UNIVERSIDAD NEBRIJA

ALUMNO:

EPS

Asignatura: Arquitectura y Programación de sistemas en Internet			
Curso: 2025/2026	Examen:	Parcial	Fecha: 21-10-2025
Semestre: Primer		Convocatoria: Ordinaria	

API y Cliente LaserDisc (TypeScript + Express + Axios)

Objetivo

Crear una pequeña **API REST** con **Express** en **TypeScript** que gestione una colección de discos LaserDisc y un cliente que la use mediante **Axios**, todo dentro de **un único archivo index.ts**.

Parte 1: API con Express

Requisitos generales

- El servidor debe ejecutarse en el puerto **3000**.
- Los equipos estarán guardados en un **array en memoria** (no se usa base de datos).
- Cada disco debe seguir este tipo:

```
type LD = {

    id: number

    filmName: string

    rotationType: “CAV” | “CLV”,

    region: string,

    lengthMinutes: number,

    videoFormat: “NTSC” | “PAL”

}
```

UNIVERSIDAD NEBRIJA

Pasos a seguir

1.Importar Express y configurar el servidor:

2.Crear un array inicial de discos (simulado):

3.Implementar los siguientes endpoints:

Método	Ruta	Descripción	Qué debe hacer
GET	/ld	Mostrar todos los discos	Devolver el array completo en formato JSON.
GET	/ld/:id	Mostrar un disco por su ID	Buscar por ID. Si no existe, devolver 404 con { message: "Equipo no encontrado" }.
POST	/ld	Guardar un nuevo disco	Recibir todos los datos del tipo en el body. Generar un id nuevo y añadirlo al array. Devolver el nuevo equipo.
DELETE	/ld/:id	Eliminar un disco	Buscar el disco por ID y eliminarlo del array. Si no existe, devolver error 404.

No es necesario crear rutas PUT ni validaciones complejas.

El id se puede generar simplemente con Date.now() o sumando 1 al último.

4.**Arrancar el servidor:**

app.listen(3000, () => console.log("Servidor en http://localhost:3000"));

UNIVERSIDAD NEBRIJA

Parte 2: Cliente con Axios

Requisitos

Crea una función testApi() en el mismo archivo que:

1. Espere 1 segundo después de arrancar el servidor (para asegurar que esté listo).
2. Use **Axios** para llamar a la API e imprimir los resultados por consola.

Pasos concretos

1. **Obtener todos los discos (GET /teams).**
2. Muestra la lista inicial en consola.
3. **Crear un nuevo disco (POST /teams).**
4. **Volver a obtener todos los equipos (GET /teams).**
5. Comprueba que aparece el nuevo equipo.
6. **Eliminar ese equipo (DELETE /teams/:id).**
7. **Mostrar la lista final.**

Criterios de evaluación (10 puntos)

Aspecto	Puntos
Servidor Express funcionando correctamente	3
Endpoints bien implementados (GET, POST, DELETE)	3
Cliente Axios que hace las llamadas en orden correcto	3
Tipado y claridad del código	1

Entrega

- Todo debe estar en **un solo archivo index.ts**.
- Script de ejecución con ts-node y nodemon funcional.
- No se debe usar ninguna carpeta ni módulos adicionales.
- Se entregará el enlace del repo directamente clonable con git. El mal uso de la herramienta como commits posteriores al examen se traducirá en un 0.
- Durante todo el examen se deberá grabar pantalla en todo momento.
- El uso de cualquier herramienta generativa está absolutamente prohibido.
- Se pueden usar apuntes e información en internet siempre que no se rompa la norma anterior.
- El incumplimiento de cualquiera de las anteriores normas dará automáticamente a un cero.