Fogadj-örökbe

Fejlesztői dokumentáció

2022.04.14

Készítette: Madarász Dávid és Lehoczki Patrícia

Cél

Egy állat nyilvántartás készítése, amely biztosítja a felhasználó számára az állatok megtekintését, örökbefogadását fajok szerint csoportosítva.

Az applikáció azért jött létre, hogy egy könnyen kezelhető, informatív felületen keresztül megkönnyítse a házi kedvencek örökbefogadásának folyamatát. Célunk a felhasználók és a menhelyek közötti kommunikáció leegyszerűsítése, minden fellelhető információ összegyűjtése az örökbefogadásról, minden információ megszerzése a felhasználótól, még az örökbefogadás előtt, tisztázva a leendő tulajdonosok felé az elvárásokat.

A backend-t Laravel php keretrendszer, az adatbázist MariaDB szerver biztosítja.
Frontendnek egy webes alkalmazás áll rendelkezésre Angular keretrendszerben.
A webes felületen kezelhetők az állatok,fajok,fajták.

Felhasznált technológiák

Backend

- Laravel API (PHP keretrendszer) -
- laravel - sanctum (Autentikációs kiegészítő csomag)
- Mariadb server (Adatbázis kiszolgáló)

Készítéshez használt programok

- Dia
- Visual Studio Code 1.66.0
- Insomnia 2021.5.3
- Mariadb Server
- phpmyadmin

Frontend

- Angular (Typescript alapú webalkalmazás keretrendszer)
- Bootstrap 5.1.3 (CSS-keretrendszer)

Kódolási konvenciók

A kódot git verziókezelővel használjuk.

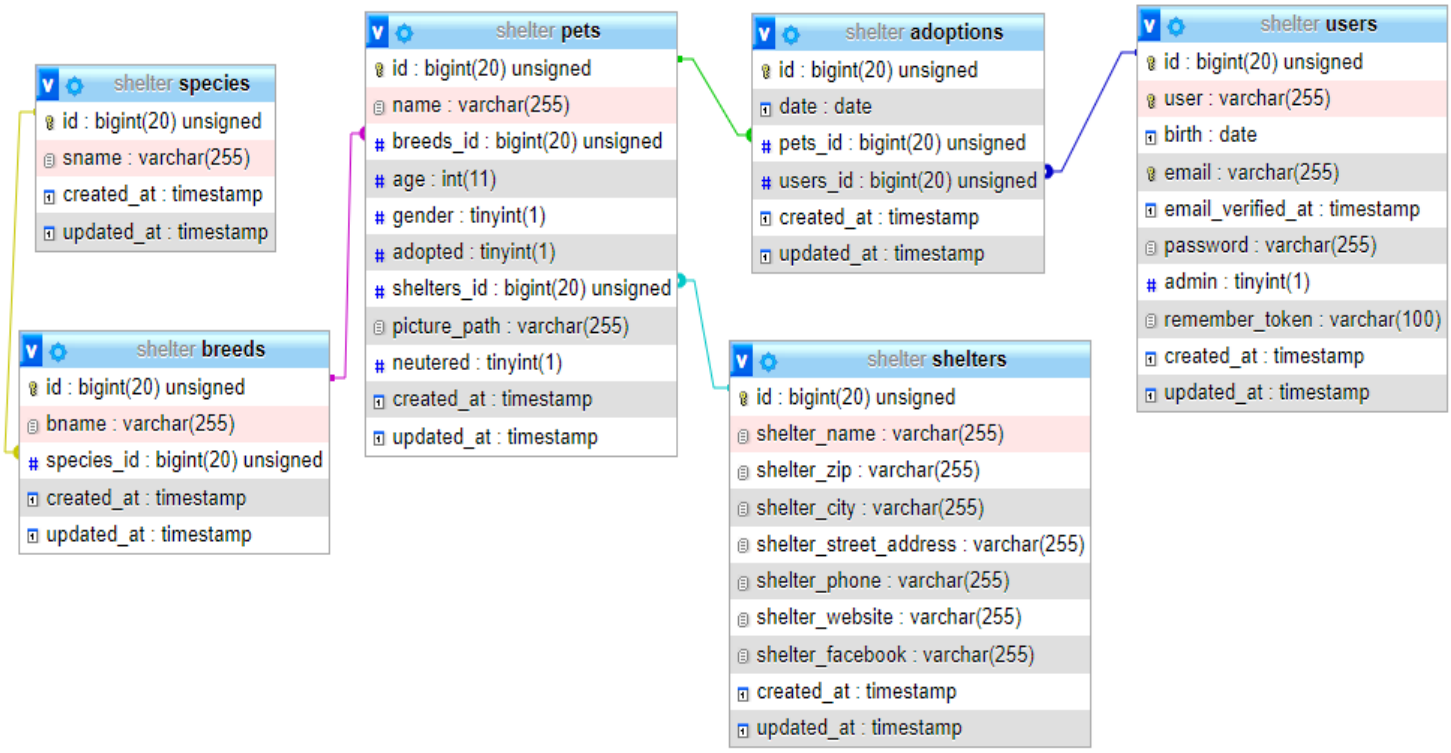
Alapkönyvtárak

- api
- database
- doc
- web

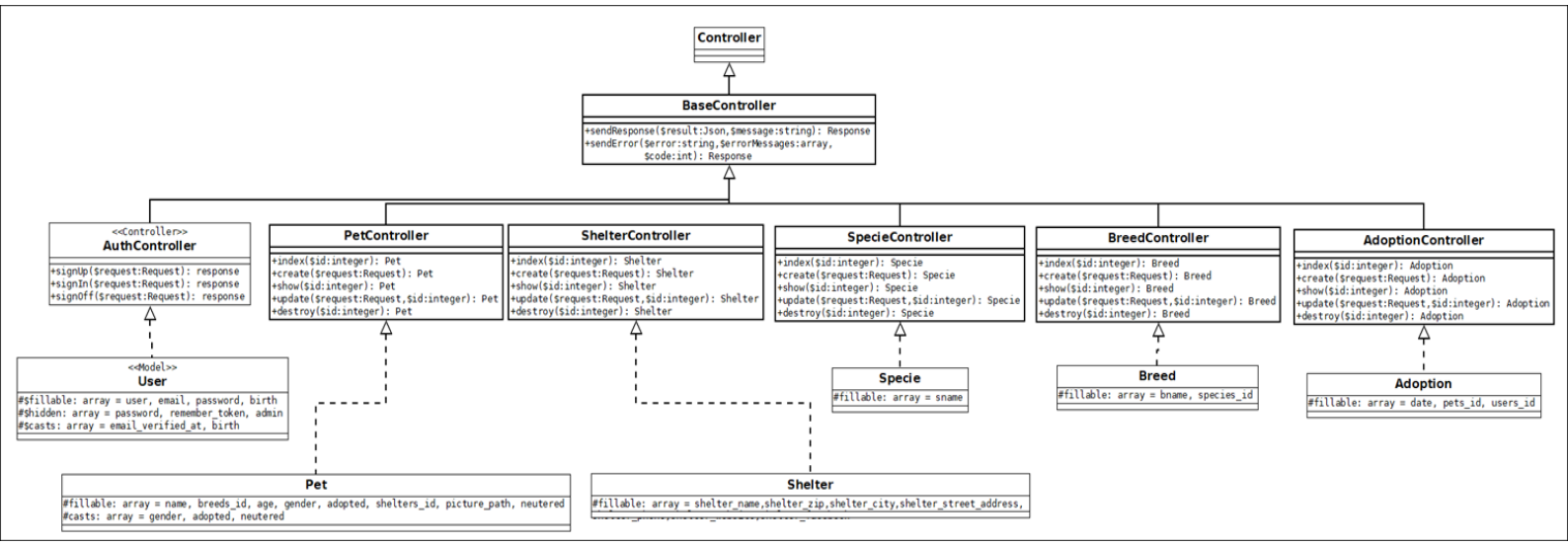
Az api backend, a web a webes frontend.

Felülettervek

Adatmodell



API UML terv



Végpontok

Végpont	Metódus	Azonosítás	Leírás
/pets	get	nem	Állatok lekérdezése
/pets	post	igen	Állat felvétele
/pets/{id}	get	igen	Állat lekérdezése
/pets/{id}	*put*	igen	Állat szerkesztése
/pets/{id}	*delete*	igen	Állat törlése
/species	get	nem	Fajok lekérdezése
/species/*{id}*	get	nem	Faj lekérdezése
/species	post	igen	Faj felvétele
/species/{id}	put	igen	Faj szerkesztése
/species/{id}	delete	igen	Faj törlése
/breeds	get	nem	Fajták lekérdezése
/breeds/{id}	get	nem	Fajta lekérdezése
/breeds	post	igen	Fajta felvétele
/breeds/{id}	put	igen	Fajta módosítása
/breeds/{id}	delete	igen	Fajta törlése
/shelters	get	nem	Menhelyek lekérdezése
/shelters/{id}	get	nem	Menhely lekérdezése
/shelters/	post	igen	Menhely felvétele
/shelters/{id}	put	igen	Menhely szerkesztése
/shelters/{id}	delete	igen	Menhely törlése
/adoptions	get	nem	Örökbefogadások lekérdezése
/adoptions/{id}	get	nem	Örökbefogadás lekérdezése
/adoptions	post	igen	Örökbefogadás felvétele
/adoptions/{id}	put	igen	Örökbefogadás szerkesztése
/adoptions/{id}	delete	igen	Örökbefogadás törlése
/register	post	nem	Felhasználó regisztrációja
/login	post	nem	felhasználó bejelentkezése

/logout	post	igen	felhasználó kijelentkezése

Back-end

Általános működés:

A REST API http kéréseket fogad, melyek tartalmazzák a műveletekhez szükséges megfelelő adatokat. A kényes műveletek végpontjai védettek, autentikáció szükséges a használatukhoz. Ilyenek a kijelentkezés, az új állat felvétele, állat adatainak módosítása, állat törlése, új faj/fajta/örökbefogadás/menhely felvétele, módosítása, törlése.
A regisztráció, a bejelentkezés, a állatok,fajok,fajták,menhelyek,örökbefogadások lekérdezésének végpontjai publikusak, ezek használatához autentikáció nem szükséges.
Az adatokat Json formátumban fogadja és feldolgozza. A vezérlést kontrollerek valósítják meg, minden adatkezelési csoportnak (autentikáció, állat, faj,fajta,menhely,örökbefogadás) külön kontrollere van, itt történik az adatfeldolgozás.
A kontrollerek modellekkel vannak kapcsolatban amelyek az adatkezelésért felelősek. Minden adatkezelési csoportnak (felhasználó, állat, faj,fajta,menhely,örökbefogadás) külön modellje van, itt történik az adatok adatbázisból kiolvasása, illetve a az adatok kiírása adatbázisba.
A modellek adatbázis táblákkal vannak kapcsolatban, melyek az adatok tárolásáért felelősek. Az adatbázis táblák adatait a modellek kezelik.

Osztályok

BaseController

Feladata a bejelentkezési műveletek válaszainak küldése. Sikeres művelet esetén a generált azonosító tokent visszaküldi, sikertelen művelet esetén hibaüzenetet küld.
Metódusok:
sendResponse()
- bejövő paraméterek: $result (A generált token), $message (A felhasználónak szánt saját üzenet)
- kimenő adatok: Response (sikeres bejelentkezés esetén a token).

sendError()
-bejövő paraméterek: $error (A php által generált hibaüzenet), $errorMessages (Saját hibaüzenet), $code (A válaszban küldendő http kód).
-kimenő adatok: Response (sikertelen bejelentkezés esetén hibaüzenet és a hiba kódja).

AuthController

Feladata egy új felhasználó felvétele, felhasználók autentikációja, felhasználók kijelentkeztetése.
Kiterjeszti a BaseController osztályt.
Metódusok:
register()
- bejövő paraméterek: $request (az regisztrációhoz szükséges adatok a kérésben, user, email, password, confirm_password)
    Az adatokat validálja, majd sikeres érvényesítés után bejegyzi az users adatbázis tábla megfelelő mezőibe.
- kimenő adatok: saját üzenet.

login()
Feladata a felhasználó azonosítása név és jelszó alapján. Sikeres autentikáció esetén generál egy tokent a felhasználó számára és bejegyzi a personal_access_tokens adatbázis tábla megfelelő mezőjébe, majd átadja a BaseController sendResponse() metódusának a saját üzenettel együtt.
- bejövő paraméterek: $request (a bejelentkezéshez szükséges adatok a kérésben, user, password).
- kimenő adatok: token, saját üzenet, admin.

logout()
Feladata a felhasználó kijelentkeztetése és a token törlése az adatbázis táblából.
- bejövő paraméterek: $request (a kijelentkezéshez szükségek adatok a kérésben, token)
- kimenő adatok: saját üzenet.

PetController

Feladata: állatok adatainak lekérése,egy új állat felvétele, meglévő állat adatainak módosítása, meglévő állat törlése, egy adott állat adatainak lekérése.
Kiterjeszti a BaseController osztályt.
Metódusok:

index()
Visszaadja az adatbázisban szereplő állatok listáját.
- bejövő paraméterek: NINCS
- kimenő adatai: Pet (Pet modell amely tartalmazza az állatok adatbázisból lekért adatait.

create()
Eltárol az adatbázisban egy új állatot.
- bejövő paraméterek: request (A állat adatait tartalmazó kérés).
- kimenő adatok: Pet (A állat adatait tartalmazó modell).

show()
Visszaadja egy kiválasztott állat adatait.
- bejövő paraméterek: $id (a kiválasztott állat azonosítója)
- kimenő adatok: Pet (a kiválasztott állat adatait tartalmazó modell).

update()
Frissíti az adatbázisban tárolt egy állat adatait.
- bejövő paraméterek: $request, $id (az állat adatait tartalmazó kérés és a állat azonosítója).
- kimenő adatok: Pet (a állat adatait tartalmazó modell).

delete()
Feladata a kiválasztott állat törlése az adatbázisból.
- bejövő paraméterek: $id (a kiválasztott állat azonosítója).
- kimenő adatok: Pet (a kiválasztott állat adatait tartalmazó modell).

SpecieController

Feladata a fajok kezelése, új faj felvétele, faj módosítása, faj törlése.
Kiterjeszti a BaseController osztályt.
Metódusok:
index()
Visszaadja az adatbázisban szereplő fajok listáját.
- bejövő paraméterek: NINCS
- kimenő adatai: Specie (Specie modell amely tartalmazza a fajok adatbázisból lekért adatait.

create()
Eltárol az adatbázisban egy új fajt.
- bejövő paraméterek: request (A faj nevét tartalmazó kérés).
- kimenő adatok: Specie (A létrehozott faj adatai, üzenet).
show()
Visszaadja egy kiválasztott faj adatait.
- bejövő paraméterek: $id (a kiválasztott faj azonosítója)
- kimenő adatok: Specie (a kiválasztott faj adatait tartalmazó modell).

update()
Frissíti az adatbázisban tárolt egy faj nevét.
- bejövő paraméterek: $request, $id (a faj nevét tartalmazó kérés és a faj azonosítója).
- kimenő adatok: Specie (üzenet a sikeres vagy sikertelen frissítésről).

delete()
Feladata a kiválasztott faj törlése az adatbázisból.
- bejövő paraméterek: $id (a kiválasztott faj azonosítója).
- kimenő adatok: Specie (üzenet a sikeres vagy sikertelen törlésről.).

BreedController

Feladata a fajták kezelése, új fajta felvétele, fajta módosítása, fajta törlése.
Kiterjeszti a BaseController osztályt.
Metódusok:
index()
Visszaadja az adatbázisban szereplő fajták listáját.
- bejövő paraméterek: NINCS
- kimenő adatai: Breed (Breed modell amely tartalmazza a fajták adatbázisból lekért adatait.

create()
Eltárol az adatbázisban egy új fajt.
- bejövő paraméterek: request (A fajta nevét,faj azonosítóját tartalmazó kérés).
- kimenő adatok: Breed (A létrehozott fajta adatai, üzenet).

show()
Visszaadja egy kiválasztott faj adatait.
- bejövő paraméterek: $id (a kiválasztott faj azonosítója)
- kimenő adatok: Breed (a kiválasztott fajta adatait tartalmazó modell).

update()
Frissíti az adatbázisban tárolt egy faj nevét.
- bejövő paraméterek: $request, $id (a fajta azonosítóját,fajta nevét, faj azonosítóját tartalmazó kérés).
- kimenő adatok: Breed (üzenet a sikeres vagy sikertelen frissítésről).

delete()
Feladata a kiválasztott fajta törlése az adatbázisból.
- bejövő paraméterek: $id (a kiválasztott faj azonosítója).
- kimenő adatok: Breed (üzenet a sikeres vagy sikertelen törlésről).

ShelterController

Feladata a menhelyek kezelése, új menhely felvétele, menhely módosítása, menhely törlése.
Kiterjeszti a BaseController osztályt.
Metódusok:
index()
Visszaadja az adatbázisban szereplő menhelyek listáját.
- bejövő paraméterek: NINCS
- kimenő adatai: Shelter (Shelter modell amely tartalmazza a fajták adatbázisból lekért adatait.)

create()
Eltárol az adatbázisban egy új menhelyet.
- bejövő paraméterek: request (A menhely nevét,irányítószámát,városát,címét,telefonszámát,weboldalát, facebook oldalát tartalmazó kérés).
- kötelező adatok: menhely neve
- kimenő adatok: Shelter (A létrehozott menhely adatai, üzenet).

show()
Visszaadja egy kiválasztott menhely adatait.
- bejövő paraméterek: $id (a kiválasztott menhely azonosítója)
- kimenő adatok: Shelter (a kiválasztott menhely adatait tartalmazó modell).

update()
Frissíti az adatbázisban tárolt egy menhely adatait.
- bejövő paraméterek: $request, $id (a menhely adatait kérés).
- kimenő adatok: Shelter (üzenet a sikeres vagy sikertelen frissítésről).

delete()
Feladata a kiválasztott menhely törlése az adatbázisból.
- bejövő paraméterek: $id (a kiválasztott menhely azonosítója).
- kimenő adatok: Shelter (üzenet a sikeres vagy sikertelen törlésről).

AdoptionController

Feladata az örökbefogadások kezelése, új örökbefogadás felvétele, örökbefogadás módosítása, örökbefogadás törlése.
Kiterjeszti a BaseController osztályt.
Metódusok:
index()
Visszaadja az adatbázisban szereplő örökbefogadások listáját.
- bejövő paraméterek: NINCS
- kimenő adatai: Adoption (Adoption modell amely tartalmazza a fajták adatbázisból lekért adatait.)

create()
Eltárol az adatbázisban egy új örökbefogadást.
- bejövő paraméterek: request (Az örökbefogadás dátumát,a felhasználót és az örökbefogadott állat azonosítóját tartalmazó kérés).
- kötelező adatok: dátum,felhasználó azonosítója,állat azonosítója
- kimenő adatok: Adoption (A létrehozott örökbefogadás adatai, üzenet).

show()
Visszaadja egy kiválasztott örökbefogadás adatait.
- bejövő paraméterek: $id (a kiválasztott örökbefogadás azonosítója)
- kimenő adatok: Adoption (a kiválasztott örökbefogadás adatait tartalmazó modell).

update()
Frissíti az adatbázisban tárolt egy örökbefogadás adatait.
- bejövő paraméterek: $request, $id (a örökbefogadás adatait kérés).
- kimenő adatok: Adoption (üzenet a sikeres vagy sikertelen frissítésről).

delete()
Feladata a kiválasztott örökbefogadás törlése az adatbázisból.
- bejövő paraméterek: $id (a kiválasztott örökbefogadás azonosítója).
- kimenő adatok: Adoption (üzenet a sikeres vagy sikertelen törlésről).

User

A regisztrált felhasználók adatait, illetve az azonosító tokent amelynek a generálása a felhasználóhoz kapcsolódik írja az adatbázis megfelelő táblájába, valamint onnan szolgáltatja szükség szerint. A User modell a users adatbázis táblával van kapcsolatban az adatokat onnan olvassa és oda írja. Az osztályt a Laravel projekt generálja, használata a Sanctum autentikációs kiegészítő csomagon keresztül történik.

Pet

Az adatbázisban felvett állatok adatait kezeli, a PetController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki.
Az adatbázis pets táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.
Mezők:
$fillable (array) elemei: name, breeds_id, age, gender, adopted, shelters_id, picture_path, neutered
A fillable mező elemei egy – egy mező a pets adatbázis táblában.
Csak a felsorolt mezőket írhatja a modell az adatbázisban.
$casts (array) elemei: gender, adopted, neutered
A casts mező elemei határozzák meg, hogy az adott mező milyen típusú.

Adoption

Az adatbázisban felvett örökbefogadások adatait kezeli, az AdoptionController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki.
Az adatbázis adoptions táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.
Mezők:
$fillable (array) elemei: date, pets_id, users_id
A fillable mező elemei egy – egy mező az adoptions táblában.
Csak a felsorolt mezőket írhatja a modell az adatbázisban.

Breed

Az adatbázisban felvett fajták adatait kezeli, a BreedController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki.
Az adatbázis breeds táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.
Mezők:
$fillable (array) elemei: bname,species_id
A fillable mező elemei egy – egy mező az breeds táblában.
Csak a felsorolt mezőket írhatja a modell az adatbázisban.

Specie

Az adatbázisban felvett fajok adatait kezeli, a SpecieController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki.
Az adatbázis breeds táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.
Mezők:
$fillable (array) elemei: sname
A fillable mező elemei egy – egy mező az species táblában.
Csak a felsorolt mezőket írhatja a modell az adatbázisban.

Shelter

Az adatbázisban felvett menhelyek adatait kezeli, a ShelterController megfelelő metódusai által kért adatot adja vissza az adatbázis megfelelő táblájából , illetve a kontrollerből érkező adatokat írja ki.
Az adatbázis shelters táblájával van kapcsolatban, az adatokat onnan olvassa és oda írja.
Mezők:
$fillable (array) elemei: shelter_name,shelter_zip,shelter_city,shelter_street_address, shelter_phone,shelter_website,shelter_facebook
A fillable mező elemei egy – egy mező az shelters táblában.
Csak a felsorolt mezőket írhatja a modell az adatbázisban.

Front-end

Általános működés:

A program a http kéréseket küldd a shelter nevű REST APInak, amik tartalmazzák a különböző műveletekhez szükséges adatokat. Az apival való kommunikáció által éri el a phpmyadmin felületen kezelt mariadb adatbázist. Az autentikációhoz kötött műveleteket/végpontokat (ilyen az új állat felvétele, módosítása, törlése, állat örökbefogadása) a front-end oldalon is egyaránt védjük authGuard szolgáltatással. Az állatok listázása nem védett művelet, így ezen végpontok publikusak. A kérésekhez az Angular http client osztályát használjuk. A service-ok végzik a http kérések küldését, ezek meghívása után a válaszokat pedig az adott oldal ts kiterjesztésű fájljaiban dolgozza fel a program.

Front-End mappaszerkezet

- web/
    - fogadj-orokbe/
        - src/
            - favicon.ico
                - Az oldal iconja
            - index.html
                - Html fejléc beállítása
            - app/
                - admin/
                    - Ebben a mappában az admin felhasználó esetén megjelenő Hozzáadás menüponton belül elérhető beállítások vannak. Ez az oldal listázza az adatbázisban szereplő összes állatot, lehetséges állatot hozzáadni az adatbázishoz, módosítani és törölni
                - cats/
                    - Ezen mappa alatt találhatóak a Macskák menüpont beállításai. Ez az oldal listázza az adatbázisban lévő örökbefogadható házi macska fajú (species) állatokat, bejelentkezés nélkül megtekinteni lehet őket, sikeres bejelentkezés után az Örökbefogadom! gombbal adoptálni lehet az adott állatot.
                - criteria/
                    - Ebben a mappában található a Tudnivalók dropdown menü Örökbefogadási feltételek oldal beállításai, amelyek ismertetik a felhasználóval az általános örökbefogadási feltételeket
                - dogs/
                    - Ebben a mappában a Kutyák oldal beállításai találhatóak meg. Funkciójában megegyezik a Macskák oldaléval.
                - main/

        - A főoldal beállításai.
    - nopage/
        - Rosszul beírt útvonal esetén „Az oldal nem található” üzenetet megjelenítő oldal
    - register/
        - A Bejelentkezés dropdown menüponton keresztül elérhető oldal, amelyen keresztül regisztrálni lehet.
    - shared/
        - Az összes service ebben a mappában található
    - shelters/
        - A Tudnivalók dropdown „Hol vannak menhelyek?” oldalának beállításai. Megjeleníti az adatbázisban található összes menhelyet, szűrni pontos irányítószám beírásával lehetséges.
    - toknow/
        - A Tudnivalók dropdown, valamint a „Tippek, tanácsok” oldal beállításai. Kérdéseket és válaszokat tartalmaz az oldal
    - app.component.html
        - Ebben a html-ben és a hasonló nevű CSS fájlban találhatóak a menü (így a login is) és a footer beállításai.
    - app.component.ts
        - Ezen fájlon keresztül történik a bejelentkezés és a kijelentkezés, itt történik a bejelentkezett felhasználó admin jogosultságának ellenőrzése, bejelentkezés állapotának vizsgálata. Ezek alapján jelenik meg a „Bejelentkezés” vagy a „Kijelentkezés” feliratú gomb.
    - app.module.ts
        - Komponensek, modulok deklarálása, importálása a projektbe
    - app-routing.module.ts
        - Útvonalak beállítása, Single Page application
- assets/
    - Képek, Iconok közös elérési mappája

Osztályok

admin.component.ts

A főbb műveletek elvégzéséért felelős osztály. Csak admin jogosultsággal érhető el.

- getAllPets()
    - A pet nevű service getPets metódusát meghívva megkapja az összes állat adatait JSON formátumban, amit foreach ciklussal egy tömbbe ment.
    - bejövő paraméterek: NINCS
    - kimenő adatai: Az összes, adatbázisban szereplő állat adatai
- getAllShelters()
    - A pet-hez hasonló metodikával lekéri az összes menhelyet.
    - bejövő paraméterek: NINCS
    - kimenő adatai: Az összes, adatbázisban szereplő menhely adatai
- getAllBreeds()
    - Az összes fajtát kéri le.
    - bejövő paraméterek: NINCS
    - kimenő adatai: Az összes, adatbázisban szereplő fajta adatai
- change<MezőNév>(e:any)
    - Lenyíló menühöz szükséges beállítások
    - bejövő paraméterek: a lenyíló menüben való változtatás eventje
    - kimenő adatai: változóba menti a kiválasztott opció értékét
- newPet()
    - Új állat adatbázisba mentéséhez kiszedi a kitöltött newPetForm-ból az input mezőkbe beírt adatokat, változókba menti, majd paraméterként átadja a pet service postPets függvényének. A választól függően alertben tájékoztat a művelet sikerességéről.
    - bejövő paraméterek: NINCS
    - kimenő adatai: A hozzáadandó állat adatait adja át a postPets függvénynek
- deletePet()

  - A kilistázott állatok melletti „Törlés” feliratú gomb akciója, az adott állat id-ját átadva hívja meg a pet service deletePet függvényét. A választól függően alertben tájékoztat a művelet sikerességéről.
  - bejövő paraméterek: A kiválasztott állat id-ja
  - kimenő adatai: A kapott id-t adja át a deletePet metódusnak
- onEdit()
  - A módosító form értékeit állítja be a kiválasztott állat adatai alapján.
  - bejövő paraméterek: Megkapja a kiválasztott állat objektumát
  - kimenő adatai: beállítja az updateForm értékeit a bejövő adatok alapján
- updatePet()
  - Változóba menti az updatePetForm form input mezőibe megadott értékeit, majd a pet service updatePet metódusának adja át őket paraméterként. A választól függően alertben tájékoztat a művelet sikerességéről.
  - bejövő paraméterek: NINCS
  - kimenő adatai: A frissítendő állat adatait adja át az updatePets függvénynek

cats.component.ts

A macskák kilistázásáért és örökbefogadásukért felelős osztály. Az örökbefogadás művelet bejelentkezéshez kötött.

- getAllPets()
    - lásd. admin.components.ts
- getPetId()
    - a megnyíló modal miatt volt szükséges data attribute-ok segítségével átadni az adott állat adatait. Ezeket változókban tároljuk
    - bejövő paraméterek: data attribute-ket fogad a kislistázott macska adataival
    - kimenő adatai: változóba menti az attribue-k értékeit
- isLoggedIn()
    - Auth service isLoggedIn függvényét hívja meg, ami által az Örökbefogadom! gomb csak bejelentkezés után látható
    - bejövő paraméterek: NINCS
    - kimenő adatai: tokennel tér vissza
- adoptCat()
    - a data attribute—k változókba elmentett értékét konvertálja át a szükséges típusokra, majd átadja a pet service updatePets függvényének a szükséges paramétereket. A választól függően alertben tájékoztat a művelet sikerességéről.
    - bejövő paraméterek: NINCS
    - kimenő adatai: A frissítendő állat adatait adja át az updatePets függvénynek

dogs.component.ts

megegyezik metodikában a cats.component.ts osztállyal

register.component.ts

Regisztrációért felelős osztály.

- ngOnInit()
    - inicializáljuk a formot, Validátorokkal látjuk el őket
- register()
    - megfelelő formátumba mentve a datepicker által kapott dátumot, változókba mentjük az input mezőkből kinyert adatot, majd meghívjuk az auth service register függvényét, átadva neki a szükséges paramétereket. A választól függően alertben tájékoztat a művelet sikerességéről.

shared/

auth.guard.ts

A /admin útvonal védelmét biztosítja a canActive() függvény segítségével.

auth.service.ts

Minden autentikációhoz köthető api kérést itt találunk.

- register(), login(), logout()
    - A metódusok a kapott paramétereket egy tömbben deklarálják, majd Json formátumba konvertálják. Megadják a kéréshez szükséges fejléceket, a végpontot, és a megfelelő http kérés fajtával megszólítják az apit.
- isLoggedIn(), isAdmin()
    - Ezek a metódusok azt ellenőrzik, hogy a felhasználó be van e jelentkezve, valamint hogy van e admin jogosultsága.

breed.service.ts

- getBreeds()
    - Az api breeds végpontját megszólítva visszatér az adatbázisban szereplő összes fajtával

pet.service.ts

- getPets()
    - Az api pets végpontját megszólítva visszatér az adatbázisban szereplő összes állattal
- updatePets(), postPets(), deletePet()
    - A metódusok a kapott paramétereket egy tömbben deklarálják, majd Json formátumba konvertálják. Megadják a kéréshez szükséges fejléceket, a végpontot, és a megfelelő http kérés fajtával megszólítják az apit.

zipfilter.pipe.ts

- A „Hol vannak menhelyek” menüpont irányítószám szűrésére alkalmas osztály.

shelter.service.ts

- Az api shelters végpontját megszólítva visszatér az adatbázisban szereplő összes menhellyel

shelters.component.ts

- lekéri az össze menhelyet, majd kilistázza őket, valamint beállítja az irányítószám filtert

app.routing.module

- itt állítja be a program az összes útvonalat az adott komponensekhez. Itt állítottuk be az admin útvonal AuthGuarddal való védelmét

Fejlesztői környezet

Backend

- Laravel API (PHP keretrendszer) -
- laravel - sanctum (Autentikációs kiegészítő csomag)
- Mariadb server (Adatbázis kiszolgáló)

Készítéshez használt programok

- Dia
- Visual Studio Code 1.66.0
- Insomnia 2021.5.3
- Mariadb Server
- phpmyadmin
- XAMPP Apache, MYSQL server

Frontend

- Angular (Typescript alapú webalkalmazás keretrendszer)
- Bootstrap 5.1.3 (CSS-keretrendszer)

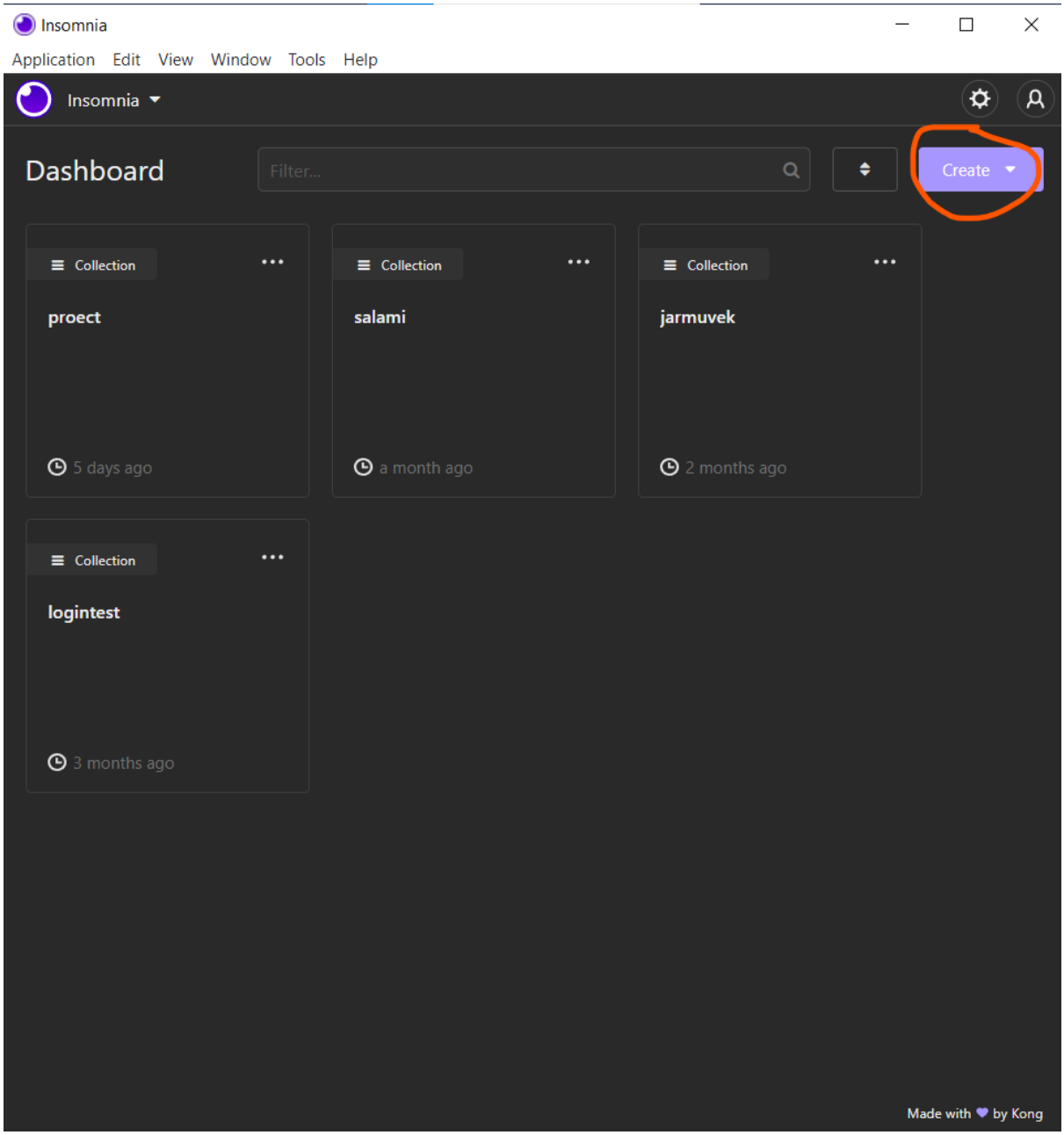
Tesztelés

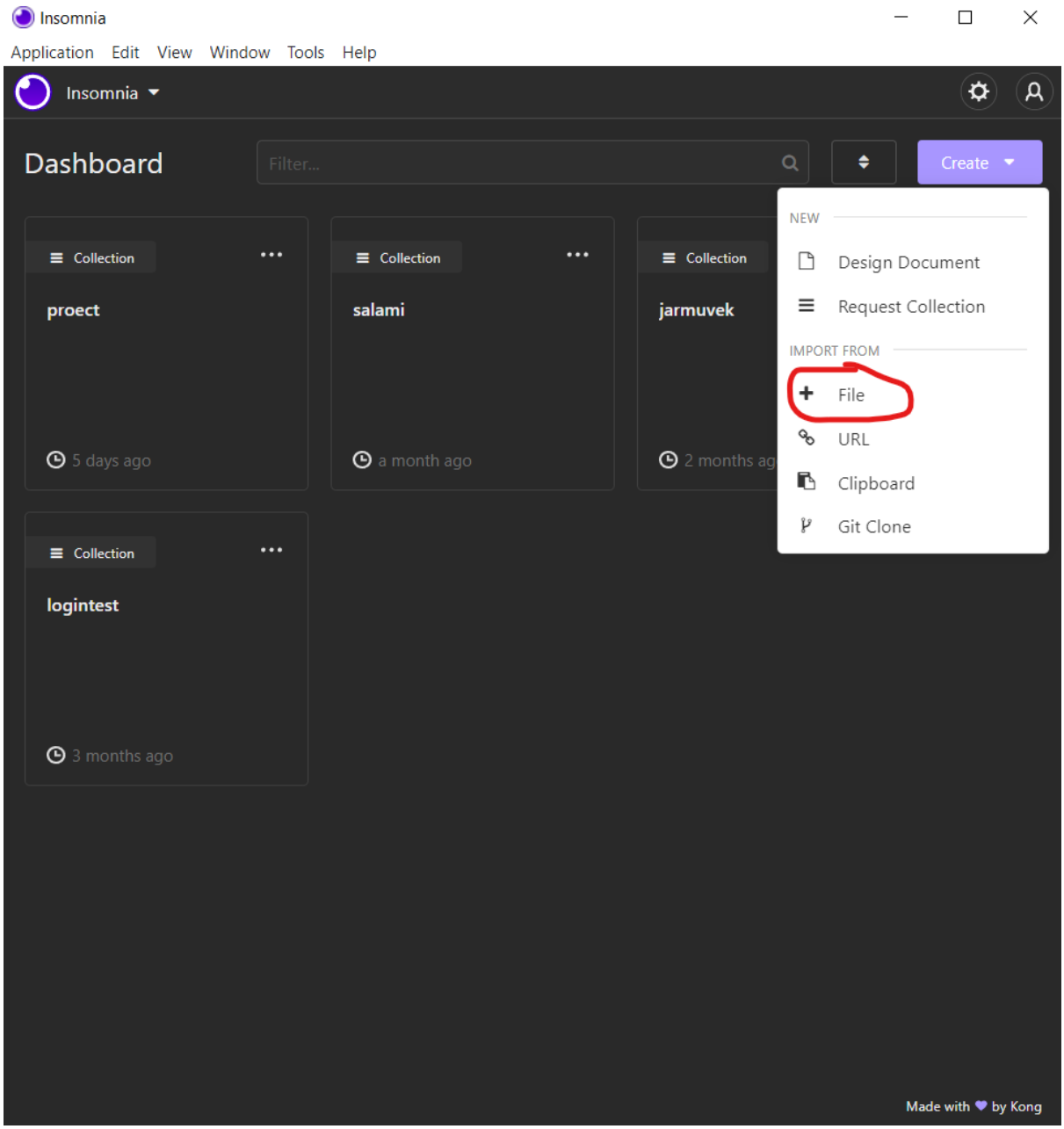
Back-end tesztelése

Az api teszteléséhez szükséges letölteni az Insomnia 2022.2.1 verzióját. Ha letöltöttük a programot, a következő fájlt kell beimportálni az alkalmazásba:

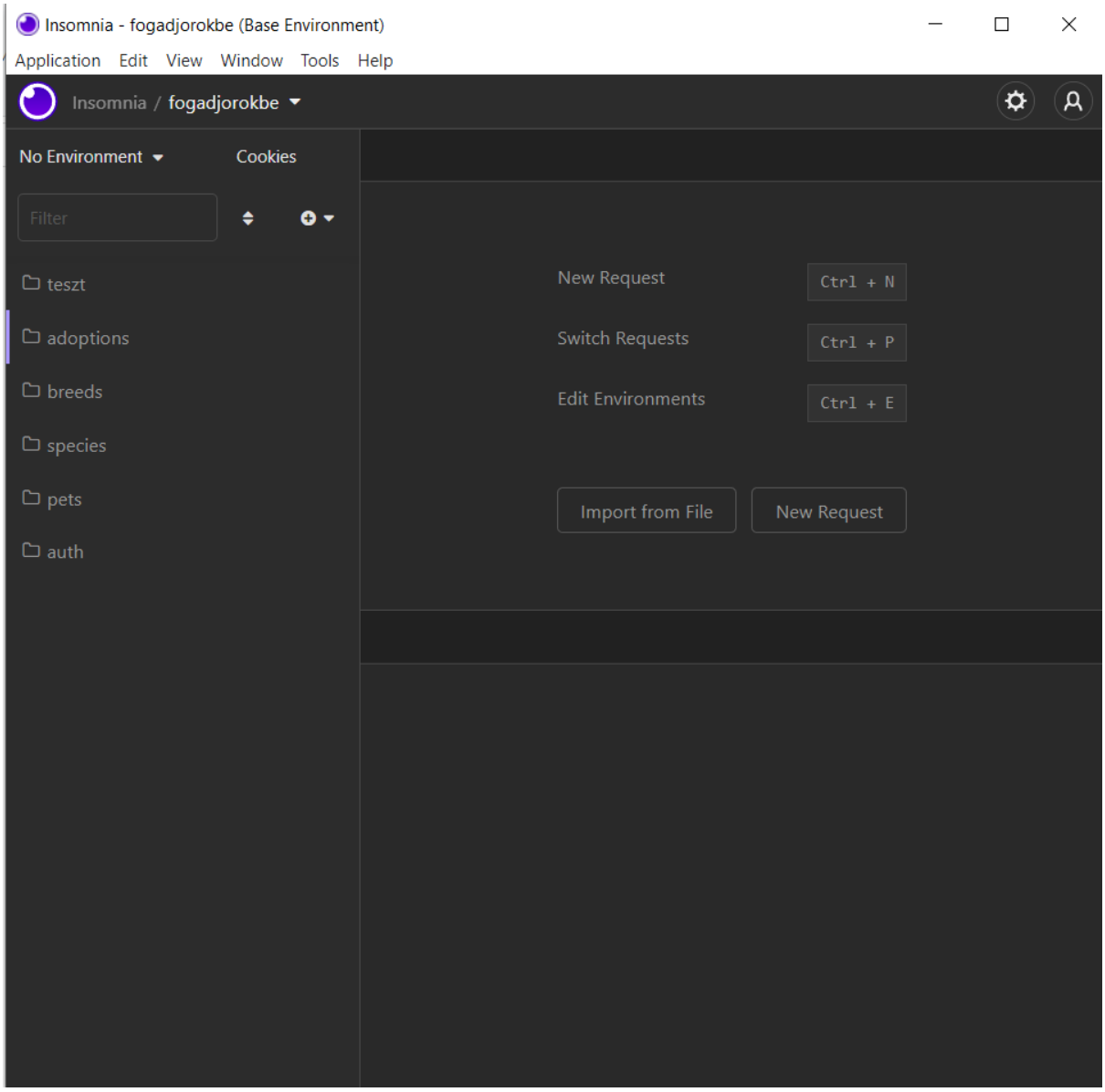
- Projekt/api/shelter/tests/Insomnia_2022-03-31.json

Az importálás a következő menüponton indítható:

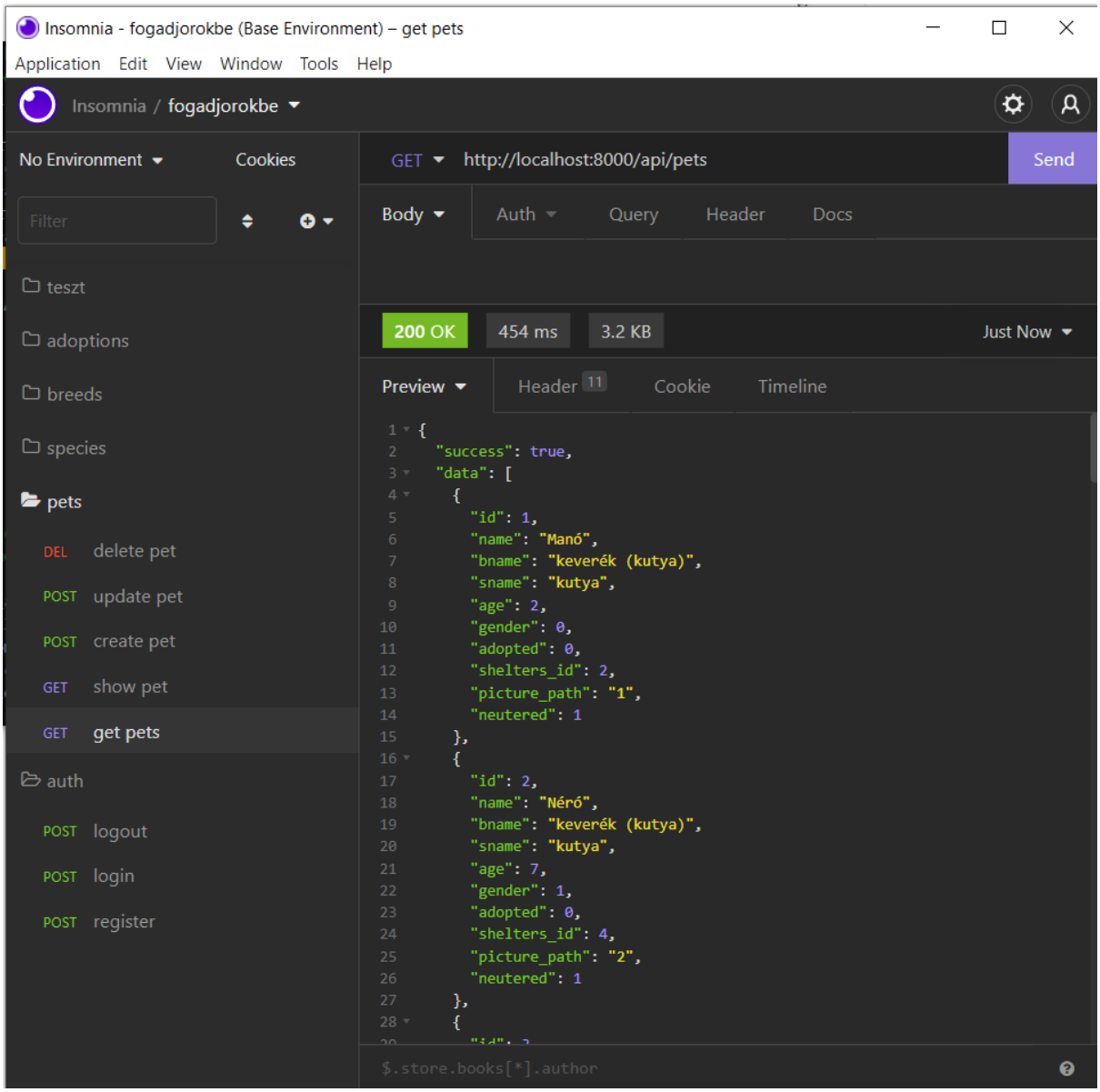




Miután sikeresen beimportáltuk a csomagot, és rákattintunk a fogadjorokbe collection-re, a következőt látjuk:

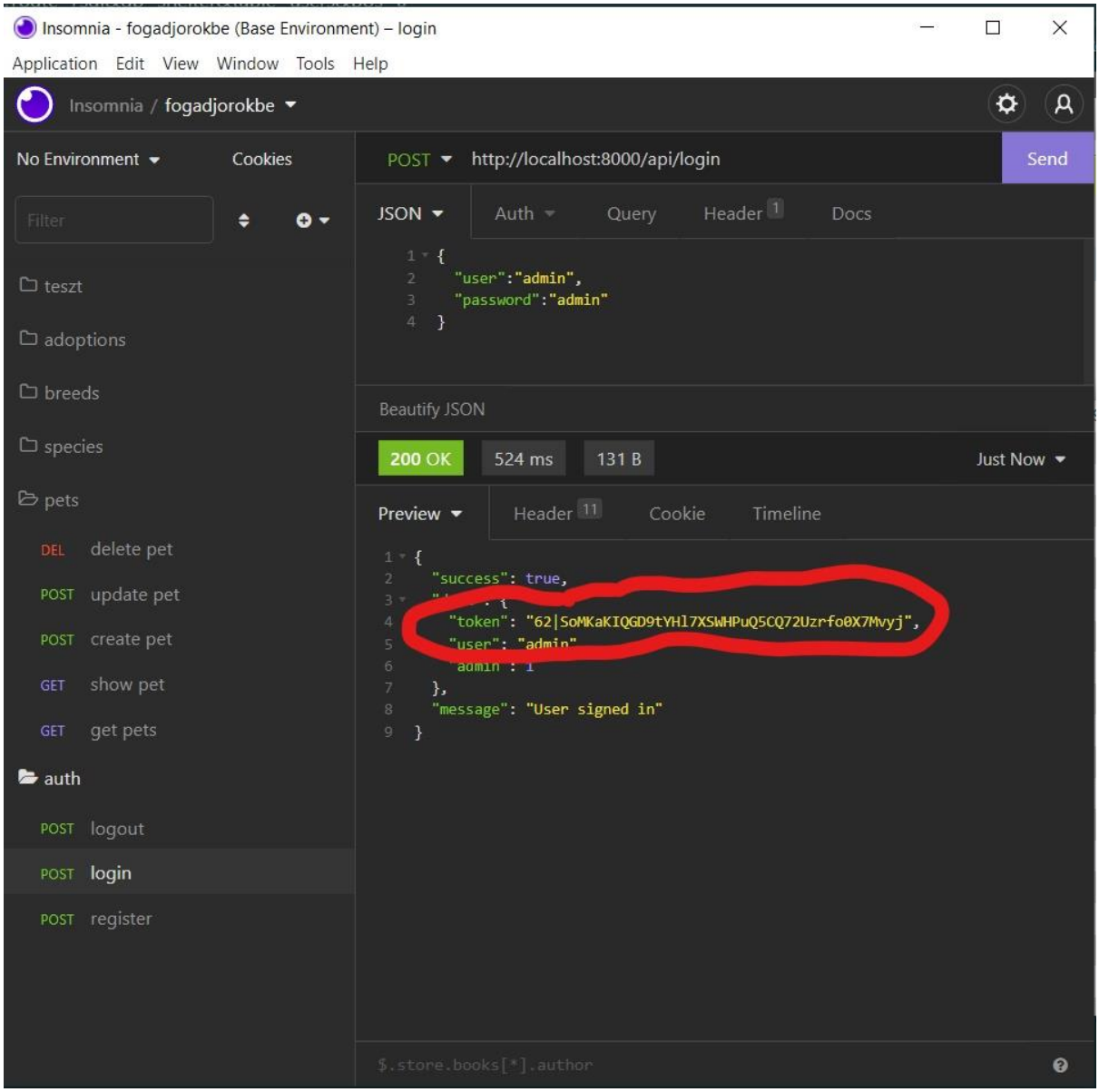


Itt mappákba szedve láthatunk kéréseket, amelyeket az apira küldve letesztelhetjük a működését.
Azokat a végpontokat, ahová nem szükséges autentikáció, rögtön tesztelni tudjuk. Például ilyen a pets mappán belül a get pets kérés, amellyel az adatbázisban található összes állatot kérhetjük le.

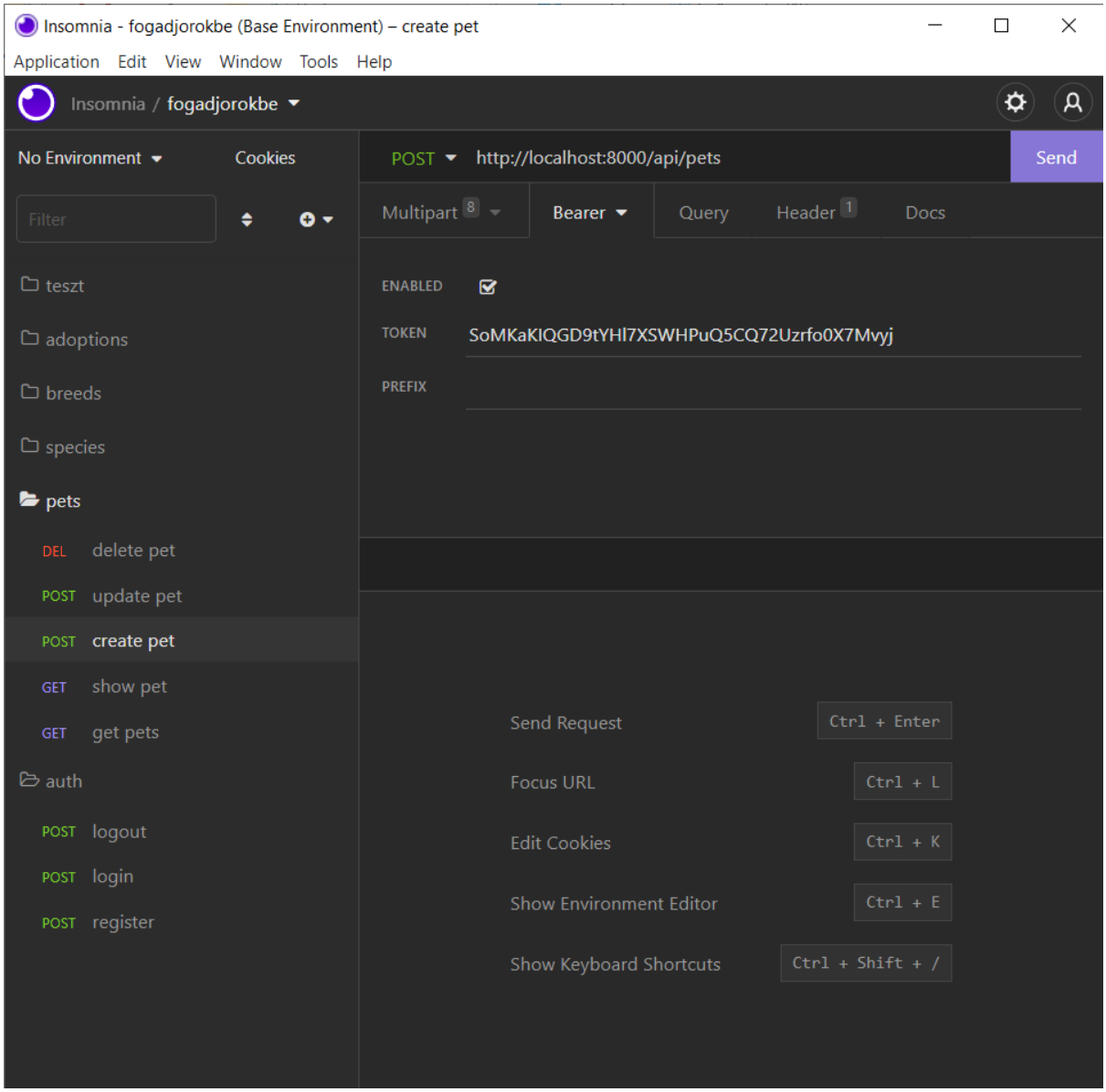


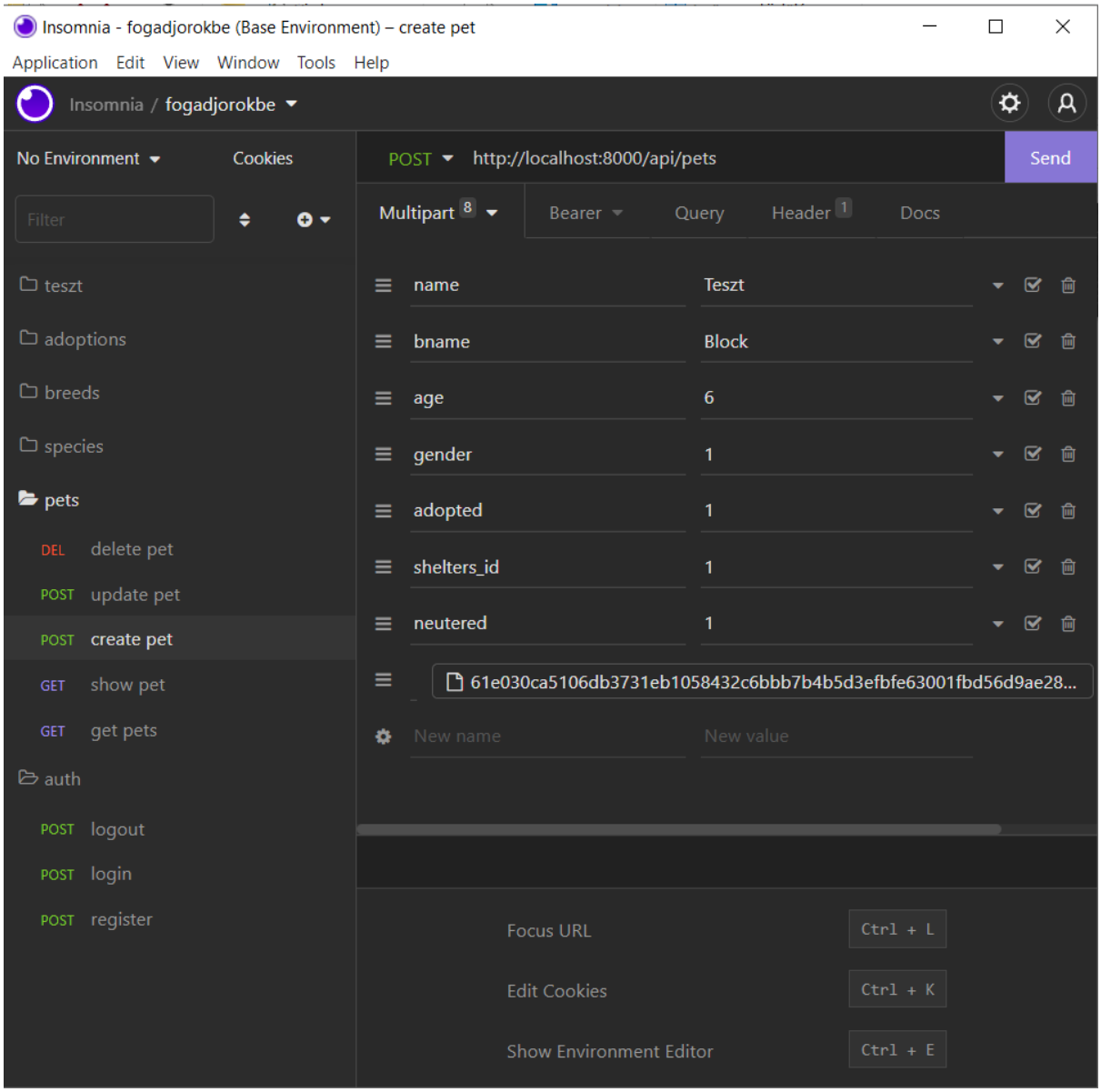
Várt válasz 200 OK státusszal, ha az adott végpont ad vissza adatot, azzal együtt, egy rövid üzenettel.

Ha a végpontokhoz szükséges autentikáció, és nincsen beregisztrált felhasználónk, akkor először az auth/register kéréssel tudunk regisztrálni. Sikeres regisztráció után a login kéréssel lehet bejelentkezni. A sikeres bejelentkezés során egy tokent kapunk amivel jogosultságot szerzünk a védett végpontokhoz is. Ezt a következőképpen kapjuk meg:



Az itt megjelenő tokent kell kimásolnunk. Ezután, ha belépünk egy védett kérésbe, például a pets/create pet-be, ott az Auth fülön a Bearer token opciónál tudjuk beilleszteni a kapott tokent. Ha ezután megadjuk az állat hozzáadásához szükséges adatokat, sikeresen tudjuk ezt a funkciót is tesztelni.


Insomnia - fogadjorokbe (Base Environment) – create pet
Application Edit View Window Tools Help
Insomnia / fogadjorokbe
No Environment
Cookies
Filter
teszt
adoptions
breeds
species
pets
DEL delete pet
POST update pet
POST create pet
GET show pet
GET get pets
auth
POST logout
POST login
POST register
POST http://localhost:8000/api/pets
Send
Multipart 8
Bearer
Query
Header 1
Docs
ENABLED
TOKEN SoMKaKlQGD9tYHl7XSWHPuQ5CQ72Uzrfo0X7Mvyj
PREFIX
Send Request Ctrl + Enter
Focus URL Ctrl + L
Edit Cookies Ctrl + K
Show Environment Editor Ctrl + E
Show Keyboard Shortcuts Ctrl + Shift + /



Front-end tesztelése

A front-end tesztelésére az Angularba beépített Jasmine program segítsége által volt lehetőség. Ha a web/fogadj-orokbe mappában állva kiadjuk az ng test parancsot, láthatjuk mely tartalmi teszteken ment át az alkalmazás.

A front-endet ezen kívül külső felhasználóval teszteltük, úgynevezett Black-Box testing, azaz Feketedobozos teszteléssel. A felhasználó minden funkciót kipróbált, problémába nem ütközött az ismert hibákon felül.

Ismert hibák

- Állatok képeinek tárolásával vannak problémák, a webes felületen sajnos nem sikerült megvalósítani a képfeltöltést, de az api funkcionálisan képes rá. Emiatt az állat frissítésekor is törlődik a mintaadatbázisba feltöltött kép. Ennek kiküszöbölésére a front-end fejlesztésére lenne szükség, de sajnos ez a funkció nem készült el időben.

- Webes felületen való állat módosításnál nem minden mezőt tölt ki automatikusan, néhány mezőt a felhasználónak kell újból kitölteni.

- A regisztrációnál lévő beágyazott naptárnál ki kell választani a legrégebbi évszámot, hogy régebbi évet is tudjunk kiválasztani.

Fejlesztési lehetőség

- Felhasználó által hozzáadott állatok kezelése
- Az örökbefogadási folyamat részletezése
- Nagyobb segítségnyújtás az örökbefogadók számára
- Külön felület menhelyek számára
- Több faj örökbefogadásának lehetősége (például hüllők)
- Részletesebb leírás az állatokról

Tartalom