TEMA 3 EXPRESS.JS

Parte II – Express aplicado a servicios REST

II.3 Ejemplo Práctica de rutas en Módulos

Ejemplo práctico a Realizar

- Vamos a realizar un ejemplo algo más complejo contra la base de datos de libros en MongoDB que hemos utilizado en temas anteriores.
- En este proyecto dividiremos el tratamiento de servicios en diferentes enrutadores, cada uno en su archivo, conectando con una base de datos MongoDB, y enlazándolo todo desde el servidor principal.
- Crea para empear una carpeta llamada "Tema3_LibrosExpress" en tu carpeta de ejercicios.
- Ábrela con Visual Studio Code e instala en ella los módulos *express*, *mongoose* y *body-parser* que hemos utilizado en el ejemplo previo.

Estructura de carpetas

- app.js: Servidor principal
- Carpeta models (con dos modelos):
  - autor.js
  - libro.js
- Carpeta routes (con los enrutadores):
  - autor.js
  - libro.js

Definición del modelo

- Añadiremos en "models/autor.js" el modelo de autores (y lo exportaremos) y en "models/libro.js" el modelo de libros (también exportado).
- ***Ejercicio 1***
- **Rellena el código de los archivos "models/autor.js" y "models/libro.js" con los modelos de autor y libro que utilizamos en el tema anterior.**

Estructura básica del Servidor

- Incorpora con require los tres módulos instalados (*mongoose, express* y *body-parser*)
- Inicializa la aplicación y añádele el *middleware* de *body-parser.*
- Derivar las rutas de los servicios a los correspondientes enrutadores, por lo que vamos a incluir los módulos respectivos y añadirlos tras el *middleware* anterior.
- Abre el fichero app.js y realiza todo lo indicado.

APP.JS

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');


const autores = require('./routes/autor');
const libros = require('./routes/libro');


mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost:27017/libros',
{useMongoClient: true});


let app = express();
app.use(bodyParser.json());
app.use('/autores', autores);
app.use('/libros', libros);


app.listen(8080);
```

Definir los enrutadores

- Vamos a hacer paso a paso un enrutador para autores que dé servicio a dos operaciones básicas: listar los autores y añadir un nuevo autor.
- En el archivo "routes/autor.js" añadimos el siguiente código para definir el enrutador, e incorporar el modelo de autor, que necesitaremos en el código después:

```
const express = require('express');
let Autor = require('../models/autor.js');
let router = express.Router();
module.exports = router;
//Después, añadimos el servicio para listar todos los autores, y
//el servicio para insertarlos (normalmente, esto va antes de
//exportar el módulo):
router.get('/', (req, res) => {
...
});

router.post('/', (req, res) => {
...
});

module.exports = router;
```

EJERCICIO 2

- Completa el código de los dos servicios anteriores para que den el resultado previsto (listado de todos los autores e inserción de un nuevo autor a la colección correspondiente)

EJERCICIO 3

- Haz algo similar a lo anterior en el enrutador para libros "routes/libro.js", definiendo servicios para:
- Listar todos los libros
- Listar un libro a partir de su *id*
- Insertar un libro

Pinceladas finales

- Para finalizar con este ejercicio, vamos a añadir un par de funciones *middleware* propias, para ver cómo crear este tipo de funciones y su utilidad en el procesamiento de peticiones de los clientes.

EJERCICIO 4

- Define un middleware en la aplicación principal "app.js", justo antes de incluir el middleware de *body-parser*, para mostrar por consola la fecha, método y URI solicitada para cada petición.

EJERCICIO 5

- Añade ahora otro middleware, justo antes del anterior (es decir, el primero de todos), que envíe por JSON un mensaje de "En mantenimiento" y no permita ofrecer ningún servicio (es decir, que no llame al método next)