FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Databáze zvířat pro neziskovou organizaci
Student:	Tomáš Taro
Vedoucí:	Ing. Lukáš Maleček
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce zimního semestru 2020/21

Pokyny pro vypracování

Cílem práce je vytvořit novou verzi webové aplikace Plemenná kniha (databázi zvířat), která slouží k evidenci informací o zvířatech v aktivním chovu, evidenci odchovů, schvalování vrhů, pomáhá členům neziskové organizace v plánování chovu a umožňuje jednoduché generování průkazů původu pro odchovy.

* Analyzujte současný stav aplikace, seznamte se s její vnitřní implementací, současnou strukturou databáze a již implementovanými funkcemi.
* Na základě analýzy navrhněte optimálnější strukturu databáze (zejména vyřešte současné duplicitní ukládání dat).
* Vytvořte zcela novou aplikaci pro správu plemenné knihy v PHP s použitím MySQL databáze. V aplikaci využijte současné návrhové a architektonické vzory a postupy.
* Implementujte nové funkce na základě jejich specifikace dodaných vedoucím práce.
* Ověřte správnou funkci aplikace pomocí unit a případně i integračních testů.
* Migrujte data z původní aplikace do nové a navrhněte postup, jak zajistit kontrolu a ověření správnosti importu.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 21. února 2019

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalárska práca

Databáza zvierat pre neziskovú organizáciu

Tomáš Taro

Katedra softwarového inženýrství

Vedúci práce: Ing. Lukáš Maleček

20. mája 2020

Poďakovanie

Touto cestou by som sa chcel poďakovať pánovi Ing. Lukášovi Malečkovi za vedenie a trpezlivosť pri riešení problematiky týkajúcej sa tejto bakalárskej práce. Taktiež by som sa chcel poďakovať rodine za neustálu podporu počas celého štúdia.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval(a) samostatne a že som uviedol(uviedla) všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, v znení neskorších predpisov, a skutočnosť, že České vysoké učení technické v Praze má právo na uzavrenie licenčnej zmluvy o použití tejto práce ako školského diela podľa § 60 odst. 1 autorského zákona.

V Prahe 20. mája 2020 . . . . . .

České vysoké učení technické v Praze

Fakulta informačních technologií

Odkaz na túto prácu

Taro, Tomáš. *Databáza zvierat pre neziskovú organizáciu.* Bakalárska práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2020.

Abstrakt

V niekoľkých vetách zhrňte obsah a prínos tejto práce v slovenčine. Po prečítaní abstraktu by mal čitateľ mať dosť informácií pre rozhodnutie, či Vašu prácu chce čítať.

Klíčová slova Nahraďte zoznamom kľúčových slov v slovenčine oddelených čiarkou.

Abstract

Sem doplňte ekvivalent abstraktu Vašej práce v angličtině.

Keywords Nahraďte zoznamom kľúčových slov v angličtine oddelených čiarkou.

Obsah

Zoznam obrázkov

Úvod

Kapitola 1

Cieľ práce

Teoretická časť práce sa zameria na analýzu súčasnej webovej aplikácie, ktorá pomáha členom organizácie s evidenciou zvierat a ich odchovov. Ďalej budú špecifikované funkčné a nefunkčné požiadavky spolu s používatelskými rolami a prípadmi použitia, na základe ktorých bude navrhnutá architektúra a technológie, ktoré budú použité v novej webovej aplikácii.

Praktická časť práce sa bude zaoberať implementáciou aplikácie na základe výstupov z teoretickej časti práce a migráciou pôvodných dát do novej aplikácie. Nakoniec bude aplikácia otestovaná pomocou integračných testov.

Konečný výstup praktickej časti bude prospešný pre členov organizácie, ktorým sa výrazne zjednoduší a sprehľadní evidencia informacií o zvieratách.

Kapitola 2

Analýza

Táto kapitola sa venuje analýze súčasnej webovej aplikácie, ktorá zahŕňa analýzu jej architektúry a databázového modelu. Následne sú vymedzené jednotlivé funkčné a nefunkčné požiadavky novej webovej aplikácie, na ktoré nadväzuje sekcia používateľských rolí. Záver kapitoly je určený pre analýzu prípadov použití, ktoré vyplývajú z funkčných požiadaviek.

2.1 Analýza súčasnej aplikácie

Táto sekcia sa zaoberá analýzou súčasnej aplikácie, ktorá bola vytvorená pre potreby neziskovej organizácie viesť evidenciu zvierat, vrhov a ich registrácií. Hlavným cieľom analýzy je zistenie súčasného riešenia aplikácie z pohľadu architektúry a štruktúry súčasnej databázy.

2.1.1 Architektúra aplikácie

Samotný kód aplikácie nie je členený – skladá sa z jednej vrstvy – čo znamená, že kód aplikácie zodpovedný za prístup do databázy, logiku aplikácie a zobrazenie aplikácie nie je od seba oddelený.

```
<?php
...
get_header();
if (!is_user_logged_in()) {
  echo ("<h1>Neautorizováno</h1>\n");
} else {
?>
<h1>Detail zvířete</h1>
<br>
<?php
  if (!empty($_REQUEST['id'])) {
    $req_id = $_REQUEST['id'];
    if (is_numeric($req_id)) {
      $sql_potkan = "SELECT * FROM `" . $tabulka_zvirata . "`
      WHERE `id_prvni_verze` = " . $req_id . " ORDER BY `id` DESC";
      $potkani = $wpdb->get_results($sql_potkan, ARRAY_A, 0);
      $zvire = $potkani[0];
      if ($zvire != null) {
?>
  <div id="zvire">
    <fieldset style="
    border: 1px solid #A0A0A0;
    margin: 2px;
    padding: 5px;">
      <legend style="padding: 5px;">Základní informace</legend>
      <span id="popis">Pohlaví:</span><span id="hodnota">
      <?php
        if ($zvire['pohlavi'] == 'M') {
          echo ("kluk");
        }
        else if ($zvire['pohlavi'] == 'F') {
          echo ("holka");
        }
        else {
          echo ("????");
        }
      ?>
      </span><br>
      ...
    </fieldset>
  </div>
```

Obr. 2.1: Ukážka súboru vrhy_detail_zvirete.php

Obrázok **??** obsahuje ukážku súboru súčasnej aplikácie, na ktorom je možné vidieť skôr spomenuté miešanie logiky zodpovednej za funkcionalitu aplikácie (riadok 1–6), kód zodpovedný za prístup do databázy (riadok 10–19) a v neposlednom rade logiku zobrazovania aplikácie (riadok 21–41).

Ako programovací jazyk súčasnej aplikácie bol použítý jazyk PHP, ktorý bol navrhnutý v roku 1994 Rasmusom Lerdorfom. Tento jazyk je univerzálny programovací jazyk na strane servera a je primárne určený na vývoj webových stránok [**?**].

Pre pridávanie, spracovanie a získavanie dát v aplikácii slúži open-source SQL databázový systém nazývaný MySQL, ktorý je vyvíjaný, distribuovaný a podporovaný spoločnosťou Oracle Corporation [**?**].

2.1.2 Databázový model aplikácie

Neoddeliteľnou časťou tejto práce je analýza databázového modelu súčasnej aplikácie. Na základe tejto analýzy bude v navrhnutá lepšia štruktúra databázy tak, aby dáta aplikácie boli jednoznačné a štruktúra databázy nespôsobovala nekonzistentnosť dát v aplikácii.

Databáza súčasnej aplikácie sa skladá zo 6tich tabuliek: `czkp_mimi`, `czkp_vrh`, `czkp_zvire`, `pp_informace`, `pp_miminka` a `pp_zadosti`.

2.1.2.1 Popis tabuliek

V tejto podsekcii sú popísané jednotlivé tabuľky, ich štruktúra a najmä význam stĺpcov, do ktorých sa jednotlivé dáta ukladajú. Pre lepšiu predstavu štruktúry databázového modelu aplikácie sa v podsekcii **??** nachádza logický model databázy v grafickej podobe.

Všetky tabuľky obsahujú umelo-vytvorený unikátny primárny kľúč záznamu zvaný `id`, podľa ktorého sa rozlišujú jednotlivé záznamy.

V nasledujúcich dvoch tabuľkách (`czkp_vrh` a `czkp_zvire`) sa vyskytujú tri rovnaké stĺpce, menovite `id_prvni_verze`, `datum_zaznamu` a `uzivatel`. V stĺpci `id_prvni_verze` je uložené `id` prvej verzie zvieraťa/vrhu. Následne v stĺpci `datum_zaznamu` je uložený dátum vytvorenia záznamu spolu s jeho časom a v stĺpci `uzivatel` identifikátor používateľa, ktorý daný záznam vytvoril. Tento identifikátor ukazuje na systémovú tabuľku `wp_users`, ktorá obsahuje informácie o používateľoch súčasného systému.

Tabuľka czkp_vrh

Táto tabuľka má za úlohu správu dát týkajúcich sa jednotlivých vrhov. Stĺpec `id_prvni_verze` odkazuje na prvý záznam daného vrhu.
V stĺpci `wp_id_majitel` je uložený identifikátor majiteľa vrhu ukazujúci na systémovú tabuľku `wp_users`. Stĺpce `otec` a `matka` slúžia k ukladaniu identifikátorov otca, resp. matky — tento identifikátor pochádza z tabuľky `czkp_zvire`.

Následne sú do tejto tabuľky ukladané aj údaje ako označenie vrhu (v stĺpci `oznaceni`), typ vrhu (`typ_priznani`), línia vrhu (`linie`), gény vrhu (`geny_vrhu`), jeho varietnosť (`varietnost`) či dátum narodenia (`datum_narozeni`). U každého vrhu sa taktiež zaznamenáva chovná stanica (`chov_stanice`), v ktorej sa daný vrh narodil, prípadne kontakt na chovateľa, ktorý je uložený v stĺpci `chov_kontakt`. Dôležité informácie, ako počet narodených a odchovaných mláďat sa ukladajú do stĺpcov `nar_mladat`, prípadne `odchov_mladat`. Počet odchovaných mláďat sa delí na počet odchovaných samcov (`odch_kluci`) a samíc (`odch_holky`). Počet mláďat uvolnených pre chov je možné nájsť v stĺpci `mlad_chovne`. Rozdiel odchovaných mláďat a mláďat uvolnených pre chov sa rovná počtu mláďat daných na maznáčika — tento údaj je možné nájsť v stĺpci `mlad_pet`.

Jednotlivé vrhy môžu byť zaregistrované pod klubom ČKP — tieto registrácie sa taktiež nachádzajú v tabuľke `czkp_vrh`. Pre účely registrácie vrhu slúžia stĺpce `datum_registrace`, v ktorom je uložený dátum registrácie, `reg_cislo_vrhu`, ktorý obsahuje registračné číslo vrhu a `rok_reg`, ktorý značí, v ktorom roku bol vrh zaregistrovaný. Registrátor, ktorý zaregistroval daný vrh, môže pripojiť poznámku počas registrácie vrhu, ktorá je uložená v stĺpci `poznamka_reg`.

Chovateľ daného vrhu môže taktiež vytvoriť poznámku k vrhu — táto poznámka sa následne uloží do stĺpca `poznamka_chov`.

Ako posledný stĺpec nachádzajúci sa v tabuľke `czkp_vrh` je stĺpec `kompletni`, ktorý značí, či o uloženom vrhu sú dostupné kompletné informácie, avšak tento údaj sa v aplikácii nepoužíva.

Tabuľka czkp_zvire

Tabuľka `czkp_zvire` slúži na ukladanie informácií o jednotlivých zvieratách, ktoré sú sledované organizáciou.

Skladá sa zo stĺpca `id_prvni_verze`, ktorý odkazuje na prvý záznam daného zvieraťa v rovnakej tabuľke – čo znamená, že pomocou tohto stĺpca sa dajú zistiť všetky úpravy daného vrhu spustením príslušného SQL príkazu.

Jej obsahom je taktiež stĺpec `datum_zaznamu`, ktorý je nastavený na dátum a čas vloženia záznamu do tabuľky. Stĺpec `uzivatel` obsahuje identifikátor používateľa, ktorý daný záznam vytvoril. Následne sa v tabuľke nachádzajú stĺpce `uzamceny_upravy`, `uzamceny_kdy` a `uzamceni_kdo`, ktoré sa ale v súčasnej aplikácii nevyužívali. Pri zvierati je potrebné ukladať jeho pohlavie – to je uložené v stĺpci `pohlavi`. Dátum narodenia zvieraťa sa nachádza v stĺpci `datum_narozeni`.

Informácie o chovateľovi, resp. majiteľovi zvieraťa sa ukladajú v šiestich stĺpcoch (3 a 3). V stĺpci `chovatel_ckp_id` je uložený identifikátor chovatela (z tabuľky `wp_users`), v `chovatel` meno chovateľa a v `chov_stanice` chovateľská stanica. Informácie o majiteľovi majú podobnú štruktúru, s tým rozdielom že sa dáta ukladajú do stĺpcov `majitel_ckp_id`, `majitel` a `majet_stanice`. V prípade, že chovateľ resp. majiteľ nepochádza z organizácie, jeho identifikátor zostáva prázdny.

Vonkajšie črty zvieraťa, ako farba očí, typ uší, typ srsti, znaky a farba srsti sa ukladajú do stĺpcov `barva_oci`, `typ_ucha`, `typ_srsti`, `bila_kresba` a `barva_srsti`. Pre ukladanie otca a matky zvieraťa slúžia stĺpce `matka` a `otec`, v ktorých sa nachádza jedinečný identifikátor nachádzajúci sa v tejto tabuľke. Pre účely zaznamenania, z akého vrhu dané zviera pochádza, slúži stĺpec `cis_vrhu`, ktoré obsahuje registračné číslo vrhu, z ktorého dané zviera pochádza. Avšak pre jednoznačnú identifkáciu vrhu, odkiaľ zviera pochádza, mal slúžiť stĺpec `id_ckp_vrhu` — ten sa ale nepoužíva.

Pre potreby organizácie sa do tejto tabuľky ukladajú údaje o registrácii zvieraťa — a to typ registrácie (stĺpec `reg_ckp_typ`), číslo registrácie (`reg_ckp_cislo`) a rok registrácie (stĺpec `reg_ckp_rok`). Toto platí iba v prípade, že zviera je zaregistrované pod klubom ČKP. Ak je zviera zaregistrované pod iným klubom, tak sa jeho registračné číslo ukladá do stĺpca `reg_c_ostatni`. Navyše sa ukladá aj identifikátor registrátora, ktorý dané zviera zaregistroval pod klubom ČKP do stĺpca `registrator`. V neposlednom rade je nutné ukladať aj dátum registrácie zvieraťa — pre túto informáciu slúži stĺpec `datum_registrace`.

Informácie o dátume úmrtia a dôvodu úmrtia zvieraťa sa ukladajú do stĺpcov `datum_umrti`, resp. do stĺpca `duvod_umrti`.

Chovateľ zvieraťa si taktiež môže pridať poznámku k zvieraťu, ktorá je následne uložená v stĺpci `poznamka_chovatel`.

Medzi ďalšie informácie, ktoré sa zbierajú o zvierati, patrí aj informácia ohľadom rizikovosti chovu (stĺpec `rizikovost_chovu`) a prípadná poznámka, prečo je chov rizikový. Táto poznámka sa ukladá samostatne do stĺpca `riziko_pozn`.

Predposledný stĺpec `overeno_pk` obsahuje informáciu, či zviera bolo overené podľa plemennej knihy a stĺpec `pozn_edit` obsahuje poznámku, ktorá mohla byť vytvorená pri editácii zvieraťa.

Tabuľka czkp_mimi

Úlohou tejto tabuľky je ukladať informácie o mláďatách narodených v jednotlivých vrhoch. Rozdiel medzi tabuľkou `czkp_mimi` a `czkp_zvire` spočíva v tom, že narozdiel od tabuľky `czkp_zvire`, v tejto tabuľke sú uložené iba mláďata, ktoré pochádzajú zo známeho vrhu.

Tabuľka obsahuje stĺpec `id_vrhu`, v ktorom je uložený identifikátor záznamu z tabuľky `czkp_vrh`. Podľa tohto stĺpcu vieme určiť, v akom vrhu sa dané mláďa narodilo. Pre ukladanie mena a pohlavia mláďata slúžia stĺpce `jmeno`, resp. `pohlavi`. Chovnosť mláďata sa ukladá do stĺpca `chov`. Stĺpec `chov_omez` ďalej slovne špecifikuje chovné obmedzenie mláďata. Pre organizáciu je potrebné sledovať, kto dané mláďa chová, a pre tento účel slúži stĺpec `chov_kdo`. Pre uloženie dodatočných informácií ohľadom mláďata, ako napríklad typ uší, typ srsti, farba srsti, farba očí a znaky slúžia stĺpce `typ_ucha`, `typ_srsti`, `barva_srsti`, `barva_oci` a `znaky`.

Tabuľka pp_informace

Do nasledujúcej tabuľky — `pp_informace` — sa ukladajú informácie spojené s jednotlivými vrhmi. Táto tabuľka nemá žiadnu spojitosť s tabuľkou `czkp_vrh`. Dôvod vzniku tejto tabuľky bol vývoj nadstavby nad pôvodnou aplikáciou.

Táto tabuľka obsahuje stĺpec `uzivatel`, ktorý obsahuje identifikátor majiteľa vrhum ktorý ukazuje do tabuľky `wp_users`. Následne sa do tabuľky ukladajú údaje o samotnom vrhu, ako napríklad typ vrhu (`typ_vrhu`), jeho označenie (`vrh_oznaceni`), dátum narodenia (`vrh_narozeni`), varietnosť (`vrh_varietnost`), počet narodených a odchovaných mláďať (`vrh_nar_mladat` a `vrh_odch_mladat`) a počet odchovaných samcov a samíc (`vrh_odch_kluci` a `vrh_odch_holky`).

Každý vrh má matku a otca vrhu — tieto údaje sú ukladané do stĺpcov `vrh_id_matka` a `vrh_id_otec`. Ich obsahom sú identifikátory rodičov daného vrhu. Títo rodičia pochádzajú z tabuľky `czkp_zvire`.

Informácie o chovatelovi vrhu sa ukladajú do dvoch stĺpcov, a to `chovatel` a `chov_kontakt`. Prvý z menovaných stĺpcov obsahuje meno chovateľa a druhý jeho kontakt.

Nakoľko každý vrh môže byť zaregistrovaný, je nutné ukladať údaje o ich

registrácii. Pre tieto účely slúžia stĺpce `reg_dat_vyplneni`, ktorý značí dátum požiadania o registráciu majiteľom vrhu, `reg_dat_schvaleni`, ktorý obsahuje dátum schválenia vrhu a `reg_cislo_vrhu` obsahujúci registračné číslo vrhu.

Tabuľka pp_miminka

Obsahom tabuľky `pp_miminka` sú mláďatá, ktoré boli narodené vo vrhoch uložených v predchádzajúcej tabuľke.

Aby bolo možné zistiť, ku ktorému vrhu jednotlivé mláďa patrí, bolo potrebné, aby tabuľka obsahovala stĺpec `id_pp`, ktorý obsahuje identifikátor vrhu, v ktorom sa mláďa narodilo. Tento identifikátor ukazuje do predchádzajúpcej tabuľky, `pp_informace`. Následne sa do tabuľky ukladajú dáta týkajúce sa predovšetkým samotného mláďata, ako napríklad jeho meno (`mimi_jmeno`), pohlavie (`mimi_pohlavi`), typ uší (`mimi_typ_ucha`), typ srsti (`mimi_typ_srsti`), farba srsti (`mimi_barva_srsti`), či farba očí (`mimi_barva_oci`) a jeho znaky (`mimi_znaky`). K danému zvieraťu taktiež prislúcha majiteľ, ktorý sa ukladá do stĺpca `mimi_majitel`. Okrem mena majiteľa sa ukladá nie len jeho kontakt (`mimi_maj_kontakt`), ale aj číslo preukazu majiteľa, v prípade že je zároveň členom organizácie (`mimi_maj_prukaz`). Medzi posledné informácie ukladajúce sa k danému mláďatu patrí taktiež informácia, či je mláďa určené pre chov (`mimi_chov`), prípadne chovné obmedzenie (`mimi_chov_omezeni`).

Tabuľka pp_zadosti

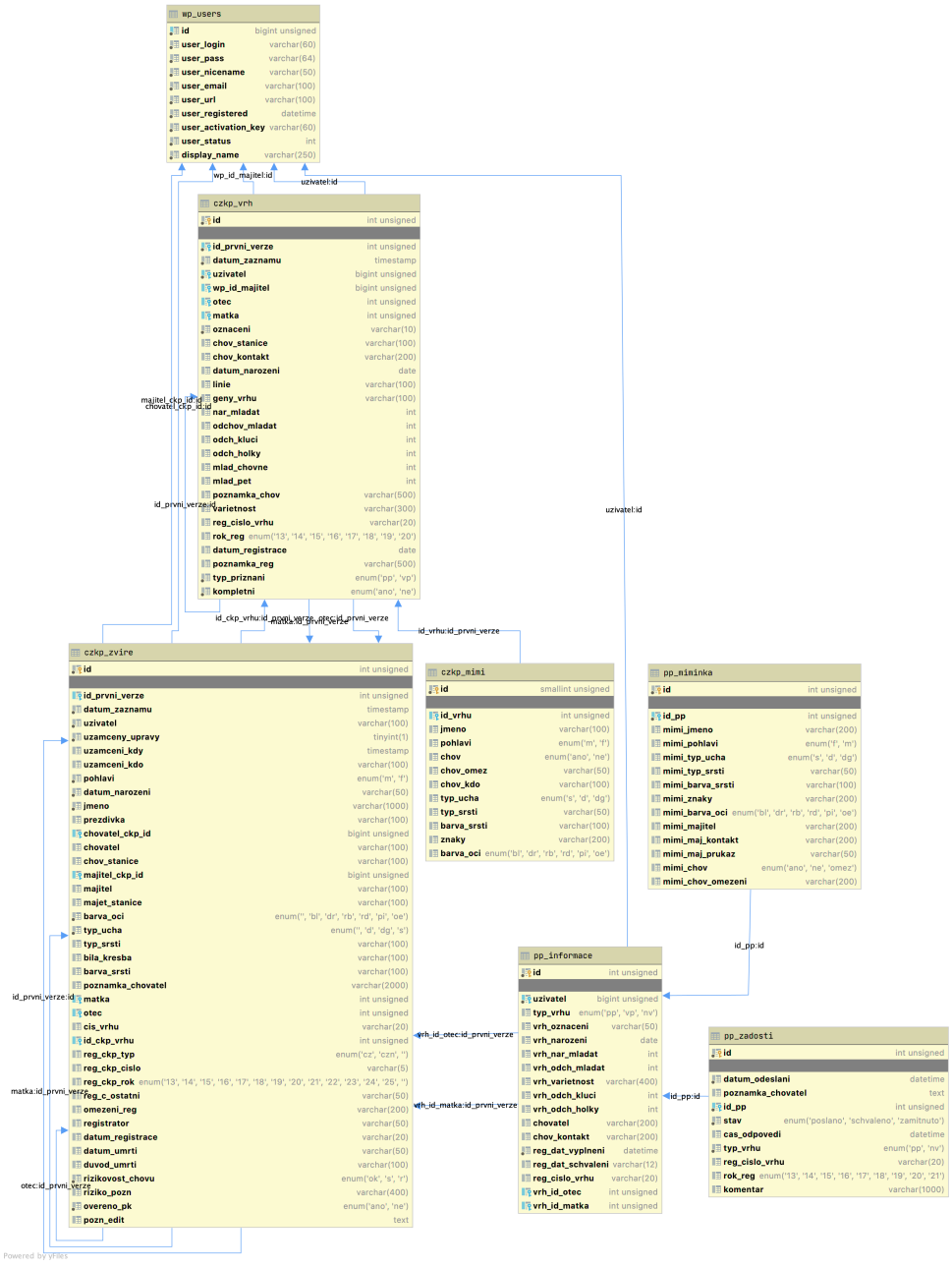
Posledná tabuľka, ktorá sa nachádza v databáze súčasnej aplikácie, je tabuľka s názvom `pp_zadosti`. Táto tabuľka má za úlohu zhromažďovať dáta o poslaných, resp. (ne)schválených žiadostiach o schválenie vrhu. Narozdiel od predchádzajúcich tabuliek, v tejto tabuľke sa nachádzajú stĺpce obsahujúce informácie iba k daným žiadostiam.

Táto tabuľka obsahuje stĺpec `id_pp`, ktorého obsahom je identifikátor vrhu z tabuľky `pp_informace`, ktorého sa týka daná žiadosť o schválenie vrhu. Tabuľka následne obsahuje stĺpec `datum_odoslani`, ktorý je automaticky nastavený na dátum a čas odoslania žiadosti a `cas_odpovedi`, ktorý indikuje dátum a čas odpovedi registrátora na danú žiadosť.

Okrem týchto informácií sa v tabuľke nachádza informácia o type schvaľovaného vrhu (`typ_vrhu`), poznámka žiadateľa o schválenie vrhu (`poznamka_chovatel`) a komentár registrátora ku žiadosti (`komentar`). V prípade, že je žiadosť schválená, `reg_cislo_vrhu` bude obsahovať registračné číslo vrhu a `rok_reg` rok registrácie daného vrhu.

2.1.2.2 Schéma tabuliek

V tejto podsekcii je možné nájsť logický model databázy v grafickej podobne, ktorý obsahuje schému tabuliek so vzťahmi medzi jednotlivými tabuľkami.



Obr. 2.2: Logický model databázy

2.2 Analýza požiadaviek

V tejto sekcii sa nachádza popis všetkých požiadaviek kladených na vznikajúcu webovú aplikáciu. Tieto požiadavky delíme na funkčné a nefunkčné.

2.2.1 Funkčné požiadavky

Funkčné požiadavky vo všeobecnosti vymedzujú hranice aplikácie v kontexte jej funkcionality, ktorú používateľ od aplikácie očakáva. Ich úlohou je taktiež spresniť odhad pracnosti na vznikajúcej aplikácii [**?**].

2.2.1.1 Evidencia zvierat

Aplikácia musí umožniť jednoduchú správu zvierat, od vytvorenia nového zvieraťa, cez zobrazenie jeho detailov, až po následnú editáciu, prípadné jeho zmazanie.

V evidencii zvierat je pre každé zviera potrebné ukladať nasledujúce údaje:

- meno a prezývku
- dátum narodenia
- majiteľa a chovateľa zvieraťa
- vrh, v ktorom sa zviera narodilo
- matku a otca zvieraťa
- pohlavie
- farbu očí
- typ uší
- farbu a typ srsti
- znaky
- dátum a dôvod úmrtia

Pre uľahčenie vytvorenia a editovania zvieraťa budú textové polia v maximálnej možnej miere interaktívne.

To znamená, že polia majiteľa a chovateľa zvieraťa vrátia na základe vstupu zoznam majitelov, resp. chovatelov, ktorí už sú evidovaní v aplikácii. Následne

si z tohto zoznamu používateľ zvolí požadovaného človeka. V prípade, že aplikácia požadovaného človeka nevráti, používateľovi bude ponúknutá možnosť ho dodatočne vytvoriť.

Na podobnom princípe bude fungovať aj textové pole pre vrh, s tým rozdielom, že podľa zvoleného vrhu aplikácia predvyplní matku a otca zvieraťa.

2.2.1.2 Evidencia registrácií zvieraťa

Je potrebné, aby aplikácia ponúkala možnosť zaregistrovať evidované zviera pod záujmový chov. Pri takejto registrácií sa budú zbierať nasledovné údaje:

- klub[1], pod ktorým bude zviera zaregistrované
- typ registrácie[2]
- registračné číslo
- dátum registrácie
- informácia, či je nasledovný chov povolený
- informácia o obmedzení chovu

2.2.1.3 Evidencia vrhov

Medzi nevyhnutnú funkcionalitu aplikácie sa radí aj evidencia vrhov. Tak ako pri evidencii zvierat, tak aj v tomto prípade používatelské rozhranie umožní vytvoriť nový vrh, zobraziť ho, respektíve ho editovať alebo vymazať.

Pre účel evidencie vrhov sa budú v aplikácii ukladať nasledujúce údaje:

- typ a označenie vrhu
- majiteľ vrhu
- meno a kontakt na chovatela
- dátum narodenia
- matka a otec vrhu
- línia
- genetické informácie

[1] Na výber z možností: ČKP, SOCHP alebo Ostatné
[2] Vyžadované iba v prípade registrácie zvieraťa pod klubom ČKP

- počet narodených a odchovaných mláďat
- počet odchovaných samčekov a samičiek
- počet mláďat určených pre maznanie a následný chov

Rovnako ako pri evidencii zvierat, tak aj tu umožní aplikácia interaktívne zvoliť matku, otca a majiteľa vrhu zobrazením zoznamu uložených zvierat/ľudí.

2.2.1.4 Správa žiadostí o schválenie vrhov

Pre organizáciu je žiaduce, aby aplikácia obsahovala správu žiadostí o schválenie vrhu.

V rámci žiadosti o schválenie vrhu rozlišujeme dva typy osôb – žiadateľa o schválenie vrhu[3], a registrátora vrhu.

Aplikácia žiadatelovi umožní poslať žiadosť o schválenie vrhu s možnosťou zanechania poznámky pre registrátora. Po odoslaní a úspešnom spracovaní tejto žiadosti serverom sa odošle e-mail všetkým registrátorom s informáciou o vytvorení novej žiadosti o schválenie vrhu. Na tento e-mail bude môcť zareagovať akýkoľvek registrátor, ktorý sa na základe dostupných informácií o vrhu rozhodne, či danú žiadosť schváli alebo zamietne. O zmene stavu žiadosti bude žiadateľ informovaný e-mailom.

2.2.1.5 Zobrazenie rodokmeňu zvieraťa a vrhu

Pre jednoduchšiu vizualizáciu predkov konkrétneho zvieraťa aplikácia ponúkne zobrazenie rodokmeňu zvieraťa vo forme jednoduchej tabuľky.

V tejto tabuľke budú okrem mien zvierat zobrazené aj dodatočné informácie o zvierati, definované v sekcii **??**. Rodokmeň bude zobrazený v rámci jednotlivých zvierat a vrhov.

V prípade vrhu bude rodokmeň zobrazovať predkov matky a otca daného vrhu.

2.2.1.6 Tvorba poznámok pre zviera a vrh

V rámci aplikácie bude potrebné implementovať poznámky, ktoré budú môcť byť priradené jednotlivým zvieratám a vrhom. Takáto poznámka by mala mať nastaviteľnú viditelnosť[4] a typ[5]. Taktiež musí poskytnúť informáciu, kedy bola vytvorená, respektíve editovaná.

[3] Zväčša majiteľ daného vrhu

[4] Poznámka môže byť buď verejná alebo súkromná

[5] Typ poznámky je jeden z nasledujúcich: všeobecná, upozornenie alebo výstraha

Nakoľko sa jedná o poznámky pre zviera a vrh, budú zobrazené u príslušných zvieratách, resp. vrhoch, ku ktorým sa vzťahujú.

2.2.1.7 Zobrazenie histórie zmien zvierat a vrhov

Medzi želanú funkcionalitu novej webovej aplikácie patrí sledovanie a následné zobrazenie histórie zmien pre všetky zvieratá a vrhy

Aplikácia bude zaznamenávať nasledujúce zmeny v systéme:

- vytvorenie zvieraťa/vrhu
- úprava údajov zvieraťa/vrhu
- zmazanie zvieraťa/vrhu
- obnova[6] zmazaného zvieraťa/vrhu

Pri každej zmene popísanej vyššie je taktiež nutné ukladať, kto a kedy danú zmenu vykonal. V prípade úpravy údajov budú navyše zaznamenané tie údaje, ktoré boli zmenené používatelom. Túto históriu zmien bude možné vidieť vo forme tabuľky u každého zvieraťa, resp. vrhu.

2.2.1.8 Generovanie preukazov

Pre potreby organizácie je nevyhnutné implementovať generovanie preukazov (osvedčení) vo forme PDF súboru. V tomto súbore bude prvá strana vyplnená informáciami o danom zvierati **??**, vrátane jeho registrácie zo sekcie **??**. V niektorých prípadoch bude taktiež zobrazená registrácia vrhu v ktorom sa zviera narodilo, ktorej obsah je definovaný v **??**. Druhá strana súboru bude vyplnená rodokmeňom zvieraťa (**??**).

Preukaz bude možné vygenerovať tlačidlom na stránke konkrétneho zvieraťa.

2.2.1.9 Filtrovanie a radenie

Pre vylepšenie používateľskej skúsenosti s aplikáciou bude potrebné implementovať filtrovanie a radenie zvierat a ako aj vrhov na príslušných stránkach so zoznamom zvierat, resp. verhov. Implementácia tejto funkcionality umožní jednoduchšiu prácu s aplikáciou a rýchlejšie nájdenie potrebných informácií.

2.2.1.10 Správa používatelov a rolí

Pre administrátorov aplikácie je nutné spravovať jednotlivých používateľov aplikácie, priradovať im príslušné role, alebo im ich naopak odoberať. Na základe tejto skutočnosti je žiaduce vytvoriť pohľad so zoznamom používateľov

[6] Obnoviť zviera/vrh bude môcť iba administrátor aplikácie

a ich rolami, s následnou možnosťou im danú rolu zmeniť, prípadne daných používateľov odobrať.

2.2.1.11 Lokalizácia aplikácie

Nakoľko budú aplikáciu používať nie len českí ale aj zahraniční používatelia, je nutné, aby aplikácia poskytovala obsah lokalizovaný do anglického jazyka. Jazyk bude možné jednoducho zmeniť v menu na paneli webovej aplikácie.

2.2.2 Nefunkčné požiadavky

Na rozdiel od funkčných požiadaviek, nefunkčné požiadavky umožňujú určiť obmedzenia kladené na aplikáciu. V neposlednom rade majú taktiež zásadný dopad na návrh architektúry webovej aplikácie [**?**].

2.2.2.1 Webová aplikácia

Nakoľko je požadovaný systém navrhnutý ako webová aplikácia, bude potrebné, aby bola prístupná z internetu pomocou moderných webových prehliadačov.

2.2.2.2 Používateľské rozhranie

Webová aplikácia bude musieť obsahovať používateľské rozhranie, s ktorým budú môcť používatelia interagovať. Prostredie bude naviac responzívne, čo uľahčí prípadný prístup do systému z mobilného prehliadača.

2.2.2.3 Technológie

Po konzultácii s vedúcim práce boli vymedzené nasledovné technológie, ktoré budú použité na strane servera.

Ako programovací jazyk bude použítý jazyk PHP vo verzii 7.3, hoci momentálne je najnovšia verzia jazyku 7.4 [**?**]. Dôvod výberu nižšej verzie jazyka je daný PHP podporou webhostingu[7], na ktorom bude aplikácia nasadená.

Pre ukladanie dát potrebných pre funkčnosť aplikácie a ich následné spracovávanie bude použitý rovnaký databázový systém, ako v súčasnej aplikácii – MySQL. Tento databázový systém budeme používať vo verzii 5.6.

Táto verzia MySQL taktiež nie je najnovšou verziou (v skutočnosti bola prvýkrát vydaná v roku 2013, avšak je stále oficiálne podporovaná [**?**]), opäť

[7]Server poskytujúci webovú aplikáciu používateľom na internete, v tomto prípade sa jedná o webhosting Endora – www.endora.cz

z dôvodu neexistujúcej podpory novšieho databázového systému webhostingom.

2.3 Používateľské role

Novovznikajúca webová aplikácia bude prístupná iba zaregistrovaným a prihláseným používateľom. Navyše, niektoré akcie budú obmedzené iba pre určitý okruh používateľov.

K tomu, aby sme umožnili prístup k vybraným akciám iba vybraným používateľom, bude potrebné do aplikácie implementovať používateľské role a práva. Následne bude aplikácia riadiť prístup používateľa k jednotlivým akciám na základe príslušnosti k vybranej roli.

V nasledujúcich podsekciách budú priblížené jednotlivé role a k nim príslušné práva, ktoré sa budú vyskytovať v aplikácii.

2.3.1 Bežný používateľ

Túto rolu bude mať každý používateľ automaticky po registrácii do webovej aplikácie. Bežný používateľ bude môcť v aplikácii:

- Vytvoriť a zobraziť svoje zvieratá
- Upraviť svoje zvieratá v prípade, že nie sú zaregistrované pod klubom ČKP
- Vytvoriť a upraviť registrácie svojich vlastných zvierat, ktoré nespadajú pod klub ČKP
- Zobraziť zoznam všetkých cudzích zvierat spolu s ich detailmi
- Vytvoriť a zobraziť vrhy, u ktorých je používateľ ich majiteľom
- Upraviť vrhy, ktorých je majitelom, pokiaľ neboli tieto vrhy schválené žiadosťou
- Zobraziť všetky vrhy typu VP alebo schválené vrhy typu PP a NV spolu s ich detailmi
- Pridať poznámku k zvieratám a vrhom, ktoré vlastní

2.3.2 Registrátor zvierat

Registrátor zvierat je v hierarchii rolí postavený nad bežným používateľom. Tým pádom má všetky práva bežného používateľa a navyše nasledujúce práva:

- Možnosť pridať poznámku k akýmkoľvek zvieratám

- Možnosť pridať, editovať a vymazať akúkoľvek registráciu u každého zvieraťa

2.3.3 Schvalovateľ vrhov

Schvalovateľ vrhov je taktiež postavený v hierarchii rolí nad bežným používateľom podobne ako registrátor zvierat, s tým rozdielom, že schvalovateľ vrhu môže v aplikácii:

- Vidieť a odpovedať na žiadosti o schválenie vrhov
- Editovať akékoľvek zviera a vrh
- Pridať poznámky k akémukoľvek vrhu
- Vygenerovať preukaz zvieraťa

2.3.4 Administrátor

Ako obvykle, pre administrátora neplatia žiadne reštrikcie, čo znamená, že bude mať prístup ku všetkej funkcionalite definovanej vo funkčných požiadavkách v sekcii **??**.

2.4 Prípady použitia

Prípady použitia sú špecifikácie rôznych činností, ktoré môžu používatelia s aplikáciou vykonávať [**?**]. Tieto prípady použitia budú zachytené vo forme scenáru a budú vychádzať nielen zo známych funkčných požiadaviek popísaných v sekcii **??**, ale aj z jednotlivých používateľských rolí, ktoré boli definované v sekcii **??**.

V tejto práci som sa rozhodol venovať iba takým prípadom použitia, ktoré sú z môjho pohľadu pre čitateľa prínosnejšie. Tie som následne rozdelil podľa príslušnosti k jednotlivým prvkom aplikácie. Všetky nasledujúce prípady použitia predpokladajú, že používateľ bude v systéme prihlásený.

Poznámka: Prvé štyri scenáre použitia sú aplikovateľné aj na vrhy, avšak z dôvodu neuvádzania duplicitných informácií nebudú ďalej rozvinuté v príslušnej podsekcii. Zároveň pri následnej aplikácii prvých štyroch scenárov na vrhy, sa výskyt slova „zviera“ automaticky nahrádza slovom „vrh“ v príslušnej podobe.

2.4.1 Prípady použitia zvierat

V tejto podsekcii budú popísané jednotlivé prípady použitia, ktoré sa týkajú zvierat.

UC1 – Vyhľadanie zvieraťa

Vyhľadanie zvieraťa je jeden zo základných prípadov použitia, kedy používateľ chce vyhľadať dané zviera.

Scenár:

1. Používateľ zvolí možnosť „Zvieratá“ z navigačného menu aplikácie.
2. Aplikácia následne zobrazí tabuľku so zvieratami a filtrom, na základe ktorého si používateľ môže nastaviť dodatočné kritéria filtrovania zvierat.
3. Používateľ nastaví filter podľa kritérií hľadaného zvieraťa a klikne na tlačidlo „Filtrovať“.
4. Aplikácia zobrazí všetky zvieratá vyhovujúce zadaným kritériám.

Alternatívny scenár:

3. Aplikácia zobrazí hľadané zviera už po načítaní tabuľky so zvieratami. V tomto prípade scenár vyhľadávania zvieraťa končí.

Tento prípad použitia realizuje viaceré funkčné požiadavky na aplikáciu, konkrétne evidenciu a filtrovanie a radenie zvierat.

UC2 – Zobrazenie detailu zvieraťa

Tento prípad použitia popisuje zobrazenie detailu zvieraťa a zahŕňa prípad **??**.

Scenár:

1. **??**
2. Používateľ klikne na meno zvieraťa, ktorého detail si želá vidieť.
3. Aplikácia zobrazí detail zvieraťa.

Popísaný prípad taktiež realizuje funkčnú požiadavku na aplikáciu, a to evidenciu zvierat.

UC3 – Vytvorenie zvieraťa

Medzi ďalší základný prípad použitia patrí pridanie zvieraťa používatelom do systému. Pri vytváraní zvieraťa sa aplikuje dodatočné obmedzenie pre bežného používateľa, ktoré spočíva v nemožnosti zvolenia majiteľa iného ako je sám používateľ (**??**).

Scenár:

1. Používateľ zvolí možnosť „Zvieratá“ z navigačného menu aplikácie, ktorá následne načíta stránku so zoznamom uložených zvierat.
2. Následne používateľ klikne na tlačidlo „Pridať zviera“.
3. Aplikácia načíta novú stránku s formulárom pre vytvorenie nového zvieraťa.
4. Používateľ vyplní formulár.
5. Po vyplnení všetkých potrebných údajov pre vytvorenie zvieraťa aplikácia umožní odoslať formulár kliknutím na tlačidlo „Uložiť“.
6. Používateľ klikne na tlačidlo „Uložiť“.
7. Po úspešnom vytvorení zvieraťa na základe vložených údajov systém presmeruje používateľa na stránku so zoznamom uložených zvierat.

Tento prípad použitia realizuje funkčnú požiadavku evidencie zvierat.

UC4 – Úprava zvieraťa

V tomto prípade sa používateľ aplikácie snaží upraviť existujúce zviera v systéme. Tento prípad použitia taktiež zahŕňa prípady **??** alebo **??**, v závislosti od spôsobu úpravy zvieraťa, ktorý si používateľ zvolí.

Scenár:

1. **??**
2. Používateľ klikne na rozbalovacie menu u zvieraťa, ktoré chce editovať.
3. Po tomto kroku aplikácia ponúkne možnosť editácie alebo zmazania zvieraťa.
4. Používateľ zvolí možnosť editácie.
5. Aplikácia následne presmeruje používateľa na stránku s editáciou zvieraťa, ktorá bude obsahovať formulár s predvyplnenými údajmi zvieraťa.
6. Používateľ upraví údaje zvieraťa.
7. Kliknutím na tlačidlo „Odoslať“ sa odošle vyplnený formulár na server.
8. Po úspešnom spracovaní formuláru je používateľ presmerovaný na stránku s detailmi upravovaného zvieraťa.

Alternatívny scenár:

1. **??**
2. Používateľ klikne na tlačidlo „Upraviť“ na pravom postrannom paneli.
3. Následne scenár pokračuje bodom 5 v pôvodnom scenári.

Popísaný prípad použitia realizuje funkčnú požiadavku evidencie zvieraťa.

UC5 – Vytvorenie registrácie zvieraťa

Tento prípad použitia popisuje vytvorenie registrácie zvieraťa. Pri samotnom vytváraní novej registrácie sa aplikujú dodatočné obmedzenia pre bežného používateľa, tak ako je popísane v **??**.

Tento prípad použitia zahŕňa prípad zobrazenia detailov zvieraťa.

Scenár:

1. **??**
2. Kliknutím na tlačidlo „Pridať novú registráciu“ aplikácia zobrazí registračný formulár v modálnom okne[8].
3. Používateľ vyplní potrebné polia.
4. Po stlačení tlačidla „Uložiť“ sa vyplnený formulár odošle na server.
5. Po úspešnom spracovaní formuláru serverom sa modálne okno zavrie.

Tento prípad použitia realizuje funkčnú požiadavku kladenú na aplikáciu, konkrétne evidenciu registrácií zvieraťa.

2.4.2 Prípady použitia vrhov

Táto podsekcia popisuje jednotlivé prípady použitie, ktoré sa týkajú vrhov.

UC6 – Vytvorenie žiadosti o schválenie vrhu

Tento prípad použitia reálne zahŕňa prípad použitia „zobrazenia detailov vrhu“, avšak pre neuvádzanie duplicitných informácii môže čitateľ vychádzať z prípadu použitia zobrazenia detailov zvieraťa. Scenáre sa v oboch prípadoch obsahovo nelíšia, tak ako bolo uvedené poznámke v sekcii **??**.

Na tento prípad sa taktiež vzťahujú dodatočné obmedzenia pre bežného používateľa vyplývajúce z **??**.

[8] Menšie okno aplikácie prekrývajúce jej hlavný obsah

Scenár:

1. **??**
2. Používateľ klikne na tlačidlo „Vytvoriť žiadosť“.
3. Aplikácia následne zobrazí modálne okno s formulárom obsahujúcim textové pole slúžiace pre poznámku registrátorovi.
4. Po stlačení tlačidla „Uložiť“ sa formulár odošle na server.
5. Po úspešnom spracovaní formuláru serverom sa vytvorí nová žiadosť o schválenie vrhu.
6. Modálne okno sa automaticky zavrie.

Popísaný prípad použitia realizuje funkčnú požiadavku správy žiadostí o schválenie vrhu.

UC7 – Odpoveď na žiadosť o schválenie vrhu

Tak ako v predchádzajúcom prípade, tento prípad použitia bude zahŕňať „zobrazenie detailov vrhu“.

Pre tento prípad použitia sa vzťahujú dodatočné obmedzenia – na žiadosť o schválenie vrhu bude môcť odpovedať používateľ majúci rolu schvalovateľa žiadostí vrhov alebo administrátora.

Scenár:

1. **??**
2. Používateľ stlačí tlačidlo „Odpovedať“ pri príslušnej žiadosti o schválenie vrhu.
3. Aplikácia zobrazí modálne okno s formulárom pre schválenie vrhu.
4. Používateľ vyplní formulár a kliknutím na tlačidlo „Uložiť“ sa odošle na server.
5. Server spracuje odoslaný formulár a ihneď odošle e-mail žiadatelovi o stave jeho žiadosti.

Alternatívny scenár:

1. Používateľ klikne na link v e-maili, ktorý mu prišiel po odoslaní novej žiadosti o schválenie vrhu žiadateľom.
2. Ako reakcia na kliknutie na link, operačný systém otvorí predvolený webový prehliadač s adresou vedúcou na detail vrhu.

3. Následne scenár pokračuje bodom 2 v pôvodnom scenári.

Tak ako v predchádzajúcom prípade použitia, aj v tomto prípade použitia je realizujovaná funkčná požiadavka správy žiadostí o schválenie vrhu.

Kapitola 3

Návrh

Táto kapitola sa venuje návrhu architektúry budúcej aplikácie, s priblížením dvoch hlavných architektúr používaných pri tvorbe webových aplikácií. Následne sú porovnané a podľa výsledkov porovnania je vybraná taká, ktorá bude použitá pri vývoji aplikácie. V nasledujúcej sekcii sú vymedzené použité technológie budúcou aplikáciou. V neposlednom rade táto kapitola obsahuje návrh modelových tried a končí návrhom REST API, ktoré bude aplikácia používať pri komunikácii so serverom.

3.1 Návrh architektúry

Dôležitý aspekt pri návrhu aplikácie spočíva vo výbere architektúry, na ktorej bude aplikácia postavená. Na základe jej výberu sa odvíjajú technológie, ktorými bude daná aplikácia disponovať.

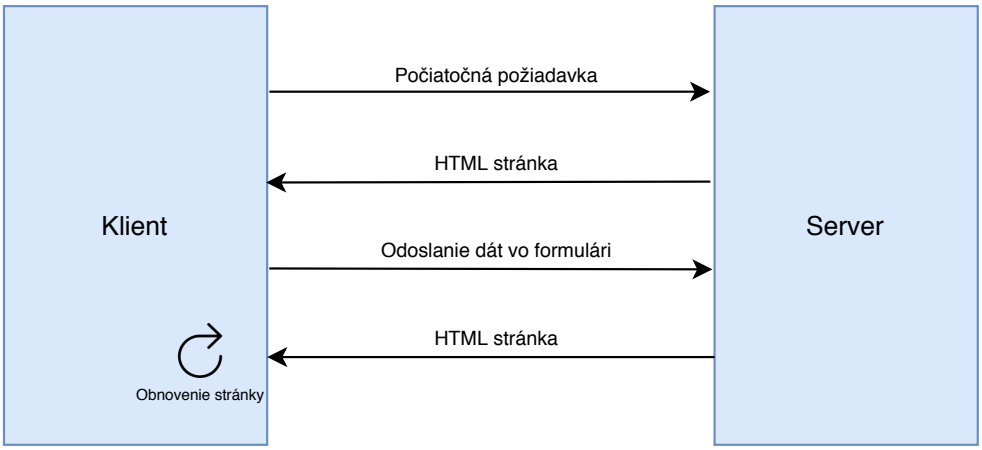
V nasledujúcich sekciách sú zdokumentované dva typy architektúr, z ktorých bude na základe porovnania a vlastných skúseností s tvorbou webových aplikácií vybraná jedna, ktorá bude definovať podobu budúcej webovej aplikácie.

3.1.1 Architektúra MPA aplikácie

MPA je skratka pre viac-stránkovú webovú aplikáciu. Takáto aplikácia vždy načíta celú stránku a zobrazí novú v prípade interakcie s aplikáciou – zvyčajne po odoslaní formuláru používateľom [**?**].

Priebeh komunikácie medzi serverom a klientom

Celkový priebeh komunikácie medzi klientom a serverom zachytáva nasledujúci diagram s jeho popisom.



Obr. 3.1: Diagram znázorňujúci architektúru MPA aplikácií

1. Klient požiada server o stránku
2. Server vráti klientovi požadovanú stránku
3. Klient vrátenú stránku serverom vykreslí
4. Klient vyplní formulár, ktorý sa následne odošle na server
5. Server prijme údaje od klienta, ktoré následne spracuje
6. Po spracovaní údajov klientovi vráti späť stránku s aktualizovanými údajmi
7. Klient túto stránku načíta, čím príde k obnoveniu stránky

Technológia AJAX čiastočne rieši problém znovunačítavania celej stránky dynamickým aktualizovaním tých častí aplikácie, ktoré boli používateľom zmenené. Avšak zakomponovanie tejto technológie do MPA sťažuje a komplikuje celý vývojový proces aplikácie [**?**].

Výhody použitia MPA architektúry

- Jednoduchá optimalizácia stránok pre webové vyhľadávače [**?**]
- Umožňuje jednoduchú správu informácií na jednotlivých stránkach [**?**]
- Je potrebný menší rozsah nástrojov a znalostí narozdiel od vývoja SPA aplikácie [**?**]
- Veľa dostupných riešení pre vývoj MPA aplikácií [**?**]

Nevýhody použitia MPA architektúry

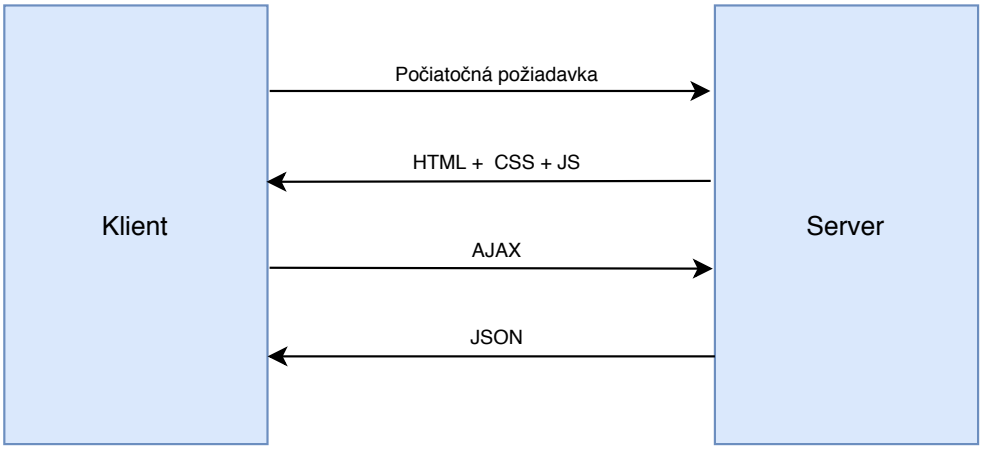
- Zvýšený čas načítavania stránok kvôli neustálemu obnovovaniu stránok (neplatí pri použití technológie AJAX) [**?**]
- Znížený výkon aplikácie, kvôli neustálemu načítavaniu väčšieho množstva informácií naraz pred následným poslaním stránky používatelovi [**?**]
- Úzka previazanosť vývoja aplikácie na strane servera a klienta, čo znemožňuje prípadné neskoršie nasadenie rôznych technológií [**?**]

3.1.2 Architektúra SPA aplikácie

SPA je skratka pre single-page aplikáciu. Single-page aplikácia je aplikácia, ktorá nepotrebuje znovunačítanie stránky počas jej používania. Na tomto type architektúry sú postavené aplikácie ako Gmail[9], Google Mapy[10], Facebook[11] či GitHub[12]. V tomto prípade sa o aktualizáciu obsahu na stránke nestará server, ale klient, zväčša pomocou frameworku[13] bežiacom v prostredí webového prehliadača [**?**].

Priebeh komunikácie medzi serverom a klientom

Celkový priebeh komunikácie medzi klientom a serverom zachytáva nasledujúci diagram s jeho popisom.



Obr. 3.2: Diagram znázorňujúci architektúru SPA aplikácií

[9] https://gmail.com
[10] https://maps.google.com
[11] https://facebook.com
[12] https://github.com
[13] Sada podporných programov uľahčujúca vývoj aplikácie

1. Klient požiada server o stránku
2. Server vráti klientovi požadovanú HTML stránku so všetkým obsahom aplikácie
3. Klient vrátenú stránku serverom vykreslí
4. Klient vyplní formulár, ktorý sa následne odošle na server pomocou AJAXu
5. Server prijme údaje od klienta, ktoré následne spracuje
6. Po spracovaní údajov server vráti späť klientovi aktualizované údaje (v JSON[14] formáte)
7. Klient zahodí staré údaje na stránke a nahradí ich novými, čím príde k obnoveniu údajov ale nie stránky

Výhody použitia SPA architektúry

- Rýchlosť a responzívnosť aplikácie založenej na aktualizovaní iba tej časti aplikácie, ktorá sa zmenila [**?**]
- Minimálna previazanosť kódu aplikácie na strane servera a klienta, čo umožňuje jednoduchú správu používaných knižníc bez ovplyvnenia ďalších častí aplikácie [**?**]
- Zjednodušení proces vývoja, nakoľko sa o vykreslenie obsahu nestará server ale klientský framework [**?**]

Nevýhody použitia SPA architektúry

- Ťažká optimalizácia aplikácie pre webové vyhľadávače, nakoľko o vykreslenie obsahu sa stará Javascript[15], ktorý webové vyhľadávače neinterpretujú [**?**]
- Prvé načítanie aplikácie trvá dlhší čas v porovnaní s MPA architektúrou, pretože pri prvom navštívení aplikácie sa musí všetok obsah aplikácie stiahnuť do zariadenia
- Nemožnosť zobrazenia aplikácie pri vypnutom Javascripte vo webovom prehliadači [**?**]
- Pri komplexnejšej aplikácii bude aplikácia zaberať viac pamäte zariadenia, z ktorého je zobrazovaná, nakoľko všetok obsah aplikácie je stiahnutý v zariadení [**?**]

[14] Formát na výmenu dát medzi klientom a serverom [**?**]
[15] Programovací jazyk bežiaci vo webovom prehliadači

3.1.3 Voľba architektúry

Pre budúcu aplikáciu som sa rozhodol vybrať SPA architektúru, nie len z dôvodu rýchlejšieho a jednoduchšieho procesu vývoja, ale aj vďaka vlastným skúsenostiam z oblasti vývoja SPA aplikácií. Nakoľko aplikácia bude dostupná iba vopred vybraním používateľom, nebude potrebné ju optimalizovať pre webové vyhľadávače.

3.2 Voľba technológií

V tejto sekcii sa presunieme od voľby architektúry, na ktorej bude aplikácia postavená, ku voľbe technológií, ktoré budu použité vo výslednej aplikácii.

3.2.1 PHP

Medzi zvolené technológie bude jednoznačne patriť programovací jazyk PHP, nakoľko jeho voľba je jedným z nefunkčných požiadaviek kladených na budúcu aplikáciu.

Framework

Vývoj softvéru je komplexný proces, ktorý zahŕňa nie len výslednú implementáciu konkrétneho softvéru, ale aj analýzu, návrh, a v neposlednom rade aj testovanie softvéru. Softvérové frameworky uľahčujú prácu vývojárom tým, že im umožňujú prevziať kontrolu nad celým procesom vývoja softvéru alebo jeho väčšiny [**?**].

Medzi niektoré výhody použitia frameworku patrí:

- Pomoc pri zavádzaní lepších programovacích postupov a vhodnom využívaní návrhových vzorov
- Väčšia bezpečnosť výsledného kódu
- Možnosť vyhnúť sa duplicite kódu
- Skrátenie času vývoju výslednej aplikácie

Pre vývoj webových aplikácií v PHP existujú viaceré frameworky, ako napríklad CakePHP[16], Nette[17], Symfony[18], Laravel[19] a iné. Spomedzi menovaných bol vybraný práve posledný z nich, na základe skúseností autora.

[16] https://cakephp.org
[17] https://nette.org
[18] https://symfony.com
[19] https://laravel.com

3.2.2 HTML 5

HTML – HyperText Markup Language – je značkovací jazyk používaný pre definovanie významu a štruktúry dokumentov. Ako už z názvu vyplýva, HTML používa značky pre definovanie elementov v dokumente s rozdielnymi vlastnosťami, ako napríklad nadpis, odstavec, tabulky a iné. Tieto značky sú interpretované webovým prehliadačom, ktorý ich následne vykreslí na stránku [**?**].

3.2.3 CSS 3

Cascading Style Sheets (CSS) je jazyk, ktorý sa používa na ilustráciu vzhľadu, štýlu a formátu dokumentu a jeho elementov napísaného v akomkoľvek značkovacom jazyku. Využíva sa najmä v spojitosti s webovými stránkami. CSS je tvorený pravidlami aplikovanými na jednotlivé elementy dokumentu. Tieto pravidlá sú definované selektorom elementu, ktorý určuje polohu elementu v dokumente, a CSS pravidlami rôzneho typu, ktoré definujú konkrétny štýl daného elementu [**?**].

CSS3 je momentálne najnovšia verzia staršej jazyka CSS.

3.3 Návrh modelových tried

3.4 Návrh REST API