# CHALLENGE MERCADO LIBRE

# Proyecto: ml-clone

Autor: Lautaro Amaya Muhlegg

*Desarrollo:*

## 1. Decisiones de Diseño

### Frontend

HTML + CSS + JS con Bootstrap y Live Server  
• Rápido de implementar: No necesito configurar entornos pesados (como React, Vue o Angular).  
• Control total del DOM: Como estaba buscando una réplica visual exacta de la página, JavaScript plano y CSS te dan control pixel-perfect sin overhead.  
• Compatible en cualquier sistema operativo: Live Server permite hacer pruebas en red (navegadores) sin configuración.  
• Bootstrap 5: Te da responsividad inmediata, grillas fluidas, y componentes similares a ML (cards, botones, alerts) sin reinventar la rueda.

### Backend

Node.js + Express + JSON  
• Simplicidad: Usar archivos .json en vez de una base de datos real es ideal para prototipos. No necesitás instalar nada adicional ni configurar servidores.  
• API RESTful real: Aunque no tenía base de datos, puede simular una API como lo haría un backend productivo.  
• Express es ligero y extensible: Podría fácilmente escalarlo si, mañana necesitara agregar rutas para productos relacionados, usuarios, o autenticación.

### Dev Experience

Scripts, Logs y Portabilidad  
• Scripts .bat y .sh: Hacer doble click y levantar todo es ideal si lo vas a presentar o entregar a un equipo técnico o evaluador.  
• Logging unificado: Es una práctica profesional guardar logs con fecha y hora, y centralizarlos por entorno.  
• Multiplataforma: Esta listo para Windows, Linux o macOS.

### ¿Y si mañana creciera?

El stack actual es escalable:  
• Podría migrar de JSON a MongoDB, SQLite o PostgreSQL fácilmente.  
• Podría migrar de HTML/JS plano a React o Next.js y seguir usando la misma API.  
• Puedo deployar en Vercel + Render o cualquier VPS sin muchos cambios.

## 2. Desafíos enfrentados y soluciones

• Efecto lupa sobre la imagen del producto: Requirió cálculos precisos de coordenadas, escalado y manejo de eventos de mouse. Se resolvió implementando una función `addZoom` personalizada con `mousemove`, `mouseover` y `mouseout`, imitando el comportamiento de MercadoLibre.

• Persistencia sin base de datos: El reto era mantener la funcionalidad completa (opiniones, preguntas, respuestas) usando solo archivos JSON locales. Se resolvió utilizando `fs.readFileSync` y `fs.writeFileSync` en cada ruta del backend, asegurando consistencia sin necesidad de un motor de base de datos.

• Inserción dinámica de contenido: Para evitar errores al mostrar preguntas, opiniones y detalles del producto desde la API, se usaron funciones como `renderProduct`, `renderQuestions`, `renderReviews`, garantizando estructura limpia y actualizable desde JSON.

• Modularidad sin frameworks: Al no usar React ni Vue, fue clave mantener claridad en el código. Se resolvió segmentando el HTML en secciones (subcards), el CSS en bloques temáticos, y el JS en funciones específicas.

## 3. Opinion personal del challenge

Agradezco mucho la oportunidad de afrontar este challenge ya que para mi resultó muy divertido y enriquecedor ya que no trabajaba con html y javascript hacía años por lo que fue un desafío volverme a reencontrar con estas tecnologías.

Además hice participar a mi familia (van a visualizar opiniones de ellos en la misma página) por lo cual pude hacer que nos divirtamos todos no solamente desarrollandolo yo sino pasando el tiempo con ellos y ellos dejando su granito de arena en este challenge.

Ante todo muchas gracias!