Proyecto 3ª evaluación

26/04/2020

—

Miguel Ferreira Cordero
Tu empresa
1º ASIR DAM Subgrupo B

Tecnologías usadas

Repl.it es un entorno de desarrollo online que permite escribir programa en una multitud de lenguajes diferentes.

El lenguaje que usaremos para programar es Node.js y TypeScript.

Node.js es un lenguaje de programación basado en JavaScript, con una entrada y salida de datos orientada a eventos.

TypeScript también está basado en JavaScript. Este lenguaje esencialmente añade tipos estáticos y objetos basados en clases. Este es usado para desarrollar aplicaciones JavaScript, y en este caso se ejecutará en extensiones para programas, como es Node.js.

Documentación: https://www.typescriptlang.org/docs/handbook/intro.html

Express es un módulo que ayudará a nuestra aplicación con respecto a la ejecución del proyecto.

Documentación: https://expressjs.com/es/guide/routing.html

Mongoose es un módulo el cual nos servirá para permitir a nuestra aplicación con MongoDB.

Documentación: https://mongoosejs.com/docs/guide.html

Estructura

Este proyecto está compuesto de 3 carpetas, dist, doc y src.

- **dist**: Aquí se encuentran los archivos .js transpilados.
- **doc**: En este directorio estarán los 3 PDFs explicando el proyecto, además todo aquello correspondiente a Mongo Atlas y Node.js junto a TypeScript.
- **src**: Aquí habrán 3 directorios más. Estos son database, datos y model:
    - **database**: En este se encuentra toda la configuración para conectar con la base de datos de Mongo Atlas.
    - **datos**: Este contiene un archivo .js llamado insert.js donde se encuentran todas las colecciones de la base de datos listas para ser insertadas.
    - **model**: Este directorio almacena los schemas y las interfaces.

  Por último, en esta carpeta está el archivo index.ts, donde está todo el código del programa, rutas, etc.

Origen de la base de datos

En este proyecto se ha creado una base datos sobre coches, en concreto de sus características más comunes y útiles. En otras colecciones a parte de la de coches, se almacenará también las categorías de coches en la base de datos, las marcas de estos y las competiciones que han ganado las marcas.

Esto servirá para un museo de coches modernos, el cual precisa recoger todos los datos anteriores

De los coches se almacenará el id del mismo, el id de la marca que lo fabricó, el o los ids de las categoría a la que pertenece, el nombre del modelo, su precio, la fecha en la que salió al mercado, las especificaciones, siendo esta un documento que recoge los cilindros que tiene el coche, la cantidad de asientos y de puertas, el caballaje y el color de la carrocería. Además se anotará el tipo de motor, si es térmico, híbrido o eléctrico.

De las categorías de estos se guardará el id de esta y su nombre.

Sobre las marcas se recogerá el id, el nombre, la fecha y el país en el que se fundaron, los ids de las competiciones en las que ha participado o sigue participando en la actualidad, y el nombre y número de competiciones que ha ganado.

Finalmente, sobre las competiciones se registrará su id y el nombre.

Model

En esta carpeta se encuentran los siguientes schemas:

Categorías:

src/model/categorias.ts

```ts
import {Schema, model} from 'mongoose'

const categoriaSchema = new Schema({
    id_categoria: Number,
    nombre_categoria: String,
})

export interface Categoria {
        id: number;
        name: string;
        salary: number;
}

export const Categorias = model('categorias', categoriaSchema)
```

Competiciones:

src/model/competiciones.ts

```ts
import {Schema, model } from 'mongoose'

const competicionSchema = new Schema({
    id_competicion: Number,
    nombre_competicion: String,
})


export interface Competicion {
    id_competicion: number,
    nombre_competicion: string,
}

export const Competiciones = model('competiciones', competicionSchema)
```

Coches // Marcas

src/model/coches.ts

```ts
import {Schema, model } from 'mongoose'
// Definimos el Schema
const cocheSchema = new Schema({
    id_coche: Number,
    id_marca: Number,
    id_categoria: [Number],
    nombre_coche: String,
    precio: Number,
    fecha_estreno: Date,
    especificaciones: {
      Cilindros: Number, HP: Number , Asientos:
      Number , Puertas: Number , Color: String
    },
    tipo_motor: {
      coche_electrico: Boolean, coche_hibrido: Boolean
    },
})


export interface Coche {
    id_coche: number,
    id_marca: number,
    id_categoria: [number],
    nombre_coche: string,
    precio: number,
    fecha_estreno: Date,
    especificaciones: {
      cilindros: number, hp: number , asientos:
      number , puertas: number , color: string
      },
    tipo_motor: {
      coche_electrico: boolean, coche_hibrido: boolean
      },
}

export const Coches = model('coches', cocheSchema)
```

src/model/marcas.ts

```ts
import {Schema, model } from 'mongoose'

const marcaSchema = new Schema({
    id_marca: Number,
    nombre_marca: String,
    fecha_fundacion: Date,
    pais: String,
    id_competicion: [Number],
    competiciones_ganadas: {
      "24 Horas de Le Mans": Number,
      "Formula E": Number,
      "WTCR": Number,
      "WRC": Number,
      "F1": Number
    },
})


export interface Marca {
    id_marca: number,
    nombre_marca: string,
    fecha_fundacion: Date,
    pais: string,
    id_competicion: [number],
    competiciones_ganadas: {
      "24 Horas de Le Mans": number,
      "Formula E": number,
      "WTCR": number,
      "WRC": number,
      "F1": number
    },
}

export const Marcas = model('marcas', marcaSchema)
```

Interfaces

src/model/interfaces.ts

```ts
export interface PrecioMarca {
  id_marca: number,
  nombre_equipo: string,
  valorTotal: number
}
```

Rutas

• /datos = Muestre los datos de todos los coches registrados en la base de datos.

• /coches/nombre/:nombre = Muestre los datos de un coche en específico según su nombre. Hay que reemplazar el ":nombre" por el modelo deseado.

• /competiciones = Muestra las marcas y las competiciones en las que estas han participado.

• /electricos = Muestra los coches eléctricos que cumplan ciertas características.

• /datosCoches = Haga una consulta los coches que cumplan varios requisitos, muestre si estos tienen derecho a subvención y la etiqueta medioambiental de la DGT según sus características.

Funciones

En este proyecto hay 5 funciones, son las siguientes:

Inicio: Breve descripción del proyecto.

```
const inicio = async (req: Request, res: Response) => {
    res.send("BASE DE DATOS DE COCHES - MIGUEL FERREIRA.")
}
```

Función 1: Hace un find a la colección coches, pero solo me muestre los datos de un modelo en específico que ha sido introducido por parámetros.

```
const fun1 = async (req: Request, res: Response) => {
  const nom = req.params.nombre
  await db.conectarBD()
  .then(
    async (mensaje) => {
      const query: any = await Coches.findOne({ "nombre_coche": { $eq: nom } })
      console.log(query)

      res.send(query)
    })
  .catch(
      (mensaje) => {
        res.send(mensaje)
        console.log(mensaje)
    })
  db.desconectarBD()
}
```

Función 2: Todas las marcas de la base de datos de este proyecto tienen al menos 1 coche registrado. Cada uno de ellos tiene registrado su precio. Con la suma de todos los coches estos tendríamos el valor de esa marca en nuestra base de datos. Sabiendo esto, lo que esta función tiene como objetivo es mostrar por pantalla el valor total de una marca, la cual especificaremos por parámetros en la barra de búsqueda. Lo primero que hace esta función es realizar un find a la colección Coches. Dicho resultado se guardará en un array y en una variable irán sumándose los precios de todos los coches debido a que se estará recorriendo el array con un for.

```
const fun2 = async (req: Request, res: Response) => {
    await db.conectarBD()
    .then(
        async (mensaje) => {
          let arrayCoches: Array<Coche>
          const query: any = await Coches.find(
          {}, {_id: 0, id_coche: 1, id_marca: 1, id_categoria: 1, nombre_coche: 1, precio: 1})

          console.log(query)
          arrayCoches = query
          console.log(arrayCoches)

          let valorTotal: number = 0
          let coche: Coche

          for (coche of arrayCoches){
            console.log(coche.precio)
            valorTotal += coche.precio
          }

          res.json("Valor total de la suma del coste de todos los coches: " + valorTotal+"€." )
    })
    .catch(
      (mensaje) => {
        res.send(mensaje)
        console.log(mensaje)
    })
    db.desconectarBD()
  }
```

Función 3: El objetivo es mostrar el o los coches que cumplan las condiciones especificadas por parámetros. En este caso se introduce la marca de un coche y su categoría. Así, se buscará la marca en la base de datos, y una vez encontrado se mostrará su id, el nombre del coche, el id y el nombre de su categoría, su precio, y la fecha en la que se estrenó.

```
const fun3 = async (req: Request, res: Response) => {
  const categoriaCoche : string = req.params.categoria
  const marcaCoche : string = req.params.marca
    await db.conectarBD()
    .then(
        async (mensaje) => {
          console.log(mensaje)
          const query: any = await Coches.aggregate([
            {
              $lookup:{
                from: "categorias",
                localField: "id_categoria",
                foreignField: "id_categoria",
                as: "categoria"
              }
            },
            {
              $lookup:{
                from: "marcas",
                localField: "id_marca",
                foreignField: "id_marca",
                as: "marca"
              }
            },
            {
              $match:{
                    $and:[
                  {
                    "categoria.nombre_categoria": categoriaCoche,
                  },
                  {
                    "marca.nombre_marca": marcaCoche
                  }
                ]
              }
            },
            {
              $project:{
                id_coche: "$id_coche",
                nombre_coche: "$nombre_coche",
                id_categoria: "$categoria.id_categoria",
                nombre_categoria: "$categoria.nombre_categoria",
                precio: "$precio",
                fecha_estreno: "$fecha_estreno",
              }
            },
            {
              $project:{
                _id:0,
                id_coche: 1,
                nombre_coche: 1,
                id_categoria: 1,
                nombre_categoria: 1,
                precio: 1,
                fecha_estreno: 1
              }
            }
          ])

          console.log(query)
          res.json(query)
    })
    .catch(
      (mensaje) => {
        res.send(mensaje)
        console.log(mensaje)
    })
    db.desconectarBD()
}
```

Función 4: El objetivo de esta función es contar el número de coches que tiene cada marca en la base de datos. Para ello, se hará un aggregate a Marcas, juntándole Coches con un lookup. Después se buscará la marca en específico con un match, la cual tendríamos que haber introducido por parámetros. Una vez hecho todo eso, se recorre la colección coches con un for, y si encaja la marca se suma con un contador.

```
const fun4 = async (req: Request, res: Response) => {
  let coche: Coche
  let resultado: Array<Coche>
  const marcaCoche = req.params.marcaCoche
  await db.conectarBD()
  .then(
    async (mensaje) => {
      let num_coches: number = 0
      console.log(mensaje)
      const query: any = await Marcas.aggregate([
    {
      $lookup:
      {
        from: "coches",
        localField: "id_marca",
        foreignField: "id_marca",
        as: "coche"
      }
    },
    {
      $unwind: "$coche"
    },
    {
      $match:{"nombre_marca": marcaCoche}
    }
  ])

  resultado = query
  console.log(query)

  for (coche of resultado){
    num_coches += 1
  }

  res.json(`La marca de coches ${marcaCoche} tiene un total de ${num_coches} coches en la base de datos.`)
  })

  .catch(
    (mensaje) => {
      res.send(mensaje)
      console.log(mensaje)
    })
  db.desconectarBD()
}
```

Función 5: Esta función tiene como objetivo calcular el valor de una marca, haciendo una suma de todos los coches de esta y así calcular su valor en la base de datos.

Se introduce la marca, o el id, y de esa forma se imprime por pantalla el resultado a través de un push.

```
const fun5 = async (req: Request, res: Response) => {
  const nombre_marca_entrada : string = req.params.nombre_marca_entrada
  const id_marca_entrada : number = parseInt(req.params.id_marca_entrada)
    await db.conectarBD()
    .then(
        async (mensaje) => {
          console.log(mensaje)
          let arrayCoches: Array<Coche>
          let arrayMarcas: Array<Marca>
          const query: any = await Coches.aggregate([
            {
              $lookup:{
                from: "marcas",
                localField: "id_marca",
                foreignField: "id_marca",
                as: "marca"
              }
            },
            {
              $project:{
                nombre_marca: "$marca.nombre_marca",
                precio: "$precio",
                id_marca: "$marca.id_marca"
              }
            },
            {
              $match:{
                $or:[
                {
                  "nombre_marca": nombre_marca_entrada
                },
                {
                  "id_marca": id_marca_entrada
                }
                ]
              }
            },
            {
              $project:{
                _id:0,
                id_coche:1,
                id_marca:1,
                id_categoria: 1,
                nombre_coche: 1,
                nombre_marca: 1,
                precio: 1
              }
            }
          ])
          console.log(query)
          arrayCoches = query
          arrayMarcas = query
          console.log(arrayCoches)

          let valorTotal: number = 0
          let nombre_de_la_marca: string = ""
          let coche: Coche
          let marca: Marca

          interface respuesta{
            nombre_marca: string,
            valor_marca:number
          }
          let resultado: Array<respuesta> = []

          for (coche of arrayCoches){
            console.log(coche.precio)
            valorTotal += coche.precio
          }

          for (marca of arrayMarcas){
            nombre_de_la_marca = marca.nombre_marca
          }

          console.log(`Valor total: ${valorTotal}`)
          resultado.push({
            nombre_marca: nombre_de_la_marca,
            valor_marca: valorTotal
          })
          res.json(resultado)
    })
    .catch(
      (mensaje) => {
        res.send(mensaje)
        console.log(mensaje)
    })
    db.desconectarBD()
}
```