Aplikace pro evidenci výsledků soutěže RoboGames

Bc. Petr Kříž

Diplomová práce
2021

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Petr Kříž**
Osobní číslo:	**A18265**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Forma studia:	**Kombinovaná**
Téma práce:	**Aplikace pro evidenci výsledků soutěže RoboGames**
Téma práce anglicky:	**An Application for the Evidencing of RoboGames Competition Results**

Zásady pro vypracování

1. Seznamte se s pravidly a organizací soutěže RoboGames na UTB.
2. Vhodným způsobem definujte požadavky na aplikaci pro evidenci výsledků soutěže.
3. Vyberte vhodné technologie pro implementaci aplikace.
4. Navrhněte architekturu aplikace.
5. Implementujte serverovou část a klientskou aplikaci.
6. Ověřte funkčnost celého systému.

Forma zpracování diplomové práce: **Tištěná/elektronická**

Seznam doporučené literatury:

1. Herbert Schildt, 2016, Java 8 Výukový kurz, Computer Press, EAN:9788025146651.
2. Elizabeth Castro, Bruce Hyslop, 2012, HTML5 a CSS3 Názorný průvodce tvorbou WWW stránek, Computer Press, ISBN:9788025137338.
3. Bruce Momjian, 2003, PostgreSQL Praktický průvodce, Computer Press, ISBN: 8072269542.
4. Robert C. Martin, 2009, Čistý kód, Computer Press, ISBN: 9788025122853.
5. Rudolf Pecinovský, 2007, Návrhové vzory, Computer Press, ISBN:9788025115824.

Vedoucí diplomové práce: **Ing. Peter Janků, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **15. ledna 2021**
Termín odevzdání diplomové práce: **17. května 2021**

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

prof. Mgr. Roman Jašek, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 15. ledna 2021

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomové práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky. Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

Petr Kříž v. r.

podpis studenta

ABSTRAKT

Diplomové práce se zabývá vývojem informačního systému napomáhajícího organizaci průběhu a vyhodnocení výsledků soutěže RoboGames. Soutěž se skládá z různých nezávislých disciplín, které plní týmy se svými mechanickými roboty. V práci je popsán dosavadní způsob organizace a vyhodnocování soutěže. Na základě získaných informací jsou definovány funkční a nefunkční požadavky, vytvořeny případy užití popsané scénáři a identifikovány základní třídy. Pro realizaci byl vybrán pracovní rámec *Spring* využívající platformu *Java*. Vytvořený informační systém pracuje jako webová aplikace postavená na třívrstvé architektuře využívající relační databázi.

Klíčová slova: systém pro správu soutěže, soutěž robotů, vývoj webového informačního systému, Java, Spring

ABSTRACT

This diploma thesis deals with development of information system which organize and evaluate RoboGames competition results. Competition is combination of independent disciplines, which each team performs with their mechanical robots. This document describes actual organization and game evaluation. Functional and nonfunctional requirements, use cases, scenarios and basic classes are based on gathered information. For implementing was chosen *Spring* framework using *Java* platform. Final information system is web application based on three layer architecture with data persistence using relational database system.

Keywords: competition management system, robot game competition, developing web based information system, Java, Spring

Nikdy nedovol, aby tě strach z prohry vyřadil ze hry

„Never let the fear of striking out keep you from playing the game“

George Herman „Babe“ Ruth
Baseball player

Poděkování:

Děkuji vedoucímu mé diplomové práce Ing. Peteru Janků Ph.D. za věcné rady, cenné připomínky a nesmírnou trpělivost.

OBSAH

ÚVOD

Fakulta aplikované informatiky Univerzity Tomáše Bati ve Zlíně pořádá soutěžní akci nejen pro studenty vysokých, středních a základních škol, ale také pro členy zájmových kroužků zejména z oblasti robotiky. Soutěž se zpravidla skládá z několika různých nezávislých disciplín ve kterých se utkávají týmy se svými vlastními mechanickými roboty. Každý tým tak získává možnost předvést své technické znalosti i dovednosti a poměřit je s ostatními soutěžícími.

Soutěžní den je zahájen registrací jednotlivých účastníků a kontroly jejich robotů, zda-li splňují požadovanou specifikaci. Po ukončení registrace jsou vytvořeny seznamy soutěžících podle jednotlivých disciplín a věkových kategorií. Průběh celé soutěže je organizován pomocí tištěných papírových archů, na které organizátoři ručně zapisují výsledky dosažené jednotlivými účastníky. V závěru soutěžního dne dojde k vyhodnocení a vyhlášení pořadí soutěžních týmů v jednotlivých disciplínách podle věkových kategorií a pro nejúspěšnější účastníky bývají připravené věcné ceny.

Tato událost, probíhající od roku 2017, byla pojmenována RoboGames.

Výše zmíněný organizační přístup je náročný nejen z hlediska přehlednosti o průběhu soutěže, ale také může přispívat ke vzniku chyb při zápisu výsledků nebo jejich následném vyhodnocení. Z tohoto důvodu vznikla tato práce, dávající si za cíl vývoj informačního systému usnadňující organizaci a automatizované vyhodnocení výsledků soutěže.

Práce je rozdělena do dvou částí. První teoretická část se zabývá rozborem problému, definicí požadavků na nově vznikající informační systém, vytvořením případů užití popsaných scénáři, indentifikací základních tříd, definicí validačních pravidel pro ověření očekávané funkcionality a popisem technologií vybraných pro implementaci. Druhá praktická část popisuje architekturu systému, bere si za úkol poskytnout základního průvodce strukturou celého informačního systému a uživatelským rozhraním.

I. TEORETICKÁ ČÁST

1 ROZBOR ZADÁNÍ

Následující kapitola vznikla na základě analýzy webové prezentace celostátní soutěže robotů na UTB ve Zlíně [1] a vlastním pozorováním průběhu soutěže. Obsahuje základní popis úkonů nutných pro zorganizování soutěže a zajištění jejího průběhu během soutěžního dne. Dále se zabývá rozborem v podobě specifikace požadavků a jejich rozepsání do případů užití. Následně jsou identifikovány základní třídy. Závěrem jsou vytvořena validační kritéria pro akceptaci funkčnosti informačního systému jako celku.

1.1 Základní popis

Soutěž RoboGames je celodenní událost, během které soutěží týmy v samostatných úkolech, nebo vzájemně proti sobě v závislosti na typu disciplíny. Organizátor určí tyto sportovní disciplíny během přípravy před vyhlášením soutěže, stejně tak jsou definovány věkové kategorie týmů, časový interval ve kterém se soutěžící týmy mohou registrovat a datum konání.

Každá disciplína má předem dané pravidla. Obecně se dá říci, že disciplína je definována svou hrací plochou, specifikací robota, hlavním úkolem, časovým omezením na plnění disciplíny a možným ohodnocením výsledku, které může být bodové - počet získaných bodů, nebo časové - délka trvání úspěšného dokončení úkolu.

Věkové kategorie bývají zpravidla tři, jedná se o žáky základních škol (do 15 let včetně), středoškolské studenty (ve věkovém rozmezí od 16 do 19 let) a dospělé soutěžící (starší 19 let).

Soutěž je vyhlášena v okamžiku zveřejnění na webovém portálu. Zveřejněné informace obsahují způsob registrace a časový interval ve kterém je umožněna registrace soutěžním týmům. Dále obsahuje organizační pokyny a doprovodné informace ohledně soutěžních disciplín, jejich pravidel, místa konání a harmonogramu soutěžního dne.

Organizace celé soutěže a průběh konání hry je nad rámec možností jednoho člověka, proto je zapotřebí rozdělit jednotlivé úlohy mezi všechny zúčastněné organizátory ještě před začátkem soutěžního dne.

Musí se zajistit uvítání soutěžních týmů, kontrolu a případnou aktualizaci týmových údajů. V neposlední řadě se ověřuje, zda-li roboti odpovídají požadované specifikaci týkající se rozměrů a váhy. Roboti musí být také zcela autonomní, je přísně zakázána jakákoliv vzdálená komunikace s robotem v průběhu plnění disciplíny.

Jakmile jsou všechny týmy zkontrolovány, dojde k vytvoření seznamů soutěžních týmů pro jednotlivé disciplíny v rámci soutěžních kategorií. Následně dojde k jejich vytištění a distribuci na příslušné stanoviště.

Dále je nutné určit osoby odpovědné za průběh konání soutěže na jednotlivých

stanovištích, zápis výsledků jednotlivých týmů a případně i jejich vyhodnocení. Nástup týmů na jednotlivé stanoviště je ohlašován ústní výzvou.

Po odehrání všech naplánovaných zápasů a úkolů je soutěž zakončena vyhodnocením všech zaznamenaných výsledků, jejich veřejným vyhlášením a oceněním nejúspěšnějších účastníků.

1.2 Pravidla disciplíny

Následují příklady pravidel dvou disciplín. [2] Podstatou první je plnění samostatného úkolu, týmy se účastní jednotlivě a vítězný tým je vybrán na základě nejlepšího dosaženého výsledku. Druhá disciplína je bojová, zápasy týmů jsou organizovány způsobem, že se každý tým utká se všemi ostatními týmy.

1.2.1 Sledování čáry

Na bílé hrací ploše je vyznačena trasa pomocí černé pásky o šířce patnácti milimetrů. Trasa může obsahovat kolmé nebo oblé zatáčky a křižovatky. Oblé zatáčky mají poloměr alespoň deset centimetrů a vzdálenost mezi dvěma nesouvislými body na trase je minimálně dvacet centimetrů (viz obrázek 1.1a).

Robot musí jet souvisle po trase, pokud se dostane mimo trasu a opět ji nalezne aniž by si přitom zkrátil cestu, může pokračovat v dokončení úkolu. V případě že trasu nalezne, ale cestu si zkrátí, bude pokus prohlášen za neplatný.

Maximální hmotnost robota je jeden kilogram a půdorysné rozměry se musí vejít do čtverce o rozměru strany dvacet pět centimetrů. Časový limit pro splnění úkolu je tři minuty. Dokud časový limit nevyprší, je možné postavit robota opět na start a zahájit nový pokus, jako výsledek se vybere nejkratší čas. V případě nedokončení trasy během limitu zapíše rozhodčí jako výsledek vzdálenost, kterou robot od startu trasy urazil. Tito roboti se umístí ve výsledkové listině až za ty, kteří dorazili do cíle v daném časovém limitu. Vítězí robot s nejkratším časem průjezdu, pokud žádný do cíle nedorazil, pořadí se určí podle ujeté vzdálenosti.

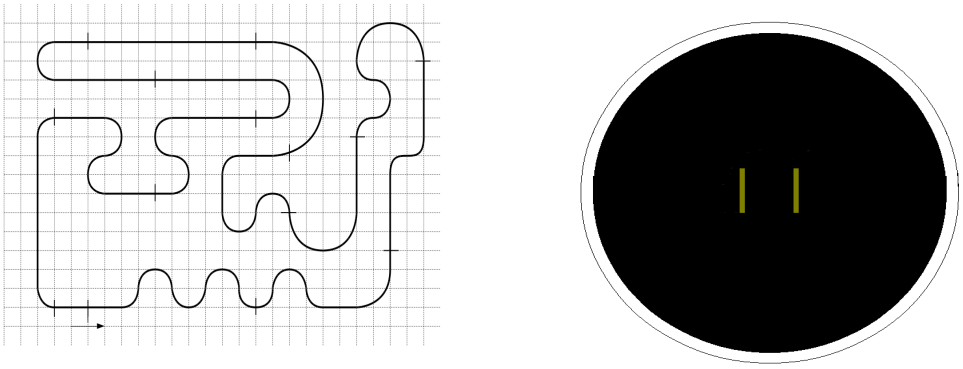
1.2.2 Robosumo

Tato disciplína má časový limit devíti minut pro jeden zápas, maximální hmotnost robota jeden kilogram a jeho půdorysné rozměry se musí vejít do čtverce o rozměru strany dvacet pět centimetrů. Kruhová hrací plocha o průměru sto padesát čtyři centimetrů má pěti centimetrový bílý okraj. Dvě startovací čáry o tloušťce a délce dva respektive dvacet centimetrů a jsou od sebe vzdáleny dvacet centimetrů (viz obrázek 1.1b).

Úkol disciplíny zní jasně: vytlačit svého protivníka mimo hrací plochu.

Zápas začíná umístěním robotů na startovací čáry a jejich aktivací na pokyn rozhodčího. Roboti se pět vteřin po aktivací nesmí pohnout, během této doby musí soutěžící odstoupit alespoň jeden a půl metru od hrací plochy. Jeden zápas má maximálně tři kola (délka jednoho kola je maximálně tři minuty) a vítěz každého kola získá jeden bod. V případě neaktivnosti robotu po dobu delší než pěti vteřin může rozhodčí nařídit nové kolo. Vyhrává tým, který získá dva body. Pokud je ve třetím kole po uplynutí časového limitu výsledek stále nerozhodný, může rozhodčí nařídit prodloužení nebo udělit vítězný bod za větší aktivitu během zápasu.

Roboti mohou po startu zápasu změnit své rozměry například roztažením ramen, nebo se může rozdělit na několik samostatných dílů, nicméně prohrává ve chvíli jakmile kterýkoliv jeho díl skončí mimo hrací plochu. Před prvním přejetím poloviny hrací plochy se musí otočit minimálně o devadesát stupňů. Během souboje je zakázáno upravovat mechanicky nebo chemicky přilnavost robota vůči hrací ploše například vypouštěním různých chemikálií. Je také zakázáno jakkoliv záměrně poškozovat protivníkův stroj.



(a) Sledování čáry (b) Robosumo

Obrázek 1.1 Hrací plochy [2]

1.3 Specifikace požadavků

Požadavky jsou rozděleny na funkční a nefunkční. Funkční požadavky formulují, co by systém měl dělat - popisují požadovanou funkcionalitu. Jedná se o funkce, které systém poskytuje svému okolí. Nefunkční požadavky specifikují vlastnosti nebo omezující podmínky kladené na systém. [3]

1.3.1 Funkční požadavky

- Systém bude autentizovat a autorizovat uživatele.
- Systém bude umožňovat vytvářet a upravovat soutěže.

- Systém bude umožňovat definování kategorií účastníků soutěže.
- Systém bude umožňovat upravovat disciplíny v rámci soutěže.
- Systém bude umožňovat spravovat hrací plochy (hřiště).
- Systém bude umožňovat týmům samoobslužnou registraci do soutěže.
- Systém bude kontrolovat unikátnost názvů jednotlivých objektů v rámci soutěže.
- Systém bude kontrolovat konzistenci referencí objektů v rámci definované soutěže.
- Systém bude umožňovat vytváření a editaci týmů.
- Systém bude umožňovat zadávání výsledků jednotlivých disciplín.
- Systém bude zobrazovat výsledky.
- Systém bude umožňovat správu uživatel systému a jejich oprávnění.

1.3.2 Nefunkční požadavky

- Systém bude autentizovat a autorizovat uživatele pomocí přihlašovacího formuláře.
- Systém bude umožňovat vytváření a editaci soutěží pouze uživateli s právy vedoucího.
- Systém bude ověřovat unikátnost názvu soutěže při jejich vytváření nebo editaci.
- Systém nebude autentizovat uživatele při samoobslužné registraci týmu.
- Systém bude umožňovat týmům samoobslužnou registraci na soutěž pouze v určitém časovém intervalu.
- Systém bude umožňovat vytváření a editaci týmů v rámci soutěže uživatelům s oprávněním asistent.
- Systém bude umožňovat zadávání výsledků jednotlivých disciplín uživatelům s oprávněním rozhodčí.
- Systém zobrazí varovné hlášení v případě vyvolání akce na kterou uživatel nemá dostatečné oprávnění.
- Uživatel s právy vedoucího má přístupné veškeré funkce systému.
- Systém umožní editovat uživatele systému pouze uživateli s právy vedoucího.
- Systém bude ukládat hesla uživatelů v šifrované formě.

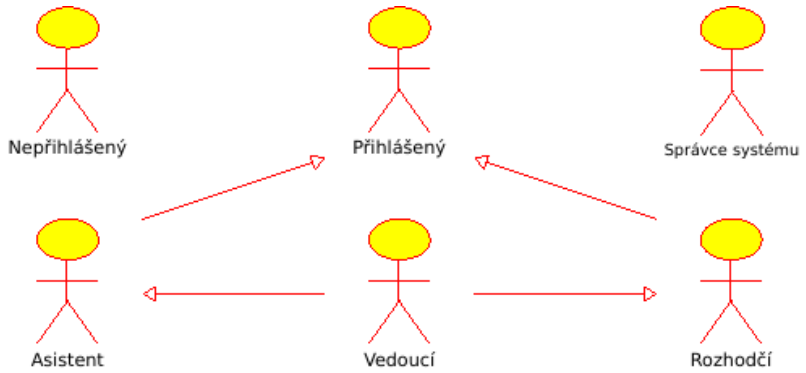
- Systém zobrazí oznámení k dostavení týmu na dané stanoviště do pěti vteřin od výzvy rozhodčího.
- Aplikace pro rozhodčí bude používána na mobilních zařízeních.
- Systém bude zobrazovat výsledky na velkoplošném zobrazovacím zařízení.

1.4 Případy užití

V následujícím oddíle identifikujeme aktéry informačního systému, které stručně popíšeme, graficky znázorníme případy užití pomocí diagramů a vytvoříme jednotlivé scénáře. [3]

1.4.1 Aktéři

Aktéři představují určité role ve kterých vystupují uživatelé informačního systém. Uživatelem systému může být buď fyzická osoba, nebo další systém využívající poskytované funkce. Z výše uvedeného popisu soutěže můžeme identifikovat hlavní tři role a to vedoucí, asistent a rozhodčí. Dále byly definovány role nepřihlášeného uživatele a správce systému. Aktéři a jejich vzájemné vztahy jsou znázorněny na obrázku 1.2.



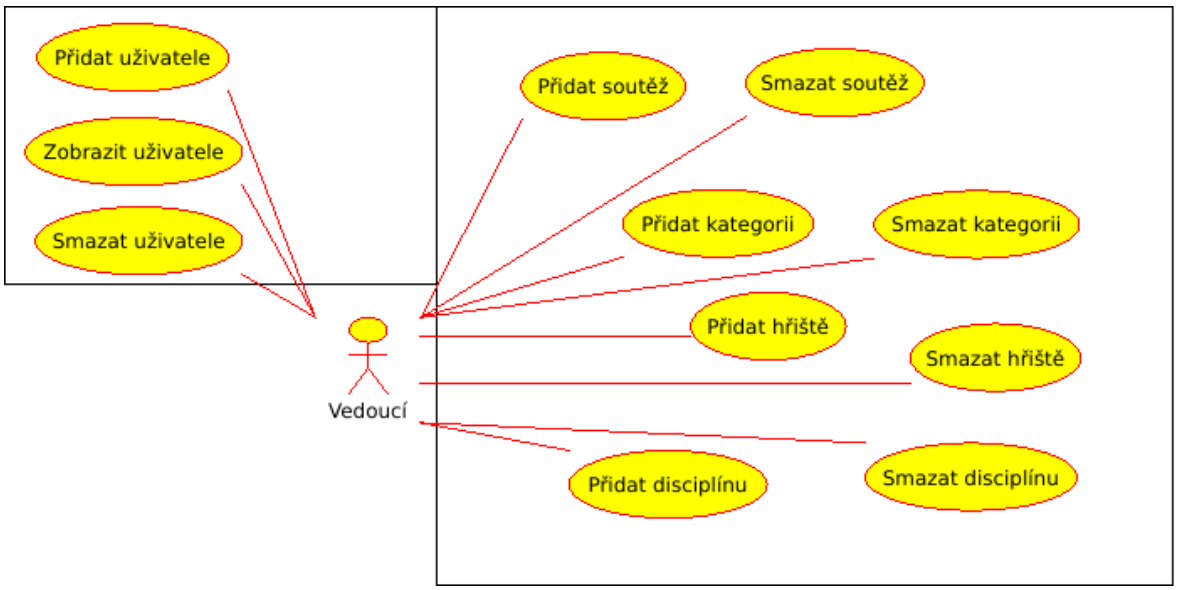
Obrázek 1.2 Aktéři

- *Vedoucí* je hlavní role, která může plně využívat veškeré dostupné funkce informačního systému. Mezi jeho pravomoci spadá správa uživatel systému, definování soutěží a jejich disciplín. V rámci této role jsou dostupné funkce systému definované pro aktéry Asistent a Rozhodčí.
- *Asistent* představuje roli pomocného organizátora, který má na starost administrativní úkony vedené vůči jednotlivým týmům a jejich členům. Jeho hlavní uplatnění je během soutěžního dne, kdy před začátkem soutěže kontroluje zadané informace o jednotlivých týmech a ověřuje, zda-li přinesení roboti odpovídají požadované specifikaci. Jakmile asistent odsouhlasí že tým splnil veškeré požadavky, označí tuto skutečnost v informačním systému a od této chvíle se daný tým může účastnit soutěže.

- *Rozhodčí* má přidělenou konkrétní hrací plochu a je zodpovědný za průběh jednotlivých kol dané disciplíny. Určuje týmy, které se zúčastní aktuálního kola a zapisuje dosažené výsledky.
- *Přihlášený* představuje uživatele, který není autorizován k provádění jakýchkoliv úprav v informačním systému. Takový uživatel může typicky pouze prohlížet aktuální informace o soutěžích.
- *Nepřihlášený* představuje roli uživatele, který neprošel autorizací ani autentizací. V tomto případě se může jednat o soutěžícího, který chce zaregistrovat svůj tým na vyhlášenou soutěž. Jakmile se uživatel úspěšně přihlásí do systému, získá autorizaci a systém ho bude považovat za rozdílného aktéra.
- *Správce systému* je role, která představuje osobu starající se o technické prostředky, které jsou využívány pro běh informačního systému. Tato role se nachází mimo rozsah samotného informačního systému. Odpovědností této role je schopnost spouštět, konfigurovat a zastavovat informační systém jako celek. I když tato role již nebude dále zmíněna v následujícím textu, je důležitá z pohledu provozu informačního systému.

1.4.2 Scénáře vedoucího

Obrázek 1.3 ilustruje případy užití vedoucí role. V tomto oddíle jsou zobrazeny pouze případy užití typické pro vedoucího, nicméně nesmíme zapomenout na skutečnost, že vedoucí je oprávněn provádět i veškeré úkony rolí asistent a rozhodčí, které budou popsány následně.



Obrázek 1.3 Diagram případů užití pro roli Vedoucí

Ve zbývající části tohoto oddílu jsou popsány jednotlivé scénáře případů užití.

Případ užití	Přidat uživatele
Stručný popis	Přidání uživatele do informačního systému.
Primární aktéři	Vedoucí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu přidat uživatele. 2. Systém zobrazí formulář pro nového uživatele. 3. Vedoucí zadá do formuláře přihlašovací jméno, heslo, celé jméno, email a popis. 4. Pokud je přihlašovací jméno uživatele prázdné nebo není unikátní: 4.1 Systém vypíše chybové hlášení, návrat k bodu 3. 5. Systém vytvoří nový účet uživatele s zašifrovaným heslem.	
Výstupní podmínky	Systém uložil nový uživatelský účet s zašifrovaným heslem.
Alternativní scénáře	Zrušit

Tabulka 1.1 Případ užití: Přidat uživatele

Případ užití	Zrušit
Stručný popis	Zrušení poslední vyvolané funkce.
Primární aktéři	Přihlášený
Vstupní podmínky	Žádné
Alternativní scénář: 1. Alternativní scénář začíná, když aktér vybere volbu Zrušit. 2. Systém zruší poslední vyvolanou funkci systému.	
Výstupní podmínky	Zrušení poslední vyvolané funkce.

Tabulka 1.2 Alternativní případ užití: Zrušit

Případ užití	Zobrazit uživatele
Stručný popis	Zobrazení informací o uživatelích.
Primární aktéři	Vedoucí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu zobrazit uživatele. 2. Vedoucí může vyplnit vyhledávací řetězec do formulářového pole. 3. Pokud je zadán neprázdný vyhledávací řetězec v kroku 2. 3.1. Systém vybere pouze uživatele, kteří obsahují zadaný řetězec ve jméně, přihlašovacím jméně, nebo popisu. 4. Pokud je zadán prázdný vyhledávací řetězec v kroku 2. 4.1. Systém vybere všechny uživatele systému. 5. Systém zobrazí vybrané uživatele.	
Výstupní podmínky	Systém zobrazil uživatele.
Alternativní scénáře	Žádné

Tabulka 1.3 Případ užití: Zobrazit uživatele

Případ užití	Smazat uživatele
Stručný popis	Odstranění uživatele z informačního systému
Primární aktéři	Vedoucí
Vstupní podmínky	Vybrán účet uživatele.
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu smazat uživatele. 2. Systém odstraní účet uživatele.	
Výstupní podmínky	Uživatelský účet odstraněn ze systému.
Alternativní scénáře	Žádné

Tabulka 1.4 Případ užití: Smazat uživatele

Případ užití	Přidat soutěž
Stručný popis	Vytvoření specifikace nové soutěže.
Primární aktéři	Vedoucí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu Přidat soutěž. 2. Systém zobrazí formulář s údaji. 3. Vedoucí vyplní název, popis soutěže a zvolí časový interval registrace týmů. 4. Pokud je název soutěže prázdný nebo není unikátní v rámci systému: 4.1. Systém vypíše chybové hlášení, návrat k bodu 3. 5. Systém uloží novou soutěž.	
Výstupní podmínky	Systém uložil novou soutěž.
Alternativní scénáře	Zrušit

Tabulka 1.5 Případ užití: Přidat soutěž

Mohli bychom také uvažovat případ užití „Upravit soutěž“, nicméně by byl téměř shodný s tabulkou 1.5, jediným rozdílem by bylo, že zobrazený formulář bude obsahovat předvyplněné údaje modifikované soutěže.

Obdobný postup je možné aplikovat i u ostatních případů užití, které vytvářejí nový objekt. Z tohoto důvodu nebudou případy užití pro úpravu dále zmíněné.

Případ užití	Smazat soutěž
Stručný popis	Odstranění vybrané soutěže ze systému
Primární aktéři	Vedoucí
Vstupní podmínky	Soutěž nebyla zahájena vygenerováním rozpisů.
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu smazat soutěž. 2. Pokud soutěž nebyla zahájena 2.1. Systém odstraní soutěž a všechny objekty s ní spojené.	
Výstupní podmínky	Soutěž byla odstraněna ze systému.
Alternativní scénáře	Žádné

Tabulka 1.6 Případ užití: Smazat soutěž

Případ užití	Přidat kategorii
Stručný popis	Přidání kategorie do soutěže.
Primární aktéři	Vedoucí
Vstupní podmínky	Vybrána konkrétní soutěž.
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu přidat kategorii. 2. Systém zobrazí formulář. 3. Vedoucí vyplní název kategorie a případně její popis. 4. Pokud je název kategorie prázdný nebo není unikátní v rámci soutěže: 4.1. Systém vypíše chybové hlášení, návrat k bodu 3. 5. Systém uloží novou kategorii do vybrané soutěže.	
Výstupní podmínky	Systém uložil novou kategorii.
Alternativní scénáře	Zrušit

Tabulka 1.7 Případ užití: Přidat kategorii

Případ užití	Smazat kategorii
Stručný popis	Smazat kategorii v rámci soutěže.
Primární aktéři	Vedoucí
Vstupní podmínky: 1. Vybrána konkrétní soutěž. 2. Vybrána konkrétní kategorie.	
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu smazat kategorii. 2. Pokud je kategorie přiřazena disciplíně: 2.1. případ užití pokračuje alternativním scénářem Zobrazit chybu. 3. Pokud je touto kategorií označen tým: 3.1. případ užití pokračuje alternativním scénářem Zobrazit chybu. 4. Systém smaže kategorii.	
Výstupní podmínky	Systém smazal kategorii
Alternativní scénáře	Zobrazit chybu

Tabulka 1.8 Případ užití: Smazat kategorii

Případ užití	Zobrazit chybu
Stručný popis	Zobrazení chybového hlášení.
Primární aktéři	Přihlášený
Vstupní podmínky	Systém nemůže řádně dokončit požadovanou akci.
Alternativní scénář: 1. Případ užití začíná, když systém nemůže dokončit požadovanou akci. 2. Systém zobrazí chybové hlášení o vzniklém problému. 3. Systém nabídne možnosti pokračování, návratu na hlavní stránku, nebo odhlášení.	
Výstupní podmínky	Systém zobrazil chybové hlášení.

Tabulka 1.9 Alternativní případ užití: Zobrazit chybu

Případ užití „Přidat hřiště“ je shodný s případem užití „Přidat kategorii“ (viz tabulka 1.8), proto je v tomto místě vynechán.

Případ užití	Smazat hřiště
Stručný popis	Odstranit hrací plochu ze soutěže.
Primární aktéři	Vedoucí
Vstupní podmínky	1. Vybrána konkrétní soutěž. 2. Vybráno konkrétní hřiště.
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu smazat hřiště. 2. Pokud je hřiště přiřazené disciplíně: 2.1. případ užití pokračuje alternativním scénářem Zobrazit chybu. 3. Systém smaže hrací plochu.	
Výstupní podmínky	Systém odstranil hrací plochu.
Alternativní scénáře	Zobrazit chybu

Tabulka 1.10 Případ užití: Smazat hřiště

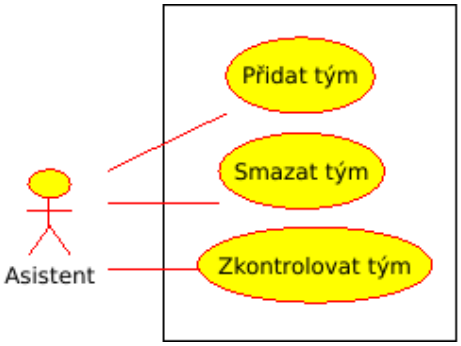
Případ užití	Přidat disciplínu
Stručný popis	Přidání nové disciplíny do soutěže.
Primární aktéři	Vedoucí
Vstupní podmínky	Vybrána konkrétní soutěž.
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu přidat disciplínu. 2. Systém zobrazí formulář. 3. Vedoucí vyplní název disciplíny a případně její popis. 4. Pokud je název disciplíny prázdný nebo není unikátní v rámci soutěže: 4.1. Systém vypíše chybové hlášení, návrat k bodu 3. 5. Systém uloží novou disciplínu do vybrané soutěže.	
Výstupní podmínky	Systém uložil novou disciplínu.
Alternativní scénáře	Zrušit

Tabulka 1.11 Případ užití: Přidat disciplínu

Případ užití	Smazat disciplínu
Stručný popis	Odstranit disciplínu ze soutěže.
Primární aktéři	Vedoucí
Vstupní podmínky	1. Vybrána konkrétní soutěž. 2. Vybrána konkrétní disciplína.
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu smazat disciplínu. 2. Pokud je disciplína přiřazena některému z týmů: 2.1. případ užití pokračuje alternativním scénářem Zobrazit chybu. 3. Systém smaže disciplínu.	
Výstupní podmínky	Systém odstranil disciplínu.
Alternativní scénáře	Zobrazit chybu

Tabulka 1.12 Případ užití: Smazat disciplínu

1.4.3 Scénáře asistenta



Obrázek 1.4 Diagram případů užití pro roli Asistent

Případ užití	Přidat tým
Stručný popis	Přidání nového týmu do soutěže.
Primární aktéři	Asistent
Vstupní podmínky	Vybrána konkrétní soutěž.
Hlavní scénář: 1. Případ užití začíná, když asistent vybere volbu přidat tým. 2. Systém zobrazí formulář. 3. Asistent vyplní název týmu, kategorii a případně další popis. Dále vybere věkovou kategorii a vyplní jméno, příjmení, email případně telefon vedoucího týmu. V případě že je v týmu více soutěžících, jsou přidány i jejich údaje. Poté označí disciplíny, kterých se chce tým zúčastnit. 4. Systém uloží nový tým do soutěže.	
Výstupní podmínky	Systém uložil nový tým.
Alternativní scénáře	Zrušit

Tabulka 1.13 Případ užití: Přidat tým

Případ užití	Smazat tým
Stručný popis	Odstranit tým ze soutěže.
Primární aktéři	Asistent
Vstupní podmínky: 1. Vybrána konkrétní soutěž. 2. Vybrán konkrétní tým.	
Hlavní scénář: 1. Případ užití začíná, když asistent vybere volbu smazat tým. 2. Systém smaže požadovaný tým.	
Výstupní podmínky	Systém odstranil tým ze soutěže.
Alternativní scénáře	Žádné

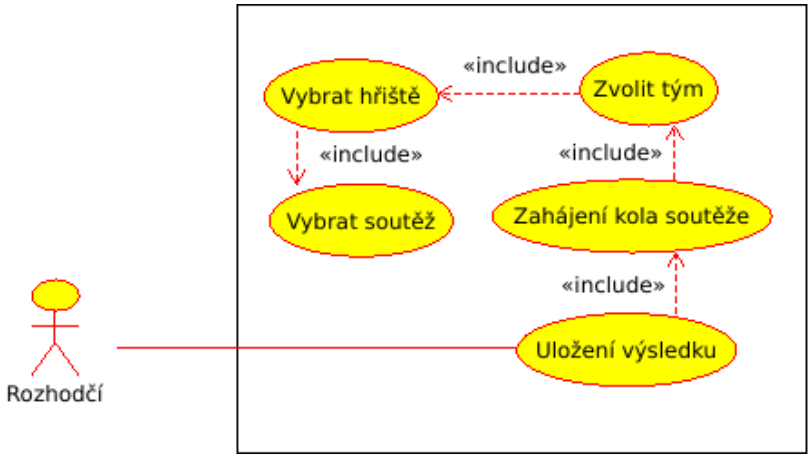
Tabulka 1.14 Případ užití: Smazat tým

Případ užití	Zkontrolovat tým
Stručný popis	Označení týmu, který splnil veškeré požadavky na účast.
Primární aktéři	Asistent
Vstupní podmínky: 1. Vybrána konkrétní soutěž. 2. Vybrán konkrétní tým.	
Hlavní scénář: 1. Případ užití začíná, když asistent označí tým za zkontrolovaný. 2. Systém označí požadovaný tým.	
Výstupní podmínky	Systém označil zkontrolovaný tým.
Alternativní scénáře	Žádné

Tabulka 1.15 Případ užití: Zkontrolovat tým

1.4.4 Scénáře rozhodčího

Obrázek 1.5 ilustruje případy užití rozhodčího, který dohlíží na průběh soutěže a pomocí svého mobilního zařízení může ukládat dosažené výsledky jednotlivých týmů.



Obrázek 1.5 Diagram případů užití pro roli Rozhodčí

Případ užití	Uložení výsledku
Stručný popis	Uložení výsledků jednoho kola disciplíny
Primární aktéři	Rozhodčí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Zahrnuje případ užití: Zahájení kola soutěže. 2. Systém zobrazí formulář, do kterého se vyplní dosažený výsledek. 3. Rozhodčí vyplní informace v závislosti na požadavcích disciplíny. 4. Systém uloží výsledek.	
Výstupní podmínky	Uložen dosažený výsledek týmu v dané disciplíně.
Alternativní scénáře	Zrušit

Tabulka 1.16 Případ užití: Uložení výsledku

Případ užití	Zahájení kola soutěže
Stručný popis	Zahájení kola soutěže na daném stanovišti.
Primární aktéři	Rozhodčí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Zahrnuje případ užití: Zvolit tým. 2. Systém označí tým, který se má dostavit. 3. Pokud se zvolený tým účastní jiného kola: 3.1. Uživatel zvolí možnost zpět. 3.2. Systém zruší označení týmu, který se má dostavit. 3.3. Návrat k bodu 1. 4. Systém zobrazí vyzvání k dostavení týmu na dané hřiště.	
Výstupní podmínky	Označen tým pro konání kola soutěže na daném hřišti.
Alternativní scénáře	Zrušit

Tabulka 1.17 Případ užití: Zahájení kola soutěže

Případ užití	Zvolit tým
Stručný popis	Vybrat tým ze seznamu pro dané hřiště.
Primární aktéři	Rozhodčí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Zahrnuje případ užití: Vybrat hřiště. 2. Systém zobrazí seznam týmů pro vybrané hřiště. 3. Pokud rozhodčí zadá vyhledávací řetězec do vstupního pole filtru. 3.1. Systém zobrazí týmy obsahující v názvu zadaný řetězec. 4. Rozhodčí vybere tým. 5. Systém si zapamatuje vybraný tým.	
Výstupní podmínky	Systém si zapamatoval vybraný tým.
Alternativní scénáře	Zrušit

Tabulka 1.18 Případ užití: Vybrat tým

Případ užití	Vybrat hřiště
Stručný popis	Výběr hřiště z množiny definovaných pro danou soutěž.
Primární aktéři	Rozhodčí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Zahrnuje případ užití: Vybrat soutěž. 2. Systém zobrazí seznam všech hřišť v dané soutěži. 3. Rozhodčí vybere hřiště. 4. Systém si zapamatuje vybrané hřiště.	
Výstupní podmínky	Systém si zapamatoval vybrané hřiště.
Alternativní scénáře	Zrušit

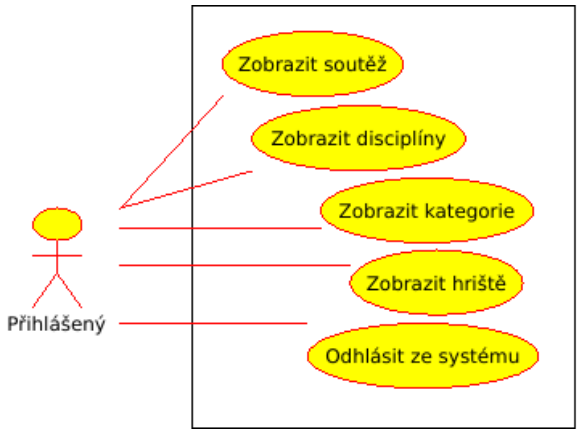
Tabulka 1.19 Případ užití: Vybrat hřiště

Případ užití	Vybrat soutěž
Stručný popis	Vybrat soutěž pro zaznamenání výsledku.
Primární aktéři	Rozhodčí
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když rozhodčí zobrazí hlavní stránku mobilní aplikace. 2. Systém vypíše seznam všech soutěží, které jsou rozehrané. 3. Rozhodčí zvolí soutěž. 4. Systém si zapamatuje vybranou soutěž.	
Výstupní podmínky	Systém si zapamatoval vybranou soutěž.
Alternativní scénáře	Zrušit

Tabulka 1.20 Případ užití: Vybrat soutěž

1.4.5 Scénáře přihlášeného aktéra

Na obrázku 1.6 jsou uvedeny případy užití společné pro přihlášené uživatele.



Obrázek 1.6 Diagram případů užití
pro roli Přihlášený

Případ užití	Odhlášení ze systému
Stručný popis	Odhlášení aktuálně přihlášeného uživatele.
Primární aktéři	Přihlášený
Vstupní podmínky	Uživatel je přihlášený do informačního systému.
Hlavní scénář: 1. Případ užití začíná, když přihlášený uživatel zvolí volbu odhlášení. 2. Systém odhlásí uživatele.	
Výstupní podmínky	Uživatel odhlášen.
Alternativní scénáře	Žádné

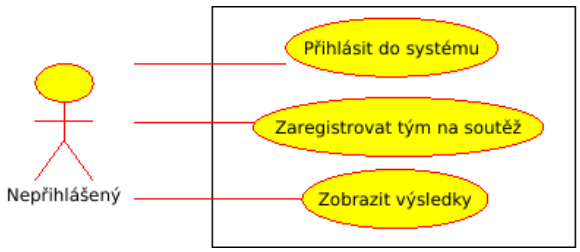
Tabulka 1.21 Případ užití: Odhlášení ze systému

Případy užití Zobrazit disciplíny, Zobrazit kategorie, Zobrazit hřiště jsou definovány se stejným záměrem, liší se pouze v typu zobrazovaného objektu a mohou navazovat na případ užití Zobrazit soutěž popsaného v tabulce 1.22.

Případ užití	Zobrazit soutěž
Stručný popis	Zobrazení informací o soutěži.
Primární aktéři	Přihlášený
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když vedoucí vybere volbu zobrazit seznam soutěží. 2. Vedoucí může vyplnit vyhledávací řetězec do formulářového pole. 3. Pokud je zadán neprázdný vyhledávací řetězec v kroku 2. 3.1. Systém vybere pouze soutěže, které obsahují zadaný řetězec v názvu soutěže nebo jeho popisu. 4. Pokud je zadám prázdný vyhledávací řetězec v kroku 2. 4.1. Systém vybere všechny soutěže. 5. Systém zobrazí vybrané soutěže. 6. Vedoucí vybere požadovanou soutěž. 7. Systém zobrazí informace o soutěži.	
Výstupní podmínky	Systém zobrazil informace o soutěži.
Alternativní scénáře	Žádné

Tabulka 1.22 Případ užití: Zobrazit soutěž

1.4.6 Scénáře nepřihlášeného aktéra



Obrázek 1.7 Diagram případů užití
pro roli Nepřihlášený

Případ užití	Přihlášení do systému
Stručný popis	Přihlášení neautentizovaného uživatele do systému.
Primární aktéři	Nepřihlášený
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když nepřihlášený zobrazí hlavní stránku systému. 2. Systém zobrazí formulář s přihlašovacím jménem a heslem. 3. Nepřihlášený uživatel vyplní požadované informace. 4. Pokud systém provedl autentizaci neúspěšně: 4.1. Systém zobrazí chybové hlášení, návrat k bodu 2. 5. Systém provede autorizaci uživatele.	
Výstupní podmínky	Uživatel byl přihlášen do informačním systému.
Alternativní scénáře	Žádné

Tabulka 1.23 Případ užití: Přihlášení do systému

Případ užití	Zaregistrovat tým na soutěž
Stručný popis	Samoobslužné přihlášení týmu na soutěž.
Primární aktéři	Nepřihlášený
Vstupní podmínky	Soutěž má povolenou registraci v čase zobrazení stránky.
Hlavní scénář: 1. Případ užití začíná, když nepřihlášený zobrazí registrační stránku. 2. Systém zobrazí formulář. 3. Nepřihlášený vyplní název týmu a může připojit další popis. Dále vybere věkovou kategorii a vyplní jméno, příjmení, email případně telefon vedoucího týmu. V případě že je v týmu více soutěžících, jsou přidány i jejich údaje. Poté označí disciplíny, kterých se chce tým účastnit. 4. Systém uloží nový tým do soutěže.	
Výstupní podmínky	Tým byl zaregistrovaný na danou soutěž.
Alternativní scénáře	Žádné

Tabulka 1.24 Případ užití: Zaregistrovat tým na soutěž

Případ užití	Zobrazit výsledky
Stručný popis	Zobrazení aktuálního pořadí týmů.
Primární aktéři	Nepřihlášený
Vstupní podmínky	Žádné
Hlavní scénář: 1. Případ užití začíná, když nepřihlášený zobrazí stránku s výsledky. 2. Systém zobrazí pořadí podle disciplín a kategorií.	
Výstupní podmínky	Systém zobrazil výsledky.
Alternativní scénáře	Žádné

Tabulka 1.25 Případ užití: Zobrazit výsledky

1.5 Obdobné existující projekty

Při hledání obdobných řešení jsem narazil na projekty, které je možné rozdělit do základních tří skupin:

- *Komerční systémy* se soustředí zejména na sportovní ligy a tradiční populární skupinové sporty jako je například hokej, fotbal nebo tenis. Umožňují generovat a plánovat rozpisy zápasů, shromažďovat jejich výsledky a generovat statistiky. Mohou také napomáhat s agendami v rámci provozu sportovního klubu. Pro různé sporty se mohou lišit množstvím zaznamenávaných statistických informací z průběhu zápasu. Jejich zaměření je spíše pro dlouhodobou podporu sportovních klubů, než pro využití na jednorázové akce.

- *Systémy podporující organizaci*, jejichž představitelem může být portál rozpisy zápasů [4], který se zabývá generováním různých typů tabulek rozpisů. V rámci

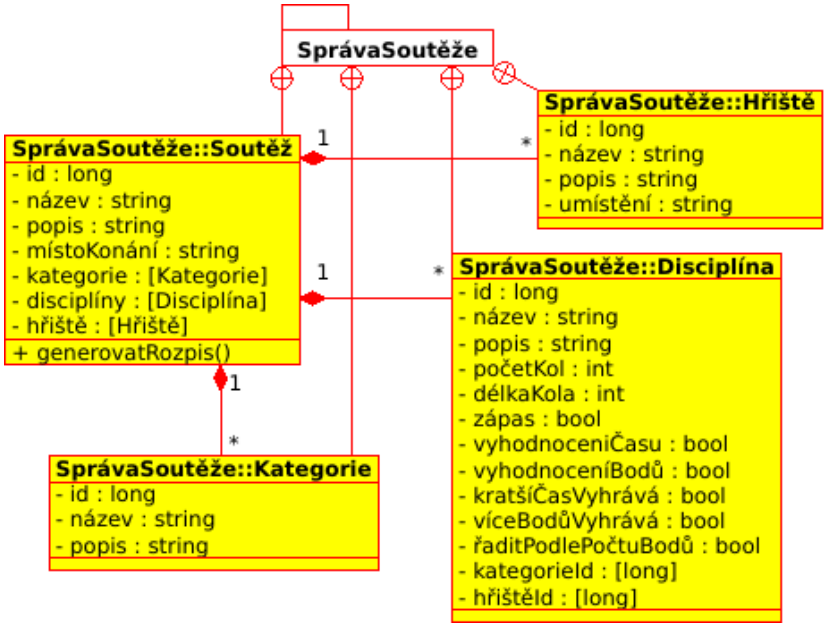
projektu je poskytován program pro zadávání výsledků na základě zhotoveného rozpisu. Tento program se spouští jako desktopová aplikace, nicméně nepodporuje distribuovaný sběr výsledků od více organizátorů v rámci soutěže. Projekt také neobsahuje řešení pro registraci týmů a jejich další organizaci v rámci soutěže.

- *Systémy s otevřeným kódem* jako je například projekt SportChef [5], který vypadal zajímavě díky svému popisu a zvolenými technologiemi, nicméně se jedná o jeden z řady nedokončených projektů umožňující pouze přihlášení a vytváření událostí, další funkce nebyly zapracovány. Z dostupných informací lze vyčíst, že práce na projektu již neprobíhají několik let.

1.6 Identifikace tříd

Na základě předchozího popisu problému byly identifikovány třídy, které modelují specifické prvky problémové domény. Pro identifikaci tříd bylo využito analýzy podstatných jmen a sloves následované metodou štítků *CRC*. Jednotlivým třídám jsou přiřazeny základní atributy a odpovědnosti.

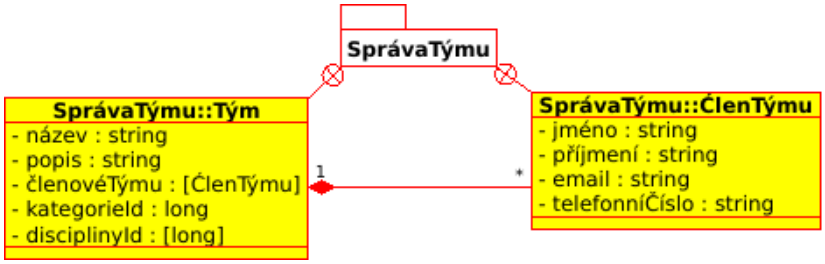
Na obrázku 1.8 je zobrazen balíček tříd reprezentujících údaje o soutěži. Všechny třídy obsahují identifikátor, název a popis. Název soutěže musí být unikátní v rámci celého informačního systému. Ostatní názvy musí být unikátní v rámci dané soutěže. Význam atributů tříd je zřejmý z jejich pojmenování.



Obrázek 1.8 Diagram tříd: Soutěž

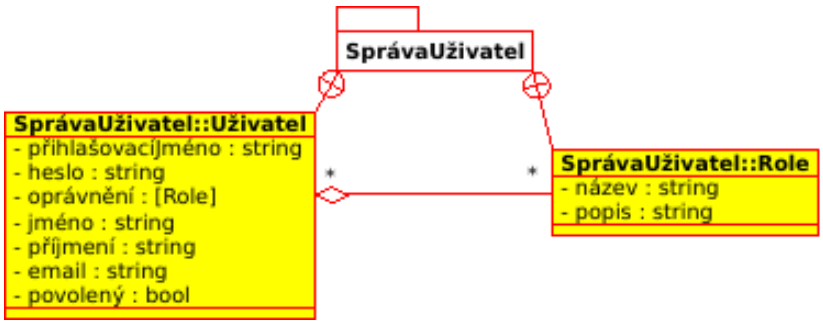
Jedinou výjimkou je třída disciplíny, která obsahuje atributy určující jakým způsobem budou vyhodnoceny výsledky. Atribut *zápas* určuje, zda-li se disciplíny účastní dva týmy, nebo se jedná o úkol pro jeden tým. Na základě tohoto atributu bude buďto

generován rozpis zápasů ve stylu každý s každým, nebo bude vytvořen list jednotlivých týmů účastnících se dané disciplíny. *Vyhodnocení času* respektive *vyhodnocení bodů* určuje, zda-li se v rámci plnění disciplíny bude vyhodnocovat jak dlouho trvalo úkol splnit, respektive kolik bodů tým získal. Volby *kratší čas vyhrává* a *více bodů vyhrává* umožňuje nastavit, jestli je v rámci vyhodnocení upřednostňován tým, který byl rychlejší v plnění disciplíny a nebo tým který získal více bodů. Poslední volba *řadit podle počtu bodů* je vyhodnocena v případě, že disciplína má nastaveno jak vyhodnocení času tak i vyhodnocení počtu bodů. Její nastavení ovlivňuje zda-li je vítězný tým primárně určen podle času nebo počtu bodů. V případě že se výsledky týmů shodují v prioritním parametru, jsou následně seřazeny podle zbývající možnosti. Poslední dva atributy jsou seznamy indentifikátorů kategorií a hřišť, které jsou disciplíně přiřazeny.



Obrázek 1.9 Diagram tříd: Tým

Na obrázku 1.9 je zobrazen balíček obsahující informace o týmech a jejich členech. Název týmu musí být unikátní v rámci soutěže. *KategorieId* je identifikátor věkové kategorie, kterou členové týmu zastupují. *DisciplínyId* je seznam identifikátorů disciplín, na které se tým v rámci soutěže přihlásil.



Obrázek 1.10 Diagram tříd: Uživatelé

Uživatelé systému jsou spravováni v rámci balíčku na obrázku 1.10. Uživatel se identifikuje svým *přihlašovacím jménem* a *heslem*, které je uloženo v zašifrované podobě. *Oprávnění* jsou uživateli přidělena na základě rolí. V předchozím textu byly identifikováni aktéři a z tohoto seznamu vyplývá, že role má následující tři instance: vedoucí, rozhodčí a asistent. Atribut *povolený* musí být pravdivý pro uživatele, který má povolení se aktuálně přihlásit do informačního systému.

1.7 Validační kritéria

V tomto oddíle jsou definována kritéria pomocí kterých budeme ověřovat funkcionalitu informačního systému. Určíme případové akce a k nim uvedeme požadovaný výsledek. Na základě takto vydefinovaných požadavků budeme demonstrovat funkčnost systému a jeho soulad se záměrem a cílem projektu.

1. Vytvoření nové soutěže
    - Akce: Vytvořit pojmenovanou soutěž do které se dá v současnou chvíli zaregistrovat s jednou kategorií, disciplínou a hrací plochou.
    - Požadovaný výsledek: Soutěž je vytvořena a uložena.
2. Samoobslužné přihlášení týmu do soutěže
    - Akce: Zaregistrovat soutěžícího na vytvořenou soutěž.
    - Požadovaný výsledek: Je vytvořen a uložen nový tým.
3. Kontrola týmu a potvrzení jeho ověření
    - Akce: Označit zkontrolovaný tým.
    - Požadovaný výsledek: Tým označen jako zkontrolovaný.
4. Výzva týmu k dostavení a plnění disciplíny
    - Akce: V aplikaci pro rozhodčí vybrat soutěž s alespoň třemi zaregistrovanými a zkontrolovanými týmy. Označit tým který se má dostavit ke zvolené disciplíně.
    - Požadovaný výsledek: Je zobrazena výzva k dostavení týmu k dané hrací ploše.
5. Uložení výsledků
    - Akce: V aplikaci pro rozhodčí vybrat soutěž s alespoň třemi zaregistrovanými a zkontrolovanými týmy. Postupně těmto týmům uložit výsledek ve zvolené disciplíně.
    - Požadovaný výsledek: Výsledky jednotlivých týmů jsou uloženy.
6. Zobrazení aktuálního pořadí týmů
    - Akce: Zobrazit aktuální pořadí týmů.
    - Požadovaný výsledek: Je zobrazeno aktuální pořadí jednotlivých týmů.

2 POUŽITÉ TECHNOLOGIE

Obsahem této kapitoly je stručný přehled vybraných technologií pro implementaci informačního systému. Většinou se jedná o standardní technologie v daném odvětví. Výběr byl zaměřen na léty prověřené komunikační protokoly, knihovny, nebo pracovní rámce, které mají širokou uživatelskou základnu a jsou nadále aktivně udržovány a rozvíjeny.

2.1 Java

Java je technologie zahrnující programovací jazyk a výpočetní platformu, která byla původně vydána firmou Sun Microsystems v roce 1995. [6] Od té doby prošla značným vývojem a stala se téměř standardem při budování rozsáhlých firemních systémů, které běží v prostředí s požadavky na rychlost, vysokou dostupnost a zabezpečení. Java je programovací jazyk vyšší úrovně, který je relativně jednoduchý, multiplatformní a není jen čistě objektově orientovaný, ale inspiruje se i dalšími programovacími paradigmaty, jako je například funkcionální programování.

Vlastní zdrojový kód se zapisuje do textových souborů, které se pomocí kompilátoru jazyka překládají do takzvaného bajtového kódu, což je strojový jazyk virtuálního stroje spouštěného v rámci Java platformy. Tento kód je přenosný mezi různými operačními systémy. Java platforma je softwarové prostředí, které umožňuje spouštět programy. Je dostupná pro různé počítačové architektury a operační systémy. Díky tomu je možné jednou napsaný kód spouštět kdekoliv kde je platforma dostupná. V současné době celou Java technologii vlastní mezinárodní korporace Oracle. [7]

2.2 Project Lombok

Knihovna jazyka Java umožňující automatizované vytváření základních metod objektu. Pokud vytvoříme jednoduchou třídu, která bude obsahovat pouze soukromé instanční proměnné, pomocí anotací můžeme definovat jaké metody se mají automaticky při překladu vygenerovat. Je tak možné používat metody pro získání nebo nastavení hodnoty instanční proměnné, konstruktory s různými parametry, ale i další základní metody objektu jako jsou například equals a hashCode. [14]

2.3 JUnit

Platforma pro spouštění automatizovaných testů v prostředí JVM. Zjednodušuje testování malých částí systémů, takzvaných jednotek a umožňuje jejich rychlé a opakované spouštění. Díky tomu je možné jednoduše ověřit, že u části kódu pokryté testy nedošlo k narušení požadované funkcionality. Proces testování je rozdělen do testovacích případů. Každý

testovací případ využívá toho, že jsou známa vstupní data a očekává se konkrétní reakce systému. Je vhodné testovat nejen reakce systému na korektní požadavky, ale taky jeho chování při uvedení do chybového stavu. [8]

2.4 Spring

Projekt zastřešující pracovní rámce usnadňující vývoj aplikací běžících na platformě Java. Zaměřuje se na poskytnutí implementace často se opakujících problematik v souvislosti s vývojem a provozem softwarových projektů. [9]

- *Spring Framework* je jádro celého projektu. Obsahuje implementaci pro správu a vkládání závislostí v rámci výsledného programu, publikování a obsluhu událostí, podporu pro testování a přístup k datům. V neposlední řadě umožňuje také tvorbu webových aplikací. [10]
- *Spring Data JDBC* usnadňuje práci s datovými repozitáři. Repozitářem je v tomto případě myšlena abstrakce relační databáze. Jeho přínosem je jednoduchá koncepce práce s daty. [11]
- *Spring Security* je výkonný a jednoduše přizpůsobitelný pracovní rámec pro identifikaci uživatelů a řízení jejich přístupů na základě oprávnění. Jeho součástí je také ochrana proti útokům. [12]

2.5 Thymeleaf

Moderní šablonovací systém pro tvorbu nejen webových stránek, běžící na straně serveru a využívající platformu Java. Jeho hlavní výhoda tkví v tom, že nezasahuje násilím do originálního značkovacího jazyka dokumentu, ale plně využívá možností jmenného prostoru k tomu, aby definoval vlastní elementy a atributy. Díky tomu je možné i samotné šablony prohlížet bez předchozího zpracování. Šablona je validní dokument původního značkovacího jazyka. Thymeleaf má také širokou podporu ze strany vývojových integrovaných prostředí. [13]

2.6 Gradle

Nástroj pro automatizaci sestavování projektů, který je navrhnut dostatečně univerzálně, aby mohl sestavovat jakýkoliv softwarový projekt. Dosahuje velkého výkonu díky tomu že vykonává jen úkoly u kterých se změnila vstupní nebo výstupní data. Pro svůj běh potřebuje mít k dispozici Java Development Kit. Nejčastěji používané vývojové integrované prostředí mají implementovanou podporu pro spouštění gradle úkolů. [15]

2.7 HTML5

Jednoduchý textový značkovací jazyk, který umožňuje popsat sémantiku jednotlivých částí dokumentu. Jeho název je zkratkou z Hyper Text Markup Language. Využívá se hlavně pro tvorbu webových stránek. Dokument se skládá z vlastního textu, který je obalen pomocí elementů a jejich atributů. Zapisuje se do textových souborů jejichž interpretaci má na starost nejčastěji internetový prohlížeč. [16]

2.8 CSS3

Pomocí jazyka Cascading Style Sheets specifikujeme vzhled dokumentu. Nejčastěji ho můžeme najít ve spojitostí s HTML dokumenty. Jedná se o strukturovaný jazyk, který elementům dokumentu přiřazuje vizuální reprezentaci pomocí definování šablon. Šablona stylů je textový soubor obsahující pravidla, vlastnosti a jejich hodnoty. Dále také umožňuje definovat zobrazení pro rozdílné zařízení. [16]

2.9 JavaScript

Skriptovací jazyk převážně spouštěný v internetovém prohlížeči uživatele. Původně byl používaný hlavně pro rozšíření funkčnosti webových stránek, v dnešní době existují i implementace bežící na serveru. Zdrojový kód je psaný do textového souboru a před spuštěním není zapotřebí předchozí kompilace, o vše se postará interpret jazyka. Jeho hlavním přínosem při tvorbě webových stránek je možnost dynamické změny obsahu nebo vzhledu. JavaScript původně pochází z dílny firmy Netscape, během následujících let se stal standardem označeným názvem ECMAScript. [17]

2.10 JSON

Jednoduchý textový formát pro výměnu dat, jehož zkratka vychází z názvu JavaScript Object Notation. Jeho výhoda je v jednoduchosti jak pro zpracování počítačem tak i přímou manipulaci člověkem. Tento formát je nezávislý na programovacím jazyku, nicméně používá konvence podobné pro rodinu programovacích jazyků typu C. Formát je založen na dvou základních strukturách, první je soubor dvojic název a hodnota, druhý je seřazený seznam hodnot. Toto jsou univerzální datové struktury, veškeré současné moderní programovací jazyky je podporují. [18]

2.11 HTTP

Je zkratka pro Hypertext Transfer Protocol, který byl původně navržen pro přenos dokumentů, jako je např. HTML, začátkem devadesátých let. V dnešní době je široce používán pro komunikaci v rámci internetové sítě, přes kterou se přenáší různé data

včetně multimediálního obsahu. Jedná se o bezstavovou komunikaci typu klient - server, kdy klient navazuje spojení, posílá dotazy a server následně odpovídá. Protože protokol samotný je bezstavový, využívají se podpůrné mechanizmy pro udržení relace v rámci aplikační logiky, jako je např. mechanizmus nazývaný HTTP cookies. [19] [20]

2.12 Liquibase

Při vývoji systému ukládající data do relační databáze je v rámci rozšiřování požadována postupná změna relačního schématu. Tento projekt řeší automatizaci a organizaci změn schématu relační databáze čímž se nejen urychluje celý proces přechodu k nové verzi, ale také eliminuje nutnost ručně prováděných změn. V neposlední řadě také předchází vzniku možných chyb z nepozornosti nebo nedostatečné zkušenosti obsluhy. Mezi jeho výhody patří možnost definice jednoho databázového schématu a jeho následnou aplikaci na kterýkoliv z podporovaných databázových systémů. [21]

2.13 H2

Implementace relační databáze v jazyce Java. Umožňuje spuštění v serverovém režimu, nebo přímé zabudování do aplikace. Její výhoda, při vývoji informačního systému, je možnost spuštění databáze v paměti, což umožňuje rychlejší inicializaci a po zastavení databáze dojde ke ztrátě veškerých informací. Její součástí je i konsolová aplikace spustitelná v internetovém prohlížeči. [22]

2.14 PostgreSQL

Tato relační databáze původně vznikla na kalifornské univerzitě v Berkeley. Díky jejímu dlouholetému vývoji se jedná o jednu z nejvyspělejších databázových technologií, kterou lze svobodně bez omezení používat. Její implementaci lze provozovat na všech současných rozšířených operačních systémech. [23]

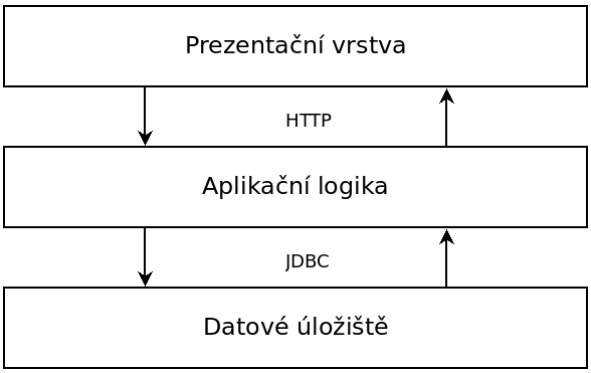
II. PRAKTICKÁ ČÁST

3 ARCHITEKTURA SYSTÉMU

V této části je popsána architektura systému z nejvyššího pohledu abstrakce na celý informační systém. Obsahuje stručný popis jednotlivých vrstev třívrstvého návrhu a způsob vzájemné komunikace mezi jednotlivými vrstvami. Následně je přiblížen architektonický přístup *MVC* použitý v rámci prezentační vrstvy.

3.1 Třívrstvá architektura

Pro implementaci informačního systému byla zvolena třívrstvá architektura skládající se z prezentační vrstvy, aplikační logiky a datového úložiště znázorněné na obrázku 3.1. Uživatelský požadavek typicky vyvolá tok událostí probíhající od prezentační vrstvy přes aplikační logiku k datovému úložišti a obráceným směrem je navrácena odpověď.



Obrázek 3.1 Schéma třívrstvé architektury

1. *Prezentační vrstva* představuje uživatelské rozhraní systému, tato vrstva bývá označována termínem „front end“. Jejím účelem je prezentace informací směrem k uživateli a přeposílání uživatelských požadavků zpět informačnímu systému. Vlastní architektura prezentační vrstvy je popsána v následujícím oddíle.

   Do této vrstvy spadají veškeré soubory s příponou html obsahující šablony nebo stránky značkovacího jazyka *HTML5*, soubory s příponou css obsahující definice kaskádových stylů, soubory s příponou js obsahující *JavaScript* a soubory obsahující obrázky.

   Komunikace mezi prezentační a aplikační vrstvou je zprostředkována pomocí komunikačního protokolu *HTTP* přes který se přenášejí data ve formátu *JSON*.

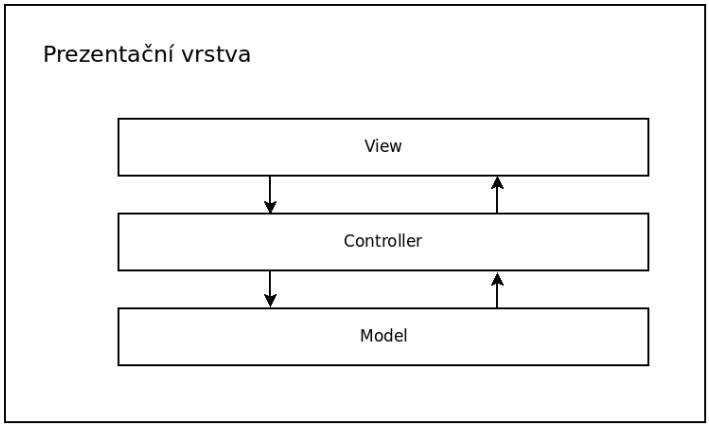
2. *Aplikační logika* obsahuje veškeré obchodní procesy a kontroly konzistence dat, bývá označována termínem „back end“.

   Komunikace mezi aplikační logikou a datovým úložištěm je zprostředkována pomocí *JDBC API*.

3. *Datové úložiště* je relační systém řízení a báze dat (*DBMS*). Úkolem této vrstvy je zajištění persistentního uložení dat.

3.2 Model View Controller

Jedná se o realizaci návrhového vzoru *MVC* [24], který odděluje části starající se jakou formou jsou data prezentována, vlastní data a programovou logiku zpracování. Tato část aplikace se spouští na uživatelově zařízení v internetovém prohlížeči. Schéma je znázorněno na obrázku 3.2.



Obrázek 3.2 Schéma prezentační vrstvy

- *View* definuje jak jsou data prezentována. Skládá se ze souborů značkovacího jazyka *HTML5*, ve kterých je definována struktura dokumentu a souborů kaskádových stylů definující jak budou prezentované informace zobrazeny.
- *Controller* zajišťuje komunikaci a výměnu dat s aplikační vrstvou, řídí zobrazování a sběr dat zadaných uživatelem. V roli kontroleru vystupují veškeré skripty jazyka *JavaScript*.
- *Model* je označení pro vnitřní reprezentaci dat. V tomto případě se jedná o data ve formátu *JSON* přenášená mezi prezentační vrstvou a aplikační logikou.

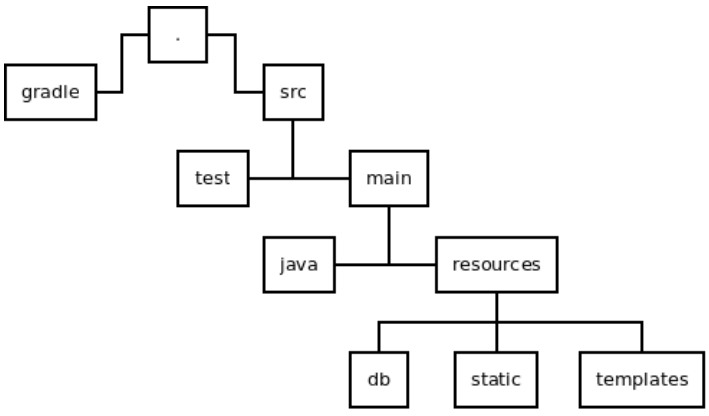
4 IMPLEMENTACE

Projekt je převážně napsán v programovacím jazyku *Java*, pro jeho sestavení je zapotřebí *Java Development Kit* verze 11. Druhý použitý jazyk je *JavaScript*, který pokrývá funkcionalitu prezentační vrstvy aplikace a je interpretován přímo v prohlížeči uživatele.

Pro automatizaci sestavování programu a správu závislostí na knihovnách a pracovních rámcích třetích stran byl použit nástroj *Gradle*, který také definuje základní hierarchii adresářů vyvíjeného projektu. Jádrem celého systému je aplikační pracovní rámec *Spring Framework* implementující mechanizmus obráceného řízení (*IoC*) a řeší správu objektových závislostí na principu jejich vkládání (*DI*). V rámci tohoto pracovního rámce je také využita implementace návrhového vzoru model-view-controller [24] modulu *Spring MVC* pro zpracování požadavků přicházejících přes internetový protokol *HTTP* a následné navrácení odpovědi.

4.1 Struktura projektu

Celým systémem jsou proloženy pomyslné hranice [25], které rozdělují zdrojové kódy do tří částí a to aplikační logiky, persistentního uložení a rozhraní pro komunikaci určené pro předávání dat s prezentační vrstvou. Výhoda této organizace je viditelná v případě modifikace projektu kdy změny v jedné části neovlivní negativně chování zbylých dvou částí. Například bude-li z nějakého důvodu potřeba změnit schéma databáze, veškeré úpravy proběhnout v části odpovědné za persistenci dat, ale již neovlivní chování aplikační logiky, nebo nenaruší rozhraní pro komunikaci s prezentační vrstvou. Objekty mezi zmíněnými hranicemi jsou předávány pomocí servisních tříd a třídních továren [24]. Třídní továrny jsou jednoduše rozpoznatelné již z jejich názvu protože končí příponou *Factory*.



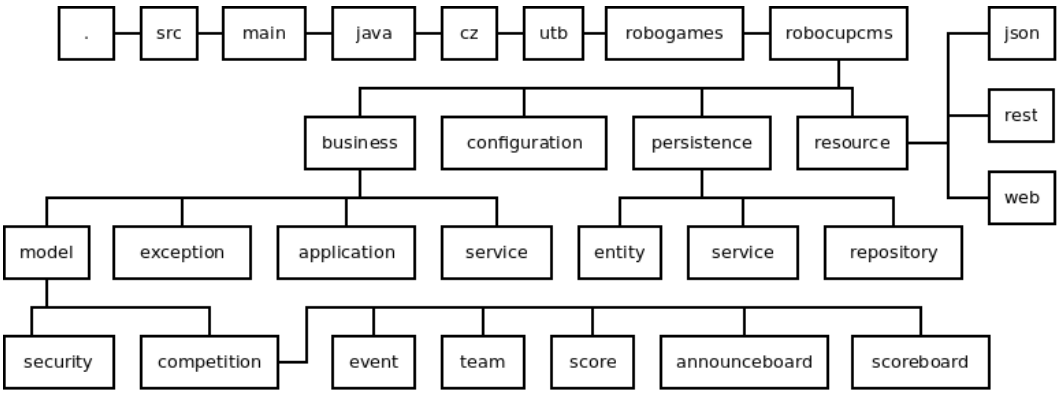
Obrázek 4.1 Schéma struktury kořenového adresáře

Na obrázku 4.1 je zobrazena stromová struktura s počátkem v kořenovém adresáři

projektu. V tomto adresáři se nachází soubory nástroje *Gradle* z nichž *build.gradle* obsahuje popis závislostí informačního systému na knihovnách a pracovních rámcích třetích stran. Sestavení projektu lze spustit pomocí skriptu *gradlew*.

Adresář *src* obsahuje veškeré zdrojové kódy, jeho podadresář *test* seskupuje testovací případy využívající pracovního rámce *JUnit*. Adresář *main* obsahuje podadresář *java* se zdrojovými kódy projektu. V podadresáři *resources* se nachází konfigurační soubor *application.properties* s vlastnostmi *Spring frameworku*, v těchto vlastnostech lze nastavit například parametry připojení k relační databázi. V podadresáři *db* je uložena definice databázového schématu ve formě *YAML* souborů zpracovávaných projektem *Liquibase*. Při startu informačního systému dojde k porovnání definice a aktuálního stavu schéma používané databáze a v případně nesouladu jsou provedeny potřebné změny. Podadresář *static* obsahuje soubory prezentační vrstvy předávané internetovému prohlížeči uživatele bez dodatečných modifikací a podadresář *template* obsahuje šablony *HTML* dokumentů o jejichž zpracování se stará projekt *Thymeleaf*.

Struktura domovského adresáře *robocupcms* pro zdrojové kódy aplikace je znázorněna na obrázku 4.2. V programovacím jazyce Java jsou adresáře souborového systému reprezentovány názvem *balíček*. V této části je možné pozorovat zmíněné rozdělení programu na aplikační logiku (nacházející se v balíčku *business*), komunikační rozhraní (balíček *resource*) a část zodpovědná za persistenci dat (balíček *persistence*). V balíčku *configuration* se nachází třídy propojující informační systém s bezpečnostním rámcem *Spring Security*, další informace jsou uvedeny v oddíle Zabezpečení.



Obrázek 4.2 Schéma struktury aplikační části

V části aplikační logiky obsahuje balíček *model* definice tříd představující základní stavební prvky aplikace, v balíčku *competition* jsou informace o soutěži (*event*), týmu (*team*), reprezentaci dosažených výsledků (*score*), informace o dostavení týmu na dané stanoviště (*announceboard*) a výsledkové listiny (*scoreboard*). Balíček *security* obsahuje třídy reprezentující uživatele systému a jejich přístupové oprávnění.

V balíčku *exception* jsou definovány specifické výjimky řešené problémové domény.

Balíček *application* využívá návrhového vzoru Pozorovatel [24] implementovaného v pracovním rámci *Spring framework*. Jedná se o metodu snížení závislostí mezi jednotlivými objekty tím, že objekt nazývaný posluchač se přihlásit k odebírání určité události od vydavatele. Jakmile dojde v systému k nějaké události, je o tom informován vydavatel, který rozešle informaci svým posluchačům. Třída *RoboCupEventPublisher* definuje službu s metodou pro rozesílání událostí. Jako příklad událost můžeme uvést instanci třídy *GenerateRosterEvent*, po jejímž odeslání dojde k vytvoření pořadníků soutěžících týmů k jednotlivým disciplínám. V posledním balíčku *service* aplikační logiky se nacházejí služby poskytující prostředky pro konzistentní manipulaci s objekty modelu.

Část starající se o persistenci objektů, nacházející se v balíčku *persistence*, do jisté míry kopíruje třídy modelu do perzistentních tříd balíčku *entity*, nicméně jejich sémantický význam je rozdílný. Tyto třídy reprezentují strukturu uložení dat v relační databázi. Balíček *service* obsahuje služby, které se starají o ukládání dat pomocí repozitářů definovaných v balíčku *repository* a převod mezi objekty aplikační logiky a persistentního uložení, k čemuž využívají již zmíněné tovární třídy taktéž definované v balíčku *entity*.

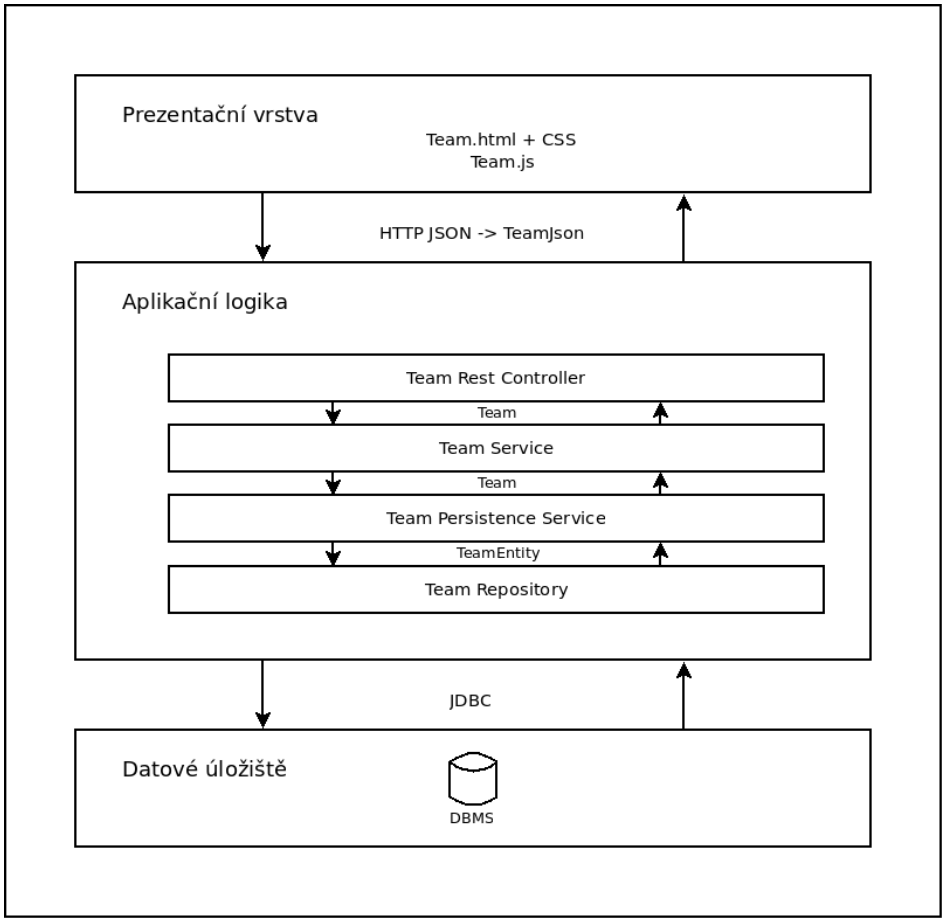
V části odpovědné za komunikaci s prezentační vrstvou nacházející se v balíčku *resource* je umístěn balíček *json*, který obdobným způsobem, jako persistence objektů, kopíruje třídy modelu do tříd reprezentujících *JSON* objekty. Stejně tak jsou přítomny i tovární metody pro převod mezi objekty modelu a *JSON* reprezentací. Balíček *web* obsahuje kontroléry pro vyřizování požadavků přístupu k *HTML* stránkám prezentační vrstvy. Balíček *rest* obsahuje kontroléry, které vyřizují požadavky přičemž přijímají nebo odesílají data ve formátu *JSON*.

4.2 Tok událostí strukturou systému

V návaznosti na popis struktury projektu je v této části sestavena názorná ukázka zpracování požadavku při manipulaci s informacemi o týmu.

V rámci prezentační vrstvy jsou načteny v internetovém prohlížeči uživatele *HTML* dokumenty se svými styly zobrazení a javascriptovými funkcemi. V případě požadavku o data je zaslán *HTTP* požadavek, který je zachycen kontrolérem (instance třídy *TeamRestController*). Kontrolér převezme zaslané informace a obrátí se s požadavkem na službu spravující aplikační logiku (instance třídy *TeamService*). Následně je požadavek předán službě zajišťující persistentní uložení (instance třídy *TeamPersistenceService*), která použije instanci repozitáře (*TeamRepository*) pro načtení dat. Data z repozitáře jsou předána v instanci třídy *TeamEntity*, persistentní služba pomocí třídní továrny *TeamEntityFactory* získá objekt třídy *Team* a navrátí ho službě aplikační logiky. Takto

se postupně objekt dostane zpět do kontroléru, který pomocí třídní továrny *TeamJsonFactory* získá objekt třídy *TeamJson* a přepošle ho v odpovědi na původní *HTTP* dotaz. Popsaný proces je znázorněn na obrázku 4.3.



Obrázek 4.3 Schéma vyřízení požadavků při práci s týmem

4.3 Persistence

Data jsou uložena do relační databáze pomocí objektově relačního mapování zprostředkovávající projekt *Spring Data JDBC* inspirovaný návrhem řízeným doménou (*DDD*). K ukládáni dat se využívají takzvané repozitáře, což jsou abstrakce datového úložiště. Zpravidla je pro každý programový balíček jazyka *Java* vybrána třída, která je považována za kořenovou. Objekty této třídy řídí životní cyklus ostatních entit stejného balíčku. Pro tuto třídu je vytvořen repozitář datového úložiště. Tento vztah mezi entity jednoho balíčku se modeluje takovým způsobem, že v případě odstranění objektu kořenové třídy jsou také odstraněny veškeré jim spravované objekty. Objekty v rámci rozdílných balíčků na sebe uchovávají reference v podobě celočíselných identifikátorů.

4.4 Zabezpečení

Vzhledem k povaze systému, který pracuje jako webová aplikace, je zabezpečení přístupu k jednotlivým zdrojům dat řízeno na základě *HTTP* požadavku. Díky pracovnímu rámci *Spring Security*, který zabezpečuje autentizaci a autorizaci, je možné jednoduše definovat přístupová oprávnění pro každého uživatele podle jemu přidělených rolí.

Třída *RoboCupUserDetailsService* zprostředkovává informace o účtech a jim přidělených rolích pro autentizaci a autorizaci uživatel bezpečnostním rámcem.

Nastavení zabezpečení je specifikováno ve třídě *WebSecurityConfiguration* a v něm vydefinované pravidla shrnuje tabulka 4.1, ve které sloupec *metoda* označuje metodu protokolu *HTTP*, sloupec *vzor* je srovnán s *URL* (znak * zastupuje libovolné množství znaků a dvojznak ** znamená libovolný počet složek v *URL* cestě) a sloupec *oprávnění* specifikuje potřebnou roli pro povolení přístupu.

metoda	vzor	oprávnění
ALL	/favicon.ico	bez oprávnění
ALL	/css/*	bez oprávnění
ALL	/img/*	bez oprávnění
ALL	/Enroll	bez oprávnění
ALL	/js/Enroll.js	bez oprávnění
ALL	/js/request.js	bez oprávnění
ALL	/rest/enroll/**	bez oprávnění
ALL	/js/Util.js	bez oprávnění
ALL	/js/AnnounceBoard.js	bez oprávnění
ALL	/AnnounceBoard	bez oprávnění
ALL	/rest/announceboard/**	bez oprávnění
ALL	/js/Scoreboard.js	bez oprávnění
ALL	/Scoreboard	bez oprávnění
ALL	/rest/scoreboard/**	bez oprávnění
GET	/rest/user/myroles	přihlášený uživatel
ALL	/rest/user*/**	vedoucí
GET	/rest/**	vedoucí, rozhodčí, asistent
ALL	/rest/team*/**	vedoucí, asistent
ALL	/rest/app/**	vedoucí, rozhodčí
ALL	/rest/**	vedoucí

Tabulka 4.1 Specifikace zabezpečení na základě *HTTP* požadavku

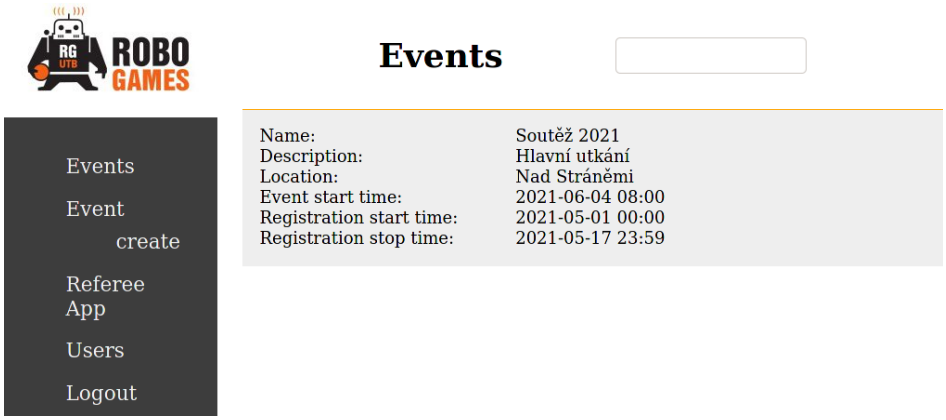
Položky tabulky 4.1 obsahující slovo *Enroll* odkazují na možnost samoobslužného přihlášení týmu do soutěže. Položky obsahující slovo *AnnounceBoard* odkazují na možnost zobrazení tabule vyzývající nástup týmu na stanoviště. Položky obsahující slovo *Scoreboard* odkazují na možnost zobrazení aktuálního pořadí týmů. Veškeré tyto možnosti jsou dostupné bez nutnosti přihlášení uživatele do informačního systému.

5 PRŮVODCE UŽIVATELSKÝM ROZHRANÍM

První přihlášení do informačního systému je popsáno v dodatku zprovoznění. Po úspěšném přihlášení se zobrazí hlavní stránka viz obrázek 5.1.

5.1 Hlavní stránka

Na hlavní stránce je v levé části menu, v pravé části se nachází seznam všech vytvořených soutěží a v pravé horní části je vstupní pole pro filtrování zobrazených výsledků. Zadaný řetězec do vstupního pole filtru vyhledává shodné podřetězce ve jméně, popisu a umístění, přičemž nezáleží na velikosti písmen. Při zvolení jedné ze soutěží dojte k přesměrování na stránku zobrazující podrobnosti týkající se vybrané soutěže. Pokud má uživatel oprávnění *Asistent*, při výběru soutěže dojde k přesměrování na stránku zobrazující týmy asociované s danou soutěží.



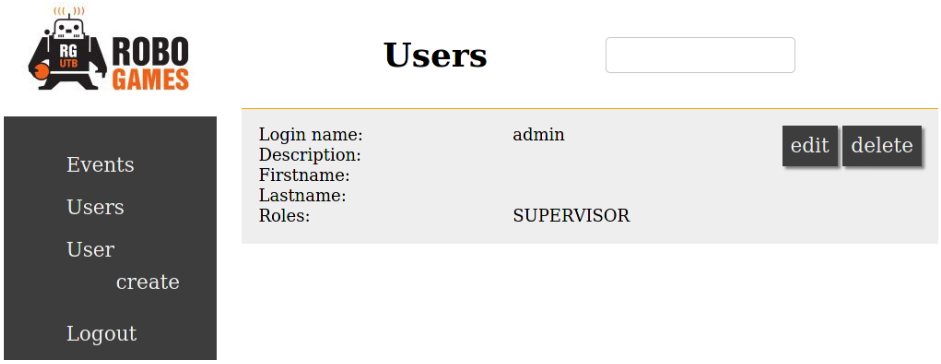
Obrázek 5.1 Uživatelské rozhraní: hlavní stránka

Význam jednotlivých položek menu: *Events* odkazuje na tuto stránku, při její aktivaci dojde k aktualizaci obsahu. *Event create* zobrazí formulář pro vytvoření nové soutěže. Po uložení soutěže dojde automaticky k přesměrování na stránku s jejími podrobnosti. *Referee App* odkazuje na aplikaci pro zadávání výsledků od rozhodčích. *Users* zobrazí stránku umožňující editaci uživatelských účtů a jejich oprávnění. Poslední volba *Logout* odhlásí aktuálního uživatele z informačního systému.

5.2 Uživatelské účty

Uživatelské účty může spravovat pouze uživatel s oprávněním vedoucího. Stránka s možnostmi pro přidání nového uživatele nebo editaci již existujících uživatelských účtů (viz obrázek 5.2) se zobrazí pokud na hlavní stránce je vybrána položka *Users*.

V nastavení uživatele je možné měnit přihlašovací jméno (toto jméno musí být v rámci systému unikátní), heslo (je ukládáno v zašifrované podobě), uživatelovo jméno,



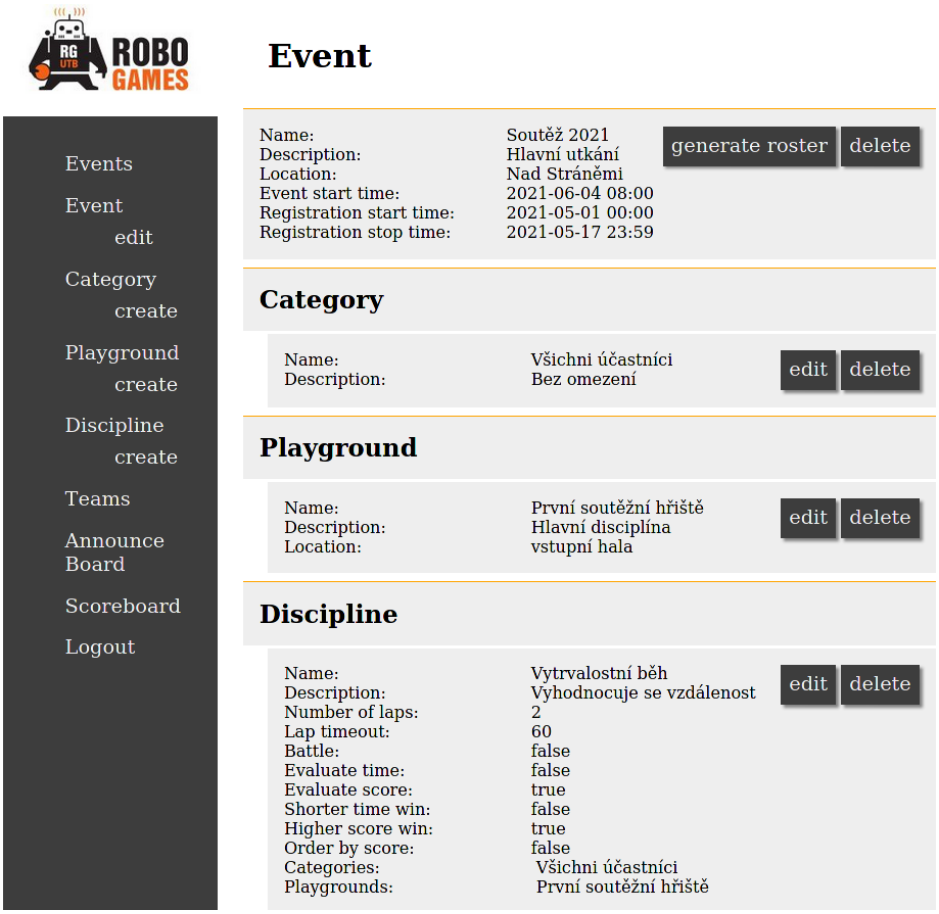
Obrázek 5.2 Uživatelské rozhraní: správa uživatel

příjmení, email, popis a možnost zneplatnění (pokud uživatel není platný, je ve výpisu zobrazen šedou barvou a není mu umožněno přihlášení do systému). Posledním nastavením je přiřazení uživatelského oprávnění, existující oprávnění jsou tři a to *Vedoucí* (*supervisor* - bez omezení jsou dostupné veškeré funkce systému), *Asistent* (*attendant* - umožňuje vytváření a editaci týmů) a *Rozhodčí* (*referee* - oprávněn zadávat výsledky disciplín v aplikaci pro rozhodčí).

5.3 Soutěž

Na stránce s podrobnostmi o soutěži je možné nastavit veškeré parametry připravované soutěže, viz obrázek 5.3. *Category* představuje věkovou kategorii, systém očekává existenci alespoň jedné kategorie, její název musí být unikátní v rámci soutěže. *Playground* představuje hrací plochu, její název musí být unikátní v rámci soutěže. Každá disciplína by měla mít přiřazenu alespoň jednu hrací plochu.

Discipline definuje konkrétní soutěžní činnost a způsob vyhodnocování vítězů, její název taktéž musí být unikátní v rámci soutěže. Její parametr *Number of Laps* informuje o maximálním počtu kol, které mohou proběhnou v rámci plnění disciplíny. *Lap timeout* udává časový interval, jak dlouho může trvat jedno kolo. *Battle* určuje, zda-li se disciplíny účastní dva týmy, nebo se jedná o úkol pro jeden tým. Na základě tohoto atributu bude buďto generován rozpis zápasů ve stylu každý s každým, nebo bude vytvořen list jednotlivých týmů účastnících se dané disciplíny. *Evaluate time* respektive *Evaluate score* určuje, zda-li se v rámci plnění disciplíny bude vyhodnocovat jak dlouho trvalo úkol splnit, respektive kolik bodů tým získal. Volby *Shorter time win* a *Higher score win* umožňuje nastavit, jestli je v rámci vyhodnocení upřednostňován tým, který byl rychlejší v plnění disciplíny a nebo tým který získal více bodů. Poslední volba *Order by score* je vyhodnocena v případě, že disciplína má nastaveno jak vyhodnocení času tak i vyhodnocení počtu bodů. Její nastavení ovlivňuje zda-li je vítězný tým primárně určen podle času nebo počtu bodů. V případě že výsledky týmů shodují v prioritním parametru, jsou následně seřazeny podle zbývající možnosti.



Obrázek 5.3 Uživatelské rozhraní: soutěž

Volba *Teams* v hlavním navigačním menu odkazuje na stránku zobrazující seznam přihlášených týmů, viz následující oddíl.

Volba *generate roster*, nacházející se v pravé horní části stránky, vytvoří rozpisy pro jednotlivé disciplíny. Vedlejším efektem je ukončení možnosti úpravy definice soutěže a registrování nových týmů.

V hlavním navigačním menu se nachází volba *Announce Board*, která odkazuje na stránku zobrazující tabuli s jednotlivými hracími plochami a názvy týmů, které aktuálně plní, nebo budou plnit určitou disciplínu. *Scoreboard* odkazuje na stránku zobrazující aktuální výsledky v rámci soutěže. Tyto dva odkazy je možné použít pro zobrazení informací na dalších zařízeních jako je např. chytrý televizor.

5.4 Seznam týmů

Tato stránka zobrazuje všechny týmy přihlášené k dané soutěži viz obrázek 5.4. V pravé horní části se nachází vstupní pole, které umožňuje filtrovat zobrazené týmy podle zadaného řetězce. Výskyt zadaného řetězce se vyhledává ve jméně a popisu týmu, přičemž nezáleží na velikosti písmen. Pokud je tým již zkontrolován a označen poža-

dateli, že vyhovuje specifikacím soutěže, je zobrazen šedou barvou. Při kliknutí na tým se zobrazí stránka s podrobnostmi o týmu. Pro vytvoření nového týmu je možné využít volbu *Team create* hlavního menu, která zobrazí formulář pro zadání unikátního názvu týmu v rámci soutěže, jeho případný popis a přiřazenou kategorii. Po uložení nového týmu systém automaticky zobrazí stránku s podrobnostmi právě vytvořeného týmu.

V hlavním menu se nachází volba *Enroll*, která odkazuje na formulář pro samoobslužné přihlášení týmu do soutěže. Tohoto odkazu je možné využít pro zobrazení přihlašovacího formuláře na dalších zařízení.

Obrázek 5.4 Uživatelské rozhraní: seznam týmů

5.5 Informace o týmu

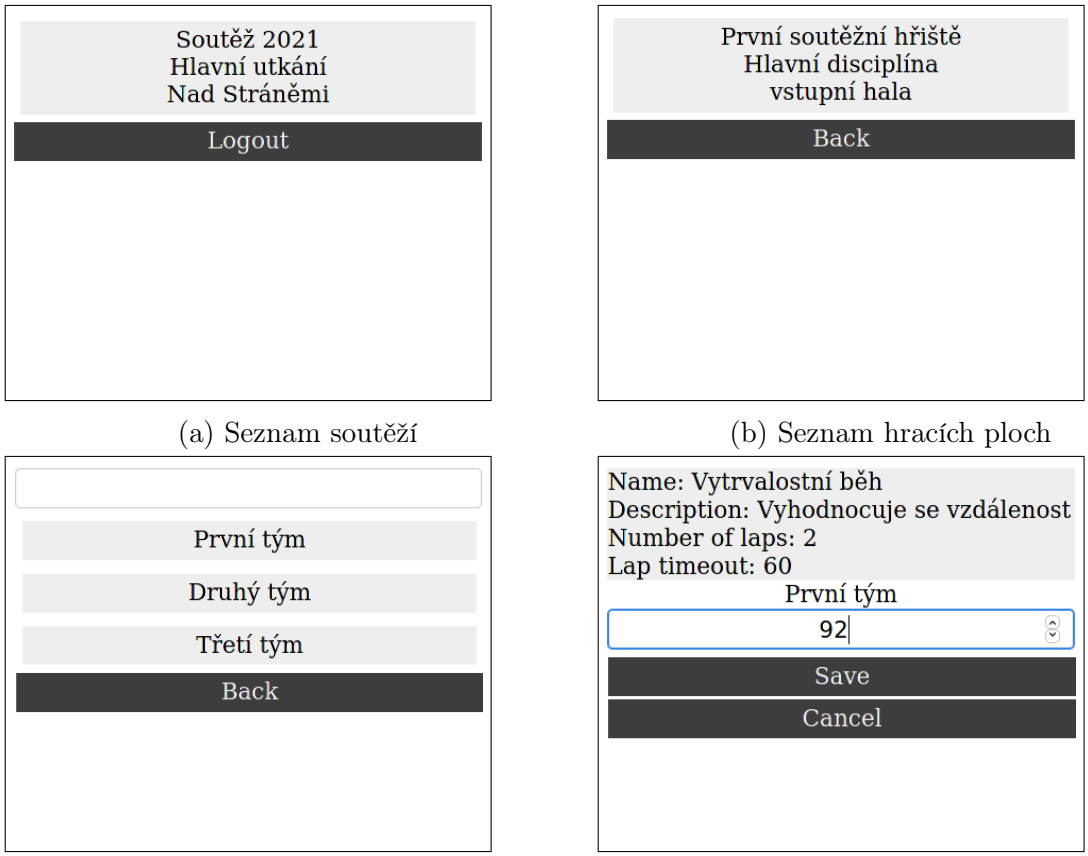
Příklad zaregistrovaného týmu lze vidět na obrázku 5.5. Na této stránce může upravit veškeré týmové informace uživatel s oprávněním *Asistent*. Lze upravovat informace o členech týmu. Také lze měnit disciplíny, kterých se tým chce účastnit pomocí volby *Disciplines edit*. Volba *Team checked* hlavního menu umožňuje označit tým jako zkontrolovaný, po jeho aktivaci systém opět zobrazí stránku se seznamem týmů.

Obrázek 5.5 Uživatelské rozhraní: informace o týmu

5.6 Aplikace rozhodčího

Tato část systému je dostupná z navigačního menu hlavní stránky, nebo zadáním adresy aplikace následované */app* do internetového prohlížeče například v mobilním zařízení. Zadávat výsledky plnění disciplíny může uživatel s oprávněním rozhodčího.

První stránka umožňuje zvolit soutěž, do které chce rozhodčí zapsat výsledek (obrázek 5.6a). Po zvolení soutěže se zobrazí seznam hracích ploch (obrázek 5.6b. Z této nabídky rozhodčí vybere hřiště, u kterého se nachází. Poslední zobrazený seznam jsou jednotlivé týmy. V horní části je vstupní pole, do kterého je možné zadat řetězec pro filtrování názvů zobrazených týmů (obrázek 5.6c). V okamžiku vybrání týmu je tento tým zobrazen na tabuli s výzvou k dostavení na danou hrací plochu. Na poslední zobrazené stránce (viz obrázek 5.6d) je možné zadat výsledek plnění disciplíny a uložit, nebo zrušit vkládání výsledku. Při zvolení jakékoliv z těchto možností dojde k odstranění jména týmu u hrací plochy z ohlašovací tabule.



(a) Seznam soutěží

(b) Seznam hracích ploch

(c) Seznam týmů

(d) Zadání výsledku

Obrázek 5.6 Aplikace rozhodčího

6 OVĚŘENÍ FUNKČNOSTI

Během vývoje informačního systému bylo k ověření funkčnosti použito automatizovaných testů a manuálně prováděných testů.

6.1 Automatizované testy

Tyto testy je možné kdykoliv opakovat, jejich hlavní výhodou je, že jsou automaticky vyhodnocovány pomocí pracovního rámce *JUnit*. Nevýhodou bývá nasazení při testování křehkých částí aplikace, jako je například uživatelské rozhraní, které prochází častou syntaktickou úpravou a v důsledku toho dochází i k úpravám testů. V rámci projektu byly psány jednotkové, integrační a funkční testy.

- *Jednotkové testy* mají za úkol otestovat jednotlivé komponenty systému, které nemají závislosti na dalších prvcích systému. Za jednotku systému můžeme pokládat třídu, nebo třídní funkci. Tento typ testu potvrzuje, že základní stavební bloky fungují správně. Je časově nejméně náročný, protože není zapotřebí sestavovat výslednou aplikaci.
- *Integrační testy* mají za úkol otestovat izolované části systému, které se skládají z více vzájemně komunikujících částí a mohou pracovat jako jeden celek. Nemusí být zaměřeny na funkcionalitu aplikační logiky, výsledek testu může znamenat, že dané komponenty spolu řádně komunikují. Tyto testy jsou časově náročnější než testy jednotkové, protože je zapotřebí sestavit testovanou část systému.
- *Funkční testy* jsou zaměřeny na aplikační logiku, testují kompletní části systému. Pro funkční testy nejsou důležité mezivýsledky, jsou testovány požadavky aplikační funkcionality a vyhodnocují se konečné výsledky. Podobně jako u integračních testů je před zahájením testování zapotřebí sestavit větší část aplikace.

Zmíněné testy nepokrývají celý projekt, byly převážně využívány jako poznávací testy [25] pro ověření funkcionality různých modulů a jejich vzájemné komunikace při vývoji informačního systému.

6.2 Manuální testy

Kontrolu shody systému s validačními kritérii je provedena pomocí manuálních testů. Následující text obsahuje postup dosažení výsledků a zhodnocení zda-li výsledky odpovídají požadovaným výsledkům validačních kritérií.

1. Vytvoření nové soutěže

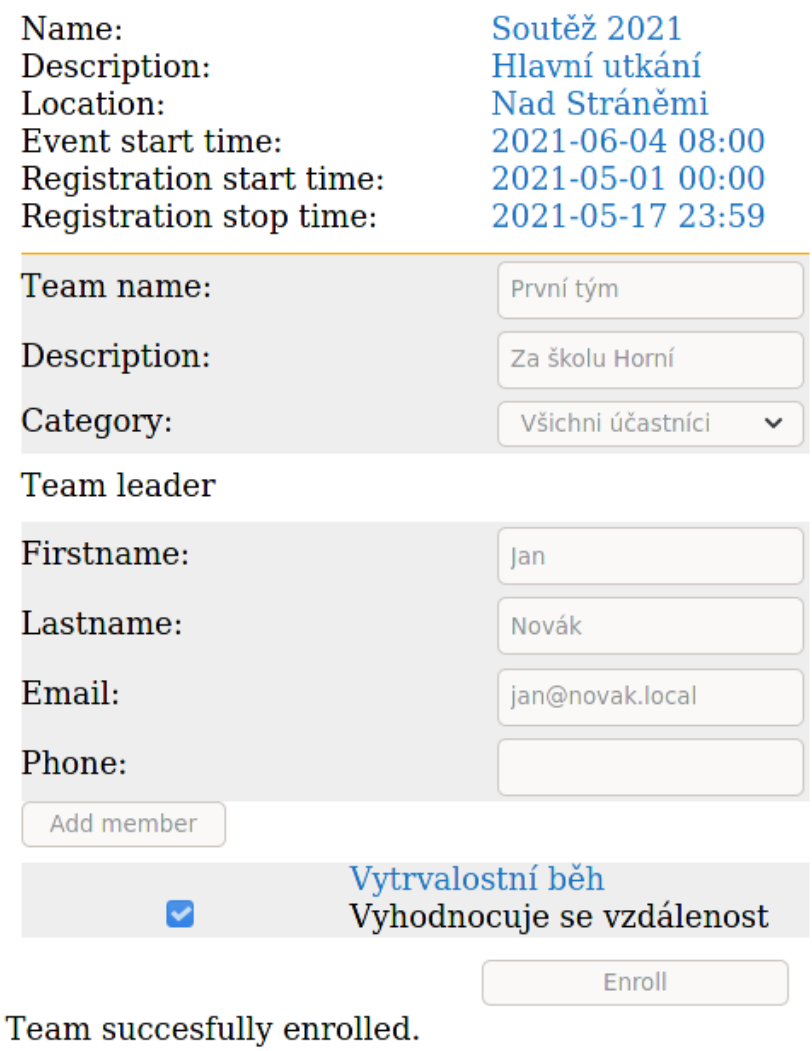
   - Postup:
     - Přihlásit do systému uživatele s rolí vedoucího.
     - Na hlavní stránce zvolit možnost vytvořit soutěž.
     - V zobrazeném formuláři vyplnit jméno soutěže, případný popis, popis místa konání, zadat čas zahájení soutěže a vyplnění registračního intervalu, což je časovém období ve kterém se mohou soutěžící sami přihlásit do soutěže pomocí registračního formuláře. Zvolit možnost uložit.
     - V nově vytvořené soutěži zvolit možnost vytvořit kategorii.
     - V zobrazeném formuláři vyplnit jméno kategorie a případný popis. Zvolit možnost uložit.
     - V nově vytvořené soutěži zvolit možnost vytvořit hřiště.
     - V zobrazeném formuláři vyplnit jméno hřiště, případný popis a umístění hřiště. Zvolit možnost uložit.
     - V nově vytvořené soutěži zvolit možnost vytvořit disciplínu.
     - V zobrazeném formuláři vyplnit jméno disciplíny, případný popis, počet kol soutěže, maximální čas trvání jednoho kola a vybrat způsob vyhodnocení z nabízených voleb. Následně přidat kategorii a hrací plochu. Na závěr zvolit možnost uložit.
   - Dosažený výsledek: Soutěž byla vytvořena a uložena.
   - Zhodnocení: Dosažený výsledek splňuje validační kritérium.

   Výsledek validačního testu odpovídá obrázku 5.3.

2. Samoobslužné přihlášení týmu do soutěže

   - Postup:
     - Nepřihlášený uživatel zobrazí stránku pro samoobslužné přihlášení do soutěže.
     - V zobrazeném formuláři zadat název týmu, případný popis, vybrat jednu z dostupných kategorií, vyplnit jméno, příjmení a kontakt na vedoucího týmu, případně přidat další členy týmu. Vybrat disciplíny, kterých se tým chce účastnit. Potvrdit přihlášení do soutěže.
   - Dosažený výsledek: Informace o týmu jsou uloženy k odpovídající soutěži.
   - Zhodnocení: Dosažený výsledek splňuje validační kritérium.

   Výsledek validačního testu je zobrazen na obrázku 6.1.

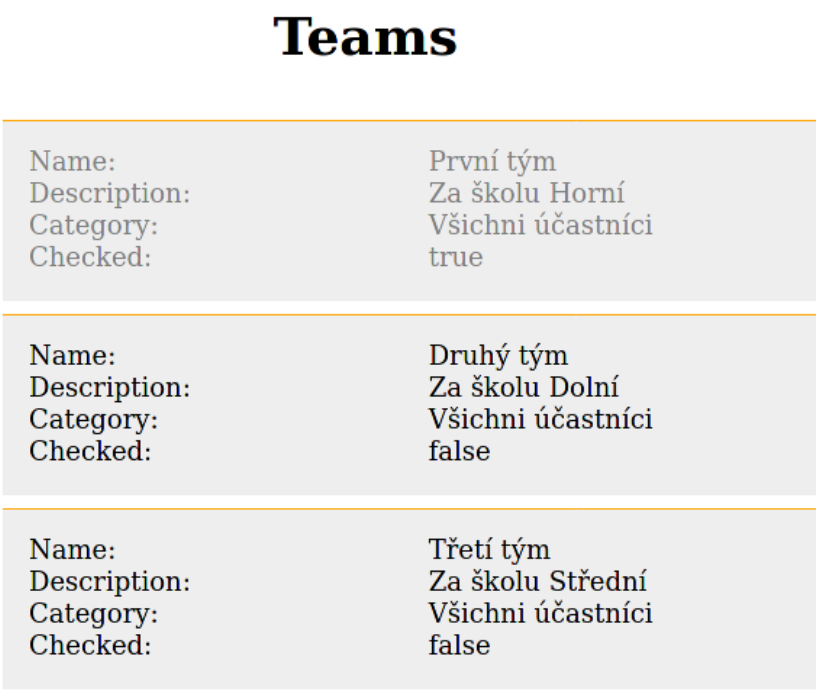


Obrázek 6.1 Validační test: Přihlášení týmu do soutěže

3. Kontrola týmu a potvrzení jeho ověření

   - Postup:
     - Přihlásit do systému uživatele s rolí asistenta.
     - Vybrat požadovanou soutěž.
     - Zvolit kontrolovaný tým. Pokud tým odpovídá požadavkům označit tým za zkontrolovaný.
   - Dosažený výsledek: Systém označil tým jako zkontrolovaný, ve výpisu týmů je zkontrolovaný tým zobrazen šedou barvou.
   - Zhodnocení: Dosažený výsledek splňuje validační kritérium.

Výsledek validačního testu je zobrazen na obrázku 6.2.

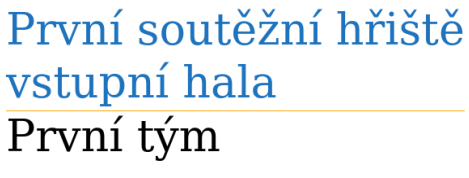


Obrázek 6.2 Validační test: Označení zkontrolovaného týmu

4. Výzva týmu k dostavení a plnění disciplíny

   - Postup:
     - Přihlásit do aplikace uživatele s rolí rozhodčího.
     - V aplikaci postupně vybrat požadovanou soutěž, hřiště a tým.
   - Dosažený výsledek: Systém zobrazí výzvu k dostavení týmu k danému hřišti.
   - Zhodnocení: Dosažený výsledek splňuje validační kritérium.

Výsledek validačního testu je zobrazen na obrázku 6.3.



Obrázek 6.3 Validační test: Výzva dostavení týmu

5. Uložení výsledků

   - Postup:
     - Přihlásit do aplikace uživatele s rolí rozhodčího.
     - V aplikaci postupně vybrat požadovanou soutěž, hřiště a tým.
     - Zadat jakého výsledku dosáhl tým při plnění disciplíny.
   - Dosažený výsledek: Systém uložil výsledek týmu v dané soutěži.
   - Zhodnocení: Dosažený výsledek splňuje validační kritérium.

   Postup validačního testu odpovídá sestavě obrázků 5.6.

6. Zobrazení aktuálního pořadí týmů

   - Postup: Zobrazit stránku s výsledky dané soutěže.
   - Dosažený výsledek: Zobrazeny aktuální výsledky soutěže.
   - Zhodnocení: Dosažený výsledek splňuje validační kritérium.

   Výsledek validačního testu je zobrazen na obrázku 6.4.

Obrázek 6.4 Validační test:
Zobrazení výsledků

ZÁVĚR

Diplomové práce se zabývá vývojem informačního systému usnadňující organizaci soutěže RoboGames a umožňující automatické vyhodnocení výsledků. Soutěž je plánována jednou ročně a probíhá v rámci vstupní haly fakulty aplikované informatiky.

První část pojednává o současném způsobu organizace soutěže a jejich obtíží při vyhodnocování výsledků. Věnuje se ukázce popisu pravidel soutěžních disciplín a jejich vyhodnocení. Definuje základní požadavky na systém vycházející z pravidel zveřejněných na webových stránkách soutěže a pozorováním vlastního průběhu soutěže z dostupných video záznamů. Zachycuje možné případy užití, které jsou popsány pomocí scénářů. Obdobné existující projekty, které byly nalezeny, je možné kategorizovat do tří skupin a to komerční systémy zaměřující se spíše na dlouhodobou podporu sportovních klubů, systémy pro podporu organizace řešící jen část rozebíraného problému a volně dostupné systémy. Pomocí analýzy podstatných jmen a sloves následované metodou štítků *CRC* byly identifikovány základní třídy vyvíjeného systému. Následně byly definovány validační kritéria pro ověření funkčnosti výsledného systému a jeho souladu se záměrem a cílem projektu. V poslední teoretické části jsou popsány vybrané technologie, pomocí kterých je projekt realizován. Výběr byl zaměřen na standardy, knihovny a pracovní rámce, které mají širokou uživatelskou základnu a představují moderní trendy v oblasti vývoje softwarových aplikací.

Druhá praktická část představuje celkový pohled na vytvořený informační systém, zabývá se popisem návrhu třívrstvé architektury z perspektivy nejvyšší úrovně a následně prochází vytvořenou programovou strukturu s ambicí objasnit základní tok objektů a požadavků procházejících aplikací od prezentační vrstvy po datové úložiště. Nejvýraznějším rysem je rozdělení projektu pomocí pomyslných hranic na tři části. První část má na starost aplikační logiku, tvoří hlavní jádro celého systému. Druhá část se stará o persistentní uložení dat a správu schéma relační databáze. Poslední třetí část má odpovědnost za komunikaci s prezentační vrstvou pomocí komunikačního protokolu *HTTP* a přenášených dat v podobě *JSON* formátu. V samostatných částech je přiblížen princip persistentního ukládání dat do databáze a způsob zabezpečení přístupu k datům. Pro snadnější orientaci byl napsán průvodce grafickým uživatelským rozhraním. V posledním oddíle praktické části byl popsán způsob testování pomocí automatizovaných a manuálních testů, podkladem pro manuální testování byly využity dříve definované validační kritéria.

Výsledný informační systém realizuje veškeré vydefinované funkční a nefunkční požadavky specifikované v teoretické části práce. Jeho následná implementace vycházela z uvedených případů užití, jejich scénářů a identifikace základních objektových tříd.

Vývoj tohoto informačního systému byl pro mě výzvou, protože se jednalo o projekt využívající nové technologie, se kterými jsem neměl předchozí zkušenost. Během vývoje jsem se nejednou dostal do zdánlivě bezradné situace, jejíž vyřešení mi zabralo několika násobně více času, než jsem původně předpokládal. Příkladem časově náročného úkolu mohu uvést nalezení vhodného modelu pro realizaci persistentního uložení objektů.

Budoucí rozvoj systému by mohl implementovat pokročilejší funkce správy týmů, například možnost rozdělení týmů do více skupin v rámci bojové disciplíny v případě většího množství přihlášených týmů, nebo naopak možnost seskupit věkové kategorie dané disciplíny v případě malého počtu přihlášených týmů. Dále by mohlo dojít ke zlepšení uživatelské přívětivosti grafického prostředí a přidání statistických informací právě probíhající soutěže, například množství odehraných kol jednotlivých disciplín.

Práce na tomto projektu mi rozšířila rozhled v oblasti vývoje aplikací a přivedla mě k dalším tématickým literárním zdrojům, které bych rád podrobněji prostudoval, příkladem může být návrh řízený doménou (*DDD*).

SEZNAM POUŽITÉ LITERATURY

[1] *Celostátní soutěž robotů na UTB ve Zlíně* [online]. [cit. 2020-12-12]. Dostupné z: https://robogames.utb.cz

[2] Pravidla. *Celostátní soutěž robotů na UTB ve Zlíně* [online]. [cit. 2020-12-12]. Dostupné z: https://robogames.utb.cz/pravidla

[3] ARLOW Jim, NEUSTADT Ila. *UML2 a unifikovaný proces vývoje aplikací*. Brno: Computer Press, 2008. ISBN 978-80-251-1503-9

[4] *Rozpisy zápasů* [online]. [cit. 2020-10-12]. Dostupné z: https://www.rozpisyzapasu.cz

[5] *SportChef* [online]. [cit. 2020-10-12]. Dostupné z: https://github.com/friedhof/sportchef

[6] ZAKHOUR Sharon. *Java 6 Výukový kurz*. Brno: Computer Press, 2007. ISBN 978-80-251-1575-6

[7] *Oracle Java* [online]. [cit. 2020-12-12]. Dostupné z: https://www.oracle.com/java

[8] HAVELKA Arnošt, PECINOVSKÝ Rudolf. *JUnit 5 - Jednotkové testování na platformě Java*. Praha: Grada Publishing, 2018. ISBN 978-80-271-0733-9

[9] *Spring* [online]. [cit. 2021-02-10]. Dostupné z: https://spring.io

[10] *Spring Framework* [online]. [cit. 2021-02-10]. Dostupné z: https://spring.io/projects/spring-framework

[11] *Spring Data JDBC* [online]. [cit. 2021-02-10]. Dostupné z: https://spring.io/projects/spring-data-jdbc

[12] *Spring Security* [online]. [cit. 2021-02-10]. Dostupné z: https://spring.io/projects/spring-security

[13] *Thymeleaf* [online]. [cit. 2021-02-05]. Dostupné z: https://www.thymeleaf.org

[14] *Project Lombok* [online]. [cit. 2021-02-05]. Dostupné z: https://projectlombok.org

[15] *Gradle Build Tool* [online]. [cit. 2021-04-08]. Dostupné z: https://gradle.org

[16] CASTRO Elizabeth, HYSLOP Bruce. *HTML5 a CSS3 Názorný průvodce tvorbou WWW stránek*. Brno: Computer Press, 2012. ISBN 978-80-251-3733-8

[17] ZAKAS C. Nicholas. *JavaScript pro webové vývojáře*. Brno: Computer Press, 2009. ISBN 978-80-251-2509-0

[18] *JSON* [online]. [cit. 2021-04-08]. Dostupné z: https://www.json.org/json-en.html

[19] DOSTÁLEK Libor, KABELOVÁ Alena. *Velký průvodce protokoly TCP/IP a systémem DNS*. Brno: Computer Press, 2012. ISBN 978-80-251-2236-5

[20] *HTTP* [online]. [cit. 2021-04-08]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP

[21] *Liquibase Open Source Version Control for Your Database* [online]. [cit. 2021-04-08]. Dostupné z: https://www.liquibase.org

[22] *H2 Database Engine* [online]. [cit. 2021-04-08]. Dostupné z: https://www.h2database.com

[23] MOMJIAN Bruce. *PostgreSQL Praktický průvodce*. Brno: Computer Press, 2003. ISBN 80-7226-954-2

[24] PECINOVSKÝ Rudolf. *Návrhové vzory*. Brno: Computer Press, 2007. ISBN 978-80-251-1582-4

[25] MARTIN C. Robert. *Čistý kód*. Brno: Computer Press, 2009. ISBN 978-80-251-2285-3

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

UTB	Univerzita Tomáše Bati ve Zlíně
CMS	Competition Management System
CRC	Class Responsibilities Collaborators
JVM	Java Virtual Machine
JRE	Java Runtime Environment
JDK	Java Development Kit
JDBC	Java Database Connectivity
DBMS	Database Management System
YAML	YAML Ain't Markup Language
DDD	Domain Driven Design
API	Application Programming Interface
IoC	Inversion of Control
DI	Dependency Injection
URL	Uniform Resource Locator
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
JSON	Javascript Object Notation
HTTP	Hypertext Transfer Protocol
MVC	Model View Controller

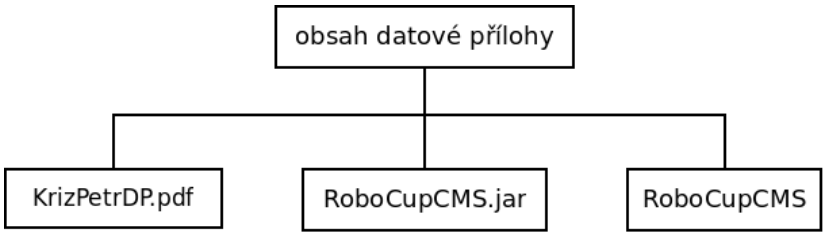
SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I. OBSAH DATOVÉ PŘÍLOHY

Struktura datové přílohy je znázorněna na obrázku 1.1.



Obrázek 1.1 Struktura datové přílohy

Význam jednotlivých položek:

- *KrizPetrDP.pdf* - soubor s kompletním textem diplomové práce ve formátu *PDF*.
- *RoboCupCMS.jar* - spustitelný *Java* archiv obsahující informační systém.
- *RoboCupCMS* - adresář obsahující veškeré zdrojové soubory projektu.

PŘÍLOHA P II. ZPROVOZNĚNÍ SYSTÉMU

- **Konfigurace**
  Ve výchozím nastavení je používán databázový systém *H2* a data jsou ukládána v dočasné paměti počítače. Po zastavení systému dojde ke ztrátě veškerých vložených informací. Změnu tohoto nastavení je možné provést v rámci *Java* archivu v souboru */BOOT-INF/classes/application.properties* (viz obrázek 2.1). Řádky začínající znakem # jsou považovány za komentáře. Na prvním řádku se nachází konfigurace databáze *H2*.

```
spring.datasource.url=jdbc:h2:mem:inmemory

#spring.datasource.url=jdbc:postgresql://localhost:5432/postgres
#spring.datasource.username=postgres
#spring.datasource.password=postgres

server.port=8080
```

Obrázek 2.1 Konfigurační soubor *application.properties*

  Připojení k databázi *PostgreSQL* je možné provést zakomentováním prvního řádku a odkomentováním následujících tří řádků, přičemž v parametru *spring.datasource.url* je možné nastavit internetovou adresu, port a název databáze. Parametr *spring.datasource.username* odkazuje na název uživatele přistupujícího k databázi a parametr *spring.datasource.password* jeho přihlašovací heslo.

  Posledním uvedeným parametrem *server.port* je možné nastavit číslo portu, na kterém aplikace naslouchá příchozím požadavkům od klientů. Změny provedené v konfiguračním souboru se projeví při následujícím spuštění informačního systému.

- **Spuštění**
  Pro spuštění projektu je zapotřebí mít nainstalován Java Runtime Environment verze 11. Samotné spuštění aplikace je možné provést pomocí příkazu: „java -jar RoboCupCMS.jar“.

- **Přihlášení do systému**
  Po spuštění informačního systému je zapotřebí zadat jeho adresu a případně port do internetového prohlížeče a potvrdit. Zobrazí se formulář vyžadující přihlášení. Výchozí uživatelské jméno je „admin“ a heslo „nimda“.