Budapesti Műszaki Szakképzési Centrum   
Petrik Lajos Két Tanítási Nyelvű Technikum



SZOFTVERFEJLESZTŐ ÉS -TESZTELŐ TECHNIKUS SZAKMA

5 0613 12 03

**ÉLETHANG**

**Készítette**: Ambrus Pálma Zita, Kovács Krisztán, Nádasdi Barbara

Budapest, 2022.

Budapesti Műszaki Szakképzési Centrum   
Petrik Lajos Két Tanítási Nyelvű Technikum

**Nyilatkozat**

Alulírott **Ambrus Pálma Zita**, **Kovács Krisztán** és **Nádasdi Barbara** kijelentjük, hogy ez a vizsgaremek saját tudásunk, önálló munkánk terméke.

A vizsgaremek közös részeit *Ambrus Pálma Zita*, *Kovács Krisztián* és *Nádasdi Barbara* készítettük, ezeket pontosan jelöltük.

*Ambrus Pálma Zita* *Kovács Krisztián* *Nádasdi Barbara*

Budapest, 2022

# Tartalomjegyzék

[Tartalomjegyzék 3](#_Toc101734116)

[1. Bevezetés 7](#_Toc101734117)

[1.1 Röviden a programról 7](#_Toc101734118)

[1.2 Célunk a programmal 8](#_Toc101734119)

[1.3. Roadmap 8](#_Toc101734120)

[2.Témaválasztás 8](#_Toc101734121)

[2.1 Témaválasztás indoklása 8](#_Toc101734122)

[Backend 9](#_Toc101734123)

[3. Fejlesztői dokumentáció 9](#_Toc101734124)

[3.1 Alkalmazott fejlesztési eszközök 9](#_Toc101734125)

[3.2 Adatmodell 9](#_Toc101734126)

[4. Tesztelési dokumentáció 11](#_Toc101734127)

[4.1 Telepítés a teszteléshez 11](#_Toc101734128)

[4.2 Api végpontok tesztelés 12](#_Toc101734129)

[Asztali Alkalmazás – Kovács Krisztián 21](#_Toc101734130)

[3. Fejlesztői dokumentáció 21](#_Toc101734131)

[3.1 Alkalmazott fejlesztői eszközök 21](#_Toc101734132)

[3.3 Részletes feladatspecifikáció, algoritmusok 21](#_Toc101734133)

[3.4 Tesztelés 33](#_Toc101734134)

[3.5 Továbbfejlesztési lehetőségek 35](#_Toc101734135)

[4. Felhasználói dokumentáció 35](#_Toc101734136)

[4.1 A program általános specifikációja 35](#_Toc101734137)

[4.2 Rendszerkövetelmények 35](#_Toc101734138)

[4.3 A program telepítésének és konfigurálásának a leírása 36](#_Toc101734139)

[4.4 A program használatának a részletes leírása 36](#_Toc101734140)

[5. Összegzés 41](#_Toc101734141)

[Weblap - Ambrus Pálma Zita 43](#_Toc101734142)

[3. Felületterv 43](#_Toc101734143)

[3.1 Fejlesztés nekikezdése előtti tervek 43](#_Toc101734144)

[3.2 Elkészült honlap 44](#_Toc101734145)

[3.3 Kinézet 45](#_Toc101734146)

[4. Útmutatók 47](#_Toc101734147)

[4.1 Fejlesztői 47](#_Toc101734148)

[4.2 Felhasználói 51](#_Toc101734149)

[5. Tesztelések 53](#_Toc101734150)

[5.1 Csak bejelentkezés után elérhető oldalak bejelentkezés előtt megpróbált elérése 53](#_Toc101734151)

[5.2 Regisztráció 54](#_Toc101734152)

[5.3 Bejelentkezés 54](#_Toc101734153)

[5.4 Kijelentkezés 55](#_Toc101734154)

[5.5 Örökbefogadás 55](#_Toc101734155)

[5.6 Kép feltöltése 55](#_Toc101734156)

[5.7 Kép leírásának a módosítása 56](#_Toc101734157)

[5.8 Kép törlése 57](#_Toc101734158)

[5.9 Ikonos átirányítás 57](#_Toc101734159)

[5.10 Statisztika változások 57](#_Toc101734160)

[6. Ismert hibák 58](#_Toc101734161)

[7. Továbbfejlesztési lehetősségek 59](#_Toc101734162)

[8. Összefoglalás a projekt készítése alatti élményeimről 59](#_Toc101734163)

[8.1 Magyarul 59](#_Toc101734164)

[8.2 In English 60](#_Toc101734165)

[Mobil Alkalmazás – Nádasdi Barbara 62](#_Toc101734166)

[3.Felületterv 62](#_Toc101734167)

[4. Fejlesztői dokumentáció 65](#_Toc101734168)

[4.1 Alkalmazott fejlesztési eszközök 65](#_Toc101734169)

[5. Részletes feladatspecifikáció, algoritmusok 65](#_Toc101734170)

[5.1 Cat mappa 66](#_Toc101734171)

[5.2 LoadingActivity 68](#_Toc101734172)

[5.3 MainActivity 68](#_Toc101734173)

[5.4 RequestHandler 69](#_Toc101734174)

[5.5 Response 70](#_Toc101734175)

[5.6 Token 70](#_Toc101734176)

[6. Felhasználói dokumentáció 71](#_Toc101734177)

[6.1 A program általános specifikációja 71](#_Toc101734178)

[6.2 Rendszerkövetelmény 71](#_Toc101734179)

[6.3. A program telepítésének és konfigurálásának leírása 72](#_Toc101734180)

[6.4 A program használatának részletes leírása 73](#_Toc101734181)

[7. Tesztelési dokumentáció 75](#_Toc101734182)

[8. Továbbfejlesztési lehetőségek 79](#_Toc101734183)

[9. Összegzés 80](#_Toc101734184)

[9.1 Magyarul 80](#_Toc101734185)

[9.2 In English 81](#_Toc101734186)

[Ábrajegyzék 83](#_Toc101734187)

[Forrásjegyzék 84](#_Toc101734188)

# 1. Bevezetés

A BMSzC Petrik Lajos Kéttannyelvű Technikumában a kétéves Szoftverfejlesztő – és Tesztelő képzésünk lezárása képen egy 2 vagy 3 fős csapatban kellett egy vizsgamunkát készíteni, amihez tartozik egy minden tag által fejlesztett backend és egy egyéni vékonykliens, aminek hasonló tulajdonságokkal kell rendelkeznie.

A csapatunk 3 tagból áll: Ambrus Pálma Zita, Kovács Krisztián és Nádasdi Barbara 14/S tanulói. Témánk választásakor figyelembe vettük, hogy lehető legjobban életszerű legyen, ezért egy létező kutyák és macskák megmentésével, illetve örökbefogadtatásával foglalkozó alapítványt vettünk alapul, amelynek neve Élethang Alapítvány.

Már témaválasztás előtt felmértük ki miben érdekelt vagy jó, ezért született meg az a döntés, hogy A. P. Zita csinálja a **weblapunkat**, amely a tervek szerint VUE keretrendszer segítségével készült volna el, de a végleges az a Laravel blade-ban készül el, az **asztali alkalmazást** K. Krisztián csinálja, mely korábban C# volt, viszont a végleges Java nyelven íródott, a **mobil alkalmazást** N. Barbara készíti szintén Java nyelven.

## 1.1 Röviden a programról

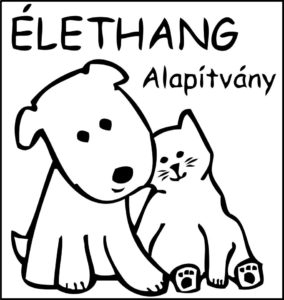
Az **Asztali alkalmazás** egy admin felület. Bejelentkezést követően, (csak a szuperadmin és a sima admin képes) hozzáférnek az adattáblákhoz, amelyek tartalmát törölni, módosítani, illetve új adatot tudnak hozzáadni. Így az itt elvégzett változások kihatással lesznek a weboldalra és a mobil alkalmazásra is.

A **Weboldal** lényege az ismeretterjesztés lesz. Az alapítvány és az állatok adatai megtalálhatóak, regisztrációra és bejelentkezésre is képes látogatóknak. Előreláthatólag az oldal naptárral is el lesz látva, ahol különböző elfoglaltságok, fontosabb alkalmak időpontjai lesznek megtalálhatóak.

A **Mobil Alkalmazásban** szintén lehetőség lesz regisztrációra és bejelentkezésre. Fellelhetőek lesznek külön oldalon a macskák és a kutyák egyaránt. Jelentkezni lehet majd a meghirdetett programokra. Ha a regisztrált felhasználó szeretne, akkor jelezni is tudja felénk, hogy örökbe szeretne fogadni egy vagy több állatot.

A **Backend** Laravel keretrendszerben készült, amely lényegében a 3 kliensnek a héttere, ami azt jelenti, hogy ez a program, biztosítja számukra a funkciók megfelelő működését. Továbbá ez áll közvetlen kapcsolatba az adatbázissal.

Az alapítványnak már volt **logója**, ezért mi is azt használtuk minden alkalmazásunkban:



1. ábra - Logó

## 1.2 Célunk a programmal

Az elsődleges célunk az volt, hogy tudjuk népszerűsíteni az alapítványt és meggyőzni az embereket egy olyam hasznos cél érdekében, hogy segítsék az elveszett/elkóborolt/kidobott állatok egészségi állapotának fenntartását és javítását. Ezt megtudják tenni az adó 1% felajánlásával, adományokkal, ami lehet pénzbeli vagy tárgyi, de ez most már akár egy telefonhívással is megoldható.

A felhasználók önkéntessé tudnak válni, azzal, hogy a weboldalon és a mobil alkalmazásban egyaránt jelezni tudják, hogy ha szeretnének örökbe fogadni vagy programokra jelentkezni, mint például kutyasétáltatásra.

## 1.3. Roadmap

2. ábra - Roadmap

# 2.Témaválasztás

## 2.1 Témaválasztás indoklása

Amikor elsőnek gondolkoztunk a vizsgaremekünkön, már tudtuk, hogy valami állatos témát szeretnénk feldolgozni. Végül a végső ötlet Nádasdi Barbarától jött, amikor mondták, hogy életszerűnek kell lennie a projecteknek. Ő jól ismeri az Alapítvány alapítóját/ tulajdonosát és megbeszélte vele a terveinket. Amint megkaptuk az engedélyt arra, hogy felhasználjuk az alapítvány valós adatait, hozzá is láttuk a tervezéshez. Olyan funkciókat próbáltunk kitalálni, ami nem csak azt mutatja be, amit az órákon tanultunk, hanem kisebb kihívásokat is kerestünk, amivel megmutathatjuk mire vagyunk képesek.

# Backend

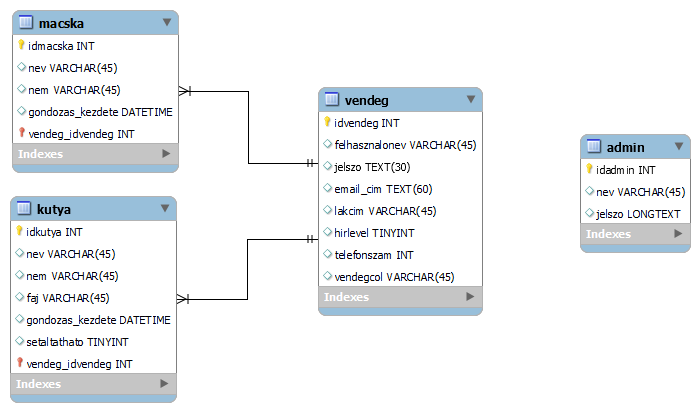
# 3. Fejlesztői dokumentáció

## 3.1 Alkalmazott fejlesztési eszközök

* Keretrendszer:
  + Laravel
* Fejlesztői környezet:
  + Visual Studio Code
* Használt programozási nyelvek:
  + PHP
  + MySQL
* Adatbázis kezelő rendszerek:
  + PHP MyAdmin
  + MySQL Workbench
* Dokumentáció, csapatmunkát segítő eszközök:
  + Microsoft Word
  + Canva
  + Roadmunk
  + Notepad++

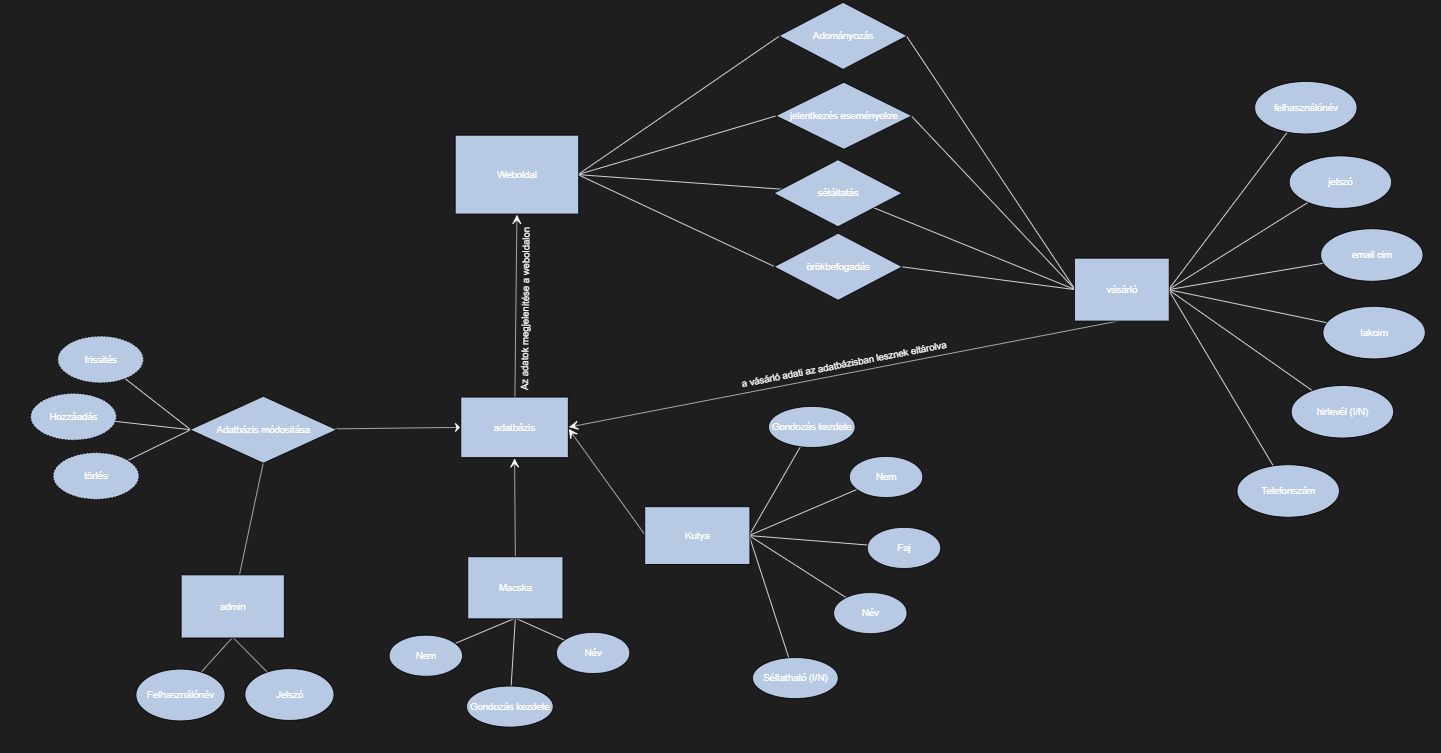
## 3.2 Adatmodell

Az adatbázisunk kezdetekbe 3 táblából állt, ami a kutya, macska és a vendég (most már felhasználók) adatait tartalmazó tábla volt, de pár órával a „kész” adatbázis terv után rájöttünk, hogy szükségünk van egy adminokat eltároló mezőre is, mivel az asztali alkalmazásunk egy komplett admin felület.



3. ábra - Kezdetleges adatbázis szerkezet

Ehhez a kezdetleges adatbázishoz csináltunk egy UML diagrammot, annak érdekében, hogy mi és a segítőkész tanárjaink is átlássák, hogy mire lesz képes a felhasználó (lásd: vendég), az admin és mindez hogyan kapcsolódik az állatokhoz.



4. ábra - Kezdetleges UML diagramm

Minden egyes alkalommal amikor belekezdtünk a tervezésbe egyre több és több táblánk lett ezért a sok-sok változtatás után a mostani egyben a végleges adatbázisunk 11 adattáblát tartalmaz. Ez az adatbázis már élő adatokkal is felvan töltve. A létrehozott kapcsolatoknak és a jól megírt metódusoknak köszönhetően, a laravel projectben már működik a programjelentkezés A képen szöveg, beltéri, képernyőkép látható

Automatikusan generált leírásés az örökbefogadás is.

. ábra - Végleges adatszerkezet

# 4. Tesztelési dokumentáció

## 4.1 Telepítés a teszteléshez

A github-on található linket le kell clone-ozni a C:\xampp\htdocs mappába VS Code-on keresztül, mivel a xampp csak így tud kapcsolatba lépni a project-ünkkel és így lehet tesztelni. Miután ezt megtettük az alábbi linkről: [*https://github.com/Kkrisztian08/elethang\_backend.git*](https://github.com/Kkrisztian08/elethang_backend.git), a PHP MyAdmin-ban létre kell hozni egy új adatbázist jelen esetben „elethang” néven.

*composer install*: Az alkalmazás alapját ezzel biztosítjuk, telepíti és frissíti a függőségeket

*cp .\.env.example .env:* Létre kell hozni az env.example másolatát, amibe át kell írnunk az adatbázisnak a nevét, arra amit előzőleg létrehoztunk, hogy tudja a laravel-project melyik adatbázishoz kell kapcsolódni

php .\artisan key:generate: A laravel project megfelelő működéséhez szükséges, enélkül nem lehet tesztelni se mert azonnal hibára fut

php artisan migrate: Az adatbázisunkat feltölti a már megírt táblákkal

php artisan db:seed: Az előzőleg feltöltött táblákba betöltjük az adatokat

php artisan serve: Elindítjuk a laravel szerverét, hogy tesztelni tudjuk a meglévő kódot Thunder Client-ben vagy az egyéni vékonyklienseken

## 4.2 Api végpontok tesztelés

|  |  |
| --- | --- |
| **/api/cat** | |
| Kérés metódus | GET |
| Leírás | Visszaadja az adatbázisban eltárolt összes macska adatát |
| Input (JSON) |  |
| Output (JSON) | [  {  "id": 1,  "name": "Zénó",  "gender": "Kandúr",  "likely\_bday": "2019-07-15",  "external\_property": "Nagy termetű, fehér alapon leopárd cirmos",  "description": "Zénó, testvérével Zillével 2019.08.25-én került a gondozásunkba, miután az akkor kb. 5 hetes apróságokat valaki kitette a gyáli zöldségpiacra. Zénó 2019 július közepén születhetett. A nálunk töltött idő alatt kezeltük a hasmenését, a külső belső élősködőit, kapott oltásokat, chipet és átesett az ivartalantáson is. A FIV/FELV tesztje negatív lett.Nagy termetű. aktív, játékos cica, más cicákkal némi válogatással megtalálja a közös hangot. Szereti a simogatást, bújik, dorombol az emberi érintés hatására.",  "interest": 1,  "adoption\_id": null  },  {  "id": 2,  "name": "Jura",  "gender": "Nőstény",  "likely\_bday": "2021-06-25",  "external\_property": "Fehér alapon fekete foltos, kisgömb forma, buci fejű",  "description": "Jura 2021.06.25-én, a születése napján a mamájával és a testvérével együtt került a gondozásunkba. Jura fehér alapon fekete foltos, kisgömb forma, buci fejű, gyönyörű nőstény kiscica. Korának megfelelő fejlettségű, önállóan eszik, alomra jár, játékos, aktív bársonytalpú. Megkapta az első oltását és a chipet, egészséges, önálló, így gazdikész. A mamája FIV/FELV negatív. Ivartalanítási kötelezettséggel.",  "interest": 1,  "adoption\_id": null  },  **.**  **.**  **.**  {  "id": 10,  "name": "Soma",  "gender": "Kandúr",  "likely\_bday": "2019-05-15",  "external\_property": "Kistestű, vékony csontú, selymes bunda",  "description": "Soma 2019 május közepén született kiskandúr, aki 4 testvérével került a gondozásunkba. A mamájuk FIV/FELV negatív. Csúnya náthával került hozzánk és a testvérek közül ő volt a legrosszabb állapotban, így kórházi kezelésre, infúzióra szorult. A cica náthából szerencsésen kigyógyult, és az egykori elfekvő, beteges kiscica mára igazi örökmozgó kisördög lett. Többször féreghajtott, parazitáktól mentes, oltási programja rendben, chippel is rendelkezik, és már az ivartalanításon is átesett. Játékos, érdeklődő, nagyon aktív kis bársonytalpú, aki dorombolva keresi az emberi simogatást, társaságot. Termetét, kinézetét tekintve a mamájára ütött, mert kistestű, vékony csontú, de gyönyörű selymes bundája van. Alomra jár, szépen eszik.",  "interest": 1,  "adoption\_id": null  }  ] |

|  |  |
| --- | --- |
| **/api/cat** | |
| Kérés metódus | POST |
| Leírás | Az adatbázisunkhoz hozzáadunk egy új macskát |
| Input (JSON) | {  "name": "Új cica",  "gender": "Nőstény",  "likely\_bday": "2018-09-19",  "external\_property": "Nagy testű fehér",  "description": "Ide jön a hosszú leírás a cicáról",  "interest": 1,  "adoption\_id": null  } |
| Output (JSON) | {  "name": "Új cica",  "gender": "Nőstény",  "likely\_bday": "2018-09-19",  "external\_property": "Nagy testű fehér",  "description": "Ide jön a hosszú leírás a cicáról",  "interest": 1,  "adoption\_id": null,  "id": 11  } |

|  |  |
| --- | --- |
| **/api/cat/11** | |
| Kérés metódus | PUT |
| Leírás | Az újonnan létrehozott macska adatait módosítjuk |
| Input (JSON) | {  "name": "Módosított cica",  "gender": "Nőstény",  "likely\_bday": "2018-09-19",  "external\_property": "Ide a külső tulajdonságok jönnek",  "description": "Ide jön a hosszú leírás a cicáról",  "interest": 1,  "adoption\_id": null  } |
| Output (JSON) | {  "id": 11,  "name": "Módosított cica",  "gender": "Nőstény",  "likely\_bday": "2018-09-19",  "external\_property": "Ide a külső tulajdonságok jönnek",  "description": "Ide jön a hosszú leírás a cicáról",  "interest": 1,  "adoption\_id": null  } |

|  |  |
| --- | --- |
| **/api/cat/11** | |
| Kérés metódus | DELETE |
| Leírás | Töröljük a legkésőbb létrehozott macskát az adatbázisból |
| Input (JSON) |  |
| Output (JSON) |  |
| Státuszkód | 204 No Content |

|  |  |
| --- | --- |
| **/api/cat/11** | |
| Kérés metódus | DELETE |
| Leírás | Töröljük a már törölt macskát az adatbázisból |
| Input (JSON) |  |
| Output (JSON) | {  "message": "A megadott azonosítóval nem található macska."  } |

|  |  |
| --- | --- |
| **/api/application** | |
| Kérés metódus | GET |
| Leírás | Lekérjük az összes programjelentkezést |
| Input (JSON) |  |
| Output (JSON) | [  {  "id": 1,  "program\_info\_id": 4,  "user\_id": 3  },  {  "id": 2,  "program\_info\_id": 5,  "user\_id": 1  },  **.**  **.**  **.**  {  "id": 5,  "program\_info\_id": 1,  "user\_id": 4  }  ] |

|  |  |
| --- | --- |
| **/api/application/5** | |
| Kérés metódus | DELETE |
| Leírás | Kitöröljük az utolsó jelentkezést |
| Input (JSON) |  |
| Output (JSON) |  |
| Státuszkód | 204 No Content |

|  |  |
| --- | --- |
| **/api/application/5** | |
| Kérés metódus | DELETE |
| Leírás | Kitöröljük az utolsó jelentkezést |
| Input (JSON) |  |
| Output (JSON) | {  "message": "A megadott azonosítóval nem található program jelentkezés."  } |
| Státuszkód | 404 Not Found |

|  |  |
| --- | --- |
| **/api/login** | |
| Kérés metódus | POST |
| Leírás | Bejelentkezés |
| Input (JSON) | {  "username": "Nbarbi11",  "password": "barbi1234"  } |
| Output (JSON) | {  "token": "7|LgpnSpz32OK3VG6eReRFtKURdKYF1qTcnEgo27KA"  } |

|  |  |
| --- | --- |
| **/api/login** | |
| Kérés metódus | POST |
| Leírás | Bejelentkezés |
| Input (JSON) | {  "username": "Nbarbi11",  "password": "barbi123"  } |
| Output (JSON) | {  "message": "Helytelen név vagy jelszó"  } |

|  |  |
| --- | --- |
| **/api/register** | |
| Kérés metódus | POST |
| Leírás | Regisztráció (weboldal) |
| Input (JSON) | {  "name": "Új felhasználó",  "username": "ujVagyok99",  "birthday": "2001-08-30",  "address": "1111 Valahol valahol utca 45",  "phone\_number": "+36302222222",  "email": "valaki@gmail.com",  "password": "Valaki99Valahonnan"  } |
| Output (JSON) | {  "admin": 0,  "name": "Új felhasználó",  "username": "ujVagyok99",  "birthday": "2001-08-30",  "address": "1111 Valahol valahol utca 45",  "phone\_number": "+36302222222",  "email": "valaki@gmail.com",  "password": "$2y$10$D6mCIvN2x9E7xSZc1CTpk.2VTMNrz6l.fjWl0ZARD0igIKsk8Otqa",  "id": 15  } |

|  |  |
| --- | --- |
| **/api/register** | |
| Kérés metódus | POST |
| Leírás | Regisztráció (mobil) |
| Input (JSON) | {  "name": "Új mobilfelhasználó",  "username": "ujVagyokMobil99",  "email": "valakiMobil@gmail.com",  "password": "Valaki99Valahonnan"  } |
| Output (JSON) | {  "admin": 0,  "name": "Új mobilfelhasználó",  "username": "ujVagyokMobil99",  "birthday": null,  "address": null,  "phone\_number": null,  "email": "valakiMobil@gmail.com",  "password": "$2y$10$bta4FlJE.ZCvNlJg9KBXkukRGaqX5BTWgUvUPqAbXAIxEPR0O3mAe",  "id": 16  } |

|  |  |
| --- | --- |
| **/api/dogAdoption/1** | |
| Kérés metódus | POST |
| Leírás | 1-es ID-val rendelkező kutya örökbefogadása |
| Input (JSON) | {  "adoption\_type\_id": 1,  "user\_id": 2  } |
| Output (JSON) | {  "adoption\_type\_id": 1,  "user\_id": 2,  "adoption\_beginning": "2022-04-10",  "id": 11  } |

# Asztali Alkalmazás – Kovács Krisztián

# 3. Fejlesztői dokumentáció

## 3.1 Alkalmazott fejlesztői eszközök

Már a munka megkezdése elött lényeges volt számomra, hogy meghatározzam milyen fejlesztő környezetet fogok használni, illetve milyen nyelven írjam meg az asztali alkalmazásomat. Viszont ez nem volt olyan egyszerű, mint ahogy azt hittem.

Eleinte úgy terveztem, hogy C# nyelven írom meg mivel a 2020-2021-es tanévben már megismerkedtünk ezzel a programnyelvvel. 2021.októberében elkezdtem a munkát a Visual Studio-ban és azon belül is a Windows Form keretrendszerben szerettem volna kialakítani a designt és megírni a kódot. Viszont időközben azt tanácsolták nekem, hogy inkább használjak más technológiát. Ezután megpróbálkoztam WPF-ben elkészíteni az alkalmazásomat viszont hamar kiderült, hogy nem ez a számomra megfelelő technológia a vizsgaremek elkészítéséhez.

Ezért hosszas gondolkodás után, a csapattal egyeztetve úgy döntöttem, hogy inkább egy JavaFX alkalmazást készítek mivel számomra sokkal megfoghatóbbnak tűnt a kezelő felülete, mint a Visual Studio-nak. Sokkal egyszerűbb, logikusabb tehát felhasználóbarátabb volt számomra, mellesleg nagyon tetszett az, hogy a stílus manipulációt egy CSS fájl segítségével tudom elvégezni, ami sokban megkönnyítette a munkámat, mivel ez nem sokban tért el az addigi CSS ismereteimtől, amit már HTML-nél elsajátítottam. Mellesleg ebben a tanévben csak Java óráim voltak, igy az ehhez kapcsolódó ismereteim is mélyebbek lettek, ezáltal talán könnyebb dolgom volt a vizsgamunka megírásával mintha C# nyelven írtam volna.

Ennek függvényében végül a választott fejlesztő környezet a JetBrains által készített IntelliJ IDEA 2021.2.4-es Ultimate verziója lett. Ahogy fentebb utaltam rá a program Java nyelven készült el.

Emellett mivel az IntelliJ-be beépített SceneBuilder nem a legmegbízhatóbb ezért a SceneBuilder szoftvert is használtam a munka során.

|  |  |  |
| --- | --- | --- |
| 6. ábra - IntelliJ IDEA | 7. ábra - Scene Builder | 8. ábra - Java programnyelv |

## 3.3 Részletes feladatspecifikáció, algoritmusok

### 3.3.1 Dogs Almappa

**Dog Osztály:**

Ez az osztály lényegében arra szolgál, hogy a backedben létrehozott adattagokat a Java alkalmazásban el tudjam tárolni. Ezért ebben az osztályban létrehoztam kilenc privát adattagot:

* private int id;
* private String name;
* private String gender;
* private String likely\_bday;
* private String species;
* private String external\_property;
* private String description;
* private int interest;
* private Integer adoption\_id;

Továbbá létrehoztam egy paraméteres konstruktort, amely feladata, hogy az osztályban található adattagokat példányosítja.

Illetve minden adattagnak létrehoztam egy ***Getter***-t és egy ***Setter***-t. Ez a két metódus típus a program működésének szempontjából elengedhetetlen, mivel a ***Getter*** segítségével tudom lekérdezni az adattagok értékét, illetve a ***Setter*** használatával tudom megváltoztatni az adott adattag értékét.

Az osztály utolsó metódusa egy ***Override*** típusú ***toString*** metódus, amelynek segítségével úgy tudom kiirtani az adatokat ahogy kedvem tartja. Tehát ez az adatok megemlítésében játszik szerepet.

**DogApi Osztály:**

Ez az osztály biztosítja a kapcsolat egy részét az adatbázissal. Annak érdekében, hogy a kapcsolat fennállásával ne legyen probléma egy ***private static final[[1]](#footnote-1)*** adattagot hoztam létre melynek a ***BASE\_URL*** nevet adtam, melynek neve is arról árulkodik, hogy ezen az URL címen lehet elérni az adatbázist. Továbbá létrehoztam egy szintén ***private static final*** adattagot, amely az ***API\_URL*** nevet viseli. Ez tárolja az API végpontot, amelyen el lehet érni a kívánt adattábla adatait, illetve mivel mindkét előbb említett adattag szöveg típusú ezért az utóbbiban össze is fűztem a két változó értékét. Erre azért volt szükség mert külön-külön nem értelmezhető a két URL és csak összefűzve érthetjük el a kívánt adattáblát.

A harmadik adattag ***Gson*** típusú, amely egy utólag hozzáadott könyvtárnak köszönhetően van jelen a programban. Ennek feladata az, hogy mivel az adatbázisból JSON formájában jönnek az adatok, ezeket a JSON formátumú adatokat fordítja le úgy, hogy azokat a program teljes mértékben fel tudja dolgozni és ezáltal tovább tud dolgozni a küldött adatokkal. Illetve ez fordítva is igaz mivel az adatokon végzett változtatásokat JSON formájában küldi vissza a program az adatbázisnak. Fogalmazhatunk úgy is, hogy ez „a fordító a két fél között”, hála ennek ez nagymértében megkönnyítette a munkámat!

Ebben az osztályban összesen 7 metódus kapott helyet, amelyek a következők:

*get():*

Ennek a metódusnak az a feladata, hogy lekérje és eltárolja az adatbázisból lekért tábla adatait egy listában ezért ez a metódus egy olyan listát vár el, amely csak *Dog* objektumokat tartalmaz.

*post():*

Ez a metódus azt a célt szolgálja, hogy képesek legyünk új adatot rögzíteni az adatbázisban. Ennek érdekében ez a metódus rendelkezik egy paraméterrel, amely egy *Dog* objektumot vár el, illetve visszatérési értékként szintén egy *Dog* objektumot vár el.

*delete():*

Ennek a metódusnak a segítségével tudunk rekordokat törölni az adatbázisból. Annak érdekében, hogy ez megfelelően működjön a metódus rendelkezik egy paraméterrel, amely a törölni kívánt adatsor azonosítóját várja el, még pedig azért, hogy biztosan azt adatot töröljük, amelyet kiválasztottunk.

*put():*

A ***post()*** metódushoz hasonlóan ez is ugyan olyan paraméterrel és visszatérési értékkel rendelkezik, viszont a funkciója már eltér. Ez a metódus azt a célt szolgálja, hogy a meglévő adatokat módosítani tudjuk.

Az osztályban található további 3 metódus lényegében statisztikai metódusok, amelyek a *StatisticController* működését segítik:

*getDogCount():*

Visszaadja, hogy összesen hány kutya van az adatbázisban

*getAdoptedDogCount()*:

Meghatározza, hogy hány olyan kutya van az adatbázisban, akiket már örökbefogadottak

*getNotAdoptedDogCount*():

Visszaadja, hogy hány olyan kutya van, akiket még nem fogadtak örökbe

**DogController:**

Ez az osztály kapcsolatban áll a hozzátartozó ***FXML*** fájllal, jelen esetben a dogs-view-val. Azért van szükség a két fájl összekötésére mivel az ***FXML*** fájlokban eltárolt felületi tervek önmagukban nem képesek semmilyen funkció végrehajtására, ezért szüksége van egy külön osztályra, amelyben meghatározzuk, hogy például: az adott gomb hajtsa végre a törlést.

Az osztályban találhatók privát adattagok, amelyeket az ***FXML*** fájl ad át a DogControllernek az előre beállított adattag azonosítók alapján, például: ***TableColumn<Dog, String> szulIdoCol;***

A ***<Dog, String>*** azt takarja, hogy az adott oszlop egy Dog objektumot vár el, illetve, hogy ennek az objektumnak szöveg formátumúnak kell lennie.

Metódusok:

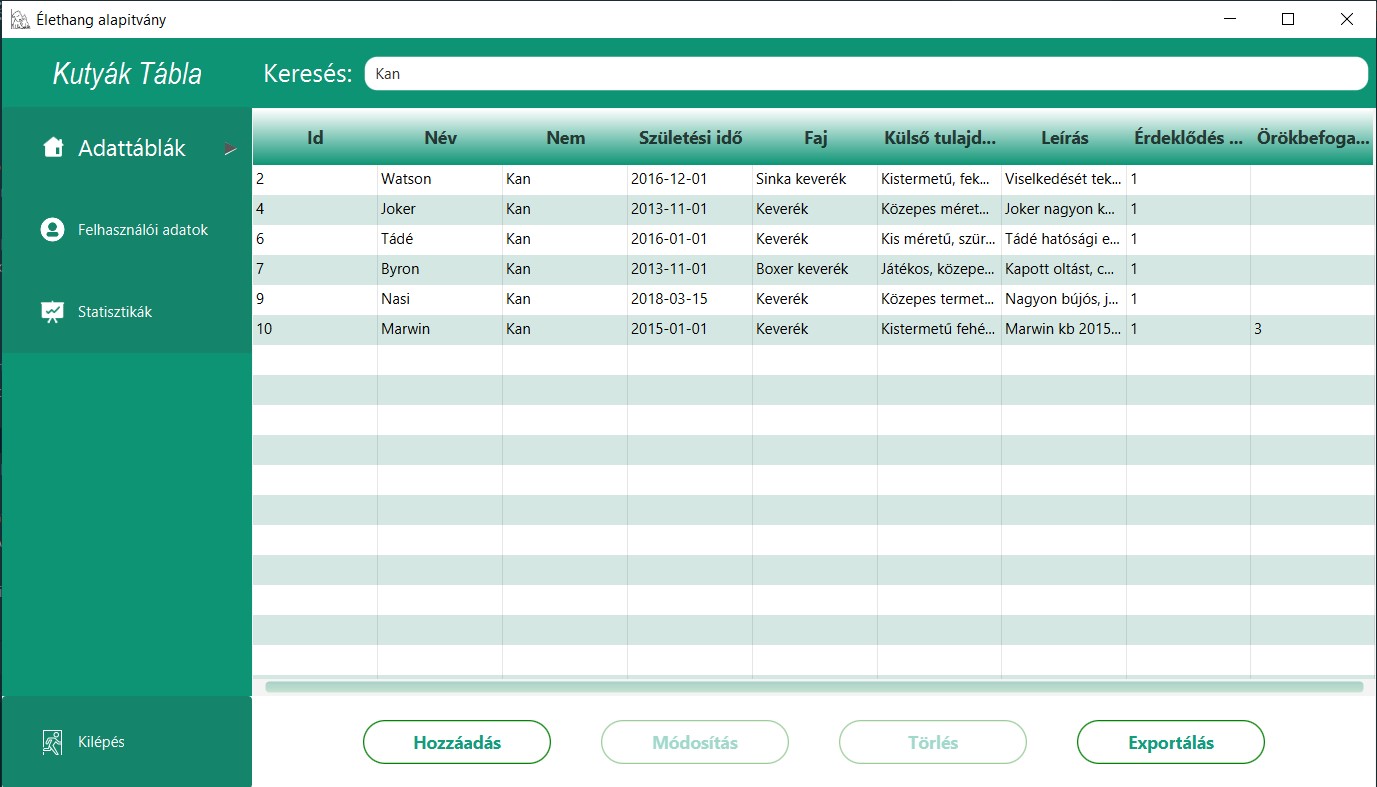
*initialize():*

Beállítja a táblázat megfelelő celláiba az értékeket. Ebben segítenek, a *Dog Osztályban* létrehozott ***Getter***-ek, amelyek biztosítják, hogy minden a megfelelő helyen kerüljön. Továbbá a metódusban meg lettek hívva a következő függvények: ***kutyakListaFeltolt()***, ***kereses()***. Ez azért van mert ezeknek a metódusoknak a funkcióira is szükség van az oldal betöltésekor, annak érdekében, hogy minden alap funkció elérhető legyen a felhasználónak.

*kutyakListaFeltolt():*

Ez a metódus felel azért, hogy a táblázatot feltöltse az adatokkal, így azok láthatóvá válnak a felhasználó számára.

*kereses():*

Ennek segítségével tud a felhasználó szűrni a megjelenített táblázatban. Például, ha beírja a **Kan** szót akkor ahogy a képen is látszódik csak azokat a kutyákat jeleníti meg amelyeknek a kan neme van.

9. ábra - Szűrés

*onHozzaadKutya():*

Meghívja és megjeleníti a ***hozzaad-view.fxml***-t és a hozzá tartozó ***DogHozzaadController*** segítségével a felhasználó képes új rekordot hozzáadni az adatbázishoz. (Az előbb említett DogHozzaadController lentebb részletesebben is ki van fejtve).

*onModositKutya():*

Meghívja és megjeleníti a ***modosit-view.fxml***-t és a hozzá ***tartozó DogModositController*** segítségével a felhasználó képes módosítani a meglévő adatokat. (Az előbb említett DogModositController lentebb részletesebben is ki van fejtve).

*onKutyaTorol():*

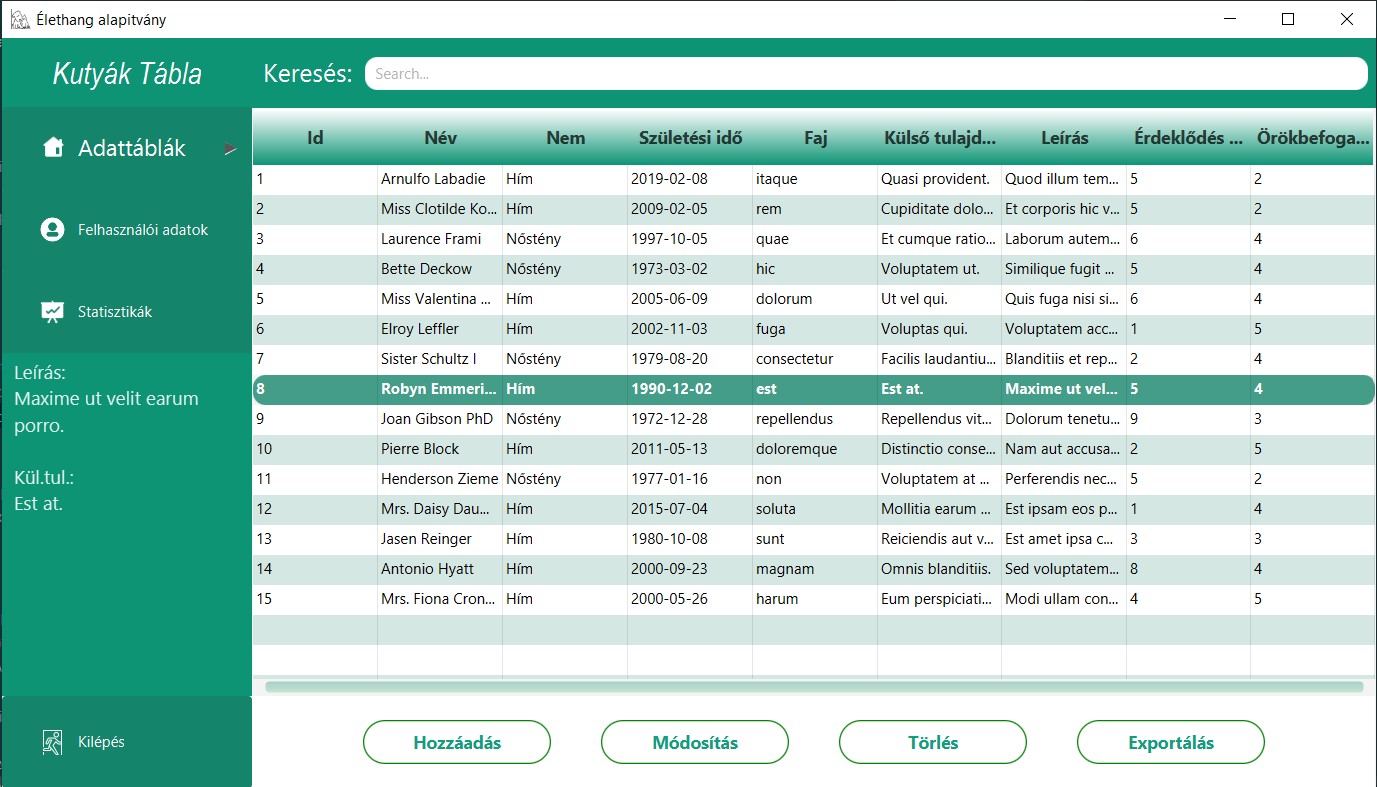
Ennek segítségével a felhasználó képes törölni adatokat az adatbázisból.

*onExportKutyakTabla():*

Ez a metódus azt a feladatot látja el, hogyha a felhasználó rányom az Exportálás gombra akkor az adott tábla adatait kimenti egy Microsoft Excel dokumentumba, viszont mielőtt ez megtörténik a felhasználónak ki kell választani a helyet, ahova el szeretné menteni a fájlt, illetve el kell neveznie ezt dokumentumot.

Ezt úgy tudtam megvalósítani, hogy egy külső könyvtárat importáltam a programba (Apache Poi). Ennek a könyvtárnak a függvényeivel pedig meg tudtam valósitani az előbb leírt funkciót.

*onSelectDog():*

A képen asztal látható

Automatikusan generált leírásAmikor a felhasználó kiválaszt egy adatsort csak akkor válnak elérhetővé a Törlés és a Módosítás gombok, illetve azok az adatok, amik nem teljes látszanak a cellába, az oldal bal oldalán jelennek meg annak érdekében, hogy minden adatot el lehessen olvasni.

10. ábra - Kiválasztás után

11. ábra - Kiválasztás elött

*onMacskakClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***cats-view.fxml***-t.

*onKutyakClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***dogs-view.fxml***-t.

*onProgramApplicationClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***programApplications-view.fxml***-t.

*onEventClick():*

Bezárja a jelenlegi ablakot és megjeleníti az ***events-view.fxml***-t.

*onAdoptionTypeClick():*

Bezárja a jelenlegi ablakot és megjeleníti az ***adoptionTypes-view.fxml***-t.

*onProgramInfoClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***programInfo-view.fxml***-t.

*onAdoptionClick():*

Bezárja a jelenlegi ablakot és megjeleníti az ***adoptions-view.fxml***-t.

*onUsersClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***users-view.fxml***-t.

*onUserDataClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***userdata-view.fxml***-t.

*onStatisticClick():*

Bezárja a jelenlegi ablakot és megjeleníti a ***statistic-view.fxml***-t.

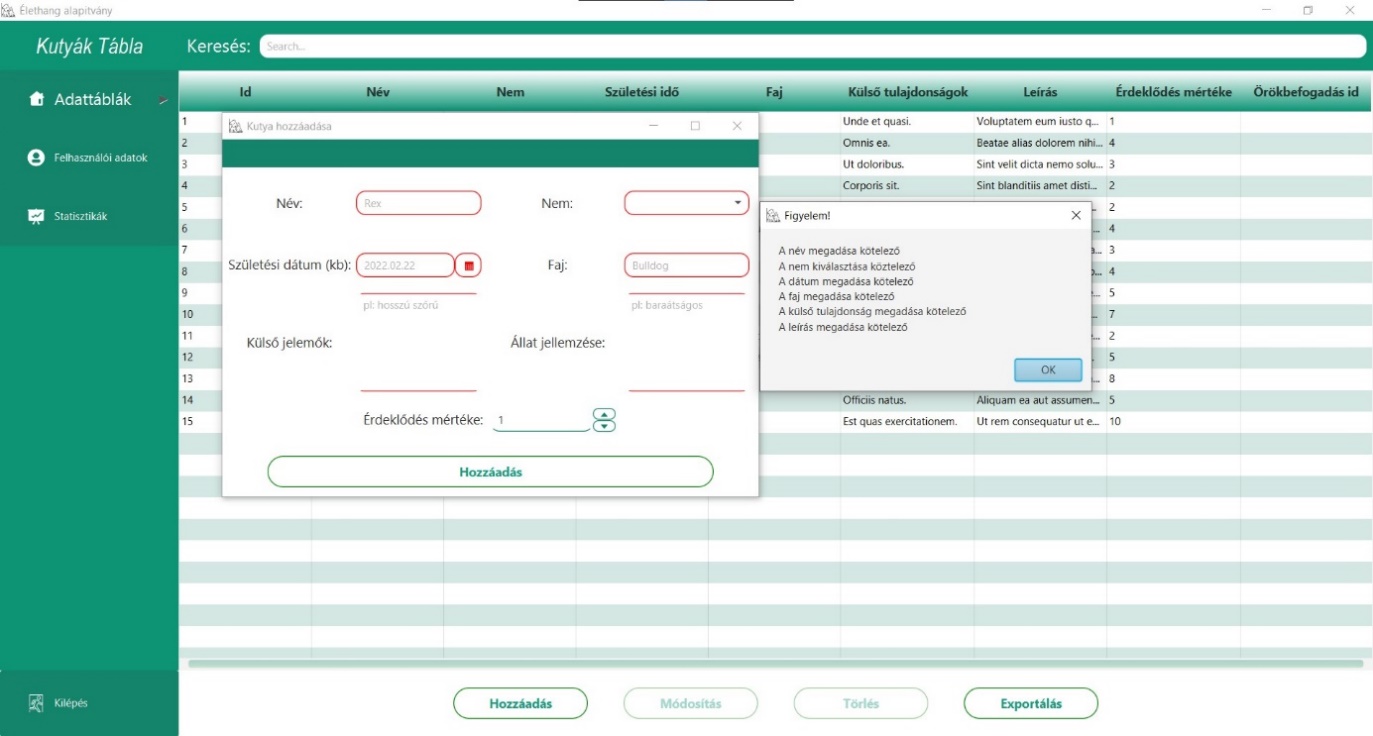
*onExit():*

Bezárja a jelenlegi ablakot és megjeleníti a ***login-view.fxml***-t.

**DogHozzaadController:**

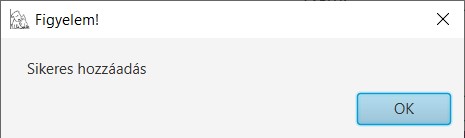
Hasonlóan, mint a *DogController*, ez az osztály is kapcsolatban áll egy ***FXML***fájllal (***hozzaad-view.fxml***), igy itt is találhatók privát adattagok, amelyeket az ***FXML*** fájl követel meg.

*onHozzaadas():*

Ebben a metódusban változók segítségével eltárolásra kerülnek a bemeneteli mezőbe bevitt adatok. Továbbá a program le ellenőrzi, hogy a megadott adatok megfelelnek-e az elvártaknak, igy azt is ellenőrzi, hogy üres-e a mező és ha olyan mezőt talál, amely üres vagy nem felel meg a követelményeknek akkor egy felrúgó ablakban figyelmezteti a felhasználót egy hibaüzenettel, illetve ilyenkor a beviteli mező kinézete is megváltozik a ***FelugroAblakHiba.css***-ben megadott változtatásoknak köszönhetően, igy egyértelmű lesz a felhasználónak, hogy pontosan melyik mezőben vár el javítást a program.

12. ábra - Hibaüzenet

Ha minden mező megfelelő adatokat tartalmaz akkor a program végrehajtja a hozzáadást és egy felugró ablakban tájékoztatja, hogy sikeres volt-e vagy nem az adat rögzítés.



13. ábra - Visszajelzés

*hibaVege():*

Megszünteti a ***FelugroAblakHiba.css***-ben deklarált kinézeti változtatásokat és visszaállítja a ***FelugroAblak.css***-ben meghatározott kinézetűre a felületet.

Ez a metódus a következő bemeneti mezőknél végzi el az előbb ismertetett változtatásokat: ***TextField*, *TextArea****.*

*hibakMegszuntet():*

Ennek a metódusnak ugyan az a feladata, mint ***hibaVege()*** metódusnak viszont ez a következő bemeneti mezőknél végzi el a változtatásokat: ***DatePicker, ChoiceBox****.*

**DogModositController:**

Ez lényegében ugyan azokat metódusokat tartalmazza, illetve ugyan azokat a funkciókat tudja, viszont apró eltérések azért akadnak, amelyek a következők:

Az osztályhoz tartozó ***FXML*** fájl ebben az esetben a ***modosit-view.fxml****.*

Ez a felület viszont csak akkor válik elérhetővé, ha a felhasználó kijelöl egy adatsort. Ez azért van igy mert a kiválasztott adatsor adatait a program visszatölti a bemeneteli mezőkbe annak érdekében, hogy a felhasználó lássa mi volt az eredeti adat. Továbbá ezeket az adatokat képes szerkeszteni és ha rányom a Módosítás gombra akkor ezeket a változtatásokat el is tudja tárolni, amiről a felhasználó vissza jelzést is kap, hogy sikeres vagy sikertelen volt-e a művelet. A hiba ellenőrzés itt is megtalálható ugyan olyan formában, mint ahogy *11.* és *12. ábra* is szemlélteti.

### 3.3.2 Api Almappa

Ez a mappa tartalmazza azokat az osztályokat, amelyek segítenek létrehozni a kapcsolatot a szerverrel, illetve ennek a kapcsolatnak a megfelelő fenntartásában és kezelésében segítenek, ezért rendkívül fontosok az alkalmazás működésének szempontjából.

**Api Osztály:**

A *get(), post(),* *delete(), put()* metódusok a *DogApi Osztály*-ban is megtalálható metódusok kiegészítései, mivel itt található meg ezeknek a metódusoknak a részletekbe menő kódja, mivel pontosan lekódoltam, hogy mit is várok el a metódustól nem fordulhat elő olyan, hogy a program nem tudja feldolgozni vagy félreérti a kért metódust és annak feladatát.

*getLogin():*

Ez a metódus a felhasználó bejelentkezésében játszik szerepet.

A metódus két paramétert vár el, egy URL-t, amelynek tartalmaznia kell a megfelelő API végpontot is, illetve egy bejelentkezési tokent.

**ApiError Osztály:**

Hiba esetén ez az osztály felel azért, hogy a megfelelő hibaüzenettel es ehhez kapcsolódóan a megfelelő státusz kóddal tájékoztassa az adminisztrátort.

Két adattag található itt, amelyek a következők:

* private String message;
* private int statusCode;

A *statusCode* felel azért, hogy eltárolja a hibához tartozó státuszkódot és ezért nem véletlen ez az adattag egy szám típus.

A *message* egy szöveg típusú adattag, amely arra szolgál, hogy eltárolja a hibához tartozó üzentet.

*ApiError():*

Ennek a konstruktornak a segítségével lehet létrehozni egy újabb hiba visszajelzést egy kéréshez. Annak érdekében, hogy megfelelően működjön ez a metódus két paramétert vár el: egy *statusCode* -t és egy *message* -t.

*getStatusCode():*

Segítségével megkaphatjuk a hiba kódját

*getMessage():*

Segítségével megkaphatjuk a hiba üzenetet.

**RequestHandler Osztály:**

Ez az osztály és az ebben található metódusok felel azért, hogy az egyes funkciókhoz kapcsolódó HTTP kéréseket megfelelően kezelje a program. Ezek nélkül nem jutna el megfelelően a szerver felé a kérés, illetve a válasz sem érkezne meg a felhasználó felé.

**Response Osztály:**

Ez osztály tárolja az egyes funkciókhoz tartozó kérések viszont válaszát.

Két adattag található itt, amelyek a következők:

* int responseCode;
* String content;

A *responseCode* felel azért, hogy eltárolja a válaszhoz tartozó státuszkódot és ezért nem véletlen ez az adattag egy szám típus.

A *content* egy szöveg típusú adattag, amely arra szolgál, hogy eltárolja a válaszhoz tartozó üzentet.

*Response():*

Ennek a konstruktornak a segítségével lehet létrehozni egy újabb választ egy kéréshez. Annak érdekében, hogy megfelelően működjön ez a metódus két paramétert vár el: egy *responseCode*-t és egy *content*-t.

*getResponseCode():*

Segítségével megkaphatjuk a válasz kódját

*getContent():*

Segítségével megkaphatjuk a válasz üzenetet.

### 3.3.3 ÉlethangApp Osztály

Ez egy olyan osztály, amelyet az IntelliJ magától legenerált amikor létrehoztam a projektet és már csak ezért is nagy jelentőségű, továbbá azért, mert ez az osztály felel azért, hogy milyen felület jelenjen meg amikor a felhasználó elindítja az alkalmazást. Jelen esetben a bejelentkezési felület lesz az első, amely meg fog jelenni a felhasználónak.

### 3.3.4 Login Almappa

**Login Osztály:**

Ebben az osztályban két privát adattag található melyek közül az egyik a felhasználó név míg másik a jelszó tárolására szolgál.

* private String username;
* private String password;

Ehhez tartozik egy paraméteres konstruktor, amely az előbb említett két adattagot várja el.

**LoginApi Osztály:**

*postLogin():*

Létrehozza a bejelentkezéshez szükséges tokent annak a felhasználónak, aki éppen be szeretne jelentkezni.

*getLoginData():*

Az előbb létrehozott token alapján a program ellenőrzi, hogy egyáltalán a felhasználó adatai szerepelnek-e az adatbázisban.

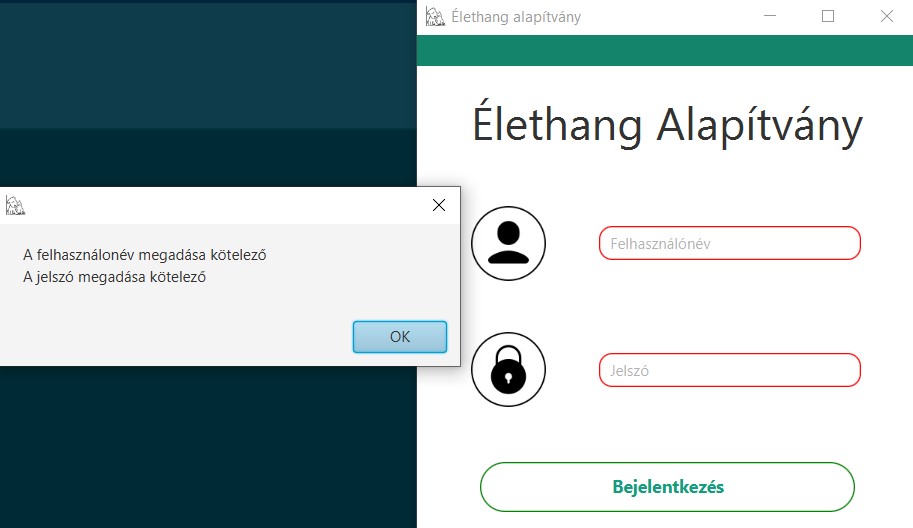
**LoginController Osztály:**

Az osztály össze van kötve egy ***FXML*** fájllal *(****login-view.fxml****)*. Ennek köszönhetően találhatók az osztályban olyan privát adattagok, amelyeket az ***FXML*** fájl követel meg.

*onBejelentkezes():*

Ebben a metódusnak a változóiban mentem el a bemeneteli mezőbe bevitt adatok. Továbbá a program ellenőrzi, azt is, hogy a megadok adatok minden tekintetben megfelelnek-e, ha bármilyen hibát észlel a program akkor egy felrúgó ablakban figyelmezteti a felhasználót egy hibaüzenettel, illetve ilyenkor a beviteli mező kinézete is piros színűre vált igy egyértelmű lesz a felhasználónak, hogy pontosan melyik mezőben szükséges a javítás.

Ha minden rendben akkor a program még elvégez egy gyors háttér ellenőrzést, hogy felhasználó rendelkezik-e a megfelelő jogosultsággal ahhoz, hogy használja az alkalmazást. Abban az esetben, ha nem rendelkezik a szükséges jogosultsággal akkor egy hibaüzenetben tájékoztatja a felhasználót erről, ellenkező esetben az alkalmazás tovább engedi a felhasználót a jogosultságának megfelelő oldalra, igy megengedi, hogy hozzáférjen az adattáblákhoz.

*hibaVege():*

14. ábra - Bejelentkezési hiba

Megszünteti a ***FelugroAblakHiba.css***-ben deklarált kinézeti változtatásokat és visszaállítja a ***FelugroAblak.css***-ben meghatározott kinézetűre a felületet.

**Token Osztály:**

Ebben az osztályban egy privát adattag található, amely a bejelentkezéshez szükséges *token* tárolására szolgál.

* private String token;

Továbbá egy paraméteres konstruktor és egy ***Getter*** metódus is található az osztályban.

### 3.3.5 SuperAdmin Almappa

**SuperAdminUserController:**

Ez osztály a super admin jogosultsággal rendelkező felhasználók kezelőfelületének funkcióit írja le. Annak érdekében, hogy mindent megfelelően megjelenjen minden a super admin számára, ez osztály össze van kötve egy ***FXML*** fájlal (***superAdminUsers-view.fxml***). Ennek megfelelően az itt létrehozott privát adattagok nagy részét az FXML fájl követeli meg, ezért ezek a működés szempontjából nagyon fontosak.

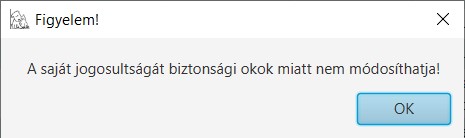
Ezen a felületen a super admin képes új admint hozzáadni az adatbázishoz, felhasználók adatait is tudja törölni, illetve exportálni is tudja az adattáblát egy Excel fájlba. Továbbá lehetősége van a felhasználók jogosultságát módosítani.

Ennek az osztálynak is vannak metódusai, amelyek leírják, hogy az adott függvénynek mi a feladata. Mivel vannak olyan függvények, amelyek lényegi működését már leírtam a **DogsAlmappa->DogsController**-ben ezért csak pár eddig nem említett metódust fogok ismertetni.

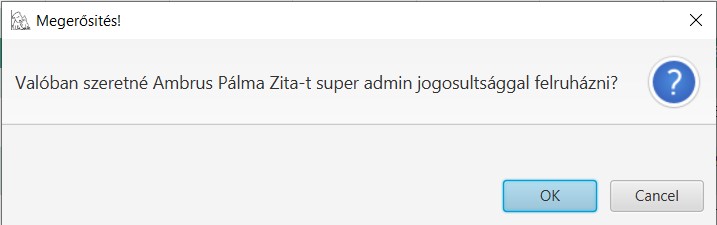
*onAdminJog():*

Ennek a metódusnak a segítségével lehetséges egy általános felhasználót admin jogosultsággal felruházni. Illetve az admin jogosultsággal rendelkező felhasználót super admin jogosultsággal felruházni.

A függvény működéséhez ki kell választanunk egy felhasználót különben nem tudjuk végrehajtani a műveletet. Ezt úgy kezeltem le, hogy amig nincs kiválasztva egy felhasználói rekord addig inaktív a *Feljogosít* gomb.

Továbbá azt is lekezeltem, hogy a bejelentkezett felhasználó ne tudja módosítani a saját jogosultságát.

15. ábra - Saját jogosultság módosításakor felrúgó üzenet

Amennyiben egy másik felhasználónak szeretné módosítani a super admin a jogosultságát a program egy felugró ablakban egy megerősítést vár, hogy biztos végre szeretné-e hajtani a műveletet.

16. ábra - Jogosultság módosítás megerősítése

*onLefokoz():*

Ezzel a metódussal képes a super admin más super adminok jogosultságát megváltoztatni, admin jogosultságra, illetve az admin jogosultsággal rendelkező felhasználókat lefokozni általános felhasználói jogosultsági szintre.

Működésben ugyan az tudja, mint a *onAdminJog()* annyi különbséggel, hogy a hibaüzenetek szövegei eltérnek .

### 3.3.6 Controller Osztály

Ez egy absztrakt osztály, amelyre azért volt szükség, hogy elkerüljem a fölösleges ismétlődő kódsorokat, mivel, ha egy osztálynak beállítom kiterjesztésként a *Controller Osztály*-takkor abban osztályban elérhetővé válnak ennek az osztálynak ametódusai.

*ujAblak():*

Ebben az osztályban írtam meg azokat alapvető dolgokat, amelyek szükségesek az oldalváltásokhoz. Bármely másik osztályban, ahol ablakváltást szeretnék megvalósítani ott elég meghívnom ez a függvényt.

*hibaKiir():*

Erre a metódusra akkor van szükség, ha szeretnénk, hogy a program ne csak egy hibaüzenet adjon vissza, hanem írja is ki mi okozta a hibát a rendszerben, illetve az ehhez tartozó hiba kódot is írja ki a program. Ez elsősorban a fejlesztőknek van nagy hasznára.

Tájékozgató jellegű felugró ablakok:

A további metódusoknak lényegében ugyan az a feladata: tájékoztassák a felhasználót, hogy a program mit vár el vagy hogy milyen esemény hajtódott végre.

Ezek a metódusok pedig a következők:

* *confirm()*
* *alert()*
* *alertBejelentkezés()*
* *alertinput()*
* *alerthiba()*
* *alertWait()*

### 3.3.7 StatisticController Osztály

Ez az osztály a statisztikai diagrammok megjelenítésére szolgál, illetve annak érdekében, hogy ez a felhasználó számára is látható legyen ez az osztály össze van kötve a ***statistic-view.fxml*** fájlal.

*statisztikaMegjelenit():*

Ez a metódus tartalmazza azt, hogy hogyan, és milyen adatok alapján jelenjenek meg a diagrammok a felületen.

Továbbá itt is megtalálhatóak azok a metódusok, amelyek arra szolgálnak, hogy a felhasználó át tudjon menni egy másik felületre. Ezek ismertetése megtalálható a ***Dogs Almappa -> DogController*** leírásában.

## 3.4 Tesztelés

|  |  |  |  |
| --- | --- | --- | --- |
| Teszteset neve | Bemenet | Elvárt eredmény | Valós eredmény |
| Bejelentkezés | Felhasználónév: Kpal10  Jelszó: kalapal10 | Sikeres bejelentkezés | Sikertelen bejelentkezés  (nem rendelkezik jogosultsággal) |
| Bejelentkezés | Felhasználónév: Kkrisz08  Jelszó: kiki1234 | Sikeres bejelentkezés | Sikeres bejelentkezés |
| Rendezvény Hozzáadás | Rendezvény neve: Postás Juniális,  Leírás: Bemutathattuk az alapítványunkat és gyűjthettünk védenceink számára.   Dátum: | Sikeres hozzáadás | Hibaüzenet:  A dátum megadása kötelező |
| Rendezvény Hozzáadás | Rendezvény neve: Postás Juniális,  Leírás: Bemutathattuk az alapítványunkat és gyűjthettünk védenceink számára.   Dátum: 2019-06-22 | Sikeres hozzáadás | Sikeres hozzáadás |
| Szűrés a kutyák táblában | Kulcsszó: szuka | Csak azoknak az állatoknak az adatai jelennek meg amelyek szuka neműek | Csak azoknak az állatoknak az adatai jelennek meg amelyek szuka neműek |
| Örökbefogadási típus törlése | Rányomunk az egyik adatsorra  utána rányomunk a törlés gombra | Sikeres törlés  adatsor eltűnik a táblázatból | Sikeres törlés  adatsor eltűnik a táblázatból |
| Felhasználók exportálása | Rákattintunk a Exportlás gombra | Kiválasztjuk, hogy hova szeretnénk menteni a fájlt    rányomunk a mentés gombra  Sikeres exportlás | nem történik semmi mert nem nevezetük el a fájlunkat  Sikertelen exportálás |
| Felhasználók exportálása | Rákattintunk a Exportlás gombra | Kiválasztjuk, hogy hova szeretnénk menteni a fájlt  elnevezzük a fájt  rányomunk a mentés gombra  Sikeres exportlás | Sikeres exportálás |
| Statisztika | Rákattintunk a Statisztikák gombra | Az alkalmazás átlép a statisztikákat megjelenítő oldalra | Az alkalmazás átlép a statisztikákat megjelenítő oldalra |
| Kijelentkezés | Rákattintunk a Kilépés gombra | Az alkalmazás kijelentkezik és átlép a bejelentkezési oldalra | Az alkalmazás kijelentkezik és átlép a bejelentkezési oldalra |

## 3.5 Továbbfejlesztési lehetőségek

* Oldalváltásokat kiszervezni egy külön osztályba. Erre azért lenne szükség mert igy optimalizálhatnám a programot.
* Ne „fagyjon” be az alkalmazás bejelentéskor és a statisztika megjelenítéskor. Mivel a backend segítségével hívja le az adatokat ezeknél az említett pontoknál egy kisebb várakozási idő lép fel és ilyenkor olyan mintha be lenne fagyva az alkalmazás mert nem lehet semmi másra rányomni amig ezek a folyamatok végbe nem mennek.
* A táblákban ne azonosítók jelenjenek meg hanem az azonosítókhoz tartozó értékek. Ez az adattábla átláthatóságát, egyértelműsítését segítené elő, illetve a felhasználó gyorsabb munkáját, mivel igy most ki kell keresnie, hogy az adott azonosító milyen adat(okat) takar.
* Oldalváltáskor tartsa meg és állítsa be az átirányított oldal méretét az előző oldal méretére.
* Kép feltöltése a felhasználói adat részben.
* A diagrammokon jelenjenek meg az értékek.

# 4. Felhasználói dokumentáció

## 4.1 A program általános specifikációja

Az asztali alkalmazás lényegében egy adminisztrátori felület, melynek feladata az, hogy segítse nyilván tartani az Élethang Alapítvány állatállományát, továbbá, hogy megjelenítse azt, hogy milyen fontosabb események történtek az alapítvány életében.

Ezek mellett az adminisztrátor képes manipulálni az adatbázist, amelyben az adatok vannak, mégpedig úgy, hogy képes hozzáadni új adatsorokat, itt érdemes megemlíteni, hogy szintén ezen a felületen lesz lehetőség örökbefogadásokat rögzíteni, illetve a meglévő és frissen létrehozott adatokat módosítani és törölni is tudja. Továbbá ezeket az adattáblákat ki tudja exportálni egy Excel táblázatba, ami az irodai munkát kívánja megkönnyíteni, mivel ennek segítségével, kimutatásokat, látványos táblázatokat, diagrammokat tud készíteni az egyes adattáblákból.

## 4.2 Rendszerkövetelmények

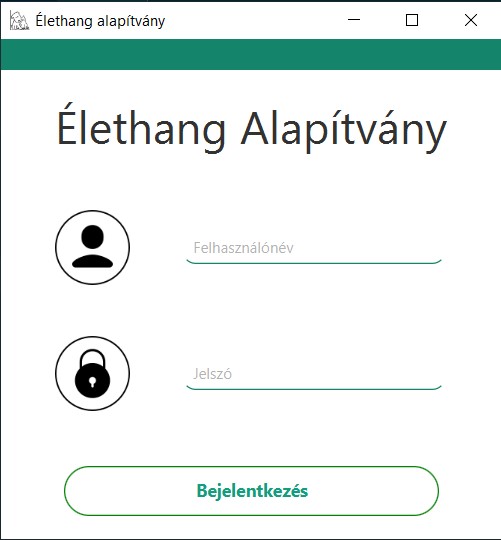
Ahhoz, hogy az alkalmazás probléma nélkül fusson, a következő feltételeknek mindenképp eleget kell tennie a gépnek, amelyre telepíteni szeretnénk a programot:

* Windows 10
* 8GBRAM memória
* 2.50 GHZ processzor
* Videokártya, ami támogatja a (1080p) képernyő felbontást
* Internetkapcsolat

## 4.3 A program telepítésének és konfigurálásának a leírása

A telepítő program (*elethangsetup.exe*) futtatásával lehetséges, ezt követően az alkalmazás azonnal futtatható.

## 4.4 A program használatának a részletes leírása

Sikeres telepítés után, ha a felhasználó megnyitja az alkalmazást a bejelentkezési felület jelenik meg először. Itt meg kell adni a felhasználónevet, illetve a jelszavát. Abban az az esetben, ha hibás adatokat adott meg a felhasználó vagy üresen hagyott egy mezőt, akkor a program tájékoztatja a felhasználót, hogy miért nem sikerült a bejelentkezés. Továbbá, ha jó adatokat adott meg viszont nem rendelkezik *Admin* vagy S*uper Admin* jogosultsággal, akkor a rendszer nem fogja beengedni az alkalmazásba!

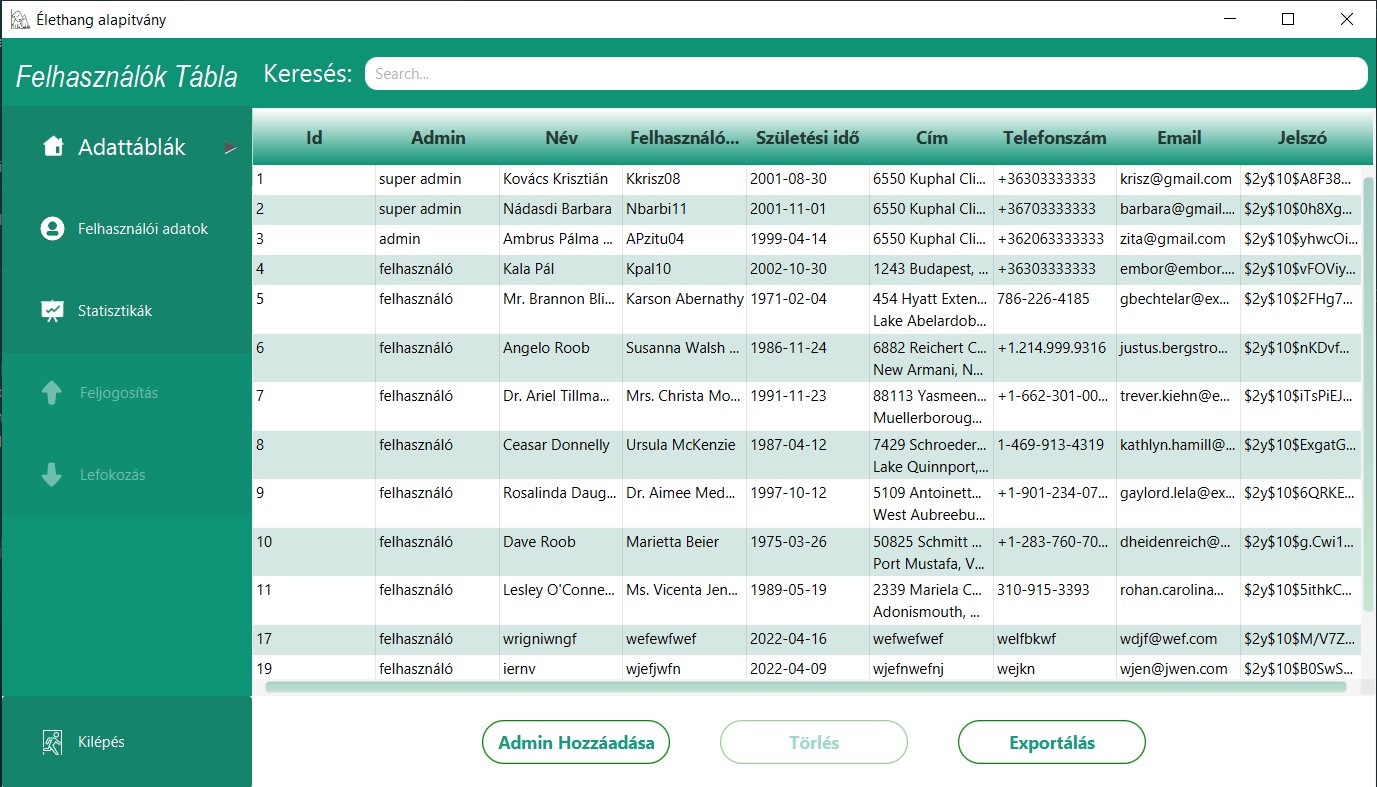
17. ábra - Bejelentkezési felület

Ha a felhasználó adatai minden esetben megfelelnek a bejelentkezés feltételeinek akkor a felhasználó hozzáfér az adattáblákhoz, illetve közvetlen a bejelentkezés után a felhasználók adattáblája jelenik meg előtte. Viszont fontos megemlíteni, hogy más-más ablakok jelennek meg annak függvényében, hogy a bejelentkezett felhasználó *Super Admin* vagy *Admin* jogosultsággal rendelkezik.

*Admin* jogosultság esetén a felhasználói táblából az adatokat csak exportálni tudja az admin, más felhasználók adatait nem tudja változtatni.

18. ábra - Admin felület belépés után

**

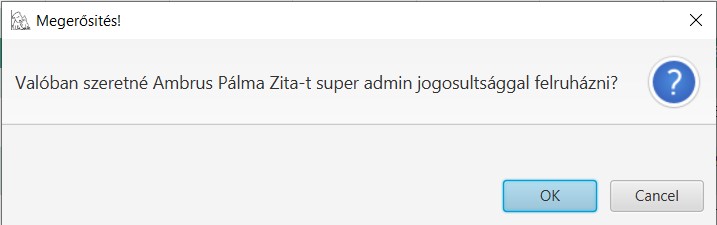
*Super Admin* jogosultság esetén az exportáláson túl a super admin képes más felhasználók jogosultságát módosítani (a sajátját nem tudja!), továbbá lehetőség van törölni felhasználói adatokat, illetve hozzáadni új admin felhasználót. Sima felhasználót azért nem lehet hozzáadni mert az átlag felhasználók a weboldalon es a mobil alkalmazásban tudnak majd regisztrálni, illetve belépni a fiókjukba.

19. ábra - jogosultság módosító gombok elérhetőek

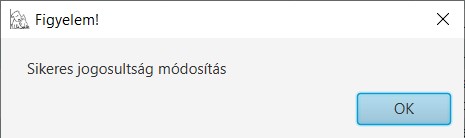
20. ábra - Super Admin felület belépés után

A mennyiben módosítani szeretnénk egy felhasznál jogosultságát, először ki kell választani egy adatsor és után elérhetővé válna a *Feljogosít* és *Lefokoz* gombok.

A *Feljogisít* gomb megnyomásával eggyel magasabb jogosultsági szintet adhatunk a felhasználónak, illetve a *Lefokoz* ennek ellentétét csinálja, tehát eggyel alacsonyabb jogosultsági szintet tudunk a kiválasztott felhasználónak adni.

Mindkét esetben egy megerősítő ablak jelenik meg hogy biztos szeretnénk-e végig csinálni a műveletet.

21. ábra - módosítás megerősítése

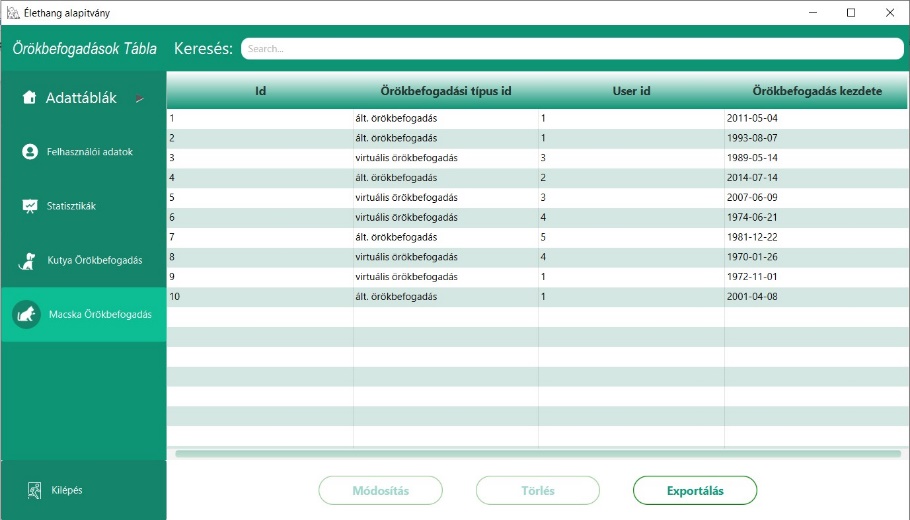
Illetve vissza jelzést is kap a felhasználó, hogy sikeres volt e művelet vagy nem.

22. ábra - művelet sikeressége

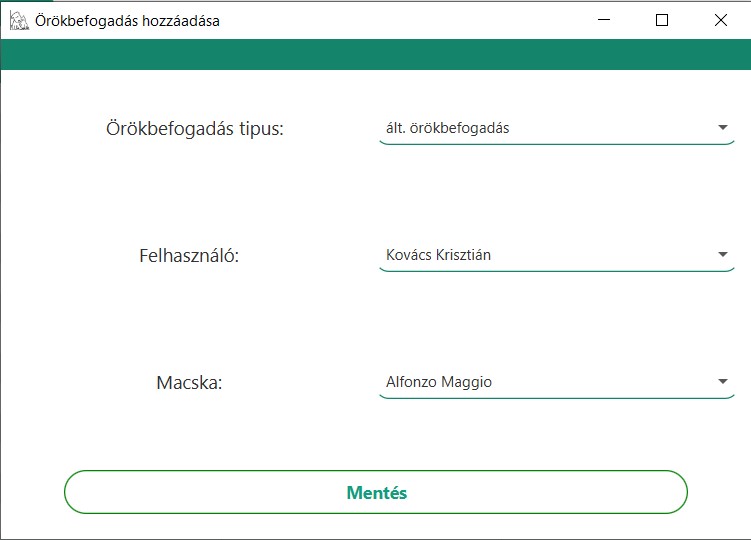
Vannak funkciók, amelyek a legtöbb oldalon elérhetők ilyen például az adat létrehozása, annak módosítása, illetve törlése, továbbá az exportálás Excelbe. Érdemes még itt megemlíteni a keresési mezőt, amely a lap tetején kapott helyet. Ennek segítségével tud a felhasználó kulcsszavak segítségével szűrni az adattáblában.

Ha felhasználó szeretne egy másik oldalra átlépni, arra is van lehetőség, még pedig úgy, hogy választ egy számára tetszőleges gombot melynek felirata megegyezik az elérni kívánt oldal nevével.

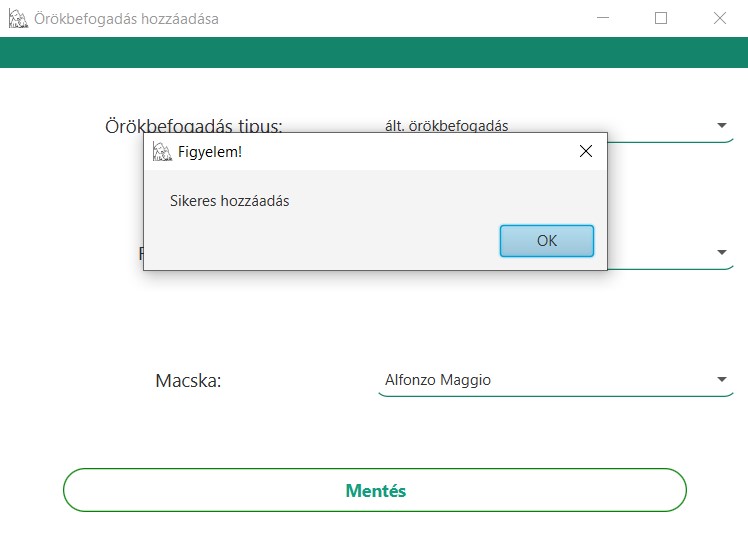
Abban az esetben, ha az örökbefogadás táblát választja a felhasználó, akkor az alap funkciókon túl , képes örökbefogadásokat rögzíteni. Ezt úgy tudja megtenni, hogy rákattint a *Kutya örökbefogadására* vagy *Macska örökbefogadására* (értelemszerűen a gomb neve alapján az adott fajú állatot van mód örökbe fogadni).



23. ábra - Macska Örökbefogadás kiválasztása

Miután a felhasználó kiválasztotta milyen állatnak szeretné rögzíteni az örökbefogadási adatait, egy adatlap jelenik meg amelyben ki kell választani az örökbefogadás típusát, az örökbefogadót, illetve az örökbe fogadni kívánt állat nevét.

24. ábra - Örökbefogadási adatlap

Végezetül a Mentés gombra nyomva a program eltárolja az adatokat, amelyről tájékoztatja a felhasználót, hogy sikeres vagy sikertelen volt-e a művelet.

25. ábra - Örökbefogadási visszajelzés

Számos más opció közül választhat még a felhasználó:

* az *Adattáblák* gombra kattintva ki tudja választani, hogy melyik táblába szeretne tevékenykedni.
* A *Felhasználói adatok* gombra nyomva megjelennek a bejelentkezett felhasználó saját adatai, illetve ezeket módosítani is tudja, ha szeretné
* A Statisztikák gombra kattintva megjelennek azok a diagrammok, amely az adatbázis adati alapján készültek, annak érdekében, hogy egy vizuális képpel is szemléltesse az adatok arányát.
* A képen szöveg látható

  Automatikusan generált leírásIlletve választhatja a kijelentkezést is, amelyet úgy tud végrehajtani, ha a *Kijelentkezés* gombra kattint.

26. ábra - Oldalváltási lehetőségek

# 5. Összegzés

A projekt kezdetekor fogalmam sem volt, hogy hogyan fogok majd egy olyan alkalmazást megcsinálni, amely minden ötletemet tartalmazza. A kezdeti bizonytalansággal szemben most teljes büszkeséggel el tudom mondani, hogy sikerült egy olyan alkalmazást alkotnom, amellyel teljes mértékig meg vagyok elégedve!

Továbbá nem csak a kezdeti ötleteimet tartalmazza, hanem azokat a funkciókat is sikerült implementálnom, amelyeket a fejlesztés közben találtam ki. Ilyen ötlet volt az adatok exportálása egy Excel táblába, vagy a statisztikai diagrammokat megjelenítő oldal. Viszont voltak olyan funkciók is, amelyeket át kellett kicsit alakitanom a kezdeti ötlethez képest, és bár egy kisebb fejtörést okoztam ezzel magamnak, de teljes mértékben megérte mivel igy az alkalmazás felhasználóbarátabb lett. Ilyen funkció az örökbefogadás, amelyet eleinte két lépésben kellett végrehajtani az admin felületen viszont most már csak egy lépésben kell megcsinálni és ezzel most már a felhasználó felé is tudok visszajelzést küldeni.

A fejlesztés során sok problémába ütköztem, amely eléggé hátráltatta a munka előre haladását, viszont úgy gondolom ez is hozzátartozott a tapasztalat szerzéshez mivel ezek által meg tanultam, hogy hogyan is kell megkeresnem egy probléma forrását, illetve azt is, hogy ha magamtól nem tudom a megoldást akkor hogyan keressek rá az interneten. Illetve nagyon nagy segítségemre volt az IntelliJ beépített Debugger funkciója melynek köszönhetően rövidebb idő alatt megtudtam találni a hibát. Továbbá a ProgramInfo mappában a *ProgramInfoHozzaadController* készítése közben egy olyan hibába ütköztem, hogy a JavaFX nem rendelkezik olyan bemeneti mezővel, amellyel időt tudnék bekérni és eltárolni. Ezt úgy oldottam meg hogy két *Spinner*-t alkalmaztam, mint bementi mező és utána az ezek által létrehozott számértékeket össze ollóztam egy közös szövegformátummá és igy tudtam kivitelezni az idő bekérést a felhasználótól. Az említett problémából azt tanultam meg hogy mindig nyitottnak kell lennem az alternatív megoldásokra és nem szabad feladnom, illetve, hogy mindig legyek kellően kreatív, hogy megtudjam oldani a problémát.

A Java ismereteim és tapasztaltim bővítése mellett nagyon boldog vagyok, hogy megismerkedhettem a Laravel keretrendszerrel, amely úgy gondolom egy nagyon innovatív és fantasztikus keretrendszer, amely megkönnyítette a csapattársaim és az én munkámat is. Ezek mellett megtapasztalhattam, hogy milyen az, amikor egy Backend és egy Frontend alkalmazás együtt dolgozik és igy alkot egy egészet. Továbbá a fejlesztés közben arra is rájöttem, hogy a jövőben szívesen foglalkoznék Backend programozással mivel nagyon izgalmasnak találom ezt az oldalát a programozásnak.

Mindent összevetve nagyon hálás vagyok a szakdolgozatomnak mivel nagyban hozzájárult a szakmai fejlődésemhez, mert új eszközökkel és technológiákkal ismerkedhettem meg és sajátíthattam el. Illetve olyan tapasztalatokat gyűjthettem, amelyek mindenképp nagy hasznomra lesznek egy munkahelyen. Továbbá szeretnék köszönetet mondani tanáromnak, Budaházi Bencének, akihez nagy szakmai tudásának köszönhetően mindig fordulhattam, ha éppen elakadtam vagy kérdésem volt.

**English Summary:**

When choosing our topic, we considered it to be as lifelike as possible, so we used a foundation for rescuing and adopting existing dogs and cats called the Voice of Life Foundation.

My job with the project is to create a desktop application that functions as an admin interface

At the start of the project, I had no idea how I was going to make an app that contained all of my ideas. In contrast to the initial uncertainty, I can now say with complete pride that I was able to create an application that I am completely satisfied with!

During the project, I worked closely with my team, often asking for their opinions on a feature or the appearance of the interface. It was a great feeling to work in this way, because it was like we were part of a real development team, and with that we could experience what our work will be like at a development company in the future.

All in all, I am very grateful for my dissertation as it has contributed greatly to my professional development because I have been able to learn and master new tools and technologies. Respectively I was able to gather experiences that will definitely be of great benefit to me in a workplace.

# Weblap - Ambrus Pálma Zita

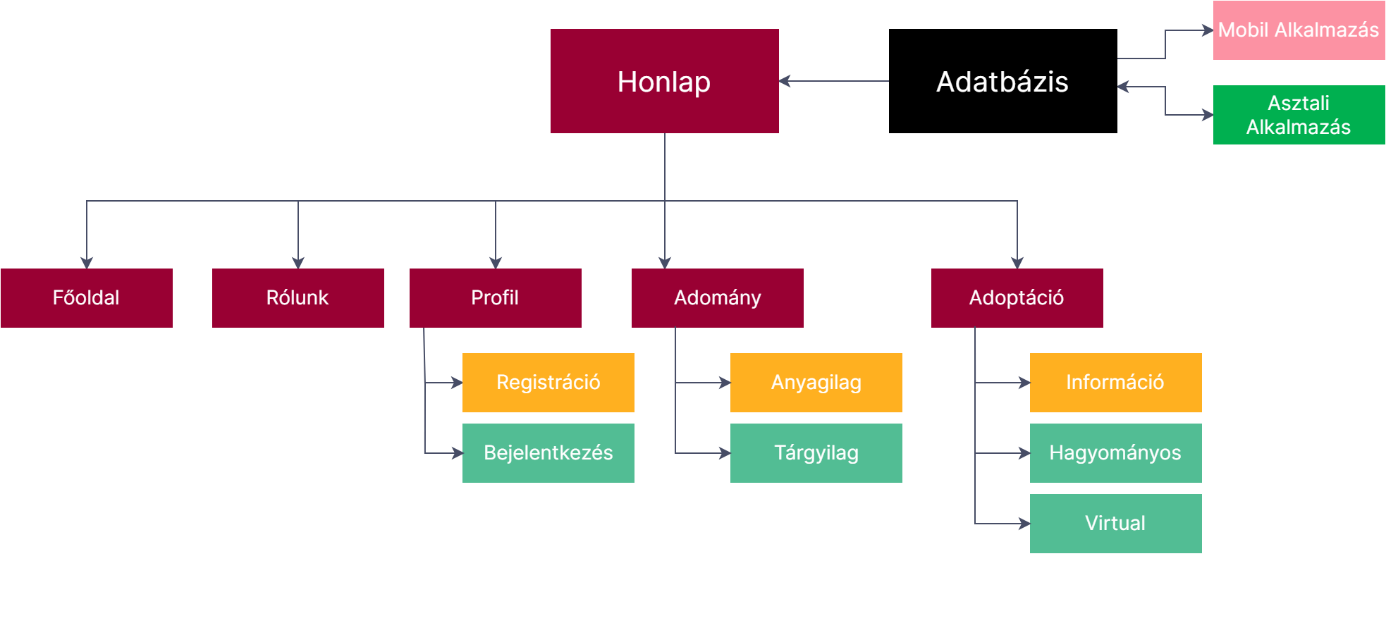
# 3. Felületterv

## 3.1 Fejlesztés nekikezdése előtti tervek

Még a fejlesztés nekikezdése előtt a honlappal kapcsolatos tervek azok voltak, hogy meg lehessen tekinteni az állatokat, a rendezvények időpontjait, az azokon készült képeket, illetve programokat, valamint az, hogy az alapítványnak az alap tulajdonságainak megismerésére is lehetőség adathasson.

A bejelentkezés vagy regisztrációt követően minden felhasználó képes legyen a programokra való jelentkezésre, valamint azokon készült képek feltöltésére, továbbá az állatok online, illetve valós időben történő örökbefogadására. Ezek mindegyikére lehetőség adódjon későbbiekben a visszavonásra, illetve az álltalunk feltett képek törlésére, ha a felhasználó úgy szeretné.

Az alapítványtól való engedélykérés sikeressége után a logó, illetve az adatok használatának lehetősége szerencsére nagyban megkönnyítette a munkánkat. Az alapítvány működő oldaláról, illetve Facebook és Instagram fiókján megtalálható adatok tartalmazták annak „történét”, elérhetőségi adatai, és képeket, információkat a gondozásuk alatt álló kutyákról és macskákról.



27. ábra - Folyamatábra

*A fenti flow-chart a honlap eredetileg tervezett működését mutatja be. Öt főoldal, valamint az azokon lehetséges folyamatok (sárga a még be nem jelentkezett, zöld a már bejelentkezett látogatóknak plusz lehetőségeit ábrázolja)*

A **Főoldalon** vélhetőleg a legújabb jövevényeink, nemrég történt és nemsokára következő programjainknak feltüntetése található majd meg, valamint a gondozás alatt lévő állatokról egy-egy kép, nevük felsorolása az oldal egyik szélén.

A **Rólunk** oldalon az alapítvány története, elérhetőségei, valamint az alapító tagjairól pár mondat foglal majd helyet néhány csoportképpel. Az oldal végére még esetleg egy mini Google térkép is behelyezésre fog kerülni, ami az alapítvány székhelyét és annak a környékét mutatja meg.

A **Profil** oldalon a már bejelentkezett felhasználónak lehetősége lehet a saját adatainak módosítására, az általa örökbefogadott, vagy örökbe fogadni kívánt állatok megtekintésére, és az örökbefogadási kérvényeinek visszavonására. A még be nem jelentkezett látogatóknak azonban ennél a fülnél csak a bejelentkezésre és a regisztrációra lesz lehetőségük.

Az **Adomány** oldalon lehetőség adódhat majd rá, hogy egy online áruházhoz hasonló módon adományozhassanak a (már bejelentkezett) felhasználóink. Erre lehetőségük adódhat például nyakörv, fésű, vagy táp vételének a formájában az Alapítvány állatainak számára. Ezen felül minden oldal végén megtalálható lenne az alapítvány számlaszáma is, arra az esetre, ha valaki pénzügyileg kívánná támogatni őket. Ez a számlaszám a valódi Élethang Alapítvány számlaszáma, és az OTP Bank Nyrt. 11705008-29915941-en található meg. Ennek a számlaszámnak a megtalálásához bejelentkezés természetesen nem szükséges.

Az **Örökbefogadás** alatt megtalálható lesz három aloldal. Az **Információ** aloldal arról nyújtana tájékoztatást, hogy milyen módon lehet adoptálni a kedvenceket, valamint milyen tényezőket ajánlott figyelembe venni adoptálás előtt, közben. Az alapítványnál lehetőség adatik virtuális és hagyományos adoptálásra is, amit a weblapon is tervezek lehetővé tenni. Ezeknek a lehetőségeknek a magyarázata megtalálható itt. Az állatok örökbefogadására lehetőséget adó aloldalak látogatására csak bejelentkezés után elérhetőek, ezen aloldalak neve **Macska** és **Kutya** lesz, könnyítve a rajtuk keresendő állatok megtalálását.

## 3.2 Elkészült honlap

**A végeredmény** a tervektől első ránézésre nem sokban tér el szerencsére, bár a lehetőségek néhol már mások. A **Főoldalon** egy rövid rólunk, pár statisztikai adat és a feltöltött képek is meg vannak jelenítve. A **Rólunk** oldal a csoportképeken, az alapító tagokról szóló pár mondatot és a GoogleMap-et leszámítva a tervekkel megegyezik. A **Profil** oldalon az adatok megváltoztatására lehetőség nincs, csak azok megtekintésére.

Az **Adomány** oldalon található valószínűleg a legnagyobb változás, ott csak fel van tüntetve pár támogató, és amelyikhez tartoznak linkek ott az általuk működtetett adományozási lehetőségek is fel vannak tüntetve. Az **Örökbefogadás** oldal és az aloldalai nem változtak az elképzeltektől, valamint az állatok egyesével való feltüntetésére szolgáló aloldal is létre lett hozva, azokon örökbefogadási lehetőség megtalálható.

Az eltervezett oldalakon felül még megjelent a bejelentkezés követően elérhető **Képek** oldal is, ahol képek feltöltése (amelyek a Főoldalon mindenki által megtekinthetőek), törlése, valamint azok leírásának módisítására van lehetőség.

## 3.3 Kinézet

|  |  |  |
| --- | --- | --- |
| 28. ábra - Natív PHP alapkinézet | Nyíl: enyhén ívelt körvonalas | 29. ábra - Laravel keretrendszeres végeredmény |

Első nekifutásra natív PHP használatával terveztem megoldani a honlapomat, melynek eredményében született meg egy alap kinézet, amit végül a Laravel keretrendszerrel, Bootstrap 5 segítségével valósítottam meg a későbbiekben.

### 3.3.1 Képek az elkészült oldalakról

|  |  |  |
| --- | --- | --- |
| Főoldal | AdományA képen szöveg látható  Automatikusan generált leírás | Rólunk |

|  |  |  |
| --- | --- | --- |
| Örökbefogadás Tudnivalók A képen szöveg látható  Automatikusan generált leírás | Örökbefogadás KutyákA képen szöveg látható  Automatikusan generált leírás | Örökbefogadás Macskák |

|  |  |  |
| --- | --- | --- |
| Örökbefogadás Egy kutya A képen szöveg látható  Automatikusan generált leírás | Örökbefogadás Egy Macska | Képek |

|  |  |  |
| --- | --- | --- |
| Profil | Regisztráció A képen szöveg látható  Automatikusan generált leírás | Bejelentkezés A képen szöveg, képernyőkép, monitor látható  Automatikusan generált leírás |
|  |  |  |

Annak érdekében, hogy kevesebb helyet foglaljanak el az oldalakról készült teljes oldalas képernyő képek a két szélső széleit körül kellett vágnom. Az utóbbi oldalakról feltüntetett képek nem az oldalak kicsinyített másai:

Örökbefogadás Egy kutya

Örökbefogadás Egy macska

Profil

Regisztráció

Bejelentkezés

# 4. Útmutatók

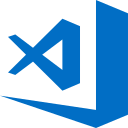
## 4.1 Fejlesztői

### 4.1.1 Rendszer követelmények:

A weblap problémák nélküli futtatásához a következő tulajdonságoknak ajánlott eleget tennie a gépnek, amin meg szeretnénk nyitni a honlapot:

* Windows 10
* 8GBRAM memória
* 2.50 GHZ processzor
* Videokártya, ami támogatja a (1080p) képernyő felbontást
* XAMPP (letölthető: <https://www.apachefriends.org/> )
* Visual Studio Code (letölthető: <https://code.visualstudio.com/download> )

### 4.1.2 Fejlesztői eszközök

A weblap a kezdetektől fogva Visual Studio Code fejlesztői környezetben készült. A Visual Studio Code egy ingyenes, nyílt forráskódú kódszerkesztő, amit a Microsoft fejleszt Linux, Mac OS X, Unix és Windows 7 vagy későbbi operációs rendszerekhez. A VS Code szerencsére a Git támogatott, és kiváló kiterjesztések érhetőek el benne. A VS Code harminc plusz támogatott nyelvvel rendelkezik, melyekből több is használatba lett véve a projekt készítése alatt (ilyenek például a HTML és a PHP). Mivel az előző tanév túlnyomó részében a Notepad++ applikációt használtam, beletelt egy rövid idő mire ehhez hozzászoktam.

30. ábra - Visual Studio Code logo

A képen szöveg, számítógép, képernyőkép, beltéri látható

Automatikusan generált leírásA XAMPP webszerver futtatása nélkül az adatbázishoz nem lehet hozzáférni (ajánlott ezt használni mivel mi is ebben dolgoztunk), ezért annak is le kell töltve lennie. Az Apache és MySQL modulok indítása után a következő bekezdésben is ismertetett módon lehet a honlapot használni.

31. ábra - XAMPP felület

A Laravel 8.83.7 és Bootstrap 5 keretrendszerek lettek használatba véve a leginkább. Előbbi a honlap alapját adja, míg utóbbi a kinézetének felépítésében segédkezett.

### 4.1.2 Létrehozás

1. A Backend clone-ozása a „Telepítés a teszteléshez” alapján
2. Ezt követően a Honlap frontendjéhez szükséges plusz parancssorok terminálba való beírása következik:

*composer* *require laravel/ui:* Backend/Frontend megfelelő működéséhez szükséges kód

*php artisan ui vue –auth:* A bejelentkezéshez, regisztrációhoz szükséges fájlokat, oldalakat létrehozó kód

*php artisan migrate:refresh –seed:* A migráció és seed-elés frissítése hogy az előző kóddal létrehozott elemek is bekerüljenek az adatbázisunkba

**A teszteléshez**, honlap használatához csak a Backend clone-ozása szükséges, amit a Backend dokumentáció rész elején megtalálható a „Telepítés a teszteléshez” fejezetnél. Segítségnek a README.md-be is bele vannak írva ezen a lépésekhez szükséges kódok.

### 4.1.3 Részletes feladat specifikáció, algoritmusok

Az **ImageUploadController**.php controllerazokat a funkciókat tartalmazza, amik a képekkel kapcsolatban kellenek. Mindegyik funkció neve beszédes, megmagyarázza miről szólnak azok:

* *addImage()*: Megállapítja, hogy egy képet a bejelentkezett felhasználó adta-e hozzá, és annak az adatainak megjelenítésében segédkezik
* *editImage(Request $request)*: A kiválasztott kép leírásának módosítását bonyolítja le
* *storeImage(Request $request)*: Egy új kép hozzáadását a public/public/Image helyre való hozzáadását bonyolítja le
* *deleteImage(Request $request)*: A kiválasztott törlését bonyolítja le

A **Postimage**.phpmodell-hez tartozik az ImageUploadController.php, ez van mindig meghívva amikor például a storeImage(Request $request) funkciónál, amikor új képet hozunk létre. Ez tartalmazza milyen adat meghívható, és miket lehet megadni adni készítés esetén. Ezek az ’image’ (kép neve), ’user\_id’ (annak a felhasználónak az id-a aki feltöltötte a képet) és a ’description’ (kép leírása).

A **2022\_04\_05\_140336\_create\_postimages\_table**.phpmigration-nel jön létre a modell az adatbázisban amikor létrehozzuk a táblát. Ehhez tartozik egy **PostimageSeeder**.php seeder, valamint **PostimageFactory**.php factory is, amelyek segítségével a clone-ozás elvégzése után már tartalmaz tizenöt képet a honlap, amit a 3-as id-val rendelkező felhasználó (én, Ambrus Pálma Zita) „töltött” fel.

Az **AdoptionController**.php controllerazokat a funkciókat tartalmazza, amik az adoptáció létrejöttéhez kellenek. Ezek a store(Request $request) és a dogStore(Request $request). Ezek a funkciók a kutyák és a macskák saját oldalán az Örökbefogadás gombra kattintva használtak:

* *store(Request $request)*: A cicák örökbefogadását lebonyolítását teszi lehetővé
* *dogStore(Request $request)*: A kutyák örökbefogadását lebonyolítását teszi lehetővé

Ehhez a controller-hez tartoznak az **Adoption**.php, az **AdoptionFactory**.php, a **2021\_02\_08\_115118\_create\_adoptions\_table**.php, az **AdoptionSeeder.**php nevekkel rendelkező modell, factory, migration és seeder fájlok.

### 4.1.4 Blade

A képen szöveg, emlősök, képernyőkép látható

Automatikusan generált leírásA resources/views/ elértetési útvonalon megtalálható blade.php végezetű fájlok, amelyek a Laravel keretrendszerben az alapértelmezett weboldal kiterjesztés. Az **app**.blade.php az a fájl, ami alapul szolgál, ebben található meg a navbar és a footer, valamint ezen honlap esetében a minden oldal alján megtalálható adományozási lehetőség.

32. ábra - Adomány

|  |  |  |
| --- | --- | --- |
|  | Nyíl: enyhén ívelt körvonalas | A képen szöveg látható  Automatikusan generált leírás |

A **login/** és **register**.blade.php a php artisan ui vue –auth kóddal jött létre. Ezek meghívásával tud a látogató regisztrálni és bejelentkezni. A navbar-on ezt jobb oldalt lehet megtalálni egy profil ikon kattintását követően, mint a legördülő menü egy-egy tagját. A bejelentkezés lebonyolítását követően ez az ikon eltűnik, helyette a bejelentkezett beceneve jelenik meg.

### 4.1.5 Routes

A **routes**/**web**.php-ban megtalálhatóak a route útvonalak. Ennek a projektnek a weboldalához összesen tizenhat útvonal készült. Ebből tizenegy get, három post, egy delete és egy patch. A **get** útvonalak a megjelenítést teszik lehetővé, az app.blade.php-ben létrehozott navbar is ezeket használja fel az oldalakra való átirányításhoz (például: ’/’ a Főoldal, a ’/donation’ az Adomány).

A Macskák és Kutyák oldalak ezt egy kicsit a következő szintre viszik, az ott foreach metódust használva megjelenített állatokat tartalmazó card-okban jobb felső sarokban lévő nyíl, azon állat id-át átadva a ’/cats/{cat}’ post útvonalnak, a kiválasztott cica és azon adatai jelennek meg legközelebb. Felül a címsorban ilyenkor, ha például a kettes id-vel rendelkező kutyára kattintunk az utóbbi URL kód jelenik meg: <http://127.0.0.1:8000/dogs/2>.

A **post** útvonalak az új adat felvétellét teszi lehetővé (például a: ’/cats/{cat}’ post útvonalon az az adat mentődik el, hogy az éppen bejelentkezett felhasználó örökbe fogadta azt a macskát ami az oldalon meg van jelenítve). A **delete** a törlésnél, a **patch** útvonal pedig a módosításnál kerül használatba (ezek csak a ’/pictures’ elérhetőségen a fent említett Képek oldalon találhatóak meg ebben a projektben).

## 4.2 Felhasználói

### 4.2.1 Magyarázatok

A **regisztráció** lehetősége a navbar-on a profil ikon-ra való kattintást követően megjelenő Regisztráció-ra kattintva lehetséges. A megjelenő oldal kitöltését követően a *Regisztráció* gombra kattintva automatikusan bejelentkeztet a honlap. **Kijelentkez**ni a profil ikon helyén megjelenő becenevedre kapcsolva található meg a legördülő menü legalján a Kijelentkezés opció amire csak rá kell kattintani, hogy az megtörténjen. Ezek után **bejelentkez**ni a regisztráció fölött megjelenő Bejelentkezés-re kattintva lehet az ezt követő oldalon keresztül.

A honlap tetején megtalálható a **navbar**. Ennek segítségével lehet az oldalak között közlekedni. A logóra és az alapítvány nevére, valamint a Főoldal-ra kattintva lehet a főoldalra menni, valamint mindegyik szó az arra kijelölt oldalra irányít, amiről a nevét kapta. Bejelentkezés előtt az Örökbefogadás gomb alatt csak a Tudnivaló részre lehet kattintani, valamint a profil ikonra kattintva be lehet jelentkezni és regisztrálni.

33. ábra - Footer

34. ábra - Navbar

A honlap alján megtalálható a **footbar**. Ennek bal oldalán megtalálható a valódi alapítvány Instagram fiókja, középen a jelenleg (2022.04.11) élő honlapja, valamint jobb oldalt a Facebook fiókja. Ezek egy-egy a hozzályuk tartozó ikonokra való kattintást követően érhetőek el.

Ezen felül még akkor van lehetőség az oldalakon keresztül ugrálni amikor már be vagyunk jelentkezve. Erre az állatok képeire, neveire kattintva a Főoldalon, valamint az állatok neve mellett lévő nyilakra kattintva a Kutya és Macska oldalakon. Ezek mindegyike a kiválasztott állat oldalára vezet át, ahol azok tulajdonságait lehet megtekinteni, illetve esetlegesen örökbefogadásuk is lehetséges.

Bejelentkezés után ezen felül még jobb oldalt a Kijelentkezés fölött megtalálhatóak a **Profil** és **Képek** opció is. A Profil opció átirányít egy oldalra, ahol a bejelentkezett felhasználó adatai találhatóak meg. A Képek opció pedig egy olyanra, ahol képeket lehet feltölteni, ezeket törölni, valamint azok leírását módosítani. Leírást megadni kötelező, minden funkció helyes használatát követően megjelenik egy annak sikerességét zöld színben pompázva tudató **üzenet**.

35. ábra - Hibaüzenet

### 4.2.2 Weblap előszöri látogatásakor ajánlott útmutató

A weblap előszöri látogatásakor ajánlott egy regisztrációt végrehajtani, ezt a fentiekben leírt módon, a Regisztráció menüpontnál lehet elvégezni. Az űrlapot teljes egészében, élő adatokkal érdemes kitölteni, hiszen a későbbiekben ezen adatoknak a megváltoztatására még nincsen lehetősség. Ha minden mező kitöltésre került akkor az űrlap végén található jól kitűnő, látszólag oda nem illő kék gombra való kattintás követően a honlap be is jelentkeztet.

Ilyenkor a Főoldal fog megint megjelenni, ahol pár érdekesség jelenik meg az alapítványról. Új kiskedvenc **örökbefogadás**ára érdekében látogatóknak, ha eddig még nem tették meg, érdemes elolvasni az információkat ezen topikról, amit az Örökbefogadás menüpont alatt a Tudnivalóknál találhatóak. Ezt követően a Tudnivalók alatt ki lehet választani kutyákat vagy macskákat keresünk, és bele is lehet vetni magunkat az állatok keresésébe.

Az állatokat ezen felül még a Főoldalon is meg lehet találni, rájuk kattintás után átirányít az állat oldalára. Itt is mint a Macskák és Kutyák aloldalakon is segítségül jön, hogy ha nőstény állatról van szó az rózsaszínű, ha hím akkor kék színű névvel rendelkezik. Örökbefogadási kérvényünk leadásához elég, ha az állat saját oldalán lévő Örökbefogadás gombra kattintunk. Ez csak akkor lehetséges, ha az még gazdátlan.

**Adomány**ozása szándékkal érkező vendégeinknek az Adomány oldalunk meglátogatása ajánlott, ehhez regisztráció sem szükséges. Itt megtalálható több adományozási lehetősség is. A DaDo-nak köszönhetően lehetősség van kutyatápot venni a gondozás alatt lévő kutyáknak, amihez a **tápadomány**t gyűjtő oldalt egy kékkel kiemelt betűszínnel rendelkező linkkel az oldalon könnyen meg is található.

**Pénz adomány**ozási lehetősség is fent áll, ez a minden oldal alján megtalálható számlaszámra való utalással történik, vagy a számlaszám alatti képen is látható telefonszám felhívásával. Ezek fölött az alapítvány e-mail címe is megtalálható, ha **kapcsolatfelvétel**i eshetőséget keresünk.

Abban az esetben, ha látogatásunk csak tudakozódási jellegű, érdemes megnézni az előbbi oldalakon kívül a Rólunk menüpontot is. Itt megtalálható az alapítvány története röviden, működésünk mikéntjei, lehetősségeink és céljaink a jövőre nézve.

# 5. Tesztelések

## 5.1 Csak bejelentkezés után elérhető oldalak bejelentkezés előtt megpróbált elérése

|  |  |
| --- | --- |
| Beírt URL  <http://127.0.0.1:8000/cats>  <http://127.0.0.1:8000/cats/1>  <http://127.0.0.1:8000/dogs>  <http://127.0.0.1:8000/dogs/1>  <http://127.0.0.1:8000/pictures>  <http://127.0.0.1:8000/profile>  <http://127.0.0.1:8000/home> | Átirányított oldal A képen szöveg, képernyőkép, monitor látható  Automatikusan generált leírás |

Mielőtt a bejelentkezés megtörténik az URL beírása után a Route::get-ek végére be lett írva, hogy ->middleware('auth') még beadta a kért oldalt ilyen-olyan módon (a profilnál például egyértelmű hibákba ütközve), de azt követően az elvárt oldal amire át is irányít az a bejelentkezési platform.

## 5.2 Regisztráció

|  |  |
| --- | --- |
| Regisztrációs űrlap kitöltése | Űrlap beküldése utánA képen szöveg látható  Automatikusan generált leírás |

Ha a regisztrációt érvényesen kitöltjük akkor a gomb lenyomása követően elvárt és megtörténő művelet az az, hogy az adatoknak az adatbázisban való létrehozásával egyidőben a honlap automatikusan bejelentkeztet.

## 5.3 Bejelentkezés

Helyes e-mail cím és jelszó megadását követően a bejelentkezés minden probléma nélkül megtörténik.

|  |  |
| --- | --- |
| Bejelentkezési űrlap kitöltése | Űrlap beküldése utánA képen szöveg látható  Automatikusan generált leírás |

## 5.4 Kijelentkezés

|  |  |
| --- | --- |
| Gombra nyomás előtt A képen szöveg látható  Automatikusan generált leírás | Gombra nyomás után |

A Kijelentkezés opció kiválasztásának bekövetkezése után a honlap kijelentkeztet és átirányít a Főoldalra.

## 5.5 Örökbefogadás

|  |  |
| --- | --- |
| Gombra nyomás előtt A képen szöveg látható  Automatikusan generált leírás | Gombra nyomás után |

Az örökbefogadás csak akkor lehetséges, ha az állat még nincsen örökbe fogadva. A gomb lenyomását követően létrejön egy új adatsor az adoptions táblában, ahol el van mentve többek között a bejelentkezett felhasználónak az id-ja, majd ennek az adoption id-ja elmentődik az állat adatsorának az adoption\_id-ba. Ezek után a gomb eltűnik és egy kapunk egy visszajelzést a sikeres műveletről. Ha az állat örökbe van fogadva már, akkor a gomb nem jelenik meg ott, csak az üzenet, hogy a kedvenc már gazdára lelt.

## 5.6 Kép feltöltése

Mivel a fájl kiválasztásának a validációja még nem teljes ezért mindaddig amíg valami ki van választva addig elmenti, de tegyük fel, hogy egy képet választottunk ki. A leírást is kötelező megadni, nem szabad üresen hagyni egyik mezőt sem. Az ikonra való kattintás utál sikeres feltöltés esetén megjelenik egy üzenet erről, és a kép is ott lesz a Képeim között.

|  |  |
| --- | --- |
| Kép választása, leírás hozzáadása  A képen szöveg látható  Automatikusan generált leírás | Gombra nyomás után |

## 5.7 Kép leírásának a módosítása

Az input mezőt nem szabad üresen hagyni az író ceruzaikonra való nyomás előtt, ha a nélkül próbálkozunk akkor meg fog jelenni egy figyelmeztetés. Ha helyesen töltünk ki mindent akkor megjelenik egy a sikerességet igazoló üzenet, és a leírás is megváltozik.

|  |  |
| --- | --- |
| Gombra nyomás előtt | Gombra nyomás után |

## 5.8 Kép törlése

|  |  |
| --- | --- |
| Gombra nyomás előtt | Gombra nyomás után  A képen szöveg látható  Automatikusan generált leírás |

A kép törlése egyszerű, csak a kártya jobb felső oldalában megtalálható kuka ikonra kell nyomni, és a sikeres műveletről visszaigazoló üzenet jelenik meg.

## 5.9 Ikonos átirányítás

A footer-ben megtalálható három ikonnak át kell irányítania az eredeti honlap Instagram fiókjára, most (2022.04.13) működő honlapjára, és a Facebook fiókjára.

|  |  |  |
| --- | --- | --- |
| Ikon | Ikon  A képen szöveg látható  Automatikusan generált leírás | Ikon |
| Átirányított oldalA képen szöveg, különböző, különféle, azonos látható  Automatikusan generált leírás | Átirányított oldal | Átirányított oldal |

## 5.10 Statisztika változások

A Főoldalon található statisztikáknak változnia kell ha például új önkéntessel bővül a csapatunk. Mivel a weblapon csak a regisztrációval lehet ezeket a számokat változtatni (új állatot felvenni, illetve azokat törölni, és egy felhasználó törlése is csak az adatbázisban való módosítással, illetve az asztali alkalmazás admin felületének használatával lehetséges).

|  |  |
| --- | --- |
| Regisztráció előtt | Regisztráció után |

# 6. Ismert hibák

Sajnálatos módon nem találtam időt arra, hogy néhány hibát megoldjak. Ezek voltak az Elfelejtett jelszó és az Emlékezz rám a bejelentkezésnél (‘/login’). Illetve a regisztrációt követően a regisztrált e-mail-re sem küld ki egy megerősítést kérdező üzenetet. E lehetőségek még a php artisan ui vue –auth kód terminal-ba való beírásakor jelentek meg, azonban maga a kód nem tette automatikusan működőképessé meg valamiért.

A képen szöveg látható

Automatikusan generált leírás Ismert hibának értelmezhető az is, hogy néhol a validáció angolul jelenik meg, illetve, hogy a képek megjelenítésekor a kártyák magassága, bennük lévő képek mérete nem esztétikusan egyelőre van beállítva. Az utóbbi mondatban felsorolt hibákat hosszú időn keresztül próbáltam megoldani sajnos sikertelenül, azonban a fent említettekre már a kezdetektől csak, mint továbbfejlesztési lehetőségként gondoltam.

Ezen kettő blokkon felül még ismert hibának mondható, hogy a felhasználó a képek feltöltésekor nem ad fel validálási hibát, ha más fájlt próbál meg feltölteni. Erre egy jó példa például az, amikor ezen dokumentumot próbáltam feltölteni.

36. ábra - Példa a kép feltöltése hibára

# 7. Továbbfejlesztési lehetősségek

A továbbfejlesztés nulladik lépéseként az ismert hibáknak a megoldását venném figyelembe. Ezek esztétikai okokból is elsőbbséget élveznek hiszen már jelen vannak a honlapon, ámde nem rendeltetésszerűen működnek. Következőleg a megoldott hibákhoz valamelyest módon kapcsolódó dolgokat venném előre, amely egyelőre még csak a képekre történő kommentelési lehetőség létrehozása, azok tetszésének valamilyen módoni kifejezését takarná.

Következőleg jönnének azon lehetőségek, amik eredetileg a projekt készülése alatt változó okok miatt elvetett ötletek. Az Adomány oldalon eredetileg egy online áruház oldalára hasonlított volna, amit JavaScript segítségével oldottam volna meg, de idő közben elfelejtettem ezt, és amikor visszaolvastam a terveimet már mások voltak az elképzeléseim ezért nem léptettem életbe az ötletet.

A profil módosítása, valamint a programokra való jelentkezés, azok módosítása és törlése is munkálatok alatt ált, mindkettőt más okból történt elvetésre. A profil módosítása a weben használható user controller meglétének a hiányában lett félre rakva, helyette kezdtem el a képekkel kapcsolatos funkciókon dolgozni. A programokkal kapcsolatos lehetőségek mellőzése az elvárt követelményekkel elkészültével lett elvétve.

A maradék három továbbfejlesztési ötlet többségében eredetileg is továbbfejlesztési ötletként jutott az eszembe. Többek között email küldését (az oldalon történő üzenet küldése a profil oldalról a bejelentkezett felhasználó e-mailjéről egy előre beállított emailre)és téma választását (sötét vagy világos honlap téma) nem is terveztem kifejleszteni. A Google térkép a Rólunk oldalon való megjelenítése azonban azért nem jött össze, mert az alapítvány nem nevezett meg székhelye, legalábbis én nem találtam meg sem a most (2022.04.17) élő honlapjukon, sem a Facebook avagy Instagram fiókjaikon.

# 8. Összefoglalás a projekt készítése alatti élményeimről

## 8.1 Magyarul

A projekt nekikezdésénél a célom az volt, hogy könnyen átlátható, szép, és összességében felhasználóbarát weblap kerüljön ki a kezeim közül, amit úgy érzek sikerült is elérnem. Szerencsésnek érzem magam, hogy a csapattársaim is beleegyeztek abba, hogy én készíthessen a honlap részét a vizsgaremekünknek, hiszen a jövőben is ebbe az irányba szeretnék tovább haladni.

A Laravel keretrendszerben való munka nagyban megkönnyítette a haladásom. Bár a route elérésekkel megakadt párszor a bajom, szerencsére nem állított meg teljesen a munkámban. Tanáraink többsége szívesen válaszolt kérdéseimre, helyes irányba tereltek. Csapattársaimra is bátran számíthattam, ha kérdésem volt.

Habár sajnos néha elfelejtettük említeni egymásnak, ha a Backendben változások történtek, de ilyenkor mindig számíthattunk egymásra, hogy ha nem is rögtön, de biztosan a másik tudására hozzuk, hogy módosulások történtek. Ilyenkor mindig értelme lett a feltünedező hibáknak, és ki is tudtuk őket javítani, anélkül, hogy tovább hátráltattuk volna egymás munkáját.

Hiszen ötletekből sosem akadt hiány, az igazi kihívást az jelentette, hogy kiválasszam mely ötleteket valósítsam meg. Ebben néha segített az is, hogy hajlamos voltam elfelejteni az ötleteimet. Így történt meg, többek között az is, hogy az Adomány oldalon nincsen online áruházhoz hasonló módon lehetőség az alapítvány számára.

Összegségében megtanultam segítséget kérni a feladataim során felmerült kérdéseimnek megoldásával kapcsolatban, valamint, hogy még sokat kell dolgoznom a kitartásomon.

## 8.2 In English

At the beginning of the project, my goal was to get an easy-to-understand, beautiful, and overall user-friendly website out of my hands, which I feel like I did manage to achieve. I feel lucky that my teammates have also allowed me to make the projects website part, as I want to continue in this direction in the future.

Working with Laravel framework has made my job a lot easier. Although I had some problem while making route access a few times, but luckily it didn't stop me completely at work. Most of our teachers were happy to answer my questions and walk me down in the right direction. I could also count on the help of my teammates if I had a question.

Although unfortunately sometimes we forgot to mention to each other when there were changes in the Backend, in such cases we could always count on each other to bring the other to the knowledge of the other, even if not immediately. The errors suddenly made sense, and we were able to correct them without further hindering each other’s work.

Because there was never a shortage of ideas, the real challenge was to choose which ideas to implement. This was sometimes helped by the fact that I tend to forget my ideas. It so happened that the Donation webpage doesn’t have the feeling of an online store.

Overall, I learned to ask for help with resolving my up-and-coming problems, since I still have a lot ahead of myself.

# Mobil Alkalmazás – Nádasdi Barbara

# 3.Felületterv

**Bejelentkezés**

|  |  |
| --- | --- |
| Tervezett | Végleges |
| A képen szöveg látható  Automatikusan generált leírás | A képen szöveg látható  Automatikusan generált leírás |

**Regisztráció**

|  |  |
| --- | --- |
| Tervezett | Végleges |
| A képen szöveg látható  Automatikusan generált leírás |  |

**Kezdő képernyő**

|  |  |
| --- | --- |
| Tervezett | Végleges |
| A képen szöveg látható  Automatikusan generált leírás | A képen szöveg, emlősök, házi macska látható  Automatikusan generált leírás |

**Kutyák/macskák**

|  |  |  |
| --- | --- | --- |
| Tervezett | Végleges | |
| A képen asztal látható  Automatikusan generált leírás | Macskák  A képen szöveg látható  Automatikusan generált leírás | Kutyák  A képen szöveg látható  Automatikusan generált leírás |

**Rendezvények**

|  |  |
| --- | --- |
| Tervezett | Végleges |
|  | A képen szöveg látható  Automatikusan generált leírás |

**Programok**

|  |  |
| --- | --- |
| Tervezett | Végleges |
| A képen szöveg látható  Automatikusan generált leírás | A képen szöveg látható  Automatikusan generált leírás |

Számos oldaltervezet volt még, mert eredetileg több ismeretközlő oldal is lett betervezve, azonmban nem szerettem volna túl egyhangúvá vagy unalmassá tenni, ezért inkább azokat kitöröltem őket. Megtalálható lett volna rajta, hogy kik azok az emberek, akikre mindig lehet számítani, milyen formában lehet támogatni vagy milyen feltételeknek kell teljesülnie ahhoz, hogy valaki újdonsült gazdivá válhasson.

Az alkalmazásomba ugyanúgy lett volna külön kutya és macska oldal, de ez csak arra szolgált volna, hogy meg lehessen nézni, milyen állatok vannak az alapítvány gondozása alatt. Ezen felül egy külön oldalon pedig azon állatok lettek volna, akik örökbefogadhatóak. Azonban mivel kevés olyan állat van, akit ideiglenesen nem lehet örökbe fogadni (például oltáshiány miatt), ezért inkább az egyszerűség és letisztultság miatt ezt egy oldalon belül oldottam meg, ahol csak azok az állatok jelennek meg, akik készen állnak családhoz költözni.

# 4. Fejlesztői dokumentáció

## Alkalmazott fejlesztési eszközök

Kulcsfontosságú volt már a fejlesztés előtt kitalálnom, melyik programozási nyelven szeretném írni a vizsgaremekem. Mivel a tanév elején elkezdtük a Kotlin programozásinyelvet tanulni, ezért sokáig úgy gondoltam ebbe fogom írni a vizsgaremekem. De mivel volt külön Java óránk és a Mobil fejlesztésen is átálltunk Java nyelvre így hatékonyabbnak ítéltem meg ha ezen a nyelven készítem el az alkalmazásomat.

Mivel Rest Api alkalmazást kellett csinálnunk, ezért a fejlesztő környezetem nem csak a JetBrains által készített *Android Studió*, hanem a *Visual Studio Code* is, mert itt fejlesztettük a Laravel keretrendszert használó Backendünket és ezáltal itt kellett elindítani a szerverünket is.

Amíg a tábláink szerkezetét terveztük, *MySQL Workench*-et használtunk, mert felhasználóbarát és ami ennél is fontosabb, hogy a vizuális megjelenítése megkönnyítette a dolgunkat, amikor elsőnek SLIM keretrendszerrel kezdtük el fejleszteni a backendünket. Így amikor a végleges Laravel keretrendszert kezdtük el használni, már ezalapján megtudtuk oldani könnyen az adatszerkezetet és a kapcsolatokat *PHP MyAdmin*-ban.

Dokumentációmat a mindenki által jól ismert Microsoft Word-ben készítettem, a mobil kinézetét pedig egy nagyon hasznos, könnyen kezelhető Balsamiq nevű alkalmazásban.

# 5. Részletes feladatspecifikáció, algoritmusok

A Java mappámban 27 fájl található összesen. Ezek közül 5 darab Activity, 3 Rest Api-t segítő osztály, 6 db Fragment és 13 segédosztály. Átláthatóság miatt minden fájlt, ami összetartozik egy almappába helyeztem el.  
Ezek mellett érdemes említeni a layout mappát, ahol a grafikai része található az alkalmazásnak. Sajnálatos módon ezeket nem lehet almappákba rakni így van 17 db .xml fájlom.   
Ezenkívül csináltam egy menu mappát a layout mappa mellé, ahol elhelyeztem a kilépő gombot, a diavetítést, és azt, hogy melyik oldalakra lehet menni az alkalmazáson belül.  
A mipmap mappában tárolom a külön hozzáadott képeket. Itt található a 6 kép a diavetítéshez és a logo is.  
Ahhoz, hogy lehessen betöltő képernyő az alkalmazásomban, létre kellett hoznom egy raw nevű mappát, ahol az animáció JSON kiterjesztésű fájlját tárolom.   
A következőkben ismertetni fogom a fontosabb mappákban lévő activityket, fragmenteket, osztályokat és metódusokat:

## 5.1 Cat mappa

### 5.1.1 Cat osztály

A *Cat* elevezésű osztály arra szolgál, hogy azokat az adatokat, amelyekben a Backendben létrehoztunk egy Java osztályban ellehessen tárolni.

Ha nem úgy neveztük el, ahogy az adatbázisunkban szerepel, megtehetjük azt ha az előtte lévő sorba beleírjuk azt hogy *@SerializedName(„”)*. A két időzőjel közé pedig azt a nevet kell beírni, ahogy az adatbázisba elneveztük.

private int id;  
**@SerializedName**("name")  
private String catName;  
private String description;

A létrehozott adattagokhoz legeneráltattam egy paraméteres konstruktort, hogy a Cat osztályban található adattagokat inicializálja.

Továbbá generáltattam egy getter és setter metódust. A Getter arra szolgál, hogy le tudjam kérdezni az adattagok értékét. A Setter pedig arra szolgál, hogy meglehessen változtatni az adattagok értékét.

### 5.1.2 Cat Adapter

Első lépésként létre kellett hoznom egy statikus *CatHolder* osztályt, amit a RecyclerView.ViewHolder-ből származtattam. Itt meg kellett adni a CatFragment-hez tartozó layout fájlban (list\_cats.xml) létrehozott fajtáját és azt a nevet, amit az „*android:id="@+id/*” után írtunk, hogy tudja hova hivatkozzon. Ezután létrehoztam egy konstruktort, hogy inicializálni tudjam az adattagokat.

Miután ezt megtettem, a CatAdaptert szintén a CatHolder Osztály segítségével lehet származtatni. Felveszünk két adattagot. Az egyiken betudjuk tölteni az eredményeket, a másik pedig maga a lista, amit megjelenítünk. Létrehozunk ezekhez az adattagokhoz egy konstruktort itt is, majd, hogy a linkünkön elérjük az adatokat az adatbázisból, létre kell hozni egy Setter-t is a listánkhoz.

Amikor származtattuk a CatAdapter osztályunkat le kellett generálttani 3 Override metódust.

* *onCreateViewHolder():* összekapcsolódik azzal a layout fájllal, amelyekből kiszedtük az adattagokat a CatHolder osztályba.
* *onBindViewHolder():* átadjuk azokat az adatokat, a holder segítségével, amelyek az adatbázisban szerepelnek, ehhez kell a Cat osztályban létrehozni a Getter-eket, valamint abban az esetben ha szeretnénk rátenni valami eseményt, akkor azt itt tudjuk megtenni  
  Az örökbefogadást itt hajtjuk végre a már előzőleg eltárolt token és egy alertdialog pozitív és negatív gombjának segítségével, itt hívjuk meg a külön létrehozott RequestTask segédosztályt
* *getItemCount():* visszaadja a listának a méretét azaz az adatokat

RequestTask osztály:

* *doInBackground*(): az alert-hez hozzárendeljük, hogy tudja hogyan szeretné a felhasználó örökbe fogadni az állatot (virtuális / általános)
* *onPostExecute*(): az elküldött üzenetet JSON formátummá alakítja és megjeleníti

### 5.1.3 Cat Fragment

Amikor megcsináljuk a blank fragmentünket, hogy hozzon létre hozzá egy layout fájlt, csinálni kell egy onCreateView() metódust, ami arra szolgál hogy ha rámegyünk majd az oldalra, akkor megtudja nekünk jeleníteni a már előzőleg létrehozott listát.

Csinálunk egy init függvényt, amibe inicializáljuk a *fragment\_cat.xml* fájlban létrehozott recyclerView-t, és ehhez állítunk be mindent a helyes megjelenítés érdekében. Itt inicializáljuk a Catadaptert is, hogy lehessen ehhez kapcsolódni.

Ebben az osztályban kell egy változóként létrehozni azt, azokat az url-eket, amivel az alkalmazás hozzá tud csatlakozni a backend-ünkhöz. Ennek az url-nek tartalmaznia kell, hogy http://, a futtatott eszköz IP címét, a port számát, /api és a backend-ben létrehozott Route mappában az ehhez tartozó megnevezést.

Emulátorral tesztelt link:  
*private* String url = "http://10.0.2.2:8000/api/cat";

Továbbá létre kellett hozni egy RequestTask osztályt és a hozzá tartozó két Override metódust.

* *doInBackground():* egy try-catch-en belül hozzákapcsolódik az előbb említett url-hez.
* *onPostExecute():* JSON formátummá alakítja a listánkat a szép megjelenítés érdekében

Nem részletezem a többi olyan osztályt, ami Rest Api-val van meghívva, mert ugyanígy épül fel. 3 db Osztály (fragment, adapter, és a backendből meghívott adattagok osztály) és 2 db layout fájl a megjelenítéshez.

Azok az osztályok, amelyek nem igényelnek Rest Api-t azok csak egy Fragment osztályt tartalmaznak és a hozzá tartozó layout fájl-t, mert csak tájékoztatás szerepet töltik be azok az oldalak az alkalmazáson belül. Egy ilyen osztály van az egész alkalmazásmon belül.

## 5.2 LoadingActivity

Ez az osztály a betöltő képernyőt tartalmazza. Kiemeltem egy külön statikus változóba azt, hogy mennyi ideig szeretném, ha megjelenne az alkalmazás betöltésekor, annak érdekében hogyha később ezt szeretném változtatni, akkor ne kelljen már a kódban keresgélni.

Egy run Override metódus-t helyeztem el ebbe az osztályba, ahol megadom neki azt, hogy amikor lejár az előzőleg megadott idő, melyik oldalra (Activity-re) dobjon át. Ehhez létre kellett hoznom 2 db xml fájl-t az anim mappában, mert enélkül nem tud működni.

Beállítottam, hogy amikor ezt a képernyő látja a felhasználó, akkor ne csúfítsa el azzal, hogy a telefon legtetején található dolgok ott vannak (például: *szolgáltató, idő, akkumulátor százalék*).

## 5.3 MainActivity

Létrehoztam külön változókba azokat az adattagokat, amelyeket az activity\_main-en létrehoztam és aztán az init metódusban inicializálni kellett őket, hogy használhassam őket a különféle függvényekben az osztályon belül.

Szintén beállítottam ebben az Activity-ben is, hogy teljesképernyős legyen, ezzel ne rontsa el az összképet az alkalmazásról, de ugyanúgy lelehet húzni ezen felül, ha szeretné a felhasználó.

A „Hamburger icon” létrehozását itt kell végrehajtani. Ez a 3 db csíkot jelöli, amire rá lehet nyomni, vagy szimplán csak a képernyő bal oldaláról elkezdjük húzni és megjelenik az a lista, ahol az oldalak találhatóak. Ha ezt nem hozzuk létre nem tudjuk használni ezt a *NavigationView*-ot.

Mivel ez a főoldal, ezért a felső zöld sávban (toolbar) lévő 2db icon-ra itt kell megírni a kattintás esemény-t. Egyiknél megvizsgálom egy felugró ablak segítségével, hogy a felhasználó biztos szeretne-e kilépni az alkalmazásból. Mind a két választáshoz külön Override metódus tartozik, majd visszaadom ennek az eredményét és úgy fog viselkedni az alkalmazás. Ha pedig az ember icon-ra megy átdobja egy másik Activity-re, ahol módosíthatja az adatait a jelszó kivételével.

Van egy külön Override metódus, ami egy switch case-ből áll, ami egy elemet kér vissza. Előzőleg amikor minden elemnek adtunk egy külön ID-t a layout fájl-ba (menu\_item.xml), hogy megtudjuk őket különböztetni, ezt felhasználva, megnézi melyik oldal lett kiválasztva és átlép. Minden oldalhoz írt case eset-be be kell állítani a láthatóságát eltűnt-re azoknak az elemeknek, amiket a MainActvity tartalmaz. De fontos, hogy a fragmentünk-nek nem állíthatjuk be a láthatóságát eltűnt-re, mert máskülönben nem fogunk tudni átmenni az adott oldalra. Majd a fragmentek cserélődnek es egymásra pakolja az alkalmazás. Mindennél fontosabb, hogy ha mindent beállítottunk, le kell zárni egyesével a case-t egy break-kel vagy különben hibára fut az alkalmazás.

## 5.4 RequestHandler

Paraméter nélküli konstruktort létrehozunk, majd 6 db statikus osztályt és mindegyiket IOEceptionnal látjuk el, hogy ha hozzáférhetőségi hiba lépne fel:

|  |  |
| --- | --- |
| **Függvény** | **Leírás** |
| *get()* | Egy url-t kap paraméterének, ezzel a metódussal tudjuk megjeleníteni az adatokat a backendből |
| *post()* | Url-en keresztül elküldjük a kérést a szervernek (pl: bejelentkezés) |
| *delete():* | Url-t kap egyetlen paraméterének és bármit lehet vele törölni, amire meg van hívva |
| *put():* | Paraméternek kap url-t, megjelenített adatokat és egy tokent. így a bejelentkezett felhasználóra vonatkozó adatokat lehet módosítani |
| *addrequestBody():* | Put segédfüggvénye, átalakítja az adatokat JSON formátumra és UTF\_8-ra állítja |
| *setupConnection():* | Url-t kap paraméternek beállítjuk, hogy jelenítse meg nekünk a kapott adatokat JSON formátumban és hogy mennyi ideig várakozzon |
| *getResponse():* | A szervernek elküldött kérés válaszkódja(lekérés) és ezt visszaküldi nekünk |
| *postResponse():* | A Post segédfüggvénye, elküldött kérés JSON formátummá alakítása és annak válaszkódja |
| *postAuth():* | Url-t, kérés-t és token-t kap paraméternek, a kapott token által tudja elküldeni a kérést a szerver felé adott felhasználóra |
| *getAuth():* | Url-t és token-t kap paraméternek, hogy azokat az adatokat lelehessen kérni, amit a bejelentkezett felhasználó választott |

## 5.5 Response

RequestHandler és a RequestTask osztályokhoz tartozó segédosztály.Visszaadja a válaszkódot és az adatot. Nincs külön requestTask osztály, csak abban a fragmentben van létrehozva, ahol épp szükség van rá.

## 5.6 Token

Amikor a felhasználó bejelentkezik, minden alkalommal kapni fog egy tokent. Ez minden belépésnél változni fog és az előző érvényét veszti. Ezt tárolja el az adott felhasználóról az osztály, és segít hozzá, hogy további funkciók legyenek elérhetőek a bejelentkezett számára. Mindezt az adatbázisban egy külön táblában tárolja el.

# 6. Felhasználói dokumentáció

## 6.1 A program általános specifikációja

A Mobil Alkalmazás nagyon nagy összhangban van a Weboldallal. A felhasználó az alkalmazás megnyitását követően tud regisztrálni, hogy önkéntes lehessen vagy ha már van fiókja akkor be tud jelentkezni.

Az alkalmazás lényege az, hogy az adatbázisból megjelenített kutyákat és macskákat virtuálisan, illetve általánosan örökbe lehessen fogadni. Ez nem jelenti azt, hogy így a felhasználó azonnal rendelkezhet 5 állattal is, csak hogy jelzi az alapítvány felé, hogy Ő igényt tart az állat/állatok örökbefogadására vagy ha virtuálisra nyomott, akkor jelzi, hogy ő támogatni fogja az állatot amíg alapítványi gondozás alatt áll.

A programok oldalon megtekinthetőek lesznek, mikor, melyik időpontban milyen programok vannak Ezekre lehet majd jelentkezni. A választott programokat megtekintheti a „Jelentkezéseim oldalon” és ha meggondolta magát a felhasználó, akkor akár ki is törölheti azokat a listájából.

Megtekintheti a nagyobb rendezvényeket, amelyeken részt vett az alapítvány.

Szintén képes lesz a bejelentkezett felhasználó módosítani az adatait.

## 6.2 Rendszerkövetelmény

Mielőtt elkezdjük használni az alkalmazást fontos, hogy fusson a szerver. Ehhez a XAMPP-ot kell elindítanunk

Abban az esetben, ha az alkalmazásunkat egy Android operációs rendszeren szeretnénk futtatni, akkor az API szintjének meg kell haladnia a 24- es szintet. Ha asztali gépen nézzük akkor ez a 7.0-as Nougat verziószám.

Fontos, megnézni telepítés előtt, hisz ezek az alkalmazások egyre többet tudnak és ha egy régebbi Android verzión próbáljuk megnézni, az alkalmazást nem tudja lefuttatni, és azonnal kilép. Azonban az is fontos, hogy legyen elég tárhely a telefonunkon, mert a jelenlegi alkalmazás több mint 20 MB és ez a továbbfejlesztések következtében gyarapodni fog. Csak olyan telefonon tudjuk teszteli / használni az alkalmazást, amely képes internetkapcsolatra, mert már a betöltőképernyőt követően a bejelentkezés igényelni fogja a Rest Api miatt. Az eszköz képernyőmérete is számít, hisz nagyobb képernyőn szét csúzhat, kisebb képernyőn pedig összecsúszik. Ezért ajánlom a minimum 5.0 maximum 7.0 colos átmérővel rendelkező eszközök használatát.

|  |  |
| --- | --- |
| A képen szöveg látható  Automatikusan generált leírás  37. ábra - Mobil nézet | A képen szöveg látható  Automatikusan generált leírás  38. ábra - Tablet nézet |

Én elsőnek egy Samsung Galaxy S6 Edge telefonon teszteltem, de amint beletettem a kezdőképernyőn található diavetítést, a telefon Android verziója nem bírta el és onnantól kezdve nem tudtam használni tesztelésre. Iskolában csapattársam telefonján teszteltem, ami egy Samsung Galaxy J3. Első alkalommal aggódtam, hogy ezen a telefonon se fog lefutni, mert hasonló életkorral rendelkezik az előbb említett telefonnal. De szerencsére tökéletesen funkcionált a kis képernyőtől eltekintve. Itthon pedig egy Samsung Galaxy A20e-vel teszteltem, ami teljesen megfelelt minden kritériumnak. Tökéletes méretű képernyője van, elég tárhely a frissítéseknek, megfelelő Android verzió és internetkapcsolatra képes. Szintén teszteltem emulátor telefonon és tableten. A telefon tökéletesen működött, de a tableten a nagy képernyőjéből kifolyólag jobban szétlettek húzva az elemek, mint én terveztem.

Összeségében törekedtem a reszponzívitás-ra, hogy ezzel is megkönnyítsem a felhasználó dolgát és egyel kevesebb dolognak kelljen megfelelnie.

Miután befejeztem az alkalmazásom fejlesztését és tudtam, hogy nem lesz benne több változás létrehoztam egy APK fájl.

## 6.3. A program telepítésének és konfigurálásának leírása

Ahhoz, hogy megtudjuk kezdeni az alkalmazás telepítését, egy olyan Android-os eszközre van szükségünk, amely megfelel az előzőleg ismertetett rendszerkövetelményeknek.

A már előzőleg ismertetett APK fájl egy olyan helyre másoljuk, ahonnan a mobiltelefonunkkal elérjük és letudjuk tölteni.

Letöltés követően megkeressük azt a mappát, ahol a letöltött fájlokat tárolja a telefonunk. Az APK fájlunkra rákattintunk es elő fog dobni egy olyan ablakot, ha elsőnek telepítünk ilyen kiterjesztésű fájlt, hogy a telefon nem engedélyezett idegen forrásból származó appok letöltésére.

A képen szöveg látható

Automatikusan generált leírásA képen szöveg látható

Automatikusan generált leírás

39. ábra - Idegen forrásból engedélyezés

. ábra - Ismeretlen fájl töltése nem engedélyezett

A Beállításokra rákattintva rávisz arra, hogy engedélyezzük az ismeretlen alkalmazások telepítését, ha szeretnénk máskor is Play Áruház nélkül telepíteni.

Ezután tudjuk megkezdeni az alkalmazás telepítését, ami kb fél percet vesz igénybe és máris használható lesz az app.

Abban az esetben ha máshonnan (pl:internetről) szeretnénk apk fájl-t letölteni, akkor a beállítások → ismeretlen alkalmazások telepítése oldalon a telefonunkon engedélyezni kell az adott oldalra vonatkozóan.

## 6.4 A program használatának részletes leírása

Amikor sikeres telepítés után megnyitjuk az alkalmazást egy mancs alakú betöltő képernyő jelenik meg előttünk, ezzel is szimbolizálva, hogy egy állatos alkalmazást nyitott meg a felhasználó. Ezt a képernyőt 3 másodpercig fogja látni a felhasználó.

Ennek a képernyő eltünését követően egy Bejelentkező oldal fog megnyílni. Itt két választási lehetőség áll fent.   
Abban az esetben, ha első látogatóként lép be az alkalmazásba, akkor regisztrálnia kell úgy, hogy a bejelentkező felületen található szövegre rákattint és az átfogja vinni a regisztrációs oldalra. Itt a következő adatok megadásával: *teljes név, felhasználónév, e-mail cím* és *jelszó*. Amíg ezeket a bemeneti mezőket nem tölti ki helyesen, addig az alkalmazás nem engedi a következő oldalra. Sikeres regisztrációt követően visszalép a bejelentkező oldalra, ahol be kell jelentkezni az előzőleg megadott felhasználónévvel és jelszóval.  
 Ha viszont már van fiókja akkor a kettő darab bemeneti mezőbe, meg kell adni a már előzőleg regisztrált felhasználónevet és a hozzá tartozó jelszót. Amíg valamelyik mező nincs helyesen kitöltve, az alkalmazás nem fogja tovább engedni a következő oldalra.

 Amint sikeresen bejelentkezett, megjelenik a kezdőképernyő. Itt az alapítvány nevét fogja találni, ezalatt pedig egy 3-3 db kutya és macska képeiből készült diavetítést, kedv csinálás szempontjából. Az oldal legtetején a zöld toolbar-ban található a három icon. Jobb oldalon 2, a bal oldalon 1.

41. ábra - Toolbar

A nyilat ábrázoló icon a kilépést jelöli. Erre rányomva megjelenik egy megerősítő ablak, hogy biztos szeretne-e kilépni. Ha nemmel válaszol akkor bent marad az alkalmazásban, ha viszont az igen gombot választja, akkor ki fog lépni az alkalmazásbő és újra be kell jelentkeznie, ha szeretne valamit csinálni az alkalmazáson belül.  
Az ember iconra rányomva megjelenik a bejelentkezett felhasználó alábbi adatai: teljes név, felhasználónév és email. A módosítás gombra kattintva mindezeket módosíthatja és ezeket a Mentés gombbal tudja elmenteni. Ha módosítás közbe rányom a vissza gombra akkor visszadobja, ahol megjelennek a saját adatai. 2. kattintással, pedig visszaviszi a felhasználót arra az oldalra, ahol az Alapítvány neve és a képek találhatóak.

|  |  |
| --- | --- |
| 42. ábra – Adatok megjelenítése | . ábra - Adatok szerkesztése |

A jobb oldali icon-ra rá se kell nyomni, hanem el lehet húzni a képernyőt balról jobbra és megjelenik a kis fülecske (navigationdrawer), ahonnan a különböző oldalakra lehet rámenni.

A Kutyák vagy Macskák oldalra rákattintva, megjelenik az összes állat neve, (születési dátuma) és egy részletes leírás az állatról. A kártyákban lesz egy gomb, ami az *Örökbefogadás* felirattal van ellátva. Erre rákattintva kiválaszthatjuk, hogy virtuálisan vagy általánosan A képen szöveg látható

Automatikusan generált leírásszeretnénk örökbe fogadni az adott állatot. Ha rányom, bekerül az adatbázisunkba, és további teendője nincs, ha viszont nem választ akkor visszaugrik arra a képernyőre, ahol kivannak listázva az állatok.

. ábra - Örökbefogadás alert

A Programok oldalon az előre egy héttel meghirdetett programokra lehet jelentkezni. Amint megjelenik a Sikeres Jelentkezés felirat a „Jelentkezésim” fül alatt megtalálható lesz az összes program amire jelentkezett a felhasználó. Ha valamelyik program mégsem megfelelő, akkor ki tudja törölni a kuka iconra kattintva

A képen szöveg látható

Automatikusan generált leírás

. ábra - Programok

Rendezvények oldal arra szolgál, hogy ha a felhasználó érdeklődik mikor- milyen nagyobb eseményeken képviselte magát az alapítvány és erről egy rövid leírás.

Ha mindennel végzett a felhasználó, amivel szeretett volna csinálni az alkalmazásban, akkor a már előzőleg említett kilépés gombra rákattintva kiléphet az alkalmazásból. Abban az esetben, ha a felhasználó gomb nélkül lép ki, akkor is kijelentkezteti a felhasználót.

# 7. Tesztelési dokumentáció

**Bejelentkezés helyes adatokkal**

|  |  |  |  |
| --- | --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény | |
| Felhasználónév: Nbarbi11  Jelszó: barbi1234 | Ezeket az adatokat megadásával, akkor a Bejelentkezés gombra kattintva, kapunk egy tokent és megjelenik a „Sikeres bejelentkezés” felirat | A képen szöveg, névjegykártya, képernyőkép látható  Automatikusan generált leírás |  |

**Bejelentkezés helytelen adatokkal**

|  |  |  |
| --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény |
| Felhasználónév: Nbarbi1234  Jelszó: barbi11 | Bejelentkezés gombra kattintva megjelenik, hogy „Helytelen felhasználónév vagy jelszó” | A képen szöveg látható  Automatikusan generált leírás |

**Bejelentkezési próba üres adatokkal**

|  |  |  |  |
| --- | --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény | |
| Felhasználónév:  Jelszó: | Bejelentkezés gombra kattintva megjelenik, hogy kötelező a felhasználónév és hogy „Helytelen felhasználónév vagy jelszó” | A képen szöveg, képernyőkép, névjegykártya látható  Automatikusan generált leírás | A képen szöveg látható  Automatikusan generált leírás |

**Regisztráció helyes adatokkal**

|  |  |  |  |
| --- | --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény | |
| Teljes név: Nádasdi Barbara  Felhasználónév: Nbarbi1234  Email: [barbi1234@gmail.com](mailto:barbi1234@gmail.com) Jelszó: Nbarbara12 | Regisztráció gombra kattintva kiírja nekünk az adatokat JSON formába és visszalép a bejelentkező felületre | A képen szöveg látható  Automatikusan generált leírás | A képen szöveg, képernyőkép, névjegykártya látható  Automatikusan generált leírás |

**Regisztráció helytelen/üres adatokkal**

|  |  |  |  |
| --- | --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény | |
| Teljes név: Nádasdi Barbara  Felhasználónév:  Email: [barbi34@gmail.com](mailto:barbi34@gmail.com) Jelszó: Nbarbara12 | Regisztráció gombra kattintva hibaüzenetet ír ki |  |  |

**Örökbefogadás**

|  |  |  |
| --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény |
| Örökbefogadás gomb → általános örökbefogadás/virtuális | Sikeresen örökbe fogadta | A képen szöveg látható  Automatikusan generált leírás |

**Örökbefogadás örökbefogadott állat**

|  |  |  |
| --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény |
| Örökbefogadás gomb → általános örökbefogadás/virtuális | A kiválasztott állat már örökbe van fogadva | A képen szöveg látható  Automatikusan generált leírás |

**Profil módosítás**

|  |  |  |
| --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény |
| Előre betöltött adatok  Teljes név:  Nádasdi Barbara Teszt  Többi adat változatlan | Sikeres módosítás | A képen szöveg látható  Automatikusan generált leírás |

**Program jelentkezés után bekerül a jelentkezéseim oldalra**

|  |  |  |
| --- | --- | --- |
| Bemenet | Elvárt eredmény | Valós eredmény |
| Jelentkezés az első programra | Jelentkezés oldalon megjelenés és lehetőség a törlésre |  |

# 8. Továbbfejlesztési lehetőségek

* Abban az esetben, ha a felhasználó nem az alkalmazáson belüli kilépő gombbal lépett ki, akkor jegyezze meg a felhasználót
* Amelyik programra jelentkezett a felhasználó, kapjon róla értesítést egy nappal korábban
* Állatoknál legördülő menüben megjeleníteni az összes adatát, mindezek alatt a gomb, amivel örökbe lehet fogadni
* A programjelentkezés bekerüljön a felhasználó személyes naptárába
* Ha a felhasználó virtuálisan fogadta örökbe az adott állatot, akkor beállíthassa milyen gyakran szeretne értesítést kapni, hogy támogassa az állatot valamilyen formában
* Közvetlenül lehessen felhívni az alapítvány segélyvonalát
* Szintén lehessen az applikációból közvetlenül utalni pénzt az alkalmazás bankszámlaszámára plusz kezelési költség levonása nélkül
* A felhasználó a kutyasétáltatás helyszínén az alkalmazásba beolvasson egy QR kódot, ahol le kell adni, melyik kutyát vitte el. Amikor visszahozza, szintén lesz egy QR kód, amivel jelzi felénk, hogy a helyén van a kutya és mikor hozta vissza. Mind két esetben megjelenik az aktuális dátum
* Ne csak a Facebook oldalon lehessen jelezni, ha valaki szeretné egyes tárgyakkal támogatni az alapítványt, hanem létrehozni egy Támogatói Webshopot, ahol egy helyen megtalálható lesz az összes termék

# 9. Összegzés

## 9.1 Magyarul

Az alkalmazásunk egy létező alapítványt vett alapul, ahonnan kutyákat, illetve macskákat lehet örökbe fogadni virtuálisan és általánosan is. Szintén lehet a meghirdetett programokra jelentkezni. A többi oldal pedig ismeretközlés.

Köztudott, hogy a mobil alkalmazás nagyon közel áll weboldalakhoz, csak kisebb képernyőkre van optimalizálva és amikor év elején ki kellett választani, hogy a csapatból ki melyik vékonykliens-et(fronted) fogja választani, bátran választottam a mobil fejlesztését, mert már az előző tanévben nagyon vártam, hogy végre tanulhassunk mobil fejlesztést.

Legfőbb célom a fejlesztés elkezdésekor az volt, hogy egy olyan alkalmazást hozzak létre, ami felhasználóbarát és könnyen lehet továbbfejleszteni annak érdekében, hogy ha az alapítvány szeretné igénybe venni. Szerintem ezt sikerült is elérnem több – kevesebb sikerrel.

Ez egy olyan alkalmazás lett, amin a későbbiekben több lehetőség is áll fent továbbfejleszteni. Nagyon az alap dolgokat tartalmazza (regisztráció, bejelentkezés, örökbefogadás, programokra jelentkezés, profil módosítás, jelentkezés törlése) kezdetleges módon megoldva és nem tartalmaz sok információt maga az alapítványról. De a kinézetével teljesen elégedett vagyok.

Ha nem tudtam valamit, nem siettem azonnal a tanárokhoz segítségért, hanem megpróbáltam magam megoldani az internet és az eddigi ismereteim segítségével. Több napot áldoztam annak, hogy videók alapján próbáltam egyes funkciókat megcsinálni, de ezek közül a legtöbb kudarcba fulladt.

Az alkalmazásom fejlesztése közben számos problémába ütköztem. Ezek közül kiemelendő, hogy nagyon nehezen értettem meg, hogy tud a mobil összekapcsolódni a Laravel backendünkkel főleg, ha egy oldalon belül több táblára is hivatkoznom kell. Emiatt rosszul osztottam be az időmet, és majdnem kicsúsztam az időből.  
Szintén nehézséget okozott az, hogy Fragment-ekkel kellett dolgoznom a leggyakrabban használt Activity-k helyett mert például másképp kell hivatkozni egy-egy meghívott eseményindítóra.

A mi csapatunkban is számos probléma volt. Nem mindig értettünk egyet, mit hogyan kéne kiviteleznünk, de mivel a 13.osztályba is már együtt csináltuk a csoportos feladatokat volt némi tapasztalatunk a helyzetek kezelésére.

Sok nehézség ellenére sem hátráltam meg ezért összeségében elégedett vagyok az alkalmazásommal és örülök, hogy új technológiákat ismerhettem meg, amelyeket későbbi tanulmányaim és munkám során hasznát vehetem. Köszönettel tartozom Budaházi Bencének és Merényi Miklósnak, akik szakmai tudásukkal járultak hozzá a fejlődésemhez és a munkám megkönnyítéséhez, illetve hálás vagyok, hogy egy ilyen összetett fejlesztésben lehetőségem nyílt részt venni.

## 9.2 In English

Our application is based on a real foundation. Dogs and cats cared for here can be adopted virtually and generally. It will be also possible to apply for the advertised programs. It can be also modify personal data. Other pages are for informational purposes only.

It is well known that the mobile application is very close to websites, just it is only optimized for smaller screens and when I had to make a decision which frontend I would like to work on, I happily chose to develop a mobile application because i was waiting to finally learn android development since last year.

My main goal was when I started the development to create an application that is user friendly, and it can be easily upgraded in order for the foundation to use it. I think I managed to do that. However, when I didn’t know something, I didn’t rush right away to the teachers for help, but instead I tried to solve it on my own using the internet and my previous knowledge.

I encountered many difficulties while I developed my program. This often hold back my progress. It was very difficult for me to understand that how can I connect my application developed in Android Studio to our Laravel backend, especially if I have to reference multiple tables within a page.

Another difficulty was that I was working with fragments instead of Activities. It was harder because events works more challenging.

As with any team, there were difficulties here too. We didn’t always agree, but as a time passed on we were able to work more and more capable as a team, mainly because when we were still in 13th grade we did the year-end project together and we had the experience to handle it when we didn’t agree.

Despite many difficulties, I was keep going not thinking about giving up to make what I wanted to achieve with my application.

I am grateful to Bence Budaházi and Miklós Merényi, who contributed their professional knowledge to help my development and facilitated my work, and I am grateful to had the opportunity to participate in such a complex project.

# Ábrajegyzék

[1. ábra - Logó 7](#_Toc101734398)

[2. ábra - Roadmap 8](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734399)

[3. ábra - Kezdetleges adatbázis szerkezet 10](#_Toc101734400)

[4. ábra - Kezdetleges UML diagramm 10](#_Toc101734401)

[5. ábra - Végleges adatszerkezet 11](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734402)

[6. ábra - IntelliJ IDEA 21](#_Toc101734403)

[7. ábra - Scene Builder 21](#_Toc101734404)

[8. ábra - Java programnyelv 21](#_Toc101734405)

[9. ábra - Szűrés 24](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734406)

[10. ábra - Kiválasztás után 25](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734407)

[11. ábra - Kiválasztás elött 25](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734408)

[12. ábra - Hibaüzenet 27](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734409)

[13. ábra - Visszajelzés 27](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734410)

[14. ábra - Bejelentkezési hiba 31](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734411)

[15. ábra - Saját jogosultság módosításakor felrúgó üzenet 32](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734412)

[16. ábra - Jogosultság módosítás megerősítése 32](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734413)

[17. ábra - Bejelentkezési felület 36](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734414)

[18. ábra - Admin felület belépés után 36](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734415)

[19. ábra - jogosultság módosító gombok elérhetőek 37](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734416)

[20. ábra - Super Admin felület belépés után 37](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734417)

[21. ábra - módosítás megerősítése 38](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734418)

[22. ábra - művelet sikeressége 38](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734419)

[23. ábra - Macska Örökbefogadás kiválasztása 39](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734420)

[24. ábra - Örökbefogadási adatlap 39](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734421)

[25. ábra - Örökbefogadási visszajelzés 40](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734422)

[26. ábra - Oldalváltási lehetőségek 40](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734423)

[27. ábra - Folyamatábra 43](#_Toc101734424)

[28. ábra - Natív PHP alapkinézet 45](#_Toc101734425)

[29. ábra - Laravel keretrendszeres végeredmény 45](#_Toc101734426)

[30. ábra - Visual Studio Code logo 48](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734427)

[31. ábra - XAMPP felület 48](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734428)

[32. ábra - Adomány 50](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734429)

[33. ábra - Footer 51](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734430)

[34. ábra - Navbar 51](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734431)

[35. ábra - Hibaüzenet 52](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734432)

[36. ábra - Példa a kép feltöltése hibára 58](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734433)

[37. ábra - Mobil nézet 72](#_Toc101734434)

[38. ábra - Tablet nézet 72](#_Toc101734435)

[39. ábra - Idegen forrásból engedélyezés 73](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734436)

[40. ábra - Ismeretlen fájl töltése nem engedélyezett 73](#_Toc101734437)

[41. ábra - Toolbar 74](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734438)

[42. ábra – Adatok megjelenítése 74](#_Toc101734439)

[43. ábra - Adatok szerkesztése 74](#_Toc101734440)

[44. ábra - Örökbefogadás alert 75](file:///C:\Users\nadas\Documents\VIZSGAREMEK\BACKEND\Élethang_dokumentáció.docx#_Toc101734441)

[45. ábra - Programok 75](#_Toc101734442)

# Forrásjegyzék

* Adatok és képekhez használt
* (2021.09.10. - 2022.04.24)
  + <https://www.facebook.com/Elethangalapitvany>
  + <https://www.elethangalapitvany.hu/>
* Backend
* (2021.11.01. - 2022.04.24)
  + <https://laravel.com/>
  + <https://stackoverflow.com/>
* Asztali Alkalmazás
* (2022.02.10. - 2022.04.24)
  + <https://openjfx.io/>
  + <https://docs.oracle.com/en/java/>
  + <https://stackoverflow.com/>
  + <https://www.youtube.com/>
  + <https://www.iconfinder.com/>
  + <https://www.subpng.com/>
  + <https://icons8.com/>
* Weblap
* (2021.10.01. - 2022.04.24)
  + [Ikonok](https://freeicons.io/)
  + [Kép megjelenítése](https://morioh.com/p/1bdefda2a2d2)
* Mobil alkalmazás
* (2022.03.01. - 2022.04.24)
  + <https://freeicons.io/>
  + <https://balsamiq.com/>
  + <https://developer.android.com/docs>
  + <https://github.com/>
  + <https://www.youtube.com/>
  + <https://lottiefiles.com/>
  + <https://medium.com/android-beginners/cardview-android-example-beginners-dde933585261>

1. *static: olyan kulcsszó, amely az adattagot bármely objektumból elérhetővé teszi.*

   *final: olyan kulcsszó, amely az adattagot véglegessé teszi, tehát ezt sehogy se lehet módosítani* [↑](#footnote-ref-1)