Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informatiky a kvantitativních metod

Webová aplikace pro podporu výuky na 1. stupni ZŠ

Diplomová práce

Autor: Veronika Němečková

Studijní obor: Aplikovaná informatika

Vedoucí práce: doc. RNDr. Petra Poulová, Ph.D.

Hradec Králové duben 2016

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracovala samostatně a s použitím uvedené literatury.

V Hradci Králové dne 25. 4. 2016 Veronika Němečková

Poděkování

Děkuji vedoucí své diplomové práce doc. RNDr. Petře Poulové, Ph.D., za metodické vedení práce, čas věnovaný konzultacím a cenné rady a připomínky, které mi poskytla.

Zároveň bych ráda poděkovala Základní škole v Kamenných Žehrovicích, která se rozhodla využít aplikaci vytvořenou v rámci této diplomové práce ve výuce.

Anotace

Cílem diplomové práce je navrhnout a implementovat webovou aplikaci, jež bude sloužit jako nástroj pro podporu výuky na 1. stupni základní školy. Výsledná aplikace bude umožňovat žákům procvičovat jednotlivá témata předmětu Český jazyk, vyučujícím bude poskytovat zpětnou vazbu o tom, která látka dělá žákům problémy.
V teoretické části je vysvětlen pojem elektronické vzdělávání a nastíněn proces tvorby elektronických kurzů. Na závěr jsou představeny technologie, jež byly použity při vývoji výše zmíněné aplikace.
Praktická část je věnována samotnému vývoji aplikace. V této části jsou popsány jednotlivé fáze vývoje aplikace od prvotní analýzy a stanovení požadavků až po vytvoření návrhu a konečnou implementaci.
V závěru práce jsou zhodnoceny dosažené výsledky a nastíněna další uvažovaná rozšíření do budoucna.

Annotation

Title: Web application for support of education at 1st grade of primary school

The aim of this diploma thesis is to design and implement web application which will serve as a tool for education support at 1st grade of primary school. The application will allow students practice topics of the course Czech language and will provide feedback for teachers, which topics are difficult for students.
In the theoretical part is introduced e-learning and process of the creation of e-learning courses. In the last topic of this part there are introduced technologies that were used to develop this application.
The practical part deals with development process of this application. In this part there are described development stages starting with an initial analysis and establishing requirements, and ending with the creation of an architectural concept and the final implementation.
The end of the thesis concludes achieved result and discusses the considered extensions of further development.

Obsah

Úvod

Pojem elektronické vzdělávání označuje výuku, v níž jsou využity moderní informační a komunikační technologie. Díky svým přednostem, jako jsou nezávislost studia na čase a místu, neomezená dostupnost výukových materiálů kdykoli a odkudkoli a přizpůsobitelnost průchodu elektronickým kurzem na míru studujícího, se e-learning stává čím dál tím oblíbenější formou vzdělávání.

V rámci diplomové práce bude navržena a implementována webová aplikace, která bude stavět na základních principech elektronického vzdělávání – jako jsou časová a prostorová flexibilita, neomezená dostupnost výukového obsahu, testování nabytých znalostí a poskytování zpětné vazby studujícím. Cílem aplikace a vytvořeného kurzu není dosažení toho, aby fungovala jako plnohodnotný elektronický kurz, nýbrž se bude jednat o doplněk k prezenční výuce, jenž bude žákům sloužit k procvičování probrané látky. Zadání této webové aplikace na procvičování bylo vytvořeno a postupně upřesňováno na základě odborných konzultací s vyučujícími na 1. stupni základní školy.

Výsledná aplikace bude nasazena na základní škole v Kamenných Žehrovicích, která se rozhodla využít ji jako podporu výuky českého jazyka ve 2. třídě.

Cíl práce

Cílem práce je vytvoření jednoduché a uživatelsky přívětivé webové aplikace pro podporu výuky na 1. stupni základní škole. Tato aplikace bude umožňovat žákům procvičování témat probraných ve škole a učitelům bude poskytovat zpětnou vazbu o tom, která látka dělá v procvičování žákům potíže, aby se k ní v případě potřeby mohli v hodinách vrátit.

Každým dokončeným procvičováním se bude aplikace přizpůsobovat potřebám žáka, aby se mohl zaměřit na ty oblasti, které mu dělají největší problémy. Aplikace bude navíc obsahovat různé motivační prvky, jenž budou žáka povzbuzovat k dalšímu procvičování.

Pro účely této diplomové práce bude pro aplikaci vypracován elektronický kurz Český jazyk pro 2. třídu.

Metodika zpracování

První polovina teoretické části je věnována elektronickému vzdělávání. Nejprve je vysvětlen pojem elektronické vzdělávání neboli e-learning, následně jsou představeny jeho typy a vyzdvihnuty největší klady, ale také zápory, které tato forma výuky přináší. Dále je věnována pozornost problematice tvorby elektronických kurzů, kde je nastíněn samotný proces a zásady jejich tvorby.
V druhé polovině teoretické části jsou představeny technologie, jež byly použity při vývoji webové aplikace.
Praktické část je věnována samotnému vývoji aplikace. Na začátku je představeno zadání projektu, na jehož základě jsou stanoveny funkční a nefunkční požadavky na budoucí aplikaci. Analýzou funkčních požadavků pak byly vytyčeny případy užití, pomocí kterých byl navržen model tříd. V druhé polovině praktické části je popsán proces implementace s ukázkami zdrojového kódu a přiložena uživatelská příručka k aplikaci.
V závěru diplomové práce jsou zhodnoceny dosažené výsledky a představena další uvažovaná rozšíření aplikace do budoucna.

I TEORETICKÁ ČÁST

1 Elektronické vzdělávání (e-learning)

E-learning neboli elektronické vzdělávání lze obecně chápat jako vzdělávací proces, v rámci něhož jsou za účelem dosažení vzdělávacích cílů využity moderní informační a komunikační technologie. V historii byla jako e-learning označována každá výuka, která zahrnovala použití moderních výpočetních prostředků v jakékoli podobě – například i spuštění obyčejného výukového CD nebo zobrazení webové stránky s učebním materiálem bylo již nazýváno elektronickým vzděláváním. V současnosti je e-learning vnímán především jako vzdělávání vedené pod dohledem tutora[1], realizované prostřednictvím počítačové sítě, zejména internetu. [1] [2]

Elektronické vzdělávání je ale široký pojem, jenž není možné vymezit jedinou, obecně platnou definicí. V současné teorii existuje pět základních přístupů k vymezení e-learningu:

- Technologické pojetí

  Toto pojetí přestavuje e-learning jako souhrn technologických prostředků, které jsou využívány v rámci vzdělávacího procesu. Informační a komunikační technologie ve výuce pomocí e-learningu usnadňují přístup ke vzdělávacímu obsahu, výměně informací, slouží jako komunikační prostředek mezi jednotlivými účastníky vzdělávacího procesu a poskytují nástroje pro řízení studia. [1] [2] [3] [4]

- Pedagogické pojetí

  V pedagogickém přístupu je e-learning vnímán jako proces vzdělávání, v němž jsou využívány multimediální technologie, internet a další elektronická média za účelem zvýšení kvality vzdělávání. Audiovizuální technické prostředky v podobě obrazových, zvukových a textových studijních materiálů slouží k obohacení obsahu výuky, internet poskytuje jednoduchý přístup ke zdrojům informací, jejich výměně a možnosti komunikace a spolupráce se vzdělávací komunitou. [1] [2] [3]

[1] osoba, která dohlíží na studující v e-learningovém kurzu, hodnotí je a poskytuje jim zpětnou vazbu [1] [2]

- Síťové pojetí

  Z pohledu síťového přístupu je e-learning vzdělávací proces realizovaný prostřednictvím počítačových sítí. Tato definice vylučuje, aby výuka doplněná multimediálními materiály (CD-ROM, DVD a jiné) byla označována jako elektronické vzdělávání. [1] [2] [3]
- Procesní pojetí

  V procesním pojetí, jak lze již z názvu odvodit, je elektronické vzdělávání chápáno jako souhrn procesů učení (osvojování znalostí a dovedností) a vyučování (předávání obsahu učiva), realizovaný pomocí elektronických prostředků v jakékoli podobě. [1] [3]
- Definice dle Evropské unie

  Podle Evropské unie představuje e-learning začleňování nových multimediálních technologií a internetu do výuky za účelem zvýšení kvality vzdělávání. [1] [3]

V zahraniční literatuře je dokonce možné setkat se s velmi obecnou definicí, podle níž je e-learning jakékoli vzdělávání založené na technologiích, a pro každý specifický případ elektronického vzdělávání existuje platný termín – například CBT (Computer-Based Training), CAL (Computer-Assisted Learning), WBT (Web-Based Training), M-learning a další. Jednotlivé formy e-learningu budou postupně vysvětleny v následujících kapitolách. [1] [5] [6]

1. 1 Formy e-learningu

S vývojem informačních a komunikačních technologií se měnila i samotná podoba e-learningu. Elektronické vzdělávání již nebylo pouze o spuštění výukového CD či přečtení si informace z internetu, nýbrž se postupem času stalo označením pro komplexní propracované výukové kurzy, ve kterých studující znalosti nejen získává, ale má možnost si je i procvičovat, ověřovat a na jejich základě odvozovat další nové vědomosti. [1] [4] [5] [6]

Elektronické vzdělávání se tak změnilo v souhrnný pojem pro označení výuky podporované moderními informačními technologiemi, který lze dále dělit do dalších podoborů. [1] [2] [7]

V následující části bude nejprve představeno rozdělení e-learningu z hlediska nutnosti připojení do sítě, poté budou následovat nejznámější formy elektronického vzdělávání podle použitých technologií. [2] [5] [6] [7]

1. 1. 1 Typy e-learningu z pohledu nutnosti připojení

V případě dělení elektronického vzdělávání podle nezbytnosti připojení k počítačové síti pro studium je e-learning rozdělen na dva základní typy – online a offline výuka, někdy také označováno jako online a offline learning. [2] [5] [6] [7]

1. 1. 1. 1 Offline výuka

U offline výuky, jak je již z názvu možné odvodit, není nutné, aby byl studující připojen k počítačové síti. Studijní materiály jsou k dispozici buď na paměťových nosičích, nebo jsou umístěny přímo v počítači, pomocí kterého je výuka realizována. [2] [5] [6] [7]

Nevýhodou této formy výuky je náročnost zajištění komunikace mezi jednotlivými účastníky vzdělávacího procesu. V případě, že e-learningový kurz neposkytuje žádné nástroje pro podporu komunikace mezi vyučujícími a studujícími, je nutné zvážit, zda se skutečně ještě jedná o elektronické vzdělávání, či o studium založené na multimédiálních studijních materiálech. [7] [8]

1. 1. 1. 2 Online výuka

Online learning se oproti offline formě bez připojení k síti již neobejde, vzdělávací obsah je distribuován prostřednictvím internetu, popř. intranetu. Jestliže výuka probíhá v reálném čase, je označována jako synchronní, v opačném případě se jedná o asynchronní online výuku. [2] [5] [7] [8]

Výhodou synchronní formy výuky je možnost vzájemné interakce všech účastníků vzdělávacího procesu ve stejném okamžiku. Znalosti a dovednosti jsou předávány (přijímány) v reálném čase a studující tak mají větší prostor pro vzájemnou komunikaci a spolupráci. Komunikace mezi studujícími a tutorem je realizována pomocí chatu, instant messagingu[2], sdíleného whiteboardu[3] nebo audio či videokonferencí. Nevýhodou synchronní formy výuky je ale již částečné omezení z hlediska nutnosti koordinace časových plánů studujících. [2] [5] [7] [8]

U asynchronní formy již nejsou účastníci omezováni stanoveným časovým rozvrhem, výuka probíhá jako řízené samostudium pod dohledem tutora. Studující si tak mohou zvolit vlastní tempo a způsob, jak budou postupovat studijním kurzem. Komunikace mezi jednotlivými účastníky probíhá například

2 internetová služba umožňující komunikaci pomocí zpráv v reálném čase [7] [8]
3 sdílená tabule (nástěnka), uživatelé vidí příspěvky ostatních a zároveň mohou přidávat své vlastní [7] [8]

prostřednictvím zpráv, e-mailů, vyvěšených aktualit, stanovením termínů v elektronickém kalendáři, na diskusním fóru apod. Jako nevýhoda asynchronní formy výuky se ale pro některé studující mohou jevit vyšší nároky na samostatnost a nepostradatelnost vlastní motivace ke studiu. [2] [5] [7] [8]

1. 1. 2 Typy e-learningu z pohledu použitých technologií

Tento způsob rozdělení e-learningu nerozlišuje elektronické vzdělávání pouze na základě toho, pomocí kterých informačních a komunikačních technologií je výuka realizována, ale zabývá se i tím, jakou roli tyto technologie ve vzdělávacím procesu zastupují. [1] [2] [6]

1. 1. 2. 1 Computer-Based Training

Computer-Based Training (CBT) neboli počítači podporovaná výuka je forma e-learningu, u které není vyžadováno, aby měl účastník k dispozici připojení do počítačové sítě. Veškerý studijní obsah se nachází na paměťových nosičích, popřípadě je uložen přímo v počítači. [2] [6] [7]
V odborné literatuře je možné se setkat i s příbuznými termíny: Computer-Based Instruction (CBI), Computer-Assisted Learning (CAL) a Computer-Assissted Instruction (CAI). Všechny tyto formy, stejně jako u Computer-Based Trainingu, představují výuku, která je realizována pomocí počítače. Do jaké konkrétní kategorie e-learningový kurz spadá, záleží právě na tom, jakou roli počítač ve výuce hraje. [1] [5] [6]

1. 1. 2. 2 Web-Based Training

Vzdělávání pomocí webových technologií, v anglickém jazyce Web-Based Training (WBT), je jedním ze zástupců online learningu. Studující mají výukové materiály k dispozici na internetu (popř. intranetu) a přistupují k nim pomocí webového prohlížeče. [2] [5] [6] [7]
Některá odborná literatura používá pro označené této formy e-learningu výraz Web-Based Instruction (WBI), který ale představuje to samé – tedy vzdělávání, které je realizováno prostřednictvím webu. [5]

1. 1. 2. 3 Blended learning

Blended learning představuje kombinaci klasické výuky s e-learningem. Může se jednat buď o prezenční vyučování doplněné elektronickým vzděláváním (popř. jen vybranými e-learningovými aktivitami), nebo naopak o elektronické vzdělávání v rámci kterého je část vyučování realizována prezenční formou. [6] [8]
Výuka formou blended learningu přebírá výhody prezenčního a elektronického vzdělávání a naopak vyvažuje nevýhody jednoho či druhého přístupu. Blended learning se tak snaží o vytvoření co nejvhodnějších podmínek pro studium a může tak být ideální volbou pro zpestření a zefektivnění tradičního prezenčního vyučování. [1] [6] [8]

1. 1. 2. 4 M-learning

Mobile Learning, zkráceně m-learning, je forma elektronického vzdělávání, které je realizováno pomocí moderních mobilních zařízení – chytrých telefonů, tabletů, ale i MP3 či MP4 přehrávačů a dalších. [1] [2] [7]
Výhodou m-learningu je mobilita zařízení, pomocí kterého studující přistupují ke vzdělávacímu obsahu – studium se tak pro ně stává přenositelným a mají tak možnost využívat k učení i čas, který by jinak trávili neefektivně (např. dojíždění hromadnou dopravou). Nevýhodou této formy e-learningu jsou ale technická omezení, které m-learning provází – zejména malá velikost diplejů, složité ovládání nebo omezená kapacita pro ukládání dat. [1] [2] [7]

1. 2 Výhody a nevýhody e-learningu

Vzdělávání pomocí e-learningu nabízí oproti klasické (prezenční) výuce nové možnosti a řadu výhod, na druhou stranu ani elektronické vzdělávání nelze považovat za tu jedinou správnou, univerzálně použitelnou, formu výuky. V následujících dvou kapitolách budou představeny největší klady a zápory, které elektronické vzdělávání přináší. [1] [2] [9]

1. 2. 1 Výhody e-learningu

1. 2. 1. 1 Časová a prostorová flexibilita

Zřejmě největším přínosem elektronického vzdělávání je nezávislost studia na čase a místu. Studující se nemusí podřizovat rozvrhu školy, sám si může naplánovat, kdy a po jakou dobu se chce studiu věnovat, a nemusí se přitom nikterak podřizovat ostatním účastníkům vzdělávacího procesu. Vzhledem k tomu, že není ani nutné, aby se všichni studující nacházeli na stejném místě, není již případný zájemce o studium omezován vzdáleností mezi působištěm dané vzdělávací instituce a svým bydlištěm. [1] [2] [5] [9]

E-learningová forma studia se tak může stát ideálním řešením pro osoby, pro které výuka vázaná na sídle vzdělávací instituce není vhodná. Účastníkem se tak může stát i rodič na mateřské dovolené, člověk s vysokým pracovním vytížením nebo i osoba se zdravotním postižením, jejíž zdravotní stav by běžné studium příliš komplikoval, či dokonce plně neumožňoval. [5]

1. 2. 1. 2 Neomezená dostupnost výukových materiálů

Dalším velkým přínosem e-learningu je dostupnost studijního obsahu prakticky kdykoli, odkudkoli. Pokud má studující potřebné technické vybavení pro daný elektronický kurz – u offline learningu je nutný pouze počítač (popř. tablet či chytrý telefon), u online formy navíc přístup k internetu či intranetu, studium se pro něho stává přenositelným a může se mu věnovat kdekoli, jak mu to čas dovolí. [1] [2] [5] [9]

1. 2. 1. 3 Přizpůsobitelnost studia

Z předchozích dvou zmíněných bodů (nezávislost studia na čase a místu a neomezená dostupnost studijních materiálů) přímo vyplývá třetí, stejně významné pozitivum, že studující má možnost naplánovat si průchod studiem podle svých potřeb a schopností. [1] [2] [5]

Elektronické vzdělávání umožňuje účastníkům postup kurzem podle vlastního pracovního tempa, mohou si individuálně upravit studijní plán podle toho, jak potřebují, zvolit si obsah, který je zajímá (či si jen chtějí prohloubit své dosavadní znalosti), mají přístup k dalším zdrojům informací a navíc se mohou kdykoli k již probrané látce vrátit. [1] [9] [10]

1. 2. 1. 4 Aktuálnost výukových materiálů

Další výhodou elektronického vzdělávání je jednoduchá správa výukového obsahu. E-learningové kurzy obsahují nástroje, které usnadňují provádění úprav a doplňování studijních materiálů a umožňují tak rychle reagovat na vývoj daného vědního oboru. Souvisejícím přínosem e-learningu pak je, že studující (v případě online formy výuky) mají k dispozici vždy aktuální výukové materiály. [1] [2] [5]

1. 2. 1. 5 Multimediální výukový obsah

Velkým přínosem e-learningu jsou multimediální studijní materiály, které nejsou již jen o pouhém textu, někdy doplněném o obrázky. Naopak – součástí výukového obsahu se stala různá videa a animace, které pomohou studujícímu lépe pochopit zkoumaný problém, simulace, ve nichž si může vyzkoušet vlastní řešení situace a rovnou vidí výsledky svého rozhodnutí, nebo výukové hry, které mu zase pomohou procvičovat nově nabyté znalosti formou zábavné hry. [2] [5]
Multimediální studijní materiály navíc cílí na více smyslů najednou a pomáhají tak lépe a rychleji porozumět nové látce a v případě interaktivních materiálů zároveň poskytují okamžitou zpětnou vazbu studujícímu. [2] [5]

1. 2. 1. 6 Testování znalostí a okamžitá zpětná vazba

Nesporným přínosem elektronického vzdělávání jsou i různé testy a otevřené úkoly pro ověřování nabytých znalostí. Výhodou automatizovaného testování je, že studující mají k dispozici okamžitou zpětnou vazbu – na základě získaného počtu bodů či procentuální úspěšnosti vidí, jak si v daném tématu stojí a mohou se tak sami rozhodnout, zda se potřebují k probrané látce ještě vrátit nebo mohou pokračovat dalším tématem elektronického kurzu. Otevřené úkoly zase slouží k odevzdávání seminárních prací nebo projektů. V tomto případě může být hodnocení v podobě získaného počtu bodů navíc doplněno o slovní hodnocení, které může být pro studující velmi účinným motivačním nástrojem. [1] [2] [9]

1. 2. 1. 7 Komunikace mezi studujícími a tutory

Elektronické vzdělávání poskytuje široké spektrum nástrojů umožňujících komunikaci všech účastníků vzdělávacího procesu. Komunikace může být realizována asynchronně pomocí zpráv, e-mailů, diskusního fóra, sdíleného whiteboardu, nebo synchronně prostřednictvím chatu, instant messagingu či audio a videokonferecí. [1] [2] [7]

1. 2. 1. 8 Výhody z pohledu tvůrců elektronických kurzů

Vzhledem k nezávislosti výuky na sídle vzdělávací instituce mohou autoři elektronického kurzu najímat odborníky na pozice tutorů z různých míst. [1] [2] [5]
Výhodou také je, že v elektronickém kurzu se může vzdělávat prakticky neomezený počet studujících. Je-li ovšem kurz vedený pod dohledem tutorem, je vhodné jejich množství regulovat, aby byl tutor schopen všem studujícím poskytovat kvalitní zpětnou vazbu. [1] [2] [5]
Dalším pozitivem elektronické formy vzdělávání je, že přes vysoké počáteční náklady na výrobu kurzu a pořízení technického vybavení, jsou další náklady na provoz a údržbu elektronického kurzu již minimální. Zároveň při aktualizaci studijních materiálů odpadá nutnost jejich opětovné distribuce. [11]
Výhodou rovněž je, že průběhu výuky prostřednictvím elektronického kurzu autoři získávají zpětnou vazbu od studujících, na základě které mohou tvůrci kurz upravovat a vylepšovat ku prospěchu studujících. [1] [2]

1. 2. 1. 9 Šetření finančních nákladů

Na závěr je také vhodné zmínit, že elektronická forma vzdělávání šetří náklady jak na straně studujících, tak i u vzdělávací instituce a tutorů. [1] [2] [5]
Studující nemusí nikam dojíždět, jsou-li zaměstnáni v pracovním poměru, nemusí si brát neplacené volno či dovolenou, popřípadě se nemusí uvolňovat ze školy, pokud studují zároveň i prezenčně jinou školu. [1] [2] [5]
Na straně vzdělávací instituce se zase jedná o úsporu za energie (topení, světlo, voda), nájem a jiné. [1] [2] [5] [9]

1. 2. 2 Nevýhody e-learningu

1. 2. 2. 1 Nepostradatelnost vlastní motivace ke studiu

Jak již bylo zmíněno dříve, velkým pozitivem e-learnigové formy výuky je, že studující nejsou omezováni žádným časovým rozvrhem a studiu se tak mohou věnovat kdykoli, kdy mají čas a chuť. Pokud jim ale e-learningový kurz má přinést nějaké výsledky, nesmí jim chybět vlastní motivace, aby skutečně výuce věnovali svůj čas. Nevýhodou elektronického vzdělávání je, že studující se špatnými studijními návyky, malou motivací nebo neschopností naplánovat si svůj osobní čas, mohou být nakonec ve studiu pomocí této formy výuky neúspěšní. [1] [2] [9]

1. 2. 2. 2 Závislost na technologiích

Vzhledem k tomu, že je elektronické vzdělávání realizováno pomocí počítačů nebo moderních mobilních zařízení, je vyžadováno, aby studující některým z těchto zařízení disponovali. V případě online formy e-learningu je navíc nutné mít připojení k síti. [2] [5] [9]

1. 2. 2. 3 Nevhodnost pro určité typy studujících a kurzů

Nevýhodou elektronického vzdělávání je, že tato forma výuky nemusí vyhovovat úplně každému. E-learningový kurz může být nevhodný pro ty studující, pro které je práce s moderními technologiemi náročná, nebo preferují tištěnou podobu studijních materiálů, jenž si mohou snadno doplňovat vlastními poznámkami. [5] [9]

Zároveň existují vědomosti či dovednosti, které lze jen velmi obtížně (nebo vůbec) předat prostřednictvím elektronického kurzu. Do této kategorie patří například hra na hudební nástroje, práce v dílnách, vaření, apod. [1]

1. 2. 2. 4 Přehlcení informačními zdroji

Vzdělávací obsah elektronických kurzů se obvykle skládá ze studijních textů, multimediálního obsahu a zdrojů, kde lze najít další informace k tématu. Pokud jsou ale studijní materiály příliš obsáhlé, mohou studující nabýt dojmu, že v rámci studia daného tématu musí projít velké množství materiálů, což může vést k poklesu jejich motivace ke studiu. [2] [5] [9]

1. 2. 2. 5 Jeden způsob výkladu látky

V prezenční výuce je možné podat novou látku studujícím více způsoby. Jestliže studující něčemu nerozuměli, není pro vyučujícího problém se k problematické látce vrátit a znovu ji vysvětlit, tentokrát ale mohou zvolit k tématu jiný přístup. V elektronickém vzdělávání je ale studijní obsah předáván tak, jak jej vytvořili autoři kurzu, a pokud není tutor k dispozici, nemají studující možnost jiného výkladu. [1]

1. 2. 2. 6 Omezený osobní kontakt

Velkou nevýhodou elektronického vzdělávání je, že se studující mohou cítit izolováni od tutora a ostatních studujících. Tutor nemusí být studujícím vždy k dispozici, když jej potřebují a studující pak musí využít některý z komunikačních prostředků. Vzhledem k tomu, že se studující věnují vzdělávání nezávisle na ostatních, může být realizace skupinových prací a spolupráce studujících na různých projektech velmi náročná. [1] [2]

1. 2. 2. 7 Nevýhody z pohledu tvůrců elektronických kurzů

Pro tvůrce elektronických kurzů jsou velkým negativem vysoké počáteční náklady na tvorbu kurzu a pořízení potřebného technického vybavení. [Drtina]
Další nevýhodou je, že ve srovnání s klasickými studijními materiály je příprava kvalitního studijního obsahu pro e-learningový kurz o mnoho náročnější a časově a finančně nákladnější. [2] [5]

2 Tvorba e-learningových kurzů

2. 1 Postup tvorby e-learningového kurzu

Před započetím tvorby elektronických kurzů je potřeba rozmyslet, pro koho je kurz určen, co je jeho cílem a jaký výukový obsah jej bude tvořit.

2. 1. 1 Vytyčení cíle e-kurzu

Na začátku tvorby e-learningového kurzu je nutné rozhodnout, jaký účel by měl daný kurz ve vzdělávacím procesu plnit. Podoba elektronického kurzu se pak bude odvíjet podle toho, zda bude kurz sloužit pouze jako doplněk k prezenční výuce, nebo se bude jednat o e-learingový kurz formou samostudia, vedeného pod dohledem tutora. [2]

2. 1. 2 Analýza cílové skupiny

E-learningový kurz je nutné tvořit s ohledem na cílovou skupinu studujících. Tvůrce kurzu by se měl zaměřit především na to, do jaké věkové kategorie budoucí uživatelé kurzu patří, jaké jsou jejich IT dovednosti, jak vysoká je jejich úroveň znalostí v daném oboru, co by od e-learningového kurzu očekávali a jaké jsou jejich dosavadní zkušenosti s e-learningem. [1] [2]

2. 1. 3 Formulace učebních cílů

Učební cíle charakterizují, co bude studující po nastudování konkrétního studijního materiálu umět, znát a co bude schopen dělat. [2]
Učební cíle jsou důležité nejen pro studující, ale i pro tutory a tvůrce e-learningových kurzů. [2]
Studujícím učební cíle poskytují představu, co je od nich v kurzu očekáváno a jaké znalosti by měli získat po prostudování dané lekce. Tutorům zase umožňují stanovené učební cíle kontrolovat postup a hodnotit dosažené výsledky studujících. Autoři e-learningových kurzů zase na základě učebních cílů rozhodují o rozsahu a uspořádání vzdělávacího obsahu. [1] [2]

2. 1. 4 Výběr vzdělávacího obsahu

Při výběru vzdělávacího obsahu pro e-learningový kurz je nutné rozhodnout, jaká látka bude do kurzu začleněna a jaký bude její rozsah a uspořádání. V rámci tohoto kroku jsou stanovovány i učební strategie budou při realizaci vzdělávacího obsahu využívány. [2]

2. 2 Zásady tvorby e-learningového kurzu

Při vytváření obsahu elektronického kurzu je vhodné vzít v úvahu následující zásady: aby byl výsledný kurz pro studující dostatečně zajímavý a přehledný, obsahoval různé motivační prvky, jenž budou vybízet k dalšímu studiu, a aby hlavně poskytoval studujícím zpětnou vazbu.

2. 2. 1 Přehlednost kurzu

Kurz, který je špatně organizován, orientace v něm je složitá a celkově působí nepřehledně, může mít na studujícího demotivující účinek. Proto je vhodné při vytváření studijního obsahu dbát především na:

- členění studijních materiálů do menších logických celků
- text studijního obsahu strukturovat do kratších odstavců, vyvarovat se dlouhým souvětím
- zvýraznění důležitých informací v textu
- odlišení rozšiřujících informací od zbytku studijního textu

Dále je vhodné vytvářet studijní materiály tak, aby byly jasné, konkrétní a srozumitelné a přidat do studijního textu statické i dynamické prvky usnadňující orientaci v textu. [2] [4] [9]

2. 2. 2 Přitažlivost studijního obsahu

Výukový obsah by měl ve studujícím vyvolat zájem o dané téma, čehož lze docílit:

- obohacením studijního textu o obrázky, grafy, schémata
- zařazením multimediálního obsahu (simulace, animace, videa, apod.)
- přidáním odkazů na další zajímavé zdroje informací k tématu
- zajímavými praktickými příklady

V neposlední řadě zajímavost studijních materiálů spočívá ve výběru vhodného obsahu dané problematiky. [2] [4]

2. 2. 3 Motivační prvky

Pro e-learningovou formu výuky platí, že motivace ke studiu je pro studující nepostradatelná. Studující s nízkou motivací mohou být ve studiu neúspěšní, a právě proto je tento faktor tolik důležitý. [1] [2] [9]

Jako motivační nástroj lze využívat:

- úvodní slovo kurzu a průvodce studiem, kde je možné pokusit se o zaujetí studujícího
- autotesty poskytující zpětnou vazbu studujícímu
- hodnocení úkolů a testů, které může být v podobě dosaženého skóre nebo slovní
- komunikaci tutora se studujícími

Největším motivačním prvkem mohou být právě různé testy a úkoly, pomocí kterých studující zjistí, jak si v daném tématu vede. [2] [9]

2. 2. 4 Zpětná vazba

Každý e-learningový kurz by měl umožňovat studujícím ověřit si nově nabyté znalosti. Ověřování znalostí může být prováděno pomocí různých testů nebo otevřených úkolů. Automatizované testy poskytují studujícím okamžitou zpětnou vazbu v podobě získaného počtu bodů či procentuální úspěšnosti. U otevřených úkolů, které slouží pro odevzdávání seminárních prací či projektů, může být rovněž hodnocení v podobě dosaženého skóre, navíc ale může být doplněno o slovní ohodnocení. [1] [2] [9]

3 Technologie použité při vývoji

V následujících kapitolách budou popsány technologie, jenž byly použity pro vývoj webové aplikace, které je věnována praktická část této práce.

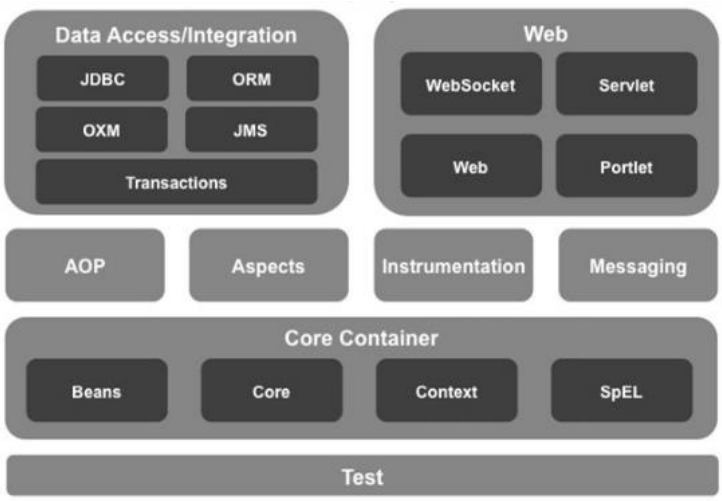
3. 1 Spring Framework

Spring Framework je komplexní open-source aplikační framework, který usnadňuje a zrychluje vývoj J2EE aplikací pro platformu Java. Předchůdcem Spring Frameworku byl Framework Interface21, který byl navržen v rámci knihy Roda Johnsona Expert One-On-One J2EE Design and Development v roce 2002. V této publikaci autor na základě vlastních zkušeností popisuje problémy vývoje J2EE aplikací a jako řešení představuje jím navržený framework Interface21. Dalším vývojem tohoto frameworku vznikla první verze Spring Frameworku, který byl představen v roce 2004 jako Spring Framework 1.0. Aktuální verzí je Spring Framework 4.0. [12] [13]

Předností Spring Frameworku je jeho modulárnost a neinvazivnost – to znamená, že vývojář může využít pouze ty moduly, které potřebuje, aniž by aplikace potřebovala další. [13]

3. 1. 1 Moduly Spring Frameworku

Spring Framework je tvořen několika moduly, které jsou seskupeny do větších logických celků: Core Container, Data Access/Integration, AOP, Aspects, Instrumentation, Messaging, Web a Test, jak je vyobrazeno na následujícím schématu. [13]



Obrázek 1 Moduly Spring Frameworku [13]

3. 1. 1. 1 Core Container

Nejdůležitější částí Spring Frameworku je komponenta Core Container, jenž se skládá z následujících čtyř modulů: Beans, Core, Context a SpEL (Spring Expression Language). [13]

- Core & Beans

  Moduly Core a Beans jsou základním stavebním kamenem celého Spring Frameworku. Zajišťují správu životního cyklu jednotlivých komponent aplikace prostřednictvím implementace BeanFactory. Bean Factory představuje centrální IoC[poznámka] kontejner, jehož úkolem je vytváření objektů, nastavování vazeb mezi nimi a poskytování těchto objektů k použití dalším komponentám.

- Context

  Modul Context staví na základech položených moduly Core a Beans. Hlavním prvkem modulu Context je rozhraní ApplicationContext, jenž rozšiřuje funkcionalitu BeanFactory například o podporu internacionalizace, zpracování zpráv a načítání externích zdrojů.

- SpEL

  Modul SpEL umožňuje použití speciálního jazyka Spring Expression Language, zkráceně SpEL, který je rozšířením standardního Expression Language ze specifikace JSP verze 2.1. Spring Expression Language se používá pro práci s objekty za běhu programu, například jej lze využít pro čtení a změnu atributů objektů, volání metod objektů, získávání obsahu kolekcí, apod.

3. 1. 1. 2 Data Access/Integration

Vrstva Data Access/Integration se skládá z modulů: JDBC, ORM, OXM, JMS a modulu Transactions. [13]

- JDBC

  Modul JDBC přináší abstraktní JDBC vrstvu, díky které není nutné psát zdlouhavý JDBC kód a zajišťovat odchytávání výjimek specifických pro danou databázi.

- ORM

  Modul ORM pracuje jako integrační vrstva pro frameworky poskytující funkcionalitu pro objektově-relační mapování jako jsou JPA, JDO a Hibernate.
- OXM

  Modul OXM poskytuje abstraktní vrstvu pro implementace Object/XML mapování jako jsou JAXB, Castor, XMLBeans a další.
- JMS

  Modul JMS (Java Messaging Service) obstarává odesílání a přijímání zpráv.
- Transactions

  Modul Transactions zajišťuje programové a deklarativní řízení transakcí pro všechny POJO (Plain Old Java Objects) a třídy, které implementují speciální rozhraní.

3. 1. 1. 3 Web

Webová vrstva se skládá ze čtyř modulů: Socket, Servlet, Web a Portlet, které slouží k integraci webových frameworků a poskytují základní webově orientované funkce – například upload souborů nebo inicializaci IoC kontejneru. Modul Servlet (také označovaný jako Web-Servlet) obsahuje implementace REST Web Services a Spring MVC, kterému bude věnována samostatná kapitola. [13]

3. 1. 1. 4 AOP, Aspects

Modul AOP poskytuje podporu pro aspektově orientované programování. Aspektově orientované programování spočívá v separování částí kódu, jenž jsou využívány opakovaně, napříč celou aplikací (například logování, transakce, validace) do takzvaných aspektů. Samostatně stojící modul Aspects umožňuje integraci s AOP frameworkem AspectJ. [13]

3. 1. 1. 5 Messaging

Messaging je modul sloužící jako základ pro aplikace na bázi zasílání zpráv. Modul dále poskytuje sadu anotací pro mapování zpráv k metodám. [13]

3. 1. 1. 6 Test

Modul Test poskytuje podporu pro testování komponent Spring Frameworku pomocí JUnit a TestNG. [13]

3. 1. 2 Dependency Injection

Základními stavebními kameny Spring Frameworku jsou Dependency Injection a Aspektově orientované programování, zkráceně AOP. [13]

Dependency Injection je forma Inversion of Control, neboli IoC. Inversion of Control je obecný návrhový vzor, na základě kterého jsou aplikace tvořeny skládáním ze znovupoužitelných komponent s cílem snížení závislostí mezi těmito komponentami. [13]

Dependency Injection je technika IoC pro vkládání závislostí mezi komponentami aplikace. Jednotlivé POJO nebo beany jsou zbaveny odpovědnosti za získávání referencí na objekty, jenž potřebují ke své činnosti, místo toho jsou jim tyto závislosti vkládány „z venku“ za běhu aplikace kontejnerem. [13]

Vložení závislostí komponentám lze provést přes konstruktor, modifikátor nebo rozhraní:

- Constructor Injection
  V případě Constructor Injection jsou závislosti vloženy v okamžiku vytváření instance komponenty kontejnerem.
- Setter Injection
  U Setter Injection vytvoří nejprve kontejner instanci pomocí konstruktoru bez parametrů a závislosti poté doplní prostřednictvím modifikátorů dané komponenty.
- Interface Injection
  Metoda Interface Injection provádí vkládání závislostí přes definované rozhraní.

Spring Framework podporuje z výše uvedených forem pouze vkládání závislostí přes konstruktor nebo prostřednictvím modifikátorů. [13]

3. 1. 3 Aspektově orientované programování

Aspektově orientované programování je koncept, jehož cílem je zvýšení modularity vyvíjené aplikace. AOP spočívá ve vyčleňování částí kódu, které jsou využívány opakovaně, do takzvaných aspektů. Typickým využitím aspektově orientovaného programování je logování událostí, validace a transakční zpracování.
Spring Framework poskytuje podporu aspektově orientovaného programování prostřednictvím modulu Spring AOP, pomocí modulu Aspects je pak možné do aplikace integrovat framework AspectsJ, který zajišťuje kompletní AOP řešení. [13]

3. 2 Spring MVC

Spring MVC Framework se používá k vývoji webových aplikací založených na architektuře Model-View-Controller neboli MVC.
Základní myšlenkou architektonického vzoru Model-View-Controller je rozdělení aplikace do tří nezávislých komponent:

- Model
  Komponenta Model je reprezentována daty aplikace, jenž jsou předávána prezentační vrstvě View, která se stará o jejich zobrazování.
- View
  Vrstva View zajišťuje zobrazování dat uživateli.
- Controller
  Komponenta Controller zpracovává požadavky a výsledek vrací zpátky prezentační vrstvě.

3. 2. 1 Proces zpracování požadavku

Zpracování požadavku od uživatele probíhá v několika krocích: [12]

1. Požadavky od uživatele přijímá `DispatcherServlet`, který jej předává ke zpracování odpovídajícímu `Controlleru`. Vzhledem ke skutečnosti, že v aplikaci obvykle bývá více než jeden `Controller`, je výběr toho správného proveden pomocí objektu `HandlerMapping`.
2. `Controller` zpracuje přijatý požadavek, jedná-li se o požadavek o data nebo na zpracování dat, `Controller` zavolá metodu příslušné `Service`, která danou akci provede. Výsledek zpracování požadavku v podobě `ModelAndView` předává `Controller` zpět `DispatcherServletu`.

3. `DispatcherServlet` pomocí objektu `ViewResolver` vyhledá `View` definované v `ModelAndView`, jež dostal od `Controlleru`.
4. Nalezenému `View` jsou předána data `Modelu`. Výsledná stránka je odeslána zpět do prohlížeče, kde je zobrazena uživateli.

3. 3 Spring Security

Spring Security je framework, který umožňuje jednoduchou implementaci zabezpečení do aplikací založených na Spring Framework. [14]
Framework Spring Security poskytuje funkcionalitu pro autentizaci a autorizaci uživatelů. Autentizací Spring Security ověřuje, zda je daný uživatel tím, za koho se vydávají, autorizací zase zjišťuje, zda má daný uživatel dostatečná oprávnění pro provedení požadované akce. [14]
O konfiguraci a používání Spring Security je dále pojednáváno v praktické části této diplomové práce.

3. 4 Hibernate

Hibernate je framework, který umožňuje objektově-relační mapování. Hibernate provádí persistenci objektů doménových tříd (entit) pomocí mapování, které může být definována na základě samostatných XML souborů nebo anotacemi. [14]
K mapování pomocí XML souborů je nutné vytvořit mapovací soubory pro jednotlivé třídy doménového modelu, které budou obsahovat název třídy a odpovídající tabulky v databázi, deklaraci atributů a názvy sloupců, na které mají být tyto hodnoty namapovány. [14]
V případě zvolení druhé varianty jsou anotace umístěny přímo do zdrojového kódu dané entity. Anotace, které definují vlastnosti jednotlivých sloupců tabulky, mohou být přidány k atributům doménové třídy nebo nad metody `get()`. [14]
Hibernate nejen zajišťuje persistenci entit do databáze, ale umožňuje také data uložená v databázi vybírat a vracet v podobě objektů. Dotazování do databáze lze provádět pomocí klasických SQL dotazů, připravených metod `get()` a `load()`, které jsou vhodné především pro získání jednoho objektu například podle jeho `id`. Pro vyhledávání v databázi lze ale také použít Hibernate Query Language (HQL) nebo kritéria. [14]

3. 4. 1 Hibernate Query Language

Hibernate Query Language neboli HQL je objektově-orientovaný dotazovací jazyk, jenž je syntaxí podobný SQL. Na rozdíl od SQL však HQL nepracuje s databázovými tabulkami, ale s objekty. Zároveň si umí poradit i s dědičností, polymorfismem a asociacemi. [14]

Nejjednodušší podoba HQL dotazu je vypadá následovně:

```
Query q = session.createQuery("from Student");
List<Student> students = query.list();
```

Tento dotaz vrátí všechny studenty, kteří se nacházejí v databázi. HQL samozřejmě podporuje spojování tabulek operátorem `JOIN`, selekci klauzulí `WHERE`, agregační funkce, řazení dat pomocí `ORDER BY` a mnoho dalšího. [14]

Výhodou je, že vyjma názvů tříd a atributů nezáleží na velikosti písmen – HQL není case-sensitivní. [14]

Více k dotazování pomocí HQL v dokumentaci [14]

3. 4. 2 Kritéria

V okamžiku, kdy je dotaz na data příliš dlouhý nebo složitý, nemusí HQL již úplně vyhovovat. Pro tyto případy se jako řešení nabízí dotazování pomocí kritérií. V tomto případě je dotaz vytvářen postupným přidáváním restrikcí, které odpovídají zadaným kritériím. [14]

Dotazování pomocí kritérií může být prováděno:

```
Criteria c = session.createCriteria(Student.class);
c.add(Restrictions.like("firstName", "J%");
c.add(Restrictions.like("lastName", "N%");
List<Student> students = c.list();
```

Výsledkem bude seznam studentů, jejichž křestní jméno začíná na písmeno J a příjmení na N. [14]

Více k dotazování pomocí kritérií v dokumentaci [14]

3. 5 PostgreSQL

PostgreSQL je open-source objektově relační databázový systém, který je možné provozovat na operačních systémech Linux, Unix a Windows. [16]
Databázový systém PostgreSQL umí pracovat s cizími klíči a provádět spojování tabulek pomocí operátoru `JOIN`. Dále umožňuje vytváření pohledů, triggerů a procedur. PostgreSQL pokrývá většinu datových typů standardu SQL92 a SQL99: `INTEGER`, `NUMERIC`, `BOOLEAN`, `CHAR`, `VARCHAR`, `DATE`, `INTERVAL` a `TIMESTAMP`. [16]
Databázový systém PostgreSQL plně splňuje seznam požadavků ACID: [16]

- Atomičnost (Atomic)
  Databázová transakce je dále nedělitelná jednotka, která je buď provedena celá, nebo vůbec.
- Konzistence (Consistent)
  Transakce převádí databázi z jednoho konzistentního stavu do druhého.
- Izolace (Isolated)
  Probíhající transakce není ovlivněna ostatními zpracovávanými transakcemi a zároveň tato transakce nezasahuje do ostatních.
- Trvanlivost (Durable)
  Po dokončení transakce jsou provedené změny trvalé.

PostgreSQL lze obsluhovat pomocí příkazové řádky klientem psql nebo prostřednictvím nástrojů pgAdmin nebo phpPgAdmin. [16]

3. 7 Apache Tiles

Apache Tiles je open-source framework, jenž zjednodušuje a zrychluje vývoj uživatelského rozhraní webových aplikací. Apache Tiles umožňuje skládání stránky, která je viděna uživatelem jako celek, z jednotlivých fragmentů na základě vytvořené šablony. [17]
Definice všech šablon se nacházejí v XML souboru, jenž pro každou šablonu obsahuje název, na kterou je poté odkazováno v jsp stránce a jednotlivé komponenty, ze kterých se skládá – například hlavička, menu, patička, apod. [17]
Popis konfigurace a použití frameworku Apache Tiles se nachází v praktické části této diplomové práce.

II PRAKTICKÁ ČÁST

4 Analýza a návrh webové aplikace

Rozvržení této kapitoly bude odpovídat prvním dvou fázím vývoje aplikace: analýza a návrh. V první části bude představeno zadání projektu, jenž je předmětem této diplomové práce. Následovat budou funkční a nefunkční požadavky, které byly stanoveny pomocí analýzy zadání, poté model případů užití, jenž bude zahrnovat popis uživatelských rolí a jednotlivých případů užití. Závěr této kapitoly bude věnován modelu tříd.

4. 1 Zadání projektu

Zadání projektu bylo vypracováno na základě odborných konzultací s vyučujícími na 1. stupni základní školy. Učitelé byli tázáni, co by od aplikace očekávali, jak by mělo procvičování probrané látky probíhat a jaké výstupy by chtěli mít k dispozici.
Pro lepší přehlednost bylo zadání rozděleno do pěti částí – popis uživatelů v aplikaci, průběh procvičování látky, požadované výstupy z procvičování žáků, motivační systém pro žáky a vzhled aplikace.

4. 1. 1 Uživatelé

V aplikaci se budou vyskytovat tři druhy uživatelů – žák, učitel a učitel s právy administrátora. Při vstupu do aplikace bude vyžadováno, aby uživatel zadal své přihlašovací jméno a heslo. Pokud autentizace proběhne úspěšně, bude přihlášený uživatel přesměrován na svou domovskou stranu, jejíž obsah se bude lišit podle toho, jakým oprávněním disponuje. V opačném případě bude informován o tom, že zadal nesprávné přihlašovací jméno nebo heslo.
Aplikace bude umožňovat vedení evidence žáků a učitelů. Bude možné přidávat nové žákya učitele, upravovat stávající a v případě potřeby i mazat z databáze. U žáků bude vyžadováno vyplnit jméno, příjmení a přezdívku, pod kterou bude žák vystupovat v aplikaci vůči svým spolužákům, aby byla zachována jistá míra anonymity. Žáci budou mít samozřejmě možnost si svou přezdívku změnit. Zároveň každý žák v aplikaci bude muset patřit do nějaké třídy, u které bude stačit evidovat pouze její název. Třídy bude možné spravovat stejným způsobem jako

žáky a učitele – bude tedy možné třídy přidávat, upravovat a mazat. U učitelů bude vyžadováno evidovat jméno a příjmení, v případě potřeby bude možné mít pro každého učitele uloženou navíc i jeho e-mailovou adresu a telefonní číslo. Učitelům také půjde přidělit práva administrátora. Správa žáků a tříd bude dostupná uživatelům s oprávněním „Učitel“ nebo „Administrátor, spravovat učitele a odstraňovat celé třídy (včetně všech žáků), budou moci ale už jen administrátoři. V rámci správy žáků, tříd a učitelů bude vyžadováno, aby před každým vymazáním záznamu z databáze musel uživatel potvrdit, že si skutečně přeje tuto akci provést.

4. 1. 2 Procvičování

Všichni uživatelé si budou moci nechat vypsat dostupné moduly (výraz pro učitele, pro žáky bude „modul“ zaměněn termínem „předmět“). Výběrem modulu (předmětu) bude uživatel přesměrován na stránku s přehledem témat (včetně všech podtémat), ze kterých se vybraný modul skládá.

Pokud bude uživatelem žák, uvidí u každého podtématu statistiku svého procvičování, která bude obsahovat: celkový počet dokončených cvičení, počet správných odpovědí a počet nesprávných. Tato statistika bude sloužit žákovi jako zpětná vazba, aby viděl, která podtémata mu stále ještě dělají problémy, a měl by se jim tím pádem věnovat přednostně. Vybráním požadovaného podtématu bude žák rovnou přesměrován na procvičování. Každé cvičení se bude skládat z deseti testových otázek, které budou vybírány na základě toho, jak v minulosti žák na tyto otázky odpovídal. Není nutné zaznamenávat pro každou otázku žákovu odpověď, nýbrž stačí ukládat informaci o tom, zda odpověděl správně či nesprávně – aby mu později aplikace vybírala k procvičování ty otázky, které mu dělají největší problémy. Po zodpovězení testové otázka žák uvidí,zda odpověděl správně či nikoliv. V případě nesprávné odpovědi mu aplikace ukáže jeho vyplněnou odpověď a správnou odpověď. V budoucnu by bylo vhodné nesprávně zodpovězené otázky zařadit na závěr procvičování. Po vyplnění poslední testové otázky by se měl žákovi zobrazit přehled všech testových otázek daného procvičování, vyplněné a správné odpovědí a poskytnout tak žákovi zpětnou vazbu, jak si v daném procvičování vedl.

Učitel si bude moci v přehledu všech témat (a podtémat) modulu nechat zobrazit detail podtématu, který bude obsahovat zadání pro dané procvičování, nápovědu a výpis testových otázek, které do něj spadají. Bylo by také vhodné, aby měl učitel

k dispozici kromě obyčejného výpisu testových otázek i otázky seřazené podle toho, jak na ně žáci v procvičování odpovídají – tedy od nejproblematičtějších až po ty (pro žáky) nejsnazší.

4. 1. 3 Výstupy z dat

Učitelé budou mít k dispozici přehledy, jak žáci procvičují jednotlivá témata (resp. podtémata) modulů. Budou mít možnost si zobrazit nejen souhrnné přehledy pro celou třídu, ale i statistiky procvičování (počet dokončených cvičení a počty správných a nesprávných odpovědí) pro každého žáka.
Aplikace také umožní učitelům náhled do testových otázek pro každé podtéma, které budou seřazeny na základě odpovědí žáků. Bylo by vhodné, aby učitel viděl, na které otázky žáci odpovídají nejčastěji nesprávně, aby se k problematické látce mohl vrátit v hodinách výuky.

4. 1. 4 Motivace

Na závěr každého procvičování bude žákovi zobrazen výsledek, jak si v daném cvičení vedl. Tento přehled bude mimo jiné obsahovat navíc i slovní ohodnocení, které se bude lišit podle toho, kolik bodů žák v procvičování získal – aby byl například v případě nízkého skóre vybídnut k dalšímu procvičování. Dále budou žáci za určitý počet dokončených cvičení odměňováni. Bylo by vhodné, aby si žák mohl zobrazit stránku s obrázky – odměnami, které se budou přidávat podle počtu dokončených procvičování v rámci jednoho podtématu. Žáci jedné třídy by měli být také motivováni navzájem mezi sebou – k tomu budou sloužit žebříčky, které budou ukazovat, jak si v procvičování vedou vůči svým spolužákům. Žáci si tedy budou moci zobrazit pro každý předmět žebříček nejpilnějších žáků podle počtu dokončených procvičování. Aby ale žáci nebyli sváděni k dokončování co nejvíce procvičování bez ohledu na výsledek, uvidí i druhý žebříček, který bude seřazen podle poměru správných a nesprávných odpovědí za všechna procvičování v rámci jednoho předmětu. Tyto žebříčky ale budou obsahovat pouze deset nejpilnějších žáků, aby se ti, kterým látka ještě nejde, necítili naopak demotivováni. Zároveň jména a příjmení žáků budou zastoupeny přezdívkami, aby byla zachována určitá míra anonymity pro ty žáky, kteří si nepřejí, aby jejich spolužáci viděli, jaké jsou výsledky jejich procvičování.

4. 1. 5 Vzhled

Aplikace bude přehledná, jednoduchá a srozumitelná, žáci budou schopni se v aplikaci rychle zorientovat. Dále nebude obsahovat žádné rušivé elementy, které by žákům při procvičování zbytečně odváděly pozornost.

4. 2 Specifikace požadavků

V následujících dvou kapitolách budou představeny funkční požadavky, které popisují, jaké chování je od vyvíjené aplikace vyžadováno, a nefunkční, které zachycují implementační omezení. Tyto požadavky byly stanoveny na základě analýzy zadání projektu.

4. 2. 1 Funkční požadavky

Z předchozího zadání projektu byly vytyčeny následující funkční požadavky. Každý z požadavků byl označen číslem a prioritou, která byla stanovena na základě důležitosti vůči ostatních funkčním požadavkům.

4. 2. 1. 1 Přihlašování do aplikace

Číslo	FP 1
Název	Přihlašování do aplikace
Popis	Ke vstupu do aplikace bude vyžadováno zadání přihlašovacích údajů. Nepřihlášený uživatel uvidí pouze dialog k přihlášení. Po zadání nesprávného přihlašovacího jména nebo hesla bude uživateli zobrazena zpráva o nezdařeném přihlášení.
Priorita	1

4. 2. 1. 2 Správa učitelů

Číslo	FP 2
Název	Správa učitelů
Popis	Aplikace bude umožňovat spravovat uživatelské účty učitelů. U každého učitele bude evidováno: jméno, příjmení, e-mail (nepovinná položka), telefonní číslo (nepovinná položka) a oprávnění. Uživatelské účty učitelů bude možné přidávat, upravovat i mazat, před vymazáním záznamu bude uživatel muset potvrdit, zda si skutečně přeje odstranit uživatelský účet daného učitele. Správa učitelů bude dostupná pouze uživatelům, kteří disponují oprávněním „Administrátor“.
Priorita	1

4. 2. 1. 3 Správa tříd

Číslo	FP 3
Název	Správa tříd
Popis	Aplikace bude umožňovat spravovat třídy. U každé třídy bude evidován pouze její název. Třídy bude možné přidávat, upravovat i mazat, před vymazáním záznamu bude uživatel muset potvrdit, zda si skutečně přeje odstranit danou třídu (i se všemi žáky). Možnost přidávat a upravovat třídy bude dostupná pouze uživatelům s oprávněním „Učitel“ nebo „Administrátor“, odstraňovat třídy bude moci pouze administrátor.
Priorita	1

4. 2. 1. 4 Správa žáků

Číslo	FP 4
Název	Správa žáků
Popis	Aplikace bude umožňovat spravovat uživatelské účty žáků. U každého žáka bude evidováno: jméno, příjmení a přezdívka. Uživatelské účty žáků bude možné přidávat, upravovat i mazat, před vymazáním záznamu bude uživatel muset potvrdit, zda si skutečně přeje odstranit

	uživatelský účet daného žáka. Správa žáků bude dostupná pouze uživatelům, kteří disponují oprávněním „Učitel“ nebo „Administrátor“.
Priorita	1

4. 2. 1. 5 Přehled dostupných modulů (předmětů)

Číslo	FP 5
Název	Přehled dostupných modulů (předmětů)
Popis	Uživatelé (učitelé i žáci) si budou moci zobrazit seznam dostupných modulů, resp. předmětů. Vybráním předmětu budou přesměrováni na seznam témat, který se bude lišit podle toho, zda je přihlášeným uživatelem učitel nebo žák (viz. FP 6 a FP 7).
Priorita	1

4. 2. 1. 6 Přehled témat modulu (předmětu) pro učitele

Číslo	FP 6
Název	Přehled témat modulu (předmětu) pro učitele
Popis	Učitelé (uživatelé s oprávněním „Učitel“) si budou moci zobrazit přehled témat vybraného modulu. U každého tématu bude mít učitel rovnou k dispozici i seznam podtémat, které dané téma obsahuje. Současně bude mít učitel možnost si u každého podtématu nechat zobrazit výpis testových otázek, které do něj spadají (viz. FP 7), a výpis testových otázek seřazených od (pro žáky) nejproblematičtějších až po ty nejsnazší (viz. FP 8).
Priorita	1

4. 2. 1. 7 Výpis testových otázek podtématu

Číslo	FP 7
Název	Výpis testových otázek podtématu
Popis	Učitelé si budou moci zobrazit výpis testových otázek vybraného podtématu. V přehledu uvidí zadání úkolu pro dané podtéma a nápovědu, které se zobrazují žákovi při procvičování. Dále bude následovat seznam testových otázek daného podtématu v podobě, jak je vidí žák.
Priorita	1

4. 2. 1. 8 Výpis testových otázek podtématu podle náročnosti

Číslo	FP 8
Název	Výpis testových otázek podtématu podle náročnosti
Popis	Učitelé budou mít možnost zobrazit si výpis testových otázek seřazených dle náročnosti od (pro děti) nejtěžších po nejsnazší. Pořadí bude vypočteno na základě toho, jak děti na danou testovou otázku odpovídaly – tedy zda jejich odpověď byla správně nebo nesprávně. Přehled bude obsahovat zadání pro dané podtéma, nápovědu a seznam testových otázek, které do něj spadají.
Priorita	2

4. 2. 1. 9 Přehled témat modulu (předmětu) pro žáky

Číslo	FP 9
Název	Přehled témat modulu (předmětu) pro žáky
Popis	Žáci (uživatelé s oprávněním „Žák“) budou mít možnost zobrazit si seznam témat vybraného modulu. U každého tématu bude mít žák rovnou k dispozici i seznam podtémat, které téma obsahuje. Současně uvidí statistiku procvičovaní daného podtématu (viz FP 10). Vybráním podtématu bude rovnou přesměrován na procvičování (viz. FP 11).
Priorita	1

4. 2. 1. 10 Evidence výsledků procvičovaní žáka za každé podtéma

Číslo	FP 10
Název	Evidence výsledků procvičovaní žáka pro každé podtéma
Popis	Žák bude mít k dispozici pro každé podtéma statistiku procvičování. Uvidí počet dokončených cvičení, počet správně zodpovězených otázek v rámci daného podtématu a počet otázek, na které odpověděl nesprávně.
Priorita	1

4. 2. 1. 11 Procvičování

Číslo	FP 11
Název	Procvičování
Popis	Každé procvičování se bude skládat z deseti testových otázek. Do procvičování budou přednostně vybírány ty otázky, které v minulosti žákovi nešly (viz. FP 12). Žák po zadání (resp. vyplnění či vybrání) odpovědi uvidí, zda na aktuální otázku zodpověděl správně či nikoliv. Pokud žák odpověděl na testovou otázku špatně, aplikace mu ukáže, jak měla odpověď správně vypadat a jakou vyplnil on. Po zodpovězení všech otázek žák uvidí výsledek procvičování (viz. FP 13).
Priorita	1

4. 2. 1. 12 Výběr testových otázek pro žáka

Číslo	FP 12
Název	Výběr testových otázek pro žáka
Popis	Testové otázky pro procvičování budou vybírány na základě toho, jak v předchozích cvičeních na danou otázku žák odpověděl (správně nebo nesprávně). Ve cvičeních se tak přednostně budou vyskytovat ty otázky, jenž žákovi v minulosti nešly, aby měl větší možnost si zapamatovat správně řešení a v budoucnu již chyby neopakovat.
Priorita	1

4. 2. 1. 13 Evaluace procvičování

Číslo	FP 13
Název	Evaluace procvičování
Popis	Po zodpovězení všech testových otázek bude žákovi ukázán přehled, jak si v daném procvičování vedl. Přehled bude obsahovat zadání testové otázky, žákovu vyplněnou odpověď a správnou odpověď. Ve výsledku také uvidí, na kolik otázek z celkového počtu odpověděl správně, a slovní ohodnocení, které se bude lišit podle dosaženého skóre.
Priorita	1

4. 2. 1. 14 Opakování nesprávně zodpovězených otázek

Číslo	FP 14
Název	Opakování nesprávně zodpovězených otázek
Popis	Pokud žák odpoví na testovou otázku nesprávně, bude tato otázka v rámci probíhajícího procvičování zařazena na závěr – tato akce se bude opakovat tolikrát, dokud žákova odpověď na otázku nebude správně.
Priorita	3

4. 2. 1. 15 Výsledky žáků (žebříčky) pro učitele

Číslo	FP 15
Název	Výsledky žáků (žebříčky) pro učitele
Popis	Učitelé si budou moci zobrazit pro každou třídu (a předmět) statistiku procvičování žáků. Po vybrání třídy uvidí seznam žáků, kde bude mít k dispozici pro každého žáka celkový počet dokončených cvičení (za celý modul, resp. předmět) a celkový počet všech správných a nesprávných odpovědí. Učitel bude mít navíc možnost si pro každého žáka nechat zobrazit podrobný výpis výsledků (viz FP 16).
Priorita	2

4. 2. 1. 16 Výsledky žáků (podrobný výpis) pro učitele

Číslo	FP 16
Název	Výsledky žáků (podrobný výpis) pro učitele
Popis	Učitelé budou mít možnost si pro každého žáka nechat zobrazit statistiku procvičování. V přehledu uvidí počty dokončených cvičení, správných a nesprávných odpovědí za každé podtéma, téma a modul (předmět).
Priorita	3

4. 2. 1. 17 Výsledky žáků (žebříčky) pro žáky

Číslo	FP 17
Název	Výsledky žáků (žebříčky) pro žáky
Popis	Žáci si budou moci pro každý předmět (v rámci své třídy) zobrazit žebříčky nejpilnějších žáků. Kritériem prvního srovnání bude počet dokončených cvičení (za celý předmět), pořadí žáků v druhém žebříčku bude na základě správných a nesprávných odpovědí (opět za celý předmět). Jména a příjemní v seznamech budou zastoupeny přezdívkami, které si žáci budou moci sami zvolit (viz. FP 18).
Priorita	3

4. 2. 1. 18 Zastoupení jména a příjmení žáka přezdívkou

Číslo	FP 18
Název	Zastoupení jména a příjmení žáka přezdívkou
Popis	Žáci v aplikaci budou vůči svým spolužákům vystupovat pouze pod svou přezdívkou, kterou si budou moci sami změnit.
Priorita	2

4. 2. 1. 19 Systém odměn za procvičování

Číslo	FP 19
Název	Systém odměn za procvičování
Popis	Žáci budou odměňováni za určitý počet procvičování v rámci jednoho podtématu. Žáci si budou moci zobrazit stránku s odměnami, která bude obsahovat získané samolepky za 10, 20 a 50 dokončených cvičení za každé podtéma.
Priorita	3

4. 2. 2 Nefunkční požadavky

Z rozboru zadání projektu byly vytyčeny také nefunkční požadavky, které vyjadřují implementační omezení vyvíjené aplikace. Stejně jako u funkčních požadavků byly i nefunkční požadavky označeny číslem a prioritou.

4. 2. 2. 1 Uživatelské rozhraní

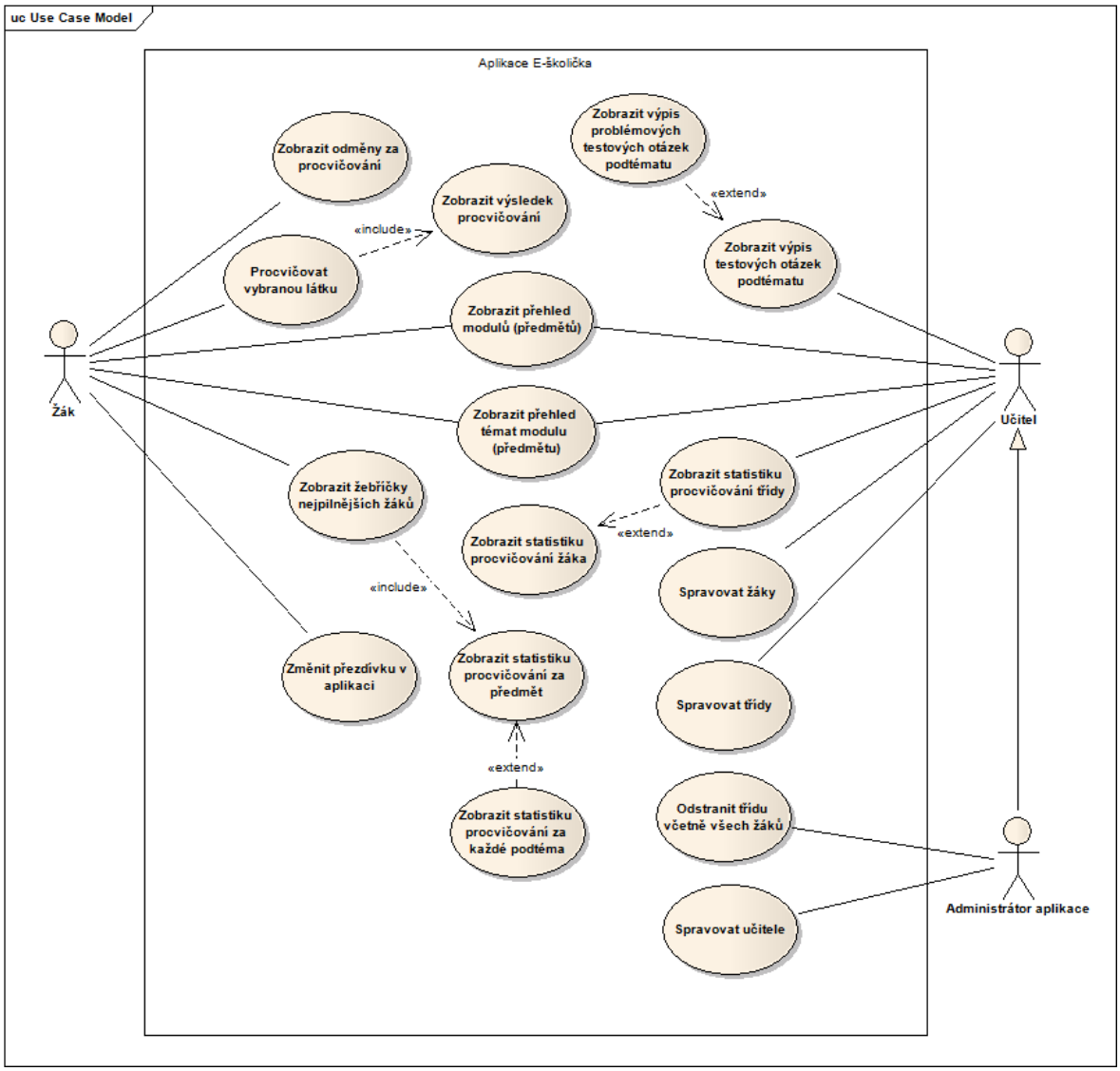
Číslo	NP 1
Název	Uživatelské rozhraní
Popis	Aplikace bude uživatelsky přívětivá – přehledná, srozumitelná a jednoduchá na ovládání.
Priorita	1

4. 2. 2. 2 Vzhled aplikace

Číslo	NP 2
Název	Vzhled aplikace
Popis	Aplikace nebude obsahovat žádné rušivé elementy, které by mohly žákům při procvičování odvádět pozornost.
Priorita	1

4. 3 Model případů užití

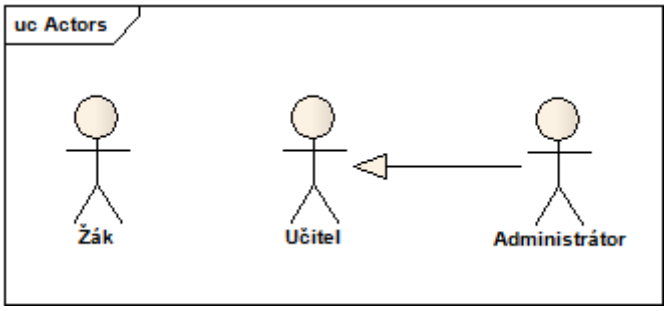
Následující model případů užití vychází z analýzy funkčních požadavků. Pro zjednodušení modelu jsou případy užití, které představují CRUD operace, sjednoceny do jednoho případu užití.



Obrázek 2 Model případů užití

4. 3. 1 Aktéři

Z analýzy zadání projektu vyplynulo, že se v aplikaci budou vyskytovat následující uživatelské role resp. aktéři: Žák, Učitel a Učitel s právy administrátora.



Obrázek 3 Aktéři

4. 3. 1. 1 Žák

Uživatelská role Žák představuje skutečného žáka, který se bude přihlašovat do aplikace. Žáci mají možnost si zobrazit všechny předměty, které mají dostupné a z nich poté pokračovat na téma (resp. podtéma), které chtějí procvičovat.

Žáci mohou v aplikaci libovolně spouštět procvičování jednotlivých podtémat, zobrazovat si statistiku svého procvičování a také žebříčky, aby viděli, jak si vedou vůči svým spolužákům.

Každý žák si také může prohlížet získané odměny (samolepky) za splnění určitého počtu cvičení v rámci každého podtématu.

4. 3. 1. 2 Učitel

Uživatelé vystupující v roli Učitel si stejně jako žáci mohou zobrazovat přehled všech modulů, včetně témat a podtémat, která obsahují. Pro každé podtéma zvoleného modulu si pak mohou nechat vypsat seznam testových otázek, které do něj patří. Učitelé si mohou dále prohlédnout testové otázky, které žákům dělají v cvičeních největší problémy, aby se k nim v případě potřeby mohli v rámci výuky vrátit.

Učitelé si také mohou prohlížet podrobné i souhrnné statistiky procvičování jednotlivých žáků popřípadě celých tříd za každý modul.

Mimo jiné také disponují právy pro správu žáků a tříd v aplikaci. Oproti uživatelské roli Učitel s právy administrátora ale nemají dostatečná oprávnění pro odstraňování celých tříd (včetně všech jejích žáků).

4. 3. 1. 3 Učitel s právy administrátora

Uživatelská role Učitel s právy administrátora zastupuje učitele, kterému byla navíc přidělena práva administrátora. Tato uživatelská role rozšiřuje roli Učitel, z toho vyplývá, že uživatel vystupující v této roli může v aplikaci provádět stejné úkony, jako učitelé, navíc má ale k dispozici i správu učitelů a oproti roli Učitel může i odstraňovat z databáze celé třídy včetně všech žáků, které obsahují.

4. 3. 2 Případy užití

V následujících podkapitolách budou popsány jednotlivé případy užití, které vyplynuly z analýzy funkčních požadavků.

4. 3. 2. 1 Spravovat žáky

Spravovat žáky mohou pouze Učitelé nebo Učitelé s právy administrátora. Tento případ užití zahrnuje veškeré CRUD operace nad žáky – tedy přidávání nových žáků, zobrazování informací o nich, jejich úpravu i mazání z databáze.

4. 3. 2. 2 Spravovat třídy

Správu tříd mají stejně jako u předchozího případu užití na starosti Učitelé, popř. Učitelé s právy administrátora. Případ užití Spravovat třídy zahrnuje pouze přidávání nových tříd, zobrazování jejich detailu a úpravu údajů o třídě. Odstraňovat třídy smí již pouze uživatelé, kteří vystupují v roli Učitel s právy administrátora.

4. 3. 2. 3 Odstranit třídu včetně všech žáků

Odstraňovat celou třídu (včetně všech žáků, které obsahuje) mohou pouze Učitelé s právy administrátora. Z tohoto důvodu je tento případ užití vyčleněn zvlášť a není součástí případu užití Spravovat třídy.

4. 3. 2. 4 Spravovat učitele

Správu učitelů mají na starosti Učitelé s právy administrátora. Případ užití Spravovat učitele je opět souhrnným případem užití, který v sobě zahrnuje přidávání nových učitelů, zobrazování a úpravu jejich údajů a odstraňování učitelů z evidence.

4. 3. 2. 5 Zobrazit přehled modulů (předmětů)

Všichni aktéři (tedy Žák, Učitel i Učitel s právy administrátora) si mohou zobrazit přehled všech modulů (resp. předmětů), které mají k dispozici.

4. 3. 2. 6 Zobrazit přehled témat modulu (předmětu)

Stejně jako v předchozím případu užití i funkcionalita Zobrazit přehled témat modulu (předmětu) je dostupná všem uživatelským rolím. Samotný výpis témat (včetně podtémat) se pak liší podle toho, je-li aktérem Žák nebo Učitel (popř. Učitel s právy administrátora). V případě Žáka je výpis témat doplněn o statistiky procvičování jednotlivých podtémat, Učitel má k dispozici zase možnost zobrazit si pro každé podtéma seznam testových otázek.

4. 3. 2. 7 Zobrazit výpis testových otázek podtématu

Učitelé a Učitelé s právy administrátora si mohou zobrazit seznam všech testových otázek vybraného podtématu.

4. 3. 2. 8 Zobrazit výpis problémových testových otázek podtématu

Případ užití Zobrazit výpis problémových testových otázek podtématu rozšiřuje případ užití Zobrazit výpis testových otázek podtématu. V tomto případě se nejedná o obyčejný výpis otázek, nýbrž jsou testové otázky seřazeny podle toho, jak dělají žákům v procvičováních problémy – tedy od (pro žáky) nejnáročnějších až po ty nejsnazší.

4. 3. 2. 9 Zobrazit statistiku procvičování třídy

Učitel má možnost zobrazit si statistiku procvičování celé třídy, která pro každého žáka ukazuje celkový počet dokončených cvičení a počty správných a nesprávných odpovědí. Navíc si může pro každého žáka nechat zobrazit podrobný výpis o procvičování. Tato funkcionalita je vyjádřena případem užití Zobrazit statistiku procvičování třídy, která může být rozšířena o případ užití Zobrazit statistiku procvičování žáka.

4. 3. 2. 10 Zobrazit statistiku procvičování žáka

Zobrazit statistiku procvičování žáka rozšiřuje případ užití Zobrazit statistiku procvičování třídy. Uživatel v roli Učitel (popř. Učitel s právy administrátora) si může zobrazit statistiku procvičování vybraného žáka, která se skládá z jednotlivých statistik za každý modul, téma a podtéma.

4. 3. 2. 11 Procvičovat vybranou látku

Případ užití Procvičovat vybranou látku přestavuje veškeré akce spojené s procvičováním – tedy spuštění testu, zodpovídání testových otázek a získání zpětné vazby o správnosti vyplněné odpovědi. Procvičovat zvolené podtéma mohou pouze uživatelé v roli Žák.

4. 3. 2. 12 Zobrazit výsledek procvičování

Zobrazit výsledek procvičování rozšiřuje případ užití Procvičovat vybranou látku. V okamžiku, kdy Žák zodpoví poslední testovou otázku cvičení, uvidí přehled všech otázek, svých vyplněných odpovědí a odpovědí, jak měly být správně. Dále mu bude poskytnuta zpětná vazba v podobě dosaženého skóre a slovního ohodnocení, aby viděl, jak si v daném cvičení vedl.

4. 3. 2. 13 Zobrazit žebříčky nejpilnějších žáků

Žák si může pro každý předmět zobrazit žebříčky nejpilnějších žáků své třídy. Tato akce představuje zobrazení dvou žebříčků – výpis deseti nejpilnějších žáků podle celkového počtu dokončených cvičení za předmět a výpis žáků podle celkového poměru správných a nesprávných odpovědí. Za účelem zachování anonymity žáků budou jména a příjmení nahrazena přezdívkami, které si mohou žáci zvolit. Případ užití Zobrazit žebříčky nejpilnější žáků zahrnuje rovněž případ užití Zobrazit statistiku procvičování za předmět.

4. 3. 2. 14 Zobrazit statistiku procvičování za předmět

Aby žák viděl, jak si stojí vůči svým spolužákům v procvičování v každém předmětu, má možnost si zobrazit žebříčky se statistikou procvičování – viz případ užití Zobrazit žebříčky nejpilnějších žáků. Součástí těchto výpisů je statistika procvičování za celý předmět daného žáka, aby měl možnost vidět,

o kolik procvičování je za svými spolužáky, jestliže se jeho přezdívka nenachází v žebříčku nejpilnějších žáků.

4. 3. 2. 15 Zobrazit statistiku procvičování za každé podtéma

Případ užití Zobrazit statistiku procvičování za každé podtéma představuje výpis statistik procvičování přihlášeného žáka (počet dokončených cvičení a počty správných a nesprávných odpovědí) za každé podtéma. Tato akce je dostupná pouze uživatelům v roli Žák.

4. 3. 2. 16 Zobrazit odměny za procvičování

Žák je v rámci každého podtématu odměňován za každých 10, 20 a 50 dokončených procvičování. Žák si může tyto nasbírané odměny nechat zobrazit, tato akce je reprezentována případem užití Zobrazit odměny za procvičování.

4. 3. 2. 17 Změnit přezdívku v aplikaci

Případ užití Změnit přezdívku v aplikaci představuje možnost si změnit přezdívku, pod kterou žák vystupuje v aplikaci vůči svým spolužákům. Tato funkcionalita je dostupná pouze uživatelům v roli Žák.

4. 4 Model tříd

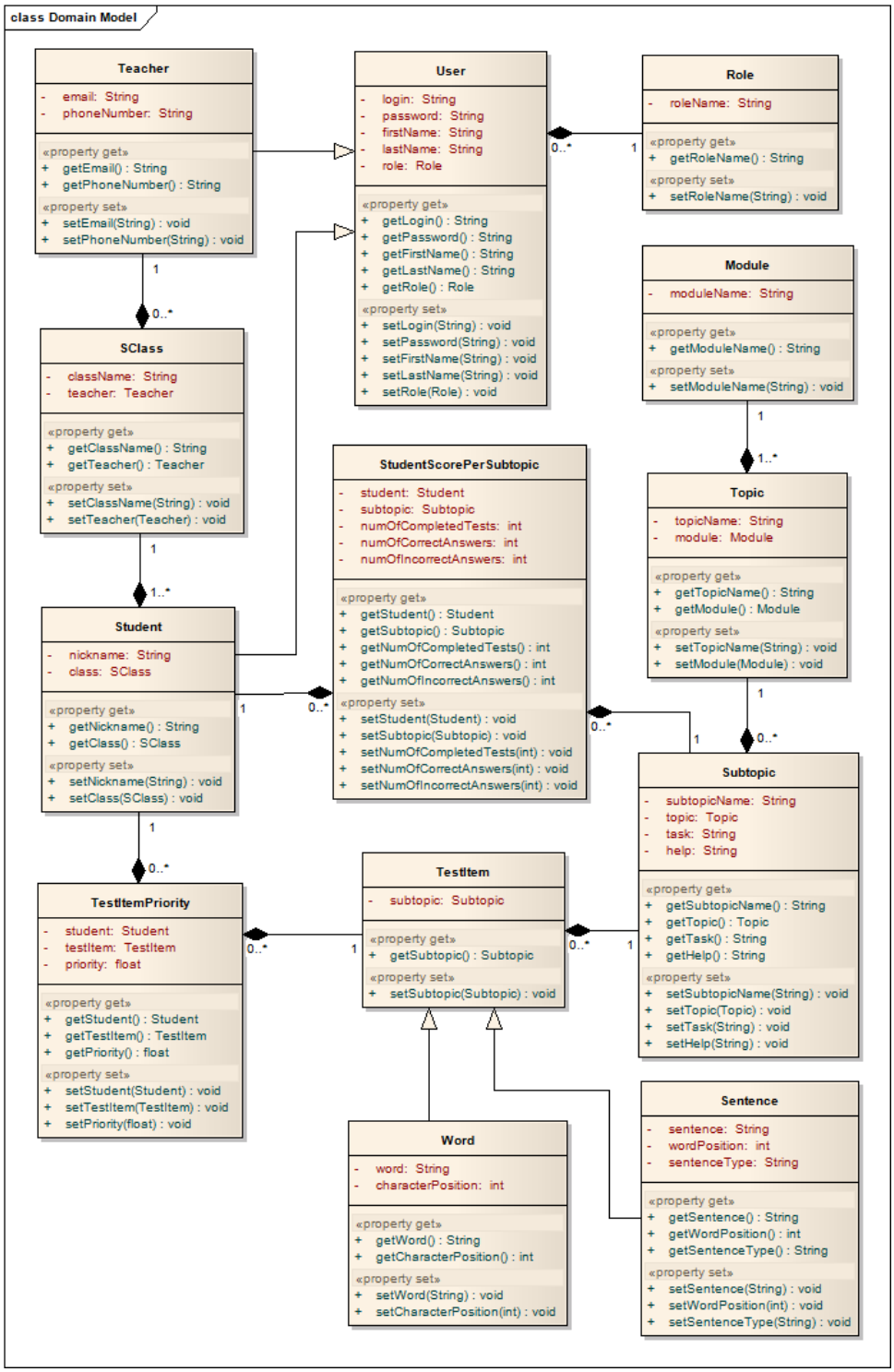
Na základě Use Case diagramu z předchozí kapitoly byly navržen následující model tříd. Nejdůležitějšími prvky tohoto modelu jsou `StudentScorePerSubtopic`, `TestItem` a `TestItemPriority`.

Třída `StudentScorePerSubtopic` bude uchovávat četnost procvičování žáka za jednotlivá subtémata. Právě z těchto hodnot budou později sestavovány žebříčky pro žáky a statistiky pro učitele.

Abstraktní třída `TestItem` v sobě bude nést informaci pouze o subtématu, do kterého patří. Z ní poté dědí jednotlivé třídy představující konkrétní testovou otázku – pro ukázku byly do modelu přidány pouze dvě implementace: `Word`, představující testovací otázku o jednom slově, do kterého bude doplňováno písmenko (například i / y) a `Sentence`, zastupující testovou otázku skládající se z věty, u které může být určován druh věty nebo doplňováno slovo (například spojka).

Třída `TestItemPriority` představuje prioritu zařazení testovací otázky do žákova procvičování. Její hodnota je vždy aktualizována na základě jeho odpovědi – pokud zodpověděl otázku správně, priorita přidělení se sníží, v opačném případě bude její hodnota navýšena.

Implementace procesu procvičování látky a dalších částí aplikace bude podrobněji popsána v následující kapitole.



Obrázek 4 Model tříd

5 Implementace

Další fází vývoje po analýze a návrhu je implementace. V první části je popsána integrace zvolených technologií, které byly představeny v kapitole Technologie použité při vývoji. Následuje popis mapování entit a ukázka validace vstupních dat od uživatele. Závěr této kapitoly je věnován popisu implementace nejdůležitějších funkčních požadavků: Správa žáků, tříd a učitelů, Procvičování, Výběr testových otázek do procvičování a Statistika procvičování.

5. 1 Konfigurace zvolených technologií

V následujících podkapitolách následuje popis konfigurace technologií, jenž byly při použity při vývoji. Jednotlivé sekce jsou doplněny ukázkami zdrojového kódu.

5. 1. 1 Spring

Konfigurace Spring Frameworku a Spring MVC může být provedena buď na základě XML souborů nebo pomocí anotací přímo v Java třídách. V této práci byla zvolena druhá varianta – anotace.
Načtení konfigurační třídy aplikace zajišťuje třída `WebAppInitializer` dědící ze třídy `AbstractAnnotationConfigDispatcherServletInitializer`, jenž je vyobrazena níže.

```
public class WebAppInitializer extends
        AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        return new Class[]{WebAppConfiguration.class};
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return null;
    }

    @Override
    protected String[] getServletMappings() {
        return new String[]{"/"};
    }

}
```

`WebAppInitializer` provede načtení konfigurační třídy aplikace `WebAppConfiguration` a vytvoření objektu `DispatcherServlet`. `DispatcherServlet` přijímá požadavky od uživatele a pomocí objektu typu `HandlerMapping` je předává odpovídajícímu Controlleru ke zpracování.

Konfigurační třída `WebAppConfiguration` obsahuje konfiguraci webové aplikace a zároveň provádí načítání konfigurace frameworků Spring Security a Hibernate.

```
@Configuration
@EnableWebMvc
@ComponentScan(basePackages = "cz.nemecve.dp")
@Import({SecurityConfiguration.class, HibernateConfiguration.class})
public class WebAppConfiguration extends WebMvcConfigurerAdapter {
```

Anotace `@Configuration` vyjadřuje, že třída `WebAppConfiguration` obsahuje jednu nebo více metod s anotací `@Bean`, které vytváří beany, jenž budou spravovány kontejnerem.

`@EnableWebMvc` je ekvivalentem pro zápis v XML konfiguraci:

`<mvc:annotation-driven />`.

Anotace `@ComponentScan` udává, kde mají být hledány jednotlivé komponenty aplikace, které jsou označené anotací (například `@Controller`, `@Service`, `@Repository`). V XML deklaraci by odpovídající konfigurace byla provedena pomocí `<context:component-scan base-package="cz.nemecve.dp" />`.

Konfigurační třída `WebAppConfigration` zodpovídá za vytvoření beany `ViewResolver`, která definuje umístění a typ souborů prezentační vrstvy.

```
@Bean
public InternalResourceViewResolver viewResolver() {
    InternalResourceViewResolver viewResolver
            = new InternalResourceViewResolver();
    viewResolver.setViewClass(JstlView.class);
    viewResolver.setPrefix("/WEB-INF/views/");
    viewResolver.setSuffix(".jsp");
    return viewResolver;
}
```

V této práci jsou soubory prezentační vrstvy umístěny v adresáři WEB-INF/views/ a mají příponu jsp.

5. 1. 2 Spring Security

Konfigurace Spring Security může být stejně jako v předchozím případě buď pomocí souborů ve formátu XML, nebo anotacemi. Pro konfiguraci Spring Security byl opět zvolen druhý přístup a veškerá nastavení se nacházejí ve třídě `SecurityConfiguration`.

Nejdůležitější částí je konfigurace je metoda `configure()`, v níž je definována autorizační logika pro jednotlivé části aplikace.

Z ukázky níže byly pro přehlednost vynechány některé definice.

```
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.authorizeRequests()
            .antMatchers("/classes/**")
                .access("hasRole('TEACHER') or hasRole('ADMIN')")
            .antMatchers("/teachers/**")
                .access("hasRole('ADMIN')")
            ...
            .antMatchers("/home")
                .access("hasRole('STUDENT')")
            .antMatchers("/admin/**")
                .access("hasRole('TEACHER') or hasRole('ADMIN')")
            .and().formLogin()
                .loginPage("/login")
                .successHandler(authenticationSuccesHandler)
                .usernameParameter("login").passwordParameter("password")
            .and().csrf()
            .and().exceptionHandling().accessDeniedPage("/accessDenied");
}
```

Pomocí metody `access()` je nastavena množina uživatelských rolí, které jsou potřebné pro přístup do definované části aplikace, respektive k přístupu k dané URL.

Ověřování uživatele zajišťuje `AuthenticationManager`, který je nakonfigurován metodou `configureGlobal()`.

```
@Autowired
public void configureGlobal(AuthenticationManagerBuilder auth)
        throws Exception {
    auth.userDetailsService(userDetailsService);
    auth.authenticationProvider(authenticationProvider());
}
```

```
@Bean
public DaoAuthenticationProvider authenticationProvider() {
    DaoAuthenticationProvider authProvider
            = new DaoAuthenticationProvider();
    authProvider.setUserDetailsService(userDetailsService);
    authProvider.setPasswordEncoder(passwordEncoder());
    return authProvider;
}
```

`AuthenticationManager` obsahuje referenci na `UserAuthenticationService`, jenž je implementací rozhraní `UserDetailService`.

`UserAuthenticationService` provádí ověření uživatele na základě zadaného přihlašovacího jména a hesla. V případě úspěšné autentizace je uživatel na základě své roli, pod kterou vystupuje v aplikaci, přesměrován na stránku definovanou ve třídě `AuthenticationSuccessHandler`. Pokud uživatel není oprávněn k zobrazení obsahu požadované stránky, je přesměrován na stránku AccessDenied.

5. 1. 3 Hibernate

Hibernate může být rovněž nakonfigurován buď v XML souborech nebo přímo v Java třídě pomocí anotací. Stejně jako v předchozích případech i zde byla zvolena druhá varianta.

```
@Configuration
@EnableTransactionManagement
@ComponentScan(basePackages = "cz.nemecve.dp")
public class HibernateConfiguration {
```

Anotace `@Configuration` opět slouží k označení, že `HibernateConfiguration` je konfigurační třída, která obsahuje metody vytvářející beany (metoda je označena anotací `@Bean`).

`@ComponentScan` zase uvádí, kde mají být hledány jednotlivé komponenty.

Samotná konfigurace frameworku Hibernate se nachází níže.

```
@Bean(name = "sessionFactory")
public SessionFactory sessionFactory() {
    LocalSessionFactoryBuilder builder
            = new LocalSessionFactoryBuilder(dataSource());
    builder.scanPackages("cz.nemecve.dp.model")
            .addProperties(getHibernateProperties());
    return builder.buildSessionFactory();
}
```

```
@Bean(name = "dataSource")
public DataSource dataSource() {
    DriverManagerDataSource dataSource
            = new DriverManagerDataSource();
    dataSource.setDriverClassName("org.postgresql.Driver");
    dataSource.setUrl("jdbc:postgresql://localhost:5432/"
            + "dp?useUnicode=true&characterEncoding=utf-8");
    dataSource.setUsername("...");
    dataSource.setPassword("...");
    return dataSource;
}

private Properties getHibernateProperties() {
    Properties properties = new Properties();
    properties.put("hibernate.format_sql", "true");
    //properties.put("hibernate.show_sql", "true");
    properties.put("hibernate.dialect",
            "org.hibernate.dialect.PostgreSQLDialect");
    //properties.put("hibernate.hbm2ddl.auto", "create");
    return properties;
}
```

Při vytváření `SessionFactory` je nutné udat, ve kterém balíčku se nacházejí doménové třídy (entity) a předat jí datový zdroj `DataSource` a Hibernate `Properties`.

V metodě `dataSource()` je nastaven ovladač, adresa databáze a přihlašovací údaje. Konkrétní hodnoty nastavení pro Hibernate jsou předány metodou `getHibernateProperties()`, které obsahují zvolený dialekt pro generování dotazů, nastavení vypisování dotazů prováděných Hibernatem do konzole.

Hibernate také umožňuje automatické smazání a znovu vytvoření celého databázového schématu nebo aktualizaci schématu, pokud detekuje provedené změny v entitách. Tohoto chování lze docílit nastavením:

```
properties.put("hibernate.hbm2ddl.auto", "create");
```

Hodnota `create` provede smazání a vytvoření databáze, hodnota `update` pouze aktualizuje schéma na základě provedených změn a původní tabulky s daty zůstanou zachovány.

5. 1. 4 Apache Tiles

Konfigurace Apache Tiles je provedena v konfigurační třídě aplikace `WebAppConfigation`.

Při vytváření beany metodou `tilesConfiguration()` je nutné zadat cestu k XML souboru, který obsahuje definice šablon použitých v aplikaci.

```
@Bean
public TilesConfigurer tilesConfigurer() {
    TilesConfigurer configurer = new TilesConfigurer();
    configurer.setDefinitions("/WEB-INF/tiles/tiles-definitions.xml");
    configurer.setPreparerFactoryClass(SpringBeanPreparerFactory.class);
    return configurer;
}
```

Samotné používání šablon definovaných pomocí Apache Tiles je popsáno v kapitole Uživatelské rozhraní.

5. 2 Mapování entit

Entita je třída, která představuje doménový objekt označený anotací `@Entity`. Mapování entit může být prováděno pomocí mapovacích souborů nebo využitím anotací uvnitř zdrojového kódu. V této práci byl použit druhý způsob – mapování entit je prováděno pomocí anotací ze specifikace JPA.

V ukázce níže se nachází část definice entity `Student`, která reprezentuje žáka.

```
@Entity
@Table(name = "student")
public class Student extends User implements Serializable {

    @NotNull
    @Size(min = 2, max = 30)
    @Column(name = "nick_name", nullable = false)
    private String nickName;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "sclass_id")
    private SClass sClass;
```

Anotace `@Entity` deklaruje, že třída `Student` je persistentní třída, jenž má být namapována. K určení tabulky, na kterou má být entita namapována, se používá anotace `@Table`.

Pomocí `@Column` je určen název sloupce odpovídající tabulky v databázi. Tato anotace také umožňuje zadat, zda tento sloupec může obsahovat `null`

hodnoty, minimální a maximální hodnoty, kterých může nabývat a výchozí hodnotu, pokud žádná nebyla vyplněna.

`@ManyToOne` reprezentuje vazbu n : 1. Existují ještě dvě další anotace pro vazby: pro označení vazby 1 : n slouží anotace `@OneToMany`, pro vazbu m : n pak anotace `@ManyToMany` a pro určení vazby 1 : 1 pak `@OneToOne`.

Data v tabulce databáze jsou rozlišována na základě svého jedinečného identifikátoru, proto i persistentní třídy musí mít deklarován atribut `id`. Identifikátor třídy `Student` se nachází v abstraktní třídě `User`, ze které dědí.

```
@Entity
@Table(name = "user")
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public abstract class User implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.TABLE)
    @Column(name = "id")
    private Long id;
```

K označení atributu, že se jedná o identifikátor, slouží anotace `@Id`. Pomocí `@GeneratedValue` a zvolené strategie je zajištěno automatické generování jedinečného identifikátoru.

Vzhledem k tomu, že `User` je abstraktní třída, ze které pak dědí třídy `Student` a `Teacher`, bylo nutné nastavit pomocí anotace `@Inheritence`, jaká strategie bude použita při práci s těmito entitami. Zvolená strategie `TABLE_PER_CLASS` vytvoří pro každou entitu, která dědí od třídy `User`, vlastní taulku.

5. 3 Validace vstupních dat

Spring MVC umožňuje jednoduchou implementaci validace vstupních dat. Kontrola hodnot vstupních dat je zajištěna pomocí anotacemi nad příslušným atributem objektu.

Validace atributu může vypadat následovně.

```
@NotNull
@Size(min = 5, max = 30,
      message = "Délka přihlašovacího jména musí být mezi 5 a 30 znaky.")
@Column(name = "login", nullable = false)
private String login;
```

Anotace `@NotNull` slouží k nastavení, že daný atribut nemůže být `null`.
`@Size` zajišťuje, že délka přihlašovacího jména musí být rovna minimálně 5 znakům, nejvýše však 30. Jestliže uživatel zadá hodnoty mimo tuto mez, bude mu vypsána odpovídající chybová hláška, která je definována v parametru `message`.

5. 4 Popis implementace vybraných požadavků

V následujících podkapitolách se je popsána implementace nejdůležitějších požadavků, jenž byly definovány v kapitole Funkční požadavky.

5. 4. 1 Správa žáků, tříd a učitelů

Správa žáků, tříd a učitelů probíhá na stejném principu, proto bude v následujících podkapitolách popsána pouze postup implementace žáků.
Správa žáků spočívá v možnosti přidávat nové žáky do databáze, zobrazení informací o nich a v případě potřeby aktualizovat jejich údaje nebo smazat jejich profil. Jinými slovy se jedná o implementaci funkcí pro vykonávání CRUD operací nad žáky.

5. 4. 1. 1 Zaslání požadavku

Požadavky uživatele o data nebo zpracování dat přijímá `DispatcherServlet`, jenž za pomoci objektu třídy `HandlerMapping` předá požadavek odpovídajícímu Controlleru. Požadavek vždy obsahuje URL adresu, v případě požadavku na zpracování dat může obsahovat navíc i například formulář vyplněný uživatelem.

5. 4. 1. 2 Zpracování požadavku

Zpracování požadavků od uživatele provádí komponenta Controller, která je opatřena odpovídající anotací `@Controller`. Požadavky na správu žáků vyřizuje `StudentController`.

```
@Controller
public class StudentController {

    @Autowired
    private StudentService studentService;

    @Autowired
    private SClassService sClassService;
```

`@Controller` vyjadřuje, že `StudenController` je Controller vyřizující požadavky na získání či zpracování dat.

Anotace `@Autowired` nad `StudentService` a `SClassService` slouží k označení, že má být injeknuta reference na tyto Service do `StudentControlleru`.

`StudentController` obsahuje metody pro zpracovávání požadavků na přidávání nových žáků, aktualizaci jejich profilu nebo smazání profilu a pro zobrazení všech žáků dané třídy. V následující ukázce je vyobrazena implementace zpracování požadavku na seznam žáků vybrané třídy.

```
@RequestMapping(value = "/classes/students/allStudents/{id}",
                method = RequestMethod.GET)
public ModelAndView allStudentsPage(@PathVariable Long id) { // class id
    ModelAndView model = new ModelAndView();
    List<Student> studentList = studentService.getAllStudentsBySClass(id);
    model.addObject("studentList", studentList);
    model.setViewName("classes/students/allStudents");
    return model;
}
```

Metoda `allStudentPages()` načte všechny žáky vybrané třídy pomocí Service `StudentService` a ty prostřednictvím `ModelAndView` předá zpátky `DispatcherServletu`. `DispatcherServlet` poté za pomoci `ViewResolver` zobrazí odpověď Controlleru uživateli.

5. 4. 1. 3 Získávání a zpracování dat

O získávání a zpracování dat se stará odpovídající Service, jenž má referenci na Dao třídu. Pro správu žáku byla implementována `StudentService`, která předává požadavek na data či zpracování dat `StudentDao`.

```
@Service
@Transactional
public class StudentServiceImpl implements StudentService {

    @Autowired
    private StudentDao studentDao;

    @Override
    public void add(Student s) {
        studentDao.add(s);
    }

    @Override
    public void update(Student s) {
        studentDao.update(s);
    }
```

```
@Override
public Student getStudentById(Long id) {
    return studentDao.getStudentById(id);
}

@Override
public List<Student> getAllStudentsBySClass(Long id) {
    return studentDao.getAllStudentsBySClass(id);
}
```

V ukázce výše se nachází část implementace `StudentService`. Kromě výše zobrazených metod na CRUD operace nad žáky a zobrazení všech žáků vybrané třídy umožňuje tato Service i například hledání žáka na základě jeho loginu nebo načtení žáka včetně jeho statistik procvičování za jednotlivá podtémata.

Třída `StudentDao` reprezentuje `Repository`, což je třída sloužící ke komunikaci s databází, opatřena anotací `@Repository`.

Implementace metody `getAllStudentBySClass()`, jenž je volána na `StudentService` na základě požadavku od Controlleru `StudentController`, vypadá následovně.

```
@Override
public List<Student> getAllStudentsBySClass(Long id) {
    Query query = getSession()
                    .createQuery("from Student where sclass_id = :id");
    query.setParameter("id", id);
    return (List<Student>) query.list();
}
```

Data jsou získána na základě HQL dotazu. V nejjednodušší podobě by měl dotaz podobu `from Student`, jelikož ale nebyl požadavek na všechny žáky, kteří se v tuto chvíli nacházejí v databázi, nýbrž jen na žáky konkrétní třídy, bylo nutno doplnit podmínku, že žák navštěvuje třídu, jejíž `id` je stejné jako `id` uživatelem vybrané třídy.

5. 4. 1. 4 Zobrazení výsledku

Výpis p o seznam žáků vybrané třídy, pak vypadá následovně.

```
<table class="table table-hover">

    <tr>
        <th>Příjmení</th>
        <th>Křestní jméno</th>
        <th>Přezdívka</th>
        <th>Akce</th>
    </tr>

    <c:forEach items="${studentList}" var="student">
        <tr>
            <td><c:out value="${student.lastName}"/></td>
            <td><c:out value="${student.firstName}"/></td>
            <td><c:out value="${student.nickName}"/></td>
            ...
        </tr>
    </c:forEach>

</table>
```

5. 4. 2 Procvičování

Výběrem podtématu, které chce žák procvičovat, je odeslán požadavek na `SubtopicTestController`. Controller zpracuje a na základě vybraného subtématu si řekne `TestService` o výběr testových otázek. Otázky pro procvičování si získá `TestService` od `TestDao`, jenž vybere z databáze deset testových otázek na základě priority přídělení.

Testovou otázku v aplikaci reprezentuje entita `TestItem`, ze které dědí jednotlivé implementace otázek.

```java
@Entity
@Table(name = "test_item")
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public abstract class TestItem implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.TABLE)
    @Column(name = "id")
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "subtopic_id")
    private Subtopic subtopic;

    @OneToMany(mappedBy = "testItem", fetch = FetchType.LAZY)
    private Collection<TestItemPriority> testItemPriorityList;
```

Implementací testové otázky je například entita `Word`, jenž reprezentuje slovo, do kterého má být doplňováno písmenko (i/y, š/ž, apod.) Další implementací je `Sentence`, která zastupuje testovou otázku typu věta, u které může být určován druh věty, znaménko na konci, nebo doplňováno slovo (například spojka).

Zobrazení testu se přizpůsobuje podle toho, jaký typ odpovědi má dané podtéma. Ukázky toho, jak daný test vypadá, se nachází v kapitole Shrnutí výsledků a v přílohách.

Po vyplnění odpovědi na testovou otázku je okamžitě provedena kontrola a aktualizována priorita přidělení dané otázky. Žák poté vidí, jestli na otázku zodpověděl správně či nikoliv, v případě nesprávné odpovědi uvidí, jak měla správně vypadat.

Po zodpovězení všech je žákovi zobrazen přehled, jak si v daném procvičování vedl. Hodnocení obsahuje počet bodů, které získal a slovní ohodnocení, které se liší podle dosaženého skóre – například v případě malého počtu správně zodpovězených otázek je žák vybídnut k dalšímu procvičování.

5. 4. 3 Výběr testových otázek do procvičování

Výběr testových otázek do procvičování je prováděn na základě priority přidělení otázky, která je uchovávána entitou `TestItemPriority`.

```
@Entity
@Table(name = "test_item_priority")
public class TestItemPriority {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "priority_id")
    private Long id;

    @Column(name = "priority", nullable = false)
    private float priority;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "student_id")
    private Student student;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id")
    private TestItem testItem;
```

Hodnota priority přidělení testové otázky se pohybuje v rozmezí <0,1>. Na začátku mají všechny testové otázky pro žáka prioritu 1, tedy nejvyšší.

V okamžiku vyhodnocení žákovy odpovědi v procvičování je automaticky aktualizována hodnota priority v databázi a to následovně:

```
public static float calculate(float priority, boolean isCorrect) {
    if (isCorrect) {
        return decreasePriority(priority);
    } else {
        return increasePriority(priority);
    }
}
```

Jestliže žák zodpověděl testovou otázku správně, je hodnota priority snížena, v opačném případě zvýšena. Cílem je, že postupným procvičováním budou žákovi vybírány ty testové otázky, které mu v minulosti dělaly největší potíže.

5. 4. 4 Statistika procvičování

Statistika procvičování žáka je vedena pomocí třídy `StudentScorePerSubtopic`. Tato entita uchovává informace o počtu správných a nesprávných odpovědí za vybrané podtéma a počtu cvičení, které žák dokončil.

```
@Entity
@Table(name = "student_score_per_subtopic")
public class StudentScorePerSubtopic implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "student_score_id")
    private Long id;

    @Column(name = "num_of_completed_tests", nullable = false)
    private int numOfCompletedTests;

    @Column(name = "num_of_correct_answers", nullable = false)
    private int numOfCorrectAnswers;

    @Column(name = "num_of_incorrect_answers", nullable = false)
    private int numOfIncorrectAnswers;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "student_id")
    private Student student;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "subtopic_id")
    private Subtopic subtopic;
```

Na základě těchto hodnot jsou pro žáky a učitele sestavovány statistiky a žebříčky. Statistiky žák vidí v přehledu všech témat (včetně podtémat) vybraného předmětu, žebříčky si může zobrazit samostatně pro každý předmět, kde uvidí deset nejpilnějších žáků podle počtu dokončených cvičení a na základě poměru správných a nesprávných odpovědí. Ukázka statistiky procvičování a žebříčků žáků se nacházejí v kapitole Shrnutí výsledků a v přílohách.

5. 5 Uživatelské rozhraní

Pro uživatelské rozhraní se používají Apache Tiles. Konfigurace tohoto nástroje byla popsána výše, nyní bude pozornost věnována samotnému vytváření šablon a jejich použití.

V konfiguračním XML souboru `tiles-definitions.xml` se nacházejí definice všech šablon. Na ukázce níže se nachází definice šablony, která kromě samotného obsahu na stránce zobrazí i menu, pravý box a patičku.

```
<definition name="defaultThinContentTemplate"

 template="/WEB-INF/templates/template/defaultThinContentTemplate.jsp">

<put-attribute name="menu" value="/WEB-INF/templates/menu.jsp" />
<put-attribute name="rightBox" value="/WEB-INF/templates/rightBox.jsp" />

    <put-attribute name="footer" value="/WEB-INF/templates/footer.jsp" />
</definition>
```

V tagu `definition` je zadán název šablony, na kterou se pak bude na stránce odkazovat a cesta k této šabloně.

Jestliže šablona obsahuje více komponent, které mohou být součástí i jiných šablon, jsou jí tyto komponenty předány tagy `put-attribue`, v nichž je zadán název této komponenty a cesta k ní.

5. 5. 1 Tvorba šablony

Šablona je jsp stránka, která obsahuje tagy pro označení míst, kam mají být vloženy jednotlivé komponenty view. Tato místa, kam bude vložen text nebo celá komponenta ve formátu jsp, jsou určena pomocí tagu `insertAttribute`.

```
<title><tiles:insertAttribute name="title" /></title>
```

V předchozí sekci byla ukázána definice šablony, která mimo obsahu stránky přidává i menu, pravý box a patičku. Tělo této šablony je vidět v následující ukázce.

```
<body>

    <tiles:insertAttribute name="menu" />

    <div id="container">
        <tiles:insertAttribute name="rightBox" />
        <tiles:insertAttribute name="content" />
    </div>

    <tiles:insertAttribute name="footer" />

</body>
```

Apache Tiles pomocí názvů komponent, které se musí shodovat s názvy uvedenými v `tiles-definitions.xml`, zajistí vložení těchto částí a složení výsledné stránky uživateli.

5. 5. 2 Použití šablony

Definice šablony, kterou používá konkrétní view, jenž je zobrazeno uživateli, je provedena následujícím způsobem.

```
<tiles:insertDefinition name="defaultWideContentTemplate">

    <tiles:putAttribute name="title" value="E-školička | Výpis žáků" />

    <tiles:putAttribute name="content">
```

V prvním kroku je nutné říct frameworku Apache Tiles, kterou šablonu má pro dané view použít. Zadaný název v tagu `insertDefinition` musí odpovídat názvu šablony v tiles-definitions.xml.

Poté již k samotnému vkládání textu nebo dalších view komponent stačí použít tag `putAttribute`, který na základě zadaného jména atributu provede vložení hodnoty (popřípadě komponenty) na místo deklarované šablonou.

6. Uživatelská příručka

6. 1 Uživatelská příručka pro Učitele a Administrátory

6. 1. 1 Přihlášení, ohlášení

Uživatel se do aplikace přihlásí pomocí svých přihlašovacích údajů: přihlašovacího jména a hesla. Jestliže přihlášení proběhne úspěšně, je přesměrován na hlavní stranu administrace aplikace, v opačném případě se mu zobrazí zpráva o neúspěšném přihlášení.
Ohlášení z aplikace se provádí kliknutím na odkaz Odhlásit se, který se nachází v menu vpravo.

6. 1. 2 Správa učitelů (nutné oprávnění Administrátor)

Správa učitelů je dostupná pouze uživatelům, kteří disponují právy administrátora.

6. 1. 2. 1 Výpis všech učitelů

Zobrazit seznam všech učitelů může uživatel přímo z hlavní strany administrace odkazem Zobrazit všechny učitele, nebo kliknutím na odkaz Učitelé v menu.

6. 1. 2. 2. Přidání nového učitele

Přidat nového učitele může uživatel přímo z hlavní strany administrace pomocí odkazu Přidat nového učitele, nebo kliknutím na odkaz Přidat nového učitele ve výpisu všech učitelů.
Pro přidání nového učitele je nutné vyplnit všechna povinná pole a údaje uložit kliknutím na tlačítko Přidat.

6. 1. 2. 3. Úprava profilu učitele

Upravit profil požadovaného učitele může uživatel pomocí odkazu Upravit ve výpisu všech učitelů. Provedené změny je nutné uložit kliknutím na tlačítko Uložit změny.

6. 1. 2. 4. Odstranění profilu učitele

Odstranit profil požadovaného učitele může uživatel pomocí odkazu Odstranit ve výpisu všech učitelů. Pro odstranění profilu učitele je nutné tuto akci potvrdit kliknutím na tlačítko Vymazat.

6. 1. 3 Správa tříd

6. 1. 3. 1. Výpis všech tříd

Zobrazit seznam všech tříd může uživatel přímo z hlavní strany administrace odkazem Zobrazit všechny třídy, nebo kliknutím na odkaz Třídy v menu.

6. 1. 3. 2. Přidání nové třídy

Přidat novou třídu může uživatel přímo z hlavní strany administrace pomocí odkazu Přidat novou třídu, nebo kliknutím na odkaz Přidat třídu ve výpisu všech tříd.
Pro přidání nové třídy je nutné vyplnit všechna povinná pole a údaje uložit kliknutím na tlačítko Přidat.

6. 1. 3. 3. Úprava záznamu třídy

Upravit záznam požadované třídy může uživatel pomocí odkazu Upravit ve výpisu všech tříd. Provedené změny je nutné uložit kliknutím na tlačítko Uložit změny.

6. 1. 3. 4. Odstranění záznamu třídy (nutné oprávnění Administrátor)

Tato akce je dostupná pouze uživatelům, kteří disponují právy administrátora. Odstranit záznam požadované třídy (včetně všech jejích žáků) může uživatel pomocí odkazu Odstranit ve výpisu všech tříd. Pro odstranění záznamu třídy je nutné tuto akci potvrdit kliknutím na tlačítko Vymazat.

6. 1. 4 Správa žáků

6. 1. 4. 1. Výpis žáků zvolené třídy

Zobrazit seznam žáků zvolené třídy může uživatel pomocí odkazu Seznam žáků ve výpisu tříd.

6. 1. 4. 2. Přidání nového žáka

Přidat nového žáka do třídy může uživatel pomocí odkazu Přidat nového žáka ve výpisu tříd. Pro přidání nového žáka je nutné vyplnit všechna povinná pole a údaje uložit kliknutím na tlačítko Přidat.

6. 1. 4. 3. Úprava profilu žáka

Upravit profil požadovaného žáka může uživatel pomocí odkazu Upravit ve výpisu žáků zvolené třídy. Provedené změny je nutné uložit kliknutím na tlačítko Uložit změny.

6. 1. 4. 4. Odstranění profilu žáka

Odstranit profil požadovaného žáka může uživatel pomocí odkazu Odstranit ve výpisu žáků zvolené třídy. Pro odstranění profilu žáka je nutné tuto akci potvrdit kliknutím na tlačítko Vymazat.

6. 1. 5 Výpis modulů

Zobrazit seznam všech modulů může uživatel přímo z hlavní strany administrace odkazem Zobrazit všechny moduly, nebo kliknutím na odkaz Moduly v menu.

6. 1. 6. Výpis témat (včetně podtémat) zvoleného modulu

Zobrazit seznam témat včetně podtémat, které dané téma obsahuje, může uživatel kliknutím na požadovaný modul ve výpisu všech modulů.

6. 1. 7 Výpis testových otázek podtématu

Zobrazit seznam testových otázek zvoleného podtématu může uživatel pomocí odkazu Zobrazit otázky ve výpisu témat.

6. 1. 8 Výpis problémových otázek podtématu

Zobrazit seznam problémových testových otázek zvoleného podtématu může uživatel pomocí odkazu Zobrazit problémové otázky ve výpisu témat.

6. 2 Uživatelská příručka pro Žáky

6. 2. 1 Přihlášení, ohlášení

Žák se do aplikace přihlásí pomocí svých přihlašovacích údajů: přihlašovacího jména a hesla. Jestliže přihlášení proběhne úspěšně, je přesměrován na svou domovskou stranu, v opačném případě se mu zobrazí zpráva o neúspěšném přihlášení.
Ohlášení z aplikace se provádí kliknutím na odkaz Odhlásit se, který se nachází v menu vpravo.

6. 2. 2 Výpis předmětů

Zobrazit seznam všech předmětů může žák přímo ze své domovské strany odkazem Procvičovat, nebo kliknutím na odkaz Procvičovat v menu.

6. 2. 3. Výpis témat (včetně podtémat) zvoleného předmětu

Zobrazit seznam témat včetně podtémat, které dané téma obsahuje, může žák kliknutím na požadovaný předmět ve výpisu všech předmětů.

6. 2. 4 Procvičování

Procvičování zvoleného podtématu spustí žák kliknutím na tlačítko Procvičovat ve výpisu všech témat. Po zodpovězení otázky si zkontroluje svou odpověď kliknutím na tlačítko Zkontrolovat, další otázku zobrazí kliknutím na tlačítko Další otázka. Po zodpovězení všech otázek mu bude automaticky zobrazen výsledek procvičování.

6. 2. 5 Zobrazení žebříčků

Zobrazit žebříčky nejpilnějších žáků může žák přímo ze své domovské strany odkazem Žebříčky, nebo kliknutím na odkaz Žebříčky v menu.

6. 2. 6 Úprava profilu

Upravit svůj profil může žák pomocí odkazu Můj profil v menu. Změnit své přezdívky provede vyplněním pole Přezdívka a kliknutím na tlačítko Změnit přezdívku, změnu svého obrázku uskuteční kliknutím na tlačítko Změnit obrázek a posléze kliknutím na požadovaný obrázek.

7 Shrnutí výsledků

Cílem práce bylo vytvořit webovou aplikaci na podporu výuky na 1. stupni základní školy, jež měla být jednoduchá, přehledná a pro uživatele srozumitelná. Na začátku byly na základě analýzy zadání projektu vymezeny požadavky, které měla výsledná aplikace splňovat.

Aktuální podoba webové aplikace E-školička splňuje všechny požadavky s prioritou 1 a 2, jež byly stanoveny v kapitole Specifikace požadavků. Aplikace umožňuje kompletní správu žáků, tříd a učitelů. Žáci mohou libovolně procvičovat jednotlivá podtémata předmětů, které mají dostupné – v aktuální verzi se v aplikaci nachází pouze kurz Český jazyk pro 2. třídu.

Žáci mají dostupné podrobné statistiky procvičování pro každé podtéma, pomocí kterých vidí, jak si v daných oblastech vedou.

Vzhled aplikace byl vytvořen autorem diplomové práce. Při návrhu uživatelského rozhraní bylo dbáno na jednoduchost a přehlednost, aby se uživatel dokázal ve webové aplikaci rychle zorientovat a uměl ji intuitivně používat.

Výsledná aplikace je dostupná na www.e-skolicka.cz. Přihlašovací údaje a uživatelská příručka se nacházejí na přiloženém CD.

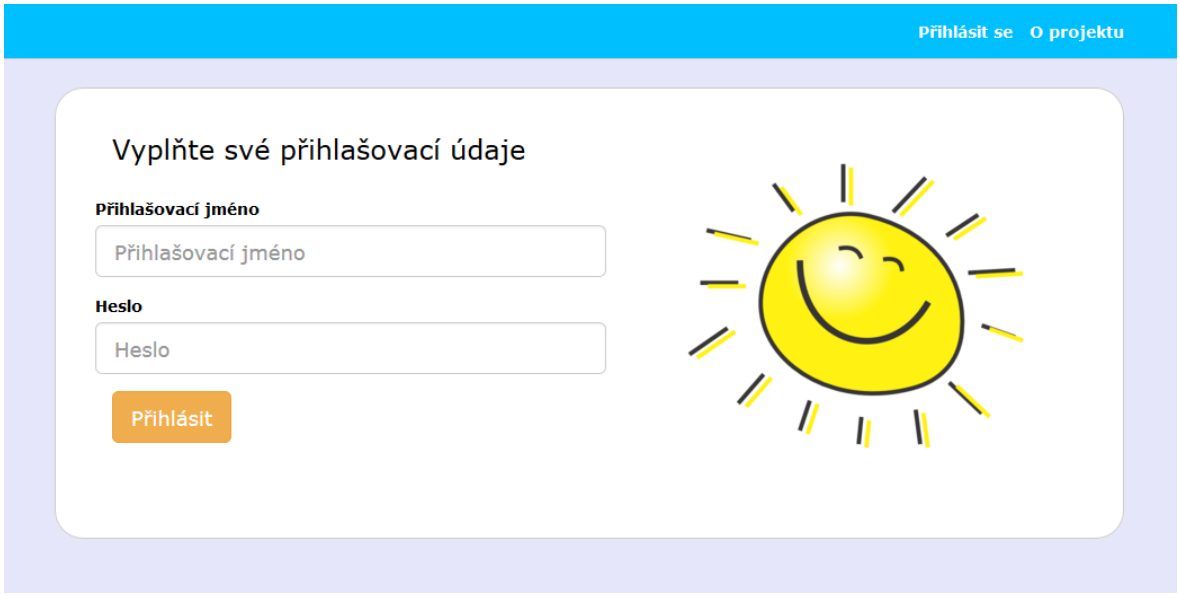
7. 1 Rozšíření do budoucna

Na základě podnětů ze stany vyučujícího, který webovou aplikaci E-školička využívá ve výuce českého jazyka ve 2. třídě základní školy, bylo navrhnuto následující rozšíření funkcionality:

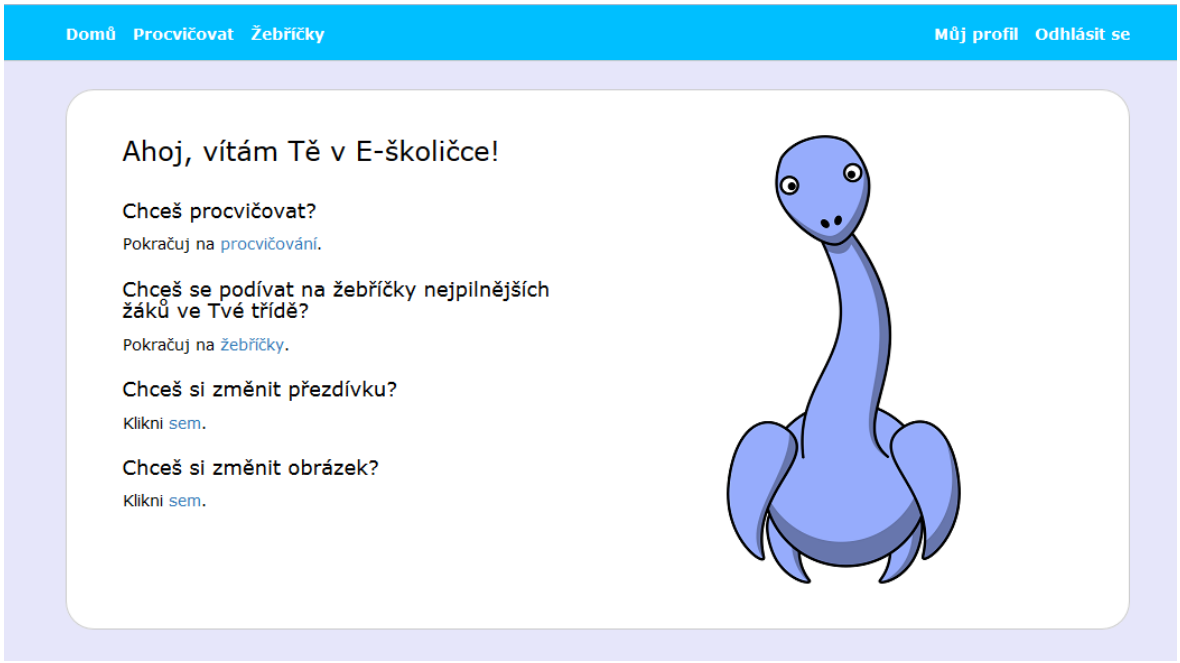
- Upozorňování žáků na dlouhou prodlevu od posledního procvičování konkrétního podtématu
- Souhrnné testy za celé téma
- Po určité době vyžadovat od žáka zopakování souhrnného testu
- Sledování aktivity žáka (frekvenci procvičování)
- Generování testů a pracovních listů pro tisk
- Souhrnné přidávání žáků do aplikace

Tyto funkční požadavky byly schváleny a jejich implementace je naplánována na červenec letošního roku (2016).

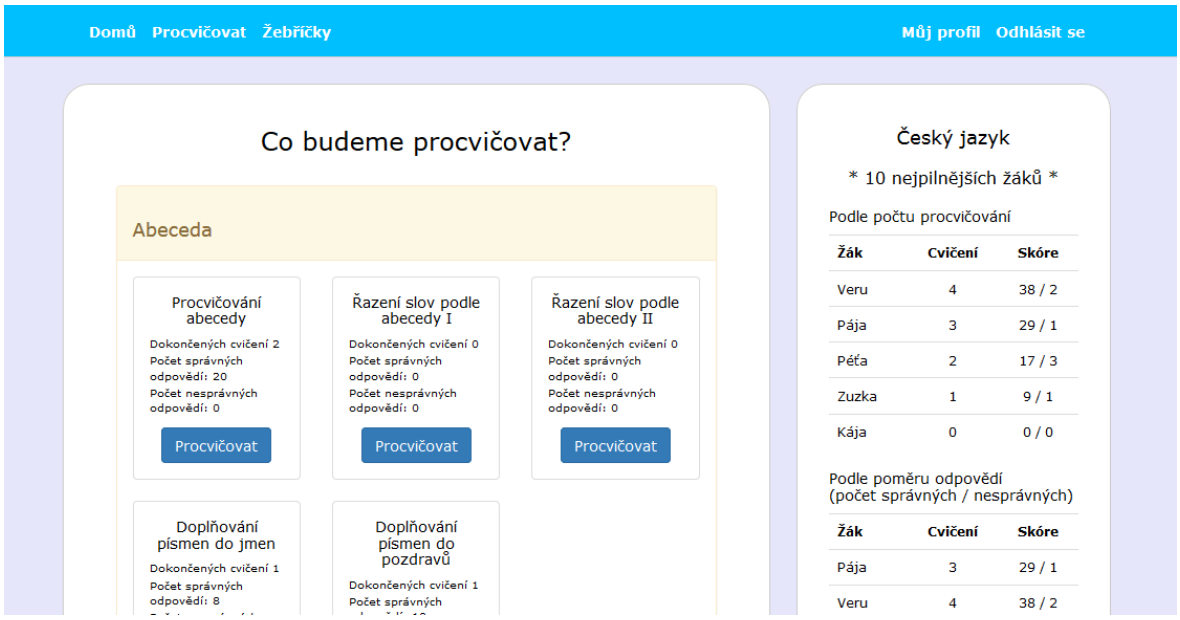
7. 2 Ukázky z aplikace E-školička



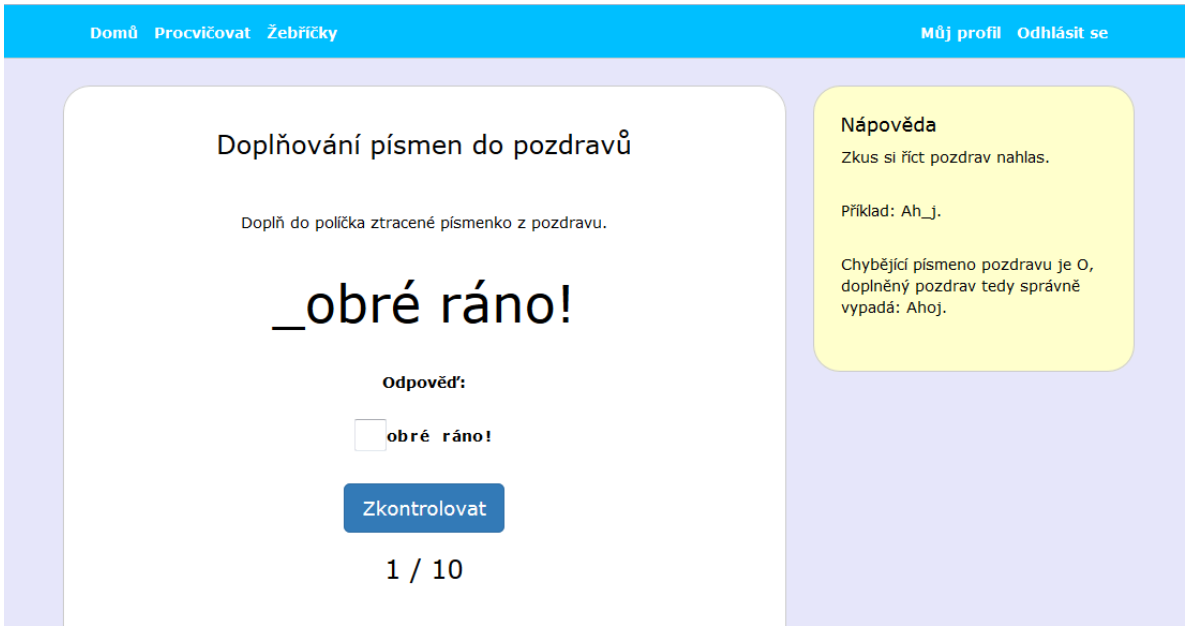
Obrázek 5 Úvodní stránka aplikace E-školička



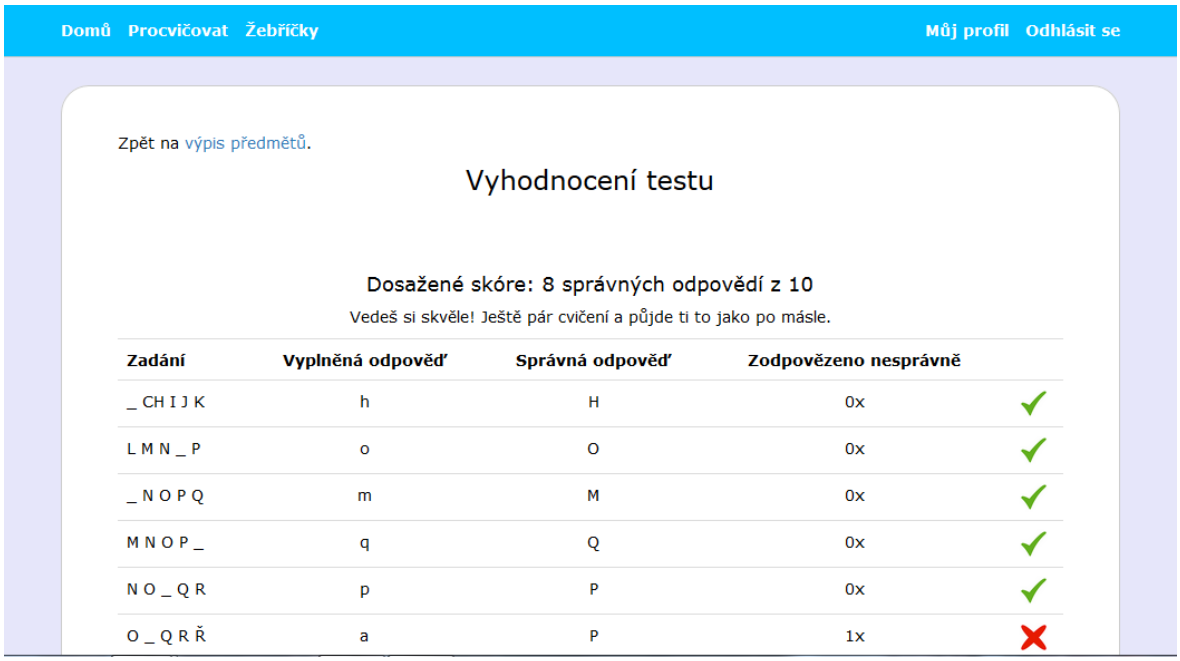
Obrázek 6 Domovská stránka pro žáky



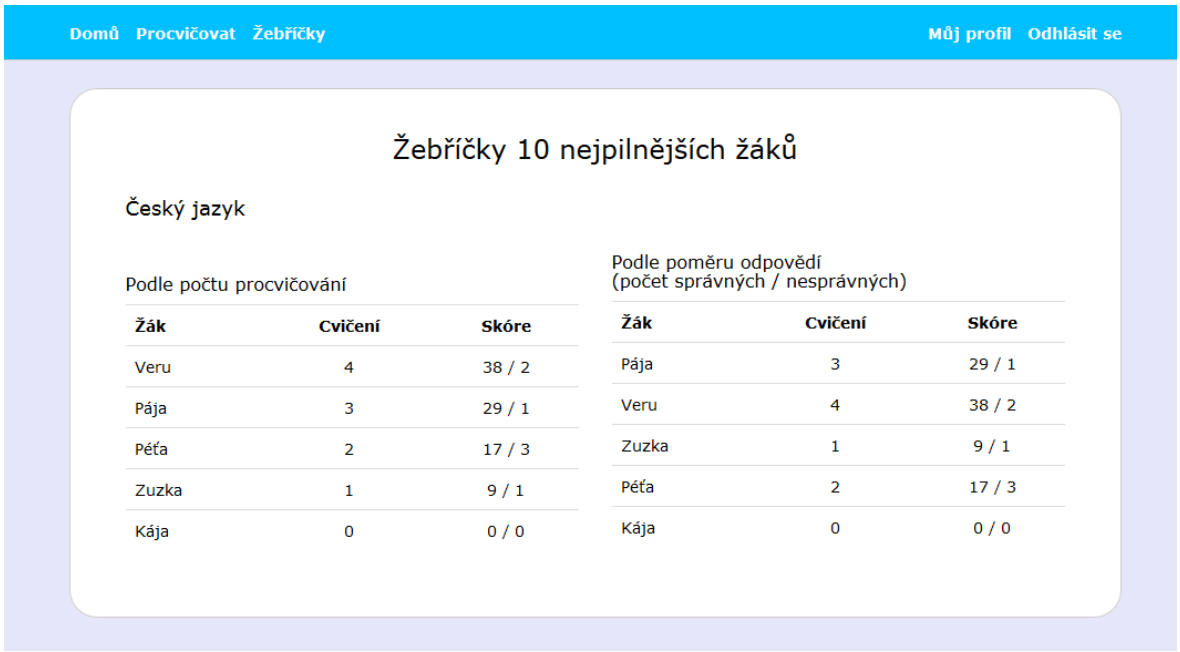
Obrázek 7 Přehled témat a podtémat k procvičování



Obrázek 8 Procvičování



Obrázek 9 Výsledky procvičování



Obrázek 10 Žebříčky nejpilnějších žáků

Závěr

Cílem diplomové práce bylo navrhnout a implementovat webovou aplikaci, která bude sloužit jako nástroj pro podporu výuky na 1. stupni základní školy. V rámci práce byla vytvořena webová aplikace E-školička, jejíž funkcionalita pokrývá stanovené požadavky, jak již bylo shrnuto v předchozí kapitole.

Aplikace E-školička byla v březnu tohoto roku (2016) nasazena na Základní škole v Kamenných Žehrovicích, kde je využívána ve 2. třídě k procvičování českého jazyka. Žákům bylo ukázáno, jak mají s aplikací pracovat a kde se nachází jednotlivé části: výpis předmětů, témat a podtémat, procvičování, žebříčky nejpilnějších žáků a profil žáka. Žáci 2. třídy se dokázali v aplikaci velmi rychle zorientovat, což bylo mimo jiné jedním z požadavků na vyvíjenou aplikaci.

Na základě podnětů od vyučujícího, který s aplikací E-školička přímo pracuje, byla definována další rozšíření do budoucna. Prvním z nich je, aby žáci byli vedeni k pravidelnému procvičování a opakování i starších témat. K tomuto účelu bude do aplikace zahrnuta logika, jež bude sledovat uplynulý čas od posledního procvičování každého podtématu – pokud bude prodleva delší než stanovený limit, žák bude vyzván k procvičování. Dalším požadavkem jsou souhrnné testy za celé téma a rovněž vyžadování po určité době souhrnný test opakovat. Posledním bodem, který bude také začleněn do aplikace, je funkcionalita pro generování testů a pracovních listů pro tisk. Posledním návrhem na vylepšení je možnost hromadného přidávání žáků do aplikace.

Cílem práce bylo vytvořit aplikaci, jež bude umožňovat žákům procvičovat látku probíranou ve škole a vyučujícím poskytne zpětnou vazbu o tom, jak si žáci v daném tématu vedou. Aplikace měla být jednoduchá, snadno ovladatelná a přehledná. Vzhledem k výsledné aplikaci a pozitivním ohlasům jak ze strany vyučujícího, tak od žáků 2. třídy Základní školy v Kamenných Žehrovicích lze usoudit, že cíl této diplomové práce byl splněn.

Seznam obrázků

Seznam použité literatury

[1] VANĚČEK, David. *Elektronické vzdělávání*. 1. vyd. Praha: České vysoké učení technické v Praze, 2011. ISBN 978-80-01-04952-5.

[2] EGEROVÁ, Dana. *Jak tvořit studijní opory pro e-learning: metodická příručka pro autory studijních opor*. 1. vyd. Plzeň: Západočeská univerzita v Plzni, 2011. ISBN 978-80-7043-982-1.

[3] KVĚTOŇ, Karel. *Základy e-learningu 2003*. Vyd. 1. Ostrava: Ostravská univerzita, 2004. Systém celoživotního vzdělávání Moravskoslezska. ISBN 80-704-2986-0.

[4] KOPECKÝ, Kamil. E-learning (nejen) pro pedagogy. 1. vyd. Olomouc: Hanex, 2006. Vzdělávání a informace. ISBN 80-857-8350-9.

[5] DRTINA, René. *Možnosti a omezení elektronické podpory kvality vzdělávání*. Vyd. 1. Praha: Extrasystem Praha, 2011. ISBN 978-80-87570-01-2.

[6] ZOUNEK, Jiří. *E-learning - jedna z podob učení v moderní společnosti*. Vyd. 1. Brno: Masarykova univerzita, 2009. ISBN 978-80-210-5123-2.

[7] KLEMENT, Milan. *E-learning: elektronické studijní opory a jejich hodnocení*. 1. vyd. Olomouc: Agentura Gevak, 2012. ISBN 978-80-86768-38-0.

[8] HAŠKOVÁ, Alena, Mária PISOŇOVÁ a Miriam BITTEROVÁ. *Didaktické prostriedky ako optimalizačný faktor procesu vzdelávania*. Vyd. 1. Hradec Králové: Gaudeamus, 2011. Recenzované monografie.
ISBN 978-80-7435-160-0.

[9] VOŽENÍLEK, Vít, Jiřina JÍLKOVÁ a Radim TOLASZ. *Klimatická změna v e-learningové výuce: východiska, stav, prototyp, nasazení*. 1. vyd. Olomouc: Univerzita Palackého v Olomouci, 2010. ISBN 978-80-244-2696-9.

[10] VANĚK, Jindřich. *E-learning, jedna z cest k moderním formám vzdělávání*. 1. vyd. V Karviné: Slezská univerzita v Opavě, Obchodně podnikatelská fakulta, 2008. ISBN 978-80-7248-471-3.

[11] VLČKOVÁ, Irena. *Nová média ve výuce: příručka ke kurzu "Využití počítače a internetu ve výuce"*. Vyd. 1. Liberec: Technická univerzita v Liberci, 2012. ISBN 978-80-7372-835-9.

[12] WALLS, Craig. a Ryan. BREIDENBACH. *Spring in action*. 2nd ed. Greenwich, CT: Manning Publications Co., c2008. ISBN 19-339-8813-4.

[13] *Spring Framework Reference Documentation* [online]. [cit. 2016-04-28]. Dostupné z: http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/

[14] *Spring Security Reference* [online]. [cit. 2016-04-28]. Dostupné z: http://docs.spring.io/spring-security/site/docs/4.0.4.RELEASE/reference/htmlsingle/

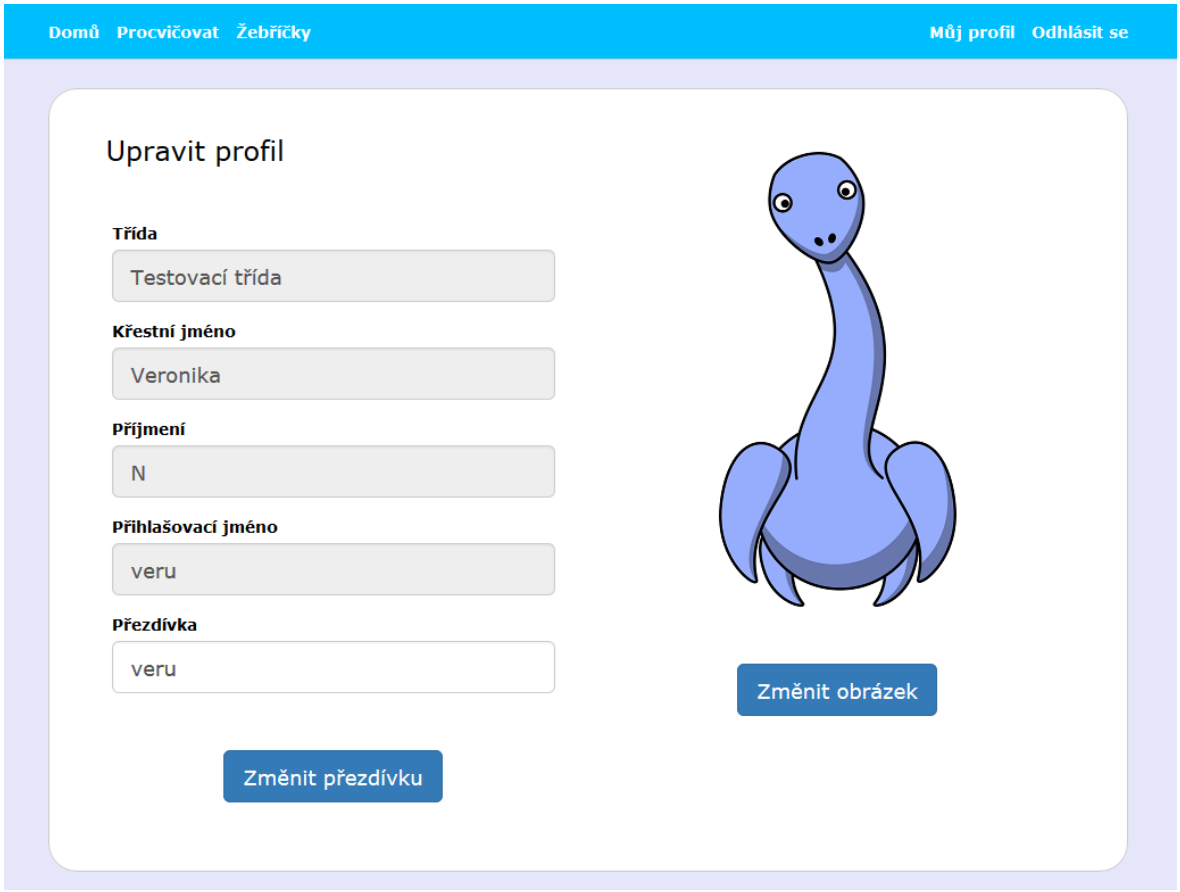
[15] *Hibernate ORM documentation* [online]. [cit. 2016-04-28]. Dostupné z: http://hibernate.org/orm/documentation/5.1/

[16] P*ostgreSQL: About* [online]. [cit. 2016-04-28]. Dostupné z: http://www.postgresql.org/about/

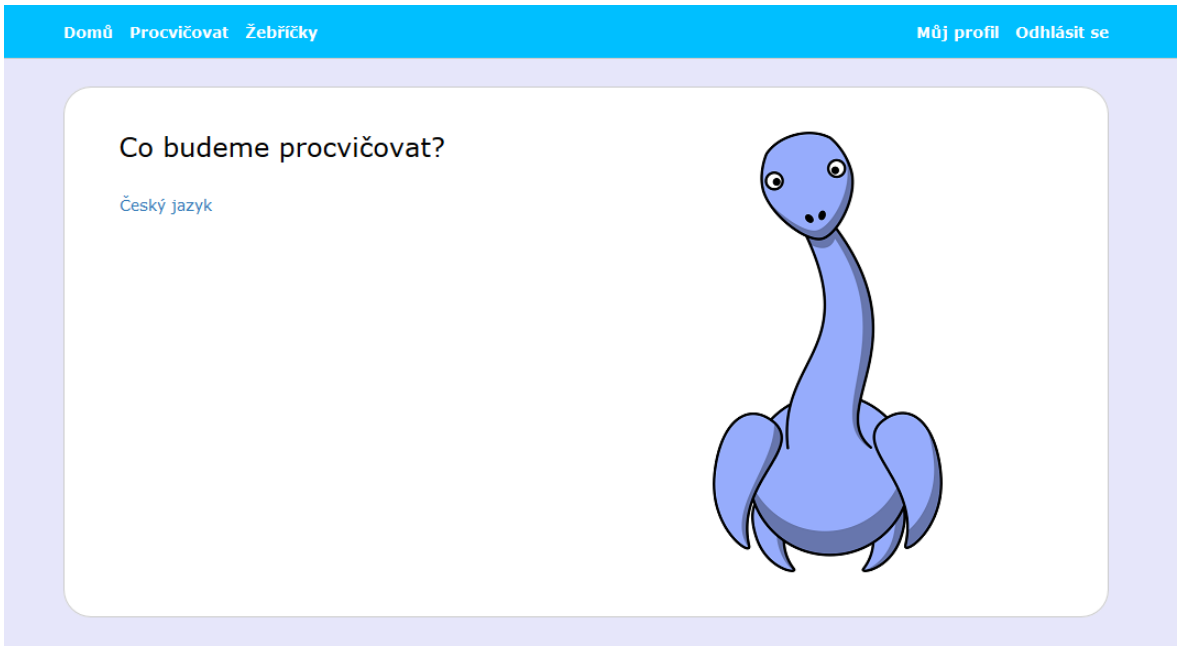
[17] *Apache Tiles* [online]. [cit. 2016-04-28]. Dostupné z: https://tiles.apache.org/framework/

[18] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2007. ISBN 978-80-251-1503-9.

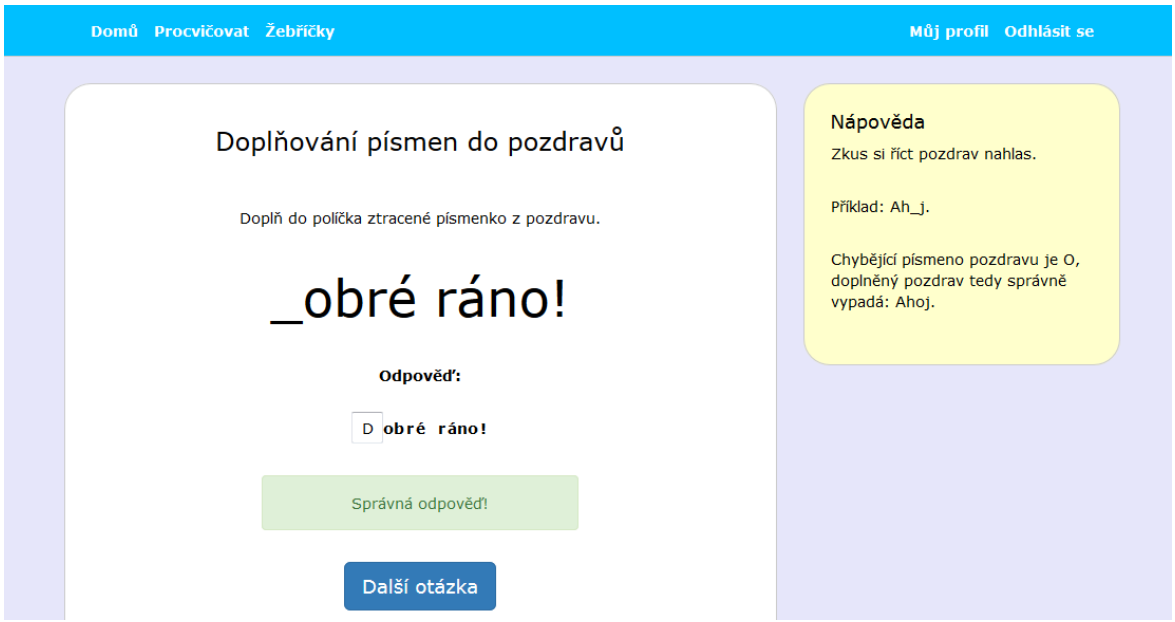
Přílohy



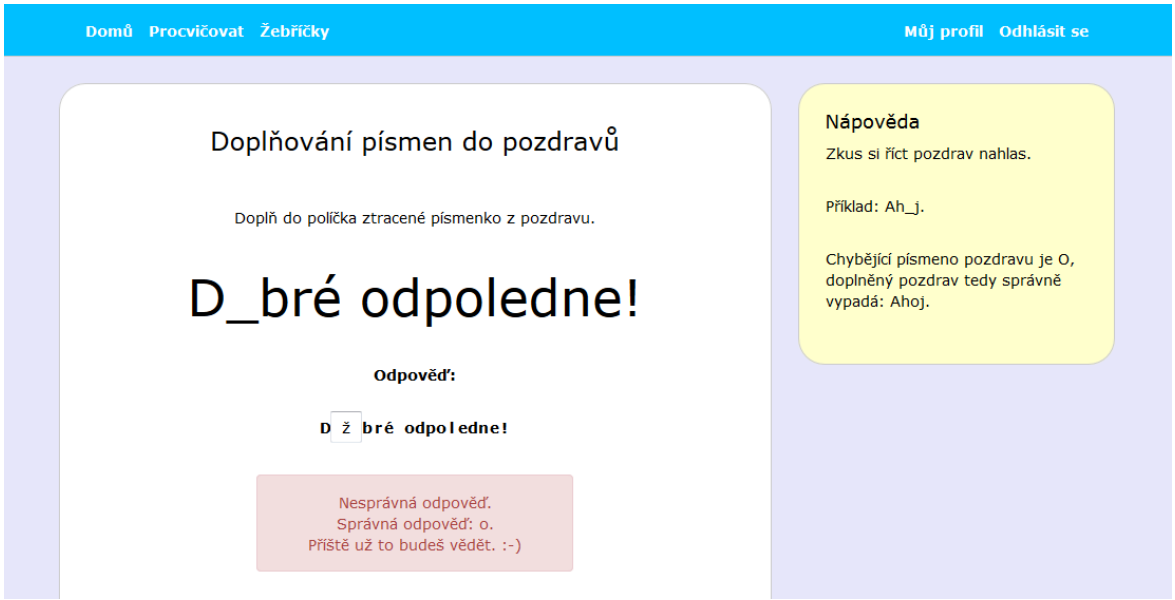
Obrázek 11 Úprava profilu žákem



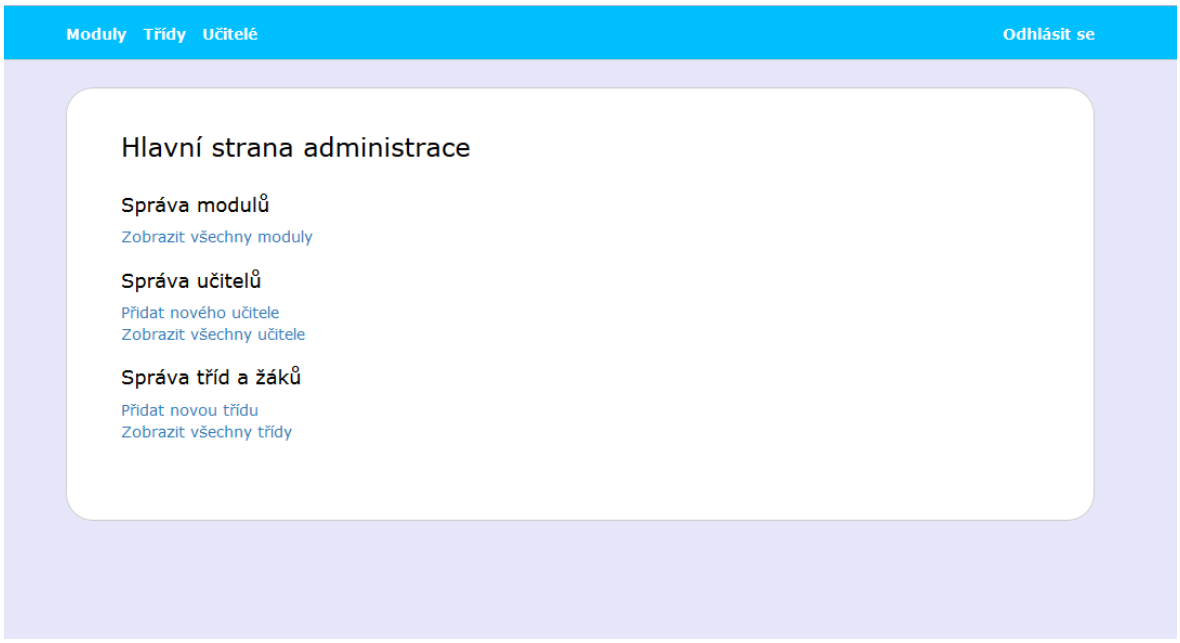
Obrázek 12 Výpis předmětů pro žáka



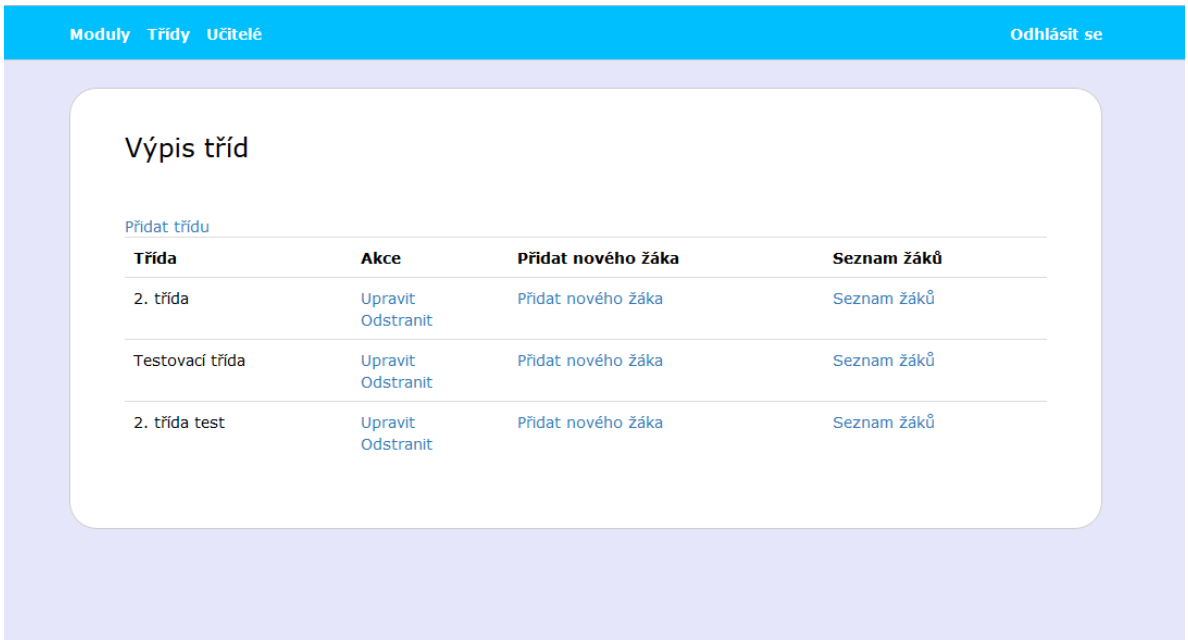
Obrázek 13 Procvičování - správná odpověď



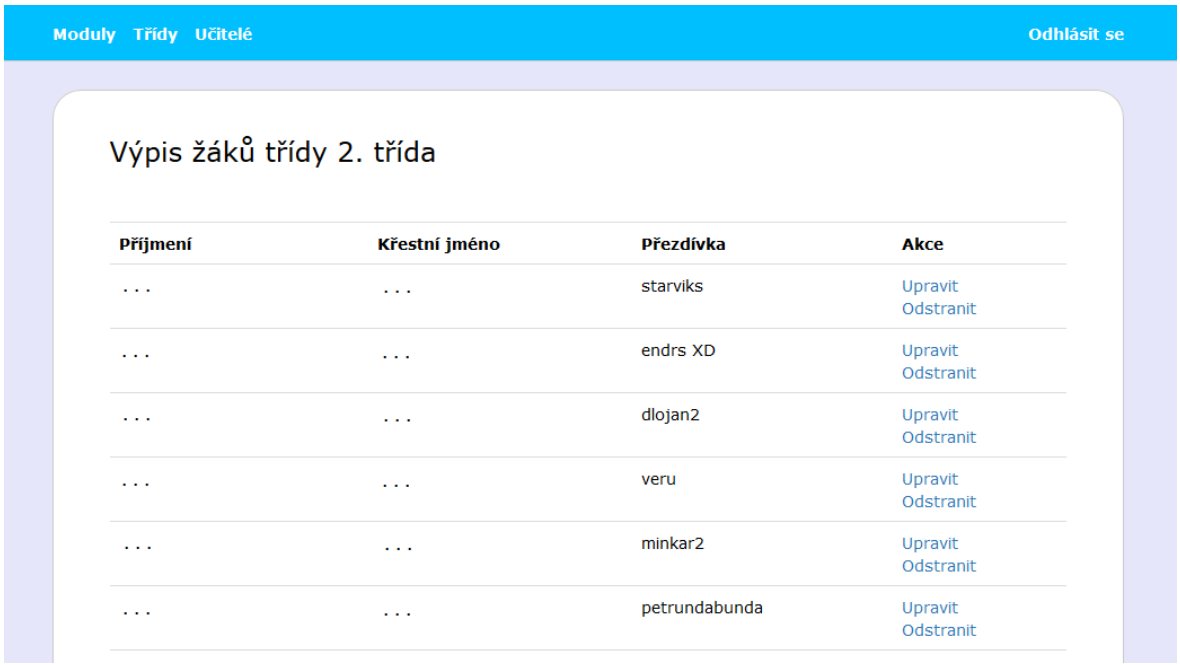
Obrázek 14 Procvičování - nesprávná odpověď



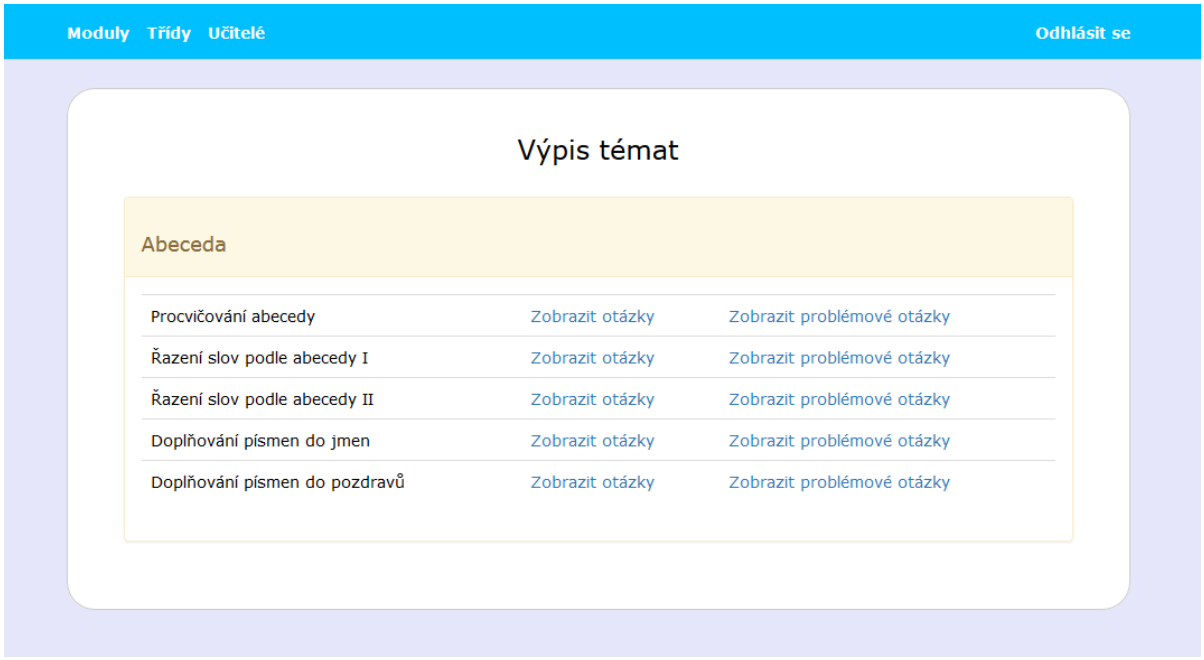
Obrázek 15 Hlavní strana administrace



Obrázek 16 Správa tříd



Obrázek 17 Správa žáků třídy



Obrázek 18 Výpis témat modulu

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2015/2016

Studijní program: Aplikovaná informatika
Forma: Prezenční
Obor/komb.: Aplikovaná informatika (ai2-p)

Podklad pro zadání DIPLOMOVÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Němečková Veronika	Pod Kopaninou 368, Lány	I1471

TÉMA ČESKY:

Webová aplikace pro podporu výuky na 1. stupni ZŠ

TÉMA ANGLICKY:

Web application for support of education at 1st grade of primary school.

VEDOUCÍ PRÁCE:

doc. RNDr. Petra Poulová, Ph.D. - KIKM

ZÁSADY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je vytvořit webovou aplikaci, která bude sloužit jako doplněk k výuce předmětů Český jazyk a Matematika. Aplikace bude umožňovat žákům procvičovat probranou látku, vyučujícím bude poskytovat zpětnou vazbu v podobě zhodnocení, které části dělají žákům největší potíže.

Osnova
1. Úvod
2. Elektronické vzdělávání
3. Popis problému
4. Analýza požadavků
5. Návrh
6. Implementace
7. Závěr

SEZNAM DOPORUČENÉ LITERATURY:

Podpis studenta: ..... **Datum:** 14.10.2015

Podpis vedoucího práce: ..... **Datum:** 14.10.2015

(c) IS/STAG , Portál - Podklad kvalifikační práce , I1471 , 14.10.2015 10:09