České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra počítačů

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Nguyen Phong Tat**

Studijní program: Softwarové technologie a management
Obor: Softwarové inženýrství

Název tématu: **Systém pro správu sportovních her**

Pokyny pro vypracování:

Navrhněte a implementujte systém pro správu sportovních aktivit pořádaných ve firmě. Systém by měl rychle zvládat správu organizace i vyhodnocování aktivit, vytvářet profesionální a konfigurovatelné tiskové výstupy, tisk diplomů a další služby potřebné pro soutěže. Navrhněte vhodnou architekturu systému a realizujte v . NET. Výsledný produkt dostatečně otestujte. Dosažené výsledky zhodnoťte.

Seznam odborné literatury:

Pressman, R.S, Maxim, B.: Software Engineering a Practitioner's Approach, McGraw-Hill, 2014, ISBN-13: 978-0078022128 ISBN-10: 0078022126
Watkins, D., Hammond, M., Abrahams, B.: Programming in the .NET Environment, Microsoft .NET development series, ISBN-13: 078-5342770186 ISBN-10: 0201770180

Vedoucí: Ing. Božena Mannová, Ph.D.

Platnost zadání: do konce letního semestru 2015/2016

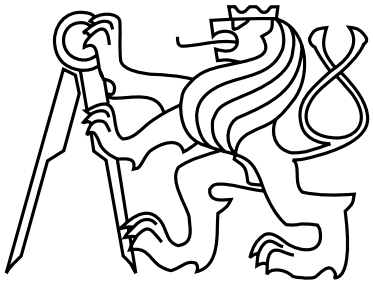
L.S.

doc. Ing. Filip Železný, Ph.D.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 17. 4. 2015

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů



BAKALÁŘSKÁ PRÁCE

SYSTÉM PRO SPRÁVU SPORTOVNÍCH HER

NGUYEN TAT PHONG

Vedoucí práce: Ing. Božena Mannová, Ph.D.

Studijní program: Softwarové technologie a management, Bakalářský

Obor: Softwarové inženýrství

PODĚKOVÁNÍ

Tímto bych velice rád poděkoval své vedoucí bakalářské práce Dr. Boženě Mannové, Ing., Ph.D za odborné vedení, cenné rady a trpělivost.

PROHLÁŠENÍ

Prohlašuji, že jsem práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 21.5.2015 .....

ABSTRAKT

Předmětem práce je problematika návrhu informačního systému pro správu sportovních aktivit pořádaných ve firmě a implementace takového systému využitím technologie .NET. Systém si vede údaje o hráčích, sportovních týmech a jednotlivých utkáních. Data umí interpretovat ve formě statistik a výkonnostních tabulek, které dokáže uživateli rozumně prezentovat s možností tisku.

ABSTRACT

The thesis analyzes how the information systems for sporting events are designed and describes an implementation of a system by using .NET technology. The system tracks information about players, sport teams and matches. It is able to interpret the data and provide a user with statistical information and tables with an option to print.

OBSAH

1 ÚVOD

Cílem projektu je navrhnout a vytvořit informační systém s webovým rozhraním pro firmy pořádající sportovní soutěže. Systém má za úkol zjednodušit uživateli organizaci sportovních utkání a má být implementován za pomoci technologie .NET od firmy Microsoft.

Součástí této práce je analýza dostupných systémů na trhu, která se nachází ve druhé kapitole. Ve třetí kapitole je popsána problematika podobných systémů a jsou v ní specifikovány cíle projektu. Čtvrtá kapitola se zabývá analytickou částí práce s katalogem požadavků, doménovým modelem a seznamem případů užití. V páté kapitole je seznam zvolených technologií a zahrnuje i popis implementace systému. Šestá kapitola se věnuje testování systému. V poslední kapitole jsou sepsány klady a zápory systému a doporučení pro případný další vývoj aplikace.

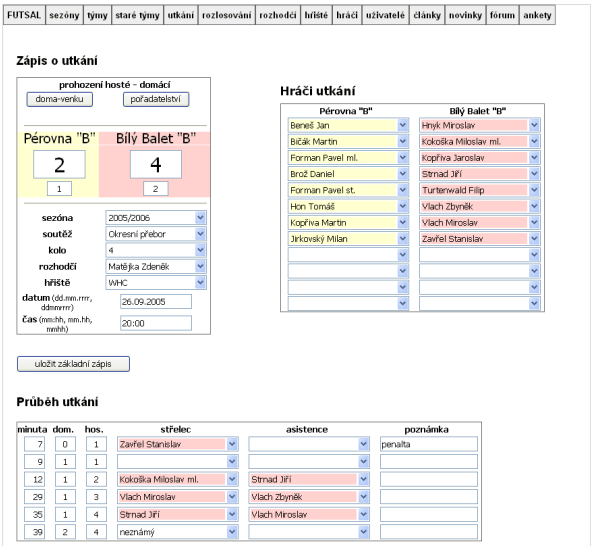
2 ANALÝZA DOSTUPNÝCH SYSTÉMŮ

Na úvod byla provedena rešerše o dostupných systémech pro pořadatele sportovních utkání a turnajů na trhu. Celkem byl jeden nalezen srovnatelný systém v češtině a čtyři systémy v angličtině.

2.1 ISPORT

iSport je webový systém poskytovaný Romanem Jankem. Aplikace je primárně určena pro kolektivní sporty a umí spravovat informace o hráčích, sportovních týmech a soutěžích. Webový systém iSport má tři hlavní funkcionality - publikační systém, administraci a generování výstupních dat.

Publikační systém má na starosti správu obsahu, tedy psaní nových aktualit, nebo publikace a editace článků. Tato část tak může sloužit jako webová prezentace sportovního klubu.



Obrázek 1: Ukázka systému iSport, podrobnosti zápisu o utkání. Zdroj: [1]

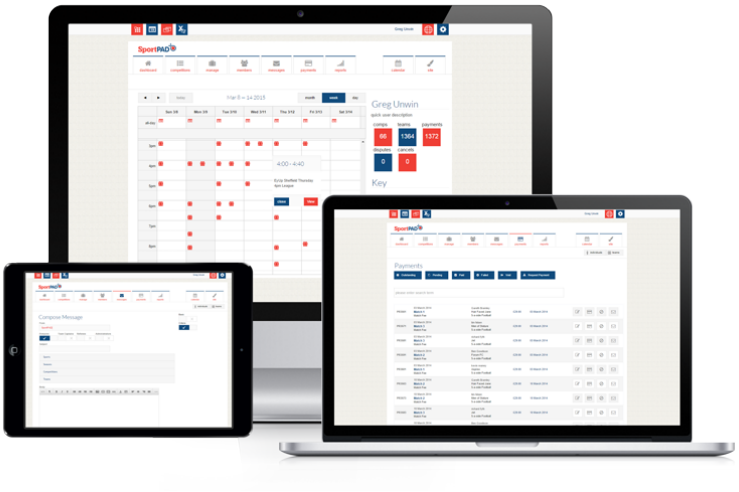
V administraci může uživatel systému do databáze zadávat nové uživatele a týmy, případně upravovat existující údaje. iSport také umožňuje uchovávání dat o rozhodčích a o hřištích, ke konkrétnímu utkání lze přiřadit jak rozhodčí, tak i hřiště, na kterém má utkání proběhnout. Velikou devizou programu je možnost rozlosování soutěže - aplikace automaticky vytvoří rozpis nových utkání, které si uloží do interní databáze.

Systém je schopen generovat komplexní tabulky pro soutěže - obsahují běžné informace, jako je poměr výher a proher, počet vstřelených branek, apod. Zároveň si eviduje i velké množství zajímavých statistik pro jednotlivé hráče, mužstva i soutěže. [1]

Velikým kladem systému iSport je obsáhlost databáze a velký počet statistik, které sleduje. Bohužel celý systém je navržen okolo kolektivních sportů, pro individuální sporty je takřka nevyužitelný.

2.2 SPORTPAD

Systém SportPAD byl vytvořen pro BUCS - asociaci, která sdružuje a dohlíží na sportovní aktivity na univerzitách ve Velké Británii. Je využíván sportovními kluby, organizacemi a fakultami, které pod BUCS spadají. SportPAD lze rozdělit do tří jednotlivých modulů.



Obrázek 2: Ukázka webového systému SportPAD. Zdroj: [2]

První modul - Sport Management - obsahuje informace o hráčích a týmech. Lze v něm vyplnit údaje soutěžících, nebo nastavit případné kapitány týmů. Dále umí generovat zprávy s účastí hráčů na tréninzích a zápasech a stav plateb členských příspěvků.

Modul pro ligy a turnaje je schopen rozlosovat účastníky soutěže a vytvořit rozvrh zápasů. Kapitáni týmů se zde mohou přihlásit, aktualizovat skóre zápasů nebo do soutěže podat přihlášku. Také pro pořadatele sleduje, zda týmy zaplatily registrační poplatky do soutěže.

Součástí SportPAD je také CMS pro psaní článků, nebo různých oznámení, která lze díky integraci se sociálními médii ihned publikovat na Facebook nebo Twitter. Systém umí členům klubu či organizace posílat zprávy - lze je tak ihned upozornit na změny v termínech zápasů nebo na nejnovější výsledky. [2]

2.3 RESULTSVAULT

Systémy ResultsVault pocházejí od australské společnosti InteractSport. Ta momentálně poskytuje celkem pět systémů, každý určen pro jiný sport.

Systémy nabízí možnost online registrací do soutěží pro zájemce, po uzavření registrace vypracuje rozvrh jednotlivých utkání. Pořadatelé mohou

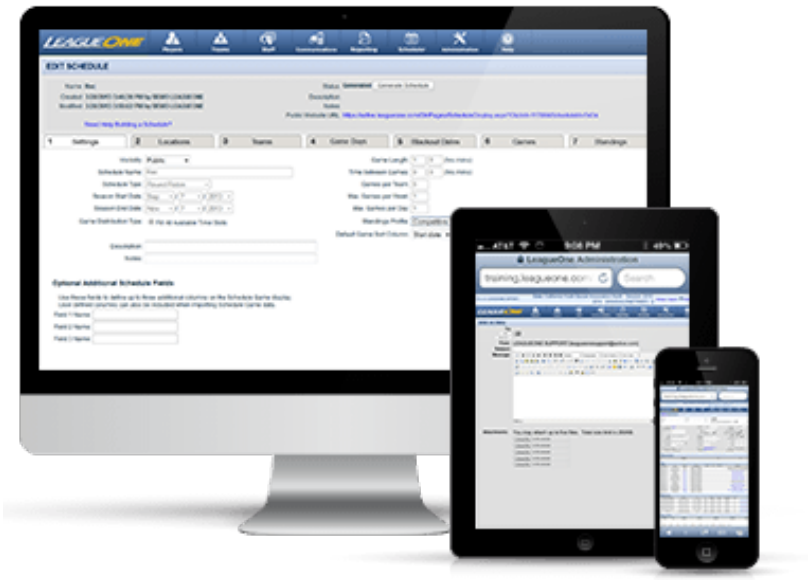
k jednotlivým utkáním přiřadit rozhodčí a systém si sám kontroluje, aby nedocházelo ke kolizím v rozvrhu rozhodčích.

Informační systém dokáže účastníkům soutěží přidělit vlastní uživatelské jméno i heslo, díky čemuž je snížená pracovní zátěž na pořadateli soutěže. Udržování aktuálních dat o účastnících a soutěžích pak není pouze na organizátorech. Samotní soutěžící a týmy si mohou nechat vygenerovat zprávu se svými výkony v soutěži. Je zde i podpora několika užitečných funkcí pro sportovní kluby - lze hromadně notifikovat členy týmu pomocí emailu a SMS o změnách nebo výsledcích utkání. Kluby také mohou odevzdat svojí nominaci k zápasů přes webové rozhraní systému. [3]

2.4 ACTIVE SPORTS

Americká společnost Active Network ve spolupráci s více než patnácti sportovními organizacemi ve Spojených Státech vyvinula celkem 16 webových systémů pro pořadatele sportovních lig, turnajů a závodů. Každý z těchto systémů je určen buď pro konkrétní sport, nebo odvětví (např. systém určený pro závodní sporty). Tyto systémy jsou relativně hodně podobné systémům ResultsVault.

Active Sports také podporují online registrace, ale je zde možnost sledovat zda byly odevzdané potřebné dokumenty - může se jednat např. o lékařské zprávy nebo souhlas pro pořizování fotografií. Pokud požadované dokumenty nebyly předloženy, systém automaticky posílá hříšníkům připomínky ve formě emailu. Active Sports obsahuje nástroj pro rozlosování a generování rozpisu zápasů. Pořadatel si může v systému nechat vygenerovat webovou stránku s CMS, kde má k dispozici má několik šablon.



Obrázek 3: Ukázka webového systému od Active Network. Zdroj: [4]

Soutěžící mohou mít v systému vlastní účty, ten jim na vyžádání může vypracovat zprávu s výkony majitele účtu v soutěži. Také si zde mohou zjistit, zda mají zaplacené registrační poplatky nebo odevzdané požadované do-

kumenty. Narozdíl od ResultsVaults neexistuje možnost notifikovat ostatní členy týmu.

Systémy od Active Network jsou velice komplexní a obsahují téměř veškerou funkcionalitu pro organizaci sportovních soutěží. Možnost sledovat, zda jsou k dispozici všechny potřebné dokumenty od účastníků je pro sportovní systémy unikátní. Dalším kladem je, že citlivá data jsou zabezpečena pomocí SSL a platební údaje podléhají PCI standardům. Bohužel Active Network své služby poskytuje výhradně klientům ve Spojených Státech. [4]

2.5 TIOPRO

Program TioPro je zadarmo poskytovaný firmou AllIsNetwork. Cílen je pouze na organizátory soutěží, kterým má pomoci s logistikou. V první řadě si uživatel programu musí do interní databáze zadat potřebná data o účastnících turnaje.

Následně soutěžící rozlosuje, podporuje i dokonce vyřazovací systém na dvě porážky a nasazování hráčů. TioPro sleduje několik zajímavých statistik - ve vyřazovacím systému počítá jak dlouho hráč čekal na svého soupeře, podle délky jednotlivých zápasů a počtu dostupných stanovišť (mohou být hřiště, hrací stoly, apod.) i počítá předpokládaný čas do konce turnaje. Program dokáže spočítat i rozdělení výher pro první tři místa a je schopen pro každého soutěžícího vypracovat tabulku s výsledky jeho zápasů. [5]

Aplikace byla naprogramována pomocí .NET frameworku a běží pouze na platformě Windows. Jejím kladem je podpora pro využívanou netradiční formu vyřazovacího systému a evidováním poměrně velkého množství statistik. Záporem je, že si neumí poradit s týmovými sporty a neukládá si informace o průběhu utkání, pouze konečné skóre.

3 POPIS PROBLÉMU, SPECIFIKACE CÍLE

3.1 POPIS PROBLEMATIKY

Aplikace používané ke správě sportovních utkání musí být dostatečně komplexní. Je nutné, aby si uchovávala informace o hráčích, sportovních týmech, jednotlivých utkání i soutěžích. Aby uživateli mohla prezentovat relevantní informace a statistiky, musí být schopna si interní data interpretovat v kontextu. Uživatel do systému přes webové rozhraní dodá informace o hráčích, případně jaké týmy reprezentují, jejich vliv na utkání (např. vstřelené branky, asistence, apod.) a konečný výsledek. Z těchto dat systém stanoví pořadí účastníků v soutěži a sleduje zajímavé statistiky pro jednotlivé účastníky (např. které hráč měl největší vliv na konečném stavu utkání, nejvíce vstřelených branek, obdržel nejvíce červených karet, apod.). Tyto vyhodnocené údaje pak může uživateli nabídnout k vytištění.

Další důležitou funkcí je možnost rozlosování jednotlivých utkání v soutěži. Organizátorům tak odpadá nutnost vytvářet rozvrhy manuálně, čímž by se snížila jejich pracovní zátěž.

3.2 ODŮVODNĚNÍ NOVÉHO ŘEŠENÍ

Z provedené rešerše o existujících systémech vyplývá, že systémy pro správu sportovních aktivit jsou v České republice poměrně nedostupné. Jediným nalezeným systémem byl systém *iSport*, který je ale značně zastaralý a neobsahuje požadovanou funkci pro konfigurovatelné tiskové výstupy.

Všechny systémy v analýze obsahovaly podporu buď pro jediný sport (*ActiveSports*, *ResultsVault*) nebo odvětví (*iSport* a *SportsPAD* pro týmové sporty; *TioPro* pro individuální sporty). Tento přístup, kdy se systém zaměřuje na specifický sport nebo odvětví, se jeví jako nejvhodnější.

Výsledný systém bude podporovat jeden sport - fotbal a sporty jemu podobné (např. futsal, florbal). Důraz bude kladen na rychlou organizaci soutěží.

3.3 SPECIFIKACE CÍLE

Cílem bakalářské práce je prozkoumat systému používané ke organizaci sportovních událostí, navrhnout takový systém s podporou pro týmové sporty (konkrétně fotbal), implementovat ho pomocí požadované technologie a řádně výsledný systém otestovat. Mají být prozkoumány možnosti frameworku .NET od firmy Microsoft.

4 ANALÝZA BUDOUCÍHO ŘEŠENÍ

V této kapitole byla provedena analýza systému, který má být implementován. Byl zpracován přehled požadavků na informační systém, vytvořen doménový model mapující problematiku organizace fotbalových soutěží a byly vypracovány jednotlivé případy užití včetně scénařů.

4.1 PŘEHLED POŽADAVKŮ

Seznam požadavků vymezuje schopnosti budoucího systému. Mělo by z nich být poznat co systém bude dělat, umět a jaké funkcionality budou implementovány. Požadavky nijak nespecifikují jak systém bude požadovanou funkcionalitu provádět ani jak bude implementována. [6]

4.1.1 Funkční požadavky

Funkční požadavky popisují požadované funkce a chování systému. [6]

- **REQ-F01 - Přihlašování do systému** - Systém bude umožňovat uživatelům přihlásit se do systému. Uživatelé se budou do systému přihlašovat pomocí uživatelského jména a hesla.
- **REQ-F02 - Správa uživatelů v systému** - Systém bude umožňovat uživatelům s potřebnými administrátorskými právy upravovat informace o jiném existujícím uživateli.
- **REQ-F03 - Správa informací o hráčích** - Systém bude evidovat informace o hráčích. Tyto informace budou dodávány uživateli systému.
- **REQ-F04 - Správa informací o týmech** - Systém bude evidovat informace o sportovních týmech. Potřebné informace o týmech poskytnou uživatelé systému.
- **REQ-F05 - Správa informací o sportovních soutěžích** - Systém bude umět evidovat informace o sportovních soutěžích. Data o soutěžích budou do systému zadávat jeho uživatelé.
- **REQ-F06 - Správa informací o sportovních utkáních** - Systém bude schopen evidovat informace o sportovních utkáních, včetně jeho průběhu. Potřebné údaje poskytují uživatelé systému.
- **REQ-F07 - Rozlosování soutěží** - Systém bude umět rozlosovat soutěž do sportovních utkání. Uživatel si bude moci vybrat z více způsobů.
- **REQ-F08 - Vyhodnocování soutěží** - Systém bude vyhodnocovat soutěž i během jejího průběhu (tzn. turnaj nebo liga ještě nebyla uzavřená a nejsou k dispozici výsledky všech utkání).
- **REQ-F09 - Tisk diplomů** - Systém bude umět vytisknout diplomy po uzavření soutěže. Tyto diplomy si uživatel může přizpůsobit s pomocí vlastních CSS stylů.
- **REQ-F10 - Tisk zpráv s výsledkami soutěží** - Systém bude umět vytisknout zprávy s výsledky soutěží. Součástí těchto zpráv budou i vy-

hodnocené relevantní statistiky (např. který hráč vstřelil nejvíc branek, apod.).

4.1.2 Nefunkční požadavky

Nefunkční (jinak také obecné) požadavky se týkají omezení systému a způsobu implementace. [6]

- **REQ-N01 - Použití technologie .NET** - Systém bude navržen a implementován pro prostředí ASP.NET.
- **REQ-N02 - Webové rozhraní** - Systém bude s uživateli komunikovat přes webové prostředí.
- **REQ-N03 - Lokalizace** - Systém bude lokalizován do anglického a českého jazyka.

4.2 DOMÉNOVÝ MODEL

Doménový model má za úkol zmapovat danou problematiku, definovat jednotlivé entity, vztahy mezi nimi a jejich chování. Nabízí tak celkový pohled na vyvíjený systém.

Diagram s doménovým modelem má být statický, nemá se v něm nacházet žádný sled událostí v systémů a měl by být nezávislý na platformě - implementace se v něm ve větší míře neodráží a nenacházejí se v něm datové typy atributů. [6]

4.2.1 Entity

Entity představují jednotlivé objekty a třídy, které se podílí na dané problematice. Do diagramů se zakreslují pomocí diagramů tříd (*Class diagram*). [7]

Player - Hráč

Entita `Player` představuje hráče.

Atributy:

- `Name` - jméno hráče.
- `Number` - číslo hráče.
- `PlayerPosition` - pozice, kterou hráč zastupuje. Každý hráč může zastupovat jednu ze čtyř pozic:
  - `Goalkeeper` - brankář.
  - `Defender` - obránce.
  - `Midfielder` - záložník.
  - `Striker` - útočník.
- `Active` - značí zda je hráč aktivní. Neaktivní hráči se nemohou zúčastňovat soutěží.

Team – Tým

Týmem se rozumí skupina hráčů.
Atributy:

- `Name` - jméno týmu.
- `Active` - značí zda je hráč aktivní. Neaktivní tým se nemůže zúčastňovat soutěží ani nabírat nové hráče.

Membership – Členství

`Membership` je asociační třída pro vztah mezi entitami `Player` a `Team`. Tato třída řeší případ, kdy hráč přestoupí do jiného týmu (entita `Player` totiž může být součástí pouze jednoho týmu.).
Atributy:

- `DateJoined` - kdy se hráč přidal k týmu.
- `DateLeft` - kdy hráč tým opustil.

CompetingTeam – Soutěžící Tým

Entitou `CompetitingTeam` se rozumí tým, jako účastník soutěže. Je generalizací třídy `Team`.
Atributy:

- `Placement` - umístění v soutěži.

CompetingPlayer – Soutěžící Hráč

Entita `CompetitingPlayer` představuje hráče, který může nastoupit do zápasu a má přímý vliv na stav utkání. Je generalizací třídy `Player`.

Competition – Soutěž

Soutěží může být buď sportovní turnaj nebo liga.
Atributy:

- `Name` - jméno soutěže.
- `DateStart` - datum začátku soutěže.
- `DateEnd` - datum konce soutěže.
- `CompetitionType` - typ soutěže:
  - `SingleRoundRobin` - soutěžící hrají každý s každým jednou.
  - `SingleElimination` - vyřazovací systém na jednu porážku.
  - `DoubleRoundRobin` - soutěžící hrají každý s každým dvakrát.
  - `Custom` - vlastní, uživatelem definovaný, formát soutěže.

Round – Kolo

Entitou `Round` se rozumí fáze soutěže.
Atributy:

- `Title` - označení fáze soutěže.
- `GroupStage` - zda se jedná o skupinovou fázi soutěže.

- `RoundType` - typ jednotlivého kola:
  - `OneMatch` - účastníci kola mezi sebou odehrají jeden zápas.
  - `TwoMatches` - účastníci kola mezi sebou odehrají dva zápasy.

Match – Utkání

Tabulka představuje zápas mezi dvěma účastníky v soutěži.
Atributy:

- `Date` - datum odehrání zápasu.
- `NeutralGround` - zda se zápas odehrává na neutrálním hřišti.

MatchEvent – Událost v utkání

Abstraktní třída `MatchEvent` představuje jednotlivé události v utkání, které mohou nastat.
Atributy:

- `Minute` - označuje minutu utkání, ve kterou nastala ona událost.

Goal – Gól

Představuje branku v utkání.
Atributy:

- `GoalType` - typ vstřelené branky:
  - `Regular` - obyčejná branka.
  - `OwnGoal` - vlastní gól.
  - `SetPiecePenalty` - standartní situace - penalta.
  - `SetPieceCornerKick` - standartní situace - rohový kop.
  - `SetPieceOther` - jiná standartní situace.

Substitute – Nahrazení

Situace, kdy byl nahrazen hráč na hřišti jiným hráčem ze stejného týmu.

Foul – Faul

Nedovolené chování hráče.

Card – Karta

Situace, kdy hráč obdržel kartu, jako napomenutí za prohřešek proti pravidlům.
Atributy:

- `CardType` - typ obdržené karty.
  - `FirstYellow` - hráč obdržel první žlutou kartu.
  - `SecondYellowRed` - hráč obdržel druhou žlutou kartu a následně i červenou.
  - `FirstRed` - hráč obdržel červenou kartu bez toho, aniž by předtím obdržel žlutou.

4.2.2 Vztahy mezi entitami

Vztahy mezi dvěma třídami bývají také nazývány jako asociace. V diagramu jsou označeny pomocí čar a šipek mezi jednotlivými entitami. [7]

Membership mezi třídami Player a Team

Hráč může být pouze členem jednoho sportovního týmu (systém nebere ohled na možnost hostování, je pro něj pouze relevantní kdo v utkání může za klub nastoupit a kdo ne). Tým může mít libovolné množství hráčů. Tento vztah je znázorněn pomocí *asociační třídy*.

Participates in mezi třídami Competition a CompetingTeam

Soutěže se může zúčastnit libovolný počet týmů. Tým není nijak omezen v počtu soutěží, do kterých se může zapsat.

Consists of mezi třídami Competition a Round

Každá soutěž se skládá z jednoho a nebo více soutěžních kol (osmifinále, čtvrtfinále, jednotlivá kola v ligách, ...).

Rekurzivní vztah Advanced to/from mezi třídami Round

Reprezentuje situaci v turnajích, kdy vítěz kola (nebo určitý počet soutěžících ze skupiny) postupuje do dalšího kola. Toto další kolo může být rovněž další skupinovou fází, takže se může stát, že jsou v něm soutěžící z více předchozích kol. V některých soutěžích (např. liga, nebo turnaj kde hraje každý s každým pouze jednou) tento vztah není potřeba.

Has mezi třídami Round a Match

Každé kolo se skládá z jednoho nebo více zápasů.

Home/Away Team mezi třídami Match a Competing Team

Označuje soutěžící, kteří se zúčastnili konkrétního zápasu. Ti jsou sice evidováni jako domácí, resp. venkovní tým, ale utkání se může odehrát na neutrální půdě. Tato možnost zajištěna pomocí atributu ve třídě `Match`.

Nominated to mezi třídami Match a CompetingPlayer

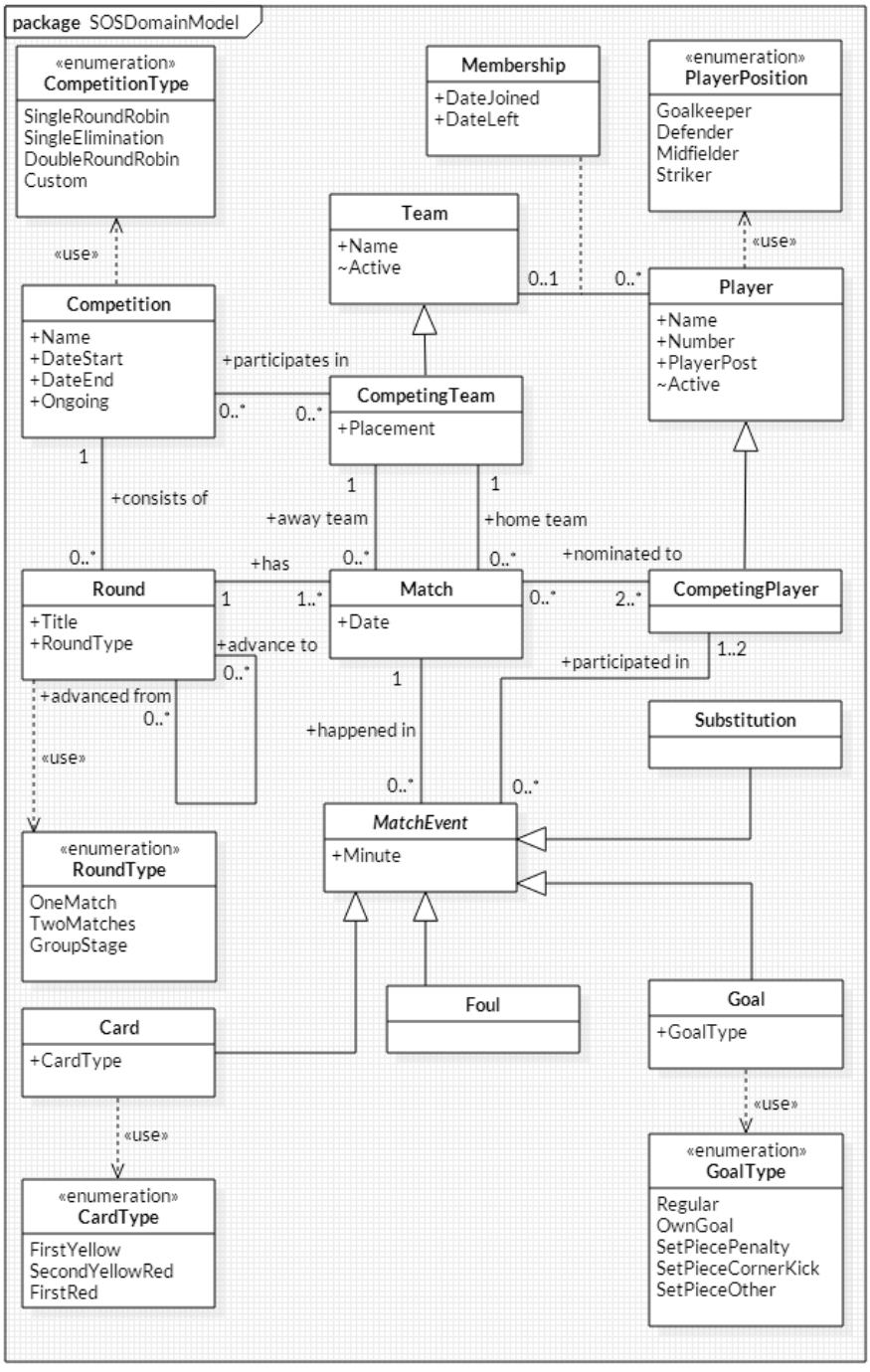
Vztah značí, že daný hráč nastoupil do utkání.

Happened in mezi třídami MatchEvent a Match

V jednom zápase se může odehrát více významnějších událostí nebo situací (góly, obdržené karty, apod.).

Participated in mezi třídami MatchEvent a CompetingPlayer

Hráč, který nastoupil do zápasu může mít svůj podíl na události v utkání (např. vstřelil gól). Na některých událostech se musí/může podílet více hráčů (např. střídání).



Obrázek 4: Diagram s doménovým modelem

4.3 DEFINICE PŘÍPADŮ UŽITÍ

Diagramy případů užití mají přiblížit fungování systému. Každý z případů užití zachycuje interakci uživatele se systémem a může mít scénář, který tuto interakci popisuje, obsahuje jednotlivé kroky (z pohledu uživatele) a jak na ně systém reaguje. [6]

Scénáře byly vypracovány pouze k netriviálním případům užití.

4.3.1 Aktéři systému

Přístup do systému bude umožněn pouze s validnímu přístupovými údaji - aktéry systému jsou pouze přihlášení uživatelé. Ti mohou zastupovat dvě role v systému:

Vlastník systému

`Vlastník systému` může upravovat přihlašovací údaje jiným uživatelům. Také smí uživatele přidávat nebo případně mazat.

Organizátor

Uživatel typu `Organizátor` má oprávnění v systému spravovat a tisknout informace o hráčích, týmech a soutěžích, může tisknout diplomy a upravovat šablony používané k tisku. Nemá přístup k přístupovým údajům jiných uživatelů.

4.3.2 Správa uživatelů

V této části jsou namapované případy užití na požadavek *REQ-F02 - Správa uživatelů v systému*.

Vytvořit nového uživatele

Uživatel typu `Vlastník systému` bude moci do systému přidat nového uživatele.

Scénář: «Basic Flow»

1. Případ užití začíná, když uživatel vybere možnost přidat nového uživatele do systému.
2. Systém zobrazí formulář pro přídání nového uživatele.
3. Uživatel formulář vyplní a svojí volbu potvrdí.
4. Systém vytvoří nového uživatele.

«Alternate»

3.A) Uživatel zruší přidávání nového uživatele do systému.
   4.A) Systém zruší přidání nového uživatele.

4.B) Pokud uživatelské jméno v systému již existuje, nebo heslo neodpovídá požadovaným parametrům, systém nového uživatele nevytvoří a opakuje se krok 3.

Upravit uživatele

Uživatel s rolí `Vlastník systému` bude smět v systému upravit přihlašovací údaje existujícího uživatele.

Scénář: «Basic Flow»

1. Případ užití začíná, když uživatel bude chtít upravit data o existujícím uživateli.
2. Systém zobrazí formulář s přihlašovacími údaji existujícího uživatele.
3. Přihlášený uživatel formulář vyplní a svojí volbu potvrdí.
4. Systém uloží změny a uživatele upozorní na úspěšné přidání.

«Alternate»

3.A) Uživatel zruší úpravu existujícího uživatele.
   4.A) Systém zruší úpravy uživatele.

4.B) Pokud nové uživatelské jméno v systému již existuje, nebo heslo neodpovídá požadovaným parametrům, systém úpravy neuloží a opakuje se krok 3.

Smazat uživatele

Uživatel `Vlastník systému` bude mít oprávnění odstranit existujícího uživatele ze systému.

Scénář: «Basic Flow»

1. Případ užití začíná, když uživatel vybere možnost smazat existujícího uživatele.
2. Systém upozorní uživatele na nevratnost této volby a požádá o potvrzení.
3. Uživatel potvrdí svojí volbu.
4. Systém existujícího uživatele ze systému vymaže a zobrazí notifikace o úspěšném vymazání.

«Alternate»

3.A) Uživatel zruší odstranění existujícího uživatele.
   4.A) Systém uživatele ze systému nesmaže.

4.3.3 Správa hráčů

V této sekci jsou popsány případy užití, které korespondují s požadavkem *REQ-F03 - Správa informací o hráčích*.

Vytvořit hráče

Uživatel typu `Organizátor` bude mít oprávnění přidávat nové hráče do systému.

Upravit hráče

Role `Organizátor` bude umožňovat uživateli upravovat informace o existujícím hráči.

Smazat hráče

`Organizátor` bude moci odstraňovat existující hráče ze systému.

4.3.4 Správa týmů

Zde jsou případy užití odpovídající požadavku *REQ-F04 - Správa informací o týmech*.

Vytvořit nový tým

Uživatel s rolí `Organizátor` bude moci přidat nový tým do systému.

Upravit tým

Uživatel typu `Organizátor` bude umožněno upravovat informace o stávajících týmech.

Smazat tým

`Organizátor` bude moci odstranit existující tým ze systému.

4.3.5 Správa soutěží

Tato část obsahuje případy užití vztahující se k požadavkům *REQ-F05 - Správa informací o sportovních soutěžích* a *REQ-F07 - Rozlosování soutěží*.

Vytvořit soutěž

Uživateli `Organizátor` bude dovoleno do systému přidávat nové soutěže.

Upravit soutěž

Role `Organizátor` bude uživateli umožňovat upravit informace o existující soutěži v systému, manipulovat se seznamem soutěžících a upravovat formát soutěže (zda se jedná o ligu nebo turnaj, jak se konkrétně budou odehrávat jednotlivá kola, apod.).

Smazat soutěž

`Organizátor` bude mít možnost smazat existující soutěž se systému.

Rozlosovat soutěž

Uživatel typu `Organizátor` bude moci rozlosovat existující soutěžící do jednotlivých utkání.

Prerekvizity

1. Do soutěže byli přidáni všichni soutěžící.
2. V soutěži byl již vybrán požadovaný formát.

Scénář: «Basic Flow»

1. Případ užití začíná, když uživatel bude chtít rozlosovat existující soutěž.

2. Uživatel zvolí, jakým způsobem bude soutěž rozlosována (náhodně, nebo podle hodnocení týmů v žebříčku).
3. Systém rozlosuje jednotlivé účastníky do jednotlivých kol.

4.3.6 Tisk zpráv a diplomů

Tato část obsahuje požadavky korespondující s požadavky *REQ-F09 - Tisk diplomů* a *REQ-F10 - Tisk zpráv s výsledkami soutěží*.

Vytisknout zprávu

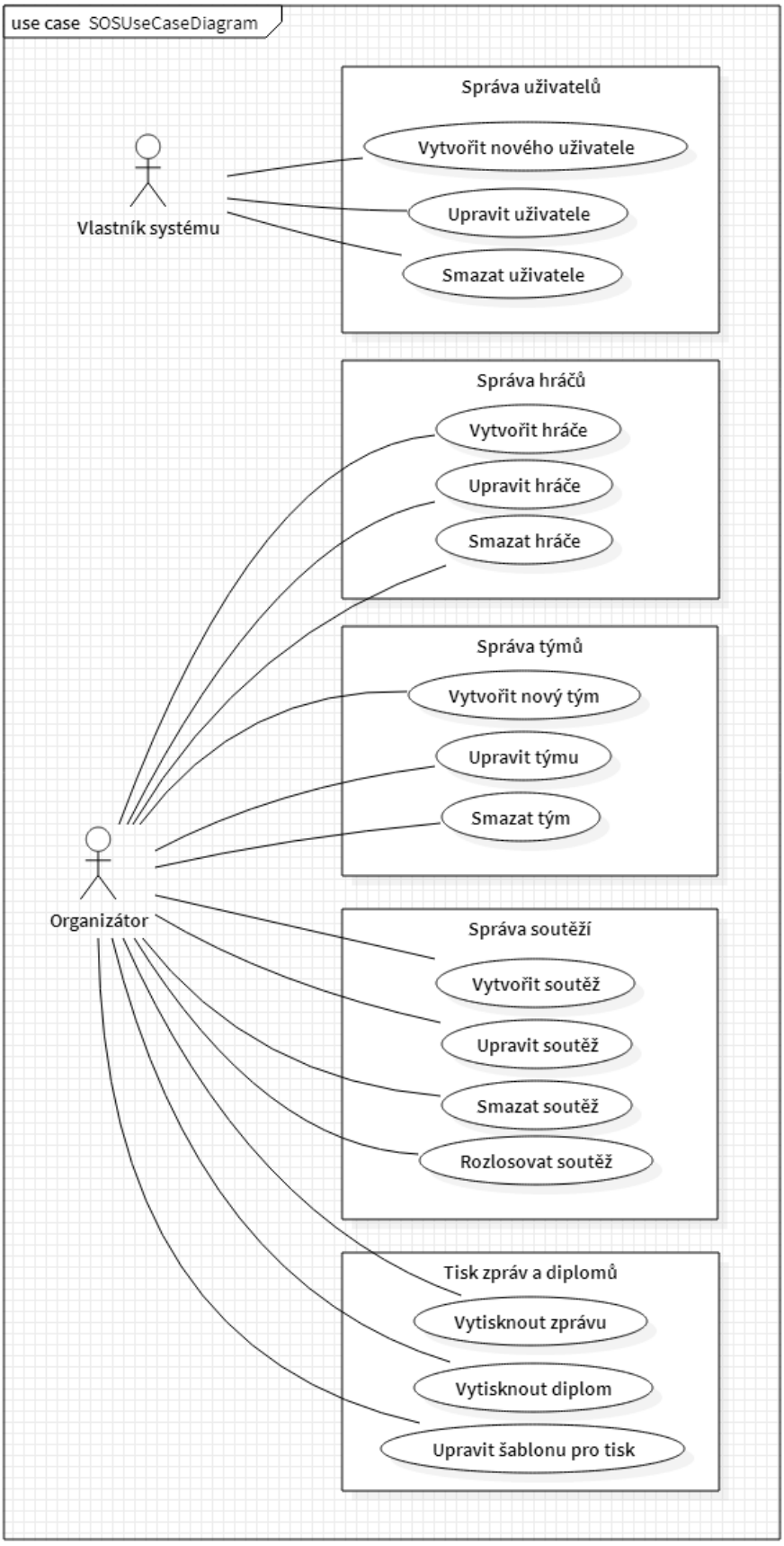
Uživatel typu `Organizátor` bude mít možnost vytisknout zprávu s výsledkami soutěží, nebo o jednotlivých hráčích, týmech nebo utkání.

Vytisknout diplom

Uživatel s rolí `Organizátor` bude moci tisknout diplomy pro jednotlivé hráče a týmy.

Upravit šablonu pro tisk

Uživateli `Organizátor` bude umožňěno upravovat šablony pro tisk diplomů pomocí jazyka CSS.



Obrázek 5: Diagram s případy užití

5 REALIZACE

5.1 ZVOLENÉ TECHNOLOGIE

5.1.1 Programovací jazyk C# a .NET

.NET je robustní framework poskytovaný společností Microsoft pod MIT open-source licencí. Primárně se pro vývoj využívá vývojové prostředí Visual Studio a objektově-orientovaný jazyk C#, jehož kompilátor je přímo určený pro .NET. Důsledkem toho je, že prakticky každý kód, který je napsaný v jazyce C# běží na platformě .NET. Součástí balíčku jsou nástroje a knihovny pro tvorbu aplikací na platformu Windows, interakci s databázemi, soubory, apod. [8]

ASP.NET MVC

ASP.NET MVC je framework využívající technologii ASP.NET, která určená pro tvorbu webových stránek, aplikací a služeb. Lze s ní využít jazyky podporované v .NET, HTML, CSS a JavaScript. Samotný framework je inspirovaný Model-View-Controller architekturou.

Serverovou část, která zahrnuje komponenty *model* a *controller*, je typicky vytvořena jazykem C# a .NET frameworkem. Klientská částí aplikací tvoří komponenta *view* tvořená HTML kódem, JavaScriptou a nezbytným malým množstvím kódu v jazyce C#. ASP.NET MVC integruje dvojice *controller* a *view* do rozhraní, čímž ulehčuje vývojářům testování aplikace. [8]

ADO.NET

ADO.NET je soubor nástrojů (Entity Framework, LINQ) ve frameworku .NET, která má na starosti komunikaci s databází. Využívá se k přístupu a změně dat v databázích. [8]

Razor view-engine

Technologie Razor, dodávaná s balíčkem ASP.NET MVC, má na starosti vykreslování komponenty *view* v MVC aplikaci. Sémantikou připomíná jazyk HTML obohacený o syntaxi převzatou z jazyků Visual Basic a C#. Jednou z výhod Razoru je poměrně jednoduché testování pomocí unit testů, není k tomu nutný speciální *controller* ani web server. [9]

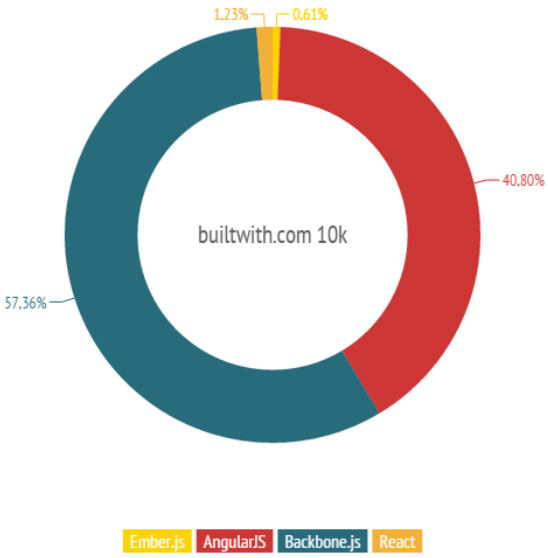
5.1.2 AngularJS

AngularJS je jedna z JavaScriptových open-source knihoven a frameworků pro budování dynamických webových aplikací a stránek. Cílem knihovny mají vývojářům usnadnit práci, například k vytvoření tabulky v klasickém JavaScriptu může zabrat několik řádek kódu, který lze nahradit mnohdy jedinou řádkou kódu odkazující se funkci v knihovně. Kromě efektivnější

a rychlejší práce má použití frameworku další výhodu - zaručují, že se aplikace bude chovat všude stejně (např. na tabletu, mobilním prohlížeči, apod.). [10]

Vše začalo v roce 2009 jako vedlejší projekt zaměstnance Googlu Miška Heveryho a jeho přítele Adama Abronse. [11] Abrons od té doby projekt opustil, ale nyní na něm společně s Heverym spolupracují další zaměstnanci Googlu - Igor Minár a Vojta Jína. [12]

Podle průzkumu JavaScriptových Model-View-* frameworků, který byl proveden společností MobileVision v březnu 2015, má AngularJS druhý největší podíl na trhu, tzn. je druhou nejpoužívanější knihovnou ze čtyř zkoumaných. Zároveň má i nejvyšší počet přispěvatelů (*contributors*) na verzovacím systému GitHub.org, což svědčí o jeho popularitě. [13]



Obrázek 6: Srovnání JavaScriptových MV* frameworků podle jejich podílu na trhu. Zdroj: [13]

Tento framework je unikátní tím, že jako jediný pro svoje komponenty podporuje návrhový vzor *dependency injection* (česky *vkládání závislostí*) a také má velice jedinečný přístup k psaní webových aplikací. Vývojář může HTML kód stránky obohatit takzvanými *directives* (do češtiny by se dalo přeložit jako směrnice, nebo pokyny), které se váží ke konkrétnímu DOM elementu. [10] [11] HTML kompilátor v AngularuJS díky těmto *directives* může zmíněný DOM element změnit (je dokonce možné změnit i jeho potomky), nebo provést jinou akci.

5.1.3 Less

Less je JavaScriptová open-source knihovna, sloužící jako nadstavba CSS (kaskádových stylů), jazyka který popisuje jak se mají zobrazit elementy na stránkách napsaných pomocí HTML, XHTML nebo i XML. Syntaxe v Less umožňuje využití proměnných a tzv. *mixins*, díky kterým vlastnosti elementu (např. barva, kulaté rohy, atd.) stačí v kódu definovat na jediném místě. V klasickém CSS se velice často stává, že se společné vlastnosti musí definovat zvlášť pro každý element. Knihovna obsahuje velké množství

funkcí, které hlavně ulehčující manipulaci s barvami a obrázky v CSS. Navíc oproti sémantice v CSS lze v Less organizovat kód do stromové struktury, selektor (HTML element definovaný pomocí tagů *id* a *class*) lze pak definovat pod jiným selektorem a celkové uspořádání kódu tím více připomíná HTML. [14]

Díky těmto vlastnostem je kód napsaný v Less mnohem přehlednější a usnadňuje se tím jeho úprava. Nicméně tento kód je stále nutné převést do klasického CSS pomocí kompilátoru, který je dostupný i pro platformu .NET. Případně lze použít JavaScriptový kompilátor nacházející se v knihovně, výsledný CSS kód se poté kompiluje v klientské straně (tzn. v prohlížeči a ne na serveru). [15]

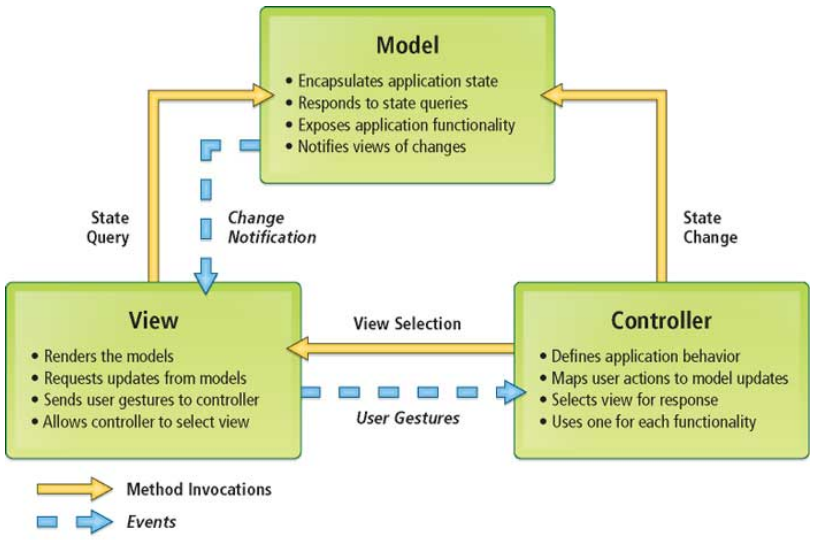
5.1.4 Team Foundation Server & TFS Version Control

Team Foundation Server je centralizovaný verzovací systém poskytován firmou Microsoft. Úkolem těchto systémů je správa změn vzniklých v průběhu softwarového projektu. Všechny změny jsou evidovány jako revize, ve kterých je zaznamenáno kdo je provedl, kdy a které soubory byly modifikovány. [16]

Přínos TFS byl značný, neboť k implementaci byly využity dva počítače - TFS zajistil, že soubory projektu byly konzistentní na obou počítačích.

5.2 MODEL-VIEW-CONTROLLER

Architektura Model-View-Controller (zkráceně MVC) byla poprvé představena na konci 70. let programátory Smalltalku ve firmě Xerox PARC, kteří se tehdy zabývali vývojem přenosného počítače Dynabook. Účelem MVC bylo zajistit, aby interakce uživatele s počítačem (hlavně manipulace dat) korespondovala s digitálním modelem uvnitř zařízení. Toho bylo docíleno rozdělením zodpovědností (*Separation of Concerns*) do tří jednotlivých komponent. [17]



Obrázek 7: Rozdělení zodpovědnosti v MVC. Zdroj: [18]

5.2.1 Model

Model (česky stejně *model*) je definován jako entity nacházející se v dané problematice. [19]

V systému se konkrétně jedná o třídy v jazyce C#, které se nacházejí ve složce `SportSystem/Models`. Data jsou uložena v Microsoft SQL Server databázi, k přístupu k těmto datům se pak využívá Entity Framework, který je součástí ADO.NET. Databázový model se nachází v příloze B.2.

5.2.2 View

View (česky *pohled*) reprezentuje informace, viditelná uživatelem. Někdy je tato komponenta také označována jako vizuální reprezentace modelu. [19]

Implementace komponent *view* se nacházejí ve složce `SportSystem/Views`. K jejich napsání byla využit zejména jazyk technologie Razor a AngularJS pro komunikaci s částí *controller*.

5.2.3 Controller

Controller (česky *řadič*) interpretuje vstupy uživatelů systému a zajišťuje adekvátní změny v komponentách *model* a *view*. [19]

Soubory s implementací komponenty *controller* napsané v jazyce C# jsou ve složce `SportSystem/Controllers`. Tyto třídy spravují základní CRUD operace s datovými entitami v systému a případný export a import dat pomocí dotazovacích metod GET a POST v HTTP.

5.3 VÝSLEDNÝ INFORMAČNÍ SYSTÉM

Vzniklá aplikace byla pojmenována Sports Organizing System, tedy zkráceně SOS.

Přehled funkcionalit byl rozdělen podle požadavků popsaných v minulé kapitole. Ukázky kódu jsou dostupné v příloze C.

Veškeré dotazovací HTTP metody, které jsou opsány mohou jsou ošetřeny proti zneužití - jsou přístupné pouze pro přihlášené uživatele.

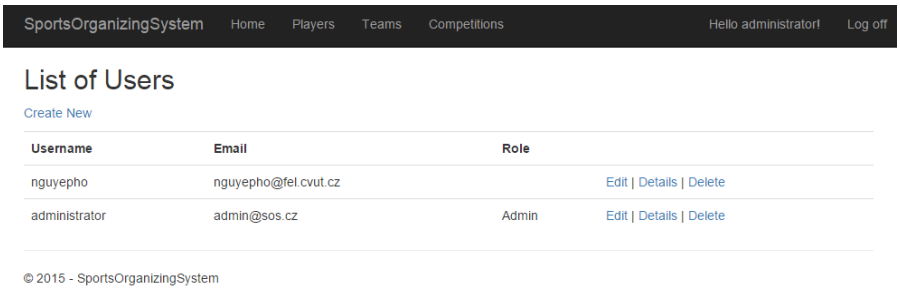
5.3.1 Správa uživatelů

O správu uživatelů se starají třídy `AccountViewModel` a `SOSUser`. Logiku přihlašování, odhlašování a registrace uživatelů byla vygenerována Entity Frameworkem ve třídě `AccountController`, kterou bylo nutné upravit vzhledem k vlastnostem systému (nového uživatele nemůže přidávat kdokoliv a byla přidána definice uživatelských rolí).

Dále bylo potřeba naimplementovat chybějící funkcionalitu ohledně upravování a mazání uživatelů ve třídě `UsersController` a dopsat k nim adekvátní komponenty *view*.

Ke správě uživatelů jsou používány tyto komponenty *view*:

- `Views/`
  - `Account/`
    - `Login.cshtml`
    - `Manage.cshtml`
    - `Register.cshtml`
  - `Users/`
    - `Delete.cshtml`
    - `Index.cshtml`



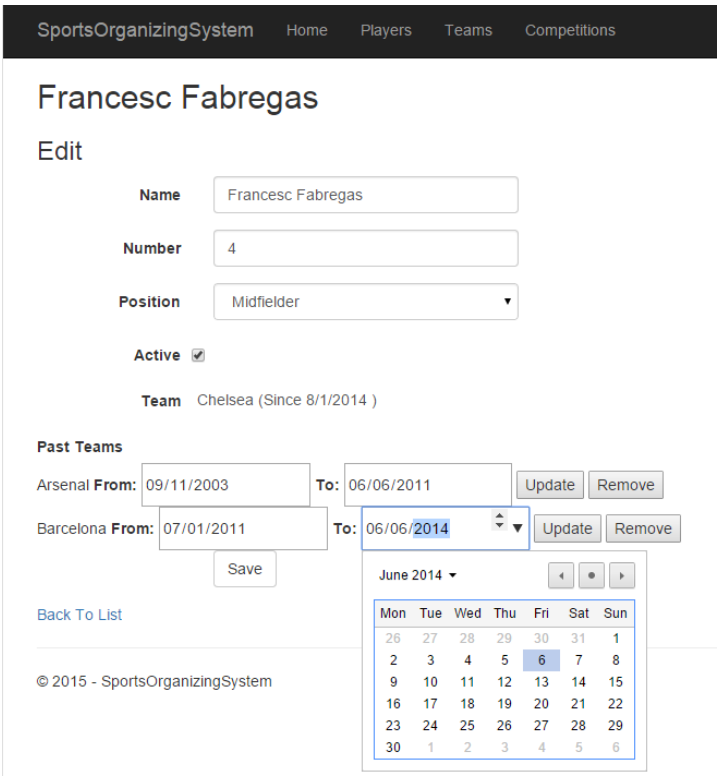
Obrázek 8: Sekce seznamu uživatelů v systému

5.3.2 Správa hráčů a týmů

Třídy `SOSPlayer` a `SOSTeam` popisují entity `Player` a `Team` z doménového modelu. Třídy `PlayersController` a `TeamsController` obsahují logiku přidávání a odebírání hráčů do/z týmu pomocí dotazovacích metod typu HTTP POST a GET.

Seznam *view*:

- `Views/`
  - `Players/`
    - `Create.cshtml`
    - `Delete.cshtml`
    - `Details.cshtml`
    - `Edit.cshtml`
    - `Index.cshtml`
  - `Teams/`
    - `Create.cshtml`
    - `Delete.cshtml`
    - `Details.cshtml`
    - `Edit.cshtml`
    - `Index.cshtml`



Obrázek 9: Upravování informací o hráči v systému

GET /Players/GetPastTeams/:id

Tato metoda vrací seznam ve formátu JSON, obsahuje týmy, jehož byl daný hráč (identifikován podle `:id`) členem včetně data (ve formátu *DateTime*), kdy se k němu připojil a kdy ho opustil.

POST /Players/UpdateMembership

Upraví informace o entitě `Membership` popsané v doménovém modelu. Povinné parametry `from` a `to` přijímá pouze ve formátu *DateTime*, který je specifický pro .NET. Při práci s touto metodou je nutné na klientské straně převádět mezi objekty typu *DateTime* a *Date*, který používá jazyk JavaScript.

Parametry:

- `membershipId` - ID existující entity `Membership`.
- `from` - datum od.
- `to` - datum do.

POST /Players/RemoveMembership

Odstraní entitu `Membership`.

Parametry:

- `membershipId` - ID existující entity `Membership`.

GET /Teams/GetDraftablePlayers

Vrací JSON objekt, který obsahuje seznam hráčů, kteří se mohou přidat k nějakému týmu. K týmu se mohou přidat pouze hráči, kteří jsou stále aktivní (parametr `Active` v entitě `Player`) a dosud nepatří k žádnému týmu.

GET /Teams/GetRoster/:id

Vrátí seznam hráčů, kteří patří k danému týmu.

POST /Teams/AddPlayer/:id

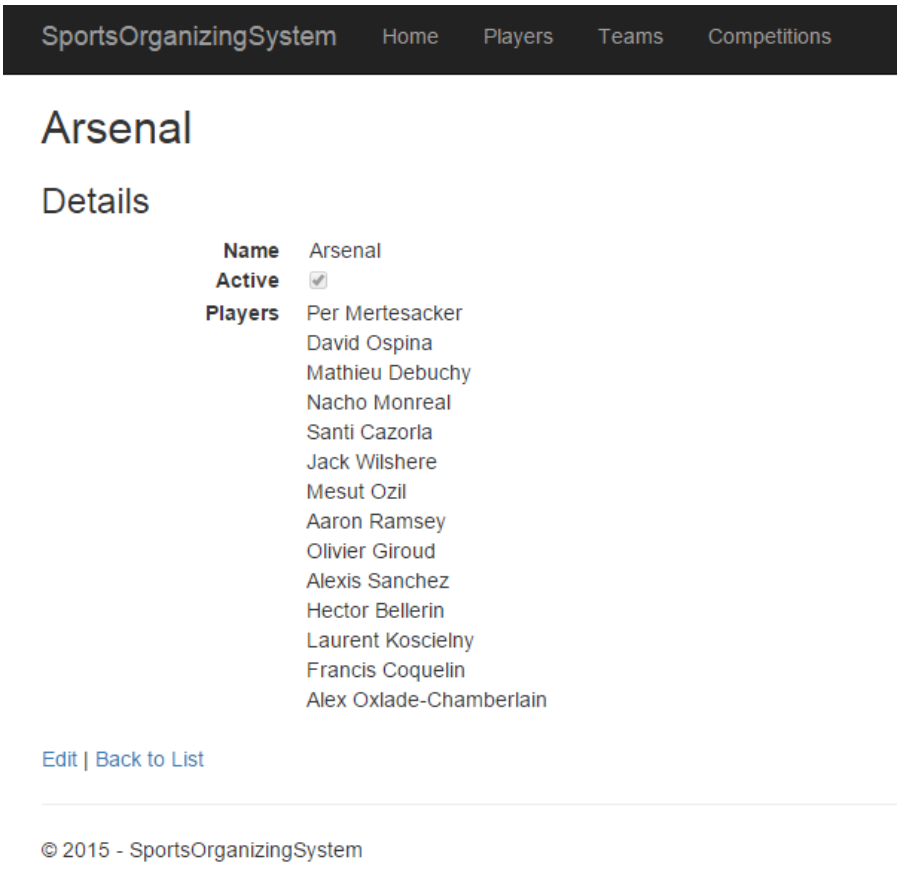
Přidá hráče k týmu.
Parametry:

- `playerId` - ID existující entity `Player`.

POST /Teams/RemovePlayer/:id

Odstraní hráče z týmu.
Parametry:

- `playerId` - ID existující entity `Player`.



Obrázek 10: Informace o týmu

5.3.3 Správa zápasů

Zápas je v systému reprezentován třídou `SOSMatch`. Jednotlivé události, které v zápase mohou nastat dědí od abstraktní třídy `SOSMatchEvent`. V projektu byla implementována pouze událost `Goal`. Logika operací s těmito entitami se nachází ve třídě `MatchesController`.

Ke správě zápasů byly napsány dva doubory *view*:

- `Views/`
  - `Matches/`
    - `Edit.cshtml`
- `Competitions/`
  - `MatchesList.cshtml`

GET /Matches/GetMatchInfo/:id

Vrací údaje o konkrétním zápase. Obsahuje indikátor, zda byl zápas odehrán, datum ve formátu *DateTime*, informace o týmech a hráčích, kteří se utkání zúčastní a veškeré góly, které v utkání padly. Zkrácená ukázka JSON objektu, která tato metoda vrací je dostupná v příloze C.3.

POST /Matches/AddPlayer/:id

Přidá nominovaného hráče do utkání.

Parametry:

- `playerId` - ID existující entity `Player`.

POST /Matches/RemovePlayer/:id

Odstraní nominovaného hráče z utkání.

Parametry:

- `playerId` - ID existující entity `Player`.

POST /Matches/AddGoal/:id

Přidá gól do utkání. Parametry `homeTeamScore` a `awayTeamScore` obsahují nové (již změnené) skóre. Parametry `scoredById` a `assistedById` nemohou být stejné a zároveň se musí jednat o ID hráčů ze stejného týmu (pokud je `assistedById` nastaveno).

Parametry:

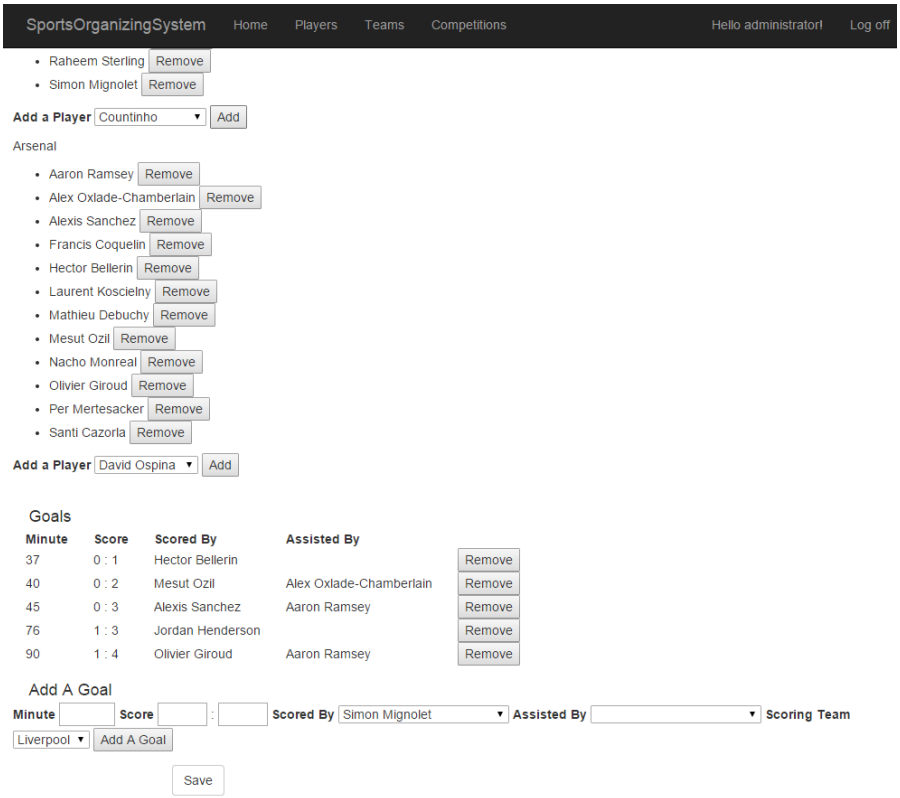
- `minute` - minuta, ve které byl gól vstřelen.
- `homeTeamScore` - skóre pro domací tým.
- `awayTeamScore` - skóre pro hostující tým.
- `scoredById` - ID hráče, který gól vstřelil.
- `assistedById` - ID hráče, který si zapsal asistenci. Nepovinný parametr.
- `goalForTeamId` - ID týmu, který vstřelil gól.

POST /Matches/RemoveGoal/:id

Odstraní gól z utkání.

Parametry:

- `goalId` - ID entity `Goal`.



Obrázek 11: Stránka pro úpravy informací o utkání

5.3.4 Správa soutěží

Soutěže jsou v systému reprezentovány třídou `SOSCompetition`. Jednotlivá kola, která může soutěž obsahovat značí třída `SOSRound`. Logiku soutěží má na starosti třída `CompetitionsController`.

Správa soutěží zahrnuje tyto *view*:

- `Views/`
  - `Competitions/`
    - `Create.cshtml`
    - `Delete.cshtml`
    - `Details.cshtml`
    - `Edit.cshtml`
    - `Index.cshtml`
    - `MatchesList.cshtml`
    - `Statistics.cshtml`

POST /Competitions/AddCompetitor/:id

Přihlásí tým do soutěže.
Parametry:

- `teamId` - ID existující entity `Team`.

POST /Competitions/RemoveCompetitor/:id

Odhlásí tým ze soutěže.
Parametry:

- `competitorId` - ID existující entity `CompetingTeam`.

GET /Competitions/GetCompetitors/:id

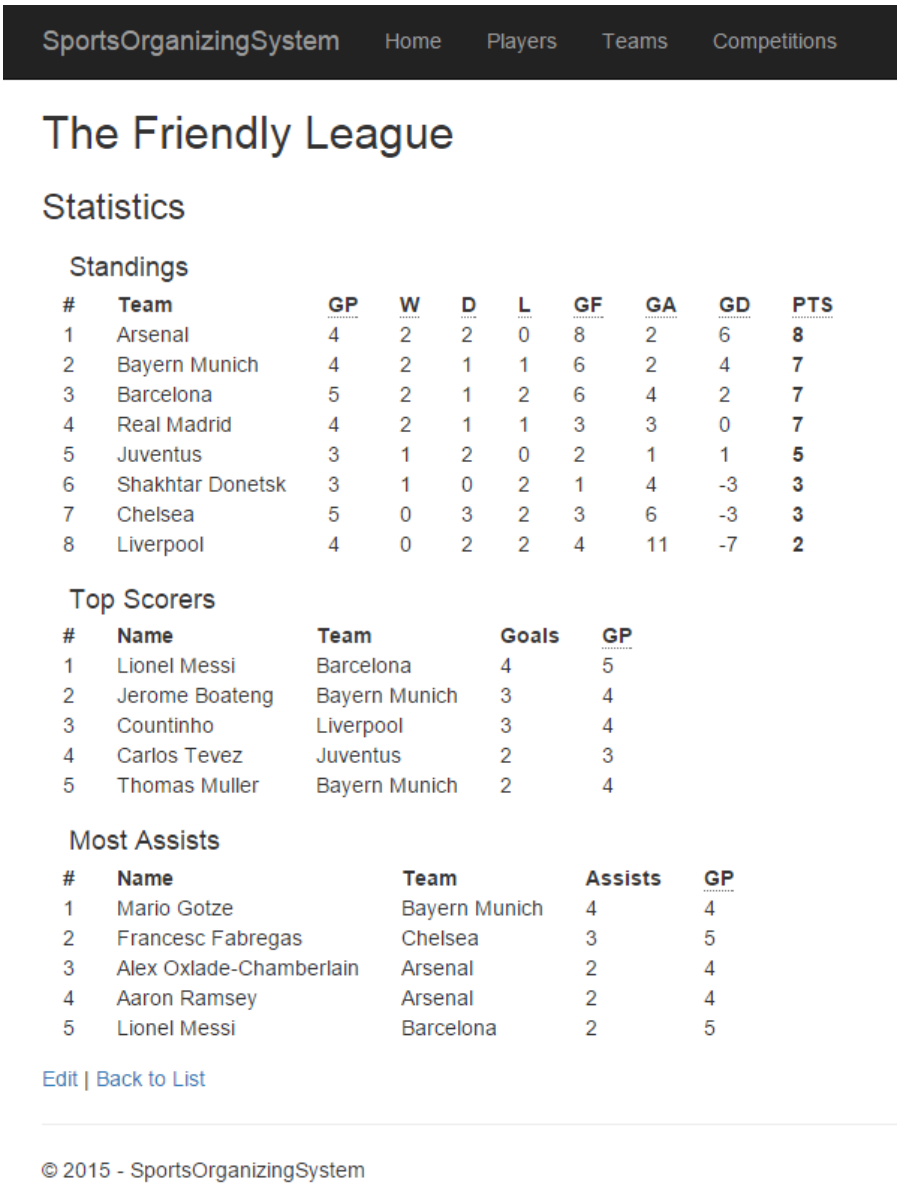
Vrátí JSON objekt se seznamem týmů, kteří jsou přihlášeni do soutěže.

GET /Competitions/GetPotentionalCompetitors/:id

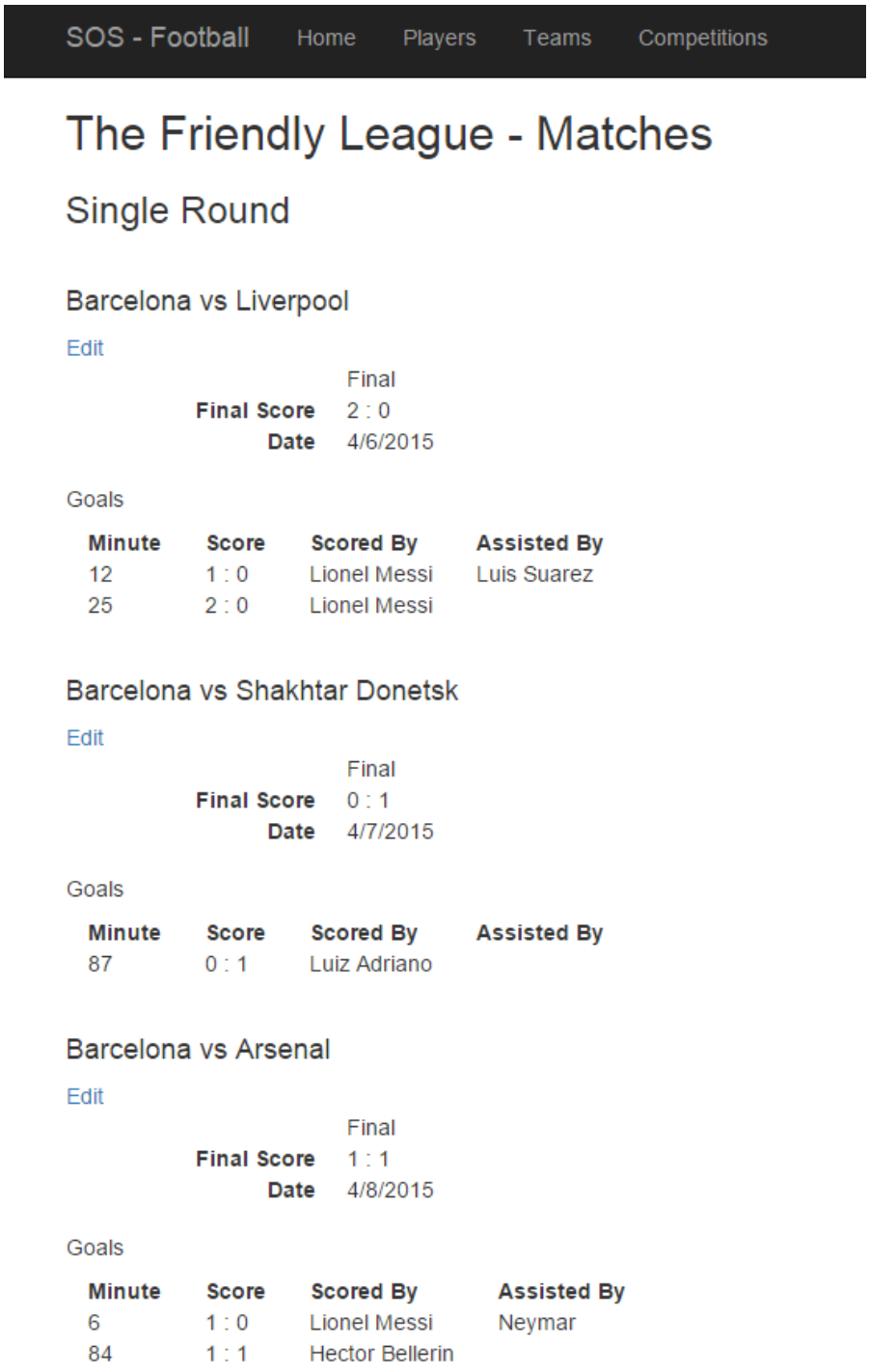
Vrátí JSON objekt se seznamem týmů, kteří se mohou přihlásit do soutěže. Přihlásit se do soutěže mohou pouze aktivní týmy (parametr `Active` v entitě *Team*).

GET /Competitions/DrawRounds/:id

Zahájí losování soutěže podle jejího typu. V projektu je prozatím podporováno losování pouze pro typ soutěže `SingleRoundRobin`, kde každý tým hraje s každým právě jednou.



Obrázek 12: Vyhodnocené statistiky v soutěži



Obrázek 13: Seznam zápasů v soutěži

5.3.5 Tisk zpráv a diplomů

Pro tisk diplomů a zpráv byla vytvořena třída `PrintoutsController`. Celkem je generována jedna zpráva a pět diplomů pro každou soutěž - tři diplomy pro tři první místa, diplom pro hráče, který nastřílel nejvíce branek a pro hráče, který zaznamenal nejvíce asistencí.

K diplomům byl vytvořen relativně jednoduchý CSS styl pro tisk. Ukázky diplomu jsou dostupné v příloze D.1.

Po vytištění diplomu se bohužel ukázalo, že technologie HTML a CSS je pro jejich vytváření zcela nevhodná. Prohlížeče Chrome i Firefox totiž mají problémy s tiskem CSS vlastností `text-shadow` a `box-shadow` (nevytisknou vůbec, nebo zcela nahradí černou čárou). Vytvořený diplom pak vypadá poněkud jednoduše.

Tento problém není vážný pro vygenerované zprávy, které mají hlavně informační účel. Ukázka zprávy je přiložená v příloze D.2.

5.3.6 Lokalizace

Systém byl lokalizován do českého a anglického jazyka pomocí XML souboru ve formátu *Microsoft ResX Schema* ve kterém jsou definovány řetězce, které aplikace používá. Každý jazyk má vlastní ResX soubor, v projektu se jedná o soubory `Resources.resx` a `Resources.cs.resx` v adresáři `Properties`. Entity Framework přepíná používaný soubor podle nastavení jazyka v prohlížeči. K lokalizaci byl použit nástroj Multilingual App Toolkit pro Visual Studio. [20] Ten sice umí generovat strojové překlady z českého jazyka do anglického, tyto překlady ale nejsou moc přesné a valnou většinu z nich bylo nutno přepsat.

6 TESTOVÁNÍ

Testování systému na za úkol najít nedostatky a odhalit chyby vzniklé během vývoje. Aplikace byla celkem otestována třemi způsoby - manuálním testováním, integračními automatizovanými testy, které probíhají v prohlížeči a uživatelským testováním. Každý z těchto testů ma jiný účel.

6.1 MANUÁLNÍ TESTOVÁNÍ

Po implementaci byla aplikace mnou manuálně otestována. Byla napsána testovací data, která byla vložena do databáze. Následně jsem systému využíval jako běžný užlvatel - manipuloval s hráči, týmy a soutěžemi, zadával a měnil výsledky zápasů. S pomocí nástroje SQL Server Object Explorer, jsem sledoval zda provedené změny byly správně promítnuty do databáze.

Tímto způsobem celkem dvě chyby. První chyba, méně závažnější chyba, byla v metodě `GetMatchInfo(id)` ve třídě `MatchesController`, vracela špatný JSON formát v případě, kdy v zápase padl gól bez zapsané asistence.

Druhá chyba byla mnohem závažnější - v metodě `EvaluateMatch(id)` ve třídě `MatchesController` - aplikace špatně vyhodnocovala výsledky zápasů, které by se promítly do databáze. Důvodem bylo simultánní zápis i čtení do databáze.

Toto testování bylo přínosem, neboť odhalily chyby, které byly při prvotní implementaci nepovšimnuty, přičemž jedna z nich by výrazně ovlivnila funkčnost systému. Obě nalezené chyby byly odstraněny.

6.2 SELENIUM IDE

Selenium IDE je nástroj pro automatizované testování v prohlížeči Firefox [21]. Testy v Selenium kontrolují, zda aplikace na klientské straně (de facto v prohlížeči) funguje korektně. Testovací kód je možné psát ve vývojovém prostředí Visual Studio díky dostupnému pluginu.

Automatické testy v prohlížeči mají tu nevýhody, že jsou silně závislé na uživatelském rozhraní. Při jeho změně je nutné veškeré testy přepsat.

Díky napsaným testům nebylo nutné po změně programu (kromě změn týkajících se komponenty *view*) kontrolovat funkčnost manuálně. Testy objevily chyby, které nastaly po změně implementace, jednalo se o neresponzivní tlačítka pro rozlosování soutěže a ukládání informací o zápase. Chyby byly po nalezení ihned odstraněny.

6.3 UŽIVATELSKÉ TESTOVÁNÍ

Testování se zúčastnili celkem tři participanti. Bylo důležité aby aplikaci předem neviděli, měli zkušenosti se sportovními soutěžemi a povědomí o fotbale.

Participantům byla předána aplikace, ve které se již nacházely testovací data a byly jim zadány následující úkoly:

1. Přihlašte se do systému s vlastními uživatelskými údaji.
2. Přidejte do systému dva nové hráče.
3. Vytvořené hráče přidejte do týmu s názvem "FC Test".
4. Vytvořte soutěž s názvem "Test League"a nastavte typ soutěže na "Single Round Robin"(v české verzi "Každý s každým na 1 zápas".
5. Do soutěže přidejte týmy "FC Test", "Arsenal"a "Barcelona". Soutěž rozlosujte.
6. Informace o rozlosovaných zápasech vyplňte podle vlastního uvážení.

Tato forma testování se ukázala jako nejpřínosnější ze všech. Bylo odhaleno několik chyb, vzniklých neočekávaným chováním uživatele. Jeden z uživatelů se například pokusil změnit seznam soutěžících potom, co soutěž rozlosoval - mě osobně by tato možnost vůbec nenapadla.

Další tester narazil na situaci, kdy se mu tlačítko "Přidat gól"jevilo neresponzivní (prohlížeč zpracovával odpověď GET requestu od serveru), tak na něj kliknul víckrát, výsledkem bylo, že metoda GET poslala několikrát a server do databáze uložil více záznamů o brance. Toto bylo vyřešeno tím, že tlačítko bylo učiněno neklikatelným dokud prohlížeči nezpracuje odpověď od serveru.

Objevilo se celkem velké množství připomínek ohledně uživatelského rozhraní aplikace, např. v první verzi aplikace bylo nutné tým, nebo soutěž do systému přidat a až poté se dalo manipulovat se seznamem hráčů, případně soutěžících, což bylo pro uživatele silně matoucí. Řešením je přesměrování v komponentě *controller*, tedy poté co uživatel do systému přidá uživatele je přesměrován na formulář může provádět další změny.

7 ZÁVĚR

Informační systém byl po obeznámení se s problematikou a analýzou v kapitole 4 úspěšně naimplementován a otestován. Byly prozkoumány možnosti nabízené frameworkem .NET od lokalizace, po psaní webových API a stránek v jazyce Razor. Vyzkoušel jsem si práci s knihovnou AngularJS, Microsoft SQL Serverem a samostatnou realizaci a nasazení většího webového projektu. Rád bych vyzdvihl přínos IDE Visual Studio, které práci značně ulehčilo.

Celkově bylo splněno jedenáct požadavků ze třinácti - v systému je podporováno rozlosování pouze pro jediný typ soutěže a umí vyhodnocovat jen vstřelené góly, nikoliv ostatní události (entity `Card`, `Foul` a `Substitution` popsané v doménovém modelu). Dále volba implementace tisku diplomů pomocí HTML a CSS se projevila jako nevhodné řešení.

7.1 BUDOUCÍ VÝVOJ SYSTÉMU

V první řaděbude nutné doimplementovat ostatní možnosti rozlosování jiných typů soutěží a vyhodnocování událostí v systému. Také bude potřeba zcela předělat systém implementace tisku diplomů.

Druhé zlepšení by se týkalo implementační stránky. V komponentách *controller* je prováděno mnoho operací nad databází, přičemž spousta z nich je se opakuje - vzniká tak duplicitní kód v rozdílných třídách. Osobně bych toto řešil návrhovým vzorem *Repository* a *Unit of Work*. [22]

Další částí, který by si zasloužila pozornost je webové rozhraní systému. V současném stavu je sice dostačující, ale bylo by ho možné výrazně vylepšit a zpřehlednit (zejména část s výpisem informací o utkání). Vyhledání konkrétního zápasu je momentálně relativně zdlouhavé.

Do budoucna by bylo možné přidat účty do systému samotným soutěžícím. Ti by si spravovali vlastní seznamy hráčů v týmu a zadávali by výsledky. Výsledky zápasů by bylo nutné nějakým způsobem validovat, validaci by prováděli organizátoři nebo výsledek by se mohl potvrdit potom, co by ho do systému zadaly oba soutěžící v zápase. Tím by se potenciálně snížila pracovní zátěž pořadatelů.

V rámci projektu bylo vypracované webové API, které by po rozšíření mohlo být využito např. mobilní aplikací. Organizátoři soutěží by pak mohli aktualizovat záznamy v systému přímo v terénu. Toto API není RESTful (RESTové), protože nebylo navrženo pro třetí strany, nedodržuje běžné konvence pro REST API a využívá pouze dotazovací metody GET a POST.

LITERATURA

[1] Roman Janko. *iSport - sportovní webový systém*. URL: http://isport.gameway.cz (ze dne 02. 02. 2014).

[2] SportPAD. *SportPAD - Official BUCS Intra Mural Platform*. URL: http://sportpad.net (ze dne 02. 12. 2014).

[3] InteractSports. *Online Sports Administration Software - InteractSport*. URL: http://www.interactsport.com/solutions.aspx?rw=c (ze dne 12. 12. 2014).

[4] Active Network LLC. *Sports League Management & Registration Software | ActiveSports*. URL: http://www.activesports.com (ze dne 08. 12. 2014).

[5] AllIsNetwork LLC. *TioPro*. URL: http://tiopro.com (ze dne 12. 12. 2014).

[6] Jim Arlow a Ila Neustadt. *UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design*. 2nd Edition. Addison-Wesley Professional, USA, 2005. ISBN: 978-0321321275.

[7] Martin Fowler a Kendall Scott. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. 2nd Edition. Academic Press, UK, 1999. ISBN: 078-5342657838.

[8] Christian Nagel, Jay Glynn a Morgan Skinner. *Professional C# 5.0 and .NET 4.5.1*. 1st Edition. Wrox, UK, 2014. ISBN: 978-1118833032.

[9] Scott Guthrie. *ScottGu's Blog - Introducing "Razor" – a new view engine for ASP.NET*. URL: http://weblogs.asp.net/scottgu/introducing-razor (ze dne 26. 03. 2015).

[10] Brat Tech LLC, Google and community. *AngularJS - Superheroic JavaScript MVW Framework*. URL: https://docs.angularjs.org/ (ze dne 14. 03. 2015).

[11] Paul Krill, InfoWorld Inc. *What's so special about Google's AngularJS*. URL: http://www.infoworld.com/article/2612801/javascript/what-s-so-special-about-google-s-angularjs.html (ze dne 17. 03. 2015).

[12] Angular Community. *angular/angular.js - HTML enhanced for web apps*. URL: https://github.com/angular/angular.js (ze dne 17. 03. 2015).

[13] VisionMobile Ltd. *Comparison of 4 popular JavaScript MV* frameworks (part 2)*. URL: http://www.developereconomics.com/comparison-4-popular-javascript-mv-frameworks-part-2/ (ze dne 17. 03. 2015).

[14] The Core Less Team. *Less.js*. URL: http://lesscss.org/ (ze dne 18. 03. 2015).

[15] Less Community. *less/less.js - The dynamic stylesheet language*. URL: https://github.com/less/less.js (ze dne 19. 03. 2015).

[16] Microsoft. *Team Foundation Server*. URL: https://msdn.microsoft.com/en-us/vstudio/ff637362.aspx (ze dne 14. 04. 2015).

[17] Trygve Reenskaug. *Trygve/MVC*. URL: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html (ze dne 26. 03. 2015).

[18] Matthew DeMeritt. *Pleasing Bosses and Customers - A compelling case for ASP.NET MVC*. URL: http://www.esri.com/news/arcuser/0609/aspnetmvc.html (ze dne 27. 03. 2015).

[19] Martin Fowler. *GUI Architectures - Model View Controller*. URL: http://martinfowler.com/eaaDev/uiArchs.html#ModelViewController (ze dne 26. 03. 2015).

[20] Microsoft. *Multilingual App Toolkit - Windows app development*. URL: https://dev.windows.com/en-us/develop/multilingual-app-toolkit (ze dne 12. 05. 2015).

[21] Selenium. *Selenium-IDE — Selenium Documentation*. URL: http://www.seleniumhq.org/docs/02_selenium_ide.jsp (ze dne 14. 05. 2015).

[22] Microsoft. *Implementing the Repository and Unit of Work Patterns in an ASP.NET MVC Application*. URL: http://www.asp.net/mvc/overview/older-versions/getting-started-with-ef-5-using-mvc-4/implementing-the-repository-and-unit-of-work-patterns-in-an-asp-net-mvc-application (ze dne 18. 05. 2015).

[23] Microsoft. *Visual C# resources*. URL: https://msdn.microsoft.com/en-us/vstudio/hh341490.aspx (ze dne 12. 04. 2015).

[24] Dino Esposito. *Design of a Domain Model | MSDN Magazine*. URL: https://msdn.microsoft.com/en-us/magazine/hh547108.aspx (ze dne 04. 04. 2015).

[25] Moq Team. *Moq/moq - The most popular and friendly mocking framework for .NET*. URL: https://github.com/Moq/moq4 (ze dne 11. 05. 2015).

[26] Microsoft. *Use Code First Migrations to Seed the Database*. URL: http://www.asp.net/web-api/overview/data/using-web-api-with-entity-framework/part-3 (ze dne 12. 05. 2015).

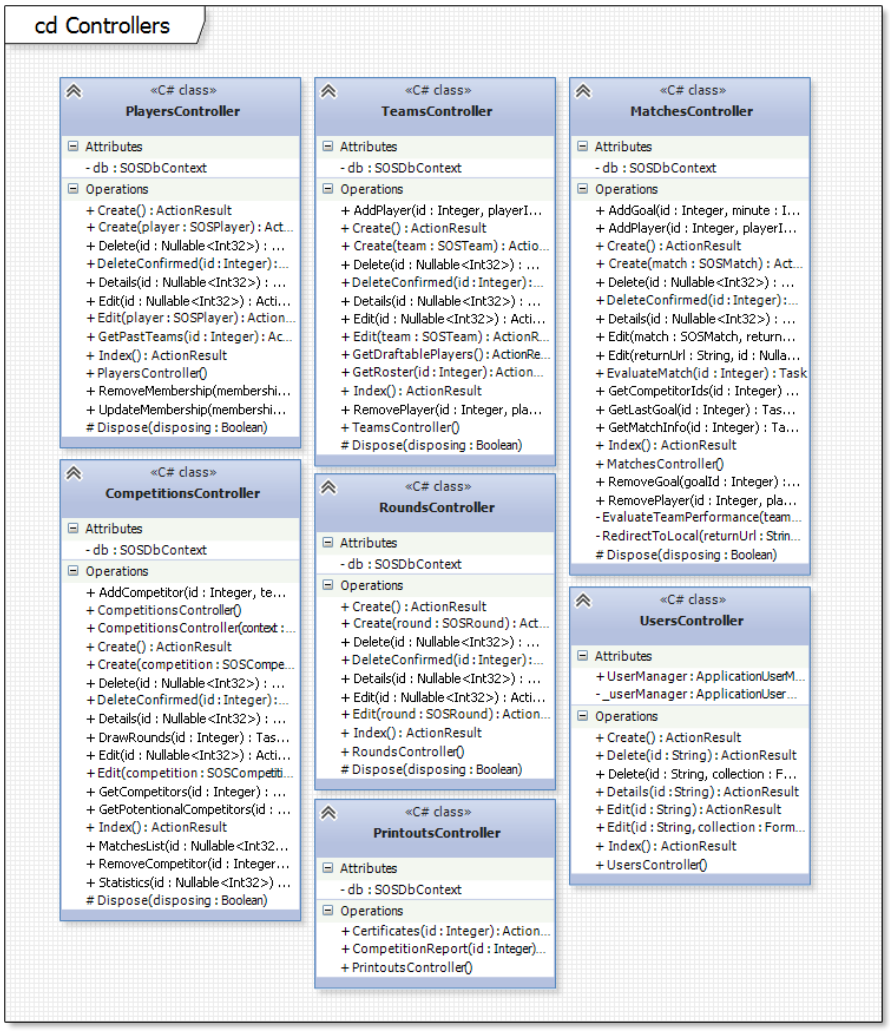
A SEZNAM POUŽITÝCH POJMŮ A ZKRATEK

.NET	Framework vyvíjený firmou Microsoft
CMS	Content Management System - systém pro správu obsahu
CRUD	Create, Read, Update, Delete - akronym pro čtyři základní operace s persistentními daty
CSS	Cascade Styling Sheets - Kaskádové styly
DOM	Document Object Model
GET	Dotazovací metoda protokolu HTTP
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment - Vývojové prostředí
JSON	JavaScript Object Notation
LESS	Nadstavba jazyka CSS
MIT	Massachusetts Institute of Technology
MVC	Softwarová architektura Model-View-Controller
PCI	Payment Card Industry Data Security Standard - soubor mezinárodních bezpečnostních standardů týkající se platebních karet
POST	Metoda protokolu HTTP
REST	Representational State Transfer
SSL	Secure Socket Layer - kryptografický protokol
XML	Extensible Markup Language

B OSTATNÍ MODELY

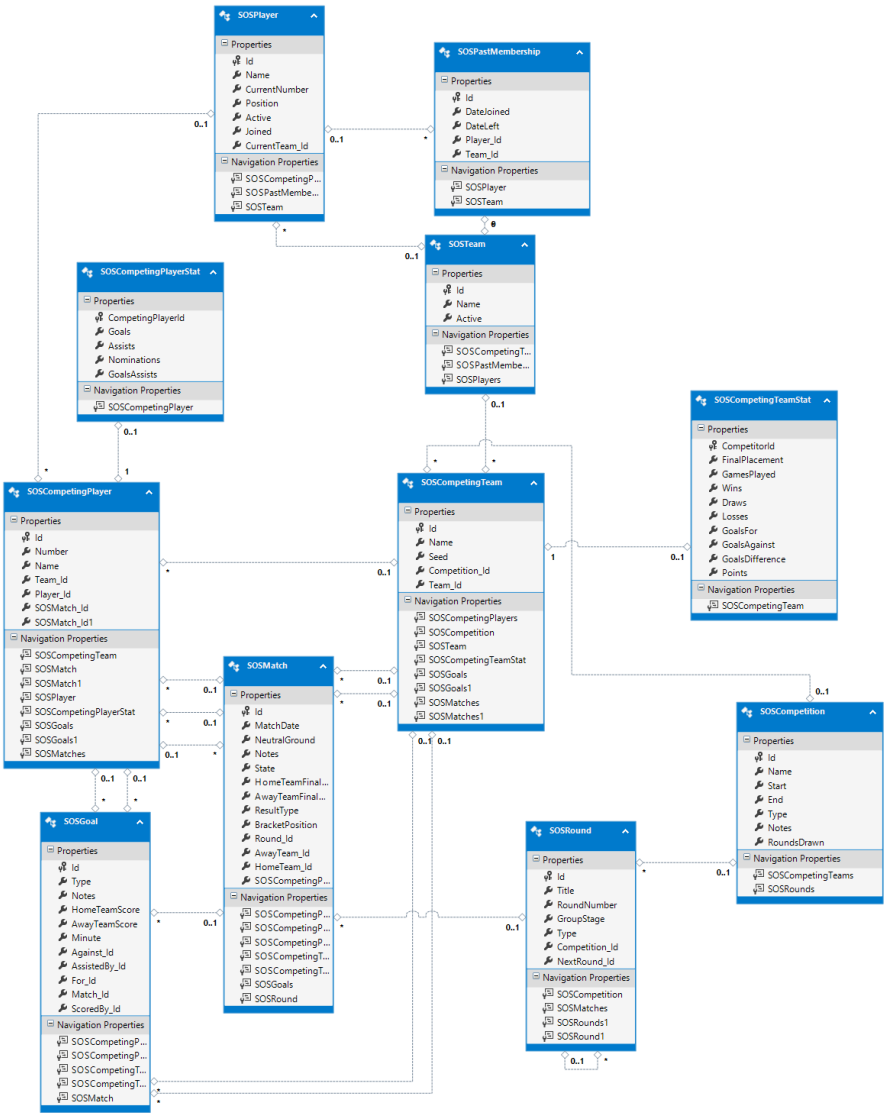
B.1 SEZNAM CONTROLLERŮ

V tomto diagramu není obsažena třída `AccountController.cs`, která používá z větší části kód vygenerovaný Entity Frameworkem.



Obrázek 14: seznam controllerů

B.2 DATABÁZOVÝ MODEL



Obrázek 15: Databázový model

C UKÁZKY KÓDU

Ukázky kódu jsou upravené (nemusejí odpovídat stoprocentně konkrétní implementaci v projektu), kvůli délce a přehlednosti, slouží totiž pouze k prezentačním účelům.

C.1 MODEL

```
public class SOSPlayer
    {
        public int Id { get; set; }

        [Display(
            Name = "Name",
            ResourceType =
                typeof(Resources.Resources))]
        [Required(
            ErrorMessageResourceName = "NameReqError",
            ErrorMessageResourceType =
                typeof(Resources.Resources))]
        public String Name { get; set; }

        [EnumDataType(
            typeof(FootballPlayerPosition))]
        public FootballPlayerPosition Position { get;
            set; }

        public bool Active { get; set; }

        [DataType(DataType.Date)]
        public DateTime Joined { get; set; }

        public virtual SOSTeam CurrentTeam { get; set;
            }

        public virtual ICollection<SOSPastMembership>
            Memberships { get; set; }

        public virtual ICollection<SOSCompetingPlayer>
            Participations { get; set; }

        public SOSPlayer()
        {
            this.Joined = DateTime.Now;
            this.CurrentNumber = null;
            this.Active = true;
```

```
            this.Position =
                FootballPlayerPosition.NotSet;
        }

        public SOSPlayer(String name)
            : this()
        {
            this.Name = name;
        }
    }
```

Ukázka 1: Implementace komponenty *model*

C.2 CONTROLLER

```
public class TeamsController : Controller
    {
        ...
        [HttpPost]
        public async Task<ActionResult> AddPlayer(int
            id, int playerId)
        {
            SOSTeam t = await db.Teams.FindAsync(id);
            SOSPlayer p = await
                db.Players.FindAsync(playerId);

            if (p == null || t == null)
            {
                throw new HttpException(400, "Bad
                    Request; Error detected while
                    trying to add a player to a team");
            }
            else
            {
                p.CurrentTeam = t;
                p.Joined = DateTime.Now;
                await db.SaveChangesAsync();
                return Json(true);
            }
        }

        ...

        [HttpGet]
        public ActionResult GetRoster(int id)
        {
            return Json(db.Players.
            Where(p => p.CurrentTeam.Id == id).
            Select(p => new {
                name = p.Name,
                id = p.Id,
                teamNumber = p.CurrentNumber}).
```

```
            OrderBy(p => p.teamNumber),
                JsonRequestBehavior.AllowGet);
        }

        ...
    }
```

Ukázka 2: Implementace vybraných metod v komponentě *controller* na straně serveru

C.3 JSON

```
{
   "matchState":2,
   "neutralGround":true,
   "homeTeam":{
      "id":6,
      "name":"Liverpool"
   },
   "awayTeam":{
      "id":1,
      "name":"Arsenal"
   },
   "homeTeamNominated":[
      {
         "id":34,
         "name":"Countinho",
         "number":10
      },
      {
         "id":37,
         "name":"Emre Can",
         "number":23
      }
      ...
   ],
   "awayTeamNominated":[
      {
         "id":8,
         "name":"Aaron Ramsey",
         "number":16
      },
      ...
   ],
   "homeTeamRoster":[
      {
         "id":30,
         "name":"Kolo Toure",
         "number":4
      },
      ...
   ],
```

```
    "awayTeamRoster":[
        {
            "id":6,
            "name":"Jack Wilshere",
            "number":10
        },
        ...
    ],
    "goals":[
        {
            "id":19,
            "minute":37,
            "homeTeamScore":0,
            "awayTeamScore":1,
            "scoredById":11,
            "scoredByName":"Hector Bellerin",
            "assistedById":null,
            "assistedByName":null,
            "goalForId":1,
            "goalAgainstId":6,
            "order":1
        },
        ...
    ]
}
```

Ukázka 3: JSON kód reprezentující informace o konkrétním utkání

D TISKOVÉ VÝSTUPY

D.1 DIPLOM


Gratulujeme týmu

SK ZÁBĚHLICE

za

2. místo

v soutěži

Okresní přebor

Počet získanách bodů:	39
Počet odehraných her:	10
Počet vstřelených branek:	3
Počet obdržených branek:	2

Datum: .....

Předal: .....

Obrázek 16: Ukázka diplomu

D.2 ZPRÁVA O SOUTĚŽI

Emirates Cup - Report

Total Matches 6
Matches Played 3
Total Goals Scored 7

Start 4/6/2015
End 4/13/2015

Standings

#	Team	GP	W	D	L	GF	GA	GD	PTS
1	Barcelona	2	1	0	1	3	2	1	**3**
2	Arsenal	1	1	0	0	1	0	1	**3**
3	Liverpool	1	0	1	0	1	1	0	**1**
4	Juventus	2	0	1	1	2	4	-2	**1**

Top Scorers

#	Name	Team	Goals	GP
1	Lionel Messi	Barcelona	2	2
2	Countinho	Liverpool	1	1
3	Hector Bellerin	Arsenal	1	1
4	Carlos Tevez	Juventus	1	2
5	Alvaro Morata	Juventus	1	2

Most Assists

#	Name	Team	Assists	GP
1	Luis Suarez	Barcelona	2	2
2	Jack Wilshere	Arsenal	1	1

Most Goal & Assists

#	Name	Team	G&A	GP
1	Luis Suarez	Barcelona	3	2
2	Lionel Messi	Barcelona	2	2
3	Hector Bellerin	Arsenal	1	1
4	Countinho	Liverpool	1	1
5	Carlos Tevez	Juventus	1	2

Most Games Played

#	Name	Team	GP
1	Lionel Messi	Barcelona	2
2	Luis Suarez	Barcelona	2
3	Neymar	Barcelona	2
4	Carlos Tevez	Juventus	2
5	Alvaro Morata	Juventus	2

Obrázek 17: Ukázka vygenerované zprávy

E INSTALAČNÍ PŘÍRUČKA

K nasazení systému je nutné mít k dispozici přiložené CD a aby na daném počítači běžel operační systém Windows.

E.1 LOKÁLNÍ NASAZENÍ SYSTÉMU

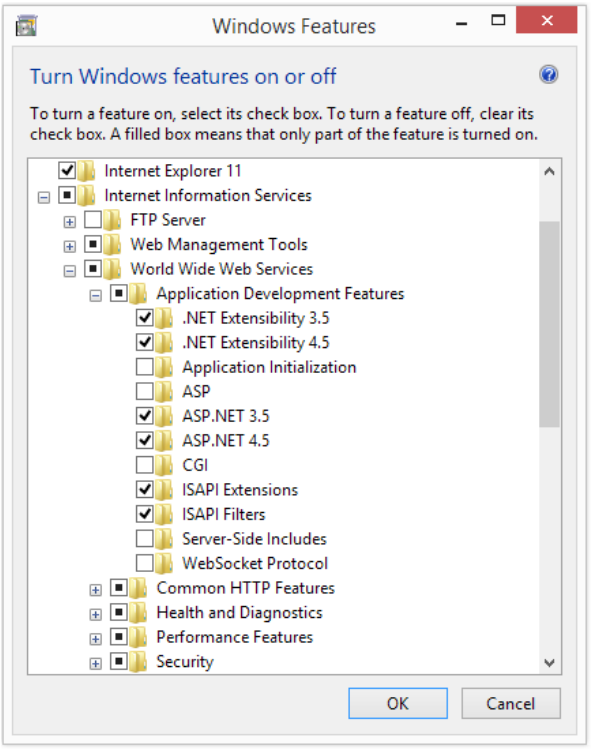
Lokální způsob nasazení je vhodný pro vyzkoušení aplikace (alternativním způsobem vyzkoušení je otevření projektu ve vývojovém prostředí Visual Studio).

1) Zkopírování adresáře DeploymentPackage

Zkopírujte si obsah adresáře `DeploymentPackage`, kde se nachází nasazovací balíček, z CD na disk. Nová lokace těchto souborů bude místo, kam bude aplikace nasazena.

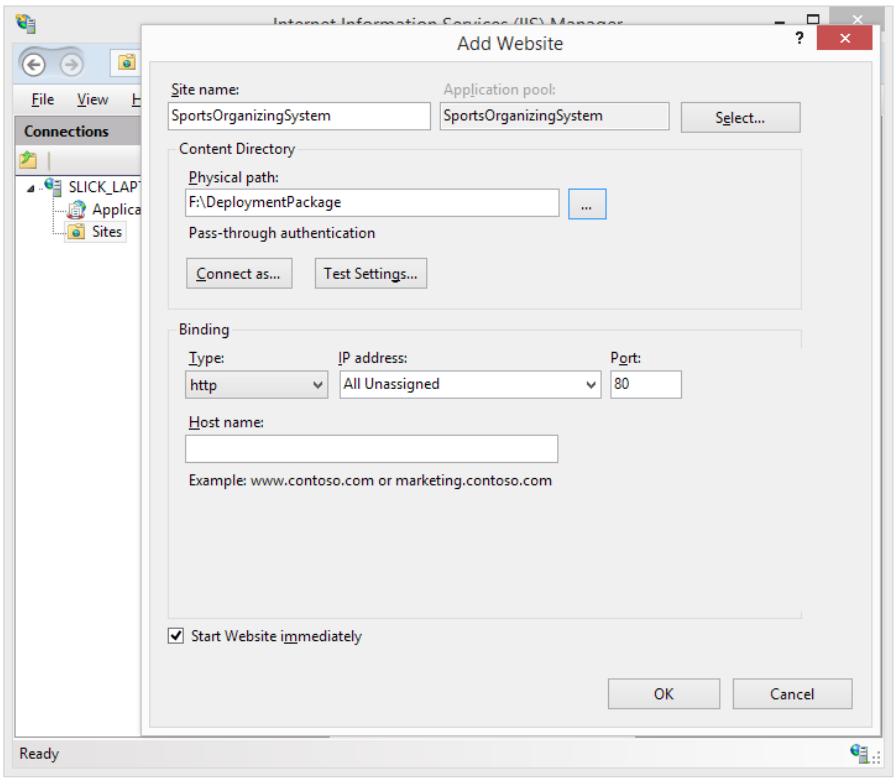
2) Zapnutí IIS

Nejprve je nutné zapnout službu *Information Internet Services*, která je součástí Windows. Je důležité, aby byla zaškrtnuta položka *ASP.NET* s nejnovější verzí!



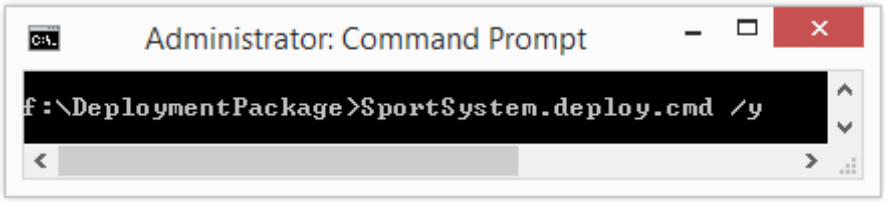
3) Konfigurace IIS

Zapněte program *Information Internet Service (IIS) Manager*. Klikněte pravým tlačítkem na položku *Sites* pod názvem počítače, a vyberte položku *Add Website*. Do položky *Site name* zadejte hodnotu *SportsOrganizingSystem*. V položce *Physical Path* nalezněte lokaci, kam jste zkopírovali obsah adresáře `DeploymentPackage` v prvním kroce. Zmáčkněte tlačítko *OK*.



4) Samotné nasazení systému

Spusťte příkazovou řádku s administrátorskými právy. Nalezněte lokaci adresáře se soubory pro nasazení (nesmí být na CD, musí se nacházet lokálně na disku!) a zadejte příkaz `SportSystem.deploy.cmd /y`.



5) Přihlášení do systému

Aplikace by měla běžet na adrese `http://localhost`. Uživatelské jméno pro přístup do systému je `administrator`, heslo je `password`.

E.2 REÁLNÉ NASAZENÍ SYSTÉMU

Pro reálné nasazení systému je doporučováno, aby na cílovém počítači byl dostupný a nakonfigurovaný Microsoft SQL Server. Pak by bylo nutné vytvořit vlastní balíček k nasazení ve výjovém prostředí Visual Studio.

Po otevření projektu klikněte pravým tlačítkem v liště *Solution Explorer* na projekt *SportSystem* a vyberte *Publish*. V liště *Connection* zadejte lokaci, kam má být balíček uložen a jako *Site name* zadejte *SportsOrganizingSystem*. V liště *Settings* je nalezněte svůj Microsoft SQL Server a zadejte příslušné přístupové údaje. Klikněte na tlačítko *Publish*.

Samotné nasazování probíha stejně jako v E.1, akorát se pracuje s novým nasazovacím balíčkem místo toho dostupného na CD.

F OBSAH PŘILOŽENÉHO CD

- `DeploymentPackage/` - Adresář s balíčkem pro nasazení informačního serveru.
- `SportSystem/` - Adresář s projektem.
  - `Packages/` - knihovny plug-iny, používané v projektu
  - `SportSystem/` - Zdrojové kódy projektu.
    - `App_Start/` - Konfigurační soubory ASP.NET
    - `Content/` - CSS soubory využívané aplikací
    - `Controllers/` - Implementace komponent *controller*
    - `Extensions/` - Pomocné třídy
    - `Migrations/` - Obsahuje konfigurační soubor pro databázi
    - `Models/` - Implementace komponent *model*
    - `Properties/` - Metadata projektu a soubory s lokalizací
    - `Scripts/` - JavaScriptové soubory využívané aplikací
      - `controllers.js` - AngularJS soubor využívaný komponentami *view*
    - `Views/` - Implementace komponent *view*
  - `SportSystem.tests/` - Zdrojové kódy testů projektu.
  - `SportSystem.sln` - Spouštěcí soubor projektu pro vývojové prostředí Visual Studio
- `NguyenTatPhong_2015_BP.pdf` - PDF soubor s bakalářskou prací.