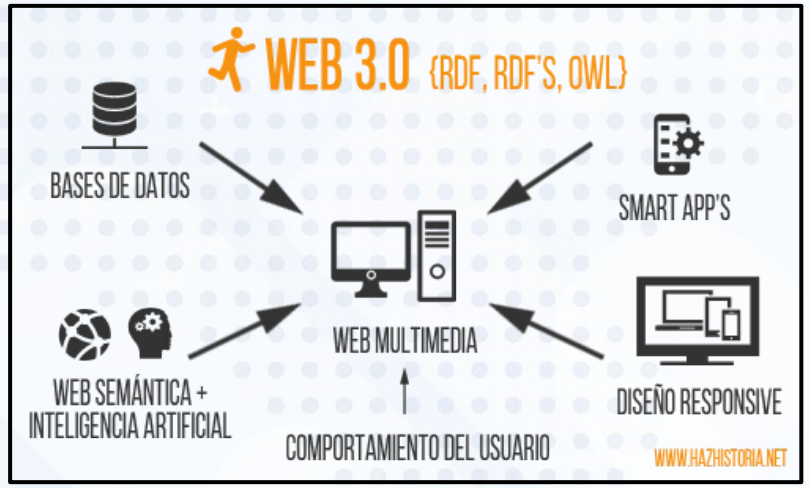
- Single Page Application - SPA
  - SPA vs MPA
  - Arquitectura SPA
  - APIs
  - Herramientas de desarrollo FrontEnd y BackEnd.

Historia

La web ha evolucionado desde su creación el año 1966, de la red Arpanet, hasta el posterior nacimiento del Internet y no ha dejado de cambiar pasando por diferentes etapas las cuales conocemos como web 1.0 a la 2.0, 3.0.







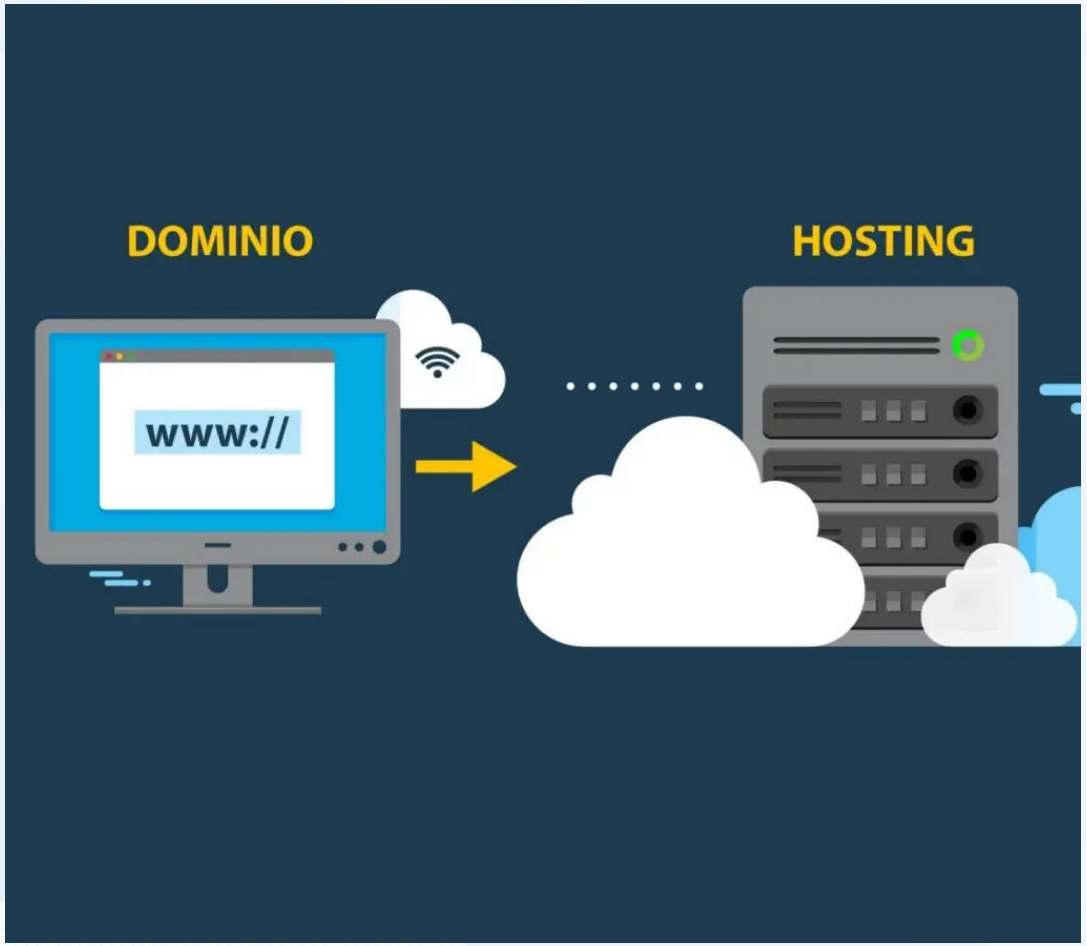
Web 4.0

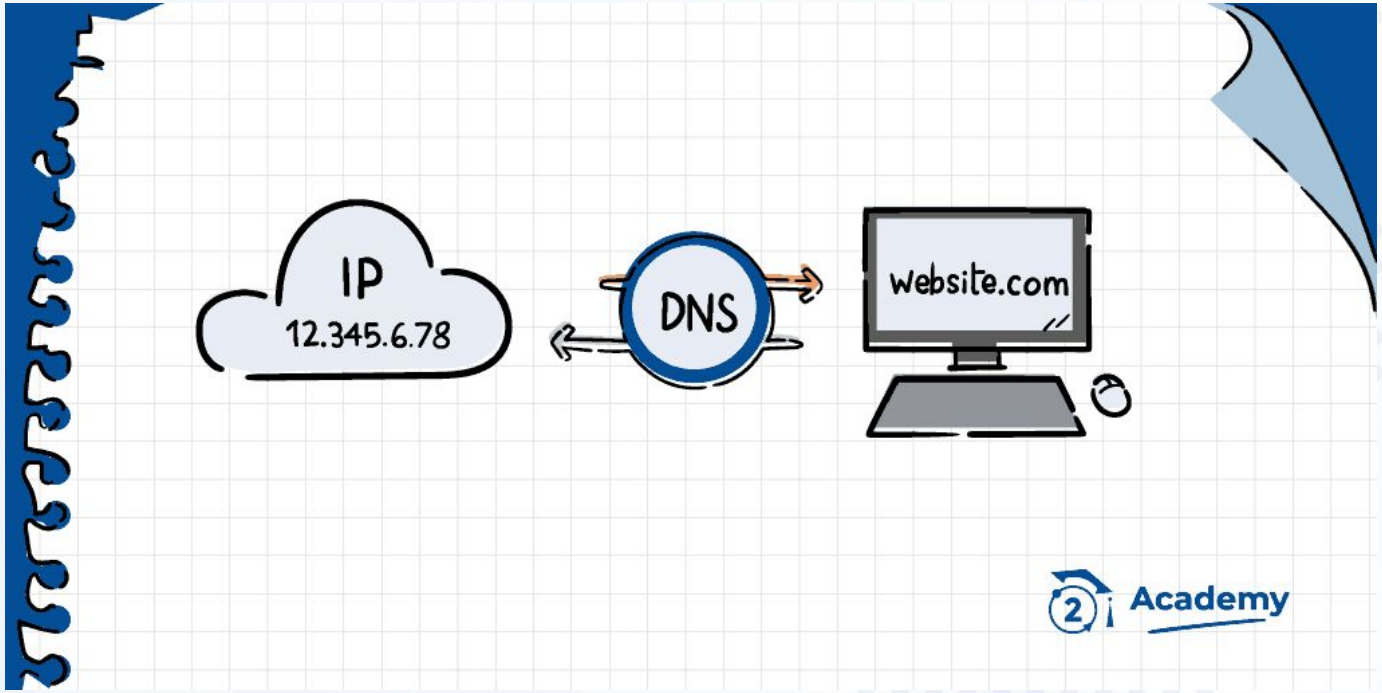
En el 2016, empezó la web 4.0, la cual se centra en ofrecer un comportamiento más inteligente y predictivo, de modo que podamos con sólo realizar una afirmación o petición, poner en marcha un conjunto de acciones que tendrán como resultando aquello que pedimos o decimos.

Esto gracias a tecnologías como Deep Learning y Machine Learning, a través de potentes ordenadores en la nube y procesan los datos, peticiones, etc.

Hoisting - DNS

Hoisting: es un servicio de alojamiento para sitios web.
DNS: Sistema de nombres de dominio





La nube

Una red mundial de servidores, diseñados para almacenar y administrar datos, ejecutar aplicaciones o entregar contenido o servicios...

Microsoft Azure

amazon web services™

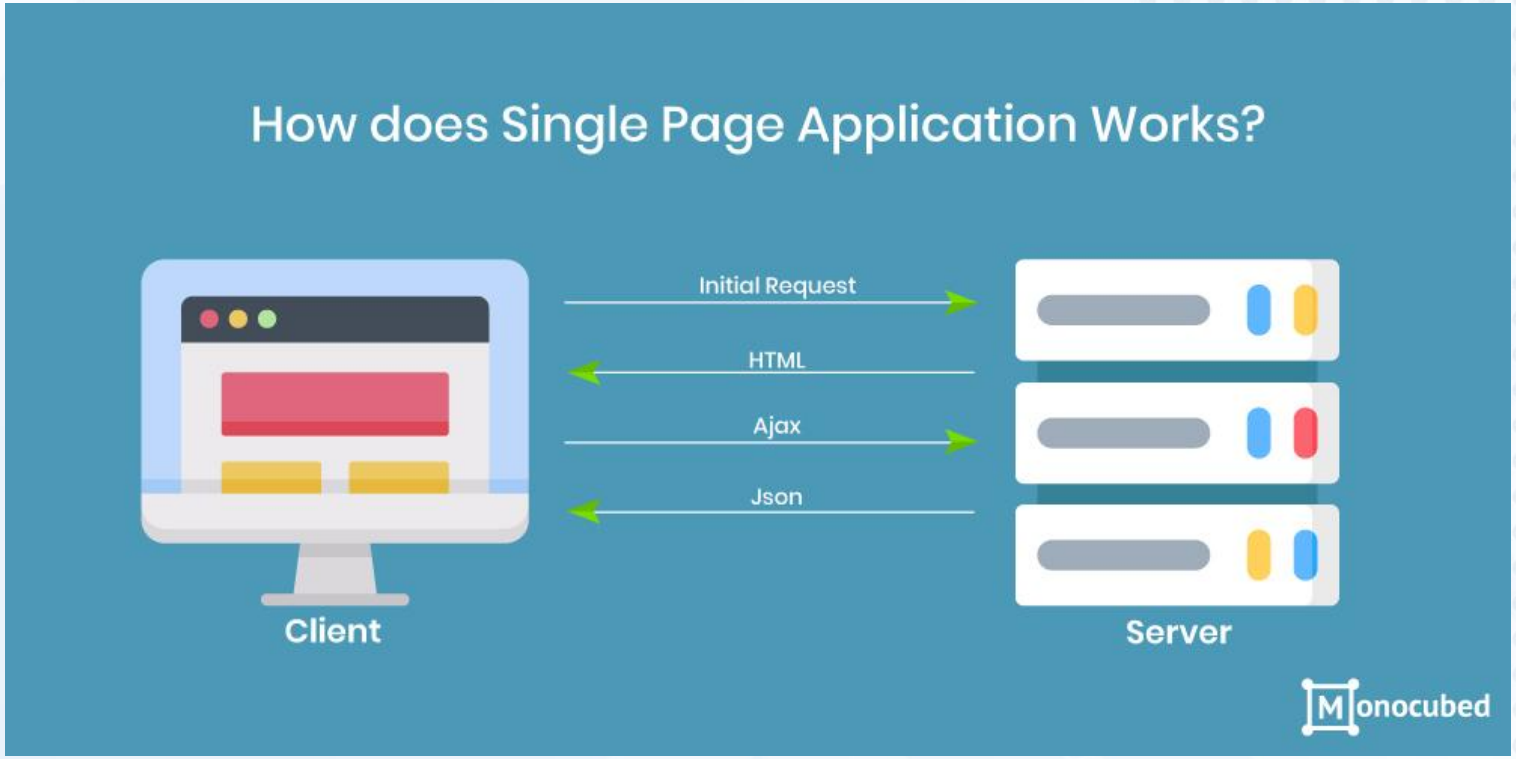
Google Cloud

IBM Cloud

ORACLE Cloud

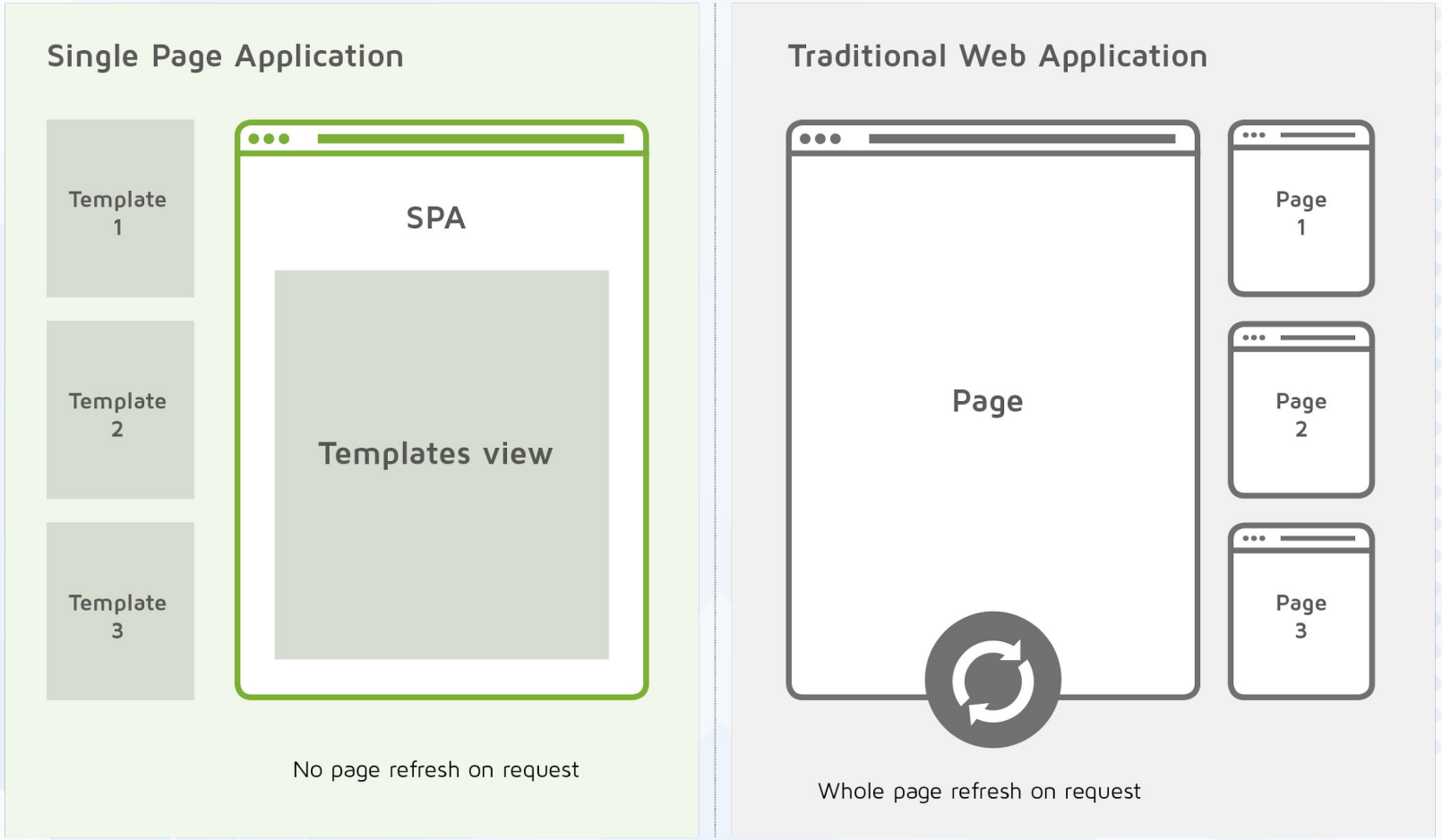
Single Page Application - SPA

Es un tipo de aplicación web que interactúa con el usuario reescribiendo dinámicamente una sola pagina con nuevos datos del servidor, a diferencia del método tradicional de cargar nuevas paginas enteras en el navegador.

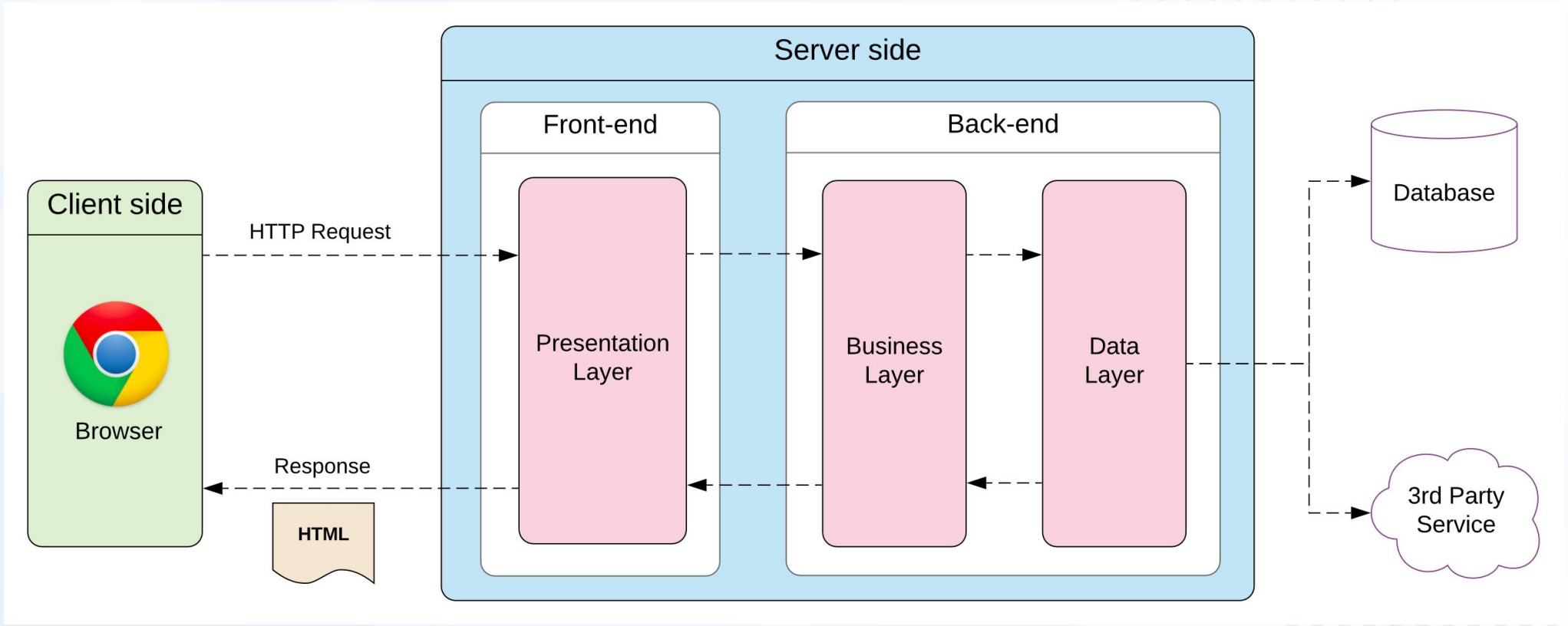


MPA vs SPA

Una SPA nunca se refresca, todo el CSS, HTML y JavaScript necesario es traído en una Single Page Load o cargando los recursos necesarios dinámicamente y añadiendoles usualmente a medida que el usuario interactúa.

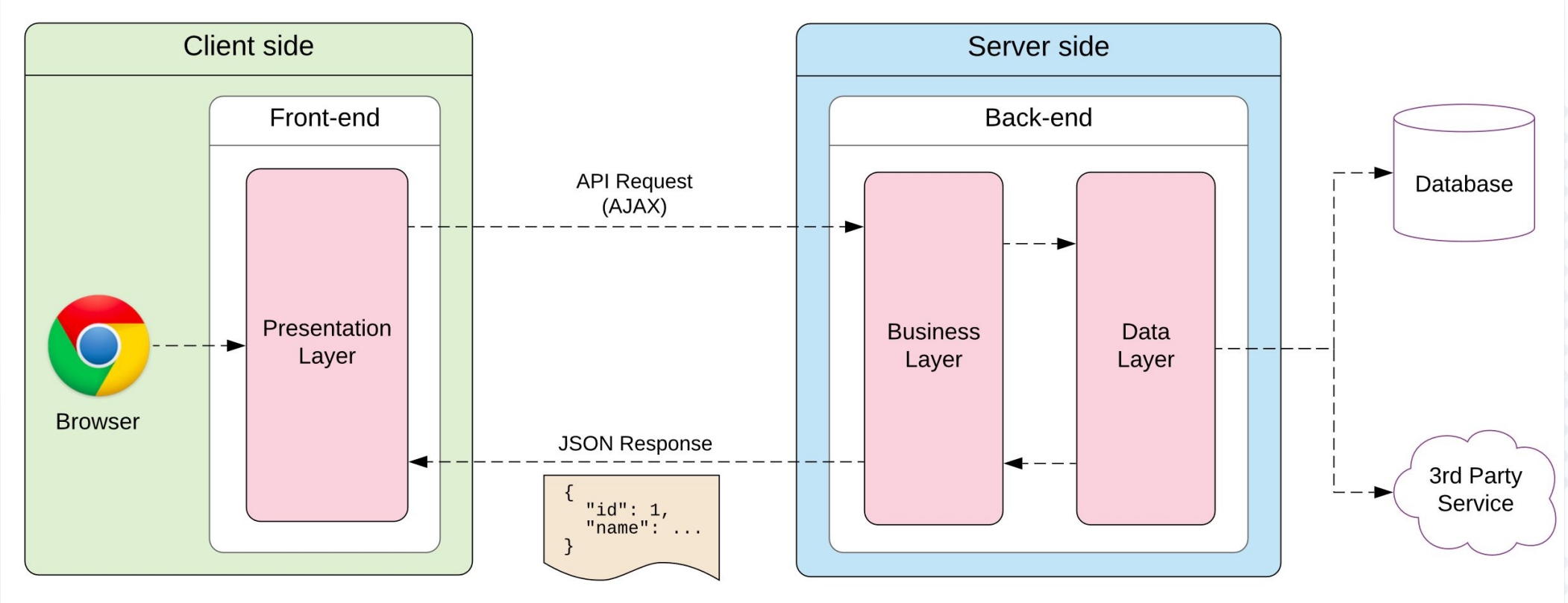


Traditional Web Application



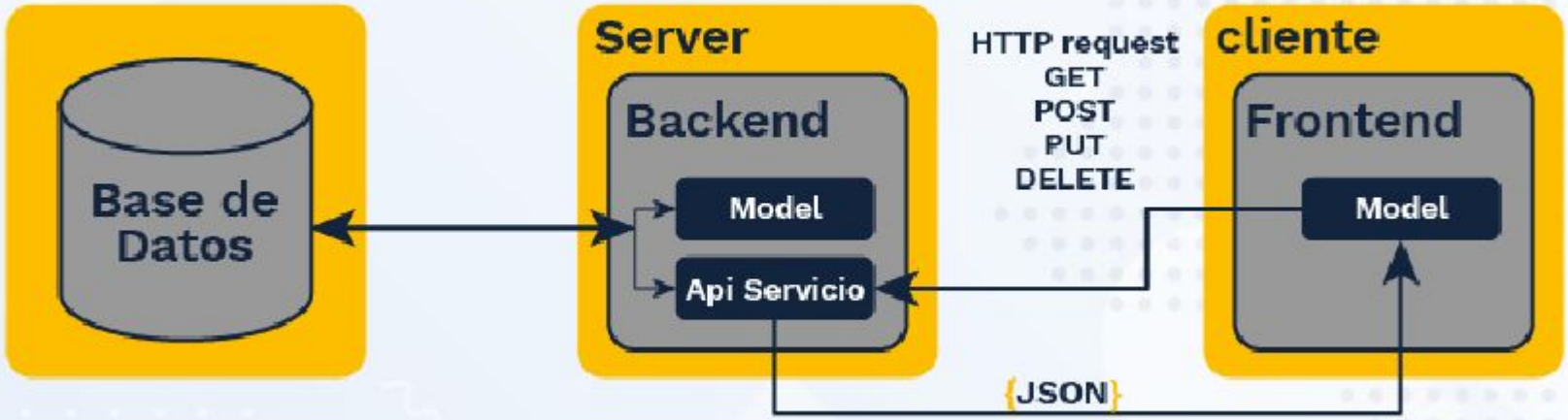
Single- Page Web Application

Varias vistas, no varias paginas



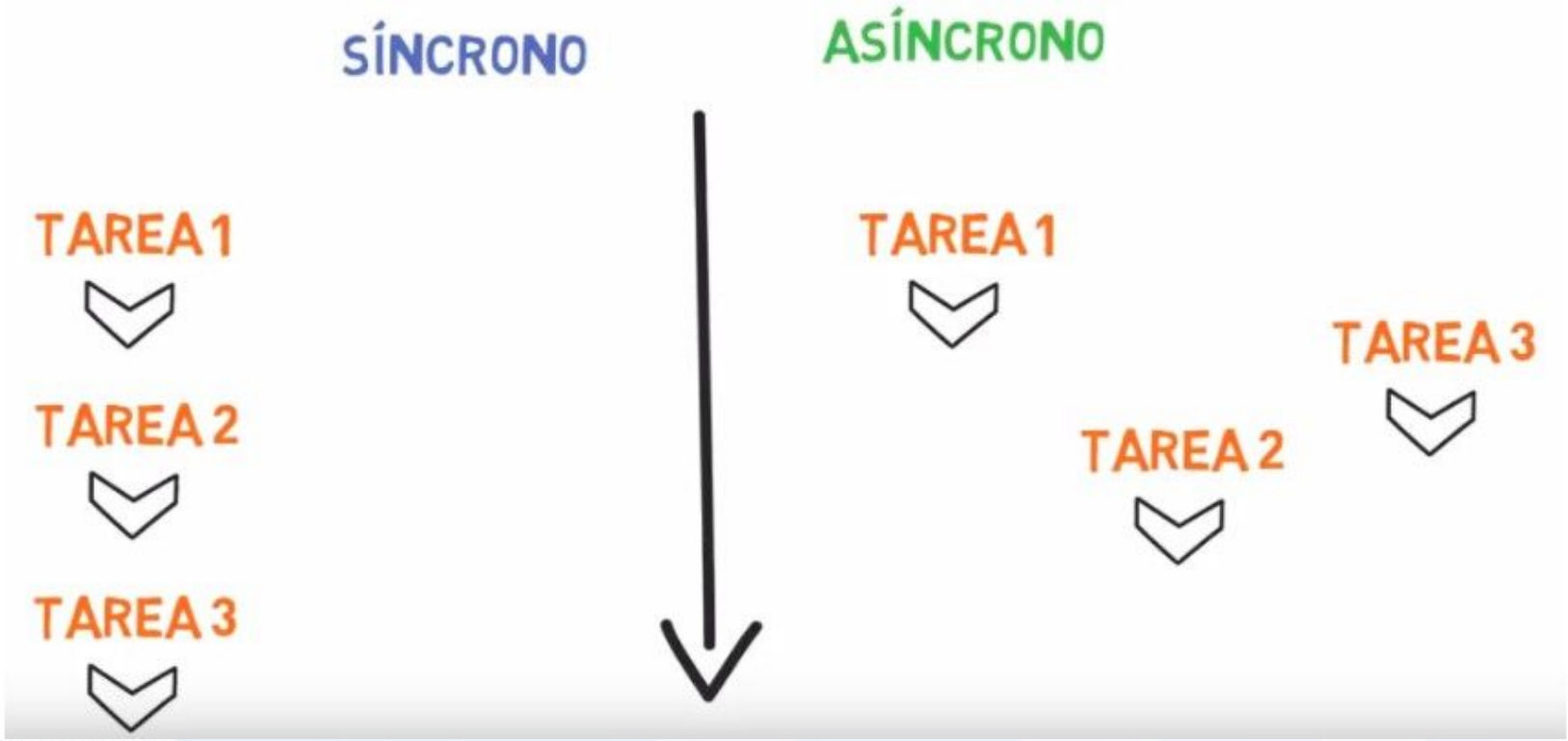
Arquitectura SPA

En SPA la mayor parte de la funcionalidad se lleva al cliente. Lo podríamos ver como un cliente pesado que se carga desde un servidor web y el código en servidor se usa básicamente para proveer de una API RESTful a nuestro código cliente usando Ajax.



Sincronismo - Asincronismo

- Sincrónico: es aquel código donde cada instrucción espera a la anterior para ejecutarse
- Asincrónico: no espera a las instrucciones diferidas y continúa con su ejecución

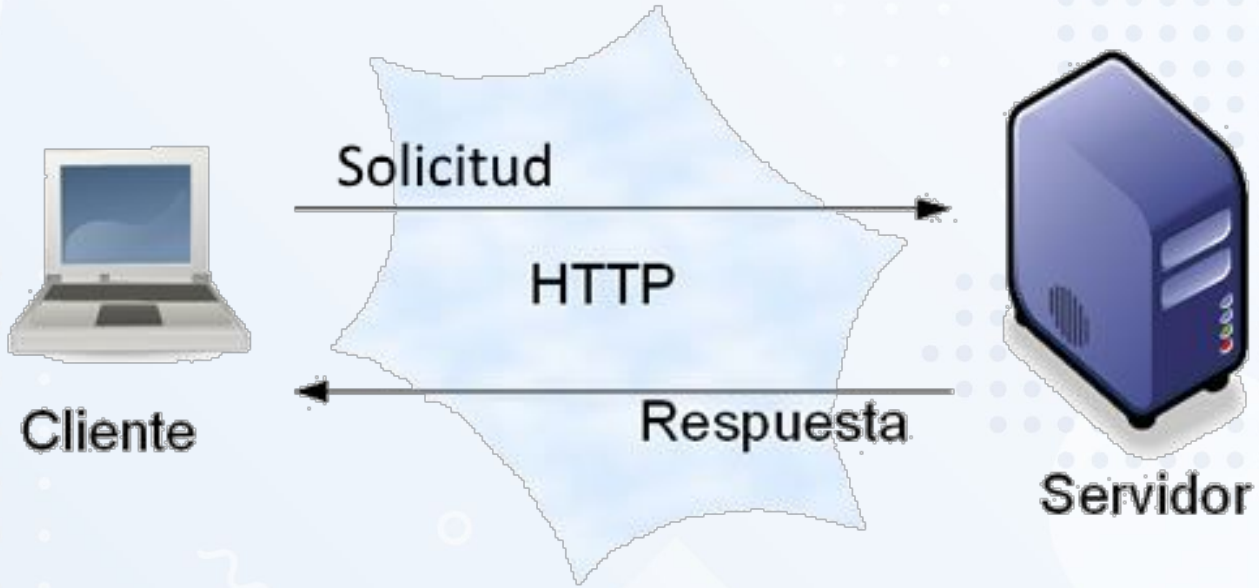


AJAX - JavaScript Asincrónico y XML

Es un conjunto de técnicas de desarrollo web que permiten que las aplicaciones web funcionen de forma asíncrona, procesando cualquier solicitud http al servidor en segundo plano, que permite que cualquier aplicación web que use AJAX, pueda enviar y recuperar datos del servidor sin la necesidad de volver a cargar toda la página.

Peticiones HTTP

Las peticiones HTTP son mensajes enviados por un cliente, para iniciar una acción en el servidor.



Evolución

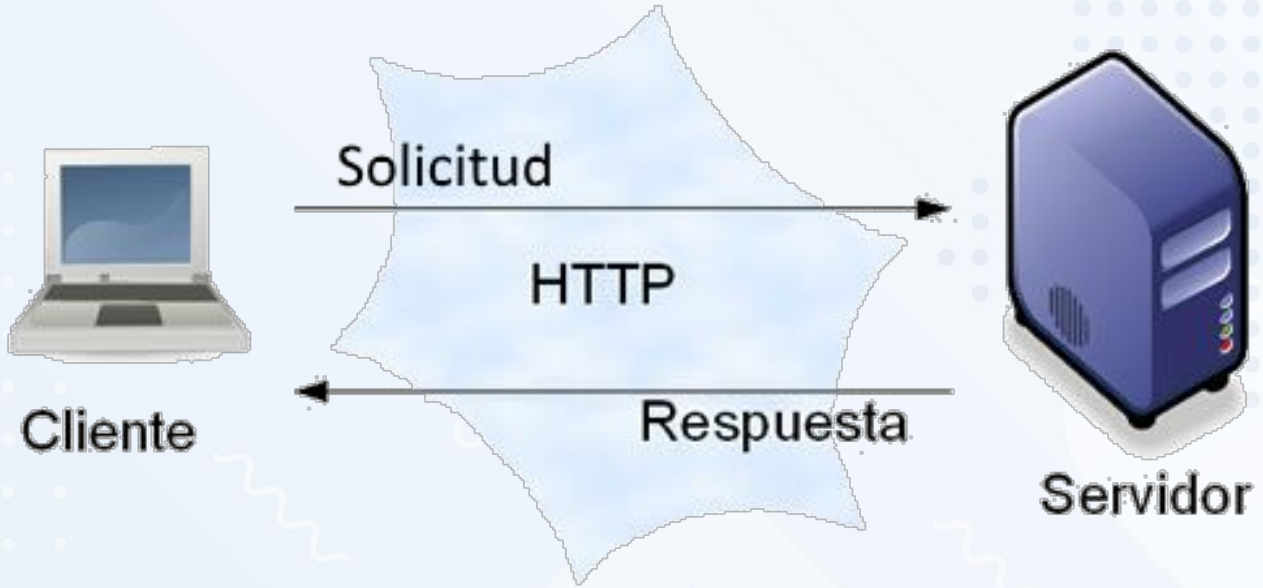

jQuery
write less, do more.


FETCH VS AXIOS
Peticiones HTTP en javascript

Peticiones HTTP

Contiene:

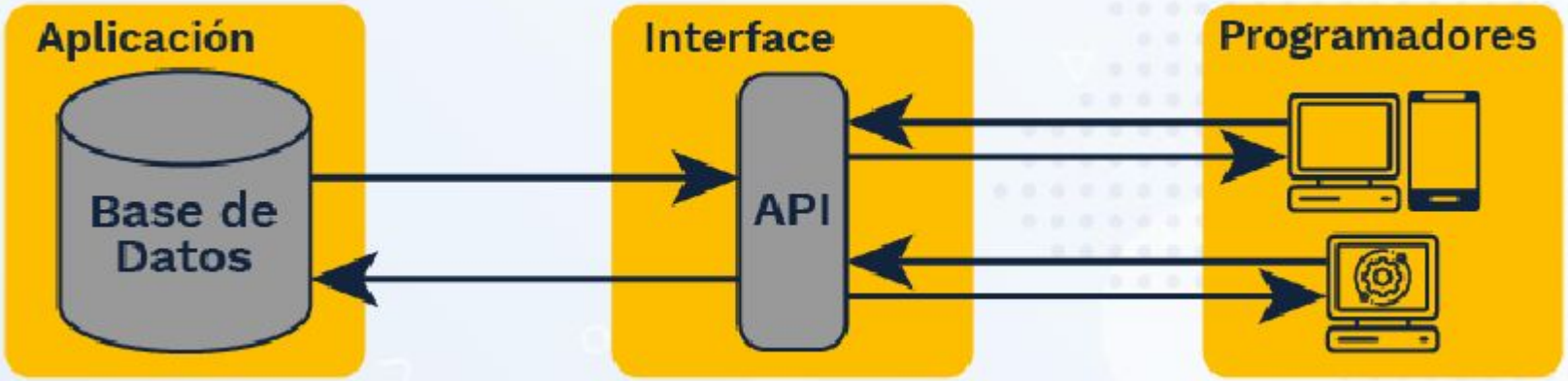
- Url
- Método
- Headers
- Params
- Body



Métodos HTTP

API – Application Programming Interface

Es un conjunto de servicios ofrecidos desde un servidor, que una aplicación puede usar en remoto para enviar o recibir información y así comunicarse con el servidor y pueden ser utilizados por nuestras propias aplicaciones o por aplicaciones de terceros, si es que queremos ofrecerlos



Herramientas

Instalaciones:

- Postman: https://www.postman.com/downloads/
- Node js: https://nodejs.org/es/download/
- MongoDb: https://www.mongodb.com/try/download/community
- Robo3T: https://robomongo.org/download

Bases de datos no relacionales

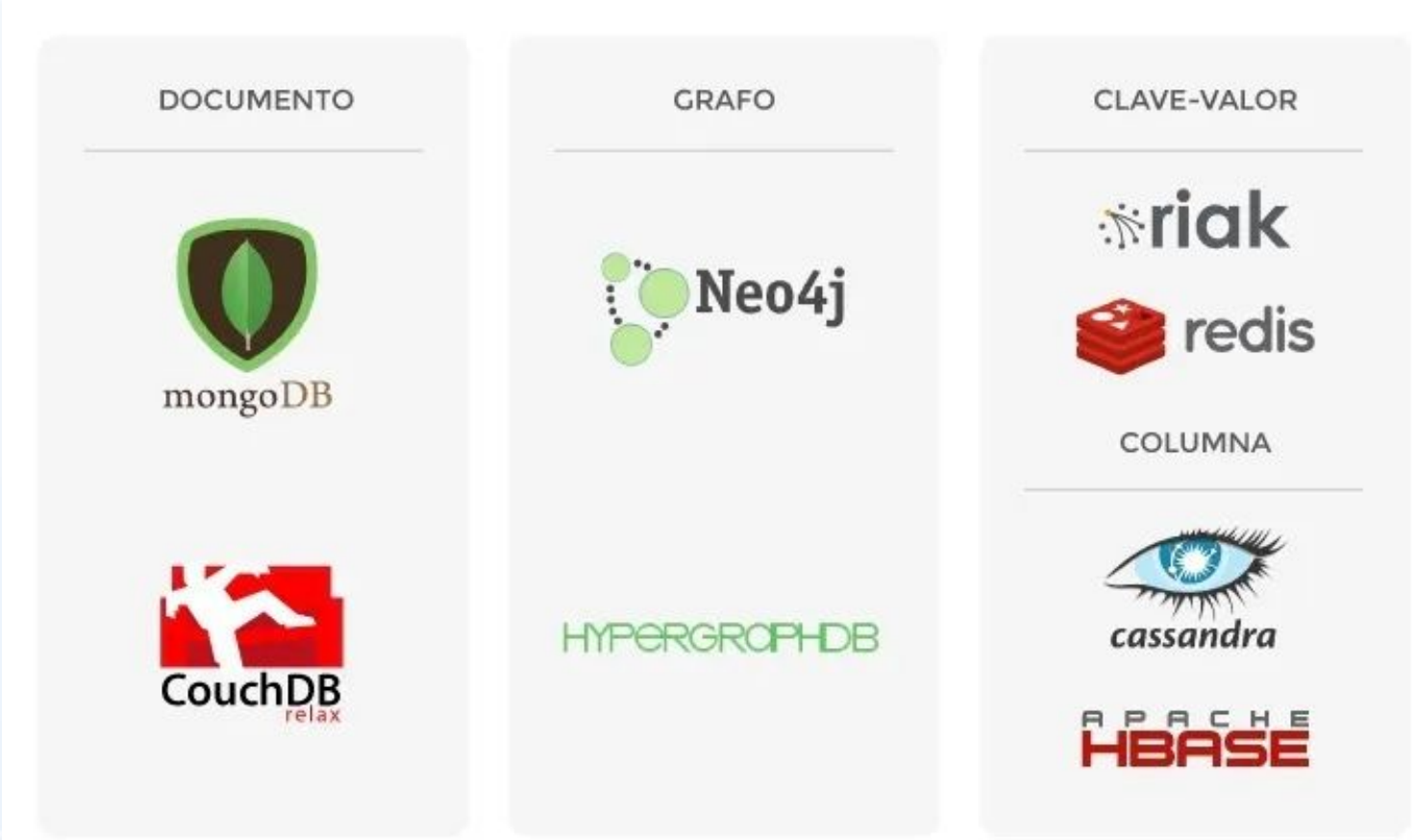
Las bases de datos no relacionales son un sistema de almacenamiento de información que se caracteriza por no usar el lenguaje SQL para las consultas.

- No trabajan con estructuras definidas, es decir, los datos no se almacenan en tablas, y la información tampoco se organiza en registros o campos.
- Tienen una gran escalabilidad y están pensadas para la gestión de grandes volúmenes de datos.
- No utilizan el lenguaje SQL para consultas, aunque sí lo pueden usar como herramienta de apoyo.

Tipos de bases de datos NoSql

Basadas en:

- Documentos
- Grafos
- Clave - valor



BD Relacional vs No Relacional

MongoDB

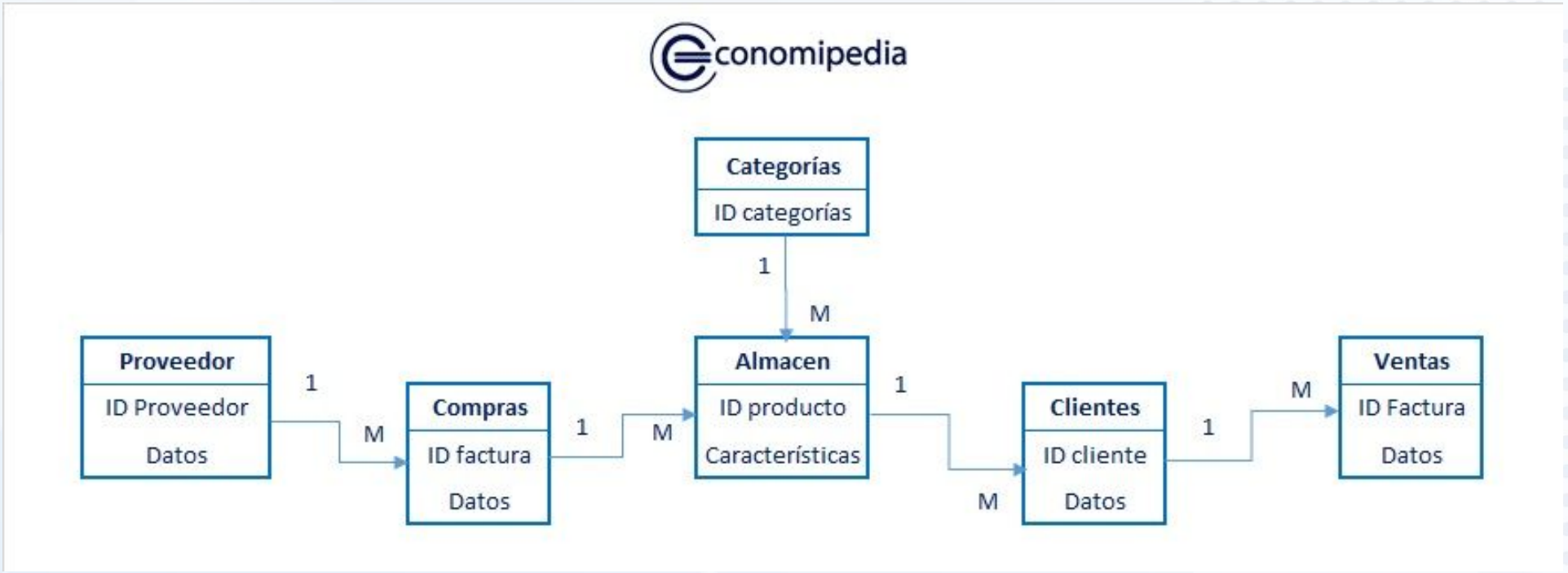
Es una base de datos no relacional orientada a documentos que ofrece una gran escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado.

- Guarda estructuras de datos en documentos tipo JSON (JavaScript Object Notation) con un esquema dinámico.
- Internamente MongoDB almacena los datos en formato BSON(Codificación binaria de documentos)
- BSON está diseñado para tener un almacenamiento y velocidad más eficiente.

mongoDB®

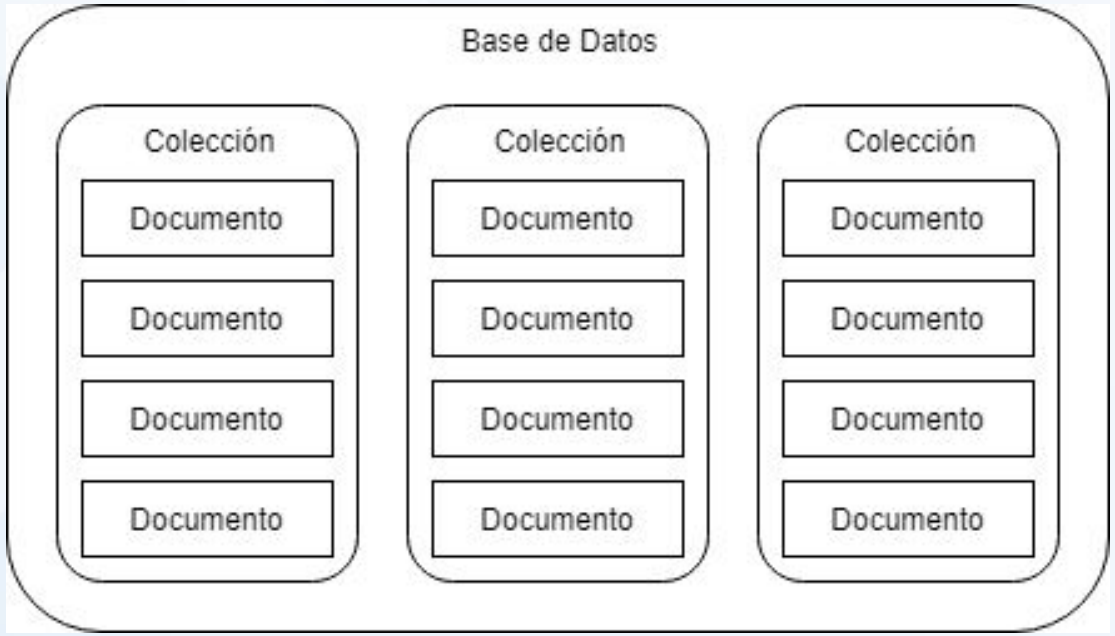
Modelado de datos

- Bd relacional: Diagrama relacional



Modelado de datos

- Bd no relacional: Colecciones




```
{
  "nombre": "SOLEPCC",
  "valor1": 1351824120,
  "sensors": [
    {
      "name": "Sensor Humedad",
      "description": "Mide Humedad",
      "sensor-type": "Humedad"
    },
    {
      "name": "Sensor temperatura",
      "description": "Mide Temperatura",
      "sensor-type": "Temperatura"
    }
  ]
}
```

Dependencias

Todos aquellos paquetes o librerías que necesitamos para construir la aplicación.

NPM

Gestor de paquetes por defecto de Nodejs - node --version && npm --version

Inicializar proyecto Node js

Comando: npm init

package.json

Archivo que contiene la información acerca del proyecto tal como descripción, licencia, dependencias y scripts.

Instalar:

- express: npm install express --save o npm install -S express
- typescript: npm install typescript
- npm install @types/express --save-dev o npm install -D @types/express

tsconfig.json

TypeScript utiliza un archivo llamado tsconfig.json para configurar las opciones del compilador para un proyecto.

```
{
  "compilerOptions": {
    "module": "commonjs",
    "esModuleInterop": true,
    "target": "es6",
    "moduleResolution": "node",
    "sourceMap": true,
    "outDir": "dist"
  },
  "lib": ["es2015"]
}
```

Actualizar package.json

```
"scripts": {
    "build": "tsc",
    "start": "npm run build && node dist/app.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  }
```

Crear el archivo app.ts

```
import express from 'express';

const app = express();
const port = 3000;
app.get('/', (req, res) => {
 res.send('Prueba del servidor');
});

app.listen(port, () => {
 return console.log(`server is listening on ${port}`);
});
```