VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MATEMATIKA PRO ZŠ

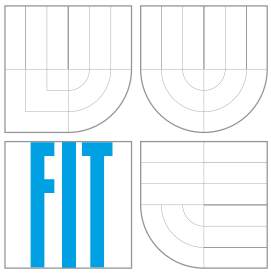
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAKUB ŽABA

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MATEMATIKA PRO ZŠ

MATHEMATICS FOR BASIC SCHOOLS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAKUB ŽABA
AUTHOR

VEDOUCÍ PRÁCE — Ing. IGOR SZŐKE, Ph.D.
SUPERVISOR

BRNO 2013

Abstrakt

Hlavním cílem této práce je uplatnit gamifikační prvky ve webové aplikaci, která umožňuje procvičování matemických příkladů pro základní školy. Aplikace je vytvořena jazykem PHP a SQL, a Nette Frameworkem. K procvičování příkladů je využit framework Khan academy. Aplikace je otestována s uživateli a v závěru je navrhnut další vývoj.

Abstract

Main goal of this theses is use gamification elements in web aplication, which allows exercise math problems for elementary schools. Aplication is created in PHP and SQL language and Nette Framework. There is used Khan academy framework to exercise problems. Aplication is tested with users and there is suggestion of next development at the end.

Klíčová slova

Vzdělávací program, PHP, Nette Framework, Khan academy, MVC

Keywords

Educational program, PHP, NET Framework, Khan Academy, MVC

Citace

Jakub Žaba: Matematika pro ZŠ, bakalářská práce, Brno, FIT VUT v Brně, 2013

Matematika pro ZŠ

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Igora Szőkeho, Ph.D.

.....
Jakub Žaba
15. května 2013

Poděkování

Mé poděkování patří Ing. Igoru Szőkemu, Ph.D. za poskytnuté rady při tvorbě této práce.

Obsah

1 Úvod **2**

2 Gamifikace **3**
2.1 Gamifikace . . . 3
2.2 Flow . . . 3
2.3 Motivace ve hrách . . . 4
2.4 Vizuální prvky gamifikace . . . 5
2.5 Gamifikované projekty . . . 6
2.6 Specifikace . . . 8

3 Návrh **11**
3.1 Případy užití . . . 11
3.2 E-R model . . . 12
3.3 Uživatelské rozhraní . . . 13

4 Výběr technologií **15**
4.1 Server . . . 15
4.1.1 Nette Framework . . . 15
4.1.2 MySQL . . . 16
4.2 Klient . . . 17
4.2.1 HTML a CSS . . . 17
4.2.2 Javascript . . . 17
4.2.3 Framework pro cvičení . . . 17

5 Implementace **21**
5.1 Adresářová struktura . . . 21
5.2 Registrace a přihlašování . . . 22
5.3 Spuštění kurzu . . . 23
5.4 Vyhodnocení úspěšnosti . . . 23
5.5 Změny v Khan academy frameworku . . . 25

6 Testování **27**
6.1 Debugování . . . 27
6.2 Testování s uživateli . . . 28

7 Závěr **30**
7.1 Směry dalšího vývoje . . . 30

A Obsah CD **33**

Kapitola 1

Úvod

Výukové počítačové programy pro základní školy doplňují školní výuku několika způsoby. Mohou pomoci při výkladu látky názornou ukázkou, která by se těžko či vůbec dala ve třídě demonstrovat. Programem se dá nová látka procvičit a tak pomoci lépe spojit nově nabyté fragmenty učiva, případně odhalit mezery v pochopení látky, které nebyli na první pohled zřejmé. Dalším případem, kdy může nastat čas sáhnout po výukovém programu je blížící se test a žákův deficit znalostí způsobený jak fyzickou, tak často i mentální nepřítomností při výkladu látky, a program tedy musí suplovat celou výuku.

Podle účelu se tedy může měnit celý koncept výukového programu. V tomto dokumentu je uvažován program, který by se hodil žákům, kteří si chtějí osvěžit určitou látku před testem či pokud potřebují látku pochopit zcela od začátku do konce, neboť bylo při prezentaci učiva zvoleno pro jedince nevyhovující tempo.

Princip, který by mohl zajistit vyšší motivaci při použití výukového programu je popsán v kapitole 2. Je zde také popsána specifikace aplikace.

V kapitole 3 je popsán návrh aplikace postavený na definované specifikaci. Jsou popsány případy užití aplikace, E-R model a z něho vytvořený databázový model. V závěru kapitoly je popsáno uživatelské rozhraní.

Následující kapitola 4 se věnuje prostředkům, které jsou na takovouto aplikaci potřeba. Je rozebrána jak serverová část, tak také klientská část.

Kapitola o implementaci 5 popisuje strukturu adresářů a jednotlivé části aplikace, které řeší jádro celé práce.

Poslední část se věnuje testování. Jsou zde probrány dvě varianty testování. V prvním případě je cílem funkční zdrojový kód, druhá varianta je pro zpětnou vazbu na celou aplikaci.

V samotném závěru jsou shrnuty výsledky práce a jsou zde uvedeny směry dalšího vývoje.

Kapitola 2

Gamifikace

V této kapitole budou představeny základní pojmy, které jsou základem této práce. Je zde rozebrán nejdůležitější pojem gamifikace. Dále je popsána přísada každé hry - pojem flow. Část kapitoly je věnována faktorům, které ovlivňují motivaci hráčů her. Jelikož jde v této práci o získání návodu, jak udělat neherní prostředí návykové jako jsou hry, je nezbytné hry samotné podrobit průzkumu.

Rozboru konkrétních vizuálních prvků je věnována navazující sekce. Sekce poté analyzuje tři projekty, na nichž jsou patrné prvky gamifikace, a jejichž rozbor umožní si udělat představu o procesu gamifikace.

Na konci je rozepsána specifikace požadavků na vyvýjenou aplikaci postavenou na získaných znalostech z předcházejících sekcí této kapitoly.

2.1 Gamifikace

Pojem gamifikace není nic nového. Jak název napovídá, jde o aplikování hry (z angl. game), a podle strohé, ovšem výstižné definice je tomu tak. Jde o použití herních mechanik v neherním prostředí. Tyto prvky lze nalézt již v dávné historii. Není to žádný výmysl posledních pár let. Jistě nebude potřeba zmiňovat více o J. A. Komenském než jeho školu hrou, kterou hlásat již kolem roku 1650, kde se právě mechanismy her snažil zavést do neherního prostředí školy. Termín gamifikace se začal používat opravdu nedávno. Poprvé byl použit v roce 2002 a do povědomí se dostal teprve v roce 2010. [7]

Otázkou může být, proč vůbec zavádět herní prvky do jiných prostředí. Počítačové hry jsou dost často brány jako něco, co bere čas a nic to nepřináší. Přesto u her lidé stráví spousty hodin týdně, v extrémních případech i spoustu hodin denně. Pokud by se tedy fakt, že u jedné věci lidé vydrží spoustu hodin, dal převést do jiného prostředí, vyřešil by se mnohdy nejdůležitější aspekt dané věci. Kupříkladu matematika, často největší strašák na základních školách, by pár hodin týdně - pro pochopení podstaty látky a procvičení - jistě uvítala.

2.2 Flow

Oním zázrakem, který se skrývá ve hrách a udržuje hráče v neustálém soustředění a zapálení, se v psychologii označuje termínem Flow. Je to duševní stav, kdy je naše pozornost motivována přes naše emoce a je docíleno naprostého soustředění. Tento termín není vázán na herní oblast, do tzv. flow se lidé dostávají běžně při činnostech, které jsou pro ně

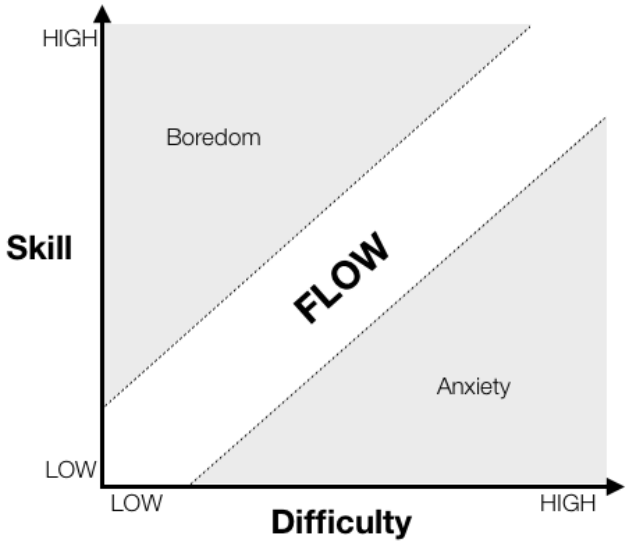
nejvyšším zdrojem radosti a zábavy, tedy i zmiňované hry.

Dlouhodobé soustředění můžeme u člověka vyvolat tím, že splní tři podmínky Flow teorie [3]:

- Úkol musí být dán jasnými kroky, které vedou k jeho splnění a musí být vidět pokrok, kterého již člověk při plnění dosáhl.
- Úkol musí získávat zpětnou vazbu a pomáhat člověku vypořádat se s danou situací podle jeho momentálních schopností.
- Člověk musí vnímat rovnováhu mezi svými schopnostmi a výzvou, kterou určitý úkol představuje, aby si byl jist, že daný úkol je dostatečně jednoduchý, aby ho dokázaů splnit, ale musí to být pro něj dostatečná výzva, aby se nenudil.

Na obr. 2.1 je zobrazen balanc dvou faktorů, které flow ovlivňují. Horizontální osa znázorňuje obtížnost či výzvu, jakou úkol představuje. Vertikální osa ukazuje momentální úroveň dovedností.

Pokud je obtížnost úkolu zvyšována se zvyšujícím se umem, je pravděpodobné, že to bude hráče bavit.



Obrázek 2.1: Znázornění závislosti flow na dovednostech a obtížnosti.

2.3 Motivace ve hrách

Stimulace motivace je ve hrách klíčovým prvkem celého úspěchu her. Spousta her je ve své podstatě dřina. Pokud ovšem dodáme určité činnosti potřebné prvky a mechanismy, tak se nudné činnosti stává zábava.

Bez následujících kroků by hry neměli čím nalákat, a byli by pouze dřinou bez přínosu.

Ve hrách se objevuje sedm faktorů, které ovlivňují naši motivaci [8]:

1. Progress bar, český překlad by mohl být indikátor průběhu. Na indikátoru vidíme, kde se zrovna nacházíme a kolik nám ještě zbývá splnit.

2. Dlouhodobé a krátkodobé cíle. Krátkodobé jsou splněny brzo, takže má hráč radost, že už něco zvládl, ale je potřeba mozek dopovat pomyšlením na splnění velkého úkolu, a tím si udržovat hráčovo zapálení.

3. Odměnovat snahu, brát chyby jako možnou součást každé snahy. Tím nedojde ke stavu, že by měl člověk strach něco provést, aby za to nebyl potrestán.

4. Rychlá, častá odezva na vše co vyzkoušíme. Pokud uděláme chybu, měli bychom dostat najevo, kde je problém a poučit se z toho.

5. Pokud je ve hře dáno, že při určité činnosti bude hráč odměněn velkou věcí s ovšem malou šancí, že se tak stane, tak i přesto je hráč velmi motivovaný pokoušet štěstí.

6. Jistota v odměňovací systém. Pokud člověk ví, že budu odměněn, dokáže více riskovat a víc toho dokáže.

7. Další lidé ve hře, se kterými může hráč sdílet své úspěchy a strasti. Spoustu věcí lidé dělají pouze z důvodu, že tím mohou ohromit druhé. Naopak pomoc druhým, kteří mají nesnáze, přináší člověku dobrý pocit.

Aplikace těchto faktorů musí být ovšem podrobena značnému testování s cílovými skupinami, neboť se takovéto herní systémy nedají nastavit bez správné zpětné vazby.

Určité zvážení si zaslouží sedmý bod výše zmíněného seznamu. V dnešní době je trendem budovat v mnoha webových a mobilních aplikacích komunitu. Nemůžeme ovšem předpokládat, že po zavedení takovéhoto kroku se začne komunita rozrůstat jen z toho důvodu, že k tomu mají prostředky. Opět musí nastat mnoho kroků testování, kterými se vývojáři aplikací dozvědí, které variace tohoto prvku fungují a vedou k požadovaným výsledkům, a které nemají žádný vliv, či naopak odrazují uživatele od námi chtěných činností. Jelikož se tento krok objevuje opravdu masivně, je na zvážení, zda je nezbytně nutné jej vyvýjet i v další aplikaci, či si uživatel vystačí se svými výsledky.

2.4 Vizuální prvky gamifikace

Faktory, které ovlivňují motivaci hráče se používají pro gamifikaci neherního prostředí. Zde popíši vizuálních prvky, které lze nalézt ve většině her, a které byli použiti v mé aplikaci. Poté se krátce věnuji dalším prvkům, které jsou použity v jiných gamifikovaných projektech, ale u této aplikace použity nebyly.

Obrázek 2.2 slouží pro představu, jak vypadá zapojení herních prvků do neherního prostředí.

Standardní stav při učení se matematiky je takový, že běžně nevíme, kolik nám ještě zbývá konkrétního učiva. Jestliže se ovšem učivo zpracuje do celků, které mají daný konečný stav, může se uživateli zobrazovat průběh plnění těchto celků. V mé aplikaci je učivo rozděleno jednak po ročnících, dále na jednotlivé kurzy určité oblasti učiva a následně na podtémata kurzu, které do dané oblasti učiva patří.

Jednotlivé kurzy představují úkoly, konkrétně úkoly dlouhodobé, neboť jejich splnění trvá daleko déle, než splnění jednoho podtématu. Podtémata také představují úkoly, tentokrát krátkodobé, jelikož je možné je dokončit během několika sérií cvičení.

Obrázek 2.2: Popis herních prvků

Průběh kurzu je na první pohled zřejmý tím, že jsou dokončená podtémata vizuálně odlišena od nedokončených. Uživatel si tímto lehce udělá představu, kolik má ještě úkolů před sebou. Další zobrazení průběhu je na samotných podtématech kurzu. Každé téma má indikátor průběhu, jehož hodnota odpovídá stavu tématu. Poslením indikátorem průběhu je indikátor určující stav, ve kterém se nachází aktuální úroveň kurzu.

Úroveň kurzu se zvyšuje plněním jednotlivých podtémat kurzu a také opakováním učiva. Dosáhnutí nové úrovně je navíc s každou další úrovní obtížnější, a uživatel může být tedy s každou další získanou úrovní náležitě hrdý.

Další vizuální prvky

Dalšími vizuálními prvky, které se v gamifikovaných projektech objevují, je například bodový systém. Za určité úkony dostane uživatel určitý počet bodů, a tyto body se mu viditelně zobrazují. I když na tomto principu funguje i herní systém u tohoto projektu, body se nezobrazují přímo, ale z jejich výše se počítají indikátory průběhu. Důvody pro toto řešení jsou popsány v kapitole o testování v sekci 6.2.

Vizuální reprezentace sociálních vazeb se řeší většinou tak, že jsou uživatelé navzájem informování o pokroku ostatních uživatelů. Sestavují se žebříčky podle nejrůznějších kritérií, nejčastěji podle součtu všech získaných bodů.

2.5 Gamifikované projekty

Pro demostraci gamifikace neherního prostředí jsem si vybral tři projekty na vzdělávávání. Prvním z nich je projekt Khan academy[1], který se zabývá - mimo jiné - přímo výukou

[1] http://www.khanacademy.org

matematiky. Dalším projektem je velice pěkná webová aplikace Tutpup[2], která má skvělou funcionalitu popsanou dále. Posledním projektem je Duolingo[3], který mě inspiroval k vytvoření tohoto projektu. Jeho popis ponechám také v sekci jemu věnované, pouze doplním, že se jedná o aplikaci, která učí cizím jazykům.

Popíši zde jejich vlastnosti a srovnám je s výše uvedeném seznamu definujícího sedm faktorů motivace ve hrách.

Khan academy

Khan academy vznikl nejprve jako sbírka videí na Youtube vysvětlující matematické témata. Později byla přidána část s možností procvičovat příklady. Na tuto část se zaměřím.

Oblasti učiva jsou rozděleny do stromu znalostí, kdy nahoře je základní učivo a postupně se k němu přidávají těžší oblasti. Pod jednotlivými body stromu se skrývá právě jedno cvičení, které se splní podle toho, jak rychle a správně uživatel odpovídá.

Tyto témata lze považovat za cíle, které si uživatel může zvolit a plnit. Objevuje se zde i prvek indikátor průběhu použitý pro vizualizaci pokroku tématu.

Při plnění cvičení je možné si zobrazit nápovědu, která daný příklad vysvětluje. Při špatné odpovědi se ovšem uživatel nedozví, v čem dělá chybu. Po podrobném prozkoumání zdrojových kódů jsem narazil na možnost sdělit uživateli chybu přímo podle zadané odpovědi, to je ovšem používáno velmi zřídka.

Je možnost získat různé odznaky za rychlou odpověď, za sérii rychlých odpovědí po sobě a podobně. Jelikož uživatel netuší, čím bude nečekaně odměněn, může mít u některých uživatelů tento odměňovací systém pozitivní vliv na jejich motivaci.

Khan academy nevyužívá faktoru komunity lidí. V aplikaci není možnost si přidat své přátelé a vidět jejich pokrok v učení. Někteří by jej určitě ocenili, ale předpokládám, že se Khan academy rozhodla tento prvek do aplikace nezapojovat z důvodů, že to není tak nezbytné.

Tutpup

V této aplikaci má uživatel také možnost si procvičit matematiku, ovšem jen základní věci. Navíc má aplikace trochu odlišný koncept než Khan academy. Nejzajímavějším prvkem je způsob procvičování, které má formu závodu. Uživatel si vybere téma, například sčítání, dále si zvolí obtížnost tohoto tématu a poté vyčká až ho aplikace spojí s dalším uživatelem, se kterým poté závodí. Na obrazovce vidí příklad, který musí aktuální vyřešit a závodní dráhu s avatary závodníků. Po úspěšném vyřešení příkladu se avatar uživatele posune o krok dopředu. Kdo takto dojede nejdále je vítěz.

Oproti Khan academy má tato aplikace méně propracovaný systém zpětné vazby. Uživatel se může dozvědět, která témata již prošel, ale nemá zpětnou vazbu o tom, která oblast je problematická.

Dají se zde také sbírat různá ocenění za to, kolik cvičení uživatel prošel, kolikrát vyhrál duelů a podobně.

Duolingo

Duolingo umožňuje se naučit cizí jazyk. Je možnost si zvolit, že umím anlicky a chci se naučit například španěštinu. Na domovské stránce jazyka je zobrazen strom témat, kdy

[2] http://www.tutpup.com
[3] http://www.duolingo.com

na začátku jsou všechna témata zamknuta, kromě prvního tématu se základy. Po splnění tématu se odemknou další témata, která dodají nová slovíčka a gramatiku, ale využívají již naučených slovíček z předchozích témat. Tím se uživateli zvětšuje aktivní slovní zásoba, kterou si neustále opakuje. Aplikace rozpozná, které slovíčko dělá problémy, nebo které je potřeba zopakovat, neboť nebylo dlouho používáno, a je tedy docíleno neustálého udržování slovní zásoby.

Strom témat představuje jednotlivé úkoly. Tyto témata mají v sobě jednotlivé lekce, které se splní tím, že se projde série dvaceti vět složených z naučených slovíček, a které je nutné přeložit. Při špatné odpovědi je uživateli oznámeno v čem udělal chybu. Navíc je uživateli sdělena alternativa překladu i při správné odpovědi, neboť většinou lze větu přeložit více způsoby.

Je to trochu odlišný systém, než u Khan academy. Zde se za každou splněnou sérii dostane stejný počet bodů, a pokud se v průběhu série odpoví špatně více jak třikrát, série vět končí a žádné body se nezískají. Jelikož aplikaci sám používám, můžu z vlastních zkušeností říct, že je to neskutečně protivné. V tomto případě se přikláním k provedení Khan academy, kde je možné sérii příkladů projít vždy, liší se pouze získaný počet bodů. Je to ovšem můj subjektivní názor.

Je zde implementován sociální prvek. Uživatel si může přidat své známé i neznámé, ze kterých se mu vytváří žebříček podle bodů. Uživatel se tak může s ostaními předhánět, kdo bude nejvýše.

Kromě učení se nových témat, je zde možnost si zopakovat již naučená slovíčka. Jak již bylo zmíněno, jak u procházení lekcí jednotlivých témat, tak i zde u opakování aplikace uživateli vkládá do vět slovíčka k procvičení, která jsou pro uživatele problematická, nebo nebyla dlouho používána.

U každého jazyka lze zvyšovat úroveň jazyka udávaný číselnou hodnotou úrovně. Uživatel si podle toho může ihned uvědomit, který jazyk je umí na vyšší úrovni, a ve kterém zaostává.

Co zde naopak není, je prvek náhodné odměny. Nemůže nás tedy u této aplikace nic překvapit, ale dá se opřít o jistý odměňovací systém, který má jasná pravidla.

2.6 Specifikace

V této části dokumentu specifikuji veškeré požadavky, které by měla výsledná aplikace splňovat.

Určení cílové skupiny je zjevné z názvu této práce. Aplikace je určena pro žáky na prvním stupni základní školy. K tomuto faktu je přihlíženo v následujících krocích, při kterých se specifikuje struktura aplikace.

Nejprve rozeberu mou představu o tom, jak by mělo být rozděleno učivo. Dále aplikuji určité prvky gamifikace a nakonec poznámka, která se týká výsledků testování aplikace.

Rozdělení učiva

Úkolem této aplikace je dodat matematické znalosti do uživatelovi hlavy. První věcí, kterou je potřeba rozmyslet, je přístup uživatele ke všem znalostem, které se má naučit, či procvičit. Khan academy a Tutpup jej rozděluje po tématech jako je násobení, dělení atd., u kterých se po splnění lehčího, může uživatel vrhnout na těžší variantu. To je dobré, pokud uživatel nezná dané téma vůbec a chce se ho naučit od začátku do konce. Žák na prvním stupni ovšem postupuje v učení násobení postupně přes násobilku, kterou se lze naučit zpaměti,

až po násobení několika cifernými čísly. Tento proces ovšem trvá celý stupeň. Není tedy praktické rozdělit látku na takovéto celky pojímající veškeré obtížnosti učiva.

Rozhodl jsem se učivo rozdělit podle ročníků, podle kterých se uživatel aplikace bude orientovat snáze, jelikož to jsou jednoduché pojmy, které pozná na první pohled, a jsou mu dobře známy. Tímto rozdělením nebude docházet ke zmatení uživatele velkým množstvím témat, která jsou k vidění v jiných aplikacích.

Dalším krokem je rozdělení učiva v rámci jednotlivých ročníků, neboť témat je v každém ročníků opět velké množství. Tyto témata se rozdělí na několik celků, jejichž obsah bude tematicky související. Takto bude v rámci ročníku vytvořeno několik celků. Celky budou representovat gamifikační prvek dlouhodobého cíle, které je možné během jednoho roku splnit. Počet takovýchto jednotek by se měl pohybovat v řádu jednotek, aby jich nebylo příliš. Rozsah učiva, který je rozdělen mezi tyto celky vychází z běžného rozsahu vyučovaného na základních školách [9].

Tyto celky nazývám ve své práci kurzy. Jde o to, aby si žák zvolil oblast učiva, například kurz násobení od 1 do 10, a vnímal tuto oblast matematiky a její souvislosti, které mezi sebou konkrétní oblast má.

Jestliže kurz představuje dlouhodobý cíl, tak jednotlivé části kurzu jsou krátkodobými cíly. Tyto části kurzu, neboli témata, představují krátké kapitoly ve velkém celku učiva. Tématem může být například násobení dvěmi, které patří do kurzu násobení od 1 do 10, což patří do druhého ročníku. Tyto souvislosti se pomocí této hierarchie lépe vnímají a pamatují, proto jsem tuto strukturu učiva zvolil.

Pokrok učiva

Je vhodné si udržovat přehled o tom, kde se v procesu učení uživatel nachází, tedy kolik toho má za sebou a kolik před sebou.

Stejnou věc má vyřešeno Duolingo tím, že nesplněné témata má zamknuta, a ty dokončená nebo rozpracovaná jsou vizuálně odlišena. Tento způsob interpretace pokroku kurzu specifikuji i v tomto projektu, neboť to považuji za přehledný a praktický způsob.

Pokrok jednotlivých témat bude zobrazen pomocí indikátoru průběhu. Na první pohled bude zřejmé, v jaké fázi se téma nachází. Splněná témata budou mít indikátor průběhu zcela vyplněný, což vzbudí v uživateli pozitivní pocit z dobře odvedené práce.

Jelikož se při procesu učení u lidí snižuje čas, potřebný na vyřešení úkolu, a počet chyb, které při tom dělají, bude vhodné odlišit ta témata, se kterými má uživatel stále ještě problémy. Uživatel se podle toho může rozhodnout, zda je to pro něj závažné, či mu dosažená znalost stačí a bude pokračovat dalším tématem. Toto rozličení bych opět provedl vizuálně. Využil bych tří barev na odlišení tří stavů úspěšnosti. Toto rozlišení barvami se běžně objevuje v tomto smyslu i na spoustě jiných místech kolem nás. Zelená barva indikátoru průběhu tématu bude určovat bezproblémový stav, oranžová barva bude značit určité nezávažné problémy při plnění tématu a červená barva uživatele upozorní na hrubé nedostatky v plnění.

Důležitým faktorem při studiu je si danou látku opakovat. Zde půjde opakovat dvěma způsoby. Jednak bude možné si zopakovat určité téma, které bude dokončené, v druhém případě bude možné si zopakovat všechna cvičení v rámci kurzu, která byla již úspěšně naučena.

Další věcí, kterou kurzy budou obsahovat, jsou úrovně. Během plnění kurzu se budou zvyšovat úrovně. Po dokončení kurzu můžou úrovně sloužit pro orientaci dalšího pokroku v kurzu, ačkoliv byl již celý splněn.

Cvičení

Každé cvičení bude mít názornou nápovědu, ve které bude vysvětleno, jak konkrétní příklad vyřešit.

Témata budou obsahovat různá cvičení, která se budou moct objevovit až po splnění předchozích cvičení v rámci stejného tématu. Půjde vlastně o rozdělení jednotlivých cvičení do úrovní. V každé úrovní bude moct být libovolně mnoho cvičení a budou mít různou pravděpodobnost zobrazení.

Pokud budu mít například téma sčítání do deseti, mohou se v něm objevit na první úrovni na rozehřátí příklady pouze do pěti, a po jejich splnění uživatel dostane příklady z další úrovně, kde mohou být již příklady do deseti.

Zpětná vazba

Bylo zamýšleno vytvořit bodový odměňovací systém, čímž by uživatel po splnění série cvičení obdržel určitý počet bodů. Toto bylo nakonec upraveno po otestování aplikace uživateli, což je popsáno v sekci 6.2.

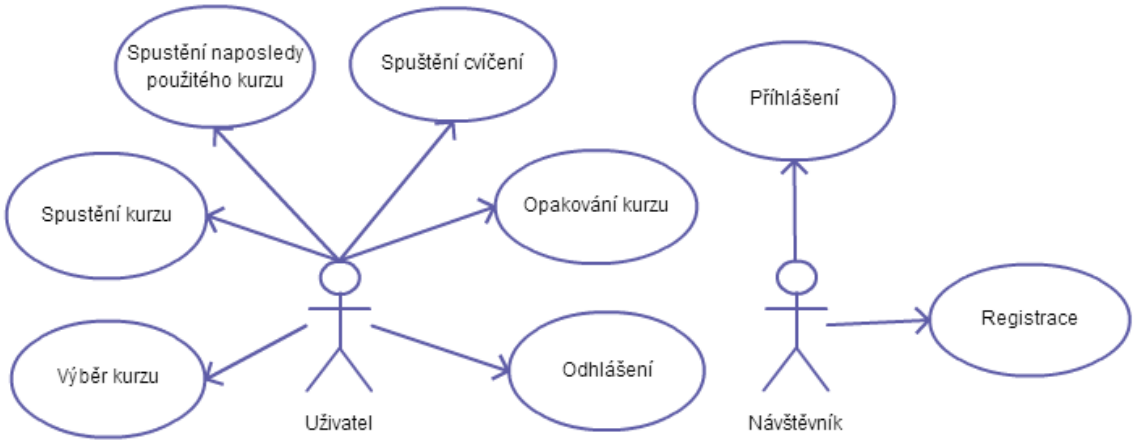
Kapitola 3

Návrh

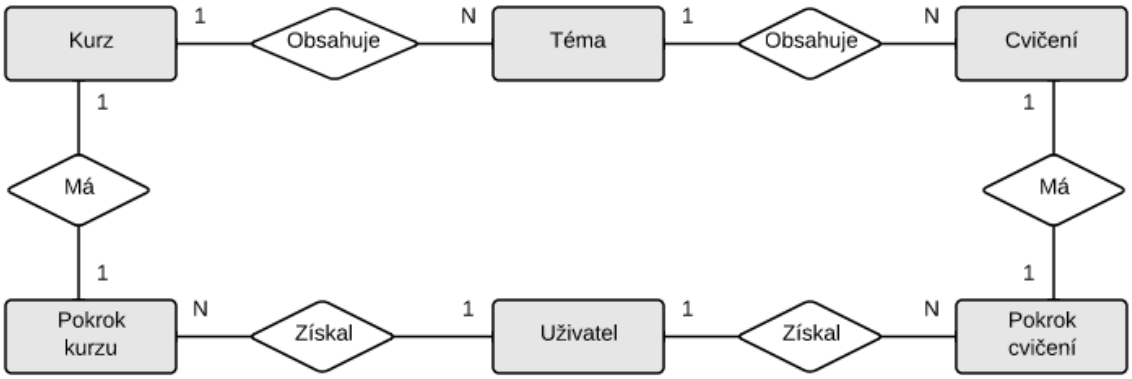
Při návrhu byl kladen důraz na jednoduchost celého programu, neboť je primárně určen mladým uživatelům. Bylo dbáno, aby výsledná aplikace mělo co nejméně nastavení, a co nejkratší cestu ke spuštění příkladů k procvičování. Veškeré nezbytné použití aplikace jsou popsány pomocí diagramu případů užití. Dále jsou pomocí E-R modelu definovány entitní množiny, které představují objekty, se kterými aplikace bude pracovat. Nakonec je popsáno uživatelské rozhraní.

3.1 Případy užití

V diagramu případu užití na obr. 3.1 je popsáno, které typy uživatelů budou aplikaci používat a jaké činnosti lze s aplikací provádět. Figurují zde dva typy uživatelů. Jeden představuje přihlášeného uživatele, který má přístup do aplikace, druhý představuje návštěvníka bez přihlášení.



Obrázek 3.1: Diagram případů užití



Obrázek 3.2: E-R model

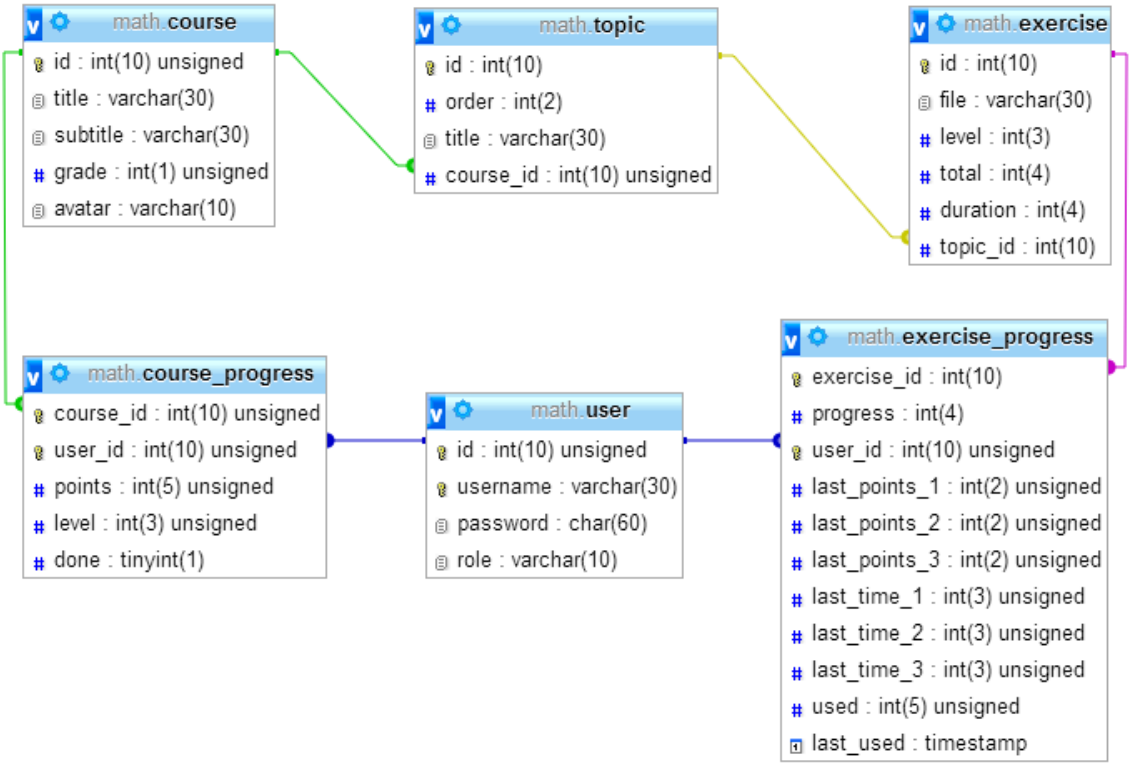
3.2 E-R model

E-R model (Entity-relationship) na obr. 3.2 je navržen podle specifikace v sekci 2.6. Je zde pomyslná hierarchie tří entitních množin. Na vrcholu je kurz, tabulka course, který představuje zvolenou látku k učení. Každá entita kurzu má vztah 1:N s množinou téma, tabulka topic. Každé téma má opět vztah 1:N s množinou cvičení, tabulka exercise. Každá entita z množiny `course` a `exercise` má k sobě entitu ve vztahu 1:1, která definuje její stav pro každého uživatele. Jelikož není nikde vztah M:N, není potřeba návrh transformovat.

Na obr. 3.3 je zobrazen databázový model vytvořený na E-R modelu.

Každá tabulka reprezentující výše zmiňovanou hierarchii a tabulka `user` mají primární klíč dán atributem `id` a tabulky určující stav kurzu, tedy tabulky `course_progress` a `exercise_progress` mají složený primární klíč ze dvou atributů. Prvním atribut s názvem odkazované tabulka a přípony `_id` a druhý je stejného principu odkazující na tabulku `user`, s názvem `user_id`.

- `course` uchovává název látky, bližší specifikaci látky v podtitulku, třídu, do které látka patří a název souboru avatara látky, který uživatele doprovází během procvičování.
- `course_progress` ukládá souhrn všech bodů získaných v kurzu, na který odkazuje v atributu `course_id`. Dále ukládá, na jaké úrovni se kurz nachází. To lze dopočítat z výše bodů, ale tato hodnota se nemění tak často, takže by bylo zbytečné ji neustále přepočítávat.
- `topic` uchovává název tématu a pořadí v kurzu, který zajišťuje postupné plnění témat vybraného kurzu.
- `exercise` obsahuje název souboru, který obsahuje celou logiku cvičení. Dále je definováno kolik je potřeba splnit bodů, aby bylo cvičení považováno za dokončené. Dobu trvání, od které se odvozuje počet získaných bodů. Úroveň, na které je cvičení zahrnuto do procvičování v rámci tématu.
- `exercise_progress` ukládá získané body za cvičení, a poslední tři použití, podle kterého se počítá úspěšnost celého tématu.
- `user` uchovává uživatelské jméno a heslo.



Obrázek 3.3: Databázový model

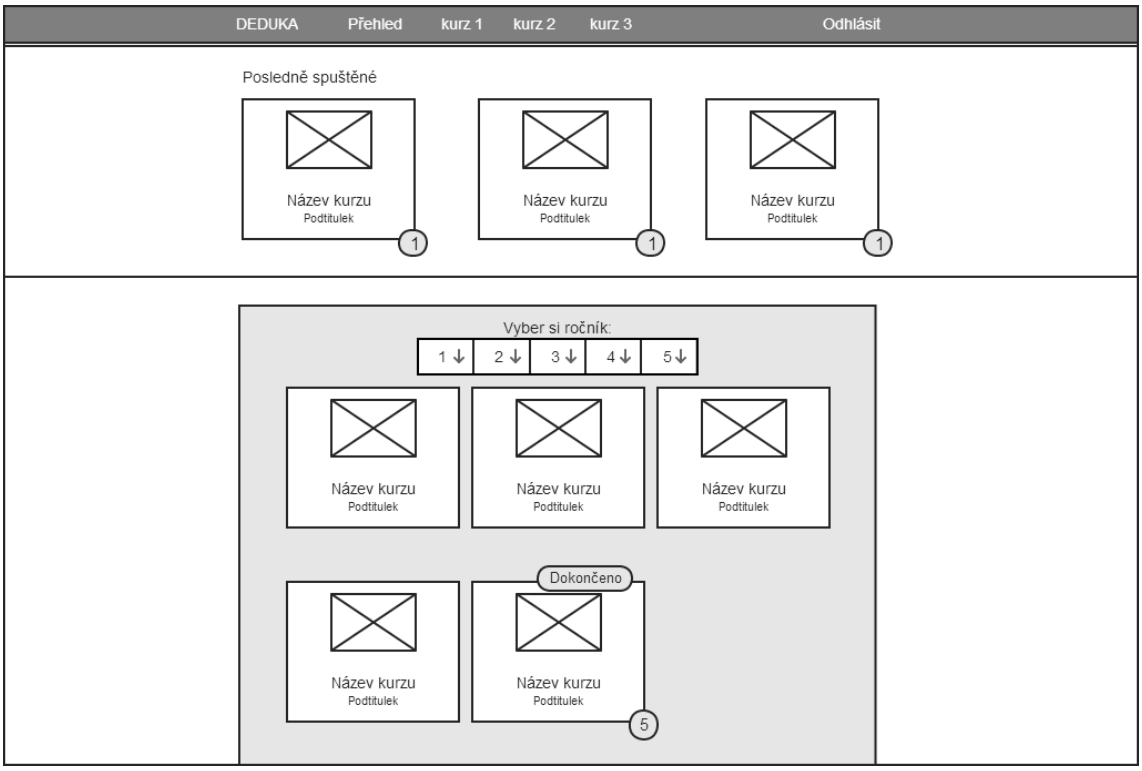
3.3 Uživatelské rozhraní

Jelikož se jedná o aplikaci pro mladé uživatele, vytvořil jsem uživatelské rozhraní s obrázky a pestré na barvy. Každému ročníku náleží barva, která se objevuje v různých odstínech na ovládacích prvcích a na pozadí stránek. Účelově zde není žádné vyhledávání témat, ani filtrování přes checkboxy a jim podobné ovládací prvky. Pohyb v aplikaci byl navržen tak, aby si uživatel intuitivně nalezl svou skupinu témat, které jsou určeny jeho dovednostem.

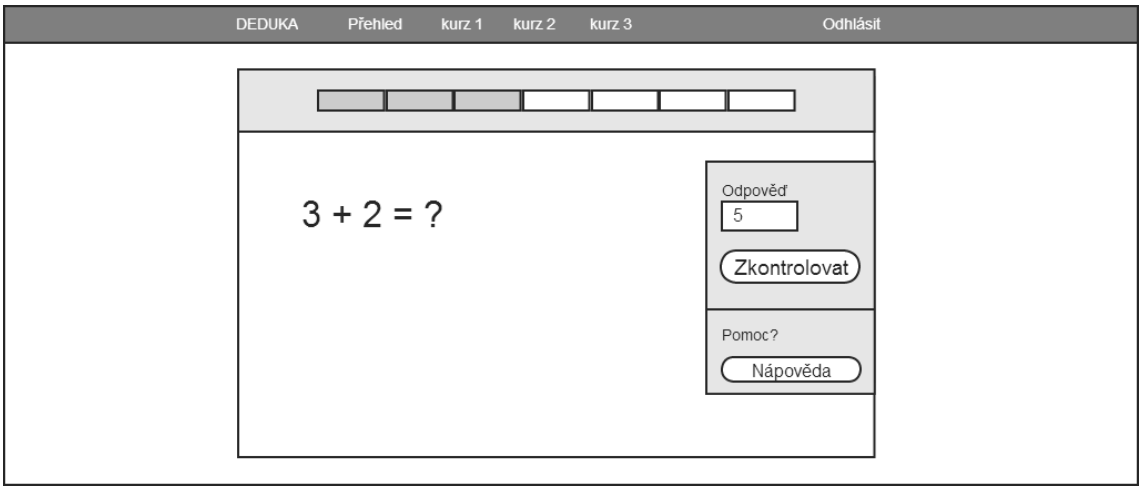
Na obr. 3.4 je zobrazena úvodní stránka po přihlášení. Vybírají se zde kurzy, které jsou rozděleny od shora dolů podle ročníků. Po otestování uživateli byly doplněny do horní části obrazovky posledně spuštěné kurzy, neboť na této stránce uživatelé hledali rozpracovaný kurz, aby v něm mohli pokračovat.

Obrazovka kurzu a její byla popsána v sekci 2.4.

Na obr. 3.5 je navrhnuto rozhraní cvičení. Byly použity pouze nezbytné prvky, aby se uživatel mohl soustředit pouze na cvičení. V horní části je graficky znázorněn aktuální průběh série příkladů, a uživatel tak má přehled, kolik mu ještě zbývá. Odpověď se vždy volí v pravé části obrazovky, kde také může získat nápovědu, pokud si neví rady.



Obrázek 3.4: Návrh úvodní obrazovky



Obrázek 3.5: Návrh obrazovky cvičení

Kapitola 4

Výběr technologií

Nejprve pár slov k tomu, proč vůbec takovouto aplikaci tvořit coby aplikaci webovou. V dnešní době, kdy má doma internet téměř každý je velmi praktické tvořit aplikace dostupné pouhým internetovým prohlížečem. Uživatel nemusí nic instalovat. Do aplikace se dostane ze kteréhokoliv počítače s internetovým připojením. Ne všechny aplikace se takto hodí tvořit, ale zamýšlená aplikace matematiky tomu zcela vyhovuje.

Tento webový projekt, jako snad téměř jakýkoliv jiný webový projekt, má svou funkčnost rozdělenou na dvě části, serverovou a klientskou. Na té serverové části se vyřizují nejrůznější požadavky a většinou se zde ukládají veškerá data. Po vyřízení požadavku se odesílá její výsledek na klienstkou část, kde se zobrazuje v internetovém prohlížeči.

4.1 Server

Pro vyřizování požadavků je potřeba technologie na jejich správu. Dále je nutné spravovat a uchovávat veškerá data, která budou potřeba k chodu aplikace. Pro takový účel bohatě postačí známá věrná dvojice většiny začínajících webových vývojářů, PHP a MySQL. Výhodou tedy je, že tyto technologie podporuje spousta hostingů a je okolo těchto technologií spoustu debat a řešení nejrůznějších problémů, které vývojář jistě ocení, až na nějaký takový problém narazí sám. Naštěstí už není potřeba používat čisté PHP, ale existují frameworky, které usnadňují vývoj aplikací jak po strukturní stránce, tak například i bezpečnostní. Jedním z hojně používaných v Česku je Nette Framework vyvýjený českými programátory.

4.1.1 Nette Framework

Framework je rozšíření pro PHP, technicky vzato jde o knihovnu.

PHP

PHP (PHP: Hypertext Preprocessor) [1] je skriptovací programovací jazyk. Je spuštěn na serveru a jeho princip spočívá ve spracovávání požadavků a odesílání výsledků zpět klientovy, který požadavek na server zaslal. Primárně se používá PHP na tvorbu dynamických stránek a webových aplikací, které se na klientské části zobrazují v HTML.

MVP

MVP (Model-View-Presenter) [5] je softwarový architektura rozdělující aplikaci na tři části, a která zpřehledňuje a usnadňuje vývoj. V aplikaci je oddělena část kódu, která obsluhuje

požadavky (presenter), od kódu který zajišťuje aplikační logiku (model), a část která se stará o zobrazení výsledku požadavku (view).

Pro lepší představu o úlohách těchto částí si zde pár vyjmenujeme:

- Presenter získá požadavek, podle kterého vykoná určité akce. Tyto akce mohou vyžadovat zpracování logiky v modelu, a poté výsledky vykreslit. Výsledkem může být nejen stránka, ale i obrázek, nebo přesměrování na jinou stránku.
- Model zajišťuje komunikaci s databází a vykonává veškerou logiku.
- View používá obvykle šablonovací systém Latte, který je další velkou výhodou Nette Frameworku. Šablona dostane data od presenteru a vykreslí výsledek, který se pak pošle klientovi.

Šablony

Šablonovací systém Latte je opravdu pokročilý a velice usnadňuje práci. Na menší ukázce můžeme demonstrovat, co umí.

```
<ul n:if="$items">
{foreach $items as $item}
   <li id="item-{$iterator->counter}">{$item|capitalize}</li>
{/foreach}
</ul>
```

Zápisu v ostrých závorkách se říká makra. Umějí vytvořit podmínku if nebo cykly jako jsou for, foreach, while a další.

Velkou výhodou je zabezpečení proti XSS a předejde se tak k vytvoření bezpečnostní chyby v aplikaci. Výpis proměnných jsou automaticky escapovány, na což by jinak mohl programátor lehce zapomenout. Dokonce jako jeden z mála umí Latte rozpoznat, kde se výpis nachází a zvolit podle toho sadu funkcí pro escapování, neboť v PHP se v různých místech dokumentu musí používat různé množství funkcí.

Pro představu jak Latte usnadňuje programování poslouží následující srovnání kódu, které jsou funkce totožné. Horní je kód Latte, pod ním zápis v PHP, což je kód, který Latte po zpracování zápisu vygeneruje:

```
{$item|capitalize}

<?php echo htmlSpecialChars(mb_convert_case($item, M_CASE_TITLE)) ?>
```

4.1.2 MySQL

MySQL je databázový systém. Jak už název napovídá, běží pomocí dotazovacího jazyka SQL (Structured Query Language [2]), přešněji v jeho dialektu s mírnými odlišnostmi. Data jsou uložena v tabulkách, které mají jednotlivé řádky se záznamy entit. U databáze, kterou používá více uživatelů, je důležité, aby nedošlo k nekonzistenci dat. MySQL má několik úložných enginů, které se hodí na různé případy. Běžně dostačující je InnoDB, které podporuje transakce, cizí klíče, zamykání řádku a body obnovení.

4.2 Klient

Klientská část webové aplikace je ta část, kterou vidí uživatel. Jde o grafické uživatelské rozhraní, přes které uživatel ovládá aplikaci. Vzhled je dán technologiemi HTML a CSS, a interaktivita je zajištěna pomocí Javascriptu. V dnešní době jsou naštěstí k dispozici Javascriptové knihovny, usnadňující práci. Pro chod cvičení je použita Javascriptový framework, který se stará o chod cvičení v projektu Khan Academy[1].

4.2.1 HTML a CSS

HTML (HyperText Markup Language) je značkovací jazyk pro tvorbu webových stránek. Jde o množinu značek a jejich atributů, kterými se určí význam obsahu takto vzniklého elementu a celkově definují strukturu dokumentu.

CSS (Cascading Style Sheets [6]) je jazyk pro definování vzhledu elementů HTML stránky. Pomocí selektoru se vybere element, či více elementů, a v části ohraničené ve složených závorkách, kterému se říká blok deklarací, se definují atributy elementu. Může se tak například měnit barva pozadí elementu, jeho šířka, výška apod.

Použitím kombinace HTML a CSS se oddělí obsah a struktura dokumentu a jeho vzhled. Lze tedy měnit vzhled HTML stránky bez zásahu do její struktury, což usnadňuje vývoj.

4.2.2 Javascript

Javascript je skriptovací jazyk, který lze použít jak na straně klienta, tak na straně serveru[4]. V tomto případě je použit pouze na klientské části, kdy je skript odeslán přímo v HTML dokumentu, nebo se na soubor s Javascriptem v dokumentu odkazuje. Javascript umožňuje ovlivňovat strukturu HTML dokumentu či vyvolat různá dialogová okna. Nejčastěji se používá tak, že se přiřadí událost, která při spuštění vykoná definovanou funkcionalitu.

Javascript je součástí technologií, které umožnující tvorbu interaktivních webových stránek, známý pod pojmem AJAX[5]. Ten umožní získat data ze serveru bez nutnosti překreslení celé stránky a získání pouze potřebných dat, které byly požadovány.

V projektu je použita knihovna JQuery. Ta umí vybírat elementy dokumentu pomocí selektorů se stejnou syntaxí jako selektory CSS. Je snadné s ní nastavit události na vybrané elementy, měnit elementům atributy, nebo provádět animace.

4.2.3 Framework pro cvičení

Khan Academy používá pro vývoj a testování nových cvičení do své aplikace Framework[1] postavený na HTML, CSS a Javascriptu.

Framework se skládá z jednotlivých cvičení, JQuery pluginu a spoustou drobných skriptů, které zajišťují výpočty nebo vykreslování pro určité typy úloh.

Proces, kterým prochází cvičení, aby se zobrazilo uživateli, je lehce komplikovanější, proto tomu věnuji pár slov. Jedná se o původní proces, který vykonával framework před úpravou pro tento projekt. HTML soubor s cvičením definuje pouze jak se bude příklad chovat. Neobsahuje žádnou definici struktury stránky, ani vzhled. Po otevření tohoto souboru

[1] http://www.khanacademy.com
[4] Např. Node.js
[5] Asynchronous Javascript and XML
[1] https://github.com/Khan/khan-exercises

dojde ke spuštění JQuery pluginu v souboru `khan-exercise.js`. To je jádrou celého frameworku. Plugin pomocí AJAXu načte soubory `khan-site.html` a `khan-exercise.html`, které definují vzhled stránky. Poté z původního souboru cvičení zjistí, jaké Javasripty je potřeba načíst, aby bylo možné zadání cvičení vytvořit do nového HTML dokumentu složeného ze získaných souborů.

Jakmile je cvičení vytvořeno, lze odpověď na zadání příkladu a zkontrolovat odpověd. Pokud je odpověď správně, zavolá se v `khan-exercise.js` opětovné vykreslení zadání téhož příkladu, pouze s jinými čísly.

Cvičení

Na uplném začátku se v tagu `<html>` definuje jaké skripty bude pro toto cvičení potřeba ve frameworku načíst.

V těle dokumentu jsou následující části, které mají pro framework specifický význam:

- `vars` obsahuje definování proměnných, které lze v dalších částech použít. Může se zde například vygenerovat náhodné číslo v daném rozmezí. Navíc lze nastavit, aby vygenerované číslo splňovalo určitou podmínku.
- `question` je textové zadání úkolu.
- `graphie` je část, která vyžaduje skript `graphie.js`. Pomocí tohoto skriptu lze definovat rozměr kreslícího plátna a na něj vykreslit grafické útvary.
- `solution` obsahuje nastavení výsledku příkladu.
- `hints` v této části se definuje nápověda. Každý blok či odstavec coby potomek tohoto elementu je jeden krok nápovědy. Může se zde opět oběvit element graphie, takže i nápověda může mít grafické útvary.

Ukázka cvičení

Pro úplnou představu zde popíši jedno ze cvičení, které má za úkol procvičovat sčítání čísel v oboru hodnot od jedné do deseti.

Cvičení se zkládá z výše zmíněných položek v seznamu.

```
<div class="vars">
    <var id="A">randRange( 1, 9 )</var>
    <var id="TMP_B">randRange( 1, 9 )</var>
    <var id="B">( A + TMP_B &gt; 10 ? 10-randRange(A, 9) : TMP_B)</var>
</div>
```

Obrázek 4.1: Inicializace proměných

Pro sčítání v oboru do deseti je nutné zajistit, aby nejvyšší možný výsledek byl 10. Proto se vygeneruje dočasná proměnná `TMP_B`, která slouží ke kontrole této podmínky, a až poté se vygeneruje konečná proměnná `B`. Zápis říká, že pokud bude součet vygenerovaných proměnných vyšší než 10, vygeneruje se nové číslo v rozsahu, který zaručí, že součet již bude maximálně 10.

```
<div class="question">
    <div class="graphie">
        init({
            range: [ [0, 12], [-1, 1] ]
        });
        label( [0, 0],
            "\\Huge{\\color{#6495ED}{" + A + "} + \\color{#28AE7B}
            {" + B + "} = {?}}", "right" );
    </div>
</div>
```

Obrázek 4.2: Vykresleni zadání

```
<div class="solution" data-forms="integer"><var>A + B</var></div>
```

Obrázek 4.3: Sestavení odpovědi

Zadání u tohoto cvičení využívá modulu `graphie`, který si inicializuje plátno a do něho barevně vykreslí vygenerované sčítance.

Stanovení výsledku je v tomto případě velice jednoduchý, neboť jde o sečtení proměnné `A` a `B`. V zápisu se ještě stanovuje formát výsledku, podle kterého se poté různě vykresluje formulář pro odpověď. V tomto případě je výsledek `integer`, formulář bude mít textové vstupní pole, a bude v něm očekávat pouze celá čísla.

```
<div class="hints">
    <div class="graphie" style="float: left;">
        drawCircles( A, "#6495ED" );
    </div>
    <div class="graphie" style="float: left;">
        drawCircles( B, "#28AE7B" );
    </div>
    <p style="clear: left;">Máme <var>plural(A, "modrou tečku",
     "modré tečky", "modrých teček")</var> a <var>plural(B,
     "zelenou tečku", "zelené tečky","zelených teček")</var>,
     tedy dohromady <var>plural(A+B, "tečku", "tečky", "teček")</var>
     .</p>
</div>
```

Obrázek 4.4: Vykreslení nápovědy

Uvnitř bloku `hints` jsou tři další elementy, čímž je určeno, že tento příklad bude mít tři kroky nápovědy. V prvním i druhém kroku se opět využívá modulu `graphie`. Vykreslí se počet teček odpovídající číslu `A` a `B`. V posledním kroku nápovědy se vypíše textová zpráva, která využívá funkce frameworku `plural`, kterou ovšem bylo nutné upravit pro český jazyk. Odpověď bude mít tímto zajištěno, že se bude text bude vypisovat se správnou gramatikou.

Plugin

Plugin je tvořen dvěma módy. Prvním je testovací mód, který slouží k lokálnímu spuštění vytvářeného cvičení. Druhý mód umí přepínat mezi různými cvičeními.

Testovací mód se nastaví standardně při otevření HTML souboru s cvičením. Mód využívá pouze funkce k vykreslení příkladu, po jehož splnění uživateli opět vykreslí stejné cvičení, pouze s jinými hodnotami. Jelikož se mají střídat různá cvičení, je tento mód nedostačující.

Druhý mód využívá celou šířku funkcí pluginu. Umožňuje během plnění příkladů získat pomocí AJAXu jiný soubor s cvičením a překreslit celé cvičení. Tento mód se nastaví, pokud je pluginu dodána externí proměnná `Exercises`, kterou v ostrém provozu na serveru Khan academy dodává jejich aplikace. Jelikož je primárně tento framework k dispozici za účelem otestovat si cvičení, není známo jakou strukturu by tato proměnná měla mít. Při pokusu takovouto proměnnou pluginu dodat jsem zjistil, že na různých místech se dožaduje spousty dat z proměnné, jejichž formát a obsah by bylo možné pouze odhadovat.

Abych docílil funkcionality popsané v 2.6, musel jsem zkombinovat oba módy. Testovací, protože zkrátka fungoval, a druhý mód, protože uměl to, co bylo potřeba. Provedení této úpravy je rozepsána v sekci 5.5.

Kapitola 5

Implementace

Tato část dokumentu se věnuje implementaci řešení, které je založeno na návrhu aplikace.

Na začátku je popsána struktura celého projektu z pohledu rozmístění zdrojových kodů v adresářích. Dále jsou rozepsány procesy, třídy, ze kterých se aplikace skládá, a poté je věnována sekce změnám ve frameworku zajišťujícímu chod cvičení.

5.1 Adresářová struktura

Adresářová struktura v tomto projektu je postavena na struktuře MVP (Model-View-Presenter), který se používá v Nette Framework[1]. Důležitá část struktury je adresář `www`, který je veřejně přístupný a spouští aplikaci v souborem `index.php`. Další důležitá část je adresář `app`, který obsahuje PHP soubory rozděleny podle MVP.

```
app/                    Zdrojové soubory serverové části
   config/              Nastavení Nette a databáze
   model/               Modelová vrstva a její třídy
   presenters/          Třídy presenterů
   templates/           Šablony Latte
libs/                   Adresář s PHP knihovnami
   Nette/               Nette knihovna
log/                    Ukládání záznamů o chybách
temp/                   Ukládá dočasné soubory a cache
www/                    Veřejný kořenový adresář webu
   css/                 Kaskádové styly
   e/                   Khan academy framework
      exercises/        HTML soubory definující cvičení
      utils/            Javascriptové kódy pro framework
   images/              Obrázky
   js/                  Soubory s Javascriptovými kódy
```

[1] Adresářová struktura Nette Framework http://doc.nette.org/cs/presenters#toc-adresarova-struktura

5.2 Registrace a přihlašování

Aplikace musí umět rozlišit, který uživatel určitou operaci provádí. Uživatele je tedy nutné před vstupem do aplikace zaregistrovat, a při každém dalším vstupu do aplikace ověřovat jeho identitu pomocí získaného `session`, nebo opětovného zadání přístupových údajů.

Nejprve popíši proces registrace, abych dodržel souslednost jednotlivých kroků, kterými uživatel při vstupu prochází.

Registrace

Registrace probíhá těmito kroky:

1. Presenter `Sign` se postará o vytvoření registračního formuláře.
2. Uživatel vyplní uživatelské jméno a heslo, a formulář odešle.
3. Stejný presenter vyplněné údaje z formuláře zpracuje, k čemuž volá metodu modelu `Users`.
4. Metoda se pokusí najít uživatele v tabulce `user`.
5. Pokud je záznam nalezen, znamená to, že je uživatel již zaregistrován. Toto zjištění se dá uživateli na vědomí zprávou a registrace se ukončí.
6. Přidá se údaje coby nový záznam. Heslo se zahashuje pomocí metody z modelu `Authenticator`.
7. Záznam je použit pro přihlášení do aplikace. Tento proces je popsán podrobně následně.

Přihlašování

Přihlašování do aplikace, neboli autentizace, probíhá v případě, pokud aplikace nemá k dispozici unikátní `session id`, které je přiřazeno uživateli na určitou dobu.

1. Presenter `Sign` vytvoří přihlašovací formulář.
2. Uživatel vyplní uživatelské jméno a heslo, a formulář odešle.
3. Stejný presenter zavolá metodu třídy knihovny Nette, která se pokusí uživatele přihlásit.
4. V případě, že uživatel nevyplnil korektní údaje, metoda pro přihlašování vyvolá výjimku `AuthenticationException`, která je v presenteru zachycena. Tento stav dá uživateli na vědomí a přihlašování se ukončí.
5. Po úspěšném ověření údajů je uživateli vytvořena instance třídy `Identity`, která obsahuje veškeré potřebné údaje o uživateli, včetně nalezeného záznamu z databáze.

Bezpečné ukládání hesel

Zvláštní pozornost věnuji bezpečnosti ukládání hesel.

Ačkoliv může vývojáře webové aplikace svádět ukládat hesla ve formě prostého textu, neboť samotná databáze je chráněna heslem, je nezbytné hesla minimálně hashovat. Webové aplikace je nutné ošetřit proti velkému množství potenciálních útoků, které mohou vést například k získání všech záznamů databáze. Přestože aplikaci ošetříme proti všemožným útokům, ročně se objeví kolem padesáti nových druhů útoků, a není tedy jisté, že se nám podaří ošetřit všechna ohrožení.

Hesla se v této aplikaci hashují funkcí `SHA-1`. Navíc je použita tzv. sůl, která zajišťuje, aby stejná hesla měla v databázi rozdílnou hash. Útočník po prolomení jednoho hesla tím nezíská přístup ke všem dalších uživatelům, kteří mají stejné heslo.

Ověření hesla probíhá tak, že se heslo z formuláře zahashuje popsaným způsobem a vytvořená hash se porovná z uloženou v databázi.

Tento způsob ukládání hesla není již v dnešní době dostačující. Po získání hashe jej může útočník relativně rychle prolomit, neboť může využít již předgenerovaných hashů nazývaných `rainbow table`. Případně může pomocí výkoného počítače generovat velké množství nových hashů. Tam, kde je důležité mít hesla opravdu zabezpečena se používají funkce `scrypt` nebo `bcrypt`, které mají mnohem větší čas převedení hesla na hash, a prolomení hesla tedy trvá mnohem déle.

Při získání hesel z této aplikace by nebyly způsoby žádné finanční či jiné škody, proto není nutné použít takto náročné vytváření hashů.

5.3 Spuštění kurzu

Při implementaci spouštění kurzu byl brán ohled na použití co možná nejméne záznamů v databázi. Návrh datového modelu definuje pouze objekty, které budou v tabulkách uloženy, nedefinuje ovšem, jakým způsobem se tyto záznamy tvoří. Datový model obsahuje dvě tabulky, `course_progress` a `exercise_progress`, které určují, jak daleko uživatel v daném kurzu došel.

Jedna možnost, jak se záznamy naplnit, je pro každého nově registrovaného uživatele naplnit obě tabulky záznamy o jeho pokroku, kdy každý pokrok pro kurz či cvičení by byl nulový. Při následovném procvičování příkladů by měla aplikace k dispozici veškeré záznamy. Mělo by se ovšem předejít situacím, kdy se uživatel pouze zaregistruje a dále již aplikaci nepoužívá, či pouze minimálně.

Další možností je, zde uplatněné, řešení přidávat záznamy jen v případě potřeby. Z toho důvodu bylo zavedeno spouštění kurzu pomocí tlačítka, které v databázi přidá záznam o pokroku kurzu, a dále přidá záznamy o pokroku ke každému cvičení, které patří do prvního tématu tohoto kurzu v tabulce `topic`. Takovéto spuštění způsobí, že se kurz oběví na hlavní stránce v posledně spuštěných, takže je pravděpodobné, že si jej uživatel příště opět spustí.

5.4 Vyhodnocení úspěšnosti

Každá část aplikace, která hierarchicky tvoří matematické oblasti učiva - tedy kurz, téma a cvičení - používají různé mechanismy, které určují stav učiva konkrétního uživatele. Zde popíši jednotlivé části a jejich mechanismy.

Vyhodnocení cvičení

Hodnocení každého cvičení závisí na dvou faktorech. Čím méně chyb, tím lepší hodnocení, a čím kratší dobu trvá odpovědět, tím také více bodů.

Plný počet bodů, které lze za jedno cvičení získat je pět bodů. Každé cvičení má v databázi stanovem limit ve vteřinách, který stanovuje maximální dobu, za kterou musí uživatel cvičení splnit, aby nebyl penalizován ztrátou bodů. Z tohoto limitu se vypočtou další hranice, po jejichž překročení se penalizace zvyšuje. Stejně tak se odečítají body, za každou špatnou odpověď. Tento mechanismus zachytí pomalé odpovídání, nebo situace, kdy uživatel odpovídá sice rychle, ale odpovídá nahodile a dělá mnoho chyb.

Vyhodnocení témat

Témata vyhodnocují dvě záležitosti - jaké procento je již dokončeno a jak úspěšně si uživatel počínal.

Procento dokončeného se vypočítá jednoduše z aktuálního součtu všech získaných bodů cvičení patřící do onoho tématu a celkového počtu všech bodů, které lze u cvičení získat. Úspěšnost těchto cvičení vychází ze statistik, které se ukládají u každého cvičení. Jsou zaznamenávány poslední vyhodnocení každého cvičení, z nichž se vypočítá, jak moc bylo chybováno a zároveň, jestli výpočty netrvali příliš dlouho. Z tohoto výpočtu vyvodí tři stavy tématu - výborně splněno, dobře splněno a špatně splněno.

Vyhodnocení tématu se vizualizuje indikátorem průběhu, který má délku splněno podle splněných cvičení, a barva indikátoru je různá podle úspěšnosti cvičení. Zelená určuje výborný stav, oranžová dobrý a červená špatný.

Vyhodnocení kurzu

Průběh kurzu lze sledovat dvěma indikátory. Jedním z nich je počet splněných témat, což lze vypozorovat na první pohled. Není potřeba nejrůznějších popisků, statistik a tak různě. Dalším indikátorem je indikátor průběhu celého kurzu založeného na počtu získaných bodů za všechna splněná cvičení. S vyšším počtem bodů se zvyšuje úroveň daného kurzu. V aplikaci je určen počet bodů, které je nutné nasbírat, aby byla úroveň kurzu zvýšena. Na indikátoru průběhu těchto úrovní je vidět stav aktuální úrovně a je vizuálně zobrazeno, kolik ještě zbývá do další úrovně.

Získat body lze z opakování dokončených cvičení. Toto opakování kurzu umožňuje získávat body, které by se již ze splněných témat nezískali. Především však umožňuje opakovat problematická místa učiva.

Výběr vhodných cvičení probíhá následujícím způsobem:

1. Všechna dokončená cvičení se ohodnotí číselným údajem, který udává svou výší úspěšnost plnění cvičení.

2. Podle výše těchto údajů o úspěšnosti se cvičení rozdělí do tří kategorií - nejméně problematické, středně problematické a nejvíce problematické.

3. Cvičení se seřadí od nejstaršího data posledního použití v rámci svých kategorií, takže bude zajištěno, aby se každý druh cvičení dostal postupem času zopakoval.

4. Do série příkladů, která se budou v režimu opakování počítat, se vyberou cvičení ze všech kategorií. Tím se aplikace přiblíží ke splnění jedné z podmínek pro možnost

dostat uživatele do Flow. Uživatel bude mít aktuálně problematické úkoly, které jsou pro něj výzvou, ale které mu zároveň méně problematické, které mu dodávají pocit, že danou sérii příkladů zvládne.

5.5 Změny v Khan academy frameworku

Jak již bylo zmíněno, framework pro cvičení používaný na Khan academy využívá Javascriptový plugin, který operuje ve dvou módech. Pro docílení funkcionality frameworku popsané v sekci 2.6 je nutné využít funkce druhého módu, které umožňují získání dalších cvičení v průběhu plnění jiného cvičení, a tím vytvořit sérii cvičení s různorodými příklady.

Po podrobném seznámení s Javascriptovým pluginem frameworku jsem zjistil, že se na mnoha místech větví program na podmínce, která testuje, zda je plugin v testovacím módu nebo není. Pokusil jsem se využít pouze druhý mód tím, že jsem dodal pluginu určitá data. Tím všechna větvení programu, s podmínkou testující mód, vykonala druhý blok programu, který přísluší druhému módu. Jelikož je framework primárně určet pouze k lokálnímu testování cvičení, nastávaly na mnoha místech požadavky na nejrůznejší data, která jsem nebyl schopen dodat, neboť data požadovaná pro přepnutí módu nejsou nikde specifikována, a jejich formát a obsah byl postaven pouze na mé domněnce.

Po marných pokusech přemluvit plugin ke spolupráci jsem se rozhodl postavit požadovanou funkcionalitu na testovacím módu, protože ten fungoval bez problému. Požadovaná funkcionalita byla docílena úpravami testovacího módu tak, že byly na určitých místech volány funkce z druhého módu a doplnění určitých dat, které jsou popsány dále. Došlo tedy ke kombinaci obou módů.

Nejprve opět zmíním fakt, že Nette framework vykresluje zpracované požadavky v Latte šablonách, které představují část View z MVP architektury. Jelikož jsou ovšem všechna cvičení Khan academy frameworku v HTML souborech a ne Latte šablonách, musely se nějakým způsobem do šablony načíst. Byla tedy vytvořena jedna šablona, která slouží pro všechna cvičení aplikace, a která spouští upravený plugin. Tato šablona přejala kostru struktury cvičení, které bylo nezbytné pro korektní spuštění pluginu. Jedná se o bloky s definicí proměnných, blok se zadáním apod., které jsou ovšem prázdné, neboť každé téma začíná jiným druhem příkladu, a nebylo by dobré vykreslovat například sčítání, když téma procvičuje obvod obdélníku.

Proces, kterým musí cvičení projít a změny ve frameworku, které kvůli tomu byly provedeny, jsou popsané v následujícím seznamu:

1. Při přejití na téma se zjistí, která cvičení do daného tématu patří.

2. Šabloně pro vykreslení cvičení se předají data, která obsahují názvy HTML souborů cvičení, maximum počtu bodů u jednotlivých cvičení, aktuální počet získaných bodů u každého cvičení, doba trvání každého cvičení a čísla úrovní, na kterých se jednotlivá cvičení nacházejí.

3. Další část probíhá z klientské části. Webový prohlížeč v podstatě dostane onu prázdnou kostru cvičení a data, která popisují veškerá data o cvičení, která uživatel bude procházet ve spuštěném tématu. Webová stránka spustí v souboru `khan-exercise.js` plugin, který pomocí AJAXu načte soubory `khan-site.html` a `khan-exercise.html`, které definují HTML strukturu bloku cvičení. Tyto soubory plugin vloží na své místo do struktury webové stránky. Stránka je takto inicializovaná, a může si začít načítat další cvičení a umožnit je plnit.

4. Plugin zjistí, které cvičení je momentálně na řade a zkontroluje, zda jeho HTML zadání nemá již staženo.

5. Pokud jej nemá stažen, což v případě prvního příkladu nemá, získá jej pomocí AJAXu a namapuje si název cvičení do proměnné `exerciseUsage`, podle které v příštím kole zjišťuje, zda již toto cvičení nestahoval dříve. Plugin si stažená cvičení ukládá do paměti a jelikož mé řešení dovoluje vracet se během procvičování k příkladům, které již jednou v rámci stejného cvičení byly načteny a spuštěny, musí se z oné proměnné zjistit index do uložených cvičení.

6. V opačném případě zjistí index cvičení a zavolá funkci pro render cvičení.

7. Po stisknutí tlačítka pro kontrolu dojde v neúspěšném případě k oznámení o chybě, a v případě úspěšného splnění příkladu se odešle souhrn dat ve formátu JSON pomocí AJAXu na server.

8. Každé splnění příkladu znamená jedno kolo v sérii cvičení, které se neustále opakuje, tak jak je popsané od 4. kroku. Po splnění sedmi kroků se procvičování ukončí, server vyhodnotí přijatá data, zaznamená do databáze úspěšnost a uživatel je přesměrován na stránku, kde je informován o průběhu procvičování.

Cvičení

Veškerá cvičení použitá v aplikaci bylo nutné přeložit do českého jazyka včetně nápovědy a upravit pro potřeby jednotlivých témat a ročníků.

Pro představu uvedu příklad. V mé aplikaci se nachází kurz Násobení a dělení. Kurz je rozdělen na děvět témat, od násobení a dělení dvěmi až po deset. Každé téma obsahuje cvičení s násobením a cvičení s dělením. Khan academy má ovšem k dispozici pouze cvičení, ve kterém se násobí zcela náhodnými čísli. Proto bylo nezbytné tato cvičení vytvořit pro každé téma zvlášť a upravit jejich zadání tak, aby se například v cvičení násobení dvěmi skutečně objevovala pouze násobilka dvěmi.

Navíc bylo implementováno, že každé cvičení patří na určitou úroveň v rámci svého tématu. Na ty cvičení s vyšší úrovní se uživatel dostane až tehdy, když splní cvičení s předcházející úrovně.

Kapitola 6

Testování

V této kapitole bude popsán proces testování aplikace. Tento proces rozdělím na dvě části podle jeho charakteru.

První věcí, která je při vývoji nutná otestovat, je správná funkčnost zdrojového kódu. Druhým faktorem úspěchu aplikace je její otestování uživateli a vyhodnocení zpětné vazby.

6.1 Debugování

Tento projekt je napsán v jazyce PHP a SQL na straně serveru, a HTML, CSS a Javascript na straně klienta. Na obě tyto části byly použity debagovací nástroje, neboť je běžné, že se při vývoji dostane nějaká chyba.

Debugování serveru

Obrázek 6.1: Chybová zpráva debugovacího nástroje

Při vývoji serverové části bylo využito nástroje pro debugování Nette frameworku [4]. Tento nástroj zobrazuje chybovou zprávu při výskytu chyby při vykonávání programu. Chybová zpráva je zobrazena na obrázku 6.1. Tato zpráva zobrazuje řádek, na kterém došlo

k chybě a ve většině případů i vysvětlení příčiny chyby. Zpráva také obsahuje stavy atributů, které se předávali do metody či funkce, ve které nastala chyba. Dále obsahuje HTTP požadavek včetně `$_GET` a `$_POST`, proměnnou `$_SERVER` a spoustu dalších, podle kterých se můžeme dopátrat příčiny chyby. Tato zpráva se zobrazuje pouze na lokálním serveru. Jakmile debugovací nástroj rozpozná, že se nachází na produkčním serveru, vypisuje místo této zprávy pouze informativní hlášku o chybě na serveru, a nestane se, že by návštěvníci webové stránky viděli stejně podrobné informace, jaké se zobrazují vývojáři.

To, že se zpráva nezobrazuje neznamená, že se tato zpráva nevytváří. Každá chyba je zapsána do logu, a navíc vytvořen HTML soubor s touto zprávou, takže si ji může vývojář na serveru otevřít a prozkoumat.

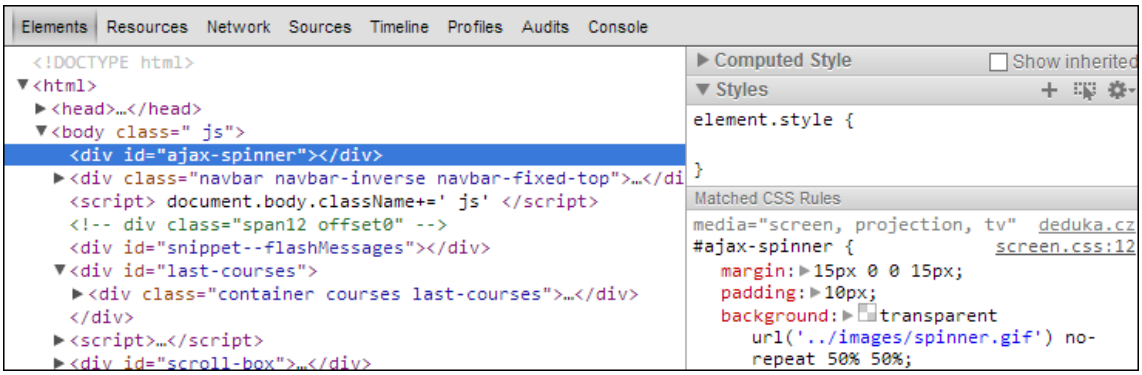
Nette také nabízí debugovací panel[1], který se opět zobrazuje pouze, pokud je na lokálním serveru. Tento panel zobrazuje řadu dat, která opět pomohou při hledání příčiny nějaké chyby.

Debugování klienta

Na straně klienta jsem využil debugovacího nástroje v prohlížeči Chrome. Ten lze použít pro běžné chyby hned pro všechny tři zmiňované jazyky klientské části.

Pomocí nástroje lze manipulovat s HTML elementy, přesouvat je, upravovat či mazat. Dá se pro každý element upravit CSS styl a změna je vidět ihned, lze takto pohodlně experimentovat se vzhledem, dokud se nám výsledek nelíbí, a změny pak přenést do zdrojového kódu.

Po přepnutí do záložky `console` se dostaneme na obrazovku, kde se vypisují případné chybové hlášení při chybě v Javascriptovém kódu.



Obrázek 6.2: Debugovací nástroj v prohlížeči Chrome

6.2 Testování s uživateli

Cílová skupina této aplikace jsou žáci prvního stupně základní školy. Usoudil jsem, že největší problémy s pochopením aplikace mohou nastat u mladších ročníků, neboť mají zpravidla méně zkušeností s technikou a méně znalostí matematiky. Proto byla aplikace otestována žáky z prvního a druhého ročníku.

Využil jsem jedné z hodin na místní základní škole a s žáky aplikaci otestoval.

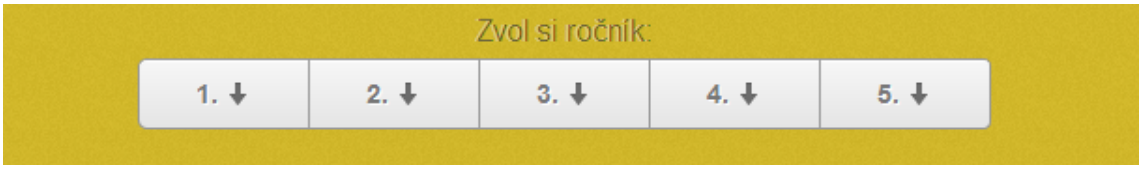
[1] V Nette nazývaný Debugger Bar

Jedním z bodů testování bylo získat zpětnou vazbu na použitelnost aplikace. Žáci neměli problémy s nalezením kurzů vhodných pro svůj ročník.

Dalším bodem k otestování byly parametry cvičení. Po skončení testování byly upraveny doby trvání příkladů, podle hodnot zanechaných v databázi, a bylo změněno množství bodů, které je nutné zdolat pro další posun.

Aplikace přidávala po projití série příkladů získané body. Ty však nebyli pro všechny zcela jasné, zvlášť s přihlédnutím, že většina mladších žáků se věnuje matematice krátce, a bodový systém byl mimo obor čísel, který žáci běžně používají. Veškeré body se projevují coby změna indikátoru průběhu. Jev plnící se oblasti, tím myslím indikátor průběhu, je pochopitelný bez problémů.

Dále byl přidán panel tlačítek na obr. 6.3, který slouží pro rychlý přesun na zvolený ročník, na čež jsem došel pozorováním používání aplikace.



Obrázek 6.3: Výběr ročníku

Kapitola 7

Závěr

Cílem bakalářské práce bylo vytvořit webovou aplikaci na procvičování matematiky pro první stupeň základní školy.

Seznámil jsem se s aplikacemi, které se věnují vzdělávání matematiky pro základní školy. Dále jsem se věnoval i dalšímu projektu, který výborně posloužil k seznámení s gamifikačními prvky. Zjistil jsem, jaké faktory ovlivňují motivaci uživatele aplikace.

Na základě těchto vzdělávacích projektů jsem stanovil specifikaci vlastní aplikace, která využívá gamifikačních prvků a také framework pro chod cvičení s příklady.

Podle definované specifikace vznikl návrh, který popisuje případy užití aplikace, její datový model a vzhled jednotlivých obrazovek aplikace.

Popsal jsem technologie, které byly při vývoji použity. Tento popis se zabývá serverovou a klientskou částí. Zde se objevuje podrobný popis frameworku pro cvičení, který byl v této práci použit.

Byla implementována aplikace, postavená na adresářové struktuře, popsané v této práci. Byly implementovány a popsány části aplikace, které se podílí na gamifikaci aplikace. Byly provedeny změny ve frameworku pro tvorbu cvičení, aby byl schopen fungovat podle definovaných specifikací.

Aplikace byla otestována žáky první a druhé třídy, čímž se zachytilo a provedlo několik změn v aplikaci.

7.1 Směry dalšího vývoje

Jedna z věcí, která by stála za vyzkoušení, je přidat do aplikace sociální vazby mezi uživateli. Uživatel by měl možnost si přidat svého spolužáka ze své školy. Tím by se mohla zvýšit motivace uživatelů aplikaci používat nejen na procvičení, dokud neslní určitý počet bodů, ale mohli by se mezi sebou předhánět.

Další věc, která by mohla vylepšit použitelnost aplikace je místo textového vstupu pro odpověď, vytvořit množinu tlačítek, na kterých by byla správná odpověď na příklad a na zbylých tlačítkách by byly vygenerovány hodnoty tak, aby nebylo na první pohled poznat, která je správná. To by ulehčilo uživatelům spoustu času se zadáváním odpovědi z klávesnice.

Možnost dalšího rozšíření aplikace je v rámci obsahu aplikace a vývoje oblasti geometrie, která v této aplikaci není.

V dokumentu jsem zmínil, že Khan academy framework umí určitým způsobem radit, v čem spočívá chyba v zadané odpovědi. Tato funkcionalita ovšem není většinou využívána,

a není ani nijak rozvynuta. Bylo by dobré tuto funkcionalitu dodělat do takového stavu, aby nebylo těžké tyto rady zadávat do zadání příkladů.

Jelikož je aplikace cílena na mladé uživatele, bylo by velkou výhodou zaimplementovat do takovéto aplikace hlasové výstupy místo textu. Zrychlilo a zpříjemnilo by to pohyb v aplikaci, místo zdlouhavého čtení textu, což je pro žáky první třídy opravdu náročná záležitost.

Literatura

[1] PHP Manual [online]. http://php.net/manual/en/index.php.

[2] SQL Tutorial [online]. http://www.w3schools.com/sql/.

[3] Andrew J. Elliot PhD and Carol S. Dweck PhD: *Handbook of Competence and Motivation,*. The Guilford Press, 2007-07-02 [cit. 2013-05-01].

[4] David Grudl: Debugování a zpracování chyb [online]. http://doc.nette.org/cs/debugging.

[5] David Grudl: MVC aplikace a presentery [online]. http://doc.nette.org/cs/presenters.

[6] Dušan Jankovský: CSS - Kaskádové styly [online]. http://www.jakpsatweb.cz/css/.

[7] Marczewski, A.: *A Simple Introduction.* Andrzej Marczewski, 2012.

[8] Tom Chatfield: 7 ways games reward the brain [online]. http://www.ted.com/talks/tom_chatfield_7_ways_games_reward_the_brain.html.

[9] Výzkumný ústav pedagogický v Praze: Rámcový vzdělávací program pro základní vzdělávání [online]. http://www.vuppraha.cz/wp-content/uploads/2009/12/RVPZV_2007-07.pdf, 2007 [cit. 2013-05-01].

Příloha A

Obsah CD

Soubor `readme.txt` - návod na použití aplikace.