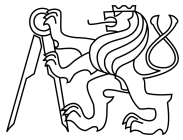
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Nástroj pro správu, sdílení a vizualizaci sportovních cvičení a tréninkových plánů
Student:	Martin Suchan
Vedoucí:	Ing. Pavel Štěpán
Studijní program:	Informatika
Studijní obor:	Web a multimédia
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

Cílem práce je webová aplikace, která bude umožňovat návrh a vizualizaci cvičení a tréninkových plánů pro sportovní trenéry s možností sdílení s ostatními.

Pokyny pro vypracování:
1. Analyzujte požadavky a podobné dostupné aplikace na trhu.
2. Navrhněte strukturu celé aplikace včetně možností sdílení.
3. Navrhněte strukturu databáze.
4. Proveďte návrh uživatelského rozhraní.
5. Pro účely tvorby vizuálních prvků navrhněte a implementujte kreslící nástroj v HTML5 a JS.
6. Implementujte aplikaci dle návrhu a důrazem na možnosti připojení nových modulů (sportů).
7. Celé řešení řádně otestujte.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 18. října 2016

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Nástroj pro správu, sdílení a vizualizaci sportovních cvičení a tréninkových plánů

Martin Suchan

Vedoucí práce: Ing. Pavel Štěpán

8. ledna 2017

Poděkování

Rád bych poděkoval vedoucímu práce, panu Ing. Pavlu Štěpánovi za ochotu, cenné rady a trpělivost. Dále bych rád poděkoval panu Miroslavu Hesovi za pomoc při vytváření zadání a konzultace k logice systému.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel dohodu, na základě níž se ČVUT vzdalo práva na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ust. § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 8. ledna 2017 .....

Odkaz na tuto práci

Suchan, Martin. *Nástroj pro správu, sdílení a vizualizaci sportovních cvičení a tréninkových plánů*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Cílem bakalářské práce je usnadnit práci trenérům, kteří připravují cvičení a tréninkové plány pro své týmy. Jak cvičení, tak i tréninkové plány mohou mezi sebou sdílet. Jednotlivá cvičení vytvářejí pomocí speciálního nástroje, který je kompatibilní ve všech prohlížečích podporujících HTML5, jak na stolních počítačích, tak v laptopech a tabletech. Práce obsahuje jednotlivé kroky analýzy, návrhu aplikace a řešení problémů, které byly objeveny až v samotném průběhu implementace. Práce také obsahuje zkušenosti z uživatelského testování aplikace odborníky a trenéry.

Klíčová slova Kreslící nástroj, sport, cvičení, tréninkové plány, testování, trenéři

Abstract

Objective of this thesis is to facilitate the work of coaches who are preparing exercises and practice plans for their teams. Drills and practice plans can be shared between coaches. Drills are created using special application, which is compatible in all browsers supporting HTML5 on desktop PC, laptops and tablets. Thesis contains individual step by step analysis, design of application and solutions to problems that were discovered during implementation. Thesis

also includes a description of real world usage of the application by experts and coaches.

Keywords Painting tool, sport, drills, practice plans, testing, coaches

Obsah

Seznam obrázků

Úvod

Práce trenéra pro dětské a dorostové kategorie většiny sportů je v dnešní době nedoceněná a nelze ji vykonávat na hlavní pracovní poměr, protože by se touto prací člověk s rodinou jen těžko uživil. To je důvod, proč téměř každý trenér těchto skupin sportovců bere trénování pouze jako koníček. Z tohoto důvodu pak nemůže svým svěřencům věnovat dostatek času a ještě k tomu mít dost času na přípravu tréninků, které jsou velmi důležité.

Tím dochází k tomu, že i když se trenéři snaží, tak bez dostatečného času na přípravu vymýšlí někdy obsah tréninku až těsně před tréninkem samotným a takový trénink velmi často ztrácí smysl. Trénink musí být dobře promyšlen, cvičení na sebe musejí správně navazovat a musí být brán zřetel na postupnou únavu sportovců.

Většina moderních trenérů se snaží svoji práci zjednodušit, aby přípravou tréninků nestrávili příliš mnoho času a i přesto byl trénink kvalitní a plnohodnotný. Na základě této poptávky vznikly různé aplikace pro počítače a tablety, které se návrh tréninků snaží usnadnit. Žádná z těchto aplikací ale nesplňuje plnou kompatibilitu mezi různými platformami. Často jsou závislé na softwaru třetích stran, mají omezenou funkčnost, nepodporují sdílení cvičení a tréninkových plánů mezi uživateli a většinou podporují pouze jeden nebo něco málo předdefinovaných sportů bez možnosti rozšíření.

Cíle práce

Cílem této práce je vytvořit online aplikaci, která umožní grafické vytváření různých cvičení. Tvorba cvičení bude probíhat ve speciálním nástroji, kde si uživatel bude vybírat z palety předdefinovaných objektů, křivek naznačujících pohyb a různých zakončení na konci křivek. Objekty bude možné přesouvat a mazat. To vše si bude trenér skládat dle potřeby na hrací plochu, kterou si vybere z nabízených hracích ploch nebo si nahraje vlastní. U každého cvičení bude možné vytvořit až čtyři nákresy s rozdílným obsahem, aby uživatel mohl

vyjádřit různé části jednoho cvičení.

Jednotlivá cvičení bude dále možné vložit do tréninkových plánů s možností jejich zobrazení na obrazovku nebo exportu do formátu PDF o velikosti stránky A4. Jak cvičení, tak i tréninkové plány budou moci uživatelé sdílet mezi sebou, a to buď mezi skupinami uživatelů, kterými jsou členy a nebo mezi všemi uživateli aplikace.

Aplikace bude přístupná pro všechny, ale uživatelé budou mít dostupné funkce podle typu oprávnění, které se určí na základě toho, jestli je či není uživatel registrovaný a zda má zaplacenou licenci. Do aplikace se bude moci registrovat každý a tím využívat některé funkce navíc oproti neregistrovanému návštěvníkovi. Pokud bude chtít uživatel používat všechny funkce aplikace, bude si muset zakoupit licenci k již registrovanému účtu.

Další funkcí aplikace bude možnost jednoduše přidávat sporty pomocí modulů. To znamená, že pokud vznikne poptávka po některém sportu, který aplikace nepodporuje, bude velmi jednoduché přidat podporu pro daný sport. V této práci bude spolu s aplikací vyvinut modul ledního hokeje.

Tato aplikace musí být kompatibilní na běžných počítačích a na tabletech. Mobilní telefony aplikaci podporovat alespoň částečně mohou, ale není to zásadní požadavek vzhledem k malé obrazovce, kde by bylo téměř nemožné vytvořit v kreslícím nástroji (dále „kreslítko") cvičení.

Kapitola 1

Analýza

Tato kapitola se zabývá analýzou existujících řešení na trhu a hledáním důvodu pro vytvoření nového řešení. Dále jsou zde uvedeny požadavky na aplikaci, jak funkční, tak i nefunkční a následně rozdělení uživatelů do různých uživatelských rolí. Asi nejdůležitější částí této analýzy je konceptuální model, podle kterého vznikne návrh jednotlivých tříd a jejich relací. Na základě získaných dat bude vybrána nejvhodnější technologie pro realizaci této práce.

1.1 Existující řešení na trhu

Na internetu se dá nalézt mnoho aplikací nabízejících databázi cvičení k různým sportům, ale jen málo takových, které umožňují uživateli současně tvořit vlastní cvičení. Mezi aplikace s databází cvičení patří například Hockey Canada Drill Hub [1] nebo Online Basketball Drills [2].

Aplikací, které nabízejí uživateli možnost tvorby vlastního cvičení je málo. Bohužel všechny nalezené aplikace až na jednu, pracují s rozšířením pro webové prohlížeče Adobe Flash Player, které není kompatibilní na většině mobilních zařízení. Testy kompatibility těchto aplikací byly provedeny na tabletech s operačními systémy Google Android 4.2.2 a Apple iOS 10, kde ani na jednom z uvedených systémů nefungovaly. Mezi tyto aplikace patří například DRILLBOOK Online [3], DRILLFY [4], HockeyShare [5] a HockeyCoachToCoach [6]. Jediná nalezená aplikace využívající HTML5 a dvě další aplikace jsou zde popsány podrobněji.

1.1.1 DRILLBOOK ONLINE

Asi nejvíce rozšířená univerzální aplikace, která je dokonce z tvorby českých programátorů, je Drillbook Online. Její návrh vytvořil stejný zadavatel, který vytvořil i zadání pro tuto práci, jenž má být svým způsobem kompatibilnější a rozšířenou verzí právě aplikace Drillbook Online. Aplikace podporuje většinu požadovaných funkcí [7]. Je tedy použitelná pro více sportů s možností při-

dat nové sporty, splňuje požadavky na možnost zakreslování vlastních cvičení a vytváření tréninkových plánů, ale neumí sdílet tréninkové plány ani cvičení pouze mezi vybranými uživateli, což je žádaná funkčnost hlavně mezi trenéry jednoho klubu, kteří aplikaci využívají.

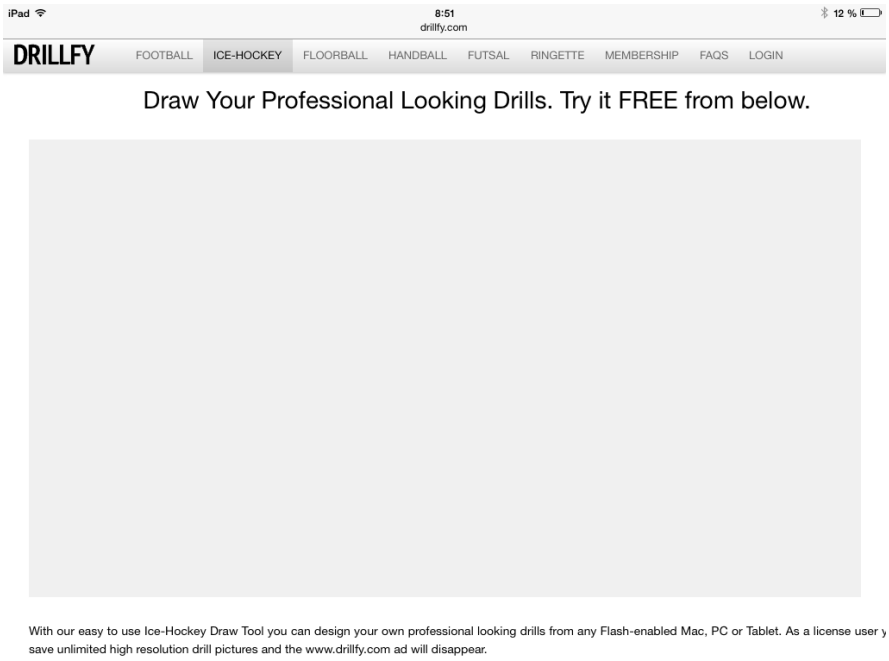
Další nevýhodou této aplikace je nekompatibilita mezi různými typy zařízení právě kvůli nutnosti používání softwaru třetích stran. Například při spuštění aplikace na mobilním zařízení s operačním systémem Apple iOS 8.3 nebo Google Android 4.2.2 se nezobrazí kreslící plocha, ale pouze infromace říkající, že internetový prohlížeč používá starou verzi programu Adobe Flash Player a odkaz na novou verzi. Po kliknutí na odkaz se stránka přesměruje na web společnosti Adobe, kde se zobrazí hláška, že pro zobrazení této stránky je potřeba zařízení, které podporuje Adobe Flash Player. Drillbook Online tedy nelze na některých platformách použít pro návrhy a zobrazení cvičení, a to ani omezeně.

1.1.2 DRILLFY

Velmi podobná aplikace jako již zmiňovaný Drillbook Online. Vizualizační prostředí pro cvičení je opět řešeno softwarem třetích stran, konkrétně pomocí softwaru Adobe Flash Player, kde vzniká naprosto shodný problém s kompatibilitou, jako u Drillbook Online. Aplikace ale na rozdíl od Drillbooku Online prezentuje, že funguje i na všech mobilních zařízeních s podporou Adobe Flash Playeru. Tato informace je ale nepravdivá, jelikož opět na tabletu s Google Android 4.2.2 a dokonce i na Apple iOS 8.3 zůstane místo kreslící plochy na obrazovce pouze šedý obdélník - viz. obrázek 1.1. Tato aplikace má ovšem velké problémy s kompatibilitou i na stolních počítačích, kde se v některých internetových prohlížečích nástroj vůbec nezobrazí. V prohlížeči Internet Explorer 11 zůstane místo nástroje pouze obdélník se šikmými černými čárami. Na rozdíl od Drillbooku Online, v této aplikaci chybí možnost sdílení cvičení a tréninkových plánů mezi uživateli v jakékoliv podobě.

1.1.3 peluu

Jediná nalezená aplikace, která používá pro své grafické návrhy HTML5, je aplikace peluu [8]. Jedná se ovšem o aplikaci, která je funkční pouze pro hokej a i přes využití HTML5 nefunguje na tabletech Apple iPad. Nelze na nich přesouvat ani mazat objekty a křivky. Dále neobsahuje možnost nastavení vlastních tréninkových plánů, což je opět požadavek většiny zájemců o tento typ aplikací. Navíc v aplikaci chybí mnoho drobných funkcí, jako například rotace objektů. Proto nelze objekty (např. hokejová brána) nastavit tak, aby byly vodorovně. Zdůvodu, že nelze vybrat jiné pozadí pro návrh a rotovat objekty, nelze zde navrhnout některá cvičení.



Obrázek 1.1: Nefungující nástroj kreslení aplikace DRILLFY

1.2 Motivace tvorby nového řešení

V předchozích odstavcích byly rozebrány dvě nejúspěšnější a jedno podobné online řešení aplikace pro přípravu cvičení, případně tréninkových plánů. Při této analýze bylo zjištěno, že v podstatě neexistuje žádná aplikace, která by byla dostupná jak z počítače, tak ze všech rozšířených platforem pro tablety, kterými jsou Android, iOS a Windows 8.

Právě toto zjištění vedlo k názoru, že by bylo vhodné udělat aplikaci, která bude nejen dostupná z většiny mobilních zařízení, ale také odstraní některé nedostatky, jako například nemožnost sdílení cvičení a tréninkových plánů pouze ve skupinách uživatelů, které budou představovat jednotlivé sportovní kluby.

1.3 Požadavky na aplikaci

1.3.1 Funkční požadavky

Funkční požadavky spadají do funkční analýzy a označují, jaké úkony, aktivity a akce je potřeba vytvořit. [9]

F.1 - správa cvičení

- všem registrovaným uživatelům s platnou licencí bude umožněno vytvářet, upravovat a mazat vlastní cvičení
- každé cvičení může mít připojené jedno video uložené v aplikaci YouTube nebo v aplikaci Vimeo
- bude možné komentovat všechna cvičení, která mají jednotliví uživatelé oprávnění zobrazit

F.2 - správa vlastních tréninkových plánů

- registrovaní uživatelé s platnou licencí budou moci vytvářet, upravovat (přidávat či odebírat cvičení) a mazat své tréninkové plány

F.3 - přizpůsobení kreslítka

- obsah kreslítka bude mít každý registrovaný uživatel s platnou licencí přizpůsobený dle vlastních požadavků. V nastavení si vybere objekty, křivky označující pohyb a zakončení, které se mu zobrazí v paletě nástrojů
- uživatel si v nastavení vybere, která pozadí se mu nabídnou při tvorbě nového nákresu

F.4 - vyhledávání cvičení, tréninkových plánů a filtrování výsledků

- každý uživatel může vyhledávat mezi cvičeními a tréninkovými plány, které jsou s ním sdíleny
- výsledky vyhledávání může filtrovat a řadit dle několika základních kritérií
- všechny nalezené výsledky se zobrazí na jednotlivých stránkách, mezi nimiž bude možno přecházet

F.5 - sdílení vlastních cvičení a tréninkových plánů

- registrovaný uživatel si bude moci vybrat, jestli jím vytvořená cvičení a tréninkové plány bude sdílet s uživateli ze skupin, v nichž je členem, se všemi uživateli aplikace a nebo je sdílet nebude vůbec

F.6 - správa skupin

- registrovaný uživatel s platnou licencí může pracovat se skupinami, a to následovně:
  - vytvářet nové skupiny

  - upravovat a mazat vlastní skupiny
  - být pozván do uzavřené skupiny
  - vstupovat do veřejných skupin
- každý uživatel může být členem maximálně 10 skupin, bez ohledu na to, jestli je jejím vlastníkem nebo pouze členem

F.7 - správa vlastního profilu

- registrovaný uživatel bude mít možnost měnit většinu dat ve svém profilu. Mezi tato data patří:
  - heslo
  - e-mail
  - jazyk dat a jazyk uživatelského rozhraní
  - preferovaný formát data
  - preferované věkové kategorie
  - fakturační údaje

1.3.2 Nefunkční požadavky

Na nefunkční požadavky musí myslet každý IT architekt při návrhu systému, pokud chce, aby jeho software byl kvalitní a konkurenceschopný. Jedná se o požadavky, mezi které patří především výkon, škálovatelnost, spolehlivost, rozšiřitelnost, udržitelnost, spravovatelnost a bezpečnost. [10]

Zde jsou uvedeny nefunkční požadavky této práce.

N.1 - Podpora webové aplikace

- podpora HTML5, CSS3 a knihovny jQuery verze 1.10.2
- design webových stránek bude situován do zelené, bílé a odstínů šedi

N.2 - Výkon

- načítání všech stránek, kromě první, bude probíhat pomocí AJAXového řešení, které bude méně zatěžovat připojení a urychlí dobu načítání stránek

N.3 - Uživatelské rozhraní

- statický design pro rozlišení o šířce 1280 pixelů. Toto rozlišení bylo vybráno jako optimální proto, protože levnější laptopy, které se dnes prodávají, mají na šířku často 1366 pixelů a více a na běžných tabletech je na šířku minimálně 1024 pixelů. Tím došlo ke kompromisu, kdy na tabletech bude stránka lehce zmenšena, aby byla přes celou obrazovku a na laptopech bude po stranách vznikat nevyužitý prostor

- funkčnost na tabletech

N.4 - Bezpečnost

- systém bude zabezpečený proti útokům, jako je například SQL injection, XSS útok a CSRF
- hesla uživatelů budou uložena v databázi tak, že dojde k "osolení" hesla a následnému zahashování. Osolení hesla znamená, že k původnímu řetězci s heslem se přidá před jeho zahashováním jiný řetězec, který je unikátní pro každého uživatele. Zahashování označuje proces, při kterém se heslo změní v posloupnost písmen a čísel, ze kterých není možné zjistit původní heslo

N.5 - Kompatibilita

- na rozšířených internetových prohlížečích
  - Internet Explorer verze 10.0 a výše
  - Mozilla Firefox verze 33.0 a výše
  - Google Chrome verze 40.0 a výše
  - Opera od verze 27.0 a výše
- na rozšířených mobilních platformách v tabletech
  - Google Android verze 4.2.2 a výše
  - Apple iOS verze 8.0 a výše
  - Microsoft Windows verze 8.1 a výše

1.4 Uživatelské role

Přístup na samotné webové stránky aplikace bude veřejný. Uživatel se zde bude moci zaregistrovat a přihlásit, čímž získá některé funkce navíc oproti nepřihlášenému uživateli. Po registraci si bude moci zakoupit licenci, díky které již získá plný přístup do systému.

Dále jsou uvedeny úrovně oprávnění podle přístupu.

1.4.1 Neregistrovaný uživatel

Uživatel, který pouze zavítal na stránky aplikace a nemá zájem se registrovat, si může prohlížet veřejně sdílená cvičení a tréninkové plány ostatních registrovaných uživatelů. Mezi takto sdílenými cvičeními a tréninkovými plány může vyhledávat a používat k vyhledávání jednoduchou filtraci. Také si může prohlížet stránky s kontakty a informacemi o systému a může se bezplatně zaregistrovat.

1.4.2 Registrovaný uživatel bez licence

Pokud se uživatel zaregistruje a nezakoupí si licenci, získá oproti neregistrovanému uživateli několik funkcí navíc. Zpřístupní se mu možnost připnout si kterákoliv veřejně sdílená cvičení a tréninkové plány ostatních uživatelů do oblíbených položek, kde bude moci mít připnuto až 10 cvičení a 10 tréninkových plánů. Další funkcí navíc bude možnost vytvořit si jeden vlastní tréninkový plán ze sdílených cvičení ostatních uživatelů.

1.4.3 Registrovaný uživatel s licencí

V případě, že si registrovaný uživatel zakoupí roční licenci, získá neomezený přístup do systému. Ten bude zahrnovat vytváření vlastních cvičení a tréninkových plánů v neomezeném množství, které bude moci sdílet s ostatními, a to buď veřejně nebo v uzavřených skupinách. Bude moci být členem až 10 skupin, které může sám vytvořit nebo být do nich pozván v případě uzavřených skupin a nebo se připojit do existujících skupin v případě, že se jedná o veřejné skupiny.

Další funkce si registrovaný uživatel s licencí bude moci nastavit dle svého uvážení v nastavení aplikace. Bude se jednat například o objekty, křivky naznačující pohyb a zakončení křivek, ze kterých si bude vybírat, jaké chce zobrazit v paletě kreslítka a jaké ne. Stejně tak mu bude umožněno, aby si vybral z přednastavených obrázků hřišť, které budou sloužit jako podklad pro kreslítko a nebo si nahrál obrázky vlastních hřišť. Vlastní obrázky budou ale omezeny počtem, aby některý uživatel nepoužíval aplikaci jako databanku pro fotografie.

1.5 Případy užití

Pro znázornění případu užití byl vytvořen a detailně popsán diagram, který pokrývá dva funkční požadavky. Jedná se o případ užití, který popisuje veškerou práci s cvičeními, tedy jejich vytváření, úpravu, mazání a komentování, tak i jejich vyhledávání a filtrování výsledků.

Diagram případu užití - viz. obrázek 1.2.

UC1 - vyhledávání mezi cvičeními

- pokud chce uživatel zobrazit různá cvičení, která splňují jeho požadavky, nastane tento UC
- nejprve klikne uživatel na položku “Cvičení” v horním menu
- vyplní filtraci pro vyhledání cvičení tak, aby byla nalezena požadovaná cvičení a stiskne “Vyhledat”
- zobrazí se mu seznam všech cvičení splňujících jeho požadavky

UC2 - zobrazení náhledu cvičení

- tento UC nastane, pokud bude chtít uživatel zobrazit náhled cvičení, kde si může cvičení buď prohlédnout nebo s ním dále pracovat
- nejprve provede vyhledání cvičení podle UC1
- ze získaného seznamu cvičení si vybere to, o které má zájem a klikne na náhled nebo název cvičení
- nyní se nachází na stránce s náhledem cvičení, kde si může cvičení buď prohlédnout nebo s ním dále pracovat

UC3 - vytvoření nového cvičení

- jakmile se uživatel rozhodne, že vytvoří nové cvičení, nastává tento UC
- klikne v menu na položku "Cvičení"
- vybere "Nové cvičení"
- na stránce s novým cvičením provede požadované úpravy
- uloží cvičení kliknutím na tlačítko "Uložit"

UC4 - úprava vlastního cvičení

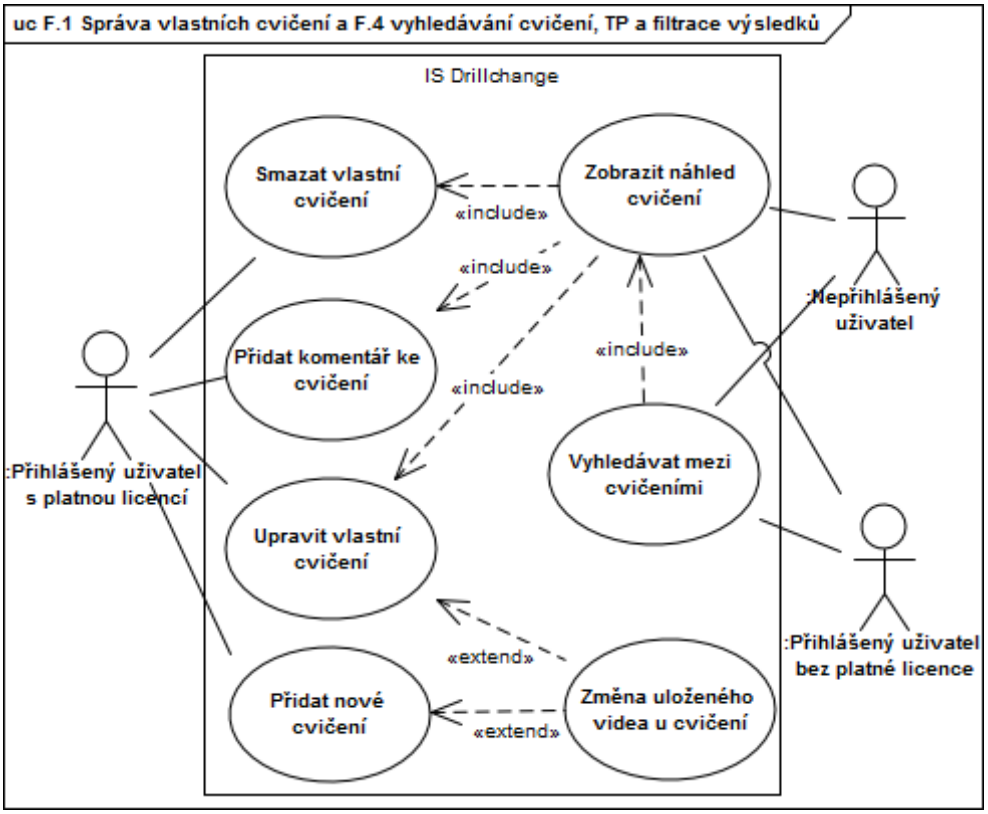
- k tomuto UC dojde v případě, že chce uživatel upravit již existující cvičení
- uživatel provede scénář podle UC2
- na stránce s náhledem cvičení stiskne tlačítko "Upravit"
- provede náležité úpravy
- uloží cvičení kliknutím na tlačítko "Uložit"

UC5 - smazání vlastního cvičení

- v případě, že se uživatel rozhodne smazat vlastní cvičení, dojde k tomuto UC
- uživatel provede scénář podle UC2
- v tuto chvíli se nachází na stránce s náhledem cvičení, kde stiskne tlačítko "Upravit"
- následně stiskne tlačítko "Smazat"
- potvrdí smazání

UC6 - komentování cvičení

- pokud bude mít uživatel dostatečná oprávnění k zobrazení cvičení a bude mít platnou licenci, bude moci ke cvičení přidat komentář. Jestliže tak bude chtít učinit, nastane tento UC



Obrázek 1.2: Případ užití správy cvičení

- nejprve uživatel provede scénář podle UC2
- na stránce s náhledem vybere v bloku s informacemi o cvičení záložku “Komentáře”
- v tuto chvíli vidí všechny komentáře u cvičení a může přidávat nové komentáře

1.6 Analytický doménový model aplikace

Pomocí analytického doménového modelu se vytvoří základ pro design, ze kterého se následně čerpá při vytváření databázového modelu 2.4 a modelu tříd 2.3.

Samotný model popisuje data, vazby mezi entitami a indetifikaci stavů entit. [11]

Při vytváření analytického doménového modelu bylo dbáno na to, aby vynikly veškeré vazby mezi uživateli, cvičeními a tréninkovými plány. Zároveň byla zahrnuta i možnost sdílení cvičení a tréninkových plánů a jejich komentování uživateli.

Analytický doménový model aplikace je možné vidět na obrázku 1.3.

1.7 Výběr vhodných technologií

Na základě předchozí analýzy, pomocí které byly získany důležité informace ohledně požadavků a funkčnosti aplikace, došlo k výběru následujících technologií pro implementaci aplikace.

1.7.1 HTML5

HyperText Markup Language neboli HTML je jazyk vytvořený pro snadnou tvorbu webových stránek. Aplikace bude využívat jeho poslední stabilní verzi s označením HTML5, která umožňuje využívání prvku canvas. V tomto prvku je možné dělat různé grafické operace bez nutnosti využívání softwaru třetích stran.

Verze HTML5 je momentálně dostupná ve všech zařízeních a prohlížečích, pro které je potřeba dodržet kompatibilitu.

1.7.2 CSS3

Kaskádové styly neboli CSS označují v informatice jazyk, který popisuje, jak se má jaký element zobrazit na výsledné HTML stránce. Bez kaskádových stylů je velmi obtížné, někdy téměř nemožné, upravovat uživatelské rozhraní tak, aby vypadalo dle představ a potřeb navrženého designu.

Tato aplikace bude využívat kaskádové styly ve verzi 3, které oproti starším verzím umožňují například plynulé animace, zaoblené rámečky nebo barevné přechody.

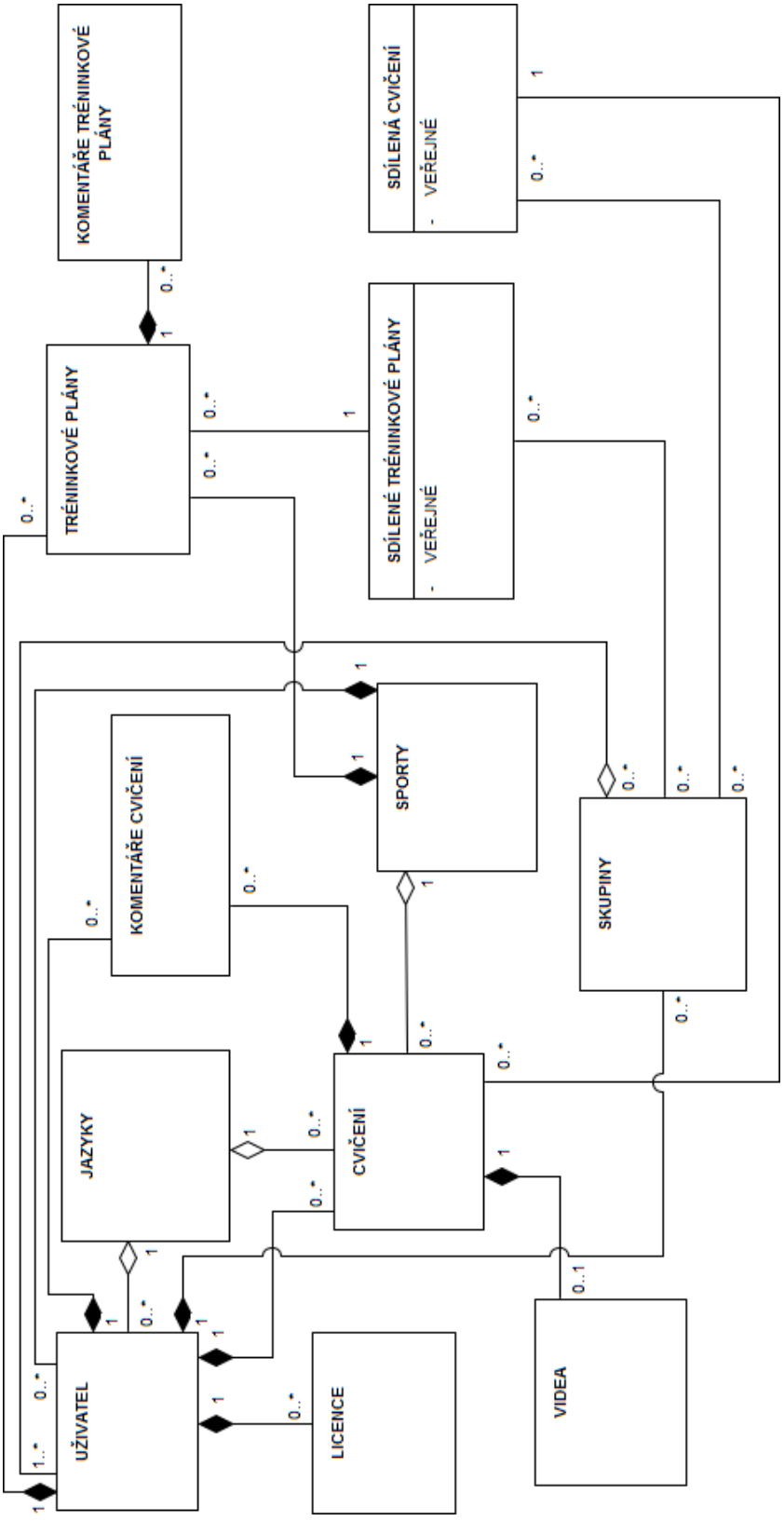
1.7.3 JavaScript

JavaScript je objektově orientovaný skriptovací jazyk, který spouští své skripty v prohlížeči na straně uživatele. JavaScript je nutné použít kvůli ověřování dat ve vstupech, vyplněných uživatelem, plynulým efektům na stránce a ovládání kreslítka.

Pomocí JavaScriptu a technologie AJAX bude probíhat načítání potřebného obsahu webových stránek bez nutnosti načítání celé stránky. Tím dojde k úspoře množství přenesených dat a tím i k vyšší rychlosti načítání celé aplikace.

1.7.4 Knihovna jQuery 1.10.2

Knihovna jQuery je populární JavaScriptová knihovna, která usnadňuje práci s JavaScriptem samotným. Zkracuje a usnadňuje syntaxi jazyka a pomocí několika metod přidává mnoho základních funkcí, jako například plynulé posouvání nebo skrývání objektů nebo velice jednoduché AJAXové řešení.



Obrázek 1.3: Analytický doménový model aplikace

jQuery je poskytováno pod licencí MIT. Všechny knihovny, které jsou vydány pod touto licencí je možné svobodně použít v jakékoliv aplikaci, a to pouze s podmínkou, že text licence MIT je dodáván společně s aplikací. [12]

1.7.5 Knihovna jQuery UI 1.11.2

Další knihovnou je knihovna jQuery UI. Tato knihovna rozšiřuje knihovnu jQuery o několik velmi zdařilých funkcionalit, z nichž v této aplikaci budou potřeba hlavně záložky, mezi kterými si bude moci uživatel vybírat v nastavení.

jQuery UI je stejně jako jQuery poskytováno pod licencí MIT.

1.7.6 Knihovna kineticJS 5.1.0

I knihovna kineticJS je JavaScriptovou knihovnou. Tato knihovna usnadňuje práci s elementem canvas v HTML5. Umožňuje vytvořit kreslící kontejner, ve kterém jsou prvky uloženy v jednotlivých vrstvách a je velmi jednoduché je editovat nebo mazat oproti obyčejnému HTML5 bez využití jakékoliv knihovny.

Sama knihovna obsahuje různé objekty, které umí vykreslit bez nutnosti detailních definic, například kruh, obdélník nebo křivky. U takovýchto prvků z knihovny lze velmi jednoduše nastavit velikost, barvu a pozici. Pokud je to nutné, je možné nahrát nový prvek pomocí obrázku. Ten ale neumožňuje stejné množství úprav jako u předdefinovaných objektů.

Všechny vrstvy lze samostatně nebo hromadně exportovat do formátu JSON, který je možné si uložit do databáze a při příštím otevření editace cvičení předat tento JSON řetězec zpět do knihovny, která načte cvičení takové, jaké bylo při opuštění.

Knihovna kineticJS je stejně jako předchozí dvě knihovny poskytována pod licencí MIT.

1.7.7 PHP 5.3

PHP je skriptovací jazyk, který se spouští na straně serveru. V něm probíhá nahrávání dat z databáze, jejich následné zpracování a vygenerování HTML stránky, která se okamžitě odešle na obrazovku uživatelova počítače. Dále v PHP probíhá opětovná kontrola všech vstupů vyplněných uživatelem před uložením do databáze a ochrana proti většině útoků.

Využíváním PHP nevznikají žádná omezení na aplikaci, jelikož je PHP pod licencí "PHP License v3.01". [13]

1.7.8 MySQL 5.5.42

Jedná se o velmi rozšířený databázový systém, který podporuje různé způsoby ukládání dat. Tato aplikace bude využívat způsob innoDB, který je ideální pro

tabulky se vzájemnými relacemi.

Omezení plynoucí z licence MySQL má vliv na licenci aplikace pouze v případě, kdy je součástí aplikace MySQL server. Tato aplikace MySQL server neposkytuje, je tedy nutné aplikaci nainstalovat na server, kde již MySQL databáze funguje. Tím nevznikají žádná licenční omezení pro aplikaci.

1.8 Bezpečnostní rizika

Jelikož se jedná o online aplikaci, která bude v určitém omezení dostupná pro každého člověka s přístupem na internet, tak je snadné se pokusit zorganizovat různé útoky. Tyto útoky by mohly aplikaci poškodit nebo vyřadit z činnosti. Proti nejčastějším útokům musí být aplikace zabezpečena.

1.8.1 SQL injection

Jedná se o typ útoku, kdy útočník vyplní do některého z uživatelských vstupů - například do přihlašovacího formuláře - taková data, která se pokusí narušit následně vytvořený SQL dotaz aplikace na databázi. Pokud nejsou v aplikaci ošetřené vstupy od uživatele, může pak útočník dělat téměř libovolné úpravy v databázi a velmi jednoduše poškodit celý systém nebo si třeba přidat maximální oprávnění.

Proti tomuto útoku se bude aplikace bránit dostatečným ošetřením vstupů od uživatele před jejich použitím. Stačí na všechny vstupy od uživatele použít PHP funkci *htmlspecialchars*, která pozmění nebezpečné znaky, jako například uvozovky, na jejich HTML kód. V SQL dotazech je ještě zapotřebí všechna tato data ohraničovat uvozovkami.

1.8.2 XSS - cross-site scripting

Další útok, ve kterém se spoléhá na neošetřené vstupy od uživatele nebo neošetřené vstupy, které se přenášejí přes URL adresu webu. Pomocí takových vstupů dostane útočník do stránky JavaScriptový kód, kterým může opět poškodit stránky nebo získávat citlivá data uživatelů. [14] Tento útok se dělí na tři různé typy:

Typ 1
: Tento typ popisuje útok, kdy útočník upraví data, která se předávají v URL adrese a následně se využívají JavaScriptem jako proměnné, které se ukládají na statickou stránku. Nejlepší obranou je nepoužívat v JavaScriptu data z URL tak, aby se promítala na stránku a tímto způsobem se bude aplikace vůči útoku bránit.

Typ 2
: Je velmi podobný útoku typu 1, ale data se z URL nezpracovávají v JavaScriptu, ale v PHP, odkud se opět vytisknou v neošetřeném stavu

na stránku. Obranou je ošetřování vstupů, aby uživatel nemohl vkládat žádné HTML ani JavaScriptové syntaxe a nebo se tyto syntaxe převedly na neškodný text a tím se nevykonaly. V aplikaci se bude používat již výše zmíněná funkce *htmlspecialchars*, která bezpečně převede i znaky < a >, které jsou právě v tomto ohledu nebezpečné.

Typ 3

Poslední typ útoku XSS spoléhá na neošetřená data ze vstupů, která se ukládají do databáze a následně se zobrazují ostatním návštěvníkům, například komentáře. Pokud se útočníkovi podaří uložit svůj JavaScriptový kód do komentáře a aplikace neošetřuje vstupy od uživatele, pak se všem uživatelům, kteří takový komentář uvidí, spustí v prohlížeči JavaScriptový kód napsaný útočníkem. Nejlepší obranou je opět ošetření vstupů stejně jako u typu 2. Tentokrát ale před uložením do databáze.

1.8.3 CSRF - cross-site request forgery

Jedná se o útok, kdy se útočník pokusí neopravitelně poškodit data v databázi. K tomu potřebuje znát URL adresy aplikace, do kterých má přístup přihlášený uživatel s oprávněním provádět změny. Na základě těchto informací vytvoří stránku, kterou zašle oprávněnému uživateli a nechá ho, aby ji vědomě nebo nevědomě spustil a tím se provedly změny v aplikaci, aniž by o tom daný uživatel či kdokoliv jiný věděl.

Proti tomuto útoku se dá bránit autorizačním tokenem. Jde o náhodně generovaný token pro všechny formuláře, které upravují nebo mažou data v aplikaci. Následně před provedením dotazu na databázi dojde k ověření, že vygenerovaný token je správný a tím dojde k nemožnosti provést CSRF útok. [15] Přesně tímto způsobem bude aplikace imunní vůči CSRF útoku.

1.8.4 Rainbow tables a solení hesla

V každé alespoň trochu zabezpečené aplikaci se ukládají hesla do databáze v zahashované podobě, aby útočník, který by se do databáze dostal, nezískal okamžitě přístupy všech uživatelů. Takový útočník má v danou chvíli pouze hashe hesel uživatelů, které pro něj nejsou tak cenné. Může ovšem použít takzvanou "duhovou tabulku". Jde o tabulku, ve které jsou předem vytořené hashe různých řetězců a útočníkovi poté stačí najít, jestli některý hash v této tabulce je shodný s hashem hesla uživatele a tím získá řetězec, který má stejný hash. Následně může vyplnit takový řetězec místo hesla do aplikace a i když je řetězec jiný, než uživatelovo reálné heslo, hash má stejný a dostane se do aplikace.

Proti tomuto útoku se dá bránit takzvaným solením hesla. Zde je k heslu před zahashováním přidán jiný řetězec, který je unikátní pro každého uživatele. Tím je útočníkovi hash z databáze téměř k ničemu. Aplikace bude

k heslu uživatele před zahashovaním přidávat nastavený statický text a uživatelovu e-mailovou adresu. Jelikož změna e-mailové adresy bude vyžadovat pro potvrzení heslo, nebude problém hash hesla při změně e-mailové adresy znovu vygenerovat.

Kapitola 2

Návrh

Kapitola návrh je zaměřena na výběr vhodné architektury a její popis. Následně je zde navržen model tříd, který je velmi důležitý pro celkový návrh aplikace. V databázovém modelu je probrána struktura databáze s promyšlenými jednotlivými tabulkami, jejich atributy a vzájemnými relacemi. Neméně důležitými jsou návrh uživatelského rozhraní a návrh struktury modulů jednotlivých sportů.

2.1 Architektura aplikace

Před samotnými návrhy aplikace je důležité vybrat vhodnou architekturu, podle které bude celá aplikace pracovat. Jelikož se systém bude svým typem řadit mezi takzvané aplikace CRUD, byla vybrána dvouvrstvá architektura. Většina náročnějších akcí, hlavně kolem kreslítka, bude prováděna na prezenční vrstvě v javascriptu a následně odesílána do PHP pomocí AJAXu. V PHP bude probíhat pouze získání dat z databáze a případně jejich zpracování do databáze. To znamená, že systém bude fungovat na prezenční a datové vrstvě.

2.1.1 CRUD aplikace

Jedná se o typ aplikace, která pro svoji práci s daty využívá jen čtyři základní operace. [16]

Create - označuje vytvoření nového záznamu, v SQL dotazu odpovídá akci *INSERT*

Read - jde o přečtení uložených dat v databázi, v SQL dotazu odpovídá akci *SELECT*

Update - slouží k provedení změn na již existujících záznamech, v SQL dotazu odpovídá akci *UPDATE*

Delete - značí možnost mazání uložených záznamů, v SQL dotazu odpovídá akci *DELETE*

2.2 Architektura prezenční vstvy

Jako nejvhodnější architektura pro prezenční vrstvu byla vybrána architektura MVC (Model-view-controller).

2.2.1 MVC - Model-view-controller

Architektura MVC vytváří z prezenční vrstvy 3 logické části, které je možné samostatně upravovat a takovéto úpravy mají na ostatní části minimální dopad. Jak je patrné z názvu, těmito částmi jsou model, view a controller. [17]

Jednotlivé části jsou dále popsány vzhledem k aplikaci, a to konkrétně na příkladu, jak probíhá načítání stránek pomocí AJAXu.

2.2.2 Model

Všechna data uložená na stránce, jako je menu, obsah stránky, postranní panel a spodní panel, jsou uloženy v modelu. Ten uchovává data hned po vytvoření, při jejich úpravě až po smazání. Pokud uživatel přejde na jinou stránku aplikace, pak controller upozorní model na tuto skutečnost. V tu chvíli model upraví svá data tak, aby odpovídala požadované stránce.

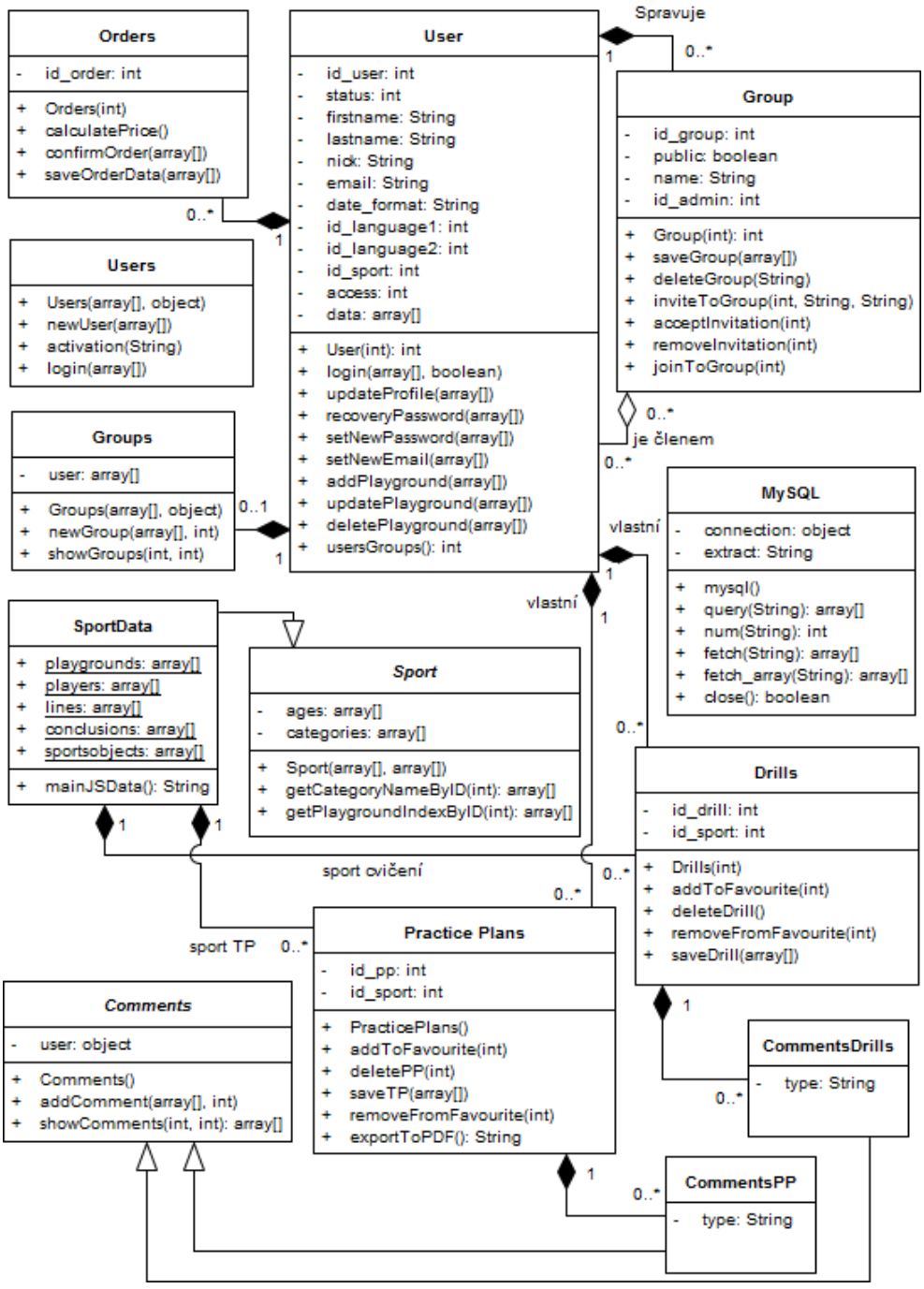
2.2.3 View

Část architektury nazvaná view se stará o to, aby data, která získá z modelu byla zobrazena uživateli na obrazovce. V našem konkrétním případě tedy půjde o to, aby se uživateli zobrazila webová stránka a na ní data, která jsou uložena v modelu.

Nejdříve dojde k počáteční inicializaci a k dalšímu překreslení bude docházet pouze tehdy, jestliže takovou skutečnost vyvolá controller při zachycení události, například při přechodu na jinou stránku.

2.2.4 Controller

Poslední částí architektury je controller. Ten se stará o to, když například uživatel klikne na odkaz na stránce, tak aby byl o této skutečnosti informován model, který obnoví svá data a zároveň při tom je upozorněn i view, který překreslí stránku s novými daty z modelu.



Obrázek 2.1: Návrhový model tříd

2.3 Návrhový model tříd

Na rozdíl od analytického doménového modelu 1.6 je návrhový model tříd zaměřem na návrh podle vybraného programovacího jazyka. Na základě takovéhoto návrhu by měl být programátor schopen naprogramovat funkční aplikaci.

Návrhový model tříd této aplikace tvoří celkem třináct tříd, z nichž dvě nemají žádné vztahy vůči ostatním třídám. Tyto dvě třídy slouží spíše pro zpřehlednění zdrojového kódu. Celý diagram návrhového modelu tříd je vidět na obrázku 2.1. Veškeré názvy tříd, metod a atributů jsou z důvodu přehlednější syntaxe pojmenované v anglickém jazyce.

Mnoho metod v návrhu přijímá hlavně parametr typu *array[]* - tedy pole prvků. Jedná se o metody, které jsou volány po odeslání formulářů uživatelem na server, kde jsou data z formulářů uložena právě v tomto poli.

Zde je ještě uveden seznam jednotlivých tříd s popisem, k čemu slouží.

2.3.1 Třída User

Každá instance této třídy reprezentuje jednoho uživatele a správu jeho dat.

V atributech se nacházejí veškerá data o uživateli. Ta, která jsou často používána, jsou uložena v samostatných atributech. Jedná se o unikátní identifikační číslo, jméno, přijmení, status, e-mailovou schránku, přezdívku viditelnou pro ostatní, formát dat, nastavený jazyk uživatelského prostředí, nastavený jazyk pro vyhledávání dat (cvičení, tréninkové plány), identifikační číslo vybraného sportu a úroveň oprávnění pro přístup do systému. Zbývající, méně důležitá data, jsou uložena v poli prvků nazvaném *data*.

Konstruktor třídy přijímá jednu číselnou proměnnou, která označuje unikátní identifikační číslo uživatele. Na základě tohoto identifikátoru se nahrají z databáze do instance třídy ostatní atributy, ve kterých jsou uložena data uživatele.

Základní metodou této třídy je metoda *login*, která provede přihlášení uživatele na základě přijatých dat.

Další metody se starají o správu uživatelských dat, převážně o úpravy v základním profilu (*updateProfile*) včetně změny hesla (*setNewPassword*) a nastavení nové e-mailové adresy (*setNewEmail*). Uživatel si bude moci nahrávat vlastní hrací plochy pro cvičení, ze kterých si bude vybírat ty, které chce zobrazit na stránce s úpravou cvičení a k tomu slouží metody *addPlayground*, *updatePlayground* a *deletePlayground*.

Metoda *recoveryPassword* zpracovává data pro obnovu hesla, v případě jeho zapomenutí a poslední metoda *usersGroups* zjistí a vrátí počet skupin, v kolika je uživatel vlastníkem nebo členem.

2.3.2 Třída Group

Instance této třídy reprezentují jednotlivé skupiny, které spravují uživatelé a stávají se jejich členy. Samotná třída se bude starat o veškerou obsluhu dané skupiny včetně všech možných akcí nabízených uživatelům.

Atributy této třídy jsou veškeré informace o skupině, tedy její identifikátor, název, správce a viditelnost skupiny.

Metody se starají o úpravu a mazání skupiny. Zároveň jsou zde i metody pro pozvání nových členů do skupiny, přijetí pozvání, odstranění pozvání a připojení se do veřejné skupiny.

Třída *Group* má dva vztahy s třídou *User*, kde jeden vztah označuje členství uživatelů ve skupině a druhý vztah kdo je správcem skupiny.

2.3.3 Třída Drills

Z této třídy se vytvářejí objekty reprezentující jednotlivá cvičení.

Jedinými atributy zde jsou identifikátor cvičení a identifikátor sportu, kterého se cvičení týká.

Konstruktor této třídy přijímá jednu číselnou hodnotu, označující identifikátor cvičení, na základě kterého se vytvoří objekt. Další metody *saveDrill* a *deleteDrill* ukládají nová cvičení, upravují stávající a případně je odstraňují.

Poslední dvě metody *addToFavourite* a *removeFromFavourite* přidávají nebo odstraňují cvičení z oblíbených cvičení uživatele, jehož identifikátor přijímají v jediném parametru.

Třída má celkem tři vztahy. Jeden s třídou *Users*, znázorňující vlastníka cvičení. Další pak s třídou *Comments Drills*, který každému komentáři určuje, k jakému cvičení patří a poslední vztah je s třídou *Sport* označující, do jakého sportu cvičení spadá.

2.3.4 Třída Orders

Tato třída znázorňuje jednotlivé objednávky uživatele. Jedná se o velmi jednoduchou třídu, která se stará pouze o ukládání dat k objednávkám (*saveOrderData*) a provedení nutných akcí po potvrzení objednávky (*confirmOrder*), jako například odeslání e-mailu uživateli a výpočet finální ceny objednávky.

Přepočítávání ceny objednávky bude probíhat během všech kroků a hlavně před uložením finální ceny, aby se nestalo, že uživatel pošle na server podvržená data z formuláře objednávky a systém je bez kontroly pouze uloží.

Jediný vztah této třídy je s třídou *User*, který označuje vlastníka dané objednávky.

2.3.5 Třída PracitcePlans

Jedná se o velmi podobnou třídu, jako je třída *Drills*. Jejich hlavním rozdílem je, že tato třída se nestará o cvičení, ale o tréninkové plány. Atributy a me-

tody jsou shodné, kromě přidané metody *exportToPDF*, která provede export tréninkového plánu do formátu PDF, aby byl pro uživatele snadno přenosný a tisknutelný.

2.3.6 Abstraktní třída Comments, třída CommentsDrills a třída CommentsPP

Abstraktní třída *Comments* je třídou, která obsahuje potřebné metody pro práci s komentáři. Jde o možnost přidání komentáře (*addComment*) a získání dat všech komentářů k danému cvičení (*showComments*). Obě tyto metody jsou napsané obecně tak, aby podle atributu *type* zvolily, jestli se jedná o komentáře ke cvičení nebo o komentáře k tréninkovým plánům.

Tento atribut se nastavuje v třídách *CommentsDrills* a *CommentsPP*, které jsou přímými potomky abstraktní třídy *Comments*. Tyto třídy samy o sobě neobsahují nic jiného, než specifikaci, o jaký typ komentářů se jedná.

2.3.7 Abstraktní třída Sport a její potomek třída SportData

Další abstraktní třída s názvem *Sport* uchovává všechny atributy a metody, které je nutné doplnit v třídě potomka a jsou nezbytné pro práci se sportem.

Atributy této třídy jsou *ages* a *categories*. Atribut *ages* je dvourozměrné pole prvků, ve kterém jsou uloženy všechny věkové kategorie pro daný sport. Druhý atribut *categories* obsahuje také dvourozměrné pole prvků, v němž jsou pro změnu uloženy všechny kategorie daného sportu.

Konstruktor této třídy přijímá v parametrech dvě pole prvků, která se následně uloží do výše zmíněných atributů. Mezi metodami jsou běžné *get* a *set* metody a k nim navíc dvě další metody. První s názvem *getCategoryNameByID* vrací název kategorie podle zadaného identifikátoru. Druhá metoda *getPlaygroundIndexByID* vrací pouze index hracího pole podle jeho ID.

Dědicem třídy *Sport* je třída *SportData*, jejíž obsah se liší podle modulu sportu. Více o modulech sportů je uvedeno v kapitole 2.8. Důležité je, že každá taková třída obsahuje statické atributy *playgrounds*, *players*, *lines*, *conclusions* a *sportobjects*, kde každý tento atribut obsahuje určitá data o možných podkladech pro hřiště, obrázky hráčů, typy křivek, typy zakončení a obrázky pro objekty, jakými mohou být například brány nebo kužele.

2.3.8 Třída Groups

Třída Groups se stará o vytvoření nové skupiny a vytvoření seznamu skupin zobrazených uživateli. Obsahuje pouze dvě metody a konstruktor. První z těchto metod (*newGroup*) obstarává ukládání nových skupin a druhá metoda (*showGroups*) vrací dvojrozměrné pole s informacemi o všech skupinách, podle parametrů, které metoda přijme. Jako první parametr je identifikátor uživatele, který o seznam skupin žádá a druhý parametr označuje vlastnosti skupin, které má metoda vrátit.

2.3.9 Třída Users

Třída *Users* je první třídou, která nemá žádné vazby na ostatní třídy. Jde o třídu, která se stará o vyhledávání uživatele po zadání přihlášovacích údajů, které následně odešle do nové instance třídy *User*. Stará se také o registraci nových uživatelů včetně všech potřebných formalit, jako odesílání e-mailových zpráv s aktivačním odkazem a o případnou aktivaci uživatele.

2.3.10 Třída MySQL

Druhá třída bez vztahu k ostatním třídám, která zajišťuje a usnadňuje dotazy na databázový server. Její konstruktor vytvoří nové připojení na databázi a následně její metody provádějí různé typy dotazů na databázi. Její hlavní účel je zjednodušení syntaxe dotazů ve zdrojovém kódu.

2.4 Konceptuální databázový model

Konceptuální databázový model reprezentuje, jaké budou tabulky a jejich atributy v databázi a jaké mezi sebou budou mít vzájemné vazby. Dále jsou zde určeny primární klíče a cizí klíče tabulek.

Primární klíč označuje atributy neboli sloupce, které mají všechna data v tabulce unikátní a neexistují žádné dva záznamy, které by měly všechny atributy, které jsou součástí primárního klíče, shodné.

Cizí klíče znamenají, že tabulka, která přijímá klíč od jiné tabulky, považuje tento klíč za součást svého primárního klíče.

Mezi entitami vznikají tři různé typy vztahů, jednotlivé vztahy jsou zde stručně popsány.

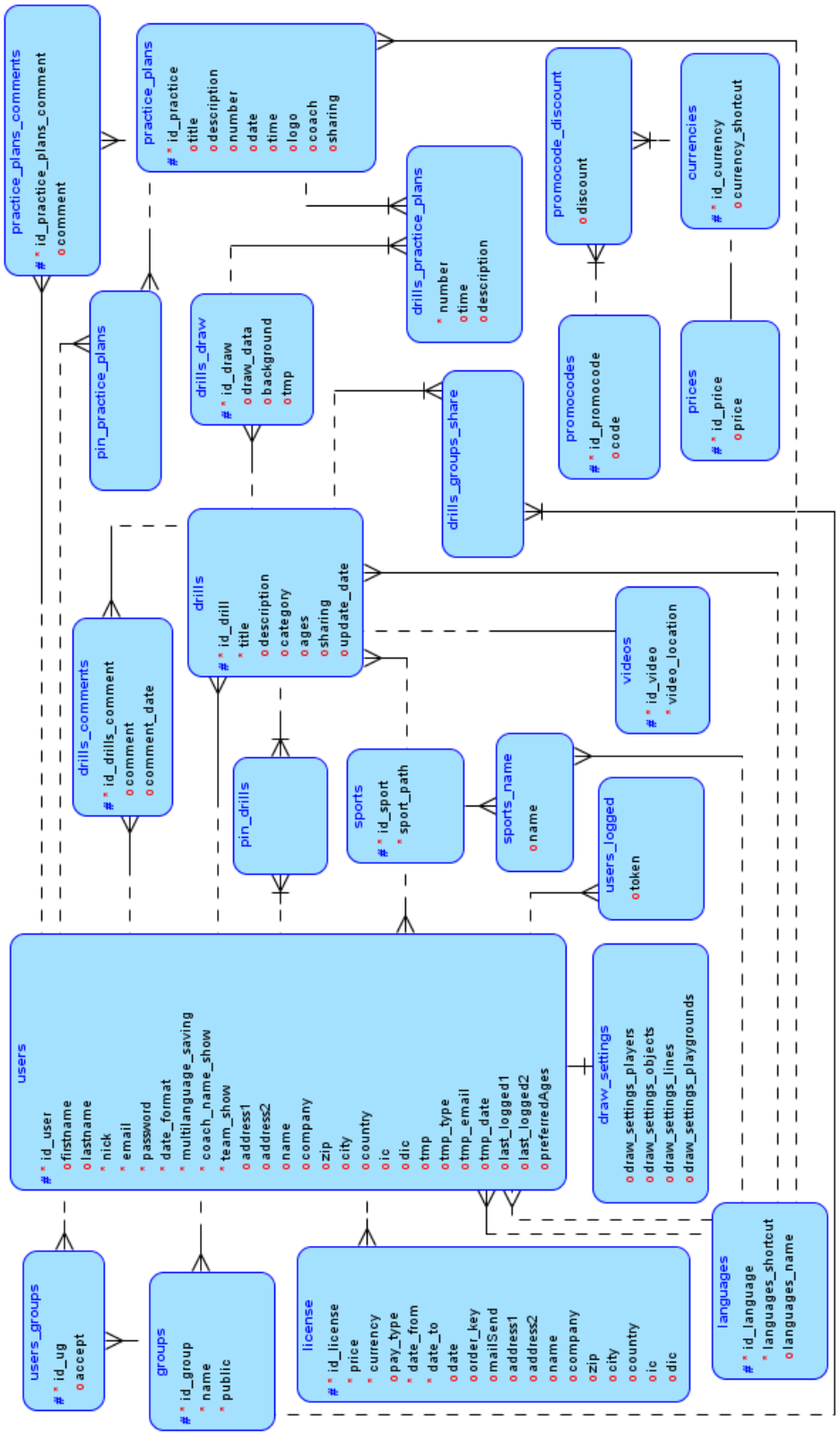
Relace 1:1 označuje, že každý záznam entity má vazbu pouze na jeden záznam jiné entity, mezi nimiž je tento vztah.

Relace 1:N umožňuje, aby jeden záznam z první entity měl vazbu s neomezeným množstvím záznamů z entity druhé, kde má ovšem záznam vazbu pouze na jeden záznam z první entity.

Relace M:N vyplňuje poslední možnou variantu, kdy jeden záznam z první entity má stejně jako u relace 1:N neomezené množství vázaných záznamů z druhé entity, které ovšem můžou mít také neomezený počet záznamů z první entity. Tato relace se často rozepisuje na dvě relace typu 1:N s jednou pomocnou tabulkou.

Poslední důležitou vlastností v relacích je povinnost každého záznamu zúčastnit se relace. To označuje plná čára mezi entitami, případně přerušovaná čára znamená, že každý záznam se nemusí relace účastnit.

Konceptuální databázový model aplikace je zobrazen na obrázku 2.2.



Obrázek 2.2: Konceptuální databázový model

Tento databázový model aplikace úmyslně neobsahuje žádné M:N relace. Tyto relace jsou rovnou rozděleny vždy na dvě 1:N relace tak, aby je bylo možné snadno implementovat v databázi MySQL.

Při návrhu databázového modelu bylo dlouho řešeno, jak se budou u sportů ukládat věkové kategorie a kategorie cvičení. První návrh byl udělat pro tato data speciální entity v databázi, ale po delším zkoumání bylo rozhodnuto, že se data o věkových kategoriích a kategoriích cvičení budou ukládat v textovém řetězci ve formátu JSON a informace o všech kategoriích budou uloženy v souborech s moduly sportů. Tím bylo docíleno snazšího přidávání nových sportů pomocí modulů.

Dalším bodem návrhu byla původní absence entity *languages*, kde bylo myšleno, že jazyky se budou automaticky načítat podle jazykových balíčků nabízených na serveru. Po podrobnějším uvážení se zjistilo, že uživatel může vytvořit cvičení v jazyce, ve kterém není nabízen jazykový balíček a byl by problém takový jazyk na žádost uživatele přidat. Na základě toho vznikla entita *languages*, která obsahuje všechny jazyky, ve kterých je možné psát cvičení s jednoduchou možností editace. Naproti tomu jazyky nabízené pro aplikaci zůstávají nezávisle načítané podle jazykových balíčků ve složce s aplikací.

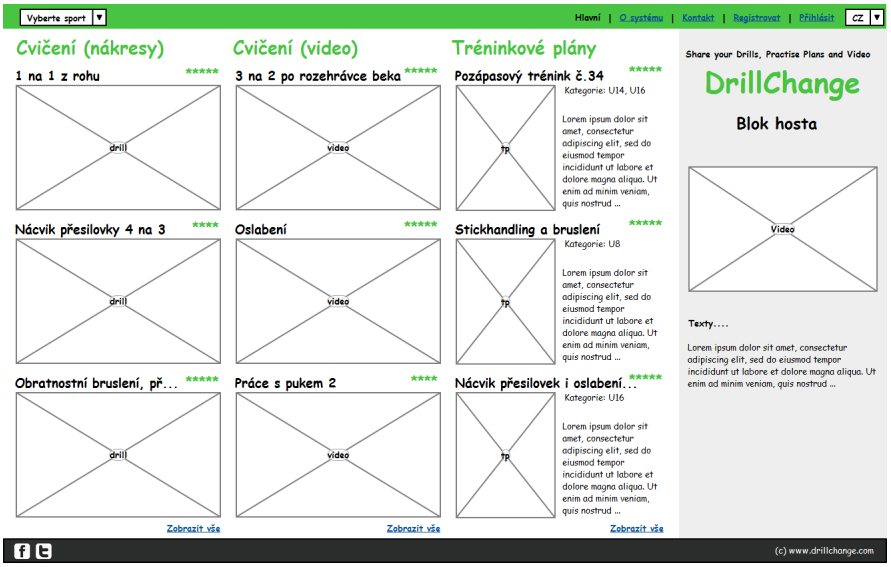
Poslední složitější úvaha byla provedena nad názvy sportů, které se zobrazují uživateli. Původní návrh byl značně nedomyšlený a zamýšlel uložit název sportu vždy v daném modulu. V takovou chvíli by muselo dojít k přečtení dat ve všech dostupných modulech kvůli zobrazení názvů všech sportů. Proto bylo nakonec rozhodnuto přidat entitu *sports_name*, která bude obsahovat názvy všech sportů ve všech jazycích dostupných v entitě *languages*.

2.5 Uživatelské rozhraní

Návrh uživatelského rozhraní je velmi důležitou částí. Je potřeba promyslet, jak nejlépe využít prostor na obrazovce a zároveň danou stránku nepřeplnit spoustou zbytečných informací.

Po důkladném zvážení bylo rozhodnuto, že pro tuto práci bude vytvořena pouze verze pro stolní počítače a tablety, jelikož mobilní telefony nedisponují dostatečně velkým displejem pro kreslení tréninků. Protože se ale bude jednat o aplikaci napsanou v HTML5, měla by bez problému fungovat i na mobilních telefonech, což ovšem není požadovanou funkcionalitou.

Dále bylo rozhodnuto, že systém bude využívat statické šířky 1280 pixelů. Na obrazovkách, které budou mít větší rozlišení bude po stranách nevyužitý prostor, ale nebudou vznikat různé artefakty, kdy by se stránka pokoušela roztahovat do libovolné šířky. Naopak na tabletech a jiných mobilních zařízeních s menším rozlišením než 1280 pixelů na šířku bude stránka zmenšena a zobrazena od kraje ke kraji s možností přiblížení, aby si uživatel zvolil ideální velikost aplikace na obrazovce.



Obrázek 2.3: Návrh uživatelského rozhraní hlavní strany aplikace

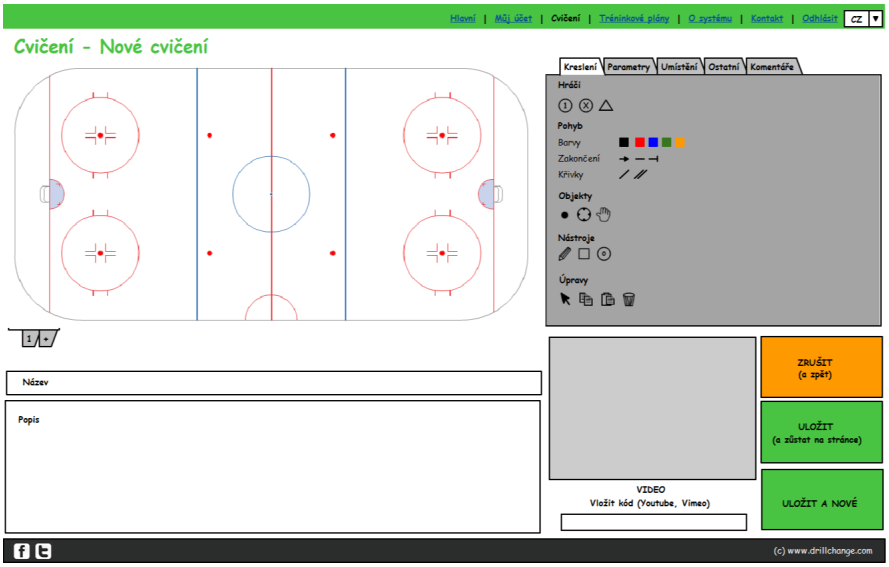
Na většině stran bude aplikace rozdělena na čtyři na sobě nezávislé kontejnery jako na obrázku 2.3. Prvním kontejnerem je horní lišta, ve které se bude zobrazovat menu. Menu bude odlišné pro přihlášeného a nepřihlášeného uživatele. Druhý kontejner (dále označován jako "hlavní kontejner") bude v levé části pod menu a bude zabírat zhruba tři čtvrtiny šířky stránky. V tomto kontejneru budou hlavní data aplikace, například nastavení profilu, informace o aplikaci a jiné, v závislosti na aktuálně vybrané stránce aplikace. Třetím kontejnerem bude postranní panel, který bude napravo od druhého kontejneru a bude široký tak, aby doplnil šířku stránky na požadovaných 1280 pixelů. Jeho obsah se bude lišit pro nepřihlášeného, přihlášeného bez licence a přihlášeného uživatele s platnou licencí. Poslední kontejner je patička, která bude pod všemi ostatními kontejnery. Bude obsahovat pouze několik odkazů, týkajících se aplikace a bude na všech stránkách stejná bez ohledu na uživatelovo oprávnění.

V aplikaci bude několik stran, které skryjí postranní panel a hlavní kontejner s obsahem stránky se roztáhne na celou šířku. Jedná se například o stránky s kreslítkem nebo o stránku s výsledky vyhledávání.

Návrh stránky s kreslítkem je zobrazen na obrázku 2.4.

2.6 AJAX řešení

Jak již bylo uvedeno dříve v kapitole *JavaScript* 1.7.3, aplikace bude využívat systém načítání stránek pomocí technologie AJAX, neboli *Asynchronous JavaScript and XML*. AJAX je označení pro technologii, která načítá veškerá



Obrázek 2.4: Návrh uživatelského rozhraní kreslítka aplikace

data stránek bez nutnosti jejich opětovného načtení, a to celé pomocí knihovny napsané v javascriptu. [18]

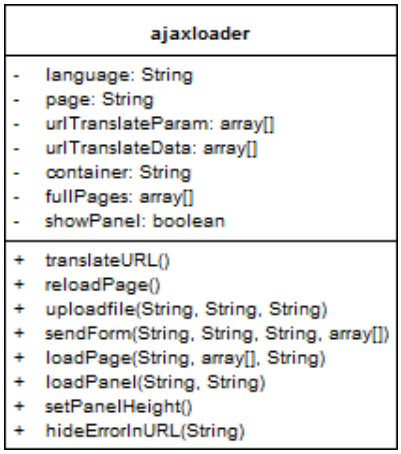
Tato aplikace bude využívat javascriptovou knihovnu jQuery, která usnadňuje práci s AJAXem, kdy je možné o data stránky žádat pouze pomocí jedné metody. Této metodě se v parametrech zvolí, z jaké adresy má načítat soubor, jaká data má do souboru odeslat a jakým způsobem. Dále se zvolí funkce, která se vykoná po dokončení načítání dat, aby je zpracovala.

Aplikace bude obsahovat javascriptovou třídu s názvem *ajaxloader*, která bude obstarávat veškerou komunikaci se serverem a zpracovávat přijatá data. Struktura třídy je vidět na obrázku 2.5.

V atributech si třída uchovává základní nastavení, jako je například vybraný jazyk, aktuální stránka, kontejner, do kterého se načítá aktuální stránka a viditelnost bočního panelu.

Jak bylo uvedeno v kapitole *Uživatelské prostředí* 2.5, webové stránky budou mít postranní panel, který se na některých specifických stránkách nebude zobrazovat. Jde například o stránky s detailem cvičení nebo stránku s editací cvičení. Souhrn těchto stránek, kde se panel nebude zobrazovat, je uvedený v atributu *fullPages*. Při přechodu na jinou stránku proběhne ověření, jestli nová stránka je nebo není v tomto poli a na základě toho se nastaví atribut *showPanel*. Podle tohoto atributu se dále pracuje se stránkou buď jako s celou plochou nebo pouze s částí plochy s postranním panelem.

Samotný princip načítání nové stránky pak bude probíhat následovně - při prvním vstupu do aplikace se vytvoří nová instance třídy *ajaxloader* a následně dojde k načtení hlavní strany aplikace tím, že se zavolá metoda *load-*



Obrázek 2.5: Javascriptová třída ajaxloader

Page s parametry pro hlavní stranu. AJAX v tento moment odešle všechna svá data do speciálního PHP souboru na serveru, který data zpracuje, ověří veškeré vstupy a oprávnění uživatele zobrazit danou stránku a vrátí AJAXu buď data stránky nebo chybovou hlášku. AJAX poté přijatá data zkopíruje do kontejneru, který má aktuálně vybraný. Ve většině případů půjde o hlavní rámec stránky. V ostatních případech pak převážně o postranní panel a horní menu aplikace. Tím proběhlo kompletní načtení úvodní strany aplikace. Při přechodu na jinou stránku aplikace uživatel klikne na libovolný odkaz, který by ho měl přesměrovat na jinou stránku. Tento odkaz ovšem pouze změní URL adresu stránky, do které uloží název stránky a data, která se mají přeposlat dané stránce. V tento moment si všimne změny URL externí funkce a zavolá metodu *loadPage* v instanci třídy *ajaxloader* a veškeré další načítání probíhá shodně jako u hlavní strany.

Složitější úpravou pro AJAXové řešení je odesílání formulářů od uživatele na server. Samotné formuláře automaticky odešlou obsah všech vstupů na vybranou adresu nebo případně na vlastní adresu. V první řadě je nutné zablokovat toto automatické odesílání, aby nedocházelo ke znovunačítání celé stránky. Dále je potřeba všechna data z formuláře získat do schopného formátu, nejlépe zřetězením do jedné proměnné, což umožňuje knihovna jQuery, aby mohla být funkce na odesílání formulářů univerzální pro všechny formuláře. Následně bude zavolána funkce *sendForm* třídy *ajaxloader*, která se již postará o předání dat do PHP, kde dojde k jejich zpracování.

2.7 Kreslící nástroje

Důležitou částí návrhu je ujasnění, jaké nástroje budou uživateli k dispozici pro vytváření nových cvičení a jak se budou dané prvky zobrazovat na kreslící

ploše. Objekty se dělí mezi hráče, křivky, zakončení křivek a ostatní objekty.

2.7.1 Hráči a ostatní objekty

Objekty, které označují hráče, jsou odlišné pro různé sporty. Z tohoto důvodu bylo rozhodnuto, že objekty hráčů budou uloženy a zpracovávány pouze jako obrázky a ne jako geometrické útvary. Na hrací plochu bude možné přidat neomezené množství hráčů, případně hráče přesouvat a odstraňovat.

Ostatní objekty budou mít stejné vlastnoti jako objekty hráčů. Tyto objekty budou znázorňovat například brány, puky, míče a podobně. Stejně jako hráči, budou i ostatní objekty odlišné pro různé sporty, například pro hokej bude vypadat objekt brány jinak, než pro fotbal.

2.7.2 Křivky

Křivka bude znázorňovat trajektorii pohybu hráče nebo objektu, například puku. Všechny výpočty pro vytvoření křivek, vyobrazené na obrázku 2.6, budou uloženy přímo v kreslítku a jednotlivé moduly sportů budou mít uložené, jaké křivky se mají uživateli zobrazit v daném sportu. Případně si uživatel bude moci křivky vybírat a skrývat v nastavení kreslítka.

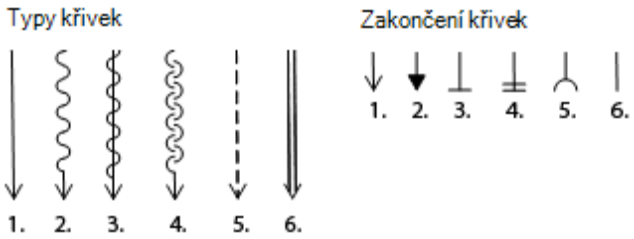
Průběh kreslení křivky na počítači bude probíhat tak, že uživatel zvolí typ a barvu křivky, kterou chce kreslit a na kreslící ploše klikne na místo, kde bude křivka začínat. Poté klikne na všechna místa, přes která bude křivka procházet a jakmile bude mít zvolené všechny tyto body (dále označované jako *řídící body*) klikne pravým tlačítkem myši, čímž aplikace dostane signál, aby vypočítala a vykreslila celou křivku. Pokud bude na dotykovém zařízení, průběh kreslení bude probíhat stejně jako na počítači a pokud klepne na předem určené tlačítko, tak ukončí nákres křivky.

Podle popisu křivky v odstavci výše musí být jednoznačně použita interpolační křivka, která prochází skrze zvolené řídící body a pro jejíž výpočet stačí pouze souřadnice řídících bodů, na základě kterých se dopočtou potřebné vektory. Jestliže bude aplikace disponovat souřadnicemi potřebných bodů a směrovými vektory v nich, pak je nejvhodnější použít rovnici pro Fergusonovu křivku, která je uvedena v následujících odstavcích.

V každém řídícím bodě dojde k vypočítání směrového vektoru. V prvním bodě pomocí rovnice 2.1, kde body A a B jsou dva sousedící řídící body.

$$v_x = B_x - A_x; v_y = B_y - A_y \tag{2.1}$$

V dalších bodech bude výpočet podobný, pouze se k výslednému vektoru připočte předchozí směrový vektor. Následně se křivka bude vykreslovat po částech mezi zadanými body. Vždy dojde k výpočtu, kolik a na jakých souřadnicích budou doplněné body mezi řídícími body tak, aby splňovaly určitou minimální a maximální vzdálenost v závislosti na požadovaném vyhlazení křivky.



Obrázek 2.6: Typy křivek a zakončení v kreslítku

Tento výpočet vždy vypočte první dva body pomocí rovnice Fergusonovy křivky a porovná jejich vzdálenost pomocí rovnice pro výpočet vzdálenosti mezi dvěma body v rovině 2.2. Výpočet se bude provádět tak dlouho, dokud od sebe nebudou body vzdáleny dle požadavku.

$$|AB| = \sqrt{(B_x - A_x)^2 + (B_y - A_y)^2} \tag{2.2}$$

Jakmile bude znám celkový počet bodů mezi řídícími body, tak se tyto body vypočítají pomocí rovnice pro výpočet Fergusonovy křivky 2.7. Do této rovnice je ještě zapotřebí dopočítat jednotlivé proměnné, kde parametry w_0, w_1, w_2 a w_3 jsou vypočítány danými rovnicemi 2.3, 2.4, 2.5 a 2.6. Parametr t se pohybuje v intervalu <0;1> a určuje pozici právě počítaného bodu mezi řídícími body. [19]

$$w_0(t) = 1 - 3t^2 + 2t^3 \tag{2.3}$$

$$w_1(t) = 3t^2 - 2t^3 \tag{2.4}$$

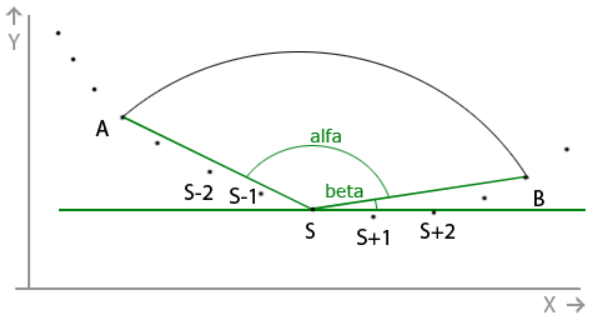
$$w_2(t) = t - 2t^2 + t^3 \tag{2.5}$$

$$w_3(t) = -t^2 + t^3 \tag{2.6}$$

$$P(t) = w_0(t)P_0 + w_1(t)P_1 + w_2(t)v_0 + w_3(t)v_1 \tag{2.7}$$

Zde je uveden výpis všech křivek (očíslovány stejně jako na obrázku) a řešení jejich nákresu v aplikaci pomocí matematiky a knihovny kineticJS.

1. Jedná se o nejjednodušší křivku. Pouze se vezmou jednotlivé body, které byly vypočítány pomocí rovnice Fergusonovy křivky, převedou se do jednoho pole prvků a nechají se zpracovat knihovnou kineticJS, která z nich vytvoří souvislou křivku.



Obrázek 2.7: Nákres vlnité křivky

2. Tato křivka již vyžaduje složitější výpočet. Knihovna kineticJS umí nakreslit oblouk, stačí zadat střed, poloměr a dva úhly označující počátek a konec oblouku. Poloměr oblouku je v kreslítku pevně definovaný, je tedy potřeba dopočítat střed a oba úhly.

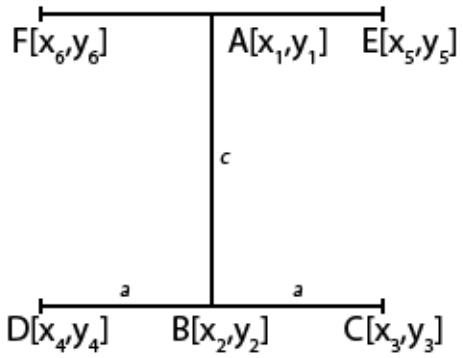
   Pro tuto křivku nebude platit výše uvedené, že body na křivce budou mezi definovanou minimální a maximální vzdáleností. Naopak se vypočte co nejvíce bodů s minimální vzdáleností tak, aby bylo možné vždy od prvního bodu (A) na křivce nalézt druhý (S), který bude vzdálen co nejpřesněji velikosti poloměru a následně třetí bod (B), který bude opět ve vzdálenosti velikost poloměru od bodu S. Bod S je v takovou chvíli středem oblouku a body A a B jsou krajní body oblouku. Následně dojde k vypočítání úhlu *beta* a úhlu *alfa+beta*, což jsou ony požadované úhly pro nákres oblouku. Tato situace je zobrazena na obrázku 2.7. Při počítání dalšího oblouku se začne od bodu B a pokračuje se stejným způsobem, pouze se oblouk vykreslí opačně, než předchozí.

3. Tento typ křivky vytvoří první i druhou křivku se stejnými souřadnicemi. Tím dojde k vytvoření požadované křivky.

4. Výpočet a vykreslení křivky bude shodné jako u křivky číslo 2, s tím rozdílem, že po vytvoření oblouku nebude začínat další oblouk od bodu B, ale bodu *S+1*.

5. Vytvoření křivky bude probíhat naprosto stejně jako u křivky 1. Knihovna kineticJS podporuje způsob vykreslení výsledné křivky se stylem “dotted”, což znamená, že se čára vykreslí čárkovaně. Stačí tedy upozornit knihovnu na tuto skutečnost před vykreslením křivky.

6. Jako nejlepší varianta se jeví počítání křivky mezi každými dvěmi uloženými body. Tyto dva body jsou označeny A a B na obrázku 2.8. Je potřeba dopočítat souřadnice bodů C, D, E a F, ze kterých vzniknou nové souřadnice pro dvě křivky. Vzdálenost bodů B a C je pevně ur-



Obrázek 2.8: Náčrt situace pro výpočet souřadnic bodů C a D

čena v proměnné a, která je známá a na jejíž velikosti bude závislá šířka křivky.

Samotný výpočet začne zjištěním směrového vektoru mezi body A a B. Směrový vektor se vypočte tak, že se souřadnice bodu A odečtou od souřadnic bodu B po jednotlivých složkách, jak již bylo uvedeno výše v rovnici 2.1. Výsledný vektor je označený jako s se souřadnicemi s_x a s_y. Poté dojde k vytvoření kolmého vektoru k vektoru s, označeného písmenem v se souřadnicemi v_x a v_y. Vektor v vznikne prohozením složek vektoru s a změnou znaménka jedné libovolné složky. Jelikož se jedná o kolmé vektory, musí platit rovnice 2.8, která bude následně potřeba k dalším výpočtům. [20]

$$s_x * v_x + s_y * v_y = 0 \tag{2.8}$$

Následně je z rovnice získáno úpravou vyjádření souřadnice v_x:

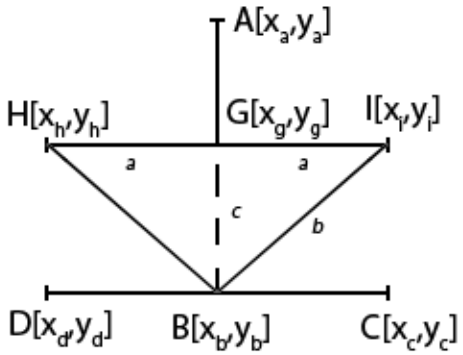
$$\frac{s_y * v_y}{-s_x} = v_x \tag{2.9}$$

Druhá rovnice, která bude sloužit pro výpočet souřadnice bodů C a D, bude rovnice pro výpočet velikosti vektoru. Z předchozího popisu je známo, že velikost vektoru v musí být rovna proměnné a, takže rovnice bude vypadat následovně:

$$\sqrt{v_x^2 + v_y^2} = a \tag{2.10}$$

Poté dojde k dosazení v_x za jeho vyjádření z předchozí rovnice:

$$\sqrt{(\frac{s_y * v_y}{-s_x})^2 + v_y^2} = a \tag{2.11}$$



Obrázek 2.9: Zakončení ve tvaru šipky

Dalšími úpravami se dojde k finální podobě rovnice:

$$\sqrt{\frac{a^2}{\frac{s_y^2}{s_x^2}+1}} = v_y \tag{2.12}$$

Druhá souřadnice vektoru v se dopočte pomocí rovnice 2.9.

Tímto výpočtem byl získán vektor v, o který stačí posunout bod B a tím jsou získány souřadnice bodu C. Stejným způsobem je možné vypočítat bod D a případně další dva body E a F.

Tím dojde k získání bodů C a E a bodů D a F, které se uloží do dvou polí souřadnic, ze kterých nakonec vzniknou dvě nezávislé křivky stejným způsobem jako první křivka - tedy předáním souřadnic do knihovny kineticJS.

2.7.3 Zakončení křivek

Zakončení bude znázorňovat, co má hráč udělat na konci pohybu, který bude znázorňovat křivka. Zakončení budou uložena globálně a v jednotlivých sportech bude navoleno, která zakončení se mají v daném sportu zobrazit, stejně jako u křivek. Druhy zakončení jsou vidět na obrázku 2.6.

Většina zakončení bude využívat podobný nebo shodný výpočet pro získání dodatečných bodů jako poslední typ křivky. Označení všech bodů v náčrtech jsou shodná pro všechna zakončení a jsou vidět na obrázku 2.9.

1. Nejdříve dojde k vypočítání bodů na kolmici, stejně jako u 6. křivky, kde jsou body označené shodně C a D. Následně je potřeba dopočítat souřadnice vektoru s tak, jako výše u křivky 6, tedy odečtením souřadnic bodu A od souřadnic bodu B. Dále je potřeba vypočítat koeficient x, kterým se vynásobí vektor s a vznikne tím nový vektor, pomocí kterého

dochází k posunu bodů C a D. Na cílových místech tak vznikly nové body H a I. Následným vykreslením přímek $|BH|$ a $|BI|$ vznikne šipka.

Výše zmíněný koeficient x se vypočte pomocí následujícího vzorce, kde proměnná c je pevně dané číslo:

$$x^2 = \frac{c^2}{s_1^2 + s_2^2} \tag{2.13}$$

Tento vzorec vznikl pouhou úpravou vzorce pro výpočet velikosti vektoru, kde obě složky vektoru s byly násobeny proměnnou x.

2. Pro toto zakončení je potřeba dopočítat všechny tři body jako u 1. zakončení. Ve výsledku se ovšem nevykreslí dvě úsečky, ale plnohodnotný trojúhelník s vrcholy v bodech H, B a I.

3. Jde pouze o kolmici, která vznikne dopočítáním bodů C a D a jejím následným spojením v úsečku.

4. Toto zakončení je kombinací 1. a 3. zakončení. Nejdříve se vytvoří 3. zakončení a následně dojde k dopočítání vektoru a koeficientu, kterým se vektor upraví na požadovanou délku. Jako poslední dojde k posunu bodů B, C a D, čímž vzniknou body G, H a I. Po získání všech bodů se už jen spojí příslušné úsečky $|HI|$ a $|GB|$.

5. Stejným způsobem jako v bodě 1. dojde k dopočítání bodů C, D a G a dojde k vykreslení oblouku se středem v bodě B a průměru odpovídajícímu délce úsečky $|HI|$. Zde je nutné dbát na to, aby poloměr oblouku byl shodný se vzdáleností bodů G a B. U tohoto zakončení je potřeba u křivky počítat s tím, že poslední část křivky mezi posledními dopočítávanými body se nevykreslí, aby její část nezasahovala do oblouku.

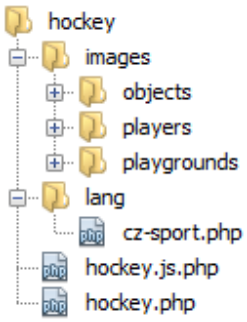
6. V tomto případě se žádné zakončení nevykreslí a nebude se nic dopočítávat.

2.8 Moduly sportů

Aplikace musí být multisportovní s možností jednoduše přidávat nové sporty. Z toho důvodu bylo rozhodnuto, že sporty budou uloženy jako jednotlivé moduly, kdy se uživateli načte právě ten modul, který používá, jelikož nikde nepůjde používat dva a více sportů najednou.

Základní informace o sportu, tedy pouze jeho název, cesta k souborům a unikátní identifikátor budou uloženy v databázi, ostatní data týkající se sportu budou uložena na serveru ve speciální složce s názvem *sports*.

Každý modul bude striktně dodržovat předepsanou strukturu složky, včetně názvů souborů a složek, jako je tomu na obrázku 2.10, kde ovšem název dvou



Obrázek 2.10: Souborová struktura modulu

souborů typu PHP bude pro každý sport jiný, podle názvu sportu, čímž bude shodný s názvem složky modulu.

Složka *images* obsahuje všechny obrázky, které ke sportu patří, roztříděné do jednotlivých složek. Další složka *lang* obsahuje PHP soubory, kde každý soubor je jazykový balíček pro sport, využitý ke správnému pojmenování kategorií cvičeních, věkových kategorií a názvů hráčů, křivek, zakončení a objektů. Jazyk je určen dvoumístným prefixem názvu souboru. V souboru *hockey.php* je definována třída *SportData* 2.3.7, která je potomkem třídy *Sport*. Poslední soubor v modulu *hockey.js.php* je do aplikace nahrán jako soubor typu *javascript*, který díky koncovce *php* (kdy může před použitím v aplikaci vykonat PHP skripty) nahraje data ze třídy SportData, aby je mohl využívat i javascript a nemusely být definovány na dvou místech.

Kapitola 3

Implementace

V kapitole implementace je pozornost věnována problémům, které vznikly v průběhu implementace a byly opomenuty v návrhu, případně nebyly navržené ideálně. V implementační části byly všechny takové problémy nalezeny a vyřešeny.

3.1 Uložení vlastního obrázku hřiště

Aplikace musí podporovat načítání nových stránek pomocí technologie AJAX. Tato technologie ovšem neumožňuje žádným jednoduchým způsobem posílání souboru směrem od uživatele na server, což je potřeba ve chvíli, kdy se uživatel rozhodne nahrát vlastní obrázek hřiště, na kterém chce vytvořit nové cvičení.

Jako nejsnazší varianta řešení se nabízí porušit pravidlo načítání technologií AJAX pro jedinou stránku, na které uživatel nahrává nové obrázky hracích ploch. Jedná se o nejjednodušší způsob. Ten byl ale zamítnut, jelikož by bylo nepříjemné, kdyby všechny stránky měnily svůj obsah plynule, ale jedna by svůj obsah nahrála znovu celý s probliknutím celé aplikace. Narušilo by to konzistenci aplikace.

Druhým způsobem, který by technologii AJAX na první pohled nijak nenarušil, by bylo vytvořit na stránce neviditelné vnořené okno - takzvaný "IFRAME", do kterého by se odeslal formulář, ve kterém uživatel vybral soubor a ve kterém by došlo k jeho uložení na server. Toto okno by pak informovalo hlavní okno o skutečnosti, jestli byl soubor úspěšně nahrán nebo došlo k chybě při nahrávání. Hlavní okno by už jen zobrazilo uživateli zprávu, jestli byl soubor nahrán nebo ne a případně by znovu načetlo seznam nahraných obrázků, a to již běžně technologií AJAX.

Po zvážení byl nakonec vybrán a implementován druhý způsob.

3.2 Náhled videa u cvičení

Uživatel uvidí na hlavní straně aplikace tři sloupečky, ve kterých budou uvedena naposledy přidaná cvičení, naposledy přidaná cvičení s videi a naposledy přidané tréninkové plány. U každého zobrazeného cvičení bude náhled na hřiště s cvičením, u cvičení s videem bude náhled na video a u tréninkových plánů bude náhled, ve kterém bude přibližný vzhled tréninkového plánu po exportu na papír ve formátu A4. Při implementaci hlavní strany aplikace bylo zjištěno, že v návrhu bylo opomenuto to, jakým způsobem se budou získávat náhledy na videa, která jsou uložena u cvičení.

Jak již bylo zmíněno v analýze, videa budou uložena na serverech *YouTube* a *Vimeo*. Před zobrazením náhledu tedy musí aplikace ověřit, jestli se v adrese videa nachází řetězec "vimeo" nebo řetězec "youtube", případně "y2u.be", což je další doména serveru *YouTube*. Jakmile aplikace ví, na kterém serveru je video uloženo, tak rozdělí adresu videa, aby z ní získala identifikátor videa. Podle identifikátoru je pak možné vyhledat miniaturu na serveru s videem.

Zatímco na serveru *Vimeo* mají všechna videa shodnou adresu, ve které je jediným rozdílem jejich identifikátor, tak server *YouTube* nabízí několik možných typů adres a je tedy složitější z takové adresy získat identifikátor videa. Řešení tohoto problému bylo nalezeno na oblíbeném serveru *stackoverflow* [21], a to takové, že adresa videa se rozdělí na několik částí popsaných v regulárním výrazu a z těchto částí se pak vybere jedna, která označuje identifikátor videa. Regulární výraz vypadá následovně:

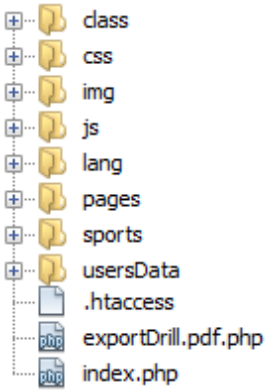
```
/^.*(youtu.be\/|v\/|u\/\w\/|embed\/|watch\?v=|\&v=)([^#\&\?]*).*/
```

Regulární výraz je řetězec, který popisuje vlastnosti a vzhled ostatních řetězců a právě díky němu je možné jednoduše rozdělit řetězce o daných parametrech na potřebné části.

Jakmile je získán identifikátor videa, stačí jen vytvořit odkaz na obrázek, který je na předdefinované adrese na serveru, kde je i video. K jeho nalezení je zapotřebí výše hledaný identifikátor.

3.3 Výstup práce

Výstupem práce je aplikace, která je funkční na všech serverech s podporou PHP 5.3 a s databázovým systémem MySQL 5.5. Následně je spustitelná na všech počítačích s nainstalovaným prohlížečem s minimální požadovanou verzí uvedenou v nefunkčních požadavcích 1.3.2 a s přístupem k serveru s aplikací. Pro ukázku je na obrázku 3.1 uveden seznam složek a souborů v hlavním adresáři aplikace.



Obrázek 3.1: Stromová struktura složky s aplikací

class - složka obsahující zdrojové kódy všech tříd v PHP

css - v této složce jsou umístěny všechny soubory obsahující CSS styly pro aplikaci

img - složka s obrázky aplikace, například s logem aplikace

js - zde jsou uloženy všechny javascriptové soubory aplikace a všechny javascriptové knihovny, které aplikace využívá

lang - v této složce jsou uloženy všechny jazykové balíčky, které aplikace podporuje

pages - složka obsahuje jednotlivé PHP stránky aplikace

sports - všechny složky s moduly sportů jsou k nalezení v této složce

usersData - v této složce jsou uživatelská data - jedná se o vlastní obrázky hřišť, náhledy cvičení a exporty

.htaccess - soubor, ve kterém se určují oprávnění k výše uvedeným složkám a tím se zabraňuje veřejnému přístupu do složek, které uživatel nepotřebuje mít dostupné

exportDrill.pdf.php - tento soubor vytváří PDF soubor, ve kterém jsou exportované tréninkové plány

index.php - hlavní soubor aplikace, který zavádí všechny potřebné knihovny, vytváří instanci třídy pracující s AJAXem a spouští aplikaci

Kapitola 4

Testování

Tato kapitola popisuje průběh testování aplikace mezi různými zařízeními s odlišnými platformami a nejběžnějšími webovými prohlížeči. Nejdříve bylo sestaveno několik scénářů testování, které se následně na všech zařízeních otestovaly. Testování prováděli osoby nejen z oboru informatiky, ale i trenéři, kteří mají pouze obecné znalosti běžného uživatele.

4.1 Testeři

Celkem provádělo testování pět osob.

Tester 1 - jedná se o programátora aplikace, který prováděl testování v průběhu celé implementace, ale zúčastnil se i závěrečného testování. Jeho testování probíhalo spíše na úrovni kvality aplikace a jen drobně na logice aplikace

Tester 2 - spoluautor zadání aplikace. Jde o testera, který se velmi aktivně věnuje sportu a zároveň je pokročilým uživatelem počítačů a tabletů. Jeho hlavním úkolem byla kontrola logiky aplikace

Tester 3 - profesionální programátor, který se v průběhu testování věnoval pouze kvalitě aplikace a hledal případné nedostatky a bezpečnostní rizika

Tester 4 - trenér malého hokejového týmu, který pracuje s počítači a tablety spíše zřídka a řadí se mezi průměrné uživatele. Jeho výsledky testování slouží ke zjištění, jak na aplikaci bude reagovat cílová skupina uživatelů

Tester 5 - trenér hokejového týmu starších žáků. Má dobré zkušenosti s prací na počítači, jelikož v zaměstnání používá počítač i tablet

4.2 Scénáře testování

Scénáře testování popisují, co se má uživatel pokusit udělat a tím otestovat aplikaci. Testování probíhalo tak, že si uživatel nejdříve prohlédl aplikaci, seznámil se s ní a následně se pokusil splnit postupně všechny scénáře.

První scénář nařizuje uživateli zaregistrovat se do systému a následně provést objednávku licence. Po splnění provede ještě úpravu svého profilu, kde změní minimálně heslo a e-mailovou schránku.

Druhý scénář navazuje na první. Nejdříve se uživatel přihlásí a následně se pokusí vytvořit nové cvičení s předem určenými parametry ve scénáři. Toto cvičení musí uložit, následně znovu otevřít v módu pro úpravu a znovu upravit stávající nákres cvičení.

Třetí a poslední scénář začíná ve chvíli, kdy se uživatel chce stát členem veřejné skupiny. Po vstupu do této skupiny je jeho úkolem vytvořit nový tréninkový plán, do kterého přidá svoje cvičení vytvořené v předchozím scénáři, dále přidá alespoň dvě cvičení sdílená ve veřejné skupině, které je nyní členem a ve finále takto vytvořený tréninkový plán exportuje do formátu PDF.

4.3 Testovaná zařízení

Veškeré testy byly provedeny pod dozorem autora a spoluautora na všech níže uvedených počítačích s různými operačními systémy a webovými prohlížeči a na různých tabletech.

1. Počítač 1 s OS Windows 8.1
   - Internet Explorer (verze 11.0.9600.17728)
   - Opera (verze 29.0.1795.47)
   - Google Chrome (verze 39.0.2171.95)
   - Mozilla Firefox (verze 37.0.2)
2. Počítač 2 s OS Windows 7
   - Internet Explorer (verze 11.0.9600.17631)
   - Opera (verze 28.0.1750.40)
3. Tablet 1 Apple iPAD mini s OS iOS 8.3
   - Safari (verze 8.0)
   - Google Chrome (verze 42.0.2311.47)
4. Tablet 2 Samsung GT-P110 s OS Android 4.2.2
   - integrovaný prohlížeč (verze 534.30)
   - Opera (verze 29.0.1809.91837)

4.4 Závěr testování

Na všech zařízeních proběhly všechny testy a provedení scénářů bez jakéhokoliv problému.

Testery, kteří viděli aplikaci poprvé, bylo zhodnoceno, že je aplikace svojí grafickou jednoduchostí velmi intuitivní na ovládání a orientaci, což byl původní záměr při tvorbě uživatelského prostředí.

Testování tedy skončilo úspěšně.

Závěr

Cílem práce bylo správné zanalyzování, navržení, implementace a řádné otestování vytvořeného řešení.

V průběhu analýzy byly prozkoumány existující řešení na trhu se zjištěním, že neexistuje žádná konkurenceschopná aplikace, což bylo následně uvedeno jako hlavní důvod motivace tvorby vlastního řešení. Dále byly probrány požadavky na aplikaci, rozdělení jednotlivých uživatelských rolí aplikace a případy užití. Na základě toho mohl vzniknout analytický doménový model aplikace a být vybrány vhodné technologie pro implementaci aplikace. Po určení vhodných technologií bylo nutné ještě prozkoumat bezpečnostní rizika v daných technologiích.

Návrh se již zabýval konkrétními problémy, které bylo nutné promyslet před samotnou implementací. Byla vybrána architektura aplikace a architektura prezenční vrstvy. Na základě analytického doménového modelu z první kapitoly byly vytvořeny návrhový model tříd a konceptuální databázový model. Dále byl navržen přibližný vzhled aplikace a rozvržen princip AJAXového řešení aplikace. Nejdůležitějším bodem návrhu bylo určení výpočtů pro získání souřadnic bodů a způsobu vykreslení jednotlivých křivek a různých typů zakončení.

Implementační část se zabývala pouze několika málo problémy, které vznikly v průběhu implementace a závěrem shrnula, co je výstupem práce.

V kapitole testování byly určeny způsoby testování a testeři, kteří testy prováděly. Nakonec bylo testování zhodnoceno jako úspěšné.

Na základě shrnutí v předchozích odstavcích bylo zadání jednoznačně splněno, a to bez větších problémů.

Možnosti rozšíření

Nejžádanějším rozšířením je umožnit vytváření animace z cvičení. Princip by byl takový, že by na stránce s úpravou cvičení byla časová osa, kde by si uživatel vytvořil nové cvičení a pak by na časové ose nastavil pozice jednotlivých

objektů a případně změny objektů a křivek v různých časových intervalech. Aplikace by pak plynule posouvala objekty tak, aby byly vždy v nastaveném intervalu na určeném místě.

Další návrhy na rozšíření vzniknou v průběhu používání aplikace uživateli na základě jejich potřeb.

Literatura

[1] Hockey Canada Drill Hub [online]. [vid. 2016-11-06]. Dostupné z: http://www.hockeycanada.ca/en-ca/Hockey-Programs/Drill-Hub

[2] Online Basketball Drills [online]. [vid. 2016-11-06]. Dostupné z: http://www.online-basketball-drills.com/

[3] Hes, M.: Stránka aplikace Drillbook Online [online]. [vid. 2016-11-06]. Dostupné z: http://www.drillbook.net/

[4] Stránka aplikace DRILLFY [online]. [vid. 2016-11-06]. Dostupné z: http://www.drillfy.com

[5] Stránka aplikace Hockey Share [online]. [vid. 2016-11-06]. Dostupné z: http://www.hockeyshare.com/

[6] Stránka aplikace HockeyCoachToCoach [online]. [vid. 2016-11-06]. Dostupné z: http://www.hockeycoachtocoach.com/

[7] Hes, M.: Popis aplikace Drillbook Online [online]. [vid. 2014-11-18]. Dostupné z: http://www.drillbook.net/db-online/popis/

[8] Stránka aplikace peluu [online]. [vid. 2016-11-19]. Dostupné z: http://www.peluu.com/

[9] Wikipedia: Analýza požadavků [online]. [vid. 2015-02-15]. Dostupné z: http://cs.wikipedia.org/wiki/Analýza_požadavků

[10] Wikipedia: Nefunkční požadavky softwarové architektury [online]. [vid. 2015-02-15]. Dostupné z: http://cs.wikipedia.org/wiki/Nefunkční_požadavky_softwarové_architektury

[11] Mlejnek, I. J.: BI-SI1 - Přednáška č.4 - Analýza problémové domény. Říjen 2013, [cit. 2015-02-26].

[12] Wikipedia: Licence MIT [online]. 2013, [vid. 2015-03-04]. Dostupné z: http://cs.wikipedia.org/wiki/Licence_MIT

[13] PHP Licensing. [vid. 2015-03-04]. Dostupné z: http://php.net/license/

[14] Wikipedia: Cross-site scripting [online]. 2013, [vid. 2015-03-04]. Dostupné z: http://cs.wikipedia.org/wiki/Cross-site_scripting

[15] Wikipedia: Cross-site request forgery [online]. 2014, [vid. 2015-03-06]. Dostupné z: http://cs.wikipedia.org/wiki/Cross-site_request_forgery

[16] Cingroš, M.: CRUD [online]. 2007, [vid. 2015-03-08]. Dostupné z: http://www.abclinuxu.cz/slovnik/crud

[17] Bernard, B.: Úvod do architektury MVC [online]. 2009, [vid. 2015-03-08]. Dostupné z: http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/

[18] AJAX [online]. [vid. 2015-03-09]. Dostupné z: http://www.adaptic.cz/znalosti/slovnicek/ajax/

[19] Alžběta Davidová a Milan Vít: Fergusonova křivka [online]. 2013, [vid. 2015-03-10]. Dostupné z: http://personal.tucna.net/cult/ferguson.html

[20] Kolmost vektorů [online]. [vid. 2015-03-10]. Dostupné z: http://www.aristoteles.cz/matematika/analyticka_geometrie/vektor/kolmost-vektoru-priklady.php

[21] Získání unikátního identifikátoru videa z URL kanálu YouTube [online]. 2010, [vid. 2017-01-05]. Dostupné z: http://stackoverflow.com/questions/3452546/javascript-regex-how-to-get-youtube-video-id-from-url

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

HTML HyperText Markup Language

PHP Hypertext Preprocessor

AJAX Asynchronous JavaScript and XML

IFRAME Inline FRAME

CRUD Create, Read, Update, Delete

SQL Structured Query Language

MVC Model-View-Controller

XSS Cross-Site Scripting

CSRF Cross-Site Request Forgery

UC Use Case

URL Uniform Resource Locator

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF