

Auxiliar 8

AJAX y Promesas



START



Asíncrono



Usualmente pensamos en los programas como una secuencia de instrucciones.

Esto puede ser lento si alguna instrucción es muy lenta.

En varios lenguajes existe el concepto de paralelismo o multitasking donde instrucciones se ejecutan de manera asíncrona

Usualmente se usa una keyword **async** para decir que un funcion es asíncrona.



Single Page Applications



La aplicación vive en el navegador del usuario.

En las aplicaciones modernas raramente se carga una página web muy seguida

A esto se le llama **Single Page Application** (SPA)

Los frameworks de Backend y Frontend tienden a favorecer este tipo de aplicaciones

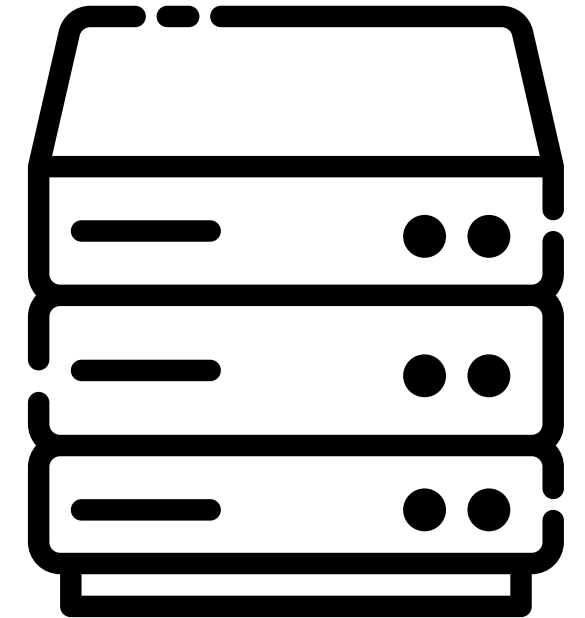
Generalmente usan una arquitectura con APIs que el usuario consulta de manera asíncrona.





GET /index

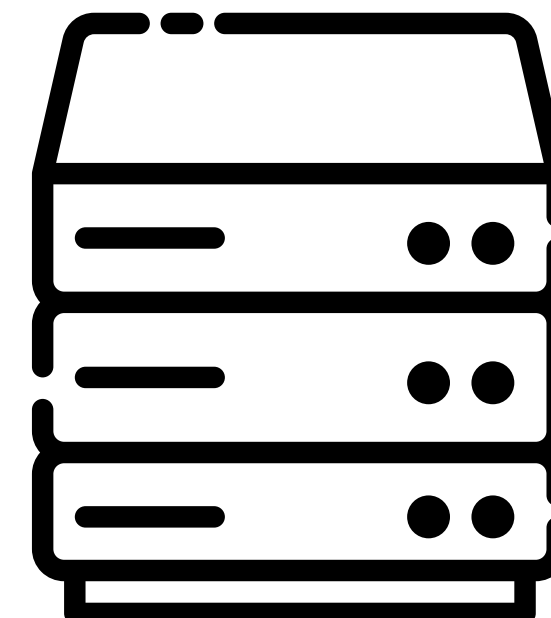
Servidor





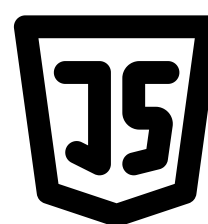
GET /index

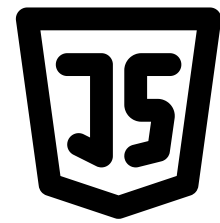
Servidor



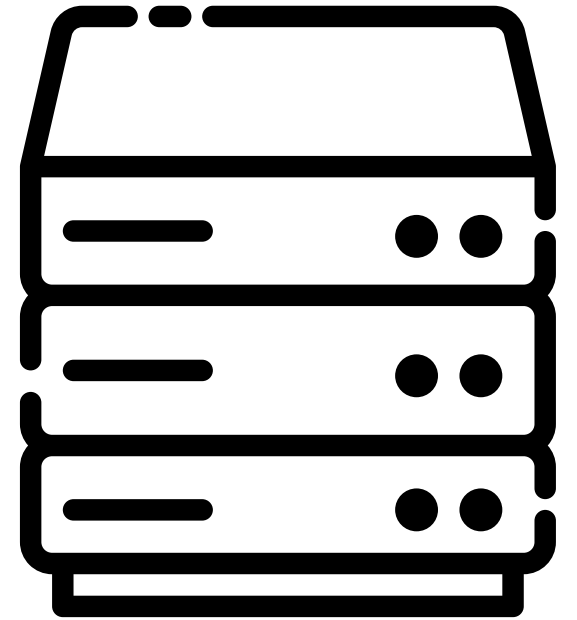
Response (Respuesta)

Status (200) Cabeceras



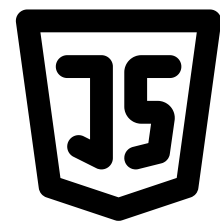
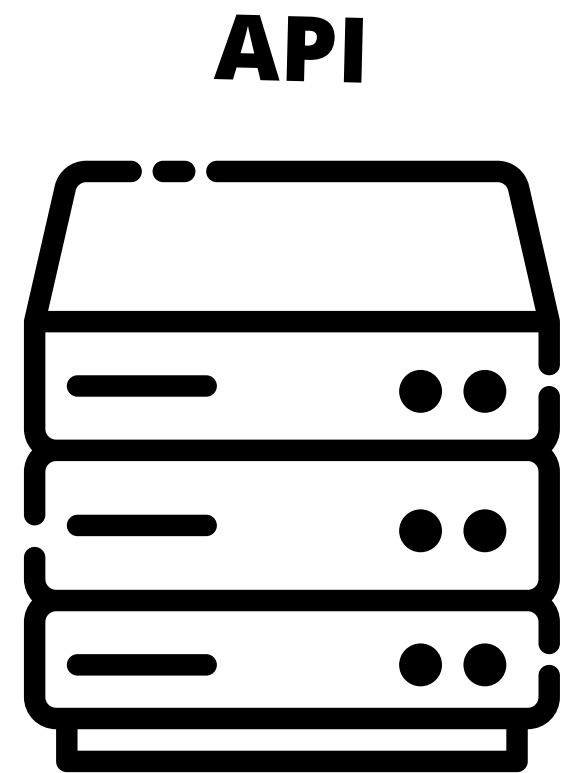


API





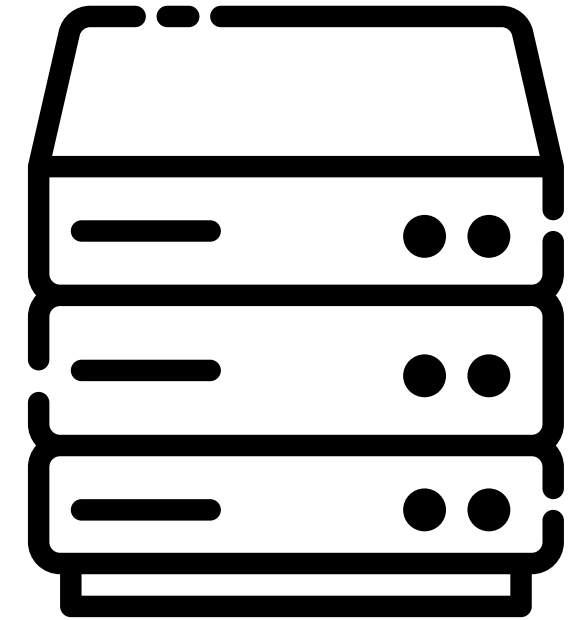
GET /mostacho




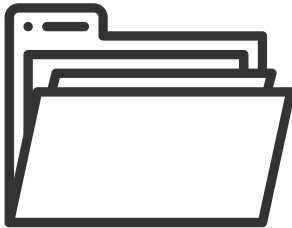
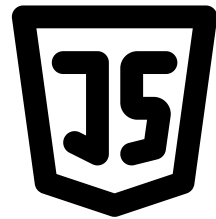


GET /mostacho

API



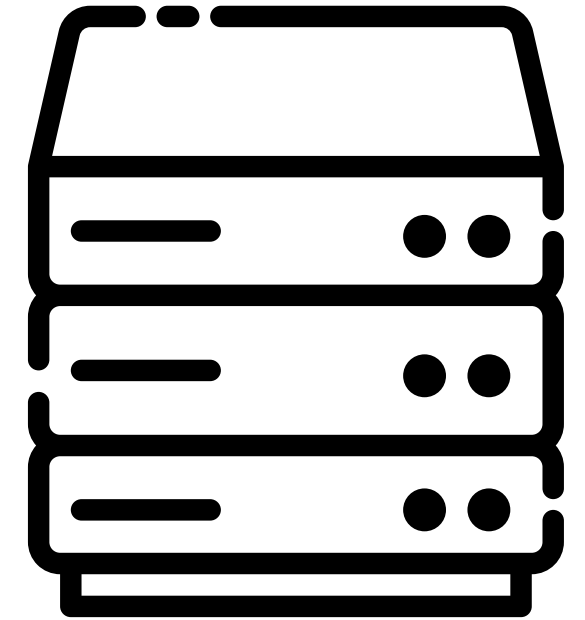
**"status": "ok",
"response": **



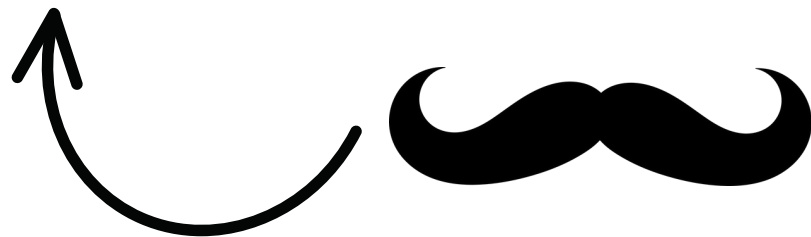
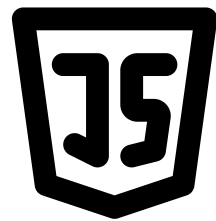


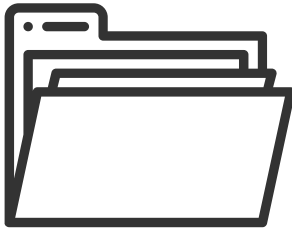
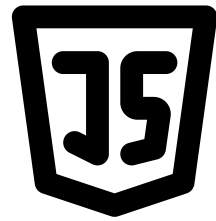
GET /mostacho

API

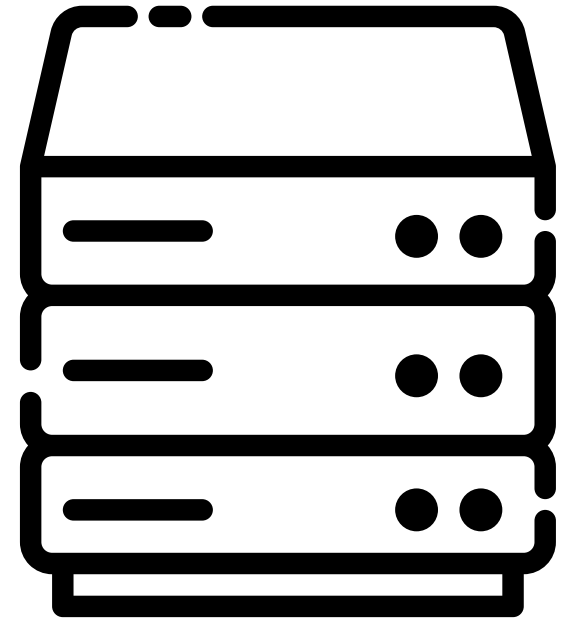


**"status": "ok",
"response":**





API



Promesas



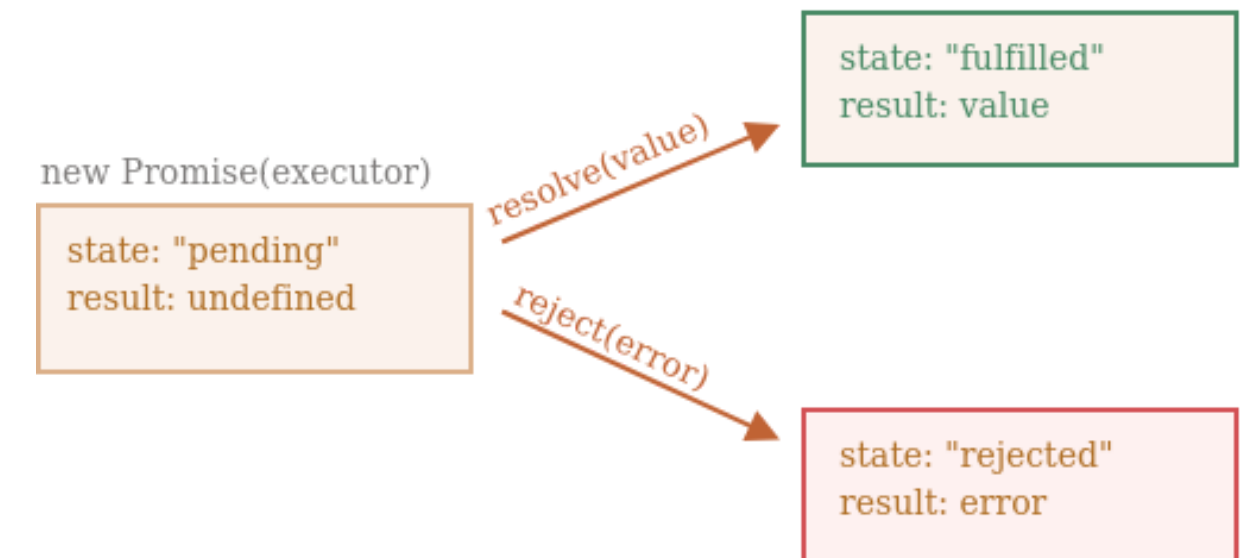
Javascript usa un ciclo de eventos para ejecutar el código.

Para ejecutar cosas asíncronas se "promete" que se va a correr.

Usualmente se usa para tareas pesadas (como pedir información de un servidor)

Como son una promesa hay que decir qué pasa cuando la promesa se resuelve o falla. -> **then** y **catch**

Hay otra forma de usar funciones con promesas con las palabras **async** y **await**. (**Referencia**)



Fetch

Se usa para buscar recursos de forma asíncrona.

Se basa en promesas y es nativo de Javascript

Hay librerías que se utilizan para hacer este tipo de llamadas. Ej: **Axios**

Necesita la URL y pueden pasarle opciones extras.

