Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Systémové inženýrství a informatika

Bakalářská práce

Vývoj webové aplikace v jazyce Java

Zuzana Vilhelmová

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Zuzana Vilhelmová

Systémové inženýrství a informatika
Informatika

Název práce

Vývoj webové aplikace v jazyce Java

Název anglicky

Web application development in Java

Cíle práce

Cílem práce bude navrhnout a implementovat webovou aplikaci sloužící jako databáze domácích mazlíčků, umožňující vyhledání vhodného partnera pomocí různých parametrů a vytváření rodokmenu z přidaných mazlíčků. Pokud má zvíře chovatele s kontaktním emailem, lze ho kontaktovat.

Metodika

Práce sestává ze dvou částí. Metodika zpracování teoretické části práce vychází ze studia odborných informačních zdrojů. Na základě syntézy zjištěných poznatků budou formulována teoretická východiska pro zpracování praktické části práce.

Praktická část práce spočívá v návrhu a implementaci webové aplikace pro evidenci domácích mazlíčků. Při zpracování praktické části budou využity standardní nástroje a metody softwarového inženýrství. Pro implementaci front-endu bude využit Angular, back-end vystavující REST rozhraní bude implementován v jazyce Java.

Aplikace bude otestována a budou shrnuty poznatky z jejího vývoje a testování. Dále pak budou nastíněny možnosti případného dalšího rozvoje aplikace.

Doporučený rozsah práce

35-40 stran

Klíčová slova

Web application, Java, Spring Boot, Angular, Mybatis, MySQL, Maven, IntelliJ IDEA

Doporučené zdroje informací

Andrew Glover. Spring Boot in Action. Manning Publications Co., 2016 ISBN 9781617292545
Joel Murach. Murach's MySQL. Mike Murach & Associates Inc. ISBN 1890774685
Joshua Bloch. Effective Java, 3rd Edition. Addison-Wesley Professional. 2017 ISBN 9780134686097
Raghuram Bharathan. Apache Maven Cookbook. Packt Publishing. 2015 ISBN 9781785286124
spring.io
Steve Fenton. Pro TypeScript. Apress. 2014 ISBN 9781430267904

Předběžný termín obhajoby

2019/20 LS – PEF

Vedoucí práce

Ing. Jiří Brožek, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 04. 11. 2020

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Vývoj webové aplikace v jazyce Java" jsem vypracovala samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autorka uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušila autorská práva třetích osob.

V Praze dne 29.11.2020 _____

Poděkování

Ráda bych touto cestou poděkovala Ing. Jiřímu Brožkovi, Ph.D. za pomoc při realizaci této práce a svým blízkým za podporu.

Vývoj webové aplikace v jazyce Java

Abstrakt

Tato bakalářská práce se zabývá vývojem webové aplikace „Pet Tree“ v jazyce Java. Aplikace má umožnit vyhledání mazlíčků, automatického vytváření rodokmenů a pomoci tak vyřešit problém s neexistující možností najít původ velkého množství domácích zvířat.

V úvodní části jsou představeny technologie a nástroje pro vývoj jak klientské, tak serverové části aplikace. Dále je popsána analýza, návrh a implementace vlastní aplikace. Nakonec je popsáno nasazení, testování a zhodnocení aplikace společně s návrhem dalších vylepšení, které přineslo testování uživateli.

Klíčová slova: Java, Spring Boot, Webová aplikace, TypeScript, Angular, MariaDB, JPA, Maven

Web application development in Java

Abstract

This bachelor thesis deals with the development of a web application "Pet Tree" in Java programming language. The application is intended to allow searching for pets, automatic creation of pedigrees and thus help solve the problem of the non-existent possibility of finding the origin of many pets.

The introductory part introduces technologies and tools for the development of both client and server parts of the application. The analysis, design and implementation of the application are also described. Finally, the deployment, testing and evaluation of the application is described, together with a proposal for other improvements that were brought by the user testing.

Keywords: Java, Spring Boot, Web application, TypeScript, Angular, MariaDB, JPA, Maven

Obsah

Seznam obrázků

Seznam zdrojových kódů

Seznam použitých zkratek

API (Application Programming Interface)
CLI (Command Line Interface)
CRUD (Create, Read, Update, Delete)
DOM (Document Object Model)
FTP (File Transfer Protocol)
GDPR (General Data Protection Regulation)
HTTP (Hypertext Transfer Protocol)
JPA (Java Persistence API)
JSON (JavaScript Object Notation)
JWT (JSON Web Token)
ORM (Object-relational mapping)
REST (Representational State Transfer)
SDK (Software Development Kit)
SQL (Structured Query Language)
SVG (Scalable Vector Graphics)
URI (Uniform Resource Identifier)
XML (Extensible Markup Language)

1 Úvod

Pokud si chce člověk v dnešní době pořídit běžně chované domácí zvíře (například psa, kočku, králíka nebo fretku) s průkazem původu buď jen jako domácího mazlíčka, nebo hledá vhodného partnera do jeho chovu, ve většině případů nenarazí na zásadní problém. Pokud ale má zájem o méně běžný druh, lze původ zvířete v některých případech zjistit pouze z plemenné knihy. Zdaleka ne každý chovatel registruje všechna svá zvířata, proto u velkého množství zvířat není možnost jejich původ zjistit.

Potenciálního majitele zajímá druh zvířete, plemeno, morfologická forma, pohlaví a jeho předci. Aplikace „Pet tree“, která je obsahem této práce, tuto funkci plní. Aplikace je mířená primárně na chovatele jako nástroj pro hledání potenciálních partnerů a vytváření rodokmenů za účelem buď zamezení křížení a snížení genetické diverzity nebo naopak křížení za účelem vytváření nových genetických mutací.

Výsledek práce bude nahrán na webový server s doménou www.pet-tree.eu .

2 Cíl práce a metodika

2.1 Cíl práce

Cílem práce je analýza, návrh a implementace webové aplikace „Pet tree“ sloužící jako databáze domácích mazlíčku umožňující vyhledání vhodného partnera a automatického vytváření jejich rodokmenů. Tato webová aplikace dále nabízí vzájemný kontakt majitelů či chovatelů zvířat.

2.2 Metodika

Práce sestává ze dvou částí. Metodika zpracování teoretické části práce vychází ze studia odborných informačních zdrojů. Na základě syntézy zjištěných poznatků budou formulována teoretická východiska pro zpracování praktické části práce.

Praktická část práce spočívá v návrhu a implementaci webové aplikace pro evidenci domácích mazlíčků. Při zpracování praktické části budou využity standardní nástroje a metody softwarového inženýrství. Pro implementaci front-end části aplikace bude využit framework Angular a back-end vystavující RESTful rozhraní, který bude implementován v jazyce Java a frameworku Spring Boot.

Aplikace bude otestována a nasazena na produkční server. Z jejího vývoje a testování budou shrnuty poznatky a dále pak budou nastíněny možnosti případného dalšího rozvoje aplikace.

3 Teoretická východiska

3.1 Java

Java je objektově orientovaný jazyk původně vyvinutý firmou Sun Microsystems pod vedením Jamese Goslinga a Billa Joye. Byla navržena jako platformově nezávislý programovací jazyk. Od verze Java SE 11, kdy Java již byla vlastněná firmou Oracle, se změnily podmínky licencování a jazyk s jeho nástroji nelze používat zdarma ke komerčním účelům. Od roku 2017 je ale firmou Oracle oficiálně podporován projekt OpenJDK, který je open source variantou jazyka Java a umožňuje použití zdarma i ke komerčním účelům. Je však vydáván se zpožděním a bez profesionální podpory. (1)

Java současně patří mezi nejpopulárnější jazyky na světě. (2)

3.1.1 Spring Framework

Spring Framework je populární open-source framework pro aplikace založené na Javě a skládá se z několika modulů.

Modul core obsahující například dependency injection (vkládání závislostí), události, lokalizaci, validaci, datovou vazbu a převody typů, modul pro testování, přístup k datům, MVC a WebFlux framework, a integrace jako je email, plánování nebo caching.

„Dalo by se říct, že všechny moduly spojuje jeden společný kontejner. Spring kontejner využívá návrhový vzor Inversion of Control (IoC). Ten uvolňuje pevné vazby mezi objekty. Pevná vazba znamená, že třída si sama inicializuje své vlastnosti (jiné třídy, se kterými má vztah) a nedostane je z venčí.“ (3)

„S IoC souvisí dependency injection. Je to jednoduchý návrhový vzor, kdy je do třídy vložena instance jiné třídy a framework se o ní stará sám. Umožňuje flexibilitu a testovatelnost.“ (4)

3.1.1.1 Spring Boot

Spring Boot je kombinací Spring Frameworku a aplikačního serveru (Tomcat, Jetty, Undertow). Zjednodušuje konfiguraci projektu pro jeho sestavení, dále také nabízí nástroje pro měření a sledování stavu aplikace. Umožňuje automatickou konfiguraci některých knihoven třetích stran a nevyžaduje xml konfigurace, jako je tomu u standardního Spring Frameworku. Vývoj i nasazení je tak snadnější a rychlejší. (5)

3.1.1.2 Spring Data JPA

Knihovna pro Spring Framework, která zpřístupňuje data pomocí enterprise specifikace jménem Jakarta Persistence API, ve zkratce JPA. Jde o standard jazyka Java, který umožňuje objektově relační mapování (ORM), jako například pomocí Hibernate.

Spring Data JPA přidává abstraktní vrstvu pro výrazné snížení běžné části kódu potřebné k implementaci datově přístupových vrstev pro různá úložiště. (6)

3.1.2 Hibernate

Hibernate je framework založený na jazyce Java pro objektově relační mapování (ORM). Je to technika pro mapování objektů aplikačních doménových modelů na tabulky relačních databází a opačně. (6)

3.1.3 Apache Maven

Apache Maven je populární nástroj pro správu, řízení a automatizaci buildů, primárně pro projekty v jazyce Java. Každý projekt je popsán pomocí pom.xml (project object model).

Kombinace `groupId`, `artifactId` a `version` unikátně identifikuje projekt. `GroupId` je identifikátor organizace, která projekt vlastní, `artifactId` je název projektu a version udává specifickou instanci projektu. Verze se obvykle udávají ve tvaru `<major>.<minor>.<patch>-<qualifier>`. Pokud verze obsahuje kvantifikátor `SNAPSHOT`, značí to, že je verze projektu nezveřejněná a je stále ve vývoji. Může to znamenat například zásadní chyby, které způsobí pád aplikace.

Další základní element `pom.xml` je seznam závislostí, které Maven stahuje z repozitáře (databáze existujících sestavených projektů a jejich verzí).

Existují dva základní typy repozitářů:

- Local – adresář v počítači, kde Maven běží. Ukládají se tam závislosti stažené ze vzdáleného repozitáře a dočasné vlastní sestavené projekty, které ještě nebyly vydané.
- Remote – jakýkoliv typ vzdáleného repozitáře přístupného například přes FTP (File Transfer Protocol) nebo HTTP (Hypertext Transfer Protocol). Může se jednat o vzdálený repozitář třetí strany nebo interní firemní repozitář pro spolupráci mezi teamy.

Při primárním nastavení stahuje závislosti ze vzdáleného Maven Central Repository, je ho možné ale přenastavit na jiný. Každá závislost představuje projekt určený identifikátory výše.

Maven poskytuje tři životní cykly:

- default – základní sestavení projektu a jeho a nasazení
- clean – smazání dočasných a jiných generovaných složek a souborů vytvořených Mavenem
- site – vytvoření dokumentace projektu

Výchozí životní cyklus sestavení se dělí na fáze (`validate`, `process-resources`, `compile`, `test`, `package`, `integration-test`, `verify`, `install` a `deploy`). Při spuštění životního cyklu Maven spouští jednotlivé fáze od začátku v daném pořadí, až do fáze, která byla zadána při spuštění.

Důležitý element `packaging` označuje typ výsledného balíčku sestavení projektu pro další integraci či nasazení. Mezi nejčastěji používané patří `jar`, `war` nebo `pom`. (7)

3.2 REST API

Representational state transfer (REST) je architektura rozhraní pro distribuované systémy založené na hypermédiích.

Hlavní výhodou REST API je, že se neváže na žádnou konkrétní implementaci. Klient i server tak mohou být napsány v odlišném libovolném jazyce, které dovedou generovat požadavky a parsovat odpovědi HTTP. (8)

„Rozhraní REST je použitelné pro jednotný a snadný přístup ke zdrojům (resources). Zdrojem mohou být data, stejně jako stavy aplikace (pokud je lze popsat konkrétními daty). Všechny zdroje mají vlastní identifikátor URI a REST definuje čtyři základní metody pro přístup k nim.

REST implementuje čtyři základní metody, které jsou známé pod označením CRUD, tedy vytvoření dat (Create), získání požadovaných dat (Retrieve), změnu (Update) a smazání (Delete). Tyto metody jsou implementovány pomocí odpovídajících metod HTTP protokolu (CREATE, GET, UPDATE, DELETE). “ (9)

Odpověď HTTP obsahuje stavový kód, který udává, zda daný požadavek skončil úspěšně. Jsou rozdělené do pěti skupin:

1. Informační (1xx)
2. Úspěšné (2xx)

3. Přesměrování (3xx)
4. Klientské chyby (4xx)
5. Serverové chyby (5xx)

Nejčastěji používané kódy v aplikaci:

- 200 OK – Úspěšný požadavek.
- 201 Created – Požadavek byl úspěšný a byl vytvořen nový obsah.
- 204 No Content – Požadavek byl úspěšný, ale nevrátil data.
- 400 Bad Request – Nevalidní požadavek z klienta na server.
- 401 Unauthorized – Klient není ověřený.
- 403 Forbidden – Klient nemá práva pro přístup ke zdroji.
- 404 Not Found – Server nenašel požadovaný zdroj.
- 409 Conflict – Požadavek nelze splnit vzhledem ke konfliktu.
- 500 Internal Server Error – Požadavek na serveru skončil chybou.
- 503 Service Unavailable – Server není připravený pro zpracování požadavku. Obvykle značí vypnutí pro údržbu, nebo přetížení.

(10)

3.3 JSON Web Token (JWT)

JSON Web Token (JWT) je standard definující kompaktní způsob pro bezpečný přenos informací mezi dvěma stranami v podobě JSON objektu. Informacím může být důvěřováno ověřením digitálního podpisu. Token je podepsán pomocí tajného klíče (secretu) a šifrovacího algoritmu, nebo pomocí páru privátního a veřejného klíče používající jeden z šifrovacích protokolů pro digitální podpis.

JWT má tři části. Hlavičku obsahující typ algoritmu a typ tokenu, payload neboli data, kde se obvykle ukládá například identifikátor uživatele a popis. (11)

3.4 MariaDB

MariaDB je jedna z nejpopulárnějších relačních databází a je založena na původním projektu MySQL. Je vyvíjena původními vývojáři MySQL a garantuje otevřenost softwaru. Je součástí většiny nabídky cloudových služeb a je instalován jako výchozí balíček nahrazující původní MySQL ve většině Linuxových distribucí. (12)

3.5 Jednostránková aplikace

Jednostránková aplikace funguje uvnitř prohlížeče bez nutnosti načítat celou stránku znova při interakci s uživatelem. Kód aplikace je tak načtený po celou dobu, kdy je stránka načtená v prohlížeči a dochází pouze k načtení konkrétních dat se kterými aplikace pracuje. Mezi nevýhody patří nutnost povoleného JavaScriptu, bez kterého aplikace nemůže fungovat. Je také náchylnější na útoky pomocí cross-site scripting (XSS), která spočívá v podstrčení cizího kódu do dynamicky načtené webové stránky. (13)

3.6 TypeScript

Typescript je open-source jazyk vyvinutý a spravovaný firmou Microsoft. Je postavený na jazyku JavaScript a rozšiřuje ho přidáním staticky validovaných typů. Typy přináší možnost definování struktury objektů a umožňují TypeScriptu validovat správnost kódu. Tato statická validace není však povinná a je možné využít pouze dynamickou validaci, kterou poskytuje jazyk JavaScript.

Kód je transformovaný kompilátorem do JavaScriptu, lze proto psát do TypeScriptu i čistý JavaScript. Je možné vynutit podporu starších prohlížečů. Má to tu výhodu, že například při použití novější funkce, bude výstup TypeScript kompilátoru kompatibilní a stále spustitelný i ve starším prohlížeči. (14)

3.6.1 Angular

Angular je framework v jazyce TypeScript pro vývoj klientských jednostránkových aplikací. Je multiplatformní a rychlý. Díky nástroji Angular CLI umožňuje rychlý vývoj, přidávání komponent, testů a rychlé nasazení. (15)

3.7 Scalable Vector Graphics (SVG)

SVG je značkovací jazyk a formát souboru, který definuje vektorovou grafiku ve formátu XML. To znamená, že každý element je dostupný v SVG DOM (Document Object Model) a lze jej programově upravovat. Podporuje také obsluhu událostí v prohlížeči. Je to standardní způsob vykreslování 2D grafiky nejen v prohlížeči. (16)

4 Vlastní práce

4.1 Analýza

Aplikace „Pet Tree“ má plnit úkoly jako je registrace a přihlášení uživatelů, správa mazlíčků a kreslení rodokmenu. Vzhledem k tomu, že tyto činnosti jsou běžné i u jiných webových aplikací, je vhodné je zanalyzovat a využít získaných znalostí pro tuto aplikaci. Dále je třeba identifikovat všechny funkční a nefunkční požadavky.

4.1.1 Analýza podobných stránek

V rámci analýzy byla objevena jediná podobná webová aplikace Pedigree Pets. (17)

Jedná se o aplikaci pro registrované chovatele na Novém Zélandu a Austrálie. Aplikace umožňuje zvlášť jak hledání dostupných mazlíčků k prodeji, tak chovatelů a vrhů. Všechna hledání obsahují filtr na druh zvířete (pes a kočka) a zmíněné oblasti.

Aplikace obsahuje také blog s několika užitečnými informacemi a zvlášť sekci se všemi plemeny psů a koček, které aplikace podporuje.

Při registraci aplikace rozlišuje, zda je uživatel chovatel či kupující hledající domácího mazlíčka. Kromě obvyklých základních údajů požaduje i telefonní číslo a biografii, kde obsah záleží na typu uživatele. Chovatel by měl uvést svá chovná zvířata včetně typu, plemena a dalších souvisejících informací. Kupující by naopak měl napsat dostatek informací pro chovatele, kterého může zajímat budoucí domov zvířete a zda je pro něj vhodný.

Další sekce opět záleží na roli uživatele (lze vybrat obojí). Po vybrání rolí následují otázky související s biografií. U chovatele je například výčet psích i kočičích plemen na výběr, díky kterým je následně v jeho profilu odkaz na detail vybraných plemen. Na konci registrace lze nahrát profilový obrázek.

Po odeslání registrace přišel email s informací nutnosti ruční verifikace administrátorem do 72 hodin a odkazy na Facebookové skupiny, které mají velikost kolem 10 uživatelů.

4.1.2 Požadavky aplikace

Ze zadání vyplývají následující funkční požadavky:

- Podpora všech druhů domácích mazlíčků

- Vyhledávání mazlíčků pomocí různých parametrů včetně lokace
- Přidávání mazlíčků
- Editace mazlíčků
- Registrace jako vlastník mazlíčků i jako chovatel
- Přihlášení
- Vytvoření rodokmenu
- Možnost kontaktovat chovatele
- Správa rolí uživatelů

Nefunkční požadavky:

- Jednoduchost ovládání
- Bezpečnost a splnění GDPR
- Rozšiřitelnost

4.1.3 Databáze druhů zvířat

Pro podporu všech druhů domácích mazlíčků a jejich snadné vyhledávání je vhodné zanalyzovat již existující API s databází zvířat:

- iNaturalist (18)
- Petfinder (19)

Obě databáze vystavují REST API, pro jehož používání je třeba registrovat aplikaci a získat pro ni identifikátor nebo token, pomocí kterého se získává nejprve potřebná autorizace. iNaturalist povoluje maximálně 100 dotazů za minutu, lépe však pod 60, nebo pod 10 000 za den. Petfinder je striktnější se základním omezením 50 dotazů za sekundu nebo 1 000 za den, umožňuje však pozdější navýšení jeli potřeba.

API webové aplikace iNaturalist je více vědecky zaměřené. Obsahuje velkou škálu zvířat a databáze je velmi aktivně udržovaná. Chybí však informace, zda se daný druh chová jako domácí mazlíček, proto není příliš vhodné pro tuto aplikaci.

Petfinder oproti tomu obsahuje výhradně domácí mazlíčky. Vrací jak druhy zvířat, tak některá plemena. Bohužel jeho struktura není vždy správná. Jako příklad lze uvést kategorie psů, obsahuje vrací správně jejich druhy a plemena, ale zobecněná kategorie nazvaná „šupiny, ploutve a ostatní“ obsahuje místo druhů spíše celé třídy a podtřídy. Pod plemeny se teprve skrývá druh zvířete. Petfinder API by se dalo tedy s určitým omezením použít, ale musela by se implementovat pro některé kategorie odlišná logika.

4.1.4 Hledání zeměpisné pozice

K hledání mazlíčků podle lokace je třeba analyzovat služby umožňující geokódování (hledání zeměpisné pozice podle textového řetězce a zeměpisných objektů na souřadnici). Analýza se bude zabývat pouze těmi, které jsou k dispozici zdarma.

- Jawg Maps (20)
- HERE (21)
- Mapbox (22)
- LocationIQ (23)

Jawg Maps umožňuje zdarma 50 000 načtení mapy a 10 000 požadavků na geolokaci za měsíc. Poskytuje zdroj pro hledání, které jde omezit například jen na region, což umožní zamezit sběru přesných adres, které nejsou potřeba. Do mapy lze vkládat vlastní značky.

HERE zdarma poskytuje 250 000 transakcí, 5 000 uživatelů a přenos 2,5 GB dat za měsíc. API podle dokumentace neumí filtr jako Jawg Maps, má však filtr na konkrétní stát.

Mapbox se zásadně oproti Jawg Maps liší pouze omezením 100 000 požadavků na geolokaci za měsíc, jinak jsou API vcelku podobná.

Poslední LocationIQ pro použití zdarma umožňuje 5 000 dotazů za den, funkčně je API podobné HERE.

4.1.5 Knihovna na generování rodokmenu

Generování rodokmenu se skládá ze dvou částí. Jedna je vizualizace a druhá získání datové struktury rodokmenu.

Vizualizaci grafu lze ve zvoleném frameworku Angular pomocí následujících tří knihoven:

- Basic Primitives (24)
- ngx-echarts (25)
- ngx-graph (26)

Basic Primitives je knihovna přímo podporující generování rodokmenů. K vygenerování ale požaduje informaci o „manželství“, kterou aplikace nezná. Důvod je dále vysvětlen v kapitole návrhu datového modelu pod entitou „pet“.

Ngx-echarts je jednoduchá knihovna, ale neposkytuje potřebné vlastnosti jako přizpůsobení vzhledu uzlu a přidání odkazu. To je v této aplikaci potřeba pro uživatelsky přívětivé rozhraní a snadnou navigaci aplikací.

Ngx-graph je naopak šablonovací systém pro generování SVG (technologie zmíněná již v kapitole teoretická východiska) grafu. Šablony lze zvlášť specifikovat pro uzly, hrany a klastry. Je tak možné vygenerovat uzel i s obrázkem a odkazem na detail příslušného zvířete. Lze tak snadno prohlížet předky a potomky. Pomocí klastrů lze také oddělit generace a tím graf vizuálně zpřehlednit.

4.1.6 Ochrana osobních dat – GDPR

„GDPR neboli General Data Protection Regulation (česky Obecné nařízení na ochranu osobních údajů) je dosud nejvíce uceleným souborem pravidel na ochranu dat na světě. Představuje nový právní rámec ochrany osobních údajů v evropském prostoru, které od 25. května 2018 přímo stanoví pravidla pro zpracování osobních údajů, včetně práv subjektu údajů (fyzické osoby). V českém právním prostředí tak Obecné nařízení nahradí zákon č. 101/2000 Sb., o ochraně osobních údajů a o změně některých zákonů.“ (27)

Jelikož se mezi osobní údaje řadí jméno a email, které tato aplikace používá k registraci uživatele, bude nutné splnit následující identifikované požadavky GDPR: (28)

- Kontaktní údaje uživatelů pod 16 let nesmí být veřejné
- Rodičovský souhlas pro uživatele mladší 16 let
- Právo být editován/smazán
- Osobní data nepotřebná k funkčnosti aplikace nesmí být shromažďována

4.1.7 Výsledek analýzy

Na základě studia existujících databází zvířat bylo odpuštěno od možnosti využití existujících databází zvířat, které by znamenalo omezení množství nebo vhodnost dostupných druhů.

Pro kreslení grafu rodokmene byla vybrána knihovna ngx-graph, protože jako jediná umožňuje vytvoření vlastních šablon, díky kterým bude možné grad přizpůsobit pro potřeby této aplikace.

Na vyhledávání geolokace bylo vybráno API Mapbox. Poskytuje vhodné filtry a umožňuje rozumný počet dotazů zdarma.

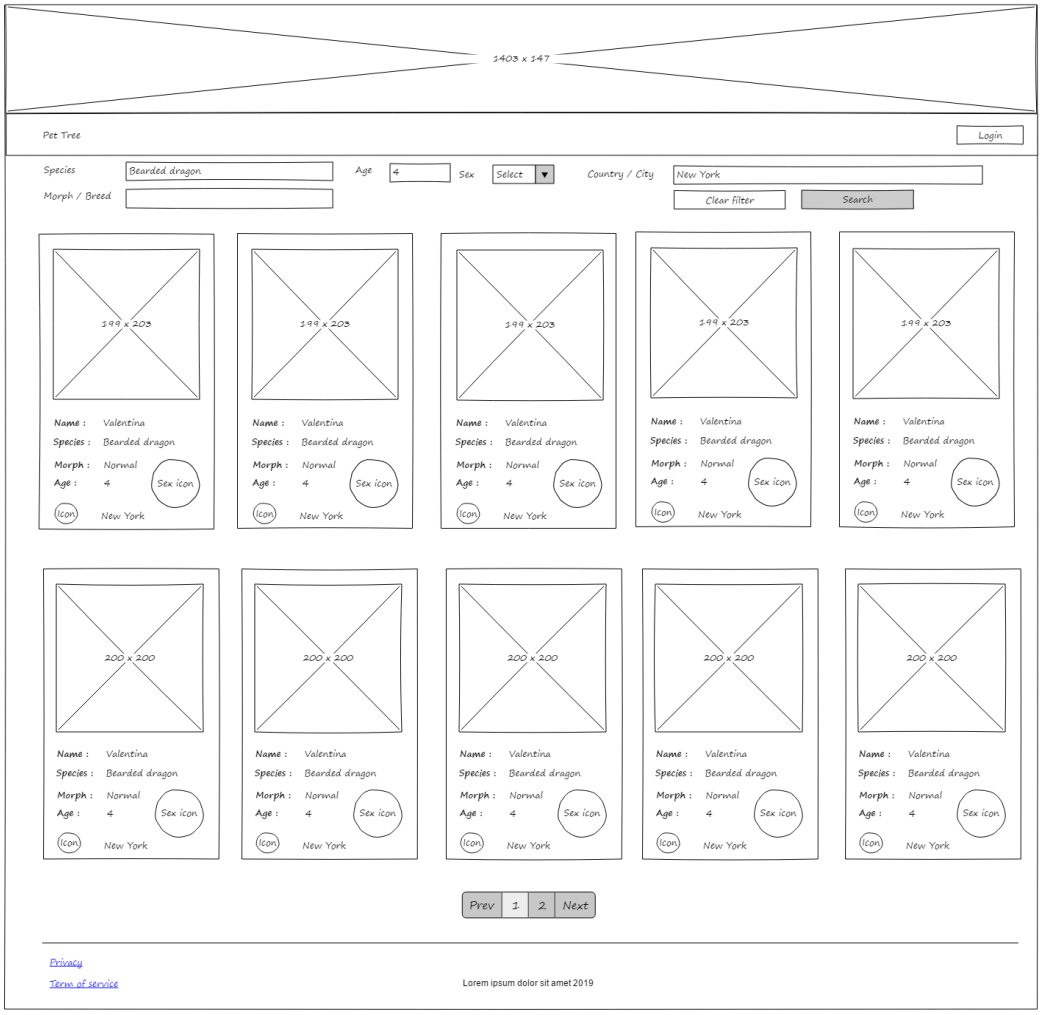
4.2 Návrh

4.2.1 Drátové modely

Všechny obrazovky budou obsahovat banner, navigační lištu s odkazem na úvodní stránku, tlačítko pro přihlášení, nebo po přihlášení uživatele tlačítko odkazující do profilu a odhlášení. Pokud má uživatel alespoň jednu z rolí administrátora, bude na navigační liště i odkaz do administrace.

Dole v patičce bude pro splnění GDPR aplikace obsahovat stránky „Soukromí“ a „Smluvní podmínky“.

4.2.1.1 Úvodní stránka

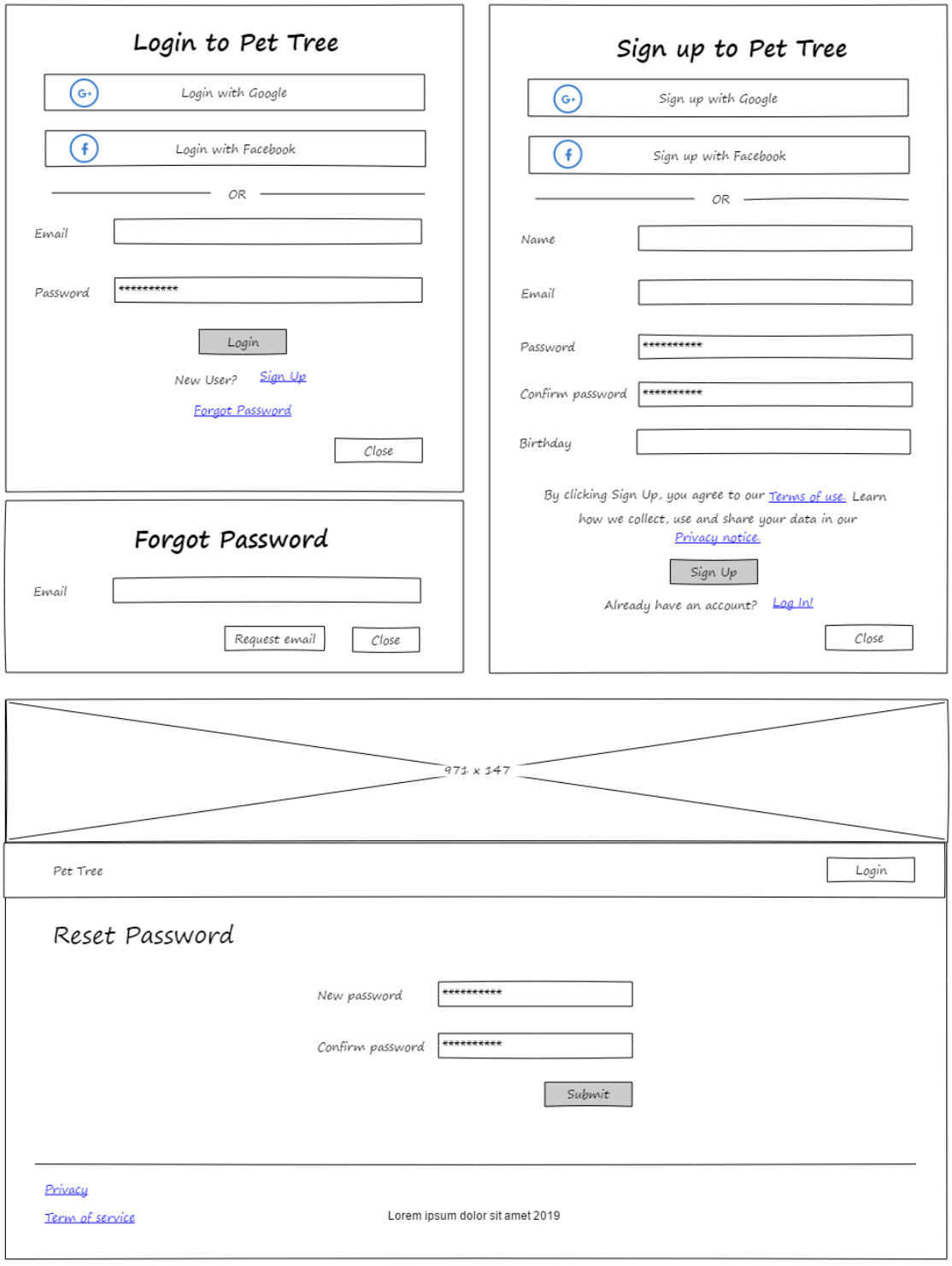


Obrázek 1 - Drátový model úvodní stránky (zdroj: autorka)

Úvodní stránka bude obsahovat filtr pro vyhledávání mazlíčků a jejich stránkovaný seznam. Filtrovat bude možné podle všech parametrů zvířete a adresy vlastníka omezené na město a stát. Pole adresy bude napojeno na vyhledávací API Mapbox.

Zvířata budou zobrazena ve stylu dlaždic s odkazem na jejich detail, zobrazenou fotkou a základními údaji.

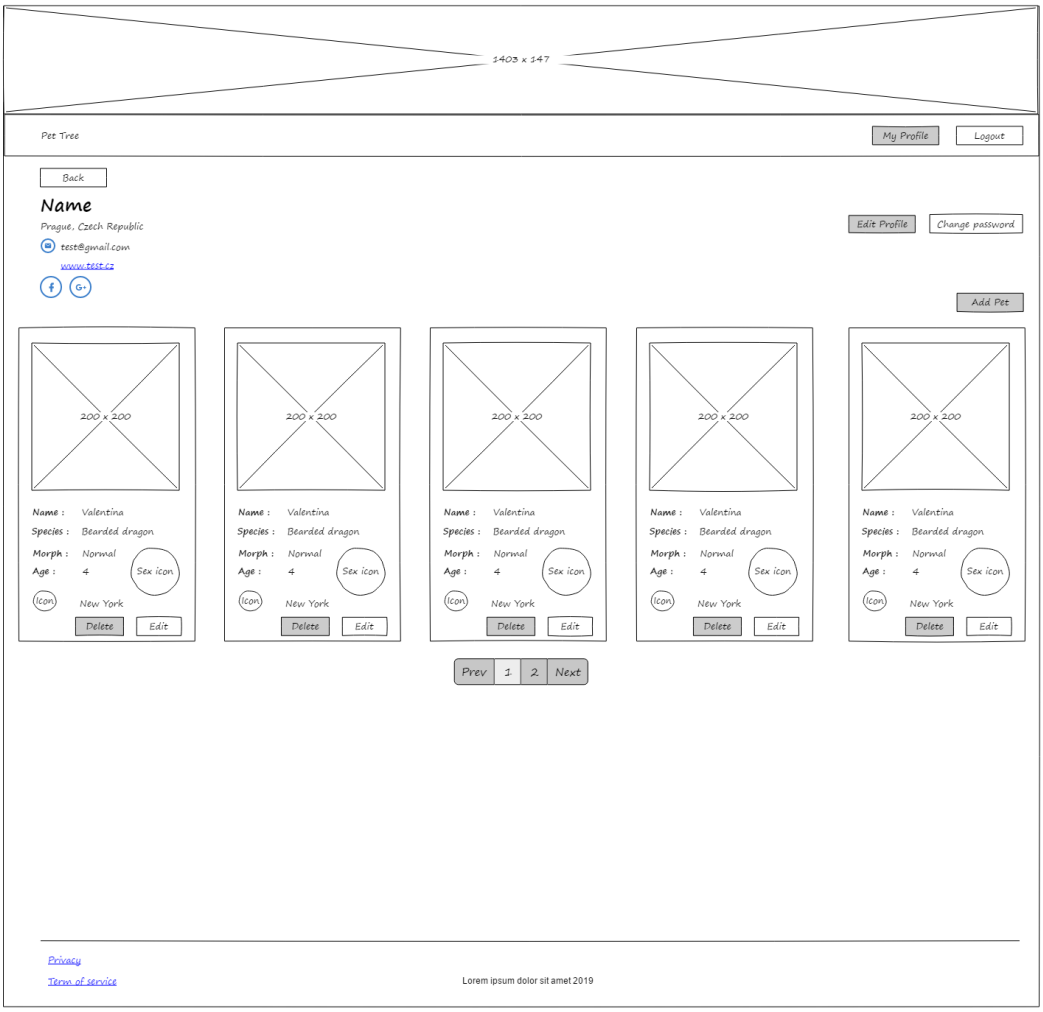
4.2.1.2 Přihlášení



Obrázek 2 - Drátový model přihlášení, registrace a resetu hesla (zdroj: autorka)

Na dialogu pro přihlášení se bude možné přihlásit pomocí Google či Facebook účtu, nebo kombinací emailu a hesla. Dialog bude možné přepínat mezi přihlášením a registrací. Pro osoby mladší 18 let nebude umožněna registrace. Nebude tak nutné řešit rodičovský souhlas. Registrace tedy bude obsahovat i pole datum narození, bude však sloužit pouze k validačním účelům a nebude se ukládat do databáze. Po registraci se uživateli pošle ověřovací email. Pod přihlášením bude tlačítko na dialog s formulářem pro zaslání emailu s odkazem na resetování hesla.

4.2.1.3 Profil

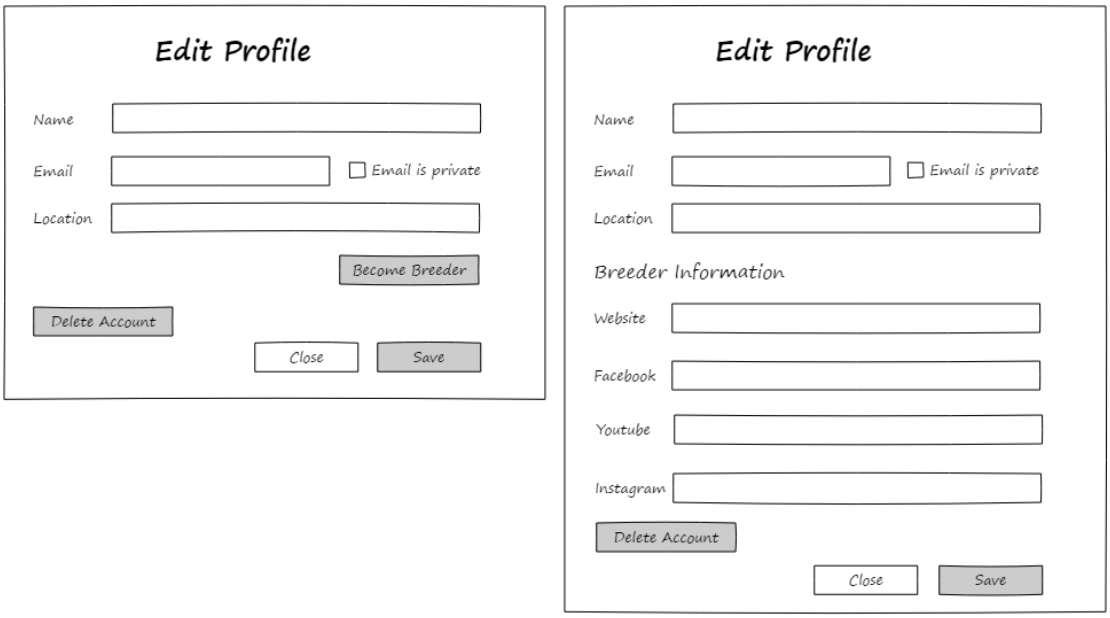


Obrázek 3 - Drátový model profilu uživatele (zdroj: autorka)

Na obrazovce profilu uživatele budou zobrazena jeho základní informace kromě emailové adresy, pokud není veřejná a nejedná se o profil přihlášeného uživatele. Nad údaji vlevo bude tlačítko zpět.

Pod údaji bude stránkovaný seznam mazlíčků uživatele stejně jako na úvodní stránce.

Pokud se jedná o profil přihlášeného uživatele, nad seznamem mazlíčku přibude tlačítko „přidat mazlíčka“, nad ním „upravit profil“ a pokud je uživatel registrován pomocí emailu a hesla, zobrazí se i tlačítko „změnit heslo“.

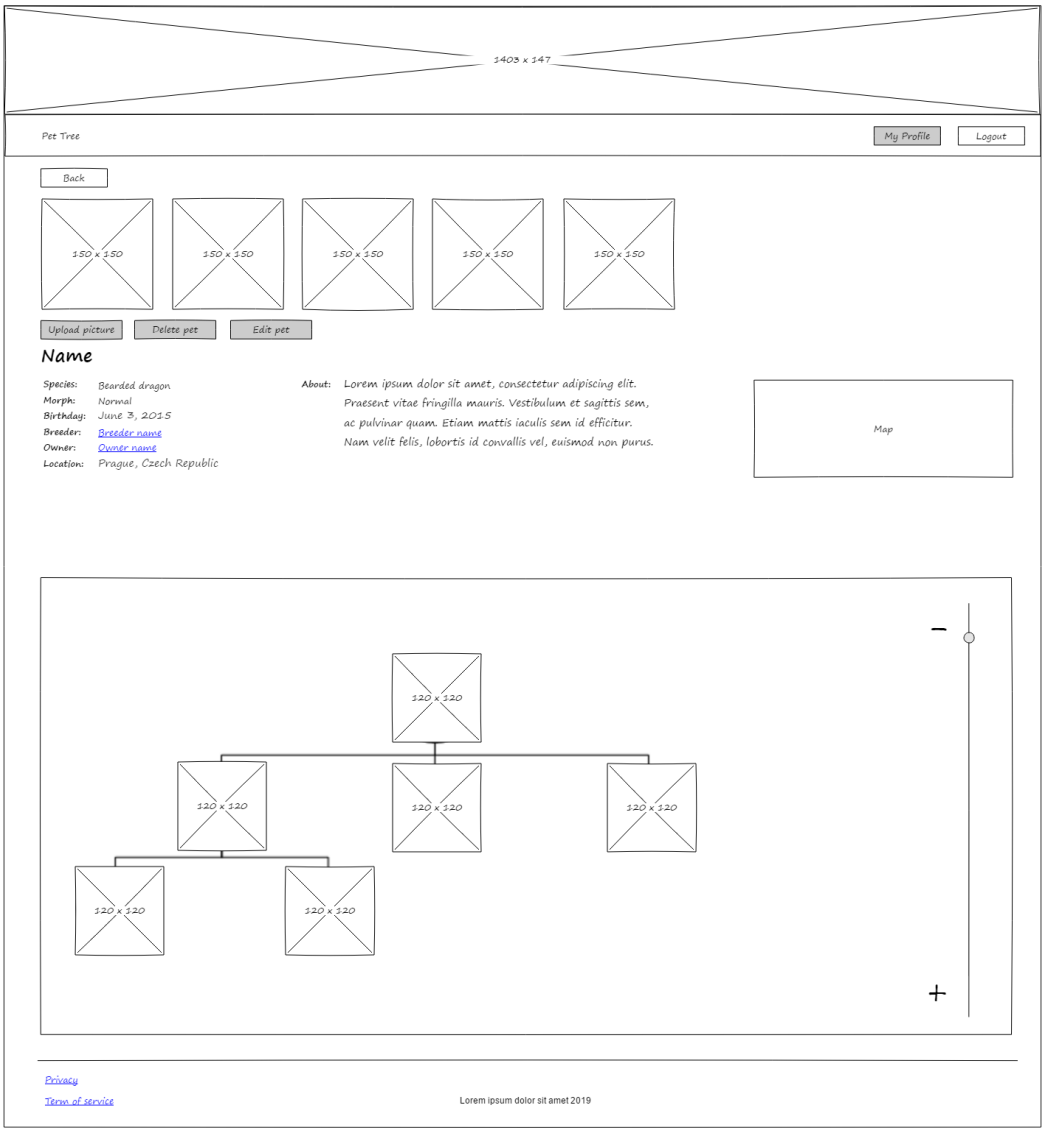


Obrázek 4 - Drátový model editace profilu uživatele (zdroj: autorka)

V dialogu pro editaci profilu bude možné kromě editace všech údajů nastavit email jako soukromý. Pole pro adresu bude stejně jako na úvodní stránce ve filtru napojené na vyhledávací API Mapbox a bude omezené na město i stát. Pokud uživatel není chovatelem, zobrazí se tlačítko „Stát se chovatelem“, jinak se dále zobrazí sekce informací o chovateli.

Na konci dialogu bude možné kompletně smazat účet včetně svých mazlíčků. Zmizí tak i jako rodiče u mazlíčků cizích.

4.2.1.4 Detail mazlíčka

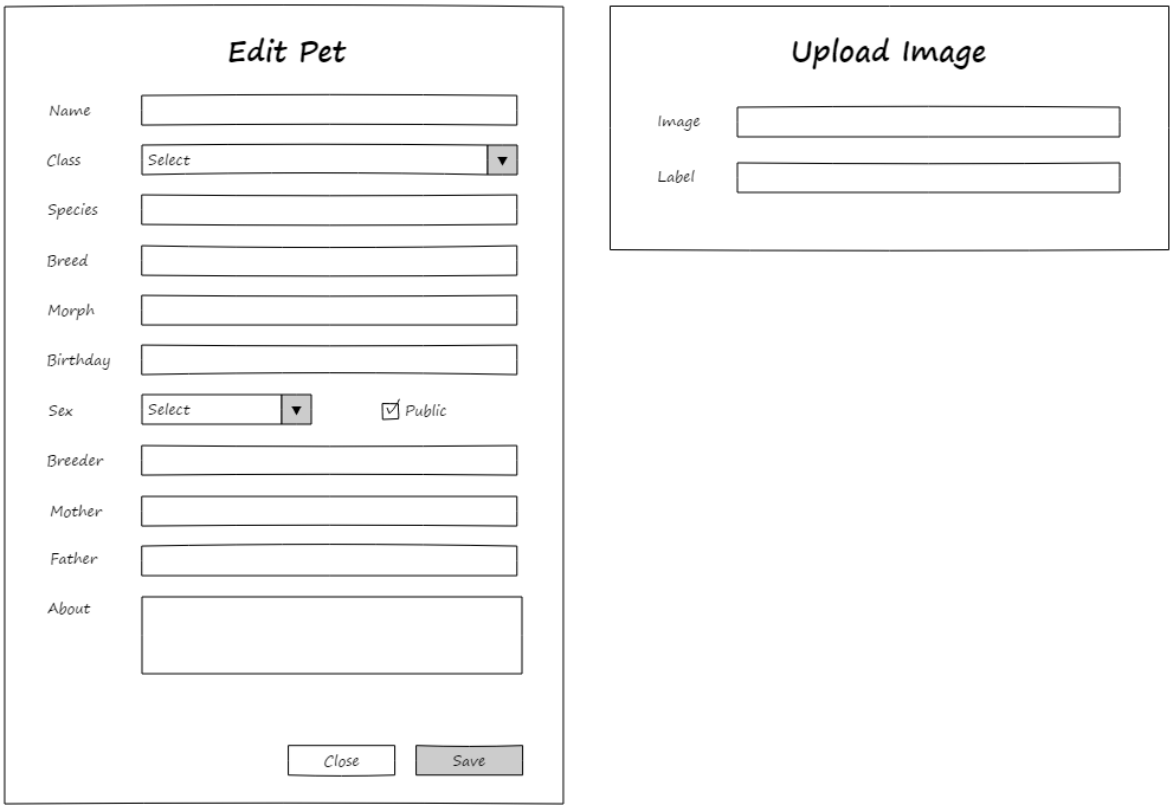


Obrázek 5 - Drátový model detailu mazlíčka (zdroj: autorka)

Na obrazovce detailu mazlíčka budou zobrazené všechny jeho údaje, včetně přibližné adresy majitele, odkaz na profil majitele, chovatele a tlačítko zpět.

Pro zvíře vlastněné přihlášeným uživatelem bude možné nahrát maximálně pět obrázků, upravovat všechny jeho údaje či ho celého smazat.

Níže se pak zobrazí vygenerovaný rodokmen s ovládacím prvkem pro omezení počtu zobrazených generací.



Obrázek 6 - Drátový model editace mazlíčka a nahrání obrázku (zdroj: autorka)

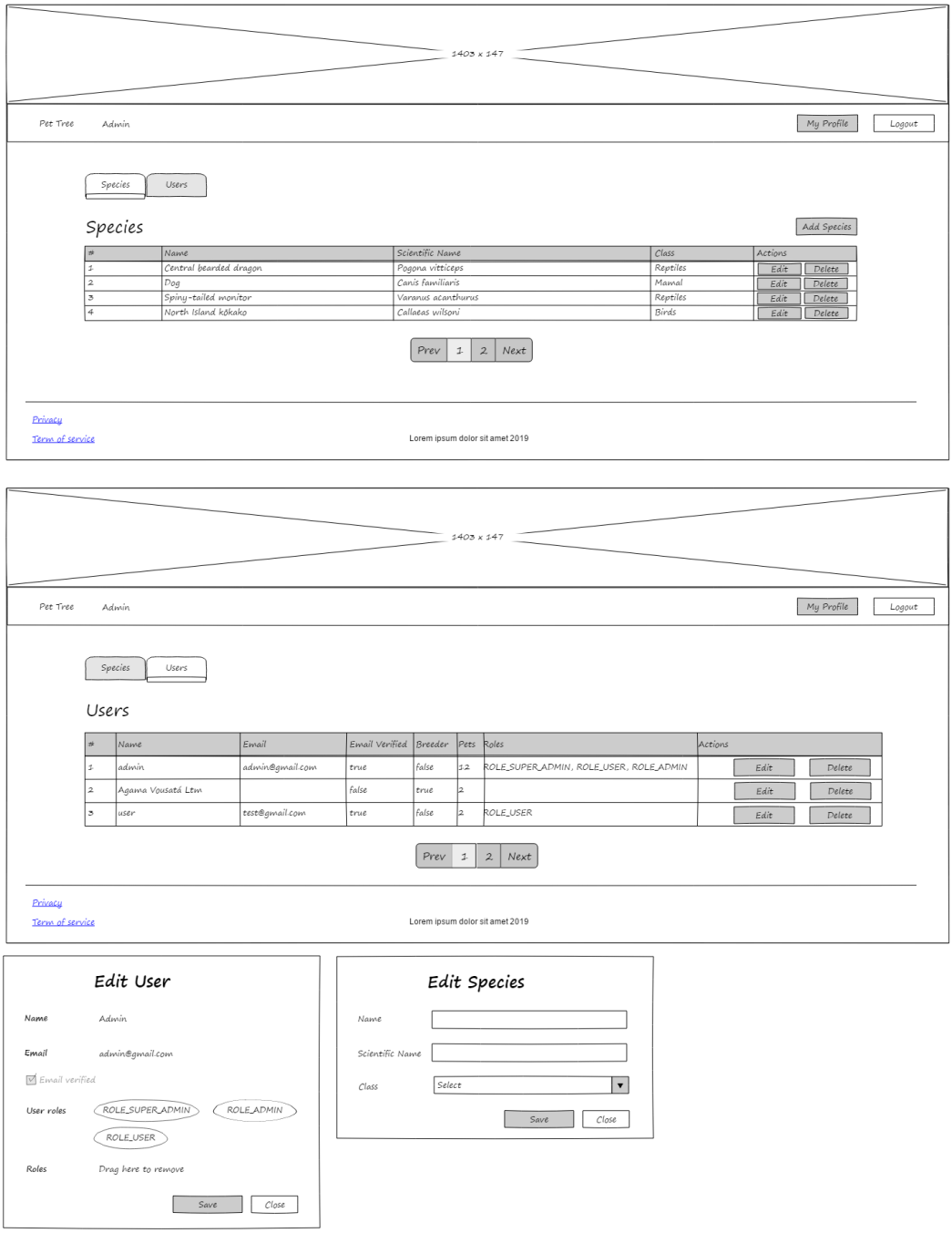
V dialogu editace mazlíčka, který je stejný pro přidávání, bude umožněno upravovat všechny jeho údaje.

Pole „druh“ nebude možné vyplnit, dokud uživatel nevybere pole „třída“. Bude napojené na vyhledávání existujících druhů v databázi filtrované pomocí pole třídy. Pokud druh neexistuje, uživatel jej bude moci vytvořit.

Pole „chovatel“ bude do seznamu pod vstupní pole napovídat jména uživatelů, kteří jsou chovateli. Uživatelům také umožní vytvořit neexistujícího chovatele, který bude ale vytvořen jen jako falešný uživatel bez emailu a nebude možné ho upravovat. Falešný uživatel zde umožní zadat i chovatele, který není v aplikaci registrovaný. Je ale stále nutné umožnit příslušnému chovateli registraci pod stejným jménem a odlišit je. Pro jejich rozeznání se vedle uživatelů s emailem zobrazí zelená značka jako je to běžné u sociálních sítí, například Twitter či Facebook.

Podobné chování bude probíhat i při vybírání matky a otce s tím rozdílem, že pro jejich vyplnění musí uživatel nejprve vyplnit druh a chovatele. Případní „falešní mazlíčci“ se vytvoří pod vybraným chovatelem a nebudou se zahrnovat do výsledků hledání na úvodní obrazovce, aby uživatele nemátli.

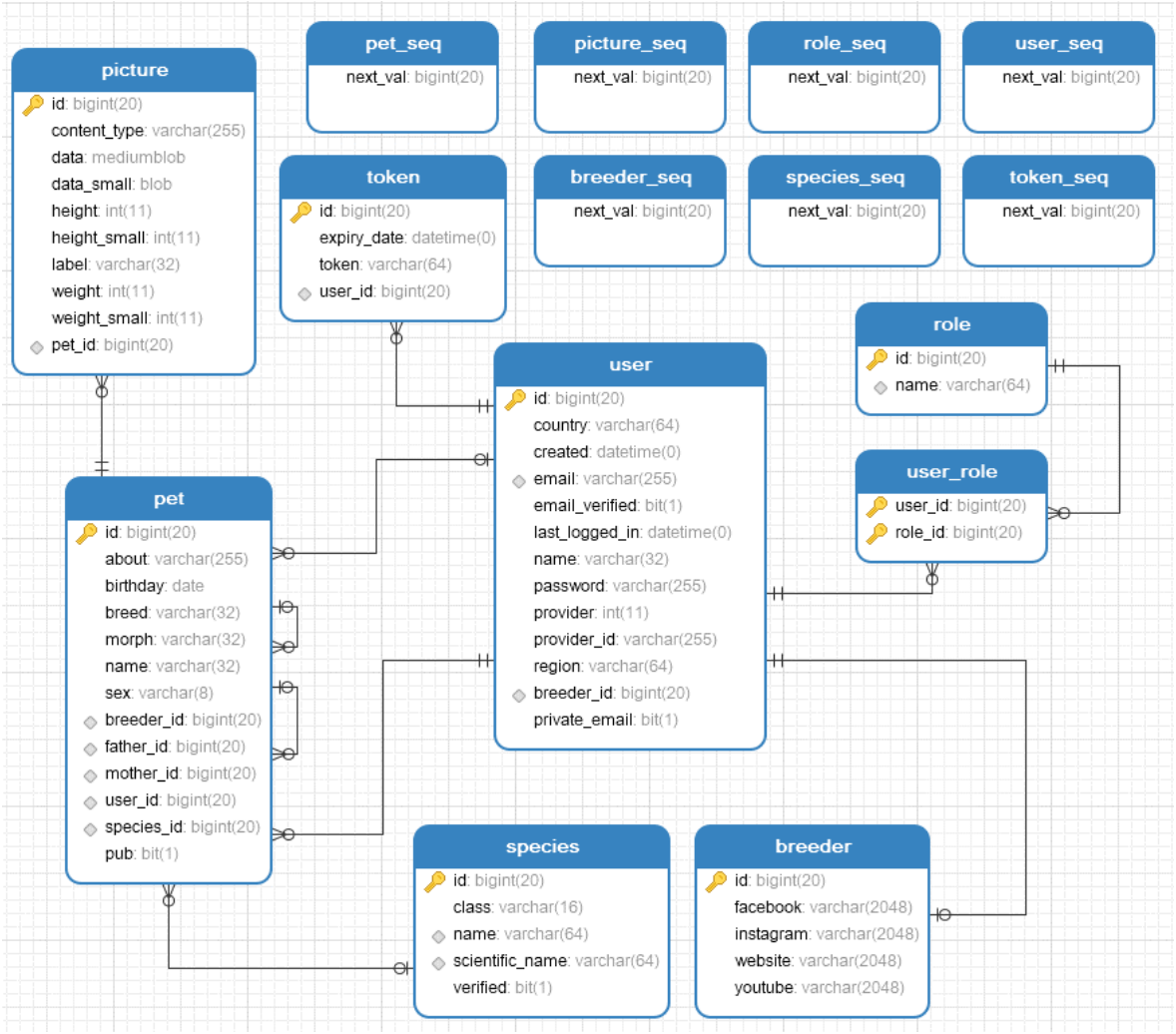
4.2.1.5 Administrace



Obrázek 7 - Drátový model administrace (zdroj: autorka)

Aplikace bude rozlišovat tři uživatelské role: Uživatel, administrátor a super administrátor. Role uživatel je přiřazena při registraci, administrátorské role je nutné přiřadit ručně. Pokud je uživatel administrátorem, bude mu umožněno upravovat, přidávat a mazat druhy zvířat. Super administrátor navíc může upravovat práva všech uživatelů.

4.2.2 Datová struktura



Obrázek 8 - Entity-relationship diagram (zdroj: autorka)

Na obrázku Entity-relationship diagram lze vidět entity a vztahy mezi nimi.

4.2.2.1 User

Entita uživatele uchovává všechny potřebné údaje pro přihlášení a adresu rozdělenou do dvou sloupců: stát a region. Tabulka porušuje pravidlo atomicity. Správně by měla existovat další tabulka adresy propojená s uživatelem, nicméně pro jednodušší práci bylo vybráno toto řešení.

4.2.2.2 Breeder

Chovatel s dalšími atributy pro kontakt rozšiřuje vlastnosti uživatele.

4.2.2.3 Pet

Každý mazlíček má jednoho vlastníka, ostatní vztahy jsou nepovinné. Entita obsahuje dva vztahy sama na sebe, které představují její rodiče. Vztahy mezi rodiči se pro zjednodušení neřeší, protože zvířata obvykle nemají dlouhodobé vztahy. Mazlíček ve vztahu s obrázky je omezen na 0-5.

4.2.2.4 Species

Entita druhu zvířete slouží hlavně k filtrování a kategorizaci mazlíčků.

4.2.2.5 Picture

Obrázek je propojený pouze s mazlíčkem, v budoucnu by se ale mohl použít pro více entit.

4.2.2.6 Token

Token je použit k identifikaci uvnitř emailů k jejich ověření po registraci a resetu hesla. Má platnost hodinu.

4.2.2.7 Role

Enumerační entita nabývající následujících tří hodnot:

- ROLE_USER
- ROLE_ADMIN
- ROLE_SUPER_ADMIN

4.2.2.8 User role

Entita propojující role a uživatele ve vztahu M:N.

4.2.3 REST API

Pro komunikaci mezi backendem a frontendem je třeba nejprve navrhnout REST API. Následují výčty CRUD metod a jejich identifikátory (URI) rozdělené podle kontrolerů, do kterých budou následně implementovány.

4.2.3.1 AuthController

Resources související s přihlášením a registrací uživatele.

- POST /login – Přihlášení uživatele do aplikace na základě emailu a hesla. Jako odpověď vrací JWT token s dobou expirace.
- POST /signup – Registrace uživatele do aplikace. Přijímá jméno, email, heslo, potvrzení hesla a věk, který nejprve validuje a dále neukládá. Po úspěšném uložení do databáze se odešle potvrzovací email. Vrací odpověď 201 created.
- GET /verify-account – Ověření tokenu z potvrzovacího emailu. Pokud je token validní, účet se označí jako ověřený a token se smaže.
- GET /regenerate-token – Resource přijímá token. Pokud v databázi existuje, vygeneruje nový, starý smaže a odešle nový potvrzovací email.
- GET /request-password-change – Resource přijímá email. Pokud v databázi existuje, vygeneruje token a odešle email s odkazem na změnu hesla.
- GET /validate – Validace tokenu vygenerovaného pro změnu hesla. Po úspěšné validaci se token smaže. Vrací odpověď s id uživatele.
- PUT /change-password – Změna hesla uživatele na základě id, nového hesla a jeho potvrzení. Validuje se rovnost hesel a existence id uživatele.

4.2.3.2 AuthenticatedUserController

Resources související s přihlášeným uživatelem. Vyžadují token v hlavičce Authorization a jsou omezené na roli ROLE_USER.

- GET /user/me – Kompletní detail uživatele.
- PUT /user/me/edit – Editace uživatele. Přijímá existujícího uživatele a po úspěšném uložení ho vrací zpět. Ověří, zda id uživatele souhlasí s id z tokenu v hlavičce Authorization a při neshodě vrací HTTP status 403 Forbidden.
- PUT /user/me/change-password – Změna hesla vyžaduje heslo původní nové a potvrzující. Validuje se správnost původního hesla a rovnost nového s potvrzujícím.
- GET /user/me/breeder – Změna uživatele na chovatele. Vytvoří se nová entita chovatele a nastaví se uživateli.

- DELETE /user/me/delete – Kompletní smazání uživatele včetně jeho mazlíčků. Nejprve ale odstraní vazby, které má jako chovatel s jinými mazlíčky a které nevlastní. Vazby mazlíčků uživatele na děti a rodiče je také nutné odstranit.

4.2.3.3 AdminController

Resources pro správu druhů zvířat a rolí uživatelů. Vyžadují token v hlavičce Authorization.

- PUT /admin/species/{id} – Editace existujícího druhu zvířete. Vyžaduje roli ROLE_ADMIN.
- DELETE /admin/species/{id} – Smazání existujícího druhu zvířete. Vyžaduje roli ROLE_ADMIN.
- GET /admin/users – Stránkovaný seznam uživatelů. Požaduje roli ROLE_SUPER_ADMIN.
- GET /admin/users/{id} – Editace rolí uživatelů. Požaduje roli ROLE_SUPER_ADMIN.

4.2.3.4 BreederController

Resources související s chovateli. Vyžadují token v hlavičce Authorization a jsou omezené na roli ROLE_USER.

- GET /breeders/{name} – Vyhledávání seznamu chovatelů podle jména.
- POST /breeders – Vytvoření falešného chovatele. Přijímá entitu user, která má nastavené pouze jméno a po uložení ji navrátí zpět.

4.2.3.5 PetController

Resources související s mazlíčky. Pouze GET metody nevyžadují token v hlavičce Authorization a nejsou omezené na roli ROLE_USER, tím pádem jsou i dostupné pro nepřihlášené uživatele.

- GET /pets – Stránkované hledání mazlíčků podle filtru s parametry věk, region, stát, pohlaví, třída, „query“ (společné pole pro jméno, druh, morfologický druh a plemeno), stránka a velikost stránky.

- GET /pets/search/{name} – Hledání mazlíčka do pole rodiče podle pohlaví (určuje, zda hledáme matku či otce), vlastníka (představuje chovatele mazlíčka, kterému patří id) a id mazlíčka pro kterého rodiče hledáme.
- GET /pets/owner/{ownerId} – Stránkovaný seznam mazlíčků podle vlastníka pro profil uživatele.
- GET /pets/{id} – Kompletní detail mazlíčka.
- GET /pets/tree/{id} – Data pro graf rodokmene. Přijímá maximální počet generací dětí a rodičů.
- POST /pets – Vytváření mazlíčka.
- PUT /pets/{id} – Editace mazlíčka. Zde se validuje, zda id uživatele z tokenu sedí s vlastníkem mazlíčka.
- DELETE /pets/{id} – Mazání mazlíčka. Je zde stejná validace jako u editace.
- POST /pets/{id}/picture – Nahrání obrázků mazlíčka. Přijímá jak původní zkomprimovaný obrázek na 1 MB, tak oříznutý pro náhledy.
- DELETE /pets/{id}/picture/{pictureId} – Smazání obrázku mazlíčka.

4.2.3.6 SpeciesController

Resources související s druhy zvířat.

- GET /species – Stránkovaný seznam druhů zvířat.
- GET /species/search/{name} – Hledání druhu podle jména.
- GET /species/{id} – Kompletní detail druhu.
- GET /species/classes – Seznam enumerovaných tříd.
- POST /species – Vytvoření druhu. Požaduje token v hlavičce Authorization a je omezené na roli ROLE_USER.

4.2.3.7 UserController

Jediný resource GET /user /{name} pro hledání uživatele podle jména. Slouží pro veřejný profil, proto nevrací email, pokud je nastaven jako soukromý.

4.3 Implementace

4.3.1 Založení projektu

Front-end aplikace byl vyvinut pomocí frameworku Angular v jazyce Typescript, backend v jazyce Java. Pro obě části bylo využito vývojové prostředí IntelliJ IDEA, které je dostupné pro studenty zdarma v plné verzi.

Na spuštění backendu je potřeba nejprve nainstalovat Java SDK 14, IntelliJ IDEA a MariaDB. V MariaDB je potřeba nastavit přihlašovací údaje pro následující konfiguraci v aplikaci. Díky Použití Spring Bootu není potřeba instalovat aplikační server, jelikož je již součástí závislosti `spring-boot-starter-web`. Při použití nástroje Spring Initializr není nutné zvlášť instalovat ani Maven. Projekt bude obsahovat maven wrapper, který ho sám stáhne a nainstaluje při spuštění. Pro frontend část je pak třeba nainstalovat Node.js a Angular CLI.

Pro založení projektu pro backend byl využit Spring Initializr. Je to nástroj pro rychlé založení Spring Boot projektu a je podporován i některými IDE včetně IntelliJ. Po nastavení základních parametrů a závislostí se automaticky vygenerovalo `pom.xml` a základní třída pro aplikace.

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.2.6.RELEASE</version>
    <relativePath/>
</parent>
<groupId>cz.vilhelmova</groupId>
<artifactId>pet-tree</artifactId>
<version>1.0</version>
<packaging>jar</packaging>
<name>pet-tree</name>

<properties>
    <java.version>14</java.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-
actuator</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-
jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.mariadb.jdbc</groupId>
        <artifactId>mariadb-java-client</artifactId>
        <version>2.7.0</version>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <scope>provided</scope>
    </dependency>
</dependencies>
```

Kód 1 - Základní závislosti v pom.xml

Kromě základních závislostí pro webovou aplikaci a připojení k databázi byla použita také knihovna „Spring Boot Actuator“ pro monitorování stavu aplikace a „Project Lombok“, který nahrazuje běžné části kódu anotacemi a generuje jej až při kompilaci. Kód se tak stává kratší a čitelnější. (29), (30)

Pro konfiguraci připojení a způsobu komunikace s databází byl vytvořen soubor `application.yml` pod složkou `src/main/resources` s potřebnými parametry.

```yaml
server:
  servlet:
    contextPath: /pet-tree/api
spring:
  datasource:
    url: jdbc:mariadb://localhost:3306/pet_tree?useSSL=false
    username: user
    password: password
  jpa:
    hibernate:
      ddl-auto: update
      naming-strategy: org.hibernate.cfg.ImprovedNamingStrategy
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MySQL5InnoDBDialect
```

Kód 2 - Základní konfigurace application.yml

Front-end část projektu byla vygenerována pomocí Angular CLI a v souboru `environment.ts` vytvořena konfigurace pro volání backend části.

```typescript
export const environment = {
  production: false,
  server: 'http://localhost:8080/',
  apiUrl: 'pet-tree/api/'
};
```

Kód 3 - Konfigurace frontend aplikace

4.3.2 Přihlášení a základní zabezpečení aplikace

Přihlášení bylo inspirováno tutoriálem „Spring Boot OAuth2 Social Login with Google, Facebook, and Github“, byla ale použita jiná novější knihovna pro JWT token. (31), (32), (33)

Při vývoji vzešel nedostatek splnění GDPR při přihlášení pomocí Google a Facebook účtu. Pro znemožnění založení účtu osobám mladší 18 let je nutné znát datum narození. Současná implementace napojení na Google API tyto informace neposkytuje, proto byla funkce v rámci této práce vypnuta.

4.3.3 Generování rodokmenu

Knihovna ngx-graph vykreslující graf konzumuje data pomocí 3 polí:

- Uzly – id, jméno, pohlaví a jeden zmenšený obrázek
- Hrany – id hrany, dítěte a rodiče

- Klastry – id (číslo generace) a pole id mazlíčků

(26)

Ve zdroji pro detail mazlíčka byl nejprve rekurzí vypočítán počet generací na obě strany. Hloubka rekurze byla omezena na 8 zanoření. Počty generací jsou pak použity pro ovládací prvky na klientské části.

Pomocí rodičovských atributů mazlíčka, které jsou nastaveny na „FetchType.LAZY“ (data se stahují na vyžádání), je aplikace schopna omezit hloubku dotazu výše získaným počtem generací.

4.3.4 Problémy a řešení

4.3.4.1 Problém s mazáním uživatele

Při implementaci smazání účtu byl objeven problém u komplikovaných vztahů mezi entitami. Pokud existovali dva uživatelé, kdy mazlíček jednoho uživatele byl ve vztahu s druhým uživatelem jako jeho chovatel, a mazlíček chovatele byl nastaven jako jeho rodič, pak smazáním prvního uživatele se kaskádovitě smazala celá databáze.

Problém způsobovalo nastavení „CascadeType.ALL“. Řešením bylo nastavit pouze „CascadeType.REMOVE“ v entitě uživatele jak na pole mazlíčků tak i chovatele, a před smazáním uživatele nejprve ručně odstranit vztahy mezi entitami odstraněním cizích klíčů.

4.3.4.2 Problém s nahráváním velkých obrázků

Při nahrávání obrázků mazlíčků vznikl problém s nastavením maximální velikosti souboru Spring Frameworku, který je ve výchozím stavu 1 MB. Limit je sice možné zvýšit, ale není to příliš rozumné řešení. Byla proto využita knihovna „ng2-img-max“ pro automatickou kompresi obrázků do dané velikosti. Knihovna je omezená na typy jpg a png. (34)

4.3.5 Nasazení

Pro vystavení aplikace byl využit virtuální server s operačním systémem CentOS Linux 8 na doméně pet-tree.eu. Po instalaci MariaDB a OpenJDK stejně jako na lokální stanici bylo třeba doinstalovat ještě Apache HTTP server. Ve složce `/var/www/html` byl vytvořen soubor `.htaccess` s následujícím nastavením:

```
RewriteEngine On

# If an existing asset or directory is requested go to it as it i
RewriteCond %{DOCUMENT_ROOT}%{REQUEST_URI} -f [OR]

RewriteCond %{DOCUMENT_ROOT}%{REQUEST_URI} -d

RewriteRule ^ - [L]


# If the requested resource doesn't exist, use index.html

RewriteRule ^ /index.html
```

Kód 4 - .httpaccess

Pro nasazení klientské části aplikace byla nejprve nastavena produkční konfigurace v souboru `environment.prod.ts` která se při spuštění příkazu `ng build –prod` nahradí za lokální konfiguraci. (15)

```
export const environment = {
  production: true,
  server: 'https://pet-tree.eu/',
  apiUrl: 'api/',
  oath2RedirectUrl: 'https://pet-tree.eu/oauth2/redirect',
  mapbox: {
    accessToken: token
  }
};
```

Kód 5 - produkční konfigurace

Obsah `/dist/pet-tree-fe/browser` byl zkopírován do složky `/var/www/html` na server.

Pro nasazení backendové části byl ručně vyměněn konfigurační soubor za produkční a sestavený jar Mavenem zkopírován na server. Díky použití Spring Bootu stačilo tento archiv aplikace spustit příkazem `java -jar pet-tree-1.0.jar`.

4.4 Testování

Po implementaci je třeba ověřit funkčnost a srozumitelnost aplikace pro uživatele. Testování funkčnosti bylo prováděno během vývoje mezi jednotlivými částmi. Při vývoji bylo také využito rozšíření v IntelliJ IDEA SonarLint pro Javu, které poskytuje okamžitou zpětnou vazbu. Pomáhá najít chyby, bezpečnostní slabiny, překlepy nebo neoptimální kód z pohledu udržovatelnosti.

Důkladně bylo třeba otestovat hlavně mazání různých entit a kontrolovat ručně databázi pomocí nástroje DBeaver, zda se nesmazala data navíc. Další důkladné testování

vyžadovaly všechny role, aby se uživatel nedostal do míst, kam nemá nebo naopak. Pro splnění GDPR bylo také třeba ověřit správnou funkčnost funkce nastavení emailu jako privátní.

Pro ověření srozumitelnosti aplikace byl zvolen jednoduchý způsob testování, kdy po volné navigaci aplikací byly položeny následující 2 otázky:

1. Je pro vás aplikace srozumitelná?
2. Co byste změnili?

Jeden z uživatelů byl zmaten na úvodní obrazovce a nechápal filtr mazlíčků. Navrhl zvýraznit filtr a obsah.

Dalším uživatelem bylo navrženo dát filtru nadpis a jeho zúžení. Mezi další návrhy byl zmíněn například zapamatování nastavení stránkování uživatele, které se momentálně vrací pokaždé do výchozího nastavení.

Po implementaci návrhů prvních uživatelů nebyl zaznamenán žádný zásadní problém s porozuměním aplikace.

Při testování na průměrně rychlém připojení měla aplikace problémy s rychlostí načítání. Je způsobené načítáním všech obrázků mazlíčka společně v detailu. Obrázky se současně načítají přes textový formát, který je méně efektivní pro ukládání dat. Jeden dotaz může díky tomu dosáhnout více než 5 MB.

5 Výsledky a diskuse

Výsledkem práce je funkční aplikace dle zadaných požadavků. Z návrhu byla vypuštěna jen jedna funkce přihlášení pomocí Google a Facebook účtů. Funkce ale není zásadní k používání aplikace.

V dalším vývoji by bylo vhodné se soustředit na zlepšení bezpečnosti, výkonu aplikace a znovu zprovoznění přihlášení pomocí účtů sociálních sítí. Dalo by se zvážit také například použití GraphQL databáze místo relační pro načítání grafů rodokmene, pokud by změna umožnila zrychlení a větší škálovatelnost.

Pro lepší zabezpečení by bylo možné omezit počet pokusů na přihlášení za určitý interval a při větším počtu neúspěšných pokusů zaslat varovný email. Další způsob zabezpečení aplikace může být reCAPTCHA při přidáváním podezřele velkého množství dat v krátkém intervalu.

Z obdržených návrhů by se dalo přidělat zapamatování nastavení stránkování pomocí local storage (úložiště v prohlížeči klienta), změnu vlastníka mazlíčka a zvláštní zdroj pro načítání obrázků.

Dále by bylo vhodné doplnit implementaci unit testů, které pomáhají při vývoji zejména dalších funkcí. Ověřují, že nedošlo k nechtěným změnám v chování aplikace. Je možné tak odhalit zásadní chyby před nasazením a zveřejněním aplikace. Také by bylo v budoucnu dobré mít testovací verzi aplikace, která je dostupná na neoficiální adrese, aby tak testování neohrozilo data uživatelů v produkci.

Pro rozšíření aplikace by bylo dobré integrovat sdílení na sociální sítě.

6 Závěr

Cílem této práce byla analýza, návrh a implementace webové aplikace „Pet tree“ sloužící jako databáze domácích mazlíčku umožňující vyhledání vhodného partnera, automatického vytváření jejich rodokmenů a pomoci tak vyřešit problém s neexistující možností najít původ velkého množství domácích zvířat.

Pro seznámení s použitými technologiemi byla nejprve představena teoretická východiska. Pro serverovou část aplikace byl představen jazyk Java, s ní spojené frameworky a technologie Spring Framework, a nad ním postavený Spring Boot s automatickou konfigurací pro snadnější vývoj, Spring Data JPA a Hibernate pro práci s databází MariaDB, Apache Maven pro správu a sestavování aplikace, JWT pro zabezpečení aplikace a REST API jako rozhraní, přes které je backend vystaven. Dále pro klientskou část aplikace byl představen pojem Jednostránková aplikace, jazyk TypeScript a v něm implementovaný Angular pro snadný vývoj, a SVG pro vektorové obrázky na webu.

Obsahem vlastní práce byla nejprve analýza problému, podobných webových aplikací, dále pak následovalo určení požadavků a jejich analýza. Na základě analýzy bylo dále navrhnuto řešení v podobě drátových modelů, datové struktury a REST API. Po návrhu následovala vlastní implementace, kde byl nejprve založen projekt, přihlášení jakožto hlavní bezpečnostní jádro aplikace, generování rodokmenu, které představuje hlavní funkční požadavek pro řešení původního problému a některé z větších vyskytnutých problémů v průběhu vývoje. Dále bylo popsáno nasazení aplikace na virtuální server a následné testování. Nakonec byla aplikace zhodnocena a byl navržen její následný vývoj.

7 Seznam použitých zdrojů

1. LOY, Marc, Patrick NIEMEYER a Daniel LEUCK. Learning Java: An Introduction to Real-World Programming with Java. 5th edition. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, 2020. ISBN 978-1-492-05627-0.
2. Index | TIOBE - The Software Quality Company. Home | TIOBE - The Software Quality Company [online]. Victory House II, Esp 401, 5633 AJ Eindhoven, The Netherlands: TIOBE Software BV, 2020 [cit. 2020-6-15]. Dostupné z: https://www.tiobe.com/tiobe-index/
3. KUNČAR, Petr. Spring - IoC Kontejner. Itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. [online]. Praha: itnetwork.cz, 2020 [cit. 2020-5-25]. Dostupné z: https://www.itnetwork.cz/java/jee/spring-ioc-kontejner
4. BLOCH, Joshua. Effective Java. Third Edition. Boston: Addison-Wesley Professional, 2017. ISBN 9780134686097.
5. GLOVER, Andrew. Spring Boot in Action. 20 Baldwin Rd, Shelter Island, NY 11964: Manning Publications Co., 2016. ISBN 9781617292545.
6. FADATARE, Ramesh. What Is the Difference Between Hibernate and Spring Data JPA? - DZone Java. DZone: Programming & DevOps news, tutorials & tools [online]. Devada: DZone, 2019, 31 Dec 2018 [cit. 2020-8-9]. Dostupné z: https://dzone.com/articles/what-is-the-difference-between-hibernate-and-sprin-1
7. BHARATHAN, Raghuram. Apache Maven Cookbook. Livery Place 35 Livery Street Birmingham B3 2PB, UK.: Packt Publishing, 2015. ISBN 978-1-78528-612-4.
8. Pokyny k návrhu API - Best practices for cloud applications | Microsoft Docs. Vývojářské nástroje, technická dokumentace a příklady kódování | Microsoft Docs [online]. Redmond: Microsoft, 2020, 1.12.2018 [cit. 2020-2-6]. Dostupné z: https://docs.microsoft.com/cs-cz/azure/architecture/best-practices/api-design
9. MALÝ, Martin. REST: architektura pro webové API - Zdroják. Zdroják - o tvorbě webových stránek a aplikací [online]. Budějovická 1550/15a, 140 00 Praha 4 Czech Republic: Devel.cz Lab, 2019, 3.8.2009 [cit. 2020-2-25]. Dostupné z: https://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/
10. HTTP response status codes - HTTP | MDN. MDN Web Docs [online]. Mountain View, California: Mozilla Corporation, 2020, 8 Nov 2020 [cit. 2020-10-25]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Status
11. JSON Web Tokens - jwt.io [online]. Bellevue: Auth0, 2014 [cit. 2020-2-23]. Dostupné z: https://jwt.io/
12. MariaDB Foundation - MariaDB.org [online]. Helsinki: MariaDB Foundation, 2009 - 2020 [cit. 2020-3-25]. Dostupné z: https://mariadb.org/
13. SKÓLSKI, Paweł. Single-page application vs. multiple-page application | by Neoteric | Medium. Software House That Helps You Innovate - Neoteric [online]. Marynarki Polskiej 163 80-868 Gdańsk, Poland: Neoteric Sp. z o.o., 2020, 1 Dec 2016 [cit. 2020-8-25]. Dostupné z: https://neoteric.eu/blog/single-page-application-vs-multiple-page-application/?utm_source=medium.com&utm_medium=social&utm_content=neo&utm_campaign=blog

14. TypeScript: Typed JavaScript at Any Scale. [online]. Redmond, Boston, SF & NYC: Microsoft, 2020 [cit. 2020-10-17]. Dostupné z: https://www.typescriptlang.org/
15. Angular [online]. California: Google, 2020 [cit. 2020-5-10]. Dostupné z: https://angular.io/
16. HTML SVG. W3Schools Online Web Tutorials [online]. Sandnes, Norsko: Refsnes Data, 2020 [cit. 2020-11-5]. Dostupné z: https://www.w3schools.com/html/html5_svg.asp
17. Pedigree Pets | The premier place to buy and sell a pedigree pet. [online]. Sterling House, 27 Hatchlands, Road, Redhill, Surrey, RH1 6RW: Pedigree Pets Limited, 2018 [cit. 2020-1-10]. Dostupné z: https://www.pedigree-pets.com/
18. CALIFORNIA ACADEMY OF SCIENCES. INaturalist API [online]. Golden Gate Park 55 Music Concourse Drive San Francisco, CA 94118: California Academy of Sciences, 2020 [cit. 2020-5-20]. Dostupné z: https://api.inaturalist.org/v1/docs/
19. SOCIÉTÉ DES PRODUITS NESTLÉ S.A. Petfinder API Support Documentation | Petfinder [online]. Avenue Nestlé 55 1800 Vevey: Société des Produits Nestlé S.A., 2020 [cit. 2020-5-20]. Dostupné z: https://www.petfinder.com/developers/v2/docs/
20. Home | Jawg Maps [online]. Bagneux: Jawg, 2020 [cit. 2020-9-10]. Dostupné z: https://www.jawg.io/en/
21. HERE Technologies | The world's #1 location platform [online]. Eindhoven: HERE, 2020 [cit. 2020-9-10]. Dostupné z: https://www.here.com/
22. Mapbox [online]. San Francisco: Mapbox, 2020 [cit. 2020-9-10]. Dostupné z: https://www.mapbox.com/
23. LocationIQ - Free & Fast Geocoding, Reverse Geocoding and Maps service [online]. Hyderabad: Unwired Labs, 2020 [cit. 2020-9-10]. Dostupné z: https://locationiq.com/
24. BASIC PRIMITIVES INC. Basic Primitives Diagrams - Data visualization diagramming components for dependencies visualization and analysis. [online]. 524 Ridelle Avenue, M6B 1K8, Toronto, ON, Canada: Basic Primitives, 2013 - 2020 [cit. 2020-6-4]. Dostupné z: https://www.basicprimitives.com/
25. XIE, Ziyu. Ngx-echarts-demo documentation [online]. New York: Xie, Ziyu, 2017 [cit. 2020-6-4]. Dostupné z: https://xieziyu.github.io/ngx-echarts/api-doc/
26. GEORGIEV, Marjan a Austin. GitHub - swimlane/ngx-graph: Graph visualization library for angular [online]. Swimlane, 2016 [cit. 2020-6-4]. Swimlane. Dostupné z: https://github.com/swimlane/ngx-graph
27. General Data Protection Regulation | GDPR.cz. GDPR | Obecné nařízení o ochraně osobních údajů — prakticky [online]. Praha: GDPR.cz, 2020 [cit. 2020-10-1]. Dostupné z: https://www.gdpr.cz/gdpr/heslo/gdpr/
28. Osobní údaje | GDPR.cz. GDPR | Obecné nařízení o ochraně osobních údajů — prakticky [online]. Praha: GDPR.cz, 2020 [cit. 2020-10-1]. Dostupné z: https://www.gdpr.cz/gdpr/heslo/osobni-udaje/
29. Spring | Home [online]. Palo Alto, California, United States: VMware, 2020 [cit. 2020-11-24]. Dostupné z: https://spring.io/
30. ZWITSERLOOT, Reinier a Roel SPILKER. Project Lombok [online]. The Project Lombok Authors, 2020 [cit. 2020-11-24]. Dostupné z: https://projectlombok.org/
31. SINGH, Rajeev. Spring Boot OAuth2 Social Login with Google, Facebook, and Github - Part 1 | CalliCoder. CalliCoder | Programming, Web & Desktop App Development Tutorials [online]. Bangalore, India: CalliCoder, 2019, 6 Nov 2018

[cit. 2020-5-7]. Dostupné z: https://www.callicoder.com/spring-boot-security-oauth2-social-login-part-1/
32. SINGH, Rajeev. Spring Boot OAuth2 Social Login with Google, Facebook, and Github - Part 2 | CalliCoder. CalliCoder | Programming, Web & Desktop App Development Tutorials [online]. Bangalore, India: CalliCoder, 2019, 7 Nov 2018 [cit. 2020-5-7]. Dostupné z: https://www.callicoder.com/spring-boot-security-oauth2-social-login-part-2/
33. SINGH, Rajeev. Spring Boot OAuth2 Social Login with Google, Facebook, and Github - Part 3 | CalliCoder. CalliCoder | Programming, Web & Desktop App Development Tutorials [online]. Bangalore, India: CalliCoder, 2019, 8 Nov 2018 [cit. 2020-5-7]. Dostupné z: https://www.callicoder.com/spring-boot-security-oauth2-social-login-part-3/
34. BENEDIKT, Berger. GitHub - bergben/ng2-img-max: Angular 2 module to resize images down to a certain width and height or to reduce the quality to fit a certain maximal filesize - all in the browser. [online]. Berger Benedikt, 2016 [cit. 2020-11-15]. Dostupné z: https://github.com/bergben/ng2-img-max

8 Přílohy

Seznam příloh na CD:

- Zdrojové kódy frontend a backend části aplikace
- SQL script pro vytvoření schéma a databázových tabulek
- Kopie bakalářské práce