Endpoints

- Find:
    - Artistas
        - artista/todos
            - lista todos los artistas
        - artista/nombre/:nombre
            - filtra por nombre
            - introducir string
        - artista/genero/:genero
            - filtra por género
            - introducir string
        - artista/activo/:activo
            - muestra el listado de los artistas activos o no activos
            - introducir booleano (true/false)
    - Álbumes
        - album/todos
            - lista todos los álbumes
        - album/titulo/:titulo
            - filtra por título
            - introducir string
        - album/artista/:nombre
            - filtra por artista
            - introducir string
        - album/genero/:genero
            - filtra por géneros
            - introducir objeto con strings “genero” en formato JSON
            - “genero” = { “genero”: “rock”, “genero: “salsa” }
    - Canciones
        - cancion/todas
            - lista todas las canciones
        - cancion/titulo/:titulo
            - filtra por título
            - introducir string
        - cancion/artista/:nombre:
            - filtra por nombre de artista
            - introducir string
        - cancion/album/:titulo
            - filtra por título del álbum
            - introducir string
        - cancion/duracion/:duracion
            - filtra por rangos de duración
            - introducir “corta” (menos de 3:10 minutos), “media” (entre 3:10 y 4:49) o “larga” (4:50 o más).
        - cancion/decada/:decada
            - filtra por décadas
            - introducir string con la década, entre 1960 y 2020. Ej: “1990”.
- Añadir entradas por POST
    - cancion/new --- artista/new --- album/new

    - introducir por JSON solo los campos que se quieran actualizar, los que no se introduzcan se mantendrán igual.
- Update
    - cancion/update/:titulo --- artista/update/:nombre --- album/update/:titulo
    - Introducir por GET el criterio de búsqueda (string) y los campos a actualizar por POST, igual que en la creación de nuevas entradas.
- Eliminar
    - Una sola entrada, por nombre o título
        - artista/del/:nombre --- artista/delc/:nombre
        - album/del/:titulo --- album/delc/:titulo
        - cancion/del/:titulo --- cancion/delc/:titulo
        - “del” pide una confirmación antes de llamar al endpoint “delc”, que elimina directamente.
        - Es posible acceder directamente al endpoint “delc”.
    - Varias entradas, por filtro
        - artista/del/activo/:activo
            - elimina todos los artistas que estén activos o no activos
            - introducir booleano para filtrar
        - album/del/genero/:genero
            - elimina todos los álbumes de un género específico
            - introducir string con género

Capturas

```
// #region ESQUEMAS
// (estructura de las colecciones)

const schArtista = new mongoose.Schema(
    {
        nombre: { type: String, unique: true },
        miembros: [{ nombre: String }],
        activo: { type: Boolean, default: false },
        genero: [{ genero: String }]
    }
);

const schAlbum = new mongoose.Schema(
    {
        titulo: { type: String, unique: true },
        artista: String,
        genero: [{ genero: String }]
    }
);

const schCancion = new mongoose.Schema(
    {
        titulo: { type: String, unique: true },
        artista: String,
        album: String,
        duracion_segundos: Number,
        fechaPublicacion: Date
    }
);
```

```
// #region MODELOS
// (colecciones en Mongo)

const artistas = mongoose.model('artistas', schArtista);
const albumes = mongoose.model('albumes', schAlbum);
const canciones = mongoose.model('canciones', schCancion);
```

```
57
58      // Añadir documentos a las colecciones
59
60      // #region artistas
61      const insertartistas = [
62          new artistas({
63              nombre: "led zeppelin",
64              miembros: [
65                  { nombre: "john bonham" },
66                  { nombre: "jimmy page" },
67                  { nombre: "john paul jones" },
68                  { nombre: "robert plant" }
69              ],
70              genero: [
71                  { genero: "hard" },
72                  { genero: "rock" },
73                  { genero: "blues" },
74                  { genero: "inglés" }
75              ]
76          }),
77          new artistas({
78              nombre: "extremoduro",
79              miembros: [
80                  { nombre: "roberto iniesta" },
81                  { nombre: "iñaki antón" }
82              ],
83              genero: [
84                  { genero: "rock" },
85                  { genero: "metal" },
86                  { genero: "español" }
```

```
// #region álbumnes
const insertalbumes = [
    new albumes({
        titulo: "iii",
        artista: "led zeppelin",
        genero: [
            { genero: "hard" },
            { genero: "rock" },
            { genero: "blues" },
            { genero: "inglés" }
        ]
    }),
    new albumes({
        titulo: "la ley innata",
        artista: "extremoduro",
        genero: [
            { genero: "rock" },
            { genero: "metal" },
            { genero: "español" }
        ]
    }),
    new albumes({
        titulo: "digan lo que digan",
        artista: "raphael",
        genero: [
            { genero: "balada" },
            { genero: "canción" },
            { genero: "español" }
```

```
286     // #region canciones
287     const insertcanciones = [
288         new canciones({
289             titulo: "tangerine",
290             artista: "led zeppelin",
291             album: "iii",
292             duracion_segundos: 193,
293             fechaPublicacion: '1970-09-18'
294         }),
295         new canciones({
296             titulo: "lo de fuera",
297             artista: "extremoduro",
298             album: "la ley innata",
299             duracion_segundos: 426,
300             fechaPublicacion: '2008-11-12'
301         }),
302         new canciones({
303             titulo: "mi gran noche",
304             artista: "raphael",
305             album: "digan lo que digan",
306             duracion_segundos: 182,
307             fechaPublicacion: '1967-04-09'
308         }),
309         new canciones({
310             titulo: "200 años",
311             artista: "invisible",
```

```
    // #region saves
    for (let i = 0; i < insertartistas.length; i++) {
        try {
            await insertartistas[i].save();
            console.log(insertartistas[i].nombre + ': artista añadido');
        } catch (E11000) {
            console.log(insertartistas[i].nombre + ': artista ya existente');
        }
    }

    for (let i = 0; i < insertalbumes.length; i++) {
        try {
            await insertalbumes[i].save();
            console.log(insertalbumes[i].titulo + ': album añadido');
        } catch (E11000) {
            console.log(insertalbumes[i].titulo + ': album ya existente');
        }
    }

    for (let i = 0; i < insertcanciones.length; i++) {
        try {
            await insertcanciones[i].save();
            console.log(insertcanciones[i].titulo + ': canción añadida');
        } catch (E11000) {
            console.log(insertcanciones[i].titulo + ': canción ya existente');
        }
    }

    console.log('\n---FIN INSERCIÓN DE DATOS---\n');
```

```
382     // #region FIND
383     // #region artistas
384
385     app.get('/artista/todos', async (req, res) => {
386         const artista = await artistas.find();
387         if (artista.length > 0) {
388             return res.json(artista);
389         } else {
390             return res.send("No hay registros.");
391         }
392     })
393
394     app.get('/artista/nombre/:nombre', async (req, res) => {
395         const artista = await artistas.findOne({ nombre: req.params.nombre });
396         if (artista) {
397             return res.json(artista);
398         } else {
399             return res.send(`Artista ${req.params.nombre} no encontrado.`);
400         }
401     })
402
403     app.get('/artista/genero/:genero', async (req, res) => {
404         try {
405             artista = await artistas.find({ 'genero.genero': req.params.genero });
406             if (artista.length > 0) {
407                 return res.json(artista);
408             } else {
409                 return res.send(`Género ${req.params.genero} no encontrado.`);
410             }
411         } catch (ERROR) {
412             return res.send(`Género ${req.params.genero} no encontrado.`);
413         }
```

```
428     // #region álbumes
429
430     app.get('/album/todos', async (req, res) => {
431         const album = await albumes.find();
432         if (album.length > 0) {
433             return res.json(album);
434         } else {
435             return res.send("No hay registros.");
436         }
437     })
438
439     app.get('/album/titulo/:titulo', async (req, res) => {
440         try {
441             album = await albumes.find({ titulo: req.params.titulo });
442             if (album.length > 0) {
443                 return res.json(album);
444             } else {
445                 return res.send("No se encontraron álbumes.");
446             }
447         } catch (ERROR) {
448             return res.send("No se encontraron álbumes.");
449         }
450     })
451
452     app.get('/album/artista/:artista', async (req, res) => {
453         try {
454             album = await albumes.find({ artista: req.params.artista });
455             if (album.length > 0) {
456                 return res.json(album);
457             } else {
458                 return res.send("No se encontraron álbumes.");
```

```
478     // #region canciones
479
480     app.get('/cancion/todas', async (req, res) => {
481         const cancion = await canciones.find();
482         if (cancion.length > 0) {
483             return res.json(cancion);
484         } else {
485             return res.send("No hay registros.");
486         }
487     })
488
489     app.get('/cancion/titulo/:titulo', async (req, res) => {
490         try {
491             cancion = await canciones.find({ titulo: req.params.titulo });
492             if (cancion.length > 0) {
493                 return res.json(cancion);
494             } else {
495                 return res.send("No se encontraron canciones.");
496             }
497         } catch (ERROR) {
498             return res.send("No se encontraron canciones.");
499         }
500     })
501
502     app.get('/cancion/artista/:artista', async (req, res) => {
503         try {
504             cancion = await canciones.find({ artista: req.params.artista });
505             if (cancion.length > 0) {
506                 return res.json(cancion);
507             } else {
508                 return res.send("No se encontraron canciones.");
```

```
Search (Ctrl+Shift+F) // #region añadir nuevo
597
598 app.post('/cancion/new', async (req, res) => {
599     try {
600         const titulo = req.body.titulo;
601         const artista = req.body.artista;
602         const album = req.body.album;
603         const duracion_segundos = req.body.duracion_segundos;
604         const fechaPublicacion = req.body.fechaPublicacion;
605
606         if (!titulo || !artista || !album || !duracion_segundos || !fechaPublicacion) {
607             return res.status(400).json({ error: 'Todos los campos son obligatorios: titulo, artista, album, duracion, fecha.' });
608         }
609
610         const newCancion = new canciones({
611             titulo: titulo,
612             artista: artista,
613             album: album,
614             duracion_segundos: duracion_segundos,
615             fechaPublicacion: fechaPublicacion
616         });
617
618         await newCancion.save();
619
620         res.status(201).send(`${newCancion.titulo} se ha añadido correctamente`);
621     } catch (error) {
622         res.status(500).json({ error: 'Error al añadir la canción' });
623     }
624 });
625
626 app.post('/artista/new', async (req, res) => {
627     try {
628         const nombre = req.body.nombre;
```

```
// #region update

app.post('/cancion/update/:titulo', async (req, res) => {
    const filtro = { titulo: req.params.titulo };
    const valores = {
        artista: req.body.artista,
        titulo: req.body.titulo,
        album: req.body.album,
        duracion_segundos: req.body.duracion_segundos,
        fechaPublicacion: req.body.fechaPublicacion
    };

    const update = await canciones.findOneAndUpdate(filtro, valores, { new: true });

    res.send(`Datos de la canción ${update.titulo} actualizados.`);
});

app.post('/artista/update/:nombre', async (req, res) => {
    const filtro = { nombre: req.params.nombre };
    const valores = {
        nombre: req.body.nombre,
        genero: req.body.genero,
        miembros: req.body.miembros,
        activo: req.body.activo
    };

    const update = await artistas.findOneAndUpdate(filtro, valores, { new: true });

    res.send(`Datos del artista ${update.nombre} actualizados.`);
});
```

```
// #region ELIMINAR
// #region con confirmación

app.get('/artista/del/:nombre', async (req, res) => {
    try {
        const artista = await artistas.findOne({ nombre: req.params.nombre });
        if (artista) {
            return res.send(`
                <html>
                    <body>
                        <h2>Confirmar eliminación</h2>
                        <p>¿Estás seguro de que deseas eliminar al artista ${artista.nombre}?</p>
                        <form action="/artista/delc/${artista.nombre}" method="POST">
                            <button type="submit">Sí, eliminar</button>
                            <a href="/">Cancelar</a>
                        </form>
                    </body>
                </html>
            `);
        } else {
            return res.send(`Artista ${req.params.nombre} no encontrado.`);
        }
    } catch (error) {
        console.error("Error:", error);
        return res.status(500).send("Error al eliminar el artista.");
    }
});

app.get('/album/del/:titulo', async (req, res) => {
    try {
        const album = await albumes.findOne({ titulo: req.params.titulo });
        if (album) {
```

```
836     // #region por filtros
837
838     app.get('/artista/del/activo/:activo', async (req, res) => {
839         let param = req.params.activo;
840         let activo = "";
841         const artista = await artistas.deleteMany({ activo: param });
842
843         console.log(artista);
844         console.log(param);
845
846         if (param === 'false') {
847             activo = " no";
848         }
849
850         if (artista.deletedCount > 0) {
851             return res.send(`${artista.deletedCount} artistas${activo} activos eliminados con éxito.`);
852         } else {
853             return res.send('Artistas no encontrados.');
854         }
855     });
856
857     app.get('/album/del/genero/:genero', async (req, res) => {
858         const album = await albumes.deleteMany({ 'genero.genero': req.params.genero });
859
860         if (album.deletedCount > 0) {
861             return res.send(`${album.deletedCount} álbumes del género ${req.params.genero} activos eliminados con éxito.`);
862         } else {
863             return res.send('Álbumes no encontrados.');
864         }
865     });
866 }
```