Audio Pool

In this assignment your job is to create a RESTful API using .NET for a new emerging company called Audio Pool. The company provides music to the public and they need a structured and fast working API to provide clean, and crisp data to their users. The CTO of the company is in love with REST and therefore REST standards must be held to high standards.

Assignment description

Here below are the functionality that should be provided in this API. The routes are categorized in unauthorized and authorized routes, depending on whether you need authentication or not to access the routes.

Rules

This applies to all endpoints enlisted below:

- If the item is not found with the given filter or restriction, the api should return a suitable HTTP status code for when a resource is not found.
- If a model is not correctly structured when either using POST or PUT, the api should return a suitable HTTP status code to indicate that there was an error made by the client.
- If an error occurs on the server, the client should be notified with a suitable HTTP status code.
- When using POST or PUT, the data sent to the server should be in JSON format within the request body.
- When a resource is created, HATEOAS should be honored for better navigation.
- Responses that make use of HATEOAS should include a property called links which should include both rel and href which describes the relationship and the anchor itself.

Models

There are code-generated values which should be populated when creating and/or updating values in the database. The **ModifiedBy** property set on the entity models, should be a hard-coded value of “AudioPoolAdmin”.

DTOs

- AlbumDetailsDto
    - Id (int)
    - Name (string)
    - ReleaseDate (datetime)
    - CoverImageUrl (string)
    - Description (string)
    - Artists (IEnumerable<ArtistDto>)
    - Songs (IEnumerable<SongDto>)
- AlbumDto
    - Id (int)
    - Name (string)
    - ReleaseDate (datetime)
    - CoverImageUrl (string)
    - Description (string)
- ArtistDetailsDto
    - Id (int)
    - Name (string)
    - Bio (string)
    - CoverImageUrl (string)
    - DateOfStart (datetime)
    - Albums (IEnumerable<AlbumDto>)
    - Genres (IEnumerable<GenreDto>)
- ArtistDto
    - Id (int)
    - Name (string)
    - Bio (string)
    - CoverImageUrl (string)
    - DateOfStart (datetime)
- GenreDetailsDto
    - Id (int)
    - Name (string)
    - NumberOfArtists (int)
- GenreDto
    - Id (int)
    - Name (string)
- SongDetailsDto
    - Id (int)
    - Name (string)
    - Duration (timespan)
    - Album (AlbumDto)
    - TrackNumberOnAlbum (int)

- SongDto
    - Id (int)
    - Name (string)
    - Duration (timespan)

Input models

- AlbumInputModel
    - Name (string)
        - Required
        - Minimum length of 3
    - ReleaseDate (datetime)
        - Required
    - ArtistIds (IEnumerable<int>)
        - Required
    - CoverImageUrl (string)
        - Optional
        - Must be a valid URL
    - Description (string)
        - Optional
        - Minimum length of 10
- ArtistInputModel
    - Name (string)
        - Required
        - Minimum length of 3
    - Bio (string)
        - Optional
        - Minimum length of 10
    - CoverImageUrl (string)
        - Optional
        - Must be a valid URL
    - DateOfStart (datetime)
        - Required
- GenreInputModel
    - Name (string)
        - Required
        - Minimum length of 3

- SongInputModel
    - Name (string)
        - Required
        - Minimum length of 3
    - Duration (timespan)
        - Required
    - AlbumId (int)
        - Required

Entities

- Album
    - Id (int)
    - Name (string)
    - ReleaseDate (datetime)
    - CoverImageUrl (string) **NULL**
    - Description (string) **NULL**
    - DateCreated (code-generated)
    - DateModified (code-generated) **NULL**
    - ModifiedBy (code-generated) **NULL**
    - Foreign keys
        - Artist (many-to-many)
        - Song (one-to-many)
- Artist
    - Id (int)
    - Name (string)
    - Bio (string) **NULL**
    - CoverImageUrl (string) **NULL**
    - DateStarted (string)
    - DateCreated (code-generated)
    - DateModified (code-generated) **NULL**
    - ModifiedBy (code-generated) **NULL**
    - Foreign keys
        - Genre (many-to-many)
        - Album (many-to-many)
- Genre
    - Id (int)
    - Name (string)
    - DateCreated (code-generated)
    - DateModified (code-generated) **NULL**
    - ModifiedBy (code-generated) **NULL**
    - Foreign keys
        - Artist (many-to-many)

- Song
    - Id (int)
    - Name (string)
    - Duration (timespan)
    - DateCreated (code-generated)
    - DateModified (code-generated) **NULL**
    - ModifiedBy (code-generated) **NULL**
    - Foreign keys
        - Album (one-to-many)

(15%) Database

All data should reside in a SQLite database. The **SQLite Browser** (https://sqlitebrowser.org/dl/) is the preferred tool to create, view and maintain the data in your local SQLite database. The database provider which is suitable for SQLite is **Microsoft.EntityFrameworkCore.Sqlite** (https://www.nuget.org/packages/Microsoft.EntityFrameworkCore.Sqlite).

- **(2%)** Create a new empty database and store it in the root of the entry application - the output will be a file with the .db extension
- **(2%)** The connection string should be stored in **appsettings.json**
- **(5%)** Create a database context called **AudioPoolDbContext** which should reside in the repository layer
    - Add entities as database sets
- **(2%)** Setup the **AudioPoolDbContext** using dependency injection in the entry application in **Program.cs**. Afterwards the **AudioPoolDbContext** can be injected to each repository class without relying on **new()**
- **(2%)** After all the steps above have been done, you can make your migrations and eventually update the database using those migrations
- **(2%)** Last step, run the population script which can be found in the assignment description in **Canvas**. If everything has been setup correctly in the steps above, the script should run without any errors

  All data access and manipulation should make use of this database

(35%) Unauthorized routes

Genres

- Get all genres
    - An endpoint (**/api/genres**) which fetches all genres.

```
[
  {
    "id": 1,
    "name": "Disco",
    "_links": {
      "self": {
        "href": "/api/genres/1"
      },
      "artists": [
        {
          "href": "/api/artists/2"
        }
      ]
    }
  }
]
```

- Get genre by id
    - An endpoint (**/api/genres/{id}**) which fetches a genre by id. A URL parameter should be used to provide a dynamic value for the genre id, e.g. /api/genres/1 would fetch a genre with the id 1.

```
{
  "id": 1,
  "name": "Disco",
  "numberOfArtists": 1,
  "_links": {
    "self": {
      "href": "/api/genres/1"
    },
    "artists": [
      {
        "href": "/api/artists/2"
      }
    ]
  }
}
```

Artists

- Get artists
    - An endpoint (**/api/artists**) which fetches all artists. By default there should be a limit of 25 news items. The limit can be changed via a query parameter called pageSize. The pages can be indexed by a query parameter called pageNumber. The artists should be ordered by a descending starting date.


```
{
  "pageNumber": 1,
  "pageSize": 25,
  "maxPages": 1,
  "items": [
    {
      "id": 3,
      "name": "Elvis Presley",
      "bio": "Elvis Aaron Presley (January 8, 1935 - August 16, 1977)
was an American singer and actor. Dubbed the \"King of Rock and Roll\",
[...]",
      "coverImageUrl": "https://tinyurl.com/2p82za3a",
      "dateOfStart": "1954-07-05T00:00:00",
      "_links": {
        "self": {
          "href": "/api/artists/3"
        },
        "edit": {
          "href": "/api/artists/3"
        },
        "delete": {
          "href": "/api/artists/3"
        },
        "albums": {
          "href": "/api/artists/3/albums"
        },
        "genres": [
          {
            "href": "/api/genres/6"
          }
        ]
      }
    }
  ]
}
```

- Get artist by id
    - An endpoint (**/api/artists/{id}**) which fetches an artist by id. A URL parameter should be used to provide a dynamic value for the artist id, e.g. /api/artists/2 would fetch an artist with the id 2.


```
{
  "id": 2,
  "name": "Daft Punk",
  "bio": "Daft Punk were a French electronic music duo formed in 1993 in
Paris by Guy-Manuel de Homem-Christo and Thomas Bangalter. Widely
regarded as one of the most influential acts in dance music history,
they achieved popularity in the late 1990s as part of the French house
movement. [...]",
  "coverImageUrl": "https://tinyurl.com/4rncccxd",
  "dateOfStart": "1993-03-01T00:00:00",
  "albums": [
    {
      "id": 3,
      "name": "Discovery",
      "releaseDate": "2001-03-12T00:00:00",
      "coverImageUrl": "https://tinyurl.com/4urzxxm5",
      "description": "Discovery is the second studio album by the French
electronic music duo Daft Punk, released on 12 March 2001 by Virgin
Records. [...]",
      "_links": {}
    },
    {
      "id": 4,
      "name": "Random Access Memories",
      "releaseDate": "2013-05-17T00:00:00",
      "coverImageUrl": "https://tinyurl.com/5yjuk6py",
      "description": "Random Access Memories is the fourth and final
studio album by the French electronic duo Daft Punk, [...]",
      "_links": {}
    }
  ],
  "genres": [
    {
      "id": 1,
      "name": "Disco",
      "_links": {}
    },
    {
      "id": 3,
      "name": "Electronic",
      "_links": {}
```

```
    },
    {
      "id": 8,
      "name": "French House",
      "_links": {}
    },
    {
      "id": 9,
      "name": "Dance",
      "_links": {}
    }
  ],
  "_links": {
    "self": {
      "href": "/api/artists/2"
    },
    "edit": {
      "href": "/api/artists/2"
    },
    "delete": {
      "href": "/api/artists/2"
    },
    "albums": {
      "href": "/api/artists/2/albums"
    },
    "genres": [
      {
        "href": "/api/genres/1"
      },
      {
        "href": "/api/genres/3"
      },
      {
        "href": "/api/genres/8"
      },
      {
        "href": "/api/genres/9"
      }
    ]
  }
}
```

- Get artist albums
    - An endpoint (**/api/artists/{id}/albums**) which fetches all albums connected to an artist. A URL parameter should be used to provide a dynamic value for the artist id, e.g. /api/artists/5/albums would fetch all albums connected to an artist with the id 5.


```
[
  {
    "id": 9,
    "name": "Duran Duran",
    "releaseDate": "1981-06-15T00:00:00",
    "coverImageUrl": "https://tinyurl.com/y4xu7mnn",
    "description": null,
    "_links": {
      "self": {
        "href": "/api/albums/9"
      },
      "delete": {
        "href": "/api/albums/9"
      },
      "songs": {
        "href": "/api/albums/9/songs"
      },
      "artists": [
        {
          "href": "/api/artists/5"
        }
      ]
    }
  }
]
```

Albums

- Get album by id
    - An endpoint (**/api/albums/{id}**) which fetches an album id. A URL parameter should be used to provide a dynamic value for the album id, e.g. /api/albums/3 would fetch an album with the id 3.


```
{
  "id": 3,
  "name": "Discovery",
  "releaseDate": "2001-03-12T00:00:00",
  "coverImageUrl": "https://tinyurl.com/4urzxxm5",
  "description": null,
  "artists": [
    {
      "id": 2,
      "name": "Daft Punk",
      "bio": "Daft Punk were a French electronic music duo formed in
1993 in Paris by Guy-Manuel de Homem-Christo and Thomas Bangalter.
Widely regarded as one of the most influential acts in dance music
history, they achieved popularity in the late 1990s as part of the
French house movement. [...]",
      "coverImageUrl": "https://tinyurl.com/4rncccxd",
      "dateOfStart": "1993-03-01T00:00:00",
      "_links": {}
    }
  ],
  "songs": [
    {
      "id": 28,
      "name": "One More Time",
      "duration": "00:05:20",
      "_links": {}
    },
    {
      "id": 29,
      "name": "Aerodynamic",
      "duration": "00:03:32",
      "_links": {}
    },
    {
      "id": 30,
      "name": "Digital Love",
      "duration": "00:05:01",
      "_links": {}
    }
```

```
  ],
  "_links": {
    "self": {
      "href": "/api/albums/3"
    },
    "delete": {
      "href": "/api/albums/3"
    },
    "songs": {
      "href": "/api/albums/3/songs"
    },
    "artists": [
      {
        "href": "/api/artists/2"
      }
    ]
  }
}
```


- Get songs on album
    - An endpoint (**/api/albums/{id}/songs**) which fetches all songs connected to an album. A URL parameter should be used to provide a dynamic value for the album id, e.g. /api/albums/7/songs would fetch all songs connected to an album with the id 7.


```
[
  {
    "id": 78,
    "name": "Money Changes Everything",
    "duration": "00:05:05",
    "_links": {
      "self": {
        "href": "/api/songs/78"
      },
      "delete": {
        "href": "/api/songs/78"
      },
      "edit": {
        "href": "/api/songs/78"
      },
      "album": {
        "href": "/api/albums/7"
      }
    }
  },
```

```
  {
    "id": 79,
    "name": "Girls Just Want to Have Fun",
    "duration": "00:03:58",
    "_links": {
      "self": {
        "href": "/api/songs/79"
      },
      "delete": {
        "href": "/api/songs/79"
      },
      "edit": {
        "href": "/api/songs/79"
      },
      "album": {
        "href": "/api/albums/7"
      }
    }
  }
]
```

Songs

- Get song by id
    - An endpoint (**/api/songs/{id}**) which fetches a song by id. A URL parameter should be used to provide a dynamic value for the song id, e.g. /api/songs/37 would fetch a song with the id 37.


```
{
  "id": 37,
  "name": "Voyager",
  "duration": "00:03:47",
  "album": {
    "id": 3,
    "name": "Discovery",
    "releaseDate": "2001-03-12T00:00:00",
    "coverImageUrl": "https://tinyurl.com/4urzxxm5",
    "description": "Discovery is the second studio album by the French electronic music duo Daft Punk, released on 12 March 2001 by Virgin Records. It marked a shift from the Chicago house of their first album, Homework (1997), to a house style more heavily inspired by disco, post-disco, garage house, and R&B. Thomas Bangalter of Daft Punk described Discovery as an exploration of song structures, musical forms and childhood nostalgia, compared to the \"raw\" electronic music of Homework.",
    "_links": {}
  },
  "trackNumberOnAlbum": 10,
  "_links": {
    "self": {
      "href": "/api/songs/37"
    },
    "delete": {
      "href": "/api/songs/37"
    },
    "edit": {
      "href": "/api/songs/37"
    },
    "album": {
      "href": "/api/albums/3"
    }
  }
}
```

(10%) Attribute

The authorized routes should be authorized by using a custom attribute. All other attempts to solve the authorization will result in receiving no grade for this component of the assignment.

1. The attribute ensures decorated routes cannot be accessed unless a token is provided
2. The token should be stored in an HTTP header called **api-token**
3. The value of the correct API token can be a hard-coded value which is validated for each authenticated request

(35%) Authorized routes

Genres

- Create a genre
    - An endpoint (**/api/genres**) which creates a new genre. A request body represented as JSON, should be sent with the request in the form of the GenreInputModel.

Artists

- Create an artist
    - An endpoint (**/api/artists**) which creates an artist. A request body represented as JSON, should be sent with the request in the form of the ArtistInputModel.
- Update an artist
    - An endpoint (**/api/artists/{id}**) which updates an artist. A URL parameter should be used to provide a dynamic value for the artist id, e.g. /api/artists/10 would update an artist with the id 10. A request body represented as JSON, should be sent with the request in the form of the ArtistInputModel.
- Link artist to genre
    - An endpoint (**/api/artists/{artistId}/genres/{genreId}**) which links an artist to a genre. Two URL parameters should be used to provide dynamic values for an artist id and a genre id, e.g. /api/artists/1/genres/2 would link an artist with the id 1 with a genre with the id 2.

Albums

- Create an album
    - An endpoint (**/api/albums**) which creates an album. A request body represented as JSON, should be sent with the request in the form of the AlbumInputModel.
- Delete an album
    - An endpoint (**/api/albums/{id}**) which deletes an album. A URL parameter should be used to provide a dynamic value for the album id, e.g. /api/albums/9 would delete an album with the id 9.

Songs

- Delete a song
    - An endpoint (**/api/songs/{id}**) which deletes a song. A URL parameter should be used to provide a dynamic value for the song id, e.g. /api/songs/9 would delete a song with the id 9.
- Update a song
    - An endpoint (**/api/songs/{id}**) which updates a song. A URL parameter should be used to provide a dynamic value for the song id, e.g. /api/songs/9 would update a song with the id 9.
- Add a song
    - An endpoint (**/api/songs**) which creates a new song. A request body represented as JSON, should be sent with the request in the form of the SongInputModel.

(10%) Structure

- **(5%)** The assignment should be setup using a three-layer structure where you make use of a presentation layer, service layer, repository layer and a project which keeps all models
- **(2.5%)** All models should reside in the Models project (**DTO, InputModels** and **Entities**)
- **(2.5%)** All data access code should reside in the repository layer of the application

Submission

Submit a single compressed file (**.zip, *.rar*) in Canvas. **In order to make the grading less time consuming, erase all data from the database and populate again using the population script.**

Other

In this assignment it is required to use .NET. Any editor will suffice, although Visual Studio Code is a good option.