Documentație Proiect BD

Stanciu Minola-Cristina

Grupa 331AB

Descriere Proiect

Aplicația web pentru **gestionarea istoricului medical dintr-un cabinet veterinar** este un sistem destinat cabinetelor veterinare, care permite urmărirea și gestionarea datelor medicale ale animalelor, medicii care i-au consultat și detaliile consultanțiilor. Aplicația include o pagină de login pentru autentificare, iar administratorii pot adăuga, modifica sau șterge datele legate de animalele înregistrate. Interfața este intuitivă, ușor de utilizat și facilitează gestionarea fișelor medicale și comunicarea eficientă între medicii veterinari și stăpânii animalelor.

Cerințe implementate

- Crearea unei baze de date relationale cu șase tabele
- Setarea cheilor primare cu auto-increment (IDENTITY(1,1))
- Inițializarea automată a sumei din factură cu valoarea 0
- Implementarea operațiilor Creare, Citire, Actualizare, Ștergere pentru toate entitățile
- Utilizarea unui front-end creat în React.js și un back-end implementat cu .NET Core

Etapa de Proiectare

Tabele și Relații

1. **Tabelul Animale**
    - **Coloane:**
        - IDAnimal (PK, auto-increment)
        - IDStapan (FK către Stapani)
        - Nume
        - Specie
        - DataNasterii
        - Rasa
2. **Tabelul Stapani**
    - **Coloane:**
        - IDStapan (PK, auto-increment)
        - Nume
        - Prenume
        - Telefon
        - Email
3. **Tabelul Medici**
    - **Coloane:**
        - IDMedic (PK, auto-increment)
        - Nume
        - Specializare

4. **Tabelul Consultatie**
    - **Coloane:**
        - IDProgramare (PK, auto-increment)
        - IDAnimal (FK către Animale)
        - IDMedic (FK către Medici)
        - Data
5. **Tabelul InregistrareMedicala**
    - **Coloane:**
        - IDInregistrare (PK, auto-increment)
        - IDProgramare (FK către Consultatie)
        - Data
        - Diagnostic
        - Tratament
        - MedicamentePrescrise
6. **Tabelul Factura**
    - **Coloane:**
        - IDFactura (PK, auto-increment)
        - IDInregistrare (FK către InregistrareMedicala)
        - Data
        - Servicii
        - Suma (default 0)

Chei Străine și Relații

- **FK IDAnimal** -> Consultatie
- **FK IDStapan** -> Animale
- **FK IDInregistrare** -> Factura
- **FK IDProgramare** -> InregistrareMedicala
- **FK IDMedic** -> Consultatie

Relații:

- **One-to-One:**
  - IDInregistrare (InregistrareMedicala - Factura)
  - IDProgramare (Consultatie - InregistrareMedicala)
- **One-to-Many:**
  - IDAnimal (Animale - Consultatie)
  - IDStapan (Stapani - Animale)
  - IDMedic (Medici - Consultatie)

Funcționalitatea Aplicației

- **Pagina de Login:** Permite autentificarea administratorilor
- **Pagina Home:** Prezintă butoane pentru fiecare interogare SQL
- **Pagini individuale pentru:** Animale, Stapani, Medici, Consultatii, Inregistrari Medicale, Facturi
- **Funcții disponibile:**
  - Adăugare, modificare, ștergere și afișare

Interogări SQL Implementate

Interogări Simple

```
// 1. Informații despre un animal și medicul său după nume
[HttpGet("AnimalMedic")]
0 references
public JsonResult GetAnimalMedic([FromQuery] string numeAnimal)
{
    string query = @"
        SELECT A.Nume AS NumeAnimal, A.Specie, A.Rasa, A.DataNasterii,
               M.Nume AS NumeMedic, M.Specializare
        FROM Animale A
        INNER JOIN Consultatie C ON A.IDAnimal = C.IDAnimal
        INNER JOIN Medici M ON C.IDMedic = M.IDMedic
        WHERE A.Nume = @NumeAnimal";

    return ExecuteQuery(query, new SqlParameter("@NumeAnimal", numeAnimal));
}
```

```
// 2. Animale care au primit medicamente "Calmante"
[HttpGet("AnimaleMedicament")]
0 references
public JsonResult GetAnimaleMedicament()
{
    string query = @"
        SELECT A.Nume AS NumeAnimal, A.Specie, A.Rasa, IM.Tratament
        FROM Animale A
        INNER JOIN Consultatie C ON A.IDAnimal = C.IDAnimal
        INNER JOIN InregistrareMedicala IM ON C.IDProgramare = IM.IDProgramare
        WHERE IM.MedicamentePrescrise = 'Calmante'";

    return ExecuteQuery(query);
}
```

```
// 3. Medicii cu cele mai multe animale tratate, în ordine descrescătoare
[HttpGet("MediciMaxAnimale")]
0 references
public JsonResult GetMediciMaxAnimale()
{
    string query = @"
        SELECT M.Nume, M.Specializare, COUNT(DISTINCT A.IDAnimal) AS NumarAnimale
        FROM Medici M
        INNER JOIN Consultatie C ON M.IDMedic = C.IDMedic
        INNER JOIN Animale A ON A.IDAnimal = C.IDAnimal
        GROUP BY M.Nume, M.Specializare
        ORDER BY COUNT(DISTINCT A.IDAnimal) DESC";

    return ExecuteQuery(query);
}
```

```
// 4. Servicii și diagnostic pentru animale de tip "Caine"
[HttpGet("ServiciiCaini")]
0 references
public JsonResult GetServiciiCaini()
{
    string query = @"
        SELECT A.Nume, A.Rasa, F.Servicii, I.Diagnostic
        FROM Animale A
        INNER JOIN Consultatie C ON A.IDAnimal = C.IDAnimal
        INNER JOIN InregistrareMedicala I ON C.IDProgramare = I.IDProgramare
        INNER JOIN Factura F ON F.IDInregistrare = I.IDInregistrare
        WHERE A.Specie = 'Caine'";

    return ExecuteQuery(query);
}
```

```
// 5. Medicii cu cele mai multe programări, în ordine descrescătoare
[HttpGet("MediciProgramariDesc")]
0 references
public JsonResult GetMediciProgramariDesc()
{
    string query = @"
        SELECT M.Nume, M.Specializare, COUNT(*) AS NrProgramari
        FROM Medici M
        JOIN Consultatie C ON M.IDMedic = C.IDMedic
        GROUP BY M.Nume, M.Specializare
        ORDER BY NrProgramari DESC";

    return ExecuteQuery(query);
}
```

```
// 6. Medicii care au dat tratamentul "Fizioterapie"
[HttpGet("MediciTratamentFizioterapie")]
0 references
public JsonResult GetMediciTratamentSomn()
{
    string query = @"
        SELECT M.Nume, M.Specializare
        FROM Medici M
        JOIN Consultatie C ON M.IDMedic = C.IDMedic
        JOIN InregistrareMedicala I ON C.IDProgramare = I.IDProgramare
        WHERE I.Tratament = 'Fizioterapie'
        ORDER BY M.Nume ASC";

    return ExecuteQuery(query);
}
```

Interogări Complexe

```
// 7. Medicii care nu au consultat animale
[HttpGet("MediciFaraConsultatii")]
0 references
public JsonResult GetMediciFaraConsultatii()
{
    string query = @"
        SELECT M.Nume, M.Specializare
        FROM Medici M
        WHERE M.IDMedic NOT IN (
            SELECT DISTINCT C.IDMedic FROM Consultatie C
        )";

    return ExecuteQuery(query);
}
```

```
// 8. Detalii despre diagnosticul care apare de x ori în înregistrări
[HttpGet("DiagnosticNumarAparitii")]
0 references
public JsonResult GetDiagnosticNumarAparitii([FromQuery] int diagnosticCount)
{
    string query = @"
        SELECT A.Nume, A.Specie, A.Rasa, I.Diagnostic, I.Tratament
        FROM Animale A
        JOIN Consultatie C ON A.IDAnimal = C.IDAnimal
        JOIN InregistrareMedicala I ON C.IDProgramare = I.IDProgramare
        WHERE I.Diagnostic IN (
            SELECT I1.Diagnostic
            FROM InregistrareMedicala I1
            GROUP BY I1.Diagnostic
            HAVING COUNT(*) = @DiagnosticCount
        )";

    return ExecuteQuery(query, new SqlParameter("@DiagnosticCount", diagnosticCount));
}
```

```
// 9. Animale cu mai multe diagnostice
[HttpGet("AnimaleMultipleDiagnostice")]
0 references
public JsonResult GetAnimaleMultipleDiagnostice()
{
    string query = @"
        SELECT A.Nume, A.Specie, A.Rasa, COUNT(*) AS NrDiagnostice
        FROM Animale A
        JOIN Consultatie C ON A.IDAnimal = C.IDAnimal
        JOIN InregistrareMedicala I ON C.IDProgramare = I.IDProgramare
        GROUP BY A.Nume, A.Specie, A.Rasa
        HAVING COUNT(*) > 2
        ORDER BY NrDiagnostice DESC";

    return ExecuteQuery(query);
}
```

```
// 10. Animalele fără stăpân
[HttpGet("AnimaleFaraStapan")]
0 references
public JsonResult GetAnimaleFaraStapan()
{
    string query = @"
        SELECT Nume, Specie, Rasa
        FROM Animale
        WHERE IDStapan IS NULL";

    return ExecuteQuery(query);
}
```

Tehnologii Utilizate

- **Back-end:** .NET Core (C#)
- **Front-end:** React.js (JavaScript)
- **Baza de Date:** SQL Server