

Criterio	REST con Polling	WebSockets (WS)	Server-Sent Events (SSE)
<b>**Complejidad Cliente**</b>	1/5: Implementación básica con fetch y setInterval.	4/5: Requiere manejar handshakes, estados de socket y protocolos propios.	2/5: Usa la API nativa EventSource, muy similar al manejo de eventos estándar.
<b>**Recursos (1k users)**</b>	Muy Alto: 1000 peticiones completas con sus headers cada 5 segundos (gran overhead).	Medio/Alto: Mantiene 1000 conexiones TCP abiertas permanentemente.	Medio: Conexión abierta unidireccional, más ligera que WS para solo lectura.
<b>**Latencia**</b>	Alta: El retraso promedio es de la mitad del intervalo (aprox. 2.5s).	Mínima: Actualización instantánea tipo "push".	Mínima: Actualización instantánea tipo "push".
<b>**Corte de Conexión**</b>	Resiliente: El siguiente ciclo de polling simplemente reintenta solo.	Frágil: Requiere lógica manual compleja para detectar cierres y reconectar.	Automático: El navegador gestiona la reconexión de forma nativa.
<b>**Mock Server**</b>	Muy Fácil: Compatible con cualquier herramienta de mock RESTful.	Difícil: Requiere servidores específicos que soporten el protocolo ws://.	Medio: Requiere configurar el header Content-Type: text/event-stream en el mock.
<b>**Mi evaluacion**</b>	Mala opcion, mucha latencia, pesima experiencia de usuario, lo unico bueno es que se hace rapido	Buena pero ta dificil de hacer, tiene mucha documentacion y hay mil y un herramientas,	Excelente opcion, instantaneo y ligero, escala bien pero tiene menos documentacion y herramientas que WS