# **PROYECTO: API REST IMDB**

El objetivo de este proyecto es implementar una API REST totalmente funcional.

### **API REST de un conjunto de profesionales.**

Tomando como base el API REST desarrollada en el reto, desarrollar una nueva API REST con los siguientes endpoints:

* GET /profesionales?id=5. Obtiene los datos del profesional con el id pasado por parámetro.
* GET /profesionales. Obtiene toda la lista de profesionales.
* POST /profesionales. Añade un nuevo profesional en la lista de profesionales.
* PUT /profesionales. Modifica los datos de un profesional
* DEL /profesionales. Elimina a un profesional de la lista.

### **API REST de una película (Sin Profesional).**

Teniendo en cuenta la definición de película de los ejercicios anteriores pero sin incluir las propiedades de tipo Profesional, hacer un API REST de una película con los siguientes endpoints.

* GET /pelicula. Obtiene los datos de una película
* POST /pelicula. Crea una nueva película
* PUT /pelicula. Actualiza los datos de una película.
* DEL /película. Elimina la película

### **API REST de una película (Con Profesional)**

Modificar el API anterior para incluir un array de profesionales (actores) y el profesional director y guionista. Esta nueva API tendrá los mismos endpoint que la anterior pero mostrará también la información relativa a los profesionales.

Además incluirá nuevos endpoints.

* GET /pelicula/actor?id=8. Devuelve los datos relativos al actor con ese id. SOLO FUNCIONA SI CREO LA PELICULA DESDE CERO
* GET /pelicula/director. Devuelve los datos relativos al director de la pelicula.
* GET /pelicula/guionista. Devuelve los datos relativos al guionista de la pelicula.
* POST /pelicula/actor. Añade un nuevo actor a la lista de actores de la película.
* POST /pelicula/director. Crea un director de la película
* POST /pelicula/guionista. Crea un guionista de la película
* PUT /pelicula/actor. Modifica los datos de un actor.
* PUT /pelicula/director. Modifica los datos del director.
* PUT /pelicula/guionista. Modifica los datos del guionista.
* DEL /pelicula/actor. Eliminar al actor de la lista de actores.
* DEL /pelicula/director. Elimina al director
* DEL /pelicula/guionista. Elimina al guionista de la pelicula

### **API REST de un conjunto de películas**

Tomando como base el ejercicio anterior hacer una nueva API REST con los siguientes endpoints:

* GET /peliculas?id = 2. Devuelve la pelicula con el id pasado por parámetro.
* GET /peliculas. Devuelva todas las películas almacenadas
* POST /peliculas. Añade una nueva pelicula en el array de películas
* PUT /peliculas. Actualiza una pelicula.
* DEL /peliculas. Elimina una pelicula de la lista de películas.

### **API REST convertir los recursos en ficheros.**

Modificar todos los ejercicios anteriores para, en lugar de guardar en objetos el recurso se guarden en ficheros JSON.

Por ejemplo, para el primer ejercicio, si teníais un objeto llamado profesionales donde guardabais el array de profesionales, ahora ese array se tendrá que obtener de un fichero profesionales.json.

### **API REST dividir cada ejercicio en un modulo diferente**

Modificar cada ejercicio para convertirlo en un modulo diferente. Así tendréis 4 ficheros distintos:

* apiRestprofesionales.js
* apiRestpelicula.js
* apiRestpeliculas.js
* index.js

El fichero index.js tendrá que importar cada uno de los ficheros de apiRest y probarlos.

Para acceder al id que te envia el cliente en un GET hay dos maneras  
Si el cliente te envia la siguiente consulta: /profesionales?id=1  
app.get("/profesionales",  
        function(request, response)  
        {  
            let id = request.query.id;  
        }  
        );  
Si el cliente te envia la siguiente consulta: /profesionales/1  
app.get("/profesionales/:id",  
        function(request, response)  
        {  
            let id= request.params.id;  
        }  
        );