1. Take-Home Feladat:

Summary: Backend fejlesztés - Teve Nyilvántartó (Camel Registry) CRUD API implementáció

Description (**Feladat leírása)**

A feladat egy ASP.NET **Core Minimal API** alapú szolgáltatás elkészítése, amely alkalmas tevék (Camels) adatainak kezelésére. A fejlesztés során SQLite adatbázist és Entity Framework Core-t kell használni. Az elkészült API-hoz átlátható dokumentációt és tesztelő felületet kell biztosítani.

Adatmodell (Camel entitás)

A pontos tulajdonságok definiálása a fejlesztő feladata, de az alábbi mezők megléte elvárt:

- **Id:** Egyedi azonosító.
- **Name:** A teve neve (kötelező mező).
- **Color:** A teve színe.
- **HumpCount:** Púpok száma (validáció: csak 1 vagy 2 lehet).
- **LastFed:** Utolsó etetés ideje.

Funkcionális követelmények (Végpontok)

Az API-nak az alábbi 5 végpontot kell tartalmaznia:

1. **Új teve létrehozása** (POST)
    - Validálja a bemeneti adatokat (pl. púpok száma).
    - Siker esetén a válasz: 201 Created státuszkód **és** a létrehozott objektum (a generált ID-val együtt).
2. **Tevék listázása** (GET)
    - Visszaadja az összes rendszerben lévő tevét egy listában.
3. **Egy adott teve lekérdezése** (GET /{id})
    - ID alapján visszaadja a kért teve adatait.
    - Ha az entitás nem létezik: 404 Not Found.

4. **Teve adatainak módosítása** (PUT vagy PATCH /{id})
    - Meglévő teve adatainak frissítése.
    - Ha az entitás nem létezik: 404 Not Found.
5. **Teve törlése** (DELETE /{id})
    - ID alapján törli az adatbázisból a tevét.

Technikai követelmények

- **Framework:** .NET 8 (vagy újabb).
- **API Architektúra:** Kizárólag **Minimal API** (Controller-ek használata nélkül).
- **Adatbázis:** SQLite.
    - Az alkalmazásnak kezelnie kell az adatbázis fájl létrejöttét (pl. EF Core Migrations vagy EnsureCreated használatával).
- **ORM:** Entity Framework Core.
- **Dokumentáció és UI:**
    - **OpenAPI (Swagger) specifikáció:** Az API végpontjainak szabványos leírása generálódjon le automatikusan (pl. Swashbuckle vagy a beépített Microsoft.AspNetCore.OpenApi segítségével).
    - **Swagger UI:** Legyen elérhető egy interaktív webes felület (pl. /swagger útvonalon), ahol a végpontok könnyen kipróbálhatók böngészőből.
- **Tesztelés: xUnit** keretrendszer használata unit tesztekhez.
- **Külső könyvtárak:**
    - Framework jellegű, mindent átfogó könyvtárak (pl. ABP Framework) használata **tilos**.
    - Kisebb, célzott library-k (pl. FluentValidation, AutoMapper stb.) használata megengedett, a fejlesztő döntése szerint.

Definition of Done (Elfogadási kritériumok)

- A forráskód lefordul és futtatható.
- Az alkalmazás induláskor automatikusan létrehozza az SQLite adatbázist, ha az még nem létezik.

- A CRUD műveletek helyesen működnek és perzisztálódnak az SQLite fájlba.
- Elérhető a Swagger UI felület, és minden végpont dokumentálva van rajta.
- A unit tesztek sikeresen lefutnak.
- A megoldás tiszta, átlátható kódszerkezettel rendelkezik.

2. Take-Home Feladat

Summary: Frontend fejlesztés – Mini Teve Nyilvántartó (Angular)

Description (Feladat leírása)

Készíts egy egyszerű Angular alkalmazást, amely egy meglévő REST API-n keresztül tevék adatainak listázását és szerkesztését valósítja meg.

Adatmodell (Camel)

Az alábbi mezők megléte kötelező:

- id: egyedi azonosító
- name: string, kötelező
- humpCount: number, csak 1 vagy 2 lehet

Funkcionális követelmények

1. Lista nézet

- A rendszerben lévő tevék megjelenítése listában.
- A lista Bootstrap alapú táblázatként jelenjen meg.
- Minden sorban legyen lehetőség a szerkesztésre.

2. Űrlap

- Ugyanaz az űrlap szolgáljon új teve létrehozására és meglévő teve szerkesztésére.
- Az űrlap Bootstrap stílusú form elemeket használjon.

- Validációk:
    - name kötelező, minimum 2 karakter
    - humpCount kötelező, értéke csak 1 vagy 2 lehet
-
- Hibás mezők vizuálisan legyenek kiemelve.
- Mentés:
    - új teve esetén POST
    - szerkesztés esetén PUT
- Mentés után a lista frissüljön.

API végpontok

- GET /api/camels
- POST /api/camels
- PUT /api/camels/{id}

Technikai követelmények

- Angular 17 vagy újabb
- TypeScript
- Reactive Forms
- Angular HttpClient használata
- API base URL environment.ts fájlban
- Bootstrap integrálása (npm vagy CDN)
- Egyedi CSS használata megengedett az alap stílusok finomhangolására
- Hibás API válasz esetén jelenjen meg egy egyszerű, Bootstrap stílusú hibaüzenet

Tesztelés

- Legalább egy unit teszt a form validációira

Definition of Done (Elfogadási kritériumok)

- Az alkalmazás ng serve paranccsal futtatható
- A tevék listája betöltődik az API-ról
- Létrehozás és szerkesztés működik
- A validációk és vizuális visszajelzések megfelelően működnek
- Bootstrap stílusok következetesen használva vannak
- A teszt sikeresen lefut
- A kód áttekinthető és érthető