DOCUMENTAȚIA PROIECTULUI

Galerie de Artă

STUDENȚI

1. Boldianu Marian

2. Encună Miruna Ștefania

**Cuprins**

1. Prezentarea proiectului, ce isi propune, ce probleme rezolva

2. Tehnologiile folosite

3. Baza de date: diagrama bazei de date + scurta prezentare a tabelelor si a relatiilor dintre ele

4. Prezentarea API-ului: screenshot la Swagger pentru a putea vedea toate endpoint-ruile + scurta descriere a CRUD-urilor

5. Prezentare despre cum poate fi utilizata aplicatia: tipuri de utilizatori, ce vede fiecare, autentificare etc

6. Concluzii si contributii (cum v-ati impartit task-urile, ce ati invatat in urma acestui proiect etc)

7. Link GIT catre codul proiectului

## **Prezentarea proiectului**

Proiectul "Galerie de Artă" își propune să creeze o platformă digitală pentru administrarea și vizualizarea operelor de artă și a artiștilor. Aplicația rezolvă probleme legate de organizarea și gestionarea eficientă a unei colecții de artă, permițând utilizatorilor să adauge, să actualizeze și să ștergă informații despre artiști și operele lor de artă.

## **Tehnologiile folosite**

* **Backend**: ASP.NET Core
* **ORM**: Entity Framework Core
* **Baza de date**: SQL Server
* **Autentificare și autorizare**: JWT - Json Web Token
* **Documentare API**: Swagger

1. **Baza de date**
   1. Artists:

* Id: Cheie primară, de tip întreg (int)
* Name: Nume, de tip text (nvarchar(max))
* Nationality: Naționalitate, de tip text (nvarchar(max))

Descriere: Tabela Artists stochează informațiile despre artiști.

* 1. Artworks:
* Id: Cheie primară, de tip întreg (int)
* Title: Titlul operei de artă, de tip text (nvarchar(max))
* Year: Anul creării operei de artă, de tip întreg (int)
* ArtistId: Cheie străină către tabela Artists, de tip întreg (int)

Descriere: Tabela Artworks stochează informațiile despre operele de artă.

* 1. Roles:
* Id: Cheie primară, de tip întreg (int)
* Name: Numele rolului, de tip text (nvarchar(max))

Descriere: Tabela Roles stochează diferitele roluri ale utilizatorilor în aplicație.

* 1. Users:
* Id: Cheie primară, de tip întreg (int)
* Username: Numele de utilizator, de tip text (nvarchar(max))
* PasswordHash: Hash-ul parolei, de tip text (nvarchar(max))

Descriere: Tabela Users stochează informațiile despre utilizatorii aplicației.

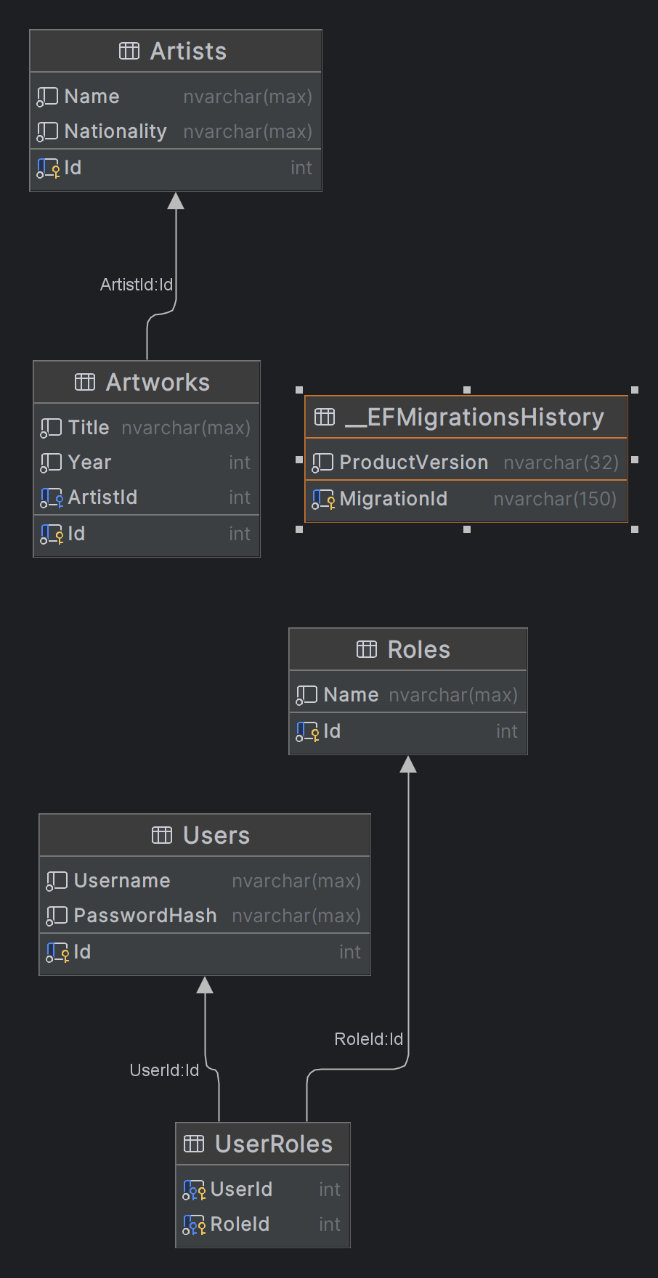
* 1. UserRoles:
* UserId: Cheie străină către tabela Users, de tip întreg (int)
* RoleId: Cheie străină către tabela Roles, de tip întreg (int)

Descriere: Tabela UserRoles este o tabelă de legătură care stochează relațiile dintre utilizatori și rolurile lor. Fiecare intrare asociază un utilizator cu un rol specific.

Relațiile dintre tabele

* **Artists și Artworks**: Relație de tip one-to-many (un artist poate avea mai multe opere de artă).
* **Users și Roles**: Relație de tip many-to-many, gestionată prin tabela de legătură UserRoles (un utilizator poate avea mai multe roluri și un rol poate fi atribuit mai multor utilizatori).

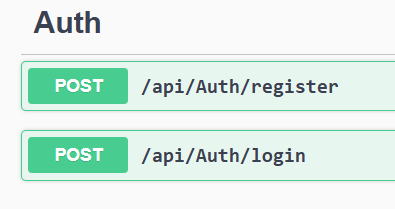
Diagrama bazei de date



## **Prezentarea API-ului**

### 

### 



### Descrierea CRUD-urilor

1. **Artists**
   * GET /api/artists: Obține lista tuturor artiștilor
   * GET /api/artists/{id}: Obține detaliile unui artist după ID
   * POST /api/artists: Adaugă un nou artist
   * PUT /api/artists/{id}: Actualizează detaliile unui artist existent
   * DELETE /api/artists/{id}: Șterge un artist
2. **Artworks**
   * GET /api/artworks: Obține lista tuturor operelor de artă
   * GET /api/artworks/{id}: Obține detaliile unei opere de artă după ID
   * POST /api/artworks: Adaugă o nouă operă de artă
   * PUT /api/artworks/{id}: Actualizează detaliile unei opere de artă existente
   * DELETE /api/artworks/{id}: Șterge o operă de artă

## **5. Utilizarea aplicației**

### Tipuri de utilizatori

* **Administrator**: Poate gestiona artiști și opere de artă (adăugare, actualizare, ștergere).
* **Client**: Poate vizualiza lista artiștilor și operelor de artă, dar nu are permisiuni de modificare.

### Autentificare și autorizare

Utilizatorii trebuie să se autentifice pentru a avea acces la funcțiile de administrare. Doar utilizatorii cu rol de **Administrator** au acces la endpoint-urile de modificare a datelor.

## **6. Concluzii și contribuții**

### Împărțirea task-urilor

* **Miruna:**

1. Implementarea autentificării și autorizării;
2. implementarea unui controller și a endpoint-urilor CRUD pentru fiecare componentă a aplicației.

* **Marian:**

1. Arhitectura bazei de date;
2. Implementarea comunicării cu baza de date.

### Ce am învățat

Acest proiect ne-a oferit oportunitatea de a învăța și de a implementa concepte esențiale precum:

* Arhitectura și implementarea bazelor de date relaționale
* Utilizarea ORM-ului Entity Framework Core
* Gestionarea autentificării și autorizării utilizatorilor
* Documentarea API-urilor folosind Swagger