BZSH Külkereskedelmi Technikum



Mázli állatmenhely

Készítette:

Király Péter

Témavezető Németh Angéla

Józsa Béla Szilágyi Levente

Szoftverfejlesztő és -tesztelő

BUDAPEST 2024

Tartalom

[A projekt célja 2](#_Toc161215416)

[Adatbázis 2](#_Toc161215417)

[Az adatbázis táblái és mezői 2](#_Toc161215418)

[Admin tábla 3](#_Toc161215419)

[Animal tábla 3](#_Toc161215420)

[Enquery tábla 3](#_Toc161215421)

[Kind tábla 4](#_Toc161215422)

[Fejlesztői dokumentáció 4](#_Toc161215423)

[Backend 4](#_Toc161215424)

[Könyvtárszerkezet 4](#_Toc161215425)

[Összefoglalás 5](#_Toc161215426)

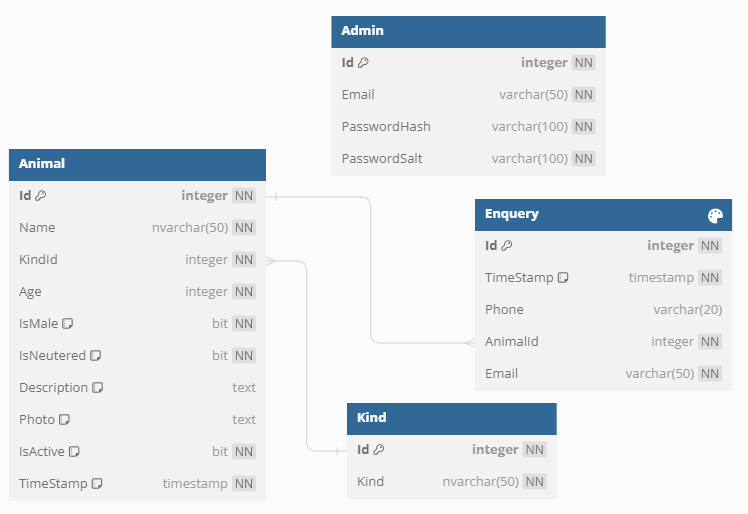
# A projekt célja

A projekt célja egy olyan webes alkalmazás létrehozása, amelyben egy állatmenhely állatait lehet megnézni és örökbe fogadni. A cél valós igényeken alapul, mert egyre több az elárvult, gazdátlanodott kisállat, aki szerető otthonra vágyik, és szerencsére egyre többen vannak azok, akik szívesen fogadnának örökbe kutyát, macskát, nyulat, ill. egyéb kis kedvencet.

# Adatbázis

Az állatok és a hozzájuk kapcsolódó műveleti adatok tárolásáért az adatbázis felelős, melyet MySQL nyelven hoztunk létre, az ingyenes Aiven szerveren tárolva. Ezen keresztül tudjuk kezelni az összes admin funkciót, mint állítok listázása, új állat felvétele, módosítása, inaktívvá tétele. Ezen felül minden user számára elérhető szolgáltatás, az állatok listázása, illetve az érdeklődés, mely gombra, ha rákattint a felhasználó, e-mailt küld az admin e-mail címére az érdeklődésről, amit később hasznos lehet visszakeresni, hogy mely állatra mikor volt érdeklődés milyen felhasználói e-mailről.

## Az adatbázis táblái és mezői



## Admin tábla

Az admin tábla tartalmaz egy Id (Primary Key) mezőt, egy e-mail mezőt, melyre megkapja az állatok iránti érdeklődéseket, a PasswordHash és PasswordSalt-al megoldjuk a biztonságos jelszókezelést, melynek során a sózott jelszót az MD5 algoritmus egy hash-elt karakterlánccá alakítja.

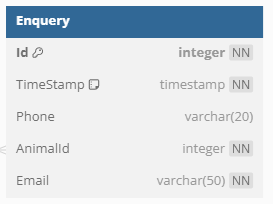
## Animal tábla

Az animal tábla tartalmazza a menhely állataira vonatkozó tulajdonságokat. Az Id mezője a PK (és AUTO INCREMENT). A Name az állat nevét tárolja, a KindId idegenkulcs a Kind tábla Id-jára. Ezen kívül megadjuk az állat korát (Age), a neme vagy hím, vagy nőstény, így egy boolean flag (1-es vagy 0) határozza meg, hogy hím-e az adott állat: IsMale. Mivel lehet ivartalanított is, felvettünk egy IsNeutered mezőt is, ezt is boolean flag tárolja. A Description-nel a tulajdonságot szövegesen tároljuk.

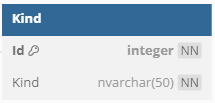
A Photo Base64-es kódolással tárolja az állatok fotóit (text típusú). Az IsActive mező boolean flag-el határozza meg, hogy a menhelyen van-e az állat, vagy elvitték már.

A TimeStamp az állat felviteli időpontját jelöli az adatbázisba.

## Enquery tábla

Az Enquery tábla tartalmazza az Id (PK) mezőt, a TimeStamp-et, mely rögzíti az érdeklődés időpontját, az érdeklődő (user) telefonszámát és e-mail címét, ill. az AnimalId-t, ami FK az Animal tábla Id-jára.

## Kind tábla

A Kind tábla az állatok fajtáit tartalmazza az Id-val (PK), és a Kind mezővel, ami a fajta neve.

# Fejlesztői dokumentáció

## Backend

Készítéshez használt programok:

ASP .NET Framework 6.0  
C#  
Visual Studio 2022

A kódot git verziókövetővel használjuk.

Könyvtárszerkezet:

backend/  
 |-Controllers/  
 | |-AdminController.cs  
 | |-AnimalController.cs  
 | |-EnqueryController.cs  
 | |-KindContoller.cs  
 |-Helpers/  
 | |-HashHelper.cs  
 | |-JWTHelper.cs  
 | |-RemoveAuthorizeFilterOperationFilter.cs  
 |-Models/  
 | |-RequestModels/  
 | | |-LoginRequestModel.cs  
 | |-ResponseModels/  
 | | |-AnimalsResponseModel.cs  
 | | |-BaseResponseModel.cs  
 | | |-EnqueryResponseModel.cs  
 | | |-KindResponseModel.cs  
 | | |-LoginResponseModel.cs  
 | |-Admin.cs  
 | |-Animal.cs  
 | |-Enquery.cs  
 | |-Kiind.cs  
 |-AllatmenhelyDbContext.cs  
 |-appsettings.json  
 |-Program.cs  
 |-BackendTests/  
 | |-TestHelper.cs  
 | |-ControllersUnitTests/  
 | | |-AdminControllerTests.cs  
 | | |-AnimalContollerTests.cs  
 | | |-EnqueryControllerTests.cs  
 | | |-KiindControlleTests.cs  
 | | |-TestBase.cs  
  
Végpontok:

|  |  |  |  |
| --- | --- | --- | --- |
| **Végpont** | **Metódus** | **Azonosítás** | **Leírás** |
| /Admin/Login | POST | nem | Bejelentkezés (csak adminnak) |
| /Animal/GetAllAnimal | get | nem | Állatok lekérdezése. |
| /Animal/GetAnimalById | get | nem | Egy állat lekérdezése. |
| /Animal/GetAnimalByKindId | get | nem | Állatok lekérdezése fajta szerint. |
| /Animal/CreateAnimal | post | igen | Állat felvétele. |
| /Animal/UpdateAnimal | put | igen | Állat módosítása. |
| /Animal/DeleteAnimal | delete | igen | Állat törlése/inaktívvá tétele. |
| /Enquery/GetAllEnqueries | get | igen? | Érdeklődések lekérdezése. |
| /EnqueryGetEnqueriesByAnimalId | get | igen? | Érdeklődések lekérdezése egy állatra. |
| /Enquery/CreateEnquery | post | nem | Érdeklődés. |
| /Kind/GetAllKinds | get | nem | Fajták lekérdezése. |
| /Kind/GetKindById | get | nem | Fajta lekérdezése Id alapján. |
| /Kind/CreateKind | post | igen | Fajta felvétele. |
| /Kind/UpdateKind | put | igen | Fajta módosítása. |
| /Kind/DeleteKind | delete | igen | Fajta inaktiválása. |

Általános működés:

A vezérlést, az üzleti logikát kontrollerek valósítják meg, melyek felelősek az adatbázis műveletekért, valamint tartalmazzák a frontend által hívható végpontokat.

A kliens aszinkron kéréseket intéz a backend REST API végpontok hívásával, melyek tartalmazzák a műveletekhez szükséges adatokat. A backend – amennyiben szükséges – az adatbázisból kinyert adatokat feldolgozva a HTTP Respose BODY-jában küldi vissza az eredményt.

## Osztályok

**Models:**

Feladata az adatbázis tábláinak reprezentálása.

RequestModel

Ez az osztály a bejelentkezési kérés modelljét reprezentálja, e-mail címet és jelszót tárol. A frontend részéről elküldött adatokat a RequestModel osztály fogadja, ezeket az adatokat a backend feldolgozza (autentikációra).

ResponseModel

AnimalsResponseModel

Az osztály olyan válaszmodelleket definiál, mely az állatokkal kapcsolatos adatok visszaadására szolgál a frontend felé.

**Controllers:**

A vezérlők feladata a weboldal backend-jén az üzleti logika kezelése, a HTTP kérések fogadása és feldolgozása (GET, POST, PUT, DELETE), validálása.

AdminController

Ez a kontroller felelős az adminisztrátor bejelentkezésének kezeléséért. Sikeres hitelesítés esetén JWT tokent generál, visszaküldi, ellenkező esetben hibaüzenetet küld.

AnimalController

Feladata az állatokkal kapcsolatos műveletek, és ha a hívás, vagy adatfeldolgozás során valamilyen problémába ütközünk, akkor egy megfelelő objektumban tájékoztatjuk a klienst.

*Metódusok*:  
GetAllAnimals()  
Az összes állat lekérdezéséért felelős az adatbázisból.

GetAnimalById()  
Egy bizonyos állat lekérdezését végzi, visszaadja annak adatait.

GetAnimalsByKindId()  
Adott fajtához tartozó állatot kérdezhetünk le ezzel a metódussal.

CreateAnimal()  
Új állat létrehozásáért felelős.

UpdateAnimal()  
 A metódus az állat módosítását végzi.

DeleteAnimal()  
Esetünkben az állat inaktívvá, ill. aktívvá tételét tehetjük meg a Delete-el.

EnqueryController

Feladata a felhasználói érdeklődésekkel kapcsolatos műveletek kezelése. Minden művelet biztosítja a hibák kezelését.

*Metódusok*:  
GetAllEnqueries()  
Az összes érdeklődés lekérdezését végzi.

GetEnqueriesByAnimalId()  
Azon érdeklődések lekérdezését végzi, melyek egy adott állathoz tartoznak.

CreateAnimal()  
Új érdeklődés felvétele.

KindController()

Feladata az állatfajtákkal kapcsolatos műveletek kezelése.

GetAllKinds()  
Összes állatfajta listázása.

CreateKind()  
Új fajta létrehozása.

UpdateKind()  
Fajta módosítása az adatbázisban.

DeleteKind()  
Állatfajta törlése az adatbázisból.

# Teszt dokumentáció

kgkgkg

# Felhasználói dokumentáció

kgkgjg

# Összefoglalás

Olyan webes felület létrehozása volt a cél, ahol a felhasználók állatokat, kis társakat kereshetnek, és fogadhatnak örökbe, ezzel is segítve egy kis élőlény sorsát, hogy szerető, gondoskodó családban nevelkedhessen. Későbbiekben szeretnénk az oldal lehetőségeit bővíteni. A weboldalt böngészve kereshetünk bizonyos állatfajokra, pl. csak kutyára, kilistázhatjuk őket, ahol is láthatjuk az adott állat tulajdonságait.

Az eredeti ötletünk egy robotkölcsönző létrehozása volt, viszont azt elvetettük, mert nem találtuk elég életszerűnek és sokkal hasznosabbnak találtuk az elhagyott, elárvult állatok segítését.

Úgy érezzük, hogy lehetőségeinkhez képest sikerült jól megvalósítani a projektet, talán a legnagyobb kihívást az jelentette, hogy eldöntsük, milyen nyelven készüljön a program.   
Úgy döntöttünk, hogy a backend C#-ban íródik majd, a frontendet pedig Angular keretrendszerben valósítottuk meg. de mindezek előtt létre kellett hoznunk az átgondolt, jól megtervezett, anomáliáktól mentes adatbázist.

A felhasználók sok esetben nem kedvelik az olyan oldalakat, ahol bizonyos funkciók regisztrációhoz kötöttek, esetünkben az „Érdeklődés” gomb váltja ki ezt a funkciót.

## További célkitűzések

* Vélemény írási lehetőség a felhasználók számára.
* Kereső fejlesztése, optimalizálása (kifejezésekre is lehessen keresni, ne csak szavakra).
* Admin felület bővítése.